

شبکه های کامپیوتری

نویسنده

اندرواس.تنن باوم

فهرست مطالب

۱۱	مقدمه	۱۱
۱۲	۱-۱ کاربردهای شبکه های کامپیوتری	۱۲
۱۲	۱-۱-۱ کاربردهای تجاری	۱۲
۱۵	۲-۱-۱ کاربردهای خانگی	۱۵
۱۸	۳-۱-۱ کاربران سیار	۱۸
۲۰	۴-۱-۱ تبعات اجتماعی	۲۰
۲۲	۲-۱ سخت افزار شبکه	۲۲
۲۳	۱-۲-۱ شبکه های محلی	۲۳
۲۵	۲-۲-۱ شبکه های شهری	۲۵
۲۶	۳-۲-۱ شبکه های گسترده	۲۶
۲۷	۴-۲-۱ شبکه های بیسیم	۲۷
۲۹	۵-۲-۱ شبکه های خانگی	۲۹
۳۱	۶-۲-۱ شبکه های	۳۱
۳۲	۳-۱ نرم افزار شبکه	۳۲
۳۲	۱-۳-۱ سلسله مراتب پروتکل ها	۳۲
۳۶	۲-۳-۱ ملاحظات در طراحی لایه ها	۳۶
۳۷	۳-۳-۱ سرویسهای اتصال-گرا و غیرمتصل	۳۷
۳۹	۴-۳-۱ عملکردهای پایه سرویس	۳۹
۴۱	۵-۳-۱ رابطه سرویس و پروتکل	۴۱
۴۲	۴-۱ مدل های مرجع	۴۲
۴۲	۱-۴-۱ مدل مرجع OSI	۴۲
۴۵	۲-۴-۱ مدل مرجع TCP/IP	۴۵
۴۷	۳-۴-۱ مقایسه مدل های OSI و TCP/IP	۴۷
۴۹	۴-۴-۱ نگاهی انتقادی به مدل OSI و پروتکل های آن	۴۹
۵۱	۵-۴-۱ نگاهی انتقادی به مدل TCP/IP	۵۱
۵۲	۵-۱ شبکه های نمونه	۵۲
۵۲	۱-۵-۱ اینترنت	۵۲
۵۲	۲-۵-۱ شبکه های اتصال-گرا: X.25 ، Frame Relay	۵۲
۶۰	و ATM	۶۰
۶۵	۳-۵-۱ اینترنت	۶۵
۶۷	۴-۵-۱ شبکه های محلی بیسیم: 802.11	۶۷
۷۰	۶-۱ استانداردهای شبکه	۷۰
۷۰	۱-۶-۱ مراجع مسئول استانداردهای مخابرات	۷۰
۷۲	۲-۶-۱ مراجع مسئول استانداردهای بین المللی	۷۲
۷۴	۳-۶-۱ مراجع مسئول استانداردهای اینترنت	۷۴
۷۵	۷-۱ واحدهای اندازه گیری	۷۵
۷۶	۸-۱ طرح کلی مباحث کتاب	۷۶
۷۸	۹-۱ خلاصه	۷۸
۷۹	مسائل	۷۹
۸۳	۲ لایه فیزیکی	۸۳
۸۳	۱-۲ مبانی نظری مخابرات داده	۸۳
۸۳	۱-۱-۲ آنالیز فوریه	۸۳
۸۴	۲-۱-۲ محدودیت پهنای باند	۸۴
۸۶	۳-۱-۲ حداکثر نرخ داده در یک کانال	۸۶
۸۷	۲-۲ رسانه انتقال هدایت پذیر	۸۷
۸۷	۱-۲-۲ رسانه مغناطیسی	۸۷
۸۸	۲-۲-۲ زوج تابیده	۸۸
۸۹	۳-۲-۲ کابل کواکسیال	۸۹
۹۰	۴-۲-۲ فیبر نوری	۹۰
۹۶	۳-۲ انتقال بیسیم	۹۶
۹۶	۱-۳-۲ طیف الکترومغناطیس	۹۶
۹۹	۲-۳-۲ مخابرات رادیویی	۹۹
۱۰۰	۳-۳-۲ مخابرات مایکروویو	۱۰۰
۱۰۲	۴-۳-۲ امواج مادون قرمز و میلیمتری	۱۰۲
۱۰۳	۵-۳-۲ مخابرات امواج نوری	۱۰۳
۱۰۳	۴-۲ ماهواره های مخابراتی	۱۰۳
۱۰۵	۱-۴-۲ ماهواره های زمین ثابت	۱۰۵
۱۰۸	۲-۴-۲ ماهواره های مدار متوسط	۱۰۸

۴-۳ پروتکل های پنجره لغزنده	۱۹۶
۱-۴-۳ پروتکل پنجره لغزنده ایجیتی	۱۹۸
۲-۴-۳ پروتکل N تا به عقب برگردد	۲۰۱
۳-۴-۳ پروتکل تکرار انتخابی	۲۰۸
۵-۳ ارزیابی پروتکل ها	۲۱۴
۱-۵-۳ مدل ماشین حالت محدود	۲۱۴
۲-۵-۳ مدل شبکه پتری	۲۱۷
۶-۳ چند نمونه از پروتکل های لینک داده	۲۱۹
۱-۶-۳ HDLC - کنترل سطح بالای لینک داده ..	۲۱۹
۲-۶-۳ لایه پیوند داده در اینترنت	۲۲۲
۷-۳ خلاصه	۲۲۷
مسائل	۲۲۷

۴ زیر لایه دسترسی به لایه انتقال .. ۲۳۱

۱-۴ مسئله تخصیص کانال	۲۳۲
۲-۴ پروتکل های دسترسی چندگانه	۲۳۵
۱-۲-۴ ALOHA	۲۳۵
۲-۲-۴ پروتکل های دسترسی چندگانه با قابلیت شنود	
سیگنال حامل (CSMA)	۲۳۹
۳-۲-۴ پروتکل های بدون تصادم	۲۴۲
۴-۲-۴ پروتکل های با رقابت محدود	۲۴۵
۵-۲-۴ پروتکل های دسترسی چندگانه مبتنی بر تقسیم	
طول موج	۲۴۹
۶-۲-۴ پروتکل های بی سیم برای شبکه محلی	۲۵۲
۳-۴ اترنت	۲۵۵
۱-۳-۴ کابل کشی اترنت	۲۵۶
۲-۳-۴ کدینگ منجستر	۲۵۹
۳-۳-۴ پروتکل زیر لایه MAC در اترنت	۲۶۰
۴-۳-۴ الگوریتم عقب گرد نمایی	۲۶۲
۵-۳-۴ کارائی (بازده) اترنت	۲۶۳
۶-۳-۴ اترنت مبتنی بر سوئیچ	۲۶۵
۷-۳-۴ اترنت سریع	۲۶۷
۸-۳-۴ اترنت گیگابیت	۲۷۱
۹-۳-۴ IEEE 802.2: کنترل منطقی لینک	۲۷۵
۱۰-۳-۴ نگاهی به گذشته اترنت	۲۷۶
۴-۴ شبکه های محلی بی سیم	۲۷۶
۱-۴-۴ پشته پروتکلی 802.11	۲۷۷

۳-۴-۲ ماهواره های مدار پائین	۱۰۸
۴-۴-۲ ماهواره یا فیبر؟	۱۱۱
۵-۲ شبکه تلفن عمومی	۱۱۲
۱-۵-۲ ساختار سیستم تلفن	۱۱۲
۲-۵-۲ تلفن و سیاست	۱۱۵
۳-۵-۲ مدار های پایانی: مودم، ADSL، و بیسیم	
۴-۵-۲ ترانک ها و مالتی پلکس کردن	۱۲۹
۵-۵-۲ سوئیچینگ	۱۳۷
۶-۲ شبکه تلفن همراه	۱۴۲
۱-۶-۲ تلفن های همراه نسل اول: صدای آنالوگ	
۲-۶-۲ تلفن های همراه نسل دوم: صدای	
دیجیتال	۱۴۷
۳-۶-۲ تلفن های همراه نسل سوم: صدای دیجیتال و	
داده	۱۵۵
۷-۲ تلویزیون کابلی	۱۵۸
۱-۷-۲ تلویزیون با آنتن مرکزی	۱۵۸
۲-۷-۲ اینترنت کابلی	۱۵۸
۳-۷-۲ تخصیص طیف فرکانسی	۱۶۰
۴-۷-۲ مودم های کابلی	۱۶۱
۵-۷-۲ مودم کابلی یا ADSL؟	۱۶۳
۸-۲ خلاصه	۱۶۴
مسائل	۱۶۵

۳ لایه پیوند داده .. ۱۷۱

۱-۳ ملاحظات طراحی لایه پیوند داده	۱۷۱
۱-۱-۳ سرویس هایی که به لایه شبکه داده	
می شود	۱۷۲
۲-۱-۳ فریم بندی	۱۷۴
۳-۱-۳ کنترل خطا	۱۷۷
۴-۱-۳ کنترل جریان	۱۷۸
۲-۳ کشف و تصحیح خطا	۱۷۹
۱-۲-۳ گدهای تصحیح خطا	۱۷۹
۲-۲-۳ گدهای کشف خطا	۱۸۱
۳-۳ چند پروتکل ساده لینک داده	۱۸۵
۱-۳-۳ پروتکل یکطرفه نامقید	۱۸۹
۲-۳-۳ پروتکل توقف-انتظار یکطرفه	۱۹۱
۳-۳-۳ پروتکل یکطرفه برای کانال های نویز دار	
.....	۱۹۳

۳۳۴	دیتاگرام	۲۷۸	۲-۴-۴ لایه فیزیکی در 802.11
۳۳۶	۲-۵ الگوریتمهای مسیریابی	۲۸۰	۳-۴-۴ پروتکل زیر لایه MAC در 802.11
۳۳۸	۱-۲-۵ اصل بهیگی	۲۸۴	۴-۴-۴ ساختار فریم 802.11
۳۳۹	۲-۲-۵ مسیریابی مبتنی بر کوتاهترین مسیر	۲۸۶	۵-۴-۴ خدمات
۳۴۱	۳-۲-۵ الگوریتم سیل آسا (Flooding)	۲۸۷	۵-۴ بی سیم با باند گسترده
۳۴۳	۴-۲-۵ مسیریابی بردار فاصله	۲۸۸	۵-۴ مقایسه 802.11 با 802.16
۳۴۷	۵-۲-۵ مسیریابی حالت لینک	۲۸۹	۲-۵-۴ پشته پروتکلی 802.16
۳۵۳	۶-۲-۵ مسیریابی سلسله مراتبی	۲۹۰	۳-۵-۴ لایه فیزیکی در 802.16
۳۵۵	۷-۲-۵ مسیریابی فراگیر (Broadcast Routing)	۲۹۲	۴-۵-۴ پروتکل زیر لایه MAC در 802.16
۳۵۷	۸-۲-۵ مسیریابی چندپختی	۲۹۴	۵-۵-۴ ساختار فریم در 802.16
۳۵۹	۹-۲-۵ مسیریابی برای ماشینهای متحرک	۲۹۵	۶-۴ بلوتوث (Bluetooth)
۳۶۳	۱۰-۲-۵ مسیریابی در شبکه های ویژه	۲۹۶	۱-۶-۴ معماری بلوتوث
۳۶۳	۱۱-۲-۵ جستجوی گره در شبکه های همتابه	۲۹۷	۲-۶-۴ کاربردهای بلوتوث
۳۶۸	همتابه	۲۹۸	۳-۶-۴ پشته پروتکلی بلوتوث
۳۷۳	۳-۵ الگوریتمهای کنترل ازدحام	۳۰۰	۴-۶-۴ لایه رادیویی در بلوتوث
۳۷۵	۱-۳-۵ اصول کلی در کنترل جریان	۳۰۰	۵-۶-۴ لایه باند پایه در بلوتوث
۳۷۷	۲-۳-۵ سیاستهای پیشگیری از ازدحام	۳۰۱	۶-۶-۴ لایه L2CAP در بلوتوث
۳۷۷	۳-۳-۵ کنترل ازدحام در زیر شبکه های مدار مجازی	۳۰۱	۷-۶-۴ ساختار فریم در بلوتوث
۳۷۸	۴-۳-۵ کنترل ازدحام در زیر شبکه های دیتاگرام	۳۰۳	۷-۴ هدایت در سطح لایه پیوند داده ها
۳۸۰	۵-۳-۵ دور ریختن بار (Load Shedding)	۳۰۵	۱-۷-۴ پلهائی از 802.x به 802.y
۳۸۳	۶-۳-۵ کنترل لرزش (Jitter Control)	۳۰۷	۲-۷-۴ بهم بندی شبکه ها به صورت محلی
۳۸۵	۴-۵ کیفیت خدمات (Quality of Service)	۳۰۹	۳-۷-۴ پلهای مبتنی بر درخت پوشا
۳۸۶	۱-۴-۵ نیازها	۳۱۰	۴-۷-۴ پلهای راه دور (Remote Bridges)
۳۸۶	۲-۴-۵ راهکارهای دستیابی به کیفیت خوب خدمات	۳۱۰	۵-۷-۴ تکرار کننده، هاب، پل، سوئیچ، مسیریاب و دروازه
۳۸۷	۳-۴-۵ خدمات مجتمع (Integrated Services)	۳۱۴	۶-۷-۴ شبکه های محلی مجازی (Virtual LANs)
۴۰۰	۴-۴-۵ خدمات متمایز	۳۲۲	۸-۴ خلاصه
۴۰۲	۵-۴-۵ سوئیچ برچسب و MPLS	۳۲۴	مسائل
۴۰۵	۵-۵ بهم بندی شبکه ها (Internetworking)		
۴۰۹	۱-۵-۵ شبکه ها از چه دیدگاهی متفاوتند؟		
۴۱۰	۲-۵-۵ چگونگی اتصال شبکه ها به یکدیگر		
۴۱۳	۳-۵-۵ مدارات مجازی الحاق شده		
۴۱۴	۴-۵-۵ بهم بندی شبکه های بدون اتصال		
۴۱۶	۵-۵-۵ ایجاد تونل (Tunneling)		
۴۱۷	۶-۵-۵ مسیریابی بین شبکه های بهم متصل		
۴۱۸	۷-۵-۵ قطعه قطعه سازی بسته ها		

۵

لایه شبکه

۳۲۹	۱-۵ مسائل طراحی لایه شبکه
۳۲۹	۱-۱-۵ هدایت (سوئیچینگ) بسته به روش «ذخیره و هدایت»
۳۳۰	۲-۱-۵ خدمات ارائه شده برای لایه انتقال
۳۳۱	۳-۱-۵ پیاده سازی خدمات بی اتصال
۳۳۳	۴-۱-۵ پیاده سازی خدمات اتصال گرا
۵-۱-۵	مقایسه زیر شبکه های مدار مجازی و

فهرست مطالب ۹

۵۲۶	۳-۴-۶ پروتکل انتقال بی درنگ
۵۳۰	۵-۶ پروتکل های لایه انتقال در اینترنت: TCP
۵۳۰	۱-۵-۶ مقدمه ای بر TCP
۵۳۱	۲-۵-۶ مدل خدمات TCP
۵۳۳	۳-۵-۶ پروتکل TCP
۵۳۴	۴-۵-۶ سرآیند قطعه TCP
۵۳۸	۵-۵-۶ برقراری اتصال TCP
۵۳۹	۶-۵-۶ خاتمه دادن به اتصال TCP
۵۴۰	۷-۵-۶ مدل سازی فرآیند مدیریت اتصال در TCP
۵۴۲	۸-۵-۶ سیاست های انتقال در TCP
۵۴۶	۹-۵-۶ کنترل ازدحام در TCP
۵۴۹	۱۰-۵-۶ مدیریت تایمرها در TCP
۵۵۲	۱۱-۵-۶ TCP و UDP بی سیم
۵۵۴	۱۲-۵-۶ TCP تراکنشی (Transactional TCP)
۵۵۶	۶-۶ مسائل مرتبط با کارایی
۵۵۷	۱-۶ مشکلات کارایی در شبکه های کامپیوتری
۵۵۹	۲-۶ اندازه گیری کارایی شبکه
۵۶۱	۳-۶ طراحی سیستم برای کارایی بهتر
۵۶۵	۴-۶ پردازش سریع TPDU
۵۶۹	۵-۶ پروتکل هایی برای شبکه های گسترده
۵۷۳	۷-۶ خلاصه
۵۷۳	مسائل

۷ لایه کاربرد ۵۷۹

۵۷۹	۱-۷ سیستم نام ناحیه DNS
۵۸۰	۱-۱-۷ فضای نام DNS
۵۸۲	۲-۱-۷ رکوردهای منابع
۵۸۵	۳-۱-۷ سرویس دهنده نام
۵۸۷	۲-۷ پست الکترونیک
۵۸۸	۱-۲-۷ معماری و سرویسها
۵۸۹	۲-۲-۷ عامل کاربر
۵۹۲	۳-۲-۷ فرمت پیامها
۵۹۸	۴-۲-۷ انتقال پیام
۶۰۱	۵-۲-۷ تحویل نهایی
۶۰۶	۳-۷ تارنمای جهانی - وب
۶۰۷	۱-۳-۷ بررسی ساختاری
۶۲۲	۲-۳-۷ سندهای وب استاتیک

۴۲۲	۶-۵ لایه شبکه در اینترنت
۴۲۴	۱-۶ پروتکل IP
۴۲۸	۲-۶ آدرسهای IP
۴۴۱	۳-۶ پروتکل های کنترل اینترنت
۴۶۵	۴-۶ OSPF: پروتکل مسیریابی برای دروازه های
۴۴۷	درونی
۵۶۵	۵-۶ BGP: پروتکل مسیریابی برای دروازه
۴۵۳	خارجی
۴۵۵	۶-۶ ارسال چندپخشی در اینترنت
۴۵۶	۷-۶ IP متحرک (Mobile IP)
۴۵۸	۸-۶ IPv6
۴۶۸	۷-۵ خلاصه
۴۶۸	مسائل

۶ لایه انتقال ۴۷۵

۴۷۵	۱-۶ خدمات انتقال (The Transport Service)
۴۷۵	۱-۱-۶ خدمات ارائه شده به لایه های بالاتر
۴۷۷	۲-۱-۶ عملکردهای اولیه و توابع بنیانی لایه انتقال
۴۸۱	۳-۱-۶ سوکت های برکلی (Berkeley Socket)
۴۸۱	۴-۱-۶ مثالی از برنامه نویسی سوکت: یک
۴۸۲	سرویس دهنده اینترنتی فایل
۴۸۷	۲-۶ مؤلفه های هر پروتکل انتقال
۴۸۸	۱-۲-۶ آدرس دهی
۴۹۱	۲-۲-۶ برقراری اتصال
۴۹۷	۳-۲-۶ خاتمه اتصال
۵۰۱	۴-۲-۶ کنترل جریان و بافر سازی
۵۰۶	۵-۲-۶ مالتی پلکسینگ (تسهیم)
۵۰۷	۶-۲-۶ جبران از کارافتادگی (Crash Recovery)
۵۱۰	۳-۶ یک پروتکل ساده انتقال
۵۱۰	۱-۳-۶ توابع اولیه ارائه خدمات در مثال فوق
۵۱۲	۲-۳-۶ واحد انتقال در مثال فوق
۵۱۲	۳-۳-۶ بررسی مثال فوق از دید «ماشین حالت
۵۱۹	محدود»
۵۲۱	۴-۶ پروتکل های لایه انتقال در اینترنت: UDP
۵۲۲	۱-۴-۶ مقدمه ای بر UDP
۵۲۳	۲-۴-۶ فراخوانی پروسیجرهای راه دور (RPC)

۴۴۸ حملہ روز تولد (The birthday Attack) . ۴۴۷	۳۳۷ سندهای وب دینامیک ۶۳۴
۵۸ مدیریت کلیدهای عمومی ۷۵۰	۴۳۷ پروتکل انتقال اُتر متن - HTTP ۶۴۱
۱۵۸ گواهینامه ها (Certificates) ۷۵۰	۵۳۷ بهبود کارایی ۶۴۵
۲۵۸ X.509 ۷۵۲	۶۳۷ وب بیسیم ۶۵۱
۳۵۸ زیر ساخت کلید عمومی ۷۵۳	۴۷ چند رسانه ای ۶۶۱
۶۸ امنیت ارتباطات ۷۵۷	۱۴۷ مقدمه ای بر صدای دیجیتال ۶۶۲
۱۶۸ IPsec ۷۵۷	۲۴۷ فشرده سازی صدا ۶۶۳
۲۶۸ دیوارهای آتش (Firewalls) ۷۶۲	۳۴۷ صدای جویباری ۶۶۵
۳۶۸ شبکه های خصوصی مجازی (VPN) ۷۶۵	۴۴۷ رادیوی اینترنتی ۶۶۹
۴۶۸ امنیت شبکه های بی سیم ۷۶۶	۵۴۷ صدا روی IP ۶۷۱
۷۸ پروتکل های احراز هویت ۷۷۱	۶۴۷ مقدمه ای بر ویدئو ۶۷۷
۱۷۸ احراز هویت بر اساس کلید مشترک و	۷۴۷ فشرده سازی ویدئو ۶۸۱
سری ۷۷۲	۸۴۷ پخش فیلم بر حسب تقاضا ۶۸۷
۲۷۸ ایجاد کلید مشترک: مبادله کلید به روش	۹۴۷ ستون فقرات چند پخش - Mbone ۶۹۳
«دیفی-هلمن» ۷۷۷	۵۷ خلاصه ۶۹۷
۳۷۸ احراز هویت توسط مرکز توزیع کلید ۷۷۹	مسائل ۶۹۷
۴۷۸ احراز هویت با استفاده از Kerberos ۷۸۲	
۵۷۸ احراز هویت با استفاده از رمزنگاری با کلید	
عمومی ۷۸۵	
۸۸ امنیت نامه های الکترونیکی ۷۸۶	
۱۸۸ PGP (Pretty Good Privacy) ۷۸۶	
۲۸۸ PEM (Privacy Enhanced Mail) ۷۹۱	
۳۸۸ S/MIME ۷۹۱	
۹۸ امنیت وب ۷۹۲	
۱۹۸ تهدیدها ۷۹۲	
۲۹۸ نامگذاری مطمئن ۷۹۳	
۳۹۸ SSL: لایه سوکنهای امن ۸۰۱	
۴۹۸ امنیت کدهای متحرک ۸۰۵	
۱۰۸ زمینه ها و پی آمدهای اجتماعی ۸۰۸	
۱۰۸ حریم خصوصی افراد (Privacy) ۸۰۸	
۲۰۸ آزادی بیان ۸۱۱	
۳۰۸ مالکیت معنوی (Copyright) ۸۱۳	
۱۱۸ خلاصه ۸۱۶	
مسائل ۸۱۷	
	امنیت شبکه ۷۰۳
	۱۸ رمزنگاری ۷۰۶
	۱۰۸ مقدمه ای بر رمزنگاری ۷۰۷
	۲۰۸ رمزهای جانشینی (Substitution Cipher) ۷۱۰
	۳۰۸ رمزنگاری جایگشتی (Transposition) ۷۱۲
	۴۰۸ رمز One-Time Pads ۷۱۳
	۵۰۸ دو اصل اساسی در رمزنگاری ۷۱۸
	۲۸ الگوریتم های رمزنگاری با کلید متقارن ۷۲۱
	۱۰۸ رمزنگاری DES ۷۲۲
	۲۰۸ استاندارد پیشرفته رمزنگاری: AES ۷۲۵
	۳۰۸ حالات رمز (Cipher Modes) ۷۲۹
	۴۰۸ رمزهای دیگر ۷۳۵
	۵۰۸ تحلیل رمز (رمز شکنی) ۷۳۵
	۳۸ الگوریتم های کلید عمومی (Public Key) ۷۳۶
	۱۰۳۸ RSA ۷۳۷
	۲۰۳۸ الگوریتم های کلید عمومی دیگر ۷۳۹
	۴۸ امضاهای دیجیتالی ۷۴۰
	۱۰۴۸ امضاهای دیجیتالی با کلید متقارن ۷۴۱
	۲۰۴۸ امضاهای با کلید عمومی ۷۴۲
	۳۰۴۸ خلاصه پیامها (Message Digests) ۷۴۳
واژه نامه ۸۲۳	
محتویات دیسک فشرده همراه کتاب ۸۲۹	

مقدمه

هر یک از سه قرن گذشته را با یک تکنولوژی خاص بعنوان نماد آن قرن می‌شناسیم. قرن هیجدهم عصر سیستمهای بزرگ مکانیکی و انقلاب صنعتی بود، و قرن نوزدهم عصر بخار. تکنولوژی کلیدی قرن بیستم نیز جمع‌آوری، پردازش و توزیع اطلاعات بود. شبکه‌های گسترده و بین‌المللی تلفن، اختراع رادیو و تلویزیون، تولد و گسترش باورنکردنی صنعت کامپیوتر، و پرتاب ماهواره‌های مخابراتی از نمادهای این عصر هستند.

با رشد سریع تکنولوژیهای جمع‌آوری، پردازش و توزیع اطلاعات، این زمینه‌ها سرعت در هم ادغام شده، و تفاوت‌های آنها در حال محو شدن است. شرکت‌هایی که در اقصی نقاط دنیا شعبه و نمایندگی دارند، می‌توانند فقط با فشار یک دکمه از آخرین وضعیت دفاتر خود (حتی دور افتاده‌ترین آنها) مطلع شوند. اما جالب اینجاست که رشد تقاضا برای روشهای پیشرفته‌تر پردازش اطلاعات همیشه یک گام از سرعت رشد این تکنولوژیها جلوتر است. با اینکه صنعت کامپیوتر از صنایع دیگر (از جمله صنایع اتومبیل، و حمل و نقل هوایی) نسبتاً جوانتر است، اما در مدتی بس کوتاه به پیشرفتهای چشمگیری دست یافته است. در دو دهه اول، سیستمهای کامپیوتری بسیار متمرکز بودند، و معمولاً در یک اتاق بزرگ جا می‌گرفتند. کم نبودند مراکزی که این اتاقها دیوارهای شیشه‌ای داشتند، و بازدیدکنندگان با حیرت این موجودات عجیب‌الخلقه الکترونیکی را برانداز می‌کردند. دانشگاهها و شرکت‌های متوسط معمولاً یکی دو کامپیوتر بیشتر نداشتند، و تعداد شرکت‌هایی که استطاعت خرید بیش از یک دوجین از آنها را داشته باشند، چندان زیاد نبود. هیچکس (شاید غیر از نویسندگان داستانهای علمی-تخیلی) حتی نمی‌توانست تصور کند که تا قبل از پایان قرن بیستم کامپیوترهایی با همان قدرت را بتوان روی یک تمبر پستی جای داد، و آنها را بصورت انبوه تولید کرد.

پیوند فرخنده کامپیوتر و مخابرات اتفاقی بود که هر دو صنعت را دچار تحولات عظیم کرد. اکنون دیگر مفهوم اتاقی با یک کامپیوتر بزرگ بنام «مرکز کامپیوتر»، که افراد کارهایشان را به آنجا می‌آوردند، بکلی منسوخ شده است. مدل قدیمی کامپیوتر بزرگی که تمام کارهای محاسباتی سازمان را انجام می‌دهد، اکنون جای خود را به تعداد زیادی کامپیوتر کوچک متصل به هم داده است. به این سیستمها شبکه‌های کامپیوتری (computer networks) گفته می‌شود؛ موضوع این کتاب نیز طراحی و ساختار این شبکه‌هاست.

در این کتاب هر جا از «شبکه کامپیوتری» سخن می‌گوئیم، منظورمان مجموعه‌ای از کامپیوترهای مستقل است، که با یک تکنولوژی واحد به هم متصل شده‌اند. دو کامپیوتر وقتی «به هم متصلند»، که بتوانند با یکدیگر اطلاعات رد و بدل کنند. الزامی نیست که این اتصال از طریق سیمهای مسی باشد؛ فیبرهای نوری، امواج مایکروویو و مادون قرمز، و ماهواره‌های مخابراتی هم می‌توانند عامل این ارتباط باشند. بعداً خواهیم دید که اندازه، شکل و ساختار

شبکه ها می تواند بسیار متفاوت باشد. همچنین بسیاری از افراد وقتی می شنوند که اینترنت یا وب هیچکدام شبکه کامپیوتری نیستند متعجب می شوند؛ اما در پایان این کتاب علت آنرا هم خواهید فهمید. فعلاً همین قدر کافیه بدانید که: اینترنت یک شبکه نیست، بلکه شبکه ایست از شبکه ها، و وب نیز یک سیستم توزیع شده است که بر پایه اینترنت کار می کند.

لازم است همین جا به یک اشتباه رایج بین دو اصطلاح شبکه کامپیوتری و سیستم توزیع شده (distributed system) اشاره کنم. یک سیستم توزیع شده مجموعه ایست از چندین کامپیوتر مستقل، که کاربر آنرا به شکل یک سیستم واحد و متجانس می بیند. در این سیستمها معمولاً یک لایه نرم افزاری (روی سیستم عامل) بنام میان افزار (middleware) است، که مدل مورد نظر را پیاده سازی می کند. وب (World Wide Web) نمونه ای از یک سیستم توزیع شده است، که در آن همه چیز از دیدگاه کاربر یک سند (صفحه وب) بنظر می رسد. در شبکه کامپیوتری این تجانس، مدل و نرم افزار وجود ندارد. کاربران بطور مستقیم با کامپیوترها در تماسند، و هیچ کوششی برای ایجاد تجانس بین آنها صورت نمی گیرد. کاربر بروشنی تفاوت های نرم افزاری و سخت افزاری کامپیوترها را می بیند، و اگر بخواهد برنامه ای را روی یکی از کامپیوترها اجرا کند، باید ابتدا وارد آن شود (log on). در حقیقت، یک سیستم توزیع شده نرم افزاریست که روی شبکه کار می کند، و تجانس و شفافیت آن توسط این نرم افزار تأمین می شود. به همین دلیل تفاوت سیستم توزیع شده با یک شبکه بیشتر در نرم افزار (بویژه سیستم عامل) نهفته است تا سخت افزار.

با این همه، شباهتهای زیادی نیز بین این دو وجود دارد. مثلاً، سیستمهای توزیع شده و شبکه ها هر دو به انتقال فایل نیاز دارند؛ تفاوت در اینست که این کار را چه کسی انجام می دهد، سیستم یا کاربر. با آنکه این کتاب درباره شبکه های کامپیوتری است، بسیاری از مطالب آن در سیستمهای توزیع شده نیز مصداق دارد. برای کسب اطلاعات بیشتر درباره سیستمهای توزیع شده به (Tanenbaum and Van Steen, 2002) نگاه کنید.

۱.۱ کاربردهای شبکه های کامپیوتری

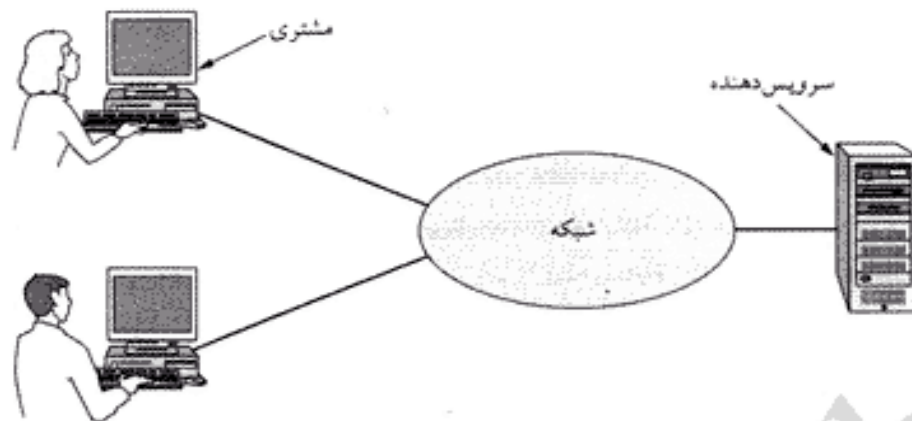
قبل از پرداختن به جزئیات فنی، بهتر است کمی درباره اینکه چرا مردم به شبکه های کامپیوتری اهمیت می دهند و چرا از آنها استفاده می کنند، صحبت کنیم (چرا که اگر کسی به شبکه اهمیت نمی داد، اصلاً شبکه ای ساخته نمی شد). ابتدا از کاربردهای سنتی (از قبیل شرکتها و افراد) شروع می کنیم، و سپس به کاربردهای جدیدتر (مانند شبکه های متحرک و خانگی) خواهیم پرداخت.

۱.۱.۱ کاربردهای تجاری

اکثر شرکتها تعداد زیادی کامپیوتر برای کارهای مختلف (تولید، انبارداری، فروش، و حسابداری) دارند. شاید در ابتدا این کامپیوترها از یکدیگر جدا باشند، ولی در مرحله ای از کار برای یکپارچه کردن اطلاعات کل شرکت، مدیریت تصمیم می گیرد تا آنها را به هم متصل کند.

به بیان کلی تر، اشتراک منابع (resource sharing) به ما اجازه می دهد تا برنامه ها، تجهیزات و بخصوص داده ها را (صرف نظر از موقعیت فیزیکی افراد و منابع) در اختیار همه آنها بی که به این شبکه متصلند، قرار دهیم. ساده ترین مثال آن، چاپگر است که برای تمام کارکنان یک دفتر به اشتراک گذاشته شده است. پیداست که تک تک این افراد به یک چاپگر اختصاصی نیاز ندارند، و علاوه بر آن یک چاپگر شبکه اغلب ارزانتر، سریعتر و کم هزینه تر از تعداد زیادی چاپگرهای پراکنده است.

با این حال، اشتراک اطلاعات بسیار مهمتر از اشتراک تجهیزات فیزیکی (مانند چاپگر، اسکنر، و CD نویس) است. امروزه تمام شرکت های بزرگ و متوسط (و بسیاری از شرکت های کوچک) بشدت به اطلاعات کامپیوتری خود



شکل ۱-۱. شبکه‌ای با دو مشتری و یک سرویس‌دهنده.

(از قبیل اطلاعات مشتریان، انبار، سندهای مالی و حسابداری، و اطلاعات مالیاتی) وابسته‌اند. بانکی که تمام کامپیوترهای آن از کار افتاده باشند، پنج دقیقه هم نمی‌تواند دوام بیاورد. حتی شرکت‌های کوچکی مانند آژانس‌های مسافرتی و دفاتر خدمات حقوقی نیز بشدت به اطلاعات کامپیوتری خود متکی هستند.

در یک شرکت کوچک تمام کامپیوترها به احتمال زیاد در یک دفتر (و یا حداکثر یک ساختمان) قرار دارند، در حالیکه کامپیوترهای یک شرکت بزرگ می‌تواند در یک شهر یا کشور (و حتی در قاره‌های مختلف) پراکنده باشد. در این حالت، ممکنست مدیر فروشی که در نیویورک نشسته، به موجودی انبار شرکت در سنگاپور نیاز داشته باشد. بعبارت دیگر، حتی ۱۵۰۰۰ کیلومتر فاصله هم نباید خللی در دسترسی به اطلاعات وارد کند. در واقع می‌توان گفت، ما بدنبال «از بین بردن فاصله‌ها» هستیم.

در ساده‌ترین شکل، اطلاعات شرکت می‌تواند در یک یا چند پایگاه داده متمرکز باشد، و کارمندان شرکت بایستی بتوانند از راه دور به آنها دسترسی داشته باشند. در این مدل، اطلاعات در کامپیوترهای پُر قدرتی بنام سرویس‌دهنده (server) - که اغلب در یک مرکز و تحت کنترل سرپرست سیستم قرار دارند - نگهداری می‌شوند. کارمندان نیز، که در اینجا به آنها مشتری (client) گفته می‌شود، از راه دور و از پای کامپیوترهای معمولی خود به این اطلاعات دسترسی پیدا می‌کنند. (گاهی به فردی که از کامپیوتر استفاده می‌کند، نیز «مشتری» گفته می‌شود؛ بهر حال، از فحوای متن باید بتوانید متوجه شوید که منظور کامپیوتر است یا کاربر.) اتصال کامپیوترهای سرویس‌دهنده و مشتری از طریق شبکه صورت می‌گیرد (شکل ۱-۱ را ببینید). در این شکل شبکه به صورت یک بیضی ساده نشان داده شده است؛ وقتی بخواهیم شبکه را بصورت کلی و انتزاعی (و بدون هیچگونه جزئیاتی) نشان دهیم، از این روش استفاده خواهیم کرد.

به این آرایش مدل مشتری-سرویس‌دهنده (client-server model) گفته می‌شود، و در بسیاری از شبکه‌های کوچک و بزرگ کاربرد دارد چون مستقل از فاصله است. وب نیز بر مبنای مدل مشتری-سرویس‌دهنده ساخته شده است؛ وقتی یک صفحه وب را باز می‌کنید، در واقع آنرا از سرویس‌دهنده وب دریافت کرده، و در کامپیوتر خود (که در اینجا مشتری است) نمایش می‌دهید. در اکثر مواقع یک سرویس‌دهنده می‌تواند به تعداد زیادی مشتری سرویس بدهد.

در مدل مشتری-سرویس‌دهنده را دقیقتر بررسی کنیم، متوجه می‌شویم که دو پروسس (process) در آن دخیل هستند: یک پروسس روی کامپیوتر مشتری، و دیگری روی کامپیوتر سرویس‌دهنده. ارتباط از لحظه‌ای آغاز می‌شود، که پروسس مشتری از طریق شبکه یک پیام به پروسس سرویس‌دهنده فرستاده، و سپس به انتظار پاسخ

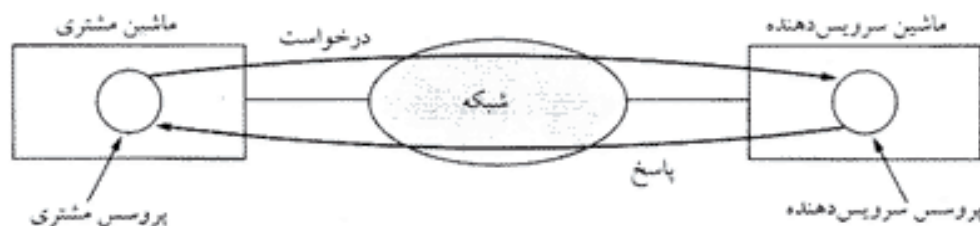
آن می ماند. وقتی پروسس سرویس دهنده درخواست مشتری را دریافت کرد، کار خواسته شده را انجام می دهد (یا اطلاعات خواسته شده را تهیه می کند)، و پاسخ را به مشتری پس می فرستد. این فرآیند را در شکل ۱-۲ ملاحظه می کنید.

گاهی در یک شبکه کامپیوتری رابطه بین افراد اهمیت بیشتری نسبت به تبادل اطلاعات بین کامپیوترها دارد. چنین شبکه ای در واقع یک رسانه ارتباطی (communication medium) است. امروزه دیگر تقریباً هیچ شرکتی را نمی توان یافت که از سرویس پست الکترونیک (ایمیل: e-mail) استفاده نکند، و در واقع بسیاری از ارتباطات روزمره کارمندان از همین طریق صورت می گیرد. این روش آنقدر ساده و کارآمد است که خود باعث بروز مشکلات جدیدی شده است، چون رؤسای شرکتها هم یاد گرفته اند چطور فقط با فشار یک دکمه می توانند پیامهای (اغلب بی محتوای) خود را به این طرف و آن طرف بفرستند!

اما ایمیل تنها شکل از ارتباطات پیشرفته ای نیست که به لطف شبکه های کامپیوتری ممکن شده است. در یک شبکه، دو نفر که فاصله زیادی هم از یکدیگر دارند، می توانند بطور مشترک روی یک گزارش یا مقاله کار کنند. وقتی یکی از آنها تغییری در این گزارش می دهد، دیگری بلافاصله آنرا خواهد دید (و دیگر نیازی نیست روزها به انتظار پستی چشم به در بدوزد). با این روش دیگر نیازی نیست غصه هماهنگ کردن کارمندانی که هر کدام ساز خود را می زنند، بخورید.

یکی دیگر از امکانات ارتباطی شبکه ها، کنفرانس ویدئویی (video conferencing) است. به کمک این تکنولوژی، کارمندانی که هزاران کیلومتر از هم فاصله دارند، می توانند یکدیگر را ببینند، صدای هم را بشنوند، و یا حتی مطالب خود را روی یک تخته سیاه مجازی بنویسند. کنفرانس ویدئویی جانشین بسیار مناسبی برای کنفرانسهای واقعی (که متضمن تحمل هزینه های سفر است) می باشد. گاهی گفته می شود که صنعت ارتباطات و حمل و نقل با یکدیگر مسابقه مرگ وزندگی گذاشته اند، و هر کدام پیروز شود، دیگری را از میدان بدر خواهد کرد. اتفاق دیگری که این روزها شتاب بیشتری گرفته، امکان تجارت الکترونیک بین شرکتهای کوچک و بزرگ است. برای مثال، سازندگان کامپیوتر، اتومبیل و هواپیما می توانند قطعات مورد نیاز خود را از طریق شبکه های کامپیوتری به سازندگان این نوع قطعات سفارش دهند، و سپس آنها را مونتاژ و تبدیل به محصول نهایی کنند. سفارش و خرید قطعات در لحظه نیاز (زمان واقعی) لزوم نگهداری و انبار کردن مقدار زیادی از آنها را مستفی می کند.

گرایش تجاری دیگری که حتی اهمیت بیشتری پیدا کرده، فروش محصولات روی اینترنت است. این روزها شرکتهای بسیاری (از قبیل خطوط هوایی، کتابفروشیها، و فروشندگان محصولات فرهنگی) به فروش محصولات خود از طریق اینترنت روی آورده اند. این شاخه از تجارت (که به تجارت الکترونیک - electronic commerce یا e-commerce - معروف است) در آینده رشد بسیار بیشتری خواهد کرد.



شکل ۱-۲. مدل مشتری سرویس دهنده بر «درخواست و پاسخ» مبتنی است.

۲-۱-۱ کاربردهای خانگی

سال ۱۹۷۷، وقتی از کین اولین (رئیس شرکت Digital Equipment Corporation - که پس از IBM بزرگترین شرکت کامپیوتری دنیا محسوب می شد) پرسیدند چرا وارد بازار کامپیوترهای شخصی نمی شود، وی پاسخ داد: "هیچ دلیلی ندارد که هر کس توی خانه اش یک کامپیوتر داشته باشد." تاریخ ثابت کرد که اولین اشتباه می کرد، و اکنون دیگر شرکت DEC وجود خارجی ندارد. اما چرا مردم برای کارهای خانگی خود کامپیوتر می خرند؟ نوشتن نامه، مقاله و حتی کتاب (و تا یادم نرفته، بازی) یکی از مهمترین دلایل آن است؛ اما این وضعیت امروزه در حال تغییر است. شاید مهمترین دلیل خرید کامپیوترهای خانگی در سالهای اخیر اینترنت باشد. کارهای که این قبیل افراد با کامپیوتر خود انجام می دهند، عمدتاً عبارتند از:

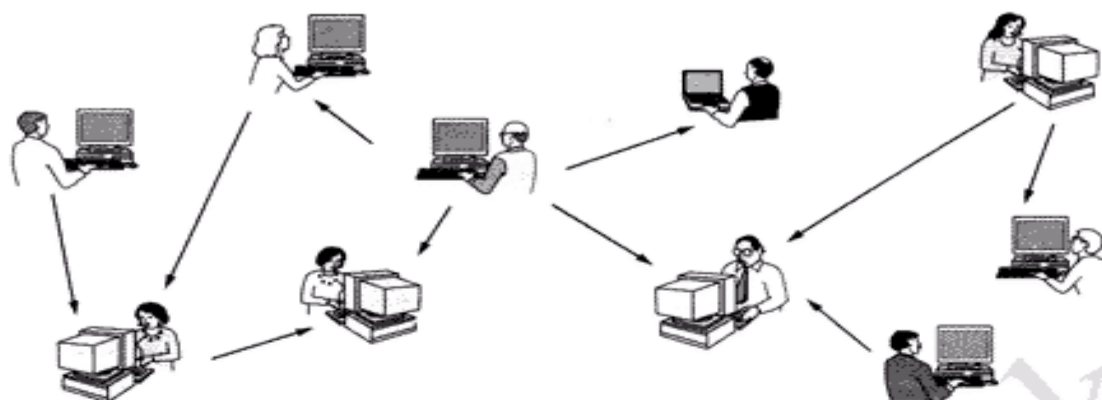
۱. دسترسی به اطلاعات پراکنده در سراسر دنیا
۲. ارتباطات دو جانبه
۳. سرگرمیهای تعاملی
۴. تجارت الکترونیک

امروزه منبع بسیار عظیمی از اطلاعات در تمامی زمینه ها (از قبیل هنر، تجارت، آشپزی، بهداشت، تاریخ، سرگرمی، علم، ورزش و تفریحات سالم - و البته گاهی ناسالم!) روی اینترنت وجود دارد، که می توان به آنها دسترسی پیدا کرد. روزنامه های بسیاری روی اینترنت منتشر می شوند، که می توان اخبار را بدلیخواه و بصورت گزینشی از آنها بدست آورد. حتی می توانید کاری کنید که مقاله دلخواه شما وقتی خواب هستید، از اینترنت بار شده و سپس چاپ شود، تا موقع صبحانه بتوانید با خیال راحت آنرا بخوانید. (به این ترتیب روزنامه فروشهای بیچاره بیکار خواهند شد، ولی مطبوعات هم هیچ وقت دل خوشی از آنها نداشتند.)

بعد از روزنامه ها و مجلات الکترونیکی نوبت کتابخانه های دیجیتالی است. بسیاری از سازمانهای علمی معتبر مانند ACM (www.acm.org) و IEEE (www.computer.org) مدتهاست که انتشارات و کنفرانسهای متعددی روی اینترنت برگزار می کنند؛ و این روند سرعت در حال گسترش است. بنظر می رسد که رواج کتابخوانی اینترنتی فقط به قیمت، اندازه و وزن کامپیوترهای کتابی بستگی دارد. (شاید هنوز عده ای به این آینده شک داشته باشند؛ اما بهتر است آنها بلایی را که دستگاه چاپ گوتنبرگ سر کتابهای زیبای خطی آورد، بیاد بیاورند.)

تمام کاربردهایی که در بالا نام بردیم، متضمن ارتباط فرد با یک منبع اطلاعات روی اینترنت بود. اما روش دیگری نیز برای برقراری ارتباط از طریق اینترنت وجود دارد، و آن ارتباط فرد به فرد است (این پاسخ تکنولوژی قرن بیست و یکم است به تلفن قرن نوزدهمی). امروزه میلیونها نفر در سراسر دنیا بطور روزمره از ایمیل استفاده می کنند؛ صوت و تصویر هم مدتهاست به جزیی جدایی ناپذیر از آن تبدیل شده است (و باید بزودی منتظر بوهای اینترنتی هم باشیم!).

این روزها همه نوجوانانی که سری میان سرها در می آورند، به برنامه های پیام رسان قوری (instant messaging) معتاد هستند. این برنامه ها (که از برنامه talk در سیستمهای یونیکس مشتق شده اند) به افراد امکان می دهند پیامهای متنی خود را بلافاصله (و بدون تأخیر زمانی) با هم مبادله کنند. نسخه هایی از این برنامه ها که به چندین نفر اجازه می دهند تا در آن واحد با هم گفتگو کنند، به اتاق گفتگو (chat room) معروفند. گروههای خبری (newgroup) از امکانات قدیمی و طرفدار اینترنت است، که امکان بحث درباره موضوعات بسیار متنوعی را به شما می دهد. در این سیستم پیامی که می فرستید، به تمام آنهایی که عضو گروه خبری هستند خواهد رسید (خوششان بیاید، یا نیاید). مبادلات گروه خبری (بر خلاف اتاق گفتگو) بصورت بلافاصله و در زمان واقعی نیست، و پیامها در نقطه ای ذخیره می شوند، تا کاربر بتواند هر زمان که خواست آنها را بخواند.



شکل ۳-۱. در یک سیستم همتا-به-همتا مشتری یا سرویس دهنده ثابتی وجود ندارد.

نوع دیگری از امکانات ارتباطی موجود در اینترنت، ارتباط همتا-به-همتا (peer-to-peer) است. این مدل تفاوت اساسی با مدل مشتری-سرویس دهنده دارد (به Parameswaran et al., 2001 نگاه کنید). در این مدل ارتباط افراد در یک گروه غیرثابت و ناپایدار صورت می گیرد (شکل ۳-۱ را ببینید). در واقع، هر فرد می تواند مستقیماً با هر فرد (یا افراد) دیگر تماس برقرار کند، و چیز ثابتی بعنوان سرویس دهنده یا مشتری وجود ندارد. بزرگترین نمونه ارتباط همتا-به-همتا در حوالی سال ۲۰۰۰ با سرویسی بنام Napster شکل گرفت؛ این سرویس در اوج خود امکان ارتباط بیش از ۵۰ میلیون نفر را فراهم می آورد، که بصورت غیر قانونی به رد و بدل کردن موزیک مبادرت می کردند (این بزرگترین نقض حق التألیف در تاریخ موسیقی بود؛ به Lam and Tan, 2001 و Macedonia, 2000 نگاه کنید). ایده کار نسبتاً ساده بود: هر نفر می توانست آهنگهایی را که در هارد دیسک خود داشت، در پایگاه داده مرکزی Napster ثبت کند؛ افرادی هم که بدنبال آهنگ خاصی بودند، این پایگاه داده را جستجو کرده، و بعد مستقیماً به سراغ آن می رفتند. Napster ادعا می کرد که هیچ حق التألیفی را نقض نمی کند، چون اساساً آهنگها در کامپیوترهای آن ذخیره نمی شوند؛ اما دادگاه با این نظر موافق نبود، و حکم به تعطیلی آن داد. سیستمهای همتا-به-همتا جدید با هوشتر شده اند، چون پایگاه داده مرکزی را حذف کرده اند و بجای آن این اطلاعات در کامپیوتر تک تک افراد ذخیره می شود، و آنها لیستی از افراد مجاور خود را هم در اختیار دارند. در این روش جستجو کمی بیشتر طول می کشد (که بار آن هم بر دوش کامپیوترهاست)، ولی در نهایت به همان اندازه مؤثر است.

همه برنامه های همتا-به-همتا هم غیرقانونی نیستند. برای مثال، برنامه هایی هستند که اجازه می دهند تا آهنگها و فیلمهای مجاز و حتی عکسهای خانوادگی خود را روی اینترنت به اشتراک بگذارید، و یا بازیهای دسته جمعی انجام دهید. در حقیقت، پرتعدادترین کاربرد اینترنت، یعنی ایمیل، ذاتاً یک سیستم همتا-به-همتا است، و بنظر می رسد در آینده این سیستمها حتی گسترده تر شوند.

جرایم الکترونیکی به دزدی آثار و نقض حق التألیف محدود نمی شود، و این روزها قمارخانه ها هم پایشان به اینترنت باز شده است. کامپیوترها قادرند هر کاری انجام دهند، پس چرا قمار نکنند؟ البته قمار در بسیاری از کشورها غیرقانونی است، اما مشکل اینجا است که در چند جا هم قانونیست (مانند انگلستان)، و صاحبان کازینوها به امکانات بالقوه اینترنت برای قمار واقف شده اند. اما اگر کازینو و قمارباز در دو کشور متفاوت (که قوانین متفاوتی هم در زمینه قمار دارند) باشند، چطور؟ سؤال خوبیست!

ارتباطات اینترنتی در زمینه تماسهای تلفنی، ویدئویی و رادیو نیز تحولات وسیعی ایجاد کرده اند. آموزش از

راه دور (telelearning) نیز یکی دیگر از امکاناتیست که اینترنت عرضه کرده است. (تصورش را بکنید که ساعت ۸ صبح سر کلاس درس حاضر باشید، بدون اینکه لازم باشد قبل از آن از رختخواب بیرون بیایید!) بنظر می رسد در دراز مدت اینترنت بزرگترین نقش را در بهبود ارتباطات انسانی بازی کند.

سومین دسته از کاربردهای خانگی شبکه های کامپیوتری، صنعت سرگرمی و تفریحات (با رشدی سرسام آور) است. داغترین بحث در این زمینه پخش فیلم برحسب تقاضا (video on demand) است. شاید تا ده سال دیگر پراحتی بتوانید فیلم دلخواه خود را انتخاب کرده، و همان لحظه روی صفحه تلویزیون تماشا کنید. فیلمهای جدید تعاملی (interactive) خواهند بود، بدین معنا که بیننده می تواند مسیر سناریو را بدلخواه خود تغییر دهد. تلویزیون زنده (شرکت مستقیم و بلافاصله در مسابقات و شوهای تلویزیونی) نیز یکی دیگر از امکانات آینده است.

بازیهای تعاملی یکی دیگر از امکانات شبکه است که شاید آینده آن حتی از پخش فیلم برحسب تقاضا نیز داغتر باشد. حتی همین حالا هم گروههای بزرگی از جوانان ماجراجو شب و روز مشغول بازی موش و گربه و جنگهای هوایی، زمینی و دریایی در زوایای تاریک و دورافتاده این دنیای مجازی (اینترنت) هستند. اگر آینده بتواند امکانات پخش سه بعدی و کیفیت بالا را عرضه کند، دیگر این دنیای مجازی هیچ چیز از دنیای واقعی کم نخواهد داشت.

چهارمین دسته از کاربردهای شبکه، شاید وسیعترین آنها باشد: خرید از خانه (home shopping). امروزه میلیونها نفر در سراسر جهان هر روز مایحتاج خود را بطور مستقیم از اینترنت تهیه می کنند، و هرگز پا از خانه بیرون نمی گذارند (حداقل برای خرید). هزاران شرکت بزرگ و کوچک کاتالوگ محصولات خود را بصورتی جذاب روی اینترنت گذاشته اند، و برای خرید هر یک از آنها کافیت روی جنس موردنظر یک کلیک کنید. کالایی را خریدید، ولی نمی دانید چگونه کار می کند؟ نگران نباشید، باز هم اینترنت به شما کمک می کند، و هر اطلاعات و راهنمایی که بخواهید در اختیارتان قرار می دهد.

می خواهید صورتحسابهای خود را پرداخت کنید؟ از آخرین وضعیت حسابهای بانکی خود مطلع شوید؟ و یا سرمایه گذاری جدیدی بکنید؟ باز هم اینترنت در خدمت شماست. امروزه میلیونها نفر در سراسر جهان کارهای مالی و بانکی خود را بصورت الکترونیکی انجام می دهند، و با تقویت مسائل امنیتی شبکه این روند حتی گسترش بیشتری نیز خواهد یافت.

یکی از زمینه هایی که شاید هیچکس تصور اینترنتی شدن آنرا نمی کرد، سمساری بود. حراج اینترنتی اشیاء دست دوم اینک به یکی از تجارتهای بزرگ تبدیل شده است. بر خلاف تجارت الکترونیک معمولی که از مدل مشتری - سرویس دهنده استفاده می کند، حراج اینترنتی در واقع یک سیستم همتا به - همتا یا خریدار به - خریدار (consumer-to-consumer) است. امروزه با اصطلاحات زیادی از این دست برخورد می کنید، که در آنها بجای "to" از "2" استفاده می شود (چون تلفظ آنها یکسان است). در شکل ۱-۴ تعدادی از رایجترین این اصطلاحات را ملاحظه می کنید.

شکی نیست که کاربردهای شبکه و اینترنت در آینده سرعت افزایش خواهد یافت، و در زمینه هایی رسوخ خواهد کرد که امروز حتی به تصور کسی نمی آید. چه کسی در سال ۱۹۹۰ می توانست تصور کند که بخش بزرگی از درآمد شرکتهای تلفن از محل پیامهای کوتاهی باشد که دانش آموزان دبیرستانی بطور خستگی ناپذیر و در حالیکه سوار اتوبوس مدرسه هستند، با تلفن همراه خود برای دوستانشان می فرستند؟ این شرکتهای بخوبی می دانند که سرویس پیام کوتاه (Short Message Service - SMS) بسیار سودآور است. شبکه های کامپیوتری برای افرادی که دور از شهرها زندگی می کنند، نیز مفید است. اینان می توانند راحت در

اصطلاح	نام کامل	مثال
B2C	فروشنده-به-خریدار	خرید کتاب روی اینترنت
B2B	فروشنده-به-فروشنده	خرید قطعات یدکی توسط تولیدکننده
G2C	دولت-به-خریدار	توزیع فرمهای مالیاتی از طریق اینترنت
C2C	خریدار-به-خریدار	حراج اشیاء دست دوم
P2P	همتا-به-همتا	اشتراک فایل

شکل ۱-۴. برخی از انواع تجارت الکترونیک.

روستاهای خود زندگی کنند، و در عین حال به تمام امکانات شهرهای بزرگ هم دسترسی داشته باشند. دانشگاههای آینده به احتمال زیاد حالت ملی و محلی خود را از دست داده، و بصورت بین المللی درخواست خواهند آمد. درمان از راه دور (telemedicine) امروزه به کنترل بیماران محدود می شود، ولی چه کسی می تواند امکانات بالقوه آنرا پیش بینی کند. (یا مثلاً، چقدر خوب می شد اگر می توانستیم یک دوربین دیجیتالی در یخچال خود نصب کنیم، تا هر وقت شیر تمام شد بتوانیم سر راه خانه شیر بخریم!)

۳-۱-۱ کاربران سیار

کامپیوترهای سیار، مانند کامپیوترهای کتابی و دستیاران دیجیتالی (PDA)، یکی از سریعترین رشدها را در صنعت کامپیوتر تجربه می کنند. اغلب دارندگان این وسایل میل دارند حتی وقتی از خانه دور و یا در سفر هستند، با کامپیوتر خانگی یا دفتری خود ارتباط داشته باشند. در این قبیل موارد دیگر شبکه های کابلی محلی از اعراب ندارد، و باید به فکر شبکه های بیسیم (wireless network) باشیم. در این قسمت نگاهی به کاربردهای شبکه های بیسیم خواهیم داشت.

جالبترین کاربرد شبکه های بیسیم در ایجاد دفاتر سیار است. اغلب افراد میل دارند در سفر همان کارهایی را انجام دهند که در دفتر کار خود انجام می دهند (ایمیل و فکس بفرستند، تلفن راه دور بزنند، فایل های خود را باز کنند، و یا در وب گشت بزنند)، و اصلاً هم کاری به این ندارند که کجا هستند! برای مثال، امروزه در اغلب کنفرانسهای کامپیوتری گردانندگان کنفرانس یک شبکه بیسیم در محوطه کنفرانس راه می اندازند، و هر کسی می تواند با استفاده از یک مودم بیسیم به اینترنت دسترسی پیدا کند. بسیاری از دانشگاهها هم در محوطه خوابگاهی خود شبکه های بیسیم دارند، و به دانشجویان امکان می دهند تا زیر درختان محوطه نشسته و ایمیل های خود را چک کنند، و یا در کتابخانه دانشگاه دنبال مقاله بگردند.

شبکه های بیسیم در امور حمل و نقل (کشتیها، کامیونها و تاکسیها) تحولی بزرگ ایجاد کرده اند. برای مثال، در بسیاری از شهرهای بزرگ رانندگان تاکسی مستقل بوده و عضو هیچ شرکت یا اتحادیه ای نیستند. وقتی کسی به تاکسی نیاز دارد، به یک سرویس مرکزی تلفن می کند، و مشخصات وی (از قبیل مبدأ و مقصد) توسط این سرویس به تمام تاکسیها ارسال می شود. اولین تاکسی که مایل به انجام این سرویس باشد، با فشار یک دکمه اعلام آمادگی کرده و به سراغ مسافر می رود.

شبکه های بیسیم از نظر نظامی نیز اهمیت فوق العاده ای دارند. هیچ ارتشی نمی تواند در جنگهای بزرگ به شبکه های عمومی تکیه کند، و بهتر است شبکه ای خاص خود بر پا کند؛ و چه چیزی بهتر از یک شبکه بیسیم. با وجود شباهتهای بسیار بین شبکه های بیسیم و کامپیوترهای سیار، آنها یکی نیستند (شکل ۱-۵ را ببینید). به تفاوت شبکه های بیسیم ثابت (fixed wireless) و شبکه های بیسیم سیار (mobile wireless) توجه کنید. در بسیاری از دفاتر، کامپیوترهای کتابی سیار بصورت ثابت به شبکه محلی متصل شده اند؛ از طرف دیگر کامپیوتری که با استفاده از مودم به شبکه وصل می شود، سیار است - ولی مسلماً به آن بیسیم نمی توان گفت.

کاربردها	سیار	بسیم
کامپیوترهای رومیزی در دفتر کار	No	No
کامپیوتری که با مودم به شبکه وصل شده	Yes	No
شبکه بیسیم مستقر در یک ساختمان	No	Yes
دفاتر سیار؛ PDA ها	Yes	Yes

شکل ۱-۵. ترکیب شبکه‌های بیسیم و کامپیوترهای سیار.

از طرف دیگر، هر شبکه بیسیمی الزاماً سیار نیست. ساختمانهای بسیاری وجود دارند، که بدلیل مشکلات کابل کشی از شبکه‌های بیسیم استفاده می‌کنند. امروزه نصب شبکه‌های بیسیم بسیار ساده شده است، و دردسرهای کابل کشی را هم ندارد.

البته ترکیب بیسیم با کامپیوترهای سیار نیز عملیست، و امروزه کاربردهای مهمی دارد. کسانی که در انبارهای بزرگ یا فروگاههای شلوغ کار می‌کنند، تجربه استفاده از این سیستمهای ترکیبی را دارند. این کامپیوترها با اطلاعات ورودی کمی که می‌گیرند، و با اتصال بیسیم به پایگاه داده مرکزی، کار خود را با سرعت و دقت انجام می‌دهند. با رشد تکنولوژی بیسیم، مسلماً کاربردهای آن نیز گسترش خواهد یافت. اجازه دهید نگاهی به این احتمالات بیندازیم.

پارکومترهای بیسیم کار دولت و افراد را راحتتر خواهند کرد. این پارکومترها می‌توانند کارت اعتباری نیز قبول کرده، و با سرعت اعتبار آنها چک کنند؛ و وقتی مدت پارک تمام شد، اگر هنوز اتومبیل شما آنجا بود، نزدیکترین پلیس را خبر می‌کنند تا آنها جریمه کند! تخمین زده شده که فقط پلیس ایالات متحده می‌تواند از این طریق ۱۰ میلیارد دلار بر درآمد خود بیفزاید (Harte et al., 2000). این روش اثرات مثبت زیست‌محیطی نیز دارد، چون رانندگان خودروها مطمئنند که راهی برای فرار از دست پلیس ندارند، و به ناچار به وسایل نقلیه عمومی روی می‌آورند.

امروزه ماشینهای خودکار فروش غذا، نوشابه و چیزهای دیگر در همه جا یافت می‌شوند. اما غذا و نوشابه که از آسمان وارد این ماشینها نمی‌شود؛ یک مأمور سوار بر کامیون هر از چند گاهی به این ماشینها سرکشی می‌کند، تا در صورت نیاز آنها را پُر کند. اگر این ماشینها هر روز موجودی خود را از طریق شبکه بیسیم به مرکز اطلاع دهند، مأمور ما می‌داند سراغ کدام ماشینها باید برود، و چه مقدار کالا احتیاج دارد (و حتی می‌تواند مسیر حرکت خود را به بهترین نحو برنامه‌ریزی کند). البته این اطلاعات را از طریق خطوط تلفن هم می‌توان به مرکز منتقل کرد؛ اما کشیدن یک خط تلفن برای هر ماشین (آن هم برای یک تماس در روز) اصلاً مقرون بصرفه نیست.

زمینه دیگری که شبکه‌های بیسیم می‌توانند باعث صرفه‌جویی شوند، قرائت کنتورهای مختلف خانگی است. اگر هر کنتور آب، برق و گاز اطلاعات خود را از طریق شبکه بیسیم به شرکت مربوطه منتقل کند، دیگر نیازی به مراجعه کنتورخوان‌ها به درب منازل نیست. به همین ترتیب، اگر آشکارسازهای دود و حرارت آلام خود را (بجای راه انداختن آژیر و سر و صدا) مستقیماً به مراکز آتش‌نشانی بفرستند، بسیار مؤثرتر خواهد بود. با کاهش قیمت دستگاههای رادیویی (که مبنای شبکه‌های بیسیم هستند)، وسایل اندازه‌گیری و گزارش‌دهی بیشتری به استفاده از آنها روی خواهند آورد.

زمینه دیگری از کاربردهای شبکه‌های بیسیم (که از مدتها قبل نیز انتظار آن می‌رفت)، ادغام تلفنهای همراه و PDA ها با کامپیوترهای بیسیم است. اولین PDA های بیسیم قادر بودند صفحات ساده وب را روی صفحات کوچک خود نمایش دهند. این سیستم که WAP 1.0 (Wireless Application Protocol) نام داشت، بدلیل خوانا نبودن صفحات، پهنای باند کم، و سرویس ضعیف با شکست مواجه شد. ولی با سرویسها و وسایل جدید WAP 2.0 اوضاع مسلماً بهتر خواهد شد.

ترکیب این تکنولوژیها می تواند به سرویس جدیدی منجر شود که می توان آنرا تجارت سیار (mobile commerce) نامید (Senn, 2000). این پدیده در واقع عامل ترکیب کننده تکنولوژی PDA های بیسیم با تجارت الکترونیک است، که این روزها همه از آن سهم می خواهند، و این امیدواری وجود دارد که افراد به خرید و انجام کارهای بانکی با آن روی آورند. PDA بیسیم می تواند در فروشگاهها و مراکز خرید بعنوان عامل انتقال پول و یا کارت اعتباری عمل کند (بدین ترتیب که این هزینه ها بعداً با صورتحساب تلفن پرداخت شود). نکته مثبت این روش برای فروشندگان آن است که هزینه های کار با شرکتهای اعتباری را به مقدار زیادی پائین می آورد. البته عیب بزرگی نیز دارد: خریداران می توانند قبل از خرید با PDA بیسیم خود قیمتها را با فروشندگان دیگر مقایسه کنند. اگر شرکتهای تلفن سرویس قرائت بارکد را هم به این PDA ها اضافه کنند، که دیگر اوضاع خرابتر می شود! چون دیگر حتی با مخفی کردن قیمتها هم نمی توان جلوی مقایسه قیمتها توسط خریدار را گرفت (کاری که خیلی از فروشندگان به آن امید بسته اند)!

از آنجائیکه محل این دستگاهها همیشه برای اپراتور سیستم مشخص است، می توان سرویسهای خاصی را در اختیار کاربران آنها گذاشت؛ مثلاً، می توان آدرس نزدیکترین کتابفروشی یا رستوران چینی را در اختیار وی گذاشت، و یا آخرین پیش بینی وضعیت هوا را به وی اعلام کرد.

امکانات و کاربردهای این سرویس جدید می تواند بسیار فراتر از مثالهای ساده فوق باشد؛ و خوبی قضیه اینست که کاربران تلفنهای موبایل عادت دارند برای هر چیزی پول بدهند (درست بر خلاف کاربران اینترنت که همه چیز را مجانی می خواهند)! اگر یک سایت اینترنتی برای قبول کارت اعتباری از شما درخواست پول کند، فوراً فریادتان بلند خواهد شد، ولی اگر همین اتفاق روی سرویسهای موبایل بیفتد، بدون هیچ اعتراضی قبول می کنید (البته فعلاً).

بگذارید کمی هم به آینده نگاه کنیم: شبکه های شخصی (Personal Area Network)، و کامپیوترهای پوشیدنی (wearable computer). به تازگی IBM ساعتی ساخته که سیستم عامل لینوکس (Linux) روی آن اجرا می شود، و می تواند به اینترنت وصل شده و ایمیل رد و بدل کند (Narayanaswami et al., 2002). در آینده دیگر چیزی بعنوان کارت ویزیت بین افراد رد و بدل نخواهد شد، و آنها می توانند با یک تماس ساعت مچی تمام اطلاعات طرف مقابل را دریافت کنند. به احتمال زیاد کامپیوترهای پوشیدنی (که اطلاعات زیستی فرد را در خود دارند) جای کارتهای مغناطیسی را برای ورود به مکانهای حساس خواهند گرفت، و یا می توانند در هر لحظه مکان فرد را اعلام کنند. امکانات این سیستمها تقریباً بی شمار است.

ساعتها و رادیوهای هوشمند را همه ما سالهاست از طریق فیلمهای جیمز باند می شناسیم، ولی آیا تا بحال گرد و غبار هوشمند به گوشتان خورده است؟ محققان دانشگاه پرکلی اخیراً یک کامپیوتر بیسیم ساخته اند، که در مکعبی با ابعاد 1 mm جای می گیرد (Warneke et al., 2001). با این کامپیوترها می توان مسیر حرکت چمدان در فرودگاه ها، و یا جانوران (مثلاً، پرندگان مهاجر) را با دقت کنترل کرد.

۴-۱-۱ تبعات اجتماعی

گسترش روزافزون شبکه های کامپیوتری باعث ایجاد مسائل اجتماعی، اخلاقی و سیاسی خاص خود شده است که در این قسمت برخی از آنها را بررسی می کنیم. یکی از امکانات شبکه های کامپیوتری تبادل آزاد و سریع اطلاعات و اخبار است. البته تا وقتی این پیامها در محدوده های فنی باقی بماند، مشکل چندانی وجود نخواهد داشت؛ در دسر وقتی شروع می شود که صحبت به مسایل حساس (از قبیل سیاست، مذهب یا سکس) کشیده شود.

نظرها و دیدگاههایی که توسط اعضای یک گروه خبری پُست می شود، ممکنست برای افراد دیگر بسیار ناهنجار و موهن باشد؛ و وقتی پیامها به متن محدود نشود، کار بدتر هم خواهد شد. امروزه براحتمی می توان

عکسهای بسیار واضح و با کیفیت عالی (و یا حتی کلیپهای ویدئویی کوچک) را از طریق اینترنت منتشر کرد. برخی از افراد در زندگی به فلسفه «زندگی کن، و بگذار زندگی کنند» معتقدند، اما عده زیادی هم هستند که احساس می کنند به برخی مطالب (مانند حمله به کشورها یا مذاهب دیگر، صور قبیحه و غیره) نباید اجازه انتشار داد. کشورهای مختلف هم دارای قوانین متفاوتی در این زمینه ها هستند؛ و جدال از همین جا شروع می شود.

در این میان بسیاری از افراد اپراتورهای شبکه را (مانند روزنامه ها و مجلات) مسئول محتویات شبکه می دانند، اما واقعیت اینست که یک شبکه بیشتر شبیه اداره تلفن و پست است تا روزنامه یا مجله (و نمی توان آنرا مسئول چیزهایی که از این طریق مبادله می شود، دانست). از طرف دیگر، اگر اپراتور شبکه اجازه داشته باشد مطالب را سانسور کند، به احتمال زیاد (برای فرار از متهم شدن) روی کوچکترین چیزها هم انگشت خواهد گذاشت، و بدین ترتیب حقوق افراد در زمینه آزادی بیان از بین خواهد رفت. بحث موافق و مخالف همچنان ادامه دارد (و پراحتی می توان حدس زد که به این زودی ها هم به نتیجه نخواهد رسید).

بحث جالب دیگر حقوق و رابطه کارگر و کارفرما است. بسیاری از افراد در محل کار خود ایمیل می فرستند، و یا ایمیل های رسیده را می خوانند. برخی از کارفرمایان ادعا می کنند که آنها حق دارند ایمیل های کارمندان خود را بخوانند و یا آنها را سانسور کنند؛ و صد البته کارمندان با این حرفها موافق نیستند!

حتی اگر بپذیریم کارفرمایان در ادعای خود محق هستند، آیا می توان این رابطه را به دانشجو و دانشگاه (و یا دانش آموز و مدرسه) تعمیم داد؟ در سال ۱۹۹۴ دانشگاه کارنگی سلون تصمیم گرفت برخی از پیامهای رسیده را که در آنها به موضوعات سکسی پرداخته شده بود، سانسور کند (با این استدلال که این مطالب برای افراد زیر ۱۸ سال مناسب نیست). سالها طول کشید تا پس لرزه های این اقدام فروکش کند.

و از همه مهمتر رابطه دولت با شهروندان است. اداره آگاهی فدرال ایالات متحده (FBI) سالهاست سیستمی را در مراکز ارائه سرویس اینترنت (ISP) نصب کرده، که به آن اجازه می دهد تا ایمیل های ورودی و خروجی را تجسس کند (Blaze and Bellovin, 2000; Sobel, 2001; Zacks, 2001). نام این سیستم Carnivore (گلی گوشتخوار) بود، که بدلیل حساسیتهای ایجاد شده در جامعه به نام کم ضررتر DCS1000 تغییر داده شد (با این حال کار آن همچنان جاسوسی در ایمیل های مردم بود). طبق اصلاحیه چهارم قانون اساسی ایالات متحده آمریکا، دولت بدون مجوز قانونی حق تجسس در احوال شخصی افراد را ندارد. اینکه این قانون نوشته شده در قرن هیجدهم هنوز در قرن بیست و یک اعتبار دارد یا خیر، را آینده روشن خواهد کرد.

فضولی در کار مردم به دولت محدود نمی شود؛ بخش خصوصی هم از این گناه در امان نیست. برای مثال، مرورگرهای وب از فایل های کوچکی بنام کوکی (cookie) استفاده می کنند که اطلاعات شخصی افراد را در اختیار شرکتها می گذارد، و حتی می تواند منجر به افشای شماره کارتهای اعتباری و اطلاعات محرمانه دیگر روی اینترنت شود (Berghel, 2001).

در شبکه های کامپیوتری می توان پیامهای بدون نام و نشانی فرستاد، که در جای خود می تواند مفید باشد. مثلاً، کارمندان، دانشجویان و یا مردم عادی می توانند بدین طریق اعمال خلاف رؤسای شرکتها، استادان و سیاستمداران را به اطلاع عموم برسانند بدون آنکه از اقدامات تلافی جویانه آنها ترسی به دل راه دهند. (البته در بسیاری از کشورها از این قبیل اطلاعات بدون منبع نمی توان در دادگاه بعنوان مدرک جرم استفاده کرد).

وضعیت فعلی شبکه های کامپیوتری شبیه موقعیت کتابهای چاپی در ابتدای اختراع این صنعت بود: افراد عادی وسیله ای بدست آورده بودند تا با آن صدای خود را به گوش دیگران برسانند. اما این آزادی با خود تبعات اجتماعی، سیاسی و اخلاقی خاصی بدنبال داشت، که همچنان لاینحل باقی مانده است.

زندگی همیشه بدین منوال است: هر سکه ای دو رو دارد. اینترنت هم از این قاعده مستثنی نیست. امکان

دسترسی سریع و آسان به اطلاعات ارمغان اینترنت است، ولی خروارها اطلاعات منفی، غلط و گمراه کننده وجه دیگر آن است. راهتمایی بهداشتی که تازگی در اینترنت خوانده اید (و احتمالاً می خواهید به آن عمل کنید)، ممکنست از یک برنده جایزه نوبل آمده باشد، یا یک دانش آموز بازیگوش دبیرستانی.

شبکه های کامپیوتری انواع جدیدی از جرم و رفتارهای ضداجتماعی را نیز با خود آورده اند. هر روز که صندوق پستی خود را باز می کنید، دهها و صدها پیام مزخرف و بدرنخور در آن می بینید (و حتی کم کم به آن عادت کرده اید). این نتیجه کار افرادیست که میلیونها آدرس ایمیل را روی یک CD جمع کرده، و (بدون رضایت صاحبان این آدرسها) به این و آن می فروشند. و تازه اینها دسته بی آزارها هستند؛ این روزها کسی پیدا نمی شود که صابون و ویروسهایی که از طریق ایمیل منتشر می شوند، به تنش نخورده باشد.

دزدی هویت یکی دیگر از خطرات سرقت اطلاعات از طریق اینترنت است. (درباره سرقت های ادبی و نقض گسترده و وسیع قانون حق التالیف در اینترنت قبلاً هم صحبت کردیم).

بسیاری از این مشکلات حاصل ضعف (و یا عدم رعایت) مسائل امنیتی در اینترنت است. اگر تمام پیامهای ایمیل بصورت رمز در آیند، سرقت اطلاعات بسیار مشکلتر خواهد شد (این تکنولوژی توسعه زیادی یافته، که در فصل ۸ مفصلاً به آن خواهیم پرداخت). مشکل اینجاست که بالا بردن سطح ایمنی مترادف است با بالا رفتن هزینه، و این چیزی نیست که به آسانی پذیرفته شود. تعداد زیادی از این مسائل نیز به مشکلات و باگهای موجود در نرم افزارها مربوط می شود، که خود حاصل بزرگتر و پیچیده تر شدن آنهاست. اگر روی برنامه ها بر حسب بزرگی و پیچیدگی آنها مالیات بسته شود، شاید این مشکل تا حدی حل شود؛ البته خیلی ها این راه حل را نمی پسندند! پس دادن پول برنامه های معیوب نیز می تواند راه حل خوبی باشد، فقط مشکل اینجاست که چنین قانونی ظرف یک سال کل صنعت نرم افزار را ورشکست خواهد کرد!

۲-۱ سخت افزار شبکه

اکنون وقت آنست که توجه خود را از مسایل متفرقه به موضوع اصلی (یعنی همان شبکه های کامپیوتری) معطوف کنیم. هیچ طبقه بندی پذیرفته شده ای که در بر گیرنده تمام انواع شبکه های کامپیوتری باشد، وجود ندارد، ولی در این میان می توان به دو عامل مهم توجه کرد: تکنولوژی انتقال و اندازه شبکه. اجازه دهید این دو را جداگانه بررسی کنیم. امروزه دو تکنولوژی انتقال بیش از همه گسترش یافته و فراگیر هستند:

۱. ارتباطات پخش (broadcast)

۲. ارتباطات همتا به همتا (peer-to-peer)

شبکه های پخش (broadcast network) دارای یک کانال مخابراتی هستند که بین همه کامپیوترهای شبکه به اشتراک گذاشته شده است. هر یک از کامپیوترها می توانند پیامهای خود را در بسته (packet) های کوچک مخابره کنند، و تمام کامپیوترهای دیگر این پیامها را دریافت خواهند کرد. آدرس کامپیوتری که این بسته در حقیقت برای وی ارسال شده، در بخشی از پیام نوشته می شود. هر کامپیوتری به محض دریافت بسته، آدرس گیرنده را چک می کند؛ اگر پیام برای او باشد، آنرا پردازش می کند؛ ولی اگر پیام متعلق به دیگری باشد، پسادگی آنرا نادیده می گیرد. بعنوان مقایسه، فرض کنید کسی در انتهای راهرویی که در دو طرف آن پُر از اتاقهای متعدد است، فریاد بزند «آقای واتسون، ببانید. با شما کار دارم.» با اینکه این پیام به گوش همه افراد می رسد، فقط آقای واتسون به آن پاسخ می دهد و دیگران توجهی به آن نخواهد کرد. یا وقتی در سالن انتظار فرودگاه اعلام می شود که «مسافران پرواز ۶۴۴ به خروجی ۱۲ مراجعه کنند»، فقط آنهايي که بلیط این پرواز را دارند، عکس العمل نشان می دهند.

در شبکه های پخش با تعبیه یک کُد خاص در فیلد آدرس (address field) می توان یک پیام را به تمام

کامپیوترها ارسال کرد. چنین پیامی را همه کامپیوترها متعلق به خود تلقی کرده، و آنرا می خوانند. به این تکنیک پخش (broadcasting) گفته می شود. در برخی از سیستمهای پخش امکان ارسال پیام به دسته ای از کامپیوترها نیز وجود دارد، که به آن پخش گروهی (multicasting) می گویند. بدین منظور، معمولاً از یک بیت خاص در فیلد آدرس استفاده می شود، و همه آنهايي که این بیت در آنها وجود دارد عضو گروه محسوب شده و پیام را می گیرند. در شبکه های همنا به همنا (peer-to-peer network) بین تک تک کامپیوترها مسیر ارتباطی مستقل وجود دارد. البته وقتی یک بسته بخواهد از کامپیوتری به کامپیوتر دیگر برود، احتمالاً سر راه خود از چند ماشین بینابینی نیز عبور خواهد کرد. معمولاً در این قبیل شبکه ها مسیرهای متعددی بین دو کامپیوتر خاص می توان برقرار کرد، که از نظر طول مسیر با هم تفاوت دارند، و یافتن کوتاهترین مسیر یکی از مسایل مهم در این گونه شبکه ها است. بعنوان یک قاعده کلی (البته با استثنای متعدد)، شبکه های کوچک، متمرکز و محلی از نوع پخش هستند، و شبکه های بزرگ و گسترده از نوع همنا به همنا. به ارتباط همنا به همنا گاهی پخش تکي (unicasting) نیز گفته می شود. روش دیگر طبقه بندی شبکه ها اندازه شبکه است. در شکل ۱-۶ نوعی طبقه بندی بر اساس اندازه را مشاهده می کنید.

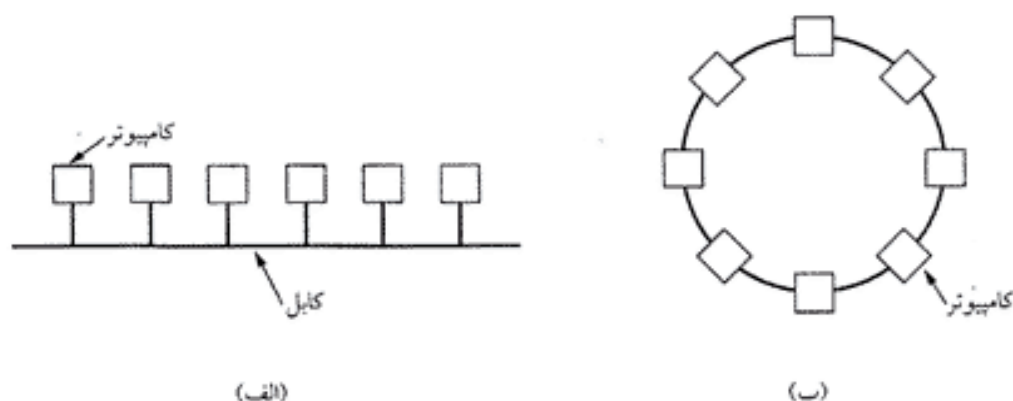
در بالا شبکه های شخصی (Personal Area Network) را می بینید (شبکه هایی که متعلق به یک فرد خاص هستند). ارتباط بیسیم بین ماوس، کی بورد، چاپگر، PDA و کامپیوتر از این نوع است. بعد از آن شبکه های محلی (LAN)، شهری (MAN) و گسترده (WAN) می آیند. در آخر هم شبکه شبکه ها (شبکه ای که هر نقطه از آن خود یک شبکه کامل است) - و اینترنت معروفترین نمونه آن است - می آید. در این طبقه بندی فاصله کامپیوترها اهمیت زیادی دارد، چون تکنولوژی ارتباطی به شدت به آن وابسته است. در این کتاب درباره تمام این شبکه ها صحبت خواهیم کرد. در زیر هر یک از این شبکه ها را مختصراً معرفی می کنیم.

۱.۲.۱ شبکه های محلی (Local Area Network)

شبکه محلی، یا LAN، شبکه ایست خصوصی واقع در یک ساختمان یا مجتمع، که حداکثر ابعاد آن یکی دو کیلومتر باشد. از این نوع شبکه معمولاً برای متصل کردن کامپیوترهای یک شرکت و به اشتراک گذاشتن منابع (مانند چاپگر) یا مبادله اطلاعات استفاده می شود. یک شبکه LAN سه مشخصه اصلی دارد، که آنرا از سایر انواع شبکه متمایز می کند: (۱) اندازه، (۲) تکنولوژی انتقال اطلاعات، و (۳) توپولوژی (topology). اندازه LAN بسیار محدود است، بگونه ای که زمان انتقال سیگنالها در آن (حتی در بدترین شرایط) بسیار کم و از قبل قابل پیش بینی است. دانستن این محدودیت ها برای طراحی شبکه بسیار مهم و اساسی است، و باعث ساده تر شدن مدیریت شبکه نیز می شود.

فاصله پردازنده ها	محل نسبی پردازنده ها	نمونه
1 m	روی یک میز	شبکه شخصی
10 m	یک اتاق	شبکه محلی
100 m	یک ساختمان	" "
1 km	یک مجتمع	" "
10 km	یک شهر	شبکه شهری
100 km	یک کشور	شبکه گسترده
1000 km	یک قاره	" "
10,000 km	کره زمین	اینترنت

شکل ۱-۶. طبقه بندی شبکه ها بر اساس اندازه و فاصله پردازنده ها.



شکل ۷-۱. دو شبکه پخش. (الف) باس. (ب) حلقوی.

تکنولوژی انتقال اطلاعات در LAN معمولاً به کابل متکیست (و از این نظر بسیار شبیه شبکه های تلفن است). سرعت انتقال اطلاعات در LAN بین ۱۰ تا ۱۰۰ میلیون بیت در ثانیه (که با Mbps مشخص می شود)، تأخیر انتشار در آن کم (در حد میکرو یا نانو ثانیه)، و خطا در آن بسیار اندک است. LAN های جدیدتر به سرعت 10 Gbps نیز دست یافته اند. سرعت انتقال در شبکه معمولاً با واحد مگابیت بر ثانیه (1,000,000 bits/sec) یا گیگابیت بر ثانیه (1,000,000,000 bits/sec) اندازه گیری می شود.

توپولوژی های مختلفی برای شبکه های محلی پخش وجود دارد، که در شکل ۷-۱ دو تا از آنها را می بینید. در یک شبکه باس (bus network - شبکه ای با کابل کشی خطی) در هر لحظه فقط یکی از کامپیوترها مجاز به استفاده از خط و ارسال اطلاعات است، و تمام ماشینهای دیگر بایستی در این مدت از ارسال هر گونه اطلاعات خودداری کنند. در این قبیل شبکه ها بایستی مکانیزمی برای حل اختلاف (در مواقعی که دو کامپیوتر همزمان با هم شروع به ارسال می کنند) وجود داشته باشد. این مکانیزم می تواند متمرکز (centralized) یا توزیع شده (distributed) باشد. یکی از مکانیزمهای حل اختلاف در شبکه های باس پخش IEEE 802.3 نام دارد (که به اترنت - Ethernet - نیز معروف است)، و با کنترل غیرمتمرکز در سرعتهای 10 Mbps تا 10 Gbps کار می کند. کامپیوترهای یک شبکه اترنت در هر زمانی می توانند اقدام به ارسال کنند، ولی اگر تصادمی بین آنها پیش آمد، هر یک از آنها مدتی (که بصورت تصادفی تعیین می شود) صبر کرده و دوباره سعی خواهد کرد.

نوع دیگری از شبکه های پخش، شبکه حلقوی (ring network) است. در یک شبکه حلقوی، هر بیت اطلاعات بصورت مستقل (و بدون اینکه بخواهد منتظر سایر بیت های بسته ای که به آن تعلق دارد، شود) در شبکه منتشر می شود. با توجه به سرعت بالای انتشار الکترونها در محیطهای رسانا، هر بیت حتی قبل از انتشار بیت های بعدی، می تواند بارها محیط شبکه را دور بزند. در این نوع شبکه هم بایستی مکانیزمی برای حل اختلاف بین کامپیوترهای متخاصم وجود داشته باشد. اغلب این مکانیزمها به نوعی نوبت بندی متکی هستند. یکی از این مکانیزمها IEEE 802.5 (یا IBM Token Ring) است، که در سرعتهای 4 Mbps و 16 Mbps کار می کند. FDDI یکی دیگر از شبکه های پخش حلقوی است.

نوع دیگری از تقسیم بندی شبکه های پخش بر حسب نحوه اختصاص کانال است، و به استاتیک و دینامیک تقسیم می شود. در اختصاص کانال استاتیک هر کامپیوتر برای مدت زمانی محدود و مشخص کانال را در دست می گیرد، و فقط در این بُرش زمانیست که می تواند اطلاعات ارسال کند. در این روش پهنای باند کانال بشدت هدر می رود، چون بسیار پیش می آید که وقتی نوبت به یک کامپیوتر می رسد، چیزی برای گفتن ندارد. به همین دلیل، سیستمهای امروزی اغلب پهنای باند را بصورت دینامیک (بر حسب نیاز) تخصیص می دهند.

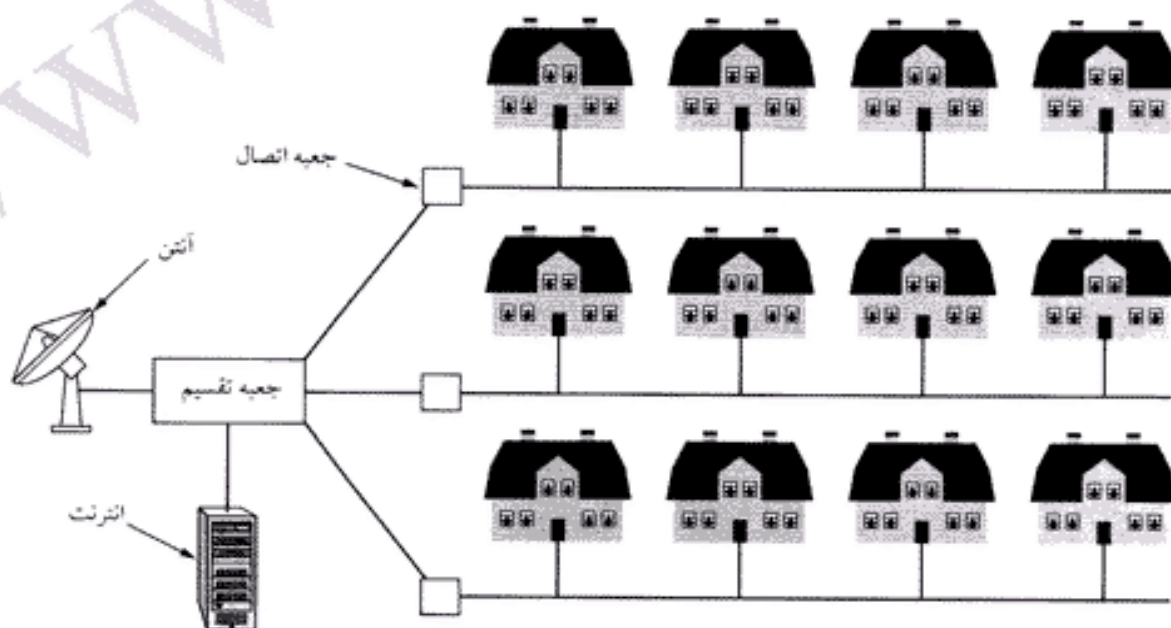
تخصیص دینامیک کانال خود بر دو نوع است: متمرکز و غیرمتمرکز. در نوع متمرکز یک موجودیت مشخص (بنام واحد تصمیم گیرنده - Arbitration Unit) وجود دارد، که درخواست ها را دریافت کرده، و بر اساس نوعی الگوریتم داخلی نوبت ها را تعیین می کند. در تخصیص کانال غیرمتمرکز این موجودیت تصمیم گیرنده وجود ندارد، و تصمیم گیری بر عهده تک تک کامپیوترهاست. شاید فکر کنید این روش جز هرج و مرج نتیجه ای ندارد، ولی چنین نیست (در آینده خواهید دید که الگوریتمهایی وجود دارند که می توانند به اوضاع سر و سامان بدهند).

۲.۲.۱ شبکه های شهری (Metropolitan Area Network)

شبکه شهری، یا MAN، شبکه ایست که یک شهر را پوشش می دهد. شبکه های تلویزیون کابلی بهترین نمونه MAN هستند. اولین شبکه های تلویزیون کابلی در نقاط کور شهرها راه اندازی شدند، بدین ترتیب که یک آنتن مرکزی و بزرگ در محلی که فرستنده اصلی را می دید نصب، و از این آنتن کابلهایی به مشترکان محروم از برنامه های تلویزیونی کشیده می شد.

در ابتدا این سیستمها بطور اختصاصی برای هر محل ساخته می شد، ولی بزودی شرکتهای بزرگ بوی پول را از آن استشمام کردند، و با کسب اجازه دولت تمام شهر را زیر پوشش کابلهای خود بردند. این شبکه ها برای پخش برنامه هم برنامه ریزی خاصی دارند، مثلاً یک شبکه فقط اخبار پخش می کند، دیگری فقط برنامه های ورزشی دارد، و آن یکی فقط آشپزی. این شبکه ها بسیار تخصصی بودند، و تا اواخر دهه ۱۹۹۰ فقط برنامه های تلویزیونی پخش می کردند.

با شروع گرایش عمومی به اینترنت، گردانندگان این شبکه ها بزودی دریافتند که با تغییری مختصر در سیستمهای خود می توانند از قسمتهای بالاستفاده پهنای باند برای ارائه سرویسهای دوطرفه اینترنت بهره ببرند. از این لحظه بود که شبکه های تلویزیون کابلی تبدیل به شبکه های شهری (MAN) شدند. در شکل ۸-۱ نمایی تقریبی از یک شبکه شهری را ملاحظه می کنید. در این شکل می بینید که ابتدا سیگنالهای تلویزیونی و اینترنتی ترکیب شده و به یک مرکز فوق-توزیع (head end) می روند، تا از آنجا در خانه های مشترکان توزیع شوند. (در فصل ۲ باز هم به این مبحث خواهیم پرداخت).



شکل ۸-۱ یک شبکه شهری مبتنی بر تلویزیون کابلی.

تلویزیون کابلی تنها مثال زنده MAN نیست. اخیراً تحقیقاتی بر روی اینترنت بیسیم پرسرعت (high-speed wireless Internet) انجام شده، که نتیجه آن نوع دیگری از MAN خواهد بود. با این استاندارد که IEEE 802.16 نام دارد، در فصل ۲ بیشتر آشنا خواهید شد.

۳-۲-۱ شبکه های گسترده (Wide Area Network)

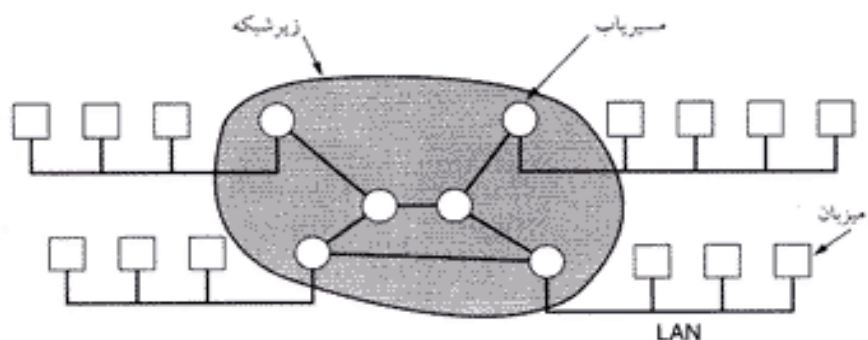
شبکه گسترده، یا WAN، گستره جغرافیایی بزرگی (مانند یک کشور یا قاره) دارد. در این نوع شبکه کامپیوترهایی هستند که برنامه های کاربردی روی آنها اجرا می شود، و معمولاً به آنها میزبان (host) می گویند. این کامپیوترها توسط زیرشبکه های مخابراتی (communication subnet) - یا بطور مختصر، زیرشبکه - به هم متصل می شوند. میزبانها متعلق به افراد هستند، در حالیکه زیرشبکه اغلب به شرکتهای مخابرات تعلق دارد. وظیفه زیرشبکه انتقال پیام از یک میزبان به میزبان دیگر است. جدا کردن این دو بخش (میزبانها و زیرشبکه) طراحی شبکه های WAN را تا حد زیادی ساده می کند.

در اغلب شبکه های گسترده، زیرشبکه از دو بخش مجزا تشکیل می شود: خطوط انتقال (transmission lines) و تجهیزات سوئیچینگ (switching elements). خطوط انتقال وظیفه رد و بدل کردن اطلاعات را بر عهده دارند، و می توان برای ایجاد آنها از سیم مسی، فیبر نوری یا حتی امواج رادیویی استفاده کرد. تجهیزات سوئیچینگ کامپیوترهای خاصی هستند که ارتباط بین خطوط انتقال را برقرار می کنند. وقتی داده ها از یک خط وارد می شود، این کامپیوتر باید مسیر خروجی آنها را مشخص کند. این کامپیوترهای سوئیچینگ به نامهای مختلفی خوانده می شوند، که می توان از معروفترین آنها به مسیریاب (router) اشاره کرد.

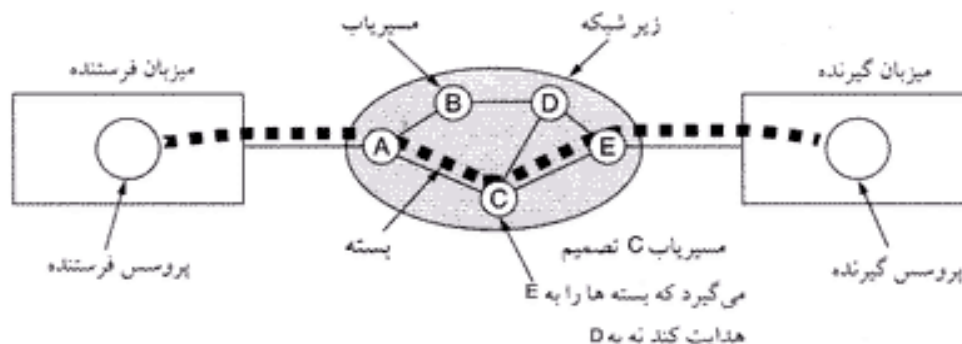
در این مدل (شکل ۹-۱) معمولاً هر کامپیوتر میزبان در یک شبکه محلی قرار دارد که از طریق یک مسیریاب به قسمتهای دیگر متصل می شود (البته در مواردی میزبان می تواند مستقیماً نیز به مسیریاب وصل باشد). به مجموعه خطوط مخابراتی و مسیریابها (منهای کامپیوترهای میزبان) زیرشبکه گفته می شود.

معنای اولیه زیرشبکه همان است که در بالا گفته شد، یعنی مجموعه خطوط مخابراتی و مسیریابها که وظیفه آنها انتقال اطلاعات از یک میزبان به میزبان دیگر است. اما سالها بعد از این اصطلاح در ارتباط با آدرس دهی شبکه ها نیز استفاده شد (فصل ۵ را ببینید). متأسفانه هنوز اصطلاح مناسبی برای کاربرد اولیه آن پیدا نشده است، و ما هم (با کمی شک و تردید) آنها را در هر دو مورد بکار خواهیم برد، که با توجه به موضوع بحث می توان معنای مورد نظر را استنباط کرد.

در بسیاری از WAN ها تعداد زیادی خطوط انتقال وجود دارد، که هر کدام یک جفت مسیریاب را به هم وصل می کنند. اگر دو مسیریاب که اتصال فیزیکی مستقیم ندارند، بخواهند با یکدیگر ارتباط برقرار کنند، باید این کار را بصورت غیرمستقیم (از طریق مسیریابهای دیگر) انجام دهند. وقتی یک بسته داده در مسیر خود (از مسیریاب



شکل ۹-۱. ارتباط بین کامپیوترهای میزبان و LAN ها در یک زیرشبکه.



شکل ۱-۱۰. استریم (جریان) بسته ها از مبدأ به مقصد.

مبدأ به مسیر یاب مقصد) از چند مسیر یاب بینایی عبور می کند، ابتدا بصورت کامل دریافت و ذخیره شده، و پس از آزاد شدن خط خروجی به سمت مقصد فرستاده می شود. زیر شبکه هایی که بر اساس این قاعده عمل می کنند، به زیر شبکه ذخیره-ارسال (store-and-forward) یا سوئیچ بسته (packet-switched) معروفند. تقریباً تمامی شبکه های WAN (بجز شبکه های ماهواره ای) از این نوع هستند. اگر اندازه بسته ها کوچک و یکسان باشد، به آنها سلول (cell) نیز گفته می شود.

به دلیل اهمیت مفهوم زیر شبکه سوئیچ بسته، لازمست کمی بیشتر درباره آن توضیح دهیم. وقتی پروسی در یک میزبان می خواهد پیامی به میزبان دیگر بفرستد، ابتدا آنرا به بسته های کوچکتر (که پشت سر هم شماره گذاری می شوند) تقسیم می کند. این بسته ها بصورت مستقل به طرف مقابل ارسال می شوند، و بعد از رسیدن تمامی آنها به مقصد، در آنجا دوباره به یکدیگر مونتاژ شده و پیام اصلی را می سازند (شکل ۱-۱۰ را ببینید).

در این شکل تمام بسته ها از طریق مسیر ACE به مقصد رسیده اند (در حالیکه مسیرهای ABDE و ACDE نیز وجود داشت). در برخی از شبکه ها این یک الزام است، یعنی تمام بسته های یک پیام باید از یک مسیر عبور کنند، در حالیکه در شبکه های دیگر این بسته ها می توانند از مسیرهای مختلف عبور کنند. البته، اگر مسیری بهترین مسیر ممکن باشد (مانند ACE در اینجا)، همه بسته ها از آن مسیر عبور خواهند کرد، حتی اگر شبکه چنین الزامی را تحمیل نکرده باشد.

تصمیم گیری درباره مسیر ارسال بسته ها امری داخلی است، یعنی هر مسیر یاب خود درباره آن تصمیم می گیرد. وقتی یک بسته به مسیر یاب A می رسد، این مسیر یاب A است که تصمیم می گیرد آنرا از طریق خط متصل به B بفرستد یا از خط متصل به C. مسیر یاب ها برای تصمیم گیری درباره مسیر بسته ها از الگوریتمهای مسیریابی (routing algorithm) استفاده می کنند، که درباره آنها در فصل ۵ صحبت خواهیم کرد.

تمام شبکه های WAN از نوع سوئیچ بسته نیستند، مانند سیستمهای ماهواره ای. در این سیستمها هر روتر آنتنی دارد که از طریق آن اطلاعات را به ماهواره می فرستد، یا اطلاعات ارسالی آن را دریافت می کند. تمام مسیر یاب های این مجموعه می توانند به ماهواره گوش کنند (و حتی برخی از آنها به اطلاعات ارسالی از مسیر یاب های همسایه نیز گوش می کنند). البته شبکه هایی هم وجود دارد که فقط برخی از مسیر یاب های آن (و نه همه آنها) ارتباط ماهواره ای دارند. شبکه های ماهواره ای ذاتاً از نوع پخش هستند، و اغلب در جاهایی بکار می روند که این طریقه پخش اهمیت داشته باشد.

۲-۱ شبکه های بیسیم (Wireless Network)

مخابرات دیجیتال بیسیم ایده جدیدی نیست. کُد موری که فیزیکدان ایتالیایی گاگلیلمو مارکونی در سال ۱۹۰۱ از یک کشتی به ساحل مخابره کرد، را می توان اولین پیام دیجیتال بیسیم محسوب کرد. سیستمهای جدید مخابرات

بیسیم فقط کارایی بهتری دارند، اما ایده اصلی در واقع همان است. در ساده‌ترین صورت، شبکه‌های بیسیم را می‌توان به سه دسته بزرگ تقسیم کرد:

۱. ارتباطات بین سیستمی
۲. LAN های بیسیم
۳. WAN های بیسیم

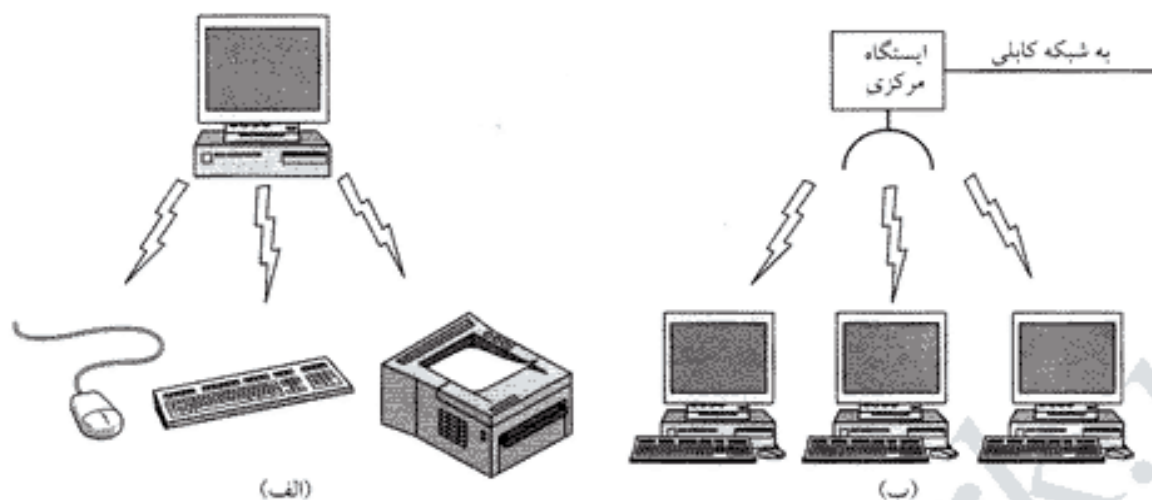
ارتباطات بین سیستمی (system interconnection) یعنی برقراری ارتباط بین قطعات داخلی یک کامپیوتر با استفاده از امواج رادیویی کوتاه بُرد. تقریباً هر کامپیوتری یک مانیتور، صفحه کلید یا ماوس دارد که معمولاً با کابل به آن متصل می‌شوند. برای بسیاری از کاربران خانگی (و حتی اداری) وصل کردن این کابلها (با اینکه آنها طوری طراحی شده‌اند که نتوان هیچکدام را به دیگری وصل کرد) یک کار شاق است، و برای این کار دست به دامان تکنسین‌های کامپیوتر می‌شوند. به همین علت، برخی از شرکتهای سازنده کامپیوتر دور هم جمع شدند، و یک شبکه بیسیم با بُرد کوتاه بنام بلوتوث (Bluetooth) اختراع کردند که این قطعات را بدون استفاده از سیم به کامپیوتر متصل می‌کند. تکنولوژی بلوتوث اجازه می‌دهد تا دستگاههایی مانند چاپگر، دوربین دیجیتال، گوشی، و اسکتر نیز (با قرار گرفتن در بُرد امواج آن) به کامپیوتر متصل شوند. برای این کار به هیچ اتصال فیزیکی یا حتی نصب درایور نیاز نیست، و فقط کافیست دستگاه را روشن کرده و در بُرد کامپیوتر قرار دهید، تا کار کند. برای بسیاری از کاربران این یک مزیت خارق‌العاده است.

ارتباطات بین سیستمی اساساً بر الگوی اصلی-پیرو (master-slave) مبتنی است (شکل ۱-۱۱ الف). در این سیستم، کامپیوتر اصلی است و با وسایل جانبی بعنوان رعایای خود صحبت می‌کند. این کامپیوتر اصلی است که به رعایا می‌گوید از چه آدرسی استفاده کنند، کی حرف بزنند، چه مدت حرف بزنند، روی چه فرکانسی صحبت کنند، و مانند آن. در باره تکنولوژی بلوتوث در فصل ۴ مفصلاً صحبت خواهیم کرد.

نوع دیگر ارتباطات بیسیم، شبکه محلی بیسیم (یا LAN بیسیم) است. در این سیستم هر کامپیوتر یک مودم رادیویی و یک آنتن دارد، که به وسیله آن با کامپیوترهای دیگر ارتباط برقرار می‌کند. در اغلب این سیستمها یک آنتن مرکزی روی پشت بام وجود دارد (شکل ۱-۱۱ ب)، که ارتباط بین کامپیوترها را تسهیل می‌کند، اما اگر شبکه باندازه کافی کوچک باشد، آنها می‌توانند مستقیماً با هم حرف بزنند. این نوع شبکه در دفاتر کوچک، خانه‌ها و جاهایی که کابل کشی مشکل است، سرعت در حال گسترش است. مهمترین استاندارد LAN های بیسیم IEEE 802.11 نام دارد، که در اغلب سیستمها از آن استفاده می‌شود.

نوع سوم ارتباطات بیسیم، سیستمهای WAN بیسیم است. شبکه رادیویی بکار رفته در سیستمهای تلفن همراه از این نوع است. این سیستمها اکنون نسل سوم خود را پشت سر می‌گذارند. نسل اول آنالوگ بود و فقط برای صدا از آن استفاده می‌شد. نسل دوم با اینکه دیجیتال شده بود، ولی باز هم فقط از صدا پشتیبانی می‌کرد. نسل سوم نیز دیجیتال است، و اینک همزمان از صدا و دیتا پشتیبانی می‌کند. WAN های بیسیم اساساً تفاوتی با LAN بیسیم ندارند، و فقط بُرد آنها بیشتر و البته نرخ انتقال داده‌ها کمتر است. LAN های بیسیم می‌توانند داده‌ها را با سرعتهایی در حد 50 Mbps (در محدوده چند ده متر) منتقل کنند. نرخ انتقال داده‌ها در WAN های بیسیم بزحمت به 1 Mbps می‌رسد، ولی بُرد آنها بجای متر با کیلومتر سنجیده می‌شود. در فصل ۲ درباره این سیستمها بسیار خواهیم گفت.

علاوه بر این شبکه‌های کم سرعت، اکنون WAN های بیسیم پُر ظرفیت نیز در دست توسعه است. دسترسی پُر سرعت به اینترنت از منزل و دفتر کار بدون استفاده از خطوط تلفن، از اولین کاربردهای این شبکه‌هاست. استاندارد این سیستم (که به آن سرویس توزیع چند نقطه‌ای محلی گفته می‌شود) IEEE 802.16 نام دارد، که درباره آن در فصل ۴ بیشتر صحبت خواهیم کرد.



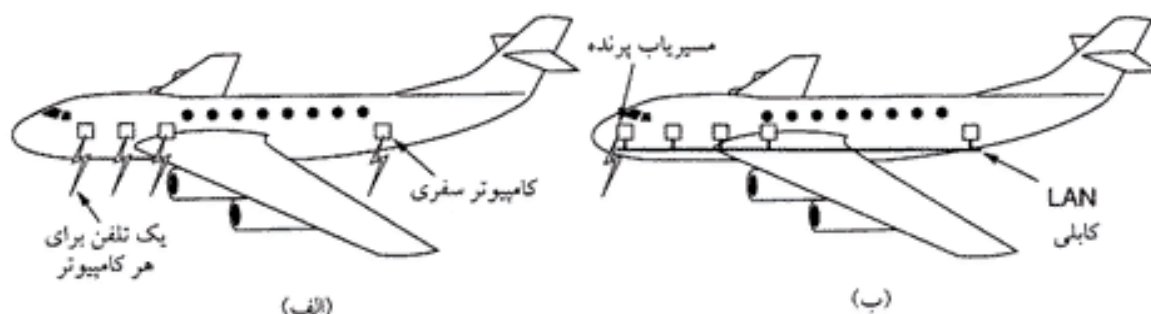
شکل ۱-۱۱. (الف) پیکربندی بلوتوث. (ب) LAN بیسیم.

تقریباً تمام شبکه‌های بیسیم باید در جایی به یک شبکه معمولی متصل شوند، تا بتوانند وظایف خود را انجام دهند؛ این کار را به روشهای مختلف می‌توان انجام داد. برای مثال، در شکل ۱-۱۲ الف هواپیمایی رامی‌بینید، که در آن تعدادی از مسافران مودمهای خود را به تلفنهای تعبیه شده در صندلی‌های هواپیما وصل کرده‌اند؛ این تماسها بکلی از هم مستقلند. اما در شکل ۱-۱۲ ب روش بهینه‌تری را ملاحظه می‌کنید، که در آن هر صندلی یک اتصال اینترنت دارد، و مجموعه آنها تشکیل یک LAN معمولی می‌دهند؛ و این LAN به یک مسیر یاب بیسیم وصل است که ارتباط آنها را با دنیای خارج برقرار می‌کند.

بسیاری افراد فکر می‌کنند که بیسیم موج آینده است (برای مثال، Bi et al., 2001؛ Leeper, 2001؛ Varshey and Vetter, 2000)، ولی حداقل یک صدای مخالف هم وجود دارد. باب متکالف، مخترع اینترنت، می‌گوید: «کامپیوترهای موبایل بیسیم مثل دستشویی‌های متحرک هستند - آنها فقط بدرد بیک نیک، کارگاه‌های ساختمانی و اردوهای کوتاه مدت می‌خورند. نصیحت من اینست که در خانه خود کابل اینترنت بکشید، و منتظر آینده بمانید» (Metcalfe, 1995). شاید تاریخ بعدها این اظهار نظر را در کنار جمله تاریخی تی جی واتسون رئیس IBM در سال ۱۹۴۵ بگذارد، که در پاسخ اینکه چرا IBM وارد بازار کامپیوتر نمی‌شود، گفته بود: «دنیا تا سال ۲۰۰۰ به چهار یا پنج کامپیوتر بیشتر نیاز نخواهد داشت».

۵-۲-۱ شبکه‌های خانگی (Home Network)

از هم اکنون می‌توان شبکه‌های خانگی را در افق آینده دید. ایده اصلی آن است که تمام یک وسایل یک خانه بتوانند



شکل ۱-۱۲. (الف) کامپیوترهای سفری منفرد. (ب) یک LAN پرنده.

با یکدیگر ارتباط برقرار کنند، و بتوان آنها را از طریق اینترنت کنترل کرد. این یکی از آن چیزهایی است که هیچکس منتظر آن نبوده (مانند، کنترل از راه دور تلویزیون و تلفن همراه)، ولی وقتی آمد دیگر هیچکس نمی تواند زندگی بدون آن را تصور کند.

وسایل زیادی را می توان در یک شبکه به هم متصل کرد، که از واضحترین آنها می توان به موارد زیر اشاره کرد:

۱. کامپیوترها (رومیزی، سفری، PDA، وسایل جانبی)
۲. وسایل سرگرمی (تلویزیون، DVD، ویدئو، دوربین دیجیتال، استریو، MP3)
۳. وسایل مخابراتی (تلفن معمولی و همراه، فکس، دستگاههای ارتباط داخلی)
۴. لوازم خانگی (میکرو، یخچال، ساعت، بخاری، تهویه مطبوع، چراغ)
۵. وسایل اندازه گیری از راه دور (آلارم دود یا دزدی، قرائت کنتور، ترموستات، دوربین اتاق بچه)

شبکه های خانگی به بسیاری از خانه ها راه یافته است؛ در این خانه ها وسایلی وجود دارد که یک ارتباط پرسرعت اینترنت را بین چند کامپیوتر به اشتراک می گذارند. با گسترش سرگرمی ها روی اینترنت، بزودی شاهد تلویزیونها و استریوهایی خواهیم بود که مستقیماً به اینترنت متصلند (و این ارتباط دو جانبه خواهد بود، چرا که شاید شما هم مایل باشید فیلمها و موزیکهای خود را با دوستان و آشنایان به اشتراک بگذارید). مخابرات بین المللی از هم اکنون یک کالای در دسترس است، ولی بزودی این سرویسها بصورت دیجیتال و از طریق اینترنت ارائه خواهند شد. امروزه کمتر خانه ای را می توان یافت که یک دوجین ساعت نداشته باشد، و با آمدن بهار و پائیز صاحبخانه مجبور است آنها را دستی جلو یا عقب بکشد؛ اگر تمام این ساعتها به اینترنت متصل باشند، می توان آنها را بصورت خودکار تنظیم کرد. کنترل خانه و مشاهده اتفاقاتی که در غیبت ما در آن می گذرد، یکی از آرزوهای دیرینه ماست، که اینک به واقعیت تبدیل شده است. (دیگر می توانید با خیال راحت به سینما بروید، و بچه ها در خانه تنها بگذارید!) شاید فکر کنید هر یک از این کاربردها به شبکه ای مجزا نیاز دارد، اما یکپارچه کردن آنها احتمالاً ایده بهتری است.

شبکه های خانگی تفاوت های ذاتی با سایر انواع شبکه دارد. اول اینکه نصب آن نباید پیچیده باشد. آنهایی که در این سالها درگیر کار نصب شبکه بوده اند، با جوابهای زیر (وقتی یک مشتری با مشکلی مواجه شده و به شما تلفن می زند) کاملاً آشنا هستند: (۱) دفترچه راهنما را با دقت بخوانید، (۲) کامپیوتر را دوباره بوت کنید، (۳) تمام سخت افزارها و نرم افزارهای اضافی - آنهایی که مال شرکت ما نیست! - را حذف کنید، (۴) جدیدترین درایور را از سایت وب ما بردارید و نصب کنید، و بالاخره وقتی هیچکدام از این کارها فایده ای نبخشید، (۵) کامپیوتر را فرمت و ویندوز را از نو نصب کنید. اگر به کسی که یک یخچال اینترنتی خریده، بگوئید آخرین ویرایش سیستم عامل یخچال اینترنتی را بار کرده و نصب کند، مسلماً باعث خوشحالی وی نخواهد شد! آنهایی که کامپیوتر می خرند، عادت دارند با سیستمهایی سر و کله بزنند که کار نمی کند، ولی خریداران اتومبیل، تلویزیون و یخچال این حرفها سرشان نمی شود؛ آنها انتظار دارند وسیله ای که خریده اند از اول بسماله بدون هیچ مشکلی کار کند.

دوم اینکه شبکه های خانگی باید بتوانند تحت هر شرایطی کار کنند. حتی همین حالا هم یک دستگاه کولر گازی با فقط چهار دکمه OFF، LOW، MEDIUM و HIGH، یک دفترچه راهنمای ۳۰ صفحه ای دارد. تصورش را بکنید وقتی این کولر اینترنتی شود، فقط قسمت امنیت آن دستکم ۳۰ صفحه خواهد بود! و این از حد تحمل قسمت اعظم کاربران این قبیل دستگاهها خارج است.

قیمت پائین سومین عامل برای موفقیت شبکه های خانگی است. خیلی از افراد حاضر نیستند ۵۰ دلار پول اضافه برای یک ترموستات اینترنتی بدهند، که چی، فقط درجه حرارت خانه خود را از اداره چک کنند! حتی ۵ دلار را هم شاید خیلی ها بزور بدهند.

از آنجائیکه دنیای آینده دنیای چندرسانه‌ایست، عامل چهارم در مقبولیت شبکه‌های خانگی بالا بودن پهنای باند آنهاست. مطمئن باشید هیچکس یک تلویزیون اینترنتی که فیلمها را با وضوح ۲۴۰×۳۲۰ پیکسل و ۱۰ فریم در ثانیه نشان می‌دهد، نخواهد خرید. حتی اینترنت (که در اغلب ادارات از آن استفاده می‌شود) نیز مناسب چندرسانه‌ای نیست. شبکه‌های خانگی برای آن که بتوانند فراگیر شوند، باید پهنای باند بیشتر را با قیمتی کمتر ارائه کنند.

پنجم، امکان گسترش شبکه‌های خانگی است. این به معنای آن است که جنگ استانداردها باید یک بار و برای همیشه حل شود. اگر امروز به مشتریان خود توصیه کنید وسایل IEEE 1394 (که به فایر وایر - FireWire - معروف است) را بخرند، و سال آینده بگویند امسال مد USB 2.0 است، مطمئن باشید همه آنها را پرانده‌اید. استانداردهای ارتباطی باید سالها (و کابل کشی، دهها سال) ثابت و بدون تغییر باقی بماند.

امنیت و قابلیت اعتماد ششمین عامل موفقیت شبکه‌های خانگی است. اینکه چند تا ایمیل و ویروسی از دست بدهید یک چیز است، و به باد رفتن تمام خانه در اثر لو رفتن کدهای امنیتی آن، یک چیز دیگر.

در این میان یکی از سؤالات جالب اینست که شبکه‌های خانگی باید بیسیم باشند یا با سیم. همین حالا هم در اغلب خانه‌ها شش شبکه وجود دارد: شبکه برق، تلفن، تلویزیون کابلی، آب، گاز و فاضلاب. اضافه کردن شبکه هفتم به خانه‌های نوساز هزینه چندانی ندارد، ولی در خانه‌های موجود کاری پُرخرج خواهد بود. شبکه‌های بیسیم کم خرج تر هستند، ولی شبکه‌های کابلی از نظر ایمنی بسیار قابل اعتمادترند. امواج رادیویی بسادگی از محدوده خانه خارج می‌شوند، و شاید یک همسایه فصول بتواند ایمیل‌های شما را بخواند، و یا دزدکی از اشتراک اینترنت شما استفاده کند. در فصل ۸ درباره روشهای رمزنگاری برای مقابله با این مشکلات صحبت خواهیم کرد، ولی در شبکه‌های خانگی موضوع امنیت از اهمیت چندگانه‌ای برخوردار است.

بطور خلاصه، شبکه‌های خانگی علاوه بر امکاناتی که ارائه می‌کنند، چالشهایی را نیز با خود به همراه دارند. اغلب این چالشها به مدیریت ساده، قابلیت اعتماد، ایمنی بالا (به‌ویژه در مورد کاربران غیرحرفه‌ای)، کارایی بالا و قیمت پائین مربوط می‌شوند.

۶-۲-۱ شبکه شبکه‌ها (Internetwork)

شبکه‌های متعددی با نرم‌افزارها و سخت‌افزارهای بسیار مختلف در سراسر دنیا وجود دارد، و بسیار پیش می‌آید که کاربری از یک شبکه بخواهد با کاربران شبکه‌های دیگر ارتباط برقرار کند. برای انجام این خواسته بایستی شبکه‌های مختلف (که بعضاً با هم ناسازگار هم هستند) با وسایلی بنام دروازه gateway - که می‌تواند سخت‌افزاری یا نرم‌افزاری باشد - به هم متصل شده، و داده‌ها از فرمتی به فرمت دیگر تبدیل شود. به مجموعه‌ای از این شبکه‌های بهم پیوسته شبکه شبکه‌ها (internetwork یا internet) گفته می‌شود. کلمه internet وقتی با i نوشته می‌شود، معنای عام می‌دهد، ولی با I همان شبکه جهانی اینترنت از آن مستفاد می‌شود.

متداولترین شکل شبکه شبکه‌ها عبارتست از تعدادی LAN که با ارتباطات WAN به هم متصل شده‌اند. در حقیقت، اگر در شکل WAN ۹-۱ را به جای subnet (زیرشبکه) قرار دهیم، هیچ چیز تغییر نخواهد کرد. در این مورد تنها تفاوت تکنیکی بین WAN و subnet وجود یا عدم وجود کامپیوترهای میزبان (host) است: اگر سیستم تاحیه خاکستری فقط از مسیر یاب تشکیل شده باشد، این یک زیرشبکه است؛ اگر در آن میزبان هم وجود داشته باشد، یک WAN است. تفاوت واقعی در مالکیت و طرز استفاده است.

اغلب افراد مفاهیم زیرشبکه (subnet)، شبکه (network)، و شبکه شبکه‌ها (internetwork) را با هم اشتباه می‌کنند. زیرشبکه (که بیشتر در شبکه‌های گسترده مفهوم پیدا می‌کند) مجموعه‌ایست از مسیر یاب‌ها و خطوط مخابراتی متعلق به راهبر شبکه. برای مقایسه، شبکه تلفن شهری از یک سری مراکز سوییچینگ، خطوط مخابراتی

پُرسرعت، و خطوط کم سرعت که مشترکان را به مرکز وصل می کند، تشکیل می شود. خطوط پُرسرعت و مراکز سونیچینگ (که متعلق به شرکت مخابرات هستند) همان زیرشبکه را می سازند. تلفنهای مشترکان که متعلق به افراد (و معادل میزبان) است، جزیی از زیرشبکه نیستند. مجموعه این دو (زیرشبکه و میزبانها) شبکه را می سازد. در شبکه های LAN (که فقط کامپیوتر و کابل است) زیرشبکه وجود ندارد.

وقتی چند تا از این شبکه ها به هم متصل می شوند، یک شبکه شبکه ها شکل می گیرد. با این تعریف، مجموعه دو LAN، یا یک LAN و یک WAN، را هم می توان شبکه شبکه ها نامید، ولی در صنعت کامپیوتر توافق یکپارچه ای درباره این اصطلاحات وجود ندارد. بعنوان یک قاعده کلی، اگر چند شرکت یا سازمان مختلف هر یک (به هزینه خود) قسمتی از یک شبکه را بنا کنند، با یک شبکه شبکه ها سروکار داریم. همچنین اگر تکنولوژی زیربنایی در قسمتهای مختلف متفاوت باشد، به احتمال زیاد دو شبکه داریم.

۳-۱ نرم افزار شبکه

در اولین شبکه های کامپیوتری سخت افزار از اهمیت ویژه ای برخوردار بود، و به نرم افزار فقط بعنوان چیزی که باید بعداً به آن فکر می شد، نگاه می کردند. اما این استراتژی دیگر کارایی ندارد. امروزه نرم افزار شبکه بسیار ساخت یافته است، که در این قسمت آنرا بررسی می کنیم. روشهای مورد بحث در این قسمت سنگ بنای کتاب را تشکیل می دهند، و در آینده بسیار به آنها مراجعه خواهیم کرد.

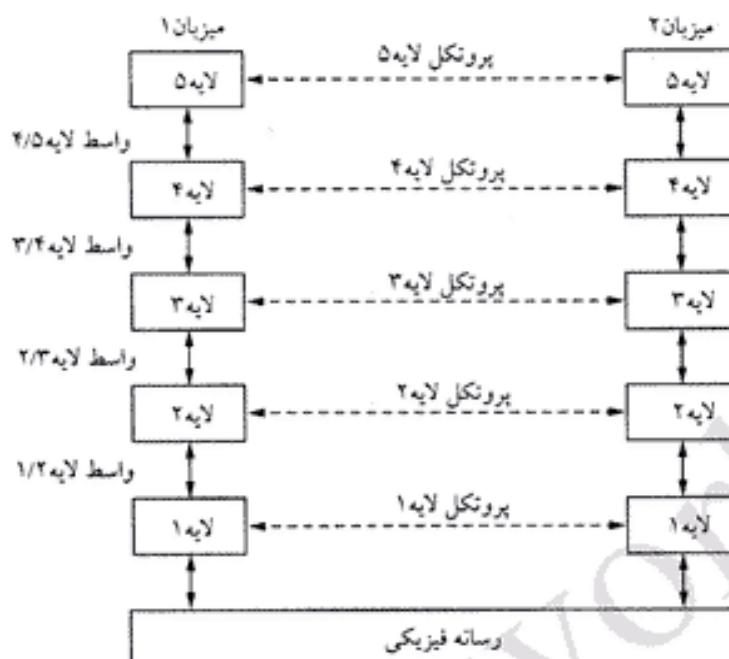
۱-۳-۱ سلسله مراتب پروتکل ها

برای کاهش پیچیدگیهای طراحی، اغلب شبکه ها بصورت مجموعه ای از چند لایه (layer) یا سطح (level) - که هر کدام روی دیگری قرار می گیرند - طراحی می شوند. تعداد لایه ها، نام هر لایه، محتوای آن، و کاری که هر لایه انجام می دهد، از شبکه ای به شبکه دیگر متفاوت است. وظیفه هر لایه ارائه سرویسهای خاص به لایه های بالاتر، و پنهان کردن جزئیات کار از دید آنهاست. در این مفهوم، هر لایه یک ماشین مجازی (virtual machine) است که سرویسهای خاصی را در اختیار لایه های بالاتر می گذارد.

این یکی از مفاهیم آشنا و کلیدی در کلیه علوم کامپیوتری است، که با نامهایی از قبیل پنهان کردن اطلاعات (information hiding)، انواع داده مجرد (abstract data types)، کپسولی کردن داده ها (data encapsulation) و برنامه نویسی شیءگرا (object-oriented programming) شناخته می شود. ایده اصلی این است که یک قطعه نرم افزار (یا سخت افزار) سرویسی را به کاربران خود عرضه کند، ولی جزئیات کار (از قبیل حالت داخلی خود و الگوریتمهای بکار رفته) را از آنها مخفی نگه دارد.

لایه n یک ماشین همیشه با لایه n ماشین دیگر حرف می زند. قواعد و قراردادهای این ارتباط را پروتکل لایه n (layer n protocol) می نامند. در ساده ترین حالت، پروتکل (protocol) عبارتست از قراردادهای توافق شده بین دو طرف برای برقراری و پیشبرد یک ارتباط. بعنوان مقایسه، وقتی یک خانم به یک آقا معرفی می شود، آن خانم می تواند دستش را پیش بیاورد؛ و بنوبه خود آن آقا هم می تواند دست خانم را (اگر یک شاهزاده اروپایی در یک میهمانی رسمی باشد) ببوسد، یا فقط با او دست بدهد (اگر آن خانم یک وکیل آمریکایی در یک جلسه کاری باشد). سرپیچی از پروتکل ها برقراری ارتباط را بسیار دشوار (اگر نگوئیم، غیرممکن) خواهد کرد.

در شکل ۱-۱۳ یک شبکه پنج لایه به تصویر کشیده شده است. به اجزایی که در یک لایه هستند، همتا (peer) گفته می شود. این همتاها می توانند پروسس های نرم افزاری، وسایل سخت افزاری، و یا حتی دو انسان باشند. عبارت دیگر، این همتاها هستند که با استفاده از پروتکل با هم رابطه برقرار می کنند.



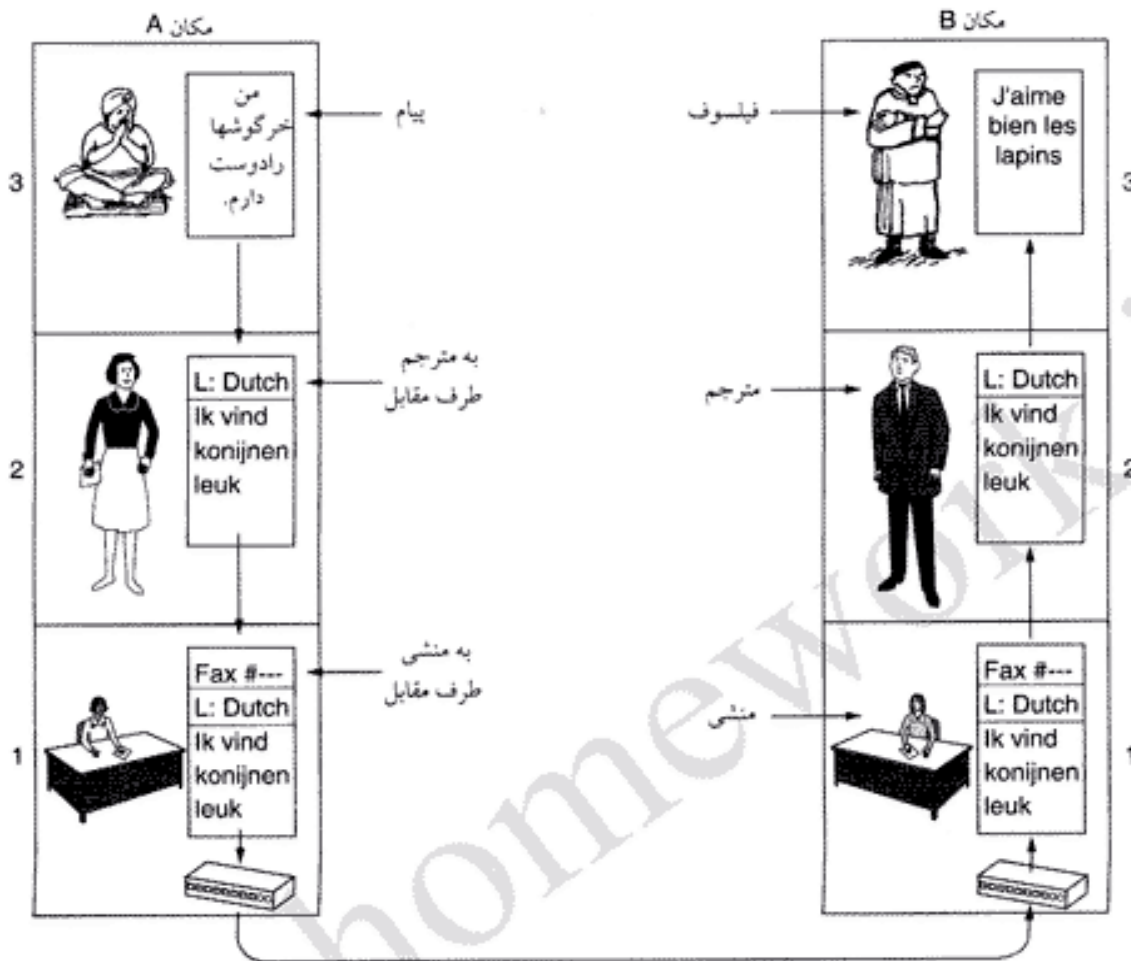
شکل ۱-۱۳. لایه ها، پروتکل ها، و واسط ها.

در حقیقت، داده ها هرگز مستقیماً از لایه n یک ماشین به لایه n ماشین دیگر منتقل نمی شوند. بلکه، هر لایه داده ها (و اطلاعات کنترلی) را به لایه زیرین خود می دهد، تا به پائین ترین لایه برسد. در زیر پائین ترین لایه (لایه ۱) رسانه فیزیکی (physical medium) قرار دارد، که داده ها را جایجا می کند. در شکل ۱-۱۳ ارتباط مجازی لایه ها با خط چین و ارتباط واقعی و فیزیکی با خط ممتد نشان داده شده است.

بین هر زوج از لایه های مجاور واسط (interface) قرار دارد. واسط مشخص می کند که هر لایه چه سرویسها و عملکردهای پایه ای در اختیار لایه بالاتر می گذارد. تعریف واسط های مناسب از مهمترین وظایف طراحان شبکه است. لازمه این امر آنست که وظایف هر لایه دقیقاً مشخص و شناخته شده باشد. علاوه بر به حداقل رساندن اطلاعات رد و بدل شده بین لایه ها، یک واسط شسته و رفته کار عوض کردن پیاده سازی لایه ها را نیز آسان می کند، چون تنها کاری که باید کرد این است که پیاده سازی جدید دقیقاً همان سرویسهای پیاده سازی قدیمی را به همسایگان خود ارائه کند. در حقیقت، پیاده سازیهای متعددی در شبکه های مختلف وجود دارد، که هیچ خللی در ارتباط لایه ها ایجاد نمی کنند.

به مجموعه لایه ها و پروتکل ها معماری شبکه (network architecture) می گویند. مشخصه های یک معماری باید آنچنان دقیق و جامع باشد که طراح شبکه بتواند نرم افزارها و سخت افزارهای لازم برای کارکرد صحیح آنرا فراهم آورد. جزئیات پیاده سازی و مشخصات واسط ها هرگز جزء معماری شبکه نیست، چون آنها باید در دل ماشین مخفی باشند (و از خارج دیده نشوند). حتی لازم نیست واسط ها در تمام ماشینهای یک شبکه یکسان باشند، مشروط باینکه تمام این ماشینها بتوانند از تمام پروتکل ها استفاده کنند. به مجموعه پروتکل هایی که در یک سیستم خاص بکار می روند (یک پروتکل در هر لایه)، پشته پروتکل (protocol stack) گفته می شود. پروتکل، پشته پروتکل و معماری شبکه از مهمترین موضوعات مورد بحث این کتاب هستند.

شاید یک مثال بتواند در روشن شدن مفهوم ارتباط چند لایه کمک کند. دو فیلسوف را، که اولی فقط زبانهای اردو و انگلیسی می داند و دیگری فقط چینی و فرانسه، در نظر بگیرید (آنها معادل پروسسهای همتا در لایه ۳ هستند). از آنجائیکه این دو فیلسوف نمی توانند مستقیماً با هم حرف بزنند، دو مترجم استخدام می کنند



شکل ۱-۱۴. معماری فیلسوف-مترجم-منشی.

(پرونده های همتا در لایه ۲)، که آنها هم بنویسند خود هر کدام یک منشی دارند (پرونده های همتا در لایه ۱). فیلسوف ۱ میل دارد علاقه خود به *oryctolagus cuniculus* را به فیلسوف ۲ (همتای خود) ابلاغ کند. برای اینکار، از طریق واسطه ۲/۳ یک پیام بزرگ انگلیسی با مضمون "I like rabbits" به مترجم خود می فرستد (شکل ۱-۱۴ را ببینید). مترجم ها بین خود توافق کرده اند که به زبان هلندی حرف بزنند، پس پیام فوق به "Ik vind konijnen leuk" تبدیل می شود. انتخاب زبانی که همتاهای لایه ۲ با آن صحبت کنند، بر عهده خود آنان (پرونده های این لایه) است.

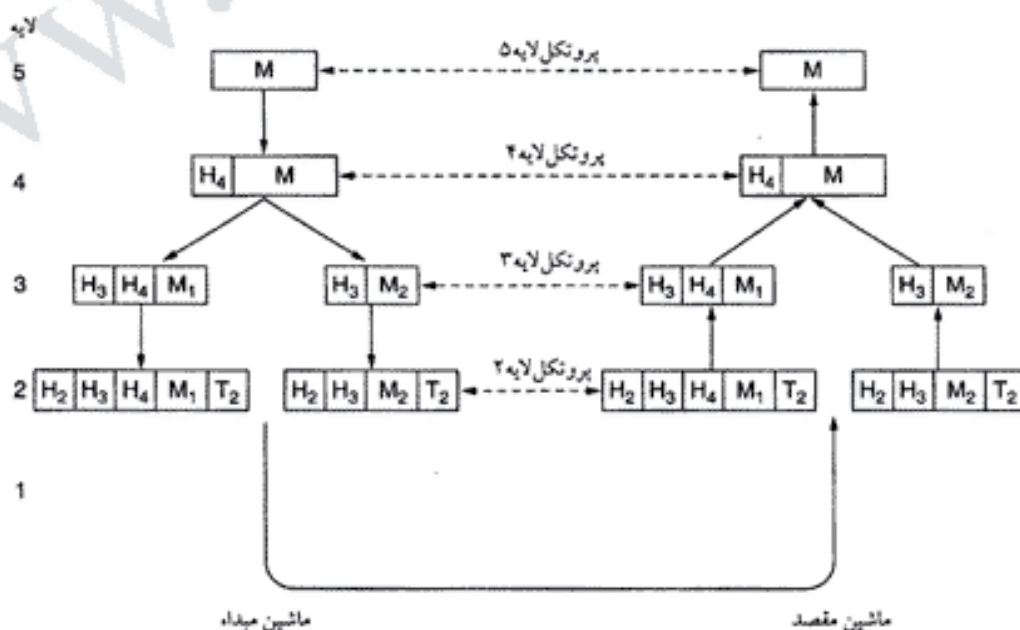
مترجم، سپس، این پیام را به منشی خود می دهد تا مثلاً از طریق فکس (پروتکل لایه ۱) ارسال شود. وقتی این پیام به منشی سمت مقابل رسید، آنرا به مترجم (لایه ۲) تحویل داد، و مترجم نیز پس از ترجمه به زبان فرانسه آنرا، از طریق واسطه ۲/۳، به فیلسوف ۲ می دهد. توجه کنید که تا زمانیکه واسطه ها تغییر نکرده باشند، پروتکل ها کاملاً از یکدیگر مستقل هستند. برای مثال، مترجم ها می توانند زبان توافقی خود را به فنلاندی تغییر دهند، مشروط باینکه واسطه آنها با لایه های ۱ و ۳ هیچ تغییری نکند. یا اینکه منشی ها می توانند از فکس به ایمیل یا تلفن سوئیچ کنند، بدون اینکه این کار هیچ تأثیری روی لایه های دیگر بگذارد. هر پروسس می تواند اطلاعات دلخواه خود را (که فقط بدرد پروسس همتای آن می خورد) به پیام اضافه کند. این اطلاعات به لایه های بالاتر منتقل نخواهند شد. حال یک مثال فنی تر را در نظر می گیریم: نحوه ارتباط در یک شبکه پنج لایه (شکل ۱-۱۵ را ببینید). برنامه ای

که در لایه ۵ اجرا می شود، یک پیام (M) تولید کرده و برای ارسال به لایه ۴ می دهد. لایه ۴ یک سرآیند (header) به این پیام اضافه، و آنرا به لایه ۳ تحویل می دهد. سرآیند حاوی اطلاعات کنترلی بین لایه های متناظر است؛ برای مثال، لایه ۴ می تواند به هر پیام یک عدد ترتیبی نسبت بدهد، تا اگر لایه های پائینتر آنها را بدون نظم و ترتیب ارسال کردند، لایه متناظر در سمت مقابل بتواند آنها را به ترتیب صحیح بازبایی کند. در برخی از لایه ها، این سرآیندها حاوی اندازه بسته، زمان ارسال و اطلاعاتی از این قبیل است.

در بسیاری از شبکه اندازه پیام در لایه ۴ هیچ محدودیتی ندارد، ولی (تقریباً همیشه) این محدودیت در لایه ۳ وجود دارد. در نتیجه، لایه ۳ باید پیام را به قطعات کوچکتر بشکند، و به هر قطعه یک سرآیند لایه ۳ اضافه کند. در این مثال، پیام M به دو قطعه M_1 و M_2 شکسته شده است.

سپس، لایه ۳ این بسته ها را به لایه بعدی (یعنی لایه ۲) تحویل می دهد. لایه ۲ علاوه بر اضافه کردن سرآیند خاص خود به ابتدای هر بسته، به انتهای آنها نیز یک دنباله (trailer) چسبانده، و آنها را به لایه بعدی (که لایه انتقال فیزیکی است) می دهد. در سمت مقابل، ماشین که این بسته ها را دریافت می کند، آنها را لایه به لایه بالا می فرستد، و هر لایه (قبل از تحویل به لایه بالاتر) اطلاعات خاص خود را از بسته ها برمی دارد. بدین ترتیب، هیچ لایه ای سرآیندها و دنباله های لایه های زیرین خود را دریافت نخواهد کرد.

نکته مهمی که در شکل ۱-۱۵ باید بدان توجه کنید، رابطه ارتباطات مجازی و حقیقی بین ماشینها، و تفاوت پروتکل ها و واسطه ها است. برای مثال، پروتکل های همتا در لایه ۴ فکر می کنند که ارتباط بین آنها یک ارتباط «افقی» است، که با استفاده از پروتکل لایه ۴ برقرار می شود. هر یک از این پروتکل ها احتمالاً ابزارهایی بنام *SendToOtherSide* (برای ارسال پیام به سمت مقابل) یا *GetFromOtherSide* (برای دریافت پیام از سمت مقابل) نیز دارند، و حتی نمی دانند که پیامهای خود را از طریق واسطه ۳/۴ به لایه پائین تر می دهند، نه به طرف مقابل. تجربیدی بودن رابطه پروتکل های همتا یک نکته کلیدی در طراحی شبکه است. با این تمهید، کار بسیار پیچیده و دشواری مانند طراحی کامل یک شبکه به کارهای کوچکتر و ساده تری (مانند طراحی لایه های جداگانه) شکسته می شود.



شکل ۱-۱۵. انتقال اطلاعات در یک شبکه پنج لایه.

با اینکه نام این بخش «نرم افزار شبکه» است، اما باید متوجه باشید که لایه های پائین تر معمولاً بصورت سخت افزاری پیاده سازی می شوند. با این حال، اینها نیز الگوریتمهای پیچیده نرم افزاری هستند، که فقط در سخت افزار حک و ثابت شده اند.

۲-۳-۱ ملاحظات در طراحی لایه ها

برخی از مفاهیم کلیدی طراحی شبکه های کامپیوتری در لایه های مختلفی حضور دارند، که در زیر برخی از مهمترین آنها را بطور مختصر بررسی خواهیم کرد.

هر لایه به مکانیزمی برای شناسایی فرستنده (sender) و گیرنده (receiver) نیاز دارد. از آنجائیکه یک شبکه معمولاً تعداد زیادی کامپیوتر دارد، و در هر کامپیوتر پروسس های متعددی در حال اجرا هستند، باید ابزاری وجود داشته باشد که هر پروسس بتواند پروسس همتای خود را دقیقاً شناسایی و مشخص کند. عبارت دیگر، برای تعیین دقیق مقصد به یک نظام آدرس دهی (addressing) نیاز داریم.

نحوه انتقال داده ها نیز نیازمند قواعد و مقررات خاص خود است. در برخی از سیستمها داده ها فقط در یک جهت حرکت می کنند، اما در برخی دیگر در هر دو جهت. همچنین پروتکل باید تعیین کند که هر ارتباط فیزیکی معادل چند کانال منطقی است، و اولویت بندی آنها چگونه است. در بسیاری از شبکه ها هر ارتباط فیزیکی حداقل دو کانال منطقی را شامل می شود، یکی برای داده های معمولی و دیگری برای داده های اضطراری.

مسئله مهم دیگر کنترل خطا (error control) است، چون هیچ ارتباط فیزیکی صد در صد کامل و عاری از خطا نیست. گداهای بسیاری برای کشف و تصحیح خطا وجود دارد، ولی هر دو طرف باید بر سر یکی از آنها توافق کنند. همچنین گیرنده باید بتواند به طریقی به فرستنده اعلام کند که کدام پیامها را درست دریافت کرده و کدامها را غلط.

همانطور که قبلاً هم گفتیم، در بسیاری از موارد بسته های تشکیل دهنده یک پیام بصورت منظم ارسال نمی شوند؛ در این حالت، پروتکل باید طوری طراحی شده باشد که گیرنده بتواند قطعات پیام را دوباره بنحو صحیح بهم بچسباند. یک راه حل شماره گذاری قطعات (بسته های) پیام است، ولی باز این سؤال باقیست که گیرنده باید چگونه با این بسته ها عمل کند.

یکی از مسائل که در تمام لایه ها وجود دارد اینست که، فرستنده با چه سرعتی باید اطلاعات را ارسال کند تا گیرنده های گنبد در گرداب داده ها غرق نشوند. برای این مشکل نیز راه حل های مختلفی وجود دارد که بعداً مفصلاً درباره آنها توضیح خواهیم داد. برخی از این راه حل ها شامل نوعی فیدبک (بازخور) از گیرنده به فرستنده است، که در هر لحظه وضعیت گیرنده را (بطور مستقیم یا غیر مستقیم) به فرستنده اعلام می کند. در برخی دیگر، دو طرف از قبل بر سر نرخ انتقال اطلاعات توافق می کنند. این مبحث کنترل جریان (flow control) نام دارد.

مسئله دیگری که در لایه های متعدد باید حل شود اینست که، اغلب پروسسها قادر نیستند پیامهایی با هر طول دلخواه دریافت کنند. این وضعیت باعث شده تا مکانیزمهایی برای شکستن پیامها به قطعات کوچک، ارسال، و سپس مونتاژ آنها در مقصد ابداع شود. مشکل دیگری که از اینجا ناشی می شود آنست که برخی از پروسسها اصرار دارند پیامها را آنقدر ریز کنند، بگونه ای که کارایی کل سیستم را مختل می کند. در اینجا راه حل چسباندن چند قطعه به یکدیگر و شکستن دوباره آنها در مقصد است.

در اغلب موارد ایجاد کانالهای ارتباطی جداگانه برای هر زوج پروسس کاری پرهزینه (و گاهی غیرممکن) است. در این موارد، لایه های پائین تر برای برقراری ارتباط بین چند پروسس مستقل، از یک کانال استفاده می کنند. این عمل (که مالتی پلکس - multiplexing - و دمالتی پلکس - demultiplexing - نام دارد) معمولاً بصورت شفاف انجام می شود، که در اینصورت می توان آنرا در هر لایه ای پیاده سازی کرد. برای مثال، وقتی تعداد خطوط

مخابراتی موجود محدود است، در لایه فیزیکی از تکنیکهای مالتی پلکس استفاده می شود. وقتی بین مبدأ و مقصد مسیرهای مختلفی وجود دارد، یکی از آنها باید انتخاب شود. گاهی تصمیم گیری در این مورد باید در چند لایه انجام شود. مثلاً، برای ارسال اطلاعات از لندن به رُم اولاً (در لایه های بالاتر) باید تصمیم بگیریم که از مسیر فرانسه استفاده کنیم یا از مسیر آلمان؛ سپس در لایه های پایینتر انتخاب مسیرهای خلوت تر از میان مسیرهای موجود پیش می آید. به این مبحث مسیریابی (routing) گفته می شود.

۳-۳-۱ سرویسهای اتصال-گرا و غیرمتصل

هر لایه می تواند دو نوع سرویس در اختیار لایه بالاتر از خود بگذارد: سرویس اتصال-گرا (connection-oriented) و سرویس غیرمتصل (connectionless). در این قسمت این سرویس ها و تفاوت های آنها را بررسی خواهیم کرد.

سرویس اتصال-گرا بر اساس مدل سیستمهای تلفن کار می کند. وقتی می خواهید با یک نفر تماس بگیرید، گوشی تلفن را برداشته، شماره می گیرید، صحبت می کنید، و بعد گوشی را می گذارید. در یک سرویس اتصال-گرا هم ابتدا اتصال برقرار شده، و بعد از تبادل اطلاعات موردنظر، اتصال قطع می شود. مهمترین نکته در مورد سرویسهای اتصال-گرا اینست که آنها مانند یک لوله عمل می کنند: فرستنده از یک طرف داده ها (بیت ها) را به داخل لوله می فرستند، و گیرنده در طرف دیگر آنها را می گیرد. در اغلب موارد داده ها بهمان ترتیبی که فرستاده شده اند، دریافت می شوند.

در برخی موارد بعد از برقراری اتصال، فرستنده، گیرنده و زیر شبکه ابتدا یک سری مذاکرات اولیه (negotiation) انجام می دهند تا بر سر مواردی از قبیل حداکثر اندازه پیامها، کیفیت سرویس موردنظر، و مانند آن توافق کنند. معمولاً، یک طرف پیشنهادی می دهد و طرفهای دیگر آنرا قبول یا رد کرده، و یا بکلی پیشنهاد جدیدی ارائه می کنند.

از سوی دیگر، سرویس غیرمتصل بر اساس مدل پست بنا شده است. هر پیام (نامه) دارای آدرس مشخصی است، و مسیری که برای رسیدن به مقصد طی می کند، کاملاً مستقل از پیامهای دیگر است. معمولاً وقتی دو نامه به یک مقصد می فرستید، اولین نامه زودتر از دومی به آنجا می رسد؛ ولی گاهی پیش می آید که اولی با تأخیر و بعد از دومی به مقصد برسد.

برای هر سرویس می توان یک کیفیت سرویس (quality of service) در نظر گرفت. برخی از سرویسها مطمئن و قابل اعتماد هستند، بگونه ای که هیچ داده ای در حین انتقال از بین نمی رود. یک سرویس قابل اعتماد معمولاً بگونه ای طراحی می شود که گیرنده دریافت صحیح داده ها را به فرستنده اعلام کند. این تصدیق دریافت (acknowledgement) باعث تحمیل یک بار اضافی و تأخیر در انتقال پیامها می شود، که اغلب ارزش آنرا دارد، ولی گاهی به زحمتش نمی ارزد.

انتقال فایل (file transfer) از جمله مواردیست که به یک سرویس مطمئن اتصال-گرا نیاز دارد؛ صاحب فایل معمولاً میل دارد تمام بیت های فایلش (با همان نظم و ترتیب) به مقصد برسد. کمتر کسی را پیدا می کنید که راضی شود یک فایل قروقاطی دریافت کند، حتی اگر اینکار به معنای سرعت بیشتر باشد.

سرویس اتصال-گرای قابل اعتماد بر دو گونه مختلف است: توالی پیام (message sequence) و جریان بایت (byte stream). در سرویس توالی پیام حد و مرز پیامها همیشه حفظ می شود: وقتی دو پیام ۱۰۲۴ بایتی می فرستید، طرف مقابل همیشه دو پیام ۱۰۲۴ بایتی دریافت خواهد کرد، نه یک پیام ۲۰۴۸ بایتی، یا چهار پیام ۵۱۲ بایتی. اما در سرویس دوم، چیزی بنام حد و مرز پیام وجود ندارد، و فقط جریانی از بایتها دیده خواهد شد. در این حالت وقتی ۲۰۴۸ بایت به مقصد می رسد، به هیچ طریقی نمی توان گفت که آیا این یک پیام ۲۰۴۸ بایتی بوده، یا دو

پیام ۱۰۲۴ بایتی، و یا ۲۰۴۸ پیام ۱ بایتی. برای مثال، اگر بخواهید صفحات یک کتاب را به یک دستگاه حروفچینی الکترونیکی بفرستید، شاید برایتان مهم باشد که حد و مرز هر صفحه مشخص باشد. از طرف دیگر، وقتی از راه دور به یک کامپیوتر متصل می شوید، جریان بایت ها از کامپیوتر مبدأ به مقصد تمام آن چیز است که نیاز دارید، و حد و مرز پیامها هیچ اهمیتی ندارد.

اما همانطور که گفتیم، در برخی از موارد تأخیری که در نتیجه تصدیق دریافت (acknowledgement) پدید می آید، غیر قابل قبول است؛ مکالمه دیجیتال یکی از این موارد است. اغلب کاربران ترجیح می دهند گاهی صدای طرف مقابل را با نویز بشنوند، تا اینکه (در نتیجه مکانیزم تصدیق دریافت) مکالمه با تأخیر و وقفه انجام شود. و یا در کنفرانسهای ویدئویی کمی برفک و نویز قابل تحمل تر است، تا اینکه تصاویر با پرشهای اعصاب خردکن دریافت شود.

از طرف دیگر، همه کاربردها به ارتباط متصل نیاز ندارند؛ پست الکترونیک (ایمیل) یکی از مواردیست که نیازی به سرویس اتصال-گرا ندارد، و بویژه در مواقعی که هزینه عملی تعیین کننده است، سرویس قابل اعتماد نیز چندان الزامی نیست. در این موارد فقط کافیسیت اولویت تحویل پیام بالا باشد. سرویس غیر متصل غیر قابل اعتماد (سرویس که به تصدیق دریافت از طرف مقابل متکی نیست) اغلب بعلت شباهتی که با سیستم تلگراف دارد، سرویس دیتاگرام (datagram service) نامیده می شود.

اما گاهی با اینکه نیازی به برقراری یک اتصال نیست (مثلاً برای ارسال یک پیام کوتاه)، ولی قابل اعتماد بودن ارتباط اهمیت دارد. در این قبیل موارد می توان از سرویس دیتاگرام همراه با تصدیق دریافت (acknowledged datagram service) استفاده کرد. این مانند پست کردن یک نامه سفارشی است، که فرستنده می خواهد از رسیدن نامه بدست گیرنده مطمئن شود. با دریافت این تأییدیه، فرستنده مطمئن می شود که نامه گم نشده و بدست طرف مقابل رسیده است.

سرویس دیگری نیز وجود دارد، که سرویس درخواست-پاسخ (request-reply service) نام دارد. در این سرویس، فرستنده دیتاگرامی که حاوی یک درخواست است ارسال می کند، و پاسخ آنرا می گیرد. برای مثال، وقتی پیامی به کتابخانه محلی می فرستید و سؤال می کنید که «زبان سواحیلی در کدام کشور تکلم می شود»، از این سرویس استفاده می کنید. از سرویس درخواست-پاسخ معمولاً در سیستمهای مشتری-سرویس دهنده استفاده می شود: مشتری درخواست خود را به سرویس دهنده فرستاده، و پاسخ آنرا دریافت می کند. در شکل ۱-۱۶ خلاصه ای از سرویسهای توضیح داده شده در این قسمت را مشاهده می کنید.

	مثال	سرویس
اتصال گرا	چند صفحه متوالی	استریم پیام قابل اعتماد
	ورود از راه دور	استریم بایت قابل اعتماد
	صدای دیجیتالی	اتصال غیر قابل اعتماد
غیر متصل		دیتا گرام غیر قابل اعتماد
	ایمیل ثبت شده	دیتا گرام تصدیق شده
	جستجوی پایگاه داده	درخواست - پاسخ

شکل ۱-۱۶. شش نوع سرویس مختلف.

شاید در نگاه اول تعجب کنید که اصولاً چرا باید از یک سرویس غیرقابل اعتماد استفاده کنیم، و اصلاً چه کسی چنین چیزی را لازم دارد؟ اول از همه اینکه، امکان دارد در مواردی سرویس قابل اعتماد اساساً در دسترس نباشد. برای مثال، اینترنت یک ارتباط قابل اعتماد نیست، و گاهی ممکنست داده‌ها در حین انتقال صدمه ببینند؛ این بر عهده پروتکل‌های لایه‌های بالاتر است که این مشکل را حل کنند. دوم اینکه، تأخیر ذاتی سرویس‌های مبتنی بر تصدیق دریافت در مواردی (مانند برنامه‌های چندرسانه‌ای) پذیرفتنی نیست. به این دلایل، وجود سرویس‌های قابل اعتماد و غیرقابل اعتماد هر دو لازم است.

۴.۳-۱ عملکردهای پایه سرویس

هر سرویس با یک سری عملکردهای پایه (primitives) که در اختیار کاربر خود می‌گذارد، شناخته می‌شود. این عملکردهای پایه یا خود کاری را انجام می‌دهند، و یا انجام کاری را در طرف مقابل گزارش می‌کنند. اگر پشته پروتکل (protocol stack) جزئی از سیستم عامل باشد (که اغلب نیز چنین است)، عملکردهای پایه نیز معمولاً جزء فراخوانی‌های سیستم (system call) هستند. این فراخوانی‌ها باعث فعال شدن کُدی در هسته سیستم عامل شده، و ارسال بسته‌های پیام انجام می‌شود.

عملکردهای پایه هر سرویس به خصلت آن سرویس بستگی دارد. برای مثال، عملکردهای پایه یک سرویس اتصال-گرا متفاوت از سرویس‌های غیرمتصل است. در شکل ۱-۱۷ حداقل عملکردهای پایه لازم برای پیاده‌سازی یک سرویس اتصال-گرای جریان بابت قابل اعتماد را در یک محیط مشتری-سرویس دهنده ملاحظه می‌کنید. طرز استفاده از این عملکردهای پایه مانند زیر است. ابتدا، کامپیوتر سرویس دهنده LISTEN را اجرا می‌کند تا نشان دهد که آماده پذیرش ارتباطات ورودی است. عملکرد LISTEN معمولاً بصورت یک فراخوانی مسدودشونده (blocking) پیاده‌سازی می‌شود؛ بدین معنا که بعد از اجرای این عملکرد، پروسس سرویس دهنده تا زمان دریافت درخواست اتصال مسدود می‌شود.

سپس، مشتری عملکرد CONNECT را اجرا می‌کند تا به سرویس دهنده متصل شود. فراخوانی CONNECT معمولاً باید مشخص کند که مقصد اتصال کجاست، بهمن دلیل ممکنست پارامتری داشته باشد که آدرس سرویس دهنده را بدست می‌دهد. با این عمل، سیستم عامل مشتری پیامی را به سمتی خود می‌فرستد، و درخواست اتصال می‌کند - این مرحله با شماره (1) در شکل ۱-۱۸ نشان داده شده است. پس از آن، پروسس مشتری تا زمان دریافت پاسخ به حالت تعلیق (suspend) درمی‌آید. وقتی این بسته به سرویس دهنده رسید، توسط سیستم عامل پردازش می‌شود، و وقتی می‌بیند یک درخواست اتصال است، بدنبال یک پروسس شتونده (listener) می‌گردد. اگر چنین پروسسی را پیدا کند، آنرا از حالت انسداد خارج کرده، و یک پیام تصدیق دریافت (acknowledgement) به مشتری پس می‌فرستد - مرحله (2). دریافت این پیام توسط مشتری باعث می‌شود تا پروسس از حالت تعلیق در آید. در این لحظه پروسس‌های سرویس دهنده و مشتری هر دو در حال اجرا هستند، و

مفهوم	عملکرد پایه
انتظار برای دریافت اتصال	LISTEN
برقراری ارتباط با سمتی منتظر	CONNECT
انتظار برای دریافت اتصال	RECEIVE
ارسال پیام به سمتی	SEND
پایان اتصال	DISCONNECT

شکل ۱-۱۷. پنج عملکرد پایه لازم برای پیاده‌سازی یک سرویس اتصال-گرای ساده.



شکل ۱-۱۸. تبادل بسته ها در یک شبکه اتصال-گرای مشتری-سرویس دهنده.

اتصال برقرار شده است. توجه به این نکته ضروریست که پیام تصدیق دریافت (2) توسط کُد پروتکل (که در سطح هسته - kernel level - اجرا می شود) ایجاد می شود، نه کُد عملکرد پایه (که یک پروسس سطح کاربر - user level - است). اگر هنگام دریافت درخواست اتصال توسط سرویس دهنده، هیچ پروسس شنونده ای وجود نداشته باشد، نتیجه نامشخص است. در برخی از سیستمها، این درخواست برای مدتی در صف (queue) می ماند، به امید اینکه شاید یک پروسس LISTEN اجرا شود.

این فرآیند بسیار شبیه تماس تلفنی مشتری با مدیر قسمت پشتیبانی مشتریان در یک شرکت است. نشستن مدیر قسمت پشتیبانی در کنار تلفن بنوعی اعلام آمادگی برای دریافت تقاضاها است (همان پروسس LISTEN). سپس یکی از مشتریان زنگ می زند (پروسس CONNECT)؛ و به محض اینکه مدیر پشتیبانی گوشی تلفن را برداشت، ارتباط برقرار می شود.

قدم بعدی را باید سرویس دهنده بردارد: اجرای عملکرد RECEIVE برای دریافت اولین درخواست. معمولاً سرویس دهنده این کار را بلافاصله بعد از برطرف شدن انسداد (و حتی قبل از اینکه پاسخ تصدیق دریافت آن به مشتری برسد) انجام می دهد. فراخوانی RECEIVE نیز جزء پروسسهای مسدودشونده است. وقتی مشتری اعلام آمادگی سرویس دهنده را دریافت کرد، درخواست خود را در قالب یک عملکرد SEND به آن می فرستد - مرحله (3).

رسیدن بسته SEND به سرویس دهنده آنرا از حالت انسداد خارج کرده، و باعث می شود تا بتواند به درخواست مشتری رسیدگی کند. پس از آماده شدن پاسخ، سرویس دهنده آنرا با اجرای عملکرد SEND به مشتری پس می فرستد - مرحله (4). رسیدن این بسته به مشتری باعث می شود تا از حالت انسداد خارج شده، و محتوای پاسخ را بررسی کند. اگر مشتری درخواست های بیشتری داشته باشد، آنها را در همین مرحله انجام می دهد؛ در غیر اینصورت، با اجرای عملکرد DISCONNECT اتصال را قطع می کند. فراخوانی DISCONNECT نیز معمولاً یک فراخوانی مسدودکننده است، و باعث تعلیق پروسس مشتری و اعلام پایان ارتباط به سرویس دهنده می شود - مرحله (5). پروسس سرویس دهنده با دریافت پیام DISCONNECT از مشتری، عملکرد DISCONNECT را در سمت خود اجرا کرده، و (پس از اعلام به مشتری) ارتباط را قطع می کند. وقتی بسته سرویس دهنده به مشتری رسید، مشتری نیز اتصال را رها می کند - مرحله (6). این بود ماجرای ساده یک ارتباط اتصال-گرا!

البته، زندگی همیشه این قدر شیرین نیست، و خیلی چیزها می تواند آنرا تلخ کند. برای مثال، فرض کنید عملکرد CONNECT قبل از LISTEN اجرا شود، یکی از بسته ها وسط راه گم شود، و اتفاقاتی از این قبیل. بعداً درباره این مسائل بیشتر صحبت خواهیم کرد، اما فعلاً به همین ارتباط ساده شکل ۱-۱۸ دقت کنید. با توجه به این نکته که برای کامل شدن سیکل این پروتکل به شش بسته نیاز داریم، شاید بپرسید که چرا بجای آن از یک ارتباط غیر متصل (که فقط به دو بسته - یک درخواست و یک پاسخ - نیاز دارد) استفاده نکنیم. جواب اینست

که در یک دنیای بی عیب و نقص می توان چنین کرد؛ اما آیا اوضاع همیشه بر وفق مراد ماست؟ اگر فایل ها خیلی بزرگ باشند، خط ارتباطی پُر از خطا باشد، بسته ها مدام گم شوند، چکار باید کرد؟ اگر پای صدها و هزارها بسته در میان باشد، و فقط چند بسته ناقابل گم شود، مشتری چگونه باید این موضوع را متوجه شود؟ مشتری از کجا بفهمد آخرین بسته ای که گرفته، واقعاً آخرین بسته بوده، یا خیلی ساده ارتباط قطع شده است؟ فرض کنید، مشتری فایل دومی را هم درخواست می کند، و بعد یکبار بسته ای می رسد که شماره ۱ دارد. آیا این اولین بسته فایل دوم است، یا اولین بسته فایل اول که تا حالا سرگردان بوده، و همین الان راه خود را پیدا کرده؟ خلاصه، در دنیای واقعی یک شبکه غیرقابل اعتماد مبتنی بر پروتکل درخواست-پاسخ نمی تواند کافی باشد. در فصل ۳ پروتکل هایی را که برای رفع این مشکلات ابداع شده اند، به تفصیل بررسی خواهیم کرد. فعلاً همین قدر کفایت بدانید که، گاهی پروتکل هایی که به روش جریان بایت با هم ارتباط برقرار می کنند، بهترین گزینه اند.

۵-۳-۱ رابطه سرویس و پروتکل

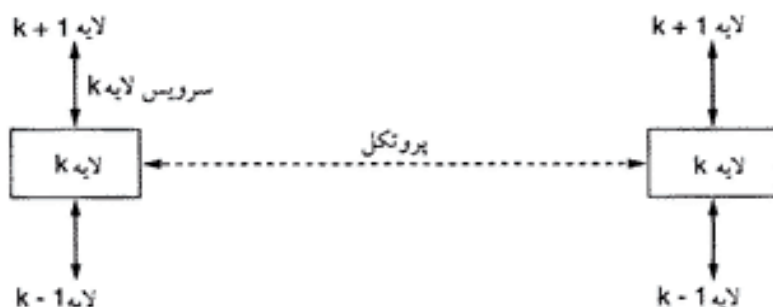
سرویس و پروتکل دو چیز متفاوتند، که اغلب افراد آنها را با هم اشتباه می گیرند. اما این تفاوت بقدری مهم است، که جا دارد باز هم بر آن تأکید کنیم. سرویس (service) عبارتست از مجموعه ای از عملکردهای پایه که یک لایه در اختیار لایه بالاتر از خود قرار می دهد. سرویس فقط می گوید که یک لایه چه کارهایی می تواند برای کاربر خود انجام دهد، ولی هیچ چیز درباره چگونگی آن نمی گوید. سرویس در واقع به واسطه بین دو لایه مربوط می شود، که در آن لایه پائین تر ارائه دهنده سرویس و لایه بالاتر مصرف کننده سرویس است.

اما، پروتکل (protocol) عبارتست از مجموعه قواعد حاکم بر فرمت، مفهوم و نحوه تبادل بسته ها و پیامها بین دو لایه همتا. در واقع این پروتکل است که سرویسهای تعریف شده در هر لایه را پیاده سازی می کند. همتاها هر لایه می توانند پروتکل ارتباطی خود را عوض کنند، مشروط بر اینکه سرویسهایی که به کاربران خود می دهند، تغییری نکنند. با این تعریف، سرویس و پروتکل کاملاً از هم مستقل هستند.

همانطور که در شکل ۱-۱۹ می بینید، سرویس به واسطه دو لایه مربوط می شود، در حالیکه پروتکل پیامهای مبادله شده بین دو لایه همتا (در دو کامپیوتر مختلف) را کنترل می کند. بسیار اهمیت دارد که این دو مفهوم را با هم مخلوط نکنید.

یک مثال مشابه از زبانهای برنامه نویسی می تواند در درک تمایز سرویس و پروتکل کمک کند. سرویس در واقع شبیه نوع داده (data type) یا شیء (object) در زبانهای شیءگرا است. با اینکه می دانیم یک شیء چه خواصی دارد، اما نمی دانیم آنها را چگونه پیاده سازی کرده است. این پیاده سازی همان پروتکل است که از چشم کاربر مخفی می ماند.

البته در بسیاری از پروتکل های قدیمی این تمایز بروشنی وجود ندارد. در این پروتکلها هر تغییری در پروتکل بلافاصله به چشم کاربر خواهد آمد. اما امروزه طراحان شبکه سعی می کنند این تمایز را رعایت کنند.



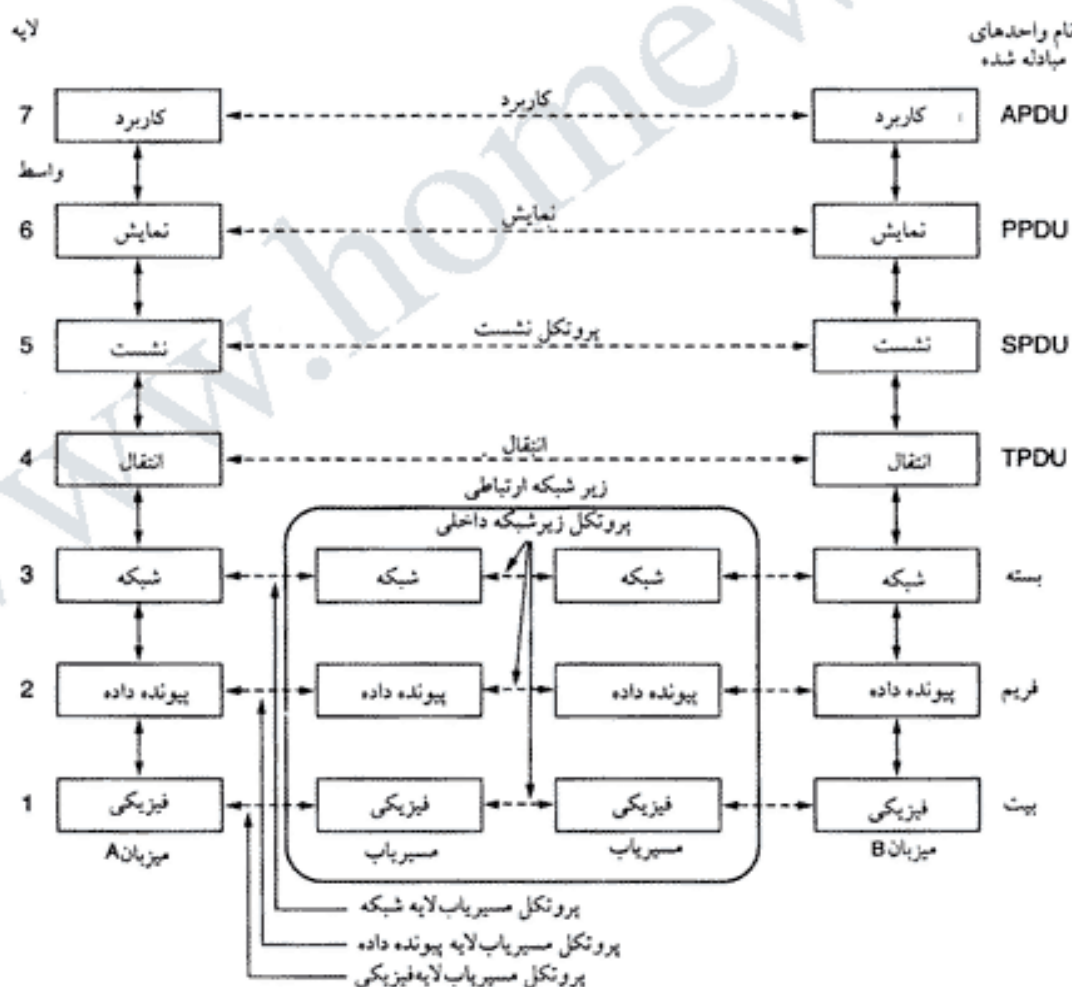
شکل ۱-۱۹. رابطه بین سرویس و پروتکل.

۴-۱ مدل های مرجع

حال که با شبکه های چند لایه بصورت تنوری آشنا شدید، وقت آنست که نگاهی به چند نمونه از این نوع شبکه ها بیندازیم. در دو قسمت آینده دو تا از مهمترین معماری های شبکه، مدل مرجع OSI و مدل مرجع TCP/IP، را بررسی خواهیم کرد. با اینکه پروتکل های مدل OSI امروزه بندرت مورد استفاده عملی دارند، اما این مدل همچنان معتبر، و مشخصات لایه های آن از اهمیت زیادی برخوردار است. وضعیت مدل TCP/IP بر عکس است: با اینکه خود این مدل کمتر مورد استفاده قرار می گیرد، ولی پروتکل های آن کاربرد وسیعی دارند. به همین دلیل، مدل های فوق را مفصلاً مورد بررسی قرار داده ایم.

۱-۴-۱ مدل مرجع OSI

در شکل ۱-۲۰-۱ مدل OSI را (منهای لایه رسانه فیزیکی) ملاحظه می کنید. این مدل بر اساس نظرات پیشنهادی سازمان بین المللی استاندارد ها (International Standards Organization - ISO) - بعنوان اولین استاندارد بین المللی شبکه های چند لایه - توسعه داده شد (Day and Zimmermann, 1983). این مدل در سال ۱۹۹۵ مورد تجدیدنظر قرار گرفت (Day, 1995). این مدل که نام کامل آن مدل مرجع ارتباطات سیستم های باز (ISO OSI - Open System Interconnection) است، با ارتباطات سیستم های باز - سیستم هایی که قادر به ارتباط با سیستم های دیگر هستند - سرو کار دارد؛ ولی ما آنرا پسادگی مدل OSI خواهیم نامید.



شکل ۱-۲۰-۱. مدل مرجع OSI.

مدل OSI هفت لایه دارد. اما، چرا هفت لایه؟ برخی از مهمترین دلایل انتخاب هفت لایه عبارتند از:

۱. هر کجا به تجرید خاصی نیاز باشد، باید یک لایه ایجاد کرد.
۲. هر لایه باید وظیفه کاملاً مشخصی را انجام دهد.
۳. وظیفه هر لایه باید با در نظر گرفتن تعریف پروتکل‌های استاندارد بین‌المللی انتخاب شود.
۴. مرزهای هر لایه باید طوری انتخاب شود، که کمترین انتقال اطلاعات از آنها لازم باشد.
۵. تعداد لایه‌ها باید آنقدر زیاد باشد، که نیازی به تعریف توابع مشابه در یک لایه نباشد؛ و باید آنقدر کم باشد که معماری شبکه بیش از حد بزرگ و پیچیده نشود.

در زیر لایه‌های مدل OSI را (از پائین به بالا) مورد بحث قرار داده‌ایم. توجه داشته باشید که مدل OSI خود یک معماری شبکه نیست، چون هیچ سرویس یا پروتکلی در آن تعریف نمی‌شود. این مدل فقط می‌گوید که هر لایه چه کاری باید انجام دهد. با اینکه سازمان استانداردهای بین‌المللی (ISO) پروتکل‌های هر لایه را نیز تعریف کرده است، ولی آنها جزء مدل OSI نیستند، و جداگانه بصورت استانداردهای بین‌المللی منتشر می‌شوند.

لایه فیزیکی

لایه فیزیکی (physical layer) وظیفه انتقال بیت‌های خام را از طریق کانال مخابراتی بر عهده دارد. مهمترین نکته در طراحی این لایه اینست که وقتی یک طرف یک بیت 1 می‌فرستد، طرف مقابل یک بیت 1 دریافت کند، نه یک بیت 0. سؤالات اساسی در این لایه عبارتند از اینکه، برای 1 و 0 از چه ولتاژهایی استفاده کنیم، هر بیت باید چند نانوثانیه (یک میلیاردیم ثانیه) روی خط دوام بیاورد، آیا انتقال همزمان در هر دو جهت امکانپذیر باشد یا خیر، اتصال اولیه چگونه شروع شود و چگونه پایان یابد، رابط شبکه (network connector) چند پایه باید داشته باشد و وظیفه هر پایه چیست. مسائل طراحی در این لایه عمدتاً از نوع مکانیکی، الکتریکی، تایمینگ (همزمانی)، و رسانه فیزیکی انتقال (که زیر لایه فیزیکی قرار دارد) هستند.

لایه پیوند داده

مهمترین وظیفه لایه پیوند داده (data link layer) عبارتست از تبدیل خط فیزیکی پُر از خطا به یک خط ارتباطی عاری از خطا برای لایه بالاتر، یعنی لایه شبکه. لایه پیوند داده این کار را با شکستن داده‌های ورودی به بسته‌های کوچک چند صد یا هزار بیتی (که فریم داده - data frame - نامیده می‌شوند)، و ارسال آنها انجام می‌دهد. وقتی گیرنده هر بسته را دریافت می‌کند، یک فریم تصدیق دریافت (acknowledgement frame) به فرستنده باز پس می‌فرستد، تا آنرا از دریافت صحیح بسته مطلع کند.

مسئله دیگری که در لایه پیوند داده (و حتی لایه‌های بالاتر) باید حل شود اینست که، چگونه یک گیرنده کُند را با یک فرستنده سریع هماهنگ کند. برای این منظور باید مکانیزمی تعبیه شود تا فرستنده در هر لحظه از مقدار پافر (buffer - حافظه موقت) گیرنده مطلع باشد. در اغلب موارد این دو ویژگی - کنترل جریان اطلاعات و مقابله با خطا - در هم ادغام می‌شوند.

لایه پیوند داده در شبکه‌های پخش (broadcast) باید با مسئله دیگری نیز دست و پنجه نرم کند: کنترل دسترسی به یک کانال مشترک. برای این منظور از زیرلایه‌ای بنام کنترل دسترسی رسانه (medium access control) در لایه پیوند داده استفاده می‌شود.

لایه شبکه

لایه شبکه (network layer) عملکرد زیر شبکه را کنترل می‌کند. یکی از مسائلی که باید در این لایه حل شود، نحوه مسیریابی بسته‌ها از مبدأ به مقصد است. این مسیرها می‌توانند مسیرهای استاتیک باشند (مسیرهایی که بطور

ثابت و بندرت متغیر در شبکه تعبیه شده اند)، یا مسیرهای نیمه استاتیک (مسیرهایی که در ابتدای هر نشست تعیین و مشخص می شوند)، و یا مسیرهای دینامیک (مسیرهایی که در هر لحظه و برای هر بسته از نو - و با توجه به بار شبکه - جستجو و مشخص می شوند).

اگر تعداد بسته های در حال حرکت در یک زیرشبکه بیش از حد باشد، آنها راه یکدیگر را بند آورده و وضعیتی را بوجود می آورند که به آن گلوگاه (bottleneck) یا ازدحام (congestion) گفته می شود. کنترل این وضعیت نیز بر عهده لایه شبکه است. به بیان کلی، کیفیت سرویس (تاخیر انتشار بسته ها، زمان انتقال آنها، و حالت های گذرا در شبکه) همگی جزء مسئولیتهای لایه شبکه است.

اگر یک بسته برای رسیدن به مقصد خود باید از یک شبکه خارج و وارد شبکه دیگری شود، مسائل جدیدی بروز خواهد کرد. اول اینکه، امکان دارد آدرس دهی در این شبکه ها متفاوت باشد. دوم اینکه، ممکن است شبکه دوم این بسته را بکلی نپذیرد، چون مثلاً اندازه آن بیش از حد بزرگ است؛ و یا اینکه پروتکلها با هم فرق داشته باشند، و مسائلی از این قبیل. حل همه این مسائل (و بهم پیوستن شبکه های ناهمگن) نیز از وظایف لایه شبکه است. در شبکه های پختی مسیریابی بسته ها ساده است، بهمین دلیل لایه شبکه یا بسیار کوچک است یا اصلاً نیست.

لایه انتقال

اصلی ترین وظیفه لایه انتقال (transport layer) گرفتن داده ها از لایه بالاتر، تقسیم آن به قطعات کوچکتر (در صورت نیاز)، ارسال آن به لایه شبکه، و حصول اطمینان از دریافت صحیح آنها در طرف مقابل است. علاوه بر آن، همه این کارها باید بگونه ای مؤثر و طوری انجام شود که لایه های بالاتر را از تغییرات اجتناب ناپذیر در سخت افزار ایزوله کند.

این لایه همچنین تعیین می کند که چه سرویس هایی باید در اختیار لایه نشست (و از آنجا، در اختیار کاربران شبکه) قرار گیرد. متداولترین نوع انتقال کانالهای نقطه-به-نقطه عاری از خطا است، که در آن باینها بهمان ترتیبی که فرستاده شده اند، در طرف مقابل دریافت می شوند. با این حال انواع دیگری از سرویس های انتقال وجود دارد، که از میان آنها می توان به انتقال پیام بدون اطمینان از دریافت منظم آنها، و ارسال همزمان پیام های پختی به چندین نقطه اشاره کرد. نوع این سرویس ها در لحظه برقراری اتصال مشخص می شود. (البته باید تصریح کنیم که کانال عاری از خطا در دنیای واقعی وجود خارجی ندارد، و آنچه از این اصطلاح برداشت می شود پائین بودن نرخ خطا است، بگونه ای که بتوان در عمل آنرا نادیده گرفت).

لایه انتقال یک لایه نقطه-به-نقطه واقعی است، که در آن کامپیوتر فرستنده (مبدأ) مستقیماً با کامپیوتر گیرنده (مقصد) ارتباط دارد. در لایه های پائینتر ارتباط ماشین مبدأ معمولاً با ماشینهای همسایه (و نه ماشین مقصد) است. تفاوت لایه های ۱ تا ۳ (که بصورت زنجیره ای هستند) با لایه های ۴ تا ۷ (که نقطه-به-نقطه هستند) در شکل ۱-۲۰ نشان داده شده است.

لایه نشست

لایه نشست (session layer) اجازه می دهد تا بین کاربران در ماشینهای مختلف نشست برقرار شود. نشست سرویس های مختلفی ارائه می کند، از جمله: کنترل دیالوگ (dialog control - کنترل اینکه نوبت چه کسی است)، مدیریت نشانه (token management - جلوگیری از تداخل اعمال مهم)، و سنکرون کردن (synchronization - کنترل عملیات انتقال طویل المدت، و از سرگیری آن از نقطه قطع شده در صورت بروز اختلال).

لایه نمایش

بر خلاف لایه های پائینتر، که عمدتاً با بیت ها سروکار دارند، لایه نمایش (presentation layer) توجه خود را

روی ساختار پیامها و مفهوم آنها متمرکز می‌کند. برای اینکه کامپیوترهایی با ساختارهای متفاوت داده بتوانند با هم ارتباط برقرار کنند، ساختار پیامهای مبادله شده بایستی کاملاً مشخص و استاندارد باشد. وظیفه لایه نمایش مدیریت این ساختارها در سطح بالاست.

لایه کاربرد

بسیاری از پروتکل‌های مورد نیاز کاربران در لایه کاربرد (application layer) قرار دارد، که از معروفترین آنها می‌توان به پروتکل HTTP (HyperText Transfer Protocol) - پروتکل اصلی وب - اشاره کرد. وقتی مرورگر وب می‌خواهد صفحه‌ای را بار کند، نام آن صفحه را با استفاده از پروتکل HTTP به سرور می‌دهد و سرور می‌فرستد؛ سرور می‌دهد و وب نیز با همین پروتکل صفحه را به مرورگر می‌گرداند. پروتکل انتقال فایل (FTP)، پروتکل انتقال خبر (NNTP)، و پروتکل‌های پست الکترونیک (SMTP و POP) نیز جزء پروتکل‌های کاربردی هستند.

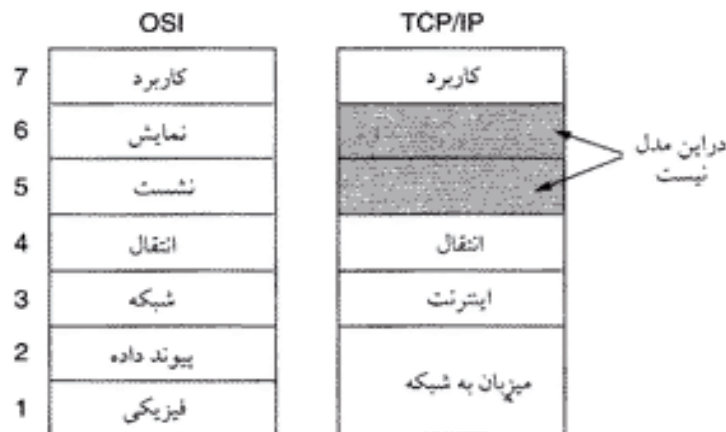
۲-۴-۱ مدل مرجع TCP/IP

اکنون اجازه دهید به مدل مرجع بکار رفته در پدربزرگ شبکه‌های کامپیوتری، آرپانت (ARPANET)، و خلف آن اینترنت، بپردازیم. آرپانت یک شبکه تحقیقاتی بود که توسط DoD (وزارت دفاع ایالت متحده آمریکا) پایه‌ریزی شد. بتدریج صدها دانشگاه و مرکز دولتی بوسیله خطوط اجاره‌ای تلفن (leased line) به این شبکه ملحق شدند. با پیشرفت مخابرات رادیویی و ماهواره، مشکلاتی در پروتکل‌های ارتباطی آرپانت بوجود آمد، که انتخاب یک معماری مرجع جدید را الزامی می‌کرد. یکی از اولین هدف‌های آرپانت ارتباط یکپارچه شبکه‌های مختلف بود، که بالاخره (بعد از بررسی چندین پروتکل) توسط مدل مرجع TCP/IP محقق شد. این مدل برای اولین بار توسط (Cerf and Kahn, 1974) تعریف شد، که در سال ۱۹۸۵ مورد تجدیدنظر قرار گرفت (Leiner et al., 1985). فلسفه طراحی مدل مرجع TCP/IP در (Clark, 1988) مورد بحث قرار گرفته است.

دغدغه همیشگی وزارت دفاع این بوده که بخشی از شبکه و وسایل با ارزش آن در یک لحظه (احتمالاً در یک حمله اتمی) نیست و نابود شود، و بهمین دلیل همواره تأکید داشته که این شبکه باید بگونه‌ای طراحی شود که حتی در صورت از بین رفتن بخشی از زیر شبکه‌های آن، بتواند بدون وقفه به کار خود ادامه دهد. بعبارت دیگر، هدف این است که دو کامپیوتر مادامیکه که کار می‌کنند، باید بتوانند با هم ارتباط داشته باشند (حتی اگر تعدادی از ماشینهای واسط بین آنها از مدار خارج شوند). علاوه بر آن، این مدل باید بتواند از عهده طیف وسیعی از کاربردهای متنوع (از انتقال فایل گرفته، تا مکالمه زمان واقعی) برآید.

لایه اینترنت

تمام این الزامات باعث شد تا در نهایت یک شبکه سوئیچینگ بسته (packet-switching) مبتنی بر یک لایه ارتباطات غیر متصل (connectionless) انتخاب شود. این لایه، که لایه اینترنت (internet layer) نام دارد، سنگ بنای معماری TCP/IP است. وظیفه این لایه اینست که به ماشینها اجازه دهد بسته‌های خود را روی شبکه و به سمت مقصد بفرستند. این لایه رسیدن پیامها را با همان ترتیبی که فرستاده شده‌اند، تضمین نمی‌کند؛ وظیفه مرتب کردن پیامها (در صورت نیاز) بر عهده لایه‌های بالاتر است. (دقت کنید که در اینجا «اینترنت» یک کلمه عام است.) لایه اینترنت تا حد زیادی شبیه سیستم پست است. اگر چند نامه را به مقصد کشوری دیگر در صندوق پست بیندازید، با کمی شانس همه آنها به دست گیرنده خواهند رسید. البته احتمال دارد که هر یک از این نامه از مسیر متفاوتی به مقصد رسیده باشد، که این موضوع از دید کاربران پنهان است (و علاقه‌ای هم به دانستن آن ندارند). فرمت بسته‌های پیام و پروتکل آنها در لایه اینترنت تعریف می‌شود، که IP (Internet Protocol) نام دارد.



شکل ۱-۲۱. مدل مرجع TCP/IP.

وظیفه لایه اینترنت اینست که بسته های IP را به مقصد برساند. مسیریابی بسته ها (و جلوگیری از ازدحام در مسیرهای شلوغ) بر عهده این لایه است، که از این نظر می توان آنرا معادل لایه شبکه در مدل OSI دانست (شکل ۱-۲۱ را ببینید).

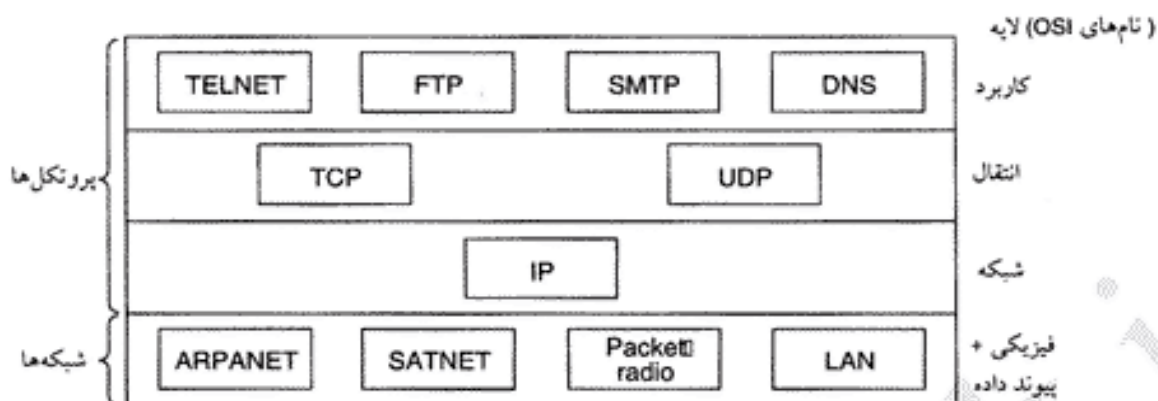
لایه انتقال

لایه بالای لایه اینترنت در مدل TCP/IP، لایه انتقال (transport layer) نام دارد. این لایه شبیه لایه انتقال در مدل OSI است، و اجازه می دهد تا عناصر همتا در کامپیوترهای مبدأ و مقصد با هم مکالمه انجام دهند. در این لایه دو پروتکل انتقال نقطه به نقطه تعریف شده است. پروتکل اول، که TCP (Transmission Control Protocol) نام دارد، یک پروتکل اتصال گرای قابل اعتماد است که اجازه می دهد تا جریانی از بایتهای بدون خطا از یک کامپیوتر در اینترنت به کامپیوتر دیگر فرستاده شود. این پروتکل جریان بایتهای را بصورت بسته بسته در آورده، و به لایه اینترنت تحویل می دهد. در ماشین مقصد عکس این عمل انجام می شود: بسته ها به هم چسبانده شده، و بصورت جریانی از بایتهای به لایه بالاتر فرستاده می شود. در پروتکل TCP کنترل جریان داده ها (flow control) نیز وجود دارد، بدین معنا که فرستنده داده ها را سریعتر از آنچه گیرنده توان دریافت آنرا دارد، ارسال نخواهد کرد.

پروتکل دوم این لایه، که UDP (User Datagram Protocol) نام دارد، یک پروتکل غیر متصل غیر قابل اعتماد است، که در مواردی که نیازی به سخت گیریهای TCP نیست از آن استفاده می شود. این پروتکل بیشتر در مواردی که سرعت اهمیت بیشتری دارد تا دقت (مانند انتقال صوت و تصویر)، یا در جاهایی که فرآیند درخواست-پاسخ فقط یک بار انجام می شود، بکار می رود. در شکل ۱-۲۲ رابطه پروتکل های IP، TCP و UDP را مشاهده می کنید. پروتکل IP اکنون در شبکه های بسیاری پیاده سازی شده است.

لایه کاربرد

مدل TCP/IP لایه های نشست یا نمایش ندارد، یعنی در واقع معتقد است که نیازی به آنها نیست. تجربه مدل OSI نیز نشان می دهد که این نظر درست است، و این دو لایه بندرت کاربردی پیدا کرده اند. در بالای لایه انتقال لایه کاربرد (application layer) قرار می گیرد، که تمام پروتکل های سطح بالا در آن قرار دارند. پروتکل های ترمینال مجازی (TELNET)، انتقال فایل (FTP) و پست الکترونیک (SMTP) از پروتکل هایی هستند که از سالها قبل در این لایه پیاده سازی شده اند (شکل ۱-۲۲). پروتکل ترمینال مجازی اجازه



شکل ۱-۲۲. پروتکل‌ها و شبکه‌ها در مدل TCP/IP.

می‌دهد تا کاربر وارد کامپیوترهای راه دور شده، و با آنها مانند یک کامپیوتر محلی کار کند. پروتکل انتقال فایل نیز ابزاریست مؤثر برای انتقال اطلاعات از یک ماشین به ماشین دیگر. پست الکترونیک در ابتدا چیزی بیش از یک انتقال فایل ساده نبود، ولی بعدها یک پروتکل خاص بنام SMTP برای آن توسعه داده شد. اکنون پروتکل‌های معروف دیگری نیز در این لایه وجود دارند، که برخی از آنها عبارتند از: پروتکل نام ناحیه (DNS) برای ترجمه نام کامپیوترها به آدرس شبکه، پروتکل انتقال خبر (NNTP) برای خواندن مقالات یوزنت (USENET)، پروتکل انتقال صفحات ابرمتن (HTTP) برای خواندن صفحات وب، و دهها پروتکل دیگر.

لایه میزبان - به - شبکه

در زیر لایه اینترنت یک شکاف بزرگ دیده می‌شود؛ در واقع مدل TCP/IP درباره این قسمت تا حد زیادی سکوت کرده است، و فقط انتظار دارد که میزبان بنحوی به شبکه وصل شده، و بتواند بسته‌های IP را ارسال کند. پروتکل انجام این کار در مدل TCP/IP تعریف نمی‌شود، و در موارد مختلف متفاوت است (حتی کتابها و مقالاتی که درباره TCP/IP نوشته شده‌اند، بندرت در این باره صحبت می‌کنند).

۳-۴-۱ مقایسه مدل‌های OSI و TCP/IP

مدل‌های OSI و TCP/IP نقاط مشترک زیادی دارند. هر دوی آنها مبتنی بر مجموعه‌ای از پروتکل‌های مستقل هستند، و عملکرد لایه‌ها نیز تا حد زیادی شبیه یکدیگر است. برای مثال، در هر دو مدل لایه‌های بالای لایه انتقال (و از جمله خود آن) بصورت نقطه-به-نقطه عمل می‌کنند، مستقل از شبکه هستند، و سرویسهای خود را (به شکلی کاربرد-گرا) در اختیار لایه‌های بالاتر می‌گذارند.

علیرغم شباهت‌های اساسی، این دو مدل تفاوت‌های بسیاری نیز با هم دارند، که در این قسمت به آنها خواهیم پرداخت. شایان ذکر است که ما در اینجا مدل‌ها را با هم مقایسه می‌کنیم، نه مجموعه پروتکل‌های آنها را (در این باره نیز بعداً صحبت خواهیم کرد). برای یک مقایسه کامل و جامع بین TCP/IP و OSI به کتاب (Piscitello and Chapin, 1993) مراجعه کنید.

در مدل OSI سه مفهوم محوری وجود دارد:

۱. سرویس (service)
۲. واسطه (interface)
۳. پروتکل (protocol)

شاید بزرگترین دستاورد مدل OSI روشن ساختن مفاهیم فوق (و تفکیک آنها) باشد. هر لایه سرویس‌هایی در اختیار لایه‌های بالاتر از خود قرار می‌دهد. تعریف این سرویس‌ها فقط می‌گوید که یک لایه چه کاری انجام می‌دهد، و هیچ حرفی درباره نحوه انجام آنها و چگونگی استفاده از سرویس‌ها نمی‌زند.

تعریف چگونگی دسترسی به سرویس‌های یک لایه بر عهده واسطه است. واسطه پارامترهای ورودی لازم، و نتیجه‌ای را که باید منتظر آن باشید، تعریف می‌کند. حتی واسطه هم نمی‌گوید که یک لایه چگونه کار خود را انجام می‌دهد.

و بالاخره، کاری که لایه انجام می‌دهد را پروتکل‌های آن لایه تعریف می‌کنند. یک لایه مادامیکه کارش را بدرستی انجام دهد، می‌تواند از هر پروتکلی استفاده کند. تغییر پروتکل‌های یک لایه هیچ تأثیری روی ارتباط آن با لایه‌های بالاتر نخواهد گذاشت.

ایده‌های فوق بسیار شبیه مفاهیم مدرن برنامه‌نویسی شیء‌گرا هستند. هر شیء، مانند یک لایه، متدها (عملکردها)ی دارد که اشیاء دیگر از آنها استفاده می‌کنند. نحوه استفاده از این متدها در واقع همان سرویس‌هاییست که این شیء در اختیار دیگران می‌گذارد. ورودیها و خروجیهای شیء واسطه آن با دنیای خارج هستند. گد اجرایی شیء نیز شبیه همان پروتکل است، که نحوه عملکرد آن از دید دیگران مخفی است.

در مدل اولیه TCP/IP تمایز بین سرویس‌ها، واسطه‌ها و پروتکل‌ها واضح و مشخص نبود، اگرچه افرادی (با توجه به تجربه موفق OSI) سعی کرده بودند آنها را هر چه بیشتر شبیه OSI کنند. برای مثال، لایه اینترنت فقط دو سرویس واقعی بنامهای SEND IP PACKET و RECEIVE IP PACKET داشت. با توجه به این وضع، پروتکل‌های OSI بهتر از TCP/IP مخفی شده‌اند، و امکان تغییر آنها براحتی وجود دارد، چیزی که هدف غایی طراحی لایه‌ای محسوب می‌شود.

مدل OSI قبل از اختراع پروتکل‌های آن طراحی و ابداع شد. این بدان معناست که مدل OSI وابستگی و تمایل خاصی به هیچ مجموعه پروتکلی ندارد، چیزی که در سایر مدل‌ها بسیار دیده می‌شود. البته این وضعیت یک نقطه ضعف نیز دارد، و آن اینست که طراحان تجربه چندانی در زمینه موضوع کار ندارند، و واقعاً نمی‌دانند کدام عملکرد را باید در کدام لایه قرار دهند. برای مثال، لایه پیوند داده در ابتدا فقط برای شبکه‌های نقطه-به-نقطه طراحی شده بود، و وقتی شبکه‌های پخش‌ی وارد بازار شد، مجبور شدند یک زیرلایه به آن اضافه کنند.

وقتی افراد شروع به طراحی شبکه با استفاده از مدل OSI و پروتکل‌های موجود کردند، بزودی دریافتند که این شبکه‌ها با سرویس‌های مورد نیاز انطباق ندارند (۱)، بنابراین مجبور شدند زیرلایه‌های زیادی به آن وصله‌پینه کنند. بالاخره، کمیته استاندارد مقرر کرد که هر کشور برای خود یک شبکه منطبق با مدل OSI (تحت نظارت دولت) داشته باشد - شبکه‌ای که به هیچ عنوان آینده (اینترنت) در آن دیده نشده بود. خلاصه، کارها آنطوری که انتظار داشتند از آب در نیامد.

در مورد TCP/IP وضع بر عکس بود: اول پروتکل‌ها اختراع و توسعه داده شدند، و سپس مدلی برای توصیف آنها ساخته شد. هیچ مشکلی در زمینه انطباق پروتکل‌ها با مدل وجود نداشت؛ همه چیز جفت و جور بود. تنها مشکل این بود که این مدل با هیچ مجموعه پروتکل دیگری جور در نمی‌آمد. این بدان معنا بود که مدل TCP/IP بدرد توصیف شبکه‌های غیر TCP/IP نمی‌خورد.

جدای از مسائل فلسفی قضیه، تفاوت دیگر در تعداد لایه‌های این دو مدل است: مدل OSI هفت لایه دارد و مدل TCP/IP چهار لایه. لایه‌های شبکه، انتقال و کاربرد در هر دو مشترکند، ولی لایه‌های دیگر فرق دارند.

تفاوت دیگر در زمینه ارتباطات اتصال-گرا و غیر متصل است. مدل OSI از هر دو نوع ارتباط اتصال-گرا و غیر متصل در لایه شبکه پشتیبانی می‌کند، ولی در لایه انتقال فقط سرویس اتصال-گرا دارد (چون این سرویس در

معرض دید کاربران است). مدل TCP/IP در لایه شبکه فقط سرویس غیر متصل دارد، ولی در لایه انتقال از هر دو نوع ارتباط پشتیبانی می کند، و دست کاربر را برای انتخاب باز می گذارد (که بویژه برای پروتکل های ساده درخواست پاسخ بسیار مهم است).

۱-۲ نگاهی انتقادی به مدل OSI و پروتکل های آن

مدل های OSI و TCP/IP (و پروتکل هایشان) هیچکدام کامل نیستند، و جا دارد برخی از نقاط ضعف آنها را برشماریم. در این قسمت و قسمت آینده، برخی از نقاط ضعف مهم مدل های OSI و TCP/IP را بررسی خواهیم کرد. با مدل OSI شروع می کنیم.

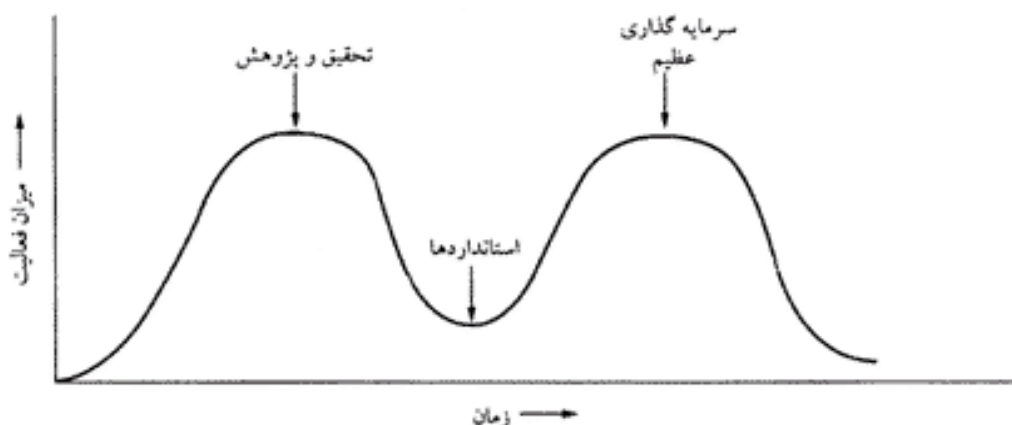
در زمان چاپ ویرایش دوم این کتاب (سال ۱۹۸۹)، بسیاری از متخصصان برجسته شبکه بر این باور بودند که آینده در بست متعلق به مدل OSI و پروتکل های آن است، و هیچ چیز نمی تواند در مقابل پیشرفت آن مقاومت کند. اما این اتفاق نیفتاد. چرا؟ نگاهی به گذشته درسهای بسیاری برای چشمان عبرت بین دارد، که می توان آنها را چنین خلاصه کرد:

۱. زمان نامناسب
۲. تکنولوژی نامناسب
۳. پیاده سازی نامناسب
۴. سیاست های نامناسب

زمان نامناسب

اولین عامل شکست مدل OSI زمان نامناسب بود. زمانی که یک استاندارد وضع می شود، اهمیت حیاتی در موفقیت یا عدم موفقیت آن دارد. دیوید کلارک از دانشگاه M.I.T فرضیه ای در زمینه استانداردها دارد که به ملاقات فیل ها معروف است، و در شکل ۱-۲۳ آنرا مشاهده می کنید.

این شکل میزان فعالیتهای حول یک موضوع جدید را نشان می دهد. وقتی موضوعی برای اولین بار کشف می شود، گردها گرد آنرا سبلی از فعالیتهای تحقیقی (به شکل بحث، مقاله و سخنرانی) فرا می گیرد. بعد از مدتی این موج فروکش می کند، و بعد از اینکه صنعت به آن موضوع علاقمند شد، موج سرمایه گذاری ها از پی می آید. بسیار مهم است که در نقطه تلاقی این دو فیل (موج تحقیق و موج سرمایه گذاری) استانداردها بطور کامل وضع شوند. اگر استاندارد زودتر از موعد (قبل از پایان تحقیقات) نوشته شود، خطر آن هست که موضوع بدرستی



شکل ۱-۲۳. فرضیه ملاقات فیل ها

درک نشده باشد، و استاندارد ضعیف از آب در آید. اگر استاندارد دیرتر از موعد (بعد از شروع موج سرمایه‌گذاری) نوشته شود، شرکت‌های بسیاری قبلاً - در مسیرهای مختلف - در آن سرمایه‌گذاری کرده‌اند، و این خطر هست که استاندارد آنها را نادیده بگیرد. اگر فاصله این دو فیل خیلی کم باشد (همه عجله داشته باشند که زودتر کار را شروع کنند)، خطر آن هست که استاندارد نویسان بین آنها له شوند.

اکنون معلوم شده است که پروتکل‌های استاندارد OSI بین فیل‌ها له شدند. وقتی پروتکل‌های OSI پایه عرصه وجود گذاشتند، پروتکل‌های رقیب (TCP/IP) مدتها بود که در دانشگاه‌ها و مراکز تحقیقاتی پذیرفته شده بودند. با اینکه هنوز موج سرمایه‌گذاری صنعتی در TCP/IP شروع نشده بود، اما بازار آکادمیک آنقدر بزرگ بود که شرکت‌های بسیاری را تشویق به تولید محصولات TCP/IP کند. و وقتی OSI بالاخره از راه رسید، کسی نبود که داوطلبانه از آن پشتیبانی کند. همه منتظر بودند دیگری قدم اول را بردارد؛ قدمی که هرگز برداشته نشد، و OSI در نطفه خفه شد.

تکنولوژی نامناسب

دلیل دیگری که OSI هرگز پا نگرفت آن بود که، این مدل و پروتکل‌های آن هر دو ناقص و معیوب بودند. انتخاب هفت لایه برای این مدل بیشتر یک انتخاب سیاسی بود تا فنی، و در حالیکه دو لایه آن (نشست و نمایش) تقریباً خالی بودند، در لایه‌های دیگر (لینک داده و شبکه) جای نفس کشیدن نبود.

مدل OSI (و سرویس‌ها و پروتکل‌های آن) بطرزی باورنکردنی پیچیده است. اگر کاغذهای چاپی این استاندارد را روی هم بچینید، ارتفاع آن از نیم متر هم بیشتر خواهد شد! پیاده‌سازی پروتکل‌های OSI بسیار دشوار، و عملکرد آنها ناقص است. در این رابطه، نقل جمله جالبی از پاول موکاپتریس (Rose, 1993) خالی از لطف نیست:

سؤال: از ترکیب یک گانگستر با یک استاندارد بین‌المللی چه چیزی بدست می‌آید؟

جواب: کسی پیشنهادی به شما می‌کند که از آن سر در نمی‌آورد.

مشکل دیگر مدل OSI، علاوه بر غیر قابل فهم بودن آن، اینست که برخی از عملکردهای آن (مانند آدرس‌دهی، کنترل جریان داده‌ها، و کنترل خطا) در تمام لایه‌ها تکرار می‌شود. برای مثال، سالتزر و همکارانش (1984) نشان دادند که کنترل خطا باید در بالاترین لایه انجام شود تا بیشترین تأثیر را داشته باشد، بنابراین تکرار آن در لایه‌های پائین‌تر نه تنها غیر ضروری است، بلکه باعث افت کارایی هم خواهد شد.

پیاده‌سازی نامناسب

با توجه به پیچیدگی بیش از حد مدل OSI و پروتکل‌های آن، جای تعجب نبود که اولین پیاده‌سازیهای آن حجیم، سنگین و کند باشد. آنهایی که با این مدل کار کرده بودند، بزودی پشیمان شدند، و طولی نکشید که کلمه OSI مترادف شد با «کیفیت بد». بعدها محصولات بهتری به بازار آمد، اما آوازه منفی OSI فراموش نشد.

از طرف دیگر، اولین پیاده‌سازی TCP/IP (که بخشی از سیستم عامل یونیکس برکلی بود) بسیار خوب از کار در آمد (و لازم به گفتن نیست که مجانی هم بود). افراد بسیاری سرعت شروع به استفاده از آن کردند، هواخواه آن شدند، آنرا توسعه دادند، و این باعث شد که باز هم به خیل طرفداران آن اضافه شود. در اینجا، برخلاف OSI، مارپیچ رو به بالا می‌رفت، نه پائین.

سیاست‌های نامناسب

بدلیل اولین پیاده‌سازی TCP/IP، بسیاری از افراد (بویژه در محیط‌های دانشگاهی) تصور می‌کردند که TCP/IP جزئی از یونیکس است، و یونیکس هم در آن دوران محبوبیتی فوق‌العاده داشت.

از سوی دیگر، این عقیده رواج داشت که OSI یک مخلوق دولتی (مخصوصاً دولتهای اروپایی و آمریکایی) است البته این عقیده فقط تا حدی درست بود، اما همین تصور هم که عده‌ای دیوانسالار دولتی بخواهد یک

استاندارد فنی را بزور جا بیندازند، باعث شد تا برنامه‌نویسان و طراحان شبکه تمایلی به همکاری از خود نشان ندهند. زبانهای برنامه‌نویسی PL/1 (که در دهه ۱۹۶۰ از سوی IBM بعنوان زبان آینده توسعه داده شد) و Ada (که وزارت دفاع آمریکا حامی آن بود) بهمین دلیل دچار سرنوشتی مشابه شدند.

۵-۴-۱ نگاهی انتقادی به مدل TCP/IP

مدل TCP/IP و پروتکل‌های آن نیز مشکلات خاص خود را دارند. اول اینکه، در این مدل مفاهیم سرویس، واسط و پروتکل بروشنی از یکدیگر تفکیک نشده‌اند؛ کاری که در مدل OSI بخوبی انجام شده است. به همین دلیل نمی‌توان از TCP/IP بعنوان ابزاری برای طراحی و توسعه شبکه‌های جدید استفاده کرد.

دوم اینکه، مدل TCP/IP به هیچ عنوان یک مدل کلی نیست، و نمی‌توان از آن برای توصیف شبکه‌های غیر TCP/IP استفاده کرد. برای مثال، توصیف بلوتوث با مدل TCP/IP بکلی غیرممکن است. سوم اینکه، با در نظر گرفتن مفاهیم شبکه‌های چند لایه، لایه میزبان-به-شبکه اساساً یک لایه واقعی نیست، بلکه فقط یک واسط (بین لایه‌های شبکه و لینک داده) است. در واقع، این یکی از مهمترین جاهانیست که مدل TCP/IP مفاهیم واسط و لایه را با هم قاطی کرده است.

چهارم اینکه، در مدل TCP/IP هیچ تمایزی بین لایه‌های فیزیکی و لینک داده نیست (و حتی حرفی از آنها بمیان نیامده است). اینها دو لایه کاملاً متفاوت هستند - لایه فیزیکی با مشخصات کابل و فیبر نوری و کانالهای مخابراتی سروکار دارد، در حالیکه وظیفه لایه پیوند داده شکستن داده‌ها به قطعات کوچکتر و اطمینان از تحویل صحیح آنها به مقصد است. در یک مدل کامل این دو لایه باید از هم جدا باشند؛ کاری که در مدل TCP/IP انجام نشده است.

و بالاخره اینکه، اگر چه پروتکل‌های TCP و IP بسیار خوب طراحی و پیاده‌سازی شده‌اند، بسیاری از دیگر پروتکل‌های این مدل چنین نیستند، و اغلب توسط دانشجویان کنجکاو (و در ساعات بیکاری) نوشته شده‌اند. این پروتکل‌ها بعلت انتشار سریع (که اغلب دلیلی جز مجانی بودن ندارد) بسرعت جامی افتند، و بهمین دلیل جایگزین کردن آنها بسیار دشوار می‌شود. برخی از این پروتکل‌ها امروز چیزی جز شرمساری نیستند. مثلاً، پروتکل TELNET اساساً برای ترمینالهای کُند و سنگین تله‌تایپ نوشته شد، و هیچ نشانی از گرافیک و ماوس در آن نیست. اما، بعد از ۲۵ سال خیلی‌ها همچنان از آن استفاده می‌کنند.

بطور خلاصه، مدل OSI (علیرغم برخی از مشکلات آن، و منهای لایه‌های نشست و نمایش) ثابت کرده که بهترین ابزار برای توصیف شبکه‌های کامپیوتریست - اما متأسفانه همچنان روی کاغذ باقی مانده است. از طرف دیگر، با اینکه چیزی بنام مدل TCP/IP در واقع وجود خارجی ندارد، اما پروتکل‌های آن در مقیاس وسیعی مورد استفاده قرار می‌گیرند. از آنجائیکه در دنیای کامپیوتر هر کسی حرف خود را می‌زند، ما هم در این کتاب از یک مدل اصلاح شده OSI استفاده خواهیم کرد، ولی توجه خود را بیشتر روی TCP/IP و پروتکل‌های وابسته به آن معطوف می‌کنیم - البته درباره پروتکل‌های جدید مانند SONET، 802 و بلوتوث هم صحبت خواهیم کرد. در حقیقت، مدلی که ما در این کتاب از آن استفاده خواهیم کرد، یک مدل ترکیبی (شکل ۱-۲۴) است.

5	لایه کاربرد
4	لایه انتقال
3	لایه شبکه
2	لایه پیوند داده
1	لایه فیزیکی

شکل ۱-۲۴. یک مدل مرجع ترکیبی، که در این کتاب از آن استفاده خواهد شد.

۵-۱ شبکه های نمونه

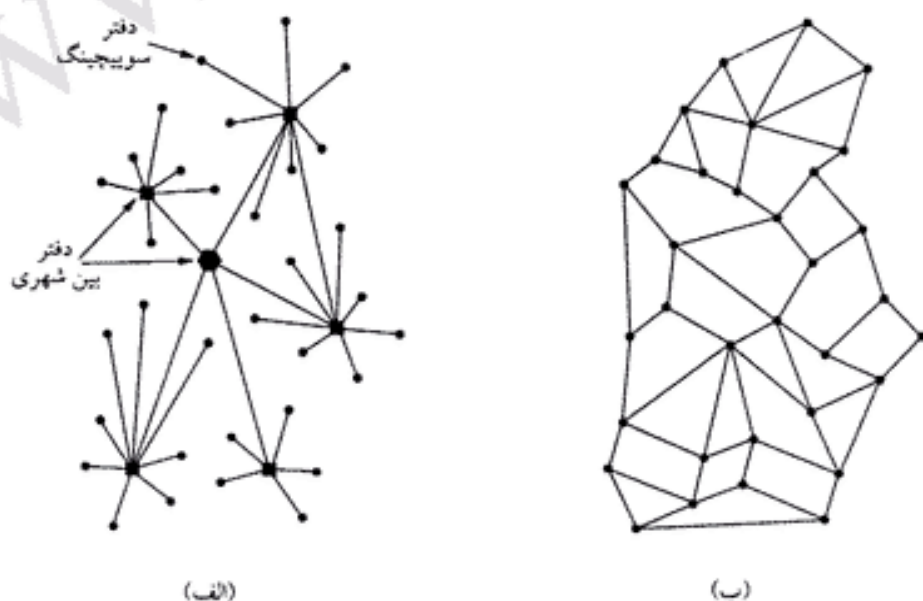
بحث شبکه های کامپیوتری انواع مختلفی از شبکه ها (کوچک و بزرگ، شناخته شده و مجهور) را در بر می گیرد. این شبکه ها در اندازه، کاربرد و تکنولوژیهای بکار رفته با هم متفاوتند. برای آن که تصویری از تنوع موجود در شبکه های کامپیوتری داشته باشید، در این قسمت نگاهی به چند نمونه از آنها خواهیم انداخت. این قسمت را با معرفی اینترنت (که احتمالاً شناخته شده ترین شبکه هاست) شروع می کنیم، و نگاهی به تاریخچه، سیر تکامل و تکنولوژی آن خواهیم داشت. پس از آن به سراغ ATM (که تفاوت های چشمگیری با اینترنت دارد، و حتی می توان گفت ضد آن است) می رویم. و بالاخره، نگاهی به IEEE 802.11 (استاندارد شبکه های محلی بیسیم) می اندازیم.

۱-۵-۱ اینترنت

اینترنت (Internet) در واقع اصلاً یک شبکه نیست، بلکه مجموعه ایست از شبکه های مختلف که از پروتکل های خاصی استفاده کرده، و سرویس های مشخصی را ارائه می کند. ویژگی غیرعادی اینترنت اینست که توسط فرد خاصی طراحی نشده، و هیچکس هم آنرا کنترل نمی کند. برای درک بهتر این مطلب، اجازه دهید ببینیم اینترنت از کجا شروع شد، و علت آن چه بود. یکی از جالبترین تاریخچه های اینترنت را می توانید در کتاب جان نافتون - 2000 - ببینید. این کتاب نه تنها برای افراد عادی، بلکه برای مورخان نیز جلب است؛ برخی از مطالب ذیل از این کتاب اقتباس شده است. البته کتابهای فنی بیشتری نیز درباره اینترنت و پروتکل های آن نوشته شده، که از آن میان می توان به (Maufer, 1999) اشاره کرد.

آرپانت (ARPANET)

داستان ما از اواخر دهه ۱۹۵۰ شروع می شود. در اوج جنگ سرد، وزارت دفاع ایالات متحده آمریکا به فکر ایجاد یک شبکه فرماندهی و کنترل افتاد که بتواند حتی در مقابل حملات هسته ای دوام بیاورد. در آن زمان تمامی مخابرات نظامی به شبکه تلفن عمومی متکی بود، که مستعد آسیب تشخیص داده شده بود. با یک نگاه به شکل ۱-۲۵ (الف) می توانید مبنای این استدلال را دریابید. در این شکل نقاط سیاه نماینده مراکز سونیچینگ شهری



شکل ۱-۲۵. (الف) ساختار شبکه تلفن. (ب) طرح بارن برای یک سیستم سونیچینگ توزیع شده.

هستند که هزاران خط تلفن از آنها منشعب می شود. این مراکز نیز بنوبه خود به مراکز بین شهری بزرگتر متصل هستند، که در مجموع شبکه تلفن کشوری را می سازند. آسیب پذیری این سیستم از آنجا ناشی می شد که تخریب چند مرکز بین شهری کلیدی می توانست تماس تلفنی را در کل کشور مختل کند.

در سال ۱۹۶۰ وزارت دفاع قراردادی را با شرکت راند (RAND Corporation) امضا کرد، که در آن وظیفه یافتن یک راه حل به آن محول شده بود. یکی از متخصصان این شرکت، بنام پل بارن (Paul Baran)، طرح یک شبکه توزیع شده (distributed) و تحمل پذیر خطا (fault-tolerant) را پیشنهاد کرد، که آنرا در شکل ۱-۲۵ (ب) می بینید. از آنجائیکه در این شبکه طول مسیر بین مراکز سوییچینگ طولانیتر از آن بود که بتوان از سیگنالهای آنالوگ استفاده کرد، بارن پیشنهاد کرد در این سیستم از تکنولوژی سوییچینگ بسته دیجیتال (digital packet-switching) استفاده شود. بارن گزارشات متعددی برای وزارت دفاع نوشت، و جزئیات سیستم پیشنهادی خود را تشریح کرد. مقامات رسمی پتاگون به ایده نهفته در این سیستم علاقمند شدند، و از AT&T (که در آن زمان انحصار شبکه تلفن کشوری را در دست داشت) خواستند که یک نمونه اولیه از آن بسازد. AT&T طرح بارن را رد کرد؛ بزرگترین و ثروتمندترین شرکت دنیا تحمل نمی کرد که یک جوان تازه از راه رسیده به آنها بگوید چگونه شبکه تلفن بسازند! آنها ادعا کردند که طرح بارن قابل اجرا نیست، و بدین ترتیب ایده آن را در نطفه خفه کردند.

سالها گذشت، و وزارت دفاع همچنان بدنبال سیستم فرماندهی و کنترل ایده آل خود بود. برای درک بهتر اتفاقات بعدی، باید کمی به عقب برگردیم: به اکتبر ۱۹۵۷، زمانی که اتحاد جماهیر شوروی (سابق) با پرتاب اولین قمر مصنوعی بنام اسپوتنیک در مسابقه فضایی از ایالات متحده پیشی گرفت. آیزنهاور، رئیس جمهور وقت ایالات متحده، در جستجو برای یافتن علت عقب افتادگی کشورش، با وحشت دریافت که نیروهای زمینی، دریایی و هوایی آمریکا مشغول دعوا بر سر تقسیم بودجه تحقیقاتی پتاگون هستند. وی بلافاصله تصمیم گرفت که یک مرکز واحد برای تحقیقات نظامی بوجود آورد؛ مرکزی که آرپا (آژانس پروژه های تحقیقاتی پیشرفته Advanced Research Projects Agency - ARPA) نام گرفت. آرپا هیچ دانشمند یا آزمایشگاهی نداشت؛ در واقع، آرپا چیزی نبود جز یک دفتر هماهنگی کوچک با بودجه ای ناچیز (البته با معیارهای پتاگون). آرپا کارش را با عقد قرارداد با واگذاری امتیاز به شرکتها یا دانشگاههایی که ایده های جالبی داشتند، انجام می داد.

در سالهای اول، آرپا بیشتر سعی داشت خطوط کلی مأموریت خود را روشن و ترسیم کند، ولی در سال ۱۹۶۷ توجه مدیرعامل آن، لاری رابرتس، به موضوع شبکه جلب شد. او با متخصصان بسیاری مشورت کرد؛ و یکی از همین متخصصان، بنام ولسی کلارک، بود که پیشنهاد ایجاد یک زیر شبکه سوییچینگ بسته را مطرح کرد (شکل ۱-۱۰).

بعد از مقداری بحثهای اولیه، رابرتس ایده را پسندید و آنرا طی یک مقاله نسبتاً مبهم به گردهمایی اصول سیستم عامل (که در اواخر ۱۹۶۷ در گاتلین بورگ، تنسی برگزار شده بود) ارائه کرد (Roberts, 1967). در میان ناباوری رابرتس، مقاله دیگری نیز به این کنفرانس ارائه شده بود که نه تنها سیستم مشابهی را توصیف می کرد، بلکه حتی صحبت از پیاده سازی آن تحت مدیریت فردی بنام دونالد دیویس در آزمایشگاه ملی فیزیک (NPL) در انگلستان به میان آمده بود. سیستم NPL در واقع سیستمی در سطح ملی نبود، بلکه فقط چند کامپیوتر را در محوطه NPL به هم متصل می کرد، اما نکته مهم این بود که نشان می داد سوییچینگ بسته در عمل کار می کند. از همه جالبتر اینکه، سیستم NPL بر اساس کارهای بارن پایه گذاری شده بود. وقتی رابرتس از گاتلین بورگ برگشت، دیگر مصمم بود چیزی را بسازد که بعدها به آرپانت (ARPANET) معروف شد.

این زیر شبکه تعدادی مینی کامپیوتر بنام IMP (Interface Message Processor) را با خطوط انتقال

56-kbps به هم متصل می کرد. برای رسیدن به قابلیت اعتماد بالا، هر IMP به حداقل دو IMP دیگر متصل می شد. این زیرشبکه در واقع یک زیرشبکه دیتاگرام (datagram subnet) بود، بنابراین اگر تعدادی از خطوط یا IMP ها از بین می رفتند، پیامها می توانستند از طریق مسیرهای جایگزین به مقصد برسند.

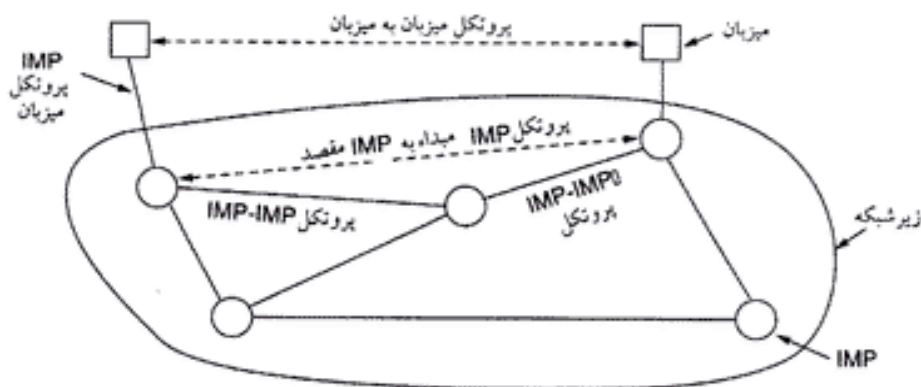
هر گره (node) این شبکه عبارت بود از یک کامپیوتر میزبان و یک IMP، که با سیمی کوتاه به هم وصل می شدند. هر میزبان می توانست پیامهایی تا سقف ۸۰۶۳ بیت به IMP خود بفرستد، و این IMP سپس پیام را به بسته های ۱۰۰۸ بیتی شکسته و آنها را بصورت مستقل به سمت مقصد می فرستاد. هر بسته قبل از اینکه به گره بعدی هدایت شود، بایستی بطور کامل دریافت می شد؛ بدین ترتیب، آرپانت اولین زیرشبکه سوئیچینگ بسته بود که بصورت ذخیره-هدایت (store-and-forward) کار می کرد.

پس از آن آرپا مناقصه ای برای ساخت این زیرشبکه اعلام کرد، که دوازده شرکت اسناد آنرا خریدند. بعد از بررسی پیشنهادات رسیده، در دسامبر ۱۹۶۸ آرپا شرکت BBN را (که یک شرکت مشاوره در کمبریج، ماساچوست بود) برای ساخت این زیرشبکه و نوشتن نرم افزارهای آن برگزید. شرکت BBN مینی کامپیوترهای اصلاح شده هانی ول DDP-316 را (که ۱۲ کیلوبایت حافظه ۱۶ بیتی داشت) بعنوان IMP انتخاب کرد. از آنجائیکه قطعات مکانیکی ذاتاً غیر قابل اعتماد فرض می شدند، این IMP ها اصلاً دیسک نداشتند، و با خطوط اجاره ای 56-kbps به هم متصل می شدند. با اینکه امروزه حتی بچه ها هم دیگر خطوط 56-kbps را قبول ندارند (و به کمتر از ADSL راضی نمی شوند)، آنروزها خطوط 56-kbps بالاترین چیزی بود که می شد آرزو کرد.

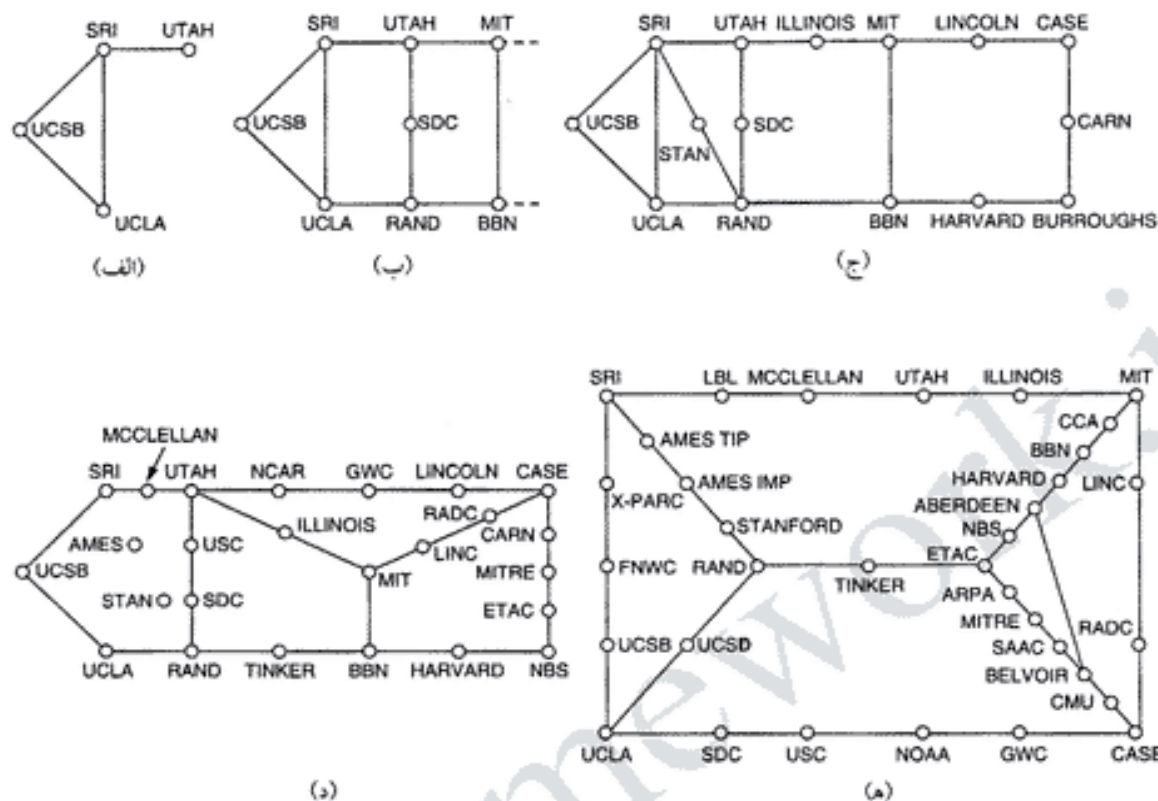
نرم افزار نیز در دو بخش مجزا طراحی شد: زیرشبکه، میزبان. نرم افزار زیرشبکه عبارت بود از پروتکل ارتباط IMP با میزبان، پروتکل IMP-IMP، و نرم افزاری برای بهبود ارتباط IMP مبدأ با IMP مقصد. در شکل ۱-۲۶ طراحی اولیه آرپانت را ملاحظه می کنید.

در خارج از زیرشبکه، میزبانها نیز به نرم افزار نیاز داشتند: پروتکل ارتباط میزبان با IMP، پروتکل میزبان-میزبان، و نرم افزارهای کاربردی. بزودی معلوم شد که BBN احساس می کند با گرفتن پیام در نقطه واسط میزبان-IMP، و تحویل آن در نقطه IMP-میزبان سمت مقابل کارش پایان یافته است.

اما رابرتس مشکل دیگری داشت: کامپیوترهای میزبان هم نیازمند نرم افزار بودند. برای حل این مشکل، در تابستان ۱۹۶۹ رابرتس همایشی از متخصصان شبکه (که عمدتاً دانشجویان تازه فارغ التحصیل بودند) در اسنوپرد، یوتا تشکیل داد. این دانشجویان فکر می کردند کسی وجود دارد که طرح کلی شبکه را برای آنها توضیح دهد، و بعد از آن می توانستند نوشتن نرم افزار را شروع کنند. آنها بسیار شگفت زده شدند وقتی فهمیدند که نه متخصصی برای توضیح طرح شبکه وجود دارد، و نه اساساً چیزی بنام طرح شبکه! آنها دریافتند که باید کار را از صفر شروع کنند.



شکل ۱-۲۶. طراحی اولیه آرپانت.



شکل ۱-۲۷. مراحل رشد آرپانت. (الف) دسامبر ۱۹۶۹. (ب) ژوئیه ۱۹۷۰. (ج) مارس ۱۹۷۱. (د) آوریل ۱۹۷۲. (ه) سپتامبر ۱۹۷۲.

با وجود همه این مشکلات، بالاخره آرپا موفق شد در دسامبر ۱۹۶۹ یک شبکه آزمایشی متشکل از چهار گره (دانشگاههای UCSB، UCLA، SRI و یوتا) راهاندازی کند. علت انتخاب این چهار دانشگاه آن بود که همگی آنها قراردادهای متعددی با آرپا داشتند، و از طرف دیگر (صرفاً برای زورآزمایی فنی) کامپیوترهای آنها بکلی با هم ناسازگار بود. با نصب IMPهای جدید این شبکه گسترش یافت، و بزودی سراسر ایالات متحده را تحت پوشش گرفت. در شکل ۱-۲۷ رشد آرپانت را در طی سه سال پس از تولد آن ملاحظه می‌کنید.

آرپا برای کمک به رشد این نوزاد تازه متولد شده (آرپانت)، در زمینه شبکه‌های ماهواره‌ای و مخابرات رادیویی نیز سرمایه‌گذاری‌هایی انجام داد. در یک آزمایش معروف، با استفاده از یک شبکه رادیویی پیامهایی از یک کامیون در حال حرکت در جاده‌های کالیفرنیا به دانشگاه SRI، و از آنجا از طریق آرپانت به ساحل شرقی ایالات متحده فرستاده شد، که سپس از آنجا از طریق شبکه ماهواره‌ای به دانشگاه کالج در لندن هدایت شد. بدین ترتیب محققانی که در کامیونی در جاده‌های کالیفرنیا نشسته بودند، توانستند با کامپیوترهایی در لندن کار کنند.

این آزمایش همچنین نشان داد که پروتکل‌های موجود آرپانت برای کار روی شبکه‌های مختلف مناسب نیستند. این نتایج منجر به تحقیقات بیشتر روی پروتکل‌ها شد، که با اختراع TCP/IP و پروتکل‌های آن به اوج رسید (Cerf and Kahn, 1974). مدل TCP/IP بویژه برای ارتباطات روی شبکه‌های مختلف و ناهمگن (که آرپانت روز بروز به سمت آن حرکت می‌کرد) طراحی شده بود.

بمنظور تشویق و ترغیب پذیرش این پروتکل‌های جدید، آرپا قراردادهایی با شرکت BBN و دانشگاه کالیفرنیا در برکلی (UCB) منعقد کرد، تا این پروتکل‌ها را با یونیکس برکلی یکپارچه کنند. محققان برکلی هم کار خود را با

نوشتن برنامه های واسط شبکه (که به سوکت - socket - معروف شدند)، و برنامه های کاربردی و مدیریتی بنحوی احسن انجام دادند.

زمانه نیز با TCP/IP یار بود؛ بسیاری از دانشگاهها تازه کامپیوترهای جدید VAX را خریده، و آنها را در شبکه های LAN به هم متصل کرده بودند، اما هیچ نرم افزاری برای شبکه کردن آنها نداشتند. وقتی یونیکس 4.2BSD (با پروتکل های TCP/IP، سوکتها و نرم افزارهای کمکی خود) بعنوان یک بسته نرم افزاری کامل به بازار آمد، بلافاصله مورد قبول جامعه دانشگاهی قرار گرفت. از همه مهمتر اینکه، با TCP/IP می شد به آرپانت وصل شد، اتفاقی که بسیاری منتظر آن بودند.

در دهه ۱۹۸۰ شبکه های بسیاری (به ویژه شبکه های محلی) به آرپانت ملحق شدند. با افزایش تعداد کامپیوترهای آرپانت مشکل جدیدی پدید آمد، و آن پیدا کردن یک کامپیوتر در میان خیل عظیم کامپیوترها بود. برای حل این مشکل سیستم نام ناحیه (Domain Name System - DNS) ابداع شد، که نام کامپیوترها را به آدرس IP آنها تبدیل می کرد. از آن به بعد، DNS تبدیل به یک پایگاه داده عمومی و توزیع شده شد، که علاوه بر آدرس IP کامپیوترها، اطلاعات دیگری را نیز در اختیار کاربران خود قرار می داد. در فصل ۷ درباره DNS مفصلاً صحبت خواهیم کرد.

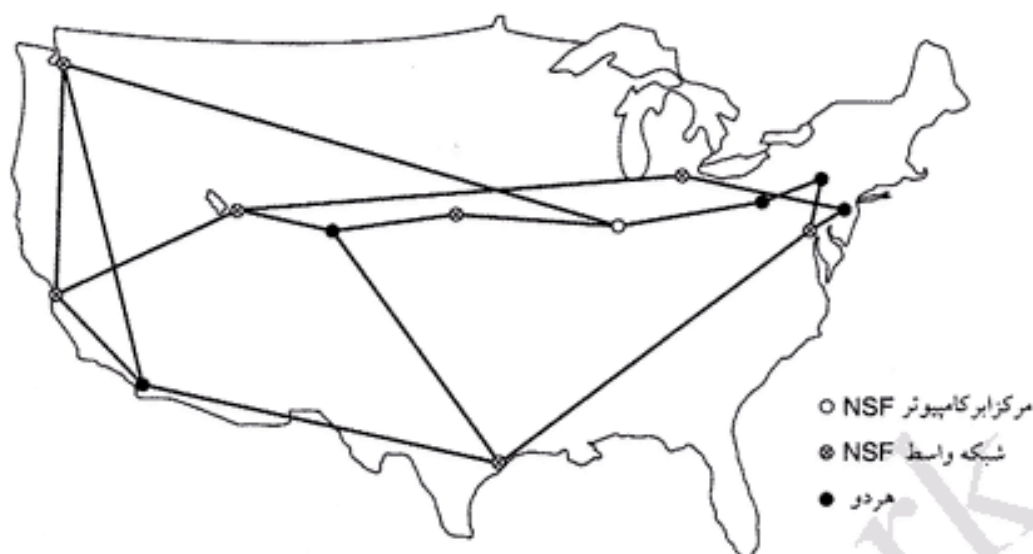
NSFNET

در اواخر دهه ۱۹۷۰ بنیاد ملی علوم ایالات متحده (U.S. National Science Foundation - NSF) شاهد تأثیر روزافزون آرپانت بر تحقیقات دانشگاهی بود. اما هر دانشگاهی که می خواست به آرپانت دسترسی داشته باشد، بایستی قراردادی با وزارت دفاع داشته باشد (که بسیاری از آنها نداشتند). پاسخ NSF به این وضعیت، راه اندازی شبکه ای مشابه آرپانت بود که تمام دانشگاهها به آن دسترسی داشته باشند. بمنظور ایجاد زیربنایی محکم برای این شبکه، NSF با متصل کردن شش ابرکامپیوتر خود در دانشگاههای سان دیه گو، بولدر، کامپاین، پیتمبورگ، ایتاکا و پرینستون، یک ستون فقرات (backbone) بوجود آورد. هر یک از این ابرکامپیوترها یک برادر کوچکتر (یک مینی کامپیوتر LSI 11، معروف به فازبال) داشت. این فازبالها به خطوط اجاره ای 56-kbps متصل بودند، و زیر شبکه را می ساختند - یعنی، شبکه NSF از نظر سخت افزاری شبیه آرپانت بود. اما، تکنولوژی نرم افزاری آن با آرپانت متفاوت بود: فازبالها از همان ابتدا به TCP/IP صحبت می کردند، که آنرا تبدیل به این اولین شبکه گسترده TCP/IP می کرد.

بعدها NSF تعداد زیادی شبکه منطقه ای تأسیس کرد، که به هزاران دانشگاه، آزمایشگاه تحقیقاتی، کتابخانه، و موزه اجازه می داد تا به هر یک از ابرکامپیوترهای آن دسترسی داشته باشند، یا اینکه مستقیماً با یکدیگر تماس برقرار کنند. این شبکه (شامل ستون فقرات و شبکه های محلی) NSFNET نامیده شد. از طریق لینکی بین یک IMP و یک فازبال در دانشگاه کارنگی-ملون، NSFNET به آرپانت نیز متصل شده بود. اولین ستون فقرات NSFNET را در شکل ۱-۲۸ مشاهده می کنید.

NSFNET یک موفقیت آنی بود، و از همان ابتدا با تراکم کاری روبرو شد. NSF بلافاصله به فکر گسترش NSFNET افتاد، و به همین منظور قراردادی با کنسرسیوم MERIT بست. برای ایجاد دومین ستون فقرات، کانالهای فیبرنوری با ظرفیت 448-kbps از MCI (که اکنون در WorldCom ادغام شده است) اجاره شد. برای مسیریاب های شبکه نیز از IBM PC-RT استفاده شد. این شبکه نیز بسیار زود با تراکم کاری روبرو شد، و در سال ۱۹۹۰ ظرفیت ستون فقرات آن به 1.5-Mbps ارتقاء داده شد.

با ادامه رشد NSFNET، بزودی NSF متوجه شد که دولت نمی تواند برای همیشه به سرمایه گذاری در شبکه ادامه دهد. از طرف دیگر، شرکتهای تجاری نیز مایل بودند به شبکه NSFNET ملحق شوند، ولی مقررات NSF



شکل ۱-۲۸. ستون فقرات NSFNET در سال ۱۹۸۸.

کاربردهای انتفاعی شبکه را ممنوع کرده بود. متعاقب آن، NSF بعنوان اولین قدم بسوی تجاری کردن شبکه، شرکت‌های IBM، MERIT و MCI را به ایجاد یک مؤسسه غیرانتفاعی (بنام Advanced Networks and Services - ANS) ترغیب کرد. در سال ۱۹۹۰، ANS کنترل NSFNET را بدست گرفت، و با ارتقاء لینکهای 1.5-Mbps به 45-Mbps شبکه ANSNET را بوجود آورد. این شرکت بعد از ۵ سال کار به AOL (America Online) فروخته شد. اما در آن زمان دیگر شرکت‌های بسیاری سرویسهای تجاری IP ارائه می‌کردند، و روشن شده بود که دولت باید پای خود را از تجارت شبکه بیرون بکشد.

برای تسهیل امور (و اطمینان از اینکه تمام شبکه‌های منطقه‌ای می‌توانند با هم تماس بگیرند)، NSF چهار قرارداد با شرکت‌های بزرگ برای ایجاد نقطه دسترسی شبکه (Network Access Point - NAP) امضا کرد. این چهار شرکت عبارت بودند از: PacBell (در منطقه سانفرانسیسکو)، Ameritech (در منطقه شیکاگو)، MFS (در منطقه واشینگتن دی.سی.)، Sprint (در منطقه نیویورک). هر اپراتور شبکه که بخواهد سرویسهای ستون فقرات به شبکه‌های منطقه‌ای NSF بدهد، بایستی به تمام NAP ها متصل باشد.

بدین ترتیب، هر بسته که بخواهد از یک منطقه به منطقه دیگر برود، می‌تواند از هر یک از این ستونهای فقرات استفاده کند، که نتیجه آن ایجاد رقابت برای سرویس بهتر و قیمت کمتر است. با این تمهید، ستون فقرات منحصر بفرد دولتی جای خود را به یک زیرساخت متنوع و رقابتی داد. بسیاری از افراد دولت فدرال را به گناه عدم خلاقیت سرزنش می‌کنند، ولی در واقع این بنیاد ملی علوم و وزارت دفاع بودند که زیرساخت‌های اینترنت را شکل داده و سپس اداره آنرا به بخش خصوصی سپردند.

در دهه ۱۹۹۰ مناطق و کشورهای بسیاری، با تأثیرپذیری از الگوی آرپانت و NSFNET، شبکه‌های ملی تحقیقاتی خود را بوجود آوردند. در اروپا، این شبکه‌ها (که EuropaNET و EBONE نام داشتند) از لینکهای 2-Mbps شروع کردند، و به 34-Mbps ارتقاء یافتند. در آنجا نیز زیرساخت‌های شبکه بتدریج به بخش خصوصی محول شد.

کاربردهای اینترنت

بعد از آنکه در اول ژانویه ۱۹۸۳ TCP/IP بعنوان تنها پروتکل رسمی آرپانت معرفی شد، تعداد شبکه‌ها، کامپیوترها و کاربران متصل به آن بسرعت افزایش یافت؛ و وقتی آرپانت و NSFNET به هم متصل شدند، رشد آن

حالت نمایی بخود گرفت. بسیاری از مناطق و کشورها (از جمله کانادا، اروپا و اقیانوسیه) به شبکه ملحق شدند. در اواسط دهه ۱۹۸۰ دیگر افراد به این مجموعه به عنوان شبکه ای از شبکه ها (که بعدها به اینترنت معروف شد) نگاه می کردند، بدون آنکه هیچگونه بخشنامه رسمی در کار باشد، یا حتی مراسم افتتاحیه ای (با قیچی و نوارهای رنگی، و تشویق و هورا) برگزار شده باشد.

چسبی که اینترنت را به هم متصل نگه می دارد، مدل TCP/IP و مجموعه پروتکل های آن است. پذیرش TCP/IP باعث شد تا سرویس های جهانی بتوانند جنبه عملی بخود بگیرند.

اما واقعاً «روی اینترنت بودن» چه معنایی دارد؟ طبق تعریف ما، ماشینی روی اینترنت است که مجموعه پروتکل های TCP/IP را اجرا کند، یک آدرس IP داشته باشد، و بتواند بسته های IP را به تمام ماشین های دیگری که روی اینترنت هستند، بفرستد. صرف توانایی ارسال و دریافت ایمیل به معنای بودن روی اینترنت نیست، چون سرویس های ایمیل می تواند به شبکه های خارج از اینترنت هدایت شود. با این حال، اوضاع با وضعیتی که در حال حاضر وجود دارد (میلیونها کامپیوتر شخصی می توانند با مودم به یک ISP وصل شده، یک آدرس IP موقت بگیرند، و بسته های IP رد و بدل کنند)، کمی مغشوش و مبهم است. اما، مادامیکه این کامپیوترها به مسیریاب ISP متصل هستند، پُر بیراه نیست که آنها را روی اینترنت بدانیم.

اینترنت سستی (از ۱۹۷۰ تا اوایل دهه ۱۹۹۰) چهار کاربرد عمده داشت:

۱. ایمیل (e-mail) - نوشتن، ارسال و دریافت نامه های پُست الکترونیک از همان روزهای اول آرپانت جزء سرویس های آن بود، و همچنان یکی از محبوبترین هاست. امروزه بسیاری از افراد روزانه دهها و صدها ایمیل دریافت می کنند، و به آن بعنوان درجه ای برای ارتباط با دنیای خارج نگاه می کنند - بسیار بیشتر از تلفن یا پُست معمولی.
۲. اخبار (news) - گروه خبری (newsgroup) یک محفل اختصاص یافته برای تبادل پیام در یک زمینه خاص است. امروزه هزاران گروه خبری در زمینه های فنی و غیرفنی (از جمله کامپیوتر، علوم، هنر و سیاست) وجود دارند. هر گروه خبری برای خود قواعد و مقرراتی دارد، که سرپیچی از آنها را برنمی تابد.
۳. ورود از راه دور (remote login) - هر روز هزاران نفر در سراسر دنیا برای ورود به کامپیوترهای دیگر از طریق اینترنت (البته آنهایی که حق ورود به آنها را داشته باشند)، از برنامه هایی مانند rlogin، telnet یا ssh استفاده می کنند.
۴. انتقال فایل (file transfer) - با استفاده از برنامه های FTP، کاربران اینترنت می توانند فایل های خود را از یک ماشین به ماشین دیگر کپی کنند. جریان انتقال دانش از این طریق بسیار گسترده و متنوع است.

تا اوایل دهه ۱۹۹۰ اینترنت جولانگاه دانشگاهیان، کارمندان دولت و محققان صنعتی بود؛ اما یک برنامه کاربردی جدید بنام WWW (World Wide Web) این وضعیت را بکلی تغییر داد، و میلیونها نفر افراد عادی نیز توانستند به کاربران حرفه ای اینترنت ملحق شوند. این برنامه، که توسط تام برنرز-لی (Tom Berners-Lee) از فیزیکدانان مرکز تحقیقات هسته ای اروپا (CERN) ابداع شد، هیچ یک از سرویس های اینترنت را عوض نکرد، ولی کاربرد آنها را ساده تر کرد. به کمک این تکنولوژی جدید، و برنامه مرورگر موزائیک (Mosaic browser)، که توسط مارک آندرسن در مرکز ملی کاربردهای ابرکامپیوتر (NCSA) نوشته شد، WWW ایجاد سایت هایی متشکل از صفحات مختلف (و با اطلاعاتی در قالب های متن، تصویر، صدا، و حتی ویدئو)، با لینک هایی به صفحات دیگر را امکان پذیر کرد. با کلیک کردن روی یک لینک (link)، کاربر مستقیماً به صفحه ای که مشخص شده، می پرد. حتماً سایت های بسیاری متعلق به شرکت های بزرگ را دیده اید، که می توانید با کلیک کردن هر یک از لینک های آن، به صفحه مربوطه (مثلاً، صفحه مربوط به محصولات شرکت، لیست قیمت های آن، پشتیبانی فنی، فروش و غیره) وارد شوید. در زمانی کوتاه صفحات جدید و متنوعی به WWW اضافه شد، صفحاتی مانند نقشه شهرها و کشورها،

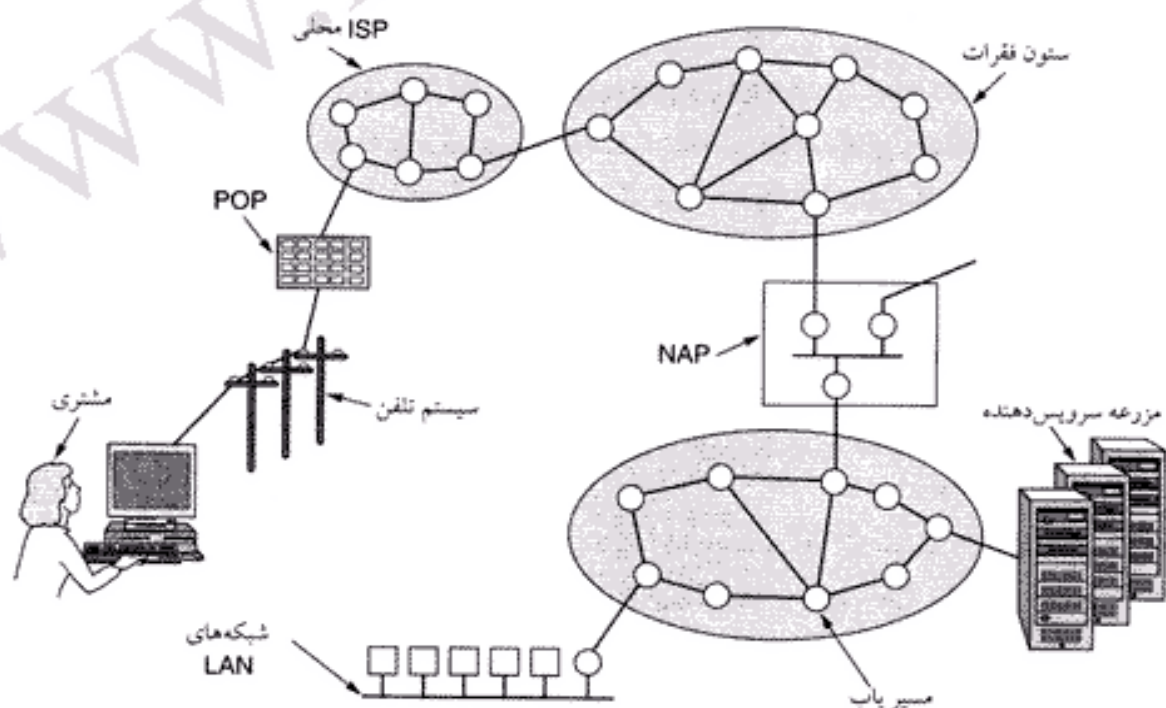
جدول قیمت سهام، کاتالوگ کارتهای کتابخانه‌ها، برنامه‌های رادیویی، و حتی متن کامل کتابهایی که از شمول قانون حق‌التألیف خارج شده‌اند (مانند کتابهای مارک تواین، چارلز دیکنز، و امثالهم). حتی بسیاری از افراد عادی نیز برای خود سایت (صفحات خانگی) ساخته‌اند.

موتور محرکه این رشد، شرکتهای ارائه‌دهنده سرویس اینترنت (ISP - Internet Service Provider) بودند. این شرکتها به افراد اجازه می‌دادند تا از خانه و با کامپیوترهای شخصی خود به اینترنت متصل شده، و از سرویسهای آن استفاده کنند. در دهه ۱۹۹۰، این شرکتها هر ساله برای دهها میلیون نفر امکان دسترسی اینترنت فراهم کردند، و چهره آنرا از محیطی دانشگاهی و نظامی به یک شبکه عمومی تغییر دادند. تعداد دقیق کاربران اینترنت در حال حاضر معلوم نیست، ولی محققاً سر به صدها میلیون نفر می‌زند، و خیلی زود از مرز یک میلیارد خواهد گذشت.

معماری اینترنت

در این قسمت سعی می‌کنیم تصویری کلی از اینترنت بدست دهیم (شکل ۱-۲۹ را ببینید). به دلیل شباهتها و تداخل وظایف زیادی که بین شرکتهای مخابرات و ISP ها وجود دارد، امروزه اوضاع بسیار در هم و مغشوش است، و بسختی می‌توان گفت کی چکاره است - به همین دلیل توضیحات ذیل ساده‌تر از آن چیز است که در واقعیت وجود دارد. اجازه دهید شکل ۱-۲۹ را جزء به جزء بررسی کنیم.

بهترین نقطه برای شروع، خانه مشتری (client) است. در اینجا فرض را بر این گذاشته‌ایم که مشتری با استفاده از یک مودم و خط تلفن به ISP متصل می‌شود. مودم (modem) وسیله‌ایست که سیگنالهای دیجیتالی کامپیوتر را به سیگنالهای آنالوگ تبدیل می‌کند، تا این سیگنالها بتوانند بدون اعوجاج روی خطوط تلفن منتقل شوند. این سیگنالها در نقطه تماس ISP (که به POP - Point Of Presence - معروف است) مجدداً تبدیل به سیگنالهای دیجیتالی شده، و وارد شبکه منطقه‌ای ISP می‌شود. از این نقطه به بعد سیستم کاملاً دیجیتال است، و بر مبنای سوئیچ بسته کار می‌کند. اگر این ISP همان شرکت مخابرات باشد، POP در مرکز سوئیچینگ تلفن واقع



شکل ۱-۲۹. یک تصویر کلی از اینترنت.

خواهد بود؛ اما اگر ISP و شرکت مخابرات یکی نباشند، POP یک مرکز سوییچینگ کوچک بین راهی خواهد بود، که از آنجا به شبکه تلفن وصل می شود.

شبکه منطقه ای هر ISP از چند مسیر یاب، که به شهرهای مختلف تحت پوشش آن ISP سرویس می دهند، تشکیل می شود. اگر مقصد بسته ارسال شده از مشتری یکی از کامپیوترهای واقع در همان شبکه منطقه ای ISP باشد، بلافاصله به آن تحویل داده می شود. ولی اگر چنین نباشد، بسته به اپراتور ستون فقرات ISP داده خواهد شد. اپراتور ستون فقرات (backbone operator) بالاترین نقطه این زنجیره است (شرکتهای At&T و Sprint جزء اپراتورهای عمده هستند). هر اپراتور یک شبکه بزرگ از ستونهای فقرات بین المللی (مشکل از هزاران مسیر یاب که با فیبرهای نوری پرسرعت به هم متصلند) را اداره می کند. شرکتهای بزرگ و آنهایی که سرویسهای میزبانی اینترنت ارائه می کنند، معمولاً مستقیماً به ستون فقرات متصل هستند. اپراتورهای ستون فقرات این نوع خدمات را تشویق می کنند، و برای آن تسهیلات ویژه ای بنام کاریر کرایه ای فراهم می آورند، که بعلاوه ارتباط نزدیک با ستون فقرات از سرعتهای بسیار بالایی برخوردارند.

اگر مقصد بسته ارسال شده یکی از ISP های متصل به ستون فقرات باشد، به نزدیکترین مسیر یاب فرستاده می شود و از آنجا بدست وی خواهد رسید. با این حال در دنیای ستونهای فقرات متعددی (با سرعتهای مختلف) وجود دارند، و احتمال دارد که این بسته وارد یکی از این شاهراههای رقیب شود. برای اینکه بسته ها بتوانند براحتی بین شاهراهها حرکت کنند، تمام آنها باید به یک NAP متصل باشند - NAP چیزی نیست بیش از اتاقی پُر از مسیر یابهای متعدد (حداقل یکی به ازای هر شاهراه)، که در یک LAN ساده به هم متصل شده اند. شاهراههای بزرگ، غیر از اتصال از طریق NAP، معمولاً بصورت مستقیم نیز به شاهراههای دیگر راه دارند (که به آن ارتباط دوجانبه گفته می شود). یکی از تناقضهای بزرگ اینترنت آن است که شرکتهایی که در انتظار عموم با هم رقابت سخت دارند، در خفا با یکدیگر ارتباطات نزدیک و تنگاتنگ برقرار می کنند (Metz, 2001).

این هم از مرووری اجمالی بر اینترنت. البته در فصلهای آینده تک تک این اجزاء، الگوریتمها، و پروتکلها را مفصلاً بررسی خواهیم کرد. نکته ای که جالبست بدانید اینست که امروزه بسیاری از شرکتهای ارتباطات داخلی شبکه خود را بر اساس مدل و تکنولوژیهای اینترنت بنا می کنند، چیزی که به اینترانت (intranet) معروف است.

۲-۵-۱ شبکه های اتصال-گرا: X.25، Frame Relay و ATM

از همان اولین روزهایی که شبکه پا به عرصه وجود گذاشت، جنگ بین طرفداران زیرشبکه های اتصال-گرا و شبکه های غیر متصل (دیتاگرام) نیز شروع شد. مهمترین برگ برنده طرفداران زیرشبکه های غیر متصل همان آرپانت/اینترنت است. بیاد دارید که قصد اولیه وزارت دفاع آمریکا از بنیانگذاری آرپانت، ایجاد شبکه ای بود که بتواند در مقابل ضربات هسته ای (و منهدم شدن بخش بزرگی از خطوط و تجهیزات انتقال) دوام بیاورد (در واقع، هدف اصلی این طرح بالا بردن ضریب تحمل خرابی شبکه بود). این رهیافت منجر به طراحی شبکه ای شد که در آن هر بسته راه خود را مستقل از بسته های دیگر طی می کند. بدین ترتیب، اگر تعدادی از مسیر یابهای شبکه از مدار خارج شوند، مادامیکه شبکه بتواند مسیرهای جدید خود را از نو پیگیرندی کند، در ارسال بسته ها از مبدأ به مقصد خللی پیش نخواهد آمد.

طرفداران زیرشبکه های اتصال-گرا معمولاً همان شرکتهای تلفن هستند. در این سیستم، آغازکننده ارتباط قبل از آنکه بتواند ارسال اطلاعات را شروع کند، بایستی منتظر برقراری ارتباط مستقیم با طرف مقابل بماند. این ارتباط فیزیکی در تمام طول تماس برقرار می ماند، و تمام بسته های اطلاعات از همین مسیر واحد عبور خواهند کرد. اگر هر یک از تجهیزات این مسیر به هر دلیلی از کار بیفتند، تماس قطع خواهد شد - چیزی که وزارت دفاع مسلماً نمی پسندد.

پس علت علاقه شرکت‌های تلفن به این سیستم چیست؟ دو دلیل اصلی این علاقه عبارتند از:

۱. کیفیت سرویس
۲. حسابرسی مصرف‌کنندگان

در شبکه‌های اتصال-گرا هر تماس مقداری از منابع زیرشبکه (از قبیل توان پردازشی مسیریاب‌ها) را بخود اختصاص می‌دهد، و در صورتیکه این منابع به حالت اشباع برسند، تماس جدید امکانپذیر نبوده، و کاربر با بوق اشغال روبرو خواهد شد. در این روش تماس‌ها (بدلیل اختصاص منابع کافی) از کیفیت بالایی برخوردار هستند. از طرف دیگر، اگر در شبکه‌های غیرمتصل تعداد زیادی بسته به یکباره وارد یک مسیریاب شوند، ممکنست برخی از آنها (بدلیل کمبود امکانات پردازشی) در داخل رواتر از بین بروند. البته فرستنده متوجه این نقص خواهد شد، و بسته‌های گمشده را از نو ارسال خواهد کرد، ولی همین موضوع باعث افت کیفیت شبکه (بویژه در مورد صدا و تصویر) می‌شود. لازم به گفتن نیست که شرکت‌های تلفن بیش از هر چیز نگران کیفیت صدا هستند، و به همین دلیل همچنان زیرشبکه‌های اتصال-گرا را ترجیح می‌دهند.

دلیل دومی که شرکت‌های تلفن سرویسهای اتصال-گرا را بیشتر می‌پسندند، امکان صدور صورتحساب برای مشترکان است (کاری که مدتهاست به آن عادت کرده‌اند). هزینه تماسهای بین شهری و خارج از کشور معمولاً بر حسب مدت مکالمه محاسبه می‌شود (علت اتخاذ این روش هم پیشتر سادگی آن بوده است). اگر تماس مستقیمی بین دو طرف مکالمه برقرار نباشد، طبعاً این شرکتها نمی‌توانند برای مشترکان خود صورتحساب صادر کنند.

سیستمهای محاسبه صورتحساب هزینه بسیار سنگینی به شرکت‌های تلفن تحمیل می‌کند. اگر یک شرکت تلفن بر اساس یک شارژ ثابت ماهیانه از مشترکان خود پول دریافت کند، علیرغم بالا رفتن مصرف مشترکان، می‌تواند پول زیادی صرفه‌جویی کند. اما عوامل زیادی (که بیشتر آنها هم سیاسی هستند) با اتخاذ این روش مخالفت می‌کنند. جالبست بدانید که در اغلب سیستمهای مشابه (مانند تلویزیونهای کابلی و برخی پارکهای تفریحی) از روش محاسبه ثابت استفاده می‌شود. در این سیستمها هم امکان پرداخت به ازای مصرف وجود دارد، ولی بدلیل هزینه بالای صدور صورتحساب معمولاً از آن اجتناب می‌شود.

به دلایل فوق، جای تعجب نیست که شرکت‌های تلفن طرفدار زیرشبکه‌های اتصال-گرا باشند؛ اما تعجب‌برانگیز این است که، اینترنت هم دارد به همین سمت پیش می‌رود - البته با این استدلال که این کار باعث بالا رفتن کیفیت سرویسهای صدا و تصویر آن خواهد شد. اکنون اجازه دهید چند شبکه اتصال-گرا را بهتر بشناسیم.

Frame Relay و X.25

اولین شبکه اتصال-گرا که وارد سرویس عمومی شد، شبکه X.25 بود. این شبکه در اوایل دهه ۱۹۷۰، و در زمانی طراحی شد که شرکت‌های تلفن بصورت انحصاری عمل می‌کردند، و هر کشور شبکه ملی خاص خود را داشت. برای استفاده از X.25، ابتدا کامپیوتر مبدأ با ماشین مقصد تماس تلفنی برقرار می‌کرد. از آنجائیکه در آن واحد تماسهای مختلفی می‌توانست وجود داشته باشد، به هر تماس تلفنی یک شماره داده می‌شد. بسته‌های داده بسیار ساده بودند: یک سرآیند ۳ بیتی و بدنه‌ای متشکل از ۱۲۸ بایت. سرآیند (header) تشکیل می‌شد از یک شماره تماس ۱۲ بیتی، یک شماره ترتیب بسته (packet sequence number)، یک عدد تصدیق دریافت (acknowledgement number)، و چند بیت متفرقه. شبکه‌های X.25 به مدت نزدیک به یک دهه با موفقیتی نسبی کار کردند.

در دهه ۱۹۸۰ شبکه‌های X.25 جای خود را به نوع جدیدی از شبکه‌های اتصال-گرا بنام frame relay (رله فریم) دادند. این شبکه جدید اساساً هیچ نوع کنترل خطا و کنترل جریانی نداشت، و بسته‌ها به همان ترتیب

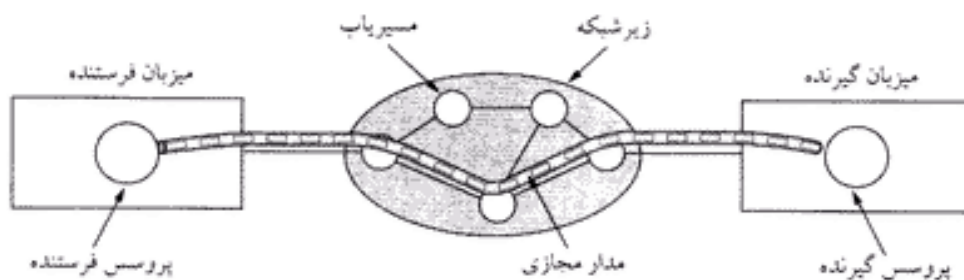
دریافت در مقصد تحویل می شدند (البته اگر به مقصد می رسیدند). این سه خصوصیت (فقدان کنترل خطا، فقدان کنترل جریان، و تحویل ترتیبی بسته ها) شبکه های frame relay را بسیار شبیه یک LAN بزرگ می کند، و در واقع بزرگترین کاربرد آن هم همین است: اتصال چند LAN دور از هم، و ایجاد یک LAN بزرگ. شبکه های frame relay هم نسبتاً موفق بودند، و هنوز در برخی جاها از آنها استفاده می شود.

حالت انتقال آسنکرون (ATM)

یکی دیگر از شبکه های اتصال-گرا (که اهمیت بسیار بیشتری نیز دارد) شبکه حالت انتقال آسنکرون (Asynchronous Transfer Mode - ATM) است. علت این نامگذاری عجیب آن است که در شبکه های تلفن اکثر تماسها بصورت سنکرون (synchronous - وابسته به پالس ساعت) هستند، در حالیکه ATM چنین نیست. شبکه ATM در اوایل دهه ۱۹۹۰ طراحی شد، و سروصدای زیادی نیز بپا کرد (Ginburg, 1996; Gorlaski, 1995; Ibe, 1997; Kime et al., 1994; Stallings, 2000). شبکه های ATM با ادغام تمام انواع شبکه و سیستم های مخابراتی (صدا، داده، تلویزیون کابلی، تلکس، تلگراف، کبوترهای نامه بر، قوطی های حلی سیمی، طیف های افریقایی، علامت های دودی سرخپوستان، و خلاصه هر چیزی که به نوعی اطلاعات منتقل می کند) به میدان آمدند - اتفاقی که هرگز نیفتاد. علت آن هم تا تقریباً شبیه همان بلایی بود که سر OSI آمد (زمان نامناسب، تکنولوژی بد، پیاده سازی نامناسب، و سیاست های غلط). شرکت های اینترنتی که منتظر وسیله ای بودند تا شرکت های تلفن را در همان راند اول از پا در آورند، به ATM امید بستند. اما این امید دیری نپایید، و شرکت های اینترنتی (حتی سرسخت ترین آنها) بزودی دریافتند که تار رسیدن به سرویس های مطلوب راه درازی در پیش دارند. البته ATM از OSI بسیار موافقت بود، و حتی امروز هم در شبکه های تلفن (و برای انتقال بسته های IP) مورد استفاده قرار می گیرد. از آنجائیکه این زیرشبکه فقط برای ارتباطات داخلی بکار می رود، اغلب کاربران معمولی از وجود آن اطلاعی ندارد، ولی ATM زنده و سر حال است.

مدار مجازی ATM

از آنجائیکه شبکه های ATM از نوع اتصال-گرا هستند، برای برقراری ارتباط اولیه ابتدا باید یک بسته خاص بفرستند. با عبور این بسته از زیرشبکه، تمام مسیر یاب هایی که در مسیر آن قرار دارند، آنرا در جدول های خود ثبت می کنند و منابع لازم را برای آن کنار می گذارند. به ارتباطی که بدین طریق برقرار می شود، مدار مجازی (virtual circuit) می گویند، چون بسیار شبیه مدار های فیزیکی در شبکه های تلفن است (شکل ۱-۳۰ را ببینید). بسیاری از شبکه های ATM از مدار های مجازی دائمی بین دو نقطه پشتیبانی می کنند (که بسیار شبیه خطوط اجاره ای در سیستم تلفن معمولی است). هر اتصال (موقت یا دائم) دارای یک شماره شناسایی است. بعد از برقراری ارتباط، دو طرف می توانند شروع به فرستادن داده کنند. ایده اصلی در ATM ارسال داده ها در بسته های کوچک و با اندازه ثابت، بنام سلول (cell)، است. هر سلول ۵۳ بایت طول دارد، که ۵ بایت آن سرآیند، و



شکل ۱-۳۰. یک مدار مجازی.

بایت	5	48
سرآیند	داده های کاربر	

شکل ۱-۳۱. یک سلول ATM.

۴۸ بایت باقیمانده داده هاست (شکل ۱-۳۱). شماره شناسایی اتصال در سرآیند سلولها نوشته می شود، بطوریکه تمام مسیرپای های مسیر می توانند تشخیص دهند که هر سلول متعلق به کدام اتصال است، و چگونه باید آنرا هدایت کنند. هدایت سلولها بصورت سخت افزاری (و با سرعت فوق العاده بالا) صورت می گیرد - در واقع، علت اصلی اندازه ثابت سلولها در شبکه های ATM اینست که ساخت مسیرپای های سخت افزاری برای آن بسیار ساده است (هدایت بسته های IP با اندازه متغیر به مسیرپای های نرم افزاری نیاز دارد، که بسیار کندتر هستند).

مزیت دیگر ATM توانایی آن در ارسال همزمان یک سلول به مسیرهای مختلف است - که این ویژگی در سیستمهای پخش تلویزیونی بسیار مفید است. از طرف دیگر، کوچک بودن سلولها باعث می شود تا هیچ خطی برای مدت طولانی اشغال نشود، و کیفیت سرویس افزایش یابد.

در ATM تمام سلولها از یک مسیر به مقصد هدایت می شوند. البته تضمینی برای رسیدن یک سلول به مقصد وجود ندارد، ولی ترتیب آنها حتماً رعایت می شود. اگر سلول ۲ بعد از سلول ۱ فرستاده شده باشد، به همان ترتیب به مقصد می رسند، و هرگز سلول ۲ پیش از سلول ۱ به مقصد نخواهد رسید. اگر هر یک از این سلولها (با هر دوی آنها) در بین راه از بین بروند، این بر عهده پروتکل های لایه های بالاتر است که آنها را بازیابی کنند. از این نظر ATM حداقل یک پله بالاتر از اینترنت می ایستد (که نه ترتیب بسته ها ضمانت می شود، نه حتی رسیدن صحیح و سالم آنها).

سازماندهی شبکه های ATM شبیه WAN های قدیمی (متشکل از خطوط تلفنی و سونیچها) است. متداولترین سرعتها در شبکه های ATM عبارتند از 155-Mbps و 622-Mbps (البته ATM از سرعت های بالاتر هم پشتیبانی می کند). علت انتخاب سرعت 155-Mbps آنست که تلویزیونهای با وضوح بالا (HDTV) به چنین سرعتی نیاز دارند - مقدار دقیق این سرعت 155.52-Mbps است، که دقیقاً معادل سرعت سیستم AT&T SONET می باشد. علت انتخاب سرعت 622-Mbps نیز اینست که از ترکیب چهار کانال 155.52-Mbps یک کانال 622-Mbps بوجود می آید.

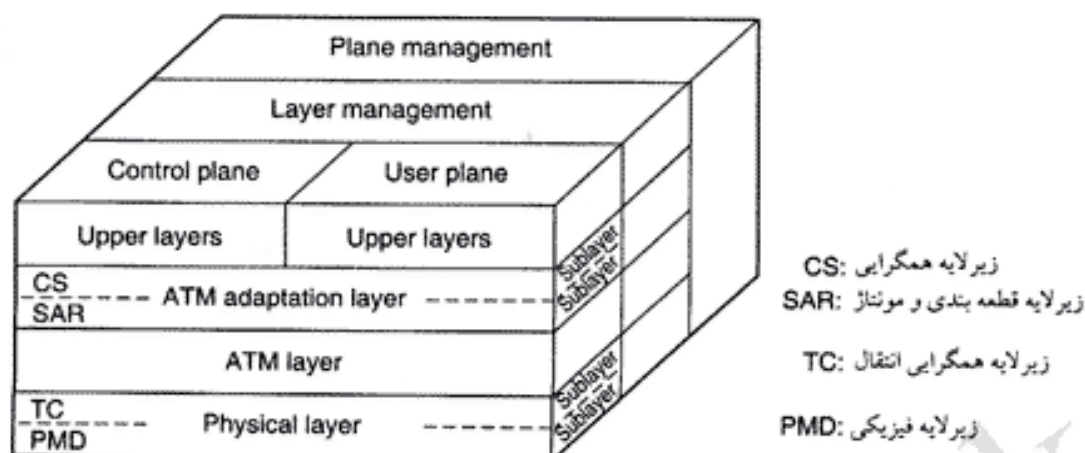
مدل مرجع ATM

شبکه ATM برای خود یک مدل مرجع مستقل دارد، که با مدل های OSI و TCP/IP فرق دارد - این مدل را در شکل ۱-۳۲ ملاحظه می کنید. این مدل سه لایه دارد: لایه فیزیکی، لایه ATM، لایه انطباق ATM (و هر چند لایه که کاربر مایل باشد روی این لایه ها سوار کند).

لایه فیزیکی با مشخصات فیزیکی سیستم (ولتاژها، زمانبندی بیت ها و غیره) سرو کار دارد. مدل ATM هیچ پیشینازی در مورد این مشخصات ندارد، و می گوید که ارسال سلولها می تواند بصورت مستقیم یا از طریق سیستمهای انتقال دیگر انجام شود. بعبارت دیگر، ATM مستقل از سیستم انتقال است.

لایه ATM با خود سلولها و انتقال آنها سرو کار دارد. ایجاد و رها کردن مدار مجازی، تعریف فیلدهای سرآیند سلول، و کنترل ازدحام (congestion control) از وظایف این لایه است.

از آنجائیکه اکثر برنامه های کاربردی تمایلی به کار کردن با بسته هایی به کوچکی سلولهای ATM ندارند، یک لایه دیگر بالای لایه ATM تعبیه شده تا این قبیل برنامه ها بتوانند بسته های بزرگتری به ATM بفرستند. این لایه



شکل ۱-۳۲. مدل مرجع ATM.

(در سمت فرستنده) بسته های داده را به سلولهای ۵۳ بایتی می شکند، و در طرف گیرنده آنها را دوباره سر هم می کند. نام این لایه، لایه انطباق ATM (ATM Adaptation Layer - AAL) است. بر خلاف مدل های قبلی که دو بُعدی بودند، مدل ATM یک مدل مرجع سه بُعدی است (شکل ۱-۳۲ را ببینید). صفحه کاربر (user plane) با انتقال داده، کنترل جریان، تصحیح خطا، و دیگر عملکردهای کاربر سروکار دارد. از طرف دیگر، صفحه کنترل (control plane) مدیریت اتصال را بر عهده دارد. وظیفه صفحه های مدیریت لایه (layer management) و مدیریت صفحه (plane management) مدیریت منابع سیستم و هماهنگ کردن لایه های بینابینی است.

لایه های فیزیکی و AAL هر یک به دو زیر لایه تقسیم شده اند، یکی در پائین برای انجام عملکردهای محوله، و دیگری در بالا برای ارتباط با لایه بالاتر (که زیر لایه همگرایی - convergence sublayer - خوانده می شود). وظیفه هر یک از این لایه ها و زیر لایه ها را در شکل ۱-۳۳ ملاحظه می کنید.

OSI لایه	ATM لایه	ATM زیر لایه	کارکرد
3/4	AAL	CS	واسط استاندارد
		SAR	قطعه بندی و مونتاژ
2/3	ATM		کنترل جریان تولید سرآیند سلول مدار مجازی - مدیریت مسیر ماتری پلکس انتقالی پلکس سلول
2	Physical	TC	ایزوله کردن سرعت سلول تولید مجموع تطبیقی تولید سلول بسته بندی و بازکردن بسته ها تولید فریم
1		PMD	زمان بندی بیت دسترسی فیزیکی

شکل ۱-۳۳. وظایف لایه ها و زیر لایه های ATM.

زیرلایه PMD (Physical Medium Dependent) مستقیماً به کابل شبکه وصل می‌شود، و کار آن ارسال و دریافت بیت‌ها و ایجاد همزمانی بین آنهاست. هر نوع کابل و سیستم انتقال زیرلایه PMD خاص خود را دارد. زیرلایه دیگر لایه فیزیکی، TC (Transmission Convergence) نام دارد. وقتی یک سلول ارسال می‌شود، لایه TC آنرا بصورت جریانی از بیت‌ها به لایه PMD می‌فرستد - که این کاری ساده است. در طرف گیرنده، لایه TC باید جریان بیت‌هایی را که از لایه PMD دریافت می‌کند، دوباره بصورت سلول در آورد - عبارت دیگر باید بتواند ابتدا و انتهای هر سلول را بدرستی تشخیص دهد. در مدل ATM این کار در لایه فیزیکی انجام می‌شود، وظیفه‌ی که در مدل OSI (و تقریباً تمام مدل‌های دیگر) بر عهده لایه پیوند داده است.

مانطور که قبلاً هم گفتیم، لایه ATM مدیریت ایجاد و انتقال سلولها را بر عهده دارد. مهمترین بخش از وظایف ATM نیز در همین لایه صورت می‌گیرد. این لایه تلفیقی است از لایه‌های لینک داده و شبکه در مدل OSI - که در ضمن هیچ زیرلایه‌ای هم ندارد.

لایه AAL به دو زیرلایه SAR (Segmentation And Reassembly) و CS (Convergence Sublayer) تقسیم شده است. لایه پائینی (SAR) در طرف فرستنده بسته‌های داده را به سلول می‌شکند، و در طرف گیرنده دوباره آنها را به هم می‌چسباند. لایه بالایی (CS) وظیفه ارائه سرویسهای مختلف به برنامه‌های کاربردی را بر عهده دارد.

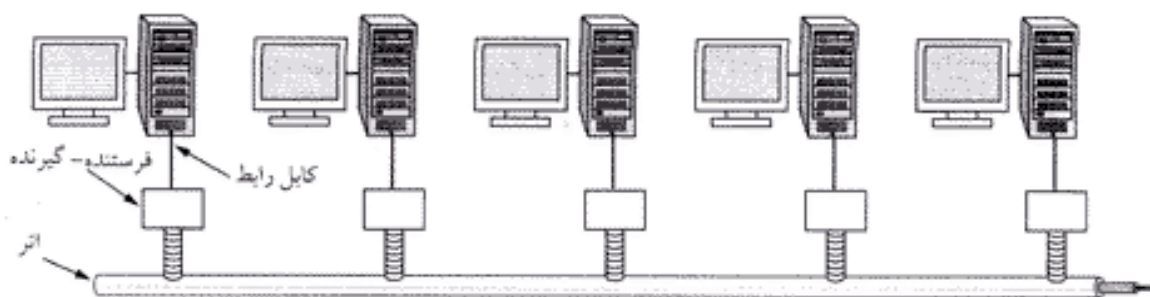
از آنجائیکه ATM در سراسرییی زوال قرار دارد، در این کتاب بیش از این درباره آن صحبت نخواهیم کرد. با این حال، بدلیل نصب در مقیاس وسیع، به احتمال زیاد تا چند سال دیگر نیز دوام خواهد آورد. برای کسب اطلاعات بیشتر درباره ATM به (Dobrowski and Grise, 2001; Gadecki and Heckart, 1997) مراجعه کنید.

۳-۵-۱ اینترنت

اینترنت و ATM هر دو شبکه‌های گسترده هستند، ولی در هر شرکت، سازمان و دانشگاه تعداد زیادی کامپیوتر وجود دارد که باید به هم متصل شوند. از همین جاست که نیاز به شبکه‌های محلی شکل می‌گیرد. در این قسمت می‌خواهیم کمی درباره متداولترین شبکه محلی، یعنی اینترنت (Ethernet)، صحبت کنیم.

داستان ما از ایالت بکر و دست نخورده‌ی هاوایی، در اوایل دهه ۱۹۷۰، شروع می‌شود - در اینجا منظور از «بکر و دست نخورده» فقدان شبکه تلفن است. با اینکه نبود شبکه تلفن برای کسانی که هاوایی را برای استراحت انتخاب می‌کردند، یک مزیت بود، ولی برای محقق بنام نورمن آبرامسون و همکارانش در دانشگاه هاوایی که می‌خواستند کاربران جزایر دورافتاده‌ی هاوایی را به کامپیوتر مرکزی در هونولولو (مرکز ایالت هاوایی) متصل کنند، چندان خوشایند نبود. کشیدن کابل از وسط اقیانوس آرام مسلماً نمی‌توانست راه حل مشکل آنها باشد، پس باید فکر دیگری می‌کردند.

یکی از راه‌حل‌هایی که آنها پیدا کردند، استفاده از امواج رادیویی با بُرد کوتاه بود. هر ترمینال به یک رادیوی کوچک وصل می‌شد، که دو فرکانس داشت: فرکانس ارسال (به کامپیوتر مرکزی)، فرکانس دریافت (از کامپیوتر مرکزی). وقتی کاربر می‌خواست به کامپیوتر مرکزی وصل شود، روی فرکانس ارسال یک بسته می‌فرستاد. اگر کس دیگری در همان لحظه در حال ارسال نبود، بسته مزبور به کامپیوتر مرکزی می‌رسید، و کاربر می‌توانست بسته تصدیق دریافت (acknowledgement) آنرا روی فرکانس دریافت بگیرد. اما اگر کانال اشغال بود، ترمینال بسته تصدیق دریافت را نمی‌گرفت، و متوجه می‌شد که باید دوباره سعی کند. از آنجائیکه روی کانال دریافت فقط یک کامپیوتر (کامپیوتر مرکزی) مجاز به ارسال اطلاعات بود، هرگز در آن انسداد پیش نمی‌آمد. این سیستم، که آلوهانت (ALOHA NET) نام گرفت، در شرایط ترافیک پائین خوب کار می‌کرد، ولی در ترافیک بالا بشدت ناکارآمد می‌شد.



شکل ۱-۳۴. معماری اترنت اولیه.

در همان زمان، دانشجویی بنام باب متکالف که تازه از M.I.T فارغ التحصیل شده بود، به قصد ادامه تحصیل در مقطع دکترا وارد دانشگاه هاروارد شد. باب در حین مطالعات خود با کارهای آبرامسون آشنا، و بشدت به آن علاقمند شد. این علاقه تا آن حد بود که باب تصمیم گرفت قبل از شروع به کار در مرکز تحقیقات زیراکس در پالو آلتو (Xerox PARC)، تابستان را در هاوایی بگذراند و با آبرامسون کار کند. وقتی باب متکالف به Xerox PARC برگشت، متوجه شد که محققان آنجا روی پروژه ای کار می کنند که بعدها به کامپیوتر شخصی (Personal Computer - PC) معروف شد. ولی این یک ماشین ایزوله و جدا از همه جا بود. باب، با استفاده از تجارب آبرامسون، و به کمک یکی از همکارانش بنام دیوید باگزر، اولین شبکه محلی را طراحی و پیاده سازی کرد (Metcalfe and Boggs, 1976).

آنها این سیستم را (بیاد ماده ای خیالی بنام اتر - ether - که تا مدتها تصور می شد محیط انتشار امواج الکترومغناطیس است) اترنت نامیدند. (بعد از کشف معادلات انتشار امواج الکترومغناطیس توسط فیزیکدان بریتانیایی، جیمز کلارک ماکسول، دانشمندان فرض را بر این گذاشتند که این امواج در محیطی بنام اتر منتشر می شوند. فقط بعد از آزمایشات معروف مایکلسون-مورلی بود که فیزیکدانان دریافتند امواج الکترومغناطیس می توانند در خلاء منتشر شوند.)

رسانه انتشار در این سیستم خلاء نبود، بلکه از یک رشته کابل هم محور (coaxial) ضخیم بطول حداکثر ۲/۵ کیلومتر (با یک تکرار کننده در هر ۵۰۰ متر) استفاده می شد (این کابل در واقع همان اتر محسوب می شد) - شکل ۱-۳۴ را ببینید. حداکثر تا ۲۵۶ کامپیوتر را می شد به این کابل متصل کرد. به چنین کابلی، که چندین ماشین بصورت موازی به آن متصل شده اند، کابل چنداتصال (multidrop cable) گفته می شود. اترنت با سرعت 2.94 Mbps کار می کرد. اترنت یکی از اشکالات عمده آلو هانت را نیز برطرف کرده بود: هر کامپیوتر قبل از ارسال بسته خود ابتدا به کابل گوش می کرد، تا مطمئن شود کس دیگری در همان لحظه در حال استفاده از آن نیست. اگر چنین بود، کامپیوتر تا خالی شدن خط کار خود را عقب می انداخت. بدین ترتیب، اترنت توانست با اجتناب از تداخل های بی مورد به کارایی بالاتری دست یابد. البته آلو هانت امکان انجام چنین کاری را نداشت، چون فرستنده یک جزیره نمی توانست امواج فرستنده های جزایر دیگر را بشنود.

علیرغم اینکه کامپیوترهای اترنت تا وقتی خط خالی نباشد، اقدام به ارسال نمی کنند، اما مسئله دیگری ممکنست بروز کند: اگر چند کامپیوتر همزمان منتظر خالی شدن خط باشند، و به محض اینکه خط خالی شد، در یک لحظه شروع به ارسال اطلاعات خود کنند، چه خواهد شد؟ راه حل این مشکل آن است که هر کامپیوتر در تمام لحظات ارسال اطلاعات خود به خط گوش کند، و اگر متوجه تداخل امواج شد، ابتدا به دیگران اخطار می فرستد، و سپس برای مدتی کوتاه (که مقدار آنرا بطور تصادفی تعیین می کند) کنار می کشد، و بعد از این مدت دوباره سعی

می‌کند؛ اگر در مرتبه بعد باز هم تداخل پیش آمد، مدت انتظار را دو برابر می‌کند، تا بالاخره یکی از آنها فرصت ارسال امواج را پیدا کند.

ایترنت زیراکس چنان موفق بود که در سال ۱۹۷۸ شرکت‌های DEC، ایتل و زیراکس استاندارد بنام DIX برای اینترنت 10-Mbps وضع کردند. استاندارد DIX در سال ۱۹۸۳ با دو تغییر جزئی به استاندارد IEEE 802.3 تبدیل شد.

متأسفانه زیراکس همواره یکی از شرکت‌هایی بوده است که اختراعات بسیار مهمی (مانند PC، اینترنت و مایکروسافت) ز آنجا منشأ گرفته، ولی نتوانسته اقدامی برای تجاری کردن آنها بعمل آورد. وقتی زیراکس علاقه چندانی به اینترنت نشان نداد (و فقط در استاندارد کردن آن همکاری کرد)، باب متکالف شرکتی بنام 3Com تأسیس کرد تا بتواند برای PC ها کارت شبکه اینترنت تولید کند.

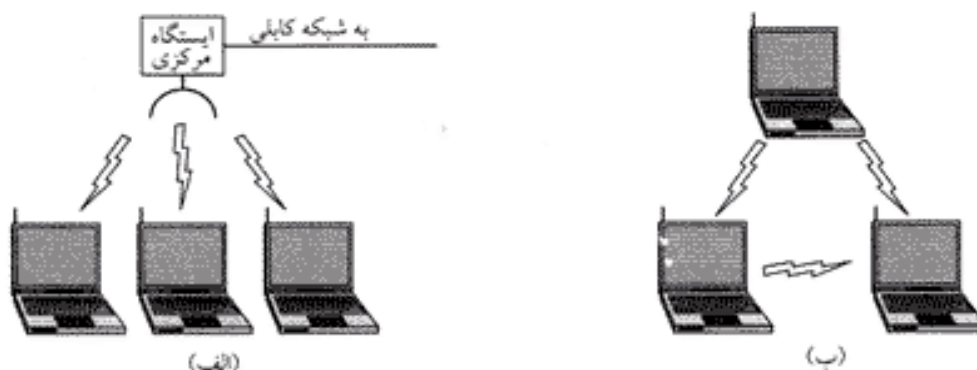
ایترنت به رشد و توسعه خود ادامه داد (رشدی که همچنان ادامه دارد)، و امروزه به سرعت‌های 100-Mbps و 1000-Mbps دست یافته است - سرعتی که انتظار می‌رود باز هم افزایش یابد. کابل کشی، سوئیچینگ و سایر جنبه‌های اینترنت نیز بهبود یافته، و ویژگی‌های جدیدی به آن اضافه شده است. در فصل ۴ مفصلاً درباره اینترنت صحبت خواهیم کرد.

البته همین جا لازم به توضیح است که اینترنت (IEEE 802.3) تنها استاندارد LAN نیست. استانداردهای خط توکین (IEEE 802.4 - Token Bus) و حلقه توکین (IEEE 802.5 - Token Ring) نیز از جمله استانداردهای معروف LAN هستند. وجود سه استاندارد کمابیش ناسازگار برای شبکه‌های محلی بیشتر از اینکه مسئله‌ای فنی باشد، موضوعی سیاسی است. در همان زمان که اینترنت در حال استاندارد شدن بود، جنرال موتورز نیز در کار ایجاد شبکه‌ای بود که از همان توپولوژی اینترنت (یعنی کابل خطی) استفاده می‌کرد، ولی نوبت ارسال هر کامپیوتر با استفاده از بسته خاصی بنام توکین، که بین کامپیوترها دست به دست می‌گشت، تعیین می‌شد. هر کامپیوتر فقط زمانی می‌توانست اقدام به ارسال اطلاعات کند که توکین را در اختیار داشته باشد، و بدین ترتیب مشکل تداخل حل می‌شد. جنرال موتورز اعلام کرد که این تکنیک برای خط تولید کارخانجات اتومبیل‌سازی ضرورت مطلق دارد، و حاضر نبود حتی یک میلیمتر از موضع خود عقب‌نشینی کند. با وجود چنین ادعایی، 802.4 اکنون از صفحه روزگار محو شده است.

غول صنعت کامپیوتر، IBM هم سوگلی خود را داشت: حلقه توکین. تنها تفاوت این سیستم با شبکه جنرال موتورز آن بود که در اینجا کابل شبکه یک مسیر بسته (حلقه) را تشکیل می‌داد. برخلاف 802.4، 802.5 هنوز در برخی از سایت‌های IBM مورد استفاده است (ولی خارج از IBM هیچکس از آن استفاده نمی‌کند). تحقیقاتی در زمینه نسل جدید و پرسرعت حلقه توکین با سرعت گیگابیت (802.5v) در جریان است، ولی بنظر نمی‌رسد بتواند با اینترنت رقابت کند. خلاصه اینکه، در جنگی که بین اینترنت، خط توکین و حلقه توکین در گرفت، اینترنت پیروز شد، چون اولین و در ضمن از رقبایش بهتر بود.

۵-۱ شبکه‌های محلی بیسیم: 802.11

تقریباً همزمان با به بازار آمدن کامپیوترهای کتابی، بسیاری افراد این رؤیا را در سر می‌پروراندند که بتوانند به محض ورود به جایی که دسترسی اینترنت وجود دارد، بلافاصله و بنحوی جادویی کامپیوترشان به اینترنت متصل شود - و همیشه وقتی رؤیایی وجود دارد، افرادی هم هستند که به فکر محقق کردن آن بیفتند. عملی‌ترین رهیافتی که برای به فعل در آوردن این ایده وجود داشت، مجهز کردن کامپیوترها به فرستنده-گیرنده‌های رادیویی بُرد کوتاه بود - از همین جا بود که شرکت‌های متعددی سرعت شبکه‌های محلی بیسیم را وارد بازار کردند.



شکل ۱-۳۵. شبکه بیسیم (الف) با ایستگاه مرکزی، و (ب) بدون ایستگاه مرکزی.

مشکل اصلی این بود که هیچکدام از این شبکه های بیسیم با هم سازگار نبودند، و کامپیوترهایی که به بیسیم های مختلف مجهز بودند، نمی توانستند با هم ارتباط برقرار کنند. بالاخره همه به این نتیجه رسیدند که استاندارد کردن شبکه های بیسیم می تواند ایده خوبی باشد، و بدنبال آن کمیته استاندارد IEEE مأمور تدوین این استاندارد شد - استاندارد 802.11 نام گرفت، و در میان عموم به WiFi معروف است. این یکی از استانداردهای مهم صنعت کامپیوتر است، و شایسته توجه کافی.

استاندارد پیشنهاد شده باید در دو حالت کار می کرد:

۱. در شرایط وجود یک ایستگاه مرکزی
۲. در شرایط فقدان ایستگاه مرکزی

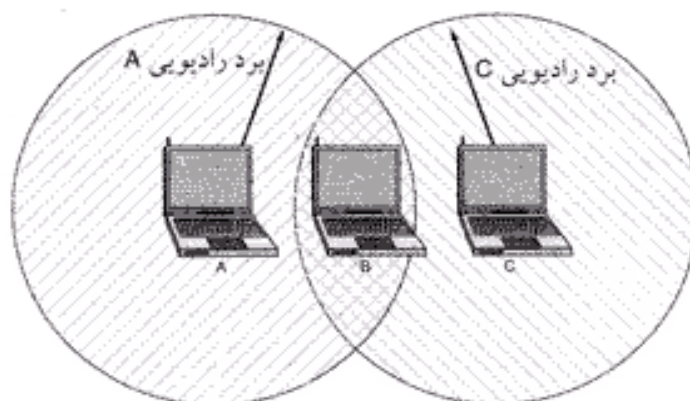
در حالت اول، تمام پیامها باید از طریق ایستگاه مرکزی، که در استاندارد 802.11 به آن نقطه دسترسی (access point) گفته می شود، مبادله شوند. اما در حالت دوم، کامپیوترها مستقیماً با یکدیگر ارتباط برقرار می کنند. این دو حالت را در شکل ۱-۳۵ مشاهده می کنید.

شروع می شود، و اعداد 1 تا 10 قبلاً استفاده شده بود، نام این استاندارد 802.11 شد. ولی بقیه کار به همین سادگی نبود.

برخی از مهمترین چالشهایی که کمیته تدوین استاندارد با آنها روبرو بود، عبارت بودند از: انتخاب یک باند فرکانسی مناسب (که ترجیحاً بین المللی نیز باشد)؛ محدود بودن بُرد سیگنالهای رادیویی؛ تأمین ایمنی مناسب؛ عمر محدود باتری در کامپیوترهای کتابی؛ مسائل بهداشتی (هنوز این مسئله بدرستی روشن نشده که آیا امواج رادیویی سرطانزا هستند یا خیر)؛ پیامدهای متحرک بودن کامپیوترها؛ و بالاخره، ایجاد سیستمی که از نظر پهنای باند ارزش اقتصادی داشته باشد.

با توجه به غالب بودن اینترنت در زمان تدوین این استاندارد، کمیته تصمیم گرفت که 802.11 باید در لایه های بالاتر از لایه پیوند داده با اینترنت سازگار باشد. بویژه، ارسال بسته های IP در شبکه های بیسیم بایستی دقیقاً به همان روش اینترنت باشد.

با این وجود، لایه های فیزیکی و لینک داده در این دو سیستم تفاوت های اساسی با هم دارند. اول اینکه، در اینترنت هر کامپیوتر قبل از شروع به ارسال اطلاعات به اتر (کابل شبکه) گوش می کند، و فقط در صورت خالی بودن آن شروع به ارسال می کند. در شبکه های بیسیم کار به همین سادگی نیست. برای درک علت آن، به شکل ۱-۳۶ نگاه کنید. فرض کنید کامپیوتر A در حال ارسال اطلاعات به کامپیوتر B است، ولی بُرد امواج آن به کامپیوتر C نمی رسد. اگر در این لحظه C بخواهد چیزی به B بفرستد، باید به اتر (در اینجا، فضا) گوش کند، ولی آیا عدم



شکل ۱-۳۶. گاهی بُرد امواج رادیویی برای پوشش دادن به تمام شبکه کافی نیست.

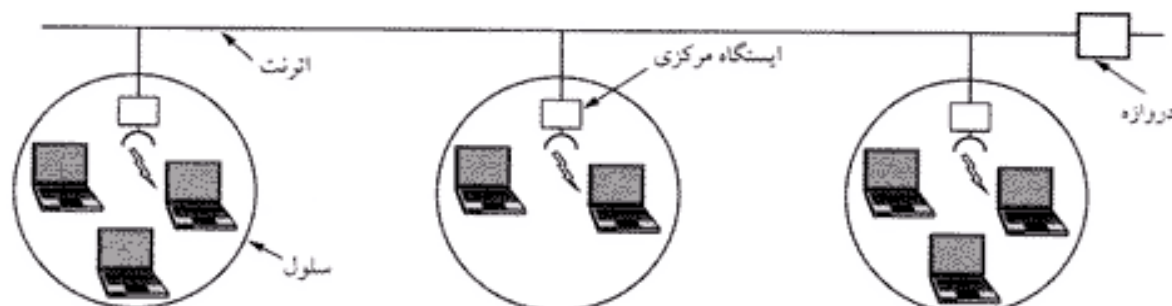
در یافت امواج به معنای آن است که می تواند با اطمینان شروع به ارسال کند؛ مسلماً خیر. استاندارد 802.11 باید این مسئله را حل می کرد.

مسئله دومی که باید حل می شد این بود که اجسام سخت امواج رادیویی را منعکس می کنند، و یک موج می تواند چندین بار (و از مسیرهای مختلف) به گیرنده برسد. این تداخل امواج باعث بروز حالتی می شود که به آن محوشدگی چندمسیره (multipath fading) می گویند.

مسئله سوم این است که بسیاری از نرم افزارهای موجود اساساً با چنین وضعیتی (متحرک بودن کامپیوتر) آشنا نیستند. برای مثال، بسیاری از برنامه ها دارای لیستی از چاپگرها هستند که می توانند روی آنها چاپ کنند. وقتی یکی از این برنامه ها (که روی یک کامپیوتر کتابی نصب شده) وارد محیط جدیدی می شود، نمی تواند تشخیص دهد که فهرست قبلی چاپگرها دیگر در این وضعیت اعتبار ندارد.

مسئله چهارم این است که وقتی یک کامپیوتر از بُرد یک ایستگاه مرکزی خارج و وارد محدوده ایستگاه دیگری می شود، باید مکانیزمی برای این جابجایی وجود داشته باشد. با اینکه تلفنهای همراه دارای چنین مکانیزمی هستند، اما این اتفاق در اینترنت نمی افتد، و باید بطریقی حل شود. یکی از راه حل های این مسئله، متصل کردن ایستگاههای مرکزی به یکدیگر از طریق کابل است (شکل ۱-۳۷ را ببینید). از دید دنیای خارج، این شبکه کاملاً شبیه یک شبکه اینترنت واحد است. نقطه اتصال سیستم 802.11 با دنیای خارج را درگاه (portal) می گویند.

بعد از مدتی کار سخت، کمیته موفق به تدوین استاندارد شد که این مسائل (و بسیاری مسائل دیگر) در آن حل شده بود. این شبکه بیسیم با سرعت های 1-Mbps و 2-Mbps کار می کرد. تقریباً بلافاصله، همه لب به شکایت گشودند که این سرعتها بسیار کم است، و کار بر روی استانداردهای سریعتر آغاز شد. در همین زمان کمیته استاندارد به دو قسمت تقسیم شد، که در سال ۱۹۹۹ هر کدام استاندارد جداگانه ای وضع کردند. استاندارد



شکل ۱-۳۷. یک شبکه 802.11 چندسلولی.

802.11a از باند فرکانسی وسیعتری نسبت به 802.11 استفاده می کند، و سرعت آن به 54-Mbps می رسد. استاندارد 802.11b در همان باند فرکانسی 802.11 کار می کند، ولی با استفاده از مدولاسیون متفاوت به سرعت 11-Mbps دست می یابد. همانطور که می بینید، هر دوی این استانداردها از 802.11 (و حتی از اینترنت اولیه) سریعترند، ولی هنوز بدرستی معلوم نیست که کدامیک از آنها برنده نهایی این مسابقه اند. برای اینکه اوضاع از این هم پیچیده تر شود، کمیته 802.11 استاندارد جدیدی بنام 802.11g تدوین کرده، که در باند فرکانسی 802.11b، ولی با مدولاسیون 802.11a، کار می کند. در فصل ۴ مفصلاً درباره 802.11 صحبت خواهیم کرد.

شکی نیست که 802.11 انقلاب جدیدی در دنیای کامپیوتر و اینترنت بها کرده است. فرودگاهها، ایستگاههای قطار، بنادر، هتلها، فروشگاهها، دانشگاهها، مراکز آموزشی (و حتی کافه های کوچک) بسرعت در حال نصب شبکه های 802.11 هستند. در واقع، 802.11 همان چیزی را به اینترنت داده است، که کامپیوترهای کتابی به دنیای کامپیوتر دادند: تحرک پذیری.

۶-۱ استانداردهای شبکه

تعداد زیادی سازنده و تأمین کننده قطعات و تجهیزات شبکه وجود دارد، که فکر می کنند می دانند چگونه باید کار خود را انجام دهند. اما بدون یک عامل هماهنگ کننده، این وضعیت می تواند به یک آشوب واقعی بینجامد، و در نهایت هیچ کاری هم انجام نشود. تنها راه برای خلاصی از چنین وضعیتی، توافق بر سر استانداردهای شبکه است. استانداردها نه تنها اجازه می دهد تا تجهیزات مختلف بتوانند با هم کار کنند، بلکه فروش محصولات متعلق با استاندارد را نیز افزایش می دهند - و فروش بیشتر یعنی تولید انبوه، کاهش هزینه ها، طراحی بهتر، کاهش قیمت ها، و افزایش مجدد درخواست (و مگر اقتصاد چیزی غیر از این است). در این قسمت نگاهی به استانداردهای بین المللی (که از اهمیت زیادی برخوردارند، ولی کمتر شناخته شده اند) خواهیم داشت.

استانداردها بر دو نوعند: استانداردهای بالفعل (de facto)، و استانداردهای قانونی (de jure). استانداردهای بالفعل آنهایی هستند که بدون هیچ طرح رسمی بوجود آمده و پذیرفته شده اند. کامپیوترهای سازگار با IBM PC (که به اختصار PC خوانده می شوند) از جمله استانداردهای بالفعل هستند، چون شرکتهای بسیاری تصمیم گرفتند تا کپی های دقیق و کاملی از این نوع کامپیوتر بسازند. یکی دیگر از استانداردهای بالفعل، سیستم عامل یونیکس (UNIX) است، که در دانشکده های کامپیوتر بعنوان سیستم عامل استاندارد پذیرفته شده است. از طرف دیگر، استانداردهای قانونی آنهایی هستند که توسط مراجع مسئول بین المللی پذیرفته شده اند. مراجع بین المللی استاندارد به دو دسته تقسیم می شوند: آنهایی که طبق معاهدات بین المللی تأسیس شده اند، و آنهایی که بصورت داوطلبانه شکل گرفته اند. در زمینه استانداردهای شبکه های کامپیوتری، سازمانهایی از هر دو دسته وجود دارند، که در زیر آنها را معرفی خواهیم کرد.

۱-۶-۱ مراجع مسئول استانداردهای مخابرات

وضعیت قانونی شرکتهای تلفن از کشوری به کشور دیگر بطرز چشمگیری متفاوت است. در یک سو ایالات متحده آمریکا قرار دارد، که در آن متجاوز از ۱۵۰۰ شرکت خصوصی در این زمینه مشغول به کار هستند. شرکت AT&T، قبل از آنکه در سال ۱۹۸۴ طبق قانون ضد تراست به دو بخش تقسیم شود، بزرگترین شرکت خصوصی دنیا بود، و مخابرات این کشور را تحت سلطه کامل خود داشت. این شرکت سرویس تلفن ۸۰ درصد مناطق آمریکا را تأمین می کرد، در حالیکه بقیه شرکتهای تنها ۲۰ درصد بازار (آن هم اغلب در مناطق روستایی) را در اختیار داشتند. بعد از تقسیم AT&T، این شرکت اجازه یافت تا فقط در مخابرات راه دور فعالیت کند (آن هم در رقابت با

سایر شرکتها). هفت شرکت منطقه ای بل (Bell)، که از AT&T منشعب و با شرکتهای دیگر متحد شده بودند، نیز خدمات تلفن شهری و تلفن همراه را بر عهده گرفتند. بدلیل ادغامها و انشعابات متعدد، صنعت مخابرات در ایالات متحده اکنون از وضعیت ثابتی برخوردار است.

در ایالات متحده، به شرکتهایی که سرویسهای مخابراتی در اختیار عموم قرار می دهند، کاریر عمومی گفته می شود. نوع سرویسها و تعرفه خدمات این شرکتها توسط کمیسیون فدرال مخابرات (برای تماسهای بین ایالتی و بین المللی) و کمیسیونهای ایالتی (برای تماسهای داخل ایالتی) تعیین می شود.

در سوی دیگر، کشورهایی قرار دارند که در آنها تمامی سرویسهای مخابراتی (از جمله، پست، تلگراف، تلفن، و حتی رادیو و تلویزیون) تحت انحصار مطلق دولت قرار دارد. اکثر کشورهای دنیا هم از این دسته اند. در برخی از این کشورها، مخابرات را یک شرکت ملی اداره می کند، و در برخی دیگر دولت مستقیماً (از طریق وزارتخانه ای بنام پ.ت.ت: پست-تلگراف-تلفن) کارها را در دست دارد. اما، در کل دنیا حرکت یکپارچه ای به سمت آزادسازی، رقابت و حذف انحصار دولت آغاز شده است. امروزه اکثر کشورهای اروپایی مخابرات خود را به بخش خصوصی سپرده اند، و در دیگر کشورها این فرآیند (هر چند کند و بطنی) ادامه دارد.

با این همه شرکت و سرویسهای مخابراتی، وجود نوعی استاندارد بین المللی برای تضمین ارتباط بین افراد (و کامپیوترها) از کشوری به کشور دیگر ضروری بنظر می رسد. البته این نیاز از مدتها قبل آشکار شده بود: در سال ۱۸۶۵، نمایندگانی از کشورهای مختلف اروپایی آنچه را که امروز ITU (اتحادیه بین المللی مخابرات - International Telecommunication Union) نامیده می شود، بنا نهادند. وظیفه این اتحادیه استاندارد کردن مخابرات بین المللی (که در آن زمان فقط شامل تلگراف می شد) بود. حتی در آن زمان نیز مشخص بود که اگر کلیه کشورها در مورد استفاده از یک کد یکسان (که همان کد مورس بود) به توافق نرسند، مشکلات عدیده ای بروز خواهد کرد. با ورود تلفن به صحنه مخابرات بین المللی، ITU وظیفه استاندارد کردن آنرا نیز بر عهده گرفت. در سال ۱۹۴۷، ITU بصورت یکی از سازمانهای تابعه سازمان ملل متحد در آمد. اتحادیه بین المللی مخابرات (ITU) سه بخش عمده دارد:

۱. بخش مخابرات رادیویی (ITU-R)
۲. بخش تدوین استانداردهای مخابراتی (ITU-T)
۳. بخش توسعه (ITU-D)

وظیفه ITU-R تخصیص فرکانسهای رادیویی به متقاضیان در سراسر دنیاست. بخش ITU-T (که بیشتر مدنظر ماست) وظیفه تدوین استاندارد برای سیستمهای مخابراتی (تلفن و داده) را بر عهده دارد. از سال ۱۹۵۳ تا ۱۹۹۳، ITU-T با نام CCITT (که از نام فرانسوی آن - Comité Consultatif International Télégraphique et Téléphonique - گرفته شده بود) شناخته می شد. در اول مارس ۱۹۹۳، CCITT برای کاهش بوروکراسی ساختار اداری اش را تغییر داد، و نام خود را نیز ITU-T گذاشت. ITU-T و CCITT هر دو توصیه هایی را در زمینه مخابرات تلفنی و داده منتشر می کردند. امروزه نیز افراد بسیاری به توصیه های CCITT مراجعه می کنند، اگر چه از سال ۱۹۹۳ این توصیه ها نام ITU-T را بر خود دارند. اعضای ITU-T به چهار دسته تقسیم می شوند:

۱. دولتها
۲. اعضای بخش
۳. اعضای وابسته
۴. نمایندگیها

تقریباً تمام کشورهای عضو سازمان ملل متحد (یعنی حدود ۲۰۰ کشور) در ITU-T عضویت دارند. از آنجائیکه ایالات متحده آمریکا وزارتخانه ای بنام «پست و تلگراف و تلفن» ندارد، فرد دیگری باید نمایندگی آنرا در ITU-T بر عهده بگیرد، که این وظیفه به وزارت امور خارجه محول شده است (شاید به این دلیل که با کشورهای خارجی سروکار دارد). تعداد اعضای بخش در ITU-T به حدود ۵۰۰ می رسد، که شرکتهای تلفن (مانند AT&T، ودفون، ورلدکام)، تولیدکنندگان تجهیزات مخابراتی (مانند سیسکو، نوکیا، نورتل)، تولیدکنندگان کامپیوتر (مانند کامپک، سان، توشیبا)، سازندگان چیپ های میکروالکترونیک (مانند اینتل، موتورولا، IT)، شرکتهای رسانه ای (مانند CBS، AOL Time Warner، سونی)، و سایر شرکتهای علاقمند (مانند بوئینگ، سامسونگ، زیراکس) از آن جمله اند. تعدادی از سازمانهای علمی غیرانتفاعی و کنسرسیوم های صنعتی (مانند IFIP و IATA) نیز در ITU-T عضو بخش هستند. اعضای وابسته شرکتهای کوچکتري هستند، که به یکی از مباحث خاص ITU-T علاقمند هستند. نمایندگی ها نیز آنهایی هستند که بر امور مخابراتی نظارت دارند، مانند کمیسیون مخابرات فدرال ایالات متحده (FCC).

وظیفه ITU-T ارائه توصیه های فنی در زمینه تلفن، تلگراف و مخابرات داده است. این توصیه ها اغلب بصورت استانداردهای جهانی پذیرفته می شوند، مانند V.24 (در ایالات متحده به نام EIA RS-232 نیز شناخته می شود) که نقش و وظیفه پایه های رابط های مودمها و ترمینالهای آسنکرون را مشخص می کند.

این نکته را باید تذکر داد که ITU-T فقط توصیه های فنی ارائه می کند، و دولتها می توانند آنها را بپذیرند یا نپذیرند (همانطور که می دانید دولتها مثل بچه های ده-یازده ساله هستند، یعنی دوست ندارند کسی به آنها دستور بدهد). البته کشوری که استاندارد غیر از استاندارد سایر کشورها را قبول کند، در عمل خود را از سایر کشورهای دنیا ایزوله کرده است - کاری که شاید فقط کره شمالی مایل به انجام آن باشد. احتمالاً نامگذاری استانداردهای ITU-T بنام «توصیه» فقط کلکی برای آرام کردن ملی گرایان کشورهای مختلف است.

کار واقعی ITU-T توسط ۱۴ گروه مطالعاتی (که اغلب آنها نزدیک به ۴۰۰ نفر عضو دارند) انجام می شود. سرفصل هایی که این گروه های مطالعاتی بررسی می کنند، از استانداردهای تهیه صورتحساب تلفن گرفته تا سرویسهای چندرسانه ای متغیر است. برای اینکه هر گروه بتواند کار خود را بهتر انجام دهد، به چند دسته کاری تقسیم می شود، که هر یک از این دسته ها بنوبه خود به چند تیم کارشناسی، و هر تیم تخصصی نیز به گروه های تخصصی تقسیم می شوند. (مثل اینکه هیچ وقت نمی توان از بوروکراسی خلاص شد!)

علیرغم این بوروکراسی، ITU-T کار خود را بخوبی انجام می دهد، و از اول تأسیس آن تاکنون نزدیک به ۳۰۰۰ توصیه (که بالغ بر ۶۰،۰۰۰ صفحه می شود) تدوین کرده است، که بسیاری از آنها بطور گسترده ای مورد استفاده قرار گرفته اند. برای مثال، استاندارد V.90 (که درباره مودمهای 56-kbps است) از توصیه های ITU-T می باشد.

با رشد و توسعه روزافزون مخابرات راه دور و جهانی شدن آن، استانداردها روز به روز اهمیت بیشتری می یابند، و سازمانهای بیشتری مایلند در تدوین آنها شرکت داشته باشند. برای کسب اطلاعات بیشتر درباره ITU (Irmer, 1994) مراجعه کنید.

۲-۶-۱ مراجع مسئول استانداردهای بین المللی

استانداردهای بین المللی توسط سازمان بین المللی استاندارد (ISO) - که البته نام واقعی آن سازمان بین المللی برای تدوین استاندارد، IOS است - تهیه و منتشر می شوند. ISO یک سازمان غیرپیمانی داوطلبانه است، در سال ۱۹۴۶ تأسیس شده، و دارای ۸۹ عضو می باشد. سازمانهای استاندارد ایالات متحده آمریکا (ANSI)، انگلستان (BSI)، فرانسه (AFNOR)، آلمان (DIN) و هشتاد و پنج کشور دیگر از اعضای ISO هستند.

تنوع استانداردهایی که ISO منتشر می‌کند، واقعاً جالب و حیرت‌آور است - استانداردهایی وجود دارد که کسی حتی فکر استاندارد بودن آنها را نمی‌کند، مانند استاندارد دانه‌های کاکائو (ISO 2451)، تورهای ماهیگیری (ISO 1530)، لباسهای زیر زنانه (ISO 4416) و غیره. تعداد استانداردهای ISO بالغ بر ۱۳,۰۰۰ است (که استانداردهای OSI نیز از آن جمله‌اند). ISO نزدیک به ۲۰۰ کمیته فنی (TC) دارد، که به ترتیب زمان ایجاد شماره‌گذاری شده‌اند و هر کدام در موضوعی خاص تخصص دارند. برای مثال، TC1 با استانداردهای پیچ و مهره سروکار دارد، و TC97 با استانداردهای صنعت کامپیوتر و پردازش اطلاعات. هر TC به چند زیرکمیته (SC)، و هر زیرکمیته به چند گروه کاری (WG) تقسیم می‌شود.

کار اصلی ISO در WG ها (که متجاوز از ۱۰۰,۰۰۰ عضو داوطلب در سراسر دنیا دارند) انجام می‌شود. بسیاری از این «داوطلبان» در استخدام شرکت‌هایی هستند که محصول آنها قرار است استاندارد شود؛ برخی دیگر نیز مقامات رسمی کشورهای هستند که مایلند روش ساخت محصولات کشورشان بصورت استانداردهای جهانی در آید. در برخی از WG ها مقامات متخصصان دانشگاهی نیز حضور دارند.

در زمینه استانداردهای صنعت مخابرات، ISO و ITU-T اغلب با یکدیگر تشریک مساعی دارند (در واقع، یکی از اعضای ITU-T است)، تا از توسعه استانداردهای ناسازگار اجتناب شود.

نماینده ایالات متحده آمریکا در ISO مؤسسه ملی استانداردهای آمریکا (ANSI) است، که برخلاف نامش یک مؤسسه غیردولتی و غیرانتفاعی است، و اعضای آن عبارتند از تولیدکنندگان، کاربرهای عمومی و شرکت‌های ذینفع. ISO اغلباً استانداردهایی که توسط ANSI وضع می‌شود، را بعنوان استانداردهای بین‌المللی می‌پذیرد. رویه‌ای که در ISO برای پذیرش استانداردها مورد استفاده قرار می‌گیرد، بگونه‌ایست که بیشترین توافق اعضا را بدنبال داشته باشد. این رویه با اعلام نیاز یکی از سازمانهای عضو به یک استاندارد جدید آغاز می‌شود. بدنبال آن یک گروه کاری تشکیل می‌شود، تا برای استاندارد جدید یک پیش‌نویس کمیته (CD) تهیه کند. این CD بین سازمانهای عضو توزیع می‌شود، و آنها شش ماه فرصت دارند تا انتقادات خود را مطرح کنند. اگر اکثریت اعضا با این CD موافق باشند، یک سند اصلاح شده بنام پیش‌نویس استاندارد جهانی (DIS) برای اظهارنظر و رأی‌گیری منتشر می‌شود. بر اساس نتایج بدست آمده از این دور، متن نهایی بعنوان استاندارد جهانی (IS) تنظیم، اصلاح و منتشر می‌شود. در مورد استانداردهایی که اختلاف نظر جدی بین اعضا وجود داشته باشد، امکان دارد CD یا DIS چندین بار اصلاح و به رأی گذاشته شود، تا بتواند رأی کافی بدست آورد، و این فرآیندی است که ممکنست سالها طول بکشد.

در ایالات متحده آمریکا، مؤسسه ملی استانداردها و تکنولوژی (National Institute of Standards and Technology - NIST) که از توابع وزارت بازرگانی است، سازمان ملی استاندارد محسوب می‌شود، و تمام خریدهای دولتی (به استثنای خریدهای نظامی، که استانداردهای خاص خود را دارند) بایستی مطابق استانداردهای آن باشد.

یکی دیگر از بازیگران بزرگ در صحنه استانداردهای جهانی، مؤسسه مهندسان برق و الکترونیک (IEEE) است، که بزرگترین سازمان حرفه‌ای دنیا محسوب می‌شود. علاوه بر انتشار کتب و مجلات متعدد و برگزاری صدها کنفرانس علمی در سال، IEEE یک گروه تدوین استاندارد نیز دارد که در زمینه مهندسی برق و کامپیوتر فعالیت می‌کند. برای مثال، کمیته 802 وظیفه تدوین استاندارد شبکه‌های LAN را در IEEE بر عهده دارد، که گروههای کاری آنها در شکل ۱-۳۸ ملاحظه می‌کنید. میزان موفقیت گروههای کاری 802 چندان بالا نیست، و داشتن عدد 802.x تضمینی برای موفقیت یک استاندارد نیست. البته استثناهایی هم در این زمینه وجود دارد، که استانداردهای 802.3 و 802.11 از آن جمله‌اند.

شماره	سرفصل
802.1	معماری LAN
802.2 ↓	کنترل لینک منطقی
802.3 *	اترنت
802.4 ↓	باس توکن
802.5	حلقه توکن
802.6 ↓	دو باس-دو صف
802.7 ↓	گروه مشورتی تکنولوژی های پخش
802.8 ↑	گروه مشورتی تکنولوژی های فیبرنوری
802.9 ↓	LAN ایزو سنکرون (برای کاربردهای زمان واقعی)
802.10 ↓	شبکه مجازی و امنیتی
802.11 *	LAN بی سیم
802.12 ↓	تقدم تقاضا (خاص)
802.13	عدد نحس! (کسی آنرا نمی خواهد)
802.14 ↓	مودم کابلی
802.15 *	شبکه های شخصی (بلوتوث)
802.16 *	بی سیم باند وسیع
802.17	حلقه بسته برگشتی

شکل ۱-۳۸. گروه های کاری 802. گروه های مهم با * مشخص شده اند. آنهایی که با ↓ مشخص شده اند، به خواب زمستانی رفته اند، و گروه هایی که با + مشخص شده اند، مدت ها است از صفحه روزگار محو شده اند.

۱-۶-۳ مراجع مسئول استانداردهای اینترنت

اینترنت نیز مکانیزم های استاندارد سازی خاص خود را دارد، که با ISO و ITU-T بسیار متفاوت است. مهمترین تفاوت آنها اینست که افرادی که در گردهمایی های ISO و ITU-T شرکت می کنند، کت و شلوار رسمی می پوشند، ولی شرکت کنندگان در نشست های استاندارد سازی اینترنت لباس جین به تن می کنند (البته اگر تابستان نباشد، و محل اجلاس هم کنار دریا نباشد).

در گردهمایی های ISO و ITU-T اغلب نمایندگان شرکتها و دولتها (که استاندارد سازی شغل آنهاست) شرکت می کنند. آنها به استاندارد بعنوان «یک چیز آسمانی» نگاه می کنند، و حاضرند از جانشان برای آن مایه بگذارند. از طرف دیگر، اینترنتی ها بی نظمی را به عنوان یک اصل پذیرفته اند. اما از آنجائیکه بدون حداقلی از اشتراکات اصولاً امکان برقراری هیچ نوع تماسی وجود ندارد، استاندارد را (در کمال تأسف) چیزی لازم می دانند. وقتی آرپانت بوجود آمد، وزارت دفاع ایالات متحده آمریکا کمیته ای رسمی را مأمور نظارت بر آن کرد. در سال ۱۹۸۳، این کمیته به هیئت نظارت بر فعالیتهای اینترنتی (Internet Activity Board - IAB) تغییر نام داد، و وظیفه همسو نگه داشتن آرپانت و اینترنت بر عهده آن گذاشته شد (کاری که می توان آنرا به چوپانی یک گله گربه تشبیه کرد!). بعدها نام این کمیته به هیئت مدیره معماری اینترنت (Internet Architecture Board) تغییر کرد، بگونه ای که حروف اختصاری آن همچنان IAB باقی ماند.

به هر یک از نزدیک به ده عضو IAB موضوع مهمی برای دنبال کردن محول شد. این اعضا سالی چند بار با هم ملاقات و تبادل نظر می کردند، و گزارش کار خود را به NFS و وزارت دفاع (که پشتیبانی مالی آنرا بر عهده داشتند) ارائه می کردند. وقتی نیاز به تدوین استاندارد جدیدی حس می شد، اعضای IAB آنرا با جار و جنجال زیاد اعلام

می کردند، تا دانشجویانی که موتور نرم افزاری اینترنت محسوب می شدند، بتوانند آنرا پیاده سازی کنند. تبادل اطلاعات فنی توسط مقالات و گزارشهایی بنام نظرخواهی (Request For Comment - RFC) انجام می شد. این RFC ها (که به ترتیب انتشار شماره گذاری می شوند) در سایت www.ietf.org/rfc نگهداری می شوند، تا همه کسانی که مایلند بتوانند به آنها دسترسی داشته باشند. هم اکنون متجاوز از ۳۰۰۰ RFC در این سایت موجود است، که در این کتاب به بسیاری از آنها اشاره خواهیم کرد.

تا سال ۱۹۸۹ اینترنت چنان گسترش یافته بود، که این روش نسبتاً رسمی دیگر نمی توانست بکار آید. در آن زمان شرکتهای بسیاری محصولات TCP/IP خود را وارد بازار کرده بودند، و نمی خواستند آنها را بصرف اینکه چند پژوهشگر به خیال خود ایده های بهتری دارند، عوض کنند. در تابستان ۱۹۸۹، ساختار سازمانی IAB تغییر کرد، و به دو بخش نیروی پژوهشی اینترنت (Internet Research Task Force - IRTF) و نیروی مهندسی اینترنت (Internet Engineering Task Force - IETF) تقسیم شد، و نیروهای جدیدی (غیر از مجامع تحقیقاتی) وارد آن شدند. در سالهای اول این سازمان ها دارای گردش بسته بودند: هر عضو برای دو سال انتخاب می شد، و اعضای جدید فقط توسط اعضای قدیمی منسوب می شدند. بعدها، انجمن اینترنت (Internet Society) بوجود آمد، که تمامی علاقمندان اینترنت می توانستند در آن عضویت یابند. انجمن اینترنت از جهاتی شبیه IEEE یا ACM است، و توسط یک هیئت امنای انتخابی، که اعضای IAB را منسوب می کند، اداره می شود. هدف از تقسیم IAB آن بود که IRTF بتواند به تحقیقات بلند مدت مشغول شود، در حالیکه IETF به کارهای مهندسی کوتاه مدت اشتغال دارد. IETF خود به چند گروه کاری تقسیم می شد، که هر یک روی موضوعی خاص کار می کردند. به منظور هماهنگ کردن فعالیتهای این گروه ها، مدیران گروه های کاری نشستهای متعددی برگزار می کردند. سرفصلهایی که گروه های کاری IETF روی آنها کار می کردند، عبارت بودند از: برنامه های کاربردی جدید، گردآوری اطلاعات کاربران، یکپارچه سازی OSI، هدایت و آدرس دهی، امنیت، مدیریت شبکه، و استاندارد سازی. تعداد این گروه های کاری بتدریج افزایش یافت، و اکنون متجاوز از ۷۰ گروه کاری ذیل IETF مشغول به کار هستند.

علاوه بر آن، فرآیند استاندارد سازی رسمی تری (که شبیه ISO بود) پذیرفته شد. برای آن که یک ایده جدید به صورت استاندارد پیشنهادی (Proposed Standard) در آید، بایستی بطور کامل در یک RFC تشریح شود، و در جامعه اینترنت نیز علاقه کافی نسبت به آن وجود داشته باشد. سپس، برای آنکه این استاندارد به مرحله پیش نویس استاندارد (Draft Standard) ارتقاء یابد، بایستی بصورت واقعی پیاده سازی شده و به مدت حداقل ۴ ماه در ۲ سایت اینترنتی بطور همه جانبه تست شود. اگر IAB متقاعد شود که ایده اصلی خوب است و نرم افزار نیز بخوبی کار می کند، می تواند این RFC را رسماً بعنوان یک استاندارد اینترنتی اعلام کند. برخی از این استانداردها حتی بصورت استانداردهای نظامی (MIL-STD) در آمده اند، و رعایت آنها در محصولات که به وزارت دفاع فروخته می شوند، اجباریست. دیوید کلارک نقل قول جالبی درباره استانداردهای اینترنت دارد که اکنون بسیار معروف است: «اجماع نظری ناقص، به همراه کُدی که کار می کند».

۲. واحدهای اندازه گیری

برای اجتناب از هر گونه سوء تفاهم، لازم است تأکید کنیم که در این کتاب بجای واحدهای اندازه گیری انگلیسی (با آن اندازه ها و ضرایب عجیب و غریب) از سیستم متریک استفاده شده است. در شکل ۱-۳۹ پیشنندهای اصلی سیستم متریک را مشاهده می کنید. اغلب از حروف اختصاری این پیشندها استفاده خواهیم کرد، و واحد اعداد بزرگتر از ۱ با حروف بزرگ می نویسیم، مانند KB و MB (البته با یک استثناء تاریخی: حروف اختصاری

پیشوند	عدد اعشاری	توان	پیشوند	عدد اعشاری	توان
Kilo	1,000	10^3	milli	0.001	10^{-3}
Mega	1,000,000	10^6	micro	0.000001	10^{-6}
Giga	1,000,000,000	10^9	nano	0.000000001	10^{-9}
Tera	1,000,000,000,000	10^{12}	pico	0.000000000001	10^{-12}
Peta	1,000,000,000,000,000	10^{15}	fermto	0.0000000000000001	10^{-15}
Exa	1,000,000,000,000,000,000	10^{18}	atto	0.0000000000000000001	10^{-18}
Zetta	1,000,000,000,000,000,000,000	10^{21}	zepto	0.0000000000000000000001	10^{-21}
Yotta	1,000,000,000,000,000,000,000,000	10^{24}	yocto	0.000000000000000000000001	10^{-24}

شکل ۱-۳۹. پیشوندهای اصلی سیستم متریک.

کیلوبیت/ثانیه بصورت kbps نوشته می شود. برای مثال، یک خط انتقال 1-Mbps در هر ثانیه 10^6 بیت اطلاعات را منتقل می کند، و 100 psec معادل 10^{-10} ثانیه است. برای تمایز بین پیشوندهای «میلی» و «میکرو» - که هر دو با m شروع می شوند -، اولی را با "m" و دومی را با "μ" نمایش می دهیم.

همچنین اشاره به این نکته خالی از فایده نیست که، واحدهایی که در صنعت کامپیوتر برای اندازه گیری ظرفیت حافظه، دیسک، و فایل بکار برده می شوند، کمی با سایر واحدها تفاوت دارند. در اینجا، «کیلو» بجای 10^3 (معادل 1000) معنای 2^{10} (یا 1024) می دهد، چون ظرفیت حافظه همیشه توانی از ۲ است - بنابراین، وقتی گفته می شود 1-KB، بمعنای 1024 بایت است، نه 1000 بایت. به همین ترتیب، 1-MB معادل 2^{20} (1,048,576 بایت) است، 1-GB معادل 2^{30} (1,073,741,824 بایت)، و 1-TB معادل 2^{40} (1,099,511,627,776 بایت). از طرف دیگر، در صحبت از سرعت انتقال داده ها، دیگر اعداد توانی از ۲ نیستند؛ برای مثال، خط 1-kbps دقیقاً 1000 بیت/ثانیه داده منتقل می کند، و یک شبکه 1-Mbps دقیقاً با سرعت 1,000,000 بیت/ثانیه کار می کند. متأسفانه، بسیاری از افراد قادر به تفکیک این دو نیستند (بویژه در مورد ظرفیت دیسکها). برای اجتناب از هر گونه ابهام، در این کتاب واحدهای KB، MB، و GB بترتیب معادل 2^{10} ، 2^{20} ، و 2^{30} بایت، و واحدهای kbps، Mbps، و Gbps بترتیب معادل 10^3 ، 10^6 ، و 10^9 بیت/ثانیه در نظر گرفته خواهند شد.

۸-۱ طرح کلی مباحث کتاب

در این کتاب شبکه های کامپیوتری از جنبه نظری و عملی مورد بررسی قرار گرفته اند. اکثر فصول کتاب با بحث در کلیات موضوع مورد نظر شروع شده، و با بررسی مثالها و نمونه های عملی ادامه می یابد. این نمونه ها از اینترنت و شبکه های بیسیم انتخاب شده اند، چون در عین اهمیت بسیار با یکدیگر تفاوت های اساسی نیز دارند. هر گاه لازم بوده به مثالهای دیگر نیز استناد کرده ایم.

در این کتاب کار خود را بر اساس مدل ترکیبی شکل ۱-۲۴ بنا نهاده ایم، و از فصل آینده بحث درباره سلسله مراتب پروتکل های این مدل را (از پائین به بالا) شروع خواهیم کرد. در فصل ۲ ابتدا پیش زمینه ای درباره سیستم های مخابرات داده (که شامل سیستم های کابلی، بیسیم و ماهواره ای می شود) بدست خواهیم داد. این سیستم ها که به لایه فیزیکی مربوط می شوند، بیشتر از جنبه معماری مورد بحث قرار گرفته اند تا جنبه سخت افزاری. در این فصل نمونه های متعددی را مورد بررسی قرار داده ایم، از شبکه های سونیچینگ تلفن معمولی و تلفن همراه گرفته، تا

شبکه‌های تلویزیون کابلی.

در فصل ۳ لایه پیوند داده و پروتکل‌های آن (به‌همراه تحلیل این پروتکل‌ها) به کمک مثال‌های متعدد مورد بحث قرار گرفته است. پس از آن، تعدادی از پروتکل‌های مهم این لایه، از جمله HDLC (که در شبکه‌های با سرعت کم تا متوسط مورد استفاده قرار می‌گیرد) و PPP (که در اینترنت کاربرد دارد) را مورد بررسی قرار خواهیم داد.

فصل ۴ با زیرلایه دسترسی داده (که بخشی از لایه پیوند داده است) سروکار دارد. بحث اصلی در اینجا نحوه دسترسی به شبکه در جاهانیست که کانال‌های فیزیکی شبکه به صورت اشتراکی مورد استفاده قرار می‌گیرند (مانند شبکه‌های LAN و برخی از شبکه‌های ماهواره‌ای). در این فصل نیز نمونه‌های متعددی (از جمله اترنت، LAN بیسیم، MAN بیسیم، بلوتوث، و شبکه‌های ماهواره‌ای) مورد بررسی قرار گرفته است. درباره پل (bridge) و سوئیچ (switch) نیز در همین فصل صحبت کرده‌ایم.

فصل ۵ به لایه شبکه، بویژه مسیریابی (routing) و الگوریتم‌های آن (استاتیک و دینامیک)، اختصاص دارد. حتی با بهترین الگوریتم‌های مسیریابی، اگر بار بیش از حد به یک شبکه تحمیل شود، امکان بروز ازدحام (congestion) در آن وجود دارد، بنابراین یکی از مباحث مهم این فصل ازدحام و راه‌های جلوگیری از آن (و از آن هم فراتر، تضمین نوعی کیفیت سرویس) است. نحوه اتصال شبکه‌های غیرمتجانس و رفع مشکلات آن از دیگر مباحث این فصل است. لایه شبکه از اهمیت زیادی در اینترنت برخوردار است.

در فصل ۶ درباره لایه انتقال (و بویژه پروتکل‌های اتصال-گرا) مفصلاً صحبت خواهیم کرد. حتی یکی از سرویس‌های ساده این لایه بصورت عملی (به‌همراه کد آن) ارائه شده است، تا با نحوه پیاده‌سازی آن بیشتر آشنا شوید. پروتکل‌های اینترنتی TCP و UDP (و کارایی آنها) نیز مفصلاً در این فصل مورد بررسی قرار گرفته‌اند. از دیگر مباحثی که در فصل ۶ مطرح خواهد شد، شبکه‌های بیسیم است.

فصل ۷ با لایه کاربرد (و پروتکل‌ها و برنامه‌های آن) سروکار دارد. اولین مبحث این فصل، دفترچه تلفن اینترنت یعنی DNS است. ایمیل و پروتکل‌های آن مبحث بعدی فصل ۷ است. پس از آن سراغ وب می‌رویم، و مباحثی مانند محتویات استاتیک و دینامیک، فعل و انفعالات سمت مشتری و سرویس دهنده، پروتکل‌های آن، کارایی وب، و وب بیسیم را مورد بررسی قرار خواهیم داد. در پایان نیز، درباره شبکه‌های چندرسانه‌ای (صدا و تصویر جویباری - streaming -، و رادیوی اینترنتی) صحبت خواهیم کرد.

فصل ۸ درباره امنیت شبکه است. از آنجائیکه مباحث این فصل با تمام لایه‌ها سروکار دارد، آنرا در آخر (بعد از پایان بحث لایه‌های شبکه) آورده‌ایم. این فصل را با معرفی تکنولوژیهای رمزنگاری (cryptography) آغاز کرده‌ایم، و نشان داده‌ایم که چگونه می‌توان به کمک این تکنولوژیها امنیت مخابرات داده، ایمیل و وب را تأمین کرد. این فصل را با بحثی درباره تقابل بین امنیت و حریم خصوصی افراد، آزادی بیان، سانسور (و سایر موضوعاتی که با امنیت شاخ به شاخ می‌شوند) به پایان رسانده‌ایم.

در فصل ۹ مقالات و کتابهایی را که می‌توانید برای مطالعه بیشتر درباره موضوعات هر فصل به آنها مراجعه کنید، به همان ترتیب فصول کتاب آورده‌ایم. در این فصل یک کتابنامه مفصل در زمینه موضوعات مطرح شده در کتاب نیز آورده شده است.

در سایت وب مؤلف کتاب

<http://www.prenhall.com/tanenbaum>

می‌توانید لینک‌های متعددی به دروسنامه‌ها، صفحات پرسش و پاسخ، شرکتها، کنسرسیومهای صنعتی، سازمانهای حرفه‌ای، سازمانهای استاندارد، تکنولوژیها، و مقالات تخصصی پیدا کنید.

۹-۱ خلاصه

از شبکه های کامپیوتری می توان برای مقاصد مختلفی (در شرکتها، یا برای افراد عادی) استفاده کرد. در شرکتها، شبکه می تواند دسترسی به منابع اطلاعاتی را برای تمام کارکنان فراهم آورد. در این شبکه ها معمولاً از مدل مشتری-سرویس دهنده (که در آن منابع مشترک روی کامپیوترهای قدرتمندی موسوم به سرویس دهنده - server - قرار می گیرند) استفاده می شود. شبکه برای افراد عادی امکان دسترسی به منابع اطلاعاتی یا تفریحی را فراهم می آورد. امروزه افراد بسیاری با یک مودم از منزل خود به شرکتهای خدمات اینترنتی (ISP) متصل شده، و از امکانات آن استفاده می کنند. یکی از زمینه هایی که امروزه سرعت رو به گسترش است، شبکه های بیسیم است، که امکان دسترسی به اینترنت را حتی از تلفنهای همراه به افراد می دهد.

در یک تقسیم بندی بسیار کلی، شبکه ها را می توان به شبکه های محلی (LAN)، شبکه های شهری (MAN)، شبکه های گسترده (WAN) و شبکه های (internetwork) تقسیم کرد، که هر کدام برای خود دارای ویژگیها، سرعت، تکنولوژی و جایگاه خاصی می باشد. شبکه محلی (LAN) سرعت بالایی دارد، ولی معمولاً به یک ساختمان منفرد محدود می شود. محدوده جغرافیایی شبکه شهری (MAN) - همانطور که از نام آن برمی آید - یک شهر است؛ تلویزیون کابلی نمونه ای از شبکه های شهری است. شبکه های گسترده (WAN) یک کشور یا قاره را در بر می گیرند. شبکه های LAN و MAN سوئیچ نشده هستند (یعنی، مسیریاب - router - ندارند)، در حالیکه شبکه های WAN سوئیچ شده اند. امروزه شبکه های بیسیم (به ویژه LAN بیسیم) از محبوبیت روزافزونی برخوردارند. از اتصال چند شبکه به یکدیگر نیز یک شبکه شبکه ها شکل می گیرد.

نرم افزار شبکه از پروتکلها (که قواعد حاکم بر ارتباط پروسس ها را تعیین می کنند) تشکیل می شود. پروتکلها با اتصال-گرا (connection-oriented) هستند یا غیرمتصل (connectionless). اغلب شبکه ها از مدل سلسله مراتبی پروتکلها استفاده می کنند، که در آن هر لایه سرویسهایی را در اختیار لایه های بالاتر قرار می دهد، و آنرا از جزئیات کار در لایه های پایینتر ایزوله می کند. مجموعه پروتکل (protocol stack) های مهم امروزی عمدتاً بر مبنای دو مدل OSI و TCP/IP بنا شده اند. هر دوی این مدلها دارای لایه های شبکه، انتقال و کاربرد هستند، ولی در لایه های دیگر با هم فرق دارند. هنگام طراحی پروتکلها باید به مسائلی از قبیل مالتی پلکس کردن (multiplexing)، کنترل جریان (flow control)، کنترل خطا (error control) و مانند آنها توجه ویژه مبذول داشت. (بخش عمده ای از این کتاب به پروتکلها و طراحی آنها اختصاص دارد).

وظیفه شبکه ارائه سرویس به کاربران است، که این سرویسها نیز می توانند اتصال-گرا یا غیرمتصل باشند. در برخی شبکه ها، یک لایه سرویس غیرمتصل در اختیار می گذارد، در حالیکه لایه بالاتر سرویس اتصال-گرا ارائه می کند.

معروفترین شبکه های موجود عبارتند از: اینترنت، ATM، اترنت و LAN بیسیم (IEEE 802.11). اینترنت محصول تکامل شبکه آرپانت (ARPANET - شبکه سوئیچ بسته وزارت دفاع ایالات متحده آمریکا) بود. اینترنت در واقع امروزه شبکه ایست از هزاران شبکه دیگر، نه یک شبکه واحد. مشخصه اصلی اینترنت استفاده از مجموعه پروتکل TCP/IP است. شبکه های ATM بیشتر در سیستمهای تلفن، و برای مخابرات راه دور، بکار می روند. اترنت (Ethernet) نیز محبوبترین تکنولوژی LAN است، که در اغلب شرکتها و دانشگاهها از آن استفاده می شود. و بالاخره، شبکه های LAN بیسیم که (با سرعت خیره کننده 54 Mbps) سرعت در حال گسترش هستند.

برای آنکه چند کامپیوتر بتوانند با هم ارتباط برقرار کنند، مهمترین موضوع وجود استانداردهای متعدد (سخت افزاری و نرم افزاری) است. وظیفه تدوین استاندارد برای شبکه های کامپیوتری بر عهده سازمانهایی از قبیل IEEE، ISO، ITU-T و IAB گذلشته شده است.

مسائل

۱. فرض کنید سگی از نژاد سنت برنارد (نوعی سگ قوی هیکل که در کشور کوهستانی سوئیس برای نجات کوهنوردان ساخته شده تربیت می شود) پنم پرنی دارید، و او را برای حمل جعبه ای حاوی سه نوار 8mm آموزش داده اید (به هر حل پُر شدن دیسک هم نوعی موقعیت اورژانس است). هر نوار 7 GB ظرفیت دارد، و پرنی می تواند تحت هر شرایطی با سرعت 18 km/h حرکت کند. تا چه فاصله ای نرخ انتقال اطلاعات پرنی همچنان از خط انتقالی با سرعت 150 Mbps (بدون در نظر گرفتن سرآیند) بیشتر است؟
۲. یکی از گزینه های رقیب شبکه های LAN سیستم های تسهیم زمانی (time sharing) بزرگ (به همراه ترمینالهایی که کاربران از آنها استفاده می کنند) است. دو مزیت شبکه های LAN مبتنی بر مدل مشتری-سرویس دهنده را نسبت به سیستم های فوق بیان کنید.
۳. کارایی سیستم های مشتری-سرویس دهنده به دو ویژگی مهم شبکه وابسته است: پهنای باند (bandwidth - حداکثر تعداد بیت هایی که شبکه می تواند در هر ثانیه منتقل کند)، و زمان تأخیر (latency time - مدت زمانی که طول می کشد تا اولین بیت از مشتری به سرویس دهنده - یا بالعکس - برسد). دو مثال از شبکه ای با پهنای باند زیاد و زمان تأخیر زیاد، و پهنای باند کم و زمان تأخیر کم بزنید.
۴. غیر از پهنای باند زیاد و زمان تأخیر کم، چه شرایط دیگری باید فراهم باشد تا یک شبکه کیفیت مناسبی برای سرویس صدای دیجیتال داشته باشد؟
۵. یک از عوامل مهم در میزان تأخیر سیستم های سوئیچینگ بسته مبتنی بر مدل ذخیره-هدایت، مدت زمان نیست که صرف ذخیره و هدایت بسته در یک سوئیچ می شود. آیا زمان سوئیچینگ 10 μsec برای یک سیستم مشتری-سرویس دهنده که بین کالیفرنیا و نیویورک کار می کند، عامل مهمی محسوب می شود. سرعت انتشار امواج الکترومغناطیس در سیم مسی را 0.667 سرعت نور در نظر بگیرید.
۶. یک سیستم مشتری-سرویس دهنده از شبکه ماهواره ای، که ماهواره آن در ارتفاع 40,000 km سطح زمین قرار گرفته، استفاده می کند. زمان تأخیر پاسخ در بهترین حالت چقدر است؟
۷. در آینده ای نه چندان دور مردم می توانند مستقیماً از طریق ترمینالهای متصل به شبکه های کامپیوتری به لوايح و طرح های قانونی رأی بدهند، و دیگر نیازی به مجالس قانونگذاری وجود نخواهد داشت. جنبه های مثبت چنین دموکراسی مستقیمی نسبتاً روشن است؛ کمی درباره نقاط منفی آن بحث کنید.
۸. می خواهیم پنج مسیر یاب را در یک زیر شبکه نقطه-به-نقطه به هم وصل کنیم. هر زوج از این مسیر یاب ها را می توان با یک خط پُر سرعت، خطی با سرعت متوسط، و یا یک خط کم سرعت به هم متصل کرد، و یا اینکه اصلاً آنها را به هم وصل نکرد. اگر یک کامپیوتر بتواند هر توپولوژی را در 100 ms طراحی و بررسی کند، چه مدت طول می کشد تا تمام توپولوژی های ممکن را بررسی کند؟
۹. تعداد 1 - 2 مسیر یاب در یک درخت باینری متقارن (یک مسیر یاب در هر گره) به هم متصل شده اند. مسیر یاب i برای ارتباط با مسیر یاب j، باید ابتدا پیام خود را به ریشه این درخت بفرستد، تا از آنجا بدست مسیر یاب j برسد. (با فرض اینکه تمام مسیر یاب یکسان هستند) عبارتی بنویسید که تعداد متوسط پرش های لازم برای رسیدن پیام یک مسیر یاب به مسیر یاب دیگر را (برای n های بزرگ) بدست دهد.
۱۰. یکی از نقاط منفی زیر شبکه های پخش ظرفیتی است که در اثر اقدام به پخش همزمان توسط چند کامپیوتر از دست می رود (تصادم - collision). اجازه دهید مسئله را ساده کرده، و فرض کنیم زمان به پرش های مساوی تقسیم شده، و در هر پرش زمانی n کامپیوتر با احتمال p اقدام به پخش روی شبکه می کنند. چه

- کسری از برشهای زمانی در اثر بروز حالت تصادم تلف خواهند شد؟
۱۱. دو دلیل برای استفاده از پروتکل های لایه ای ارائه کنید.
 ۱۲. رئیس یک شرکت رنگسازی ایده جدیدی برای تولید یک محصول بدیع و نو ظهور دارد. موضوع را با اداره حقوقی شرکت در میان می گذارد، و آنها هم از اداره مهندسی درخواست کمک می کنند. سرپرست اداره مهندسی درباره زوایای مختلف این طرح با یکی از هم تایان خود در شرکتی دیگر مشورت می کند، که وی نیز موضوع را به اداره حقوقی شرکت خود منتقل می کند. در پایان نیز، رؤسای دو شرکت بر سر جزئیات مالی معامله با یکدیگر به مذاکره می پردازند. آیا می توان این سناریو را مطابق با پروتکل های چند لایه مدل OSI دانست؟
 ۱۳. دو تفاوت عمده بین ارتباطات اتصال-گرا و غیر متصل چیست؟
 ۱۴. دو شبکه سرویسهای اتصال-گرای قابل اعتماد ارائه می کنند: یکی از آنها بصورت استریم بایت، و دیگری بصورت استریم پیام. آیا این دو یکسان هستند؟ اگر پاسخ مثبت است، چه دلیلی برای نامگذاری جداگانه آنها وجود دارد؟ اگر خیر، تفاوت آنها در چیست؟
 ۱۵. در بحث پروتکل های شبکه، «مذاکره» (negotiation) چه معنایی دارد؟ یک مثال بزنید.
 ۱۶. در شکل ۱-۱۹ یک سرویس نشان داده شده است. آیا سرویس دیگری در این شکل وجود دارد؟ اگر آری، کجا؟ اگر خیر، چرا؟
 ۱۷. در برخی از شبکه ها، لایه پیوند داده با درخواست ارسال مجدد فریم هایی که بدرستی دریافت نشده اند، نوعی کنترل خطا انجام می دهد. اگر احتمال خراب شدن یک فریم p باشد، متوسط تعداد دفعاتی که یک فریم باید فرستاده شود، چقدر است؟ فرض کنید فریم های تصدیق دریافت (acknowledgement) هرگز خراب نمی شوند.
 ۱۸. کدامیک از لایه های OSI وظایف ذیل را بر عهده دارند:
(الف) تقسیم استریم بیتها به فریم
(ب) تعیین مسیری در زیر شبکه، که باید از آن استفاده شود
 ۱۹. اگر واحد تبادل داده در لایه پیوند داده فریم، و در لایه انتقال بسته نام داشته باشد، فریمها در بسته پیچیده می شوند، یا بسته ها در فریم؟ توضیح دهید.
 ۲۰. ساختار سلسله مراتبی پروتکل های یک سیستم n لایه دارد. برنامه های کاربردی در این سیستم پیام هایی بطول M بایت تولید می کنند، و در هر لایه یک سرآیند h بایتی به پیام لایه بالاتر اضافه می شود. چه کسری از پهنای باند شبکه را این سرآیندها اشغال می کنند؟
 ۲۱. دو شباهت و دو تفاوت مدل های مرجع را OSI و TCP/IP را بیان کنید.
 ۲۲. تفاوت اصلی TCP و UDP چیست؟
 ۲۳. زیر شبکه شکل ۱-۲۵ (ب) طوری طراحی شده که در مقابل حملات هسته ای دوام بیاورد. چند بمب اتمی لازم است تا این زیر شبکه به دو قسمت کاملاً مجزا تقسیم شود؟ فرض کنید برای نابود کردن هر گره و تماء لینکهای متصل به آن یک بمب اتمی کافیت.
 ۲۴. تخمین زده شده است که اینترنت هر ۱۸ ماه دو برابر می شود. (و با اینکه کسی واقعاً تعداد آنها را نمی داند) طبق برآوردهای تخمینی در سال ۲۰۰۱ تعداد کامپیوترهای اینترنت ۱۰۰ میلیون بوده است. با استفاده از این

مفروضات، تعداد کامپیوترهای اینترنت در سال ۲۰۱۰ را محاسبه کنید. آیا این عدد را باور می‌کنید؟ توضیح دهید.

۲۵. هنگام تبادل فایل بین دو کامپیوتر، دو استراتژی تصدیق دریافت ممکن است. در روش اول، فایل به قطعاتی تقسیم شده، و برای هر قطعه تصدیق دریافت جداگانه مطالبه می‌شود، اما برای کل فایل، خیر. در روش دوم، برای تک تک قطعات تصدیق دریافت مطالبه نمی‌شود، ولی در پایان دریافت کل فایل باید به تأیید طرف مقابل برسد. درباره این دو روش بحث کنید.

۲۶. چرا ATM از سلولهای کوچک و با اندازه ثابت استفاده می‌کند؟

۲۷. هر بیت در استاندارد اولیه 802.3 چند متر طول داشت؟ سرعت کار این شبکه را 10 Mbps، و سرعت سیگنالهای الکتریکی در کابل کواکسیال را 2/3 سرعت نور در خلاء فرض کنید.

۲۸. ابعاد یک تصویر 1024×768 پیکسل است، و هر پیکسل آن 3 بایت جا می‌گیرد (و فرض کنید این تصویر فشرده نشده است). انتقال این تصویر روی یک خط 56-kbps چقدر طول می‌کشد؟ روی یک خط 1-Mbps چقدر؟ روی شبکه اینترنت 10-Mbps چقدر؟ و روی شبکه اینترنت 100-Mbps چقدر؟

۲۹. اینترنت و شبکه‌های بیسیم شباهتها و تفاوتهایی دارند. یکی از ویژگیهای اینترنت اینست که در هر لحظه فقط یک فریم می‌تواند روی شبکه منتقل شود. آیا در شبکه‌های 802.11 نیز چنین است؟ توضیح دهید.

۳۰. نصب شبکه‌های بیسیم بسیار ساده است، که همین باعث ارزانی آنها شده است، چون معمولاً هزینه‌های نصب در مقابل هزینه تجهیزات رقم قابل توجهی را تشکیل می‌دهد. با این حال، این نوع شبکه معایبی نیز دارد. دو تا از این معایب را نام ببرید.

۳۱. دو مزیت و دو عیب برای استاندارد کردن پروتکل‌های شبکه بشمارید.

۳۲. وقتی یک سیستم از دو قسمت ثابت و متحرک تشکیل می‌شود (مانند درایو CD، و دیسک CD-ROM)، استاندارد کردن آن از اهمیت زیادی برخوردار است، تا محصولات شرکت‌های مختلفی که این قطعات را تولید می‌کنند، با هم سازگار باشد. سه مثال از خارج صنعت کامپیوتر بزنید، که چنین استاندارد در آنها وجود دارد. حال سه مثال از خارج صنعت کامپیوتر بزنید، که چنین استاندارد در آنها وجود نداشته باشد.

۳۳. تمام فعالیتهای خود را که در طول شبانه روز بنحوی با شبکه سروکار دارد، فهرست وار بنویسید. اگر این شبکه‌ها به یکباره از کار بیفتند، چه تأثیری روی زندگی شما خواهد گذاشت؟

۳۴. شبکه‌هایی را که در محل کار یا تحصیل خود می‌شناسید (به‌مراه نوع، توپولوژی، و روش سونیچینگ آنها)، مشخص کنید.

۳۵. برنامه ping به شما اجازه می‌دهد تا بسته‌ای را به یک مقصد مشخص فرستاده، و زمان رفت و برگشت آنرا محاسبه کنید. از این برنامه برای تعیین زمان رفت و برگشت بسته‌ها به چند نقطه مختلف استفاده کنید. با استفاده از این اطلاعات، نمودار زمان انتقال بسته‌ها روی اینترنت (در جهت رفت) را بر حسب فاصله رسم کنید. بهتر است برای این منظور از دانشگاه‌های مهم و سرشناس (که محل آنها دقیقاً مشخص است) استفاده کنید. برای مثال، سایت berkeley.edu در برکلی-کالیفرنیا است، سایت mit.edu در کمبریج-ماساچوست،

سایت vu.nl در آمستردام-هلند، سایت www.usyd.edu.au در سیدنی-استرالیا، و سایت www.uct.ac.za در کیپ تاون-آفریقای جنوبی.

۳۶. سری به سایت IETF (به آدرس www.ietf.org) بزنید، و ببینید چکار می کنند. یکی از پروژه هایی را که به آن علاقه دارید، انتخاب کرده، و درباره آن مقاله ای نیم صفحه ای بنویسید.

۳۷. استاندارد سازی در دنیای شبکه های کامپیوتری بسیار مهم است، و همانطور که دیدید سازمانهای ITU و ISO این وظیفه را بر عهده دارند. به سایت وب این سازمانها (www.iso.org و www.iyu.org) رفته، و با کارهای استاندارد سازی آنها آشنا شوید. گزارش کوتاهی درباره انواع چیزیهایی که این دو سازمان استاندارد کرده اند، بنویسید.

۳۸. اینترنت از تعداد زیادی شبکه های مختلف تشکیل شده است، که آرایش آنها توپولوژی اینترنت را مشخص می کند. در خود اینترنت اطلاعات زیادی در زمینه توپولوژی آن وجود دارد. با استفاده از یک موتور جستجو (search engine) در باره این موضوع تحقیق کرده، و خلاصه ای از یافته های خود را در یک گزارش بنویسید.

لایه فیزیکی



در این فصل پانین ترین لایه سلسله مراتب شکل ۱-۲۴ را مورد بررسی قرار خواهیم داد. مشخصات مکانیکی، الکتریکی و تایمینگ (همزمانی) شبکه در این لایه تعریف می شود. برای شروع کمی درباره تئوری مخابرات صحبت می کنیم، فقط برای اینکه نشان دهیم دست طبیعت چه محدودیتهایی را در زمینه انتقال داده ها به ما تحمیل کرده است.

سپس با رسانه های فیزیکی که برای انتقال داده ها از آنها استفاده می شود، آشنا می شوید: رسانه های هدایت پذیر (مانند سیم مسی و فیبر نوری)، بیسیم (امواج رادیویی زمینی)، و ماهواره. آشنایی با این رسانه ها برای شناخت تکنولوژیهای مدرن مخابراتی اهمیت اساسی دارد.

در ادامه سه نمونه از سیستمهای مخابراتی که در شبکه های کامپیوتری کاربرد گسترده ای دارند، را مفصلاً مورد بررسی قرار خواهیم داد: سیستم تلفن ثابت، شبکه تلفن همراه، و تلویزیون کابلی. در تمام این سیستمها از فیبر نوری بعنوان ستون فقرات (backbone) استفاده می شود، ولی تکنولوژی بکار رفته در آخرین قطعه اتصال (از مرکز تلفن یا ایستگاه توزیع تا مصرف کننده) در آنها متفاوت است.

۱-۲ مبانی نظری مخابرات داده

برای انتقال اطلاعات روی سیم می توان از متغیرهای الکتریکی مانند ولتاژ یا جریان استفاده کرد. با نمایش تغییرات این ولتاژ یا جریان بصورت تابعی از زمان، می توان رفتار سیگنال را مدل سازی و تحلیل ریاضی کرد. موضوع این بخش تحلیل ریاضی سیگنالهای الکتریکی است.

۱-۱-۲ آنالیز فوریه

در اوایل قرن نوزدهم میلادی، ریاضیدان فرانسوی ژان بپتیست فوریه ثابت کرد که هر تابع متناوب، $g(t)$ ، با دوره تناوب T را می توان به صورت مجموع بینهایت جمله سینوسی و کسینوسی نوشت:

$$g(t) = \frac{1}{T} c + \sum_{n=1}^{\infty} a_n \sin(\gamma \pi n f t) + \sum_{n=1}^{\infty} b_n \cos(\gamma \pi n f t) \quad (1-2)$$

که در آن $f = 1/T$ فرکانس اصلی، a_n و b_n ضرایب هارمونی (جمله n ام تابع سینوس و کسینوس، و c یک عدد ثابت است. به تجزیه یک تابع به مجموع بینهایت جمله سینوسی و کسینوسی سری فوریه (Fourier series) گفته می شود. اگر ضرایب جملات سری فوریه یک تابع و دوره تناوب آن (T) معلوم باشد، می توان این تابع را با

استفاده از معادله (۱-۲) ایجاد کرد.

اگر یک سیگنال داده دارای دوره محدودی باشد (که همیشه هم چنین است)، می توان آنرا تکرار بینهایت یک الگوی مشخص فرض کرد (بعبارت دیگر، این موج در فاصله زمانی T تا $2T$ با فاصله 0 تا T کاملاً یکسان است). برای محاسبه ضرایب a_n کافیهست دو طرف معادله (2-1) را در $\sin(2\pi kft)$ ضرب کرده، و سپس در فاصله 0 تا T از آن انتگرال بگیریم. از آنجائیکه

$$\int_0^T \sin(\gamma \pi kft) \sin(\gamma \pi nft) dt = \begin{cases} 0 & \text{for } k \neq n \\ T/\gamma & \text{for } k = n \end{cases}$$

فقط جمله ای که ضریب a_n دارد، باقی می ماند، و جملات b_n بکلی از بین خواهند رفت. به همین ترتیب، اگر دو طرف معادله (2-1) را در $\cos(2\pi kft)$ ضرب کرده، و سپس در فاصله 0 تا T از آن انتگرال بگیریم، ضریب b_n بدست خواهد آمد. برای بدست آوردن ثابت c هم کافیهست از تابع $g(t)$ در همین فاصله زمانی انتگرال بگیریم. خلاصه عملیات فوق را در فرمولهای ذیل مشاهده می کنید:

$$a_n = \frac{\gamma}{T} \int_0^T g(t) \sin(\gamma \pi nft) dt \quad b_n = \frac{\gamma}{T} \int_0^T g(t) \cos(\gamma \pi nft) dt \quad c = \frac{\gamma}{T} \int_0^T g(t) dt$$

۲-۱-۲ محدودیت پهنای باند

اما برای اینکه ببینید اینها چه ربطی به مخابرات داده دارد، اجازه دهید یک مثال بزنیم: مخابره کاراکتر آسکی حرف "b" که بصورت ۸ بیتی کُد شده است. حرف "b" در استاندارد آسکی ۸ بیتی بصورت 01100010 کُد می شود، و اینها بیتهایی هستند که باید مخابره شوند. در نمودار سمت چپ شکل ۱-۲ (الف) ولتاژهای خروجی کامپیوتر برای حرف "b" را می بینید. با استفاده از آنالیز فوریه، ضرایب ثابت این سیگنال بصورت زیر بدست می آیند:

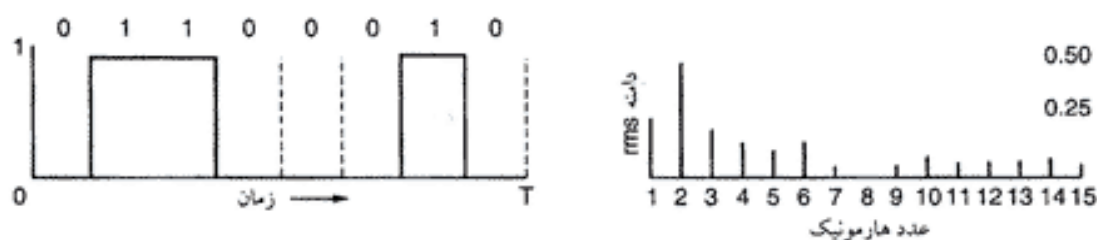
$$a_n = \frac{1}{\pi n} [\cos(\pi n/4) - \cos(3\pi n/4) + \cos(6\pi n/4) - \cos(7\pi n/4)]$$

$$b_n = \frac{1}{\pi n} [\sin(3\pi n/4) - \sin(\pi n/4) + \sin(7\pi n/4) - \sin(6\pi n/4)]$$

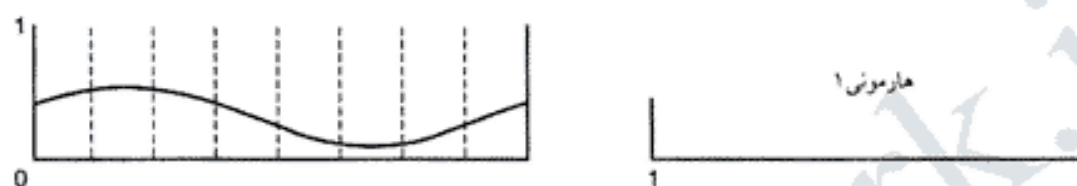
$$c = 3/4$$

در نمودار سمت راست شکل ۱-۲ (الف) مقدار rms ضرایب فوریه، یعنی $\sqrt{a_n^2 + b_n^2}$ ، چند هارمونی اول این سیگنال را مشاهده می کنید. علت اهمیت این ضرایب آنست که انرژی انتشار سیگنال با مجذور آنها (در یک فرکانس خاص) متناسب است.

از طرف دیگر می دانیم که هیچ سیستم پخشی بدون اتلاف انرژی نیست، ولی اگر تمام ضرایب فوریه به یکسان تضعیف شوند، سیگنال حاصله فقط ضعیفتر می شود، و بهیچوجه دچار اعوجاج (تغییر شکل) نخواهد شد. متأسفانه، در سیستمهای پخش مختلف ضرایب فوریه به میزانهای متفاوت تضعیف می شوند، و همین باعث بروز اعوجاج در شکل موج خواهد شد. معمولاً، تضعیف موج فقط از فرکانس خاصی به بالا (که به فرکانس قطع - f_c معروف است و با هرتز - Hz - یا سیکل بر ثانیه - cycles/sec - سنجیده می شود) روی می دهد، و زیر این فرکانس تضعیف موج وجود ندارد. طیف فرکانسی که موج می تواند بدون تضعیف منتشر شود، به پهنای باند (bandwidth) معروف است. از آنجائیکه در عمل فرکانس قطع عددی دقیق و قطعی نیست، پهنای باند بعنوان فرکانسی که در آن فقط نیمی از انرژی موج عبور می کند، تعریف می شود.



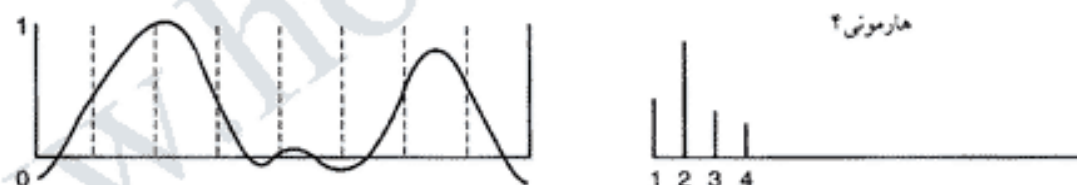
(الف)



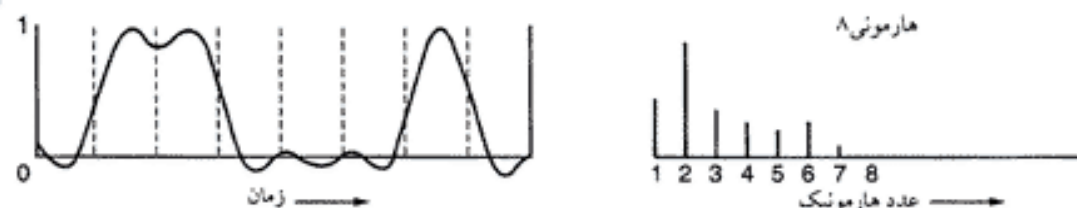
(ب)



(ج)



(د)



(ه)

شکل ۲-۱. (الف) یک سیگنال باینری و ضرایب فوریه rms آن. (ب)-(ه) تقریب‌های متوالی سیگنال اولیه.

پهنای باند یکی از خواص فیزیکی رسانه انتقال است، و معمولاً به نوع، شکل، ضخامت و طول آن بستگی دارد. در برخی موارد با قرار دادن یک فیلتر پهنای باندی را که در اختیار مشتری است، عملاً محدود می‌کنند. برای مثال، پهنای باند یک خط تلفن در فواصل کوتاه می‌تواند به 1 MHz برسد، ولی شرکت‌های تلفن با قرار دادن

فیلترهای ویژه آنرا به حدود 3100 Hz محدود می کنند. این پهنای باند برای انتقال واضح و مفهوم صدا کافیست، و از اتلاف منابع نیز جلوگیری می کند.

حال اجازه دهید ببینیم اگر پهنای باند خط انتقال آنقدر کم باشد که فقط فرکانسهای پائین بتوانند منتقل شوند (بعبارت دیگر فقط هارمونی های اول معادله 2-1) اجازه عبور داشته باشند، سیگنال شکل ۱-۲ (الف) به چه صورتی در می آید. در شکل ۱-۲ (ب) شکل موج را در حالتی که فقط هارمونی اول (هارمونی اصلی، f) اجازه عبور دارد، می بینید. در شکلهای ۱-۲ (ج)-(ه) نیز شکل موج برای پهنای باندهای بالاتر نشان داده شده اند. اگر نرخ انتقال اطلاعات b bits/sec باشد، زمان لازم برای ارسال (مثلاً) ۸ بیت معادل $8/b$ sec است، بنابراین فرکانس هارمونی اصلی $b/8$ Hz خواهد بود. همانطور که گفتیم، فرکانس یک خط تلفن معمولی (که به خط رده صوتی - voice-grade - معروف است) بصورت مصنوعی کمی بالای 3000 Hz نگه داشته می شود. این محدودیت باعث می شود تا بالاترین هارمونی که می تواند از این خط عبور کند، تقریباً $3000/(b/8)$ یا $24000/b$ باشد.

در شکل ۲-۲ اعداد محاسبه شده برای برخی از مهمترین نرخهای انتقال را می بینید. همانطور که از این شکل برمی آید، اگر بخواهیم داده ها را با سرعت 9600 bps روی یک خط رده صوتی بفرستیم، موج ما از شکل ۱-۲ (الف) به شکل ۱-۲ (ه) تبدیل خواهد شد - که باعث می شود تشخیص بیت های اولیه قدری مشکل شود. از همینجا پیداست که برای سرعت های بالای 38.4 kbps هیچ شانس برای ارسال داده های باینری وجود ندارد (حتی اگر خط ما کاملاً بدون نویز باشد). بعبارت دیگر، محدود کردن پهنای باند باعث محدود شدن نرخ انتقال اطلاعات (حتی در بهترین کانالها) خواهد شد. با این حال، راههایی وجود دارد (مثلاً، استفاده از چند سطح ولتاژ بجای دو سطح ولتاژ) که می توان به نرخهای انتقال بالاتری دست یافت، و در ادامه همین فصل درباره آنها صحبت خواهیم کرد.

۳-۱-۲ حداکثر نرخ داده در یک کانال

در سال ۱۹۲۴ یکی از مهندسان AT&T بنام هنری نایکوئیست دریافت که حتی بهترین کانال انتقال هم ظرفیت محدودی دارد، و توانست معادله حداکثر نرخ انتقال داده یک کانال بدون نویز را برای یک پهنای باند مشخص بدست آورد. در سال ۱۹۴۸ کلود شانون کار نایکوئیست را تکمیل کرد، و معادله وی را برای حالتی که کانال در معرض نویز تصادفی (نویز ترمودینامیک یا حرارتی) است، توسعه دهد (Shannon, 1948). در این قسمت کارهای این دو دانشمند را بطور مختصر بررسی خواهیم کرد.

تعداد هارمونی فرستاده شده	اولین هارمونی (Hz)	T (msec)	Bps
80	37.5	26.67	300
40	75	13.33	600
20	150	6.67	1200
10	300	3.33	2400
5	600	1.67	4800
2	1200	0.83	9600
1	2400	0.42	19200
0	4800	0.21	38400

شکل ۲-۲. رابطه بین نرخ انتقال داده و هارمونی ها.

نایکونیست ثابت کرد که اگر سیگنالی از یک فیلتر پائین گذر (low-pass filter) با پهنای باند H عبور داده شود، سیگنال خروجی (فیلتر شده) را می توان با داشتن فقط $2H$ نمونه در ثانیه بازسازی کرد. نمونه برداری از خط با نرخ بالاتر از $2H$ دقت بیشتری به همراه نخواهد داشت، چون فرکانسهای بالاتری که بدین ترتیب می توان بازیابی کرد، قبلاً توسط فیلتر حذف شده اند. اگر این سیگنال دارای V سطح مجزا باشد، طبق قضیه نایکونیست:

$$\text{maximum data rate} = 2H \log_2 V \text{ bits/sec}$$

برای مثال، یک کانال 3-kHz بدون نویز نمی تواند سیگنالهای باینری (دو سطح ولتاژ) بالاتر از 6000 bps را انتقال دهد.

تا اینجا کانالها را بدون نویز فرض کردیم، که در واقع چنین نیست. اگر در کانال انتقال نویز تصادفی (حرارتی) وجود داشته باشد، اوضاع سرعت رو به وخامت خواهد گذاشت (و می دانیم که هیچ گریزی از نویز حرارتی، که حاصل حرکت مولکولهاست، نیست). مقدار نویز حرارتی با نسبت توان سیگنال به توان نویز سنجیده می شود، و به نسبت سیگنال به نویز (signal-to-noise ratio) معروف است. اگر توان سیگنال را با S و توان نویز را با N نشان دهیم، نسبت سیگنال به نویز S/N خواهد بود. معمولاً آنچه که به عنوان نسبت سیگنال به نویز داده می شود، نه خود S/N بلکه $10 \log_{10} S/N$ است، که به دسی بل (decibel) معروف است و با dB نشان داده می شود. اگر نسبت سیگنال به نویز 10 باشد، معادل 10 دسی بل خواهد بود، نسبت 100 معادل 20 دسی بل است، نسبت 1000 معادل 30 دسی بل، و الی آخر. سازندگان تقویت کننده های استریو اغلب پهنای باندی را که دستگاه آنها بصورت خطی عمل می کند، بصورت فرکانس 3 دسی بل در هر انتهای طیف مشخص می کنند (فرکانس 3 دسی بل فرکانسی است که ضریب تقویت دستگاه نصف می شود، چون $\log_{10} 3 = 0.5$).

شانون معادله نایکونیست را برای کانالهای نویزدار بصورت زیر تصحیح کرد:

$$\text{maximum number of bits/sec} = H \log_2 (1 + S/N)$$

که در آن H پهنای باند کانال (بر حسب هرتز)، و S/N نسبت سیگنال به نویز است. برای مثال، در کانالی با پهنای باند 3000 Hz که دارای نویز حرارتی 30 dB است (نویز معمول در بخش آنالوگ سیستمهای تلفن)، هرگز نمی توان بیش از 30,000 bps داده ارسال کرد (بدون توجه به اینکه تعداد سطوح ولتاژ این سیگنال یا تعداد نمونه برداری ها چقدر باشد).

معادله شانون از تئوری اطلاعات (information theory) استنتاج شده، و برای هر کانالی که تحت تاثیر نویز حرارتی باشد صادق است. کانالی که این قاعده را نقض کند، باید معادل یک ماشین حرکت دائم فرض کرد (ماشینی که طبق اصول ترمودینامیک نمی تواند وجود خارجی داشته باشد). البته باید توجه داشته باشید که این عدد فقط حد بالای ظرفیت کانال را مشخص می کند، و سیستمهای واقعی بندرت به این حد دست پیدا می کنند.

۲-۲ رسانه انتقال هدایت پذیر

وظیفه لایه فیزیکی انتقال بیت های خام از یک ماشین به ماشین دیگر است. برای اینکار از رسانه های فیزیکی مختلفی می توان استفاده کرد، که هر کدام پهنای باند، تأخیر انتشار، هزینه، و سهولت نصب و نگهداری خاص خود را دارند. این رسانه ها را می توان به دو دسته کلی هدایت پذیر (مانند سیم مسی یا فیبر نوری) و هدایت ناپذیر (مانند امواج رادیویی و لیزری) تقسیم بندی کرد. در قسمتهای آینده این رسانه ها را مورد بررسی قرار خواهیم داد.

۱-۲-۲ رسانه مغناطیسی

یکی از متداولترین راههای انتقال اطلاعات از کامپیوتری به کامپیوتر دیگر، نوشتن آنها روی نوار مغناطیسی، دیسک، CD یا DVD و سپس خواندن آنها در کامپیوتر مقصد است. اگرچه این روش بخوبی استفاده از مخابرات

ماهواره‌ای نیست، ولی هزینه آن بسیار کمتر است، بویژه در مواردی که پهنای باند زیاد و قیمت کم جزء عوامل کلیدی باشند.

اجازه دهید با یک مثال ساده مطلب را روشن‌تر کنیم. امروزه نوارهای مغناطیسی با ظرفیت 200 GB بوفور در بازار یافت می‌شوند، و یک جعبه مقوایی به ابعاد $60 \times 60 \times 60$ cm می‌تواند 1000 تا از این نوارها را در خود جای دهد، که بدین ترتیب ظرفیت آن به 200 TB (ترابایت، معادل هزار گیگابایت) یا 1600 Tbit (که معادل 1.6 Pbit - پتابایت - است) می‌رسد. این بسته را می‌توان با استفاده از پست سریع‌السیر در کمتر از ۲۴ ساعت به هر نقطه‌ای در ایالات متحده آمریکا تحویل داد. بدین ترتیب، پهنای باند مؤثر این سیستم انتقال 1600 terabits/86,400 sec یا 19 Gbps است - و اگر فاصله این دو نقطه فقط یک ساعت باشد، پهنای باند تا 400 Gbps نیز افزایش خواهد یافت. هیچ شبکه کامپیوتری حتی نمی‌تواند به این سرعت نزدیک شود.

برای شبکه‌های بانکی که در هر روز باید چندین گیگابایت اطلاعات را از نقطه‌ای به نقطه دیگر منتقل کنند (تا در صورت بروز فجایع طبیعی نیز بتوانند به کار خود ادامه دهند) هیچ چیز نمی‌تواند جای نوار مغناطیسی را بگیرد. البته شبکه‌ها هر روز سریع‌تر می‌شوند، ولی ظرفیت نوارهای مغناطیسی نیز رو به افزایش است.

در مورد هزینه نیز وضعیت مشابهی وجود دارد. قیمت هر نوار مغناطیسی 200 GB چیزی حدود 40 دلار است (البته در خریدهای عمده). و از هر نوار می‌توان حداقل ۱۰ بار استفاده کرد. بنابراین، هزینه هر بار استفاده از جعبه نوار ۱۰۰۰ تا 4000 دلار خواهد شد. اگر 1000 دلار نیز برای حمل و نقل به آن اضافه کنیم (که البته معمولاً بسیار کمتر است)، هزینه کل به 5000 دلار برای انتقال 200 TB اطلاعات می‌رسد - و این یعنی ۳ سنت (معادل 0.01 دلار) برای هر گیگابایت، که هیچ شبکه‌ای نمی‌تواند با آن رقابت کند. نتیجه اخلاقی داستان:

هرگز پهنای باند کامیونی پُر از نوارهای مغناطیسی را که با سرعت در بزرگراه در حال حرکت است، دست کم نگیرید.

۲-۲-۲ زوج تابیده

با اینکه پهنای باند نوار مغناطیسی بسیار عالیست، تأخیر انتشار آن (زمانی که طول می‌کشد تا اولین بیت اطلاعات به مقصد برسد) ناامیدکننده است - در شبکه‌ها معمولاً با میلی‌ثانیه سروکار داریم نه روز و ساعت! در بسیاری از کاربردها برقرار بودن دائمی ارتباط یک امر حیاتی است. یکی از قدیمی‌ترین (و همچنان متداول‌ترین) رسانه‌های انتقال زوج تابیده (twisted pair) است. زوج تابیده عبارتست از یک زوج سیم مسی عایق‌دار (بضخامت 1 mm)، که صورت مارپیچ به دور یکدیگر تابیده‌اند (مانند زنجیره مولکول DNA). علت تابیدن سیمها آنست که دو سیم مسی معمولی مانند یک آنتن عمل کرده، و انرژی تلف می‌کنند. در حالت تابیده امواج سیمها یکدیگر را خنثی کرده، و تشعشع به حداقل می‌رسد.

بیشترین کاربرد زوج تابیده در شبکه‌های تلفن است: تقریباً تمام تلفنها با استفاده از زوج تابیده به مرکز تلفن وصل می‌شوند. از زوجهای تابیده می‌توان بطول چندین کیلومتر بدون نیاز به تقویت‌کننده استفاده کرد، ولی برای مسافتهای طولانیتر به تکرارکننده (repeater) نیاز هست. تعداد زیادی زوج تابیده که در یک غلاف محافظ جمع شده باشند، تشکیل یک کابل زوج تابیده را می‌دهند. اگر سیمهای این کابل بصورت دو به دو به یکدیگر نتابیده باشند، تداخل شدیدی بین آنها بوجود خواهد آمد.

از زوجهای تابیده برای انتقال سیگنالهای آنالوگ و دیجیتال می‌توان استفاده کرد. پهنای باند این خطوط به ضخامت سیمها و مسافت بستگی دارد، و در فواصل کوتاه (دو تا سه کیلومتر) می‌توان به پهنای باند چندین مگاهرتز بر ثانیه دست یافت. بدلیل کارایی کافی و هزینه پائین، از زوج تابیده بنحوی گسترده‌ای استفاده شده است، و بنظر می‌رسد تا سالها نیز این وضعیت ادامه یابد.



(الف)

(ب)

شکل ۲-۳. (الف) زوج تابیده Cat 3، (ب) زوج تابیده Cat 5.

انواع مختلفی از کابلهای زوج تابیده وجود دارد، که دو تا از آنها در شبکه های کامپیوتری از اهمیت بیشتری برخوردار هستند. در کابلهای Category 3 (که به Cat 3 نیز معروف است) سیمها با شدت کمتری به هم تابیده اند. کابل Cat 3 از چهار زوج تابیده، که در یک غلاف پلاستیکی قرار می گیرند، تشکیل می شود. تا قبل از سال ۱۹۸۸ تقریباً در همه جا از این نوع کابل استفاده می شد. کابل Cat 3 می تواند چهار خط تلفن معمولی (یا دو خط تلفن مرکب) را از ایستگاه تلفن به نقطه مورد نظر برساند.

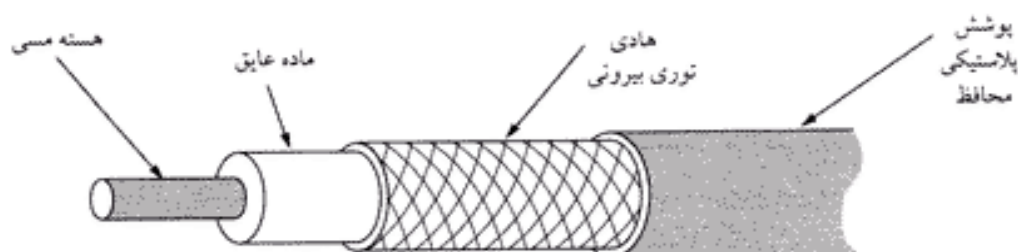
در سال ۱۹۸۸ کابل پیشرفته تر Category 5 (یا همان Cat 5) وارد بازار شد. این کابل شبیه Cat 3 است، با این تفاوت که تعداد دورهای آن در واحد طول بیشتر بوده، و به همین دلیل تداخل سیگنال در آن کاهش یافته و برای شبکه های پرسرعت مناسبتر است. کابلهای Cat 6 و Cat 7 نیز در حال آمدن به بازار هستند، که سرعت آنها پرتیب به 250 MHz و 600 MHz می رسد (برای مقایسه، سرعت کابلهای Cat 3 و Cat 5 پرتیب 16 MHz و 100 MHz است).

تمام این کابلها در بازار بنام UTP (زوج تابیده بدون زره - Unshielded Twisted Pair) شناخته می شوند، تا از کابلهای STP (زوج تابیده زره دار - Shielded Twisted Pair) که IBM در اوایل دهه ۱۹۸۰ معرفی کرد (و بدلیل گرانی و سختی کار با آن، استقبال چندانی از این نوع کابل نشد)، متمایز باشند. در شکل ۲-۳ دو نوع زوج تابیده Cat 3 و Cat 5 (و تفاوت آنها) را ملاحظه می کنید.

۳-۲-۲ کابل کواکسیال

یکی دیگر از رسانه های رایج کابل کواکسیال (هم محور - coaxial) است (که اغلب به آن کابل کواکس می گویند). این کابل بدلیل داشتن زره (غلاف فلزی) کارایی بهتری نسبت به زوج تابیده (هم از نظر سرعت، هم از نظر مسافت) دارد. کابل کواکسیال دو نوع دارد. اولی کابل 50-ohm است، که از ابتدا برای مخابرات دیجیتال در نظر گرفته شده؛ و دومی کابل 75-ohm، که ابتدا برای مخابرات آنالوگ و تلویزیون کابلی بکار می رفت، ولی امروزه با گسترش اینترنت کابلی از اهمیت روزافزونی برخوردار شده است. این تمایز بیش از آن که فنی باشد، جنبه تاریخی دارد: امپدانس آنتنهای دوقطبی قدیمی 300-ohm بود، و ترانسفورماتورهای تطبیق امپدانس 4:1 بوفور در بازار یافت می شد.

کابل کواکسیال تشکیل می شود از یک سیم مسی سخت بعنوان هسته (core)، یک لایه عایق استوانه ای به دور این هسته، یک لایه توری فلزی که به دور عایق بافته شده، و لایه پلاستیکی محافظ خارجی (شکل ۲-۴).



شکل ۲-۴. ساختمان کابل کواکسیال.

ساختمان و نحوه عایق بندی کابل کواکسیال باعث شده تا این نوع کابل از نظر سرعت و مصونیت در مقابل نویز کارایی بسیار خوبی داشته باشد. پهنای باند کابل های کواکسیال به کیفیت مواد آن، طول کابل و نسبت سیگنال به نویز امواج ارسالی بستگی دارد، و در کابل های جدید به 1 GHz نیز می رسد. از کابل کواکسیال بیشتر در سیستم های تلفن راه دور استفاده می شد، که امروزه بتدریج جای خود را به فیبر های نوری می دهد. با این حال، در شبکه های شهری و تلویزیون کابلی هنوز هیچ رقیبی برای کابل کواکس وجود ندارد.

۴-۲-۲ فیبر نوری

بسیاری از افرادی که در صنعت کامپیوتر کار می کنند، از شتاب خیره کننده رشد تکنولوژی آن به خود می بالند. اولین کامپیوتر شخصی (PC) که IBM در سال ۱۹۸۱ به بازار عرضه کرد، با سرعت ساعت 4.77 MHz کار می کرد؛ بعد از گذشت بیست سال، امروزه PC ها می توانند با سرعتی متجاوز از 2 GHz کار کنند، و این یعنی ۲۰ برابر شدن سرعت در هر ۱۰ سال - چندان هم بد نیست!

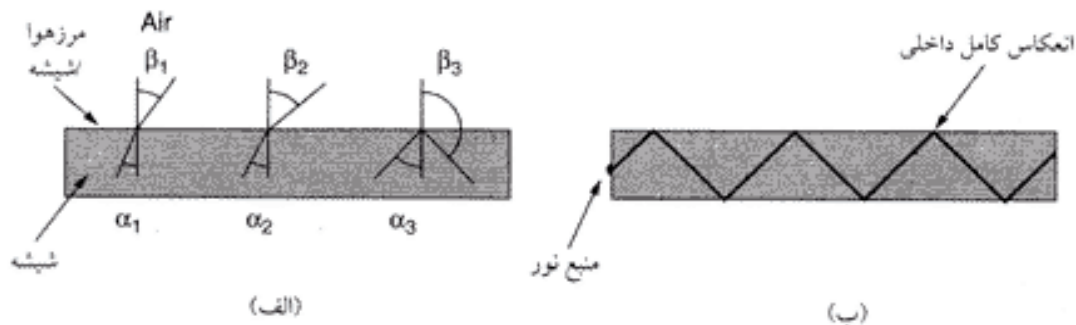
در همین دوره، سرعت مخابرات راه دور از 56-kbps (آرپانت) به 1 Gbps (مخابرات نوری جدید) رسیده است - این یعنی رشدی معادل ۱۲۵ برابر در هر دهه (در حالیکه میزان خطا هم از یک بیت در هر 10^3 بیت به تقریباً صفر رسیده).

اما پیشرفت هم مرزهای فیزیکی خاص خود را دارد؛ CPU ها مدتیست با موانعی از قبیل محدودیت سرعت نور و مشکل اتلاف گرما روبرو هستند. در حالیکه این مسائل در مخابرات وجود ندارد، و تکنولوژی فیبر نوری می تواند براحتی به پهنای باندی فراتر از 50,000 Gbps (یا 50 Tbps) دست یابد (و از هم اکنون بسیاری افراد با جدیت بدنبال تکنولوژی های بهتر نیز هستند). سرعت فعلی مخابرات فیبر نوری به چیزی در حدود 10 Gbps محدود می شود، ولی علت اصلی آن محدودیت سرعت تبدیل سیگنال های الکتریکی به پالس های نوری است، نه محدودیت ذاتی فیبر های نوری (البته در آزمایشگاهها به سرعت 100 Gbps نیز دست یافته اند).

در مسابقه بین کامپیوتر و مخابرات، مسلماً مخابرات برنده است. البته هنوز عده زیادی از دانشمندان و مهندسان کامپیوتر (که به قانون ناپکوئیست و شانون فکر می کنند، و نمی دانند که این قانونها مربوط به سیم های مسی است) نمی توانند تصور پهنای باند نامحدود را به ذهن خود راه دهند. قانون آینده باید این باشد که سرعت کامپیوترها هرگز به گرد شبکه ها نیز نخواهد رسید، و باید از هر نوع پردازشی روی اطلاعات شبکه خودداری کرد (حتی اگر به معنای تلف شدن مقداری از پهنای باند باشد). اما اجازه دهید ببینیم فیبر نوری چگونه کار می کند.

یک سیستم انتقال نوری سه مولفه کلیدی دارد: منبع نور، رسانه انتقال، آشکارساز. طبق قرارداد، یک پالس نوری معادل بیت 1 و فقدان نور معادل بیت 0 است. رسانه انتقال یک رشته (فیبر) فوق العاده نازک شیشه است. آشکارساز (detector) نیز دستگاهیست که با برخورد نور به آن یک پالس الکتریکی تولید می کند. با قرار دادن یک منبع نور در یک سر فیبر نوری و یک آشکارساز در سمت دیگر، یک سیستم انتقال نوری یکطرفه خواهیم داشت که سیگنال الکتریکی را گرفته، آنرا به پالس های نوری تبدیل کرده، و در طرف دیگر پالس دریافتی را به سیگنال های الکتریکی تبدیل می کند.

اما می دانیم که نور از شیشه خارج می شود، و چنین سیستمی بکلی بی فایده خواهد بود. اینجاست که یکی از قوانین جالب فیزیک نور به کمک ما می آید. این قانون (که به قانون شکست نور معروف است) می گوید که وقتی پرتو نور از یک محیط (مثلاً، شیشه) وارد محیط دیگر (مثلاً، هوا) می شود، در مرز این دو محیط دچار خمیدگی یا شکست (refraction) می شود. به شکل ۵-۲ (الف) نگاه کنید؛ در این شکل یک پرتو نور می بینید که با زاویه α_1 به مرز شیشه و هوا برخورد کرده، و با زاویه β_1 از شیشه خارج و وارد هوا می شود. مقدار خمیدگی یا شکست پرتو نور به خواص فیزیکی دو محیط (بویژه ضریب شکست آنها) بستگی دارد. اگر زاویه برخورد نور از یک مقدار



شکل ۲-۵. (الف) سه مثال از پرتوهای نوری که به مرز شیشه-هوا برخورد کرده، و شکسته می‌شوند. (ب) پرتو نور بدلیل شکست کلی در داخل شیشه گرفتار شده است.

بحرانی بیشتر باشد، پرتو نور دچار شکست کلی شده و دوباره به داخل شیشه برمی‌گردد، و هرگز وارد هوا نخواهد شد. نوری که با این زاویه (یا بیشتر از آن) به داخل شیشه تابانده شود، برای همیشه در آن محبوس می‌شود، و می‌تواند مسافت‌های طولانی را بدون اتلاف انرژی در فیبر نوری بپیماید - شکل ۲-۵ (ب) را ببینید. در شکل ۲-۵ (ب) فقط یک پرتو نور نشان داده شده است، ولی از آنجائیکه هر پرتو نوری که با زاویه بالاتر از زاویه بحرانی به مرز شیشه و هوا برخورد کند به داخل شیشه برمی‌گردد، در هر لحظه پرتوهای متعددی با زاویه‌های مختلف در داخل فیبر نوری به بالا و پائین حرکت می‌کنند، و اصطلاحاً گفته می‌شود که هر یک از این پرتوها دارای حالت (mode) خاص خود است. به فیبری که چنین ویژگی داشته باشد، فیبر چندحالتی (multimode fiber) گفته می‌شود.

ولی اگر قطر فیبر فقط چند برابر طول موج نور باشد، پرتو نور فقط می‌تواند در جهت مستقیم (بدون جهش به بالا و پائین) حرکت کند. به چنین فیبری که بصورت یک موج‌بر (wave guide) عمل می‌کند، فیبر تک‌حالتی (single-mode fiber) گفته می‌شود. فیبرهای تک‌حالتی گرانتر از فیبرهای چندحالتی هستند، ولی در مسافت‌های طولانی اغلب از این نوع فیبرها استفاده می‌شود. فیبرهای تک‌حالتی امروزی می‌توانند با ظرفیتهایی تا 50 Gbps و بطول 100 کیلومتر بدون نیاز به تقویت‌کننده کار کنند. (در آزمایشگاهها برای مسافت‌های کوتاهتر به پهنای باند بالاتری نیز دست یافته‌اند.)

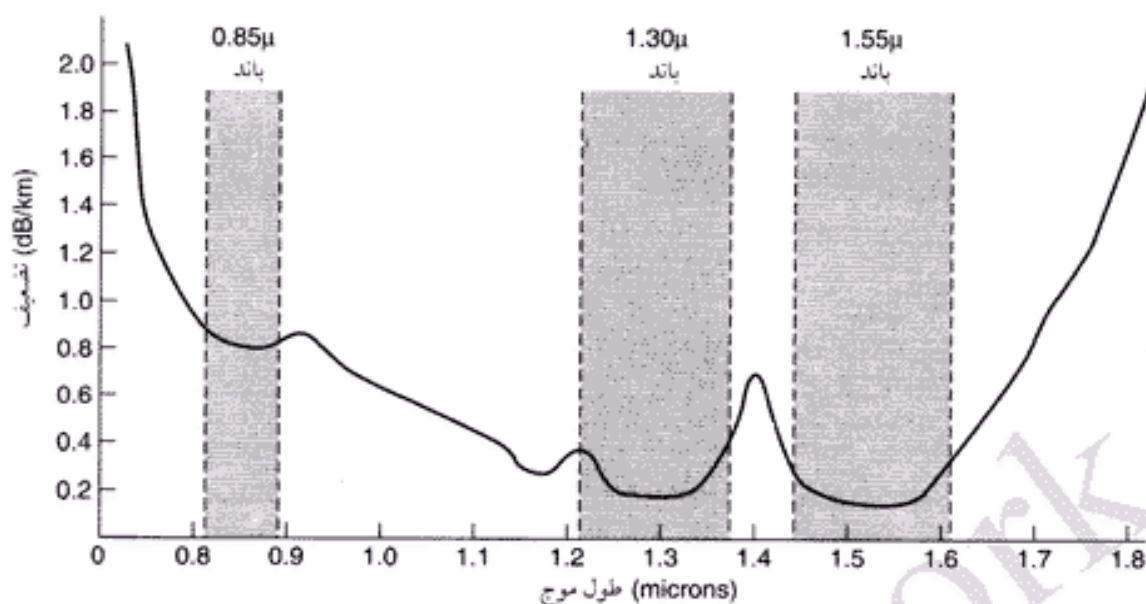
عبور نور در درون فیبر نوری

فیبرهای نوری از شیشه ساخته می‌شوند و خود شیشه هم از شن، ماده‌ای که بوفور در طبیعت یافت می‌شود. شیشه برای اولین بار در مصر باستان ساخته شد، ولی این شیشه‌ها بقدری کدر بودند که در ضخامتهای بیشتر از 1 mm نور را از خود عبور نمی‌دادند. شیشه‌ای که برای استفاده در پنجره‌ها مناسب باشد، در دوره رنسانس ساخته شد. شیشه‌ای که در ساخت فیبرهای نوری بکار می‌رود آنقدر شفاف است که اگر اقیانوس را با آن پُر کنیم، کف آن بوضوح دیده خواهد شد.

تضعیف نور (attenuation) در شیشه به طول موج آن و برخی از خواص فیزیکی شیشه بستگی دارد. در شکل ۲-۶ میزان تضعیف نور در فیبرهای نوری بر حسب دسی‌بل بر کیلومتر (خطی) نشان داده شده است. معادله تضعیف نور چنین است:

$$\text{attenuation (dB)} = 10 \log_{10} \frac{\text{transmitted power}}{\text{received power}}$$

برای مثال، اگر قدرت دریافتی در خروجی نصف قدرت ورودی باشد، میزان تضعیف سیگنال در شکل ۲-۶ نمودار تضعیف نور را در ناحیه مادون قرمز طیف، که در عمل نیز از آن $10 \log_{10} 2 = 3 \text{ dB}$ است.



شکل ۲-۶. تضعیف نور عبوری از فیبر نوری در ناحیه مادون قرمز.

استفاده می شود، می بینید. طول موج نور مرئی (0.4-0.7 میکرون، یا 400-700 نانومتر) قدری کوتاهتر از نور مادون قرمز است ($1 \mu\text{m} = 10^{-6} \text{ m}$ و $1 \text{ nm} = 10^{-9} \text{ m}$).

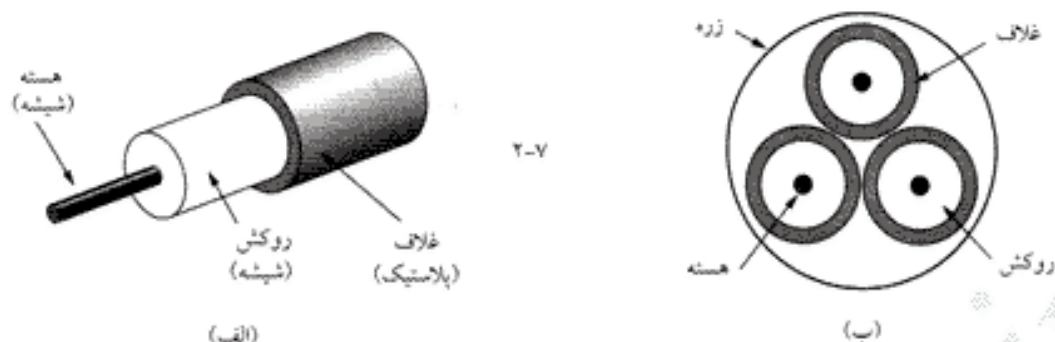
برای مخابرات نوری سه باند از طول موجهای نور مادون قرمز مورد استفاده قرار می گیرد، که بترتیب حول طول موجهای 0.85، 1.30 و 1.55 میکرون متمرکز شده اند. دو باند آخر دارای مشخصات تضعیف خوبی (کمتر از ۵ درصد در هر کیلومتر) هستند. باند 0.85 میکرون میزان تضعیف بالاتری دارد، ولی در این طول موج می توان تجهیزات نوری و الکترونیکی را از یک ماده نیمه هادی واحد (گالیوم آرسنید) ساخت. پهنای این باندها همگی بین 25,000 GHz تا 30,000 GHz است.

وقتی پالسهای نور از فیبر نوری عبور می کنند، پهنای آنها زیاد می شود، که این پدیده به پراکنش کروماتیک (chromatic dispersion) معروف است، و مقدار آن به طول موج نور بستگی دارد. یکی از راههای جلوگیری از تداخل این پالسهای پهن شده، افزایش فاصله آنهاست، و این کار نیز فقط با کاهش نرخ ارسال سیگنال ممکن است. خوشبختانه با کشف این موضوع که می توان با تولید پالسهای که تقارن کسینوس هذلولی دارند، اثر پراکنش کروماتیک را بکلی حذف کرد، امکان آن بوجود آمده که پالسهای نوری را بدون تغییر شکل محسوس تا مسافتهای هزاران کیلومتری مخابره کرد. امروزه تلاشهای زیادی در دست انجام است تا این پالسها را، که به آنها سولیتون (soliton) گفته می شود، از حالت آزمایشگاهی خارج و در عمل بکار بگیرند.

کابل فیبر نوری

کابل فیبر نوری بسیار شبیه کابل کواکسیال است، با این تفاوت که غلاف توری فلزی بیرونی را ندارد (به شکل ۲-۷ (الف) نگاه کنید). رشته شیشه ای که نور از آن عبور می کند، در مرکز کابل قرار دارد. در فیبرهای چندحالتی قطر این رشته معمولاً $50 \mu\text{m}$ (تقریباً معادل ضخامت موی سر انسان) است، و در فیبرهای تک حالتی قطر آن به 8 تا 10 میکرون می رسد.

هسته فیبر نوری با یک روکش شیشه ای (با ضریب شکست کمتر) پوشانده می شود، تا تمام نور در رشته مرکزی باقی بماند. پوشش بیرونی نیز (که اغلب پلاستیکی است) نقش محافظ را بازی می کند. معمولاً چند کابل تک رشته را در یک غلاف گرد هم می آورند (شکل ۲-۷ (ب) را ببینید).



شکل ۲-۷. (الف) نمای کناری یک فیبر منفرد. (ب) سطح مقطع کابلی با سه فیبر.

کابلهای زمینی معمولاً در عمق یک متری سطح زمین دفن می‌شوند (که در آنجا از آسیب بیلهای مکانیکی یا موشهای جوونده در امان نیستند). در قسمتهای کم عمق ساحل، کابلهای زیر دریایی را در کانالهای مخصوص پنهان می‌کنند، ولی در آبهای عمیق (که کندن کانال عملی نیست) آنها را آزاد در کف دریا رها می‌کنند (و با این کار آنها را در معرض آسیب از طرف کشتی‌های ماهیگیری و اسکوئیدهای غول پیکر قرار می‌دهند).

سه روش برای متصل کردن فیبرهای نوری وجود دارد. در روش اول، به انتهای کابل پایانه‌های مخصوص وصل کرده، و آنها را به سوکت فیبر نوری متصل می‌کنند. این پایانه‌ها ۱۰ تا ۲۰ درصد نور را تلف می‌کنند، ولی کار با آنها بسیار ساده است.

دوم اینکه، می‌توان رشته‌ها را بطور مکانیکی بهم متصل کرد. در این روش دو سر رشته‌ها را که با دقت بریده شده‌اند، در یک غلاف روبروی هم قرار می‌دهند، و آنها را در جای خود محکم می‌کنند. در این روش با عبور دادن نور و تنظیم رشته‌ها می‌توان به حداکثر سیگنال عبوری دست یافت. اتلاف نور در این روش فقط ۱۰ درصد است، و یک فرد تعلیم دیده می‌تواند در عرض ۵ دقیقه چنین اتصالی را بوجود آورد.

در روش سوم، دو سر رشته‌ها ذوب و در هم فرو برده می‌شود، تا یک اتصال یکپارچه بوجود آید. این بهترین نوع اتصال است، چون رشته‌ها در واقع یکی می‌شوند، ولی حتی در این روش هم مقداری افت توان وجود دارد. در هر سه روش فوق، محل اتصال می‌تواند مقدارن از نور را بازتابش کند، که این نور با سیگنال اصلی تداخل خواهد کرد.

تبدیل سیگنالهای الکتریکی به پالسهای نوری معمولاً به دو روش صورت می‌گیرد: لیزرهای نیمه‌هادی و LED (Light Emitting Device). هر کدام از این منابع نوری ویژگیهای خاص خود را دارند (به شکل ۲-۸ نگاه کنید). طول موج یک منبع نور را می‌توان با قرار دادن تداخل‌سنج (interferometer) هایی از نوع فابری-پروت (Fabry-Perot) یا ماخ-زندر (Mach-Zehnder) بین منبع نور و فیبر نوری تنظیم کرد. تداخل‌سنج فابری-پروت یک حفره تشدید (resonant cavity) ساده است، که از دو آینه موازی تشکیل می‌شود. نور بصورت عمود به این آینه‌ها تابانده می‌شود، و فقط طول موجهایی که مضرب صحیحی از طول حفره باشند، می‌توانند از آن خارج شوند. در تداخل‌سنج ماخ-زندر نور به دو پرتو جداگانه تقسیم می‌شود، که پس از طی مسافتی کوتاه دوباره با هم ترکیب می‌شوند؛ این دو پرتو فقط در طول موجهای خاصی با یکدیگر هم‌فاز هستند. در انتهای دیگر فیبر نوری یک فتودیود (photodiode) قرار می‌گیرد، که با هر پالس نوری یک سیگنال الکتریکی تولید می‌کند. زمان پاسخ این نوع دیودها معمولاً ۱ nsec است ($1 \text{ nsec} = 10^{-9} \text{ sec}$)، که باعث می‌شود نرخ داده‌ها به 1 Gbps محدود شود. از آنجائیکه نویز حرارتی در اینجا هم می‌تواند باعث بروز مشکل می‌شود، پالس نوری باید آنقدر انرژی داشته باشد که بتوان آنرا از نویز تشخیص داد (در فیبرهای نوری می‌توان ضریب خطا را با افزایش انرژی پالسهای نوری به میزان دلخواه پائین آورد).

لیزر نیمه هادی	LED	آیتم
زیاد	کم	سرعت داده
چند حالت یا تک حالت	چند حالت	نوع فیبر
بلند	کوتاه	فاصله
کم	زیاد	طول عمر
قابل توجه	کم	حساسیت به دما
زیاد	کم	قیمت

شکل ۲-۸. مقایسه ای بین لیزرهای نیمه هادی و LED بعنوان منبع نور.

شبکه های فیبر نوری

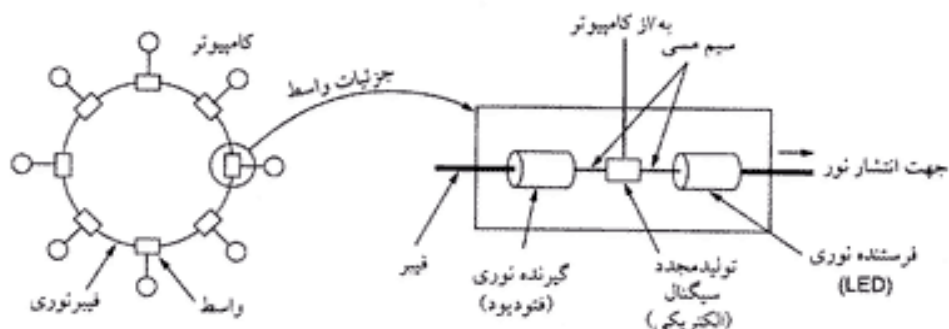
از فیبر نوری، علاوه بر مخابرات راه دور، در شبکه های محلی نیز می توان استفاده کرد - اگرچه این کار نسبت به اینترنت با مشکلات بیشتری همراه است. یک شبکه حلقوی را در نظر بگیرید (شکل ۲-۹) - این شبکه در واقع مجموعه ایست از چند اتصال نقطه-به-نقطه. هر کامپیوتر یک اتصال T (T junction) به شبکه دارد، که می تواند پالسهای نور را از خود عبور دهد، یا آنها را دریافت کند.

دو نوع اتصال وجود دارد: غیرفعال (passive) و فعال (active). در اتصال غیرفعال دو تویی به فیبر اصلی متصل می شود، که یکی LED یا دیود لیزری دارد (برای ارسال)، و دیگری فتودیود (برای دریافت). خود تویی هیچ عنصر فعالی ندارد و به همین دلیل فوق العاده قابل اعتماد است، چون خراب شدن LED یا فتودیود باعث قطعی شبکه نخواهد شد و فقط همان یک کامپیوتر از شبکه قطع می شود.

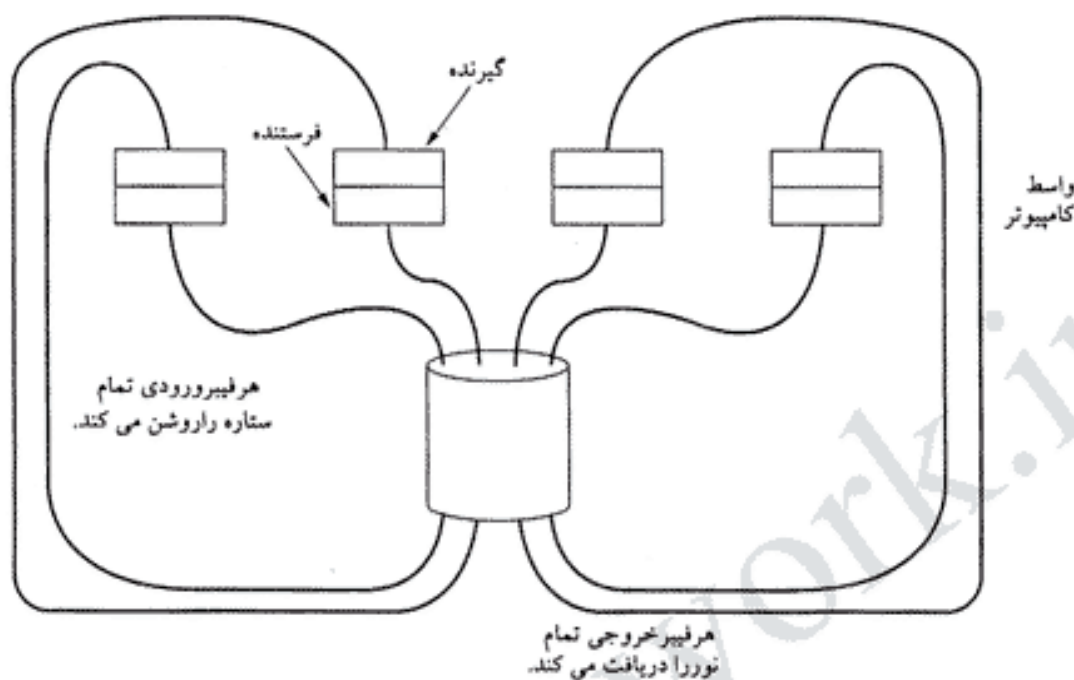
اتصال دیگری که در شکل ۲-۹ می بینید، تکرارکننده فعال (active repeater) است. در اینجا، نور ورودی ابتدا به سیگنال الکتریکی تبدیل شده، در صورت نیاز تقویت و دوباره بصورت نور منتشر می شود. اتصال به کامپیوتر توسط یک سیم مسی ساده که از تقویت کننده سیگنال منشعب شده، برقرار می شود. امروزه تقویت کننده های تمام نوری (که مرحله تبدیل به سیگنال الکتریکی - و بالعکس - در آنها حذف شده) نیز به بازار آمده اند؛ پهنای باند این تکرارکننده ها بسیار بالاست.

اگر یک تکرارکننده فعال خراب شود، حلقه شکسته شده و کل شبکه از کار می افتد. از طرف دیگر به علت تقویت سیگنال، فاصله کامپیوترها در این شبکه می تواند به چندین کیلومتر برسد، و از نظر تعداد کامپیوترها هم هیچ محدودیتی وجود ندارد. (در نوع غیرفعال، هر اتصال مقداری از توان نوری را هدر می دهد، بنابراین تعداد کامپیوترها و طول کل حلقه بشدت محدود است.)

توپولوژی حلقه تنها راه برای ایجاد شبکه های محلی با فیبر نوری نیست، و می توان به کمک سخت افزارهای خاص (انتشاردهنده های نوری) ساختاری موسوم به ستاره غیرفعال (passive star) بوجود آورد (شکل ۲-۱۰).



شکل ۲-۹. یک شبکه حلقوی فیبر نوری، با تکرارکننده های فعال.



شکل ۲-۱. شبکه‌ای با ساختار ستاره غیرفعال.

انتشاردهنده نوری یک استوانه شیشه‌ای خاص است، که فیبرهای ورودی (گیرنده) به یک طرف و فیبرهای خروجی (فرستنده) به طرف دیگر آن جوش خورده‌اند. وقتی یک کامپیوتر سیگنالی می‌فرستد، پالس نوری حاصله از طریق این استوانه در تمام فیبرهای متصل به گیرنده‌ها پخش می‌شود. در واقع، ستاره غیرفعال پالسهای ورودی را ترکیب کرده، و نور حاصله را روی تمام فیبرها می‌فرستد. از آنجائیکه انرژی نوری دریافتی در ستاره غیرفعال بین تمام رشته‌ها پخش می‌شود، تعداد کامپیوترهای متصل به این شبکه محدود (و وابسته به حساسیت فتودیودها) است.

مقایسه فیبر نوری و سیم مسی

در اینجا بد نیست مقایسه‌ای بین فیبر نوری و سیم مسی داشته باشیم؛ حتماً آموزنده خواهد بود. فیبر نوری مزایای متعددی نسبت به سیم مسی دارد. از همه مهمتر اینکه پهنای باند آن بسیار بیشتر از سیم مسی است، و همین دلیل کافیت تا در شبکه‌های پرسرعت انتخاب اول باشد. دیگر اینکه بدلیل اتلاف قدرت ناچیز فقط در هر ۵۰ کیلومتر به تکرارکننده نیاز دارد، در حالیکه این مسافت برای سیم مسی ۵ کیلومتر بیشتر نیست، که این باعث صرفه‌جویی زیادی در هزینه‌ها خواهد شد. مزیت دیگر فیبر نوری عدم تأثیرپذیری آن نسبت به نویز و تداخل‌های الکترومغناطیسی است. مقاومت عالی شیشه در مقابل خوردگی‌های شیمیایی نیز یکی دیگر از نقاط قوت فیبر نوری محسوب می‌شود.

جالب است بدانید که علت علاقه شرکت‌های تلفن به فیبر نوری اساساً چیز دیگریست: نازکی و سبکی. کانالهای زیرزمینی که شرکت‌های تلفن به نقاط مختلف حفر کرده‌اند، اکنون تا حد اشباع پُر شده‌اند، و دیگر جای خالی برای اشعاعات جدید ندارند. جایگزین کردن کابل‌های مسی با فیبر نوری، علاوه بر خالی کردن این کانالها، درآمد جدیدی نیز برای این شرکتها به همراه دارد: فروش مس خالص به پالایشگاههای مس. علاوه بر آن، فیبرهای نوری بسیار سبکتر از سیمهای مسی هستند: هر کیلومتر از کابل مسی با هزار زوج تابیده نزدیک به ۸۰۰۰ کیلوگرم وزن

دارد، در حالیکه وزن کابل فیبر نوری با همان ظرفیت فقط ۱۰۰ کیلوگرم است. کاهش وزن نیز معادل است با کاهش نیاز به وسایل مکانیکی سنگین برای نصب و پشتیبانی سیستم، بگونه ای که امروزه دیگر در مسیرهای جدید فقط از فیبر نوری استفاده می شود.

آخر اینکه، نور از فیبرهای نوری نشت نمی کند، و گرفتن انشعاب غیرمجاز از آن بسیار مشکل است. این ویژگی باعث شده تا فیبر نوری از ایمنی بسیار بالایی در مقابل سارقان اطلاعات برخوردار باشد.

البته فیبر نوری چندان بی عیب و نقص هم نیست. اول اینکه، این تکنولوژی هنوز بسیار جدید است و حتی بسیاری از مهندسين نیز توانایی کار با آن را ندارند. دیگر اینکه فیبر نوری بسیار آسیب پذیرتر از سیم مسی است، و حتی خم کردن بیش از حد باعث خرابی آن می شود. از آنجائیکه انتقال نوری ذاتاً یک طرفه است، برای ارتباط دوطرفه باید از دو رشته فیبر (با یک رشته فیبر با دو باند فرکانسی) استفاده کرد. دست آخر اینکه، تجهیزات ارتباطی نوری گرانتر از انواع الکتریکی آن است. با این وجود، آینده مخابرات در فواصلی حتی بیش از چند متر از آن فیبرهای نوری است. برای بحثی جامع درباره جنبه های مختلف فیبر نوری و شبکه های آن، به (Hecht, 2001) مراجعه کنید.

۳-۲ انتقال بیسیم

یکی از پدیده های عصر ما معنادار اینترنتی است: کسانی که می خواهند بیست و چهار ساعته بر خط (on-line) باشند. برای این قبیل افراد (که دائماً در حال جابجا شدن هستند) دیگر زوج نابیده، کابل کوکس و فیبر نوری کاربرد ندارد. آنها می خواهند بدون مقید شدن به هیچیک از سیستمهای مخابراتی دست و پا گیر، اطلاعات مورد نیازشان را روی کامپیوتر کیفی، جیبی و حتی مچی (I) دریافت کنند؛ و مخابرات بیسیم تنها چیزیست که می تواند توقعات اینها را برآورده کند. در این قسمت نگاهی کلی به مخابرات بیسیم خواهیم انداخت، نه فقط بخاطر اینکه عده ای دوست دارند کنار دریا هم از دوستان اینترنتی شان جدا نشوند، بلکه به دلیل آنکه این تکنولوژی نقش مهمی در زندگی امروزی ما بازی می کند.

برخی افراد معتقدند که در آینده فقط دو نوع مخابرات وجود خواهد داشت: فیبر نوری و بیسیم. تمام تجهیزات ثابت (کامپیوترهای رومیزی، تلفن و فکس) از فیبر نوری، و تجهیزات متحرک از بیسیم بهره خواهند گرفت. حتی در مواردی می توان از بیسیم برای تجهیزات ثابت استفاده کرد. مثلاً، اگر کابل کشی به یک ساختمان (بعلت وجود موانع طبیعی مانند کوه، جنگل و باتلاق) مشکل باشد، استفاده از بیسیم ترجیح دارد. همانطور که قبلاً هم گفتیم، مخابرات بیسیم دیجیتال برای اولین بار در مجمع الجزایر هاوایی بکار گرفته شد.

۱-۳-۲ طیف الکترومغناطیس

امواج الکترومغناطیس حاصل حرکت الکترونها هستند، که می توانند در فضا (حتی در خلأ) منتشر شوند. وجود چنین امواجی اولین بار بصورت تئوری در سال ۱۸۶۵ توسط فیزیکدان انگلیسی جیمز کلرک ماکسول پیش بینی شد، و در سال ۱۸۸۷ فیزیکدان آلمانی هاینریش هرتز موفق شد آنها را مشاهده کند. تعداد نوسانهای یک موج در ثانیه فرکانس، f ، نامیده می شود، و واحد آن (به افتخار هاینریش هرتز) هرتز (Hz) نامگذاری شده است. فاصله بین دو قله (یا حضیض) متوالی موج را نیز طول موج می گویند، و آنرا با حرف یونانی λ نشان می دهند.

اگر آنتنی با اندازه مناسب به یک مدار الکتریکی وصل شود، می تواند امواج الکترومغناطیسی منتشر کند، که این امواج را می توان در فواصل مناسب دریافت کرد. تمام سیستمهای مخابرات بیسیم بر این اساس کار می کنند.

امواج الکترومغناطیسی (صرفنظر از فرکانس آنها) در خلأ با سرعت ثابتی (سرعت نور) حرکت می کنند، که مقدار تقریبی آن 3×10^8 m/sec است، و با c نشان داده می شود (سرعت نور حد در طبیعت است، یعنی

هیچ چیز نمی تواند سریعتر از آن حرکت کند.) سرعت این امواج در سیم مسی یا فیبر نوری به حدود $\frac{2}{3}c$ کاهش می یابد، که تا حدی نیز به فرکانس وابسته است.

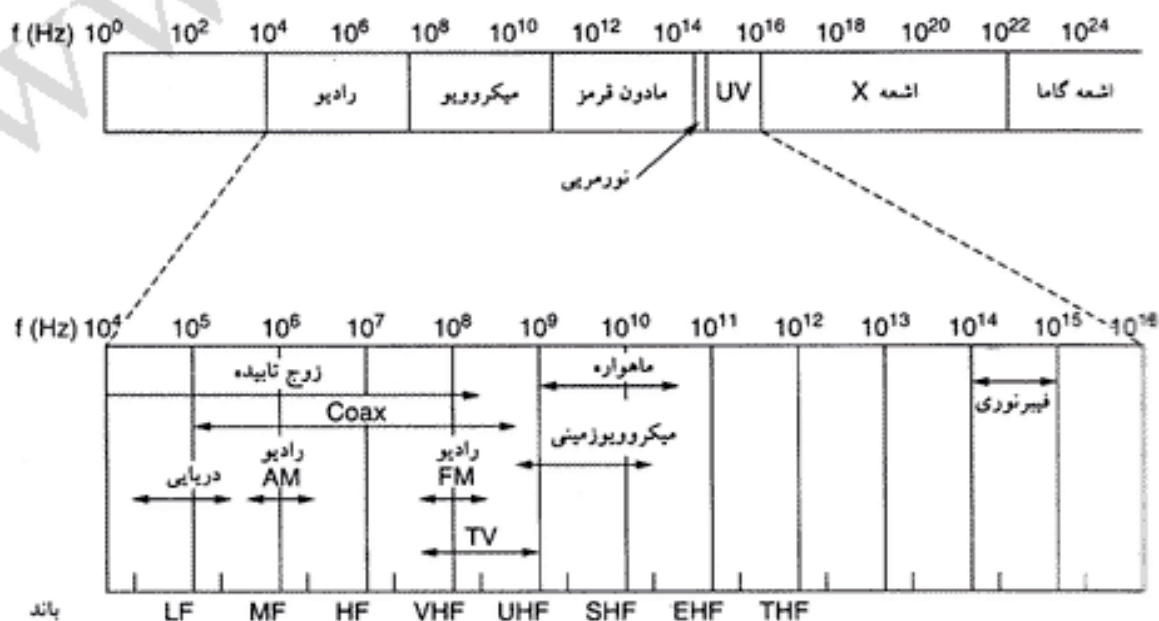
رابطه اساسی بین f ، λ و c (در خلأ) چنین است:

$$\lambda f = c \quad (2-2)$$

از آنجائیکه c ثابت است، با داشتن f می توان λ را محاسبه کرد، و بالعکس. اگر λ را برحسب متر و f را برحسب MHz داشته باشیم، رابطه بالا بصورت $300 \approx \lambda f$ در می آید. برای مثال، در فرکانس 100-MHz طول موج 3 m است، که در فرکانس 1000-MHz طول موج آن به 0.3 m کاهش می یابد؛ موجی با طول موج 0.1 m فرکانسی معادل 3000-MHz (در خلأ) دارد.

در شکل ۱۱-۲ طیف الکترومغناطیس را ملاحظه می کنید. از ناحیه های رادیویی، مایکروبو، مادون قرمز، و مرئی این طیف می توان (با مدولاسیون دامنه، فرکانس، و فاز) برای انتقال اطلاعات استفاده کرد. نور ماوراء بنفش، اشعه X و اشعه گاما بدلیل فرکانس بالاتر برای منظور بهتر هستند، ولی از آنجائیکه تولید و مدولاسیون آنها مشکل است، از اجسام غیر شفاف عبور نمی کنند، و برای سلامتی موجودات زنده خطرناکند، کاربرد مخابراتی ندارند. باندهایی که در پائین شکل ۱۱-۲ می بینید، رسماً توسط ITU (برحسب طول موج) تقسیم بندی و نامگذاری شده اند. اصطلاحات LF، MF و HF بترتیب معادل فرکانس پائین، فرکانس متوسط و فرکانس بالا هستند. پیداست وقتی این نامگذاریها انجام می شد، کسی تصور فرکانسهای بالاتر از 10 MHz را هم نمی کرد، و بهمین دلیل امروزه برای باندهای بالاتر از آن از نامهای VHF (فرکانس خیلی بالا)، UHF (فرکانس بسیار بالا)، SHF (فرکانس فوق العاده بالا)، EHF (فرکانس شدیداً بالا)، و THF (فرکانس خارق العاده بالا) استفاده می شود - خوشبختانه برای فرکانسهای بالاتر هنوز نامی انتخاب نشده، و شما هم می توانید ذوق خود را در این زمینه امتحان کنید.

مقدار اطلاعاتی که یک موج الکترومغناطیس می تواند حمل کند، به پهنای باند آن بستگی دارد. با تکنولوژی امروزی، می توان در فرکانسهای پائین بازای هر هرتز از پهنای باند ۲ تا ۳ بیت، و در فرکانسهای بالا تا ۸ بیت



شکل ۱۱-۲. طیف الکترومغناطیس و کاربردهای مخابراتی آن.

اطلاعات منتقل کرد، بنابراین یک کابل کواکسیال با پهنای باند 750 MHz می تواند در هر ثانیه تا چندین گیگابایت اطلاعات منتقل کند. با یک نگاه به شکل ۲-۱۱ می توان براحتی دریافت که چرا فیبر نوری این همه طرفدار دارد.

اگر معادله (2-2) را برای f حل کرده و سپس از آن نسبت به λ دیفرانسیل بگیریم، خواهیم داشت

$$\frac{df}{d\lambda} = -\frac{c}{\lambda^2}$$

حال اگر بجای دیفرانسیل تفاضل محدود و قدر مطلق مقادیر را در نظر بگیریم، داریم

$$\Delta f = \frac{c\Delta\lambda}{\lambda^2}$$

بنابراین با داشتن پهنای یک باند، $\Delta\lambda$ ، می توان فرکانس متناظر با آن باند، Δf ، و از آنجا نرخ داده آن، را محاسبه کرد. هر چه پهنای یک باند بزرگتر باشد، نرخ انتقال داده آن باند بیشتر است. برای مثال، باند $1.30 \mu\text{m}$ را در شکل ۲-۶ در نظر بگیرید. در اینجا، $\lambda = 1.3 \times 10^{-6} \text{ m}$ و $\Delta\lambda = 0.17 \times 10^{-6} \text{ m}$ است، بنابراین Δf تقریباً 30 THz خواهد شد. با چنین پهنای باندی (و با فرض 8 bits/Hz) می توان به نرخ انتقال داده 240 Tbps دست یافت. در اکثر سیستمهای انتقال از باندهای فرکانس باریک استفاده می شود (یعنی، $\frac{\Delta f}{f} \ll 1$)، تا گیرنده بتواند بیشترین توان را دریافت کند. اما گاهی نیز استفاده از باند فرکانسی وسیع ضروریست، که بر دو نوع است. در نوع طیف گسترده با پرش فرکانسی (frequency hopping spread spectrum)، فرستنده در هر ثانیه صدها بار فرکانس خود را عوض می کند (بعبارت دیگر از فرکانسی به فرکانس دیگر می پرد). این تکنیک بویژه در مخابرات نظامی کاربرد دارد، چون تشخیص فرکانس فرستنده بسیار مشکل است و پارازیت انداختن روی آن نیز تقریباً غیرممکن است. در این روش تضعیف موج در اثر انعکاس نیز وجود ندارد، چون وقتی موج انعکاسی (کمی بعد از موج اصلی) به گیرنده می رسد، فرکانس آن تغییر کرده است و فرکانسهای قبلی را دیگر قبول نمی کند. در سالهای اخیر این تکنیک در محصولات تجاری هم کاربرد پیدا کرده است - برای مثال، بلوتوث و 802.11 هر دو از این تکنیک استفاده می کنند.

اختراع این تکنیک نیز داستانی جالب دارد، که شنیدن آن خالی از لطف نیست. یکی از دو مخترع تکنیک طیف گسترده با پرش فرکانسی، خانم هدی لامار هنرپیشه اتریشی الاصل است. شوهر اول این خانم، که سازنده تسلیحات نظامی بود، برای وی توضیح داد که چگونه می توان با استفاده از سیگنالهای رادیویی از درها و موشکها را کنترل کرد. وقتی خانم لامار فهمید که شوهرش به هیتلر اسلحه می فروشد، دچار وحشت شد، با لباس مبدل گریخت، و به هالیوود رفت تا به حرفه مورد علاقه اش یعنی هنرپیشگی ادامه دهد. لامار به کمک دوستش جورج آنتیل (که یک آهنگساز بود) و برای کمک به متفقی، تکنیک پرش فرکانسی را اختراع کرد. طرح اولیه آنها دارای ۸۸ فرکانس (به تعداد کلیدهای پیانو) بود، که به شماره 2,292,387 در اداره ثبت اختراعات ایالات متحده آمریکا به ثبت رسید. با این حال، آنها نتوانستند مفید و عملی بودن این اختراع را به نیروی دریایی آمریکا بقبولانند، و هرگز پولی بابت آن دریافت نکردند. تنها سالها پس از سپری شدن از حق الاختراع آن بود که این تکنیک مورد توجه محافل علمی قرار گرفت.

نوع دوم باندهای فرکانسی وسیع، طیف گسترده با توالی مستقیم (direct sequence spread spectrum) نام دارد، که در آن سیگنال روی طیف وسیعی از فرکانسها پخش می شود. این تکنیک نیز امروزه، بویژه در تلفنهای همراه نسل دوم، کاربردهای تجاری پیدا کرده است، و با آمدن نسل سوم تلفنهای همراه تسلط آن بر بازار کامل خواهد شد، زیرا دارای کارایی طیفی خوب، مصونیت در برابر نویز عالی و بسیاری ویژگیهای دیگر است. در برخی از شبکه های محلی بیسیم نیز از تکنیک طیف گسترده با توالی مستقیم استفاده شده است. در قسمتهای آینده همین

فصل باز هم به مبحث طیف گسترده برمی گردیم؛ اگر به تکنیکهای مخابرات طیف گسترده و تاریخچه آن علاقمند هستید، کتاب (Scholtz, 1982) را جالب خواهید یافت.

فعالاً فرض را بر این می گذاریم که همه جا از باند فرکانسی باریک استفاده می شود، و بحث خود را با کاربرد بخشهای مختلف طیف الکترومغناطیس شکل ۲-۱۱ ادامه می دهیم - از رادیو شروع می کنیم.

۲-۳-۲ مخابرات رادیویی

امواج رادیویی کاربرد گسترده ای در مخابرات (در فضاهای سرپوشیده یا باز) دارند، چون باسانی می توان آنها را تولید کرد، بُرد زیادی دارند، و از ساختمانها و موانع عبور می کنند. امواج رادیویی همه-طرفه هستند، یعنی در تمام جهات منتشر می شوند، بنابراین نیازی به تنظیم دقیق موقعیت گیرنده و فرستنده نسبت بیکدیگر نیست.

البته همه-طرفه بودن امواج رادیویی همیشه هم خوب نیست. در دهه ۱۹۷۰، شرکت جنرال موتورز تصمیم گرفت اتومبیلهای کادیلاک جدید خود را به ترمز ضد-قفل کامپیوتری (ABS) مجهز کند. در این سیستم وقتی راننده ترمز می گیرد، کامپیوتر مرکزی پالسهای به ترمزها می فرستد، که آنها را چندین بار در ثانیه باز و بسته می کند، و ترمزها دیگر قفل نمی کنند. مدتی بعد در یک روز آفتابی، یکی از پلیسهای گشت بزرگراه ایالت اوهایو تصمیم گرفت رادیوی بیسیم جدید خود را امتحان کرده و با مرکز فرماندهی تماس بگیرد. با این کار، اتومبیل کادیلاکی که در نزدیکی وی حرکت می کرد، بلافاصله مثل یک اسب چموش شروع به حرکات عجیبی کرد. وقتی افسر پلیس اتومبیل خاطی را متوقف کرد، راننده ادعا کرد که هیچ کاری انجام نداده و اتومبیل بیکپاره دیوانه شده است.

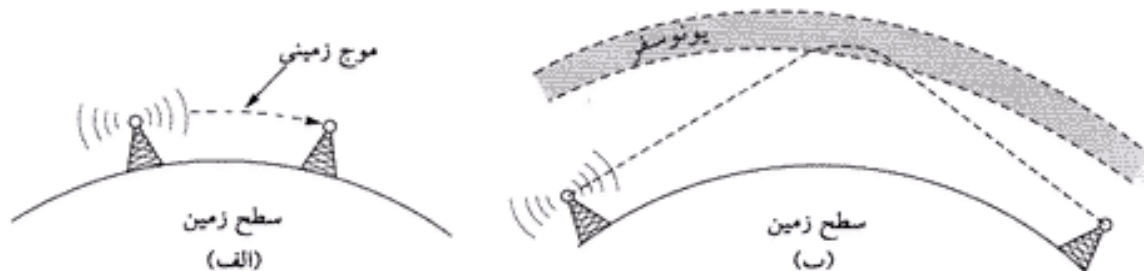
کمی بعد ماجرا روشتتر شد: اتومبیلهای کادیلاک همه جا خوب کار می کردند، و فقط در بزرگراههای ایالت اوهایو، آن هم وقتی پلیس می دیدند، دیوانه می شدند. تا مدتها جنرال موتورز سرگردان بود و نمی توانست بفهمد چرا کادیلاک در همه جا جز بزرگراههای ایالت اوهایو خوب کار می کند (کادیلاک حتی در خیابانها و جاده های معمولی اوهایو هم خوب کار می کرد). فقط بعد از تحقیقات گسترده بود که معلوم شد، مدارهای سیمکشی کادیلاک برای فرکانس سیستم رادیویی جدید پلیس بزرگراه اوهایو تبدیل به یک آنتن خوب می شود، و پالسهای این رادیو سیستم ABS را فعال می کند.

ویژگیهای امواج رادیویی به فرکانس آنها وابسته است. در فرکانسهای پائین، امواج براحتی از موانع عبور می کنند، ولی توان آنها بر حسب فاصله بر سرعت افت می کند (با ضریب $1/r^2$). در فرکانسهای بالا، امواج رادیویی به خط مستقیم حرکت می کنند، و در برخورد با موانع منعکس می شوند. حتی قطرات باران هم امواج با فرکانس بالا را جذب می کند. امواج رادیویی در تمام فرکانسها در معرض تداخل ناشی از کارکرد وسایل الکتریکی (مانند موتور) هستند.

بدلیل بُرد زیاد امواج رادیویی، خطر تداخل آنها در سرتاسر دنیا وجود دارد، بهمین دلیل دولتها نحوه استفاده از فرستنده های رادیویی را (البته با یک استثنا) پشدت کنترل می کنند.

امواج رادیویی در باندهای VLF، LF و MF از انحنای زمین تبعیت می کنند - به شکل ۲-۱۲ (الف) نگاه کنید. این امواج نزدیک به ۱۰۰۰ کیلومتر بُرد دارند. ایستگاههای رادیویی AM در باند MF کار می کنند، و بهمین دلیل بُرد آنها چنین زیاد است. امواج رادیویی در این باند براحتی از موانع و ساختمانها عبور می کنند، و علت دریافت آنها در داخل ساختمانها نیز همین است. تنها مشکل این امواج پهنای باند کم آنهاست (معادله ۲-۳ را ببینید).

در باندهای HF و VHF، آن قسمت از امواج رادیویی که نزدیک سطح زمین حرکت می کنند، جذب زمین می شوند. ولی بخشی از امواج که به سمت فضا می روند، پس از برخورد با یونوسفر (ionosphere - لایه ای از جو زمین به ارتفاع ۱۰۰ تا ۵۰۰ کیلومتر، که حاوی ذرات باردار است) به طرف زمین برمی گردند؛ شکل ۲-۱۲ (ب) را



شکل ۲-۱۲. (الف) در باندهای VLF، LF و MF امواج رادیویی از انحناي زمین تبعیت می کنند. (ب) در باندهای HF و VHF آنها بین زمین و یونسفر رفت و برگشت می کنند.

نگاه کنید. در شرایط خاصی این امواج می توانند چندین بار بین زمین و یونسفر رفت و برگشت کرده، و صدها کیلومتر دورتر دریافت شوند. طرفداران رادیو آماتوری و مخابرات نظامی از باندهای HF و VHF استفاده می کنند.

۳-۳-۲ مخابرات مایکروویو

در فرکانسهای بالای 100 MHz امواج تقریباً به خط مستقیم حرکت می کنند، و می توان آنها را دقیقاً روی یک نقطه متمرکز کرد. متمرکز کردن تمام انرژی موج در یک پرتو باریک (با استفاده از آنتنهای بشقابی) باعث بالا رفتن نسبت سیگنال به نویز می شود، ولی از طرف دیگر لازم است تا گیرنده و فرستنده بدقت تنظیم شوند. علاوه بر آن، باریک بودن پرتوها باعث می شود تا تداخل فرستنده ها به حداقل برسد، مشروط بر اینکه کمی با هم فاصله داشته باشند. تا قبل از اختراع فیبر نوری، این امواج (که به مایکروویو معروفند) ستون فقرات مخابرات راه دور محسوب می شدند. در واقع، یکی از شرکتهای رقیب AT&T بنام MCI شبکه گسترده ای از برجهای مایکروویو با فواصل دهها کیلومتر ایجاد کرده بود (حتی نام این شرکت - Microwave Communications, Inc. - هم بنوعی گویای استفاده از امواج مایکروویو است). MCI مدتهاست به فیبر نوری روی آورده، و اکنون در WorldCom ادغام شده است.

از آنجائیکه امواج مایکروویو به خط مستقیم حرکت می کنند، اگر فاصله ایستگاه مبدأ و مقصد زیاد باشد، انحناي زمین مانع از رسیدن امواج خواهد شد. به همین دلیل لازم است تا در فاصله بین آنها از برجهای تکرارکننده استفاده شود. هر چه ارتفاع این برجها بیشتر باشد، فاصله آنها می تواند زیاده تر باشد. فاصله دو ایستگاه تقریباً با توان دوم ارتفاع آنها رابطه دارد؛ برای مثال، دو برج ۱۰۰ متری می توانند چیزی در حدود ۸۰ کیلومتر فاصله داشته باشند. امواج مایکروویو، برخلاف امواج رادیویی فرکانس پائین، نمی توانند بخوبی از موانع عبور کنند. علاوه بر این، با وجود متمرکز شدن موج در فرستنده، امواج مایکروویو در طول مسیر دچار پراکندگی جزئی می شوند. قسمتی از موج مایکروویو نیز که توسط لایه های پائین جو منعکس می شود، فاصله بیشتری را طی کرده و هنگام رسیدن به گیرنده دیگر با موج اصلی هم فاز نیست، و باعث خنثی شدن آن می شود. این پدیده که به محوشدگی چندمسیره (multipath fading) معروف است، یکی از مشکلات جدی در مخابرات رادیویی محسوب می شود، و به وضع هوا و فرکانس موج بستگی دارد. در برخی از موارد، ۱۰ درصد ظرفیت کانال برای مقابله با این وضعیت کنار گذاشته می شود، تا در صورت بروز محوشدگی چندمسیره بتواند موقتاً از فرکانسهای جایگزین استفاده کنند.

تقاضا برای پهنای باند بیشتر باعث شده تا فرکانسها هر روز بالا و بالاتر برود. امروزه باندهای 10 GHz نیز در حال کار هستند، ولی از فرکانس 4 GHz به بالا مشکل جدیدی خودنمایی می کند: جذب شدن انرژی امواج توسط

آب. طول موج این قبیل امواج فقط چند سانتی متر است، و براحتی جذب قطرات باران می شوند. این پدیده شاید برای کسانی که می خواهند یک اجاق مایکروویو بزرگ بسازند و پرندگان را در هوا کباب کنند خوب باشد، ولی برای مخابرات مایکروویو یک مشکل جدی است. تنها راه حل این مشکل (مانند محوشدگی چندمسیره) قطع کردن کانالهای باران زده، و استفاده از کانالهای جایگزین است.

خلاصه اینکه، استفاده از امواج مایکروویو در مخابرات راه دور، تلفنهای همراه، و تلویزیون چنان گسترش یافته، که امروزه دیگر در طیف مایکروویو جایی برای استفاده بیشتر نمانده است. مخابرات مایکروویو مزایای زیادی نسبت به فیبر نوری دارد. اول اینکه نیازی به تصرف زمین برای حفر کانال و کشیدن کابل نیست، و با خرید چند تکه زمین کوچک در فواصل ۵۰ کیلومتری و نصب چند برج مایکروویو، می توان یک سیستم مخابراتی مستقل ایجاد کرد. بهمین خاطر بود که MCI توانست سرعت بعنوان یکی از غولهای صنعت تلفن راه دور قد علم کند. (یکی دیگر از شرکتهای بزرگ مخابراتی یعنی Sprint، مسیر کاملاً متفاوتی در پیش گرفت: این شرکت که متعلق به راه آهن پاسیفیک جنوبی بود، از زمینهای حاشیه خطوط آهن استفاده کرده، و کابلهای خود را در آنجا دفن کرد.)

دیگر اینکه، مایکروویو نسبتاً ارزان است. نصب چند برج ساده برای نصب آنتنها (که می تواند فقط یک دکل ساده با چهار سیم مهارکننده باشد)، بسیار ارزانتر از خرید زمین در مناطق شلوغ شهری یا صعب العبور کوهستانی (و یا حتی اجاره خطوط تلفنی از شرکتهای تلفن) است.

طیف الکترومغناطیس و سیاست

برای جلوگیری از هرج و مرج و آشوب، توافقهایی در سطح ملی و بین المللی برای نحوه استفاده از فرکانسها شکل گرفته است. از آنجائیکه هر کس پهنای باند بیشتری می خواهد، دولتها مجبور به دخالت هستند، و پهنای باند لازم برای رادیوی AM و FM، تلویزیون، تلفنهای همراه، شرکتهای تلفن، پلیس، ناوبری هوایی و دریایی، مصارف نظامی و دولتی (و خلاصه، همه آنها) که فرکانس می خواهند را به آنها اختصاص می دهند. در سطح بین المللی نیز یکی از شعبات ITU-R (بنام WARC) به هماهنگی این اقدامات اختصاص یافته، تا کارکرد وسایل مخابراتی را در کشورهای مختلف تضمین کند. با این همه، توصیه های ITU-R الزام آور نیست، و گاهی پیش می آید که FCC (Federal Communication Commission - سازمانی که مسئول تخصیص باندهای فرکانسی در ایالات متحده آمریکا است) توصیه های آنرا نادیده بگیرد، تا (مثلاً) یک طیف فرکانسی را به یک گروه قدرتمند سیاسی اختصاص دهد.

علاوه بر اختصاص قسمتی از طیف فرکانسی به یک کاربرد خاص (مثلاً، تلفن همراه)، باید مشخص شود که کدام حامل (carrier) ها مجاز به استفاده از این فرکانسها هستند. در گذشته، برای این منظور سه روش کاربرد گسترده تری داشت. در قدیمی ترین آنها، که به مسابقه زیبایی معروف بود، هر حامل بایستی توضیح می داد که چرا پیشنهاد آنها بیشترین نفع را برای جامعه دارد - و این مقامات دولتی بودند که بهترین پیشنهاد را انتخاب می کردند. از آنجائیکه انتخاب یک حامل منافع میلیاردری برای آن در بر داشت، در این روش رشوه، فساد و پارتی بازی پیداد می کرد. علاوه بر آن، حتی وقتی یک کارمند وظیفه شناس تشخیص می داد که پیشنهاد یک شرکت خارجی بهتر از شرکتهای داخلی است، قبولاندن آن به مقامات بالاتر کار ساده ای نبود.

مشکلات روش انتخاب دولتی، به روش دوم منجر شد: قرعه کشی بین حامل های متقاضی یک طیف فرکانسی. مشکل این روش آن بود که حتی شرکتهایی که هیچ نفعی در استفاده از طیف نداشتند، می توانستند در قرعه کشی شرکت کنند. بدین ترتیب حتی صاحب یک رستوران یا کفش فروشی هم می توانست برنده قرعه کشی شده، و بعد با خیال راحت و بدون هیچ خطری امتیاز خود را به شرکتهای ذینفع بفروشد و سود کلانی به جیب بزند.

بخشیدن چنین ثروتهای بادآورده ای به افراد زرنگ و هوشیار، باعث انتقادات زیادی شد، و منجر به اتخاذ روش سوم اعطای امتیاز حامل طیفهای فرکانسی گردید: حراج کردن پهنای باند. وقتی انگلستان در سال ۲۰۰۰ طیف فرکانسی تلفنهای همراه نسل سوم را به حراج گذاشت، پیش بینی می کرد که از این راه چهار میلیارد دلار عایدی داشته باشد. ولی در واقع از این مزایده چهل میلیارد دلار سود برد، چون شرکتهای زیادی (از ترس اینکه از قافله تلفن همراه عقب بمانند) به تکاپوی خرید طیف فرکانسی افتادند. این حادثه طمع سایر دولتها را تحریک کرد، و آنها هم به فکر راه انداختن مزایده فروش طیف فرکانسی افتادند. شرکتهای بسیاری به همین دلیل تا مرز ورشکستگی پیش رفتند، و آنهایی که ماندند باید سالها تلاش کنند تا پول از دست رفته را جبران کنند.

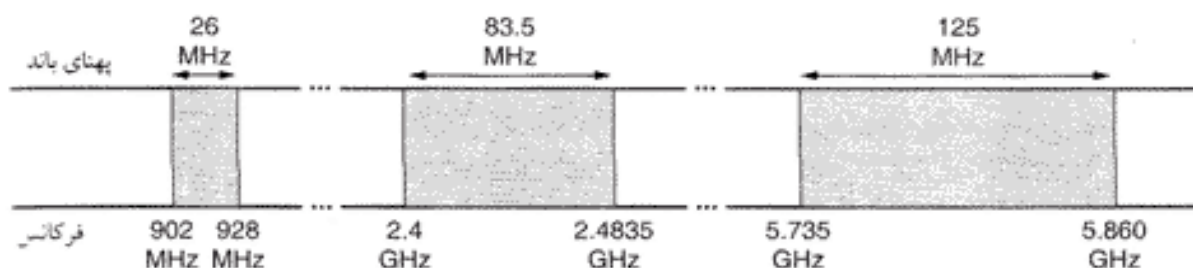
روش دیگر تخصیص فرکانس این است که اصلاً آنرا به کسی اختصاص ندهند: اجازه دهید هر کس با هر فرکانسی که می خواهد کار کند، فقط توان فرستنده های آنها را بگونه ای کنترل کنید که بُرد کمی داشته باشند، و با فرستنده های دیگر تداخل نکنند. بسیاری از دولتها بخشی از طیف فرکانسی خود را (که به باند ISM : صنعتی، علمی، پزشکی معروف است) با همین نیت آزاد می گذارند تا هر کس که مایل است از آنها استفاده کند. در باز کن های برقی، تلفنهای بیسیم خانگی، اسباب بازی های کنترل از راه دور، و بسیاری از وسایل خانگی مجهز به کنترل رادیویی از باندهای ISM استفاده می کنند. برای به حداقل رساندن تداخل بین این دستگاهها، FCC و سایر دولتها سازندگان این قبیل وسایل را مجبور می کنند تا از تکنیکهای طیف گسترده استفاده کنند.

محل طیف ISM از کشوری به کشور دیگر متفاوت است. برای مثال، در ایالات متحده آمریکا وسایلی با توان تشعشعی کمتر از ۱ وات می توانند (بدون نیاز به اخذ مجوز FCC) از باندهای مشخص شده در شکل ۲-۱۳ استفاده کنند. باند 900-MHz بهترین کارایی را دارد، ولی علاوه بر اینکه خیلی شلوغ است، در تمام دنیا هم آزاد نیست. باند 2.4-GHz در اغلب کشورهای دنیا آزاد است، ولی در معرض تداخل امواج اجاقهای مایکروویو و تأسیسات رادار قرار دارد. بلوتوث و برخی از شبکه های محلی 802.11 در این باند کار می کنند. باند 5.7-GHz نسبتاً جدید است و وسایل آن هنوز گران هستند، ولی از آنجائیکه استاندارد 802.11a در این باند کار می کند، بزودی شاهد رواج آن خواهیم بود.

۲-۳ امواج مادون قرمز و میلیمتری

امواج هدایت نشده مادون قرمز و میلیمتری کاربرد زیادی در مخابرات بُرد کوتاه دارند. دستگاههای کنترل از راه دور وسایل صوتی و تصویری همگی از امواج مادون قرمز استفاده می کنند. این دستگاهها جهت دار، ارزان و ساده هستند، ولی یک عیب عمده دارند: امواج مادون قرمز از اجسام صلب عبور نمی کند (بین دستگاه کنترل از راه دور و تلویزیون خود بایستید، و ببینید هنوز کار می کند یا نه). در حالت کلی، هر چه به سمت فرکانسهای طیف مرئی نور برویم، امواج بیشتر شبیه نور (و کمتر شبیه امواج رادیویی) عمل می کنند.

از طرف دیگر، عبور نکردن امواج مادون قرمز از موانع سخت یک مزیت محسوب می شود: شما که نمی خواهید تلویزیونتان با کنترل از راه دور همسایه روشن و خاموش شود. استراق سمع امواج مادون قرمز نیز



شکل ۲-۱۳. باندهای ISM در ایالات متحده آمریکا.

مشکلاتر از امواج رادیویی است، و بهمین دلیل از ایمنی بالاتری برخوردار است. برخلاف امواج رادیویی، استفاده از امواج مادون قرمز هیچ نیازی به کسب مجوز از مقامات رسمی ندارد. با وجود برخی کاربردهای امواج مادون قرمز در صنعت کامپیوتر (مانند اتصال ماوس و چاپگر)، این امواج در صحنه مخابرات حرفه‌ای برای گفتن ندارند.

۵-۳-۲ مخابرات امواج نوری

ارسال علائم نوری قرنهایست که شناخته شده و بکار برده می‌شود. امروزه برای متصل کردن دو شبکه که در ساختمانهای جداگانه قرار دارند، از لیزرهایی که روی پشت بام ساختمانها قرار می‌گیرند، استفاده می‌شود. پرتوهای لیزر اساساً یکطرفه هستند، بنابراین هر ساختمان باید فرستنده و گیرنده لیزری جداگانه داشته باشد. پهنای باند این روش بسیار بالا و هزینه آن نیز بسیار پائین است، و علاوه بر اینکه نصب ساده‌ای (نسبت به تجهیزات مایکروویو) دارد، نیازی به مجوز FCC نیز ندارد.

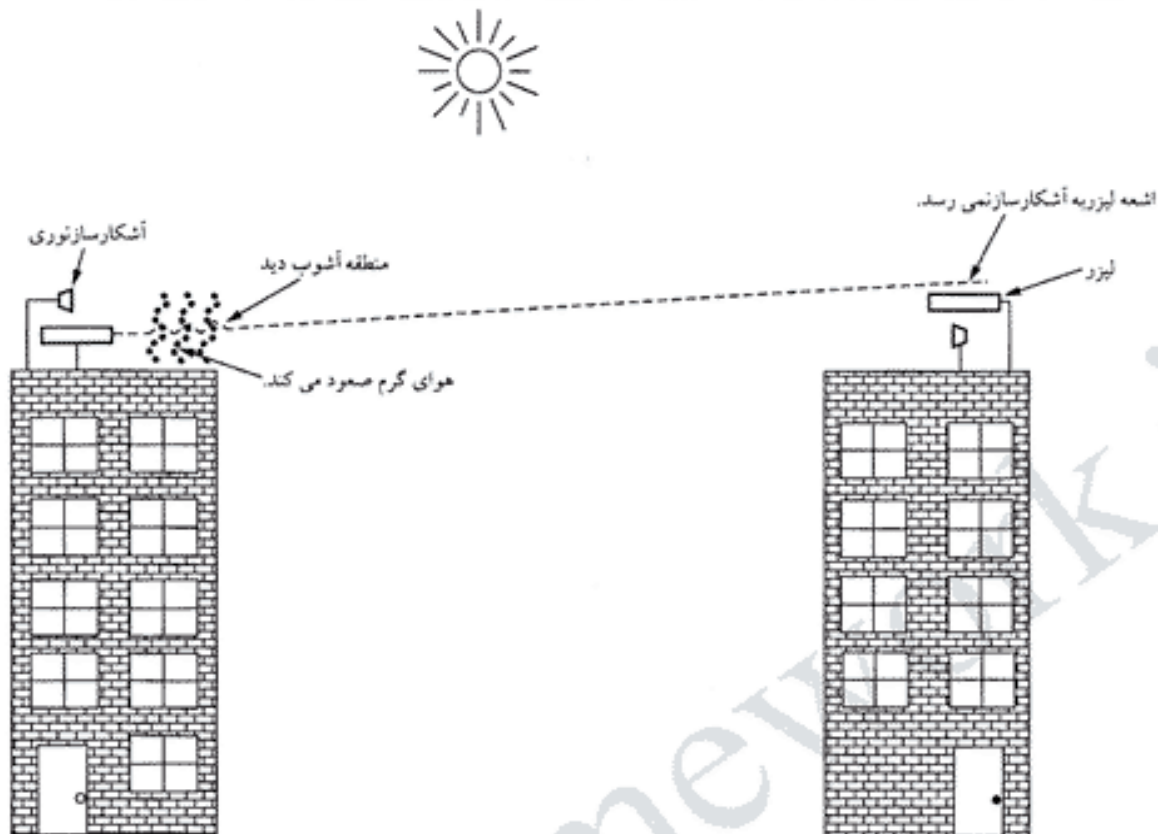
البته مزیت اصلی پرتو لیزر (یعنی باریک بودن آن) در اینجا یک نقطه ضعف محسوب می‌شود. هدف‌گیری یک پرتو لیزر به قطر ۱ میلی‌متر روی هدفی باندازه‌ی سه سوزن در فاصله ۵۰۰ متری حتی برای قهرمان تیراندازی المپیک هم دشوار است. در این سیستمها معمولاً از عدسیهای خاصی برای پراکنده کردن نور لیزر و پهن کردن پرتوهای آن استفاده می‌شود.

یکی دیگر از معایب لیزر اینست که نمی‌تواند در روزهای بارانی و مه‌آلود بخوبی عمل کند، ولی برای هوای صاف و آفتابی ایده‌آل است. تجربه جالبی که مؤلف این کتاب از چنین سیستمی دارد، می‌تواند آموزنده باشد. چندی قبل در یک کنفرانس علمی که در یکی از هتل‌های مدرن اروپایی ترتیب یافته بود، شرکت داشتم. مسئولین با درایت کنفرانس تعدادی کامپیوتر در یک اتاق نصب کرده بودند، تا مدعوین بتوانند در حین شرکت در سخنرانی‌ها ایمیل خود را نیز چک کنند. از آنجائیکه شرکت مخابرات محلی حاضر نشده بود تعداد زیادی خط تلفن را فقط برای سه روز برگزاری کنفرانس در اختیار برگزارکنندگان آن قرار دهد، آنها یک دستگاه لیزر روی پشت بام هتل قرار داده بودند، تا آنجا را به ساختمان دپارتمان کامپیوتر دانشگاه که در چند کیلومتری آن محل قرار داشت، وصل کنند. سیستم شب قبل از افتتاح کنفرانس تست شده بود، و همه چیز مرتب بنظر می‌رسید. ساعت ۹ صبح روز بعد (که روزی صاف و آفتابی بود) ارتباط بکلی قطع شد، و تمام آن روز نیز وصل نشد. عصر همان روز، برگزارکنندگان کنفرانس دوباره سیستم را تست کردند، و این بار هم همه چیز درست و عالی بود. اما صبح روز بعد (و روز بعد از آن) دوباره همان اتفاق تکرار شد. فقط بعد از پایان کنفرانس بود که علت کشف شد.

با شروع روز، گرمای خورشید باعث گرم شدن بام ساختمان شده، و هوای گرم به بالا صعود می‌کرد (شکل ۱۴-۲). این جریان هوای آشفته باعث می‌شد تا پرتو لیزر به مقدار ناچیز منحرف شده، و تمرکز آن روی گیرنده ساختمان هدف بهم بخورد. این دقیقاً همان پدیده‌ایست که باعث می‌شود تا ستارگان در شب چشمک بزنند، و یا در روزهای داغ تابستان در جاده‌ها همه چیز مواج بنظر برسد.

۴ ماهواره‌های مخابراتی

در سالهای ۱۹۵۰ و اوایل ۱۹۶۰ برخی افراد تلاش کردند تا با استفاده از بالونهای هوای گرم که پوششی فلزی داشتند، نوعی سیستم انعکاس امواج رادیویی بسازند. متأسفانه سیگنال برگشتی چنان ضعیف بود که به هیچ دردی نمی‌خورد. سپس نیروی دریایی آمریکا متوجه بالونی شد که همیشه در آسمان است؛ ماه؛ سیستمی که ساخته شد با استفاده از انعکاس امواج از ماه، بین کشتی‌ها و پایگاههای ساحلی ارتباط برقرار می‌کرد. پیشرفت بیشتر در مخابرات فضایی فقط زمانی ممکن شد که انسان اولین ماهواره‌های مخابراتی را به فضا

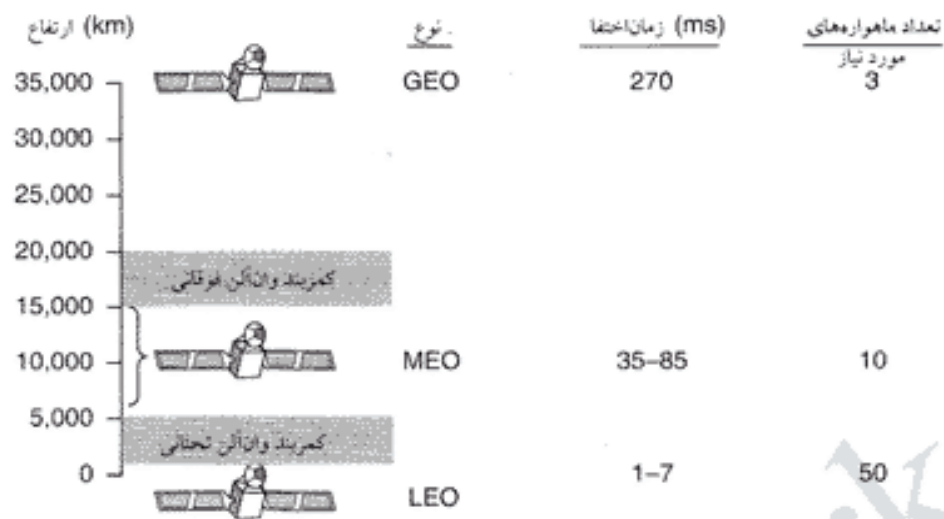


شکل ۲-۱۴. جریانهای همرفتی می تواند باعث انحراف پرتوهای لیزر شود.

پرتاب کرد. تفاوت کلیدی بین ماهواره های مصنوعی و اجسام فضایی این بود که ماهواره های ساخته دست بشر می توانستند قبل از برگشت دادن سیگنال آنها تقویت کنند - بدین ترتیب، یک کنجکاو عجیب تبدیل به یکی از ارکان زندگی بشر شد.

ماهواره های مخابراتی چند ویژگی دارند که آنها را برای کاربردهای مختلف جذاب می کند. یک ماهواره مخابراتی، در ساده ترین شکل خود، یک تکرارکننده مایکروویو بزرگ در آسمان است. چنین ماهواره ای دارای چندین ترانسپاندر (تکرارکننده - transponder) است، که به فرکانسهای خاصی گوش می کنند، سیگنال دریافتی را تقویت کرده، و سپس آنها را با فرکانس متفاوتی برمی گردانند (تا با سیگنال ورودی تداخل نکند). پرتو برگشتی می تواند وسیع باشد و بخش بزرگی از سطح کره زمین را بپوشاند، یا اینکه باریک باشد و فقط منطقه ای به قطر چند صد کیلومتر مربع را پوشش دهد. به این حالت شیپور خمیده (bent pipe) گفته می شود.

طبق قانون کپلر، دوره گردش مداری یک ماهواره با توان $\frac{3}{4}$ شعاع مدار آن متناسب است - هر چه مدار ماهواره بالاتر باشد، دوره گردش آن بیشتر است. در مدارهای نزدیک زمین دوره گردش ماهواره ها معمولاً ۹۰ دقیقه است، و آنها بسرعت از دید ایستگاههای زمینی خارج می شوند. برای پوشش دادن تمام سطح کره زمین با چنین ماهواره هایی، به تعداد زیادی از آنها نیاز است. در مداری به ارتفاع 35,800 km دوره گردش ماهواره ۲۴ ساعت (معادل یک دور گردش زمین به دور خود) است. اگر ارتفاع ماهواره را تا 384,000 km بالا ببریم، دوره گردش آن یک ماه خواهد شد - اگر باور ندارید، به ماه نگاه کنید؛ ارتفاع مداری ماه از سطح زمین دقیقاً همین مقدار است. دوره گردش ماهواره اهمیت زیادی دارد، ولی تنها عاملی نیست که تعیین می کند ماهواره کجا باید قرار گیرد. عامل مهم دیگر، وجود کمربند وان آلن (Van Allen belt) است - کمربند وان آلن لایه ای از ذرات باردار پُرانرژی است که توسط میدان مغناطیسی زمین بدام افتاده اند. هر ماهواره ای که در داخل این لایه پرواز کند،



شکل ۲-۱۵. ماهواره‌های مخابراتی و برخی از ویژگی‌های آنها، از جمله ارتفاع از سطح زمین، زمان تأخیر رفت و برگشت سیگنال، و تعداد ماهواره‌های که برای پوشش کل زمین لازم است.

بسرعت توسط ذرات باردار پُرانرژی آن از بین می‌رود. از ترکیب این دو عامل سه ناحیه مشخص می‌شود، که می‌توان ماهواره‌ها را با اطمینان خاطر در آنها قرار داد (شکل ۲-۱۵ را ببینید). در قسمتهای آینده ماهواره‌هایی را که در هر یک از این ناحیه‌ها قرار داده می‌شوند، تشریح خواهیم کرد.

۱-۴-۲ ماهواره‌های زمین ثابت

در سال ۱۹۴۵، نویسنده‌ی داستانهای علمی-تخیلی آر تور سی. کلارک با محاسبات خود نشان داد که اگر ماهواره‌ای در یک مدار استوایی و در ارتفاعی معادل 35,800 km قرار گیرد، نسبت به زمین ثابت بنظر خواهد رسید، و نیازی نیست که ایستگاه زمینی آنرا تعقیب کند (Clarke, 1945). وی در این مقاله یک سیستم کامل مخابراتی را با استفاده از این ماهواره‌های زمین ثابت (geostationary satellites) تشریح کرده بود، و از جمله مدار آنها، پانلهای خورشیدی (برای تأمین انرژی مورد نیاز)، فرکانسهای رادیویی، و نحوه پرتاب ماهواره‌ها را بدقت توضیح داده بود. متأسفانه، وی در این مقاله نتیجه گرفته بود که این طرح غیر عملی است، چون قرار دادن تقویت‌کننده‌های لامپ خلأ (که بسیار پرمصرف، حجیم و شکننده بودند) در مدار زمین غیر ممکن است، و هرگز این ایده را دنبال نکرد، اگر چه چند داستان علمی-تخیلی در این زمینه نوشت.

اختراع ترانزیستور همه چیز را تغییر داد، و اولین ماهواره مخابراتی بنام تل استار (Telstar) در ژوئیه ۱۹۶۲ به فضا پرتاب شد. از آن زمان به بعد، ماهواره‌های مخابراتی صنعتی چندمیلیارد دلاری را بوجود آورده‌اند، که تنها زمینه سودآور تحقیقات فضایی است. به این ماهواره‌های بلندپرواز اغلب ماهواره‌های GEO (Geostationary Earth Orbit) گفته می‌شود.

با تکنولوژی موجود (برای جلوگیری از تداخل امواج) حداقل فاصله دو ماهواره زمین ثابت نمی‌تواند کمتر از ۲ درجه در صفحه استوایی باشد. بدین ترتیب، در هر زمان بیش از $360 / 2 = 180$ نمی‌توانند در مدار باشند. با این حال برای بالا بردن پهنای باند در دسترس، می‌توان در هر ترانسپاندر از فرکانسها و پولاریزاسیونهای مختلف استفاده کرد.

در این مورد هم برای جلوگیری از هرج و مرج در مدار زمین، تخصیص مکانهای مداری توسط ITU انجام می‌شود. پای سیاست به اینجا هم باز شده است: کشورهایی که بزحمت از عصر حجر فاصله گرفته‌اند، سهم خود

را از مکانهای مداری طلب می کنند، تا بعد بتوانند آنرا به قیمت خوب به متقاضیان اجاره بدهند. کشورهایی هم هستند که ادعا می کنند قلمرو خاک آنها تا کره ماه ادامه دارد، و هیچکس حق ندارد در آسمان بالای سر آنها ماهواره داشته باشد. و برای شلوغی بیشتر اوضاع، فقط مخابراتی ها نیستند که بر سر تصاحب مدارها می جنگند؛ تلویزیونهای ماهواره ای، دولتها، و نظامیها هم می خواهند سهمی از مدارهای زمین داشته باشند.

ماهواره های امروزی بسیار بزرگ هستند، که وزن آنها گاهی به ۴۰۰۰ کیلوگرم می رسد، و دهها کیلووات انرژی الکتریکی مصرف می کنند (انرژی که توسط پانلهای خورشیدی تولید می شود). مدار و موقعیت این ماهواره ها تحت تأثیر جاذبه خورشید، ماه و سایر سیارات منظومه شمسی پیوسته در حال تغییر است، که این تأثیرات توسط موتورهای موشکی کوچکی که در آنها تعبیه شده، خنثی می شود (به این کار نگهداری ایستگاه - station keeping - می گویند). از آنجائیکه سوخت این موتورها محدود است، بعد از مدتی (که معمولاً حدود ۱۰ سال است) ماهواره بکلی از کنترل خارج و بلااستفاده می شود. ارتفاع چنین ماهواره هایی بتدریج کاهش می یابد، و پس از ورود به جو زمین می سوزند، و یا در برخورد با سطح زمین متلاشی می شوند.

مکانهای مداری تنها چیزی نیست که سر آن دعواست. فرکانسها نیز خواهان زیادی دارد، چون احتمال تداخل فرکانسهای ارسالی از ماهواره (downlink) با فرکانسهای زمینی وجود دارد. برای رفع این مشکل، ITU چند باند فرکانسی را به کاربردهای ماهواره ای اختصاص داده است، که آنها را در شکل ۲-۱۶ ملاحظه می کنید. باند C اولین باندهای بود که به کاربردهای تجاری اختصاص یافت. این باند دو محدوده دارد: محدوده پائینی که برای ارسال از ماهواره (downlink) استفاده می شود، و محدوده بالایی که به ارسال به ماهواره (uplink) تخصیص یافته است. این باندها بسیار شلوغ هستند، چون در مخابرات مایکروویو زمینی هم از آنها استفاده می شود. باندهای L و S در سال ۲۰۰۰ طبق توافقات بین المللی اضافه شدند، اما این باندها نیز بسیار باریک و شلوغ هستند.

باند بعدی که در اختیار مخابرات تجاری قرار دارد، باند Ku (K under) است. این باند هنوز پُر نشده، و در فرکانسهای آن می توان ماهواره ها را تا فاصله ۱ درجه مداری به هم نزدیک کرد، ولی یک مشکل عمده دارد: باران. آب جاذب خوبی برای انرژی امواج در این طول موج است. خوشبختانه، طوفانهای بزرگ معمولاً به یک منطقه کوچک محدود هستند، و اگر بجای یک ایستگاه زمینی از چندین ایستگاه با فواصل زیاد استفاده کنیم، این مشکل مرتفع خواهد شد (که البته این راه حل هزینه بسیار بالایی دارد). باند Ka (K above) نیز به مصارف تجاری تخصیص یافته، ولی تجهیزات آن هنوز بسیار گران است. علاوه بر این باندهای تجاری، دهها باند دولتی و نظامی نیز وجود دارد.

یک ماهواره امروزی در حدود ۴۰ ترانسپاندر دارد، که پهنای باند هر کدام از آنها ۸۰-MHz است. معمولاً هر ترانسپاندر بعنوان یک شیپور خمیده عمل می کند، ولی در ماهواره های جدیدتر (که امکانات پردازشی بیشتری دارند) می توان بگونه ای دیگر نیز عمل کرد. در ماهواره های اولیه، تقسیم ترانسپاندرها به کانالهای مختلف استاتیک بود: پهنای باند بصورت ساده به چند باند فرکانسی ثابت تقسیم می شد. امروزه، پرتو ترانسپاندر به بُرشهای زمانی

مشکلات	پهنای باند	ارسال به ماهواره	دریافت از ماهواره	باند
پهنای باند کم به شلوغ	15 MHz	1.6 GHz	1.5 GHz	L
پهنای باند کم به شلوغ	70 MHz	2.2 GHz	1.9 GHz	S
تداخل زمینی	500 MHz	6.0 GHz	4.0 GHz	C
باران	500 MHz	14 GHz	11 GHz	Ku
باران با قیمت بالای تجهیزات	3500 MHz	30 GHz	20 GHz	Ka

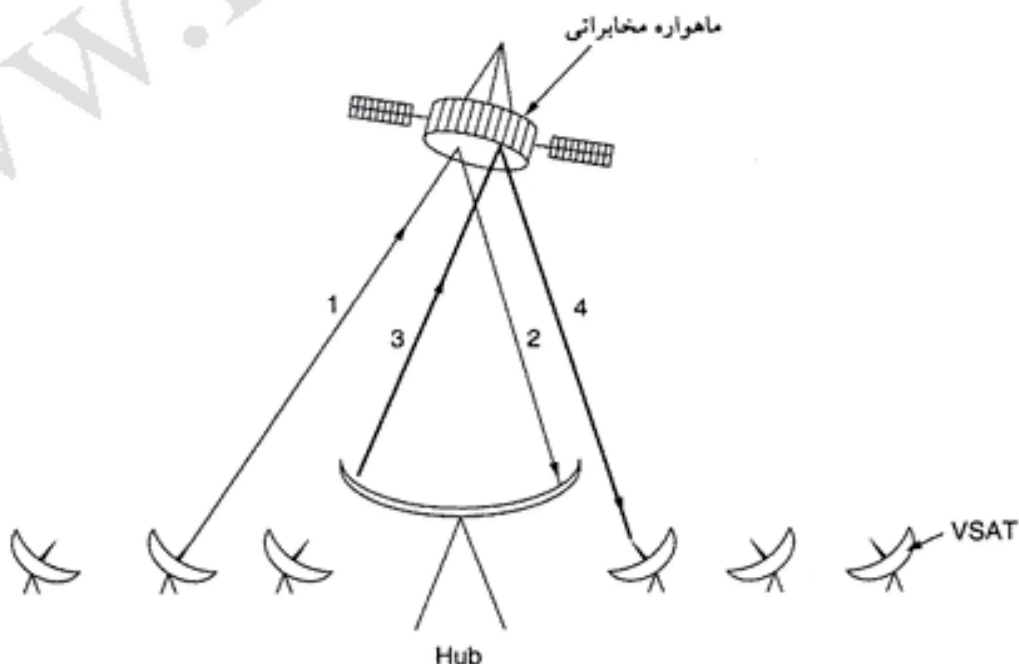
شکل ۲-۱۶. باندهای ماهواره ای.

تقسیم می‌شود، و هر کاربر می‌تواند بنوبت از آن استفاده کند. در قسمتهای آینده همین فصل این دو تکنیک (مالتی پلکس تقسیم فرکانسی و مالتی پلکس تقسیم زمانی) را به تفصیل بررسی خواهیم کرد.

اولین ماهواره‌های زمین ثابت یک پرتو فضایی واحد داشتند، که تقریباً $\frac{1}{3}$ سطح زمین را می‌پوشاند (به این ناحیه جای پای ماهواره - footprint - گفته می‌شود). اما با افت شدید قیمت، اندازه، و توان مصرفی وسایل میکروالکترونیک، استراتژیهای هوشمندانه‌تری امکان پذیر گردید. هر ماهواره به چندین آنتن و ترانسپاندر مجهز شد، و پرتو ارسالی از ماهواره چنان باریک شد که فقط منطقه کوچکی را در بر می‌گرفت، و بدین ترتیب امکان مخابره همزمان چندین سیگنال بوجود آمد. این پرتوهای نقطه‌ای (spot beam) معمولاً بیضی شکل هستند، و می‌توان آنها را بقدری باریک کرد که فقط ناحیه‌ای به قطر چند صد کیلومتر را پوشش دهند. برای مثال، یک ماهواره مخابراتی که فقط برای ایالات متحده آمریکا به فضا پرتاب شده، را می‌توان طوری تنظیم کرد که یک پرتو برای ۴۸ ایالت خاک اصلی آمریکا، یک پرتو برای آلاسکا، و یکی برای هاوایی داشته باشد.

یکی از پیشرفتهای جدید در دنیای مخابرات ماهواره‌ای، ایستگاههای ارزان قیمتی هستند که VSAT (ترمینال با باریکه بسیار کوچک - Very Small Aperture Terminal) نامیده می‌شوند (Abramson, 2000). این ترمینالهای بسیار کوچک آنتنهایی بقطر ۱ متر یا کمتر دارند (برای مقایسه، قطر آنتنهای استاندارد GEO ۱۰ متر است)، و قدرت تشعشعی آنها در حدود ۱ وات است. سرعت ارسال این سیستمها در حدود 19.2-kbps است، ولی می‌توانند تا 512-kbps یا بیشتر دریافت کنند. تلویزیونهای ارسال مستقیم ماهواره‌ای (DB-SAT) معمولاً از این تکنولوژی استفاده می‌کنند.

در بسیاری از سیستمهای VSAT، ایستگاههای زمینی بدلیل توان پائین نمی‌توانند مستقیماً با هم ارتباط برقرار کنند (البته از طریق ماهواره). در این موارد، از یک ایستگاه زمینی خاص، با آنتنی بزرگ و قوی بعنوان هاب (hub)، برای رله کردن سیگنالها از یک VSAT به VSAT دیگر استفاده می‌کنند (شکل ۲-۱۷). در این حالت، یکی از طرفین باید آنتنی بزرگ با یک تقویت کننده قوی داشته باشد. زمان تأخیر سیگنال در این روش بیشتر از ارتباط



شکل ۲-۱۷. چند سیستم VSAT به همراه یک هاب.

مستقیم است، ولی کاربران معمولاً هزینه کمتر را به کمی تأخیر ترجیح می دهند. سیستمهای VSAT بیشترین کاربرد را در مناطق روستایی دارد. متأسفانه، هنوز نیمی از جمعیت دنیا به تلفن دسترسی ندارند، و کشیدن خطوط تلفن به هزاران دهکده کوچک و دورافتاده از توان اغلب کشورهای جهان سوم خارج است - ولی نصب یک آنتن VSAT یک متری (که با سلولهای خورشیدی کار کند) کار چندان مشکلی نیست. VSAT تکنولوژی است که می تواند همه مردم دنیا را به هم وصل کند.

مخابرات ماهواره ای ویژگیهایی دارد که بشدت با ارتباطات نقطه-به-نقطه زمینی متفاوتند. اولین ویژگی اینست که، با آنکه سیگنالهای ماهواره ای با سرعت نور (نزدیک $300,000 \text{ km/sec}$) حرکت می کنند، (بعلمت فاصله زیاد ماهواره تا سطح زمین) ارتباط با ماهواره های GEO دارای زمان تأخیر سیگنال قابل ملاحظه ای است. بسته به فاصله کاربر از ایستگاه زمینی و زاویه ماهواره نسبت به افق در آن محل، زمان ارتباط نقطه-به-نقطه بین 25° تا 30° میلی ثانیه متغیر است. زمان متوسط این تأخیر 27° میلی ثانیه است (که برای سیستمهای VSAT دارای هاب به دو برابر، یعنی 54° میلی ثانیه، می رسد).

برای مقایسه بد نیست بدانید که، زمان تأخیر انتشار در لینکهای میکروویو زمینی حدود $3 \mu\text{sec/km}$ ، و برای کابلهای کواکسیال و فیبر نوری تقریباً $5 \mu\text{sec/km}$ است (امواج الکترومغناطیس در هوا سریعتر از مواد جامد حرکت می کنند).

ویژگی مهم دیگر ماهواره ها اینست که آنها ذاتاً رسانه های پخش هستند؛ فرستادن پیام برای یک نفر هیچ تفاوتی با هزاران نفر ندارد. در برخی از کاربردها (مانند تبلیغات اینترنتی) این ویژگی بسیار مفید است. با اینکه بوسیله ارتباطات نقطه-به-نقطه هم می توان چنین وضعیتی را شبیه سازی کرد، ولی این کار با استفاده از ماهواره بسیار ارزاتر تمام می شود. از سوی دیگر، از دیدگاه امنیت و حفظ حریم خصوصی افراد، ماهواره یک فاجعه تمام عیار است: هر کسی می تواند پیامهای خصوصی دیگران را بشنود. اینجاست که اهمیت رمزنگاری (encryption) روشن می شود.

در مخابرات ماهواره ای، هزینه انتقال پیام به فاصله فرستنده و گیرنده بستگی ندارد: تماس با آن سوی اقیانوسها از نظر هزینه هیچ فرقی با تماس با خانه روبرویی ندارد. مخابرات ماهواره ای از نظر نرخ خطا بسیار عالیست، و زمان به بهره برداری رسیدن آن نیز بسیار کوتاه است (نکته ای که در مخابرات نظامی بسیار اهمیت دارد).

۲-۴-۲ ماهواره های مدار متوسط

در مداری بسیار پائینتر از مدار زمین ثابت، و بین دو کمربند وان آلن، ماهواره های مدار متوسط که به MEO (Medium-Earth Orbit) معروفند، قرار می گیرند. این ماهواره ها بطور متوسط هر ۶ ساعت یکبار دور زمین می گردند، و بهمین دلیل ایستگاه زمینی باید آنها را تعقیب کند. بعلمت ارتفاع پائین، ماهواره های MEO جای پای کوچکتری نسبت به ماهواره های GEO دارند، ولی توان تشعشعی لازم برای ارسال به آنها نیز بسیار کمتر است. در حال حاضر از این ماهواره ها برای مقاصد مخابراتی استفاده نمی شود، بنابراین ما هم درباره آنها بیش از این صحبت نخواهیم کرد. ماهواره های ۲۴ گانه GPS (سیستم مکان یابی جهانی - Global Positioning System) که در ارتفاع 18,000 km پرواز می کنند، در این دسته قرار می گیرند.

۳-۴-۲ ماهواره های مدار پائین

اگر باز هم پائینتر بیانیم، به ماهواره های مدار پائین (Low-Earth Orbit) LEO می رسیم. بدلیل سرعت زیاد گردش مداری این ماهواره ها، برای ایجاد سیستمی با پوشش جهانی به تعداد زیادی از آنها نیاز داریم. از طرف دیگر، بدلیل ارتفاع کم ماهواره های LEO، برای ارتباط با آنها به توان کمی نیاز است، و زمان رفت و برگشت

سیگنال نیز بسیار کم (در حدود چند میلی ثانیه) خواهد بود. در این قسمت سه سیستم ماهواره ای LEO را - که دو تای آنها به سرویس صدا و یکی به سرویسهای اینترنت اختصاص دارند - مورد بررسی قرار خواهیم داد.

ایریدیوم

همانطور که گفتیم بعلت سرعت زیاد گردش مداری ماهواره های LEO - و عبور سریع از مقابل آنتنهای زمینی - تا همین اواخر از این ماهواره ها برای مقاصد مخابراتی استفاده نمی شد. این وضعیت در سال ۱۹۹۰ تغییر کرد: در این سال موتورولا درخواستی برای دریافت مجوز پرتاب ۷۷ ماهواره مخابراتی LEO به FCC ارائه کرد - این پروژه ایریدیوم (Iridium) نام داشت (ایریدیوم عنصر ۷۷ام جدول تناوبی است). بعدها این پروژه تغییر کرد، و قرار شد از فقط ۶۶ ماهواره استفاده شود - بالطبع نام پروژه را هم باید به دیسپروسیوم (عنصر ۶۶ام جدول تناوبی) تغییر می دادند، اما نام دیسپروسیوم بیشتر شبیه یک بیماری خطرناک است تا یک سیستم ماهواره ای! ایده اصلی در این سیستم آن است که به محض خارج شدن یک ماهواره از دید آنتن زمینی، ماهواره دیگری بلافاصله جای آنرا می گیرد. این پیشنهاد به یک هیجان عمومی در میان شرکت های مخابراتی دامن زد، و همه خواهان آن بودند که زنجیره ای از ماهواره های LEO به فضا پرتاب کنند.

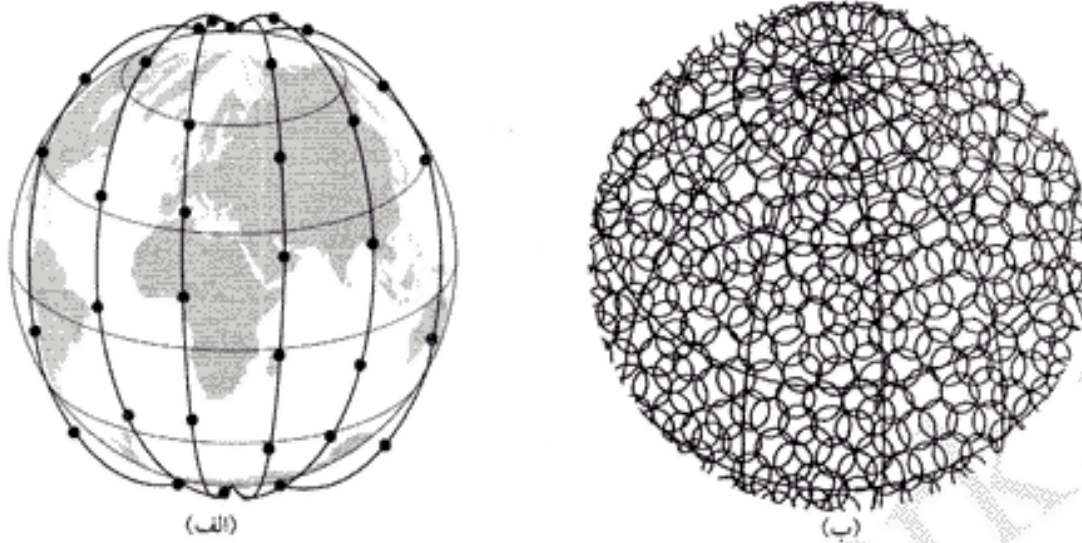
بعد از هفت سال اشتراک مساعی فنی و مالی بین چند شرکت بزرگ، بالاخره ماهواره های ایریدیوم در سال ۱۹۹۷ به فضا پرتاب شدند، و از نوامبر ۱۹۹۸ سرویسهای مخابراتی آنها شروع بکار کرد. اما متأسفانه بدلیل گسترش شبکه تلفنهای همراه، تقاضای ناچیزی برای این سرویسها در بازار وجود داشت، و متعاقب آن در آگوست ۱۹۹۹ پروژه ایریدیوم (طی یکی از فاجعه بارترین ورشکستگی های تاریخ) متوقف شد. ماهواره ها و سایر تجهیزات این پروژه (که ۵ میلیارد دلار ارزش داشت) به قیمت ۲۵ میلیون دلار به یک سرمایه گذار فروخته شد؛ در واقع، پروژه ایریدیوم را باید بزرگترین بازار اسقاطی فضایی بشمار آورد. سرویسهای ایریدیوم از مارس ۲۰۰۱ دوباره راه اندازی شد.

کار اصلی ایریدیوم ارائه سرویسهای مخابراتی از طریق تجهیزاتی که مستقیماً با ماهواره ارتباط برقرار می کنند، بود (و هست). ایریدیوم سرویسهای صدا، داده، فکس، پیجر، و هدایت و ناوبری را در تمام نقاط زمین (خشکی، دریا و هوا) در اختیار کاربران خود می گذارد. کاربران این سرویسها را عمدتاً دریانوردان، هوانوردان، کارکنان سکویهای اکتشاف نفت، و مسافران و جهانگردانی که به مناطق فاقد زیرساختهای مخابراتی (مانند کوه، صحرا، جنگل، و برخی از کشورهای جهان سوم) سفر می کنند، تشکیل می دهند.

ماهواره های ایریدیوم در مدار قطبی و در ارتفاع ۷۵۰ کیلومتری زمین پرواز می کنند. این ماهواره ها بصورت کمربندهایی که از قطبهای شمال-جنوب زمین می گذرند، آرایش یافته اند، و با یکدیگر ۳۲ درجه عرض جغرافیایی فاصله دارند - شکل ۲-۱۸ (الف) را ببینید. برای پوشش دادن تمام سطح زمین شش تا از این حلقه ها کفایت می کند. آنهایی که کمی با شیمی آشنایند، می توانند این ماهواره ها را الکترونهای یک اتم غول آسای دیسپروسیوم فرض کنند که بدور هسته اتم (کره زمین) در گردشند.

هر ماهواره حداکثر ۴۸ سلول (پرتو نقطه ای) دارد، که بدین ترتیب تعداد کل سلولها به ۱۶۲۸ می رسد، و می توانند کل سطح زمین را پوشش دهند - شکل ۲-۱۸ (ب) را ببینید. هر ماهواره ایریدیوم ۳۸۴۰ کانال - و کل سیستم ۲۵۳،۴۴۰ کانال - ظرفیت دارد، و می توان از آنها برای سرویسهای مختلف استفاده کرد.

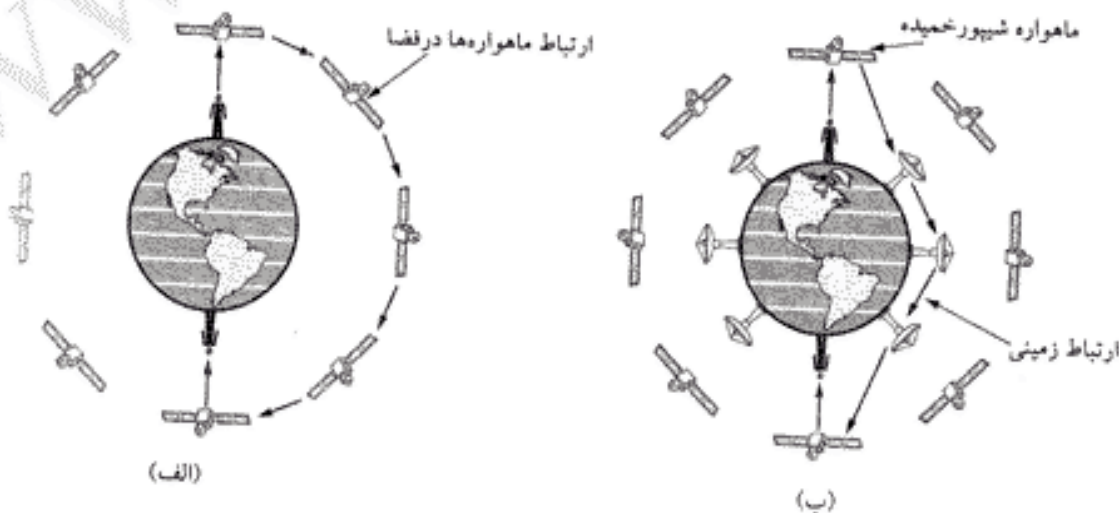
یکی از نکات جالب در مورد ایریدیوم اینست که ارتباط کاربرانی که از یکدیگر فاصله زیادی دارند، در فضا صورت می گیرد (بعبارت دیگر، رله کردن اطلاعات بین ماهواره ها در فضا انجام می شود)؛ شکل ۲-۱۹ (الف) را ببینید. همانطور که در این شکل می بینید، وقتی کاربری در قطب شمال بخواهد با قطب جنوب تماس بگیرد، سیگنال را به ماهواره ای که بالای سرش قرار دارد، می فرستد. سپس این سیگنال از یک ماهواره به ماهواره دیگر رله می شود، تا به قطب جنوب برسد.



شکل ۲-۱۸. (الف) ماهواره های ایریدیوم شش حلقه بدور زمین می سازند. (ب) تعداد سلولهای متحرک ایریدیوم به ۱۶۲۸ می رسد.

گلوبال استار

یکی از طرحهای رقیب ایریدیوم، پروژه گلوبال استار (Globalstar) است. این پروژه از ۴۸ ماهواره LEO تشکیل شده، ولی روش سونیچینگ آن با ایریدیوم فرق دارد، و در آن از سونیچینگ زمینی استفاده می شود - شکل ۲-۱۹ (ب) را ببینید. در اینجا سیگنالها از ماهواره مبدأ به یک ایستگاه زمینی بزرگ واقع در سانتا ورک شاپ هدایت شده، و از آنجا به نزدیکترین ایستگاه زمینی مقصد فرستاده می شوند، تا بالاخره (بعد از ارسال به ماهواره) به گیرنده برسند. مزیت این روش آنست که (بر خلاف ایریدیوم) قسمت پیچیده کار روی زمین انجام می شود، و مدیریت آن ساده تر است. همچنین، با آنتنهای بزرگ زمینی (که سیگنالهای قویتری می فرستند، و می توانند سیگنالهای ضعیفتری دریافت کنند) از تجهیزات انفرادی ساده تری می توان استفاده کرد (توان شعشی تلفنها در حد چند میلی وات است، و حتی بعد از تقویت در ماهواره باز هم چندان قوی نیست).



شکل ۲-۱۹. (الف) رله سیگنال در فضا. (ب) رله سیگنال روی زمین.

تله‌دزیک

کاربران اصلی ایریدیوم افرادی هستند که به جاهای پرت و دورافتاده می‌روند. نمونه دیگر سیستمهای ماهواره‌ای تله‌دزیک (Teledesic) است، که کاربران اینترنت پرسرعت را (در تمام دنیا) هدف گرفته است. ایده این سیستم به کریگ مک‌کاو (پیشگام صنعت تلفنهای همراه) و بیل گیتس (بنیانگذار میکروسافت) تعلق دارد، که از سرعت لاک‌پشتی دسترسی اینترنت ناراضی بودند. هدف سیستم تله‌دزیک فراهم آوردن دسترسی به اینترنت با سرعتهای بالا (تا 100-Mbps ارسال، و تا 720-Mbps دریافت) برای میلیونها کاربر همزمان است. این سیستم از آنتنهای کوچک و ثابت (شبیه VSAT) استفاده می‌کند، و هیچ نیازی به ارتباطات تلفنی معمولی ندارد.

در طرح اولیه سیستم تله‌دزیک قرار بود از ۲۸۸ ماهواره پرتو باریک، که در دوازده گروه در ارتفاع ۱۳۵۰ کیلومتری (درست زیر کمربند وان‌آلن تحتانی) پرواز می‌کنند، استفاده شود. ولی این طرح بعدها به ۳۰ ماهواره با جای پای بزرگتر تغییر کرد. ماهواره‌ها در باند نسبتاً خلوت Ka و بصورت سونیچینگ بسته‌ای کار خواهند کرد، و می‌توانند بسته‌ها را بین خود رد و بدل کنند. وقتی یک کاربر به پهنای باند نیاز پیدا می‌کند، درخواست خود را بصورت یک بسته می‌فرستد، و ۵۰ میلی‌ثانیه بعد پهنای باند مورد نیاز بصورت خودکار به وی اختصاص داده می‌شود. اگر همه چیز طبق نقشه پیش برود، این سیستم در سال ۲۰۰۵ عملیاتی خواهد شد.

۲-۴-۴ ماهواره یافیبیر؟

مقایسه‌ای بین مخابرات ماهواره‌ای و زمینی آموزنده خواهد بود. تا همین ۲۰ سال پیش هیچکس شک نداشت که آینده مخابرات متعلق به ماهواره‌هاست، چون سیستمهای تلفن در ۱۰۰ سال گذشته تغییر عمده‌ای نکرده بودند، و بنظر نمی‌رسد در ۱۰۰ سال بعدی هم هیچ اتفاق خاصی بیفتد. شرکت‌های تلفن سرویسهای صوتی خوب (با قیمت مناسب) ارائه می‌کردند، و سرمایه‌گذاری آنها سودی تضمین شده داشت. برای آنهایی هم که سرویس داده می‌خواستند، مودمهای 1200-bps موجود بود - و این تمام بضاعت شرکت‌های تلفن بود.

اما در سال ۱۹۸۴ آتش رقابت ایالات متحده آمریکا و اروپا را در بر گرفت، و اوضاع تغییر کرد. شرکت‌های تلفن در سرویسهای راه‌دور فیبرهای نوری را جایگزین کابلهای مسی کردند، و سرویسهای با پهنای باند زیاد مانند ADSL (خط دیجیتال نامتقارن) در اختیار کاربران خود قرار دادند. تغییر اساسی دیگر کاهش نرخهای مخابرات راه‌دور بود، که تا آن زمان بطور مصنوعی و به نفع کاربران محلی بالا نگه داشته شده بود. بنظر می‌رسید فیبر نوری برنده جنگ باشد، ولی ماهواره هم مزایایی دارد که فیبر نمی‌تواند (و احتمالاً نخواهد توانست) با آنها رقابت کند. اول اینکه، پهنای باند فیبر (که شاید یک رشته آن از تمام ماهواره‌های پرتاب شده بیشتر باشد) در اختیار اغلب کاربران نیست. فیبر نوری بیشتر در مخابرات راه‌دور و برای متصل کردن شبکه‌های تلفن بکار می‌رود، تا رساندن پهنای باند بالا به کاربران منفرد. در حالیکه ماهواره‌ها (به کمک یک آنتن ساده در بالای پشت بام) می‌توانند مستقیماً و بدون هیچ واسطه‌ای پهنای باند زیاد را در اختیار تک تک کاربران قرار دهند. سیستم تله‌دزیک بر اساس همین ایده شکل گرفته است.

مخابرات سیار دیگر قلمرو دست‌نیافتنی ماهواره‌هاست. امروزه بسیاری از افراد مایلند در حال قدم‌زدن، اتومبیل‌سواری، و حتی قایقرانی و پرواز به سرویسهای مخابراتی دسترسی داشته باشند. اینجا دیگر فیبر نوری بکلی بی‌استفاده است، و فقط مخابرات ماهواره‌ای می‌تواند راه چاره باشد. البته برای آنهایی که شعاع حرکتشان محدود است، می‌توان از ترکیب فیبر و بیسیمهای رادیویی استفاده کرد، ولی در هوا و دریا این روش هم دیگر کارایی ندارد.

مزیت دیگر ماهواره در مخابرات پخش (broadcasting) است. وقتی بخواهید یک پیام را در آن واحد به هزاران نفر برسانید (مانند ارسال نرخ کالا و ارز، یا قیمت سهام و اوراق قرضه)، هیچ چیز جای ماهواره را (از نظر

سهولت و هزینه) نمی‌گیرد.

پراکندگی جمعیت و سرزمین از دیگر عواملیست که بکارگیری مخابرات ماهواره‌ای را مقرون بصرفه می‌کند. برای مثال، کشور اندونزی برای هدایت ترافیک تلفن داخلی خود نیز از ماهواره استفاده می‌کند، چون پرتاب یک ماهواره بسیار ساده‌تر است، تا کشیدن کابل‌های زیر دریایی به ۱۳۶۷۷ جزیره‌ای که این کشور را تشکیل می‌دهند. در جاهایی که تملک زمین برای کشیدن کابل زمینی دشوار (و یا پرهزینه) است، نیز ماهواره مزیت نسبی دارد. سرعت نصب و راه‌اندازی سیستم نیز یکی از جاذبه‌های ماهواره بر کابل زمینی پیروز می‌شود. وقتی جنگی در می‌گیرد، و نیروهای نظامی به تماس با نقاط جدید نیاز پیدا می‌کنند، پرتاب یک ماهواره معمولاً سریع‌ترین راه حل ممکن است.

بطور خلاصه، بنظر می‌رسد که جریان اصلی مخابرات در آینده بر فیبرهای نوری و تلفنهای همراه متکیست، ولی در برخی از کاربردهای خاص ماهواره‌ها برتری دارند. اما همه آنها تابع یک چیز هستند: اقتصاد. با اینکه فیبر نوری پهنای باند بیشتری ارائه می‌کند، اما حرف آخر را قیمت می‌زند. اگر تکنولوژی جدیدی اختراع شود که هزینه پرتاب ماهواره‌ها را بشدت کاهش دهد (مانند شاتل‌های فضایی که بتوانند هر بار دهها ماهواره را به مدار ببرند)، یا کاربرد غیر منتظره‌ای برای ماهواره‌های مدار پائین پیدا شود، معلوم نیست که فیبر نوری در تمام زمینه‌ها برنده باشد.

۵-۲ شبکه تلفن عمومی

اگر بخواهیم دو کامپیوتر را که نزدیک به هم و یا در یک ساختمان هستند، به یکدیگر متصل کنیم، کار بسیار ساده است و فقط کافیست یک رشته کابل بین آنها بکشیم - این همان شبکه محلی یا LAN است. اما اگر فاصله کامپیوترها زیاد باشد، یا کابل بایستی از املاک خصوصی یا شهری عبور داده شود، هزینه انجام کابل کشی معمولاً به مانعی بزرگ تبدیل می‌شود (البته اگر این کار غیرقانونی نباشد، که در اغلب کشورها هست). در نتیجه، طراحان شبکه به تأسیسات مخابراتی موجود روی می‌آورند.

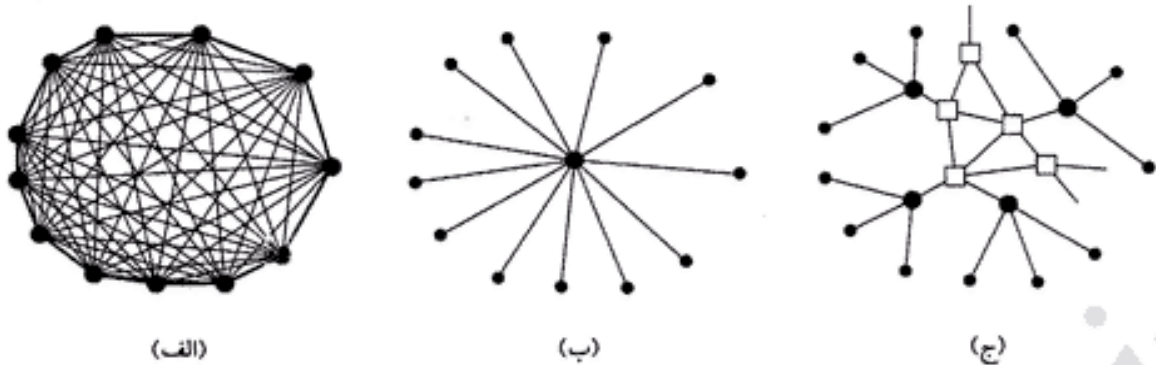
این تأسیسات بویژه شبکه تلفن عمومی - PSTN (Public Switched Telephone Network) - سالها قبل و با هدفی دیگر طراحی شده‌اند: انتقال صدای انسان بگونه‌ای کمابیش قابل تشخیص. کارایی این تجهیزات برای ارتباط کامپیوتر-کامپیوتر در بهترین حالت اغلب حاشیه‌ای است، اما با به بازار آمدن فیبر نوری و تکنولوژیهای دیجیتال این وضعیت سرعت در حال تغییر است. در هر حال، سیستم تلفن عمومی چنان با شبکه‌های (گسترده) کامپیوتری عجین است، که جا دارد وقت بیشتری به بررسی آن اختصاص دهیم.

برای بهتر شکافتن صورت مسئله، اجازه دهید مقایسه‌ای کلی (ولی روشن‌کننده) بین ارتباط دو کامپیوتر از طریق کابل مستقیم شبکه و از طریق خطوط تلفن داشته باشیم. کابل مستقیم شبکه می‌تواند داده‌ها را با سرعتی معادل 10^9 bps (یا حتی بیشتر) منتقل کند، در حالیکه حداکثر سرعت یک خط تلفن 56-kbps بیشتر نیست - تفاوتی در حد ۲۰,۰۰۰ برابر. تفاوت این دو مانند تفاوت سرعت یک مرغابی که سلانه سلانه در علفزار راه می‌رود، با موشکی است که به طرف ماه پرواز می‌کند. اگر به جای خط تلفن از اتصال ADSL استفاده کنیم، تفاوت سرعت به ۱۰۰۰ تا ۲۰۰۰ برابر خواهد رسید.

ناگفته پیداست که چنین سطح از تفاوتی برای طراحان سیستمهای کامپیوتری مشکل ساز است، و آنان تمام تلاش خود را برای بهینه کردن آن متمرکز کنند. در قسمتهای آینده سیستم تلفن و طرز کار آن را تشریح خواهیم کرد؛ برای اطلاعات بیشتر در این زمینه به (Bellamy, 2000) مراجعه کنید.

۱۵-۲ ساختار سیستم تلفن

بلافاصله بعد از آن که آلکساندر گراهام بل در سال ۱۸۷۶ (و درست چند ساعت زودتر از رقیبش، الیساگری)



شکل ۲-۲۰. (الف) شبکه‌ای با اتصالات داخلی کامل. (ب) سونیچ مرکزی. (ج) سلسله مراتب دو سطحی.

اختراع خود را به ثبت رساند، سیل تقاضا برای آن سرازیر شد. تلفنهای اولیه صورت جفتی کار می‌کردند، و مشتریان مجبور بودند بین خودشان یک رشته سیم بکشند (مسیر برگشت الکترونها از زمین بود). اگر کسی می‌خواست با n نفر تماس تلفنی داشته باشد، مجبور بود به این n نقطه سیم بکشد. در کمتر از یک سال شهرها تبدیل شدند به جنگلی از سیمهای تو در تو که از خانه‌ای به خانه دیگر (از فراز ساختمانها و درختها) کشیده شده بود. خیلی زود معلوم شد که مدل اتصال هر تلفن به تمام تلفنهای دیگر مدلی عملی نیست - شکل ۲-۲۰ (الف). خوشبختانه بل خیلی زود متوجه این مشکل شد، و با تأسیس شرکتی بنام شرکت تلفن بل، اولین مرکز سونیچینگ را در سال ۱۸۷۸ (در نیویورک، کانتیکات) راه‌اندازی کرد. این شرکت یک رشته کابل به خانه یا دفتر هر مشتری می‌کشید. برای برقراری تماس، مشتری دسته تلفن را می‌چرخاند تا زنگی در مرکز تلفن بصدا در آید و توجه اپراتور جلب شود؛ سپس این اپراتور ارتباط وی را توسط یک رشته کابل با مقصد موردنظر برقرار می‌کرد. این مدل را در شکل ۲-۲۰ (ب) ملاحظه می‌کنید.

بزودی در هر شهر و دهکده‌ای مراکز سونیچینگ بل دایر شد، و بل مجبور شد برای برقراری تماسهای راه دور این مراکز را نیز به یکدیگر متصل کند. اما در اینجا هم همان مشکل قبلی خود را نشان داد: ارتباط مستقیم هر مرکز سونیچینگ با تمام مراکز دیگر غیرممکن بود، بنابراین مراکز سونیچینگ سطح دوم اختراع شد. پس از مدتی کوتاه تعداد مراکز سطح دوم نیز یشت افزایش یافت - شکل ۲-۲۰ (ج) را ببینید. بعدها این سلسله مراتب تا پنج سطح بالا رفت.

تا سال ۱۸۹۰ این سیستم تلفن سه بخش عمده داشت: مراکز سونیچینگ، سیمهایی که بین مراکز سونیچینگ و مشترکان کشیده می‌شد (این سیمها دیگر سیمهای لخت با برگشت زمین نبود، بلکه تبدیل به سیمهای زوج تابیده عایق‌دار شده بود)، و اتصالات راه دور بین مراکز سونیچینگ. با آن که پیشرفتهایی در هر یک از این سه بخش صورت گرفته، اما مدل اولیه بل در ۱۰۰ سال گذشته تقریباً بدون تغییر باقی مانده است. برای دیدن تاریخچه‌ای مختصر از سیستم تلفن بل، به (Hawley, 1991) نگاه کنید.

تا قبل از دو پاره شدن AT&T در سال ۱۹۸۴، سیستم تلفن سیستمی با سلسله مراتب چندسطحی (و با پراکندگی زیاد) بود. در بحث زیر این ساختار تا حد زیادی ساده شده، ولی عصاره اصلی آن همچنان حفظ شده است. هر تلفن با دو رشته سیم مسی به نزدیکترین ایستگاه پایانی (end office یا local central office) وصل می‌شود. فاصله این دو معمولاً بین ۱ تا ۱۰ کیلومتر است (و در شهرها کمتر از مناطق روستاییست). فقط در ایالات متحده آمریکا نزدیک به ۲۲,۰۰۰ ایستگاه پایانی وجود دارد. به اتصال دو سیمه بین ایستگاه پایانی و مشترک تلفن

اصطلاحاً مدار پایانی (local loop) گفته می شود. اگر مدارهای پایانی موجود در سراسر دنیا را بدنبال هم ردیف کنند، می تواند هزار بار فاصله زمین تا ماه را بپیماید.

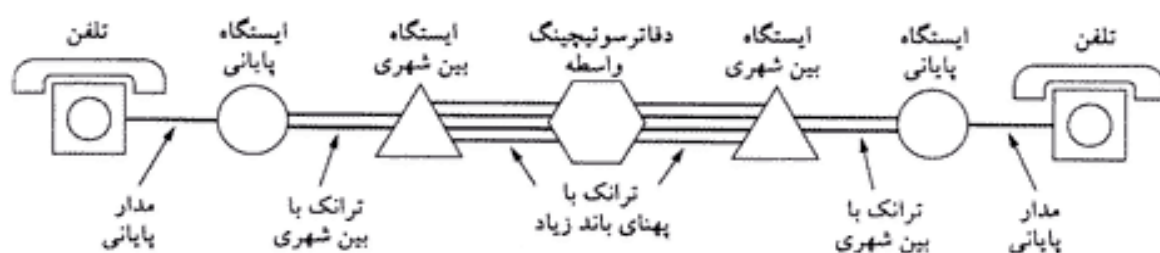
طبق یکی از تخمین ها، ۸۰ درصد ارزش سرمایه AT&T مس موجود در مدارهای پایانی آن است - بدین ترتیب، می توان AT&T را بزرگترین معدن مس دنیا بحساب آورد. خوشبختانه بورس کالا از این موضوع اطلاع چندانی ندارد - اگر این را می دانستند، بلافاصله AT&T را می خریدند، تمام سرویسهای تلفن را قطع می کردند، مس ها را در می آوردند، و دوباره به صنایع مس می فروختند.

وقتی یک مشترک به مشترک دیگری که به همان ایستگاه پایانی وصل است تلفن می زند، دستگاههای سوئیچینگ بین این دو مدار پایانی یک ارتباط الکتریکی مستقیم برقرار می کنند - این ارتباط مستقیم در تمام طول تماس برقرار می ماند.

اما اگر مشترک موردنظر متعلق به ایستگاه پایانی دیگری باشد، روش کار فرق می کند. هر ایستگاه پایانی دارای چند خط ارتباطی با یک یا چند مرکز سوئیچینگ، که اصطلاحاً ایستگاه بین شهری (toll office) نامیده می شوند، است - اگر این ایستگاهها در یک منطقه باشند، به آنها ایستگاه شریک (tandem office) نیز گفته می شود. این خطوط ارتباطی را ترانک های مرتبط کننده بین شهری (toll connecting trunks) می نامند. اگر ایستگاههای پایانی تلفن کننده و تلفن شونده به یک ترانک وصل باشند (که در صورت نزدیکی آنها بسیار محتمل است)، امکان دارد ارتباط در همان ایستگاه بین شهری برقرار شود. در شکل ۲-۲۰ (ج) یک شبکه تلفن ساده را می بینید، که در آن تلفنها با نقاط سیاه کوچک، ایستگاههای پایانی با نقاط سیاه بزرگ، و ایستگاههای بین شهری با مربع نشان داده شده اند.

اگر تلفن کننده و تلفن شونده دارای ایستگاه بین شهری مشترک نباشند، مسیر ارتباطی بایستی در سطح بالاتری برقرار شود. ایستگاههای بین شهری از طریق ایستگاههای اولیه (primary office)، ناحیه ای (sectional office) و منطقه ای (regional office) به هم متصل می شوند. اتصال این ایستگاهها از طریق ترانک بین شهری (intertoll trunk یا interoffice trunk) برقرار می شود. نوع مراکز سوئیچینگ و توپولوژی آنها (مثلاً، اینکه دو ایستگاه ناحیه ای می توانند مستقیماً به یکدیگر وصل شوند، یا باید این ارتباط از طریق یک ایستگاه منطقه ای باشد؟) از کشوری به کشور دیگر فرق می کند، و به تراکم تلفن در آن کشور بستگی دارد. در شکل ۲-۲۱ نحوه هدایت یک تماس راه دور را ملاحظه می کنید.

در مخابرات راه دور از رسانه های مختلفی استفاده می شود. امروزه در مدارهای پایانی از کابل Cat 3 استفاده می شود، در حالیکه در سالهای اولیه اختراع تلفن از سیمهای بدون پوششی که با فاصله ۲۵ سانتیمتر بر فراز تیرهای تلفن کشیده می شد، استفاده می کردند. بین مراکز سوئیچینگ معمولاً کابل کواکسیال، مایکروویو یا فیبر نوری بکار برده می شود.



شکل ۲-۲۱. مدار هدایت یک تماس راه دور.

در گذشته انتقال سیگنالهای تلفن بصورت آنالوگ بود؛ ابتدا صدا به ولتاژ الکتریکی تبدیل شده، و سپس همین سیگنال روی خط تلفن ارسال می شد. با اختراع فیبر نوری، الکترونیک دیجیتال و کامپیوتر، امروزه دیگر تمام ترانکها و سوئیچها دیجیتال هستند، و تنها قسمتی که هنوز از تکنولوژی آنالوگ استفاده می کند، همان مدار پایانی است. مزیت انتقال دیجیتال در اینست که دیگر مشکل اعوجاج سیگنال در تقویت کننده های مختلف وجود ندارد، و فقط کفایت بتوانیم 0 را از 1 تشخیص دهیم. انتقال دیجیتال مطمئنتر، ارزاتر، و نگهداری آن ساده تر است.

بطور خلاصه، سیستم تلفن از سه قسمت عمده تشکیل شده است:

۱. مدارهای پایانی (زوجهای تابیده آنالوگ که به خانه ها و دفاتر کشیده می شوند)
۲. ترانکها (فیبرهای نوری دیجیتال که مراکز سوئیچینگ را به یکدیگر متصل می کنند)
۳. مراکز سوئیچینگ (مراکزی که تماسهای تلفنی را از یک خط اصلی به خط دیگر هدایت می کنند)

اجازه دهید قبل از پرداختن به ادامه این بحث، کمی هم درباره تلفن و سیاست صحبت کنیم. بعد از آن خواهیم دید که مدارهای پایانی و ترانکها چگونه عمل می کنند، و سوئیچینگ چگونه انجام می شود.

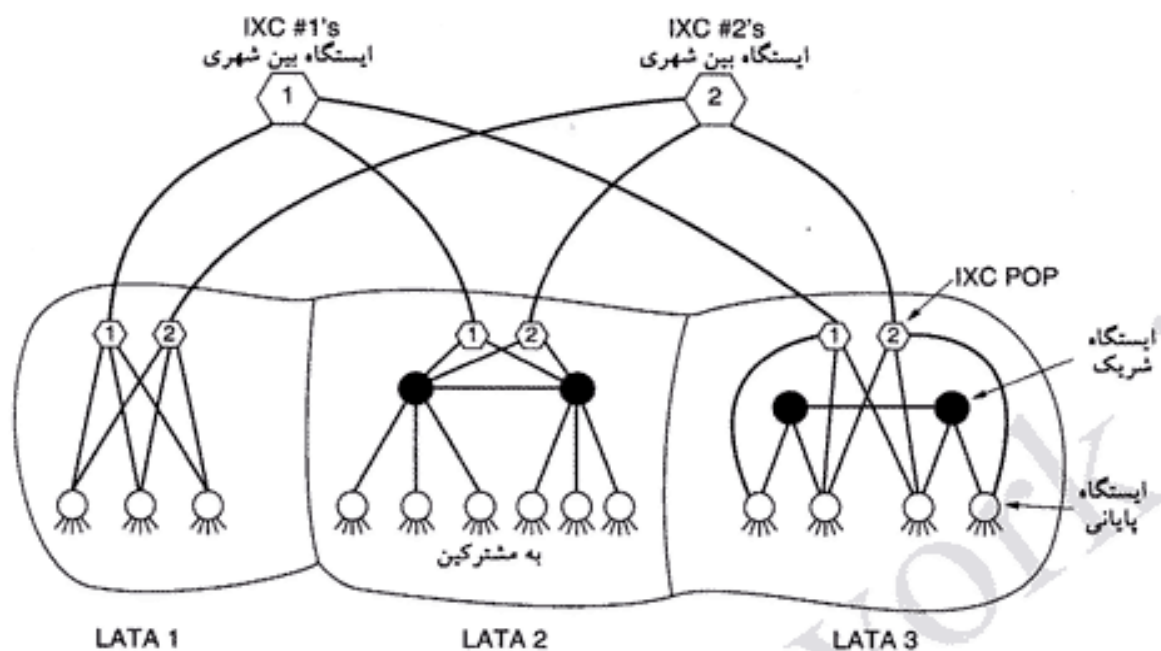
۲-۵-۲ تلفن و سیاست

تا قبل از سال ۱۹۸۴ برای چندین دهه انحصار تلفن شهری و راه دور ایالات متحده آمریکا در انحصار شرکت بل سیستم بود. در دهه ۱۹۷۰ دولت فدرال به این نتیجه رسید که این انحصار غیرقانونی است، و برای شکستن آن به دادگاه شکایت کرد. در اول ژانویه ۱۹۸۴ رأی دادگاه به نفع دولت فدرال صادر شد، و شرکت AT&T به شرکتهای AT&T Long Lines، BOC ۲۳ (شرکتهای Bell Operating Company)، و چند شعبه کوچک دیگر شکسته شد. شرکتهای BOC برای آن که بتوانند در بازار رقابت کنند، هفت گروه منطقه ای (RBOC) تشکیل دادند. این رأی (که به Modified Final Judgment - MFJ معروف است) مخابرات راه دور ایالات متحده را یک شبه دچار دگرگونی اساسی کرد.

دستور MFJ منجر به تغییرات مهمی در سرویسهای مخابراتی شد: افزایش رقابت بین شرکتهای مخابراتی، بهبود سرویسها، و کاهش قیمت مخابرات راه دور. از طرف دیگر قیمت سرویسهای شهری بشدت بالا رفت، چون این شرکتها بایستی بنوعی کاهش درآمد خود را (که قبلاً از درآمد مخابرات راه دور تأمین می شد) جبران می کردند. بسیاری از کشورهای دیگر نیز در حال پیروی از این مدل هستند.

برای آن که مشخص شود چه کسی مجاز به انجام چه کاریست، ایالات متحده به ۱۶۴ LATA (مناطق محلی دسترسی و انتقال - Local Access and Transport Area) تقسیم شد. تقسیم بندی LATA ها تا حد زیادی (اما نه کاملاً) بر گندهای منطقه (area code) منطبق است. هر LATA دارای یک LEC (کاربر تبادل محلی - Local Exchange Carrier) است که انحصار کامل سرویسهای تلفن در آن منطقه را در اختیار دارد. مهمترین این LEC ها همان BOC ها هستند، ولی در برخی از مناطق شرکتهای مستقل نیز (که تعداد آنها به ۱۵۰۰ می رسد) فعالیت دارند.

کلید ترافیک بین LATA ها توسط شرکتهایی بنام IXC (کاربر تبادل - InterExchange Carrier) انجام می شود. تا مدتی قبل AT&T Long Lines تنها IXC فعال در بازار بود، ولی اکنون شرکتهای بزرگی مانند Sprint و WorldCom نیز در این زمینه فعال هستند. یکی از دغدغه هایی که بعد از شکسته شدن AT&T وجود داشت آن بود که IXC ها در زمینه کیفیت خط، تعرفه ها، و تعداد رقمهای پیش شماره یکسان باشند. روش این در شکل ۲-۲۲ نشان داده شده است؛ در این شکل سه LATA با تعدادی ایستگاه پایانی می بینید. LATA های ۲ و ۳ با ایستگاههای بین شهری نیز ارتباط دارند.



شکل ۲-۲۲. ارتباط بین LATA ها و LEC ها و IXC ها

هر IXC که بخواهد مجری تماسهای یک LATA باشد، یک ایستگاه سونیچینگ بنام POP (نقطه تماس - Point Of Presence) در آنجا تأسیس می کند. وظیفه برقراری تماس IXC با ایستگاههای پایانی بر عهده LEC آن منطقه است (خواه بصورت مستقیم مانند LATA های ۱ و ۳، یا بصورت غیرمستقیم مانند LATA ۲). تماس (خواه فنی یا مالی) باید برای تمام IXC ها یکسان باشد. بدین ترتیب، مشترکی که مثلاً در LATA ۱ است، می تواند تصمیم بگیرد توسط کدام IXC با مشترکی در LATA ۳ تماس بگیرد.

در MFJ تصریح شده است که IXC ها نباید وارد بازار تلفن محلی شوند، و LEC ها هم از حضور در سرویسهای راه دور منع شده اند (البته آنها می توانند هر کار دیگری انجام دهند، مثلاً چپس و پفک بفروشند). در سال ۱۹۸۴ این دستور نسبتاً واضح بود، ولی تکنولوژی همواره راههای جالبی برای منسوخ کردن قوانین پیدا می کند. از آنجائیکه تلفنهای همراه و تلویزیون کابلی مشمول دستور MFJ نمی شوند، LEC ها و IXC ها بسمت خرید این شرکتها (یا ادغام با آنها) روی آوردند.

در سال ۱۹۹۵ کنگره متوجه شد که جدا نگه داشتن حوزه فعالیت شرکتهای مختلف دیگر عملی نیست، و بهمین دلیل با تصویب یک لایحه به شرکتهای تلفن شهری، راه دور و تلویزیون کابلی اجازه داد تا وارد بازارهای یکدیگر شوند. ایده اصلی این لایحه اینست که سرویسهای صدا، داده و تلویزیون کابلی را یکپارچه کرده، و باعث رقابت بین شرکتهای مختلف (برای سرویس بهتر و قیمت کمتر) شود. این قانون از فوریه ۱۹۹۶ به اجرا گذاشته شد، و باعث شد تا تعدادی از BOC ها وارد حوزه IXC شوند، و از طرف دیگر شرکتهای تلویزیون کابلی نیز به ارائه سرویسهای تلفن بپردازند (و با LEC ها رقابت کنند).

یکی از نکات جالب قانون ۱۹۹۶ آنست که LEC ها بایستی شماره های تلفن را به گونه ای تنظیم کنند که قابل انتقال باشند. این بدان معناست که یک مشترک می تواند از این منطقه به منطقه دیگر برود، بدون آنکه نیازی باشد شماره تلفنش را عوض کند. این ویژگی باعث می شود تا افراد آزادی بیشتری در عوض کردن LEC خود داشته باشند، و در نتیجه تنور رقابت داغتر شود. با قانون ۱۹۹۶، مخابرات ایالات متحده دستخوش تغییرات ساختاری شدیدی شده است. کشورهای بسیاری نیز ترغیب شده اند تا دست به چنین اقداماتی بزنند. البته در اغلب

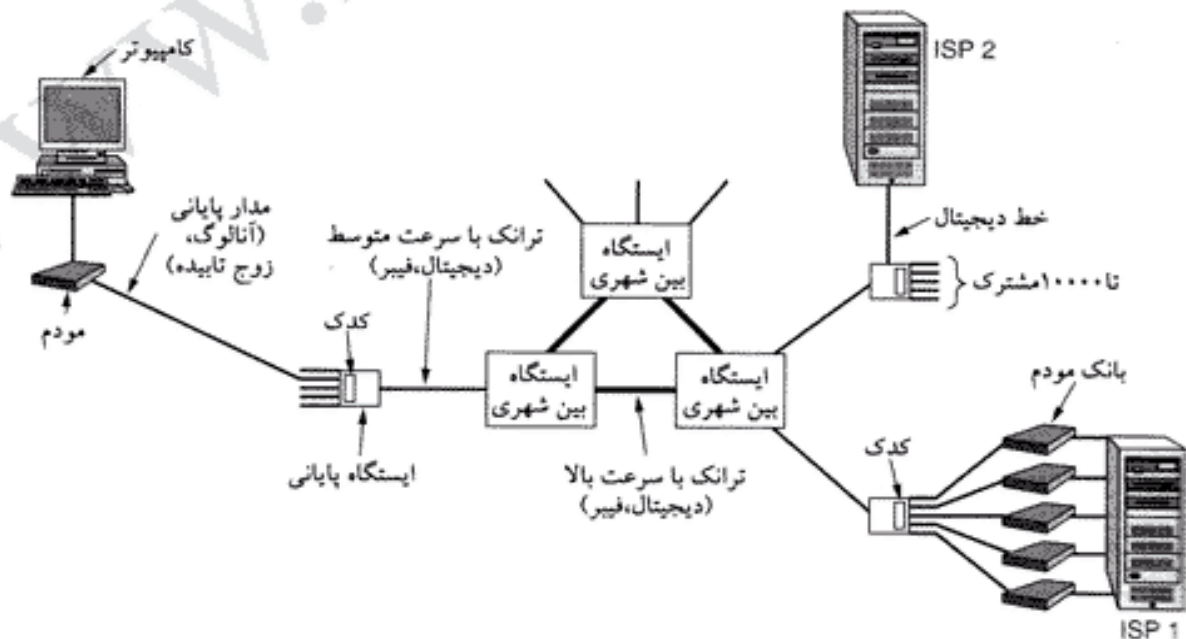
موارد کشورهای دیگر صبر می کنند تا نتیجه کار در ایالات متحده مشخص شود؛ اگر نتیجه مثبت بود، همان راه را می روند؛ اگر نتیجه منفی بود، راه دیگری را امتحان می کنند.

۳-۵-۲ مدارهای پایانی: مودم، ADSL، ویسیم

اکنون زمان آن است که تا به بررسی نحوه کار سیستم تلفن پردازیم. بخشهای اصلی این سیستم در شکل ۲-۲۳ نشان داده شده است. در این شکل مدارهای پایانی، ترانک ها، ایستگاه های بین شهری و ایستگاههای پایانی را می بینید. هر ایستگاه پایانی (در ایالات متحده و کشورهای بزرگ) تا ۱۰,۰۰۰ مدار پایانی دارد. در حقیقت تا همین اواخر، کد منطقه + شماره ناحیه مشخص کننده ایستگاه پایانی بود - برای مثال، 601-xxxx (212) یک ایستگاه پایانی با ۱۰,۰۰۰ مشترک (از ۰۰۰۰ تا ۹۹۹۹) است. با ایجاد شدن امکان رقابت بر سر سرویسهای محلی دیگر این وضعیت عملی نیست، چون شرکتهای زیادی طالب بدست آوردن کدهای محلی هستند. روشهای شماره گذاری نیز بایستی بکلی عوض شود، چون اغلب کدهای منطقه ای مصرف شده اند.

اجازه دهید با بخشی که بیشتر مردم با آن آشنا نیست، شروع کنیم: دو رشته سیمی که از ایستگاه پایانی شرکت تلفن به محل مشترک (خانه یا محل کار) کشیده می شود. به این دو رشته سیم اغلب «کیلومتر آخر» گفته می شود، اگرچه طول آن ممکنست به چندین کیلومتر هم برسد. در طول یکصد سال گذشته در این بخش از سیگنالهای آنالوگ استفاده شده است، و (بدلیل هزینه زیاد تجهیزات دیجیتال) احتمالاً در چند سال آینده نیز وضع به همین منوال خواهد بود. با این حال، در این آخرین سنگر آنالوگ نیز تغییرات شروع شده است. در این قسمت مدارهای پایانی و آخرین تحولات آنرا (با تأکید بر مخابرات داده بین کامپیوترها) بتفصیل بررسی خواهیم کرد.

وقتی یک کامپیوتر می خواهد داده های دیجیتال را روی یک خط تلفن آنالوگ ارسال کند، ابتدا باید آنها را به سیگنالهای آنالوگ تبدیل کند - کاری که با دستگاهی بنام مودم (modem) انجام می شود. در ایستگاه پایانی این اطلاعات مجدداً به سیگنالهای دیجیتال تبدیل شده، و برای ارسال روی ترانک راه دور (trunk) آماده می شود. اگر مقصد اطلاعات یک کامپیوتر باشد، سیگنال مجدداً به آنالوگ تبدیل می شود تا بتوان آنرا روی مدار پایانی



شکل ۲-۲۳. ترکیبی از انتقال دیجیتال و آنالوگ برای تماس کامپیوتر با کامپیوتر. تبدیل سیگنال بوسیله مودمها و کدکها انجام می شود.

آن ارسال کرد؛ و در مقصد یک مودم دیگر اطلاعات را از آنالوگ به دیجیتال تبدیل می کند. در شکل ۲-۲۳، ISP 1 (ارائه دهنده سرویس های اینترنتی) یک بانک مودم دارد، که هر کدام از آنها به یک خط تلفن (مدار پایانی) مستقل متصلند. این ISP می تواند در آن واحد به تعداد مودم های خود به افراد مختلف سرویس بدهد (البته با این فرض که کامپیوترهایش به اندازه کافی قوی باشند). این آرایش تا وقتی که مودم های 56-kbps وارد بازار شدند، آرایش متداولی بود (بزودی علت آنرا خواهید فهمید).

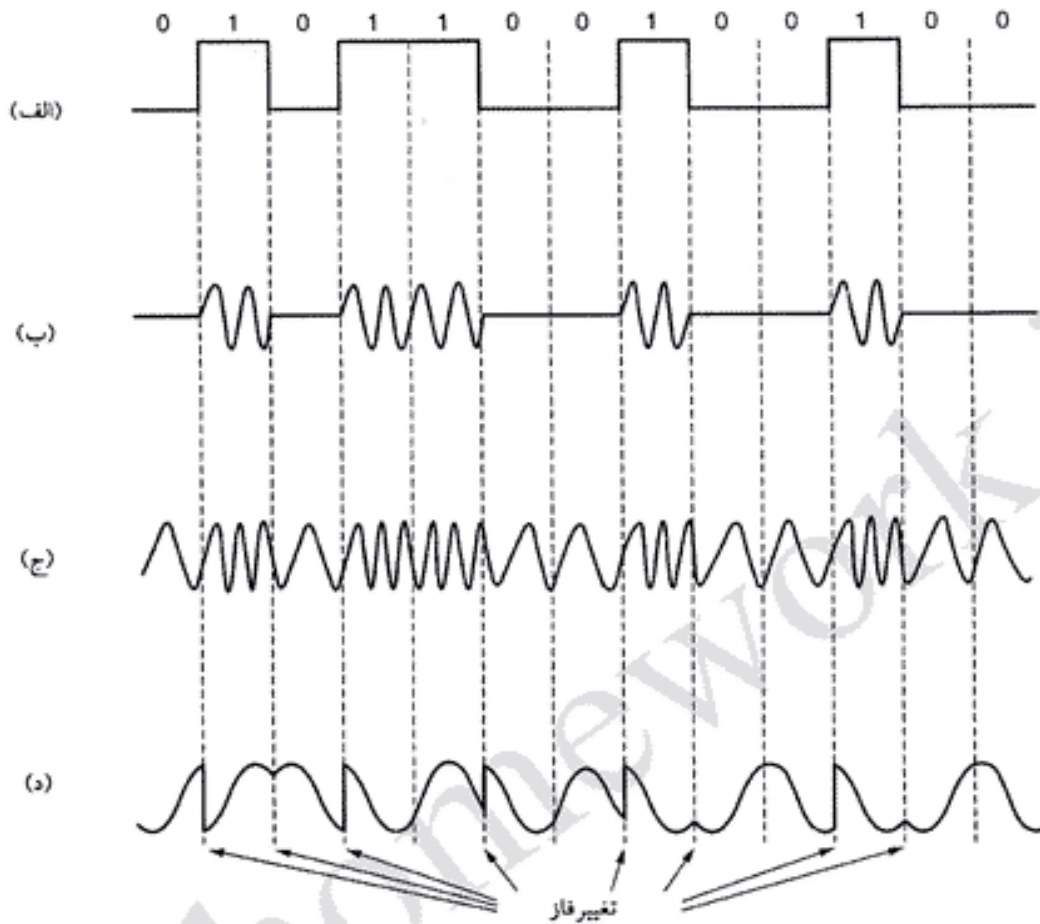
در مخابرات آنالوگ برای انتقال اطلاعات از ولتاژی که با زمان تغییر می کند، استفاده می شود. اگر رسانه انتقال کامل و بدون نقص باشد، گیرنده دقیقاً همان سیگنال را دریافت خواهد کرد که فرستنده ارسال کرده است. اما متأسفانه چنین نیست، و گیرنده همان سیگنال ارسالی را دریافت نمی کند - و در مخابرات دیجیتال این یعنی خطا. انتقال سیگنال های الکتریکی روی خطوط انتقال با سه مشکل عمده روبروست: تضعیف سیگنال، اعوجاج تأخیری، و نویز. وقتی یک سیگنال در رسانه انتقال منتشر می شود، انرژی خود را از دست می دهد که به آن تضعیف (attenuation) می گویند، و بر حسب دسی بل بر کیلومتر اندازه گیری می شود. میزان تضعیف یک سیگنال به فرکانس آن بستگی دارد. شاید فکر کنید که این وابستگی به فرکانس اشکال زیادی ایجاد نمی کند، ولی وقتی یک موج را بصورت مجموعه ای از مؤلفه های فوریه در نظر بگیرید، تأثیر وابستگی به فرکانس خود را نشان خواهد داد. در واقع هر یک از مؤلفه های فوریه بگونه ای متفاوت تضعیف می شوند، و ترکیب مجدد آنها در گیرنده موج کاملاً متفاوتی ایجاد می کند.

اما وضع از این هم بدتر است، چون مؤلفه های فوریه با سرعت های متفاوتی در رسانه انتقال (سیم) منتشر می شوند. این تفاوت سرعتها باعث اعوجاج تأخیری (delay distortion) سیگنال در گیرنده می شود. مشکل دیگر نویز (noise) - انرژی ناخواسته از منابعی غیر از فرستنده - است. نویز حرارتی حاصل حرکات تصادفی الکترون ها در سیم است، و بکلی نمی توان از آن اجتناب کرد. نویز القایی نیز حاصل القای ولتاژ در اثر عبور جریان از سیم های مجاور است. گاهی پیش آمده که وقتی تلفنی با کسی صحبت می کنید، مکالمه دیگری را نیز در زمینه می شنوید؛ علت این پدیده (که به همشنوایی - crosstalk - مشهور است) نویز القایی می باشد. در اثر قطع و وصل خطوط قدرت نویز دیگری بنام نویز ضربه ای روی خطوط مخابراتی القا می شود، که می تواند چندین بیت از اطلاعات را از بین ببرد.

مودم

بدلیل مشکلاتی که در بالا گفته شد (بویژه وابستگی تضعیف سیگنال و سرعت انتشار آن به فرکانس)، سعی می شود از سیگنال هایی با محدوده فرکانسی پائین استفاده شود. متأسفانه، شکل موج مربعی سیگنال های دیجیتال دارای طیف فرکانسی وسیعی است، و بشدت در معرض تضعیف و اعوجاج تأخیری قرار دارد. این تأثیرات باعث شده تا سیگنال های بیس باند (DC) فقط برای سرعت های پائین و مسافت های کوتاه مناسب باشد.

برای حل مشکل سیگنال های DC بویژه در خطوط تلفن، از سیگنال AC استفاده می شود. در این جا یک تون ۱۰۰۰ تا ۲۰۰۰ هرتزی بعنوان موج حامل سینوسی (sine wave carrier) بکار برده می شود. برای انتقال اطلاعات می توان دامنه، فرکانس یا فاز این موج حامل را مدوله کرد. در مدولاسیون دامنه (amplitude modulation) از دو دامنه متفاوت بعنوان 0 و 1 استفاده می شود. در مدولاسیون فرکانس (frequency modulation) - که به کدگذاری با شیف فرکانس (frequency shift keying) نیز معروف است - از دو تون متفاوت برای 0 و 1 استفاده می شود. (در صنعت مخابرات، اصطلاحات مدولاسیون و کدگذاری معادل یکدیگرند.) در مدولاسیون فاز (phase modulation) موج حامل در فواصل یکنواخت 0 یا 180 درجه شیف پیدا می کند. با استفاده از شیفتهای 45، 135، 225 یا 315 درجه ای می توان در هر فاصله زمانی بجای یک بیت، ۲



شکل ۲-۲۴. (الف) سیگنال باینری. (ب) مدولاسیون دامنه. (ج) مدولاسیون فرکانس. (د) مدولاسیون فاز.

بیت اطلاعات منتقل کرد. همچنین، وجود تغییر فاز در انتهای هر فاصله زمانی تشخیص مرزهای آنها را برای گیرنده آسانتر می‌کند.

شکل ۲-۲۴ این سه نوع مدولاسیون را نشان می‌دهد. در شکل ۲-۲۴ (ب) یکی از دامنه‌ها صفر و دیگری غیر صفر است. در شکل ۲-۲۴ (ج) از دو فرکانس متفاوت برای نمایش ۰ و ۱ استفاده شده است. در شکل ۲-۲۴ (د) در هر فاصله زمانی وجود (یا عدم وجود) تغییر فاز نشان‌دهنده تغییر مقدار بیت (یا عدم تغییر آن) است.

دستگاهی که جریان بیت‌ها را بعنوان ورودی گرفته، و با ایجاد یک موج حامل و اعمال یکی از انواع مدولاسیون (یا ترکیبی از آنها) یک خروجی آنالوگ تولید می‌کند (و یا بر عکس، با گرفتن موج آنالوگ اطلاعات دیجیتال را از آن استخراج می‌کند)، مودم (modulator-demodulator) نامیده می‌شود. مودم بین کامپیوتر (دیجیتال) و سیستم تلفن (آنالوگ) قرار می‌گیرد.

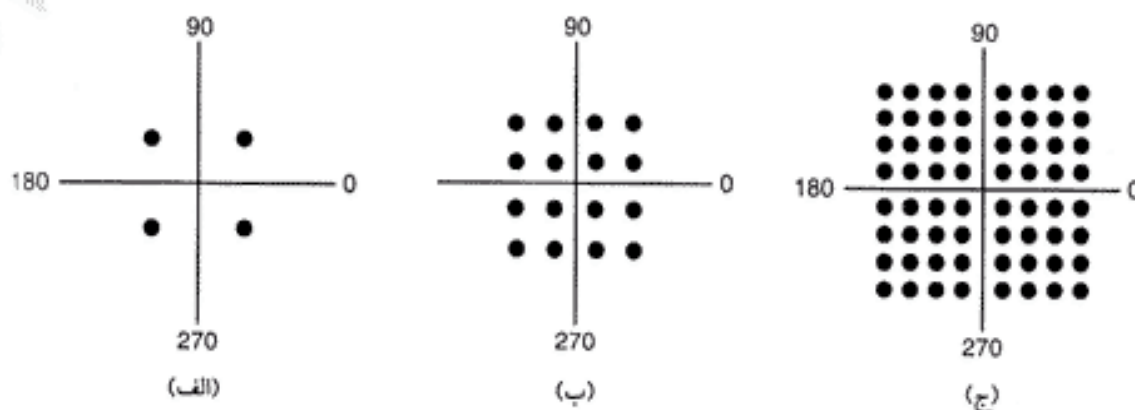
بالا بردن سرعت پسادگی و فقط با زیاد کردن نرخ نمونه‌برداری (sampling rate) ممکن نیست. قضیه ناپکوئیست می‌گوید که برای یک خط کامل 3000 Hz (که خطوط تلفن مسلماً چنین نیستند)، حداکثر نرخ نمونه‌برداری 6000 Hz است. در عمل، اکثر مودمها با نرخ 2400 times/sec نمونه‌برداری می‌کنند، ولی سعی می‌کنند در هر نمونه‌برداری بیت‌های بیشتر را بخوانند.

به تعداد نمونه ها در ثانیه باد (baud) گفته می شود، و در هر باد یک سمبل (symbol) فرستاده می شود. بنابراین، یک خط n -baud در هر ثانیه n سمبل ارسال می کند (برای مثال، یک خط 2400-baud در هر 416.667 μsec یک سمبل می فرستد). اگر این سمبل فقط حاوی ولتاژ 0 v (برای نمایش 0 منطقی) یا 1 v (برای نمایش 1 منطقی) باشد، سرعت مودم 2400 bps خواهد بود. اما اگر از ولتاژهای 0، 1، 2 و 3 ولت استفاده کنیم، هر سمبل می تواند حاوی 2 بیت باشد، و سرعت انتقال اطلاعات مودم به 4800 bps می رسد. در مدولاسیون فاز چهار درجه ای نیز می توان در هر سمبل 2 بیت ارسال کرد، و بدین ترتیب سرعت ارسال داده دو برابر سرعت باد خواهد بود. این تکنیک که کاربرد زیادی گسترده ای نیز دارد، QPSK (کُدگذاری با شیفت فاز چهارگانه - Quadrature Phase Shift Keying) نامیده می شود.

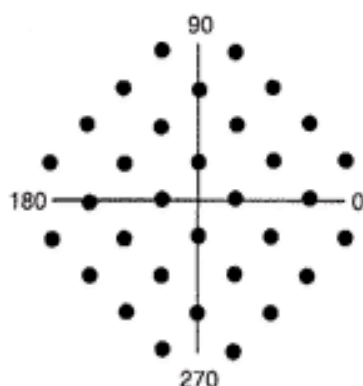
مفاهیم پهنای باند، باد، سمبل، و نرخ بیت بسیار با هم اشتباه می شوند، بنابراین اجازه دهید یک بار دیگر آنها را بیان کنیم. پهنای باند یک رسانه محدوده فرکانسی است که می تواند سیگنال را با کمترین تضعیف عبور دهد. این یکی از ویژگیهای فیزیکی رسانه انتقال است، و با هرتز (Hz) سنجیده می شود. به تعداد نمونه برداری در هر ثانیه باد گفته می شود، و هر نمونه حاوی یک قطعه از اطلاعات (یا سمبل) است. بنابراین، نرخ باد (baud rate) و نرخ سمبل (symbol rate) در واقع یکی هستند. تعداد بیت بر سمبل توسط نوع مدولاسیون (مثلاً، QPSK) تعیین می شود. نرخ بیت (bit rate) مقدار اطلاعاتیست که روی یک کانال فرستاده می شود، و برابر است با نرخ سمبل \times (symbol/sec) \times تعداد بیت در هر سمبل (bits/symbol).

تمام مودمهای پیشرفته برای ارسال بیشترین بیتهای ممکن در هر باد، از مدولاسیونهای ترکیبی (چند دامنه ای و چند فازی) استفاده می کنند. در شکل ۲-۲۵ (الف) نقاطی را می بینید که در فواصل یکسان از مبدأ مختصات (دامنه یکسان)، و با زاویه ها (فازها) 45، 135، 225 و 315 درجه قرار گرفته اند (فاز هر نقطه زاویه ایست که با جهت مثبت محور x می سازد). در شکل ۲-۲۵ (الف) چهار ترکیب معتبر وجود دارد، که بدین ترتیب می توان در هر سمبل 2 بیت ارسال کرد - این همان QPSK است.

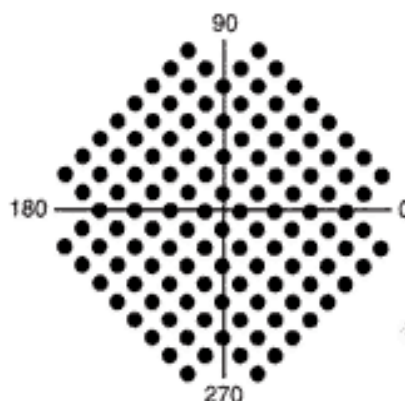
در شکل ۲-۲۵ (ب) مدولاسیون دیگری را می بینید، که در آن از چهار دامنه و چهار فاز مختلف استفاده شده و 16 ترکیب معتبر بدست می دهد. با این مدولاسیون می توان در هر سمبل چهار بیت ارسال کرد، و به آن QAM-16 (مدولاسیون دامنه چهارگانه - Quadrature Amplitude Modulation) گفته می شود (گاهی نیز 16-QAM خوانده می شود). با مدولاسیون QAM-16 می توان روی یک خط 2400-baud تا سرعت 9600 bps داده ارسال کرد.



شکل ۲-۲۵. (الف) مدولاسیون QPSK، (ب) مدولاسیون QAM-16، (ج) مدولاسیون QAM-64.



(ب)



(ج)

شکل ۲-۲۶. (الف) استاندارد V.32 برای 9600 bps. (ب) استاندارد V.32 bis برای 14,400 bps.

در شکل ۲-۲۵ (د) مدولاسیون دیگری را می بینید که شامل دامنه های بیشتری است. در این طرح ۶۴ ترکیب ممکنه وجود دارد، که بدین ترتیب می توان به ازای هر سمبل ۶ بیت را ارسال کرد. این مدولاسیون QAM-64 نام دارد (از مدولاسیون QAM با مراتب بالاتر نیز استفاده می شود).

دیاگرامهایی مانند شکل ۲-۲۵ که ترکیبات ممکنه دامنه و فاز را نشان می دهند، به دیاگرام فلکی (constellation diagram) معروفند. هر استاندارد مودمهای سرعت-بالا دارای دیاگرام فلکی خاص خود است، و فقط قادر به ارتباط با مودمهاییست که دارای دیاگرام مشابهی باشند (البته این مودمها می توانند تمام سرعتهای پائینتر را شبیه سازی کنند).

با بالا رفتن تعداد نقاط در دیاگرام فلکی حتی نویزهای کوچک نیز می توانند باعث بروز خطا در آشکارسازی دامنه یا فاز سیگنال شوند، که بدنبال آن بیتهای زیادی از دست می رود. برای کاهش احتمال خطا، در مودمهای سرعت-بالا با اضافه کردن بیت های اضافی نوعی تصحیح خطا (error correction) انجام می شود. به این روش TCM (مدولاسیون کدگذاری تاروپودی - Trellis Coded Modulation) می گویند. برای مثال، استاندارد V.32 از ۳۲ نقطه فلکی برای ارسال ۴ بیت داده و یک بیت برابری (parity) در هر سمبل استفاده کرده، و با نرخ 2400-baud به سرعت 9600 bps (با تصحیح خطا) دست می یابد. دیاگرام فلکی این استاندارد را در شکل ۲-۲۶ (الف) مشاهده می کنید. (چرخش ۴۵ درجه ای حول محور مختصات فقط بدلیل مهندسی اتخاذ شده است، و هیچگونه تأثیری روی ظرفیت اطلاعات ندارد).

بعد از 9600 bps قدم بعدی 14,400 bps است، که استاندارد آن V.32 bis نامیده می شود. برای رسیدن به این سرعت باید (با نرخ 2400-baud) در هر سمبل ۶ بیت داده و ۱ بیت برابری ارسال شود. دیاگرام فلکی این مودم را، که (وقتی از QAM-128 استفاده شود) ۱۲۸ نقطه دارد، در شکل ۲-۲۶ (ب) می بینید. مودمهایی که قابلیت فکس دارند، برای ارسال فکس از این استاندارد استفاده می کنند. مدولاسیون QAM-256 در هیچ یک از مودمهای تلفنی استفاده نمی شود، ولی در شبکه های کابلی کاربرد دارد (بعداً آنرا خواهید دید).

استاندارد مودم تلفنی بعدی V.34 است، که با سرعت 28,800 bps (12 data bits/symbol در 2400-baud) کار می کند. آخرین مودم در این سری استاندارد V.34 bis نام دارد، که به سرعت 33,600 bps (14 data bits/symbol در 2400-baud) دست پیدا می کند.

برای رسیدن به سرعتهای بیشتر، بسیاری از مودمها از تکنیکهای فشرده سازی استفاده می کنند تا به سرعتهای

بالتر از 33,600 bps برسند. از سوی دیگر، تقریباً تمام مودمها قبل از شروع ارسال داده‌ها کیفیت خط را چک می‌کنند، و اگر نقصی در کیفیت خط وجود داشته باشد، سرعت خود را آنقدر پائین می‌آورند تا ارسال مطمئن داده‌ها امکانپذیر باشد. بهمین دلیل، سرعت مؤثر یک مودم می‌تواند کمتر، مساوی، و یا بیشتر از سرعت رسمی آن باشد.

تمام مودمهای جدید (با استفاده از فرکانسهای متفاوت برای ارسال و دریافت) اجازه می‌دهند تا ارسال و دریافت همزمان انجام شود. به چنین ارتباطی دو-طرفه همزمان (full duplex) گفته می‌شود، مانند یک اتوبان دو-سبانه. اگر در هر لحظه فقط از یک طرف ارتباط ممکن باشد، به آن دو-طرفه ناهمزمان (half duplex) گفته می‌شود (مانند راه‌آهن‌های یک-خطه). و اگر ارتباط فقط در یک جهت مجاز باشد، به آن یکطرفه (simplex) می‌گویند (مانند یک خیابان یکطرفه). یک رشته فیبر نوری که در یک سمت فقط دیود لیزری و در سمت دیگر فقط آشکارساز نوری دارد، نیز سیستمی یکطرفه است.

همانطور که قبلاً گفتیم، طبق قانون شانون سرعت انتقال روی خطوط تلفن از 35-kbps نمی‌تواند فراتر رود، بهمین مودمهای استاندارد از سرعت 33,600 bps بالاتر نمی‌روند. شاید پی رسید پس مودمهای 56-kbps چطور کار می‌کنند؟ کمی صبر کنید، به آن هم خواهیم رسید.

اما این حد 35-kbps از کجا آمده است؟ این محدودیت به طول متوسط مدارهای پایانی و کیفیت آنها بستگی دارد. به شکل ۲-۲۳ نگاه کنید: ارتباطی که از کامپیوتر سمت چپ شروع شده و به 1 ISP ختم شود، از دو مدار پایانی آنالوگ (یکی در مبدأ و دیگری در مقصد) عبور می‌کند. هر یک از این حلقه‌ها میزان نویز سیگنال را بالا می‌برند. اگر بتوانیم یکی از این حلقه‌ها را حذف کنیم، می‌توانیم حداکثر سرعت را به دو برابر برسانیم.

این دقیقاً همان کاریست که 2 ISP انجام داده: این ISP ارتباط خود با نزدیکترین ایستگاه پایانی را بصورت کاملاً دیجیتال در آورده است. سیگنال دیجیتال ترانک مستقیماً به 2 ISP فرستاده شده، و مودمها و کودکهای آنالوگ بکلی حذف شده‌اند. بدین ترتیب، حداکثر سرعت ارتباط با این ISP می‌تواند به 70-kbps نیز برسد. بین دو نقطه که از مودم و خطوط آنالوگ استفاده می‌کنند، حداکثر سرعت همان 33.6 kbps است.

اما علت اینکه مودمهای 56-kbps می‌توانند در عمل کار کنند، به قضیه نایکوئیست برمی‌گردد. پهنای باند کانالهای تلفنی (بأنضمام باند محافظ) 4000 Hz است، و طبق قضیه نایکوئیست حداکثر نرخ نمونه‌برداری در چنین کانالی 8000 خواهد بود. در ایالات متحده آمریکا تعداد بیت‌های هر نمونه ۸ است، که (با احتساب یک بیت کنترلی) حداکثر سرعت مجاز به 56,000 bits/sec می‌رسد. در اروپا از بیت کنترلی استفاده نمی‌شود، و می‌توان تمام بیت‌ها را به داده‌ها اختصاص داد، بنابراین حداکثر سرعت در آنجا می‌تواند تا 64,000 bits/sec افزایش یابد، ولی طبق یک توافق بین‌المللی همان سرعت 56,000 انتخاب شده است.

این استاندارد V.90 نامیده می‌شود. در این استاندارد سرعت ارسال کاربر به ISP همان 33.6 kbps است، ولی سرعت دریافت از ISP به 56-kbps می‌رسد (چون معمولاً آنچه که کاربر از ISP می‌گیرد، بسیار بیشتر از آن چیزیست که به آن می‌فرستند). از لحاظ نظری امکان رساندن کانال ارسال به ISP تا 56-kbps نیز وجود داشت، ولی (بعلت نویزی بودن خطوط تلفنی) تصمیم گرفته شد تا این کانال به 33.6 kbps محدود شده، و پهنای باند آن به کانال دریافت از ISP داده شود تا احتمال رسیدن آن به سرعت 56-kbps افزایش یابد.

قدم بعدی در این بازی استاندارد‌ها، V.92 است. مودمهای این استاندارد می‌توانند تا 48-kbps روی کانال ارسال داده بفرستند (البته مشروط باینکه خط تلفن توانایی آنرا داشته باشد). زمان شناسایی کیفیت خط و تعیین سرعت مناسب در این استاندارد نیز بسیار کمتر از مودمهای دیگر است (تقریباً نصف ۳۰ ثانیه‌ای که مودمهای دیگر صرف این کار می‌کنند). و بالاخره، مودمهای استاندارد V.92 اجازه می‌دهند تا یک تماس تلفنی بتواند تماس اینترنتی را قطع کند، مشروط باینکه خط دارای سرویس انتظار مکالمه (call waiting) باشد.

خط مشترک دیجیتال (DSL)

وقتی شرکت های تلفن بالاخره موفق شدند سرویس 56-kbps ارائه کنند، خیلی به خود غره شدند؛ اما در همان حال، شرکت های تلویزیون کابلی ارتباطاتی با سرعت 10 Mbps (و ماهواره ها سرعت ارسال 50 Mbps) عرضه می کردند. با داغ شدن بازار اینترنت، شرکت های تلفن (LEC) دریافتند که برای رقابت به محصول جدیدی نیاز دارند. پاسخ آنها به این وضعیت ارائه سرویس های دیجیتال روی مدارهای پایانی بود. این سرویس ها پهنای باند بیشتری داشتند و به آنها سرویس باند-وسیع (broadband) گفته می شد (اگرچه این نامگذاری بیشتر جنبه تبلیغاتی داشت، تا فنی).

در ابتدا این سرویس ها بسیار متنوع بودند، و تحت نام xDSL (خط مشترک دیجیتال: Digital Subscriber Line - که در آن x نوع سرویس را مشخص می کند) دسته بندی می شدند. مهمترین این سرویس ها - که عامل اصلی موفقیت آن هم بود - ADSL (DSL نامتقارن - Asymmetric DSL) نام دارد، که در زیر با آن آشنا خواهید شد. از آنجائیکه ADSL هنوز در حال توسعه و تکامل است و استانداردهای آن هنوز بطور کامل تدوین نشده، ممکنست در آینده تغییراتی در آن رخ دهد، ولی تصویر کلی همان است که خواهید دید. برای اطلاعات بیشتر درباره ADSL به (Summers, 1999; Vetter et al., 2000) نگاه کنید.

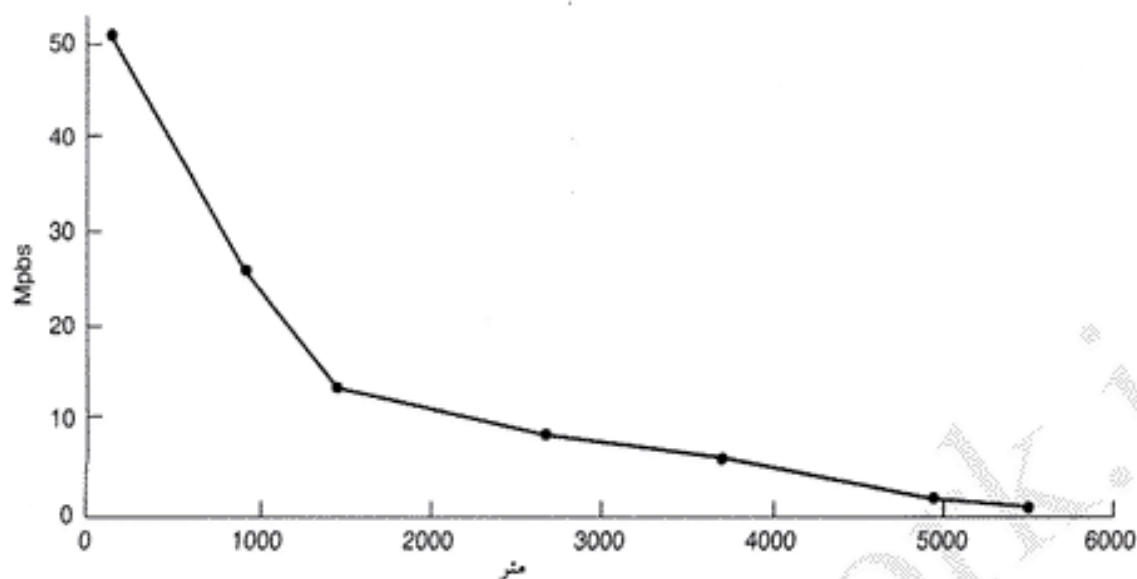
علت آن که مودمها اینقدر کُند هستند اینست که، شبکه تلفن اساساً برای انتقال صدای انسان طراحی و توسعه داده شده، و سرویس های داده فرزندخوانده آن محسوب می شود. در نقطه ای که مدار پایانی وارد ایستگاه پایانی تلفن می شود، فیلترهایی قرار داده شده که تمام فرکانسهای زیر 300 Hz و بالای 3400 Hz را تضعیف می کنند. البته قطع فرکانس بیکباره صورت نمی گیرد - از لحاظ فنی، 300 Hz و 3400 Hz نقاط 3 dB هستند - بهمین دلیل پهنای باند را معمولاً 4000 Hz فرض می کنند، اگرچه فاصله نقاط 3 dB فقط 3100 Hz است. داده نیز به همین باند باریک محدود است.

برای اجتناب از این وضعیت در سرویس xDSL، خط مشترک بدون عبور از فیلترهای مزبور مستقیماً به نوع خاصی از سوئیچها متصل می شود، تا بتواند از تمام ظرفیت مدار پایانی استفاده کند. در این حالت دیگر محدودیت پهنای باند فقط به خواص فیزیکی مدار پایانی بستگی دارد، نه به محدوده ای که فیلترها بطور مصنوعی برای آن بوجود آورده اند.

متأسفانه، ظرفیت مدار پایانی نامحدود نیست، و به عواملی از قبیل طول خط، ضخامت سیم، و کیفیت کلی آن بستگی دارد. در شکل ۲-۲۷ نمودار پهنای باند بر حسب مسافت را ملاحظه می کنید - در اینجا فرض شده که سایر عوامل پهنه هستند (سیمهای نو، کابل های نه چندان قطور، و غیره).

نمودار فوق مشکل اصلی شرکت های تلفن را بخوبی نشان می دهد: شعاع ارائه این سرویس به مشترکان بشدت محدود است. این بدان معناست که وقتی کاربری که خارج از این شعاع زندگی می کند، برای دریافت سرویس xDSL مراجعه کند، باید با کمال تأسف از او عذرخواهی کنیم که امکان ارائه سرویس به وی را نداریم. برای بیشتر کردن شعاع سرویس، باید سرعت آنرا پائین بیاوریم، ولی پائین آوردن سرعت همان و از دست دادن جذابیت همان. اینجاست که تکنولوژی مغلوب اقتصاد می شود. (راه حل دیگر آنست که ایستگاه های کوچک و پراکنده ای در نقاط نزدیک به هم تأسیس کنیم، که البته این هم اقتصادی نیست.)

سرویس xDSL با اهداف مشخصی طراحی شده است: اول اینکه، این سرویس ها باید بتوانند با خطوط زوج تاییده Cat 3 کار کنند؛ دوم اینکه، این سرویس ها نباید هیچ اختلالی در دستگاه های تلفن و فکس معمولی بوجود آورند؛ سوم اینکه، باید از 56-kbps سریعتر باشند؛ و چهارم اینکه، این سرویس ها باید دائماً برقرار باشند، و هزینه آنها هم ثابت (و مثلاً ماهیانه) باشد - نه مانند تلفن های معمولی، بصورت دقیقه ای.

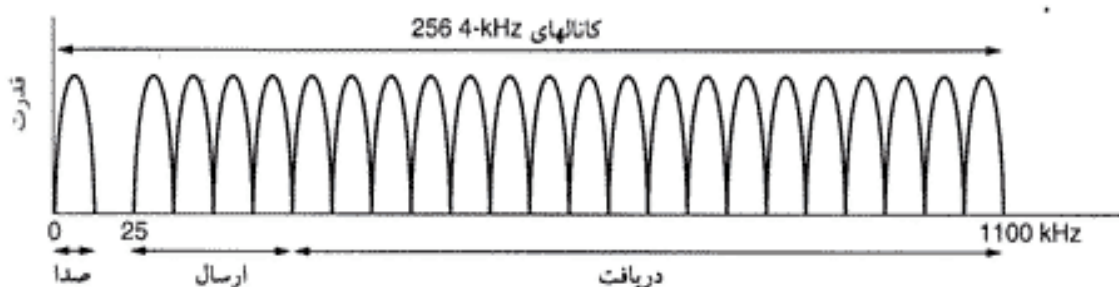


شکل ۲-۲۷. نمودار پهنای باند سرویس DSL بر حسب طول سیم در کابل Cat 3 UTP.

اولین سرویس ADSL توسط AT&T ارائه شد، که در آن پهنای باند موجود در مدار پایانی (که تقریباً 1.1 MHz است) به سه باند تقسیم شده بود: باند POTS (سرویس تلفن معمولی - Plain Old Telephone Service)، باند ارسال از کاربر (به ایستگاه پایانی)، و باند ارسال به کاربر (از ایستگاه پایانی). تکنیک تقسیم پهنای باند به فرکانسهای مختلف مالتی پلکس تقسیم فرکانس (frequency division multiplexing) نام دارد، که بعداً درباره آن مفصلاً صحبت خواهیم کرد. شرکت های بعدی برای ارائه این سرویس از تکنیک متفاوتی استفاده کردند، و از آنجائیکه احتمالاً این روش غالب خواهد شد، ابتدا آنرا توضیح می دهیم.

این روش که DMT (تون چندگانه گسسته - Discrete MultiTone) نامیده می شود، در شکل ۲-۲۸ نشان داده شده است. در حقیقت، کاری که در اینجا انجام شده تقسیم پهنای باند موجود (1.1 MHz) به 256 کانال مستقل 4312.5 Hz است. از کانال 0 بعنوان POTS (سرویس تلفن معمولی) استفاده می شود. کانالهای 1-5 خالی رها شده اند، تا تداخلی بین صدا و داده پیش نیاید. از 250 کانال باقیمانده، یکی برای کنترل ارسال از کاربر و یکی برای کنترل ارسال به کاربر تخصیص یافته، و از بقیه کانالها می توان برای داده استفاده کرد.

در تئوری می توان از هر یک از این کانالها برای ارتباط دو-طرفه همزمان استفاده کرد، ولی هارمونیکها، همشنوایی و اثرات دیگر باعث می شود تا در عمل ظرفیت سیستم بسیار کمتر باشد. این ارائه دهنده سرویس است که تعیین می کند چند کانال برای ارسال از کاربر اختصاص یافته، و چند کانال برای ارسال به کاربر. از نظر تکنیکی می توان این نسبت را بصورت 50-50 تعریف کرد، ولی از آنجائیکه اغلب کاربران اطلاعات دریافتی خیلی بیشتری دارند، ۸۰٪ الی ۹۰ درصد پهنای باند به دریافت کاربر اختصاص داده می شود. از اینجا است که حرف "A" (بمعنای



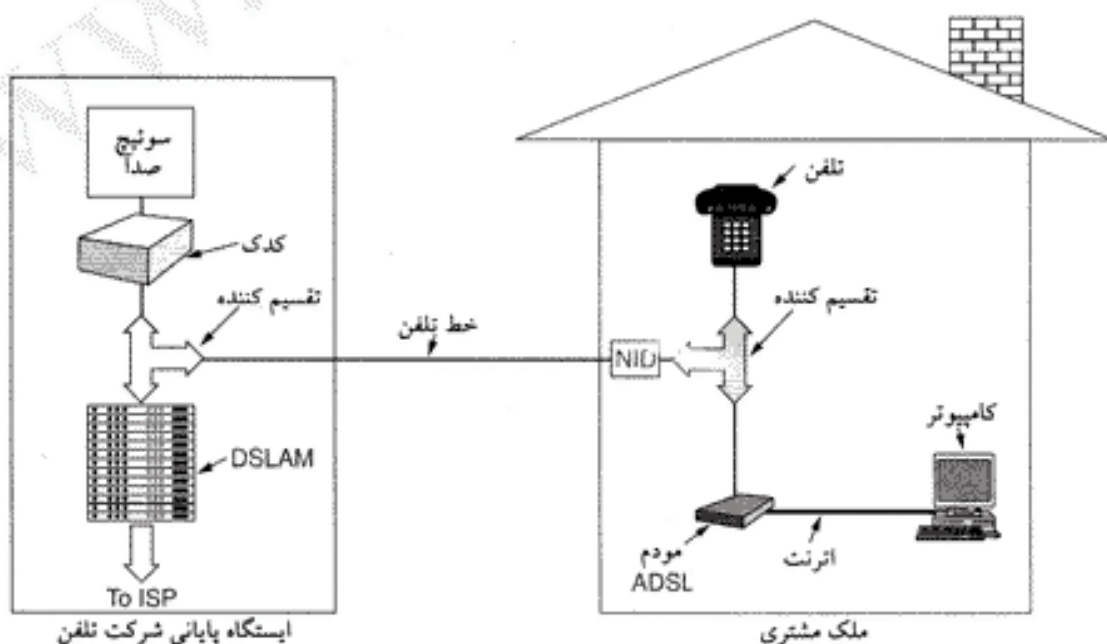
شکل ۲-۲۸. عملکرد ADSL با مدولاسیون تون چندگانه گسسته.

"نامتقارن" در ADSL ظاهر می شود. در اغلب موارد ۳۲ کانال به ارسال کاربر، و بقیه به دریافت آن اختصاص می یابد. به منظور افزایش پهنای باند، امکان اختصاص چند کانال فوقانی به ارتباط دوطرفه نیز وجود دارد، ولی این روش به تجهیزات اضافی برای خنثی کردن پژواک (echo) در خط نیاز دارد.

در استاندارد ADSL (ITU G.992.1 و ANSI T1.413) تا 8 Mbps برای دریافت کاربر و تا 1 Mbps برای ارسال کاربر مجاز است. اما، کمتر شرکتی پیدا می کنید که چنین سرویسهایی ارائه کند. در عمل، این سرویسهها معمولاً بصورت 512 kbps دریافت و 64 kbps ارسال (سرویس معمولی)، و 1 Mbps دریافت و 256 kbps ارسال (سرویس ویژه) ارائه می شوند.

در هر کانال از مدولاسیونی شبیه V.34 (با نرخ نمونه برداری 4000-baud بجای 2400-baud) استفاده می شود. کیفیت خط در هر کانال مستقلاً (و بصورت پیوسته) ارزیابی شده، و در صورت نیاز سرعت مجدداً تنظیم می شود، بنابراین سرعت کانالها می تواند کاملاً متفاوت باشد. برای ارسال داده ها از مدولاسیون QAM با نرخ 15 bits/ baud (شبیه شکل ۲-۲۵ ب) استفاده می شود. برای مثال، اگر 224 کانال دریافت با نرخ 15 bits/ baud در 4000-baud داشته باشیم، پهنای باند دریافتی معادل 13.44 Mbps خواهیم داشت. در عمل، نرخ سیگنال به نویز هرگز آنقدر خوب نیست که اجازه چنین سرعتهای را بدهد، ولی در مسافتهای کوتاه و با کابل کشی مناسب می توان به سرعت 8 Mbps رسید، که همین برای استقبال عامه کافیت.

در شکل ۲-۲۹ یک طرح ADSL را مشاهده می کنید. در این طرح، تکنسین شرکت تلفن بایستی یک (دستگاه واسط شبکه - Network Interface Device) در محل سکونت مشتری نصب کند. این جعبه پلاستیکی کوچک انتهای مایملک شرکت تلفن و شروع مایملک مشتری را مشخص می کند. کنار NID (یا حتی گاهی بصورت ترکیبی با آن) یک تقسیم کننده (splitter - فیلتر آنالوگی که باند POTS را از باند داده جدا می کند) نصب می شود. سیگنال POTS به دستگاه تلفن، و سیگنال داده به یک مودم ADSL داده می شود. مودم ADSL در واقع یک DSP (پردازشگر سیگنال دیجیتال - Digital Signal Processor) است، که بگونه ای تنظیم شده تا بعنوان 256 مودم QAM موازی (با فرکانسهای مختلف) کار کند. از آنجائیکه اکثر مودمهای ADSL بصورت خارج از



شکل ۲-۲۹. یک طرح ساده ADSL.

کامپیوتر هستند، ارتباط آنها با کامپیوتر نیز باید از نوع پرسرعت باشد. برای این منظور از پورتهای اترنت یا USB استفاده می شود. بدون شک در آینده شاهد به بازار آمدن مودمهای داخلی ADSL نیز خواهیم بود. در انتهای دیگر خط (در ایستگاه پایانی)، یک تقسیم کننده متناظر قرار می گیرد. در اینجا، سیگنال صدا (0-4000 Hz) از داده جدا شده و به سونچهای معمولی تلفن فرستاده می شود. سیگنال بالای 26 kHz نیز جدا شده، و به دستگاهی بنام DSLAM (مالتی پلکسر دسترسی DSL - DSL Access Multiplexer) که بسیار شبیه مودم ADSL است، می رود. پس از تبدیل این سیگنال به جریان بیهیجیتال، از آنجا به ISP فرستاده می شود. جداسازی کامل سیستم صدا از ADSL، پیاده سازی آنرا برای شرکت های تلفن بسیار ساده کرده است: تمام کاری که باید انجام دهد اینست که در سمت خود یک تقسیم کننده و DSLAM، و در سمت مشتری یک تقسیم کننده ساده نصب کند. در سیستمهای پرسرعت دیگر (مانند، ISDN) تجهیزات بسیار پیچیده تری باید نصب شود.

یکی از معایب طرح شکل ۲-۲۹ وجود تجهیزاتی است که باید در محل سکونت مشترک نصب شود (NID و تقسیم کننده). این کار مستلزم آنست که یک تکنسین به آنجا مراجعه کرده، و وسایل را نصب کند. به همین دلیل استاندارد جدیدی که به تقسیم کننده نیازی ندارد، توسعه داده شده است. نام غیررسمی این استاندارد G.lite است، ولی رسماً به آن ITU G.992.2 گفته می شود. این طرح شبیه شکل ۲-۲۹ است، که فقط تقسیم کننده حذف شده، و از خط تلفن بهمان صورت موجود استفاده می شود. تنها تفاوت اینست که باید میکروفیلترهایی را بین پریز تلفن و تجهیزات (دستگاه تلفن و مودم ADSL) قرار داد. میکروفیلتر تلفن یک فیلتر پائین گذر (low-pass) است که فرکانسهای بالای 3400 Hz را حذف می کند؛ میکروفیلتر مودم ADSL یک فیلتر بالاگذر (high-pass) است که فرکانسهای زیر 26 kHz را حذف می کند. با این حال کارایی G.lite مانند زمانی که تقسیم کننده بکار می رود، نیست و تنها می تواند به سرعت 1.5 MHz برسد (ولی هزاران مراجعه مستقیم به منازل مشترکان را حذف می کند). در استاندارد G.lite به تقسیم کننده ایستگاه پایانی همچنان نیاز هست.

باید بیاد داشت که، ADSL یک استاندارد لایه فیزیکی است، و آنچه که روی آن اجرا می شود به کاربر تلفن بستگی دارد. یکی از این کاربردها ATM است، چون در ATM امکان کنترل کیفیت وجود دارد و بسیاری از شرکت های تلفن زیرساخت های آنرا در اختیار دارند.

مدار پایانی بیسیم

از سال ۱۹۹۶ در ایالات متحده آمریکا (و کمی بعد در کشورهای دیگر)، شرکت هایی که مایل بودند وارد رقابت با انحصارگر تلفن شهری (که به ILEC شهرت داشت) شوند، اجازه آنرا یافتند. شرکت های تلفن راه دور (IXC ها) اولین کسانی بودند، که وارد گود شدند. هر IXC که می خواست وارد بازار تلفن شهری شود، باید کارهای ذیل را انجام می داد. اول، زمین یا ساختمانی برای تأسیس ایستگاه پایانی (end office) بخرد یا اجاره کند. دوم، تجهیزات و سونچهای لازم را خریداری و در ایستگاه پایانی نصب کند. سوم، یک (یا چند) رشته فیبر نوری بین این ایستگاه پایانی و نزدیکترین ایستگاه بین شهری (tool office) بکشد، تا مشترکان آن بتوانند به شبکه تلفن کشوری دسترسی داشته باشند. چهارم، در صدد جلب مشتری برآید (با تبلیغ و ارائه سرویسهای بهتر و قیمت کمتر). و قسمت سخت کار همین جاست.

فرض کنید یک مشتری پیدا شده و می خواهد از شرکت تلفن شهری جدید (که آنرا CLEC می نامیم) سرویس تلفن بخرد. این شرکت چگونه باید مشتری را به ایستگاه پایانی خود - که خیلی هم برای آن خرج کرده - وصل کند؟ خریدن زمین در تمام طول مسیر، کندن کانال، و کشیدن یک خط تلفن بسیار پرهزینه است. بسیاری از این CLEC ها راه ساده تر و کم هزینه تری پیدا کرده اند: WLL (مدار پایانی بیسیم - Wireless Local Loop).

از یک نظر تلفنهای ثابت با مدار پایانی بیسیم شبیه تلفنهای همراه هستند، ولی سه تفاوت فنی مهم و اساسی با آنها دارند: اول، مشترکان WLL دسترسی اینترنت پرسرعت (با سرعتی حداقل معادل ADSL) می خواهند. دوم، این قبیل مشترکان انتظار ندارند (و اغلب اجازه نمی دهند) یک آنتن پشقای بزرگ بالای بام خانه هایشان نصب شود. سوم، جای این تلفنها اساساً ثابت است، و مشکلات جابجا شدن سیگنال را (که در تلفنهای همراه وجود دارد، و بعداً در همین فصل با آنها آشنا خواهید دید) ندارند. بدین ترتیب یک صنعت کاملاً جدید متولد می شود: بیسیم ثابت (سرویس تلفن معمولی و اینترنت روی مدار پایانی بیسیم).

با اینکه آغاز بکار جدی WLL از سال ۱۹۹۸ است، ولی منشأ آن به سال ۱۹۶۹ برمی گردد. در این سال، FCC دو کانال تلویزیونی (هر یک با پهنای باند 6 MHz در فرکانس 2.1 GHz) به مصارف آموزشی اختصاص داد. در سالهای بعد تعداد این کانالها به ۳۳ (و پهنای باند مجموع آنها به 198 MHz در 2.5 GHz) افزایش یافت.

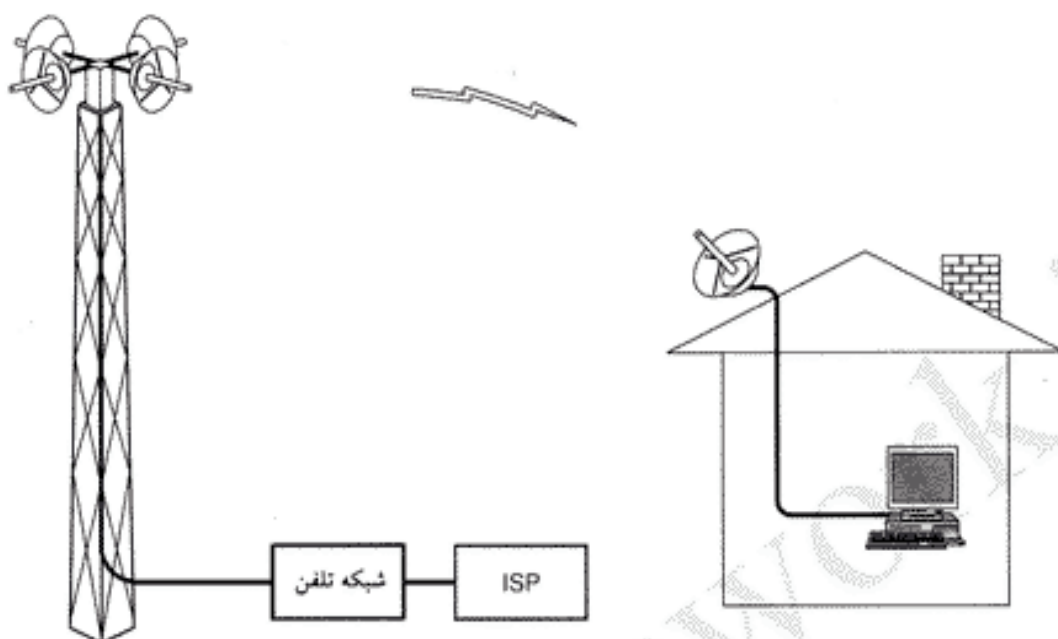
تلویزیون آموزشی هرگز پا نگرفت و در سال ۱۹۹۸، FCC این کانالها را پس گرفت و به ارتباطات رادیویی دوطرفه اختصاص داد - که بلافاصله در مدارهای پایانی بیسیم بکار گرفته شد. در این فرکانسها، طول موج امواج 10-12 cm است. بُرد این امواج حدود ۵۰ کیلومتر است، و از گیاهان و قطرات باران نسبتاً خوب عبور می کنند. سرویس جدیدی، که MMDS (سرویس توزیع چندکاناله چندنقطه ای - Multichannel Multipoint Distribution Service) نام گرفت، تمامی پهنای باند 198 MHz را مصرف کرد؛ MMDS را می توان (مانند پسر عمویش LMDS - که در همین قسمت آنرا توضیح می دهیم) یک شبکه شهری (MAN) در نظر گرفت.

مزیت بزرگ این سرویس تکنولوژی جا افتاده و موجود بودن تجهیزات آن است؛ و بزرگترین عیب آن این است که پهنای باند متوسطی دارد، که آن هم باید بین افراد زیادی در یک محدوده جغرافیایی وسیع به اشتراک گذاشته شود.

پهنای باند پائین MMDS باعث جذب شرکتهای تلفن به امواج میلیمتری شد. در فرکانسهای 28-31 GHz در آمریکا (و 40 GHz در اروپا) تمام باندها آزاد هستند، چون تکنولوژی تجهیزات نیمه هادی که در این فرکانسها کار کنند، پیچیده و گرانقیمت است. اما این مشکل هم با اختراع مدارات مجتمع گالیوم-آرسناید (Ga-As IC) حل شده، و راه برای ارتباطات رادیویی در باندهای میلیمتری هموار شده است. در پاسخ به تقاضاهای موجود، FCC یک پهنای باند 1.3 GHz به سرویس بیسیم جدید بنام LMDS (سرویس توزیع چندنقطه ای محلی - Local Multipoint Distribution Service) اختصاص داد - که این بزرگترین پهنای باندی بود که FCC تا به آن روز به یک کاربرد خاص اختصاص می داد. پهنای باند مشابهی نیز در اروپا (در فرکانس 40 GHz) به این سرویس اختصاص داده شده است.

عملکرد LMDS را در شکل ۲-۳۰ ملاحظه می کنید. در این شکل یک برج مخابراتی می بینید، که تعدادی آنتن در جهت های مختلف روی آن نصب شده است. از آنجائیکه امواج میلیمتری شدت خطی هستند، هر آنتن زاویه محدودی (که به قطاع معروف است) را پوشش می دهد، که مستقل از سایر آنتنهاست. در این فرکانس، بُرد امواج 2-5 km است، و بهمین دلیل برای پوشش کامل یک شهر به تعداد زیادی از این برجهای مخابراتی نیاز هست.

در سرویس LMDS نیز، مانند ADSL، تخصیص پهنای باند نامتقارن (و به نفع کانال دریافت) است. با تکنولوژی موجود، هر قطاع می تواند تا 36 Mbps روی کانال دریافت کاربر و تا 1 Mbps روی کانال ارسال کاربر ظرفیت داشته باشد، که بین تمام کاربران موجود در آن قطاع به اشتراک گذاشته می شود. اگر هر کاربر در هر دقیقه سه صفحه 5-KB دریافت کند، بطور متوسط 2000 bps از پهنای باند را اشغال می کند، که بدین ترتیب همزمان حداکثر 18,000 کاربر می توانند در هر قطاع کار کنند. البته برای قابل قبول بودن سرویس، نباید در هر لحظه بیش از ۹۰۰۰ کاربر فعال داشته باشیم. با فرض داشتن چهار قطاع در یک برج (مانند شکل ۲-۳۰)، تعداد کاربرانی که در



شکل ۲-۳۰. معماری یک سیستم LMDS.

هر لحظه می توانند فعال باشند به ۳۶,۰۰۰ می رسد - و اگر فرض کنیم در ساعات اوج مصرف از هر سه کاربر یکی فعال است، تعداد کل مشترکانی که یک برج مخابراتی در شعاع ۵ کیلومتری می تواند پوشش دهد، به ۱۰۰,۰۰۰ بالغ می شود. (بر مبنای همین محاسبات بود که تعداد زیادی از CLEC ها به این نتیجه رسیدند که با یک سرمایه گذاری متوسط در برجهای مخابراتی امواج میلیمتری، می توانند در بازار تلفن شهری و اینترنت با شرکت های تلویزیون کابلی رقابت کنند.)

اما، LMDS هم مشکلات خاص خود را دارد. اول اینکه، امواج میلیمتری کاملاً خطی هستند، و هیچ مانعی در خط دید برج مخابراتی و آنتن گیرنده نپایستی وجود داشته باشد. دیگر اینکه، برگ درختان جاذب خوبی برای این امواج است، بنابراین برج فرستنده بایستی آنقدر مرتفع باشد که درختان مانع امواج آن نشوند (و باید توجه داشت مسیری که در زمستان صاف است، ممکنست در تابستان که درختان پر از برگ هستند، چنین نباشد). باران نیز امواج میلیمتری را جذب می کند. این مشکل را می توان تا حدی با گداهای تصحیح خطا، و افزایش توان تشعشعی فرستنده در روزهای بارانی جبران کرد. معهذاً، سرویس LMDS در مناطق خشک شانس بیشتری برای موفقیت دارد، تا مناطق مرطوب و پُر باران.

مدار پایانی بیسیم بدون وجود استانداردهای معتبر شانس برای موفقیت ندارد، چون فقط در صورت وجود این استانداردهاست که سازندگان تجهیزات الکترونیکی رغبت ساخت وسایل مورد نیاز این صنعت را پیدا می کنند، و مشترکان نیز می توانند با خیال راحت (بدون نگرانی از اینکه با عوض کردن شرکت تلفن، وسایل را هم باید تعویض کنند) مشترک این سرویسها شوند. بمنظور تدوین استانداردهای LMDS، کمیته 802.16 توسط IEEE تأسیس شد، که این استاندارد را (که شبکه شهری بیسیم نامیده می شود) در آوریل ۲۰۰۲ عرضه کرد.

استاندارد IEEE 802.16 تلفن دیجیتال، دسترسی اینترنت، ارتباط بین شبکه های محلی، ایستگاههای پخش برنامه های رادیویی و تلویزیونی (و چندین کاربرد دیگر) را در بر می گیرد. در فصل ۴ درباره این استاندارد بیشتر صحبت خواهیم کرد.

۴-۵-۲ ترانک ها و مالتی پلکس کردن

ساخت و ساز نقش مهمی در اقتصاد سیستمهای تلفن بازی می کند. نصب یک خط اصلی با پهنای باند زیاد (high-bandwidth trunk) بین دو مرکز سوییچینگ به همان اندازه یک خط اصلی با پهنای باند کم (low-bandwidth trunk) پول لازم دارد، چون هزینه اصلی در اینجا هزینه حفر کانالهاست نه هزینه کابل مسی یا فیبر نوری. بهمین دلیل، شرکت های تلفن از روشهای پیچیده ای برای ارسال همزمان چندین مکالمه روی یک خط فیزیکی استفاده می کنند، که به این روشها مالتی پلکس (multiplex) گفته می شود. تکنیکهای مالتی پلکس کردن به دسته بزرگ تقسیم می شوند: FDM (مالتی پلکس تقسیم فرکانس - Frequency Division Multiplexing) و TDM (مالتی پلکس تقسیم زمان - Time Division Multiplexing). در FDM، طیف فرکانسی به باندهای مختلفی تقسیم می شود، و هر کاربر انحصاراً از یک باند استفاده می کند. در TDM، هر کاربر برای لحظه ای کوتاه (برش کوچکی از زمان) کل پهنای باند را در اختیار می گیرد.

در رادیوهای AM هر دو نوع مالتی پلکس دیده می شود. طیف تخصیص داده شده به هر کانال (ایستگاه) در حدود 1 MHz (تقریباً بین 500-1500 kHz) است، و هر ایستگاه در باند اختصاصی خود کار می کند. بین ایستگاهها نیز آنقدر فاصله وجود دارد، که با هم تداخل نکنند. این سیستم نمونه ای از FDM است. علاوه بر آن، در برخی از کشورها هر ایستگاه دارای دو زیرکانال منطقی است: کانال موزیک و کانال آگهی. فرستنده در برشهای کوتاه زمانی (و بصورت یک در میان) موزیک و آگهی پخش می کند، که در این حالت بصورت TDM کار می کند. در ادامه، ابتدا مالتی پلکس تقسیم فرکانس (و کاربرد آن در فیبرهای نوری، که به مالتی پلکس تقسیم طول موج معروفست) را خواهیم دید. سپس با مالتی پلکس تقسیم زمان (و یکی از کاربردهای پیشرفته آن در فیبرهای نوری، بنام SONET) آشنا می شوید.

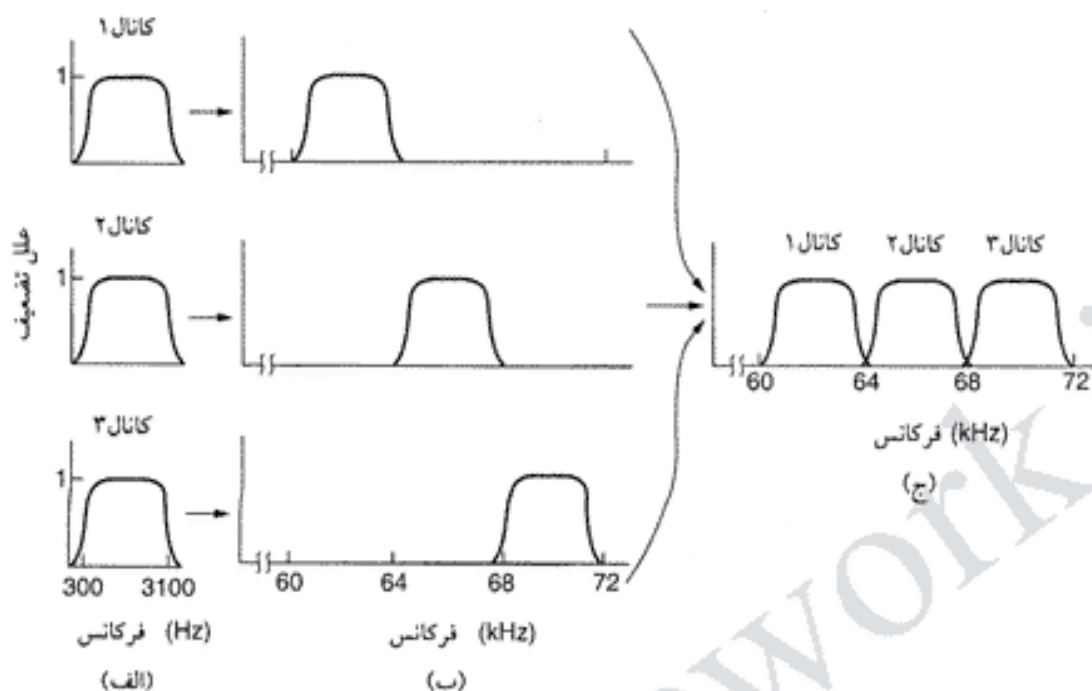
مالتی پلکس تقسیم فرکانس

در شکل ۲-۳۱ مالتی پلکس ۳ کانال صوتی با استفاده از FDM نشان داده شده است. پهنای باند هر کانال با استفاده از فیلترهای مخصوص به 3100 Hz محدود شده، ولی هنگام مالتی پلکس کردن پهنای هر کانال 4000 Hz در نظر گرفته می شود تا بین آنها تداخل پیش نیاید. در اولین قدم، فرکانس هر کانال به مقدار مشخصی (که با سایر کانالها متفاوت است) بالا برده می شود. سپس می توان این کانالها را با هم ترکیب کرد، چون دیگر هیچکدام از آنها فرکانس یکسانی ندارند و با هم مخلوط نمی شوند. دقت کنید که با وجود در نظر گرفتن یک حاشیه امنیتی برای هر کانال، آنها تا حدی روی هم می افتند، چون نقطه قطع فیلترها کاملاً تیز نیست. این روی هم افتادگی باعث بروز نوعی نویز غیرحرارتی در کانالهای مجاور می شود.

روشهای FDM که در سرتاسر دنیا بکار برده می شوند، تا حدی استاندارد شده اند. یکی از این استانداردها مالتی پلکس کردن دوازده کانال صوتی 4000-Hz روی باند 60-108 kHz است. این واحد گروه (group) نامیده می شود. گاهی اوقات از باند 12-60 kHz نیز بعنوان یک گروه استفاده می شود. بسیاری از شرکت های تلفن سرویسهای خطوط اجاره ای 48 kbps تا 56 kbps را بصورت همین گروه ها به مشتریان خود ارائه می کنند. از مالتی پلکس پنج گروه (یعنی ۶۰ کانال صوتی) یک فوق گروه (supergroup) بوجود می آید؛ و از مالتی پلکس پنج (در استاندارد CCITT) یا ده (در استاندارد شرکت پل) فوق گروه یک ابرگروه (mastergroup) شکل می گیرد. در برخی از استانداردها حتی تا ۲۳۰,۰۰۰ کانال صوتی در یک باند مالتی پلکس می شود.

مالتی پلکس تقسیم طول موج

برای کانالهای فیبر نوری از نوع دیگری از مالتی پلکس تقسیم فرکانس استفاده می شود، که مالتی پلکس تقسیم



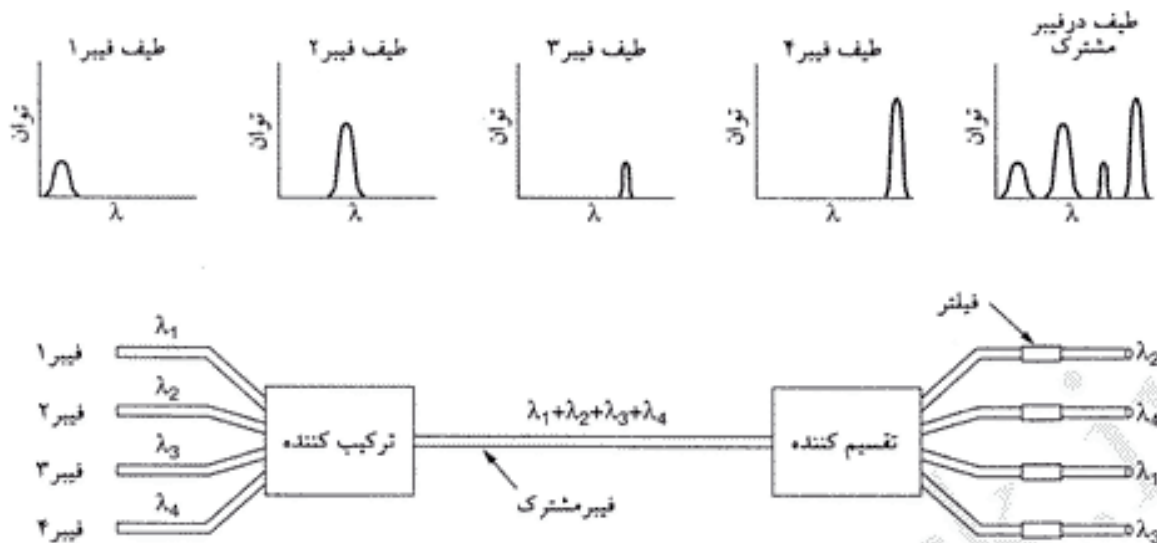
شکل ۳۱-۲. مالتی پلکس تقسیم فرکانس. (الف) کانالهای اولیه. (ب) بالا بردن فرکانس کانالها. (ج) کانالهای مالتی پلکس شده.

طول موج (Wavelength Division Multiplexing) نام دارد. اصول WDM در شکل ۳۲-۲ نشان داده شده است. در اینجا پرتوهای چهار فیبر نوری، که هر کدام طول موج متفاوتی دارند، وارد یک ترکیب کننده نوری شده، و برای ارسال آماده می شوند. در انتهای دیگر نیز عمل عکس انجام شده، و پرتوها مجدداً تفکیک می شوند. این کار توسط فیلترهای مخصوصی که فقط به یک طول موج اجازه عبور می دهند، انجام می شود.

در واقع هیچ چیز جدیدی در این تکنیک وجود ندارد، و این یک مالتی پلکس تقسیم فرکانس در فرکانسهای بسیار بالاست. مادامیکه هر کانال دارای فرکانس (یا، طول موج) خاص خود باشد، می توان آنها را با هم ترکیب (مالتی پلکس) کرد. تنها فرق این روش با FDM الکتریکی اینست که در سیستمهای نوری از فیلترهای انکساری منفعل استفاده می شود، که بسیار قابل اطمینان هستند.

سرعت پیشرفت تکنولوژی WDM آنقدر زیاد است که صنعت کامپیوتر به گرد آن هم نمی رسد. این تکنولوژی در سال ۱۹۹۰ اختراع شد، و اولین سیستم تجاری WDM دارای هشت کانال 2.5 Gbps بود. در سال ۱۹۹۸، سیستمهایی با ۴۰ کانال 2.5 Gbps به بازار عرضه شد، و در سال ۲۰۰۱ شاهد سیستمهایی با ۹۶ کانال 10 Gbps بودیم، که ظرفیت کل آنها به 960 Gbps می رسد (این پهنای باند برای ارسال ۳۰ فیلم کامل - با فرمت MPEG-2 - در هر ثانیه کافیت). در آزمایشگاهها سیستمهایی با ۲۰۰ کانال نیز تست شده اند. وقتی تعداد کانالها خیلی زیاد باشد، و طول موجها فاصله کمی (در حد 0.1 nm) با هم داشته باشند، به آن DWDM (چگال - Dense WDM) گفته می گویند.

علت محبوبیت سیستمهای WDM اینست که انرژی هر فیبر نوری فقط چند گیگاهرتز پهنای دارد. چون در حال حاضر امکان تبدیل سریعتر سیگنالهای الکتریکی به پالسهای نوری (و بالعکس) وجود ندارد. با ترکیب چند کانال با طول موجهای مختلف، پهنای باند بصورت خطی افزایش پیدا می کند. از آنجائیکه پهنای باند یک فیبر نوری در حدود 25,000 GHz است (شکل ۲-۶ را ببینید)، ظرفیت آن می تواند به ۲۵۰۰ کانال 10-GHz برسد - و این



شکل ۲-۳۲. مالتی پلکس تقسیم طول موج.

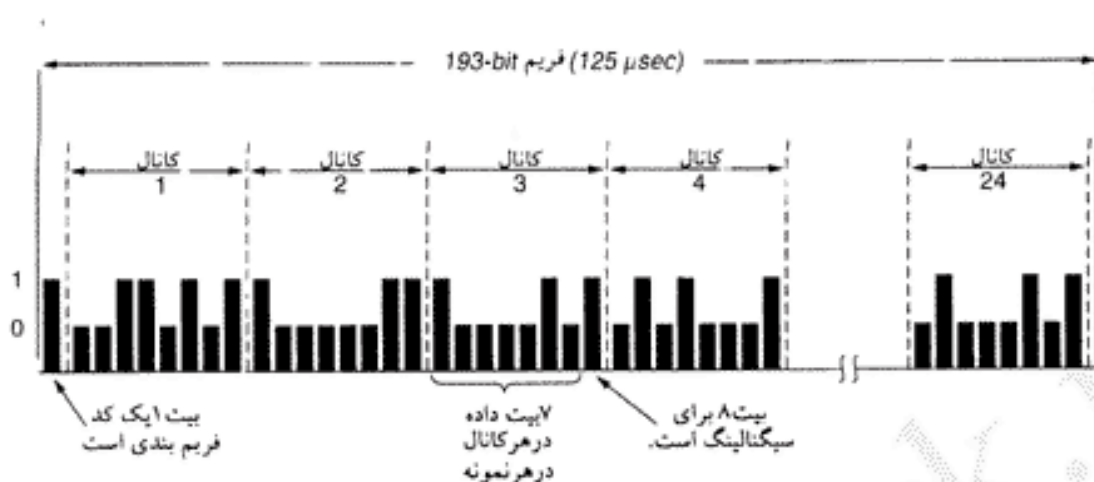
در 1 bit/Hz است، که البته نرخهای بالاتر هم امکان دارد. پیشرفت جدیدی که در این زمینه بدست آمده، تقویت کننده های تمام نوری است. در گذشته، لازم بود تا در فواصل ۱۰۰ کیلومتری پالسهای نوری به سیگنالهای الکتریکی تبدیل شده، و پس از تقویت دوباره به پالسهای نوری تبدیل و ارسال شوند. امروزه، تقویت کننده های تمام نوری می توانند (بدون نیاز به تبدیل کننده های نوری-الکتریکی) عمل تقویت را در هر ۱۰۰۰ کیلومتر انجام دهند.

مثال شکل ۲-۳۲ یک سیستم ثابت است: بیتهای فیبر ۱ به خروجی ۳ می روند، بیتهای فیبر ۲ به خروجی ۱ می روند، و الی آخر. ولی، ساخت سیستمهای سونچینگ WDM نیز امکانپذیر است. در این قبیل دستگاهها، می توان فیلترهای خروجی را با استفاده از تداخل سنجهای فابری-پروت یا ماخ-زندر تنظیم کرد. برای کسب اطلاعات بیشتر در زمینه WDM و کاربرد آن در سونچینگ بسته اینترنت، به (Elmirghani and Moustafa, 2000; Hunter and Andonovic, 2000; Listani et al., 2001) مراجعه کنید.

مالتی پلکس تقسیم زمان

تکنولوژی WDM بسیار جالب و مهیج است، ولی هنوز هزاران هزار کیلومتر سیم مسی در شبکه تلفن موجود است که باید فکری هم بحال آنها کرد. با اینکه FDM هنوز کاربرد گسترده ای در کابلهای مسی و کانالهای مایکروویو دارد، اما این تکنولوژی اساساً آنالوگ است و نمی توان آنرا با کامپیوتر انجام داد. ولی (Time Division Multiplexing - تقسیم زمانی) را می توان بطور کامل دیجیتالی کرد، و به همین دلیل در سالهای اخیر کاربرد گسترده ای یافته است. متأسفانه، از TDM فقط برای داده های دیجیتال می توان استفاده کرد، و از آنجائیکه مدارهای پایانی سیگنالهای آنالوگ تولید می کنند، باید این سیگنالها را در ایستگاه پایانی به دیجیتال تبدیل کرده، و سپس روی ترانکها (که دیجیتال هستند) ارسال کرد.

در این قسمت نحوه دیجیتالی کردن سیگنالهای آنالوگ صدا، و ترکیب آنها برای ارسال روی ترانکهای دیجیتال را توضیح می دهیم - سیگنالهایی که کامپیوترها از طریق مودم ارسال می کنند نیز آنالوگ است، بنابراین توضیحات زیر در مورد آنها هم صادق است. سیگنالهای آنالوگ در ایستگاه پایانی توسط دستگاهی بنام کدیک (codec) دیجیتالی شده، و یکسری اعداد ۸ بیتی تولید می شود. این کدک در هر ثانیه ۸۰۰۰ نمونه می گیرد (در هر



شکل ۲-۳۳. کاربرد T1 (1.544 Mbps).

۱۲۵ میکروثانیه یک نمونه). چون طبق قضیه نایکوئیست این مقدار برای گرفتن تمام اطلاعات کانالی با پهنای باند 4-kHz کافیست - با نرخ نمونه برداری پانزده اطلاعات از دست می رود، و با نرخ بالاتر اطلاعات بیشتری بدست نمی آید. این تکنیک که PCM (مدولاسیون کد پالس - Pulse Code Modulation) نامیده می شود، قلب سیستم های جدید تلفن است. در نتیجه، تمام فواصل زمانی در سیستم های تلفن مضارب از 125 μsec هستند. با ورود تکنولوژی مخابرات دیجیتال، CCITT نتوانست بر سر استاندارد بین المللی برای آن به توافق برسد، به همین دلیل سیستم های دیجیتال متعددی در سراسر دنیا مورد استفاده قرار گرفتند که عمدتاً با هم ناسازگار بودند. تکنیکی که در آمریکای شمالی و ژاپن از آن استفاده شد، کاربرد T1 است که آنرا در شکل ۲-۳۳ ملاحظه می کنید. (بخواهیم دقیقتر صحبت کرده باشیم، این فرمت DS1 و کاربرد آن T1 خوانده می شوند، ولی در اینجا قصد نداریم با آنچه در صنعت جا افتاده مخالفت کنیم) هر کاربرد T1 عبارتست از ۲۴ کانال صوتی که با یکدیگر مالتی پلکس شده اند. معمولاً، این سیگنال های آنالوگ در فواصل زمانی منظم و متوالی به یک کدک داده شده و دیجیتالیز می شوند (بجای آنکه از ۲۴ کدک استفاده کرده، و خروجی آنها را ترکیب کنیم). هر یک از این ۲۴ کانال بنوبت ۸ بیت به خروجی می دهند. با احتساب ۷ بیت داده و یک بیت کنترل، در هر کانال $7 \times 8000 = 56,000$ bps داده و $1 \times 8000 = 8000$ bps اطلاعات کنترلی خواهیم داشت.

هر فریم دارای $24 \times 8 = 192$ بیت است، که (با احتساب یک بیت اضافی برای فریم بندی) 193 بیت در هر 125 μsec خواهیم داشت، که بدین ترتیب نرخ داده 1.544 Mbps به می رسد. بیت ۱۹۳ ام برای سنکرون کردن فریم مورد استفاده قرار می گیرد. این بیت شکل 0101010101... بخود می گیرد، و گیرنده بکمک این بیت می تواند از سنکرون بودن اطلاعات اطمینان یابد. اگر گیرنده همزمانی خود را با فرستنده از دست دهد، می تواند با جستجوی این طرح بیت دوباره با فرستنده سنکرون شود. کاربران آنالوگ بکلی نمی توانند چنین طرحی از بیت ها تولید کنند، چون این طرح بیت معادل موج سینوسی 4000-Hz است که حذف خواهد شد. کاربران دیجیتال می توانند چنین طرحی را تولید کنند، ولی مشکل اینجاست که این طرح با لغزش فریم نیز می تواند ظاهر شود. برای بازیابی سریعتر سیستم در صورت بروز چنین لغزش هایی، هنگامی که از T1 فقط برای ارسال داده استفاده می شود، کانال ۲۴ ام به یک طرح خاص سنکرون کردن اختصاص می یابد (و فقط در ۲۳ کانال داده ارسال می شود).

وقتی بالاخره CCITT بر استانداردهای PCM بتوافق دست یافت، احساس کرد که ۸۰۰۰ بیت کنترلی

خیلی زیاد است. بنابراین استاندارد 1.544-Mbps بر اساس بسته های داده ۸ بیتی بنا شده نه ۷ بیتی؛ بعبارت دیگر، هر سیگنال آنالوگ بجای ۱۲۸ سطح به ۲۵۶ سطح مجزا دیجیتایز می شود. دو ویرایش (ناسازگار) از این استاندارد تهیه شد. در سیگنالینگ کانال مشترک (common-channel signaling) بیت اضافی (که در اینجا بجای ابتدا به انتهای فریم ۱۹۳ بیتی چسبانده می شود) در فریمهای فرد مقدار 10101010... بخود می گیرد، و در فریمهای زوج (تمام کانالها) حاوی اطلاعات سیگنالینگ است.

در ویرایش دیگر، سیگنالینگ وابسته به کانال (channel-associated signaling)، هر کانال دارای زیرکانال سیگنالینگ مخصوص بخود است. در هر زیرکانال یک بیت از هر هشت بیت داده در هر شش فریم به سیگنالینگ اختصاص داده می شود. بنابراین از هر شش فریم پنج تای آنها ۸ بیتی و یکی ۷ بیتی است. یکی دیگر از پیشنهادات CCITT، کاربرد PCM با ظرفیت 2.048 Mbps است که E1 نام دارد. این کاربرد دارای ۳۲ بسته ۸ بیتی در هر 125 μsec است، که ۳۰ تای آنها برای داده و دو تا برای سیگنالینگ بکار می روند. در هر گروه چهار فریمی ۶۴ بیت سیگنالینگ وجود دارد، که نیمی از آن به سیگنالینگ وابسته به کانال اختصاص داده شده، و نیمی دیگر برای سنکرون کردن فریمها کنار گذاشته شده است (کشورهای مختلف می توانند از این نیمه بدخواه استفاده کنند). خارج از آمریکای شمالی و ژاپن بجای T1 از کاربرد E1 استفاده می شود.

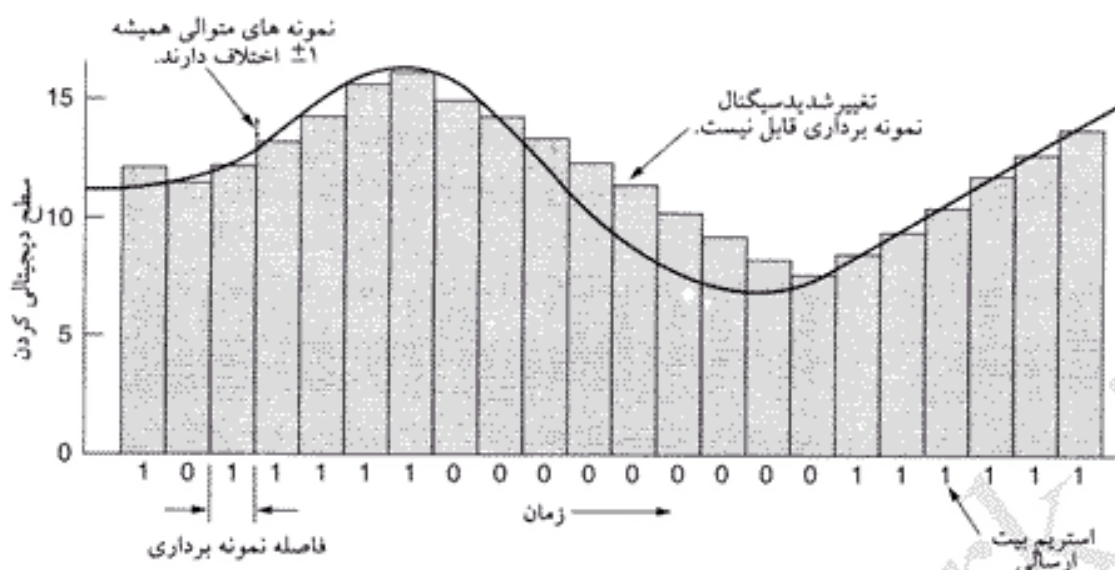
بعد از دیجیتایز کردن سیگنال صوتی، می توان با استفاده از تکنیکهای آماری تعداد پیتهای مورد نیاز هر کانال را کاهش داد (فشرده سازی). از این تکنیکها برای دیجیتایز کردن هر نوع سیگنال آنالوگ می توان استفاده کرد. تمام تکنیکهای فشرده سازی بر این اصل استوارند که تغییرات سیگنال نسبت به فرکانس نمونه برداری نسبتاً کمتر است، بنابراین قسمت اعظم سطوح دیجیتال ۷ یا ۸ بیتی تکراری و زائد هستند.

در یکی از این روشها، که مدولاسیون کُد پالس تفاضلی (differential pulse code modulation) نام دارد، نه خود دامنه دیجیتایز شده بلکه تفاوت آن با مقدار قبلی بکار برده می شود. از آنجائیکه در یک مقیاس ۱۲۸ تایی پرهشایی بیشتر از ± 16 ممکن نیست، بجای ۷ بیت به فقط ۵ بیت نیاز داریم. اگر سیگنال در مواردی خاص بیش از ۱۶ سطح اختلاف داشته باشد، برای نمونه برداری آن به بیش از یک دوره زمانی نیاز هست. خطایی که بدین ترتیب روی می دهد، در دیجیتایز کردن صدا قابل چشم پوشی است.

در نوع اصلاح شده این تکنیک فشرده سازی، هر نمونه باید با نمونه قبلی ± 1 اختلاف داشته باشد. در این شرایط بیتی که فرستاده می شود برای آنست که مشخص کند، نمونه جدید بیشتر از قبلی است یا کمتر از آن. این تکنیک را، که مدولاسیون دلتا (delta modulation) نامیده می شود، در شکل ۲-۳۴ ملاحظه می کنید. مانند سایر تکنیکهای فشرده سازی (که فرض را بر تغییرات اندک در سیگنال آنالوگ می گذارند) مدولاسیون دلتا هم در تغییرات شدید سیگنال دچار خطا می شود (که در چنین مواردی اطلاعات از دست می رود).

روش بهبود یافته PCM تفاضلی بر پیش بینی مقدار بعدی سیگنال با استفاده از برون یابی چند مقدار قبلی، و سپس محاسبه تفاضل این مقدار پیش بینی شده با مقدار واقعی سیگنال استوار است - به این روش کُد کردن پیشگویانه (predictive encoding) گفته می شود. البته در این روش فرستنده و گیرنده هر دو باید از روش پیش بینی یکسانی استفاده کنند. روش کُد کردن پیشگویانه باعث کم شدن اندازه اعدادی که باید کُد شوند، می شود و بهمین دلیل تعداد بیتهای ارسالی کاهش خواهد یافت.

تکنیک TDM اجازه می دهد تا چندین کاربرد T1 روی یک کاربرد مرتبه بالاتر مالتی پلکس شوند - در شکل ۲-۳۵ این تکنیک را ملاحظه می کنید. در اینجا چهار کانال T1 (سمت چپ) روی یک کانال T2 مالتی پلکس شده اند. در کانال T1 مالتی پلکس بصورت بایت به بایت انجام می شود، ولی از T2 به بعد مالتی پلکس بیت به بیت صورت می گیرد. چهار استریم T1 که هر کدام 1.544 Mbps هستند، بایستی خروجی 6.167 Mbps تولید کند،



شکل ۲-۳۴. مدولاسیون دلتا.

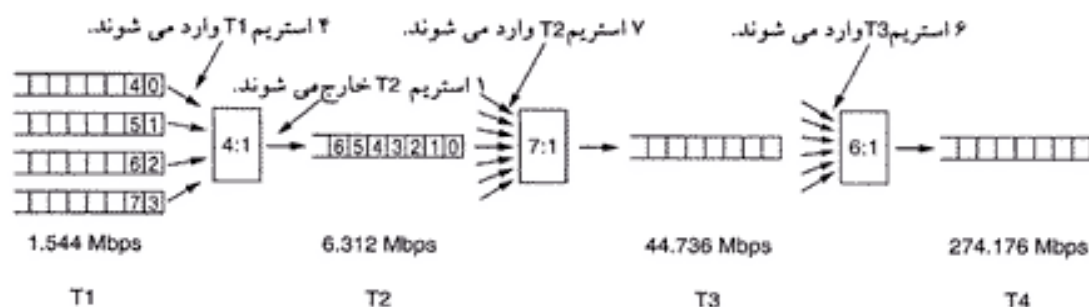
ولی ظرفیت T2 در واقع 6.312 Mbps است. بیهیای اضافی برای فریم بندی و بازیابی کانال در صورت لغزش کاربر منظور شده اند. کاربرهای T1 و T3 در میان مشترکین شناخته شده تر هستند، چون T2 و T4 بیشتر برای مصارف داخلی سیستم تلفن بکار می رود.

در مرحله بعد، هفت استریم T2 بصورت بیت به بیت روی یک استریم T3 مالتی پلکس می شوند؛ از ترکیب شش استریم T3 هم یک T4 بوجود می آید. در هر مرحله مقدار کمی سرآیند نیز برای فریم بندی و تصحیح خطای لغزش کاربر اضافه می شود.

همانطور که بر سر کاربر اصلی توافق کمی بین ایالات متحده و سایر کشورهای جهان وجود دارد، نحوه مالتی پلکس آنها نیز استاندارد دقیقی ندارد. کمتر کسی را در دنیا پیدا می کنید که با ضرایب پیشنهادی آمریکا (۴، ۷ و ۶) موافق باشد، به همین دلیل CCITT از مضرب یکنواخت ۴ در تمام مراحل استفاده می کند. همچنین نحوه فریم بندی و بازیابی کانال هم در ایالات متحده و CCITT متفاوت است. در استاندارد پیشنهادی CCITT تعداد کانالها به ترتیب ۳۲، ۱۲۸، ۵۱۲، ۲۰۴۸ و ۸۱۹۲ است، که با سرعتهای 2.048 Mbps، 8.848 Mbps، 34.304 Mbps، 139.264 Mbps و 565.148 Mbps کار می کنند.

SONET/SDH

در روزهای اولیه فیبر نوری، هر شرکت تلفن برای خود یک سیستم TDM نوری اختصاصی داشت. بعد از تجزیه



شکل ۲-۳۵. مالتی پلکس استریم T1 روی کاربرهای مرتبه بالاتر.

AT&T در سال ۱۹۸۴، شرکت های تلفن شهری مجبور شدند به کاربر های بین شهری مختلف که هر کدام سیستم TDM نوری خاص خود را داشت متصل شوند، به همین دلیل به فکر استاندارد کردن آن افتادند. در ۱۹۸۵، شرکت Bellcore (بازوی تحقیقاتی شرکت های RBOC) کار روی استاندارد ی بنام SONET (شبکه نوری سنکرون - Synchronous Optical Network) را شروع کرد. بعدها CCITT نیز وارد این معرکه شد، که نتیجه آن استاندارد های موازی SONET، G.707 و G.709 بود. توصیه های CCITT که SDH (سلسله مراتب دیجیتال سنکرون - Synchronous Digital Hierarchy) خوانده می شوند، فقط چند تفاوت جزئی با SONET دارند. امروزه تقریباً تمامی ترافیک تلفن راه دور در ایالات متحده و اکثر نقاط دنیا، از SONET در لایه فیزیکی ترانک استفاده می کنند. برای کسب اطلاعات بیشتر درباره SONET به (Bellamy, 2000; Goralski, 2000; Shepard, 2001) مراجعه کنید.

در طراحی SONET چهار هدف اصلی مد نظر بوده است. اول و از همه مهمتر، SONET بایستی کاری می کرد که کاربر های مختلف بتوانند با هم کار کنند. برای رسیدن به این هدف به یک استاندارد سیگنالینگ (در زمینه های طول موج، تایمینگ، ساختار فریم بندی و غیره) نیاز بود.

دوم، SONET باید راهی برای یکپارچه کردن سیستم های دیجیتال آمریکایی، اروپایی و ژاپنی (که همگی از کانال های 64-kbps PCM، ولی با روش های مختلف و ناسازگار، استفاده می کردند) پیدا می کرد.

سوم، SONET باید راهی برای مالتی پلکس کردن کانال های دیجیتال متعدد پیدا می کرد. در زمان تدوین SONET سریعترین کاربر دیجیتال، که کاربرد گسترده ای در ایالات متحده داشت، T3 با سرعت 44.736 Mbps بود. کاربر T4 تعریف شده، ولی هنوز بطور کامل عملیاتی نشده بود (برای بالاتر از آن هم هنوز کسی فکری نکرده بود). بخشی از مأموریت SONET ادامه راه تا gigabits/sec و بالاتر بود. همچنین باید راهی برای مالتی پلکس کردن کانال های کندتر در یک کانال SONET پیش بینی می شد.

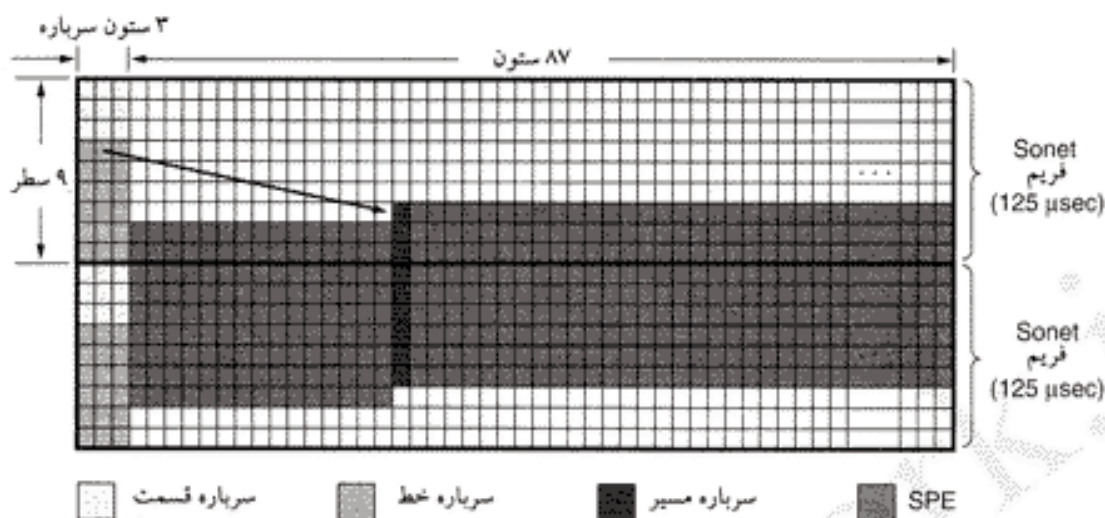
چهارم، SONET باید از یک سیستم جامع عملیات، مدیریت و نگهداری (OAM) پشتیبانی می کرد - وظیفه ای که در سیستم های قبلی توجه چندانی به آن نشده بود.

از همان ابتدا تصمیم بر آن شد تا SONET یک سیستم ساده TDM باشد. بگونه ای که تمام پهنای باند فیبر نوری یک کانال واحد را تشکیل دهد، و هر زیرکانال برای کسری از ثانیه کانال را در اختیار بگیرد. بدین ترتیب، SONET یک سیستم سنکرون خواهد بود، که توسط یک ساعت اصلی (با دقت 1 در 10^9) کنترل می شود. در یک خط SONET بیت ها با فواصل زمانی فوق العاده دقیق (که توسط ساعت اصلی کنترل می شود) ارسال می شوند. وقتی بعدها سوئیچینگ سلول بعنوان مبنای ATM انتخاب شد، برای تمایز آن با روش سنکرون SONET، از کلمه آسنکرون (Asynchronous Transfer Mode) استفاده شد. در SONET، فرستنده و گیرنده به یک ساعت مشترک وابسته اند، در حالیکه در ATM چنین نیست.

در SONET فریمها 810 بیتی هستند، که در فواصل زمانی 125 μ sec ارسال می شوند. از آنجائیکه SONET یک سیستم سنکرون است، خواه داده ای باشد یا نباشد، فریمها روی خط فرستاده می شوند. با این نرخ (8000 frames/sec)، SONET کاملاً با کانال های PCM سازگار خواهد بود.

فریمهای ۸۱۰ بیتی SONET را می توان جدولی با ۹۰ ستون و ۹ سطر فرض کرد. بنابراین در هر ثانیه ۸۰۰۰ فریم $810 \times 9 = 7290$ بیتی ارسال می شود، که سرعت کل آنرا به 51.84 Mbps می رساند. این کانال پایه SONET است، که STS-1 (سیگنال انتقال سنکرون 1 - Synchronous Transport Signal-1) نامیده می شود. تمام ترانک های SONET مضاربی از STS-1 هستند.

همانطور که در شکل ۲-۳۶ می بینید، سه ستون اول هر فریم به اطلاعات مدیریت سیستم اختصاص یافته اند.



شکل ۲-۳۶. دو فریم متوالی SONET.

در سه سطر اول این بایتها محتوی سرآیند قسمت (section overhead) هستند، و در شش سطر بعدی محتوی سرآیند خط (line overhead). سرآیند قسمت در ابتدا و انتهای هر قسمت ایجاد و چک می شود، و سرآیند خط در ابتدا و انتهای خط.

فرستنده SONET فریمهای ۸۱۰ بیتی را بصورت متوالی (back-to-back) ارسال می کند، حتی اگر هیچ داده ای در کار نباشد - که در این حالت اطلاعات ساختگی ارسال می شود. از دید گیرنده این یک استریم پیوسته از بیتهاست، پس چگونه می تواند تشخیص دهد ابتدای هر فریم کجاست؟ پاسخ اینست که دو بایت ابتدایی هر فریم طرح خاصی دارند، که گیرنده می تواند آنرا تشخیص دهد. بمحض یافتن این طرح، گیرنده با فرستنده سنکرون می شود. شاید بگوئید که احتمال دارد این طرح بیت ها در داده های کاربر نیز وجود داشته باشد، اما از آنجائیکه داده های چندین کاربر در یک فریم مالتی پلکس می شوند (و به دلایل دیگر) این اتفاق نخواهد افتاد.

ستونهای باقیمانده (۸۷ ستون) حاوی داده های کاربر هستند ($87 \times 9 \times 8 \times 8000 = 50.112 \text{ Mbps}$)، که به آنها SPE (بسته کاری سنکرون - Synchronous Payload Envelope) گفته می شود. اما داده های کاربر همیشه از سطر ۱ ستون ۴ شروع نمی شوند - SPE می تواند از هر کجای فریم شروع شود. اشاره گر نقطه شروع SPE در اولین سطر از سرآیند خط نوشته می شود. اولین ستون SPE سرآیند مسیر (path overhead) است - که سرآیند یست برای پروتکل های نقطه به نقطه.

با این تمهید (شروع SPE از هر نقطه فریم SONET، و امکان ادامه آن در فریمهای بعدی) انعطاف پذیری سیستم بسیار بالا می رود (شکل ۲-۳۶ را ببینید). برای مثال، اگر پس از شروع یک فریم SONET (و ارسال مقداری اطلاعات ساختگی) داده های واقعی از راه برسند، می توان بجای انتظار تا شروع فریم بعدی، آنها را از همان نقطه وارد فریم کرد.

سلسله مراتب مالتی پلکس SONET را در شکل ۲-۳۷ ملاحظه می کنید؛ نرخها از STS-1 تا STS-192 تعریف شده اند. هر $STS-n$ یک کاربر نوری متناظر بنام $OC-n$ دارد، که بیت به بیت با آن یکسان است (باستثنای یک جابجایی کوچک که برای سنکرون شدن لازم است). نامگذاری در استاندارد SDH متفاوت است و از $OC-3$ شروع می شود، چون در سیستمهای CCITT نزدیکی به 51.84 Mbps وجود ندارد. علت وجود کاربر $OC-9$ هم وجود یک خط اصلی پرسرعت در ژاپن است، که چنین سرعتی دارد. از کاربرهای $OC-9$ و $OC-18$

SONET		SDH	سرعت داده (Mbps)		
الکترونیکی	نوری	نوری	ناخالص	SPE	کاربر
STS-1	OC-1		51.84	50.112	49.536
STS-3	OC-3	STM-1	155.52	150.336	148.608
STS-9	OC-9	STM-3	466.56	451.008	445.824
STS-12	OC-12	STM-4	622.08	601.344	594.432
STS-18	OC-18	STM-6	933.12	902.016	891.648
STS-24	OC-24	STM-8	1244.16	1202.688	1188.864
STS-36	OC-36	STM-12	1866.24	1804.032	1783.296
STS-48	OC-48	STM-16	2488.32	2405.376	2377.728
STS-192	OC-192	STM-64	9953.28	9621.504	9510.912

شکل ۲-۳۷. نرخهای مالتی پلکس SONET و SDH.

در ژاپن استفاده می شود. نرخ داده ناخالص شامل تمام سرآیندها می شود؛ در نرخ داده SPE سرآیندهای قسمت و خط حذف شده اند؛ و در نرخ داده کاربر تمام سرآیندها حذف، و فقط ۸۶ ستون محاسبه شده است. اگر یک کاربر مالتی پلکس نشده و فقط شامل داده های یک منبع باشد، یک حرف c (بمعنای پیوسته) به انتهای نام آن اضافه می شود. برای مثال، OC-3 نشان دهنده یک کاربر 155.52-Mbps است که از مالتی پلکس شدن سه خط OC-1 حاصل شده، ولی OC-3c استریمی است از یک منبع واحد با نرخ 155.52 Mbps. سه استریم OC-1 در یک استریم OC-3c بصورت ستون-در-میان چیده می شوند: ستون ۱ از استریم ۱، سپس ستون ۱ از استریم ۲، سپس ستون ۱ از استریم ۳، بدینال آن ستون ۲ از استریم ۱، و الی آخر - که بدین ترتیب فریمی مرکب از ۲۷۰ ستون و ۹ سطر بوجود می آید.

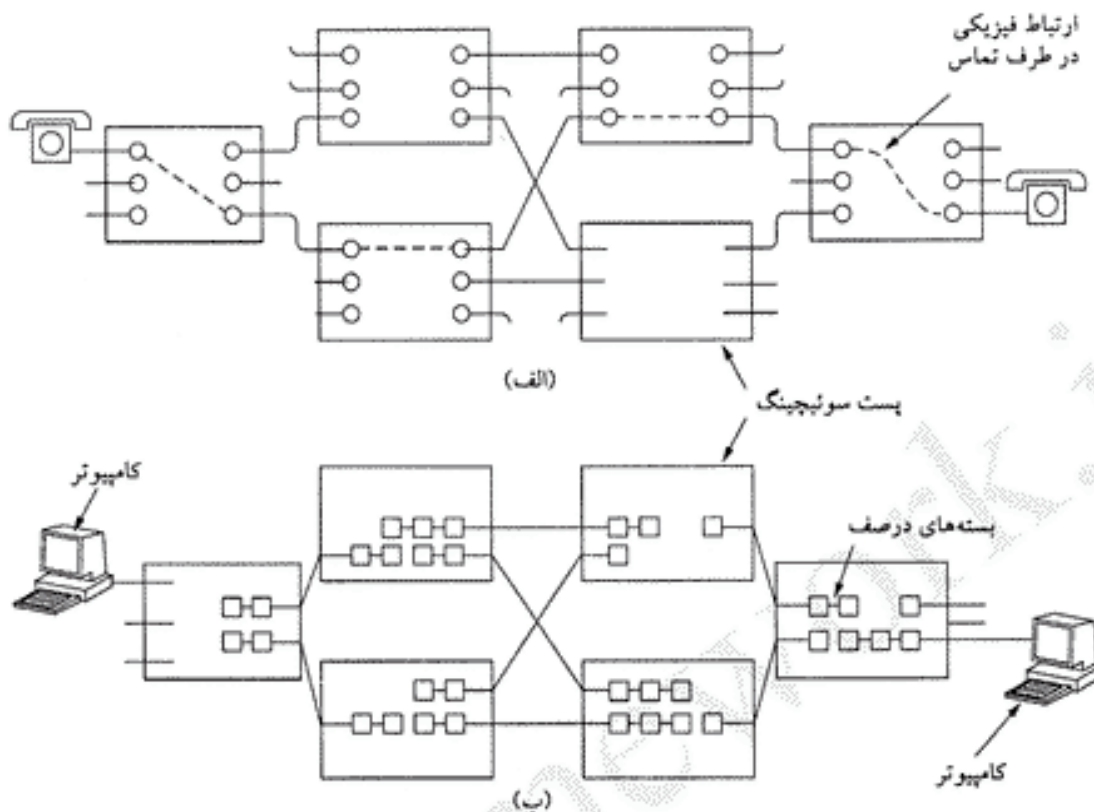
۵-۵-۲ سوئیچینگ

از دید اکثر مهندسان مخابرات، شبکه تلفن دو بخش عمده دارد: خارجی (مدارهای پایانی و ترانک ها - تجهیزاتی که در فضای باز و بیرون مرکز تلفن نصب می شوند) و داخلی (سوئیچها - تجهیزاتی که در فضای بسته و داخل مرکز تلفن نصب می شوند). تا اینجا با بخش خارجی آشنا شدیم؛ اکنون وقت آنست که نگاهی به داخل مرکز تلفن بیندازیم.

امروزه از دو تکنیک متفاوت سوئیچینگ استفاده می شود: سوئیچینگ مداری (circuit switching) و سوئیچینگ بسته ای (packet switching). ابتدا هر دو تکنیک را مختصراً معرفی می کنیم، و سپس مفصلاً به سوئیچینگ مداری (که تکنیک اصلی شبکه تلفن است) می پردازیم. در فصلهای آینده درباره سوئیچینگ بسته ای بیشتر صحبت خواهیم کرد.

سوئیچینگ مداری

وقتی یک تماس تلفنی می گیرید، دستگاههای سوئیچینگ سیستم تلفن در صدد یافتن یک مسیر فیزیکی بین شما و تلفن مقصد بر می آیند. به این تکنیک که آنرا در شکل ۲-۳۸ (الف) ملاحظه می کنید، سوئیچینگ مداری گفته می شود. هر یک از مستطیلهایی که در این شکل می بینید، یک مرکز سوئیچینگ (شهری، بین شهری، و غیره) است. در این مثال، هر مرکز سوئیچینگ سه خط ورودی و سه خط خروجی دارد. وقتی تماس تلفنی از یکی از این مراکز سوئیچینگ می گذرد، یک ارتباط فیزیکی بین آن خط ورودی و یکی از خطوط خروجی برقرار می شود، که در این



شکل ۲-۳۸. (الف) سوییچینگ مداری. (ب) سوییچینگ بندهای.

شکل با خط چین نشان داده شده است.

در روزهای اولیه تلفن، این ارتباط توسط اپراتور و یکمک یک سیم فنی که پرز ورودی را به خروجی متصل می کرد، انجام می شد. اختراع دستگاه سوییچینگ خودکار تلفن داستان جالبی دارد: این دستگاه در قرن نوزدهم در ایالت میسوری توسط فردی بنام آلن ب. استراوگر، که شغل وی کفن و دفن بود، اختراع شد. در آن روزها وقتی کسی می مرد، یکی از بازماندگان وی با اپراتور تلفن شهر تماس می گرفت و می گفت، «لطفاً مرا به یک مؤسسه کفن و دفن وصل کنید.» در شهر آقای استراوگر دو مؤسسه کفن و دفن وجود داشت، و از شانس بد این آقای اپراتور تلفن همسر رقیب بود. آقای استراوگر خیلی زود دریافت که اگر می خواهد ورشکست نشود، باید یک دستگاه سوییچینگ خودکار تلفن اختراع کند - و این کار را کرد. همه آنهايي که در سراسر دنیا با دستگاههای سوییچینگ خودکار تلفن سروکار دارند، آنها را با نام «دستگاه استراوگر» می شناسند. (تاریخ نمی گوید آیا این خانم بعد از بیکار شدن توانست شغلی مانند اپراتور اطلاعات تلفن - که باید به سؤالاتی از قبیل «لطفاً شماره یک مؤسسه کفن و دفن را بدهید،» پاسخ دهد - بدست آورد یا خیر؟)

البته شکل ۲-۳۸ (الف) بسیار ساده شده است، چون مسیر فیزیکی بین دو تلفن می تواند از لینکهای مایکروویو یا فیبر نوری (که هزاران تماس تلفنی روی آنها مالتی پلکس می شود) عبور کند. با این حال مفهوم کلی آن همچنان معتبر است: وقتی تماس تلفنی برقرار می شود، یک مسیر فیزیکی بین دو دستگاه تلفن بوجود می آید که تا آخر تماس باقی می ماند.

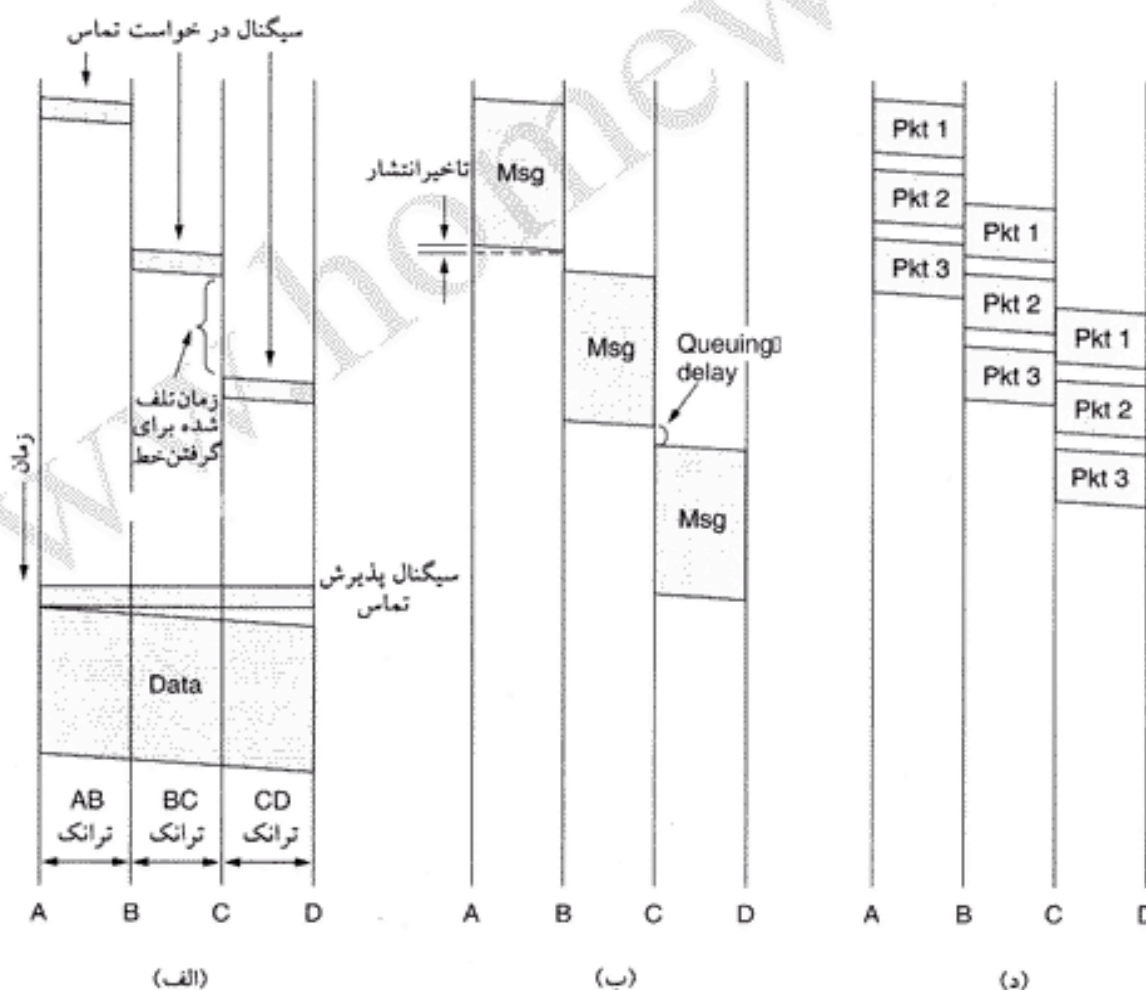
در شکل ۲-۳۸ (ب) روش جایگزین سوییچینگ مداری، که سوییچینگ بندهای نام دارد، را می بینید. در این تکنولوژی هر بنده مستقل فرستاده می شود، بدون آنکه از قبل مسیری به آن اختصاص داده شده باشد؛ این بر عهده هر بنده است که راه خود را به سمت مقصد پیدا کند.

مهمترین ویژگی سوئیچینگ بسته‌ای اینست که قبل از ارسال هر گونه داده‌ای، بایستی یک مسیر نقطه-به-نقطه بین مبدأ و مقصد برقرار شده باشد. فاصله زمانی بین شماره‌گیری در مبدأ و زنگ خوردن تلفن مقصد بر راحتی می‌تواند به ۱۰ ثانیه برسد (که این زمان در تماسهای راه‌دور و بین‌المللی حتی بیشتر است). در این مدت سیستم تلفن بدنبال یافتن یک مسیر مناسب است (شکل ۲-۳۸ الف) را ببینید). توجه داشته باشید که قبل از شروع ارسال داده، سیگنال درخواست (request) باید تمام مسیر را تا مقصد طی کرده، و تصدیق (acknowledgement) آن بازگردد. این تأخیر در بسیاری از کاربردهای کامپیوتری (مانند بررسی اعتبار مشتری در خریدهای اعتباری) قابل قبول نیست.

اما بمحض آنکه مدار برقرار شد، دیگر تأخیر چندانی بین گیرنده و فرستنده وجود ندارد (فقط زمان تأخیر انتشار امواج الکترومغناطیس، که آن هم چیزی در حدود $5 \mu\text{sec/km}$ است). دیگر اینکه (بدلیل وجود مدار اختصاصی)، در سوئیچینگ مداری پدیده‌ای بنام ازدحام (congestion) وجود ندارد (البته بدلیل اینکه ظرفیت خطوط و مراکز سوئیچینگ نامحدود نیست، قبل از برقراری مدار همیشه احتمال شنیدن بوق اشغال وجود دارد).

سوئیچینگ پیام

یکی دیگر از اشکال سوئیچینگ، که آنرا در شکل ۲-۳۹ (ب) ملاحظه می‌کنید، سوئیچینگ پیام



شکل ۲-۳۹. همزمانی رویدادها در (الف) سوئیچینگ مداری، (ب) سوئیچینگ پیام، (ج) سوئیچینگ بسته‌ای.

(message switching) است. در این نوع از سوییچینگ نیز مسیر فیزیکی ثابتی بین فرستنده و گیرنده وجود ندارد. وقتی فرستنده یک بلوک از داده ها را ارسال می کند، این داده ها در اولین مرکز سوییچینگ (که همان راتر است) ذخیره شده (store) و سپس به مرکز بعدی هدایت (forward) می شود. هر بلوک ابتدا بطور کامل دریافت شده، از نظر خطا بررسی و سپس ارسال می شود. همانطور که در فصل قبل هم گفتیم، به شبکه ای که با این روش کار می کند شبکه ذخیره-هدایت (store-and-forward) گفته می شود.

اولین تجهیزات مخابرات الکترومغناطیسی تجهیزات سوییچینگ پیام بودند، که برای ارسال تلگرام بکار گرفته شدند. پیام ابتدا در اداره تلگراف روی نوارهای کاغذی سوراخ می شد، و بعد از خوانده شدن توسط دستگاههای خاص به مرکز بعدی فرستاده می شد، که در آنجا بصورت یک نوار کاغذی سوراخ شده از دستگاه بیرون می آمد. اپراتور مسئول دستگاه کاغذ را پاره کرده، و در یک دستگاه نوارخوان (tape reader) می گذاشت تا بتواند پیام را بخواند (هر خط مخابراتی یک دستگاه نوارخوان داشت). این هم در واقع نوعی سوییچینگ است، که به سوییچینگ نوار پاره (torn tape switching) معروف بود. نوارهای کاغذی و سوییچینگ پیام مدتهاست که از دور خارج شده اند، و ما هم بیش از این درباره آنها صحبت نخواهیم کرد.

سوییچینگ بسته ای

اندازه پیام در شبکه سوییچینگ پیام هیچ محدودیتی ندارد، و این به آن معناست که دستگاههای مسیریاب برای نگهداری پیامها به وسایل ذخیره سازی (از قبیل، دیسک) نیاز دارند. پیامد دیگر این روش آنست که یک پیام واحد می تواند آنقدر بزرگ باشد که برای دقایقی خط مسیریاب-مسیریاب را اشغال کند، که بدین ترتیب کاربرد سوییچینگ پیام را در ارتباطات تعاملی (interactive) بشدت محدود می کند. برای حل این مشکل، سوییچینگ بسته ای (packet switching) اختراع شد. در شبکه های سوییچینگ بسته ای روی اندازه بسته ها محدودیت شدیدی اعمال می شود، و به همین دلیل مسیریاب ها نیازی به دیسک برای ذخیره کردن بسته ها ندارند، و می توانند آنها در در حافظه اصلی خود ذخیره کنند. با محدود کردن اندازه بسته ها، و اطمینان از اینکه یک کاربر نمی تواند خط انتقال را برای مدتی طولانی - که البته در اینجا منظور از طولانی بیش از چند میلی ثانیه است - به انحصار خود در آورد، شبکه های سوییچینگ بسته ای برای کاربردهای تعاملی بسیار مناسبند. با مقایسه شکل های ۲-۳۹ (ب) و (ج) یکی دیگر از مزایای روش سوییچینگ بسته ای بر سوییچینگ پیام را مشاهده می کنید: بسته اول یک پیام چندبسته ای می تواند حتی قبل از رسیدن بسته دوم به مسیریاب بعدی فرستاده شود، که این زمان تأخیر را کاهش داده و کارایی سیستم را بالا می برد. به دلایل فوق، شبکه های کامپیوتری اغلب از سوییچینگ بسته ای استفاده می کنند؛ سوییچینگ مداری نیز در موارد خاصی بکار برده می شود، ولی سوییچینگ پیام هیچ کاربردی در شبکه های کامپیوتری ندارد.

سوییچینگ مداری و سوییچینگ بسته ای تفاوتهای زیادی با یکدیگر دارند. برای مثال، در یک شبکه سوییچینگ مداری قبل از ارسال اطلاعات بایستی مدار فیزیکی بین فرستنده و گیرنده برقرار شده باشد، در حالیکه در شبکه های سوییچینگ بسته ای چنین الزامی وجود ندارد و ارسال بسته ها می تواند بلافاصله شروع شود. پیامد لزوم برقراری مدار ثابت در سوییچینگ مداری، اختصاص پهنای باند در تمام طول مسیر بین فرستنده و گیرنده است: تمام بسته ها باید از این مسیر عبور کنند. از طرف دیگر وقتی تمام بسته ها مجبور به عبور از یک مسیر باشند، نمی توانند خارج از ترتیبی که ارسال شده اند، به مقصد برسند. در سوییچینگ بسته ای هیچ مسیر ثابتی وجود ندارد، و بسته ها می توانند از هر مسیری که (در آن لحظه خاص) در شبکه موجود است عبور کنند، و حتی خارج از نظم و ترتیب اولیه به مقصد برسند.

ویژگی تحمل خطا در شبکه های سوئیچینگ بسته ای بسیار بهتر از شبکه های سوئیچینگ مداری است - و در واقع دلیل اختراع آن هم همین بوده است. وقتی در شبکه سوئیچینگ بسته ای یک مسیر یاب از کار می افتد، بسته ها می توانند از مسیرهای دیگری که وجود دارد، استفاده کرده و مسیر یاب مرده را دور بزنند.

البته وجود یک پهنای باند اختصاصی در شبکه های سوئیچینگ مداری این مزیت را دارد که بسته ها بمحض رسیدن به یک مسیر یاب به مسیر یاب بعدی فرستاده می شود، و زمان تأخیر ارسال بشدت کاهش می یابد؛ در حالیکه در شبکه های سوئیچینگ بسته ای چنین پهنای باندی اختصاصی وجود ندارد، و بسته ها باید تا رسیدن نوبت ارسال در صف منتظر بمانند.

وجود مدار اختصاصی در شبکه های سوئیچینگ مداری بدان معناست که (بعد از برقراری مدار) دیگر حالت ازدحام - انتظار برای باز شدن راه - بروز نخواهد کرد. البته همیشه این احتمال وجود دارد که در شروع ارتباط بدلیل شلوغی شبکه، امکان اختصاص مدار وجود نداشته باشد؛ این نوع دیگری از ازدحام - انتظار برای تخصیص مدار - است.

پهنای باندی که به یک کاربر تخصیص داده می شود، در تمام مدت در اختیار وی است، حتی اگر هیچ چیز برای ارسال نداشته باشد. امکان استفاده از این مدار برای کاربران دیگر وجود ندارد. در شبکه های سوئیچینگ بسته ای اتلاف پهنای باند به شکل فوق وجود ندارد، و بهمین دلیل کارایی کلی آن بسیار بهتر است. درک این تفاوت بین سوئیچینگ بسته ای و سوئیچینگ مداری بسیار مهم است: تفاوت تضمین سرویس به قیمت اتلاف منابع، با استفاده بهینه از منابع به قیمت عدم تضمین سرویس.

سوئیچینگ بسته ای از تکنیک ذخیره - هدایت استفاده می کند. در این روش هر بسته قبل از ارسال به مسیر یاب بعدی باید بطور کامل دریافت و در حافظه مسیر یاب ذخیره شود. این روش تأخیر نسبتاً قابل ملاحظه ای ایجاد می کند؛ در حالیکه در سوئیچینگ مداری، بیت ها بطور پیوسته روی مدار منقل می شوند و چنین تأخیری وجود ندارد.

تفاوت دیگر اینست که سوئیچینگ مداری بطور کامل شفاف است: فرستنده و گیرنده می توانند از هر نرخ بیت، فرمت، و یا روش فریم بندی که می خواهند استفاده کنند؛ کاربر در این مورد هیچ چیز نمی داند، و به آن اهمیتی هم نمی دهد. اما در سوئیچینگ بسته ای این کاربر است که پارامترهای اصلی را تعیین می کند. تفاوت این دو تقریباً مانند جاده و راه آهن است: در جاده این مسافر است که سرعت، اندازه و نوع وسیله نقلیه را انتخاب می کند، در حالیکه در راه آهن انتخاب این پارامترها بر عهده شرکت مسافری (کاربر) است. همین شفافیت است که به سیستم تلفن اجازه می دهد انواع اطلاعات (صوت، فکس و داده) را منتقل کند.

آخرین تفاوت سوئیچینگ مداری و سوئیچینگ بسته ای روش محاسبه هزینه است. در سوئیچینگ مداری (بدلیل تاریخی) هزینه بر اساس مسافت و مدت محاسبه می شود. در تلفنهای همراه، مسافت (البته باستثنای مکالمات بین المللی) نقشی ندارد، و مدت مکالمه نیز نقش ناچیزی دارد (برای مثال، تفاوت هزینه مکالمه در روز، شب یا ایام تعطیل). در سوئیچینگ بسته ای، اساساً چیزی بنام مدت مکالمه وجود ندارد، و فقط گاهی حجم ترافیک نقشی در هزینه ها بازی می کند. در مصارف خانگی، معمولاً هزینه ها بصورت ماهیانه ثابت اخذ می شود، چون این روش برای ISP ها ساده تر است و کاربران نیز راحت تر با آن کنار می آیند، ولی کاربرهای اصلی شبکه هزینه ها را بر اساس حجم ترافیک دریافت می کنند. تفاوتهایی را که در این قسمت برشمردیم، بصورت خلاصه در شکل ۲-۴۰ ملاحظه می کنید.

سوئیچینگ مداری و بسته آنقدر مهم هستند که بزودی دوباره به این مبحث برگشته، و تکنولوژیهای مختلف آنها را به تفصیل بررسی خواهیم کرد.

سوئیچینگ بسته ای	سوئیچینگ مداری	آیتم
لازم ندارد	لازم دارد	برقراری تماس
خیر	بلی	مسیر فیزیکی اختصاصی
خیر	بلی	تمام بسته ها از یک مسیر عبور می کنند
خیر	بلی	بسته ها به ترتیب دریافت می شوند
خیر	بلی	خرابی سوئیچ مرگ آور است
متغیر	ثابت	پهنای باند موجود
در هر بسته	در لحظه شروع	زمان ازحام احتمالی
خیر	بلی	پهنای باند تلف شده
بلی	خیر	ذخیره - هدایت
خیر	بلی	شفافیت
به ازای هر بسته	در دقیقه	هزینه

شکل ۲-۴۰. مقایسه شبکه های سوئیچینگ مداری و سوئیچینگ بسته ای.

۶-۲ شبکه تلفن همراه

سیستم تلفن معمولی (حتی اگر سرعت آن به دهها گیگاهابت برسد) دسته خاصی از کاربران را هرگز راضی نخواهد کرد: آنهایی که در یک جا بند نمی شوند. امروزه مردم می خواهند از هر جایی که هستند (داخل هواپیما، سوار بر اتومبیل، کنار دریا، نوک قله کوهها و یا از اعماق جنگل) تلفن بزنند - و حتماً تا چند سال دیگر انتظار دارند از این نقاط ایمیل خود را نیز چک کنند، و یا در وب گشت بزنند. پیامد این گرایش رشد چشمگیر تلفنهای غیر ثابت در سالهای اخیر است. در این قسمت قصد داریم مبحث تلفن همراه را بررسی کنیم.

تلفنهای غیر ثابت به دو دسته بزرگ تقسیم می شوند: گوشی بیسیم (cordless phone)، و تلفن همراه (mobile phone) - که گاهی به آن تلفن سلولی (cell phone) نیز گفته می شود. گوشی بیسیم عبارتست از یک دستگاه مرکزی و یک گوشی متحرک، که برای مصارف خانگی (مسافتهای کوتاه) طراحی شده است. این وسیله هرگز کاربردی در شبکه نداشته، و ما هم بیش از این به آن نخواهیم پرداخت. تلفن همراه، که امروزه کاربرد گسترده ای در ارتباطات صدا و داده دارد، موضوع اصلی بحث ماست.

تکامل تلفنهای همراه سه نسل را با تکنولوژیهای متفاوت پشت سر گذاشته است:

۱. صدای آنالوگ
۲. صدای دیجیتال
۳. صدای دیجیتال و داده (اینترنت، ایمیل، و غیره)

با اینکه بیشتر تکنولوژی این سیستمها مورد علاقه ماست، اما جالبست بدانید که سیاست و تصمیمات اقتصادی چه تأثیری بر تکامل این سیستمها دارد. اولین سیستم تلفن همراه در ایالات متحده آمریکا و توسط AT&T اختراع شد، که FCC هم بلافاصله آنرا در تمام کشور اجباری کرد. در نتیجه، تمام ایالات متحده صاحب یک سیستم واحد تلفن همراه (آنالوگ) شد، و تلفنی که در کالیفرنیا خریده شده بود، در نیویورک هم کار می کرد. اما در اروپا اوضاع کاملاً عکس این بود، و هر کشوری سیستم خاص خود را طرحی کرد، که نتیجه آن یک هرج و مرج کامل بود.

اروپا از اشتباه خود درس گرفت و وقتی نوبت تلفن همراه دیجیتال رسید، تمام شرکتهای مخابرات دولتی دور

هم جمع شده و بر سر یک استاندارد واحد (GSM) به توفیق رسیدند، که در نتیجه تلفنهای موبایل اروپایی در تمام نقاط این قاره کار می‌کند. در همان زمان ایالات متحده به این نتیجه رسیده بود که دولت نباید در موضوع استاندارد دخالت کند، و در نتیجه سرنوشت تلفن همراه دیجیتال به بازار سپرده شد. این تصمیم باعث شد تا انواع مختلفی از تلفن همراه دیجیتال تولید و وارد بازار شود - ایالات متحده اکنون دو سیستم بزرگ تلفن همراه دیجیتال (بعلاوه یک سیستم کوچکتر) دارد، که هیچکدام با دیگری سازگار نیست.

اروپا فقط در یک دوره زمانی کوتاه در زمینه تعداد کاربران تلفن همراه از آمریکا عقب بود، ولی اکنون از آن بسیار پیش افتاده است، که یکی از دلایل آن سیستم یکپارچه تلفن همراه در اروپاست، اما دلایل دیگری هم برای آن وجود دارد. تفاوت دیگر سیستم تلفن همراه آمریکا و اروپا (که باعث سرافکنندگی آمریکایی‌هاست) شماره‌های آنهاست. در ایالات متحده شماره‌های تلفن همراه با شماره تلفنهای معمولی (ثابت) مخلوط است. هیچ راهی برای تماس گیرنده وجود ندارد تا تشخیص دهد شماره‌ای که دارد می‌گیرد (مثلاً، 234-5678 (212))، یک شماره معمولی (کم‌هزینه) است یا یک شماره تلفن همراه (با هزینه زیاد). برای عصبی‌تر کردن مردم، شرکت‌های تلفن قرار گذاشته‌اند تا هزینه تماسها را به پای صاحب تلفن همراه بنویسند. بهمین دلیل اغلب مردم در خرید تلفن همراه تردید دارند، و از صورتحسابهای نجومی که ممکنست (در نتیجه تماس دیگران) برای آنها بیاید، می‌ترسند. در اروپا، شماره‌های تلفن همراه کد ناحیه مشخصی دارند (مانند تلفنهای 800 یا 900)، و بسادگی از تلفنهای معمولی قابل تشخیص هستند. هزینه تماس هم مثل تلفنهای معمولی به پای تماس گیرنده نوشته می‌شود (البته با استثنای تماسهای بین‌المللی، که هزینه تماس بین دو طرف تقسیم می‌شود).

دلیل دیگری که باعث پذیرش گسترده تلفن همراه در اروپا شده، ابداع تلفنهای همراه از پیش پرداخت شده (pre-paid) است، که تا ۷۵٪ تلفنهای این قاره را در برخی نقاط شامل می‌شود. این تلفنها را می‌توان در هر فروشگاهی (بدون هیچگونه تشریفات خاص) خرید - فقط پولش را بده و استفاده کن. این تلفنها معمولاً با اعتبار ۲۰ یا ۵۰ یورو (واحد پول اروپا) عرضه می‌شوند، و بعد از صفر شدن اعتبار می‌توان آنها را (با استفاده از یک PIN code سری) دوباره شارژ کرد. امروزه هر نوجوان (و حتی بچه‌ای) در اروپا یک چنین تلفن همراهی دارد، و والدین وی می‌توانند بسادگی محلی او را پیدا کنند (بدون اینکه از صورتحسابهای نجومی آن بترسند). این تلفنها (البته اگر از آنها برای تماس گرفتن استفاده نشود) ماهها دوام می‌آورند، چون نه هزینه ماهیانه دارند نه برای تلفنهایی که به آنها می‌شود، اعتباری کسر می‌شود.

۱۶-۲ تلفن‌های همراه نسل اول: صدای آنالوگ

صحبت از سیاست و روشهای بازاریابی دیگر پس است؛ اجازه دهید به کار اصلی خود یعنی تکنولوژی بپردازیم. حتی از اوایل قرن بیستم نیز تلفنهای متحرک رادیو-سیسم در زمینه‌های نظامی و مسافرتها دریایی کارایی خود را به اثبات رسانده بودند. در سال ۱۹۴۶، اولین تلفنهای مخصوص اتومبیل در سنت لوئیس راه اندازی شدند. در این سیستم آنتنهای (فرستنده-گیرنده) بزرگی روی ساختمانهای بلند نصب شده بود، و ارسال و دریافت روی یک کانال واحد صورت می‌گرفت. صحبت کردن و شنیدن در آن واحد امکان نداشت: برای صحبت کردن باید یک دکمه را فشار می‌دادید، و با رها کردن آن دیگر نمی‌توانستید حرف بزنید. تا اوایل دهه ۱۹۵۰، این سیستم، که به فشار بده-حرف بزن (push-to-talk) معروف بود، در بسیاری از شهرهای بزرگ آمریکا نصب شد. این سیستم امروزه هم در اتومبیلهای پلیس و تاکسی تلفنی مورد استفاده قرار می‌گیرد.

در دهه ۱۹۶۰، IMTS (سیستم تلفن همراه بهبود یافته - Improved Mobile Telephone System) نصب و راه اندازی شد. این سیستم نیز از فرستنده-گیرنده‌های قوی (۲۵۰ وات) در نقاط مرتفع استفاده می‌کرد، ولی برای ارسال و دریافت دو فرکانس متفاوت داشت، که بدین ترتیب نیازی به دکمه «فشار بده-حرف بزن» نبود. علاوه بر

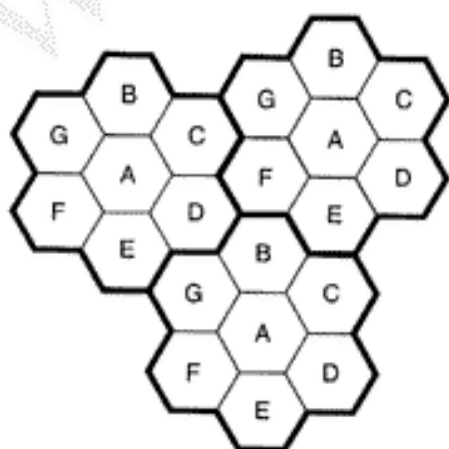
آن، برای ارسال نیز از فرکانسهای متعددی استفاده می‌شد، و خطر شنیده شدن صحبت دیگران (بر خلاف تلفنهای تاکسی تلفنی) نیز وجود نداشت.

سیستم IMTS از ۲۳ کانال (در طیف 150-450 Mhz) پشتیبانی می‌کرد. بدلیل کم بودن تعداد این کانالها، کاربران مجبور بودند برای شنیدن بوق آزاد مدت زیادی صبر کنند. همچنین بدلیل قدرت زیاد فرستنده-گیرنده های این سیستم، تا شعاع چند صد کیلومتری در سیستمهای رادیویی اختلال ایجاد می‌کردند. همه این دلایل IMTS را به سیستمی غیر عملی تبدیل کرده بود.

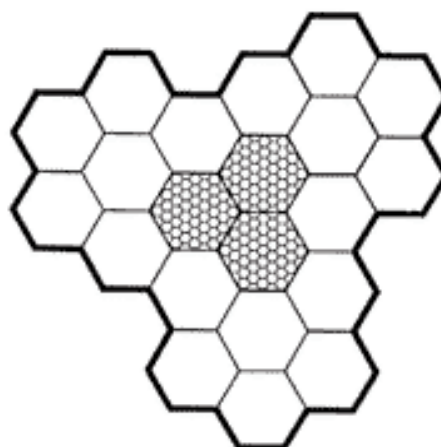
سیستم تلفن همراه پیشرفته

در سال ۱۹۸۲، با اختراع AMPS (سیستم تلفن همراه پیشرفته - Advanced Mobile Phone System) توسط شرکت Bell Labs اوضاع تغییر کرد. این سیستم در انگلستان (با نام TACS) و ژاپن (با نام MCS-L1) نیز نصب و بکار گرفته شد. با آنکه این سیستم دیگر تکنولوژی برتر محسوب نمی‌شود، اما از آنجائیکه بمنظور سازگاری با گذشته بسیاری از ویژگیهای بنیادی آن در تلفن همراه دیجیتال (DAMPS) لحاظ شده است، قدری درباره آن صحبت خواهیم کرد.

در تمام سیستمهای تلفن همراه، یک منطقه جغرافیایی به تعدادی سلول (cell) تقسیم می‌شود - که بهمین دلیل به آن تلفن سلولی (cell phone) نیز می‌گویند. در AMPS هر سلول ۱۰ تا ۲۰ کیلومتر قطر دارد؛ در سیستمهای دیجیتالی این سلولها کوچکترند. هر سلول دارای تعدادی فرکانس است، که سلولهای همسایه از آنها استفاده نمی‌کنند. ایده کلیدی در سیستمهای سلولی که باعث شده تا ظرفیت آنها بسیار بیشتر از سیستمهای قبلی باشد، استفاده از سلولهای نسبتاً کوچک و بکارگیری فرکانسها در سلولهای نزدیک هم (البته، نه سلولهای مجاور) است. در حالیکه یک سیستم IMTS با قطر ۱۰۰ کیلومتر می‌تواند روی هر فرکانس یک تماس داشته باشد، یک سیستم AMPS قادر است (با جدا کردن فرکانس سلولهای مجاور) روی هر فرکانس ۱۰ تا ۱۵ تماس برقرار کند. هر چه اندازه سلولها کوچکتر (و تعداد آنها بیشتر) باشد، ظرفیت سیستم بالاتر خواهد رفت. با کوچکتر شدن سلولها می‌توان قدرت فرستنده-گیرنده ها را نیز کمتر کرد، و از تجهیزات ساده تر و ارزاتری استفاده کرد. طبق مقررات FCC، حداکثر قدرت تشعشعی دستگاههای تلفن همراه ۰.۶ وات، و فرستنده-گیرنده های مخصوص اتومبیل ۳ وات تعیین شده است.



(الف)



(ب)

شکل ۲-۴۱. (الف) فرکانسهای سلولهای مجاور یکسان نیستند. (ب) برای افزایش

تعداد کاربران، می‌توان از سلولهای کوچکتر استفاده کرد.

ایده تکرار فرکانسها را در شکل ۲-۴۱ (الف) ملاحظه می کنید. سلولها تقریباً به شکل دایره هستند، ولی برای سادگی کار بهتر است آنها را شش ضلعی فرض کنیم. در این شکل تمام سلولها هم اندازه اند، و در گروههای هفت تایی دسته بندی شده اند. هر حرف نشان دهنده مجموعه ای از فرکانسها است. توجه کنید که هر مجموعه فرکانسی با نزدیکترین سلول مشابه حداقل و سلول فاصله دارد، که این باعث به حداقل رسیدن تداخل فرکانسها خواهد شد.

یکی از بزرگترین معضلات سیستمهای تلفن همراه، یافتن نقاط مناسب برای نصب آنتنهای مرکزی سلولهاست. این مشکل باعث شده تا شرکتها تلفن برای استفاده از مناره های رفیع کلیساها دست بدامن آنها شوند. وقتی در یک منطقه تعداد کاربران افزایش می یابد و سیستم دیگر جوابگوی بار مکالمات نیست، (با کم کردن قدرت آنتنها) هر سلول به سلولهای کوچکتر (microcell) تقسیم می شود، تا بتوان از فرکانسها بدفعات بیشتر استفاده کرد - شکل ۲-۴۱ (ب) را ببینید. در مواقعی که تعداد زیادی تلفن همراه برای مدتی کوتاه در یک نقطه گرد می آیند (مانند مسابقات ورزشی و کنسرتها موسیقی)، شرکتها تلفن با استقرار آنتنهای متحرک (که ارتباط ماهواره ای دارند) موقتاً سلولهای کوچکتری ایجاد می کنند. اینکه اندازه یک سلول چقدر باید باشد، مسئله پیچیده ایست که در (Hac, 1995) درباره آن بحث شده است.

در مرکز هر سلول یک ایستگاه قرار دارد که تمام تلفنهای داخل سلول امواج خود را به آن می فرستند. در سیستمهای کوچک، تمام این ایستگاهها به یک دستگاه مرکزی بنام MTSO (مرکز سوییچینگ تلفن همراه - Mobile Telephone Switching Office) یا MSC (مرکز سوییچینگ همراه - Mobile Switching Center) متصل می شوند. در سیستمهای بزرگتر چندین MTSO وجود دارند، که بنویه خود به یک MTSO بالاتر متصلند. MTSO ها، در واقع، همان ایستگاههای پایانی سیستم تلفن همراه هستند، که به حداقل یک ایستگاه پایانی سیستم تلفن ثابت نیز ارتباط دارند. ارتباط MTSO ها با تلفنهای همراه، با یکدیگر و با مراکز PSTN از طریق یک شبکه سوییچینگ بسته ای صورت می گیرد.

هر تلفن همراه در هر لحظه در یک سلول (و تحت کنترل ایستگاه مرکزی آن) قرار دارد. وقتی این تلفن سلول را ترک می کند، ایستگاه مرکزی متوجه ضعیف شدن سیگنال آن شده و از تمام ایستگاههای مجاور میزان قدرت دریافتی از این تلفن را می پرسد. سپس، این ایستگاه کنترل تلفن مزبور را به ایستگاهی که بیشترین قدرت را گزارش کرده (و در واقع، تلفن وارد آن شده)، تحویل می دهد. رئیس جدید نیز به تلفن همراه اطلاع می دهد که (اگر مایل به ادامه مکالمه است) کانال خود را عوض کند (چون فرکانس قبلی آن در هیچیک از سلولهای همسایه وجود ندارد). این فرآیند (که به آن پاس کاری - handoff - گفته می شود) تقریباً 300 msec طول می کشد. تخصیص کانال توسط MTSO صورت می گیرد (چون ایستگاههای مرکزی چیزی جز رله های رادیویی ساده نیستند).

پاس کاری می تواند به دو طریق صورت گیرد. در پاس کاری نرم (soft handoff) تلفن همراه قبل از آن که ایستگاه قدیم آنرا قطع کند، به ایستگاه جدید متصل می شود. در این روش کاربر هیچ نوع قطعی احساس نمی کند. اما مشکل این روش آنست که دستگاه تلفن همراه باید بتواند در آن واحد خود را روی دو فرکانس (ایستگاه قبلی و ایستگاه جدید) تنظیم کند. تلفنهای همراه نسل اول و دوم هیچکدام قادر به انجام چنین کاری نیستند.

در پاس کاری سخت (hard handoff) ایستگاه قدیمی قبل از اتصال تلفن همراه به ایستگاه جدید، آنرا قطع می کند. اگر ایستگاه جدید به هر دلیلی (مثلاً، نداشتن باند خالی) نتواند تلفن را تحویل بگیرد، ارتباط کاربر یکباره قطع خواهد شد. این یکی از مشکلات اجتناب ناپذیر تلفنهای همراه نسل اول و دوم است.

کانال ها

سیستم AMPS دارای ۸۳۲ کانال دو-طرفه همزمان است، که هر کانال خود از دو کانال یکطرفه ساده تشکیل می شود (۸۳۲ کانال دریافت و ۸۳۲ کانال ارسال). کانالهای یکطرفه دریافت روی فرکانسهای 824-849 MHz و کانالهای یکطرفه ارسال روی فرکانسهای 869-894 MHz کار می کنند (پهنای باند هر کانال 30 kHz است). همانطور که می بینید، AMPS از FDM برای تقسیم پهنای باند استفاده می کند.

امواج در فرکانس 800 MHz طول موجی در حدود 40 cm دارند، و به خط مستقیم حرکت می کنند. درختان و گیاهان این امواج را جذب می کنند، و در برخورد با زمین نیز منعکس می شوند. موجی که به ایستگاه مرکزی سلول می رسد، می تواند مستقیماً از گوشی تلفن همراه آمده باشد، و یا انعکاس آن از سطح زمین یا ساختمانها باشد، که این می تواند باعث پژواک (echo) یا محوشدگی چندمسیره (multipath fading) شود. گاهی حتی امکان شنیدن مکالماتی که در مسافتی طولانی چندین بار به سطح زمین خورده و منعکس شده، نیز وجود دارد.

۸۳۲ کانال AMPS به چهار دسته تقسیم می شوند:

۱. کانالهای کنترل (ایستگاه به تلفن همراه) برای مدیریت سیستم
۲. کانالهای فراخوانی (ایستگاه به تلفن همراه) برای اعلام اینکه مکالمه ای در انتظار کاربر است
۳. کانالهای دسترسی (دو طرفه) برای برقراری مکالمات و تخصیص کانال
۴. کانالهای داده (دو طرفه) برای صدا، فکس، یا داده

تعداد کانالهای کنترل ۲۱ عدد است، که بطور ثابت در PROM تلفنهای همراه نوشته شده است. از آنجائیکه استفاده از فرکانسهای مشابه در سلولهای مجاور مجاز نیست، تعداد کانالهای قابل استفاده در هر سلول بسیار کمتر از ۸۳۲ (و در واقع، چیزی نزدیک به ۴۵ کانال) است.

مدیریت مکالمه

هر تلفن همراه در سیستم AMPS دارای یک شماره سریال ۳۲ بیتی و یک شماره تلفن ۱۰ رقمی است، که در PROM آن نوشته شده است. شماره تلفن همراه دارای یک کد ناحیه ۳ رقمی (که بصورت ۱۰ بیتی کد شده) و یک شماره مشترک ۷ رقمی (که بصورت ۲۴ بیتی کد شده) - مجموعاً ۳۴ بیت - است.

وقتی تلفن روشن می شود، ۲۱ کانال کنترل از پیش برنامه ریزی شده را اسکن می کند، تا قویترین سیگنال را پیدا کند. سپس، شماره سریال ۳۲ بیتی و شماره تلفن ۳۴ بیتی خود را منتشر می کند. تمام اطلاعات کنترلی در AMPS دیجیتال هستند (برخلاف کانالهای صدا، که آنالوگ می باشند)، و این اطلاعات به دفعات و همراه با کدهای تصحیح خطا منتشر می شوند.

وقتی ایستگاه مرکزی سلول این اعلام را دریافت کرد، آنرا به MTSO می فرستد، که آن هم (پس از ثبت کاربر جدید) محل وی را به نزدیکترین MTSO اعلام می کند. در حالت عادی، یک تلفن همراه هر ۱۵ دقیقه خود را به ایستگاه مرکزی معرفی می کند.

برای برقراری یک تماس، کاربر (بعد از روشن کردن تلفن) شماره مورد نظر را وارد کرده، و دکمه CALL را فشار می دهد. تلفن این شماره را به همراه کد شناسایی خود روی یکی از کانالهای دسترسی (access channel) به ایستگاه مرکزی می فرستد. (اگر تداخلی پیش آید، تلفن این عملیات را بعداً تکرار می کند). وقتی ایستگاه مرکزی سلول این درخواست را دریافت کرد، به MTSO اطلاع می دهد. اگر تماس گیرنده یکی از مشترکین آن MTSO باشد، MTSO بدنبال یک کانال خالی می گردد تا به وی تخصیص دهد. اگر کانال خالی موجود بود، شماره آن روی کانال کنترل به تلفن برگشت داده می شود. با گرفتن کانال دسترسی از MTSO، تلفن همراه بطور خودکار به آن

کانال سوئیچ کرده و منتظر می ماند تا طرف مقابل گوشی را بردارد.

تماسهای ورودی به طریق دیگری عمل می کنند. تلفنهایی که بیکار هستند، دائماً به کانال فراخوانی (paging channel) گوش می کنند، تا پیامهایی را که برای آنان می رسد دریافت کنند. وقتی یک شماره تلفن همراه گرفته می شود (خواه از یک تلفن ثابت یا یک تلفن همراه دیگر)، یک بسته به MTSS ی آن تلفن ارسال می شود تا محل وی را پیدا کند. MTSS نیز که محل تمام مشترکین فعال خود را می داند، یک بسته به ایستگاه مرکزی آن تلفن می فرستد، که آن هم بنوبه خود یک پیام (با مضمون: «تلفن 14، تو آنجایی؟») روی کانال فراخوانی منتشر می کند. تلفن مقصد هم با ارسال پاسخ "Yes" روی کانال دسترسی جواب می دهد. وقتی ایستگاه مرکزی پاسخ "Yes" را دریافت کرد، با پیامی مانند این عکس العمل نشان می دهد: «تلفن 14، روی کانال 3 به تماس جواب بده». در اینجا، تلفن همراه به کانال 3 سوئیچ کرده، و شروع به زنگ زدن (یا نواختن یکی از ملودیهای عجیب و غریبی که این روزها همه جا شنیده می شود) می کند.

۲-۶-۲ تلفن های همراه نسل دوم: صدای دیجیتال

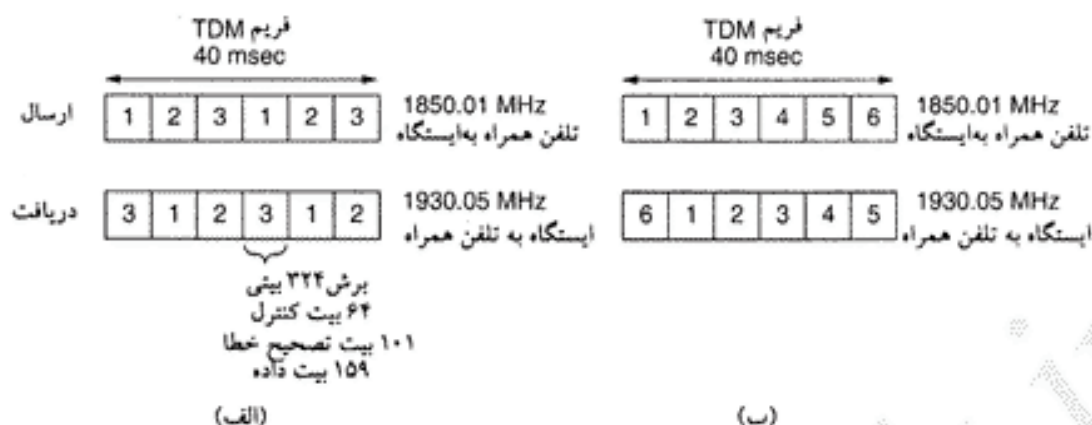
تلفن همراه نسل اول آنالوگ بود، و تلفن همراه نسل دوم دیجیتال. همانطور که نسل اول استاندارد جهانی نداشت، نسل دوم هم چنین استاندارد جهانی ندارد. امروزه چهار سیستم تلفن همراه نسل دوم مشغول بکار هستند: D-AMPS، GSM، CDMA و PDC. در این قسمت سه سیستم اول را بررسی خواهیم کرد؛ PDC فقط در ژاپن راه اندازی شده، و در واقع همان D-AMPS است که برای سازگاری با سیستمهای نسل اول آنالوگ ژاپن تغییراتی در آن صورت گرفته است. اصطلاح PCS (سرویسهای مخابرات شخصی - Personal Communications Services) از لحاظ فنی به معنای تلفن همراه دیجیتال با فرکانس 1900 MHz است، اما امروزه این کلمه عمدتاً به سیستمهای نسل دوم (یعنی، دیجیتال) تعبیر می شود.

D-AMPS : سیستم تلفن همراه پیشرفته دیجیتال

نسل دوم سیستمهای AMPS (که کاملاً دیجیتال است) D-AMPS (سیستم تلفن همراه پیشرفته دیجیتال - Digital Advanced Mobile Phone System) نام دارد. این سیستم تحت استاندارد بین المللی IS-54 (و خلف آن تحت استاندارد IS-136) تعریف شده است. سیستم D-AMPS پگونه ای طراحی شده که با AMPS سازگاری کامل داشته باشد، به همین دلیل تلفنهای نسل اول و دوم می توانند همزمان در یک سلول کار کنند. D-AMPS از همان کانالهای 30 kHz نسل اول (و دقیقاً با همان فرکانسها) استفاده می کند، بنابراین در یک سلول یک کانال می تواند آنالوگ کار کند، در حالیکه کانال مجاور در حال کار بصورت دیجیتال است. این MTSS ی مسئول سلول است که (بر اساس نسبت تلفنهای نسل اول و دوم در هر سلول) تعیین می کند کدام کانالها آنالوگ و کدام کانالها دیجیتال کار کنند - و (با تغییر این نسبت) می تواند ترکیب کانالها را بصورت دینامیک تغییر دهد.

بعد از عملیاتی شدن سرویسهای D-AMPS و بمنظور پاسخگویی به تقاضای رو به فزونی بازار، باند فرکانسی جدیدی به آن تخصیص داده شد. کانالهای دریافت از کاربر (upstream) در محدوده 1850-1910 MHz، و کانالهای ارسال به کاربر (downstream) در محدوده 1930-1990 MHz هستند. طول موج در این باند فقط 16 cm است، بنابراین آنتن استاندارد (آنتن یک چهارم طول موج) تلفنهایی که در این باند کار می کنند، فقط 4 cm خواهد بود، و به همین دلیل این تلفنها می توانند بسیار کوچکتر ساخته شوند. با این حال، اغلب تلفنهای D-AMPS هر دو باند فرکانسی (850 MHz و 1900 MHz) را پشتیبانی می کنند.

در یک تلفن همراه D-AMPS، سیگنال خروجی میکروفون بعد از دیجیتایز شدن با استفاده از مدلهایی که بسیار بهینه تر از مدولاسیون دلتا (delta modulation) و کد کردن پیشگویانه (predictive encoding) هستند،



شکل ۲-۴۲. (الف) یک کانال D-AMPS با سه کاربر. (ب) یک کانال D-AMPS با شش کاربر.

فشرده می شود. این مدل های فشرده سازی با استفاده از خصوصیات دستگاه صوتی انسان، پهنای باند لازم را از 56-kbps به 8 kbps یا کمتر تقلیل می دهند. این کار که توسط دستگاهی بنام vocoder انجام می شود (Bellamy, 2000) را ببینید، در دستگاه تلفن صورت می گیرد نه در ایستگاه مرکزی، که بدین ترتیب تعداد بیت های منتقل شده روی لینک هوایی کاهش خواهد یافت. در تلفن های ثابت این تکنیک هیچ مزیتی بدنبال ندارد، چون کاستن از ترافیک مدار پایانی هیچ تأثیری روی ظرفیت کل سیستم نخواهد داشت.

فشرده کردن صدای دیجیتال تلفن های همراه (همان کاری که در D-AMPS انجام می شود) مزیت فوق العاده ای به همراه دارد، چون می توان با استفاده از حالتی پلکس تقسیم زمانی (TDM) یک زوج فرکانسی (ارسال و دریافت) را بین سه کاربر به اشتراک گذاشت. هر زوج فرکانسی از 25 frame/sec (هر فریم معادل 40 msec) پشتیبانی می کند. هر فریم نیز بنوبه خود به شش بُرش زمانی 6.67 msec تقسیم می شود (شکل ۲-۴۲ (الف) را ببینید).

هر فریم به سه کاربر سرویس (ارسال و دریافت) می دهد، که به نوبت از آن استفاده می کنند. برای مثال در بُرش زمانی 1 در شکل ۲-۴۲ (الف)، کاربر ۱ می تواند در حال ارسال به ایستگاه مرکزی باشد، در حالیکه در همان زمان کاربر ۳ در حال دریافت است. هر بُرش زمانی طولی معادل 324 بیت دارد، که 64 بیت آن برای مصارف کنترلی اختصاص یافته، و بقیه 260 بیت در اختیار کاربر است. از این 260 بیت، 101 بیت برای کدهای تصحیح خطا (که در لینک های هوایی پُر نویز بسیار هم لازمند) بکار می رود، که بدین ترتیب فقط 159 بیت برای انتقال صدا باقی می ماند. با احتساب 50 بُرش زمانی در هر ثانیه، پهنای باند موجود برای صدای فشرده فقط 8 kbps (پهنای باند PCM) است.

با استفاده از الگوریتم های فشرده سازی بهتر، می توان پهنای باند لازم را حتی به 4 kbps کاهش داد، تا شش کاربر بتوانند در آن واحد از یک فریم استفاده کنند (شکل ۲-۴۲ ب). از دید شرکتهای تلفن، توانایی فشرده کردن سه یا شش کاربر D-AMPS در کانالی که فقط یک کاربر AMPS می تواند از آن استفاده کند، یک بُرد واقعی است و دلیل محبوبیت PCM نزد شرکتهای تلفن همراه نیز همین است. البته کیفیت صدای 4 kbps هرگز به پای صدای 56 kbps نمی رسد، اما شرکتهای تلفن همراه هم کمتر روی کیفیت صدای سرویسهای خود تبلیغ می کنند. در مورد سرویسهای داده، کیفیت یک خط 8 kbps حتی با مودمهای 9600-bps نیز قابل مقایسه نیست.

ساختار کنترلی D-AMPS نسبتاً پیچیده است: هر ۱۶ فریم تشکیل یک ابرفریم (superframe) می دهند، که اطلاعات کنترلی بدفعات محدود در این ابرفریم گنجانده می شوند. در کل ۶ کانال کنترلی مورد استفاده قرار

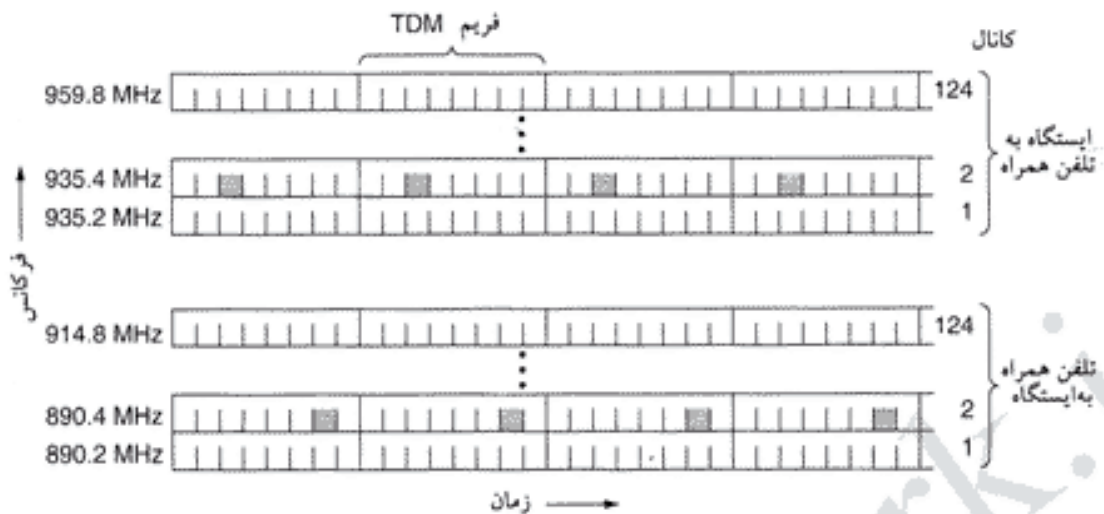
می‌گیرد: پیکربندی سیستم، کنترل بی‌درنگ (real-time) و غیر بی‌درنگ (nonreal-time)، فراخوانی (paging)، پاسخ دسترسی (access response) و پیام کوتاه (short message). طرز کار D-AMPS از نظر مفهومی شبیه AMPS است. وقتی گوشی تلفن همراه روشن می‌شود، با ایستگاه مرکزی تماس گرفته و بعد از معرفی خود، به کانالهای کنترل گوشی می‌کند. در این سیستم هم تعیین محل (سلول) تلفن همراه بر عهده MTSO است. یکی از تفاوت‌های D-AMPS با AMPS در نحوه پاس‌کاری است. در AMPS این کار بطور کامل بر عهده MTSO است و دستگاه تلفن هیچ دخالتی در آن ندارد. اما همانطور که در شکل ۲-۴۲ دیده می‌شود، در D-AMPS در $\frac{1}{4}$ از کل زمان تماس دستگاه تلفن هیچ کاری انجام نمی‌دهد، و می‌تواند از این زمان خالی برای اندازه‌گیری کیفیت خط استفاده کند. وقتی تلفن متوجه می‌شود که قدرت سیگنال در حال کاهش است، به MTSO اطلاع داده و MTSO هم ارتباط را قطع می‌کند، که در این لحظه تلفن می‌تواند به یک ایستگاه (فرکانس) قویتر سوییچ کند - به این تکنیک MAHO (پاس‌کاری یکمک تلفن همراه - Mobile Assisted HandOff) می‌گویند. در D-AMPS نیز (مانند AMPS) این فرآیند حدوداً 300 msec طول می‌کشد.

GSM : سیستم سراسری مخابرات همراه

سیستم D-AMPS بصورت گسترده در ایالات متحده (و بصورت اصلاح شده در ژاپن) در حال استفاده است. اما تقریباً در تمام نقاط دیگر دنیا از سیستمی بنام GSM (سیستم سراسری مخابرات همراه - Global System for Mobile Communications) استفاده می‌کنند، و حتی در آمریکا نیز این سیستم در نقاط محدودی راه‌اندازی شده است. GSM و D-AMPS از چند نظر شبیه یکدیگرند: هر دوی آنها سیستمهای سلولی هستند؛ هر دوی آنها از FDM و فرکانسهای متفاوت برای ارسال و دریافت (که فرکانس دریافت تلفن بیشتر از فرکانس ارسال آن است - 80 MHz بالاتر در D-AMPS و 55 MHz بالاتر در GSM) استفاده می‌کنند؛ و در هر دو سیستم چندین تلفن با استفاده از TDM مشترکاً از یک زوج فرکانس استفاده می‌کنند. اما در GSM پهنای کانالها بسیار بیشتر از D-AMPS (200 kHz در مقابل 30 kHz) و تعداد کاربران هر کانال نیز نسبتاً کمتر است (8 کاربر در مقابل 3 کاربر)، که در نتیجه نرخ داده آن بسیار بهتر از D-AMPS خواهد بود. در زیر توضیح مختصری درباره ویژگیهای اصلی GSM خواهیم داد. اما توجه داشته باشید که استاندارد چاپی GSM متجاوز از ۵۰۰۰ برگ است، و در آن جنبه‌های فنی و مهندسی این سیستم به تفصیل تشریح شده، که ما در اینجا بد آن‌ها نخواهیم پرداخت.

همانطور که قبلاً هم گفته شد (و در شکل ۲-۴۳ می‌بینید)، هر باند فرکانسی 200 kHz پهنای دارد. هر سیستم GSM دارای ۱۲۴ زوج کانال یکطرفه ساده (simplex) است، که هر کانال 200 kHz پهنای دارد و از ۸ ارتباط همزمان (بصورت ماتریکس تقسیم زمانی) پشتیبانی می‌کند. به هر ایستگاه فعال در هر بُرش زمانی یک زوج کانال تخصیص داده می‌شود، که بدین ترتیب تعداد کانالهای موجود به 992 می‌رسد. البته (بدلیل جلوگیری از تداخل فرکانسی ایستگاههای مجاور) این تعداد کانال برای تمام ایستگاهها قابل استفاده نیست. در شکل ۲-۴۳، هشت بُرش زمانی خاکستری می‌بینید که همگی متعلق به یک تماس هستند (۴ تا برای ارسال، و ۴ تا برای دریافت). ارسال و دریافت همزمان انجام نمی‌شود، چون دستگاههای رادیویی GSM برای سوییچ کردن فرکانس به زمان نیاز دارند، و نمی‌توانند همزمان هر دو کار را انجام دهند. اگر به یک ایستگاه فرکانس 890.4/935.4 MHz اختصاص داده شده باشد و بُرش زمانی 2 بخواهد چیزی به این ایستگاه بفرستد، باید از بُرشهای خاکستری رنگ قسمت پائین تصویر (به هر تعداد که نیاز دارد) برای کار خود استفاده کند.

بُرشهای TDM نشان داده شده در شکل ۲-۴۳ بخشی از یک سلسله مراتب پیچیده فریم‌بندی هستند. هر بُرش TDM دارای ساختار خاصیتیست که ترکیب آنها نیز به روشی خاص تشکیل یک فریم چندگانه می‌دهد. در شکل ۲-۴۴ شکل ساده شده‌ای از این ساختار سلسله مراتبی را ملاحظه می‌کنید. همانطور که در این شکل



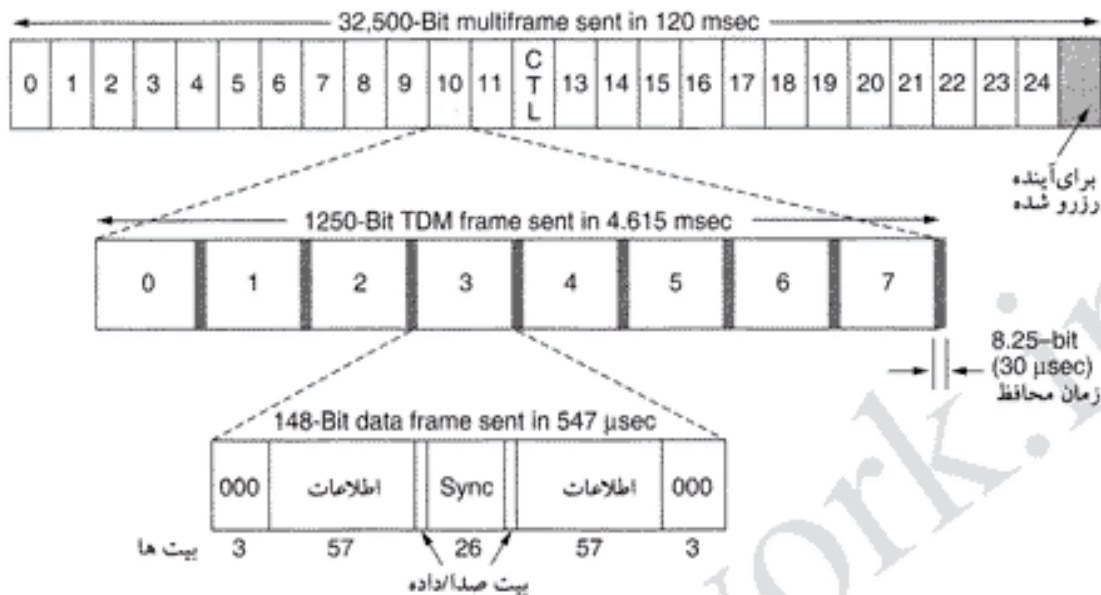
شکل ۲-۲۳. GSM از ۱۲۴ کانال فرکانسی استفاده می کند، که هر یک از آنها به ۸ بُرش زمانی تقسیم می شود.

می بینید، هر بُرش TDM از یک فریم داده ۱۴۸ بیتی تشکیل می شود، که کانال را برای مدت $577 \mu\text{sec}$ (شامل $30 \mu\text{sec}$ زمان حفاظت در هر بُرش) به اشغال خود در می آورد. بمنظور همزمان کردن فریمها، هر فریم داده با ۳ بیت 0 شروع و ختم می شود. هر فریم دو فیلد اطلاعاتی (Information) ۵۷ بیتی دارد، که هر کدام دارای ۱ بیت کنترلی هستند که مشخص می کند این فیلد اطلاعاتی حاوی داده است یا صدا. بین این فیلدهای اطلاعاتی یک فیلد نظم دهنده (Sync) ۲۶ بیتی وجود دارد که گیرنده از آن برای همزمان شدن با فرستنده استفاده می کند. ارسال هر فریم داده $547 \mu\text{sec}$ طول می کشد، ولی از آنجائیکه هر کانال بین ۸ ایستگاه به اشتراک گذاشته شده، فرستنده فقط در هر 4.615 msec می تواند یک فریم بفرستد. نرخ داده (ناخالص) هر کانال $270,833 \text{ bps}$ است که ۸ کاربر مشترکاً از آن استفاده می کنند. با یک تقسیم ساده مشخص می شود که نرخ داده ناخالص هر کاربر 33.854 kbps (بیش از دو برابر سیستمهای D-AMPS، یعنی 16.2 kbps) است. اما در اینجا هم مانند AMPS سرآیندها بخش زیادی از پهنای باند را می بلعند، و در نهایت 24.7 kbps برای داده واقعی کاربر (قبل از تصحیح خطا) باقی می ماند. آنچه پس از تصحیح خطا برای صدا باقی می ماند، 13 kbps است که باز هم کیفیت صدای بسیار بهتری نسبت به D-AMPS می دهد (البته به قیمت مصرف پهنای باند بسیار بیشتر).

همانطور که در شکل ۲-۲۴ می بینید، هر ۸ فریم تشکیل یک فریم TDM و هر ۲۶ فریم TDM تشکیل یک فریم چندگانه ۱۲۰-msec را می دهند. از ۲۶ فریم TDM، فریم ۱۲ برای کارهای کنترلی مورد استفاده قرار می گیرد، فریم ۲۵ نیز برای مصارف آتی کنار گذاشته شده است، و فقط ۲۴ فریم برای ارسال و دریافت باقی می ماند.

علاوه بر ساختار ۲۶ فریمی شکل ۲-۲۴، در GSM از ساختار دیگری با ۵۱ فریم نیز استفاده می شود. در این ساختار نیز برخی از فریمها کارکرد کنترلی (مدیریت سیستم) دارند. کانال کنترل پخش (broadcast control channel) یکی از این کانالهاست، که استریم خروجی پیوسته ایست که از ایستگاه مرکزی منتشر می شود و شامل هویت ایستگاه و اطلاعاتی درباره وضعیت کانالهای آن می شود. تمام دستگاههای تلفن همراه دائماً قدرت این سیگنال را چک می کنند تا بتوانند موقعیت خود را در میان سلولها تشخیص دهند.

کانال کنترل اختصاصی (dedicated control channel) یکی دیگر از کانالهای کنترلی است که برای به روز در آوردن موقعیت تلفنهای همراه، ثبت آنها و برقراری تماس بکار می رود. بویژه، هر ایستگاه مرکزی دارای پایگاه



شکل ۲-۴۴. بخشی از ساختار فریم بندی GSM.

داده‌ای از تمام تلفنهای موجود در محدوده قدرت خود است، که با استفاده از اطلاعاتی که روی این کانال فرستاده می‌شود، به روز در می‌آید.

دیگر کانال کنترلی، کانال کنترل مشترک (common control channel) است که به سه زیرکانال تقسیم می‌شود. اولین زیرکانال، کانال فراخوانی (paging channel) است که ایستگاه مرکزی از آن برای اعلام تماس ورودی استفاده می‌کند (و هر تلفن دائماً این کانال را چک می‌کند تا ببیند آیا تماسی دارد یا خیر). زیرکانال دوم، کانال دسترسی تصادفی (random access channel) است، که اجازه می‌دهد تا کاربران درخواستی برای یک بُرش از کانال کنترلی اختصاصی را روی آن ارسال کنند؛ کاربران از این بُرش برای برقراری تماس با دیگران استفاده می‌کنند. اگر دو کاربر همزمان وارد این کانال شوند، تداخل پیش می‌آید و باید (پس از کمی تأخیر) عملیات را از نو تکرار کنند. بُرش اختصاص داده شده به کاربر روی زیرکانال سوم، کانال اختصاص دسترسی (access grant channel)، به وی اعلام می‌شود.

CDMA: دسترسی چندگانه با تقسیم کد

سیستمهای D-AMPS و GSM هر دو سیستمهایی نسبتاً سستی و معمولی هستند، که از FDM و TDM برای تقسیم طیف فرکانسی به کانالها و تقسیم کانالها به بُرشهای زمانی استفاده می‌کنند. اما بازیگر سومی بنام CDMA (دسترسی چندگانه با تقسیم کد - Code Division Multiple Access) نیز در این صحنه حاضر است، که به روشی کاملاً متفاوت بازی می‌کند. وقتی CDMA برای اولین بار در صنعت مطرح شد، همان واکنشی را برانگیخت که پیشنهاد کریستف کلمب برای رسیدن به هندوستان از راه سفر به غرب. اما در نتیجه پایداری و مقاومت یک شرکت بنام Qualcomm، اکنون CDMA به جایی رسیده که نه تنها بعنوان یک سیستم قابل قبول مطرح است، بلکه به آن به چشم تنها مبنای مطمئن برای سیستمهای تلفن همراه نسل سوم نگاه می‌کنند. در ایالات متحده، CDMA حتی اکنون (در سیستمهای نسل دوم) نیز بعنوان رقیبی جدی برای D-AMPS مطرح است - برای مثال، شرکت Sprint PCS از CDMA استفاده می‌کند، در حالیکه AT&T Wireless از D-AMPS. استاندارد CDMA در سند IS-95 تدوین شده، و گاهی به این نام هم شناخته می‌شود؛ نام cdmaOne نیز یکی از نامهای تجاری آن است.

CDMA بکلی از AMPS، D-AMPS و GSM متفاوت است: بجای تقسیم طیف فرکانسی به کانالهای باریک، CDMA اجازه می دهد تا تمام تلفن ها و ایستگاهها از تمام طیف فرکانسی برای ارسال و دریافت استفاده کنند. و برای تفکیک آنها از یکدیگر از تنوری رمزگذاری (coding theory) استفاده می کنند. در سیستمهای CDMA این فرض که فریمهای تداخل کرده غیر قابل استفاده اند را نیز بکلی کنار می گذارد. و بجای آن فرض می کند که این سیگنالها بصورت خطی با هم جمع می شوند.

برای درک بهتر CDMA از یک مثال آشنا استفاده می کنیم: سالتی پر از جمعیت که دو به دو مشغول صحبت هستند. TDM مانند آن است که این افراد را وسط سالن جمع کنید، ولی فقط به نوبت به آنها اجازه صحبت کردن بدهید. FDM مانند آن است که این افراد را با فاصله زیاد از یکدیگر بچینید، و به آنها اجازه دهید دو به دو همزمان (و البته مستقل) با هم صحبت کنند. CDMA نیز مانند این است که همه افراد را وسط سالن جمع کنید، ولی آنها (همزمان) به زبانهای مختلف با هم صحبت کنند. برای مثال، دو نفری که به فرانسه با هم حرف می زنند، هر چیزی غیر از کلمات فرانسوی را بعنوان پارازیت (نویز) شنیده می گیرند. نکته کلیدی در CDMA همین استخراج سیگنال مورد نظر (و دور ریختن هر چیز دیگر) است. در زیر روش کار یک سیستم ساده شده CDMA را توضیح می دهیم.

در CDMA، هر بیت به m فاصله زمانی کوتاه، موسوم به چپ (chip)، تقسیم می شود. معمولاً هر بیت دارای ۶۴ یا ۱۲۸ چپ است، ولی در اینجا برای سادگی هر بیت را به فقط ۸ چپ تقسیم کرده ایم. به هر تلفن همراه (یا ایستگاه) یک کد m بیتی منحصر بفرد، که به آن توالی چپ (chip sequence) می گویند، اختصاص داده می شود. برای ارسال بیت ۱، ایستگاه توالی چپ خود را می فرستد، و برای ارسال بیت ۰ مکمل یک (one's complement) توالی چپ خود را - یک ایستگاه مجاز نیست هیچ چیز دیگری بفرستد. برای مثال، با فرض $m = 8$ ، اگر ایستگاهی بنام A دارای توالی چپ 00011011 باشد، برای ارسال بیت ۱ توالی 00011011 و برای ارسال ۰ توالی 11100100 را می فرستد.

افزایش مقدار اطلاعات ارسالی از b bits/sec به mb chips/sec فقط وقتی امکانپذیر است که پهنای باند موجود m برابر شود، که بدین ترتیب CDMA به سیستمی با طیف گسترده تبدیل می شود. اگر پهنای باند موجود برای ۱۰۰ ایستگاه 1 MHz باشد، با تکنیکهای FDM هر ایستگاه فقط 10 kHz در اختیار خواهد داشت و می تواند حداکثر 10 kbps (با فرض 1 bit/Hz) ارسال کند. با CDMA هر ایستگاه می تواند از تمامی پهنای باند 1-MHz استفاده کند، و به سرعت 1 megachips/sec برسد. اگر تعداد چپ بر بیت کمتر از ۱۰۰ باشد، پهنای باند مؤثر CDMA باز هم بیشتر از FDM است (و مشکلات تخصیص کانال نیز دیگر وجود ندارد).

در این قسمت برای درک بهتر مطلب روش دو-علامتی را بکار می بریم، یعنی بجای ۰ باینری از -۱ و بجای ۱ باینری از +۱ استفاده خواهیم کرد. توالی چپها را نیز در پرانتز نمایش می دهیم، که بدین ترتیب توالی چپ ایستگاه A به $(+1 +1 -1 -1 +1 -1 -1 +1)$ تبدیل می شود. در شکل ۲-۴۵ (الف) توالی چپ چهار ایستگاه بنامهای A، B، C و D را می بینید؛ در شکل ۲-۴۵ (ب) نیز همین توالی ها را به روش دو-علامتی نشان داده ایم. هر ایستگاه توالی چپ خاص خود را دارد. فرض می کنیم S بردار m چپیی ایستگاه S، و \bar{S} بردار منفی (مکمل یک) آن است. تمام بردارهای توالی چپ متعامد (orthogonal) هستند، بعبارت دیگر ضرب داخلی نرمال شده هر دو بردار S و T $(S \cdot T)$ صفر است. با استفاده از روشی بنام کدهای والش (Walsh codes) می توان چنین بردارهایی تولید کرد. متعامد بودن دو بردار را به زبان ریاضی می توان چنین نوشت:

$$S \cdot T \equiv \frac{1}{m} \sum_{i=1}^m S_i T_i = 0$$

همانطور که خواهید دید، متعامد بودن بردارها یکی از ویژگیهای کلیدی این سیستم است. توجه داشته باشید که اگر $S \cdot T = 0$ ، آنگاه $S \cdot \bar{T} = 0$ ، ضرب داخلی نرمال شده هر بردار در خودش نیز 1 است:

$$S \cdot S = \frac{1}{m} \sum_{i=1}^m S_i S_i = \frac{1}{m} \sum_{i=1}^m S_i^2 = \frac{1}{m} \sum_{i=1}^m (\pm 1)^2 = 1$$

از آنجائیکه هر یک از جملات این دنباله 1 است، مجموع m جمله آن معادل m می شود، که بعد از ضرب در $\frac{1}{m}$ به عدد 1 خواهیم رسید. همچنین توجه داشته باشید که، $S \cdot \bar{S} = -1$.

همانطور که گفتیم، هر ایستگاه می تواند برای ارسال بیت 1 توالی چیب خود، و برای ارسال بیت 0 مکمل یک توالی چیب خود را بفرستد، و یا اصلاً سکوت کند و چیزی نفرستد. فعلاً فرض را بر این می گذاریم که تمام ایستگاهها سنکرون هستند، یعنی ارسال توالی های خود را در یک زمان شروع می کنند.

وقتی دو یا چند ایستگاه بطور همزمان شروع به ارسال می کنند، سیگنال دو-علامتی آنها بصورت خطی با هم جمع می شود. برای مثال، اگر (در دوره زمانی یک چیب) سه ایستگاه سیگنال +1 و یک ایستگاه سیگنال -1 بفرستند، مجموع آنها +2 خواهد شد. می توان این وضعیت را مانند جمع ولتاژها تصور کرد: مجموع سه ولتاژ +1 ولت و یک ولتاژ -1 ولت معادل +2 ولت می شود. در شکل ۲-۴۵ (ج) شش نمونه از ارسال همزمان یک یا چند ایستگاه را ملاحظه می کنید. در مثال اول، ایستگاه C مبادرت به ارسال یک بیت 1 می کند، بعبارت دیگر توالی چیب خود (01011100) را می فرستد. در مثال دوم، هر دو ایستگاه B و C یک بیت 1 می فرستند، که بدین ترتیب مجموع توالی های دو-علامتی آنها چنین خواهد شد:

$$(-1 -1 +1 -1 +1 +1 +1 -1) + (-1 +1 -1 +1 +1 +1 -1 -1) = (-2 \ 0 \ 0 \ 0 +2 +2 \ 0 -2)$$

A: 00011011
B: 00101110
C: 01011100
D: 01000010

(الف)

A: (-1 -1 -1 +1 +1 -1 +1 +1)
B: (-1 -1 +1 -1 +1 +1 +1 -1)
C: (-1 +1 -1 +1 +1 +1 -1 -1)
D: (-1 +1 -1 -1 -1 -1 +1 -1)

(ب)

شش مثال:

--1--	C	$S_1 = (-1 +1 -1 +1 +1 +1 -1 -1)$
-11--	B + C	$S_2 = (-2 \ 0 \ 0 \ 0 +2 +2 \ 0 -2)$
10--	A + B	$S_3 = (0 \ 0 -2 +2 \ 0 -2 \ 0 +2)$
101--	A + B + C	$S_4 = (-1 +1 -3 +3 +1 -1 -1 +1)$
1111	A + B + C + D	$S_5 = (-4 \ 0 -2 \ 0 +2 \ 0 +2 -2)$
1101	A + B + C + D	$S_6 = (-2 -2 \ 0 -2 \ 0 -2 +4 \ 0)$

(ج)

$S_1 \cdot C = (1 +1 +1 +1 +1 +1 +1 +1)/8 = 1$
 $S_2 \cdot C = (2 +0 +0 +0 +2 +2 +0 +2)/8 = 1$
 $S_3 \cdot C = (0 +0 +2 +2 +0 -2 +0 -2)/8 = 0$
 $S_4 \cdot C = (1 +1 +3 +3 +1 -1 +1 -1)/8 = 1$
 $S_5 \cdot C = (4 +0 +2 +0 +2 +0 -2 +2)/8 = 1$
 $S_6 \cdot C = (2 -2 +0 -2 +0 -2 -4 +0)/8 = -1$

(د)

شکل ۲-۴۵. (الف) توالی چیب باینری چهار ایستگاه. (ب) توالی چیب دو-علامتی همان ایستگاهها. (ج) شش نمونه از ارسال همزمان ایستگاهها. (د) استخراج سیگنال ایستگاه C.

در مثال سوم، ایستگاه A یک بیت 1 و ایستگاه B یک بیت 0 می فرستد، و بقیه ایستگاهها ساکت هستند. در مثال چهارم، ایستگاههای A و C یک بیت 1 و ایستگاه B یک بیت 0 می فرستد. در مثال پنجم، هر چهار ایستگاه یک بیت 1 می فرستند؛ و بالاخره در مثال آخر، ایستگاههای A ، B و D یک بیت 1 می فرستند و ایستگاه C یک بیت 0. توجه داشته باشید که در تمام مثالهای فوق، توالی های S_1 تا S_6 فقط یک بیت را نشان می دهند.

برای تشخیص و استخراج استریم بیت های یک ایستگاه باید توالی چپ آن ایستگاه را از قبل بدانیم. این کار را می توان با محاسبه ضرب داخلی نرمال شده توالی چپ دریافت شده (جمع خطی سیگنال تمام ایستگاهها در آن لحظه) با توالی چپ ایستگاه مورد نظر انجام داد. اگر توالی دریافت شده را S بنامیم، و بخواهیم توالی چپ (سیگنال ارسالی) ایستگاه C را از آن بیرون بکشیم، کافیت ضرب داخلی نرمال شده $S \cdot C$ را محاسبه کنیم.

برای اینکه ببینید این روش چگونه کار می کند، مثال چهارم شکل ۲-۴۵ (ج) را در نظر بگیرید. چیزی که گیرنده دریافت می کند، $S = A + \bar{B} + C$ است، و برای استخراج سیگنال C بایستی عبارت $S \cdot C$ را محاسبه کند:

$$S \cdot C = (A + \bar{B} + C) \cdot C = A \cdot C + \bar{B} \cdot C + C \cdot C = 0 + 0 + 1 = 1$$

همانطور که می بینید، دو جمله اول (بدلیل متعامد بودن بردارها) صفر شده اند. از اینجا می توانید علت انتخاب بردارهای متعامد را دریابید.

این وضعیت را می توان بصورت سه سیگنال مستقل نیز در نظر گرفت: سیگنالها بصورت مستقل و جدا از هم دریافت شده، و پس از محاسبه ضرب داخلی، با هم جمع می شوند. بعلمت متعامد بودن بردارها، تمام ضربهای داخلی (بجز $C \cdot C$) صفر خواهند شد. عبارت دیگر، تقدم ضرب داخلی یا جمع تأثیری بر نتیجه نهایی ندارد.

برای درک بهتر روش از رمز خارج کردن سیگنالها به شکل ۲-۴۵ (د) نگاه کنید. فرض کنید گیرنده می خواهد از شش سیگنال S_1 تا S_6 بیت های ارسال شده از ایستگاه C را استخراج کند. برای این کار، گیرنده تک تک سیگنالهای S را در بردار C (شکل ۲-۴۵ ب) ضرب کرده، و سپس $1/8$ آنرا محاسبه می کند (چون، $m = 8$). همانطور که می بینید، گیرنده توانسته است بطور صحیح بیت های ارسالی از C را استخراج کند.

در یک سیستم ایده آل (بدون نویز) CDMA تعداد ایستگاهها را می توان به تعداد دلخواه زیاد کرد (همانگونه که در یک کانال بدون نویز نایکونیست می توان نرخ نمونه برداری را بدلدخواه افزایش داد). اما در عمل، محدودیتهای فیزیکی ظرفیت سیستم را بشدت پائین می آورند. اول اینکه، فرض کردیم تمام چپها از نظر زمانی سنکرون هستند، در حالیکه این وضعیت عملاً غیر ممکن است. بهترین کاری که می توان کرد اینست که فرستنده با ارسال یک توالی چپ از پیش تعریف شده (که باندازه کافی طولانی است) به گیرنده امکان دهد تا خود را با فرستنده سنکرون کند. در این حالت تمام سیگنالهای غیر سنکرون بعنوان نویز تلقی خواهند شد. اگر تعداد این سیگنالهای غیر سنکرون چندان زیاد نباشد، الگوریتم فوق همچنان بخوبی کار خواهد کرد. در زمینه بر هم نهی (superposition) توالی های چپ با سطح نویز تحقیقات تئوریک مفصلی انجام شده است (Pickholtz et al., 1982). همانطور که می توان انتظار داشت، هر چه توالی چپ طولانی تر باشد، احتمال استخراج آن در محیط های پرنویز بیشتر خواهد بود. حتی می توان در توالی بیت ها از گدهای تصحیح خطا نیز استفاده کرد - البته در توالی چپ هرگز از گدهای تصحیح خطا استفاده نمی شود.

یکی دیگر از مفروضات ضمنی بحث فوق یکسان بودن قدرت سیگنالها نیست که به گیرنده می رسند. در سیستمهای تلفن همراه (و از جمله CDMA) فاصله تلفنهای همراه از ایستگاه مرکزی (و در نتیجه قدرت تشعشعی آنها) متغیر است، و قدرت سیگنالهایی که از تلفنها به ایستگاه مرکزی می رسد یکسان نیست. یکی از روشهای ابتکاری برای حل این مشکل آنست که تلفنهای همراه با کاهش سطح سیگنال دریافتی از ایستگاه مرکزی، توان تشعشعی خود را بالا ببرند؛ بعبارت دیگر، هر چه از ایستگاه مرکزی دور می شوند، سیگنال قویتری بفرستند.

ایستگاه مرکزی نیز می‌تواند صریحاً به تلفنهای همراه فرمان دهد تا (بسته به فاصله‌شان از ایستگاه) قدرت سیگنالهای خود را افزایش یا کاهش دهند.

همچنین فرض کردیم که گیرنده از هویت فرستنده آگاه است. از نظر تنوری این امکان هست که (با فرض وجود قدرت محاسباتی)، یک گیرنده به تمام سیگنالها گوش دهد و الگوریتمهای از رمز خارج کردن را روی تمام آنها اجرا کند. اما مثل معروفی هست که می‌گوید، «حرف زدن همیشه از عمل کردن ساده‌تر است». CDMA پیچیدگیهای دیگری نیز دارد، که در این بحث مختصر آنها را نادیده گرفتیم. با این حال، CDMA یکی از هوشمندانه‌ترین سیستمهای مخابرات بیسیم محسوب می‌شود، که سرعت در حال گسترش است. این سیستم معمولاً از یک باند 1.25 MHz (برخلاف 30 kHz در D-AMPS، و 200 kHz در GSM) استفاده می‌کند، و تعداد کاربرانی که پشتیبانی می‌کند، از هر دو سیستم قبلی بیشتر است. در عمل، پهنای باند موجود برای هر کاربر در سیستمهای CDMA حتی از GSM هم بهتر است.

یکی از بهترین منابع موجود در این زمینه (Lee and Miller, 1998) است. روشهای دیگری نیز در (Crespo et al., 1995) و (Sari et al., 2000) تشریح شده‌اند، که البته فهم آنها به دانش زیادی در زمینه مهندسی مخابرات نیاز دارد.

۳-۶-۲ تلفن‌های همراه نسل سوم: صدای دیجیتال و داده

آینده تلفن همراه چیست؟ اجازه دهید نگاه سریعی به آن بیندازیم. صنعت تلفن همراه عوامل محرک و پیشبرنده متعددی دارد. اول، ترافیک داده مدتهاست از ترافیک صدا پیشی گرفته و همچنان به رشد تصاعدی خود ادامه می‌دهد، ولی ترافیک صدا سالهاست دیگر رشد چندانی را تجربه نمی‌کند. بسیاری از متخصصان پیش‌بینی می‌کنند که بزودی در تلفنهای همراه نیز شاهد پیشی گرفتن ترافیک داده از ترافیک صدا خواهیم بود. دوم، صنایع تلفن، تفریحات و کامپیوتر همگی دیجیتالی شده‌اند و روز بروز بیشتر به یکدیگر نزدیک می‌شوند. مردم مدتهاست با شور و هیجان از دستگاه کوچک، سبک و قابل حملی صحبت می‌کنند که بتواند بعنوان تلفن، پخش CD، پخش DVD، ترینال ایمیل و وب، وسیله بازی و سرگرمی، واژه‌پرداز (و غیره و غیره) کار کند، و قادر باشد در هر نقطه‌ای بصورت بیسیم و با پهنای باند بالا به اینترنت متصل شود. این دستگاه رؤیایی همان تلفن همراه نسل سوم است؛ برای اطلاعات بیشتر (Huber et al., 2000; 2000) and Sarikaya را ببینید.

در سال ۱۹۹۲، ITU کوشید این رؤیا را کمی بیشتر به واقعیت نزدیک کند، و به همین منظور طرح اولیه‌ای بنام IMT-2000 (که IMT مخفف سیستم مخابرات تلفن همراه بین‌المللی - International Mobile Telecommunication - است) منتشر کرد. عدد 2000 سه چیز را نشان می‌داد: (۱) سالی که این سیستم بایستی عملیاتی شود، (۲) فرکانسی (بر حسب MHz) که سیستم تحت آن کار می‌کند، و (۳) پهنای باند این سرویس (بر حسب kHz).

البته IMT-2000 به هیچیک از این اهداف نرسید. در سال ۲۰۰۰ هیچ سیستمی نصب و راه‌اندازی نشد. ITU پیشنهاد کرده بود که دولتها طیف 2 GHz را برای این منظور کنار بگذارند تا این سرویس بتواند بین کشورهای مختلف بدون اشکال کار کند؛ تنها کشوری که به این توصیه عمل کرد، چین بود. و بالاخره، مشخص شد که در حال حاضر پهنای باند 2 Mbps برای کاربرانی که بیش از حد متحرک هستند، عملی نیست (علت آن هم دشواری پاس‌کاری این قبیل کاربران است). پهنای باند عملی‌تر عبارتست از 2 Mbps برای کاربران ثابت خانگی (که می‌تواند با ADSL رقابت کند)، 384 kbps برای کاربرانی که قدم می‌زنند، و 144 kbps برای آنهایی که سوار اتومبیل هستند. با این حال، حوزه فعالیت 3G (نسل سوم) بسیار پُر جنب و جوش است. نسل سوم شاید کمی دیرتر از راه برسد و کمی کمتر از آنچه که انتظار می‌رود باشد، ولی حتماً می‌آید.

سرویسهایی که قرار است شبکه IMT-2000 در اختیار کاربران خود بگذارد، عبارتند از:

۱. انتقال صدا با کیفیت عالی.
۲. پیام رسانی (سرویس که جایگزین ایمیل، فکس، SMS، chat و غیره خواهد شد).
۳. مالتی مدیا (پخش موسیقی، تماشای ویدئو، فیلم، تلویزیون و غیره).
۴. دسترسی اینترنت (گشت و گذار در وب، از جمله صفحاتی که صدا، تصویر و فیلم دارند)

سرویسهای دیگری از قبیل کنفرانس ویدئویی (video conferencing)، حضور از راه دور (telepresence)، بازیهای گروهی و تجارت-همراه (m-commerce: پرداخت بهای اجناس خریداری شده در فروشگاه با نزدیک کردن تلفن همراه به صندوق) را نیز می توان از این شبکه انتظار داشت. علاوه بر آن، تمام این سرویسها بین المللی خواهند بود (یعنی در تمام نقاط دنیا می توان به آنها دسترسی داشت: در جاهایی که خطوط ارتباطی زمینی وجود ندارد، تلفن همراه بطور خودکار از لینکهای ماهواره ای استفاده خواهد کرد)، و همیشه با کیفیت تضمین شده در دسترس هستند.

ITU برای IMT-2000 تکنولوژی واحدی را در نظر گرفته است، بگونه ای که تلفنهای همراه بتوانند در هر نقطه ای از دنیا کار کنند (مانند دستگاههای ضبط صوت و پخش CD، نه تلویزیونها و تلفنهای همراه امروزی). یکسان شدن تکنولوژی، علاوه بر تسهیل کار شرکتهای مخابرات، افراد بیشتری را به استفاده از این سرویسها تشویق خواهد کرد. جنگ تکنولوژیها (مانند آنچه بین ویدئوهای Betamax و VHS رخ داد) هیچگاه به نفع صنعت و تجارت نبوده است.

پیشنادهای متعددی ارائه شد، که بعد از غربال شدن آنها، سرانجام دو طرح باقی ماند. طرح اول، بنام W-CDMA (CDMA پهن باند - Wideband CDMA)، از طرف شرکت سوئدی اریکسون ارائه شد. این طرح از توالی مستقیم با طیف گسترده (مانند آنچه در بالا گفتیم) استفاده می کند. این سیستم در یک باند 5-MHz کار می کند، و بگونه ای طراحی شده که بتواند با شبکه های GSM (البته نه GSM قدیمی) کار کند. تلفنهای این سیستم می توانند بدون اختلال در ارتباط یک سلول W-CDMA را ترک کرده و وارد یک سلول GSM شوند. اتحادیه اروپا بشدت از این سیستم پشتیبانی می کند، و به آن نام UMTS (سیستم مخابرات تلفن همراه جهانی - Universal Mobile Telecommunications System) را داده است.

طرح دیگر که از طرف شرکت Qualcomm (مخترع CDMA) پیشنهاد شده، CDMA2000 نام دارد. این سیستم نیز اساساً همان توالی مستقیم با طیف گسترده (شکل اصلاح شده استاندارد IS-95) است، که با آن سازگاری دارد. CDMA2000 نیز از یک باند 5-MHz استفاده می کند، ولی نمی تواند با سلولهای GSM (و طبیعتاً، با سلولهای D-AMPS) پاس کاری انجام دهد. CDMA2000 تفاوتهای دیگری، از قبیل نرخ چیب، زمان فریم، طیف فرکانسی، و روش سنکرون شدن، نیز با W-CDMA دارد.

اگر مهندسان اریکسون و Qualcomm را در اتاقی حبس کنند و از آنها بخواهند یک سیستم مشترک طراحی کنند، به احتمال زیاد می توانند. بهر حال، هر دو سیستم از تکنیکهای CDMA و یک کانال 5-MHz استفاده می کنند، و چیزهای دیگر هم مسلماً ارزش خودکشی ندارند. در اینجا هم (مثل همیشه) مشکل نه فنی و مهندسی بلکه سیاسی است. اروپا سیستمی می خواهد که با GSM سازگاری داشته باشد، و ایالات متحده آمریکا دنبال سیستمی است که با IS-95 سازگار باشد. هر دو نیز از شرکتهای محلی خود دفاع می کنند: اروپا از اریکسون (که شرکتی سوئدی است) و آمریکا از Qualcomm (که در کالیفرنیا است). اریکسون و Qualcomm دعوای حقوقی بیشماری نیز بر سر حق اختراع CDMA داشته اند.

بالاخره در مارس ۱۹۹۹ اریکسون Qualcomm را خرید، و دعوای خاتمه یافت. آنها بر سر یک استاندارد 3G نیز به توافق رسیدند، ولی این استاندارد ناسازگاری های زیادی داشت. اما این منازعات بالاخره به پایان می رسد، و بزودی شاهد تلفن ها و سرویس های 3G خواهیم بود.

درباره سیستم های 3G مطالب زیادی نوشته شده، و برخی آنرا بعنوان یکی از بزرگترین اختراعات بشری ستوده اند. برای نمونه به: (Collins and Smith, 2000; De Vriendt et al., 2002; Harte et al., 2002; 2002; Lu, and Sarikaya, 2000) اشاره می کنیم. اما 3G متقدانی نیز دارد که معتقدند اساساً راه را اشتباه می رود، و می توان از میان آنها از (Garber, 2002; and Goodman, 2000) نام برد.

عده ای هم که از جنگ 3G خسته شده اند، به فکر افتادند تا قدم کوچکی به سمت آن بردارند، و نام آنرا هم 2.5G گذاشتند (اگر چه شاید نام 2.1G برای آن مناسب تر است). یکی از این سیستم ها EDGE (نسخه داده بهبود یافته برای تکامل GSM - Enhanced Data rates for GSM Evolution) نام دارد، که اساساً چیزی نیست جز همان GSM با bits/ baud بیشتر. مشکل اینجا است که bits/ baud بیشتر یعنی خطای بیشتر، و به همین دلیل EDGE در سرعت های متفاوت از ۹ روش مختلف برای مدولاسیون و تصحیح خطا استفاده می کند.

یکی دیگر از طرح های 2.5G، GPRS (سرویس عمومی بسته رادیویی - General Packet Radio Service) نام دارد، که عبارتست از یک شبکه سوئیچینگ بسته ای که روی GSM یا D-AMPS کار می کند. این سیستم اجازه می دهد تا تلفن های همراه در سلول های صوتی بسته های IP رد و بدل کنند. GPRS برای این منظور از بُرش های زمانی و فرکانس های اختصاصی استفاده می کند، که تعداد و محل آنها (به نسبت ترافیک صوت و داده در سلول) بطور دینامیک توسط ایستگاه مرکزی تعیین می شود.

بُرش های زمانی موجود به چندین کانال منطقی تقسیم شده، و به مصارف مختلف می رسند. تخصیص بُرش های زمانی به هر کانال توسط ایستگاه مرکزی صورت می گیرد. یکی از این کانال های منطقی برای ارسال بسته ها از ایستگاه مرکزی به تلفن های همراه است، و هر بسته می داند که مقصد آن کجاست. برای ارسال یک بسته IP، تلفن همراه درخواستی برای تخصیص یک یا چند بُرش زمانی به ایستگاه مرکزی می فرستد. اگر این درخواست بدون مشکل به ایستگاه مرکزی برسد، ایستگاه مرکزی فرکانس و بُرش های زمانی تخصیص یافته را به تلفن همراه اعلام می کند. همین که بسته های IP ارسالی از تلفن همراه به ایستگاه مرکزی رسید، ایستگاه مرکزی (از طریق ارتباطاتی که دارد) آنها را به اینترنت منتقل می کند. از آنجائیکه GPRS فقط لایه ایست روی سیستم های صوتی موجود، می توان آنرا در بهترین حالت یک قدم به سمت 3G بشمار آورد.

با اینکه شبکه های 3G هنوز بطور کامل راه اندازی نشده اند، برخی از محققان آنرا موضوعی تحقق یافته (که دیگر ارزش توجه ندارد) تلقی کرده و سراغ سیستم های نسل چهارم (4G) رفته اند (Berezdivin et al., 2002; Guo and Chaskar, 2002; Huang and Zhuang, 2002; Kellerer et al., 2002; and Misra et al., 2002). برخی از مشخصات پیشنهادی سیستم های 4G عبارتند از: پهنای باند زیاد، دسترسی در همه جا، یکپارچگی کامل با شبکه های کابلی (و بویژه شبکه های IP)، مدیریت تطبیقی منابع و طیف فرکانسی، رادیوی نرم افزاری، و سرویس های مالتی مدیا با کیفیت عالی.

از سوی دیگر، با توجه به تعداد بسیار زیاد شبکه های محلی بیسیم 802.11 که امروزه نصب شده، برخی معتقدند 3G به دنیا نیامده، مرده است. اینها می گویند، «هر جا که بروید بالاخره تحت پوشش یک شبکه 802.11 هستید، و این همان چیز است که 3G می خواهد با آن همه منت به شما بدهد.» پنج سال دیگر معلوم می شود حق با چه کسی بوده است.

۷-۲ تلویزیون کابلی

تا اینجا سیستمهای تلفن ثابت و بیسیم را بررسی کردیم. هر دوی این سیستمها مسلماً نقش مهمی در آینده شبکه بازی خواهند کرد. اما بازیگر دیگری نیز در زمینه شبکه های ثابت وارد صحنه شده، و هر روز اهمیت بیشتری می یابد: **تلویزیون کابلی (Cable TV)**. امروزه افراد بسیاری سرویسهای تلویزیون و اینترنت خود را از طریق کابل دریافت می کنند، و شرکت های تلویزیون کابلی نیز با جدیت بدنبال سهم بیشتری از بازار هستند. در این قسمت تلویزیون کابلی را از دیدگاه شبکه مورد بررسی دقیقتر قرار خواهیم داد. برای کسب اطلاعات بیشتر نیز می توانید به (Laubach et al., 2001; Louis, 2002; Ovadia, 2001; and Smith, 2002) مراجعه کنید.

۱-۷-۲ تلویزیون با آنتن مرکزی

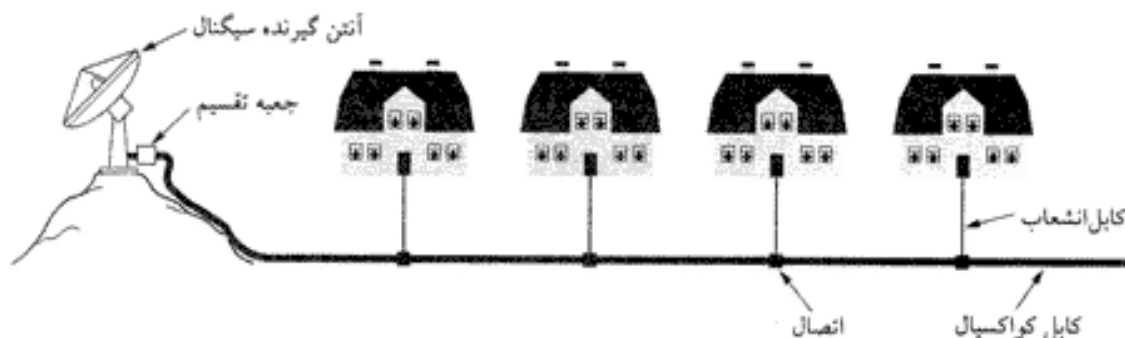
تلویزیون کابلی در اواخر دهه ۱۹۴۰ بعنوان راهی برای ارائه سرویسهای بهتر به مناطق روستایی و کوهستانی ابداع شد. این سیستم عبارت بود از یک آنتن بزرگ بر فراز نقطه ای مرتفع (برای دریافت امواج تلویزیونی)، یک تقویت کننده (موسوم به **جعبه تقسیم - head end**) برای تقویت سیگنال، و یک رشته کابل کواکسیال برای انتقال سیگنال تلویزیون به منازل؛ شکل ۲-۴۶ را ببینید.

در آن سالها به این سیستم **تلویزیون با آنتن مرکزی (Community Antenna Television)** گفته می شد. این سیستم بقدری ساده بود که هر کسی با کمترین تجربه سیمکشی نیز می توانست آنرا راه بیندازد، و هزینه آن هم معمولاً بین مشترکین سرشکن می شد. با اضافه شدن مصرف کنندگان فقط کافی بود انشعابهای بیشتری از کابل اصلی گرفته (و احیاناً تقویت کننده های بیشتری نصب) شود. این سیستم اساساً یکطرفه (از جعبه تقسیم سیگنال به مشترکین) بود، و تا دهه ۱۹۷۰ هزاران نمونه از آن نصب شده بود.

در سال ۱۹۷۴ شرکت رسانه ای تایم کانال جدیدی بنام **Home Box Office** راه اندازی کرد که فیلمهای سینمایی پخش می کرد، و فقط از طریق کابل قابل دسترسی بود. کانالهای کابلی دیگر نیز سرعت راه افتادند که اخبار، مسابقات ورزشی، آشپزی (و بسیاری موضوعات دیگر) پخش می کردند. این تحول باعث دو تغییر اساسی در صنعت تلویزیون شد. اول اینکه، شرکت های بزرگ شروع به خریدن شبکه های کابلی موجود در شهرها کردند، و در بسیاری جاها نیز خود رأساً شروع به کابل کشی و جذب مشترکین جدید کردند. دوم اینکه، برای ارائه سرویس در سراسر کشور لازم بود بین شهرها نیز کابل کشی شود، پس شرکت های مزبور شروع به متصل کردن شهرها به یکدیگر کردند. این وضعیت درست مانند شبکه تلفن در ۸۰ سال قبل بود، که شرکت های تلفن برای ارائه سرویسهای راه دور بین شهرها کابل تلفن کشیدند.

۲-۷-۲ اینترنت کابلی

در طول سالها سیستمهای کابلی رشد کرد، و بین شهرهای مختلف فیبرهای نوری با پهنای باند زیاد کشیده شد. این

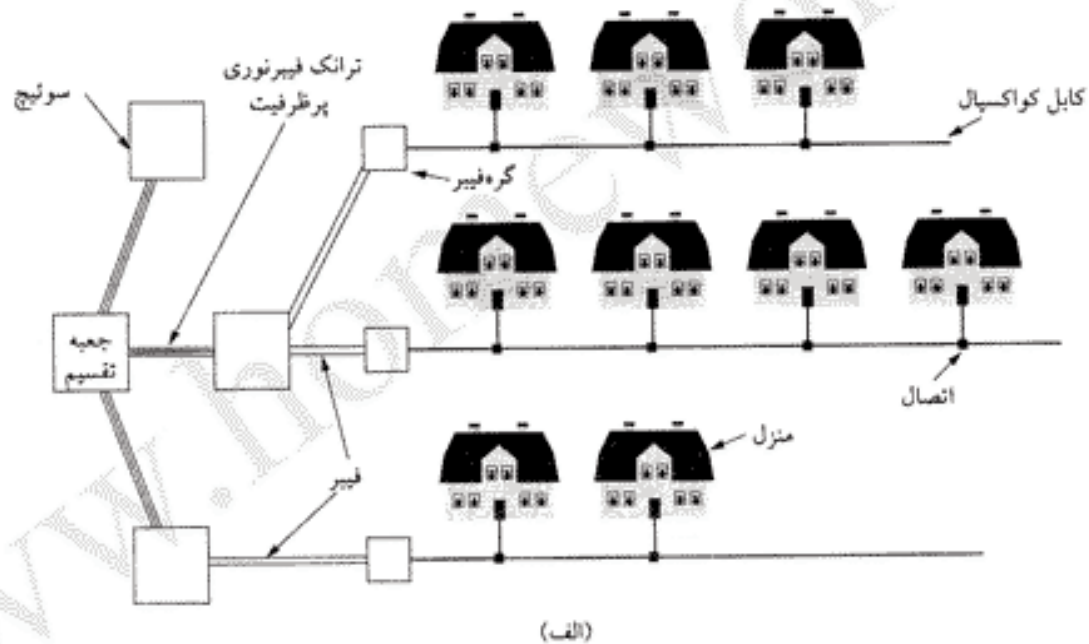


شکل ۲-۴۶. یک سیستم تلویزیون کابلی اولیه.

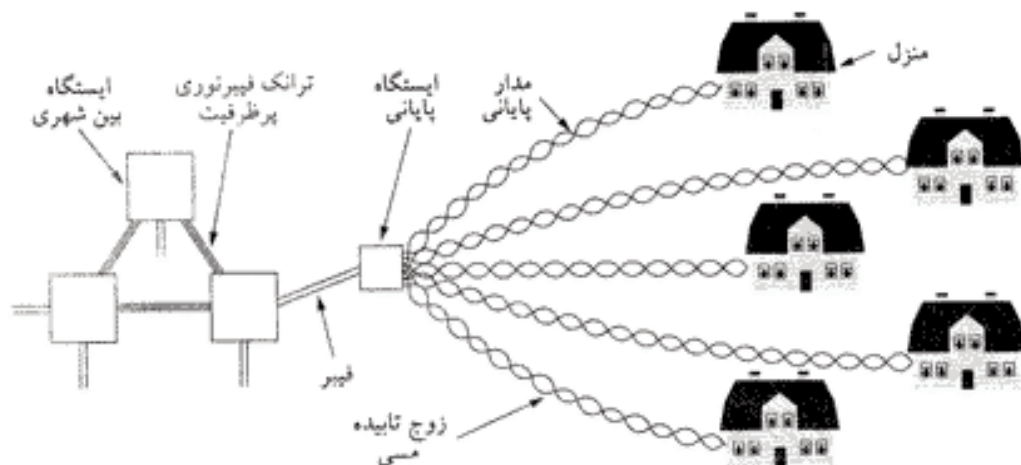
سیستم که ترکیبی بود از فیبر نوری (بین شهرها) و کابلهای کواکسیال (از مراکز توزیع تا منازل) HFC (آمیخته فیبر-کواکس - Hybrid Fiber Coax) نام دارد. در نقاطی که گره فیبر (fiber node) خوانده می‌شوند، سیگنالهای نوری به الکترونیکی (و بالعکس) تبدیل می‌شود. از آنجائیکه پهنای باند فیبر نوری بسیار بیشتر از کواکس است، یک گره فیبر می‌تواند تعداد زیادی کابل کواکسیال را تغذیه کند. در شکل ۲-۴۷ (الف) یک سیستم HFC مدرن را ملاحظه می‌کنید.

در سالهای اخیر، بسیاری از شرکتهای کابلی تصمیم گرفته‌اند وارد تجارت دسترسی اینترنت (و همچنین تلفن) شوند. با این حال تفاوت‌های فنی شبکه‌های کابلی و تلفن چنان است که تاکنون مانع از تحقق کامل این خواسته شده است. یکی از مشکلاتی که می‌توان بعنوان نمونه به آن اشاره کرد، تقویت‌کننده‌های یکطرفه‌ای است که باید با تقویت‌کننده‌های دوطرفه جایگزین شوند.

اما تفاوت مهمتری بین سیستم HFC شکل ۲-۴۷ (الف) و سیستم تلفن (شکل ۲-۴۷ ب) وجود دارد، که برطرف کردن آن بسیار مشکلتر است. در سیستم HFC یک رشته کابل کواکسیال بین خانه‌های متعددی مشترک



(الف)



(ب)

شکل ۲-۴۷. (الف) تلویزیون کابلی، (ب) سیستم تلفن ثابت.

است، در حالیکه در سیستم تلفن شهری هر خانه دارای یک اتصال (مدار پایانی) خاص خود است. در پخش برنامه های تلویزیونی این مشترک بودن کابل اهمیت چندانی ندارد: یک برنامه پخش می شود، و اهمیتی ندارد که ۱۰ نفر آنرا می بینند یا ۱۰,۰۰۰ نفر. اما وقتی پای دسترسی اینترنت به میان می آید، ۱۰ نفر با ۱۰,۰۰۰ نفر فرق بسیاری با هم دارند. اگر یکی از کاربران بخواهد فایل بزرگی را از اینترنت بگیرد، پهنای باند لازم برای این کار از کاربران دیگر گرفته می شود. هر چه تعداد کاربران بیشتر شود، رقابت بر سر پهنای باند شدیدتر خواهد بود. در سیستم تلفن چنین مشکلی وجود ندارد: اگر شما مشغول گرفتن یک فایل بزرگ روی خط ADSL خود باشید، تأثیری روی کار همسایه تان نخواهد گذاشت. از طرف دیگر، پهنای باند یک کابل کواکس بسیار بیشتر از زوج-تابیده است.

شرکتهای کابلی برای غلبه بر این مشکل، کابل های بلند را تکه تکه کرده و هر کدام را مستقیماً به یک گروه فیبر متصل می کنند. اگر تعداد کاربران هر کابل کواکس خیلی زیاد نباشد، پهنای باند فیبر نوری عملاً نامحدود خواهد بود و می توان ترافیک را بخوبی مدیریت کرد. امروزه هر کابل کواکس به ۵۰۰ تا ۲۰۰۰ نفر سرویس می دهد، ولی با زیاد شدن تعداد مشترکین این سیستم شاید لازم باشد از کابلها و گروه های فیبر بیشتری استفاده کرد.

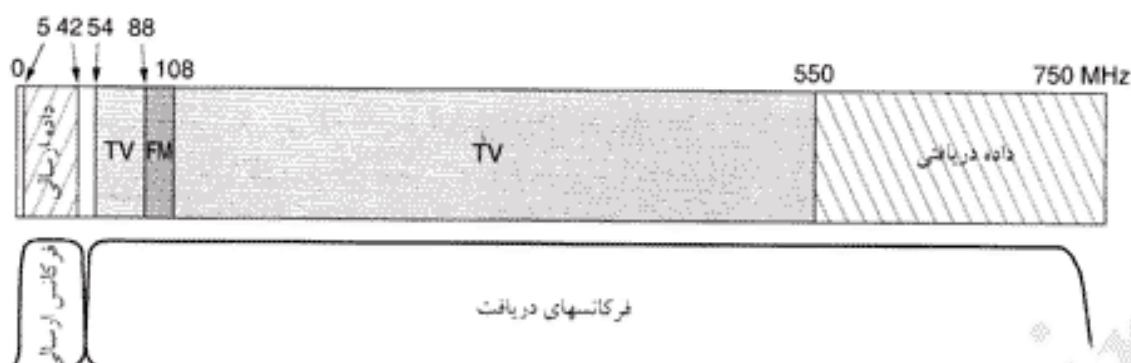
۳-۷-۲ تخصیص طیف فرکانسی

حذف تمام کانالهای تلویزیونی و یکارگیری زیرساخت های کابلی برای دسترسی اینترنت باعث نارضایتی تعداد زیادی از مشترکین خواهد شد، و به همین دلیل شرکتهای کابلی در این کار تردید دارند. علاوه بر آن، در اغلب شهرها قوانین سختگیرانه ای برای کنترل آنچه روی کابلها فرستاده می شود، وجود دارد، و حتی اگر شرکتهای کابلی واقعاً هم بخواهند اجازه چنین کاری را ندارند. در نتیجه، آنها باید راهی برای همزیستی تلویزیون کابلی و اینترنت روی یک کابل می یافتند.

کانالهای تلویزیون کابلی در منطقه آمریکای شمالی در ناحیه 54-550 MHz طیف قرار دارند (البته باستانی محدود 88-108 MHz که به رادیوی FM اختصاص دارد). هر کانال (با احتساب باند های محافظ) 6 MHz پهنای دارد. در اروپا، حد پائین طیف معمولاً 65 MHz است و کانالها نیز 6-8 MHz پهنای دارند، چون سیستمهای تلویزیونی PAL و SECAM کیفیت بالاتری دارند. از قسمت پائین این باند استفاده نمی شود. کابل های جدید می توانند بالاتر از 550 MHz (اغلب تا 750 MHz) نیز کار کنند. راه حل انتخاب شده این بود که کانالهای ارسال به اینترنت (upstream) در باند 5-42 MHz (در اروپا، کمی بالاتر) و کانالهای دریافت از اینترنت (downstream) در فرکانسهای بالاتر تعریف شوند. در شکل ۲-۴۸ طیف فرکانسی کابل کواکسیال را ملاحظه می کنید.

از آنجائیکه سیگنالهای تلویزیونی فقط در یک جهت (مرکز توزیع به منازل) منتشر می شوند، تقویت کننده های ارسال فقط در باند 5-42 MHz، و تقویت کننده های دریافت فقط در باند بالاتر از 54 MHz کار می کنند. همانطور که می بینید، بین باندهای ارسال و دریافت یک عدم تقارن بوجود آمده، و باند دریافت از اینترنت بسیار وسیعتر است (البته این وضعیت چندان هم بد نیست، چون اغلب ترافیک اینترنت در همین جهت است). قبلاً هم دیدید که شرکتهای تلفن عمداً (و بدون اینکه اجبار تکنیکی وجود داشته باشد) سرویس DSL را بصورت نامتقارن در می آورند.

توانایی کابل های بلند کواکسیال در انتقال سیگنالهای دیجیتال چندان بهتر از سیمهای زوج-تابیده نیست، و به همین دلیل آنها هم به نوعی مدولاسیون آنالوگ احتیاج دارند. برای مدولاسیون هر کانال 6 MHz (یا 8 MHz) از روش QAM-64 (و اگر کیفیت کابل فوق العاده خوب باشد، از QAM-256) استفاده می شود. با یک کانال 6-MHz و مدولاسیون QAM-64، نرخ انتقال داده ای معادل 36 Mbps بدست می آید، که اگر سرآیند از آن کسر شود، تقریباً 27 Mbps باقی می ماند. با مدولاسیون QAM-256 نرخ انتقال داده (بعد از کسر سرآیند) 39 Mbps



شکل ۲-۴۸. تخصیص فرکانس در یک سیستم تلویزیون کابلی برای دسترسی اینترنت.

خواهد بود. (ظرفیت سیستمهای اروپایی 1/3 بیشتر است.) در کانالهای ارسال به اینترنت بدلیل وجود نویز بالا (از منابع مایکروویو زمینی، و رادیوهای محلی) حتی QAM-64 نیز بخوبی کار نمی کند، و باید از روشهای محافظه کارانه تری (مانند QPSK) استفاده کرد. روش QPSK ظرفیت بسیار کمتری نسبت به مدولاسیون QAM دارد (شکل ۲-۲۵ را ببینید) - 2 bits/ baud بجای 6-8 bits/ baud. این اوصاف، عدم تقارن کانالهای ارسال و دریافت حتی از آنچه در شکل ۲-۴۸ دیده می شود، نیز بیشتر است. شرکت های کابلی مجبور شدند تقویت کننده های ساده خود را نیز با سیستم های دیجیتالی هوشمند جایگزین کنند. نام این تجهیزات نیز از جعبه تقسیم (headend) به CTMS (سیستم پایانه ای مودم کابلی - Cable Modem Termination System) تغییر کرده است، ولی ما در قسمتهای آتی همچنان از اصطلاح «جعبه تقسیم» برای اشاره به این تقویت کننده ها استفاده خواهیم کرد.

۲-۷ مودمهای کابلی

مودم کابلی (cable modem) دستگاهیست با دو سر: یک سر به کامپیوتر وصل می شود، و سر دیگر به شبکه کابلی. در سالهای اولیه اینترنت کابلی، هر شرکت مودم کابلی خاصی داشت، که توسط تکنسینهای آن نصب می شد. اما یزودی معلوم شد که وجود سیستمی با استاندارد باز موجب گسترش رقابت در بازار، کاهش قیمتها، و در نتیجه اقبال عمومی به این سرویسها خواهد شد. علاوه بر آن، خریدن یک مودم از فروشگاه و نصب آن توسط خود مشتری بسیار ساده تر از مراجعه یک تکنسین برای نصب هر مودم است (که هزینه بسیار بالایی نیز دارد). در نتیجه، شرکت های کابلی بزرگ برای تولید یک مودم کابلی استاندارد (و تست سازگاری آنها) با شرکتی بنام CableLabs متحد شدند. این استاندارد که DOCSIS (Data Over Cable Service Interface Specification) نام داشت، فقط نقطه شروعی بود برای جایگزینی مودمهای متنوع و ناسازگار - ویرایش اروپایی این استاندارد نیز EuroDOCSIS نام گرفت. اما ایده استاندارد کردن مودمهای کابلی برای همه شرکت های کابلی جالب نبود، چون آنها پول خوبی بابت اجازه دادن مودم به علاقمندان این سرویسها به جیب می زدند. یک استاندارد باز پای دهها تولیدکننده جدید را به این قلمرو باز می کرد، و به این تجارت پُر منفعت پایان می داد. ارتباط مودم به کامپیوتر ساده و سراسر است: در حال حاضر از اینترنت 10-Mbps (و گاهی هم USB) برای این منظور استفاده می شود. بطور مسلم در آینده شاهد کارتهای مودم کابلی (شبه کارتهای V.9x فعلی) نیز خواهیم بود.

ارتباط در سر دیگر پیچیده تر است، و بخش عمده ای از استاندارد آن به مهندسی رادیو مرتبط می شود (که از

حاصله این کتاب خارج است). تنها نکته ای که باید پیاد داشته باشید این است که، یک مودم کابلی (مانند مودمهای ADSL) همیشه روشن و متصل است. ارتباط این مودمها به محض روشن شدن برقرار شده و تا زمان خاموش شدن در همین حالت باقی می ماند (هزینه آنها نیز بر حسب زمان محاسبه نمی شود).

برای درک بهتر طرز کار مودمهای کابلی، اجازه دهید ببینیم با روشن کردن آنها چه اتفاقی می افتد. مودم به محض روشن شدن تمام کانالهای دریافت را برای پیدا کردن بسته خاصی که در فواصل منظم از طرف منبع ارسال می شود (و حاوی پارامترهای سیستم برای مودمهایی که تازه روشن شده اند، می باشد) جستجو می کند. پس از پیدا کردن این بسته، مودم جدید حضور خود را از طریق یکی از کانالهای ارسال به منبع اعلام می کند. منبع نیز با تخصیص کانالهای ارسال و دریافت پاسخ مودم را می دهد. البته این کانالها هر زمان که منبع لازم ببیند (مثلاً، برای متعادل کردن بار)، می توانند عوض شوند.

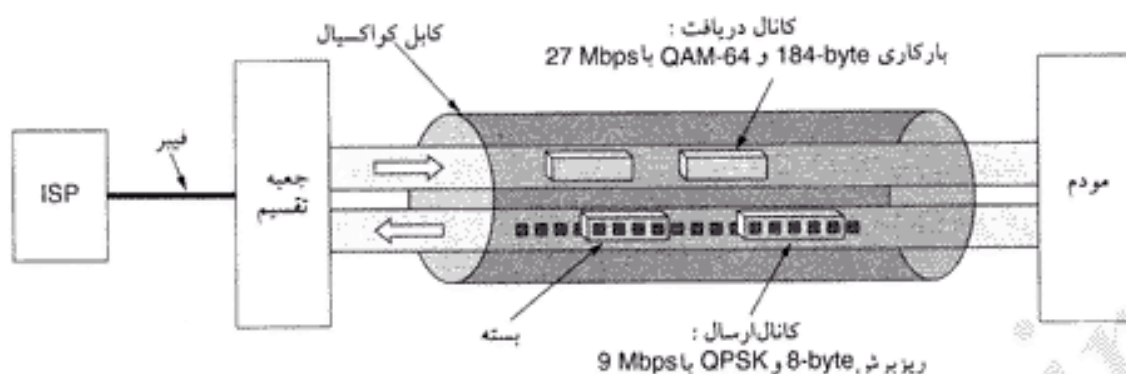
پس مودم با ارسال یک بسته خاص به منبع و محاسبه زمان رفت و برگشت آن، فاصله خود با منبع را تعیین می کند؛ به این کار تعیین فاصله (ranging) گفته می شود. این کار برای کارکرد صحیح کانالهای ارسال و همزمانی لازم است. این فاصله زمانی به ریزبُرشها (minislots) تقسیم می شود؛ هر بسته باید دقیقاً در یک یا چند ریزبُرش متوالی جا شود. منبع در فواصل منظم دور جدیدی از ریزبُرشها را اعلام می کند، ولی (بدلیل یکسان نبودن فاصله مودمها از منبع) این شلیک شروع مسابقه همزمان در تمام مودمها شنیده نخواهد شد. هر مودم، با دانستن فاصله خود از منبع، می تواند بفهمد که اولین ریزبُرش واقعاً در چه زمانی شروع شده است. طول ریزبُرشها به شبکه بستگی دارد (و ظرفیت آن معمولاً معادل ۸ بایت است).

در حین آماده سازی اولیه، منبع به هر مودم یک ریزبُرش تخصیص می دهد که می تواند از آن برای درخواست بهنای پاند ارسال استفاده کند. قاعدتاً تعداد مودمها از ریزبُرشها بیشتر است و یک ریزبُرش به چند مودم اختصاص داده می شود، که این می تواند موجب کشمکش بین آنها شود. وقتی کامپیوتر اطلاعاتی را برای ارسال به مودم می فرستد، مودم تعداد ریزبُرشهای لازم را محاسبه کرده و درخواست تخصیص آنها را از منبع می کند. اگر این درخواست پذیرفته شود، منبع ریزبُرشهای تخصیص داده شده را (روی یک از کانالهای دریافت) به مودم اعلام می کند. مودم نیز در ریزبُرشهای اختصاص یافته اطلاعات خود را ارسال می کند (و اگر به ریزبُرشهای بیشتری نیاز داشت، می تواند آنها را از طریق یکی از فیلدهای سرآیند درخواست کند).

اگر جوابی از منبع نرسید (که درخواست همزمان چند مودم می تواند یکی از علت های آن باشد)، مودم مدتی صبر کرده و دوباره اقدام خواهد کرد. اگر اقدام دوباره هم با شکست مواجه شد، مودم زمان انتظار را بیشتر می کند (پیاد دارید که این همان روش تخصیص بُرش زمانی در شبکه ALOHA است؛ از تکنیکهای اینترنت نمی توان استفاده کرد، چون مودم راهی برای گوش کردن به رسانه و تشخیص تصادم ندارد در فصل ۴ این مبحث را مفصلتر بررسی خواهیم کرد).

مدیریت کانالهای دریافت از اینترنت بکلی متفاوت است. اول اینکه، در اینجا فقط یک فرستنده وجود دارد (منبع)، و بالطبع کشمکشی هم در میان نیست و منبع می تواند از تکنیکهای مالتی پلکس تقسیم زمانی آماری استفاده کند. دیگر اینکه، ترافیک دریافت از اینترنت معمولاً بسیار بیشتر از ترافیک ارسال به اینترنت است، و به همین دلیل از بسته هایی با طول ثابت (۲۰۴۸ بایت) استفاده می شود. قسمتی از این بسته سرآیندهاست (کُد تصحیح خطای Reed-Solomon و مانند آن). و در نهایت ۱۸۴ بایت برای داده های کاربر باقی می ماند. این عدد برای سازگاری با تلویزیون دیجیتال MPEG-2 انتخاب شده، بنابراین فرمت کانالهای دریافت از اینترنت و تلویزیون یکسان است (شکل ۲-۴۹ را ببینید).

به فرآیند آماده سازی مودم برگردیم؛ همین که مودم فاصله خود را با منبع مشخص کرد، و کانالهای ارسال و



شکل ۲-۴۹. کانالهای ارسال و دریافت در منطقه آمریکای شمالی.

دریافت و ریزبرشها را گرفت، آماده است تا اطلاعات خود را بفرستد. اولین بسته‌ای که مودم می‌فرستد، بسته خاصیت حاوی درخواست یک آدرس IP از ISP با استفاده از پروتکل DHCP (که در فصل ۵ درباره آن توضیح خواهیم داد). همچنین وقت دقیق نیز از منبع پرسیده شد، و منبع به آن جواب می‌دهد. در قدم بعد ایمنی اطلاعات بایستی تضمین شود. از آنجائیکه کابل مودم بین تعداد زیادی از افراد مشترک است، هر کسی می‌تواند اطلاعات تمام کاربران دیگر را بخواند. برای جلوگیری از سرک کشیدن دیگران، ترافیک در هر دو جهت (ارسال و دریافت) رمز می‌شود. بخشی از فرآیند آماده‌سازی مودم شامل ایجاد کلیدهای رمز است. شاید در نگاه اول این کار غیرممکن بنظر برسد؛ چگونه می‌توان در روز روشن و زیر نگاه هزاران غریبه کلیدهای رمز را رد و بدل کرد؟ این کار با استفاده از الگوریتمی بنام دیفی-هلمان (Diffie-Hellman) ممکن است، اما اجازه دهید توضیحات بیشتر را به فصل ۸ موکول کنیم.

در پایان، مودم هویت منحصر بفرد (شامل نام کاربر و کلمه عبور) خود را روی کانالی که اکنون از امنیت کافی برخوردار است، به ISP اعلام می‌کند. در اینجا فرآیند آماده‌سازی به پایان می‌رسد، و کاربر می‌تواند به کار عادی خود ادامه دهد.

به همین توضیح مختصر درباره مودمهای کابلی بسنده می‌کنیم، ولی می‌توانید اطلاعات تکمیلی را در (Adams and Dulchinos, 2001; Donaldson and Jones, 2001; and Dutta-Roy, 2001) بیابید.

۵-۷-۲ مودم کابلی یا ADSL ؟

کدامیک بهتر است: ADSL یا مودم کابلی؟ این مانند آنست که پرسید کدام سیستم عامل، یا کدام زبان بهتر است. جواب به کسی که این سؤال را از او می‌پرسید، بستگی دارد. اما اجازه دهید ADSL و مودم کابلی را از چند نقطه نظر با هم مقایسه کنیم. هر دوی این سیستمها در بخش ستون فقرات از فیبر نوری استفاده می‌کنند، ولی بخش انتهایی آنها متفاوت است. مودم کابلی در بخش انتهایی (مدار پایانی) از کابل کواکسیال استفاده می‌کند، و ADSL از زوج-تابیده. از نظر تنوری، ظرفیت کواکس صدها برابر زوج-تابیده است. اما، تمام این ظرفیت در اختیار کاربر اینترنت نیست، چون پهنای باند کابل صرف چیزهای بدردنخوری (مثل برنامه‌های تلویزیونی) هم می‌شود.

در عمل، مقایسه ظرفیت این سیستمها بسیار مشکل است. در ADSL، شرکت سرویس دهنده ظرفیت خط را به صراحت مشخص می‌کند (مثلاً، 1 Mbps دریافت از اینترنت و 256 kbps ارسال به اینترنت)، و در اغلب موارد ۸۰٪ این ظرفیت نیز محقق می‌شود. اما شرکت‌های کابلی هرگز درباره ظرفیت سرویس خود صحبت نمی‌کنند، چون این ظرفیت به تعداد کاربرانی که در هر لحظه روی یک کابل هستند، بستگی دارد. گاهی کابل بهتر از ADSL است، و گاهی بدتر. اما چیزی را نمی‌توان از قبل پیش‌بینی کرد (و این از همه آزاردهنده‌تر است). در یک لحظه

کیفیت کابل خوب است، و لحظه بعد (وقتی یکی از آن خوره های اینترنت کامپیوتر خود را روشن می کند) غیر قابل تحمل.

در ADSL افزایش تعداد کاربران تأثیری روی کیفیت خط کاربران موجود نخواهد گذاشت، چون هر کاربر دارای یک خط مستقل است. در حالیکه در کابل این افزایش موجب افت کیفیت می شود. تنها راه مقابله با این مشکل هم انشعاب کابل های بیشتر از گره فیبر، و کم کردن تعداد کاربران هر خط است. اما این یعنی هزینه بیشتر، چیزی که صاحبان شرکت های کابلی در برابر آن مقاومت می کنند.

سیستم های تلفن همراه هم وضعیتی شبیه مودم کابلی دارد؛ گروهی از کاربران بطور مشترک از یک بخش خاص از پهنای باند استفاده می کنند. از آنجائیکه ترافیک صدا نسبتاً یکنواخت است، تقسیم پهنای باند بین کاربران (که با تکنیک های FDM و TDM انجام می شود) نیز یکنواخت و ثابت است. اما در ترافیک داده، تقسیم ثابت بهیچوجه کارایی ندارد، چون توزیع زمانی استفاده هر کاربر از کانال اختصاصی (روشی که در مودم کابلی بکار می رود) یکنواخت نیست، و موجب هدر رفتن منابع خواهد شد. با این همه، مودم کابلی حتی از این نظر نیز بیشتر شبیه تلفن همراه است تا تلفن ثابت.

سهولت دستیابی یکی دیگر از تفاوت های ADSL و مودم کابلی است. هر کاربری یک خط تلفن دارد، اما همه آنها آنقدر به ایستگاه تلفن نزدیک نیستند که بتوانند ADSL بگیرند. از طرف دیگر، کابل نیز چیزی نیست که همه داشته باشند، اما اگر از قبل کابل دارید و شرکت طرف قرارداد شما دسترسی اینترنت هم عرضه می کند، باید گفت شانس آورده اید. در اینجا دیگر فاصله تا گره فیبر یا منبع اهمیتی ندارد. باید خاطر نشان کرد که، از آنجائیکه کابل اساساً رسانه ای تلویزیونی است (و از ابتدا برای این منظور نصب شده)، در اغلب شرکت ها و دفاتر اداری وجود ندارد.

خطوط ADSL (بدلیل ماهیت نقطه-به-نقطه آنها) ذاتاً از ایمنی بالاتری نسبت به کابل برخوردارند. هر کاربری که به کابل دسترسی داشته باشد، می تواند تمام بسته های منتشر شده روی آنرا بخواند. البته همانطور که قبلاً گفتیم، تمام شرکت های معتبر سرویس های کابلی ترافیک را در هر دو جهت رمز می کنند. اما همین که کسی بتواند اطلاعات رمز شده شما را هم بگیرد، ضریب ایمنی را پائین می آورد.

سیستم تلفن عموماً قابل اعتمادتر از کابل است. برای مثال، شرکت های تلفن دارای سیستم های برق اضطراری هستند، و سرویس های آنها حتی در صورت بروز خاموشی کامل هم قطع نمی شود. اما در سیستم های کابلی، اگر برق هر یک از تقویت کننده های میانی در یکی از زنجیره ها قطع شود، ترافیک دریافت آن خط بکلی متوقف خواهد شد.

و بالاخره اینکه، اغلب شرکت هایی که سرویس ADSL ارائه می کنند، انتخاب ISP را بر عهده خود شما می گذارند (در برخی نقاط حتی اجبار قانونی برای این کار وجود دارد). در مورد شرکت های کابلی، اغلب خود آنها ارائه دهنده سرویس اینترنت هم هستند، و دست شما برای انتخاب ISP باز نیست. نتیجه نهایی این بحث آنست که، ADSL و کابل بیشتر به هم شبیه اند تا متفاوت. آنها سرویس های مشابهی ارائه می کنند، و با افزایش رقابت قیمت آنها هم احتمالاً روز بروز به یکدیگر نزدیکتر خواهد شد.

۸-۲ خلاصه

لایه فیزیکی اساس تمام شبکه هاست. طبیعت دو محدودیت بنیادی به تمام کانال های ارتباطی تحمیل کرده، و همین محدودیت هاست که پهنای باند آنها را مشخص می کند. این دو محدودیت عبارتند از: حد نایکوئیست (Nyquist limit) که با کانال های بدون نویز سروکار دارد، و حد شانون (Shannon limit) که به کانال های نویزدار مربوط می شود.

رسانه انتقال می تواند هدایت پذیر (guided) باشد یا هدایت ناپذیر (unguided). مهمترین رسانه های هدایت پذیر عبارتند از: زوج-تابیده، کابل کواکسیال، و فیبر نوری. رسانه های هدایت ناپذیر نیز عبارتند از: امواج رادیویی، مایکروویو، مادون قرمز، و لیزر. یکی از رسانه های انتقال رو به رشد ماهواره های مخابراتی (بویژه سیستم های مدار پائین - LEO) هستند.

سیستم تلفن یکی از کلیدی ترین اجزای شبکه های گسترده (WAN) است، که مهمترین عناصر آن عبارتند از: مدارهای پایانی (local loop)، ترانک ها (trunk)، و سوئیچ ها (switch). مدارهای پایانی مدارهای آنالوگ زوج-تابیده هستند، که برای انتقال داده های دیجیتال روی آنها باید از مودم (modem) استفاده کرد. ADSL با استفاده از تکنیک های مدولاسیون و تقسیم مدار پایانی به کانال های مجازی می تواند به سرعت 50 Mbps برسد. مدارهای پایانی بیسیم (WLL) یکی از تکنولوژی های جدیدیست که در آینده از آن (بویژه از LMDS) بیشتر خواهید شنید.

ترانک های شبکه تلفن دیجیتالی هستند، و از تکنیک های مالتی پلکس (شامل FDM، TDM و WDM) در آنها استفاده می شود. تکنیک های سوئیچینگ نیز بر دو نوع سوئیچینگ مداری (circuit switching) و سوئیچینگ بسته ای (packet switching) است، که هر دو اهمیت زیادی دارند.

برای کاربردهایی که تحرک زیادی دارند، سیستم تلفن ثابت چندان مناسب نیست. تلفن های همراه امروزه بطور گسترده ای برای ارتباطات صدا مورد استفاده قرار می گیرند، و در آینده نزدیک ترافیک داده نیز در آنها به حد قابل ملاحظه ای خواهد رسید. نسل اول تلفن های همراه آنالوگ بود، که عمدتاً به سیستم های AMPS متکی بود. نسل دوم تلفن همراه دیجیتال است، که در آن از سیستم های GSM، D-AMPS و CDMA استفاده می شود. نسل سوم تلفن های همراه نیز دیجیتال خواهد بود، و در آن CDMA به نفع استفاده خواهد شد.

یکی از سیستم هایی که می توان از آن برای کاربردهای شبکه نیز بهره گرفت، تلویزیون کابلی (که از تلویزیون با آنتن مرکزی به سیستم های آمیخته فیبر-کواکس تکامل یافته) است. این سیستم پهنای باند بالقوه زیادی دارد، ولی پهنای باند واقعی آن به تعداد کاربران فعال (و اینکه مشغول چه کاری هستند) بستگی دارد.

مسائل

۱. ضرایب فوریه تابع $f(t) = t$ را محاسبه کنید ($0 \leq t \leq 1$).
۲. هر 1 msec از یک کانال بدون نویز 4-kHz نمونه برداری می شود. حداکثر نرخ داده این کانال چقدر است؟
۳. کانال های تلویزیونی 6 MHz پهنای دارند. اگر از یک سیگنال دیجیتال چهارسطحی استفاده کنیم، چند bit/sec می توان در این کانال مخابره کرد؟ فرض کنید کانال بدون نویز است.
۴. اگر یک سیگنال باینری در کانالی 3-kHz که نسبت سیگنال به نویز آن 20 dB است، مخابره شود، حداکثر نرخ داده قابل دستیابی چقدر است؟
۵. برای آن که بتوان کاربر T1 را روی یک خط 50-kHz قرار داد، نسبت سیگنال به نویز چقدر باید باشد؟
۶. فرق ستاره غیرفعال و تکرارکننده فعال در یک شبکه فیبر نوری چیست؟
۷. پهنای باند موجود در طیفی به پهنای $0.1 \mu\text{m}$ در طول موج $1 \mu\text{m}$ چقدر است؟
۸. می خواهیم یکسری تصاویر کامپیوتری اسکن شده را روی یک رشته فیبر نوری بفرستیم. وضوح هر تصویر 480×640 پیکسل، و هر پیکسل ۲۴ بیت است. تصاویر با سرعت ۶۰ صفحه بر ثانیه اسکن می شوند. پهنای باند مورد نیاز چقدر است؟ اگر از باند $1.30 \mu\text{m}$ استفاده کنیم، به چند میکرون از طول موج نیاز داریم؟
۹. آیا قضیه نایکونیست برای فیبرهای نوری هم صادق است، یا فقط برای کابلهای مسی کاربرد دارد؟
۱۰. در شکل ۲-۶، باند سمیت چپ از بقیه باندها باریکتر است. چرا؟

۱۱. یک آنتن زمانی بهترین بهره را دارد که قطر آن معادل طول موج امواج رادیویی باشد. قطر قابل قبول آنتنها بین ۱ تا ۵ متر است. این قطر معادل کدام طیف فرکانسی است؟
۱۲. محوشدگی چندمسیره زمانی به حداکثر می رسد که دو موج با اختلاف فاز 180° درجه وارد گیرنده شوند. برای به حداکثر رسیدن محوشدگی چندمسیره در یک لینک مایکروویو 1-GHz بطول 50-km این مقدار چقدر باید باشد؟
۱۳. پرتو لیزری بقطر 1 mm روی آشکارسازی بقطر 1 mm که روی پشت بامی در فاصله 100 m قرار دارد، نشانه گرفته شده است. حداکثر انحراف زاویه ای (بر حسب درجه) چقدر باید باشد، تا پرتو لیزری هدف را گم نکند؟
۱۴. ۶۶ ماهواره پروژۀ ایریدیوم به شش کمربند بدور زمین تقسیم شده اند. در ارتفاعی که این ماهواره ها قرار دارند، دورۀ گردش مداری ۹۰ دقیقه است. متوسط زمان پاس کاری یک فرستندۀ زمینی بین دو ماهواره چقدر است؟
۱۵. ماهواره ای در مدار زمین ثابت با صفحه استوای زمین زاویه ϕ می سازد. آیا برای فردی که روی زمین در مدار ϕ درجه شمالی ایستاده، این ماهواره در آسمان ثابت بنظر می رسد؟ اگر نه، حرکت آنرا توضیح دهید.
۱۶. قبل از سال ۱۹۸۴ (وقتی هر ایستگاه پایانی با کد سه رقمی ناحیه و سه رقم اول شماره تلفن مشخص می شد) در سیستم تلفن چند کد ایستگاه پایانی می توانست وجود داشته باشد؟ کد ناحیه با عددی بین ۲ تا ۹ شروع می شد، رقم دوم می توانست ۰ یا ۱ باشد، و رقم سوم محدودیتی نداشت. دو رقم اول کد محلی نیز بایستی بین ۲ تا ۹، و رقم سوم می توانست هر عددی باشد.
۱۷. فقط با اطلاعاتی که در اینجا بدست آوردید، آیا می توانید بگوئید حداکثر شماره تلفنهایی که (بدون تغییر در روش شماره گذاری یا اضافه کردن تجهیزات) می توان در ایالات متحده نصب کرد، چه تعداد است؟ آیا این تعداد شماره قابل دستیابی است؟ هر دستگاه فکس یا کامپیوتر را نیز یک تلفن در نظر بگیرید، و فرض کنید هر مشترک فقط یک دستگاه تلفن دارد.
۱۸. یک سیستم ساده تلفن را که در آن دو ایستگاه پایانی و یک ایستگاه بین شهری بوسیله خطوط دو-طرفه همزمان 1-MHz به هم متصل شده اند، در نظر بگیرید. هر دستگاه تلفن بطور متوسط در هر روز کاری ۸ ساعته ۴ تماس برقرار می کند، و زمان متوسط هر تماس ۶ دقیقه است. ده درصد تماسها راه دور هستند (یعنی از ایستگاه بین شهری رد می شوند). حداکثر تعداد شماره هایی که هر ایستگاه پایانی می تواند پشتیبانی کند، چقدر است؟ مدارها را 4-kHz در نظر بگیرید.
۱۹. یک شرکت تلفن منطقه ای ۱۰ میلیون مشترک دارد، که بوسیله زوج-تاییده به ایستگاه مرکزی متصل شده اند. متوسط طول مدارهای پایانی ۱۰ کیلومتر است. ارزش مس موجود در مدارهای پایانی این سیستم چقدر است؟ سطح مقطع سیمها را دایره ای بقطر 1 mm، چگالی مس را 9.0 gr/cm^3 و ارزش هر کیلوگرم مس را 3 دلار در نظر بگیرید.
۲۰. یک خط لوله نفت سیستمی یکطرفه است، یا دو-طرفه ناهمزمان، یا دو-طرفه همزمان، یا هیچکدام؟
۲۱. قیمت میکروپروسسورهای سریع آنقدر کاهش یافته، که می توان در هر دستگاه مودم یک میکروپروسسور قرار دارد. این کار چه تأثیری روی مقابله با خطاهای خطوط تلفن دارد؟
۲۲. دیاگرام فلکی شکل ۲-۲۵ چهار نقطه داده در مختصات $(1, 1)$ ، $(1, -1)$ ، $(-1, 1)$ و $(-1, -1)$ دارد. مودمی با این پارامترها در 1200 baud به چه سرعتی (bps) می تواند دست یابد؟
۲۳. مودمی با دیاگرام فلکی شبیه شکل ۲-۲۵ دارای نقاط داده ای در مختصات $(0, 1)$ و $(0, 2)$ است. این مودم

- از مدولاسیون فاز استفاده می‌کند، یا مدولاسیون دامنه؟
۲۴. در یک دیاگرام فلکی تمام نقاط روی دایره‌ای به مرکز مبدأ مختصات واقع شده‌اند. مدولاسیون این مودم چیست؟
۲۵. یک مودم دو-طرفه همزمان QAM-64 از چند فرکانس استفاده می‌کند؟
۲۶. در یک سیستم ADSL که از DMT استفاده می‌کند، 3/4 کانالهای موجود به لینک دریافت اختصاص داده شده است. هر کانال از مدولاسیون QAM-64 استفاده می‌کند. ظرفیت لینک دریافت چقدر است؟
۲۷. در سیستم LMDS چهار قطاعی شکل ۲-۳۰، هر قطاع یک کانال اختصاصی 36-Mbps دارد. طبق تئوری صف، اگر 50% کانال پُر باشد، زمان انتظار در صف معادل زمان بار شدن است. در چنین شرایطی، بار شدن یک صفحه وب 5-KB چقدر طول خواهد کشید؟ بار شدن این صفحه روی یک خط ADSL با سرعت 1 Mbps چقدر طول می‌کشد؟ یا یک مودم 56-kbps چقدر؟
۲۸. ده سیگنال، که هر کدام به پهنای باند 4000 Hz نیاز دارند، با استفاده از FDM روی یک کانال مالتی پلکس شده‌اند. حداقل پهنای باند مورد نیاز این کانال چقدر است؟ پهنای باندهای محافظ را 400 Hz در نظر بگیرید.
۲۹. چرا زمان نمونه برداری PCM در 125 μsec ثابت شده است؟
۳۰. درصد سرآیند یک کاربر T1 (درصدی از 1.544 Mbps که بکار داده‌های کاربر نمی‌آید) چقدر است؟
۳۱. حداکثر نرخ داده یک کانال بدون نویز 4-kHz را با استفاده از تکنیکهای زیر مقایسه کنید:
(الف) کدگذاری آنالوگ (مثلاً، QPSK) با 2 bits/sample
(ب) سیستم T1 PCM
۳۲. اگر یک سیستم T1 دچار لغزش شود، برای سنکرون شدن مجدد از اولین بیت هر فریم استفاده می‌کند. برای سنکرون شدن مجدد با احتمال خطای 0.001، چند فریم باید بررسی شود؟
۳۳. فرق بخش دمودلاتور یک مودم با بخش دکودر یک کدک چیست (و آیا اساساً فرقی دارند)؟ توجه داشته باشید که هر دوی اینها سیگنالهای آنالوگ را به دیجیتال تبدیل می‌کنند.
۳۴. سیگنالی بصورت دیجیتال روی یک کانال بدون نویز 4-kHz (با یک نمونه در هر 125 μsec) فرستاده می‌شود. با هر یک از روشهای کدگذاری زیر چند بیت در ثانیه ارسال می‌شود:
(الف) استاندارد CCITT 2.048 Mbps
(ب) سیستم DPCM با مقدار نسبی سیگنال 4-bit
(ج) مدولاسیون دلنا.
۳۵. یک سیگنال سینوسی کامل با دامنه A با استفاده از مدولاسیون دلنا (با x samples/sec) کُد شده است. خروجی 1+ معادل $A/8$ تغییر در سیگنال ورودی، و خروجی 1- معادل $A/8$ تغییر در سیگنال ورودی است. بیشترین فرکانسی که این سیستم می‌تواند بدون خطای تجمعی تعقیب کند، چقدر است؟
۳۶. ساعت‌های SONET خطایی معادل 1 در 10^9 دارند. چه مدت طول می‌کشد، تا این اختلاف باندازه 1 بیت شود؟ عوارض جانبی این پدیده چیست؟
۳۷. در شکل ۲-۳۷، نرخ داده کاربر OC-3 از 148.608 Mbps شروع شده است. نشان دهید این عدد چگونه از پارامترهای SONET OC-3 بدست آمده است.
۳۸. سیستم SONET برای انطباق با سرعت‌های پائین‌تر از STS-1 از روشی بنام انشعابات مجازی (Virtual Tributaries - VT) استفاده می‌کند. یک VT عبارتست از یک بار جزئی، که می‌توان آنرا به همراه بارهای

- جزئی دیگر در یک فریم STS-1 قرار داد. VT1.5 از ۳ ستون، VT2 از ۴ ستون، VT3 از ۶ ستون و VT6 از ۱۲ ستون فریم STS-1 استفاده می کنند. کدام VT با هر یک از سرویسهای زیر منطبق است؟
- (الف) سرویس DS-1 (1.544 Mbps).
- (ب) سرویس European CEPT-1 (2.048 Mbps).
- (ج) سرویس DS-2 (6.312 Mbps).
۳۹. تفاوت بنیادی سونیچینگ مداری با سونیچینگ بسته ای چیست؟
۴۰. پهنای باند کاربر یک اتصال OC-12c چقدر است؟
۴۱. سه شبکه سونیچینگ بسته ای هر کدام ۳ گره دارند. شبکه اول دارای توپولوژی ستاره (با سونیچ مرکزی) است، شبکه دوم حلقه (دوطرفه) است، و شبکه سوم اتصالات کامل داخلی دارد (یعنی هر گره مستقیماً به تمام گره های دیگر متصل است). بهترین، بدترین و متوسط پرش (hop) در ارتباط از یک نقطه به نقطه دیگر در هر یک از این شبکه ها چیست؟
۴۲. زمان تأخیر ارسال یک پیام x -bit در مسیری با k پرش در یک شبکه سونیچینگ مداری و یک شبکه سونیچینگ بسته ای (با بار کم) را با یکدیگر مقایسه کنید. زمان برقراي مدار را s ثانیه، زمان تأخیر در هر پرش را d ثانیه، اندازه هر بسته را p بیت، و نرخ انتقال داده را b pbs در نظر بگیرید. در چه شرایطی تأخیر ارسال شبکه سونیچینگ بسته ای کمتر است؟
۴۳. فرض کنید می خواهیم x بیت اطلاعات را در یک شبکه سونیچینگ بسته ای با k پرش، بصورت بسته هایی با p بیت داده و h بیت سرآیند (با این فرض که $x \gg p + h$) منتقل کنیم. نرخ انتقال داده خطوط b pbs، و زمان تأخیر انتشار در آنها قابل صرف نظر کردن است. چه مقداری از p تأخیر کلی را به حداقل می رساند؟
۴۴. در یک سیستم تلفن همراه با سلولهای شش ضلعی، استفاده از باندهای فرکانسی مشابه در سلولهای مجاور ممنوع است. اگر ۸۴۰ باند فرکانسی داشته باشیم، در هر سلول از چند فرکانس می توان استفاده کرد؟
۴۵. طرح کلی سلولهای یک شبکه تلفن همراه بندرت مانند شکل ۲-۴۱ منظم است؛ حتی شکل هر سلول نیز منظم نیست. یک علت برای این وضعیت بیاورید.
۴۶. برای پوشش دادن به شهری با مساحت 120 km^2 ، به چه تعداد سلول PCS با قطر 100 m نیاز داریم؟ (تخمین بزنید.)
۴۷. گاهی هنگام عبور یک کاربر تلفن همراه از سلولی به سلول دیگر، با وجود اینکه تمام فرستنده ها و گیرنده ها بخوبی کار می کنند، ارتباط ناگهان قطع می شود. چرا؟
۴۸. کیفیت صدای D-AMPS بسیار پایتتر از GSM است. آیا این بخاطر اجبار D-AMPS در حفظ سازگاری با AMPS است (در حالیکه GSM چنین محدودیتی ندارد)؟ یا علت دیگری دارد؟
۴۹. حداکثر تعداد کاربران همزمان در یک سلول D-AMPS را محاسبه کنید. آیا چنین محاسبه ای برای GSM هم امکان دارد؟ علت را توضیح دهید.
۵۰. فرض کنید سه ایستگاه A ، B و C در یک سیستم CDMA (با توالبهای چپب شکل ۲-۴۵ ب) همزمان اقدام به ارسال بیت های 0 می کنند. توالی چپب حاصله چیست؟
۵۱. در بحث متعامد بودن بردارهای توالی چپب CDMA، گفتیم که اگر $S \cdot T = 0$ ، آنگاه $S \cdot \bar{T} = 0$ ثابت کنید.
۵۲. اجازه دهید متعامد بودن بردارهای توالی چپب CDMA را به روشی دیگر بیان کنیم: هر بیت در یک جفت توالی یا یکسان هستند، یا نیستند. متعامد بودن بردارها را با استفاده از اصطلاحات یکسان بودن و یکسان نبودن توضیح دهید.

۵۳. یک گیرنده CDMA توالی $(+1 +1 -3 +1 -1 -3 +1 +1)$ را دریافت می کند. با فرض توالیهای چپ شکل ۲-۴۵، تعیین کنید کدام ایستگاه ها، چه بیت هایی را ارسال کرده اند؟
۵۴. شبکه تلفن در بخش انتهایی دارای توپولوژی ستاره است، که در آن تمام انشعابات به ایستگاه پایانی ختم می شوند. در حالیکه در تلویزیون کابلی، یک کابل مشترک مانند ماری بین مشترکین مختلف خزیده است. فرض کنید در آینده در شبکه های کابلی بجای کابل های مسی از فیبر نوری 10 Gbps استفاده شود. آیا با چنین سیستمی می توان مدل شبکه تلفن (یک خط مستقل از هر مشترک به ایستگاه مرکزی) را شبیه سازی کرد؟ اگر پاسخ مثبت است، هر فیبر چند کاربر تلفن می تواند داشته باشد؟
۵۵. سیستم های تلویزیون کابلی معمولاً دارای ۱۰۰ کانال تجاری هستند، که بطور متناوب برنامه و آگهی پخش می کنند. این سیستم بیشتر شبیه TDM است یا FDM؟
۵۶. یک شرکت کابلی تصمیم می گیرد به ۵۰۰۰ مشترک خود سرویس اینترنت ارائه دهد. این شرکت از کابل های کوکسیال استفاده می کند، که هر کابل می تواند تا 100 Mbps روی کانال دریافت از اینترنت ظرفیت داشته باشد. شرکت مزبور برای جذب مشتریان تصمیم می گیرد که تا دریافت 2 Mbps را برای هر مشترک تضمین کند. توضیح دهید این شرکت برای رسیدن به هدف فوق چه کاری باید انجام دهد؟
۵۷. با توجه به تخصیص فرکانس نشان داده شده در شکل ۲-۴۸ و اطلاعات داده شده در متن کتاب، یک سیستم کابلی چه مقدار (Mbps) از ظرفیت را به ارسال به اینترنت و چه مقدار را به دریافت از اینترنت اختصاص می دهد؟
۵۸. اگر سیستم کابلی کاملاً بیکار باشد، کاربر با چه سرعتی می تواند اطلاعات را دریافت کند؟
۵۹. مالتی پلکس کردن استریم های متعدد STS-1 (که به آنها انشعاب گفته می شود) نقش مهمی در سیستم SONET بازی می کند. یک مالتی پلکسر 3:1 سه ورودی STS-1 را در یک خروجی STS-3 مالتی پلکس می کند. اینکار بصورت بایت به بایت انجام می شود، یعنی سه بایت اول خروجی بترتیب بایتهای اول انشعابهای 1، 2 و 3 هستند؛ سه بایت دوم خروجی بترتیب بایتهای دوم انشعابهای 1، 2 و 3؛ و الی آخر. برنامه ای بنویسید که این مالتی پلکسر 3:1 را شبیه سازی کند. برنامه شما باید پنج روال داشته باشد: روال اصلی (که چهار روال دیگر را اجرا می کند)، یک روال برای هر یک از انشعابهای STS-1 (مجموعاً سه روال)، و یکی برای مالتی پلکسر. هر روال انشعاب یک فریم STS-1 را از فایلی بطول ۸۱۰ بایت خوانده، و این فریمها را (بایت به بایت) به روال مالتی پلکسر می فرستد. روال مالتی پلکسر این بایتها را خوانده، و یک فریم STS-3 را (بایت به بایت) روی خروجی استاندارد (stdout) می نویسد. برای ارتباط بین پردازشها از پایپ (pipe) استفاده کنید.

لایه پیوند داده



در این فصل اصول طراحی لایه دوم، لایه پیوند داده (data link layer)، را بررسی خواهیم کرد، و طی آن با الگوریتمهای لازم برای دستیابی به یک ارتباط قابل اطمینان و کارا بین دو کامپیوتر همسایه (در لایه پیوند داده) آشنا خواهیم شد. منظور از دو کامپیوتر همسایه، کامپیوترهایی هستند که یک کانال ارتباطی سیم-مانند (کابل کواکسیال، خط تلفن، و یا ارتباط بیسیم) بین آنها برقرار است. خصلت بنیادی یک کانال «سیم-مانند» اینست که بیت‌ها دقیقاً با همان نظمی که فرستاده می‌شوند، در گیرنده دریافت شوند.

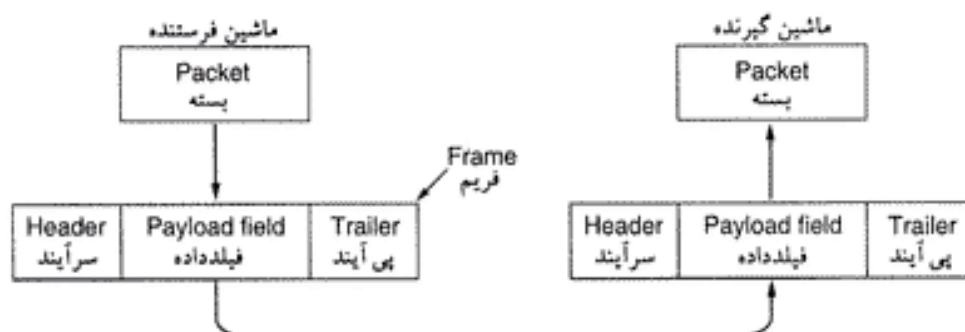
شاید در نگاه اول این خصلت آنقدر ساده و ابتدایی بنظر برسد، که فکر کنید چه نیازی به الگوریتم و نرم‌افزار هست: ماشین A بیت‌ها را می‌فرستد، و ماشین B آنها را می‌گیرد. متأسفانه، مسئله بهمین سادگی نیست، چون مدارهای مخابراتی پر از نویز و خطا هستند. علاوه بر آن ظرفیت کانالهای مخابراتی نامحدود نیست، و بین ارسال و دریافت بیت‌ها یک تأخیر زمانی نیز وجود دارد. این محدودیت‌ها تأثیر جدی روی کارایی سیستمهای انتقال داده می‌گذارند. پروتکل‌های مخابراتی (که موضوع اصلی این فصل هستند) باید تمام این ملاحظات را در نظر بگیرند. بعد از آشنایی با نکات کلیدی در طراحی لایه پیوند داده، پروتکل‌های این لایه را بررسی خواهیم کرد. برای شروع خصلت خطاهای کانالهای مخابراتی، منشأ آنها و نحوه کشف و رفع این خطاها را بررسی کرده، و سپس پروتکل‌های لازم برای حل آنها را مورد مطالعه قرار می‌دهیم. در پایان، صحت این مدل‌ها را بررسی کرده، و چندان نمونه از پروتکل‌های واقعی لایه پیوند داده ارائه خواهیم کرد.

۱-۳ ملاحظات طراحی لایه پیوند داده

لایه پیوند داده وظایف خاصی دارد که باید انجام دهد. این وظایف عبارتند از:

۱. ارائه سرویسهای مشخص به لایه شبکه.
۲. مدیریت خطاهای انتقال.
۳. تنظیم جریان داده‌ها (بگونه‌ایکه گیرنده‌های گنبد زیر بمباران فرستنده‌های سریع غرق نشوند).

برای رسیدن به این اهداف، لایه پیوند داده بسته‌های رسیده از لایه شبکه را گرفته و آنها بصورت فریم (frame) در می‌آورد. هر فریم سه قسمت دارد: سرآیند (header)، داده اصلی، و پی‌آیند (trailer)؛ شکل ۱-۳ را ببینید. مدیریت فریم‌ها کلیدی‌ترین وظیفه لایه پیوند داده است، که در بخشهای آینده بتفصیل درباره آن صحبت خواهیم کرد

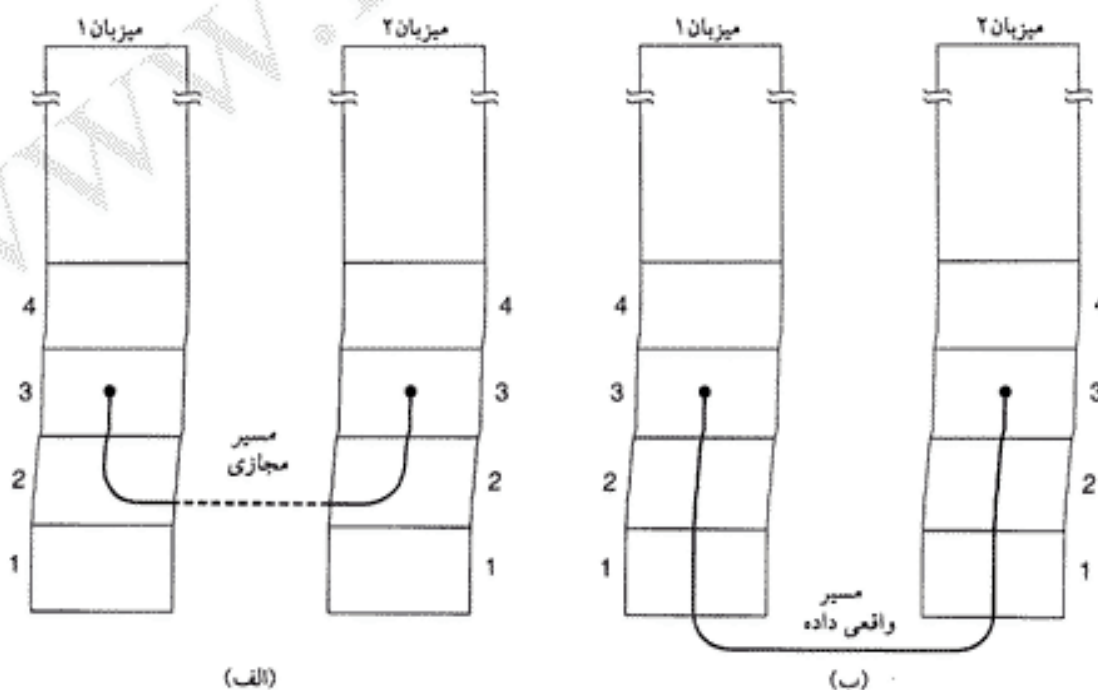


شکل ۱-۳. رابطه بسته و فریم.

با اینکه این فصل منحصرأ درباره لایه پیوند داده و پروتکل های آن است، اصولی که در اینجا خواهید دید (مانند کنترل خطا و کنترل جریان داده) در لایه های دیگر (مانند لایه انتقال) نیز کاربرد دارند. در حقیقت، در بسیاری از شبکه ها این کارکردها را فقط در لایه های بالاتر (نه در لایه پیوند داده) می توانید پیدا کنید. اما صرفنظر از اینکه آنها را کجا می توان پیدا کرد، اصول کار یکسان است و اهمیتی ندارد که در این فصل آنها را بررسی کنیم یا فصلهای دیگر. تنها مزیت لایه پیوند داده آنست که این تکنیکها در این لایه ساده تر و واضح ترند، بنابراین بخوبی می توان آنها را مطالعه کرد.

۱-۱-۳ سرویسهایی که به لایه شبکه داده می شود

وظیفه لایه پیوند داده ارائه سرویس به لایه شبکه است. مهمترین این وظایف عبارتست از انتقال داده ها از لایه شبکه ماشین مبدأ به لایه شبکه ماشین مقصد. در لایه شبکه ماشین مبدأ چیزی هست بنام پروسس، که تعدادی بیت را به لایه پیوند داده می دهد تا به مقصدی خاص منتقل کند. وظیفه لایه پیوند داده ماشین مبدأ انتقال این بیت ها به ماشین مقصد، و رساندن آنها بدست لایه شبکه مقابل است؛ شکل ۲-۳ (الف) را ببینید. البته مسیری که این بیت ها



شکل ۲-۳. (الف) ارتباط مجازی. (ب) ارتباط واقعی.

واقعاً طی می‌کنند، مانند شکل ۳-۲ (ب) است، ولی ساده‌ترست تصور کنیم دو پروسس در لایه پیوند داده آنها را بین خود رد و بدل می‌کنند. بهمین دلیل در این فصل همه جا از مدل شکل ۳-۲ (الف) استفاده خواهیم کرد. لایه پیوند داده را می‌توان بگونه‌ای طراحی کرد که سرویسهای مختلفی ارائه کند، که این سرویسها از سیستم به سیستم دیگر متفاوت است. معقولترین این سرویسها عبارتند از:

۱. سرویس غیرمتصل بدون تصدیق دریافت (unacknowledged connectionless).
۲. سرویس غیرمتصل با تصدیق دریافت (acknowledged connectionless).
۳. سرویس اتصال-گرا با تصدیق دریافت (acknowledged connection-oriented).

اجازه دهید این سرویسها را یکی یکی بررسی کنیم.

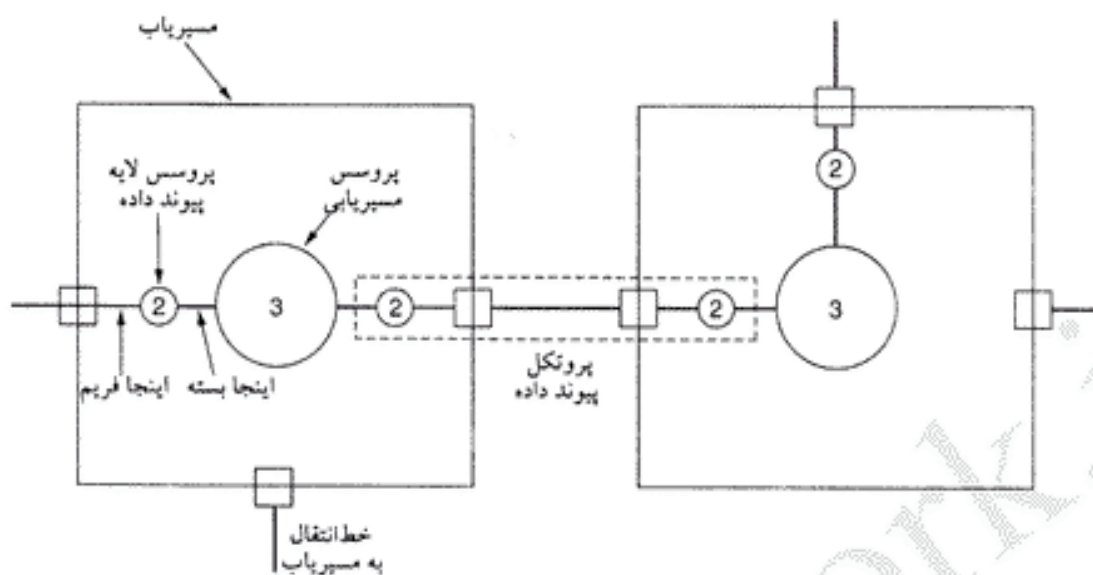
در سرویس غیرمتصل بدون تصدیق دریافت ماشین مبدأ فریمهای مستقلی را به ماشین مقصد می‌فرستد، بدون اینکه منتظر تصدیق دریافت آنها از طرف ماشین مقصد بماند. هیچ اتصال منطقی بین دو ماشین برقرار نمی‌شود، پس نیازی به قطع اتصال هم نیست. اگر فریمی در اثر نویز خط از بین برود، هیچ کوششی برای تشخیص این موضوع و مقابله با آن در لایه پیوند داده صورت نمی‌گیرد. این سرویس برای مواقعی مناسب است که نرخ خطا بسیار پائین باشد، و در این حالت مقابله با خطا به لایه‌های بالاتر واگذار می‌شود. این سرویس برای ترافیک زمان-واقعی (مانند سرویس صدا)، که در آن دیر رسیدن بدتر از نرسیدن است، نیز مناسب است. در اغلب LAN ها نیز لایه پیوند داده از سرویسهای غیرمتصل بدون تصدیق دریافت استفاده می‌کند.

سرویس بعدی که قابلیت اعتماد بیشتری دارد، سرویس غیرمتصل با تصدیق دریافت است. در این سرویس نیز هیچ اتصال منطقی بین مبدأ و مقصد وجود ندارد، ولی دریافت فریمها از سوی ماشین مقصد تصدیق می‌شود. بدین ترتیب، فرستنده می‌تواند پی ببرد که آیا فریمها بدرستی دریافت شده‌اند یا خیر. اگر فریمی در مدت زمان معین به مقصد نرسد، می‌توان آنرا دوباره ارسال کرد. این سرویس برای کانالهای غیر قابل اعتماد (مانند سیستمهای بیسیم) مناسب است.

لازمست تأکید کنیم که توجه به تصدیق دریافت در لایه پیوند داده فقط برای بهینه‌سازی سیستم است و هیچ الزامی در آن نیست، چون این کار را همیشه می‌توان در لایه شبکه انجام داد. اگر تصدیق دریافت در زمان مشخص از راه نرسد، فرستنده می‌تواند بسته را دوباره ارسال کند. مشکل اینجااست که فریمها معمولاً طول مشخصی دارند، در حالیکه بسته‌ای که لایه شبکه می‌فرستد چنین نیست. اگر بسته‌ای به، مثلاً، ۱۰ فریم شکسته شود، و ۲۰ درصد این فریمها در راه گم شوند، زمان ارسال بسته بسیار طولانی خواهد شد. اما اگر برای هر فریم تصدیق دریافت درخواست شود، این کار سریعتر می‌شود. در کانالهای قابل اعتماد مانند فیبر نوری، بار اضافی چنین پروتکل‌های سختگیرانه‌ای در لایه پیوند داده غیر ضروری است، اما در محیطهای ذاتاً پرنویز مانند بیسیم ارزشش را دارد.

بهترین سرویسی که لایه پیوند داده می‌تواند به لایه شبکه بدهد، سرویس اتصال-گرا (connection-oriented) است. در این سرویس قبل از شروع ارسال داده از مبدأ به مقصد، یک اتصال بین آنها برقرار می‌شود. هر فریمی که روی این اتصال فرستاده می‌شود شماره‌گذاری شده است، و لایه پیوند داده دریافت آنها را نیز تضمین می‌کند. همچنین تضمین می‌شود که هر فریم فقط یک بار (و به همان ترتیب ارسال) دریافت شود. اما در سرویسهای غیرمتصل، می‌توان انتظار داشت که بسته‌ای چندین بار ارسال (و در نتیجه چندین بار هم دریافت) شود. سرویس اتصال-گرا استریم قابل اعتمادی از بیت‌ها را در اختیار لایه شبکه می‌گذارد.

ارسال داده‌ها در سرویس اتصال-گرا سه مرحله دارد. در مرحله اول اتصال برقرار شده، و متغیرهای لازم (برای شمارش فریمها، و اینکه کدام فریمها دریافت شده‌اند و کدامها خیر) ست می‌شوند. در مرحله دوم، فریمها منتقل می‌شوند. و در مرحله آخر، اتصال قطع شده و منابع آن (متغیرها و بافرها) آزاد می‌شود.



شکل ۳-۳. محل فعالیت لایه پیوند داده.

اجازه دهید یک مثال بزنیم: یک زیرشبکه WAN متشکل از چند مسیریاب که با خطوط نقطه-به-نقطه تلفن به یکدیگر متصل شده‌اند، را در نظر بگیرید. وقتی یک فریم به مسیریاب می‌رسد، سخت‌افزار (با استفاده از تکنیکهایی که در همین فصل خواهید دید) آنرا از نظر خطا چک می‌کند، و سپس به نرم‌افزار لایه پیوند داده (که می‌تواند روی چیپهای کارت شبکه قرار داشته باشد) تحویل می‌دهد. نرم‌افزار لایه پیوند داده فریم را چک می‌کند تا مطمئن شود همان چیزیست که باید باشد، و اگر چنین بود، قسمت داده اصلی آنرا به نرم‌افزار مسیریابی می‌دهد. نرم‌افزار مسیریابی مسیر خروجی مناسب را تعیین کرده، و بسته را به لایه پیوند داده پس می‌دهد تا ارسال شود (شکل ۳-۳ را ببینید).

نرم‌افزارهای مسیریابی معمولاً حوصله بسته‌هایی که مدام گم می‌شوند را ندارند، و دوست دارند بسته‌ها درست و مرتب روی خطوط نقطه-به-نقطه تحویل شوند. این دیگر بر عهده پروتکل لایه پیوند داده است که خطوط پرنویز و غیر قابل اعتماد را بصورتی مطمئن (یا نسبتاً مطمئن) در آورد. با اینکه در شکل ۳-۳ نرم‌افزار لایه پیوند داده (در هر مسیریاب) در دو نقطه دیده می‌شود، اما این در واقع یک پروسس واحد است که (به کمک جدول‌ها و ساختمان داده‌های مختلف) تمام کارها را انجام می‌دهد و تمام خطوط را کنترل می‌کند.

۲-۱-۳ فریم‌بندی

لایه پیوند داده به لایه شبکه سرویس می‌دهد، و خود نیز از سرویسهای لایه فیزیکی استفاده می‌کند. چیزی که لایه فیزیکی می‌گیرد، استریمی است از بیت‌ها که باید آنرا به طرف مقابل تحویل دهد. هیچ تضمینی وجود ندارد که این استریم سالم و عاری از خطا به مقصد برسد. تعداد بیت‌های رسیده می‌تواند کمتر، مساوی یا بیشتر از بیت‌های ارسال شده باشد، و یا حتی مقدار برخی از آنها تغییر کرده باشد. این بر عهده لایه پیوند داده است که خطاها را کشف کرده، و در صورت لزوم آنها را برطرف کند.

یکی از روشهای متداول اینست که استریم بیت‌ها در لایه پیوند داده به چند فریم شکسته شده، و برای هر فریم جمع تطبیقی (checksum) محاسبه شود. (الگوریتمهای جمع تطبیقی را در همین فصل خواهید دید). وقتی فریمها به مقصد می‌رسند، جمع تطبیقی آنها مجدداً محاسبه شده و با جمع تطبیقی مبدأ (که به انتهای فریم ضمیمه شده) مقایسه می‌شود. اگر این دو یکی نباشند، لایه پیوند داده متوجه می‌شود که خطایی در فریم رخ داده، و به

سراغ روشهای مقابله با خطا می رود (که یکی از این روشها می تواند دور انداختن فریم، و درخواست ارسال مجدد آن باشد).

شکستن استریم بیت ها به فریم (که به آن فریم بندی - framing - گفته می شود) از آنچه در نگاه اول بنظر می رسد، مشکلتر است. یکی از روشهای فریم بندی می تواند انداختن فاصله زمانی در نقاطی از استریم بیت ها باشد (مانند فاصله انداختن بین کلمات متن). ولی در شبکه ها پندرت زمان بندی وجود دارد، و امکان دارد این فاصله ها از بین بروند و یا بیشتر شوند.

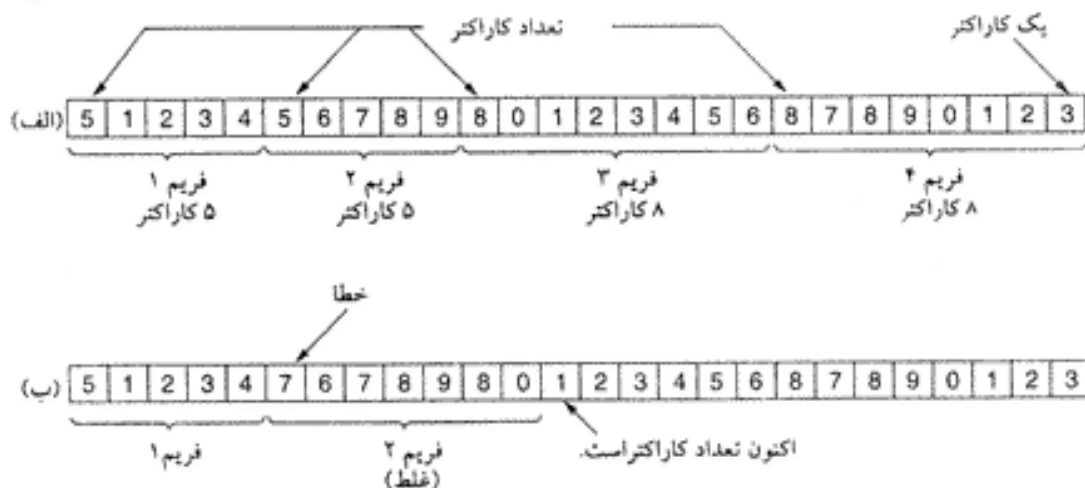
از آنجائیکه تکیه بر زمان بندی برای تعیین ابتدا و انتهای فریمها بسیار خطرناک است، روشهای دیگری برای اینکار ابداع شده، که در این قسمت با چهار تا از آنها آشنا می شوید:

۱. شمارش کاراکترها.
۲. بایت های پرچم، یا لاگذاری بایت.
۳. پرچمهای شروع و پایان، یا لاگذاری بیت.
۴. حالت های غیرمجاز گذگذاری لایه فیزیکی.

در اولین روش فریم بندی تعداد کاراکترهای فریم در یکی از فیلدهای سرآیند آن نوشته می شود. وقتی این فریم به مقصد می رسد، لایه پیوند داده می تواند به کمک این فیلد ابتدا و انتهای فریم را مشخص کند. در شکل ۳-۴ (الف) چهار فریم با تعداد کاراکترهای ۵، ۵، ۸ و ۸ را می بینید.

اشکال این روش آنست که فیلد تعداد کاراکترها نیز می تواند دچار خطا شود. برای مثال در شکل ۳-۴ (ب)، فیلد تعداد کاراکترها در فریم دوم از ۵ به ۷ تبدیل شده است، و ماشین مقصد دیگر قادر نیست فریمهای بعدی را بدرستی بخواند (چون قادر نیست ابتدای آنها را تشخیص دهد). حتی اگر جمع تطبیقی اشتباه باشد و ماشین مقصد متوجه باشد که خطایی رخ داده، باز هم تشخیص نقطه شروع بعدی برای آن غیرممکن است. درخواست ارسال مجدد نیز کمکی نمی کند، چون ماشین مقصد نمی داند فریمها تا کجا درست بوده، و اشتباه از کجا رخ داده است و نیاز به ارسال مجدد دارد.

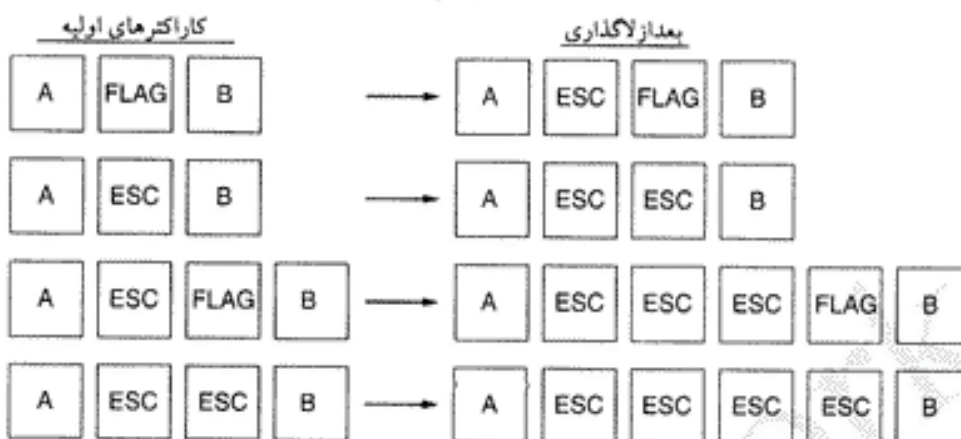
مشکل سنکرون شدن مجدد مبدأ و مقصد بعد از بروز خطا در روش دوم فریم بندی (بایت های پرچم، یا لاگذاری بایت) حل شده است، بدین ترتیب که هر فریم با تعدادی بایت خاص شروع و پایان می یابد. در گذشته، بایتهای شروع و پایان متفاوت بودند. ولی در سالهای اخیر از بایتهای یکسانی بعنوان بایت پرچم (flag byte) در



شکل ۳-۴. استریم کاراکترها. (الف) بدون خطا. (ب) با خطا.

FLAG	Header (سرآیند)	Payload field (فیلد بارکاری)	Trailer (پی آیند)	FLAG
------	--------------------	---------------------------------	----------------------	------

(الف)



(ب)

شکل ۵-۳. (الف) تعیین ابتدا و انتهای فریم با استفاده از بایت پرچم. (ب) چهار توالی

بایت قبل و بعد از لاگذاری بایت.

شروع و پایان فریم استفاده می شود. این بایتها را در شکل ۵-۳ (الف) با عنوان FLAG ملاحظه می کنید. در این روش اگر گیرنده همزمانی خود با فرستنده را از دست بدهد، فقط کفایت با جستجوی بایت پرچم انتهای فریم فعلی را پیدا کند. دو بایت پرچم که پشت سر هم بیایند، بمعنای پایان یک فریم و شروع فریم بعدی هستند.

یکی از مشکلات جدی این روش آنست که طرح بیت بایت پرچم می تواند در داده های اصلی نیز وجود داشته باشد (بویژه اگر اطلاعات از نوع برنامه های اجرایی یا اعداد اعشاری باشد). این وضعیت گیرنده را به اشتباه خواهد انداخت. یکی از راه های حل این وضعیت آنست که پروتکل لایه پیوند داده در سمت فرستنده قبل از هر توالی بیت پرچم که در داده اصلی ظاهر می شود، یک بایت گریز (escape byte) خاص قرار دهد. لایه پیوند داده مقصد این بایتها را حذف کرده، و داده های اصلی را به لایه شبکه تحویل می دهد. به این تکنیک لاگذاری بایت (byte stuffing) یا لاگذاری کاراکتر (character stuffing) گفته می شود. با این روش بایت پرچم بآسانی قابل تشخیص است، چون قبل از آن بایت گریز وجود ندارد.

اما حالا سؤال دیگری پیش می آید: اگر در وسط داده اصلی طرحی مشابه بایت گریز وجود داشت، چه اتفاقی می افتد؟ جواب اینست که قبل از این بایت هم یک بایت گریز قرار داده می شود؛ بعبارت دیگر دو بایت گریز پشت سر هم یعنی یک بایت گریز در داده اصلی. در شکل ۵-۳ (ب) چند نمونه از حالتی که می تواند پیش آید، آورده شده است. در هر مورد آن چیزی که گیرنده می گیرد، دقیقاً مشابه آن چیزیست که فرستنده ارسال کرده است.

تکنیک لاگذاری بایت که در شکل ۵-۳ نشان داده شده، شکل ساده شده آن چیزیست که در پروتکل PPP (یکی از مهمترین پروتکل های ارتباط با اینترنت در کامپیوترهای شخصی) مورد استفاده قرار می گیرد. (بعداً در همین فصل درباره PPP صحبت خواهیم کرد.)

یکی از معایب بزرگ فریم بندی بایت های پرچم با لاگذاری بایت وابستگی شدید آن به کاراکترهای ۸-بیتی است، و همانطور که می دانید تمام گدها ۸-بیتی نیستند (برای مثال، در استاندارد یونی کد از کاراکترهای ۱۶-بیتی استفاده می شود). فرض ۸-بیتی بودن کاراکترها در مکانیزم فریم بندی یکی از مشکلات جدی آن محسوب می شود، بهمین دلیل روش جدیدی که در آن طول کاراکتر می تواند متغیر باشد، ابداع شده است.

در این روش جدید طول کاراکترها اهمیتی ندارد، و فریمها می توانند تعداد بیتهای دلخواه داشته باشند. طرز کار این تکنیک جدید چنین است: هر فریم با طرح بیت خاصی (01111110 - که در واقع یک بایت پرچم است) شروع می شود. هرگاه لایه پیوند داده در سمت فرستنده پنج 1 پشت سر هم در داده اصلی دید، بطور خودکار یک 0 بعد از آن قرار می دهد. این روش، که به آن لاگذاری بیت (bit stuffing) گفته می شود، بسیار شبیه لاگذاری بایت است. وقتی گیرنده پنج 1 متوالی ببیند که یک 0 پشت سر آنها آمده، بطور خودکار این 0 را حذف می کند. لاگذاری بیت نیز مانند لاگذاری بایت بکلی از دید لایه شبکه در هر دو کامپیوتر پنهان (شفاف) است. اگر در داده کاربر طرح بیت 01111110 وجود داشته باشد، لایه پیوند داده فرستنده آنرا به 0111111010 تبدیل می کند، و در سمت گیرنده این 0 اضافی حذف شده و طرح بیت 01111110 به لایه بالاتر تحویل داده می شود. به یک مثال در شکل ۳-۶ توجه کنید.

(الف) 011011111111111111110010

(ب) 011011111011111101111010010

بیت های لاگذاری

(ج) 011011111111111111110010

شکل ۳-۶. لاگذاری بیت. (الف) داده اولیه. (ب) داده ها بصورتی که روی خط فیزیکی

ارسال می شود. (ج) داده ها بصورتی که در گیرنده دریافت می شود.

در روش لاگذاری بیت نیز محدوده فریم با استفاده از پرچمهای شروع و پایان مشخص می شود، و گیرنده می تواند از آنها برای سنکرون شدن با فرستنده استفاده کند.

آخرین روش فریم بندی فقط در شبکه هایی قابل بکارگیری است که در کدگذاری لایه فیزیکی آنها نوعی افزونگی (redundancy) وجود داشته باشد. برای مثال در برخی از شبکه های LAN هر بیت داده با دو بیت فیزیکی نمایش داده می شود: بیت 1 با زوج بالا-پائین، و بیت 0 با زوج پائین-بالا. بدین ترتیب هر بیت داده دارای نوعی تغییر ولتاژ است، که تشخیص آنرا برای گیرنده ساده تر می کند. در چنین شبکه هایی زوج بالا-پائین و پائین-پائین برای داده ها استفاده نمی شود، و می توان از آنها برای مشخص کردن محدوده فریمها سود برد.

لازم به ذکر است که در بسیاری از پروتکل های لینک داده برای اطمینان بیشتر از ترکیب روش شمارش کاراکترها با یکی دیگر از تکنیکهای گفته شده استفاده می شود. در این روش، انتهای فریم با استفاده از فیلد تعداد کاراکترها مشخص می شود، ولی فقط زمانی مورد قبول قرار می گیرد که جمع تطبیقی فریم نیز معتبر بوده و در این نقطه طرح بیت پایان فریم وجود داشته باشد. اگر چنین نباشد، گیرنده طرح بیت پایان فریم را در نقاط دیگر جستجو خواهد کرد.

۳-۱-۳ کنترل خطا

بعد از حل مسئله ابتدا و انتهای فریمها، نوبت به مسئله بعدی می رسد: چگونه می توان تمام فریمها را سالم و با ترتیب صحیح به مقصد رساند؟ فرض کنید فرستنده فقط فریمها را می فرستد و کاری ندارد که آنها به مقصد می رسند یا خیر. این وضعیت برای سرویسهای غیر متصل بدون تصدیق دریافت خوب است، ولی برای سرویسهای قابل اعتماد (مانند سرویس اتصال-گرا یا تصدیق دریافت) مسلماً خوب نیست.

یک سرویس قابل اعتماد باید بنحوی از رسیدن بسته ها به مقصد و آنچه در آنجا اتفاق می افتد، مطلع شود. معمولاً در این موارد پروتکل درخواست می کند که یک فریم کنترلی خاص (که محتوی تصدیق یا عدم تصدیق

دریافت صحیح فریمهاست) به فرستنده باز پس فرستاده شود. اگر فرستنده تصدیق مثبت دریافت کند، مطمئن می شود که فریم به سلامت به مقصد رسیده است. اما تصدیق منفی نشان می دهد که اوضاع روبراه نیست، و فریم باید مجدداً فرستاده شود.

مشکل دیگر اینجاست که گاهی (در اثر اشکالات سخت افزاری) یک فریم بکلی گم و ناپدید می شود. در این حالت گیرنده هیچ عکس العملی نشان نمی دهد، چون اساساً چیزی نگرفته که عکس العمل نشان دهد. بروشنی پیداست که در این حالت پروتکل سمت فرستنده نا اید منتظر دریافت تصدیق از گیرنده می شود، تصدیقی که هرگز نخواهد رسید.

این مشکل را می توان با تعبیه یک تایمر در لایه پیوند داده حل کرد. وقتی فرستنده فریمی را می فرستد، تایمر را هم راه اندازی می کند. زمانی که این تایمر اندازه می گیرد آنقدر طولانی هست که بتوان با اطمینان گفت «فریم باید به مقصد رسیده، و تصدیق دریافت آن برگشته باشد». اگر همه چیز خوب پیش رفته باشد، معمولاً قبل از اینکه زمان تایمر به انتها برسد، تصدیق دریافت فریم به فرستنده برمی گردد (و تایمر ریست می شود).

اگر فریم یا پاسخ آن در راه گم شوند، تایمر در انتهای زمان مقرر اخطار می دهد؛ و ساده ترین راه حل همانا ارسال مجدد فریم است. اما ارسال چندباره فریمها این خطر را در بر دارد که چند تا از این فریمهای یکسان به مقصد برسند و به لایه شبکه تحویل داده شوند. برای اجتناب از این وضعیت، فرستنده به هر فریم یک شماره ترتیبی می دهد تا گیرنده بتواند فریمهای مشابه و تکراری را تشخیص دهد.

با تمام این تمهیدات (تایمر و شماره ترتیبی فریمها) می توان مطمئن بود که از هر فریم یک (و فقط یک) نسخه به لایه شبکه می رسد - و این یکی از مهمترین وظایف لایه پیوند داده است. در ادامه این فصل خواهید دید که لایه پیوند داده چگونه این وظیفه را انجام می دهد.

۴-۱-۳ کنترل جریان

یکی دیگر از مسائل مهم در طراحی لایه پیوند داده (ولایه های بالاتر) اینست که با فرستنده هایی که سریعتر از توان دریافت گیرنده مبادرت به ارسال اطلاعات می کنند، چه باید کرد؟ اگر کامپیوتر طرف فرستنده قویتر از گیرنده باشد (و یا بار کاری کمتری داشته باشد)، این وضعیت براحتی می تواند پیش بیاید. در این حالت گیرنده در سیلاب فریمهای ارسالی از فرستنده غرق می شود. حتی اگر کانال ارتباطی کاملاً عاری از خطا باشد، لحظه ای می رسد که گیرنده دیگر قادر به پردازش فریمهای ارسال شده نیست، و برخی از آنها را از دست می دهد. روشن است که باید کاری برای جلوگیری از این وضعیت کرد.

دو رهیافت برای مقابله با این وضعیت به کار گرفته می شود. در رهیافت اول، که کنترل جریان بر اساس بازخور (feedback-based flow control) نام دارد، این گیرنده است که آمادگی خود را برای دریافت اطلاعات بیشتر به فرستنده اعلام می کند (یا حداقل اعلام می کند در چه وضعیتی است). در رهیافت دوم، کنترل جریان بر اساس نرخ (rate-based flow control)، پروتکل مکانیزمی دارد که بدون استفاده از بازخور گیرنده نرخ ارسال اطلاعات را محدود می کند. در این فصل با روشهای کنترل جریان بر اساس بازخور آشنا می شوید، ولی از آنجائیکه رهیافت دوم هرگز در لایه پیوند داده کاربرد ندارد، توضیح درباره آنرا به فصل ۵ موکول می کنیم.

انواع مختلفی از کنترل جریان بر اساس بازخور وجود دارد، ولی همه آنها اصول مشترکی دارند: پروتکل قواعد تعریف شده ای دارد که زمان ارسال فریم بعدی را مشخص می کنند. طبق این قواعد فرستنده نمی تواند فریم بعدی را بفرستد، مگر اینکه (بطور صریح یا ضمنی) اجازه گیرنده را دریافت کرده باشد. مثلاً، وقتی اتصال برقرار می شود، گیرنده می تواند به فرستنده بگوید: «اکنون می توانی ۸ فریم بفرستی، ولی بعد از آن تا اجازه نداده ام چیزی نفرست.»

۲-۳ کشف و تصحیح خطا

همانطور که در فصل ۲ دیدید، سیستم تلفن سه بخش عمده دارد: سونیچها، ترانکها، و حلقه های محلی. در اکثر کشورهای توسعه یافته دو بخش اول تماماً دیجیتال هستند. اما قسمت اعظم حلقه های محلی کماکان آنالوگ است، که بایستی با صرف هزینه های هنگفت در آینده به دیجیتال تبدیل شود. با اینکه در بخش دیجیتال خطا پندرت روی می دهد، نرخ آن در حلقه های محلی آنالوگ همچنان بالاست. علاوه بر آن، مخابرات بیسیم نیز سرعت گسترش می باید، که نرخ خطا در این قبیل سیستمها چندین برابر کانالهای فیبر نوری است. نتیجه اخلاقی: فعلاً تا مدتها باید با خطاهای انتقال در سیستمهای مخابراتی بسازیم. در این قسمت خواهید دید چگونه.

خصلت خطا به منبع آن بستگی دارد؛ برای مثال، در سیستمهای رادویی خطا بصورت فورانی (burst) رخ می دهد، نه تکی. این نوع خطا مزایا و معایبی دارد. توجه داشته باشید که کامپیوترها اطلاعات خود را بصورت بسته ای ارسال می کنند. اگر هر بسته داده ۱۰۰۰ بیت و نرخ خطا نیز ۱ در ۱۰۰۰ باشد، می توان انتظار داشت که (در حالت غیرفوزانی) تقریباً تمام بسته ها با خطا به مقصد برسند. اما اگر خطا بصورت فورانه ای ۱۰۰ بیتی رخ دهد، بطور متوسط فقط یک یا دو بسته را خراب خواهد کرد. عیب بزرگ خطاهای فورانی آنست که کشف و تصحیح خطا در آنها بسیار دشوارتر است.

۱-۲-۳ گندهای تصحیح خطا

طراحان شبکه دو استراتژی کلی برای مقابله با خطاهای توسعه داده اند. یک راه اضافه کردن اطلاعات پراکنده به هر بلوک از داده هاست، بطوریکه گیرنده بتواند داده واقعی را از آن استنتاج کند. در روش دیگر فقط آنقدر اطلاعات اضافی به داده اصلی اضافه می شود که گیرنده از وقوع یا عدم وقوع خطا آگاهی یابد، و در صورت لزوم تکرار ارسال را خواستار شود. استراتژی اول گندهای تصحیح خطا (error-correcting codes) و استراتژی دوم گندهای کشف خطا (error-detecting codes) نام دارند. به کاربرد گندهای تصحیح خطا اغلب تصحیح پیشگیرانه خطا نیز گفته می شود.

هر یک از این تکنیکها جایگاه خاص خود را دارند. در کانالهای قابل اطمینان، مانند فیبر نوری، مقرون بصرفه تر است که از گندهای کشف خطا استفاده کرده و بسته های معدودی را که خراب می شوند، دوباره ارسال کنیم. اما در کانالهایی مانند لینکهای بیسیم که پر از خطا هستند، بهتر است از تکنیکهای تصحیح خطا استفاده کرده و اجازه دهیم گیرنده خود داده واقعی را بدست آورد (چون با احتمال زیاد ارسال مجدد بسته ها هم عاری از خطا نخواهد بود). برای مقابله با خطاهای، ابتدا باید بدانیم خطا واقعاً چیست. معمولاً، یک فریم m بیت داده اصلی (یعنی، پیام) و r بیت داده پراکنده (یا اطلاعات چک کننده) دارد، که در مجموع n بیت می شود ($n = m + r$). به این واحد n بیتی (داده های اصلی و پراکنده) اغلب کلمه کد n بیتی گفته می شود.

دو کلمه کد 10001001 و 10110001 را در نظر بگیرید: براحتی می توان مشخص کرد که این دو کلمه چند اختلاف دارند. در این مورد ۳ بیت اختلاف وجود دارد. برای تعیین تعداد اختلافها می توان دو کلمه کد را با هم XOR (OR انحصاری) کرد، و تعداد 1 ها را شمرد:

$$\begin{array}{r} 10001001 \\ 10110001 \\ \hline 00111000 \end{array}$$

به تعداد اختلافهای دو کلمه کد فاصله همینگ (Hamming distance) گفته می شود (Hamming, 1950). اهمیت این فاصله در آنجاست که می توان ثابت کرد برای تبدیل شدن اتفاقی دو کلمه با فاصله d ، بایستی d خطای تکبیتی روی دهد.

در اکثر سیستمهای انتقال، تمامی 2^m حالت ممکنه داده اصلی مجاز است، ولی بدلیل روش محاسبه بیت های افزونگی، تمام 2^n حالت کلمه کُد مجاز نیست. با توجه به الگوریتم محاسبه بیت های افزونگی، می توان لیستی از تمام حالت های مجاز کلمه کُد بدست آورد، و از این لیست دو کلمه ای که کمترین فاصله همینگ را دارند، پیدا کرد. این فاصله فاصله همینگ الگوریتم یا کُد مورد نظر است.

خصوصیات تصحیح خطا یا کشف خطای یک کُد به فاصله همینگ آن بستگی دارد. برای کشف d خطا، به کُدی با فاصله همینگ $d + 1$ نیاز داریم، چون با چنین کُدی هیچ d خطای تک بیتی وجود ندارد که بتواند یک کلمه کُد مجاز را به کلمه کُد مجاز دیگر تبدیل کند. اگر گیرنده کلمه کُد غیر مجازی دریافت کرد، می تواند با اطمینان بگوید که خطایی رخ داده است. بهمین ترتیب، برای تصحیح d خطا، به کُدی با فاصله $2d + 1$ نیاز داریم، چون در این حالت کلمات کُد چنان از هم فاصله دارند که حتی با بروز d خطا، کلمه کُد خراب شده هنوز نزدیکترین فاصله را با کلمه کُد اصلی دارد، و تشخیص آن براحتی ممکن است.

بعنوان نمونه ای از کُدهای کشف خطا، کُدی با یک بیت توازن (parity bit) را در نظر بگیرید. این بیت توازن بگونه ای انتخاب می شود که تعداد بیت های 1 کلمه کُد همواره زوج (یا فرد) شود. برای مثال، اگر بخواهیم کلمه 1011010 را با توازن زوج (even parity) ارسال کنیم، یک بیت 0 به انتهای آن اضافه می کنیم (10110100)؛ اما اگر بخواهیم همین کلمه را با توازن فرد (odd parity) ارسال کنیم، باید یک بیت 1 به انتهای آن اضافه کنیم (10110101). کُدی با یک بیت توازن دارای فاصله همینگ 2 است، چون هر خطای تک بیتی کلمه کُدی با توازن اشتباه تولید می کند. این کُد می توان یک خطا در هر کلمه را آشکار کند.

بعنوان یک نمونه ساده از کُدهای تصحیح خطا، کُدی را در نظر بگیرید که فقط چهار کلمه کُد مجاز دارد:

0000000000, 0000011111, 1111100000, 1111111111

فاصله همینگ این کُد 5 است، بنابراین می تواند دو خطا را تصحیح کند. اگر گیرنده کلمه کُدی بصورت 0000000111 دریافت کند، می داند که کلمه اصلی باید 0000011111 بوده باشد. اما اگر سه خطا کلمه 0000000000 را به 0000000111 تبدیل کرده باشد، دیگر نمی توان آنرا بدرستی تصحیح کرد.

فرض کنید می خواهیم کُدی با m بیت داده اصلی و r بیت افزونگی طراحی کنیم که بتواند تمام خطاهای تک بیتی را تصحیح کند. هر یک از 2^m پیام مجاز دارای n کلمه کُد غیر مجاز است که با آن 1 فاصله دارد (این را می توان پسادگی از معکوس کردن هر یک از بیت های کلمه کُد n بیتی فهمید). بنابراین هر یک از 2^m پیام مجاز به طرحی اختصاصی با $n + 1$ بیت نیاز دارد. از آنجائیکه تعداد ترکیبات ممکنه کلمه کُد 2^n است، بایستی داشته باشیم: $2^n \leq (n + 1)2^m$. با قرار دادن $n = m + r$ در این رابطه، داریم: $2^n \leq (m + r + 1)2^m$. با داشتن m ، از این رابطه حداقل بیت های افزونگی لازم (r) برای تصحیح خطاهای تک بیتی بدست می آید.

همینگ در یکی از مقالات خود (1950) روشی برای بدست آوردن این حداقل معرفی کرد. وی بیت های کلمه کُد را از چپ براسست شماره گذاری کرد. بیت هایی که توانایی از 2 هستند (1، 2، 4، 8، 16، و غیره)، بیت های چک کننده اند؛ سایر بیت ها (3، 5، 6، 7، 9، و غیره) بیت های پیام (m) هستند. هر بیت چک کننده توازن مجموعه ای از بیت (از جمله خودش) را زوج (یا فرد) می کند. هر بیت می تواند در بیش از یک مجموعه توازن محاسبه شود. برای دیدن اینکه کدام بیت های چک کننده در محاسبه توازن بیت داده ای در موقعیت k دخالت دارند، k را بصورت مجموع توانهای 2 می نویسیم. برای مثال، $11 = 1 + 2 + 8$ ، و $29 = 1 + 4 + 8 + 16$. هر بیت فقط با بیت های چک کننده ای که در موقعیت های بدست آمده از مجموع توانهای 2 قرار دارند، چک می شود (مثلاً، بیت موقعیت 11 فقط با بیت های چک کننده 1، 2 و 8 چک می شود).

وقتی یک کلمه کُد به گیرنده می رسد، گیرنده یک شمارنده را 0 می کند. سپس تمام بیت های چک کننده (k) را

کاراکتر	ASCII	بیت های چک کننده
H	1001000	00110010000
a	1100001	10111001001
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	01111001111
c	0100000	10011000000
o	1100011	11111000011
d	1101111	10101011111
e	1100100	11111001100
	1100101	00111000101

ترتیب انتقال بیت ها

شکل ۳-۷. استفاده از کد همینگ برای تصحیح خطاهای فورانی.

از نظر توازن چک می کند ($k = 1, 2, 4, 8, \dots$). اگر توازن k درست نباشد، گیرنده k را به شمارنده اضافه می کند. اگر پس از پایان این عملیات شمارنده همچنان 0 باشد، کلمه کد صحیح تلقی و قبول می شود. اگر شمارنده 0 نباشد، حتماً شماره بیت خطا را نشان می دهد. برای مثال، اگر توازن بیت های چک کننده 1، 2 و 8 اشتباه باشد، بیت 11 غلط است، چون این تنها بیتی است که با بیت های چک کننده 1، 2 و 8 چک می شود. در شکل ۳-۷ چند کاراکتر آسکی ۷-بیتی را که با کد همینگ ۱۱-بیتی کد شده اند، می بینید. فراموش نکنید که داده های اصلی در موقعیتهای 3، 5، 6، 7، 9، 10 و 11 قرار دارند.

کدهای همینگ فقط می توانند خطاهای تک بیتی را تصحیح کنند. با این حال روشی وجود دارد که اجازه می دهد تا این کد خطاهای فورانی را نیز تصحیح کند. در این روش k کلمه کد متوالی بصورت ماتریس (یک کلمه کد در هر سطر) چیده می شوند. معمولاً، این کلمات تک به تک (از چپ بر راست) ارسال می شوند. برای تصحیح خطاهای فورانی، بایستی داده ها را بصورت ستونی (باز هم از چپ بر راست) ارسال کرد. وقتی k بیت اول (ستون اول) ارسال شد، نوبت به ستون دوم (و سپس ستونهای بعدی) می رسد (شکل ۳-۷ را ببینید). وقتی این فریم به گیرنده رسید، ماتریس از نو (ستون به ستون) ساخته می شود. اگر یک خطای فورانی به طول k رخ داده باشد، حداکثر یک بیت در هر کلمه کد تغییر خواهد کرد، و از آنجائیکه کد همینگ می تواند یک خطا را تصحیح کند، تمام بلوک قابل تصحیح خواهد بود. در این روش برای مصون کردن k بیت داده در مقابل خطاهای فورانی با طول k (یا کمتر)، از k^2 بیت چک کننده استفاده شده است.

۲-۲-۳ کدهای کشف خطا

کدهای کشف خطا در لینکهای بیسیم، که در مقایسه با سیم مسی و فیبر نوری بطور وحشتناکی نویزی هستند، کاربرد گسترده ای دارد. بدون این کدها شاید اساساً نتوان چیزی روی این لینکها رد و بدل کرد. اما در سیمهای مسی و فیبرهای نوری نرخ خطا بسیار کمتر است، و تشخیص خطا و ارسال مجدد بسته هایی که (ندرتاً) خراب می شوند، کاملاً کفایت می کند.

بعنوان مثال، کانالی را در نظر بگیرید که نرخ خطا در آن 1 در 10^6 و خطاها غیر فورانی هستند؛ اندازه هر بلوک را هم 1000 بیت فرض می کنیم. برای داشتن ویژگی تصحیح خطا، هر بلوک ۱۰۰۰ بیتی به ۱۰ بیت چک کننده نیاز دارد، بعبارت دیگر برای ارسال 1 Mb داده باید 10 kb اطلاعات افزونگی (بیت های چک کننده) را نیز به همراه آن

بفرستیم. اما برای کشف خطا فقط یک بیت توازن در هر بلوک کافیتست. در این روش بار اضافی کشف خطا + ارسال مجدد یک بلوک خراب برای 1 Mb داده فقط 2001 بیت است، که در مقایسه با 10,000 بیت کُد همینگ بسیار کمتر است.

اگر در هر بلوک از یک بیت توازن برای کشف خطا استفاده کنیم و یک خطای فورانی رخ دهد، احتمال اینکه بتوانیم خطا را کشف کنیم فقط ۵۰٪ است، که بهیچوجه قابل قبول نیست. اما با تشکیل ماتریسی با n ستون و k سطر (که در بالا توضیح دادیم) اوضاع بنحو قابل توجهی بهتر خواهد شد. در این روش برای هر ستون یک بیت توازن محاسبه، و در آخرین سطر ماتریس نوشته می شود. هنگام ارسال نیز این ماتریس بصورت ستونی فرستاده می شود. گیرنده بعد از دریافت کل ماتریس، تمام بیت های توازن را چک می کند؛ و اگر هر یک از این بیت ها غلط باشد، ارسال مجدد ماتریس را درخواست می کند. این کار تا زمانی که ماتریس بطور کامل و بدون خطای توازن به دست گیرنده برسد، تکرار خواهد شد.

روش فوق می تواند خطاهای فورانی با طول حداکثر n بیت را آشکار کند، چون در این حالت فقط یک بیت در هر ستون تغییر خواهد کرد. اما اگر یک خطای فورانی با طول $1 + n$ رخ دهد بگونه ای که فقط بیت اول و آخر را تغییر دهد (و سایر بیت ها تغییر نکنند)، نمی توان آنرا کشف کرد، زیرا بیت اول و آخر در یک سطر قرار می گیرند و توازن این سطر بدون تغییر خواهد ماند. (یک خطای فورانی الزاماً بمعنای معکوس شدن تمام بیت ها نیست: فقط می توان از معکوس شدن بیت اول و آخر مطمئن بود.) اگر طول خطای فورانی خیلی زیاد باشد یا تعدادی خطای فورانی کوتاه و پشت سر هم رخ دهد، احتمال اینکه یکی از ستونها تصادفاً صاحب توازن درست شود، ۵۰٪ است، بنابراین احتمال اینکه چنین بلوکی (به اشتباه) صحیح تلقی شود، 2^{-n} خواهد بود.

با اینکه روش فوق در مواردی کفایت می کند، اما در عمل از روش دیگری استفاده می شود: کُد چندجمله ای (polynomial code)، که به CRC (چک افزونگی چرخه ای - Cyclic Redundancy Check) نیز معروفست. در کدهای چندجمله ای مبنای این است که هر رشته یک چندجمله ایست با ضرایب 0 و 1. با این فرض، یک فریم k -بیتی معادلت با عبارتی k جمله ای، با ضرایب x^{k-1} تا x^0 . این چندجمله ای از درجه $k-1$ است. با ارزشترین بیت (منتهی الیه سمت چپ) ضریب x^{k-1} است، بیت بعدی ضریب x^{k-2} ، و الی آخر. برای مثال، رشته 110001 دارای 6 بیت است بنابراین نشاندهنده یک شش جمله ایست با ضرایب 1، 1، 0، 0، 0، و 1، که می توان آنرا چنین نوشت: $x^5 + x^4 + x^0$.

محاسبات چندجمله ایها در مدول 2 (و طبق قوانین جبر میدان) انجام می شود. در جمع و تفریق 2 بر 1 نادیده گرفته می شود، بعبارت دیگر شبیه XOR است. برای مثال،

$$\begin{array}{r} 10011011 \\ + 11001010 \\ \hline 01010001 \end{array} \quad \begin{array}{r} 00110011 \\ + 11001101 \\ \hline 11111110 \end{array} \quad \begin{array}{r} 11110000 \\ - 10100110 \\ \hline 01010110 \end{array} \quad \begin{array}{r} 01010101 \\ - 10101111 \\ \hline 11111010 \end{array}$$

تقسیم درست مانند تقسیم باینری است، با این تفاوت که تفریق ها در مدول 2 (مانند بالا) انجام می شود. در هنگام استفاده از روش کُد چندجمله ای، فرستنده و گیرنده بایستی از قبل بر سر یک چندجمله ای مولد (generator polynomial)، که آنرا $G(x)$ می نامیم، توافق کنند. با ارزشترین (چپ ترین) و کم ارزشترین (راست ترین) بیت های چندجمله ای مولد باید 1 باشد. برای محاسبه مجموع چک (checksum) یک فریم m -بیتی (که چندجمله ای متناظر با آن $M(x)$ است)، این فریم باید طولانیتر از چندجمله ای مولد باشد. ایده آنست که یک مجموع چک به انتهای فریم اصلی چسبانده شود، بگونه ای که فریم حاصله بر $G(x)$ قابل تقسیم باشد. اگر

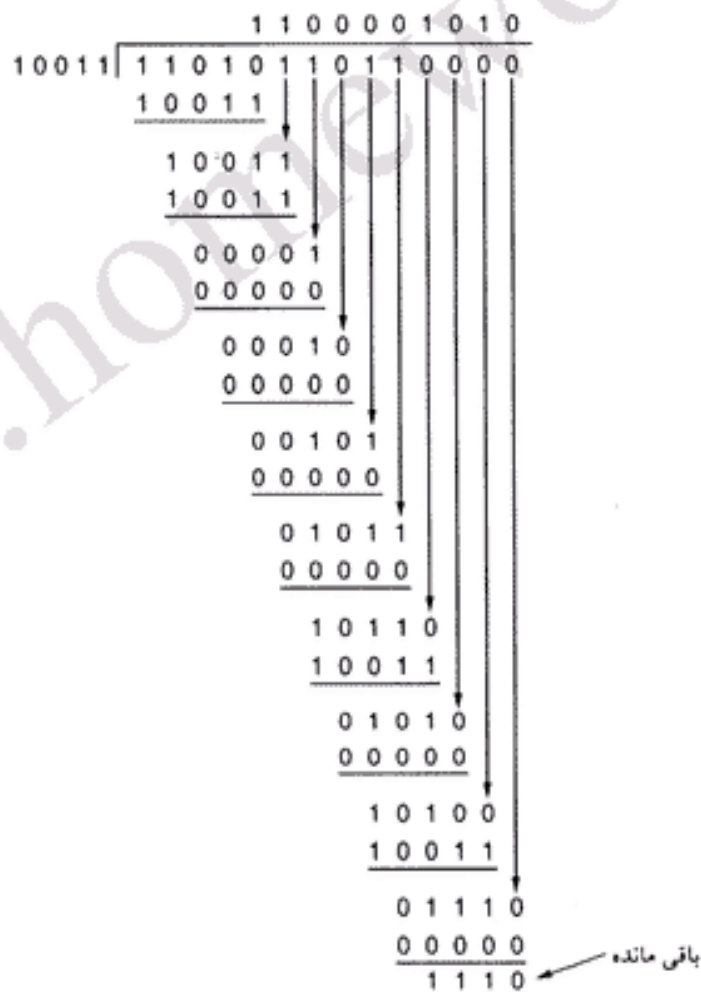
تقسیم این فریم بر $G(x)$ در سمت گیرنده باقیمانده آورد، معلوم می شود که خطایی رخ داده است. الگوریتم محاسبه مجموع چک چنین است:

۱. فرض می کنیم چندجمله ای $G(x)$ از درجه r است. r بیت 0 به سمت راست فریم اضافه می کنیم تا تعداد بیت های آن به $m + r$ برسد. این چندجمله ای معادل $x^r M(x)$ خواهد شد.
 ۲. رشته $x^r M(x)$ را (در مدول 2) بر $G(x)$ تقسیم می کنیم.
 ۳. باقیمانده را (که همیشه r بیت یا کمتر دارد) از $x^r M(x)$ کم می کنیم (این تفریق هم در مدول 2 انجام می شود). حاصل تفریق همان فریم موردنظر (فریم اولیه + مجموع چک) است، که آنرا $T(x)$ می نامیم.
- در شکل ۸-۳ طرز محاسبه مجموع چک برای فریم 1101011011 را با مولد $G(x) = x^4 + x + 1$ ملاحظه می کنید.

فریم: 1101011011

مولد: 10011

پیام بعد از اضافه شدن ۴ بیت: 11010110110000



فریم اضافه شده: 11010110111110

شکل ۸-۳ محاسبه مجموع چک کد چندجمله ای.

همانطور که براحتی معلوم می شود، $T(x)$ بر $G(x)$ (در مدول 2) بخش پذیر است (چون وقتی باقیمانده تقسیم را از مقسوم کم کنیم، عدد حاصله بطور حتم بر مقسوم علیه بخش پذیر خواهد بود). بطور مثال، اگر 210,278 را (در مبنای 10) بر 10,941 تقسیم کنیم، باقیمانده 2399 می شود که اگر آنرا از 210,278 کم کنیم، آنچه باقی می ماند (207,879) ، بر 10,941 بخش پذیر خواهد بود.

حال اجازه دهید قدرت این روش را بررسی کنیم. این روش چه نوع خطاهایی را می تواند کشف کند؟ فرض کنید خطایی رخ داده، و بجای $T(x)$ رشته $T(x) + E(x)$ به گیرنده رسیده است، بطوریکه هر بیت 1 در $E(x)$ متناظر با یک بیت تغییر یافته است (بعبارت دیگر، اگر در این عبارت، $E(x)$ ، k بیت 1 وجود داشته باشد، k خطای تکبیتی رخ داده است). خطای فورانی نیز عبارتست از خطایی که با یک بیت 1 شروع و ختم شود، و بین آنها هر ترکیبی از 0 و 1 می تواند وجود داشته باشد.

وقتی این فریم به مقصد می رسد، گیرنده آنرا بر $G(x)$ تقسیم می کند (بعبارت دیگر $[T(x)+E(x)]/G(x)$ را محاسبه می کند). از آنجائیکه $T(x)/G(x) = 0$ ، این تقسیم معادل $E(x)/G(x)$ است. همانطور که می بینید، اگر خطای رخ داده دقیقاً طرحی شبیه $G(x)$ نداشته باشد، بطور مسلم آشکار خواهد شد.

فرض کنید یک خطای تکبیتی رخ داده است، یعنی، $E(x) = x^i$ (که i بیت خطاست). اگر $G(x)$ بیش از دو جمله داشته باشد، $E(x)$ هرگز بر آن بخش پذیر نخواهد بود - پس، این روش می تواند تمام خطاهای تکبیتی را آشکار کند.

اگر دو خطای تکبیتی جدا از هم رخ دهد، بطوریکه $E(x) = x^i + x^j$ (که در آن $i > j$)، می توان $E(x)$ را به صورت $x^j(x^{i-j} + 1)$ تجزیه کرد. اگر $G(x)$ بر x بخش پذیر نباشد، شرط کافی برای اینکه تمام خطاهای دوبیتی قابل کشف باشد آن است که $E(x)$ عبارت $x^k + 1$ را (برای تمام k های کوچکتر از $i - j$) بخش نکند. چندجمله ای های ساده و از درجه پائینی می شناسیم که می توان با آنها فریمهای نسبتاً طولیل را محافظت کرد. مثلاً، چندجمله ای $1 + x^{14} + x^{15}$ هیچ عبارت $x^k + 1$ را برای تمام k های کوچکتر از 32,768 بخش نمی کند.

اگر تعداد خطاهای رخ داده عددی فرد باشد، تعداد جملات $E(x)$ نیز فرد خواهد بود (برای مثال، تعداد جملات $1 + x^2 + x^5$ فرد است، ولی $1 + x^2 + x^5$ چنین نیست). جالبست بدانید که هیچ چندجمله ای با تعداد جملات فرد وجود ندارد که (در مدول 2) بر $x + 1$ بخش پذیر باشد. بدین ترتیب اگر $G(x)$ را طوری انتخاب کنیم که بر $x + 1$ بخش پذیر باشد، می توانیم هر خطایی که تعداد بیتهای تغییر کرده فرد باشد را کشف کنیم.

برای اثبات اینکه هیچ چندجمله ای فرد وجود ندارد که بر $x + 1$ بخش پذیر باشد، فرض کنید $E(x)$ چندجمله ای فردیست که چنین خاصیتی دارد (بر $x + 1$ بخش پذیر است). اگر از $x + 1$ فاکتور بگیریم، $E(x)$ بصورت $Q(x)(x + 1)$ در می آید. حال $E(1) = (1 + 1)Q(1)$ را محاسبه می کنیم. از آنجائیکه (در مدول 2) $1 + 1 = 0$ ، $E(1)$ باید 0 باشد. اما اگر تعداد جملات $E(x)$ فرد باشد، قرار دادن 1 بجای x در آن همیشه نتیجه 1 می دهد. بنابراین فرض ما نمی تواند درست باشد، و هیچ چندجمله ای فرد بر $x + 1$ بخش پذیر نیست.

بالاخره، و از همه مهمتر، یک کد چندجمله ای با r بیت چک کننده تمام خطاهای فورانی با طول کمتر یا مساوی r را آشکار می کند. یک خطای فورانی با طول k را می توان با $1 + x^{k-1} + \dots + x^{k-1}$ نشان داد، که در آن نقطه شروع خطای فورانی از سمت راست فریم است. اگر مولد $G(x)$ دارای جمله x^0 باشد، بر x^k بخش پذیر نخواهد بود؛ بنابراین اگر درجه عبارت داخل پراتر از درجه $G(x)$ کمتر باشد، باقیمانده تقسیم هرگز نمی تواند 0 شود.

اگر طول خطای فورانی $r + 1$ باشد، باقیمانده تقسیم بر $G(x)$ صفر می شود فقط و فقط اگر طرح بیت خطا با $G(x)$ یکسان باشد. طبق تعریف بیت های اول و آخر خطای فورانی باید 1 باشند، بنابراین یکسان بودن آنها به $r - 1$ بیت میانی بستگی دارد. اگر تمام ترکیبات این $r - 1$ بیت را یکسان فرض کنیم، احتمال بروز این وضعیت $\frac{1}{2^{r-1}}$ خواهد بود.

همچنین می توان نشان داد که اگر طول خطای فورانی از $1 + 2^r$ بزرگتر باشد یا چند خطای فورانی کوتاهتر رخ دهد، احتمال کشف نشدن خطا (با فرض یکسان بودن تمام ترکیبات ممکنه) $\frac{1}{2^r}$ است.

برخی از چندجمله ایها بصورت استاندارد بین المللی در آمده اند، که از میان آنها می توان به چندجمله ای زیر (که در IEEE 802 از آن استفاده می شود) اشاره کرد:

$$1 + x^1 + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{12} + x^{16} + x^{22} + x^{23} + x^{26} + x^{32}$$

از ویژگیهای جالب این چندجمله ای آن است که هر نوع خطای فورانی با طول 32 یا کمتر، و خطاهای فورانی که تعداد پشتهای تغییر کرده فرد باشد، را آشکار می کند.

با اینکه بنظر می رسد محاسبه مجموع چک و تست آن کار پیچیده ای باشد، پترسون و براون (1961) نشان دادند که می توان این کار را با یک مدار شیفت رجیستر (shift register) ساده بصورت سخت افزاری انجام داد. در واقع، این مدار در تمام کارتهای شبکه تعبیه شده است، و بسیاری از خطوط نقطه به نقطه هم از آن استفاده می کنند.

برای مدتهای مدید تصور بر آن بود که فریمهایی که مجموع چک آنها محاسبه می شود دارای طرح بیت تصادفی هستند، و تمام الگوریتمهای محاسبه مجموع چک نیز فرض را بر این می گذاشتند. اما بررسی دقیق داده های واقعی نشان داده که این فرض بکلی اشتباه است. در نتیجه، خطاهایی که (تحت شرایط خاص) کشف نشده می مانند شایعتر از آن چیزیست که قبلاً تصور می شد (Partridge et al., 1995).

۳-۳ چند پروتکل ساده لینک داده

برای آشنایی با پروتکل های لایه پیوند داده، در این قسمت سه پروتکل را (که بتدریج پیچیده تر می شوند) بررسی خواهیم کرد. برای خوانندگان علاقمند، شبیه ساز این پروتکلها (و پروتکلهایی که در آینده خواهید دید) را در سایت وب کتاب قرار داده ایم (<http://www.prenhall.com/tanenbaum>). اما قبل از اینکه سراغ این پروتکلها برویم، اجازه دهید چند تا از فرض هایی را که درباره مدل ارتباطی زیربنایی داشته ایم، توضیح دهیم. اول اینکه فرض کرده ایم در لایه های فیزیکی، لایه پیوند داده و شبکه پروسسهای هستند که مستقل از یکدیگرند، و ارتباط آنها از طریق رد و بدل کردن پیام صورت می گیرد. در بسیاری از موارد، پروسسهای لایه فیزیکی و لینک داده در پردازنده کارت شبکه اجرا می شوند، و پروسسهای لایه شبکه در CPU اصلی کامپیوتر. البته پیاده سازیهای دیگری نیز محتمل است (مثلاً، تمام پروسسهای لایه فیزیکی، لینک داده و شبکه در پردازنده کارت شبکه اجرا شوند، یا همگی آنها را CPU اصلی اجرا کند). در هر حال، مستقل دانستن این پروسسها بحث درباره آنها را بسیار ساده تر می کند، و تأکیدست بر مستقل بودن لایه ها.

فرض کلیدی دیگر اینست که، ماشین A با استفاده از یک سرویس اتصال-گرای قابل اعتماد استریم طولی از داده ها را به ماشین B می فرستد. بعدها این حالت که B هم همزمان به A داده بفرستد، را بررسی خواهیم کرد. علاوه بر آن، فرض کرده ایم ماشین A منبع بی پایانی از داده ها دارد که ارسال کند، و هرگز منتظر آمدن داده ها نخواهد شد. بعبارت دیگر، هر گاه لایه پیوند داده A درخواست داده کند، لایه شبکه بلافاصله اجابت می کند. (بعدها این فرض را هم کنار خواهیم گذاشت.)

فرض دیگر ما اینست که این کامپیوترها هرگز از کار نمی افتند؛ یعنی، پروتکلهای ما فقط با خطاهای مخابراتی سروکار دارند، نه هنگ کردن کامپیوتر و مسائلی از این قبیل.

دیگر اینکه، تا آنجا که به لایه پیوند داده مربوط است، بسته ای که به لایه شبکه داده می شود، داده خالص است، و باید تا آخرین بیت به آن تحویل شود. این که بخشی از این داده ها سرآیند بسته هستند و لایه شبکه آنها را دور می ریزد، به خودش مربوط است نه به لایه پیوند داده.

وقتی لایه پیوند داده بسته ای از لایه شبکه می گیرد تا ارسال کند، آنرا فریم بندی کرده و سرآیند (header) و پی آیند (trailer) های لازم را به آن می چسباند (شکل ۳-۱ را ببینید). بنابراین هر فریم از سه بخش تشکیل می شود: قسمتی از بسته ای که لایه شبکه فرستاده، یک سرآیند شامل اطلاعات کنترلی، و یک پی آیند شامل مجموع چک فریم. سپس این فریم به لایه پیوند داده ماشین مقصد فرستاده می شود. فرض ما بر این است که روال های کتابخانه ای *to_physical_layer* (برای ارسال) و *from_physical_layer* (برای دریافت) از قبل وجود دارند. مجموع چک نیز (مثلاً، با استفاده از گدهای چندجمله ای) توسط سخت افزار محاسبه (و به انتهای فریم اضافه) می شود، بنابراین لازم نیست لایه پیوند داده نگران آن باشد.

در ابتدا، لازم نیست گیرنده کاری انجام دهد؛ فقط منتظر می ماند تا اتفاقی بیفتد. در مثالهای این فصل، فرض کرده ایم که لایه پیوند داده این کار را با روالی بنام *wait_for_event(&event)* انجام می دهد. این روال فقط وقتی به پایان می رسد (و کنترل را به برنامه اصلی برمی گرداند) که اتفاقی افتاده باشد (یعنی، یک فریم دریافت شده باشد). اینکه چه اتفاقی افتاده است، را متغیر *event* مشخص می کند؛ و این که چه اتفاقی می تواند بیفتد، به تعریف پروتکل بستگی دارد. توجه داشته باشید که در دنیای واقعی لایه پیوند داده (مانند این مثالها) در یک حلقه بی انتها منتظر رسیدن فریمها نمی ماند، بلکه با استفاده از وقفه (*interrupt*) به آنها رسیدگی می کند. با این حال برای اجتناب از پیچیدگی مطلب، فرض کرده ایم که لایه پیوند داده هیچ کار دیگری جز رسیدگی به کانال ارتباطی ما ندارد.

وقتی یک فریم به گیرنده می رسد، سخت افزار مجموع چک آنرا محاسبه می کند. اگر این مجموع چک اشتباه باشد (یعنی خطایی رخ داده)، به لایه پیوند داده اطلاع داده می شود (*event = cksun_err*). اگر فریم بدرستی دریافت شده باشد، باز هم به اطلاع لایه پیوند داده می رسد (*event = frame_arrival*). در این حالت لایه پیوند داده با استفاده از تابع *from_physical_layer* فریم را گرفته، اطلاعات کنترلی موجود در سرآیند آنرا چک می کند، و اگر همه چیز مرتب باشد، سرآیند را جدا کرده و بخش اصلی داده را به لایه شبکه تحویل می دهد.

تحت هیچ شرایطی سرآیند فریم به لایه شبکه تحویل نمی شود، و برای این کار دلیل خوبی وجود دارد. پروتکل های لینک داده و شبکه باید کاملاً از یکدیگر مستقل باشند. مستقل بودن پروتکل های این دو لایه باعث می شود که بتوان هر کدام از این پروتکلها را تغییر داد، بدون اینکه نیاز باشد پروتکل های لایه دیگر تغییر کند (البته تحت هر شرایطی نحوه تعامل و ارتباط لایه ها نباید تغییر کند). جدا و مستقل بودن لایه ها تا حد زیادی طراحی آنها را ساده می کند، چون می توان بدون نگرانی از اتفاقاتی که در لایه های دیگر می افتد، روی طراحی عملکردهای همان لایه تمرکز کرد.

در شکل ۳-۹ مقداری تعریف (به زبان C) می بینید، که در طراحی پروتکل های این قسمت به آنها نیاز داریم. در اینجا پنج ساختار داده تعریف شده است: *boolean*، *seq_nr*، *packet*، *frame_kind* و *frame*. ساختار *boolean* از نوع شمارشی (enum) است، و می تواند دو مقدار *true* و *false* بگیرد. ساختار *seq_nr* از نوع عدد صحیح بدون علامت (*unsigned int*) تعریف شده، و برای شماره گذاری فریمها از آن استفاده خواهیم کرد. شماره گذاری فریمها از ۰ تا *MAX_SEQ* (که بسته به نیاز هر پروتکل تعریف می شود) انجام می گیرد. *packet* (بسته) واحدی از اطلاعات است که بین لایه شبکه و لایه پیوند داده (روی یک ماشین، یا روی ماشین های جداگانه) رد و بدل می شود. در مدل ما هر بسته همیشه حاوی *MAX_PKT* بایت داده است، ولی به واقعیت نزدیکتر است که طول بسته را متغیر در نظر بگیریم.

هر *frame* از چهار فیلد تشکیل شده: *kind*، *seq*، *ack* و *info* - که سه تای اول اطلاعات کنترلی هستند، و آخری همان داده هائیکه باید منتقل شود. به مجموعه فیلدهای کنترلی سرآیند فریم (frame header) گفته می شود.

```

#define MAX_PKT 1024 /* determines packet size in bytes */

typedef enum {false, true} boolean; /* boolean type */
typedef unsigned int seq_nr; /* sequence or ack numbers */
typedef struct {unsigned char data[MAX_PKT];} packet; /* packet definition */
typedef enum {data, ack, nak} frame_kind; /* frame_kind definition */

typedef struct { /* frames are transported in this layer */
    frame_kind kind; /* what kind of a frame is it? */
    seq_nr seq; /* sequence number */
    seq_nr ack; /* acknowledgement number */
    packet info; /* the network layer packet */
} frame;

/* Wait for an event to happen; return its type in event. */
void wait_for_event(event_type *event);

/* Fetch a packet from the network layer for transmission on the channel. */
void from_network_layer(packet *p);

/* Deliver information from an inbound frame to the network layer. */
void to_network_layer(packet *p);

/* Go get an inbound frame from the physical layer and copy it to r. */
void from_physical_layer(frame *r);

/* Pass the frame to the physical layer for transmission. */
void to_physical_layer(frame *s);

/* Start the clock running and enable the timeout event. */
void start_timer(seq_nr k);

/* Stop the clock and disable the timeout event. */
void stop_timer(seq_nr k);

/* Start an auxiliary timer and enable the ack_timeout event. */
void start_ack_timer(void);

/* Stop the auxiliary timer and disable the ack_timeout event. */
void stop_ack_timer(void);

/* Allow the network layer to cause a network_layer_ready event. */

```

```
void enable_network_layer(void);
```

```
/* Forbid the network layer from causing a network_layer_ready event. */
```

```
void disable_network_layer(void);
```

```
/* Macro inc is expanded in-line: Increment k circularly. */
```

```
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0
```

شکل ۳-۹. تعریف های مورد نیاز برای پروتکل هایی که در این فصل می نویسیم. این تعریف ها در فایل بنام *protocol.h* قرار داده می شوند.

فیلد *kind* می گوید که آیا داده ای در فریم وجود دارد یا خیر، چون برخی از پروتکل ها فریم های بدون داده را از فریم هایی که داده دارند، تمیز می دهند. فیلدهای *seq* و *ack* بترتیب برای شماره ترتیبی فریم و تصدیق دریافت مورد استفاده قرار می گیرند؛ بعداً در این باره بیشتر توضیح خواهیم داد. داده اصلی (بسته) در فیلد *info* فریم قرار دارد؛ فریم های کترلی نیز وجود دارند که اساساً در آنها فیلد *info* وجود ندارد. در پروتکل های واقعی که طول فیلد *info* می تواند متغیر باشد، نیازی به تمایز بین فریم های داده و فریم های کترلی نیست (چون فریم کترلی فریمی است که طول فیلد *info* در آن ۰ است).

در اینجا لازم است تفاوت فریم (*frame*) و بسته (*packet*) را مجدداً یادآور شویم. لایه شبکه با گرفتن پیام از لایه انتقال و اضافه کردن سرآیند، آنرا بصورت بسته در می آورد. سپس این بسته به لایه پیوند داده تحویل می شود، که در آنجا در فیلد *info* یک فریم قرار داده شده و برای ارسال آماده می شود. وقتی این فریم به لایه پیوند داده ماشین مقصد رسید، بسته از فیلد *info* استخراج شده و به لایه شبکه تحویل می شود. این فرآیند بکلی شفاف است، و لایه های شبکه در دو ماشین متقابل تصور می کنند که مستقیماً در حال تبادل بسته اند.

در برنامه شکل ۳-۹ چند روال (تابع) نیز تعریف شده است. اینها روال های کتابخانه ای هستند، که فقط کاری که انجام می دهند برای ما مهم است (و اصلاً اهمیتی ندارد این کار را چگونه انجام می دهند). روال *wait_for_event* (همانطور که قبلاً گفتیم) در یک حلقه بی انتها به انتظار می ماند تا اتفاقی بیفتد. روال های *to_network_layer* و *from_network_layer* بترتیب برای ارسال بسته به لایه شبکه و برای گرفتن بسته از این لایه بکار می روند (اینها واسطه لایه های ۲ و ۳ هستند). برای تبادل اطلاعات با لایه فیزیکی نیز از روال های *to_physical_layer* و *from_physical_layer* استفاده می شود (اینها واسطه لایه های ۱ و ۲ هستند).

در پروتکل های واقعی فرض بر اینست که کانال ارتباطی نامطمئن است، و احتمال این هست که فریم ها در راه از بین بروند. برای مقابله با چنین وضعیتی، لایه پیوند داده همزمان با ارسال هر فریم، باید یک تایمر را راه اندازی کند. اگر بعد از مدتی معین پاسخی از طرف مقابل نرسید، تایمر مزبور لایه پیوند داده را (با استفاده از یک وقفه) مطلع می کند.

در پروتکل های ما، در چنین وضعیتی روال *wait_for_event* مقدار *event = timeout* برمی گرداند. روال های *start_timer* و *stop_timer* نیز بترتیب تایمر را روشن و خاموش می کنند (البته سپری شدن زمان مقتضی - *timeout* - فقط زمانی اتفاق می افتد که تایمر روشن باشد). یک تایمر را می توان (قبل از منقضی شدن آن) با اجرای مجدد روال *start_timer* ریست کرد.

روال های *start_ack_timer* و *stop_ack_timer* تایمر دیگری را (برای ایجاد فریم های تصدیق دریافت در شرایطی خاص) کنترل می کنند.

از روالهای `enable_network_layer` و `disable_network_layer` در پروتکل‌های پیچیده‌تر استفاده می‌شود (ما در پروتکل‌های ساده این قسمت از این روالها استفاده نخواهیم کرد، چون فرض کرده‌ایم که لایه شبکه همیشه می‌تواند به لایه پیوند داده بسته تحویل دهد). وقتی لایه پیوند داده لایه شبکه را فعال می‌کند (`enable_network_layer`)، لایه شبکه اجازه دارد آماده شدن بسته داده را با یک وقفه به لایه پیوند داده اطلاع دهد (این کار با `event = network_layer_ready` انجام می‌شود). اگر لایه شبکه غیرفعال باشد (`disable_network_layer`)، اجازه چنین کاری را ندارد. با استفاده دقیق و بسجا از روالهای `enable_network_layer` و `disable_network_layer`، لایه پیوند داده می‌تواند مطمئن شود که (هنگام پر شدن بافر) در سیلاب بسته‌های ارسالی از لایه شبکه غرق نخواهد شد.

شماره ترتیبی فریم همیشه بین 0 تا `MAX_SEQ` (و از جمله خود این دو عدد) است، که البته `MAX_SEQ` در پروتکل‌های مختلف می‌تواند متفاوت باشد. شماره ترتیبی فریمها معمولاً یکی یکی اضافه می‌شود، و وقتی به `MAX_SEQ` رسید، دوباره 0 خواهد شد - این کار بر عهده ماکرو `inc` گذاشته شده است. برای سرعت بخشیدن به اجرای این عملیات، `inc` بصورت ماکرو (`macro`) تعریف شده است. [کامپایلر با دیدن یک ماکرو، دستور معادل را جایگزین آن می‌کند، و مانند تابع آنرا فراخوانی نمی‌کند. م.] همانطور که بعداً خواهید دید، سرعت اجرای پروتکلها یکی از عوامل کلیدی در کارایی شبکه است، و استفاده از ماکرو (بجای تابع) تأثیر زیادی بر بهبود این کارایی دارد. همچنین، از آنجائیکه `MAX_SEQ` در پروتکل‌های مختلف مقادیر متفاوتی دارد، تعریف `inc` بصورت ماکرو امکان می‌دهد تا بدون هیچ مشکلی از آن در پروتکل‌های مختلف استفاده کنیم.

تعاریف شکل ۳-۹ بخشی از پروتکل‌هاییست که در قسمتهای آینده خواهیم نوشت. البته می‌توانستیم آنها را در ابتدای هر پروتکل نیز بیاوریم، ولی با جمع کردن آنها در یک فایل کدهای آینده بسیار ساده‌تر خواهند شد. در زبان C، این کار با استفاده از دستور `#include` و نوشتن نام فایل تعاریف (در اینجا `protocol.h`) انجام می‌شود.

۱-۳-۳ پروتکل یکطرفه نامقید

در اولین مثال یک پروتکل بسیار ساده را در نظر می‌گیریم، که در آن داده‌ها فقط در یک جهت منتقل می‌شوند. لایه شبکه در فرستنده و گیرنده آماده کار هستند، زمان پردازش را می‌توان نادیده گرفت، از نظر بافر هیچ کمبودی وجود ندارد، و مهمتر از همه اینکه کانال ارتباطی بین دو لایه پیوند داده کامل و بدون نقص است، و هیچ خطایی در آن رخ نمی‌دهد. این پروتکل غیرواقعی (که شاید «اتوپیا» - پروتکل آرمانی - مناسبترین نام برای آن باشد) را در شکل ۳-۱۰ ملاحظه می‌کنید.

```
/* Protocol 1 (utopia) provides for data transmission in one direction only, from
sender to receiver. The communication channel is assumed to be error free
and the receiver is assumed to be able to process all the input infinitely quickly.
Consequently, the sender just sits in a loop pumping data out onto the line as
fast as it can. */
```

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"
```

```
void sender1(void)
{
```

```

frame s;                                /* buffer for an outbound frame */
packet buffer;                          /* buffer for an outbound packet */

while (true) {
    from_network_layer(&buffer);        /* go get something to send */
    s.info = buffer;                    /* copy it into s for transmission */
    to_physical_layer(&s);              /* send it on its way */
}                                       /* Tomorrow, and tomorrow, and tomorrow,
                                   Creeps in this petty pace from day to day
                                   To the last syllable of recorded time.
                                   - Macbeth, V, v */
}

void receiver1(void)
{
    frame r;
    event_type event;                   /* filled in by wait, but not used here */

    while (true) {
        wait_for_event(&event);         /* only possibility is frame_arrival */
        from_physical_layer(&r);        /* go get the inbound frame */
        to_network_layer(&r.info);      /* pass the data to the network layer */
    }
}

```

شکل ۳-۱۰. پروتکل یکطرفه نامقید.

این پروتکل دارای دو روال مجزا است: فرستنده و گیرنده. فرستنده در لایه پیوند داده ماشین مبدأ، و گیرنده در لایه پیوند داده ماشین مقصد اجرا می شود. در اینجا از شماره ترتیبی فریمها و تصدیق دریافت استفاده ای نمی شود، بنابراین به MAX_SEQ هم نیازی نیست. تنها رویداد قابل انتظار $frame_arrival$ (رسیدن صحیح و سالم فریم) است.

فرستنده یک حلقه بی انتهای `while` است که داده ها را با حداکثر توان به بیرون پمپ می کند. بدنه این حلقه سه کار انجام می دهد: آوردن یک بسته از لایه شبکه (که همیشه آماده خدمت است)، ایجاد فریم خروجی با استفاده از متغیر s ، و فرستادن فریم. این پروتکل فقط از فیلد $info$ فریم استفاده می کند، چون فیلدهای دیگر مربوط به کنترل جریان و خطا هستند، که طبق فرض ما چنین محدودیتهایی در اینجا وجود ندارد.

گیرنده هم بهمان اندازه ساده است: انتظار بی پایان برای دریافت فریمی که همیشه سالم و بی نقص است. وقتی یک فریم از راه رسید، روال `wait_for_event` کنترل را به برنامه اصلی برمی گرداند، و متغیر $event$ را به $frame_arrival$ ست می کند (که بهر حال استفاده ای از آن نمی شود). با فراخوانی روال `from_physical_layer`، فریم تازه از راه رسیده از بافر سخت افزاری برداشته شده و در متغیر r قرار داده می شود (تا گیرنده می تواند آنرا بردارد). در پایان، بخش داده این فریم (فیلد $info$) به لایه شبکه فرستاده شده، و لایه پیوند داده به انتظار فریم بعدی می نشیند.

۲-۳-۳ پروتکل توقف-انتظار یکطرفه

حال اجازه دهید غیر واقعی ترین بخش از پروتکل ۱ را کنار بگذاریم: نامحدود بودن توانایی این پروتکل در دریافت بسته ها از لایه شبکه، و پردازش فریمهای ورودی (که به معنای نامحدود بودن بافر لایه پیوند داده در سمت گیرنده است). اما کانال ارتباطی را همچنان بدون خطا، و ارتباط را یکطرفه فرض کرده ایم.

مهمترین مشکلی که با آن روبرو هستیم، این است که چگونه از غرق شدن گیرنده در سیلاب فریمهایی (که پردازش آنها از توان وی خارج است) جلوگیری کنیم. واضحست که، اگر گیرنده برای اجرای روالهای *from_physical_layer* و *to_network_layer* به زمان Δt نیاز داشته باشد، سرعت متوسط ارسال فرستنده بایستی از یک فریم بر Δt کمتر باشد. همچنین اگر فرض کنیم که سخت افزار گیرنده بطور خودکار عمل بافر کردن فریمها را انجام نمی دهد، فرستنده نباید قبل از برداشته شدن یک فریم از بافر لایه فیزیکی (که توسط روال *from_physical_layer* انجام می شود)، فریم بعدی را ارسال کند چون در غیر اینصورت فریم قبلی از بین می رود (به این حالت روهم نویسی - overrun - می گویند).

در شرایط خاصی (مانند ارتباط سنکرون، و گیرنده ای که تنها وظیفه آن گرفتن اطلاعات از خط ورودی است)، با ایجاد تأخیر در قسمت فرستنده پروتکل ۱ و کند کردن آن می توان به اهداف فوق دست یافت. اما بسیار محتملتر است که یک لایه پیوند داده مجبور باشد چندین خط ورودی را پردازش کند، که در این حالت فاصله زمانی دریافت فریمها و پردازش آنها می تواند بسیار متغیر باشد. اگر طراحان شبکه بتوانند بدترین حالت گیرنده را محاسبه کنند، می توانند فرستنده را آنقدر کند کنند که روهم نویسی هرگز اتفاق نیفتد. اما این روش بسیار محافظه کارانه است و بنحو بسیار بدی پهنای باند را تلف می کند، مگر اینکه تفاوت بهترین و بدترین حالت چندان زیاد نباشد (یعنی تفاوت پاسخهای لایه پیوند داده ناچیز باشد).

راه حل بهتر این معضل، برگرداندن بازخور (feedback) از گیرنده به فرستنده است. بعد از تحویل بسته به لایه شبکه، گیرنده یک فریم کوچک (که لازم نیست معنی خاصی هم داشته باشد) به فرستنده می فرستد، که در واقع مجوز ارسال فریم بعدی محسوب می شود. فرستنده بعد از ارسال یک فریم، آنقدر منتظر می ماند تا این فریم کوچک (که در واقع همان تصدیق دریافت - acknowledgement - است) از راه برسد. استفاده از بازخور گیرنده برای اطلاع به فرستنده (و دادن مجوز ارسال فریمهای بعدی) یکی از نمونه های کنترل جریان (flow control)، که قبلاً به آن اشاره کردیم، است.

پروتکلهایی که در آنها فرستنده قبل از ارسال فریم بعدی منتظر تصدیق دریافت فریم قبلی از گیرنده می ماند، به پروتکلهای توقف-انتظار (stop-and-wait) معروفند. در شکل ۳-۱۱ یک نمونه از پروتکلهای توقف-انتظار را ملاحظه می کنید.

/* Protocol 2 (stop-and-wait) also provides for a one-directional flow of data from sender to receiver. The communication channel is once again assumed to be error free, as in protocol 1. However, this time, the receiver has only a finite buffer capacity and a finite processing speed, so the protocol must explicitly prevent the sender from flooding the receiver with data faster than it can be handled. */

```
typedef enum {frame_arrival} event_type;
```

```
"h.locotorp" edulcni#
```

```

void sender2(void)
{
    frame s;                                /* buffer for an outbound frame */
    packet buffer;                          /* buffer for an outbound packet */
    event_type event;                       /* frame_arrival is the only possibility */

    while (true) {
        from_network_layer(&buffer);        /* go get something to send */
        s.info = buffer;                    /* copy it into s for transmission */
        to_physical_layer(&s);              /* bye-bye little frame */
        wait_for_event(&event);             /* do not proceed until given the go ahead */
    }
}

void receiver2(void)
{
    frame r, s;                             /* buffers for frames */
    event_type event;                       /* frame_arrival is the only possibility */
    while (true) {
        wait_for_event(&event);             /* only possibility is frame_arrival */
        from_physical_layer(&r);            /* go get the inbound frame */
        to_network_layer(&r.info);          /* pass the data to the network layer */
        to_physical_layer(&s);              /* send a dummy frame to awaken sender */
    }
}

```

شکل ۳-۱۱. پروتکل توقف-انتظار یکطرفه.

با اینکه این پروتکل یکطرفه (simplex) است (یعنی ما فقط در یک جهت ارسال می‌کنیم)، اما فریمها می‌توانند در هر دو جهت رفت و آمد کنند. برای این منظور لازم است کانال ارتباطی ما چنین قابلیت‌هایی داشته باشد؛ البته یک کانال دوطرفه ناهمزمان (half-duplex) هم کفایت می‌کند، چون فرستنده و گیرنده در آن واحد اقدام به فرستادن فریمها نمی‌کنند: فرستنده یک فریم می‌فرستد، گیرنده پاسخ می‌دهد، فرستنده فریم بعدی را می‌فرستد، گیرنده پاسخ می‌دهد، و الی آخر.

در اینجا هم (مانند پروتکل ۱) فرستنده همان سه کار قبلی را انجام می‌دهد: آوردن یک بسته از لایه شبکه، ایجاد فریم خروجی، و فرستادن آن. اما برخلاف پروتکل ۱، قبل از ادامه کار (آوردن بسته بعدی و ارسال آن در قالب یک فریم) باید منتظر رسیدن فریم تصدیق دریافت از گیرنده بماند. نیازی نیست که لایه پیوند داده فرستنده فریم دریافتی را بررسی کند، چون فقط یک احتمال وجود دارد: این فریم همیشه تصدیق دریافت گیرنده است. تنها تفاوت receiver2 با receiver1 این است که بعد از تحویل بسته به لایه شبکه و قبل از ورود به حالت انتظار، receiver2 یک فریم تصدیق دریافت به فرستنده باز پس می‌فرستد. از آنجائیکه فقط خود فریم مهم است و نه محتویات آن، گیرنده هیچ داده‌ای در فیلد info این فریم قرار نمی‌دهد.

۳-۳-۳ پروتکل یکطرفه برای کانالهای نویزدار

اکنون به یک حالت واقعی تر می پردازیم: کانالهایی که نویز دارند. فریمها می توانند با خطا به مقصد برسند، و یا بکلی گم شده و اصلاً به مقصد نرسند. با این حال، فرض می کنیم که اگر فریمی با خطا به مقصد رسید، سخت افزار لایه فیزیکی با محاسبه جمع تطبیقی متوجه خطا می شود. پروتکل ما در یک حالت به اشتباه عمل خواهد کرد: خطای رخ داده آنقدر شدید باشد که جمع تطبیقی تصادفاً درست از کار در آید (اتفاقی که بسیار نامحتمل است). در نگاه اول با یک تغییر کوچک در پروتکل ۲ (اضافه کردن یک تایمر) می توان آن را با وضعیت جدید تطبیق داد. فرستنده می تواند در هر زمانی یک فریم بفرستد، ولی گیرنده فقط وقتی فریم تصدیق دریافت را برمی گرداند که این فریم را بدرستی دریافت و پردازش کرده باشد. اگر فریم ناقص به مقصد برسد، دور انداخته خواهد شد. بعد از مدتی تایمر فرستنده به انتها می رسد، و چون هنوز تصدیق دریافت گیرنده را نگرفته، مجدداً اقدام به ارسال فریم می کند. این ماجرا تا زمانی که فریم به سلامت به مقصد برسد، تکرار خواهد شد. اما طرح بالا یک مشکل اساسی دارد. قبل از خواندن ادامه کتاب، کمی فکر کنید و ببینید می توانید متوجه اشکال آن شوید.

برای درک مشکل، بیاد بیاورید که وظیفه برقراری یک کانال ارتباطی عاری از خطا بین لایه های شبکه بر عهده لایه پیوند داده است. لایه شبکه ماشین A یک سری بسته به لایه پیوند داده می دهد، و باید مطمئن باشد که این بسته ها به همان ترتیبی که ارسال شده اند بدست لایه شبکه ماشین B خواهند رسید. بویژه، لایه شبکه ماشین B هیچ راهی ندارد تا بفهمد که یک بسته گم شده یا تکراریست. به همین دلیل لایه پیوند داده ماشین B باید تضمین کند که هیچ بسته ای گم نمی شود، و یا تکراری نیست. سناریوی زیر را در نظر بگیرید:

۱. لایه شبکه ماشین A بسته ۱ را به لایه پیوند داده می دهد. این بسته صحیح و سالم به لایه پیوند داده ماشین B می رسد، و تحویل لایه شبکه می شود. ماشین B یک فریم تصدیق دریافت به A می فرستد.
۲. فریم تصدیق دریافت ماشین B در راه از بین می رود، و هرگز به A نمی رسد. اگر فقط فریمهای داده گم می شدند و این اتفاق برای فریمهای کنترلی نمی افتاد، زندگی چقدر شیرین تر بود! ولی متأسفانه کانالهای مخابراتی اهل تبعیض نیستند!
۳. تایمر لایه پیوند داده A به انتها می رسد، و چون هیچ فریم تصدیق دریافتی بدستش نرسیده، (باشتابه) تصور می کند که فریم به مقصد نرسیده (یا خراب شده)، پس آنرا دوباره می فرستد.
۴. فریم تکراری (در کمال صحت و سلامت) به لایه پیوند داده B می رسد، و این لایه هم (بی خبر از همه جا) آنرا به لایه شبکه تحویل می دهد. تصور کنید که اگر A در حال ارسال یک فایل به B باشد، تکراری بودن بخشی از آن چه فاجعه ای پیر خواهد آورد. همانطور که می بینید، پروتکل ما یک شکست کامل است.

چیزی که ما به آن احتیاج داریم، وسیله ایست که بتوان فریمهای تکراری را از فریمهایی که برای اولین بار دریافت می شوند، تشخیص داد. راه حل واضح این مشکل آن است که فرستنده یک شماره ترتیبی (sequence number) در سرآیند فریمهایی که می فرستد، قرار دهد. گیرنده می تواند این شماره را چک کرده، و فریمهای تکراری را دور بیندازد.

از آنجائیکه سرآیند یک فریم باید حتی الامکان کوچک باشد، سؤالی که پیش می آید اینست که: حداقل تعداد بیتهای لازم برای فیلد شماره ترتیبی چندتا است؟ در پروتکل ما تنها ابهام در فریم m و فریم بعدی آن یعنی $m + 1$ است. اگر فریم m خراب شود یا از بین برود، گیرنده فریم تصدیق دریافت آنرا بر نمی گرداند، پس فرستنده سعی می کند آنرا دوباره بفرستد. همین که این فریم سالم به مقصد رسید، گیرنده فریم تصدیق دریافت را به فرستنده پس می فرستد. همین جاست که مشکل بروز می کند: اگر فریم تصدیق دریافت صحیح و سالم به فرستنده بفرستد،

فرستنده فریم بعدی (یعنی $m + 1$) را می فرستد، در غیر اینصورت فریم m را خواهد فرستاد. برای ارسال فریم $m + 2$ ، فرستنده باید قبلاً تصدیق دریافت فریم $m + 1$ را گرفته باشد. اما این بدان معناست که فریم m به سلامت به مقصد رسیده و تصدیق دریافت آن هم بدرستی به فرستنده برگشت داده شده است (چون در غیر اینصورت فرستنده فریم $m + 1$ را هم نمی فرستاد، چه رسد به فریم $m + 2$). بنابراین، تنها ابهامی که می تواند وجود داشته باشد، بین یک فریم و فریم بعدی آن است. برای تشخیص این دو هم یک شماره ترتیبی یک بیتی (0 یا 1) کافیست. عبارت دیگر، در هر لحظه گیرنده باید بدنبال شماره بعدی باشد. اگر فریمی با شماره اشتباه دریافت شد، گیرنده آنرا تکراری تلقی کرده و دور می اندازد. اما اگر شماره ترتیبی فریم درست بود، به لایه شبکه تحویل داده می شود. با این توصیف فیلد شماره ترتیبی باید در مدول 2 افزایش داده شود (بعبارت دیگر، 1 به 0 تبدیل می شود، و 0 به 1).

پروتکلی با این مشخصات را در شکل ۳-۱۲ ملاحظه می کنید. به پروتکل هایی که فرستنده برای ارسال فریم بایستی منتظر یک تصدیق دریافت مثبت بماند، PAR (تصدیق دریافت مثبت با ارسال مجدد - Positive Acknowledgement with Retransmission) یا ARQ (درخواست تکرار خودکار - Automatic Repeat reQuest) نیز گفته می شود. این پروتکل هم، مانند پروتکل ۲، فقط در یک جهت داده می فرستد. تفاوت پروتکل ۳ با دو نای قبلی اینست که، روالهای فرستنده و گیرنده متغیری دارند که مقدار آن حتی در زمانی که لایه پیوند داده به حالت انتظار می رود، دست نخورده باقی می ماند. فرستنده باید شماره ترتیبی فریم بعدی که می خواهد بفرستد، را بداند: `next_frame_to_send`؛ و گیرنده هم باید شماره ترتیبی فریم بعدی که باید منتظر آن باشد، را بداند: `frame_expected`.

```
/* Protocol 3 (par) allows unidirectional data flow over an unreliable channel. */
#define MAX_SEQ 1 /* must be 1 for protocol 3 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"
void sender3(void)
{
    seq_nr next_frame_to_send; /* seq number of next outgoing frame */
    frame s; /* scratch variable */
    packet buffer; /* buffer for an outbound packet */
    event_type event;
    next_frame_to_send = 0; /* initialize outbound sequence numbers */
    from_network_layer(&buffer); /* fetch first packet */
    while (true) {
        s.info = buffer; /* construct a frame for transmission */
        s.seq = next_frame_to_send; /* insert sequence number in frame */
        to_physical_layer(&s); /* send it on its way */
        start_timer(s.seq); /* if answer takes too long, time out */
    }
}
```

```

*/
    wait_for_event(&event);          /* frame_arrival, cksum_err, timeout
*/
    if (event == frame_arrival) {
        from_physical_layer(&s);      /* get the acknowledgement */
        if (s.ack == next_frame_to_send) {
            stop_timer(s.ack);        /* turn the timer off */
            from_network_layer(&buffer); /* get the next one to send */
            inc(next_frame_to_send);  /* invert next_frame_to_send */
        }
    }
}

void receiver3(void)
{
    seq_nr frame_expected;
    frame r, s;
    event_type event;
    frame_expected = 0;
    while (true) {
        wait_for_event(&event);          /* possibilities: frame_arrival,
cksum_err */
        if (event == frame_arrival) {    /* a valid frame has arrived. */
            from_physical_layer(&r);      /* go get the newly arrived frame
*/
            if (r.seq == frame_expected) { /* this is what we have been
waiting for. */
                to_network_layer(&r.info); /* pass the data to the network
layer */
                inc(frame_expected);      /* next time expect the other
sequence nr */
            }
            s.ack = 1 - frame_expected;    /* tell which frame is being acked
*/
            to_physical_layer(&s);        /* send acknowledgement */
        }
    }
}

```

شکل ۳-۱۲. پروتکل تصدیق دریافت مثبت با ارسال مجدد (PAR).

فرستنده، بعد از ارسال فریم، تایمر را راه می اندازد. اگر تایمر از قبل در حال اجرا باشد، این کار آنرا ریست کرده و آماده کار بعدی می کند. فاصله زمانی تایمر باید بگونه ای انتخاب شود که وقت کافی برای سه رویداد بدست دهد: رسیدن فریم به گیرنده، پردازش آن در گیرنده (در بدترین حالت)، و برگشت فریم تصدیق دریافت به فرستنده. فقط پس از سپری شدن این زمان است که می توان مطمئن شد فریم (یا تصدیق دریافت آن) به مقصد نرسیده، و فرستنده باید دوباره آنرا بفرستد. اگر فاصله زمانی تایمر کم انتخاب شود، تعداد دفعاتی که فرستنده فریم تکراری می فرستد افزایش یافته، و (با اینکه این کار تأثیر منفی روی گیرنده ندارد) به کارایی سیستم لطمه می زند. بعد از ارسال فریم و راه انداختن تایمر، فرستنده منتظر یک اتفاق هیجان انگیز می ماند. البته فقط سه احتمال برای چنین اتفاقی وجود دارد: فریم تصدیق دریافت صحیح و سالم از راه برسد، فریم تصدیق دریافت با خطا وارد شود، تایمر منقضی شود (زمان مشخص شده به انتها برسد). در حالت اول، فرستنده بسته دیگری از لایه شبکه گرفته، و در بافر خود (buffer) قرار می دهد. سپس، شماره ترتیبی فریم (next_frame_to_send) را بالا می برد. اما در دو حالت دیگر (خراب شدن فریم تصدیق دریافت یا نرسیدن آن)، بافر و شماره ترتیبی هیچکدام تغییری نمی کند، بنابراین در هیچ حالتی فریم تکراری فرستاده نخواهد شد. وقتی یک فریم به گیرنده می رسد، گیرنده شماره ترتیبی آن را چک می کند، تا از تکراری نبودن آن مطمئن شود. این فریم فقط در صورت تکراری نبودن به لایه شبکه تحویل داده می شود. بدین ترتیب، فریمهای تکراری و خراب به لایه شبکه نخواهند رسید.

۴-۳ پروتکل های پنجره لغزنده

در پروتکل های قبل، فریمهای داده فقط در یک جهت ارسال می شدند. اما در عمل باید بتوانیم در هر دو جهت انتقال داده داشته باشیم. یکی از راههای داشتن یک کانال دوطرفه همزمان (full-duplex) استفاده از دو کانال یکطرفه (simplex) در دو جهت مخالف است، که هر کدام فقط در یک جهت داده می فرستند (و البته فریمهای تصدیق دریافت می کنند). اما این روش چیزی جز اتلاف پهنای باند (و پول) نیست.

روش بهتر استفاده از یک کانال واحد برای ارسال داده در دو جهت است (مگر نه اینکه در پروتکل های ۲ و ۳ در هر دو جهت فریم ارسال کردیم). از آنجائیکه در این مدل، فریمهای داده و تصدیق دریافت در هر دو جهت می توانند فرستاده شوند، باید کاری کنیم که گیرنده بتواند آنها را از یکدیگر تشخیص دهد. برای این منظور می توانیم از فیلد info در سرآیند فریمها استفاده کنیم.

با اینکه ترکیب فریمهای داده و تصدیق دریافت روی یک مدار واحد (بجای دو مدار جداگانه) یک قدم به جلو محسوب می شود، اما باز هم می توان کارایی سیستم را بهبود بخشید. وقتی گیرنده یک فریم داده دریافت می کند، بجای اینکه بلافاصله یک فرم کنترلی پس بفرستد، منتظر می ماند تا بسته بعدی را برای ارسال از لایه شبکه بگیرد. تصدیق دریافت فریم قبلی در فیلد ack فریم داده ای که اکنون می خواهد فرستاده شود، قرار داده می شود، و در واقع فریم تصدیق دریافت از فریم داده سواری مجانی (piggyback) می گیرد (و به همین نام هم خوانده می شود). یکی از مزایای تکنیک سواری مجانی نسبت به ارسال مستقل فریمهای تصدیق دریافت، استفاده بهینه تر از پهنای باند موجود است: فیلد ack فقط چند بیت از سرآیند را اشغال می کند، در حالیکه یک فریم مستقل برای خود سرآیند، جمع تطبیقی، و تصدیق دریافت دارد. علاوه بر آن، هر چه تعداد فریمهایی که در یک جهت فرستاده می شوند کمتر باشد، گیرنده بهتر می تواند به کارهای دیگرش (پردازش فریمهای رسیده و خالی کرن بافرها) برسد، و این هم به بهبود کارایی سیستم کمک می کند. در پروتکلی که در این قسمت می نویسیم، فیلد سواری مجانی فقط یک بیت به سرآیند فریم اضافه می کند (و بندرت پیش می آید که مقدار آن از چند بیت بیشتر شود).

اما سواری مجانی هم خالی از اشکال نیست. برای مثال، در این حالت لایه پیوند داده گیرنده چقدر باید منتظر بسته از لایه شبکه خود شود؟ اگر خیلی منتظر بماند، تایمر فرستنده به انتها رسیده و فریم را تکرار می‌کند، که این نقض غرض (از ارسال فریم تصدیق دریافت) است. اگر لایه پیوند داده قدرت پیشگویی داشت، می‌توانست زمان دریافت بسته بعدی از لایه شبکه را پیشگویی کند، و آنوقت می‌توانست تصمیم بگیرد که منتظر این بسته بماند یا بلافاصله فریم تصدیق دریافت را به فرستنده بفرستد. اما متأسفانه لایه پیوند داده نمی‌تواند آینده را پیشگویی کند، پس باید روش ساده‌تری (انتظار بمدت ثابت، مثلاً چند میلی‌ثانیه) پیدا کنیم. اگر در این فاصله بسته‌ای از لایه شبکه رسید، لایه پیوند داده تصدیق دریافت را سوار آن می‌کند؛ در غیر اینصورت یک فریم مستقل تصدیق دریافت به سمت فرستنده ارسال می‌کند.

پروتکل‌هایی که در این قسمت خواهید دید، به کلاس پروتکل‌های پنجره لغزنده (sliding window) تعلق دارند، که فقط از نظر کارایی، پیچیدگی و بافر با هم تفاوت دارند. در این پروتکل‌ها، مانند سایر پروتکل‌های پنجره لغزنده، هر فریم خروجی یک شماره ترتیبی (از 0 تا یک حداکثر) دارد. حداکثر شماره فریم‌ها معمولاً $2^n - 1$ است، بنابراین در یک فیلد n -بیتی بخوبی جا می‌شود. پروتکل پنجره لغزنده توقف-انتظار از $n = 1$ استفاده می‌کرد، اما در پروتکل‌های دیگر این عدد می‌تواند بیشتر باشد.

ایده اصلی در تمام پروتکل‌های پنجره لغزنده این است که، فرستنده در هر لحظه از زمان لیستی از شماره‌های ترتیبی متناظر با فریم‌هایی که می‌تواند ارسال کند، در اختیار دارد. اصطلاحاً گفته می‌شود که این فریم‌ها در پنجره ارسال (sending window) قرار دارند. گیرنده هم یک پنجره دریافت (receiving window) دارد که متناظر است با فریم‌هایی که مجاز به دریافت آنهاست. الزامی نیست که پنجره ارسال و پنجره دریافت حد پائین و بالای مشابه داشته باشند، یا حتی هم‌اندازه باشند. در برخی پروتکل‌ها اندازه این پنجره‌ها ثابت است، ولی در پروتکل‌های دیگر می‌تواند کوچک یا بزرگ شود.

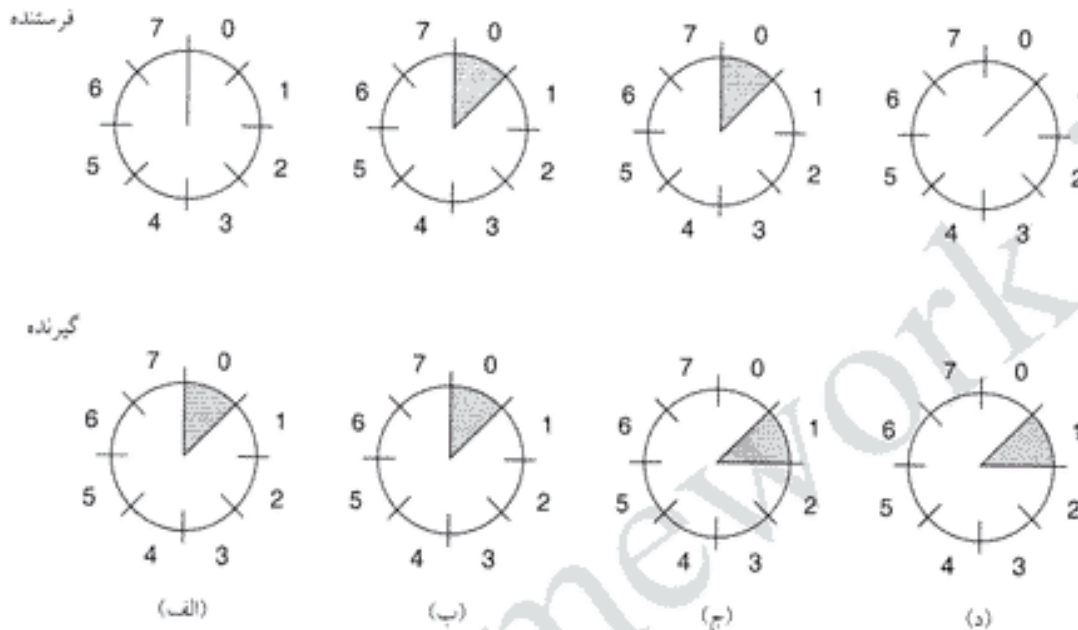
با اینکه این پروتکل‌ها آزادی عمل بیشتری به لایه پیوند داده در ارسال و دریافت فریم‌ها می‌دهند، لازم است مجدداً تأکید کنیم که تحویل بسته‌ها به لایه شبکه مقصد باید با همان ترتیبی صورت گیرد که در ماشین مبدأ تحویل لایه پیوند داده شده‌اند. (و یک بار دیگر خاطرنشان می‌کنیم که، لایه فیزیکی یک کانال ارتباطی ساده است که فریم‌ها را به همان ترتیبی که به آن داده شده، منتقل می‌کند.)

شماره‌های موجود در پنجره ارسال شماره فریم‌هاییست که باید ارسال شوند، و یا ارسال شده‌اند ولی هنوز تصدیق دریافت آنها برنگشته است. وقتی یک بسته جدید از لایه شبکه می‌رسد، لایه پیوند داده بالاترین شماره ترتیبی موجود را به آن می‌دهد، و لبه بالایی پنجره ارسال را یکی زیاد می‌کند. وقتی یک تصدیق دریافت وارد شد، لبه پائینی پنجره ارسال نیز بالا برده می‌شود. بدین ترتیب پنجره ارسال همیشه شامل لیست فریم‌هاییست که دریافت آنها هنوز توسط گیرنده تصدیق نشده است. در شکل ۳-۱۳ یک نمونه را ملاحظه می‌کنید.

از آنجائیکه امکان گم یا خراب شدن فریم‌هایی که در حال حاضر در پنجره ارسال قرار دارند، همیشه وجود دارد، فرستنده باید آنها را برای ارسال مجدد (احتمالی) در حافظه نگه دارد. بنابراین اگر حداکثر اندازه این پنجره n باشد، فرستنده باید بافری باندازه n فریم تصدیق نشده داشته باشد. اگر پنجره ارسال از حداکثر پیش‌بینی شده بزرگتر شود، لایه پیوند داده باید گرفتن بسته از لایه شبکه را تا زمان آزاد شدن بافر متوقف کند.

پنجره دریافت در گیرنده متناظر است با فریم‌هایی که گیرنده مجاز به دریافت آنهاست. هر فریمی که خارج از این پنجره قرار گیرد، بدون هیچ توضیحی دور انداخته خواهد شد. وقتی فریمی که شماره ترتیبی آن معادل لبه پائین پنجره دریافت است، از راه می‌رسد، به لایه شبکه تحویل شده و پس از ایجاد تصدیق دریافت آن، پنجره دریافت یک واحد می‌چرخد. بر خلاف پنجره فرستنده، پنجره گیرنده همیشه به همان اندازه اولیه می‌ماند. توجه

کنید که پنجره دریافت ۱ بمعنای اینست که لایه پیوند داده فریمها را فقط به ترتیب می پذیرد، ولی در پنجره های بزرگتر الزاماً چنین نیست (با این حال، لایه شبکه همیشه داده ها را به ترتیب صحیح تحویل لایه پیوند داده می دهد).



شکل ۳-۱۳. یک پنجره لغزنده یک واحدی، با شماره ترتیبی ۳-بیتی. (الف) در شروع کار. (ب) بعد از ارسال اولین فریم. (ج) بعد از آنکه اولین فریم دریافت شد. (د) بعد از آنکه فرستنده اولین تصدیق دریافت را گرفت.

در شکل ۳-۱۳ حداکثر اندازه پنجره ۱ است. در لحظه اول لبه های پائین و بالای پنجره ارسال یکی هستند، ولی با گذشت زمان موقعیت آنها طبق شکل تغییر می کند.

۳-۴-۱ پروتکل پنجره لغزنده ۱-بیتی

قبل از پرداختن به حالت کلی، اجازه دهید ابتدا پروتکلی با پنجره لغزنده ۱-بیتی را مورد بررسی قرار دهیم. در واقع این پروتکل نوعی پروتکل توقف-انتظار است، چون فرستنده قبل از گرفتن تصدیق دریافت فریم فرستاده شده، فریم بعدی را ارسال نخواهد کرد.

در شکل ۳-۱۴ پروتکل پنجره لغزنده ۱-بیتی را ملاحظه می کنید. مانند پروتکل های قبلی، این پروتکل هم با تعریف متغیرها شروع می شود. متغیر *next_frame_to_send* فریم بعدیست که فرستنده باید بفرستد. در طرف مقابل هم، متغیر *frame_expected* فریمی را نشان می دهد که گیرنده منتظر آن است. در هر دو طرف، تنها حالت های مجاز فقط ۰ یا ۱ است.

```
/* Protocol 4 (sliding window) is bidirectional. */
#define MAX_SEQ 1 /* must be 1 for protocol 4 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"
void protocol4 (void)
```

```

{
    seq_nr next_frame_to_send;           /* 0 or 1 only */
    seq_nr frame_expected;               /* 0 or 1 only */
    frame r, s;                          /* scratch variables */
    packet buffer;                       /* current packet being sent */
    event_type event;
    next_frame_to_send = 0;              /* next frame on the outbound
stream */
    frame_expected = 0;                  /* frame expected next */
    from_network_layer(&buffer);         /* fetch a packet from the network
layer */
    s.info = buffer;                     /* prepare to send the initial frame */
    s.seq = next_frame_to_send;          /* insert sequence number into
frame */
    s.ack = 1 - frame_expected;          /* piggybacked ack */
    to_physical_layer(&s);                /* transmit the frame */
    start_timer(s.seq);                  /* start the timer running */
    while (true) {
        wait_for_event(&event);          /* frame_arrival, cksum_err, or
timeout */
        if (event == frame_arrival) {    /* a frame has arrived
undamaged. */
            from_physical_layer(&r);      /* go get it */
            if (r.seq == frame_expected) { /* handle inbound frame
stream. */
                to_network_layer(&r.info); /* pass packet to network
layer */
                inc(frame_expected);      /* invert seq number expected
next */
            }
            if (r.ack == next_frame_to_send) { /* handle outbound frame
stream. */
                stop_timer(r.ack);        /* turn the timer off */
                from_network_layer(&buffer); /* fetch new pkt from network
layer */
                inc(next_frame_to_send);  /* invert sender's sequence
number */
            }
        }
        s.info = buffer;                  /* construct outbound frame */
        s.seq = next_frame_to_send;      /* insert sequence number into it

```

```

*/
    s.ack = 1 - frame_expected;          /* seq number of last received
frame */
    to_physical_layer(&s);                /* transmit a frame */
    start_timer(s.seq);                    /* start the timer running */
}
}

```

شکل ۳-۱۴. پروتکل پنجره لغزنده آبینی.

در شرایط عادی، یکی از دو طرف پیش دستی کرده و اولین فریم را می فرستد. بعبارت دیگر، فقط یکی از دو لایه پیوند داده روالهای `to_physical_layer` و `start_timer` را خارج از حلقه اصلی برنامه اجرا می کند. اگر در پیشامدی نادر هر دو طرف بطور همزمان شروع به ارسال اولین فریم کنند، وضعیت عجیبی پیش می آید، که بعداً آنرا توضیح خواهیم داد. ماشین شروع کننده اولین بسته را از لایه شبکه گرفته، یک فریم از آن می سازد، و سپس ارسال می کند. وقتی این فریم (یا هر فریم دیگری) به طرف مقابل رسید، لایه پیوند داده گیرنده چک می کند که تکراری نباشد (درست مثل پروتکل ۳). اگر این همان فریم موردنظر باشد، به لایه شبکه تحویل شده و پنجره دریافت یک واحد به جلو لغزانده می شود.

فیلد تصدیق دریافت حاوی شماره آخرین فریمیست که بدون خطا دریافت شده است. اگر این شماره با شماره فریمی که فرستنده در صدد ارسال آن است یکی باشد، فرستنده می فهمد که دیگر نیازی به فریم داخل بافر ندارد و می تواند بسته بعدی را از لایه شبکه بگیرد. اگر شماره ها یکی نباشند، فرستنده باید به ارسال همان فریم قبلی ادامه دهد. وقتی یک فریم دریافت شود، فریمی نیز پس فرستاده می شود.

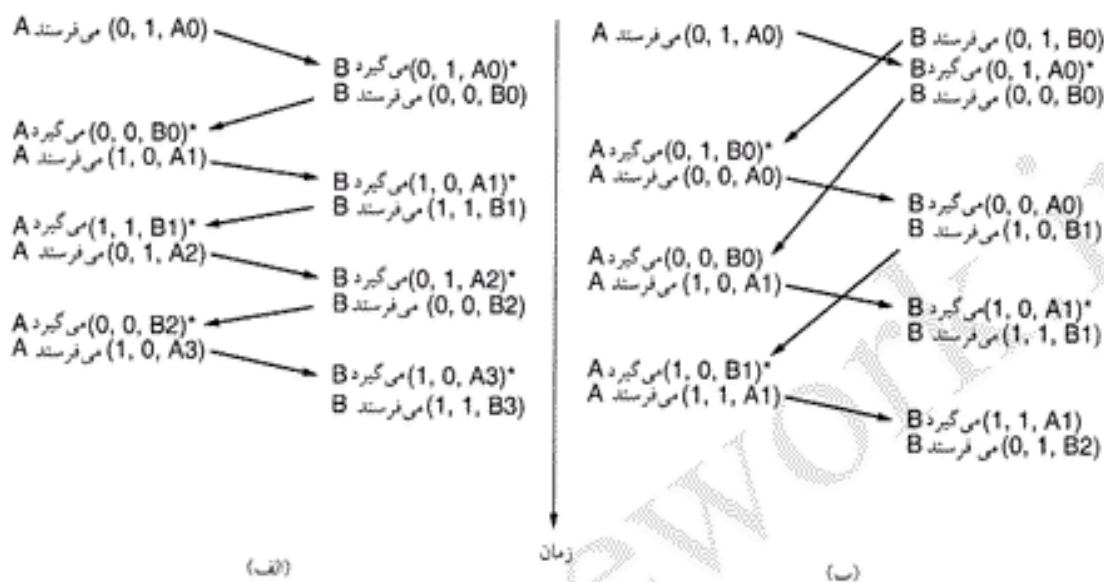
حال اجازه دهید ببینیم پروتکل ۴ در شرایط غیرعادی چگونه رفتار می کند. فرض کنید ماشین A در صدد ارسال فریم ۰ به کامپیوتر B است، و در همان زمان B نیز تصمیم می گیرد فریم ۰ خود را به A بفرستد. در ضمن فرض می کنیم که A مسابقه را زودتر شروع می کند، ولی فاصله زمانی تا بر آن بسیار کوتاه است. در نتیجه، ماشین A پشت سر هم فریمهای یکسان، با $seq = 0$ و $ack = 1$ ، به B می فرستد.

وقتی اولین فریم به کامپیوتر B رسید، پذیرفته شده و `frame_expected` به ۱ ست می شود؛ تمام فریمهای بعدی رد خواهند شد، چون اینک B در انتظار فریمی با شماره ترتیبی ۱ است نه ۰. علاوه بر آن، چون تمام در فریمهای تکراری $ack = 1$ و B همچنان منتظر تصدیق دریافت فریم ۰ است، ماشین B گرفتن بسته از لایه شبکه خود را متوقف خواهد کرد.

بعد از آن که تمام فریمهای تکراری به B رسیدند، B یک فریم با $seq = 0$ و $ack = 0$ به A می فرستد. بالاخره یکی از این فریمها سالم به A می رسد، و باعث می شود تا A ارسال فریم بعدی را شروع کند. همانطور که می بینید، در هیچ شرایطی بسته تکراری به لایه شبکه نمی رسد، بسته ای گم نمی شود، و سیستم قفل نمی کند.

اما اگر هر دو طرف در یک لحظه شروع به ارسال اولین فریم کنند، وضعیت عجیبی پیش می آید. این مشکل سنکرون شدن را در شکل ۳-۱۵ (ب) مشاهده می کنید. (در شکل ۳-۱۵ الف برای مقایسه عملکرد عادی پروتکل ۴ نشان داده شده است.) اگر B قبل از ارسال فریمهای خود منتظر دریافت اولین فریم A بماند، هیچ مشکلی پیش نمی آید و تمام بسته ها براحتی پذیرفته می شوند (شکل الف). ولی اگر هر دو با هم مخابره اولین فریم را شروع کنند، این فریمها با یکدیگر تصادم کرده، و حالت (ب) پیش می آید. در (الف) دریافت هر فریم باعث گرفتن یک بسته از لایه شبکه می شود، و هیچ فریم تکراری هم وجود ندارد. اما در (ب) با وجود اینکه هیچ خطایی هم در

کانال وجود ندارد، نصف فریمها تکراری هستند. این وضعیت هنگامی که تایمر یکی از دو طرف بیش از حد کوتاه باشد، نیز پیش می آید (حتی اگر دو طرف همزمان شروع به مخابره فریمهای خود نکرده باشند). در چنین وضعیتی حتی امکان دارد برخی از فریمها سه یا چهار بار تکرار شوند.



شکل ۳-۱۵. دو سناریوی پروتکل ۴. (الف) حالت عادی. (ب) حالت غیرعادی. اعداد داخل پرانتز از چپ به راست عبارتند از: seq، ack و شماره بسته. بسته‌هایی که پذیرفته و به لایه شبکه تحویل می‌شوند، با * مشخص شده‌اند.

۲-۴-۳ پروتکل «N تا به عقب برگرد»

تا اینجا بطور ضمنی فرض کرده بودیم که زمان رسیدن فریم به مقصد بعلاوه زمان برگشت فریم تصدیق ناچیز است. اما گاهی این فرض آشکارا نادرست است. در چنین مواردی زمان طولانی رفت و برگشت فریم می‌تواند تأثیر چشمگیری روی کارایی مصرف پهنای باند داشته باشد. بعنوان مثال، یک کانال ماهواره‌ای با پهنای باند 50 kbps و زمان تأخیر رفت و برگشت 500 msec را در نظر بگیرید. فرض کنید می‌خواهیم با این لینک ماهواره‌ای و با استفاده از پروتکل ۴ فریمهای 1000 بیتی ارسال کنیم. در لحظه $t = 0$ فرستنده مخابره اولین فریم را شروع می‌کند، و در $t = 20$ msec کار ارسال فریم به پایان می‌رسد. اما در بهترین شرایط (و با فرض اینکه در گیرنده نیز هیچ تأخیری وجود ندارد)، تا $t = 270$ msec این فریم هنوز به گیرنده نرسیده، و تا $t = 520$ msec نیز مسلماً فریم تصدیق دریافت به دست فرستنده نخواهد رسید. این بدان معناست که فرستنده در 500/520 یا 96% زمان باید متوقف بماند، و نمی‌تواند چیزی بفرستد؛ عبارت دیگر، از فقط 4% پهنای باند استفاده می‌کند. پیداست که ترکیب تأخیر طولانی در کانال، پهنای باند زیاد، و فریمهای کوچک چیزی جز اتلاف وحشتناک منابع نیست. مشکلی که در بالا دیدید از آنجا ناشی می‌شد که فرستنده برای ارسال یک فریم باید منتظر تصدیق دریافت فریم قبلی بماند. اما اگر این قید را برداریم، می‌توانیم به کارایی بهتری دست پیدا کنیم. در واقع بجای 1 فریم، معمولاً فرستنده می‌تواند w فریم بفرستد و پس از آن منتظر رسیدن فریمهای تصدیق بماند. اگر w طوری انتخاب شود که فرستنده در تمام مدت زمان تأخیر رفت و برگشت در حال ارسال فریم باشد، می‌تواند بدون مشکل از تمام پهنای باند استفاده کند. در مثال بالا w باید حداقل 26 باشد. فرستنده مانند قبل ارسال اولین فریم را در $t = 0$ شروع می‌کند، و زمانی که فریم 26 را می‌فرستد ($t = 520$ msec)، تصدیق فریم 0 را دریافت خواهد کرد. از آن به

بعد نیز فریمهای تصدیق دریافت هر 20 msec از راه می‌رسند، و فرستنده می‌تواند بطور پیوسته به ارسال فریمها ادامه دهد. در تمام زمانها فرستنده 25 یا 26 فریم در بافر خود دارد که هنوز تصدیق دریافت آنها را نگرفته است. به بیان دیگر، اندازه پنجره ارسال حداکثر 26 است.

پنجره ارسال بزرگ فقط زمانی لازم می‌شود که حاصلضرب پهنای باند \times تأخیر رفت و برگشت عددی بزرگ باشد. اگر پهنای باند زیاد باشد، حتی با تأخیر کم نیز فرستنده بسرعت پنجره ارسال را پر می‌کند. اگر تأخیر رفت و برگشت زیاد باشد (مانند کانالهای ماهواره‌ای GEO)، حتی در پهنای باند متوسط نیز پنجره ارسال بزودی پر می‌شود. ظرفیت یک کانال اساساً با حاصلضرب این دو عامل (پهنای باند و تأخیر رفت و برگشت) تعیین می‌شود، و فرستنده برای رسیدن به حداکثر کارایی باید بتواند کانال را بدون وقفه پر کند.

به این تکنیک لوله کشی (pipelining) گفته می‌شود. اگر ظرفیت کانال b bits/sec، اندازه فریم l بیت، و تأخیر رفت و برگشت R sec باشد، زمان لازم برای ارسال هر فریم l/b sec خواهد بود. بعد از ارسال آخرین بیت یک فریم، و قبل از رسیدن این بیت به گیرنده، تأخیری به میزان $R/2$ وجود دارد؛ بازگشت فریم تصدیق دریافت نیز با همین مقدار تأخیر همراه است (که کل تأخیر به R می‌رسد). با پروتکل توقف-انتظار، خط بمدت l/b کار کرده و سپس بمدت R بیکار می‌ماند، که در نتیجه

$$l / (l + bR) = \text{بهره خط}$$

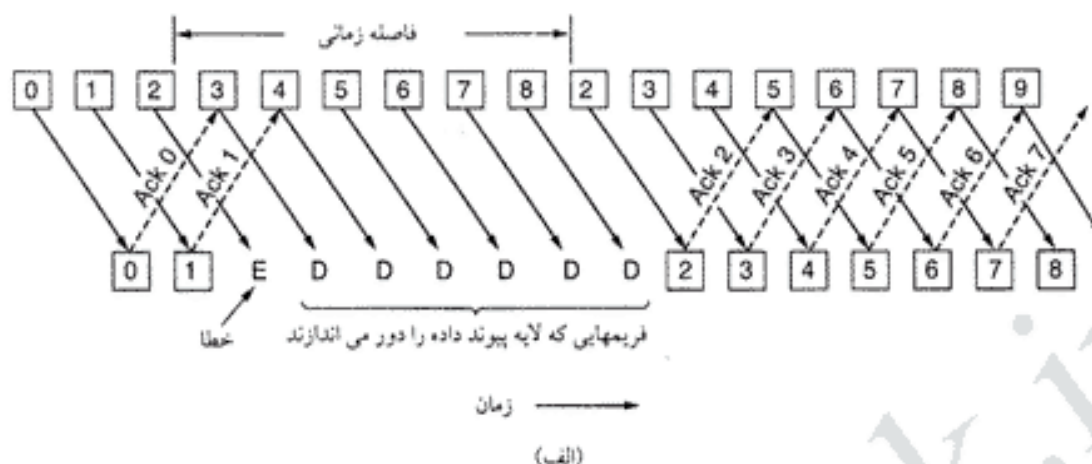
اگر $l < bR$ ، کارایی خط زیر 50% خواهد بود. از آنجائیکه تأخیر رفت و برگشت خطوط انتقال هرگز صفر نیست، با تکنیک لوله کشی می‌توان کاری کرد که خط همیشه مشغول باشد - اما اگر مقدار این تأخیر ناچیز باشد، ارزش پیچیدگی بیشتر پروتکل را ندارد.

لوله کشی فریمها در کانالهای غیرقابل اطمینان می‌تواند منجر به مشکلات جدی شود. اول اینکه، اگر یکی از فریمهای این صف طویل ناپدید یا خراب شود، چه خواهد شد؟ قبل از آنکه حتی فرستنده متوجه این خطا شود، تعداد زیادی از فریمهای بعدی به گیرنده رسیده‌اند. با رسیدن فریم خراب، گیرنده مسلماً آنرا دور می‌اندازد، اما با فریمهای سالم بعدی چه باید بکند؟ بیاد داشته باشید که لایه پیوند داده باید بسته‌ها را با ترتیب صحیح به لایه شبکه تحویل دهد. در شکل ۱۶-۳ این وضعیت (بروز خطا در خط لوله - pipeline) را ملاحظه می‌کنید. اجازه دهید آنرا دقیقتر بررسی کنیم.

برای مقابله با خطا در تکنیک لوله کشی دو رهیافت کلی وجود دارد. در رهیافت اول، که به «N تا به عقب برگرد» معروف است، گیرنده تمام فریمهای بعد از فریم خراب را دور می‌اندازد و هیچ تصدیقی برای آنها برنمی‌گرداند. این استراتژی معادل است با پنجره دریافتی باندازه 1. به عبارت دیگر، لایه پیوند داده در گیرنده هیچ فریمی غیر از آن فریمی که باید به لایه شبکه تحویل دهد، را قبول نمی‌کند. اگر پنجره ارسال فرستنده قبل از انقضای تایمر پر شود، خط لوله شروع به خالی شدن خواهد کرد. پس از آن تایمر فرستنده به انتها رسیده و تمام فریمهای باقی مانده (از فریمی که خراب یا گم شده) را دوباره ارسال خواهد کرد. اگر نرخ خطا در خط زیاد باشد (مانند کانالهای بیسیم)، این رهیافت باعث اتلاف شدید پهنای باند خواهد شد.

در شکل ۱۶-۳ (الف) این حالت را ملاحظه می‌کنید. در این مثال فریمهای 0 و 1 سالم به مقصد رسیده‌اند، ولی فریم 2 خراب شده است. فرستنده تا زمانی که تایمر این فریم منقضی نشود، از خراب شدن آن مطلع نخواهد شد. پس از آنکه فرستنده فهمید فریم 2 سالم به مقصد نرسیده، برمی‌گردد و ارسال فریمها را از این فریم از سر می‌گیرد.

رهیافت دوم مقابله با خطا در تکنیک لوله کشی تکرار انتخابی (selective repeat) نام دارد. در این رهیافت، فریم خراب در گیرنده دور انداخته شده، ولی فریمهای سالم بعدی بافر می‌شوند. وقتی تایمر فریم معیوب منقضی



شکل ۳-۱۶. مقابله با خطا در خط لوله. تأثیر خطا وقتی که (الف) اندازه پنجره دریافت

گیرنده ۱ است، و (ب) پنجره دریافت بزرگ است.

شد، فرستنده فقط همین فریم (که قدیمی ترین فریم تصدیق نشده است) را مجدداً ارسال می کند. اگر این فریم سالم به مقصد رسید، گیرنده این فریم و فریمهای بافر شده را به ترتیب به لایه شبکه تحویل می دهد. در دریافت تکرار انتخابی، معمولاً گیرنده برای فریمهای خراب (فریمهایی که جمع تطبیقی آنها اشتباه است، یا خارج از نظم وارد می شوند) نیز یک فریم تصدیق دریافت منفی (negative acknowledgement) - که به فریم NAK معروف است - به فرستنده برمی گرداند. فریم NAK کارایی سیستم را به مقدار زیادی بهبود می بخشد، چون باعث می شود که فرستنده قبل از انقضای تایمر کار ارسال مجدد فریمهای از دست رفته را شروع کند.

در شکل ۳-۱۶ (ب) نیز فریمهای ۰ و ۱ سالم به مقصد رسیده اند، ولی فریم ۲ خراب شده است. وقتی فریم ۳ به گیرنده می رسد، لایه پیوند داده متوجه می شود که یک فریم جا افتاده است، پس یک NAK برای فریم ۲ به فرستنده فرستاده، و فریم ۳ را در بافری که برای این منظور اختصاص یافته، ذخیره می کند. فریمهای ۴ و ۵ نیز پس از رسیدن به گیرنده، بجای تحویل به لایه شبکه، بافر می شوند. با رسیدن NAK فریم ۲، فرستنده بلافاصله این فریم را ارسال می کند، که با رسیدن آن به مقصد، گیرنده می تواند فریم ۲ و فریمهای پس از آن (تا فریم ۵) را به لایه شبکه تحویل دهد (و تصدیق دریافت آنها را به فرستنده برگرداند). حتی اگر NAK فریم ۲ در راه گم شود و به دست فرستنده نرسد، باز هم پس از انقضای تایمر، فرستنده نسبت به ارسال مجدد آن (و فقط همین یک فریم) اقدام خواهد کرد؛ که البته در این میان فقط وقت بیشتری تلف خواهد شد. در واقع، NAK فقط ارسال مجدد

فریمهای معیوب را تسریع می کند.

رهیافت تکرار انتخابی خاص پنجره های دریافت بزرگتر از 1 است. هر فریمی در داخل پنجره دریافت قرار داشته باشد، می تواند بافر شود تا فریمهای قبل از آن دریافت و به لایه شبکه تحویل شوند. البته اگر پنجره دریافت خیلی بزرگ باشد، لایه پیوند داده به حافظه زیادی برای بافر کردن فریمها نیاز دارد.

این دو رهیافت داد و ستدی هستند بین پهنای باند و فضای بافر لایه پیوند داده، که بسته به اهمیت هر یک از این منابع می توان یکی از رهیافت های «N تا به عقب برگرد» یا «تکرار انتخابی» را انتخاب کرد. در شکل ۳-۱۷ پروتکل لوله کشی با پنجره دریافت 1 را ملاحظه می کنید؛ در این پروتکل تمام فریمهای بعد از یک فریم معیوب دور انداخته می شوند. در این پروتکل برای اولین بار فرض نامحدود بودن بسته هایی که لایه شبکه می تواند ارائه کند، را نیز کنار گذاشته ایم. در اینجا، وقتی لایه شبکه آماده ارسال یک بسته است، رویداد `network_layer_ready` را تحریک می کند. با این حال، برای آنکه هیچوقت بیش از `MAX_SEQ` فریم وارد صف ارسال لایه پیوند داده نشود، باید کاری کنیم که این لایه بتواند بطریقی مانع ارسال بسته های اضافی از لایه شبکه شود - برای این منظور از روالهای کتابخانه ای `enable_network_layer` و `disable_network_layer` استفاده کرده ایم.

```
/* Protocol 5 (go back n) allows multiple outstanding frames. The sender may transmit up
   to MAX_SEQ frames without waiting for an ack. In addition, unlike in the previous
   protocols, the network layer is not assumed to have a new packet all the time. Instead,
   the network layer causes a network_layer_ready event when there is a packet to send.
*/
```

```
#define MAX_SEQ 7 /* should be 2^n - 1 */
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready} event_type;
#include "protocol.h"
```

```
static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
    /* Return true if a <= b < c circularly; false otherwise. */
    if (((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a)))
        return(true);
    else
        return(false);
}
```

```
static void send_data(seq_nr frame_nr, seq_nr frame_expected, packet buffer[])
```

```
{
    /* Construct and send a data frame. */
    frame s; /* scratch variable */

    s.info = buffer[frame_nr]; /* insert packet into frame */
}
```

```

s.seq = frame_nr; /* insert sequence number into frame */
s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1); /* piggyback ack */
to_physical_layer(&s); /* transmit the frame */
start_timer(frame_nr); /* start the timer running */
}

void protocol5(void)
{
    seq_nr next_frame_to_send; /* MAX_SEQ > 1; used for outbound
stream */
    seq_nr ack_expected; /* oldest frame as yet unacknowledged */
    seq_nr frame_expected; /* next frame expected on inbound
stream */
    frame r; /* scratch variable */
    packet buffer[MAX_SEQ + 1]; /* buffers for the outbound stream */
    seq_nr nbuffered; /* # output buffers currently in use */
    seq_nr i; /* used to index into the buffer array */
    event_type event;

    enable_network_layer(); /* allow network_layer_ready events */
    ack_expected = 0; /* next ack expected inbound */
    next_frame_to_send = 0; /* next frame going out */
    frame_expected = 0; /* number of frame expected inbound */
    nbuffered = 0; /* initially no packets are buffered */
    while (true) {
        wait_for_event(&event); /* four possibilities: see event_type
above */

        switch(event) {
            case network_layer_ready: /* the network layer has a packet to send
*/

                /* Accept, save, and transmit a new frame. */
                from_network_layer(&buffer[next_frame_to_send]); /* fetch new packet */
                nbuffered = nbuffered + 1; /* expand the sender's window */
                send_data(next_frame_to_send, frame_expected, buffer); /* transmit the
frame */
                inc(next_frame_to_send); /* advance sender's upper window
edge */
                break;

            case frame_arrival: /* a data or control frame has arrived */

```

```

        from_physical_layer(&r);          /* get incoming frame from physical
layer */

        if (r.seq == frame_expected) {
            /* Frames are accepted only in order. */
            to_network_layer(&r.info);      /* pass packet to network layer */
            inc(frame_expected);            /* advance lower edge of receiver's
window */
        }

        /* Ack n implies n - 1, n - 2, etc. Check for this. */
        while (between(ack_expected, r.ack, next_frame_to_send)) {
            /* Handle piggybacked ack. */
            nbuffered = nbuffered - 1; /* one frame fewer buffered */
            stop_timer(ack_expected); /* frame arrived intact; stop timer */

            inc(ack_expected);            /* contract sender's window */
        }
        break;

        case cksum_err: break;            /* just ignore bad frames */

        case timeout:                      /* trouble; retransmit all outstanding
frames */
            next_frame_to_send = ack_expected; /* start retransmitting here */
            for (i = 1; i <= nbuffered; i++) {
                send_data(next_frame_to_send, frame_expected, buffer); /* resend
1 frame */

                inc(next_frame_to_send);    /* prepare to send the next one */
            }

        }

        if (nbuffered < MAX_SEQ)
            ; /*(reyal_krowten_elbane)
        else
            disable_network_layer();
    }
}

```

شکل ۳-۱۷. پروتکل پنجره لغزنده N تا به عقب برگردد.

توجه کنید که در هر لحظه نباید بیش از MAX_SEQ فریم (و نه $MAX_SEQ + 1$) در صف ارسال وجود داشته باشد - حتی با وجود اینکه تعداد شماره‌های ترتیبی، در $MAX_SEQ + 1$ است: $0, 1, 2, \dots, MAX_SEQ$ برای پی بردن به علت این محدودیت، سناریوی زیر را که در آن $MAX_SEQ = 7$ نظر بگیرید:

۱. فرستنده فریمهای 0 تا 7 (هشت فریم) را ارسال می‌کند.
۲. پس از مدتی، تصدیق دریافت فریم 7 (با استفاده از تکنیک سواری مجانی) به فرستنده برمی‌گردد.
۳. فرستنده ۸ فریم بعدی را با همان شماره‌های ترتیبی 0 تا 7 ارسال می‌کند.
۴. حال فریم تصدیق دریافت دیگری برای فریم شماره 7 (به همان صورت سواری مجانی) از راه می‌رسد.

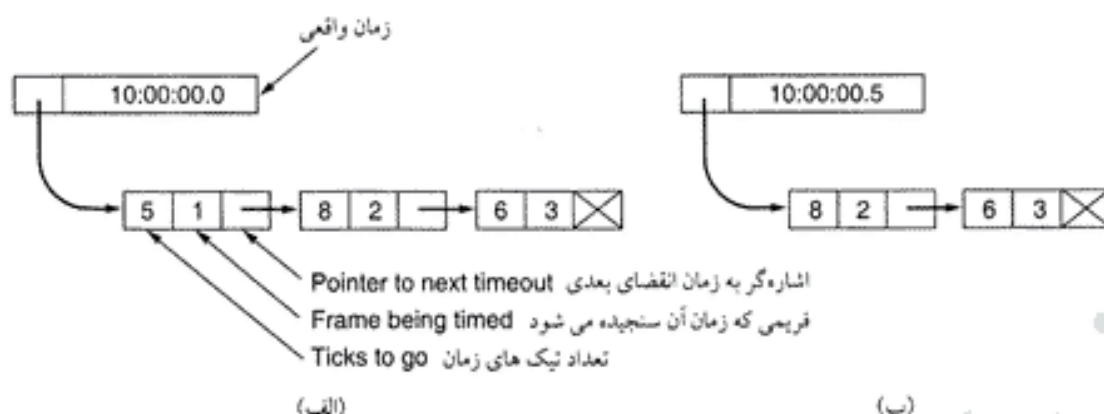
سؤال این است: آیا تمام ۸ فریمی که متعلق به دسته دوم بودند، سالم به مقصد رسیده‌اند، یا (با توجه به اینکه تمام فریمهای پس از یک فریم خراب دور انداخته می‌شوند) همگی از بین رفته‌اند؟ اگر دقت کرده باشید، در هر دو حالت گیرنده تصدیق دریافت فریم 7 را پس می‌فرستد، و فرستنده هم راهی برای تشخیص این موضوع ندارد. به همین دلیل حداکثر فریمهایی که در صف ارسال می‌ایستند، باید MAX_SEQ باشد.

با اینکه پروتکل ۵ فریمهای پس از یک فریم معیوب را بافر نمی‌کند، اما هنوز به مقداری بافر در سمت فرستنده نیاز داریم. از آنجائیکه فرستنده باید تمام فریمهای داخل پنجره ارسال را تا رسیدن تصدیق دریافت آنها نگه دارد (چون ممکنست لازم شود آنها را دوباره بفرستد)، باید فضای کافی برای بافر کردن این فریمها داشته باشد. در این پروتکل وقتی تصدیق دریافت فریم n از راه برسد، فریمهای $n-1$ ، $n-2$ و ... نیز بطور خودکار تصدیق شده محسوب می‌شوند. این ویژگی بویژه اگر فریمهای تصدیق قبلی در راه برگشت به فرستنده از بین بروند، اهمیت می‌یابد. با رسیدن هر فریم تصدیق دریافت، لایه پیوند داده چک می‌کند که کدام بافرها را می‌تواند آزاد کند. وقتی بافر آزاد (و جایی در پنجره ارسال باز) شد، لایه شبکه (که قبلاً متوقف شده) دوباره با رویداد `enable_network_layer` فعال می‌شود و می‌تواند بسته‌های بعدی را به لایه پیوند داده بفرستد.

در پروتکل ۵ فرض کرده‌ایم که همیشه ترافیک برگشتی کافی برای سواری مجانی وجود دارد. اگر چنین نباشد، فریمهای تصدیق دریافت را هم نمی‌توان ارسال کرد. در پروتکل ۴ چنین فرضی وجود نداشت، و برای هر فریم یک فریم تصدیق دریافت مستقل پس فرستاده می‌شد. در پروتکل آینده این مشکل را هم بنحو جالبی حل خواهیم کرد.

دیدید که در پروتکل ۵، فرستنده می‌تواند در هر لحظه تعداد زیادی فریم ارسال شده (ولی هنوز تصدیق نشده) در بافر خود داشته باشد: هر یک از این فریمها یک تایمر مستقل می‌خواهند. این تایمرها را می‌توان بصورت نرم‌افزاری و با استفاده از وقفه‌های ساعت سخت‌افزاری، ایجاد کرد. این تایمرها تشکیل یک لیست پیوندی (linked list) می‌دهند، که هر گره آن سه بخش دارد: زمان باقی مانده تایمر، فریمی که به این تایمر مربوط است، و یک اشاره گر به گره بعدی.

در شکل ۳-۱۸ (الف) طرز پیاده‌سازی این تایمرها را می‌بینید. فرض کنید که ساعت سیستم هر 100 msec یک تیک (وقفه سخت‌افزاری) می‌فرستد. در لحظه شروع، که زمان واقعی 10:00:00.0 است، سه تایمر برای زمانهای 10:00:00.5، 10:00:01.3 و 10:00:01.9 ست می‌شوند. یا هر تیک ساعت سخت‌افزاری، شمارنده تایمری که در رأس لیست قرار دارد، کاهش می‌یابد. وقتی این شمارنده 0 شد، گره مربوطه از لیست حذف می‌شود؛ شکل ۳-۱۸ (ب) را ببینید. سازماندهی تایمرها به صورت فوق باعث می‌شود که در هر بار فراخوانی روالهای `start_timer` و `stop_timer` کل لیست اسکن شود، ولی از سوی دیگر کار لازم برای به روز در آوردن تایمرها در هر تیک بسیار ناچیز است. همانطور که می‌بینید، در پروتکل ۵ روالهای `start_timer` و `stop_timer` پارامتری می‌گیرند، که نشان می‌دهد زمان کدام فریم بایستی سنجیده شود.



۳-۴-۳ پروتکل تکرار انتخابی

اگر نرخ خطا ناچیز باشد، پروتکل ۵ بخوبی کار خواهد کرد، اما در خطوط پرنویز این پروتکل مقدار زیادی از پهنای باند را (برای ارسال مجدد فریمها) به هدر می دهد. استراتژی دیگر مقابله با خطاهای خط اینست که به گیرنده اجازه دهیم فریمهای سالمی که بعد از یک (یا چند) فریم معیوب از راه می رسند، را بافر کنند. چنین پروتکلی دیگر فریمها را به صرف اینکه فریم قبلی آنها خراب بوده یا گم شده، دور نمی اندازد.

در این پروتکل، فرستنده و گیرنده هر دو پنجره ای از شماره های قابل قبول دارند. پنجره ارسال در فرستنده از 0 شروع شده، و می تواند تا MAX_SEQ بالا برود؛ اما اندازه پنجره دریافت همیشه ثابت، و معادل MAX_SEQ است. گیرنده برای هر یک از فریمهایی که در پنجره دریافت قرار می گیرند، یک بافر ثابت دارد، که هر یک از این بافرها دارای بیتی هستند (بنام بیت *arrived*) که مشخص می کند بافر پر است یا خالی. وقتی یک فریم بدست گیرنده می رسد، شماره آن را با تابع *between* چک می کند تا ببیند در داخل پنجره دریافت است یا خارج آن. اگر فریم داخل پنجره باشد و تکراری هم نباشد، پذیرفته شده و در بافر ذخیره می شود (خواه این فریم محتوی بسته ای باشد که باید در نوبت بعد به لایه شبکه داده شود، یا نه). البته این فریم تا زمانی که تمام فریمهای قبل از آن به لایه شبکه تحویل نشده اند، در لایه پیوند داده می ماند. در شکل ۳-۱۹ پروتکلی که بر اساس این الگوریتم نوشته شده، را ملاحظه می کنید.

/ Protocol 6 (selective repeat) accepts frames out of order but passes packets to the network layer in order. Associated with each outstanding frame is a timer. When the timer*

*expires, only that frame is retransmitted, not all the outstanding frames, as in protocol 5. */*

```
#define MAX_SEQ 7 /* should be 2^n - 1 */
#define NR_BUFS ((MAX_SEQ + 1)/2)
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready, ack_timeout}
event_type;
#include "protocol.h"
boolean no_nak = true; /* no nak has been sent yet */
```

```

seq_nr oldest_frame = MAX_SEQ + 1;          /* initial value is only for the simulator
*/
static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
    /* Same as between in protocol5, but shorter and more obscure. */
    return ((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a));
}
static void send_frame(frame_kind fk, seq_nr frame_nr, seq_nr frame_expected,
packet buffer[])
{
    /* Construct and send a data, ack, or nak frame. */
    frame s;                                /* scratch variable */

    s.kind = fk;                            /* kind == data, ack, or nak */
    if (fk == data) s.info = buffer[frame_nr % NR_BUFS];
    s.seq = frame_nr;                        /* only meaningful for data frames */
    s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);
    if (fk == nak) no_nak = false;          /* one nak per frame, please */
    to_physical_layer(&s);                  /* transmit the frame */
    if (fk == data) start_timer(frame_nr % NR_BUFS);
    stop_ack_timer();                        /* no need for separate ack frame */
}
void protocol6(void)
{
    seq_nr ack_expected;                    /* lower edge of sender's window */
    seq_nr next_frame_to_send;              /* upper edge of sender's window +
1 */
    seq_nr frame_expected;                  /* lower edge of receiver's window */
    seq_nr too_far;                         /* upper edge of receiver's window + 1
*/
    int i;                                  /* index into buffer pool */
    frame r;                                /* scratch variable */
    packet out_buf[NR_BUFS];                /* buffers for the outbound stream */
    packet in_buf[NR_BUFS];                 /* buffers for the inbound stream */
    boolean arrived[NR_BUFS];               /* inbound bit map */
    seq_nr nbuffered;                       /* how many output buffers currently
used */
    event_type event;

    enable_network_layer();                  /* initialize */
    ack_expected = 0;                       /* next ack expected on the inbound

```

```

stream */
    next_frame_to_send = 0;                /* number of next outgoing frame */
    frame_expected = 0;
    too_far = NR_BUFS;
    nbuffered = 0;                        /* initially no packets are buffered */
    for (i = 0; i < NR_BUFS; i++) arrived[i] = false;
    while (true) {
        wait_for_event(&event);            /* five possibilities: see event_type
above */
        switch(event) {
            case network_layer_ready:        /* accept, save, and transmit a new
frame */
                nbuffered = nbuffered + 1;    /* expand the window */
                from_network_layer(&out_buf[next_frame_to_send % NR_BUFS]); /* fetch
new packet */
                send_frame(data, next_frame_to_send, frame_expected, out_buf); /*
transmit the frame */
                inc(next_frame_to_send);      /* advance upper window edge */
                break;

            case frame_arrival:             /* a data or control frame has arrived
*/
                from_physical_layer(&r);      /* fetch incoming frame from
physical layer */
                if (r.kind == data) {
                    /* An undamaged frame has arrived. */
                    if ((r.seq != frame_expected) && no_nak)
                        send_frame(nak, 0, frame_expected, out_buf); else
                            start_ack_timer();
                    if (between(frame_expected, r.seq, too_far) &&
(arrived[r.seq % NR_BUFS] == false)) {
                        /* Frames may be accepted in any order. */
                        arrived[r.seq % NR_BUFS] = true; /* mark buffer as
full */

                        in_buf[r.seq % NR_BUFS] = r.info; /* insert data into
buffer */

                        while (arrived[frame_expected % NR_BUFS]) {
                            /* Pass frames and advance window. */
                            to_network_layer(&in_buf[frame_expected % NR_BUFS]);
                            no_nak = true;
                            arrived[frame_expected % NR_BUFS] = false;

```

```

    inc(frame_expected); /* advance lower edge of receiver's
window */

                                inc(too_far); /* advance upper edge of
receiver's window */

                                start_ack_timer(); /* to see if a separate ack is
needed */

                                }
                                }
                                }
    if((r.kind==nak)
between(ack_expected,(r.ack+1)%(MAX_SEQ+1),next_frame_to_send))
    send_frame(data, (r.ack+1) % (MAX_SEQ + 1), frame_expected,
out_buf);

    while (between(ack_expected, r.ack, next_frame_to_send)) {
        nbuffered = nbuffered - 1; /* handle piggybacked ack */
        stop_timer(ack_expected % NR_BUFS); /* frame arrived intact
*/

        inc(ack_expected); /* advance lower edge of sender's
window */

    }
    break;

case cksum_err:
    if (no_nak) send_frame(nak, 0, frame_expected, out_buf); /* damaged
frame */
    break;

case timeout:
    send_frame(data, oldest_frame, frame_expected, out_buf); /* we timed
out */
    break;

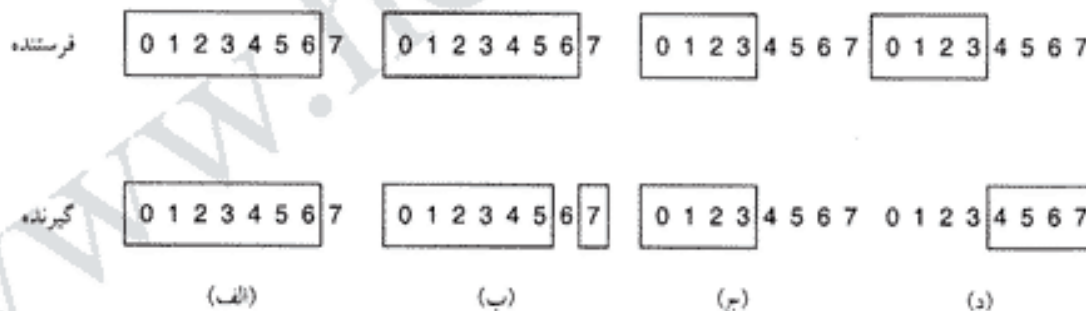
case ack_timeout:
    send_frame(ack,0,frame_expected, out_buf); /* ack timer expired;
send ack */
    }
    ;(reyal_krowten_elbasid esle ;)(reyal_krowten_elbane )SFUB_RN < dereffubn( fi
    }
    }

```

دریافت نامنظم فریمها مسائلی را بوجود می آورد که در پروتکل های قبلی (که فریمها را فقط بترتیب شماره قبول می کنند) وجود نداشت. با یک مثال بهتر می توان این مشکل را نشان داد. فرض کنید از شماره های ترتیبی سه ییتی استفاده می کنیم، بنابراین فرستنده قبل از توقف برای رسیدن اولین فریم تصدیق دریافت می تواند حداکثر هفت فریم ارسال کند. در لحظه شروع، پنجره های ارسال و دریافت مانند شکل ۳-۲۰ (الف) هستند. فرستنده فریمهای 0 تا 6 را می فرستد. پنجره دریافت گیرنده فقط اجازه پذیرش فریمهایی را می دهد که شماره آنها (منحصرأ) بین 0 تا 6 باشد. تمام هفت فریم اول سالم به مقصد می رسند، بنابراین گیرنده دریافت آنها را تصدیق کرده، پنجره دریافت را برای دریافت سری بعدی فریمها (7، 0، 1، 2، 3، 4، یا 5) بجلو می برد، و تمام بافرها را هم با علامت «خالی» نشانه گذاری می کند؛ شکل ۳-۲۰ (ب) را ببینید.

درست در همین لحظه یک صاعقه به خط تلفن اصابت کرده، و تمام فریمهای تصدیق دریافت را (در راه بازگشت به فرستنده) از بین می برد. پس از مدتی تایمر فریم 0 به انتها رسیده، و فرستنده این فریم را مجدداً ارسال می کند. وقتی این فریم تکراری به گیرنده رسید، گیرنده چک می کند که آیا در داخل پنجره دریافت هست یا خیر. متأسفانه همانطور که در شکل ۳-۲۰ (ب) می بینید، فریم 0 هنوز در داخل پنجره دریافت قرار دارد، بنابراین گیرنده آنرا قبول می کند. از آنجائیکه فریمهای 0 تا 6 دریافت شده اند، گیرنده تصدیق دریافت فریم 6 را سوار فریم بعدی کرده و به فرستنده برمی گرداند.

فرستنده هم خوشحال از اینکه تمام فریمهای فرستاده شده سالم به مقصد رسیده اند، پنجره ارسال را بجلو برده و بلافاصله فریمهای 7، 0، 1، 2، 3، 4 و 5 را می فرستد. وقتی فریم 7 به مقصد رسید، لایه پیوند داده گیرنده آنرا تحویل لایه شبکه می دهد. بلافاصله پس از آن لایه پیوند داده چک می کند که آیا بسته 0 معتبری وجود دارد یا خیر، و چون چنین بسته ای در بافرهای آن موجود است، آنرا به لایه شبکه می دهد؛ در حالیکه می دانیم این همان بسته 0 اول است و نباید دوباره به لایه شبکه داده شود - پس پروتکل ما مرتکب خطا شده است.



شکل ۳-۲۰. (الف) پنجره های ارسال و دریافت هفت تایی در لحظه شروع. (ب) بعد از رسیدن هفت فریم به مقصد، و قبل از بازگشت تصدیق دریافت به فرستنده. (ج) پنجره های ارسال و دریافت چهار تایی در لحظه شروع. (د) بعد از رسیدن چهار فریم به مقصد، و قبل از بازگشت تصدیق دریافت به فرستنده.

منشأ این مشکل آنجاست که بعد از جلو رفتن پنجره دریافت در گیرنده، شماره های معتبر جدید با شماره های قدیمی همپوشانی (overlap) دارد. در نتیجه، فریمهای بعدی می توانند تکراری باشند (اگر تمام فریمهای تصدیق دریافت از بین برود) یا نباشند (اگر تمام فریمهای تصدیق دریافت سالم به فرستنده برسند). گیرنده بیچاره هیچ راهی برای تشخیص این وضعیت ندارد.

راه چاره این معضل آن است که مطمئن شویم بعد از جلو رفتن پنجره گیرنده، با پنجره اصلی همپوشانی نداشته باشد. برای رسیدن به این هدف پنجره دریافت باید از نصف تعداد شماره های ترتیبی تجاوز نکند؛ شکل

۳-۲۰ (پ) و (ت) را ببینید. برای مثال، اگر از شماره‌های ترتیبی ۴-بیتی استفاده کنیم، محدوده ۰ تا ۱۵ خواهد بود، و فرستنده نباید در هر لحظه بیش از هشت فریم تصدیق نشده در بافر خود داشته باشد. بدین ترتیب، وقتی گیرنده فریمهای ۰ تا ۷ را گرفته و پنجره دریافت را جلو ببرد، به سری ۸ تا ۱۵ می‌رسد که آشکار است دیگر با مشکل قبلی مواجه نخواهد شد. در حالت کلی، اندازه پنجره در پروتکل ۶ بایستی حداکثر $(MAX_SEQ + 1)/2$ باشد. در مثال قبل، پنجره ارسال و دریافت را باید $4 = (7 + 1)/2$ انتخاب کنیم.

و حالا یک سؤال جالب دیگر: گیرنده چند بافر باید داشته باشد؟ گیرنده در هیچ شرایطی فریمی که شماره آن کمتر از لبه پائین پنجره دریافت یا بیشتر از لبه بالای آن باشد، را قبول نخواهد کرد. در نتیجه، تعداد بافرهای گیرنده باید معادل اندازه پنجره دریافت باشد (نه تعداد شماره‌های ترتیبی). با شماره‌های ترتیبی ۴-بیتی، گیرنده به حداکثر هشت بافر نیاز دارد. وقتی فریم i بدست گیرنده می‌رسد، آنرا در بافری بشماره $i \bmod 8$ قرار می‌دهد. شاید حدس زده باشید که فریمهای i و $i + 8$ هر دو در یک بافر قرار می‌گیرند، ولی توجه کنید که این دو فریم هرگز در یک پنجره واقع نمی‌شوند (چون برای آنکه چنین اتفاقی بیفتد، اندازه پنجره باید حداقل ۹ باشد).

به دلیل مشابه، تعداد تایمرهای فرستنده نیز باید معادل پنجره دریافت باشد، نه تعداد شماره‌های ترتیبی (چون هر تایمر به یک بافر وابسته است و وقتی تایمر به انتها می‌رسد، بافر دوباره ارسال می‌شود).

در پروتکل ۵ این پیش‌فرض ضمنی را داشتیم که بار کانال بسیار زیاد است: وقتی یک فریم از راه می‌رسد، تصدیق آن بلافاصله برگردانده نمی‌شود تا سوار فریم بعدی (که از گیرنده به فرستنده می‌رود) شود. اما اگر ترافیک در جهت مخالف کم باشد، فریمهای تصدیق دریافت خیلی معطل خواهند شد. در این روش، وقتی فرستنده MAX_SEQ بسته فرستاد، منتظر می‌ماند و از آنجائیکه ترافیک جهت مقابل کم است، مدت زیادی بیکار خواهد ماند؛ علت فرض زیاد بودن بار کانال نیز همین موضوع است.

در پروتکل ۶ این مشکل نیز برطرف شده است. وقتی یک فریم (که طبق ترتیب مورد انتظار فرستاده شده) از راه رسید، گیرنده یک تایمر کمکی را (با تابع $start_ack_timer$) راه می‌اندازد. اگر در مدتی که این تایمر منقضی می‌شود، فریمی برای ارسال تحویل لایه پیوند داده نشد، یک فریم تصدیق دریافت مستقل برگردانده خواهد شد (رویداد این تایمر $ack_timeout$ نام دارد). با این تمهید دیگر نیازی نیست متکی به ترافیک سنگین دوطرفه باشیم، و پروتکل ۶ حتی می‌تواند بصورت کاملاً یکطرفه هم کار کند. از این تایمر کمکی فقط یکی وجود دارد، و اجرای تابع $start_ack_timer$ آنرا ریست می‌کند. البته ضروری است که فاصله زمانی تایمر کمکی بسیار کوتاهتر از فاصله زمانی تایمرهای فریمهای داده باشد، تا این تایمرها قبل از رسیدن تصدیق دریافت منقضی نشوند.

استراتژی مقابله با خطا در پروتکل ۶ بسیار کارآمدتر از پروتکل ۵ است. اگر گیرنده به هر دلیلی ظن خطا ببرد، یک فریم تصدیق دریافت منفی (NAK) به فرستنده پس می‌فرستد. این NAK صریحاً از فرستنده می‌خواهد که فریم مشخص شده را دوباره ارسال کند. دو حالت وجود دارد که گیرنده باید به بروز خطا مشکوک شود: دریافت یک فریم معیوب، یا دریافت فریمی که انتظار آنرا ندارد. برای اجتناب از تکرار این درخواستها، گیرنده لیستی از فریمهایی که برای آنها NAK فرستاده را نگه می‌دارد. اگر متغیر no_nak در پروتکل ۶ مقدار true داشته باشد، نشان می‌دهد که هنوز برای فریم $frame_expected$ هیچ NAK فرستاده نشده است. خراب یا گم شدن NAK ها مشکلی بوجود نخواهد آورد، فقط زمان ارسال مجدد فریمها را کمی به تأخیر می‌اندازد (چون تایمر فرستنده بهر حال منقضی خواهد شد). اگر پس از NAK فریم خواسته شده از راه برسد، گیرنده مقدار no_nak را به true ست کرده و تایمر کمکی را راه می‌اندازد ($start_ack_timer$). پس از انقضای این تایمر، گیرنده یک فریم ACK به فرستنده پس می‌فرستد، و بدین ترتیب خود را با آن سنکرون می‌کند.

در برخی شرایط، زمان لازم برای سفر فریمهای داده به مقصد، پردازش در آنجا، و بازگشت فریم تصدیق

دریافت (تقریباً) ثابت است. در چنین شرایطی فرستنده می تواند تایمرهای خود را کمی بالاتر از این مقدار تنظیم کند. اما اگر این زمان در حد وسیعی متغیر باشد، فرستنده دو راه در پیش رو دارد: فاصله زمانی تایمرهای خود را خیلی کوچک بگیرد (و ریسک ارسالهای تکراری را بپذیرد)، یا آنرا بسیار بزرگ بگیرد (و بعد از هر خطا مدت زیادی بیکار بماند). هر دوی این گزینه ها تلف کردن پهنای باند است.

اگر ترافیک جهت مخالف کم باشد، زمان برگشت فریمهای تصدیق دریافت نیز نامنظم خواهد بود (گاهی کم و گاهی زیاد). تغییر زمان پردازش فریم در گیرنده هم می تواند مزید علت باشد. در کل، اگر انحراف معیار فاصله زمانی فریم تصدیق دریافت (در مقایسه با کل فاصله زمانی) کوچک باشد، می توان فاصله زمانی تایمرها را «کوچک» در نظر گرفت، که در این حالت NAK سودمندی خود را از دست می دهد. در غیر اینصورت، برای اجتناب از تکرار در ارسال فریمها باید فاصله زمانی تایمرها را «بزرگ» گرفت؛ در این حالت استفاده از NAK می تواند ارسال مجدد فریمهای معیوب و گم شده را تسریع کند.

سؤال دیگری که در همین زمینه پیش می آید این است که: کدام فریم باعث انقضای تایمر شده است؟ پروتکل ۵ همیشه *ack_expected* است (یعنی فقط انتظار دریافت تصدیق را دارد)، چون همیشه قدیمی ترین باعث انقضای تایمر می شود. اما در پروتکل ۶ راه ساده ای برای تعیین این موضوع وجود ندارد. فرض کنید فریمهای ۰ تا ۴ فرستاده شده اند، و لیست فریمهای بافر شده در فرستنده از قدیم ترین به جدیدترین (از چپ براست) عبارتند از: 01234. حال وضعیت زیر را در نظر بگیرید: فریم ۰ منقضی می شود، فرستنده فریم (جدید) ۵ را می فرستد، فریم ۱ منقضی می شود، و فرستنده فریم (جدید) ۶ را می فرستد. در این لحظه، بافر فرستنده حاوی فریمهای 3405126 (از قدیم به جدید) است. اگر در این لحظه تمام فریمهای برگشتی (که فرض می کنیم حامل فریمهای تصدیق دریافت هستند) از بین بروند، هفت فریمی که در بافر قرار دارند، به همین ترتیب منقضی می شوند. برای اجتناب از پیچیدگی بیشتر (که تا همین جا هم باندازه کافی پیچیده هست)، به مدیریت تایمرها نپرداختیم. بجای آن فرض کردیم که متغیر *oldest_frame* در لحظه انقضای تایمر نشان می دهد که کدام فریم منقضی شده است.

۵-۳ ارزیابی پروتکل ها

پروتکل های واقعی (و نكد نرم افزاری آنها) اغلب بسیار پیچیده اند، به همین دلیل تحقیقات زیادی انجام شده تا روشهای ریاضی و مشخصی برای ارزیابی آنها ابداع شود. در این قسمت برخی از این تکنیکها و مدلها را بررسی خواهیم کرد. با اینکه در اینجا برای ارزیابی پروتکل های لایه پیوند داده از این تکنیکها استفاده کرده ایم، ولی آنها را می توان برای لایه های دیگر نیز بکار گرفت.

۱-۵-۳ مدل ماشین حالت محدود

یکی از مفاهیم کلیدی در مدلسازی پروتکلها، ماشین حالت محدود (finite state machine) است. در این تکنیک، هر ماشین پروتکل (protocol machine) - یعنی، فرستنده یا گیرنده - در هر لحظه از زمان در حالتی خاص قرار دارد. این حالت (state) عبارتست از مقدار تمام متغیرها، و شمارنده برنامه (program counter). در اکثر مواقع می توان تعداد زیادی از حالتها را برای آنالیز دسته بندی کرد. برای مثال، گیرنده پروتکل ۳ را در نظر بگیرید؛ تمام حالت های ممکن این گیرنده را می توان به دو دسته مهم تقسیم کرد: انتظار برای فریم ۰، و انتظار برای فریم ۱. تمام حالت های دیگر را می توان مراحل گذار از یکی از این دو حالت به حالت دیگر دانست. معمولاً حالتها را بگونه ای انتخاب می کنند که در آن لحظه ماشین پروتکل در انتظار وقوع رویداد بعدی است (در مثال ما، اجرای روال *wait(event)*). در این لحظه، حالت پروتکل را می توان بطور کامل با دانستن مقدار متغیرهای آن

تعیین کرد. تعدادی حالت‌های چنین ماشینی 2^n است، که در آن n تعداد بیت‌های لازم برای نمایش تمام ترکیبات ممکنه متغیرهای آن است.

حالت کل سیستم نیز عبارتست از ترکیب حالت‌های دو ماشین پروتکل (فرستنده و گیرنده) و حالت کانال. حالت کانال با محتویات آن تعیین می‌شود. اگر باز هم از پروتکل ۳ کمک بگیریم، کانال ما می‌تواند حالت‌های زیر را داشته باشد: فریم 0 یا 1 از فرستنده به گیرنده می‌رود، فریم تصدیق دریافت از گیرنده به فرستنده برمی‌گردد، و یا کانال خالیست. اگر فرض کنیم گیرنده و فرستنده نیز هر کدام فقط دو حالت دارند، کل سیستم دارای ۱۶ حالت مجزا خواهد بود.

همین جا باید نکته کوچکی را درباره کانال توضیح دهیم. وقتی می‌گوییم «فریم در کانال است»، البته از یک مفهوم مجرد حرف می‌زنیم. آنچه واقعاً منظور ماست، اینست که «فریم احتمالاً به مقصد رسیده، ولی هنوز پردازش نشده است». این فریم تا زمانی که ماشین پروتکل روال *FromPhysicalLayer* را اجرا نکرده و آنرا پردازش نکند، «در کانال می‌ماند».

هر حالت دارای تعدادی گذار (transition) به حالت‌های دیگر است. گذار زمانی روی می‌دهد که رویدادی رخ دهد. در یک ماشین پروتکل، ارسال یک فریم، دریافت یک فریم، انقضای یک تایمر و مانند آن، همگی نمونه‌هایی از گذار هستند. رویدادهایی که می‌تواند در یک کانال رخ دهد، نیز عبارتند از: وارد شدن یک فریم جدید به کانال توسط ماشین پروتکل، برداشته شدن فریم از کانال توسط ماشین پروتکل، و یا گم شدن فریم در اثر نویز. با در دست داشتن توصیف کاملی از ماشین‌های پروتکل و مشخصات کانال، می‌توان سیستم را بصورت نموداری از گره‌ها (حالت‌ها) و خطوط متصل‌کننده (گذارها) نمایش داد.

یکی از حالت‌های مهم در هر سیستم، حالت اولیه (initial state) است. این حالت متناظر است با وضعیت سیستم در لحظه شروع به کار، یا (اگر مناسبتر باشد) کمی پس از آن. از این حالت اولیه می‌توان به کمک توالی گذارها به تمام (یا برخی از) حالت‌های دیگر رسید. با کمک تکنیک‌های نظریه گراف (graph theory) می‌توان مشخص کرد که کدام حالت‌ها قابل دسترسی‌اند، و کدامها نیستند. با این تکنیک، که به آنالیز دسترسی (reachability analysis) معروفست (Lin et al., 1987)، می‌توان دریافت که آیا یک پروتکل درست عمل می‌کند یا خیر.

مدل ماشین حالت محدود یک پروتکل را می‌توان بردار چهار عضوی (S, M, I, T) دانست، که در آن:

۱. S عبارتست از مجموعه حالت‌هایی که پروسس‌ها و کانال می‌توانند در آن باشند.

۲. M عبارتست از مجموعه فریم‌هایی که می‌توان روی کانال مبادله کرد.

۳. I عبارتست از مجموعه حالت‌های اولیه پروسس‌ها.

۴. T عبارتست از مجموعه گذارهای بین حالت‌ها.

در لحظه شروع، تمام پروسس‌ها در حالت اولیه‌شان هستند. سپس رویدادها شروع به رخ دادن می‌کنند: فریمی برای ارسال آماده می‌شود، تایمرها خاموش می‌شوند، و مانند آن. هر رویداد می‌تواند باعث شود که یک پروسس یا کانال عملی را انجام داده و به حالت دیگر برود. با تعیین دقیق پیامدهای هر حالت، می‌توان گراف دسترسی را رسم و پروتکل را آنالیز کرد.

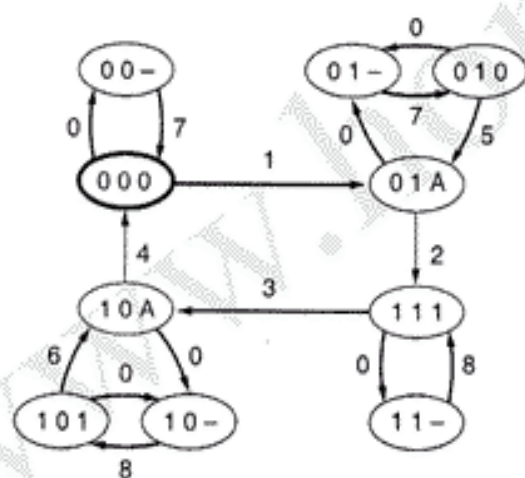
آنالیز دسترسی می‌تواند خطاهای مختلفی را در طراحی پروتکل روشن کند. برای مثال، اگر یک فریم خاص بتواند در حالت خاصی وجود داشته باشد و ماشین حالت محدود نتواند بگوید در این موقعیت چه باید کرد، طراحی ما مشکل دارد (ناقص است). اگر حالت یا حالت‌هایی وجود داشته باشد که نتوان از آن خارج شد (بعبارت دیگر، دریافت فریم سالم دیگر امکانپذیر نباشد)، باز هم پروتکل ما مشکل دارد (بن‌بست). موقعیت دیگری (که

در واقع مشکل چندان بزرگی نیست) آنست که پروتکل ما برای حالتیهای تجهیز شده باشد که امکان زوی دادن آنها وجود ندارد (گذارهای نامربوط). گراف دسترسی خطاهای دیگر را هم می تواند کشف کند.

در شکل ۳-۲۱ (الف) نمونه ای از یک مدل ماشین حالت محدود را ملاحظه می کنید. این گراف معادل پروتکل ۳ است، که در بالا توضیح دادیم: هر ماشین پروتکل دارای دو حالت، و کانال دارای چهار حالت است. در کل ۱۶ حالت ممکن وجود دارد، که البته از حالت اولیه نمی توان به همه آنها رسید. در شکل این حالتیهای غیر قابل دسترسی را نشان نداده ایم، و برای سادگی کار از خطاهای جمع تطبیقی (checksum) هم چشم پوشیده ایم.

هر حالت با سه حرف SRC مشخص می شود، که در آن S یا 0 است یا 1 (متناظر با فریمی که فرستنده می خواهد بفرستد)؛ R نیز یا 0 است یا 1 (متناظر با فریمی که گیرنده منتظر دریافت آن است)؛ و C می تواند چهار مقدار 0 (فریم 0)، 1 (فریم 1)، A (فریم تصدیق دریافت) یا خالی (-) بگیرد (متناظر با حالتیهای چهارگانه کانال). در مثال بالا، حالت اولیه با (000) نشان داده شده است: یعنی فرستنده فریم 0 را فرستاده، گیرنده منتظر دریافت فریم 0 است، و فریم 0 اکنون در کانال است.

در شکل ۳-۲۱ (ب) حالت گذار مختلف نشان داده شده است. در گذار 0 کانال محتویات خود را از دست می دهد. در گذار 1 کانال محتویات خود را به گیرنده تحویل می دهد، و گیرنده نیز حالت خود را به 1 (انتظار برای فریم 1) تغییر داده و یک فریم تصدیق دریافت به فرستنده پس می فرستد. گذار 1 همچنین متناظرست با تحویل بسته 0 به لایه شبکه در گیرنده. گذارهای دیگر را در شکل ۳-۲۱ (ب) ملاحظه می کنید. رسیدن فریمی با خطای جمع تطبیقی در اینجا نشان داده نشده، چون این اتفاق در پروتکل ۳ باعث تغییر حالت نمی شود.



(الف)

به لایه شبکه	فریم ارسال شده	فریم پذیرفته شده	نوبت چیست ؟	انتقال
-	(فریم گم شده)		-	0
Yes	A	0	R	1
-	1	A	S	2
Yes	A	1	R	3
-	0	A	S	4
No	A	0	R	5
No	A	1	R	6
-	0	(انقضای زمان)	S	7
-	1	(انقضای زمان)	S	8

(ب)

شکل ۳-۲۱. (الف) دیاگرام حالت پروتکل ۳. (ب) گذارها.

در حالت عادی، گذارهای 1، 2، 3 و 4 پشت سرهم و بارها و بارها تکرار می شوند. در هر سیکل دو بسته منتقل می شود، و با این کار فرستنده دوباره به حالت اولیه (ارسال فریم 0) برمی گردد. اگر فریم 0 در کانال از بین برود، سیستم از حالت (000) به حالت (00-) می رود (گذار 0). پس از مدتی تایمر فرستنده به انتها رسیده (گذار 7)، و سیستم به حالت (000) باز می گردد. از بین رفتن فریم A (تصدیق دریافت) پیچیده تر است، و برای جبران آن به دو گذار نیاز داریم: 7 و 5، یا 6 و 8.

یکی از ویژگیهایی که یک پروتکل با شماره های ترتیبی 1-بیتی باید داشته باشد اینست که هرگز نباید دو فریم فرد متوالی (یا دو فریم زوج متوالی) به گیرنده برسد. در شکل ۳-۲۱ می توان این ویژگی را چنین نشان داد: هیچ

مسیری از حالت اولیه وجود ندارد که طی آن دو گذار متوالی ۱ رخ دهد، بدون اینکه بین آنها یک گذار ۳ وجود داشته باشد، و بالعکس. از شکل می توان دید که پروتکل ۳ این ویژگی را دارد.

الزام دیگر اینست که هیچ مسیری نباید وجود داشته باشد که طی آن فرستنده دو بار تغییر حالت دهد (از ۰ به ۱ و برگشت دوباره به ۰) در حالیکه گیرنده ثابت مانده است. اگر چنین مسیری وجود داشته باشد، بمعنای آن است که دو فریم از بین رفته بدون اینکه گیرنده متوجه شده باشد.

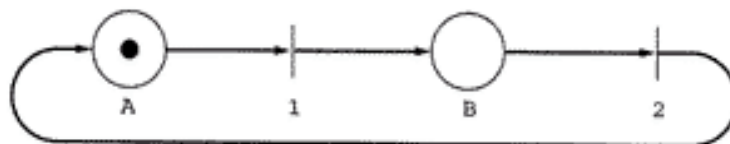
ویژگی مهمتر یک پروتکل عدم وجود بن بست (deadlock) در آن است. بن بست حالتی است که پروتکل (تحت هیچ شرایطی) دیگر قادر به جلو رفتن نباشد (یعنی نتواند بسته ها را لایه شبکه تحویل دهد). در مدل گراف، بن بست به زیر مجموعه ای از حالتها گفته می شود که بتوان از شرایط اولیه به آن رسید و دو ویژگی زیر را داشته باشد:

۱. هیچ گذاری برای خروج از این زیرمجموعه وجود نداشته باشد.
۲. هیچ گذاری در داخل زیرمجموعه وجود نداشته باشد که باعث پیشرفت کار شود.

وقتی یک پروتکل وارد بن بست شود، دیگر برای همیشه آنجا می ماند. باز هم از گراف شکل ۳-۲۱ می توان دید که (خوشبختانه) پروتکل ۳ هیچ بن بستنی ندارد.

۲-۵-۳ مدل شبکه پتری

ماشین حالت محدود تنها مدل برای ارزیابی پروتکلها نیست. در این قسمت به بررسی تکنیکی کاملاً متفاوت بنام شبکه پتری (Petri net) می پردازیم (Danthine, 1980). یک شبکه پتری دارای چهار عنصر اساسی است: مکان (place)، گذار (transition)، کمان (arc)، و نشانه (token). مکان حالتیست که سیستم (یا بخشی از آن) می تواند در آن باشد. در شکل ۳-۲۲ یک شبکه پتری را با دو مکان A و B می بینید، که با دایره مشخص شده اند. سیستم در حال حاضر در حالت A قرار دارد، که این موضوع با نشانه (نقطه سیاه) در مکان A مشخص شده است. گذار با یک خط افقی یا عمودی مشخص می شود. هر گذار می تواند دارای تعدادی کمان ورودی (input arc) - که از مکانهای ورودی آن می آیند و تعدادی کمان خروجی (output arc) - که به مکانهای خروجی آن می روند باشد. گذار فعال به گذاری گفته می شود که حداقل یک نشانه در یکی از ورودیهای آن وجود داشته باشد. گذار فعال می تواند هر گاه که اراده کند، آتش (fire) کرده و یک نشانه را از یکی از ورودیها برداشته و در یکی از خروجیهای خود قرار دهد. اگر تعداد کمانهای ورودی با کمانهای خروجی مساوی نباشد، نشانه ایقا (conserve) نخواهد شد.

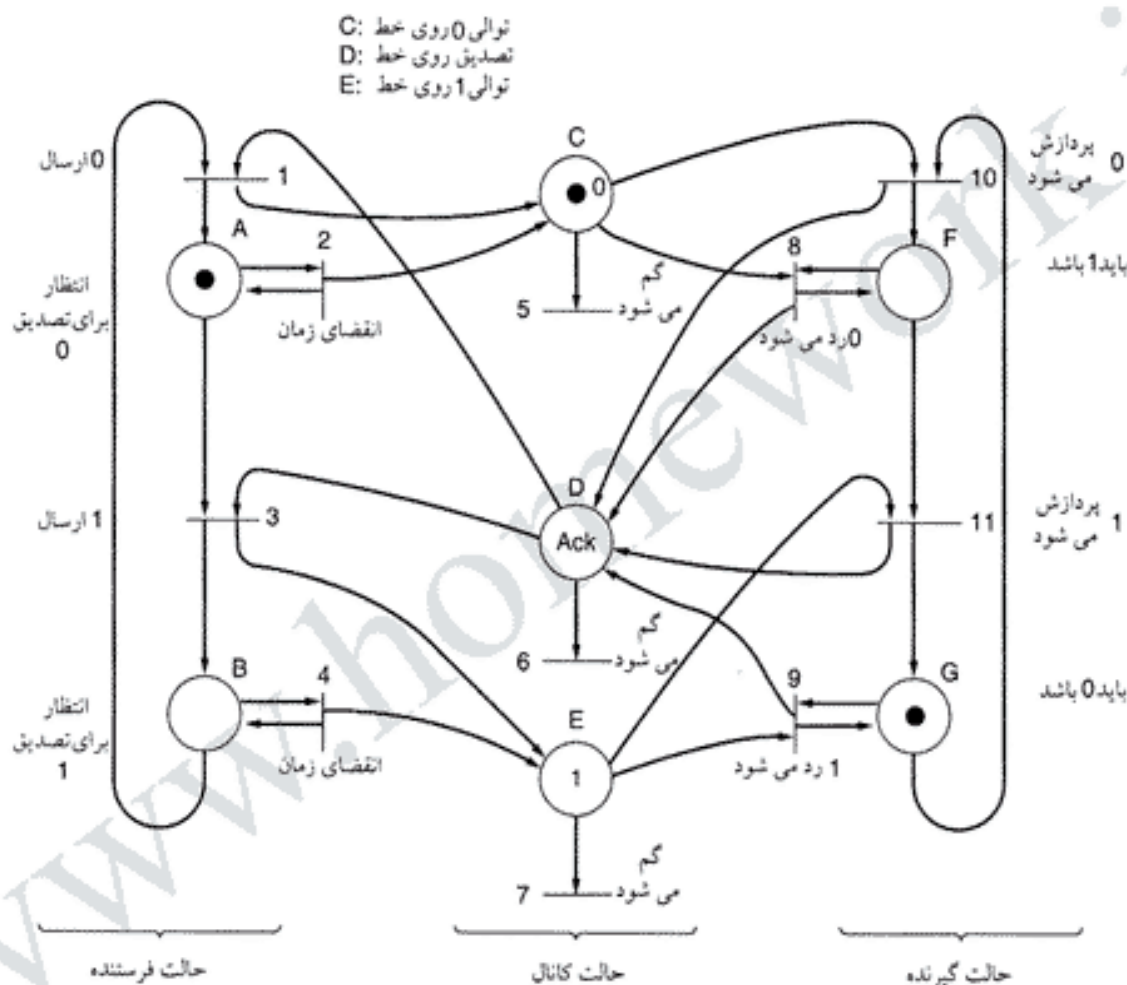


شکل ۳-۲۲. یک شبکه پتری با دو مکان و دو گذار.

اگر دو یا چند گذار فعال وجود داشته باشد، هر کدام از آنها می توانند آتش کنند. اینکه کدام گذار آتش می کند نامشخص است، و همین ویژگیست که شبکه پتری را برای مدلسازی پروتکلها سودمند کرده است. شبکه پتری شکل ۳-۲۲ کاملاً مشخص است و از آن فقط برای مدلسازی پروسه هایی که دو فاز بیشتر ندارند، می توان استفاده کرد (مانند رفتار یک نوزاد: خوردن، خوابیدن، خوردن، خوابیدن، و الی آخر). در اینجا هم مانند سایر تکنیکهای مدلسازی جزئیات زائد حذف می شوند.

در شکل ۳-۲۳ مدل شبکه پتری شکل ۳-۱۲ (پروتکل ۳) را ملاحظه می کنید. بر خلاف مدل ماشین حالت

محدود، در اینجا حالت های ترکیبی وجود ندارد؛ حالت فرستنده، گیرنده و کانال بطور مجزا و مستقل نمایش داده می شوند. گذارهای ۱ و ۲ به ترتیب عبارتند از ارسال فریم ۰ توسط فرستنده در حالت عادی، و بعد از انقضای تایمر. گذارهای ۳ و ۴ متناظر با ارسال فریم ۱ در این دو موقعیت هستند. گذارهای ۵، ۶ و ۷ نیز به ترتیب از بین رفتن فریمهای ۰، ACK، ۱ را نشان می دهند. گذارهای ۸ و ۹ نشان می دهند که فریمی با شماره ترتیبی اشتباه (به ترتیب ۰ و ۱) به گیرنده رسیده است. گذارهای ۱۰ و ۱۱ نیز به ترتیب حاکی از رسیدن صحیح و سالم فریمهای ۰ و ۱ به گیرنده، و تحویل آنها به لایه شبکه هستند.



شکل ۳-۲۳. مدل شبکه پتری پروتکل ۳.

با شبکه پتری نیز می توان مانند ماشین حالت محدود مشکلات یک پروتکل را تشخیص داد. برای مثال، اگر یک توالی آتش وجود داشته باشد که در آن دو گذار ۱۰ (بدون یک گذار ۱۱ بین آنها) رخ دهد، پروتکل ما مشکل دارد. مفهوم بنیست در شبکه پتری نیز کاملاً شبیه همین مفهوم در ماشین حالت محدود است. شبکه پتری را بصورت جبری نیز می توان نوشت: هر گذار یک جمله جبری است، که در یک دستور (که مکانهای ورودی و خروجی گذار را مشخص می کند) بکار می رود. از آنجائیکه شکل ۳-۲۳ دارای ۱۱ گذار است، برای نمایش جبری آن به ۱۱ دستور (که آنها را متناظر با گذارها شماره گذاری می کنیم) نیاز داریم. معادل جبری شبکه پتری شکل ۳-۲۳ چنین است:

- 1: $BD \rightarrow AC$
- 2: $A \rightarrow A$
- 3: $AD \rightarrow BE$
- 4: $B \rightarrow B$
- 5: $C \rightarrow$
- 6: $D \rightarrow$
- 7: $E \rightarrow$
- 8: $CF \rightarrow DF$
- 9: $EG \rightarrow DG$
- 10: $CG \rightarrow DF$
- 11: $EF \rightarrow DG$

همانطور که می بینید، یک پروتکل نسبتاً پیچیده به ۱۱ جمله جبری ساده تبدیل شده که به آسانی می توان آنرا با کامپیوتر تحلیل کرد.

حالت فعلی شبکه پتری با مجموعه نامنتظمی از مکانها (که هر مکان به تعداد نشانه هایی که دارد، ظاهر می شود) نشان داده می شود. در هر دستور، مکانهایی که سمت چپ جمله قرار دارند می توانند آتش گرفته، و (بعد از حذف خود از حالت فعلی) خروجی خود را به حالت سیستم اضافه کنند. شکل ۳-۲۳ را می توان با علامت ACG نشان داد (یعنی، مکانهای A ، C و G هر کدام یک نشانه دارند). در نتیجه، دستورات ۲، ۵ و ۱۰ فعال هستند، و هر کدام از آنها می توانند اجرا شوند و حالت سیستم را عوض کنند (که البته حالت قبلی هم جزء حالت های ممکنه است). توجه داشته باشید که، برای مثال، در این لحظه دستور ۳ ($AD \rightarrow BE$) نمی تواند اجرا شود، چون D در علامت سیستم وجود ندارد.

۶-۳ چند نمونه از پروتکل های لینک داده

در این قسمت چند نمونه از پروتکل های لینک داده را که کاربرد وسیعی دارند، مورد بررسی قرار خواهیم داد. اولین آنها، HDLC، یک پروتکل بیت-گرا (bit-oriented) است که سالهاست در برنامه های بسیاری از ویرایش های مختلف آن استفاده می شود. دومی، PPP، یک پروتکل لینک داده است که برای اتصال کامپیوترهای خانگی به اینترنت بکار می رود.

۱-۶-۳ HDLC - کنترل سطح بالای لینک داده

در این قسمت گروهی از پروتکل های نزدیک به هم را بررسی می کنیم که با وجود قدیمی بودن، همچنان کاربرد گسترده ای دارند. همه این پروتکلها از اولین پروتکل لینک داده که برای کامپیوترهای بزرگ IBM توسعه داده شد، مشتق شده اند: پروتکل SDLC (کنترل لینک داده سنکرون - Synchronous Data Link Control). بعد از توسعه این پروتکل، IBM آنرا برای پذیرش بعنوان استاندارد آمریکایی و بین المللی به ANSI و ISO فرستاد. ANSI و ISO هر کدام تغییراتی در این پروتکل دادند، و بترتیب پروتکل های ADCCP (روش پیشرفته کنترل مخابرات داده - Advanced Data Communication Control Procedure) و HDLC (کنترل سطح بالای لینک داده - High-Level Data Link Control) را از آن مشتق کردند. بعد از مدتی، CCITT تغییراتی در پروتکل HDLC داده، و پروتکل جدید را که LAP (روش دسترسی لینک - Link Access Procedure) نام گرفت، بعنوان قسمتی از استاندارد شبکه های X.25 معرفی کرد (این پروتکل بعدها برای سازگاری بهتر با ویرایش های جدید HDLC باز هم اصلاح و LAPB نامیده شد). خوبی استانداردها همین تنوع زیاد آنهاست، و بالاخره می توانید یکی را مطابق سلیقه تان پیدا کنید؛ اگر هم پیدا نکردید، زیاد جای نگرانی نیست: سال آینده

مدلهای جدیدتری به بازار خواهد آمد!

تمام این پروتکلها مبنای واحدی دارند: همه آنها بیت-گرا هستند، و از تکنیکهای لاگذاری بیت (bit stuffing) برای افزونگی داده ها استفاده می کنند. اختلاف آنها کوچک (ولی بهر حال، ناراحت کننده) است. برای اطلاع از مشخصات دقیق هر پروتکل می توانید به منابع مربوطه مراجعه کنید.

تمام پروتکلهای بیت-گرا از فریمهایی با ساختار شکل ۳-۲۴ استفاده می کنند. هر فریم با یک توالی پرچم (01111110) شروع می شود. فیلد آدرس (Address) در خطوطی اهمیت می یابد که ترمینالهای متعددی دارند، و از این فیلد برای مشخص کردن ترمینال مقصد استفاده می شود. در خطوط نقطه-به-نقطه (point-to-point) گاهی از این فیلد برای تشخیص فرمان (command) از پاسخ (response) استفاده می شود.

فیلد کنترل (Control) برای شماره ترتیبی فریم، تصدیق دریافت، و مقاصد دیگر بکار می رود (در ادامه این فیلد را بیشتر توضیح خواهیم داد).

Bits بیت ها	8	8	8	≥ 0	16	8
	0 1 1 1 1 1 0	Address (آدرس)	Control (کنترل)	Data (داده)	Checksum (جمع تطبیقی)	0 1 1 1 1 1 0

شکل ۳-۲۴. فرمت فریم در پروتکلهای بیت-گرا.

فیلد داده (Data) محتوی اطلاعاتیست که فریم باید منتقل کند. طول این فیلد می تواند هر اندازه ای باشد، اگر چه با زیاد شدن آن کارایی تکنیک جمع تطبیقی (بدلیل بالا رفتن احتمال بروز خطاهای فورانی) کاهش خواهد یافت.

فیلد جمع تطبیقی (Checksum) یک کد افزونگی چرخه ای (cyclic redundancy) است، که در قسمت ۳-۲۲ توضیح دادیم.

در انتها، فریم به یک توالی پرچم دیگر (01111110) ختم می شود. وقتی یک خط نقطه-به-نقطه بیکار است، بطور پیوسته توالیهای پرچم را ارسال می کند. هر فریم باید حداقل سه فیلد (مجموعاً ۳۲ بیت) داشته باشد (البته منهای پرچمهای ابتدا و انتها).

فریمها بر سه نوعند: اطلاعاتی (Information)، سرپرستی (Supervisory)، و بدون شماره (Unnumbered). در شکل ۳-۲۵ فیلد کنترل هر یک از این سه نوع فریم را ملاحظه می کنید. این پروتکل از تکنیک پنجره لغزنده، با شماره های ترتیبی ۳-بیتی، استفاده می کند. در هر لحظه تا هفت فریم تصدیق نشده می تواند در بافر فرستنده وجود داشته باشد. فیلد Seq در شکل ۳-۲۵ (الف) شماره ترتیبی فریم را نشان می دهد. فیلد Next نیز فیلد سواری مجانی برای تصدیق دریافت است. با این حال، در تمام انواع پروتکلهای HDLC مرسوم است که بجای سوار کردن شماره آخرین فریم دریافت شده، شماره اولین فریمی که هنوز دریافت نشده (فریمی که گیرنده منتظر آن است) برگردانده شود. این دو روش هیچ مزیتی بر یکدیگر ندارند، و انتخاب یکی از آنها به طراح پروتکل بستگی دارد؛ البته مشروط باینکه همواره از یک روش استفاده کند.

فیلد P/F مخفف Poll/Final (سرکشی/پایان) است. این فیلد در کامپیوترها یا مودمهایی بکار می رود که به چندین ترمینال سرکشی (polling) می کنند. اگر این فیلد محتوی P باشد، کامپیوتر (یا مودم) ترمینال را دعوت به ارسال داده می کند. در تمام فریمهایی که ترمینال می فرستد، فیلد P/F مقدار P دارد (بجز در آخرین فریم، که مقدار آن F است).

در برخی از پروتکلها بیت P/F باعث می شود تا ماشین طرف مقابل بلافاصله فریم سرپرستی را بفرستد، و منتظر سواری مجانی نشود. در ارتباطاتی که از فریمهای بدون شماره سود می برند، نیز این بیت کاربرد دارد.

Bits	1	3	1	3
(الف)	0	Seq (توالی)	P/F	Next (بعدي)
(ب)	1	0	Type (نوع)	P/F
(ج)	1	1	Type (نوع)	Modifier (تغییر دهنده)

شکل ۳-۲۵. فیلد کنترل در یک (الف) فریم اطلاعاتی، (ب) فریم سرپرستی، (ج) فریم بدون شماره.

انواع مختلف فریمهای سرپرستی یا فیلد *Type* مشخص می شود. نوع 0 فریم تصدیق دریافت (که رسماً RECEIVE READY نامیده می شود) است، و مشخص می کند که گیرنده آماده دریافت فریم بعدیست. این فریم وقتی ارسال می شود که ترافیک جهت برگشت برای اجرای تکنیک سواری مجانی وجود نداشته باشد. نوع 1 فریم تصدیق دریافت منفی (که رسماً REJECT نامیده می شود) است، از آن برای تصدیق دریافت فریم با خطا استفاده می شود. در این حالت فیلد *Next* حاوی اولین فریمیست که درست دریافت نشده است (یعنی فریمی که باید دوباره ارسال شود). در اینجا فرستنده باید تمام فریمهای بعد از فریم *Next* را مجدداً بفرستد، و از این نظر شبیه پروتکل 5 است تا پروتکل 6.

نوع 2 فریم RECEIVE NOT READY است؛ این فریم اعلام می کند که تمام فریمهای قبل از *Next* درست دریافت شده اند، ولی خود *Next* خیر (و از این نظر شبیه RECEIVE READY است، با این تفاوت که جلوی ادامه ارسال را می گیرد). فریم RECEIVE NOT READY در واقع نوعی اعلام مشکل است از سوی گیرنده، مثلاً مشکل پُر شدن بافرها. بعد از برطرف شدن این وضعیت، گیرنده یکی از فریمهای کنترلی دیگر (مانند RECEIVE READY یا REJECT) را می فرستد.

نوع 3 فریم SELECTIVE REJECT است. این فریم فقط زمانی فرستاده می شود که گیرنده خواستار ارسال مجدد یک فریم خاص باشد؛ HDLC از این نظر شبیه پروتکل 6 است تا پروتکل 5، و برای مواقعی مفید است که اندازه پنجره ارسال نصف شماره ترتیبی باشد. بنابراین اگر گیرنده بخواهد فریمهای نامنظم را بافر کرده و فقط ارسال مجدد برخی از فریمهای معیوب را طلب کند، می تواند از SELECTIVE REJECT استفاده کند. این فریم کنترلی فقط در HDLC و ADCCP وجود دارد، ولی در SDLC و LAPB تعریف نشده است.

سومین نوع از فریمها، فریمهای بدون شماره (Unnumbered) است. این نوع گاهی کاربردهای کنترلی دارد، ولی در سرویسهای غیرقابل اعتماد غیرمتصل نیز می توان از آن برای انتقال داده استفاده کرد. برخلاف دو نوع دیگر، این نوع فریم عملکردهای متفاوتی در انواع پروتکل های بیت-گرا است. در این حالت پنج بیت برای تعیین کارکرد فریم وجود دارد، ولی تمام ۳۲ حالت آن استفاده نمی شود.

در تمام پروتکلها فرمانی وجود دارد بنام DISC (قطع ارتباط) که اجازه می دهد یک ماشین خاموش شدن خود را به ماشینهای دیگر اعلام کند. فرمان دیگری وجود دارد که به یک ماشین اجازه می دهد تا بازگشت خود را اعلام کرده، و تمام شماره های ترتیبی را به 0 برگرداند؛ این فرمان SNRM (ست شدن حالت پاسخ عادی - Set Normal Response Mode) نام دارد. متأسفانه، «حالت پاسخ عادی» هر چیزی هست جز عادی. این حالت عبارتست از یک رابطه نامنتظر، که در آن یک سر خط «ارباب» و سر دیگر خط «رعیت» است. این فرمان از زمانی

به جای مانده که یک کامپیوتر بزرگ مرکزی وجود داشت و تعداد زیادی ترمینال به آن متصل می شد، که در این حالت مسلماً رابطه نامتقارن اریاب و رعیتی «عادی» محسوب می شود. برای آن که این پروتکلها بتوانند پاسخگوی نیازهای جدید (رابطه متقارن) باشند، در HDLC و LAPB فرمان دیگری وجود دارد بنام SABM (ست شدن حالت آسکرون متعادل - Set Asynchronous Balanced Mode)، که خط را ریست کرده و دو سر آنرا به حالت هم ارز و متعادل درمی آورد. در این پروتکلها دو فرمان دیگر وجود دارد بنامهای SABME و SNRME که بترتیب با SABM و SNRM متناظر هستند، و فقط در آنها شماره ترتیبی فریمها (بجای ۳-بیتی) ۷-بیتی است. فرمان سومی که در تمام پروتکلها وجود دارد، FRMR (FRaMe Reject) است که نشان می دهد جمع تطبیقی فریم صحیح است ولی از نظر شکلی درست نیست. از میان چنین شکلهای نادرستی می توان به فریم سرپرستی نوع 3 در LAPB، فریمی که کوتاهتر از ۳۲ بیت است، و تصدیق دریافت فریمی که خارج از پنجره دریافت است، اشاره کرد. در فریم FRMR یک فیلد داده ۲۴ بیتی وجود دارد که علت خطا را توضیح می دهد. اطلاعاتی که در این فیلد مخابره می شود، عبارتند از: فیلد کنترل فریم معیوب، پارامترهای پنجره دریافت، و اطلاعاتی درباره طبیعت خطا.

احتمال خراب یا گم شدن فریمهای تصدیق دریافت نیز مانند سایر فریمها وجود دارد، بنابراین برای آنها نیز باید نوعی تصدیق دریافت پس فرستاد. برای این منظور فریم کنترلی خاصی بنام UA (تصدیق دریافت بدون شماره - Unnumbered Acknowledgement) در نظر گرفته شده است. از آنجائیکه همیشه فقط یک فریم تصدیق دریافت تصدیق نشده در گیرنده وجود دارد، هیچ ابهامی وجود ندارد که منظور از یک فریم UA کدام فریم تصدیق دریافت است.

سایر فریمهای کنترلی کارهایی مانند آماده سازی (initialization)، سرکشی (polling) و گزارش وضعیت (status report) انجام می دهند. فریم کنترلی دیگری نیز وجود دارد بنام UI (Unnumbered Information)، که می توان با آن اطلاعات دلخواه ارسال کرد؛ این اطلاعات برای لایه پیوند داده هستند، و به لایه شبکه داده نمی شوند.

پروتکل HDLC علیرغم کاربرد وسیع آن، بهیچوجه کامل نیست. در (Fiorini et al., 1994) می توانید بحثی درباره مشکلات این پروتکل را ببینید.

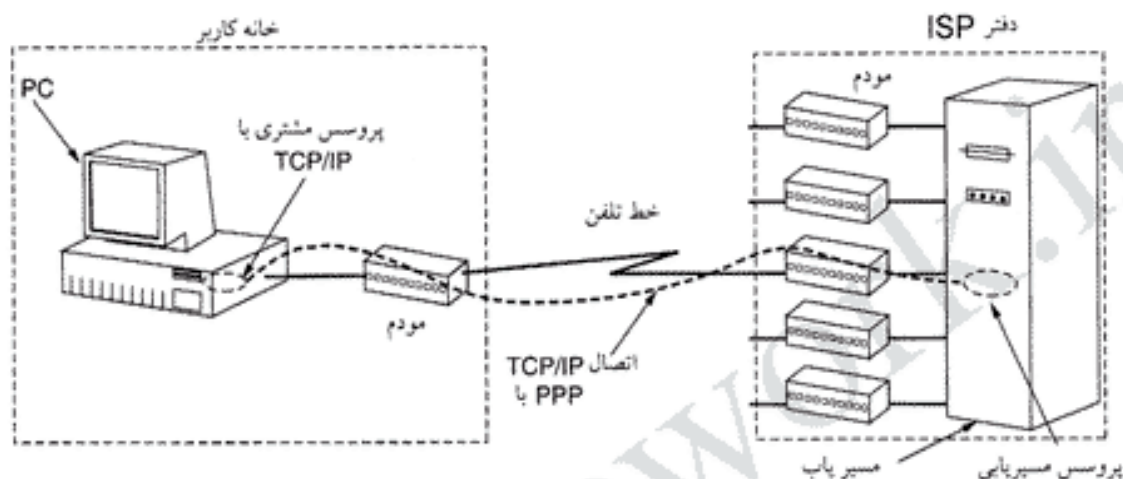
۲-۶-۳ لایه پیوند داده در اینترنت

اینترنت شامل ماشینهای متعددی (میزبان و مسیریاب) است، که توسط یک ستون فقرات به یکدیگر متصل می شوند. در یک ساختمان کوچک می توان از تکنیکهای LAN برای ارتباط استفاده کرد، ولی در اینترنت اغلب ارتباطات از نوع نقطه-به-نقطه (point-to-point) است. در فصل ۴ درباره لایه پیوند داده در LAN صحبت خواهیم کرد؛ در این قسمت به لایه پیوند داده در خطوط نقطه-به-نقطه می پردازیم.

در عمل، ارتباط نقطه-به-نقطه در دو حالت بکار برده می شود. اول، هزاران شرکت و مؤسسه دارای شبکه های محلی با تعداد زیادی ماشین میزبان (کامپیوترهای رومیزی، ایستگاههای کاری، کامپیوترهای سرور و غیره) و یک مسیریاب (یا یک پل - bridge) که در عمل همان وظیفه را انجام می دهد) هستند، و این مسیریابها در یک شبکه بزرگتر یکدیگر متصل شده اند. معمولاً این مسیریابها بوسیله ارتباط نقطه-به-نقطه و از طریق خطوط اجاره ای (leased line) به یک (یا دو) مسیریاب دیگر متصل می شوند. همین مسیریابها و خطوط اجاره ای هستند که زیر شبکه اینترنت را می سازند.

دومین حالتی که ارتباط نقطه-به-نقطه بکار برده می شود، میلیونها کاربر اینترنتی هستند که از منزل و با یک مودم (از طریق خط تلفن) به اینترنت متصل می شوند. اتفاقی که معمولاً می افتد اینست که کامپیوتر کاربر به

مسیریاب سرویس دهنده اینترنت (ISP) زنگ می زند، و از آن طریق (درست مثل یک میزبان معمولی) به اینترنت متصل می شود. در این روش فرقی نمی کند که خط تلفن معمولی است یا اجاره ای، فقط بعد از اینکه کاربر دیگر نیازی به آن نداشت، ارتباط قطع می شود. در شکل ۳-۲۶ این نوع ارتباط نقطه-به-نقطه را ملاحظه می کنید. مودمی که در این شکل نشان داده ایم یک مودم خارجی است، اما مودمهای داخلی نیز دقیقاً همان کار را انجام می دهند.



شکل ۳-۲۶. یک کامپیوتر خانگی می تواند نقش میزبان اینترنت را بازی کند.

در هر دو حالت (ارتباط مسیریاب-مسیریاب یا میزبان-مسیریاب) به یک پروتکل لینک داده نقطه-به-نقطه نیاز داریم تا وظایفی از قبیل فریم بندی، کنترل خطا و مانند آن را انجام دهد. پروتکلی که در اینترنت از آن استفاده می شود (و در این قسمت آنرا بررسی خواهیم کرد)، PPP نام دارد.

PPP - پروتکل نقطه-به-نقطه

اینترنت در موارد مختلفی، از قبیل ترافیک مسیریاب-به-مسیریاب یا ترافیک کاربر-به-ISP، به پروتکل نقطه-به-نقطه نیاز دارد. این پروتکل PPP (پروتکل نقطه-به-نقطه - Point-to-Point Protocol) نام دارد، که در RFC 1661 تعریف شده و در چند RFC دیگر (از قبیل RFC 1662 و RFC 1663) مشخصات آن بهبود یافته است. PPP ویژگیهای کنترل خطا دارد، از پروتکلهای مختلف پشتیبانی می کند، اجازه می دهد تا آدرس IP در زمان اتصال به طرف مقابل درخواست شود، تعیین هویت (authentication) انجام می دهد، و دهها ویژگی دیگر. مهمترین بخشهای PPP عبارتند از:

۱. یک روش فریم بندی، که ابتدا و انتهای فریمها را بوضوح مشخص می کند. فرمت فریم در PPP تشخیص خطا را نیز انجام می دهد.
۲. یک پروتکل کنترل لینک برای برقراری ارتباط، تست آن، مذاکره برای سایر گزینه ها، و در پایان قطع ارتباط بصورتی آبرومندانه. این پروتکل LCP (پروتکل کنترل لینک - Link Control Protocol) نام دارد.
۳. روشی برای ارتباط و مذاکره درباره گزینه های لایه شبکه، که از طرز کار این لایه مستقل است. در این روش برای هر نوع لایه شبکه یک NCP (پروتکل کنترل شبکه - Network Control Protocol) وجود دارد.

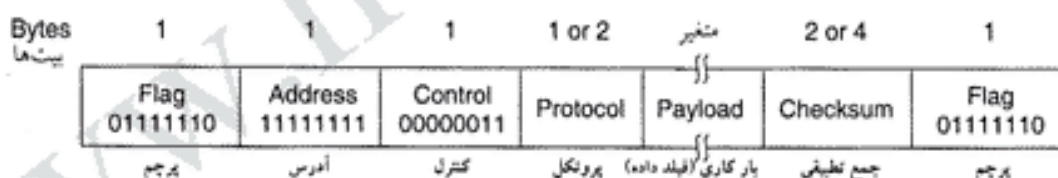
برای اینکه ببینید این قطعات چگونه با یکدیگر جفت می شوند، سناریوی ارتباط کاربر-به-ISP را در نظر می گیریم. ابتدا PC کاربر از طریق مودم خود به مسیریاب ISP زنگ می زند. بعد از اینکه مودم مسیریاب گوشی را

برداشت و ارتباط برقرار شد، PC از طریق فیلد داده یک یا چند فریم PPP چند بسته LCP به مسیریاب می فرستد. پارامترهای PPP از طریق همین بسته ها و پاسخ آنها انتخاب می شود.

بعد از آنکه بر سر این پارامترها توافق حاصل شد، یک سری بسته NCP برای پیکربندی لایه شبکه رد و بدل می شود. معمولاً PC هایی که به اینترنت متصل می شوند از TCP/IP استفاده می کنند، پس PC ما به یک آدرس IP نیاز دارد. تعداد آدرسهای IP آنقدر زیاد نیست که بتوان به همه آدرس ثابت اختصاص داد، به همین دلیل ISP تعدادی آدرس IP را بصورت دینامیک به PC هایی که به آن وصل می شوند، اختصاص می دهد. اگر یک ISP دارای n آدرس IP باشد، می تواند در هر لحظه (حداکثر) n کاربر متصل به اینترنت داشته باشد (البته تعداد کل مشترکان آن می تواند خیلی بیشتر باشد). یکی از بسته های NCP که درخواست IP می کند، این آدرس را به PC اختصاص می دهد.

از این لحظه به بعد PC ما درست مثل یک میزبان معمولی اینترنت است، و می تواند بسته های IP رد و بدل کند. وقتی کاربر ارتباط را قطع کند، NCP نیز ارتباط لایه شبکه را قطع کرده و آدرس IP را آزاد می کند. پس از آن LCP ارتباط لایه پیوند داده را قطع می کند، و کامپیوتر نیز به مودم دستور می دهد که گوشی را بگذارد و به ارتباط فیزیکی پایان دهد.

فرمت فریمهای PPP بسیار شبیه HDLC است (همیشه که نباید چرخ را از نو اختراع کرد). تفاوت اصلی PPP و HDLC اینست که، PPP کاراکتر-گرا (character-oriented) است نه بیت-گرا. بویژه، PPP از تکنیک لاگذاری بایت (byte stuffing) استفاده می کند، بنابراین تعداد بایتهای یک فریم همیشه عددی صحیح است. در PPP (برخلاف HDLC) نمی توان فریمی فرستاد که مثلاً 30.25 بایت داشته باشد. البته PPP می تواند (علاوه بر خطوط تلفن معمولی) روی SONET یا خطوط بیت-گرای HDLC نیز کار کند. فرمت فریم PPP را در شکل ۳-۲۷ ملاحظه می کنید.



شکل ۳-۲۷. فرمت فریم کامل PPP برای حالت بدون شماره.

تمام فریمهای PPP با بایت پرچم استاندارد HDLC (یعنی 01111110)، شروع می شوند، که اگر این بایت در داخل داده ها وجود داشته باشد از تکنیک لاگذاری بایت برای متمایز کردن آن استفاده می شود. بعد از آن فیلد Address می آید، که همیشه 11111111 است و مشخص می کند که تمام گیرنده ها باید این فریم را قبول کنند. با استفاده از این آدرس مشکل تخصیص آدرس های لینک داده نیز هم حل می شود.

بدنبال آدرس فیلد Control قرار دارد، که مقدار پیش فرض آن 00000011 است، و نشان می دهد که فریمها بدون شماره (unnumbered) هستند. عبارت دیگر، در PPP فریمها شماره ترتیبی ندارند، و از فریم تصدیق دریافت نیز خبری نیست. البته در محیط های پرنویز (مانند لینکهای بیسیم) می توان از فریمهای شماره دار استفاده کرد. جزئیات دقیق این حالت در استاندارد RFC 1663 مشخص شده، ولی در عمل بندرت از آن استفاده می شود.

از آنجائیکه در پیکربندی پیش فرض فیلدهای Address و Control همواره ثابت هستند، LCP مکانیزم خاصی را بین دو سر خط پیاده می کند که این دو بایت بکلی حذف و فریمها کوتاهتر شوند.

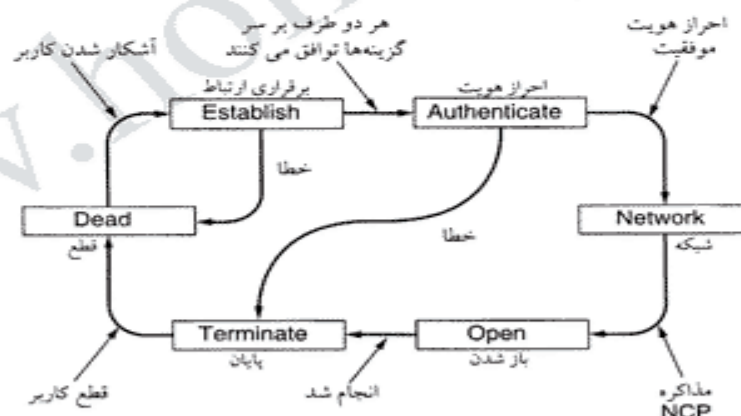
فیلد چهارم PPP فیلد Protocol است، و مشخص می کند که داده موجود در فریم (قسمت Payload) از چه

نوعیست. برای پروتکل‌های مختلف از قبیل LCP, NCP, IP, IPX, Apple Talk و پروتکل‌های دیگر گندهایی تعریف شده است. پروتکل‌هایی که با 0 شروع می‌شوند، پروتکل‌های لایه شبکه (مانند IP, IPX, OSI CLNP و XNS) هستند؛ آنهایی که با 1 شروع می‌شوند، برای مذاکره در باره پروتکل‌های دیگر بکار می‌روند (از جمله LCP, و یک NCP خاص برای هر یک از انواع لایه‌های شبکه). اندازه پیش‌فرض این فیلد 2 بایت است، ولی دو طرف می‌توانند از طریق LCP مذاکره کرده و اندازه آنرا به 1 بایت تقلیل دهند.

طول فیلد Payload متغیر است، و حداکثر مقدار آن در مذاکره اولیه مشخص می‌شود. اگر دو طرف در مذاکره اولیه (هنگام برقراری ارتباط) بر سر این عدد به توافق نرسند، از عدد پیش‌فرض 1500 بایت استفاده خواهد شد. اگر مقدار داده ارسالی به این حد نرسد، لایه پیوند داده بقیه را با کاراکتر خاصی پُر خواهد کرد. بدنبال Payload فیلد Checksum می‌آید، که مقدار آن معمولاً 2 بایت است، ولی دو طرف می‌توانند بر سر جمع تطبیقی 2 بایتی هم توافق کنند.

بطور خلاصه، PPP یک مکانیزم فریم‌بندی چندپروتکلی است، که می‌توان از آن روی مودم، خطوط بیت-گرای HDLC, SONET و سایر لایه‌های فیزیکی استفاده کرد. این پروتکل از کشف خطا، مذاکره برای گزینه‌های مطلوب، فشرده‌سازی سرآیند (header compression) (و در صورت نیاز، از ارتباط قابل اعتماد با فرمت HDLC) پشتیبانی می‌کند.

اکنون اجازه دهید ببینیم در PPP برقراری خط و قطع ارتباط چگونه انجام می‌شود. در شکل ۳-۲۸ مراحل (ساده شده) برقراری خط، بکارگیری، و قطع آن نشان داده شده است. این مراحل برای ارتباط مودمی و یا مسیریاب-به-مسیریاب هر دو صادق است.



شکل ۳-۲۸. مراحل ساده شده برقراری و قطع خط در پروتکل PPP.

در شروع کار پروتکل خط در وضعیت DEAD است، و این به معنای آنست که هیچگونه کاربر یا ارتباطی در لایه فیزیکی وجود ندارد. بعد از برقراری ارتباط در لایه فیزیکی، خط به وضعیت ESTABLISH می‌رود. در این لحظه مذاکره بر سر گزینه‌های LCP شروع می‌شود، که اگر موفقیت‌آمیز باشد، به وضعیت AUTHENTICATE منجر می‌شود. حال دو طرف می‌توانند در صورت تمایل هویت یکدیگر را چک کنند. بعد از ورود به مرحله NETWORK، لایه شبکه با اجرای پروتکل NCP مناسب پیکربندی می‌شود. اگر این پیکربندی موفقیت‌آمیز

باشد، مرحله *OPEN* فرا رسیده و تبادل اطلاعات می تواند شروع شود. وقتی تبادل اطلاعات انجام شد، خط به مرحله *TERMINATE* رفته، و از آنجا دوباره به وضعیت *DEAD* برمی گردد و کاربر قطع می شود. در مرحله *ESTABLISH*، مذاکره بر سر گزینه های پروتکل لینک داده توسط LCP انجام می شود. البته خود LCP هیچ علاقه ای به این گزینه ها ندارد، و فقط مکانیزمی برای مذاکره فراهم می آورد (بعبارت دیگر، پیشنهادی را فرستاده و بعد از دریافت پاسخ آنرا پذیرفته یا رد می کند). بررسی کیفیت خط (و اینکه آیا برای برقراری ارتباط اندازه کافی خوب هست یا نه) نیز بر عهده LCP است. قطع خط (بعد از پایان یافتن کار) یکی دیگر از وظایف پروتکل LCP است.

در RFC 1661 یازده نوع فریم LCP تعریف شده است، که آنها را در شکل ۳-۲۹ ملاحظه می کنید. چهار فریم *Configure-* به آغازکننده (I) اجازه می دهند تا گزینه ای را پیشنهاد کنند، و پاسخ دهنده (R) می تواند آنها را پذیرفته یا رد کند. اگر پاسخ دهنده گزینه ای را رد کند، می تواند پیشنهاد موردنظر خود را ارائه کرده، و یا اعلام کند که اساساً مایل نیست راجع به آن مذاکره کند. گزینه ها و پاسخ آنها جزئی از فریمهای LCP هستند.

نام	جهت	توضیح
Configure-request	I → R	لیست گزینه ها و مقادیر پیشنهادی
Configure-ack	I ← R	تمام گزینه ها قبول می شوند
Configure-nak	I ← R	بعضی از گزینه ها قبول نمی شوند
Configure-reject	I ← R	بعضی از گزینه ها مذاکره نمی شوند
Terminate-request	I → R	تقاضای قطع خط
Terminate-ack	I ← R	قبول، خط قطع شد
Code-reject	I ← R	دریافت درخواست نامعلوم
Protocol-reject	I ← R	درخواست پروتکل نامعلوم
Echo-request	I → R	لطفاً این فریم را پس بفرست
Echo-reply	I ← R	فریم پس فرستاده شد
Discard-request	I → R	این فریم را نادیده بگیر (برای تست بود)

شکل ۳-۲۹. انواع فریمهای LCP.

کدهای *Terminate-* برای قطع کردن خط (وقتی که دیگر نیازی به آن نیست) هستند. کدهای *Code-reject* و *Protocol-reject* مشخص می کنند که پاسخ دهنده چیزی را دریافت کرده که معنی آنرا نمی فهمد. یکی از دلایل این وضعیت می تواند رخ دادن خطاهای کشف نشده روی خط باشد، ولی با احتمال بیشتر دلیل آن یکی نبودن ویرایش پروتکل LCP در دو سمت خط است. فریمهای *Echo-* برای تست کیفیت خط بکار برده می شوند. و بالاخره، فریم *Discard-request* برای دیباگ کردن بکار می آید (و نویسنده پروتکل می تواند از آن برای تست پروتکل خود استفاده کند). گیرنده این قبیل فریمها را نادیده می گیرد، و هیچ عکس العملی به آنها نشان نمی دهد.

برخی از مهمترین گزینه هایی که می توان درباره آنها مذاکره کرد، عبارتند از: حداکثر اندازه قسمت *Payload* فریم، فعال کردن احراز هویت و انتخاب پروتکل آن، فعال کردن مانیتورینگ کیفیت خط در طول عملیات، و انتخاب گزینه های فشرده سازی سرآیند.

درباره پروتکلهای NCP حرف کلی چندانی نمی توان زد. هر پروتکل NCP خاص یک نوع لایه شبکه است، و بر سر گزینه های پیکربندی آن مذاکره می کند. برای مثال، در یک لایه شبکه IP احتمالاً مهمترین گزینه تخصیص آدرس IP است.

۷-۳ خلاصه

وظیفه لایه پیوند داده تبدیل استریم خام بیت‌های لایه فیزیکی به استریمی از فریمها، و هدایت این استریم به لایه شبکه است. روشهای مختلفی برای فریم‌بندی وجود دارد، که شمارش کاراکتر، لاگذاری بایت، و لاگذاری بیت از آن نمونه است. پروتکل‌های لینک داده می‌توانند خطاها را کنترل کرده، و در صورت نیاز فریمهای معیوب را مجدداً ارسال کنند. برای جلوگیری از غرق شدن گیرنده‌های گنبد در سیلاب داده‌های فرستنده‌های پرسرعت، لایه پیوند داده جریان داده‌ها را نیز کنترل می‌کند. یکی از پروتکل‌هایی که کنترل خطا و کنترل جریان در آن لحاظ شده، پروتکل پنجره لغزنده است.

پروتکل‌های پنجره لغزنده را می‌توان بر حسب اندازه پنجره ارسال و دریافت دسته‌بندی کرد. وقتی اندازه این پنجره‌ها هر دو 1 باشد، پروتکل توقف-انتظار نام دارد. اگر پنجره ارسال و دریافت (برای اجتناب از قفل شدن فرستنده در محیطهای با تأخیر زیاد) بزرگتر از 1 باشند، گیرنده می‌تواند فریمهای خارج از نظم را بکلی دور انداخته و یا آنها را بافر کند.

در این فصل چندین پروتکل را مورد بررسی قرار دادیم. پروتکل 1 برای محیطهای عاری از خطا طراحی شده، و می‌تواند انتقال اطلاعات را با هر سرعتی انجام دهد. در پروتکل 2 محیط همچنان بدون خطا فرض شده، ولی قادر است جریان داده‌ها را کنترل کند. در پروتکل 3 برای کنترل خطا از شماره ترتیبی فریمها و الگوریتم توقف-انتظار استفاده شده است. ترافیک در پروتکل 4 می‌تواند دوطرفه باشد، و در آن تکنیکی بنام سواری مجانی معرفی شده است. پروتکل 5 از تکنیک پنجره لغزنده «N» تا به عقب برگرد استفاده می‌کند. و بالاخره، در پروتکل 6 از تکنیکهای تکرار انتخابی و تصدیق دریافت منفی استفاده کردیم.

برای تست صحت و کارایی پروتکلها، روشهای مختلف مدل‌سازی (از جمله مدل ماشین حالت محدود و مدل شبکه پتری) را معرفی کردیم.

شبکه‌های بسیاری در لایه پیوند داده از پروتکل‌های بیت-گرا استفاده می‌کنند، که از آن میان می‌توان به SDLC، HDLC، ADCCP، یا LAPB اشاره کرد. در تمام این پروتکلها ابتدا و انتهای فریم با یک بایت پرچم مشخص می‌شود، و اگر این بایت در داده‌ها موجود باشد، از تکنیک لاگذاری بیت برای مشخص کردن آن استفاده می‌شود. همه این پروتکلها از یک پنجره لغزنده برای کنترل جریان سود می‌برند. در اینترنت نیز PPP بعنوان پروتکل اصلی لایه پیوند داده در خطوط نقطه-به-نقطه کاربرد گسترده‌ای دارد.

مسائل

- یک بسته از لایه بالاتر به ۱۰ فریم تقسیم شده، و احتمال اینکه هر یک از آنها سالم به مقصد برسد، ۸۰٪ است. اگر در لایه پیوند داده کنترل خطا انجام نشود، این پیام چند بار باید فرستاده شود تا تمام آن صحیح و سالم به مقصد برسد؟
- در لایه پیوند داده از گدگذاری زیر استفاده شده است:
A: 01000111; B: 11100011; FLAG: 01111110; ESC: 11100000
در هر یک از حالت‌های زیر، توالی بیت (باینری) فریم چهار کاراکتری A B ESC FLAG را نشان دهید:
(الف) شمارش کاراکتر.
(ب) بایت پرچم یا لاگذاری بایت.
(ج) بایت پرچم در ابتدا و انتها، یا لاگذاری بیت.
- با تکنیکی که در کتاب شرح داده شد، قطعه داده A B ESC C ESC FLAG FLAG D در وسط یک استریم ظاهر شده است. خروجی لاگذاری این قطعه چیست؟

۴. یکی از همکلاسی های شما عقیده دارد که قرار دادن یک بایت پرچم در ابتدای فریم و یکی در انتهای آن کار بیهوده ایست، و یک بایت کاملاً کفایت می کند (این بایت انتهای یک فریم، و ابتدای فریم بعدی محسوب خواهد شد). نظر شما چیست؟
۵. می خواهیم رشته 011110111110111110 را از طریق لایه پیوند داده ارسال کنیم. این رشته بعد از لاگذاری بیت به چه شکلی در می آید؟
۶. آیا در هنگام استفاده از لاگذاری بیت احتمال دارد که خطایی (از قبیل اضافه، کم شدن و یا تغییر یک بیت) رخ دهد، ولی جمع تطبیقی آنرا کشف نکند؟ اگر خیر، چرا؟ اگر بلی، چگونه؟ آیا طول جمع تطبیقی در اینجا نقشی دارد؟
۷. آیا فکر می کنید حالتی وجود دارد که یک پروتکل حلقه باز (مانند کد همینگ) بر پروتکل های مبتنی بر بازخورد (که در این فصل دیدید) ارجحیت داشته باشد؟
۸. برای اطمینان بیشتر، بجای یک بیت توازن از کدی استفاده می کنیم که یک بیت توازن برای بیت های فرد و یک بیت توازن دیگر برای بیت های زوج دارد. فاصله همینگ این کد چقدر است؟
۹. برای ارسال یک پیام ۱۶ بیتی از کد همینگ استفاده کرده ایم. برای آنکه گیرنده بتواند تمام خطاهای تک بیتی را کشف و تصحیح کند، به چند بیت چک کننده نیاز داریم. روش کار را برای پیام 1101001100110101 نشان دهید. فرض کنید در این کد همینگ از توازن زوج استفاده شده است.
۱۰. می خواهیم یک بایت ۸ بیتی با مقدار باینری 10101111 را با استفاده از روش همینگ توازن زوج کد کنیم. خروجی چیست؟
۱۱. یک کد همینگ ۱۲ بیتی با مقدار هگزادسیمال 0xE4F به گیرنده می رسد. مقدار هگزادسیمال اولیه چه بوده است؟ فرض کنید بیش از یک بیت خطا رخ نداده است.
۱۲. یکی از تکنیک های تشخیص خطا، ارسال داده ها بصورت ماتریسی از n سطر و k ستون است که هر سطر و ستون دارای بیت های توازن خاص خود است. آخرین بیت در منتهی الیه سمت راست بیتی است که سطر و ستون مربوطه را چک می کند. آیا این روش می تواند خطاهای تک بیتی را کشف کند؟ خطاهای دوبیتی را چگونه؟ خطاهای سه بیتی را چگونه؟
۱۳. ماتریسی با n سطر و k ستون دارای بیت های توازن افقی و عمودی است. فرض کنید در هنگام انتقال اطلاعات ۴ بیت تغییر کرده است. احتمال کشف نشدن این خطا را بصورت یک عبارت ریاضی استخراج کنید.
۱۴. حاصل تقسیم $x^7 + x^5 + 1$ بر چندجمله ای مولد $x^3 + 1$ چیست؟
۱۵. می خواهیم با استفاده از تکنیک CRC استریم 10011101 را ارسال کنیم. چندجمله ای مولد $x^3 + 1$ است. استریم فرستاده شده چیست؟ فرض کنید بیت سوم از سمت چپ در حین ارسال معکوس می شود. نشان دهید که گیرنده می تواند این خطا را کشف کند.
۱۶. پروتکل های لینک داده همیشه CRC را بجای سرآیند در پی آیند پیام قرار می دهند. چرا؟
۱۷. یک کانال 4-kbps دارای تأخیر انتشار 20 msec است. تا چه اندازه فریمی کارایی پروتکل توقف-انتظار بیش از ۵۰٪ است؟
۱۸. یک ترانک T1 بطول 3000 km از فریم های 64 بیتی و پروتکل 5 استفاده می کند. اگر تأخیر انتشار $6 \mu\text{sec/km}$ باشد، تعداد بیت های شماره ترتیبی چقدر باید باشد؟
۱۹. آیا در پروتکل 3 فرستنده می تواند تایمری را که در حال کار است، از نو شروع کند؟ اگر بلی، در چه موقعیتی؟ اگر خیر، چرا؟
۲۰. فرض کنید در یک پروتکل پنجره لغزنده تعداد بیت های شماره ترتیبی آنقدر زیاد است که «برگشت»

- هرگز رخ نمی دهد. چه رابطه ای باید بین لایه های پنجره ها و اندازه پنجره (که ثابت، و در فرستنده و گیرنده یکی است) برقرار باشد؟
۲۱. اگر روال *between* در پروتکل 5 بجای $a \leq b < c$ شرط $a \leq b \leq c$ را چک کند، چه تأثیری روی درستی یا کارایی پروتکل خواهد گذاشت؟ توضیح دهید.
۲۲. در پروتکل 6 وقتی یک فریم به گیرنده می رسد، چک می کند که آیا شماره ترتیبی آن همانی است که باید باشد، و آیا *no_nak* مقدار *true* دارد یا خیر. اگر هر دو شرط *true* باشند، یک *NAK* فرستاده می شود؛ در غیر اینصورت، تایمر کمکی راه اندازی می شود. اگر قسمت *else* را حذف کنیم، چه تأثیری روی درستی پروتکل می گذارد؟
۲۳. فرض کنید حلقه *while* سه دستوری نزدیک به انتهای پروتکل 6 را حذف کرده ایم. آیا این کار بر درستی پروتکل اثر می گذارد، یا فقط کارایی آنرا تحت تأثیر قرار می دهد؟ توضیح دهید.
۲۴. فرض کنید بخش *case* مربوط به خطاهای جمع تطبیقی را از دستور *switch* پروتکل 6 حذف کرده ایم. این کار چه تأثیری بر عملکرد پروتکل خواهد گذاشت؟
۲۵. *frame_arrival* در پروتکل 6 بخشی برای *NAK* ها دارد. این بخش زمانی اجرا می شود که فریم ورودی یک *NAK* باشد و شرط دیگری وجود داشته باشد. سناریویی را شرح دهید که در آن وجود این شرط دوم الزامی باشد.
۲۶. فرض کنید می خواهید برای لایه پیوند داده خطی برنامه بنویسید که در آن داده ها فقط به سمت شما می آیند، و شما هیچ چیزی نمی فرستید. سمت مقابل از پروتکل HDLC با شماره ترتیبی 3 بیتی، و پنجره 7 فریمی استفاده می کند. برای بالا بردن کارایی سیستم، تصمیم گرفته اید حداکثر فریمهای خارج از نظم ممکنه را بافر کنید، ولی مجاز به دستکاری در نرم افزار سمت فرستنده نیستید. آیا می توان پنجره دریافتی بزرگتر از 1 داشت، و تضمین کرد که این پروتکل هرگز با شکست مواجه نشود؟ اگر بلی، بزرگترین پنجره ای که می توان با اطمینان بکار برد، چقدر است؟
۲۷. پروتکل 6 را روی یک خط 1-Mbps عاری از خطا در نظر بگیرید. حداکثر اندازه فریم 1000 بیت است. بسته ها با فواصل یک ثانیه ای تولید می شوند. فاصله زمانی انقضای تایمر 10 msec است. اگر تایمر مخصوص تصدیق دریافت را حذف کنیم، انقضاهای غیر لازم رخ خواهد داد. یک پیام با طول متوسط چند بار باید مجدداً ارسال شود؟
۲۸. در پروتکل 6 داریم: $MAX_SEQ = 2^n - 1$. با اینکه این شرط برای استفاده بهینه از بیت های سرآیند آشکارا مناسب است، نشان ندادیم که الزامی هم هست. اگر، برای مثال $MAX_SEQ = 4$ ، آیا باز هم این پروتکل بدرستی کار می کند؟
۲۹. با استفاده از یک کانال ماهواره ای 1-Mbps که زمان رسیدن سیگنال از زمین به ماهواره 270 msec است، فریمهای 1000 بیتی ارسال می کنیم. فریمهای تصدیق دریافت همیشه با سواری مجانی برمی گردند، و سرآیند فریمها بسیار کوتاه است. حداکثر نرخ مصرف قابل دستیابی در پروتکل های زیر چقدر است؟ (الف) پروتکل توقف-انتظار. (ب) پروتکل 5. (ج) پروتکل 6.
۳۰. مقدار اتلاف پهنای باند پروتکل 6 را (مربوط به سرآیند و ارسال مجدد) در یک کانال ماهواره ای 50-kbps پرتوافیک، با فریمهایی متشکل از 40 بیت سرآیند و 3960 بیت داده، محاسبه کنید. فرض کنید زمان رسیدن سیگنال از زمین به ماهواره 270 msec است؛ فریمهای ACK هرگز ارسال نمی شوند؛

- فریمهای NAK دارای طولی معادل 40 بیت هستند؛ نرخ خطا در فریمهای داده ۱٪، و در فریمهای NAK قابل چشم پوشی است؛ و شماره های ترتیبی 8 بیتی هستند.
۳۱. می خواهیم روی یک کانال ماهواره ای عاری از خطا با ظرفیت 64 kbps فریمهای 512 بیتی (در یک جهت)، با فریمهای تصدیق دریافت بسیار کوتاه (در جهت دیگر)، بفرستیم. اگر اندازه پنجره 1، 7، 15 و 127 باشد، حداکثر ظرفیت خط چقدر خواهد بود؟ زمان ارسال سیگنال از زمین به ماهواره را 270 msec بگیرد.
۳۲. یک کابل T1 بطول 100 km را در نظر بگیرید. سرعت انتشار امواج در این کابل 2/3 سرعت نور در خلأ است. چند بیت در این کابل جا می شود؟
۳۳. فرض کنید پروتکل 4 را با ماشین حالت محدود مدل کرده ایم. هر ماشین چند حالت دارد؟ کانال مخابراتی چند حالت دارد؟ سیستم کامل (دو ماشین و یک کانال مخابراتی) چند حالت دارد؟ از خطاهای جمع تطبیقی صرف نظر کنید.
۳۴. توالی آتش شبکه پتری شکل ۳-۲۳ که با توالی حالت (010)، (01A)، (01-)، (01A)، (000) در شکل ۳-۲۱ متناظر است، را در نظر بگیرید. توضیح دهید که این توالی چه چیزی را نشان می دهد.
۳۵. شبکه پتری دستورات گذار $AC \rightarrow B$ ، $AC \rightarrow B$ ، $AC \rightarrow B$ ، $CD \rightarrow E$ ، و $E \rightarrow CD$ را رسم کنید. از این شبکه پتری گراف دسترسی حالت محدود برای رسیدن به حالت ACD را رسم کنید. این گراف چه مفهوم شناخته شده ای را مدل می کند؟
۳۶. پروتکل PPP از HDLC مشتق شده، که در آن برای جلوگیری از تفسیر اشتباه بایت پرچم در داده ها از تکنیک لاگذاری بیت استفاده می شود. یک دلیل پیابرد که چرا PPP از لاگذاری بایت استفاده می کند، نه لاگذاری بیت.
۳۷. حداقل سرباره یک بسته IP که با PPP فرستاده شده، چقدر است؟ فقط سرباره PPP را در نظر بگیرید، نه سرباره سرآیند IP را.
۳۸. هدف از این تمرین آزمایشگاهی پیاده سازی یک مکانیزم کشف خطا با استفاده از الگوریتم CRC (که در متن کتاب توضیح دادیم) است. دو برنامه بنویسید: مولد (generator)، و تست کننده (verifier). برنامه مولد یک رشته n بیتی از 0 ها و 1 ها را از ورودی استاندارد (با فرمت ASCII) می خواند. خط دوم یک چند جمله ای k بیتی است، که آن هم بصورت ASCII از ورودی خواننده می شود. برنامه خروجی خود را، که تشکیل شده از $n + k$ بیت 0 و 1 (و در واقع همان ورودی کُد شده است)، روی خروجی استاندارد می نویسد. (برنامه مولد این خروجی را هم با فرمت ASCII بیرون می دهد). برنامه تست کننده خروجی مولد را گرفته، و با پیامی نشان می دهد که آیا این رشته درست است یا خیر. برای تست این دو برنامه، یک برنامه کمکی دیگر (بنام alter) بنویسید که یکی از بیت های خط اول را معکوس کند و سایر بیت ها را بهمان صورت باقی بگذارد (شماره بیتی که باید معکوس شود را - با آغاز شمارش از سمت چپ - بصورت آرگومان ورودی به این برنامه بدهید). با نوشتن
- ```
generator < file | verifier
```
- برنامه باید پاسخ «درست است» بدهد، اما با دستور
- ```
generator < file | alter arg | verifier
```
- باید پیام «درست نیست» بگیرد.
۳۹. برنامه ای بنویسید که رفتار یک شبکه پتری را شبیه سازی کند. این برنامه باید دستورات گذار، و لیستی از حالت های لایه شبکه (هنگام ارسال و دریافت یک بسته جدید) را بعنوان ورودی بخواند. برنامه باید از حالت اولیه (که آنرا هم از ورودی می خواند) یکی از گذارهای فعال را بصورت تصادفی آتش کند، و سپس چک کند که آیا هر یک از ماشینها ۲ بسته قبول می کند، بدون این که بین آنها یک بسته جدید بفرستد.

زیر لایه نظارت بر دسترسی به رسانه انتقال

به گونه ای که در فصل اول اشاره کردیم، شبکه ها را می توان به دو رده تقسیم بندی کرد: آنهایی که از اتصالات نقطه به نقطه استفاده می کنند و آنهایی که از کانالهای پخش فراگیر (Broadcast) بهره می گیرند. در این فصل به شبکه های پخش فراگیر و پروتکل های آنها خواهیم پرداخت.

در هر شبکه فراگیر مسئله اصلی آنست که وقتی برای دسترسی به کانال انتقال، رقابت وجود دارد چگونه می توان تعیین کرد که چه کسی باید از کانال استفاده کند. برای روشن تر شدن قضیه یک کنفرانس تلفنی را با حضور شش نفر، در نظر بگیرید که از طریق شش خط تلفن بهم متصل شده و نشستی را ترتیب داده اند و هر کدام قادرند با دیگران گفت و شنود داشته باشند. در چنین وضعیتی کاملاً طبیعی است که وقتی یک نفر صحبت خود را قطع می کند دو یا چند نفر به طور همزمان شروع به حرف زدن نمایند و برای لحظاتی بی نظمی و اغتشاش بر جلسه حاکم شود. در ملاقات های رو در رو یکمک اشاره و علامت، از بروز بی نظمی احتراز می شود؛ مثلاً شخص مایل به گفتگو، دست خود را به علامت اجازه خواستن برای شروع سخن، بالا می برد. وقتی فقط یک کانال منفرد در اختیار است، تعیین نفر بعدی برای ارسال، دشوارتر خواهد بود. برای حل این مسئله پروتکل های متعددی عرضه شده است که معرفی آنها شاکله این فصل را تشکیل می دهد. در ادبیات شبکه، کانال های فراگیر گاهی با عناوین «کانال های با دسترسی چندگانه» (Multiaccess Channel) یا «کانال های با دسترسی تصادفی» (Random Access Channel) معرفی می شوند.

پروتکل هایی که برای تعیین نفر بعدی در استفاده از کانال مشترک کاربرد دارند متعلق به زیر لایه ای از لایه پیوند داده ها هستند که اصطلاحاً زیر لایه MAC (Medium Access Control) نامیده می شود. زیر لایه MAC در شبکه های محلی که اغلب آنها از کانال های مشترک به عنوان زیربنای ارتباط استفاده می کنند، از اهمیت ویژه ای برخوردار است. در مقابل، به غیر از شبکه های ماهواره ای، تمام شبکه های WAN از خطوط نقطه به نقطه بهره گرفته اند [و در آنها زیر لایه MAC جایگاهی ندارد]. از آنجایی که کانال های با دسترسی چندگانه و شبکه های LAN کاملاً به یکدیگر مرتبط هستند لذا در این فصل بطور عام شبکه های LAN را بررسی می کنیم و موارد معدودی را نیز که مستقیماً مربوط به زیر لایه MAC نمی شوند، بررسی خواهیم نمود.

از دیدگاه فنی، زیر لایه MAC بخش زیرین لایه پیوند داده ها محسوب می شود و متفقاً می بایست قبل از آنکه در فصل سوم به پروتکل های نقطه به نقطه بپردازیم ابتدا این زیر لایه را بررسی می کردیم. ولیکن برای اغلب افراد، فهم پروتکل های دوطرفه ساده تر از پروتکل های چند طرفه است. [منظور از پروتکل دوطرفه پروتکلیست که در آن فقط و فقط دو ماشین درگیر مبادله داده هستند.] به همین دلیل در روند تشریح مطالب (که از لایه های پائین به

سمت بالا خواهد بود)، اندکی از این ترتیب تخطی کردیم.

۱.۴ مسئله تخصیص کانال

مضمون اصلی این فصل آنست که چگونه یک کانال مشترک و فراگیر را بین کاربران رقیب تقسیم نمائیم. در ابتدا به الگوهای تخصیص «ایستا» و «پویا» نگاهی می اندازیم؛ سپس به تشریح تعدادی از الگوریتمهای خاص در این زمینه خواهیم پرداخت.

۱.۴.۱ تخصیص ایستای کانال در شبکه های LAN و MAN

روش سنتی در تخصیص یک کانال منفرد، (مثل خطوط اصلی تلفن - Telephone Trunk)، بین چندین کاربر که برای استفاده از آن رقابت می کنند، روش FDM (تسهیم در حوزه فرکانس) است. اگر N کاربر حضور داشته باشند، پهنای باند کانال به N بخش متساوی تقسیم می شود و به هر کاربر یکی از این بخشها اختصاص داده می شود (شکل ۳۱-۲ را ببینید). از آنجایی که هر کاربر دارای یک باند فرکانس اختصاصی است لذا کاربران هیچگونه تداخل و مزاحمتی برای یکدیگر ندارند. وقتی تعداد کاربران ثابت و کم باشد و هر کدام نیز دارای بار سنگین (و بافر شده) ترافیک باشند (همانند مراکز سوییچ تلفن)، روش FDM مکانیزمی ساده و کارآمد برای تخصیص کانال خواهد بود. ولیکن وقتی تعداد ارسال کنندگان زیاد و دائماً در حال تغییر باشد، یا ترافیک ارسالی آنها به صورت لحظه ای و انفجاری تولید شود، FDM مشکلات متعددی را بروز خواهد داد: اگر طیف فرکانسی کانال به N بخش مجزا تقسیم شود ولی تعداد کاربرانی که تمایل به ارسال و مخابره داده دارند کمتر از N باشد بخش ارزشمندی از این طیف فرکانسی تلف خواهد شد. اگر تعداد کاربران بیش از N باشد، برخی از آنها بدلیل کمبود پهنای باند، مجوز ارسال نخواهند داشت؛ حتی اگر برخی از کاربرانی که بدانها باند فرکانسی تخصیص داده شده به ندرت بخواهند چیزی ارسال یا دریافت کنند.

با این وجود، حتی اگر بتوان تعداد کاربران را عدد ثابت N فرض کرد، باز هم تقسیم ایستای طیف فرکانسی کانال منفرد به تعدادی زیرکانال، ذاتاً روشی ناکارآمد و بی کفایت تلقی می شود. مشکل اساسی آنست که وقتی برخی از کاربران، تقاضای ارسال نداشته باشند پهنای باند آنها هدر می رود. گذشته از آن در بسیاری از سیستم های کامپیوتری، ترافیک داده ها بشدت انفجاری است (نسبت حداکثر ترافیک به متوسط ترافیک در حدود 1000:1 است). در نتیجه اکثر اوقات کانالها خالی و بلااستفاده باقی می ماند.

کارآئی بسیار ضعیف روش FDM را می توان با یک محاسبه ساده در «نظریه صف» (Queuing Theory) اثبات کرد. برای شروع، فرض کنید بخواهیم زمان متوسط تاخیر T را برای کانالی محاسبه کنیم که در آن نرخ ارسال C بیت بر ثانیه، نرخ دریافت فریمها λ فریم بر ثانیه و طول هر فریم تصادفی است و از تابع چگالی احتمالی نمایی با میانگین $1/\mu$ بیت بر فریم تبعیت می کند.

با این پارامترها، نرخ دریافت فریمها λ فریم بر ثانیه و نرخ سرویس دهی $\mu.C$ فریم بر ثانیه خواهد بود. با استفاده از نظریه صف، می توان نشان داد که اگر نرخ دریافت و زمان سرویس دهی به فریمها از تابع توزیع پواسون تبعیت کند خواهیم داشت:

$$T = \frac{1}{\mu.C - \lambda}$$

بعنوان مثال اگر C معادل 100 مگابیت بر ثانیه، متوسط طول فریمها (یعنی $1/\mu$) معادل 10000 بیت و نرخ دریافت فریمها (یعنی λ) معادل 5000 فریم بر ثانیه باشد، متوسط تاخیر هر فریم $T = 200 \mu s$ خواهد بود. دقت کنید که اگر از تاخیر صف (یعنی تاخیر انتظار فریم) صرف نظر می کردیم و زمان ارسال 10000 بیت را بر روی یک شبکه

100Mbps محاسبه می نمودیم به پاسخ اشتباه 100 میکروثانیه می رسیدیم. این نتیجه فقط زمانی صدق می کند که هیچ رقابتی در بدست گرفتن کانال وجود نداشته باشد. حال بیایید این کانال منفرد را به N زیرکانال مستقل با ظرفیت C/N بیت بر ثانیه تقسیم کنیم. در این حالت نرخ متوسط ورودی به هر یک از این زیرکانالها λ/N خواهد بود. با محاسبه مجدد T به دست می آوریم:

$$T_{FDM} = \frac{1}{\mu(C/N) - (\lambda/N)} = \frac{N}{\mu.C - \lambda} = N.T \quad \text{رابطه (۴-۱)}$$

متوسط تاخیر در FDM، N برابر بیشتر از زمانی است که تمام فریمها به ترتیب در یک صف مرکزی طولانی و مرتب شده، پشت سرهم ارسال شوند.

تمام استدلالات و مباحثاتی که در مورد روش FDM اعمال کردیم دقیقاً در مورد TDM نیز صدق می کند. به هر کاربر بصورت ثابت یکی از N برش زمانی (slot time) اختصاص داده می شود. اگر یک کاربر از برش زمانی تخصیص یافته به خود استفاده نکند، آن زمان بلااستفاده مانده و هدر خواهد رفت. با استفاده مجدد از مثال قبلی، اگر یک شبکه 100Mbps را به ده شبکه 10Mbps تقسیم کرده و هر کاربر از یکی از آنها استفاده کند، میانگین تاخیر از 200 میکروثانیه به 2 میلی ثانیه افزایش خواهد یافت.

از آنجا که هیچیک از روشهای معمول و ایستای تخصیص کانال در محیطهای با ترافیک انفجاری کار نخواهد کرد لذا در ادامه به بررسی روشهای پویا می پردازیم.

۲-۱-۴ تخصیص پویای کانال در LAN و MAN

قبل از آنکه به اولین روش از روشهای متعدّد تخصیص کانال بپردازیم دسته بندی و فرموله کردن مسائل و مشکلات تخصیص کانال مفید خواهد بود. کل کاری که باید برای تخصیص کانال انجام شود مبتنی بر پنج فرض اساسی است که در زیر تشریح شده اند:

۱. **مدل ایستگاه (Station Model):** این مدل شامل N ایستگاه مستقل (مثل کامپیوتر، تلفن یا دستگاه های مخابرات شخصی) است که در هر کدام از آنها یک برنامه یا کاربر، فریمهایی را برای ارسال تولید می کند. برخی از اوقات به ایستگاه ها، «پایانه» (یا ترمینال) نیز گفته می شود. احتمال آنکه در بازه زمانی ΔT فریمی تولید شود، $\lambda \Delta T$ است که در آن λ یک مقدار ثابت است (در حقیقت λ میانگین نرخ تولید فریمهای جدید است). به محض آنکه فریمی تولید گردد، ایستگاه متوقف شده و تا زمانی که آن فریم به صورت موفقیت آمیز ارسال نشود کاری انجام نمی دهد.

۲. **فرض کانال منفرد (Single Channel Assumption):** در این حالت تنها یک کانال منفرد و مشترک برای مخابرات داده در اختیار ایستگاه ها است. تمام ایستگاه ها می توانند اطلاعات خود را بروی این کانال بفرستند یا از آن دریافت کنند. از دیدگاه سخت افزاری تمام ایستگاه ها هم ارز و معادل یکدیگرند؛ اگرچه ممکن است در نرم افزار پروتکل به هر ایستگاه، اولویتی خاص داده شود.

۳. **فرض تصادم (Collision Assumption):** هرگاه دو فریم بطور همزمان [برروی کانال مشترک] ارسال شوند با یکدیگر تداخل کرده و سیگنال حاصل بی ارزش و نامعتبر خواهد بود. این رخداد اصطلاحاً «تصادم» (Collision) نامیده می شود. هر ایستگاه می تواند از وقوع تصادم آگاه گردد. فریمی که در حین ارسال آن تصادم رخ داده است باید از نو فرستاده شود. در این مدل هیچ خطائی به غیر از خرابی فریم در اثر تصادم لحاظ نمی شود.

۴. **الف: مدل زمان پیوسته (Continuous Time):** در این مدل، ارسال فریمها می تواند در هر لحظه از زمان

شروع شود و هیچگونه سیگنال ساعت مرکزی (سراسری) که زمان را به برشهای گسسته و مجزا تقسیم کند، وجود ندارد.

۴. ب: «مدل زمان گسسته» (Slotted time): در این مدل، زمان به برشهای گسسته و مستقلی تقسیم می شود و ارسال فریم همیشه باید در ابتدای یکی از این برشهای زمانی انجام گیرد. در هر برش زمان ممکن است صفر، یک، یا چند فریم ارسال شود که به ترتیب: صفر فریم به معنای بیکار و بلااستفاده ماندن آن برش زمانی، یک فریم به معنای ارسال موفق و چند فریم معادل تصادم خواهد بود.

۵. الف: شنود سیگنال حامل (Carrier Sense): در این مدل، ایستگاهها قبل از شروع به ارسال فریم خود، قادرند تشخیص بدهند که آیا کانال مشغول است یا آزاد؟ اگر ایستگاهی احساس کند که کانال مشغول است هیچگاه سعی در استفاده از آن نخواهد کرد مگر آنکه مجدداً کانال بیکار شود.

۵. ب: عدم شنود سیگنال حامل (No Carrier Sense): در این مدل، ایستگاهها قادر نیستند قبل از استفاده از کانال، آنرا بشنوند و سیگنال روی آنرا احساس کنند؛ لذا فقط پس از فرستادن فریم می توان تعیین کرد که آیا ارسال موفق بوده یا تصادم پدیده آمده است.

اندکی توضیح در خصوص برخی از فرضیات فوق مفید خواهد بود: اولین بند اذغان می دارد که ایستگاهها مستقل هستند و فریمها را با نرخ ثابتی تولید می کنند. [بعبارتی نرخ میانگین تولید فریمها ثابت است. -م] همچنین در این بند تلویحاً فرض شده که هر ایستگاه تنها یک کاربر یا برنامه فعال دارد و بدین ترتیب هرگاه یک ایستگاه، متوقف (بلوکه) شود هیچ فریم جدیدی تولید نخواهد شد. مدلهای پیچیده دیگری نیز وجود دارد که در آنها ایستگاهها اجازه دارند به صورت چندبرنامه ای (Multiprogrammed) تولید فریمها را ادامه بدهند (حتی وقتی که ایستگاه در انتظار ارسال فریم قبلی بلوکه شده است)؛ ولیکن تحلیل چنین ایستگاههایی بسیار دشوار است. فرض کانال منفرد، هسته اصلی این مدل است. ایستگاهها بجز یک کانال واحد و مشترک راهی برای مبادله اطلاعات ندارند. ایستگاهها نمی توانند همانند یک کلاس درس یا بالا بردن دست خود، از معلم کلاس برای صحبت کردن اجازه بگیرند!

فرض تصادم نیز محوری است، اگر چه برخی از سیستمها (بویژه سیستمهای مبتنی بر «طیف گسترده» - Spread Spectrum) از این فرض مستثنی بوده و نتایج شگفت آوری نیز به همراه دارند؛ همچنین در شبکه های توکن رینگ (Token Ring) یک نشانه خاص (توکن) ایستگاه به ایستگاه می چرخد و هر ایستگاه که آنرا در اختیار بگیرد اجازه ارسال فریم خود را خواهد داشت ولیکن در بخشهای بعدی به کانالهای منفردی خواهیم پرداخت که برای استفاده از آنها رقابت وجود دارد و تصادم پدید می آید.

در خصوص زمان ارسال فریمها، دو فرض (۱) زمان پیوسته (۲) زمان گسسته قابل اعمال است. برخی از سیستمها از مدل اول پیروی می کنند و برخی دیگر از مدل دوم؛ بنابراین ما هر دو مدل را تحلیل خواهیم کرد. در هر سیستم فقط یکی از این مدلها قابل اعمال است.

همچنین در یک شبکه ممکن است سیگنال روی کانال احساس شود (فرض ۵-الف) یا شنود سیگنال میسر نباشد (فرض ۵-ب). شبکه های محلی (LAN) عموماً قادر به احساس سیگنال روی کانال هستند ولیکن در شبکه های بی سیم این مدل قابل استفاده نخواهد بود چراکه برخی از ایستگاهها در محدوده شنود سیگنال ایستگاههای دیگر نیستند. ایستگاههایی که به کانالهای سیمی متصل هستند در مدل «شنود سیگنال حامل» (Carrier Sense) قرار می گیرند و قادرند به محض کشف پدیده تصادم، ارسال فریم را خاتمه بدهند. کشف تصادم در شبکه های بی سیم به دلایل فنی مهندسی به ندرت قابل انجام است. دقت کنید که کلمه «حامل»

(Carrier) در اینجا اشاره به یک سیگنال الکتریکی بر روی کابل دارد.

۲-۴ پروتکل های دسترسی چندگانه

الگوریتم های بیشماری در مورد تخصیص کانال های با دسترسی چندگانه [کانال های مشترک] معرفی شده اند. در بخش های آتی نمونه هایی از جالبترین این پروتکل ها را بررسی کرده و مثال هایی از کاربرد آنها ارائه خواهیم نمود.

۱-۲-۴ ALOHA

در آوان ۱۹۷۰، نورمن أبرامسون و همکاران او در دانشگاه هاوانی روشی جدید و جالب برای حل مسئله تخصیص و دسترسی به کانال های اشتراکی ابداع کردند. بعداً، کار آنها توسط بسیاری از پژوهشگران ادامه یافت و تکمیل شد. (مراجع Abramson, 1985) اگر چه کار آقای أبرامسون که سیستم ALOHA نامیده شد براساس پخش امواج رادیویی زمینی بود، ولیکن نظریه آنها در هر سیستمی که در آن کاربران برای استفاده از یک کانال مشترک، به صورت ناهماهنگ رقابت می کنند، قابل اعمال و پیاده سازی است.

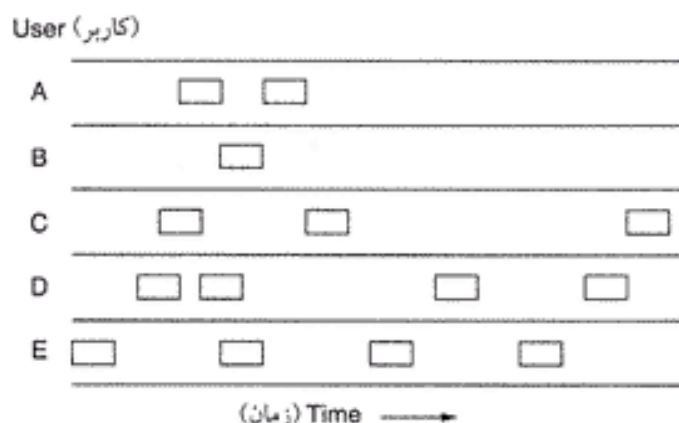
در اینجا دو نسخه متفاوت از ALOHA یعنی Pure ALOHA و Slotted ALOHA را تشریح خواهیم کرد. تفاوت این دو نسخه در تقسیم بندی زمان به برشهایی است که فریمها بتوانند در خلال یکی از آنها ارسال شوند. Pure ALOHA نیازی به هماهنگی زمانی (Time Synchronization) ندارد در حالیکه Slotted ALOHA نیازمند این هماهنگی است.

Pure ALOHA

ایده اصلی در سیستم ALOHA بسیار ساده است: کاربران اجازه دارند هر زمان که داده ای برای ارسال داشتند آنرا بفرستند. البته تصادمهایی رخ خواهد داد و فریمهایی که تصادم کنند از بین خواهند رفت. با این وجود در ALOHA، یک «کانال بازگشت سیگنال» (Feedback) وجود دارد که فرستنده با گوش دادن به این کانال، می تواند متوجه بروز تصادم و خرابی فریم شود. در شبکه های محلی LAN بروز تصادم به صورت سریع و آبی کشف می شود در حالیکه در شبکه های ماهواره ای یک ایستگاه پس از گذشت ۲۷ میلی ثانیه می تواند متوجه شود که آیا ارسال او موفق بوده یا نه! هرگاه به هر دلیلی امکان نشود سیگنال بازگشتی در حین ارسال وجود نداشته باشد بایستی دریافت فریمها توسط گیرنده تایید گردد. [با ارسال فریمهای مستقلى به نام Ack] هرگاه فریمی خراب شود، فرستنده آن، به اندازه یک زمان تصادفی صبر خواهد کرد و آن را مجدداً ارسال می کند. زمان انتظار قطعاً باید تصادفی باشد وگرنه تصادمها در یک دور نامتناهی تکرار خواهند شد. سیستمهایی که در آنها چندین کاربر از یک کانال مشترک به نحوی استفاده کنند که احتمال تصادم و تلاقی وجود دارد اصطلاحاً «سیستم های رقابتی» نامیده می شوند. در شکل ۴-۱ نمایشی از تولید فریم در سیستم ALOHA نشان داده شده است. در این شکل طول تمام فریمها را یکسان در نظر گرفته ایم زیرا اگر طول فریمها، اندازه ثابتی داشته باشند کارآیی و توان خروجی ALOHA حداکثر خواهد بود.

هرگاه دو فریم بطور همزمان بر روی کانال ارسال شوند، تصادم رخ داده و هر دو خراب خواهند شد. حتی اگر اولین بیت از یک فریم جدید با آخرین بیت از فریم قبلی تداخل کند هر دو فریم بطور کامل خراب شده و بعداً باید از نو ارسال شوند زیرا کدهای کشف خطا نمی توانند (و نباید هم بتوانند) تشخیص بدهند خطا در کجا اتفاق افتاده و بدین ترتیب کل فریم بلااستفاده خواهد بود.

سوال جالبی که پیش می آید آنست که کارآیی کانال در ALOHA چقدر است؟ عبارت دیگر می خواهیم بدانیم در این محیط نامنظم و هرج و مرج، چند درصد از کل فریمهای ارسالی از تصادم جان سالم به در می برند؟



شکل ۴-۱. در Pure ALOHA فریمها در زمانهای کاملاً دلخواه ارسال می شوند.

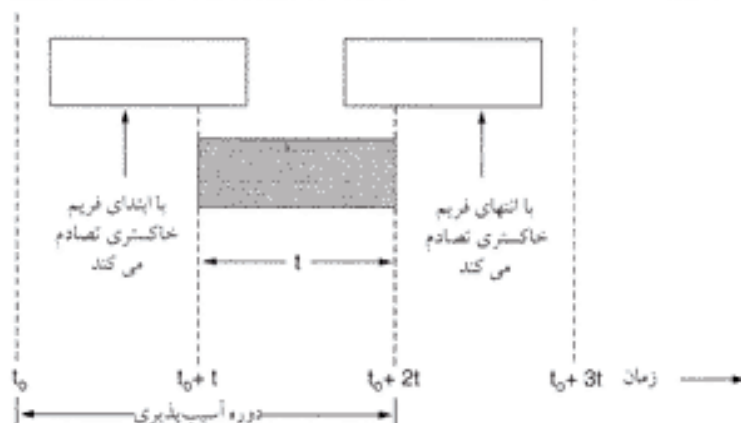
در ابتدا فرض می کنیم که تعدادی نامتناهی از کاربران در شبکه وجود دارند که پشت کامپیوترهای خود نشسته اند؛ هر کاربر در یکی از دو وضعیت «تایپ» یا «انتظار» قرار دارد. هرگاه تایپ یک خط به اتمام رسید و کلید Enter فشار داده شد، کاربر از تایپ دست می کشد و در انتظار پاسخ باقی می ماند. ایستگاه، فریم حاوی این خط را بلافاصله بر روی کانال می فرستد و بررسی می کند که آیا ارسال موفقیت آمیز بوده است؟ اگر فرآیند ارسال موفق باشد کاربر پاسخ خود را دریافت کرده و عمل تایپ را از سر می گیرد و در غیر اینصورت، کاربر باز هم منتظر می ماند تا ارسال فریم آنقدر تکرار شود تا بالاخره یکی از آنها به سلامت به مقصد برسد.

«زمان فریم» (Frame Time)، مقدار زمانی است که طول می کشد تا یک فریم با طول ثابت و استاندارد ارسال شود. (به عبارت دیگر این زمان معادل با طول فریم تقسیم بر نرخ ارسال خواهد بود). در اینجا فرض را بر آن می گذاریم که تعداد نامحدودی کاربر، مبتنی بر تابع توزیع پواسون و با میانگین N فریم در واحد زمان، فریم های جدید تولید می کنند. (واحد زمان در اینجا زمان لازم برای ارسال یک فریم است). فرض بی نهایت بودن کاربران از آن جهت لازم است که مطمئن باشیم وقتی یک کاربر متوقف و منتظر می شود از تعداد کاربران کاسته نخواهد شد. اگر $N > 1$ باشد مجموع کاربران با نرخی بیشتر از ظرفیت کانال، فریم تولید کرده اند و تقریباً تمام فریمها در اثر تصادم نابود خواهند شد. برای آنکه کارآیی قابل ملاحظه ای داشته باشیم انتظار می رود که $0 < N < 1$ باشد. هر ایستگاه علاوه بر فریمهای جدید خود، فریمهایی را که قبلاً در اثر تصادم خراب شده اند، نیز ارسال می کند.

اجازه بدهید فرض کنیم که احتمال «تلاش برای ارسال k فریم در واحد زمان» نیز از تابع توزیع پواسون با متوسط G تبعیت کند. (به خاطر داشته باشید که واحد زمان در اینجا زمان لازم جهت ارسال یک فریم -Frame Time- است). بدیهی است که $G \geq N$. [زیرا N متوسط تولید فریمهای جدید است در حالیکه G متوسط تولید فریمهای جدید و فریمهای قبلی خراب شده می باشد.]

در بار پائین (یعنی $N \approx 0$) تعداد تصادمها نیز اندک بوده و طبعاً ارسال مجدد فریمها ناچیز خواهد بود لذا $G \approx N$ است. در بار بالا تصادمهای زیادی رخ می دهد یعنی $G > N$ است؛ در هر شرایط بار، بازده مفید کانال (یعنی S) مساوی است با حاصل ضرب میزان بار (یعنی G) در احتمال موفقیت در ارسال (یعنی P_0) بنابراین داریم: $S = G \times P_0$ که در آن P_0 احتمال عدم خرابی یک فریم در اثر تصادم است.

فقط وقتی یک فریم در اثر تصادم خراب نخواهد شد که در زمان ارسال آن هیچ فریم دیگری ارسال نشود؛ به شکل ۴-۲ دقت کنید. تحت چه شرایطی فریمی که در شکل بصورت سایه دار نشان داده شده است سالم به مقصد خواهد رسید؟ t را زمان لازم برای ارسال یک فریم در نظر بگیرید. هرگاه کاربر دیگری در فاصله زمانی t_0 تا $t_0 + t$ فریمی را تولید و ارسال کرده باشد انتهای فریم او با ابتدای فریم سایه دار تصادم خواهد کرد. در حقیقت سرنوشت



شکل ۴-۲. دوره آسیب پذیری برای فریم خاکستری.

فریم سایه دار به گذشته نیز بستگی دارد، حتی قبل از آنکه اولین بیت آن ارسال شود؛ چراکه در سیستم Pure ALOHA، ایستگاه قبل از شروع به ارسال یک فریم قادر نیست به کانال گوش داده و متوجه شود که فریم دیگری در حال ارسال است. بدلیل مشابه اگر فریم جدیدی در فاصله زمان $t_0 + t$ تا $t_0 + 2t$ ارسال شود با انتهای فریم سایه دار در شکل تصادم خواهد کرد.

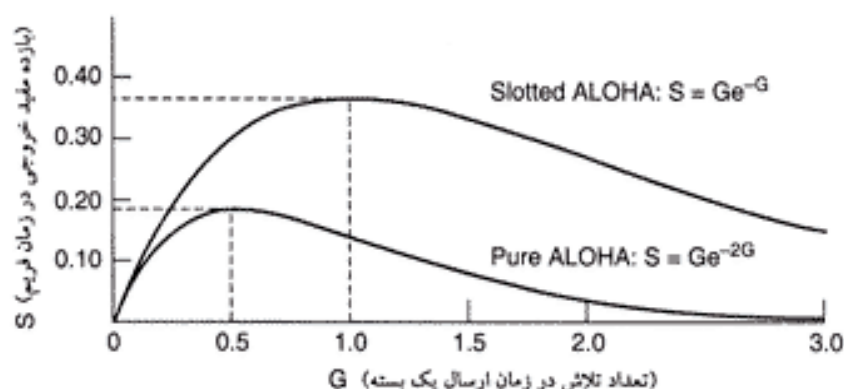
احتمال آنکه در زمان ارسال یک فریم [یعنی در زمان t] تعداد K فریم تولید شود از تابع توزیع پواسون تبعیت می کند یعنی:

$$\text{رابطه (۴-۲)} \quad \Pr[k] = \frac{G^k \cdot e^{-G}}{k!}$$

بدین ترتیب احتمال ارسال صفر فریم معادل با e^{-G} است. در فاصله زمان ارسال دو فریم [یعنی $2t$] میانگین فریم تولید شده معادل با $2G$ است. احتمال آنکه در طول «زمان آسیب پذیری» یک فریم، هیچ ترافیک دیگری تولید و ارسال نشود مساوی با $P_0 = e^{-2G}$ است. با اعمال رابطه $S = G \times P_0$ به دست خواهیم آورد:

$$S = G \times e^{-2G}$$

در شکل ۴-۳، رابطه بین ترافیک و بازده مفید کانال (Throughput) نشان داده شده است. بیشترین بازده به ازای $G = 0.5$ بدست می آید و در این حالت بازده کانال معادل $S = \frac{1}{2e}$ ، یعنی چیزی حدود 0.184 خواهد بود. به عبارت دیگر، بیشترین بهره مورد انتظار کانال (Channel Utilization) چیزی حدود ۱۸ درصد است. این مقدار بهره کانال چندان جالب نیست ولی در سیستمی که ایستگاهها در زمان دلخواه فریم خود را ارسال می کنند نمی توان انتظار داشت بهره کانال صددرصد باشد. [یعنی هیچ تصادفی اتفاق نیفتد].



شکل ۴-۳. بازده مفید کانال برحسب ترافیک عرضه شده در سیستم ALOHA.

Slotted ALOHA

در سال ۱۹۷۲، شخصی به نام روبرتز، روشی برای دو برابر کردن ظرفیت مفید سیستم ALOHA ارائه کرد. (مرجع Roberts, 1972) پیشنهاد وی مبنی بر آن بود که زمان به برشهای گسسته ای تقسیم شود و هر برش زمان معادل با زمان لازم برای ارسال یک فریم باشد. این روش مستلزم آن بود که کاربران محدوده این برشهای زمانی (Time Slots) را به درستی بدانند. برای رسیدن به چنین هماهنگی می توان یک ایستگاه خاص را به خدمت گرفت تا در ابتدای هر برش زمانی سیگنالی همانند سیگنال ساعت منتشر نماید.

در روش روبرتز که امروزه به نام Slotted ALOHA مشهور شده است، برخلاف روش Pure ALOHA هیچ کامپیوتری مجاز نیست وقتی که کلید Enter (Carriage Return) فشار داده شد، داده ها را بر روی کانال بفرستد؛ در عوض باید آنقدر منتظر بماند تا به آغاز برش زمان بعدی برسد. بنابراین روش پیوسته Pure ALOHA به روش گسسته Slotted ALOHA تبدیل شده است. از آنجایی که «دوره آسیب پذیری» (یعنی زمانی که در خلال آن نباید فریم دیگری ارسال شود) نصف شده است لذا احتمال آنکه هیچ داده دیگری در خلال یک برش (اسلات) تولید نشود e^{-G} خواهد بود؛ بدین ترتیب بدست می آوریم:

$$S = G \times e^{-G} \quad \text{رابطه (۳-۴)}$$

به گونه ای که از شکل ۳-۴ مشهود است، در سیستم Slotted ALOHA، بهره کانال در $G=1$ به مقدار حداکثر خود می رسد و در این نقطه بهره کانال معادل با $S = \frac{1}{e}$ (یعنی حدود ۰.۳۶۸) خواهد بود. اگر این سیستم در شرایط $G=1$ عمل کند احتمال آنکه یک برش زمانی خالی باشد [و بتوان ارسال موفق داشت] حدود ۰.۳۶۸ خواهد بود. (طبق رابطه ۳-۴) بیشترین موفقیتی که می توان از Slotted ALOHA انتظار داشت عبارتست از: ۳۷ درصد برای خالی ماندن یک اسلات، ۳۷ درصد برای ارسال موفق و ۲۶ درصد برای تصادم خواهد بود. اگر این سیستم با مقدار بیشتر G کار کند، تعداد برشهای خالی کاهش یافته و میزان تصادمها به صورت نمائی افزایش خواهد داشت. [G: تلاش برای ارسال G عدد فریم در واحد زمان است و واحد زمان نیز زمان لازم برای ارسال یک فریم می باشد.] برای آنکه بررسی کنیم افزایش سریع تعداد تصادمها با افزایش G ، از کجا منشاء می گیرد ارسال یک فریم آزمایشی را مد نظر قرار بدهید: احتمال آنکه این فریم از تصادم جان سالم به در ببرد (یعنی تمام ایستگاههای دیگر در این برش زمانی ساکت باشند) e^{-G} است و بالطبع احتمال تصادم معادل با $1 - e^{-G}$ خواهد بود. احتمال ارسال موفق فریم، منوط به k تلاش پیاپی خواهد بود (یعنی $k-1$ تصادم متوالی و نهایتاً یک ارسال موفق) یعنی:

$$P_k = e^{-G} \cdot (1 - e^{-G})^{k-1}$$

پس از فشار داده شدن کلید Enter، میانگین دفعات ارسال یعنی E ، معادل است با:

$$E = \sum_{k=1}^{\infty} k \cdot P_k = \sum_{k=1}^{\infty} k \cdot e^{-G} (1 - e^{-G})^{k-1} = e^{-G}$$

در نتیجه، از آنجایی که E به صورت نمائی با G ارتباط دارد اندکی افزایش در بار کانال، بهره کانال را بشدت کاهش خواهد داد.

روش Slotted ALOHA بدلائلی که در بدو امر چندان مشهود و روشن به نظر نمی رسد، از اهمیت ویژه ای برخوردار است. این روش در دهه ۱۹۷۰ ابداع گردید، در چند سیستم آزمایشی و ابتدائی به کار گرفته شد و پس از آن تقریباً به دست فراموشی سپرده شد ولیکن وقتی دسترسی به اینترنت از طریق کابل اختراع شد، ناگاه این مشکل بزرگ به میان آمد که چگونه می توان کانالی مشترک را به کاربران رقیب اختصاص داد. اینجا بود که Slotted ALOHA بار دیگر به صحنه آمد. پروتکلهایی بوده اند که اگرچه درست و موثر کار می کرده اند ولی بدلائل سیاسی [غیر علمی] کنار گذاشته شده اند (مثلاً برخی از شرکتهای بزرگ علاقمندند که همه دنباله روی عملکرد آنها باشند)

ولیکن سائها بعد اشخاص زیرک و با هوش بدین حقیقت می‌رسند که این پروتکل‌های فراموش شده می‌توانند مشکلات فعلی آنها را حل کنند. به همین دلیل در این فصل به پروتکل‌های زیبا و جالبی خواهیم پرداخت که در حال حاضر کاربرد گسترده‌ای ندارند ولیکن ممکن است در کاربردهای آتی مفید واقع شوند؛ بشرط آنکه طراحان شبکه نسبت به آنها آگاهی داشته باشند. البته ما به بررسی پروتکل‌های متعددی نیز پرداخته‌ایم که در حال حاضر کاربرد بسیار گسترده‌ای دارند.

۲-۲-۴ پروتکل‌های دسترسی چندگانه با قابلیت شنود سیگنال حامل (CSMA)

در روش Slotted ALOHA بیشترین بهره مفیدی که می‌توان بدست آورد $1/c$ [معادل 0.368] است. این بهره چندان جالب نیست چراکه در این روش هر ایستگاه بدون اعتنا به وضعیت بقیه ایستگاه‌ها و به دلخواه ارسال خود را انجام می‌دهد، لذا در این سیستم تعداد تصادمها زیاد خواهد بود. لیکن در شبکه‌های محلی امکان آن وجود دارد که هر ایستگاه بتواند تشخیص بدهد دیگر ایستگاه‌ها چه می‌کنند و بر اساس این تشخیص عملکرد خود را تنظیم نماید. در چنین شبکه‌هایی می‌توان به بهره کانال بسیار بالاتر از $1/c$ دست یافت. در این بخش چندین پروتکل برای افزایش کارآئی و بهره کانال معرفی می‌کنیم.

پروتکل‌هایی که در آنها هر ایستگاه به سیگنال حامل روی کانال گوش داده و براساس وضعیت کانال عمل می‌کنند اصطلاحاً «پروتکل‌های شنود حامل» (Carrier Sense Protocols) نامیده می‌شوند. تاکنون تعداد بی‌شماری از این پروتکل‌ها معرفی شده‌اند. کلین زاک و توباک (۱۹۷۵) معدودی از این پروتکل‌ها را تحلیل کرده‌اند. در ذیل چندگونه از پروتکل‌های مبتنی بر شنود سیگنال حامل را بررسی خواهیم کرد.

Persistent and Nonpersistent CSMA

اولین پروتکل مبتنی بر شنود سیگنال حامل که در اینجا بررسی خواهیم کرد، روش 1-Persistent CSMA است. در این روش هرگاه یک ایستگاه، داده‌ای برای ارسال داشته باشد ابتدا به کانال گوش می‌دهد تا ببیند آیا در این لحظه کسی دیگری در حال ارسال هست یا خیر. اگر کانال مشغول باشد ایستگاه آنقدر منتظر می‌ماند تا کانال آزاد شود؛ ولی اگر ایستگاه، کانال را آزاد تشخیص بدهد فریم خود را ارسال می‌کند. اگر تصادمی رخ بدهد ایستگاه به اندازه یک زمان تصادفی صبر کرده و تمام مراحل را از نو آغاز می‌نماید. این پروتکل، اصطلاحاً 1-Persistent (پروتکل پافشاری بر ارسال) نامیده می‌شود چراکه وقتی یک ایستگاه کانال را آزاد تشخیص بدهد با احتمال ۱ ارسال خود را آغاز می‌کند. [یعنی اگر ایستگاه کانال را آزاد تشخیص بدهد یقیناً و به صورت غیرمشرط ارسال خود را آغاز می‌نماید. -م]

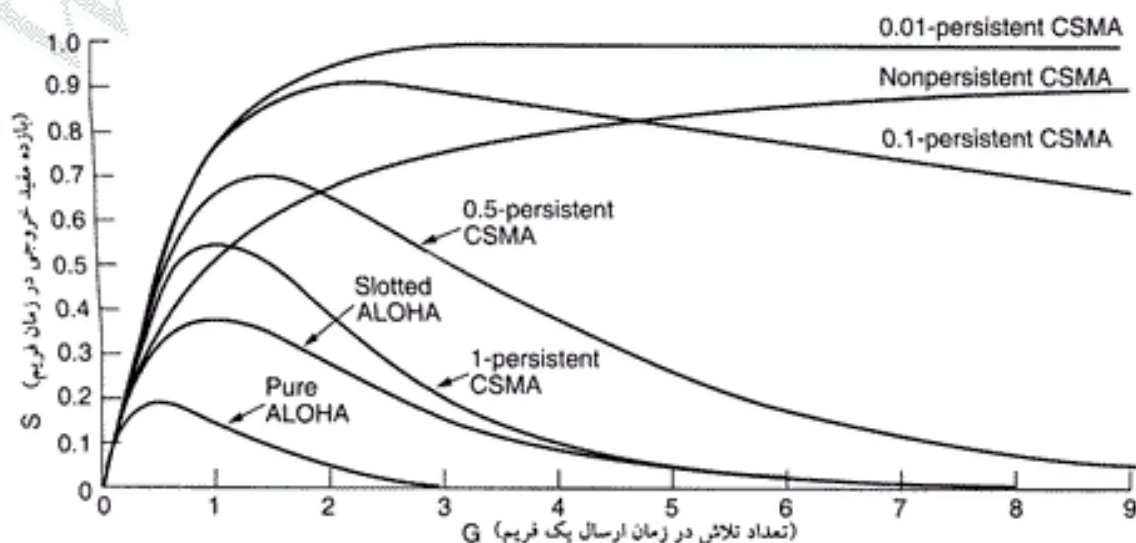
«تاخیر انتشار» (Propagation Delay) تأثیر بسزائی در کارآئی این پروتکل دارد. احتمال ناچیزی وجود دارد که دقیقاً پس از شروع ارسال فریم توسط یک ایستگاه، ایستگاه دیگری نیز آماده ارسال شده و کانال را بررسی و شنود نماید. اگر سیگنال ایستگاه اول هنوز به ایستگاه دوم نرسیده باشد [بدلیل تاخیر انتشار]، دومی نیز کانال را آزاد تشخیص داده و ارسال خود را آغاز می‌کند و طبعاً منجر به تصادم (Collision) خواهد شد. هر چه تاخیر انتشار بیشتر باشد تأثیر مخرب آن بیشتر و منجر به کارآئی بدتر پروتکل خواهد شد.

حتی اگر تاخیر انتشار صفر باشد باز هم «تصادم» وجود خواهد داشت؛ هرگاه دو ایستگاه در خلال ارسال ایستگاه ثانی آماده ارسال فریم شوند هر دوی آنها مودبانه منتظر خاتمه ارسال فریم جاری می‌شوند و به محض آزاد شدن کانال بطور همزمان ارسال فریم خود را آغاز می‌نمایند که منجر به تصادم خواهد شد. اگر این دو ایستگاه در ارسال فریم خود عجل و مضر نبودند تصادم‌های کمتری رخ می‌داد! با این حال این پروتکل بسیار بهتر از Pure ALOHA عمل می‌کند زیرا در این پروتکل دو ایستگاه [یا شنود کانال] از تلاقی با ارسال فریم ایستگاه در

حال ارسال اجتناب می کنند. می توان به صورت ذهنی کارآئی این روش را بسیار بیشتر از Pure ALOHA ارزیابی کرد. بدلیل مشابه، کارآئی این روش از Slotted ALOHA نیز بیشتر است. دومین پروتکل که آن نیز مبتنی بر شنود سیگنال است Nonpersistent CSMA نام دارد. در این پروتکل، تلاش آگاهانه جهت ارسال بر روی کانال با اصرار کمتری نسبت به پروتکل قبلی انجام می گیرد؛ هر ایستگاه قبل از ارسال، کانال را بررسی (شنود) می کند؛ اگر کسی دیگر در حال ارسال نباشد ایستگاه، فریم خودش را می فرستد ولیکن هرگاه کانال از قبل در اختیار دیگری باشد، ایستگاه بطور دائم به شنود کانال نخواهد پرداخت. در عوض [هرگاه ایستگاه کانال را مشغول تشخیص بدهد] به اندازه یک زمان تصادفی صبر کرده و پس از آن الگوریتم فوق را تکرار می کند. [تنها تفاوت این روش با روش قبلی در آنست که هرگاه کانال مشغول باشد ایستگاه مترصد آزاد شدن آن باقی نمی ماند و به اندازه یک زمان تصادفی کانال را به حال خود رها می کند و بدان گوش نمی دهد. -م] این الگوریتم طبعاً بهره کانال بهتری را ارائه می دهد [چرا که تصادمها در لحظه آزاد شدن کانال کاهش می یابد] ولیکن تاخیر بیشتری نسبت به روش 1-Persistent CSMA دارد. [تاخیر ارسال]

آخرین پروتکل، p-Persistent CSMA نام دارد. این روش فقط بر روی کانالهای زمان بندی شده (Slotted Time Channels) قابل اعمال است و بدین ترتیب عمل می کند: هرگاه ایستگاهی آماده ارسال شود ابتدا کانال را شنود می نماید؛ اگر کانال آزاد باشد فریم خود را با احتمال p ارسال می کند و یا به احتمال $q=1-p$ ارسال خود را تا فرا رسیدن برش بعدی زمان [اسلات بعدی] به تعویق می اندازد. یعنی حتی اگر یک اسلات خالی باشد ممکن است با احتمال p فریم خود را بفرستد یا با احتمال q به تعویق بیندازد. این فرآیند آنقدر تکرار می شود تا آنکه یا فریم ارسال شود یا آنکه ایستگاهی دیگر ارسال خود را آغاز نماید. در حالت دوم [یعنی ایستگاهی دیگر موفق به ارسال شود] ایستگاه ناموفق، همانند وقتی که تصادم رخ داده عمل می کند یعنی به اندازه یک زمان تصادفی صبر کرده و از نو شروع می نماید. اگر ایستگاه در همان ابتدا کانال را مشغول تشخیص بدهد تا اسلات بعدی صبر می کند و الگوریتم فوق را به اجرا می گذارد.

شکل ۴-۴ منحنی ظرفیت مفید (Throughput) کانال را بر مبنای حجم ترافیک تولید شده، برای سه پروتکل فوق و پروتکل های Pure ALOHA و Slotted ALOHA نشان می دهد.

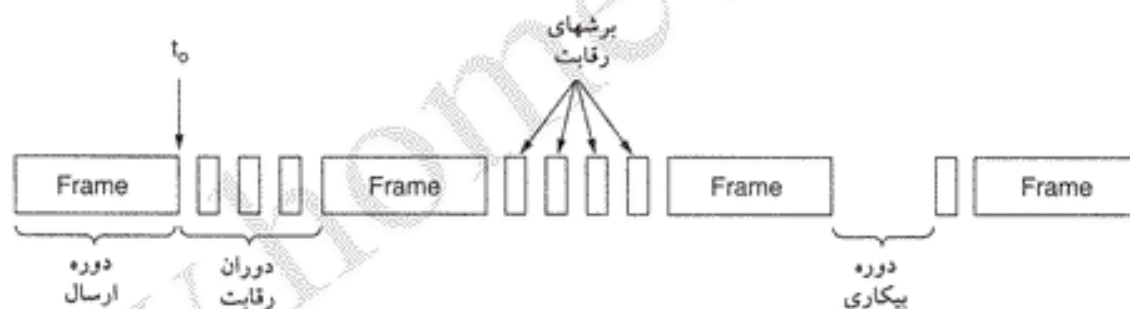


شکل ۴-۴. مقایسه بهره وری کانال (ظرفیت مفید) بر حسب بار برای پروتکل های گوناگون دسترسی تصادفی به کانال.

پروتکل CSMA با تشخیص تصادم

پروتکل‌های Persistent & Nonpersistent CSMA به روشنی بهینه‌تر از ALOHA هستند زیرا مطمئناً اگر ایستگاهی کانال را مشغول تشخیص بدهد ارسال خود را آغاز نخواهد کرد. بهبود دیگر این روشها آنست که ایستگاهها به محض آنکه از وقوع تصادم آگاه شدند ارسال خود را نیمه کاره رها کنند. به عبارت دیگر هرگاه دو ایستگاه، کانال را آزاد احساس کرده و همزمان شروع به ارسال نمایند، تقریباً هر دوی آنها بلافاصله از وقوع تصادم مطلع خواهند شد. در چنین حالتی به محض کشف پدیده تصادم، ایستگاهها بجای ارسال کامل فریمهای آسیب دیده، به ارسال خود خاتمه می‌دهند. قطع سریع ارسال فریمهای آسیب‌دیده، در زمان و پهنای باند صرفه‌جویی خواهد کرد. چنین پروتکلی که اصطلاحاً CSMA/CD نام دارد بطور گسترده در شبکه‌های محلی به کار گرفته شده است. به ویژه، این پروتکل مبنای شبکه محلی و شناخته شده اترنت است، لذا ارزش آنرا دارد که اندک زمان بیشتری برای بررسی جزئیات آن صرف کنیم.

CSMA/CD نظیر بسیاری از پروتکل‌های دیگر LAN از مدل مفهومی نشان داده شده در شکل ۴-۵ تبعیت می‌کند. در لحظه‌ای که با نماد t_0 مشخص شده، ایستگاهی ارسال فریم خود را به پایان رسانده است. در این لحظه ایستگاههایی که فریمی برای ارسال دارند ممکن است برای ارسال آن تلاش کنند. اگر دو یا چند ایستگاه بطور همزمان تصمیم به ارسال بگیرند تصادم رخ خواهد داد. وقوع تصادم را می‌توان با بررسی توان مصرفی یا اندازه‌گیری و مقایسه پهنای پالس سیگنال دریافتی از کانال و مقایسه آن با سیگنال ارسالی تشخیص داد.



شکل ۴-۵. CSMA/CD می‌تواند در یکی از سه وضعیت: «رقابت»، «ارسال» یا «بیکار» قرار داشته باشد.

پس از آنکه یک ایستگاه متوجه وقوع تصادم شد، ارسال خود را ناتمام رها کرده و از نو شروع می‌کند؛ (البته با این فرض که ایستگاه دیگری در این بینابین ارسال خود را آغاز ننماید). بدین ترتیب مدل ارائه شده برای CSMA/CD، شامل: (۱) چندین مرحله متناوب «رقابت» (Contention) (۲) بازه‌های ارسال و (۳) بازه‌های بیکاری خواهد بود. (بازه‌های بیکاری بدین معناست که تمام ایستگاهها بدلیل عدم نیاز به ارسال ساکت بوده‌اند). حال جزئیات الگوریتم رقابت را دقیقتر بررسی می‌نماییم: فرض کنید دو ایستگاه بطور همزمان ارسال فریم خود را در زمان t_0 آغاز نمایند. چقدر طول می‌کشد تا متوجه شوند تصادم اتفاق افتاده است؟ پاسخ این سوال برای تعیین طول زمان رقابت و همچنین محاسبه تاخیر و ظرفیت مفید کانال حیاتی است.

حداقل زمان لازم برای تشخیص وقوع تصادم، معادل زمانی است که طول می‌کشد تا سیگنال ارسالی از یک ایستگاه به ایستگاه دیگر منتشر شود. براساس این استدلال شاید شما تصور کنید که یک ایستگاه تا پس از زمانی معادل با زمان انتشار سیگنال بر روی کل کابل از وقوع تصادم مطلع نشده و از تصرف کانال، مطمئن نخواهد بود. (منظورمان از «تصرف» کانال آنست که ایستگاههای دیگر متوجه شوند که او در حال ارسال است و تداخلی پیش نیاید). این نتیجه‌گیری صحیح نیست. به سناریوی زیر که در بدترین شرایط در نظر گرفته شده، دقت نمایید: زمان

لازم برای انتشار سیگنال بین دورترین ایستگاه ها از یکدیگر را τ فرض کرده ایم؛ در لحظه t_0 یک ایستگاه ارسال خود را آغاز می کند. در لحظه $t_0 - \tau$ یعنی دقیقاً قبل از لحظه ای که سیگنال به دورترین ایستگاه می رسد، آن ایستگاه شروع به ارسال می کند. مشخصاً این ایستگاه وقوع تصادم را کشف می کند ولیکن سیگنال نویزی که در اثر تصادم تولید می شود تا زمان $t_0 - \tau$ به ایستگاه مبداء باز نخواهد گشت. به عبارت دیگر در بدترین حالت، تا انتقضای زمان 2τ پس از شروع ارسال، ایستگاه مبداء نمی تواند از تصرف کانال مطمئن باشد. به همین دلیل ما بازه رقابت را همانند روش Slotted ALOHA مدل کرده ایم که در آن پهنای هر برش زمانی (اسلات) معادل 2τ می باشد. بر روی یک کابل کوآکسیال به طول ۱ کیلومتر، 2τ حدوداً معادل ۵ میکروثانیه است. برای سادگی فرض را بر آن خواهیم گذاشت که هر برش زمان در برگیرنده تنها یک بیت [به طول 2τ] است و به محض آنکه کانال تصرف شد، ایستگاه می تواند با هر سرعت دلخواه ارسال خود را انجام بدهد ولیکن بدیهی است که با ارسال با نرخ $1\text{bit}/2\tau$ نخواهد بود!

درک این موضوع که فرآیند کشف تصادم به صورت آنالوگ انجام می شود اهمیت دارد. سخت افزار هر ایستگاه باید در حین ارسال به کابل گوش بدهد. اگر آنچه را که از کابل بازخوانی می کند با آنچه بر روی کابل قرار داده، متفاوت باشد متوجه وقوع تصادم می شود. کشف تصادم مستلزم آنست که روش کدینگ سیگنال این امکان را فراهم بیاورد (زیرا مثلاً تصادم دو سیگنال صفر ولت قابل کشف نیست). به همین دلیل معمولاً از روشهای کدینگ خاص [مثل منچستر] استفاده می شود.

بدیهی است که ایستگاه فرستنده باید بطور مداوم بر کانال نظارت کند و منتظر شنیدن سیگنال نویزی که مشخص کننده وقوع تصادم است باقی بماند. به همین دلیل CSMA/CD با یک کانال منفرد ذاتاً سیستمی Half Duplex (دو طرفه غیرهمزمان) است. در چنین سیستمی برای ایستگاه این امکان وجود ندارد که ارسال و دریافت فریمها را بطور همزمان انجام بدهد زیرا بخش گیرنده آن حتی در خلال ارسال مشغول و در حال پیگیری بروز تصادم بر روی کانال است. [البته در حین ارسال، گیرنده فقط آماده کشف تصادم به صورت آنالوگ است و داده های روی کانال، دریافت یا ذخیره نمی شوند].

لازم است بدین نکته بدیهی اشاره کنیم که پروتکل زیرلایه MAC دریافت مطمئن فریمها را تضمین نمی کند چرا که اگر حتی تصادمی رخ ندهد گیرنده فریم ممکن است بدلائل گوناگون نتواند فریم را به درستی دریافت کند. (دلائلی مثل فقدان فضای بافر یا وقفه های گم شده)

۳-۲-۴ پروتکل های بدون تصادم

اگرچه در روش CSMA/CD پس از آنکه یک ایستگاه بطور مطمئن کانال را تصرف کرد دیگر وقوع تصادم منتفی است ولیکن وقوع تصادمهای مکرر در دوره رقابت^۱ کاملاً طبیعی است. این تصادمها تاثیر زیانباری بر روی کارآئی سیستم خواهد داشت؛ بالاخص زمانی که کابل، طولانی (و در نتیجه τ یا همان تاخیر انتشار بالاست) و یا فریمها کوتاه هستند، این تاثیر بسیار مخرب است. در چنین حالاتی روش CSMA/CD قابل استفاده نخواهد بود. در این بخش چند پروتکل را بررسی می کنیم که در آنها مسئله رقابت حل شده و بروز تصادم حتی در زمان رقابت منتفی است. اگرچه بسیاری از این روشها در حال حاضر بر روی سیستمهای شناخته شده به کار گرفته نشده است ولیکن در اختیار داشتن پروتکل هایی با ویژگیهای عالی، برای به کارگیری در سیستمهای پیشرفته آینده یک حسن محسوب می شود!

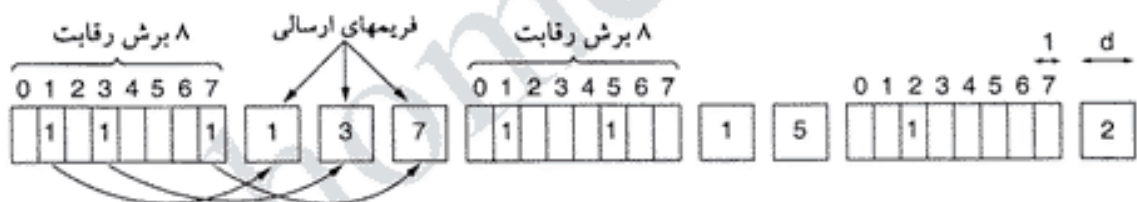
در پروتکل هایی که بررسی می شوند فرض را بر آن گذاشته ایم که دقیقاً N ایستگاه به کانال مشترک متصلند و

۱. Contention Period

هر ایستگاه آدرسی بین 0 تا $N-1$ دارد که درون سخت افزار ایستگاه حک شده است. این موضوع که ممکن است برخی از ایستگاهها در برخی از مواقع غیر فعال و خاموش باشند چندان اهمیت ندارد. همچنین فرض کرده ایم که تاخیر انتشار قابل صرف نظر باشد. مسئله بنیادی هنوز باقی است: «پس از آنکه ارسال فعلی یک ایستگاه به پایان رسید کدامین ایستگاه حق دارد کانال را در اختیار بگیرد و فریم خود را ارسال نماید؟» در این بخش نیز از مدل شکل ۴-۵ با برشهای گسسته رقابت^۱ استفاده می کنیم.

یک پروتکل مبتنی بر نشانه های بیتی (Bit Map)

در اولین پروتکل بدون تصادم که Basic Bit-Map Method نام دارد بازه رقابت متشکل از دقیقاً N برش زمانی است. هرگاه ایستگاه شماره صفر، فریمی آماده ارسال داشته باشد در برش (اسلات) شماره صفر، بیت ۱ را بر روی کانال می گذارد؛ هیچ ایستگاه دیگری حق ندارد در این برش چیزی بر روی کانال بگذارد. فارغ از آنکه ایستگاه صفر چه کاری می کند ایستگاه شماره ۱ نیز این فرصت را دارد که در برش شماره ۱ با قرار دادن بیت ۱ بر روی کانال تقاضای ارسال فریم بدهد. (البته اگر فریمی برای ارسال آماده داشته باشد). بطور کلی ایستگاه شماره j حق دارد در صورتی که فریمی جهت ارسال داشته باشد در برش شماره j با گذاشتن بیت ۱ بر روی کانال، تقاضای خود را اعلام نماید. پس از آنکه کل N برش رقابت سپری شد تمام ایستگاهها فهرست کاملی از ایستگاههای متقاضی ارسال، در اختیار دارند. پس از این لحظه، ایستگاههای متقاضی ارسال، به ترتیب شماره، ارسال فریمهای خود را آغاز می کنند. (شکل ۴-۶ را ملاحظه کنید)



شکل ۴-۶. پروتکل پایه مبتنی بر نشانه های بیتی (Basic Bitmap Protocol).

از آنجایی که تمام ایستگاهها در هر لحظه می دانند که چه ایستگاهی حق ارسال دارد به هیچ وجه تصادم رخ نخواهد داد. پس از آنکه آخرین ایستگاه، فریم خود را ارسال کرد، (رخدادی که همه ایستگاهها پسادگی می توانند متوجه آن شوند) مجدداً دوره رقابت، شامل N برش (با عبارتی N بیت) شروع می شود. هرگاه ایستگاهی پس از گذشت برش (اسلات) متعلق به او، آماده ارسال شود فرصت را از دست داده و بایستی خاموش بماند تا دیگر ایستگاهها شانس خود را آزموده و کارشان را انجام بدهند تا دور بعدی رقابت فرا برسد. پروتکلهایی همانند این روش که در آنها هر ایستگاه تقاضای خود را قبل از ارسال و به صورت فراگیر به اطلاع همه می رساند اصطلاحاً پروتکل های رزرو سازی (Reservation) نامیده می شوند.

اجازه بدهید مختصراً به کارآیی این پروتکل بپردازیم. برای سادگی، واحد زمان را معادل طول بیت هر برش رقابت در نظر گرفته ایم و طول هر فریم داده را معادل d واحد زمان فرض کرده ایم. [یعنی هر یک از برشهای رقابت را یک واحد و بر این اساس، طول فریم را d واحد زمان، تلقی کرده ایم.] در شرایطی که بار ایستگاهها پائین است، برشهای رقابت بطور متوالی تکرار می شوند و بدلیل عدم تقاضا برای ارسال فریم مکرراً خالی می مانند. حال وضعیت را از دیدگاه یک ایستگاه با شماره پائینی مثل صفر یا یک بررسی می کنیم. [بدون آنکه به عمومیت استدلال لطمه ای بخورد] وقتی چنین ایستگاهی آماده ارسال می شود باید تا رسیدن برش متناظر با شماره خودش صبر کند

۱. Discrete Contention slots

در حالیکه ممکن است شماره برش فعلی یکی از شماره های میانی باشد. بطور «میانگین» ایستگاه مجبور است به اندازه $N/2$ صبر کند تا دور فعلی خاتمه یافته و در برش متعلق به خودش تقاضا بدهد؛ سپس باید اندازه N برش صبر کند تا تمام برشها به ترتیب بگذرند و او بتواند ارسال خود را آغاز کند. [چون نوبت او گذشته بطور میانگین باید $N/2$ برش دیگر صبر کند تا زمان ارسال او فرا برسد. بنابراین بطور میانگین زمانی معادل $1.5N$ معطل می شود.] چشم انداز شانس ایستگاه های با شماره بالا روشتر است. عموماً چنین ایستگاه هایی بطور میانگین مجبورند قبل از شروع به ارسال، $N/2$ برش صبر کنند تا نوبت به اعلام تقاضا و سپس ارسال آنها برسد. ایستگاه های با شماره بالا به ندرت مجبورند که یک دور کامل صبر کنند تا دور بعدی فرا برسد. از آنجایی که ایستگاه های با شماره پائین باید بطور میانگین $1.5N$ و ایستگاه های با شماره بالا باید $0.5N$ منتظر بمانند لذا میانگین این دو زمان N خواهد شد. با این استدلال، محاسبه کارآئی کانال در بار پائین ساده است: میزان سرباری که باید برای هر فریم d بیتی متحمل شد N بیت است بنابراین این کارآئی کانال در بار پائین عبارتست از: $d/(N + d)$

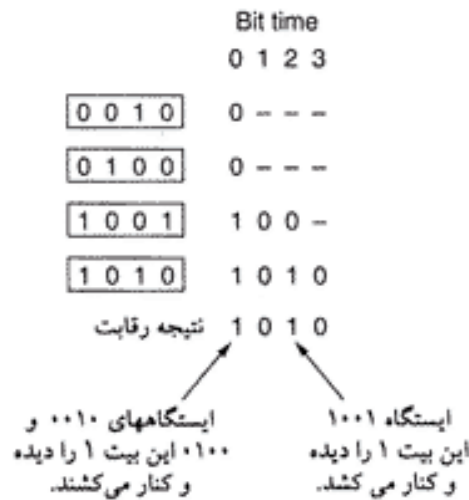
در بار بالا، یعنی وقتی تمام ایستگاه ها چیزی برای ارسال داشته باشند، N برش رقابت برای ارسال N تا فریم متوالی صرف می شود و این سربار به ازای هر فریم 1 بیت خواهد بود [یعنی N بیت سربار بر روی N فریم d بیتی سرشکن می شود لذا به ازای هر فریم d بیتی فقط یک بیت سربار تحمیل می شود.] بنابراین کارآئی کانال در بار بالا عبارتست از: $d/(d+1)$. میانگین تاخیر ارسال هر فریم نیز، معادل با زمان تاخیر در صف داخلی هر ایستگاه به اضافه مقدار $N(d+1)/2$ (معادل تاخیر ارسال پس از آنکه فریم به سر صف داخلی ایستگاه می رسد) خواهد بود.

روش شمارش دودویی معکوس (Binary Countdown)

مشکل پروتکل Bitmap آنست که به ازای هر ایستگاه یک بیت سربار تحمیل می شود فلذا شبکه قابل گسترش به مقیاس هزاران ایستگاه، نخواهد بود. می توان براساس آدرس دودویی ایستگاه ها، فرآیند رزرو سازی را به روش بهتری انجام داد: ایستگاهی که می خواهد کانال را به خدمت بگیرد شماره آدرس خود را به صورت دنباله ای از بیتها که از پرارزشترین آن شروع می شود بر روی کانال منتشر می کند. فرض بر آنست که تمام آدرسها دارای طول یکسانی هستند. هر یک بیتهای آدرس پس از انتشار بر روی کانال بطور منطقی با یکدیگر (Wired OR) OR می شوند. این روش که «پروتکل شمارش دودویی معکوس» نام دارد برای اولین بار در سیستم Datakit (Fraser, 1987) به کار گرفته شد. در این روش به صراحت فرض شده که تاخیر انتشار خط ناچیز است و تمام ایستگاه ها می توانند بطور همزمان بیتهای ارسالی را بر روی کانال را بشنوند.

برای پیشگیری از هرگونه برخورد، باید یک «قاعده دوری» (Arbitration Rule) اعمال شود: به محض آنکه ایستگاهی متوجه شود که بیت پرارزش صفر او که بر روی کانال قرار گرفته با بیت 1 بازنویسی شده از دور رقابت کنار می کشد. به عنوان مثال اگر ایستگاه های 0100 ، 0010 ، 1001 و 1010 بخواهند کانال را بدست بیاورند، همگی در اولین برش زمانی، بیت پرارزش خود را بر روی کانال می گذارند، یعنی به ترتیب بیتهای 0 ، 0 ، 1 و 1 بر روی کانال ظاهر می شود. این بیتها با یکدیگر Wired OR شده و نتیجه آن 1 خواهد بود. ایستگاه های 0100 و 0010 متوجه ظهور 1 بر روی خط شده و نتیجه می گیرند که ایستگاهی با شماره بالاتر برای تصرف کانال رقابت می کند، لذا از دور فعلی خارج می شوند. در ادامه فقط ایستگاه های 1001 و 1010 به رقابت می پردازند.

بیت بعدی هر دو صفر است فلذا این دو ایستگاه باز هم ادامه می دهند. بیت بعدی روی کانال 1 خواهد بود لذا ایستگاه 1001 کنار می کشد. در نهایت ایستگاه 1010 برنده این رقابت خواهد بود چراکه دارای بالاترین شماره آدرس است. پس از پیروزی در این رقابت (که به صورت مزایده برگزار شده)، ایستگاه برنده می تواند فریم خود را ارسال کند و پس از آن دور بعدی «مزایده» آغاز می شود. عملکرد این پروتکل در شکل $4-7$ به تصویر کشیده شده



شکل ۴-۷. پروتکل شمارش دودویی معکوس. خط تیره علامت سکوت ایستگاه است.

است. پروتکل فوق دارای این ویژگی است که ایستگاه‌های با شماره بالاتر اولویت بیشتری نسبت به ایستگاه‌های با شماره پائینتر دارند؛ این ویژگی براساس زمینه و نوع کار می‌تواند خوب یا بد باشد. کارآئی کانال در این روش معادل $d/(d + \log_2 N)$ است. با اینحال اگر قالب فریم بگونه‌ای زیرکانه انتخاب شود که آدرس فرستنده فریم، همان اولین فیلد باشد مقدار سربار $\log_2 N$ بیت نیز تلف نخواند شد و کارآئی کانال صددرصد است!

دو پژوهشگر به نامهای «مارک» و «وارد» (۱۹۷۹)، گونه‌ای از روش «شمارش دودویی معکوس» را با استفاده از واسطه‌های موازی (Parallel Interfaces) به جای واسطه‌های سریال معرفی و تشریح کردند. [یعنی ارسال اطلاعات به جای سریال به صورت موازی انجام می‌شود.] آنها پیشنهاد کردند که برای آدرس‌دهی ایستگاه‌ها، [به جای شماره‌های ثابت] از شماره‌های مجازی استفاده شود. پس از آنکه ایستگاهی موفق به ارسال شد، به شماره تمام ایستگاه‌های قبل از آن، تا شماره صفر، یک واحد اضافه می‌شود تا در مرحله بعد، آنهایی که موفق به ارسال نشده‌اند اولویت بالاتری داشته باشند. [بدین ترتیب مسئله قبضه شدن کانال توسط یک ایستگاه حل خواهد شد. -م]

بعنوان مثال فرض کنید ایستگاه‌های A, B, C, D, E, F, G, H به ترتیب دارای اولویت‌های ۱, ۲, ۳, ۴, ۵, ۶, ۷, ۸ هستند و [از بین ایستگاه‌های متقاضی مثل A, B, C, D, E, F, G, H] موفق به ارسال فریم خود شده است. پس از ارسال، D به انتهای این فهرست رفته و ترتیب اولویت‌ها به صورت A, B, C, D, E, F, G, H و D تغییر می‌کند. بدین ترتیب C به صورت مجازی ایستگاه شماره ۷ باقی می‌ماند ولی شماره A از ۴ به ۵ صعود و شماره D، از ۵ به ۶ صفر سقوط می‌کند. حال ایستگاه D فقط در صورتی قادر به در اختیار گرفتن کانال خواهد شد که هیچ ایستگاه دیگری بدان احتیاج نداشته باشد.

روش شمارش دودویی معکوس نمونه‌ای است از یک پروتکل ساده، جالب و کارآمد که در انتظار کشف مجدد، روزگار می‌گذراند و در آینده، منزلتی را برای خود خواهد یافت!!

۴-۲-۴ پروتکل‌های با رقابت محدود

تاکنون دو استراتژی بنیادی را برای دستیابی به کانال در شبکه‌های مبتنی بر کابل، بررسی کرده‌ایم: روش‌های رقابت و تصادم مثل CSMA و روش‌های بدون تصادم. هر کدام از این استراتژی‌ها را می‌توان بر اساس دو معیار مهم رده‌بندی کرد: (۱) میزان تاخیر در بار پائین (۲) کارآئی و بهره کانال در بار بالا.

در بار پایین روشهای مبتنی بر رقابت (مثل ALOHA یا CSMA) ارجحتر است چراکه تاخیر ناچیزی دارند؛ ولیکن وقتی بار افزایش می‌یابد روشهای مبتنی بر رقابت ارزش خود را از دست می‌دهند زیرا سربار تحمیل شده در اثر مبارزه بر سر بدست آوردن کانال، به شدت افزایش می‌یابد. دقیقاً عکس این موضوع در مورد پروتکل‌های بدون تصادم صادق است: در بار پائین تاخیر بالایی دارند ولی وقتی بار شبکه افزایش می‌یابد (برعکس پروتکل‌های مبتنی بر رقابت)، کارآئی کانال رو به افزایش می‌گذارد.

روشن است که اگر بتوانیم دو ویژگی ممتاز این روشها را با هم ترکیب کنیم به یک پروتکل مطلوب خواهیم رسید که در بار پائین بروش رقابت عمل می‌کند (تا تاخیر کمی داشته باشد) ولی در بار بالا از روش بدون تصادم بهره می‌گیرد (تا کارآئی کانال افزایش یابد). این پروتکلها به نام «پروتکل‌های با رقابت محدود» مشهور هستند و بررسی آنها، مطالعات ما را در مورد شبکه‌های مبتنی بر شنود سیگنال، به نتیجه نهانی می‌رساند.

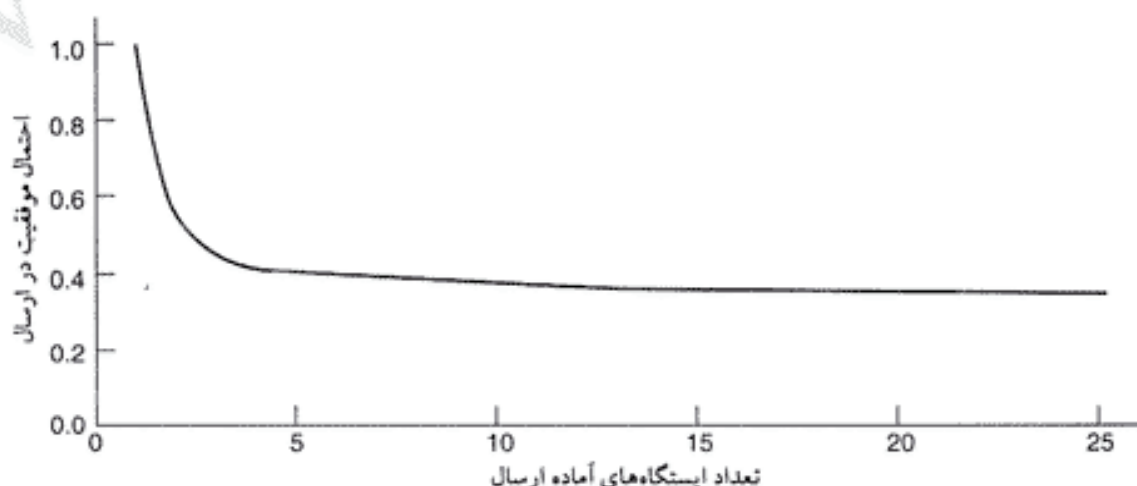
پروتکل‌های مبتنی بر رقابت که تاکنون بررسی کرده‌ایم «متقارن» (Symmetric) هستند بدین معنا که هر کدام از ایستگاه‌ها، با احتمال p تلاش می‌کند کانال را در اختیار بگیرد و برای تمام ایستگاه‌ها، این احتمال یکسان و مساوی p فرض شده است. می‌توان کارآئی کل سیستم را با انتساب احتمالات مختلف به ایستگاه‌های متفاوت، بهبود داد. قبل از آنکه به پروتکل‌های نامتقارن (Asymmetric) بپردازیم اجازه بدهید تا مروری اجمالی بر کارآئی شبکه در حالت متقارن داشته باشیم. فرض کنید تعداد k ایستگاه برای دسترسی به کانال رقابت می‌کنند و هر کدام در ابتدای یک برش زمان، با احتمال p اقدام به ارسال می‌نمایند. احتمال آنکه یکی از ایستگاه‌ها موفق به در اختیار گرفتن کانال در یکی از برشهای زمانی شود عبارتست از:

$$k.p.(1-p)^{k-1}$$

به منظور پیدا کردن مقداری بهینه برای p [به نحوی که احتمال فوق به حداکثر برسد] از رابطه فوق بر حسب p مشتق گرفته و حاصل را مساوی صفر قرار می‌دهیم؛ سپس p را محاسبه می‌نمائیم. پس از محاسبه، بهترین مقدار p معادل $1/k$ بدست می‌آید. با قرار دادن $1/k$ به جای p در رابطه فوق خواهیم داشت:

$$P_s[\text{موفقیت در ارسال با بالاترین احتمال}] = \left[\frac{k-1}{k}\right]^{k-1} \quad \text{رابطه (۴-۴)}$$

منحنی این رابطه در شکل ۴-۸ ترسیم شده است. اگر تعداد ایستگاه‌ها کم باشد، احتمال موفقیت بالاست ولی به محض آنکه تعداد ایستگاه‌ها به ۵ یا بیشتر می‌رسد، احتمال موفقیت در ارسال، به مقدار تقریباً ثابت $1/e$ ، میل می‌کند.



شکل ۴-۸ منحنی احتمال موفقیت در تصرف کانال برای یک کانال متقارن.

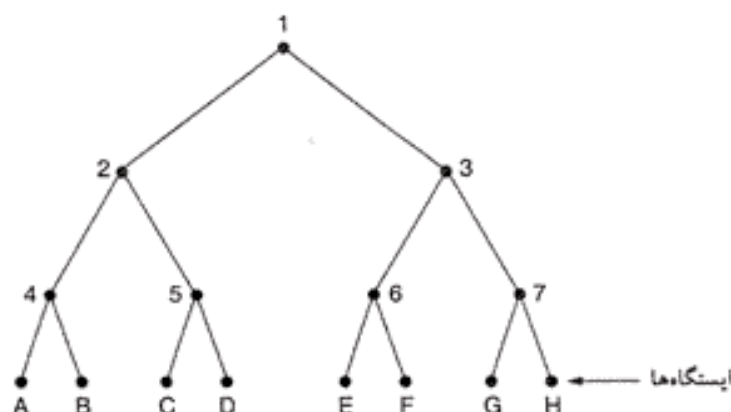
از شکل ۴-۸ بخوبی مشهود است که می توان احتمال موفقیت ایستگاه ها را با کاهش دادن حجم رقابت، افزایش داد؛ «پروتکل های با رقابت محدود» سعی می کنند دقیقاً همین کار را انجام بدهند. این پروتکلها ابتدا ایستگاه ها را به چند گروه تقسیم می نمایند. (لزومی به از هم جدا بودن گروه ها نیست.) در برش زمانی شماره صفر، فقط اعضای گروه شماره صفر حق شرکت در رقابت را دارند. اگر یکی از آنها موفق شد، کانال را در اختیار می گیرد و فریم خود را ارسال می نماید. اگر برش زمانی شماره صفر خالی ماند یا تصادم رخ داد، اعضای گروه شماره ۱ در برش شماره ۱ رقابت می کنند و کار به همین ترتیب ادامه می یابد. با تقسیم بندی صحیح و مناسب ایستگاه ها به چند گروه، میزان رقابت در هر برش زمانی کاهش یافته و طبق منحنی شکل ۴-۸، شبکه در نزدیکی سمت چپ نمودار کار می کند.

نکته ظریف در اینجا است که ایستگاه ها را چگونه به برش های زمانی مستقل منتسب نمائیم. [یعنی تعیین آنکه چه ایستگاه هایی در چه برش زمانی رقابت کنند. -م] یکی از حالات ویژه آنست که هر گروه صرفاً یک عضو بیشتر نداشته باشد. چنین انتسابی تضمین کننده آنست که هیچ تصادمی بوجود نخواهد آمد چرا که برای تصاحب هر برش زمانی فقط یک ایستگاه رقابت می کند. با چنین پروتکل هایی قبلاً برخورد کرده ایم (مثل روش شمارش دودونی معکوس). حالت خاص دیگر آنست که در هر گروه دو ایستگاه قرار بگیرد. احتمال آنکه هر دوی آنها بطور همزمان و در یک برش زمانی بخواهند ارسال داشته باشند معادل p^2 است که برای مقادیر کوچک p ، بسیار ناچیز خواهد بود. هر چه تعداد ایستگاه ها در هر گروه بیشتر باشد احتمال تصادم در برش های زمانی متعلق به آن گروه افزایش خواهد یافت ولی در عوض تعداد برش های زمانی لازم برای رقابت گروه ها کاهش می یابد. در محدودترین حالت تمام ایستگاه ها در یک گروه واحد قرار می گیرند؛ در این حالت عملکرد آن همانند روش Slotted ALOHA خواهد بود. بالطبع به روشی نیاز داریم که برش های زمانی (Time Slots) را بطور پویا به ایستگاه ها اختصاص بدهد یعنی وقتی بار کم است تعداد زیادی ایستگاه در یک برش رقابت کنند ولیکن وقتی بار بالاست تعداد کمی ایستگاه (حتی یک ایستگاه) در هر گروه رقابت نمایند.

پروتکل پیمایش واقعی درخت (Adaptive Tree Walk)

یکی از روش های بسیار ساده برای انجام عمل انتساب فوق الذکر، الگوریتمی است که توسط ارتش ایالات متحده در جنگ جهانی دوم برای آزمایش سربازان و تشخیص بیماری سیفلیس ابداع شد. (Dorfman, 1943) ارتش، نمونه خون N سرباز را می گرفت و بخشی از این نمونه های خون، درون یک لوله آزمایش واحد ریخته و با هم مخلوط می شد. سپس این نمونه مخلوط شده، مورد آزمایش قرار می گرفت و اگر هیچ آنتی بادی، در آن پیدا نمی شد تمام سربازان سالم تشخیص داده می شدند. در صورت تشخیص آلودگی در نمونه خون، دو نمونه دیگر از خونها تهیه می شد: نمونه اول شامل مخلوطی از نمونه خون سربازان ۱ تا $N/2$ و نمونه دومی از خون مابقی افراد. این فرآیند آنقدر تکرار می شد تا سربازان آلوده مشخص شوند.

برای نسخه کامپیوتری این الگوریتم (Capetanakis, 1979)، ساده تر آنست که ایستگاه ها را به عنوان برگ های یک درخت دودونی فرض کنید (همانند آنچه که در شکل ۴-۹ می بینید). در اولین برش رقابت، (یعنی پس از خاتمه ارسال یک ایستگاه و آزاد شدن کانال) تمام ایستگاه ها مجازند برای در اختیار گرفتن کانال رقابت کنند. اگر تصادمی به وقوع پیوست، در برش شماره ۱، فقط ایستگاه هایی که در پوشش گره ۲ از درخت واقعند به رقابت می پردازند. اگر یکی از آنها کانال را تصاحب و فریم خود را ارسال کرد، برش زمان بعدی، برای رقابت ایستگاه های تحت پوشش گره ۳ در نظر گرفته می شود ولیکن اگر بیش از یکی از ایستگاه های تحت پوشش گره ۲، بخواهند ارسال داشته باشند در همان برش شماره ۱ تصادم رخ می دهد و در برش شماره ۲ نوبت به رقابت ایستگاه های گره ۴ خواهد بود.



شکل ۹-۴. یک درخت برای هشت ایستگاه.

هرگاه در برش شماره تصادمی رخ بدهد، کل درخت به صورت «عمقی»^۱ (Depth First) پیمایش می شود تا ایستگاه‌ها به ترتیب شناسایی و تعیین موقعیت شده و ارسال خود را انجام بدهند. هر برش رقابت، به یک گره خاص در درخت تعلق دارد. اگر تصادمها تکرار شود، پیمایش و جستجو از چپ به راست و به صورت بازگشتی (Recursive) ادامه می یابد. اگر یکی از برشها خالی بماند یا فقط یک ایستگاه، بدون تصادم کانال را صاحب شود جستجو در آن گره خاتمه می یابد، چراکه تمام ایستگاه‌های آماده ارسال در آن گره، مشخص شده اند. (اگر بیش از یک ایستگاه در آن گره تمایل به ارسال می داشتند تصادم رخ می داد).

وقتی بار شبکه بالا است به ندرت اتفاق می افتد که مسئله تخصیص کانال در همان برش رقابت شماره صفر که متعلق به گره ۱ است حل شود زیرا به احتمال زیاد بیش از یک ایستگاه آماده ارسال هستند. بدلیل مشابه این مسئله در برش ۲ و ۳ نیز حل نخواهد شد و رقابت به مراحل بعدی خواهد کشید. سوال آنست که بطور کلی، جستجو از چه سطحی آغاز شود؟ روشن است که هر چه بار سنگینتر باشد، جستجو باید از سطوح پائین تر شروع شود. فرض را بر آن خواهیم داشت که هر ایستگاه تخمین خوبی از تعداد کل ایستگاه‌های آماده ارسال (که آنرا q می نامیم) در اختیار دارد و این تخمین را به روشی مثل نظارت بر ترافیک جاری شبکه بدست آورده است.

در ادامه اجازه بدهید سطوح درخت را شماره گذاری کرده و رأس آن یعنی گره شماره ۱ را سطح صفر بنامیم. بدین ترتیب، گره ۲ و ۳ در سطح یک قرار می گیرند و شماره گذاری سطوح بهمین نحو ادامه می یابد. دقت کنید که هرگره در سطح i ، کسر $\frac{1}{2^i}$ از کل ایستگاه‌های شبکه را در بر می گیرد. اگر q ایستگاه آماده ارسال، بطور یکنواخت در گره‌ها توزیع شده باشند، میانگین تعداد ایستگاه‌های آماده ارسال که تحت پوشش گرهی خاص در سطح i قرار دارند، معادل $2^{-i}q$ خواهد بود. بطور حسی می توان انتظار داشت که سطح بهینه ای که جستجو باید از آن سطح آغاز شود همان سطحی است، تعداد متوسط ایستگاه‌های آماده در زیر هر گره ۱ باشد، یعنی همان سطحی که در آن $2^{-i}q = 1$ است. با حل این معادله بدست می آوریم: $i = \log_2 q$.

دو پژوهشگر بنامهای Gallager و Bertsekas (1992)، نسخه های متعدد و پیشرفته تری از الگوریتم فوق را ابداع و جزئیات آنها را تشریح کرده اند. به عنوان مثال، حالتی را در نظر بگیرید که در آن ایستگاه های G و H [تحت پوشش گره ۷] تنها ایستگاه های آماده ارسال هستند. در گره ۱ تصادم رخ خواهد داد، در حالیکه وقتی در برش شماره ۲ نوبت به رقابت اعضای گره ۲ می رسد، آن برش خالی خواهد ماند. مجدداً در حین آزمایش گره ۳ تصادم پیش می آید؛ (تا اینجا می دانیم که دو یا چند ایستگاه در گره ۱ تمایل به ارسال دارند ولی قطعاً در گره ۲ واقع

۱. در مبحث ساختمان داده و الگوریتمها، پیمایش عمقی روشی برای پیمایش درخت محسوب می شود. -م

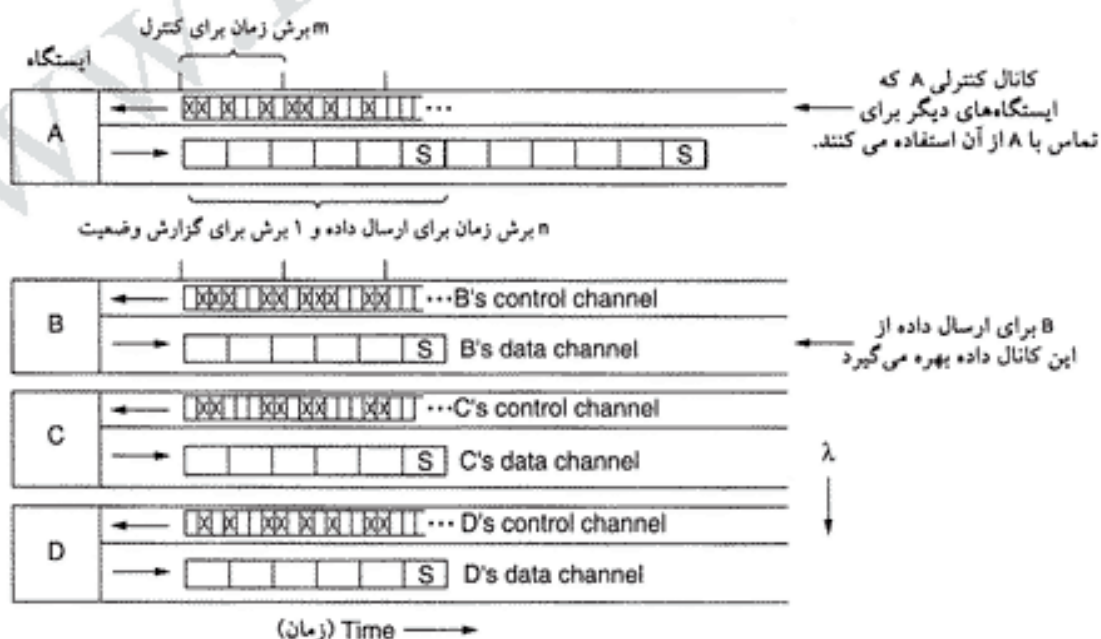
نشده‌اند). آزمایش گره ۳ ناکام مانده و به آزمون گره ۶ می‌انجامد. باز هم در آزمایش گره ۷ تصادم رخ می‌دهد و عاقبت در آخرین تلاش G موفق به ارسال خواهد شد. دو پژوهشگر فوق برای حل مشکلات این چنینی، راهکارهایی را معرفی کرده‌اند.

۵-۲-۴ پروتکل‌های دسترسی چندگانه مبتنی بر تقسیم طول موج^۱

یک راهکار متفاوت برای تخصیص کانال آنست که آنرا بروشی مثل FDM یا TDM (یا تلفیقی از هر دو) به چندین «زیرکانال» تقسیم کرده و آنها را در صورت نیاز به صورت پویا به ایستگاه‌ها تخصیص بدهیم. چنین الگویی، به طور رایج بر روی شبکه‌های محلی با فیبرنوری کاربرد دارد تا با استفاده از طول موجهای مختلف (یعنی فرکانسهای متفاوت)، امکان محاوره و ارتباط همزمان ایستگاه‌ها فراهم آید. در این بخش یکی از این پروتکلها را بررسی خواهیم کرد. (Humblot et al., 1992)

ساده‌ترین راه برای ساختن یک شبکه محلی کاملاً نوری، آنست که شبیه به شکل ۱۰-۲، از ترویج کننده غیرفعال با توپولوژی ستاره استفاده نمائیم. در حقیقت دو فیبرنوری منشعب شده از هر ایستگاه به یک استوانه شیشه‌ای وارد شده‌اند. [در این استوانه تمام پرتوهای نوری با هم ممزوج شده، در هم می‌آمیزند. -م] یکی از تارهای فیبر نوری بعنوان خروجی و دیگری بعنوان ورودی ایستگاه از این استوانه منشعب می‌شوند. پرتوی نور خارج شده از یک ایستگاه، در درون استوانه منتشر شده و در ورودی تمام ایستگاهها قابل دریافت و تشخیص است. شبکه‌های نوری غیر فعال با توپولوژی ستاره می‌توانند صدها ایستگاه داشته باشند.

برای آنکه ارسال همزمان چندین ایستگاه میسر باشد. طیف نوری به چندین کانال (باند‌های مختلف با طول موج متفاوت) تقسیم می‌گردد. (شکل ۳۱-۲ را ببینید). در این پروتکل که WDMA (دسترسی چندگانه مبتنی بر تقسیم طول موج) نامیده شده به هر ایستگاه دو کانال متناسب می‌شود: یک کانال با پهنای باند باریک که از آن به عنوان کانال کنترل، جهت هماهنگی (سیگنالینگ) با ایستگاه استفاده می‌شود و یک کانال با پهنای باند وسیع که برای ارسال فریم، در اختیار ایستگاه قرار می‌گیرد.



شکل ۱۰-۴. دسترسی چندگانه مبتنی بر تقسیم طول موج.

به نحوی که در شکل ۴-۱۰ دیده می شود هر کانال به چندین گروه برش زمانی مستقل (اسلات) تقسیم شده است. اجازه بدهید تعداد برشهای زمانی در کانال کنترل را m و تعداد برشهای زمانی کانال داده را $n+1$ فرض کنیم؛ n تا از برشهای کانال داده، برای ارسال فریم و آخرین آنها برای گزارش وضعیت خود ایستگاه، کاربرد دارد (در حقیقت، این برش برای گزارش برشهای آزاد در دو کانال به کار می آید). در هر دو کانال [داده و کنترل] دنباله برشهای زمانی بطور دائم و متناوب تکرار می شوند و از برش شماره صفر آغاز می گردد. برش شماره صفر بگونه ای نشانه گذاری شده که دیگر ایستگاه ها بسادگی قادر به تشخیص آن هستند. تمام کانالها با استفاده از یک سیگنال ساعت سراسری سنکرون می شوند.

این پروتکل از سه رده متفاوت ترافیک حمایت می کند: (۱) ترافیک اتصال گرا با نرخ ثابت برای عملیاتی نظیر ارسال تصاویر ویدیویی غیرفشرده (۲) ترافیک اتصال گرا با نرخ متغیر برای عملیاتی نظیر انتقال فایل (۳) ترافیک دیتاگرام مثل ارسال بسته های UDP.

در دو پروتکل اتصال گرا، نظریه آن بوده که وقتی A بخواهد با B ارتباط برقرار کند ابتدا یک فریم کنترلی خاص به نام «فریم تقاضای اتصال» (CONNECTION REQUEST) در یک برش خالی از کانال کنترلی متعلق به B قرار می دهد. اگر B این تقاضا را پذیرفت مبادله، از طریق کانال داده انجام می شود. هر ایستگاه دارای دو فرستنده و دو گیرنده به ترتیب زیر است:

۱. یک گیرنده با طول موج ثابت برای گوش دادن به کانال کنترلی خودش
 ۲. یک فرستنده قابل تنظیم برای ارسال بر روی کانالهای کنترلی دیگر ایستگاهها
 ۳. یک فرستنده با طول موج ثابت برای انتقال فریمهای داده بر روی خروجی خودش
 ۴. یک گیرنده با طول موج قابل تنظیم برای انتخاب یکی از فرستنده های داده و گوش دادن به آن
- به عبارت دیگر هر ایستگاه مدام به کانال کنترلی خودش گوش فرا می دهد تا تقاضاهای ارتباط را دریافت کند ولیکن برای دریافت داده های دیگران، باید خود را با طول موج فرستنده تنظیم کرده و تطبیق بدهد. تنظیم طول موج به کمک ابزاری به نام «ایتر فیر و متر فابری-پرو یا ماخ-ژندر»^۱ انجام می شود که در حقیقت نوعی فیلتر نوری است که در آن به غیر از یک باند خاص از طول موجها، بقیه حذف می شوند.
- حال ببینیم که ایستگاه A، چگونه یک کانال رده ۲ [یعنی ارسال اتصال گرا با نرخ متغیر] با B، مثلاً برای انتقال فایل ایجاد می کند. ابتدا A، گیرنده داده خود را بر روی طول موج ایستگاه B تنظیم می نماید و انتظار می کشد تا به یک «برش گزارش وضعیت»^۲ (Status Slot) برسد. این برش زمانی، مشخص می کند که کدامیک از برشها در کانال کنترلی، خالی و کدام تخصیص داده شده اند. بعنوان مثال، در شکل ۴-۱۰ می بینیم که از میان هشت برش برش کنترلی متعلق به ایستگاه B، شماره های ۰ و ۴ و ۵ آزاد و بقیه اشغال هستند. (علامت x به معنای پر بودن برش است). ایستگاه A یکی از برشهای آزاد کانال کنترل (مثلاً ۴) را انتخاب کرده و پیام «تقاضای اتصال» (CONNECTION REQUEST) خود را در آن می گذارد. از آنجایی که B بطور مدام بر کانال کنترل خود نظارت دارد این تقاضا را می بیند و برش ۴ را به A انتساب می دهد. این انتساب در «برش گزارش وضعیت» از کانال داده متعلق به ایستگاه B، اعلام می شود. وقتی A این اعلام را دریافت می کند متوجه می شود که اتصالی یک طرفه برایش مهیا است. اگر A تقاضای اتصالی دوطرفه داشته باشد، ایستگاه B نیز باید همین روال را انجام بدهد. ممکن است درست در زمانی که A سعی می کند برش شماره ۴ از کانال کنترل B را برای خود تصرف نماید، C

۱. Fabry-Perot or Mach-Zehnder Interferometer

۲. «برش گزارش وضعیت» (Status Slot)، آخرین برش در هر گروه از برشهای زمانی است.

نیز همین کار را بکند. در این حالت هیچکدام موفق نخواهند شد و با نظارت بر «برش گزارش وضعیت» متوجه این شکست می شوند. [چون پاسخی در «برش گزارش وضعیت» دریافت نمی کنند.] در این حالت هر یک به اندازه یک مقدار زمان تصادفی صبر کرده و از نو شروع می نمایند.

در این لحظه، هر یک از طرفین روشی بدون اشکال برای ارسال پیامهای کنترل کوتاه خود به طرف دیگر در اختیار دارند. برای انجام عملیات انتقال فایل، اکنون A می تواند پیغام کنترلی کوتاهی برای B مثلاً بدین مضمون بفرستد: «لطفاً داده های من را در برش داده شماره ۳ دریافت کنید؛ در این برش یک فریم داده برای شما ارسال شده است!» وقتی B این پیام کنترلی را دریافت می کند گیرنده خود را با طول موج کانال خروجی A تنظیم می کند. بسته به پروتکل لایه بالاتر، B نیز می تواند در صورت تمایل از چنین مکانیزمی برای برگرداندن پیامهای تصدیق (ACK) استفاده نماید.

دقت کنید که مسئله دیگری نیز ممکن است بوجود بیاید و آن هم اینکه اگر A و C اتصال با B داشته باشند و هر کدام بطور همزمان به B اعلام کند که به برش شماره ۳ مراجعه کند. B تقاضای یکی از آنها را بطور تصادفی انتخاب کرده و دیگری قادر به ارسال نخواهد شد.

برای ترافیک با نرخ ثابت، گونه دیگری از این پروتکل به کار گرفته می شود: وقتی A تقاضای ایجاد یک اتصال می کند، بطور لحظه ای فریمی را بدین مضمون برای B می فرستد: «آیا مجاز به ارسال دائم در برش شماره ۳ برای شما هستم؟» اگر B قادر به پذیرش چنین تقاضایی باشد (یعنی هیچ قرار قبلی برای این برش نگذاشته باشد)، یک اتصال با پهنای باند تضمین شده، ایجاد خواهد شد. در غیر این صورت A می تواند پیشنهاد دیگری را براساس خالی بودن برشهای دیگر ارائه بدهد. برای ترافیک رده سوّم (یعنی ترافیک دیتاگرام) نیز از گونه متفاوتی استفاده می شود: به جای نوشتن پیغام «تقاضای اتصال» در یک برش کنترلی، پیام DATA FOR YOU IN SLOT 3 را بر روی کانال کنترل قرار می دهد (به مضمون آنکه در برش ۳ داده ای بر شما وجود دارد). اگر B در خلال برش شماره ۳ از کانال داده آزاد باشد، انتقال انجام می شود و در غیر این صورت فریم از دست خواهد رفت. بدین ترتیب نیاز به برقراری هیچ اتصالی نخواهد بود.

گونه های متفاوتی از این پروتکل قابل اجرا است. مثلاً به جای آنکه هر ایستگاه بطور مستقل برای خودش کانال کنترل داشته باشد، یک کانال کنترلی واحد بین همه ایستگاه ها مشترک باشد. به هر ایستگاه نیز مجموعه ای از برشهای هر گروه متسبب می شود تا بتوان چندین کانال مجازی را بر روی یک کانال فیزیکی واحد مالتی پلکس کرد. همچنین این امکان وجود دارد که فقط از یک فرستنده قابل تنظیم و یک گیرنده قابل تنظیم در هر ایستگاه استفاده شود و کانال واحد هر ایستگاه به m برش کنترلی و بدنبال آن n+1، برش داده تقسیم گردد. اشکال این روش آنست که ایستگاه های فرستنده مجبورند برای در اختیار گرفتن کانال، زمان بیشتری را منتظر بمانند و از طرفی دنباله فریمهای داده با فاصله بیشتری از هم ارسال می شوند چراکه فریم کنترلی مابین آنها قرار گرفته است.

تاکنون پروتکل های WDM بی شماری پیشنهاد و پیاده سازی شده که در بسیاری از جزئیات با هم متفاوت هستند. برخی از آنها دارای یک کانال کنترل واحد هستند در حالیکه برخی دیگر چندین کانال کنترلی دارند. بعضی از آنها تاخیر انتشار را به حساب آورده اند در حالیکه بعضی از آن چشمپوشی کرده اند. برخی از آنها زمان تنظیم (فیلتر گیرنده یا فرستنده) را به عنوان بخشی مشخص از مدل خود در نظر گرفته اند در حالیکه برخی از آن صرف نظر کرده اند. این پروتکلها از دیدگاه پیچیدگی پردازش، ظرفیت و بهره مفید و قابلیت گسترش نیز با یکدیگر متفاوتند. به سیستمی که از تعداد زیادی فرکانس استفاده می کند، اصطلاحاً DWDM (Dense Wavelength Division Multiplexing) اطلاق می شود. برای آگاهی بیشتر به مراجع ذیل مراجعه کنید:

(Bogineni et al., 1993; Chen, 1994; Goralski, 2001; Kartalopoulos, 1999; Levine & Akyidiz, 1995)

۶-۲-۴ پروتکل های بی سیم برای شبکه محلی

به موازات رشد تعداد دستگاه های محاسباتی و مخابراتی قابل حمل و نقل (همراه)، تقاضا برای اتصال آنها به دنیای خارج نیز افزایش یافته است. اگرچه حتی نخستین سری تلفن های همراه امکان اتصال به تلفن دیگر را داشتند ولیکن کامپیوتر های کیفی، فاقد چنین قابلیتی بودند. چیزی نگذشت که مودمها به عنوان یک ابزار معمولی بر روی کامپیوتر های همراه جا خوش کردند ولیکن برای برقراری ارتباط، می بایست سیم این کامپیوترها را به پریزهای دیواری تلفن وصل کرد. نیاز به سیم جهت اتصال به یک شبکه ثابت، بدین معناست که کامپیوترها قابل حمل و نقل هستند ولی «همراه» (Mobile) تلفی نمی شوند.

برای تحقق معنای واقعی «همراه بودن»، کامپیوتر های کیفی نیاز به ارتباط از طریق سیگنال های رادیویی (یا مادون قرمز) داشتند. در چنین شرایطی، کاربران می توانند در حین گردش یا قایقرانی، نامه های خود را بخوانند یا نامه بفرستند. به گونه ای که در بخش ۱-۵-۴ اشاره شد، کامپیوتر های کیفی که از طریق سیگنال های رادیویی با یکدیگر مخابرات اطلاعات می کنند، شبکه محلی بی سیم تلفی می شوند. اینگونه شبکه های محلی تفاوت های عمده ای با شبکه های محلی رایج دارند و به پروتکل های خاصی در زیرلایه MAC نیازمندند. در این بخش برخی از این پروتکلها را بررسی خواهیم کرد. برای آگاهی بیشتر در مورد شبکه های محلی بی سیم مراجع (Geier, 2002; O'Hara and Petrick, 1999) مفیدند.

پیکربندی رایج برای شبکه های محلی بی سیم بدین نحو است که در یک ساختمان اداری تعدادی ایستگاه ثابت (که نقاط دسترسی -Access Point- نامیده می شوند) نصب می شود. تمام این ایستگاه های ثابت از طریق فیبر نوری یا سیم مسی به یکدیگر متصل هستند. اگر توان انتقال رادیویی ایستگاه های ثابت و کامپیوتر های کیفی بگونه ای تنظیم شود که محدوده ای حدود ۳ تا ۴ متر را پوشش بدهد هر اتاق در ساختمان نقش یک «سلول» را ایفاء خواهد کرد و کل ساختمان همانند سیستم های معمولی تلفن سلولی (یعنی شبکه تلفن همراه که در فصل ۲ بدان پرداختیم) عمل می کند ولی برخلاف سیستم تلفن سلولی، هر سلول تنها دارای یک کانال واحد است که کل پهنای باند موجود را در برگرفته و تمام ایستگاه های درون آن سلول را پوشش خواهد داد. عموماً پهنای باند این شبکه بین ۱۱ تا ۴۵ مگاهرتز بر ثانیه است.

در توضیحات زیر برای ساده تر شدن بحث، فرض را بر آن گذاشته ایم که تمام فرستنده های رادیویی، دارای برد ثابت و محدودی هستند. وقتی گیرنده ای در برد دو ایستگاه فعال و در حال ارسال قرار بگیرد سیگنال دریافتی او، مصدوم و بلااستفاده خواهد بود؛ درک این حقیقت که در اغلب شبکه های محلی بی سیم، تمام ایستگاه ها الزاماً در برد یکدیگر قرار ندارند، بسیار مهم است و منجر به پیچیدگی هائی خواهد شد. به علاوه هرگاه شبکه محلی بی سیم در داخل ساختمان قرار گرفته باشد وجود دیوار بین ایستگاه ها می تواند تاثیر مخربی بر روی برد هر ایستگاه داشته باشد.

یک روش ناشیانه برای تخصیص کانال در شبکه محلی بی سیم، بکارگیری روش CSMA است: ایستگاه ها به کانال گوش بدهند و در صورتی که هیچ ایستگاه دیگری در حال ارسال نبود، انتقال فریم انجام شود. مشکلی که در این پروتکل وجود دارد [در حالیکه در شبکه های مبتنی بر کابل وجود ندارد] آنست که در محیط بی سیم، تداخل امواج و تصادم فقط در گیرنده اهمیت دارد نه در فرستنده! برای آنکه به ماهیت این مشکل پی ببریم به شکل ۴-۱۱ که در آن چهار ایستگاه بی سیم ترسیم شده نگاهی بیندازید. اینکه کدام ایستگاه ثابت و کدامیک کامپیوتر کیفی [همراه] هستند، برای هدفی که در پیش داریم اهمیتی ندارد. برد رادیویی ایستگاه ها به گونه ای است که A و B در



شکل ۴-۱۱. یک شبکه محلی بی سیم (الف) A در حال ارسال (ب) B در حال ارسال.

برد یکدیگر هستند و سیگنال آنها می توانند با یکدیگر تداخل کرده، خراب شود. C نیز می تواند با B و D تداخل سیگنال داشته باشد ولی با A تداخل ندارد [چون در برد او نیست]. ابتدا فرض کنید که A در حال ارسال برای B، است: (شکل ۴-۱۱-الف) اگر C کانال را شنود کند هیچ چیزی از A نمی شنود چون در برد A نیست و بدین ترتیب به اشتباه نتیجه می گیرد که می تواند برای B ارسال داشته باشد. اگر C ارسال خود را آغاز نماید در ایستگاه B [با سیگنال ارسالی از A] تداخل کرده و فریم رسیده از A را نابود خواهد کرد. این مشکل که یک ایستگاه قادر نیست حضور یک رقیب را بر روی کانال تشخیص بدهد، «مشکل ایستگاه پنهان» (Hidden Station Problem) نامیده می شود و از آنجا ناشی می شود که ایستگاه ها از هم دور بوده و سیگنال های یکدیگر را نمی شنوند.

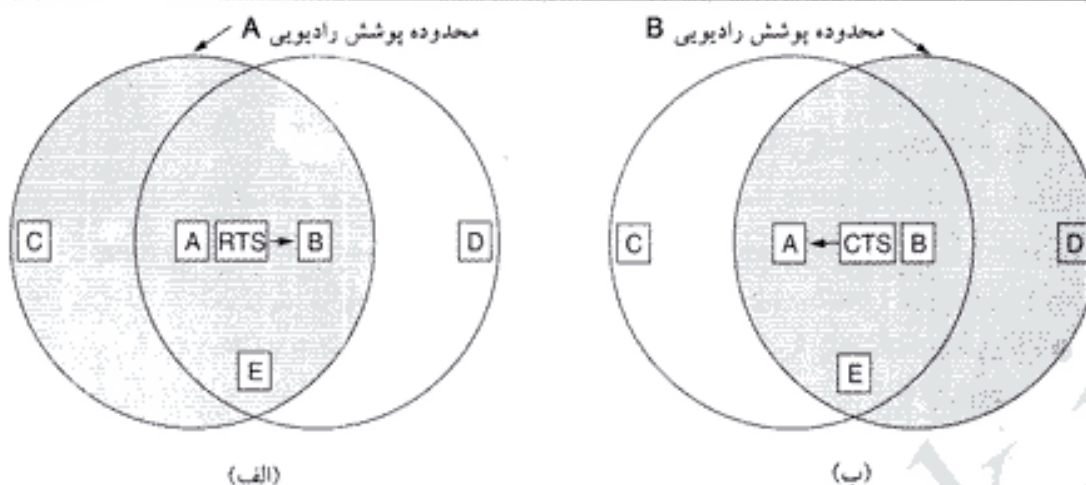
اکنون حالت برعکس شرایط فوق را بررسی می کنیم: به نحوی که در شکل ۴-۱۱-ب می بینید، B در حال ارسال برای A است. اگر C به شنود کانال بپردازد متوجه می شود که کانال اشغال است و به غلط نتیجه می گیرد که نباید برای D ارسال داشته باشد در حالیکه ارسال برای D هیچ اشکالی ندارد چرا که A و D در ناحیه ای دور از یکدیگر قرار گرفته اند و تصادمی پدید نخواهد آمد. این مشکل به نام «مشکل ایستگاه آشکار» (Exposed Station Problem) نامیده می شود.

مسئله اصلی آنست که یک دستگاه قبل از شروع به ارسال می خواهد بداند که آیا در پیرامون گیرنده، فعالیتی وجود دارد یا خیر؟ روش CSMA صرفاً می تواند در مورد فعالیت پیرامون فرستنده تشخیص خود را ارائه بدهد. به خاطر داشته باشید که بر روی سیم، سیگنالها به تمامی ایستگاه ها منتشر می شوند و طبقاً در آن واحد، فقط یک ایستگاه می تواند ارسال داشته باشد. در سیستمی که مبتنی بر امواج رادیویی یا برد کوتاه است چندین ایستگاه می توانند بطور همزمان ارسال داشته باشند البته مشروط به آنکه ایستگاه های مقصد، متفاوت و در محدوده برد یکدیگر نباشند.

روش دیگری جهت اندیشیدن به این مشکل، تجسم اداره ای است که کارمندان آن کامپیوترهای کیفی به همراه کارت شبکه بی سیم در اختیار دارند. فرض کنید کارمندی مثل لیندا می خواهد پیامی را برای میلتن بفرستد. کامپیوتر لیندا محیط پیرامون خود را شنود کرده و چون فعالیتی را تشخیص نمی دهد ارسال را شروع می کند. غافل از آنکه در محل دفتر کار میلتن، تصادم رخ خواهد داد زیرا شخص ثالثی هم اکنون در حال ارسال برای اوست و از آنجا که دفتر کار آن شخص از لیندا دور بوده، فعالیت او تشخیص داده نشده است.

MACAW, MACA

یکی از اولین پروتکل های طراحی شده برای شبکه های محلی بی سیم، پروتکل MACA (پروتکل دسترسی چندگانه با اجتناب از تصادم) است. (Karn, 1990) ایده اصلی در این روش آنست که فرستنده به نحوی گیرنده را تحریک به ارسال یک فریم کوتاه برای ایستگاه های پیرامون خود کند تا آنها بی که در برد او هستند و این فریم کوتاه را می شنوند، از ارسال اطلاعات در خلال زمان دریافت فریم، خودداری کنند. روش MACA در شکل ۴-۱۲ نشان داده شده است.



شکل ۱۲-۴. پروتکل MACA. (الف) در حال ارسال یک فریم RTS به B. (الف) در حال ارسال فریم پاسخ CTS به A.

حال بررسی کنیم که چگونه A فریمی را برای B می فرستد. ایستگاه A کارش را با ارسال یک فریم کوتاه به نام RTS^۱ برای B شروع می کند. (شکل ۱۲-۴-الف) درون این فریم کوتاه ۳۰ بیتی، طول کل فریم داده ای که قرار است در آینده ارسال گردد مشخص شده است. B در پاسخ، فریمی به نام CTS^۲ ارسال می نماید. فریم CTS نیز طول فریم داده ای را که قرار است ارسال شود، مشخص می کند؛ (شکل ۱۲-۴-ب). پس از دریافت فریم CTS، ایستگاه A می تواند ارسال خود را آغاز نماید.

حال ببینیم ایستگاه هایی که این فریمها را می شنوند چه عکس العملی نشان می دهند. هر ایستگاهی که RTS را می شنود به A نزدیک است و باید به اندازه ای صبر کند تا پیام CTS، بدون تداخل و تصادم باز گردد. هر ایستگاه که CTS را می شنود، نزدیک به B (گیرنده فریم داده) است و باید در خلال ارسال فریم کامل توسط A، ساکت بماند. از آنجا که طول فریم داده ای که در آینده قرار است ارسال شود، در فریم CTS مشخص شده، ایستگاه های شنونده CTS نیز می توانند زمان سکوت و انتظار، را تخمین بزنند.

در شکل ۱۲-۴ ایستگاه C در برد A است ولی در برد B نیست لذا RTS منتشره از A را می شنود ولی CTS ارسالی از B را نمی شنود. این ایستگاه مجاز است پس از آنکه به اندازه زمان ارسال فریم CTS منتظر ماند، همزمان با ارسال فریم داده A، او هم به ارسال خود مشغول باشد. در سمت مقابل، D در برد B هست ولی در برد A قرار ندارد، فلذا RTS را نمی شنود در حالیکه CTS را می شنود. شنیدن CTS بدین معناست که ایستگاه در نزدیکی ایستگاه گیرنده واقع شده و ایستگاه های شنونده CTS باید ارسال خود را آنقدر به تعویق بیندازند تا ارسال فریم مورد انتظار به پایان برسد. ایستگاه E هر دو فریم کنترل RTS و CTS را می شنود و همانند D بایستی منتظر بماند تا ارسال فریم خاتمه یابد.

با تمام این تمهیدات، باز هم وقوع تصادم محتمل است. به عنوان مثال اگر B و C هر دو بطور همزمان فریم RTS برای A بفرستند، تصادم رخ می دهد و فریم از بین خواهد رفت. در صورت بروز تصادم، فرستنده ناموفق (یعنی فرستنده ای که به مدت زمان مشخصی پس از ارسال RTS، فریم CTS دریافت نکند) به اندازه یک زمان تصادفی صبر کرده و از نو تلاش می کند. الگوریتم انتظار و تکرار مجدد به نام «الگوریتم عقبگرد نمایی» شهرت دارد و در مطالعه شبکه اتترنت آنرا تشریح خواهیم کرد.

به منظور افزایش کارایی پروتکل MACA، پژوهشگری به نام Bharghavan و گروه همکار او (۱۹۹۴) با

۲. مجوز ارسال یا Clear To Send

۱. تقاضا برای ارسال یا Request to Send

بررسی نتایج مطالعات شبیه سازی، اصلاحاتی را پیشنهاد کرده و پروتکل جدید را MACAW^۱ نامیدند. در بدو کار آنها متوجه شدند که اگر در لایه پیوند داده، پس از دریافت یک فریم سالم، پیام اعلام وصول (ACK) آن ارسال نگردد، طبعاً فریمهای از بین رفته، ارسال مجدد نخواهد شد تا آنکه مدتی بعد و در لایه انتقال، عدم وجود آنها کشف شود؛ این زمان انتظار بسیار طولانی و وقتگیر است و کارایی پروتکل را بشدت کاهش خواهد داد. آنها این مشکل را با معرفی یک فریم جدید ACK که پس از دریافت موفق یک فریم داده ارسال می شود، حل کردند. همچنین آنها متوجه شدند که برخی از ویژگیهای CSMA باز هم در این شبکه قابل استفاده است؛ مثلاً برای آنکه یک ایستگاه، همزمان با ایستگاه دیگری برای یک مقصد مشابه، فریم RTS نفرستد، بهتر است قبل از ارسال RTS کانال شنود شود؛ بدین ترتیب قابلیت شنود سیگنال حامل نیز به پروتکل اضافه شد. به علاوه آنها تصمیم گرفتند که الگوریتم «عقبگرد نمایی» را به جای آنکه برای هر ایستگاه به کار ببرند بطور مستقل برای یک استریم داده (که بین یک زوج مبدا و مقصد جریان دارد) اعمال نمایند. در آخر آنها به این پروتکل مکانیزمهایی افزودند تا ایستگاهها با یکدیگر اطلاعاتی در خصوص ازدحام (Congestion) رد و بدل کنند. همچنین الگوریتم «عقبگرد نمایی» را بگونه ای تغییر دادند تا در خصوص مشکلات موقت، عکس العمل شدید از خودش نشان ندهد و بدین ترتیب کارایی سیستم بهبود یابد.

۳-۴ اترنت

تا اینجا پروتکل های تخصیص کانال را بصورت کلی و اجمالی مطالعه کردیم؛ اکنون زمان آن فرا رسیده تا این قواعد را بر روی سیستم های واقعی و خصوصاً شبکه های LAN، اعمال نماییم. همانگونه که در بخش ۳.۵.۱ اشاره شد، IEEE تعدادی از شبکه های محلی و بین شهری را با نام IEEE 802 استاندارد سازی کرده است. در فهرست شکل ۳۸-۱ دیدیم که برخی از این شبکه ها هنوز وجود دارند و برخی دیگر به تاریخ پیوسته اند. شاید آنهایی که به «نظریه تناسخ روح» معتقدند بتوانند اینگونه بیندیشند که آقای چارلز داروین برگشته و به عنوان یکی از اعضای کمیته IEEE در حال حذف، پالایش و تکمیل زنجیره استانداردهاست!!! از بین استانداردهای باقیمانده از گذشته، می توان به 802.3 (اترنت) و 802.11 (شبکه محلی بی سیم) اشاره کرد. هنوز زود است که در خصوص 802.15 (Blue tooth) و 802.16 (شبکه بین شهری بی سیم) قضاوت کنیم؛ در این مورد می توانید به ویرایش پنجم این کتاب مراجعه کنید! لایه فیزیکی و زیرلایه MAC در شبکه های 802.3 و 802.11 تفاوت بنیانی دارند ولی این تفاوتها در زیرلایه منطقی (تعریف شده در 802.2 LLC)، همگرا خواهد شد و بدین ترتیب هر دوی آنها، واسطه (Interface) مشابهی با لایه شبکه دارند. [LLC فوقانی ترین زیرلایه از لایه پیوند داده هاست و وظیفه دارد تفاوتها و ناهمگونیهای اجتناب ناپذیر زیرلایه های پایین را از دید لایه شبکه مخفی نگاه دارد تا سرویسهایی که به این لایه ارائه می شود استاندارد و یکسان باشد. -م]

در خصوص اترنت در بخش ۱.۵.۳ توضیحاتی داده شده است؛ لذا آن مفاد را در اینجا تکرار نخواهیم کرد. در عوض، بر روی جزئیات تخصصی اترنت، پروتکل های مرتبط با آن و پیشرفتهای اخیر در زمینه اترنت سرعت بالا (گیگابیت اترنت) متمرکز خواهیم شد. از آنجایی که «اترنت» و IEEE 802.3 به غیر از دو تفاوت جزئی که به آنها اشاره خواهیم کرد، از بقیه جهات یکسان هستند، بسیاری از افراد این دو شبکه را معادل هم در نظر گرفته و ما نیز این دو را یکی فرض خواهیم کرد. برای اطلاعات بیشتر در خصوص اترنت از مراجع زیر استفاده کنید:

(Breyer and Riley, 1999; Seifert, 1998; and Spurgeon 2000)

۱. Multiple Access with Collision Avoidance for Wireless networks

۱-۳-۴ کابل کشی اترنت

از آنجایی که نام اترنت در اصل برگرفته از واژه «اتر» است که به کابل اشاره دارد ما نیز توضیحات خود را از کابل شروع خواهیم کرد. در این شبکه چهار نوع کابل که فهرست آنها در شکل ۴-۱۳ آمده، رایج هستند:

مزایا	تعداد سیم در هر قطعه	حداکثر طول قطعه	نوع کابل	نام کابل
کابل اصلی و اولیه (از رده خارج)	100	500 m	Thick coax	10Base5
به هاب نیازی نیست	30	185 m	Thin coax	10Base2
ارزان ترین سیستم	1024	100 m	Twisted pair	10Base-T
بهترین انتخاب برای مابین ساختمانها	1024	2000 m	Fiber optics	10Base-F

شکل ۴-۱۳. رایجترین انواع کابل کشی اترنت.

از منظر تاریخ، نخستین نوع کابل در اترنت، 10Base5 بود که عموماً به «اترنت ضخیم» شهرت داشت. این کابل به شیلنگهای زرد رنگ باغبانی شبیه است و هر ۲/۵ متر بر روی آن علامتی گذاشته شده تا محل انشعاب تزریقی مشخص باشد. (البته طبق استاندارد 802.3، زرد بودن کابل الزامی نیست ولی پیشنهاد شده است.) اتصال به کابل عموماً از طریق «انشعاب تزریقی» (Vampire Tap) انجام می شود که در آن یک سوزن به دقت در مرکز کابل کوآکسیال فرو می رود. نماد 10Base5 بدین معنی است که شبکه با نرخ 10Mbps کار می کند، از سیگنالینگ باند پایه (Baseband) بهره گرفته و طول حداکثر یک قطعه کابل ۵۰۰ متر است. گونه دیگری کابل با نام 10Broad36، در باند وسیع (Broadband) طراحی شد ولی هیچگاه به بازار نیامد و عملاً از صحنه محو گردید. در صورتی که کانال از نوع کابل کوآکس باشد، عددی که بعد از کلمه Base ظاهر می شود، حداکثر طول کابل را بر مبنای ۱۰۰ متر مشخص می کند.

نوع دوم کابل کشی اترنت 10Base2 یا «اترنت نازک» نام داشت که برخلاف کابل قبلی (که شبیه به شیلنگ باغبانی و غیر قابل انعطاف بود) به راحتی خم می شد. برای اتصال به این نوع کابل، به جای استفاده از انشعاب تزریقی، می توان از کانکتورهای BNC معمولی و ایجاد یک اتصال بشکل T، بهره گرفت. کانکتورهای BNC بسیار قابل اعتماد و کاربرد آنها ساده تر است. کابل های اترنت نازک نیز، بسیار ارزان و نصب آن راحت است ولیکن حداکثر طول کانال به ۱۸۵ متر کاهش یافته و به هر قطعه کابل حداکثر می توان ۳۰ ایستگاه متصل کرد.

تشخیص طول بیش از اندازه کابل، انشعابات بد، شل شدن اتصالات یا هرگونه پارگی در جانی از آن، از اساسی ترین مشکلات این دو نوع کابل محسوب می شوند [زیرا بروز یکی از این اشکالات کل شبکه از کار خواهد انداخت]. به همین دلیل تکنیکی برای تشخیص این معایب ابداع شده است: یک پالس الکتریکی با شکل معمولی به درون کابل تزریق می شود. اگر این پالس به یک مانع [پارگی] یا به انتهای کابل برخورد کند، یک سیگنال بازتاب (Echo) تولید شده و باز خواهد گشت. با اندازه گیری دقیق زمان بین ارسال پالس و زمان دریافت بازتاب آن، پیدا کردن محل تقریبی عامل بازتاب [محل خرابی] کشف خواهد شد. به این روش اصطلاحاً «بازتاب سنجی در حوزه زمان» (Time Domain Reflectometry) گفته می شود.

مشکلات تشخیص محل پارگی در کابل، باعث شد که الگوی متفاوتی در سیم کشی این نوع شبکه به کار گرفته شود؛ در روش جدید هر ایستگاه یک کابل اختصاصی دارد که آنرا به یک هاب مرکزی متصل می کند. این هاب اتصال الکتریکی تمام ایستگاه ها را از درون، برقرار می سازد. معمولاً این سیم ها، از نوع زوج سیم های معمولی خطوط تلفن هستند زیرا در ساختمان های اداری این سیم کشی از قبل وجود دارد و تعداد زیادی از این زوج سیم ها بلااستفاده رها شده اند. به این روش سیم کشی اصطلاحاً 10BaseT گفته می شود. هابها ترافیک داده های ورودی را

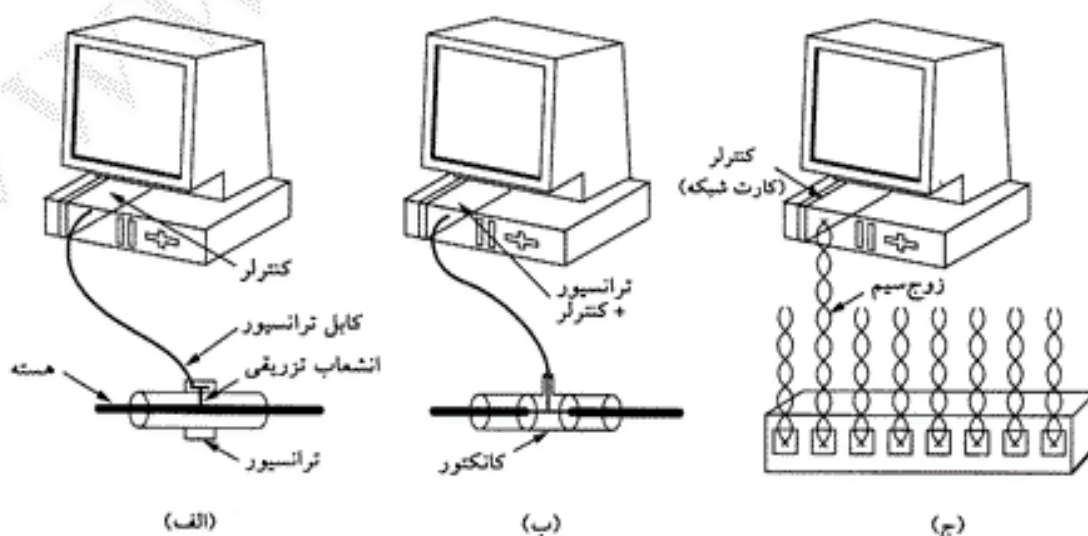
بافر نمی کنند [بلکه فقط ایستگاهها را به کانال مشترک متصل می نمایند]. در ادامه همین فصل نسخه پیشرفته تر این ایده (یعنی سونیچها) را تشریح خواهیم کرد که قادرند ترافیک ورودی را بافر کنند.

این سه روش سیم کشی در شکل ۴-۱۴ به تصویر کشیده شده است. در سیم کشی 10Base5، مدار ترانسیور^۱، باید بدقت و با اطمینان به دور کابل اصلی، محکم شود تا اتصال آن با هسته مرکزی کابل برقرار گردد. مدار ترانسیور دارای مدار الکترونیکی کوچکی است که حضور سیگنال حامل و بروز تصادم را تشخیص می دهد و در صورتی که متوجه بروز تصادم شود یک سیگنال خاص و غیر معتبر بر روی کابل می گذارد تا مطمئن شود بقیه نیز از تصادم مطلع شده اند.

در 10Base5، یک کابل بنام ترانسیور (کابل اتصال) ارتباط کارت واسط ایستگاه و کابل اصلی را برقرار می کند. طول این قطعه کابل، می تواند تا ۵۰ متر باشد و درون آن ۵ جفت سیم زره دار (Shielded) وجود دارد. دو جفت از آنها برای داده های ورودی و خروجی از ایستگاه است. دو جفت نیز برای سیگنالهای کنترلی ورودی/خروجی به کار گرفته می شوند. زوج پنجم که ممکن است از آن استفاده نشود برای تغذیه مدار الکترونیکی ترانسیور کاربرد دارد. همچنین، بعضی از ترانسیورها اجازه می دهند تا حداکثر هشت کامپیوتر نزدیک به هم، بدانها متصل شود تا تعداد ترانسیورهای مورد نیاز کاهش یابد.

کابل ترانسیور از یک طرف به کارت شبکه در درون کامپیوتر متصل است؛ کارت شبکه شامل یک تراشه کنترلر است که ارسال یا دریافت فریم به ترانسیور را بر عهده دارد. این کنترلر وظیفه دارد داده ها را در قالب یک فریم مناسب سازماندهی کند. همچنین محاسبه کدهای کشف خطا برای فریمهای خروجی و ارزیابی صحت فریمهای ورودی بر عهده همین کنترلر است. برخی از این تراشه ها دارای مقداری بافر هستند تا فریم های ورودی را جهت ارسال به صف نمایند؛ همچنین قادرند داده ها را بروش DMA به حافظه اصلی کامپیوتر منتقل کنند و در ضمن برخی از عملیات مدیریت شبکه را نیز انجام می دهند.

در سیم کشی 10Base2، اتصال کامپیوتر با کابل یکمک یک کانکتور BNC معمولی (از نوع T-شکل) انجام می شود. در این نوع سیم کشی، بخش الکترونیکی ترانسیور بر روی بُرد کنترلر قرار گرفته است و بدین ترتیب هر ایستگاه ترانسیور خود را دارد.



شکل ۴-۱۴. سه روش کابل کشی اترنت (الف) 10Base5 (ب) 10Base2 (ج) 10Base-T.

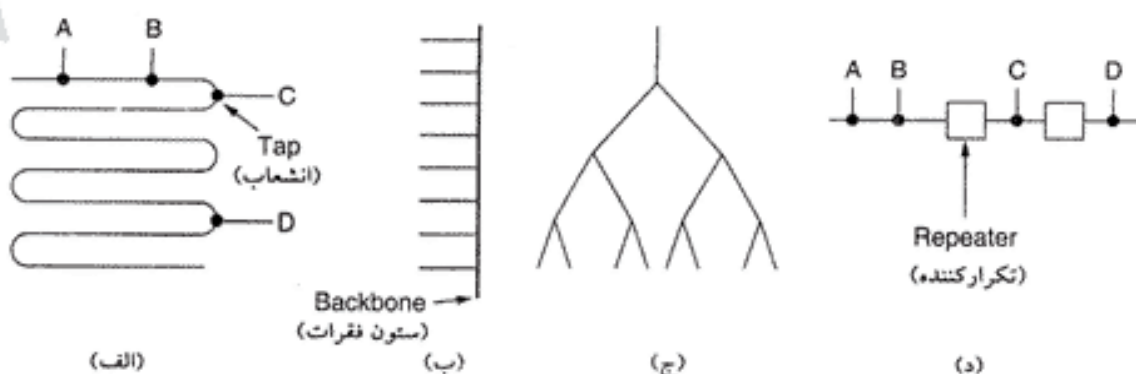
۱. ترانسیور (Tranciever) قطعه ای الکترونیکی است که اتصال کابل نازک متصل به ایستگاه و کابل ضخیم اصلی را برقرار می کند. -م

در ساختار 10Base-T، هیچ کابلی مشترکی وجود ندارد بلکه فقط یک هاب (جعبه ای پر از مدارات الکترونیک) وجود دارد که تمام ایستگاه ها با یک کابل اختصاصی (غیر مشترک) بدان متصل می شوند. حذف یا اضافه یک ایستگاه بدین ساختار بسیار ساده و هرگونه پارگی در کانال به راحتی قابل کشف است. اشکال ساختار 10Base-T آنست که طول کابل متصل به هاب، حداکثر می تواند صد متر و در صورت استفاده از کابل با کیفیت و گرانی مثل Cat 5، حداکثر دویست متر باشد. علیرغم این محدودیت، ساختار 10Base-T بدلیل سادگی نصب و پشتیبانی و همچنین استفاده از سیم کشی موجود ساختمانها، بسیار رایج شده است. نسخه سریعت 10Base-T یعنی 100Base-T در ادامه همین فصل تشریح خواهد شد.

روش چهارم کابل کشی، 10Base-F است که در آن از فیبرنوری بهره گرفته شده است. این گزینه بدلیل هزینه بالای اتصالات و پایانه های^۱ مورد نیاز، بسیار گران است ولی در عوض ایمنی بسیار بالایی در مقابل نویز دارد و انتخاب مناسبی برای کابل کشی بین ساختمانها یا هابهای دور از هم، به شمار می رود. در این روش، رشته هایی با طول کیلومتر نیز مجاز هستند. همچنین در این روش ضریب امنیت اطلاعات بسیار بالاست چراکه انشعاب گرفتن از این کانال [و استراق سمع داده ها] بسیار دشوارتر سیم های مسی است.

شکل ۴-۱۵ روشهای مختلف کابل کشی ساختمان را نشان می دهد. در شکل ۴-۱۵-الف یک کابل واحد، اطاق به اطاق کشیده شده و هر ایستگاه در نزدیکترین نقطه بدان متصل شده است. در شکل ۴-۱۵-ب یک رشته عمودی در نقش ستون فقرات از طبقه همکف تا بام کشیده شده و در هر طبقه کابل های افقی از طریق تقویت کننده های خاص (بنام تکرارکننده) به این ستون فقرات متصل شده اند. در برخی از ساختمانها کابل های افقی، نازک و کابل ستون فقرات، از نوع ضخیم انتخاب می شود. رایجترین ساختار، توپولوژی درختی است که در شکل ۴-۱۵-ج دیده می شود.

برای هر یک از روشهای کابل کشی اترنت، طول هر قطعه کابل نباید از یک مقدار حداکثر تجاوز کند. برای شبکه های وسیع می توان شبیه به شکل ۴-۱۵-د از چندین قطعه کابل که توسط «تکرارکننده» بهم متصل شده اند، استفاده کرد. تکرارکننده یک ابزار در لایه فیزیکی است داده ها را دریافت، تقویت (بازتولید) و مجدداً ارسال می کند. از دیدگاه نرم افزار، مجموعه ای از قطعات کابل که از طریق تکرارکننده به هم متصل شده اند هیچ تفاوتی با یک شبکه با کابل یکپارچه ندارد (مگر اندکی تاخیر اضافی که توسط تکرارکننده ها تحمیل می شود). یک سیستم ممکن است از چندین قطعه کابل و چند تکرارکننده تشکیل شده باشد ولیکن فاصله بین هر دو ترانسپور نبایستی از ۲/۵ کیلومتر تجاوز کند و در مسیر بین هر دو ترانسپور نباید بیش از ۴ تکرارکننده موجود داشته باشد.



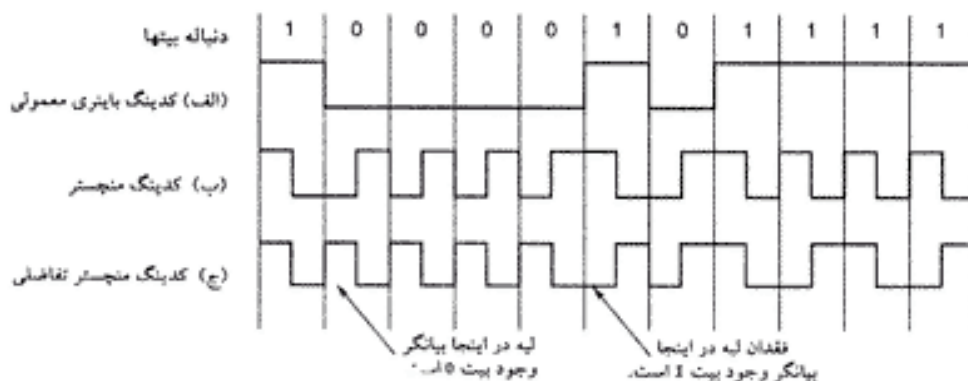
شکل ۴-۱۵. توپولوژی های مختلف کابل (الف) خطی (ب) ستون فقرات (ج) درختی (د) چندبخشی.

۲-۳-۴ کدینگ منچستر

در هیچیک از نسخه‌های مختلف اترنت، از روش معمولی کدینگ (یعنی صفر ولت برای بیت 0 و ۵ ولت برای بیت 1) استفاده نشده است چراکه منجر به برخی اشکالات و ابهام [در دریافت بیت‌ها] خواهد شد: مثلاً اگر ایستگاهی رشته بیت 0001000 را ارسال کند، ایستگاه‌های دیگر، ممکن است به غلط آنرا 10000000 یا 01000000 تشخیص بدهد زیرا ایستگاه نمی‌تواند تفاوت بین آزاد بودن خط (معادل صفر ولت) و بیت صفر (باز هم معادل صفر ولت) را تشخیص بدهد. این مسئله را می‌توان با در نظر گرفتن +۱ ولت برای بیت یک و -۱ ولت برای بیت صفر، حل کرد ولیکن باز هم مشکل اختلاف فرکانس نمونه‌برداری از سیگنال (نسبت به فرکانس اصلی) باقی می‌ماند. تفاوت در سرعت سیگنال ساعت فرستنده و گیرنده باعث خواهد شد که این دو از حالت سنکرون خارج شوند و در اینصورت محدوده هر بیت قابل تشخیص نخواهد بود؛ بالاخص وقتی یک دنباله طولانی پی‌درپی صفر یا دنباله یک پشت سرهم ارسال شود. روش معمولی کدینگ را در شکل ۴-۱۶-الف می‌بینید.

به روشی نیاز است تا گیرنده بتواند به درستی شروع، پایان یا وسط هر بیت را بدون نیاز به یک سیگنال ساعت خارجی تشخیص بدهد. دو روش به نامهای «کدینگ منچستر» و «کدینگ منچستر تفاضلی» دارای چنین قابلیت‌هایی هستند. در روش منچستر هر بیت از لحاظ زمانی به دو نیم بیت تقسیم می‌شود: برای ارسال بیت ۱، در نیم بیت اول، ولتاژ بالا (High) و در نیمه دوم ولتاژ پائین (Low) قرار داده می‌شود. برای بیت صفر برعکس عمل می‌شود: در نیمه اول ولتاژ پائین و در نیمه دوم ولتاژ بالا ارسال می‌گردد. این روش اطمینان می‌دهد که در وسط هر بیت یک لبه (Transition) وجود دارد. اشکال روش منچستر آنست که در مقایسه با روش معمولی، به پهنای باند دو برابر نیاز دارد چرا که طول هر پالس نصف طول یک بیت است. [در حقیقت برای هر بیت ۲ پالس ارسال می‌شود]. به عنوان مثال برای ارسال داده با سرعت ده مگابیت بر ثانیه، سیگنال تولیدی باید با نرخ ۲۰ میلیون بار در ثانیه، تغییر سطح ولتاژ داشته باشد. روش کدینگ منچستر در شکل ۴-۱۶-ب نشان داده شده است.

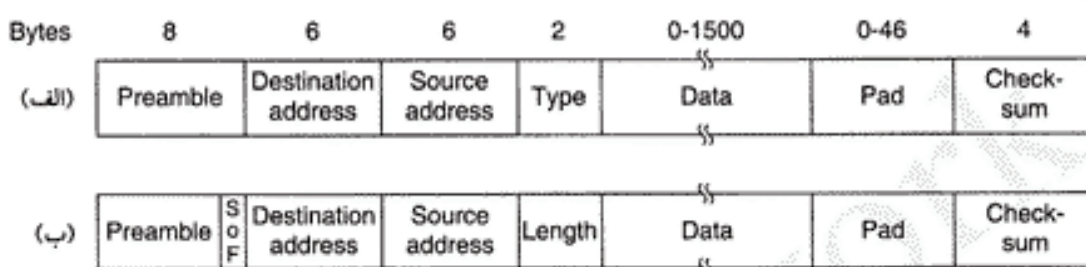
به گونه‌ای که در شکل ۴-۱۶-ج می‌بینید روش منچستر دیفرانسیلی نیز گونه‌ای از روش منچستر معمولی است. عدم وجود هرگونه لبه در ابتدای هر بیت (ابتدای Bit Time) نشان دهنده بیت ۱ است و وجود لبه در ابتدای هر بیت، بیت صفر را مشخص می‌کند. در هر دو حالت، قطعاً یک لبه در میانه هر بیت وجود دارد. روش منچستر تفاضلی به ابزارهای پیچیده‌تری نیازمند است ولیکن در عوض ایمنی بیشتری در مقابل نویز از خود نشان می‌دهد. تمام سیستمهای اترنت برای سادگی از روش کدینگ منچستر معمولی استفاده کرده‌اند و در آنها سطح بالای سیگنال (حالت High) مقدار +۰/۸۵ ولت و سطح پایین سیگنال (حالت Low) مقدار -۰/۸۵ ولت و مقدار DC سیگنال نیز صفر است. در اترنت از روش منچستر تفاضلی استفاده نمی‌شود ولیکن در برخی از شبکه‌های محلی دیگر (مثل IEEE 802.5 Token Ring) از آن استفاده شده است.



شکل ۴-۱۶. (الف) کدینگ با پهنای معمولی (ب) کدینگ منچستر (ج) کدینگ منچستر تفاضلی.

۳-۳-۴ پروتکل زیرلایه MAC در اترنت

قالب اصلی فریم DIX (پیشنهاد شرکت های DEC، Intel، Xerox) در شکل ۴-۱۷ الف نشان داده شده است. هر فریم با هشت بایت Preamble (دیبایچه) شروع می شود که تمام بایتها دارای الگوی 10101010 هستند. کدینگ منچستر این الگوی هشت بایتی، بمدت ۶/۴ میکروثانیه یک سیگنال ساعت مربعی ده مگاهرتز تولید می کند تا یکمک آن گیرنده بتواند سیگنال ساعت خود را با سیگنال ساعت فرستنده سنکرون کند. گیرندگان موظفند با بهره گیری از ویژگی کدینگ منچستر تا انتهای فریم، سنکرون باقی بمانند و محدوده بیتها را به درستی تشخیص بدهند.



شکل ۴-۱۷. قالب فریم (الف) اترنت DIX (ب) اترنت IEEE 802.3

در این فریم دو آدرس تعریف شده است: یکی برای آدرس مقصد و یکی برای آدرس مبدا. طبق استاندارد، استفاده از آدرسهای ۲ یا ۶ بایتی مجاز است ولیکن در مشخصات معرفی شده برای استاندارد 10Mbps، آدرسها صرفاً شش بایتی هستند.^۱ اگر مقصد فریم یک ایستگاه واحد باشد، پرازشترین بیت آدرس مقصد، صفر و اگر مقصد فریم یک گروه از ایستگاهها باشد، ۱ است. آدرسهای گروهی این امکان را فراهم می آورند تا چندین ایستگاه بتوانند به یک آدرس واحد گوش بدهند. وقتی فریمی به یک آدرس گروهی ارسال می شود تمام ایستگاههای گروه قادرند آنرا دریافت نمایند. ارسال فریم برای یک گروه، اصطلاحاً چندپخش (Multicast) نامیده می شود. هرگاه تمام بیتهای آدرس مقصد در یک فریم، ۱ باشند به این آدرس «پخش فراگیر» (Broadcast) گفته می شود. فریمی که تمام بیتهای فیلد آدرس مقصد آن ۱ است توسط همه ایستگاههای شبکه دریافت خواهد شد. تفاوت بین ارسال «چندپخش» و «فراگیر» آنقدر مهم است که ارزش تکرار مجدد را دارد: یک فریم چندپخش برای یک گروه انتخابی از ایستگاهها در شبکه اترنت ارسال می شود در حالیکه فریم فراگیر، برای تمام ایستگاههای شبکه ارسال می گردد. ارسال چندپخش، انتخابی و منعطف است ولیکن نیاز به مدیریت گروهها دارد. در عوض ارسال فراگیر غیر منعطف و غیرانتخابی است ولیکن به مدیریت گروه نیازی ندارد.

یکی دیگر از ویژگیهای جالب آدرس دهی در اترنت آنست که بیت چهل و ششم (مجاور بیت پرازش)، سراسری یا محلی بودن آدرسها را مشخص می کند. آدرسهای محلی، آدرسهای هستند که توسط مسئول شبکه تعیین شده و در خارج از شبکه محلی هیچ معنا و ارزشی ندارند. در مقابل آدرسهای سراسری بطور خاص و مرکزی توسط IEEE اختصاص داده می شوند تا این اطمینان حاصل شود که هیچ دو ایستگاهی در کل دنیا دارای آدرس سراسری یکسان نیستند. ۴۶ بیت باقیمانده (۴۶ = ۲ - ۴۸) فضائی معادل $10^{13} \times 7$ آدرس سراسری ایجاد می کند. ایده اصلی آنست که هر ایستگاه به صورت یکتا و منحصر به فرد آدرس دهی شود. تعیین موقعیت و مشخص کردن ایستگاه مقصد، برعهده لایه شبکه گذاشته شده است.

در ادامه، فیلد Type تعریف شده که به گیرنده فریم تفهیم می کند که با این فریم چه کاری بکند. وقتی که

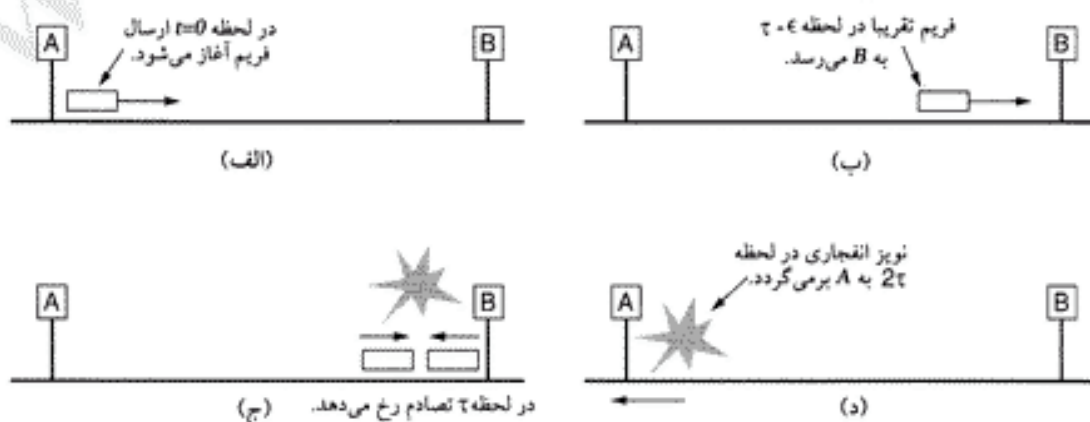
۱. امروزه در هیچیک از گونه های مختلف اترنت آدرسهای دوبایتی وجود خارجی ندارند. -م

چندین پروتکل لایه شبکه، بطور همزمان بر روی یک ماشین واحد اجرا شده باشند، هسته سیستم عامل پس از دریافت فریم، براساس این فیلد تصمیم می‌گیرد که آنرا به دست چه پروسه‌ای بسپارد. محتوای فیلد Type، پروسه‌ای که باید این فریم را تحویل بگیرد، مشخص می‌کند.

در ادامه، «فیلد داده» قرار می‌گیرد که گنجایش آن حداکثر ۱۵۰۰ بایت است. محدودیت ۱۵۰۰ بایتی، در زمان ارائه استاندارد DIX، بدخواه انتخاب شد و انگیزه اصلی طراحان از این انتخاب، صرفه‌جویی در میزان حافظه لازم برای کارتهای شبکه بوده است زیرا در آن سالها (۱۹۷۸) حافظه RAM گران تمام می‌شد.

گذشته از آنکه طول فریم دارای سقف حداکثر است، حداقل طول فریم نیز محدودیت دارد. گاهی ارسال فریمهای بدون داده، مفید خواهد بود ولی استفاده از چنین فریمهایی می‌تواند مشکل آفرین باشد. وقتی تصادم کشف می‌شود، فرستنده از ارسال باقیمانده فریم جاری صرفنظر می‌کند و بدین نحو دنباله‌ای از بیتها (که در حقیقت قطعه ابتدایی فریم است) روی کانال رها می‌شود. برای تشخیص فریمهای معتبر از فریمهای آشغال، در اترنت نیاز است که طول فریمهای معتبر حداقل ۶۴ بایت باشد که این ۶۴ بایت از ابتدای آدرس مقصد تا انتهای فیلد Checksum در نظر گرفته می‌شود. اگر بخش داده فریم از ۴۶ بایت (۱۸-۶۴) کمتر باشد در فیلد Pad آنقدر صفر اضافه می‌شود تا اندازه فریم به حداقل اندازه مجاز برساند.

دلیل دیگر (و بسیار مهمتر) در محدودیت حداقل اندازه برای هر فریم، آنست که مبدا ایستگاه قبل از رسیدن اولین بیت از فریمش به انتهای کابل، ارسال خود را به پایان برساند زیرا ممکن است تصادم بوجود بیاید [و چون ارسال فریم خاتمه یافته، ایستگاه از این موضوع باخبر نخواهد شد. -م] این مشکل در شکل ۴-۱۸ نشان داده شده است. در زمان $t=0$ ایستگاه A در یک طرف شبکه شروع به ارسال فریم روی کابل می‌کند. فرض کنید زمان تاخیر رسیدن فریم به انتهای کابل τ ثانیه باشد. دقیقاً قبل از رسیدن فریم به انتهای کابل (یعنی در لحظه $t = \tau$) آخرین ایستگاه متصل به کابل یعنی B شروع به ارسال فریم خود می‌نماید. حال وقتی B تشخیص می‌دهد که توان دریافتی از کانال از توان ارسالی بیشتر است متوجه بروز تصادم می‌شود و با قطع ارسال خود و تولید یک نویز ۴۸ بیتی قوی، بروز تصادم را به اطلاع بقیه ایستگاهها می‌رساند. عبارت دیگر با ارسال یک نویز قوی این اطمینان حاصل می‌شود که هیچ ایستگاهی از پدیده تصادم بی‌خبر نخواهد ماند. در زمان حدود 2τ ، فرستنده متوجه نویز شده و از ارسال دست می‌کشد. سپس قبل از تلاش مجدد برای ارسال، به اندازه یک عدد تصادفی صبر می‌کند.



شکل ۴-۱۸. کشف تصادم می‌تواند تا زمان 2τ طول بکشد.

اگر ایستگاه تلاش کند فریمی کوتاه ارسال نماید و تصادم پدید بیاید، اگر چه ممکن است از تصادم مطلع شود ولی چون قبل از زمان بازگشت نویز (2τ)، ارسال فریم به پایان رسیده، فرستنده به غلط نتیجه می‌گیرد که فریم او

با موفقیت ارسال شده است و تکرار ارسال ممکن نخواهد بود. برای اجتناب از چنین وضعیتی، تمام فریمها بایستی از لحاظ زمانی از (2 τ) بیشتر باشند تا اطمینان حاصل شود که قبل از خاتمه ارسال فریم، تصادم کشف خواهد شد. برای شبکه اترنت 10Mbps با کابلی به طول حداکثر ۲۵۰۰ متر و چهار تکرارکننده (طبق تعریف 802.3)، زمان رفت و برگشت سیگنال (با احتساب تاخیر انتشار چهار تکرارکننده) در بدترین حالت تقریباً پنجاه میکروثانیه محاسبه شده که قطعاً قابل صرفنظر نیست. بدین ترتیب حداقل طول فریم باید به اندازه ای باشد که از لحاظ زمانی حداقل ۵۰ میکروثانیه باشد. در سرعت 10Mbps هر بیت ۱۰۰ نانوثانیه طول می کشد لذا کوچکترین طول فریم که عملکرد صحیح شبکه را تضمین می کند ۵۰۰ بیت خواهد بود. برای افزودن به حاشیه امنیت این مقدار، ۵۱۲ بیت معادل ۶۴ بایت در نظر گرفته شده است. در فیلد Pad از فریمهای کوچکتر از ۶۴ بایت، باید آنقدر داده زائد اضافه شود تا طول آن به ۶۴ بایت برسد.

به تناسب بالا رفتن سرعت شبکه یا باید طول کانال را کاهش داد یا آنکه به طول حداقل فریمها افزود. در یک شبکه LAN با کابلی به طول ۲۵۰۰ متر و نرخ ارسال 1 Gbps، طول حداقل هر فریم، ۶۴۰۰ بایت خواهد بود! در عوض اگر بیشترین فاصله دو ایستگاه به ۲۵۰ متر کاهش یابد طول حداقل فریم به ۶۴۰ بایت کاهش خواهد یافت. این محدودیت زمانی که به سمت شبکه های چند گیگابیتی حرکت کنیم بسیار مشکل آفرین خواهد بود.

آخرین فیلد از فریم اترنت فیلد Checksum (کد کشف خطا) است. در حقیقت این فیلد یک کد ۳۲ بیتی استخراج شده از داده ها است. هرگاه تعدادی از بیت های فریم (در اثر نویز روی کانال) اشتباه دریافت شوند، کد Checksum آن نادرست خواهد بود و بدین ترتیب خطا کشف می شود. کد کشف خطای به کار رفته در اترنت از نوع CRC است که آنرا در فصل سوم تشریح کردیم. این کدها فقط برای کشف خطا کاربرد دارند و قادر به تصحیح خطا نیستند.

زمانی که IEEE اترنت را استاندارد کرد، دو تغییر کوچک در قالب فریم DIX ایجاد کرد؛ این تغییرات در شکل ۴-۱۷-ب مشهود است. اولین تغییر آن بود که طول Preamble (دبیاچه) را به ۷ بایت کاهش داد و بایت هشتم را به عنوان «بایت مشخص کننده ابتدای فریم» (Start Of Frame Delimiter) به کار برد تا با استانداردهای 802.4 و 802.5 سازگار باشد. تغییر دوم آن بود که فیلد Type را به فیلد Length تغییر کاربری داد. البته در اینصورت گیرنده نمی تواند تشخیص بدهد که با فریم دریافتی چه کند ولی این مسئله با اضافه کردن یک سرآیند اضافی به قسمت داده حل شده است. قالب قسمت داده را در همین فصل و در توضیح «کنترل منطقی لینک» تشریح خواهیم کرد. متأسفانه در زمان ارائه 802.3 سخت افزار و نرم افزار زیادی بر اساس اترنت DIX در حال استفاده بود و شرکت های سازنده و کاربران، چندان علاقمند به تبدیل فیلد Type به Length نبودند! در سال ۱۹۹۷، IEEE کوتاه آمد و اذعان کرد که هر دو روش قابل قبول است. خوشبختانه، تمام مقادیری که در فیلد Type درج می شد بالاتر از عدد ۱۵۰۰ بودند. بدین ترتیب قرار بر آن شد که اگر عدد درون این فیلد از ۱۵۰۰ بیشتر باشد به عنوان فیلد Type فرض شود و در غیر اینصورت بعنوان فیلد Length تعبیر گردد. حال دیگر IEEE می تواند همه را راضی نگه دارد: چه آنهایی که از استاندارد او استفاده می کردند و چه آنهایی که می خواستند کار با استاندارد DIX را بدون احساس کمبود ادامه بدهند!!

۴-۳-۴ الگوریتم عقب گرد نمایی

در این بخش بررسی خواهیم کرد که وقتی تصادم به وقوع می پیوندد روال تولید عدد تصادفی [به منظور انتظار و تلاش مجدد] چگونه است. الگوی ما کماکان مدل شکل ۴-۵ می باشد. پس از بروز تصادم، زمان به تعدادی برش مجزا تقسیم می شود؛ طول این برشها معادل با بیشترین زمان رفت و برگشت سیگنال بر روی کانال (یعنی 2 τ) است. با در نظر گرفتن بیشترین طول کابل در اترنت، هر یک از این برشهای زمانی معادل ۵۱۲ بیت یعنی ۵۱/۲

میکروثانیه خواهد بود.

پس از اولین تصادم هر ایستگاه قبل از تلاش مجدد، به صورت تصادفی صفر یا یک برش زمانی منتظر می ماند. (یعنی بصورت تصادفی یکی از اعداد ۰ یا ۱ را تولید می کند.) اگر دو ایستگاه با هم تصادم کنند و اعداد تصادفی تولید شده مشابه باشند، تصادم تکرار خواهد شد. لذا در دومین تصادم متوالی، یکی از اعداد ۰، ۱، ۲، ۳ انتخاب و بهمان تعداد برش زمانی انتظار می کشد. [یعنی اگر عدد ۲ باشد $2 \times 51/2$ میکروثانیه منتظر می ماند.] اگر سومین تصادم متوالی رخ بدهد (که احتمال چنین رخدادی ۰/۲۵ است) تعداد برشهای زمانی انتظار، عددی تصادفی بین صفر تا ۱ - 2^3 (۷-۵) خواهد بود.

بطور عام در تصادم پیاپی i ام، عددی تصادفی بین صفر تا $2^i - 1$ انتخاب می شود و متناسب با عدد انتخابی، بر مبنای برشهای $51/2$ میکروثانیه ای منتظر می ماند. با این حال پس از دهمین تصادم متوالی، بازه تولید اعداد تصادفی بین صفر تا 1024 ثابت خواهد ماند. پس از شانزدهمین تصادم پیاپی، کنترلر کارت شبکه، دیگر ادامه نداده و پیغامی مبنی بر وجود اشکال جدی در شبکه، به کامپیوتر گزارش خواهد کرد. تشخیص بیشتر، بر عهده لایه های بالاتر است.

این الگوریتم که اصطلاحاً «عقب گرد نمایی دودویی» (Binary Exponential Backoff) نامیده می شود، بدین دلیل انتخاب شده تا بتواند خود را بصورت پویا با هر تعداد ایستگاه که در تلاش برای ارسال هستند، تطبیق بدهد. اگر در هر تصادم، زمان تصادفی انتظار به صورت ثابت و در محدوده صفر تا 1023 انتخاب می شد اگر چه احتمال دو تصادم متوالی بسیار ناچیز بود ولی در عوض زمانی که ایستگاه های تصادم کننده باید صبر می کردند، بالغ بر صدها بازه زمانی می شد و تاخیر بالا می رفت. برعکس، اگر بطور دائم یکی از اعداد صفر یا یک انتخاب شود آنگاه اگر به فرض صد ایستگاه سعی در ارسال کنند تصادم های پیاپی آنقدر تکرار می شود تا زمانی که ۹۹ ایستگاه تصادفاً ۱ و یکی از آنها ۰ را انتخاب کند. این اتفاق ممکن است سالها طول بکشد! اگر الگوریتمی داشته باشیم که در آن بازه های زمانی انتظار در پی تصادمات متوالی، بطور نمایی رشد کند، این اطمینان حاصل می شود وقتی تعداد ایستگاه های آماده ارسال کم است تاخیر کمی بوجود بیاید و وقتی تعداد ایستگاه های آماده ارسال زیاد است در مدت زمان معقولی مسئله تصادم حل شود. گذاشتن سقف 1023 ، اجازه رشد بیش از اندازه زمان انتظار را نخواهد داد.

بگونه ای که قبلاً اشاره کردیم در CSMA/CD دریافت فریم تایید نمی شود [یعنی پس از ارسال فریم سالم، دریافت آن گزارش نخواهد شد]. از آنجایی که جان به در بردن از تصادم تضمین کننده عدم خرابی بیتها در اثر نویز کانال نیست، لذا در مقصد هر فریم باید بررسی و در صورت صحت، پیغام تصدیق (ACK) برگردانده شود. طبعاً پیغام تایید وصول، خودش یک فریم معمولی است که برای ارسال آن نیز همانند فریم داده [برای در اختیار گرفتن کانال] باید مبارزه شود. با این وجود با یک تغییر ساده در الگوریتم رقابت، می توان به دریافت پیغام ACK سرعت بخشید. (Tokoro and Tamaru, 1977) تمام کاری که باید انجام شود آنست که پس از خاتمه ارسال فریم، اولین بازه زمانی به ایستگاه مقصد اختصاص داده شود تا پیغام ACK خود را ارسال نماید. [فریم ACK بسیار کوتاه است]. متأسفانه در استاندارد اترنت، چنین قابلیتی گنجانیده نشده است.

۵-۳-۴ کارائی (بازده) اترنت

در اینجا اجازه بدهید کارائی اترنت را در شرایط بار سنگین و ثابت، یعنی شرایطی که در آن همیشه k ایستگاه آماده ارسال هستند، بررسی نماییم. تحلیل دقیق الگوریتم عقب گرد نمایی پیچیده است. به جای آن روش «متکالف» و «باگز» را دنبال کرده و فرض می کنیم که احتمال تکرار ارسال فریم در هر برش رقابت، ثابت باشد. اگر هر ایستگاه در خلال برش رقابت با احتمال p اقدام به ارسال فریم نماید احتمال آنکه یک ایستگاه در همان برش موفق به

ارسال شود مساویست با:

$$A = k.p.(1 - p)^{k-1}$$

A زمانی حداکثر خواهد شد که $p=1/k$ باشد؛ با فرض $p=1/k$ ، وقتی k به سمت بی نهایت میل می کند مقدار A، $1/e$ خواهد بود. احتمال آنکه دوره رقابت دقیقاً از مرحله متوالی ادامه یابد، مساوی با $A.(1-A)^{k-1}$ است. فلذا میانگین تعداد دفعات تصادم در هر ارسال طبق رابطه زیر بدست می آید:

$$\sum_{j=0}^{\infty} j.A.(1-A)^{j-1} = 1/A$$

از آنجایی که هر برش زمانی حدوداً 2τ طول می کشد، میانگین زمان رقابت $W = 2\tau/A$ خواهد بود. با فرض مقدار بهینه برای p (یعنی $1/e$) میانگین تعداد برشهای رقابت هرگز از $2\tau.e$ بیشتر نخواهد شد [زیرا مقدار A در رابطه $W=2\tau/A$ حداکثر $1/e$ است] بدین ترتیب W معادل $2\tau.e$ یعنی حدود 5.45 خواهد بود.

با فرض آنکه تعداد ایستگاه های آماده ارسال، زیاد باشد و ارسال فریم، P ثانیه به طول بینجامد داریم:

$$\text{رابطه ۴-۶} \quad \text{کارآئی کانال} = \frac{P}{P + 2\tau/A}$$

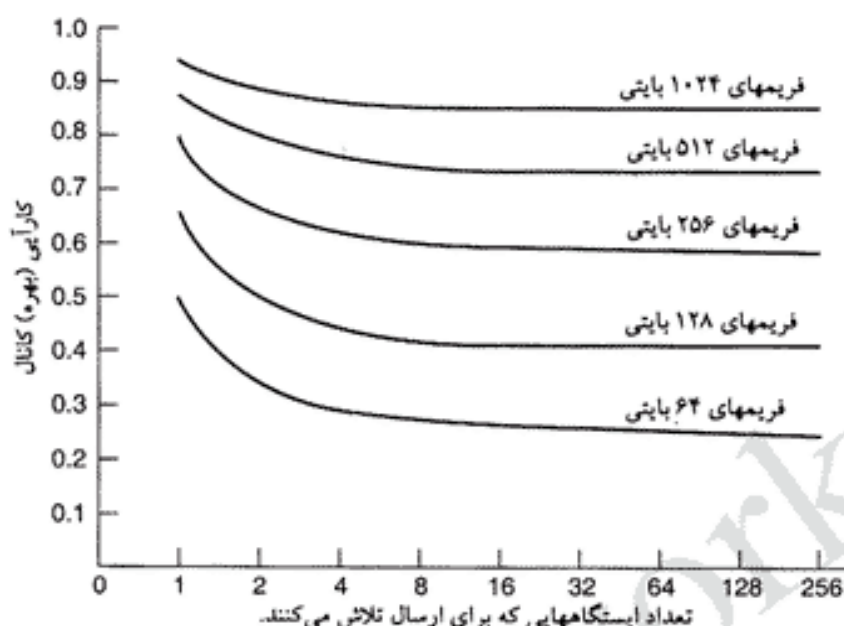
در اینجا می بینیم که فاصله حداکثر کابل بین دو ایستگاه در کارآئی این شبکه وارد می شود (زیرا مقدار τ به طول کابل بستگی دارد). این موضوع نشان می دهد که برای فواصل طولانی باید به سراغ یک توپولوژی بغیر از آنچه که در شکل ۴-۱۵ الف دیده می شود، رفت. هر چه طول کانال زیادتر باشد طول دوره رقابت بیشتر خواهد شد. این نتیجه بروشنی مشخص می کند که چرا در استاندارد اترنت، طول کابل محدودیت دارد. بهتر آنست که رابطه ۴-۶ را به صورت واضحت و بر حسب طول فریم، F، پهنای باند شبکه، B، طول کابل، L و سرعت انتشار سیگنال بر روی کانال، C و مقدار بهینه A یعنی $1/e$ بنویسیم. با قرار دادن $P = F/B$ معادله ۴-۶ به صورت زیر در می آید:

$$\text{معادله ۴-۷} \quad \text{کارآئی کانال} = \frac{1}{1 + 2BLE/cF}$$

وقتی دومین عبارت یعنی $(2B.L.e/c.F)$ بزرگ باشد کارآئی کانال پائین خواهد بود. بالاخص با افزایش پهنای باند یا طول کانال شبکه (یا بطور کلی با افزایش حاصل ضرب B.L)، کارآئی شبکه (برای فریمهای با طول مشخص) کاهش خواهد یافت. متأسفانه بیشتر پژوهشهایی که برای بالا بردن سرعت سخت افزار شبکه انجام می شود، این حاصل ضرب را افزایش می دهد زیرا امروزه عموم مردم به پهنای باند بالا و کانالهای طولانی نیاز دارند (مثلاً برای شبکه فیبر نوری MAN) و محاسبه فوق نشان می دهد که شبکه اترنت برای چنین محیطهایی مناسب نیست. در ادامه همین فصل خواهیم دید که روش دیگری برای پیاده سازی اترنت پیشنهاد شده که در مبحث اترنت مبتنی بر سوئیچ، بدان خواهیم پرداخت.

در شکل ۴-۱۹، منحنی کارآئی کانال بر حسب تعداد ایستگاه های آماده جهت ارسال و با در نظر گرفتن $2\tau = 51.2\mu s$ و نرخ ارسال $B=10Mbps$ ترسیم شده است. [هر یک از منحنی ها به ازای یک F یعنی طول فریم مشخص ترسیم شده اند]. برای فریمهای ۶۴ بیتی کارآئی کانال چندان جالب نیست. در طرف مقابل، با نظر گرفتن ۶۴ بایت برای هر برش رقابت، میانگین زمان رقابت برای فریمهای ۱۰۲۴ بیتی، معادل ۱۷۴ بایت بوده و راندمان کانال تقریباً ۸۵٪ است.

برای آنکه تعداد متوسط ایستگاه های آماده ارسال را در بار سنگین محاسبه نمائیم می توان از روش زیر بهره گرفت: هر فریم برای مدت زمانی معادل: «طول دوره رقابت به اضافه زمان ارسال فریم» یعنی به اندازه $P+W$ ثانیه، کانال را درگیر خواهد کرد. بدین ترتیب تعداد فریمها در هر ثانیه $1/(P+W)$ خواهد بود. هرگاه ایستگاهها با نرخ



شکل ۴-۱۹. منحنی کارآئی کانال در اترنت 10 Mbps با فرض برشهای رقابت ۵۱۲ بیتی.

میانگین k فریم بر ثانیه به تولید فریم مشغول باشند و سیستم «در حالت k »^۱ باشد، مجموع نرخ تولید فریم در ایستگاههای فعال k فریم بر ثانیه خواهد بود. از آنجایی که در حالت «آرامش» (Equilibrium) بایستی نرخ ورودی و خروجی یکسان باشد لذا می توانیم این دو عبارت را معادل هم قرار داده و آنرا بر حسب k حل کنیم. (دقت کنید که w خود تابعی از k است). روش تحلیل پیچیده و دقیقتری توسط (Bertsekas and Gallager, 1992) ارائه شده است.

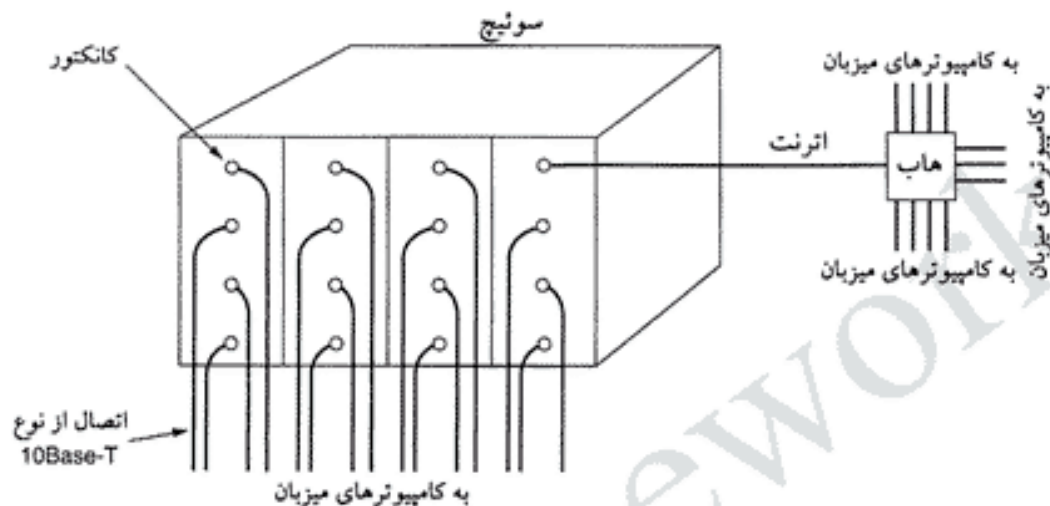
اشاره به این نکته خالی از لطف نیست که تاکنون حجم بسیار گسترده‌ای از مطالعات در خصوص تحلیل کارآئی شبکه اترنت (و شبکه‌های دیگر) انجام شده است؛ در تمام آنها مجازاً فرض شده که ترافیک ایستگاه‌ها مبتنی بر تابع توزیع پواسون تولید می‌شود. وقتی پژوهشگران مطالعات خود را به مدل حقیقی تولید ترافیک، (در عمل) معطوف کردند، مشخص شد که ترافیک شبکه به ندرت پواسون است و «ترافیک هر ایستگاه خاص فقط شبیه به خودش است»! (Paxon and Floyd, 1994; and Willinger et al., 1995) این بدین معناست که میانگین‌گیری در یکدوره طولانی از زمان، «میانگین همواره شده» (Smoothed Average) ترافیک را محاسبه نخواهد کرد؛ یعنی متوسط تعداد فریمها در «هر دقیقه از ساعت» دارای واریانس (پراکندگی) بیشتری نسبت به متوسط تعداد فریمها، در «یک ثانیه از هر دقیقه» خواهد داشت. این نتیجه‌گیری بدین معناست که اغلب مدل‌های عرضه شده برای ترافیک شبکه قابل اعمال در محیط‌های واقعی نیست.

۴-۳-۶ اترنت مبتنی بر سوئیچ

هر چه بر تعداد ایستگاههای شبکه اترنت افزوده شود، ترافیک شبکه افزایش یافته و سرانجام شبکه LAN اشباع خواهد شد. یکی از روشهای کاهش این مشکل آنست که از اترنت 10Mbps به سوی اترنت 100Mbps حرکت کنیم ولیکن با رشد کاربردهای چندرسانه‌ای، اترنت 100Mbps یا حتی 1-Gbps نیز، اشباع خواهد شد.

۱. «حالت k » و «حالت آرامش» از اصطلاحات نظریه صف است. برای درک این مبحث باید اندکی در خصوص نظریه صف (Queuing Theory) مطالعه قبلی داشته باشید. - م

خوشبختانه راه دیگری برای مواجهه با بار سنگین وجود دارد: «اترنت مبتنی بر سوئیچ». این مدل در شکل ۴-۲۰ به تصویر کشیده شده است. در مرکز این سیستم «سوئیچ» قرار گرفته که از یک Backplane پرسرعت و فضائی برای نصب ۴ تا ۳۲ کارت اتصال^۱ خط^۱ تشکیل شده است و هر کارت ۱ تا ۸ کانکتور دارد. بطور معمول، کانکتورها از نوع 10Base-T هستند که از طریق سیمهای زوجی بهم تابیده، مستقیماً به یک کامپیوتر میزبان متصل می شوند.



شکل ۴-۲۰. مثالی ساده از اترنت مبتنی بر سوئیچ.

هرگاه ایستگاهی بخواهد یک فریم اترنت را بفرستد، آن فریم را در قالب عادی و استاندارد سازماندهی کرده و آنرا برای سوئیچ می فرستد. کارت واسط دریافت کننده فریم، ابتدا آدرسهای آنرا بررسی می کند تا ببیند آیا گیرنده فریم (مقصد) به همان کارت متصل است؛ اگر مقصد به همان کارت واسط متصل نبود، فریم بر روی Backplane پرسرعت ارسال می شود تا کارت واسط مقصد آنرا دریافت کند. Backplane با سرعتی حدود چندین گیگابیت در ثانیه و با پروتکلی اختصاصی کار می کند.

حال ببینیم وقتی دو ایستگاه متصل به یک کارت مشابه، به بطور همزمان فریمی را ارسال کنند چه اتفاقی می افتد؟ این مسئله به ساختار طراحی آن کارت بستگی دارد. امکان دارد که تمام پورت های یک کارت از طریق یک سیم، مستقیماً به هم وصل شده و یک شبکه محلی [باس] تشکیل شده باشد. آنگاه، تصادماتی که ممکن است بر روی این LAN داخلی رخ بدهد بروش معمولی CSMA/CD حل و فصل شده و ارسال مجدد نیز طبق الگوریتم «عقب گرد دودویی نمایی» (Binary Exponential Backoff) انجام می گیرد. در چنین کارتهائی، از بین ایستگاه های متصل به هر کارت، در آن واحد فقط یک ایستگاه می تواند ارسال کند در حالیکه ایستگاه های متصل به کارتهای متفاوت می توانند بطور موازی [همزمان] ارسال داشته باشند. در این طرح هر کارت برای خودش یک «حوزه تصادم»^۲ (Collision Domain) تشکیل داده و این حوزه، مستقل از حوزه تصادم دیگر کارتهاست. هرگاه فقط یک ایستگاه در هر حوزه وجود داشته باشد [یعنی فقط یک ایستگاه در هر کارت] آنگاه تصادم متنی است و کارائی افزایش خواهد داشت.

در انواع دیگر کارتها، هر یک از پورت های ورودی دارای بافری اختصاصی هستند و بدین ترتیب فریمهای

۱. Plug-in Line Card.

۲. ایستگاههایی که مستقیماً به یک کانال مشترک متصلند و برای ارسال بروش CSMA/CD رقابت می کنند اصطلاحاً در یک

«حوزه تصادم» قرار دارند. س

ورودی در درون حافظه RAM کارت، ذخیره می‌شوند. در این ساختار تمام ایستگاه‌ها می‌توانند بطور همزمان و دوطرفه (Full Duplex)، ارسال و دریافت داشته باشند؛ عملی که در ساختار معمولی CSMA/CD، با یک کانال واحد ممکن نیست. هنگامی که یک فریم بطور کامل دریافت شود، کارت مربوطه می‌تواند بررسی کند که آیا ماشین مقصد به پورتی بر روی همان کارت متصل است یا باید به سمت پورتی بر روی کارت دیگر هدایت شود. در حالت اول، فریم مستقیماً به سمت مقصد ارسال می‌شود. در حالت دوم، فریم از طریق Backplane به سمت کارت مناسب هدایت می‌گردد. در این طراحی، هر پورت برای خود «حوزه تصادم» کاملاً مستقل دارد، یعنی هرگز تصادم رخ نمی‌دهد. در این ساختار، ظرفیت کل سیستم در مقایسه با 10Base5 (که در آن برای کل سیستم یک حوزه تصادم واحد وجود دارد) چندین برابر افزایش خواهد داشت.

از آنجایی که سوئیچ، فریمهای استاندارد اترنت را از پورتهای ورودی خود می‌پذیرد لذا می‌توان از برخی پورتهای در نقش «متمرکز کننده» (Concentrator) استفاده کرد. در شکل ۴-۲۰ پورت سمت چپ/بالایی سوئیچ، بطور مستقیم به یک ایستگاه واحد متصل نیست بلکه مثلاً به یک هاب ۱۲ پورت متصل است. وقتی فریمی به هاب وارد می‌شود، بروش معمول برای بدست آوردن کانال رقابت می‌کند؛ در حالیکه ممکن است تصادم رخ داده و از الگوریتم عقب‌گرد نمایی استفاده شود. فریمی که موفق به ارسال روی کانال شده و به سوئیچ وارد گردد، همانند یک فریم معمولی با آن برخورد می‌شود یعنی برای رسیدن به خط خروجی مناسب از طریق Backplane ارسال و به مقصد هدایت خواهد شد.

اگر چه هابها ارزانتر از سوئیچها هستند ولیکن با سقوط قیمت سوئیچها، هاب به سرعت از صحنه خارج می‌شود، ولی هنوز میراث هابها باقی است!

۴-۳-۷ اترنت سریع

در بدو آن ایام، سرعت 10Mbps برای اترنت بسیار شگفت‌انگیز و رویایی به نظر می‌رسید؛ همانگونه که ظهور مودم‌های ۱۲۰۰ bps در دورانی که فقط مودم‌های آکوستیکی و ابتدائی ۳۰۰ bps رایج بود، پدیده‌ای اعجاب‌انگیز محسوب می‌شد؛ ولیکن هر پدیده نوظهوری به سرعت فرسوده و محو می‌شود. به عنوان نتیجه قانون پارکینسون می‌توان منتظر زمانی بود که ارسال داده‌ها با نرخی معادل با پهنای باند طبیعی هر کانال ارسال شوند. (قانون پارکینسون بیان می‌کند که: «یک کار تا زمانی که وقت برای تکمیل شدن داشته باشد، ادامه می‌یابد.») برای افزایش سرعت شبکه‌ها، گروه‌های صنعتی مختلف دو طرح جدید مبتنی بر شبکه حلقه (Ring) و فیبرنوری ارائه کردند: یکی از آنها طرح شبکه FDDI^۱ و دیگری طرح Fibre Channel^۲ نام داشت. برای کوتاه کردن داستان مفصل آنها، باید اشاره کنیم که اگر چه از هر دو به عنوان ستون فقرات شبکه استفاده شد ولیکن هیچیک از آنها نتوانست برای اتصال کامپیوترهای رومیزی به کاربرد، چراکه در این دو شبکه، مدیریت ایستگاه‌ها بسیار پیچیده بود و نیاز به تراشه‌های پیشرفته داشت که به گران شدن قیمت آنها می‌انجامید؛ سرانجام پرونده آنها به تاریخ پیوست. درسی که باید از سرنوشت اینگونه شبکه‌ها گرفت آنست که باید هر سیستمی را ساده و سهل طراحی کرد.

بهر تقدیر، ناکامی و شکست شبکه‌های LAN مبتنی بر فیبرنوری [مثل FDDI]، کمبود و خلاء اترنت سریعتر از 10Mbps را هر چه بیشتر نمایان می‌کرد. در بسیاری از محیطها، افراد به پهنای باند بیشتری نیاز داشتند و برای رفع این مشکل به ناچار چندین شبکه محلی 10Mbps را نصب و از طریق تکرارکننده (Repeater)، پل

۱. Fiber Distributed Data Interface

۲. نام این طرح Fibre Channel است نه Fiber Channel، زیرا ویرایشگر این طرح بریتانیایی بوده نه آمریکایی! -اندی تاننهام

(Bridge) یا مسیریاب (دروازه - Gateway) به هم متصل کرده بودند. این ساختار بهم پیچیده، گاهی چنان سردرگم و پراشکال می شد که مسئول شبکه احساس می کرد این شبکه ها به وسیله آدامس بادکنکی یا نخ به هم کوک خورده اند!

چنین فضا و شرایطی، IEEE را بر آن داشت تا در سال ۱۹۹۲ کمیته 802.3 را دور هم جمع کند تا در خصوص شبکه محلی سریعتر از ده مگابیت تصمیم بگیرند. یک پیشنهاد آن بود که 802.3 بهمان شکل اصلی حفظ شده و فقط سرعت آن افزایش یابد. پیشنهاد دیگر آن بود که 802.3 بطور کل از نو تدوین شده و ویژگی های جدیدی مثل ارسال ترافیک بی درنگ و مبادله دیجیتالی سیگنال صوت به آن اضافه شود ولی در عین حال (بدلیل مسائل بازار و فروش) نام قبلی آن حفظ شود. پس از بحث و مناقشه فراوان، کمیته تصمیم گرفت که ساختار شبکه 802.3 را به همان شکل اصلی حفظ کند و فقط به سرعت آن بیفزاید.

گروهی که پیشنهاد آنها در خصوص طراحی مجدد 802.3 پذیرفته نشد برای خودشان کمیته تشکیل دادند و شبکه محلی پیشنهادی خود را با نامی دیگر استاندارد کردند که نهایتاً شبکه 802.12 پدید آمد. (کاری که معمولاً هر شخصی با دید صنعتی یا اقتصادی در چنین شرایطی انجام خواهد داد انجام طرحی است که درست و منطقی به نظر می رسد. ولی بازار چیز دیگری می گوید! تاریخ نشان داد که شبکه 802.12 که شرکت هیولت پاکارد بر روی آن سرمایه گذاری کرد اقبالی بدست نیاورد!)

به سه دلیل زیر، کمیته 802.3 تصمیم گرفت اترنت را بدون تغییر در بنیان آن، فقط توسعه بدهد:

۱. نیاز به سازگاری شبکه جدید با شبکه های اترنت موجود

۲. نگرانی از آنکه پروتکل جدید مشکلات پیش بینی نشده داشته باشد!

۳. تمایل به آنکه قبل از تغییر تکنولوژی بتوانند کار را به اتمام برسانند و مشمول زمان نشود.

کار به سرعت انجام گرفت (البته با سرعتی که در عرف کمیته استاندارد است) و حاصل کار، استاندارد 802.3u بود که به صورت رسمی در ژوئن ۱۹۹۵ توسط IEEE تائید و معرفی شد. از دیدگاه فنی، 802.3u استاندارد جدیدی محسوب نمی شود بلکه یک ضمیمه مکمل برای استاندارد موجود 802.3 است. (تا بر سازگاری با استاندارد قبل تاکید شده باشد). از آنجایی که عموم افراد به استاندارد جدید به جای 802.3u، «اترنت سریع» (Fast Ethernet) می گویند ما نیز همین اصطلاح را به کار خواهیم گرفت.

ایده اصلی در اترنت سریع بسیار ساده بود: تمام ویژگیها مثل قالب فریم، واسطها، قواعد و الگوریتمها را بدون تغییر نگاه داشته و فقط «زمان یک بیت» را از 100 نانوثانیه به 10 نانوثانیه کاهش بدهیم. از دیدگاه فنی، این کار در شبکه اترنت با کابل 10Base5 یا 10Base2 بسادگی میسر است و به شرط آنکه طول حداکثر کانال با ضریب ده کاهش یابد، تصادمها نیز بموقع کشف خواهد شد. با این وجود محاسن سیم کشی مبتنی بر 10Base-T (یعنی استفاده از زوج سیم های به هم تابیده) بقدری مفید و چشمگیر بود که اترنت سریع کلاً بر اساس این نوع کابل طراحی شد. لذا کل سیستم اترنت سریع، از هاب یا سوئیچ بهره می گیرد و استفاده از کابل های چندانشعابی، کانکتورهای BNC یا انشعاب های تزریقی (Vampire Tap) مجاز نیست.

ولیکن هنوز چند گزینه دیگر باقی مانده بود که بایستی برای آنها نیز تصمیم گرفته می شد؛ مهمترین آنها، انتخاب انواع سیم زوجی بود که اترنت سریع باید از آنها پشتیبانی می کرد. یکی از نامزدها کابل زوجی رده ۳ (Cat 3) بود. [کابل Cat 3 برای سیم کشی تلفن در ساختمانها به کار می رود]. استدلال این انتخاب آن بود که در دنیای غرب در هر دفتر از ساختمانهای اداری حداقل چهار جفت سیم Cat 3 (یا حتی بهتر از Cat 3) از قبل وجود دارد که در فاصله ای حدود صد متر به جعبه تقسیم تلفن کشیده شده است. بنابراین با استفاده از کابل زوجی Cat 3 این امکان وجود داشت که بدون نیاز به سیم کشی مجدد ساختمان، بتوان کامپیوترهای رومیزی را از طریق اترنت

سریع به هم متصل کرد؛ این ویژگی، امتیاز بزرگی برای اغلب سازمانها و موسسات به حساب می آید. بزرگترین اشکال سیم های زوجی Cat 3 آنست که قادر نیستند سیگنالی با نرخ تغییر 200Mbaud/sec را در فاصله صد متر حمل کنند (100Mbps در روش منچستر معادل 200Mbaud/sec است)؛ در استاندارد 10Base-T، فاصله صد متر، بیشترین فاصله مجاز کامپیوتر از هاب است. از طرفی کابل های رده ۵ (یعنی Cat 5) می توانند چنین سیگنالی را به راحتی در فاصله صد متری منتقل کنند؛ فیبر نوری حتی قادر است با نرخ بسیار بالاتر، این کار را انجام بدهد.

تصمیم نهائی بر آن شد که مطابق با مشخصات شکل ۴-۲۱، هر سه گزینه در استاندارد جدید مجاز شمرده شود، ولیکن به گزینه استفاده از سیم های Cat 3 بهبودهایی داده شد تا بتواند سیگنال با نرخ مورد نیاز را حمل کند.

مزایا	حداکثر طول هر قطعه	نوع کابل	نام کابل
از کابل های معمولی تلفن (UTP Cat 3) استفاده می کند.	100 m	Twisted pair	100Base-T4
ارسال دوطرفه کامل با نرخ 100Mbps و کابل UTP Cat 5	100 m	Twisted pair	100Base-TX
ارسال دوطرفه کامل با نرخ 100Mbps در فواصل طولانی	2000 m	Fiber optics	100Base-FX

شکل ۴-۲۱. کابل کشی اترنت سریع.

در الگوی سیم کشی با کابل های معمولی Cat 3 (از نوع UTP) که 100Base-T4 نامیده می شود از نرخ سیگنالینگ ۲۵ مگاهرتز استفاده شده است فلذا فقط ۲۵ درصد سریعتر از استاندارد ۲۰ مگاهرتز در اترنت استاندارد است (با مرور شکل ۴-۱۶ به یاد بیاورید که در کدینگ منچستر، هر یک از ده میلیون بیت در ثانیه نیاز به دو پالس ساعت دارد). بهر حال برای تامین پهنای باند مورد نیاز در 100Base-T4 به چهار زوج سیم Cat 3 نیاز است. [۴×۲۵MHz] از آنجایی که سالهاست در استاندارد سیم کشی تلفن ساختمانها، از کابل هایی استفاده می شود که دارای چهار جفت سیم هستند لذا بسیاری از ادارات چنین امکانی را در اختیار داشتند. البته این موضوع بدین معناست که مرکز تلفن را کنار بگذارید ولی در عوض هزینه بسیار کمتری برای داشتن سیستم پست الکترونیکی سریعتر صرف خواهد شد!

از بین چهار زوج سیم، از یکی به عنوان خط ورودی دائم به هاب، از یکی به عنوان خط خروجی دائم از هاب و از دوتای دیگر بصورت قابل تنظیم در جهت فعلی ارسال [از هاب به کامپیوتر یا بالعکس] استفاده می شود. برای صرفه جویی در پهنای باند مورد نیاز، از روش کدینگ منچستر استفاده نشده است چراکه با ابداع مولدهای مدرن سیگنال ساعت و در فاصله ای بدین کوتاهی، اصولاً نیازی هم به کدینگ منچستر نیست. بجای آن از سیگنال های Ternary استفاده شده که در این کدینگ، در هر سیکل سیگنال ساعت یکی از مقادیر ۰، ۱ یا ۲ ارسال می شود. (یعنی پالسها سه سطحی هستند). با داشتن سه زوج سیم در جهت ارسال و روش سیگنالینگ سه سطحی، ۲۷ (۳×۳×۳) سمبل مختلف قابل ارسال است؛ بدین ترتیب با ارسال هر سمبل، می توان ۴ بیت را (به همراه مقداری افزونگی) انتقال داد. [۴ بیت معادل ۱۶ سمبل و ۱۱ حالت افزونگی] ارسال چهار بیت در هر سیکل از سیگنال های ۲۵ مگاهرتزی، نرخ معادل 100 مگابیت بر ثانیه را فراهم می آورد. همچنین، در جفت سیم باقیمانده (زوج چهارم) یک کانال معکوس ۳۳/۳ مگابیت بر ثانیه ایجاد می شود. این ساختار که اصطلاحاً 8B/6T نامیده می شود (بمعنای آنکه هشت بیت در چهار تریت Trit- نگاشته می شود) روشی جالب و جذابی نیست ولی بهر حال با ساختار موجود سیم کشی، کار می کند.

الگوی سیم کشی با سیم های زوجی Cat 5 (که اصطلاحاً 100Base-TX نامیده می شود) ساده تر است زیرا این سیمها قادر به حمل سیگنال هایی با نرخ ۱۲۵ مگاهرتز هستند. بدین ترتیب برای هر ایستگاه فقط به دو زوج سیم

نیاز است: یکی ورودی به هاب و دیگری خروجی از هاب. در این الگو نیز از روش کدینگ معمولی استفاده نشده است: در عوض از روش 4B/5B بهره گرفته شده که مشابه و سازگار با FDDI است. در روش 4B/5B، هر گروه متشکل از پنج کلاک متوالی (معادل پنج بیت) و شامل ۳۲ حالت مختلف است. ۱۶ تا از این ترکیبات برای ارسال ۴ بیت داده به کار می رود. [بعبارت ساده تر در روش 4B/5B به ازای هر چهار بیت، پنج بیت ارسال می شود.] برخی از ۱۶ حالت باقیمانده برای عملیات کنترلی نظیر مشخص کردن ابتدا و انتهای فریم کاربرد دارد. ترکیبات شانزده گانه داده به نحوی انتخاب شده اند که در الگوی سیگنال تولیدی، لبه لازم برای سنکرون ماندن سیگنال ساعت وجود داشته باشد. سیستم 100Base-TX دو طرفه همزمان است: ایستگاه می تواند داده ها را با نرخ ۱۰۰ مگابیت در ثانیه ارسال و بطور همزمان دریافت نماید. اغلب به روشهای 100Base-TX و 100Base-T4 به اختصار 100Base-T گفته می شود.

آخرین گزینه، 100Base-FX است که در آن از دو رشته فیبر نوری مالتی مود (Multimode) استفاده می شود؛ هر یک از تارهای نوری قادر به حمل ۱۰۰ مگابیت در ثانیه به صورت همزمان هستند. مضاف بر این، فاصله بین ایستگاه و هاب تا ۲ کیلومتر قابل افزایش است.

در پاسخ به نیاز عمومی، کمیته IEEE 802 در سال ۱۹۹۷ روش کابل کشی جدیدی به نام 100Base-T2 به استاندارد اضافه کرد که اجازه می داد اترنت سریع با دو زوج سیم معمولی Cat 3 کار کند ولیکن بدلیل استفاده از روش کدینگ خاص، به یک پردازنده سیگنال دیجیتال (DSP Processor) پیچیده نیاز است که این انتخاب را اندکی گران قیمت می کند. اکنون از این ساختار به دلانلی مثل پیچیدگی و قیمت بالا و این حقیقت که بسیاری از ساختمان های اداری به سیم کشی با کابل Cat 5 تن داده اند، به ندرت استفاده می شود.

در 100Base-T، جهت اتصال ایستگاه ها به یکدیگر می توان از دو نوع ابزار استفاده کرد: هاب یا سوئیچ. (شکل ۴-۲۰) در هاب تمام خطوط ورودی (یا حداقل تمام ورودی هائی که به یک کارت واحد وارد می شوند) به صورت منطقی به هم متصل هستند و یک حوزه تصادم واحد را ایجاد می کنند. در این ابزار دقیقاً مثل اترنت، تمام قواعد استاندارد (نظیر الگوریتم عقب گرد نمایی) اعمال می شود، خصوصاً آنکه در هر لحظه تنها یک ایستگاه می تواند ارسال داشته باشد؛ بعبارت دیگر، در هاب ماهیت ارتباط، «دو طرفه غیر همزمان» (Half Duplex) است. در یک سوئیچ هر یک از فریمهای ورودی بر روی حافظه کارت متصل به خط، بافر می شود و در صورت نیاز از طریق یک Backplane بسیار سریع، از کارت مبدا به کارت مقصد هدایت می گردد. ساختار Backplane استاندارد سازی نشده و نیازی هم بدین امر نبوده است چرا که به عنوان بخشی پنهان در درون سوئیچ انجام وظیفه می کند. با تکیه بر تجارب گذشته می توان پیش بینی کرد که تولید کنندگان سوئیچ به شدت در حال رقابت هستند تا Backplane سوئیچها سریعتر شده و کارائی کل سیستم افزایش یابد. از آنجایی که کابل های 100Base-FX برای کشف بموقع تصادم در شبکه اترنت بیش از اندازه طولانی هستند، فلذا این کابلها لزوماً باید. به سوئیچها متصل شوند؛ بدین ترتیب هر ایستگاه برای خود یک «حوزه تصادم» مستقل پدید آورده و تصادم متفی خواهد بود. استفاده از هاب در 100Base-FX مجاز نیست.

به عنوان آخرین نکته باید اشاره کرد که تمام سوئیچها (مجازاً) می توانند با تلفیقی از ایستگاه های 10Mbps و 100Mbps کار کنند تا ارتقاء سیستمهای موجود به سیستمهای سریعتر ساده تر شود. در صورت افزایش نیاز یک سایت به تعداد بیشتری ایستگاه 100Mbps، تنها کافی است به تعداد لازم کارت های ورودی خریداری شده و درون سوئیچ نصب شود. [به شکل ۴-۲۰ نگاه کنید.] در حقیقت در خود استاندارد روشی پیش بینی شده تا دو ایستگاه بتوانند بر سر نرخ ارسال (10Mbps یا 100Mbps) با هم توافق کرده و ماهیت ارتباط (Full Duplex یا Half Duplex بودن ارتباط) را انتخاب نمایند. بسیاری از محصولات اترنت از این ویژگی برخوردار هستند تا

بتوانند به صورت خودکار خودشان را پیکربندی نمایند.

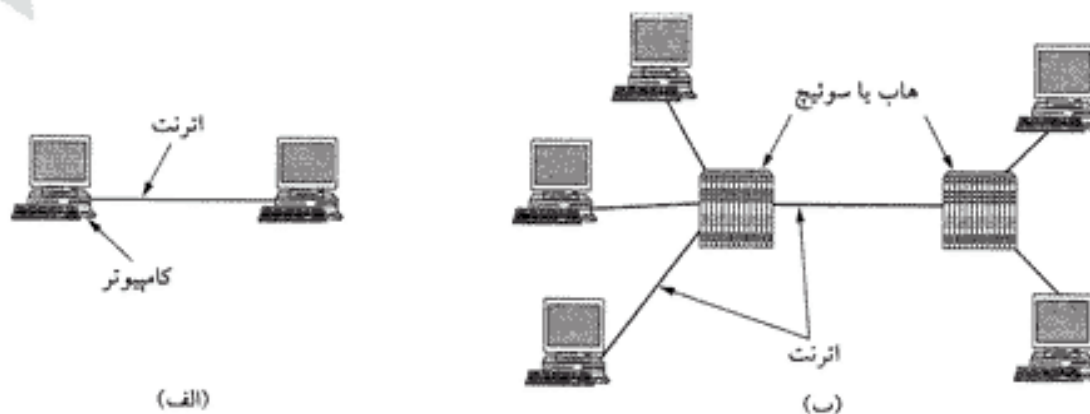
۸-۳-۴ اترنت گیگابیت (Gigabit Ethernet)

هنوز جوهر مستندات استاندارد اترنت سریع، خشک نشده بود که کمیته 802 کار را بر روی اترنت سریعتر از آن آغاز کرد (۱۹۹۵). این تحقیقات سریعاً تحت عنوان اترنت گیگابیت به ثمر رسید و در سال ۱۹۹۸ توسط IEEE با نام 802.3z تصویب شد. پیشنهاد حرف z در نام این استاندارد بدین مضمون بود که اترنت گیگابیت آخرین حلقه در مسیر تکاملی اترنت خواهد بود مگر آنکه کسی بتواند حرف جدیدی پس از z ابداع کند!! در ادامه برخی از ویژگی‌های اساسی اترنت گیگابیت را تشریح خواهیم کرد. برای آگاهی بیشتر می‌توان به مقاله (Seifert, 1998) مراجعه کرد.

اهداف کمیته 802.3z مشابه با اهداف کمیته 802.3u بود: اترنت ده برابر سریعتر شود و همچنان با استاندارد موجود اترنت سازگاری داشته باشد. بویژه اترنت گیگابیت باید از خدمات «انتقال دیتاگرام بدون تصدیق دریافت فریم»^۱ به صورت تک‌پخش و چندپخش^۲ و با استفاده از همان ساختار ۴۸ بیتی آدرس موجود، پشتیبانی می‌کرد. همچنین قالب فریمها و طول حداقل و حداکثر هر فریم باید مشابه با قبل انتخاب می‌شد. استاندارد نهایی، تمام این اهداف را برآورده کرده است.

کل پیکربندی شبکه اترنت گیگابیت، به جای آنکه همانند اترنت کلاسیک دارای ساختار باس چنداتصال (Multidrop) باشد صرفاً نقطه‌به‌نقطه (Point To Point) است. [یعنی به جای آنکه همه ایستگاه‌ها به یک کابل مشترک متصل باشند بطور مستقیم به یک سوئیچ متصل می‌شوند. -م] ساده‌ترین الگوی پیکربندی اترنت گیگابیت، در شکل ۴-۲۲ الف نشان داده است؛ در این شکل دو کامپیوتر به صورت مستقیم به هم متصل شده‌اند. الگوی رایج دیگر که در شکل ۴-۲۲ ب دیده می‌شود شامل یک سوئیچ یا هاب است که به چندین کامپیوتر یا چند سوئیچ یا هاب دیگر متصل هستند. در هر دو الگو، به هر کابل اترنت صرفاً دو ابراز متصل است، نه کمتر نه بیشتر [یعنی دو کارت اترنت گیگابیت در دو سر کابل].

اترنت گیگابیت می‌تواند به دو روش عمل کند: «حالت دوطرفه همزمان» (Full Duplex) و «حالت دوطرفه غیرهمزمان» (Half Duplex). عملکرد طبیعی و پیش فرض سیستم، حالت دو طرفه همزمان است و بدین ترتیب این امکان وجود دارد که ترافیک داده‌ها بتواند به صورت همزمان در دو جهت جریان داشته باشد. از این حالت زمانی استفاده می‌شود که یک سوئیچ مرکزی وجود داشته و به کامپیوتر دیگر (یا سوئیچهای دیگر) در پیرامون



شکل ۴-۲۲. (الف) اترنت گیگابیت با دو ایستگاه. (ب) اترنت گیگابیت با چند ایستگاه.

۱. Unacknowledged Datagram

۲. Unicast & Multicast

خود متصل باشد. در این پیکربندی تمام خطوط بافر شده اند^۱ و طبعاً هر کامپیوتر یا سوئیچ اجازه دارد هر زمان که تمایل داشت فریم خود را ارسال کند. فرستند مجبور نیست کانال را شلوغ کند و حضور دیگران بر روی کانال را تشخیص بدهد چرا که در این ساختار اصولاً هرگونه رقابتی متفی است و هرگز تصادم رخ نخواهد داد. از آنجایی که بین سوئیچ و کامپیوتر دو کابل مستقل وجود دارد، ارسال همزمان از کامپیوتر به سوئیچ (حتی اگر سوئیچ در حال ارسال فریم برای همان کامپیوتر باشد)، میسر است. از آنجایی که هیچ رقابتی مطرح نیست لذا پروتکل CSMA/CD کاربردی ندارد و حداکثر طول کانال فقط بر اساس توان سیگنال ارسالی [و میزان تضعیف آن روی کابل] تعیین می شود و ربطی به مدت زمان بازگشت نویز حاصل از تصادم [و تاخیر انتشار کابل] ندارد. سوئیچها آزادند که با نرخ مختلف ارسال و به صورت تطبیقی عمل کنند. در اترنت گیگابیت همانند اترنت سریع، از پیکربندی خودکار حمایت می شود.

حالت دیگر عملکرد اترنت گیگابیت، حالت دو طرفه غیر همزمان (Half Duplex) است و زمانی کاربرد دارد که کامپیوترها به جای وصل به سوئیچ به هاب متصل باشند. یک هاب فریمهای ورودی را بافر نخواهد کرد و در عوض به صورت داخلی تمام خطوط را به صورت الکتریکی به هم متصل کرده و همانند اترنت کلاسیک کابلی چندانصالی را شبیه سازی می کند. در این حالت وقوع تصادم محتمل بوده و به پروتکل CSMA/CD نیاز خواهد بود. از آنجایی که کوتاه ترین فریم مجاز (یعنی فریم ۶۴ بیتی)، در اترنت گیگابیت صدبرابر سریعتر از اترنت کلاسیک ارسال می شود لذا حداکثر طول کابل بایستی ۱۰۰ برابر کاهش یابد تا آنکه ویژگی بنیادی مورد نیاز در CSMA/CD یعنی «عدم خاتمه ارسال فریم قبل از بازگشت نویز حاصل از تصادم»، برآورده شود. (یعنی طول کابل باید حداکثر ۲۵ متر باشد). با کابلی به طول ۲۵۰۰ متر و سرعت 1Gbps، قبل از آنکه یک فریم ۶۴ بیتی بتواند حتی یکدهم از مسیر خود را طی کند، فرستنده آن ارسال خود را به پایان رسانده و از تصادم احتمالی مطلع نخواهد شد!

کمیت 802.3z بدین نتیجه رسید که شعاع ۲۵ متر برای اترنت گیگابیت قابل قبول و مناسب نیست، فلذا دو ویژگی جدید به استاندارد افزود تا شعاع شبکه افزایش یابد. اولین ویژگی که «توسیع حامل» (Carrier Extension) نامیده شده سخت افزار را وادار می کند تا آنقدر اطلاعات زائد در انتهای فریم معمولی اضافه کند تا طول فریم حداقل ۵۱۲ بایت شود. از آنجایی که این اطلاعات زائد در سخت افزار مبداء اضافه و در سخت افزار مقصد حذف می شوند لذا نرم افزار از انجام چنین عملی بی اطلاع خواهد بود و هیچ تغییری در ساختار نرم افزار موجود نیاز نخواهد بود. البته طبیعی است که فرستادن فریمی ۵۱۲ بیتی برای ارسال ۴۶ بایت داده خام کاربر، بهره مفیدی معادل ۹ درصد دارد.

دومین ویژگی که اصطلاحاً Frame Bursting نامیده شده اجازه می دهد که فرستنده دنباله ای از چندین فریم را در یک بار ارسال بفرستد. اگر مجموع دنباله فریمها کمتر از ۵۱۲ بایت شود باز هم سخت افزار، داده های زائد [به انتهای آخرین فریم] اضافه می کند. هرگاه تعداد فریمهای منتظر ارسال کافی باشد این روش بسیار کارآمد و نسبت به روش «توسیع حامل» ارجح تر است. این ویژگی جدید، شعاع شبکه را به ۲۰۰ متر افزایش می دهد که برای بسیاری ادارات و موسسات کافی است. اگرچه در اترنت گیگابیت از CSMA/CD پشتیبانی می شود ولیکن تصور آن بسیار دشوار است که سازمانی کارتهای اترنت گیگابیت خریداری و نصب کند ولی آنها را به یک هاب متصل و اترنت معمولی را شبیه سازی نماید! [بهره واقعی اترنت گیگابیت با سوئیچها بدست می آید نه با هاب] اگر چه هابها از سوئیچها بسیار ارزانتر هستند ولی هنوز قیمت کارت های اترنت گیگابیت نسبتاً گران است.

۱. یعنی داده های ارسالی مستقیماً به درون حافظه موقت منتقل می شوند. -م

خرید هابهای ارزان قیمت [برای کارتهای گرانقیمت گیگابیتی] و کاستن از بهره واقعی سیستم، کوفته فکری به نظر می رسد! بهرحال ویژگی «سازگاری با قبل» یکی از نگرانیهای صنایع کامپیوتریست و طبعاً نیاز بوده که کمیته 802.3z آنرا در استاندارد خود لحاظ کند. [ولیکن دلیلی ندارد از ویژگیهای جدید آن به بهای سازگاری با قبل صرفنظر شود]

اترنت گیگابیت مطابق با فهرست ۴-۲۳ از کابلهای مسی و فیبرهای نوری پشتیبانی می کند. تولید سیگنال با نرخ نزدیک به 1-Gbps بدان معناست که منبع مولد نور باید در زمانی زیر یک نانوثانیه خاموش و روشن شود. مولد نور معمولی یعنی LED نمی تواند با این سرعت عمل کند و طبعاً به لیزر نیاز است. استفاده از پرتوهای لیزر با طول موج ۰/۸۵ میکرون (طول موج کوتاه) و ۱/۳ میکرون (طول موج بلند) مجاز است. لیزرهای 0.85 میکرونی ارزانتر هستند ولیکن بر روی فیبرنوری تک مود (Single Mode) کار نمی کنند.

مزایا	حداکثر طول هر قطعه	نوع کابل	نام کابل
از فیبرهای چندموده ۵۰ و ۶۲/۵ میکرون استفاده می کند.	550 m	Fiber optics	1000Base-SX
با فیبرهای تک موده ۱۰ میکرون یا چندموده ۵۰ و ۶۲/۵ میکرون	5000 m	Fiber optics	1000Base-LX
از کابلهای زوجی زره دار (STP) استفاده می کند.	25 m	2 Pairs of STP	1000Base-CX
از کابلهای استاندارد UTP Cat 5 استفاده می کند.	100 m	4 Pairs of UTP	1000Base-T

شکل ۴-۲۳. کابل کشی اترنت گیگابیت.

در اترنت گیگابیت، استفاده از فیبرهای نوری با قطر ۱۰، ۵۰ و ۶۲/۵ میکرون مجاز است. مورد اول برای فیبرنوری تک مود (Single Mode) و دو مورد بعدی برای فیبرهای چندموده هستند. تمام شش ترکیب مختلف [یعنی ترکیبات مختلف دو نوع لیزر و سه قطر] مجاز نیست ولیکن طول حداکثر کابل به ترکیب مورد استفاده وابسته است. اعدادی که در شکل ۴-۲۳ ارائه شده اند برای بهترین حالت ممکن هستند. رسیدن به طول کابل ۵۰۰۰ متری فقط با لیزر ۱/۳ میکرون و صرفاً با کابل تک موده به قطر ۱۰ میکرون امکان پذیر است ولیکن این انتخاب علیرغم قیمت گران آن بهترین گزینه برای پیاده سازی ستون فقرات شبکه های ناحیه ای (Campus) محسوب می شود.

در الگوی 1000Base-CX، از کابلهای مسی زره دار کوتاه (STP) استفاده شده است. این انتخاب از یک طرف با فیبرنوری با کارآئی بالا و از طرف دیگر با سیم های ارزان قیمت UTP در رقابت است زیرا نه به ارزانی UTP است و نه به کارآمدی فیبر نوری! لذا احتمالاً از آن استقبال نخواهد شد.

آخرین انتخاب، استفاده از چهار زوج سیم Cat 5 است که همزمان با یکدیگر کار می کنند. بدلیل آنکه حجم زیادی از این نوع کابل از قبل نصب شده لذا می توان از آن به عنوان گونه فقیرانه و کم خرج اترنت گیگابیت یاد کرد! اترنت گیگابیت برای کدپنگ سیگنال روی فیبرنوری از روشی جدید استفاده می کند. بهره گیری از روش منچستر با سرعت 1-Gbps، نیاز به تغییر در سطح سیگنال با نرخ معادل 2-G baud/sec خواهد داشت که رسیدن به آن بسیار دشوار بوده و نیاز به پهنای باند بسیار زیادی دارد. در عوض از روشی جدید به نام 8B/10B استفاده شده است. در این روش هر بایت هشت بیتی قبل از ارسال بر روی فیبرنوری به یک الگوی ده بیتی نگاشته می شود؛ به همین دلیل نام آن 8B/10B انتخاب شده است. از آنجایی که کلمه کد ده بیتی خروجی (که به ازای هر بایت ورودی تولید می شود) دارای ۱۰۲۴ حالت مختلف است لذا برای هر یک از ۲۵۶ حالت مختلف ورودی می توان یک کلمه کد، متناسب با شرایط کانال انتخاب کرد. برای انتخاب کلمه کد، دو قاعده زیر به کارگرفته می شود:

۱. هیچ کلمه کدی نباید بیش از ۴ بیت مشابه و پشت سرهم داشته باشد.

۲. هیچ کلمه کدی نباید جمعاً بیش از ۶ بیت صفر یا ۶ بیت یک داشته باشد.

این معیارها باعث خواهد شد که در جریان سیگنال تولید شده در خروجی بقدر کافی لبه (Transition) وجود داشته باشد تا این اطمینان حاصل شود که گیرنده با فرستنده هماهنگ و سنکرون باقی مانده و تعداد صفرها و یکهای ارسالی بر روی فیبر، حتی الامکان مساوی یا نزدیک به هم باشد. مضاف بر این بسیاری از بایتهای ورودی می توانند بیش از یک کلمه کد هم ارز داشته باشند. وقتی که کدکننده، در انتخاب کلمه کد آزادی انتخاب داشته باشد می تواند کلمه کدی را انتخاب کند که برآیند تعداد صفرها و یکهایی که تاکنون ارسال شده اند حتی الامکان معادل باشد. تاکید می شود که بر روی معادل بودن تعداد صفرها و یکها وجود دارد از آن جهت است که مولفه DC سیگنال حتی الامکان پائین بوده و در صورت عبور از مبدلها (Transformers) تغییری در شکل سیگنال ایجاد نشود. اگر چه دانشمندان کامپیوتر تمایلی ندارند که خصوصیات یک ترانسفورمر، نوع کدینگ آنها را تعیین کند ولیکن بهر حال نکته ای است که باید رعایت شود.

در اترنت گیگابیت برای الگوی سیم کشی 1000Base-T، از روش کدینگ متفاوتی بهره گرفته شده است چرا که ارسال پالسهای داده بر روی سیم مسی در زمان یک نانوثانیه بسیار دشوار است. در کدینگ جدید از چهار زوج سیم Cat 5 استفاده شده تا چهار سمبل به صورت همزمان و موازی ارسال شوند. هر سمبول با پنج سطح ولتاژ متفاوت کد می شود. این الگو اجازه می دهد تا یک سمبول بتواند یکی از حالات 00، 01، 10، یا 11 و یک حالت کنترل خاص را کد نماید. بنابراین در هر سیکل سیگنال ساعت، بر روی هر زوج سیم ۲ بیت و بر روی کل سیمها ۸ بیت ارسال می شود. سیگنال ساعت در فرکانس ۱۲۵ مگاهرتز کار می کند که اجازه ارسال یک گیگابیت در هر ثانیه را خواهد داد. دلیل آنکه بجای استفاده از چهار سطح ولتاژ از پنج سطح استفاده شده آنست که برخی از ترکیبات آن برای عملیات کنترلی و فریمینگ باقی بماند.

سرعت 1 Gbps بسیار سریع و بالاست: به عنوان مثال هرگاه یک گیرنده، فقط برای یک میلی ثانیه به کاری دیگر مشغول باشد و نتواند بافر ورودی یکی از خطوط را خالی کند، در این وقفه یک میلی ثانیه ای، ۱۹۵۳ فریم در بافر جمع خواهد شد!! همچنین اگر کامپیوتری در شبکه اترنت گیگابیت، داده های را برای کامپیوتری در شبکه اترنت کلاسیک بفرستد به احتمال بسیار زیاد داده ها در بافر روی هم نوشته شده و از دست خواهد رفت. پیامد این دو پدیده آن بود که در اترنت گیگابیت از کنترل جریان (Flow Control) پشتیبانی شود. (همچنان که در اترنت سریع نیز کنترل جریان وجود دارد ولیکن بروشی متفاوت)

برای عملیات کنترل جریان یکی از طرفین با ارسال یک فریم کنترلی خاص، از طرف مقابل خود می خواهد که برای مدتی ارسال داده را متوقف نماید. فریمهای کنترلی، فریمهای معمولی اترنت هستند که در فیلد Type آنها عدد 0x8808 قرار گرفته است. در این صورت، دو بایت ابتدائی از فیلد داده نوع فرمان را مشخص می کند و بایتهای بعدی به عنوان پارامتر آن فرمان تلقی می شوند (در صورت وجود). برای کنترل جریان، فریم PAUSE به کار می رود و پارامتر آن مدت زمان توقف را (بر مبنای طول حداقل فریم) مشخص می نماید. در اترنت گیگابیت واحد زمان ۵۱۲ نانوثانیه است و فریم PAUSE می تواند ایستگاه را تا ۳۳/۶ میلی ثانیه متوقف کند.

پس از آنکه اترنت گیگابیت استاندارد شد کمیته ۸۰۲ که خسته شده بود می خواست پی کار خود برود ولی IEEE از آنها خواست که کار را بر روی اترنت ده گیگابیت شروع کنند. پس از جستجوی سخت برای حرفی که که بتواند جایگزین z [در 802.3z] شود آنها به این نتیجه رسیدند که از پسوند دو حرفی استفاده کنند!! کار آنها نتیجه داد و استاندارد مربوطه با نام 802.3ae به تایید IEEE رسید. آیا اترنت صد گیگابیت بر ثانیه نیز می تواند محقق شود؟

۹-۳-۴ IEEE 802.2: کنترل منطقی لینک

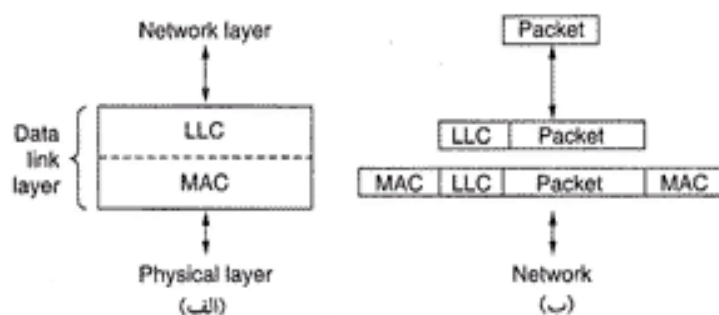
شاید زمان آن فرا رسیده باشد تا اندکی به عقب بازگردیم و برآنچه که در این فصل و ماقبل آن آموختیم، مروری تطبیقی و مقایسه‌ای داشته باشیم. در فصل ۳ آموختیم که چگونه دو ماشین می‌توانند با استفاده از پروتکل‌های پیوند داده، بر روی خطی غیرقابل اعتماد، به صورت مطمئن مبادله اطلاعات کنند. این پروتکلها امکان نظارت برخطا (با استفاده از تصدیق دریافت فریم - Ack) و کنترل جریان (با استفاده از پنجره لغزان) را فراهم آورده‌اند.

برعکس، در این فصل در خصوص مبادله مطمئن داده‌ها سخنی به میان نیاوردیم. تمام گونه‌های اترنت و دیگر پروتکل‌های ۸۰۲ تلاش می‌کنند سرویس دیتاگرام عرضه کنند. [یعنی ارسال داده‌ها بدون تصدیق دریافت آنها -Ack]. در اغلب موارد، این سرویس کافی به نظر می‌رسد. به عنوان مثال برای انتقال بسته‌های IP، به هیچ تضمینی در رسیدن بسته‌ها نیازی نیست. یک بسته IP را می‌توان بسادگی درون فیلد حمل داده از فریم ۸۰۲ قرار داد و آنرا ارسال کرد. اگر فریم از بین برود مهم نیست.

علیرغم این، سیستم‌هایی وجود دارند که به پروتکلی با قابلیت‌های نظارت برخطا و کنترل جریان نیاز دارند. IEEE پروتکلی تعریف کرده که می‌تواند بر روی پروتکل اترنت یا هر پروتکل سری ۸۰۲ قرار گرفته و اجرا شود. این پروتکل که LLC (Logical Link Control) نامیده می‌شود، قادر است تفاوت‌های انواع مختلف شبکه‌های ۸۰۲ را از طریق تعریف یک قالب و واسطه (Interface) واحد و مشترک مخفی کند. این پروتکل بسیار شبیه به پروتکل HDLC است که در فصل ۳ آنرا بررسی کردیم. LLC نیمه بالائی لایه پیوند داده‌ها را تشکیل می‌دهد، در حالیکه زیرلایه MAC، نیمه پائینی این لایه محسوب می‌شود. (شکل ۴-۲۴)

استفاده رایج از LLC بدین ترتیب است: لایه شبکه در ماشین فرستنده، بسته‌ای را یکمک توابع پایه و بنیادی لایه LLC، بدان تحویل می‌دهد. زیرلایه LLC سرآیند لازم را به بسته می‌افزاید؛ این سرآیند شامل شماره ترتیب (Seq No) و شماره تصدیق (Ack No) است. بسته حاصل، درون فیلد حمل داده از فریم ۸۰۲ قرار گرفته و ارسال می‌شود. در گیرنده نیز عکس این فرآیند انجام می‌شود.

LLC از سه رده خدمات حمایت می‌کند: (۱) خدمات ارسال نامطمئن دیتاگرام، (۲) خدمات دیتاگرام بانصدیق وصول و (۳) خدمات ارسال اتصال‌گرای مطمئن. سرآیند LLC سه فیلد را در برمی‌گیرد: «نقطه دسترسی در مقصد»، «نقطه دسترسی در مبدا» و «فیلد کنترل». نقطه دسترسی مشخص می‌کند که این فریم از چه پروسه‌ای آمده و به کدام پروسه باید تحویل شود که در حقیقت نقش همان فیلد Type در فریم DIX را ایفاء می‌کند. فیلد کنترل نیز در برگیرنده شماره ترتیب و شماره تصدیق است که شباهت فراوانی با ساختار HDLC دارد (شکل ۳-۲۴)، ولی کاملاً با آن یکسان نیست. از این فیلدها زمانی استفاده می‌شود که در سطح لایه پیوند داده‌ها به یک اتصال مطمئن (Reliable Connection) نیاز باشد که در این حالت شبیه به روشی عمل می‌شود که در فصل سوم تشریح شد. برای شبکه اترنت، صرف تلاش در تحویل بسته IP کفایت می‌کند و به هیچ پیغام اعلام وصول فریم در سطح LLC نیازی نیست.



شکل ۴-۲۴. (الف) موقعیت LLC در پشته پروتکلی (ب) قالب‌های پروتکل.

۴-۳-۱۰ نگاهی به گذشته اینترنت

اینترنت برای حدود ۲۰ سال در صحنه بوده و تقریباً هیچ رقیب جدی نداشته است و به نظر می‌رسد در سالهای آتی نیز همچنان یک‌تاز باشد. تعداد بسیار کمی سیستم عامل، زبان برنامه نویسی یا معماری CPU وجود داشته که بتواند برای دو دهه متوالی بر قله افتخار بایستد و در حال ورود به دهه سوم افتخار خود باشد. روشن است که اینترنت حرفه‌ای برای گفتن داشته که بدین گونه دوام آورده است؛ اینها چه بوده‌اند؟

شاید دلیل اصلی بقای اینترنت «سادگی» و «قابلیت اعتماد» آن بوده است. در عمل، معیار «سادگی» به «قابلیت اعتماد»، «ارزانی»، «سهولت نصب و نگهداری» تعبیر می‌شود. وقتی در اینترنت انشعابات تزیینی (Vampire Tap) با کانکتورهای BNC عوض شد خرابیها به شدت کاهش یافت. بطور معمول، عموم افراد حاضر نیستند از ابزارهایی که بخوبی کار می‌کنند رو بگردانند و این پافشاری از آنجاست که بر همه روشن شده بسیاری از محصولات بنجل در صنعت کامپیوتر باهیا هو می‌آیند ولی بسیار ضعیف عمل می‌کنند، حتی گاهی محصولاتی که با عنوان «ارتقاء» (Upgrade) معرفی می‌شوند بسیار بدتر و ناسازگارتر از وقتی عمل می‌کنند که بطور کامل عوض شوند! ولیکن نسخه‌های ارتقایافته اینترنت، اعتماد عمومی را جلب کرد.

«سادگی» [در مورد اینترنت] به ارزان بودن نیز تعبیر می‌شود. اینترنت نازک و سیم‌های زوجی نسبتاً کم بها و ارزان هستند. کارت‌های واسطه اینترنت نیز بسیار ارزانند. فقط در دورانی که هاب و سوئیچ معرفی شد، به مقداری سرمایه‌گذاری نیاز داشت ولیکن در زمان عرضه آنها، شبکه اینترنت بقدر کافی جا افتاده بود.

اینترنت از لحاظ نصب و نگهداری ساده است. هیچ نرم‌افزار اضافی نباید نصب شود (مگر نرم‌افزارهای بسیار کوچک راه‌انداز آنها) و به هیچ جدول یا تنظیمات پیگیربندی خاصی نیاز ندارد که مدیریت آن (یا هرگونه اشتباه در تنظیم آن) کار را دشوار کند. در ضمن اضافه کردن یک ماشین جدید به شبکه به سادگی وصل آن به هاب یا سوئیچ است و کار چندانی ندارد.

امتیاز دیگر اینترنت آن بود که بسادگی با TCP/IP که پروتکل غالب دنیاست، کار می‌کند. IP پروتکلی بدون اتصال (Connectionless) و دقیقاً متناسب با اینترنت است که آن هم بدون اتصال عمل می‌کند. در مقابل، IP سازگاری بسیار کمی با ATM (که اتصال‌گراست) داشت و این عدم سازگاری به موقعیت ATM لطمه بسیار فراوانی زد.

در آخر آنکه اینترنت ظرفیت پیشرفت و بهبود فراوانی از خود نشان داد. سرعت آن چند ده برابر شد، هاب و سوئیچ عرضه گردید ولی با تمام این تغییرات نیازی به تغییر در نرم‌افزار نبود. تصور کنید وقتی یک عرضه کننده محصولات شبکه، تجهیزات مفصلی را ارائه کرده و می‌گوید: «این شبکه جدید و عالی را برای شما تدارک دیده‌ایم. برای استفاده از آن باید زحمت بکشید و سخت‌افزارهای قبلی خود را دور ریخته و تمام نرم‌افزارهای خود را از نو بنویسید!! او در فروش شبکه خود قطعاً به مشکل برخورد!!! اینترنت چنین مشکلی نداشت. شبکه‌هایی مثل FDDI, Fibre Channel و ATM در زمان عرضه بسیار سریعتر از اینترنت بودند ولیکن هیچکدام از آنها با اینترنت سازگار نبودند، بسیار پیچیده‌تر و مدیریت آنها دشوارتر از اینترنت بود. سرانجام وقتی سرعت اینترنت بهبود یافت این شبکه‌ها دیگر هیچ مزیتی نداشتند و سریعاً از میان رفتند؛ البته بجز ATM که آن هم در هسته سیستمهای تلفنی به کار گرفته شده بود.

۴-۴ شبکه‌های محلی بی‌سیم

اگر چه اینترنت در سطح گسترده‌ای رایج است ولی رقیب جدیدی برای آن در حال ظهور است. شبکه‌های بی‌سیم

در حال رواج هستند و بطور فزاینده‌ای در دفاتر اداری، فرودگاه‌ها و دیگر مکانهای عمومی به کار گرفته می‌شوند. شبکه‌های بی‌سیم به نحوی که در شکل ۱-۳۵ دیدیم به دو روش پیکربندی می‌شوند: «در کنار یک ایستگاه ثابت» و «بدون ایستگاه ثابت». استاندارد 802.11 هر دوی این پیکربندیها را مد نظر قرار داده و برای هر دو، تدارک لازم را دیده است.

در بخش ۱-۵-۴ پیش‌زمینه‌ای از 802.11 ارائه نمودیم. حال زمان آن رسیده تا نگاهی دقیقتر به این تکنولوژی بیندازیم. در بخشهای آتی نگاهی به پشته پروتکلی، تکنیکهای ارسال رادیویی در لایه فیزیکی، پروتکل زیرلایه MAC، ساختار فریم و خدمات ارائه شده در این شبکه خواهیم انداخت. برای کسب آگاهی بیشتر در خصوص 802.11 به مراجع زیر مراجعه نمایید:

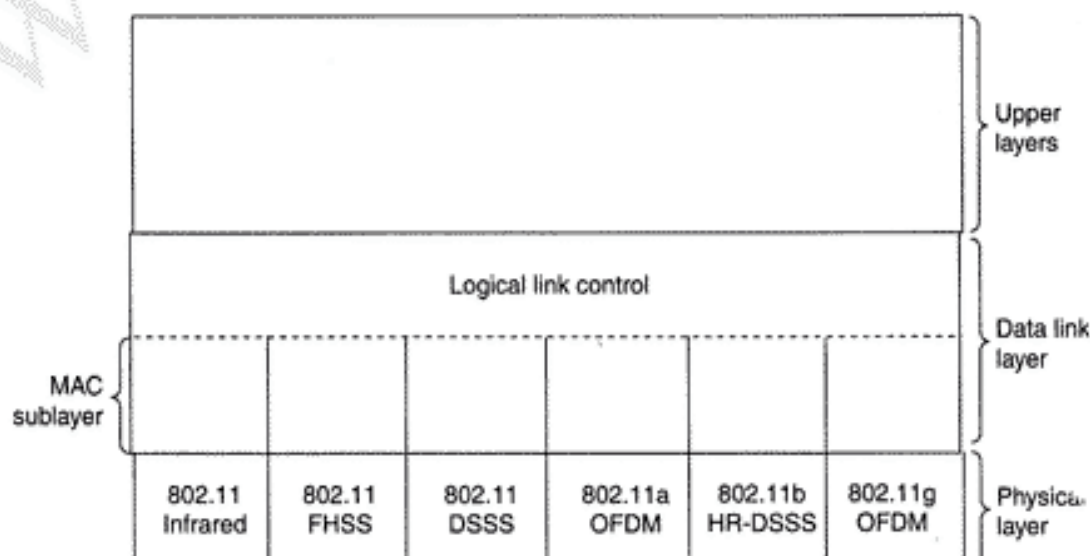
(Crow et al., 1997; Geier, 2002; Heegard et al., 2001; Kapp, 2002; O'Hara & Petric, 1999; and Severance, 1999)

برای شنیدن حقیقت درست و مسلم این شبکه، باید مستقیماً به استاندارد IEEE 802.11 مراجعه نمایید.

۱-۴-۴ پشته پروتکلی 802.11

پروتکلهایی که در استانداردهای سری ۸۰۲ و از جمله اترنت به کار گرفته شده مشترکات ساختاری فراوان دارند. شمای کلی پشته پروتکلی 802.11، در شکل ۴-۲۵ نمایش داده شده است. لایه فیزیکی به خوبی هم تراز با لایه فیزیکی از مدل OSI است در حالیکه لایه پیوند داده از دو لایه مستقل تشکیل شده است. در 802.11 زیرلایه MAC (زیرلایه کنترل دسترسی به کانال)، چگونگی دسترسی و تخصیص کانال، یعنی ایستگاهی را که باید در ادامه ارسال داشته باشد، مشخص می‌کند. بر روی آن، زیرلایه LLC قرار می‌گیرد که وظیفه اصلی آن مخفی کردن تفاوتهای موجود در گونه‌های مختلف ۸۰۲ است، بگونه‌ای که این تفاوتها برای لایه شبکه [لایه سوم در مدل OSI] مخفی و غیر قابل تشخیص باشد. در همین فصل وقتی اترنت را بررسی می‌کردیم، LLC را نیز مطالعه نمودیم و نیازی به تکرار آنها نیست.

در سال ۱۹۹۷ استاندارد 802.11 سه تکنیک مختلف انتقال رادیویی، برای به کارگیری در لایه فیزیکی معرفی کرد. مثلاً روش مبتنی بر امواج مادون قرمز، بسیار شبیه به روشی است که در کنترل از راه دور تلویزیونها به کار رفته



شکل ۴-۲۵. بخشی از پشته پروتکلی 802.11.

است. در روش دیگر از امواج رادیویی برد کوتاه و از تکنیک هائی به نام FHSS و DSSS بهره گرفته شده است. هر دوی این روشها از محدوده ای در طیف فرکانس استفاده می کنند که نیازی به اخذ مجوز از دولت ندارد (باند 2.4 GHz). به عنوان مثال دریاکن های کنترل از راه دور نیز از همین باند فرکانسی استفاده می کنند لذا کامپیوتر کیفی شما ممکن است در حین استفاده از کانال، خودش را رقیب درب گاراژ شما ببیند!!! تلفن های بی سیم و اجاق های مایکروویو نیز از همین باند فرکانسی بهره گرفته اند.

تکنیک های ارسال رادیویی با نرخ ۱ تا ۲ مگابیت در ثانیه و با توان بسیار کمی عمل می کنند تا تداخل رادیویی این ابزارها با یکدیگر حداقل باشد. در سال ۱۹۹۹ دو تکنیک جدید معرفی شد تا پهنای باند (نرخ ارسال) آن افزایش یابد. این دو تکنیک جدید OFDM و HR-DSSS نامیده شده اند و به ترتیب با سرعت های ۵۴ و ۱۱ مگابیت بر ثانیه عمل می کنند. در سال ۲۰۰۱ گونه دومی از مدولاسیون OFDM ولی در باند فرکانسی متفاوت نسبت به OFDM اولیه، معرفی شد. در ادامه بطور مختصر آنها را بررسی خواهیم کرد. از دیدگاه فنی این تکنیکها به لایه فیزیکی تعلق دارند و باید در فصل دوم بررسی می شدند ولی از آنجایی که این تکنیکها به LAN و خصوصاً زیرلایه 802.11 MAC وابسته هستند، در این بخش بدانها پرداخته ایم.

۲-۴-۴ لایه فیزیکی در 802.11

تمام پنج تکنیک انتقال رادیویی، این امکان را فراهم کرده اند که یک فریم MAC از ایستگاهی به ایستگاه دیگر منتقل شود. تفاوت های آنها در تکنولوژی به کاررفته و سرعت قابل حصول آنهاست. پرداختن به جزئیات این روشها از حوصله این کتاب خارج است ولیکن چند کلمه صحبت در مورد آنها و بخصوص معرفی کلمات کلیدی آن می تواند به علاقمندان کمک کند تا بتوانند برای کسب آگاهی بیشتر، در اینترنت یا مراجع دیگر جستجو کنند. در گزینه «امواج مادون قرمز» از امواج بخشی (Diffused) با طول موج ۸۵۰/۰ یا ۹۵۰ میکرون بهره گرفته شده است. در این روش سرعت های ۱ و ۲ مگابیت در ثانیه مجاز می باشد. در نرخ 1 Mbps از روش کدینگ خاصی به نام Gray Code استفاده شده که در آن گروه های ۴ بیتی به یک کلمه کد ۱۶ بیتی تبدیل می شوند، به نحوی که در این کلمه ۱۶ بیتی تنها یک بیت ۱ و پانزده بیت ۰ وجود دارد. این کد دارای این ویژگی اساسی است که خطائی کوچک در سنکرونیزاسیون زمان، تنها یک بیت خطا در خروجی ایجاد خواهد کرد. در سرعت 2 Mbps، دو بیت اخذ و یک کد چهار بیتی تولید می شود که در آن تنها یک بیت ۱ وجود دارد. (یعنی هر دو بیت به یکی از چهار حالت 0001, 0010, 0100, 1000 نگاشته می شود). سیگنال های مادون قرمز نمی توانند در موانعی مثل دیوار نفوذ کنند لذا سلول هایی که در اتاق های مختلف ایجاد می شوند کاملاً از هم جدا و تفکیک شده هستند. علیرغم این، بدلیل نرخ ارسال پائین (و این حقیقت که نور خورشید، امواج مادون قرمز را در خود غرق و محو می کند) از این گزینه استقبال چندانی نشد.

در FHSS (Frequency Hopping Spread Spectrum) از ۷۹ کانال مستقل استفاده شده که هر یک از این کانالها 1 MHz پهنای باند دارند و از پائین ترین فرکانس باند 2.4 GHz ISM شروع می شوند. برای مشخص کردن دنباله فرکانس هائی که باید بدانها پرش شود از یک مولد اعداد شبه تصادفی استفاده شده است. مادامیکه تمام ایستگاه ها در الگوریتم مولد اعداد از نقطه شروع (Seed) یکسانی استفاده کنند و از لحاظ زمانی با هم سنکرون باشند همگی بطور همزمان به فرکانس های یکسانی پرش خواهند کرد. مدت زمانی که ایستگاه ها در یک فرکانس خاص باقی می مانند، اصطلاحاً dwell time نامیده می شود و پارامتری قابل تنظیم است ولیکن باید کمتر از ۴۰۰ میلی ثانیه باشد. استفاده تصادفی از باندهای فرکانسی، روش مناسبی برای تخصیص کانال بروشی غیر معمول در باند ISM است. این ویژگی کم و بیش به امنیت اطلاعات کمک خواهد کرد چرا که اگر یک اختلالگر ترتیب پرش های فرکانس یا پارامتر Dwell time را نداند، نمی تواند اطلاعات کانال را استراق سمع کند. در فواصل دور،

محوشدگی سیگنال بدلیل «چندمسیره شدن سیگنال» (Multipath Fading) اشکال عمده‌ای به حساب می‌آید و خوشبختانه FHSS در مقابله با این مشکل، موفق عمل می‌کند. همچنین این روش نسبت به تداخل رادیویی، نسبتاً حساس نیست و برای ایجاد لینک بین ساختمانها بسیار مناسب خواهد بود. بزرگترین اشکال این روش پهنای باند کم آنست. (1 Mbps)

سومین روش مدولاسیون یعنی DSSS (Direct Sequence Spread Spectrum) نیز به یکی از نرخ‌های ۱ یا ۲ مگابیت بر ثانیه محدود شده است. تکنیک به کار رفته در DSSS تا حدودی مشابه به سیستم CDMA است که در بخش ۲.۶.۲ بررسی شد ولیکن از برخی جهات با آن تفاوت‌هایی دارد. هر بیت در قالب یازده Chips ارسال می‌شود که به دنباله بارکر (Barker Sequence) مشهور است. در این روش از مدولاسیون تغییر فاز (Phase Shift) با نرخ تغییر 1 Mbaud استفاده شده که وقتی در نرخ 1 Mbps عمل می‌کند در هر تغییر فاز یک بیت ولی وقتی در نرخ 2 Mbps عمل می‌کند در هر تغییر فاز، دو بیت منتقل می‌گردد. برای سالیهای متمادی سازمان [FCC سازمان تخصیص فرکانس] فقط اجازه می‌داد که تجهیزات مخابرات بی‌سیم در ایالات متحده، صرفاً از باند ISM و طیف گسترده (Spread Spectrum) استفاده کنند ولی ظهور تکنولوژیهای جدید، به لغو این قانون در سال ۲۰۰۲ انجامید.

اولین شبکه محلی بی‌سیم پرسرعت یعنی 802.11a، با بهره‌گیری از مدولاسیون OFDM^۱ در باند فرکانسی 5-GHz عمل می‌کرد تا به سرعت 54 Mbps دست یابد. اگر بخواهیم با استعارات FDM سخن بگوئیم، در OFDM از ۵۲ زیرکانال فرکانسی استفاده شده که ۴۸ تا از آنها برای داده و ۴ تا برای سنکرونیزاسیون کاربرد دارد و بی‌شباهت به ADSL نیست. از آنجایی که ارسال، بطور همزمان بر روی فرکانسهای متفاوتی انجام می‌شود لذا این روش گونه‌ای از روشهای مبتنی بر «طیف گسترده» محسوب می‌شود ولیکن با روشهای CDMA یا FHSS کاملاً متفاوت است. تقسیم سیگنال به تعداد بسیار زیادی باند باریک در مقایسه با استفاده از یک باند عریض و واحد، مزایای بسیار مهمی در بردارد که از جمله می‌توان به ایمنی بیشتر در مقابل تداخل امواج باند باریک و امکان استفاده از باندهای غیر مجاور (noncontiguous band) اشاره کرد. در این روش از سیستم کدینگ پیچیده‌ای مبتنی بر مدولاسیون تغییر فاز برای سرعت زیر 18 Mbps و مدولاسیون QAM برای سرعت‌های بالاتر استفاده شده است. در سرعت 54 Mbps، ۲۱۶ بیت داده به سمبول‌های ۲۸۸ بیتی کد می‌شود. بخشی از انگیزه‌های خلق OFDM سازگاری آن با سیستم اروپائی HiperLAN/2 بوده است. (Doufexi et al., 2002) این روش دارای کارائی بسیار بالائی در استفاده از طیف فرکانسی (برحسب bits/HZ) بوده و ایمنی خوبی در مقابل پدیده «محوشدگی ناشی از مسیرهای چندگانه» دارد.

نهایتاً به HR-DSSS (High Rate Direct Spread Spectrum) می‌رسیم که روشی دیگر مبتنی بر تکنیک طیف گسترده است و با بهره‌گیری 11 million chips/sec در باند 2.4 GHz، به نرخ ارسال یازده مگابیت در ثانیه رسیده است. این روش 802.11b نامیده شده ولیکن دنباله روی 802.11a نبوده است. در این روش از نرخهای ۱، ۲، ۵/۵ و ۱۱ مگابیت در ثانیه حمایت می‌شود. دو نرخ ارسال پائین [یعنی ۱ و ۲ مگابیت بر ثانیه] به ترتیب از سیگنالی با نرخ تغییر 1 Mbaud/sec بهره می‌گیرد (یعنی در هر تغییر فاز، ۱ یا ۲ بیت کد می‌شود). در HR-DSSS به منظور سازگاری با DSSS، از روش مدولاسیون تغییر فاز بهره گرفته شده است. برای نرخ ارسال سریعتر (۵/۵ و ۱۱ مگابیت بر ثانیه) از سیگنالی با نرخ تغییر 1.375 Mbaud/sec استفاده شده و در هر تغییر به ترتیب ۴ یا ۸ بیت کد می‌شود؛ کدها از نوع Walsh/Hadamard هستند. نرخ ارسال به صورت پویا و در خلال عملیات شبکه تعیین

۱. Orthogonal Frequency Division Multiplexing

می‌شود تا سرعت بهینه بر اساس شرایط فعلی حاکم بر شبکه (شامل نویز محیط و بار) تنظیم گردد. در عمل، سرعت شبکه 802.11b تقریباً همیشه 11 Mbps است. اگر چه سرعت 802.11b از سرعت 802.11a کمتر می‌باشد ولیکن بُرد این شبکه حدوداً هفت برابر بیشتر است که این ویژگی در بسیاری از محیطها اهمیت بسزایی دارد.

نسخه بهبود یافته 802.11b یعنی 802.11g، در نوامبر سال ۲۰۰۱ (پس از کشمکش فراوان بر سر آنکه از کدام تکنولوژی استفاده شود) به تائید IEEE رسید. این استاندارد از مدولاسیون OFDM (به کار رفته در 802.11a) بهره می‌گیرد ولیکن مثل 802.11b در باند 2.4 GHz عمل می‌کند. از نظر تئوری این سیستم می‌تواند در سرعت 54 Mbps عمل کند، ولیکن هنوز روشن نیست که آیا این سرعت در عمل نیز محقق خواهد شد یا خیر. بدین ترتیب کمیته 802.11 سه شبکه محلی پرسرعت بی‌سیم معرفی کرده است. 802.11a، 802.11b و 802.11g (به سه شبکه کدتر فعلاً کاری نداریم) شاید این سوال درست به ذهن شما خطور کند که آیا تعریف سه استاندارد متفاوت کار درستی است؟ شاید عدد ۳، عدد طلایی شانس باشد!

۳-۴-۴ پروتکل زیرلایه MAC در 802.11

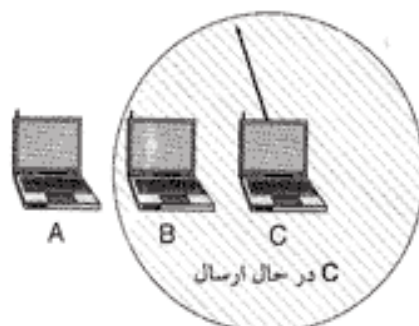
حال اجازه بدهید از فضای مهندسی برق به سرزمین مهندسی کامپیوتر برگردیم. پروتکل زیرلایه MAC در 802.11 کاملاً با اترنت تفاوت دارد زیرا شرایط حاکم بر محیطهای بی‌سیم در مقایسه با سیستمهای سیم‌دار، دارای پیچیدگیهای ذاتی است. در اترنت یک ایستگاه منتظر می‌ماند تا کانال آزاد شود؛ سپس ارسال خود را شروع می‌کند. اگر در خلال ارسال ۶۴ بایت اول فریم، هیچ نویز شدیدی برنگشت، می‌توان اعتقاد داشت که فریم به درستی تحویل مقصد شده است. در شبکه بی‌سیم چنین وضعیتی حاکم نیست.

برای شروع باز هم یادآور می‌شویم که مشکل ایستگاه مخفی (که قبلاً تشریح و مجدداً در شکل ۴-۲۶ به تصویر کشیده شده است) ایجاد اشکال خواهد کرد. از آنجایی که تمام ایستگاهها در برد رادیویی یکدیگر نیستند فلذا ارسال سیگنال در بخشی از یک سلول، ممکن است در ناحیه دیگری از همان سلول قابل دریافت و شنود نباشد. در این مثال ایستگاه C در حال ارسال به ایستگاه B است ولی اگر A کانال را بررسی و شنود کند هیچ چیزی نمی‌شنود و به غلط نتیجه می‌گیرد که باید ارسال برای B را آغاز کند. [شکل ۴-۲۶-الف]

همچنین عکس این مشکل نیز وجود دارد. در شکل ۴-۲۶-ب ایستگاه B می‌خواهد فریمی را برای C بفرستد و به همین دلیل به کانال گوش می‌دهد. وقتی سیگنال در حال انتقال را شنود می‌کند به غلط نتیجه می‌گیرد که نباید برای ایستگاه C چیزی بفرستد ولی علیرغم آنکه A در حال ارسال برای D است (D در شکل نشان داده نشده) ایستگاه B می‌تواند برای C ارسال داشته باشد. مضاف براین، ارتباط بی‌سیم اغلب ماهیت «دوطرفه غیرهمزمان» (Half Duplex) دارد بدین معنا که ایستگاهها نمی‌توانند در حین ارسال و بطور همزمان، کانال را برای آگاهی از وضعیت تصادم بروی همان باند فرکانسی شنود کنند. [این کار در کانالهای سیمی به راحتی امکان‌پذیر است] در نتیجه 802.11، برخلاف اترنت از CSMA/CD استفاده نمی‌کند.

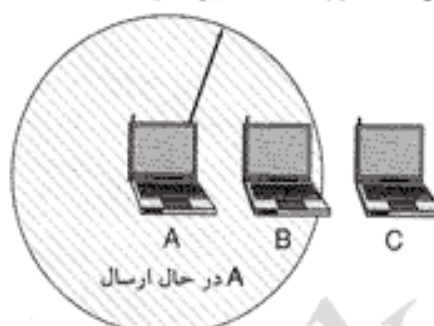
برای رفع این مشکلات، استاندارد 802.11 از دو روش عملکرد پشتیبانی می‌کند: در اولین روش که DCF نامیده می‌شود (Distributed Coordination Function) هیچ‌گونه کنترلی مرکزی وجود ندارد (و از این دیدگاه مشابه با اترنت است). در روش دیگر که PCF (Point Coordination Function) نامیده می‌شود، برای کنترل و نظارت بر کلیه فعالیتهای درون هر سلول، از یک ایستگاه ثابت استفاده شده است. در پیاده‌سازی استاندارد 802.11، باید از DCF پشتیبانی شود در حالیکه حمایت از PCF اختیاری است. به ترتیب این دو روش را تشریح خواهیم کرد.

A تعامل دارد برای B ارسال داشته باشد ولی قادر به شنود آنکه B مشغول است نمی باشد.



(الف)

B تعامل دارد برای C ارسال داشته باشد ولی به اشتباه فکر می کند ارسال که او با شکست روبرو خواهد شد...



(ب)

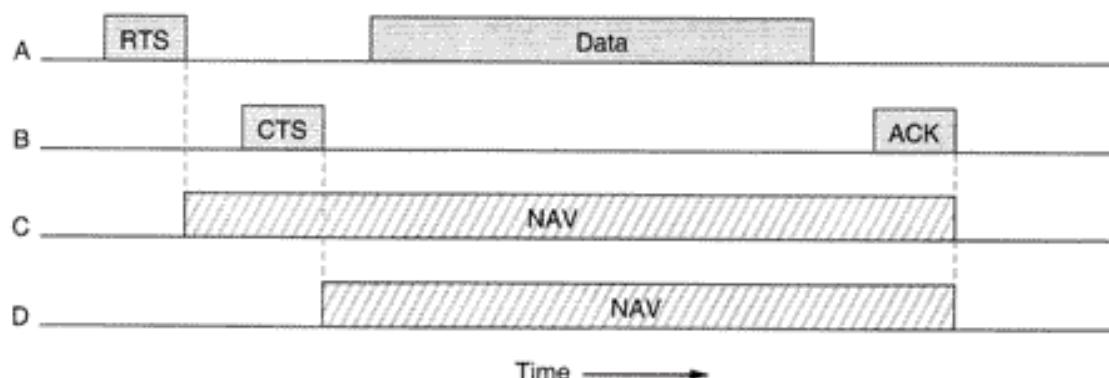
شکل ۴-۲۶. (الف) مشکل ایستگاه مخفی (ب) مشکل ایستگاه آشکار.

وقتی از حالت DCF استفاده می شود، 802.11 از پروتکلی به نام CSMA/CA^۱ بهره می گیرد. در این پروتکل هم کانال فیزیکی و هم کانال مجازی شنود می شوند. CSMA/CA از دو عملکرد متفاوت پشتیبانی می کند: در روش اول وقتی یک ایستگاه می خواهد فریمی را ارسال کند، ابتدا به شنود می پردازد و اگر کانال آزاد بود ارسال خود را آغاز می کند. در حین ارسال فریم، کانال شنود نمی شود و کل فریم منتقل می گردد، در حالیکه ممکن است این فریم، بدلیل تداخل رادیویی در گیرنده از بین برود. برعکس، اگر کانال اشغال باشد فرستنده ارسال خود را تا زمان آزاد شدن کانال به تعویق انداخته و سپس شروع می کند. در صورت بروز تصادم، ایستگاه های تصادم کننده به اندازه یک زمان تصادفی (که بر اساس الگوریتم عقب گرد نمایی تعیین می شود) منتظر مانده و از نو تلاش می کنند. [تا اینجا همه چیز شبیه به پروتکل CSMA/CD است.]

روش دیگر به کار رفته در CSMA/CA مبتنی بر MACAW است که از روش «شنود کانال مجازی» بهره می گیرد. در مثال شکل ۴-۲۷، ایستگاه A می خواهد فریمی را برای B بفرستد. C، ایستگاهی در برد ایستگاه A است (شاید C هم در برد ایستگاه B باشد ولی مهم نیست). D، ایستگاهی در برد B است ولی در برد A قرار ندارد. پروتکل زمانی آغاز می شود که A تصمیم به ارسال داده برای B می گیرد. او کارش را با ارسال یک فریم کوتاه RTS برای B آغاز و تقاضای مجوز ارسال فریم می نماید. هرگاه B این تقاضا را دریافت کند (احتمالاً) تصمیم به صدور مجوز می گیرد؛ در این حالت فریم CTS را برمی گرداند. پس از دریافت فریم CTS، ایستگاه A ارسال فریم خود را شروع کرده و یک زمان سنج خاص به نام ACK-Timer را روشن می کند. پس از دریافت فریم داده، ایستگاه B با ارسال فریم ACK به مبادله داده خاتمه می دهد. اگر زمان سنج، قبل از آنکه فریم ACK باز گردد، منقضی شود [یعنی فریم ACK در زمان معقولی برنگردد] کل این فرآیند باید از نو اجرا شود.

حال بیایید از دیدگاه ایستگاه های C و D به این فرآیند مبادله فریم، نگاه کنیم. C ایستگاهی است که در برد A است و طبقاً RTS را دریافت می کند. اگر اینگونه باشد متوجه خواهد شد که شخص دیگری بزودی ارسال خود را آغاز خواهد کرد، فلذا برای احتیاط کامل، تا تکمیل عملیات مبادله داده، از ارسال هر چیزی اجتناب می کند. C می تواند از طریق اطلاعاتی که از فریم RTS بدست می آید، کل زمان ارسال را تخمین بزند (با احتساب زمان برگشت فریم ACK)، لذا برای خودش فرض می کند که در خلال زمانی که در شکل ۴-۲۷ با عنوان NAV

^۱CSMA with Collision Avoidance



شکل ۴-۲۷. کاربرد کانال مجازی در روش CSMA/CA.

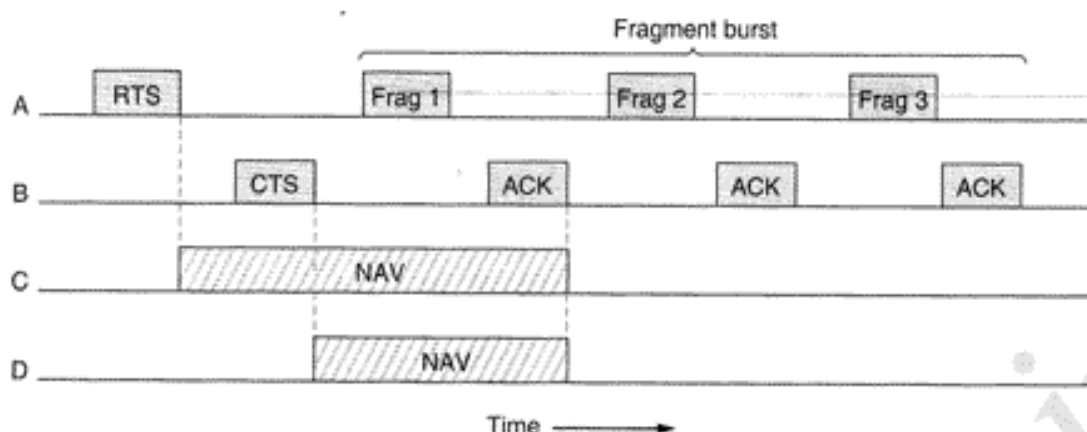
(Network Allocation Vector) مشخص شده، کانال مجازی مشغول است. ایستگاه D فریم RTS را نمی شنود ولیکن CTS را خواهد شنید لذا او هم برای خودش به اندازه زمان NAV کانال را مشغول فرض می کند. دقت کنید که سیگنال NAV به هیچ وجه ارسال نمی شود بلکه فقط یک پادداشت داخلی است که ایستگاه را در مدت زمان معینی ساکت نگه می دارد.

برخلاف شبکه های سیمی، شبکه های بی سیم غیر قابل اعتماد و نویزی هستند چرا که دستگاه های مختلف مثل اجاق های مایکروویو (که آنها نیز در باند ISM کار می کنند) نویز قوی و مخرب تولید می نمایند. در نتیجه، احتمال انتقال موفق فریم با افزایش طول فریم کاهش خواهد یافت. هرگاه احتمال خرابی یک بیت p باشد احتمال آنکه یک فریم n بیتی، کامل و سالم دریافت شود $(1-p)^n$ خواهد بود. برای مثال اگر $p=10^{-4}$ باشد احتمال سالم رسیدن یک فریم کامل اترنت (۱۵۱۴ بیتی) کمتر از ۳۰ درصد است. برای $p=10^{-5}$ ، از هر ۹ فریم یکی خراب خواهد شد. برای $p=10^{-6}$ حدود یک درصد از کل فریمها آسیب خواهد دید که تقریباً معادل یک دوجین فریم در ثانیه خواهد بود. بطور خلاصه اگر یک فریم بزرگ باشد شانس کمتری در سالم رسیدن آن به مقصد وجود دارد و احتمالاً باید مجدداً ارسال شود.

برای کاهش مشکل کانالهای نویزی، 802.11 اجازه داده که هر فریم به قطعات کوچکتری تقسیم شده و هر کدام کد کشف خطای خود را داشته باشند. قطعات بطور مجزا شماره گذاری شده و دریافت آن به روش «توقف و انتظار» (Stop & Wait) تائید می شود. (به عبارت دیگر فرستنده قطعه شماره $k+1$ را نخواهد فرستاد مگر آنکه پیام ACK قطعه k -مبنی بر دریافت صحیح آن- دریافت شود). پس از آنکه به کمک RTS و CTS، کانال در اختیار ایستگاهی قرار گرفت، طبق شکل ۴-۲۸، آن ایستگاه می تواند متوالیاً چندین قطعه مستقل، ارسال کند. دنباله قطعات متوالی، اصطلاحاً Fragment Burst نامیده می شود.

عملیات قطعه سازی فریمها، کارائی مفید شبکه را افزایش خواهد داد چرا که به جای ارسال مجدد کل فریم فقط قطعات کوچکی که در اثر خطای کانال خراب شده اند از نو ارسال خواهند شد. طول هر قطعه به صورت قطعی و ثابت در استاندارد تعیین نشده است بلکه جزو پارامترهای قابل تنظیم هر سلول محسوب و توسط ایستگاه ثابت (Base Station) تنظیم می شود. مکانیزم NAV ایستگاهها را آنقدر ساکت نگاه می دارد تا زمانیکه دریافت کل فریم تایید شود، ولیکن برای آنکه ایستگاهها آنقدر تامل کنند تا دنباله کل قطعات فریم (Fragment Burst) بدون تداخل ارسال شود، مکانیزم دیگری که در زیر تشریح شده، بکار می رود.

تمام توضیحات فوق در حالت DCF از استاندارد 802.11 قابل اعمال و صائب است. یادآوری می کنیم که در حالت DCF هیچ کنترل و نظارت مرکزی وجود ندارد و تمام ایستگاهها مشابه با آنچه که در اترنت اتفاق می افتد برای بدست آوردن کانال رادیونی رقابت می کنند. در حالت دیگر یعنی PCF، یک ایستگاه ثابت یکی یکی به



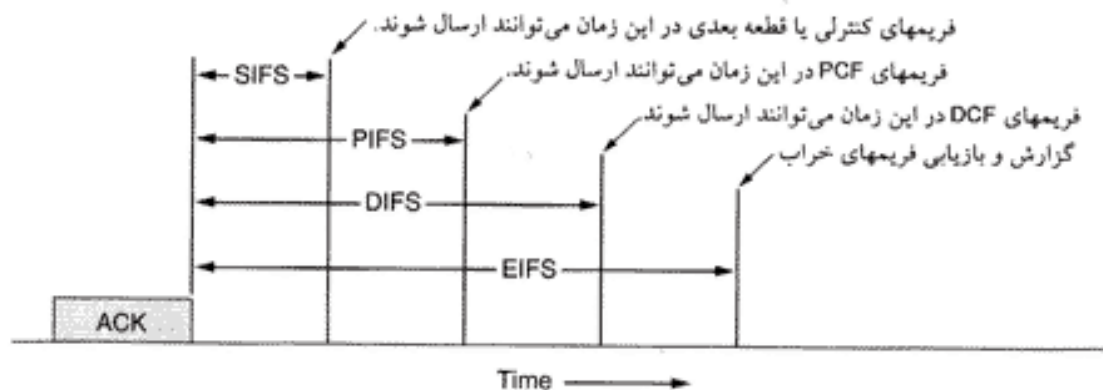
شکل ۲۸۴. ارسال انفجاری چند قطعه.

ایستگاه‌ها سرکشی کرده و از آنها سوال می‌کند که آیا فریمی جهت ارسال دارند یا خیر. از آنجایی که در حالت PCF بر تمام تقاضاهای ارسال فریم به صورت مرکزی نظارت می‌شود لذا هیچگونه تصادم اتفاق نخواهد افتاد. استاندارد 802.11، مکانیزم سرکشی به ایستگاه‌ها را تعیین کرده است ولیکن دفعات سرکشی، ترتیب سرکشی یا حتی تصمیم‌گیری در خصوص آنکه آیا همه ایستگاه‌ها باید به یک اندازه از شبکه سرویس بگیرند، در استاندارد تعریف و تعیین نشده است.

مکانیزم اصلی در سرکشی به ایستگاه بدین نحو است که یک ایستگاه فریم خاصی به نام Beacon Frame (فریم فانوس دریائی) را بطور متناوب در فضای پیرامون خود منتشر می‌کند. (ده تا صد بار در هر ثانیه) «فریم فانوس دریائی» شامل پارامترهای مختلف سیستم مثل: «ترتیب پرش فرکانسی» (Hopping Sequence) و پارامتر dwell time (برای مدولاسیون FHSS) و پارامتر سنکرون سازی سیگنال ساعت و نظایر آن، می‌باشد. همچنین توسط این فریم از ایستگاه‌های جدید دعوت می‌شود تا به منظور سرکشی شدن ثبت‌نام کنند. پس از آنکه ایستگاهی برای دریافت خدمات سرکشی با نرخ معین، ثبت‌نام کرد، این تضمین را دارد که بخش معینی از پهنای باند شبکه به او اختصاص داده خواهد شد و بدین ترتیب آن ایستگاه می‌تواند «کیفیت خدمات» (Quality of Service) خود را تضمین کند. [برای مطالعه در خصوص کیفیت خدمات به بخش ۱-۳-۳ و فصل ۵ مراجعه کنید.]

طول عمر باتری یکی از مسائل مهم در ابزارهای همراه و بی‌سیم بوده و هست؛ به همین دلیل در استاندارد 802.11 به مسئله مدیریت توان مصرفی، توجه ویژه‌ای شده است. خصوصاً در حالت PCF ایستگاه ثابت می‌تواند ایستگاه همراه را به «حالت استراحت» (Sleep State) ببرد؛ تا زمانی که بطور مشخص ایستگاه ثابت یا کاربری دیگر آن را از این حالت بیرون آورده و بتواند فعالیت عادی خود را از سر بگیرد. مجبور کردن یک ایستگاه به استراحت، بدین معناست که ایستگاه ثابت مسئولیت دارد تمام فریم‌هایی را که برای ایستگاه غیرفعال (در حال استراحت) ارسال می‌شود، دریافت و بافر کند. بعداً این فریم‌ها به صورت یکجا تحویل خواهد شد.

در درون یک سلول می‌توان بطور همزمان هم حالت PCF و هم حالت DCF را به کار گرفت. در نگاه اول ممکن است به نظر برسد که نظارت مرکزی و نظارت توزیع شده بطور همزمان میسر نباشد ولی در استاندارد 802.11 برای رسیدن به چنین هدفی راهکاری مناسب اندیشیده شده است. برای این کار، بازه‌های زمانی بین فریم‌ها بدقت تعریف می‌شود. پس از آنکه یک فریم ارسال شد و قبل از آنکه ایستگاهی بتواند فریم بعدی را ارسال نماید به مدتی «زمان مرده» نیاز است. در این زمان مرده، چهار بازه زمانی مجزا با اهداف خاص، تعریف



شکل ۲۹-۴. فاصله زمانی بین فریمها در 802.11.

شده است. این چهار بازه زمانی در شکل ۲۹-۴ نشان داده شده است. کوتاه ترین بازه زمانی، بازه SIFS (Short InterFrame Spacing) است. این بازه زمانی به ایستگاه ها فرصت می دهد تا به ارسال فریمهای کنترل خاص پردازند. در این زمان ایستگاه ها اجازه می یابند عملیاتی مثل ارسال CTS (در پاسخ به RTS)، ارسال فریم ACK در پاسخ به یک فریم کامل یا یک قطعه از فریم، ارسال یک قطعه از دنباله قطعات (بدون ارسال RTS مجدد) یا نظائر این را انجام بدهند.

همیشه فقط یک ایستگاه است که پس از زمان SIFS، به منظور پاسخ دهی [و ارسال فریم کنترل مناسب] حق ارسال دارد. اگر ایستگاه مربوط نتواند از این فرصت استفاده کند و زمان PIFS (PCF InterFrame Spacing) منقضی شود، ایستگاه ثابت می تواند «فریم فانوس دریایی» (Beacon) یا «فریم سرکشی» ارسال نماید. این مکانیزم اجازه می دهد که ایستگاه در حال انتقال فریم یا دنباله قطعات، بدون آنکه ایستگاه دیگری در این میان مداخله کند ارسال فریم خود را به پایان برساند در حالیکه ایستگاه ثابت نیز این فرصت را خواهد داشت که وقتی ایستگاه قبلی کارش را به اتمام رساند کانال را بدون رقابت با کاربران متمایل به ارسال تصرف نماید.

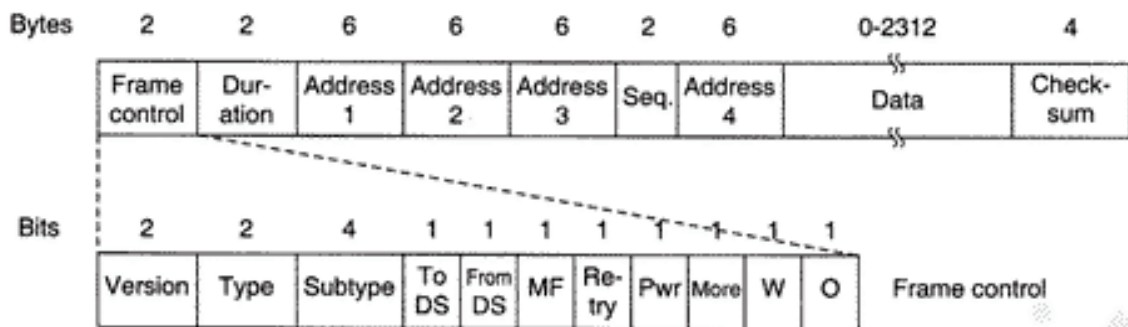
هرگاه ایستگاه ثابت چیزی برای ارسال نداشته باشد و زمان DIFS (DCF InterFrame Spacing) منقضی شود هر ایستگاه می تواند بخت خود را در تصرف کانال و ارسال فریم بیازماید. در این لحظه، برای در اختیار گرفتن کانال، روش معمولی رقابت و در صورت تصادم الگوریتم عقب گرد نمایی اعمال می شود.

آخرین بازه زمانی یعنی EIFS (Extended InterFrame Spacing) مورد استفاده ایستگاهی قرار می گیرد که یک فریم خراب یا فریمی ناشناس دریافت کند تا بتواند این مسئله را گزارش بدهد. دلیل اصلی آنکه به این بازه زمانی کمترین اولویت داده شده [آخرین بازه زمانی است] آن بوده که چون گیرنده نمی داند چه اتفاقی در جریان است لذا باید مدت زمان قابل توجهی صبر کند تا از هرگونه مداخله در گفتگوی دو ایستگاه اجتناب نماید.

۴-۴ ساختار فریم 802.11

استاندارد 802.11 سه رده مختلف فریم برای ارسال بر روی کانال تعریف کرده است: «فریم داده»، «فریم کنترل» و «فریم مدیریتی». هر یک از این فریمها سرآیند (Header) خاص خود را دارند که در هر سرآیند، فیلدهائی جهت استفاده در زیرلایه MAC تعریف شده است. مضاف بر این سرآیندهائی جهت به کارگیری در لایه فیزیکی تعریف گردیده که اغلب تکنیک های مدولاسیون [و پارامترهای مخابراتی] را مشخص می کنند، لذا در اینجا بدانها نخواهیم پرداخت.

قالب فریم داده، در شکل ۳-۴ نشان داده شده است. در ابتدا «فیلد کنترل» ظاهر شده که این فیلد خودش دارای یازده فیلد فرعی است. اولین زیرفیلد، شماره نسخه پروتکل (Protocol Version) را مشخص می کند؛



شکل ۳-۴. فریمهای داده در 802.11.

بدین ترتیب در آن واحد و در یک سلول مشابه، به کارگیری دو پروتکل متفاوت ممکن خواهد بود. در ادامه، زیرفیلد دو بیتی Type آمده که نوع فریم را (اعم از فریم داده، کنترلی و مدیریتی) مشخص می‌نماید؛ پس از آن زیرفیلد Subtype قرار گرفته که مشخصات دقیقتر فریم (مثل RTS و CTS) را تعریف می‌کند. سپس بیتهای ToDS و FromDS آمده که مشخص می‌کنند که آیا فریم از یک «سیستم توزیع درون‌سلولی» (مثلاً شبکه اترنت) بیرون آمده یا بدانجا رهسپار است. بیت MF نشان می‌دهد که هنوز قطعاتی از فریم در پیش رو هستند (و هنوز قطعاتی از فریم باقیست). بیت Retry نشانگر آنست که فریم جاری قبلاً یکبار ارسال شده است. ایستگاه ثابت بکمک بیت Power Management (که در شکل با نماد Pwr نشان داده شده است)، ایستگاه متحرک را به حالت استراحت (Sleep) برده یا آنرا از حالت استراحت بیرون می‌آورد. بیت More نشان می‌دهد که فرستنده باز هم فریمهایی برای ارسال به گیرنده آماده دارد. بیت W مشخص می‌کند که بدنه فریم با استفاده از الگوریتم WEP (Wire Equivalent Privacy) (فصل نهم) رمزنگاری شده است. نهایتاً بیت O به گیرنده تفهیم می‌کند دنباله‌ای از فریمها که این بیت در آنها ۱ است باید الزاماً به ترتیب (و پشت سرهم) پردازش شوند.

فیلد دوم از فریم داده یعنی فیلد Duration، مشخص‌کننده آنست که ارسال فریم جاری و دریافت ACK آن، جمعاً چه زمانی کانال را به حالت اشغال در خواهد آورد. این فیلد که در فریمهای کنترلی نیز وجود دارد مشخص می‌نماید که ایستگاه‌های دیگر به چه نحوی باید مکانیزم NAV خود را مدیریت کنند. سرآیند فریم حاوی چهار آدرس است که همگی منطبق با قالب استاندارد هستند که توسط IEEE 802 تعریف شده است. بدیهی است که به دو فیلد آدرس مبداء و مقصد نیاز بوده ولیکن دوتای دیگر چه کاربردی دارند؟ به خاطر داشته باشید که یک فریم ممکن است از طریق ایستگاه ثابت به یک سلول وارد یا از سلول خارج شود. دو آدرس اضافی، برای تعیین ایستگاه ثابت مبداء و مقصد بکار می‌رود. (وقتی که ترافیک داده‌ها بین چند سلول در حال جریان است).

فیلد Sequence اجازه می‌دهد تا قطعات یک فریم شماره گذاری شوند. از شانزده بیت موجود در این فیلد ۱۲ بیت، هویت فریم را و ۴ بیت شماره قطعه را مشخص می‌کند. در فیلد Data داده‌های خام قرار می‌گیرد که می‌تواند حداکثر ۲۳۱۲ بایت باشد. در آخر نیز فیلد Checksum قرار گرفته که به منظور کشف خطا کاربرد دارد.

«فریم مدیریتی» قالبی مشابه با «فریم داده» دارد با این تفاوت که این فریم یکی از آدرسهای ایستگاه ثابت را ندارد، زیرا فریمهای مدیریتی فقط محدود به یک سلول خاص هستند (و بین سلولهای متفاوت مبادله نخواهند شد). فریم کنترلی باز هم کوتاهتر هستند و فقط یک یا حداکثر دو فیلد آدرس دارند. این فریمها، فاقد فیلد داده و فیلد Sequence هستند. اطلاعات اساسی اینگونه فریمها، درون فیلد Subtype نهفته است که عموماً نوع RTS، CTS یا ACK را مشخص می‌کند.

۵-۴-۴ خدمات

استاندارد 802.11 بیان داشته که هر شبکه محلی بی سیم باید ۹ نوع خدمات عرضه نماید. این خدمات به دو رده تقسیم بندی شده اند. پنج نوع خدمات «توزیعی» و چهار نوع خدمات «ایستگاهی». خدمات توزیعی در خصوص مدیریت بر عضویت ایستگاههای درون سلول و تعامل با ایستگاههای خارج از سلول است. در مقابل، خدمات ایستگاهی صرفاً در خصوص فعالیتهای درون یک سلول واحد ارائه می شود.

پنج نوع خدمات توزیعی که توسط ایستگاههای ثابت عرضه می شوند، در خصوص قابلیت تحرک ایستگاههای همراه، ورود و خروج آنها به سلولها و اتصال یا انفصال از ایستگاه ثابت کاربرد دارند. این خدمات عبارتند از:

۱. Association (پیوستن به شبکه): ایستگاههای متحرک از این سرویس بهره می گیرند تا خود را به ایستگاه ثابت متصل نمایند. بطور معمول زمانی که یک ایستگاه متحرک وارد محدوده رادیویی یک ایستگاه ثابت می شود با استفاده از این سرویس، هویت و قابلیتهای خود را معرفی می کند. قابلیتها عبارتند از نرخ ارسال داده ها (نرخهای متعددی که از آن حمایت می شود)، نیاز به خدمات متعدد PCF (مثل سرکشی) و نیازمندیهای آن در خصوص مدیریت توان مصرفی. ایستگاه ثابت می تواند حضور ایستگاه متحرک را بپذیرد یا رد کند. اگر ایستگاه متحرک پذیرفته شد بایستی هویت خود را اثبات کند. [روشهای احراز هویت را در فصل نهم مطالعه کنید].

۲. Disassociation (ترک شبکه): ممکن است ایستگاه متحرک یا ایستگاه ثابت، در هر زمان اراده کنند از یکدیگر جدا شوند و ارتباط خود را قطع نمایند. وقتی ایستگاهی بخواهد شبکه را ترک کند یا خاموش شود، از این سرویس بهره می گیرد اما ایستگاه ثابت نیز قبل از قطع موقت ارتباط ممکن است از آن استفاده کند.

۳. Reassociation (پیوستن مجدد): یک ایستگاه متحرک می تواند با استفاده از این سرویس، ایستگاه ثابت خود را تغییر بدهد. این قابلیت زمانی کاربرد دارد که ایستگاههای متحرک بخواهند از یک سلول به سمت سلول دیگر حرکت نمایند. اگر از این امکان به موقع و صحیح استفاده شود در اثر این جابجائی، هیچ داده ای از دست نخواهد رفت.

۴. Distribution (توزیع): این سرویس مسیر ارسال فریمهای ارسالی به سوی ایستگاه ثابت را تعیین می نماید. اگر مقصد فریمها در همان محل ایستگاه ثابت واقع شده باشد فریم می تواند مستقیماً از طریق هوا ارسال شود. در غیر این صورت فریمها باید از طریق شبکه سیمی (ارتباط سیمی بین ایستگاههای ثابت) به مقصد هدایت شود.

۵. Integration (یکپارچگی): اگر نیاز باشد فریمی به شبکه ای غیر از 802.11 (با ساختار آدرس و ساختار فریم متفاوت) ارسال شود، این سرویس می تواند وظیفه ترجمه و تبدیل قالب فریم (متناسب با شبکه مقصد) را بر عهده بگیرد.

چهار سرویس باقیمانده در درون یک سلول بکار می آیند. (بعبارت دیگر در خصوص عملیات درون یک سلول واحد کاربرد دارند). این سرویسها پس از پیوستن ایستگاهها به یک ایستگاه ثابت [به عنوان عضوی از شبکه] مورد استفاده قرار گرفته و به عبارت زیر هستند:

۱. Authentication (احراز هویت): از آنجایی که در مخابرات بی سیم ایستگاههای غیر مجاز و بیگانه نیز می توانند ارسال و دریافت داشته باشند فلذا هر ایستگاه باید قبل از دریافت مجوز ارسال، هویت خود را اثبات کند. پس از آنکه یک ایستگاه متحرک به عنوان عضو، به یک ایستگاه ثابت پیوست (یا بعبارت دیگر درون

سلول پذیرفته شد)، ایستگاه ثابت یک فریم خاص به نام «فریم چالش» (Challenge) برای او می‌فرستد تا ببیند آیا آن ایستگاه کلید سری (کلمه عبور) خود را می‌داند یا نه؟ ایستگاه، آگاهی خود از کلید سری را با رمز کردن فریم و بازگرداندن آن، اثبات می‌نماید. اگر نتیجه درست باشد، ایستگاه متحرک در درون سلول ثبت‌نام و عضو می‌شود. در استاندارد اولیه لازم نبود که ایستگاه ثابت نیز عضویت خود را احراز کند در حالیکه اصلاح این اشکال بزرگ در دست اجرا است.

۲. Deauthentication (لغو حضور در شبکه): وقتی یک ایستگاه که قبلاً احراز هویت (و عضو) شده بخواهد شبکه را ترک کند باید لغو حضور خود در شبکه را به اطلاع ایستگاه ثابت برساند. پس از لغو حضور و ترک شبکه، ایستگاه دیگر نمی‌تواند از شبکه بهره بگیرد.

۳. Privacy (محرمانه نگاه داشتن اطلاعات): برای محرمانه ماندن اطلاعاتی که از طریق شبکه بی‌سیم مبادله می‌شود، بایستی رمز شوند. این کار از طریق رمزنگاری و رمزگشایی میسر است. الگوریتم رمزنگاری مورد استفاده روش RC4 می‌باشد که توسط رونالد ری‌وست در دانشگاه MIT ابداع شده است.

۴. Delivery (تحویل): انتقال داده‌ها کل آن هدفی است که در پی آن هستیم لذا در 802.11 روشی برای ارسال و دریافت داده‌ها ارائه شده است. از آنجایی که 802.11 مبتنی بر مدل شبکه اینترنت است و مبادله اطلاعات در اینترنت تضمین صد درصد ندارد، 802.11 نیز مبادله مطمئن داده‌ها را تضمین نمی‌کند. لایه‌های بالاتر باید در خصوص کشف و تصحیح خطا کاری انجام بدهند.

در استاندارد 802.11، هر سلول پارامترهایی دارد که می‌توان آنها را بررسی و در صورت نیاز تنظیم کرد. این پارامترها در خصوص عملیات رمزنگاری، نرخ ارسال، بازه‌های زمانی [زمان‌های انقضای مهلت یا Timeouts]، تناوب ارسال فریمهای Beacon و نظایر اینها، تعریف شده‌اند.

شبکه‌های محلی بی‌سیم 802.11، در حال ورود به عرصه ادارات، فروشگاه‌ها، هتلها، رستورانها و محوطه‌های دانشگاهی هستند و انتظار می‌رود رشد سریع و چشمگیری داشته باشند. برای آنکه در خصوص استفاده گسترده از 802.11، اطلاعات و تجاربی بدست بیاورید به مرجع (Hills, 2001) مراجعه کنید.

۵-۴ بی‌سیم با باند گسترده

بیش از اندازه به درون (شبکه‌های LAN) پرداخته‌ایم. حال اجازه بدهید پا را فراتر گذاشته و ببینیم آیا در خارج از دنیای LAN هم شبکه‌های جالبی وجود دارند. با رفع موانع قانونی از مقررات حاکم بر عرضه خدمات تلفن، در بسیاری از کشورها شرکت‌های مخابراتی رقیب اجازه یافتند تا به ارائه خدمات صوت و خدمات اینترنت پرسرعت بپردازند. امروزه نیاز عمومی در این خصوص بسیار بالا و پررونق است. برای چنین شرکت‌هایی مسئله اساسی آنست که کشیدن کابل‌های فیبرنوری، کابل کوآکسیال یا سیم‌های زوجی (از نوع CAT 5) تا درب در میلیون‌ها خانه و محل کار، بسیار گران و نامعقول به نظر می‌رسد. پس این رقبا برای تجاری برای عرضه خدمات ارزان، چه باید می‌کردند؟

پاسخ به این نیاز «شبکه بی‌سیم باند گسترده» است. برافراشتن یک آنتن بزرگ بر روی قله یا یک تپه و نصب آنتنهای بر روی پشت بام مشتریان که به سمت آنتن اصلی جهت‌گیری شده، بسیار ارزان‌تر از حفر زمین و کابل‌کشی است. بهمین دلیل شرکت‌های مخابرات راه دور بیشتر گرایش دارند که برای عرضه خدمات اینترنت، ارسال صدا و ارائه نمایشهای ویدئویی راه دور، از سیستم‌های چندگشایی بی‌سیم بهره بگیرند. همانگونه که در شکل ۳-۲ ملاحظه نمودید، LMDS به همین منظور ابداع شد ولیکن تاکنون هر یک از شرکت‌های عرضه‌کننده

LMDs، به سلیقه خود سیستمی را طراحی و نصب کرده است. فقدان یک استاندارد واحد، بدین معناست که سخت افزار و نرم افزار آنها نمی تواند بطور انبوه تولید شود و طبعاً قیمت آنها بالا و استقبال عمومی از آنها کم خواهد بود.

نهایتاً بسیاری از دست اندرکاران صنعت بدین نتیجه رسیدند که داشتن یک استاندارد برای بی سیم باند گسترده [با نرخ ارسال بسیار بالا] یکی از مولفه های اصلی و حلقه مفقوده در کار آنهاست، لذا از IEEE خواستند تا برای تدوین چنین استانداردی، کمیته ای با حضور دست اندرکاران شرکت های مهم و مراکز دانشگاهی، تشکیل بدهد. اولین عدد اختصاصی داده نشده در ردیف شماره های 802.x، ۱۶ بود و به همین دلیل استاندارد، با عنوان IEEE 802.16 معرفی شد. کار کمیته در ژولای ۱۹۹۹ شروع و استاندارد نهانی در آوریل ۲۰۰۲ به تائید رسید. نام رسمی استاندارد «واسط هوایی برای سیستم های بی سیم غیر متحرک با پهنای باند وسیع»^۱ انتخاب شده ولیکن برخی از افراد ترجیح می دهند آنرا «شبکه بین شهری بی سیم» یا «حلقه بی سیم» بنامند. ما تمام این واژه ها را معادل یکدیگر فرض خواهیم کرد.

همانند بسیاری از استانداردهای دیگر سری ۸۰۲، استاندارد 802.16 نیز شدیداً تحت تاثیر مدل OSI بوده و این تاثیر در (زیر) لایه ها، اصطلاحات، سرویس های پایه و موارد دیگر بخوبی محسوس است. متأسفانه این استاندارد نیز شبیه به OSI پیچیده شده است. در بخش بعدی بطور مختصر نکات و ویژگی های برجسته 802.16 را برخواهیم شمرد ولی این توضیحات به هیچ وجه کامل نبوده و جزئیات آن ناگفته خواهد ماند.

۱.۵.۴ مقایسه 802.11 با 802.16

در همان ابتدا ممکن است بدین نکته بیندیشید که چرا استاندارد جدیدی ابداع شد؟ چرا از 802.11 استفاده نشد؟ دلایل متعدد و محکمی برای عدم استفاده از 802.11 وجود دارد. در اصل 802.11 و 802.16 نیازهای متفاوتی را برآورده می کنند. قبل از پرداختن به فناوری 802.16 شاید چند کلمه ای بحث در خصوص دلایل نیاز به استاندارد جدید، خالی از لطف نباشد.

محیطی که 802.11 و 802.16 در آن، عمل می کنند از چند منظر شبیه به هم هستند: در اصل هر دوی این استانداردها بدان جهت طراحی شده اند که ارتباط بی سیم با پهنای باند بسیار بالا را میسر نمایند. ولیکن این دو شبکه از جهات بسیار مهمی با هم متفاوتند. اصلی ترین تفاوت آنست که 802.16 برای ارائه خدمات به ساختمانها طراحی شده است و طبعاً ساختمانها حرکت نمی کنند!!! ساختمانها تغییر سلول نمی دهند! بخش اعظم عملیات 802.11 با مسائل ناشی از متحرک بودن ایستگاه ها سروکار دارد و چنین مسائلی در 802.16، محلی از اعراب ندارد. گذشته از آن، در ساختمانها ممکن است بیش از یک کامپیوتر وجود داشته باشد و پیچیدگی های چنین محیطی در جایی که ایستگاه نهانی یک کامپیوتر کیفی واحد است، بروز نخواهد کرد. از آنجایی که مالکین ساختمان معمولاً آمادگی پرداخت پول بیشتری در مقایسه با صاحب یک کامپیوتر کیفی دارند لذا می توان خدمات ارتباط رادیویی بهتری در اختیارشان گذاشت. این تفاوت بدین معناست که در 802.16 می توان ارتباط دوطرفه همزمان (Full Duplex) داشت در حالیکه برای پائین نگاه داشتن هزینه ارتباط رادیویی در 802.11 از آن اجتناب شده است. [ارتباط 802.11 دو طرفه غیر همزمان - Half Duplex - و با برد بسیار کم می باشد.]

از آنجایی که 802.16 در محدوده بخشی از یک شهر به اجرا در می آید، فواصل [بین نقاط] می تواند تا چندین کیلومتر باشد و این موضوع بدان معناست که توان مورد نیاز ایستگاه ثابت می تواند بسته به موقعیت و فاصله ها متغیر باشد. این تفاوتها و تغییرها بر روی نسبت سیگنال به نویز (SNR) تاثیر گذاشته و در نتیجه، باید به اجبار از

۱. Air Interface for Fixed Broadband Wireless Access Systems

چندین روش مدولاسیون استفاده شود. همچنین مخابرات آزادانه در سطح یک شهر مبین آنست که تمهیدات امنیتی و حفظ حریم افراد، نیازی بنیادی و اجتناب ناپذیر است.

به علاوه، در مقایسه با سلول های معمول 802.11، در اینجا سلولها می توانند کاربران بسیار زیادتری را در بر بگیرند و این کاربران نیز توقع پهنای باند بیشتری نسبت به کاربران 802.11 دارند. گذشته از آن به ندرت اتفاق می افتد که شرکتی از پنجاه کارمند خود دعوت کند تا برای مشاهده اشباع شدن [و از کار افتادن 802.11] دور هم جمع شده و پنجاه فیلم مجزا تماشا کنند!! [در حالیکه در یک ساختمان مسکونی احتمال دارد پنجاه نفر بدیدن فیلم از طریق شبکه مشغول باشند]. به همین دلیل به پهنای وسیعتری از طیف فرکانسی موجود در باند ISM نیاز است و طبعاً 802.11 مجبور است در محدوده فرکانسی ۱۰ تا ۶۶ گیگاهرتزی عمل کند؛ یعنی تنها محدوده استفاده نشده از طیف فرکانس که هنوز موجود است.

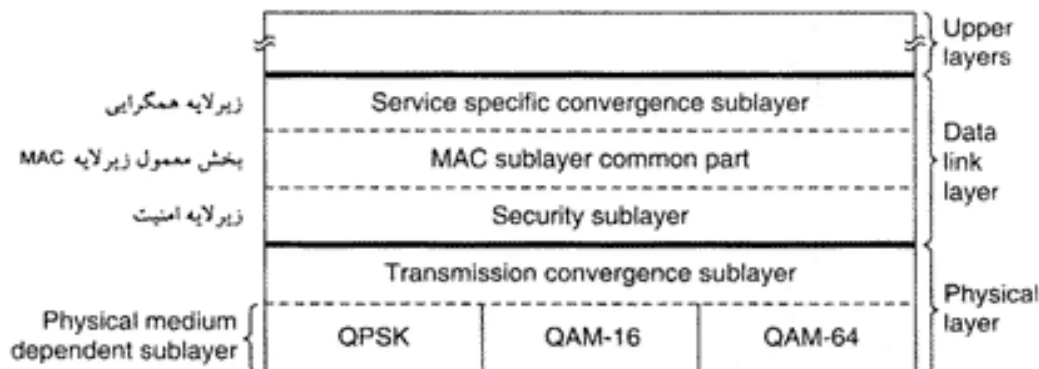
ولیکن این امواج با طول موج میلی متری نسبت به امواج با طول موج بیشتر در باند ISM، ویژگی های فیزیکی کاملاً متفاوتی دارند و در نتیجه به لایه فیزیکی کاملاً متفاوتی نیاز است. یکی از ویژگی های امواج میلی متری آنست که توسط آب شدیداً جذب می شوند. (بالاخص باران، در برخی از مواقع برف، تگرگ و متاسفانه مه غلیظ) در نتیجه مدیریت خطا بسیار با اهمیت تر از محیط های داخلی (مثل محیط LAN) است. امواج میلی متری می توانند به خط مستقیم متمرکز و منتشر شوند (در حالیکه در 802.11 انتشار امواج در همه جهات است) فلذا گزینه ها و تمهیدات پیش بینی شده در ارتباط با انتشار چندمسیره (Multipath) در 802.16 محلی از اعراب ندارد.

مورد دیگر مربوط به «کیفیت خدمات» (QoS) است. اگر چه 802.11 برای ترافیک بی درنگ پشتیبانی به عمل آورده (در حالت PCF) ولیکن حقیقتاً برای کاربردهای تلفنی و عملیات چندرسانه ای سنگین و دائم طراحی نشده است. در مقابل، از 802.16 انتظار می رود که کاملاً از چنین کاربردهایی پشتیبانی نماید زیرا برای استفاده در محیط های مسکونی و اداری مد نظر بوده است.

کوتاه سخن آنکه 802.11 طراحی شده تا یک اترنت متحرک باشد در حالیکه 802.16 طراحی شده تا شبیه به تلویزیون کابلی، ثابت ولی بی سیم عمل نماید. این تفاوتها آنقدر بنیادیست که استانداردهای حاصل اختلاف فراوانی دارند و در هر یک سعی در بهینه سازی و حل و فصل نیازهای متفاوتی شده است. مقایسه ای بسیار کوتاه با سیستم تلفن سلولی [تلفن همراه] نیز خالی از فایده نیست. وقتی در مورد تلفن های همراه صحبت می کنیم روی سخن با ایستگاه هائی همراه با محوریت ارسال صوت، توان مصرفی پائین و پهنای باند باریک است که از امواج مایکروویو با طول موج متوسط بهره گرفته اند. هیچ کس یک فیلم دو ساعته و با کیفیت بالا را بر روی گوشی موبایل GSM خود تماشا نمی کند (البته فعلاً!! حتی UMTS نیز امیدی به تغییر چنین وضعیتی ندارد. [امید به افزایش چشمگیر پهنای باند در GSM] در عبارتی کوتاه در دنیای شبکه های بین شهری بی سیم (Wireless MAN) تقاضا برای پهنای باند بسیار بیشتر از دنیای تلفن همراه است و طبعاً به سیستمهائی کاملاً متفاوت نیاز می باشد. اینکه آیا 802.16 می تواند در آینده برای ابزارهای متحرک و همراه نیز به کار گرفته شود سوال جالبی است: این شبکه برای چنین محیطی بهینه نشده ولیکن شاید بشود! فعلاً تمرکز آن بر روی شبکه داده بی سیم و غیر متحرک است.

۲-۵-۴ پشته پروتکلی 802.16

شکل ۴-۳۱ پشته پروتکلی 802.16 را نشان می دهد. ساختار کلی این پشته، شبیه به شبکه های دیگر 802 است ولیکن تعداد بیشتری زیرلایه دارد. پائینترین زیرلایه با انتقال فیزیکی بیتها بر روی کانال سروکار دارد و در آن از رادیوی باند باریک و روش های معمول مدولاسیون، بهره گرفته شده است. بر روی «لایه فیزیکی انتقال»، زیرلایه «همگرایی» (Convergence Sublayer) قرار گرفته تا تکنولوژیهای متفاوت به کار رفته در زیر را از دید لایه پیوند



شکل ۳-۴. پشته پروتکلی 802.16.

داده ها مخفی نگاه دارد. در حقیقت، 802.11 نیز چیزی شبیه به همین زیرلایه را دارد ولی کمیته مربوطه، آنرا با استفاده از اسامی رایج در مدل OSI تعریف نکرده است.

اگرچه در شکل ۳-۴ نشان نداده ایم ولی در آینده دو پروتکل جدید به لایه فیزیکی اضافه می شود: (۱) استاندارد 802.16a که از روش مدولاسیون OFDM در باند ۲ تا ۱۱ گیگاهرتز پشتیبانی می نماید. (۲) استاندارد 802.16b که در باند ۵ گیگاهرتز ISM عمل می کند. در هر دوی اینها تلاش شده تا 802.16 به 802.11 نزدیکتر شود.

لایه پیوند داده متشکل از سه زیرلایه است: زیرلایه پائینی با امنیت و محرمانه نگاه داشتن اطلاعات سروکار دارد که برای شبکه های عمومی در محیط های باز بسیار حیاتی تر از شبکه های خصوصی بسته مثل اترنت است. این زیرلایه عملیات رمزنگاری، رمزگشایی و مدیریت کلیدها را برعهده دارد.

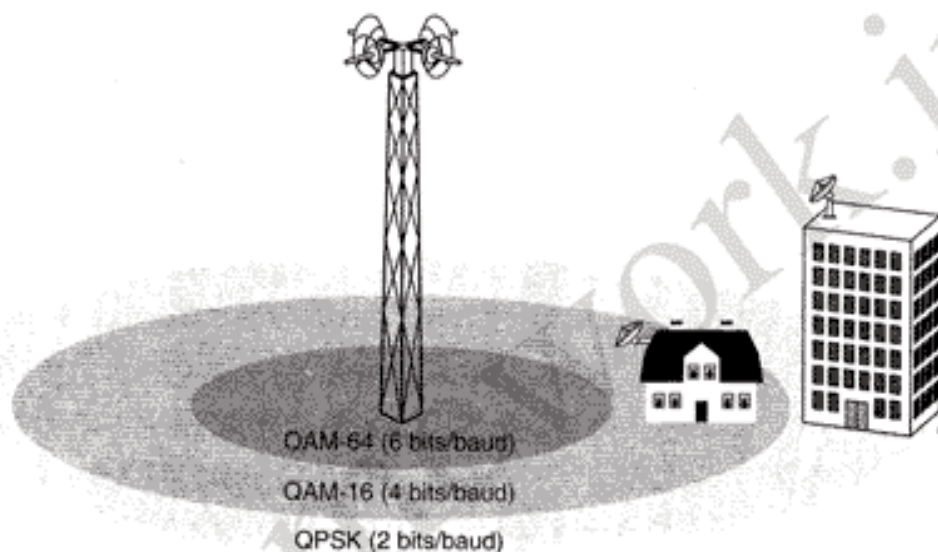
سپس بخش مشترک از زیرلایه MAC قرار می گیرد. این لایه همان نقطه ای است که پروتکل های اساسی مثل پروتکل های مدیریت کانال در بر می گیرد. در این مدل مبنا آنست که ایستگاه ثابت، سیستم را کنترل می کند. این ایستگاه می تواند «جریان اطلاعات از ایستگاه ثابت به مشترکین» (که اصطلاحاً downstream نام دارد) را به صورت کاملاً متفاوت و متمایز زمان بندی نماید و نیز نقش بسیار مهمی در مدیریت «جریان اطلاعات از مشترکین به ایستگاه ثابت» (Upstream) ایفا می کند. یکی از ویژگی های نامتعارف در زیرلایه MAC آنست که برخلاف شبکه های دیگر 802، این استاندارد کلاً اتصال گرا (Connection Oriented) است، تا بتواند «کیفیت خدمات» (QoS) را برای ارتباطات تلفنی و سایر کاربردهای چندرسانه ای تضمین نماید.

زیرلایه «همگرایی خاص سرویس دهی» (Service Specific Convergence Sublayer) به جای «زیرلایه لینک منطقی» (یعنی زیرلایه معمول LLC) قرار گرفته است. عملکرد این زیرلایه ایجاد واسطی متناسب با لایه شبکه است. پیچیدگی این زیرلایه از آنجا نشأت می گیرد که 802.16 به نحوی طراحی شده تا بتواند با پروتکل های دیتاگرام (مثل PPP، IP و اترنت) و همچنین شبکه اتصال گرای ATM قابل جمع باشد و با آنها کار کند. مشکل آنجاست که پروتکل های «مبتنی بر بسته» بدون اتصال (Connectionless) هستند در حالیکه ATM اتصال گراست. این بدین معناست که هر اتصال ATM بایستی به یک اتصال در 802.16 نگاشته شود که اصولاً کار ساده و سراسری است. ولی سوال این است که یک بسته IP ورودی باید بروی کدامین اتصال 802.16 نگاشته شود؟ این مشکل در همین زیرلایه حل خواهد شد.

۳-۵-۴ لایه فیزیکی در 802.16

بالا اشاره شد که در بی سیم یا پهنای باند وسیع، به حس گسترده تری از طیف فرکانسی نیاز است و تنها محل

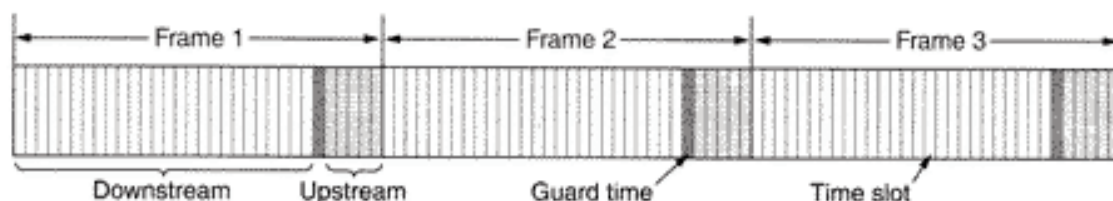
دسترسی به چنین وسعتی محدوده ۱۰ تا ۶۶ گیگاهرتزی است. این امواج با طول موج میلی متری ویژگیهای جالبی دارند که امواج مایکروویو با طول موج بلندتر ندارند: آنها برخلاف صوت و مشابه با نور به خط مستقیم سیر می کنند. در نتیجه، ایستگاه ثابت بایستی چندین آنتن داشته باشد و هر یک از آنها بسوی قطاع خاصی از مناطق پیرامون خود نشانه رفته باشد. (شکل ۴-۳۲) هر قطاع، کاربران خاص خود را دارد و مستقل از قطاعهای همجوار خود است؛ چنین ساختاری برای رادیوسوی سلولی صادق نیست چرا که آنتنهای آنها «همه جهته» (Omnidirectional) هستند.



شکل ۴-۳۲. محیط انتقال در 802.16.

از آنجایی که در باند امواج میلی متری توان سیگنال براساس فاصله از ایستگاه ثابت، شدیداً کاهش می یابد لذ نسبت سیگنال به نویز (SNR) نیز برحسب فاصله افت خواهد داشت. به همین دلیل 802.16 بسته به فاصله یک مشترک از ایستگاه ثابت، سه روش مدولاسیون متفاوت را به کار گرفته است. برای مشترکین نزدیک، از روش QAM-64 با مشخصه 6 bits/baud استفاده شده است. برای مشترکین با فاصله متوسط از QPSK با مشخصه 2 bits/baud استفاده می شود. برای مشترکین دور دست، روش QPSK با مشخصه 2 bits/baud به کار می رود. برای مثال به ازای پهنای معمول ۲۵ مگاهرتزی از طیف فرکانس، پهنای باند روش QAM-64 نرخ ۱۵۰ مگابیت بر ثانیه، روش QAM-16 نرخ ۱۰۰ مگابیت بر ثانیه و QPSK نرخ ۵۰ مگابیت بر ثانیه را در اختیار قرار می دهد. به عبارت دیگر هر چه مشترکین از ایستگاه ثابت دورتر باشند، نرخ ارسال پائینتر خواهد بود (شبیه به آنچه که در خصوص ADSL در شکل ۲۷-۲ دیدید) در شکل ۲۵-۲ نمودار فضایی این سه مدولاسیون، نمایش داده شده است.

با توجه به آنکه هدف اصلی ایجاد سیستمی با باند وسیع بوده و با در نظر داشتن محدودیتهای فیزیکی. طراحان 802.16 کوشیدند تا از طیف موجود به نحو بهینه استفاده کنند. آنها به روش به کار گرفته در GSM و DAMPS علاقه ای نداشتند. هر دوی این سیستمها از فرکانسهای متفاوت ولی در باند مشابهی برای ارسال جریان ترافیک رو به بالا و رو به پایین (Upstream/Downstream) بهره می گیرند. برای صوت شاید ترافیک در اکثر بخشها متقارن باشد [عبارت دیگر برای صوت میزان ارسال و دریافت محدوداً به یک اندازه است] ولیکن برای دسترسی به اینترنت غالباً حجم ترافیک دریافتی بیشتر از ترافیک ارسالی است. در نتیجه 802.16 روشی متعطفتر برای تخصیص پهنای باند ارائه کرده و از دو روش FDD (Frequency Division Duplexing) و TDD (Time Division Duplexing) بهره گرفته است. روش دوم یعنی TDD، در شکل ۴-۳۳ نشان داده شده



شکل ۳۳-۴. فریمها و برشهای زمانی در روش TDD (Time Division Duplexing).

است. در اینجا ایستگاه ثابت بطور متناوب فریمهایی را منتشر می کند. هر فریم شامل تعدادی برش زمانی مستقل (Time Slot) است. برشهای ابتدایی هر فریم، برای ارسال ترافیک روبه پائین [یعنی از ایستگاه ثابت به کاربر] در نظر گرفته شده است. پس از آن «زمان مراقبت» (Guard Time) فرا می رسد که به ایستگاه ها مهلت می دهد تا جهت ارسال و دریافت خود را تغییر بدهند. در آخر برشهایی را برای ارسال ترافیک رو به بالا [از کاربر به ایستگاه ثابت] خواهیم داشت. تعداد برشهای زمانی را که به ارسال در هر یک از جهات، اختصاص می یابد، می توان به صورت پویا تغییر داد تا پهنای باند در هر جهت، با حجم ترافیک تطبیق داشته باشد.

ترافیک روبه پائین توسط ایستگاه ثابت در درون برشهای زمانی نگاشته می شود. ایستگاه ثابت بطور کامل بر این جهت از جریان، کنترل دارد. ترافیک روبه بالا [که توسط کاربران تولید می شود] با پیچیدگی بیشتری مواجه بوده و میزان آن به کیفیت خدمات (QoS) مورد نیاز بستگی دارد. در ادامه وقتی به تشریح زیرلایه MAC پرداختیم به روش تخصیص برشهای زمانی هم خواهیم رسید.

ویژگی جالب دیگر در لایه فیزیکی، توانایی آن در ارسال متوالی و پشت سرهم فریمهای MAC، در قالب یک انتقال فیزیکی واحد است. این ویژگی سرشار ناشی از بیتبهای آغازین (Preamble) و سرآیند لازم در لایه فیزیکی را کاهش داده و در نتیجه بازده مفید طیف را افزایش خواهد داد.

نکته قابل توجه دیگر، استفاده از کدهای همینگ (Hamming) به منظور تصحیح مستقیم خطا در لایه فیزیکی است.^۱ تقریباً در تمام شبکه ها، به کدهای کشف خطا بسنده می شود و هرگاه فریم دریافتی دارای خطا باشد ارسال مجدد صورت می گیرد، ولی از آنجاییکه در محیط های باز و در نرخ ارسال بالا، احتمال بروز خطا در حین انتقال، خیلی بیشتر است لذا گذشته از عملیات کشف خطا (که در لایه های بالاتر انجام می شود) در لایه فیزیکی نیز عملیات تصحیح خطا صورت می گیرد. تاثیر نهانی تصحیح خطا آنست که کانال بهتر از آنی که هست به نظر می رسد. (بدلیل مشابه اگر چه CD-ROM ها قابل اعتماد به نظر می رسند ولیکن این اعتماد، حاصل از تخصیص بیش از نیمی از بیتها به تصحیح خطا در لایه فیزیکی است.) [به عبارت دیگر در هر CD، بیش از نیمی از بیتبهای سطح دیسک فقط به منظور عملیات کشف و تصحیح خطا ذخیره شده اند!]

۴-۵-۴ پروتکل زیرلایه MAC در 802.16

همانگونه که در شکل ۳۱-۴ دیدیم، در 802.16، لایه پیوند داده ها به سه زیرلایه تقسیم می شود. از آنجایی که ما تا فصل هشتم به مطالعه رمزنگاری نخواهیم پرداخت لذا تشریح عملکرد «زیرلایه امنیت» در اینجا سخت است. به همین مقدار بسنده می کنیم که برای سری نگاه داشتن داده های ارسالی از رمزنگاری در سطح لایه فیزیکی و به صورت بی درنگ، بهره گرفته شده است. صرفاً بخش داده هر فریم رمزنگاری می شود و سرآیند آن (Header) رمز نخواهد شد. این خصوصیت بدین معناست که یک جاسوس قادر است بفهمد چه کسی با چه کسی محاوره می کند ولی قادر به فهم آنچه با یکدیگر می گویند نیست.

۱. Forward Error Correction

اگر با رمزنگاری آشنائی قبلی داشته باشید در حد یک پارگراف، زیرلایه امنیت را بررسی می‌کنیم ولیکن در صورت عدم آشنائی با رمزنگاری، پارگراف بعدی چیزی به دانش شما نخواهد افزود. (می‌توانید این پارگراف را پس از اتمام فصل ۸ مجدداً مطالعه نمایید).

زمانی که یک مشترک به ایستگاه ثابت متصل می‌شود، دویه دو یکدیگر را با استفاده از روش «رمزنگاری RSA» و مبتنی بر «گواهینامه‌های X.509» احراز هویت می‌کنند. بخش داده فریمهای آنها با استفاده از یک سیستم رمزنگاری متقارن رمز می‌شود که این سیستم یا مبتنی بر «DES» یا زنجیره‌سازی بلوکها^۱ است و یا از روش «DES سه گانه با دو کلید» استفاده می‌شود. احتمالاً به زودی روش AES (Rijndael) نیز به آن اضافه می‌شود. عملیات بررسی صحت داده‌ها نیز با استفاده از SHA-1 انجام می‌گیرد. تا اینجا چندان بد نبود، نظر شما چیست؟!۲

حال اجازه بدهید نگاهی به بخش مشترک از زیرلایه MAC بیندازیم. فریمهای MAC، تعداد مشخصی از برشهای زمانی لایه فیزیکی را اشغال می‌کنند. هر فریم از چندین فریم کوچکتر (Subframe) تشکیل شده که دو تای ابتدائی آن به ترتیب برای نگاشت و حمل ترافیک روبه بالا (Upstream) و روبه پائین (Downstream) در نظر گرفته شده است. ساختار این نگاشت (یعنی درج داده‌ها درون برشهای زمانی) بگونه‌ای است که مشخص می‌کند چه چیزی در یک برش زمانی درج شده و کدامیک از برشهای زمانی آزاد هستند. نگاشت ترافیک روبه پائین نیز شامل پارامترهای مختلف سیستمی است تا شروع فعالیت یک ایستگاه جدید را به اطلاع ایستگاه ثابت برساند. عملکرد کانال روبه پائین نسبتاً ساده و سر راست است. ایستگاه ثابت بسادگی تصمیم می‌گیرد که چه چیزی را در کدام فریم کوچک (Subframe) قرار بدهد. عملکرد کانال رو به بالا پیچیده‌تر است زیرا در اینجا مشترکین رقیب و ناهماهنگ، برای در اختیار گرفتن آن با یکدیگر رقابت می‌کنند. روش تخصیص کانال به مواردی در خصوص «کیفیت خدمات» (QoS) بستگی دارد. در 802.16 چهار رده از خدمات، به ترتیب ذیل تعریف شده‌اند:

۱. خدمات با نرخ ارسال ثابت (Constant Bit Rate Service)
۲. خدمات بی‌درنگ با نرخ ارسال متغیر (Real-time Variable Bit Rate)
۳. خدمات غیر بی‌درنگ با نرخ ارسال متغیر (NonReal-time Variable bit Rate)
۴. خدمات مبتنی بر «حداکثر تلاش» (Best Effort Service)

تمام خدمات عرضه شده در 802.16 اتصال‌گرا هستند و به هر اتصال^۱ فقط یکی از رده‌های خدمات فوق تعلق می‌گیرد و نوع خدمات نیز در زمان برقراری اتصال تعیین می‌شود. این طراحی بسیار متفاوت از 802.11 یا اترنت است که در آنها هیچ اتصالی در زیرلایه MAC ایجاد نمی‌شود.

«خدمات با نرخ ارسال ثابت» برای انتقال سیگنال صوتی غیر فشرده همانند یک کانال T1 مد نظر بوده است. به این خدمات از آن جهت نیاز است که بتوان حجم معینی از داده‌ها را در زمان مشخصی ارسال کرد. برای رسیدن به این هدف، به هر اتصال از این نوع، تعداد ثابت و معینی برش زمانی (در هر فریم^۲) اختصاص داده می‌شود. هرگاه پهنای باند لازم اختصاص داده شود این برشهای زمانی بطور خودکار و بدون نیاز به درخواست بعدی در اختیار خواهد بود.

«خدمات بی‌درنگ با نرخ ارسال متغیر» برای کاربردهای چند رسانه‌ای فشرده‌شده و هرگونه عملیات بی‌درنگ که در آنها مقدار پهنای باند مورد نیاز، متغیر است سودمند خواهد بود. این کار بدین نحو انجام می‌شود که ایستگاه ثابت در فواصل زمانی مشخص به مشترک خود سرکشی کرده و از او در خصوص میزان پهنای باند مورد نیازش

۱. اتصال را یک ارتباط منطقی و هماهنگ شده بین دو نقطه در شبکه در نظر بگیرید. -م

۲. معنای واژه فریم در اینجا با معنای متعارف آن تفاوت دارد. معنای فریم 802.16 در شکل ۴-۳۳ مشخص است. -م

سوال می کند.

«خدمات غیربی درنگ با نرخ ارسال متغیر» برای حجم ترافیک سنگینی که بی درنگ نیست (همانند انتقال فایل های طولانی) در نظر گرفته شده است. برای ارائه چنین خدماتی، اغلب ایستگاه ثابت به مشترک خود سرکشی می کند ولیکن فواصل زمانی سرکشی به مشترک، قطعی و منظم نیست. یک مشترک که از نرخ ثابت بهره می گیرد می تواند یک بیت خاص را در یکی از فریمهای خود تنظیم کرده و تقاضای سرکشی بدهد تا بتواند ترافیک اضافی (با نرخ متغیر) ارسال نماید.

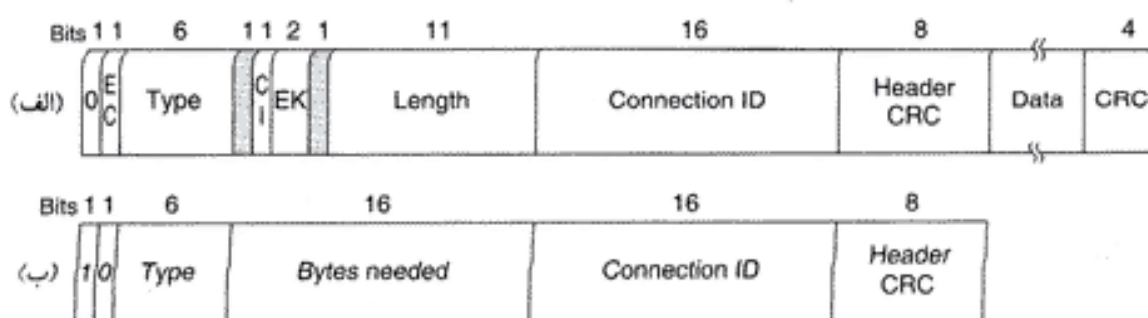
اگر ایستگاهی k بار متوالی سرکشی شود و پاسخی ندهد، ایستگاه ثابت او را در یک «گروه چندپخش» (Multicast Group) قرار داده و از آن به بعد بصورت اختصاصی سرکشی نخواهد شد. در عوض وقتی به یک «گروه چندپخش» سرکشی می شود هر کدام از ایستگاه های گروه می توانند پاسخ بدهند و برای دریافت خدمات رقابت کنند. بدین ترتیب ایستگاه هایی که ترافیک ناچیزی دارند زمان باارزش سرکشی را هدر نخواهند داد.

خدمات مبتنی بر «بهترین تلاش» در موارد متفاوتی کاربرد دارد. در اینجا هیچ سرکشی انجام نمی گیرد و هر مشترک برای دریافت این خدمات باید با مشترکین دیگر (که آنها نیز در پی خدمات مبتنی بر بهترین تلاش هستند) رقابت کند. تقاضای پهنای باند با علامتگذاری در یکی از برشهای زمانی ترافیک روبه بالا که برای رقابت تدارک دیده شده انجام می گیرد. اگر تقاضا به صورت موفقیت آمیز اعلان شود، این موفقیت در نگاشت ترافیک روبه پائین [فریم مربوط به Downstream] مشخص خواهد شد. اگر تقاضا موفق نبود، مشترک بایستی مجدداً تلاش کند. برای آنکه تصادمها حداقل شود از الگوریتم عقب گرد نمایی در اترنت بهره گرفته شده است.

استاندارد 802.16، دو نوع تخصیص پهنای باند تدارک دیده است: تخصیص پهنای باند به ازای هر ایستگاه و تخصیص پهنای باند به ازای هر اتصال (per-connection). در روش اول، ایستگاه نصب شده در یک ساختمان، کلیه تقاضاهای کاربران را به صورت یکجا جمع کرده و به نیابت از همه آنها تقاضای پهنای باند می کند و اگر توانست پهنای باند درخواستی را بدست بیاورد، این پهنای باند را به تناسب بین کاربران تقسیم می کند. در روش دوم، ایستگاه ثابت هر اتصال را مستقلاً مدیریت می کند.

۵-۵-۴ ساختار فریم در 802.16

تمام فریمهای MAC با یک سرآیند عمومی شروع می شوند. به نحوی که در شکل ۴-۳۴ می بینید پس از سرآیند، بخشی اختیاری جهت حمل داده و کد اختیاری کشف خطا (CRC) قرار گرفته است. در فریمهای کنترلی به بخش داده (Payload) نیازی نیست (مثلاً در فریمهایی که تقاضای برش زمانی می دهند). کد کشف خطا (Checksum) نیز اختیاری است زیرا در لایه فیزیکی، عملیات تصحیح خطا انجام می گیرد و همچنین قرار نیست فریمهای بی درنگ [در صورت خراب شدن] از نو ارسال شوند. اگر قرار نباشد که فریمی از نو ارسال شود چرا با افزودن کد کشف خطا، خود را به زحمت بیندازیم.



شکل ۴-۳۴. (الف) قالب عمومی فریم (ب) فریم تقاضای پهنای باند.

فیلدهای سرآیند شکل ۴-۳۴-الف را به اختصار معرفی می‌کنیم: بیت EC مشخص می‌کند که آیا بخش داده، رمزنگاری شده است؟ فیلد Type، نوع فریم را تعیین می‌کند، بالاخص آنکه آیا داده‌ها به صورت مجموعه‌ای از قطعات یا آنکه به صورت یکجا ارسال می‌شود. فیلد CI مشخص می‌کند که آیا فیلد کشف خطا (Checksum) در پایان فریم وجود دارد یا خیر. فیلد EK مشخص‌کننده آنست که کدامیک از کلیدهای رمزنگاری به کارگرفته شده است. (بشرط آنکه رمزنگاری انجام و چندین کلید تعریف شده باشد). فیلد Length طول کل فریم را با احتساب سرآیند تعیین می‌کند. فیلد Connection Identifier مشخص می‌کند که فریم متعلق به کدام «اتصال» است. در آخر، فیلد Header CRC در برگرفته‌ی کد کشف خطائی است که فقط از بخش سرآیند و با استفاده از چندجمله‌ای $x^8 + x^2 + x + 1$ استخراج می‌شود.

نوع دوم سرآیند که فقط برای فریمهائی که پهنای باند تقاضا می‌دهند تعریف شده، در شکل ۴-۳۴-ب نشان داده شده است. این سرآیند به جای بیت صفر با بیت ۱ شروع شده ولیکن ترکیب کلی آن با ترکیب فریم عمومی [شکل ۴-۳۴-الف] مشابه است، با این تفاوت که بایت دوم و سوم این فریم یک عدد ۱۶ بیتی را تشکیل داده و محتوای آن مشخص می‌کند که برای مبادله تعداد مشخص بایت، به چه اندازه پهنای باند نیاز است. فریم تقاضای پهنای باند دارای بخش حمل داده (Payload) و کد کشف خطا برای کل فریم نیست. اگر چه می‌توان مفصلاً در خصوص 802.16 شرح داد ولی در اینجا به بحث خاتمه می‌دهیم. برای آگاهی بیشتر مستقیماً به استاندارد آن مراجعه کنید.

۶-۴ بلوتوث (Bluetooth)

در سال ۱۹۹۸ شرکت ال.ام. اریکسون علاقمند شد تا گوشی تلفنهای همراه تولیدی او بتوانند بصورت بی‌سیم به ابزارهای دیگر (مثل PDA) وصل شوند. اریکسون و چهار شرکت دیگر (آی.بی.ام. ایتل. نوکیا و توشیبا) یک «گروه SIG»^۱ تشکیل دادند تا استاندارد بی‌سیم برای اتصال ابزارهای مخابراتی/رایانه‌ای و ابزارهای جانبی آنها طراحی کنند که بردی کوتاه، مصرف توان پائین و قیمتی ارزان داشته باشد. نام این پروژه، بلوتوث انتخاب شد که برگرفته از نام «هرالد بلاتاند دوم» (مشهور به Bluetooth)، یکی از پادشاهان وایکینگ است (۹۸۱-۹۴۰) که دانمارک و نروژ را با هم متحد کرد (البته با زور و بدون کابل!!!)

اگر چه تفکر اصلی، رهایی از سُرکابلهای مابین دستگاه‌های دیجیتالی بود ولی به سرعت در حوزه‌های دیگر نیز گسترش یافت و به تدریج در حیطه شبکه‌های محلی بی‌سیم نیز وارد شد. اگرچه گسترش و رشد این استاندارد، روز بروز کاربرد آنرا بیشتر می‌کرد ولی در عوض چالشهایی بین این استاندارد و 802.11 پدید آورد. نقطه شدت این چالش آنجاست که این دو سیستم از لحاظ الکتریکی با یکدیگر تداخل فرکانسی دارند. اشاره به این نکته مهم است که شرکت هیولت پاکارد چندین سال قبل از آن شبکه‌ای مبتنی بر نور مادون قرمز برای وصل بی‌سیم دستگاه‌های جانبی کامپیوتر عرضه کرده بود، ولی استقبال چندانی از آن نشد.

فارغ از همه اینها، در ژولای ۱۹۹۹ گروه طراح بلوتوث، مشخصات هزار و پانصد صفحه‌ای از نسخه ۱ آن یعنی V1.0 را منتشر نمود. به فاصله کوتاهی، گروه استانداردسازی IEEE که در اندیشه تدوین استاندارد 802.15 برای «شبکه‌های شخصی بی‌سیم» بودند مستندات استاندارد بلوتوث را به عنوان مبنای کار خود برگزیدند و شروع به پالایش و تکمیل آن نمودند. اگر چه استانداردسازی چیزی که مشخصات تفصیلی و مشروح آن در اختیار است و پیاده‌سازیهای متنوع و ناسازگار ندارد (که نیاز به یکنواخت‌سازی و هماهنگی داشته باشد) عجیب به نظر می‌رسد

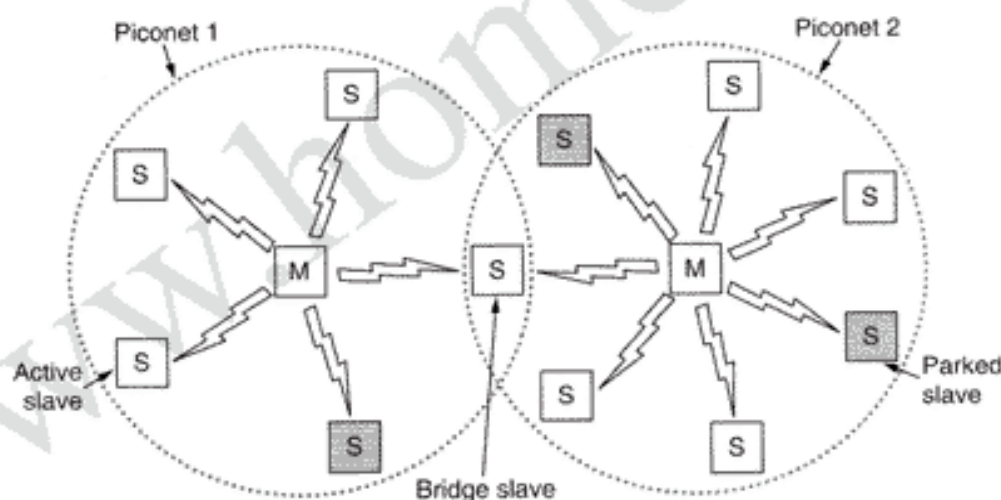
۱. SIG به معنای گروهی با گرایش خاص یا همان کنسرسیوم است.

ولی تاریخ نشان داده که وجود یک «استاندارد باز» که توسط سازمانی بی طرف مثل IEEE تدوین و مدیریت می شود عموماً کاربری یک تکنولوژی را ترویج و ترغیب خواهد کرد. اگر بخواهیم اندکی دقیقتر سخن بگوئیم باید اشاره کنیم که توصیف استاندارد بلوتوث برای سیستمی کامل تدوین شده که از لایه فیزیکی تا لایه کاربرد را در بر می گیرد درحالیکه کمیته IEEE 802.15 فقط لایه های فیزیکی و پیوند داده را استانداردسازی کرده و باقیمانده پشته پروتکلی خارج از برنامه این استاندارد است.

هرچند IEEE اولین «استاندارد شبکه شخصی»^۱ (PAN) را در سال ۲۰۰۲ با عنوان 802.15.1 به تصویب رساند ولی هنوز کنسرسیوم بلوتوث فعال و سرگرم بهبود و توسعه آنست. اگر چه نسخه استاندارد عرضه شده توسط کنسرسیوم بلوتوث و IEEE یکی نیستند ولی انتظار می رود بزودی به یک استاندارد واحد همگرا شوند.

۴-۱۶ معماری بلوتوث

اجازه بدهید بررسی سیستم بلوتوث را با مروری سریع بر دستاوردها و اهداف آن آغاز نمائیم. واحد پایه در سیستم بلوتوث یک «پیکونت» (Piconet) است که از یک «گره اصلی» (Master Node) و حداکثر هفت «گره پیرو فعال» (Active Slave Node) به فاصله حداکثر ده متر، تشکیل شده است. در یک فضای بزرگ و واحد می توان چندین پیکونت داشت و حتی می توان آنها را از طریق یک گره که نقش پل (Bride) ایفاء می کند به هم متصل کرد (به شکل ۴-۳۵ نگاه کنید). به مجموعه ای از پیکونتهای متصل بهم اصطلاحاً Scatternet (شبکه متفرق/پراکنده) گفته می شود.



شکل ۴-۳۵. دو پیکونت می توانند با اتصال بهم یک Scatternet تشکیل بدهند.

در یک پیکونت علاوه بر هفت گره فعال پیرو، می تواند تا ۲۵۵ گره غیرفعال وجود داشته باشد. اینها دستگاه هایی هستند که گره اصلی آنها را در حالت استراحت و کم توان وارد کرده تا مصرف باتری آن کاهش یابد. یک ایستگاه در حالت غیر فعال هیچ کاری نمی تواند انجام بدهد به جز آنکه به سیگنال فعال سازی خود یا سیگنال Beacon که از گره اصلی می رسد، پاسخ بدهد. به غیر از این حالات، دو حالت میانی در مصرف توان به نامهای حالت Hold و Sniff نیز وجود دارد که در اینجا بدان نخواهیم پرداخت.

دلیل طراحی Master/Slave (اصلی/پیرو) آن بود که طراحان آن در نظر داشتند قیمت کل سیستم بلوتوث پیاده سازی شده بر روی تراشه، زیر پنج دلار باشد. نتیجه این تصمیم گیری آنست که گره های پیرو [مثل صفحه

کلیدها، موس، چاپگر] تقریباً غیر هوشمند و ساده هستند و اساساً آنچه را که گره اصلی (Master) به آنها دستور بدهد اجرا می کنند. یک پیکونت سیستمی مبتنی بر TDM متمرکز (Centralized TDM) است که در آن هسته مرکزی (یعنی گره اصلی یا Master) بر سیگنال ساعت نظارت دارد و تعیین می کند که چه دستگاهی و در کدام برش زمانی (Slot) مخابره داشته باشد. تبادل اطلاعات صرفاً بین گره مرکزی و گره های پیرو انجام می شود و ارتباط مستقیم دو گره پیرو [مثلاً دو صفحه کلید یا دو چاپگر] ممکن نیست.

۲-۶-۴ کاربردهای بلوتوث

بیشتر پروتکل های شبکه فقط کانالی را بین چند مولفه مخابراتی، سازماندهی و ایجاد می کنند و اجازه می دهند طراحان برنامه های کاربردی به پیاده سازی هر آنچه که مورد نیاز است بپردازند. به عنوان مثال 802.11 مشخص نکرده که کاربران باید صرفاً از کامپیوتر کیفی خود برای خواندن ایمیل یا جستجو در وب یا هر چیز دیگری بهره بگیرند. برعکس، در تشریح بلوتوث نسخه ۱.۱ از ۱۳ کاربرد مختلف که باید از آنها پشتیبانی شود، نام برده شده و برای هر یک، پشته پروتکلی متفاوتی ارائه گردیده است. متأسفانه این راهکار به پیچیدگی بسیار زیاد منتهی می شود و ما از آن صرف نظر خواهیم کرد. این سیزده کاربرد که «پرو فایل» نام گرفته اند در شکل ۴-۳۶ فهرست شده اند. با نگاهی اجمالی به پرو فایلها ممکن است به آنچه که کنسرسیوم بلوتوث در پی انجام آن بوده بیشتر پی ببریم.

نام پرو فایل	عملکرد
Generic access	پروسیجریهایی برای مدیریت لینک
Service discovery	پروتکلی برای کشف سرویسهای عرضه شده
Serial port	جایگزینی برای کابل معمولی پورت سریال
Generic object exchange	مدل ارتباطی بین سرویس دهنده و مشتری برای جابجایی (انتقال) اشیاء را تعریف می کند.
LAN access	پروتکل ارتباطی بین کامپیوتر همراه و شبکه محلی ثابت (با کابل سیمی)
Dial-up networking	امکان برقراری تماس یک کامپیوتر کیفی را از طریق تلفن همراه فراهم می آورد.
Fax	امکان ارتباط بین یک دستگاه دورنگار بی سیم و تلفن همراه را فراهم می آورد.
Cordless telephony	ارتباط بین یک دستگاه گوشی تلفن بی سیم و ایستگاه ثابت و محلی آن را برقرار می کند.
Intercom	امکانانی برای واکای تاکی دیجیتالی
Headset	امکان ارتباط از طریق هندزفری (Handsfree) را فراهم می آورد.
Object push	روشی برای مبادله اشیاء ساده
File transfer	عرضه کننده امکانات عمومی بیشتر جهت انتقال فایل
Synchronization	امکان سنکرون سازی داده های یک PDA با کامپیوتری دیگر را فراهم می آورد.

شکل ۴-۳۶. پرو فایل های بلوتوث.

«پرو فایل عمومی دسترسی» (Generic Access) حقیقتاً یک برنامه کاربردی نیست بلکه بیشتر یک زیربناست که بر اساس آن برنامه های کاربردی حقیقی ساخته و پیاده می شوند. وظیفه اصلی آن ارائه تمهیداتی است که بتوان بین گره اصلی (Master) و گره های پیرو (Slave) یک کانال مطمئن برقرار و آنرا حفظ کرد. «پرو فایل تشخیص خدمات» (Service Discovery) که آنهم تقریباً عمومی و کلی است توسط دستگاه ها برای آگاهی از خدماتی که دیگر دستگاه ها ارائه می دهند، استفاده می شود. تمام دستگاه های مبتنی بر بلوتوث موظف به پیاده سازی این دو پرو فایل هستند. بقیه پرو فایلها اختیاریند.

«پرو فایل درگاه سریال» (Serial Port) یک پروتکل انتقال است که بقیه پرو فایلها از آن بهره می گیرند. این

پرو فایل یک درگاه سریال را شبیه‌سازی می‌کند و بطور خاص برای کاربردهای قدیمی که به خط سریال نیاز دارند، سودمند است.

«پرو فایل عمومی مبادله شیء» (Generic Object Exchange) یک ارتباط مبتنی بر مدل مشتری/سرویس‌دهنده، برای انتقال داده‌ها تعریف کرده است. اگرچه همیشه مشتری، آغازکننده عملیات است ولیکن یک گره پیرو می‌تواند هم سرویس‌دهنده و هم مشتری باشد. همانند پرو فایل «درگاه سریال» این پرو فایل نیز زیربنای دیگر پرو فایلهاست.

گروه سه تایی پرو فایلهای بعدی به منظور کاربردهای شبکه‌ای (Networking) تعریف شده‌اند. «پرو فایل دسترسی به LAN» اجازه می‌دهد که یک دستگاه مبتنی بر بلوتوث به یک شبکه ثابت متصل شود. این پرو فایل رقیب مستقیم 802.11 است. «پرو فایل شبکه مبتنی بر شماره‌گیری» (Dialup) انگیزه اصلی کل این پروژه بوده است. این پرو فایل اجازه می‌دهد که یک کامپیوتر کیفی بتواند به یک تلفن همراه که دارای مودم داخلی بی سیم است متصل شود. «پرو فایل دورنگار» (Fax) شبیه به پرو فایل شماره‌گیری است با این تفاوت که اجازه می‌دهد ماشینهای دورنگار بی سیم از طریق یک دستگاه تلفن همراه و بدون نیاز به سیم، اقدام به ارسال یا دریافت دورنگار کنند.

سه پرو فایل بعدی در خصوص تلفن کاربرد دارند. پرو فایل «تلفن بی سیم» (Cordless Telephony) راهی را برای اتصال گوشی یک تلفن بی سیم به ایستگاه ثابت است. در حال حاضر تلفنهای بی سیم خانگی را نمی‌توان به عنوان تلفن همراه به کارگرفت ولی در آینده شاید تلفن بی سیم و تلفن همراه در هم ادغام شوند. «پرو فایل Intercom» این امکان را فراهم می‌کند تا دو تلفن، شبیه به «واکی تاکی» (Walkie/Talkie) بهم متصل گردند. نهایتاً «پرو فایل گوشی - Headset-» امکان ارتباط بی سیم بین گوشی و ایستگاه ثابت (مثل هندزفری - Hands Free-) را فراهم می‌کند که به عنوان مثال برای صحبت با تلفن در حین رانندگی مفید است.

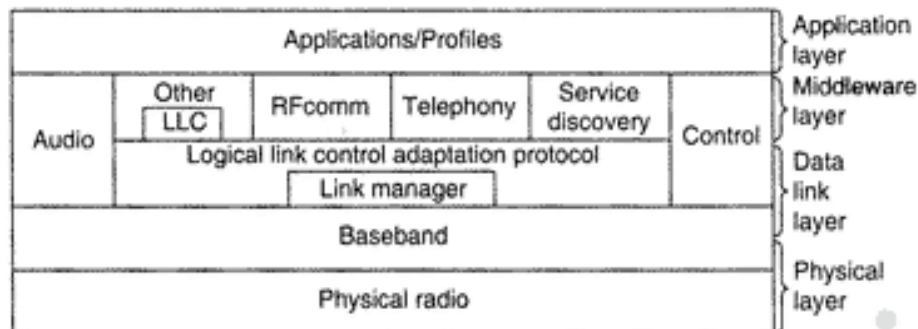
سه پرو فایل باقیمانده برای مبادله اشیاء بین دو ابزار تعریف شده است. اشیاء می‌توانند فایل‌های داده، تصویر یا کارتهای تجاری باشند. «پرو فایل سنکرواسیون»، برای بار کردن داده در درون کامپیوتر کیفی یا PDA (کامپیوترهای دستی) در حین ترک منزل و جمع آوری اطلاعات پس از برگشت، مفید است.

آیا واقعاً نیاز بوده که تمام این کاربردها به تفصیل تحلیل شوند و برای هر کدام پشته پروتکلی متفاوتی تعریف گردد؟ شاید نه! اما بخشهای متفاوت این استاندارد را گروه‌های کاری مختلف طراحی کرده‌اند و چون هر یک از این گروه‌ها بر روی نیازهای خاص خود متمرکز بودند، در نتیجه هر یک پرو فایل موردنظر خود را پدید آوردند. برای توجیه این موضوع به قانون کانوی (Conway's Law) بیندیشید. (در یکی از شماره‌های مجله Datamation در آوریل ۱۹۶۸، ملوین کانوی مشاهدات خود را بدین نحو منتشر کرد که اگر n شخص را به نوشتن یک کامپایلر بگمارید، آنچه که بدست خواهید آورد یک کامپایلر n -pass است [یعنی کامپایلری که در آن تعداد مراحل ترجمه یک برنامه، n گذر می‌باشد]. عبارت عام ساختار نهانی یک نرم‌افزار آینه تمام‌نمای ترکیب گروهی است که آن را تولید کرده‌اند. شاید می‌شد که به جای سیزده پشته پروتکلی به دو پشته کلی بسنده کرد: یکی برای انتقال فایل و دیگری برای انتقال بی‌درنگ جریان اطلاعات.

۳-۶-۴ پشته پروتکلی بلوتوث

استاندارد بلوتوث پروتکل‌های متعددی دارد که بطور ناموزون در چند لایه گروه‌بندی شده‌اند. ساختار لایه‌ها از مدل OSI، مدل TCP/IP، مدل 802 یا هر مدل شناخته شده دیگر تبعیت نمی‌کند. با این وجود IEEE در حال اصلاح بلوتوث است تا با مدل 802 سازگارتر شود. معماری پروتکل بلوتوث که توسط کمیته 802 اصلاح شده، در شکل ۳۷-۴ مشاهده می‌شود.

لایه زیرین، «لایه رادیو فیزیکی» است که تقریباً متناظر با لایه فیزیکی از مدل OSI یا مدل 802 می‌باشد. این



شکل ۴-۳۷. نسخه ۸۰۲.۱۵ از معماری پروتکل بلوتوث.

لایه با انتقال رادیویی و مدولاسیون سرکار دارد. بسیاری از ملاحظات که در طراحی این لایه باید مورد توجه قرار می‌گرفت آن بود که سیستم ارزان قیمت باشد و بطور انبوه در بازار عرضه شود.

«لایه باند پایه» (Baseband) از جهاتی شبیه به زیرلایه MAC است ولیکن مولفه‌هایی از لایه فیزیکی را نیز در بر می‌گیرد. این لایه با مسائلی مثل چگونگی نظارت گره اصلی (Master) بر برشهای زمانی و چگونگی گروه‌بندی این برشهای زمانی در قالب فریمها، سرکار دارد.

سپس لایه‌ای شامل یک گروه از پروتکل‌های مرتبط با هم، تعریف شده است. مدیر لینک (Link Manager) عملیات ایجاد کانالهای منطقی بین دستگاه‌ها، شامل «مدیریت توان مصرفی»، «احراز هویت» (Authentication) و «کیفیت خدمات» (QoS) را بر عهده دارد. «پروتکل تطبیق کنترل لینک منطقی» (که اغلب L2CAP گفته می‌شود) وظیفه دارد لایه‌های بالایی را از درگیری با جزئیات ارسال، راحت کند. این لایه مشابه با استاندارد زیرلایه LLC ۸۰۲ است ولی از لحاظ مسائل فنی با آن متفاوت است. دو پروتکل «کنترل» و «صدا» همانگونه که از نامشان بر می‌آید با مسائل انتقال صدا و عملیات کنترل سروکار دارند. برنامه‌های کاربردی می‌توانند بدون نیاز به L2CAP، مستقیماً این دو پروتکل را به خدمت بگیرند.

لایه بعدی یک لایه میانی است (Middleware) و تلفیقی از پروتکل‌های متفاوت را در بر می‌گیرد. در این لایه از پروتکل IEEE ۸۰۲ LLC، بمنظور سازگاری با دیگر شبکه‌های سری ۸۰۲ استفاده شده است. پروتکل‌های RFcomm، Telephony و Service Discovery صرفاً مرتبط با بلوتوث هستند. RFcomm (Radio Frequency Communication)، پروتکلی جهت شبیه‌سازی استاندارد درگاه سریال (Serial port) است که در تمام PCها از آن برای اتصال صفحه کلید، موس، مودم و امثال آن استفاده می‌شود. این پروتکل برای آن طراحی شده تا بتوان از دستگاه‌های قدیمی به‌سبب استفاده کرد. «پروتکل تلفنی» (Telephony) پروتکلی بی‌درنگ است که برای سه پروفایل مبتنی بر انتقال صدا بکار می‌آید. این پروتکل همچنین تنظیم و قطع ارتباط را بر عهده دارد. نهایتاً پروتکل «تشخیص خدمات» (Service Discovery) برای کشف و تشخیص انواع خدماتی که درون شبکه عرضه می‌شود، کاربرد دارد.

بالاترین لایه، محل قرار گرفتن انواع برنامه‌های کاربردی و پروفایلها است. این لایه برای انجام کار از خدمات پروتکل‌های موجود در لایه‌های زیر بهره می‌گیرد. هر برنامه کاربردی، زیرمجموعه‌ای از پروتکل‌های مختص به خود را به خدمت می‌گیرد. ابزارهای ویژه‌ای مثل گوشی بی‌سیم (Headset) بسته به نوع برنامه کاربردی آنها، فقط به برخی از پروتکل‌ها نیازمندند.

در بخشهای بعدی سه لایه پائینی از پشته پروتکلی بلوتوث را بررسی خواهیم کرد چرا که تقریباً متناظر با زیرلایه‌های فیزیکی و MAC است.

۴-۶-۴ لایه رادیویی در بلوتوث

لایه رادیویی بیتها را از گره اصلی به گره پیرو و بالعکس، منتقل می کند. این لایه، سیستمی با توان کم و برد ده متر است که در باند فرکانسی 2.4GHz ISM عمل می کند. این باند به ۷۹ کانال یک مگاهرتزی تقسیم می شود. مدولاسیون به کاررفته FSK (Frequency Shift Keying) و هر هرتز (هر سیکل) معادل یک بیت است که جمعاً نرخ یک مگابیت بر ثانیه را در اختیار می گذارد ولی بیشتر این پهنای باند به دلیل سربار تلف می شود. برای تخصیص مناسب این کانالها از روش پرش فرکانس در طیف گسترده (Spread Spectrum) با نرخ پرش 1600 hops/sec و Dwell time معادل ۶۲۵ میکروثانیه بهره گرفته شده است.

چون بلوتوث و 802.11، هر دو در باند 2.4GHz ISM و دقیقاً در همان ۷۹ کانال کار می کنند لذا با یکدیگر تداخل فرکانسی خواهند داشت. از آنجایی که پرش فرکانس در بلوتوث سریعتر از 802.11 است لذا دستگاه های مبتنی بر بلوتوث به احتمال بیشتری در انتقال 802.11 اختلال خواهد کرد تا 802.11 در بلوتوث! چون 802.11 و 802.15 هر دو استانداردهای IEEE هستند لذا IEEE به دنبال راه حلی برای این مشکل می گردد ولی حل این مشکل چندان ساده نیست چرا که هر دو سیستم، بدلیل مشابهی از این باند فرکانسی بهره گرفته اند: زیرا، برای استفاده از این باند فرکانسی، به اخذ هیچ مجوزی نیاز نیست. استاندارد 802.11a از باند دیگر ISM (باند 5GHz) استفاده می کند ولیکن برد کمتری نسبت به 802.11b دارد (بدلیل ماهیت فیزیکی امواج رادیویی این باند) لذا استفاده از 802.11a راه حل مناسبی نخواهد بود. برخی از شرکتها این مشکل را با ممنوعیت استفاده از بلوتوث حل کرده اند. راه حل بازاری این مشکل آنست که صبر کنیم تا عاقبت شبکه ای که از لحاظ اقتصادی و سیاسی جایگاه مستحکم تری در بازار پیدا کرد، از طرف مقابل بخواهد تا استاندارد خود را برای حل مشکل تداخل، اصلاح نماید. در این خصوص مطالبی در مرجع (Lansford et al., 2001) ارائه شده است.

۴-۶-۴ لایه باند پایه در بلوتوث

لایه باند پایه شبیه ترین بخش بلوتوث با زیرلایه MAC است. این لایه، دنباله بیتهای خام را به فریمها تبدیل می کند و بدین منظور چندین قالب مهم فریم تعریف نموده است. در ساده ترین حالت، گره اصلی در هر پیکونت دنباله ای از برشهای زمانی ۶۲۵ میکروثانیه ای (Time Slot) تولید می کند، با این توصیف که ارسال داده های گره اصلی در برشهای زمانی با شماره زوج انجام می شود و گره های پیرو (Slaves) در برشهای زمانی فرد ارسال می نمایند. این روش مشابه با روش تسهیم زمانی (TDM) معمولی است که در آن، گره اصلی نیمی از برشهای زمانی را در اختیار دارد و بقیه گره ها (حداکثر هفت گره) در نیم دیگر سهیم هستند. ارسال هر فریم می تواند ۱، ۳ یا ۵ برش زمانی طول بکشد.

در هر پرش فرکانسی ۲۵۰ تا ۲۶۰ میکروثانیه طول خواهد کشید تا مدار رادیویی بتواند پایدار شود. پایداری سریعتر نیز ممکن است ولی هزینه پیاده سازی بیشتری دارد. برای فریمهای که فقط به یک برش زمانی نیاز دارند پس از هر پرش فرکانسی، ۳۶۶ بیت از کل ۶۲۵ بیت باقی خواهد ماند. از این مقدار ۱۲۶ بیت به «کد دسترسی» (Access Code) و «سرآیند» اختصاص دارد و ۲۴۰ بیت، برای داده ها باقی می ماند. وقتی پنج برش زمانی به هم ملحق می شوند [برای ارسال فریمهایی به طول ۵ برش] تنها به یک زمان پایداری نیاز خواهد بود و طبقاً زمان کوتاهتری برای زمان پایداری، تلف می شود و $5 \times 625 = 3125$ بیت در پنج برش زمانی ارسال می گردد که از این مقدار ۲۷۸۱ بیت برای ارسال داده در اختیار «لایه باند پایه» خواهد بود. بنابراین در بلوتوث فریمهای طولانی کارآمدتر از فریمهای کوچک هستند.

هر فریم بر روی یک کانال منطقی که اصطلاحاً «لینک بین گره اصلی و گره پیرو» نام دارد، ارسال خواهد شد. دو نوع لینک وجود دارد: لینک اول ACL است (Asynchronous Connection-Less) که برای ارسال داده ها در

برشهای زمانی نامنظم کاربرد دارد. این داده‌ها از لایه L2CAP در سمت فرستنده تولید و در سمت گیرنده تحویل لایه L2CAP می‌شوند. ترافیک داده‌های ACL مبتنی بر روش «بیشترین تلاش» (Best Effort) ارسال می‌شود ولی هیچ تضمینی در تحویل آنها نیست. فریم‌ها می‌توانند گم شده یا از بین بروند، بدون آنکه ارسال مجدد شوند. هر گره پیرو فقط می‌تواند یک لینک ACL با گره اصلی داشته باشد.

لینک دیگر، Synchronous Connection Oriented (SCO) نام دارد که برای ارسال داده‌های بی‌درنگ، مثل ارتباط تلفنی کاربرد دارد. این نوع کانال با تخصیص برشهای زمانی مشخص در هر دو جهت، ایجاد می‌شود. بدلیل آنکه لینکهای SCO نسبت به زمان حساس هستند فلذا فریمهای ارسالی بر روی این لینک هرگز ارسال مجدد (Retransmit) نخواهند شد، در عوض از روش «تصحیح مستقیم خطا» استفاده شده تا اطمینان بیشتری داشته باشد. [ارسال مجدد فریمهای خراب بر عهده لایه بعدی است و در این لایه در صورت امکان - به تصحیح خطا بسنده می‌شود. -] هر گره پیرو می‌تواند حداکثر سه لینک SCO با گره اصلی داشته باشد. هر لینک SCO می‌تواند یک کانال صدا مبتنی بر PCM با نرخ 64000 بیت بر ثانیه را حمل کند.

۴-۶-۴ لایه L2CAP در بلوتوث

لایه L2CAP سه دسته عملیات مهم را بر عهده دارد: اول آنکه بسته‌هایی با طول حداکثر ۶۴ کیلوبایت را از لایه‌های بالائی پذیرفته و آنها را جهت انتقال، به فریمهای کوچکتری می‌شکند. در سمت مقابل این فریمها مجدداً به بسته اصلی بازسازی خواهند شد.

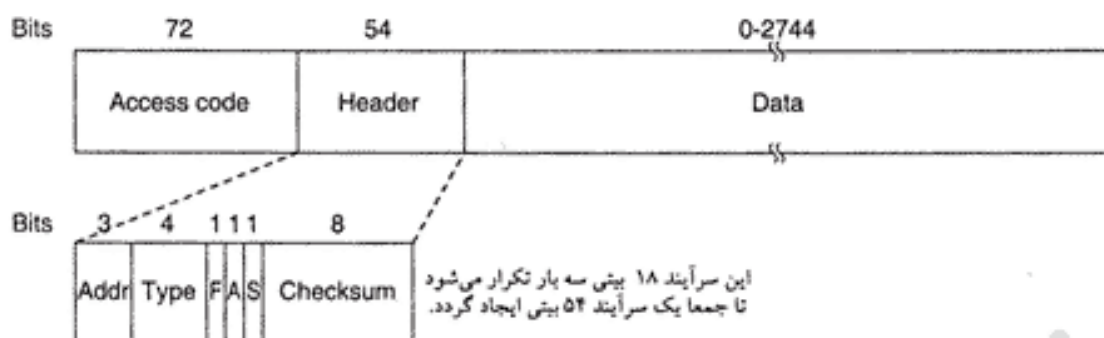
دوم آنکه این لایه عمل جمع‌آوری و توزیع بسته‌هایی که از چندین مبدا آمده [یا به چندین مقصد می‌روند] را بر عهده دارد. وقتی یک بسته بازسازی می‌شود، لایه L2CAP تعیین خواهد کرد که باید به کدام پروتکل در لایه بالاتر (مثلاً RFcomm یا Telephony) تحویل شود.

سوم آنکه این لایه، عملیات تامین «کیفیت خدمات» (QoS) را بر عهده دارد، چه در هنگام ایجاد لینک و چه در خلال عملکرد طبیعی. همچنین در زمان ایجاد لینک، بر سر اندازه حداکثر و مجاز طول داده، توافق صورت می‌گیرد تا ابراز هائی با طول بسته بزرگ از ارسال چنین بسته‌ای به ابزار هائی با طول بسته کوچک اجتناب کنند. از آنجایی که تمام ابزارهای مبتنی بر بلوتوث نمی‌توانند بسته‌هایی با طول ۶۴ کیلوبایت را بپذیرند فلذا به این ویژگی نیاز است.

۴-۶-۴ ساختار فریم در بلوتوث

چندین نوع قالب فریم در بلوتوث وجود دارد که مهمترین آنها در شکل ۴-۳۸ نشان داده شده است. این فریم با فیلد «کد دسترسی» (Access Code) شروع می‌شود که عموماً هویت یک گره اصلی (Master) را مشخص خواهد کرد تا بدینگونه یک گره پیرو که در برد رادیویی دو گره اصلی قرار دارد، گیرنده حقیقی ترافیک داده‌ها را مشخص نماید. سپس یک سرآیند ۵۴ بیتی آمده که شامل فیلدهای معمولی زیرلایه MAC است. سپس فیلد داده قرار گرفته که حداکثر ۲۷۴۴ بیت را (برای انتقال در پنج برش زمانی) در بر می‌گیرد. در فریمهائی که تنها در یک برش زمانی ارسال می‌شوند، قالب فریم همین است با این تفاوت که فیلد داده آنها حداکثر ۲۴۰ بیت است.

حال اجازه بدهید به فیلدهای سرآیند، نگاهی سریع بیندازیم. «فیلد آدرس» هویت گیرنده فریم را از بین هشت دستگاه فعال در هر پیکونت مشخص می‌کند. «فیلد Type»، اولاً نوع فریم را (از بین انواع ACL، SCO، POLL، یا NULL)، ثانیاً روش تصحیح خطای داده‌ها را و ثالثاً تعداد برشهای زمان که ارسال فریم جاری بدان نیاز دارد را مشخص می‌نماید. «بیت Flow» توسط گره‌های پیرو و زمانی تنظیم [فعال] می‌شود که بافر آنها پر شده باشد و نتوانند داده بیشتری دریافت کنند. این بیت، شکل ابتدائی کنترل جریان داده‌ها، به حساب می‌آید. «بیت



شکل ۳۸-۴. قالب کلی فریم داده در بلوتوث.

Acknowledgement بدین منظور است تا دریافت صحیح یک فریم از طرف مقابل، در فریم ارسالی جاسازی و اعلام شود [یعنی فرآیند Piggybacking]. «بیت Sequence» برای شماره گذاری فریمهاست تا بسته های تکراری کشف شوند. در بلوتوث پروتکل ارسال مجدد، روش «توقف و انتظار» (Stop & wait) است و طبقاً یک بیت برای شماره گذاری فریمها کفایت می کند. در ادامه «فیلد هشت بیتی Checksum» برای کشف خطای احتمالی در سرآیند تعریف شده است. کل این سرآیند ۱۸ بیتی سه بار تکرار می شود تا سرآیند ۵۴ بیتی نشان داده شده در شکل ۳۸-۴ بوجود آید. در سمت گیرنده، با یک مدار ساده سه نسخه تکراری هر بیت بررسی می شود. اگر هر سه بیت مثل هم بودند، آن بیت پذیرفته می شود در غیر اینصورت، بیتی که بیشترین تکرار را دارد قبول می شود. بدین ترتیب، برای ارسال یک سرآیند ۱۰ بیتی ظرفیتی معادل ۵۴ بیت صرف می شود. دلیل آن این بوده که برای ارسال مطمئن داده ها در محیطی سرشار از نویز و با استفاده از ابزاری از اوزان قیمت و توانی ناچیز (2.5mW) و قدرت پردازش پائین، به افزونگی بسیار زیادی نیاز خواهد بود.

برای فیلد داده در فریمهای ACL، قالبهای متفاوتی تعریف شده است. فریمهای SCO ساده تر هستند: فیلد داده همیشه ۲۴۰ بیتی است. سه گزینه دیگر نیز تعریف شده که در آنها مقدار واقعی داده ها ۸۰، ۱۶۰ یا ۲۴۰ بیتی است و باقیمانده بیتها برای تصحیح خطا به کار می آیند. در مطمئن ترین نسخه (یعنی داده ۸۰ بیتی)، بخش داده همانند سرآیند، سه بار متوالی تکرار می شود.

از آنجایی که گرو پیرو (Slave) تنها می تواند از برشهای زمانی با شماره فرد استفاده نماید فلذا در هر ثانیه ۸۰۰ برش زمانی بدست خواهد آورد. (همینطور گره اصلی) بدین ترتیب با فیلد داده ۸۰ بیتی، ظرفیت کانال از سمت گره پیرو به سمت گره اصلی معادل ۶۴۰۰۰ بیت بر ثانیه خواهد بود (ظرفیت کانال از گره اصلی به گره پیرو نیز ۶۴۰۰۰ بیت است) لذا این کانال دقیقاً برای یک کانال صوتی دو طرفه مبتنی بر PCM کافی است. (دلیل انتخاب نرخ 1600 Hops/sec برای تغییر فرکانس همین بوده است). این اعداد و ارقام بدین معنا هستند که یک کانال صوتی دو طرفه PCM با نرخ ۶۴۰۰۰ بیت بر ثانیه، در حالت مطمئن [یعنی وقتی داده های ۸۰ بیتی با سه بار تکرار ارسال می شوند]، کل پهنای باند موجود در پیکونت را (علیرغم پهنای باند 1 Mbps آن)، اشباع خواهد کرد. با گزینه نامطمئنتر (یعنی ۲۴۰ بیت در هر برش زمانی بدون هیچگونه افزونگی یا تکرار) می توان از حداکثر سه کانال صوتی همزمان حمایت کرد و به همین دلیل حداکثر سه لینک SCO برای هر گره پیرو مجاز شمرده شده است.

مطالب فراوانتری در خصوص بلوتوث می توان گفت که از حوصله این کتاب خارج است. برای آگاهی بیشتر به مراجع زیر مراجعه نمایید.

Bhagwat, 2001; Bisdikian, 2001; Bray and Sturman, 2002; Haartsen, 2000; Johansson et al., 2001; Miller and Bisdikian, 2001; Sairam et al., 2002)

۷-۴ هدایت در سطح لایه پیوند داده‌ها (Data Link Layer Switching)

بسیاری از سازمانها دارای LANهای متعددی هستند و تمایل دارند آنها را به هم متصل کنند. شبکه‌های محلی (LAN) را می‌توان از طریق دستگاه‌هایی که در لایه پیوند داده‌ها عمل می‌کنند و «پل» (Bridge) نامیده می‌شوند به هم متصل نمود. «پلها» برای مسیریابی و هدایت داده‌ها، آدرسهای «لایه پیوند داده‌ها» را بررسی می‌نمایند. از آنجایی که قرار نیست محتوای فیلد داده (از فریم‌هایی که باید هدایت شوند) بررسی گردد لذا این فریم‌ها می‌توانند بسته‌های IPv4 (که اکنون در اینترنت به کار می‌رود)، IPv6 (که در آینده در اینترنت به کار گرفته خواهد شد)، بسته‌های OSI، ATM، AppleTalk یا هر نوع بسته دیگر را در خود حمل کنند. برخلاف پل، «مسیریابها» آدرس درون بسته‌ها را بررسی کرده و بر این اساس، آنها را هدایت (مسیریابی) می‌کنند. اگرچه این تعریف تمایز بین «مسیریاب» و «پل» را تعیین می‌کند ولی پیشرفتهای جدیدی مثل ابداع شبکه LAN مبتنی بر سوئیچ (Switched Ethernet)، آب را گل کرده و به نحوی که در بخشهای آتی بدانها خواهیم پرداخت این تمایز را ابهام‌آلود کردند! در بخشهای بعدی به پلها و سوئیچها و بالاخص آنهایی که برای اتصال شبکه‌های محلی 802 به کار می‌آیند، نگاهی خواهیم انداخت. برای بررسی دقیق پلها، سوئیچها و عناوین مهم در این خصوص، به مراجع (Perlman, 2000) مراجعه نمایید.

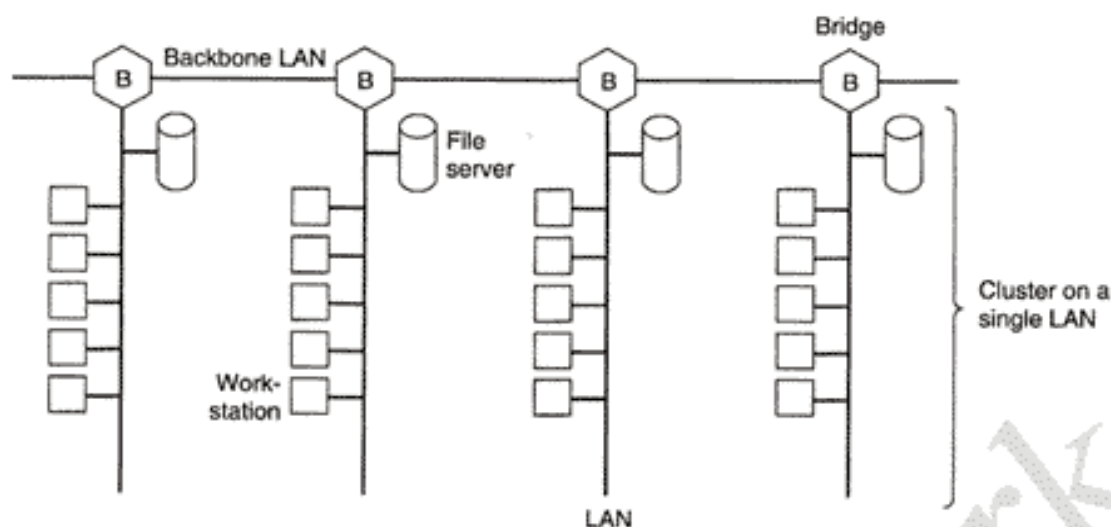
قبل از پرداختن به تکنولوژی پل، بررسی شرایطی که در آن، استفاده از پلها سودمند است، خالی از لطف نخواهد بود. شش دلیل ارائه می‌کنیم که چرا یک سازمان واحد، ممکن است دارای چندین LAN باشد.

اول آنکه بسیاری از دانشگاه‌ها و بخشهای مختلف شرکتها، LAN مختص به خود را دارند تا بتوانند کامپیوترهای شخصی، ایستگاه‌های کاری (Workstation) و سرورهای خاص خود را به هم متصل کنند. از آنجایی که بخشهای مختلف یک موسسه، اهداف متفاوتی را دنبال می‌کنند لذا در هر بخش، فارغ از آنکه دیگر بخشها چه می‌کنند، LAN متفاوتی پیاده می‌شود. دیر یا زود نیاز می‌شود که این LANها با یکدیگر تعامل و ارتباط داشته باشند. در این مثال، پیدایش LANهای متعدد ناشی از اختیار و آزادی مالکان آن بوده است.

دوم آنکه ممکن است سازمانها به صورت جغرافیایی در ساختمانهایی با فاصله قابل توجه، پراکنده باشند. شاید داشتن چندین LAN مجزا در هر ساختمان و وصل آنها از طریق «پلها» و لینک‌های لیزری ارزان‌تر از کشیدن یک کابل واحد بین تمام سایتها تمام شود.

سوم آنکه گاهی برای تنظیم بار و تعدیل ترافیک، لازم است که یک LAN منطقی و واحد به چندین LAN کوچکتر تقسیم شود. به عنوان مثال در بسیاری از دانشگاه‌ها هزاران ایستگاه کاری در اختیار دانشجویان و هیئت علمی قرار گرفته است. عموماً فایلها در ماشینهای سرور دهنده فایل نگهداری می‌شوند و حسب تقاضای کاربران بر روی ماشینشان منتقل و بارگذاری می‌شود. مقیاس بسیار بزرگ این سیستم مانع از آن می‌شود که بتوان تمام ایستگاه‌های کاری را در یک شبکه محلی واحد قرار داد چرا که پهنای باند مورد نیاز بسیار بالا خواهد بود. در عوض، مشابه با شکل ۴-۳۹ از چندین LAN که توسط «پل» به هم متصل شده، استفاده می‌شود. هر شبکه LAN گروهی از ایستگاه‌ها و سرور دهنده فایل خاص خود را دربرمی‌گیرد که بدین ترتیب، بیشتر ترافیک در حوزه یک LAN واحد محدود می‌شود و بار زیادی به ستون فقرات شبکه اضافه نخواهد شد.

اشاره به این نکته ارزشمند است که اگرچه عموماً شبکه‌های LAN را به صورت یک کابل چنداتصالی (Multidrop) یا ساختار باس ترسیم می‌کنیم (نمایش کلاسیک) ولیکن امروزه اغلب آنها توسط هاب و خصوصاً سوئیچها پیاده‌سازی می‌شوند. با این حال یک کابل طولانی با چندین ماشین متصل به آن، با یک هاب که ماشینها را از درون، بهم متصل می‌کند، از لحاظ عملکرد یکسان هستند. در هر دو حالت تمام ماشینها به یک «حوزه تصادم» (Collision Domain) یکسان متعلقند و تمام آنها برای ارسال فریم از پروتکل CSMA/CD استفاده می‌کنند.



شکل ۳۹-۴. چندین LAN از طریق یک ستون فقرات بهم متصل شده اند تا ظرفیت کل حمل بار آن از ظرفیت یک LAN واحد بیشتر شود.

شبکه های مبتنی بر سوئیچ متفاوت هستند و اگر چه قبلاً آنها را بررسی کرده ایم ولی باز هم نگاهی به آنها خواهیم انداخت .

چهارم آنکه در برخی از شرایط اگرچه یک شبکه محلی واحد از نظر حجم بار کفایت می کنند ولیکن فاصله فیزیکی بین ماشینهای دور، بسیار زیاد است. (مثلاً بیش از ۲/۵ کیلومتر در اترنت). حتی اگر عملیات کابل کشی ساده باشد ولیکن شبکه، در اثر تاخیر بسیار زیاد رفت و برگشت سیگنال (Round Trip Delay) کار نخواهد کرد. تنها راه حل آنست که LAN به چند بخش تقسیم شده و بین آنها پل نصب گردد. با استفاده از پل، می توان فاصله فیزیکی کل شبکه را افزایش داد.

پنجم مسئله قابلیت اعتماد است؛ بر روی یک LAN واحد، یک گره خراب که دنباله ای پیوسته از خروجی اشغال تولید می کند قادر است کل شبکه را فلج نماید. پلها را می توان در نقاط حساس قرار داد تا یک گره خراب و مغشوش نتواند کل سیستم را مختل کند. برخلاف یک تکرارکننده (Repeater) که ورودی خود را بی قید و شرط بازتولید می نماید یک پل را می توان به نحوی برنامه ریزی کرد تا در خصوص آنچه که هدایت می کند یا هدایت نمی کند تصمیم آگاهانه بگیرد.

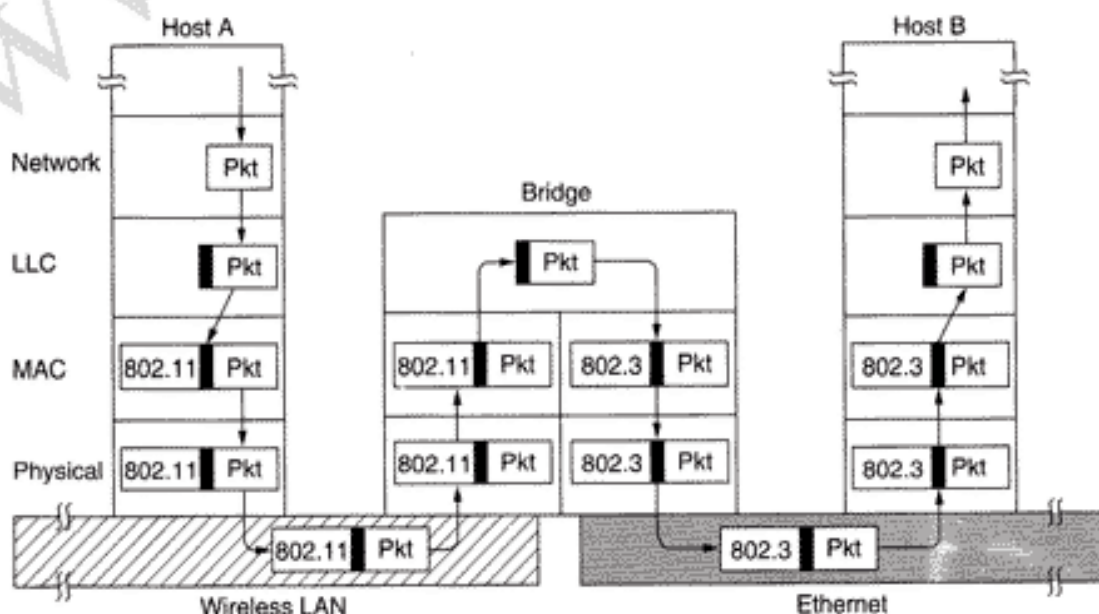
ششم و آخر آنکه پلها می توانند به امنیت اطلاعات در یک سازمان کمک نمایند. بیشتر کارتهای واسطه شبکه های LAN دارای حالتی به نام «حالت بی قید» (Promiscuous mode) هستند که در چنین حالتی تمام فریمهای جاری بر روی شبکه تحویل گرفته می شود، نه فریمهایی که دقیقاً به آدرس او ارسال شده اند. جاسوسان و فضولان به این ویژگی علاقمند هستند. با قرار دادن پلها در نقاط مختلف و اطمینان از عدم هدایت اطلاعات حساس به بخش های نامطمئن، مسئول سیستم می تواند بخشهایی از شبکه را از دیگر بخشها جدا کرده تا ترافیک آنها به خارج راه پیدا نکرده و در اختیار افراد نامطمئن قرار نگیرد.

هدف آرمانی آنست که پلها کاملاً نامرئی (شفاف-transparent) باشند، بدین معنا که بتوان ماشین را از یک بخش از شبکه به بخش دیگر منتقل کرد بدون آنکه به هیچگونه تغییری در سخت افزار، نرم افزار یا جداول پیکربندی نیاز باشد. همچنین باید این امکان وجود داشته باشد که تمام ماشینهای یک بخش از شبکه بتوانند فارغ از آنکه نوع LAN آنها چیست با ماشینهای بخش دیگر، مبادله اطلاعات داشته باشند. این هدف گاهی برآورده می شود ولیکن نه همیشه!

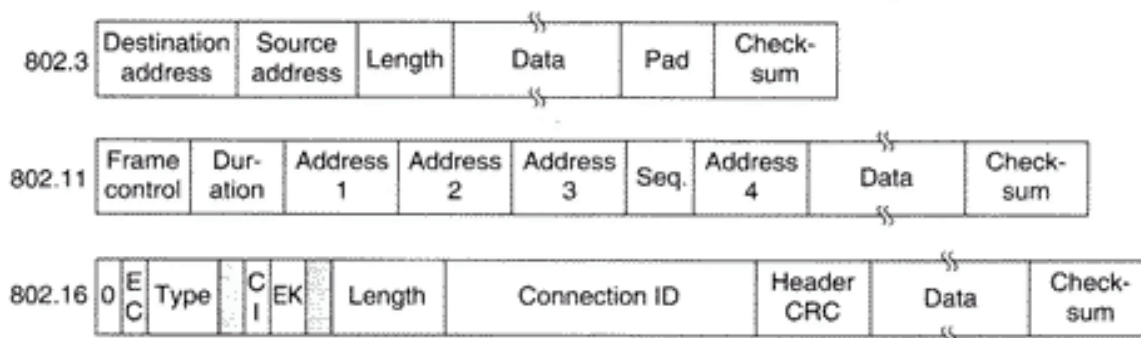
۴-۷-۱ پلهائی از 802.x به 802.y

پس از بررسی آنکه چرا به پلهای نیاز است اجازه بدهید بدین سوال بپردازیم که عملکرد آنها چگونه است؟ شکل ۴-۴۰ عملکرد یک پل ساده با دو درگاه (Port) را به تصویر کشیده است. ماشین میزبان A در یک شبکه محلی بی سیم، بسته ای برای ارسال به ماشین میزبان و ثابت B در شبکه اترنت (802.3) (که از طریق پل به شبکه بی سیم متصل شده) آماده کرده است. این بسته از زیرلایه LLC عبور کرده و سرآیند LLC به آن اضافه می شود (این سرآیند در شکل به صورت سیاه نشان داده شده است). سپس به زیرلایه MAC تحویل شده و در آنجا نیز سرآیند 802.11 به ابتدای آن افزوده می شود. (یک بخش انتهایی نیز به آخر فریم اضافه خواهد شد که در شکل نشان داده نشده است). این واحد داده، در هوا منتشر و توسط ایستگاه ثابت دریافت می شود؛ پس از بررسی، ایستگاه ثابت متوجه می گردد که باید آنرا به سمت اترنت ثابت هدایت کند. پس از رسیدن آن به پل (که شبکه 802.11 را به شبکه 802.3 متصل کرده)، پل کار دریافت آن از لایه فیزیکی را شروع کرده و آنرا به سمت زیرلایه های بالا هدایت می کند. در زیرلایه MAC از پل، سرآیند 802.11 حذف می شود. بسته اصلی (به همراه سرآیند LLC) تحویل زیرلایه LLC از پل می شود. در این مثال، بسته به سمت شبکه محلی 802.3 رهسپار است، لذا بسته از طریق بخش 802.3 در پل به سمت پائین حرکت کرده و نهایتاً بر روی شبکه اترنت منتقل می شود. دقت کنید که یک پل که k شبکه مختلف را به هم متصل می کند دارای k زیرلایه MAC و تعداد k لایه فیزیکی است (به ازای هر نوع یکی). تا اینجا به نظر می رسد که انتقال فریم از یک LAN به LAN دیگر ساده است اما واقعیت این چنین نیست. در این بخش، برخی از دشواریهای ساخت یک پل را که انواع مختلف شبکه های LAN (و همچنین شبکه MAN) سری 802 را به هم متصل می کند، متذکر می شویم. ما بر روی شبکه های 802.3، 802.11 و 802.16 متمرکز خواهیم شد ولی انواع دیگر آن هم وجود دارد که هر کدام مسائل خاص خود را دارند.

برای شروع باید یادآوری کرد که هر LAN قالب فریم خاص خود را دارد. (شکل ۴-۴۱ را ببینید) اگرچه تفاوتی که بین قالب فریم در شبکه های اترنت، «توکن رینگ» (Token Ring) و «توکن باس» (Token Bus) وجود دارد بیشتر ناشی از اتکاء به نفس شرکتهای بزرگ ابداع کننده آنها (یعنی به ترتیب زیراکس، آی بی ام و جنرال موتورز) و همچنین شرایط زمانی آن دوران بوده ولیکن تفاوت شبکه های کنونی تقریباً لازم و



شکل ۴-۴۰. عملکرد یک پل از شبکه 802.11 به 802.3.



شکل ۴-۴. انواع قالب فریمهای IEEE 802 (طول هر فریم در شکل، مقیاس اندازه واقعی آن نیست).

واقعی است. به عنوان مثال فیلد Duration (طول زمان) که در 802.11 وجود دارد ناشی از ماهیت پروتکل MACAW و همچنین عدم توانایی شنود کانال در این شبکه می باشد (برخلاف اترنت). در نتیجه، انتقال یک فریم بین دو شبکه متفاوت LAN، مستلزم دگرگونی در قالب فریمهاست که طبقاً نیاز به زمان CPU، محاسبه مجدد کدهای جدید کشف خطا (Checksum) دارد و این احتمال نیز پدید می آید که در اثر خرابی بیتها در حافظه پل، خطاهای غیرقابل کشف، داده‌ها را آلوده کند.

مشکل دوم آنست که LANهای به هم متصل شده الزاماً با سرعت مشابهی کار نمی کنند. وقتی «پل» یک دنباله پی در پی از فریمها را از LAN سریع برای هدایت به LAN کندتر می پذیرد قادر نخواهد بود با همان سرعتی که فریم را دریافت می کند از دست آنها رهائی یابد. به عنوان مثال هرگاه یک شبکه اترنت گیگابیتی با بالاترین سرعت، بیتها را به سوی شبکه یازده مگابیت درثانیه‌ای LAN 802.11b روانه کند، پل باید بتواند آنها را موقتاً در حافظه نگهداری کند و این حافظه نباید سرریز شود. [که با چنین سرعتی بعید نیست.] در ضمن، برخی پلها سه یا چندین شبکه LAN را به هم متصل می کنند. در چنین پلهایی اگر چندین شبکه بطور همزمان تلاش کنند فریمهای خود را به یک LAN مشابه بفرستند مشکل سرریز شدن حافظه بوجود خواهد آمد، حتی اگر تمام LANها با سرعت یکسانی کار کنند.

مشکل سوم که خطرناکترین مشکل بالقوه پلها محسوب می شود آنست که در شبکه‌های محلی و مختلف سری 802، حداکثر طول فریم، متفاوت است. این مشکل زمانی بروز می کند که یک فریم بزرگ باید به سوی شبکه‌ای هدایت شود که آن LAN قادر به دریافت آن نیست. شکستن فریم به تعدادی قطعه، خارج از حیطه وظایف این لایه است. در تمام پروتکلهای این لایه فرض بر آنست که فریمها یا می رسند یا نمی رسند و هیچ تمهیدی برای شکستن یک فریم به قطعات و بازسازی آن اندیشیده نشده است. نمی توان گفت که چنین پروتکلی ابداع نشده است. چنین پروتکلی ابداع شده و وجود هم دارد ولیکن در حیطه وظایف پروتکلهای پیونده داده نیست. پلها نیز نباید به محتوای هر فریم کاری داشته باشند. اساساً (در چنین وضعیتی) راه حلی برای این مشکل وجود ندارد. فریمهای بسیار طولانی باید به جای انتقال، حذف شوند. [جهت اجتناب از هرگونه مداخله در پیکربندی سخت افزار یا نرم افزار]

نکته بعدی امنیت داده‌هاست. شبکه‌های 802.11 و 802.16، هر دو از رمزنگاری در سطح لایه پیوند داده‌ها پیشنهادی می کنند در حالیکه اترنت چنین امکانی ندارد. بدین معنا که خدمات متنوع رمزنگاری عرضه شده در شبکه‌های بی سیم، در خلال ورود ترافیک به شبکه اترنت از دست می رود. بدتر از آن، اینکه اگر یک دستگاه در شبکه بی سیم از رمزنگاری در سطح لایه پیوند داده‌ها بهره گرفته باشد هیچ راهی برای رمزگشایی آن هنگام

دریافت در یک ایستگاه اترنت وجود ندارد. از طرفی اگر ایستگاه بی سیم از رمزنگاری استفاده نکند ترافیک داده‌ها از طریق لینک هوایی در معرض شنود همگان قرار می‌گیرد.

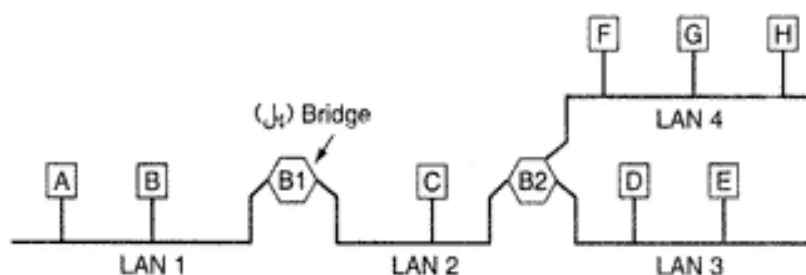
یک راه حل برای مشکل امنیت آنست که رمزنگاری در لایه‌های بالایی انجام شود ولیکن در این روش ایستگاه 802.11 باید بداند که آیا با ایستگاه دیگری در شبکه 802.11 صحبت می‌کند (تا از رمزنگاری در لایه پیوند داده بهره بگیرد) یا نه (تا از چنین امکانی استفاده نکند). وادار کردن ایستگاه به تصمیم‌گیری «شفافیت» را از بین خواهد برد.

نکته آخر مسئله «کیفیت خدمات» است. هر دو شبکه 802.11 و 802.16 این خدمات را به نحو متفاوتی در اختیار می‌گذارند: اولی در «حالت PCF» و دومی با استفاده از «اتصال با نرخ ارسال ثابت» (Constant Bit Rate Connection). در شبکه اترنت چیزی به نام کیفیت خدمات بی مفهوم است و بدین ترتیب ترافیک هر یک از این دو شبکه در حین عبور از اترنت «کیفیت خدمات» را از دست می‌دهند. [به عنوان مثال اگر چه می‌توان میزان حداکثر تاخیر در دو شبکه اول را تضمین کرد ولی در اترنت چنین تضمینی وجود ندارد و با وصل این دو شبکه نمی‌توان تاخیر کل را تضمین نمود و مقدار آن تصادفی خواهد بود. -م]

۴-۷-۲ بهم‌بندی شبکه‌ها به صورت محلی (Local Internetworking)

در بخش قبلی به مشکلاتی که در وصل دو شبکه محلی نوع IEEE 802 بروز می‌کند، پرداختیم. با این وجود در سازمانهای بزرگ با شبکه‌های LAN متعدد، متصل کردن تمام آنها به یکدیگر مشکلات گوناگونی را بوجود می‌آورد، حتی اگر تمام آنها از نوع اترنت باشند. حالت آرمانی آنست که بتوان به راحتی از سازمان بیرون رفت و چند پل مبتنی بر استاندارد IEEE خریداری کرد، سپس با وصل تمام کابلها به پلها، شبکه فوراً و به درستی بکار بیفتد. نباید به هیچ تغییر سخت‌افزاری، نرم‌افزاری، تنظیم سوییچها، بازگذاری یا تغییر در جداول مسیریابی یا پارامترهای آن یا هر تغییر دیگری نیاز باشد. فقط باید کابلها را وصل کرد و رفت! به علاوه عملکرد طبیعی هیچکدام از شبکه‌های LAN نباید تحت الشعاع قرار بگیرد. به عبارت دیگر، پلها بایستی کاملاً «شفاف» باشند (یعنی از دیدگاه سخت‌افزار و نرم‌افزار غیرقابل رویت باشند). شگفت آنکه این کار عملاً ممکن است. حال اجمالاً بررسی کنیم که این کار جادوئی به چه نحو انجام می‌شود!

در ساده‌ترین حالت، یک «پل شفاف» در «حالت بی‌قید» (Promiscuous Mode) عمل کرده و تمام فریمهای جاری بر روی تمام LANهایی را که بدانها متصل است، می‌پذیرد. به عنوان یک مثال، به پیکربندی شکل ۴-۴۲ توجه نمایید. پل B1 به شبکه‌های محلی ۱ و ۲ متصل شده و پل B2 نیز به شبکه‌های محلی ۲ و ۳ و ۴ متصل است. فریمی که به پل B1 می‌رسد ولی مقصد آن ماشین A است می‌تواند فوراً توسط پل نادیده انگاشته شود. (حذف شود) زیرا این فریم بر روی LAN صحیحی [که به مقصد ختم می‌شود] قرار گرفته است ولی فریمی که از LAN 1 به مقصد C یا F دریافت می‌شود باید منتقل شود.



شکل ۴-۴۲. یک پیکربندی از شبکه‌های متصل بهم با چهار شبکه محلی و دو پل.

وقتی فریمی دریافت می شود، پل در ابتدا باید تصمیم بگیرد که آیا باید آنرا حذف کند یا باید آنرا منتقل نماید؛ سپس در صورت نیاز به انتقال، باید مشخص شود که به کدام LAN هدایت گردد. این تصمیم گیری با جستجوی آدرس مقصد درون یک جدول بزرگ در حافظه پل انجام می گیرد. (به این جدول، Hash Table گفته می شود). این جدول فهرست تمام ماشینهای مقصد را در اختیار دارد و می تواند تعیین کند که این ماشینها به کدامیک از خطوط پل تعلق دارند. مثلاً جدول پل B2 می تواند مشخص کند که ماشین A به LAN 2 تعلق دارد، زیرا تمام آنچه که لازم است B2 بداند آنست که فریمهایی با مقصد A را بروی چه شبکه ای ارسال کند. در واقع برای پل B2 مهم نیست که این فریم، بعداً چگونه هدایت و منتقل می شود.

وقتی پلها برای اولین بار به کار می افتند تمام جداول Hash خالی هستند. هیچیک از پلها نمی دانند که هر یک از ماشینهای مقصد در کجا قرار گرفته اند، لذا برای انتقال فریم از «الگوریتم سیل آسا» (Flooding Algorithm) استفاده می نمایند یعنی تمام فریمهای ورودی که مقصدشان ناشناخته است بروی تمام شبکه هائی که پل بدانها متصل است، ارسال می شود (البته به استثنای شبکه ای که فریم از آن دریافت شده است). به مرور زمان، پل متوجه خواهد شد که هر ماشین مقصد، در کجا قرار گرفته است. (طبق الگوریتمی که در زیر بدان اشاره خواهیم کرد). هرگاه یک ماشین مقصد شناسائی شد، فریمهایی که بعداً بدان مقصد روانه می شوند توسط پل بروی LAN مناسب هدایت خواهد شد و از روش سیل آسا استفاده نمی شود.

الگوریتمی که توسط پلهای شفاف به کار گرفته می شود روش «یادگیری غیرمستقیم» (Backward Learning) است. قبلاً اشاره شد که پل در «حالت بی قید» کار می کند لذا هر فریمی را که بروی یکی از شبکه های متصل به او مبادله می شود، می بیند (و همچنین دریافت و پردازش می نماید). با نگاهی به آدرس مبدا هر فریم، پل می تواند بفهمد که کدام ماشین از طریق کدام LAN قابل دسترسی است. به عنوان مثال در شکل ۴-۲۲ فرض کنید که پل B1 فریمی را بر روی LAN 2 می بیند که توسط C تولید شده است؛ بدین ترتیب متوجه می شود که می توان از طریق LAN 2 به C رسید و بدین ترتیب در جدول Hash خود درج می کند که فریمهای روانه به سمت C باید از طریق LAN 2 منتقل و هدایت شوند. بعداً تمام فریمهایی که به مقصد C و از طریق LAN 1 به پل وارد می شوند منتقل می شوند ولی فریمهایی که به مقصد C ولی از طریق LAN 2 به پل می رسند حذف خواهند شد.

با خاموش یا روشن شدن پلها و ماشینها و یا جابجائی آنها در سطح شبکه، توپولوژی شبکه تغییر می کند. برای آنکه توپولوژی شبکه به صورت پویا دنبال شود وقتی یک «درایه» (Entry) در جدول هر پل درج می گردد زمان دریافت فریم نیز در آن «درایه» یادداشت می شود. هرگاه بعداً فریمی که آدرس مبدا آن قبلاً در جدول درج گردیده، دریافت شود زمان درج شده در درایه متناظر آن، با زمان فعلی بهنگام سازی می شود. بدین ترتیب زمان درج شده در هر «درایه» (Entry) آخرین زمانی که فریمی از آن ماشین دریافت شده را تعیین می کند.

بطور متناوب، یک پروسه در درون پل، جدول Hash را جستجو و پوش می کند و تمام درایه هائی را که برای بیش از چند دقیقه به هنگام نشده اند، حذف می نماید. بدین ترتیب اگر یک کامپیوتر از شبکه خود جدا و در ساختمان جابجا شده یا به شبکه دیگری متصل شود پس از گذشت چند دقیقه عملیات عادی خود را از سر می گیرد و نیازی به مداخله و تنظیمات دستی نیست. البته این الگوریتم بدین معنا هم هست که اگر کامپیوتری برای چندین دقیقه ساکت باشد (یعنی هیچ داده ای نفرستد) درایه متناظر با او از جدول پلها حذف شده و از آن به بعد هر ترافیکی که برایش ارسال می شود بروش سیل آسا هدایت خواهد شد مگر آنکه بعداً خودش فریمی ارسال کند [و آدرس او در جدول پلها درج شود. -م]

روند مسیریابی فریمهای ورودی پل، به شبکه LAN مبدا (Source LAN) و شبکه LAN مقصد (Destination LAN) بستگی دارد. این روال به ترتیب ذیل است:

۱. اگر شبکه LAN مبدأ و مقصد یکسان هستند فریم را نادیده بگیر.

۲. اگر شبکه های مبدأ و مقصد متفاوت هستند فریم را منتقل کن.

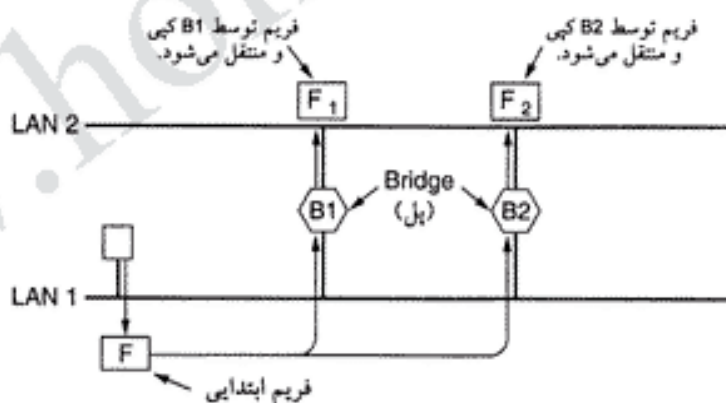
۳. اگر شبکه مقصد ناشناخته است بروش «سیل آسا» عمل کن.

این الگوریتم به ازای ورود هر فریم باید یکبار اعمال گردد. یک تراشه خاص VLSI عملیات جستجو و به هنگام سازی جدول و درایه های (Entries) را در عرض چند میکروثانیه انجام می دهد.

۳-۷-۴ پلهای مبتنی بر درخت پوشا (Spanning Tree)

برخی از سایتها برای افزایش قابلیت اعتماد، بین دو LAN دو یا چند پل موازی نصب می کنند. (به شکل ۴-۴۳ نگاه کنید). ولیکن چنین آرایشی مشکلات دیگری را ایجاد خواهد کرد چراکه در ساختار توپولوژی، حلقه ایجاد می شود.

چنین مشکلاتی را می توان یکمک مثال شکل ۴-۴۳ و با مشاهده روند هدایت فریمی از ماشین F به مقصدی ناشناخته، تحلیل کرد. هر پل طبق قاعده عمومی در مواجهه با فریمهایی که به مقصد ناشناخته روانه هستند از روش «سیل آسا» استفاده می کند یعنی در اینجا فریم بر روی LAN 2 منتقل می شود. [فریم منتقل شده بر روی LAN 2 را F₂ بنامید.] اندکی بعد، پل ۱ فریم F₂ را می بیند که مقصد آن ناشناخته است و آنرا بر روی LAN 1 هدایت می کند و فریم F₃ تولید می شود. (فریم F₃ در شکل نشان داده نشده است). بروش مشابه پل ۲ فریم F₁ را بر روی LAN 1 منتقل کرده و F₄ تولید می شود. (F₄ نیز نشان داده نشده است) [F₄، F₃، F₂، F₁] نسخه های مشابه با فریم F هستند که توسط پل منتقل می شوند. [حال پل ۲ فریم F₄ را منتقل و پل ۱ فریم F₃ را منتقل می نماید و این دور باطل تا ابد ادامه دارد.



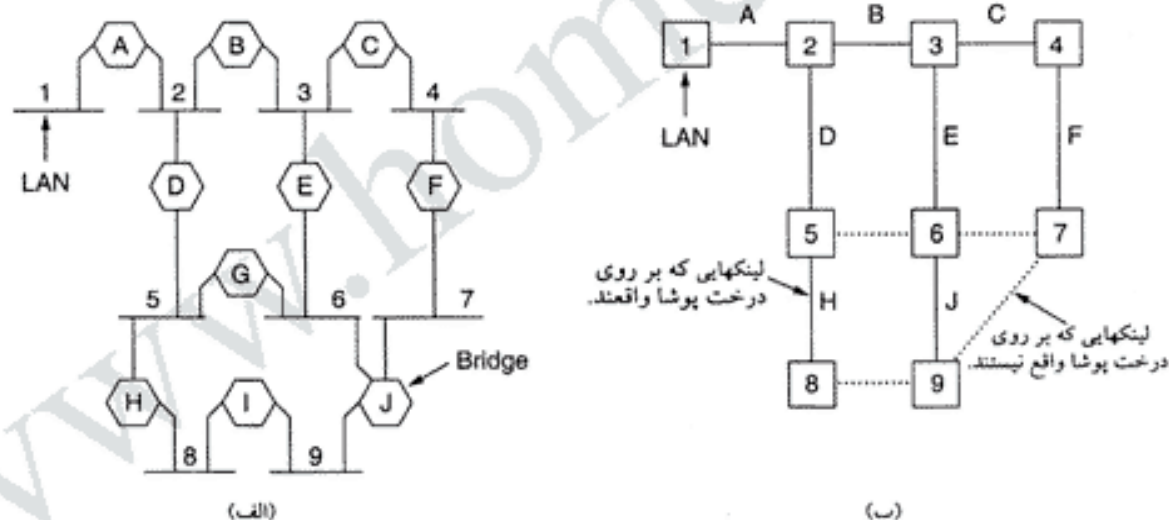
شکل ۴-۴۳. دو پل شفاف (نامرئی) موازی.

راه حل این مشکل آنست که پلها با یکدیگر ارتباط و محاوره داشته باشند و توپولوژی واقعی را به صورت یک «درخت پوشا» (Spanning Tree) که در آن به تمام LANها مسیر دسترسی وجود دارد، در نظر بگیرند. در نتیجه برخی از اتصالات بالقوه که بین LANها وجود دارد، نادیده گرفته می شود تا مجازاً یک توپولوژی بدون حلقه ایجاد شود. به عنوان مثال در شکل ۴-۴۴ الف، نه شبکه LAN را می بینیم که توسط ده پل متصل شده اند. این پیکربندی را می توان در قالب گرافی در نظر گرفت که در آن، هر کدام از LANها یک «گره» تلقی می شوند. هر «کمان» (Arc) اتصال دو LAN توسط پل را نشان می دهد. با حذف برخی از «کمانها» (که در شکل ۴-۴۴ ب به صورت خط چین نشان داده شده)، این گراف به یک درخت پوشا کاهش می یابد. با استفاده از این درخت، بین دو LAN دقیقاً یک مسیر وجود دارد. پس از آنکه پلها درخت پوشا را تشکیل دادند، هدایت تمام فریمها (بین

شبکه های محلی) از ساختار درخت پوشا تبعیت خواهد کرد. از آنجایی که بین هر مبدا و مقصد در شبکه فقط و فقط یک مسیر وجود دارد لذا ایجاد حلقه غیر ممکن است.

برای ایجاد درخت پوشا، پلها بایستی یک پل را به عنوان ریشه این درخت انتخاب کنند. جهت این انتخاب، پلها شماره سریال خود را که توسط کارخانه سازنده تنظیم و یکتا بودن آن در کل دنیا تضمین شده است، به صورت فراگیر (Broadcast) به اطلاع همه می رسانند. پل که دارای کوچکترین شماره سریال است به عنوان «ریشه» انتخاب می شود. سپس درختی با کوتاهترین مسیر که از ریشه شروع شده و تمام پلها و LANها را در بر می گیرد، ایجاد می شود. این درخت همان «درخت پوشا» است. اگر یک پل یا شبکه LAN از کار بیفتند، این درخت از نو محاسبه می شود.

نتیجه این الگوریتم آنست که بین هر LAN و ریشه و بدین ترتیب از ریشه به بقیه LANها، یک مسیر یکتا ایجاد می شود. اگر چه این درخت تمام LANها را در بر می گیرد ولیکن تمام پلها لزوماً در این درخت قرار نمی گیرند (برای اجتناب از حلقه). حتی پس از ایجاد درخت پوشا و در خلال عملکرد طبیعی، این الگوریتم به صورت متناوب اجرا می شود تا هر گونه تغییر در توپولوژی به صورت خودکار تشخیص داده شده و درخت اصلاح شود. این الگوریتم توزیع شده که برای ایجاد درختهای پوشا مورد استفاده قرار می گیرد توسط خانم رابیدا پرلن ابداع و به تفصیل در مرجع (Perlman, 2000) تشریح شده است. همچنین این الگوریتم در IEEE 802.1D استاندارد سازی شده است.

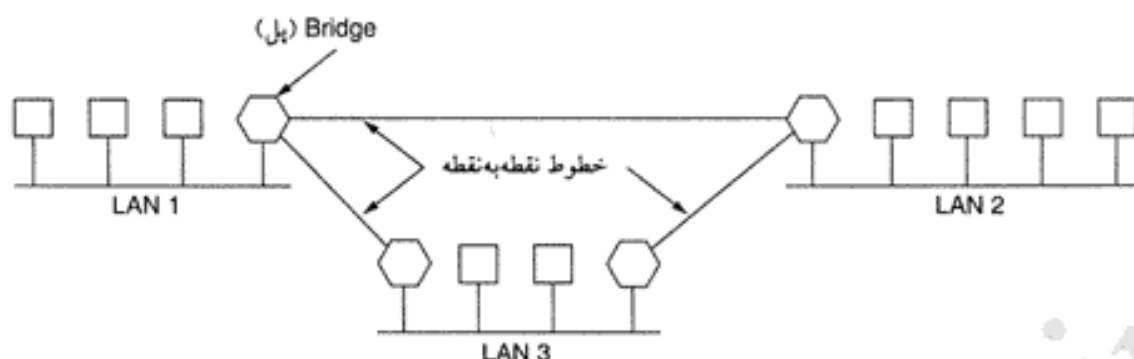


شکل ۴۴-۴. (الف) چند شبکه LAN بهم متصل (ب) یک درخت پوشا که تمام شبکه های LAN را در بر می گیرد. (خطوط نقطه چین جزو درخت پوشا نیستند).

۴-۷-۲ پلهای راه دور (Remote Bridges)

کاربرد متداول پلها آنست که دو یا چند LAN راه دور را بهم متصل کنند. بعنوان مثال ممکن است یک شرکت دارای کارخانه هایی در چند شهر باشد و هر کارخانه، LAN مختص به خود را داشته باشد. در حالت آرمانی باید تمام LANها به هم متصل شده باشند تا کل سیستم نقش یک LAN عظیم را ایفاء کند.

این هدف با قرار دادن یک پل بین هر دو LAN و وصل کردن پلها به وسیله خطوط نقطه به نقطه (مثلاً بکمک خطوط اجاره ای عرضه شده توسط شرکتهای تلفن) برآورده خواهد شد. در شکل ۴-۴۵، یک سیستم ساده با سه LAN به تصویر کشیده شده است. در اینجا روشهای رایج مسیریابی اعمال می شود. ساده ترین راه برای تحلیل این



شکل ۴-۴۵. برای اتصال شبکه های محلی راه دور می توان از پلهای راه دور بهره گرفت.

ساختار آنست که سه خط نقطه به نقطه را به مثابه یک شبکه LAN بدون هیچ ماشین میزبان (Hostless LAN)، تصور کنید. در این صورت یک سیستم معمولی یا شش LAN داریم که با چهار پل به هم متصل شده اند. در هیچ کجا از مطالبی که تاکنون مطالعه کرده ایم گفته نشده که حتماً باید به یک LAN ماشین میزبان متصل شده باشد! برای خطوط نقطه به نقطه می توان از پروتکل های متنوعی بهره گرفت. یک انتخاب آنست که از یکی از استانداردهای موجود برای خطوط نقطه به نقطه مثل PPP استفاده شود و کل فریمهای MAC [شامل سرآیند و فیلدهای پایانی] درون فیلد حمل داده آن (یعنی Payload) قرار بگیرد. این استراتژی بشرط آنکه تمام LAN ها مثل هم باشند، به نحو احسن کار می کند و تنها مسئله باقیمانده آنست که فریمها به LAN صحیح تحویل شوند. انتخاب دیگر آنست که سرآیند و پی آیند^۱ هر فریم MAC در پل مبداء حذف شود و باقیمانده در فیلد حمل داده از فریم مربوط به پروتکل نقطه به نقطه قرار بگیرد. سپس در پل مقصد سرآیند و پی آیند MAC جدیدی برای آن قطعه داده تولید شود. اشکال این روش آنست که کد کشف خطایی که فریم در حین دریافت در ماشین مقصد دارد همانی نیست که توسط ماشین مبداء تولید شده و بدین ترتیب ممکن است خطاهایی که در حافظه پل، داده ها را آلوده می کند کشف نشود.

۵-۷-۴ تکرارکننده^۲، هاب^۳، پل^۴، سوئیچ^۵، مسیریاب^۶ و دروازه^۷

تا اینجا کتاب روشهای گوناگونی را مرور کردیم که وظیفه همگی تحویل فریمها یا بسته ها از یک بخش کابل به بخش دیگر است. همچنین اشاره ای به تکرارکننده ها، پلها، سوئیچها، هابها، مسیریابها و دروازه ها داشتیم. از تمام این ابزارها به یک منظور استفاده می شود ولیکن تفاوت های مشهود و نامشهود زیادی دارند. از آنجایی که این ابزارها بسیار متنوعند، لذا مروری بر همه آنها و بررسی شباهتها و تفاوت های آنها ارزشمند خواهد بود.

برای شروع باید گفت که این ابزارها (به نحوی که در شکل ۴-۴۶ الف دیده می شود) در لایه های متفاوتی عمل می کنند. بسته به آنکه هر یک از این ابزارها در چه لایه ای عمل می کنند مکانیزم هدایت اطلاعات متفاوت است. در قالب یک نمایشنامه عمومی: کاربر مقداری اطلاعات برای ارسال به یک ماشین راه دور تولید می نماید. این داده ها تحویل لایه انتقال (Transport Layer) شده و بدان سرآیند لازم اضافه می گردد (مثلاً سرآیند TCP). سپس، واحد اطلاعاتی حاصل به سمت لایه پائین یعنی لایه شبکه عبور داده می شود. لایه شبکه سرآیند خاص خود را بدان افزوده و یک بسته مخصوص لایه شبکه ساخته می شود (مثلاً یک بسته IP). در شکل ۴-۴۶ ب یک بسته IP را می بینیم که به صورت خاکستری نشان داده شده است. این بسته تحویل لایه پیوند داده (Data Link)

۱. Header and Trailer

۲. Repeater

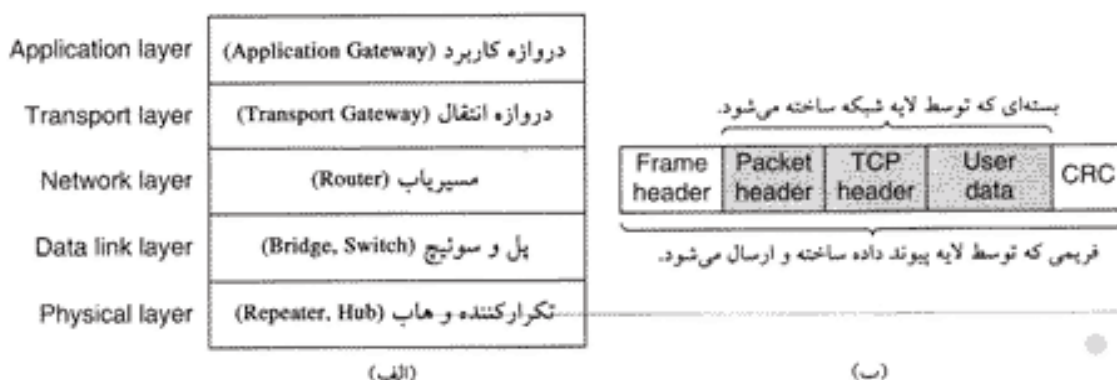
۳. Hub

۴. Bridge

۵. Switch

۶. Router

۷. Gateway

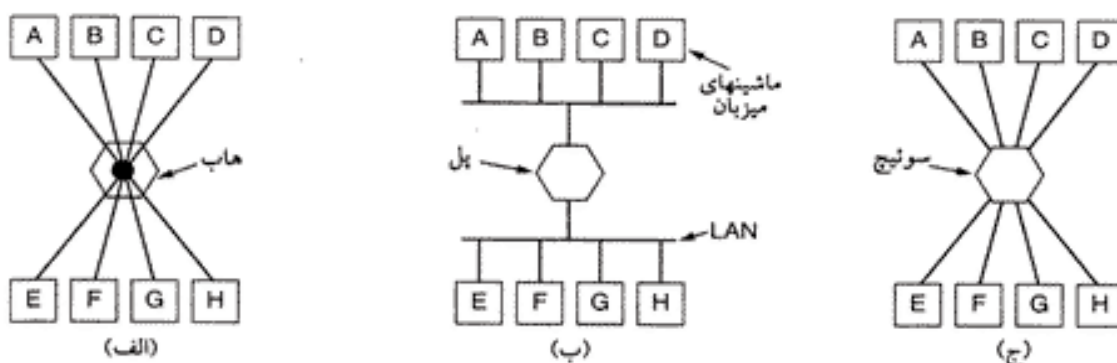


شکل ۴-۴۶. (الف) جایگاه هر ابزار در پشت پرده پروتکلی (ب) فریمها، بسته ها و سرآیندها.

می شود و آن هم سرآیند خاص خود و کد کشف خطا (CRC) را بدان افزوده و فریم حاصل را جهت ارسال به لایه فیزیکی تسلیم می کند. (مثلاً برای ارسال بر روی LAN)

حال اجازه بدهید نگاهی به ابزارهای هدایت اطلاعات بیندازیم و ببینیم که این ابزارها چه ارتباطی با بسته ها و فریمها دارند. در پائین ترین سطح یعنی در لایه فیزیکی به «تکرارکننده ها» بر می خوریم. تکرارکننده ابزاری است آنالوگ، که دو قطعه کابل را بهم متصل می کند. سیگنالی که بر روی یکی از این قطعات ظاهر گردد، تقویت (بازتولید) شده و بر روی قطعه دیگر قرار داده می شود. تکرارکننده ها هیچگونه درکی از «فریم»، «بسته» یا «سرآیند» ندارند. آنها صرفاً با مفهوم «ولت» آشنا هستند! به عنوان مثال در اترنت کلاسیک اجازه داده شده برای افزایش طول حداکثر کابل از ۵۰۰ متر به ۲۵۰۰ متر از چهار تکرارکننده استفاده شود.

سپس به هاب (Hub) می رسیم: یک هاب معمولی (از نوع غیرفعال)، دارای تعدادی خط ورودی است که این خطوط از لحاظ الکتریکی در درون هاب به هم متصل شده اند. فریمی که از یک خط ورودی دریافت می شود بر روی خطوط دیگر ارسال خواهد شد. هرگاه دو فریم بطور همزمان به هاب ارسال شوند، تصادم رخ خواهد داد؛ دقیقاً همانند اتفاقی که بر روی کابل کواکسیال می افتد. به عبارت دیگر، کل هاب یک «حوزه تصادم» (Collision Domain) واحد را تشکیل خواهد داد. تمام خطوط ورودی هاب، باید با سرعت یکسانی کار کنند. هابها متفاوت از تکرارکننده ها هستند، از آن جهت که هاب (معمولاً) سیگنالهای ورودی را تقویت نمی کند و طراحی آنها به گونه ای است که چندین کارت واسط خط (Line Card) دارند و هر یک از کارت ها خود چندین ورودی دارند ولیکن در مجموع این تفاوتها ناچیز است. همانند تکرارکننده ها، هابها نیز آدرسهای 802 [آدرسهای MAC] را بررسی نکرده و به هیچ وجه از آنها استفاده نمی کنند. در شکل ۴-۴۷-الف تصویری نمادین از یک هاب نشان داده شده است.



شکل ۴-۴۷. (الف) یک هاب (ب) یک پل (ج) یک سوییچ.

حال اجازه بدهید به سمت لایه پیوند داده یعنی لایه‌ای که در آن «پلها» و «سوئیچها» تعریف شده‌اند حرکت کنیم. قبلاً «پلها» را تا حدی مطالعه کردیم. یک پل دو یا چند شبکه LAN را همانند شکل ۴-۴۷-ب به هم متصل می‌کند. وقتی فریمی دریافت می‌شود نرم‌افزار درون پل، آدرس مقصد را از سرآیند فریم استخراج و آنرا درون جدول خود جستجو می‌کند تا محلی را که فریم باید بدانجا ارسال شود، بیابد. در اترنت، این آدرس همان فیلد ۴۸ بیتی آدرس در شکل ۴-۱۷ است. شبیه به هاب، پل‌های پیشرفته نیز دارای «کارت‌های خط» (Line Card) هستند که معمولاً چهار تا هشت خط ورودی از یک نوع شبکه معین در آنها تعبیه شده است. یک «کارت خط اترنت» (Ethernet Line Card) نمی‌تواند مثلاً فریم‌های شبکه توکن رینگ را بپذیرد چرا که نمی‌داند در کجای سرآیند فریم آدرس مقصد را پیدا کند! با این وجود یک «پل» می‌تواند برای انواع شبکه‌های مختلف با سرعت متفاوت، «کارت‌های خط» مجزا داشته باشد. در یک «پل» برخلاف هاب، هر خط «حوزه تصادم» خاص خود را دارد. «سوئیچها» شبیه به پلها هستند چرا که هر دوی آنها براساس آدرسهای درون فریم، آنها را مسیریابی و هدایت می‌کنند. در واقع بسیاری از افراد این دو واژه را به صورت معادل به کار می‌برند. تفاوت اصلی آنها در این است که یک سوئیچ شبیه به شکل ۴-۴۷-ج، برای وصل کردن کامپیوترهای منفرد به یکدیگر، کاربرد دارد. طبعاً وقتی ماشین میزبان A در شکل ۴-۴۷-ب می‌خواهد فریمی را برای ماشین B بفرستد، پل اگر چه فریم را دریافت می‌کند ولی آن را نادیده می‌گیرد. برخلاف آن در شکل ۴-۴۷-ج، سوئیچ باید بلافاصله فریم را از A به سمت B هدایت نماید چرا که هیچ راهی برای تحویل فریم به B [جز از طریق سوئیچ] نیست. از آنجایی که معمولاً هر یک از پورت‌های یک سوئیچ به یک کامپیوتر منفرد وصل می‌شود لذا یک سوئیچ باید پورت‌های بسیار بیشتری (در مقایسه با پل‌هایی که برای وصل تعداد کمی LAN طراحی شده‌اند) داشته باشد. در ضمن هر یک از «کارت‌های خط» بایستی فضای بافر کافی برای ذخیره فریم‌های دریافتی از هر یک از پورت‌ها در اختیار داشته باشند. از آنجایی هر یک از پورت‌ها «حوزه تصادم» مجزا و متعلق به خود را دارند لذا هیچ فریمی در اثر تصادم از بین نخواهد رفت. با این وجود اگر فریم‌ها با نرخی بیش از ظرفیت سوئیچ، وارد گردند ممکن است فضای بافر پر شده و سوئیچ مجبور به حذف آنها شود.

برای آنکه این مشکل کمی کاهش یابد در سوئیچ‌های مدرن به محض آنکه فیلد آدرس مقصد از فریم وارد گردید، عمل هدایت و انتقال فریم شروع می‌شود، حتی قبل از آنکه مابقی فریم بطور کامل دریافت شده باشد. البته در صورتی که خط خروجی مورد نظر آزاد باشد. این سوئیچ‌ها روش «ذخیره و هدایت» (Store & Forward) را به کار نمی‌برند. [در روش «ذخیره و هدایت» ابتدا کل فریم دریافتی ذخیره شده و سپس عملیات هدایت آغاز می‌شود]. این نوع از سوئیچ‌ها که اغلب به نام Cut-Through Switches (سوئیچ‌های میانبر) مشهورند، کاملاً به صورت سخت‌افزاری پیاده سازی می‌شوند در حالیکه پلها عموماً دارای یک CPU واقعی بوده و عمل «ذخیره و هدایت» را توسط نرم‌افزار انجام می‌دهند. بهر حال چون تمام پلها و سوئیچ‌های مدرن دارای تراشه مدار مجتمع ویژه‌ای جهت هدایت و انتقال فریم‌ها هستند لذا امروزه تفاوت‌های بین پل و سوئیچ بیشتر به مسائل بازاری بستگی دارد تا مسائل فنی.

تا اینجا تکرارکننده‌ها و هابها را بررسی کرده‌ایم که کاملاً شبیه به هم هستند و هم‌منظور به پلها و سوئیچ‌ها پرداختیم که آنها نیز شباهت زیادی به یکدیگر دارند. حال به سوی «مسیریاب» (Router) حرکت خواهیم کرد که با تمام ابزارهای فوق تفاوت دارد. وقتی بسته‌ای به یک مسیریاب وارد می‌شود ابتدا سرآیند و فیلدهای انتهایی فریم حذف شده و سپس بسته جاسازی شده در درون فیلد حمل داده فریم (Payload) (که در شکل ۴-۴۶ به صورت خاکستری نشان داده شده)، تحویل نرم‌افزار مسیریابی می‌شود. این نرم‌افزار برای انتخاب خط خروجی از مشخصات واقع در سرآیند بسته استفاده می‌کند. در بسته‌های IP، سرآیند هر بسته شامل یک آدرس ۳۲ بیتی

(در IPv4 یا ۱۲۸ بیتی (در IPv6) است ولی آدرس ۴۸ بیتی 802 ندارد. نرم افزار مسیریابی نمی تواند آدرس فریمها را ببیند و حتی نمی تواند متوجه شود که بسته از طریق یک LAN دریافت شده یا از روی یک خط نقطه به نقطه. در فصل پنجم مسیریابها و فرایندهای مسیریابی را تشریح خواهیم کرد.

در لایه بالاتر به «دروازه های انتقال» (Transport Gateway) بر می خوریم. این دروازه ها ارتباط دو کامپیوتر را که از پروتکل های اتصال گرای متفاوتی در لایه انتقال استفاده می کنند، برقرار می نماید. به عنوان مثال فرض کنید کامپیوتری که از پروتکل اتصال گرای TCP/IP استفاده کرده، می خواهد با کامپیوتری که از پروتکل اتصال گرای ATM استفاده می کند، محاوره نماید. یک دروازه انتقال می تواند بسته هایی که از طریق یک «اتصال» دریافت می شوند را پس از تغییرات لازم بر روی «اتصال» دیگر بفرستد.

در نهایت، «دروازه های کاربرد» (Application Gateway) قالب و محتوای داده ها را تشخیص می دهند و یک «پیام» را به پیامی دیگر ترجمه می کنند. بعنوان مثال یک «دروازه پست الکترونیکی» می تواند یک پیام اینترنتی [نامه الکترونیکی] را به پیام SMS برای گوشی های تلفن همراه ترجمه نماید.

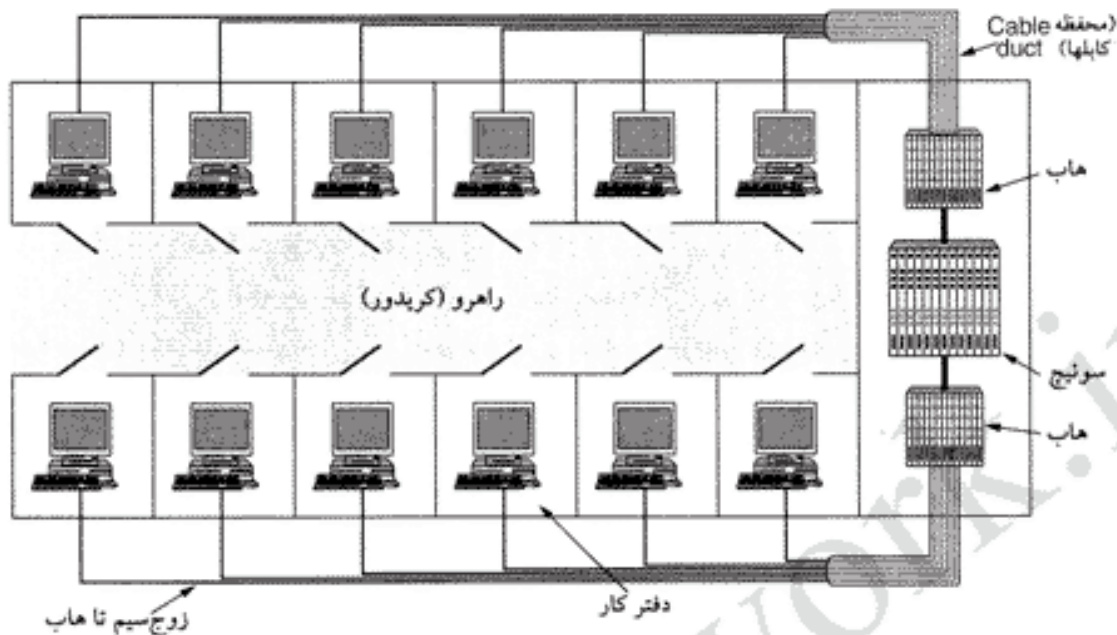
۶-۷-۴ شبکه های محلی مجازی (Virtual LANs)

در دوران اولیه به کارگیری شبکه های محلی، کابلهای زرد رنگ ضخیم از طریق مجراهای مخصوص (duct)، بین دفاتر ساختمانها کشیده می شد و هر کامپیوتری که مسئولین امر تصمیم می گرفتند با وصل به این کابل، به شبکه ملحق می شد. اغلب، کابلهای بی شماری وجود داشت که به یک «ستون فقرات مرکزی» (Central Backbone) یا به یک هاب مرکزی متصل می شد (شکل ۴-۳۹) و این موضوع که کدام کامپیوتر به کدام LAN متصل می شود چندان مهم نبود. تمام افرادی که در دفاتر مجاور هم بودند بر روی یک LAN مشابه قرار می گرفتند، فارغ از آنکه کارشان ارتباطی با هم داشت یا نه! یعنی «منطق برتر جغرافیایی». [بعبارت دیگر به جای آنکه یک منطق علمی، ساختار و اعضای یک LAN را تبیین کند جغرافیای محیط این ساختار و اعضاء را مشخص می کرد.]

با ابداع 10Base-T و هابها در اوایل ۱۹۹۰ همه چیز عوض شد. همه ساختمانها از نو سیم کشی شدند (البته با صرف هزینه قابل توجه) تا تمام کابلهای زرد رنگ با قطر شیلنگ باغبانی برچیده شود و به جای آنها از هر دفتر یک زوج سیم به یک جعبه تقسیم مرکزی واقع در انتهای راهروها یا یک اتاق مرکزی کشیده شود. (شکل ۴-۴۸ را ببینید). اگر معاون رئیس یا مسئول سیم کشی بلند پرواز بود سیم های زوجی رده ۵ (Cat 5) نصب می کرد ولی اگر تنگ نظر بود از سیم کشی موجود خطوط تلفن، استفاده می نمود (که باز هم چند سال بعد و با پدیدار شدن اینترنت سریع باید عوض می شد!)

در اینترنت مبتنی بر هاب (و بعداً مبتنی بر سوئیچ)، اغلب این امکان وجود داشت که LAN ها به جای پیکربندی جغرافیایی به صورت منطقی پیکربندی شوند. اگر شرکتی به k شبکه LAN نیاز داشته باشد، k عدد هاب خریداری می کند. با دقت به آنکه کدام کابل رابط باید به کدام هاب متصل شود اعضای هر LAN را می توان به گونه ای انتخاب کرد که با ساختار سازمانی آنها مطابقت داشته باشد بدون آنکه جغرافیای محل، تاثیر چندانی در این انتخاب بگذارد. البته اگر دو نفر از یک دپارتمان مشابه از سازمان، در ساختمانهای متفاوتی مستقر بودند احتمالاً به دو هاب متفاوت و طبعاً به دو LAN متفاوت متصل می شدند. با این حال، شرایط جدید خیلی بهتر از زمانی است که اعضای یک LAN صرفاً بر اساس جغرافیای محل تعیین شوند.

آیا این مسئله که چه کسی بر روی کدام شبکه LAN قرار گرفته اهمیت دارد؟ مگر نه اینست که سرانجام در هر سازمان تمام LAN ها به هم متصل می شوند؟ کوتاه سخن آنکه جواب مثبت است و مسئله فوق اغلب مواقع اهمیت دارد. مسئولان شبکه بدانند که کاربران را به نحوی بر روی LAN گروه بندی کنند که گروه ها به جای آنکه منعکس کننده نقشه فیزیکی ساختمانها باشند، جلوه ای از ساختار سازمانی باشند. یکی از



شکل ۴۸۴. یک ساختمان با سیم‌کشی مرکزی با بهره‌گیری از هاب و سوئیچ.

موارد و دلایل، «امنیت» است. هر کارت شبکه می‌تواند در حالت «بی‌قید» (Promiscuous) قرار بگیرد و تمام ترافیک جاری بر روی کانال را دریافت نماید. بسیاری از دپارتمانها همانند دپارتمان پژوهش، ثبت و حسابداری، اطلاعاتی را در اختیار دارند که نمی‌خواهند به بیرون از دپارتمان خودشان راه پیدا کند. در چنین شرایطی قرار دادن تمام افراد بر روی یک LAN واحد و جلوگیری از خروج ترافیک از آن LAN، معقول به نظر می‌رسد. مدیریت سازمان تمایلی به شنیدن آنکه «چنین آرایشی ممکن نیست» ندارند مگر آنکه تمام افراد هر دپارتمان، در دفتر هم‌جوار جای داده شده باشند و در کار یکدیگر فضولی نکنند!!

مورد دیگر «میزان بار» است: برخی از LANها نسبت به بقیه، زیاده‌تر مورد استفاده قرار می‌گیرند و ممکن است که تفکیک آنها مطلوب‌تر باشد. به عنوان مثال اگر گروه پژوهش در حال اجرای انواع آزمایشاتی باشند که گاه ترافیکی بیش از اندازه تولید و شبکه LAN را اشباع می‌کند، گروه حسابداری ممکن است علاقمند نباشد که برای کمک به آنان بخشی از ظرفیت [پهنای باند] خود را وقف آنان کند!

مورد سوم، «پخش فراگیر» (Broadcasting) است. اغلب LANها از ارسال فراگیر حمایت می‌کنند و بسیاری از پروتکل‌های لایه‌های بالاتر از این ویژگی در سطح گسترده‌ای استفاده می‌کنند. به عنوان مثال وقتی کاربری می‌خواهد بسته‌ای برای یک ماشین با آدرس IP معادل X بفرستد چگونه آدرس MAC آن ماشین را بدست می‌آورد تا در فریم مربوطه قرار بدهد؟ ما پاسخ این پرسش را در فصل پنجم بررسی خواهیم کرد ولی اگر بخواهیم بطور خلاصه جمع‌بندی نمائیم پاسخ آنست که ماشین فریمی را به صورت پخش فراگیر بر روی شبکه قرار می‌دهد که حاوی این سوال است: «چه کسی صاحب آدرس IP معادل با X است؟»... سپس منتظر پاسخ باقی می‌ماند. نمونه‌های کاربردی زیادی می‌توان یافت که متکی به پخش فراگیر هستند. هر چه LANهای بیشتری به یکدیگر متصل شوند تعداد فریمهای فراگیر که به هر ماشین وارد می‌شوند به صورت خطی و متناسب با تعداد ماشینها افزایش خواهد یافت.

یکی دیگر از مشکلات پخش فراگیر آنست که اگر زمانی یک کارت شبکه از عملکرد طبیعی خود خارج شده و شروع به تولید جریان بی‌پایانی از فریمهای فراگیر نماید تکلیف چیست. نتیجه به پا شدن این «طوفان فریمهای

فراگیر» آنست که (۱) کل ظرفیت LAN با این فریمها، اشغال و تپاه می شود. (۲) تمام ماشینهای واقع در LANهای متصل به هم، به واسطه صرف زمان جهت پردازش و سپس حذف این فریمهای فراگیر، زمین گیر می شوند. در بدو امر ممکن است به نظر برسد که «طوفان فریمهای فراگیر» را می توان با جدا کردن LANها توسط پل یا سوئیچ محدود کرد ولی اگر هدف نهایی «شفافیت» باشد سوئیچها و پلها موظف به هدایت فریمهای پخش فراگیر هستند. (به عبارتی یک ماشین باید بتواند به یک LAN متفاوت تغییر موقعیت بدهد و هیچکس متوجه این موضوع نشود و در محل جدید نیز قادر به پخش فراگیر باشد).

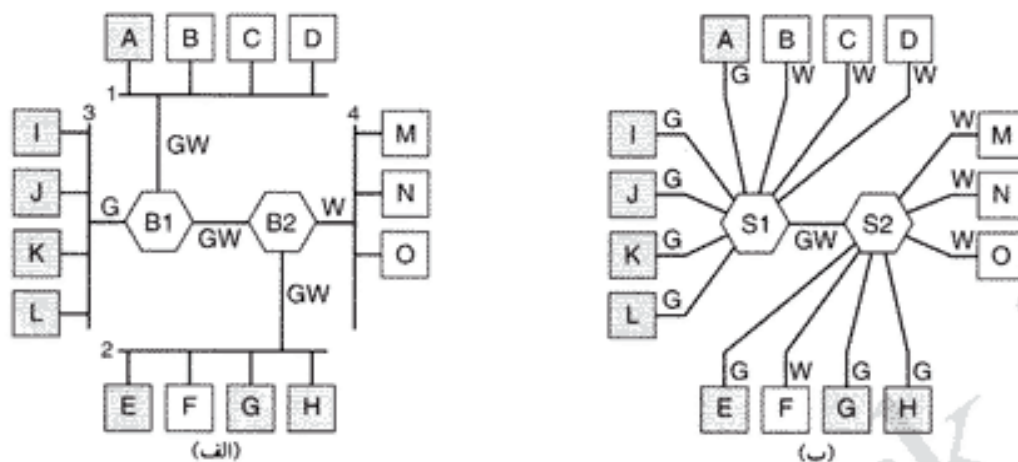
پس از آگاهی از آنکه چرا شرکتها ممکن است بخواهند LANهای متعددی با وسعت محدود داشته باشند، اجازه بدهید به مسئله اصلی یعنی جدا کردن توپولوژی منطقی از توپولوژی فیزیکی بپردازیم. فرض کنید که یک کاربر در یک شرکت بدون تغییر محل دفتر کار خود از یک واحد اداری به واحد دیگر منتقل می گردد یا برعکس، دفتر کار خود را بدون جابجایی از واحد قبلی خود تغییر می دهد. در سیم کشی مبتنی بر هاب، تغییر موقعیت به یک LAN جدید مستلزم آنست که مسئول شبکه به سوی جعبه تقسیم (Wiring Closet) رفته و کابل رابط ماشین آن کاربر را از محل فعلی بیرون کشیده و آنرا به هاب جدید متصل کند.

در بسیاری از شرکتها تغییر و تحولات سازمانی یک امر عادی است و این مستلزم آنست که مسئول شبکه وقت بسیار زیادی را صرف جدا کردن کابلهای رابط و قرار دادن آن در محل جدید نماید. البته در برخی از حالات نیز چنین تغییراتی به هیچ وجه ممکن نیست چرا که مثلاً فاصله ماشین کاربر از هاب جدید بسیار دور است.

در پاسخ به نیاز کاربران به قابلیت انعطاف بیشتر، عرضه کنندگان محصولات شبکه کار بر روی طرحی را آغاز کرده اند که بر اساس آن بتوان سیم کشی ساختمانها را به صورت «نرم افزاری» تغییر داد [یعنی پیکربندی ماشینهای هر LAN فارغ از ساختار اتصال فیزیکی آنها ممکن باشد]. نتیجه چنین نظریه ای «شبکه محلی مجازی» یا VLAN نامیده می شود و توسط کمیته IEEE استاندارد سازی شده است. اکنون بسیاری از سازمانها آرایشی مبتنی بر VLAN دارند. اجازه بدهید نگاهی به آن بیندازیم. برای کسب آگاهی بیشتر مراجع (Breyer and Riley, 1999, Seifert, 2000) مفیدند.

شبکه های VLAN، یکمک سوئیچهای خاص و سازگار با VLAN، پیاده سازی می شوند، هر چند ممکن است همانند شکل ۴-۴۸ دارای چند هاب جانبی و معمولی نیز باشند. برای پیکربندی شبکه مبتنی بر VLAN، مسئول شبکه تصمیم می گیرد که (۱) چند VLAN باید تعریف شود. (۲) چه کامپیوترهایی بر روی هر VLAN قرار می گیرند. (۳) هر یک از VLANها چگونه نامگذاری می شوند. اغلب، یک VLAN با رنگها نامگذاری می شود، (البته به صورت غیررسمی) زیرا بدین ترتیب امکان آنکه بتوان نقشه فیزیکی ماشینهای شبکه را به صورت رنگی چاپ کرد، وجود دارد. اعضای شبکه LAN قرمز با رنگ قرمز مشخص می شوند، اعضای شبکه LAN سبز با رنگ سبز و به همین ترتیب. با این روش نقشه منطقی و فیزیکی شبکه در یک نمای واحد دیده می شود.

به عنوان یک مثال، چهار LAN نشان داده شده در شکل ۴-۴۹ را مد نظر قرار بدهید که در آن هشت ماشین، متعلق به شبکه VLAN خاکستری (G) هستند، هفت تا متعلق به سفید (W). چهار LAN فیزیکی نیز توسط دو پل B1 و B2 بهم متصل شده اند. اگر در این شکل به جای ساختار باس از هابهای مرکزی مبتنی بر زوج سیم استفاده می شد ممکن بود که چهار هاب نیز وجود داشته باشد (که در شکل نشان داده نشده است)، ولیکن از دیدگاه منطقی کابلهای چنداتصال و هاب مشابه به هم هستند. اگر می خواستیم شکل را با هاب ترسیم کنیم، گویانی خود را از دست می داد. امروزه واژه پل بیشتر اشاره به ساختاری شبیه به شکل ۴-۴۹-ب دارد که در آن چندین ماشین از طریق پورتهای آن به هم متصل شده اند در حالیکه امروزه «پل» و «سوئیچ» مفهوم معادلی دارند. در شکل



شکل ۴-۴۹. (الف) چهار LAN فیزیکی با استفاده از دو پل، دو VLAN خاکستری و سفید تشکیل داده‌اند. (ب) همان پانزده ماشین یکمک دو سوئیچ، دو VLAN تشکیل داده‌اند.

۴-۴۹-ب همان تعداد ماشین و همان تعداد VLAN [معادل با شکل ۴-۴۹-الف] فقط با استفاده از سوئیچ نشان داده شده است بگونه‌ای که بر روی هر پورت سوئیچ فقط یک ماشین وجود دارد.

برای آنکه VLAN بدرستی کار کند بایستی یک جدول پیکربندی درون هر پل یا سوئیچ تنظیم شود. این جدول مشخص‌کننده آنست که از طریق کدام یک از پورتها می‌توان به کدام VLAN دسترسی داشت. وقتی فریمی از روی یک VLAN مثلاً خاکستری به سوئیچ وارد شود صرفاً باید بر روی تمام پورتهایی که علامت G دارند منتقل شوند. همانند ترافیک معمولی تک‌پخش (Unicast)، ارسال ترافیک چندپخش (Multicast) و ارسال ترافیک فراگیر (Broadcast) نیز به همین نحو ممکن خواهد بود.

دقت کنید که ممکن است یک پورت برچسب رنگی چندین VLAN را داشته باشد. در شکل ۴-۴۹-الف چنین ساختاری را مشاهده می‌نمایم. فرض کنید ماشین A یک فریم «فراگیر» (Broadcast) را برای همه اعضای VLAN خود ارسال کند. پل B1 این فریم را دریافت کرده و متوجه می‌شود که توسط ماشینی بر روی VLAN خاکستری تولید شده است لذا آن فریم را بر روی تمام پورتهایی که علامت G دارند (به استثنای پورتهایی که فریم از روی آن دریافت شده) ارسال می‌نماید. از آنجایی که B1 فقط دو پورت دیگر با برچسب G دارد لذا فریم را بر روی هر دوی آنها ارسال می‌کند.

در B2 داستان به گونه دیگری است: در اینجا پل می‌داند که هیچ ماشین خاکستری بر روی LAN 4 وجود ندارد لذا فریم بر روی آن هدایت نخواهد شد و فقط به LAN 2 ارسال می‌شود. اگر یکی از کاربران LAN 4 ملزم به تغییر واحد اداری خود شده و به VLAN خاکستری نقل مکان نماید باید جدول B2 بهنگام‌سازی شده و تنها پورت آن که با برچسب W مشخص شده به GW تغییر داده شود. اگر ماشین F به شبکه مجازی خاکستری نقل مکان کند، پورت متصل به LAN 2 باید از GW به G تغییر برچسب داده شود.

حال فرض کنید ماشینهای عضو LAN 2 و LAN 4 همگی به جمع خاکستری‌ها پیوندند، لذا نه تنها پورتهای پل B2 که متصل به LAN 2 و LAN 4 هستند علامت G می‌گیرند بلکه پورت اتصال B1 به B2 نیز از GW به G تغییر علامت می‌دهد چرا که لازم نیست فریمهای سفیدی که از LAN 3 یا LAN 1 می‌رسند به B2 هدایت شوند. در شکل ۴-۴۹-ب همین وضعیت حاکم است و تمام پورتهایی که به یک ماشین واحد متصل هستند با یک برچسب تک‌رنگ مشخص می‌شوند چرا که هر ماشین تنها به یک VLAN متعلق است.

تاکنون فرض را بر آن گذاشته بودیم که پلها و سوئیچها به نحوی می‌دانند که رنگ یک فریم ورودی چیست.

آنها چگونه از این موضوع آگاه می شوند؟ برای این کار سه روش زیر کاربر دارد:

۱. به هر پورت رنگ VLAN انتساب داده شود.
۲. به هر آدرس MAC یک رنگ VLAN منتسب گردد.
۳. به آدرسهای لایه ۲ یا آدرس IP ماشین یک رنگ VLAN منتسب شود.

در روش اول، به هر پورت یک برچسب رنگ داده می شود که این برچسب، VLAN مربوطه را مشخص می کند. با این حال این روش فقط زمانی کار خواهد کرد که تمام ماشینهای متصل به آن پورت، به یک VLAN مشابه متعلق باشند. در شکل ۴-۴۹-الف این ویژگی برای پورت بین پل B1 و LAN 3 صادق است در حالیکه برای پورت متصل به LAN 1 صادق نیست.

در روش دوم، پل یا سوئیچ دارای جدولی است که در آن فهرست آدرسهای ۴۸ بیتی تمام ماشینهای متصل به آن (یعنی آدرس MAC) و مشخصات VLAN هر ماشین، درج می شود. تحت این شرایط، می توان بر روی یک LAN فیزیکی (مثل LAN 1 در شکل ۴-۴۹-الف) چند VLAN مختلط تعریف کرد. وقتی فریمی دریافت می شود، آنچه که هر پل یا سوئیچ باید انجام بدهد آنست که آدرس MAC آن را استخراج و درون جدول به دنبال آن بگردد و ببیند که فریم از کدام VLAN می آید.

در روش سوم، پل یا سوئیچ موظف است تا محتوای فیلد حمل داده (Payload) از هر فریم را بررسی کرده و به عنوان مثال تمام ماشینهای مبتنی بر IP را در یک VLAN و ماشینهای مبتنی بر Apple Talk را در VLAN دیگر دسته بندی کند. در حالت اول می توان برای تشخیص هویت هر ماشین از آدرس IP آن استفاده کرد. این استراتژی زمانی بسیار سودمند خواهد بود که تعداد بی شماری از ماشینهای شبکه کامپیوترهای کیفی هستند و می توانند در هر یک از مکانهای متعدد و مجاز قرار بگیرند. از آنجایی که هر ایستگاه آدرس MAC خاص خود را دارد لذا با توقف ایستگاه در محل جدید، آدرس MAC آن نمی تواند چیزی در مورد آن VLAN که ایستگاه عضو آنست، مشخص کند.

تنها مشکل این روش آنست که یکی از قوانین اساسی در شبکه بندی را نقض می کند: «استقلال لایه ها» تشخیص آنکه چه چیزی در فیلد حمل داده از فریم قرار گرفته برعهده لایه پیوند داده ها نیست. این لایه نباید محتوای داده های درون هر فریم را بررسی کند و یا براساس محتوای درون آن تصمیمی بگیرد. یکی از تبعات به کارگیری این روش آنست که هرگاه در پروتکل لایه ۳ تغییری ایجاد شود (مثلاً IPv4 به IPv6 ارتقاء پیدا کند) سوئیچ بلافاصله از کار خواهد افتاد. متأسفانه سوئیچهایی که بدین نحو کار می کنند به وفور در بازار عرضه شده اند. البته تعریف VLAN بر اساس آدرس IP، اشکالی در مسیریابی بسته های IP بوجود نخواهد آورد ولیکن تلفیق وظایف لایه ها سرآغاز بروز مشکلات پیش بینی نشده خواهد بود. (تقریباً کل فصل پنجم به مسیریابی IP اختصاص دارد). شاید عرضه کنندگان سوئیچ چنین استدلالی را به مسخره بگیرند و بگویند سوئیچهای آنه IPv4 و IPv6 را به رسمیت می شناسند و همه چیز به درستی پیش خواهد رفت ولیکن اگر زمانی IPv7 مطرح شد چه اتفاقی می افتد؟ شاید فروشنده بگوید که در آن زمان سوئیچهای جدید بخرید! آیا این کار خیلی شاق است؟

استاندارد IEEE 802.1Q

اندکی اندیشه بیشتر در خصوص VLAN ما را به این نتیجه می رساند که آنچه واقعاً اهمیت دارد آنست که هر فریم ارسالی نام VLAN خود را با خود حمل کند نه آنکه VLAN ماشین فرستنده را سوئیچ بروشهای دیگری مشخص کند. اگر روشی برای مشخص کردن VLAN در سرآیند هر فریم وجود داشته باشد دیگر نیازی به بررسی داده های

درونی هر فریم نخواهد بود. برای شبکه‌های جدیدی مثل 802.11 یا 802.16، اضافه کردن فیلد VLAN به سرآیند هر فریم نسبتاً ساده است. در حقیقت فیلد Connection Identifier در 802.16 ذاتاً چیزی شبیه به VLAN Identifier است. ولی در مورد اترنت چه کاری می‌توان کرد؟ شبکه‌ای که رایجترین شبکه دنیاست و هیچ فیلد اضافی برای تعبیه VLAN Identifier در آن، تعریف نشده است.

کمیته 802 IEEE این مسئله را در دستور کار خود قرار داد و پس از مباحثات فراوان کاری غیرقابل تصور انجام داد و سرآیند اترنت را عوض کرد. قالب جدید فریم، در استاندارد IEEE 802.1Q تدوین و در سال ۱۹۹۸ منتشر شد. در قالب جدید، هر فریم دارای یک برچسب VLAN است (یا نام VLAN Tag) که مختصراً آن را بررسی خواهیم کرد. متأسفانه تغییر در استاندارد دی مثل سرآیند اترنت که کاملاً جا افتاده و بطور رایج از آن استفاده می‌شود، چندان ساده نیست. در این خصوص ممکن است سوالات زیر به ذهن خطور کند:

۱. آیا باید چندین میلیون کارت اترنت موجود را دور انداخت؟
۲. اگر نه، چه کسی فیلدهای جدید را تولید نماید؟
۳. برای فریمهایی که اندازه حداکثر دارند چه اتفاقی می‌افتد؟ [چون نمی‌توان به فریمی که بر طول حداکثر آن محدودیت گذاشته شده، داده‌ای افزود].

البته کمیته 802 از این مشکلات آگاه بود و می‌بایست راه حلی مناسب ارائه می‌کرد، کاری که نهایتاً انجام شد. نکته اصلی در راه حل ارائه شده آنست که فیلدهای VLAN فقط توسط پلها و سوئیچها مورد استفاده قرار می‌گیرد و ماشینهای کاربران بدان نیازی ندارند. بدین ترتیب در شکل ۴-۴۹ وقتی فریمها مستقیماً به سوی یک ایستگاه پایانی ارسال می‌شوند به این فیلدها نیازی نیست و فقط روی خطوطی بین پلها یا سوئیچها، بکار می‌آیند. بنابراین برای بکارگیری VLAN، این پلها یا سوئیچها هستند که باید VLAN را به رسمیت بشناسند؛ این موضوع نیز در پلها و سوئیچها از قبل پیش‌بینی شده و یک نیاز تلقی می‌شود. حال باید نیازمندیهایی جدیدی را معرفی نمایم که 802.1Q بدانها پاسخ داده است.

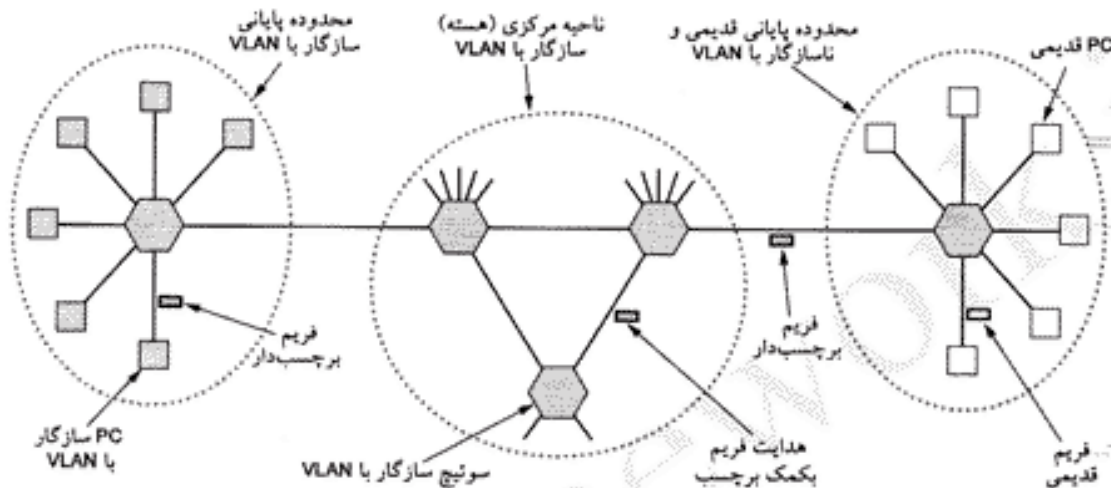
پاسخ به این سوال که «آیا باید تمام کارتهای اترنت موجود را دور انداخت؟» منفی است. به خاطر داشته باشید که کمیته 802.3 حتی نتوانست به افراد بقبولاند که فیلد Type را به فیلد Length تغییر بدهند. حال شما تصور کنید عکس‌العمل مردم در قبال اعلام دور انداختن کارتهای اترنت، چه می‌شد! البته وقتی کارتها جدید به بازار بیایند شاید بتوان امیدوار بود که سازگار با 802.1Q بوده و فیلدهای مربوط به VLAN در آنها تعبیه شده باشند.

به هر حال اگر کارت شبکه تولیدکننده یک فریم، فیلدهای VLAN را بدان اضافه نکند، چه کسی باید این کار را انجام بدهد؟ پاسخ این سوال آنست که اولین پل یا سوئیچ مبتنی بر VLAN که فریم را دریافت می‌کند، این فیلدها را به فریم افزوده و آخرین آنها در مسیر، این فیلدها را حذف می‌نماید. ولیکن یک سوئیچ یا پل از کجا می‌فهمد که یک فریم به کدام VLAN تعلق دارد؟ پاسخ آنست که اولین پل یا سوئیچ می‌تواند به هر یک از پورتهای خود یک شماره VLAN نسبت بدهد یا آنکه به آدرس MAC آن فریم نگاه کند و یا محتوای داده‌های درونی را بررسی کند (کار ممنوع!).

در خصوص کارتهای اترنت که سازگار با 802.1Q هستند، مشکلی وجود ندارد. انتظار قابل تحقق آنست که تمام کارتهای اترنت گیگابیت از همان ابتدا سازگار با 802.1Q باشند و با ارتقاء کارتهای قدیمی به اترنت گیگابیت، 802.1Q نیز به صورت خودکار جای خود را باز کند. برای حل این مشکل که فریمهای اترنت نباید از 1518 بایت بیشتر باشند در 802.1Q طول حداکثر به ۱۵۲۲ بایت افزایش یافته است.

در دوران گذار از اترنت فعلی به اترنت گیگابیت، در بسیاری از شبکه‌ها برخی از ماشینهای قدیمی (عموماً

اترنت کلاسیک و اترنت سریع) با VLAN سازگار نیستند، در حالیکه برخی دیگر (عموماً اترنت گیگابت) از آن پشتیبانی می کنند. چنین وضعیتی در شکل ۴-۵۰ به تصویر کشیده شده است و در آن نمادهای خاکستری سازگار با VLAN و نمادهای بی رنگ ناسازگار هستند. برای سادگی بحث، فرض را بر آن گذاشته ایم که تمام سوئیچها با VLAN سازگار هستند. اگر اینگونه نباشد اولین سوئیچ سازگار با VLAN بکمک آدرس MAC یا آدرس IP درون فریم، برچسب لازم را بدان خواهد افزود.



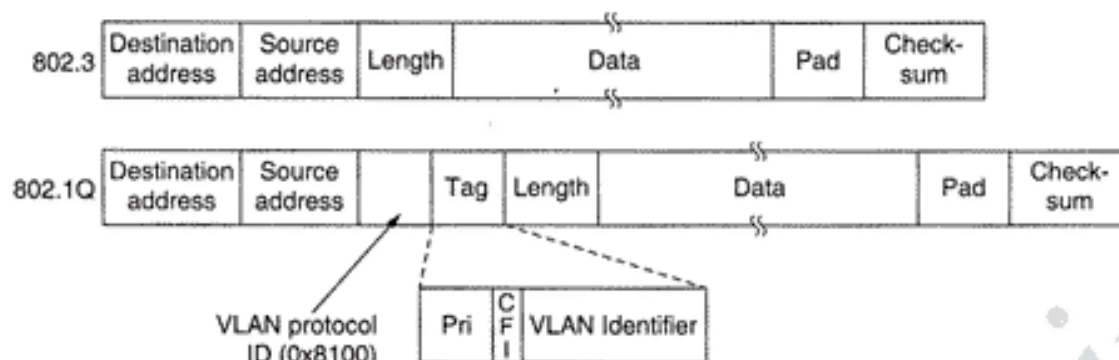
شکل ۴-۵۰. گذر از اترنت قدیمی به اترنت سازگار با VLAN. نمادهای خاکستری با VLAN سازگارند؛ نمادهای بی رنگ سازگار نیستند.

در این شکل کارتهای اترنت سازگار با VLAN، مستقیماً فریمهای برچسب دار VLAN (یعنی فریمهای 802.1Q) را تولید می نمایند و سوئیچهای بعدی از این برچسب استفاده خواهند کرد. برای عملیات هدایت فریمها (یعنی عمل سوئیچنگ) هر سوئیچ بایستی بداند که کدام VLAN از طریق کدام پورت در دسترس است. (طبق توضیحات قبلی) دانستن آنکه فریم جاری به VLAN مثلاً خاکستری تعلق دارد فایده چندانی نخواهد داشت مگر آنکه سوئیچ بداند کدام پورتها به ماشینهای عضو VLAN خاکستری متصل هستند. بنابراین سوئیچ نیازمند به داشتن جدولی اندیس دار و مرتب شده بر حسب مشخصه VLAN است که تعیین می کند از کدامیک از پورتها باید استفاده شود و آیا ماشینهای متصل بدین پورتها سازگار با VLAN هستند یا قدیمند.

وقتی یک PC قدیمی فریمی را برای یک سوئیچ سازگار با VLAN می فرستند، سوئیچ مربوطه با استفاده از دانش قبلی^۱ خود در خصوص VLAN متناظر با فرستنده فریم (مثلاً با استاندارد به شماره پورت فیزیکی، آدرس MAC یا آدرس IP) برچسب لازم را به فریم چسبانده و فریم جدیدی تولید می نماید. از این دیدگاه، دیگر قدیمی بودن ماشین فرستنده هیچ اهمیتی ندارد. به روش مشابه هرگاه نیاز باشد فریمی برچسب دار (مبتنی بر 802.1Q) تحویل ماشین قدیمی شود قبل از تحویل، ساختار فریم به قالب بدون برچسب تبدیل خواهد شد.

حال بیایید نگاهی به قالب فریم 802.1Q بیندازیم. قالب این نوع فریم در شکل ۴-۵۱ نشان داده شده است. تنها تغییر، اضافه شدن یک جفت فیلد ۲ بایتی (جمعاً ۴ بایت) به قالب قدیمی است. یکی از آنها فیلد مشخصه VLAN Protocol ID است که همیشه مقدار 0x8100 دارد. از آنجایی که این مقدار بیش از ۱۵۰۰ است تمام

۱. به خاطر داشته باشید که سوئیچهای مدرن سازگار با VLAN، بصورت نرم افزاری پیکربندی می شوند و بعضاً دارای سیستم عامل و فرامین مخصوص هستند. بنابراین در خصوص ماشینهای با کارت اترنت قدیمی تنظیمات VLAN، باید بصورت دستی انجام شود. -م



شکل ۵۱-۴. قالب فریم قدیمی اترنت 802.3 و فریم 802.1Q.

کارتهای اترنت آنرا به عنوان فیلد «نوع فریم» (Type) تفسیر می کنند و به فیلد طول داده Length تعبیر نخواهد شد. عکس العملی که کارتهای اترنت قدیمی در برخورد با چنین فریمهایی از خود نشان می دهند جای بحث دارد چراکه فرض بر آن است که چنین فریمهایی برای کارتهای قدیمی ارسال نمی شود.

فیلد دو بیتی بعدی شامل سه زیرفیلد است که اصلی ترین آنها یعنی VLAN Identifier (شناسه VLAN)، ۱۲ بیت کم ارزش را به خود اختصاص داده است. این فیلد کل آن چیزی است بدان نیازمند بوده ایم: یعنی مشخص می کند که «این فریم به کدام VLAN تعلق دارد». فیلد ۳ بیتی Priority (اولویت) فعال کاری در خصوص VLAN بر عهده ندارد و عملاً بلا استفاده است. استدلال کمیته 802.1Q در تعریف این زیرفیلد آن بوده که چون تغییر در سرآیند اترنت فرآیند دشوار و زمانبری است، حالا که این کار خطیر میسر شده چرا چیزهای خوب دیگر نیز بدان اضافه نشود؟ این فیلد ۳ بیتی این امکان را فراهم آورده تا بتوان «ترافیک بی درنگ از نوع سخت و نرم» (Soft & Hard Realtime Traffic) [مثل صدا و تصویر یا نظائر آن] را از ترافیک غیر حساس به زمان [مثل ترافیک ارسال نامه های الکترونیکی] تشخیص داد و امکان ارائه «کیفیت خدمات» (QoS) در اترنت نیز مسیر شود. در آینده به این فیلد جهت ارسال صدا بر روی اترنت نیاز خواهد شد. (هر چند مشابه با همین فیلد در پروتکل IP ربع قرن قبل در نظر گرفته شده بود ولیکن هیچگاه از آن استفاده نشد!!)

آخرین زیرفیلد، بیت CFI (Canonical Format Indicator) است که در حقیقت می بایست بیت CEI (Corporate Ego Indicator) بمعنای مشخصه شخصی شرکت نامیده می شد. اساساً این بیت باید برای مشخص کردن آدرسهای Big Endian از آدرسهای Little Endian بکار می رفت ولی پس از بحث و جدل فراوان از آن صرف نظر شد.^۱ اکنون معنای این بیت آن است که درون فیلد حمل داده از فریم اترنت، یک فریم کامل از نوع 802.5 (فریم شبکه Token Ring) قرار دارد و انتظار می رود به یک شبکه محلی از نوع 802.5 دیگر تحویل شود چراکه اتصال آنها از طریق اترنت برقرار شده است. (به عبارت دیگر این فیلد مشخص می کند که شبکه اترنت باید به عنوان حامل میانی، یک فریم نوع 802.5 را به شبکه ای دیگر از همین نوع تحویل بدهد).^۲ البته این بیت هیچ کاری در مورد شبکه VLAN انجام نمی دهد ولیکن سیاستهای کمیته استاندارد سازی IEEE متفاوت با سیاستهای عادی است!!!

با دقت در توضیحات فوق، وقتی یک فریم برچسب دار به یک سوئیچ سازگار با VLAN می رسد آن سوئیچ از

۱. معنای آدرس Big Endian آن است که بایتهای پر ارزش آدرس در ابتدا و بایتهای کم ارزش در انتها قرار گرفته و به همان نحو ارسال می شوند. معنای آدرس Little Endian آن است که بایتهای کم ارزش در ابتدا و بایتهای پر ارزش آدرس بعد از آن ارسال می شوند.

۲. یعنی نوعی از پدیده تونل زنی (Tunneling) در سطح لایه پیوند داده. هم

فیلد VLAN ID به عنوان اندیس (یا عبارتی کلید جستجو) در جدول خود استفاده کرده و به کمک آن پورتهایی که این فریم باید بر روی آنها فرستاده شود را مشخص می کند. ولی سوال اساسی اینجاست که این جدول از کجا بدست می آید؟ اگر این جدول باید به صورت دستی تنظیم شود به پله شماره صفر برگشته ایم یعنی «پیکربندی دستی پلها»؛ زیبایی پلهای شفاف در آن است که به محض وصل، فوراً به کار می افتند (یعنی Plug & Play است) و نیازی به پیکربندی دستی ندارند. از دست دادن این ویژگی بسیار ناگوار است! خوشبختانه پلهای سازگار با VLAN می توانند خود را بر اساس برچسب فریمهای ورودی به صورت خودکار پیکربندی کنند. مثلاً اگر فریمی با برچسب VLAN 4 از روی پورت شماره 3 وارد شود، به روشنی مشخص است که ماشین متصل به پورت 3 متعلق به VLAN 4 است. استاندارد 802.1Q روش ایجاد پویای این جداول را تشریح کرده ولی در اکثر جاهای این توضیحات خواننده را به بخشهایی خاص از الگوریتم پرلن ارجاع داده است؛ الگوریتم پرلن نیز در 802.1D استاندارد شده است.

قبل از خاتمه بحث مسیریابی در VLAN، اشاره به یک نکته پایانی خالی از لطف نیست. بسیاری از افراد در دنیای اینترنت و اترنت طرفدار ساختار شبکه های «بی اتصال» (Connectionless) هستند و در مقابل هر چیزی که زمینه نیاز به «اتصال» در لایه پیوند داده یا لایه شبکه را فراهم کند به شدت مقاومت می کنند. در VLAN چیزهایی معرفی شده که شبیه به مفهوم «اتصال» هستند. برای عملکرد صحیح VLAN هر فریم با خود یک مشخصه خاص و جدید دارد [یعنی همان VLAN ID] که از آن بعنوان اندیس جستجو در جدول درون سوئیچ، استفاده می شود. این دقیقاً همان اتفاقی است که در شبکه های اتصال گرا می افتد. در شبکه های بدون اتصال به تنها چیزی که برای مسیریابی و هدایت فریم نیاز است آدرس MAC می باشد و به هیچ نوع مشخصه اتصال یا امثال آن نیاز نیست. در فصل پنجم در خصوص مفاهیم اتصال گرایی (Connectionism) بیشتر خواهیم آموخت.

۸-۴ خلاصه

برخی از شبکه ها تنها دارای یک کانال انتقال مشترک هستند که همه ایستگاهها از آن، برای مبادله داده های خود استفاده می کنند. در این شبکه ها، مسئله اصلی در طراحی لایه پیوند داده، تسهیم و تخصیص کانال بین ایستگاه های رقیبی است که علاقمندند از آن استفاده نمایند. الگوریتمهای بی شماری برای مسئله تخصیص کانال ابداع شده است. خلاصه ای از مهمترین روشهای تخصیص کانال در جدول ۴-۵۲ فهرست شده است.

ساده ترین روش تخصیص، FDM و TDM است. این روشها زمانی مفیدند که تعداد ایستگاهها کم و ثابت و ترافیک آنها پیوسته باشد. هر دوی این روشها در محیطهایی با این ویژگی مثلاً برای تقسیم پهنای باند در خطوط اصلی تلفن (شاهراهها)، کاربرد دارند. وقتی تعداد ایستگاهها زیاد و متغیر یا ترافیک آنها نسبتاً انفجاری است روش های TDM و FDM گزینه های مناسبی نیستند. پروتکل ALOHA (چه نوع معمولی و چه نوع Slotted آن) بعنوان روشهای جایگزین TDM و FDM معرفی شدند. گونه های مختلف ALOHA نیز تشریح و تحلیل شده اند. برخی از این گونه ها امروزه در سیستمهای واقعی به کار می روند.

وقتی بتوان حالت فعلی کانال را شنود (احساس) کرد، ایستگاهها قادر خواهند بود مادامیکه ایستگاه دیگر در حال ارسال است از ارسال فریم خود اجتناب کنند. این تکنیک یعنی «شنود سیگنال حامل» منتج به ابداع پروتکلهای متنوعی شد که می تواند در شبکه های LAN و MAN به کار گرفته شود.

رده ای از پروتکلها قادرند مسئله رقابت بر سر تصرف کانال را متفی کنند یا حداقل این رقابت را تا حد قابل توجهی کاهش بدهند. روش «شمارش دودونی معکوس» (Binary Countdown) رقابت را بطور کامل حذف می کند. پروتکل «پیمایش درخت و فقی» (Adaptive Tree Walk) با تقسیم بندی پویای ایستگاهها به گروه های از

روش	توصیف عملکرد
FDM	به هر ایستگاه یک باند فرکانسی تخصیص می‌دهد.
WDM	الگوی پویای FDM برای فیبر نوری
TDM	به هر ایستگاه یک برش زمان تخصیص می‌دهد.
Pure ALOHA	ارسال ناهماهنگ در هر لحظه دلخواه
Slotted ALOHA	ارسال تصادفی در پرشهای زمانی مشخص
1-persistent CSMA	استاندارد دسترسی چندگانه مبتنی بر شنود کانال
Nonpersistent CSMA	تاخیر تصادفی در هنگامی که کانال اشغال تشخیص داده می‌شود.
P-persistent CSMA	همان CSMA است با این تفاوت که احتمال اصرار در ارسال فریم P است.
CSMA/CD	همان CSMA است با این تفاوت که بمحض تشخیص تصادم ادامه ارسال را قطع می‌کند.
Bit map	زمان‌بندی به نوبت و چرخشی (Round-Robin) یکم یک الگوی بیتی (Bitmap)
Binary countdown	ایستگاه با بزرگترین شماره حق ارسال در نوبت بعدی را دارد.
Tree walk	رقابت محدود با فعالیتهای انتخابی (تعریف گروه‌های رقیب در ساختار درختی)
MACA, MACAW	پروتکل‌های شبکه‌های محلی بی‌سیم
Ethernet	همان CSMA/CD با الگوریتم عقبگرد نمایی
FHSS	جهش‌های فرکانسی در طیف گسترده (Frequency Hopping Spread Spectrum)
DSSS	Direct Sequence Spread Spectrum
CSMA/CA	استاندارد دسترسی چندگانه مبتنی بر شنود کانال و اجتناب از تصادم

شکل ۴-۵۲. روشها و سیستمهای تخصیص یک کانال مشترک.

هم جدا، مشکل رقابت را بطور جدی کاهش می‌دهد. تقسیم‌بندی گروه‌ها حتی‌الامکان به نحوی است که فقط گروههایی که در آنها فقط یک ایستگاه فریمی برای ارسال آماده دارد مجاز به ارسال هستند. شبکه‌های محلی بی‌سیم مشکلات و راه حل‌های خاص خود را دارند. عمده‌ترین مشکل توسط ایستگاه‌های «پنهان» ایجاد می‌شود و بدین دلیل CSMA جوابگوی چنین شبکه‌هایی نخواهد بود. رده‌ای از این راه‌حلها که در طبقه MACA و MACAW دسته‌بندی شده‌اند سعی دارند ایستگاه‌ها را وادار به ارسال در پیرامون ماشین مقصد نمایند تا بدین نحو CSMA عملکرد بهتری داشته باشد. روش‌های مبتنی بر طیف گسترده از نوع Frequency Hopping و Direct Sequence نیز در این شبکه‌ها به کار گرفته شده است. IEEE 802.11 روش CSMA و MACAW را تلفیق کرد و روش CSMA/CA را معرفی نمود.

اترنت رایجترین نوع شبکه‌های محلی است که برای تخصیص کانال از روش CSMA/CD بهره می‌گیرد. نسخه قدیمی آن از یک کابل واحد که بین تمام ماشینها کشیده می‌شد، استفاده می‌کردند، در حالیکه اکنون استفاده از هاب، سوئیچ و کابل‌های زوجی بسیار رایج است. سرعت آن از ۱۰ مگابیت بر ثانیه تا یک گیگابیت بر ثانیه افزایش یافته و هنوز هم در حال افزایش است.

امروز شبکه‌های محلی بی‌سیم نیز در حال رواج هستند که در این بین استفاده از 802.11 وسیعتر است. لایه فیزیکی در این شبکه، امکان پنج نوع انتقال و مدولاسیون متفاوت را فراهم آورده است که شامل مادون قرمز، گونه‌های متفاوت مبتنی بر طیف گسترده و «سیستم FDM چندکاناله» است. این شبکه می‌تواند یک ایستگاه ثابت در هر سلول داشته باشد ولیکن قادر است بدون ایستگاه ثابت نیز کار کند. این پروتکل گونه‌ای از MACAW و متکی بر شنود مجازی سیگنال حامل است.

شبکه‌های بین شهری بی‌سیم (Wireless MAN) در حال پیدایش و رواج هستند. این گونه شبکه‌ها

سیستمهای باند گسترده‌ای هستند که از امواج رادیویی بهره می‌گیرند تا جایگزین ارتباطات تلفنی شوند. در آنها از تکنیک‌های مدولاسیون باند باریک استفاده شده است. کیفیت خدمات نیز از موارد مهمی است که استاندارد 802.16 چهار رده مختلف از این گونه خدمات را تعریف کرده است: ارسال با نرخ ثابت، دو روش ارسال با نرخ متغیر و روش ارسال مبتنی بر بهترین تلاش (Best Effort).

سیستم بلوتوث نیز بی‌سیم است ولیکن هدف آن وصل ابزارهای رومیزی است؛ مثلاً برای اتصال گوشیها و ابزارهای جانبی کامپیوترهای شخصی بدون نیاز به سیم کاربرد دارد. همچنین به منظور وصل ابزارهایی مثل دورنگار (فکس) و تلفن‌های همراه به کار می‌آید. بلوتوث شبیه به 802.11 از تکنیک طیف گسترده مبتنی بر پرس فرکانس بهره گرفته و در باند ISM [یعنی 2.4 Ghz] کار می‌کند. به خاطر بالا بودن سطح نویز در بسیاری از محیطها و نیاز به فعل و انفعال بی‌درنگ، در پروتکل‌های مختلف آن از روش تصحیح خطای مستقیم (Forward Error Correction) بهره گرفته شده است.

بدلیل وجود انواع متفاوت LAN به روشی جهت اتصال آنها به یکدیگر نیاز است. از پل و سوئیچ به همین منظور استفاده می‌شود. در پلهای نوع Plug & Play از «الگوریتم درخت پوشا» (Spanning Tree) استفاده شده است. پیشرفت جدید در دنیای شبکه‌ها، VLAN است که کمک آن توپولوژی منطقی LANها از توپولوژی فیزیکی آن جدا می‌شود. قالب جدیدی برای فریمهای اترنت معرفی شده است تا راحتتر بتوان VLAN را در سازمان‌ها پیاده سازی کرد.

مسائل

۱. برای حل این مسئله از رابطه‌ای که در همین فصل آمده بهره بگیرید ولیکن در ابتدا آن را بیان نمائید. بطور تصادفی فریمهایی برای انتقال بر روی کانالی با نرخ ارسال 100 Mbps تولید می‌شوند. اگر در زمان تولید یک فریم کانال اشغال بود تا رسیدن نوبت به آن در صف منتظر خواهد ماند. طول فریمها دارای تابع توزیع نمایی با میانگین 10000 bits/Frame است. برای هر یک از نرخهای تولید فریم که در زیر مشخص شده، تأخیری را که هر فریم (با طول متوسط) با آن مواجه می‌شود (شامل زمان انتظار صف و زمان انتقال) حساب نمائید.

الف) ۹۰ فریم در ثانیه

ب) ۹۰۰ فریم در ثانیه

ج) ۹۰۰۰ فریم در ثانیه

۲. یک گروه N تایی از ایستگاه‌ها دارای کانالی مشترک با نرخ 56kbps هستند و روش تخصیص کانال نیز Pure ALOHA است. بطور متوسط هر ایستگاه در هر صد ثانیه یک فریم هزار بیتی تولید می‌کند (حتی اگر فریم قبلی هنوز ارسال نشده باشد چرا که ایستگاه‌ها می‌توانند فریمهای خروجی را در بافر خود نگاه دارند). مقدار حداکثر N چقدر می‌تواند باشد؟

۳. تأخیر Pure ALOHA را در مقایسه با روش Slotted ALOHA و در شرایط بار پائین مد نظر قرار بدهید. تأخیر کدامیک کمتر است؟ پاسخ خود را تشریح کنید.

۴. ده هزار ایستگاه رزرو بلیط هواپیما، برای استفاده از یک کانال واحد، بروش Slotted ALOHA با هم رقابت می‌کنند. هر ایستگاه بطور متوسط ۱۸ تقاضا در هر ساعت خواهد داشت. برشهای زمانی (Time Slot) ۱۲۵ میکروثانیه‌ای هستند. مقدار تقریبی بار کل کانال چقدر است؟

۵. جمع کثیری از کاربران سیستم ALOHA در هر ثانیه ۵۰ تقاضا تولد می‌نمایند (که این مقدار شامل تعداد

- تقاضاهای جدید و تقاضاهای ارسال مجدد فریمهایی است که قبلاً در اثر تصادم خراب شده‌اند). زمان به برشهای متساوی ۴۰ میلی‌ثانیه‌ای تقسیم شده است:
- الف) احتمال موفقیت ارسال در همان دفعه اول چقدر است؟
- ب) احتمال بروز دقیقاً k تصادم پیاپی و سپس یک موفقیت چقدر است؟
- ج) به طور متوسط برای ارسال یک فریم چند بار تلاش لازم است؟
۶. اندازه‌گیری پار کانال Slotted ALOHA با تعداد نامحدود کاربر، نشان می‌دهد که ده درصد از برشهای زمانی بلااستفاده مانده‌اند:
- الف) بار کانال یعنی G چقدر است؟
- ب) بازده مفید کانال (Throughput) چقدر است؟
- ج) آیا بار کانال بیش از اندازه بالاست یا کمتر از حد متعادل است؟
۷. در یک سیستم Slotted ALOHA با کاربران نامحدود، متوسط تعداد برش زمانی که هر ایستگاه بین تصادم و ارسال مجدد صبر می‌کند، ۴ است. منحنی «تاخیر» را بر حسب «بازده مفید کانال» (Delay Versus throughput) در این سیستم ترسیم نمایید.
۸. یک ایستگاه مثل S را در شبکه‌ای با پروتکل‌های زیر در نظر بگیرید. این ایستگاه قبل از ارسال فریمش بر روی کانال، (در بدترین حالت) چه مدت زمانی باید انتظار بکشد:
- الف) پروتکل Basic Bit Map
- ب) پروتکل‌های Mok و Wark با جایگشت مجازی شماره ایستگاه‌ها
۹. یک LAN از پروتکل «شمارش دودویی معکوس» نسخه Mok و Wark، بهره می‌گیرد. در یک لحظه خاص، ده ایستگاه دارای شماره‌های مجازی ۸، ۲، ۴، ۵، ۱، ۷، ۳، ۶، ۹ و ۵ هستند. سه ایستگاه ارسال‌کننده بعدی به ترتیب عبارتند از: ۴، ۳ و ۹. پس از آنکه این ایستگاه‌ها ارسال خود را به اتمام رساندند شماره مجازی ایستگاه‌ها چند خواهد بود؟ [ترتیب شماره‌ها را از راست به چپ در نظر بگیرید].
۱۰. شانزده ایستگاه که از ۱ تا ۱۶ شماره‌گذاری شده‌اند برای استفاده از یک کانال اشتراکی بروش «پیمایش درخت وقفی» رقابت می‌کنند. اگر تمام ایستگاه‌هایی که آدرس آنها اعداد اول است بطور ناگهانی آماده ارسال شوند برای خاتمه مراحل رقابت چند «برش بیتی» (Bit Slots) نیاز خواهد بود؟
۱۱. مجموعه‌ای از 2^n ایستگاه برای دسترسی به یک کابل اشتراکی از روش «پیمایش درخت وقفی» بهره گرفته‌اند. در یک زمان خاص دو ایستگاه آماده ارسال می‌شوند. اگر $2^{n-1} > 1$ باشد مقدار حداقل، حداکثر و میانگین تعداد برشهای زمانی برای پیمایش این درخت [و خاتمه رقابت] چقدر است؟
۱۲. در شبکه محلی بی‌سیم که مطالعه کردیم به جای استفاده از پروتکل CSMA/CD از پروتکل نظیر MACA استفاده شده بود. تحت چه شرایطی (در صورت امکان) استفاده از روش CSMA/CD میسر خواهد بود؟
۱۳. چه ویژگی‌هایی از پروتکل‌های دسترسی به کانال در GSM و WDMA مشترک هستند؟ GSM را در فصل ۲ مرور نمایید.
۱۴. شش ایستگاه A تا F با بکارگیری پروتکل MACA با یکدیگر در ارتباط و تبادل داده هستند. آیا امکان انتقال همزمان اطلاعات در این شبکه وجود دارد؟ (پاسخ خود را شرح بدهید).
۱۵. در هر طبقه از یک ساختمان هفت طبقه، ۱۵ دفتر کار همجوار وجود دارد. در هر دفتر یک پریز دیواری (Wall Socket) برای وصل یک پایانه در جلوی آن تعبیه شده است و بدین ترتیب پریزها یک مستطیل در چهارگوش یک صفحه قائم تشکیل داده‌اند که فاصله بین پریزها به صورت عمودی و افقی چهار متر است.

- فرض کنید امکان کابل‌کشی مستقیم بین هر جفت پریز، چه عمودی، افقی یا مورب وجود داشته باشد. برای اتصال این پریزها به هم، به چند متر کابل نیاز است اگر:
- الف) پیکربندی مبتنی بر «ستاره» و با یک مسیر یاب در وسط، مدنظر باشد.
- ب) شبکه محلی 802.3 مدنظر باشد.
۱۶. نرخ تغییرات سیگنال (یعنی Baud Rate) در استاندارد ده‌مگابیتی اترنت چقدر است؟
 ۱۷. سیگنال حاصل از کدینگ «منچستر» را برای رشته بیت 0001110101 نشان بدهید.
 ۱۸. سیگنال حاصل از کدینگ «منچستر تفاضلی» را برای رشته بیت مسئله بالا ترسیم نمایید. فرض کنید خط در هنگام شروع به ارسال در حالت LOW قرار داشته باشد.
 ۱۹. در یک شبکه محلی 10Mbps مبتنی بر CSMA/CD (البته نه 801.3) با طول یک کیلومتر، سرعت انتشار سیگنال ۲۰۰ متر بر میکروثانیه است. طول فریمهای داده ۲۵۶ بیت شامل مجموعاً ۳۲ بیت سرآیند، کد کشف خطا و سربارهای دیگر است. اولین برش بیتی (Bit Slot) پس از ارسال موفق، برای گیرنده فریم در نظر گرفته شده تا بتواند برای ارسال پیام ACK (تصدیق دریافت فریم) کانال را در اختیار بگیرد. نرخ ارسال مفید (بدون در نظر گرفتن سربار و با فرض عدم وجود تصادم) چقدر است؟
 ۲۰. دو ایستگاه CSMA/CD تلاش می‌کند تا یک فایل بزرگ (شامل چندین فریم) را ارسال نمایند. پس از ارسال هر فریم، آنها برای دسترسی به کانال طبق الگوریتم «عقب‌گرد نمایی» با یکدیگر رقابت می‌کنند. احتمال اینکه هر رقابت در k دور به اتمام برسد و همچنین تعداد متوسط دورها (rounds) در هر رقابت چقدر است؟
 ۲۱. ساختمانی را در نظر بگیرید که در آن شبکه‌ای CSMA/CD با سرعت 1Gbps و یک کیلومتر کابل، بدون هیچ تکرارکننده (Repeater) کار می‌کند. سرعت انتشار سیگنال روی کابل ۲۰۰۰۰۰ کیلومتر بر ثانیه است. مقدار حداقل طول هر فریم چقدر است؟
 ۲۲. طول یک بسته IP جهت ارسال بر روی اترنت، ۶۰ بایت (شامل کل سرآیندها) است. اگر از LLC استفاده نشده باشد آیا در فریم اترنت به عمل اضافه کردن داده‌های زائد (Padding) نیاز است و اگر نیاز است چند بایت؟
 ۲۳. طول فریمهای اترنت باید حداقل ۶۴ بایت باشد تا این اطمینان حاصل شود که در صورت تصادم در انتهای کابل، فرستنده کماکان در حال ارسال است. [تا بتواند قبل از خاتمه فریم تصادم را کشف کند]. حداقل طول فریمها در «اترنت سریع»، ۶۴ بایت است در حالی که ارسال بیتها بر روی خروجی ۱۰ برابر سریعتر شده است. چگونه ممکن بوده که بتوان طول حداقل فریمها را در همان ۶۴ بایت نگه داشت؟
 ۲۴. در برخی از کتابها طول حداکثر هر فریم اترنت ۱۵۱۸ بایت ذکر شده است. آیا اشتباهی در کار است؟ پاسخ خود را تشریح نمایید.
 ۲۵. در تشریح مشخصات 1000Base-SX بیان شده که سیگنال ساعت در فرکانس ۱۲۵۰ MHz کار می‌کند، در حالیکه در اترنت گیگابیت فرض بر آن است که نرخ تحویل داده‌ها 1Gbps است. آیا این سرعت بالاتر، برای ایجاد حاشیه اطمینان بیشتر بوده است؟ اگر نه، چه نکته‌ای در کار است؟
 ۲۶. اترنت گیگابیت از عهده ارسال یا دریافت چند فریم در هر ثانیه بر می‌آید؟ به دقت بیندیشید و تمام حالات را در نظر بگیرید. راهنمایی: واقعیاتی که در شبکه اترنت گیگابیتی وجود دارد را مد نظر قرار بدهید.
 ۲۷. دو شبکه را نام ببرید که اجازه می‌دهند فریمها بطور پشت سرهم و در یک بار ارسال انتقال یابند. به چه دلیل چنین خصوصیاتی ارزشمند است؟

۲۸. در شکل ۴-۲۷ چهار ایستگاه A، B، C و D نشان داده شده‌اند. به نظر شما کدام یک از ایستگاه‌ها به A نزدیکتر هستند؟
۲۹. فرض کنید که در یک شبکه محلی بی‌سیم 802.11b [پس از در اختیار گرفتن کانال] تعدادی فریم پشت سر هم به طول ۶۴ بایت بر روی کانال با نرخ خطای 10^{-7} منتقل گردد. به طور متوسط چند فریم در هر ثانیه در اثر خطا آسیب خواهد دید؟
۳۰. یک شبکه 802.16 دارای کانالی با عرض باند 20MHz است. در هر ثانیه چند بیت برای یک ایستگاه قابل ارسال خواهد بود؟
۳۱. استاندارد IEEE 802.16 از چهار رده خدمات پشتیبانی می‌کند. کدامیک از این رده‌های خدمات، بهترین انتخاب برای ارسال تصاویر ویدیویی فشرده‌نشده می‌باشد؟
۳۲. دو دلیل بیاورید که چرا برخی شبکه‌ها ممکن است از کدهای تصحیح خطا به جای فرآیند کشف خطا و ارسال مجدد استفاده نمایند.
۳۳. در شکل ۴-۳۵ می‌بینیم که یک ابزار مبتنی بر بلوتوث می‌تواند به طور همزمان درون دو پیکونت قرار داشته باشد. آیا دلیلی وجود دارد که یک ابزار نتواند بطور همزمان نقش گره اصلی (Master) را در هر دو پیکونت ایفا کند؟
۳۴. شکل ۴-۲۵، چندین پروتکل لایه فیزیکی را نشان می‌دهد. کدام یک از آنها به پروتکل لایه فیزیکی در بلوتوث نزدیکتر است؟ کدام یک از آنها بیشترین اختلاف را دارد؟
۳۵. بلوتوث از دو نوع لینک بین گره اصلی (Master) و گره‌های پیرو (Slave) پشتیبانی می‌کند. این دو کدام است و هر کدام برای چه کاری در نظر گرفته شده‌اند؟
۳۶. فریمهای Beacon در گونه‌ای از شبکه 802.11 که مبتنی بر روش «طیف گسترده با پرش فرکانسی» (Frequency Hopping Spread Spectrum) است زمانی به نام Dwell time را مشخص می‌کند. آیا به نظر شما در فریمهای مشابه Beacon در بلوتوث نیز زمانی به نام dwell time وجود دارد؟ پاسخ خود را شرح دهید.
۳۷. به شبکه‌های LAN متصل به هم که در شکل ۴-۴۱ نشان داده شده دقت کنید. فرض کنید ماشینهای میزبان a و b بر روی LAN 1 و c بر روی LAN 2 و d بر روی LAN 8 واقع هستند. در ابتدا جداول Hash پلها خالی است و «درخت پوشای» نشان داده شده در شکل ۴-۴۱-ب به کار گرفته شده است. نشان دهید که جداول پلهای مختلف در اثر هر یک از رخدادهای زیر که به ترتیب حروف الفبا اتفاق می‌افتند چگونه تغییر می‌کند.
- الف) a برای d ارسال می‌کند.
- ب) c برای a فریمی می‌فرستد.
- ج) d برای c فریمی می‌فرستد.
- د) d به LAN 6 نقل مکان می‌کند.
- ه) d برای a فریمی می‌فرستد.
۳۸. یکی از تبعات استفاده از «درخت پوشا» برای هدایت فریمها در یک LAN گسترش یافته آن است که برخی از پلها در فرآیند هدایت فریمها نقشی ایفاء نخواهند کرد. در شکل ۴-۴۴ سه تا از اینگونه پلها را مشخص کنید. آیا دلیلی دارد که چنین پلهایی را در شبکه داشته باشیم، حتی وقتی در هدایت فریمها نقشی ایفاء نمی‌کنند؟
۳۹. تصور کنید که یک سوئیچ دارای چند «کارت خط» (Line Card) است و هر یک از کارتها چهار خط

ورودی دارند. فرض کنید که بطور متناوب فریمهایی که به یکی از خطوط کارت وارد می شوند از خط دیگری بر روی همان کارت خارج می گردند؛ طراح سوئیچ چه راهکاری در مواجهه با این مسئله پیش رو دارد؟

۴۰. یک سوئیچ که برای به کارگیری در اترنت سریع طراحی شده، دارای یک Backplane است که می تواند 10Gbps را منتقل کند. در سنگین ترین بار این سوئیچ قادر به انتقال چند فریم در هر ثانیه است؟

۴۱. به شبکه شکل ۴-۴۹-الف دقت کنید. اگر ماشین L به ناگاه «سفید» شود [یعنی به VLAN سفید بپیوندد] آیا تغییری در برجسبها لازم است؟ اگر این چنین است چگونه؟

۴۲. به طور مختصر تفاوت بین سوئیچهای نوع «ذخیره و هدایت» (Store & Forward) و سوئیچهای Cut Through را توضیح بدهید.

۴۳. سوئیچهای نوع «ذخیره و هدایت» در مقایسه با سوئیچهای Cut Through از دیدگاه آسیب رسیدن به فریمها برتر هستند. توضیح بدهید که این برتری از کجا ناشی می شود؟

۴۴. برای آنکه شبکه های VLAN شروع به کار کنند باید جداول پلها و سوئیچها تنظیم و پیکربندی شوند. اگر در شکل ۴-۴۹-الف به جای استفاده از شبکه مبتنی بر کابل چنداتصال، در VLAN از هاب استفاده شود آیا باز هم به تنظیم این جداول نیاز است؟ آیا هابها نیز نیاز به پیکربندی جدول دارند؟ چرا بله یا چرا نه؟

۴۵. در شکل ۴-۵۰، شبکه موجود و قدیمی سمت راست به یک سوئیچ سازگار با VLAN متصل شده است؛ آیا امکان دارد در اینجا نیز از یک سوئیچ قدیمی استفاده کرد؟ اگر جواب منفی است چرا؟

۴۶. برنامه ای بنویسید تا رفتار پروتکل CSMA/CD را در شبکه اترنت (در شرایطی که پس از ارسال یک فریم، N ایستگاه آماده ارسال هستند) شبیه سازی کند. برنامه شما باید زمانهایی را که هر ایستگاه موفق می شود ارسال فریم خود را آغاز کند گزارش نماید. فرض کنید که به ازای هر برش ۵۱/۲ میکروثانیه ای یک تیک ساعت اتفاق می افتد و کشف تصادم و ارسال سیگنال نویز (دنباله Jam پس از کشف تصادم) جمعاً ۵۱/۲ میکروثانیه طول می کشد. فریمها می توانند بیشترین طول مجاز را داشته باشند.

لایه شبکه

وظیفه لایه شبکه آن است که بسته های داده را به هر طریق از مبدا به مقصد برساند. هر بسته برای رسیدن به مقصد ممکن است از چندین مسیر یاب (Router) در میانه راه گذر کند. این عملکرد به وضوح با عملکرد لایه پیوند داده (که هدف آن انتقال فریمها از انتهای یک سیم به نقطه دیگر آن است) تفاوت دارد. بنابراین لایه شبکه تحتانی ترین لایه ای است که با انتقال «انتها به انتها» (End-To-End) سر و کار دارد.

لایه شبکه برای نیل به اهداف خود، موظف است از توپولوژی «زیر شبکه ارتباطی»^۱ (یا به عبارتی مجموعه تمام مسیر یابها) اطلاع داشته باشد و با استفاده از این دانش مسیری مناسب را برگزیند. همچنین باید مراقب باشد تا از تحمیل بار اضافی بر روی برخی از خطوط ارتباطی و مسیر یابها (در حالی که برخی دیگر آزاد هستند) اجتناب نماید. نهایتاً وقتی که مبدا و مقصد یک بسته در دو شبکه با پروتکل های متفاوت و ناسازگار قرار گرفته اند مشکلات جدیدی بروز می کند که حل آنها بر عهده لایه شبکه گذاشته شده است. در این فصل به مطالعه و تشریح برخی از این موارد خواهیم پرداخت. تمرکز اصلی ما بر اینترنت و لایه شبکه آن یعنی IP است، هر چند شبکه های بی سیم را نیز خاطرنشان خواهیم نمود.

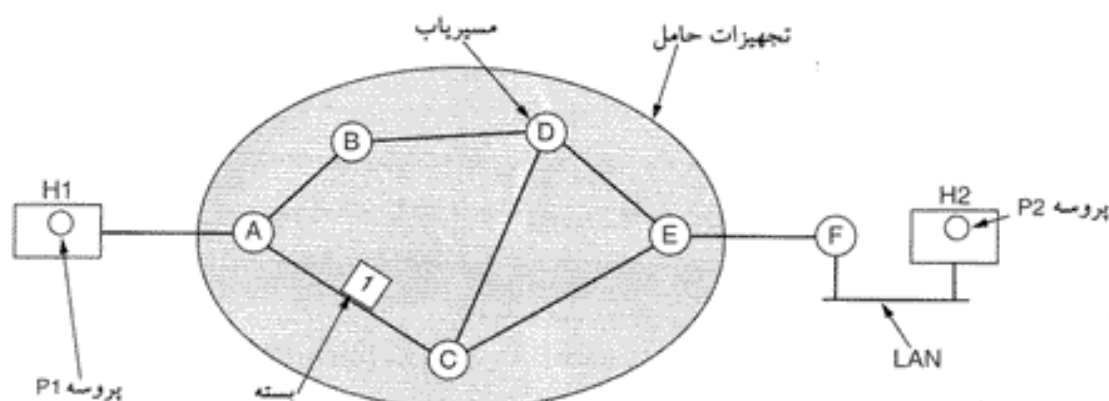
۱-۵ مسائل طراحی لایه شبکه

در بخشهای آتی مقدمه ای خواهیم داشت بر مسائلی که طراحان لایه شبکه با آنها دست به گریبان خواهند بود. این مسائل شامل خدماتی است که به لایه انتقال ارائه می گردد، یا به طراحی داخلی «زیر شبکه» (Subnet) مربوط می شود.

۱-۱-۵ هدایت (سوئیچینگ) بسته به روش «ذخیره و هدایت»

قبل از آنکه به تشریح جزئیات لایه شبکه بپردازیم شاید مروری بر حیطه و زمینه عملکرد پروتکل های لایه شبکه، خالی از لطف نباشد. الگوی کلی چنین محیطی در شکل ۱-۵ دیده می شود. مؤلفه های اصلی این سیستم عبارتند از: تجهیزات حامل (Carrier Equipments) که در این شکل درون بیضی سایه دار قرار گرفته اند و تجهیزات مشتریان (Customer's Equipments) که در خارج از بیضی قرار دارند. ماشین میزبان H1 از طریق یک خط اجاره ای، مستقیماً به یکی از مسیر یابهای حامل یعنی A، متصل شده است. در مقابل H2 بر روی یک شبکه LAN

۱. در این فصل به کرات از واژه «زیر شبکه» به معنای Communication Subnet (مجموعه تمام مسیر یابهای شبکه و خطوط ارتباطی بین آنها) بهره خواهیم گرفت. -م



شکل ۵-۱. محیطی که پروتکل های لایه شبکه در آن عمل می کنند.

قرار گرفته که دارای یک مسیریاب است و تحت فرمان مشتری عمل می کند. مسیریاب F، دارای یک خط اجاره ای با یکی از «تجهیزات حامل» است. ما در این شکل، مسیریاب F را خارج از بیضی در نظر گرفته ایم چرا که عضو بخش حامل [یعنی زیر شبکه اصلی] نیست ولیکن ممکن است از لحاظ ساختار، نرم افزار و پروتکل تفاوتی با مسیریاب های حامل نداشته باشد. اینکه آیا چنین مسیریابی متعلق به زیر شبکه هست یا نه قابل بحث است ولیکن با اهدافی که در این فصل مدنظر داریم مسیریاب های سمت مشتری را نیز به عنوان بخشی از زیر شبکه در نظر می گیریم چرا که آنها نیز همانند مسیریاب های حامل از الگوریتم های مشابهی استفاده می کنند (و اهم کار ما نیز الگوریتم ها هستند).

از این تجهیزات به نحو زیر استفاده می شود: یکی از ماشین های میزبان که بسته ای را برای ارسال آماده دارد آن را برای نزدیکترین مسیریاب می فرستد، خواه این مسیریاب بر روی LAN خودش واقع باشد و خواه از طریق یک خط اجاره ای مستقیماً به زیر شبکه حامل متصل شده باشد. بسته ارسالی در آن مسیریاب موقتاً ذخیره می شود تا بطور کامل دریافت و کد کشف خطای آن، بررسی گردد؛ سپس به مسیریاب بعدی واقع بر مسیر، هدایت می شود و این روند آنقدر تکرار می شود تا بسته به ماشین مقصد رسیده و تحویل داده شود. این مکانیزم همان روش «ذخیره و هدایت» است که در فصول قبلی بدانها پرداختیم.

۲-۱-۵ خدمات ارائه شده برای لایه انتقال

لایه شبکه خدماتی را برای لایه انتقال تدارک می بیند که توسط واسطه میانی لایه شبکه و انتقال^۱ عرضه می شود. سؤال مهم آن است که لایه شبکه چه نوع خدماتی را برای لایه انتقال تدارک می بیند؟ خدمات لایه شبکه با در نظر داشتن اهداف زیر طراحی شده اند:

۱. این خدمات بایستی مستقل از تکنولوژی بکار رفته در مسیریاب باشد.
۲. لایه انتقال باید از درگیری با جزئیاتی مثل تعداد، نوع و توپولوژی مسیریاب های موجود دور نگه داشته شود.
۳. آدرس های شبکه که در اختیار لایه انتقال قرار می گیرند بایستی از ساختار متحدالشکل و استاندارد برخوردار باشند (خواه در شبکه LAN و خواه در شبکه WAN).

با در نظر داشتن این اهداف، طراحان لایه شبکه در تدوین شرح مبسوط خدماتی که باید به لایه انتقال عرضه شود آزادی عمل دارند. این آزادی عمل اغلب به مناقشات دیرینه دو گروه با طرز فکر مخالف دامن می زند. مناقشه

۱. Network Layer/Transport Layer Interface

بر سر آن است که آیا لایه شبکه باید خدمات اتصال‌گرا (Connection Oriented) ارائه کند یا خدمات بدون اتصال (Connectionless).

یکی از طرفین (به رهبری دست‌اندرکاران اینترنت) استدلال می‌کند که وظیفه یک مسیریاب هدایت بسته‌هاست و نه چیز دیگر! از دیدگاه آنها (به پشتوانه سی سال تجربه واقعی با یک شبکه کامپیوتری حقیقی و در حال کار) زیرشبکه ذاتاً غیرقابل اعتماد است و چگونگی طراحی آن اهمیت ندارد. بدین ترتیب ماشینهای میزبان باید این حقیقت را بپذیرند که زیرساخت ارتباطی شبکه آنها غیرقابل اعتماد است و خودشان موظفند عملیات نظارت بر خطاهای احتمالی (شامل کشف و تصحیح) و همچنین کنترل جریان (Flow Control) را بر عهده بگیرند.

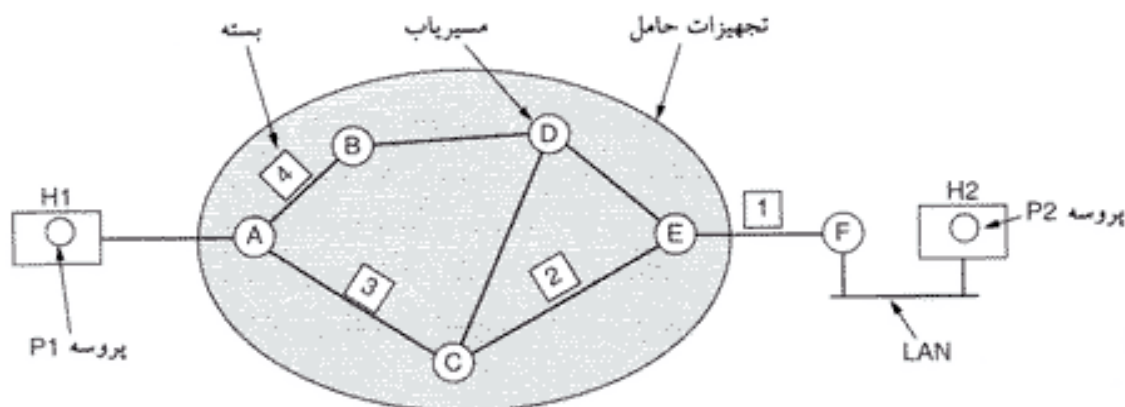
این دیدگاه سریعاً بدانجا منتهی می‌شود که خدمات لایه شبکه باید «بدون اتصال» بوده و فقط عملیات پایه و ابتدایی SEND PACKET و RECEIVE PACKET و تعداد کمی دیگر از همین عملیات را انجام بدهند. خصوصاً نباید عملیات مرتب‌سازی بسته‌ها [از لحاظ ترتیب ارسال] و کنترل جریان انجام شود چرا که ماشینهای میزبان در هر حال چنین کاری را انجام می‌دهند و انجام دوباره یک کار فایده چندانی نخواهد داشت. همچنین هر بسته باید مشخصات کامل آدرس مقصد را با خود داشته باشد چرا که بسته‌های ارسالی باید مستقل از یکدیگر و فارغ از آنکه مسیریاب قبلی کدام بوده، هدایت شوند.

طرف مقابل این مناقشه (به رهبری شرکت‌های مخابراتی) دلیل می‌آورد که زیرشبکه باید خدماتی اتصال‌گرا و قابل اعتماد ارائه کند. اینان نیز مدعیند که سابقه درخشان و صد ساله آنها در سیستمهای جهانی تلفن، دلیل عالی و رهنمای راه آنهاست. از دیدگاه این گروه، کیفیت خدمات (QoS) عمده‌ترین مسئله است و اگر زیرشبکه اتصال‌گرا نباشد دستیابی به کیفیت خدمات (بالاخص برای ترافیک داده‌های بی‌درنگ مثل صدا و تصویر) دشوار خواهد بود. این دو طرف دعوا نمونه‌های خاص خود یعنی اینترنت و ATM را در دست داشتند: اینترنت خدمات لایه شبکه را بدون اتصال عرضه می‌کند در حالی که ATM در لایه شبکه خدماتی اتصال‌گرا ارائه می‌نماید. ولیکن اشاره به این نکته جالب است که هر چه تضمین کیفیت خدمات بیشتر و بیشتر اهمیت پیدا می‌کند، اینترنت نیز رشد می‌نماید. خصوصاً آنکه در راه کسب ویژگیهایی است که عموماً متناسب به خدمات اتصال‌گرا هستند؛ بعداً در این خصوص اشاراتی خواهیم داشت. در خلال مطالعه شبکه‌های VLAN در فصل چهارم نمونه‌ای از این رشد و تکامل را ارائه کردیم.

۳-۱-۵ پیاده‌سازی خدمات بی‌اتصال

پس از نگاهی بر دو رده از خدماتی که لایه شبکه می‌تواند به کاربران عرضه نماید وقت آن فرا رسیده که ببینیم درون این لایه چه می‌گذرد. بسته به نوع خدمات ارائه شده، دو نوع معماری متفاوت ممکن خواهد بود. اگر خدمات بدون اتصال عرضه شود بسته‌ها بطور مجزا و مستقل به درون زیرشبکه تزریق و مستقل از یکدیگر هدایت و مسیریابی می‌شوند و هیچگونه تنظیمات قبلی نیاز نیست. در چنین حالتی عموماً به بسته‌ها «دیتاگرام» (Datagram) و به زیرشبکه نیز «زیرشبکه دیتاگرام» اطلاق می‌شود. اگر خدمات اتصال‌گرا عرضه شود قبل از ارسال هر گونه بسته داده بایستی از قبل یک مسیر بین مسیریاب مبداء و مسیریاب مقصد تنظیم و ایجاد شود. چنین ارتباطی اصطلاحاً VC (مخفف Virtual Circuit) نامیده می‌شود (مترادف با ایجاد یک مدار فیزیکی در شبکه تلفن) و به زیرشبکه نیز، «زیرشبکه مدار مجازی» (Virtual Circuit Subnet) گفته می‌شود. در این بخش ابتدا به زیرشبکه‌های دیتاگرام می‌پردازیم و در بخش بعدی زیرشبکه‌های مدار مجازی را مطالعه خواهیم نمود.

حال بپایانید به بررسی عملکرد زیرشبکه مبتنی بر دیتاگرام بپردازیم. فرض کنید پروسه P1 در شکل ۵-۲ پیامی طولانی برای پروسه P2 دارد. او این پیام را به لایه انتقال سپرده و از او می‌خواهد که پیام را به پروسه P2 بر روی



جدول A

	initially	later
A	-	-
B	B	B
C	C	C
D	B	B
E	C	B
F	C	B

جدول C

A	A
B	A
C	-
D	D
E	E
F	E

جدول E

A	C
B	D
C	C
D	D
E	-
F	F

(خط) Dest. Line (مقصد)

شکل ۵-۲. مسیریابی در یک زیر شبکه دیناگرام.

ماشین میزبان H2 تحویل بدهد. کد نرم افزار لایه انتقال که عموماً بخشی از سیستم عامل به حساب می آید اجرا شده و سرآیند لایه انتقال در ابتدای پیام درج و نتیجه، تحویل لایه شبکه می شود که آن نیز یک پروسه اجرایی (روال اجرایی) در سیستم عامل است.

فرض کنیم که پیام چهار برابر طولانی تر از اندازه مجاز بسته های داده باشد؛ لذا لایه شبکه مجبور است آنها را به چهار بسته ۱ و ۲ و ۳ و ۴ بشکند و هر یک از آنها را بطور مجزا و از طریق یک پروتکل نقطه به نقطه (نظیر PPP)^۱ برای مسیریاب A می فرستد. از اینجا به بعد ادامه کار بر عهده زیر شبکه حامل قرار می گیرد. هر مسیریاب دارای جدولی داخلی است که مشخص می کند برای رسیدن به تمام مسیریابهای مقصد در شبکه، بسته باید بر روی کدام خط خروجی [کدام مسیریاب بعدی] ارسال شود. هر یک از «درایه های» این جدول (Table Entry) شامل یک جفت آیتم اطلاعاتی است که یکی «مقصد» و دیگری خط خروجی برای رسیدن بدان مقصد را تعیین می نماید. برای هدایت بسته فقط می توان از خطوطی که مستقیماً به یک مسیریاب مجاور متصلند بهره گرفت. به عنوان مثال در شکل ۵-۲، مسیریاب A فقط دو خط خروجی (به مسیریابهای B و C) دارد و طبعاً هر یک از بسته های ورودی باید برای یکی از این دو مسیریاب ارسال گردد، حتی اگر مقصد نهایی به مسیریابی کاملاً متفاوت متصل باشد. جدول مسیریابی ابتدایی A با عنوان «Initialy»، در شکل ۵-۲ مشخص شده است.

پس از ورود بسته های ۱ و ۲ و ۳ به مسیریاب A، موقتاً در حافظه ذخیره می شوند (تا کدهای کشف خطای آنها بررسی شود). سپس بر اساس جدول مسیریابی A، این بسته ها به سمت مسیریاب C هدایت می شوند. به همین ترتیب و در ادامه طی مسیر، بسته ۱ از E و نهایتاً F می گذرد. پس از آنکه بسته به F رسید درون فریم لایه پیوند داده

۱. پروتکل PPP یک پروتکل برای خطوط نقطه به نقطه است که در لایه پیوند داده ها عمل می کند یعنی یک فریم را بر روی کانالی واحد و غیر مشترک، از نقطه ای به نقطه دیگر تحویل می دهد. به فصل سوم مراجعه کنید. -م.

جاسازی (کپسوله) شده و از طریق شبکه LAN برای H2 ارسال می‌گردد. بسته‌های ۲ و ۳ نیز به همین طریق مسیریابی و هدایت می‌شوند.

ولیکن (به عنوان مثال) برای بسته ۴ اتفاق متفاوتی می‌افتد. وقتی این بسته به A می‌رسد اگرچه به مقصد F طی مسیر می‌کند ولیکن برخلاف قبل به سوی مسیریاب B ارسال می‌شود. به دلایل متعدد A تصمیم گرفته تا بسته ۴ را از مسیری متفاوت با مسیر سه بسته قبلی، به سمت F بفرستد. شاید این مسیریاب متوجه ازدیاد ترافیک و ازدحام بر روی مسیر ACE شده و جدول مسیریابی خود را به جدولی جدید (که در شکل با عنوان Later مشخص شده) اصلاح و بهنگام‌سازی کرده است. الگوریتمی که مدیریت این جدول را بر عهده دارد و در خصوص مسیریابی تصمیم می‌گیرد اصطلاحاً «الگوریتم مسیریابی» (Routing Algorithm) نامیده می‌شود. الگوریتمهای مسیریابی یکی از اصلیت‌ترین مفاد این فصل هستند.

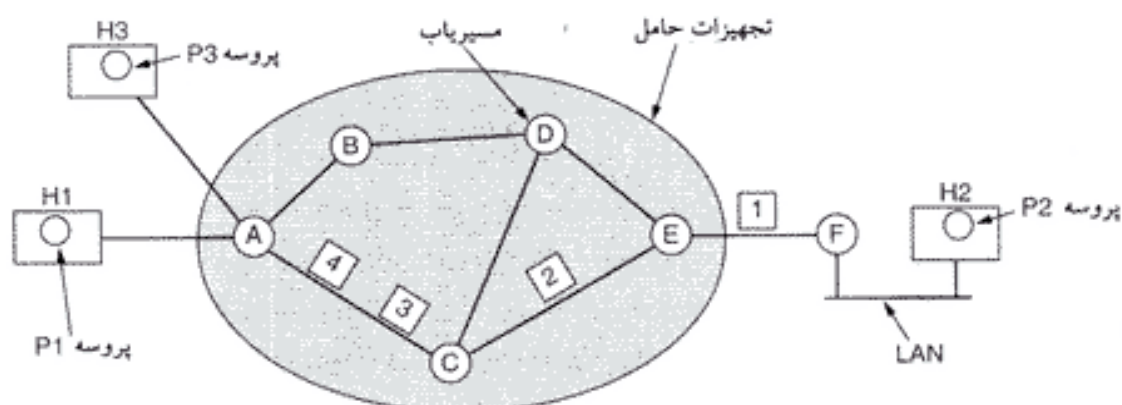
۴-۱-۵ پیاده‌سازی خدمات اتصال‌گرا

رای عرضه خدمات اتصال‌گرا، به زیرشبکه‌های مبتنی بر «مدار مجازی» (Virtual Circuit) نیازمندیم. حال ببینیم که چگونه کار می‌کند: ایده اصلی در مدار مجازی آن است که برخلاف مثال شکل ۵-۲، از انتخاب مسیرهای جدید برای هر بسته اجتناب شود. در عوض وقتی یک «اتصال» ایجاد شد، به عنوان بخشی از عملیات تنظیم اتصال (Connection Setup)، مسیری بین ماشین مبدا و ماشین مقصد انتخاب می‌شود و مشخصات آن در جدول داخلی هر مسیریاب درج می‌گردد. دقیقاً مشابه با سیستم تلفن پس از برقراری اتصال، تمام بسته‌ها از طریق همین مسیر هدایت خواهند شد. پس از آنکه اتصال برقرار شده خاتمه یافت مدار مجازی متناظر نیز پایان یافته و حذف می‌شود. در خدمات اتصال‌گرا هر بسته با خود یک شماره شناسایی حمل می‌کند که مشخص می‌کند به کدام مدار مجازی تعلق دارد.^۱

به عنوان مثال، به وضعیت نشان داده شده در شکل ۵-۳ توجه کنید. در اینجا ماشین میزبان H1 اتصال شماره ۱ را با ماشین میزبان H2 برقرار نموده است. مشخصه این اتصال به عنوان اولین درایه (Entry) در جدول مسیریابی درج شده است. اولین سطر از جدول A بیان می‌کند که اگر بسته‌ای با شماره شناسایی ۱ از H1 دریافت شود باید با همین شماره شناسایی برای مسیریاب C ارسال گردد. به روش مشابه مسیریاب C مطابق با اولین درایه خود، بسته را با شماره شناسایی ۱ به سوی مسیریاب E هدایت می‌نماید.

حال بررسی کنیم که اگر H3 نیز بخواهد اتصالی را با H2 برقرار کند چه اتفاقی می‌افتد. او نیز شماره شناسایی اتصال را ۱ در نظر می‌گیرد (چرا که این اتصال، اولین اتصالی است که او برقرار کرده است) و سپس از زیر شبکه می‌خواهد که برایش مداری مجازی ایجاد کند. انجام این تقاضا منجر به اضافه شدن سطر دوم به جدول است. دقت کنید که در اینجا یک تناقض وجود دارد و آن هم اینکه اگرچه A می‌تواند بسته‌هایی که با شماره شناسایی ۱ از H1 می‌رسند را از بسته‌هایی با همین شماره که از H3 می‌رسند تمیز بدهد ولی مسیریاب C قادر به چنین کاری نیست [چرا که هر دو با یک شماره و از A می‌رسند]. به همین دلیل A برای ترافیک خروجی که در قالب اتصال ۲ خارج می‌شوند شماره شناسایی متفاوتی را در نظر می‌گیرد. برای اجتناب از تناقضاتی از این دست، مسیریابها باید بتوانند شماره شناسایی اتصال هر بسته خروجی را تغییر بدهند.

۱. اتصال را در ذهن خود با یک تماس تلفنی قیاس کنید. ابتدا شماره می‌گیرید، تماس شما برقرار می‌شود، تا زمان دلخواه گفتگو می‌کنید و نهایتاً تماس را قطع می‌نمایید. اتصال در زیر شبکه مدار مجازی بمعنای هماهنگی قبلی بین مبدا، مقصد و مسیریابهای میانی تلقی می‌شود. به مسیری که با هماهنگی قبلی ایجاد می‌شود یک «مدار مجازی» می‌گویند و دارای یک شماره شناسایی است. به این شماره شناسایی اصطلاحاً «شناسه مدار مجازی» یا «شناسه اتصال» (Connection ID) اطلاق می‌شود. -م



جدول A				جدول C				جدول E			
H1	1	C	1	A	1	E	1	C	1	F	1
H3	1	C	2	A	2	E	2	C	2	F	2
In		Out									

شکل ۵-۳. مسیریابی در یک زیرشبکه مدار مجازی.

۵-۱-۵ مقایسه زیرشبکه های مدار مجازی و دیتاگرام

هر یک از روشهای مدار مجازی و دیتاگرام حامیان و منتقدین خود را دارند. حال تلاش می کنیم مباحثات آنها را به طور خلاصه ارائه نماییم. محورهای اصلی این مباحثات در شکل ۵-۴ فهرست شده اند؛ هر چند ممکن است سفسطه کنندگان بتوانند برای هر یک از موارد این جدول، مثال نقضی پیدا کنند.

در هر زیرشبکه می توان اصولی را برای قیاس زیرشبکه های مدار مجازی و دیتاگرام مطرح کرد. یکی از آنها مسئله میزان حافظه مسیریاب در مقایسه با پهنای باند تلفاتی است. روش مدار مجازی اجازه می دهد که بسته ها به جای همراه داشتن آدرسهای کامل فقط شماره مدار مجازی را با خود داشته باشند. هرگاه بسته ها نسبتاً کوتاه باشند وجود آدرس کامل در هر بسته ممکن است حجم سربار زیادی را تحمیل کرده و بدین ترتیب بخشی از پهنای باند مفید هدر می رود. در عوض، هزینه ای که برای یکارگیری مدار مجازی پرداخت می شود فضای حافظه ای است که جدول مشخصات مدارات مجازی در درون مسیریاب، به خود اختصاص می دهد. بسته به قیمت پهنای باند کانالهای مخابراتی در مقایسه با حافظه مسیریاب یکی از این دو ارزانتر تمام می شود.

یکی دیگر از مسائل قیاسی، «زمان تنظیم و ایجاد مدار مجازی» در مقایسه با زمان جستجو و تحلیل آدرسها است.^۱ استفاده از مدار مجازی منوط به طی مراحل تنظیم مسیر است که فرآیندی زمان بر بوده و منابع زیرشبکه را مصرف می کند. با این وجود در زیرشبکه مبتنی بر مدار مجازی، تشخیص آنکه چه کاری با یک بسته داده باید انجام شود ساده است: مسیریاب از شماره شناسایی مدار مجازی به عنوان اندیس جدول مسیریابی استفاده می کند تا مسیری که بسته باید طی کند مشخص شود. در زیرشبکه دیتاگرام برای جستجو و پیدا کردن درایه متناظر با مقصد، به روالی پیچیده تر نیاز است.

مورد دیگر، فضای مورد نیاز جدول مسیریابی در حافظه مسیریاب است. یک زیرشبکه دیتاگرام نیازمند آن است که به ازای هر مقصد در شبکه، یک درایه (Entry) در جدول مسیریابی خود داشته باشد، در حالی که در

۱. Setup Time versus Address Parsing Time

مورد	زیر شبکه دیتاگرام	زیر شبکه مدار مجازی
تنظیم مدار (Circuit Setup)	نیازی نیست	نیاز است.
آدرس دهی	هر بسته آدرس دقیق و کامل مبدا و مقصد را با خود حمل می کند.	هر بسته فقط یک شماره کوتاه مدار مجازی (VC) را با خود دارد.
اطلاعات وضعیت	مسیریاب نیازی به نگهداری اطلاعاتی در خصوص وضعیت هر اتصال ندارد.	به ازای هر مدار مجازی تمام مسیریابها باید اطلاعاتی در خصوص وضعیت آن نگاه دارند.
مسیریابی	هر بسته بطور مستقل مسیریابی می شود.	مسیر فقط یکبار و آنهم در هنگام تنظیم مدار مجازی انتخاب می شود و تمام بسته ها از همان مسیر حرکت می کنند.
تأثیر خرابی مسیریاب	بی تأثیر، مگر در مورد بسته هایی که در حین خرابی از بین رفته اند.	تمام مدارات مجازی که از مسیریاب خراب شده می گذشته اند قطع می شوند.
تضمین کیفیت خدمات	دشوار	اگر برای هر مدار مجازی منابع لازم از قبل تخصیص یابد، بسیار آسان است.
کنترل ازدحام	دشوار	اگر برای هر مدار مجازی منابع لازم از قبل تخصیص یابد، بسیار آسان است.

شکل ۴-۵. مقایسه زیر شبکه های دیتاگرام و مدار مجازی.

زیر شبکه مدار مجازی، فقط به ازای هر اتصال به یک درایه نیاز است. ولیکن این حسن تا حدودی گمراه کننده و تو خالی است چرا که بسته های تنظیم اتصال [که برای اولین بار ارسال می شوند] بهر نحو باید همانند روش دیتاگرام یکبار مسیریابی شوند و آنها نیز دارای آدرسهای کامل مقصد هستند. [یعنی در زیر شبکه مدار مجازی هم به جدول مسیریابی و هم به جدول مدارات مجازی نیاز است. -م]

مدارهای مجازی دارای محاسنی در خصوص تضمین «کیفیت خدمات» (QoS) هستند و از بروز ازدحام (congestion) در زیر شبکه جلوگیری می کنند چرا که «منابع» (شامل بافر، پهنای باند و سیکل های CPU) می تواند پیشاپیش و در هنگام ایجاد اتصال، رزرو شود. بعداً به محض دریافت بسته ها، پهنای باند مورد نیاز و ظرفیت لازم در مسیریاب، مهیا و آماده است. در زیر شبکه های دیتاگرام، اجتناب از ازدحام دشوارتر است.

در «سیستمهای پردازش تراکنشی»^۱، (همانند وقتی که خرید با کارتهای اعتباری بررسی می شوند) مسئله تنظیم یک مدار مجازی و سپس حذف آن ممکن است کاربرد این روش را از اعتبار ساقط کند. [به عبارت بهتر در سیستمهایی که تعامل و مبادله داده با ماشینها، کوتاه و سریع است ایجاد یک مدار مجازی و ختم آن سر بار زیادی به شبکه تحمیل می نماید. -م] اگر بخش اعظم ترافیک از این نوع باشد استفاده از مدار مجازی درون یک زیر شبکه چندان مناسب نیست. در طرف مقابل هرگاه حجم داده های ارسالی انبوه باشد، مدارهای مجازی دانم که به صورت دستی تنظیم می شوند و برای ماهها یا سالها تغییر نمی کنند می تواند سودمندتر باشد.

مدارهای مجازی مشکل «آسیب پذیری» دارند. اگر یک مسیریاب از کار بیفتد و حافظه خود را از دست بدهد (حتی در صورتی که چند ثانیه بعد به زیر شبکه برگردد)، قطعاً تمام مدارهای مجازی که از آن عبور می کنند، از دست خواهند رفت. در مقابل اگر یک مسیریاب مبتنی بر دیتاگرام از کار بیفتد تنها کاربرانی آسیب می بینند که

۱. Transaction Processing System

بسته های آنها هنگام بروز مشکل، درون حافظه مسیریاب در صف منتظر بوده و بر حسب آنکه آیا دریافت بسته های داده قبلاً اعلام شده باشد یا نه، حتی ممکن است فقط بخشی از آنها با مشکل مواجه شوند. از دست رفتن یک خط مخابراتی برای مدارات مجازی که از آن خط استفاده کرده اند بحران زاست در حالی که اگر از روش دیتاگرام استفاده شده باشد این اشکال براحتی جبران خواهند شد. در روش دیتاگرام مسیریابها مجازند ترافیک را بر روی زیر شبکه توزیع و موازنه کنند لذا ممکن است در حین ارسال یک دنباله طولانی از بسته ها مسیرها عوض شوند.

۲-۵ الگوریتمهای مسیریابی

وظیفه اصلی لایه شبکه، مسیریابی و هدایت بسته های داده از ماشین مبدا به ماشین مقصد است. در اکثر زیر شبکه ها، بسته ها برای طی مسیر خود بایستی چندین «گام» (HOP) راه پیمایند.^۱ الگوریتمی که این مسیرها را انتخاب می کنند و همچنین ساختمان داده مورد استفاده، یکی از زمینه های مهم در طراحی لایه شبکه محسوب می شود.

«الگوریتمهای مسیریابی» آن بخش از نرم افزار لایه شبکه اند که مسئولیت دارند در خصوص خط خروجی که یک بسته ورودی باید بر روی آن ارسال شود، تصمیم گیری کنند. اگر در داخل زیر شبکه از روش دیتاگرام استفاده شده باشد این تصمیم گیری باید به ازای هر بسته دریافتی از نو تکرار شود چرا که در هر لحظه ممکن است بهترین مسیر تغییر نماید. اگر زیر شبکه از روش مدار مجازی بهره گرفته باشد، تصمیم گیری در خصوص مسیرها فقط یکبار و آنهم در هنگام تنظیم و ایجاد هر مدار مجازی جدید صورت می گیرد و طبعاً بسته های داده، همگی از مسیری که قبلاً ایجاد شده هدایت می شوند. این حالت گاهی روش «مسیریابی مبتنی بر نشست» (Session Routing) نامیده می شود چرا که مسیر ایجاد شده در خلال نشست کاربر برقرار خواهد ماند؛ (مثلاً در حین نشست از راه دور یا ترمینال Login session). یا نشست انتقال فایل).

گاهی تمایز قائل شدن بین فرآیند «مسیریابی» (که به تصمیم گیری در خصوص مسیرهای بهینه اطلاق می شود) و فرآیند «هدایت» (Forwarding) (آنچه که با ورود بسته اتفاق می افتد) مفید است: می توانید بدینگونه بیندیشید که هر مسیریاب دارای دو پروسه در درون خود است: یکی از آنها بسته ها را به محض ورود پردازش کرده و از طریق جدول مسیریابی، یک خط خروجی مناسب برای آنها انتخاب می نماید. این پروسه همان «عمل هدایت» است. پروسه دیگر موظف به پر کردن و بهنگام سازی محتویات جدول مسیریابی است. این همان جایی که «الگوریتمهای مسیریابی» (Routing Algorithm) به میدان می آیند.

فارغ از آنکه آیا برای هر بسته، مسیرها بطور مستقل و جداگانه انتخاب می شوند یا آنکه فقط یکبار و آن هم در حین ایجاد یک اتصال [مدار مجازی] تعیین می گردد، از یک الگوریتم مسیریابی ویژگیهای خاصی انتظار می رود:

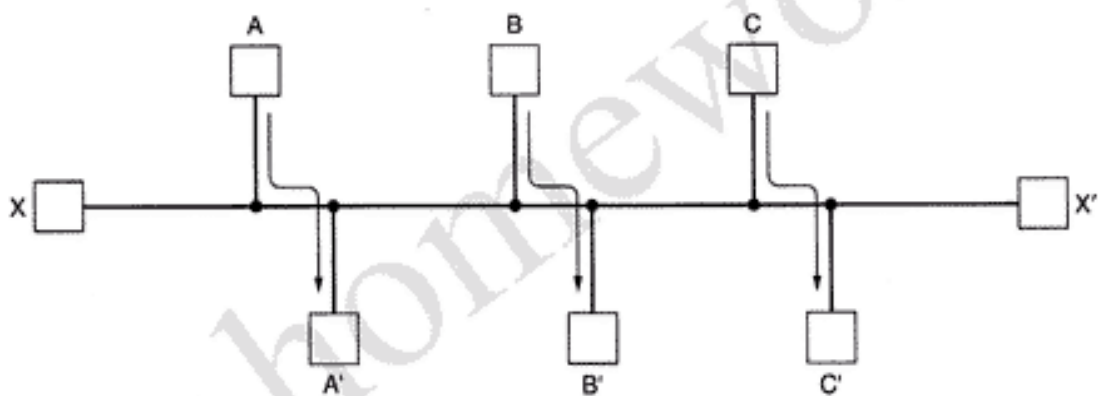
- (۱) صحت عملکرد (correctness) (۲) سادگی (simplicity) (۳) قابلیت تحمل (Robustness) (۴) پایداری (stability) (۵) مساوات (Fairness) (۶) بهینگی (Optimality).

ویژگی «صحت» و «سادگی» نیازی به توضیح ندارد در حالی که نیاز به «قابلیت تحمل» در بدو امر چندان روشن به نظر نمی رسد. با ورود یک شبکه عظیم به صحنه عمل، انتظار آنست که برای سالها بدون بروز هیچگونه خرابی سیستم در سطح وسیع، به کار خود ادامه بدهد. در خلال این دوره زمانی ممکن است انواع سخت افزارها و نرم افزارها از کار بیفتند. ماشینهای میزبان، مسیریابها و خطوط ارتباطی به کرات از کار می افتند و توپولوژی شبکه

۱. به گذر بسته از یک مسیریاب، گام یا Hop گفته می شود

چندین بار تغییر می‌کند. الگوریتم مسیریابی باید بتواند هر گونه تغییر در توپولوژی و ترافیک را بدون ایجاد وقفه در کار ماشینهای میزبان یا نیاز به راه‌اندازی مجدد شبکه (در صورت از کار افتادن برخی از مسیرها)، اصلاح و مدیریت نماید.

یکی دیگر از اهداف و ویژگیهای الگوریتم مسیریابی «پایداری» است. برخی از الگوریتمهای مسیریابی هرگز به یک عملکرد ثابت و پایدار همگرا نمی‌شوند بدون آنکه مدت زمان در حال کار و اجرا بودن آنها اهمیت و تأثیری داشته باشد. ویژگیهای «مساوات» و «بهینگی» ممکن است واضح به نظر برسند و مطمئناً هیچ عقل سلیمی با آن مخالف نیست ولی گاهی در مرحله عمل با یکدیگر در تناقض و تضاد هستند. به عنوان یک مثال از این تناقض، به شکل ۵-۵ نگاه کنید. فرض نمائید که ترافیک بین A و A'، بین B و B' و بین C و C' بقدری است که لینک ارتباط افقی اشباع می‌شود. برای آنکه جریان کلی اطلاعات به حداکثر برسد باید ترافیک بین X و X' قطع گردد؛ [تا بهینگی رعایت شده باشد]. متأسفانه X و X' ممکن است متوجه چنین موضوعی نباشند ولیکن باید بین «کارایی» و «مساوات» حالتی بینابین و متعادل مدنظر قرار بگیرد. اگر ترافیک بین X و X' قطع شود بهینگی بدست می‌آید ولی در عوض مساوات رعایت نخواهد شد!



شکل ۵-۵. تضاد بین «بهینگی» و «مساوات».

قبل از آنکه بتوان بین «مساوات» و «بهینگی» تعادل ایجاد کرد باید ابتدا تصمیم بگیریم که در پی بهینه کردن چه چیزی هستیم! به حداقل رساندن متوسط تأخیر بسته‌ها نامزد واضحی برای بهینه شدن است ولی به حداکثر رساندن بازده مفید شبکه (Throughput) نیز نامزد دیگری است. به علاوه این دو هدف با یکدیگر در تناقض هستند چرا که مثلاً هرگاه یک سیستم صف‌بندی [همانند صفی که در هر مسیر یاب ایجاد می‌شود] در ظرفیت حداکثر خود کار کند تأخیری طولانی را تحمیل خواهد کرد.^۱ به عنوان یک حالت بینابین، در برخی از شبکه‌ها سعی شده تا تعداد «گامهای مسیر» (که یک بسته باید طی کند) به حداقل برسد زیرا کاهش تعداد گامها باعث کاهش تأخیر و پهنای باند مصرفی خواهد شد که نهایتاً به بهبود بازده مفید شبکه می‌انجامد.

الگوریتمهای مسیریابی را می‌توان به دو رده کلی تقسیم‌بندی کرد: «غیروفاقی» (nonadaptive) و «وفاقی» (adaptive). روشهای غیروفاقی تصمیم‌گیری خود در خصوص مسیریابی را بر مبنای اندازه‌گیری و تخمین هوشمند ترافیک و توپولوژی فعلی شبکه، قرار نداده‌اند. در عوض برای انتخاب مسیری بین I و J (برای هر I و J مفروض)، یک مسیر از قبل و به صورت offline محاسبه شده و در هنگام راه‌اندازی شبکه درون مسیریابها بارگذاری می‌شود. به این روال اغلب «مسیریابی ایستا» (Static Routing) گفته می‌شود.

۱. در صف نگاه داشتن بسته‌ها می‌تواند به بازده مفید شبکه بیفزاید ولی تأخیر را نیز افزایش خواهد داد. رجوع کنید به نظریه صف - م

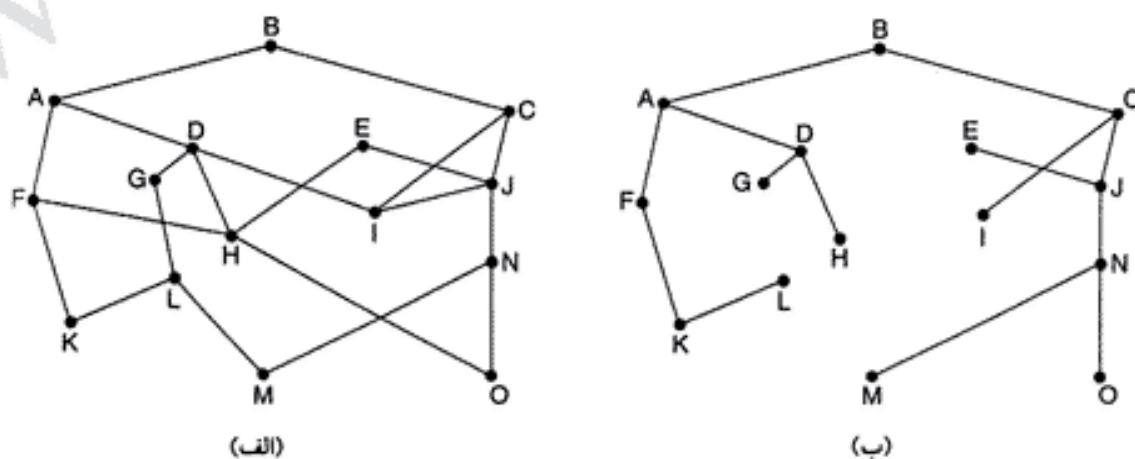
در مقابل، «الگوریتمهای افقی» تصمیم گیری در خصوص مسیریابی را برحسب تغییرات توپولوژی و عموماً ترافیک لحظه ای، بصورت هوشمند انجام می دهند. الگوریتمهای مسیریابی افقی در نحوه گردآوری «اطلاعات مسیر» با یکدیگر متفاوتند (مثلاً اینکه آیا این اطلاعات به صورت محلی گردآوری می شود یا از مسیریابها مجاور بدست می آید یا از همه مسیریابها می رسد). الگوریتمهای افقی همچنین در پارامترهایی نظیر: (۱) زمان بهنگام سازی اطلاعات مسیر (مثلاً بطور متناوب هر ΔT ثانیه بهنگام شوند یا فقط زمانی که توپولوژی عوض شود) (۲) معیاری که برای بهینه سازی مورد استفاده قرار می گیرد (مثل فاصله، تعداد گام (HOP) یا زمان تخمینی انتقال) با یکدیگر متفاوت هستند. در بخش بعدی گونه های مختلفی از الگوریتمهای مسیریابی اعم از پویا و ایستا را تشریح خواهیم کرد.

۱-۲-۵ اصل بهیگی

قبل از آنکه بر روی یک الگوریتم خاص متمرکز شویم، شاید پرداختن به این نکته مفید باشد که هر کسی می تواند بدون توجه به توپولوژی یا ترافیک شبکه یک ارزیابی و تعبیر کلی در خصوص مسیرهای بهینه ارائه بدهد. این تعبیر به نام «اصل بهیگی» مشهور است و بیان می کند که اگر مسیریاب J بر روی مسیر بهینه بین مسیریاب I تا مسیریاب K واقع شده باشد بنابراین مسیر بهینه از J تا K نیز بر روی همان مسیر خواهد بود. برای بررسی این موضوع، آن قسمت از مسیر که بین I تا J قرار دارد را r_1 و مابقی مسیر را r_2 بنامید. اگر مسیری بهتر از r_2 بین J تا K وجود داشته باشد می توان آن را به r_1 افزود تا مسیر بهتری از I تا K ایجاد شود؛ این موضوع با اصل قضیه که بیان می کرد مسیر $r_1 r_2$ بهینه است تناقض دارد.

به عنوان یک نتیجه مستقیم از اصل بهیگی می توان اثبات کرد که مجموعه مسیرهای بهینه بین تمام گره های مبدا و یک گره مقصد خاص، درختی را تشکیل می دهد که ریشه آن بر روی گره مقصد قرار دارد. به چنین درختی اصطلاحاً Sink Tree گفته می شود که شمای آن در شکل ۵-۶ به تصویر کشیده شده است. در این شکل معیار فاصله «تعداد گام» در نظر گرفته شده است. دقت کنید که درخت Sink Tree الزاماً یکتا و منحصر به فرد نیست و ممکن است درختی متفاوت با طول مسیرهای مشابه پیدا شود. هدف اصلی الگوریتمهای مسیریابی آن است که برای تمام مسیریابها درخت Sink Tree محاسبه شده و مورد استفاده قرار بگیرد.

بدیهی است که Sink Tree، حداقل یک درخت است، بنابراین بهیچوجه دارای حلقه نیست و بدین ترتیب هر بسته پس از طی چند گام محدود و معین، تحویل مقصد خواهد شد. ولیکن در دنیای عمل همیشه کار بدین سادگی



شکل ۵-۶. (الف) یک زیر شبکه (ب) درخت Sink Tree برای مسیریاب B.

نخواهد بود. لینکها و مسیر یابها می توانند به ناگاه از کار بیفتند و سپس مجدداً عملیات طبیعی خود را از سر بگیرند. لذا مسیر یابهای مختلف ممکن است توپولوژی شبکه را به گونه متفاوتی ببینند.^۱ همچنین ممکن است بدین نکته ظریف بیندیشیم که آیا هر مسیر یاب مجبور است مستقلاً اطلاعات لازم در خصوص توپولوژی شبکه را جمع آوری کرده و Sink Tree مربوطه را خودش محاسبه نماید یا آنکه این اطلاعات به روشهای دیگری بدست می آیند. در بخشهای آتی مختصراً بدین موضوعات خواهیم پرداخت. بهر حال، «اصل بهینگی» و «Sink Tree» روشی برای محک زدن و ارزیابی الگوریتمهای مسیر یابی دیگر هستند.

۲-۲-۵ مسیر یابی مبتنی بر کوتاهترین مسیر

اجازه بدهید مطالعات خود پیرامون الگوریتمهای مسیر یابی را با پرداختن به روشی آغاز کنیم که بطور گسترده ای از آن استفاده می شود چرا که ساده و فهم جزئیات آن راحت است. ایده اصلی آن است که گراف زیر شبکه را به گونه ای تشکیل بدهیم که در آن هر گره از گراف یک مسیر یاب را مشخص می کند و هر یک از «کمانها» (Arcs) مشخص کننده یک خط ارتباطی (که اغلب لینک نامیده می شوند) هستند. برای انتخاب بهترین مسیر بین یک زوج مسیر یاب مفروض، یک الگوریتم خاص، «کوتاهترین مسیر» بین آن دو را در این گراف پیدا می کند.

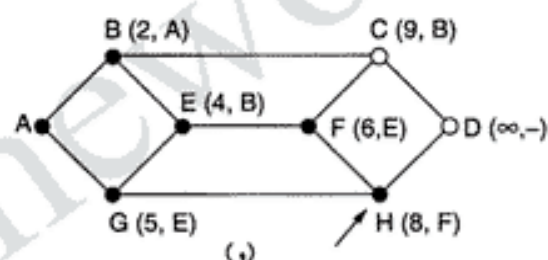
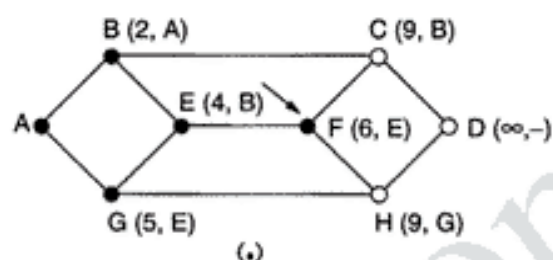
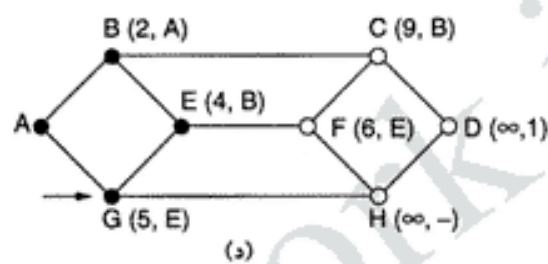
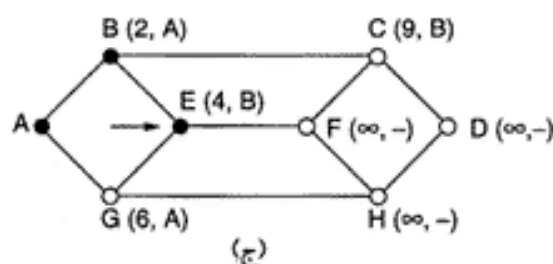
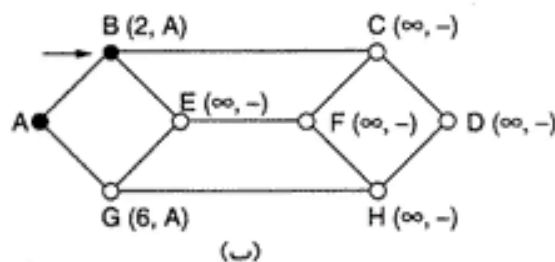
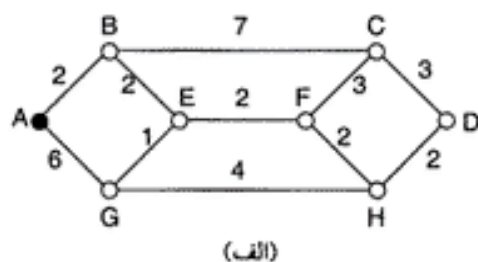
مفهوم «کوتاهترین مسیر» نیاز به اندکی توضیح دارد: یکی از روشهای محاسبه طول مسیر، شمارش تعداد «گام» است. با استناد به این معیار، مسیرهای ABC و ABE در شکل ۷-۵ طول معادلی دارند. یکی دیگر از معیارها، فاصله جغرافیایی گره ها از یکدیگر بر حسب کیلومتر است که در این حالت ABC به وضوح از ABE طولانی تر می باشد (با فرض آنکه شکل به گونه ای ترسیم شده که مقیاسی از نقشه واقعی را ارائه می دهد).

با این وجود، به غیر از معیارهای تعداد گام یا فاصله فیزیکی، معیارهای دیگری نیز وجود دارند. به عنوان مثال هر کمان از گراف می تواند بر چسبی داشته باشد که مقدار متوسط تأخیر صف [صف درون حافظه مسیر یاب] و تأخیر انتقال را تعیین کند؛ این بر چسبها بطور متناوب و با استفاده از برخی بسته های استاندارد محاسبه می شوند. در چنین گراف بر چسب داری، کوتاهترین مسیر همانا سریعترین مسیر است (یعنی مسیر با حداقل تأخیر)، نه مسیری که کمترین تعداد کمان یا کوتاهترین فاصله جغرافیایی را دارد.

در حالت کلی، بر چسب هر کمان می تواند بر حسب تابعی از پارامترهای «فاصله»، «پهنای باند»، «میانگین ترافیک»، «هزینه ارتباط»، «طول متوسط صف»، «تأخیر اندازه گیری شده» یا عوامل دیگر، محاسبه شود. با تغییر در «تابع وزن دهی» (Weighting Function)، این الگوریتم می تواند «کوتاهترین مسیر» را بر حسب یکی از این معیارها یا ترکیبی از آنها بدست بیاورد.

تاکنون الگوریتمهای متعددی برای محاسبه کوتاهترین مسیر بین دو گره از گراف، معرفی شده اند. یکی از آنها توسط «دایکسترا» (Dijkstra, 1959) ارائه شد. در الگوریتم دایکسترا، هر گره دارای بر چسبی است که فاصله آن گره را تا یک گره مبدا بر حسب بهترین مسیر، مشخص می کند. (در شکل ۷-۵ این بر چسبها درون پرانتز درج شده اند). در ابتدا هیچ مسیری مشخص نیست فلذا تمام گره ها در بر چسب خود علامت بی نهایت دارند. در حین اجرای الگوریتم و پیدا شدن مسیرها این بر چسبها تغییر می کنند تا مسیرهای بهتر را مشخص نمایند. هر بر چسب می تواند یکی از دو حالت «موقت» (Tentative) یا «دائم» (Permanent) داشته باشد. در ابتدای کار تمام بر چسبها در حالت موقتی علامت خورده اند. پس از آن که مشخص شد یک بر چسب کوتاهترین مسیر ممکن را از مبدا تا آن گره مشخص کرده، حالت آن گره به حالت «دائم» تغییر کرده و از آن به بعد هیچگاه عوض نخواهد شد.

۱. اگر هر یک از مسیر یابها، مسیرهای بهینه را متفاوت از دیگری تشخیص بدهند و بر اساس اطلاعات ناقص خود مسیر انتخاب کنند، گاهی حلقه بی نهایت ایجاد می شود. -م



شکل ۵-۷. پنج مرحله نخست از روال محاسبه کوتاهترین مسیر از A به D فلشها «گره کار» را مشخص می کنند.

برای آن که بینیم الگوریتم برچسب گذاری چگونه کار می کند، گراف «بدون جهت» و «وزن دار» شکل ۷-۵ الف را در نظر بگیرید که در آن وزنهای فرضاً فاصله را مشخص کرده است. می خواهیم کوتاهترین مسیر از A به D را پیدا کنیم. کار را با تغییر علامت گره A به حالت «ثابت» (که به صورت یک دایره توپر مشخص شده) شروع می کنیم. سپس به نوبت هر یک از گره های مجاور گره A (که گره کار نام دارد) را آزمایش می کنیم و برچسبشان را برحسب فاصله آنها تا A، تغییر می دهیم. هرگاه برچسب یک گره را تغییر دادیم، در آن برچسب، نام گره ای که آزمایش بر مبنای آن صورت گرفته درج می شود تا بعداً بتوان مسیر نهایی را بدست آورد. [یعنی مثلاً وقتی گره های همسایه A یعنی B و G را بررسی می نماییم و فاصله آنها تا A را در برچسبشان تغییر می دهیم، باید نام گره کار قبلی یعنی A در برچسب آنها درج شود. -م] پس از آزمایش گره های مجاور A، تمام گره هایی که به صورت موقتی علامت خورده اند [یعنی دایره های تو خالی] را در «کل گراف» بررسی کرده و گره ای با کوچکترین برچسب را انتخاب و آن را به صورت «دائم» علامت گذاری می کنیم. (شکل ۷-۵ ب) این گره به عنوان «گره کار» جدید انتخاب می شود.

حال از گره B شروع کرده و تمام گره های مجاور آن را آزمایش می نماییم. اگر مجموع برچسب B و فاصله B تا گره مورد نظر از آنچه که درون برچسب B نوشته شده کمتر باشد برچسب آن گره تغییر می کند چراکه مسیری کوتاه تر را پیدا کرده ایم. [به عبارت دیگر وقتی گره های همسایه B را بررسی می نماییم - یعنی E و C را - مجموع فاصله C تا B یعنی ۷ را با فاصله B تا A یعنی ۲ جمع می کنیم و چون ۹ از مقدار بی نهایت در برچسب C کمتر است برچسب آن به صورت (9,B) تغییر می کند یعنی تا رسیدن به گره A فاصله ۹ و گره قبلی B است. -م]

پس از آن که گره‌های مجاور «گره کار» بررسی شدند و در صورت لزوم برچسبهای موقت تغییر نمودند، برای پیدا کردن گره‌ای با علامت موقت و کمترین مقدار فاصله، کل گراف را جستجو می‌نماییم. این گره به حالت «دائم» تغییر وضعیت داده و گره کار جدید در مرحله بعد خواهد شد. شکل ۵-۷ پنج مرحله اول این الگوریتم را نشان می‌دهد.

برای آنکه ببینیم چرا این الگوریتم بدرستی کار می‌کند به شکل ۵-۷ رجوع کنید. در این مرحله، حالت گره E را به حالت «دائم» تغییر داده‌ایم. فرض کنید که مسیری کوتاه‌تر به غیر از ABE مثل XYZE وجود داشته باشد. دو امکان وجود دارد: یا گره Z قبلاً به حالت «دائم» علامت خورده است یا حالت «موقت» دارد. اگر حالت دائم داشته باشد بنابراین نتیجه می‌گیریم که E قبلاً بررسی شده است (در همان مرحله‌ای که گره Z در آن مرحله علامت دائم خورده است) لذا مسیر XYZE از چشم ما دور نمانده و قاعدتاً نمی‌توانسته کوتاه‌ترین مسیر باشد. اکنون حالتی را در نظر بگیرید که در آن Z هنوز برچسب موقت دارد. اگر برچسب Z بیشتر یا مساوی آنچه که در برچسب E درج شده، باشد که در این حالت مسیر XYZE نمی‌تواند از ABE کوتاه‌تر باشد؛ یا آنکه برچسب Z کوچکتر از برچسب E است که در این حالت Z بجای E علامت دائم می‌خورد و E باید قبل از Z بررسی شود [یعنی از طریق E باید Z بررسی شود و مسیر AXYEZ خواهد بود. -م]

این الگوریتم در شکل ۵-۸ ارائه شده است. متغیرهای سراسری n و dist توصیف‌کننده گراف هستند و قبل از فراخوانی تابع shortest_path مقداردهی می‌شوند. تنها تفاوت بین این برنامه و الگوریتمی که در بالا تشریح شد آن است که در برنامه شکل ۵-۸ محاسبه کوتاه‌ترین مسیر از آخر (یعنی از گره پایانی) شروع شده است. از آنجایی که کوتاه‌ترین مسیر از t به s در یک گراف بی‌جهت هیچ تفاوتی با کوتاه‌ترین مسیر از s به t ندارد لذا آنکه از کدام گره شروع کنیم اهمیتی ندارد. (مگر آن که کوتاه‌ترین مسیرها بیش از یکی باشند و جستجوی معکوس مسیر دیگری را نتیجه بدهد.) دلیل آنکه جستجوی کوتاه‌ترین مسیر برعکس انجام می‌گیرد آن است که برچسب هر گره با گره قبلی خود (نه گره بعدی) تغییر می‌کند و وقتی مسیر نهایی در متغیر خروجی یعنی path کپی می‌شود، مسیر معکوس خواهد شد [چون عمل کپی از آخر به اول انجام می‌گیرد]. چون از گره آخر به اول شروع کردیم و نهایتاً مسیر بطور معکوس کپی شد این دو، اثر هم را خنثی کرده و جواب نهایی بطور صحیح تولید خواهد شد. [جواب نهایی در متغیر path، بهترین مسیر از گره مبدا به گره مقصد خواهد بود.]

۳-۲-۵ الگوریتم سیل آسا (Flooding)

یکی از الگوریتمهای ایستا در ارسال بسته‌ها، «الگوریتم سیل آسا» است که براساس آن هر بسته ورودی به یک مسیریاب، بر روی تمام خطوط خروجی (به استثنای خطی که بسته از طریق آن دریافت شده) ارسال می‌شود. این فرآیند سیل آسا، به وضوح منجر به تولید تعداد بسیار زیادی بسته‌های تکراری (در حقیقت بی‌نهایت بسته) خواهد شد مگر آنکه برای خاتمه آن، سنجشی صورت بگیرد. یکی از معیارهای سنجش، آنست که در سرآیند هر بسته یک «شمارنده گام» (Hop Counter) داشته باشیم و در هر گام یک واحد از آن کم شود و هنگامی که مقدار این شمارنده به صفر رسید بسته حذف شود. حالت آرمانی آن است که مقدار اولیه این شمارنده معادل طول مسیر بین مبدا و مقصد در نظر گرفته شود ولیکن اگر فرستنده نداند که طول مسیر چقدر است می‌تواند مقدار این شمارنده را به بدترین حالت یعنی بزرگترین طول مسیر در شبکه تنظیم نماید.^۱

راهکار دیگر برای خاتمه دادن به این فرآیند آن است که فهرست بسته‌هایی که قبلاً یکبار به صورت سیل آسا ارسال شده‌اند را نگاه داریم تا از ارسال مجدد آن اجتناب شود. برای انجام این کار، مسیریاب مبدا باید در هر بسته

۱. در ادبیات شبکه به بزرگترین طول مسیر، «قطر شبکه» گفته می‌شود.

```

#define MAX_NODES 1024          /* maximum number of nodes */
#define INFINITY 1000000000     /* a number larger than every maximum path */
int n, dist[MAX_NODES][MAX_NODES]; /* dist[i][j] is the distance from i to j */
void shortest_path(int s, int t, int path[])
{
    struct state {               /* the path being worked on */
        int predecessor;        /* previous node */
        int length;             /* length from source to this node */
        enum {permanent, tentative} label; /* label state */
    } state[MAX_NODES];
    int i, k, min;
    struct state *p;
    for (p = &state[0]; p < &state[n]; p++) { /* initialize state */
        p->predecessor = -1;
        p->length = INFINITY;
        p->label = tentative;
    }
    state[t].length = 0; state[t].label = permanent;
    k = t;          /* k is the initial working node */
    do {            /* Is there a better path from k? */
        for (i = 0; i < n; i++) /* this graph has n nodes */
            if (dist[k][i] != 0 && state[i].label == tentative) {
                if (state[k].length + dist[k][i] < state[i].length) {
                    state[i].predecessor = k;
                    state[i].length = state[k].length + dist[k][i];
                }
            }
        /* Find the tentatively labeled node with the smallest label. */
        k = 0; min = INFINITY;
        for (i = 0; i < n; i++)
            if (state[i].label == tentative && state[i].length < min) {
                min = state[i].length;
                k = i;
            }
        state[k].label = permanent;
    } while (k != s);

    /* Copy the path into the output array. */
    i = 0; k = s;
    do {path[i++] = k; k = state[k].predecessor; } while (k >= 0);
}

```

شکل ۸-۵ الگوریتم دایجکسترا برای محاسبه کوتاهترین مسیر در یک گراف.

که از یک ماشین میزبان دریافت می‌کند یک «شماره ترتیب» قرار بدهد. هر مسیر یاب نیز باید فهرستی تشکیل بدهد و در آن شماره ترتیب بسته‌هایی را که قبلاً از یک مسیر یاب دریافت کرده، درج کند. با ورود هر بسته ابتدا شماره ترتیب آن درون این فهرست جستجو می‌شود؛ اگر شماره آن در این فهرست وجود داشته باشد دیگر بهیچوجه (به روش سیل آسا) ارسال نخواهد شد (چون قبلاً یکبار ارسال شده است).

برای آنکه از رشد بی‌نهایت این فهرست اجتناب شود در فهرست به ازای هر مسیر یاب مبداء، تنها یک شمارنده نگهداری می‌شود و مقدار این شمارنده یعنی k نشان می‌دهد که بسته‌های تا شماره k قبلاً دریافت شده‌اند. [یعنی در حقیقت فقط شماره ترتیب آخرین بسته تولید شده توسط یک مسیر یاب مشخص می‌شود. -م] وقتی بسته‌ای وارد یک مسیر یاب می‌شود آزمون تکراری بودن آن بسیار ساده است [اگر شماره آن k یا کمتر از k بود تکراری است؛ اگر بسته تکراری بود حذف می‌شود. بدین ترتیب به فهرست بسته‌های دریافتی با شماره کمتر از k نیازی نیست و فقط در اختیار داشتن آخرین شماره یعنی k کفایت می‌کند.

گونه دیگری از روش سیل آسا که قابلیت اجرایی بیشتری دارد به نام «روش سیل آسا بصورت انتخابی» (Selective Flooding) مشهور است. در این الگوریتم، مسیر یاب تمام بسته‌های دریافتی از یک خط را بر روی تمام خطوط خروجی ارسال نمی‌کند بلکه فقط آنها را بر روی خطوطی ارسال می‌کند که به صورت تخمینی در مسیر صحیحی قرار دارند. هدایت بسته‌هایی که قرار است به سمت غرب بروند به سمت شرق، توجه چندانی ندارد مگر آنکه توپولوژی زیر شبکه نوع عجیب و ویژه‌ای باشد و مسیر یاب از آن به درستی آگاه باشد.

الگوریتم سیل آسا در اغلب کاربردها عملی نیست ولیکن در برخی از موارد خاص کاربردهایی دارد. به عنوان مثال در کاربردهای نظامی که گاهی باید برای تعداد زیادی از مسیر یابها و در لحظه‌ای کوتاه، حجم زیادی اطلاعات ارسال شود، قدرت روش سیل آسا بسیار مطلوب خواهد بود. در پایگاه داده‌های توزیع شده (Distributed Database) گاهی لازم است که تمام پایگاههای داده بطور همزمان بهنگام‌سازی شوند؛ در چنین موردی نیز روش سیل آسا مفید خواهد بود. در شبکه‌های بی‌سیم تمام پیامهایی که توسط یک ایستگاه ارسال می‌شود توسط بقیه ایستگاههایی که در برد رادیویی آن قرار گرفته‌اند قابل شنود است و در حقیقت نوعی از روش سیل آسا محسوب می‌شود و برخی از الگوریتمها از این ویژگی در راستای کار خود بهره می‌گیرند. چهارمین کاربرد روش سیل آسا، آنست که می‌توان از آن به عنوان معیاری برای مقایسه با الگوریتمهای دیگر مسیر یابی بهره گرفت. در روش سیل آسا همیشه کوتاهترین مسیرها انتخاب می‌شود چرا که بسته‌ها از تمام مسیرهای ممکن به صورت موازی ارسال خواهد شد. طبعاً هیچ الگوریتم دیگری نمی‌تواند تأخیر کمتری نسبت به این روش داشته باشد (به شرط آنکه از سربار تحمیل شده توسط خود پروسه سیل آسا، قابل چشمپوشی باشد).

۵-۲-۵ مسیریابی بردار فاصله (Distance Vector Routing)

شبکه‌های کامپیوتری مدرن، عموماً بجای استفاده از روشهای ایستا که در قبل بدانها اشاره شد از روشهای پویا بهره می‌گیرند زیرا الگوریتمهای ایستا بار فعلی شبکه را در محاسبات مربوط به بهترین مسیرها دخالت نمی‌دهند. دو الگوریتم پویا یعنی «مسیریابی بردار فاصله» (DVR) و «مسیریابی مبتنی بر حالت لینک» (Link State Routing) از رایج‌ترین روشهای موجود هستند. در این بخش بطور خاص مروری بر الگوریتم نوع اول (یعنی بردار فاصله) خواهیم داشت و در بخش آتی به مطالعه الگوریتم دوم می‌پردازیم.

الگوریتم «مسیریابی بردار فاصله» (DVR) بدین نحو کار می‌کند: هر مسیر یاب جدولی را در خود نگه می‌دارد (به عبارتی یک بردار را) که در آن بهترین فاصله تا هر مسیر یاب مقصد [یا به عبارتی کمترین هزینه رسیدن به هر مسیر یاب در زیر شبکه] و خطی که برای رسیدن به آن مقصد باید مورد استفاده قرار بگیرد، درج شده است. این جداول با مبادله اطلاعات بین مسیر یابهای همسایه، بهنگام‌سازی می‌شود.

الگوریتمهای مسیریابی بردار فاصله گاهی با نامهای دیگری معرفی می شوند که اهم این اسامی عبارتند از «الگوریتم مسیریابی بلمن-فورد توزیع شده» (Distributed Bellman-Ford) و «الگوریتم فورد-فوکرسون» (Ford-Fulkerson) که به افتخار نام پژوهشگرانی است که آن را ابداع کرده اند. (Bellman, 1957; Ford and Fulkerson, 1962) این روش اولین الگوریتم مسیریابی در ARPANET بود و بعداً نیز با نام RIP (Routing Information Protocol) در اینترنت بکار گرفته شد.

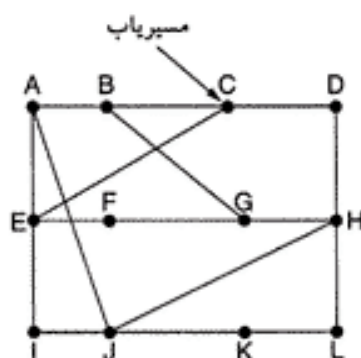
در «مسیریابی بردار فاصله» هر مسیریاب یک جدول مسیریابی دارد که به ازای هر مسیریاب موجود در زیر شبکه، یک «درایه» (Entry) در آن درج شده است. [کل جدول براساس آدرس هر مسیریاب، ایندکس شده است. -م] هر درایه دارای دو بخش است: (۱) خط خروجی مناسب برای رسیدن به مقصد مورد نظر (۲) تخمینی از زمان یا فاصله رسیدن بدان مقصد. معیار هزینه می تواند تعداد گام (Number of Hops)، تأخیر بر حسب میلی ثانیه، تعداد کل بسته های به صف شده در آن مسیر یا چیزی شبیه به اینها باشد.

فرض بر آن است که هر مسیریاب «فاصله» خود تا هر یک از همسایه هایش را می داند. اگر معیار بکار رفته تعداد گام باشد فاصله هر مسیریاب از همسایه هایش دقیقاً ۱ است. اگر معیار، طول صف باشد [یعنی تعداد بسته هایی که برای ارسال بر روی یک خط خروجی منتظر هستند] مسیریاب می تواند پسادگی هر یک از صفها را بررسی نماید. اگر معیار، تأخیر باشد مسیریاب می تواند با ارسال بسته های خاصی به نام Echo و دریافت بسته پاسخ (که گیرنده به آن «مهر زمان» (Timestamp) زده و سریعاً برمی گرداند) این تأخیر را مستقیماً اندازه بگیرد.

به عنوان مثال فرض کنید از میزان تأخیر به عنوان معیار بهینگی استفاده شده باشد و مسیریاب تأخیر خود تا هر یک از همسایه ها را بداند. هر T میلی ثانیه یکبار، تمام مسیریابها برای همسایه های خود فهرستی از تأخیر تخمینی در رسیدن به هر یک از مسیریابهای مقصد را ارسال می نماید. در ضمن فهرست مشابهی را از همسایه های خود دریافت می دارد. تصور کنید که یکی از این جداول از طرف مسیریاب X دریافت شود و نماد X_i مشخص کننده میزان تأخیر برای رسیدن به مسیریاب i باشد. اگر این مسیریاب بداند که تأخیر خودش تا X معادل m میلی ثانیه است می تواند نتیجه بگیرد که با تأخیر $X_i + m$ میلی ثانیه به هر مسیریاب i می رسد. با انجام این محاسبه برای تمام جداولی که از همسایه ها می رسند مسیریاب می تواند نتیجه بگیرد که کدام یک از مسیرها بهتر هستند و مقدار تخمینی تأخیر و همچنین خط متناظر با بهترین مسیر را در جدول جدید خود درج می نماید. بخاطر داشته که جدول قدیمی هر مسیریاب در محاسبات دخالت داده نمی شود.

فرآیند بهنگام سازی، در شکل ۵-۹ به تصویر کشیده شده است. قسمت «الف» از شکل، ساختار یک زیر شبکه را نشان می دهد. چهار ستون سمت چپ در بخش «ب» شکل، «بردار تأخیر» چهار همسایه مسیریاب J است که در لحظه بهنگام سازی دریافت شده است. مسیریاب A ادعا کرده که تا B دوازده میلی ثانیه تأخیر دارد؛ ۲۵ میلی ثانیه تأخیر تا C، ۴۰ میلی ثانیه تأخیر تا D و به همین ترتیب. فرض کنید مسیریاب J تأخیر خود را تا A و I و H و K به ترتیب مقادیر ۸، ۱۰، ۱۲ و ۶ اندازه گیری کرده یا تخمین زده باشد.

حال ببینیم مسیریاب J چگونه مسیرهای جدید برای رسیدن به مسیریاب G را محاسبه می کند. J می داند که برای رسیدن به A، ۸ میلی ثانیه تأخیر خواهد داشت و A نیز ادعا کرده که قادر است با تأخیر ۱۸ میلی ثانیه به G برسد، لذا J متوجه می شود که اگر بسته های خود را که به مقصد G روانه هستند برای A بفرستد کل تأخیر حدود ۲۶ میلی ثانیه [۸ + ۱۸] خواهد بود. به همین طریق، J تأخیر خود تا G از طریق I، H و K را به ترتیب ۴۱ (۱۰ + ۳۱)، ۱۸ (۶ + ۱۲) و ۳۷ (۶ + ۳۱) میلی ثانیه محاسبه می نماید. از بین این مقادیر، ۱۸ بهترین است لذا J در جدول مسیریابی خود، درایه ای برای G تشکیل می دهد و در آن درج می کند که تأخیر رسیدن به G، ۱۸ میلی ثانیه است و مسیر مربوطه از طریق H می گذرد. [یعنی از آن به بعد هر بسته ای که به J وارد شود و بخواهد به G برود به سمت



تخمین تأخیر دیگر
مسیریابها تا J

To	A	I	H	K	Line
A	0	24	20	21	8 A
B	12	36	31	28	20 A
C	25	18	19	36	28 I
D	40	27	8	24	20 H
E	14	7	30	22	17 I
F	23	20	19	40	30 I
G	18	31	6	31	18 H
H	17	20	0	19	12 H
I	21	0	14	22	10 I
J	9	11	7	10	0 -
K	24	22	22	0	6 K
L	29	33	9	9	15 K

تأخیر خط	تأخیر خط	تأخیر خط	تأخیر خط
JA	JI	JH	JK
8	10	12	6

جدول مسیریابی جدید در J

بردارهای دریافتی از چهار همسایه J

(الف) (ب)

شکل ۵-۹. (الف) یک زیر شبکه (ب) جداول دریافتی از A, I, H و K و جدول مسیریابی جدید در J.

مسیریاب H ارسال خواهد شد و تأخیر تخمینی کل مسیر، ۱۸ میلی ثانیه خواهد بود. -م] همین محاسبه برای دیگر مسیریابهای مقصد نیز انجام می شود و نهایتاً جدول مسیریابی جدید (نشان داده شده در سمت راست شکل ۵-۹) بدست خواهد آمد.

مشکل شمارش تا بی نهایت (Count-To-Infinity Problem)

مسیریابی بردار فاصله از دیدگاه تئوری به درستی کار می کند ولیکن در عمل اشکالات جدی خواهد داشت: اگرچه نهایتاً به جواب صحیح همگرا خواهد شد ولی این همگرایی بسیار کند خواهد بود.^۱ بالاخص، این الگوریتم به «خبرهای خوب» واکنش سریع نشان می دهد در حالی که «خبرهای بد» را به آهستگی منتقل می کند.^۲ یک مسیریاب را در نظر بگیرید که در جدول او تأخیر رسیدن به مقصد X، مقدار بسیار بالا، محاسبه و درج شده است. اگر در مرحله بعدی مبادله جداول [یعنی در لحظه بهنگام سازی]، مسیریاب همسایه او (مثلاً A)، هزینه کمتری را برای رسیدن به X اعلام کند، او بلافاصله مسیر قبلی خود را تغییر داده و از آن به بعد مسیر ارسال ترافیک به X را از طریق A انتخاب خواهد کرد. در هر بار مبادله بردارهای هزینه، «خبرهای خوب» به سرعت پردازش می شوند. برای آنکه ببینیم چگونه اخبار خوب منتشر می شوند به زیر شبکه پنج گره ای (خطی) شکل ۵-۱۰ دقت کنید که در آن معیار تأخیر «تعداد گام» است. فرض کنید A در همان ابتدا از کار افتاده و تمام مسیریابها از این موضوع آگاه هستند. به عبارت دیگر تمام مسیریابها تأخیر خود تا A را به مقدار بی نهایت تنظیم کرده اند. وقتی A به کار می افتد دیگر مسیریابها با مبادله بردار [جدول هزینه] بلافاصله از این موضوع آگاه خواهند شد. برای سادگی فرض کرده ایم یک ناقوس بزرگ در جایی وجود دارد که بطور متناوب به صدا در می آید تا همه

۱. منظور از همگرایی رسیدن به یک جدول مسیریابی با مقادیر درست، واقعی و پایدار است. -م

۲. منظور از خبرهای خوب کاهش تأخیر در مسیرها، اضافه شدن لینک یا مسیریاب جدید و منظور از خبرهای بد خرابی یک خط، از کار افتادن یک مسیریاب یا افزایش تأخیر است. -م

A	B	C	D	E		A	B	C	D	E	
•	•	•	•	•	Initially در بدو شروع	•	1	2	3	4	Initially در بدو شروع
	•	•	•	•	پس از اولین مبادله جدول		3	2	3	4	پس از اولین مبادله جدول
	1	•	•	•	پس از دومین مبادله جدول		3	4	3	4	پس از دومین مبادله جدول
	1	2	•	•	پس از سومین مبادله جدول		5	4	5	4	پس از سومین مبادله جدول
	1	2	3	•	پس از چهارمین مبادله جدول		5	6	5	6	پس از چهارمین مبادله جدول
	1	2	3	4			7	6	7	6	پس از پنجمین مبادله جدول
							7	8	7	8	پس از ششمین مبادله جدول
							•	•	•	•	

(الف)

(ب)

شکل ۵-۱۰. مشکل «شمارش تابی نهایت».

مسیریابها بطور همزمان عملیات مبادله بردارها (جدول) را شروع نمایند!! در اولین مبادله، B آگاه می شود که همسایه سمت چپ او تأخیری معادل صفر به A دارد. B در جدول مسیریابی خود یک درایه (Entry) ایجاد و در آن تأخیر رسیدن به A را ۱ درج می نماید. هنوز بقیه مسیریابها گمان می کنند که A غیرفعال است. تأخیر رسیدن به A در درایه های جدول مسیریابی تمام مسیریابها (در این لحظه) در سطر دوم از شکل ۵-۱۰-الف مشاهده می شود. در مبادله بعدی C متوجه می شود که B مسیری به A با طول ۱ دارد و به همین دلیل جدول خود را بهنگام می کند و در آن مسیری به A از طریق B با طول ۲ را مشخص می نماید ولی هنوز D و E از این خبر خوب آگاه نیستند. در یک زیرشبکه، اخبار خوب در هر بار مبادله جداول، یک گام به جلو منتشر می شود. اگر در یک زیرشبکه، طولانی ترین مسیر N گام باشد پس از N بار مبادله، همه مسیریابها از اضافه شدن خطوط یا مسیریابهای جدید به زیرشبکه (خبر خوب)، آگاه خواهند شد.

حال به وضعیت شکل ۵-۱۰-ب نگاه کنید که در آن تمام خطوط و مسیریابها در ابتدا فعال هستند. فاصله مسیریابهای A، B، C، D و E تا A به ترتیب عبارتند از: ۱، ۲، ۳ و ۴. به ناگاه مسیریاب A از کار می افتد یا مثلاً خط بین A و B قطع می شود که هر دوی این حالات از دیدگاه B فرقی نمی کنند.

پس از اولین مبادله بسته، B از A چیزی نمی شنود. خوشبختانه C می گوید: «نگران نباش! من مسیری به A به طول ۲ دارم». B به درستی نمی داند که مسیری که C اعلام کرده از خود او (یعنی B) می گذرد. آنچه که B حدس زده آنست که شاید C ده خط دیگر دارد و می توان از طریق این خطوط با هزینه ۲، به A رسید. در نتیجه، B گمان می کند که قادر است از طریق C با هزینه ۳ به A برسد! D و E درایه های جدول مسیریابی خود را در اولین مبادله تغییر نمی دهند.

در دومین مبادله، C متوجه می شود که هر یک از همسایه های او ادعا کرده اند که مسیری به A با هزینه ۳ دارند لذا یکی از آنها را انتخاب کرده و فاصله جدید رسیدن به A را به ۴ تنظیم می کند. (به گونه ای که در سطر سوم از شکل ۵-۱۰-ب نشان داده شده است). مبادلات بعدی جداول، نتایج نشان داده شده در ادامه شکل ۵-۱۰-ب را تولید خواهد کرد.

از این شکل باید روشن شده باشد که چرا خبرهای بد به کندی منتشر می شوند: همه مسیریابها هزینه جدید را یک واحد بیش از نتیجه مینیمم گیری از هزینه های اعلام شده، در نظر می گیرند. [به عبارت دیگر هزینه های جدید براساس مقادیری محاسبه می شود که قدیمی و اشتباه است. -م] به تدریج مقدار فاصله درج شده در جدول تمام مسیریابها به سمت بی نهایت رشد می کند ولیکن تعداد دفعات مبادله جداول به عددی که برای مقدار «بی نهایت» در

نظر گرفته شده بستگی دارد. به همین دلیل بهترین کار آن است که مقدار بی نهایت، معادل طول بزرگترین مسیر در زیر شبکه به اضافه ۱، در نظر گرفته شود. اگر معیار هزینه، «زمان تأخیر» باشد هیچ حد بالایی را نمی توان تعریف کرد مگر آنکه حد بالا را آنقدر زیاد فرض کنیم که با مسیری که بطور طبیعی تأخیر آن بالاست به عنوان مسیر از کار افتاده و خراب رفتار نشود. مشکل فوق الذکر به نام «مشکل شمارش تا بی نهایت» مشهور است. تلاشهایی برای حل این مشکل انجام گرفته (مثل روش split horizon with poisoned reverse که در RFC 1058 تشریح شده) ولی هیچکدام از آنها موفق عمل نکرده اند. اصل این مشکل از آنجایی ناشی شده که وقتی X به Y می گوید که مسیری به جایی دارد، Y بهیچوجه نمی تواند بفهمد که آیا خودش بر روی این مسیر قرار گرفته یا نه!

۵-۲-۵ مسیریابی حالت لینک (Link State Routing)

از «مسیریابی بردار فاصله» تا سال ۱۹۷۹ در ARPANET استفاده می شد و در همین ایام به «مسیریابی حالت لینک» (Link State) تغییر کرد. دو مشکل اساسی منجر به زوال آن شد: اول آنکه در این الگوریتم معیار تأخیر، طول صف^۱ در نظر گرفته می شد و پهنای باند هر یک از خطوط در محاسبات انتخاب بهترین مسیر، دخالت داده نمی شد. در ابتدا تمام خطوط 56Kbps بودند لذا پهنای باند خط مورد مهمی نبود ولی پس از آنکه برخی از خطوط به سرعت 230 Kbps و برخی دیگر به 44 Mbps ارتقاء یافتند، بحساب نیابردن پهنای باند، مشکلی عمده به حساب می آمد. البته این امکان وجود داشت که بتوان معیار تأخیر را به پهنای باند خط تغییر داد ولیکن مشکل دیگری ایجاد می شد و آن هم اینکه همگرایی الگوریتم بسیار طولانی می شد (بدلیل مشکل شمارش تا بی نهایت). به همین دلایل با الگوریتم کاملاً جدیدی که اکنون «مسیریابی حالت لینک» (LS) نامیده می شود، تعویض شد. امروزه گونه های متفاوتی از «مسیریابی حالت لینک» مورد استفاده قرار می گیرد. ایده اصلی و زیربنای مسیریابی حالت لینک (LS)، ساده است و می توان آن را در پنج بند بیان کرد. هر مسیریاب باید به ترتیب زیر عمل کند:

۱. همسایه های خود را شناسایی کرده و آدرسهای شبکه آنها را بدست بیاورد.
 ۲. تأخیر یا هزینه هر یک از همسایه های خود را اندازه گیری نماید.
 ۳. پسته ای بسازد و اطلاعاتی که از همسایه ها کسب کرده، در آن جاسازی کند.
 ۴. این پسته را برای تمام مسیریابهای دیگر بفرستد.
 ۵. کوتاهترین مسیر رسیدن به دیگر مسیریابها را محاسبه نماید.
- بدین ترتیب، توپولوژی کامل زیر شبکه و تمام تأخیرها (از طریق آزمایش) اندازه گیری شده و بین تمام مسیریابها توزیع می شود. برای پیدا کردن کوتاهترین مسیرها به تمام مسیریابهای زیر شبکه، می توان از «الگوریتم دایکسترا» بهره گرفت. در ادامه هر یک از این پنج مرحله را به تفصیل بررسی خواهیم نمود.

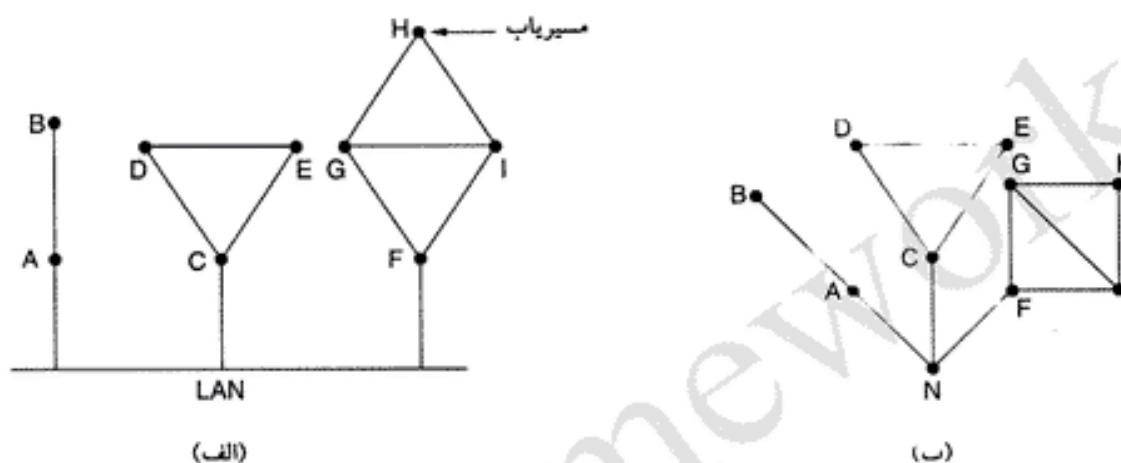
شناسایی همسایه ها

وقتی یک مسیریاب آغاز به کار می کند [بوت می شود] اولین وظیفه او شناسایی همسایه های خودش است. این کار با ارسال یک پسته خاص به نام «پسته سلام» (HELLO Packet) بر روی تمام خطوط نقطه به نقطه انجام می شود. انتظار می رود که مسیریابهایی که در طرف مقابل هر خط هستند پاسخی برگردانده و خود را معرفی کنند. این نامها [یا به عبارت بهتر آدرس مسیریابها] باید جهانی، منحصر به فرد و یکتا باشند چرا که بعداً وقتی یک مسیریاب راه دور می شنود که سه مسیریاب همگی مثلاً به F متصلند، دانستن این موضوع که آیا هر سه F یکی هستند بسیار

۱. تعداد پسته های به صف شده منتظر ارسال بر روی یکی از خطوط خروجی مسیریاب، به طول صف تعبیر می شود. -م

حیاتی است.

وقتی دو یا چند مسیر یاب از طریق LAN به هم متصل شده باشند وضعیت اندکی پیچیده تر است. شکل ۱۱-۵ الف یک شبکه LAN را نشان می دهد که سه مسیر یاب A، C و F مستقیماً بدان متصل شده اند. بگونه ای که نشان داده شده این مسیر یابها به یک یا چند مسیر یاب دیگر نیز متصلند. یک روش برای مدلسازی LAN آنست که همانند شکل ۱۱-۵ ب، آن را به عنوان یک گره در نظر بگیریم. در اینجا یک گره مصنوعی جدید به نام N معرفی کرده ایم که A و C و F بدانها متصل هستند. این واقعیت که رسیدن از A به C فقط از طریق LAN ممکن است با مسیر ANC بیان می شود.



شکل ۱۱-۵. الف) نه مسیر یاب و یک شبکه LAN. ب) مدل گراف از شکل الف.

اندازه گیری هزینه خط

«الگوریتم حالت لینک» نیازمند آن است که هر مسیر یاب از تأخیر هر یک از همسایه های خود آگاه باشد (یا حداقل تخمینی معقول از تأخیر آنها داشته باشد). راه مستقیم اندازه گیری این تأخیر آن است که یک بسته خاص به نام Echo بر روی خط مورد نظر ارسال شده و طرف مقابل موظف به برگرداندن فوری آن بسته باشد. با اندازه گیری زمان رفت و برگشت این بسته و تقسیم آن بر ۲، مسیر یاب فرستنده می تواند تخمینی معقول از تأخیر بدست بیاورد. برای بدست آوردن نتایج بهتر می توان این آزمون را چندین بار تکرار کرد و میانگین مقادیر اندازه گیری شده را در نظر گرفت. البته در این روش تلویحاً فرض بر آن گذاشته شده که تأخیرها متقارن^۱ است در حالی که ممکن است همیشه اینگونه نباشد.

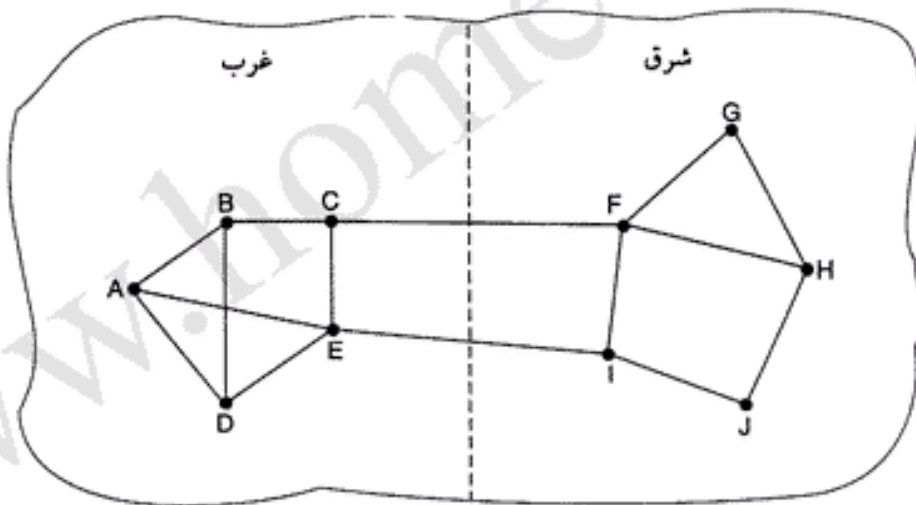
یک مسئله جالب آن است که آیا باید در اندازه گیری تأخیر، «میزان بار» (load) را نیز به حساب آورد؟ برای وارد کردن «میزان بار» در محاسبات، «تایمر سنجش زمان رفت و برگشت» (Round-Trip Timer) باید زمانی آغاز به کار کند که بسته Echo به انتهای صف ارسال وارد می شود. برای نادیده گرفتن میزان بار، تایمر زمانی شروع به اندازه گیری می کند که بسته Echo به سر صف رسیده باشد.

می توان در خصوص هر دوی این روشها بحث کرد. در نظر گرفتن تأخیرات ناشی از ترافیک در محاسبات، بدین معنی است وقتی یک مسیر یاب می خواهد از بین دو خط با پهنای باند مساوی یکی را انتخاب کند (در شرایطی که یکی از آنها برخلاف دیگری با بار سنگین مواجه است)، مسیر یاب خطی را که فاقد بار است به عنوان

۱. متقارن بودن تأخیر بدین معناست که تأخیر رسیدن از A به B با تأخیر رسیدن از B به A یکسان است، در حالیکه در بسیاری از محیطها اینگونه نیست. - م

مسیر کوتاهتر در نظر می‌گیرد. چنین انتخابی، کارایی بهتری را در بر خواهد داشت. متأسفانه استدلالی بر علیه نظریهٔ «دخالت دادن میزان بار در مسیر تأخیر» وجود دارد: به زیرشبکه شکل ۱۲-۵ توجه کنید که در آن زیرشبکه به دو بخش شرقی و غربی تقسیم و توسط دو خط CF و EI بهم متصل شده‌اند.

فرض کنید که بیشتر حجم ترافیک بین شرق و غرب از طریق خط CF مبادله شود و در نتیجه این خط با بار سنگین و تأخیر بالایی مواجه است. در نظر گرفتن تأخیر انتظار (یعنی تأخیر صف) در محاسبات کوتاهترین مسیر باعث می‌شود که خط EI جلب توجه نماید. پس از آنکه جدول جدید مسیریابی تنظیم و اعمال شود خط EI به عنوان خط بهینه انتخاب شده و از آن به بعد بخش اعظم ترافیک شرق به غرب از طریق EI عبور خواهد کرد و طبعاً این خط با بار سنگین مواجه خواهد شد. طبعاً در بهنگام‌سازی بعدی، مجدداً خط CF به عنوان مسیر بهینه شناخته می‌شود. در نتیجه ممکن است جدول مسیریابی بطور نوسانی تغییر کرده و منجر به مسیریابی نامنظم شده و اشکالات بالقوهٔ فراوانی تولید کند. اگر از میزان بار چشمپوشی شده و فقط پهنای باند در نظر گرفته شود این مشکل رخ نخواهد داد. البته می‌توان بار را بر روی هر دو خط توزیع کرد ولیکن در این راه حل از مسیر بهینه، استفادهٔ کامل نخواهد شد. علیرغم این، برای اجتناب از بروز تغییرات نوسانی در انتخاب بهترین مسیر، شاید توزیع بار بر روی چند خط راهکاری معقول باشد (البته باید کسری از بار که بر روی هر خط توزیع می‌شود از قبل تعیین شده باشد).

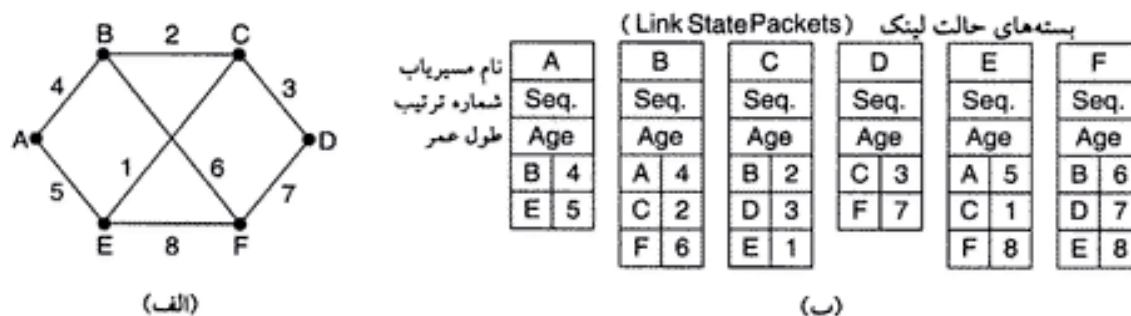


شکل ۱۲-۵. زیرشبکه‌ای که در آن دو بخش شرقی و غربی با دو خط به یکدیگر متصل شده‌اند.

ساخت بسته‌های حالت لینک (Building Link State Packet)

به محض آنکه اطلاعات لازم جهت مبادله گردآوری شد، گام بعدی هر مسیریاب، ساختن بسته‌ای است که این داده‌ها را در بر بگیرد.^۱ در ابتدای هر بسته، هویت فرستندهٔ آن درج می‌شود، سپس فیلدهای «شماره ترتیب» و «طول عمر» (Age) بسته می‌آید (در مورد طول عمر بسته در ادامه توضیح خواهیم داد)؛ در آخر نیز فهرست همسایه‌ها و تأخیر رسیدن بدانها، مشخص می‌شود. در شکل ۵-۱۳-الف یک زیرشبکه نمونه نشان داده شده که در آن، تأخیرها به صورت برچسب عددی، بر روی هر خط مشخص شده‌اند. بسته‌های LS ایجاد شده در هر یک

۱. «بسته‌های حالت لینک» را یک ساختمان دادهٔ استاندارد همانند یک استراکچر در زبان C در نظر بگیرید. از این به بعد این بسته‌ها را بسته‌های LS می‌نامیم. م



شکل ۵-۱۳. (الف) یک زیرشبکه (ب) بسته های حالت لینک (LS) برای این زیر شبکه.

از شش مسیریاب، در شکل ۵-۱۳ ب مشاهده می شود.

ساخت بسته های LS ساده است. مشکل ترین بخش قضیه تعیین زمانی است که مسیریابها باید به ساختن این بسته ها اقدام کنند. یک راه آن است که بطور متناوب و در فواصل زمانی مشخص ایجاد شوند. راه دیگر آن است که مسیریاب فقط زمانی مبادرت به ساخت این بسته ها کند که اتفاق خاصی رخ بدهد، مثلاً یکی از خطوط یا مسیریابهای همسایه از کار بیفتند یا مجدداً فعال شود یا ویژگیهای آن بطور محسوسی تغییر کند. [منظور از ویژگی، میزان تأخیر، بار، پهنای باند یا نظایر آنهاست. -م]

توزیع بسته های «حالت لینک»

ظریفترین بخش این الگوریتم، توزیع مطمئن بسته های LS است. به محض آنکه بسته ها، توزیع و در جداول اعمال شدند، آن مسیریاب که اول از همه این بسته ها را دریافت کند مسیرهای خود را تغییر می دهد [در حالیکه ممکن است بقیه هنوز چنین کاری نکرده باشند]. در نتیجه مسیریابهای متفاوت ممکن است برای لحظاتی از نسخه های متفاوتی از توپولوژی زیر شبکه استفاده کنند که این مسئله منجر به مشکلاتی در مسیریابی صحیح، بروز حلقه بینهایت، از دسترس خارج شدن برخی از ماشینها و مشکلات دیگر شود.

در ابتدا به تشریح الگوریتم اساسی توزیع می پردازیم و در ادامه اصلاحاتی را بر روی آن انجام خواهیم داد. ایده اصلی آن است که برای توزیع بسته های LS از روش «سیل آسا» استفاده شود. برای آنکه کنترل این فرآیند را در دست داشته باشیم هر بسته دارای شماره ترتیب است؛ به ازای تولید هر بسته جدید، به این شماره ترتیب یک واحد اضافه می شود. هرگاه مسیریابها بسته ای از این نوع را دریافت کنند زوج آیتیم «آدرس مسیریاب مبدا، شماره ترتیب» آنرا در جایی ذخیره می کنند. وقتی یک بسته LS جدید وارد شود ابتدا بررسی می شود که آیا این بسته قبلاً نیز دریافت شده است. اگر بسته جدید بود بر روی تمام خطوط خروجی (به استثنای خطی که از روی آن دریافت شده) ارسال می شود ولیکن اگر تکراری بود نادیده گرفته می شود. اگر بسته ای دریافت شود و شماره ترتیب آن از بزرگترین شماره ای که تاکنون دریافت شده، کوچکتر باشد به عنوان بسته قدیمی تلقی و در نظر گرفته نخواهد شد چرا که مسیریاب، نسخه جدیدتری از آن را در اختیار دارد.

این الگوریتم چند مشکل دارد ولی این مشکلات قابل مدیریت و کنترل هستند: اول آنکه اگر شماره ترتیب با افزایشهای متوالی، در جایی به صفر برگردد مشکلاتی بروز خواهد کرد. راه حل آن است که از یک شماره ترتیب ۳۲ بیتی استفاده شود؛ در این صورت اگر در هر ثانیه یک بسته LS ارسال شود، ۱۳۷ سال طول می کشد تا این عدد به صفر برگردد لذا احتمال بروز چنین مشکلی قابل چشمپوشی است!

دوم آنکه اگر یکی از مسیریابها از کار بیفتند روند شماره ترتیب بسته های خود را از دست خواهد داد و اگر بعد از راه اندازی مجدد، شماره ترتیب را از صفر شروع کند بسته های ارسالی او تکراری تلقی شده و

در نظر گرفته نمی شود.

سوم آنکه اگر شماره ترتیب به نحوی دچار خطا شود و مثلاً در اثر یک بیت خطا شماره ۴ به ۶۵۵۴۰ تبدیل شود از آن به بعد بسته های ۵ تا ۶۵۵۴۰ به عنوان بسته قدیمی دور انداخته خواهد شد چرا که گمان می رود که شماره فعلی ۶۵۵۴۰ است.

راه حل تمام این مشکلات آن است که پس از شماره ترتیب، طول عمر بسته نیز درج گردد و با گذشت هر ثانیه یک واحد از آن کم شود. وقتی عمر بسته به صفر برسد اطلاعاتی که از آن مسیر یاب دریافت شده باید حذف گردد. در یک روال طبیعی (مثلاً هر ده ثانیه یکبار)، بسته های جدید LS وارد مسیر یاب شده و طبقاً اطلاعات قبلی بطور منظم و قبل از انقضای مهلت اعتبارشان تازه سازی می شوند؛ بدین ترتیب اعتبار اطلاعات مربوط به هر مسیر یاب زمانی منقضی خواهد شد که آن مسیر یاب از کار بیفتد (یا آنکه مثلاً در اثر هر گونه رخداد، متوالیاً شش بسته LS از آن مسیر یاب دریافت نشود). فیلد «طول عمر» (Age) در خلال فرآیند ارسال سیل آسا و توسط هر مسیر یاب نیز یک واحد کاهش می یابد تا مطمئن شویم که هیچ بسته ای نمی تواند بطور نامحدودی زنده و در زیر شبکه سرگردان بماند. (هر گاه عمر بسته صفر شود، حذف خواهد شد.)

چند اصلاح در این الگوریتم، آنرا قدرتمندتر خواهد کرد. وقتی یک بسته LS برای ارسال به روش سیل آسا وارد یک مسیر یاب می شود بلافاصله در صف ارسال وارد نخواهد شد بلکه ابتدا به یک «فضای انتظار» وارد می شود تا برای مدتی کوتاه، منتظر بماند. اگر قبل از ارسال بسته فعلی، بسته مشابه دیگری از همان مبدا وارد شود، شماره ترتیب این دو بسته مقایسه شده و در صورت مساوی بودن، بسته تکراری حذف خواهد شد و در صورت تکراری نبودن از بسته قدیمی تر صرف نظر می شود. برای پیشگیری از خطاهای احتمالی بر روی خطوط مستقیم بین دو مسیر یاب، دریافت تمام بسته های LS ورودی، تصدیق (Ack) خواهد شد.^۱ وقتی یکی از خطوط خروجی مسیر یاب آزاد گردد «فضای انتظار» به روش Round Robin (نوبت چرخشی) پویش می شود تا یک بسته LS یا یک بسته اعلام وصول (یعنی Ack) برای ارسال انتخاب گردد.

در جدول شکل ۵-۱۴ ساختمان داده مورد استفاده در مسیر یاب B که برای زیر شبکه شکل ۵-۱۳ الف ایجاد شده، دیده می شود. در این جدول، هر سطر متناظر با یک بسته LS تازه وارد است که هنوز بطور کامل پردازش نشده است. در این جدول مبدا هر بسته، شماره ترتیب، طول عمر آن و مقداری داده ثبت می شود. بعلاوه به ازای هر یک از سه خط متصل به B (یعنی خطوط A و C و F) دو پرچم «ارسال» و «اعلام وصول»^۲ در نظر گرفته شده است. «پرچم ارسال» بدین معناست که بسته باید بر روی خط مشخص شده، ارسال شود. «پرچم اعلام وصول» بدین معناست که دریافت این بسته باید بر روی خط مربوطه، به آگاهی طرف مقابل برسد.

از جدول شکل ۵-۱۴، مشخص است که یک بسته LS مستقیماً از A دریافت شده و طبق آنچه که بیت های پرچم نشان می دهند این بسته باید برای C و F ارسال شود و ضمناً وصول آن به A اعلام گردد. به طور مشابه بسته ای که از F آمده باید به A و C نیز ارسال شده و دریافت آن به F اعلام گردد.

با این حال شرایط بسته سوم که از E می رسد متفاوت است. این بسته دو بار، یکبار از طریق EAB و یکبار از طریق EFB دریافت شده است. در نتیجه بنحوی که بیت های پرچم نشان می دهند، این بسته باید برای C ارسال شود ولیکن فقط کافی است وصول آن به A و F اعلام گردد.

اگر بسته ای تکراری دریافت شود در حالی که نسخه اصلی آن هنوز در بافر موجود است، بیت های پرچم باید تغییر کنند. به عنوان مثال اگر بسته متعلق به C قبل از آنکه مطابق با درایه چهارم (4th Entry) برای A و F ارسال

۱. دریافت هر بسته LS به مسیر یاب همسایه (که آنرا فرستاده) اعلام خواهد شد نه مبدا آن. سم

۲. Send Flag and Acknowledgement Flag.

Source (مبدأ)	Seq.	Age	Send flags			ACK flags			Data
			A	C	F	A	C	F	
A	21	60	0	1	1	1	0	0	
F	21	60	1	1	0	0	0	1	
E	21	59	0	1	0	1	0	1	
C	20	60	1	0	1	0	1	0	
D	21	59	1	0	0	0	1	1	

شکل ۵-۱۴. بافر بسته های LS برای مسیر یاب B در شکل ۵-۱۳.

شود یکبار دیگر از طریق F دریافت گردد، شش بیت پرچم به 100011 تغییر خواهد کرد بدین معنا که باید وصول بسته به F اعلام شود ولی لازم نیست خود بسته برای F ارسال گردد.

محاسبه مسیرهای جدید

به محض آنکه مسیر یاب مجموعه کامل بسته های LS را گردآوری کرد می تواند گراف کل زیر شبکه را تشکیل بدهد چرا که همه لینکها و هزینه آنها مشخص شده است. هر لینک در حقیقت دو بار معرفی شده است، یکبار در هر جهت. [چرا که مثلاً لینک بین A و B، یکبار توسط A و یکبار توسط B شناسایی و معرفی می شوند. -م] حال می توان الگوریتم دایکسترا را به صورت محلی اجرا کرد و کوتاهترین مسیر ممکن به تمام نقاط مقصد در زیر شبکه را بدست آورد.^۱ نتیجه این الگوریتم می تواند در جدول مسیر یابی درج شده و مسیر یابها عملکرد طبیعی خود را از سر بگیرند.

برای زیر شبکه ای با n مسیر یاب که هر کدام k همسایه دارند، حافظه لازم برای ذخیره داده های جدول مسیر یابی، متناسب با nxk است که در زیر شبکه های بزرگ می تواند مشکل ایجاد کند. در ضمن زمان محاسبه، نیز می تواند مسئله ساز باشد ولیکن علیرغم همه اینها «الگوریتم مسیر یابی حالت لینک» (Link State Routing) در محیطهای واقعی بخوبی کار می کند.

با این حال، هر گونه اشکال سخت افزاری یا نرم افزاری می تواند مشکلات بسیار جدی برای این الگوریتم به بار بیاورد. (همچنین در عملکرد الگوریتم دیگر مسیر یابها نیز مشکل ایجاد می کند.) به عنوان نمونه، اگر یک مسیر یاب ادعا کند که خطی در اختیار دارد در حالی که نداشته باشد یا فراموش کند خطی را که در اختیار دارد اعلام کند، گراف زیر شبکه صحیح نخواهد بود. همچنین اگر یک مسیر یاب در ارسال بسته های LS مشکل بهم بزند یا آنها را در حین ارسال خراب کند مشکلاتی جدی بروز خواهد کرد. سرانجام آنکه اگر حافظه مسیر یاب سرریز شود یا نتیجه محاسبات مسیر اشتباه باشد رخدادهای ناگواری در مسیر یابی بسته ها بوقوع می پیوندد. هر گاه رشد زیر شبکه به دهها، صدها یا حتی هزاران گره برسد احتمال از کار افتادن ناگهانی یک یا چند مسیر یاب را نمی توان نادیده گرفت. تنها راه ممکن آن است که تلاش شود تا در هنگام بروز چنین رخدادهایی میزان آسیبها و مشکلات در حد محدودی کنترل شود. پرلمن (Perlmán, 1988) این مشکلات و راه حل های آنها را تشریح کرده است. روش «مسیر یابی حالت لینک» بطور گسترده ای در شبکه های واقعی بکار گرفته شده است لذا مختصری در

۱. اجرای الگوریتم دایکسترا به صورت محلی بدین معناست که هر یک از مسیر یابها خودشان مستقل از دیگری آن را اجرا می کنند. -م

مورد چند پروتکل نمونه که از آن بهره گرفته اند خالی از لطف نیست. پروتکل OSPF که در اینترنت بسیار رایج است از همین الگوریتم استفاده می کند. در بخش ۴-۶-۵، OSPF را تشریح خواهیم کرد.

یکی دیگر از «پروتکل های مبتنی بر حالت لینک»، پروتکل IS-IS^۱ است که توسط DECnet طراحی شد و بعداً توسط ISO جهت بکارگیری در کنار پروتکل بی اتصال لایه شبکه خود یعنی CLNP، مورد پذیرش و تأیید قرار گرفت. البته بعداً در آن تغییراتی ایجاد شد تا بتواند با پروتکل های دیگری مثل IP نیز کار کند. پروتکل IS-IS در بخشهایی از ستون فقرات شبکه اینترنت (مثلاً در ستون فقرات قدیمی متعلق به NSFNET) و در برخی از سیستم های دیجیتال سلولی مثل CDPD بکار رفته است. شرکت Novell Network نیز گونه ساده تری از IS-IS را برای هدایت بسته های IPX خود بکار گرفته است.

IS-IS اساساً تصویری از توپولوژی مسیریاب را در زیر شبکه توزیع می کند و از طریق آن کوتاهترین مسیرها محاسبه می گردد. هر مسیریاب در هنگام اعلام اطلاعات مربوط به «حالت لینک» (LS) مشخص می کند که به چه آدرس هایی مستقیماً دسترسی دارد. (آدرسها مربوط به لایه شبکه و جهانی هستند). این آدرسها می توانند IP، IPX، AppleTalk یا هر آدرس دیگر باشند. IS-IS همچنین می تواند بطور همزمان از چندین پروتکل لایه شبکه پشتیبانی نماید.

بسیاری از نوآوری هایی که در IS-IS طراحی شده بود بعداً مورد پذیرش و استفاده OSPF قرار گرفت. (OSPF چندین سال پس از IS-IS ابداع شد). برخی از اینها عبارت بودند از روشی برای خاتمه خودکار فرآیند ارسال بسته ها به روش سیل آسا، مفهوم «مسیریاب برگزیده» (Designated Router) در شبکه LAN و روش محاسبه و پشتیبانی از تقسیم مسیر و همچنین تعریف معیارهای چندگانه هزینه. در نتیجه اختلاف ناچیزی بین IS-IS و OSPF وجود دارد. مهمترین اختلاف این دو آن است که IS-IS به گونه ای بسته های LS را تعریف و کد کرده است که بسادگی و بطور همزمان می تواند اطلاعاتی در خصوص چندین پروتکل لایه شبکه با خود حمل کند، خصوصیتی که OSPF فاقد آن است. این حسن در محیط های بزرگ که با چندین پروتکل مختلف کار می کنند بسیار ارزشمند است.

۶-۲-۵ مسیریابی سلسله مراتبی (Hierarchical Routing)

با رشد اندازه شبکه، جداول مسیریابی به همان نسبت رشد می کنند. جداول مسیریابی رو به رشد، نه تنها حافظه بیشتری مصرف می کنند بلکه به زمان CPU بیشتری برای پویش و جستجوی این جدول و همچنین پهنای باند زیادتری برای ارسال بسته های LS (بسته های گزارش وضعیت لینک) نیاز خواهد بود. ممکن است رشد شبکه بدان حد برسد که دیگر امکان نگهداری یک «دراپ» (Entry) به ازای هر مسیریاب، در جدول مسیریابی وجود نداشته باشد و مسیریاب مجبور به مسیریابی سلسله مراتبی (همانند شبکه تلفن) شود.

وقتی از مسیریابی سلسله مراتبی استفاده می شود، مسیریابها به تعدادی «منطقه» (Region) تقسیم می شوند؛ هر مسیریاب تمام جزئیات منطقه خود و مسیرهای دقیق رسیدن به هر مقصد در منطقه خود را می داند ولی چیزی در خصوص ساختار داخلی مناطق دیگر نمی داند. وقتی شبکه های مختلف بهم متصل می شوند طبیعی است که هر یک از آنها را به عنوان مناطق مجزا در نظر بگیریم تا مسیریابهای درون یک منطقه از دانستن ساختار توپولوژی ناحیه دیگر بی نیاز و فارغ باشند.

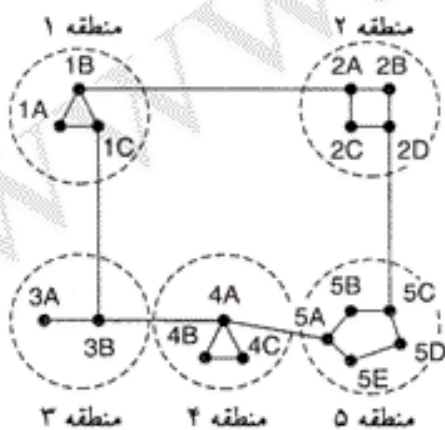
در شبکه های عظیم ممکن است سلسله مراتب دو سطحی کفایت نکند. ممکن است نیاز باشد که هر «منطقه» (Region) به تعدادی «دسته» (Cluster)، هر «دسته» به تعدادی «ناحیه» (Zone) و هر ناحیه به تعدادی «گروه»

(Group) تقسیم شود و به همین ترتیب ادامه یابد تا جایی که برای تقسیم بندی بیشتر اسمی باقی نماند!! به عنوان مثالی از سلسله مراتب چندسطحی، به چگونگی هدایت یک بسته از برکلی کالیفرنیا به مالدی در کنیا دقت کنید. مسیر یاب واقع در برکلی جزئیات توپولوژی زیر شبکه خود در کالیفرنیا را می داند ولیکن تمام ترافیک متعلق به خارج از ایالت کالیفرنیا را به مسیر یاب لوس آنجلس می فرستد. مسیر یاب واقع در لوس آنجلس تنها قادر به مسیریابی ترافیک داده ها بین مسیر یابهای داخل کشور است و هر گونه ترافیک خارجی را به نیویورک می فرستد. مسیر یاب واقع در نیویورک به گونه ای برنامه ریزی شده که این بسته را به سمت مسیر یابی که در کشور مقصد مسئول دریافت ترافیک خارجی است (مثلاً مسیر یاب نایروبی) بفرستد. نهایتاً این بسته در کنیا به همین سیاق مسیر خود را ادامه می دهد تا به مالدی برسد.

در شکل ۵-۱۵ یک نمونه کمی از مسیریابی سلسله مراتبی دو سطحی با پنج منطقه را ارائه کرده است. مطابق با شکل ۵-۱۵ جدول کامل مسیریابی در مسیر یاب 1A دارای ۱۷ «درایه» است. وقتی مسیریابی به صورت سلسله مراتبی انجام می شود به ازای هر مسیر یاب محلی یک درایه در جدول مسیریابی درج می شود ولیکن مناطق دیگر فقط در یک مسیر یاب خلاصه می شوند لذا کل ترافیک مربوط به منطقه ۲ از خط 1B-2A عبور خواهد کرد و مابقی ترافیک خارجی از طریق 1C-3B هدایت خواهد شد. مسیریابی سلسله مراتبی، تعداد درایه های جدول مسیریابی 1A را از ۱۷ به ۷ تا کاهش می دهد. [۳ تا برای مسیر یابهای داخل منطقه و ۴ تا به ازای مناطق دیگر] هر چه نسبت تعداد مناطق به تعداد مسیر یابهای درون هر منطقه افزایش یابد میزان صرفه جویی در فضای جدول بیشتر خواهد بود.

متأسفانه صرفه جویی در فضای حافظه رایگان بدست نمی آید. بهایی که باید برای آن پرداخت، افزایش طول مسیرها [یا به عبارتی عدم بهینگی کامل مسیرها] است. به عنوان مثال بهترین مسیر از 1A به 5C از منطقه ۲ می گذرد در حالی که در مسیریابی سلسله مراتبی مسیر تمام ترافیک متعلق به ناحیه ۵ از منطقه ۳ تعیین شده چرا که برای

جدول مسیریابی سلسله مراتبی 1A			جدول کامل مسیریابی 1A		
Dest.	Line	Hops	Dest.	Line	Hops
1A	—	—	1A	—	—
1B	1B	1	1B	1B	1
1C	1C	1	1C	1C	1
2A	1B	2	2	1B	2
2B	1B	3	3	1C	2
2C	1B	3	4	1C	3
2D	1B	4	5	1C	4
3A	1C	3			
3B	1C	2			
4A	1C	3			
4B	1C	4			
4C	1C	4			
5A	1C	4			
5B	1C	5			
5C	1B	5			
5D	1C	6			
5E	1C	5			



(الف)

(ب)

(ج)

شکل ۵-۱۵. مسیریابی سلسله مراتبی.

بیشتر مسیر یابهای مقصد واقع در ناحیه ۵، مسیر عبوری از منطقه ۲ مناسبتر است. سؤال قابل توجه آن است که وقتی یک شبکه واحد، بشدت رشد می کند، سلسله مراتب باید چند سطحی باشد؟ به عنوان مثال زیر شبکه ای با ۷۲۰ مسیر یاب را در نظر بگیرید. اگر سلسله مراتب وجود نداشته باشد هر مسیر یاب در جدول مسیر یابی خود به ۷۲۰ درایه نیاز خواهد داشت. اگر زیر شبکه به ۲۴ منطقه و ۳۰ مسیر یاب در هر منطقه تقسیم شود، هر مسیر یاب به ۳۰ درایه برای مسیر یابهای محلی خود به علاوه ۲۳ درایه برای مناطق خارجی، نیاز دارد و بدین ترتیب جدول جمعاً ۵۳ درایه خواهد داشت. اگر سلسله مراتب، سه سطحی انتخاب شده و جمعاً هشت «دسته» (Cluster)، هر دسته شامل ۹ ناحیه و در هر ناحیه ۱۰ مسیر یاب تعریف شده باشد، هر مسیر یاب در جدول خود به ۱۰ درایه برای مسیر یابهای محلی خود، ۸ درایه برای مسیر یابی در نواحی داخل دسته خودش و ۷ درایه برای دسته های خارجی (Distant Cluster) یعنی جمعاً ۲۵ درایه نیاز دارد. دو پژوهشگر بنامهای Kamoun و Kleinrock (۱۹۷۹) پی بردند که بهترین تعداد سطوح سلسله مراتب در زیر شبکه ای با N مسیر یاب، معادل $\ln(N)$ است و هر مسیر یاب به جمعاً $e \cdot \ln(N)$ عدد درایه نیاز خواهد داشت. همچنین نشان دادند که افزایش طول مؤثر مسیرها یا به عبارت دیگر کاهش بهینگی مسیرها، که از مسیر یابی سلسله مراتبی منشاء می گیرد بقدر کافی کم و نتیجه معمولاً قابل قبول است.

۷-۲-۵ مسیر یابی فراگیر (Broadcast Routing)

در برخی از کاربردها، ماشینهای میزبان نیازمند ارسال پیام به همه ماشینهای شبکه یا تعدادی از آنها هستند. به عنوان مثال برای خدمات توزیع گزارشات هواشناسی، بهنگام سازی اطلاعات بازار سهام یا برنامه های زنده رادیویی، راهکار مناسبتر آنست که داده ها را بصورت پخش فراگیر برای تمام ماشینها ارسال کرده و آنها را آزاد بگذاریم تا در صورت تمایل بسته های داده را دریافت کرده و بخوانند. «ارسال همزمان یک بسته به تمام ماشینهای مقصد، اصطلاحاً پخش فراگیر (Broadcasting) نامیده می شود و راهکارهای متنوعی برای اجرای آن پیشنهاد شده است.» یکی از روشهای پخش فراگیر که به هیچ ویژگی خاصی از زیر شبکه نیاز ندارد آن است که ماشین مبدا هر بسته خود را بطور جداگانه برای یکایک ماشینهای مقصد بفرستد. این روش نه تنها پهنای باند را تلف خواهد کرد بلکه ماشین مبدا نیازمند آن است که فهرست کاملی از تمام ماشینهای مقصد در اختیار داشته باشد. اگرچه گاهی در عمل این روش تنها راه ممکن است ولیکن روش چندان مطلوبی نیست.

روش ارسال سیل آسا، نامزد دیگری برای پخش فراگیر محسوب می شود. هر چند روش سیل آسا، راهکار نامناسبی برای شبکه های نقطه به نقطه است ولیکن برای ارسال فراگیر می توان بر روی آن حساب باز کرد، خصوصاً وقتی که هیچیک از راهکارهایی که در ادامه معرفی می شوند عملی نباشد. استفاده از روش ارسال سیل آسا همان مشکلی را دارد که در الگوریتم مسیر یابی نقطه به نقطه به آن اشاره کردیم: یعنی بسته های بسیار زیاد تولید می شود و پهنای باند بسیار وسیعی مصرف و تباه خواهد شد.

الگوریتم سوم «مسیر یابی چند مقصدی» (Multidestination Routing) نام دارد. اگر از این روش استفاده شود هر بسته فهرستی از کلیه مقصدهای مورد نظر یا «نقشه ای بیتی» (Bitmap) از این مقصدها با خود حمل می کند. هر گاه بسته ای به یک مسیر یاب برسد، آن مسیر یاب فهرست مقصدهای بسته را بررسی می کند تا مجموعه خطوط خروجی منتهی به هر مقصد تعیین شوند. (هر خط خروجی که حداقل به یکی از مقاصد مورد نظر منتهی شود، انتخاب می گردد.) مسیر یاب به ازای هر یک از خطوط خروجی تعیین شده، یک نسخه جدید از آن بسته را تولید کرده و در آن بسته فقط آدرس مقصدهایی را قرار می دهد که مسیر آنها بر روی خط خروجی مربوطه است. [بقیه آدرسهای مقصد حذف می شوند چرا که برای دیگر مقصدها خط خروجی جدا انتخاب و نسخه ای جداگانه تولید می شود. -م] در نتیجه مجموعه آدرسهای مقصد هر بسته بر روی خطوط جداگانه تقسیم و توزیع می شود.

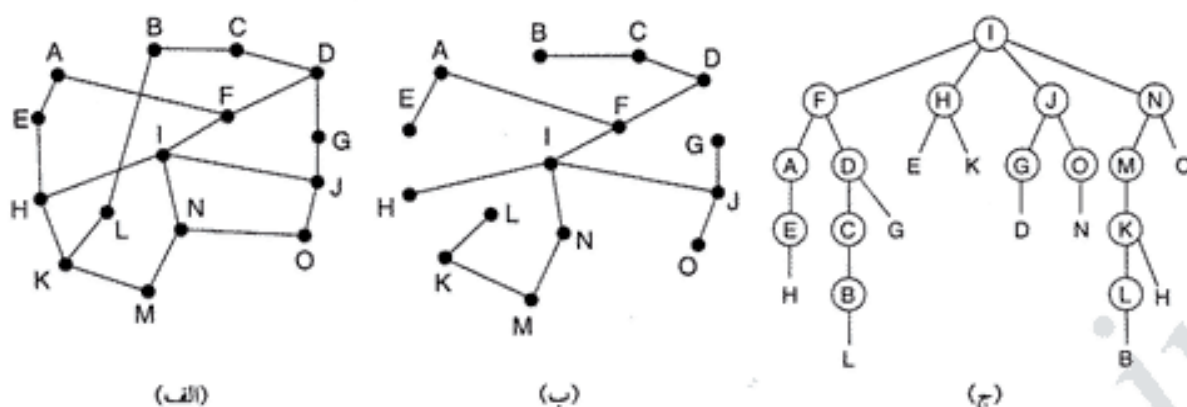
پس از چندین گام، هر بسته فقط یک آدرس مقصد با خود حمل می کند و با آن به عنوان یک بسته معمولی رفتار می شود. «مسیریابی چند مقصدی» همانند بسته های مستقل و با آدرس جداگانه عمل می کنند، با این تفاوت که بجای ارسال چندین بسته بر روی یک مسیر مشابه، فقط یک بسته ارسال و بدین نحو فقط هزینه ارسال یک بسته پرداخت می شود. [در این روش، هر بسته با رسیدن به یک مسیریاب به تعداد کافی تکثیر می شود. -م]

الگوریتم چهارم برای پخش فراگیر آن است که مسیریاب آغازگر این فرآیند، از درخت Sink Tree یا هر درخت پوشای مناسب (Spanning Tree) استفاده نماید. یک «درخت پوشا» زیر مجموعه ای از کل زیر شبکه است که تمام مسیریابها را در بر می گیرد ولیکن هیچ حلقه ای در آن نیست. اگر هر مسیریاب بداند که کدامیک از خطوط او در «درخت پوشا» قرار گرفته اند، می تواند یک بسته فراگیر ورودی را فقط بر روی خطوطی در خروجی، بفراستد که بر روی این درخت پوشا هستند. (البته به استثنای خطی که بسته از آن دریافت شده است.) بهره وری پهنای باند در این روش بسیار عالی است و برای انجام کار، کمترین بسته لازم تولید خواهد شد. تنها مشکل این روش آن است که هر مسیریاب برای یکارگیری این روش باید دانش و اطلاعات کافی در خصوص درختهای پوشا در زیر شبکه داشته باشد. برخی اوقات این اطلاعات در دسترس هستند (مثلاً در مسیریابی حالت لینک - LS) ولی گاهی موجود نیستند (مثل مسیریابی بردار فاصله - DV).

آخرین الگوریتم پخش فراگیر، آنست که حتی وقتی مسیریاب چیزی در خصوص درختهای پوشا نمی داند تلاش کند بصورت تقریبی، رفتار الگوریتم درخت پوشا را از خود نشان بدهد! این ایده، «هدایت بر روی مسیر معکوس» (Reverse Path Forwarding) نامیده شده و به نظر بسیار ساده می رسد. وقتی یک بسته از نوع پخش فراگیر به یک مسیریاب وارد می شود آن مسیریاب ابتدا بررسی می کند که «آیا بسته از روی همان خطی وارد شده که بسته های معمولی برای ارسال به مبدا آن بسته، بر روی همان خط ارسال می شوند؟» اگر اینگونه باشد این احتمال بالا وجود دارد که بسته خودش در مسیر بهینه قرار دارد و طبعاً اولین نسخه بسته ای است که او دریافت داشته است. در این حالت مسیریاب این بسته را بر روی تمام خطوط خود (به استثنای خطی که بسته از آن وارد شده) کپی و ارسال می کند. اگر بسته از روی خطی وارد شود که آن خط بر روی مسیر بهینه برای رسیدن بدان مبدا نیست بسته نادیده گرفته می شود چرا که احتمالاً تکراری است.

مثالی از روش «هدایت بر روی مسیر معکوس» در شکل ۵-۱۶ نشان داده شده است. بخش (الف) زیر شبکه را نشان می دهد و بخش (ب) درخت Sink Tree به مبدا مسیریاب I و بخش (ج) عملکرد الگوریتم «هدایت بر روی مسیر معکوس» را به تصویر کشیده است. در اولین گام، بسته ای فراگیر برای F و H و J و N (یعنی گره های سطح دوی درخت) می فرستد. هر یک از این بسته ها از طریق خطی دریافت می شوند که بر روی مسیری بهینه به I قرار دارند^۱ (با فرض آنکه مسیرهای بهینه از F و H و J و N به مسیریاب I، بر روی درخت Sink Tree قرار گرفته اند)؛ در شکل به دور نام هر مسیریاب که بر روی مسیری بهینه به I قرار گرفته یک دایره ترسیم شده است. در گام دوم، بسته تولید می شود (در هر مسیریاب که در گام اول بسته ای را دریافت کرده، بسته تولید می گردد) از این هشت بسته ۵ عدد به مسیریابی می رسند که بر روی مسیر بهینه به سمت I قرار دارند [یعنی A و D و G و O و M؛ سه بسته دیگر چون بر روی مسیر بهینه به I نیستند پس از دریافت می شوند. -م]. از شش بسته ای که در گام سوم تولید شده فقط سه تای آنها از طریق خطی که بر روی مسیر بهینه به I قرار دارد دریافت شده اند (از طریق C و E و K) و بقیه تکراری تلقی می شوند. پس از پنج گام و تولید ۲۴ بسته، فرآیند ارسال فراگیر به پایان می رسد، در حالیکه اگر ارسال بسته ها دقیقاً از طریق درخت Sink Tree انجام شود، تعداد گامها ۴ و کل بسته های تولیدی

۱. عبارت دیگر مسیرهای N، J، H، F همگی برای ارسال یک بسته معمولی برای مبدا بسته یعنی I، از همان خطی استفاده می کنند که بسته پخش از روی همان خط وارد شده است. -م



شکل ۵-۱۶. هدایت بر روی مسیر معکوس (الف) یک زیر شبکه (ب) یک درخت Sink Tree (ج) درختی که در روش هدایت بر روی مسیر معکوس ساخته می شود.

۱۴ عدد خواهد بود.

مزیت اساسی روش «هدایت بر روی مسیر معکوس» آن است که هم در حد معقولی کارآمد و هم پیاده سازی آن ساده است. در این روش هر مسیریاب نیازی به دانستن درختهای پوشا (Spanning Tree) ندارد و سربار ناشی از قرار دادن فهرست مقاصد چندگانه در بسته های فراگیر تحمیل نخواهد شد. همچنین این روش نیاز به مکانیزم خاصی جهت خاتمه دادن به پروسه تولید نامحدود بسته ها ندارد در حالی که در روش ارسال سیل آسا به چنین مکانیزمی نیاز است (یعنی در این روش به تمهیداتی مثل درج شمارنده گام در هر بسته یا اطلاع قبلی از قطر زیر شبکه یا ذخیره فهرست بسته هایی که تاکنون از یک مبدا منتشر شده اند، نیاز نیست).

۸-۲-۵ مسیریابی چندپخش (Multicast Routing)

در برخی از کاربردها نیاز است که پروسه های مجزا و پراکنده، در یک گروه با یکدیگر کار کنند؛ به عنوان نمونه می توان به گروهی از پروسه ها که یک «سیستم پایگاه توزیع شده اطلاعات»^۱ را پیاده سازی کرده اند، اشاره نمود. در چنین شرایطی، بطور متناوب لازم می شود که یک پروسه برای دیگر پروسه های عضو گروه خود، پیام بفرستد. اگر این گروه کوچک باشد می توان پیام را بطور جداگانه برای یکایک اعضا ارسال کرد ولیکن اگر گروه بزرگ باشد این راهکار بسیار پر هزینه است. برخی اوقات می توان از روش پخش فراگیر (Broadcasting) بهره گرفت اما روش پخش فراگیر برای رساندن پیامی به ۱۰۰۰ ماشین بر روی شبکه ای که یک میلیون گره دارد، بهیچوجه کارآمد نیست زیرا اکثر گیرندگان پیام هیچ تمایلی به دریافت این گونه پیامها ندارند (بدتر از همه آنکه دیگران علاقمند به دریافت چنین پیامهایی باشند ولی مجاز به دریافت آنها نباشند) بنابراین به راهکاری نیازمندیم که بتوان پیامهایی را برای گروه کاملاً مشخصی ارسال کرد؛ گروهی که اگرچه از لحاظ تعداد، بزرگ به نظر می رسد ولی در مقایسه با کل شبکه بسیار کوچک است.

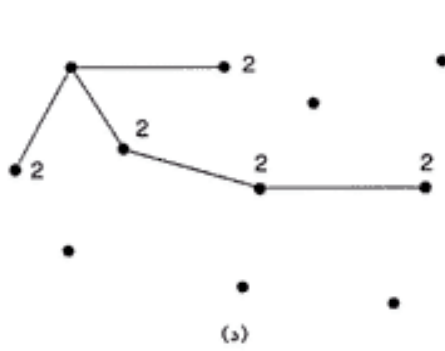
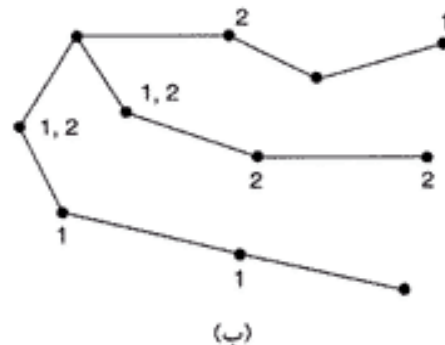
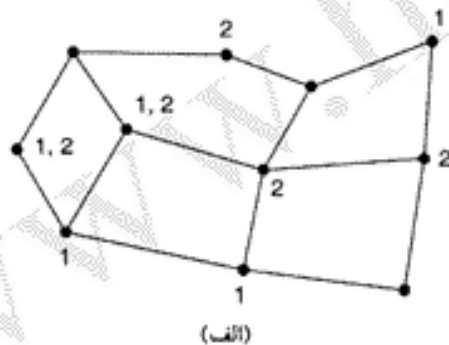
ارسال یک پیام به یک چنین گروهی، «چندپخشی» (Multicasting) و به الگوریتم مسیریابی آن «مسیریابی چندپخشی» (Multicast Routing) گفته می شود. در این بخش یکی از روشهای «مسیریابی چندپخشی» را مطالعه خواهیم نمود. برای کسب آگاهی بیشتر به مراجع زیر نگاهی بیندازید:

(Chu et al., 2000; Costa et al., 2001; Kaseria et al., 2000; Madruga and Garcia-Luna-Aceves, 2001; Zhang and Ryu, 2001).

مسیریابی چندپخشی به مدیریت گروه نیاز دارد؛ یعنی به روشهایی احتیاج است که بتوان گروه ایجاد و حذف کرد و پروسه ها اجازه داشته باشند به گروههایی بپیوندند یا از آنها جدا شوند. چگونگی انجام چنین کارهایی به الگوریتم مسیریابی ربطی ندارد. وقتی پروسه ای به گروهی می پیوندد باید ماشین میزبان خود را از این موضوع آگاه کند. آنچه اهمیت دارد آنست که مسیریاب می داند هر یک از ماشینهای میزبان او عضو چه گروههایی هستند. ماشینهای میزبان باید هر گونه تغییر عضویت خود در یک گروه را به اطلاع مسیریابهای خود برسانند یا آنکه مسیریابها باید خودشان بطور متناوب در این خصوص از ماشینهای میزبان سؤال کنند. به هر حال، مسیریابها از اینکه چه ماشینی عضو چه گروهی است باخبر می شوند. از آن به بعد، مسیریابها به همسایه های خود خبر می دهند و بدین ترتیب این اطلاعات در کل زیر شبکه منتشر خواهد شد.

برای انجام عملیات مسیریابی چندپخشی، هر مسیریاب یک «درخت پوشا» (Spanning Tree) که کل مسیرها را در بر می گیرد، ایجاد می کند. به عنوان مثال در شکل ۱۷-۵ الف، دو گروه ۱ و ۲ را تعریف کرده ایم. به نحوی که در این شکل ملاحظه می کنید برخی از مسیریابها به ماشینهای میزبانی متصلند که بعضاً عضو یک یا هر دوی این گروهها هستند. [در این شکل، برچسب ۱ یا ۲ بر روی هر مسیریاب، مبین آنست که ماشینهای میزبان متصل به آن مسیریاب عضو چه گروهی هستند. -م] در شکل ۱۷-۵ ب، یک درخت پوشا برای مسیریاب سمت چپ نشان داده شده است.

وقتی پروسه ای، یک بسته چندپخشی (Multicast Packet) را برای گروهی ارسال می کند اولین مسیریاب، درخت پوشای خود را بررسی کرده و آن را پیرایش (Prune) می کند، یعنی تمام خطوطی را که نهایتاً به ماشینهای عضو این گروه ختم نمی شوند، حذف می نماید. در این مثال شکل ۱۷-۵ ج درخت پوشا و پیرایش شده گروه ۱ را نشان می دهد. به روش مشابه شکل ۱۷-۵ د درخت پوشا و پیرایش شده گروه ۲ را به تصویر کشیده است. بسته های چندپخشی فقط از طریق درخت پوشای متناسب با گروه مقصد هدایت می شوند.



شکل ۱۷-۵. (الف) ساختار یک شبکه (ب) درخت پوشا برای مسیریاب سمت چپ (ج) درخت چندپخشی برای گروه ۱ (د) درخت چندپخشی برای گروه ۲.

روشهای گوناگونی برای پیرایش درخت پوشا وجود دارد. ساده‌ترین روش، زمانی قابل استفاده است که از روش «مسیریابی حالت لینک» (LS) بهره گرفته شده باشد و هر مسیریاب از توپولوژی کامل شبکه و از جمله عضویت ماشینهای میزبان در گروه‌ها آگاه باشد. در این حالت می‌توان با شروع از انتهای هر مسیر و حرکت به سمت ریشه درخت و حذف تمام مسیریابهایی که در گروه مورد نظر عضو نیستند، درخت پوشا را پیرایش کرد. در روش «مسیریابی بردار فاصله» (DV) می‌توان از راهکار متفاوتی برای پیرایش درخت پوشا بهره گرفت. الگوریتم اصلی همان روش «هدایت بر روی مسیر معکوس» است (Reverse Path Forwarding)، ولیکن هرگاه یک مسیریاب دارای هیچ ماشینی عضو یک گروه خاص، نبوده و همچنین به هیچ مسیریاب دیگری که علاقمند به دریافت بسته‌های آن گروه است، متصل نشده باشد با ارسال یک بسته خاص به نام PRUNE به فرستنده اعلام می‌کند که بسته‌های متعلق بدان گروه خاص را برایش نفرستد. اگر یک مسیریاب از تمام خطوط ورودیش پیغام PRUNE دریافت کند و خودش نیز ماشینی که در آن گروه عضو است نداشته باشد، او نیز پیغام PRUNE را صادر می‌نماید. در چنین حالتی، زیر شبکه به صورت بازگشتی (Recursively) پیرایش می‌شود.^۱

اشکال بالقوه این الگوریتم آن است که در مقیاس شبکه‌های بزرگ، ضعیف عمل می‌کند. فرض کنید که شبکه‌ای دارای n گروه و در هر گروه بطور متوسط m عضو وجود داشته باشد. به ازای هر گروه باید m «درخت پیرایش شده پوشا» ایجاد و ذخیره گردد که جمعاً معادل $m \times n$ درخت است. وقتی گروه‌های زیادی ایجاد شده باشد برای ذخیره این درختها، به حجم حافظه قابل توجهی نیاز خواهد بود.

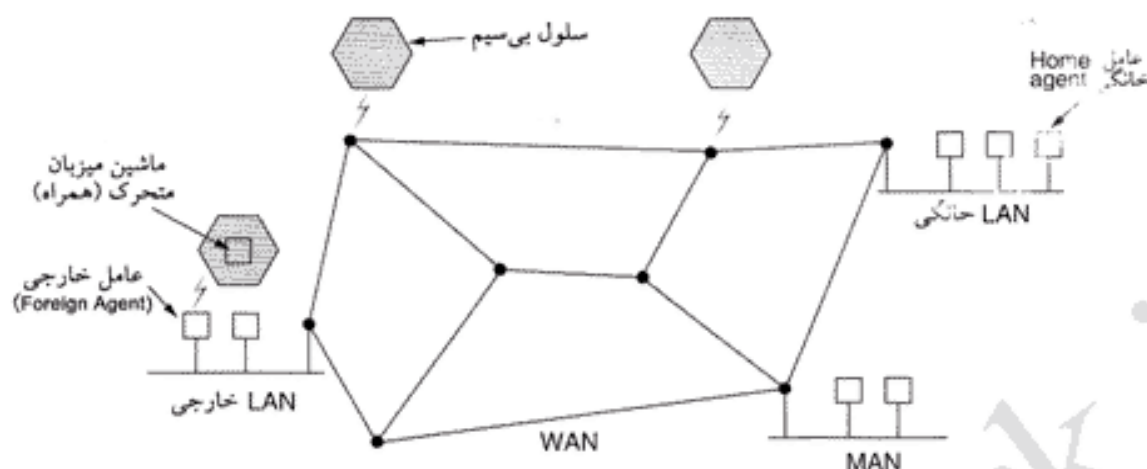
در طرحی دیگر، از «درختهای مبتنی بر هسته» (Core-Based Tree) بهره گرفته شده است. (Ballardie et al., 1993) در این طرح به ازای هر گروه فقط یک درخت پوشا محاسبه می‌شود، بگونه‌ای که «ریشه درخت» (یا عبارتی هسته) در نزدیکی مرکز آن گروه قرار می‌گیرد. برای ارسال یک پیام چندپخشی، ماشین میزبان آن را به سوی هسته می‌فرستد و هسته نیز عملیات پخش آن بسته را در گروه، از طریق درخت پوشا انجام می‌دهد. اگرچه این درخت برای تمام مسیریابهای مبداء کاملاً بهینه نیست ولی در عوض کاهش هزینه ذخیره‌سازی m درخت به یک درخت در هر گروه، صرفه جویی بسیار قابل توجهی محسوب می‌شود.

۹.۲.۵ مسیریابی برای ماشینهای متحرک

امروزه میلیونها نفر از مردم کامپیوترهای قابل حمل دارند و عموماً تمایل دارند در هر کجای دنیا که باشند نامه‌های الکترونیکی خود را بخوانند و به سیستم فایل همیشگی خود دسترسی داشته باشند. این ماشینهای متحرک مشکل جدیدی را بوجود می‌آورند: قبل از هدایت یک بسته به سوی ماشین متحرک، در ابتدا شبکه باید آن ماشین را پیدا کند. موضوع پیوستن ماشینهای میزبان متحرک (Mobile Hosts) به شبکه، موضوعی بسیار جدید و نو است ولیکن در این بخش به ارائه برخی از راهکارها در این زمینه خواهیم پرداخت.

مدلی که طراحان شبکه عموماً برای ساختار ارتباطی جهان در نظر می‌گیرند در شکل ۵-۱۸ نشان داده شده است. در این شکل یک WAN با تعدادی مسیریاب و ماشین میزبان دیده می‌شود. به WAN تعدادی LAN،

۱. عبارت ساده‌تر در ابتدای کار هر بسته چندپخشی بروش «هدایت بر روی مسیر معکوس» که در بخش قبل بررسی شد بر روی خطوط خروجی مسیریابها هدایت می‌شوند و تا اینجای کار هیچ تفاوتی با پخش فراگیر ندارد. به محض پخش اولین بسته بصورت فراگیر، تمام مسیریابهایی که انتظار دریافت چنین بسته‌ای را نداشته‌اند با ارسال بسته کنترلی Prune از همسایه خود که این بسته را برایشان فرستاده خواهش می‌کنند این کار را تکرار نکنند. وقتی یک مسیریاب خودش هیچ عضوی در این گروه ندارد و تمام همسایه‌هایش نیز با ارسال Prune از او خواسته‌اند که بسته‌های آن گروه را برایشان نفرستد، لزومی به دریافت بسته‌های آن گروه نمی‌بیند و او هم پیغام Prune به همسایه قبلی خود می‌فرستد. بدین ترتیب از آخر مسیر به اول درخت پوشای فرضی اصلاح می‌شود. -م



شکل ۵-۱۸. یک شبکه WAN که شبکه های LAN، MAN و سلولهای بی سیم بدان متصل شده اند.

MAN و شبکه بی سیم (از نوعی که در فصل ۲ مطالعه کردیم)، متصل شده است. به ماشینهای میزبان که حرکت نمی کنند ماشینهای ثابت (Stationary) گفته می شود. این ماشینها از طریق سیمهای مسی یا فیبرهای نوری به شبکه متصل شده اند. برعکس، دو نوع دیگر ماشین میزبان، قابل رؤیت است: «ماشینهای میزبان مهاجر» (Migratory Hosts) ماشینهای ثابتی هستند که گاه به گاه از یک سایت ثابت به سایتی دیگر نقل مکان می کنند ولی فقط زمانی در شبکه قرار می گیرند که اتصال فیزیکی آنها برقرار شود. «ماشینهای میزبان سیار» (Roaming Hosts) در حال حرکت هم کار می کنند و باید ارتباط خود را در حین حرکت حفظ کنند. ما این دو رده از ماشینهای میزبان را «ماشینهای میزبان متحرک» (Mobile Hosts) می نامیم: یعنی تمام ماشینها دور از محل استقرار همیشگی خود هستند و می خواهند به شبکه متصل شوند.

فرض بر آن است که تمام ماشینها دارای یک «محل استقرار دائمی» (Home Location) هستند که هیچگاه تغییر نمی کند. همچنین هر ماشین میزبان دارای آدرس ثابتی در محل استقرار دائم خود است که موقعیت این محل را مشخص می کند (شبه به یک شماره تلفن در آمریکا مثل ۱۲۱۲۰۵۵۵۱۲۱۲، با کد کشوری ۱ و کد منطقه، ۲۱۲). هدف نهایی فرآیند مسیریابی در سیستمی با ماشینهای متحرک، آنست که بتوان بسته ها را به کمک آدرس دائم آنها به ماشینها رساند و ماشینهای میزبان (در هر کجا که باشند) بتوانند بسته های خود را تحویل بگیرند. ظریفترین بخش قضیه آنست که ابتدا باید این ماشینها را پیدا کرد.

در مدل شکل ۵-۱۸ کل دنیا از نظر جغرافیایی به چندین بخش کوچک تقسیم شده است. اجازه بدهید آنها را «ناحیه» بنامیم؛ «ناحیه»، خود یک شبکه LAN یا یک «سلول بی سیم» (Wireless Cell) است. در هر ناحیه یک یا چند «عامل خارجی» (Foreign Agent) وجود دارد که در عمل پروسه هایی هستند که فهرست ماشینهای متحرک و میهمان آن ناحیه را پیگیری و مدیریت می کنند. بعلاوه هر ناحیه یک «عامل خانگی» (Home Agent) دارد که فهرست ماشینهایی را نگهداری می کند که محل استقرار دائمی آنها همین ناحیه است ولی فعلاً در نواحی دیگر به سر می برند.

وقتی ماشین جدیدی وارد یک ناحیه می شود (از طریق اتصال فیزیکی با شبکه مثلاً با اتصال کابل آن به LAN و یا با پرسه زدن در درون سلول بی سیم)، باید خودش را در «عامل خارجی» ثبت نام نماید. روال ثبت نام عموماً به شکل زیر انجام می شود:

۱. هر «عامل خارجی» بطور متناوب یک بسته فراگیر پخش کرده و حضور خود و آدرسش را به اطلاع همه

می‌رساند. یک ماشین متحرک تازه وارد باید منتظر چنین پیامهایی بماند ولیکن اگر در اسرع وقت، چنین پیامی نرسد ماشین متحرک می‌تواند بسته‌ای فراگیر برای همه بفرستد با این مضمون که: «آیا هیچ عامل خارجی در اینجا هست؟»

۲. ماشین متحرک در عامل خارجی ثبت‌نام می‌کند، آدرس محل استقرار دائم خود، آدرس فیزیکی (یعنی آدرس لایه پیوند داده‌ها) فعلی خود و برخی اطلاعات امنیتی (جهت احراز هویت) را ارائه می‌دهد.

۳. عامل خارجی با عامل خانگی آن ماشین متحرک تماس گرفته و می‌گوید: یکی از ماشینهای شما در ناحیه ما به سر می‌برد! در این پیام که از سوی «عامل خارجی» به سوی «عامل خانگی» ارسال می‌شود آدرس شبکه عامل خارجی [آدرس خودش] مشخص می‌شود. این پیام همچنین شامل برخی اطلاعات امنیتی است تا عامل خانگی را متقاعد کند که ماشین متحرک واقعاً در ناحیه اوست.

۴. عامل خانگی اطلاعات امنیتی ارائه شده را که محتوی «مهر زمان» (Timestamp) نیز هست بررسی می‌کند تا برایش ثابت شود که این پیام در خلال همین چند ثانیه اخیر تولید شده است. اگر متقاعد شود به «عامل خارجی» اجازه ادامه کار را می‌دهد.

۵. وقتی «عامل خارجی» پیام تأیید «عامل داخلی» را دریافت کند یک درایه (Entry) در جدول خود ایجاد کرده و به ماشین متحرک اعلام می‌کند که ثبت‌نام شده است.

آرامانی آن است که وقتی یک ماشین میزبان ناحیه‌ای را ترک می‌کند به همان ترتیب اعلام کند تا عضویت آن لغو شود، ولی بسیاری از کاربران به محض آنکه کارشان تمام می‌شود بلافاصله کامپیوتر خود را خاموش می‌کنند و مهلتی برای اعلام ترک ناحیه، باقی نمی‌ماند!

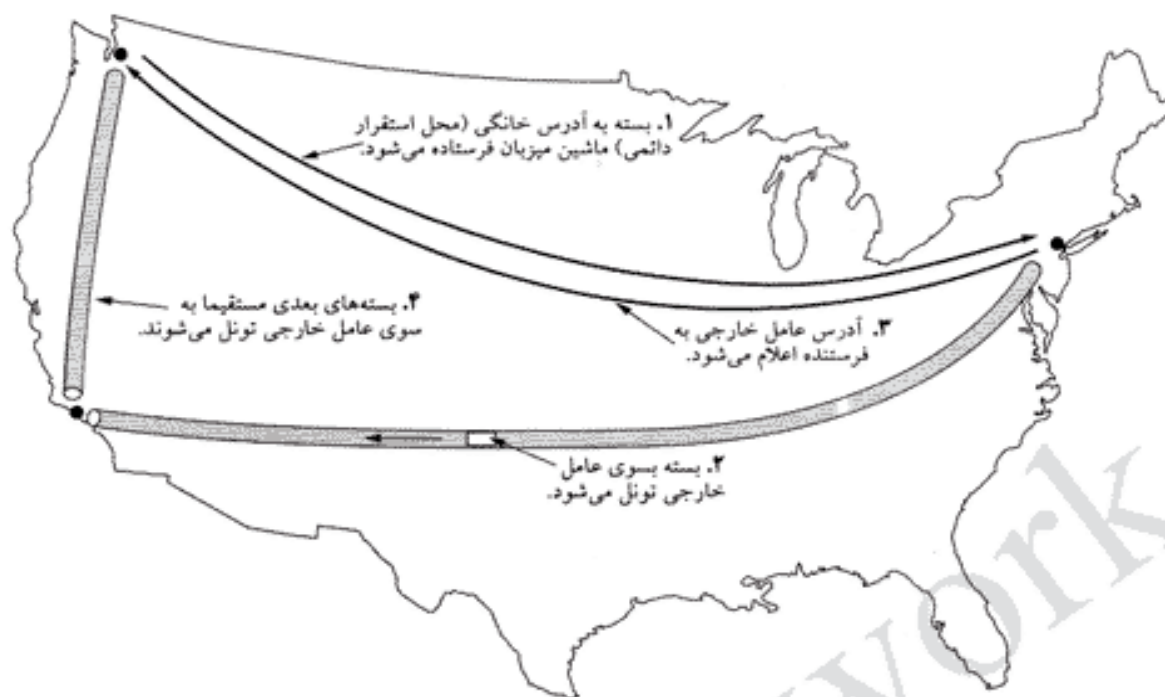
وقتی بسته‌ای برای یک ماشین متحرک ارسال می‌شود ابتدا آن بسته به سوی شبکه LAN خانگی او [یعنی محل استقرار دائم او] مسیریابی و هدایت می‌شود زیرا در حقیقت بسته به آدرس دائم او ارسال شده است. (مرحله ۱ از شکل ۵-۱۹) در این شکل فرستنده‌ای در شهری از سیاتل آمریکا می‌خواهد بسته‌ای را برای یک ماشین میزبان متحرک در نیویورک بفرستد. بسته ابتدا به آدرس شبکه LAN خانگی او در نیویورک ارسال می‌شود و در آنجا توسط «عامل خانگی» تحویل گرفته می‌شود. عامل خانگی در حافظه خود محل استقرار جدید (و موقت) ماشین متحرک را جستجو کرده و آدرس عامل خارجی که آن ماشین را در پوشش خود گرفته (مثلاً در لوس آنجلس) پیدا می‌کند.

در این لحظه، عامل خانگی دو کار انجام می‌دهد: اول آن که این بسته را در درون فیلد داده از یک بسته بیرونی دیگر جاسازی کرده و آن را برای عامل خارجی می‌فرستد. (مرحله ۲ از شکل ۵-۱۹) به این مکانیزم «ایجاد تونل» (Tunneling) گفته می‌شود و بعداً به تفصیل در مورد آن صحبت خواهیم کرد. پس از دریافت بسته جاسازی شده توسط عامل خارجی، بسته اصلی از درون فیلد داده آن جدا شده و به ماشین متحرک تحویل می‌شود.

دوم آنکه عامل خانگی به فرستنده بسته اعلام می‌کند که از این به بعد بسته‌های خود را با جاسازی درون یک بسته دیگر (که به صراحت آدرس «عامل خارجی» در آن درج شده) مستقیماً به عامل خارجی بفرستد. (مرحله ۳) از آن به بعد بدون در نظر گرفتن موقعیت دائمی ماشین متحرک، بسته‌ها مستقیماً به سوی عامل خارجی هدایت شده و تحویل ماشین متحرک می‌شود.^۱

روشهایی که تاکنون معرفی شده‌اند از چندین جهت با هم تفاوت دارند: اولین مورد آن که چه مقدار از وظایف

۱. در فرآیند تونل یک بسته کامل در درون یک بسته دیگر جاسازی می‌شود: آدرس بسته بیرونی مربوط به «عامل خارجی» است و آدرس بسته درونی هویت صاحب اصلی بسته یعنی ماشین متحرک را مشخص می‌کند. - م



شکل ۵-۱۹. مسیرابی بسته ها پسوی ماشینهای متحرک.

این پروتکل توسط مسیر یابها و چه مقدار توسط ماشینهای میزبان انجام می شود و آن بخش از وظایف که در ماشین میزبان باید انجام شود در چه لایه ای تعریف شده است. دوم آن که در برخی از روشها، مسیر یابهای واقع بر روی مسیر قادرند «نگاشتهای آدرس» [یعنی تغییر آدرس ماشین متحرک به آدرس جدید] را ثبت کرده و بدون هیچ دخالت بیرونی در میانه راه جلوی ترافیک ارسالی به آدرس قبلی را گرفته و آن را به سمت آدرس جدید تغییر مسیر بدهند.

سوم آن که در برخی دیگر از این روشها به هر ماشین متحرک که میهمان شبکه ای جدید شده یک آدرس موقت ولیکن منحصر به فرد داده می شود در حالی که در برخی دیگر این آدرس موقت مربوط به آن عامل خارجی می باشد که هدایت ترافیک تمام میهمانان خود را بر عهده گرفته است.

تفاوت چهارم این روشها آنست که وقتی بسته ها برای رسیدن به یک ماشین دیگر آدرس دهی شده اند چگونه تحویل ماشین دیگری می شوند. [بدین معنا که بسته هایی که به سوی آدرس قبلی یک ماشین روانه شده اند به چه نحو برای هدایت به جایی دیگر تغییر آدرس می دهند. -م] گزینه اول آنست که در هر بسته، آدرس مقصد عوض شود و بسته تغییر یافته از نو ارسال شود. گزینه دوم آنست که کل بسته به همراه آدرس دائم (خانگی) و با کلیه مشخصات [به صورت دست نخورده]، در درون فیلد داده از یک بسته دیگر جاسازی (کپسوله) شود و بسته جدید به آدرس موقت ماشین ارسال گردد.

تفاوت آخر آنکه روشهای مختلف از دیدگاه تمهیدات امنیتی با یکدیگر فرق می کنند. عموماً وقتی یک ماشین میزبان یا مسیر یاب، پیامی با مضمون این مثال دریافت می کند: «لطفاً از هم اکنون به بعد تمام نامه های استفانی را برای من بفرست» دو سؤال مطرح می شود: این پیام واقعاً متعلق به کیست (با چه کسی صحبت می شود) و آیا چنین کاری اصولاً به صلاح است؟ در مراجع زیر چندین پروتکل در خصوص ماشینهای متحرک تشریح و مقایسه شده اند:

(Hac and Guo, 2000; Perkins, 1998a; Snoeren and Balakrishnan, 2000; Solomon, 1998; Wang and Chen, 2001)

۱۰-۲-۵ مسیریابی در شبکه‌های ویژه (Routing in Ad Hoc Networks)

تاکنون روشهای مسیریابی در محیطهایی را که ماشینهای میزبان متحرکند ولی مسیریابها ثابت هستند، بررسی کرده‌ایم. یک حالت استثنایی آن است که مسیریابها نیز خودشان متحرک باشند! از چنین محیطهایی می‌توان به موارد زیر اشاره کرد:

۱. وسایل نقلیه نظامی در صحنه نبرد بدون دسترسی به هیچگونه زیرساخت ارتباطی
۲. ناوگان دریایی بر روی دریا
۳. نیروی امداد در شرایط اضطراری مثل زلزله که کلیه زیرساختها را نابود کرده است
۴. گروههایی افراد با کامپیوتر کیفی، در ناحیه‌ای بدون شبکه بی‌سیم 802.11

در تمام این موارد (و نظایر آن) هر گره شامل یک مسیریاب و یک ماشین میزبان است که اغلب هر دوی آنها بر روی یک ماشین قرار می‌گیرند. شبکه‌ای از این گره‌ها که در کنار هم قرار می‌گیرند اصطلاحاً «شبکه‌های ویژه» یا MANET^۱ نامیده می‌شود. اجازه بدهید مختصراً این شبکه‌ها را بررسی نماییم. برای آگاهی بیشتر می‌توان به (Perkins, 2001) مراجعه کرد.

آنچه که «شبکه‌های ویژه» را از «شبکه‌های سیمی» متمایز می‌کند آن است که تمام قواعد طبیعی در خصوص توپولوژی ثابت، همسایه‌های شناخته شده ثابت، تناظر دائم بین موقعیت فیزیکی و آدرس IP ماشینها و مواردی از این قبیل در خصوص «شبکه‌های ویژه» صادق نیست. مسیریابها در رفت و آمد هستند و ممکن است در هر لحظه از محل جدیدی سر در آورند! در یک شبکه سیمی هر گاه یک مسیریاب، مسیری معتبر به برخی از نقاط مقصد در شبکه داشته باشد آن مسیر تا ابد معتبر خواهد بود (مگر آن که در جایی از سیستم خرابی بوجود بیاید). در یک شبکه ویژه، توپولوژی شبکه به طور دائم در تغییر است لذا اعتبار مسیرها و بهینگی آنها به ناگاه و بی هیچ هشدار قبلی تغییر می‌کند. آشکار است که با چنین وضعیتی، مسیریابی در شبکه‌های ویژه کاملاً متفاوت از شبکه‌های ثابت می‌باشد.

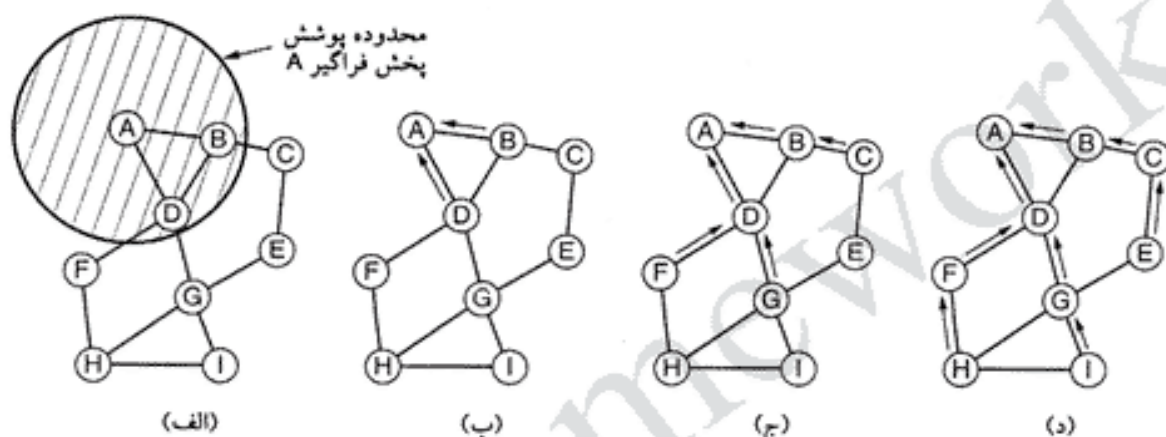
گونه‌های متعددی از الگوریتمهای مسیریابی برای شبکه‌های ویژه پیشنهاد شده است. یکی از جالبترین آنها الگوریتم مسیریابی AODV^۲ است. (Perkins and Royer, 1999) این الگوریتم گونه‌ای از «الگوریتم بردار فاصله بلمن-فورد» محسوب می‌شود که برای کار در محیطهای متحرک تطبیق داده شده و در آن پهنای باند محدود و عمر کم باتری ماشینها در این محیط [در محاسبات مربوط به مسیرهای بهینه] در نظر گرفته شده است. یکی دیگر از ویژگیهای نامتعارف این روش آن است که الگوریتم «برحسب تقاضا» (On-Demand) عمل می‌کند بدین معنا که مسیر رسیدن به برخی از نقاط مقصد، فقط وقتی تعیین می‌شود که کسی بخواهد بسته‌ای را بدان مقصد بفرستد. اجازه بدهید ببینیم قضیه از چه قرار است.

کشف مسیر

یک شبکه ویژه را در هر لحظه از زمان می‌توان با گرافی از گره‌ها توصیف کرد. (گره‌ها عبارتند از مسیریابها + ماشینهای میزبان) اگر دو ماشین بتوانند از طریق سیستم رادیویی خود مستقیماً با یکدیگر ارتباط برقرار کنند می‌گوییم این دو گره (Node) بهم متصل هستند. (یعنی در گراف شبکه، بین این دو گره یک کمان وجود دارد). از آنجایی که امکان دارد یکی از این دو ماشین، فرستنده پرقدرت‌تری نسبت به دیگری داشته باشد لذا این امکان وجود دارد که ارتباط A به B برقرار باشد ولی ارتباط B با A میسر نباشد؛ ولیکن برای سادگی فرض را بر آن می‌گذاریم که تمام ارتباطات، دو طرفه و متقارن هستند. همچنین باید بدین نکته اشاره کرد که هر گاه دو گره در برد

رادیویی یکدیگر باشند تضمینی وجود ندارد که ارتباط آنها برقرار باشد؛ ممکن است بین آنها ساختمان، تپه یا موانع دیگری وجود داشته باشد که جلوی ارتباط آنها را بگیرد. [هر چند در برد یکدیگر واقعند].

برای توصیف الگوریتم، شبکه ویزه شکل ۵-۲۰ را در نظر بگیرید که در آن یک پروسه در گره A می خواهد بسته ای را برای گره I بفرستد. در الگوریتم AODV، هر گره دارای جدولی است که کلید این جدول، آدرس مقصد است^۱ و هر یک از رکوردهای این جدول، اطلاعاتی در خصوص مقصد و آنکه برای رساندن بسته ای به آن مقصد باید بسته را به کدامیک از همسایه های آن فرستاد، در خود نگاهداری می کنند. فرض کنید که A در جدول خود جستجو کرده و هیچ درایه ای متناظر با I در آن نمی یابد. حال بایستی مسیری به I کشف کند. همین ویژگی که مسیرها فقط در هنگام لزوم کشف می شوند به الگوریتم، ویژگی On-Demand یعنی «برحسب تقاضا» داده است.



شکل ۵-۲۰. (الف) محدوده پوشش پخش فراگیر A (ب) پس از آنکه B و D پخش فراگیر A را دریافت کردند. (ج) پس از آنکه C و F و G پخش فراگیر A را دریافت کردند. (د) پس از آنکه E و H و I پخش فراگیر A را دریافت کردند. گره های سایه دار دریافت کنندگان جدید هر مرحله محسوب می شوند. فلشها مسیر معکوس (مسیر برگشت) را مشخص می کنند.

برای پیدا کردن موقعیت I، گره A یک بسته خاص به نام ROUTE REQUEST (تقاضای مسیر) ساخته و آن را به صورت فراگیر منتشر می کند. به گونه ای که در شکل ۵-۲۰ الف مشهود است، این بسته به B و D می رسد. در حقیقت دلیل آنکه در این گراف، B و D به A متصل شده اند آنست که قادرند سیگنال مخابراتی A را دریافت کنند. به عنوان مثال بین گره A و F در گراف هیچ کماتی ترسیم نشده است چرا که F نمی تواند سیگنال رادیویی A را دریافت کند. بنابراین ارتباط مستقیمی بین A و F وجود ندارد.

قالب بسته ROUTE REQUEST در شکل ۵-۲۱ نشان داده شده است. این بسته شامل «آدرس مبدا» و «آدرس مقصد» است و مشخص می کند که چه کسی در جستجوی چه کسی است. (عموماً از آدرسهای IP آنها استفاده می شود) این بسته همچنین حاوی یک «شناسه تقاضا» (Request ID) است. این شناسه در حقیقت یک شمارنده محلی است که در هر گره وجود دارد و هرگاه که یک بسته ROUTE REQUEST منتشر گردید یک واحد به آن اضافه می شود. ترکیب «آدرس مبدا» و فیلد «شناسه تقاضا»، هویت بسته ROUTE REQUEST را به صورت یکتا و منحصر بفرد تعیین کرده و بدین ترتیب گره ها قادرند بسته هایی که احتمالاً به صورت تکراری دریافت می شوند را تشخیص داده و حذف کنند.

هر گره به غیر از «شمارنده شناسه تقاضا» یک شمارنده دیگر هم دارد که هرگاه بسته ROUTE REQUEST

۱. یعنی جستجوی رکوردها در این جدول براساس آدرس مقصد انجام می شود. -م

Source address	Request ID	Destination address	Source sequence #	Dest. sequence #	Hop count
----------------	------------	---------------------	-------------------	------------------	-----------

شکل ۵-۲۱. قالب بسته ROUTE REQUEST.

ارسال شود (یا پاسخی برای بسته ROUTE REQUEST دیگران دریافت گردد) یک واحد بدان اضافه می‌گردد. عملکرد این شمارنده تا حدودی شبیه به یک ساعت است و کاربرد آن تشخیص مسیر جدید از مسیر قدیمی است. [یعنی وقتی دو بسته یکسان در مورد مشخصات یک مسیر دریافت می‌شود این شمارنده مشخص می‌کند که کدام جدیدتر از دیگری است] فیلد چهارم از شکل ۵-۲۱ مقدار همین شمارنده را برای گره مبدا (که در این مثال A است) مشخص می‌کند. [در این مثال A در جستجوی I است لذا در بسته ROUTE REQUEST شماره ترتیب تقاضای خود را درج کرده است.] فیلد پنجم مشخص کننده آخرین و جدیدترین شماره بسته‌ای است که A در مورد I دریافت کرده است. (و صفر است اگر تاکنون در مورد I چیزی دریافت نکرده باشد.) آخرین فیلد یعنی «شمارنده گام» (Hop Counter) مشخص می‌کند که بسته تاکنون چند گام را طی کرده است. مقدار اولیه این فیلد صفر است و به ازای عبور از هر گره یک واحد به آن اضافه می‌شود.

وقتی بسته ROUTE REQUEST به یک گره می‌رسد (در این مثال به B و D) طبق مراحل زیر پردازش می‌شود:

۱. ابتدا جفت مشخصه (آدرس مبدا، شناسه تقاضا) در یک جدول محلی (که سوابق دریافت چنین بسته‌هایی را نگهداری می‌کند) جستجو می‌شود تا مشخص گردد که آیا این بسته قبلاً نیز دریافت و پردازش شده است؟ اگر تکراری بود بسته حذف شده و پردازش آن در همین جا خاتمه می‌یابد. اگر تکراری نبود این زوج مشخصه بسته را در جدول سوابق وارد می‌کند تا در آینده نیز بسته‌ای مشابه با این بسته پردازش نشود. سپس فرآیند پردازش ادامه می‌یابد.

۲. سپس گیرنده در جدول مسیریابی خود، آدرس مقصد را جستجو می‌کند. اگر یک مسیر جدید و «تازه» به این مقصد پیدا شود یک بسته ROUTE RERLY برای مبدا برگردانده می‌شود تا مبدا نیز از چگونگی رسیدن به مقصد مورد نظر آگاه شود. (اغلب، پاسخ بدین نحو است: از طریق من عمل کن!) «تازه» بودن مسیر بدین معنی است که فیلد «شماره ترتیب مقصد» ذخیره شده در جدول مسیریابی باید بزرگتر یا مساوی همین فیلد در بسته ROUTE REQUEST باشد. اگر کمتر بود بدین معنی تلقی می‌شود که مشخصات ذخیره شده در حافظه، قدیمی‌تر از مسیری است که خود مبدا برای رسیدن به مقصد در اختیار دارد^۱، در این صورت مرحله سوم به اجرا در می‌آید.

۳. از آنجایی که گیرنده هیچ مسیر جدیدی بدان مقصد نمی‌شناسد، به مقدار فیلد «شمارنده گام» (Hop Count) یک واحد اضافه کرده و بسته ROUTE REQUEST را مجدداً پیرامون خود منتشر می‌نماید. البته داده‌های درون بسته را استخراج کرده و آن را به عنوان یک درایه جدید در جدول «مسیرهای معکوس» ذخیره می‌کند.^۲ این اطلاعات بدان جهت مفید است که می‌توان مسیرهای معکوس ایجاد کرد و از

۱. کاربرد فیلد شماره ترتیب مقصد (Destination Sequence #) را در ذهن خود اینگونه فرض کنید که مثلاً A به همسایه‌های خود اعلام می‌کند که من خودم اطلاعاتی در خصوص مسیر رسیدن به I دارم که شماره ترتیب آن (مثلاً) Destination Sequence # = ۱۲۳۴ است. اگر شما اطلاعات جدیدتری دارید لطفاً برای من بفرستید! -م

۲. یعنی اگرچه مبدا به دنبال یافتن یک مقصد خاص است ولیکن در بسته تقاضا حداقل خود را معرفی کرده است فلذا مسیرهای گیرنده این تقاضا، لااقل می‌توانند از حضور این مبدا و مسیر رسیدن به آن کسب آگاهی کنند؛ به چنین مسیری «مسیر معکوس» گفته می‌شود. -م

طریق آن، در آینده پاسخ این تقاضا را به مبدا برگرداند. فلشهایی که بر روی شکل ۵-۲۰ دیده می شوند چگونگی ساخته شدن مسیرهای معکوس را نشان می دهند. به محض ایجاد یک مسیر معکوس و جدید، یک زمان سنج (تایمر) برای آن تنظیم می شود. اگر مهلت این زمان سنج منقضی شود و پاسخی به بسته ROUTE REQUEST برنگردد، مسیر ایجاد شده حذف خواهد شد.

هیچیک از گره های B و D نمی دانند که I کجاست لذا این دو نیز ضمن ساختن مسیر معکوس جهت بازگشت پاسخ به A، با تغییر فیلد Hop Count به ۱، آنرا مجدداً منتشر می کنند. (به شکل ۵-۲۰ دقت کنید). انتشار مجدد بسته توسط B، به C و D می رسد. C نیز درایه ای در جدول مسیرهای معکوس خود ایجاد و آن را از نو منتشر می کند. در مقابل، D آنرا به عنوان بسته ای تکراری حذف می کند. [چون قبلاً از طریق A دریافت کرده است.] به همین طریق بسته منتشره توسط D در B تکراری تشخیص داده شده و حذف می گردد. ولیکن به گونه ای که در شکل ۵-۲۰-ج دیده می شود بسته منتشره توسط D در F و G پذیرفته و ذخیره می شود.

پس از آن که E و H و I نیز بسته منتشر شده را دریافت کردند، عاقبت پیغام ROUTE REQUEST به مقصد خود می رسد (یعنی به خود I یا به گره ای که از موقعیت I خبر دقیق دارد می رسد). (شکل ۵-۲۰-د را ببینید). اگرچه ما کل فرآیند انتشار را در سه مرحله جدا نشان داده ایم ولیکن انتشار بسته ها از گره های مختلف به صورت هماهنگ و همزمان انجام نمی شود.

گره I در پاسخ به تقاضای ورودی، یک بسته ROUTE REPLY مطابق با شکل ۵-۲۲ ایجاد می کند. فیلدهای «آدرس مبدا»، «آدرس مقصد» مستقیماً از بسته تقاضا استخراج و در بسته پاسخ کپی می شوند ولیکن «شماره ترتیب مقصد» (Destination Sequence Number) برگرفته از مقدار شمارنده ای است که درون حافظه I قرار دارد. فیلد «شمارنده گام» نیز به صفر تنظیم می شود. فیلد «طول عمر» (Lifetime) مشخص می کند که مشخصات مسیر اعلام شده تا چه زمانی معتبر است. این بسته صرفاً برای گره ای ارسال می شود که بسته تقاضا (یعنی ROUTE REQUEST) از طریق او دریافت شده است (در این مثال گره G). این بسته، مسیر معکوس خود به D و نهایتاً به A را طی می کند. در هر گره به مقدار فیلد شمارنده پیام یک واحد اضافه می شود تا هر گره که آن را می بیند بفهمد که فاصله اش تا گره مقصد (در این مثال I) چقدر است.

Source address	Destination address	Destination sequence #	Hop count	Lifetime
----------------	---------------------	------------------------	-----------	----------

شکل ۵-۲۲. قالب بسته ROUTE REPLY.

هر گره میانی این بسته را در راه بازگشت، بررسی می کند. اگر یکی از سه شرط زیر برقرار باشد، اطلاعاتی در خصوص مسب رسیدن به I، در جدول مسیریابی هر گره میانی ذخیره خواهد شد:

۱. اگر هیچ مسیر شناخته شده ای به I نداشته باشد.
۲. اگر شماره ترتیب I (یعنی شماره ای که I در فیلد Destination Seq.No. گذاشته است) بزرگتر از شماره ای باشد که در جدول مسیریابی درج شده است.
۳. اگر شماره ترتیب یکسان باشد ولی مسیر جدید کوتاه تر باشد. [کوتاه بودن از فیلد شمارنده گام مشخص می شود.]

در این روش تمام گره هایی که بر روی مسیر معکوس قرار دارند، مجاناً از مسیر رسیدن به I آگاه می شوند (یکی از محاسن جانی کشف مسیر توسط A، آگاهی گره های میانی است). گره هایی که بسته ROUTE REQUEST را

در مسیر رفت دریافت کرده‌اند ولیکن در مسیر معکوس نیستند (در این مثال B و C و E و F و H) پس از متقاضی شدن زمان سنج (تایمر)، مسیر معکوس به A را حذف می‌کنند.

در شبکه‌های عظیم، این الگوریتم بسته‌های فراگیر بسیار زیادی را تولید می‌کند، حتی اگر گره مقصد در نزدیکی مبدا باشد. برای کاهش تولید بسته‌ها در فرآیند انتشار، می‌توان بدین نحو عمل کرد: فیلد TTL (Time To Live) در بسته IP، توسط فرستنده آن به مقداری نزدیک به «قطر»^۱ شبکه تنظیم شده و به ازای عبور از هر گره، یک واحد از آن کسر گردد؛ هر گاه به صفر رسید، بسته به جای انتشار مجدد حذف شود. فرآیند کشف مسیر را می‌توان بدین نحو اصلاح کرد: برای یافتن موقعیت مقصد، فرستنده، بسته ROUTE REQUEST را با تنظیم فیلد TTL به ۱ ارسال می‌کند. اگر در یک مدت زمان معقول پاسخی برنگشت بسته بعدی را با TTL معادل ۲ ارسال می‌نماید. همین روال با مقادیر ۳ و ۴ و ۵ و... ادامه می‌یابد تا بالاخره پاسخی برگردد. در این روش، جستجو ابتدا به صورت محلی و در پیرامون گره شروع شده و در هر مرحله محدوده جستجو گسترده‌تر می‌شود.

نگهداری مسیر (Route Maintenance)

از آنجایی که گره‌ها می‌توانند جابجا شده یا کلاً خاموش شوند لذا توپولوژی شبکه گاه به گاه تغییر می‌کند. به عنوان مثال در شکل ۵-۱۲ اگر G به ناگاه خاموش شود، A نخواهد فهمید مسیری که برای رسیدن به I در اختیار داشته (یعنی مسیر ADGI) دیگر برقرار نیست. این الگوریتم باید بتواند به نحوی این مسئله را حل و فصل کند. هر گره در شبکه بطور متناوب «پیغام سلام» (Hello Message) منتشر می‌کند. انتظار می‌رود که همسایه‌ها به این پیام پاسخ بدهند. اگر پاسخی باز نگشت، متشکرکننده پیام آگاه می‌شود که همسایه او از بُردش خارج شده و دیگر ارتباط آنها برقرار نیست. به روش مشابه اگر بسته‌ای معمولی برای همسایه خود بفرستد و پاسخی نگیرد متوجه می‌شود که آن همسایه در دسترس نیست.

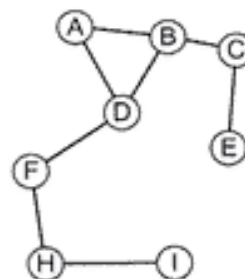
از این اطلاعات می‌توان برای پاک کردن مسیرهایی که دیگر کار نمی‌کنند بهره گرفت. هر گره مثل N برای یکایک گره‌های مقصد که همسایه‌هایش در خلال ΔT ثانیه گذشته بسته‌ای را از طریق او بدان مقصد ارسال کرده‌اند، فهرستی را نگه می‌دارد. این فهرست اصطلاحاً «همسایه‌های فعال N» نامیده می‌شوند. بدین منظور گره N دارای جدولی است که کلید آن، آدرس مقصد گره‌های شبکه است؛ همچنین در این فهرست، گره بعدی برای رسیدن بدان مقصد، تعداد گام برای رسیدن به آن مقصد، «آخرین شماره ترتیب» و «فهرست همسایه‌های فعال» درج شده است. به عنوان نمونه جدول مسیریابی گره D برای توپولوژی مثال قبلی، چیزی شبیه به شکل ۵-۲۳-الف است.

وقتی یکی از همسایه‌های N از دسترس خارج می‌شود، گره N جدول مسیریابی خود را بررسی می‌کند تا ببیند کدامیک از گره‌های شبکه در مسیرهای خود از گره حذف شده استفاده می‌کرده‌اند. این موضوع به همسایه‌های فعال اطلاع داده می‌شود که: تمام مسیرهای آنها از طریق N نامعتبر است (چرا که N بسته‌های آنها از طریق گره حذف شده ارسال می‌شده است)؛ همسایه‌های فعال نیز به همسایه‌های فعال خود خبر می‌دهند و مکرراً این کار انجام می‌شود تا تمام مسیرهایی که وابسته به گره تازه از دست رفته بوده‌اند از کل جداول مسیریابی حذف شوند. به عنوان مثالی از روش «نگهداری مسیر»، مثال قبلیمان را مدنظر قرار بدهید ولیکن فرض کنید که G به ناگاه خاموش شده است. توپولوژی تغییر یافته شبکه در شکل ۵-۲۳-ب نشان داده شده است؛ وقتی D متوجه می‌شود که G از میان رفته است به جدول مسیریابی خود مراجعه می‌کند و می‌بیند که G بر روی مسیری قرار داشته که به E،

۱. قطر شبکه به معنای طول بزرگترین مسیر در شبکه است. -م

Dest.	Next hop	Distance	Active neighbors	Other fields
A	A	1	F, G	
B	B	1	F, G	
C	B	2	F	
E	G	2		
F	F	1	A, B	
G	G	1	A, B	
H	F	2	A, B	
I	G	2	A, B	

(الف)



(ب)

شکل ۵-۲۳. (الف) جدول مسیریابی D قبل از آنکه G از کار بیفتد. (ب) گراف پس از حذف G از شبکه.

G و I ختم می شده است. اجتماع مجموعه همسایه های فعال این سه گره مقصد عبارت است از {A و B}. به عبارت دیگر A و B در برخی از مسیرهای خود به G متکی بوده اند لذا باید به این مسیریابها اطلاع داد که G دیگر کار نمی کند. D با ارسال بسته های خاصی این موضوع را بدانها اطلاع داده تا آنها نیز جدول مسیریابی خود را اصلاح نمایند. خود D نیز درایه های متناظر با E، G و I را از جدول مسیریابی خود حذف می کند.^۱

شاید از توضیحاتی که در خصوص الگوریتم AODV ارائه کردیم مشخص نشده باشد که بنیانی ترین تفاوت بین این الگوریتم و الگوریتم بلمن-فورده آن است که گره ها بطور متناوب کل جدول مسیریابی خود را منتشر نمی کنند، بلکه فقط بخش کوچکی از آنرا و آنهم بر حسب تقاضا، ارسال می نمایند. این ویژگی، در پهنای باند مصرفی و طول عمر باتری گره های متحرک صرفه جویی می کند.

AODV همچنین قادر به پخش داده های فراگیر (Broadcast) و مسیریابی چندپخشی (Multicast) است. برای آگاهی بیشتر به مرجع (Perkins and Royer, 2001) مراجعه نمایید. مسیریابی در شبکه های ویژه، یک موضوع پژوهشی داغ و جدید است و اخیراً در این خصوص مقالاتی تألیف و منتشر شده است. برخی از آنها را می توان در مراجع ذیل یافت:

Chen et al., 2002; Hu and Johnson, 2001; Li et al., 2001; Raju and Garcia-Luna-Aceves, 2001; Ramanathan and Redi, 2002; Royer and Toh, 1999; Spohn and Garcia-Luna-Aceves, 2001; Tseng et al., 2001; and Zadeh et al., 2002).

۱۱-۲-۵ جستجوی گره در شبکه های همتا به همتا (Peer-to-Peer)

یکی از پدیده های نسبتاً جدید، شبکه های همتا به همتا است که در آن تعداد کثیری از افراد برای به اشتراک گذاشتن منابع مشترک خود، مستقیماً با یکدیگر در تماسند (این افراد عموماً از طریق یک اتصال ثابت و پسیو به اینترنت متصل شده اند). اولین کاربرد بسیار گسترده از تکنولوژی «همتا به همتا» به یکی از بحث برانگیزترین ماجراهای «گناه جمعی» بدل شد: ۵۰ میلیون از کاربران Napster می توانستند فایل های موزیک و آواز را که حق امتیاز آنها در

۱. توضیحی برای رفع ابهام از مفهوم همسایه های فعال خالی از لطف نیست. به سطر آخر از جدول ۵-۲۳-ب دقت کنید. این جدول فرضاً متعلق به D است و آخرین رکورد آن بیان می کند که برای رسیدن به گره I باید بسته ها به G فرستاده شوند و تا رسیدن به I فقط دو گام باقی مانده است؛ در فیلد «همسایه های فعال» نام A و B درج شده است یعنی: A و B برای ارسال بسته هایشان به سوی I آنها را به من -D- می دهند. لذا در صورت خرابی G باید به این دو همسایه اطلاع داد تا آنها نیز از این موضوع آگاه شده و جداول خود را اصلاح کنند. -م

تملک دیگران بود بدون اجازه از صاحبین آنها با یکدیگر رد و بدل کنند؛ نهایتاً Napster پس از مناقشات فراوان به حکم دادگاه تعطیل شد.^۱ بهر تقدیر، تکنولوژی همنا به همنا کاربردهای قانونی بسیار جالبی دارد. این تکنولوژی با مسائل و مشکلات مسیریابی مواجه است؛ ولیکن این مسائل با آنچه که تاکنون مطالعه کرده ایم کاملاً یکسان نیست، لذا مروری اجمالی بر آن خالی از لطف نخواهد بود.

آنچه که سیستمهای «همنا به همنا» را جذاب کرده آنست که سیستمی کاملاً توزیع شده ایجاد می کند یعنی تمام گره ها متقارن هستند و هیچگونه کنترل مرکزی یا سلسله مراتب خاصی بر آن حاکم نیست. در یک سیستم رایج همنا به همنا، هر یک از کاربران دارای مقداری اطلاعات هستند که ممکن است برای کاربران دیگر جالب باشد. این اطلاعات می تواند نرم افزارهای رایگان و عمومی، موزیک، عکس و نظایر آنها باشد. هر گاه تعداد کاربران زیاد باشد، یکدیگر را نمی شناسند و نمی دانند آنچه را که در جستجوی آن هستند در کجا پیدا کنند. یک راه حل آن است که یک پایگاه داده مرکزی و بسیار بزرگ از فهرست آنها داشته باشیم ولی ممکن است به دلایلی امکان پذیر نباشد (مثلاً هیچکس حاضر به میزبانی و نگهداری آن نشود) بنابراین مسئله اساسی آن است که در غیاب یک پایگاه مرکزی یا یک فهرست مرکزی، یک کاربر چگونه می تواند گره ای را که اطلاعات مورد نیاز او را در اختیار دارد، پیدا کند.

اجازه بدهید فرض را بر آن بگذاریم که هر کاربر یک یا چند آیتم مثل موزیک، عکس، برنامه، فایل و نظائر آنها در اختیار دارد و کاربران دیگر علاقمند به خواندن آنها هستند. هر آیتم توسط یک رشته ASCII نامگذاری شده است. کاربران احتمالی فقط همین رشته ASCII را می دانند و می خواهند بدانند آیا فردی یا افرادی آن را در اختیار دارند و اگر دارند آدرس IP آنان چیست؟

به عنوان مثال به پایگاه توزیع شده مردم شناسی (شجره نامه مردم) دقت کنید. هر کسی که به مردم شناسی علاقمند است فرضاً در خصوص اجداد و بستگان خود اطلاعاتی مثل عکس، صدا یا حتی قطعات ویدیویی در اختیار دارد. ممکن است جد اعلای بسیاری از افراد یکی باشد و امکان دارد اطلاعات مربوط به آن در چندین «گره» وجود داشته باشد. نام هر رکورد عموماً نام فرد مورد نظر (در شکل و قالب مشخص) است. حال در جایی یک نفر که علاقمند به تحقیق در مورد شجره نامه خود بوده متوجه می شود که نام جد اعلای او در آرشیو یک گره خاص وجود دارد و ساعت جیبی و طلائی خود را برای برادرزاده اش به ارث گذاشته است! حال او می خواهد بداند که نام این برادرزاده چیست و آیا در جایی دیگر رکوردی در این رابطه وجود دارد. بدون وجود یک پایگاه اطلاعات مرکزی، چگونه می توان کسی که چنین رکوردی را در اختیار دارد، پیدا کرد؟

برای حل این مسئله الگوریتمهای گوناگونی پیشنهاد شده است. الگوریتمی که بررسی می کنیم «الگوریتم Chord» نام دارد. (Dabek et al., 2001; Stoica et al., 2001) شرح ساده عملکرد آن بدینگونه است: سیستم Chord از مجموعه n کاربر شرکت کننده تشکیل شده است. هر یک از این کاربران ممکن است رکوردهایی را در خود ذخیره کرده و همچنین آمادگی داشته باشند بخشی از فهرست (ایندکس) سیستم را برای استفاده دیگر کاربران نگهداری و عرضه کنند. هر یک از کاربران دارای یک آدرس IP هستند که این آدرس را می توان توسط یک «تابع درهم سازی» (Hash Function) به یک آدرس m بیتی تبدیل کرد. در سیستم Chord از تابع درهم سازی SHA-1 استفاده می شود؛ SHA-1 در مبحث رمزنگاری کاربرد دارد و در فصل هشتم نگاهی بدان خواهیم انداخت. فعلاً به همین اندازه بسنده می کنیم که این تابع یک رشته از بایتهای طول دلخواه را گرفته و براساس آن یک عدد ۱۶۰ بیتی

۱. در حقیقت کاری که Napster انجام می داد این بود که مشترکین خود را که فایل های موزیک بر روی کامپیوتر خود داشتند بهم معرفی می کرد تا بتوانند به صورت بی واسطه و رو در رو (یعنی همان همنا به همنا) فایل های خود را رد و بدل کنند. اینکار خشم دست اندرکاران صنعت موسیقی را برانگیخت! -م-

بشدت تصادفی تولید می کند. بنابراین می توانیم یک آدرس IP را به عددی ۱۶۰ بیتی و یکتا تبدیل کنیم که به این عدد اصطلاحاً «شناسه گره» (Node Identifier) گفته می شود.

از دیدگاه مفهومی تجسم کنید که تمام 2^{160} شماره شناسه گره ها به صورت صعودی پیرامون یک دایره بزرگ چیده شده اند. برخی از این شماره ها مربوط به گره های شرکت کننده (کاربران فعال) هستند ولی بیشتر آنها پوچند. در شکل ۵-۲۴ الف ما دایره شماره ها را برای $m=5$ نشان داده ایم. (فعلاً به کمانهای میانی کاری نداشته باشید). $m=5$ بدین معناست که شماره ها پنج بیتی هستند و شماره های از ۰ تا ۳۱ را در بر می گیرند. در این مثال گره های ۱، ۴، ۷، ۱۲، ۱۵، ۲۰ و ۲۷ مربوط به گره های واقعی بوده و در شکل به صورت سایه دار نشان داده شده اند. مابقی شماره ها وجود خارجی ندارند. (شماره های پوچ)

تابع $successor(k)$ را بدین مضمون تعریف می کنیم که اولین «شناسه گره» مربوط به اولین کاربر واقعی که پس از عدد k قرار گرفته را بر می گرداند. به عنوان مثال $successor(6)=7$ ، $successor(8)=12$ و $successor(22)=27$. اسامی رکوردها (مثل نام فایل های آواز و نظائر آن) نیز توسط تابع درهم سازی (یعنی SHA-1) به یک رشته ۱۶۰ بیتی تبدیل می شوند که این رشته «کلید» (Key) نامیده می شود. لذا برای تبدیل یک نام (یعنی نام ASCII هر رکورد) به «کلید» متناظر با آن، رابطه $Key=hash(name)$ را تعریف کرده ایم. برای این کار فقط کافی است پروسیجر محلی $hash$ را فراخوانی نماییم. اگر کسی که یک رکورد مفید با نام $name$ در اختیار دارد، بخواهد آن را در دسترس عموم قرار بدهد ابتدا یک شاخص دوتایی شامل (IP Address, name) ساخته و برای ذخیره کردن این شاخص بر روی یک گره مناسب، تابع $successor(hash(name))$ را فراخوانی می نماید. اگرچه ممکن است چندین رکورد با همین نام در گره های متفاوت موجود باشد ولیکن شاخص دوتایی آنها در یک گره ذخیره می شود.^۱ بدین طریق، ایندکس رکوردها (شاخص دوتایی هر رکورد) به صورت تصادفی بر روی گره ها توزیع می شوند. برای آن که تحمل خرابی (Fault Tolerance) این سیستم بالا باشد می توان از تعداد p تابع مختلف $hash$ استفاده کرد تا هر شاخص دوتایی در p گره مختلف ذخیره شود ولیکن فعلاً به این موضوع نخواهیم پرداخت.

اگر بعداً کاربری بخواهد آیتمی با نام $name$ را جستجو نماید ابتدا آن را توسط تابع $hash$ درهم سازی می کند تا کلید آن بدست آید. سپس تابع $successor(key)$ را فراخوانی می کند تا آدرس IP گره ای که فهرست شاخصها را ذخیره کرده، پیدا شود. اگرچه اولین مرحله ساده است ولی مرحله دوم چندان ساده نیست. برای آن که بتوان آدرس IP گره ای را که متناظر با یک کلید خاص است پیدا کرد، هر گره باید چندین «ساختمان داده نظارتی»^۲ در خود ذخیره نماید. یکی از این ساختمانهای داده، آدرس IP گره ای است که در دایره «شناسه گره ها»، پس از خود گره قرار گرفته است. به عنوان مثال در شکل ۵-۲۴، گره بعدی ۴، گره ۷ است و گره بعدی ۷، گره ۱۲ است.

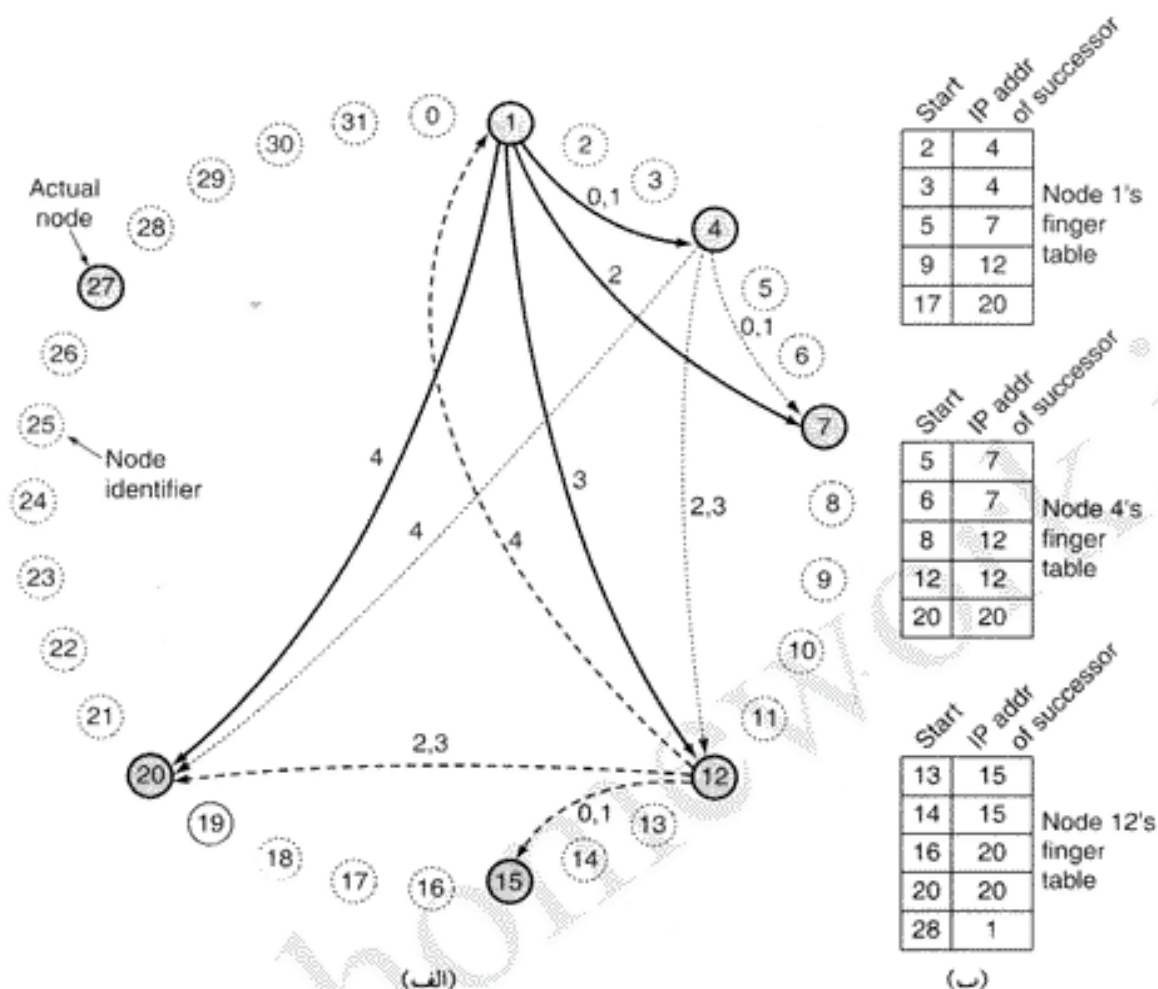
حال مراحل جستجو به ترتیب زیر ادامه می یابد: گره متقاضی بسته ای را برای گره بعدی خود ارسال کرده و ضمن اعلام آدرس IP خود، کلید مورد جستجو را مشخص می نماید. این بسته حول این دایره منتشر می شود تا آن که گره مورد جستجو پیدا شود. هر گره بررسی می کند که آیا اطلاعات مورد جستجو و منطبق با کلید را در اختیار دارد یا خیر. اگر داشته باشد آن اطلاعات را مستقیماً به آدرس IP متقاضی ارسال می نماید و در غیر اینصورت بسته تقاضا را برای بعدی خود در دایره می فرستد.

اولین بهینه سازی این سیستم آن است که هر گره آدرس IP گره قبلی و بعدی خود را داشته باشد تا تقاضاها بتوانند در دو جهت ساعتگرد و پادساعتگرد (بسته به آن که کدام مسیر کوتاهتر است)، ارسال شوند. به عنوان مثال

۱. در حقیقت شناسنامه هر رکورد مثل فایل های صدا، تصویر یا برنامه، که شامل نام آن و آدرس IP ماشین نگهدارنده آنست، بر

۲. Administrative Data Structure

روی ماشینهای دیگری قرار می گیرد. -م



شکل ۵-۲۴. (الف) مجموعه‌ای از ۳۲ شناسه گره که پیرامون یک دایره مرتب شده‌اند. گره‌های خاکستری رنگ متناظر با یک ماشین واقعی هستند. فلشها اشاره گره‌های جدول Finger را نشان می‌دهند. برجسبهای هر فلش اندیسهای جدول Finger هستند. (ب) مثالی از جدول Finger

در شکل ۵-۲۴، گره ۷ می‌تواند برای یافتن گره ۱۰ در جهت ساعتگرد اقدام کند در حالی که برای یافتن گره ۳ می‌تواند به صورت پادساعتگرد عمل نماید.

حتی با وجود دو جهت برای جستجو، در سیستمهای عظیم «همتا به همتا» جستجوی خطی (Linear Search) بسیار ناکارآمد و کند عمل می‌کند چرا که در هر جستجو بطور متوسط باید $n/2$ گره بررسی شود. برای بالا بردن سرعت جستجو، هر گره دارای جدولی است که در سیستم Chord اصطلاحاً Finger Table نامیده شده است. جدول Finger حاوی m درایه (Entry) است که از صفر تا $m-1$ شماره‌گذاری می‌شود و هر یک از آنها به آدرس گره‌های واقعی اشاره می‌کنند. هر یک از این درایه‌ها دارای دو فیلد هستند: فیلد start و آدرس IP گره‌ای که شناسه آن معادل $successor(start)$ است. در شکل ۵-۲۴ ب، جدول Finger برای سه گره نمونه نشان داده شده است. مقادیر فیلدهای مربوط به درایه i در گره k عبارتست از:

$$start = k + 2^i \quad (\text{modulo } 2^m)$$

$$\text{IP address of } successor(start[i])$$

دقت کنید که هر گره، آدرس IP تعداد نسبتاً کمی از گره‌ها را در خود ذخیره می‌کند و اغلب این گره‌ها دارای شماره

شناسه نزدیک بهم هستند. [یعنی در هر گره آدرس IP گره هایی نگهداری می شوند که بر روی دایره شناسه ها، تقریباً در کنار آن گره باشند. -م]

با استفاده از جدول Finger، جستجوی کلید Key در گره k به ترتیب زیر انجام می شود: اگر مقدار key بین k و $successor(k)$ باشد، گره ای که اطلاعاتی در خصوص key در اختیار دارد همان $successor(k)$ است و جستجو خاتمه می یابد. در غیر اینصورت در جدول Finger جستجو می شود تا درایه ای که در آن، مقدار $start$ ، به $predecessor(k)$ [یعنی شناسه گره قبلی k در دایره] نزدیکتر است پیدا شود. این تقاضا مستقیماً به آدرس IP گره فوق الذکر ارسال می شود تا ادامه جستجو در آنجا پیگیری شود. (آدرس IP گره مذکور از جدول Finger بدست می آید.) از آنجایی شناسه این گره نزدیک به key و در عین حال کمتر از آن است لذا این شانس خوب وجود دارد که با تعداد کمی پرس و جوی دیگر نتیجه مورد نظر بدست آید. در حقیقت چون که در هر جستجو فاصله باقیمانده تا هدف مورد نظر نصف می شود می توان نشان داد که متوسط تعداد جستجوها $\log_2 n$ است.

به عنوان اولین مثال فرض کنید در گره ۱، کلید $key=3$ را جستجو می کنیم. از آنجایی که گره ۱ می داند که ۳ بین خودش و گره بعدی او یعنی ۴ واقع شده، لذا گره مورد نظر همان ۴ است و جستجو خاتمه یافته و آدرس IP گره ۴ بدست می آید.

در مثال دوم، فرض کنید در جستجوی کلید $key=14$ بر روی گره ۱ هستیم. چون که ۱۴ بین ۱ و ۴ نیست، از جدول Finger کمک گرفته می شود. نزدیکترین عدد قبل از ۱۴، ۹ است بنابراین، تقاضا به سوی آدرس IP گره حقیقی پس از ۹ که در جدول ۱۲ تعیین شده است هدایت می شود. گره ۱۲ می بیند که گره ۱۴ بین خودش و گره بعدیش یعنی ۱۵ قرار گرفته، لذا آدرس IP گره ۱۵ را برمی گرداند.

به عنوان مثال سوم، فرض کنید بر روی گره ۱ به دنبال کلید $key=16$ هستیم. مجدداً این تقاضا برای گره ۱۲ ارسال می شود ولی در اینجا گره ۱۲ پاسخ این تقاضا را نمی داند و به همین دلیل نزدیکترین شماره قبل از ۱۶ یعنی ۱۴ را یافته و از طریق جدول خود آدرس IP گره ۱۵ را بدست می آورد؛ سپس تقاضا برای آن گره ارسال می شود. گره ۱۵ می بیند که عدد ۱۶ بین خودش و گره بعدی یعنی ۲۰ قرار دارد لذا آدرس IP گره ۲۰ را به سوال کننده بر می گرداند و به همین ترتیب کار ادامه می یابد تا به گره ۱ برسد.

از آنجایی که گره ها بطور متوالی به شبکه می پیوندند یا از آن جدا می شوند فلذا Chord نیازمند روشی است که بتواند این مسئله را نیز حل و فصل کند. فرض را بر آن می گذاریم که وقتی این سیستم آغاز به کار کرده تعداد گره ها آنقدر کم بوده که نتوانسته اند مستقیماً با یکدیگر مبادله اطلاعات کرده و اولین دایره و جداول Finger را بسازند. پس از آن، به یک روال خودکار نیاز است: وقتی گره جدید r می خواهد به این سیستم بپیوندد باید با یکی از گره های موجود تماس برقرار کرده و از او بخواهد که آدرس IP گره $successor(r)$ را برایش پیدا کند. پس از پیدا شدن، گره جدید از $successor(r)$ می خواهد که گره قبلی خود خود را معرفی کند. در آخر، گره جدید از این دو گره می خواهد که او را به عنوان گره r در دایره وارد کند. به عنوان مثال اگر در شکل ۵-۲۴ گره ۲۴ بخواهد به سیستم بپیوندد از یکی از گره های فعال تقاضا می کند که $successor(24)$ را برایش پیدا نماید که در این جا ۲۷ است. سپس از ۲۷ در مورد گره ماقبل او یعنی ۲۰ سوال می کند. پس از آن که حضور خود را به این دو گره (یعنی ۲۰ و ۲۷) اعلام کرد گره ۲۰ را به عنوان گره بعدی خود و گره ۲۷ آن را به عنوان گره قبلی خود به رسمیت می شناسند. مضاف بر این، گره ۲۷ آن دسته از درایه هایی را که کلیدشان در محدوده ۲۱ تا ۲۴ است به گره ۲۴ تقدیم می دارد. در این لحظه گره ۲۴ بطور کامل به جمع پیوسته است. ولیکن در این لحظه بسیاری از جداول Finger غلط هستند. برای اصلاح این جداول، هر گره یک پروسه را که در پس زمینه اجرا می شود، اجرا می کند تا جداول مربوطه (با فراخوانی متناوب تابع $successor$) از نو محاسبه و اصلاح شوند. هر گاه در خلال عملیات تازه سازی، یکی از این

تقاضاها به گره جدیدی بر بخورد درایه مربوط به آن نیز در جدول Finger بهنگام می شود. وقتی یک گره به صورت مسالمت آمیز سیستم را ترک می کند کلیدهای خود را به گره بعدی خود در حلقه تسلیم می کند تا به او اطلاع بدهد که در آستانه خروج از سیستم است و آن گره بتواند پیوند خود را با گره ماقبل از گره در آستانه خروج، برقرار نماید. وقتی گره ای به ناگاه از کار بیفتد مشکل پیش می آید چراکه گره ماقبل از گره خراب شده، هیچ گره معتبر و فعال بعدی ندارد. برای کاهش اثر این مشکل، هر گره نه تنها آدرس مستقیم گره بعدی خود را نگه می دارد بلکه آدرس مستقیم s گره بعد از خود را نیز دارد تا حتی اگر s-1 گره متوالی از کار بیفتد باز هم بتوان دایره را اصلاح کرد.

از سیستم Chord برای ایجاد سیستم توزیع شده فایل (Dabek et al., 2001) و چند کاربرد دیگر استفاده شده و تحقیقات بر روی آن کماکان ادامه دارد. یک سیستم همتابه همتای دیگر به نام Pastry و کاربردهای آن در مرجع (Rowstron and Druschel, 2001) تشریح شده است. سیستم سومی به نام Freenet نیز معرفی شده که مستندات آن در (Clark et al., 2002) در دسترس عموم قرار دارد. چهارمین سیستم از این نوع نیز در مرجع (Ratnasamy, 2001) تشریح شده است.

۳-۵ الگوریتمهای کنترل ازدحام

وقتی به بخشی از زیر شبکه، تعداد بسیار زیادی بسته تحویل شود کارایی آن کاهش می یابد. بدین وضعیت «ازدحام» (Congestion) گفته می شود. شکل ۵-۲۵، علائم بروز این وضعیت را نشان می دهد. هر گاه تعداد بسته هایی که توسط ماشینهای میزبان به زیر شبکه سرازیر می شوند متناسب با ظرفیت حمل زیر شبکه باشد تمام این بسته ها تحویل مقصدشان خواهند شد (به استثنای آنهایی که در اثر خطای انتقال آسیب می بینند) و تعداد بسته های تحویلی متناسب با تعداد بسته های ارسالی است. ولیکن به محض افزایش بی رویه ترافیک، مسیر یابها قادر نیستند از عهده آن بر آمده و بسته ها شروع به از دست رفتن می کنند. این مسئله حادث تر نیز می شود و در ترافیک بسیار بالا کارایی بطور کامل سقوط کرده و هیچ بسته ای تحویل مقصد نخواهد شد!



شکل ۵-۲۵. وقتی ترافیک تحویلی به شبکه بیش از اندازه باشد ازدحام بوجود می آید و کارایی بشدت افت می کند.

چندین عامل می تواند به بروز ازدحام بینجامد. اگر به ناگاه دنباله ای از بسته ها بر روی سه یا چهار خط ورودی دریافت شده و خط خروجی همه آنها یکی باشد صف تشکیل خواهد شد و اگر فضای حافظه کافی برای نگهداری تمام آنها وجود نداشته باشد، بسته ها از بین می روند. شاید اضافه کردن حافظه مفید به نظر برسد ولیکن «ناگل» (Nagle, 1987) بدین نتیجه رسید که حتی اگر حافظه مسیر یاب نامحدود باشد، وضعیت ازدحام نه تنها بهتر

نمی شود بلکه بدتر هم خواهد شد زیرا بسته ها، زمانی به سر صف می رسند که مهلت رسیدنشان به مقصد تمام شده و نسخه های تکراری آن ارسال شده اند. تمام این بسته های تکراری نیز برحسب وظیفه شناسی به مسیر یاب بعدی هدایت شده و در تمام مسیر رسیدن به مقصد، «بار» افزایش می یابد. [بعبارت دیگر، «ازدحام» قابلیت انتشار در کل یک مسیر دارد.]

پردازنده های گنبد نیز می توانند عامل بروز ازدحام باشند. اگر پردازنده اصلی مسیر یاب در انجام وظایف محوله به خود (مثل عملیات صف بندی، بهنگام سازی جداول مسیر یابی و نظائر آن) گنبد عمل نماید، صف ایجاد می شود، حتی وقتی که ظرفیت خطوط خروجی بیش از حد مورد نیاز است.

به دلیل مشابه، خطوط با پهنای باند کم نیز می توانند منجر به بروز ازدحام شوند. ارتقاء پهنای باند خطوط بدون تغییر در پردازنده ها یا بالعکس اغلب فایده چندانی ندارد و فقط جای گلوگاه و منشاء مشکل را تغییر می دهد. همچنین ارتقاء کامل یک بخش کوچک (مثل تغییر یک مسیر یاب و ارتقاء ظرفیت خطوط آن) بدون تغییر در کل سیستم، فقط محل بروز مشکل و گلوگاه را جابجا می کند و در بهبود کارایی کل سیستم تأثیر چشمگیری نخواهد داشت. مشکل اساسی یک سیستم، عدم تطابق و تناسب بخشهای مختلف آنست. این مشکل تا زمانی که کلیه مؤلفه های سیستم متعادل نشوند باقی خواهد ماند.

باید به صراحت تفاوت بین «کنترل ازدحام»^۱ و «کنترل جریان»^۲ و همچنین ارتباط ظریف این دو مکانیزم را تبیین کنیم. «کنترل ازدحام» مکانیزمهایی است جهت ایجاد اطمینان از این که زیر شبکه قادر به حمل ترافیک عرضه شده به آن هست. این مشکل یک مورد همگانی و سراسری است و از رفتار و عملکرد تمام ماشینهای میزبان، کلیه مسیر یابها، عملیات پردازشی «ذخیره و هدایت» (Store & Forward) درون مسیر یابها یا هر عامل دیگری که ظرفیت حمل زیر شبکه را کاهش بدهد، ناشی می شود.

در مقابل، «کنترل جریان» به ترافیک نقطه به نقطه بین یک فرستنده و گیرنده مفروض مربوط می شود و وظیفه اصلی آن ایجاد اطمینان از این موضوع است که یک فرستنده سریع نمی تواند متوالیاً داده ها را با سرعتی بیش از توانایی دریافت گیرنده، ارسال نماید. در کنترل جریان، گزارشات و فیدبکهای مستقیمی از گیرنده به فرستنده ارسال می شود تا به فرستنده طرف مقابل تفهیم کند که کارها را چگونه انجام بدهد.

برای درک اختلاف بین این دو مفهوم، یک شبکه فیبرنوری با ظرفیت 1000 Ggagabits/sec را در نظر بگیرید که در آن یک ابر کامپیوتر سعی می کند با سرعت یک گیگابیت بر ثانیه فایلی را برای یک کامپیوتر شخصی، ارسال کند! اگرچه هیچگونه ازدحامي رخ نخواهد داد (خود شبکه مشکلی ندارد) ولیکن برای آن که بتوان به نحوی ابر کامپیوتر را وادار کرد که هرازگاهی متوقف شود و اجازه نفس کشیدن به کامپیوتر شخصی بدهد به مکانیزمهای کنترل جریان نیاز است.

در سمت مقابل، یک شبکه «ذخیره و هدایت» با خطوط 1Mbps و هزار کامپیوتر بزرگ را در نظر بگیرید که در آن نیمی از کامپیوترها سعی می کنند با سرعت 100Kbps برای نیم دیگر، فایل ارسال کنند! در اینجا مشکل تحت فشار قرار گرفتن گیرنده گنبد توسط فرستنده سریع مطرح نیست بلکه مسئله آنست که ترافیک تحمیل شده به شبکه از میزان ظرفیت و توانایی آن بیشتر است.

دلیل آن که مفهوم کنترل ازدحام، اغلب با مفهوم کنترل جریان اشتباه می شود آن است که در برخی از الگوریتمهای کنترل ازدحام، هنگام بروز مشکل برای شبکه، پیغامهایی را برای ماشینهای مبدا ارسال کرده و به آنها تفهیم می کند که باید سرعت خود را پایین بیاورند. بنابراین یک ماشین میزبان ممکن است به دو دلیل پیغام

Slowdown (آهسته تر ارسال کن) را دریافت کند: اول آن که گیرنده نتواند با همان سرعتی که فرستنده ارسال می کند داده ها را دریافت نماید. دوم آن که شبکه با ازدحام مواجه شود و از عهده ارسال داده ها بر نیاید. بعداً باز هم به این موضوع بر می گردیم.

مطالعات خود پیرامون کنترل ازدحام را با نگاهی به مدل و روشهای عمومی برخورد با آن آغاز می کنیم. سپس به راهکارهای گسترده ای خواهیم پرداخت که سعی می کنند در همان ابتدا از بروز این مشکل پیشگیری کنند. همچنین الگوریتمهای پویایی را مرور می کنیم که پس از بروز مشکل ازدحام، سعی در حل و فصل آن می کنند.

۵-۱۳ اصول کلی در کنترل جریان

بسیاری از مشکلات و مسائل سیستمهای پیچیده مثل شبکه های کامپیوتری را می توان از دیدگاه نظریه کنترل بررسی کرد. در این روش تمام راه حلها به دو گروه تقسیم می شوند: «حلقه باز» و «حلقه بسته»^۱. راه حلهای «حلقه باز» سعی می کنند یک مسئله را با طراحی خوب حل کرده و از همان ابتدا اطمینان بدهند که مشکلی رخ نخواهد داد. وقتی این سیستم شروع به کار و انجام فعالیت نمود هیچگونه نظارت یا تصحیح عملکرد ممکن نیست.

تمهیداتی که برای کنترل ازدحام به شبکه «حلقه باز» پیش بینی شده عبارتند از: تصمیم گیری در خصوص زمان پذیرش ترافیک جدید، تصمیم گیری در خصوص زمان حذف بسته ها، انتخاب بسته هایی که باید حذف شوند و تصمیم گیری در خصوص زمان بندی صحیح در شبکه؛ حقیقت مشترک در تمام این موارد آنست که تصمیم گیری آنها مبتنی بر شرایط جاری شبکه نیست.

در مقابل، راه حلهای «حلقه بسته» مبتنی بر مفهوم حلقه های فیدبک هستند. اینگونه راهکارهای کنترل ازدحام، سه بخش را در بر می گیرند:

۱. نظارت بر سیستم به منظور تشخیص آنکه در کجا و چه وقت ازدحام رخ داده است.
۲. تحویل این اطلاعات به محلی که بایستی واکنش نشان بدهد.
۳. تنظیم عملکرد سیستم برای رفع مشکل

برای نظارت و تشخیص بروز ازدحام در زیر شبکه می توان معیارهای گوناگونی را بکار گرفت. مهمترین آنها عبارتند: درصد بسته هایی که به دلیل فقدان فضای کافی بافر حذف می شوند، متوسط طول صف، تعداد بسته هایی که مهلت ارسال آنها منقضی شده و از نو ارسال گردیده اند، متوسط تأخیر بسته و انحراف معیار^۲ تأخیر بسته. در تمام این معیارها رشد اعداد نمایانگر افزایش ازدحام است.

دومین مرحله از «حلقه فیدبک» آن است که اطلاعاتی در خصوص ازدحام، از محل تشخیص و بروز آن به محلی که می تواند کاری برای حل آن انجام بدهد، انتقال یابد. بدیهی ترین روش آن است که مسیریاب کشف کننده ازدحام بسته ای خاص برای ماشین یا ماشینهای مبدأ بفرستد و مشکل را به آنها اعلام نماید. البته این بسته های اضافی خودشان به بار شبکه می افزایند، آن هم دقیقاً زمانی که شبکه ظرفیت هیچ بار اضافی ندارد یعنی دقیقاً حین ازدحام در زیر شبکه!

با این وجود روشهای دیگری نیز امکان پذیر است. به عنوان مثال می توان یک بیت یا یک فیلد در هر بسته کنار گذاشت تا هرگاه ازدحام از یک حد آستانه فراتر رفت، مسیریاب آنرا پر کند. هرگاه مسیریاب تشخیص بدهد که ازدحام رخ داده در تمام بسته های خروجی، این فیلد را پر می کند تا به همسایه های خود در این خصوص هشدار بدهد.

راهکار دیگر آن است که ماشینهای میزبان یا مسیریابها بطور متناوب بسته های «آزمون» تولید و ارسال کرده و

۱. Open loop , Close loop

۲. Standard deviation

مستقیماً در خصوص ازدحام کسب آگاهی کنند. بکمک این اطلاعات می‌توان بسته‌ها را بنحوی مسیریابی کرد که از ناحیه بروز مشکل، ردّ نشوند. به مثابه ایستگاههای کنترل ترافیک که هلی کوپترهای آنها بر روی شهرها به پرواز درمی‌آیند تا به شتودگان در حال حرکت هشدار بدهند تا بسوی نقاط بحران، حرکت نکنند.

در تمام روشهای مبتنی بر فیدبک، انتظار می‌رود که آگاهی از ازدحام موجب شود ماشینهای میزبان برای کاهش ازدحام از خود واکنش مناسبی نشان بدهند. برای آنکه این روشها به درستی کار کنند بایستی مقیاس زمان به دقت تنظیم شده باشد. اگر وقتی دو بسته پشت سر هم می‌رسند مسیریاب فریاد بزند «ایست» و به محض آنکه فقط برای ۲۰ میکروثانیه آزاد می‌شود فریاد بزند: «حرکت»، سیستم بشدت نوسانی عمل می‌کند و هیچگاه همگرا و پایدار نخواهد شد. برعکس اگر قبل از اعلام هر چیزی برای اطمینان به مدت ۳ دقیقه صبر کند، مکانیزم کنترل ازدحام آنقدر کند واکنش نشان می‌دهد که عملاً هیچ سودی در دنیای واقعی نخواهد داشت. برای آنکه عملکرد خوبی انتظار داشته باشیم به حالت میانه‌ای نیازمندیم ولیکن بدست آوردن «ثابت زمانی سیستم» (Time Constant) مسئله چندان ساده‌ای نیست.

الگوریتمهای متعددی برای کنترل ازدحام معرفی شده‌اند. برای آنکه بتوان این الگوریتمها را به روش قابل فهمی سازماندهی کرد، Yang و Reddy (۱۹۹۵) الگوریتمهای کنترل ازدحام را طبقه‌بندی کرده‌اند. آنها به همان ترتیبی که در بالا تشریح شد تمام الگوریتمها را به دو رده «حلقه باز» و «حلقه بسته» تقسیم‌بندی نمودند. مضاف بر این، الگوریتمهای «حلقه باز» را برحسب اینکه در ماشینهای مبدا عمل می‌کنند یا در ماشینهای مقصد به دو رده تقسیم کردند. الگوریتمهای «حلقه بسته» نیز به دو رده فرعی تقسیم شده‌اند: «الگوریتمهای مبتنی بر فیدبک مستقیم» و «الگوریتمهای مبتنی بر فیدبک ضمنی». در الگوریتمهای مبتنی بر فیدبک مستقیم، برای هشدار دادن به مبدا، بسته‌های فیدبک دقیقاً از محل بروز ازدحام برگردانده می‌شوند. در الگوریتمهای مبتنی بر فیدبک ضمنی، مبدا با استناد به برخی از مشاهدات و علائم محلی (همانند زمان لازم برای برگشت بسته‌های ACK) بروز ازدحام را حدس می‌زند.

بروز ازدحام بدین معناست که «بار شبکه» موقتاً از «منابع» موجود در برخی از بخشهای سیستم بیشتر شده و شبکه از عهده این بار بر نمی‌آید. [منظور از منابع پهنای باند، توان پردازشی CPU، حافظه مسیریاب و نظایر آنهاست. -م] دو راه حلّ به ذهن متبادر می‌شود: افزایش منابع یا کاهش بار. به عنوان مثال زیر شبکه می‌تواند برای افزایش پهنای باند بین دو نقطه، موقتاً از یک خطّ تلفن (Dialup) کمکی بهره بگیرد. در سیستمهای ماهواره‌ای، افزایش توان فرستنده اغلب پهنای باند را افزایش می‌دهد. در ضمن اگر بجای استفاده دائمی از بهترین مسیر، بخشی از بار بر روی مسیرهای دیگر تقسیم شود پهنای باند شبکه بطور مؤثری افزایش می‌یابد. نهایتاً آنکه می‌توان مسیریابهای یکنی را که فقط به عنوان پشتیبان در شبکه نصب شده‌اند (برای آنکه سیستم تحمل خرابی داشته باشد) به خدمت گرفت تا در هنگام بروز ازدحام، ظرفیت زیر شبکه افزایش یابد.

علیرغم این راهکارها، بسیاری از اوقات افزایش ظرفیت زیر شبکه ممکن نیست یا آنکه این ظرفیت تا آخرین حدّ افزایش یافته است. تنها راه باقیمانده برای رفع ازدحام کاهش بار است. چندین روش برای کاهش بار وجود دارد که از آن جمله می‌توان به این موارد اشاره کرد: (۱) اجتناب از سرویس دادن به برخی از کاربران (۲) کاهش سطح سرویس دهی به برخی یا تمام کاربران (۳) مجبور کردن کاربران به زمان‌بندی تقاضاهای خود به روشی قابل پیش‌بینی.^۱

۱. از آنجایی که برخی از ماشینها ترافیک انفجاری و غیر قابل پیش‌بینی تولید می‌کنند لذا می‌توان آنها را وادار کرد تا بکمک مکانیزمهای خاص (همانند مکانیزم بافرینگ یا مکانیزمهای شکل‌دهی به ترافیک) حجم ترافیک خود را متعادل و قابل پیش‌بینی کنند. -م

برخی از این روشها که به اختصار آنها بررسی خواهیم کرد در زیر شبکه های نوع «مدار مجازی» به بهترین نحو قابل اعمال هستند. در زیر شبکه های مدار مجازی، این روشها در لایه شبکه پیاده می شوند. در زیر شبکه های دیتاگرام علیرغم اصراری که بر تفکیک وظایف لایه ها وجود دارد بخشی از این روشها باید در اتصالات لایه انتقال بکار گرفته شوند. در این فصل به کاربرد این روشها در لایه شبکه می پردازیم و در فصل بعدی به عملیات مدیریت ازدحام در لایه انتقال، نگاهی خواهیم انداخت.

۲-۳-۵ سیاستهای پیشگیری از ازدحام

اجازه بدهید مطالعه روشهای کنترل ازدحام را با بررسی سیستمهای «حلقه باز» شروع کنیم. اینگونه سیستمها به گونه ای طراحی می شوند تا بجای آنکه بگذارند ازدحام اتفاق بیفتد و بعداً واکنش نشان بدهند، در همان ابتدا ازدحام را به حداقل برسانند. این روشها برای رسیدن بدین هدف، از سیاستهای خاص و مناسب در سطوح مختلف، بهره می گیرند. در شکل ۵-۲۶ سیاستهای مختلفی که در لایه پیوند داده، لایه شبکه و لایه انتقال، می تواند بر پدیده ازدحام تأثیر بگذارد، فهرست شده اند. (Jain, 1990)

سیاستها	لایه
<ul style="list-style-type: none"> سیاستهای ارسال مجدد سیاستهای ذخیره بسته هایی که خارج از ترتیب می رسند. سیاستهای تصدیق وصول بسته ها (Ack) سیاستهای کنترل جریان تعیین زمان انقضای مهلت نابرها 	انتقال
<ul style="list-style-type: none"> مدار مجازی در مقابل روش دیتاگرام در زیر شبکه مکانیزمهای صف بندی بسته ها و روشهای متنوع سرویس دهی سیاستهای حذف بسته الگوریتم مسیریابی مدیریت طول عمر بسته ها 	شبکه
<ul style="list-style-type: none"> سیاستهای ارسال مجدد سیاستهای ذخیره بسته هایی که خارج از ترتیب می رسند. سیاستهای تصدیق وصول بسته ها (Ack) سیاستهای کنترل جریان 	پیوند داده

شکل ۵-۲۶. سیاستهایی که بر پدیده ازدحام تأثیر می گذارند.

اجازه بدهید از لایه پیوند داده شروع کرده و سپس به سمت بالا حرکت کنیم. سیاستهای ارسال مجدد (Retransmission Policy) یا مسائلی از این قبیل سر و کار دارد: (۱) با چه سرعتی مهلت فرستنده (برای دریافت Ack یک بسته) خاتمه می یابد؟ (۲) در اثر انقضای مهلت چه چیزی ارسال می شود؟

یک فرستنده عجول که مهلت آن به سرعت خاتمه می یابد و مبتنی بر پروتکل Go Back n تمام بسته های ارسالی را از نو می فرستد، بار بسیار سنگینتری را نسبت به فرستنده ای که از روش Selective Repeat استفاده کرده به زیر شبکه تحمیل می کند. یکی دیگر از مسائل که ارتباط تنگاتنگی با این موضوع دارد سیاستهای بافرینگ است. اگر گیرنده، بسته هایی را که خارج از ترتیب می رسند دور بیندازد تمام آنها باید از نو ارسال شوند و بار اضافی به زیر شبکه تحمیل خواهد شد. از دیدگاه کنترل ازدحام، روش «تکرار انتخابی» (Selective Repeat) به وضوح بهتر از روش Go Back n عمل می کند.

سیاست تصدیق دریافت بسته ها (Acknowledgement Policy) نیز بر روی ازدحام تأثیر می گذارد. اگر هر

بسته فوراً اعلام وصول شود، بسته‌های اعلام وصول (Ack) ترافیک زائد و اضافی تحمیل خواهند کرد. در عوض اگر برای صرفه‌جویی، از روش Piggybacking استفاده شود و اعلام وصول داده‌ها از طریق ترافیک معکوس انجام شود ممکن است مهلت فرستنده داده‌ها متوالیاً منقضی شود و تکرار ارسال رخ بدهد که آن هم منجر به تحمیل بار اضافه خواهد شد. استفاده از یک روش محکم و سختگیرانه در الگوی کنترل جریان (مثلاً داشتن پنجره کوچک) می‌تواند نرخ ارسال داده را کاهش داده و به کاهش ازدحام کمک کند.

در لایه شبکه، انتخاب بین روش «مدار مجازی» یا «دیتاگرام» تأثیر مستقیمی بر پدیده ازدحام دارد چراکه بسیاری از الگوریتمهای کنترل ازدحام فقط در زیرشبکه‌های مدار مجازی قابل اعمال هستند. صف‌بندی بسته‌ها و سیاستهای سرویس‌دهی نیز در کنترل ازدحام موثرند. این سیاستها بدین موضوع مرتبطند که آیا مسیریاب به ازای هر خط ورودی یک صف دارد، به ازای هر خط خروجی یک صف دارد یا هر دو صف را تشکیل می‌دهد. همچنین ترتیب پردازش بسته‌ها (مثلاً نوبت‌بندی چرخشی Round Robin یا روش مبتنی بر اولویت‌بندی) جزو سیاستها صف‌بندی محسوب می‌شود. «سیاست حذف بسته‌ها» قاعده‌ای است که براساس آن تعیین می‌شود که وقتی فضایی برای نگهداری بسته‌ها نیست کدامیک از بسته‌ها را باید حذف کرد. اتخاذ یک سیاست مناسب می‌تواند به کاهش ازدحام کمک کند و سیاستهای غلط شرایط ازدحام را بدتر خواهد کرد.

یک الگوریتم مسیریابی خوب می‌تواند با توزیع مناسب ترافیک بین خطوط مختلف به پیشگیری از ازدحام کمک کند در حالی که یک الگوریتم بد می‌تواند با ارسال ترافیک بسیار زیاد بر روی یک خط که با ازدحام مواجه شده شرایط را بدتر کند. مدیریت طول عمر بسته (Packet Lifetime) با این مسئله سروکار دارد که چه مدت طول می‌کشد تا بسته [در صورت نرسیدن به مقصد] حذف شود. اگر این زمان بیش از حد طولانی باشد بسته‌های گمشده و سرگردان ممکن است تا قبل از موعد حذف شدن، شبکه را با انباشتگی و ازدحام مواجه کنند، در حالی که اگر این زمان کوتاه در نظر گرفته شود ممکن است بسته‌ها قبل از آنکه فرصت رسیدن به مقصد را داشته باشند حذف شوند و به تکرار ارسال بینجامد.

در لایه انتقال همان موارد و سیاستهایی که در لایه پیوند داده وجود داشت دنبال می‌شود ولی مضاف بر آنها روش تعیین بازه‌های زمانی انقضای مهلت (Timeout Interval) پیچیده‌تر است چراکه زمان عبور بسته‌ها از میان یک شبکه [با تعدادی مسیریاب و کانالهای متعدد] در مقایسه با عبور بسته‌ها از یک سیسم واحد، چندان قابل پیش‌بینی نیست. اگر بازه‌های انقضای مهلت بیش از اندازه کوتاه باشد بسته‌های بی‌مصرف زیادی ارسال خواهد شد. اگر این زمان بیش از اندازه طولانی باشد، ازدحام کاهش می‌یابد ولیکن وقتی بسته‌ای به هر دلیل از دست برود زمان پاسخ [یعنی تأخیر ارسال مجدد] مشکل‌آفرین خواهد شد.

۳-۳-۵ کنترل ازدحام در زیرشبکه‌های مدار مجازی

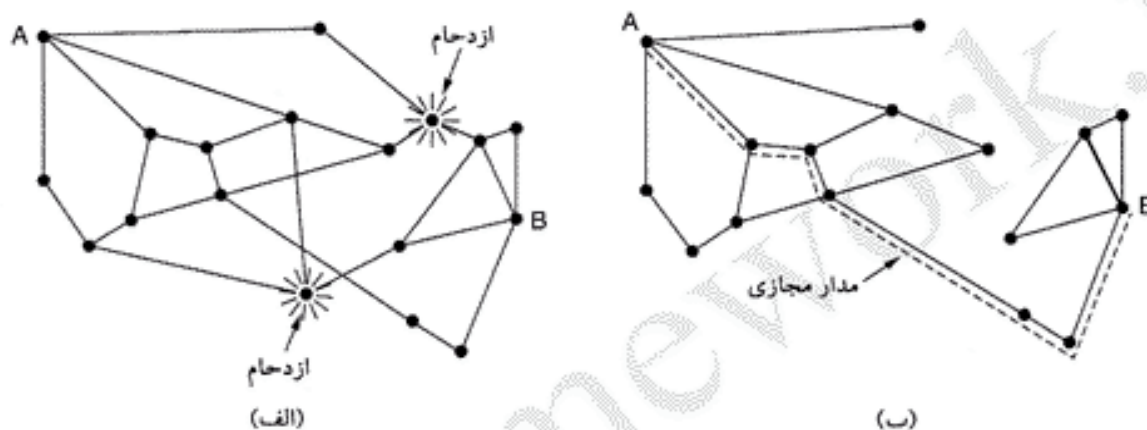
روشهای کنترل ازدحام که در بالا معرفی شدند اساساً «حلقه باز» هستند یعنی تلاش می‌کنند در همان ابتدا و قبل از بروز، از ازدحام پیشگیری کنند. در این بخش راهکارهایی را بررسی خواهیم کرد که در زیرشبکه‌های مدار مجازی به صورت پویا ازدحام را کنترل می‌کنند. در دو بخش آتی نیز نگاهی به تکنیکهای قابل استفاده در هر نوع زیرشبکه خواهیم انداخت.

یکی از تکنیکهایی که بطور گسترده‌ای از آن در جهت پیشگیری از وخیم شدن مشکل ازدحام (که از قبل شروع شده) استفاده می‌شود روش «کنترل پذیرش»^۱ (Admission Control) نام دارد. ایده این روش ساده است: هر گاه ازدحام گزارش شود تا رفع کامل مشکل، هیچگونه تقاضای تنظیم مدار مجازی پذیرفته نخواهد شد. بدین

۱. بمعنای محدود کردن پذیرش ارتباط و امتناع از ایجاد مدار مجازی جدید

ترتیب هر گونه تلاش برای ایجاد اتصال در لایه انتقال با شکست مواجه خواهد شد چراکه پذیرش افراد جدید وضع ازدحام را وخیم تر خواهد کرد. هر چند این روش خام و ناشیانه است ولی در عوض ساده و سهل الاجراست. در سیستمهای تلفن نیز هر گاه یک سوئیچ یا بار بسیار زیاد مواجه شود، با قطع بوق آزاد (Dial Tone)، پذیرشهای جدید را محدود می نماید.

راهکار دیگر آن است که اجازه تنظیم مدار مجازی جدید داده شود ولیکن مسیرهای انتخابی برای مدارات مجازی جدید در خارج از نواحی بحران زده انتخاب گردد. به عنوان مثال زیر شبکه شکل ۵-۲۷-الف را در نظر بگیرید که در آن دو مسیر یاب با ازدحام مواجه شده اند.



شکل ۵-۲۷. (الف) یک زیر شبکه مواجه با ازدحام (ب) ترسیم مجدد گراف زیر شبکه پس از حذف مناطق درگیر ازدحام. مدار مجازی مناسب بین A و B نشان داده شده است.

فرض نمایید که ماشین میزبان متصل به مسیر یاب A بخواهد یک «اتصال» با ماشین متصل به B برقرار نماید. طبیعتاً این اتصال از یکی از مسیر یابهای دارای مشکل عبور خواهد کرد. برای اجتناب از این وضعیت می توانیم زیر شبکه را مطابق با شکل ۵-۲۷-ب از نو ترسیم کرده و مسیر یابهای دچار ازدحام و تمام خطوط آنها را حذف نماییم. خط نقطه چین، مسیر ممکن برای ایجاد مدار مجازی بین A و B را نشان می دهد. در این مسیر، مسیر یابها و خطوط دچار ازدحام وجود ندارد.

استراتژی دیگر در زیر شبکه های مدار مجازی، آنست که در حین تنظیم یک مدار مجازی، بین ماشین میزبان و زیر شبکه توافقاتی صورت بگیرد. در این توافقات، عموماً حجم و شکل ترافیک، کیفیت مورد نیاز خدمات و پارامترهای دیگر مشخص می شوند. به منظور عمل به مفاد این قرارداد، زیر شبکه در هنگام تنظیم مسیر، منابع مورد نیاز را در طول مسیر، از قبل کنار می گذارد. این منابع شامل فضای بافر، فضای جدول در هر مسیر یاب و پهنای باند خطوط است. در این روش احتمال بروز ازدحام در مدارات مجازی نادر است چرا که تمام منابع مورد نیاز از قبل رزرو و موجود بودن آن تضمین می شود.

این گونه رزرو سازی می تواند به صورت یک روال عملیاتی استاندارد همیشه انجام شود و یا فقط در شرایطی اتخاذ شود که شبکه با ازدحام مواجه شده است. اشکال استفاده همیشگی از این روش آن است که می تواند منابع را هدر بدهد. به عنوان مثال اگر شش مدار مجازی که هر یک به پهنای باند 1Mbps نیاز مندند همگی از یک خط فیزیکی 6Mbps عبور کرده باشند مسیر یاب مجبور به علامت گذاری آن به عنوان خط پر خواهد بود در حالی که به ندرت اتفاق می افتد که هر شش مدار مجازی بطور همزمان مورد استفاده قرار بگیرند. نتیجتاً در شرایط معمولی هزینه ای که برای کنترل ازدحام پرداخت می شود پهنای باند بلا استفاده (تلفاتی) است.

۵-۳-۴ کنترل ازدحام در زیر شبکه های دیتاگرام

حال اجازه بدهید به راهکارهایی پردازیم که می توان از آنها در زیر شبکه های دیتاگرام (و همچنین زیر شبکه های مدار مجازی) بهره گرفت. هر مسیریاب می تواند براحتی بر میزان بهره وری (Utilization) خطوط خروجی و دیگر منابع خودش نظارت داشته باشد. به عنوان مثال مسیریاب می تواند به هر یک از خطوط خود یک متغیر اعشاری u که مقداری بین 0.0 تا 1.0 دارد نسبت بدهد. مقدار این متغیر میزان بهره وری خط را منعکس می کند. برای آنکه بتوان تخمین دقیقی از u داشت می توان بطور متناوب و در لحظات خاص از میزان بهره وری خط^۱ نمونه برداری کرد و با فرض آنکه این مقدار f محاسبه شده باشد (f نیز بین صفر و یک است)، مقدار u را طبق رابطه زیر بهنگام سازی نمود:

$$u_{\text{new}} = \alpha \cdot u_{\text{old}} + (1 - \alpha) \cdot f$$

α یک ثابت است که تعیین می کند مسیریاب با چه سرعتی وضعیت گذشته را فراموش خواهد کرد.

هر گاه u از یک مقدار آستانه (Threshold) [یا به عبارتی از حد مجاز] تجاوز کند آن خط خروجی در وضعیت «هشدار» وارد می شود و در این صورت باید عملیاتی برای خروج از این وضعیت انجام گیرد. این عملیات می تواند یکی از گزینه های زیر باشد که در ادامه آنها را تشریح خواهیم کرد.

بیت هشدار (The Warning Bit)

در معماری قدیم شبکه های DECNET، «وضعیت هشدار» با تنظیم یک بیت خاص در سرآیند بسته ها، اعلام می شد. در شبکه Frame Relay نیز همینگونه است. وقتی بسته ای به مقصد خود می رسد، لایه انتقال همان بیت را در درون بسته Ack کپی کرده و آن را برای مبدا پس می فرستاد و بدین نحو ماشین مبدا ترافیک خود را تقلیل می داد.

مادامیکه مسیریاب در وضعیت هشدار قرار داشت، تنظیم این بیت نیز ادامه می یافت بدین معنا که ماشین مبدا در بسته های بعدی Ack بازم این بیت را دریافت می کرد. مبدا نیز با شمارش بسته های Ack حساب می کرد که در چه کسری از این بسته ها بیت هشدار وجود دارد و براساس آن نرخ انتقال خود را تنظیم می نمود. تا وقتی که دریافت این بیتها ادامه می یافت مبدا نیز به کاهش نرخ ارسال خود ادامه می داد. پس از کاهش نرخ ارسال تا حد نهایی، مجدداً نرخ ارسال شروع به افزایش می کرد [در صورت خروج از وضعیت هشدار]. دقت کنید که چون هر مسیریاب واقع بر روی مسیر می توانست «بیت هشدار» را به ۱ تنظیم کند لذا فقط زمانی ترافیک افزایش می یافت که هیچ یک از مسیریابها مشکلی نداشتند.

بسته های دعوت به آرامش (Choke Packet)

الگوریتم کنترل ازدحام قبلی نسبتاً زیرکانه است چرا که با یک بیان غیر مستقیم و تلویحی به مبدا تفهیم می کند که باید از سرعت خود بکاهد. چرا این کار مستقیماً انجام نشود؟ برای این کار مسیریاب یک بسته به نام «دعوت به آرامش» (Choke Packet) به ماشین مبدا بر گردانده و در آن آدرس مقصد بسته را نیز درج می کند [تا ماشین مبدا آگاه شود بسته های ارسالی او در راه رسیدن به کدام مقصد در مسیر پر ازدحام قرار گرفته اند و بداند که نرخ ارسال برای چه مقصدی را باید کاهش بدهد چرا که یک ماشین ممکن است بطور همزمان برای چند ماشین بسته ارسال کند -م]. البته بسته اصلی، علامتگذاری شده و به راه خود ادامه می دهد (فقط یک بیت خاص در سرآیند

۱. نسبت داده های ارسالی بر روی خط به ظرفیت کل خط را بهره وری آن خط در نظر بگیرید. اندازه گیری این نسبت برای هر خط چندان دشوار نیست و برخی از سخت افزارهای شبکه امکان این اندازه گیری را در اختیار می گذارند. -م

بسته تنظیم می شود). این علامتگذاری از آن جهت انجام می شود که بسته های «دعوت به آرامش» بیشتری در طول مسیر تولید نشود؛ سپس بسته اصلی بروش معمول به سوی مقصد خود هدایت می شود.

هرگاه ماشین مبدا، بسته دعوت به آرامش دریافت کند، ملزم به کاهش ترافیک ارسالی بدان مقصد خاص (تا X درصد) می باشد. از آنجایی که ممکن است بسته های دیگری که قبلاً به همان مقصد روانه شده اند در همان مسیر حرکت کرده باشند و آنها نیز منجر به تولید بسته های «دعوت به آرامش» دیگری شده باشند لذا ماشین مبدا پس از دریافت اولین بسته «دعوت به آرامش» به مدت زمان ثابت و مشخصی بسته هایی از این نوع را نادیده می گیرد. پس از طی این مدت، ماشین میزبان مجدداً به اندازه زمان مشخصی گوش می دهد که ببیند آیا «بسته دعوت به آرامش» دیگری دریافت می شود؟ اگر چنین بسته ای دریافت شد، خط کماکان در وضعیت ازدحام است فلذا باز هم جریان داده های خود را کاهش می دهد و مجدداً برای مدتی بسته های دعوت به آرامش را نادیده می گیرد. برعکس اگر در خلال مدت گوش دادن هیچ بسته دعوت به آرامش دریافت نشد، ماشین میزبان می تواند جریان داده های خود را افزایش بدهد. در این پروتکل «فیدبک ضمنی» کمک می کند تا بتوان قبل از آنکه جریان بسته ها بطور کل مسدود شود از ازدحام پیشگیری کرد مگر آنکه مشکلی جدی رخ بدهد.

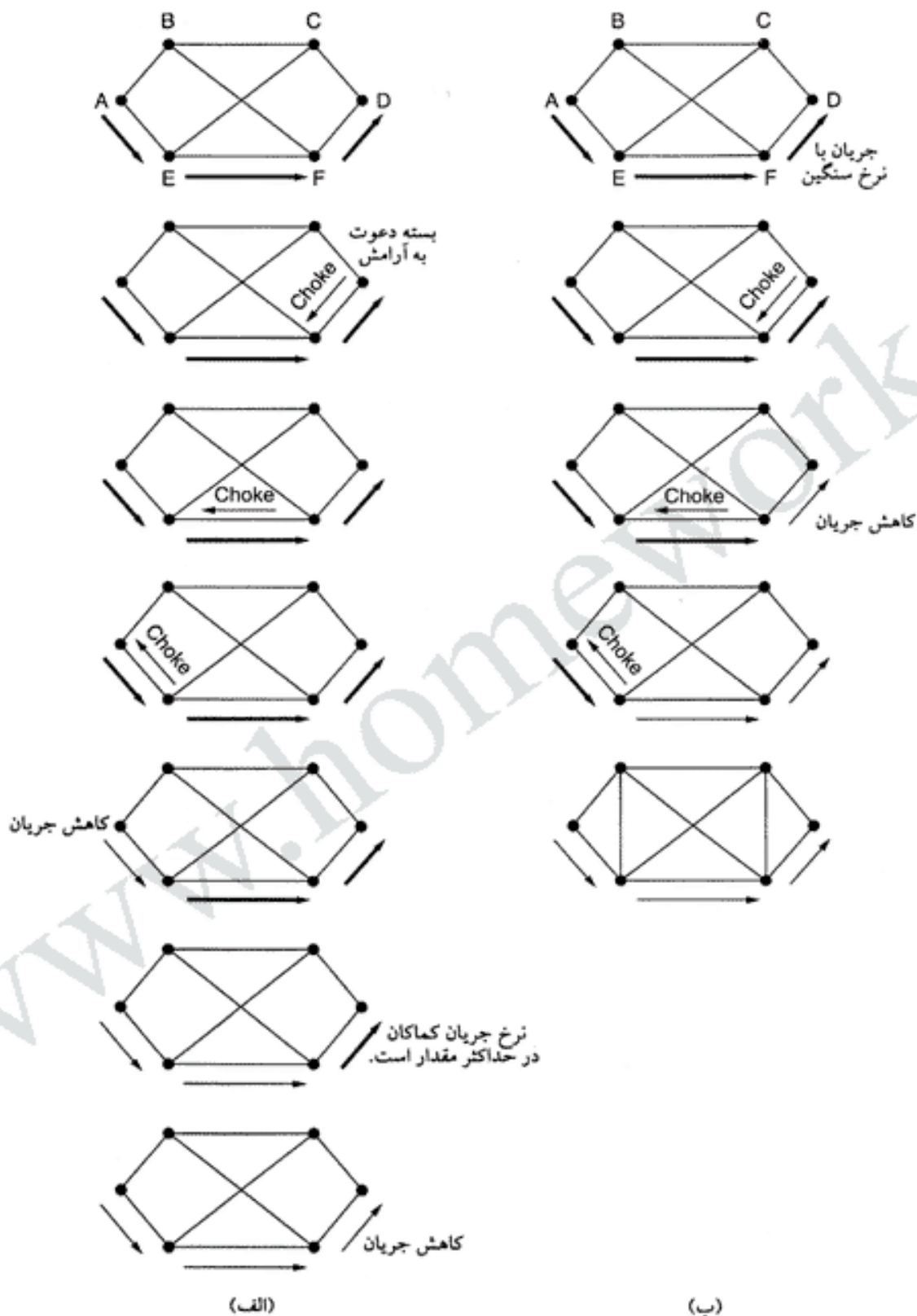
ماشینهای میزبان می توانند با تنظیم پارامترها و تغییر در سیاستهای کنترل ازدحام مثل اندازه پنجره، ترافیک ارسالی خود را کاهش بدهند. عموماً دریافت یک بسته «دعوت به آرامش» باعث پنجاه درصد کاهش در نرخ قبلی ارسال می شود. بسته بعدی، ترافیک را به ۲۵ درصد تقلیل می دهد و به همین ترتیب. [به عبارت دیگر دریافت بسته دعوت به آرامش مقدار فعلی نرخ ترافیک را عموماً نصف می کند. -م] افزایش نرخ ارسال معمولاً ضریب رشد کمتری دارد تا از بروز مجدد و پیاپی ازدحام پیشگیری شود. [یعنی مثلاً اگر دریافت بسته دعوت به آرامش نرخ ارسال را نصف می کند عدم دریافت مجدد آن در یک دوره زمانی مشخص، باعث دو برابر شدن نرخ ارسال نخواهد شد. -م]

چندین گونه از این الگوریتم برای کنترل ازدحام ارائه شده است. مثلاً در یکی از آنها، مسیر یاب چندین سطح آستانه (Threshold) در نظر می گیرد. بسته به آنکه وضعیت از کدامیک از سطوح تجاوز کرده باشد بسته های دعوت به آرامش می توانند در برگیرنده یکی از سه هشدار «وضعیت احتیاط»، «وضعیت حاد» و «بحران قطعی» باشند.

در گونه دیگری از این الگوریتم برای صدور علان هشدار دهنده، بجای استفاده از معیار بهره وری خط از معیار طول صف یا میزان استفاده از فضای بافر بهره گرفته می شود. البته می توان از یک رابطه وزن دهی نمایی (Exponential Weighting) استفاده کرد و براساس تمامی این معیارها یک مقدار تلفیقی برای میزان بهره وری (u) تعریف نمود.

بسته های دعوت به آرامش گام به گام (Hop-by-Hop Choke Packet)

در سرعتهای بالا یا در مسیرهای طولانی، ارسال بسته های «دعوت به آرامش» به ماشین مبدا، کارایی خوبی نخواهد داشت چراکه واکنش این روش بسیار کند است. به عنوان مثال فرض کنید که ماشینی در سان فرانسیسکو (مثلاً مسیر یاب A در شکل ۵-۲۸) ترافیکی از بسته ها را با سرعت ۱۵۵ مگابیت در ثانیه برای ماشینی در نیویورک (مثلاً مسیر یاب D در شکل ۵-۲۸) ارسال کند. اگر بافر ماشین در نیویورک، رو به پر شدن بگذارد حدوداً سی میلی ثانیه طول می کشد تا بسته های دعوت به آرامش به سان فرانسیسکو برگردد و از مبدا بخواهد کندتر ارسال کند. در شکل ۵-۲۸ الف انتشار و بازگشت «بسته دعوت به آرامش» در مراحل دوم، سوم و چهارم نشان داده شده است. در خلال این سی میلی ثانیه تاخیر، ۴/۶ مگابیت اطلاعات دیگر فرستاده خواهد شد. حتی اگر پس از دریافت



شکل ۵-۲۸. (الف) یک بسته دعوت به آرامش که فقط مبداء را تحت تاثیر قرار می دهد. (ب) یک بسته دعوت به آرامش که در هر گام، بر روی مسیر یابها واقع بر مسیر تاثیر می گذارد.

این هشدار ماشین واقع در سانفرانسیسکو فوراً ارسال خود را بطور کلی قطع نماید، ۴/۶ مگابیت اطلاعات بر روی خط به سوی ماشین روانه هستند و باید به نحوی تکلیف آنها مشخص شود. فقط در مرحله هفتم از شکل ۵-۲۸-الف مسیر یاب واقع در نیویورک متوجه کاهش ترافیک خواهد شد.

راهکار دیگر آن است که بسته دعوت به آرامش در هر گام از مسیری که می‌پیماید تأثیری بر عملکرد مسیر یابهای میانی بگذارد. (به شکل ۵-۲۸-ب دقت کنید). در اینجا به محض آنکه بسته دعوت به آرامش به مسیر یاب F می‌رسد، F ملزم به کاهش ترافیک ارسالی به D خواهد بود. انجام این کار منوط به آن است که F فضای بافر بیشتری را برای جریان بسته‌ها اختصاص بدهد چرا که ماشین مبدا با سرعت کامل در حال ارسال است و F در این میان می‌تواند کمکی موقت و سریع به D بنماید [با ایجاد توقف مصنوعی]؛ این کار را می‌توانید همانند آگهی‌های تجاری فرض کنید که شما را موقتاً از سردرد ناشی از دیدن یک برنامه طولانی نجات می‌دهند!! در گام بعدی بسته دعوت به آرامش به E می‌رسد و به او نیز تفهیم می‌شود که ترافیک خود به F را کاهش بدهد. این عمل نیز مستلزم وجود فضای بافر بیشتری در E است ولی F را موقتاً یاری می‌دهد. نهایتاً بسته دعوت به آرامش به A می‌رسد و جریان بطور واقعی و از مبدا آن کاهش می‌یابد.

تأثیر نهایی این روش گام به گام (Hop-by-Hop) آن است که در لحظه بروز ازدحام به سرعت چاره‌ای اندیشیده می‌شود ولی این راهکار به بهای افزایش فضای بافر ارسال، تمام خواهد شد. در این روش می‌توان بدون از دست رفتن هیچ بسته‌ای، مانع از افزایش ازدحام شد. این نظریه و نتایج شبیه‌سازی آن در مرجع (Mishra and Kanakia, 1992) تشریح شده است.

۵-۳-۵ دور ریختن بار (Load Shedding)

هر گاه هیچیک از روشهای فوق مشکل ازدحام را رفع نکنند، مسیر یابها می‌توانند آخرین تیر ترکش خود را بیازمایند: «دور ریختن بار»! عمل دور ریختن بار، آخرین و بدترین روشی است که به مسیر یاب اجازه می‌دهد هرگاه با سیل بسته‌هایی که نمی‌تواند از عهده هدایت آنها برآید، مواجه شود آنها را دور بریزد. (این اصطلاح از ادبیات شرکتهای تولید انرژی برق گرفته شده که در آنجا نیز مثلاً در یک روز گرم تابستان که مصرف برق از میزان تولید بیشتر می‌شود، برق بخشی از مناطق عمداً قطع می‌شود تا کل مناطق با مشکل قطع برق مواجه نشوند!)

یک مسیر یاب که غرق در بسته‌های اطلاعاتی شده می‌تواند تصادفاً برخی از آنها را انتخاب و حذف کند ولی از این بهتر هم می‌تواند عمل کند. اینکه کدام بسته باید حذف شود به نوع برنامه کاربردی که آنرا تولید کرده بستگی دارد. در انتقال فایل بسته‌های قدیمی ارزشمندتر از بسته‌های جدید هستند چرا که حذف بسته ۶ و نگه داشتن بسته‌های ۷ تا ۱۰ باعث ایجاد یک شکاف در میان داده‌های گیرنده شده و ممکن است فرستنده مجبور شود بسته‌های ۶ تا ۱۰ را نیز از نو ارسال کند (اگر گیرنده بطور طبیعی بسته‌های خارج از ترتیب را حذف کند). در یک فایل ۱۲ بسته‌ای، حذف بسته ۶ ممکن است منجر به تکرار ارسال بسته‌های ۶ تا ۱۰ شود در حالی که حذف بسته ۱۰ ممکن است فقط به تکرار بسته‌های ۱۰ تا ۱۲ نیاز داشته باشد. در طرف مقابل برای کاربردهای چند رسانه‌ای (مثل ارسال صدا یا تصویر)، بسته‌های جدید مهمتر از بسته‌های قدیمینند.

برای آن که سطح هوشمندی الگوریتم حذف بسته‌ها از این هم فراتر برود به همکاری فرستنده بسته‌ها نیاز است. در بسیاری از برنامه‌های کاربردی، برخی از بسته‌ها بسیار مهمتر از بقیه هستند. به عنوان مثال در برخی الگوریتمهای ارسال ویدیوی فشرده شده، در لحظات خاصی یک فریم تصویر کامل ارسال می‌شود و پس از آن فریمهای تصویر بعدی کامل نیستند بلکه فقط تفاوت این فریمها با آخرین فریم کامل محاسبه و پس از فشرده‌سازی ارسال می‌شوند. در چنین حالتی حذف بسته‌ای که بخشی از فریمهای فرعی محسوب می‌شود ارجح‌تر از حذف بسته‌ای است که به فریم اصلی و کامل تعلق دارد. به عنوان مثالی دیگر، انتقال یک سند

(Document) حاوی تصویر و متن ASCII را در نظر بگیرید. از دست دادن یک خط از نقاط تصویر در برخی از عکسها، کم ضررتر از دست دادن یک خط از متن است.

برای پیاده‌سازی یک سیاست هوشمند برای حذف بسته‌ها، برنامه‌های کاربردی باید رده اولویت مورد نظر را در بسته‌های خود مشخص کنند تا میزان اهمیت آنها مشخص گردد. اگر چنین کاری انجام شده باشد زمانی که مسیریاب مجبور به حذف بسته‌ها می‌شود می‌تواند از بسته‌های با اولویت پایین آغاز کند و بعداً به سراغ رده‌های بالاتر برود. البته فقط در موارد خاص می‌توان بسته‌هایی را در بالاترین رده اولویت علامتگذاری کرد.

البته چون هر ماشین در ارسال بسته‌ها با هر اولیوی، آزادی عمل دارد لذا باید به نحوی در کاربر انگیزه ایجاد کرد تا بسته‌هایش را با اولویت پایین بفرستد؛ پول می‌تواند این انگیزه را ایجاد کند؛ اگر برای حمل بسته‌های با اولویت بالا هزینه بیشتری گرفته شود کاربران سعی می‌کنند بی‌مورد از اولویتهای بالا استفاده نکنند. البته فرستنده ممکن است اجازه ارسال بسته‌های با اولویت بالا را در شرایط پارسیک به کاربر بدهد ولیکن با افزایش بار حذف خواهند شد. این موضوع کاربران را تشویق می‌کند از ارسال بی‌مورد بسته‌های با اولویت بالا صرف‌نظر کنند.

گزینه دیگر آنست که به ماشینهای میزبان اجازه بدهیم گاهی بیش از حد توافق شده در هنگام ایجاد مدار مجازی، ارسال داشته باشند ولیکن مشروط بدانکه ترافیک اضافی اولویت پایینی داشته باشد و به محض آشکار شدن علائم ازدحام حذف شود. چنین روشی ایده‌بدی نیست چراکه از منابع آزاد سیستم استفاده مفید می‌شود؛ به ماشینهای میزبان اجازه داده‌ایم مادامی که کس دیگر به منابع آزاد نیاز نداشته باشد از آن استفاده کنند ولیکن در شرایط دشوار [بار بالا] حقی برای کسی ایجاد نکرده‌ایم.

تشخیص زودهنگام (Random Early Detection)

بر هر کسی روشن است که رفع ازدحام به محض کشف علائم آن بسیار کارآمدتر از آن است که بگذاریم اوضاع را بهم بریزد و سپس به رفع آن اقدام کنیم. چنین تجربه‌ای بدین ایده منتهی می‌شود که قبل از اشباع شدن کل فضای بافر، مسیریاب به حذف برخی از بسته‌ها اقدام کند. الگوریتم رایجی که در این خصوص کاربرد دارد اصطلاحاً الگوریتم RED (Random Early Detection) نامیده می‌شود. در برخی از پروتکل‌های لایه انتقال (مثل TCP) اگر بسته‌ها در حین انتقال از بین بروند مبداء، نرخ ارسال بسته‌هایش را کاهش می‌دهد. استدلالی که در پشت این منطق نهفته است آن است که TCP در اصل برای شبکه‌های سیمی طراحی شده و از آنجایی که شبکه‌های سیمی بسیار قابل اعتماد و کم خطا هستند لذا دلیل از دست رفتن بسته‌ها اغلب ناشی از پر شدن بافر است تا خطاهای انتقال. از این حقیقت می‌توان برای کمک به کاهش ازدحام بهره گرفت.

دلیل آنکه اجازه می‌دهیم مسیریاب قبل از آنکه وضعیت وخیم شود بسته‌ها را حذف کند آن است که بتوان قبل از دیر شدن کاری انجام داد. برای تعیین زمان شروع حذف بسته‌ها، هر مسیریاب میانگین طول صفهای خود را نگه می‌دارد. هر گاه طول متوسط صف بر روی برخی از خطوط، از حد مجاز تجاوز کرد آن خط با ازدحام مواجه شده و عملیات حذف شروع می‌شود.

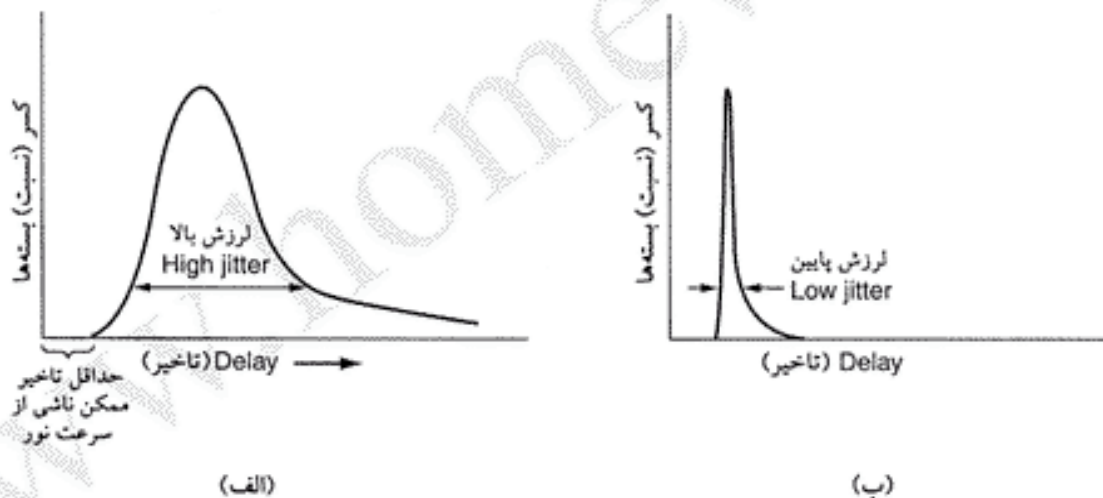
از آنجا که مسیریاب احتمالاً نمی‌تواند بفهمد کدام مبداء باعث مشکل بیشتری شده لذا این ایده که یکی از بسته‌ها را تصادفاً از صف دچار ازدحام بیرون بکشد، می‌تواند ایده خوبی باشد.

مسیریاب چگونه می‌تواند بروز مشکل را به مبداء بسته اعلام کند؟ یک روش آن است که بک «بسته دعوت به آرامش» برای مبداء بسته ارسال شود. اشکال این روش در آن است که بار بیشتری را بر روی شبکه‌ای که از قبل دچار ازدحام شده، تحمیل می‌کند. راهکار دیگر آن است که بسته انتخاب شده را حذف کرده و هیچ گزارشی هم اعلام نکند. در این صورت مبداء بسته از عدم برگشت بسته اعلام وصول (Ack) متوجه حذف بسته شده و اقدام

لازم را صورت می دهد. از آنجایی که مبداء بسته می داند که از بین رفتن بسته ها ناشی از ازدحام و حذف بسته ها است لذا بجای آنکه سخت تر تلاش کند، سرعت خود را کاهش می دهد. این «فیدبک ضمنی» صرفاً زمانی کار می کند که مبداء بسته وقتی متوجه از دست رفتن آن شود سرعت خود را کاهش بدهد. در شبکه های بی سیم که از بین رفتن بسته ها بیشتر ناشی از نویز لینک رادیویی است نمی توان از این روش استفاده کرد.

۶-۳-۵ کنترل لرزش (Jitter Control)

برای کاربردهایی مثل ارسال جریان داده های صدا و تصویر (ویدئو)، این که آیا ۲۰ میلی ثانیه طول می کشد تا بسته ای تحویل مقصد شود یا ۳۰ میلی ثانیه، موضوع مهمی نیست بلکه مهم آنست که این زمان انتقال ثابت باشد. میزان تغییر در زمان رسیدن بسته ها (یا به عبارتی انحراف معیار زمان تحویل بسته ها) اصطلاحاً «لرزش» (Jitter) نامیده می شود. وقتی «میزان لرزش» بالاست یعنی مثلاً وقتی که برخی از بسته ها در ۲۰ میلی ثانیه و برخی دیگر در ۳۰ میلی ثانیه به مقصد می رسند، کیفیت ارسال صدا و تصویر ویدیویی را کاهش می دهد. نمودار «لرزش» در شکل ۵-۲۹ به تصویر کشیده شده است. در این مثال، هر گاه ۹۹ درصد از بسته ها تأخیری بین ۲۴/۵ تا ۲۵/۵ میلی ثانیه داشته باشند، برای ارسال صدا و تصویر احتمالاً مناسب و قابل قبول خواهد بود. البته باید بازه تغییرات تأخیر، قابل تحقق و امکان پذیر باشد. در ضمن باید تأخیر ناشی از انتقال سیگنال با سرعت نور و همچنین حداقل تأخیر عبور از مسیریابها را به حساب آورد ولیکن می توان برخی از تأخیرهای کوچک و غیرقابل پیش بینی را نادیده گرفت.



شکل ۵-۲۹. (الف) لرزش زیاد (ب) لرزش کم.

«لرزش» را می توان با محاسبه میانگین زمان انتقال در هر گام از کل مسیر، محدود کرد؛ وقتی بسته ای به یک مسیریاب می رسد آن مسیریاب بررسی می کند تا ببیند آن بسته به چه مقدار از برنامه زمانی خود جلوتر یا عقب تر است. این اطلاعات در درون هر بسته ذخیره شده و در هر گام بهنگام می شود. اگر بسته از برنامه زمانی خود جلو باشد آنقدر معطل می شود تا به برنامه زمانی خود برگردد ولیکن اگر از برنامه عقب افتاده باشد مسیریاب سعی می کند آنرا به سرعت بر روی خروجی بفرستد.

در حقیقت الگوریتمی که تعیین می کند کدامیک از بسته های منتظر ارسال، برای خروج از یک خط اولویت دارد می تواند بسته ای را انتخاب کند که بیشتر از بقیه، از برنامه زمانی خود عقب افتاده است. بدین ترتیب بسته هایی که از برنامه زمانی خود جلوتر هستند آهسته تر و بسته هایی که از برنامه زمانی خود عقب افتاده اند سریعتر هدایت می شوند و در هر دو حالت میزان لرزش کاهش خواهد یافت.

در برخی از کاربردها مثل ارسال «ویدئو برحسب تقاضا» (Video on-Demand)، خود گیرنده می تواند با بافر

کردن بسته ها و استخراج آنها از بافر جهت نمایش [در زمان مناسب] میزان لرزش را کاهش بدهد (بجای آنکه این کار به صورت بی درنگ به شبکه محول شود). ولیکن در برخی دیگر از کاربردها خصوصاً زمانی که نیاز به محاوره بی درنگ بین مردم باشد (مثل تلفن اینترنتی یا کنفرانسهای ویدیویی از راه دور)، تأخیر ناشی از بافر کردن داده ها، قابل قبول نخواهد بود.

روشهای کنترل ازدحام، حوزه ای فعال برای پژوهش و تحقیق است. پژوهشهای تازه و دستاوردهایی که در این زمینه حاصل شده در مرجع (Gevros et al., 2001) جمع بندی و ارائه شده است.

۴-۵ کیفیت خدمات (Quality of Service)

تکنیکهایی که در بخشهای قبلی مرور کردیم به منظور کاهش ازدحام و بهبود کارایی شبکه طراحی شده بودند. با این حال با رشد شبکه های چند رسانه ای، این ارزیابی های خاص و تمهیدات موردی کفایت نمی کند و به تلاشی جدی در تضمین کیفیت خدمات شبکه و طراحی پروتکل ویژه نیاز است. در بخشهای آتی نیز مطالعات خود را در خصوص کارایی شبکه، ادامه خواهیم داد با این تفاوت که تمرکز ویژه ای بر روی روشهای «تضمین کیفیت خدمات» متناسب با نیاز برنامه های کاربرد، خواهیم داشت. ولی در همین ابتدا باید متذکر شویم که بسیاری از این ایده ها در حال تغییر هستند و عوض خواهند شد.

۴-۵-۱ نیازها

به دنباله ای از بسته ها که از مبدا به سوی مقصد روانه می شوند اصطلاحاً «جریان» (Flow) گفته می شود. در شبکه های اتصال گرا تمام بسته هایی که به یک «جریان» تعلق دارند، مسیر مشابهی را طی می کنند در حالی که در شبکه های بدون اتصال (Connectionless) بسته ها ممکن است از مسیرهای متفاوتی حرکت کنند. نیازهای مطلوب برای هر «جریان» را می توان با چهار پارامتر ابتدایی مشخص کرد: (۱) قابلیت اطمینان (۲) تأخیر (۳) لرزش (Jitter) (۴) پهنای باند. مجموعه این چهار پارامتر، «کیفیت خدمات» یا به اختصار QoS مورد نیاز یک «جریان» را تعیین می کند. در شکل ۵-۳۰ برخی از کاربردهای رایج و سطح نیازمندیهای آنها فهرست شده اند.

نوع کاربرد	قابلیت اطمینان	تأخیر	لرزش	پهنای باند
پست الکترونیکی	بالا	پایین	پایین	پایین
انتقال فایل	بالا	پایین	پایین	متوسط
دسترسی به وب	بالا	متوسط	پایین	متوسط
ورود به سیستم از راه دور	بالا	متوسط	متوسط	پایین
دریافت صوت برحسب تقاضا	پایین	پایین	بالا	متوسط
دریافت ویدیو برحسب تقاضا	پایین	پایین	بالا	بالا
تلفن اینترنتی	پایین	بالا	بالا	پایین
ویدیو کنفرانس	پایین	بالا	بالا	بالا

شکل ۵-۳۰. سطح نیازمندی برنامه های کاربردی به کیفیت خدمات.

چهار کاربرد اول نیاز شدیدی به «قابلیت اطمینان» (Reliability) دارند یعنی هیچ یک از بسته ها نباید اشتباه تحویل شوند. [به عبارت دیگر حتی یک بیت خطا در کل جریان داده ها قابل قبول نیست.] با اضافه کردن کدهای کشف خطا به هر بسته و بررسی آنها در مقصد می توان به این هدف نائل شد. اگر بسته ای در حین انتقال آسیب دید، وصول آن اعلام و تصدیق نمی شود و طبعاً باید از نو ارسال شود. این استراتژی قابلیت اطمینان بالایی برای بسته ها

فراهم می‌کند. چهار کاربرد آخر (صدا / ویدیو) می‌توانند اندکی خطا را تحمل کنند لذا کدهای کشف خطا برای هر بسته محاسبه و بررسی نخواهد شد.

کاربردهای انتقال فایل، شامل پست الکترونیکی یا دریافت تصاویر، حساسیت چندانی به تأخیر ندارند. حتی اگر تمام بسته‌ها به طور یکنواخت تا چند ثانیه هم تأخیر داشته باشند هیچ مشکل حادی پدید نمی‌آید. کاربردهای محاوره‌ای (Interactive) همانند جستجو در وب یا ورود به سیستم از راه دور (Remote Login) حساسیت بیشتری به تأخیر دارند.

کاربردهای بی‌درنگ مثل تلفن یا کنفرانس از راه دور حساسیت شدیدی به تأخیر دارند. اگر کلمات ادا شده در یک تماس تلفنی همگی با 2.000 ثانیه تأخیر برسند. کاربران چنین تماسی را نامناسب و غیرقابل قبول خواهند دانست. برعکس، اجرای فایل‌های صدا یا ویدیو که بر روی یک سرویس دهنده ذخیره شده به تأخیر پایین نیاز ندارند.^۱

سه کاربرد اول حساسیت چندانی به رسیدن بسته‌ها یا فاصله زمانی نامشخص (نسبت به یکدیگر) ندارند؛ به عبارت بهتر لرزش بالا (High Jitter) مشکل چندانی برای این کاربردها ایجاد نمی‌کند. کاربرد چهارم یعنی ورود به سیستم از راه دور (Remote Login) تا حدودی به لرزش حساستر هستند چراکه در اثر لرزش زیاد ممکن است کاراکترها به صورت ناگهانی و نامتعارف بر روی صفحه نمایش ظاهر شوند. تصاویر ویدیویی و خصوصاً صدا شدیداً به «لرزش» حساس هستند. همانگونه که اشاره شد کاربری که در حال تماشای یک نمایش ویدیویی غیرزنده از روی شبکه است و فریمهای تصویر، همگی با 2.000 ثانیه تأخیر می‌رسند مشکلی پدید نخواهد آمد در حالی که اگر زمان انتقال بین ۱ تا ۲ ثانیه متغیر باشد، مشکلات جدی ایجاد خواهد شد. برای «صدا» لرزش چند میلی‌ثانیه‌ای نیز بوضوح شنیده خواهد شد.

در آخر، کاربردها نیاز متفاوتی به پهنای باند دارند: پست الکترونیکی و Remote Login به پهنای باند زیادی نیاز ندارند در حالی که ارسال ویدیو در تمام اشکال [فشرده شده یا غیرفشرده] پهنای باند بسیار زیادی را می‌طلبد. شبکه‌های ATM، «جریان» (Flow) را براساس QoS مورد نیاز در چهار رده وسیع دسته‌بندی کرده است:

۱. ارسال با نرخ ثابت (مثل تلفن از طریق شبکه)
۲. ارسال بی‌درنگ با نرخ متغیر (مثل کنفرانس ویدیویی فشرده شده)
۳. ارسال غیربی‌درنگ با نرخ متغیر (مثل تماشای نمایش غیرزنده از طریق اینترنت)
۴. ارسال با هر نرخ ممکن (برای انتقال فایل) (Available Bit Rate)

این رده‌ها در شبکه‌های دیگر و اهداف دیگر نیز مفید و راهگشا هستند. رده «ارسال با نرخ ثابت» تلاشی است در جهت شبیه‌سازی یک اتصال سیمی با پهنای باند ثابت و تأخیر یکنواخت.

«ارسال با نرخ متغیر» زمانی مفید خواهد بود که تصاویر ویدیویی، فشرده شده باشند و ضریب فشرده سازی فریمهای تصویر با هم فرق کند. بنابراین ارسال یک فریم تصویر با جزئیات و ظرافت زیاد نیاز به ارسال تعداد بیت زیادی دارد در حالیکه ارسال یک فریم تصویر که از یک دیوار سفید گرفته شده ممکن است تا حد بسیار بالایی فشرده شود. رده «ارسال با هر نرخ ممکن» برای کاربردهایی مثل پست الکترونیکی کاربرد دارد که به تأخیر یا لرزش حساس نیستند.

۲-۴-۵ راهکارهای دستیابی به کیفیت خوب خدمات

تا اینجا چیزهایی را در خصوص نیازهای QoS آموخته‌ایم؛ سؤال این است که چگونه به آنها برسیم؟ در همین جا

۱. چراکه مشاهده یک فیلم یا شنیدن یک موسیقی غیرزنده با چند ثانیه تأخیر دائم مشکل حادی ایجاد نمی‌کند. -م

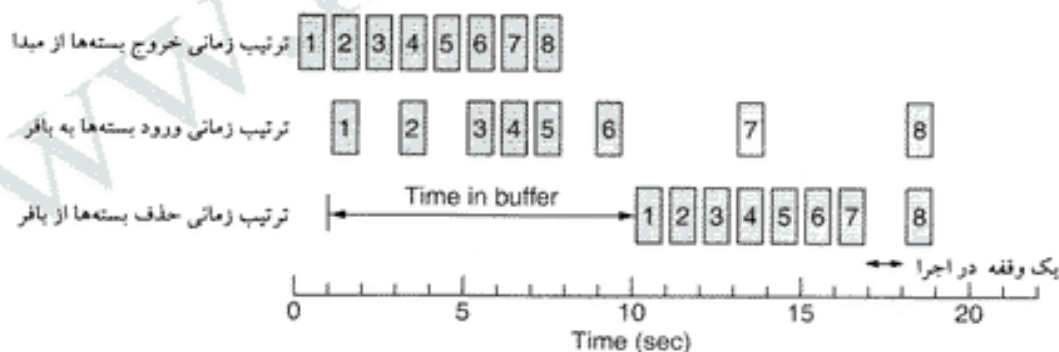
باید اذعان داشت که هیچ راهکار واحد و جادویی برای نیل به این اهداف وجود ندارد. به عبارت دیگر روشی واحد که تمام نیازهای QoS را به صورت بهینه برآورده نماید موجود نیست بلکه راهکارهای گوناگونی تعریف شده که در آنها برای ارائه راه حل های عملی، ترکیبی از چندین روش مختلف بکار گرفته شده است. حال به بررسی برخی از این راهکارها که طراحان سیستم برای تأمین کیفیت خدمات از آنها بهره گرفته اند می پردازیم.

تمهیدات کافی

یک راه حل ساده آن است که برای مسیر یاب ظرفیت پردازش، فضای بافر و پهنای باند زیادتری تدارک دیده شود تا بسته ها بتوانند بسادگی و بسرعت از آنها بگذرند. اشکال این راه حل آن است که گران تمام می شود! به مرور زمان، طراحان ایده های بهتری در خصوص پیش بینی میزان منابع مورد نیاز مطرح می کنند؛ در آن زمان شاید بتوان از این روش در عمل استفاده کرد. از این دیدگاه سیستم تلفن، سیستمی با تمهیدات کافی است و به ندرت گوشی تلفن را برمی دارید و فوراً بوق آزاد آنرا نمی شنوید زیرا ظرفیت این سیستم، بیش از حد معمولی تقاضا در نظر گرفته شده است.

بافر کردن

«جریان» داده ها می تواند در طرف گیرنده و قبل از تحویل به پروسه اصلی، بافر شود. بافر کردن جریان تأثیری در قابلیت اطمینان ندارد، تأخیر را افزایش می دهد ولیکن در عوض میزان لرزش را متعادل تر می کند. برای «صدا و تصویر» برحسب تقاضا^۱، لرزش یک مشکل اساسی است و این روش کمک زیادی به حل آن می کند. تفاوت بین لرزش زیاد و لرزش کم را در شکل ۵-۲۹ مشاهده کردیم. در شکل ۵-۳۱ دنباله ای از بسته ها را می بینید که با «لرزش» قابل توجهی تحویل می شوند. [فرض شده این بسته ها حاوی صدا یا تصویر هستند.] بسته اول در لحظه $t=0$ ارسال و در لحظه $t=1\text{sec}$ تحویل ماشین گیرنده شده است. بسته دوم با تأخیر بیشتری مواجه شده و ۲ ثانیه در راه بوده تا برسد. به محض آنکه این بسته ها از راه می رسند در ماشین گیرنده بافر می شوند.



شکل ۵-۳۱. یکنواخت کردن استریم خروجی یکمک بافرسازی بسته ها.

در $t=10\text{sec}$ اجرای آنها (Playback) شروع می شود. در این لحظه بسته های ۱ تا ۶ بافر شده اند فلذا می توان در فواصل زمانی مشخص و یکنواخت آنها را از بافر استخراج کرد و اجرای متعادل و مناسبی را انتظار داشت. متأسفانه لحظه ای که نوبت به اجرای بسته ۸ فرا می رسد به دلیل تأخیر نامتعارف، در بافر موجود نیست و اجرای آن باید تا زمان رسیدن آن متوقف شود که این مسئله یک توقف نامطلوب در اجرای موزیک یا فیلم ایجاد می کند. این مشکل را می توان با ایجاد تأخیر بیشتر در زمان شروع اجرا کاهش داد ولی در عوض به حجم بافر بزرگتری نیاز

است. سایتهای وب تجاری که دارای صدا یا تصویر هستند از برنامه‌هایی برای اجرای صدا و تصویر استفاده می‌کنند که قبل از شروع اجرا، داده‌ها را تا ۱۰ ثانیه بافر می‌کنند.

شکل‌دهی ترافیک (Traffic Shaping)

در مثال بالا ماشین مبداء، بسته‌ها را در فواصل زمانی یکنواخت بیرون می‌فرستد، در حالی که مواردی وجود دارد که بسته‌ها بطور نامنظم تولید و ارسال می‌شوند و این مسئله ممکن است منجر به بروز ازدحام در شبکه شود. خروجی غیریکنواخت عموماً وقتی است که سرویس دهنده مربوطه، بطور همزمان درگیر ارسال چندین «جریان ویدیویی یا صدا» است و در ضمن امکان عملیات دیگری مثل «جلو و عقب رفتن سریع» (Fast Forward/Rewind) در اجرای صدا و تصویر) و همچنین احراز هویت کاربران و نظایر آنرا نیز فراهم آورده است. از طرفی در برخی از کاربردها مثل کنفرانس از راه دور، بافر کردن داده‌ها بطور کل منطقی نیست.^۱ با این حال اگر بتوان کاری انجام داد تا سرویس دهنده (و کلاً ماشینهای میزبان) مجبور به ارسال با نرخ یکنواخت باشند، کیفیت خدمات (QoS) بهتر خواهد بود. در اینجا به بررسی روشی با عنوان «شکل‌دهی به ترافیک» می‌پردازیم که بجای آنکه بر ماشین گیرنده تمرکز داشته باشد، ترافیک تولیدی در سمت سرویس دهنده را متعادل و یکنواخت می‌کند.

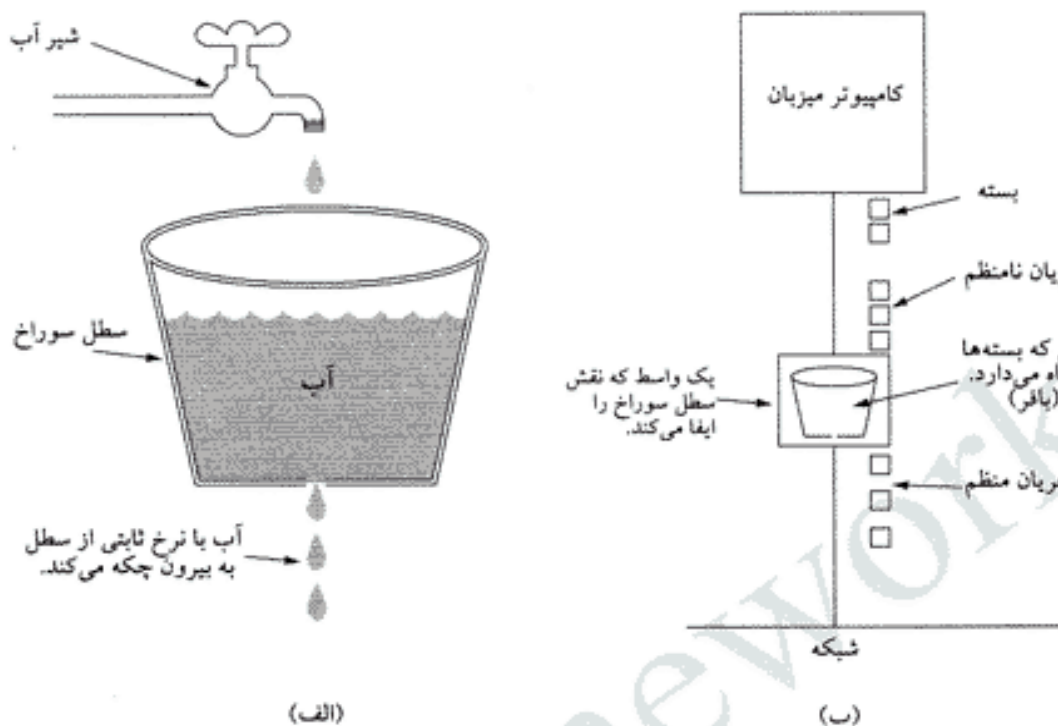
«شکل‌دهی به ترافیک» در خصوص منظم کردن نرخ متوسط (و کاهش حالت انفجارگونه) تولید داده‌های ارسالی است. «پروتکل پنجره لغزان» (Sliding Window Protocol) که قبلاً مطالعه کردیم فقط میزان داده‌هایی را که بطور همزمان ارسال می‌شوند، محدود می‌کند ولی متوسط نرخ ارسال آنها را محدود نمی‌کند. در روش شکل‌دهی به ترافیک، وقتی یک «اتصال» تنظیم می‌شود، کاربر و زیرشبکه (یا به عبارتی مشتری و حامل) بر روی الگوی خاصی از ترافیک (شکل ارسال ترافیک) برای آن مدار توافق می‌کنند. گاهی به این توافق، اصطلاحاً «توافق بر روی سطح خدمات» (Service Level Agreement) گفته می‌شود. مادامیکه مشتری بر طبق این توافق عمل کند و بسته‌ها را براساس قرارداد مورد توافق ارسال نماید، حامل (یعنی زیرشبکه) نیز متعهد است که بسته‌ها را بطور منظم تحویل بدهد. این عمل ازدحام را کاهش داده و اجازه می‌دهد زیرشبکه حامل بتواند برای عمل به تعهدات خود به فعالیت طبیعی ادامه بدهد. چنین توافقی برای عملیاتی مثل انتقال فایل چندان مهم نیست بلکه برای داده‌های بی‌درنگ مثل ارسال صدا و تصویر که شدیداً به کیفیت خدمات نیازمند است، اهمیت حیاتی دارد.

در نتیجه برای عملیات شکل‌دهی به جریان، مشتری به حامل می‌گوید: «الگوی ارسال من بدین نحو است؛ آیا از عهده آن بر می‌آیی؟» اگر زیرشبکه حامل این الگو را پذیرفت، مورد دیگری به میان خواهد آمد: حامل چگونه متوجه می‌شود که آیا مشتری از مفاد توافق پیروی می‌کند؟ و اگر پیروی نکرد چه عکس‌العملی باید نشان بدهد؟ نظارت بر جریان ترافیک اصطلاحاً «اعمال سیاست بر ترافیک» (Traffic Policing) نامیده می‌شود. توافق در خصوص شکل ترافیک و نظارتهای بعدی، در زیرشبکه‌های مبتنی بر مدار مجازی ساده‌تر از زیرشبکه‌های مبتنی بر دیتاگرام است. با این حال حتی در زیرشبکه‌های دیتاگرام، از همین ایده‌ها می‌توان برای اتصالات ایجاد شده در لایه انتقال استفاده کرد.

الگوریتم سطل سوراخ (Leaky Bucket Algorithm)

سطحی را در نظر بگیرید که همانند شکل ۵-۳۲-الف رخنه کوچکی در کف آن وجود دارد. بدون توجه به نرخ ورود آب به این سطل، جریان نشتی از آن ثابت است؛ مادامیکه آب در سطل وجود دارد نرخ خروجی معادل ρ و

۱. بافر کردن مثلاً ۱۰ ثانیه از یک کنفرانس ویدیویی آنرا از حالت زنده خارج خواهد کرد و بهیچوجه مطلوب کاربران نیست. -



شکل ۵-۳۲. (الف) یک سطل سوراخ دار آب (ب) یک سطل سوراخ دار «پسته».

وقتی سطل خالی است معادل صفر است. همچنین اگر آب اضافی به سطل پر وارد شود، از اطراف آن بیرون ریخته و هدر می رود. (یعنی در ظرفی دیگری که زیر این سوراخ قرار گرفته وارد نخواهد شد.)

به نحوی که در شکل ۵-۳۲ ب نشان داده شده از همین ایده می توان برای پسته ها نیز استفاده کرد. بطور مفهومی هر ماشین میزبان از طریق یک کارت واسطه به شبکه متصل شده که این کارت واسطه دارای سطلی سوراخ (یعنی یک صف داخلی بی نهایت) است. اگر پسته ای بخواهد در حالی که صف پر است وارد آن شود، حذف خواهد شد. به عبارت دیگر هرگاه یک یا چند پروسه در ماشین میزبان سعی در ارسال پسته ای نمایند در حالی که تعداد پسته های صف به حداکثر ممکن رسیده باشد، پسته جدید بدون هیچ تشریفاتی حذف خواهد شد. این عملیات می تواند توسط سخت افزار واسطه انجام شود یا آنکه توسط سیستم عامل ماشین میزبان شبیه سازی شود. این الگوریتم برای اولین بار توسط Turner (۱۹۸۶) پیشنهاد و الگوریتم سطل سوراخ نام گرفت. در حقیقت این روش چیزی نیست مگر یک سیستم صف بندی با یک سرویس دهنده واحد که سرویس دهی به عناصر منتظر در صف با نرخ ثابتی انجام می شود.

ماشین میزبان اجازه دارد در هر تیک ساعت یک پسته بر روی شبکه قرار بدهد. این کار نیز می تواند توسط کارت واسطه شبکه یا سیستم عامل ماشین انجام و مدیریت شود. این مکانیزم جریانی نامنظم و بی قاعده از پسته های تولید شده توسط پروسه های کاربری ماشین را به جریانی منظم بر روی شبکه تبدیل کرده و بدین ترتیب حالت انفجارگونه ترافیک، معتدلتر شده و احتمال ازدحام کاهش چشمگیری می یابد.

اگر پسته ها دارای اندازه ای ثابت و یکسان باشند (مثل سلولهای ATM) می توان طبق توضیح بالا از این الگوریتم بهره گرفت: «یک سلول در هر تیک ساعت». ولیکن وقتی اندازه پسته ها متغیر باشد، بهتر آن است که در هر تیک ساعت به جای یک پسته، تعداد بایت ثابتی ارسال شود. بدین ترتیب اگر قرار باشد در هر تیک ساعت ۱۰۲۴ بایت ارسال شود، می توان یک پسته ۱۰۲۴ بایتی یا دو پسته ۵۱۲ بایتی یا چهار پسته ۲۵۶ بایتی و به همین

ترتیب، ارسال کرد. اگر تعداد بایتهای باقیمانده ناچیز باشد، بسته بعدی باید تا تیک ساعت بعدی منتظر بماند. پیاده‌سازی الگوریتم سطل سوراخ (نسخه اصلی آن) ساده است. سطل سوراخ مشکل از یک صف متناهی (محدود) است. هرگاه بسته‌ای دریافت شود و فضای کافی در صف وجود داشته باشد به آخر صف ملحق می‌شود و در غیر این صورت حذف می‌شود. در هر تیک ساعت یک بسته ارسال می‌شود (مگر آنکه صف خالی باشد). «الگوریتم سطل سوراخ مبتنی بر شمارش بایت» نیز تقریباً به همین روش پیاده‌سازی می‌شود. در هر تیک ساعت شمارنده بایت، به مقدار «تنظیم می‌شود. اگر بسته سر صف تعداد بایت کمتری از مقدار فعلی شمارنده داشته باشد، ارسال شده و به تعداد بایتهای بسته از شمارنده کم می‌شود. مادامیکه مقدار شمارنده از بسته‌های موجود در سر صف بیشتر است می‌توان این بسته‌ها را ارسال کرد. وقتی مقدار شمارنده از اندازه بسته بعدی در سر صف کمتر شود، عمل ارسال تا تیک بعدی متوقف می‌شود تا مقدار شمارنده بایت مجدداً به مقدار «تنظیم شده و جریان بسته‌ها ادامه یابد.

به عنوان مثالی از الگوریتم سطل سوراخ، کامپیوتری را در نظر بگیرید که می‌تواند داده‌هایی با سرعت ۲۵ میلیون بایت در ثانیه (معادل 200 Mbps) تولید کند و شبکه نیز با همین سرعت کار می‌کند، ولیکن مسیر یاب فقط می‌تواند چنین نرخ داده‌ای را در یک فاصله زمانی محدود بپذیرد (تا زمانی که بافرهایش پر شود). در زمانهای طولانی بهتر آن است که نرخ ارسال از ۲ میلیون بایت در ثانیه تجاوز نکند.^۱ حال فرض کنید هر ۴۰ میلی‌ثانیه، داده‌ها در یک حالت انفجارگونه و بصورت توده‌های یک میلیون بایتی ارسال شوند. [یعنی هر ۴۰ میلی‌ثانیه یک توده یک میلیون بایتی با حداکثر سرعت یعنی ۲۵ میلیون بایت در ثانیه تولید و ارسال می‌شوند.] برای آنکه نرخ متوسط ارسال به 2MByte/Sec کاهش یابد، باید از «سطل سوراخ» با $\rho = 2\text{MByte/sec}$ و فضای بافر 1MByte بهره گرفته شود. این بدین معناست که توده‌های داده‌ای که انفجارگونه ارسال می‌شوند با طول حداکثر 1MByte قابل دریافت و مدیریت هستند و چیزی از دست نخواهد رفت. این سطل نیز در طول ۵۰۰ میلی‌ثانیه تخلیه می‌شود و اینکه داده‌ها با چه سرعتی وارد می‌شوند اهمیتی ندارد.

در شکل ۵-۳۳ الف ورود داده‌ها به سطل سوراخ در خلال ۴۰ میلی‌ثانیه و با نرخ 25 MByte/Sec صورت گرفته است [معادل ۱ مگابایت داده و متناسب با ظرفیت سطل]. در شکل ۵-۳۳ ب می‌بینیم که این مقدار از داده در خلال ۵۰۰ میلی‌ثانیه و با نرخ 2MByte/sec تخلیه و ارسال شده است.

الگوریتم سطل نشانه‌دار (The Token Bucket Algorithm)

الگوریتم سطل سوراخ الگوی خروجی سختگیرانه و با نرخ میانگین و ثابتی را تحمیل می‌کند و آنکه ترافیک تا چه اندازه انفجاری است برایش مهم نیست.^۲ در بسیاری از کاربردها وقتی توده‌ای انفجاری از داده‌ها می‌رسد بهتر آن است که اجازه بدهیم سرعت تا حدی افزایش یابد؛ بنابراین به الگوریتمی احتیاج داریم که قابلیت انعطاف بیشتری داشته باشد و ترجیحاً هیچ داده‌ای از دست نرود. یکی از این الگوریتمها «الگوریتم سطل نشانه‌دار» (Token Bucket Algorithm) است. در این الگوریتم، سطل سوراخ (یا عبارتی بافر موجود در کارت شبکه یا

۱. یعنی اگرچه ظرفیت ارسال شبکه 25 Mbyte/Sec است ولی مسیر یاب بدلیل محدودیت سرعت، فقط از عهده پردازش 2Mbyte/Sec برمی‌آید و در غیر اینصورت با ازدحام مواجه می‌شود. اگر فرستنده در لحظاتی با نرخ بیشتری ارسال کرد در عوض باید برای لحظاتی متوقف شود. -م

۲. یادآوری می‌کنیم که انفجاری بودن ترافیک بدین معناست که فقط در لحظات کوتاهی حجم انبوهی اطلاعات تولید و ارسال می‌شود و در بخش زیادی از زمان ارسال داده‌ها ناچیز است. فاکتور انفجاری بودن ترافیک را «حاصل تقسیم ماکزیم ترافیک بر میانگین ترافیک» در نظر بگیرید. -م

سیستم عامل)، نشانه ای (اصطلاحاً یک توکن^۱) را نگه می دارد که این توکن در هر ΔT ثانیه یکبار تولید می شود. در شکل ۵-۳۴ الف، سطلی را مشاهده می کنید که در آن سه نشانه (توکن) وجود دارد و همچنین پنج بسته منتظر ارسال هستند. هر گاه بسته ای بخواهد ارسال شود باید ابتدا یک نشانه بدست آورده و آن را از بین ببرد. در شکل ۵-۳۴ ب می بینیم که سه بسته از مجموع پنج بسته جواز خروج گرفته و ارسال شده اند در حالی که دو بسته باقیمانده در انتظار تولید دو نشانه دیگر به سر می برند.

نوع و مکانیزم شکل دهی به ترافیک در «الگوریتم سطل نشانه دار» نسبت به «الگوریتم سطل سوراخ» متفاوت است. الگوریتم سطل سوراخ به ماشینهای یکبار اجازه نمی دهد که وقتی یکبار هستند سهمیه ارسال خود را نگه داشته و از این سهمیه به یکباره جهت ارسال انفجارگونه بسته ها استفاده کنند. در طرف مقابل، الگوریتم سطل نشانه دار اجازه می دهد که ماشین سهمیه خود را تا حداکثر حجم سطل (n بسته) پس انداز کنند. این ویژگی بدین معناست که توده ای انفجاری از بسته ها را (تا حداکثر n بسته) می توان یکجا فرستاد. [البته به شرط پس انداز تعداد n نشانه]. این ویژگی اجازه می دهد که خروجی بتواند تا حدودی حالت انفجارگونه داشته باشد و هنگامی که ورود بسته ها انفجاری است، پاسخ سریعتری داده شود.

تفاوت دیگر این دو الگوریتم آن است که الگوریتم سطل نشانه دار در هنگامی که سطل پر شود نشانه ها را دور می اندازد (یا به عبارتی ظرفیت ارسال را) در حالی که هرگز بسته ها را حذف نمی کند. در مقابل، الگوریتم سطل سوراخ به محض پر شدن سطل، بسته ها را حذف می نماید.

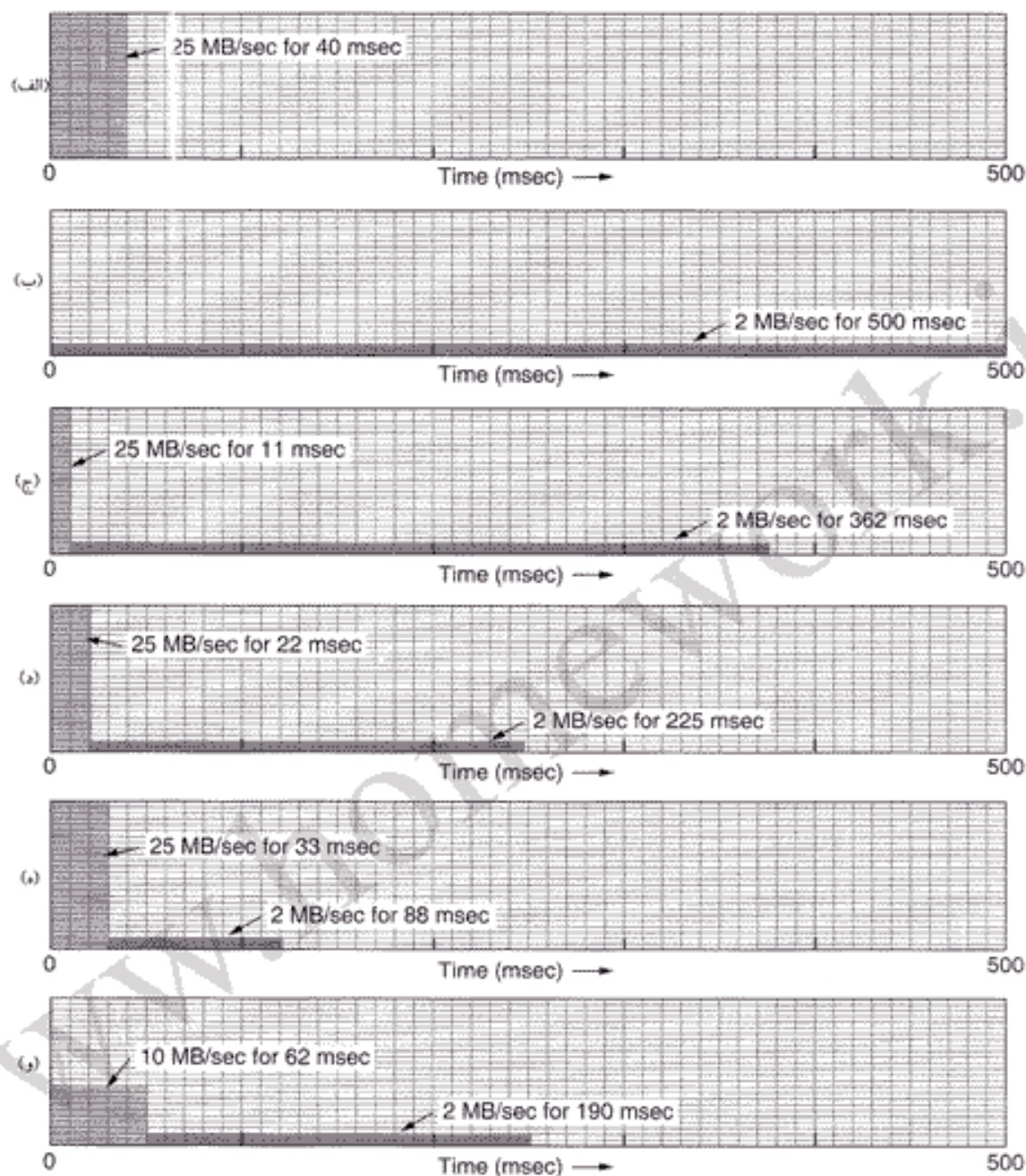
در اینجا یک تغییر کوچک ممکن است و آن اینکه هر نشانه (توکن) مجوز ارسال k بایت تلقی شود نه ارسال یک بسته. در این حالت یک بسته را فقط زمانی می توان ارسال کرد که به تعداد کافی نشانه (معادل با اندازه بسته) پس انداز شده باشد. نشانه هایی که سهمیه ارسال آنها (برحسب بایت) کوچکتر از اندازه بسته فعلی است برای استفاده بعدی نگه داشته می شود.

الگوریتم سطل سوراخ و الگوریتم سطل نشانه دار را می توان به همان نحوی که برای منظم کردن ترافیک خروجی ماشینهای میزبان بکار می رود برای متعادل کردن ترافیک بین مسیریابها نیز استفاده کرد. با این حال یک تفاوت بدیهی بین این دو کاربرد وجود دارد و آن هم اینکه الگوریتم سطل نشانه دار می تواند یک ماشین میزبان را وادار به توقف ارسال نماید، در حالی که وادار کردن یک مسیریاب به توقف (در حالی که بافرهای ورودی آن پر است)، می تواند به از دست رفتن داده ها بینجامد.

پیاده سازی الگوریتم سطل نشانه دار، به سادگی تعریف یک متغیر است تا این متغیر تعداد نشانه ها (توکنها) را بشمارد. این شمارنده در هر ΔT ثانیه یک واحد افزایش داده شده و با ارسال یک بسته، کاهش می یابد. وقتی این شمارنده به صفر برسد هیچ بسته ای را نمی توان ارسال کرد. در گونه دیگر این الگوریتم یعنی روش شمارش بایت، شمارنده هر ΔT ثانیه k واحد (k بایت) افزایش یافته و با ارسال بسته به اندازه طول آن از شمارنده کسر می شود.

اصولاً کاری که الگوریتم سطل نشانه دار انجام می دهد آن است که اجازه ارسال انفجارگونه بسته ها را صادر می کند ولیکن با سقف حداکثر و تنظیم شده ای که قابل کنترل و اداره باشد. برای مثال به شکل ۵-۳۳ ج دقت کنید. در اینجا یک سطل نشانه دار با ظرفیت ۲۵۰ کیلوبایت در اختیار داریم. نشانه ها (توکنها) متناسب با نرخ ۲ مگابایت بر ثانیه تولید می شوند. [یعنی نشانه ها با نرخی تولید می شوند که فقط اجازه ارسال ۲ مگابایت اطلاعات در ثانیه را صادر می کنند. -م] فرض کنید هنگامی که یک توده انفجارگونه ۱ مگابایتی تولید می شود، سطل پر است. سطل، قادر است بمدت حدود یازده میلی ثانیه و با سرعت 25MByte/sec، داده های خود را خالی کند. از آن به بعد

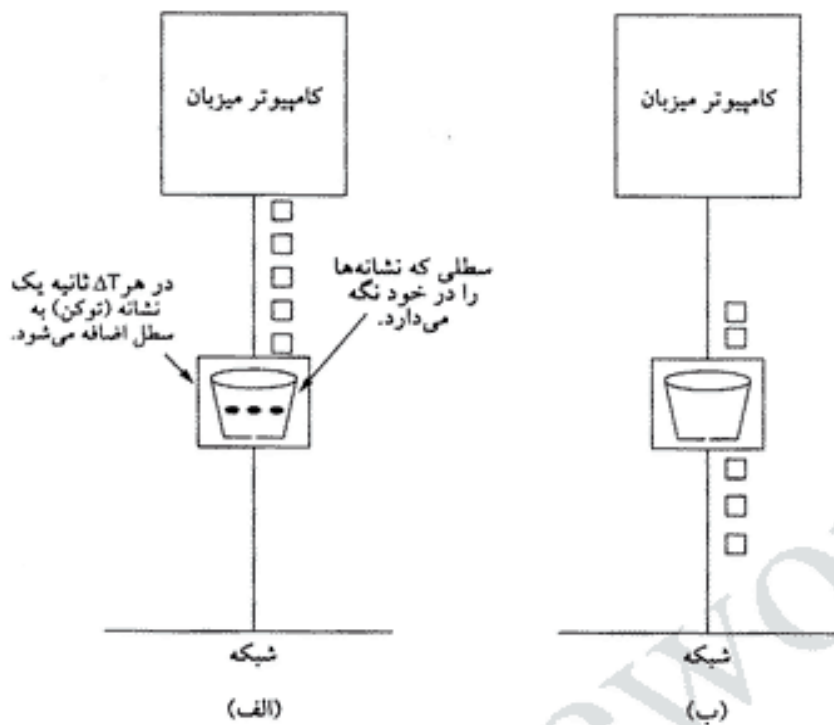
۱. نشانه یا «توکن» را در اینجا جواز یا سهمیه ارسال تعداد مشخصی بایت در نظر بگیرید. -م



شکل ۵-۳۳. (الف) ورودی به یک سطل سوراخ‌دار. (ب) خروجی از سطل سوراخ‌دار. خروجی از یک سطل نشانه‌دار با ظرفیت: (ج) 250 KByte (د) 500 KByte (ه) 750 KByte. (و) خروجی یک سطل نشانه‌دار با ظرفیت 500KByte به یک سطل سوراخ‌دار با نرخ خروجی 10-Mbps اعمال شده است.

مجبور است تا وقتی که توده انفجاری داده ارسال نشده سرعت خروجی را بر روی 2MB/sec ثابت نگه دارد.^۱ محاسبه طول زمانی که می‌توان بصورت انفجاری و با نرخ حداکثر، توده تولید شده را ارسال کرد، اندکی پیچیده است. یعنی سقف حداکثر زمان ارسال انفجارگونه، معادل تقسیم «سهمیه موجود در سطل» بر «نرخ ارسال»

۱. یادآوری می‌کنیم که در الگوریتم سطل نشانه‌دار اگر بسته‌ها بیش از ظرفیت (سهمیه) موجود در سطل تولید شوند، حذف نخواهند شد.



شکل ۵-۳۴. الگوریتم سطل نشانه دار. (الف) قبل از ورود بسته (ب) بعد از خروج بسته.

نیست چرا که در خلال ارسال این توده از بسته ها، نشانه های جدیدی تولید می شوند. اگر طول زمان ارسال انفجاری بسته ها را S بنامیم، ظرفیت موجود در سطل نشانه دار C بایت، نرخ تولید نشانه ρ بایت بر ثانیه و حداکثر نرخ خروجی M bytes/sec باشد، حداکثر نرخ ارسال انفجارگونه خروجی $C + \rho.S$ بایت است.^۱ همچنین می دانیم که اگر طول زمان ارسال، S ثانیه و حداکثر نرخ ارسال M بایت بر ثانیه باشد، تعداد بایتی که در این زمان ارسال می شود $M.S$ بایت خواهد بود. بنابراین داریم:

$$C + \rho.S = M.S$$

با حل این معادله بدست می آوریم:

$$S = \frac{C}{M - \rho}$$

در مثال فوق با پارامترهای $C=250$ KByte، $\rho=2$ MByte/sec و $M=25$ MByte/sec حداکثر زمان ارسال انفجارگونه بسته ها یازده میلی ثانیه بدست خواهد آمد. شکل های ۵-۳۳-د و ۵-۳۳-ه سطل نشانه دار را برای ظرفیت ۵۰۰ کیلوبایت و ۷۵۰ کیلوبایت نشان می دهد.

مشکل بالقوه الگوریتم سطل نشانه دار آن است که کماکان اجازه می دهد توده ای از بسته ها به صورت انفجارگونه تولید و ارسال شود (حتی اگر طول زمان تولید به دقت و براساس انتخاب صحیح ρ و M تنظیم شده باشد). اغلب مطلوب آن است که نرخ حداکثر تولید بسته ها کاهش یابد ولی بدون آنکه بخواهیم به مقدار کم و ثابت

۱. یادآوری می کنیم که اگر سهمیه موجود در سطل C بایت باشد، می توان C بایت از داده ها را با نرخ حداکثر و بصورت انفجاری ارسال کرد و پس از آن باید نرخ ارسال را متناسب با نرخ تولید نشانه کاهش داد. در خلال ارسال انفجاری این C بایت که به مدت S ثانیه طول می کشد تعدادی نشانه جدید (سهمیه جدید) تولید می شود و اجازه می دهد حجم ارسال انفجاری اندکی افزایش یابد. این حجم معادل $C + \rho.S$ بایت است: C بایت سهمیه قبلی و $\rho.S$ بایت سهمیه جدید تولید شده در زمان ارسال. -م.

الگوریتم سطل سوراخ بسنده نماییم.

یک روش برای متعادل کردن ترافیک آن است که بعد از «سطل نشانه دار» یک «سطل سوراخ» قرار داده شود. نرخ خروجی سطل سوراخ باید بیشتر از نرخ خروجی سطل نشانه دار (یعنی مقدار ρ) انتخاب شود ولی این مقدار باید از حداکثر نرخ مجاز شبکه کمتر باشد. شکل ۵-۳۳ و خروجی یک سطل نشانه دار با ظرفیت ۵۰۰ کیلوبایت که پس از آن یک سطل سوراخ 10MByte/sec قرار داده شده را نشان می دهد.

نظارت بر این روشها ممکن است اندکی پیچیده باشد. اصولاً شبکه باید این الگوریتم را شبیه سازی کند و مطمئن گردد که بسته یا بایتهای بیشتر از حد مجاز و مشخص ارسال نمی شود. در هر حال این ابزارها روشهایی برای شکل دهی به ترافیک شبکه به گونه ای قابلیت مدیریت است تا بتوان نیازهای QoS (کیفیت خدمات) را برآورده نمود.

رزرو منابع (Resource Reservation)

توانایی شکل دهی و تنظیم ترافیک ارسالی، تمهید خوبی برای تضمین «کیفیت خدمات» (QoS) محسوب می شود ولیکن استفاده از این روشها زمانی کارآمد خواهد بود که تمام بسته ها از مسیر یکسانی عبور کنند. پراکندگی تصادفی بسته ها بر روی مسیرهای متفاوت، تضمین هر چیزی را بسیار دشوار می کند. بنابراین برای تامین کیفیت خدمات باید بین مبداء و مقصد چیزی شبیه به یک مدار مجازی ایجاد و تنظیم شود و تمام بسته های یک «جریان» از این مسیر حرکت کنند.

هر گاه برای جریان داده ها، مسیر ویژه داشته باشیم می توان منابع لازم را در طول این مسیر، رزرو کرده و موجود بودن ظرفیت مورد نیاز را تضمین کرد. سه نوع متفاوت از منابع را می توان از قبل رزرو کرد:

۱. پهنای باند

۲. فضای بافر

۳. سیکل های CPU [ظرفیت پردازش مورد نیاز]

رزرو پهنای باند آشکارترین مورد است: اگر یک جریان به 1Mbps پهنای باند نیاز داشته باشد و ظرفیت خط خروجی 2Mbps باشد تلاش برای هدایت سه جریان همزمان بر روی این خط عملی نیست. لذا رزرو پهنای باند به معنای آن نیست که می توان یک خط خروجی را بیش از ظرفیت آن رزرو کرد.

دومین منبع که اغلب با محدودیت مواجه است، فضای بافر می باشد. وقتی یک بسته دریافت می شود معمولاً توسط سخت افزار در حافظه کارت شبکه ذخیره می گردد. بعداً نرم افزار مسیریاب باید آن را به یک بافر در فضای RAM اصلی منتقل کرده و پس از پردازش و تعیین مسیر، آنرا در صف مربوط به خط خروجی مستخب، وارد نماید. اگر هیچ بافری موجود نباشد، بسته دریافتی حذف خواهد شد چرا که فضایی برای ذخیره آن موجود نیست. برای تضمین کیفیت خوب خدمات، باید برای هر «جریان» مشخص مقداری بافر کنار گذاشت تا لازم نباشد بسته های آن جریان برای بدست آوردن فضای بافر با دیگر بسته ها رقابت کنند. در این صورت به محض نیاز به بافر، فضای لازم در اختیار خواهد بود (البته تا حد معقول و مشخص).

در آخر، «سیکل های CPU» نیز منبع کمیابی است: پردازش یک بسته به اندازه معینی وقت مسیریاب را می گیرد لذا هر مسیریاب قادر است در یک ثانیه، تعداد مشخص و محدودی از بسته ها را پردازش کند. برای آنکه بتوان مطمئن شد که بسته ها در روالی منظم و به موقع پردازش می شوند باید مراقب بود که بار CPU بیش از حد زیاد نشود.

در نگاه اول شاید به نظر برسد که اگر پردازش هر بسته مثلاً یک میکروثانیه طول بکشد، پس مسیریاب

می تواند یک میلیون بسته را در هر ثانیه پردازش و هدایت نماید. چنین نتیجه گیری غلط است زیرا به دلیل تغییرات آماری بار [یعنی ثابت نبودن میزان بار]، پردازنده در دوره هایی از زمان بیکار می ماند. اگر CPU موظف باشد که در هر سیکل، کار پردازش را شروع و تکمیل کند، حتی از دست رفتن چند سیکل (ناشی از بیکاریهای مقطعی) انباشته شده و قابل جبران نخواهد بود.

با این حال حتی وقتی میزان بار کمتر از ظرفیت تنور یک مسیریاب باشد باز هم ممکن است صف تشکیل و تأخیر ایجاد شود. وضعیتی را در نظر بگیرید که در آن بسته ها بطور کاملاً تصادفی و با نرخ متوسط λ بسته بر ثانیه وارد می شوند. زمان مورد نیاز برای پردازش هر بسته نیز تصادفی است و بطور متوسط μ بسته در هر ثانیه پردازش می شود. با این فرض که تابع توزیع دریافت بسته ها و همچنین پردازش آنها «پواسون» باشد می توان به کمک «نظریه صف» (Queuing Theory) ثابت کرد که متوسط تأخیری که هر بسته با آن مواجه خواهد شد (یعنی T) معادل است با:

$$T = \frac{1}{\mu} \times \frac{1}{1-\lambda/\mu} = \frac{1}{\mu} \times \frac{1}{1-\rho}$$

که در آن ρ معادل است با λ/μ و از آن به عنوان «میزان بهره وری CPU» (CPU Utilization) یاد می شود. فاکتور $1/\mu$ ، زمان لازم برای پردازش یک بسته است. (هنگامی که رقابت و صف وجود ندارد). فاکتور $(1-\rho)$ ، $1/\mu$ میزان کاهش سرعت پردازش، در اثر تشکیل صف (رقابت) تلقی می شود. به عنوان مثال اگر $950000 \text{ Packet/sec} = \lambda$ و $1,000,000 \text{ Packet/sec} = \mu$ باشد میزان بهره وری CPU یعنی ρ معادل با 95% است و متوسط تأخیر بسته ها بجای 1 میکروثانیه، 20 میکروثانیه خواهد بود. این زمان در برگیرنده زمان انتظار در صف و زمان سرویس دهی [زمان پردازش] بسته است. اگر مثلاً سی مسیریاب در مسیر جریان بسته ها قرار گرفته باشد زمان تأخیر انتظار صف در کل مسیر پهنایی معادل با 600 میکروثانیه است.

کنترل پذیرش (Admission Control)

حال در مرحله ای هستیم که ترافیک ورودی از یک «جریان» (Flow) خاص به خوبی شکل و نظم داده شده و بسته ها از یک مسیر واحد حرکت می کنند و پیشاپیش ظرفیت مورد نیاز در طول مسیر، پیش بینی و رزرو شده است. با چنین فرضی، هر گاه جریانی از بسته ها به یک مسیریاب تسلیم شود براساس ظرفیت موجود خود و سطح تعهداتی که در خصوص دیگر جریانها پذیرفته، باید در خصوص قبول یا رد آن تصمیم بگیرد.

تصمیم گیری در خصوص پذیرش یا رد یک «جریان»، یک مقایسه ساده بین میزان پهنای باند، بافر و سیکلهای CPU مورد نیاز و ظرفیت باقیمانده در مسیریاب نیست. این تصمیم گیری اندکی پیچیده تر از این مقایسه ساده است. اگرچه برخی از برنامه های کاربردی ممکن است میزان نیاز خود به پهنای باند را بدانند ولیکن آگاهی کمی از بافر یا حجم پردازش مورد نیاز بسته ها در مسیریابهای واقع بر روی مسیر دارند لذا حداقل می توان گفت که به روشی متفاوت برای توصیف و ارزیابی جریان نیاز است. در ثانی، برخی از کاربردها در مقابل سرآورده نشدن انتظاراتشان، تحمل بیشتری از خود نشان می دهند. در آخر آنکه ممکن است برخی از کاربردها بخواهند که در خصوص «پارامترهای جریان» چانه زنی و مذاکره کنند در حالی که امکان دارد برخی دیگر از کاربردها چنین خصوصیتی نداشته باشند. به عنوان مثال یک برنامه نمایش دهنده فیلم که معمولاً 30 فریم تصویر در ثانیه را نمایش می دهد ممکن است بخواهد در صورت عدم وجود پهنای باند کافی برای این حجم از داده، تعداد فریمهای تصویر را تا 25 Frame/sec کاهش بدهد. به همین ترتیب ممکن است تعداد نقاط تصویر در هر فریم (Pixel Per Frame)، پهنای باند صدا و دیگر ویژگیها نیز قابل تغییر باشد.

چونکه برای رسیدن به توافق نهایی در خصوص تامین نیازهای یک «جریان»، باید مولفه های متعددی در

مذاکرات شرکت داشته باشند (اعم از فرستنده، گیرنده و تمام مسیربایهای واقع بر روی مسیر)، لذا هر «جریان» باید برحسب پارامترهای مشخصی بدقت توصیف شود تا بتوان بر روی این پارامترها مذاکره و توافق کرد. مجموعه چنین پارامترهایی اصطلاحاً «مشخصات توصیفی جریان» (Flow Specification) نامیده می شود. بدین ترتیب یک فرستنده (مثل سرویس دهنده ویدیو) مشخصات توصیفی جریان را به صورت پارامترهای پیشنهادی و مورد نظر خود تعریف می نماید. این پارامترهای پیشنهادی در طول مسیر منتشر می شود و هر مسیربای واقع بر مسیر آنها را بررسی کرده و در صورت نیاز در آنها تغییراتی ایجاد می کند. این تغییرات فقط کاهش است نه افزایشی (یعنی مثلاً نرخ مورد نظر ارسال داده ها را کاهش می دهد نه افزایش). وقتی این پارامترها به طرف مقابل برسد، به اجرا گذاشته می شوند.

در مثال شکل ۵-۳۵، نمونه ای از پارامترهای توصیف جریان، مبتنی بر RFC 2210 و RFC 2211 نشان داده شده است. در اینجا پنج پارامتر وجود دارد که اولین آنها پارامتر Token Bucket Rate (نرخ تولید نشانه در الگوریتم سطل نشانه دار) است و تعداد بایتهایی را مشخص می کند که در هر ثانیه به سطل وارد می شوند. در حقیقت این پارامتر حداکثر نرخ ارسال مجاز و قابل قبولی است که فرستنده می تواند ارسال نماید و میانگین ارسال در یک زمان طولانی تلقی می شود.

واحد	پارامترهای توصیف جریان
Bytes/sec	Token bucket rate
Bytes	Token bucket size
Bytes/sec	Peak data rate
Bytes	Minimum packet size
Bytes	Maximum packet size

شکل ۵-۳۵. مثالی از توصیف جریان.

پارامتر دوم، حجم سطل را برحسب بایت مشخص می کند. به عنوان مثال اگر پارامتر Token Bucket Rate یک مگابایت در ثانیه و حجم سطل 500KByte باشد حتی در صورت توقف ارسال، این سطل می تواند تا چهار ثانیه پر شود و پس از آن سرریز خواهد شد. [سطلی با گنجایش ۵۰۰ کیلوبایت معادل ۴ مگابایت، می تواند جریانی از داده ها با نرخ 1Mbps را بمدت ۴ ثانیه بافر کند. -م]

پارامتر سوم یا Peak Data Rate (حداکثر نرخ ارسال) بیانگر حداکثر نرخ قابل تحمل حتی در بازه های زمانی کوتاه است. فرستنده هیچگاه نباید از این نرخ تجاوز نماید.

دو پارامتر آخر مقدار حداقل و حداکثر طول بسته ها را مشخص می نماید. (این طول شامل سرآیند بسته لایه شبکه و لایه انتقال است). حداقل طول بسته ها نیز اهمیت دارد چراکه پردازش هر بسته (حتی اگر کوچک باشد) مدت زمان ثابتی طول می کشد. یک مسیربای ممکن است آمادگی پذیرش و هدایت ۱۰,۰۰۰ بسته یک کیلوبایتی در هر ثانیه را داشته باشد ولیکن آمادگی پذیرش و هدایت ۱۰۰,۰۰۰ بسته ۵۰ بایتی در ثانیه را نداشته باشد، هر چند که مورد دوم از لحاظ حجم داده کمتر از مورد اول است. طول حداکثر هر بسته نیز به دلیل محدودیتهای درونی شبکه اهمیت دارد و نباید از این مقدار حداکثر تجاوز شود. به عنوان مثال اگر بخشی از مسیر از درون شبکه اترنت بگذرد، حداکثر طول بسته نباید بیش از ۱۵۰۰ بایت باشد [به دلیل معماری سخت افزار اترنت].

یک سؤال جالب آن است که یک مسیربای با استفاده از مشخصات توصیفی جریان، چگونه می تواند منابع مورد نیاز را رزرو نماید. نگاشت «مشخصات توصیفی جریان» به میزان منابع مورد نیاز، در مقوله پیاده سازی

می گنجد و استاندارد سازی نشده است. فرض کنید که یک مسیریاب قادر به پردازش صد هزار بسته در ثانیه باشد. اگر «جریان» پیشنهادی با نرخ 1MB/sec و مقدار حداقل و حداکثر طول بسته ها ۵۱۲ بایت باشد، مسیریاب بدین نتیجه می رسد که از این جریان در هر ثانیه ۲۰۴۸ بسته دریافت خواهد کرد و بدین منظور ۲ درصد از CPU خود را برای پردازش این بسته ها در نظر می گیرد. ترجیحاً این مقدار باید بیشتر هم باشد تا از تأخیرات ناشی از ایجاد صف های طولانی اجتناب شود. اگر سیاست های یک مسیریاب بر این اساس باشد که هیچگاه بیش از ۵۰ درصد از وقت CPU خود را اختصاص ندهد (یعنی تلویحاً تأخیرات را دو برابر فرض کرده است) و قبلاً ۴۹ درصد از این وقت رزرو شده باشد نمی تواند یک «جریان» با مشخصات فوق را بپذیرد [چرا که به دو درصد از CPU احتیاج دارد در حالی که فقط یک درصد باقیمانده است]. برای منابع دیگر [مثل حافظه و پهنای باند] نیز محاسبات و پیش بینی های مشابه فوق انجام می شود.

توصیف دقیقتر «جریان»، برای مسیریابها مفیدتر خواهد بود. اگر در توصیف یک «جریان» بیان شود که پارامتر Token Bucket Rate (نرخ تولید نشانه در الگوریتم سطل) معادل 5MB/sec است ولیکن طول بسته ها بین ۵۰ تا ۱۵۰۰ بایت متغیر اعلام شود در این توصیف نادقیق، نرخ ارسال بسته ها بین ۳۵۰۰ تا ۱۰۵,۰۰۰ بسته در ثانیه متغیر خواهد بود و ممکن است مسیریاب از عدد ۱۰۵۰۰۰ نگران شده و چنین جریانی را نپذیرد در حالی که اگر مقدار حداقل طول بسته ۱۰۰۰ بایت پیشنهاد شود احتمالاً این جریان پذیرفته خواهد شد.

مسیریابی نسبی

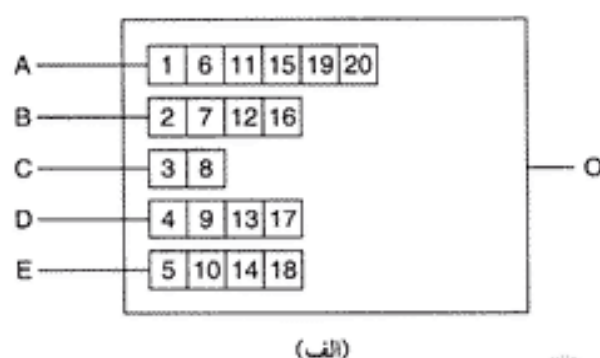
در اغلب الگوریتم های مسیریابی سعی بر آن است که بهترین مسیر به هر مقصد پیدا شود و پس از آن تمام ترافیک به یک مقصد از مسیر بهینه ارسال خواهد شد. راهکار دیگر برای ارائه کیفیت بهتر خدمات آنست که ترافیک بسته های ارسالی برای یک مقصد، از چندین مسیر هدایت و ارسال شود. از آنجایی که مسیریابها دید جامعی از ترافیک سرتاسر شبکه ندارند لذا تنها راه ممکن برای توزیع ترافیک بر روی چندین مسیر، استفاده از اطلاعات موجود و محلی است. یک روش ساده آن است که ترافیک بطور مساوی یا به تناسب ظرفیت هر یک از خطوط خروجی مسیریاب، بر روی آنها توزیع شود. با این حال الگوریتم های پیچیده تری در این خصوص وجود دارد. (Nelakuditi and Zhang, 2002)

زمان بندی بسته ها

هر گاه یک مسیریاب هدایت چندین «جریان» را بر عهده داشته باشد این خطر وجود دارد که یک «جریان» از حدود و ظرفیت مجاز خود تجاوز نماید و در نتیجه جریانه های دیگر را با کمبود منابع (starvation) مواجه سازد. اگر پردازش بسته ها به ترتیب ورودشان انجام گیرد باعث می شود که یک فرستنده متجاوز بتواند بیشتر ظرفیت مسیریابی را که بر روی خط سیر بسته های او هستند اشغال کرده و کیفیت خدمات دیگران کاهش یابد. برای خنثی کردن چنین تلاشی، الگوریتم هایی جهت زمان بندی بسته ها پیشنهاد شده است. (Bhatti and Crowcroft, 2000)

یکی از اولین روشها، الگوریتم «صف بندی بی طرفانه» (Fair Queuing) است. (Nagle, 1987) جوهره این الگوریتم آنست که مسیریابها باید برای هر خط خروجی و به ازای هر «جریان» که از آن خط خروجی می گذرد، صف های جداگانه ای تشکیل بدهند. هر گاه خطی بیکار شود، مسیریاب صف ها را به ترتیب پویش کرده و از سر هر صف یکی را بر می دارد. بدین ترتیب، در شرایطی که n ماشین میزبان برای یک خط خروجی رقابت می کنند، از هر n بسته ارسالی بر روی خط یک بسته به هر ماشین میزبان تعلق می گیرد. افزایش نرخ ارسال بسته ها، در نسبت سهم هر ماشین تغییری ایجاد نخواهد کرد.

البته در همین ابتدا، الگوریتم فوق دارای یک مشکل است: آن ماشینهای میزبان که طول بسته‌هایشان بزرگ است نسبت به ماشینهایی که بسته‌های کوچک تولید می‌کنند، سهم بیشتری از پهنای باند را بخود اختصاص خواهند داد. پژوهشگری به نام Demer و همکاران او (۱۹۹۰) پیشنهادی جهت بهبود این الگوریتم ارائه دادند. پیشنهاد آن بود که بجای ارسال یک بسته از هر صف به صورت نوبت چرخشی (Round Robin)، سهم ارسال هر صف بر حسب بایت باشد. در نتیجه صفها بطور متوالی و بایت به بایت پویش شده و بسته‌های هر صف بر حسب زمان خاتمه ارسالشان مرتب شده و بهمان ترتیب ارسال می‌شوند. یعنی به جای آنکه از هر صف فقط یک بسته انتخاب شود تعدادی بایت و متناسب با سهم زمانی هر صف ارسال می‌شود. این الگوریتم در شکل ۵-۳۶ به تصویر کشیده شده است.



(الف)

بسته	زمان خاتمه ارسال
C	8
B	16
D	17
E	18
A	20

(ب)

شکل ۵-۳۶. (الف) مسیریابی که در آن پنج بسته برای خروج از خط O به صف شده‌اند. (ب) زمان خاتمه ارسال این پنج بسته.

در شکل ۵-۳۶ الف بسته‌هایی به طول ۲ تا ۶ بایت می‌بینیم. در هر تیک ساعت (مجازی) اولین بایت از بسته دریافتی از خط A ارسال می‌شود. در تیک بعدی اولین بایت از بسته دریافتی از خط B ارسال می‌گردد و کار به همین ترتیب ادامه می‌یابد. اولین بسته‌ای که ارسال آن پس از هشت تیک ساعت خاتمه خواهد یافت بسته C است.^۱ در شکل ۵-۳۶ ب فهرست مرتب شده بسته‌ها به ترتیب ارسال مشخص شده است. هرگاه بسته جدیدی دریافت نشود بسته‌ها به ترتیب فوق‌الذکر (از C تا A) ارسال خواهد شد.

یک اشکال این الگوریتم آن است که به تمام ماشینهای میزبان، اولویت یکسانی می‌دهد. در بسیاری از محیطها مطلوبتر آن است که به سرویس دهنده‌های ویدیو (Video Server) اولویت بیشتری نسبت به یک سرویس دهنده معمولی فایل داده شود و در هر تیک ساعت، سهم آن دو یا چند بایت باشد. این الگوریتم اصلاح شده به نام «الگوریتم صف‌بندی بی‌طرفانه وزن‌دار» (Weighted Fair Queuing) مشهور است و کاربرد گسترده‌ای دارد. گاهی اوقات وزن هر صف معادل با تعداد «جریان» (Flow) منشعب از یک ماشین در نظر گرفته می‌شود و بدین ترتیب هر پروسه مولد جریان، پهنای باند یکسانی دریافت می‌دارد.^۲ روش پیاده‌سازی مؤثر این الگوریتم در مرجع (Shreedhar & Varghese, 1995) تشریح شده است. امروزه در عمل، هدایت بسته‌ها توسط سخت‌افزار مسیریاب یا سوئیچ انجام می‌شود و الگوریتمهای فوق بر روی سخت‌افزاری پیاده‌سازی شده‌اند. (Elhanany et al., 2001)

۱. دقت کنید که شماره‌هایی که درون مربعهای هر بسته از شکل ۵-۳۶ نوشته شده شماره ترتیب ارسال تلقی می‌شود و هر مربع شماره‌دار صرفاً معادل یک بایت است. -م
 ۲. به عبارت دیگر به جای آنکه به ماشینهای میزبان پهنای باند مساوی داده شود به پروسه‌های هر ماشین پهنای باند یکسان داده می‌شود. -م

۳-۴-۵ خدمات مجتمع (Integrated Services)

در خلال سالهای ۱۹۹۵ تا ۱۹۹۷، تلاش IETF بر آن بود که برای انتقال داده های مالتی مدیا (Multimedia Streaming) معماری مناسبی ابداع کند. نتیجه کار چندین RFC به شماره های ۲۲۰۵ تا ۲۲۱۰ بود. این پروژه با نام کلی «الگوریتمهای مبتنی بر جریان» (Flow-based algorithms) یا «خدمات مجتمع» (Integrated Services) شناخته می شود و کاربردهای چندپخش (Multicast) و تک پخش (Unicast) را در بر می گیرد. به عنوان یک نمونه از کاربردهای تک پخش، کاربری را در نظر بگیرید که قطعه ای ویدیو را از یک سایت تماشا می کند. به عنوان مثالی از کاربردهای چندپخش، ایستگاههای پخش تلویزیون دیجیتال را در نظر بگیرید که برنامه های خود را در قالب جریانی از بسته های IP به گیرندگان بی شمار و پراکنده خود ارسال می دارند. در ادامه بر روی کاربردهای چندپخش متمرکز خواهیم شد چرا که کاربردهای تک پخش حالت خاص چندپخش هستند. در اغلب کاربردهای چندپخش، گروههای مختلف می توانند به صورت پویا عضویت خود را تغییر بدهند: مثلاً افرادی را مجسم کنید که در یک کنفرانس ویدیویی وارد می شوند و پس از مدتی حوصله آنها سر می رود و به یک کانال پخش آواز می پیوندند! در چنین شرایطی روش رزرو پهنای باند به خوبی کار نخواهد کرد چرا که هر فرستنده باید دائماً ورود و خروج اعضای خود را پیگیری نماید. برای سیستمی که مثلاً برای پخش تلویزیونی طراحی شده و میلیونها مشترک دارد این روشها هرگز کار نخواهد کرد!

RSVP: پروتکل رزرو منابع

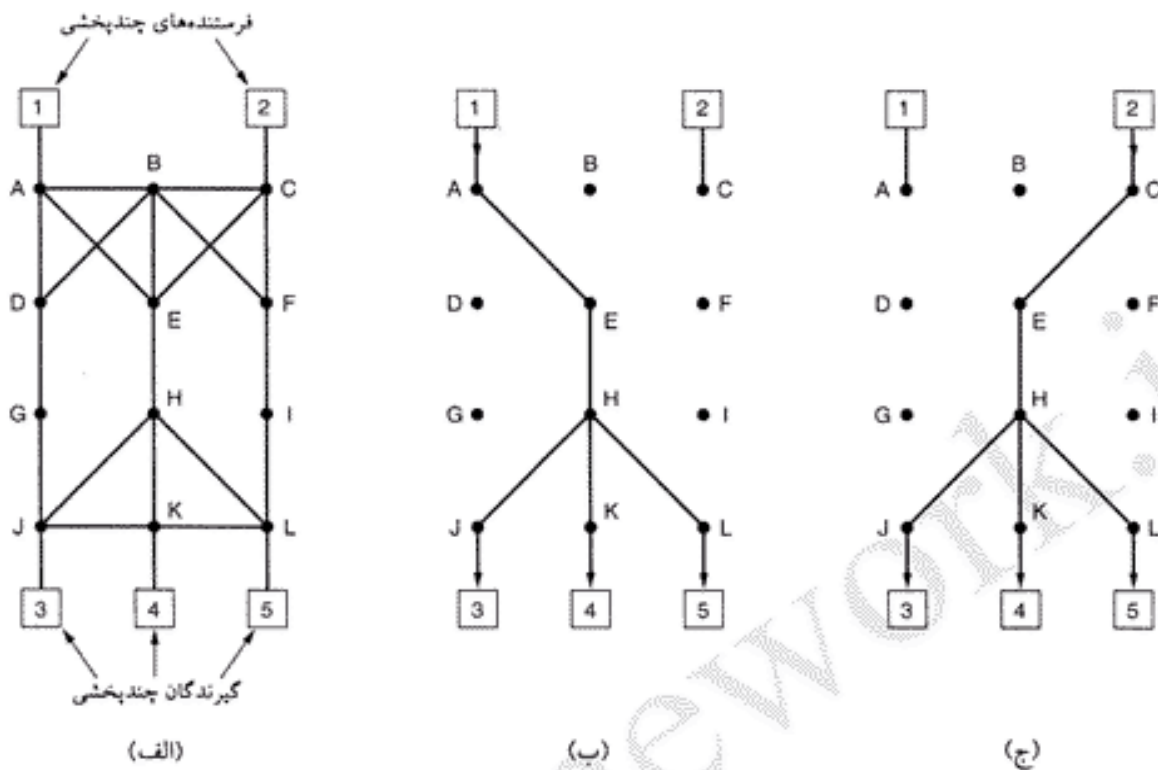
اصلیترین پروتکل پیشنهاد شده توسط IETF برای ارائه خدمات مجتمع، RSVP نامیده می شود. این پروتکل در RFC 2205 تشریح شده است و برای رزرو کردن پهنای باند بکار می آید. پروتکلهای دیگری که در RFC 2206 تا RFC 2210 تشریح شده اند چگونگی ارسال داده ها را توصیف می کنند. RSVP اجازه می دهد که چندین فرستنده بتواند برای چندین گروه از گیرندگان خود داده بفرستد و همچنین امکان آن را فراهم کرده که گیرندگان بتوانند کانال مورد نظر خود را آزادانه عوض کنند. در عین حال پروتکل RSVP، استفاده از پهنای باند را بهینه سازی کرده و از بروز ازدحام جلوگیری می کند.

در ساده ترین حالت، این پروتکل از روش «مسیریابی چندپخش مبتنی بر درخت پوشا»^۲ که قبلاً تشریح شد، بهره می گیرد. به هر گروه یک آدرس یکتا انتساب داده می شود و برای ارسال یک بسته به گروه خاص، آدرس آن گروه در بسته قرار می گیرد. سپس توسط الگوریتم استاندارد مسیریابی چندپخش، یک درخت پوشا که تمام اعضای آن گروه را در بر می گیرد، ایجاد می گردد. الگوریتم مسیریابی چندپخش جزو استاندارد RSVP محسوب نمی شود و تنها تفاوت آن با الگوریتم معمولی مسیریابی چندپخش آنست که بطور متناوب، مقداری اطلاعات اضافی برای هر گروه ارسال می شود تا مسیریابهای واقع بر روی درخت، آنها را در ساختمان داده خاصی در حافظه خود ذخیره نمایند.

به عنوان مثال شبکه شکل ۵-۳۷-الف را در نظر بگیرید. ماشینهای میزبان ۱ و ۲ فرستنده های چندپخش و ماشینهای ۳ و ۴ و ۵ گیرندگان چندپخش هستند. در این مثال فرستنده ها و گیرنده ها کاملاً از هم جدا (Disjoint) هستند ولی در حالت کلی ممکن است مجموعه ماشینهای فرستنده و گیرنده عضو مشترک هم داشته باشند. درختهای چندپخش برای فرستنده شماره ۱ و شماره ۲ در شکلهای ۵-۳۷-ب و ۵-۳۷-ج نشان داده شده اند. برای دریافت بهتر و جلوگیری از ازدحام، هر یک از گیرندگان یک گروه می توانند پیامی برای رزرو پهنای باند

۱. Resource Reservation Protocol

۲. Spanning Tree Multicast Routing (به بخش ۵-۲-۸ مراجعه کنید).



شکل ۵-۳۷. (الف) ساختار یک شبکه (ب) «درخت پوشای چندبخشی» برای ماشین میزبان ۱ (ج) «درخت پوشای چندبخشی» برای ماشین میزبان ۲.

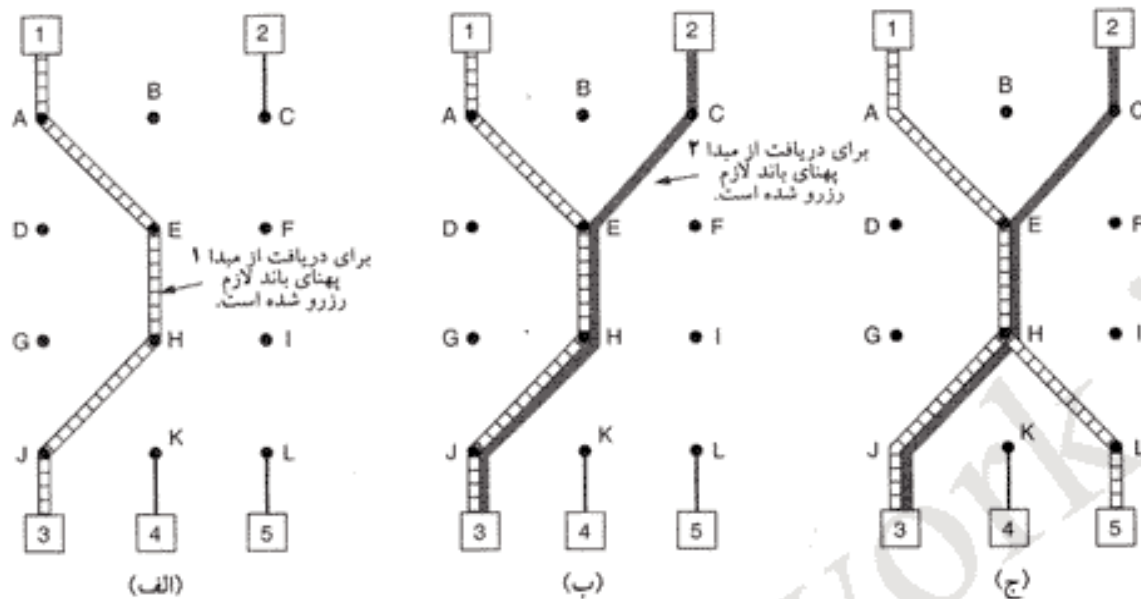
ارسال کنند تا از طریق درخت به فرستنده برسد. این پیام به کمک «الگوریتم هدایت در مسیر معکوس»^۱ که قبلاً تشریح شد، در درخت انتشار می‌یابد. در هر گام، مسیریاب به این پیام توجه کرده و پهنای باند لازم را رزرو می‌نماید. اگر پهنای باند کافی موجود نباشد، «پیغام شکست» برخواهد گشت. اگر این پیغام موفقیت‌آمیز به مبدا آن برگردد، پهنای باند لازم در کل مسیر رزرو شده است.

مثالی از چگونگی رزروسازی در شکل ۵-۳۸-الف نشان داده شده است. در اینجا ماشین میزبان ۳ تقاضای برقراری یک کانال با ماشین ۱ را داده است. به محض برقراری چنین کانالی، بسته‌ها می‌توانند بدون مسئله ازدحام از ۱ به ۳ جاری شوند. حال ببینیم اگر ماشین ۳ بعداً بخواهد کانالی دیگر با ماشین میزبان ۲ برقرار کند چه اتفاقی می‌افتد (زیرا مثلاً می‌خواسته دو کانال تلویزیونی را بطور همزمان تماشا کند). مسیر دوم مطابق شکل ۵-۳۸-ب رزرو می‌شود. دقت کنید که از ماشین گیرنده ۳ تا مسیریاب E به دو کانال مستقل نیاز است چراکه قرار است دو «استریم ویدیویی مستقل»^۲ ارسال شود.

در شکل ۵-۳۸-ج، ماشین میزبان ۵ تصمیم می‌گیرد برنامه‌ای را که توسط ماشین ۱ پخش می‌شود تماشا کند و او نیز رزرواسیون لازم را انجام می‌دهد. ابتدا پهنای باند لازم تا H رزرو می‌شود. از H به بعد، مسیریابها متوجه می‌شوند که از قبل کانالی با ماشین ۱ دارند و از داده‌های آن تغذیه می‌شوند، لذا نیازی به رزرو نیست. دقت کنید که ممکن است ماشینهای ۳ و ۵ به پهنای باند متفاوتی نیاز داشته باشند (چرا که مثلاً ماشین ۳ تصاویر تلویزیونی را سیاه و سفید نگاه می‌کند و نیازی به اطلاعات رنگی ندارد)، فلذا ظرفیت رزرو شده باید آنقدر زیاد باشد تا بتواند از گیرنده‌ای را که به بیشترین مقدار پهنای باند احتیاج دارد، برآورده سازد.

^۲ Independent Stream

^۱ Reverse Path Forwarding (به پخش ۵-۲-۷ مراجعه کنید)



شکل ۵-۳۸. (الف) ماشین میزبان ۳ تقاضای کانالی به ماشین ۱ می‌کند. (ب) ماشین میزبان ۳ تقاضای

کانالی دیگر به ماشین ۲ می‌کند. (ج) ماشین میزبان ۵ تقاضای کانالی به ماشین ۱ می‌کند.

در RSVP به هر گیرنده آزادی عمل داده شده تا بتواند با یکبار عملیات رزرو، کانالهایی با بیش از یک فرستنده، ایجاد نماید. همچنین می‌تواند مشخص کند که آیا انتخاب او برای دوره‌ای از زمان ثابت است یا آنکه حق تغییر مبداء ارسال (فرستنده) را برای خود محفوظ نگاه داشته است. مسیریاب می‌تواند به کمک این اطلاعات برآورد بهینه و مناسبی از پهنای باند داشته باشد.

دلیل اتخاذ این استراتژی در محیطهای کاملاً پویا و در حال تغییر آنست که پهنای باند رزرو شده از انتخاب مبداء (فرستنده) مستقل و مجزاست: به محض آنکه یک گیرنده که قبلاً پهنای باند لازم را برای دریافت از یک مبداء خاص رزرو کرده، بخواهد مبداء دریافت خود را تغییر بدهد بخشی از مسیر قبلی برای مبداء جدید نیز معتبر است و نیازی به تخصیص پهنای باند ندارد. در ضمن اگر ماشین ۲ در حال ارسال همزمان چندین جریان ویدیویی (Video Streams) باشد، ماشینی مثل ۳ می‌تواند بدون نیاز به هیچگونه رزرواسیون مجدد و تغییر، از بین این کانالهای ویدیویی یکی را انتخاب نماید زیرا مسیریابها به آنچه که گیرنده تماشا می‌کند دقت و اعتنایی نمی‌کنند.^۱

۵-۴ خدمات متمایز (Differentiated Services)

«الگوریتمهای مبتنی بر جریان»^۲ قابلیت عرضه کیفیت خوب خدمات به یک یا چند جریان را دارند زیرا در طول مسیر هر منبعی را که نیاز است از قبل رزرو می‌کنند. ولی این روشها یک اشکال دارند: در این الگوریتمها نیاز است که برای هر جریان (Flow)، پیشاپیش تنظیمات لازم انجام شود در حالی که در مقیاس کلان یعنی وقتی که هزاران یا میلیونها «جریان» وجود دارد قابلیت اجرایی خود را از دست می‌دهند. از طرفی در هر مسیریاب «وضعیت» هر

۱. عبارت روشنتر وقتی یک گیرنده مثل A کانالی با پهنای باند معین با فرستنده B رزرو می‌کند و B بطور همزمان مثلاً ده استریم ویدیویی بخش می‌کند، گیرنده A در انتخاب یکی از این ده استریم آزادی عمل دارد و نیازی نیست که به مسیریابها در این خصوص اطلاع داده شود چرا که آنها پهنای باند لازم را رزرو کرده‌اند و اصراری ندارند که بدانند فرستنده چه چیزی برای گیرنده،

۲. Flow-Based Algorithms

می‌فرستند. -م

جریان بطور جداگانه نگهداری می شود و عملکرد این الگوریتم ها در مقابل خرابی یک مسیریاب آسیب پذیر خواهد بود. نهایتاً آنکه برای تنظیم و ایجاد «جریان» باید تبادل اطلاعات پیچیده ای بین مسیریابها انجام گیرد. در نتیجه RSVP یا الگوریتمهای مشابه آن، بسیار کم پیاده سازی عملی شده اند.

به همین دلایل، IETF راهکاری ساده تر برای تأمین کیفیت خدمات (QoS) ابداع کرد؛ روشی که بدون نیاز به هیچ تنظیمات قبلی یا تعیین کل مسیر، می تواند به صورت محلی و مجزا در هر مسیریاب پیاده سازی شود. این راهکار اصطلاحاً «روش مبتنی بر کلاس» (Class-Based) برای تضمین کیفیت خدمات نامیده می شود (در مقابل روشهای مبتنی بر جریان). IETF یک معماری مناسب به نام «خدمات متمایز» برای آن طراحی و استانداردسازی کرده است که در مستندات RFC به شماره های ۲۴۷۴ و ۲۴۷۵ و مستندات دیگر تشریح شده است. در ادامه به تشریح این روش می پردازیم.

«خدمات متمایز» (که به اختصار DS گفته می شود) می تواند توسط مجموعه ای از مسیریابها که در یک «حوزه مدیریتی واحد» (Administrative Domain) قرار می گیرند (مثلاً یک ISP یا شرکت مخابرات)، عرضه شود. مدیریت مسئول شبکه، مجموعه ای از کلاسهای متفاوت خدمات و متناظر با آن، قواعد هدایت بسته ها (Forwarding Rules) را تعریف می کند.

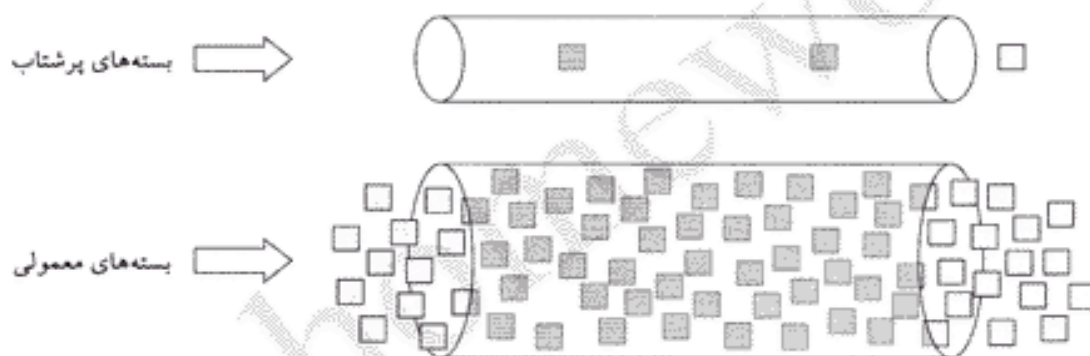
اگر یک مشتری برای دریافت خدمات نوع DS تقاضای ورود به شبکه را بدهد، بسته های ارسالی او در ورود به حوزه، فیلد «نوع خدمات» (Type of Service) را با خود حمل می کنند تا به برخی از آنها خدمات بهتری (مثل خدمات ویژه^۱) ارائه شود. ممکن است لازم باشد ترافیک تعریف شده در هر کلاس از شکل خاصی پیروی نماید (مثلاً باید از الگوریتم سطل سوراخ با نرخ خروجی مشخص تبعیت کند). متصدی شبکه با گرایشهای اقتصادی و تجاری ممکن است برای انتقال «بسته های ویژه» (Premium Packets) هزینه اضافی بگیرد یا مثلاً به ازای بهای اشتراک ثابت و ماهانه، تعداد N بسته ویژه از کاربر پذیرفته و هدایت شود. دقت کنید که این الگو نیاز به تنظیمات قبلی، رزروسازی منبع و نیازی به اتلاف وقت برای مذاکره بین طرفین نهایی در هر «جریان» ندارد. به همین دلیل پیاده سازی خدمات DS بسیار آسان است.

«خدمات مبتنی بر کلاس» در صنایع دیگر نیز وجود دارد. به عنوان مثال، شرکتهای تحویل محموله های پستی (Package Delivery) نیز سه نوع خدمات عرضه می کنند: شبانه روزی، دو روزه یا سه روزه. یا مثلاً خطوط هواپیمایی «خدمات کلاس برتر» (First Class)، «کلاس تجاری» (Business Class) و «کلاس معمولی» عرضه می کنند. در قطارهای دورپیما نیز خدمات در کلاسهای متفاوتی عرضه می شود و حتی قطارهای زیرزمینی (مترو) نیز در دو کلاس مختلف خدمات ارائه می کنند. برای بسته های حاوی اطلاعات، کلاسهای متفاوت خدمات برحسب میزان «تاخیر»، «لرزش» (Jitter)، احتمال حذف بسته در صورت بروز ازدحام و امکاناتی نظیر همینها تعیین می شود.

برای آنکه تفاوت بین «کیفیت خدمات مبتنی بر جریان» و «کیفیت خدمات مبتنی بر کلاس» روشنتر شود نمونه ای مثل «تلفن اینترنتی» را مدنظر قرار بدهید. در روش مبتنی بر جریان، هر تماس تلفنی منابع خاص خود و تضمینهای لازم را شبکه اخذ می کند. در روش مبتنی بر کلاس تمام تماسهای تلفنی همگی از منابع رزرو شده ای که برای «کلاس تلفنی» تهیه دیده شده، استفاده می کنند. این منابع در اختیار بسته هایی که در کلاس انتقال فایل یا کلاسهای دیگر هستند، قرار نمی گیرد و صرفاً برای «کلاس تلفنی» پیش بینی شده است ولی اینگونه هم نیست که برای هر تماس تلفنی منابع اختصاصی و مجزا در نظر گرفته شود.

هدایت پُرشتاب (Expedited Forwarding)

انتخاب کلاس خدمات بر عهده کارفرمای شبکه است ولیکن از آنجایی که بسته ها از چندین زیرشبکه مجزا (یا کارفرمای مستقل) عبور می کنند [و ممکن است کلاس خدمات هر زیرشبکه متفاوت و سلیقه ای باشد]، IETF در حال کار بر روی تعریفی واحد برای کلاسهای خدمات است به گونه ای که مستقل از نوع شبکه باشد. ساده ترین کلاس، کلاس «هدایت پرشتاب» است که با آن شروع می کنیم. این کلاس در RFC 3246 تشریح شده است. ایده ای که در پشت روش «هدایت پرشتاب» نهفته است ساده به نظر می رسد. خدمات در دو کلاس قابل ارائه است: «معمولی» و «پرشتاب» (Regular & Expedited) بخش اعظم ترافیک شبکه از نوع معمولی هستند در حالی که فقط کسر کوچکی از آن نیاز به «خدمات پُرشتاب» دارند. بسته های پُرشتاب باید بگونه ای در زیرشبکه حرکت کنند که گویی هیچ بسته دیگری وجود ندارد. توصیفی نمادین از مفهوم سیستم «دو تونلی» (Two-Tube) در شکل ۵-۳۹ ارائه شده است. به خاطر داشته باشید که کماکان یک خط فیزیکی در اختیار است؛ دو لوله منطقی که در شکل نشان داده شده فقط نماد رزرو بخشی از پهنای باند هستند نه آنکه یک خط فیزیکی دیگر هم وجود داشته باشد.



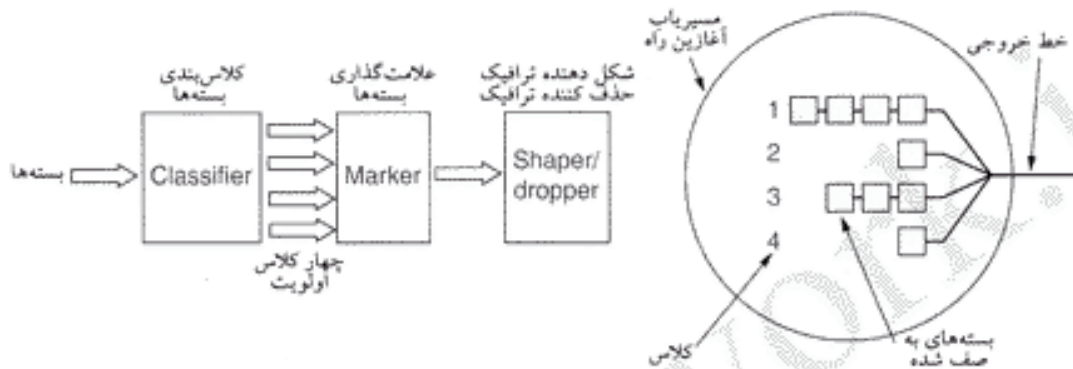
شکل ۵-۳۹. بسته های پرشتاب با شبکه ای بدون ترافیک مواجه می شوند.

یکی از روشهای پیاده سازی این استراتژی آن است که مسیر یابها به نحوی برنامه ریزی شوند که برای هر یک از خطوط خروجی خود دو صف مجزا تشکیل بدهند: یکی برای بسته های پرشتاب و دیگری برای بسته های معمولی. بسته های دریافتی برحسب نوع آنها به یکی از این صفها وارد می شوند. برای زمان بندی بسته های معمولی روشی مثل «روش صف بندی وزن دار» (Weighted Queuing) بهره گرفت. مثلاً اگر ده درصد از بسته ها از نوع پرشتاب و مابقی از نوع معمولی باشند، تخصیص ۲۰ درصد از پهنای باند برای ترافیک پرشتاب و هشتاد درصد باقیمانده برای ترافیک معمولی مناسب خواهد بود. با این کار پهنای باند اختصاص داده شده به ترافیک پرشتاب دو برابر مقدار مورد نیاز آن است و بدین ترتیب تأخیر پائینی خواهد داشت. برای اجرای چنین راهکاری می توان به ازای ارسال ۴ بسته معمولی یک بسته پرشتاب ارسال کرد (البته با فرض آن که اندازه بسته ها توزیعی مشابه و یکنواخت داشته باشد). انتظار می رود در این روش حتی در صورت سنگین بودن بار زیرشبکه، بسته های پرشتاب زیرشبکه را بی بار و خلوت ببینند.

هدایت تضمین شده (Assured Forwarding)

برای مدیریت انواع کلاسهای خدمات، روشی دقیقتر به نام «هدایت تضمین شده» ارائه و در RFC 2597 تشریح شده است. در این روش چهار کلاس اولویت تعریف شده و هر کلاس منابع خاص خود را در اختیار دارد. علاوه بر این، سه احتمال برای حذف بسته در اثر بروز ازدحام تعریف شده است: احتمال پایین، متوسط و زیاد. مجموع

ترکیبات مختلف این دو عامل، دوازده کلاس خدمات متفاوت ایجاد می‌کند. در شکل ۴۰-۵ یک روش برای پردازش بسته‌ها به منظور هدایت تضمینی آنها، نشان داده شده است. در مرحله اول بسته‌ها برحسب کلاس اولویتشان در یکی از چهار کلاس، رده‌بندی می‌شوند. این مرحله می‌تواند در ماشین میزبان فرستنده بسته‌ها انجام شود (به نحوی که در شکل نشان داده شده است) یا آنکه در اولین مسیریاب (مدخل ورودی به زیرشبکه) انجام شود.



شکل ۴۰-۵. پیاده‌سازی مکانیزم هدایت تضمین شده برای یک «جریان داده».

در مرحله ۲ بسته‌ها برحسب کلاسشان علامتگذاری می‌شوند. بدین منظور در سرآیند هر بسته به فیلد خاصی نیاز است. خوشبختانه یک فیلد هشت بیتی به نام Type of Service (نوع خدمات) در بسته IP وجود دارد که در آینده آن را به اختصار بررسی خواهیم کرد. در RFC 2597 شش بیت از این هشت بیت برای تعیین کلاس بسته‌ها تعریف شده و دو بیت باقیمانده برای استفاده‌هایی که از قبل داشته یا استفاده در آینده، رها شده‌اند.

در مرحله سوم بسته‌ها از یک «فیلتر شکل دهنده / حذف کننده» (Shaper / Dropper Filter) عبور کرده و برای آنکه ترافیک بسته‌های هر یک از چهار کلاس شکل قابل قبولی داشته باشند به برخی از آنها تأخیر مصنوعی تحمیل می‌شود (مثلاً به کمک الگوریتم سطل سوراخ یا سطل نشانه‌دار). در صورتی که تعداد بسته‌ها در هر کلاس، از حد مجاز بیشتر شده باشد در این مرحله برخی از آنها حذف می‌گردند. (روشهای دقیقتری نیز برای حذف بسته‌ها وجود دارد که از فیدبک بهره می‌گیرند).

در این مثال هر سه مرحله فوق‌الذکر توسط ماشین فرستنده انجام شده و جریان خروجی بسته‌ها به اولین مسیریاب ارسال می‌شود. بدیهی است که این مراحل می‌تواند توسط یک نرم‌افزار خاص شبکه یا سیستم عامل هر ماشین انجام شود تا نیازی به تغییر در برنامه‌های کاربردی موجود نباشد.

۵-۵-۵ سوئیچ برچسب و MPLS

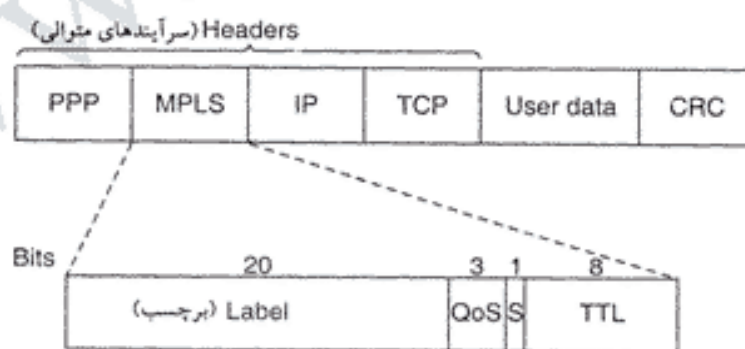
در خلال زمانی که IETF بر روی موضوع «خدمات مجتمع» کار می‌کرد، چندین تولیدکننده محصولات مسیریابی نیز بر روی روشهای بهتر هدایت بسته‌ها متمرکز شده بودند. کار آنها بر این محور بود که در ابتدای هر بسته یک «برچسب» (Label) اضافه شود و بجای آنکه مسیریابی و هدایت بسته‌ها مبتنی بر آدرس مقصد باشد براساس این «برچسب» انجام شود. با استفاده از این «برچسب» به عنوان یک اندیس در جدول داخلی هر مسیریاب، خط خروجی صحیح و مناسب برای هر بسته پیدا می‌شود. به کمک این روش، مسیریابی بسته‌ها به سرعت انجام شده و

منابع مورد نیاز در طول مسیر رزرو خواهد شد.

البته برچسب گذاری بر روی هر «جریان» شباهت عجیبی به مدارهای مجازی پیدا می کند. در شبکه های ATM، X.25 و Frame Relay یا هر زیرشبکه مدار مجازی دیگر نیز یک «برچسب» (یا به عبارتی یک شناسه مدار مجازی^۱) در هر بسته قرار داده می شود و با استفاده از آن به عنوان یک اندیس برای درایه های جدول^۲، مسیر مناسب بدست می آید. علیرغم آنکه بسیار از افراد در جامعه اینترنت از شبکه های اتصال گرا بشدت گریزان هستند، به نظر می رسد که این ایده با هدف مسیریابی سریع و تأمین کیفیت خدمات (QoS) بار دیگر به صحنه برگشته است. ولیکن بین روشی که در اینترنت برای تعیین مسیر بکار می رود و روشی که در شبکه های مدار مجازی اعمال می شود تفاوت های بنیادی وجود دارد و تکنیک برچسب گذاری بسته ها با روش سنتی سوئیچینگ متفاوت است. ایده جدید سوئیچینگ با نامهای متنوعی مثل «سوئیچینگ برچسب»^۳ یا «سوئیچینگ علامت»^۴ شناخته می شود. در نهایت IETF آن را تحت نام MPLS^۵ استاندارد کرد. ما نیز در ادامه از نام MPLS استفاده می کنیم. این استاندارد در RFC 3031 و چندین RFC دیگر تشریح شده است.

مضاف بر این، برخی افراد بین «مسیریابی» و «سوئیچینگ» فرق می گذارند. مسیریابی فرآیند جستجو در جدول مسیریابی به دنبال آدرس مقصد هر بسته و پیدا کردن خط مناسب برای آن است. برعکس در فرآیند سوئیچینگ از برچسب هر بسته به عنوان یک اندیس در جدول مسیریابی استفاده می شود و با استفاده از این اندیس بلافاصله خط خروجی پیدا می شود، بدون آن که نیازی به جستجو باشد. البته این تعاریف و تعابیر جهان شمول و همگانی نیستند.

اولین مسئله آنست که این برچسب در کجا قرار داده شود. از آنجایی که بسته های IP برای شبکه های مدار مجازی طراحی نشده بودند، طبعاً هیچ فیلدی در سرآیند بسته IP برای درج شماره های مدار مجازی وجود ندارد. به همین دلیل سرآیند جدید MPLS، باید در جلوی سرآیند هر بسته IP قرار بگیرد. در خطوط مستقیم بین هر دو مسیریاب که مبتنی بر «فریمینگ PPP» کار می کنند ترتیب سرآیندها طبق شکل ۵-۴۱ عبارتند از: سرآیند PPP، سرآیند MPLS، سرآیند IP و نهایتاً سرآیند TCP. در واقع باید MPLS را در لایه ۲/۵ فرض کرد!!!



شکل ۵-۴۱. ارسال یک قطعه TCP (TCP Segment) با استفاده از IP، MPLS، و PPP.

سرآیند عمومی MPLS (MPLS Header) چهار فیلد دارد که مهمترین آنها فیلد Label (فیلد برچسب) است که در آن یک اندیس درج می شود. فیلد QoS، کلاس خدمات را مشخص می کند. فیلد S بدان منظور تعریف شده که در شبکه های سلسله مراتبی چندین سرآیند MPLS متوالیاً به بسته اضافه گردد. (این موضوع در زیر تشریح

۳. Label Switching

۲. Table Entries

۱. Virtual Circuit Identifier

۵. Multi-Protocol Label Switching

۴. Tag Switching

شده است.) فیلد TTL زمان حیات بسته را مشخص می‌کند و به ازای هر گام یک واحد از آن کم می‌گردد؛ هر گاه مقدار این فیلد به صفر برسد، بسته حذف می‌شود. این ویژگی بدان منظور مفید است که از حلقه بی‌نهایت که در اثر ناپایداری (عدم همگرایی) جدول مسیریابی بروز می‌کند، اجتناب شود.

از آنجایی که سرآیند MPLS بخشی از بسته لایه شبکه یا فریم لایه پیوند داده‌ها محسوب نمی‌شود لذا MPLS تا حد زیادی مستقل از هر دو لایه است. از بین تمام محاسن دیگر، دستاورد ویژگی «استقلال از دیگر لایه‌ها» آنست که می‌توان سونیچهای MPLS را به گونه‌ای ساخت که بتواند هم بسته‌های IP و هم سلولهای ATM را برحسب مورد، هدایت کند. این ویژگی همانی است که براساس آن کلمه Multiprotocol در ابتدای نام MPLS ظاهر شده است.

وقتی یک بسته یا سلول غنی‌شده با سرآیند MPLS در یک مسیریاب MPLS دریافت می‌شود از برحسب آن به عنوان اندیسی در جدول داخلی مسیریاب استفاده شده و خط خروجی متناسب با آن تعیین می‌شود و قبل از خروج بسته از آن خط، برحسب جدیدی در فیلد مربوطه درج می‌گردد. تغییر در برحسبها در تمام زیرشبکه‌های مدار مجازی معمول و متعارف است چرا که برحسبها در هر مسیریاب معنای محلی دارند و دو مسیریاب متفاوت ممکن است بسته‌های نامربوط را با برحسبی یکسان برای مسیریاب دیگر بفرستند چرا که این بسته‌ها همگی در بخشی از مسیر مشترکند.^۱ به همین دلیل در هر گام برحسبهای بسته قبل از انتقال بر روی خط خروجی به برحسب جدید و معتبر در مسیریاب بعدی نگاشته می‌شود. این مکانیزم را در شکل ۵-۳ مشاهده کردیم. MPLS نیز از روش مشابهی بهره گرفته است.

یکی از تفاوت‌های MPLS با شبکه‌های مدار مجازی، «میزان تجمیع» (Aggregation Level) و صرفه‌جویی در تعداد درایه‌های جدول^۲ مسیریابی است. در MPLS این امکان وجود دارد که هر «جریان» در زیرشبکه، دارای مجموعه برحسبهای خاص خود باشد ولی این قابلیت مهم نیز وجود دارد که گروهی از جریانها که همگی به یک مسیریاب خاص یا یک LAN ختم می‌شوند با برحسب یکسان و واحدی مشخص شوند. [بدین ترتیب تعداد درایه‌های جدول مسیریابی کاهش یافته و اصطلاحاً عمل تجمیع یا Aggregation انجام می‌شود.] گروهی از جریانها که با یک برحسب واحد مشخص می‌شوند اصطلاحاً به یک FEC^۳ مشابه متعلق هستند. [یعنی همه آنها به یک طریق هدایت می‌شوند.] کلاس FEC نه تنها مقصد همه بسته‌ها را مشخص می‌کند، بلکه کلاس خدمات مورد نیاز آنها را نیز تعیین می‌نماید (از دیدگاه انواع خدمات متمایز که در بخش ۵-۴ بدان اشاره شد). طبعاً فرآیند هدایت تمام بسته‌های یک FEC یکسان خواهد بود.

در مسیریابی متعارف به روش مدار مجازی، این قابلیت که بتوان چندین مسیر مجزا با نقاط پایانی متفاوت را با «شناسه مدار مجازی» واحد مشخص کرد وجود ندارد چرا که در این صورت راهی برای مشخص کردن مقصد نهایی بسته‌ها وجود نخواهد داشت در حالی که در MPLS بسته‌ها [به غیر از سرآیند ۴ بایستی MPLS] آدرس واقعی ماشین مقصد را نیز با خود حمل می‌کنند و بدین ترتیب در انتهای مسیری که با برحسب مشخص شده می‌توان سرآیند حاوی برحسب را حذف کرد و هدایت بسته‌ها به روش معمول و مبتنی بر آدرس لایه شبکه [مثل آدرس IP] ادامه یابد.

یکی از تفاوت‌های بنیادی بین MPLS و شبکه‌های مدار مجازی در چگونگی تشکیل جداول مسیریابی است.

۱. عبارت دیگر برحسب هر بسته هویت آنرا مشخص نمی‌کند بلکه مسیر خروج آن از مسیریاب فعلی را مشخص می‌کند لذا ضمیمی است که بسته‌های خروجی از یک مسیریاب که هیچ ربطی بهم ندارند ولی لااقل گام بعدی مسیر آنها یکی است (یعنی از خط مشابهی وارد مسیریاب بعدی و از خط مشابهی، از آن خارج می‌شوند) دارای برحسب یکسانی باشند. -م

در شبکه های مدار مجازی وقتی یک کاربر بخواهد یک «اتصال» ایجاد کند، توسط لایه شبکه یک بسته خاص جهت تنظیم مسیر به زیرشبکه روانه می شود تا ضمن ایجاد یک مسیر درایه های لازم در جداول مسیریابی درج شود. MPLS بدین نحو عمل نمی کند چرا که در آن عمداً هیچ مرحله ای برای تنظیم اتصال پیش بینی نشده است. (زیرا در غیر این صورت نرم افزارهای موجود اینترنت دچار شکاف و ناسازگاری می شد.) در عوض برای تنظیم و ایجاد درایه های جدول مسیریابی از دو راهکار جدید استفاده شده است.

در راهکار اول که «روش متکی به داده» (Data driven) نامیده می شود هرگاه بسته ای در اولین مسیریاب دریافت شود، آن مسیریاب با مسیریاب واقع بر روی مسیر جریان، تماس گرفته و از او می خواهد که یک برچسب برای این جریان ایجاد نماید. این فرآیند به صورت بازگشتی (Recursive) ادامه می یابد تا مجموعه برچسبها ایجاد شوند. در واقع این روش را می توان ایجاد «مدار مجازی برحسب تقاضا»^۱ فرض کرد.

پروتکل هایی که عمل برچسب دهی را انجام می دهند مراقب هستند تا از بروز حلقه اجتناب شود. برای این کار از تکنیکی به نام «رسمانهای رنگی» (Colored Thread) بهره گرفته می شود. انتشار معکوس یک FEC را می توان با یک «رسمان رنگی و یکتا» [تمثیلی از یک مسیر در شبکه] در زیرشبکه مقایسه کرد. اگر مسیریاب رنگی را مشاهده کند که خودش نیز به همان رنگ است متوجه می شود که در انتخاب مسیر، حلقه ایجاد شده و برای رفع آن اقدام می کند.^۲ روش «برچسب دهی متکی به داده» (Data driven) در شبکه هایی کاربرد دارد که زیر ساخت انتقال آنها ATM است. (همانند بیشتر سیستمهای تلفن)

راهکار دیگر برای برچسب دهی به جریان داده ها، در شبکه هایی کاربرد دارد که زیربنای آنها ATM نیست. این روش اصطلاحاً «روش متکی به کنترل» (Control Driven) نامیده می شود و گونه های متنوعی از آن وجود دارد. یکی از این گونه ها به ترتیب ذیل عمل می کند: وقتی یک مسیریاب راه اندازی (بوت) می شود ابتدا بررسی می کند که در انتهای چه مسیرهایی قرار دارد (یعنی مثلاً چه ماشینهایی بر روی LAN متصل به او قرار دارند). سپس برای تمام آنها یک یا چند FEC [شناسه یک گروه با کلاس معادل] تولید کرده و ضمن تخصیص یک برچسب به هر یک از این گروه ها، آنها را به همسایه های خود اطلاع می دهد. آنها نیز به ترتیب برچسبها را در جدول مسیریابی خود وارد کرده و با تعیین برچسبی جدید [متناظر با هر برچسب قبلی] آنها را به همسایه های خود اطلاع می دهند تا آنکه تمام مسیریابها از مسیرهای جدید آگاه شوند. در حین ایجاد مسیر می توان منابع لازم را نیز برای تضمین کیفیت خدمات رزرو کرد.

MPLS می تواند بطور همزمان در چندین سطح عمل کند. در بالاترین سطح، هر زیرشبکه حامل را می توان یک نوع Metarouter فرض کرد که بین هر مبداء و مقصد مسیری وجود دارد که از این متاروترها می گذرد؛ در این مسیر از MPLS استفاده می شود.^۳ با این حال در درون یک زیرشبکه حامل نیز می توان از MPLS بهره گرفت و بدین ترتیب مسیریابهای داخلی نیز برچسب دومی به هر بسته می افزایند و برچسب گذاری سطح دوم پدید می آید. در حقیقت یک بسته می تواند دنباله ای از برچسبهای MPLS را به همراه داشته باشد. بیت S در شکل ۵-۴۱ مسیریاب را آگاه می کند که آیا برچسبهای دیگری هم وجود دارد. در آخرین برچسب، بیت S یک است؛ در حالیکه در بقیه برچسبها بیت S صفر می باشد. در عمل می توان از این قابلیت برای پیاده سازی VPN یا تونلهای بازگشتی

۱. On-Demand Virtual Circuit

۲. به عبارت بهتر اگر یک مسیر بین دو نقطه را در قالب یک رسمان رنگی مجسم کنیم هر بسته ای که با رنگ همان مسیر دو بار دریافت شود نشان می دهد که بسته در یک حلقه قرار گرفته است و گرنه باید تا رسیدن به مقصد راه خود را ادامه بدهد. -م

۳. یعنی مسیر بین مبداء و مقصد از زیرشبکه های متفاوتی می گذرند که هر یک از این زیرشبکه های حامل یک مسیریاب واحد به نام «متاروتر» فرض شده است. -م

(Recursive Tunnel) بهره گرفت.

اگرچه ایده بنیادی MPLS ساده است ولیکن جزئیات آن بی نهایت پیچیده است و تنوع و بهینه سازیهای گسترده ای دارد، لذا ما بیش از این به موضوع فوق نخواهیم پرداخت. برای آگاهی بیشتر از مراجع ذیل استفاده کنید:

Davie and Rekhter, 2000; Lin et al., 2002; Pepelnjak and Guichard, 2001; Wang, 2001.

۵- بهم بندی شبکه ها (Internetworking)

تا اینجا تلویحاً فرض کرده ایم که تنها یک شبکه واحد و همگن وجود دارد که تمام ماشینهای چنین شبکه ای، در تمام لایه ها از پروتکل مشابهی بهره گرفته اند. متأسفانه چنین فرضی خیلی خوش باورانه است. شبکه ها اعم از LAN، MAN و WAN، انواع بسیار گوناگونی دارند. در هر لایه نیز از پروتکلهای متعددی استفاده می شود. در بخشهای آتی مواردی را موشکافی خواهیم کرد که در وصل دو یا چند شبکه و تشکیل «اینترنت»^۱ (internet) با آن مواجه خواهیم بود.

مناقشات گسترده ای پیرامون این سؤال وجود دارد که آیا کثرت بسیار زیاد انواع شبکه ها در حال حاضر، وضعیتی گذرا و موقتی است و به محض آنکه عموم مردم به شگفتی های یک شبکه خاص (مثلاً شبکه مورد نظر شما!) پی برند این کثرت به وحدت می رسد یا آنکه تکثر انواع امری اجتناب ناپذیر و همیشگی در جهان است و باقی خواهد ماند. وجود شبکه های متفاوت مستلزم داشتن پروتکلهای متفاوت است.

اعتقاد ما بر آن است که به دلایل ذیل گونه های متفاوتی از شبکه ها (و به تبع آن پروتکلهای متفاوت) تا ابد وجود خواهد داشت: اول آنکه شبکه های بسیار متنوعی در محیطهای متفاوتی نصب شده اند: تقریباً تمام کامپیوترهای شخصی بر مبنای TCP/IP کار می کنند. بسیاری از مؤسسات تجاری دارای کامپیوترهای بزرگ (Mainframe) با معماری شبکه SNA (متعلق به شرکت IBM) هستند. تعداد قابل توجهی از شرکت های مخابرات تلفنی، شبکه های ATM را به خدمت گرفته اند. در برخی از شبکه های محلی هنوز از پروتکل NCP/IPX (متعلق به شرکت Novell) یا AppleTalk (متعلق به شرکت Apple) بهره می گیرند. و از همه گذشته شبکه های بی سیم با پروتکلهای متنوعی در حال ظهور هستند. این رویه برای سالها ادامه خواهد داشت چرا که تکنولوژیهای جدید ظهور می کند، اشکالات و ناکارآمدیهای گذشته آشکار می شود؛ در عین حال با عرصه تکنولوژی جدید نمی توان یک شبه مشتریان را وادار کرد سیستمی جدید را بپذیرند و سیستمهای قدیمی خود را دور بریزند.

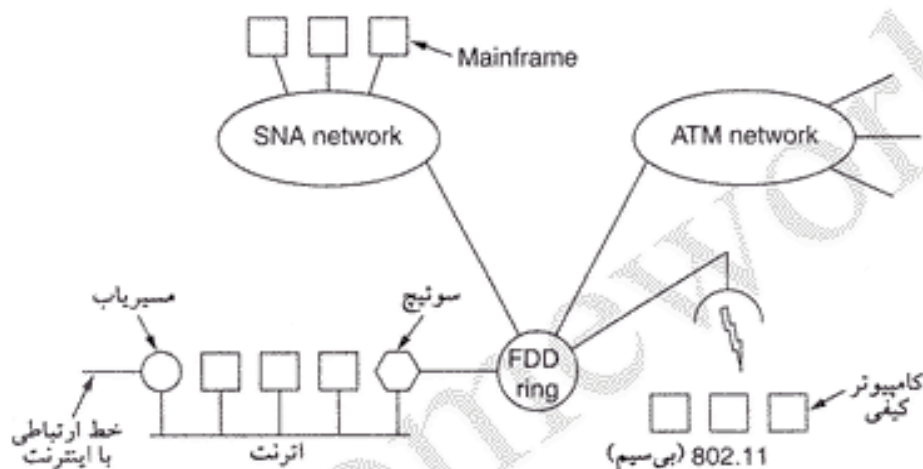
دلیل دوم آنکه کامپیوترها و شبکه ها روز به روز ارزانتر می شوند و تصمیم گیری در خصوص انتخاب و پیاده سازی شبکه ها به رده های پائینی یک سازمان محول می شود. بسیاری از شرکتها این سیاست را پیش گرفته اند که خریدهای بالای یک میلیون دلار باید توسط مدیر ارشد آن شرکت تأیید شود و خریدهای بالای صد هزار دلار توسط مدیران میانی مجاز است در حالی که خریدهای زیر صد هزار دلار توسط مدیران هر واحد بدون اجازه مدیران مافوق انجام می گیرد. این رویه به آنجا منتهی می شود که مثلاً واحد فنی مهندسی در یک سازمان، ایستگاههایی مبتنی بر یونیکس و پروتکل TCP/IP پیاده کند و واحد فروش، کامپیوترهای Mac با پروتکل AppleTalk را به خدمت بگیرد.

دلیل سوم آنکه شبکه های مختلف (مثل ATM و بی سیم) تکنولوژی شدیداً متفاوتی دارند لذا دور از انتظار نیست که وقتی سخت افزار جدیدی خلق می شود نرم افزار جدیدی نیز برای به خدمت گیری آن ایجاد شود. به

۱. وقتی کلمه internet تماماً با حروف کوچک نوشته می شود به شبکه عظیم و جهانی اینترنت اشاره نمی کند بلکه مراد از آن وصل چند شبکه کوچک و بزرگ و یکپارچه سازی آنهاست؛ مخفف internetwork

عنوان مثال امروزه منازل که در آنها کامپیوتر وجود دارد همانند ادارات در ده سال قبل است: مملو از کامپیوترهایی که ارتباطی با هم ندارند! در آینده ممکن است تلفن، تلویزیون و دستگاههای الکترونیکی خانگی نیز با یکدیگر شبکه شوند و بتوان از راه دور آنها را کنترل کرد. این تکنولوژی جدید بی شک شبکه های جدید و پروتکل های جدیدی را به صحنه خواهد آورد.

به عنوان مثالی از چگونگی اتصال شبکه های متفاوت به یکدیگر به شکل ۵-۴۲ دقت کنید. در این شکل شبکه یکپارچه ای را می بینیم که اجزای آن در چندین موقعیت فیزیکی پراکنده هستند و از طریق یک شبکه گسترده ATM بهم متصل شده اند. در یکی از مکانها یک شبکه فیبر نوری FDDI به عنوان ستون فقرات نصب شده است تا ارتباط یک شبکه اترنت، یک شبکه بی سیم 802.11 و یک شبکه متمرکز SNA را با یکدیگر برقرار کند.



شکل ۵-۴۲. مجموعه ای از شبکه های بهم متصل.

هدف از اتصال تمام این شبکه ها آن است که کاربران هر یک از آنها بتوانند با یکدیگر مبادله اطلاعات داشته باشند یا آنکه هر کاربر بتواند به اطلاعات مورد نظر خود در هر نقطه از شبکه دسترسی پیدا کند. رسیدن بدین هدف متضمن آن است که بسته ها، بین این شبکه ها رد و بدل شوند. از آنجایی که شبکه ها اختلافات بنیانی با یکدیگر دارند، رساندن بسته ها از یک شبکه به شبکه دیگر به نحوی که در ادامه خواهیم دید، چندان هم ساده نیست.

۵-۵-۱ شبکه ها از چه دیدگاهی متفاوتند؟

شبکه ها در موارد مختلفی با یکدیگر تفاوت ذاتی دارند. برخی از این تفاوتها مثل تکنیکهای مدولاسیون یا قالب فریم، مربوط به لایه فیزیکی یا لایه پیوند داده ها است. اینگونه تفاوتها مدنظر ما نیستند. در مقابل، در شکل ۵-۴۳ برخی از تفاوتهایی را که در لایه شبکه بروز می کنند، فهرست نموده ایم. تشریح این تفاوتهاست که نشان می دهد بهم بستن شبکه ها دشوارتر از کار کردن در یک شبکه واحد و همگون است.

وقتی بسته ارسالی از یک مبدا در یکی از شبکه ها، مجبور باشد برای رسیدن به شبکه مقصد از یک یا چند شبکه خارجی عبور کند (که این شبکه ها نیز ممکن است با شبکه مبدا اختلاف بنیادی داشته باشند)، در مرز ارتباطی بین دو شبکه مشکلات عدیده ای رخ می دهد. اولین مورد آن است که وقتی بسته هایی از یک شبکه «اتصال گرا» مجبور به عبور از یک شبکه «بدون اتصال» باشند (که احتمالاً ترتیب بسته ها را بهم می ریزد) این مشکل بروز می کند که فرستنده بسته ها انتظار چنین رخدادی را ندارد و گیرنده نیز کاری نمی تواند انجام بدهد. [چرا که فرض فرستنده و گیرنده آن است که بسته ها به ترتیب ارسال و به ترتیب نیز دریافت می شود. -م]

غالباً بین دو شبکه به تبدیل پروتکل نیاز است و اگر عملکرد مورد نیاز برآورده نشود این تبدیل دشواریهایی را در

مورد اختلاف	برخی از رویکردهای ممکن
نوع سرویس ارائه شده	سرویسهای اتصال گرا در مقابل سرویسهای بدون اتصال
انواع پروتکل	IP, IPX, SNA, ATM, MPLS, AppleTalk, ...
الگوی آدرس دهی	روش مسطح (Flat) مثلا در استانداردهای ۸۰۲ در مقابل روش سلسله مراتبی در IP
چندپخش	در برخی از شبکه ها از آن پشتیبانی می شود و در برخی نمی شود.
اندازه بسته	هر شبکه برای خودش یک سقف حداکثر برای طول بسته تعریف کرده است.
کیفیت خدمات (QoS)	در برخی از شبکه ها از آن پشتیبانی می شود (آنها در رده های متفاوت) و در برخی نمی شود.
مدیریت خطا	تحويل مطمئن و به ترتیب در مقابل تحويل غیرقابل اطمینان و خارج از ترتیب
کنترل جریان	پنجره لغزان، کنترل نرخ ارسال یا حتی بدون مکانیزم کنترل جریان
کنترل ازدحام	اعمال الگوریتم سطل سوراخ، الگوریتم سطل نشانه دار، بسته های دعوت به آرامش و نظائر آن
امنیت	قوانین امنیتی، اعمال روشهای رمزنگاری و نظائر آن
پارامترها	مقادیر مختلف زمان انقضای مهلت تایمرها، پارامترهای متفاوت توصیف جریان و نظائر آن
حسابرسی و دریافت هزینه	بر حسب زمان اتصال، بر حسب تعداد بسته، بایت یا حتی هیچکدام

شکل ۵-۲۳. موارد بی شمار اختلاف شبکه ها.

پی خواهد داشت. همچنین اغلب به تبدیل و نگاشت آدرسها نیاز است که متضمن وجود گونه ای از یک «سیستم فهرست» (Directory System) خواهد بود. از طرفی عبور یک بسته چندپخش (Multicast) از شبکه ای که نمی تواند از مسیریابی چندپخش پشتیبانی کند مستلزم تولید بسته های جداگانه برای یکایک ماشینهای مقصد است.

تفاوت در مقدار حداکثر طول هر بسته داده در شبکه های مختلف، می تواند یک معضل اساسی باشد: چگونه می توان یک بسته ۸۰۰۰ بایتی را از شبکه ای عبور داد که حداکثر طول بسته های آن ۱۵۰۰ بایت است؟ تفاوت در کیفیت خدمات دو شبکه (QoS) نیز موردی است که برای تحويل بسته های بی درنگ از طریق شبکه ای که بی درنگ بودن ارسال را تضمین نمی کند، به یک معضل جدی تبدیل می شود.

کنترل خطا، کنترل جریان و کنترل ازدحام نیز در شبکه های مختلف، تفاوت دارد. اگر مبداء و مقصد، هر دو انتظار داشته باشند که تمام بسته ها به ترتیب و بدون خطا تحويل شوند ولی یک شبکه میانی به محض احساس بروز ازدحام برخی از بسته ها را حذف نماید، بسیاری از برنامه های کاربردی درهم خواهند شکست. همچنین اگر بسته ها مدتی را بی هدف سرگردان شوند و به ناگاه راه خود را پیدا کرده و تحويل داده شوند و گیرنده انتظار چنین رفتاری را نداشته و از عهده دریافت این بسته ها بر نیاید معضلی جدی پدیدار می شود. همچنین مکانیزمهای تضمین امنیت، تنظیم پارامترها، قواعد دریافت هزینه (حسابرسی) و حتی قوانین ملی متفاوت، می تواند منجر به بروز مشکلاتی شود.

۲-۵-۵. چگونگی اتصال شبکه ها به یکدیگر

به گونه ای که در فصل چهارم بررسی کردیم شبکه ها را می توان به کمک ابزارهای متفاوتی به یکدیگر متصل کرد. اجازه بدهید اجمالا به آن مفاد پردازیم: در لایه فیزیکی شبکه ها را می توان توسط «تکرارکننده» (Repeater) یا هاب به یکدیگر متصل کرد. این دو ابزار فقط بیتها را از یک شبکه به شبکه ای از همان نوع منتقل می نمایند. اینها اغلب ابزارهای آنالوگ هستند و هیچ درکی در خصوص پروتکل های دیجیتال [پروتکل های لایه بالاتر] ندارند و فقط سیگنالهای دریافتی را «باز تولید» (Regenerate) می نمایند.

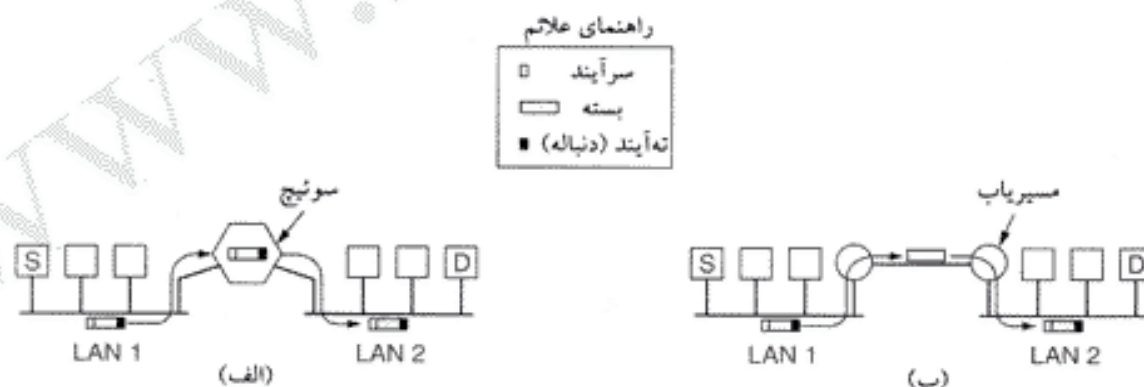
در یک لایه بالاتر به پلها و سوئیچها بر می خوریم که در لایه پیوند داده ها کار می کنند. این دستگاهها فریمها را می پذیرند، آدرسهای MAC را بررسی می کنند و آنها را به شبکه ای دیگر هدایت می نمایند. در ضمن اگر لازم باشد تبدیل پروتکل نیز انجام می دهند مثلاً آترنت را به FDDI یا 802.11 تبدیل می کنند.

در لایه شبکه، مسیریابها را داریم که می توانند دو شبکه را به یکدیگر متصل کنند. اگر دو شبکه، لایه های شبکه متفاوتی داشته باشند، مسیریاب ممکن است بتواند قالب بسته ها را به یکدیگر ترجمه نماید اگرچه امروزه ترجمه و تبدیل بسته ها به یکدیگر، به ندرت انجام می شود. یک مسیریاب را که بتواند با چندین پروتکل مختلف کار کند، اصطلاحاً «مسیریاب چند پروتکلی» (Multiprotocol Router) می نامند.

در لایه انتقال به «دروازه های انتقال» (Transport Gateway) می رسیم که می توانند واسطه بین دو اتصال در لایه انتقال شوند. به عنوان مثال یک «دروازه انتقال» می تواند این امکان را فراهم کند که جریان بسته ها بین یک شبکه TCP و یک شبکه SNA (که پروتکل لایه انتقال آنها متفاوت است)، مبادله شود. این دروازه، یک «اتصال TCP» (TCP connection) را به یک «اتصال SNA» می چسباند.

در آخر به لایه کاربرد و «دروازه کاربرد» (Application Gateway) می رسیم که این دروازه محتوای پیامها را بهم ترجمه می نماید. به عنوان مثال یک دروازه بین سیستم پست الکترونیکی در اینترنت (مثل سیستم RFC822) و سیستم پست الکترونیکی X.400، باید بتواند محتوای پیام نامه های الکترونیکی را تجزیه و تحلیل کرده و فیلدهای مختلف سرآیند آنها را به یکدیگر تبدیل نماید.

در این فصل به موضوع بهم بندی شبکه ها در لایه شبکه خواهیم پرداخت. برای آنکه ببینید هدایت اطلاعات در لایه پیوند داده ها چه تفاوتی با هدایت در لایه شبکه دارد، شکل ۵-۴۴ را در نظر بگیرید. در شکل ۵-۴۴-الف ماشین مبدا یعنی S می خواهد بسته ای را برای ماشین مقصد D بفرستد. این دو ماشین بر روی دو شبکه آترنت جدا که از طریق سوئیچ بهم متصل شده اند، واقع هستند. ماشین S بسته ای را در درون یک فریم جاسازی کرده و آن را به خروجی می فرستد. این فریم به سوئیچ رسیده و با بررسی آدرس MAC مشخص می شود که باید به LAN2 برود. سوئیچ، این فریم را از LAN1 برداشته و به LAN2 روانه می کند.



شکل ۵-۴۴. (الف) دو شبکه آترنت که از طریق سوئیچ بهم متصل شده اند. (ب) دو شبکه آترنت که از طریق مسیریاب بهم متصل شده اند.

حال همین وضعیت را در شرایطی در نظر بگیرید که این دو شبکه آترنت به جای سوئیچ از طریق یک جفت مسیریاب به یکدیگر متصل شده اند. ارتباط بین مسیریابها از طریق یک خط نقطه به نقطه برقرار شده است؛ این خط می تواند یک خط استیجاری با هزاران کیلومتر طول باشد. در اینجا فریم توسط مسیریاب اول دریافت شده و بسته جاسازی شده در درون آن، از فیلد داده فریم استخراج می شود. مسیریاب، آدرس درون بسته را (مثلاً آدرس IP

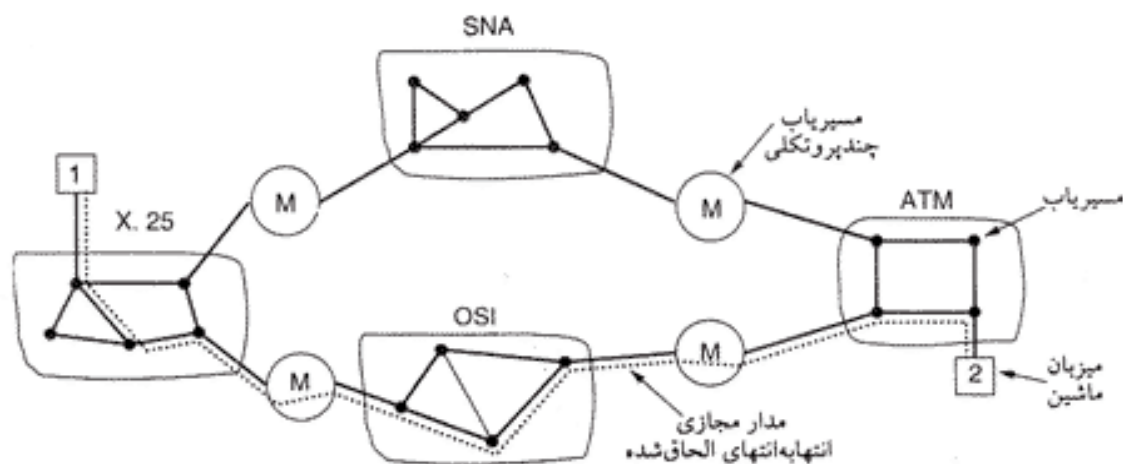
را) بررسی کرده و آنرا در درون جدول مسیریابی خود جستجو می‌نماید؛ سپس براساس این آدرس به نتیجه می‌رسد که باید بسته را به مسیریاب راه دور بفرستد و احتمالاً برای این کار مجبور خواهد شد که مبتنی بر پروتکل این خط، آنرا در فریم جدید و متفاوتی جاسازی نماید. در مسیریاب مقابل این بسته مجدداً درون فیلد داده از یک فریم اترنت جاسازی شده و بر روی LAN2 ارسال می‌شود.

تفاوت بنیادی بین حالتی که سوئیچ (یا پل) در میان است با وقتی که مسیریاب اتصال شبکه را برقرار کرده، آنست که در سوئیچ یا پل کل یک فریم براساس آدرس MAC آن هدایت و منتقل می‌شود در حالیکه در یک مسیریاب، یک بسته از درون فریم استخراج شده و سپس از آدرس درون بسته برای تصمیم‌گیری در خصوص محل ارسال آن استفاده می‌شود. سوئیچها مجبور نیستند که برای هدایت بسته‌ها درکی از پروتکل لایه شبکه داشته باشند در حالیکه مسیریابها اینگونه‌اند.

۳-۵-۵ مدارات مجازی الحاق‌شده (Concatenated Virtual Circuit)

دو روش برای بهم پندی شبکه‌ها ممکن است: روش الحاق زیرشبکه‌های مدار مجازی به صورت اتصال‌گرا و روش الحاق دیتاگرام. به نوبت این دو روش را بررسی خواهیم کرد ولی در ابتدا به ذکر نکته‌ای می‌پردازیم. در گذشته اکثر شبکه‌های عمومی اتصال‌گرا بودند (و شبکه‌هایی مثل SNA, Frame Relay و 802.16 و ATM هنوز هم اینگونه‌اند). با رشد و مقبولیت سریع اینترنت، شبکه دیتاگرام مد روز شد ولیکن این تصور که شبکه‌های دیتاگرام ابدی هستند، اشتباه است. در دنیای شبکه‌ها تنها چیزی که جاوید و ابدی می‌ماند «تغییر» است. با رشد شبکه‌های چند رسانه‌ای، احتمالاً اتصال‌گرایی با یکی از اشکال خود مجدداً به صحنه برمی‌گردد چراکه در شبکه‌های اتصال‌گرا راحتتر می‌توان کیفیت خدمات را تضمین نمود. لذا در ابتدا شبکه‌های اتصال‌گرا را مورد بحث و بررسی قرار می‌دهیم.

در مدل الحاق شبکه‌های مدار مجازی که در شکل ۴۵-۵ نشان داده شده، ایجاد «اتصال» با یک ماشین در شبکه‌ای دور دست، بروشی مشابه با روش معمولی انجام می‌گیرد: زیرشبکه می‌بیند که مقصد در شبکه‌ای دیگر واقع شده لذا یک مدار مجازی با آن مسیریاب که به شبکه مقصد نزدیکتر است، ایجاد می‌کند. از آن مسیریاب نیز یک مدار مجازی با یک «دروازه خارجی» ایجاد می‌شود. (دروازه خارجی یا External Router، یک مسیریاب چند پروتکلی است.) آن «دروازه» نیز مشخصات این مدار مجازی را در جدول خود درج کرده و کار را با ایجاد یک مدار مجازی جدید با مسیریاب زیرشبکه دیگر ادامه می‌دهد. این فرآیند ادامه می‌یابد تا آنکه مدار مجازی به ماشین مقصد ختم شود.



شکل ۴۵-۵. بهم پندی شبکه 'ک' کمک مدارات مجازی الحاق‌شده.

هرگاه یک بسته داده، در مسیری جریان یابد، هر یک از دروازه های میانی این بسته را رله می کنند و در صورت نیاز قالب بسته را تبدیل نموده و شماره های مدار مجازی را تغییر می دهند. بدیهی است که تمام بسته های داده باید به یک ترتیب از دروازه ها بگذرند؛ نتیجتاً ترتیب بسته های متعلق به یک جریان هرگز به هم نخواهد ریخت. ویژگی بنیادی این راهکار آنست که دنباله ای از مدارات مجازی بین ماشین مبدا و مقصد (از طریق دروازه های میانی) ایجاد و تنظیم می شود. هر دروازه جدولی را در خود نگاهداری می کند که این جدول فهرست مدارات مجازی را که از آن دروازه می گذرند و همچنین طریقه مسیریابی و شماره های جدید مدار مجازی را تعیین کرده است.

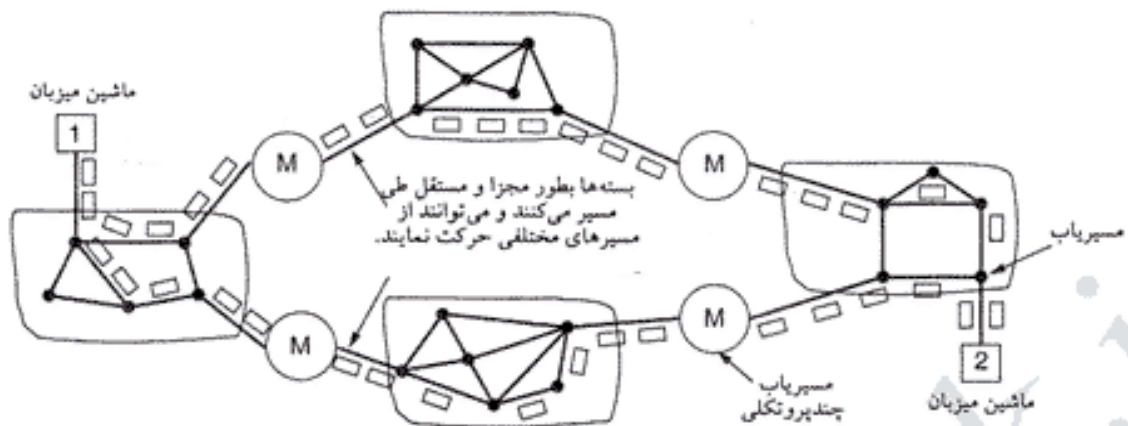
این ساختار زمانی به بهترین نحو کار می کند که تمام شبکه های میانی از ویژگیهای مشابهی برخوردار باشند. به عنوان مثال اگر تمام آنها تحویل مطمئن بسته های لایه شبکه را تضمین کرده باشند، آنگاه جریان بین مبدا و مقصد، مطمئن و قابل اعتماد خواهد بود (به استثنای وقتی که یکی از مسیربایهای میانی از کار بیفتند). به دلیل مشابه اگر هیچیک از شبکه های میانی تحویل مطمئن بسته ها را تضمین نکرده باشند، الحاق مدارات مجازی نیز نامطمئن خواهد بود. ولیکن اگر ماشین مبدا بر روی شبکه ای باشد که تحویل مطمئن بسته ها را تضمین کرده در حالی که یکی از شبکه های میانی استعداد از بین بردن بسته ای را داشته باشد، الحاق مدارات مجازی منجر به نامطمئن شدن کل مدار مجازی شده و طبیعت خدمات پیش بینی شده مثل تحویل مطمئن و حفظ ترتیب بسته ها تغییر می کند. الحاق مدارات مجازی در لایه انتقال نیز رایج است. بویژه این امکان وجود دارد که یک «خط انتقال بیت» (Bit Pipe) بین شبکه ای مثل SNA که به یک دروازه منتهی می شود و اتصالی TCP با دروازه دیگر دارد، ایجاد نمود. بدین ترتیب می توان یک مدار مجازی «انتها به انتها» (End To End) ایجاد کرد در حالی که چندین شبکه با پروتکل های مختلف در میانه راه قرار گرفته اند.

۵-۵-۵ بهم بندی شبکه های بدون اتصال (Connectionless Internetworking)

مدلی دیگر از بهم بندی شبکه ها، مدل دیتاگرام است؛ (به شکل ۵-۶ دقت کنید). در این مدل تنها خدمتی که لایه شبکه به لایه انتقال ارائه می دهد آن است که بسته های دیتاگرام را بر روی زیر شبکه توزیع کند؛ زیر شبکه نیز حداکثر تلاش خود را در جهت تحویل آن به عمل می آورد. در این مدل هیچگونه نشانی از مدار مجازی در سطح لایه شبکه وجود ندارد و فقط شبکه ها به هم متصل و ملحق می شوند. در این مدل نیازی نیست که بسته های متعلق به یک اتصال [تولید شده توسط یک ماشین] به ترتیب از دروازه های یکسانی بگذرند. در شکل ۵-۶ دیتاگرامهای ارسالی توسط ماشین ۱ که به سوی ماشین ۲ روانه شده اند از مسیرهای متفاوتی در شبکه عبور کرده اند. تصمیم گیری در خصوص مسیر هر بسته، بطور جداگانه انجام می شود و این تصمیم گیری بستگی به ترافیک لحظه ارسال بسته دارد. در این استراتژی از چندین مسیر بهره گرفته می شود و طبعاً پهنای باند بیشتری در مقایسه با مدل الحاق شبکه های مدار مجازی حاصل خواهد شد. ولی در مقابل تضمینی در به ترتیب رسیدن بسته ها به مقصد وجود ندارد، بلکه فقط فرض بر تحویل بسته ها است نه حفظ ترتیب آنها.

مدل شکل ۵-۶ به همین سادگی که بنظر می رسد نیست. اولین مورد اشکال آنکه، اگر هر یک از شبکه های میانی، پروتکل لایه شبکه خاص خودشان را داشته باشد، انتقال بسته از یک شبکه به شبکه ای دیگر ممکن نخواهد بود. شاید کسی تصور کند که یک مسیر یاب چند پروتکلی قادر به ترجمه قالب بسته ها به یکدیگر است در حالی که این تصور زمانی درست است که قالب بسته ها نزدیک به یکدیگر بوده و فیلدهای اطلاعاتی هر بسته مشابه باشند و در غیر این صورت چنین تبدیلی ناقص بوده و محکوم به شکست است. به همین دلیل به ندرت تلاش می شود چنین تبدیلی انجام گیرد.

دومین مشکل جدی، مسئله آدرس دهی است. یک حالت ساده را مدنظر قرار بدهید: یک ماشین بر روی شبکه،



شکل ۵-۴۶. بهم‌بندی شبکه‌های بدون اتصال.

اینترنت تلاش می‌کند یک بسته IP برای ماشینی بر روی یک شبکه SNA (متصل به اینترنت) بفرستد. آدرسهای IP و SNA متفاوت از هم هستند. آدرسهای IP و SNA باید در هر دو جهت به یکدیگر نگاشته و ترجمه شوند. مضاف بر این، در شبکه‌های متفاوت مفهوم «چیزهایی که قابل آدرس‌دهی هستند» فرق دارد. در IP ماشینهای میزبان (یا در حقیقت کارتهای واسطه شبکه) دارای آدرس هستند. در SNA هر «موجودیت» (Entity) مثل هر ابزار سخت‌افزاری می‌تواند آدرس داشته باشد. در بهترین حالت هر دروازه باید دارای یک پایگاه اطلاعاتی بوده و بتواند آدرسها را بهم‌بند (تبدیل کند) ولی همین کار منشاء بروز مشکلات جدی است.

یک نظریه دیگر آن است که یک بسته جهانی و استاندارد طراحی شود و تمام مسیریابها آنرا به رسمیت بشناسند. این راهکار در حقیقت همین IP است که بسته‌های آن به گونه‌ای طراحی شده که هر شبکه‌ای قادر به حمل و هدایت آنهاست. البته ممکن است به نظر برسد IPv4 (پروتکل فعلی اینترنت) نهایتاً تمام ساختارها و پروتکل‌های دیگر را از صحنه خارج می‌کند و IPv6 (پروتکل آینده اینترنت) نیز راه به جایی نمی‌برد و هیچ پروتکل جدیدی ابداع نمی‌شود! ولی تاریخ نشان داده که اینگونه نیست. جلب موافقت عموم افراد برای پذیرش یک قالب واحد بسیار دشوار است چرا که شرکتهای مختلف مصالح و سود خود را در آن می‌بینند که قالب و ساختار اختصاصی و تحت کنترل خود را داشته باشند.

حال بیا باید به اختصار مروری بر دو روش شبکه‌بندی داشته باشیم. مدل الحاق شبکه به روش مدار مجازی همان مزایایی را دارد که مدار مجازی در یک زیرشبکه واحد خواهد داشت: یعنی پیشاپیش می‌توان بافرها را از قبل رزرو کرد، ترتیب بسته‌ها حفظ می‌شود، سرآیند کوتاهتری برای بسته‌ها نیاز است و از مشکلاتی که ناشی از تکراری شدن بسته‌هایی که با تأخیر می‌رسند، احتراز می‌شود.

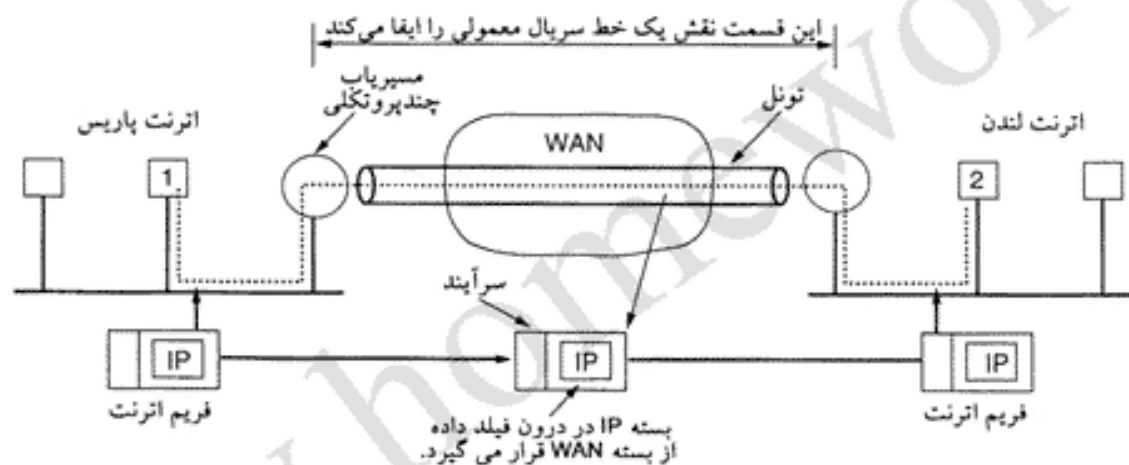
این روش معایبی نیز دارد: در مسیریابها به فضای قابل توجهی برای نگهداری جدول اتصالات باز نیاز است، برای احتراز از مناطق مواجهه با ازدحام مکانیزم مسیریابی و مسیرها تغییر نمی‌کند و مسیرها نسبت به خرابی مسیریابها بسیار آسیب‌پذیر هستند. همچنین این اشکال بزرگ وجود دارد که اتصال چنین شبکه‌ای به یک شبکه نامطمئن دیتاگرام اگر غیرممکن نباشد بسیار دشوار است.

ویژگی بهم‌بندی شبکه‌ها به روش دیتاگرام، دقیقاً مشابه با ویژگی زیرشبکه‌های دیتاگرام است: استعداد بروز ازدحام دارد ولی در عوض قابلیت بیشتری نیز برای رفع آن دارد، در مواجهه با خرابی یک مسیریاب قابلیت تحمل بیشتری از خود نشان می‌دهد و به سرآیند طولانی‌تری برای هر بسته نیاز است؛ استفاده از انواع الگوریتمهای وفقی و پویای مسیریابی میسر است و در مجموع تمام مزایا و معایب یک زیرشبکه واحد و منفرد را بطور مشابه دارد.

بزرگترین مزیت روش دیتاگرام برای بهم‌بندی شبکه آن است که می‌توان این روش را برای وصل هر شبکه‌ای که در درون از مدار مجازی بهره گرفته، استفاده کرد. بسیاری از شبکه‌های LAN، شبکه‌های متحرک (Mobile) مثل شبکه‌های ناوگانهای هوایی و دریایی و حتی برخی از شبکه‌های WAN، در رده شبکه‌های دیتاگرام قرار می‌گیرند و هر گاه استراتژی بهم‌بندی این شبکه‌ها مبتنی بر مدارات مجازی باشد، مشکلات جدی بروز خواهد کرد.

۵-۵-۵ ایجاد تونل (Tunneling)

اتصال دو شبکه مختلف در حالت کلی بسیار سخت است ولیکن یک حالت خاص و رایج وجود دارد که راهگشا و قابل مدیریت است. این حالت زمانی اتفاق می‌افتد که ماشینهای مبدا و مقصد بر روی شبکه‌هایی از یک نوع هستند ولی یک شبکه متفاوت در میان آنها قرار گرفته است. به عنوان مثال یک بانک بین‌المللی را مدنظر قرار دهید که یک شبکه اترنت مبتنی بر پروتکل TCP/IP در پاریس و همچنین یک اترنت با پروتکل TCP/IP در لندن دارد ولی به نحوی که در شکل ۵-۴۷ نشان داده شده، یک شبکه بغیر از IP (مثل ATM) در میان آنها قرار گرفته است.



شکل ۵-۴۷. ارسال یک بسته از طریق ایجاد تونل بین پاریس و لندن.

راهکار حل این مشکل تکنیکی به نام «ایجاد تونل» است. برای ارسال یک بسته IP به ماشین ۲، ماشین ۱، بسته‌ای حاوی آدرس IP ماشین ۲ ساخته و آنرا در درون فریم شبکه اترنت قرار داده و آدرس مسیریاب چند پروتکلی واقع در پاریس را در فیلد آدرس این فریم درج کرده و آن را بر روی اترنت قرار می‌دهد. هرگاه این «مسیریاب چندپروتکلی» این فریم را دریافت کند بسته IP را از درون فریم استخراج کرده و آن را در فیلد حمل داده (Payload) از بسته لایه شبکه WAN قرار داده و آدرس بسته جدید را آدرس مسیریاب چندپروتکلی واقع در لندن قرار می‌دهد. [به عبارتی دو بسته تودرتو تشکیل می‌شود که آدرس بسته درونی، آدرس ماشین مقصد و آدرس بسته بیرونی آدرس مسیریاب لندن است.] هرگاه این بسته به مسیریاب واقع در لندن برسد آن مسیریاب بسته IP اصلی را از درون آن استخراج کرده و آنرا با قرار دادن در یک فریم اترنت برای ماشین ۲ می‌فرستد.

شبکه WAN را می‌توان یک تونل بزرگ در نظر گرفت که از یک مسیریاب چندپروتکلی شروع و به مسیریابی دیگر از همین نوع، در طرف دیگر WAN ختم می‌شود. یک بسته IP در حالی که درون یک بسته دیگر جا گرفته از یک طرف تونل شروع به طی مسیر به طرف دیگر تونل می‌کند و جای هیچگونه نگرانی در خصوص ساختار WAN وجود ندارد [چرا که تنها کاری که WAN در این میان انجام می‌دهد آنست که مبتنی بر پروتکل فعلی خود، بسته IP را به عنوان یک قطعه داده خام، در یک نقطه دریافت و در نقطه دیگر تحویل می‌دهد. -م] در این مثال در خصوص ماشینهای هر یک از شبکه‌های اترنت نیز نگرانی وجود ندارد [چون هر دو از یک نوع و مبتنی بر پروتکل

مشابه هستند). فقط «مسیریاب چندپروتکلی» است که باید هم بسته های IP و هم بسته های WAN را بفهمد و مدیریت کند. در حقیقت کل شبکه ای که در میان این دو مسیریاب چندپروتکلی قرار گرفته شبیه به یک خط سریال ایفاء می کند.

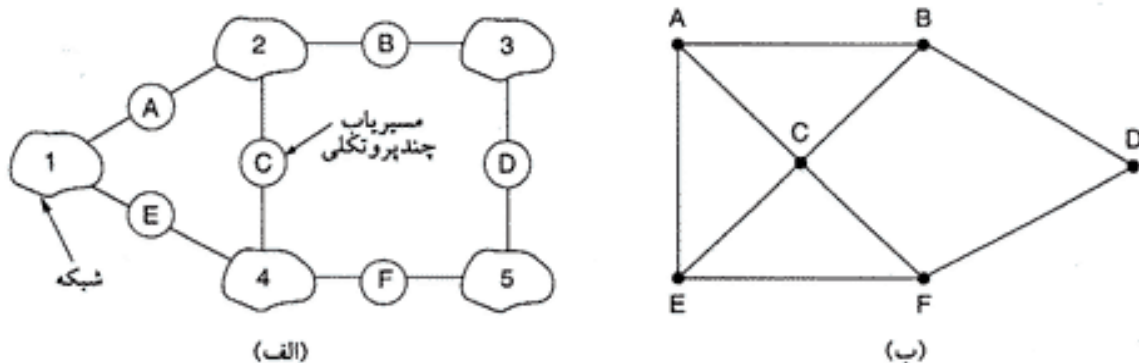
شاید یک تمثیل بتواند مفهوم تونل را روشنتر کند. شخصی را در نظر بگیرید که با خودروی خودش از پاریس به لندن مسافرت می کند. در فرانسه او با خودروی خود تا کنار کانال انگلیس رانندگی می کند ولی با رسیدن به این کانال (که رانندگی در آن ممنوع است)، خودروی او در یک قطار سریع السیر بار شده و از طریق قطار به انگلیس منتقل می شود. در حقیقت، به نحوی که در شکل ۵-۴۸ نشان داده شده، این خودرو همانند یک محموله عادی جابجا می شود. در طرف دیگر، این خودرو مجدداً بر زمین گذاشته شده و اجازه می یابد با نیروی محرکه خودش در جاده های انگلیس حرکت نماید. ایجاد تونل برای بسته ها از طریق یک شبکه خارجی مشابه با همین عملکرد است.



شکل ۵-۴۸. تونل کردن (Tunneling) یک خودرو از فرانسه به انگلستان.

۵-۶-۵ مسیریابی بین شبکه های بهم متصل

مسیریابی بین چند شبکه متصل بهم مشابه با مسیریابی در یک زیر شبکه واحد است (البته با پیچیدگی های بیشتر). به عنوان مثال ساختار ارتباطی شبکه الف-۴۹-۵ را در نظر بگیرید که در آن پنج شبکه توسط شش مسیریاب (احتمالاً از نوع چندپروتکلی) بهم متصل شده اند. تشکیل مدل گراف از این وضعیت پیچیده است چرا که در گراف، هر مسیریاب باید مستقیماً با مسیریاب دیگر در ارتباط باشد (بسته بفرستد) در حالیکه اینجا مسیریابها مستقیماً به یک شبکه میانی متصلند. مثلاً مسیریاب B در شکل ۵-۴۹-الف می تواند از طریق شبکه ۲ به مسیریابهای A و C دسترسی داشته باشد، همچنین مسیریاب D از طریق شبکه ۳ به B دسترسی دارد. اگر هر یک از شبکه ها را به مثابه یک خط سریال فرض نماییم، گراف شکل ۵-۴۹-ب بدست می آید.



شکل ۵-۴۹. (الف) «شبکه ای از شبکه ها» (ب) گراف متناظر.

به محض آنکه گراف شبکه تشکیل شد می‌توان یکی از الگوریتمهای شناخته شده مسیریابی مثل «الگوریتم بردار فاصله» یا «الگوریتم حالت لینک» (Link State Algorithm) را بر روی مجموعه این مسیریابهای چندپروتکلی اعمال کرد. در نتیجه یک الگوریتم مسیریابی دوسطحی ایجاد می‌شود: در درون هر شبکه از یک «پروتکل مسیریابی درونی» (Interior Gateway Protocol) استفاده می‌شود در حالی که بین شبکه‌ها از «پروتکل مسیریابی بیرونی» (Exterior Gateway Protocol) بهره گرفته می‌شود. (اصطلاح Gateway نام قدیمی مسیریاب است.) در حقیقت، چون شبکه‌ها مستقل هستند ممکن است از الگوریتمهای مسیریابی مختلفی در درون شبکه بهره گرفته باشند. از آنجایی که باهم‌پندی شبکه‌ها هر شبکه استقلال داخلی خود را حفظ می‌کند، به هر یک از این شبکه‌های مستقل «سیستم خودمختار» یا به اختصار AS گفته می‌شود.

در چنین شبکه‌ای، یک بسته نوعی از یک ماشین بر روی LAN مسیر خود را آغاز کرده و به آدرس «مسیریاب چندپروتکلی» متصل به آن LAN ارسال می‌شود (با قرار دادن آدرس آن مسیریاب در فیلد آدرس از فریم MAC). پس از آن که بسته به مسیریاب رسید، نرم‌افزار آن مسیریاب به کمک جدول مسیریابی خود مشخص می‌کند که این بسته باید به سوی کدامیک از مسیریابهای دیگر هدایت شود. اگر بسته بتواند با همان پروتکل لایه شبکه (که بسته براساس آن تولید شده) به مقصد رساند، این بسته مستقیماً هدایت و ارسال می‌شود. در غیر این صورت از مکانیزم ایجاد تونل (Tunneling) استفاده شده و کل بسته درون یک بسته متناسب با پروتکل مسیریابهای میانی جاسازی و ارسال می‌گردد. این فرآیند آنقدر تکرار می‌شود تا بسته به شبکه مقصد برسد.

یکی از تفاوت‌های بین مسیریابی درون یک شبکه خودمختار (Intranetwork Routing) و مسیریابی بین چند شبکه خودمختار آن است که در مورد دوم ممکن است بسته‌ها نیاز به تردد از مرزهای بین‌المللی کشورها داشته باشند. در چنین حالتی، ملاحظات قانونی متعدد مطرح خواهد شد چرا که مثلاً طبق قوانین کشور سوئد، خارج کردن اطلاعات فردی شهروندان سوئدی از کشور مجاز نیست. یا مثلاً طبق قوانین کشور کانادا، ترافیک داده‌هایی که مبداء آنها در کشور کانادا و مقصد آنها نیز در کانادا است نباید از کشور خارج شود. طبق این قانون ترافیک داده‌هایی که مبداء آن شهر «وینزور» یا «انتاریو» و مقصد آن «ونکوور» است نمی‌تواند از طریق مسیر نزدیکتری که از «دیترویت» می‌گذرد، مسیریابی و هدایت شود حتی اگر استفاده از این مسیر سریعتر و ارزاتر باشد.

تفاوت دیگر بین مسیریابی درونی و مسیریابی بیرونی (Interior/Exterior Routing) موضوع هزینه (قیمت) است. در یک شبکه خودمختار و واحد، بطور معمول از یک الگوریتم مشخص برای تعیین هزینه استفاده می‌شود ولیکن در شبکه‌های مختلف که مدیریت متفاوتی دارند ممکن است یک مسیر از لحاظ قیمت از مسیر دیگر مقرون به صرفه‌تر باشد. همینطور سطح کیفیت خدمات عرضه شده در شبکه‌های متفاوت فرق می‌کند و همین مورد ممکن است دلیل انتخاب یک مسیر از بین دیگر مسیرها باشد.

۷-۵-۵ قطعه‌سازی بسته‌ها (Fragmentation)

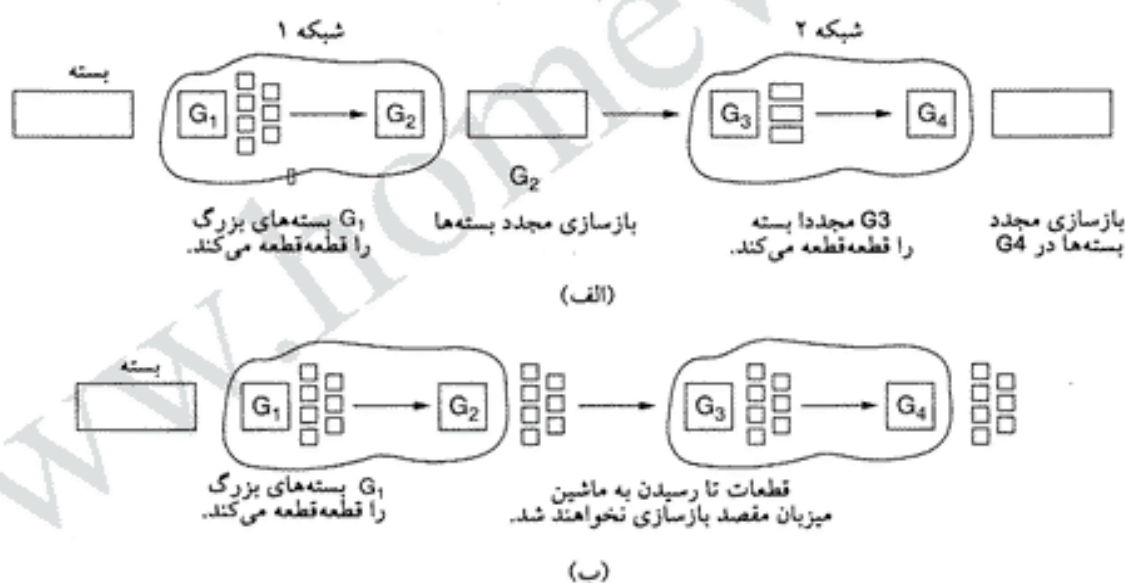
در هر شبکه اندازه هر بسته نمی‌تواند از یک حد مجاز بیشتر باشد. این محدودیت، علل مختلفی دارد که از این میان می‌توان به موارد ذیل اشاره کرد:

۱. سخت‌افزار (مثلاً طول فریم اترنت نمی‌تواند از حدود ۱۵۰۰ بایت بیشتر باشد).
۲. سیستم عامل (مثلاً بافرها ۵۱۲ بایتی هستند).
۳. پروتکل (مثلاً تعداد بیت‌های فیلدی که میزان داده‌های موجود در هر بسته را مشخص می‌کند کم است).
۴. سازگاری با برخی از استانداردهای بین‌المللی
۵. تمایل به کاهش حجم ارسال مجدد داده‌هایی که در اثر خطای انتقال از بین می‌روند
۶. اجتناب از اشغال بیش از حد کانال توسط یک بسته بزرگ

نتیجه این عوامل آن می شود که دست طراحان شبکه در انتخاب حداکثر طول بسته ها باز نیست. حداکثر طول داده هایی که می تواند درون یک بسته قرار بگیرد از ۴۸ بایت (در سلولهای ATM) تا ۶۵۵۱۵ بایت (در بسته های IP) متغیر است اگرچه طول داده ها در لایه های بالاتر، می تواند از این هم بیشتر باشد.

یکی از مشکلات بدیهی، زمانی رخ می دهد که یک بسته بزرگ بخواهد به شبکه ای وارد شود که طول حداکثر بسته آن کوچک است. یک راه حل آن است که از همان ابتدا این اطمینان حاصل شود که چنین مشکلی پیش نمی آید. به عبارت دیگر در اتصال شبکه ها، از یک الگوریتم مسیریابی استفاده شود که از ارسال بسته ها به شبکه ای که از عهده پذیرش آن بر نمی آید اجتناب کند، ولیکن این راه حل همیشه عملی نیست: اگر همه بسته های تولید شده توسط مبداء، دارای اندازه ای باشند که شبکه مقصد از عهده دریافت آن بر نمی آید چه اتفاقی می افتد؟ الگوریتم مسیریابی به ندرت می تواند چنین مقصدی را نادیده گرفته و آنرا کنار بگذارد.

اصولاً در چنین مواقعی می توان برای حل مشکل، به «دروازه» (Gateway) اجازه داد که بسته ها را به چند «قطعه» (Fragment) شکسته و هر یک از این قطعات را در پوشش یک بسته مستقل ارسال کند؛ ولیکن والدین کودکان خردسال به خوبی می دانند که تکه تکه کردن اشیاء بزرگ به قطعات کوچک ساده تر از سرهم آنهاست!!! (فیزیکدانان برای این پدیده نامی انتخاب کرده اند: قانون دوم ترمودینامیک) شبکه های سوئیچ بسته نیز برای بازسازی قطعات به بسته اصلی با مشکل روبرو هستند.



شکل ۵-۵۰. (الف) قطعه قطعه کردن نامرئی (شفاف) (ب) قطعه قطعه کردن غیر شفاف (مرئی).

برای بازسازی قطعات و تشکیل بسته اصلی، دو استراتژی متضاد قابل اعمال است: استراتژی اول آن است که قطعه قطعه سازی بسته ها در شبکه ای که بسته های کوچک دارد به گونه انجام شود که این عمل از دید مقصد نهایی بسته، پنهان بماند. در این راهکار که در شکل ۵-۵۰-الف نشان داده شده است، در هر شبکه با بسته کوچک یک «دروازه» (یا به عبارتی یک مسیریاب خاص) وجود دارد که واسطه دیگر شبکه ها است. وقتی بسته ای با طول بیش از حد به این دروازه می رسد، آن را قطعه قطعه می کند؛ سپس هر یک از قطعات به آدرس دروازه خروجی در مقصد ارسال شده و در آنجا قطعات به شکل اصلی بازسازی می شوند. بدین ترتیب عبور بسته های بزرگ از شبکه ای با بسته کوچک، از دید شبکه های بیرونی پنهان می ماند؛ به عبارتی شبکه های بیرونی از قطعه قطعه شدن بسته ها آگاه نمی شوند. به عنوان مثال، شبکه ATM دارای سخت افزار خاصی است که به صورت نامرئی، بسته ها را به سلولهای

۵۳ بایستی شکسته و در طرف مقابل آنها را به بسته اصلی بازسازی می نماید. در شبکه ATM به عمل شکستن بسته ها اصطلاحاً «قطعه بندی» (Segmentation) گفته می شود و مفهومی معادل با آنچه که در بالا بدان اشاره شد دارد، بلکه فقط جزئیات پیاده سازی آن متفاوت است.

استراتژی قطعه قطعه کردن بسته به صورت نامرئی، اگرچه ساده و سراسر است ولیکن مشکلاتی را در بر دارد. اولین مورد آن است که دروازه خروجی باید تشخیص بدهد که آیا تمام قطعات یک بسته را دریافت کرده است یا آنکه هنوز قطعاتی در راه هستند و به همین دلیل باید یک بیت خاص پایان دنباله قطعات هر بسته را مشخص نماید. مورد دیگر آن است که تمام قطعات یک بسته بایستی از یک دروازه مشابه و یکسان خارج شوند. وقتی اجازه نمی دهیم که قطعات یک بسته برای رسیدن به مقصد نهایی خود از مسیرهای مختلفی حرکت کنند، بخشی از کارایی و بهینگی را از دست خواهیم داد. آخرین اشکال آنست که وقتی یک بسته بزرگ مجبور است مکرراً از شبکه هایی بگذرد که آن را قطعه قطعه و بازسازی می کنند، سربار نسبتاً زیادی تحمیل خواهد شد. به هر تقدیر شبکه ای مثل ATM به قطعه قطعه سازی نامرئی [استراتژی فوق الذکر] نیاز دارد.

استراتژی دیگر برای قطعه قطعه کردن بسته ها آنست که از بازسازی بسته ها در دروازه های میانی اجتناب شود: هرگاه بسته ای قطعه قطعه شد، با هر یک از قطعات به مثابه یک بسته واقعی و اصلی رفتار شود. در این استراتژی به نحوی که در شکل ۵-۵-۵-ب نشان داده شده تمام قطعات به همان نحو از دروازه خروجی شبکه گذر می کنند. بازسازی بسته ها فقط در ماشین مقصد انجام می شود. IP به همین نحو عمل می کند.^۱

قطعه قطعه سازی غیرشفاف نیز با اشکالاتی مواجه است. به عنوان مثال، هر ماشین باید قادر به بازسازی بسته باشد. مشکل دیگر آنست که وقتی یک بسته بزرگ به قطعات کوچک تقسیم می شود، سربار داده ها افزایش می یابد زیرا هر قطعه نیاز به سرآیند مستقل دارد، در حالی که در استراتژی اول به محض خروج قطعات از شبکه ای با بسته کوچک، سربار تحمیل شده حذف می شود ولیکن در این روش سربار ناشی از سرآیند اضافی، تاخاتمه مسیر باقی خواهد ماند. مزیت روش قطعه قطعه سازی غیرشفاف آن است که قطعات هر بسته می توانند از مسیری مجزا و دروازه های خروجی متفاوت به سوی مقصد نهایی خود هدایت شوند و طبعاً کارایی بالاتر و بهینه تری حاصل خواهد شد. البته اگر شبکه های مختلف طبق مدل مدار مجازی به یکدیگر متصل و ملحق شده باشند [بخش ۵-۵-۳] این مزیت بکار نخواهد آمد.

هرگاه بسته ای قطعه قطعه شود، قطعات آن باید به نحوی شماره گذاری شوند که بتوان ترتیب اصلی داده ها را بازیابی کرد. یکی از روشهای شماره گذاری، روش درختی است: اگر بسته شماره ۰ نیاز به قطعه شدن داشته باشد قطعات به صورت ۰.۰، ۰.۱، ۰.۲ و به همین ترتیب، شماره گذاری می شوند. اگر هر یک از این قطعات، باز هم نیاز به شکسته شدن داشته باشند ترتیب شماره ها (از چپ به راست) به صورت ۰.۰.۰، ۰.۰.۱، ۰.۰.۲، ...، ۰.۱.۰، ۰.۱.۱، ۰.۱.۲، ۰.۱.۳، ... خواهد بود. اگر برای این منظور به تعداد کافی فیلد در سرآیند بسته پیش بینی شده باشد، الگوی فوق این اطمینان را می دهد که بتوان بسته ها را در مقصد بازسازی کرد. (حتی اگر قطعات به ترتیب دریافت نشوند.)

ولی اگر در یکی از شبکه های بین راه قطعه ای گم شود یا حذف گردد نیاز است که مجدداً کل بسته از مبداء آن ارسال و از نو قطعه قطعه و ارسال شود. فرض کنید که یک بسته ۱۰۲۴ بیتی در ابتدا به چهار قطعه مساوی با شماره های ۰.۰، ۰.۱، ۰.۲ و ۰.۳ تقسیم شده باشد. اگر قطعه ۰.۱ از دست برود ولی بقیه قطعات سالم به مقصد

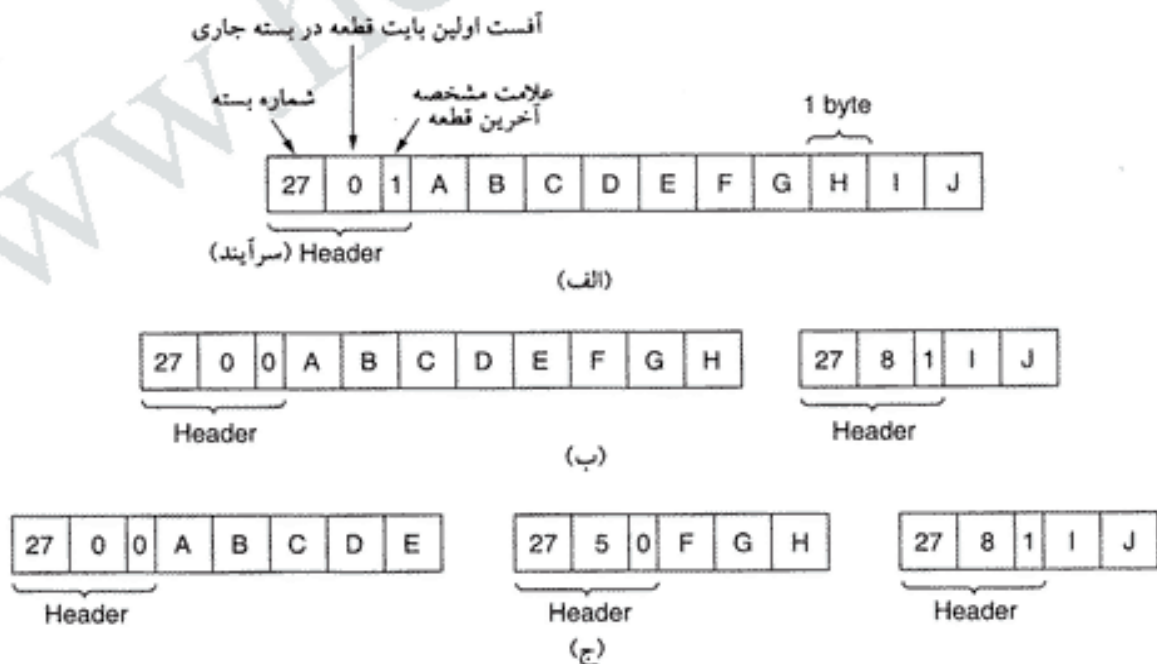
۱. بدین استراتژی، قطعه قطعه سازی غیرشفاف و مرئی (Nontransparent Fragmentation) گفته می شود. هرگاه بسته ای در هر نقطه از مسیر قطعه قطعه شد هیچکسی آنها را بازسازی نخواهد کرد مگر ماشین مقصد نهایی. -م

برسند، پس از مدتی مهلت فرستنده بسته به سر آمده و کل بسته را از نو ارسال می کند. حال فرض کنید که به ناگاه محدودیت طول بسته از ۲۵۶ بیت به ۵۱۲ بیت افزایش پیدا کرده و در اینجا بسته به جای آنکه ۴ قسمت شود به دو قطعه تقسیم گردد. حال وقتی بسته شماره ۰.۱ به مقصد می رسد گیرنده تصور می کند که تمام چهار قطعه ای که منتظر آن بوده، تکمیل شده و بسته را به صورت اشتباه بازسازی می کند!

یک روش شماره گذاری متفاوت و بهتر آن است که در پروتکل شبکه، یک اندازه پایه و حداقل برای بسته ها تعریف شود و این اندازه به قدری کوچک باشد که بسته بتواند از هر شبکه ای عبور نماید. هرگاه بسته ای قطعه قطعه می شود، اندازه تمام قطعات به مقدار پایه تنظیم خواهد شد (به استثنای قطعه آخر که ممکن است کوچکتر باشد). در سرآیند هر یک از قطعات باید شماره بسته اصلی و همچنین شماره قطعه مشخص شده باشد. همچنین باید در سرآیند قطعه یک بیت در نظر گرفته شود تا بتوان آخرین قطعه هر بسته را مشخص نمود. (یعنی هر بسته را فقط یکبار می توان قطعه قطعه کرد.) در این روش، در سرآیند هر بسته به دو فیلد شماره ترتیب نیاز است: یکی برای شماره بسته اصلی و دیگری برای شماره قطعه.

می توان یک حالت بینابین برای شماره گذاری بسته ها انتخاب کرد: به جای آن که برای هر قطعه شماره آن در سرآیند هر قطعه درج شود، آفست آن قطعه در بسته اصلی، مشخص شود. [یعنی آن که مشخص شود قطعه جاری در کجای بسته اصلی قرار می گیرد.] اگر مطابق شکل ۵-۵۱ بجای شماره هر قطعه، آفست محل قرار گرفتن قطعه در بسته اصلی در نظر گرفته شود می توان بسته را به هر اندازه ممکن (حتی یک بایت) کوچک کرد.

در برخی از پروتکل های شبکه بطور عام از این روش استفاده شده است حتی تا جایی که کل داده های ارسالی بر روی یک مدار مجازی به عنوان یک بسته بسیار بزرگ تلقی می شود و شماره هر قطعه، شماره ترتیب «اولین بایت قطعه» نسبت به «اولین بایت بسته» فرض می شود. [یعنی محل قرار گرفتن قطعه نسبت به شماره اولین بایت بسته اصلی مشخص می شود.]



شکل ۵-۵۱. قطعه سازی بسته ها با فرض آنکه مبنای پایه طول داده ها ۱ بایت باشد. (الف) بسته اصلی حاوی ده بایت داده. (ب) مجموعه قطعات پس از عبور از شبکه ای که در آن طول حداکثر بسته ها با احتساب سرآیند، ۸ بایت است. (ج) مجموعه قطعات پس از عبور از «دروازه ای» (Gateway) که در آن طول حداکثر بسته ها، ۵ بایت است.

۵-۶ لایه شبکه در اینترنت

قبل از آنکه به توصیف لایه شبکه در اینترنت بپردازیم، مروری بر اصول طراحی آن در گذشته و دلایل موفقیت آن در حال حاضر، خالی از لطف نیست. اصولی که به نظر می رسد اکنون به دست فراموشی سپرده شده اند. این اصول و قواعد در سند RFC 1958 تشریح شده اند و مطالعه آن بسیار ارزشمند است. (مطالعه آن برای طراحان پروتکل باید الزامی شود و در پایان نیز از آنان براساس همین سند امتحان به عمل آید!) این RFC پشتات تحت تأثیر افکاری است که دو محقق Clark (۱۹۸۸) و Saltzer (همکاران و ۱۹۸۴) ارائه نموده اند. در اینجا به اختصار ده مورد از اصول اساسی طراحی پروتکل لایه شبکه را (به ترتیب اهمیت) بررسی می نماییم:

۱. اطمینان از عملکرد صحیح: طراحی یا استاندارد را هیچگاه نهایی و مختومه فرض نکنید مگر آنکه چندین نسخه اولیه و آزمایشی (پروتوتایپ) آن بتوانند با یکدیگر مبادله داده نمایند. بسیار اتفاق افتاده که طراحان شبکه یک استاندارد ۱۰۰۰ صفحه ای تدوین و آنرا به تایید رسانده اند ولی بعداً متوجه اشکالات اساسی و عدم عملکرد آن شده و مجبور به نوشتن نسخه ۱.۱ استاندارد خود شده اند. این روش به نتیجه نخواهد رسید.

۲. پروتکل را ساده طراحی کنید: در هر کجا که تردید دارید، از ساده ترین راه حل بهره بگیرید. «ویلیام اوگم» این اصل را در قرن چهاردهم بیان کرده است. در یک عبارت امروزی و مدرن: «حداقل ویژگیهای زائد». اگر داشتن یک ویژگی، حیاتی و اساسی نیست آنرا رها کنید بالاخص زمانی که این ویژگی را می توان براساس ترکیبی از ویژگیهای دیگر بدست آورد.

۳. تصمیمات روشن و شفاف بگیرید: اگر برای انجام یک کار چندین راه حل پیش رو دارید فقط یکی را انتخاب نمایید. اگر دو یا چند روش را مدنظر قرار بدهید برای خود مشکل تراشیده اید. امروزه بسیاری از استانداردها دارای گزینه های متعدد، حالات و پارامترهای مختلف هستند چراکه هر یک از طراحان، بر آنکه روش آنها بهترین است، پافشاری کرده اند. طراحان باید با قدرت در برابر این گرایشات فردی مقاومت کرده و فقط بگویند نه!!

۴. طراحی شما مابجولار باشد: این اصل بدین نظریه منتهی خواهد شد که باید پشته ای از پروتکلها را داشت و هر لایه مستقل از دیگر لایه ها باشد. بدین ترتیب هر گاه نیاز شد یک مابجول یا یک لایه تغییر کند، بقیه لایه ها تحت تأثیر قرار نخواهند گرفت.

۵. ناهمگونی عناصر را مدنظر قرار بدهید: در یک شبکه بزرگ، انواع سخت افزار، امکانات مخابراتی و برنامه های کاربردی پیدا می شود. برای آن که بتوان همه آنها را به خدمت گرفت و اداره کرد، طراحی شبکه باید ساده، عمومی و قابل انعطاف باشد.

۶. از گزینه ها و پارامترهای ثابت پرهیز نمایید: اگر در جایی الزاماً به تعریف پارامتر نیاز دارید (مثلاً تعریف طول حداکثر بسته ها) بهترین کار آن است که به جای تعریف پارامترهای ثابت اجازه بدهید فرستنده و گیرنده با یکدیگر مذاکره و بر سر مقدار آن توافق کنند.

۷. به دنبال طراحی خوب باشید: لازم نیست که یک طراحی خوب ایده آل و بی نقص باشد: بسیاری از طراحان، طرح خوبی را در اختیار دارند ولی طرح آنان نمی تواند از عهده برخی از موارد نادر و عجیب برآید. به جای آن که اینگونه طرحها را بهم بریزید بایستی در پی یک طراحی خوب بوده و از خود در مقابل خواسته های عجیب و غریب سلب مسئولیت نمایید.

۸. هنگام «ارسال» سخت گیر و دقیق و هنگام «دریافت» با تحمل باشید: به عبارتی دیگر فقط بسته هایی را

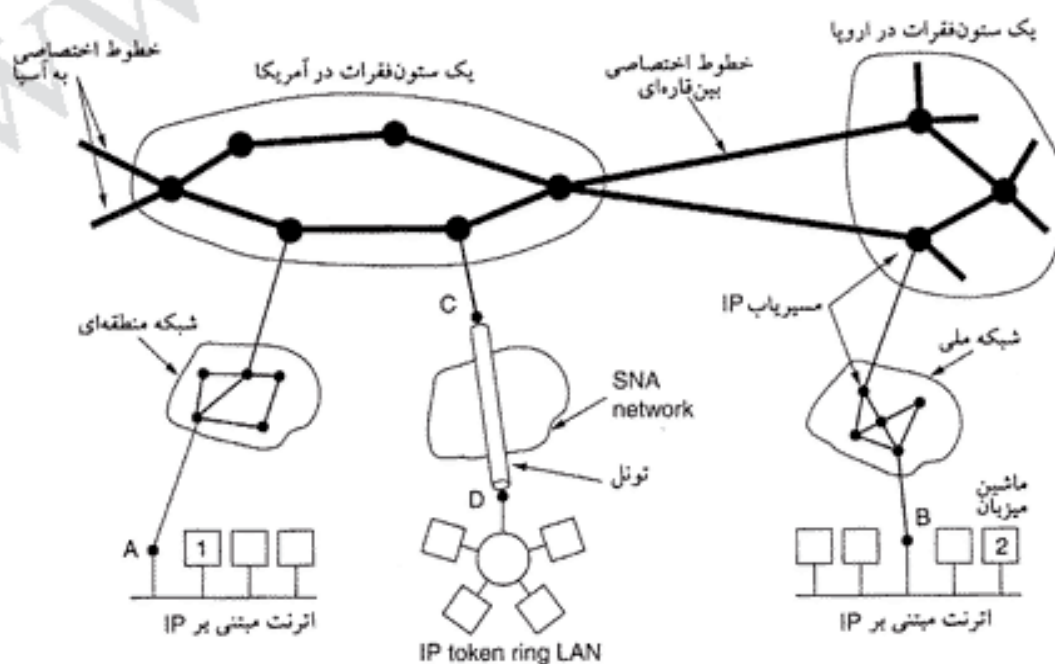
بر روی شبکه بفرستید که دقیقاً منطبق با استاندارد است ولی انتظار آن را داشته باشید که بسته‌های دریافتی انطباق کامل با استاندارد نداشته باشد.

۹. در اندیشه قابلیت گسترش و توسعه پذیری باشید: اگر یک سیستم مجبور به مدیریت میلیون‌ها ماشین و میلیارد‌ها کاربر است، استفاده از هیچ نوع پایگاه اطلاعات مرکزی قابل تحمل و به صلاح نیست و باید بار آن تا حد ممکن بر روی منابع موجود پخش شود.

۱۰. به کارایی و هزینه دقت داشته باشید: اگر شبکه‌ای کارایی پایین یا هزینه سرسام‌آور داشته باشد هیچکس از آن استقبال نخواهد کرد.

حال اصول کلی طراحی را کنار گذاشته و به جزئیات لایه شبکه در اینترنت خواهیم پرداخت. از دیدگاه لایه شبکه، اینترنت را می‌توان مجموعه‌ای از زیرشبکه‌ها یا اصطلاحاً «شبکه‌های خودمختار» در نظر گرفت که بهم متصل شده‌اند. هیچ ساختار مشخص و قطعی بر اینترنت حاکم نیست ولی چندین ستون فقرات (Backbone) برای آن وجود دارد. ستون فقرات از خطوط با پهنای باند بسیار بالا و مسیرهای سریع تشکیل شده است. شبکه‌های منطقه‌ای (Regional Network) به ستون فقرات اینترنت متصل می‌شوند و شبکه‌های محلی دانشگاه‌ها، شرکت‌ها و مؤسسات ارائه دهنده خدمات اینترنت (ISP) با شبکه‌های منطقه‌ای در ارتباط هستند. نمادی از این ساختار نسبتاً سلسله‌مراتبی در شکل ۵-۵۲ نشان داده شده است.

آنچه که تمام اجزاء شبکه اینترنت را به هم چسبانده است همان پروتکل IP است. (IP/Internet Protocol) برخلاف بسیاری از پروتکل‌های قدیمی لایه شبکه، این پروتکل از صفر طراحی شده و از همان ابتدا در تفکر وصل شبکه‌ها به یکدیگر بوده است. برای آن که بتوانید وظیفه لایه شبکه را در ذهن خود مجسم کنید بدان بیندیشید که این لایه حداکثر تلاش خود را می‌کند تا یک دیتاگرام را از مبدا به مقصد برساند. [دیتاگرام را یک توده اطلاعات مستقل و دارای هویت در نظر بگیرید.] IP رسیدن بسته‌ها را تضمین نمی‌کند (بلکه حداکثر تلاش خود را مصروف آن می‌کند) و اینکه آیا ماشینهای مبدا و مقصد بر روی یک شبکه واقعند یا آن که شبکه‌های دیگری در این بین قرار گرفته‌اند، مهم نیست.

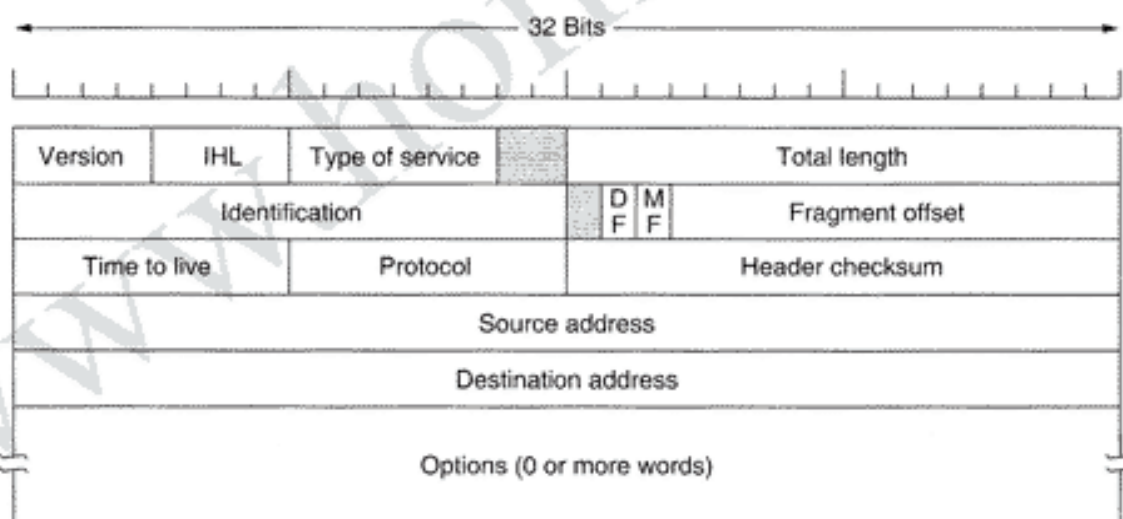


شکل ۵-۵۲. اینترنت مجموعه تعداد بی‌شماری شبکه است.

ارتباط در اینترنت بدین ترتیب است که: لایه انتقال یک دنباله از داده ها را تحویل گرفته و آنها را به چند دیتاگرام تقسیم می کند. در تئوری، طول حداکثر هر دیتاگرام می تواند تا ۶۴ کیلوبایت باشد ولی در عمل، بطور معمول حول و حوش ۱۵۰۰ بایت است (تا بتواند در یک فریم اترنت جا بگیرد). هر دیتاگرام از طریق اینترنت ارسال می شود و در صورت نیاز، در طول مسیر به قطعات کوچکتر شکسته می شود. وقتی تمام قطعات به ماشین مقصد رسیدند توسط لایه شبکه (IP) به دیتاگرام اصلی بازسازی می شود. این دیتاگرام تحویل لایه انتقال شده و توسط این لایه به پروسه گیرنده تسلیم می شود. همانگونه که در شکل ۵-۵۲ می بینید، بسته ای که از ماشین ۱ منشأ گرفته برای رسیدن به ماشین ۲ باید از شش شبکه بگذرد. در عمل این تعداد از شش تا هم بیشتر است.

۵-۶-۱ پروتکل IP

برای شروع به تحلیل پروتکل IP، بهترین کار آن است که قالب بسته های IP را بررسی نماییم.^۱ یک بسته IP در برگرفته دو بخش سرآیند و بخش محتوی است. سرآیند هر بسته دارای یک بخش ثابت ۲۰ بیتی و یک بخش اختیاری با طول متغیر است. ساختار سرآیند بسته IP در شکل ۵-۵۳ نشان داده شده است. ترتیب ارسال بسته IP از بایت پرارزش به کم ارزش است یعنی بایتها به ترتیب از چپ به راست ارسال می شوند؛ بنابراین اولین فیلدی که ارسال می شود فیلد Version (شماره نسخه پروتکل) است. ماشینهای SPARC داده ها را از پرارزش به کم ارزش ذخیره و ارسال می کنند و به آنها ماشینهای Big Endian گفته می شود در حالیکه ماشینهایی مثل پنتیوم برعکس هستند یعنی داده ها را از کم ارزش به پرارزش ذخیره و ارسال می کنند (ماشینهای Little Endian) و بالطبع نرم افزار اینگونه ماشینها باید قبل از ارسال یا پس از دریافت، تبدیل لازم را انجام بدهند.



شکل ۵-۵۳. سرآیند IPv4 در پروتکل اینترنت.

فیلد Version مشخص می کند که بسته براساس چه نسخه ای از پروتکل IP سازماندهی و ارسال شده است. با قرار دادن شماره نسخه پروتکل در هر دیتاگرام می توان فرآیند گذر از یک نسخه قدیمی به نسخه جدید را به صورت تدریجی و در خلال چندین سال به انجام رساند و در این مدت برخی از ماشینها از نسخه قدیمی و برخی از نسخه جدید استفاده کنند. اکنون در حال گذر از IPv4 به IPv6 هستیم و اگرچه سالهاست که انتظار این جایگزینی می رود ولی به نظر می رسد هنوز هم به سالها وقت نیاز دارد.

۱. به بسته های IP اصطلاحاً «دیتاگرام IP» گفته می شود. «دیتاگرام IP» یک قطعه داده دارای هویت و شناسنامه است. -م

(Durand, 2001; Wiljakka, 2002; Waddington and Chang, 2002) حتی برخی افراد معتقدند این تغییر هیچگاه محقق نخواهد شد. (Weiser, 2001) مضاف بر این، IPv5 نیز یک بعنوان پروتکل آزمایشی برای انتقال بی درنگ استریم داده^۱ طراحی گردید ولیکن استفاده چندان از آن نشد.

از آنجایی که طول سرآیند ثابت نیست لذا فیلد IHL بدان منظور در نظر گرفته شده که مشخص کند طول سرآیند (بر مبنای کلمات ۳۲ بیتی) چقدر است. مقدار حداقل این فیلد (یعنی زمانی که هیچ داده‌ای در بخش اختیاری سرآیند وجود ندارد) معادل ۵ است. حداکثر مقدار این فیلد ۴ بیتی، ۱۵ است و طبعاً طول سرآیند به ۶۰ بایت (۱۵×۴) محدود می‌شود؛ یعنی حداکثر طول بخش اختیاری ۴۰ بایت (۶۰-۲۰) است. این فضای ۴۰ بیتی برای برخی از گزینه‌هایی که می‌توان درون این فیلد اختیاری درج کرد (مثل گزینه ثبت مسیر طی شده توسط بسته) بسیار ناکافی است و عملاً فیلد اختیاری را بی‌استفاده کرده است.

فیلد Type of Service (نوع خدمات)، یکی از معدود فیلدهایی است که مضمون و عملکرد آن در طول این سالها تغییر کرده است. از ابتدا (و همچنین اکنون) این فیلد به منظور مشخص کردن کلاسهای مختلف خدمات مدنظر بوده است. در این فیلد می‌توان ترکیبی از خدمات مثل قابلیت اعتماد (Reliability) و سرعت (Speed) را برای بسته تقاضا کرد. برای انتقال دیجیتال صدا، تحویل سریع داده‌ها ارجح تر از تحویل مطمئن و بدون خطاست. برعکس، برای انتقال فایل ارسال بدون خطا بسیار مهمتر از سرعت انتقال است.

فیلد شش بیتی Type of Service (به ترتیب از چپ به راست) در برگیرنده یک فیلد سه بیتی به نام Precedence (فیلد تقدم) و سه بیت علامت به نامهای D و T و R است. فیلد Precedence اولویت بسته را مشخص می‌کند: از صفر (اولویت معمولی) تا ۷ (بالاترین اولویت برای بسته‌های کنترلی شبکه). سه بیت علامت به ماشین میزبان اجازه می‌دهد که از بین سه ویژگی (D: تأخیر، T: ظرفیت خروجی، R: قابلیت اعتماد) نیازهای خود را توصیف نماید. از دیدگاه تئوری این فیلدها امکان آنرا فراهم آورده‌اند که مسیریاب مثلاً از بین یک خط ماهواره‌ای با ظرفیت خروجی بالا و تأخیر زیاد و همچنین یک خط اجاره‌ای با ظرفیت خروجی کم و تأخیر پایین، انتخاب مناسبی داشته باشد، ولیکن در عمل مسیریابها فیلد Type of service را نادیده می‌گیرند.

عاقبت IETF متقاعد شد که مضمون این فیلد را تغییر بدهد تا براساس آن بتوان رده‌های مختلف خدمات مورد نیاز را توصیف کرد. ۶ بیت این فیلد برای مشخص کردن رده کلاس خدماتی است که بسته بدان کلاس تعلق دارد. این کلاسها که قبلاً در بخش ۵-۴-۴ معرفی شدند، شامل: ۴ اولویت صف‌بندی، ۳ احتمال حذف بسته و تعدادی کلاس برای سازگاری با قبل هستند.

فیلد Total Length طول کل بسته را (شامل سرآیند و داده) مشخص می‌نماید. حداکثر طول یک بسته ۶۵۵۳۵ بایت است. در حال حاضر این مقدار حداکثر کفایت می‌کند ولی در آینده با رواج اترنت گیگابیت ممکن است به دیتاگرام با طول بیشتری نیاز باشد.

فیلد Identification بدان جهت نیاز است که مشخص کنیم قطعه دریافتی، به کدام دیتاگرام متعلق است. تمام قطعات یک دیتاگرام واحد دارای مقداری مشابه در فیلد Identification هستند. [بخش ۵-۵-۷ را مطالعه نمایید]. در ادامه یک بیت بلااستفاده و سپس یک فیلد تک بیتی به نام DF^۲ قرار گرفته است. این بیت به مسیریابها فرمان می‌دهد که دیتاگرام (بسته) جاری را قطعه‌قطعه نکنند چرا که مقصد از بازسازی آن عاجز است. مثلاً وقتی

۱. Real-Time Stream Protocol.

۲. DF مخفف کلمات Don't Fragment به معنای عدم قطعه‌قطعه‌سازی بسته است.

کامپیوتری بوت می شود، ROM بوت کننده آن ممکن است تقاضا کند که «تصویری از حافظه» (Memory Image) برای او ارسال شود.^۱ با علامتگذاری در بیت DF، فرستنده مطمئن خواهد شد که کل بسته به صورت یکجا تحویل خواهد شد، حتی اگر این بسته به دلیل محدودیتهایی که مسیر بهینه (در هدایت بسته های بزرگ) دارد مجبور به عبور از مسیرهای غیر بهینه شود. تمام ماشینها ملزم به پذیرش قطعات با طول ۵۷۶ بایت (یا کمتر) هستند. بیت MF (مخفف More Fragment) به معنای آنست که دنباله قطعات هنوز ادامه دارد. در تمام قطعات به استثنای قطعه آخر این بیت ۱ است. به این بیت از آن جهت نیاز است که متوجه شویم آیا تمام قطعات یک دیتاگرام دریافت شده یا هنوز دنباله قطعات ادامه دارد.

فیلد Fragment Offset مشخص کننده آنست که قطعه جاری در کجای دیتاگرام اصلی واقع شده است. طول تمام قطعات (به استثنای قطعه آخر) باید ضربی از ۸ بایت باشد. از آنجایی که این فیلد ۱۳ بیتی است حداکثر می توان ۸۱۹۲ قطعه در هر دیتاگرام داشت و بدین ترتیب طول حداکثر هر دیتاگرام ۶۵۵۳۶ بایت خواهد شد.

فیلد Time to live (زمان حیات بسته) شمارنده ای است که طول عمر بسته را تعیین می نماید. فرض بر آنست که این فیلد زمان را برحسب ثانیه مشخص کند و طول عمر بسته حداکثر ۲۵۵ ثانیه باشد. به ازای هر گام (یعنی عبور بسته از یک مسیر یاب)، یک واحد از این فیلد کاسته می شود. همچنین فرض شده که هر گاه بسته درون یک مسیر یاب، زمان زیادی را در صف منتظر ماند تعداد بیشتری از این فیلد کم شود. با تمام این تفصیلات، در عمل فقط تعداد گام محاسبه می شود.^۲ به محض آن که مقدار این فیلد به صفر برسد بسته حذف شده و یک پیغام هشدار به مبدا آن برگردانده می شود. این ویژگی از سرگردان ماندن بسته ها در زیر شبکه (که در اثر اشتباه در جدول مسیر یابی بروز می کند) جلوگیری می نماید.

هر گاه لایه شبکه یک دیتاگرام کامل را بازسازی کرد باید بداند که چه کاری با آن بکند. فیلد Protocol (شماره پروتکل)، مشخص می کند که بسته را باید به کدام پروتکل در لایه انتقال تحویل داد. پروتکل TCP یکی از گزینه هاست؛ ولی پروتکل UDP و چندین پروتکل دیگر نیز می توانند دیتاگرام را تحویل بگیرند. شماره پروتکلها جهانی و در سرتاسر اینترنت استاندارد هستند. پروتکلها و دیگر شماره های استاندارد در سند RFC 1700 ثبت شده است؛ این شماره ها را می توانید در پایگاه اطلاعاتی www.iana.org بدست بیاورید.

فیلد Header Checksum یک کد کشف خطا برای سرآیند است. این کد کشف خطا که صرفاً سرآیند بسته را در برمی گیرد قادر است خطاهایی را که از خرابی در حافظه مسیر یاب ناشی می شود آشکار نماید. الگوریتم کشف خطا بدین ترتیب است که کل سرآیند، در همان بدو ورود در قالب نیم کلمات ۱۶ بیتی با یکدیگر جمع شده و نهایتاً «مکمل یک»^۳ آن محاسبه می شود. اگر سرآیند بسته بدون خطا باشد جمع تمام نیم کلمات ۱۶ بیتی سرآیند، باید صفر باشد. این الگوریتم قدرتمندتر از جمع معمولی است. دقت کنید که فیلد Header Checksum در هر گام باید از نو محاسبه شود چرا که حداقل یکی از فیلدهای سرآیند تغییر می کند (یعنی فیلد Time To Live) ولی برای سرعت بخشیدن به این محاسبه می توان از راهی میانبر استفاده کرد.

فیلدهای Source Address (آدرس مبدا) و Destination Address (آدرس مقصد) شماره شبکه و شماره ماشین مبدا و مقصد را مشخص می کنند. این دو آدرس را در بخشی مجزا بررسی خواهیم کرد.

فیلد Options بدان منظور طراحی شده تا در نسخه های بعدی پروتکل بتوان اطلاعاتی را در سرآیند قرار داد که در نسخه اصلی آن پیش بینی نشده است. همچنین می توان از آن برای آزمودن ایده های جدید بدون درهم

۱. تصویری از هسته سیستم عامل که بصورت کدهای اجرایی ارسال می شود.

۲. یعنی امروزه فقط به ازای عبور بسته از یک مسیر یاب، یک واحد از مقدار این فیلم کم می شود و زمان (برحسب ثانیه) هیچ

تاثیری در این فیلد ندارد. ۳. One's Complement

ریختن آن بهره گرفت. فیلد Option دارای طولی متغیر است؛ هر یک از گزینه های مندرج در این فیلد با یک کد یک بایتی شروع می شود تا ماهیت گزینه را مشخص کند. در برخی از گزینه ها پس از این کد یک بایتی، فیلد یک بایتی دیگر، طول گزینه را برحسب بایت مشخص می نماید. نهایتاً به فیلد Options تعدادی بایت اضافی و زائد افزوده می شود تا طول آن ضریبی از ۴ شود. در ابتدا فقط پنج گزینه فهرست شده در شکل ۵-۵۴ تعریف شده بود ولی پس از آن تعدادی گزینه دیگر بدان افزوده شد. فهرست کامل این گزینه ها در آدرس www.iana.org/assignments/ip-parameters در دسترس عموم قرار گرفته است.

گزینه	توصیف عملکرد
Security	میزان محرمانه بودن دیتاگرام جاری را تعیین می کند.
Strict source routing	فهرست کامل مسیری را که باید دنبال شود تعریف می کند.
Loose source routing	فهرستی از مسیریابها که بسته حتماً باید از آنها رد شود.
Record route	مسیریابها را وادار می کند تا آدرس IP خود را در بسته درج کنند.
Timestamp	مسیریابها را وادار می کند تا آدرس IP خود را به همراه مهر زمان در بسته درج کنند.

شکل ۵-۵۴. برخی از گزینه های IP.

گزینه Security مشخص کننده آن است که اطلاعات موجود در بسته تا چه حد محرمانه است. از دیدگاه تنوری، کاربرد این گزینه آن است که مسیریابهای نظامی این بسته را از مسیرهایی که از کشورها یا مناطقی می گذرند که به زعم آنها نامطمئن و خطرناک هستند، هدایت نکنند. در عمل تمام مسیریابها این فیلد را نادیده می گیرند و تنها کاربرد عملی آن می تواند کمک به جاسوسان برای انتخاب بسته های مورد نظرشان باشد!

با گزینه Strict Source Routing می توان مسیر کامل بین مبدا و مقصد را در قالب دنباله ای از آدرسهای IP مشخص کرد. بسته ارسالی ملزم به عبور از این مسیر است. این گزینه به مدیران شبکه کمک می کند تا در هنگام خرابی جداول مسیریابی یا عملکرد غیر صحیح مسیریابها بتوانند بسته های اضطراری خود را به مقصد برسانند یا می تواند برای اندازه گیری زمانی و تخمین ترافیک یک مسیر مفید واقع شود.

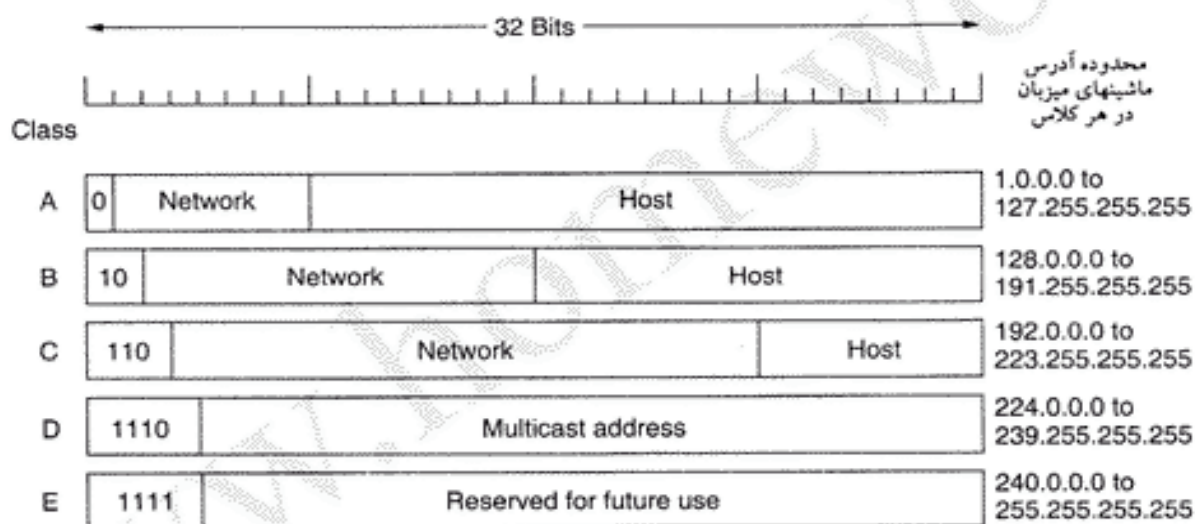
با گزینه Loose Source Routing، بسته ملزم به عبور از مسیریابهای مشخص (طبق ترتیب تعیین شده) می باشد ولی اجازه دارد در مسیر خود از مسیریابهای دیگری نیز که در فهرست نیامده، عبور کند. [یعنی در حقیقت فقط برخی از نقاط مسیر مشخص می شود و بقیه نقاط توسط مسیریاب انتخاب می شود.] بطور معمول در این گزینه فقط تعدادی از مسیریابها مشخص می شوند و همین تعداد می تواند مسیر مورد نظر را تبیین نماید. به عنوان مثال برای آنکه بسته ای را مجبور کنیم برای رفتن از لندن به سیدنی به جای شرق از سمت غرب حرکت کند کافی است در این گزینه سه مسیریاب در نیویورک، لوس آنجلس و هانولولو مشخص شود. این گزینه زمانی سودمند خواهد بود که به دلایل سیاسی یا اقتصادی بخواهیم بسته ها از مسیرهایی که در برخی کشورهای خاص واقعند عبور نکنند.

گزینه Record Route (ثبت مسیر) به مسیریابهای واقع بر روی مسیر تفهیم می کند که باید آدرس IP خود را در همین فیلد (فیلد Option) درج نمایند. این گزینه به مدیران شبکه کمک می کند تا بتوانند اشکالات احتمالی در الگوریتم مسیریابی را پیگیری و کشف نمایند (مثلاً: «چرا بسته ای که از هیوستون به دالاس روانه شده در توکیو دیده شده است؟») زمانی که برای اولین بار ARPANET راه اندازی شد کل مسیریابهای این شبکه ۹ تا بیشتر نبود و فضای چهل بایتی فیلد Options کفایت می کرد ولی امروزه به دلیل رشد سرسام آور شبکه اینترنت این فضای چهل بایتی بسیار ناکافی و عملاً بی کاربرد است.

در آخر، گزینه Timestamp (مهر زمان) عملکردی شبیه به گزینه قبلی (یعنی Record Route) دارد با این تفاوت که به غیر از ثبت آدرس IP هر مسیر، یک «مهر زمان» ۳۲ بیتی نیز در کنار آن درج می شود. این گزینه نیز برای اشکال زدایی از الگوریتمهای مسیریابی کاربرد دارد.

۲-۶-۵ آدرسهای IP

هر ماشین میزبان و هر مسیر، در شبکه اینترنت دارای یک آدرس IP است که در آن شماره شبکه و شماره ماشین، مشخص می شود. ترکیب این دو شماره، منحصر به فرد و یکتا است: بر اساس این اصل اساسی که هیچ دو ماشینی در اینترنت نباید آدرس IP یکسانی داشته باشند. تمام آدرسهای IP، ۳۲ بیتی هستند و برای مشخص کردن آدرس ماشین مبدأ و مقصد، در درون فیلدهای مربوطه در بسته های IP قرار می گیرند. اشاره به این نکته بسیار مهم است که آدرس IP در حقیقت یک ماشین میزبان را مشخص نمی کند بلکه به یک «کارت واسط شبکه» اشاره دارد، فلذا اگر ماشینی بر روی دو شبکه واقع شده باشد باید دو آدرس IP داشته باشد ولی در عمل بیشتر ماشینها فقط بر روی یک شبکه قرار گرفته اند و طبعاً یک آدرس IP بیشتر ندارند.^۱



شکل ۵-۵۵. قالب آدرسهای IP.

در چند دهه ابتدایی، آدرسهای IP در پنج رده مختلف دسته بندی شده بودند. این پنج رده در شکل ۵-۵۵ نشان داده شده است. تخصیص فضای آدرس مطابق با این روش، اصطلاحاً روش «آدرس دهی دارای کلاس» (Classful Addressing) نامیده می شود. این روش امروزه کاربرد خود را از دست داده است [از حدود سال ۱۹۹۶] ولی هنوز هم در کتب مختلف به آن اشاره می شود. این روش آدرس دهی را به اختصار توضیح می دهیم. کلاسهای A و B و C و D این امکان را فراهم آورده اند تا بتوان به ترتیب: ۱۲۸ شبکه با ۱۶ میلیون ماشین میزبان (در کلاس A)، ۱۶۳۸۴ شبکه با ۶۵۵۳۶ ماشین (در کلاس B) و حدود ۲ میلیون شبکه با ۲۵۶ ماشین (در کلاس C) را آدرس دهی کرد (اگرچه برخی از این آدرسها برای کارهای ویژه در نظر گرفته شده اند و به هیچ ماشینی متسبب نمی شوند). همچنین در کلاس D از ارسال چندپخش (Multicast) پشتیبانی می شود و می توان بسته ای را مستقیماً برای چندین ماشین در شبکه فرستاد.

۱. مسیریابها که چندین کارت واسط دارند بیش از یک آدرس IP خواهند داشت.

آدرسهای که با چهار بیت ۱۱۱۱ شروع می‌شوند (یعنی کلاس E) هیچ استفاده‌ای ندارند و برای کاربردهای بعدی رزرو شده‌اند. امروزه بیش از ۵۰۰,۰۰۰ شبکه به اینترنت متصل شده‌اند و این تعداد سال به سال افزایش می‌یابد. شماره‌های شبکه [یعنی فیلد Network در آدرس IP] توسط یک موسسه غیرانتفاعی به نام ICANN^۱ مدیریت می‌شود تا تناقضی در انتساب شماره‌های تکراری پدید نیاید. ICANN تخصیص بخشهایی از فضای آدرس دهی را به تعدادی «مراکز مجاز منطقه‌ای» واگذار کرده تا آنها آدرسهای IP را بین سرویس دهنده‌های اینترنتی (ISP) و دیگر شرکتها توزیع نمایند.

آدرسهای IP که ۳۲ بیتی هستند معمولاً با نماد دهدهی نقطه‌دار نمایش داده می‌شوند. در این روش هر یک از چهار بایت آدرس به صورت مجزا و در مبنای ده (از صفر تا ۲۵۵) نوشته می‌شود. مثلاً آدرس ۳۲ بیتی C0290614 (در مبنای شانزده) به صورت 192.41.6.20 نمایش داده می‌شود. کمترین آدرس IP معادل 0.0.0.0 و بزرگترین آن معادل 255.255.255.255 است.

مقدار صفر و ۱- (یعنی فیلدی که تمام بیت‌هایش ۱ است) معانی خاص دارد. شکل ۵-۵۶ آدرسهای خاص را مشخص کرده است. مقدار ۰ خود ماشین یا شبکه را مشخص می‌نماید.^۲ مقدار ۱- برای پخش فراگیر (Broadcast) کاربرد دارد، یعنی اگر بیت‌های آدرس مقصد بسته تماماً ۱ باشد، گیرنده بسته، تمام ماشینهای آن شبکه می‌باشد.

0 0	This host (همین ماشین)
0 0 . . . 0 0	Host A host on this network (ماشینی بر روی همین شبکه)
1 1	Broadcast on the local network (پخش فراگیر بر روی شبکه محلی)
Network 1 1 1 1 . . . 1 1 1 1	Broadcast on a distant network (پخش فراگیر بر روی شبکه راه دور)
127 (Anything)	Loopback (آدرس حلقه بازگشت)

شکل ۵-۵۶. آدرسهای IP خاص.

آدرس IP معادل 0.0.0.0 توسط ماشینهای میزبان و در حین راه‌اندازی (بوت) مورد استفاده قرار می‌گیرد. هر گاه شماره شبکه در آدرس IP صفر باشد چنین آدرسی به شبکه فعلی اشاره می‌کند. این آدرسها اجازه می‌دهد که ماشینها بدون دانستن شماره شبکه خود، به آن اشاره کنند [یعنی اگر ماشین بخواهد برای ماشین دیگری در شبکه خودش بسته‌ای بفرستد، دانستن شماره شبکه، مهم نیست]. البته باید هر ماشین، حداقل کلاس آدرس شبکه خود را بداند تا تشخیص بدهد فیلد شماره شبکه چند بیتی است و آن را با صفر پر کند. آدرسی که تمام بیت‌های آن ۱ است اجازه می‌دهد که بسته به صورت فراگیر بر روی شبکه محلی پخش شود. [یعنی بسته‌ای با چنین آدرسی را همه ماشینهای موجود بر روی شبکه محلی دریافت می‌دارند]. اگر شماره شبکه با یک مقدار درست تنظیم شده ولی تمام بیت‌های شماره ماشین مساوی ۱ باشد می‌توان بسته‌ای را برای تمام ماشینهای یک شبکه LAN، در هر کجای اینترنت ارسال نمود. (البته بسیاری از مسئولین شبکه این ویژگی را غیرفعال و ناممکن کرده‌اند).

۱. Internet Corporation for Assigned Names and Numbers

۲. یعنی اگر فیلد Network در آدرس IP معادل صفر باشد به معنای خود شبکه و اگر فیلد Host صفر باشد به معنای خود آن ماشین است. بسته‌ای با این آدرسها از شبکه یا ماشین بیرون نخواهد رفت. -۳

تمام آدرسهای که به شکل 127.xx.yy.zz هستند به نام «حلقه بازگشت» (Loopback) مشهورند و برای آزمایش نرم افزار، کنار گذاشته شده اند: هر بسته ای که به چنین آدرسهایی ارسال شود به صورت واقعی بر روی سیم منتقل نخواهد شد بلکه به صورت داخلی پردازش شده و با آن همانند بسته ورودی رفتار می شود. این آدرس اجازه می دهد بدون آنکه ماشین فرستنده شماره خود را بداند برای خودش بسته بفرستد.

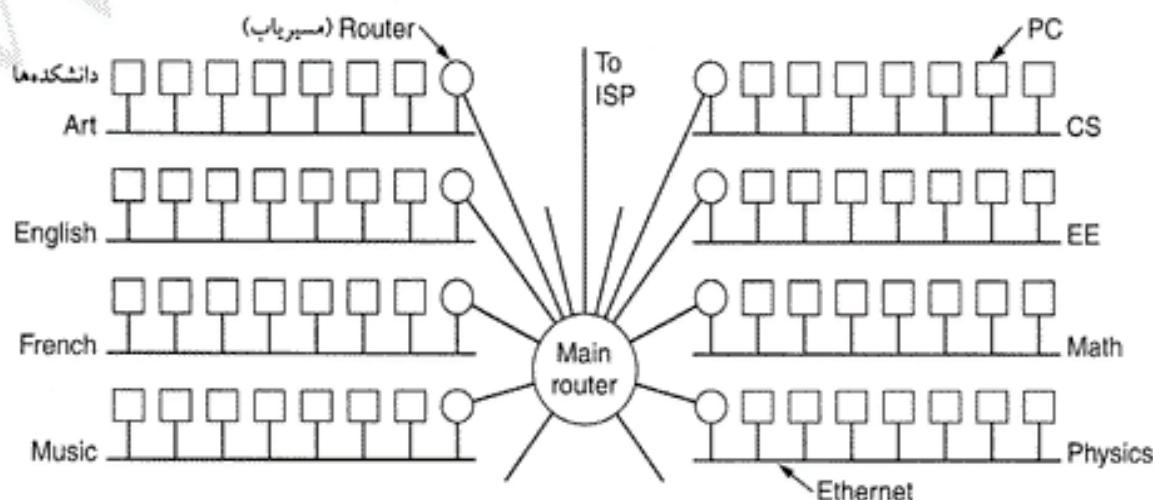
زیرشبکه ها

همانگونه که قبلاً بررسی کردیم تمام ماشینهای یک شبکه باید دارای شماره شبکه یکسانی باشند. با رشد شبکه، این ویژگی در آدرس دهی IP، باعث بروز مشکلاتی می شود. به عنوان مثال دانشگاهی را مدنظر قرار بدهید که در ابتدا با یک آدرس کلاس B شروع کرده و آن را در شبکه اترنت متعلق به دانشکده علوم کامپیوتر بکار گرفته است. یک سال بعد دانشکده مهندسی برق می خواهد که به اینترنت متصل شود و با خرید یک تکرارکننده (Repeater) سعی می کند به شبکه اترنت دانشکده علوم کامپیوتر متصل شود. با گذشت زمان، دانشکده های دیگر نیز تقاضای وصل به شبکه می کنند و بدین ترتیب به دلیل محدودیت در تعداد تکرارکننده های اترنت این کار غیرممکن می شود. بنابراین به سازماندهی متفاوتی نیاز است.

از طرف دیگر، ثبت و دریافت یک کلاس آدرس مجزا برای شبکه های جدید نیز دشوار است چراکه آدرسهای شبکه محدود و کم هستند و مضاف بر این، آدرس کلاس B اختصاص داده شده به این دانشگاه برای آدرس دهی به بیش از ۶۰۰۰۰ ماشین کافی است. مشکل اینجا است که یک کلاس A، B یا C به یک شبکه واحد اشاره می کند نه به یک مجموعه از شبکه های LAN متصل بهم.

پس از آنکه سازمانهای متعددی به مشکل کمبود آدرس شبکه، دچار شدند یک تغییر بسیار کوچک در این آدرسها مشکل را حل کرد: راه حل آن است که اجازه بدهیم شبکه به چندین بخش مستقل داخلی تقسیم شده ولی کماکان در دنیای خارج به عنوان یک شبکه واحد تلقی شود.

یک «شبکه دانشگاهی» (Campus)، نمادی شبیه به شکل ۵-۵۷ دارد که در آن یک مسیریاب مرکزی از طرفی به یک ISP (یا یک شبکه منطقه ای) و از طرف دیگر به چندین شبکه اترنت در دانشکده های مختلف وصل شده است. هر یک از شبکه های اترنت دارای یک مسیریاب محلی هستند که از طریق آن به مسیریاب مرکزی متصل شده اند. (البته ممکن است به جای مسیریاب مرکزی یک ستون فقرات قرار گرفته باشد ولیکن چگونگی اتصال مسیریابها در این مثال مهم نیست).



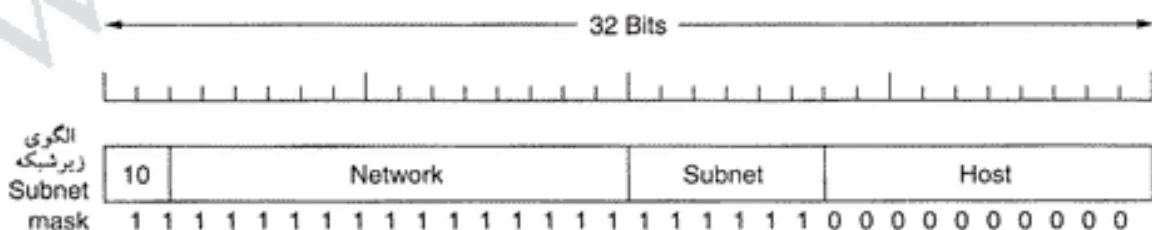
شکل ۵-۵۷. یک شبکه دانشگاهی که شامل چندین LAN متعلق به دانشکده های مختلف است.

در ادبیات شبکه اینترنت، به هر بخش از یک شبکه بزرگ و خودمختار، اصطلاحاً «زیرشبکه» (Subnet) گفته می‌شود. (مثلاً در مثال بالا هر یک از شبکه‌های اترنت یک «زیرشبکه» تلقی می‌شوند.) همانطور که در فصل اول اشاره کردیم، این استعاره می‌تواند با واژه «زیرشبکه» به معنای مجموعه‌ای از مسیرهای و خطوط ارتباطی در یک شبکه اشتباه شود. البته زمینه و مفاد یک بحث، معنای مورد نظر آن را مشخص خواهد کرد. در این بخش و بخش آتی، هر گاه واژه «زیرشبکه» را بکار بردیم، تعریف جدید آن مدنظر است.

وقتی بسته‌ای به مسیر یاب مرکزی وارد می‌شود، مسیر یاب چگونه تشخیص می‌دهد که آنرا به کدام زیرشبکه (اترنت) تحویل بدهد؟ یک راه آن است که جدولی با ۶۵۵۳۶ درایه (Entry) داشته باشیم و هر درایه محل هر ماشین میزبان را در کل شبکه مشخص کند. این ایده عملی است ولی به یک جدول بسیار بزرگ در مسیر یاب مرکزی نیاز است و در صورت اضافه شدن، حذف یا تغییر محل ماشینها، باید تنظیمات لازم بصورت دستی انجام شود.

به جای این روش، الگوی جدیدی ابداع شده است. در این روش به جای آنکه یک کلاس B (با ۱۴ بیت برای شماره شبکه و ۱۶ بیت برای شماره ماشین) داشته باشیم، بخشی از فضای شانزده بیتی شماره ماشین برای تعیین «شماره زیرشبکه» در نظر گرفته می‌شود. به عنوان مثال اگر دانشگاه ۳۵ دانشکده داشته باشد، می‌تواند ۶ بیت را برای شماره زیرشبکه و ده بیت باقیمانده را برای شماره ماشین در نظر بگیرد. بدین ترتیب می‌تواند حداکثر ۶۴ زیرشبکه اترنت با حداکثر ۱۰۲۲ ماشین میزبان داشته باشد. (قبلاً اشاره شد که شماره‌های ۰ یا ۱- کاربرد خاص داشته و قابل استفاده نیست). این تقسیم بندی را بعداً می‌توان تغییر داد.

برای تعریف و پیاده‌سازی زیرشبکه‌ها، مسیر یاب مرکزی به یک «الگوی زیرشبکه» (Subnet Mask) نیاز دارد تا مطابق با شکل ۵-۵۸ بتوان مرز بین شماره ماشین و شماره شبکه و شماره زیرشبکه را مشخص و تفکیک کرد. الگوی زیرشبکه (Subnet Mask) نیز به صورت نماد نقطه‌دار ده‌دهی نوشته می‌شود. همچنین برای راحتی نمایش، می‌توان پس از آدرس IP یک علامت "3" گذاشت و پس از آن تعداد بیت‌های ۱ موجود در الگوی زیرشبکه را تعیین کرد. به عنوان مثال در شکل ۵-۵۸ الگوی زیرشبکه را می‌توان به صورت 255.255.252.0 نوشت. راه دیگر آن است که از نماد 22/ استفاده نماییم که مشخص کننده آن است که الگوی زیرشبکه ۲۲ بیت طول دارد. (۲۲ بیت ۱ از سمت چپ به راست)



شکل ۵-۵۸. یک کلاس B به ۶۴ زیرشبکه بخش بندی شده است.

خارج از محدوده این شبکه، این چنین تقسیم بندی و تعداد زیرشبکه‌ها مشهود و مشخص نیست لذا تعریف زیرشبکه جدید مستلزم آن نیست که با ICANN تماس گرفته شده یا تغییری در پایگاه اطلاعات خارج از آن شبکه داده شود. در این مثال، اولین زیرشبکه از آدرسهای استفاده می‌کند که مثلاً از 130.50.4.1 شروع می‌شوند. آدرسهای دومین زیرشبکه از 130.50.8.1 و آدرس سومین زیرشبکه از 130.50.12.1 شروع می‌شوند. برای آنکه ببینیم چرا شماره زیرشبکه‌ها چهار تا چهار تا اضافه می‌شود، شکل دودویی این آدرسها را در نظر می‌گیریم:

زیر شبکه ۱	10000010	00110010	000001 00	00000001
زیر شبکه ۲	10000010	00110010	000010 00	00000001
زیر شبکه ۳	10000010	00110010	000011 00	00000001

در الگوی بالا خط عمودی که به صورت | نشان داده شده مرز بین شماره زیر شبکه و شماره ماشین را مشخص می‌نماید. در سمت راست، ۶ بیت برای شماره زیر شبکه و درست چپ ۱۰ بیت برای شماره ماشین در نظر گرفته شده است.

بررسی عملکرد زیر شبکه‌ها مستلزم آنست که تشریح کنیم بسته‌های IP در مسیریابها چگونه پردازش می‌شوند. هر مسیریاب جدولی را در اختیار دارد که در آن مجموعه‌ای از آدرسهای IP به صورت زوجهای (۰، شماره شبکه) و (شماره ماشین، همین شبکه) درج شده است. نوع اول مشخص می‌کند که چگونه می‌توان به یک شبکه راه دور رسید. نوع دوم، آدرس ماشینهای محلی را [که مستقیماً به آن مسیریاب متصلند] مشخص می‌کند. همچنین در این جدول، آدرس کارت واسط شبکه برای رسیدن به هر یک از این زیر شبکه‌ها (و پاره‌ای اطلاعات اضافی) درج شده است.

وقتی یک بسته IP توسط یک مسیریاب دریافت می‌شود، آدرس مقصد آن در جدول مسیریابی جستجو می‌شود. اگر مقصد بسته یک شبکه راه دور باشد از طریق یکی از کارتهای واسط به سوی مسیریاب بعدی هدایت می‌شود. اگر مقصد بسته یکی از ماشینهای محلی باشد (مثلاً یکی از ماشینهای LAN متصل به مسیریاب)، مستقیماً برای آن ماشین ارسال می‌شود. اگر مقصد بسته وجود نداشته یا شناسایی نشود، بسته به سوی یک مسیریاب پیش فرض (Default Router) هدایت می‌شود. این مسیریاب دارای جدول مسیریابی کاملتر بوده و احتمالاً یک «مسیریاب مرزی» است. پس طبق این الگوریتم، هر مسیریاب فقط باید مشخصات دیگر شبکه‌ها (یا به عبارت بهتر زیر شبکه‌ها) و همچنین ماشینهای محلی خود را بداند و بدین ترتیب اندازه جدول مسیریابی شدت کاهش می‌یابد.

وقتی زیر شبکه جدیدی تعریف می‌شود، جدول مسیریابی تغییر می‌کند و در آن درایه‌هایی به شکل (0، شماره زیر شبکه، شماره همین شبکه) و (0، شماره همین زیر شبکه، شماره همین شبکه)^۱ اضافه می‌شود. بدین ترتیب یک مسیریاب که به زیر شبکه k متصل است فقط می‌داند که چگونه باید به زیر شبکه‌های دیگر برسد و همچنین آدرس تمام ماشینهای متصل به زیر شبکه خودش (یعنی زیر شبکه k) را می‌داند. بنابراین مسیریابها مجبور نیستند جزئیات آدرس ماشینهای واقع در زیر شبکه‌های دیگر را بدانند. در حقیقت تنها کاری که مسیریاب در برخورد با آدرسهای IP متعلق به ماشینهای خارج از زیر شبکه خود انجام می‌دهد آنست که: آدرسهای IP را با «الگوی زیر شبکه» (Subnet Mask) به صورت بولین AND می‌کند تا بخش شماره ماشین آن حذف شود. سپس آدرس حاصل را در جدول خود جستجو می‌کند. به عنوان مثال فرض کنید بسته‌ای به آدرس 130.50.15.6 ارسال و توسط مسیریاب مرکزی (شکل ۵.۵۷) دریافت شده باشد. مسیریاب مرکزی این آدرس را با الگوی زیر شبکه 255.255.252.0/22 به صورت بولین AND می‌کند و نتیجه 130.50.12.0 بدست می‌آید. این آدرس در جدول مسیریابی جستجو می‌شود تا خط خروجی مناسب برای رسیدن به زیر شبکه ۳ مشخص گردد. تعریف زیر شبکه در حقیقت با ایجاد یک سلسله مراتب سه سطحی حجم جدول مسیریابی را کاهش می‌دهد. این سلسله مراتب سه سطحی تشکیل شده از: شماره شبکه + شماره زیر شبکه + شماره ماشین می‌زبان.^۲

۱. (Network, 0) (this-network, host)

۲. (this-network, subnet, 0) (this-network, this subnet, host)

۳. مسیریابی سلسله‌مراتبی را در بخش ۵-۲-۶ از همین فصل مطالعه کنید.

CIDR^۱: مسیریابی براساس آدرسهای بدون کلاس

چندین دهه است که از پروتکل IP استفاده گسترده‌ای می‌شود. عملکرد این پروتکل بسیار عالی بوده و همانگونه که اشاره کردیم به رشد نمایی شبکه اینترنت کمک کرده است. متأسفانه IP در حال قربانی شدن در پای محبوبیت خود است چراکه با کمبود فضای آدرس مواجه شده است. از زمانی که افق این مشکل پدیدار شد، بحثها و مناقشات بسیار گسترده‌ای را در جامعه اینترنت دامن زد که چه کاری می‌توان برای آن انجام داد. در بخش جاری، این مشکل و راه‌حلهای پیشنهادی را تشریح می‌نماییم.

تا سال ۱۹۸۷ افراد دوراندیشی که حدس می‌زدند در روزگاری تعداد شبکه‌های متصل به اینترنت به مرز ۱۰۰,۰۰۰ برسد، انگشت‌شمار بودند و حتی اغلب متخصصین این فن نیز چنین ایده‌ای را به مسخره می‌گرفتند ولی صدهزارمین شبکه دنیا در سال ۱۹۹۶ به اینترنت پیوست و به نحوی که در بالا اشاره شد اینترنت در حال مواجه شدن با کمبود آدرسهای IP است. در اصل بیش از دو میلیارد آدرس IP وجود دارد ولی در عمل به دلیل تقسیم‌بندی نامناسب فضای آدرس در قالب چند کلاس، میلیونها آدرس IP به هدر رفته است. (شکل ۵-۵۵ را ببینید). بالاخص، بیشترین محدودیت در کلاس B است. برای بسیاری از مؤسسات یک شبکه کلاس A با شانزده میلیون آدرس، بسیار بزرگ و بی‌مصرف و کلاس C با ۲۵۶ آدرس بسیار کوچک و ناکافی است؛ فقط شبکه کلاس B با ۶۵۵۳۶ آدرس مناسب است. این مشکل در فرهنگ رایج اینترنت به نام مشکل «سه خرس» مشهور شده است. (برگرفته از داستان موطلایی و سه خرس)

حقیقت آنست که حتی کلاس B هم برای بسیاری از سازمانها و مؤسسات، بسیار بزرگ است. بررسیها نشان داده که بیش از نیمی از شبکه‌های کلاس B، کمتر از ۵۰ ماشین دارند! برای چنین شبکه‌هایی استفاده از کلاس C کفایت می‌کند ولی بی‌تردید تمام مؤسساتی که متقاضی کلاس B بوده‌اند بدان می‌اندیشیده‌اند که روزی فیلد هشت بیتی کلاس C برای شبکه آنها کافی نخواهد بود. نگاهی به گذشته نشان می‌دهد که بهتر آن بوده در کلاس C فیلد شماره شبکه به جای هشت بیت، ده بیت می‌بود تا در هر شبکه ۱۰۲۲ ماشین میزبان قابل آدرس دهی باشد. اگر چنین حالتی را می‌داشتیم شاید بسیاری از مؤسسات و سازمانها به کلاس C اکتفا می‌کردند و در عین حال نیم میلیون از چنین شبکه‌هایی قابل تعریف بود. (برخلاف کلاس B که فقط ۱۶۳۸۴ شبکه قابل تعریف است).

نمی‌توان تقصیر را به گردن طراحان اینترنت انداخت که چرا تعداد کلاسهای B را بیشتر (و با فضای آدرس کوچکتر) در نظر نگرفتند. در روزگاری که تصمیم گرفته شد فقط سه کلاس وجود داشته باشد، اینترنت شبکه‌ای تحقیقاتی بود که فقط مراکز پژوهشی و دانشگاهی مهم ایالات متحده را بهم متصل می‌کرد. (البته به اضافه معدود شرکتها و سایتهای نظامی که آنها نیز در کار پژوهش بودند). هیچکس نمی‌توانست پیش‌بینی کند که اینترنت روزی بتواند در حد وسیع و به عنوان سیستمی ارتباطی حتی با شبکه‌های تلفن رقابت نماید! در آن زمان هیچکس در این ادعا شکی نداشت که اگر تمام ۲۰۰۰ کالج و دانشگاه ایالات متحده و حتی اکثر دانشگاههای جهان به اینترنت بپیوندند باز هم ۱۶۰۰۰ تا نمی‌شود چراکه این تعداد دانشگاه در کل دنیا وجود نداشت. به علاوه در آن زمان برای آنکه پردازش بسته‌ها سریعتر انجام بشود بهتر بود فیلد شماره ماشین در آدرس IP، تعداد صحیحی از بایت باشد. [به دلیل سرعت پایین پردازنده‌ها در آن زمان، پردازش بیتی فیلدهای آدرس، از سرعت مسیریابی می‌کاست].

ولیکن در طرف مقابل اگر مثلاً برای فیلد شماره شبکه در کلاس B، بیست بیت کنار گذاشته می‌شد مشکل دیگری بروز می‌کرد: «مشکل رشد انفجاری جداول مسیریابی». از دید مسیریابها فضای آدرسهای IP، سلسله مراتب دوسطحی شامل شماره شبکه و شماره ماشینها است. مسیریابها مجبور به دانستن شماره ماشینهای میزبان

۱. CIDR : Classless InterDomain Routing

نیستند ولی باید شماره شبکه ها را بدانند. اگر حتی نیمی از شبکه های کلاس C بکار گرفته شده باشند هر مسیریاب در کل اینترنت نیاز به جدولی با نیم میلیون درایه (Entry) دارد تا بتواند خط خروجی مناسب برای رسیدن به هر شبکه را مشخص نماید. (گذشته از اطلاعات دیگری که به ازای هر شبکه باید در جدول مسیریابی درج شود). شاید تهیه فضای فیزیکی لازم برای ذخیره نیم میلیون درایه (Entry) در جدول مسیریابی، امکان پذیر باشد هر چند برای مسیریابهایی که جداول مسیریابی را در حافظه نوع ایستا (Static RAM) ذخیره می کنند، چنین فضایی بسیار گران تمام می شود. مشکل اساسی در پیچیدگی الگوریتمهای مدیریت و پردازش چنین جدولی است. پیچیدگی زمانی این الگوریتمها غیرخطی است.^۱ از این بدتر آنکه نرم افزار یا سخت افزار طراحی شده برای مسیریابهای موجود، زمانی طراحی شده که به اینترنت بیش از هزار شبکه متصل نبود و به نظر می رسید که یک دهه طول می کشد تا این تعداد به ۱۰۰۰۰ برسد. امروزه اینگونه طراحیها بهینه نیستند.

مضاف بر این، در الگوریتمهای مختلف مسیریابی نیاز است که جداول مسیریابی بطور متناوب ارسال و مبادله شود. (مثل الگوریتم بردار فاصله) هر چه جداول مسیریابی بزرگتر باشد احتمال آنکه بخشی از آن در حین مبادله از دست برود بیشتر خواهد شد و به نقص داده های جدول مسیریابی و احتمالاً ناپایداری فرآیند هدایت بسته ها خواهد انجامید.

مشکل جداول مسیریابی را می توان با افزایش «سطوح سلسله مراتب» حل کرد. مثلاً می توان آدرسهای IP را بدین نحو تعریف کرد که شامل فیلدهای کشور، ایالت / استان، شهر، شبکه و شماره ماشین میزبان باشد. بدین ترتیب مسیریابهای ستون فقرات در جهان فقط باید در خصوص مسیرهای رسیدن به هر کشور آگاهی داشته باشند، مسیریابهای درون کشور در خصوص مسیرهای رسیدن به هر ایالت یا استان، مسیریابهای ایالت یا استان در مورد مسیرهای رسیدن به هر شهر و مسیریابهای هر شهر فقط باید راه رسیدن به هر شبکه را بدانند. متأسفانه چنین راه حلی نیازمند فضایی بزرگتر از ۳۲ بیت برای آدرس IP است و طبعاً از فضای آدرس استفاده بهینه نخواهد شد. (چرا که مثلاً در این الگو کشور لیختن اشتاین به همان تعداد بیت در آدرس IP دارد که کشور ایالات متحده!)

کوتاه سخن آنکه هر یک از راه حلهای ارائه شده مشکلی را حل و مشکل جدیدی را ایجاد می کردند. راه حل نهایی و پیاده شده در اینترنت که توانست به اینترنت اجازه نفس کشیدن بدهد، روش CIDR (مسیریابی براساس آدرسهای بدون کلاس) بود. ایده اصلی در CIDR که در سند RFC 1519 تشریح شده آنست که آدرسهای IP بدون در نظر گرفتن کلاس و به صورت بلوکهایی با طول متغیر تخصیص یابد. مثلاً اگر یک سایت نیاز به ۲۰۰۰ آدرس داشته باشد یک بلوک آدرس ۲۰۴۸ تایی به او داده می شود.

حذف کلاسهای آدرس، فرآیند هدایت بسته ها را پیچیده تر می کند. در سیستم مبتنی بر کلاس، فرآیند هدایت بدین نحو بود که وقتی بسته ای به یک مسیریاب می رسید، یک کپی از آدرس IP به اندازه ۲۸ بیت به راست شیفت داده می شد تا فقط چهار بیت سمت چپ آدرس (که کلاس آدرس را مشخص می کند) باقی بماند. براساس این چهار بیت (۱۶ حالت مختلف) بسته ها در یکی از کلاسهای A و B و C (و D در صورت پشتیبانی از آن) مرتب می شدند. (از این ۱۶ حالت مختلف، هشت حالت برای کلاس A است - 0xxx - چهار حالت برای کلاس - 10xx - B، دو حالت برای کلاس C - 110x - و دو حالت برای کلاسهای D و E است.) پس از تشخیص کلاس آدرس، برای بدست آوردن شماره شبکه، آدرس IP با یکی از الگوهای 8-، 16-، 24- به صورت بولی AND و بخش شماره ماشین حذف می شد. سپس شماره شبکه در هر یک از جداول مربوط به آدرسهای کلاس A، B و C

۱. پیچیدگی غیرخطی این الگوریتمها عموماً $O(n^2)$ و $O(n \log n)$ است. م

جستجو می‌شد. جداول مسیریابی برای کلاسهای A و B برحسب شماره شبکه ایندکس شده بودند.^۱ در عوض جدول مسیریابی برای کلاس C مبتنی بر روش جدول Hash (Hash Table) پیاده شده بود. پس از آنکه درایه متناظر با آدرس شبکه در یکی از این جداول پیدا می‌شد خط خروجی متناسب با آن شبکه مشخص شده و بسته بر روی آن خط هدایت می‌گردید.

در CIDR این الگوریتم ساده، کار نخواهد کرد. در عوض به هر یک از درایه‌های جدول مسیریابی یک فیلد ۳۲ بیتی جدید افزوده شده که الگوی آدرس را [از طریق یک MASK سی و دو بیتی] مشخص می‌کند. بدین ترتیب برای تمام شبکه‌ها فقط یک جدول مسیریابی یکتا وجود دارد که در حقیقت یک آرایه سه ستونی متشکل از آدرس IP، الگوی زیرشبکه (Subnet Mask) و خط خروجی است. وقتی بسته‌ای وارد می‌شود ابتدا آدرس IP آن استخراج می‌شود. سپس جدول مسیریابی درایه به درایه (Entry by Entry) جستجو و آدرس مقصد بسته پس از AND شدن با الگوی زیرشبکه از هر درایه با آدرس IP از آن درایه مقایسه می‌شود. این فرآیند آنقدر تکرار می‌گردد تا به موارد مطابقت برسد. این امکان وجود دارد که چندین درایه با یک آدرس IP مطابقت داشته باشد (به دلیل طول متفاوت الگوهای زیرشبکه). در این حالت درایه‌ای که طول الگوی زیرشبکه آن از همه بزرگتر است از بین آنها انتخاب می‌شود. به عبارتی اگر دو مورد تطابق با طول الگوی 20/ (255.255.240.0) و الگوی 24/ (255.255.255.0) پیدا شود، درایه دوم انتخاب می‌شود.

برای سرعت بخشیدن به فرآیند جستجو و مطابقت، الگوریتمهای پیچیده‌ای ابداع شده است. (Ruiz-Sanches et al. 2001) مسیریابهای تجاری در بازار امروز از تراشه‌های VLSI خاصی بهره گرفته‌اند که الگوریتم مذکور را به صورت یک «سخت‌افزار درون‌کار» (Embedded Hardware) پیاده‌سازی کرده‌اند.

برای آنکه فهم فرآیند هدایت بسته‌ها در CIDR را ساده‌تر کنیم مثالی را مدنظر قرار بدهید که در آن میلیون‌ها آدرس تعریف شده است و آدرس شروع 194.24.0.0 است. فرض کنید که دانشگاه کمبریج به ۲۰۴۸ آدرس نیاز دارد و آدرسهای 194.24.0.0 تا 194.24.7.255 به آن اختصاص داده شده است. (الگوی زیرشبکه نیز 255.255.248.0 است). بعداً دانشگاه آکسفورد تقاضای ۴۰۹۶ آدرس IP می‌دهد. از آنجایی که بلوکهای آدرس ۴۰۹۶ تایی باید در مرز ۴۰۹۶ بایستی قرار بگیرد نمی‌توان آدرسهای که از 194.24.8.0 شروع می‌شود را به آن اختصاص داد. در عوض آدرس اختصاص داده شده به او در محدوده 194.24.16.0 تا 194.24.31.255 و با الگوی 255.255.240.0 خواهد بود. در اینجا دانشگاه ادینبورو تقاضای ۱۰۲۴ آدرس داده و فضای 194.24.8.0 تا 194.24.11.255 با الگوی 255.255.252.0 به او تعلق می‌گیرد. این انتسابها در جدول ۵-۵ خلاصه شده‌اند.

الگوی نمایش	تعداد آدرس	آخرین آدرس	اولین آدرس	دانشگاه
194.24.0.0/21	2048	194.24.7.	194.24.0.0	Cambridge
194.24.8.0/22	1024	194.24.11.255	194.24.8.0	Edinburgh
194.24.12/22	1024	194.24.15.255	194.24.12.0	در دسترس و آزاد
194.24.16.0/20	4096	194.24.31.255	194.24.16.0	Oxford

شکل ۵-۵. انتساب آدرسهای IP.

۱. به عبارت ساده به دلیل کم بودن تعداد شبکه‌ها جداول مسیریابی برای کلاسهای A و B در ساختمان داده‌ای شبیه به آرایه ذخیره می‌شد. -م

حال جداول مسیریابی در تمام مسیریابهای واقع بر ستون فقرات اینترنت در جهان باید با این سه درایه جدید به‌هنگام شود. هر درایه دارای یک آدرس مبنا و یک الگوی زیرشبکه است. این درایه‌ها در مبنای دو عبارتند از:

الگوی زیرشبکه (Subnet Mask)	آدرس
C: 11000010 00011000 00000000 00000000 11111111 11111111 11111000 00000000	
E: 11000010 00011000 00001000 00000000 11111111 11111111 11111100 00000000	
O: 11000010 00011000 00010000 00000000 11111111 11111111 11110000 00000000	

حال ببینیم وقتی که بسته‌ای با آدرس 194.24.17.4 وارد یک مسیریاب می‌شود چه اتفاقی می‌افتد. این آدرس به صورت دودویی عبارت است از:

11000010 00011000 00010001 00000100

ابتدا این آدرس با الگوی زیرشبکه کمبریج، AND می‌شود و نتیجه زیر بدست می‌آید:

11000010 00011000 00010000 00000000

این مقدار با آدرس مبنای دانشگاه کمبریج مطابقت ندارد. حال مجدداً آدرس اصلی با الگوی زیرشبکه دانشگاه ادینبورو AND شده و نتیجه زیر بدست می‌آید:

11000010 00011000 00010000 00000000

این مقدار نیز با آدرس مبنای دانشگاه ادینبورو تطابق ندارد و همین کار برای دانشگاه آکسفورد تکرار شده مقدار زیر بدست می‌آید:

11000010 00011000 00010000 00000000

این مقدار با آدرس مبنای دانشگاه آکسفورد مطابقت دارد. اگر هیچ مورد تطبیق دیگری در جدول یافت نشد بسته بر روی خطی ارسال می‌شود که در درایه متناظر با شبکه دانشگاه آکسفورد درج شده است.

حال اجازه بدهید، آدرس این سه دانشگاه را از دید یک مسیریاب در نبراسکای اوهاما بررسی کنیم. این مسیریاب چهار خط به مینیاپولیس، نیویورک، دالاس و دنور دارد. وقتی نرم‌افزار مسیریاب اوهاما، این سه درایه جدید را جهت درج در جدول مسیریابی خود دریافت می‌دارد، متوجه می‌شود که قادر است هر سه تای آنها را در یک «درایه واحد و تجمیع شده» (Aggregate Entry) به صورت 194.24.0.0/19 ادغام نماید.^۱ آدرس والگوی زیرشبکه در مبنای دو به صورت زیر است:

11000010 00000000 00000000 00000000 11111111 11111111 11100000 00000000

طبق این درایه تمام بسته‌هایی که به مقصد یکی از این سه دانشگاه روانه شده‌اند به سوی نیویورک هدایت می‌شوند. با تجمیع این سه درایه، مسیریاب اوهاما توانسته به میزان دو درایه حجم جدول خود را کاهش بدهد. به همین ترتیب اگر مسیریاب نیویورک برای تمام ترافیک منتهی به انگلستان فقط یک خط به لندن داشته باشد او نیز سه درایه فوق را در یک درایه ادغام می‌کند ولیکن اگر برای لندن و ادینبورو دو خط مجزا داشته باشد باید هر سه تای آنها را بطور مجزا در جدول خود ذخیره کند. عمل تجمیع (Aggregation) در اینترنت به طور گسترده‌ای مورد استفاده قرار گرفته تا حجم جداول مسیریابی کاهش یابد.

آخرین نکته در این مثال آن است که بر طبق درایه ادغام شده در جدول مسیریابی مسیریاب واقع در اوهاما

۱. از آن جهت امکان تجمیع این سه آدرس وجود داشته که بسته‌هایی که مقصدشان هریک از این سه دانشگاه است باید بر روی خط خروجی یکسانی بروند. سم

حتی بسته‌هایی که به آدرس اختصاص داده نشده روانه هستند [یعنی آدرسهای بین 194.24.12.0 تا 194.24.15.255] نیز به سوی نیویورک هدایت می‌شوند. مادامی که این آدرسها به کسی اختصاص داده نشده هیچ مشکلی به وجود نمی‌آید چرا که بنا نیست بسته‌هایی با این آدرسها تولید شوند. ولی اگر این بلوک آدرس، به شرکتی در کالیفرنیا داده شود باید درایه‌ای جدید به شکل 194.24.12.0/22 در جداول مسیریابی تمام مسیریابها درج شود تا بسته‌هایی به مقصد این شبکه نیز بدرستی مسیریابی شوند.

NAT: ترجمه آدرسهای شبکه

آدرسهای IP کمیاب و ارزشمند هستند: یک ISP ممکن است یک بلوک آدرس با الگوی 16/ (همان کلاس B سابق) و توانایی آدرس دهی ۶۵۵۳۴ ماشین میزبان، داشته باشد. اگر تعداد مشتریان این ISP از این تعداد بیشتر شود مشکل بهم می‌زند. برای مشتریان خانگی که از طریق خطوط تلفن متصل می‌شوند، راه حل این مشکل آن است که وقتی مشتری شماره‌گیری کرد و وارد شد به او موقتاً یک آدرس IP پویا اختصاص داده شود و پس از پایان نشست و قطع ارتباط، این آدرس پس گرفته شود. در این روش شبکه‌ای با آدرس 16/ (کلاس B) می‌تواند حداکثر ۶۵۵۳۴ کاربر فعال داشته باشد که این تعداد حتی برای یک ISP با چند صد هزار مشتری نیز کفایت می‌کند. به محض آنکه یک نشست خاتمه یافت آدرس IP متناسب شده قبلی، به تماس گیرنده بعدی داده می‌شود. این استراتژی اگرچه برای یک ISP با تعداد متوسطی از کاربران خانگی به خوبی کار می‌کند ولی برای ISPهایی که به مشتریان اداری خدمات می‌دهند مفید نیست.

مشکل از اینجا ناشی می‌شود که مشتریان اداری انتظار دارند که حداقل در ساعات کاری روز خطی دائم و فعال (On-Line) داشته باشند. امروزه، چه دفاتر کوچک اداری مثل یک آژانس مسافرتی با سه کارمند و چه شرکت‌های بزرگ که دارای تعداد زیادی کامپیوتر و شبکه محلی هستند، نیاز به خط دائم و فعال دارند. برخی از این کامپیوترها، PC کارمندان و برخی دیگر مثلاً سرورس دهنده‌های وب هستند. عموماً در هر LAN یک مسیریاب وجود دارد که از طریق یک خط اجاره‌ای (Leased) به ISP متصل شده است. چنین ساختاری متضمن آن است هر کامپیوتر آدرس IP خود را داشته باشد و به طور روزانه تغییر نکند. در نتیجه، تعداد کل کامپیوترهایی که در اختیار مشتریان اداری است نباید از تعداد آدرسهای IP متعلق به ISP بیشتر شود. برای آدرس 16/، حداکثر تعداد کامپیوترها ۶۵۵۳۴ است. برای یک ISP با دهها هزار مشتری اداری، این فضا سریعاً اشباع می‌شود.

آنچه که مشکل را حادتر می‌کند آن است که روز به روز بر تعداد مشترکین اینترنت از طریق مودمهای کابلی ADSL افزوده می‌شود. ویژگی چنین سرویسی عبارت است از: (۱) کاربر یک آدرس IP دائم و ثابت می‌گیرد. (۲) هزینه شماره‌گیری و اتصال ندارد (مگر یک هزینه ثابت ماهانه) بدین ترتیب اینگونه کاربران همیشه در شبکه حضور دارند. این موضوع، مشکل کمبود آدرسهای IP را افزایش می‌دهد. در اینجا تخصیص موقت آدرسهای IP (شبیه به مکانیزمی که برای کاربران تلفنی داشتیم) عملی نیست.

حتی از این هم پیچیده‌تر آنکه ممکن است کاربران ADSL و اینترنت کابلی دارای دو یا چند کامپیوتر در خانه باشند و تمام اعضای خانواده از طریق همین خط فعال و مشترک به ISP متصل شوند. یک راه حل آن است که تمام PCها از طریق یک LAN به هم متصل شده و با یک مسیریاب به ISP وصل شوند. از دیدگاه ISP شبکه این خانواده فرقی با یک دفتر اداری کوچک ندارد.

مشکل کمبود آدرسهای IP یک مسئله تئوریک نیست که در آینده‌ای دور رخ بدهد. همین الان با این مشکل مواجه هستیم. راه حل طولانی مدت و همیشگی این مشکل آنست که به سوی IPv6 (که آدرسهای آن ۱۲۸ بیتی

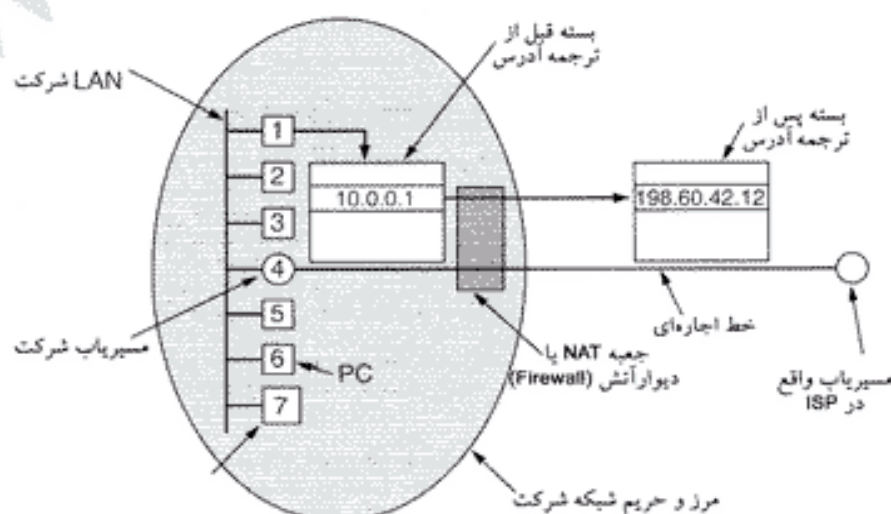
است) حرکت نماییم ولیکن گذار از نسخه ۴ به نسخه ۶ به آهستگی صورت می‌گیرد و سالها طول می‌کشد تا این تغییر به طور کامل انجام شود. در نتیجه بسیاری افراد احساس کردند که به یک راه حل سریع و کوتاه مدت نیاز است. این راه حل سریع، NAT (ترجمه آدرس شبکه) است که در RFC 3022 تشریح شده و در ادامه آنرا مختصراً بررسی می‌نماییم. برای کسب آگاهی بیشتر به مرجع (Dutcher, 2001) مراجعه نمایید.

ایده اصلی در NAT آن است که به هر شرکت یک یا تعداد کمی آدرس IP معتبر و جهانی اختصاص بدهیم. درون این شرکت، هر کامپیوتر دارای یک آدرس IP یکتا است که برای مسیریابی ترافیک داخلی بکار می‌آید. با این حال وقتی بسته‌ای بخواهد شرکت را ترک کرده و به ISP برود باید قبل از خروج ترجمه آدرس صورت بگیرد. برای آنکه این روش ممکن باشد سه محدوده از فضای آدرس IP جهت بکارگیری در شبکه‌های داخلی، به صورت «خصوصی» (Private) تعریف شده است و شرکتها می‌توانند به صورت دلخواه از آنها استفاده کنند. [نیازی به ثبت جهانی آنها نیست.] تنها قانون آن است که هیچ بسته‌ای نباید با چنین آدرسی بر روی اینترنت ظاهر شود. این سه محدوده رزرو شده عبارتند از:

10.0.0.0	- 10.255.255.255/8	(16,777,216 Hosts)
172.16.0.0	- 172.31.255.255/12	(1,048,576 Hosts)
192.168.0.0	- 192.168.255.255/16	(65535 Hosts)

در محدوده اول ۱۶۷۷۷۲۱۶ آدرس (به استثنای ۰ و ۱-) در دسترس است و عموماً اکثر شرکتها از آن استفاده می‌کنند هر چند نیازی به چنین تعداد آدرسی نداشته باشند.

عملکرد NAT در شکل ۵-۶ نشان داده شده است. ماشینها در درون شبکه دارای یک آدرس یکتا به فرم 10.x.y.z هستند. ولیکن وقتی بسته‌ای بخواهد مرز شرکت را ترک کند ابتدا باید از درون یک «جعبه NAT» (NAT BOX) عبور کرده و آدرس مبدا آن با آدرس IP حقیقی شرکت جانشین شود. مثلاً در شکل ۵-۶ آدرس 10.0.0.1 با آدرس 198.60.42.1 عوض شده است. اغلب «جعبه NAT» در یک «دیوار آتش» (Firewall) ادغام می‌شود تا این ابزار ضمن ترجمه آدرس، امنیت شبکه را نیز با نظارت دقیق بر ورود و خروج اطلاعات تضمین نماید. در فصل ۸ مفهوم «دیوار آتش» را بررسی خواهیم کرد. همچنین می‌توان «جعبه NAT» را در مسیریاب شرکت قرار داد. اکثر مسیریابهای امروزی از فرآیند NAT پشتیبانی می‌کنند.



شکل ۵-۶. مکان و عملکرد «جعبه NAT»

تا اینجا جزئیات کمی از فرآیند NAT مطرح کرده ایم؛ وقتی پاسخ یک بسته بر می گردد (مثلاً از سرویس دهنده وب) طبعاً آدرس ماشین گیرنده پاسخ، 192.60.42.1 است. سؤال این است که جعبه NAT از کجا بداند که آدرس کدام ماشین داخلی را به جای آن قرار بدهد؟ مسئله اصلی در NAT همین نکته است. اگر فیلدی اضافی در سرآیند بسته IP وجود داشت می شد از آن برای درج آدرس واقعی گیرنده بسته بهره گرفت ولیکن در سرآیند بسته تنها یک بیت بلااستفاده مانده است. همچنین می توان یک گزینه جدید [در فضای فیلد اختیاری Option] تعریف کرد تا آدرس حقیقی ماشین مبدا بسته را نگاه دارد ولی انجام این کار مستلزم آن است که کُد نرم افزار IP در تمام ماشینها و در کل اینترنت تغییر کند تا گزینه جدید به رسمیت شناخته شده و به درستی تعبیر شود. این راه حل نیز فرآیند زمان بری است و مشکل را در کوتاه مدت حل نخواهد کرد.

آنچه که بطور واقعی اتفاق می افتد به نحو ذیل است: طراحان NAT بدین نتیجه رسیده بودند که اغلب بسته های IP در درون فیلد داده خود یک بسته TCP یا UDP حمل می کنند. هرگاه در فصل ششم TCP و UDP را بررسی کردیم، خواهید دید که هر دوی این پروتکلها دارای سرآیندی برای بسته های خود هستند که دو فیلد «شماره پورت مبدا» و «شماره پورت مقصد» جزو آنهاست. در زیر اگرچه تمرکز ما بر پورت های TCP است ولی همین قضیه برای پورت های UDP نیز صادق است. این پورتها که اعداد صحیح ۱۶ بیتی هستند، مشخص می کنند که اتصال TCP از چه پروسه ای شروع و به چه پروسه ای ختم می شود. این شماره پورتها فیلدهای لازم برای عملکرد NAT را فراهم آورده اند.

هرگاه یک پروسه بخواهد یک اتصال TCP با یک پروسه راه دور برقرار کند یک شماره پورت بلااستفاده برای خود بر میگزیند. این پورت، اصطلاحاً «پورت مبدا» نام دارد و به کد برنامه TCP تفهیم می کند که باید بسته های ورودی با این شماره پورت را برای او بفرستد. همچنین هر پروسه یک شماره پورت مقصد تعیین می کند تا مشخص شود که بسته ها باید به کدام پروسه در ماشین مقصد تحویل شود. شماره پورت های صفر تا ۱۰۲۳ برای سرویس دهنده های مشهور رزرو شده است. به عنوان مثال پورت شماره ۸۰ توسط سرویس دهنده های وب بکار گرفته شده است، لذا برنامه های مشتری (Client) براحتی با آنها ایجاد ارتباط می کنند. کوتاه سخن آنکه، هر پیام خروجی از TCP دارای شماره پورت مبدا و شماره پورت مقصد است و این دو شماره پورت هویت پروسه های طرفین ارتباط را مشخص می نمایند.

تمثیلی از یک نمونه می تواند به فهم شماره های پورت کمک کند: یک شرکت را در نظر بگیرید که دارای یک شماره تلفن اصلی و واحد است. وقتی افراد با این شماره تماس می گیرند بلافاصله اپراتور مربوطه از آنها سؤال می کند که کدام شماره داخلی مدنظر آنهاست؛ سپس خط داخلی را وصل می کند. شماره اصلی به مثابه آدرس IP است و شماره های داخلی، مشابه با شماره پورت هستند. پورتها، ۱۶ بیت آدرس اضافی دیگر هستند که هویت پروسه گیرنده بسته ها را مشخص می نمایند.

با استفاده از فیلد «شماره پورت مبدا» می توان مشکل نگاشت آدرسها در NAT را حل کرد. هرگاه بسته ای برای خروج از شبکه به NAT وارد شود، آدرس مبدا آن که به شکل 10.x.y.z است با آدرس IP حقیقی و معتبر شرکت عوض می شود. مضاف بر این فیلد شماره پورت مبدا (TCP Source Port) با عددی عوض می شود که در حقیقت این عدد اندیس جدول ترجمه آدرس در جعبه NAT است. هر یک از درایه های این جدول، آدرس IP اصلی و همچنین شماره پورت واقعی آن بسته را نگه می دارند. در آخر کد کشف خطای بسته TCP و بسته IP از نو محاسبه و در بسته قرار داده می شود. [چرا که هم فیلد شماره پورت و هم فیلد آدرس IP مبدا در جعبه NAT عوض می شود. م] عوض کردن مقدار فیلد پورت مبدا (Source Port) الزامی است چراکه ممکن است بطور همزمان از دو ماشین به آدرسهای 10.0.0.1 و 10.0.0.2 یک اتصال TCP اتفاقاً با شماره پورت مبدا یکسان (مثلاً

۵۰۰۰) ایجاد شود، لذا شماره پورت مبدا نمی تواند هویت واقعی پروسه ارسال کننده بسته ها را مشخص کند.^۱ وقتی بسته ای از طریق ISP به جعبه NAT وارد می شود ابتدا مقدار فیلد پورت مبدا استخراج شده و از آن به عنوان اندیس جدول نگاشت در جعبه NAT استفاده می شود. پس از پیدا شدن درایه متناظر، آدرس IP داخلی و شماره اصلی پورت مبدا بسته استخراج شده و در درون بسته قرار می گیرد. سپس این بسته از طریق مسیر یاب داخلی شرکت، برای تحویل به آدرس 10.x.y.z مسیر طبیعی خود را طی می کند.

همچنین از NAT می توان برای تخفیف مشکل کمبود آدرس IP برای کاربران ADSL و کاربران کابلی بهره گرفت. هر گاه ISP بخواهد به هر یک از این کاربران آدرس انتساب بدهد، از آدرسی در فضای 10.x.y.z بهره می گیرد. قبل از آنکه بسته های ماشین کاربران، ISP را ترک کنند و به اینترنت وارد شوند باید ابتدا وارد جعبه NAT شده و آدرس غیرحقیقی و محلی آنها به آدرس واقعی متعلق به ISP نگاشته شود. در مسیر برگشت، عکس فرآیند نگاشت انجام می شود. بدین نحو از دیدگاه اینترنت، این ISP (و کاربران ADSL یا کابلی آن) دقیقاً مثل یک شرکت بزرگ به نظر می رسند، هر چند تعداد آدرسهای واقعی و معتبر ISP ناچیز است.

اگرچه این روش مشکل کمبود آدرسهای IP را حل می کند ولیکن بسیاری از افراد در جامعه اینترنت از آن به عنوان کاری بی ارزش و مردود یاد می کنند. برخی از مخالفت های آنان را به اختصار ارائه می نماییم. اول آنکه NAT مدل معماری IP را نقض می کند چرا که در این مدل بیان شده که آدرس IP به صورت یکتا ماشینی واحد را در کل جهان مشخص می نماید. ساختار تمام نرم افزارهای اینترنت با تکیه بر این واقعیت بنیان گذاشته شده است. با NAT ممکن است هزاران ماشین از آدرس 10.0.0.1 استفاده کنند (و می کنند).

دوم آنکه NAT اینترنت را از حالت «بدون اتصال» به شبکه ای «اتصال گرا» تبدیل می نماید. مسئله اینجاست که جعبه NAT باید اطلاعاتی را در خصوص نگاشت اتصالهایی که از آن می گذرند در خود نگاه دارد. نگهداری وضعیت هر اتصال ویژگی شبکه های اتصال گراست و سختی با شبکه های بدون اتصال ندارد. اگر جعبه NAT به ناگاه از کار بیفتد و جدول نگاشت آن از دست برود تمام اتصالات TCP برقرار شده از دست می رود. بدین ترتیب با وجود NAT، اینترنت به یک شبکه آسیب پذیر مدار مجازی تبدیل می شود.

سوم آنکه، NAT اصول بنیانی لایه بندی پروتکلها را نقض می کند: لایه k نباید هیچ تصویری از آنچه که لایه k+1 در فیلد حمل داده از بسته او قرار می دهد، داشته باشد یا در آن دخالتی کند.^۲ اصل اساسی در معماری لایه به لایه آنست که تمام لایه ها مستقل از دیگری باشند. اگر مثلاً زمانی TCP به نسخه TCP-2 ارتقاء یابد و سرآیند بسته ها تغییر کنند (مثلاً شماره پورتها ۳۲ بیتی شوند)، NAT از کار خواهد افتاد. ایده اصلی در پروتکل های لایه ای آن بوده که تغییر در یک لایه نیازی به تغییر در لایه های دیگر نداشته باشد در حالیکه NAT این عدم وابستگی را از بین می برد.

چهارم آنکه پروسه های اینترنت مجبور به استفاده از TCP یا UDP نیستند. اگر فرضاً کاربری بر روی ماشین A تصمیم بگیرد برای محاوره و مبادله داده با کاربری بر روی ماشین B از یک پروتکل جدید در لایه انتقال استفاده کند (مثلاً برای کاربردهای چند رسانه ای)، وجود NAT منجر به عدم کارکرد آن برنامه کاربردی خواهد شد چرا که NAT نخواهد توانست فیلد پورت مبدا را به درستی پیدا کرده و از آن استفاده نماید.

پنجم آنکه برخی از برنامه های کاربردی آدرس IP ماشین خود را در متن اطلاعات ارسالی قرار می دهند. گیرنده نیز این آدرس را استخراج کرده و از آن در جایی استفاده می کند. از آنجایی که NAT چیزی در مورد این

۱. عبارت روشتر چون تمام بسته ها در برگشت آدرس IP یکسانی دارند فلذا این آدرس پورت هر بسته است که هویت گیرنده واقعی بسته را مشخص می کند و طبعاً NAT باید در هنگام خروج بسته ها ضمن عوض کردن شماره پورت، یکتا بودن آنها تضمین کند. - م
۲. عبارتی از دیدگاه لایه k آنچه که از لایه k+1 می رسد تعدادی بابت خام است. - م

آدرسهای مخفی نمی‌داند فلذا قادر به تغییر آنها نبوده و هر گونه تلاش برای استفاده از این آدرسهای ناصحیح (در ماشین راه دور) با شکست مواجه می‌شود. FTP، یعنی استاندارد انتقال فایل در اینترنت به همین ترتیب عمل می‌کند و با وجود NAT از کار می‌افتد مگر آنکه اقدامات احتیاطی خاصی به عمل آید. به دلیل مشابه، پروتکل H.323 که برای تلفن اینترنتی کاربرد دارد (و در فصل هفتم به معرفی آن خواهیم پرداخت) با وجود NAT کار نخواهد کرد. البته می‌توان با تغییرات اصلاحی در NAT آن را بکار گرفت ولی این که با معرفی هر برنامه کاربردی جدید مجبور به اصلاح NAT شویم، اصلاً ایده جالبی نیست.

ششم آنکه چون فیلد آدرس پورت مبدا، ۱۶ بیتی است، حداکثر ۶۵۵۳۶ ماشین را می‌توان به یک آدرس IP واحد نگاشت. این تعداد حقیقتاً مقدار کمی است (گذشته از آن، ۴۰۹۶ شماره پورت نیز برای کاربردهای خاص کنار گذاشته شده‌اند)، ولیکن اگر تعداد آدرسهای IP معتبر و جهانی که در اختیار ISP قرار دارد، بیش از یکی باشد به ازای هر یک می‌توان ۶۱۴۴۰ ماشین را با آدرسهای غیرحقیقی مدیریت کرد.

این مشکلات به همراه مسائل دیگر NAT، در RFC 2993 تشریح شده است. عموماً مخالفین NAT می‌گویند که با حل نازیا و موقتی مسئله کمبود آدرسهای IP، اصرار بر روی راه حل واقعی و نهایی که همانا رفتن به طرف IPv6 است، کم می‌شود و تعویق انداختن در این تغییر و تحول اصلاً خوب نیست!

۳-۶-۵ پروتکل‌های کنترل اینترنت

اینترنت مضاف بر IP که برای انتقال داده‌ها کاربرد دارد، چندین پروتکل کنترلی دیگر دارد که همگی در لایه شبکه به کار گرفته می‌شوند. این پروتکلها عبارتند از: ICMP، ARP، RARP، BOOTP و DHCP. در این بخش، این پروتکلها را به ترتیب مرور می‌کنیم.

ICMP: پروتکل ارسال پیامهای کنترلی در اینترنت

عملکرد اینترنت توسط مسیریابها به صورت دقیق نظارت و کنترل می‌شود. هر گاه اتفاق غیرمنتظره‌ای بیفتد، رخداد مزبور توسط پروتکل ICMP گزارش می‌شود. این پروتکل همچنین برای آزمایش و رفع عیب شبکه کاربرد دارد. تقریباً یک دوجین از پیامهای ICMP به منظور اطلاع‌رسانی تعریف شده است. مهمترین این پیامها در شکل ۵-۶۱ فهرست شده‌اند. هر پیام ICMP مستقیماً درون یک بسته IP جاسازی و ارسال می‌شود.

نوع پیام	توصیف عملکرد
Destination unreachable	بهر دلیلی بسته را نمی‌توان به مقصد تحویل داد.
Time exceeded	زمان حیات بسته به صفر رسیده است.
Parameter problem	فیلدی از سرآیند بسته مقدار نامعتبر داشته است.
Source quench	بسته دعوت به آرامش
Redirect	حاروی اطلاعاتی در خصوص جغرافیای مسیر و اعلام اشتباه در مسیریابی
Echo	درخواست از یک ماشین تا اگر فعال است پاسخ بدهد.
Echo reply	پاسخ به پیام Echo به منظور تایید فعالیت
Timestamp request	همانند پیام Echo به همراه مهر زمان
Timestamp reply	همانند پیام Echo Reply به همراه مهر زمان

شکل ۵-۶۱. انواع پیامهای اساسی ICMP.

از پیام DESTINATION UNREACHABLE زمانی استفاده می شود که زیر شبکه یا مسیریاب نتواند موقعیت مقصد را تشخیص بدهد یا وقتی که بیت DF^۱ در بسته ای به ۱ تنظیم شده باشد و آن بسته مجبور به عبور از شبکه ای باشد که طول بسته های آن کوچک است. [طبعاً بسته حذف می شود].

پیام TIME EXCEEDED زمانی ارسال می شود که بسته به دلیل صفر شدن شمارنده TTL (شمارنده زمان حیات) حذف گردد. این رخداد نشانه آن است که بسته ها در حلقه بی نهایت افتاده اند یا آنکه ازدحام سنگینی رخ داده یا آنکه مقدار اولیه این شمارنده بسیار کم بوده است.

پیام PARAMETER PROBLEM مشخص کننده آنست که در سرآیند بسته IP مقدار نامعتبر و اشتباهی درج شده است. [لذا مسیریاب آن را حذف کرده و این پیام را ارسال نموده است.] این مشکل از وجود یک اشکال در نرم افزار IP ماشین فرستنده یا احتمالاً نرم افزار مسیریابهای میانی پرده بر می دارد.

پیام SOURCE QUENCH سابقاً برای آن بکار می رفته تا به ماشینهایی که بیش از اندازه بسته ارسال می کنند اخطار داده شود. هر ماشین که این پیام را دریافت کند باید نرخ ارسال بسته های خود را کاهش بدهد. ولیکن از این پیام به ندرت استفاده می شود چرا که وقتی ازدحام رخ بدهد، ارسال چنین پیامهایی بر هیزم این آتش می افزاید. کنترل ازدحام در اینترنت عمدتاً در لایه انتقال انجام می شود. این روش را در فصل ششم خواهیم آموخت. از پیام REDIRECT زمانی استفاده می شود که مسیریاب احساس کند بسته ای در مسیر غلط قرار گرفته است. این پیام برای گزارش خطای احتمالی به ماشین مبدا بسته، کاربرد دارد.

پیامهای ECHO REPLY و ECHO بدین منظور کاربرد دارند که ببینیم آیا یک ماشین مقصد در دسترس و فعال است یا خیر. انتظار می رود هر ماشینی که پیام ECHO دریافت می کند، در پاسخ به آن، پیام ECHO REPLY را برگرداند.

پیامهای TIMESTAMP REQUEST و TIMESTAMP REPLY مشابه با دو پیام قبلی هستند با این تفاوت که در بدنه پاسخ، زمان دریافت پیام و همچنین زمان ارسال پیام درج می شود. از این قابلیت می توان برای اندازه گیری کارایی شبکه بهره گرفت.

مضاف بر پیامهای فوق الذکر، پیامهای دیگری نیز تعریف شده اند که می توانید فهرست آنها را در آدرس زیر بیابید: www.iana.org/assignments/icmp-parameters

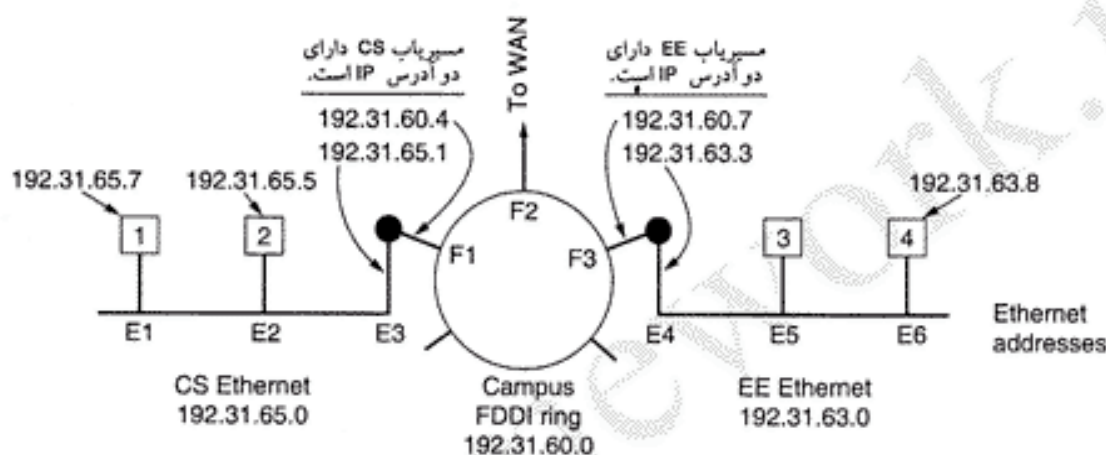
۲. ARP: پروتکل تحلیل آدرس

اگرچه هر ماشین در اینترنت دارای یک (یا چند) آدرس IP است، ولیکن این آدرسها حقیقتاً نمی توانند برای ارسال بسته ها بکار گرفته شوند چرا که سخت افزار لایه پیوند داده ها هیچ درکی از آدرسهای IP ندارد. امروزه اغلب ماشینهای میزبان در شرکتها یا دانشگاهها به کمک یک کارت واسط شبکه به LAN متصل شده اند و این کارتها فقط آدرسهای MAC (آدرس سخت افزاری کارت شبکه) را می شناسند. به عنوان مثال هر کارت شبکه اترنت پس از تولید در کارخانه، دارای یک آدرس ۴۸ بیتی یکتا است. سازندگان کارتهای اترنت برای تعبیه آدرس در کارتهای تولیدی خود ابتدا از یک مرکز مجاز تقاضا می کنند که یک بلوک آدرس به آنها اختصاص بدهد و بدین ترتیب اطمینان حاصل می شود که هیچ دو کارت شبکه در کل جهان دارای آدرس یکسان نیست. (تا از هر گونه برخورد دو کارت شبکه با شماره های یکسان بر روی LAN اجتناب شود). کارتهای شبکه، فریمها را براساس آدرسهای ۴۸ بیتی اترنت، ارسال یا دریافت می نمایند و هیچ چیزی در خصوص آدرسهای IP نمی دانند. اکنون این سؤال پیش می آید که چگونه آدرسهای IP به آدرسهای لایه پیوند داده (مثل آدرسهای اترنت)

۱. Don't Fragment

۲. Address Resolution Protocol

نگاشته و تبدیل می‌شوند؟ برای تشریح عملیات نگاشت آدرس از مثال شکل ۵-۶۲ استفاده کرده‌ایم. در این شکل یک دانشگاه کوچک با شبکه‌ای با چند کلاس C (به عبارتی شبکه 24/2) به تصویر کشیده شده است. در این شکل دو شبکه اترنت وجود دارد که یکی از آنها با آدرس 192.31.65.0 به دانشکده کامپیوتر و دیگری با آدرس 192.31.63.0 به دانشکده برق تعلق دارد. این دو شبکه از طریق یک شبکه حلقه (مثل FDDI) با آدرس 192.31.60.0 که نقش ستون فقرات شبکه کل دانشگاه را ایفاء می‌کند، به یکدیگر متصل شده‌اند. هر ماشین به اترنت دارای یک آدرس MAC پکتا است که در شکل با برجسبهای E1 تا E6 مشخص شده‌اند. هر ماشین متصل به شبکه حلقه نیز آدرس FDDI خود را دارد که آنها نیز با F1 تا F3 نشان داده شده‌اند.



شکل ۵-۶۲. سه شبکه بهم متصل با الگوی 24/2: دو شبکه اترنت و یک شبکه حلقه از نوع FDDI.

حال ابتدا بررسی کنیم که کاربری بر روی ماشین میزبان ۱ چگونه بسته‌ای را برای کاربری بر روی ماشین ۲ می‌فرستد. فرض را بر آن گذاشته‌ایم که فرستنده، نام گیرنده مورد نظر را می‌داند و مثلاً این نام mary@eagle.cs.uni.edu است. در اولین گام باید آدرس IP ماشین eagle.cs.uni.edu را بدست بیاورد. این جستجو از طریق «سیستم نام دامنه» (DNS) که در فصل هفتم آن را مطالعه خواهیم کرد، انجام می‌گیرد. در اینجا فرض می‌کنیم که DNS، آدرس IP ماشین ۲ را 192.31.65.5 برگردانده است.

در اینجا نرم‌افزار لایه بالاتر در ماشین ۱ بسته‌ای می‌سازد و درون فیلد «آدرس مقصد» آن مقدار 192.31.65.5 را درج می‌کند و آن را جهت ارسال تحویل نرم‌افزار IP می‌دهد. نرم‌افزار IP با بررسی آدرس بدین نتیجه می‌رسد که ماشین مقصد، در شبکه خودش قرار گرفته است^۱ ولی محتاج روشی است تا بتواند آدرس اترنت ماشین مقصد را پیدا کند. یک راه حل آنست که یک فایل پیکربندی در جایی از سیستم ذخیره شده باشد و از طریق آن آدرس IP به آدرسهای اترنت نگاشته شود. اگرچه این روش عملی است ولی برای موسساتی که هزاران ماشین دارند به روز نگه داشتن چنین فایل‌هایی، کاری بسیار وقتگیر و منشاء خطاهای سهوی است.

راه حل بهتر آنست که ماشین میزبان ۱ بسته‌ای را به صورت فراگیر (Broadcast) بر روی اترنت پخش کند و از همه سؤال کند که: «آدرس 192.31.65.5 متعلق به کیست؟» این اعلام همگانی به یکایک ماشینهای شبکه اترنت با آدرس 192.31.65.0 خواهد رسید. تنها ماشین ۲ به این سؤال پاسخ خواهد داد و آدرس اترنت خود یعنی E2 را اعلام خواهد کرد. بدین طریق ماشین ۱ متوجه می‌شود که آدرس 192.31.65.5 متعلق به ماشینی است که آدرس

۱. نرم‌افزار IP از آنجا متوجه می‌شود که ماشین مقصد در شبکه محلی خودش واقع است که طبق الگوی 24/2 بخش شماره شبکه خودش و شماره شبکه مقصد یکسان است. (شماره شبکه هر دو 192.31.65.0 است.) -م

اترنت آن E2 است. پروتکلی که برای این پرسش و پاسخ بکار می رود **ARP** نام دارد و تقریباً تمام ماشینهای اینترنت آنرا اجرا کرده اند. ARP در سند RFC 826 تبیین شده است.

مزیت استفاده از ARP به جای ذخیره فایل های پیکربندی، سادگی آنست. مدیر سیستم فقط باید برای هر ماشین، آدرس IP و الگوی زیر شبکه (Subnet Mask) آنرا مشخص کند. مابقی کارها را ARP انجام می دهد.

پس از بدست آمدن آدرس E2، نرم افزار IP در ماشین میزبان ۱، یک فریم اترنت به آدرس E2 ساخته و بسته IP (با آدرس 192.31.65.5) را در درون فیلد داده آن قرار داده و آن را بر روی اترنت روانه می کند. کارت شبکه ماشین ۲ این فریم را گرفته و تشخیص می دهد که متعلق به خود اوست؛ آنرا پذیرفته و یک «وقفه»^۱ تولید می کند. نرم افزار راه انداز اترنت، بسته IP را از درون فریم استخراج کرده و آنرا به نرم افزار IP تحویل می دهد. نرم افزار IP متوجه می شود که این بسته به آدرس صحیحی آمده و طبقاً آن را پردازش می نماید.

برای آنکه ARP کارآمدتر عمل کند می توان بهینه سازی هایی انجام داد. اولین بهینه سازی آنست که هر گاه ARP اجرا شد و آدرسی را بدست آورد، آن را در «حافظه نهان»^۲ خود ذخیره نماید تا در دفعات بعدی بتواند با استفاده از این آدرس، سریعاً با ماشین مربوطه ارتباط برقرار کند. دفعه بعد، ARP نگاشت آدرس IP به آدرس اترنت را در حافظه نهان خود خواهد یافت و نیاز مجدد به سؤال همگانی نیست. در بسیاری از حالات، ماشین ۲ (پس از دریافت بسته) باید پاسخی پس بفرستد و در نتیجه او نیز مجبور است برای یافتن آدرس اترنت فرستنده تقاضا، ARP را اجرا نماید. برای آنکه در اینجا از ارسال فراگیر بسته های ARP جلوگیری شود می توان بدین نحو عمل کرد که وقتی مثلاً ماشین ۱ برای یافتن آدرس ماشین ۲ به صورت پخش فراگیر از همه سؤال می کند، «نگاشت آدرس IP به آدرس اترنت» خود را نیز اعلام نماید. وقتی بسته های پخش ARP به ماشین ۲ می رسد جفت آدرس (E1 و 192.31.65.7) وارد حافظه نهان ARP می شود تا برای استفاده های آتی نیاز به سؤال نباشد. در حقیقت تمام ماشینهای متصل به اترنت می توانند این جفت آدرس نگاشته شده را در حافظه نهان ARP خود ذخیره نمایند.^۳

روشی دیگر برای بهینه سازی ARP آنست که هر ماشین به محض راه اندازی (بوت) «نگاشت آدرس سخت افزاری» به آدرس IP خود را به صورت پخش فراگیر به همه اعلام نماید.^۴ عموماً این کار بدین نحو انجام می گیرد که ماشین در حال بوت، آدرس سخت افزاری معادل با IP خودش را سؤال می کند! طبعاً چنین سؤالی هیچ پاسخی ندارد ولی نتیجه جانبی آن این است که در اثر پخش این سؤال (که پاسخ خود را به همراه دارد) بر روی کل شبکه محلی، نگاشت آدرس او به صورت یک درایه (Entry) درون حافظه نهان ARP ماشینهای فعال درج می شود. اگر پاسخ چنین سؤالی به صورت غیر منتظره دریافت گردد، مشخص می شود که دو ماشین دارای آدرس IP مشابه هستند. ماشین دوم باید این موضوع را به مدیر سیستم اطلاع داده و بوت نشود.

برای آنکه جدول نگاشت آدرس در ARP بتواند تغییر کند، درایه های موجود در حافظه نهان ARP باید پس از چند دقیقه اعتبار خود را از دست بدهند و با سؤال مجدد، نگاشت آدرسها از نو انجام شود چرا که مثلاً اگر یک کارت شبکه اترنت، خراب و با کارتی جدید عوض شود آدرس اترنت معادل با آدرس IP آن ماشین عوض می شود و اگر بقیه ماشینها بخواهند از جدول نگاشت قدیمی خود استفاده کنند این ماشین از دسترس دیگران

۱. Interrupt

۲. ARP Cache

۳. به عبارت دیگر وقتی یک ماشین، آدرس سخت افزاری یک ماشین دیگر را سؤال می کند، جفت آدرس سخت افزاری و IP خود را نیز به همه اعلام می کند و از آن به بعد ماشینها نیازی به سؤال کردن ندارند. بدین نحو حافظه نهان ARP سریعاً پر شده و فرایند پرسش و پاسخ بهبود یافته و اضافی انجام نخواهد شد. -م

۴. واژه های آدرس MAC، آدرس فیزیکی، آدرس سخت افزاری، آدرس اترنت همگی معادلند و به آدرس تعبیه شده بر روی کارت شبکه (آدرس لایه پیوند داده ها) اشاره دارند. -م

خارج می شود.

حال مجدداً به شکل ۵-۶۲ نگاهی بیندازید: هنگامی که ماشین ۱ بخواهد بسته ای برای ماشین ۴ (با آدرس 192.31.63.8) بفرستد با شکست مواجه می شود زیرا ماشین ۴ نمی تواند پخش فراگیر بسته های پرسش را دریافت کند (مسیریابها بسته های پخش فراگیر را به شبکه های دیگر هدایت نمی کنند). برای حل این مشکل دو راه وجود دارد: اول آنکه مسیریاب CS به گونه ای پیکربندی شود تا به تمام بسته های ARP که در مورد شبکه 192.31.63.0 (یا سایر شبکه های محلی) آدرسی را سؤال می کنند، جواب بدهد و آدرس سخت افزاری خودش را اعلام نماید. در این حالت، ماشین ۱ در جدول خود یک درایه به صورت (E3 و 192.31.63.8) درج می نماید و تمام ترافیک خود برای ماشین ۴ را برای مسیریاب محلی می فرستد. این راه حل اصطلاحاً Proxy ARP نامیده می شود. (ARP وکالتی) راه حل دوم آنست ماشین ۱ فوراً تشخیص بدهد که مقصد مورد نظر او بر روی شبکه ای دیگر قرار گرفته و تمام ترافیک راه دور و غیر محلی خود را به آدرس اترنت مسیریاب پیش فرض که در این مثال E3 است بفرستد. [تشخیص محلی یا غیر محلی بودن آدرسهای IP به کمک الگوی زیر شبکه Subnet Mask میسر است.] در این راه حل نیازی نیست که مسیریاب CS بداند که به چه شبکه هایی سرویس می دهد.

در هر حال آنچه که در ادامه اتفاق می افتد آنست که: ماشین ۱ بسته IP را درون فیلد حمل داده از فریم اترنت جاسازی کرده و مقصد فریم را آدرس اترنت E3 قرار می دهد. وقتی مسیریاب CS این فریم اترنت را دریافت می کند و بسته IP را از درون آن جدا کرده و آدرس IP آن را درون جدول مسیریابی خود جستجو می نماید و بر این اساس متوجه می شود که تمام بسته های متعلق به شبکه 192.31.63.0 باید به مسیریاب با آدرس 192.31.60.7 تحویل شود و اگر از قبل آدرس سخت افزاری کارت FDDI متناظر با 192.31.60.7 را نداند، با پخش فراگیر یک بسته ARP بر روی حلقه، آن را سؤال کرده و متوجه می شود که این آدرس F3 است. لذا بسته IP را درون فیلد حمل داده از فریم FDDI قرار داده و با درج آدرس F3 در فریم جدید آن را بر روی حلقه قرار می دهد.

در مسیریاب EE، نرم افزار راه انداز کارت FDDI، بسته را از درون فریم استخراج کرده و آن را به نرم افزار IP تحویل می دهد و IP نیز به روش مشابه متوجه می شود که باید این بسته را به آدرس 192.31.63.8 بفرستد. اگر این آدرس IP درون حافظه نهان ARP پیدا نشد، مجدداً با پخش فراگیر یک بسته ARP بر روی شبکه اترنت EE، آن را سؤال می کند و متوجه می شود که آدرس سخت افزاری مقصد E6 است، لذا یک فریم اترنت به آدرس E6 ساخته و بسته را مجدداً درون فیلد حمل داده آن قرار داده و فریم را بر روی شبکه اترنت EE قرار می دهد. وقتی فریم اترنت به ماشین ۴ می رسد، بسته از درون آن استخراج شده و جهت پردازش تحویل نرم افزار IP می شود. حرکت بسته ها از ماشین ۱ به یک شبکه دور در WAN نیز به روشی مشابه با روش قبلی انجام می شود با این تفاوت که در مثال بالا مسیریاب CS از طریق جدول مسیریابی خود متوجه می شود که باید بسته را برای مسیریاب متصل به WAN با آدرس F2 بفرستد.

پروتکل های RARP، BOOTP و DHCP

پروتکل ARP مسئله پیدا کردن آدرس اترنت متناظر با یک IP مشخص را حل می کند. [به عبارت ساده تر آدرس IP یک ماشین را گرفته و آدرس سخت افزاری آن ماشین را پیدا کرده، بر می گرداند.] برخی اوقات وارون این مسئله باید حل شود، یعنی با داشتن آدرس اترنت یک ماشین، به آدرس IP متناظر با آن نیاز است. بالاخص زمانی با این مسئله مواجه هستیم که یک ایستگاه بدون دیسک سخت بخواهد از طریق شبکه بوت شود. بطور معمول چنین ماشینی «تصویر باینری سیستم عامل» خود را از یک سرویس دهنده فایل تحویل گرفته و بارگذاری می کند. ولی چگونه می تواند آدرس IP خود را بفهمد؟

اولین راه حل ابداعی برای این مسئله آن بود که از پروتکل RARP^۱ (تشریح شده در سند RFC 903) استفاده شود. این پروتکل امکان آن را فراهم آورده که ماشین تازه بوت شده آدرس اترنت خود را به صورت فراگیر بر روی شبکه پخش کند و بگوید: «آدرس اترنت ۴۸ بیتی من مثلاً 14.04.05.18.01.25 است. آیا کسی آدرس IP مرا می‌داند؟» سرویس دهنده RARP این درخواست را می‌بیند و در فایل‌های پیکربندی خودش، آدرس اترنت اعلام شده را جستجو می‌نماید و آدرس IP متناظر با آن را بر می‌گرداند.

استفاده از RARP بهتر از آنست که آدرس IP یک ماشین در «تصویر حافظه»^۲ ارسالی جاسازی شود چراکه این امکان فراهم می‌آید که از «تصویر حافظه» مشابهی برای تمام ماشینها بهره گرفته شود. اگر قرار باشد آدرس IP درون تصویر حافظه ارسالی جاسازی شود، هر ایستگاه در شبکه به «تصویر» خاص خود احتیاج خواهد داشت. اشکال RARP آنست که بیت‌های فیلد آدرس مقصد را در تمام فریمهای ارسالی خود ۱ می‌گذارد تا بدین ترتیب این فریمها به صورت پخش فراگیر به سرویس دهنده RARP برسند ولیکن فریمهای پخش شده بر روی شبکه محلی، توسط مسیریابها به خارج از شبکه هدایت نمی‌شوند، فلذا بر روی هر شبکه محلی باید یک سرویس دهنده RARP وجود داشته باشد. برای حل این مشکل یک پروتکل خاص دیگر به نام BOOTP برای راه‌اندازی ایستگاههای بدون دیسک (Diskless) ابداع شده است. این پروتکل می‌تواند به غیر از آدرس IP ایستگاه بدون دیسک، اطلاعات اضافه‌تری را مثل آدرس IP سرویس دهنده فایل (که تصویر اجرایی سیستم عامل را در اختیار دارد)، آدرس IP مسیریاب پیش فرض، الگوی زیرشبکه (Subnet Mask)، به ایستگاهها ارائه بدهد. برخلاف RARP، در پروتکل BOOTP از بسته‌های UDP استفاده شده و مسیریابها این بسته‌ها را هدایت می‌نمایند. [لذا به ازای چندین شبکه محلی که از طریق مسیریاب بهم متصل شده‌اند فقط به یک سرویس دهنده BOOTP نیاز است.] پروتکل BOOTP در RFCهای 951، 1048 و 1084 تشریح شده است.

مشکل جدی پروتکل BOOTP آنست که جدول نگاشت آدرس IP به آدرس اترنت باید به صورت دستی تنظیم و پیکربندی شود. وقتی ماشین جدیدی به LAN اضافه می‌شود قادر به بوت شدن نیست مگر آنکه مسئول شبکه یک آدرس IP به آن انتساب داده و آن را در قالب: (آدرس IP + آدرس اترنت) به صورت دستی در فایل پیکربندی BOOTP وارد نماید. برای آنکه این مرحله اشکال‌زا حذف شود پروتکل BOOTP پیشرفته‌تر شد و با نام جدید DHCP^۳ معرفی گردید. پروتکل DHCP این امکان را فراهم آورده که بتوان آدرس IP ایستگاهها را هم به صورت دستی و هم به صورت خودکار به آنها انتساب داد. این پروتکل در RFCهای ۲۱۳۱ و ۲۱۳۲ تشریح شده است و در اغلب سیستمها جایگزین RARP و BOOTP شده است.

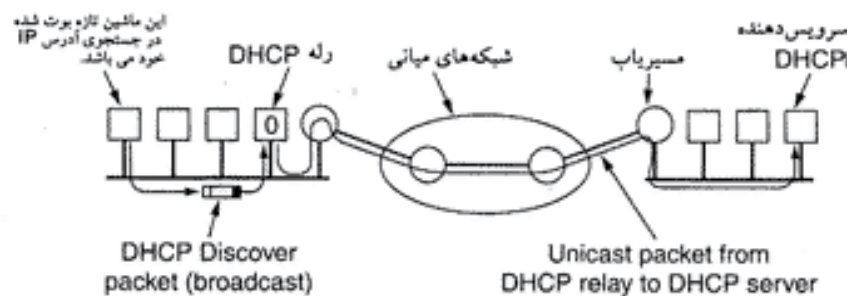
شبیه به RARP و BOOTP، پروتکل DHCP نیز متکی به یک سرویس دهنده ویژه در شبکه است تا به ماشینهایی که تقاضای آدرس IP می‌کنند، آدرس و اطلاعات لازم را تقدیم نماید. لازم نیست که این سرویس دهنده بر روی همان LAN باشد که ماشین متقاضی آدرس بر روی آن واقع شده است و از آنجایی که ممکن است دسترسی به این سرویس دهنده از طریق پخش فراگیر بسته‌های تقاضا میسر نباشد فلذا بر روی هر LAN به یک «عامل رله DHCP» (DHCP Relay Agent) نیاز است. (به شکل ۵-۶۳ نگاه کنید).

ماشینی که تازه بوت شده برای بدست آوردن آدرس IP خود، بسته‌ای به نام DHCP DISCOVER را به صورت فراگیر بر روی LAN خود منتشر می‌کند. «عامل رله DHCP» در هر LAN تمام بسته‌های پخش شده

۱. Reverse Address Resolution Protocol

۲. منظور از تصویر حافظه یا Memory Image قطعه کد اجرایی و باینری از هسته سیستم عامل است که بلافاصله پس از بارگذاری، اجرا می‌شود. —

۳. Dynamic Host Configuration Protocol



شکل ۵-۶۳. عملکرد DHCP.

DHCP را می گیرد و وقتی متوجه شود که یک بسته از نوع DHCP DISCOVER است آن را به صورت تک پخش (Unicast) برای سرویس دهنده اصلی DHCP (که می تواند بر روی شبکه ای دور واقع باشد) می فرستد. «عامل رله DHCP» فقط به آدرس IP از سرویس دهنده DHCP احتیاج دارد. یکی از مسائلی که در خصوص انتساب خودکار آدرسها پیش می آید آنست که یک آدرس IP برای چه مدت زمانی در اختیار ماشین قرار بگیرد، اگر ماشینی بدون اطلاع قبلی شبکه را ترک کند و آدرس IP خود را به سرویس دهنده DHCP برگرداند، آن آدرس برای همیشه گم می شود. به مرور زمان ممکن است آدرسهای زیادیتری به این نحو گم شوند. برای آنکه چنین مشکلی اتفاق نیفتد، انتساب آدرسهای IP فقط برای مدت زمان محدود و ثابتی انجام می شود. این تکنیک «اجاره IP» (IP Leasing) نام دارد. هر ماشین قبل از آنکه مهلت اجاره آدرس IP او به سر برسد، باید با ارسال بسته ای خاص تقاضای تجدید اجاره کند. اگر ماشین موفق به ارسال این تقاضا نشد یا تقاضای تجدید پذیرفته نشود، ماشین میزبان نمی تواند بیش از این از آدرس IP اجاره ای خود استفاده نماید.^۱

۵-۶ OSPF: پروتکل مسیریابی برای دروازه های درونی

تا اینجا بررسی پروتکل های کنترل اترنت را به اتمام رسانده ایم. زمان آن فرا رسیده که به مورد بعدی بپردازیم: «مسیریابی در اترنت». قبلاً اشاره کردیم که شبکه اترنت از تعداد بسیار زیادی «سیستم خودمختار» (Autonomous System) تشکیل شده است. هر سیستم خودمختار که اصطلاحاً AS نامیده می شود توسط سازمان یا نهاد خاصی پیاده و اداره می شود و طبیعی است که آن مؤسسه برای مسیریابی بسته ها در درون شبکه، از الگوریتم مورد نظر خود بهره بگیرد. به عنوان مثال شبکه های داخلی سه شرکت X و Y و Z، از منظر اترنت در قالب سه AS دیده می شود. (البته به شرطی که به اترنت متصل شده باشند). از دیدگاه اترنت، جزئیات درونی یک شبکه AS قابل رویت نیست. علیرغم آنکه جزئیات درونی هر AS مستقل از دیگری است ولیکن داشتن یک استاندارد برای مسیریابی در درون یک AS پیاده سازی «مرز»^۲ AS ها را آسان می کند. در این بخش مسیریابی در درون یک AS و در بخش بعدی مسیریابی بین AS ها را مطالعه خواهیم کرد. مسیریابی در درون یک AS اصطلاحاً «پروتکل دروازه درونی»^۳ و الگوریتم مسیریابی بین AS ها «پروتکل دروازه خارجی»^۴ نامیده می شود.

۱. در DHCP مسئول شبکه تعدادی آدرس IP یا محدوده ای از فضای آدرس IP مورد نظر خود را مشخص می کند تا این سرویس دهنده آنها را برحسب نیاز به ماشینهای شبکه اجاره بدهد. به این آدرسها اصطلاحاً IP Pool گفته می شود. م.

۲. منظور از «مرز» نقطه ای است که AS ها به یکدیگر متصل می شوند و مسیریابهایی که این اتصال را برقرار می کنند مسیریابهای مرزی (Border Gateway) نامیده می شوند. این مسیریابها هم با درون AS و هم با دیگر AS ها در ارتباط فعال هستند؛ یعنی هم مسیرهای بهینه داخلی و هم مسیرهای بهینه بین AS ها را می دانند. م.

۳. Interior Gateway Protocol

۴. Exterior Gateway Protocol

اولین پروتکل «دروازه درونی اینترنت»، یک پروتکل بردار فاصله با نام RIP بود که از الگوریتم «پلمن-فوردر» بهره می‌گرفت. [بخش ۵-۲-۴] اگرچه این پروتکل در سیستمهای کوچک به خوبی کار می‌کند ولی با بزرگ شدن ASها، کارایی خود را از دست می‌دهد. این پروتکل همچنین از مشکل «شمارش تا بی‌نهایت» رنج می‌برد و «همگرایی» کندی دارد. به همین دلیل در ماه می ۱۹۷۹ یک پروتکل مبتنی بر «حالت لینک» (Link State) جایگزین آن شد. در ۱۹۸۸، IETF کار را بر روی یک پروتکل جدید آغاز کرد. این پروتکل OSPF (Open Shortest Path First) نام گرفت و در سال ۱۹۹۰ استاندارد شد. امروزه اغلب تولیدکنندگان مسیریاب از آن حمایت می‌کنند و تقریباً به مهمترین پروتکل مسیریابی دروازه‌های درونی تبدیل شده است. در زیر شمائی از عملکرد OSPF ارائه می‌دهیم. برای آگاهی دقیقتر از کل قضیه به سند RFC 2328 مراجعه نمایید.

گروه طراح این پروتکل با داشتن تجربه طولانی از دیگر پروتکلهای مسیریابی، تصمیم گرفتند با تدوین فهرست بالا بلندی از نیازها و انتظارات، آنرا از نو طراحی کنند. نخستین نیاز آن بود که به صورت «باز» (Open) طراحی شود بدین معنا که امتیاز راه‌حلهای و روشهای خاص آن برای موسسه خاصی ثبت نشود و همچنین به بستر سخت‌افزاری یا نرم‌افزاری ویژه وابسته نباشد. حرف 'O' در نام OSPF به همین مضمون است.

نیاز دوم آن بود که پروتکل جدید بتواند برای تعیین مسیر بهینه، از معیارهای گوناگون هزینه مثل معیار «فاصله فیزیکی»، «تاخیر» و نظایر آن پشتیبانی کند. نیاز سوم آن بود که الگوریتم پویا باشد و هر گونه تغییر در توپولوژی زیرشبکه را به سرعت و به صورت خودکار تشخیص داده و خود را با آن تطبیق بدهد.

نیاز چهارم آنکه OSPF می‌بایست بسته‌ها را برحسب نوع خدمات درخواستی، مسیریابی و هدایت نماید. این پروتکل باید قادر می‌بود که ترافیک داده‌های بی‌درنگ را نسبت به ترافیک داده‌های معمولی به روش متفاوتی مسیریابی نماید. پروتکل IP در هر بسته یک فیلد به نام Type of Service تعریف کرده بود که هیچ پروتکل مسیریابی از آن حمایت نمی‌کرد. اگرچه این فیلد در OSPF گنجانده شد ولیکن باز هم کسی از آن استقبال نکرد و عاقبت حذف گردید.

نیاز پنجم (که با موارد فوق مرتبط است) آن بود که پروتکل جدید مکانیزم «موازنه بار» (Load Balancing) را اعمال کند و بار را بر روی چندین خط تقسیم نماید. اغلب پروتکلها تمام بسته‌ها را بر روی بهترین مسیر ارسال می‌کنند و از مسیری که از لحاظ بهینگی در رتبه دو قرار می‌گیرد هیچ استفاده‌ای نمی‌شود؛ در بسیاری از حالات تقسیم بار بر روی چندین خط، کارایی بهتری دارد.

ششم آنکه نیاز بود از مسیریابی سلسله‌مراتبی پشتیبانی شود. تا سال ۱۹۸۸ اینترنت آنقدر بزرگ شده بود که نمی‌شد انتظار داشت یک مسیریاب بتواند توپولوژی کل آن را بداند. پروتکل جدید می‌بایست به گونه‌ای طراحی می‌شد که هیچ مسیریابی نیاز به دانستن توپولوژی کل شبکه نداشته باشد.

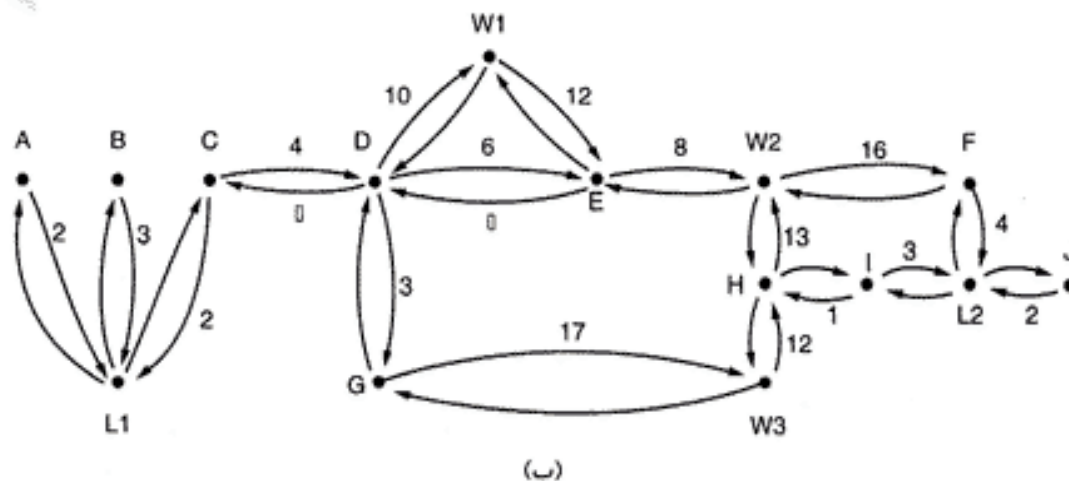
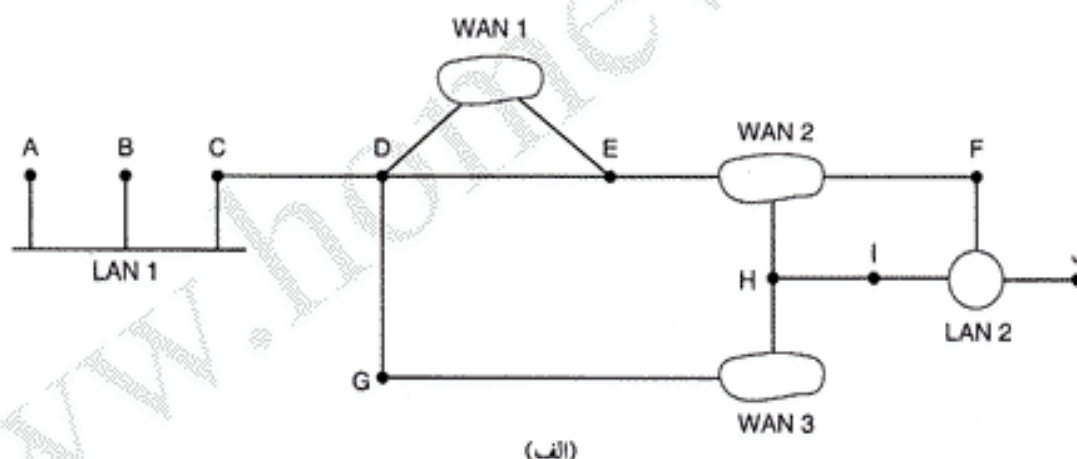
هفتم آنکه به سطحی از امنیت نیاز بود تا یک دانشجوی بازیگوش نتواند با ارسال اطلاعات جعلی و نادرست در خصوص مسیرها، مسیریاب را به اشتباه بیندازد. در آخر آنکه به تمهیداتی نیاز بود تا مسیریابهایی که با مکانیزم «ایجاد تونل» (Tunneling) و از طریق اینترنت بهم متصل می‌شوند را به درستی مدیریت نماید. OSPF از سه نوع شبکه و خطوط اتصال پشتیبانی می‌کند:

۱. خطوط نقطه به نقطه بین دو مسیریاب
۲. شبکه‌های با دسترسی چندگانه که کانال آنها از نوع پخش فراگیر (Broadcast) است. (مثل اغلب شبکه‌های LAN)
۳. شبکه‌های با دسترسی چندگانه بدون آنکه کانال آنها از نوع پخش فراگیر باشد. (مثل اغلب شبکه‌های سوئیچ بسته در WAN)

شبکه با دسترسی چندگانه (Multiaccess Network)، شبکه‌ای است که می‌تواند چندین مسیر یاب داشته باشد و هر یک از مسیر یابها می‌توانند مستقیماً یا یکدیگر در ارتباط باشند. تمام شبکه‌های LAN و WAN دارای این ویژگی هستند. شکل ۵-۶۴-الف یک AS را نشان می‌دهد که در آن هر سه نوع شبکه وجود دارد. دقت کنید که از دیدگاه OSPF، ماشینهای میزبان شبکه عموماً نقش خاصی را ایفاء نمی‌کنند.

OSPF بدین نحو عمل می‌کند که مجموعه شبکه‌ها، مسیر یابها و خطوط ارتباطی را در قالب یک «گراف جهت‌دار» (Directed Graph) مدل می‌کند و به هر کمان در گراف (Arc) یک مقدار «هزینه» (مثل تأخیر، فاصله یا امثال آن) انتساب می‌دهد. سپس براساس وزن هر یک از کمانها، مسیر بهینه را پیدا می‌کند. یک خط ارتباطی سریال بین دو مسیر یاب، در گراف با یک جفت کمان (Arc) نشان داده می‌شود (یک کمان به ازای هر جهت) و وزنهای هر کمان می‌تواند با دیگری متفاوت باشد. یک شبکه با دسترسی چندگانه (LAN)، با یک گره به ازای خود شبکه و یک گره به ازای هر مسیر یاب مدل می‌شود. وزن کمانی که از گره شبکه به یک مسیر یاب وارد می‌شود، صفر در نظر گرفته شده و از گراف حذف می‌گردد.

شکل ۵-۶۴-ب، نمایش گراف متناظر با شبکه ۵-۶۴-الف را نشان می‌دهد. وزنهای متناظر هستند مگر آنهایی که به صراحت مشخص شده‌اند. آنچه که OSPF انجام می‌دهد مدل کردن شبکه در قالب گرافی شبیه به این مثال و سپس محاسبه مسیرهای بهینه از هر مسیر یاب به هر مسیر یاب دیگر است.



شکل ۵-۶۴. (الف) یک سیستم خودمختار (ب) نمایش گراف از شکل الف.

بسیاری از AS ها در اینترنت خودشان بسیار عظیم هستند و مدیریت آنها ساده نیست. OSPF این امکان را فراهم آورده که بتوان چنین شبکه هایی را به تعدادی «ناحیه شماره گذاری شده» (Numbered AREA) تقسیم کرد. هر ناحیه خود یک شبکه یا مجموعه ای از شبکه های بهم پیوسته مجاور است. نواحی با یکدیگر همپوشانی ندارند^۱ ولی لازم نیست که نواحی تعریف شده کل شبکه AS را پوشش بدهد و ممکن است برخی از مسیر یابها در هیچ ناحیه ای قرار نگیرند. یک ناحیه، شکل کلی و عمومی یک زیر شبکه مستقل است و در خارج از ناحیه، توپولوژی و جزئیات درونی آن مشهود نیست.

در هر شبکه خود مختار (AS) ناحیه ای به نام «ستون فقرات» وجود دارد که ناحیه صفر نامیده می شود. تمام نواحی به ستون فقرات متصل می شوند (به صورت مستقیم یا توسط ایجاد تونل) لذا براحتی می توان به کمک ستون فقرات از هر ناحیه در شبکه AS به ناحیه دیگر رفت. یک «تونل» نیز در گراف توسط یک کمان مشخص می شود که دارای هزینه است. هر مسیر یاب که به دو یا چند ناحیه متصل باشد (یعنی مسیر یابهای مشترک بین دو یا چند ناحیه) جزیی از ستون فقرات شبکه محسوب می شود. همانند بقیه نواحی، توپولوژی ناحیه ستون فقرات نیز در خارج از آن مشهود نیست. (بعبارتی مسیر یابهای درون دیگر نواحی از توپولوژی ستون فقرات چیزی نمی دانند.)

درون یک ناحیه، هر یک از مسیر یابها نسخه مشابهی از پایگاه اطلاعاتی در خصوص مسیرها و هزینه ها در اختیار دارند و الگوریتم محاسبه کوتاهترین مسیر آنها یکسان است. هر مسیر یاب وظیفه دارد که کوتاهترین مسیرها از خودش به تمام مسیر یابهای دیگر ناحیه را محاسبه نماید. از جمله هر مسیر یاب باید کوتاهترین مسیر از خود تا یک مسیر یاب واقع بر روی ستون فقرات را پیدا کند. یک مسیر یاب که در مرز دو ناحیه واقع است باید پایگاه اطلاعاتی هر دو ناحیه را در اختیار داشته باشد و الگوریتم کوتاهترین مسیر را بطور جداگانه بر روی آنها اجرا کند. [تا تمام مسیرهای بهینه از خودش تا بقیه مسیر یابها در هر دو ناحیه بدست بیاید.]

در حین عملیات طبیعی، احتمالاً به سه نوع مسیر نیاز است: (۱) مسیرهای درون ناحیه^۲ (Intra-Area) (۲) مسیرهای بیرون ناحیه (Inter-Area) (۳) مسیرهای بین AS (Inter-AS).

مسیرهای درون ناحیه ساده ترین نوع مسیر هستند چرا که مسیر یاب مبدا، از قبل و به دقت کوتاهترین مسیر رسیدن به مسیر یاب مقصد را می داند. مسیر یابی بین نواحی (Inter-Area)، همیشه در سه مرحله انجام می گیرد: حرکت از مبدا تا ستون فقرات، سپس حرکت از ستون فقرات به سوی ناحیه مقصد و نهایتاً حرکت در درون ناحیه به سمت مقصد. این الگوریتم ایجاب می کند که در پروتکل OSPF یک توپولوژی «ستاره» برای مسیر یابها قائل شویم: ستون فقرات در مرکز قرار می گیرد و بقیه نواحی از آن منشعب می شوند. در OSPF بسته ها به همان نحوی که از مبدا تولید شده اند به سوی مقصد مسیر یابی و هدایت می شوند یعنی بسته ها در درون بسته دیگری «جاسازی» (Encapsulate) یا «تونل» (Tunnel) نمی شوند، مگر آنکه بسته مجبور باشد برای رسیدن از یک ناحیه به ستون فقرات، از مسیری حرکت کند که از طریق تونل ایجاد شده است.^۳ شکل ۵-۶۵ بخشی از اینترنت را با چهار AS و تعدادی ناحیه نشان می دهد.

OSPF چهار کلاس مسیر یاب را به رسمیت می شناسد:

۱. مسیر یابهای درونی (که کاملاً در داخل یک ناحیه قرار می گیرند).
۲. مسیر یابهای واقع بر مرز دو ناحیه (که دو یا چند ناحیه را به هم متصل می کنند).

۱. یعنی هر مسیر یاب صرفاً به یک ناحیه متعلق است. -م

۲. یعنی مسیرهایی که از یک مسیر یاب در درون ناحیه شروع و به یک مسیر یاب در همان ناحیه ختم می شود. -م

۳. مفهوم تونل را در بخش ۵-۵-۵ مطالعه نمایند.

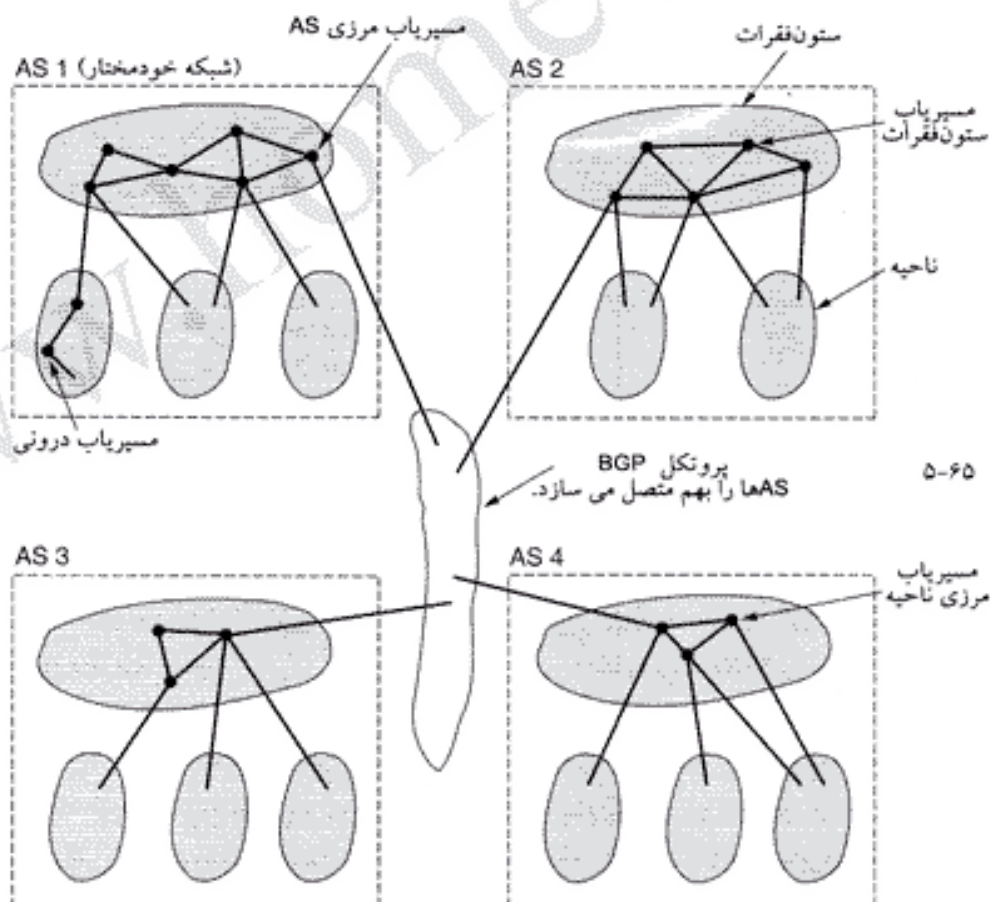
۳. مسیریابهای ستون فقرات (که بر روی ستون فقرات قرار می گیرند).

۴. مسیریابهای مرزی AS که می توانند با مسیریابهای AS دیگر محاوره کنند.

البته امکان آن وجود دارد که مسیریابها در دو یا چند کلاس قرار بگیرند. به عنوان مثال تمام مسیریابهای مرزی، به صورت خودکار جزئی از ستون فقرات نیز هستند. مضاف بر این، یک مسیریاب که بر روی ستون فقرات واقع است ولی در هیچ ناحیه ای قرار نگرفته، خودش یک مسیریاب داخلی محسوب می شود. در شکل ۵-۶۵ نمونه ای از تمام کلاسهای مسیریاب، دیده می شود.

وقتی یک مسیریاب بوت می شود، ابتدا بر روی تمام خطوط نقطه به نقطه و کانالهای اشتراکی (مثل کانالهای LAN که چند مسیریاب از طریق آن بهم متصل شده اند)، پیامی به نام «پیام سلام» (Hello Message) می فرستد. در خطوط WAN [مثل خطوط ISDN] احتمالاً قبل از هر گونه مبادله پیام به اطلاعات پیکربندی خاص نیاز است تا هویت تماس گیرنده مشخص شود.^۱ هر مسیریاب با دریافت پاسخ سلام، مسیریابهای همسایه خود را شناسایی می کند. مسیریابهایی که به یک LAN واحد متصلند، همگی همسایه یکدیگر محسوب می شوند.

عملکرد OSPF مبتنی بر مبادله اطلاعات با «مسیریابهای مجاور» (Adjacent) است. مفهوم مسیریابهای مجاور با مفهوم «مسیریابهای همسایه» (Neighbor) فرق می کند؛ اینکه هر مسیریاب متصل به LAN بتواند با مسیریاب دیگر محاوره و مبادله پیام کند کافی و کارآمد نیست. برای اجتناب از این وضعیت، از بین کل مسیریابهای



شکل ۵-۶۵. ارتباط بین AS ها، ستون فقرات و نواحی در OSPF.

واقع بر LAN، یک مسیریاب به نام «مسیریاب برگزیده» (Designated Router) انتخاب می گردد. این مسیریاب «مسیریاب مجاور» بقیه مسیریابهای واقع بر LAN محسوب می شود و با آنها به مبادله اطلاعات می پردازد. برای «مسیریاب برگزیده» یک مسیریاب پشتیبان نیز در نظر گرفته شده که آن نیز همیشه اطلاعات به روز از وضعیت مسیرها در اختیار دارد تا در صورت از کار افتادن «مسیریاب برگزیده»، به سرعت جایگزین آن شود.

در حین عملکرد طبیعی، هر مسیریاب بطور متناوب پیامهای LINK STATE UPDATE خود را به صورت سیل آسا (Flooding) برای تمام مسیریابهای مجاور خود می فرستد. [این پیامها حاوی اطلاعاتی در خصوص هویت همسایه ها، لینکها و هزینه آنهاست.] بدین ترتیب، با این پیامها حالت هر لینک (Link State) و هزینه آن، برای درج در «پایگاه اطلاعات توپولوژیکی» به دست می آید. وصول پیامهایی که به صورت سیل آسا ارسال می شوند، تصدیق خواهد شد تا از دریافت آنها اطمینان حاصل شود.^۱ هر پیام یک شماره ترتیب دارد تا مسیریاب بفهمد آیا پیام LINK STATE UPDATE قدیمی یا جدید است. ارسال این پیامها به صورت سیل آسا زمانی آغاز می شود که خطی از کار بیفتد یا مجدداً فعال شود یا هزینه آنها تغییر کند. البته حتی اگر چنین اتفاقی نیفتد، این پیامها بطور متناوب و هر از چند ده ثانیه ارسال خواهند شد.

پیام DATABASE DESCRIPTION تمام شماره های ترتیب از درایه های جدول حالت لینک را که اخیراً در پایگاه اطلاعاتی فرستنده آن ذخیره شده، اعلام می کند. هر یک از گیرندگان این پیام، شماره های اعلام شده را با شماره های ترتیب درایه های خودش مقایسه می کند تا بفهمد چه کسی جدیدترین مقادیر را در اختیار دارد. از این پیامها زمانی استفاده می شود که خطی فعال شود.

هر یک از طرفین یک لینک می توانند با استفاده از پیام LINK STATE REQUEST، «اطلاعات حالت پیوند» یکدیگر را مطالبه نمایند. نتیجه این الگوریتم آنست که هر جفت مسیریاب مجاور، آخرین اطلاعات به روز شده یکدیگر را بررسی کرده و بدین نحو اطلاعات جدید در کل ناحیه پخش می شود. تمام این پیامها به کمک بسته های معمولی IP ارسال می شود. فهرست پنج پیام فوق در جدول ۵-۶۶ خلاصه شده است.

نوع پیام	توصیف عملکرد
Hello	از این پیام برای شناسایی همسایه ها استفاده می شود.
Link state update	هزینه فرستنده پیام تا همسایه هایش را معین می کند.
Link state ack	دریافت بسته Link State Update را تایید می کند.
Database description	مسیریاب با این پیام فهرست درایه های بهنگام سازی خود را اعلام می کند.
Link state request	از شریک خود اطلاعاتی را درخواست می کند.

شکل ۵-۶۶. پنج نوع از پیامهای OSPF.

حال می توان مجموعه عوامل فوق را بدین نحو جمع بندی کرد: به کمک روش ارسال سیل آسا، هر مسیریاب به تمام اعضاء ناحیه خود، از همسایه هایش و هزینه رسیدن به آنها خبر می دهد. این اطلاعات امکان آن را فراهم می آورد تا یکایک مسیریابها بتوانند گراف ناحیه خود را تشکیل داده و کوتاهترین مسیرها را محاسبه نمایند. ستون فقرات نیز همین کار را می کند. [چراکه ستون فقرات خود یک ناحیه مستقل است.] مضاف بر این، مسیریابهای واقع بر روی ستون فقرات، اطلاعات ارسالی از مسیریابهای مرزی هر ناحیه را هم می پذیرند تا بتوانند کوتاهترین مسیرها از ستون فقرات تا دیگر مسیریابها را محاسبه نمایند. این اطلاعات مجدداً به مسیریابهای مرزی هر ناحیه

۱. یعنی به ازای دریافت هر پیام از این نوع یک پیام ACK بر می گردد.

بازگردانده می شود تا آنها نیز در درون ناحیه خودشان اعلام نمایند. به کمک این اطلاعات، یک مسیریاب که می خواهد بسته ای را به خارج از ناحیه خود بفرستد، بهترین مسیریاب مرزی متصل به ستون فقرات را به عنوان دروازه خروج بسته برمیگزیند.

۵-۶-۵ BGP^۱: پروتکل مسیریابی برای دروازه خارجی

در درون یک AS واحد، پروتکل مسیریابی OSPF، بهترین گزینه است (اگرچه OSPF، تنها پروتکل مورد استفاده و رایج به حساب نمی آید). بین AS ها از پروتکل متفاوتی به نام BGP استفاده می شود. به دلیل آنکه اهداف پروتکل مسیریابی درونی با پروتکل مسیریابی خارجی فرق می کند به پروتکل متفاوتی برای مسیریابی بین AS ها نیاز است. پروتکل مسیریابی درونی باید به سریعترین وجه ممکن بسته ها را از مبدأ به مقصد برساند و اهمیتی به سیاستهای جانبی نمی دهد.

پروتکلهای مسیریابی خارجی بایستی در حد وسیعی ملاحظات سیاسی را مدنظر قرار بدهند. (Metz, 2001) برای مثال ممکن است یک شبکه AS بخواهد که بسته ها را به هر سایت اینترنت بفرستد یا بسته ها را از هر سایت اینترنت دریافت کند ولی تمایلی به حمل بسته هایی که از یک AS خارجی تولید شده و به یک AS خارجی دیگر می رود نداشته باشد، حتی اگر AS او بر روی کوتاهترین مسیر بین دو AS خارجی قرار گرفته باشد. (یعنی با این استدلال که: «مشکل آنها به ما ربطی ندارد!» بسته های دیگران را ترانزیت نکنند.) از طرف دیگر ممکن است بخواهد در ازای دریافت هزینه خدمات، ترافیک همسایه های خود و AS های خاصی را مسیریابی و هدایت کند. مثلاً شرکتهای مخابرات تلفن ممکن است مایل باشند به عنوان حامل مشتریان خود عمل کنند ولی نه برای دیگران (و بدون اخذ وجه)! پروتکلهای مسیریابی خارجی بطور عام و پروتکل BGP بطور خاص این امکان را فراهم کرده اند که بتوان سیاستهای گوناگونی را بر روی ترافیک بین AS ها اعمال کرد. سیاستهای کلی حول مسائل سیاسی، امنیتی یا ملاحظات اقتصادی دور می زنند. چند مثال از ملاحظات مسیریابی عبارتند:

۱. عدم اجازه عبور ترافیک داده ها از میان AS های خاص
۲. مسیری که از پنتاگون شروع می شود هرگز از عراق عبور نکند!
۳. مسیر رسیدن از بریتیش کلمبیا به اونتاریو از ایالات متحده نگذرد!
۴. فقط زمانی از مسیر آلبانی عبور شود که هیچ راه دیگری به مقصد وجود نداشته باشد!
۵. ترافیک داده هایی که از IBM شروع شده یا بدان ختم می شود از مایکروسافت عبور نکند!!

عموماً سیاستهای مسیریابی، به صورت دستی بر روی مسیریابهای BGP تنظیم و پیکربندی می شود (یا در قالب نوعی «اسکرپت» بر روی مسیریاب اجرا می گردد). البته این تنظیمات جزئی از پروتکل بحساب نمی آید. از دیدگاه یک مسیریاب BGP، کل جهان از چندین AS و خطوط ارتباطی بین آنها تشکیل شده است. دو AS وقتی متصل تلقی می شوند که بین مسیریابهای مرزی هر یک از آنها حداقل یک خط وجود داشته باشد. با توجه به تأکید BGP بر حمل (یا عبارتی ترانزیت) ترافیک، هر شبکه AS در یکی از سه رده ذیل قرار می گیرد: (۱) رده اول «شبکه های پایانی» (Stub Network) نامیده می شوند؛ این گونه از شبکه ها فقط و فقط یک اتصال با گراف BGP دارند و نمی توانند ترافیک داده ها را از خود عبور بدهند چرا که در طرف دیگر آنها شبکه ای وجود ندارد. (۲) رده بعدی، «شبکه های چنداتصال» (Multiconnected Networks) هستند. این شبکه ها می توانند حمل ترافیک

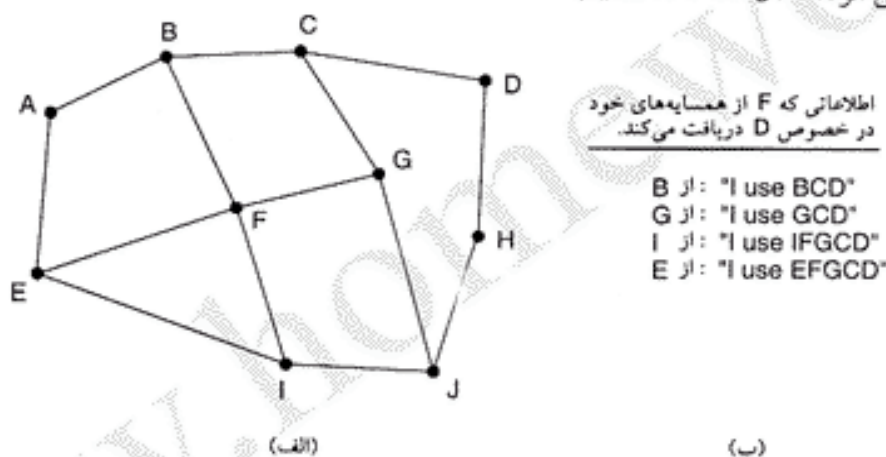
۱. Border Gateway Protocol

۲. شبکه های پایانی را به مثابه یک کوچه بن بست فرض کنید. -م

بسته های دیگران را بر عهده بگیرند مگر آنکه عمداً از این کار امتناع کنند. (۳) در رده آخر، «شبکه های ترانزیت» (Transit Network) قرار می گیرند که در نقش ستون فقرات، تمایل دارند بسته های شبکه دیگران را (طبق برخی از محدودیتها و احتمالاً دریافت هزینه) حمل (ترانزیت) کنند.

یک جفت مسیر یاب BGP از طریق برقراری یک اتصال TCP، با یکدیگر مبادله اطلاعات می نمایند. این مکانیزم، امکان مبادله مطمئن و بدون خطای داده ها را فراهم کرده و جزئیات درونی شبکه را پنهان نگاه می دارد. پروتکل BGP ذاتاً مبتنی بر الگوریتم بردار فاصله است ولی از بسیاری جهات با RIP فرق دارد؛ تفاوت پنیانی در اینجا است که به جای آنکه فقط هزینه و خط رسیدن به یک مقصد در شبکه محاسبه و نگهداری شود، مسیر یاب BGP کل مسیر رسیدن به هر مقصد را نگه می دارد. همچنین به جای آنکه به همسایه های خود در خصوص هزینه رسیدن به هر مقصد در شبکه، خبر بدهد کل مسیرهای واقعی را اعلام می کند.

به عنوان مثال مسیر یابهای BGP در شکل ۵-۶۷-الف و به ویژه جدول مسیر یابی F را در نظر بگیرید. فرض کنید که این مسیر یاب برای رسیدن به D از مسیر FGCD استفاده می کند. وقتی همسایه ها اطلاعات مسیر یابی خود را به او می دهند مسیرهای کامل را مشابه فهرست ۵-۶۷-ب اعلام می نمایند. (برای سادگی فقط بخشی از مسیرها که به D ختم می شوند نشان داده شده است.)



شکل ۵-۶۷. (الف) مجموعه ای از مسیر یابهای BGP. (ب) اطلاعات ارسالی برای F.

پس از آنکه F تمام مسیرها را از همسایه های خود دریافت کرد، آنها را بررسی می کند تا ببیند کدامیک از آنها بهترین است. مثلاً F برای تعیین بهترین مسیر رسیدن به D، فوراً مسیرهای اعلام شده توسط I و E را حذف می کند. چرا که این مسیرها خودشان از F می گذرند. انتخاب نهایی از بین مسیرهای اعلام شده B و G است. هر مسیر یاب BGP دارای یک ماحول خاص است که مسیرهای رسیدن به یک مقصد خاص را بررسی کرده و به آنها «نمره» می دهد و در نهایت برای هر مسیر عددی را به عنوان «معیار فاصله» (Distance) بر می گرداند. هر مسیری که ملاحظات و محدودیتهای اتخاذ شده را نقض کرده باشد، نمره بی نهایت می گیرد. پس از این مرحله، مسیر یاب مسیری با کمترین هزینه را می پذیرد. تابع نمره دهی^۱ به مسیرها بخشی از پروتکل BGP بحساب نمی آید و می تواند به دلخواه مدیر سیستم انتخاب و تعریف شود.

پروتکل BGP به سادگی مشکل «شمارش تابی نهایت» (که الگوریتمهای بردار فاصله را آزار می دهد) حل کرده است.^۲ به عنوان مثال فرض کنید G از کار بیفتند یا خط FG قطع شود. لذا F فقط از سه همسایه باقیمانده خود (یعنی B و I و E) اطلاعات مسیر می گیرد. مسیرهایی که همسایه های او برای رسیدن به D اعلام می کنند عبارتند از:

۲. زیرا مسیرهای کامل به همسایه ها اعلام می شود. بخش ۵-۲-۴ را ببینید. -

۱. Scoring Function.

BCD، IFGCD و EFGCD. مسیریاب F بلافاصله متوجه می شود که مسیرهای دوم و سوم بی فایده هستند چراکه از خود F می گذرند، لذا FBCD به عنوان مسیر جدید انتخاب می شود. الگوریتمهای دیگر «بردار فاصله» بدان دلیل در انتخاب مسیر به اشتباه می افتند که همسایه ها نمی توانند اعلام کنند کدامیک از مسیرهای رسیدن به مقصد، مسیر مستقلی است. (یعنی از خود همسایه نمی گذرد). پروتکل BGP در اسناد RFC 1771 تا RFC 1774 تشریح شده است.

۶-۶-۵ ارسال چندپخش در اینترنت (Internet Multicasting)

عملکرد طبیعی پروتکل IP، مبادله بسته های داده بین یک گیرنده و یک فرستنده است؛ با این وجود در برخی از کاربردها این قابلیت که یک پروسه بتواند بطور همزمان برای تعداد بسیار زیادی گیرنده، ارسال داشته باشد، کارساز و مفید است. مثالهایی از این قبیل عبارتست از: به هنگام سازی همزمان نسخه های توزیع شده یا تکراری پایگاههای اطلاعاتی، اعلام همزمان قیمت سهام به متولیان و واسطه ها، مدیریت و اجرای کنفرانسهای دیجیتال و گردهم آیی تلفنی.

IP با استفاده از آدرسهای کلاس D از فرآیند چندپخش (Multicasting) پشتیبانی می نماید. هر آدرس کلاس D، هویت یک «گروه» از ماشینها را مشخص می کند. برای مشخص کردن هر گروه در کلاس D، ۲۸ بیت در اختیار است، لذا بطور همزمان می توان بیش از ۲۵۰ میلیون گروه تعریف کرد. هرگاه یک پروسه، بسته ای را به یک آدرس کلاس D ارسال کند، برای رساندن آن بسته به یکایک اعضای گروه، مسیریابها حداکثر تلاش ممکن را بعمل می آورند ولیکن هیچ تضمینی داده نمی شود و احتمال دارد برخی از اعضا بسته را دریافت نکنند!

در IP از دو نوع آدرس گروهی حمایت می شود: آدرسهای دائم (Permanent) و آدرسهای موقت (Temporary). «گروه دائم» همیشه وجود دارد و نیازی به تنظیم و هماهنگی قبلی نیست. هر گروه دائم دارای آدرس ثابت و بلا تغییر است. چند نمونه از آدرس گروههای ثابت عبارتند از:

گروه 224.0.0.1 : تمام سیستمهای متصل به یک LAN

گروه 224.0.0.2 : تمام مسیریابهای متصل به یک LAN

گروه 224.0.0.5 : تمام مسیریابهای OSPF متصل به یک LAN

گروه 224.0.0.6 : تمام مسیریابهای «برگزیده» OSPF متصل به یک LAN

گروههای موقتی باید قبل از استفاده، ایجاد شوند. یک پروسه می تواند از ماشین خود درخواست کند که به یک گروه خاص بپیوندد؛ همچنین می تواند از آن بخواهد که گروه را ترک کند. وقتی که آخرین پروسه در یک ماشین، گروهی را ترک کند، ماشین مربوطه از آن گروه خارج خواهد شد. هر ماشین می داند که پروسه های او عضو چه گروههایی هستند.

ارسال چندپخش توسط مسیریابهای خاصی پیاده سازی و پشتیبانی می شود و طبعاً ممکن است در کنار مسیریابهای استاندارد، اینگونه مسیریابها وجود نداشته باشد، لذا توانایی ارسال چندپخش به نوع مسیریابهای شبکه و پیکربندی آنها برمی گردد. مسیریابهای چندپخش تقریباً در هر دقیقه یکبار، یک پیام سخت افزاری (یعنی یک فریم لایه پیوند داده ها) را به صورت چندپخش برای تمام ماشینهای متصل به LAN خودش (یعنی به آدرس 224.0.0.1) می فرستد و از آنها می خواهد که اعلام کنند پروسه هایشان در چه گروههایی عضو هستند. در پاسخ به این سؤال، یکایک ماشینها، آدرسهای کلاس D گروههای مورد نظر خود را برمی گردانند.

این بسته های پرسش و پاسخ مبتنی بر پروتکلی به نام IGMP^۱ است که تا حدودی به ICMP شبیه است. در

۱. Internet Group Management Protocol

این پروتکل فقط دو نوع بسته تعریف شده است: بسته پرسش و بسته پاسخ. هر یک از این بسته ها دارای قالبی ساده و ثابت هستند: در اولین کلمه از فیلد داده (Payload) برخی اطلاعات کنترلی قرار می گیرد و در کلمه بعدی یک آدرس کلاس D درج می شود. [کلمات چهار بیتی هستند.] این پروتکل در سند RFC 1112 تشریح شده است.

مسیریابی چندپخشی در اینترنت به کمک «درختهای پوشا» (Spanning Tree) انجام می شود. هر مسیریاب که از فرآیند چندپخشی حمایت می کند، به کمک «پروتکل اصلاح شده بردار فاصله» با همسایه های خود اطلاعاتی را مبادله می کند تا هر کدام بتوانند به ازای هر گروه یک «درخت پوشا» تشکیل بدهند. (به نحوی که تمام اعضای هر گروه را در بر بگیرد.) به منظور آنکه درخت به نحوی سازماندهی و پیرایش شود تا مسیریابها یا شبکه های غیر عضو در گروه از ساختار درخت حذف گردند، بهینه سازیهایی صورت گرفته است. این پروتکل از مکانیزم «ایجاد تونل» استفاده زیادی می کند تا بدین نحو برای گره هایی که در درخت پوشا قرار ندارند سردرگمی و اشکال ایجاد نشود.

۷-۶-۵ IP متحرک (Mobile IP)

بسیاری از کاربران اینترنت از کامپیوترهای کیفی قابل حمل استفاده می کنند و طبعاً علاقمندند وقتی به یک محل جدید نقل مکان می نمایند یا حتی در طول راه، اتصالشان با اینترنت برقرار بماند. متأسفانه، سیستم آدرس دهی IP به گونه ای طراحی شده که کار کردن با آن دور از خانه، فقط در گفتار ساده است تا در عمل! در این بخش این مشکل و راه حل آن را بررسی می کنیم. (Perkins, 1998a)

نقطه ضعف واقعی IP، در روش آدرس دهی آن نهفته است. یک آدرس IP متشکل از یک شماره شبکه و یک شماره ماشین است. به عنوان مثال ماشینی با آدرس 160.80.40.20/16 را در نظر بگیرید. شماره شبکه 160.80 (یا 8272 در مبنای ده) و شماره ماشین 40.20 (یا 10260 در مبنای ده) است. مسیریابهای کل جهان یک جدول مسیریابی دارند که در آن خط مناسب برای رسیدن به شبکه 160.80 مشخص شده است. وقتی بسته ای با آدرس IP به شکل 160.80.xxx.yyy، به مسیریاب وارد گردد بر روی آن خط ارسال خواهد شد.

بدین نحو اگر ماشینی با آدرس فوق به محل جدیدی نقل مکان کند، کماکان بسته های ارسالی برای او به شبکه یا مسیریاب خانگی هدایت شده و از آن به بعد صاحب ماشین قادر به دریافت نامه های الکترونیکی یا نظائر آن نخواهد بود. انتساب آدرس IP جدید (متناظر با محل جدید) به ماشین سیار، راهکار جالبی نیست چراکه این تغییر باید به تعداد زیادی از افراد، بانکهای اطلاعاتی و برنامه های کاربردی اطلاع داده شود.

راهکار دیگر آنست که مسیریابها در جدول مسیریابی به جای آنکه فقط شماره شبکه را نگه دارند آدرس کامل ماشینها را درج نمایند ولیکن این روش نیز مستلزم ذخیره میلیونها درایه (Entry) در جدول مسیریابی است و هزینه های نجومی به اینترنت تحمیل می کند.

از زمانی که تقاضا برای این قابلیت (که افراد بتوانند در هر کجا که هستند به اینترنت متصل شوند)، بالا گرفت، IETF یک گروه ویژه برای پیدا کردن راه حل مناسب تشکیل داد. این گروه کاری به سرعت اهدافی را که هر پیشنهاد یا راه حل باید آنها را برآورده می ساخت، تدوین و تعریف کردند. مهمترین این اهداف عبارت بود از:

۱. هر ماشین متحرک باید بتواند هر جا که هست از آدرس IP همیشگی و خانگی خود استفاده کند.
۲. هر گونه تغییر نرم افزاری در ماشینها مجاز شمرده نمی شود.
۳. هر گونه تغییر در نرم افزار مسیریابها یا جداول آنها مجاز نیست.
۴. اکثر بسته هایی که به سوی ماشینهای متحرک هدایت می شوند نباید از مسیر واقعی منحرف شوند.

۵. وقتی ماشین میزبان در محل همیشگی خود است نباید هیچ سربرار اضافهای تحمل کند.^۱

راه حل انتخابی همانی بود که در بخش ۵-۲-۸ آن را تشریح کردیم. در یک مرور اجمالی: هر سایتی که تمایل داشته باشد امکان سیار بودن را برای کاربران خود فراهم کند موظف به ایجاد یک «عامل خانگی» (Home Agent) است. همچنین هر سایتی که تمایل به پذیرش کاربران خارجی (کاربران سیار) داشته باشد موظف به ایجاد یک «عامل خارجی» (Foreign Agent) است. هر گاه یک ماشین متحرک در محدوده یک سایت خارجی ظاهر می شود، ابتدا با عامل خارجی تماس برقرار کرده و ثبت نام می کند. عامل خارجی با عامل خانگی آن کاربر تماس گرفته و آدرسی جدید از محل کاربر متحرک به آن اعلام می کند. این آدرس عموماً آدرس IP خود عامل خارجی است.

وقتی بسته ای به LAN محل استقرار دائمی کاربر می رسد قبل از انتشار بر روی LAN به مسیریاب متصل به آن LAN وارد می شود. آن مسیریاب به روش معمول و همیشگی سعی می کند ماشین آن کاربر را پیدا کند لذا یک بسته ARP منتشر کرده و مثلاً می پرسد: «آدرس اترنت ماشین 160.80.40.20 چند است؟». «عامل خانگی» به نیابت از کاربر، به این سؤال پاسخ داده و آدرس اترنت خود را اعلام می کند. از آن به بعد، مسیریاب تمام بسته های متعلق به 160.80.40.20 را برای «عامل خانگی» می فرستد. «عامل خانگی» با مکانیزم ایجاد تونل بسته اصلی را درون یک بسته IP جدید جاسازی کرده و آدرس مقصد بسته جدید را آدرس عامل خارجی قرار داده و آنرا ارسال می نماید. عامل خارجی پس از دریافت، بسته اصلی را از درون آن استخراج کرده و آنرا به آدرس MAC ماشین متحرک (یعنی آدرس سخت افزاری تعریف شده در لایه پیوند داده ها) ارسال می دارد. مضاف بر این، عامل خانگی کاربر، آدرس IP عامل خارجی را (که کاربر فعلاً مهمان اوست) به فرستنده بسته ها اعلام می کند تا بسته های بعدی به کمک مکانیزم تونل مستقیماً به آدرس عامل خارجی ارسال شوند. این راه حل تمام نیازها و اهداف فوق الذکر را برآورده می کند.

اشاره به برخی از جزئیات ارزشمند است: در لحظه ای که ماشین متحرک، محل خود را ترک می کند احتمالاً مسیریاب در جدول نهان ARP^۲ آدرس اترنت این ماشین را (که دیگر معتبر نیست) درج کرده و از آن استفاده می کند.^۳ برای جایگزین کردن آدرس اترنت «ماشین عامل» به جای آدرس ماشین متحرک از راهکاری زیرکانه به نام «Gratuitous ARP» استفاده می شود. این روش مبتنی بر ارسال یک پیام خاص و ناخواسته برای مسیریاب است که باعث می شود یک درایه دلخواه در جدول ARP به مقداری جدید تغییر کند. در اینجا درایه ای که باید تغییر کند متعلق به ماشین متحرک و در حال خروج از شبکه است. وقتی بعداً ماشین متحرک به شبکه خود باز می گردد از همین راهکار برای بهنگام سازی مجدد جدول ARP در مسیریاب، استفاده می شود.^۴

هیچ مانعی وجود ندارد که یک ماشین متحرک نتواند «عامل خارجی» خودش باشد ولی این راهکار زمانی عملی است که ماشین متحرک در موقعیت جدید خود، به اینترنت دسترسی داشته باشد. در ضمن ماشین متحرک باید بتواند در محل جدید یک آدرس IP موقت جهت اعلام به «عامل خانگی» خودش بدست بیاورد. این آدرس

۱. یعنی اگر سربرار کوچکی تحمیل می شود باید فقط زمانی باشد که ماشین میزبان در محل همیشگی خود نیست. -م

۲. ARP Cache

۳. یعنی چون پس از خروج ماشین متحرک، هنوز آدرس MAC آن در جدول ARP مسیریاب، وجود دارد تا موقعی که این آدرس اعتبار خود را حفظ می کند بسته ها به آدرسی که دیگر وجود ندارد ارسال خواهد شد. -م

۴. این راهکار آنست که «ماشین عامل» بدون آنکه هیچ سؤالی از او شده باشد یک بسته پاسخ ARP (یعنی ARP Response) به صورت مصنوعی تولید و ارسال می کند. بدین ترتیب مسیریاب به اشتباه افتاده و محتوای آنرا در جدول خود درج می کند، هر چند هرگز در این خصوص سؤالی نپرسیده بود! -م

IP متعلق به LAN جدیدی است که ماشین متحرک موقتاً بدان متصل شده و مهمان آنست. راه حل پیشنهادی IETF برای ماشینهای متحرک، مسائل متعدد دیگری را هم حل کرد که تا آن زمان بدان توجه نشده بود. به عنوان مثال یکی از مسائل این بود که ماشینهای عامل چگونه پیدا شوند؟ راه حل این مسئله آنست که هر عامل بطور متناوب آدرس خود و نوع سرویسهایی را که علاقمند به ارائه آنهاست، به صورت فراگیر منتشر نماید. وقتی ماشین متحرک به جایی وارد می شود ابتدا سعی می کند به این پیامهای انتشاری که اصطلاحاً «اعلان» (Advertisement) نامیده می شود، گوش فرا بدهد. گزینه دیگر آنست که ماشین متحرک ورود خود را با انتشار بسته ای فراگیر (Broadcast) اعلام کرده و در انتظار پاسخ ماشین «عامل خارجی» باقی بماند.

مشکل دیگری که باید حل شود آنست که با ماشینهای متحرکی که با عجله و بدون خداحافظی شبکه فعلی خود را ترک می کنند، چه باید کرد؟ راه حل آنست که حضور یک ماشین متحرک فقط برای مدت زمان ثابت و محدودی اعتبار داشته باشد. اگر ماشین متحرک بطور مرتب، اعتبار حضور خود را تمدید نکند و مهلت اعتبار او منقضی شود، عامل خارجی جداول خود را از مشخصات او پاک می نماید.

مشکل دیگر، موضوع امنیت است: وقتی یک عامل خانگی پیامی دریافت می کند که از او خواسته شده تمام بسته های متعلق به کاربری به نام روبرتا را به آدرس IP خاصی بفرستد، بهتر است این تقاضا پذیرفته نشود مگر آنکه اثبات شود که مبدا این درخواست، واقعاً روبرتا است نه کسی که خودش را به جای او جا زده است. بدین منظور از پروتکل های احراز هویت مبتنی بر رمزنگاری استفاده می شود. این پروتکلها را در فصل هشتم تشریح خواهیم کرد.

آخرین نکته ای که گروه کاری IETF بدان پرداخت در خصوص «سطوح تحرک» (Levels of Mobility) ماشینهاست. هواپیمایی را در نظر بگیرید که برای وصل کامپیوترهای مخصوص ناوبری و کنترل پرواز شبکه ای از نوع اترنت را بکار گرفته است. بر روی این شبکه یک مسیریاب استاندارد وجود دارد که از طریق یک لینک رادیویی با شبکه اینترنت مستقر بر روی زمین در ارتباط است. از طرفی این ایده در ذهن برخی از افراد زیرک جرقه زده که در کنار دسته صندلی هر مسافر یک کانکتور اترنت تعبیه شود تا مسافری بتواند در خلال پرواز، کامپیوترهای کیفی خود را به آن متصل کرده و به اینترنت وصل شوند. در این حالت باید برای کامپیوترهای متحرک دو سطح قائل شد: کامپیوترهای خود هواپیما، که نسبت به اترنت موجود در آن ثابت (Stationary) محسوب می شوند و کامپیوتر مسافران که نسبت به آن متحرک هستند. البته مسیریاب هواپیما نسبت به مسیریاب مستقر بر روی زمین، متحرک است. می توان متحرک بودن کامپیوترها نسبت به مسیریابهایی که خودشان متحرک هستند را از طریق ایجاد تونلهای متوالی پیاده سازی و اداره کرد.

۸-۶-۵ IPv6

اگرچه ممکن است مکانیزمهای CIDR و NAT چند سالی دیگر به دوام نسخه چهارم IP کمک کنند ولی تقریباً بر همه آشکار شده که نسخه های پروتکل IP در شکل کنونی آن، به شماره افتاده است. مضاف بر مشکلات فنی IP، برخی از موارد پشت صحنه و زمینه ای دیگر نیز مطرح است. در سالهای اولیه، از اینترنت عموماً در دانشگاهها، صنایع پیشرفته و دولت ایالات متحده (خصوصاً وزارت دفاع) استفاده می شد. با گرایش بسیار زیاد مردم به اینترنت که از اواسط دهه نود شروع شد، گروههای مختلفی از افراد به آن رو آوردند؛ افرادی که نیازها و انتظارات متفاوتی داشتند. یکی از موارد آنست که افراد با کامپیوترهای بی سیم قابل حمل برای در ارتباط بودن با محل استقرار دائمی خود (ایستگاههای خانگی) می خواهند از اینترنت بهره بگیرند. مورد دیگر آن که با همگرایی قریب الوقوع صنایع کامپیوتر و مخابرات و صنایع تولید بازی و ابزار تفریح، دیری نخواهد پایید که حتی دستگاههای تلفن و تلویزیون در دنیا، به عنوان گرهی از اینترنت، به آن خواهد پیوست و در آن زمان میلیاردها

ماشین، از صدا و تصویر بهره خواهند گرفت. با در نظر داشتن چنین چشم اندازی، IP بوضوح نیازمند تغییرات اساسی است و باید انعطاف بیشتری داشته باشد.

IETF که چنین افقی را پیش روی خود می دید در اوان ۱۹۹۰ کار را بر روی نسخه جدیدی از پروتکل IP شروع کرد که در آن فضای آدرس هرگز با کمبود مواجه نشود و مشکلات عدیده ای را حل کند؛ قابلیت انعطاف بیشتری داشته باشد و در ضمن کارآمدتر باشد. اهداف عمده IPv6 عبارت بودند از:

۱. پشتیبانی از میلیاردها ماشین میزبان حتی در صورتی که تخصیص فضای آدرس ناکارآمد و با اسراف انجام شود.
۲. کاهش اندازه جداول مسیریابی
۳. ساده سازی پروتکل به منظور افزایش سرعت پردازش مسیریابها
۴. ارائه امنیت بهتر در مقایسه با نسخه فعلی IP (شامل احراز هویت و سری ماندن داده ها)
۵. توجه بیشتر به نوع خدمات و QoS، به ویژه برای داده های بی درنگ
۶. کمک به فرآیند ارسال چندپختی از طریق توصیف حوزه ها (Scopes)
۷. فراهم آوردن امکان جابجایی ماشینهای میزبان بدون تغییر در آدرس
۸. امکان ایجاد تغییر و پیشرفت در آینده
۹. امکان همزیستی پروتکل های جدید و قدیم در طی سالها

برای توسعه پروتکلی که تمام نیازهای فوق الذکر را برآورده نماید، IETF با انتشار RFC 1550 و در طی یک فراخوان، خواستار پیشنهادات دیگران در این خصوص شد. ۲۱ پیشنهاد دریافت گردید که اغلب آنها جامع نبودند. تا دسامبر ۱۹۹۲ فقط هفت طرح پیشنهادی قابل توجه در دستور کار قرار داشت. این طرحهای پیشنهادی از اصلاحات جزئی در نسخه فعلی IP تا پیشنهاد دور انداختن آن و جایگزینی با یک پروتکل کاملاً متفاوت را شامل می شد.

یک پیشنهاد آن بود که TCP بر روی CLNP اجرا شود؛ پروتکلی که با آدرسهای ۱۶۰ بیتی فضای آدرس دهی نامحدود و جاویدان را فراهم کرده بود و دو پروتکل عمده و مهم لایه شبکه را متحد و یکپارچه می کرد. ولیکن بسیاری افراد احساس کردند که پذیرش آن مهر تأییدی است بر این ادعا که هر کاری که OSI انجام داده صحیح تلقی می شود، داعیه ای که لااقل در حوزه اینترنت به دلایل خاص نادرست است. الگوی CLNP بسیار شبیه به IP بود و این دو، تفاوت چندانی با هم ندارند. در آخر نیز طرحی انتخاب شد که تفاوت بسیار زیادی با IP و از آن بیشتر با CLNP دارد. ضربه دیگری که CLNP خورد از آنجا بود که پشتیبانی ضعیفی از «نوع خدمات» (Type of Service) می کرد، خصوصیتی که برای انتقال کارآمد داده های چند رسانه ای به شدت نیاز بود.

سه تا از بهترین طرحهای پیشنهادی در ژورنال IEEE Network منتشر شد.^۱ پس از مباحثات فراوان، بازبینی، ارزیابی موقعیت و سنجش استقبال عمومی، نسخه ای ترکیبی از طرحهای پیشنهادی Deering و Francis که SIPP^۲ نامیده می شد به عنوان طرح برگزیده معرفی و با عنوان IPv6 معرفی گردید.

IPv6 بخوبی اهداف مورد نظر را برآورده می کند: ویژگیهای خوب IP را نگه داشته، ویژگیهای بد را کنار گذاشته یا کمرنگ کرده و ویژگیهای جدیدی به آن افزوده است. بطور کلی IPv6 با IPv4 سازگار نیست ولی با تمام پروتکل های جانبی اینترنت مثل TCP، UDP، ICMP، IGMP، OSPF، BGP و DNS سازگار است. (البته

۱. Deering, 1993; Francis, 1993; and Katz and Ford, 1993

۲. Simple Internet Protocol Plus

ممکن است به دلیل آنکه آدرسها طولانی‌تر شده‌اند نیاز به اندکی تغییر داشته باشند.) ویژگیهای اساسی IPv6 در زیر تشریح شده است. برای آگاهی بیشتر در خصوص آن به RFCهای 2460 تا 2466 مراجعه نمایید.

اولین و مهمترین ویژگی آنست که IPv6 آدرسهای بسیار طولانی‌تری نسبت به IPv4 دارد. این آدرسها ۱۶ بایت طول دارند و دقیقاً مشکلی را حل کرده که به همان دلیل طراحی شد: یعنی تقریباً فضای نامحدودی از آدرسهای IP را فراهم آورده است. در این خصوص بیشتر صحبت خواهیم کرد.

دومین پیشرفت عمده IPv6 ساده‌سازی سرآیند آنست. این سرآیند جمعاً هفت فیلد دارد (در مقابل سیزده فیلد در IPv4). این تغییر، امکان آنرا فراهم آورده که مسیر یاب بسته‌ها را سریعتر پردازش نماید و ظرفیت مفید مسیر یاب را افزایش و تأخیر را کاهش دهد. در ادامه مختصراً به سرآیند خواهیم پرداخت.

سومین بهبود عمده آن پشتیبانی از گزینه‌های اختیاری (Options) است. این تغییر برای سرآیند جدید حیاتی بود چراکه برخی از فیلدهایی که در نسخه قبلی وجودشان الزامی است در نسخه فعلی اختیاریند. مضاف بر این، روش درج گزینه‌ها در این فیلد متفاوت از قبل است و اجازه می‌دهد مسیر یابها بتوانند به سادگی از گزینه‌هایی که برایشان مهم نیست رد شوند. [یعنی در نسخه جدید، دسترسی تصادفی و مستقیم به گزینه‌ها ممکن شده است.] این ویژگی سرعت پردازش بسته‌ها را افزایش می‌دهد.

چهارمین موضوعی که IPv6 در آن پیشرفت عمده‌ای داشته است، «امنیت» است. IETF با انبوهی از گزارشهای مطبوعاتی در خصوص نایب‌های دوازده ساله‌ای مواجه بود که با کامپیوترهای شخصی خود، به شبکه‌های بانکی یا پایگاههای نظامی در سراسر اینترنت، نفوذ کرده بودند و جوّ شدیدی براه افتاده بود که باید برای بهبود امنیت شبکه‌ها کاری کرد. در نسخه جدید IP، «احراز هویت» (Authentication) و «حفظ امنیت اطلاعات» (Privacy) جزو ویژگیهای کلیدی به شمار می‌رود. البته این ویژگیها بعداً به IPv4 نیز افزوده شد [با عنوان IPsec]. فلذا در حال حاضر این دو، در زمینه امنیت تفاوت چندانی با هم ندارند.

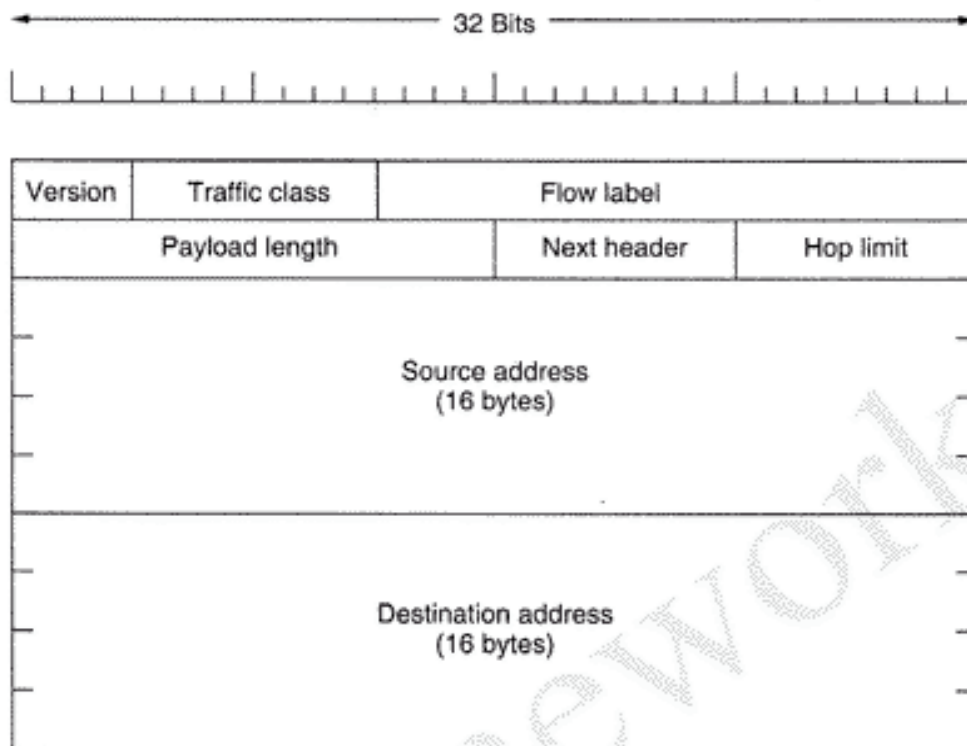
موضوع آخر آنکه در نسخه جدید به «کیفیت خدمات» (QoS) دقت بیشتری شده است. قبل از آن نیز تلاشهای جسته و گریخته‌ای در این رابطه انجام شده بود ولیکن با رشد کاربردهای چند رسانه‌ای در اینترنت، این موضوع جدی‌تر و حساس‌تر به نظر می‌رسید.

سرآیند اصلی IPv6

سرآیند اصلی IPv6 در شکل ۵-۶۸ نشان داده شده است. فیلد Version (شماره نسخه پروتکل) برای IPv6 همیشه ۶ است. (کما اینکه برای IPv4 نیز همیشه ۴ است!). در دوران گذار از IPv4 به نسخه جدید که ممکن است یک دهه طول بکشد، مسیر یابها قادرند با بررسی این فیلد تشخیص بدهند که با چه نوع بسته‌ای روبرو هستند. البته از آنجایی که بررسی این فیلد به چندین دستورالعمل اجرایی CPU نیاز دارد و این کار زمان مفید پردازش هر بسته را هدر می‌دهد لذا در بسیاری از پیاده‌سازیهای عملی، برای اجتناب از این زمان تلفاتی، تشخیص آنکه یک بسته از نوع IPv4 است یا IPv6، با استفاده از فیلد خاصی در سرآیند لایه پیوند داده‌ها بر عهده سخت‌افزار گذاشته شده است.^۱ بدین ترتیب بسته‌ها براساس نوعشان مستقیماً به نرم‌افزار مناسب در لایه شبکه هدایت می‌شوند. البته این الزام که لایه پیوند داده از جزئیات نوع بسته‌های لایه شبکه آگاه باشد با این اصل اساسی که «هر لایه نباید از معنای پشتهایی که از لایه بالاتر تحویل او می‌شود، آگاه باشد» در تناقض است. بدون شک بحث و مناقشه بین طرفداران ایده‌های «انجام اصولگرایانه و صحیح کار» و «تسریع کار» به شدت ادامه خواهد داشت.

فیلد Traffic Class (کلاس ترافیک) برای تشخیص تفاوت بسته‌ها از لحاظ نیازمندیهای تحویل بی‌درنگ و

۱. مثلاً در فریم انترنت فیلد Type نوع بسته درون فریم را تعیین می‌کند. -



شکل ۵-۶۸. سرآیند ثابت و الزامی در IPv6.

QoS درخواستی، بکار می‌آید. فیلدی با همین منظور از ابتدا در IP وجود داشت ولیکن استفاده از آن به صورت پراکنده و سلیقه‌ای بر روی مسیریابها پیاده‌سازی شد و اغلب مسیریابها آن را نادیده می‌گرفتند. اکنون تجربیات گذشته چراغ راهی شده تا بتوان بهترین راه و روش تحویل بسته‌های اطلاعات چندرسانه‌ای را تعیین کرد. فیلد Flow Label (برچسب جریان) همچنان آزمایشی است ولی کاربرد مورد نظر آن، این بوده که بتوان یک «شبه اتصال» (Pseudoconnection) بین مبدا و مقصد، با ویژگیها و نیازمندیهای خاص ایجاد کرد. به عنوان مثال، جریانی از بسته‌ها که از یک پروسه در مبدا خاص تولید و به سوی یک پروسه بر روی مقصد خاص روانه می‌شوند احتمالاً نیاز به تضمین تأخیر محدود و مشخص دارد و در نتیجه باید پهنای باند لازم را رزرو کرد. در چنین مواردی می‌توان پیشاپیش یک «جریان» (Flow) با مشخصات درخواستی تنظیم کرد و به آن یک شناسه اختصاص داد. هر گاه مسیریاب بسته‌ای دریافت کند و فیلد Flow Label آن غیر صفر باشد، با مراجعه به جداول درونی خود تشخیص می‌دهد که با این بسته چگونه رفتار کند. در حقیقت استفاده از مفهوم «جریان»^۱ در IPv6، تلاشی است برای رسیدن به قابلیت انعطاف در زیر شبکه‌های دیتاگرام و تضمین کیفیت خدمات در زیر شبکه‌های مدار مجازی.

هویت هر «جریان» برحسب آدرس مبدا، آدرس مقصد و شماره جریان (برچسب جریان) مشخص می‌شود. فلذا بین دو مبدا و مقصد در شبکه می‌توان بطور همزمان چندین «جریان» فعال تنظیم کرد. همچنین در این روش حتی اگر دو جریان متفاوت با شماره جریان یکسان از دو ماشین میزبان مختلف تولید و از مسیریابهای مشابهی عبور کنند، مسیریابها به کمک آدرس مبدا و مقصد قادر به تشخیص آنها خواهند بود. انتظار آنست که «برچسبهای جریان» به جای آنکه به صورت ترتیبی و از ۱ شروع شوند به صورت کاملاً تصادفی انتخاب گردند تا مسیریاب

۱. برای آشنایی با مفهوم جریان رجوع کنید به بخش ۵-۴-۱.

بتواند آنها را در Hash Table خود درج کند.^۱

فیلد Payload Length (طول قسمت حمل داده) مشخص می‌کند که پس از سرآیند ۴۰ بایتی در شکل ۵-۶۸ چند بایت داده قرار گرفته است. همین فیلد در IPv4 با نام Total Length وجود داشت. تغییر نام به آن دلیل بوده که در نسخه جدید، سرآیند جزو طول بسته به حساب نمی‌آید بلکه فقط اندازه بخش حمل داده تعیین می‌شود. فیلد Next Header ساختار بسته را سبکبار کرده است! دلیل آنکه سرآیند بسته ساده شده آنست که می‌توان در صورت لزوم سرآیند اضافی و انتخابی داشت. این فیلد مشخص می‌کند که پس از سرآیند ۴۰ بایتی کدامیک از سرآیندهای ششگانه اضافی قرار گرفته است (در صورت وجود). اگر سرآیند اخیر، آخرین سرآیند بسته IP باشد، این فیلد مشخص می‌کند که کدام پروتکل در لایه انتقال محتوای بسته را تحویل خواهد گرفت (مثلاً TCP، UDP و نظائر آن).^۲

کاربرد فیلد Hop Limit آنست که بسته‌ها عمر محدودی داشته باشند. این فیلد در عمل مشابه با فیلد Time to Live (زمان حیات بسته) در IPv4 است یعنی به ازای عبور بسته از یک مسیر یاب، یک واحد از مقدار آن کاسته می‌شود. در تئوری، مبنایی که IPv4 برای این فیلد در نظر داشت، «زمان بر مبنای ثانیه» بود در حالی که هیچ مسیریابی از چنین مبنایی استفاده نمی‌کند (بلکه به ازای هر گام یک واحد از آن می‌کاهد) لذا نام این فیلد را به گونه‌ای عوض کردند که عملکرد واقعی آن را نشان بدهد. هرگاه مقدار این فیلد در یک بسته به صفر برسد آن بسته حذف خواهد شد.

در ادامه فیلدهای Source Address و Destination Address (آدرس مبدا و مقصد) قرار گرفته‌اند. در طرح پیشنهادی آقای Deering ۸ بایتی انتخاب شده بودند در حالی که در مراحل بازمینی دیگران احساس کردند که شاید این فضای آدرس نیز در خلال چند دهه، IPv6 را نیز با کمبود فضای آدرس مواجه کند، در حالی که با آدرسهای ۱۶ بایتی هرگز چنین کمبودی رخ نخواهد داد. برخی از افراد معتقد بودند که آدرسهای ۱۶ بایتی بیش از حد بزرگ هستند در حالی برخی دیگر اعتقاد داشتند باید از آدرسهای ۲۰ بایتی استفاده شود تا با پروتکل دیتاگرام پیشنهادی OSI سازگار باشد. گروه دیگری نیز به آدرسهای با طول متغیر گرایش داشتند. پس از بحث و جدل فراوان، به این نتیجه رسیدند که آدرسهای با طول ثابت ۱۶ بایتی بهترین انتخاب است. با توجه به طول زیاد آدرسهای IP، نماد جدیدی برای نوشتن آنها پیشنهاد شد. این آدرسها به صورت هشت گروه که با علامت :: از هم جدا شده، نوشته می‌شوند. هر گروه نیز به صورت چهار رقم هگزادسیمال نمایش داده می‌شود:

8000:0000:0000:0000:0123:4567:89AB:CDEF

از آنجایی که در آدرسها، تعداد ارقام صفر زیاد است، سه نوع بهینه‌سازی مجاز شمرده شده: صفرهای سمت چپ در هر گروه نوشته نمی‌شوند یعنی 0123 به صورت 123 نشان داده می‌شود؛ دوم آنکه اگر یک یا چند گروه شانزده بیتی تماماً صفر باشند با یک زوج علامت :: نشان داده می‌شود. بنابراین آدرس مثال بالا به صورت زیر نوشته خواهد شد:

8000::123:4567:89AB:CDEF

نهایتاً آنکه آدرسهای IPv4 را می‌توان با یک جفت :: و سپس آدرس نقطه‌دار قدیمی، نشان داد:

::192.31.20.46

۱. به عبارت دیگر مسیریاب انتظار دارد این شماره‌ها را Hash کند لذا این شماره‌ها نباید متوالی باشند.

۲. به عبارت دیگر محتوای این فیلد به صورت بازگشتی سرآیندهای بعدی را مشخص می‌کند تا نهایتاً به سرآیند آخر برسد که نوع بسته لایه انتقال را تعیین می‌نماید. -م

شاید لازم به گفتن نباشد که آدرسهای شانزده بایتی، فضایی معادل 2^{128} آدرس هستند که چنین فضایی تقریباً معادل 3×10^{38} آدرس است. اگر کل کره زمین شامل خشکیها و دریاها پر از کامپیوتر شوند باز هم IPv6 می تواند برای هر مترمربع 7×10^{23} آدرس IP فراهم کند. دانشجویان رشته شیمی می دانند که این عدد حتی از عدد آووگادرو نیز بزرگ است. [عدد آووگادرو 6.02×10^{23} است]. چون در نظر نبوده که حتی به مولکولهای سطح زمین آدرس بدهیم آدرسهای IP شانزده بایتی، بهیچوجه کم نخواهد آمد!!

در عمل از فضای آدرس IP، بخوبی استفاده نخواهد شد. (دقیقاً همانند فضای شماره های تلفن که مثلاً فضای شماره های تلفن منتهن با پیش شماره ۲۱۲ پر شده ولی فضای شماره های ویومینگ (Wyoming) با پیش شماره ۳۰۷ تقریباً خالی مانده است.) دو نفر به نامهای Huitema و Durand در سند RFC 3194 محاسبه کرده اند که با ایده گرفتن از تخصیص شماره های تلفن و حتی در بدبینانه ترین حالت ممکن، باز هم می توان برای هر مترمربع از کره زمین، 10^{50} آدرس IP کنار گذاشت. در حالت کلی نیز می توان تریلیونها آدرس IP برای هر مترمربع از زمین در نظر گرفت. کوتاه سخن آنکه، در آینده هیچگاه به مشکل فضای آدرس بر نخواهیم خورد.

مقایسه سرآیند IPv4 (شکل ۵-۵۳) با سرآیند IPv6 (شکل ۵-۶۸) از این دیدگاه که چه فیلدی و چرا حذف شده است، آموزنده خواهد بود: فیلد IHL حذف شده زیرا سرآیندهای IPv6 طول ثابتی دارد. فیلد «پروتکل» وجود ندارد چرا که فیلد Next Header مشخص می کند که پس از آخرین سرآیند چه بسته دیگری آمده است (بسته TCP، UDP یا نظائر آن).

تمام فیلدهایی که در ارتباط با «قطعه قطعه سازی» بسته ها در IPv4 تعریف شده بود در IPv6 حذف گردیده است زیرا پروتکل اخیر راهکار دیگری برای مکانیزم قطعه قطعه سازی برگزیده است. انتظار آنست که تمام ماشینهای سازگار با IPv6 بتوانند به صورت خودکار و پویا اندازه دیتاگرامها را تعیین کنند و بدین نحو نیاز به قطعه قطعه شدن بسته ها کمتر اتفاق می افتد. همچنین حداقل طول بسته ای که هر ماشین موظف به پذیرش آنست از ۵۷۶ بایت به ۱۲۸۰ بایت افزایش یافته تا بتوان یک قطعه داده 1024 بایتی را به همراه تعداد زیادی سرآیند (۲۵۶ بایت)، بدون نیاز به قطعه قطعه شدن ارسال و دریافت کرد. مضاف بر این، وقتی ماشین یک بسته بیش از حد بزرگ IPv6 را ارسال می دارد مسیر یاب ناتوان از هدایت آن، به جای قطعه قطعه کردن بسته آن را حذف کرده و پیام خطایی را باز می گرداند. این پیام به ماشین میزبان تفهیم می کند که باید بسته هایش را بشکند. البته اگر ماشین میزبان خودش بسته ها را با اندازه مناسب ارسال کند کارآمدتر از آنست که بسته ها در طول مسیر شکسته شود.

فیلد Checksum نیز حذف شد زیرا محاسبه آن کارایی و سرعت پردازش بسته ها را به نحو چشمگیری کاهش خواهد داد. با توجه به قابلیت اعتماد شبکه های کنونی و با در نظر داشتن این حقیقت که در لایه پیوند داده و لایه انتقال نیز (بطور مجزا) صحت داده ها بررسی می شود، محاسبه یک کد کشف خطای دیگر مثل checksum در مقایسه با کاهش کارایی ارزشی ندارد. حذف این ویژگیهای زائد، IPv6 را به پروتکل متعادل و جمع و جور تبدیل کرده است. بدین ترتیب IPv6 به اهداف مورد نظر خود که همانا انعطاف، سرعت و فضای بزرگ آدرس بوده، نائل شده است.

سرآیندهای اضافی (سرآیندهای توسعه یا Extension Header)

گاهی به برخی از فیلدهای حذف شده IPv4 نیاز می شود و به همین منظور در IPv6 مفهوم جدیدی به نام «سرآیندهای توسعه» معرفی شده است. این سرآیندهای اختیاری برای افزودن اطلاعات به هر بسته بکار می آیند ولیکن روش کدینگ (و جاسازی) آنها کارآمد و سریع است. شش نوع مختلف سرآیند توسعه که تاکنون معرفی شده، در شکل ۵-۶۹ فهرست گردیده است. هر کدام از این سرآیندها اختیاریند ولیکن اگر به بیش از یک سرآیند نیاز باشد باید بطور پیاپی، پس از سرآیند ثابت و ترجیحاً به ترتیب فهرست، قرار بگیرند.

توصیف عملکرد	نام سرآیند توسیع (سرآیند اضافی)
حاوی اطلاعات گوناگون برای مسیریابی	Hop-by-hop options
اطلاعات اضافی برای مقصد	Destination options
فهرست ناکاملی از مسیریابها که بسته باید از آنها بگذرد.	Routing
مدیریت قطعات دیتاگرام	Fragmentation
بررسی هویت فرستنده	Authentication
اطلاعاتی در خصوص محتوای رمزنگاری شده بسته	Encrypted security payload

شکل ۵-۶۹. سرآیندهای توسیع در IPv6 (Extension Header).

برخی از سرآیندها دارای قالب ثابتی هستند در حالی که برخی دیگر تعداد متغیری فیلد با طول متفاوت دارند. به همین دلیل هر آیتم در قالب سه تایی (نوع، طول، مقدار)^۱ سازماندهی و کُد می شود. فیلد Type مشخص می کند که نوع گزینه چیست. مقدار فیلد نوع (Type) به نحوی انتخاب شده است که دو بیت ابتدایی آن به مسیریابهایی که نمی دانند آن گزینه را چگونه پردازش کنند، راه و چگونگی کار را نشان می دهند. این راهکارها عبارتند از: (۱) گزینه مربوط را نادیده بگیر (۲) بسته را حذف کن (۳) بسته را حذف و یک بسته ICMP برگردان (۴) بسته را حذف کن و یک بسته ICMP برگردان ولیکن بسته ICMP را برای آدرسهای «چندپخشی» (Multicast) نفرست. (تا یک بسته چندپخشی اشتباه، منجر به تولید میلیونها گزارش ICMP نشود). فیلد یک بایتی طول (Length) مشخص می کند که فیلد مقدار (Value) چند بایتی است. (صفر تا ۲۵۵ بایت). فیلد مقدار (Value) در برگیرنده اطلاعات مورد نیاز است و حداکثر می تواند ۲۵۵ بایت باشد. «سرآیند توسیع Hop-by-Hop» (گام به گام) برای حمل اطلاعاتی کاربرد دارد که مسیریابهای واقع بر مسیر باید آنها را بررسی نمایند. قبلاً یکی از گزینه ها را معرفی کردیم: پشتیبانی از دیتاگرامهایی با طول بیش از ۶۴ کیلوبایت. قالب این سرآیند در شکل ۵-۷۰ نشان داده شده است. وقتی از این سرآیند استفاده می شود باید فیلد Payload Length (در سرآیند اصلی) به صفر مقداردهی شود.

Next header	0	194	4
Jumbo payload length			

شکل ۵-۷۰. «سرآیند توسیع Hop-by-Hop» (گام به گام) برای دیتاگرامهای بسیار بزرگ (جامبوگرام).

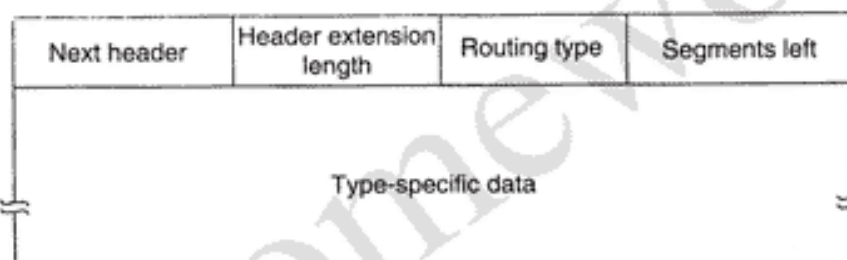
همانند تمام سرآیندهای اختیاری دیگر، این سرآیند نیز با فیلدی یک بایتی به نام Next Header شروع می شود و مشخص می کند که سرآیند بعدی از چه نوع است. پس از این بایت، بایت دیگری قرار گرفته که طول سرآیند Hop-by-Hop را بر مبنای بایت تعیین می کند ولیکن در مقدار آن، ۸ بایت ابتدایی (که وجود آن الزامی است) لحاظ نمی شود. تمام سرآیندهای توسیع دیگر نیز به همین نحو شروع می شوند. در ادامه فیلدی دو بایتی آمده که بایت اول مشخص می کند که این گزینه (Option) قرار است اندازه دیتاگرام را تعریف کند (کد ۱۹۴) و بایت بعدی مشخص می کند که اندازه دیتاگرام یک شماره چهار بایتی است. چهار بایت آخر این سرآیند، طول دیتاگرام را مشخص می کند. اندازه زیر ۶۵۵۳۶ مجاز نیست و منجر به حذف بسته در اولین مسیریاب و بازگشت پیام خطای ICMP خواهد شد. دیتاگرامهایی که از این سرآیند اختیاری (یعنی Hop-by-Hop Header) استفاده

^۱. (Type, Length, Value)

کرده اند اصطلاحاً Jumbogram (دیتاگرام عظیم) نامیده می شوند. [بدین ترتیب در IPv6 می توان قطعات داده بسیار بزرگ را به کمک سرآیند فوق به صورت یکجا ارسال کرد.] کاربرد جامبوگرامها در سوپر کامپیوترها که باید چندین گیگابایت اطلاعات را از طریق اینترنت منتقل کنند، بسیار حیاتی است.

«سرآیند توسیع Destination Options» برای درج فیلدهایی در نظر گرفته شده که صرفاً توسط ماشین مقصد پردازش و تفسیر می شوند. در نسخه اولیه IPv6، مقدار این گزینه پوچ در نظر گرفته شده و کاربردی نداشته است. وجود چنین فیلدی برای آن بوده که نرم افزار ماشینهای میزبان و مسیر یابها چنین سرآیندی را به رسمیت بشناسند تا اگر روزگاری به آن نیاز شد، شرایط مهیا باشد وگرنه باید پروتکل عوض شود.

«سرآیند توسیع Routing» فهرست مسیر یابهایی را مشخص می نماید که بسته باید در راه رسیدن به مقصد از آنها عبور کند. این گزینه شباهت زیادی به گزینه Loose Source Routing در IPv4 دارد. فهرست آدرسهای که در این سرآیند مشخص شده باید در طول مسیر و به ترتیب ملاقات شوند ولی این امکان وجود دارد که مسیر یابهایی هم که آدرس آنها در فهرست نیست مابین مسیر باشند. قالب سرآیند Routing در شکل ۷۱-۵ مشخص شده است.



شکل ۷۱-۵. سرآیند توسیع برای مسیر یابی (Routing).

چهار بایت اول از این سرآیند، شامل چهار فیلد یک بایتی است: دو فیلد Next Header و Header Extension Length را قبلاً تعریف کردیم. فیلد Routing Type، ساختار مابقی سرآیند را مشخص می نماید: مقدار صفر مشخص کننده آنست که پس از کلمه چهار بایتی اول، یک کلمه چهار بایتی دیگر قرار گرفته و پس از آن آدرس IPv6 (یعنی آدرسهای ۱۲۸ بیتی) مسیر یابها قرار می گیرد. به غیر از این ساختار، فعلاً ساختار دیگری تعریف نشده مگر آنکه در آینده چیز جدیدی ابداع شود. فیلد آخر یعنی Segment Left تعداد آدرسهای را مشخص می کند که هنوز ملاقات نشده اند. مقدار اولیه این فیلد معادل با تعداد مسیر یابهایی است که آدرس آنها در فهرست مورد نظر درج شده است و به ازای ملاقات در هر مسیر یاب که آدرس آن در فهرست آمده یک واحد از این فیلد کاسته می شود. وقتی مقدار این فیلد در بسته به صفر برسد بسته روال طبیعی طی مسیر خود را از سر می گیرد بدون آنکه اجبار به عبور از مسیر خاصی داشته باشد. معمولاً در چنین لحظه ای بسته به مقصد خود نزدیک شده است.

سرآیند Fragmentation مشابه با IPv4، با مسئله قطعه قطعه سازی بسته ها سر و کار دارد. در این سرآیند نیز فیلدهای «شماره شناسایی دیتاگرام»، «شماره قطعه» و یک بیت MF تعریف شده اند که بیت MF مشخص می کند که آیا قطعه جاری آخرین قطعه دیتاگرام است یا آنکه قطعات دیگری در ادامه وجود دارند. البته در IPv6 برخلاف IPv4، فقط ماشین مبدا می تواند بسته ای را قطعه قطعه کند و مسیر یابهای واقع بر روی مسیر قادر به چنین کاری نیستند. اگرچه این موضوع از لحاظ فلسفی یک واپسگرایی محسوب می شود ولی در عوض کار مسیر یابها را ساده تر کرده و فرآیند مسیر یابی سریعتر خواهد شد. همانگونه که قبلاً اشاره کردیم هر گاه یک مسیر یاب با بسته ای

بیش از حد بزرگ مواجه گردد آن را حذف کرده و بسته ICMP (حامل پیغام خطا و اطلاعات مفید دیگر) به مبدا آن برمی‌گرداند. اطلاعات ارسالی به مبدا بسته، امکان آنرا می‌دهد که به کمک این سرآیند، بسته را به قطعات کوچکتر تقسیم و آنها را از نو ارسال کند.

سرآیند Authentication (سرآیند احراز هویت) مکانیزمی را فراهم آورده تا گیرنده بتواند از هویت فرستنده بسته مطمئن شود. سرآیند Encrypted Security Payload اجازه می‌دهد تا محتوای بسته رمزنگاری شود و بدین ترتیب فقط گیرنده مورد نظر قادر به خواندن آنست. این گونه سرآیندها برای انجام مأموریت خود از تکنیکهای رمزنگاری بهره می‌گیرند.

اختلاف نظرها و مناقشات

نظر به آنکه فرآیند طراحی IPv6، «باز» بوده و افراد درگیر در طراحی، بر عقاید خود تأکید داشته‌اند فلذا شگفت‌آور نیست که بسیاری از گزینه‌های انتخابی در IPv6 متناقض باشند. در زیر اجمالاً برخی از آنها را بررسی خواهیم کرد. برای آگاهی از جزئیات ماجراییه RFCهای مربوطه مراجعه نمایید.

قبلاً اشاره کردیم که بحث و جدل گسترده‌ای پیرامون طول آدرسها وجود داشت و توافق نهایی آن بود که آدرسها با طول ثابت و ۱۶ بیتی باشند.

جدل دیگری بر سر حداکثر تعداد گام (Hop Limit) در گرفت. یک گروه احساس می‌کرد که محدود کردن حداکثر تعداد گام (Hop) به ۲۵۵ یک اشتباه محض است چرا که اگرچه در آن زمان حداکثر طول مسیرها عموماً از ۳۲ تجاوز نمی‌کرد ولی مدعی بودند که ممکن است ده سال بعد مسیرها طولانی‌تر از ۲۵۵ باشند. استدلال آنها این بود که فضای آدرس ۱۶ بیتی آینده‌نگری بیش از اندازه و در عوض مقدار کم Hop Count، کوتاه‌نظری است. از دیدگاه آنها بزرگترین اشتباه یک دانشمند کامپیوتر، آنست که برای هر فیلدی، تعداد بیت کمی در نظر بگیرد.

پاسخ گروه مقابل آن بود که افزایش بی‌مورد فضای هر فیلد منجر به تشکیل یک سرآیند حجیم خواهد شد. همچنین استدلال دیگرشان آن بود که وظیفه فیلد Hop Count جلوگیری از سرگردانی بسته‌ها به مدت طولانی است و ۶۵۵۳۵ گام بیش از حد زیاد است. استدلال آخر آنکه با رشد اینترنت، لینکهای بسیار طولانی ساخته می‌شوند و این امکان فراهم می‌شود که برای رسیدن از یک کشور به کشور دیگر به کمتر از ده گام نیاز باشد. اگر یک بسته برای رسیدن از مبدا به مقصد مجبور شود از ۱۲۵ مسیر یاب بین‌المللی بگذرد، ستون فقرات این شبکه بین‌المللی در جایی اشکال دارد! بدین ترتیب طرفداران فیلد ۸ بیتی در عقیده خود پیروز شدند.

یکی دیگر از بحثهای داغ بر سر حداکثر طول بسته‌ها بود. سوپرکامپیوترها به بسته‌هایی با طول بیش از ۶۴ کیلوبایت احتیاج داشتند. وقتی یک سوپرکامپیوتر شروع به ارسال می‌کند و به کار خود مشغول می‌شود نباید به ازای هر ۶۴ کیلوبایت یکبار متوقف شود. استدلال گروه مخالف آن بود که اگر یک بسته یک مگابایتی در طول مسیر به یک خط T1 برسد آن خط به مدت حداقل ۵ ثانیه اشغال شده و کاربران دیگری که در این خط سهیم هستند با تأخیر قابل توجهی روبرو خواهند شد.^۱ توافق نهایی بدینجا ختم شد که بسته‌های معمولی حداکثر ۶۴ کیلوبایتی باشند ولی به کمک سرآیند اختیاری Hop-by-Hop بتوان جامبوگرامهایی با هر طول دلخواه ارسال کرد. موضوع سوم مناقشه، حذف فیلد IPv4 checksum (کد تشخیص خطاهای احتمالی در سرآیند) بود. برخی از افراد حذف این فیلد را مشابه با برداشتن ترمزهای یک خودرو می‌دانستند که اگرچه ماشین را سبکبار و سریعتر می‌کند ولی اگر اتفاق غیر مترقبه‌ای رخ بدهد مشکل جدی بوجود می‌آید. لذا، گروه مقابل آن بود که هر برنامه کاربردی که نگران صحت داده‌های خود است باید از پروتکلی در لایه انتقال بهره بگیرد که داده‌ها را از لحاظ

۱. نرخ ارسال خط T1 را 1.544 مگابیت در نظر بگیرید.

سلامت بررسی می‌کند. لذا اضافه کردن کد کنترلی دیگر به لایه IP برای کشف خطا (در حالی که هر بسته یکبار هم در لایه پیوند داده بررسی می‌شود) بیهوده و زائد است. مضاف بر آن، تجربه نشان داده بود که محاسبه جمع کنترلی (Checksum) در IPv4 هزینه بالایی دارد. در این مناقشه نیز طرفداران حذف کد کشف خطا پیروز شدند.

موضوع دیگر، بحث ماشینهای همراه بود: وقتی یک کامپیوتر قابل حمل، در نیمی از کل دنیا حرکت می‌کند (مثلاً درون هواپیما)، آیا می‌تواند با همان آدرس IPv6 قبلی، کار خود را ادامه بدهد یا آنکه مجبور به استفاده از ساختار «عامل خانگی» و «عامل خارجی» است؟ ماشینهای همراه مشکل «عدم تقارن» را به سیستم مسیریابی تحمیل می‌کنند: یک کامپیوتر همراه و کوچک براحتی قادر به شنیدن سیگنال قوی متشتره از مسیریاب ثابت خود هست ولی مسیریاب ثابت براحتی قادر به احساس سیگنال ضعیف ارسال شده توسط کامپیوتر همراه نیست. در نتیجه برخی از افراد گرایش داشتند که در IPv6 از ماشینهای همراه حمایت شود ولی تمام این تلاشها به دلیل آنکه بر روی هیچیک از طرحهای پیشنهادی توافقی بدست نیامد، با شکست مواجه گردید.

شاید بزرگترین مناقشه بر سر موضوع «امنیت» بود: همه بر این اصل که «امنیت لازم است» اشتراک داشتند. دعوا بر سر چگونگی رسیدن به امنیت و محل پرداختن به آن بود. اولین محل پرداختن به امنیت لایه شبکه است. استدلال موافقین مبنی بر آن بود که پیاده‌سازی امنیت در لایه شبکه، سرویسی استاندارد فراهم می‌کند که تمام برنامه‌های کاربردی بدون هیچگونه برنامه‌ریزی قبلی می‌توانند از آنها بهره بگیرند. استدلال مخالفین نیز آن بود که برنامه‌های کاربردی امن، عموماً به هیچ مکانیزمی کمتر از رمزنگاری انتها به انتها (End-to-End Encryption) احتیاج ندارند، به نحوی که پروسه مبدا خودش داده‌های ارسالی خود را رمز کرده و پروسه مقصد آنها را از رمز خارج کند. هر چیزی کمتر از این، می‌تواند کاربر را با خطراتی مواجه کند که از اشکالات امنیتی لایه شبکه ناشی می‌شود و هیچ تقصیری از او نیست. پاسخ به این استدلال آن بود که کاربر می‌تواند امنیت لایه IP را نادیده بگیرد و کار خودش را انجام بدهد! پاسخ نهایی مخالفین نیز آن بود که افرادی که به عملکرد صحیح شبکه (در خصوص امنیت) اعتماد ندارند چرا باید هزینه پیاده‌سازی سنگین و کندی IP را بپردازند!!

یکی دیگر از جنبه‌های مربوط به امنیت این حقیقت بود که بسیاری از کشورها قوانین سخت‌گیرانه‌ای در مورد صادرات محصولات مرتبط با رمزنگاری وضع کرده‌اند. از مثالهای بارز می‌توان به فرانسه و عراق اشاره کرد که حتی استفاده از رمزنگاری در داخل را نیز محدود کرده‌اند و عموم افراد نمی‌توانند چیزی را از پلیس مخفی نگه دارند. در نتیجه هر گونه پیاده‌سازی از IP که از روشهای رمزنگاری قوی استفاده می‌کند مجوز صدور از ایالات متحده (و بسیاری از کشورهای دیگر) را نخواهد گرفت. پیاده‌سازی دو نرم‌افزار یکی برای کاربرد داخلی و یکی برای صادرات، موضوعی است که عرضه‌کنندگان صنعت کامپیوتر با آن مخالفند.

موضوعی که پیرامون آن هیچ اختلاف نظر پیش نیامد آن بود که نمی‌توان انتظار داشت صبح روز یکشنبه IPv4 را در اینترنت از کار انداخت و صبح دوشنبه IPv6 را روشن نمود. در عوض مسیریابها و ماشینهایی که به IPv6 مجهز می‌شوند به مثابه جزایر مستقل با استفاده از تونل با یکدیگر مبادله داده می‌کنند و با افزایش این جزایر، در یکدیگر ادغام شده و جزیره بزرگتری پدید می‌آید. در نهایت تمام این جزایر به هم می‌پیوندند و اینترنت کاملاً متحول می‌شود.

سرمایه‌گذاری حجیمی که از قبل بر روی مسیریابهای مبتنی بر IPv4 صورت گرفته، فرآیند تغییر و تحول اینترنت را سالها به تأخیر می‌اندازد. به همین دلیل تلاش زیادی صورت می‌گیرد تا این اطمینان حاصل شود که گذار از IPv4 به IPv6 حتی الامکان بدون زحمت و گرفتن انجام گیرد. برای کسب آگاهی بیشتر در خصوص IPv6 به مرجع (Loshin, 1999) مراجعه نمایید.

۷-۵ خلاصه

لایه شبکه به لایه انتقال خدماتی را عرضه می کند. این لایه می تواند مبتنی بر «مدارات مجازی» یا «دیتاگرام» باشد. در هر دو حالت، وظیفه اصلی این لایه آنست که بسته های داده را از مبدا به مقصد برساند. در «زیرشبکه های مدار مجازی»، تصمیم گیری در خصوص مسیر هدایت بسته ها، فقط یکبار و آن هم در حین تنظیم مدار مجازی انجام می شود. در «زیرشبکه های دیتاگرام»، این تصمیم گیری به ازای هر بسته تکرار می شود.

در شبکه های کامپیوتری از الگوریتمهای مسیریابی متنوعی استفاده می شود: الگوریتمهای ایستا نظیر «مسیریابی کوتاهترین مسیر» و «ارسال سیل آسا» (Flooding) و الگوریتمهای پویا شامل «مسیریابی بردار فاصله» و «مسیریابی حالت لینک» (Link State). در اغلب شبکه های واقعی بزرگ، از یکی از این دو روش مسیریابی پویا استفاده شده است. از موارد مهم دیگر در مسیریابی می توان به روش مسیریابی سلسله مراتبی، مسیریابی در محیط ماشینهای ستار، مسیریابی بخش فراگیر، مسیریابی چندپخشی، و مسیریابی در شبکه های همتابه همتا اشاره کرد. زیرشبکه ها ممکن است به سادگی با ازدحام مواجه شوند که منجر به افزایش تأخیر و کاهش ظرفیت مفید خروجی مسیریابها خواهد گردید. طراحان شبکه سعی می کنند با طراحی مناسب از بروز آن جلوگیری نمایند. این تکنیکها عبارتند از: تغییر در سیاستهای ارسال مجدد بسته ها، ذخیره سازی در حافظه نهان، کنترل جریان و نظائر آنها. به هر حال اگر ازدحام رخ داد باید به نحوی به رفع آن اقدام کرد. می توان «بسته های دعوت به آرامش» (Choke Packet) پس فرستاد، بخشی از بار را دور ریخت و یا مکانیزمی نظیر به اینها را بکار بست.

مرحله بعدی پس از حل مسئله ازدحام، تلاش در تضمین کیفیت خدمات است. بدین منظور می توان از روشهایی نظیر «بافر کردن داده ها در ماشین مشتری»، «شکل دهی به ترافیک»، «رزرو منابع» و «کنترل پذیرش» استفاده کرد. راهکارهایی که برای تضمین کیفیت خوب خدمات طراحی و پیاده سازی شده عبارتند از: خدمات مجتمع (شامل RSVP)، خدمات متمایز و MPLS.

شبکه ها از جهات متعدد با هم تفاوت دارند، لذا وقتی می خواهند به یکدیگر متصل شوند مشکلاتی رخ خواهد داد. برخی از این مشکلات را می توان با تکنیک ایجاد تونل (Tunneling) از میان شبکه واسط کاهش داد ولی اگر شبکه های مبدا و مقصد خودشان متفاوت باشند این راهکار نیز با شکست مواجه می شود. وقتی شبکه های متفاوت حداکثر طول بسته هایشان فرق کند به تکنیک قطعه قطعه سازی بسته ها نیاز خواهد شد.

در اینترنت مجموعه وسیعی از پروتکل های مرتبط با لایه شبکه وجود دارد. از بین تمام اینها، پروتکل IP انتقال واقعی بسته ها را بر عهده دارد ولیکن پروتکل های کنترلی دیگر مثل ICMP، ARP و RARP و همچنین پروتکل های مسیریابی مثل OSPF و BGP در کنار آن به فرآیند هدایت و انتقال بسته ها کمک می کنند. اینترنت سریعاً با مشکل کمبود فضای آدرس مواجه شد و برای رفع این مشکل نسخه جدید IP یعنی IPv6 طراحی گردید.

مسائل

۱. دو مسأله از برنامه های کاربردی کامپیوتر ارائه بدهید که برای آنها سرویس اتصال گرا (Connection Oriented) مناسب است. دو نمونه دیگر از برنامه های کاربردی معرفی کنید که در آنها سرویس بدون اتصال مناسبتر است.
۲. آیا وضعیتی پیش می آید که در سرویس اتصال گرا بسته ها خارج از ترتیب تحویل مقصد شوند؟ پاسخ خود را تشریح کنید.
۳. زیرشبکه های دیتاگرام، هر بسته را به صورت واحدهای مجزا و مستقل از هم مسیریابی و هدایت می کنند. زیرشبکه های مدار مجازی بدین نحو عمل نمی کنند فلذا بسته های داده، مسیری از قبل تعیین شده را

- می‌پیمایند. آیا این تعبیر بدان معناست که زیرشبکه‌های مدار مجازی به این قابلیت که بتوانند بسته‌های مجزا و مستقل را از یک مبدا دلخواه به مقصد مورد نظر هدایت کنند نیاز ندارند؟ پاسخ خود را شرح بدهید.
۴. سه نمونه از پارامترهایی را که می‌توان در حین تنظیم یک اتصال، بر روی آنها مذاکره و توافق کرد، نام ببرید.
۵. به مسئله زیر در خصوص پیاده‌سازی «سرویس مدار مجازی» دقت نمایید: اگر در درون زیرشبکه از الگوی مدار مجازی استفاده شده باشد هر بسته داده باید یک سرآیند سه بایتی داشته باشد و هر مسیریاب نیز برای شناسایی هر مدار مجازی به ۸ بایت احتیاج دارد. اگر در درون زیرشبکه از الگوی دیناگرام استفاده شده باشد، به سرآیندهای ۱۵ بایتی نیاز خواهد بود ولی در عوض مسیریاب به فضای حافظه اضافی نیاز نخواهد داشت. هزینه ارسال را برای هر 10^6 بایت یک سنت (به ازای عبور از هر مسیریاب) فرض کنید. حافظه بسیار سریع لازم برای مسیریابها را یک سنت به ازای هر بایت در نظر بگیرید که با فرض ۴۰ ساعت کار در هر هفته، در عرض دو سال مستهلک می‌شود. بطور متوسط هر نشست هزار ثانیه طول می‌کشد و در خلال نشست ۲۰۰ بسته منتقل می‌شود.^۱ هر بسته نیز بطور متوسط از چهار مسیریاب عبور می‌کند. پیاده‌سازی کدامیک از این الگوها ارزان‌تر تمام می‌شود و به چه میزان؟
۶. فرض کنید که تمام مسیریابها و ماشینها به درستی کار می‌کنند و تمام نرم‌افزارها از خطا مصون باشند. آیا باز هم احتمال آنکه یک بسته به اشتباه تحویل ماشینی دیگر شود وجود دارد؟ (حتی اگر این احتمال بسیار ناچیز باشد).
۷. به شبکه شکل ۵-۷ دقت کنید ولیکن وزنه‌های خطوط را نادیده بگیرید. فرض کنید مسیریابی طبق الگوریتم سیل آسا (Flooding) انجام می‌شود. اگر بسته‌های ارسالی از A به D دارای «شماره گام» (Hop count) معادل ۳ باشند فهرست مسیریابی را که بسته از آنها عبور می‌کند، مشخص نمایید. همچنین تعیین کنید که کل گامهایی که بسته‌ها پیموده و پهنای باندی که به ازای آن تلف می‌شود چقدر است.
۸. روشی ساده و ذهنی برای پیدا کردن دو مسیر از یک مبدا مشخص به مقصدی خاص در شبکه ارائه بدهید به نحوی که بتواند در مقابل از بین رفتن خطوط انتقال، دوام بیاورد. (با فرض آنکه حداقل دو مسیر وجود دارد). مسیریابها را قابل اعتماد فرض کنید لذا نگران از کار افتادن آنها نباشید.
۹. به زیرشبکه شکل ۵-۱۳ الف دقت کنید. در این زیرشبکه از «مسیریابی بردار فاصله» استفاده شده و بردارهای ذیل توسط مسیریاب C دریافت شده است:
- از B : (5,0,8,12,6,2) از D : (16,12,6,0,9,10) از E : (7,6,3,9,0,4)
- تاخیر اندازه‌گیری شده تا B و D و E به ترتیب ۶ و ۳ و ۵ است. جدول جدید مسیریابی در C چیست؟ خط خروجی و تاخیر تخمینی رسیدن به هر مقصد را مشخص نمایید.
۱۰. اگر میزان تاخیر به صورت اعداد هشت بیتی ذخیره شود و شبکه ۵۰ مسیریاب داشته باشد و بردارهای تاخیر در هر ثانیه دو بار مبادله شوند چه مقدار از پهنای باند یک خط دو طرفه همزمان (Full Duplex)، در اثر استفاده از الگوریتم مسیریابی توزیع شده [بردار فاصله] تلف می‌شود؟ فرض کنید هر مسیریاب سه خط با مسیریابهای دیگر دارد.
۱۱. در شکل ۵-۱۴، حاصل OR کردن دو مجموعه بیت‌های ACF با یکدیگر ۱۱۱ می‌شود. آیا این موضوع تصادفی است یا در هر زیرشبکه و با هر وضعیتی رخ می‌دهد؟
۱۲. برای مسیریابی سلسله‌مراتبی در زیرشبکه‌ای با ۴۸۰۰ مسیریاب، اندازه هر منطقه (Region) و دسته (Cluster) چقدر باشد تا جدول مسیریابی (در سلسله‌مراتب سه سطحی) حداقل شود؟ نقطه شروع مناسب

۱. نشست را در یک عبارت غیرفنی می‌توانید ارتباط و محاوره یک زوج ماشین در طول زمان فرض کنید.

آنست که فرض کنید داشتن k دسته و k منطقه و k مسیر یاب تقریباً بهینه است، فلذا k تقریباً ریشه سوم 4800 (تقریباً ۱۶) می شود. با سعی و خطا در همسایگی عدد ۱۶، بهترین تعداد دسته، منطقه و مسیر یاب را پیدا کنید.

۱۳. در متن گفته شد که وقتی ماشین متحرک در محل همیشگی خود نیست، بسته های ارسالی به سوی LAN خانگی، متوقف و به سمت LAN موقت او تغییر مسیر داده می شود. در یک شبکه IP که بر روی اترنت پیاده شده، این تغییر مسیر چگونه انجام می شود؟

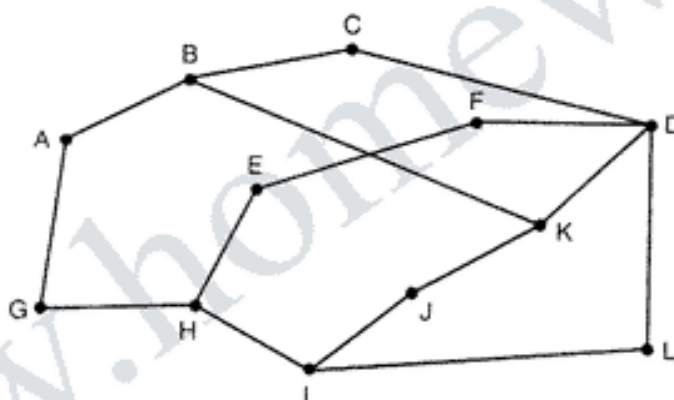
۱۴. با در نظر گرفتن زیر شبکه شکل ۵-۶، اگر B بسته ای را به صورت فراگیر پخش کند، چه تعداد بسته در زیر شبکه تولید می شود اگر:

الف) اگر از روش هدایت در مسیر معکوس (Reverse Path Forwarding) استفاده شود؟

ب) از روش Sink Tree استفاده شود؟

۱۵. شبکه شکل ۵-۱۶ الف را مدنظر قرار بدهید. فرض کنید که خط جدیدی بین F و G اضافه می شود ولی کماکان Sink Tree شکل ۵-۱۶ الف بی تغییر باقی می ماند. در شکل ۵-۱۶ ج چه تغییری حاصل می شود؟

۱۶. برای زیر شبکه بعدی درخت پوشای چند پخش (Multicast Spanning Tree) را برای مسیر یاب C ترسیم کنید به نحوی که اعضای گروه مورد نظر مسیر یابهای A و B و C و D و E و F و I و K باشند.



۱۷. در شکل ۵-۲۰، آیا گره های H یا I در جستجویی که از مبدا A شروع شده، هیچگونه ارسال فراگیر داشته اند؟

۱۸. در شکل ۵-۲۰ فرض کنید که گره B از نو راه اندازی شده است و هیچگونه اطلاعات مسیر یابی در جدول خود ندارد. این گره به ناگاه احتیاج به یافتن مسیری به H پیدا می کند و بسته هایی پخش (Broadcast) با مقدار TTL معادل ۱ و ۲ و ۳... را منتشر می کند. این کار چند دور طول می کشد تا نهایتاً مسیری پیدا شود؟

۱۹. در ساده ترین نسخه الگوریتم Chord برای «جستجوی همتابه همتا» (Peer-to-Peer)، برای جستجو از جدول Finger استفاده نمی شود و به جای آن، جستجو به صورت خطی (یعنی گره به گره) در دو جهت بر روی دایره انجام می گیرد. آیا یک گره می تواند به درستی حدس بزند که جستجو را باید در کدام یک از جهات انجام بدهد؟ پاسخ خود را شرح بدهید.

۲۰. دایره Chord نشان داده شده در شکل ۵-۲۴ را مدنظر قرار بدهید. فرض کنید که گره ۱۰ به ناگاه فعال و وارد خط می شود. آیا ورود این گره در جدول Finger از گره ۱ تأثیری می گذارد و اگر این چنین است چگونه؟

۲۱. به عنوان یک مکانیزم کنترل ازدحام در زیر شبکه ای که در درون از مدار مجازی بهره گرفته است، هر مسیر یاب می تواند از اعلام وصول یک بسته (Ack) طفره برود مگر آنکه: (۱) آگاه شود که آخرین ارسال او

بر روی مدار مجازی، با موفقیت دریافت شده است و (۲) بافر آزاد داشته باشد. برای سادگی فرض کنید مسیریاب برای اعلام وصول از پروتکل «توقف و انتظار» (Stop & Wait) استفاده کرده و برای هر مدار مجازی یک بافر اختصاصی (برای ترافیک هر یک از دو جهت) اختصاص داده است. اگر انتقال هر بسته T ثانیه طول بکشد (انتقال بسته یا پیام Ack به یک اندازه طول می کشد) و n مسیریاب بر روی مسیر قرار گرفته باشد، در چنین شرایطی بسته ها با چه نرخ تحویل ماشین مقصد می شوند؟ فرض کنید که خطای انتقال بسیار ناچیز و اتصال بین ماشین و مسیریاب بی نهایت سریع است.

۲۲. در یک زیرشبکه دیتاگرام به مسیریابها اجازه داده شده تا در صورت نیاز بسته ها را حذف نمایند. احتمال حذف بسته در هر مسیریاب را p فرض کنید. حالتی را در نظر بگیرید که ماشین مبدا به مسیریاب مبدا، مسیریاب مبدا مستقیماً به مسیریاب مقصد و مسیریاب مقصد نیز مستقیماً به ماشین مقصد متصل شده است. اگر یکی از این مسیریابها بسته ای را حذف نمایند، نهایتاً مهلت ماشین مبدا به سر می رسد و آن را از نو ارسال می نماید. اگر خط بین ماشین و مسیریاب را مثل خط بین دو مسیریاب، یک «گام» (Hop) فرض کنیم، مقدار هر یک از موارد زیر را حساب کنید:

الف) میانگین تعداد گامی که یک بسته در هر بار ارسال طی می کند.

ب) میانگین دفعات ارسال یک بسته

ج) میانگین گامهایی که یک بسته دریافتی طی کرده است.

۲۳. دو تفاوت عمده روش «بیت هشدار» (Warning Bit) و روش RED را توضیح دهید.

۲۴. دلیلی ارائه دهید که چرا در الگوریتم «سطل سوراخ»، در هر تیک ساعت مجوز ارسال یک بسته صادر می شود، فارغ از آنکه طول بسته چقدر است.

۲۵. گونه ای از الگوریتم سطل سوراخ که مبتنی بر شمارش بایت است در سیستم خاصی بکار گرفته شده است. قاعده آنست که در هر تیک ساعت یک بسته ۱۰۲۴ بایتی یا دو بسته ۵۱۲ بایتی یا معادل آن، ارسال شود. یکی از محدودیتهای جدی چنین سیستمی را که در متن درس بدان پرداخته ایم، بیان کنید.

۲۶. در یک شبکه ATM از روش سطل نشانه دار برای شکل دهی به ترافیک استفاده شده است. در هر پنج میکروثانیه یک توکن جدید در سطل قرار داده می شود. هر توکن برای ارسال یک سلول حاوی ۴۸ بایت داده، مناسب می باشد. حداکثر نرخ ارسال مجاز چقدر است؟

۲۷. ترافیک یک کامپیوتر متصل به شبکه 6 Mbps طبق الگوریتم سطل نشانه دار، شکل دهی و منظم می شود. سطل نشانه دار با نرخ یک مگابیت بر ثانیه پر می شود. ظرفیت سطل هشت مگابیت است که در همان ابتدا پر شده است. کامپیوتر در چه مدت می تواند با سرعت ۶ مگابیت بر ثانیه ارسال داشته باشد؟

۲۸. فرض کنید در توصیف مشخصات یک «جریان» حداکثر طول هر بسته ۱۰۰۰ بایت، نرخ سطل نشانه دار ده میلیون بایت در ثانیه، اندازه سطل نشانه دار یک میلیون بایت و حداکثر نرخ ارسال ۵۰ میلیون بایت در ثانیه تعیین شده است. یک ماشین در چه مدتی قادر به ارسال انفجاری داده ها با حداکثر سرعت خود می باشد؟

۲۹. شبکه شکل ۵-۳۷ از RSVP و درخت چندپخشی نشان داده شده برای ماشینهای ۱ و ۲، بهره گرفته است. فرض کنید که ماشین ۳ برای دریافت یک «جریان» از ماشین ۱، تقاضای کانالی با پهنای باند 2MB/sec می دهد. همچنین برای دریافت جریانی از ماشین ۲ تقاضای 1MB/sec می دهد. بطور همزمان ماشین ۴، 2MB/sec برای دریافت جریان از ماشین ۱ و همچنین ماشین ۵ برای دریافت جریان از ماشین ۲، 1MB/sec پهنای باند درخواست می کنند. مجموع کل پهنای باند رزرو شده در مسیربایهای A و B و C و E و H و J و K و L برای این درخواستها چقدر است؟

۳۰. پردازنده یک مسیریاب قادر به پردازش ۲ میلیون بسته در هر ثانیه است. بار عرضه شده به آن ۱/۵ میلیون بسته در ثانیه است. اگر در مسیر رسیدن از مبدا به مقصد، ۱۰ مسیریاب قرار گرفته باشند، تأخیر ناشی از صف بندی و پردازش بسته چقدر است؟
۳۱. فرض کنید کاربری از خدمات متمایز و «هدایت پرشتاب» بهره می گیرد. آیا تضمینی وجود دارد که بسته های پرشتاب تأخیری کمتر از بسته های معمولی داشته باشند؟ دلیل خود را تشریح کنید.
۳۲. آیا به فرآیند قطعه قطعه سازی بسته ها در شبکه های بهم متصل شده مدار مجازی نیز احتیاج است یا آنکه این نیاز فقط در شبکه های دیتاگرام وجود دارد؟
۳۳. فرآیند ایجاد تونل از میان یک شبکه الحاق شده مدار مجازی ساده و سرراست است: مسیریاب چند پروتکلی در یکی از دو طرف یک مدار مجازی با مسیریاب طرف دیگر ایجاد و تنظیم می کند و بسته ها را از طریق این مدار مجازی مبادله می نماید. آیا ایجاد تونل از میان یک شبکه دیتاگرام نیز ممکن است؟ اگر جواب مثبت است چگونه؟
۳۴. فرض کنید که ماشین میزبان A به مسیریاب R1، R1 نیز به مسیریاب دیگری به نام R2، و R2 نیز به ماشین میزبان B متصل است. فرض کنید که یک پیام TCP حاوی ۹۰۰۰ بایت داده و ۲۰ بایت سرآیند، جهت تحویل به B به نرم افزار IP در ماشین A تسلیم می شود. محتوای فیلدهای Total Length، Identification، DF، MF و Fragment offset را در هر بسته IP که از یکی از سه لینک A-R1، R1-R2، R2-B عبور می کند، مشخص نمایید؛ لینک A-R1 می تواند از فریمی با طول حداکثر ۱۰۲۴ بایت (که ۱۴ بایت آن هم سرآیند فریم لایه پیوند داده ها محسوب می شود) پشتیبانی می کند. لینک R1-R2 می تواند از فریمی با طول ۵۱۲ بایت حمایت کند (با احتساب ۸ بایت سرآیند فریم) و لینک R2-B از فریمهایی با طول حداکثر ۵۱۲ بایت (با احتساب ۱۲ بایت سرآیند فریم) پشتیبانی می کند.
۳۵. یک مسیریاب بسته های IP با طول ۱۰۲۴ بایت (با احتساب داده و سرآیند) در خروجی خود گسیل می دارد. با فرض آنکه بسته ها فقط برای ده ثانیه زنده می مانند، مسیریاب حداکثر با چه سرعتی می تواند بسته ها را بفرستد بدون آنکه خطر به صفر برگشتن فیلد شماره شناسایی دیتاگرام (Datagram ID) وجود داشته باشد؟
۳۶. یک دیتاگرام IP که از گزینه Strict Source Routing استفاده کرده، مجبور است قطعه قطعه شود. به نظر شما آیا این گزینه بایستی در تمام قطعات آن قرار داده شود یا آنکه قرار دادن این گزینه در قطعه اول کفایت می کند؟ پاسخ خود را شرح بدهید.
۳۷. فرض کنید در کلاس B به جای ۱۶ بیت برای شماره شبکه، ۲۰ بیت در نظر گرفته می شد. در چنین حالتی چند شبکه کلاس B می توان داشت؟
۳۸. آدرس IP با نمایش هگزادسیمال C22F1582 را به نماد نقطه دار تبدیل کنید.
۳۹. شبکه ای در اینترنت، از الگوی زیر شبکه 255.255.240.0 استفاده کرده است. این شبکه حداکثر چند ماشین میزبان می تواند داشته باشد؟
۴۰. تعداد بسیار زیادی آدرس IP متوالی از نقطه شروع 198.16.0.0 در اختیار می باشد. فرض کنید چهار سازمان A و B و C و D به ترتیب ۴۰۰۰، ۲۰۰۰، ۴۰۰۰ و ۸۰۰۰ آدرس IP درخواست می کنند. برای هر یک از اینها اولین آدرس IP، آخرین آدرس IP و الگوی زیر شبکه را به شکل w.x.y.z/s معین کنید.
۴۱. یک مسیریاب آدرسهای IP جدیدی طبق فهرست زیر دریافت می کند (جهت درج در جدول مسیریابی):
57.6.120.0/21 و 57.6.112.0/21 و 57.6.104.0/21 و 57.6.96.0/21
اگر برای رسیدن به تمام این شبکه ها از خط مشترکی استفاده شود، آیا می توان این آدرسها را «تجمیع»

(Aggregate) کرد؟ اگر می توان به چه نحو؟ اگر خیر، چرا؟

۴۲. مجموعه ای از آدرسهای IP از 29.18.0.0 تا 29.18.128.255 به صورت 29.18.0.0/17 «تجمیع» شده اند. ولیکن یک فاصله خالی ۱۰۲۴ تایی متناسب نشده در محدوده 29.18.60.0 تا 29.18.63.255 وجود داشته که به ناگاه به شبکه ای اختصاص داده می شود که خط خروجی رسیدن به آن، با بقیه فرق می کند. آیا در اینجا باید فضای تجمیع شده آدرسها را مثل اول به بلوکهای اولیه تقسیم کرد و بلوک جدید به جدول اضافه و فرایند تجمیع از نو انجام گیرد؟ اگر نه چه کار دیگری می توان انجام داد؟
۴۳. یک مسیریاب در جدول مسیریابی خود درایه های CIDR زیر را در اختیار دارد:

Address/mask (آدرس/الگوی زیر شبکه)	Next Hop (گام بعدی)
135.46.56.0/22	Interface 0
135.46.60.0/22	Interface 1
192.53.40.0/23	Router 1
default	Router 2

اگر بسته ای با یکی از آدرسهای IP زیر دریافت شود، مسیریاب با آن بسته چه می کند:

- الف) 135.46.63.10
ب) 135.46.57.14
ج) 135.46.52.2
د) 192.53.40.7
ه) 192.53.56.7

۴۴. سیاست بسیاری از شرکتها بر آنست که دو (یا چند) مسیریاب متصل به اینترنت در شبکه خود داشته باشند تا در صورت از کار افتادن یکی از آنها، از دیگری استفاده شود. آیا با چنین سیاستی باز هم می توان از NAT بهره گرفت؟ پاسخ خود را تشریح کنید.
۴۵. فرض نمایید پروتکل ARP را برای دوستان توضیح داده اید. پس از اتمام توضیحاتتان او می گوید: «متوجه شدم: ARP سرویسی را به لایه شبکه ارائه می دهد و طبقاً جزئی از لایه پیوند داده ها است» چه حرفی برای گفتن به او دارید؟
۴۶. ARP و RARP هر دو آدرسهای را از یک فضا به فضایی دیگر می نگارند (IP به MAC و بالعکس). از این دیدگاه هر دو مشابه یکدیگر هستند، ولی پیاده سازی آنها متفاوت از یکدیگر است. عمده ترین تفاوت آنها در چیست؟
۴۷. روشی را برای بازسازی قطعات یک بسته IP در مقصد، معرفی و تشریح نمایید.
۴۸. اغلب الگوریتمهای بازسازی دیتاگرامهای IP، دارای زمان سنج (تایمر) خاصی هستند تا در صورت از بین رفتن یک قطعه، بقیه قطعات تا ابد در بافر نمانند. فرض کنید یک دیتاگرام به چهار قطعه تقسیم شده است. سه قطعه اول سر موقع دریافت می شوند ولی قطعه آخر با تأخیر مواجه می شود. عاقبت مهلت دریافت آن منقضی شده و سه قطعه دیگر از حافظه گیرنده، پاک می گردد. اندکی بعد، آخرین قطعه از راه می رسد. گیرنده با آن چه باید بکند؟
۴۹. چه در IP و چه در ATM، کد کشف خطای checksum فقط سرآیند بسته را در بر می گیرد نه بخش داده را. فکر می کنید منطق این طراحی چه بوده است؟
۵۰. شخصی که در بوستون زندگی می کند در سفری به مینیاپولیس کامپیوتر کیفی خود را به همراه می برد. خوشبختانه، شبکه LAN او در مینیاپولیس یک شبکه محلی بی سیم و مبتنی بر IP است و او مجبور به وصل

- کابل به جایی نیست. آیا برای آنکه ترافیک پست الکترونیکی یا دیگر داده ها به درستی تحویل او شود، کماکان لازم است که از دو عامل خانگی و خارجی استفاده شود؟
۵۱. آدرسها در IPv6 شانزده بیتی هستند. اگر در هر پیکوتانیه یک بلوک یک میلیون تایی آدرس IP رزرو شود چه مدت طول می کشد تا آدرسها تمام شوند؟
۵۲. فیلد پروتکل که در IPv4 وجود داشت اکنون در سرآیند ثابت IPv6 وجود ندارد. به چه دلیل؟
۵۳. آیا وقتی که پروتکل IPv6 به صحنه عمل آمد نیازی به تغییر در ARP هست؟ اگر این چنین است آیا این تغییرات در عملکرد و مفهوم است یا در پیاده سازی فنی؟
۵۴. برنامه ای بنویسید که فرآیند مسیریابی به روش سیل آسا را شبیه سازی نماید. هر بسته دارای یک فیلد شمارنده است که به ازای هر گام (Hop) یک واحد از آن کم می شود. وقتی این شمارنده به صفر برسد، بسته حذف می گردد. زمان را «گسسته» (Discrete) فرض کنید و در هر واحد زمان فقط یک بسته بر روی خط ارسال یا دریافت می شود. برنامه خود را در سه نسخه تهیه کنید: (۱) بسته ورودی بر روی تمام خطوط ارسال گردد. (۲) بسته ورودی بر روی تمام خطوط به استثنای خطی که از آن داخل شده، فرستاده شود. (۳) بسته ورودی بر روی k تا از بهترین خطوط (به صورت آماری و غیر دقیق) ارسال گردد. روش سیل آسا را با روش معمولی (یعنی وقتی بسته ورودی بر روی بهترین خط خروجی ارسال می شود) از لحاظ پهنای باند و تأخیر مقایسه نمایید.
۵۵. برنامه ای بنویسید که یک شبکه کامپیوتری را به صورت زمان گسسته شبیه سازی کند. در هر واحد زمان، بسته ای که سر صف هر یک از خطوط خروجی مسیریابها قرار گرفته اند، ارسال و یک گام جلو می روند. هر مسیریاب فضای بافر محدودی دارد. اگر بسته ای دریافت شود ولی جایی برای آن وجود نداشته باشد، حذف شده و از نو ارسال نخواهد شد. در عوض یک پروتکل انتها به انتها (End to End) وجود دارد که در صورت ارسال داده ای که دریافت آن تصدیق نشده از مبداء، داده ها را مجدداً ارسال می کند. توان مفید (ظرفیت مفید یا Throughput) شبکه را برحسب تابعی از زمان انقضای مهلت (Timeout) و پارامتر نرخ خطا ترسیم نمایید.
۵۶. تابعی بنویسید که عملیات هدایت (Forwarding) را در یک مسیریاب مبتنی بر IP انجام بدهد. این تابع یک پارامتر ورودی دارد و آن هم آدرس IP است. فرض کنید این تابع به یک جدول مسیریابی سراسری دسترسی دارد. این جدول آرایه ای است که سه ستون دارد و محتویات هر ستون، اعداد صحیح هستند. این ستونها عبارتند از: آدرس IP، الگوی زیرشبکه (Subnet Mask) و شماره خط خروجی مورد استفاده [برای رسیدن به شبکه ای با آدرس IP متناظر]. این تابع، باید آدرس IP دریافتی از پارامتر ورودی را در جدول فوق الذکر به روش CIDR جستجو کرده و شماره خط خروجی متناسب با آن را به عنوان مقدار برگشتی باز گرداند.
۵۷. با استفاده از فرامین اجرایی traceroute (در یونیکس) یا tracert (در ویندوز)، مسیر رسیدن از کامپیوتر خودتان به چند دانشگاه در دیگر قاره ها را پیدا کنید. برخی از سایتهای متعلق به دانشگاههای معروف که می توانید در بررسی خود از آنها استفاده کنید، عبارتند از:

www.berkeley.edu	(کالیفرنیا)
www.mit.edu	(ماساچوست)
www.vu.nl	(آمستردام)
www.ucl.ac.uk	(لندن)
www.usyd.edu.au	(سیدنی)
www.u-tokyo.ac.jp	(توکیو)
www.uct.ac.za	(کیپ تاون)

لایه انتقال



لایه انتقال (Transport Layer)، فقط در یک لایه خلاصه نمی شود بلکه قلب تپنده سلسله پروتکل های شبکه است. وظیفه این لایه آن است که داده ها را به روشی قابل اعتماد و کم هزینه از ماشین مبدا به ماشین مقصد انتقال بدهد، فارغ از آن که ماهیت شبکه یا شبکه های فیزیکی مورد استفاده چیست. بدون لایه انتقال مفهوم پروتکل های لایه ای، معنای حقیقی خود را پیدا نخواهند کرد. در این فصل جزئیات لایه انتقال را شامل خدماتی که عرضه می کند، طراحی آن، پروتکل های مرتبط و کارایی آنها را بررسی خواهیم کرد.

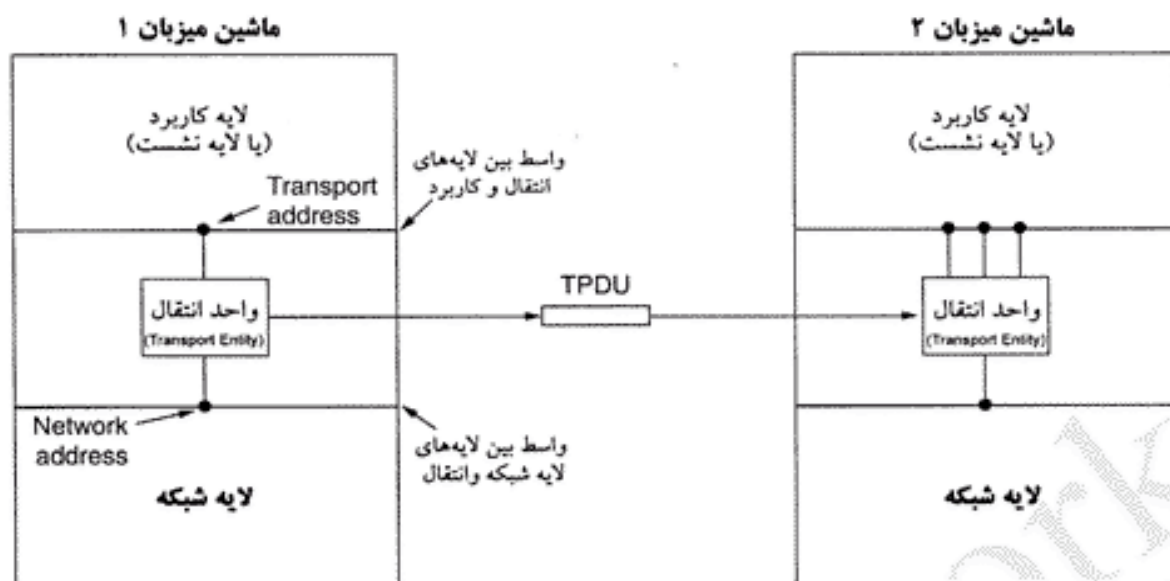
۱-۶ خدمات انتقال (The Transport Service)

در بخش های آتی، خدماتی را که لایه انتقال ارائه می نماید، اجمالاً بررسی کرده و به انواع سرویس هایی که به لایه کاربرد عرضه می شود نگاهی خواهیم انداخت. برای آن که حقیقت خدمات انتقال را آشکارتر کرده باشیم دو مجموعه از «عملکردهای اولیه» (Primitives) برای لایه انتقال تعریف کرده ایم: مجموعه اول عملکردهای ساده و فرضی هستند که به فهم ایده اصلی کمک می کند. سپس مجموعه ای از واسطه ها (Interfaces) را که عموماً در اینترنت بکار گرفته می شوند، معرفی خواهیم کرد.

۱-۱-۶ خدمات ارائه شده به لایه های بالاتر

هدف نهایی لایه انتقال آن است که سرویسی کارآمد، مطمئن و کم هزینه به کاربران خود بدهد. کاربران لایه انتقال، پروسه های لایه کاربرد (یا به عبارتی برنامه های کاربردی) هستند. برای نائل آمدن به این هدف، لایه انتقال از خدماتی که لایه شبکه عرضه کرده، بهره می گیرد. نرم افزار یا سخت افزاری که در لایه انتقال این عملیات را انجام می دهد اصطلاحاً «واحد انتقال» (Transport Entity) نامیده می شود. واحد انتقال ممکن است درون هسته سیستم عامل یا به صورت یک پروسه کاربردی مجزا یا در قالب یک بسته کتابخانه ای^۱ در بطن برنامه های کاربردی شبکه و یا حتی در درون کارت واسط شبکه تعبیه شده باشد. در شکل ۱-۶ ارتباط منطقی بین لایه های شبکه، انتقال و کاربرد به تصویر کشیده شده است.

همانگونه که خدمات لایه شبکه بر دو نوع اتصال گرا و بدون اتصال ارائه می شود، خدمات لایه انتقال نیز بر همین دو نوع است. خدمات مبتنی بر اتصال (هماهنگیهای قبلی) در لایه انتقال از بسیاری جهات مشابه با خدمات اتصال گرای لایه شبکه است. در هر دو مورد، «اتصال» سه مرحله تکوین دارد: «ایجاد اتصال» (Establishment)،



شکل ۶-۱. لایه‌های شبکه، انتقال و کاربرد.

«انتقال داده» (Data Transfer)، «ختم اتصال» (Release)، آدرس‌دهی و کنترل جریان نیز در هر دو لایه مشابه هم هستند. مضاف بر این، خدمات بدون اتصال در لایه انتقال نیز شباهت بسیار زیادی به خدمات بدون اتصال در لایه شبکه دارد.

در اینجا یک سؤال بدیهی به ذهن خطور می‌کند: اگر خدمات لایه انتقال این قدر به خدمات لایه شبکه شبیه است، پس چرا این دو لایه از هم جدا هستند؟ چرا فقط به یکی از آنها بسنده نمی‌شود؟ پاسخ این سؤال سراسرست ولی بنیادی است و از شکل ۱-۹ نشأت می‌گیرد. کد اجرایی لایه انتقال کلاً بر روی ماشین کاربر اجرا می‌شود در حالی که لایه شبکه اغلب بر روی مسیریابها اجرا می‌گردد که آنها نیز تحت مدیریت یک «حامل» (Carrier) هستند (حداقل در WAN اینگونه است). اگر لایه شبکه خدماتی ناکافی عرضه کند (که اغلب اینگونه است) چه اتفاقی می‌افتد؟ اگر لایه شبکه (به دلایلی مثل ازدحام، بروز حلقه یا پایان عمر بسته) تعدادی از بسته‌ها را از دست بدهد چه باید کرد؟ اگر یک مسیریاب هر از گاهی از کار بیفتد چه می‌شود؟

در موارد فوق فقط می‌توان گفت که مشکلات جدی رخ می‌دهد! کاربران هیچ کنترل واقعی بر لایه شبکه ندارند و طبعاً نمی‌توانند مشکلات ناشی از خدمات ناقص مسیریابها را مثلاً با تعویض مسیریاب یا مدیریت بهتر خطا در لایه پیوند داده برطرف کنند. تنها راه ممکن آن است که بر روی لایه شبکه، لایه دیگری قرار داده شود تا کیفیت خدمات را بهبود بدهد. در یک زیر شبکه اتصال‌گرا، اگر «واحد انتقال» (Transport Entity) در حین انتقال طولانی مدت بسته‌ها، به ناگاه متوجه شود که اتصال شبکه قطع شده و از سرنوشت داده‌های در حال انتقال نیز بی‌خبر باشد، می‌تواند یک اتصال جدید یا «واحد انتقال راه دور» (Remote Transport Entity) برقرار نماید؛ به کمک اتصال جدید، واحد انتقال می‌تواند از همتای خود سؤال کند که کدام بخش از داده‌ها دریافت کرده و کدام بخش را دریافت نکرده و از جایی که داده‌ها نرسیده‌اند، شروع به ارسال مجدد نماید.

در حقیقت، وجود لایه انتقال این امکان را فراهم آورده تا خدمات انتقال داده‌ها، قابل اعتمادتر از خدمات لایه زیرین یعنی لایه شبکه باشد. بسته‌های گمشده یا تکراری توسط لایه انتقال کشف می‌شوند. مضاف بر اینها، عملکردهای اولیه لایه انتقال می‌تواند به صورت فراخوانی توابع کتابخانه‌ای پیاده‌سازی و در اختیار پروسه‌های لایه بالاتر گذاشته شود تا لایه بالاتر از عملکردهای اولیه و توابع پایه لایه شبکه (و درگیری با جزئیات این لایه که

امکانات ضعیفی نیز ارائه می‌کند) مستقل باشد زیرا خدماتی که لایه شبکه عرضه می‌نماید در شبکه‌های متفاوت می‌تواند تفاوت‌های بنیادی داشته باشد (مثلاً خدمات یک شبکه بدون اتصال با خدمات اتصال‌گرای یک WAN تفاوت اساسی و ماهوی دارد). پنهان کردن خدمات لایه شبکه در پشت یک مجموعه از «عملکردها و توابع اولیه» موجب می‌شود که در صورت تغییر در خدمات لایه شبکه بتوان با عوض کردن توابع کتابخانه‌ای (به گونه‌ای که همان خدمات و عملکرد قبلی خود را در شبکه جدید ارائه کنند)، این تغییر را در سطح همین لایه محدود نگاه داشت، [بدین ترتیب هیچ یک از اجزاء لایه‌های بالاتر از چنین تغییری آگاه نخواهند شد].

با تکیه بر لایه انتقال، برنامه‌نویسان نرم افزارهای کاربردی می‌توانند کد برنامه خود را مبتنی بر یک مجموعه استاندارد از توابع و عملکردهای اولیه بنویسند و هیچگونه نگرانی از بابت تفاوت در اجزاء و مکانیزمهای زیر شبکه یا انتقال نامطمئن و غیرقابل اعتماد نداشته باشند. اگر تمام شبکه‌های واقعی، بدون اشکال و قابل اعتماد بودند و همه آنها خدمات مشابهی ارائه می‌کردند و این تضمین وجود می‌داشت که هیچگاه تغییر نکنند، آنگاه به خدمات لایه انتقال نیازی نبود. ولیکن [به دلیل اختلافات بنیانی و تنوع بسیار زیاد لایه‌های زیرین] در دنیای واقعی، وجود این لایه، لایه‌های بالایی را از درگیری با جزئیات تکنولوژی، طراحی و نواقص زیر شبکه دور نگه می‌دارد.

به همین دلیل بسیاری از افراد، لایه‌های یک تا چهار را در یک دسته و لایه‌های بالاتر از ۴ را در دسته‌ای دیگر قرار می‌دهند. چهار لایه اول را می‌توان «ارائه دهنده خدمات انتقال» (Transport Service Provider) تصور کرد در حالی که بقیه لایه‌های فوقانی، «استفاده‌کنندگان از خدمات انتقال» (Transport Service User) محسوب می‌شوند. تفکیک بین لایه‌های ارائه دهنده خدمات و لایه‌های استفاده‌کننده از این خدمات، نقش لایه انتقال را حساس و کلیدی کرده است چرا که این لایه در مرز بین این دو دسته قرار گرفته و باید به لایه‌های فوقانی خدمات انتقال مطمئن داده‌ها را ارائه بدهد.

۲-۱-۶ عملکردهای اولیه و توابع بنیانی لایه انتقال

برای آن که امکان دسترسی به خدمات لایه انتقال برای استفاده‌کنندگان این خدمات فراهم شود، این لایه باید در قالب یک «واسط خدمات انتقال» (Transport Service Interface) مجموعه‌ای از عملیات و توابع را در اختیار برنامه‌های کاربردی بگذارد.^۱ هر رده از خدمات لایه انتقال، «واسط» مختص به خود را دارد. در این بخش خدمات ساده و فرضی لایه انتقال و واسط آن را بررسی می‌کنیم تا با اصول پایه آشنا شویم. سپس در بخشهای آتی، مروری بر مثالهای واقعی خواهیم داشت.

خدمات لایه انتقال در عین شباهت با خدمات لایه شبکه، تفاوت‌های مهم دارند: تفاوت عمده آنها در این است که لایه شبکه خدمات شبکه واقعی و در حال استفاده را مدل می‌کند و شبکه‌های واقعی به دلایل مختلفی ممکن است بسته‌ها را از دست بدهند فلذا خدمات لایه شبکه عموماً قابل اعتماد نیستند.

در عوض خدمات انتقال اتصال‌گرا در لایه انتقال کاملاً قابل اعتماد است. در حالی که شبکه‌های واقعی مصون از خطا نیستند، لایه انتقال بر روی این شبکه نامطمئن قرار می‌گیرد و با مکانیزمهای خاصی که بدان خواهیم پرداخت، تمام این خطاها و مشکلات را جبران و برطرف می‌نماید.

به عنوان مثال، دو پروسه را در نظر بگیرید که در محیط یونیکس از طریق یک «لوله» (Pipe) به یکدیگر مرتبط شده‌اند. این دو پروسه فرض را بر آن می‌گذارند که ارتباط آنها بی‌نقص و مصون از هر نوع خطایی است. این دو تمایلی ندارند که در خصوص مکانیزمهایی مثل تصدیق دریافت داده‌ها (Acknowledgement)، بسته‌هایی که از بین می‌روند، ازدحام یا مسائلی نظیر آن چیزی بدانند. آنچه که این پروسه‌ها انتظار دارند یک ارتباط صددرصد

۱. مفهوم «واسط یا interface» را همان مفهوم نرم‌افزاری آن در مدل برنامه‌نویسی در نظر بگیرید. م

مطمئن و مصون از خطاست. پروسه A داده های خود را در انتهای «لوله» قرار می دهد و پروسه B داده ها را از ابتدای این لوله بر می دارد. تمام آنچه که خدمات اتصال گرای لایه انتقال باید ارائه بدهند چیزی شبیه به همین مفهوم است: یعنی پروسه های کاربری واقع بر هر نقطه از شبکه ای که نامطمئن است، بتوانند یک دنباله بیت مصون از خطا (Error Free Bit Stream) را ارسال یا دریافت کنند و نقایص زیر شبکه از چشم آنها پنهان بماند.

البته لایه انتقال می تواند خدمات نامطمئن و بدون اتصال (از نوع دیتاگرام) نیز ارائه بدهد ولیکن حرف زیادی برای گفتن ندارد و تمرکز اصلی ما در این فصل بر روی خدمات اتصال گرا و قابل اعتماد در لایه انتقال است. علیرغم آن، برخی از برنامه های کاربردی مثل عملیات چند رسانه ای، از مزایای انتقال بدون اتصال بهره گرفته اند لذا چند کلمه ای در خصوص آن صحبت خواهیم کرد.

تفاوت دیگر بین خدمات لایه شبکه و لایه انتقال آن است که استفاده کنندگان از خدمات آنها تفاوت بنیادی دارند. خدمات لایه شبکه به «واحد انتقال» (Transport Entity) ارائه می شود. کاربران بسیار کمی هستند که بخواهند «واحد انتقال» اختصاصی برای خود بنویسند یا برنامه های آنها مستقیماً از خدمات حداقل و ناقص لایه شبکه استفاده نمایند. برعکس، اکثر برنامه ها (و طبعاً برنامه نویسان) فقط عملکردهای اولیه و توابع بنیادی لایه انتقال را می بینند. در نتیجه، خدمات لایه انتقال باید سهل الوصول و استفاده از آنها ساده و سراسر باشد.

برای آن که احساسی از ماهیت خدمات لایه انتقال پیدا کنید به پنج عملکرد (تابع) اولیه (Primitives) که در جدول ۶-۲ فهرست شده، دقت کنید. این توابع که نقش یک «واسط انتقال» (Transport Interface) را ایفاء می کنند، حداقل نیازهای مورد انتظار از لایه انتقال محسوب می شوند. این مجموعه توابع، به برنامه های کاربردی امکان می دهد تا بتوانند اتصالاتی «ایجاد»، «استفاده» و نهایتاً آن را «ختم» نمایند. چنین امکانی برای اغلب برنامه های کاربردی کافی است.

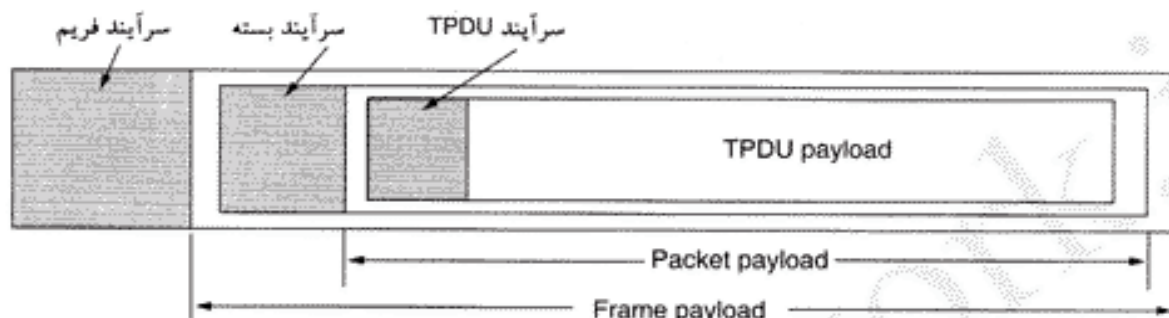
توصیف	بسته ارسالی	نام عملکرد (تابع) اولیه
متوقف (بلوکه) می شود تا آنکه پروسه ای سعی در برقراری اتصال کند.	(none)	LISTEN
بصورت فعال سعی در برقراری یک اتصال می کند.	CONNECTION REQ.	CONNECT
داده می فرستد.	DATA	SEND
متوقف می شود تا آنکه بسته داده برسد.	(none)	RECEIVE
تلاش برای قطع (خاتمه) اتصال	DISCONNECTION REQ.	DISCONNECT

شکل ۶-۲. عملکردهای (توابع) اولیه برای ارائه خدمات ساده انتقال.

برای آن که ببینیم از این عملکردها و توابع چگونه استفاده می شود، یک برنامه کاربردی شامل یک «سرویس دهنده» و تعدادی «مشری» راه دور را مد نظر قرار بدهید. برای شروع، برنامه سرویس دهنده با فراخوانی سیستمی، تابع LISTEN را اجرا می کند. اجرای این تابع سرویس دهنده را بلوکه کرده و به حالت انتظار می برد تا آن که یک «مشری» ظاهر شود (بعبارت دیگر از راه دور تقاضای ارتباط کند). در طرف مقابل، وقتی مشری می خواهد با سرویس دهنده محاوره کند، تابع CONNECT را اجرا می کند. «واحد انتقال» (Transport Entity) ضمن بلوکه کردن برنامه مشری، با ارسال بسته ای به سرویس دهنده این تقاضا را به اطلاع او می رساند. در درون فیلد داده از این بسته، پیام لایه انتقال قرار گرفته که نهایتاً تحویل «واحد انتقال» خواهد شد.

در اینجا بایستی یک واژه را معرفی نمائیم: به دلیل فقدان واژه بهتر، بایستی مجبوریم برای نامگذاری ساختار پیامهایی که بین «واحدهای انتقال» مبادله می شوند، از واژه TPDU^۱ استفاده نمائیم. طبعاً TPDU (که بین لایه های

انتقال مبادله خواهند شد) در درون «بسته لایه شبکه» [مثلاً درون یک بسته IP] جاسازی و حمل می شود. خود «بسته» نیز در درون یک فریم جاسازی و توسط لایه پیوند داده، مبادله می شود. وقتی فریمی دریافت گردد، ابتدا لایه پیوند داده، سرآیند آن فریم را پردازش کرده و محتوای آن را به «واحد شبکه» (Network Entity) تسلیم می کند. واحد شبکه نیز سرآیند بسته را پردازش کرده و محتوای فیلد داده آن بسته را تحویل «واحد انتقال» در لایه بالا می نماید. شکل ۳-۶، تودرتویی این بسته ها را به تصویر کشیده است.



شکل ۳-۶. تودرتویی TPDUs، بسته ها و فریم ها.

به مثال برنامه سرویس دهنده/مستری خودمان برگردیم: فراخوانی تابع CONNECT در برنامه مشتری موجب می شود که یک بسته TPDU CONNECTION REQUEST به سوی سرویس دهنده، ارسال شود. هر گاه این بسته دریافت شود، واحد انتقال بررسی می کند که آیا سرویس دهنده ای با اجرای LISTEN بلوکه شده و منتظر است؟ (به عبارتی بررسی می کند که آیا پروسه ای علاقمند به پردازش و پاسخ به چنین تقاضایی هست یا خیر)؛ اگر چنین باشد، پروسه مربوطه را از حالت بلوکه خارج کرده و در پاسخ، بسته CONNECTION ACCEPTED TPDU به مشتری برگردانده می شود. وقتی این TPDU دریافت شود، برنامه مشتری نیز از حالت بلوکه خارج شده و اتصال برقرار می شود.

پس از این مراحل، داده ها می توانند با استفاده از توابع SEND و RECEIVE بین دو برنامه مبادله شوند. در ساده ترین حالت، هر یک از طرفین می توانند به کمک تابع RECEIVE به حالت انتظار وارد شده و منتظر بمانند تا طرف مقابل با تابع SEND اقدام به ارسال داده نماید. وقتی این TPDU دریافت شد، گیرنده از حالت بلوکه خارج می گردد. این پروسه نیز TPDU مربوطه را دریافت و پاسخ لازم را برمی گرداند. مادامی که طرفین ارتباط به نوبت ارسال نمایند این الگو به خوبی کار می کند.

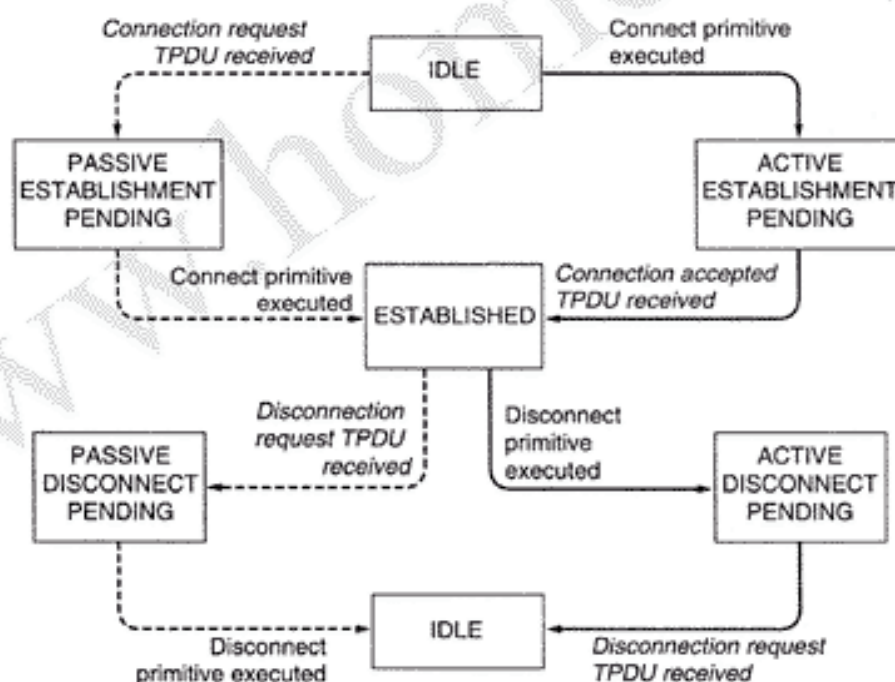
دقت کنید که لایه انتقال حتی برای مبادله یکطرفه و ساده داده ها بسیار پیچیده تر از لایه شبکه عمل می کند. دریافت یکایک بسته هایی که ارسال می شوند باید به تایید طرف مقابل برسد؛ (با ارسال پیغام Ack). حتی بسته هایی که حامل TPDUs کنترلی هستند نیز باید (به صورت مستقیم یا ضمنی) اعلام وصول شوند. تصدیق وصول بسته ها بر عهده «واحد انتقال» (Transport Entity) است و این کار از دید کاربران لایه انتقال مخفی می ماند. بنابراین (طبق آنچه که در فصل سوم اشاره شد)، واحد انتقال باید نگران زمان سنجها (تایمرها) و ارسال مجدد (Retransmission) داده ها باشد. هیچیک از این عملیات برای کاربران لایه انتقال مشهود نیست. از دید کاربران لایه انتقال، یک «اتصال» (Connection) به مثابه یک «لوله مطمئن انتقال بیت» (Bit Pipe) است: یکی از کاربران ببتها را در ابتدای این لوله تزریق می کند و به همان صورت در انتهای دیگر لوله، تحویل داده می شود. توانایی پنهان سازی پیچیدگیهای زیرین، پروتکل های لایه ای را به یک ابزار قدرتمند تبدیل کرده است.

دیگر به یک اتصال نیازی نباشد باید آن را خاتمه داد تا فضای جدولی که در حافظه هر کدام از

«واحدهای انتقال» به آن اختصاص داده شده، آزاد شود. قطع یک اتصال (Disconnection) به دو صورت ممکن است: نامتقارن و متقارن. در روش «نامتقارن» (Asymmetric) هر یک از پروسه های روبرو می توانند با صدور تابع پایه DISCONNECT، به صورت یکطرفه اقدام به ختم ارتباط نمایند. صدور DISCONNECT موجب ارسال یک TPDU کنترلی خاص (DISCONNECT TPDU) به سوی «واحد انتقال» طرف مقابل می شود. پس از دریافت این TPDU ارتباط قطع می شود.

در روش «متقارن» (Symmetric) [چون ارتباط دوجته - Bidirectional - است] هر یک از جهت های انتقال به طور جداگانه بسته می شود: وقتی یکی از طرفین DISCONNECT می کند، منظورش آن است که داده دیگری برای ارسال ندارد ولیکن کماکان آماده پذیرش داده های شریک مقابل خود است. در این مدل، یک «اتصال»، صرفاً زمانی بطور کامل قطع می شود که هر دو طرف DISCONNECT نمایند.

در شکل ۴-۶ یک «دیاگرام حالت»^۱ برای فرآیند ایجاد و ختم اتصال (به کمک توابع اولیه) ترسیم شده است. «گذار»^۲ از یک «حالت»^۳ به حالت دیگر بر اثر بروز یک «رخداد»^۴ اتفاق می افتد، خواه این رخداد در اثر صدور یک تابع اولیه در پروسه محلی کاربر، حادث شود و خواه در اثر ورود یک بسته کنترلی (مثل Connection Request) رخ بدهد. برای سادگی فرض می کنیم دریافت هر TPDU بطور مجزا تصدیق (Ack) شود.^۵ همچنین فرض کرده ایم که قطع ارتباط به صورت متقارن انجام می شود و پروسه مشتری زودتر تقاضای قطع اتصال می دهد. لطفاً دقت کنید که این مدل کاملاً ساده است. بعداً در خصوص مدل های ختم ارتباط بیشتر توضیح می دهیم.



شکل ۴-۶. یک دیاگرام حالت برای الگوی مدیریت ساده اتصال. «گذار از یک حالت» که با قلم ایتالیک نشان داده شده در اثر ورود یک بسته حادث می شود. خطوط توپر توالی حالات برنامه مشتری و خطوط نقطه چین توالی حالات برنامه سرویس دهنده را نشان می دهند.

۱. State Diagram ۲. Transition ۳. State ۴. Event

۵. یعنی از روش Piggybacking (جاسازی Ack درون بسته های داده ارسالی) که در فصل سوم تشریح کردیم استفاده نشده باشد. -م

۳-۱-۶ سوکت های برکلی (Berkeley Socket)

حال اجازه بدهید به اختصار مجموعه دیگری از عملکردهای اولیه و توابع بنیادی لایه انتقال را که با نام «توابع سوکت» در یونیکس برکلی برای پروتکل TCP تعریف شده، بررسی نماییم. از این توابع به طرز گسترده ای برای برنامه نویسی اینترنت استفاده می شود. این توابع در شکل ۵-۶ فهرست شده اند. در یک عبارت نادقیق، این توابع از همان مدل مثال قبلی ما پیروی می کنند با این تفاوت که ویژگیهای بهتر و انعطاف بیشتری دارند. فعلاً در اینجا به جزئیات ساختار TPDU در هر یک از این توابع نمی پردازیم و بررسی آن را تا تشریح TCP در همین فصل به تعویق می اندازیم.

توصیف	نام عملکرد (تابع)
یک نقطه ارتباط پایانی جدید ایجاد می کند.	SOCKET
به سوکت ایجاد شده یک آدرس محلی (شماره) مقید می کند.	BIND
تمایل برنامه کاربردی به پذیرش تقاضاهای اتصال را مشخص نموده و طول صف را معین می کند.	LISTEN
فراخواننده را آنقدر متوقف و منتظر نگاه می دارد تا کسی سعی در ایجاد اتصال کند.	ACCEPT
تلاش جهت ایجاد اتصال بصورت فعال	CONNECT
مقداری داده بر روی اتصال مشخص شده می فرستد.	SEND
مقداری داده از اتصال مشخص شده می خواند.	RECEIVE
ختم اتصال	CLOSE

شکل ۵-۶. توابع اولیه سوکت برای TCP.

در هر برنامه سرویس دهنده، چهار تابع پایه جدول ۵-۶، به ترتیب اجرا می شوند. تابع SOCKET در پروسه کاربردی یک نقطه پایانی (End-Point) تعریف کرده و فضای حافظه لازم را در واحد انتقال (Transport Entity) اختصاص می دهد.^۱ پارامترهای لازم برای فراخوانی این تابع عبارتند از: (۱) قالب آدرس دهی مورد نظر (Addressing Format)، (۲) نوع خدمات مورد انتظار (مثل استریم مطمئن از باینها یا سرویس نامطمئن دیتاگرام) و (۳) پروتکل مورد نظر.

اگر فراخوانی تابع SOCKET موفق باشد یک «اشاره گر فایل» به عنوان مقدار برگشتی بازگردانده می شود تا در فراخوانیهای بعدی مورد استفاده قرار گیرد؛ دقیقاً مشابه با همان کاری که تابع OPEN برای باز کردن یک فایل انجام می دهد.

سوکت هایی که ایجاد می شوند دارای آدرسهای شبکه نیستند.^۲ برای انتساب آدرسها به هر سوکت از تابع پایه BIND استفاده می شود. به محض آنکه سرویس دهنده، آدرسی را به یک سوکت نسبت داد هر مشتری راه دور می تواند اقدام به برقراری اتصال با آن کند.^۳ دلیل آن که با فراخوانی تابع SOCKET، آدرس لازم به آن انتساب داده نمی شود و تابعی مجزا برای آن در نظر گرفته شده، آن است که برخی از پروسه ها در مورد آدرس مورد نظر خود مطمئن هستند (مثلاً برای سالها از یک آدرس استفاده می نمایند و همه مشتریان از این شماره آگاهند) در حالی که برای برخی دیگر از پروسه ها، مقدار این آدرس و ثبات آن اصلاً اهمیت ندارد.

۱. در حقیقت با تابع SOCKET هویت هر یک از طرفین در دو سر خط لوله انتقال مشخص می شود. -م

۲. به عبارت دیگر، هر چند یک نقطه پایانی در پروسه ایجاد می شود ولی هیچ آدرسی که بتوان از راه دور با آن اقدام به برقراری تماس کرد وجود ندارد. -م

۳. به فرآیند انتساب آدرس به یک سوکت عمل «مقیدسازی» سوکت - Binding - گفته می شود. -م

در ادامه، پروسه سرویس دهنده تابع پایه LISTEN را فراخوانی می کند. اجرای این تابع فضای لازم را برای صف بندی تقاضاهای ایجاد اتصال، اختصاص می دهد. بدین ترتیب سرویس دهنده می تواند بطور همزمان با چندین مشتری اتصال ایجاد کند. برخلاف مدل مثال اول، در اینجا فراخوانی تابع LISTEN پروسه سرویس دهنده را بلوکه و معلق نخواهد کرد.

برای آن که پروسه تا دریافت تقاضای ایجاد اتصال، معلق و منتظر بماند سرویس دهنده، تابع ACCEPT را فراخوانی می کند. هر گاه یک TPDU مبنی بر تقاضای ایجاد یک اتصال دریافت شود، واحد انتقال سوکت جدیدی با همان مشخصات سوکت اصلی ایجاد کرده و اشاره گر آن را بر می گرداند. برنامه سرویس دهنده می تواند یک پروسه فرزند (Child Process) یا یک «ریسمان» (Tread) برای آن ایجاد کند تا به سرویس دهی به اتصال جدید مشغول شود. سپس مجدداً به حالت انتظار برگشته تا بتواند تقاضاهای اتصال جدید را بپذیرد. تابع ACCEPT یک اشاره گر معمولی بر می گرداند تا با استفاده از آن بتوان به روش معمولی داده ای خواند یا نوشت. (دقیقاً شبیه به عملیات خواندن و نوشتن از فایل که با اشاره گر مربوطه انجام می شود).

حال اجازه بدهید نگاهی به پروسه سمت مشتری بیندازیم: در اینجا نیز بایستی با فراخوانی تابع پایه SOCKET، یک سوکت ایجاد شود ولیکن نیازی به فراخوانی تابع BIND نیست چرا که در اینجا آدرس پروسه مشتری برای سرویس دهنده اهمیتی ندارد.^۱ سپس با فراخوانی تابع CONNECT، پروسه فراخواننده معلق شده و همان وقت فرآیند ایجاد اتصال آغاز می شود. وقتی این فرآیند تکمیل شود (یعنی وقتی پاسخ مناسب از سرویس دهنده دریافت گردد) پروسه مشتری از حالت تعلیق به در آمده و اتصال مورد نظر ایجاد می شود. در این نقطه هر یک از طرفین این اتصال می توانند با فراخوانی توابع پایه SEND و RECV اقدام به ارسال یا دریافت دوطرفه و همزمان (Full Duplex) داده ها نمایند. در یونیکس حتی می توان از فراخوانیهای سیستمی READ و WRITE به جای SEND و RECV استفاده کرد.

مدل قطع ارتباط در سوکتهای فوق، متقارن است یعنی زمانی که هر دو طرف، تابع CLOSE را اجرا کنند، اتصال ایجاد شده خاتمه خواهد یافت.

۴-۶-۱ مثال از برنامه نویسی سوکت: یک سرویس دهنده اینترنتی فایل

به عنوان مثالی از چگونگی فراخوانی توابع سوکت، کدهای برنامه سرویس دهنده و مشتری را در شکل ۴-۶ مدنظر قرار بدهید. در این مثال یک سرویس دهنده ابتدایی فایل به همراه یک برنامه مشتری (که از آن بهره می گیرد)، ارائه شده است. این قطعه کد از محدودیتهای بی شماری که در زیر تشریح کرده ایم، رنج می برد ولیکن در هر حال می توان کد برنامه سرویس دهنده را کامپایل کرده و آن را بر روی هر سیستم یونیکس متصل به اینترنت اجرا کرد. کد برنامه مشتری را نیز می توان پس از کامپایل، بر روی هر ماشین یونیکس متصل به اینترنت در سراسر دنیا اجرا و از آن استفاده کرد. برنامه مشتری در خط فرمان (به همراه دو آرگومان) اجرا می شود تا یکمک آن هر فایلی را که برنامه سرویس دهنده بدانها دسترسی دارد، دریافت کرد. فایل دریافتی بر روی «خروجی استاندارد» نوشته می شود و بدیهی است که می توان به کمک مفهوم «تغییر مسیر» (Redirection) در یونیکس، آنرا به درون یک فایل یا یک «لوله» (Pipe) هدایت کرد.

اجازه بدهید به کد برنامه سرویس دهنده نگاهی بیندازیم. این برنامه با تعریف چند include file^۲ شروع می شود. سه فایل آخر، در برگیرنده تعاریف مرتبط با اینترنت و ساختمان داده مورد نیاز شبکه است. سپس ثابت

۱. دقت کنید که پروسه مشتری قطعاً دارای آدرس هست ولی فقط مقدار آن مهم نیست و در اینجا انتخاب آن بر عهده لایه انتقال

۲. مفهوم #include را در زبان C، در نظر داشته باشید.

گذاشته شده است. -سم

SERVER-PORT معادل با 12345 تعریف شده است. این عدد کاملاً اختیاری است: اعداد بین ۱۰۲۴ تا ۶۵۵۳۵ (به شرط آن که توسط پروتکل دیگری استفاده نشده باشد) قابل انتخاب است. البته برنامه سرویس دهنده و مشتری باید از شماره مشابهی استفاده کنند. اگر روزی این سرویس دهنده جهانی شود باید به آن یک شماره پورت زیر ۱۰۲۴ انتساب داده شود و در سایت www.iana.com به اطلاع همه برسد!!

دو خط بعدی از برنامه سرویس دهنده، دو ثابت مورد نیاز برنامه را تعریف کرده است. اولین آنها حداکثر طول قطعات ارسالی فایل را مشخص نموده است. دومین ثابت، تعیین می کند که حداکثر چند اتصال معلق و منتظر را می توان حفظ کرد.

پس از تعریف متغیرهای محلی، کد برنامه سرویس دهنده آغاز می شود. در ابتدا یک ساختمان داده با آدرس IP ماشین سرویس دهنده، مقداردهی اولیه می شود. در ادامه، این ساختمان داده به سوکت سرویس دهنده، «مقید» (Bind) خواهد شد. فراخوانی تابع *memset* کل ساختمان داده را با صفر پر می کند. سپس سه دستور انتساب بعدی فیلدهای این ساختمان داده را مقداردهی می نماید. آخرین دستور انتساب، شماره پورت سرویس دهنده را مشخص کرده است. توابع *htonl* و *htons* مقادیر فیلدها را به قالب استاندارد تبدیل می کند تا این برنامه بتواند بدرستی بر روی ماشینهای Big Endian (مثل ماشینهای SPARC) و ماشینهای Little Endian (مثل ماشینهای پتیتوم) اجرا شود.^۱

در ادامه، برنامه سرویس دهنده سوکتی را ایجاد کرده و خطای احتمالی را بررسی می نماید. (هرگونه خطا با $s < 0$ مشخص می شود.) در نسخه واقعی این برنامه، پیامهای خطا می توانند گویاتر و مفصل تر باشند. فراخوانی تابع *setsockopt* بدان جهت نیاز است که بتوان از آن پورت به دفعات استفاده کرد و برنامه سرویس دهنده بتواند به صورت نامحدود اجرا شود. حال به سوکت ایجاد شده آدرس IP و پورت، مقید (Bind) شده و بررسی می شود که آیا این عمل موفق بوده است. آخرین گام در مقداردهی اولیه، فراخوانی تابع *listen* است تا تمایل سرویس دهنده به پذیرش تقاضاهای اتصال را به اطلاع سیستم رسانده و مشخص کند که سیستم فقط حق دارد به تعداد QUEUE_SIZE از متقاضیان اتصال را (در حین پردازش یکی از آنها) پذیرفته و معلق نگه دارد. اگر صف مربوطه پر باشد و تقاضای اتصال جدیدی برسد، نادیده گرفته می شود.

در این نقطه، برنامه سرویس دهنده به حلقه اصلی خود وارد می شود؛ حلقه ای که هرگز از آن خارج نخواهد شد. تنها راه متوقف کردن این برنامه، «کشتن» آن از بیرون (با فرمان kill) است. فراخوانی تابع *accept*، سرویس دهنده را بلوکه می کند تا آنکه یک یا چند مشتری سعی کنند با آن اتصالی را برقرار نمایند. اگر فراخوانی تابع *accept* موفق باشد، یک اشاره گر معمولی فایل، بازگردانده می شود تا به کمک آن بتوان داده ارسال یا دریافت کرد. (به همان نحوی که با اشاره گر فایل می توان درون یک لوله (Pipe) نوشت یا از آن خواند.) ولیکن برخلاف «لوله» در یونیکس که یک طرفه است، سوکتها دوطرفه هستند لذا با در اختیار داشتن آدرس سوکت (یعنی متغیر *sa*) می توان بر روی اتصال ایجاد شده نوشت (معادل ارسال) یا از آن خواند (معادل دریافت).

پس از آن که اتصال ایجاد شد، برنامه سرویس دهنده نام فایل مورد نظر مشتری را از روی آن اتصال می خواند. اگر داده ای دریافت نشده باشد، سرویس دهنده بلوکه شده و منتظر می ماند. پس از دریافت نام فایل مورد نظر مشتری، سرویس دهنده، فایل مربوطه را باز کرده و در حلقه دیگری وارد می شود تا متوالیاً بلوکهای فایل را خوانده و آن را بر روی سوکت ارسال نماید. این حلقه آنقدر ادامه می یابد تا کل فایل منتقل شود. سپس فایل و اتصال متناظر را می بندد و منتظر اتصال بعدی می ماند. این حلقه تا ابد ادامه دارد.

۱. برای آشنایی با تعریف این واژه ها به فصول قبلی مراجعه کنید.

```

/* This page contains a client program that can request a file from the server program
 * on the next page. The server responds by sending the whole file.
 */

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define SERVER_PORT 12345          /* arbitrary, but client & server must agree */
#define BUF_SIZE 4096             /* block transfer size */

int main(int argc, char **argv)
{
    int c, s, bytes;
    char buf[BUF_SIZE];             /* buffer for incoming file */
    struct hostent *h;              /* info about server */
    struct sockaddr_in channel;     /* holds IP address */

    if (argc != 3) fatal("Usage: client server-name file-name");
    h = gethostbyname(argv[1]);     /* look up host's IP address */
    if (!h) fatal("gethostbyname failed");

    s = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (s < 0) fatal("socket");
    memset(&channel, 0, sizeof(channel));
    channel.sin_family = AF_INET;
    memcpy(&channel.sin_addr.s_addr, h->h_addr, h->h_length);
    channel.sin_port = htons(SERVER_PORT);

    c = connect(s, (struct sockaddr *) &channel, sizeof(channel));
    if (c < 0) fatal("connect failed");

    /* Connection is now established. Send file name including 0 byte at end. */
    write(s, argv[2], strlen(argv[2])+1);

    /* Go get the file and write it to standard output. */
    while (1) {
        bytes = read(s, buf, BUF_SIZE); /* read from socket */
        if (bytes <= 0) exit(0);         /* check for end of file */
        write(1, buf, bytes);           /* write to standard output */
    }

    fatal(char *string)
    {
        printf("%s\n", string);
        exit(1);
    }
}

```

شکل ۶-۶. کد برنامه مشتری با بهره گیری از سوکتها. کد برنامه سرورس در صفحه بعدی آمده است.

```

#include <sys/types.h> /* This is the server code */
#include <sys/fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define SERVER_PORT 12345 /* arbitrary, but client & server must agree */
#define BUF_SIZE 4096 /* block transfer size */
#define QUEUE_SIZE 10

int main(int argc, char *argv[])
{
    int s, b, l, fd, sa, bytes, on = 1;
    char buf[BUF_SIZE]; /* buffer for outgoing file */
    struct sockaddr_in channel; /* holds IP address */

    /* Build address structure to bind to socket. */
    memset(&channel, 0, sizeof(channel)); /* zero channel */
    channel.sin_family = AF_INET;
    channel.sin_addr.s_addr = htonl(INADDR_ANY);
    channel.sin_port = htons(SERVER_PORT);

    /* Passive open. Wait for connection. */
    s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP); /* create socket */
    if (s < 0) fatal("socket failed");
    setsockopt(s, SOL_SOCKET, SO_REUSEADDR, (char *) &on, sizeof(on));

    b = bind(s, (struct sockaddr *) &channel, sizeof(channel));
    if (b < 0) fatal("bind failed");

    l = listen(s, QUEUE_SIZE); /* specify queue size */
    if (l < 0) fatal("listen failed");

    /* Socket is now set up and bound. Wait for connection and process it. */
    while (1) {
        sa = accept(s, 0, 0); /* block for connection request */
        if (sa < 0) fatal("accept failed");

        read(sa, buf, BUF_SIZE); /* read file name from socket */

        /* Get and return the file. */
        fd = open(buf, O_RDONLY); /* open the file to be sent back */
        if (fd < 0) fatal("open failed");

        while (1) {
            bytes = read(fd, buf, BUF_SIZE); /* read from file */
            if (bytes <= 0) break; /* check for end of file */
            write(sa, buf, bytes); /* write bytes to socket */
        }
        close(fd); /* close file */
        close(sa); /* close connection */
    }
}

```

حال اجازه بدهید به کد برنامه مشتری نگاهی بیندازیم. برای آشنایی با عملکرد این برنامه، لازم است با روش فراخوانی و اجرای آن در خط فرمان، آشنا شوید. با فرض آن که نام این برنامه client انتخاب شده باشد، فراخوانی آن همانند زیر است:

```
client flits.cs.vu.nl /usr/tom/filename > f
```

فراخوانی برنامه فوق زمانی موفق است که برنامه سرویس دهنده از قبل بر روی ماشین flits.cs.vu.nl اجرا شده و همچنین فایلی با مشخصات /usr/tom/filename موجود و دسترسی برنامه سرویس دهنده به آن مجاز باشد. اگر فراخوانی برنامه با موفقیت انجام شود، فایل فوق الذکر از طریق اینترنت منتقل و درون f نوشته می شود؛ پس از آن برنامه client به پایان می رسد. از آنجایی که اجرای برنامه سرویس دهنده پس از انتقال یک فایل ادامه خواهد یافت فلذا برنامه مشتری را می توان بارها اجرا و فایل های دیگری را دریافت کرد.

در ابتدای برنامه مشتری نیز برخی از تعاریف اولیه آمده است. اجرای واقعی برنامه با بررسی تعداد آرگومانها آغاز می شود. ($argc=3$ به معنای سه آرگومان، شامل نام برنامه به همراه دو آرگومان دیگر است.) دقت کنید که $argv[1]$ حاوی نام ماشین سرویس دهنده (مثل flits.cs.vu.nl) است و بعداً به کمک تابع سیستمی `gethostbyname` به آدرس IP ترجمه و تبدیل خواهد شد. این تابع سیستمی برای تبدیل نام از سیستم DNS بهره می گیرد. DNS را در فصل هفتم تشریح خواهیم کرد.

در ادامه یک سوکت ایجاد و مقداردهی اولیه می شود. سپس برنامه مشتری تلاش می کند با استفاده از تابع `connect` یک اتصال TCP با سرویس دهنده برقرار نماید. اگر سرویس دهنده بر روی ماشین با نام مورد نظر، اجرا شده باشد و همچنین به `SERVER_PORT` گوش بدهد و فضای کافی در صف `listen` وجود داشته باشد، یک اتصال برقرار می شود. از طریق این اتصال، برنامه مشتری نام فایل مورد نظر خود را برای سرویس دهنده ارسال می کند. این کار با تابع `write` انجام می شود. پارامتر آخر در تابع `write`، تعداد بایت هایی را که باید ارسال شوند، مشخص می کند؛ از آنجایی که نام فایل به همراه کاراکتر '\0' ارسال می شود لذا تعداد بایت ها یکی بیشتر از طول واقعی رشته حاوی نام در نظر گرفته شده است.

در اینجا برنامه مشتری به یک حلقه وارد شده و فایل را بلوک به بلوک از سوکت خوانده و در خروجی استاندارد کپی می نماید. پس از این کار، برنامه به اتمام می رسد.

عملکرد زیر برنامه `fatal` (در صورت فراخوانی) آنست که، یک پیام خطا بر روی خروجی چاپ کرده و از برنامه خارج شود. دقت کنید که برنامه سرویس دهنده نیز به همین زیر برنامه احتیاج دارد ولی به دلیل کمبود جا در صفحه، از تعریف آن چشمپوشی شده است. از آنجایی که برنامه سرویس دهنده و برنامه مشتری بطور جداگانه کامپایل و بر روی کامپیوترهای متفاوتی اجرا می شوند لذا نمی توانند از کد زیر برنامه `fatal` به صورت اشتراکی بهره بگیرند. این دو برنامه را (به همراه برخی دیگر از مفاد مرتبط با کتاب) می توانید از وب سایت <http://www.prenhall.com/tanenbaum> بدست بیاورید. در این سایت بر روی تصویر روی جلد کتاب کلیک کنید؛ در صفحه ای که ظاهر می شود می توانید برنامه های فوق را دریافت و آن را بر روی هر سیستم سازگار با یونیکس (مثل سولاریس، BSD و لینوکس) به نحو زیر کامپایل کنید:

```
cc -o client client.c -lsocket -lnst
cc -o server server.c -lsocket -lnst
```

برنامه سرویس دهنده با درج نام برنامه بر روی خط فرمان اجرا می شود:

```
server
```

به نحوی که اشاره شد برنامه مشتری به دو آرگومان نیاز دارد. نسخه Windows این دو برنامه نیز در وب سایت

فوق در دسترس می باشد.

پادآوری می کنیم که این سرویس دهنده، از بسیاری جهات تکامل یافته نیست. پردازش و مدیریت خطای آن بسیار ضعیف و گزارشهای خطا ناچیز است. هیچ تمهیدی در خصوص امنیت در آن اندیشیده نشده و فراخوانیهای سیستمی یونیکس در آن، «ویژگی استقلال از محیط اجرا»^۱ را مخدوش کرده است. در ضمن این برنامه براساس فرضیاتی نوشته شده است که اصولاً اشتباه هستند؛ به عنوان مثال فرض شده که نام فایل در بافر جا می گیرد و به صورت یکجا ارسال می شود. از آنجایی که این برنامه تمام تقاضاها را صرفاً به صورت ترتیبی و پشت سرهم پاسخ می دهد (چرا که فقط یک «ریسمان» Thread دارد)، فلذا کارایی آن بشدت پایین است. با تمام این کاستی ها، برنامه فوق کامل بوده و می تواند به صورت یک سرویس دهنده فایل ایفای نقش کند. خوانندگان گرامی رای به توسعه این برنامه دعوت می نمایم. برای کسب آگاهی بیشتر در خصوص برنامه نویسی سوکت به مرجع (Stevens, 1997) مراجعه نمایید.

۲-۶ مؤلفه های هر پروتکل انتقال

خدمات انتقال توسط «پروتکل های انتقال» پیاده سازی شده و از آنها بین دو «واحد انتقال» (Transport Entity) استفاده می شود. از بسیاری جهات، پروتکل های انتقال با پروتکل های لایه پیوند داده که در فصل سوم تشریح شد، شبیه هستند. هر دو با مسئله نظارت بر خطا، حفظ ترتیب داده ها و کنترل جریان و مواردی از این قبیل، سر و کار دارند.

در عین حال، تفاوت های چشمگیری بین این دو وجود دارد. این تفاوتها عمدتاً ناشی از محیط و بستری است که این دو پروتکل در آن کار می کنند. به شکل ۶-۷ دقت کنید: در لایه پیوند داده دو مسیر یاب از طریق یک کانال فیزیکی مستقیماً با یکدیگر به تبادل داده می پردازند در حالی که در لایه انتقال این کانال فیزیکی جای خود را به یک زیر شبکه کامل [با تعدادی مسیر یاب و کانال مخابراتی] می دهد. این تفاوت مستلزم پیش بینی تمهیدات بسیار مهمی است که در این فصل آنها را بررسی خواهیم کرد.



شکل ۶-۷. (الف) محیط لایه پیوند داده (ب) محیط لایه انتقال.

اولین تفاوت آن است که در لایه پیوند داده، مسیر یاب نیاز ندارد که بداند با کدام مسیر یاب در حال محاوره است، چرا که هر یک از خطوط خروجی به صورت یکتا، صرفاً یک مسیر یاب خاص را مشخص می کنند.^۲ برعکس در لایه انتقال، نیاز است که آدرس دقیق مقصد مشخص باشد.

^۱ Platform Independency.

^۲ به عبارتی دیگر، چون خطوط بین دو مسیر یاب مستقیم و نقطه به نقطه هستند، می توان با آن مستقیماً و بدون آن که آدرس طرف مقابل مشخص باشد، مبادله داده کرد. -م

مورد اختلاف دیگر آن است که فرآیند ایجاد یک «اتصال» بر روی سیم (نشان داده شده در شکل ۶-۷-الف) ساده است: طرف مقابل همیشه هست (مگر آنکه از کار بیفتد یا رخدادی جدی حادث شود که در این صورت هیچ کاری نمی‌توان انجام داد) در لایه انتقال به نحوی که خواهیم دید، ایجاد اتصال فرآیندی پیچیده است. یکی دیگر از تفاوت‌های بسیار آزاردهنده بین لایه پیوند داده و لایه انتقال آن است که زیر شبکه مستعد نگهداری و معطل کردن بسته‌ها در حافظه است. وقتی یک مسیر یاب فریمی را بر روی لینک خود ارسال می‌کند بالاخره می‌رسد یا به دلیل خطا از دست می‌رود ولی این فریم قادر نیست مدتی سرگردان باشد، در گوشه‌ای از دنیا کور و گم شود و سی‌ثانی بعد به ناگاه در مقصد ظاهر گردد. اگر زیر شبکه‌ای از روش دیتاگرام و مسیریابی پویا بهره گرفته باشد، احتمال آن که بسته‌ای در آن ذخیره و چندین ثانیه بعد تحویل مقصد شود قابل چشم‌پوشی نیست. تبعات ناشی از استعداد زیر شبکه در ذخیره و تعلیق بسته‌ها، بسیار مشکل‌آفرین و برای رفع آنها به پروتکل‌های خاص نیازمند است.

تفاوت نهایی بین لایه پیوند داده و لایه انتقال بیشتر در میزان انتظارات است تا آن که در نوع انتظارات باشد: بافرسازی و کنترل جریان در هر دو لایه الزامی است ولیکن وجود تعداد بسیار زیاد و متغیر «اتصال» (Connection) در لایه انتقال به راهکارها و مکانیزم‌های متفاوتی (در مقایسه با راهکارهای اتخاذ شده در لایه پیوند داده) احتیاج دارد. در فصل سوم دیدیم که برخی از پروتکل‌های لایه پیوند داده تعداد ثابت و مشخصی بافر برای هر خط اختصاص می‌دهند و هر وقت که فریمی از راه برسد، بافر مورد نیاز موجود خواهد بود. ولیکن در لایه انتقال، ایده اختصاص تعداد ثابتی بافر برای هر اتصال (به دلیل آن که تعداد اتصالات می‌تواند بسیار زیاد و متغیر باشد) چندان عملی نیست. در بخش‌های آتی کلیه این موارد و مواردی از این قبیل بحث و بررسی خواهند شد.

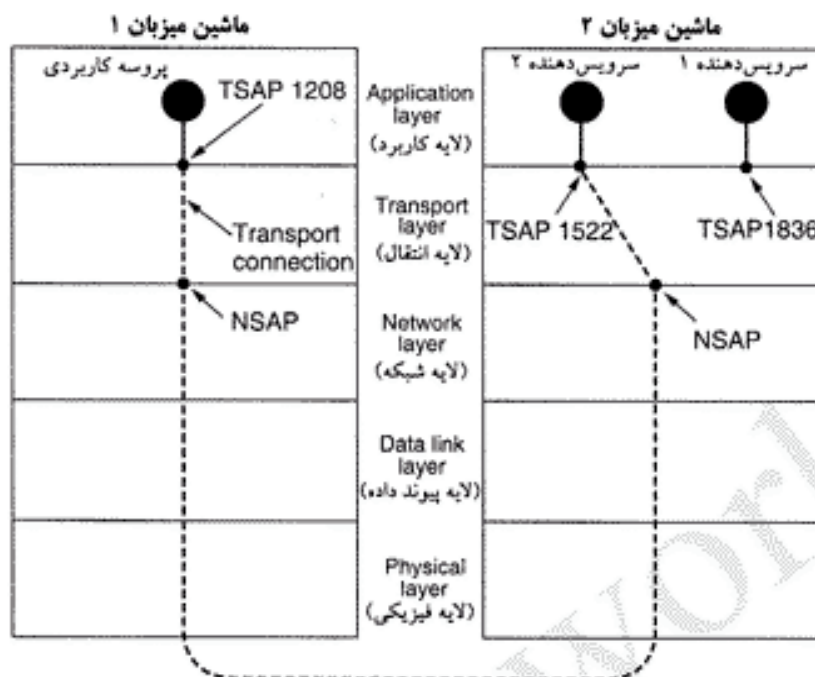
۱-۲-۶ آدرس‌دهی

وقتی یک پروسه کاربردی (برنامه کاربر) می‌خواهد با یک پروسه کاربردی راه دور، «اتصال» ایجاد کند باید پروسه مورد نظرش را دقیقاً مشخص نماید. (انتقال بدون اتصال نیز با همین مسئله مواجه است: هر پیام برای چه پروسه‌ای ارسال می‌شود؟) روشی که عموماً مورد استفاده قرار می‌گیرد تعریف «آدرس انتقال» (Transport Address) برای پروسه‌هایی است که برای دریافت تقاضای اتصال در حال انتظار (شنود Listening) هستند. [یعنی پروسه‌های در حال شنود توسط یک آدرس یکتا، مشخص می‌شوند.] در اینترنت، به این نقاط پایانی «پورت» گفته می‌شود. در شبکه ATM این نقاط، AAL_SAP نام گرفته‌اند. برای آن که عمومیت مطالب تحت تأثیر این واژه‌ها قرار نگیرد ما از واژه کلی TSAP^۱ استفاده خواهیم کرد. (مشابه با نقاط پایانی در لایه شبکه - یعنی آدرسهای لایه شبکه - که اصطلاحاً NSAP نامیده می‌شد.)

شکل ۶-۸ ارتباط بین NSAP، TSAP و اتصالات لایه انتقال را به تصویر کشیده است. پروسه‌های کاربردی اعم از سرویس‌دهنده یا مشتری می‌توانند خود را به یک TSAP متصل نموده تا بتوانند با یک «TSAP راه دور»^۲ تماس (اتصال) برقرار کنند. به گونه‌ای که در این شکل نشان داده شده هر اتصال از طریق NSAP برقرار می‌شود.^۳ هدف از تعریف TSAP آن است که در شبکه‌ای عظیم که هر کامپیوتر متصل به آن عموماً با یک NSAP واحد شناسایی می‌شود، بتوان نقاط پایانی متعددی را که همگی یک NSAP مشترک دارند، از هم تمیز داد.

۱. Transport Service Access Point ۲. Remote TSAP

۳. NSAP یک ماشین واحد را در کل شبکه و TSAP یک پروسه را بر روی آن ماشین مشخص می‌کند. ترکیب «TSAP : NSAP» به پروسه‌ها هویت جهانی و منحصر به فرد می‌دهد. -م



شکل ۶-۸ مفهوم TSAP، NSAP و اتصال.

سناریویی برای ایجاد یک اتصال در لایه انتقال، می تواند به ترتیب زیر باشد:

۱. یک پروسه سرویس دهنده (مثلاً پروسه سرویس دهنده Time of Day که تاریخ و ساعت را اعلام می کند) بر روی ماشین ۲ خودش را به TSAP 1522 متصل کرده و منتظر دریافت تقاضای تماس دیگران می ماند. چگونگی وصل شدن یک پروسه به یک TSAP صرفاً به سیستم عامل محلی هر ماشین بستگی دارد و در محدوده مدل استاندارد شبکه قرار نمی گیرد. به عنوان مثال می توان تابع سیستمی LISTEN را فراخوانی کرد.
۲. پروسه کاربردی بر روی ماشین ۱ می خواهد زمان و تاریخ روز را بداند لذا تقاضای CONNECT را صادر کرده و در آن هویت مبدا (یعنی خودش) را با TSAP 1028 و هویت مقصد (یعنی سرویس دهنده) را با TSAP 1522 مشخص می کند. این کار موجب می شود که یک اتصال بین پروسه کاربردی روی ماشین ۱ با پروسه سرویس دهنده روی ماشین ۲ ایجاد شود.
۳. در اینجا پروسه کاربردی تقاضای «زمان و تاریخ» را ارسال می کند.
۴. در پاسخ پروسه سرویس دهنده، زمان فعلی را اعلام می کند.
۵. اتصال (تماس) قطع می شود.

دقت کنید که ممکن است بر روی ماشین میزبان ۲، سرویس دهنده های دیگری نیز اجرا و به TSAP متفاوتی متصل شده باشند؛ همه این سرویس دهنده ها NSAP مشابهی دارند.^۱ تصویری که ارائه شد مفید و گویاست ولیکن نکته کوچکی را از قلم انداخته ایم: پروسه کاربردی ۱ از کجا بداند که سرویس دهنده زمان (Time-of-Day Server) به TSAP 1522 متصل شده است؟ یک راه آن است که این سرویس دهنده، سالها از TSAP 1522 استفاده کرده باشد و کاربران شبکه به تدریج از این شماره آگاه شده باشند. در این رویکرد، سرویس دهنده ها TSAP ثابت و پایداری خواهند داشت و می توان آنها را درون فایلی در یک

۱. در اینترنت NSAP همان آدرس IP است. -م

محل شناخته شده مثل فایل *etc/services* ذخیره کرد. (در این فایل، فهرست تمام سرویس دهنده های استاندارد و مشهور به همراه شماره پورت هایی که بدان متصل هستند، درج شده است.)

آدرسهای ثابت و پایدار TSAP فقط برای سرویس دهنده های مهم و کلیدی (مثل سرویس دهنده وب) مفید خواهد بود، در حالی که پروسه های کاربری عموماً می خواهند با پروسه هایی محاوره کنند که به صورت موقتی اجرا شده اند و هیچ آدرس TSAP که از قبل مشخص باشد، ندارند. مضاف بر این، سرویس دهنده های متنوعی وجود دارند که اغلب آنها کاربرد چندانی ندارند و به صورت موردی اجرا می شوند، لذا اختصاص آدرس TSAP ثابت به آنها و فعال نگه داشتنشان بی فایده است. کوتاه سخن آن که به روش بهتری نیاز است!

یک روش مناسب در شکل ۶-۹ (به صورت ساده و خلاصه) به تصویر کشیده شده است. این روش به نام «پروتکل اتصال اولیه» (Initial Connection Protocol) مشهور است. به جای آن که هر سرویس دهنده به یک TSAP شناخته شده گوش بدهد، ماشینی که علاقمند به سرویس دهی به کاربران راه دور است پروسه خاصی به نام Process Server را اجرا می کند. Process Server، وکیل تمام سرویس دهنده هایی است که کمتر مورد استفاده قرار می گیرند. این پروسه بطور همزمان به مجموعه ای از پورتها گوش داده و منتظر تقاضای برقراری ارتباط می ماند. کاربران احتمالی، با ارسال تقاضای CONNECT، شماره TSAP سرویس دهنده مورد نظرشان را تعیین می کنند. اگر هیچ پروسه ای به TSAP مورد نظر آنها متصل نشده و در حال شنود نباشد، لاجرم اتصال آنها مطابق با شکل ۶-۹ الف با Process Server برقرار می شود.

وقتی تقاضای برقراری اتصال دریافت شد، Process Server ضمن تشخیص سرویس دهنده مورد تقاضا آن را راه اندازی و اجرا می کند و اتصال ایجاد شده بین خودش و کاربر را به پروسه سرویس دهنده تحویل می دهد. در این لحظه سرویس دهنده تازه اجرا شده، مشغول انجام عملیات درخواستی می شود در حالی که Process Server به سر کار اصلیش یعنی گوش دادن به تقاضاهای جدید برمی گردد.^۱ (شکل ۶-۹ ب)

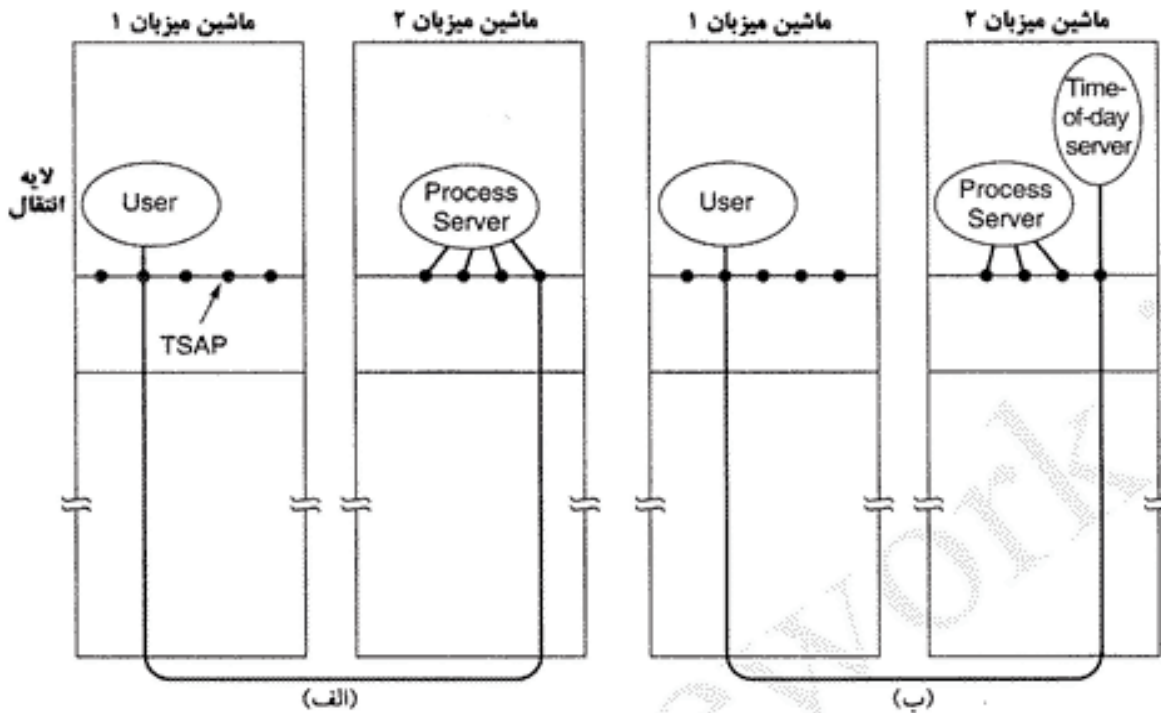
هر چند «پروتکل اتصال اولیه» برای سرویس دهنده هایی که فقط برحسب تقاضا و به صورت موردی اجرا می شوند بسیار مفید و کارآمد است ولی گاهی اوقات سرویس دهنده هایی هستند که مستقل از Process Server کار می کنند. به عنوان مثال یک سرویس دهنده فایل باید به صورت دائمی بر روی یک ماشین خاص اجرا شده باشد و نمی توان آن را برحسب تقاضا و موردی اجرا کرد.

برای حل این مسئله، از الگوی دیگری استفاده می شود. در این الگو یک پروسه خاص دیگر به نام Name Server (سرویس دهنده نام) یا Directory Server (سرویس دهنده دایرکتوری) بر روی ماشین اجرا می شود. برای یافتن آدرس TSAP معادل با نام یک سرویس دهنده (مثل سرویس دهنده Time-of-Day)، کاربر ابتدا اتصالی با «سرویس دهنده نام» که همیشه به یک شماره پورت مشهور و شناخته شده گوش می دهد، برقرار می کند. سپس با ارسال پیامی، نام سرویس دهنده مورد نظر خود را سوال کرده و سرویس دهنده نام در پاسخ، آدرس TSAP آن را بر می گرداند. در اینجا کاربر اتصال خود را با سرویس دهنده نام خاتمه داده و اتصال جدیدی با سرویس دهنده مورد نظر خود برقرار می نماید.^۲

در این مدل هر گاه سرویس دهنده جدیدی خلق شود بایستی خودش را در سرویس دهنده نام، ثبت کند و نام سرویس (عموماً به صورت یک رشته ASCII) و همچنین TSAP خود را مشخص نماید. سرویس دهنده نام این

۱. در حقیقت به جای آن که مثلاً n پروسه سرویس دهنده به طور همزمان اجرا شود و به پورت های مورد نظر خود گوش بدهند یک پروسه به نیابت از همه آنها و به طور همزمان به n پورت گوش می دهد و در صورت برقراری هر گونه اتصال، پروسه متناظر را اجرا کرده و تماس برقرار شده را به سوی او بر می گرداند. -م

۲. این سرویس دهنده نام را با سرویس دهنده DNS اشتباه نگیرید.



شکل ۶-۹. چگونگی ایجاد اتصال توسط یک پروسه کاربری بر روی ماشین ۱ با سرویس دهنده Time of Day (سرویس دهنده تاریخ و ساعت) بر روی ماشین ۲.

اطلاعات را در پایگاه اطلاعات داخلی خود درج می کند تا در پرس و جوهای بعدی پاسخها را بداند. عملکرد سرویس دهنده نام شبیه به عملکرد اپراتورهای راهنما در سیستم تلفن (۱۱۸) است: اسامی را به شماره، ترجمه می نماید. اینجا نیز مثل سیستم تلفن دانستن آدرس TSAP سرویس دهنده نام الزامی است. این شماره حداقل اطلاعاتی است که هر کسی باید بداند. اگر شما شماره اپراتور راهنمای تلفن را ندانید نمی توانید برای جستجوی شماره دیگران با این اپراتور تماس بگیرید. اگر فکر می کنید شماره اپراتور راهنما بدیهی است (مثلاً ۱۱۸)، سعی کنید آن را برای یک کشور خارجی نیز امتحان نمائید!

۲-۲-۶ برقراری اتصال (Connection Establishment)

برقراری یک اتصال ساده به نظر می رسد ولیکن حقیقتاً بسیار پیچیده است. در نگاه اول شاید اینگونه به نظر برسد که کافی است «واحد انتقال» (Transport Entity) در یک طرف، بسته تقاضای CONNECTION REQUEST TPDU را به سوی مقصد بفرستد و منتظر پاسخ CONNECTION ACCEPTED بماند. اگر شبکه مستعد از بین بردن بسته ها، ذخیره و تعلیق آنها یا تولید بسته های تکراری (Duplicate) باشد، در برقراری یک اتصال مطمئن، مشکلات جدی رخ می دهد و موجب پیچیدگیهای فراوان در پروتکل های لایه انتقال می شود.

زیر شبکه ای را مجسم کنید که با ازدحام مواجه شده و پیامهای تصدیق دریافت بسته ها (یعنی پیامهای Ack) سر موعد مقرر بر نمی گردند و هر بسته به دلیل اتمام مهلت، دو یا سه بار ارسال می شود. در ضمن فرض کنید که زیر شبکه در درون از روش دیتاگرام بهره گرفته و بسته ها از مسیرهای متفاوتی به سوی مقصد می روند. ممکن است برخی از بسته ها در شلوغی ترافیک زیر شبکه گیر افتاده و تحویل آن به مقصد، مدتی طول بکشد. (یعنی در

زیر شبکه ذخیره شده و با تأخیر از آن بیرون بیاید.

بدترین کابوس ممکن بدین نحو اتفاق می‌افتد: کاربری یک اتصال با سرویس دهنده بانک خود برقرار کرده و با ارسال پیامی از آن می‌خواهد که مقدار قابل توجهی پول به حساب یک شخص نه چندان امین! واریز نماید و سپس اتصال را قطع می‌کند. متأسفانه تمام بسته‌ها در این سناریو به صورت تکراری تولید و در زیر شبکه ذخیره می‌شوند.^۱ پس از آنکه اتصال قطع شد بسته‌های معلق از زیر شبکه خارج شده و به ترتیب تحویل مقصد می‌شود یعنی به همان ترتیب قبل با بانک یک اتصال برقرار می‌شود، مجدداً تقاضای انتقال پول انجام می‌گیرد و اتصال قطع می‌گردد. بانک، راهی برای تشخیص تکراری بودن این بسته‌ها ندارد و طبقاً باید فرض را بر آن بگذارد که تقاضاها جدید هستند و انتقال پول را انجام بدهد! تا پایان این بخش به بررسی مسئله بروز بسته‌های تکراری در اثر تأخیر خواهیم پرداخت و بر روی الگوریتمهای برقراری مطمئن اتصال (به نحوی که کابوس فوق هرگز اتفاق نیفتد!) تأکید ویژه خواهیم داشت.

معمای این مسئله در وجود بسته‌هایی تکراری نهفته است که با تأخیر دریافت می‌شوند. به روشهای مختلفی می‌توان به این بسته‌ها حمله کرد و آنها را نابود کرد ولی عملکرد هیچکدام صد در صد مطلوب نیست. یکی از این روشها استفاده از آدرسهای انتقال متغیر با زمان است.^۲ در این رویکرد هر گاه به آدرس انتقال [یعنی آدرس TSAP] نیاز شد، یک شماره جدید تولید می‌شود. وقتی اتصالی قطع شد، آن آدرس کنار گذاشته شده و دیگر از آن استفاده نمی‌شود. این استراتژی موجب می‌شود که مدل مبتنی بر Process Server در شکل ۶-۹، ناممکن و غیر عملی باشد.

رویکرد دیگر آن است که به هر اتصال یک «شناسه اتصال» (Connection Identifier) (یا به عبارتی یک شماره ترتیب که به ازای ایجاد اتصال جدید یک واحد افزایش می‌یابد) به انتخاب تماس گیرنده، منسوب شود. این شناسه در هر TPDU درج و ارسال می‌شود. پس از آن که یک اتصال قطع شد، «واحدهای انتقال» (Transport Entities) در دو طرف، شناسه این اتصال را در جدول شناسه‌های قدیمی درج می‌نمایند. در آینده وقتی تقاضای جدیدی مبنی بر برقراری یک اتصال جدید از راه می‌رسد، ابتدا باید به کمک این جدول بررسی شود که مبادا متعلق به اتصالی باشد که قبلاً قطع شده است.

متأسفانه این روش نیز یک اشکال اساسی دارد: هر یک از «واحدهای انتقال» موظفند پیشینه این شناسه‌ها را برای همیشه در جدول خود نگاه دارند. اگر ماشینی از کار بیفتد و حافظه خود را از دست بدهد دیگر نمی‌تواند تشخیص بدهد که کدامیک شناسه‌های اتصال تکراری هستند.

به جای اینها، باید از راه دیگری وارد شویم: به جای آن که اجازه بدهیم بسته‌ها در زیر شبکه عمر جاودان داشته باشند بایستی مکانیزمی اتخاذ کنیم که بسته‌های با عمر زیاد و سرگردان نابود شوند. اگر مطمئن شویم که بسته‌ها هیچگاه بیش از یک مدت معین عمر نمی‌کنند، مشکل فوق‌الذکر قابل کنترل خواهد شد.

طول عمر بسته‌ها را می‌توان با تکنیکهای زیر در حد مشخصی محدود کرد:

۱. طراحی محدود شده زیر شبکه (Restricted Subnet Design)

۲. درج شمارنده گام در هر بسته (Hop Counter)

۳. درج مهر زمان در هر بسته (Timestamp)

۱. چرا که مثلاً هر بسته پس از ارسال در زیر شبکه بیش از حد معطل شده و مهلت فرستنده به سرآمده و یک نسخه دیگر از آنرا فرستاده است؛ غافل از آنکه هر دو بسته تکراری نهایتاً با تأخیر به مقصد خواهند رسید. -م
۲. عبارت دیگر آدرس TSAP برنامه مشتری در هر بار ایجاد اتصال عوض شود. -م

اولین روش، متضمن استفاده از هر راهکاری است که از چرخیدن بسته‌ها در یک حلقه جلوگیری کرده و همچنین تأخیر ناشی از ازدحام را بر روی طولانی‌ترین مسیر، در حد معینی محدود نگه دارد. دومین روش، مستلزم آن است که در هر بسته یک شمارنده گام قرار داده شده و در مبدأ یک مقدار اولیه مناسب به آن منتسب گردد و به ازای عبور از هر مسیریاب یک واحد از آن کم شود. پروتکل لایه شبکه بسته‌هایی را که مقدار این شمارنده در آنها به صفر رسیده حذف می‌نماید. سومین روش نیازمند آن است که هر بسته، زمان ایجاد خود را با خود حمل نماید و بر این اساس مسیریابها طبق توافق، بسته‌هایی که بیش از یک مدت معین قدیمی شده‌اند را حذف کنند. استفاده از این روش مستلزم آن است که ساعت تمام مسیریابها به دقت با هم تنظیم شده باشد ولیکن این خودش کار ساده‌ای نیست مگر آن که عمل تنظیم ساعتها از بیرون شبکه انجام شود. (مثلاً به کمک سیستم ماهواره‌ای GPS یا ایستگاههای رادیویی خاص تا ساعت دقیق را به همه اعلام کند).

در عمل نه تنها لازم است که تضمین کنیم بسته‌های قدیمی نابود شده‌اند بلکه باید تمام بسته‌های اعلام وصول آنها (Ack's) نیز از بین رفته باشد. با این استدلال، زمانی بنام T را معرفی می‌کنیم که مقدار آن چند برابر حداکثر طول عمر واقعی بسته‌هاست. (چند برابر بودن، به پروتکل مورد استفاده بستگی دارد). اگر پس از ارسال یک بسته به اندازه زمان T صبر کنیم، می‌توانیم مطمئن شویم که نه تنها تمام آثار آن بسته از زیر شبکه پاک شده بلکه حتی پیامهای Ack آن نیز از بین رفته است.

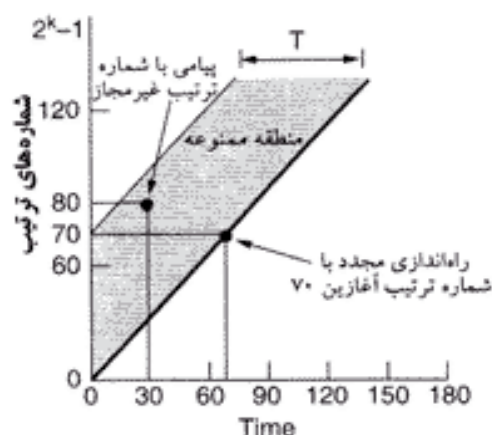
با محدود شدن طول عمر بسته‌ها، می‌توان روشی مطمئن و بی‌خطا برای برقراری اتصال ابداع کرد. روشی را که در ادامه معرفی می‌نمائیم، پیشنهاد تاملینسون (Tomlinson/۱۹۷۵) است. این روش مشکل مذکور را حل می‌کند ولی پیچیدگیهای خاص خود را دارد. این روش بعداً توسط دو نفر به نامهای Dalal و Sunshine (۱۹۷۸) بهینه شد. گونه‌های متفاوتی از این روش در عمل مورد استفاده قرار گرفته است (مثلاً در TCP).

برای آن که مشکل از کار افتادن ماشین و از دست رفتن محتوای حافظه رفع شود، تاملینسون پیشنهاد کرد که در هر ماشین یک ساعت تعبیه شود. لازم نیست که ساعت ماشینهای مختلف با یکدیگر تنظیم شده باشند. فرض بر آن است که این ساعت همانند یک شمارنده باینری عمل می‌کند که در فواصل زمانی مشخص یک واحد افزایش می‌یابد. تعداد بیت‌های شمارنده ساعت باید بزرگتر یا مساوی با تعداد بیت‌های شماره ترتیب بسته‌ها باشد. مورد آخر و مهمتر از همه آنکه ساعت مذکور باید حتی وقتی ماشین از کار می‌افتد یا خاموش می‌شود، کار کند.

ایده اصلی این روش آن است که مطمئن شویم در یک زمان، دو TPDU با شماره مشابه تولید و ارسال نمی‌شود. برای ایجاد یک اتصال، از k بیت کم ارزش ساعت به عنوان شماره ترتیب اولیه استفاده می‌شود. (فیلد شماره ترتیب بسته‌ها نیز k بیتی است). بنابراین برخلاف پروتکل‌های معرفی شده در فصل ۳، در هر اتصال، TPDUها از شماره ترتیب متفاوتی استفاده می‌کنند. فضای در نظر گرفته شده برای شماره ترتیب باید آنقدر بزرگ باشد که در صورت برگشت مقدار این شماره به صفر^۱، TPDUهای قدیمی با همین شماره‌ها، مدتها پیش از بین رفته باشند. رابطه خطی بین «زمان» و «شماره ترتیب اولیه» در شکل ۶-۱۰ نشان داده شده است.

به محض آن که واحدهای انتقال (Transport Entity) در دو طرف، بر روی شماره ترتیب اولیه به توافق رسیدند، می‌توان از هر پروتکلی نظیر «پروتکل پنجره لغزان» (Sliding Window Protocol) برای کنترل جریان استفاده کرد. در دنیای واقعی، منحنی شماره ترتیب (که در شکل با خطوط پررنگ نشان داده شده) خطی نیست بلکه حالت پلکانی دارد زیرا شمارش ساعت به صورت گسسته (مثلاً در هر هزارم ثانیه) صورت می‌گیرد. برای سادگی از این جزئیات صرف‌نظر خواهیم کرد.

۱. به برگشت یک شمارنده به صفر پدیده Wrapping Around گفته می‌شود. -م



شکل ۶-۱۰. (الف) شماره ترتیب TPDUs نباید به منطقه ممنوعه وارد شود. (ب) مشکل سنکرون سازی مجدد شماره ها

مشکل زمانی بروز می کند که ماشین میزبان از کار بیفتد: وقتی از نو شروع به کار می کند، «واحد انتقال» از شماره ترتیب قبلی خود مطلع نیست. یک راه حل آن است که «واحد انتقال» پس از راه اندازی مجدد مدت T ثانیه بیکار بماند تا تمام TPDUs قدیمی از بین بروند. ولی در شبکه های بزرگ و پیچیده، مقدار T می تواند بسیار بزرگ باشد و بدین ترتیب استراتژی فوق چندان جالب و مفید نیست.

برای آن که نیازی به این زمان مرده نباشد باید محدودیت جدیدی بر روی شماره های ترتیب گذاشته شود. با یک مثال به توضیح محدودیت جدید می پردازیم: فرض کنید که T (یعنی حداکثر طول عمر بسته ها) ۶۰ ثانیه باشد و ساعت سیستم در هر ثانیه یک تیک بزند. به نحوی که در شکل ۶-۱۰-الف با خط تیره رنگ نشان داده شده، شماره ترتیب هر اتصال که در زمان X آغاز می شود، همان X است. فرض کنید که در لحظه $t=30$ یک بسته TPDUs معمولی با شماره ترتیب ۸۰ بر روی اتصال ۵ (که قبلاً ایجاد شده) ارسال گردد. این بسته را X TPDUs بنامید. بلافاصله پس از ارسال X TPDUs، ماشین میزبان از کار افتاده و سریعاً راه اندازی می شود. در لحظه $t=60$ این ماشین اتصالات ۰ تا ۴ را از نو ایجاد می کند. در لحظه $t=70$ ، اتصال ۵ را نیز از نو برقرار می سازد و طبقاً شماره ترتیب بسته ها از ۷۰ شروع می شود. در ۱۵ ثانیه بعدی، این ماشین TPDUs ۷۰ تا ۸۰ را ارسال می نماید. در لحظه $t=85$ یک TPDUs جدید با شماره ۸۰ بر روی اتصال ۵ ارسال و درون زیر شبکه تزریق می شود.^۱ متأسفانه X TPDUs هنوز در زیر شبکه حضور دارد و اگر قبل از TPDUs شماره ۸۰ برسد، پذیرفته شده و بسته جدید (یعنی 80 TPDUs) به عنوان بسته تکراری حذف خواهد شد.

برای پیشگیری از چنین مشکلاتی، باید راهکاری اتخاذ شود تا شماره های ترتیبی که در TPDUs جدید درج می شود تا قبل از انقضای زمان T بهیچوجه مشابه با شماره های قبلی نباشد. در شکل ۶-۱۰-الف شماره های ترتیبی که در هر لحظه از زمان استفاده از آنها مجاز نیست، تحت عنوان «ناحیه ممنوعه» (Forbidden Region) نشان داده شده است.^۲ قبل از ارسال هر گونه TPDUs بر روی یک اتصال، «واحد انتقال» باید مقدار باینری ساعت

۱. فقط شماره آغازین از ساعت سیستم اخذ می شود و برای بسته های بعدی این شماره فقط افزایش می یابد. -م

۲. به خاطر داشته باشید که ملاک انتخاب شماره ترتیب برای بسته های TPDUs در ماشینی که از کار افتاده و سریعاً راه اندازی شده ساعت سیستم است. به همین دلیل نمودار «ناحیه ممنوعه» در دو محور زمان و شماره ترتیب ترسیم شده است. تعبیر ساده تر ناحیه ممنوعه آن است که هر ماشین موظف است برای شماره گذاری بسته ها، از عددی شروع کند که نسبت به مقدار

فعلی را خوانده و بررسی کند که مبادا در ناحیه ممنوعه قرار گرفته باشد.

پروتکل فوق از دو جهت به در دسر می افتد: اگر ماشین میزبان تعداد بسیار زیادی بسته را با سرعت فوق العاده بالا بر روی اتصال ایجاد شده، بفرستد، آنگاه منحنی «شماره ترتیب برحسب زمان» شیب بیشتری نسبت به منحنی «شماره ترتیب اولیه» در شکل ۶-۱۰ پیدا خواهد کرد.^۱ این مسئله بدین معناست که حداکثر نرخ ارسال بسته بر روی یک اتصال، نباید از یک TPDU در هر تیک ساعت تجاوز کند. همچنین «واحد انتقال» در ماشینی که از کار افتاده و مجدداً راه اندازی شده بایستی قبل از باز کردن هر اتصال جدید آنقدر صبر کند تا ساعت سیستم، تیک بزند مبادا شماره انتخابی، تکراری شود. هر دوی این موارد فوق، ایجاب می کند که تیکهای ساعت بسیار کوتاه باشد. (مثلاً هر چند میکروثانیه یکبار یا حتی کمتر تیک بزند).

متأسفانه، ورود به ناحیه ممنوعه (در اثر سرعت بسیار زیاد فرستنده) تنها دلیل بروز مشکل نیست. در شکل ۶-۱۰ ب می بینیم که اگر نرخ ارسال کمتر از نرخ تیک زدن ساعت باشد، ورود به ناحیه ممنوعه از سمت چپ اتفاق می افتد.^۲ هر چه شیب منحنی شماره ترتیب برحسب زمان، بیشتر باشد بروز این مشکل به تعویق خواهد افتاد. همانگونه که قبلاً اشاره کردیم، قبل از ارسال هر TPDU، «واحد انتقال» باید بررسی کند که مبادا به ناحیه ممنوعه وارد شود و اگر اینچنین است بایستی ارسال TPDU را به اندازه T ثانیه به تعویق بیندازد یا آن که شماره های ترتیب را از نو سنکرون نماید.

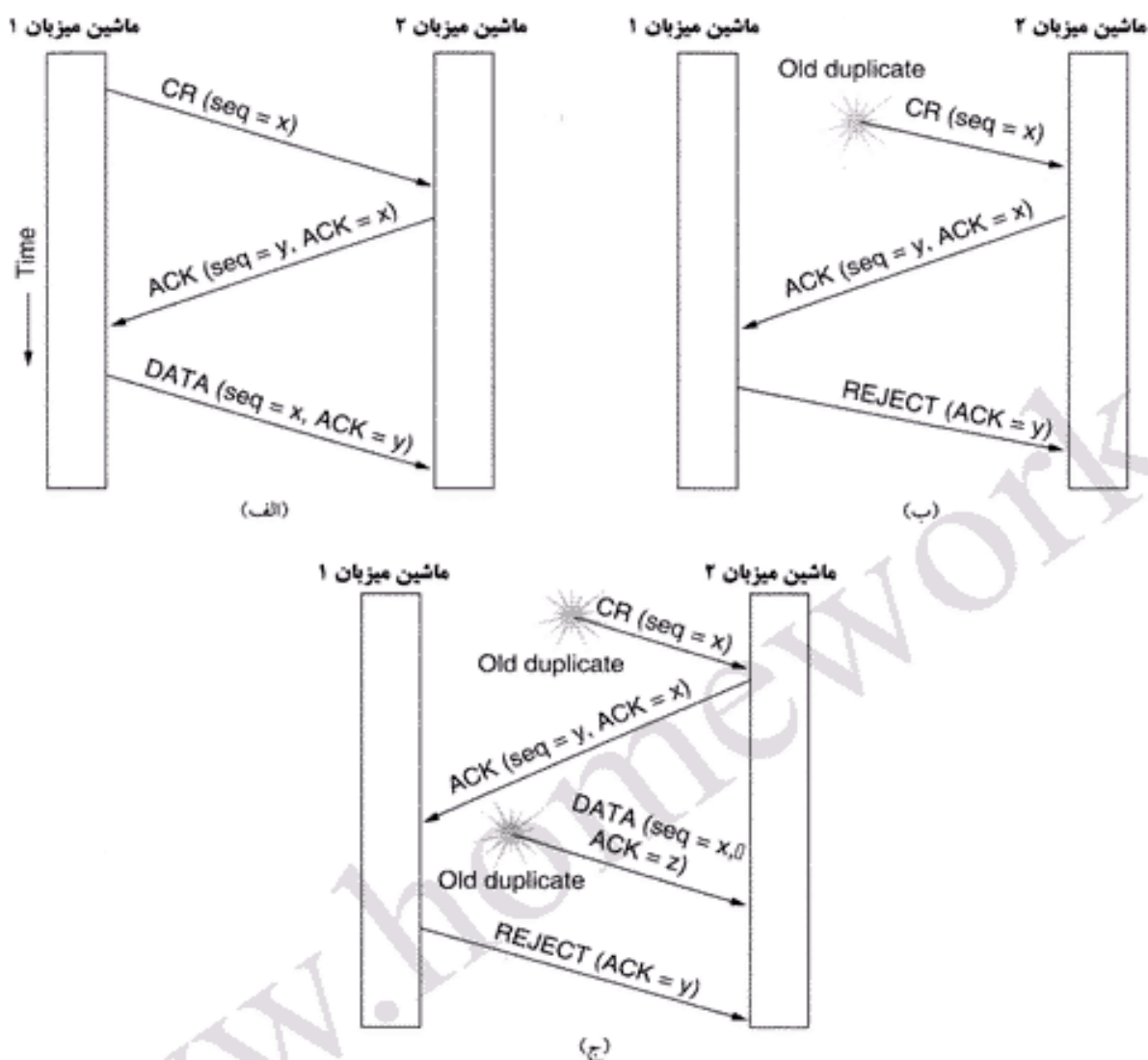
روش مبتنی بر ساعت اگرچه مشکل بسته های تکراری (ناشی از تأخیر) را حل می کند ولیکن برای آن که این راهکار مفید فایده باشد، ابتدا باید «اتصال» مورد نظر ایجاد شود. از آنجایی که بسته های کنترل TPDU نیز ممکن است با تأخیر مواجه شوند، این استعداد وجود دارد که طرفین، در حین توافق بر روی شماره ترتیب با مشکل مواجه شوند. به عنوان مثال فرض کنید که با ارسال یک بسته CONNECTION REQUEST TPDU از طرف ماشین ۱ به ماشین ۲ (به همراه شماره پورت مقصد و شماره ترتیب اولیه) اقدام به برقراری یک اتصال شود. گیرنده یعنی ماشین میزبان ۲، نیز با ارسال بسته کنترل CONNECTION ACCEPTED TPDU، دریافت بسته قبلی را تصدیق (Ack) می نماید. اگر بسته CONNECTION REQUEST TPDU از دست رفته و نسخه تکراری آن با تأخیر به ماشین ۲ برسد این اتصال به نحو نادرستی برقرار خواهد شد.

برای حل این مشکل، تامپنسون (۱۹۷۵) روشی به نام «دست تکانی سه مرحله ای» (Three Way Handshake) پیشنهاد کرد. برای برقراری اتصال، در این پروتکل نیازی نیست که طرفین بر روی شماره ترتیب مشترک و مشابهی توافق نمایند، فلذا در این پروتکل می توان از روشی به غیر از روش مبتنی بر ساعت سراسری سیستم، استفاده کرد. روال معمولی برقراری یک اتصال (که در آن ماشین ۱ شروع کننده آن است)، در شکل ۶-۱۱ الف نشان داده شده است: ماشین ۱، یک شماره ترتیب دلخواه مثل x انتخاب کرده و با ارسال بسته CONNECTION REQUEST TPDU آن را به طرف مقابل اعلام می کند. ماشین ۲ نیز با ارسال بسته ای به نام ACK TPDU، ضمن تصدیق شماره ترتیب x، شماره ترتیب بسته های خودش (یعنی y) را به همتای خود اعلام می دارد. نهایتاً ماشین ۱، شماره ترتیب انتخاب شده توسط ماشین ۲ را (در اولین بسته داده ای که ارسال می کند)، تصدیق خواهد نمود.

عددی ساعت فعلی، T واحد جلوتر باشد. - م

۱. به عبارت بهتر به دلیل آن که شماره های ترتیب در هر تیک ساعت یکبار افزایش می یابند لذا اگر بسته های TPDU با سرعت بیشتری نسبت به تیکهای ساعت تولید شوند اجباراً برخی از شماره های ترتیب تکراری می شوند. - م

۲. به عبارت بهتر اگر نرخ ارسال کمتر از نرخ تیک زدن باشد، چون شماره های ترتیب بایتری و با طول محدود است، به دلیل بالا بودن نرخ تیک زدن، این شماره به صفر برگشته و از نو وارد ناحیه ممنوعه می شود. - م



شکل ۶-۱۱. سه سناریوی مختلف برای برقراری اتصال طبق روش «دست تکانی سه مرحله‌ای». CR مخفف CONNECTION REQUEST است. (الف) عملکرد طبیعی. (ب) نسخه تکراری و قدیمی بسته CONNECTION REQUEST به ناگاه ظاهر می‌شود. (ج) نسخه تکراری بسته‌های CONNECTION REQUEST و همچنین ACK دریافت می‌شوند.

حال اجازه بدهید بررسی کنیم که این پروتکل در برخورد با نسخه‌های تکراری بسته‌های کنترلی TPDU (که در اثر تأخیر زیر شبکه تولید می‌شوند) چگونه کار می‌کند. در شکل ۶-۱۱ ب، فرض شده که بسته CONNECTION REQUEST که نسخه‌ای تکراری و بجا مانده از یک اتصال قدیمی است، با تأخیر دریافت می‌شود. این TPDU (بدون آن که ماشین ۱ از آن اطلاع داشته باشد) به ماشین ۲ می‌رسد و طبقاً ماشین ۲ با ارسال بسته کنترلی ACK TPDU به آن پاسخ می‌دهد. در حقیقت ماشین ۲ با ارسال این بسته خواسته که بررسی کند آیا ماشین ۱ واقعاً تقاضای برقراری اتصالی جدید داده است. وقتی ماشین ۱، تلاش ماشین ۲ برای برقراری این اتصال را رد می‌کند، ماشین ۲ می‌فهمد که با دریافت یک نسخه تکراری از بسته تقاضا، فریب خورده و از پذیرش اتصال امتناع می‌کند. بدین ترتیب، بسته‌های تکراری (ناشی از تأخیر) خطری ندارند.

بدترین حالت زمانی است که هر دو نسخه تأخیر یافته و تکراری CONNECTION REQUEST و ACK

در زیر شبکه شناور باشند. این حالت در شکل ۶-۱۱-ج دیده می شود. همانند مثال قبلی، ماشین ۲ یک نسخه تکراری از بسته کنترل CONNECTION REQUEST دریافت کرده و بدان پاسخ می دهد. در اینجا درک این موضوع اهمیت حیاتی دارد که ماشین ۲ با ارسال y پیشنهاد کرده شماره ترتیب ترافیک خودش (ماشین ۲) به ماشین ۱ از شماره y شروع شود و y را به گونه ای انتخاب کرده که هیچ بسته TPDU یا بسته Ack با شماره y در شبکه سرگردان نمانده باشد.^۱ وقتی بسته تأخیر یافته دوم (یعنی بسته Ack تکراری و باقیمانده از قبل) دریافت می شود، با توجه بدان که به جای y ، شماره ترتیب z مورد تأیید قرار گرفته، ماشین ۲ می فهمد که این بسته قدیمی است. نکته مهم در اینجا است که هیچ ترکیبی از بسته های قدیمی نمی تواند باعث شکست پروتکل و ایجاد اتصال ناخواسته گردد.^۲

۳-۲-۶ خاتمه اتصال

خاتمه دادن به یک اتصال ساده تر از ایجاد آن است ولیکن کار به آن سادگی هم که انتظار می رود، نیست. قبلاً اشاره کردیم که برای خاتمه دادن به یک اتصال دو سبک وجود دارد: متقارن (Symmetric) و نامتقارن (Asymmetric). روش نامتقارن همان روشی است که در سیستم تلفن هم وجود دارد: وقتی یکی از طرفین گوشی را قطع می کند، تماس (اتصال) قطع می شود. در خاتمه متقارن، با هر اتصال به صورت دو «ارتباط یکطرفه» رفتار می شود که هر یک از آنها به صورت مستقل و مجزا خاتمه می یابند.

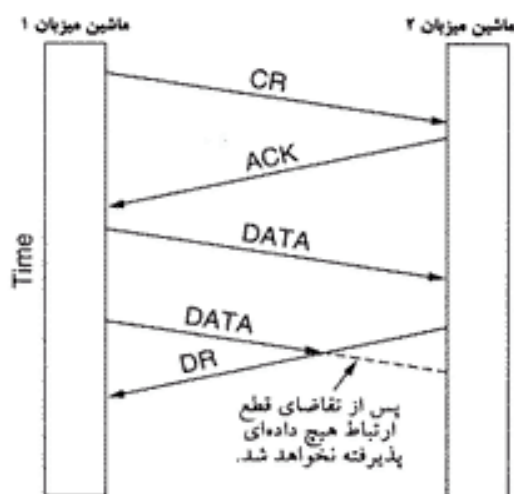
روش خاتمه نامتقارن، فرآیندی ناگهانی است و می تواند منجر به از دست رفتن بخشی از داده ها شود. به سناریوی شکل ۶-۱۲ دقت کنید: پس از برقراری اتصال، ماشین ۱ یک بسته TPDU برای ماشین ۲ می فرستد و به درستی تحویل می شود. سپس ماشین ۱ بسته ای دیگر ارسال می کند. متأسفانه قبل از تحویل بسته دوم، ماشین ۲ تقاضای قطع ارتباط (DISCONNECT) را صادر می نماید. این کار موجب می شود که اتصال قطع شود و بسته دوم از دست برود.

بدیهی است برای آنکه هیچ داده ای از دست نرود به پروتکل پیچیده تری برای خاتمه دادن به اتصال نیاز است. یک راهکار آن است که از روش متقارن برای قطع اتصال استفاده شود تا اتصال در هر یک از دو طرف به صورت مستقل از دیگری خاتمه یابد. در این روش ماشینی که با ارسال بسته کنترل DISCONNECT TPDU تقاضای ختم ارتباط می کند کماکان به دریافت داده ادامه خواهد داد.

روش خاتمه متقارن (Symmetric Release) زمانی خوب کار می کند که هر پروسه حجم ثابت و مشخصی داده برای ارسال داشته باشد و دقیقاً بداند چه زمانی آنها را ارسال کرده است. در شرایطی بغیر از این، تعیین آن که آیا کار به اتمام رسیده و اتصال بایستی قطع شود، ساده نخواهد بود. شاید یک پروتکل پیشنهادی آن باشد که ماشین ۱ بسادگی بگوید: «کار من تمام است. کار شما چطور؟» هر گاه ماشین ۲ پاسخ بدهد که «کار من نیز تمام شد؛ خداحافظ!»، اتصال بین آنها بطور مطمئن خاتمه یابد.

۱. یعنی شماره ترتیب بسته های خودش را با اطمینان از عدم وجود بسته هایی تکراری و سرگردان با همین شماره ها، انتخاب می کند. -م

۲. فرآیند فوق را می توان در یک عبارت ساده تر، بدینگونه خلاصه نمود: ماشین ۱ با ارسال بسته REQUEST اعلام می دارد که تمایل به برقراری یک اتصال دارد و در ضمن مایل است بسته های خود را از شماره x شروع کند. x شماره ای تصادفی است و از تکراری نبودن آن اطمینان دارد چرا که مثلاً در دو دقیقه قبل تاکنون از آن استفاده نکرده است. ماشین ۲ در پاسخ، ضمن تأیید شماره x اعلام می دارد که با برقراری اتصال موافق است و او نیز بسته های خود را از شماره y آغاز می کند. لایه شماره ای تصادفی و غیر تکراری انتخاب می شود. در مرحله سوم x و y به تأیید نهایی می رسد. بدین ترتیب، از آنجایی که بسته های تکراری، شماره های x یا y آنها فرق می کند - فارغ از آن که از نوع REQUEST باشند یا Ack - پذیرفته نخواهند شد. -م

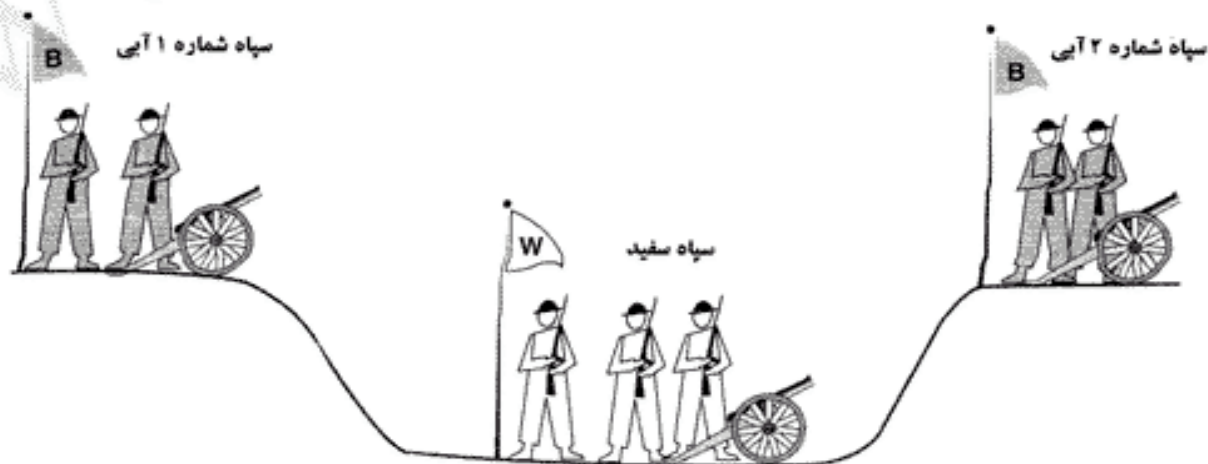


شکل ۶-۱۲. قطع ناگهانی اتصال و از دست رفتن بخشی از داده ها.

متأسفانه این پروتکل همیشه کار نخواهد کرد؛ در این خصوص مسئله مشهوری وجود دارد که مشکل پروتکل را مشخص می نماید و به نام «مسئله دو سپاه» (Two Army Problem) معروف است. تجسم کنید که سپاه سفید در پایین یک دره اردو زده است. به گونه ای که در شکل ۶-۱۳ نشان داده شده، دو سپاه آبی، دره را از دو طرف محاصره کرده اند. نفرات سپاه سفید از یک سپاه آبی بیشتر است ولی مجموع نفرات دو سپاه آبی از سپاه سفید افزونتر می شود. اگر هر یک از سپاهیان آبی به تنهایی حمله را آغاز کنند شکست خواهند خورد مگر آن که هر دو سپاه آبی در یک زمان یورش ببرند.

سپاهیان آبی می خواهند که حمله خود را با یکدیگر هماهنگ نمایند ولیکن تنها راه ارتباط آنها، فرستادن یک پیک به پایین و عبور از دره است. کاری که ممکن است منجر به دستگیری پیک و از دست رفتن پیغام شود. (به عبارتی آنها با یک کانال ارتباطی نامطمئن مواجه هستند.) سؤال این است که آیا پروتکلی وجود دارد که براساس آن سپاهیان آبی پیروز شوند؟

فرض کنید که فرمانده سپاه یکم آبی پیامی بدین مضمون ارسال می کند: «من پیشنهاد می کنم که حمله را در سپیده دم روز ۲۹ مارچ شروع کنیم. نظر شما چیست؟» فرضاً این پیام به سلامت می رسد و فرمانده سپاه دوم آبی



شکل ۶-۱۳. مسئله دو سپاه.

موافقت کرده و پاسخ او نیز به سپاه یکم آبی بر می‌گردد. آیا حمله در موعد مقرر انجام می‌شود؟ شاید نه! چرا که فرمانده سپاه دوم نمی‌داند که آیا پاسخ او به سلامت رسیده است یا خیر! به گمان او اگر در راه بازگشت بلایی بر سر پیک آمده باشد، سپاه یکم آبی حمله نمی‌کند، بنابراین شروع چنین نبردی احمقانه خواهد بود!!

حال بیایید پروتکل فوق را از دو مرحله به سه مرحله دست تکانی (Three Way Handshake) بهبود ببخشیم. فرماندهی که پیشنهاد اول را می‌دهد، باید در نهایت پاسخ طرف مقابل خود را تصدیق کند. حال فرض کنید که هیچ پیامی از دست نرود و سپاه دوم آبی، پیام تصدیق را دریافت نماید ولیکن در اینجا فرمانده سپاه یکم آبی به تردید می‌افتد! چرا که مطمئن نیست که آیا پیام تصدیق او به سپاه دوم رسیده است یا خیر. اگر نرسیده باشد سپاه دوم آبی حمله نخواهد کرد! شاید بتوان پروتکل را چهار مرحله‌ای کرد ولیکن باز هم کمک چندانی نمی‌کند.^۱ در حقیقت می‌توان ثابت کرد که هیچ پروتکلی که بتواند بدون تردید عمل کند وجود ندارد. فرض کنید که چنین پروتکلی وجود داشته باشد. در اینجا یا آخرین پیام پروتکل حیاتی هست یا نیست؛ اگر حیاتی نیست باید آن را حذف کرد. در حقیقت تمام پیامهای غیرضروری باید حذف شوند تا در پروتکل فقط پیامهای ضروری باقی بمانند. حالا چه اتفاقی می‌افتد اگر آخرین پیام، به مقصد نرسد؟ باید گفت که این پیام ضروری بوده و اگر از دست برود حمله شروع نخواهد شد. از آنجایی که فرستنده آخرین پیام از سرنوشت پیامش مطمئن نیست، خود را با شروع حمله به خطر نخواهد انداخت. بدتر آن که، سپاه طرف مقابل نیز در همین گمان که ممکن است طرف مقابل او حمله نکند، عملیات را آغاز نخواهد کرد!!

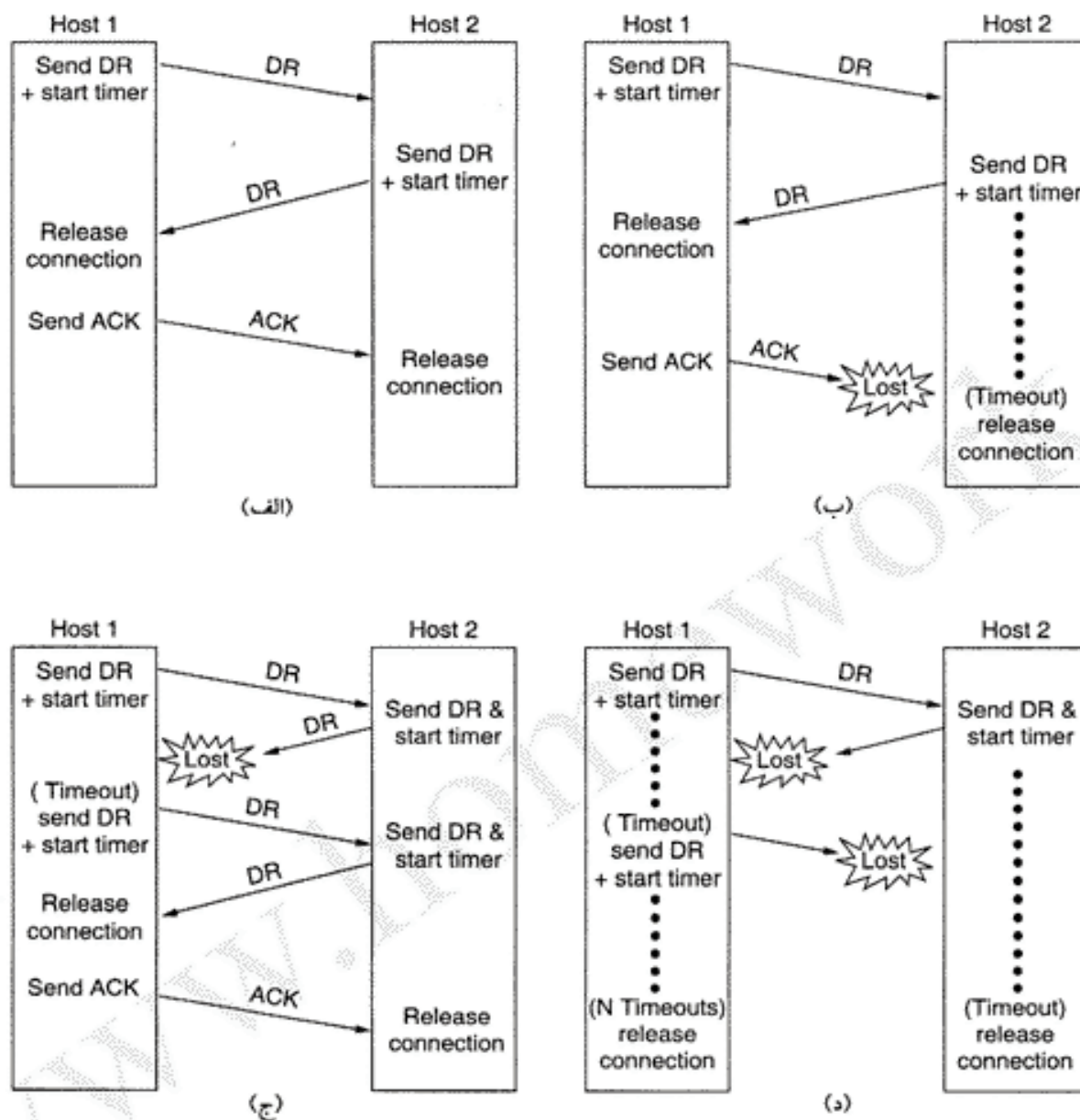
برای آن که بتوانید مسئله دو سپاه را به مسئله خاتمه اتصال ربط بدهید و با هم قیاس کنید به جای واژه «حمله»، واژه «قطع اتصال» بگذارید. اگر هیچیک از طرفین متقاعد نشوند که طرف مقابل واقعاً آماده قطع اتصال است، اتصال هیچگاه خاتمه نمی‌یابد!

در عمل، خطر خاتمه یک اتصال از خطر حمله به سپاه سفید کمتر است!! و در ضمن یکی از طرفین آمادگی بیشتری جهت پذیرش چنین خطری را دارد، لذا وضعیت فوق چندان هم ناامیدکننده نیست! در شکل ۶-۱۴ چهار سناریوی مختلفی که ممکن است در پروتکل قطع سه مرحله‌ای اتصال، اتفاق بیفتد به تصویر کشیده شده است. اگرچه این پروتکل چندان مصون از خطا نیست ولیکن معمولاً کفایت می‌کند.

در شکل ۶-۱۴ الف، حالت طبیعی قطع اتصال را مشاهده می‌کنیم که در آن ماشین ۱ با ارسال بسته کنترلی DR TPDU (DISCONNECT REQUEST) تقاضای خاتمه اتصال را به طرف مقابل خود ارسال کرده است. وقتی در طرف مقابل این بسته دریافت شود، گیرنده آن با ارسال متقابل بسته DR TPDU، پاسخ داده و یک تایمر را فعال می‌کند تا در صورت از بین رفتن این بسته از تایمر کمک بگیرد. وقتی این بسته DR دریافت شود، فرستنده اصلی با بازگرداندن بسته ACK TPDU، به این فرآیند پایان داده و اتصال از طرف او قطع می‌شود. در نهایت وقتی بسته ACK TPDU به طرف مقابل می‌رسد او نیز به این اتصال خاتمه می‌دهد. «خاتمه یک اتصال» بدین معناست که «واحد انتقال» (Transport Entity) تمام اطلاعاتی را که در خصوص آن اتصال در «جدول اتصالات باز» ذخیره نموده پاک کند و به نحوی به صاحب آن اتصال (کاربر) اطلاع بدهد. این عمل با صدور تابع DISCONNECT که توسط کاربر لایه انتقال انجام می‌گیرد متفاوت است.

شکل ۶-۱۴ ب بیانگر حالتی است که بسته ACK TPDU از دست رفته است (این موضوع توسط تایمر آشکار می‌شود). وقتی مهلت تایمر مربوطه منقضی گردد، خواه ناخواه اتصال مربوطه قطع می‌شود. اکنون حالتی را در نظر بگیرید که در آن دومین بسته کنترلی DR از بین می‌رود. کاربری که در ابتدا اقدام به تقاضای قطع اتصال کرده پاسخ مورد نظر خود را دریافت نخواهد کرد و این کار را از نو تکرار می‌نماید. شکل

۱. زیرا همیشه در آخرین پیام شک و تردید وجود دارد و یک دور باطل ایجاد می‌شود. م.



شکل ۶-۱۴. چهار سناریوی مختلف خاتمه دادن به اتصال. (الف) حالت طبیعی دست نکانی

سه مرحله ای. (ب) آخرین ACK از دست رفته است. (ج) پاسخ از دست رفته است. (د) اولین پاسخ و تمام تقاضاهای بعدی قطع ارتباط (یعنی تمام DRها) از دست رفته اند.

۶-۱۴-ج چنین حالتی را به تصویر کشیده است (با این فرض که بعد از بسته DR دوم، هیچ بسته دیگری از دست نرود و تمام بسته های TPDU به سلامت و سر وقت تحویل شود).

در آخرین سناریو که در شکل ۶-۱۴-د نشان داده شده، همه شرایط مثل شکل ۶-۱۴-ج است با این تفاوت که فرض کرده ایم به غیر از بسته کنترلی DR اول، تمام بسته های بعدی نیز از بین رفته اند و هر گونه تلاش جهت ارسال مجدد با شکست مواجه شده است. پس از N بار تلاش پیاپی، فرستنده ناامید شده و به ناچار اتصال را قطع می نماید. عاقبت، تایمر گیرنده طرف مقابل نیز منقضی شده و او نیز اتصال را خاتمه می دهد. اگرچه برای دنیای عمل، این پروتکل کفایت می کند ولی از دیدگاه تنوری، اگر بسته DR اول و تمام N تکرار

بعدی آن از بین بروند، پروتکل با شکست مواجه می‌شود، زیرا اگرچه فرستنده اولین بسته ناامید شده و پس از انقضای مهلت مقرر، به اتصال خاتمه می‌دهد ولیکن طرف مقابل او از تمام این تلاشها و ارسال پیاپی بسته‌ها بی‌خبر است و طبقاً فعال باقی می‌ماند. در چنین شرایطی، اتصال به صورت «نیمه باز» (Half-Open) باقی می‌ماند.

برای اجتناب از چنین وضعیتی می‌توان فرستنده را وادار کرد که حتی پس از N تلاش مجدد باز هم ناامید نشود و آنقدر کار را ادامه بدهد تا بالاخره پاسخی دریافت نماید. ولیکن اگر مهلت طرف مقابل منقضی شده و به اتصال خاتمه بدهد، فرستنده هرگز پاسخی دریافت نخواهد کرد و طبقاً در حلقه بی‌نهایت می‌افتد. اگر هم به طرف مقابل اجازه ندهیم که در اثر انقضای مهلت اتصال را قطع کند آنگاه در شرایطی که در شکل ۶-۱۴ نشان داده شده، پروتکل قفل خواهد شد.

یکی از روشهای حذف اتصالات نیمه باز وضع این قانون است که اگر پس از گذشت زمان معینی (برحسب ثانیه) هیچ بسته TPDU دریافت نشد، اتصال باید به صورت خودکار قطع شود. بدین ترتیب اگر یکی از طرفین به صورت یکجانبه اتصال را قطع نماید، طرف مقابل متوجه عدم فعالیت او شده و او نیز به اتصال خاتمه می‌دهد. البته اگر بخواهیم چنین قانونی اعمال شود لازم است که «واحد انتقال» دارای تایمر خاصی باشد که با ارسال یک TPDU شروع به اندازه‌گیری زمان می‌نماید. اگر پس از ارسال آن TPDU، مهلت تایمر منقضی شد و بسته دیگری جهت ارسال وجود نداشت فرستنده موظف به ارسال یک بسته پوچ (بدون داده) برای طرف مقابل است تا از قطع شدن اتصال (در اثر انقضای مهلت مقرر) جلوگیری نماید. در سمت مقابل نیز اگر قانون قطع خودکار اتصال اعمال شده باشد و برای مدت مشخصی هیچ بسته TPDU دریافت نشود (یا به هر دلیلی بسته‌های پوچ در میانه راه از بین رفته باشند) اتصال به صورت خودکار خاتمه می‌یابد.^۱

پیش از این به جزئیات نخواهیم پرداخت ولیکن تا اینجا باید مشخص شده باشد که خاتمه دادن به یک اتصال بدون از دست دادن داده، به آن سادگی هم که به نظر می‌رسد نیست!

۴-۲-۶ کنترل جریان و بافرسازی (Flow Control and Buffering)

پس از بررسی جزئیات روشهای برقراری و خاتمه یک اتصال، اجازه بدهید چگونگی مدیریت یک اتصال را در حین کار، مطالعه نماییم. یکی از موارد بسیار مهم «کنترل جریان» (Flow Control) است. اگرچه از بسیاری جهات، مسائل مربوط به کنترل جریان در لایه انتقال با مکانیزمهای کنترل جریان در لایه پیوند داده مشابه هستند ولیکن از جهات دیگر دارای تفاوت هستند. تشابه عمده در هر دو لایه آن است که برای جلوگیری از پیشی گرفتن فرستنده سریع از گیرنده کند و از دست رفتن داده‌ها در اثر عدم هماهنگی سرعت ارسال و دریافت، باید از پروتکل «پنجره لغزان» (Sliding Window) یا روشی مشابه استفاده شود. تفاوت اساسی این دو لایه آن است که یک مسیر یاب معمولاً دارای تعداد کمی خط ارتباطی است در حالی که یک ماشین میزبان (Host) می‌تواند بطور همزمان تعداد بی‌شماری اتصال مختلف برقرار کند. این تفاوت موجب می‌شود که نتوان استراتژیهای بافرسازی پیاده شده در لایه پیوند داده را به لایه انتقال نیز تعمیم داده و اعمال نمود.

در پروتکل‌های پیوند داده که در فصل سوم بررسی شدند، فریم‌ها هم در مسیر یاب فرستنده و هم در مسیر یاب گیرنده، بافر می‌شدند. به عنوان مثال در پروتکل ششم، به ازای هر خط ارتباطی هم فرستنده و هم گیرنده به تعداد

۱. به عبارت ساده‌تر اگر هر یک از طرفین ارتباط برای مدت معینی از طرف مقابل خود بسته‌ای دریافت نکند، اتصال را قطع خواهد کرد لذا طرفین موظفند حتی اگر برای لحظاتی داده برای ارسال نداشته باشند، از خود علائم حیاتی نشان داده و بسته‌های پوچ و بدون داده (Dummy) ارسال نمایند تا قانون قطع خودکار اتصالات نیمه باز به درستی کار کند. -م

MAX_SEQ+1 بافر اختصاصی نیاز داشتند که از این تعداد نصفی برای فریمهای ورودی و نصف دیگر برای فریمهای خروجی در نظر گرفته می شد. برای ماشینی که حداکثر تعداد اتصالات آن مثلاً ۶۴ تاست و شماره های ترتیب بسته ها ۴ بیتی هستند، این پروتکل به ۱۰۲۴ بافر نیازمند است.

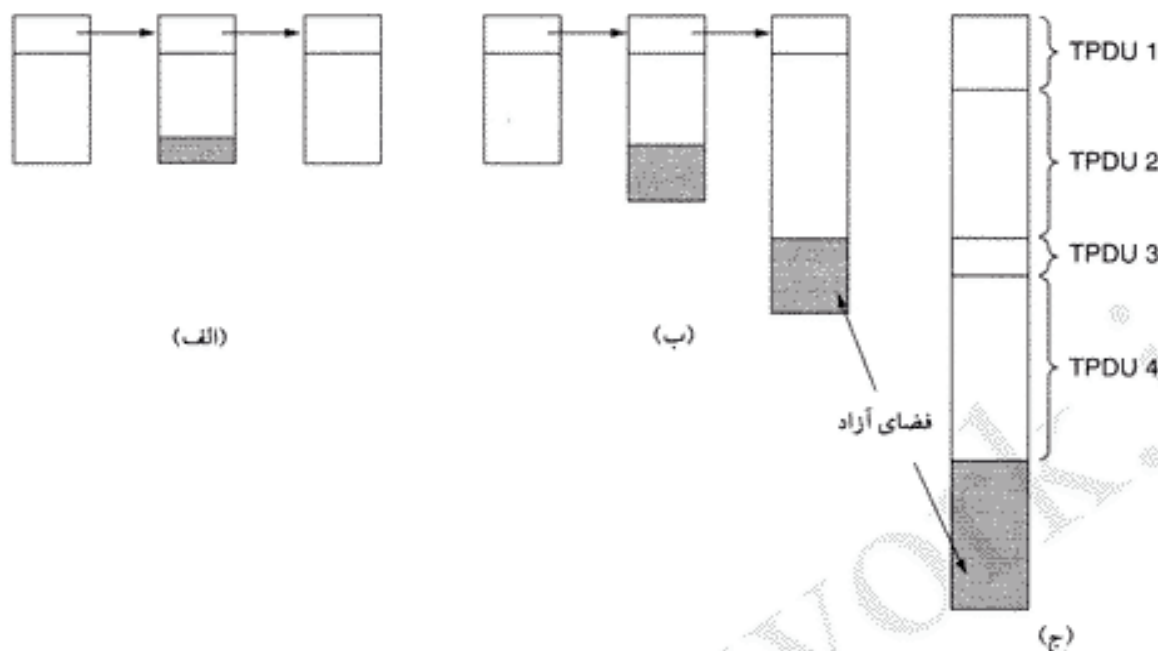
در لایه پیوند داده، فرستنده فریمهای ارسالی خود را بافر می کند تا در صورت نیاز به ارسال مجدد، آنها را در اختیار داشته باشد. اگر زیر شبکه فقط سرویس دیتاگرام عرضه کرده باشد، «واحد انتقال» (Transport Entity) نیز به دلیل مشابه باید بسته های ارسالی خود را در بافر نگهداری کند. در لایه انتقال اگر گیرنده بداند که فرستنده تمام بسته های TPDUs ارسالی خود را تا زمانی که دریافت آنها تصدیق (Ack) نشود در بافر نگهداری می کند، می تواند بر حسب شرایط برای اتصالات خاصی بافر ورودی اختصاصی در نظر بگیرد. (یا برعکس به هر اتصال، بافر ورودی و خروجی مجزا اختصاص بدهد.) به عنوان مثال در لایه انتقال، گیرنده می تواند از یک فضای بزرگ و مرکزی به عنوان بافر ورودی برای تمام اتصالات استفاده کند. بدین نحو وقتی یک TPDUs وارد می شود، گیرنده سعی می کند به صورت پویا از این بافر فضایی را به آن اختصاص بدهد. اگر چنین فضایی موجود بود بسته TPDUs را می پذیرد و در غیر این صورت آن را حذف می نماید. از آنجایی که فرستنده بسته، برای ارسال مجدد بسته هایی که در زیر شبکه از دست می روند، آمادگی دارد فلذا حذف TPDUs توسط گیرنده مشکلی ایجاد نخواهد کرد، اگرچه این کار منجر به از دست رفتن منابع [مثل پهنای باند] شبکه خواهد شد. فرستنده، ارسال مجدد بسته ها را آنقدر تکرار می کند تا بالاخره دریافت آن تصدیق شود.

کوتاه سخن آن که، اگر سرویس شبکه غیر قابل اعتماد باشد، فرستنده باید تمام بسته های TPDUs ارسالی خود را (همانند لایه پیوند داده) بافر کند، ولیکن اگر سرویس شبکه قابل اعتماد باشد می توان از راهکارهای بینابینی بهره گرفت. بالاخص اگر گیرنده اطمینان داشته باشد که گیرنده همیشه فضای بافر کافی در اختیار دارد مجبور نیست که نسخه ای از TPDUs ارسالی خود را نگه دارد ولیکن اگر گیرنده نتواند تضمین کند که بسته های TPDUs ورودی را قطعاً خواهد پذیرفت، فرستنده در هر حال مجبور به بافر کردن آنهاست. در حالت دوم، فرستنده نمی تواند به پیغامهای اعلام وصول بسته ها (Ack) اعتماد کند چرا که پیغامهای اعلام وصول فقط بدین معناست که بسته ها رسیده اند نه آن که پذیرفته شده اند.^۱ در ادامه باز هم به این نکته مهم باز خواهیم گشت.

حتی اگر گیرنده موافق بافر کردن بسته ها باشد باز هم سؤالی باقی می ماند و آن هم حجم بافر مورد نیاز آن است. اگر اغلب TPDUs ها اندازه ای تقریباً مساوی داشته باشند، طبیعی آنست که یک فضای حافظه بزرگ با تعدادی بافر هم اندازه ایجاد نماییم (یک بسته در هر بافر). شکل ۶-۱۵ الف این مفهوم را به تصویر کشیده است. ولیکن اگر اندازه بسته های TPDUs تفاوت فاحش داشته باشند (مثلاً از چند بایت تا یک مگابایت) - مثلاً Telnet - گرفته تا هزاران کاراکتر در حین انتقال فایل)، استفاده از بافرهایی با طول ثابت مشکل ساز خواهد شد: از یک طرف اگر اندازه بافرها را معادل با اندازه بزرگترین بسته TPDUs در نظر بگیریم، برای بسته های کوچک فضای حافظه هدر خواهد رفت. از طرف دیگر، اگر اندازه بافر کمتر از حداکثر طول بسته های TPDUs انتخاب شود برای بسته های بزرگ به چندین بافر نیاز خواهد بود و به پیچیدگی روش می انجامد.

راهکار دیگر برای حل مسئله طول بافر آنست که از بافرهایی با طول متغیر بهره بگیریم. این مفهوم در شکل ۶-۱۵ ب نشان داده شده است. مزیت این روش، استفاده مفید از فضای حافظه است ولیکن به بهای پیچیدگی مکانیزمهای مدیریت بافر تمام می شود. راه حل دیگر آن است که به ازای هر اتصال، یک بافر چرخه ای بزرگ (Circular Buffer) اختصاص بدهیم. (به شکل ۶-۱۵ ج دقت نمایید.) در این سیستم از حافظه، بخوبی استفاده می شود و در اتصالی که حجم مبادله بار سنگین است کارایی خوبی دارد ولی برای اتصالی که حجم مبادله بار کمی

۱. ممکن است بسته ای به سلامت در ماشین گیرنده دریافت شود ولی پروسه ای که صاحب آن است به هر دلیل آن را نپذیرد. -م



شکل ۶-۱۵. (الف) بافرهای زنجیره‌ای با طول ثابت. (ب) بافرهای زنجیره‌ای با طول متغیر. (ج) یک بافر زنجیره‌ای بزرگ به ازای هر اتصال.

دارد ضعیف عمل می‌کند.

حالت پهنه و متعادل در بافرسازی بسته‌ها در سمت گیرنده و فرستنده به نوع ترافیکی بستگی دارد که از طریق هر اتصال [بین دو پروسه] مبادله می‌شود. برای ترافیکهای انفجاری (Bursty) که به پهنای باند کمی احتیاج دارند (مثل داده‌هایی که از طریق یک ترمینال محاوره‌ای تولید و ارسال می‌شود) بهتر آن است که هیچ بافری از قبل اختصاص داده نشود و در عوض بافر مورد نیاز در هر دو سمت (گیرنده و فرستنده) به صورت پویا و برحسب نیاز تخصیص داده شود. (Dynamic Allocation) از آنجایی که در تخصیص پویا، فرستنده اطمینان ندارد که گیرنده قادر به تخصیص حافظه لازم خواهد بود لذا فرستنده باید نسخه‌ای از هر بسته TPDU ارسالی خود را (مادامی که دریافت آنها تصدیق نشده)، در بافر نگاه دارد. از طرف دیگر برای انتقال فایل یا هر کاربرد دیگری که به پهنای باند بالا نیاز دارد بهتر آن است که گیرنده، یک پنجره کامل از بافرها را از قبل رزرو کند تا داده‌ها بتوانند با حداکثر نرخ ممکن جریان داشته باشند.^۱ بنابراین برای ترافیک انفجاری ولی با پهنای باند کم بهتر است که فرآیند بافرسازی به نحو جدی در سمت فرستنده انجام شود ولی برای ترافیک یکنواخت با پهنای باند بالا بافرسازی در سمت گیرنده مفیدتر خواهد بود.

وقتی اتصالی باز یا بسته می‌شود یا الگوی ترافیک تغییر می‌کند، گیرنده و فرستنده ملزم به تنظیم خودکار و پویای فضای بافرهای خود هستند. در نتیجه پروتکل لایه انتقال باید این امکان را فراهم کند که ماشین فرستنده، فضای بافر مورد نیاز را در ماشین سمت مقابل خود، سفارش داده و رزرو نماید. بافرها را می‌توان برای هر اتصال بطور مجزا اختصاص داد یا آن که بافرها به صورت یکجا و برای کل اتصالاتی که بین دو ماشین برقرار می‌شود، رزرو گردد. همچنین در سمت مقابل، گیرنده‌ای که از وضعیت بافر خود آگاه است (ولی حجم ترافیک عرضه شده توسط ماشین مقابل را نمی‌داند) می‌تواند به فرستنده اعلام کند که مثلاً: «من به تعداد x بافر برای شما کنار گذاشته‌ام».

۱. تا بدلیل مشکلاتی که در حین تخصیص پویای حافظه رخ می‌دهد جریان داده‌ها ناگزیر به قطع موقت نشود. -م

هرگاه تعداد اتصالات باز افزایش یابد ممکن است هر یک از طرفین بدلیل محدودیت حافظه مجبور به کاهش حجم بافر تخصیص یافته شوند، لذا پروتکل لایه انتقال باید از چنین قابلیتی برخوردار باشد.

روش عمومی و عقلانی مدیریت پویای بافرها آن است که مسئله بافرسازی را از مسئله اعلام وصول بسته ها (Acknowledgements) تفکیک نماییم.^۱ (برخلاف پروتکل پنجره لغزان که در فصل سوم تشریح شد.) در نتیجه مدیریت پویای بافرها مستلزم داشتن پنجره ای با طول متغیر است. در ابتدا فرستنده براساس پیش بینی های اولیه میزان بافر مورد نیاز خود را اعلام می کند. گیرنده تا حدی که برایش مقدور است حجم خواسته شده را اختصاص می دهد. هر وقت فرستنده یک TPDU ارسال کرد، حجم آن را از میزان فضای اختصاص داده شده کم می کند و هرگاه میزان این فضا به صفر رسید، ارسال را متوقف می نماید. گیرنده نیز اعلام وصول بسته ها (Ackها) و همچنین فضای بافر موجود خود را در ترافیک برگشتی (Reverse Traffic) اعلام می کند.

شکل ۶-۱۶ مثالی از مدیریت پویای پنجره را در زیر شبکه ای دیتاگرام نشان می دهد که در آن شماره های ترتیب ۴ بیتی هستند. فرض کنید که اطلاعات لازم برای تخصیص بافر در بسته های جداگانه TPDU ارسال گردد و در ترافیک برگشتی، جاسازی (Piggyback) نشود. در ابتدا، A هشت بافر تقاضا می دهد ولی فقط با چهار بافر موافقت می شود. A سه بسته TPDU ارسال می کند که سومین آنها از بین می رود. [به سطرهای ۱ تا ۵ از شکل ۶-۱۶ نگاه کنید.] در بسته TPDU ششم دریافت بسته های ۰ و ۱ تأیید می شود و A اجازه می یابد تا بافرهای متعلق به این بسته ها را آزاد کند [سطر ششم از شکل]. عبارت $\langle \text{ack}=1, \text{buf}=3 \rangle$ بدین معناست که بسته ها تا شماره ۱ به درستی دریافت شده اند و فرستنده باید بسته های از ۲ به بعد را ارسال کند؛ در ضمن $\text{buf}=3$ تعداد بافرها را مشخص می کند و طبعاً فرستنده مجاز به ارسال حداکثر سه بسته TPDU است (یعنی بسته های ۲، ۳، ۴). A می داند که بسته شماره ۲ را قبلاً ارسال کرده، فلذا گمان می کند که باید بسته های ۳ و ۴ را بفرستد و این کار را انجام می دهد. در این نقطه A متوقف می شود و صبر می کند تا فضای بافر طرف مقابل آزاد شده و به او اعلام شود. در سطر نهم از شکل، می بینیم که در اثر انقضای مهلت، بسته دوم (که قبلاً از بین رفته) از نو ارسال شده است. (البته ممکن بود انقضای مهلت تایمر زمانی رخ بدهد که به دلیل عدم وجود فضای بافر، فرستنده متوقف شده باشد.) در سطر دهم از شکل، B دریافت تمام بسته های TPDU تا شماره ۴ را اعلام می کند [یعنی بسته های ۲ و ۳ و ۴ به سلامت رسیده اند] ولیکن کماکان به A اجازه ادامه ارسال را نمی دهد [چون با اعلام $\text{buf}=0$ فضای بافر خود را صفر گزارش کرده است و A اجازه ندارد چیزی بفرستد.] به خاطر داشته باشید که چنین وضعیتی برای «پروتکل پنجره ثابت» (Fixed Window Protocol) هرگز پیش نمی آید [چرا که در آنجا اعلام وصول فریمها به منزله آزاد شدن فضای بافر نیز تلقی می شود.] در سطر یازدهم، یک بسته TPDU از B به A ارسال شده و ضمن اعلام وجود یک بافر خالی، به A اجازه ادامه ارسال می دهد.

روشهای تخصیص بافر همانند روش مثال فوق، مستعد بروز مشکلاتی هستند که در اثر از بین رفتن بسته های کنترلی (Control TPDU) در زیر شبکه های دیتاگرام، رخ می دهند. به سطر شانزدهم از شکل ۶-۱۶ دقت نمایید: B چهار بافر آزاد برای A اختصاص داده و آن را در یک بسته کنترلی به سوی A فرستاده ولیکن این بسته از بین رفته است. از آنجایی که بسته های کنترلی شماره گذاری نشده اند و هیچ تایمری جهت ارسال مجدد ندارند فلذا A

۱. یعنی دریافت Ack فقط بیانگر آنست که داده های ارسالی سلامت رسیده اند ولی نباید بدین معنا تلقی شود که فرستنده حق ارسال بسته های بعدی را دارد چرا که ممکن است بسته ها هنوز تحویل پروسه کاربردی نشده باشد و هیچ بافر خالی دیگر برای دریافت بسته بعدی موجود نباشد. -م

A	پیام	B	توضیح
1 →	< request 8 buffers >	→	A wants 8 buffers
2 ←	<ack = 15, buf = 4>	←	B grants messages 0-3 only
3 →	<seq = 0, data = m0>	→	A has 3 buffers left now
4 →	<seq = 1, data = m1>	→	A has 2 buffers left now
5 →	<seq = 2, data = m2>	...	Message lost but A thinks it has 1 left
6 ←	<ack = 1, buf = 3>	←	B acknowledges 0 and 1, permits 2-4
7 →	<seq = 3, data = m3>	→	A has 1 buffer left
8 →	<seq = 4, data = m4>	→	A has 0 buffers left, and must stop
9 →	<seq = 2, data = m2>	→	A times out and retransmits
10 ←	<ack = 4, buf = 0>	←	Everything acknowledged, but A still blocked
11 ←	<ack = 4, buf = 1>	←	A may now send 5
12 ←	<ack = 4, buf = 2>	←	B found a new buffer somewhere
13 →	<seq = 5, data = m5>	→	A has 1 buffer left
14 →	<seq = 6, data = m6>	→	A is now blocked again
15 ←	<ack = 6, buf = 0>	←	A is still blocked
16 ...	<ack = 6, buf = 4>	←	Potential deadlock

شکل ۶-۱۶. تخصیص بافر بصورت پویا. (فلشها جهت ارسال را مشخص می کنند. علامت ...

نشانگر یک TPDU از دست رفته، می باشد.)

در یک بن بست (Deadlock) قرار می گیرد.^۱ برای پیشگیری از بروز چنین وضعیتی، هر ماشین باید بطور متناوب بسته های کنترلی TPDU (شامل وضعیت بافرها و شماره Ack) را بر روی هر اتصال بیکار ارسال نماید. بدین نحو، دیر یا زود بن بست شکسته خواهد شد.

تا اینجا بطور ضمنی فرض کرده ایم که تنها محدودیتی که بر روی نرخ ارسال داده های فرستنده تحمیل می شود، فضای بافر موجود در گیرنده است.^۲ در حالی که با روند روبه کاهش قیمت حافظه، امکان آن فراهم شده که هر ماشین میزبان به حجم بسیار انبوه حافظه مجهز گردد و بدین ترتیب کمبود حافظه به ندرت رخ داده و مشکل بافرها حل می شود.

وقتی فضای بافر محدودیتی بر روی حداکثر میزان جریان اعمال نکند گلوگاه دیگری بروز می کند: «ظرفیت حمل زیر شبکه».^۳ اگر مسیر یابهای مجاور قادر به مبادله حداکثر x بسته در هر ثانیه باشند و در مجموع k مسیر مستقل و مجزا بین یک زوج ماشین وجود داشته باشد، این دو ماشین به هیچ روشی نمی توانند بیش از kx بسته TPDU (در هر ثانیه) مبادله نمایند و فضای بافر موجود در هر یک از طرفین در این مقدار تأثیری ندارد. اگر فرستنده بار سنگینی را به شبکه تزریق کند (به عبارتی بسته های TPDU را با سرعتی بیش از kx TPDU/sec ارسال کند)، زیر شبکه با ازدحام مواجه می شود، زیرا قادر نیست بسته ها را با همان سرعتی که دریافت می کند، تحویل بدهد و بدین ترتیب بخشی از آنها از بین می روند.

۱. زیرا B با خود می اندیشد که چون به A گفته که بافر خالی دارد احتمالاً A داده ای برای ارسال نداشته و A هم با خود می اندیشد که شاید بافرهای B هنوز خالی نشده است و بدین نحو یک دور باطل ایجاد می شود. -م

۲. ازین به بعد فرض بر آنست که گیرنده به طور متناوب فضای بافر خود را به فرستنده اعلام می کند و فرستنده ملزم به ارسال

داده متناسب با این فضا است. -م

۳. Network's Carrying Capacity

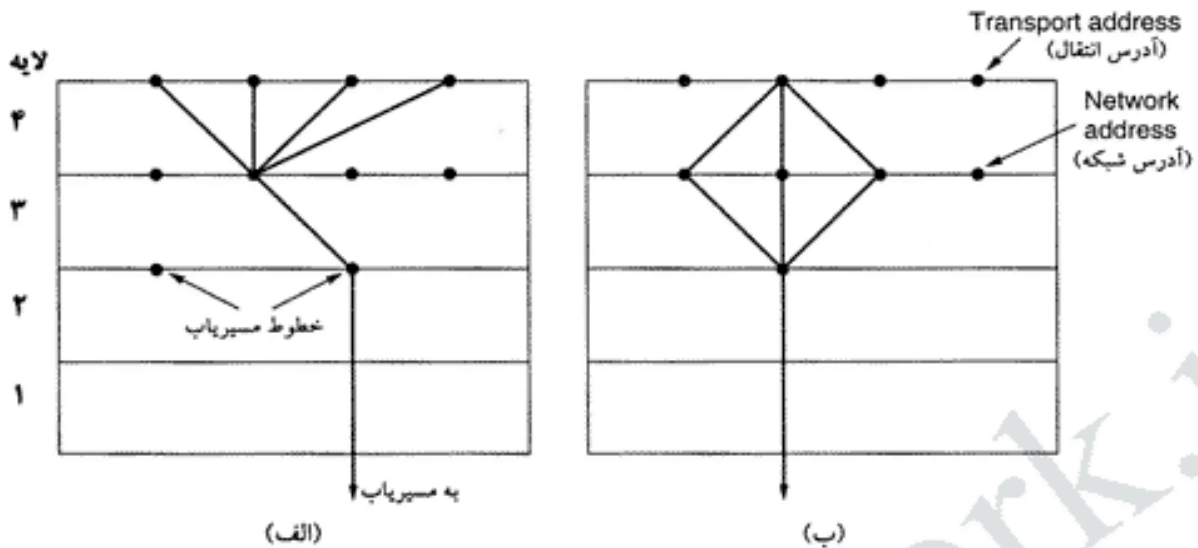
در اینجا به مکانیزمی نیاز داریم که مبتنی بر ظرفیت حمل زیرشبکه، جریان بسته ها را کنترل کند نه براساس ظرفیت بافر در گیرنده. روشن است که مکانیزم کنترل جریان بایستی در سمت فرستنده اعمال گردد تا بسته های بیهوده ای که دریافت آنها تصدیق نشده، متوالیاً ارسال و در زیرشبکه سرگردان نشوند.^۱ «پلسنس» (Belsnes) در سال ۱۹۷۵ با استفاده از «پروتکل پنجره لغزان» (Sliding Window) روشی پیشنهاد کرد که در آن فرستنده برای کنترل جریان، باید اندازه پنجره خود را (به صورت پویا) به نحوی تنظیم کند که با ظرفیت حمل شبکه متناسب باشد. اگر شبکه بتواند حداکثر $c \cdot TPDU/sec$ را حمل نماید و «زمان گردش» (شامل زمان انتقال، تأخیر انتشار، تأخیر انتظار در صف، زمان پردازش در گیرنده و زمان برگشت پیام اعلام وصول Ack) نیز r ثانیه فرض شود، ظرفیت پنجره فرستنده باید معادل $c \cdot r$ باشد. اگر پنجره ای با این اندازه انتخاب شود، فرستنده می تواند در شرایطی کاملاً عادی و به مثابه یک خط لوله (Pipeline) عمل کند. بدون آن که بسته ای در اثر ازدحام از دست برود. [یک کاهش کوچک در کارایی شبکه می تواند موجب توقف او شود.

برای آن که بتوان اندازه پنجره را به صورت متناوب و خودکار تنظیم کرد، فرستنده می تواند بر مقدار این دو پارامتر [یعنی ظرفیت حمل شبکه و زمان گردش] نظارت داشته باشد و اندازه پنجره خود را براساس آنها تعیین نماید. برای محاسبه ظرفیت حمل شبکه می توان به سادگی تعداد بسته هایی را که در دوره زمانی مشخص، ارسال و دریافت آنها تصدیق می شود، شمرد و حاصل را بر طول زمان تقسیم کرد. در خلال زمان محاسبه، فرستنده باید بسته های خود را با حداکثر نرخ ممکن ارسال نماید تا مطمئن شود آنچه که برگشت بسته های اعلام وصول (Ack) را محدود کرده ظرفیت حمل شبکه است نه سرعت پایین خودش. زمان ارسال یک بسته TPDU و برگشت پیام وصول آن (Ack) را می توان به دقت اندازه گیری کرد. از آنجایی که ظرفیت فعلی شبکه متغیر با زمان است، اندازه پنجره نیز باید به دفعات تنظیم شود تا هر گونه تغییر در ظرفیت شبکه را دنبال کرده و اندازه پنجره را با آن تطبیق بدهد. بعداً خواهیم دید که اینترنت از روشی شبیه به همین الگو استفاده می کند.

۵-۲-۶ مالتی پلکسینگ (تسهیم)

مالتی پلکس کردن چندین محاوره همزمان (Conversation) بر روی اتصالات، مدارات مجازی یا لینکهای فیزیکی، نقش بسیار مهمی در لایه های مختلف معماری یک شبکه ایفاء می کند.^۲ در لایه انتقال از چندین جهت به عمل مالتی پلکس نیاز است. به عنوان مثال اگر یک ماشین میزبان فقط دارای یک آدرس در شبکه باشد^۳، تمام اتصالات برقرار شده در لایه انتقال مجبور به مالتی پلکسینگ هستند. یعنی به راهکاری نیاز است تا هرگاه یک TPDU وارد می شود بتوان پروسه تحویل گیرنده آنرا، مشخص کرد. به این وضعیت «مالتی پلکس روبه بالا» (Upward Multiplexing) گفته می شود و شمائی از آن در شکل ۶-۱۷-الف نشان داده شده است. در این شکل چهار اتصال ایجاد شده در لایه انتقال، همگی از اتصال مشترکی در لایه شبکه (مثلاً آدرس IP مشترک) بهره گرفته اند.^۴

۱. عبارت دیگر اگر ظرفیت حمل شبکه محدود باشد فقط فرستنده می تواند جریان ارسال داده های خود را متناسب با این ظرفیت تنظیم کند؛ بنابراین هر مکانیزمی بدین منظور، فقط بر روی فرستنده قابل اجراست. -م
۲. به عبارت دیگر عمل مالتی پلکس را می توان مکانیزمی جهت استفاده مشترک چندین پروسه از یک لینک واحد، دار مجازی واحد یا آدرس مشترک تعبیر کرد. -م
۳. عموماً اینگونه است و اغلب ماشینهای اینترنت فقط یک آدرس IP دارند. -م
۴. به عبارت دیگر تمام داده های این چهار پروسه وقتی به لایه شبکه می رسند درون بسته هایی قرار می گیرند که آدرس مبدا، همه آنها یکسان است و همگی نهایتاً بر روی یک لینک مشترک ارسال می شوند و باید بنحوی در لایه انتقال از یکدیگر تفکیک و به پروسه های متناظر خود تحویل داده شوند. فرآیند تفکیک بسته ها بین پروسه ها در لایه انتقال، «مالتی پلکس روبه بالا» نام دارد.



شکل ۶-۱۷. (الف) مالتی پلکس روبه بالا. (ب) مالتی پلکس روبه پایین.

مالتی پلکس در لایه انتقال، از جهت دیگری نیز می تواند مفید واقع شود. به عنوان مثال فرض کنید در زیرشبکه ای که از درون مبتنی بر مدار مجازی (Virtual Circuit) است، بر روی حداکثر نرخ ارسال هر مدار مجازی، محدودیت گذاشته شده است. اگر کاربر به پهنای باند بیشتری نسبت به نرخ حداکثر هر مدار مجازی نیاز داشته باشد، یک راهکار آن است که چندین اتصال مدار مجازی همزمان در شبکه ایجاد شود و ترافیک داده های یک پروسه به نوبت و چرخشی (Round Robin) بر روی این مدارات مجازی توزیع شود. این مفهوم که در شکل ۶-۱۷-ب به تصویر کشیده شده است، «مالتی پلکس روبه پایین» نام دارد. با داشتن k اتصال باز در سطح شبکه (بعبارت دیگر k مدار مجازی همزمان)، پهنای باند مؤثر با ضریب k افزایش می یابد. مثالی از مالتی پلکس روبه پایین را می توان کاربران خانگی عنوان کرد که از خط ISDN بهره می گیرند. این خط دو اتصال 64kbps دو طرفه را در اختیار می گذارد. با استفاده همزمان از این دو اتصال برای وصل به یک شرکت ارائه دهنده خدمات اینترنت و توزیع ترافیک بر روی آنها، پهنای باند مؤثر 128kbps حاصل می شود.

۶-۲-۶ جبران از کارافتادگی (Crash Recovery)

اگر ماشینهای میزبان یا مسیربایها مستعد خرابی و از کارافتادگی باشند، موضوع احیاء و برگرداندن آنها به فعالیت طبیعی، مسئله مهمی است. اگر «واحدهای انتقال» (Transport Entities) بطور کلی در درون ماشینهای میزبان مستقر باشند، از کارافتادگی شبکه و مسیربای به سادگی اصلاح و جبران خواهد شد. اگر چنین شبکه ای خدمات دیتاگرام عرضه نماید، «واحدهای انتقال» همیشه آمادگی از بین رفتن بسته های TPDUs را دارند و روش برخورد با چنین مشکلی را می دانند.^۱ اگر لایه شبکه خدمات مدار مجازی عرضه کرده باشد، از دست رفتن یک مدار مجازی بدین نحو جبران می شود که مدار مجازی جدیدی برقرار شده و از واحد انتقال در ماشین راه دور سؤال می گردد که چه TPDUs را دریافت کرده و کدام را دریافت ننموده است؛ آنهایی که دریافت نشده اند از نو ارسال می شوند. مسئله دردسرافرین، آنست که ماشینهای میزبان را چگونه پس از خرابی به فعالیت طبیعی بازگردانیم؟ برای آن که دشوار بودن این عمل را نشان بدهیم فرض کنید که یک ماشین میزبان در حال ارسال یک فایل طولانی برای

۱. بعبارتی از بین رفتن بسته های دیتاگرام امری طبیعی است و نیاز به مکانیزمی اضافی ندارد، لذا خرابی یک کانال یا از کارافتادن موقت مسیربایهای میانی مشکلی برای لایه انتقال ایجاد نمی کند. -م

ماشین میزبان دیگر (مثلاً سرویس دهنده فایل) به روش «توقف و انتظار» (Stop & Wait) است. [یعنی یک قطعه از فایل ارسال شده و منتظر اعلام وصول آن می ماند.] لایه انتقال در ماشین سرویس دهنده، بسته های TPDU را یکی یکی به پروسه کاربردی مربوطه تحویل می دهد. در الثای کار، سرویس دهنده از کار می افتد؛ پس از راه اندازی مجدد، وقتی این ماشین فعالیت طبیعی خود را از سر می گیرد، جداول او مقداردهی اولیه می شوند فلذا او نمی داند که دقیقاً در کجای کار این خرابی اتفاق افتاده و طبعاً همه چیز باید از نو تکرار شود.

برای آن که بتوان وضعیت را به حالت قبلی برگرداند، سرویس دهنده می تواند یک بسته TPDU را به روش پخش فراگیر (Broadcast) برای بقیه ماشینهای میزبان بفرستد و با اعلام آن که از کار افتاده بوده از تمام ماشینهای مشتری (Clients) درخواست کند که وضعیت تمام اتصالات باز خود را [که با او برقرار کرده بوده اند] اعلام نمایند. هر مشتری ممکن است یکی از این دو وضعیت را داشته باشد: منتظر یک TPDU باشد (وضعیت S1) منتظر هیچ TPDU نباشد (وضعیت S0). براساس این اطلاعات وضعیت، مشتری می تواند در خصوص ارسال مجدد TPDU های اخیر تصمیم بگیرد.

در نگاه اول این فرآیند ساده به نظر می رسد: مشتری باید بسته های TPDU اعلام وصول نشده را از نو ارسال نماید (یعنی وضعیت S1). ولیکن بررسی موشکافانه این روش، مشکلات آن را آشکار می کند. به عنوان مثال وضعیتی را در نظر بگیرید که واحد انتقال در ماشین سرویس دهنده، پیام اعلام وصول (Ack) یک بسته TPDU را ارسال کرده و محتوای آن را به برنامه کاربردی تحویل داده است. نوشتن یک TPDU در یک استریم و سپس ارسال پیغام اعلام وصول دو رخداد کاملاً مجزا است و نمی تواند بطور همزمان انجام شود. اگر خرابی ماشین دقیقاً زمانی رخ بدهد که پیغام اعلام وصول بسته، ارسال شده ولی محتوای آن بسته هنوز در استریم متعلق به برنامه کاربردی نوشته نشده، مشتری پیغام اعلام وصول بسته را دریافت می کند و طبعاً در وضعیت S0 قرار می گیرد ولیکن محتوای بسته حقیقتاً به برنامه کاربردی نرسیده و ماشین مشتری آن را از نو ارسال نخواهد کرد چرا که به غلط فکر می کند که این TPDU دریافت شده است. چنین تصمیمی منجر به از دست رفتن یک TPDU خواهد شد.

در اینجا ممکن است با خود بیندیشید که: «این مشکل را براحتی می توان حل کرد. کل کاری که باید انجام شود آنست که واحد انتقال (Transport Entity) بازنویسی و اصلاح شود تا اول بسته را بنویسد و سپس آن را اعلام وصول کند» ولیکن تجسم کنید که ابتدا عمل نوشتن در استریم برنامه کاربردی انجام شود ولی دقیقاً قبل از ارسال پیغام اعلام وصول، ماشین از کار بیفتد. این رخداد منجر به تولید بسته های TPDU تکراری شده و چون تکراری بودن آنها کشف نمی شود در استریم خروجی پروسه کاربردی سرویس دهنده نوشته خواهد شد.

فارغ از آنکه برنامه سرویس دهنده و مشتری چگونه برنامه نویسی شده اند همیشه وضعیتی وجود دارد که پروتکل در حین برگشت به حالت طبیعی (پس از خرابی ماشین) با شکست جدی مواجه می شود. سرویس دهنده را می توان به یکی از دو روش ذیل برنامه نویسی کرد: اول پیغام اعلام وصول بسته را بفرستد یا اول محتوای بسته را در استریم پروسه کاربردی بنویسد. برنامه مشتری را می توان به چهار روش نوشت: (۱) همیشه آخرین TPDU ارسالی خود را مجدداً بفرستد. (۲) هیچگاه آخرین بسته TPDU را نفرستد. (۳) فقط وقتی بسته آخر را مجدداً ارسال کند که در وضعیت S0 باشد. (۴) بسته آخر را فقط وقتی از نو ارسال کند که در وضعیت S1 باشد. ترکیب اینها هشت حالت مختلف را پدید می آورد ولیکن به گونه ای که خواهیم دید برای هر ترکیب، مجموعه ای از رخدادها منجر به شکست پروتکل می شود.

در سرویس دهنده وقوع سه رخداد محتمل است: (۱) ارسال پیام Ack (A) (۲) نوشتن در پروسه خروجی (W) (۳) از کار افتادگی (C). این سه رخداد می تواند به شش ترتیب مختلف اتفاق بیفتد: AWC, AC(W),

$WC(A)$ ، $C(WA)$ ، WAC و $WC(A)$. پراوتزها نمایانگر آن هستند که وقتی سیستم از کار می افتد طبعاً مراحل بعدی که درون پراوتز نشان داده شده اند، نمی تواند ادامه پیدا کند. (یعنی وقتی سیستم خراب شد، کار تمام است و رخدادهای درون پراوتز فرصت وقوع پیدا نمی کنند.) شکل ۶-۱۸ هشت ترکیب مختلف ناشی از عملکرد سرویس دهنده و مشتری و ترکیبات معتبر آنها نشان داده شده اند. دقت کنید که در هر استراتژی، دنباله ای از رخدادهای می تواند منجر به شکست پروتکل شود. به عنوان مثال، اگر ماشین مشتری، همیشه آخرین بسته ارسالی خود را ارسال کند، استراتژی AWC^1 منجر می شود که بسته ای تکراری و غیرقابل تشخیص، دریافت و تحویل پروسه شود، حتی اگر دو رویداد دیگر [یعنی AW] به درستی انجام گردد.

استراتژیهای بکار رفته در ماشین گیرنده

استراتژیهای بکار رفته در ماشین فرستنده	First ACK, then write			First write, then ACK		
	AC(W)	AWC	C(AW)	C(WA)	WAC	WC(A)
Always retransmit	OK	DUP	OK	OK	DUP	DUP
Never retransmit	LOST	OK	LOST	LOST	OK	OK
Retransmit in S0	OK	DUP	LOST	LOST	DUP	OK
Retransmit in S1	LOST	OK	OK	OK	OK	DUP

OK = پروتکل بدستی عمل می کند.
 DUP = پروتکل یک پیام تکراری تولید می کند.
 LOST = پروتکل پیامی را از دست می دهد.

شکل ۶-۱۸. ترکیبات مختلف استراتژیهای اتخاذ شده در سرویس دهنده و مشتری.

پیچیده و هوشمندتر کردن پروتکل کمک چندانی به حل مشکل نمی کند. حتی اگر مشتری و سرویس دهنده، قبل از آنکه سرویس دهنده تلاش کند داده ها را به پروسه کاربردی تحویل بدهد (یا به عبارت دیگر آنها را بنویسد) چندین بسته TPDU مبادله کنند تا مشتری بداند چه اتفاقی در حال وقوع است، باز هم در صورت از کار افتادن سرویس دهنده و برگشت به حالت طبیعی، مشتری نمی تواند بفهمد که وقوع خرابی قبل از نوشتن بسته ها بوده یا بعد از آن. بنابراین ناگزیر به پذیرش این نتیجه هستیم: در شرایطی که رخدادهای بطور همزمان اتفاق نمی افتند^۲ از کار افتادن ماشین میزبان و بازگشت آن به حالت طبیعی را نمی توان بی سر و صدا اصلاح کرد و از دید لایه های بالاتر مخفی نخواهد ماند.^۳

اگر بخواهیم نتیجه گیری فوق را تعمیم بدهیم می توان آن را بدین نحو بیان کرد که «اگر لایه N دچار از کار افتادگی شود، بازبایی و بازگرداندن آن به شرایط طبیعی، تنها در لایه $N+1$ مقدور است و آن هم مشروط به آنکه لایه بالایی اطلاعات کافی از وضعیت فعلی نگه داشته باشد.» در بالا اشاره شد که هر گونه خرابی در لایه شبکه می تواند توسط لایه انتقال اصلاح و جبران شود به شرط آن که هر یک از طرفین یک اتصال، وضعیت فعلی خود و دیگری را نگه داشته باشند.

۱. AWC یعنی ابتدا اعلام وصول بسته، سپس تحویل داده ها به پروسه و سپس از کار افتادن ماشین -م

۲. یعنی اول اعلام وصول و بعد تحویل داده ها به پروسه انجام می شود یا بالعکس ولی نه همزمان -م

۳. بدون تمهیدات برنامه های کاربردی در لایه کاربردی، داده هایی که قبل از خرابی ماشین دریافت شده اند از درجه اعتبار ساقط است. -م

این مسئله ما را متوجه معنای حقیقی موضوعی می کند که اصطلاحاً با عنوان «تصدیق دریافت داده ها به صورت انتباه انتها» (End-to-End Acknowledgement) معرفی می شود. اصولاً پروتکل لایه انتقال «انتباه انتها» است و همانند لایه های زیرین به صورت «زنجیره ای» عمل نمی کند.^۱ حال وضعیتی را مد نظر قرار بدهید که یک کاربر تقاضایی را بر روی یک پایگاه داده راه دور اعمال می کند. فرض کنید واحد انتقال (Transport Entity) به گونه ای برنامه نویسی شده که ابتدا بسته های TPDUs را به لایه بالاتر تحویل بدهد و بعد از آن پیغام تصدیق دریافت (Ack) بفرستد. در چنین حالتی حتی بازگشت پیغام Ack به ماشین کاربر، الزاماً به معنای آن نیست که این درخواست واقعاً بر روی بانک اطلاعاتی اعمال و بهنگام سازی شده است. یک مکانیزم «تصدیق انتباه انتها» که در آن دریافت Ack به معنای انجام صد درصد کار و عدم دریافت آن به معنای انجام نشدن کار باشد، در عمل دست نیافتنی است. این نکته به تفصیل در مرجع (Saltzer et al. 1984) بحث شده است.

۳-۶ یک پروتکل ساده انتقال

برای تبیین ایده هایی که تا اینجا بررسی شدند، در این بخش به مطالعه مبسوط یک مثال عملی از لایه انتقال می پردازیم. خدمات اولیه ارائه شده همان توابع اولیه (Primitives) اتصال گرا هستند که در شکل ۶-۲ نشان داده شده اند. انتخاب این توابع و عملکرد اتصال گرای اولیه، مثال ما را بسیار شبیه به پروتکل TCP (ولی ساده تر از آن) خواهد نمود.

۱-۳-۶ توابع اولیه ارائه خدمات در مثال فوق

اولین مسئله ای که با آن مواجهیم تشریح صحیح و دقیق «عملکردهای اولیه انتقال» است. عمل CONNECT ساده است: ما یک تابع کتابخانه ای به نام connect خواهیم داشت که برای برقراری یک اتصال، می توان آنرا با پارامترهای مناسب فراخوانی کرد. این پارامترها عبارتند از شناسه TSAP مبدا (محلی) و شناسه TSAP مقصد (راه دور). پس از فراخوانی، پروسه صدا زننده متوقف می شود (به عبارت دیگر معلق می گردد) تا واحد انتقال برای برقراری اتصال اقدام کند. اگر برقراری اتصال انجام شود، پروسه صدا زننده از حالت توقف خارج شده و می تواند شروع به ارسال داده بنماید.

هر گاه پروسه ای در سمت سرویس دهنده بخواهد قادر به پذیرش تقاضاهای ارتباط باشد، تابع listen را فراخوانی کرده و TSAP مشخصی را که باید شنود شود، تعیین می کند. پروسه صدا زننده این تابع، مادامی که یک پروسه راه دور تقاضای برقراری اتصال با این TSAP را ندهد در حالت توقف باقی خواهد ماند.

دقت کنید که این مدل کاملاً نامتقارن است: یک طرف «غیرفعال» (Passive) است و با اجرای تابع listen آنقدر منتظر می ماند تا اتفاقی بیفتد [پروسه ای ارتباط برقرار کند]. طرف دیگر فعال (Active) است و در زمان دلخواه شروع به برقراری اتصال می کند. یک سؤال جالب آن است که اگر طرف فعال زودتر شروع کند چه باید کرد. یک استراتژی آن است که اگر هیچ پروسه در حال شنود به TSAP راه دور وجود نداشته نباشد [یعنی سرویس دهنده هنوز اجرا نشده باشد]، تلاش برای برقراری ارتباط ناکام گذاشته شود. استراتژی دیگر آن است که پروسه آغازکننده آنقدر منتظر نگاه داشته شود تا بالاخره پروسه ای شروع به گوش دادن به TSAP مورد نظر کند.

۱. یعنی مثلاً یک بسته در لایه شبکه ممکن است به صورت زنجیره ای دهها بار در مسیر یابهای واقع بر مسیر پردازش شوند ولی یک بسته TPDUs در مبدا تولید و در مقصد پردازش و مصرف می شود لذا تمام رخدادهای اتفاقی در لایه های زیرین در مقصد قابل بررسی و اصلاح است. مثلاً اگر یک مسیر یاب به ناگاه خراب شود و چند بسته TPDUs را با خود از بین ببرد هیچ اتفاقی برای لایه انتقال نمی افتد چرا که فرستنده متوجه عدم وصول آنها شده و پس از انقضای مهلت آنها را از نو ارسال می کند. - م

در مثالمان از یک حالت میانه استفاده کرده ایم: تقاضای برقراری اتصال برای یک مدت زمان مشخص منتظر نگه داشته می شود تا اگر قبل از انقضای مهلت، پروسه ای در سمت مقابل تابع *listen* را فراخوانی کرد، اتصال برقرار شود. در غیر این صورت تقاضای برقراری اتصال رد شده و پروسه صدازنده از حالت توقف خارج و پیغام خطایی به او برگردانده می شود.

برای خاتمه دادن به یک اتصال، از تابع کتابخانه ای *disconnect* بهره گرفته ایم. وقتی هر دو طرف، ارتباط را قطع کردند، اتصال خاتمه می یابد. به عبارت دیگر از مدل «متقارن» برای ختم ارتباط استفاده نموده ایم.

مسئله انتقال داده دقیقاً شبیه به برقراری اتصال است: عمل ارسال ماهیتی «فعال» و فرایند دریافت ماهیتی «غیرفعال» دارد؛ لذا برای انتقال داده از راه حل مشابه با برقراری اتصال استفاده خواهیم کرد: فراخوانی تابع *send* داده ها را فوراً ارسال می کند در حالی که فراخوانی تابع *receive* غیرفعال بوده و تازمانی که یک بسته TPDU دریافت نشود منجر به توقف پروسه می شود.

بدین ترتیب مجموعه سرویسهای ارائه شده ما، شامل پنج عملکرد اولیه (تابع پایه) است: (۱) CONNECT (۲) LISTEN (۳) DISCONNECT (۴) SEND (۵) RECEIVE. به ازای هر یک از این پنج عملکرد اولیه دقیقاً یک تابع کتابخانه ای خاص وجود دارد که سرویس مربوطه را پیاده کرده است. پارامترهای این توابع کتابخانه ای به صورت زیر هستند:

```
connum = LISTEN(local)
connum = CONNECT(local,remote)
status = SEND(connum,buffer,bytes)
status = RECEIVE(connum,buffer,bytes)
status = DISCONNECT(connum)
```

تابع اولیه LISTEN اعلام می کند که پروسه صدازنده آن علاقمند به پذیرش آن دسته از تقاضاهای برقراری اتصال است که با شناسه TSAP مشخص شده، وارد می شوند. پروسه ای که این تابع را فراخوانی کرده، مادامیکه که یک پروسه راه دور سعی در برقراری اتصال نکند، متوقف و منتظر خواهد ماند. در اینجا هیچ مهلتی تعیین نشده است.

تابع اولیه CONNECT دو پارامتر را دریافت می کند: (۱) شناسه TSAP محلی (Local TSAP) (که در تعریف تابع با نام *local* ظاهر شده است). (۲) شناسه TSAP راه دور (با نام *remote*)؛ سپس سعی می کند یک اتصال بین این دو برقرار نماید. اگر این کار با موفقیت انجام شد یک عدد نامنفی در متغیر *connum* برمی گرداند تا برای مشخص کردن هویت اتصال در فراخوانیهای بعدی مورد استفاده قرار بگیرد. اگر ایجاد اتصال با شکست مواجه شود، عددی منفی در *connum* برمی گرداند. در مدل ساده ما، هر TSAP می تواند فقط برای ایجاد یک اتصال مورد استفاده قرار بگیرد، لذا یک دلیل موجه برای شکست در برقراری اتصال آن است که آدرس انتقال (یعنی TSAP مورد نظر) در حال استفاده توسط پروسه دیگر باشد. برخی از دلایل دیگر عبارتند از: خاموش بودن ماشین راه دور، آدرس محلی نامعتبر، آدرس راه دور نامعتبر.

تابع اولیه SEND محتویات بافر را به عنوان یک پیام، بر روی اتصال مشخص شده ارسال می کند؛ البته در صورت نیاز داده ها را در چند قطعه (چند بسته TPDU) می فرستد. خطاهای احتمالی در متغیر *status* برگردانده می شود. برخی از علل خطاهای احتمالی عبارتند از: عدم وجود اتصال، نامعتبر بودن آدرس بافر یا منفی بودن پارامتر تعداد بایتی که باید ارسال شود (یعنی منفی بودن پارامتر *bytes*).

تابع اولیه RECEIVE، مشخص کننده آن است که پروسه صدازنده آن تمایل به دریافت داده دارد. اندازه پیام دریافتی در پارامتر *bytes* قرار داده می شود. اگر پروسه راه دور اتصال را قطع کرده باشد یا آدرس بافر نامعتبر (مثلاً

آدرسی خارج از فضای برنامه کاربر) باشد، یک کد خطا در متغیر status برگردانده می شود تا ماهیت خطا را مشخص نماید.

تابع اولیه DISCONNECT اتصال مورد نظر را قطع می نماید. پارامتر connum، مشخص می کند که کدام اتصال باید بسته شود. علل خطاهای احتمالی عبارتند از: connum شماره اتصالی متعلق به پروسه ای دیگر باشد یا شناسه اتصال نامعتبر باشد. بهر حال در متغیر status، کدی برگردانده می شود که مقدار صفر به معنای موفقیت و مقدار غیر صفر به معنای کد مشخصه خطاست.

۲-۳-۶ واحد انتقال در مثال فوق

قبل از آن که کدهای برنامه «واحد انتقال» (Transport Entity) را بررسی نماییم لطفاً به خاطر بسپارید که این مثال شبیه به مثالهایی که در فصل سوم عرضه شدند جنبه آموزشی دارد و طرحی جدی محسوب نمی شود. برای پرهیز از پیچیدگی، بسیاری از جزئیات فنی (مثل کنترل دقیق و گسترده خطا) که در سیستمهای واقعی مورد نیاز هستند، حذف شده اند.

لایه انتقال برای ارسال یا دریافت هر TPDU از سرویسهایی که لایه شبکه در قالب توابع اولیه در اختیار گذاشته، بهره می گیرد. در این مثال ابتدا بایستی نوع سرویس لایه شبکه و توابع اولیه مورد استفاده مشخص شود. یک گزینه آن است که از سرویس نامطمئن دیتاگرام بهره گرفته شود. برای آن که مثالمان ساده باشد ما این گزینه را انتخاب نمی کنیم. در سرویس دیتاگرام کد برنامه لایه انتقال بسیار طولانی و پیچیده خواهد شد، زیرا بخش بزرگی از این برنامه صرف مدیریت بسته های از بین رفته و مسائل ناشی از تأخیر خواهد شد. مضاف بر این، بسیاری از ایده های مرتبط با این موضوعات قبلاً به تفصیل در فصل سوم بررسی شده اند.

در عوض، ترجیح داده ایم از سرویسهای مطمئن یک شبکه اتصال گرا بهره بگیریم. بدین ترتیب قادر خواهیم بود که بر روی موضوعاتی از لایه انتقال متمرکز شویم که در لایه های زیرین مطرح نمی شوند. این موضوعات عبارتند از: برقراری اتصال، خاتمه دادن به اتصال، مدیریت اعتبار (Credit Management). سرویسهای یک لایه انتقال ساده که بر روی شبکه ATM به کار گرفته می شود، شبیه به مثال ما خواهد بود.

«واحد انتقال» عموماً بخشی از سیستم عامل ماشین میزبان است؛ گاهی هم در قالب یک مجموعه از توابع کتابخانه ای ارائه و در فضای آدرس برنامه کاربر اجرا می شود. برای سادگی، برنامه مثال ما در قالب یک مجموعه از توابع کتابخانه ای عرضه شده است ولی با تغییرات اندک می توان آن را تبدیل به بخشی از سیستم عامل کرد.

به هر حال اشاره به این نکته خالی از لطف نیست که «واحد انتقال» در این مثال یک بخش کاملاً مستقل نیست و بخشی از پروسه کاربردی محسوب می شود. خصوصاً وقتی کاربر یک تابع اولیه نظیر LISTEN (که برنامه را بلوکه می کند) اجرا نماید، کل واحد انتقال نیز متوقف می شود. اینگونه طراحی برای ماشینهایی مناسب است که تک کاربره و تک پروسه ای هستند، در حالی که در ماشینی با چندین کاربر و پروسه، طبعاً نیاز به یک واحد انتقال مستقل داریم تا از پروسه های کاربران مجزا باشد.

تعامل با لایه شبکه از طریق پروسیجرهای to_net و from_net انجام می شود. هر یک از این پروسیجرها، شش پارامتر دارند: اولین پارامتر شناسه اتصال مدار مجازی است. پارامترهای بعدی، بیت های Q و M هستند که هر گاه به ۱ تنظیم شده باشند، به ترتیب مشخص می کنند که (۱) پیام از نوع کنترلی است (۲) بخشی از داده های این پیام در بسته های بعدی می آید. پس از آن نوع بسته مشخص می شود که می تواند یکی از شش نوع معرفی شده در جدول ۶-۱۹ باشد. سپس اشاره گری که آدرس محل شروع داده ها را مشخص می کند و نهایتاً یک عدد صحیح می آید که تعداد بایتهای داده را تعیین می نماید.

وقتی پروسیجر to_net فراخوانی می شود، «واحد انتقال» تمام این پارامترها را پس از مقداردهی در اختیار لایه

توصیف	نام بسته لایه شبکه
به منظور ایجاد اتصال ارسال می شود.	CALL REQUEST
پاسخ بسته CALL REQUEST (بمعنای موافقت با برقراری اتصال).	CALL ACCEPTED
به منظور قطع اتصال ارسال می شود.	CLEAR REQUEST
پاسخ بسته CLEAR REQUEST (بمعنای موافقت با ختم اتصال).	CLEAR CONFIRMATION
برای انتقال داده بکار می رود.	DATA
یک بسته کنترلی برای مدیریت پنجره	CREDIT

شکل ۶-۱۹. بسته هائی از لایه شبکه که در مثالمان از آنها بهره گرفته ایم.

شبکه قرار می دهد؛ برعکس، با فراخوانی `from_net`، لایه شبکه این پارامترها را از بطن بسته های ورودی بیرون کشیده و تحویل واحد انتقال می دهد. وقتی اطلاعات لازم به صورت پارامترهای پروسیجر به لایه شبکه تحویل می شود (به جای آن که این اطلاعات با تحویل بسته های واقعی رد و بدل گردد)، لایه انتقال از درگیری با جزئیات پروتکل لایه شبکه دور نگاه داشته می شود.^۱ هرگاه «پنجره لغزان مدار مجازی»^۲ پر شده باشد و واحد انتقال سعی در ارسال بسته ای کند، اجرای `to_net` آنرا آنقدر معطل نگاه خواهد داشت تا فضای کافی در پنجره آزاد گردد. این مکانیزم با استفاده از دستوراتی شبیه به `enable_transport_layer` یا `disable_transprot_layer` (که در فصل سوم مشابهشان را بررسی کردیم) در لایه شبکه انجام می گیرد و بطور کامل از دید واحد انتقال پنهان خواهد ماند. مدیریت پنجره ها نیز در لایه شبکه انجام می شود.

اضافه بر مکانیزم «تعطیل» (*suspension*) فوق الذکر [که توسط لایه شبکه به لایه انتقال تحمیل می شود و لایه انتقال نقشی در آن ندارد]، در لایه انتقال دو پروسیجر `sleep` و `wakeup` (که در فهرست نیامده اند) تعریف شده که به منظور ایجاد وقفه در واحد انتقال، مورد استفاده قرار می گیرند. پروسیجر `sleep` زمانی فراخوانی می شود که واحد انتقال منطقاً در انتظار وقوع یک رخداد (مثلاً دریافت یک بسته) باشد. پس از فراخوانی `sleep`، اجرای واحد انتقال (و به تبع آن پروسه کاربردی) متوقف می شود.

کُد برنامه «واحد انتقال» در شکل ۶-۲۰ نشان داده شده است. هر اتصال همیشه در یکی از هفت وضعیت زیر قرار دارد:

۱. وضعیت IDLE: هنوز هیچ اتصالی برقرار نشده است.
۲. وضعیت WAITING: تابع CONNECT اجرا و تقاضای برقراری اتصال (CALL REQUEST) ارسال شده است.
۳. وضعیت QUEUED: تقاضای CALL REQUEST دریافت شده ولیکن هنوز LISTEN نشده است.
۴. وضعیت ESTABLISHED: اتصال برقرار شده است.
۵. وضعیت SENDING: کاربر منتظر دریافت مجوز ارسال یک بسته است.
۶. وضعیت RECEIVING: تابع RECEIVE اجرا و در حال دریافت بسته است.
۷. وضعیت DISCONNECTING: تابع DISCONNECT جهت قطع اتصال بصورت محلی اجرا شده است.

۱. یعنی بجای آنکه واحد انتقال مستقیماً خودش بسته هایی را که قرار است بر روی شبکه ارسال شوند، تولید کند پارامترهای لازم را به لایه شبکه تحویل می دهد تا اینکار در لایه شبکه انجام شود. -م

۲. Virtual Circuit's Sliding Window

```

#define MAX_CONN 32          /* max number of simultaneous connections */
#define MAX_MSG_SIZE 8192    /* largest message in bytes */
#define MAX_PKT_SIZE 512     /* largest packet in bytes */
#define TIMEOUT 20
#define CRED 1
#define OK 0

#define ERR_FULL -1
#define ERR_REJECT -2
#define ERR_CLOSED -3
#define LOW_ERR -3

typedef int transport_address;
typedef enum {CALL_REQ,CALL_ACC,CLEAR_REQ,CLEAR_CONF,DATA_PKT,CREDIT} pkt_type;
typedef enum {IDLE,WAITING,QUEUED,ESTABLISHED,SENDING,RECEIVING,DISCONN} cstate;

/* Global variables. */
transport_address listen_address; /* local address being listened to */
int listen_conn; /* connection identifier for listen */
unsigned char data[MAX_PKT_SIZE]; /* scratch area for packet data */

struct conn {
    transport_address local_address, remote_address;
    cstate state; /* state of this connection */
    unsigned char *user_buf_addr; /* pointer to receive buffer */
    int byte_count; /* send/receive count */
    int clr_req_received; /* set when CLEAR_REQ packet received */
    int timer; /* used to time out CALL_REQ packets */
    int credits; /* number of messages that may be sent */
} conn[MAX_CONN + 1]; /* slot 0 is not used */

void sleep(void); /* prototypes */
void wakeup(void);
void to_net(int cid, int q, int m, pkt_type pt, unsigned char *p, int bytes);
void from_net(int *cid, int *q, int *m, pkt_type *pt, unsigned char *p, int *bytes);

int listen(transport_address t)
{ /* User wants to listen for a connection. See if CALL_REQ has already arrived. */
    int i, found = 0;

    for (i = 1; i <= MAX_CONN; i++) /* search the table for CALL_REQ */
        if (conn[i].state == QUEUED && conn[i].local_address == t) {
            found = i;
            break;
        }

    if (found == 0) {
        /* No CALL_REQ is waiting. Go to sleep until arrival or timeout. */
        listen_address = t; sleep(); i = listen_conn;
    }
    conn[i].state = ESTABLISHED; /* connection is ESTABLISHED */
    conn[i].timer = 0; /* timer is not used */
}

```

```

listen_conn = 0;                                /* 0 is assumed to be an invalid address */
to_net(i, 0, 0, CALL_ACC, data, 0);             /* tell net to accept connection */
return(i);                                       /* return connection identifier */
}

int connect(transport_address l, transport_address r)
{ /* User wants to connect to a remote process; send CALL_REQ packet. */
  int i;
  struct conn *cptr;

  data[0] = r; data[1] = l;                      /* CALL_REQ packet needs these */
  i = MAX_CONN;                                  /* search table backward */
  while (conn[i].state != IDLE && i > 1) i = i - 1;
  if (conn[i].state == IDLE) {
    /* Make a table entry that CALL_REQ has been sent. */
    cptr = &conn[i];
    cptr->local_address = l; cptr->remote_address = r;
    cptr->state = WAITING; cptr->clr_req_received = 0;
    cptr->credits = 0; cptr->timer = 0;
    to_net(i, 0, 0, CALL_REQ, data, 2);
    sleep();                                     /* wait for CALL_ACC or CLEAR_REQ */
    if (cptr->state == ESTABLISHED) return(i);
    if (cptr->clr_req_received) {
      /* Other side refused call. */
      cptr->state = IDLE;                         /* back to IDLE state */
      to_net(i, 0, 0, CLEAR_CONF, data, 0);
      return(ERR_REJECT);
    }
  } else return(ERR_FULL);                       /* reject CONNECT: no table space */
}

int send(int cid, unsigned char bufptr[], int bytes)
{ /* User wants to send a message. */
  int i, count, m;
  struct conn *cptr = &conn[cid];

  /* Enter SENDING state. */
  cptr->state = SENDING;
  cptr->byte_count = 0;                           /* # bytes sent so far this message */
  if (cptr->clr_req_received == 0 && cptr->credits == 0) sleep();
  if (cptr->clr_req_received == 0) {
    /* Credit available; split message into packets if need be. */
    do {
      if (bytes - cptr->byte_count > MAX_PKT_SIZE) /* multipacket message */
        count = MAX_PKT_SIZE; m = 1; /* more packets later */
      } else {
        /* single packet message */
        count = bytes - cptr->byte_count; m = 0; /* last pkt of this message */
      }
      for (i = 0; i < count; i++) data[i] = bufptr[cptr->byte_count + i];
      to_net(cid, 0, m, DATA_PKT, data, count); /* send 1 packet */
      cptr->byte_count = cptr->byte_count + count; /* increment bytes sent so far */
    } while (cptr->byte_count < bytes); /* loop until whole message sent */
  }
}

```

```

cptr->credits--; /* each message uses up one credit */
cptr->state = ESTABLISHED;
return(OK);
} else {
cptr->state = ESTABLISHED;
return(ERR_CLOSED); /* send failed: peer wants to disconnect */
}
}

int receive(int cid, unsigned char bufptr[], int *bytes)
{ /* User is prepared to receive a message. */
struct conn *cptr = &conn[cid];

if (cptr->clr_req_received == 0) {
/* Connection still established; try to receive. */
cptr->state = RECEIVING;
cptr->user_buf_addr = bufptr;
cptr->byte_count = 0;
data[0] = CRED;
data[1] = 1;
to_net(cid, 1, 0, CREDIT, data, 2); /* send credit */
sleep(); /* block awaiting data */
*bytes = cptr->byte_count;
}
cptr->state = ESTABLISHED;
return(cptr->clr_req_received ? ERR_CLOSED : OK);
}

int disconnect(int cid)
{ /* User wants to release a connection. */
struct conn *cptr = &conn[cid];

if (cptr->clr_req_received) { /* other side initiated termination */
cptr->state = IDLE; /* connection is now released */
to_net(cid, 0, 0, CLEAR_CONF, data, 0);
} else { /* we initiated termination */
cptr->state = DISCONN; /* not released until other side agrees */
to_net(cid, 0, 0, CLEAR_REQ, data, 0);
}
return(OK);
}

void packet_arrival(void)
{ /* A packet has arrived, get and process it. */
int cid; /* connection on which packet arrived */
int count, i, q, m;
pkt_type ptype; /* CALL_REQ, CALL_ACC, CLEAR_REQ, CLEAR_CONF, DATA_PKT, CREDIT */
unsigned char data[MAX_PKT_SIZE]; /* data portion of the incoming packet */
struct conn *cptr;

from_net(&cid, &q, &m, &ptype, data, &count); /* go get it */
cptr = &conn[cid];

```

```

switch (ptype) {
    case CALL_REQ: /* remote user wants to establish connection */
        cptr->local_address = data[0]; cptr->remote_address = data[1];
        if (cptr->local_address == listen_address) {
            listen_conn = cid; cptr->state = ESTABLISHED; wakeup();
        } else {
            cptr->state = QUEUED; cptr->timer = TIMEOUT;
        }
        cptr->clr_req_received = 0; cptr->credits = 0;
        break;
    case CALL_ACC: /* remote user has accepted our CALL_REQ */
        cptr->state = ESTABLISHED;
        wakeup();
        break;
    case CLEAR_REQ: /* remote user wants to disconnect or reject call */
        cptr->clr_req_received = 1;
        if (cptr->state == DISCONN) cptr->state = IDLE; /* clear collision */
        if (cptr->state == WAITING || cptr->state == RECEIVING || cptr->state == SENDING) wakeup();
        break;
    case CLEAR_CONF: /* remote user agrees to disconnect */
        cptr->state = IDLE;
        break;
    case CREDIT: /* remote user is waiting for data */
        cptr->credits += data[1];
        if (cptr->state == SENDING) wakeup();
        break;
    case DATA_PKT: /* remote user has sent data */
        for (i = 0; i < count; i++) cptr->user_buf_addr[cptr->byte_count + i] = data[i];
        cptr->byte_count += count;
        if (m == 0) wakeup();
}
}

void clock(void)
{ /* The clock has ticked, check for timeouts of queued connect requests. */
    int i;
    struct conn *cptr;
    for (i = 1; i <= MAX_CONN; i++) {
        cptr = &conn[i];
        if (cptr->timer > 0) { /* timer was running */
            cptr->timer--;
            if (cptr->timer == 0) { /* timer has now expired */
                cptr->state = IDLE;
                to_net(i, 0, 0, CLEAR_REQ, data, 0);
            }
        }
    }
}
}

```

شکل ۶-۲۰. مثالی از کد برنامه واحد انتقال (Transport Entity).

گذار از یک وضعیت به وضعیتی دیگر زمانی اتفاق می افتد که یکی از وقایع ذیل اتفاق بیفتد: یک تابع اولیه اجرا شود، بسته ای دریافت شود یا مهلت تایمر منقضی شود.

پروسیجرهای نشان داده شده در شکل ۶-۲۰ بر دو نوعند: اغلب آنها مستقیماً قابل فراخوانی در برنامه کاربر هستند در حالیکه *Packet-arrival* و *clock* متفاوتند. این دو تابع در اثر رخدادهای خارجی به صورت خودکار شروع به کار می کنند: اولی در اثر دریافت یک بسته و دومی با هر تیک ساعت به کار می افتد. در حقیقت این دو، روتینهای وقفه (Interrupt Routines) هستند. فرض خواهیم کرد که این دو تابع در حین اجرای هیچیک از پروسیجرهای واحد انتقال فراخوانی نمی شوند بلکه فقط زمانی که پروسه کاربر در حالت توقف (sleeping) یا در حال اجرای کدی غیر از پروسیجرهای لایه انتقال باشد، فراخوانی می شوند. این ویژگی برای عملکرد صحیح کد برنامه، الزامی و حیاتی است.

وجود بیت *Q* (Qualifier) در سرآیند بسته اجازه می دهد که از سربار پروتکل لایه انتقال جلوگیری شود: پیامهای حاوی داده های معمولی، به صورت بسته هایی با $Q=0$ ارسال می شوند؛ پیامهای کنترلی لایه انتقال (که در مثال ما فقط یک پیام و آن هم CREDIT است) در قالب بسته ای ارسال می شوند که در آنها $Q=1$ است. این پیامهای کنترلی در سمت گیرنده توسط «واحد انتقال» کشف و پردازش می شوند.

ساختمان داده اصلی که توسط واحد انتقال مورد استفاده قرار می گیرد، آرایه *conn* است که به ازای هر اتصال یک رکورد در آن ذخیره می شود. این رکورد وضعیت فعلی یک اتصال را نگه می دارد و شامل اطلاعات ذیل است: (۱) آدرس انتقال طرف مقابل (یعنی آدرس TSAP طرف مقابل) (۲) تعداد پیامهای ارسالی یا دریافتی از آن اتصال (۳) وضعیت جاری [یکی از هفت وضعیت] (۴) اشاره گر آدرس بافر کاربر (۵) تعداد بایتهایی که تاکنون از پیام فعلی ارسال یا دریافت شده اند. (۶) یک بیت که مشخص می کند کاربر راه دور، دستور DISCONNECT را صادر کرده است. (۷) یک «شمارنده مجوز» (Permission Counter) که ارسال پیامها را فعال و ممکن می سازد. البته در مثال ساده ما از تمام این فیلدها استفاده نمی شود بلکه یک «واحد انتقال» کامل و دقیق به این فیلدها و حتی فیلدهای بیشتر، نیاز خواهد داشت. فرض بر آنست که هر درایه از آرایه *conn* در وضعیت اولیه IDLE قرار می گیرد. [یعنی با IDLE مقداردهی اولیه می شود].

وقتی کاربری تابع CONNECT را فراخوانی می کند، به لایه شبکه دستور داده می شود که یک بسته از نوع CALL REQUEST برای ماشین راه دور بفرستد؛ سپس کاربر به حالت توقف (استراحت) می رود. وقتی بسته CALL REQUEST در طرف مقابل دریافت شود، به «واحد انتقال» وقفه (اینترپت) داده می شود تا با اجرای روتین *packet_arrival* بررسی کند که آیا یک کاربر محلی به آدرس مشخص شده گوش می دهد یا خیر؛ اگر اینگونه باشد، بسته CALL ACCEPTED بازگردانده شده و کاربر راه دور مجدداً فعال می شود. اگر اینگونه نباشد، CALL REQUEST در صف انتظار قرار می گیرد تا ساعت، به تعداد TIMEOUT تیک بزنند. اگر در خلال این مدت عمل LISTEN انجام شود، اتصال برقرار خواهد شد؛ در غیر این صورت مهلت زمان، منقضی و با ارسال بسته CLEAR REQUEST این اتصال رد خواهد شد تا طرف مقابل تا ابد در حالت توقف نماند.

اگرچه در مثالمان سرآیند پروتکل انتقال را حذف کرده ایم ولی بهرحال به راهی نیاز داریم تا بتوان متوجه شد که هر بسته متعلق به کدام اتصال است، چراکه ممکن است بطور همزمان چندین اتصال فعال وجود داشته باشد. ساده ترین راه حل آن است که از شماره مدار مجازی در لایه شبکه به عنوان شماره اتصال در لایه انتقال استفاده شود. مضاف بر این، از شماره مدار مجازی می توان به عنوان اندیس آرایه *conn* بهره گرفت. یعنی وقتی بسته ای به مدار مجازی شماره k در لایه شبکه وارد شود متعلق به اتصال شماره k در لایه انتقال می باشد، و وضعیت این اتصال در رکورد *conn[k]* قرار گرفته است. برای اتصالاتی که از یک ماشین میزبان شروع می شود، شماره اتصال

توسط «واحد انتقال مبدا»^۱ انتخاب می شود. برای تقاضاهای اتصال، لایه شبکه این انتخاب را انجام می دهد و طبقاً یک شماره بلااستفاده از بین شماره های مدار مجازی انتخاب می کند.

برای آن که مجبور به مدیریت بافرها در درون واحد انتقال نباشیم از یک مکانیزم کنترل جریان خاص و متفاوت از روش معمولی پنجره لغزان استفاده کرده ایم. وقتی یک کاربر، تابع RECEIVE را فراخوانی می کند یک پیام خاص به نام CREDIT (پیام اعتبار) برای واحد انتقال در ماشین فرستنده ارسال می شود و در آرایه conn ثبت می شود. وقتی تابع SEND فراخوانی شود، واحد انتقال ابتدا بررسی می کند که آیا برای اتصال مربوطه «اعتبار» لازم وجود دارد یا خیر؛ اگر اعتبار لازم وجود داشته باشد، پیام ارسال می شود (در صورت لزوم در چند بسته)؛ سپس به اندازه طول پیام از اعتبار موجود کاسته می شود؛ در غیر این صورت، واحد انتقال خودش را به حالت توقف (Sleep) می برد تا اعتبار لازم برسد. این مکانیزم تضمین می کند که هیچ پیامی ارسال نشود مگر آن که طرف مقابل قبلاً فرمان RECEIVE را اجرا کرده باشد. لذا وقتی پیامی از راه برسد مطمئناً بافر لازم برای آن موجود و آماده خواهد بود. این روش را می توان براحتی تعمیم داد تا گیرندگان بتوانند چندین بافر تهیه کرده و تقاضای چند پیام بدهند.

سادگی کد برنامه نشان داده شده در شکل ۶-۲۰ را به خاطر بسپارید. یک واحد انتقال واقعی باید اعتبار تمام پارامترهای عرضه شده توسط کاربر را بررسی نماید، جبران از کار افتادگی در لایه شبکه را بر عهده بگیرد، برخورد دو تماس (Call Collisions) را مدیریت کند و از سرویسهای انتقال عمومی تری شامل تسهیلات وقفه، سرویس دیتاگرام و «نسخه های غیرقابل تعلیق SEND و RECEIVE»^۲ پشتیبانی نماید.

۶-۳-۳ بررسی مثال فوق از دید «ماشین حالت محدود»

نوشتن یک برنامه برای «واحد انتقال» خصوصاً برای پروتکل های واقعی و عملی تر، بسیار دشوار است. برای آن که احتمال خطا کاهش یابد، توصیف وضعیت پروتکل در قالب یک «ماشین حالت محدود» (Finite State Machine) اغلب سودمند است. قبلاً دیدیم که در پروتکل مثال ما، برای هر اتصال هفت وضعیت وجود دارد. در مجموع می توان ۱۲ رخداد مختلف تعریف کرد که می توانند وضعیت یک اتصال را تغییر بدهند. پنج تا از این رخدادهای، در اثر فراخوانی توابع اولیه حادث می شوند. شش تای بعدی در اثر ورود ۶ بسته معتبر رخ می دهند. آخرین رخداد ناشی از انقضای مهلت تایمر است. در شکل ۶-۲۱ فعالیتهای اصلی پروتکل به شکل ماتریسی نشان داده شده است. ستونها بیانگر «وضعیت» و سطرها بیانگر ۱۲ «رخداد» مختلف است.

هر درایه (Entry) در ماتریس شکل ۶-۲۱ (یا به عبارتی در ماشین حالت محدود) حداکثر سه فیلد دارد: (۱) گزاره (Predicate) (۲) کنش (Action) (۳) وضعیت جدید (new state)

«گزاره» مشخص می کند که تحت چه شرایطی «کنش» تعیین شده انجام خواهد گرفت. به عنوان نمونه، درایه گوشه بالا و سمت چپ ماتریس نشان می دهد که اگر LISTEN اجرا شود ولی در جدول، فضای حافظه باقی نمانده باشد (گزاره P1)، اجرای LISTEN با شکست مواجه شده و وضعیت پروتکل تغییر نخواهد کرد. از طرف دیگر، اگر یک بسته درخواست CALL REQUEST برای آدرسی که یک پروسه در حال گوش دادن به آن است، بیاید (گزاره P2)، اتصال مربوطه بلافاصله برقرار می شود. حالت ممکن بعدی آن است که گزاره P2 نادرست باشد یعنی هیچ بسته CALL REQUEST دریافت نشده باشد که در این حالت، اتصال در وضعیت بیکار (IDLE) و در انتظار بسته CALL REQUEST باقی خواهد ماند.

اشاره به این نکته مهم است که انتخاب وضعیتهایی که در ماتریس فوق بکار رفته است، در برگزیده تمام

		State						
		Idle	Waiting	Queued	Established	Sending	Receiving	Dis-connecting
Primitives (عملگرهای اولیه)	LISTEN	P1: ~Idle P2: A1/Estab P2: A2/Idle		~Estab				
	CONNECT	P1: ~Idle P1: A3/Wait						
	DISCONNECT				P4: A5/Idle P4: A6/Disc			
	SEND				P5: A7/Estab P5: A8/Send			
	RECEIVE				A9/Receiving			
Incoming packets (پستهای ورودی)	Call_req	P3: A1/Estab P3: A4/Queue'd						
	Call_acc		~Estab					
	Clear_req		~Idle		A10/Estab	A10/Estab	A10/Estab	~Idle
	Clear_conf							~Idle
	DataPkt						A12/Estab	
Clock	Credit				A11/Estab	A7/Estab		
	Timeout			~Idle				

Predicates

P1: Connection table full
P2: Call_req pending
P3: LISTEN pending
P4: Clear_req pending
P5: Credit available

Actions

A1: Send Call_acc A7: Send message
A2: Wait for Call_req A8: Wait for credit
A3: Send Call_req A9: Send credit
A4: Start timer A10: Set Clr_req_received flag
A5: Send Clear_conf A11: Record credit
A6: Send Clear_req A12: Accept message

شکل ۶-۲۱. مدل پروتکل مثال قبل در قالب «ماشین حالت محدود». هر درایه (Entry) یک «گزاره» اختیاری، یک «کنش» اختیاری و یک «وضعیت» جدید دارد. علامت ~ بیانگر آنست که هیچ «کنشی» صورت نمی گیرد. علامت - بر روی یک گزاره، منفی آنرا نشان می دهد. درایه های خالی مربوط به رخداد های ناممکن یا نامعتبر هستند.

وضعیت های ممکن نیست: در این مثال، وضعیت LISTENING (یعنی وضعیت انتظار برای برقراری اتصال) وجود ندارد در حالی که این وضعیت، پس از صدور فرمان LISTEN اتفاق می افتد و وضعیتی معقول به شمار می رود. چرا این وضعیت لحاظ نشده است؟ زیرا تصمیم گرفته ایم که از شماره مدار مجازی بکار رفته در لایه شبکه به عنوان شناسه اتصال (Connection Identifier) استفاده کنیم و به واسطه صدور فرمان LISTEN هیچ رکورد اتصال خاصی که مبین وضعیت LISTENING باشد در جدول اتصالات ایجاد نخواهد شد. یعنی برای LISTEN شماره مدار مجازی، نهایتاً توسط لایه شبکه و در هنگام ورود بسته CALL REQUEST انتخاب می شود. کنشهای A1 تا A12، عملیات اصلی تلقی می شوند؛ عملیاتی نظیر ارسال بسته و راه اندازی تایمرها. عملیات

کوچک و جانبی مثل مقداردهی اولیه به فیلدهای یک رکورد اتصال، فهرست نشده‌اند. اگر یک «کنش» منوط به فعال کردن یک پروسه معلق باشد، عملیات لازم پس از فعال‌سازی پروسه نیز جزو آن کنش محسوب می‌شود: به عنوان مثال هرگاه بسته CALL REQUEST وارد شود و پروسه‌ای در انتظار آن معلق و منتظر مانده باشد، ارسال بسته CALL ACCEPT که بلافاصله پس از فعال شدن پروسه انجام می‌گیرد به عنوان بخشی از عمل CALL REQUEST محسوب می‌شود. پس از انجام هر عمل، اتصال در وضعیت جدیدی قرار می‌گیرد. (به شکل ۶-۲۱ دقت کنید.)

توصیف عملکرد پروتکل در قالب ماتریس سه مزیت دارد: اول آن که بدین شکل برنامه‌نویس ساده‌تر می‌تواند ترکیبات مختلف «وضعیت» و «رخدادها» را بررسی کرده و نیاز به یک «کنش» را تشخیص بدهد. در پیاده‌سازی عملی و واقعی پروتکل، برخی از این ترکیبات (که اکنون به حال خود رها شده‌اند) برای مدیریت خطا کاربرد دارند. در شکل ۶-۲۱ هیچ تمایزی بین «وضعیت‌های غیرممکن» و «وضعیت‌های نامعتبر» قائل نشده‌ایم.^۱ به عنوان مثال هرگاه یک اتصال در وضعیت «انتظار» (waiting) قرار داشته باشد، رخداد DISCONNECT غیرممکن است چراکه در وضعیت انتظار، پروسه کاربر متوقف شده و هرگز نمی‌تواند هیچ تابعی را اجرا نماید. برعکس، در وضعیت sending (ارسال)، انتظار دریافت بسته نمی‌رود زیرا هیچ اعتباری جهت ارسال برای طرف مقابل صادر نشده است و دریافت یک بسته داده، خطای پروتکل محسوب می‌شود.

دومین مزیت نمایش ماتریسی پروتکل، پیاده‌سازی آنست. برنامه‌نویس می‌تواند آرایه‌ای دو بعدی در نظر بگیرد که در هر عنصر آن یعنی $a[i][j]$ ، اشاره‌گر شروع یا اندیس پروسه‌یجری قرار گرفته که هنگام بروز رخداد i در وضعیت j ، باید اجرا شود و اداره امور را بر عهده بگیرد. یکی از راهکارهای پیاده‌سازی «واحد انتقال» آنست که برنامه را در قالب یک حلقه کوتاه (حلقه تکرار در برنامه‌نویسی) بنویسیم که در ابتدای حلقه منتظر وقوع یک رخداد باقی می‌ماند. وقتی یک رخداد اتفاق می‌افتد اتصال مربوطه شناسایی و تعیین شده و وضعیت آن استخراج می‌گردد. براساس وضعیت و رخدادهایی که در این ماتریس تبیین شده، «واحد انتقال» اندیس مناسب را استخراج و با رجوع به آرایه a ، پروسه‌یجری مناسب را فراخوانی و اجرا می‌نماید.

حسن سوم در روش «ماشین حالت محدود»، سادگی توصیف پروتکل است. در مستندات برخی از استانداردها، هر پروتکل در قالب یک «ماشین حالت محدود» (همانند آنچه که در شکل ۶-۲۲ نشان داده شده)، توصیف و تشریح شده است. هرگاه واحد انتقال در قالب یک ماشین حالت محدود توصیف شده باشد راحت‌تر می‌توان از طریق آن به سوی پیاده‌سازی عملی آن حرکت کرد.

اشکال روش «ماشین حالت محدود» آن است که فهم و تحلیل آن دشوارتر از برنامه‌نویسی معمولی است. البته این اشکال را می‌توان با ترسیم ماشین حالت محدود به صورت گراف، تا حدودی حل کرد. این نوع نمایش در شکل ۶-۲۲ نشان داده شده است.

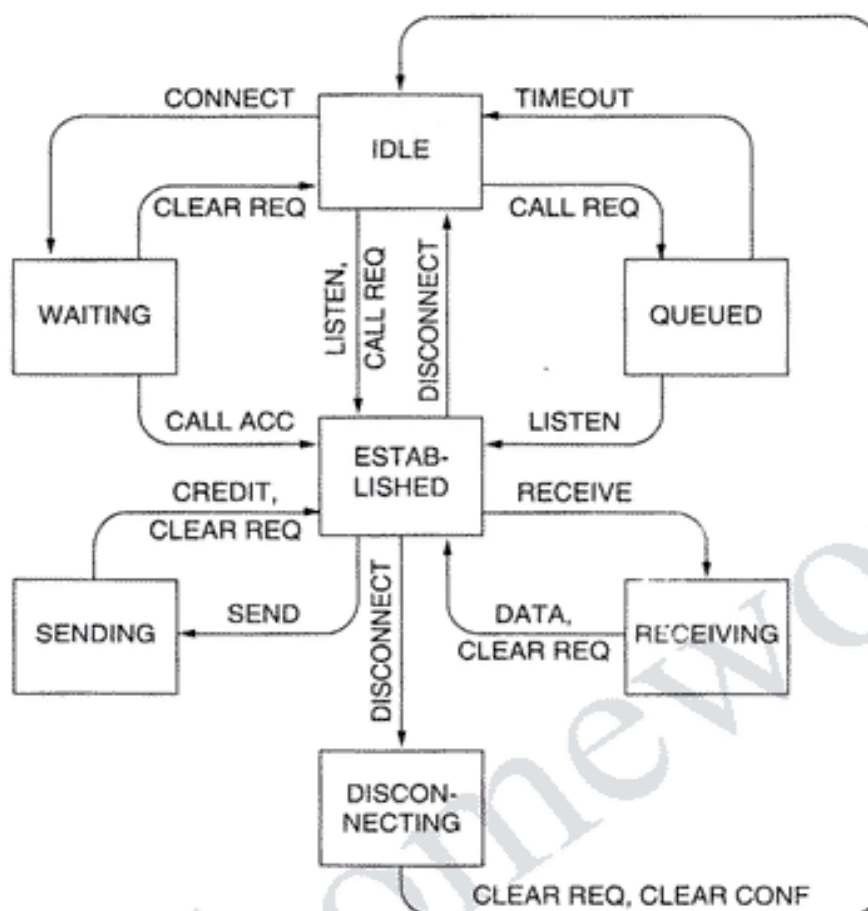
۴ پروتکل‌های لایه انتقال در اینترنت: UDP^۲

اینترنت در لایه انتقال دو پروتکل عمده دارد: یکی بدون اتصال و دیگری اتصال‌گرا. در بخش‌های آتی به مطالعه این دو پروتکل خواهیم پرداخت. پروتکل بدون اتصال در اینترنت، UDP و پروتکل اتصال‌گرای آن، TCP نام دارد. از آنجایی که UDP اساساً همان IP به همراه یک سرآیند کوتاه است لذا با آن شروع خواهیم کرد. در ضمن به دو کاربرد مختلف UDP خواهیم پرداخت.

۱. «وضعیت‌های غیرممکن» یعنی وضعیت‌هایی که هرگز اتفاق نخواهد افتاد و «وضعیت‌های نامعتبر» وضعیت‌هایی است که نباید اتفاق

۲. User Datagram Protocol

بیفتد! - م

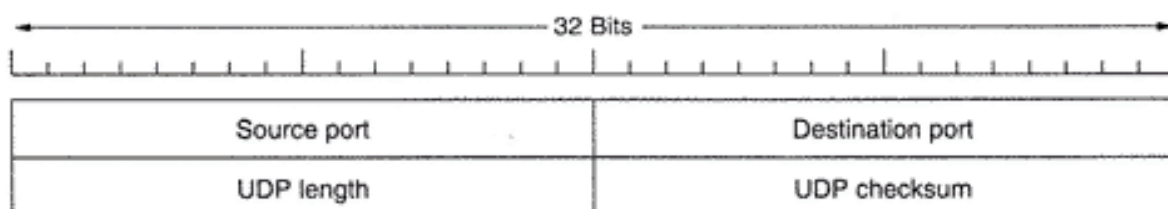


شکل ۶-۲۲. مدل پروتکل مثال قبلی در قالب گراف. برای سادگی «گذارهایی» (Transitions) که وضعیت اتصال را تغییر نمی دهند از گراف حذف شده اند.

۱-۴-۶ مقدمه ای بر UDP

پشته پروتکلی اینترنت در لایه انتقال از یک پروتکل بدون اتصال به نام UDP پشتیبانی می کند. UDP امکان آن را فراهم آورده که برنامه های کاربردی بتوانند بدون ایجاد هر گونه اتصال و هماهنگی قبلی، داده ای را درون یک دیتاگرام IP جاسازی کرده و آن را بفرستند. پروتکل UDP در RFC 768 تشریح شده است.

UDP داده ها را در قالب قطعاتی^۱ ارسال می کند که در ابتدای آنها ۸ بایت سرآیند و سپس داده های لایه کاربرد قرار می گیرد. این سرآیند در شکل ۶-۲۳ نشان داده شده است. دو فیلد شماره پورت^۲ به منظور شناسایی نقاط پایانی (پرونده های نهایی) در ماشینهای مبدا و مقصد به کار می آیند. وقتی یک بسته UDP از راه می رسد، محتوای آن به پرونده متصل^۳ به شماره پورت مقصد، تحویل داده می شود. عمل اتصال پرونده به یک پورت از طریق تابع اولیه BIND (که تعریف آنرا برای TCP در شکل ۶-۶ دیدیم) انجام می شود. (فرآیند مقیدسازی پرونده به یک پورت در TCP یا UDP تفاوتی ندارد.) در حقیقت، آنچه که UDP در مقایسه با IP معمولی اضافه تر دارد پورتهای مبدا و مقصد هستند. بدون فیلدهای مربوط به پورت، لایه انتقال نمی داند که با یک بسته چه کار کند. با این فیلدها، قطعه داده به درستی تحویل پرونده مربوطه خواهد شد.



شکل ۶-۲۳. سرآیند UDP (UDP Header).

برای آن که بتوان برای پروسه مبداء پاسخی برگرداند، به شماره پورت مبداء نیاز است. بدین منظور محتوای فیلد پورت مبداء (Source Port) از بسته ورودی، در فیلد پورت مقصد از بسته خروجی، کپی و ارسال می شود. بدین ترتیب فرستنده پاسخ، پروسه تحویل گیرنده بسته را مشخص می نماید.

فیلد UDP Length اندازه کل قطعه داده (شامل ۸ بایت سرآیند) را مشخص می نماید. فیلد UDP checksum به منظور کشف خطاهای احتمالی کاربرد دارد و اختیاری است؛ در صورت عدم محاسبه، در این فیلد عدد صفر درج می شود. (البته اگر مقدار واقعی این کد صفر باشد به جای آن تمام بیت های ۱ پر می شوند.) عدم استفاده از این فیلد نوعی سهل انگاری است مگر آن که کیفیت داده ها چندان مهم نباشد. (مثلاً در مورد صدای دیجیتال)

شاید اشاره صریح به کارهایی که UDP انجام نمی دهد، ارزشمند باشد. این پروتکل کنترل جریان (Flow Control) و کنترل خطا (Error Control) ندارد و در صورت دریافت یک قطعه داده خراب، آن را «ارسال مجدد» نخواهد کرد. تمام این عملیات بر عهده پروسه کاربر گذاشته شده است. آنچه که این پروتکل انجام می دهد عبارت است از ایجاد یک واسطه (Interface) بین پروسه های کاربردی و پروتکل IP و انجام عمل دی مالتی پلکس قطعات داده بین پروسه های کاربردی.^۱ این تمام کاری است که UDP انجام می دهد. برای برنامه های کاربردی که نیاز به کنترل دقیق و جدی جریان، کنترل خطا و زمان بندی دارند، UDP امکانات چندانی را در اختیار نمی گذارد و آن را به خود برنامه محول کرده است.

یکی از زمینه هایی که استفاده از UDP مفید است، برنامه های خاص سرویس دهنده/مشتري هستند که در آنها مشتري تقاضای کوتاهی را برای سرویس دهنده می فرستد و انتظار پاسخی کوتاه دارد. اگر پیام تقاضا یا پاسخ از بین برود، مهلت مشتري به سر می آید و از نو شروع می کند. در این حالت، نه تنها کد برنامه ساده تر می شود بلکه به مبادله پیامهای کمتری در دو جهت نیاز است.

یکی از برنامه های کاربردی که از UDP بهره گرفته، DNS^۲ است که آن را در فصل هفتم مطالعه خواهیم کرد. خلاصه عملکرد DNS بدین نحو است: یک برنامه که در جستجوی آدرس IP یک ماشین میزبان با نامی مثل www.cs.berkeley.edu است یک بسته UDP حاوی نام ماشین میزبان به سوی DNS می فرستد. سرویس دهنده DNS، با ارسال یک بسته UDP، آدرس IP ماشین میزبان مورد نظر را به اطلاع ماشین سوال کننده می رساند. در این حالت نیازی به هماهنگی قبلی و ایجاد یا ختم اتصال نیست و در مجموع فقط دو پیام رد و بدل می شود.

۲-۴-۶ فراخوانی پروسیجرهای راه دور (RPC : Remote Procedure Call)

از برخی جهات، ارسال یک پیام برای یک ماشین راه دور و دریافت پاسخ، با فراخوانی یک تابع در زبانهای

۱. یعنی با بررسی شماره پورت مقصد از هر بسته UDP آنرا بین پروسه های مربوطه توزیع می نماید. م-

۲. Domain Name System

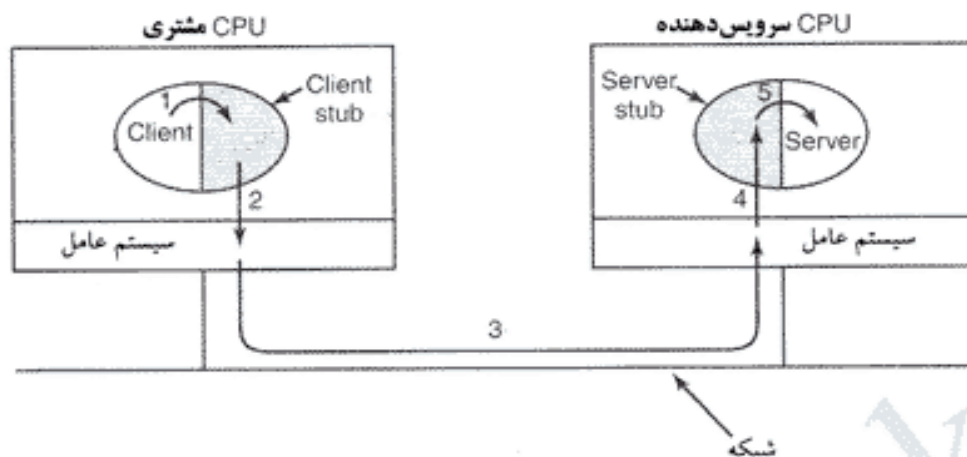
برنامه نویسی مشابهت دارد. در هر دو مورد یک یا چند پارامتر را تحویل می دهید و نتیجه ای به شما بازگردانده می شود. این شباهت، افراد را بدین نکته رهنمون ساخت که عملیات مبتنی بر «پرسش و پاسخ»^۱ در شبکه را می توان در قالب فراخوانی پروسیجر سازماندهی و تنظیم کرد. چنین ساختاری، برنامه نویسی برنامه های کاربردی شبکه را ساده تر و از لحاظ مفهومی آشناتر و ملموس تر می کند. به عنوان مثال، پروسیجر به نام `get_IP_address(host_name)` را مد نظر قرار بدهید که با ارسال یک بسته UDP برای سرویس دهنده DNS کار خود را آغاز کرده و منتظر پاسخ آن باقی می ماند و اگر پاسخ آن در مهلت مقرر دریافت نشود، تلاش خود را تکرار می کند. بدین ترتیب تمام جزئیات شبکه از دید برنامه نویس مخفی می ماند.

در این زمینه کارهایی اساسی توسط دو نفر به نامهای Birrell و Nelson (۱۹۸۴) انجام شده است. چکیده آنچه که Birrell و Nelson پیشنهاد کردند آنست که به برنامه ها اجازه داده شود پروسیجرهایی را بر روی ماشینهای راه دور فراخوانی نمایند. وقتی پروسه ای بر روی ماشین ۱، پروسیجر را بر روی ماشین ۲ فراخوانی می کند، پروسه صدازنده بر روی ماشین ۱ متوقف و معلق شده و اجرای پروسیجر بر روی ماشین ۲ آغاز می شود. اطلاعات لازم از پروسه صدازنده، در قالب چند پارامتر تحویل پروسیجر شده و نتیجه اجرای پروسیجر در پارامترهایی بازگردانده می شود. بدین نحو رد و بدل هر گونه پیام، از دید برنامه نویس مخفی می ماند. این تکنیک اصطلاحاً RPC نامیده می شود (Remote Procedure Call) و به زیربنای بسیاری از برنامه های کاربردی شبکه تبدیل شده است. بطور مرسوم پروسیجر صدازنده با عنوان «مشتري» و پروسیجر فراخوانی شده، با عنوان «سرویس دهنده» معرفی می شوند و ما نیز از همین اسامی به یه می گیریم.

ایده ای که در ورای RPC قرار دارد آن است که فراخوانی پروسیجرهای راه دور حتی الامکان شبیه به فراخوانی پروسیجرهای محلی باشد. در ساده ترین شکل ممکن، برای فراخوانی پروسیجرهای راه دور، باید به برنامه مشتری یک پروسیجر کتابخانه ای کوچک به نام `client stub` مقید (bind) شود که پروسیجر سرویس دهنده را در فضای آدرس برنامه مشتری تصویر می کند. به روش مشابه، سرویس دهنده نیز به پروسیجر کوچکی به نام `server stub` مقید می شود که وظیفه آن، پنهان سازی این واقعیت است که فراخوانی پروسیجر توسط یک برنامه مشتری بر روی سرویس دهنده محلی انجام نشده است.

در شکل ۶-۲۴، مراحل واقعی یک فراخوانی RPC نشان داده شده است. در مرحله ۱، برنامه مشتری پروسیجر `client stub` را فراخوانی می کند. این مرحله، یک فراخوانی پروسیجر محلی است و طبق معمول پارامترهای لازم به درون پشته هدایت می شوند. در مرحله ۲، پروسیجر `client stub` این پارامترها را در یک پیام جاسازی کرده و برای ارسال آن یک فراخوانی سیستمی انجام می دهد. جاسازی پارامترها در یک پیام اصطلاحاً عمل «مارشالینگ» (تشریفات) نام دارد. در مرحله سوم هسته سیستم عامل، این پیام را از ماشین مشتری به سوی ماشین سرویس دهنده ارسال می کند. در مرحله چهارم هسته سیستم عامل سرویس دهنده، بسته ورودی را به `server stub` تحویل می دهد. نهایتاً در مرحله پنجم، پروسیجر `server stub` پروسیجر سرویس دهنده را با پارامترهای دریافتی اجرا می نماید. فرآیند ارسال پاسخ در جهت مقابل نیز طبق همین روال انجام خواهد شد.

اشاره به این نکته کلیدی مهم است که در برنامه مشتری که کاربر آن را می نویسد، فراخوانی پروسیجر `client stub` به روش معمولی انجام می شود و نام آن با نام پروسیجر سرویس دهنده یکی است. از آنجایی که برنامه مشتری و پروسیجر `client stub` در فضای آدرس مشابهی قرار دارند لذا تحویل پارامترهای لازم برای فراخوانی پروسیجر به روش معمول (و از طریق پشته) انجام می گیرد. به روش مشابه، پروسیجر سرویس دهنده



شکل ۶-۲۴. مراحل لازم برای عمل فراخوانی پروسجر راه دور (RPC). Client Stub و Server Stub نشان داده نشده است.

توسط پروسجر دیگر یعنی server stub (در فضای آدرس خودش و با پارامترهای لازم) به روش معمولی فراخوانی می‌شود. برای پروسجرهای سرویس دهنده هیچ چیزی غیر معمول و نامتعارف نیست. بدین ترتیب به جای آن که عملیات I/O از طریق سوکت انجام شود، مبادله اطلاعات در شبکه با شبیه سازی و تقلید فراخوانی پروسجر انجام می‌گیرد.

فارغ از زیبایی مفهوم RPC، مار در آستین آن نهان است!! مهمترین مشکل آن پارامترهای نوع اشاره گر هستند. بطور طبیعی، ارسال یک اشاره گر به پروسجرهای محلی، اشکالی ندارد. پروسجر محلی می‌تواند از اشاره گرهای همان نحوی که پروسه صدا زننده از آنها بهره می‌گیرد، استفاده کند چرا که هر دوی آنها در فضای آدرس مجازی یکسانی به سر می‌برند. در RPC، ارسال پارامترهایی که از نوع اشاره گر هستند غیر ممکن است چرا که مشتری و سرویس دهنده در فضای آدرس متفاوتی هستند.

البته در برخی موارد می‌توان برای ارسال اشاره گر به پروسجرها از روشهای زیرکانه بهره گرفت. فرض کنید که اولین پارامتر پروسجر، یک اشاره گر به عدد صحیح k باشد. پروسجر client stub می‌تواند با سازماندهی و تنظیم «مقدار k» در درون یک بسته، آن را برای سرویس دهنده بفرستد. پروسجر server stub پس از ذخیره مقدار k در حافظه، یک اشاره گر برای k ساخته و طبق معمول آن را به پروسجر سرویس دهنده تسلیم می‌کند. وقتی پروسجر سرویس دهنده، کنترل اجرا را به پروسجر server stub برمی‌گرداند، این پروسجر مقدار جدید k را برای مشتری باز می‌گرداند تا مقدار قدیم k با مقدار جدیدی که سرویس دهنده محاسبه کرده، جایگزین شود. در حقیقت مکانیزم «فراخوانی به روش ارجاع»^۱ با روش «فراخوانی از طریق تحویل مقدار و بازایی نتیجه» جایگزین شده است. متأسفانه این راهکار همیشه کار نمی‌کند چرا که اگر اشاره گر مثلاً به یک گراف یا ساختمان داده پیچیده اشاره کرده باشد نمی‌توان از این روش بهره گرفت. به همین دلیل باید بر روی پارامترهایی که برای فراخوانی پروسجرهای راه دور مورد نیاز هستند، محدودیتهایی اعمال کرد.

مسئله دوم آن است که در زبانهایی مثل C که در آن محدودیتهای «نوع داده» (Data Type) چندان قوی نیست نوشتن پروسجرهایی که مثلاً ضرب داخلی دو بردار (دو آرایه) را حساب می‌کند، بدون مشخص کردن بزرگی اندازه آرایه‌ها، کاملاً صحیح و معتبر است. حال مثلاً قرارداد شده که انتهای این آرایه‌ها با یک مقدار ویژه مشخص گردد و

پروسیجر صدازنده و صداشونده هر دو آنرا به رسمیت می شناسند. در چنین شرایطی client stub اساساً نمی تواند چنین پارامترهایی را استخراج، سازماندهی (مارشالینگ) و ارسال نماید چرا که اندازه پارامترها را نمی داند. مسئله سوم آن است که همیشه این امکان وجود ندارد که بتوان نوع داده ای پارامترها را از روی توصیف رسمی (Formal Specification) یا کد برنامه استخراج کرد. مثلاً printf می تواند به هر تعداد پارامتر داشته باشد (حداقل یکی) و پارامترها نیز می توانند مخلوطی از متغیرهای صحیح (از نوع long, short, int)، کاراکتر، رشته ای (string)، اعداد اعشاری با طول دلخواه و انواع دیگر باشند. تلاش در فراخوانی پروسیجر printf از راه دور، عملاً غیر ممکن است چرا که زبان C در خصوص انواع داده آسان می گیرد. از طرفی اگر قانونی وضع شود که استفاده از RPC را منوط به عدم استفاده از C (یا ++C) کند، آنرا از عمومیت و رواج می اندازد.

مسئله چهارم در ارتباط با متغیرهای سراسری برنامه است. بطور طبیعی پروسیجرهای صدازنده و صداشونده می توانند به غیر از پارامترها از طریق متغیرهای سراسری نیز با یکدیگر تبادل داده داشته باشند. اگر پروسیجر فراخوانی شده به یک ماشین راه دور منتقل شود، این کد با شکست مواجه خواهد شد چرا که متغیرهای سراسری در آنجا معتبر و قابل استفاده نیستند.

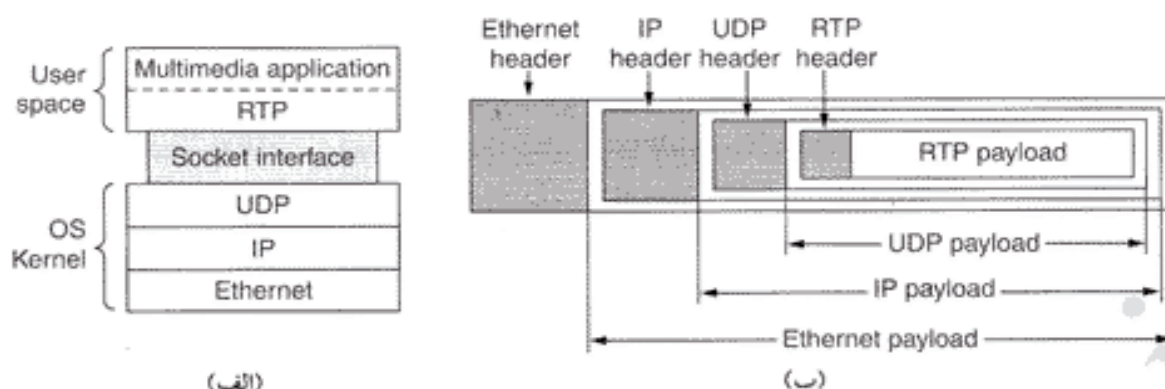
این مسائل بدین معنا نیست که باید از RPC سلب امید کرد. در واقع بطور گسترده ای از آن استفاده می شود ولیکن باید محدودیتهایی را اعمال کرد تا در عمل بدرستی کار کند.

البته RPC لزوماً از UDP استفاده نمی کند ولیکن RPC و UDP زوج متناسبی هستند و عموماً برای RPC از UDP بهره گرفته می شود. علیرغم این، وقتی پارامترها یا نتیجه اجرای پروسیجر از اندازه حداکثر یک بسته UDP بیشتر باشد یا وقتی که عملیات مورد تقاضا قابل تکرار نباشند یا تکرار آن، نتایج نامشخصی به همراه داشته باشد ممکن است به جای استفاده از UDP مجبور به تنظیم یک اتصال TCP و ارسال تقاضا از طریق آن باشیم.

۴-۴-۳ پروتکل انتقال بی درنگ

مدل سرویس دهنده/مشرقی مبتنی بر RPC، زمینه ای است که در آن از UDP استفاده گسترده ای می شود. زمینه دیگر استفاده از UDP، برنامه های کاربردی چند رسانه ای بی درنگ است. خصوصاً با رواج روزافزون رادیوی اینترنتی، تلفن اینترنتی، موزیک درخواستی از شبکه، ویدیو کنفرانس و دیگر کاربردهای چند رسانه ای، عموم افراد دریافته اند که همه در حال حرکت به سوی یک پروتکل انتقال بی درنگ کم و بیش مشابه هستند. به تدریج روشن شد که داشتن یک پروتکل عمومی و استاندارد برای انتقال بی درنگ داده ها ایده خوبی است. بدین ترتیب پروتکل RTP^۱ متولد شد. این پروتکل در سند RFC 1889 تشریح شده و امروزه رواج گسترده ای دارد.

جایگاه RTP در پشته پروتکلی اندکی عجیب به نظر می رسد. تصمیم گرفته شد که RTP در فضای پروسه کاربر قرار گرفته و از طریق UDP اجرا شود. عملکرد RTP به نحو زیر است: برنامه چند رسانه ای حاوی چندین قطعه صدا، ویدیو، متن و احتمالاً استریمهای دیگر می باشد؛ این داده ها به «کتابخانه RTP»^۲ که در فضای برنامه کاربر قرار دارد و به همراه آن اجرا می گردد، خوراند می شود. این کتابخانه، استریم داده ها را مالتی پلکس کرده و آنها را در بسته های RPT جاسازی می کند. سپس آنها را بر روی یک سوکت قرار می دهد. در آنسوی سوکت، (یعنی در هسته سیستم عامل)، بسته های UDP تولید شده و در درون بسته های IP قرار می گیرند. حال اگر کامپیوتر بر روی شبکه اترنت واقع شده باشد، بسته های IP جهت انتقال بر روی کانال، در درون یک فریم اترنت جاسازی می شود. در چنین شرایطی، پشته پروتکلی شبیه به شکل ۶-۲۵-الف خواهد بود. تودرتویی بسته ها نیز در شکل ۶-۲۵-ب نشان داده شده است.



شکل ۶-۲۵. (الف) موقعیت RTP در پشته پروتکلی. (ب) ترتیب تودرتویی بسته‌ها.

در نتیجه این سبک طراحی، به سادگی نمی‌توان گفت که RTP در کدام لایه قرار می‌گیرد. از آنجایی که RTP در فضای برنامه کاربر اجرا می‌شود و نهایتاً به برنامه کاربردی لینک می‌شود فلذا به یک پروتکل لایه کاربرد شبیه است. از طرف دیگر، RTP یک پروتکل مستقل از لایه کاربرد است و امکانات لایه انتقال را عرضه می‌کند لذا شبیه به یک پروتکل لایه انتقال به نظر می‌رسد. شاید بهترین تعبیر آن باشد که: «RTP یک پروتکل لایه انتقال است که در لایه کاربرد پیاده‌سازی شده است».

عملکرد اصلی پروتکل RTP آنست که چندین استریم از داده‌های بی‌درنگ را در یک استریم از بسته‌های UDP مالتی‌پلکس کند. این استریم UDP را می‌توان برای یک ماشین مقصد (یعنی تک‌پخشی) یا چندین ماشین مقصد (یعنی چندپخشی) فرستاد. از آنجایی که RTP صرفاً از بسته‌های معمولی UDP بهره می‌گیرد لذا مسیر یابها با این بسته‌ها به صورت ویژه رفتار نمی‌کنند مگر آن که ویژگی کیفیت خدمات (QoS) در IP پیاده و فعال شود. خصوصاً آنکه هیچ تضمینی در خصوص تحویل این بسته‌ها، میزان تأخیر و لرزش (Jitter) وجود ندارد.

به هر بسته در یک استریم RTP، شماره‌ای داده می‌شود که یک واحد از شماره بسته قبلی بیشتر است. شماره‌گذاری بسته‌ها به مقصد اجازه می‌دهد تا گم شدن احتمالی بسته‌ها را تشخیص بدهد. اگر بسته‌ای از دست برود بهترین کاری که مقصد می‌تواند انجام بدهد آنست که مقادیر بسته از دست رفته را به روش درون‌یابی (Interpolation) تخمین بزند.^۱ ارسال مجدد بسته‌های گم‌شده از لحاظ عملی ممکن نیست زیرا بسته‌هایی که مجدداً ارسال می‌شوند، احتمالاً آنقدر دیر به مقصد می‌رسند که دریافت آنها هیچ فایده‌ای نخواهد داشت. در نتیجه پروتکل RTP هیچ مکانیزمی برای کنترل جریان، کنترل خطا، اعلام وصول بسته (Ack) یا تقاضای ارسال مجدد ندارد.

محتوای بسته‌های RTP می‌تواند انواع مختلف داده را در برگیرد و طبعاً به دلخواه برنامه کاربردی کُد می‌شود. برای جلوگیری از تنوع زیاد و ناسازگاری، RTP چندین پروفایل (مثل استریم صدا) تعریف و برای هر پروفایل چندین روش کدینگ پیشنهاد کرده است. به عنوان مثال یک استریم صدا می‌تواند به روش PCM هشت بیتی با نرخ نمونه‌برداری 8KHz یا روش «مدولاسیون دلتا»^۲، «کدینگ پیشگویانه»^۳، «کدینگ GSM»^۴، روش MP3 یا نظائر آن کُد و ارسال شود. پروتکل RTP فیلدی را در سرآیند هر بسته پیش‌بینی کرده تا به کمک آن مبداء بتواند

۱. درون‌یابی بسته‌های گم‌شده که برای صدا و تصاویر ویدیویی کاربرد دارد بدین نحو است که گیرنده از روی فریمهای قبلی و بعدی بخشی از داده‌های از دست رفته را بصورت تقریبی محاسبه و آنرا جایگزین می‌کند تا کیفیت تصویر غیرقابل تحمل نشود.

۲. Predictive Encoding

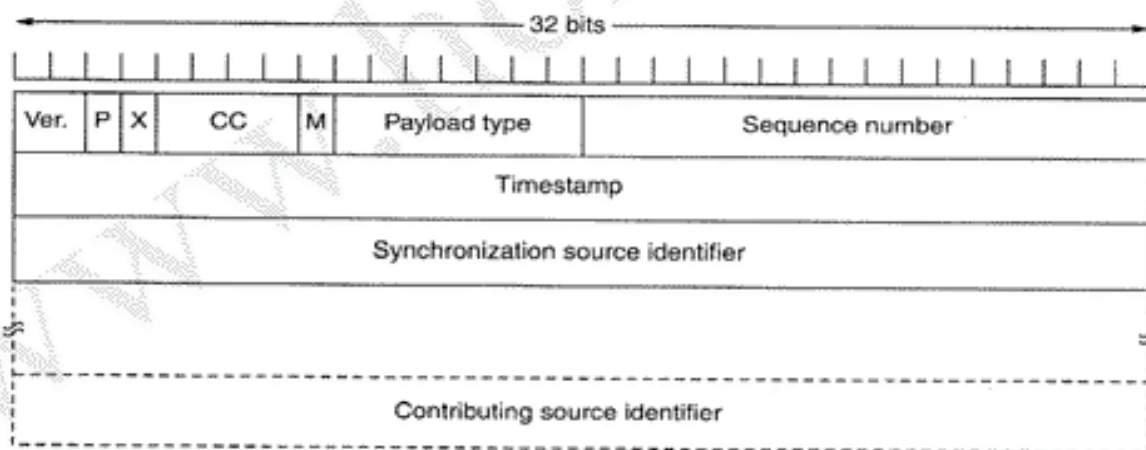
۳. Delta Modulation

نوع کدپنگ خود را مشخص کند.

امکان دیگری که بسیاری از کاربردهای بی درنگ بدان نیاز دارند، «درج مهر زمان» (Timestamping) بر روی بسته هاست. ایده اصلی آنست که به مبداء اجازه بدهیم تا به اولین نمونه^۱ از هر بسته یک مهر زمان نسبت بدهد. مهر زمان نسبت به زمان شروع استریم محاسبه می شود فلذا فقط اختلاف مقادیر مهر زمان بسته ها اهمیت دارد و مقادیر مطلق آنها بی ارزش است. در این مکانیزم، مقصد نیاز کمی به بافرسازی بسته ها خواهد داشت و فارغ از آن که بسته ها چه زمانی دریافت شده اند، براساس مهر زمان درج شده بر روی آنها، سر موعد مقرر اجرا (Play) می شوند.^۲ درج مهر زمان نه تنها تاثیر منفی «لرزش» (Jitter) را کاهش می دهد بلکه امکان آنرا فراهم می آورد تا استریمها از لحاظ زمانی سنکرون بمانند. به عنوان مثال یک برنامه تلویزیون دیجیتالی می تواند یک استریم ویدیو و دو استریم صدا داشته باشد. این دو استریم صدا می توانند یکی برای پخش فیلم با صدای اصلی و دیگری با صدای دوبله شده به زبان محلی باشد تا بیننده به میل خود یکی از آنها را انتخاب کند. هر یک از این استریمها از ابزارهای فیزیکی متفاوتی منشاء می گیرند ولیکن اگر با یک شمارنده واحد شماره گذاری شوند، می توان آنها را صورت هماهنگ و سنکرون اجرا کرد، حتی اگر به صورت نامنظم و پس و پیش ارسال شده باشند.

سرآیند بسته RTP در شکل ۶-۲۶ نشان داده شده است. این سرآیند از سه کلمه ۳۲ بیتی و چند بخش اختیاری توسعه (Extension) تشکیل شده است. اولین کلمه، در برگیرنده این فیلدهاست: فیلد Version که فعلاً ۲ است. امیدواریم که این نسخه نزدیک به نسخه متکامل و نهایی آن باشد چراکه فقط یک شماره بیشتر (یعنی نسخه ۳) باقی نمانده است. (البته می توان عدد ۳ را بدین نحو تعریف و تعبیر کرد که شماره واقعی نسخه پروتکل درون سرآیند اختیاری قرار گرفته است!)

بیت P، بیانگر آنست که به بسته، داده های زائدی اضافه شده تا طول آن ضریبی از ۴ باشد. در حقیقت مقدار آخرین بایت زائد مشخص می کند که چند بایت به داده ها اضافه شده است. ۳ بیت X مشخص می کند که در این بسته



شکل ۶-۲۶. سرآیند RTP (RTP Header).

۱. sample

۲. در ذهن خود، «مهر زمان» را برای صدا (یا ویدیو) بدینگونه تصور کنید: اولین نمونه صدا در لحظه $t=t_0$ تولید شده است. بنابراین فرضاً در مهر زمان اولین بسته ارسالی ۰ درج می شود. زمان تولید بسته های بعدی نسبت به اولین بسته، محاسبه و درون فیلد مهر زمان درج می شود. بنابراین گیرنده حتی اگر بسته ها را بصورت نامنظم دریافت کند براساس این مقدار می تواند زمان دقیق اجرای هر بسته را مشخص نماید. - م

۳. عبارتی بیت P مشخص می کند که حداقل یک بایت زائد به داده ها اضافه شده است. آخرین بایت، تعداد پایت های زائد را مشخص می کند. - م

«سرآیند توسعه» (Extension Header) وجود دارد. قالب و معنای سرآیند توسعه در پروتکل تعریف نشده است. جز آنکه اولین کلمه سرآیند توسعه، طول آنرا مشخص می‌کند. این سرآیند امکان آنرا فراهم آورده تا بتوان نیازهای پیش‌بینی نشده را مرتفع کرد.

فیلد CC بیانگر آنست که چند مبداء در تولید بسته دخالت داشته‌اند (از صفر تا ۱۵). در ادامه به این موضوع خواهیم پرداخت. بیت M یک علامت ویژه برنامه‌های کاربردی است. مثلاً می‌توان از این علامت به معنای شروع اولین فریم ویدیو، شروع اولین کلمه در کانال صوتی (یا هر چه که برنامه کاربردی بخواهد) استفاده کرد.

فیلد Payload Type روش کدینگ مورد استفاده را تعیین می‌کند (به عنوان مثال صدای هشت بیتی فشرده‌نشده، MP3 یا امثال آن). از آنجایی که هر بسته، این فیلد را با خود حمل می‌کند فلذا می‌توان در حین ارسال روش کدینگ را تغییر داد.

فیلد Sequence Number (شماره ترتیب) یک شمارنده است که به ازای ارسال هر بسته RTP یک واحد افزایش می‌یابد. از این شماره برای کشف بسته‌های گمشده استفاده می‌شود. در فیلد Timestamp، «مهر زمان» درج می‌شود تا مشخص گردد بسته جاری (نسبت به زمان تولید اولین بسته) دقیقاً در چه زمانی تولید شده است.^۱ با درج مهر زمان، «لرزش» (Jitter) کاهش می‌یابد زیرا لحظه اجرای^۲ هر بسته از زمان دریافت بسته جدا می‌شود. فیلد Synchronization Source Identifier مشخص می‌کند که بسته جاری به کدام استریم تعلق دارد.^۳ به کمک این فیلد استریمهای متعدد در یک دنباله از بسته‌های UDP معمولی، مالتی‌پلکس یا دی‌مالتی‌پلکس می‌شوند.

در آخر، فیلد Contributing Source Identifier قرار می‌گیرد که در صورت استفاده، بیانگر آنست که در استودیو، میکسر وجود دارد.^۴ در چنین حالتی، میکسر موظف به سنکرونیزاسیون استریمهاست؛ در ضمن استریمهایی که باید میکس گردند در این فیلد فهرست می‌شوند.

پروتکل RTP یک خواهر کوچک به نام RTCP^۵ دارد. (شاید هم دوقلو و همزاد باشند!) این پروتکل عملیات فیدبک، سنکرون سازی و واسطه کاربری را مدیریت می‌کند ولیکن هیچ داده‌ای را انتقال نمی‌دهد. اولین کاربرد آن گرفتن فیدبک از میزان تأخیر، لرزش (Jitter)، پهنای باند، ازدحام (congestion) و کسب آگاهی از دیگر ویژگیهای شبکه برای ماشین مبداء است. از این اطلاعات می‌توان در فرآیند کدینگ داده‌ها بهره گرفت تا مثلاً وقتی که شبکه خوب عمل می‌کند، نرخ ارسال را افزایش بدهد (با کیفیت را بهبود ببخشد) و هنگامی که مشکلی در شبکه حادث می‌شود (مثل ازدحام) نرخ ارسال را پایین بیاورد. با در اختیار داشتن یک فیدبک دائم از شرایط فعلی شبکه، الگوریتم کدینگ می‌تواند همیشه بهترین روش را در تناسب با شرایط جاری، انتخاب نماید. به عنوان مثال وقتی پهنای باند در حین ارسال افزایش یا کاهش می‌یابد، الگوریتم کدینگ می‌تواند از روش MP3 به PCM (یا بالعکس) تغییر روش بدهد. وجود فیلد Payload Type این امکان را فراهم آورده که در هر لحظه از زمان بتوان الگوریتم کدینگ را به روش مناسب تغییر داد.

پروتکل RTCP، سنکرون سازی بین استریمها را نیز بر عهده دارد. مشکل از آنجا ناشی می‌شود که

۱. دقت کنید که این زمان مربوط به تولید اولین کلمه بسته یا به عبارت دقیقتر اولین نمونه (Sample) در بسته است زیرا بسته مملو از کلمات یا نمونه‌های مربوط به صدا، تصویر یا امثالهم است که از لحاظ زمانی بعد از اولین نمونه تولید شده‌اند. -م

۲. Playback time

۳. استریمهای صدا، تصویر و امثالهم می‌توانند بطور همزمان تولید یا دریافت شوند لذا باید راهی برای تفکیک و سنکرونیزاسیون بسته‌های متعلق به هر استریم وجود داشته باشد. -م

۴. بدین معنا که همانند یک استودیو، منابع تولید استریم بسیار متعدّدند و نهایتاً باید میکس شوند. -م

۵. Real-time Transport Control Protocol

استریمهای متفاوت از سیگنالهای ساعت با مشخصات متفاوتی مثل فرکانس، Drift بهره می‌گیرند. پروتکل RTCP می‌تواند برای سنکرون‌سازی این استریمها بکار گرفته شود. در آخر، RTCP روشی برای نامگذاری مبداء استریمهای مختلف (مثلاً در قالب اسامی ASCII) ارائه کرده است. این اطلاعات می‌تواند بر روی صفحه نمایش گیرنده، نشان داده شود تا مشخص شود چه کسی در حال صحبت یا ارسال استریم است. اطلاعات بیشتر در خصوص RTP را می‌توانید در مرجع (Perkins, 2002) بیابید.

۵-۶ پروتکل‌های لایه انتقال در اینترنت: TCP^۱

UDP پروتکل بسیار ساده‌ای است و کاربردهای خاص خود را (مثل تعامل سرویس دهنده/مشتري یا کاربردهای چند رسانه‌ای) دارد ولیکن اغلب کاربردهای اینترنت به تحویل مطمئن و مرتب شده داده‌ها نیازمندند. UDP چنین امکانی را فراهم نمی‌کند و طبعاً به پروتکل دیگری احتیاج است. این پروتکل TCP نام دارد و بیشتر بار اینترنت را به دوش می‌کشد. اجازه بدهید جزئیات این پروتکل را بررسی نماییم.

۱-۵-۶ مقدمه‌ای بر TCP

TCP بدین منظور طراحی شد تا یک دنباله (استریم) از بایتهای را به صورت مطمئن و عاری از خطا بین دو نقطه پایانی (یعنی دو پروسه) از شبکه‌ای که نامطمئن و مستعد خطاست، منتقل نماید. «شبکه‌ای از شبکه‌ها» (Internetwork) از لحاظ ساختاری با یک شبکه واحد تفاوت دارد زیرا هر بخش از آن دارای توپولوژی مختلف، پهنای باند، تأخیر، طول بسته و پارامترهای متفاوت است. TCP طراحی شد تا بطور پویا خود را با ویژگیهای چنین شبکه ناهمگونی تطبیق بدهد و در مقابل ناکارآمدیها و شکستهای گوناگون، تحمل‌پذیر (Robust) باشد. پروتکل TCP رسماً در RFC 793 تعریف شده است. به مرور زمان، انواع خطاها و تناقضات آن آشکار شد و در برخی از زمینه‌ها، نیازها نیز تغییر کرد. تبیین این مفاد و برخی از اصلاحیه‌ها در RFC 1122 آمده است. بسط و بهبود آن نیز در RFC 1323 تشریح شده است.

هر ماشین که از TCP پشتیبانی می‌کند باید دارای «واحد انتقال TCP»^۲ باشد، خواه در قالب پروسه‌های کتابخانه‌ای یا یک پروسه کاربری و خواه در قالب بخشی از هسته سیستم عامل. در تمامی این حالات، واحد انتقال، استریمهای TCP را مدیریت کرده و به صورت واسطی بین لایه IP و پروسه‌های کاربردی انجام وظیفه می‌کند. «واحد انتقال» دنباله‌ای از داده‌های کاربر را از پروسه‌های محلی گرفته و آن را به قطعاتی با طول کمتر از 64KB می‌شکند (البته در عمل این قطعات اغلب ۱۴۶۰ بایتی هستند تا پس از الحاق سرآیند IP و TCP به آنها، جمعاً ۱۵۰۰ بایت شده و متناسب با ظرفیت یک فریم اترنت باشند). سپس هر یک از این قطعات را [پس از افزودن سرآیند لازم] در قالب یک دیتاگرام مستقل IP ارسال می‌نماید. وقتی دیتاگرامهای حاوی داده‌های TCP به ماشین مقصد می‌رسند، به واحد انتقال تحویل داده شده و براساس آنها دنباله بایتهای اصلی ساخته خواهد شد. برای سادگی اغلب بجای «واحد انتقال TCP» (یعنی یک قطعه نرم‌افزار) و هم بجای پروتکل TCP (یعنی مجموعه‌ای از قوانین) به تنهایی از واژه TCP استفاده می‌نماییم. مضمون کلام مشخص می‌کند که منظورمان از TCP چیست. مثلاً وقتی می‌گوییم «کاربر داده‌ها را به TCP تحویل می‌دهد» روشن است که منظور از TCP همان «واحد انتقال TCP» است.

لایه IP هیچ تضمینی در تحویل صحیح و مرتب بسته‌ها نمی‌دهد لذا این وظیفه بر عهده TCP است که پس از

انقضای مهلت مقرر آنها را از نو ارسال نماید. از طرفی ممکن است دیتاگرامها به صورت نامرتب و پس و پیش به مقصد برسند لذا وظیفه دیگر TCP آن است که آنها را مرتب کرده و پیام اصلی را بازسازی نماید. کوتاه سخن آنکه TCP باید آن قابلیت اطمینانی را که کاربران می خواهند ولی IP عرضه نمی کند، در اختیار آنان قرار بدهد.

۲-۵-۶ مدل خدمات TCP

برای استفاده از خدمات TCP، پروسه های گیرنده و فرستنده یک نقطه پایانی به نام «سوکت» ایجاد می کنند. (در مورد سوکت در بخش ۶-۱-۳ بحث شد.) هر سوکت دارای یک «شماره سوکت» (یا به عبارتی آدرس سوکت) است که این شماره متشکل از آدرس IP ماشین میزبان و یک عدد ۱۶ بیتی یکتا و محلی است که اصطلاحاً «پورت» (Port) نامیده می شود. «پورت» معادل نام TSAP در ادبیات پروتکل TCP است. برای بهره گیری از خدمات TCP باید یک اتصال بین سوکت ماشین مبدا و سوکت ماشین مقصد ایجاد شود. توابع اولیه مرتبط با سوکت در شکل ۵-۶ فهرست شده اند.

می توان از یک سوکت برای برقراری چندین اتصال همزمان بهره گرفت. به عبارت دیگر ممکن است دو یا چند اتصال به یک سوکت واحد ختم شوند. هویت هر «اتصال» توسط شناسه های سوکت طرفین در قالب (*socket1* , *socket2*) مشخص می گردد. یعنی از شماره های مدار مجازی یا هر شناسه دیگر استفاده نمی شود. پورتهایی با شماره زیر ۱۰۲۴ اصطلاحاً به نام «پورتهای شناخته شده»^۱ مشهورند و برای سرویسهای استاندارد رزرو شده اند. مثلاً هر پروسه ای که می خواهد به منظور انتقال فایل، با ماشینی ایجاد اتصال کند می تواند با پورت ۲۱ از ماشین مقصد که «دیمون FTP» (FTP Daemon) بدان گوش می کند تماس بگیرد. فهرست پورتهای شناخته شده در آدرس www.iana.com در دسترس می باشد. تاکنون بیش از ۳۰۰ شماره از آنها منتسب شده اند که چند تا از مشهورترین شماره پورها را در شکل ۶-۲۷ ملاحظه می کنید.

در هنگام راه اندازی سیستم، دیمون FTP می تواند خود را به پورت ۲۱ متصل کند؛ دیمون TelNet به پورت ۲۳ و دیگر پروسه ها نیز به همین ترتیب می توانند خود را به یک پورت متصل نمایند. از آنجایی که اکثر این دیمونها در بیشتر زمانها بیکار هستند لذا اجرای آنها در هنگام راه اندازی سیستم، فضای حافظه را بیهوده تلف خواهد کرد. به جای همه آنها، عموماً یک دیمون واحد که در یونیکس به «دیمون inetd»^۲ مشهور است، اجرا شده و بطور همزمان خود را به چندین پورت متصل می نماید و منتظر تماسهای ورودی می شود. وقتی تقاضای برقراری یک اتصال دریافت شد، پروسه inetd یک پروسه جدید ایجاد کرده و دیمون مناسب را برای سرویس دهی به این تقاضا، راه اندازی و اجرا می نماید. بدین نحو، تمام دیمونهای دیگر به غیر از دیمون inetd فقط و فقط زمانی فعال می شوند که کاری برای انجام داشته باشند. دیمون inetd شماره پورتهایی که باید به آنها گوش بدهد را از فایل پیکربندی خاص خود استخراج می نماید. مسئول سیستم (admin) می تواند سیستم را به گونه ای پیکربندی کند که برخی از دایمونها به طور دائم به یک پورت گوش بدهند (مثل پورت ۸۰) و به باقی پورها فقط دایمون inetd گوش بدهد.

تمام اتصالات TCP، «دوطرفه کامل»^۳ و «نقطه به نقطه» هستند. «دوطرفه کامل» یعنی ترافیک می تواند بطور همزمان در دو جهت جریان داشته باشد؛ نقطه به نقطه نیز یعنی یک اتصال صرفاً دو نقطه پایانی (End Point) دارد.^۴ TCP از ارسال چند پخش (Multicasting) یا پخش فراگیر (Broadcast) پشتیبانی نمی کند. یک اتصال TCP استریمی از بایتها را منتقل می نماید نه دنباله ای از پیامها را؛ یعنی مرز پیامهایی که بین دو نقطه

۲. Internet Daemon

۱. Well-known Ports

۳. Full Duplex و Point-to-Point

۴. یا عبارتی صرفاً دو پروسه مبدا و مقصد با یکدیگر محاوره می کنند. -م

پورت	پروتکل	کاربرد
21	FTP	انتقال فایل
23	Telnet	ورود به سیستم از راه دور
25	SMTP	پست الکترونیکی (ایمیل)
69	TFTP	پروتکل ساده انتقال فایل
79	Finger	جستجوی اطلاعات در خصوص یک کاربر
80	HTTP	تور جهان‌گستر یا همان سیستم جهانی وب
110	POP-3	دسترسی به نامه‌های الکترونیکی از راه دور
119	NNTP	اخبار یوزنت (سیستم خبررسان)

شکل ۶-۲۷. برخی از شماره‌های پورت انتساب داده شده مشهور.

مبادله می‌شوند حفظ نخواهد شد. به عنوان مثال اگر پروسه فرستنده، چهار بسته ۵۱۲ بایتی را بر روی یک استریم بفرستد ممکن است در قالب چهار قطعه داده ۵۱۲ بایتی یا دو قطعه ۱۰۲۴ بایتی یا یک قطعه ۲۰۴۸ یا بترتیبی مشابه، به پروسه گیرنده تحویل شود. (شکل ۶-۲۸ را ببینید.) هیچ راهی برای آن که گیرنده بتواند اندازه قطعات داده را تشخیص بدهد، وجود ندارد.



شکل ۶-۲۸. (الف) چهار قطعه ۵۱۲ بایتی در قالب دیتاگرامهای مستقل IP ارسال شده‌اند. (ب) توده ۲۰۴۸ بایتی داده‌ها، بطور یکجا و با یکبار فراخوانی READ، تحویل برنامه کاربردی شده است.

در یونیکس، فایلها نیز همین ویژگی را دارند یعنی پروسه‌ای که از یک فایل می‌خواند نمی‌تواند بفهمد که آیا فایل به صورت بلوکی نوشته شده، بایت به بایت نوشته شده یا به صورت یکجا و در آن واحد نوشته شده است. همانند یک فایل در یونیکس، نرم‌افزار TCP نیز هیچ تصویری از معنای بایتها ندارد. یک بایت فقط یک بایت است! وقتی یک برنامه کاربردی داده‌های خود را به TCP تحویل می‌دهد، ممکن است TCP آنها را فوراً ارسال نماید یا آنها را موقتاً بافر کند. با این حال برخی از برنامه‌های کاربردی می‌خواهند که داده‌هایشان فوراً ارسال شود. مثلاً فرض کنید کاربری از راه دور به ماشینی وارد شده باشد (عمل login). پس از آنکه خط فرمان تکمیل و کلید Enter زده شد، انتظار می‌رود که این خط فوراً به ماشین راه دور تحویل شود و بهیچوجه تا رسیدن خط فرمان بعدی بافر نگردد. برای آن که TCP مجبور به ارسال سریع و آنی داده‌ها شود، برنامه کاربردی می‌تواند از بیت پرچم PUSH (PUSH Flag) بهره بگیرد تا به TCP تفهیم شود که انتقال داده‌ها نباید به تأخیر بیفتد. (در خصوص این بیت بعداً صحبت خواهیم کرد.)

برخی از برنامه‌های کاربردی ابتدایی، از بیت پرچم PUSH به عنوان نوعی نشانه برای تعیین مرز هر پیام بهره می‌گرفتند.^۱ اگرچه این حقه گاهی اوقات کار می‌کند ولی گاهی نیز با شکست مواجه می‌شود زیرا در تمام

۱. به عبارت دیگر وقتی یک بسته TCP که در آن بیت پرچم PUSH فعال شده، می‌رسد اگرچه بدین معناست که نباید بافر شود و سریعاً باید به برنامه کاربردی تحویل گردد ولی برخی برنامه‌های کاربردی ابتدایی از این بیت به عنوان حقه‌ای جهت تفهیم پایان پیام جاری به طرف مقابل، بهره می‌گرفتند. -م

پیاده‌سازیهای TCP، در طرف گیرنده الزاماً بیت پرچم PUSH به برنامه کاربردی تحویل داده نمی‌شود. مضاف بر این، اگر قبل از آنکه اولین بسته با پرچم PUSH مجال ارسال پیدا کند (مثلاً به دلیل شلوغی خط خروجی)، چند بسته دیگر از همین نوع تولید و تحویل TCP شود، TCP مجاز خواهد بود که تمام بسته‌های با علامت PUSH را پشت سرهم ادغام کرده و آن را در قالب یک دیتاگرام IP به صورت یکجا ارسال نماید که در این صورت مرز قطعات مختلف قابل تشخیص نخواهد بود.

یکی دیگر از خدمات TCP که اشاره به آن خالی از لطف نخواهد بود، موضوع «داده‌های اضطراری» (Urgent Data) است. وقتی کاربری که از راه دور با یک برنامه‌ای کاربردی در تعامل است کلید DEL یا CTRL-C را فشار می‌دهد (تا روند اجرای برنامه‌ای که از قبل شروع شده را قطع کند)، برنامه کاربردی فرستنده، پاره‌ای اطلاعات کتلی در «استریم داده» قرار داده و بیت پرچم URGENT را در آن فعال و آنرا جهت ارسال به TCP تسلیم می‌کند. این رخداد باعث می‌شود که TCP به جمع‌آوری داده خاتمه بدهد و هر آنچه را که دارد سریعاً برای طرف مقابل بفرستد.

وقتی «داده‌های اضطراری» به مقصد می‌رسند به برنامه کاربردی گیرنده، «وقفه» داده می‌شود (در اصطلاح یونیکس به آن برنامه سیگنال داده می‌شود)؛ آن برنامه طبقاً کارش را متوقف کرده و استریم داده را می‌خواند تا داده‌های اضطراری را یافته و سریعاً پردازش نماید. پایان داده‌های اضطراری علامت‌گذاری شده است لذا برنامه کاربردی به راحتی می‌تواند انتهای این داده‌ها را تشخیص بدهد. البته ابتدا داده‌های اضطراری علامت‌گذاری نمی‌شود و تشخیص آن بر عهده برنامه کاربردی است. این روش فقط مکانیزم سیگنال‌دهی به پروسه‌ها را ارائه می‌کند و مدیریت بقیه امور بر عهده برنامه کاربردی گذاشته شده است.

۳-۵-۶ پروتکل TCP

در این بخش یک دید کلی از پروتکل TCP ارائه خواهیم کرد و در بخش بعدی سرآیند پروتکل را فیلد به فیلد بررسی می‌نماییم.

ویژگی کلیدی در پروتکل TCP (که طراحی کل پروتکل از آن تاثیر پذیرفته) آنست که هر بایت ارسالی بر روی یک اتصال TCP دارای یک شماره ترتیب ۳۲ بیتی است. زمانی که اینترنت شروع به کار کرد، خطوط ارتباطی بین مسیربایها، اغلب خطوط اجاره‌ای 56kbps بودند و حتی اگر یک ماشین با تمام سرعت، اقدام به ارسال داده می‌کرد بیش از یک هفته طول می‌کشید تا این شماره ۳۲ بیتی به آخر رسیده و به صفر برگردد. در سرعت‌های جدید شبکه، شماره‌های ترتیب می‌توانند در مدت کوتاه و خطرناکی به آخر رسیده و تکرار شوند. (بعدها به این مسئله بیشتر می‌پردازیم). برای اعلام وصول داده‌ها (یعنی Acknowledgement) و «مکانیزم پنجره» نیز از فیلدها و شماره‌های ترتیب متفاوتی استفاده شده است.

واحدهای انتقال TCP در سمت فرستنده و گیرنده، داده‌ها را در قالب یکسری «قطعه» (Segment) رد و بدل می‌کنند. هر قطعه TCP متشکل از ۲۰ بایت سرآیند ثابت و اجباری (و در صورت نیاز یک بخش اختیاری در سرآیند) است و به دنبال آن به تعداد صفر یا چند بایت داده قرار می‌گیرد. نرم‌افزار TCP راساً در مورد اندازه هر قطعه تصمیم می‌گیرد.^۱ TCP ممکن است داده‌هایی را که در چند مرحله جهت ارسال در استریم نوشته شده‌اند، جمع‌آوری کرده و آنها را در قالب یک قطعه TCP بفرستد و یا بالعکس داده‌هایی را که در یک مرحله تحویل می‌شوند، در چند قطعه ارسال نماید. دو عامل می‌تواند محدودکننده اندازه قطعات باشد. اول آن که هر قطعه

۱. یعنی برنامه کاربردی نمی‌تواند در خصوص آنکه بسته‌ها (قطعه‌ها) در چه اندازه‌ای ارسال شود، اعمال نظر کند. -م

(شامل سرآیند آن) باید بتواند در فضای ۶۵۵۱۵ بایتی فیلد داده از بسته IP جا بگیرد؛ دوم آن که در هر شبکه پارامتری به نام MTU (Maximum Transfer Unit) (یعنی حداکثر طول بسته قابل انتقال) وجود دارد و اندازه هر بسته باید متناسب با این مقدار باشد. در عمل، MTU عموماً ۱۵۰۰ بایت است (اندازه فیلد حمل داده اترنت) و طبعاً حد بالایی طول هر قطعه باید متناسب با این مقدار باشد.

TCP از پروتکل «پنجره لغزان» بهره گرفته است: به محض آن که فرستنده، قطعه‌ای را ارسال می‌کند یک تایمر برای آن روشن می‌نماید. وقتی این قطعه به مقصد رسید، واحد TCP (TCP Entity)، قطعه‌ای را بر می‌گرداند که در آن «شماره تصدیق بسته دریافتی» (Ack.No.) درج شده است. این قطعه می‌تواند حاوی داده‌های متقابل گیرنده باشد و در صورت عدم وجود هر گونه داده، بدون داده ارسال شود. شماره تصدیق بسته دریافتی در حقیقت حاوی شماره ترتیب بایتی است که گیرنده از آن شماره به بعد منتظر دریافت آنهاست. اگر مهلت تایمر به پایان برسد و در این زمان دریافت قطعه ارسالی، تصدیق نشود، فرستنده، قطعه قبلی را از نو ارسال می‌کند.

اگرچه این پروتکل ساده به نظر می‌رسد ولی پیچیدگیها و ظرافتهایی چند، در آن نهفته است که در ادامه به آن خواهیم پرداخت. قطعات ارسالی می‌توانند به صورت نامرتب به گیرنده برسند؛ مثلاً اگر بایتهای ۳۰۷۲ تا ۴۰۹۵ در قالب یک قطعه برسد نمی‌توان آن را اعلام وصول کرد چراکه فرضاً بایتهای ۲۰۴۸ تا ۳۰۷۱ نرسیده‌اند. همچنین ممکن است قطعات ارسالی آنقدر در شبکه معطل شوند که مهلت فرستنده منقضی شده و آنها را از نو ارسال نماید و منجر به تولید قطعات تکراری شود. همچنین ارسال مجدد داده‌ها ممکن است در قالب قطعات جدیدی صورت بگیرد.^۱ فلذا این موضوع نیز باید به دقت مدیریت و نظارت شود تا گیرنده بتواند بر دنباله بایتهایی که دریافت کرده و آنهایی که دریافت نکرده به دقت کنترل داشته باشد. خوشبختانه چون هر بایت در یک دنباله (استریم) دارای یک آفست یکتاست فلذا انجام چنین کار مهمی امکان‌پذیر است.

TCP باید بتواند تمام این مسائل را به نحو کارآمد و مؤثری حل و فصل نماید. در ضمن تلاشهای وافری برای بهینه‌سازی کارایی استریمهای TCP صورت گرفته است تا در مواجهه با مشکلات شبکه خود را تطبیق بدهد. تعدادی از این الگوریتمها که در اغلب پیاده‌سازیهایی عملی TCP بکار رفته را در ادامه معرفی کرده‌ایم.

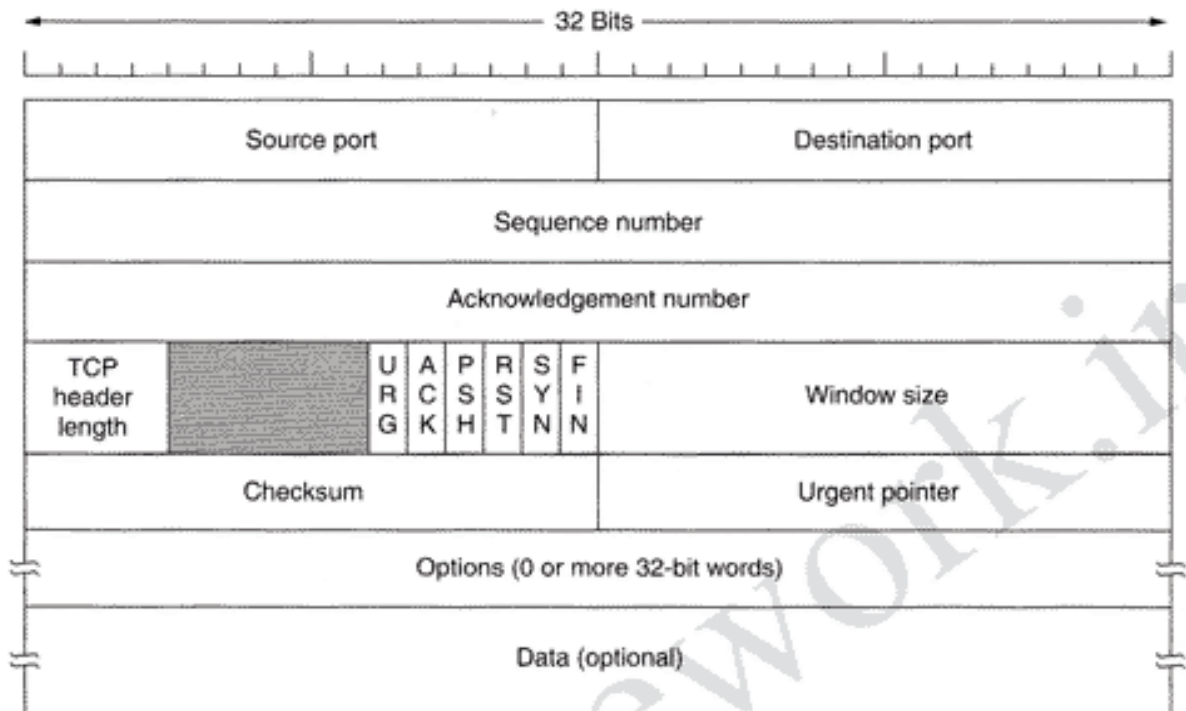
۴-۵-۶ سرآیند قطعه TCP

شکل ۶-۲۹، ساختار یک قطعه TCP (TCP Segment) را نشان می‌دهد. هر قطعه با یک سرآیند ۲۰ بایتی و با قالب ثابت شروع می‌شود. پس از این سرآیند ثابت، ممکن است یک سرآیند اختیاری قرار بگیرد. در صورت عدم وجود سرآیند اختیاری، در ادامه می‌تواند حداکثر ۶۵۴۹۵ (۶۵۵۳۵-۲۰-۲۰) بایت داده قرار بگیرد که کسریست بایت اول مربوط به سرآیند IP و بیست بایت دوم مربوط به سرآیند TCP است. قطعات بدون داده [یعنی فقط سرآیند]، معتبر و قانونی هستند و عموماً برای اعلام وصول داده‌ها (Ack) و پیامهای کنترلی کاربرد دارند. حال اجازه بدهید که سرآیند TCP را فیلد به فیلد کالبدشکافی کنیم.

فیلدهای «پورت مبدا» (Source Port) و «پورت مقصد» (Destination Port) نقاط انتهایی دو طرف یک اتصال را مشخص می‌نمایند.^۲ شماره پورتهای شناخته‌شده و مشهور در آدرس www.iana.org فهرست شده‌اند ولی هر ماشین میزبان می‌تواند به دلخواه خود، آنها را به پروسه‌ها اختصاص بدهد. یک شماره پورت ۱۶ بیتی به همراه آدرس IP ماشین میزبان، آدرس ۴۸ بیتی یک نقطه پایانی را تشکیل می‌دهد. آدرس نقاط پایانی مبدا و

۱. یعنی مثلاً یک قطعه حاوی ۱۵۰۰ بایت ارسال شده ولی چون وصول آن تصدیق نشده ممکن است ارسال مجدد آن در قالب دو قطعه ۱۰۰۰ و ۵۰۰ بایتی انجام شود. — م

۲. یعنی این دو فیلد هویت پروسه‌های گیرنده و فرستنده را تعیین می‌نمایند. — م



شکل ۶-۲۹. سرآیند TCP.

مقصد، هویت یک اتصال را تبیین می‌کنند.

فیلدهای «شماره ترتیب» (Sequence Number) و «شماره تصدیق» (Acknowledgement Number) عملکرد طبیعی خود را دارند یعنی اولی شماره ترتیب قطعه داده و دومی اعلام وصول داده‌ها است. دقت کنید که فیلد «شماره تصدیق»، شماره بایتی را مشخص می‌کند که از آن بایت به بعد منتظر دریافت داده‌هاست نه آخرین بایت دریافتی.^۱ هر دوی این فیلدها ۳۲ بیتی هستند زیرا در استریم TCP هر بایت دارای شماره ترتیب است. فیلد TCP Header Length مشخص می‌کند که سرآیند قطعه TCP چند کلمه ۳۲ بیتی است. از آنجایی که فیلد Options اندازه متغیری دارد فلذا به وجود این فیلد که طول سرآیند را مشخص می‌کند، نیاز خواهد بود. از دیدگاه فنی این فیلد «نقطه شروع داده‌ها در هر قطعه» را بر مبنای کلمات ۳۲ بیتی مشخص می‌کند ولی از دیدگاه دیگر می‌توان مقدار این فیلد را «طول سرآیند قطعه TCP» بر مبنای کلمه فرض کرد ولیکن تأثیر هر دوی این تعبیر یکی است.

در ادامه یک فیلد شش بیتی بلااستفاده آمده است. این حقیقت که از چنین فیلدی برای حدود ربع قرن استفاده نشده، مؤید آن است که TCP با تفکر و بینش بسیار جامعی طراحی شده است. کمتر پروتکلی برای اصلاح اشکالات طرح اصلی TCP، از این فیلد استفاده کرده است.

در ادامه شش پرچم تک‌بیتی (Flag) آمده است: اگر درون فیلد Urgent Pointer مقدار شعبری قرار گرفته باشد، بیت پرچم URG به ۱ تنظیم می‌شود. مقدار فیلد Urgent Pointer مشخص می‌کند که «داده‌های اضطراری» (نسبت به اولین بایت داده‌های قطعه جاری) از چه موقعیتی شروع می‌شوند. در حقیقت این فیلد نقش «پیام وقفه» (Interrupt Message) را ایفاء می‌کند. همانگونه که قبلاً اشاره کردیم این فیلد روش سراسری برای سگینال

۱. به عبارتی اگر در فیلد شماره تصدیق مثلاً عدد ۱۲۳۴۵ درج شده باشد بدین معنا تعبیر می‌شود که تا بایت ۱۲۳۴۴ دریافت شده و باید از بایت شماره ۱۲۳۴۵ به بعد ارسال شود. -م

دادن فرستنده به گیرنده (از راه دور) است.

برای آن که مشخص شود فیلد «شماره تصدیق» (Acknowledgement Number) دارای مقدار معتبر است، بیت پرچم ACK به ۱ مقداردهی می شود. اگر بیت ACK معادل صفر باشد، قطعه TCP جاری دارای هیچ «شماره تصدیق» معتبری نیست و مقدار آن نادیده گرفته می شود.

بیت PSH مشخص کننده آن است که داده های قطعه جاری باید سریعاً تحویل داده شود (PUSH شود). با تنظیم این بیت در سرآیند قطعه TCP، از گیرنده خواش می شود که داده های موجود در قطعه را بلافاصله تحویل برنامه کاربردی بدهد و تا پر شدن بافر (که معمولاً برای افزایش کارایی و سرعت کاربرد دارد) منتظر نماند. از بیت RST زمانی استفاده می شود که TCP به هر دلیلی (مثل از کار افتادن ماشین میزبان) بخواهد یک اتصال را بطور ناگهانی و یکطرفه قطع کند. همچنین زمانی که TCP بخواهد یک قطعه نامعتبر یا تلاش برای برقراری یک اتصال را نپذیرد از این بیت استفاده می کند. کلاً هر گاه قطعه ای دریافت شود که در آن بیت RST به ۱ تنظیم شده، نشان دهنده وجود یک مشکل است.

از بیت SYN برای برقراری یک اتصال استفاده می شود. در حقیقت تقاضای برقراری اتصال با ارسال قطعه ای با مشخصات $SYN=1$ و $ACK=0$ انجام می گیرد که در آن $SYN=1$ علامت تقاضای برقراری ارتباط و $ACK=0$ به معنای عدم وجود مقدار معتبر در فیلد Acknowledgement Number است. پاسخ به این تقاضا به شکل $SYN=1$ و $ACK=1$ است و در فیلد Ack.No مقدار معتبری وجود دارد. [بعداً بدین مورد خواهیم پرداخت]. در اصل، بیت SYN برای اعلام هر یک از پیامهای CONNECTION REQUEST و CONNECTION ACCEPTED بکار می رود و تمایز بین این دو پیام با بیت ACK مشخص می شود.

بیت FIN برای خاتمه دادن به یک اتصال بکار می رود. ۱ بودن این بیت بدان معناست که فرستنده، داده دیگری برای ارسال ندارد. ولیکن پروسه ای که با ارسال چنین قطعه ای اتصال را از سمت خود می بندد می تواند تا هر زمانی به دریافت داده های طرف مقابل ادامه بدهد. قطعات حاوی بیت $SYN=1$ یا $FIN=1$ دارای شماره ترتیب هستند، فلذا تضمین می شود که این بسته ها به درستی پردازش خواهند شد.^۱

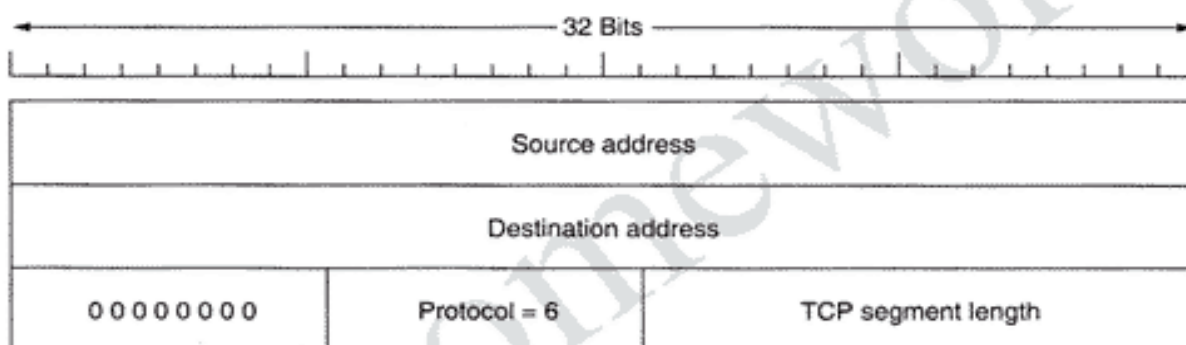
کنترل جریان در TCP به کمک یک پنجره لغزان با اندازه متغیر انجام می شود. مقدار درج شده در فیلد Window Size در هر بسته TCP، به طرف مقابل تفهیم می کند که از بایتی که شماره آن در فیلد Acknowledgement Number درج شده، به اندازه چند بایت حق ارسال داده دارد. درج عدد صفر در فیلد Window Size کاملاً قانونی و معتبر است و در حقیقت بیان می کند که اگرچه تا بایت شماره Acknowledgement Number-1، دریافت شده ولیکن به دلیل کمبود فضای بافر، تا اطلاع ثانوی نمی تواند پذیرای داده بیشتری باشد. گیرنده بعداً می تواند به فرستنده مجوز ارسال بدهد: این کار با فرستادن یک بسته که در آن فیلد Acknowledgement Number همان مقدار قبلی را دارد و فیلد Window Size آن غیرصفر است، صورت می گیرد.

در پروتکل های فصل سوم، تصدیق وصول فریمها و مجوز ارسال فریمهای جدید یکی هستند. [یعنی دریافت یک پیغام Ack، ضمن تصدیق وصول بسته به فرستنده اجازه می دهد که بسته جدیدی بفرستد]. این مسئله از آنجا ناشی می شود که طول پنجره هر یک از پروتکل های یاد شده ثابت و بدون تغییر است. در پروتکل TCP، اعلام وصول داده ها (یعنی Ack) و مجوز ارسال داده بیشتر از هم تفکیک شده اند. در نتیجه یک گیرنده می تواند بگوید:

۱. به عبارت دیگر اگر به هر دلیلی یک بسته $FIN=1$ برای پروسه ای ارسال شود، پردازش نخواهد شد مگر آن که شماره ترتیب آن صحیح و دقیق باشد.

«من تا بایت شماره k را به درستی دریافت کرده‌ام ولیکن فعلاً تمایلی به دریافت داده دیگر ندارم» تفکیک این دو مفهوم و داشتن پنجره‌ای با طول متغیر، قابلیت انعطاف بیشتری به پروتکل اعطا می‌کند. در ادامه این موضوع را به تفصیل بررسی خواهیم کرد.

فیلد Checksum یک کد کشف خطا است و جهت اطمینان از صحت داده‌ها کاربرد دارد. این کد حاصل جمع سرآیند، داده‌ها و یک «شبه‌سرآیند فرضی» (Pseudoheader) است. قالب شبه‌سرآیند فرضی در شکل ۶-۳۰ مشخص شده است. برای محاسبه این کد ابتدا فیلد Checksum صفر فرض می‌شود و در صورت فرد بودن تعداد بایتها، تعدادی صفر زائد به انتهای داده‌ها اضافه می‌گردد تا تعداد بایتها زوج شود. الگوریتم محاسبه Checksum بسیار ساده است: مجموعه بایتها به صورت کلمات ۱۶ بیتی (یعنی دو بایت دو بایت) با هم جمع شده و حاصل جمع به صورت «متمم ۱» (One's Complement) منفی می‌شود و درون فیلد Checksum قرار می‌گیرد. نتیجتاً وقتی درگیرنده این محاسبه بر روی کل قطعه (شامل فیلد Checksum) انجام می‌شود نتیجه آن باید صفر باشد. در غیر این صورت داده‌ها قابل اعتماد و سالم نیستند.



شکل ۶-۳۰. شبه‌سرآیندی که در محاسبه کد کشف خطای TCP Checksum دخالت داده می‌شود.

شبه‌سرآیند (Pseudoheader) از فیلدهای زیر تشکیل شده‌اند: آدرسهای IP سی و دو بیتی مبدا و مقصد، شماره پروتکل TCP (یعنی عدد ۶) و تعداد کل بایتهای قطعه TCP (شامل سرآیند آن). وارد کردن این شبه‌سرآیند در محاسبه مقدار Checksum اگرچه به کشف بسته‌هایی که به اشتباه دریافت شده‌اند کمک خواهد کرد ولیکن اصول و قواعد تفکیک سلسله‌مراتب پروتکلها و مخفی ماندن جزئیات هر لایه را نقض می‌کند چراکه آدرسهای IP متعلق به لایه IP است و ربطی به لایه TCP ندارد. UDP نیز برای محاسبه کد Checksum از همین شبه‌سرآیند استفاده می‌کند.

فیلد Options برای درج گزینه‌های اختیاری در قطعه TCP تعریف شده است و می‌توان از آن برای اضافه کردن فیلدهایی که در پروتکل TCP پیش‌بینی نشده، استفاده کرد. مهمترین گزینه اختیاری، گزینه‌ای است که به ماشین میزبان اجازه می‌دهد طول حداکثر قطعات TCP که تمایل به پذیرش آنها را دارد، به اطلاع طرف مقابل برساند. استفاده از قطعات بزرگ، کارآمدتر از ارسال قطعات کوچک است زیرا سربار ناشی از سرآیند ۲۰ بایتی، بر روی داده بیشتری سرشکن می‌شود ولی ممکن است ماشینهای میزبان کوچک از عهده پذیرش قطعات بزرگ برنیایند. در خلال برقراری یک اتصال، هر یک از طرفین طول حداکثر قطعات مورد پذیرش خود را به اطلاع طرف مقابل خود می‌رساند. اگر ماشینی از این گزینه استفاده نکند، اندازه پیش فرض داده‌ها در هر قطعه ۵۳۶ بایت خواهد بود. تمام ماشینهای میزبان در اینترنت موظف به پذیرش قطعات TCP با اندازه $556 = 536 + 20$ بایت هستند. الزامی به یکسان بودن اندازه قطعات ارسالی در دو جهت وجود ندارد.

برای خطوط با پهنای باند بالا یا تأخیر زیاد (یا هر دو)، پنجره ۶۴ کیلوبایتی می تواند مشکل را باشد.^۱ در یک خط T3 (با سرعت 44.736Mbps)، تخلیه و ارسال یک بافر ۶۴ کیلوبایتی فقط ۱۲ میلی ثانیه طول می کشد. یا مثلاً اگر تأخیر انتشار رفت و برگشت [یعنی رفت بسته و برگشت Ack آن] ۵۰ میلی ثانیه طول بکشد (که برای خطوط فیبرنوری بین قاره ای کاملاً طبیعی است)، فرستنده سه چهارم از زمان مفید خود را در انتظار بازگشت پیام اعلام وصول (Ack) تلف می کند. برای ارتباطات ماهواره ای وضع از این هم بدتر است.^۲ استفاده از پنجره ای با طول بزرگتر از ۶۴ کیلوبایت می تواند از توقف فرستنده به دلیل پر شدن فضای پنجره جلوگیری کند ولیکن با فیلد ۱۶ بیتی Window Size نمی توان اندازه چنین بافری را اعلام کرد. در RFC 1323 یک گزینه اختیاری به نام Window Scale پیشنهاد شده که به کمک آن فرستنده و گیرنده می توانند «ضریب مقیاس پنجره» را به اطلاع یکدیگر برسانند. این گزینه اختیاری در فیلد Options قرار خواهد گرفت و اجازه می دهد که هر یک از طرفین بتوانند فیلد Window Size خود را تا ۱۴ بیت به سمت چپ شیفت بدهند. بدین ترتیب اندازه پنجره می تواند تا ۲^{۳۰} بایت (یک گیگابایت) افزایش یابد. امروزه در اغلب پیاده سازیهای TCP، از این گزینه حمایت می شود. گزینه دیگری که در RFC 1106 پیشنهاد شده و در اغلب پیاده سازیهای TCP از آن پشتیبانی می شود آن است که به جای استفاده از پروتکل Go Back n (عقبگرد به اندازه n) از پروتکل «تکرار انتخابی» (Selective Repeat) بهره گرفته شود. اگر گیرنده، ابتدا قطعه ای خراب و به دنبال آن چندین قطعه بزرگ و سالم دریافت نماید، در صورتی که از پروتکل معمولی TCP استفاده شود، فرستنده پس از انقضای مهلت مقرر تمام قطعات اعلام وصول نشده را از نو ارسال خواهد کرد (حتی بسته هایی را که سالم رسیده اند) زیرا از پروتکل Go Back n بهره گرفته شده است. در RFC 1106 مفهوم جدیدی به نام NAK (پیغام عدم وصول) معرفی شده که به گیرنده اجازه می دهد قطعه خاصی را درخواست بدهد. پس از دریافت قطعه ناقص، وصول تمام داده هایی که از قبل بافر شده اند به یکباره تصدیق می شود و بدین ترتیب حجم داده هایی که بیهوده ارسال مجدد می شوند، کاهش خواهد یافت.

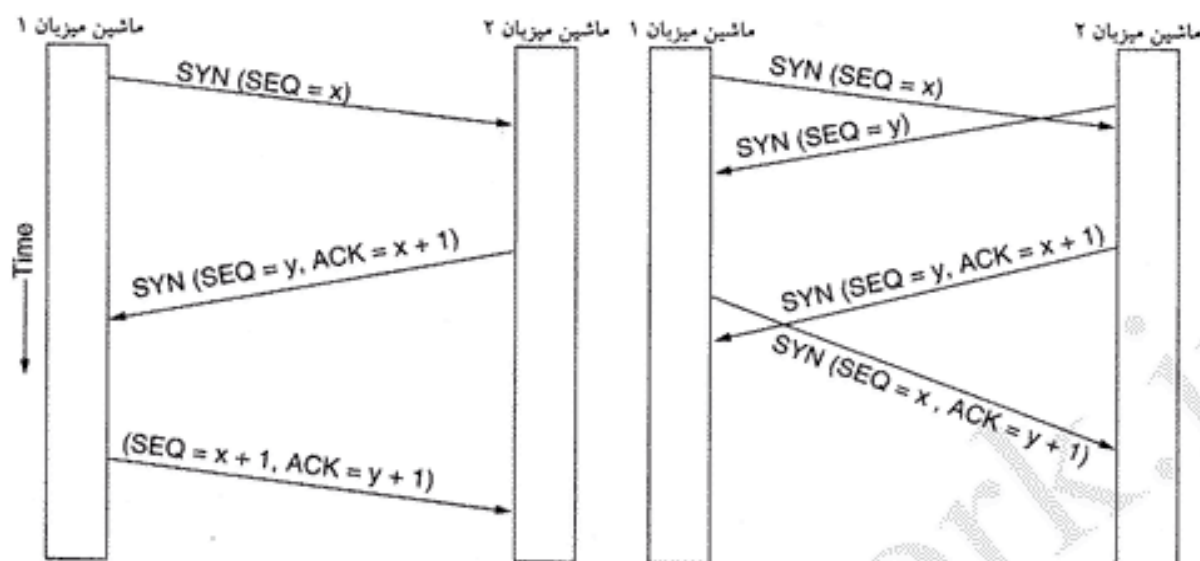
۵-۵-۶ برقراری اتصال TCP

در پروتکل TCP برای برقراری اتصال از روش «دست تکانی سه مرحله ای» (Three-Way Handshake) (که در بخش ۲-۲-۶ تشریح شد) بهره گرفته می شود. برای آن که اتصالی ایجاد شود باید یکی از طرفین (که آن را سرویس دهنده می نامیم) از طریق اجرای توابع اولیه LISTEN و ACCEPT منتظر تقاضاهای ورودی باقی بماند. (البته می تواند مبداء مورد نظر خود را تعیین کند یا آن که مبداء خاصی را مد نظر نداشته باشد). سمت مقابل (یعنی مشتری)، تابع اولیه CONNECT را اجرا کرده و آدرس IP و پورت کسی را که می خواهد با آن اتصال برقرار کند و همچنین طول حداکثر قطعات TCP مورد پذیرش (و در صورت نیاز برخی از اطلاعات کاربری نظیر کلمه عبور) را مشخص می نماید. تابع اولیه CONNECT، یک قطعه TCP با مشخصات SYN=1 و ACK=0 برای طرف مقابل فرستاده و منتظر برگشت پاسخ باقی می ماند.

وقتی چنین بسته ای به مقصد می رسد، واحد TCP در آنجا ابتدا بررسی می کند که آیا پروسه ای توسط تابع LISTEN در حال گوش دادن به شماره پورت مشخص شده در فیلد شماره پورت مقصد (Destination Port) هست یا خیر؟ اگر چنین نبود با ارسال یک قطعه TCP حاوی بیت RST=1، تقاضای برقراری اتصال را رد می کند. اگر پروسه ای در حال گوش دادن به پورت مربوطه باشد، قطعه TCP ورودی، بدان پروسه تحویل داده خواهد

۱. از آنجایی که فیلد Window Size شانزده بیتی است لذا طول پنجره به 64KB (۶۵۵۳۵ بایت) محدود شده است. - م

۲. تأخیر رفت بسته و برگشت Ack آن، در کانالهای ماهواره ای ژئوسنکرون حدود ۸۰۰ میلی ثانیه است. - م



شکل ۶-۳۱. (الف) ایجاد یک اتصال TCP در شرایط معمولی (ب) تلاقی دو تماس.

شد. پروسه مربوطه می‌تواند آن اتصال را بپذیرد یا رد کند. اگر پذیرفته شود یک قطعه تصدیق (Acknowledgement) برگردانده خواهد شد. توالی ارسال قطعات TCP جهت برقراری اتصال، در شکل ۶-۳۱-الف (در حالت طبیعی) نشان داده شده است.

در حالتی که تصادفاً دو ماشین بطور همزمان بخواهند اتصالی بین دو سوکت یکدیگر برقرار نمایند، رخدادهایی با ترتیب شکل ۶-۳۱-ب اتفاق می‌افتد. نتیجه نهایی این رخدادها آن است که فقط و فقط یک اتصال برقرار می‌شود نه دو تا، زیرا هویت هر اتصال برحسب نقاط پایانی آن [یعنی آدرسهای IP و پورت دو پروسه پایانی] شناسایی می‌شود؛ اگر اولین اتصال برقرار شده با شناسه (x,y) مشخص شود، اتصال دوم نیز دارای همین شناسه خواهد بود و در جدول اتصالات فعال فقط یک درایه (Entry) با شناسه (x,y) درج خواهد شد.

شماره ترتیب اولیه پیشنهادی در هر اتصال به دلیلی که قبلاً بدان اشاره کردیم از صفر شروع نخواهد شد. برای انتخاب این شماره از ساعتی استفاده می‌شود که هر ۴ میکروثانیه تیک می‌زند. برای اطمینان بیشتر هرگاه ماشینی از کار بیفتد می‌تواند به مدت حداکثر طول عمر بسته‌هایی که از اتصال قبلی در جایی از شبکه اینترنت سرگردان مانده‌اند، صبر کند و سپس از نو راه‌اندازی شود.

۶-۵-۶ خاتمه دادن به اتصال TCP

هر چند اتصالات TCP، دو طرفه کامل (Full Duplex) هستند ولی برای فهم چگونگی قطع اتصال، بهتر است یک اتصال TCP را به صورت یک جفت اتصال یکطرفه (Simplex) در نظر بگیریم. هر اتصال یکطرفه، مستقل از جفت خود قطع می‌شود. برای خاتمه دادن به یک اتصال، هر یک از طرفین می‌توانند یک قطعه TCP که در آن بیت FIN به تنظیم شده، ارسال نمایند. چنین قطعه‌ای بدین معناست که داده دیگری جهت ارسال وجود ندارد. وقتی دریافت این قطعه تصدیق شد اتصال در یکی از جهات، خاتمه می‌یابد ولیکن جریان داده‌ها در جهت مخالف می‌تواند بطور نامحدود ادامه داشته باشد. وقتی که اتصال در هر دو جهت قطع شد، اتصال TCP خاتمه می‌یابد. طبیعتاً برای خاتمه دادن به یک اتصال، به ارسال چهار قطعه TCP نیاز است: یکی برای FIN و یکی برای ACK، در هر یک از جهات. با این حال ممکن است اولین ACK برگشتی با FIN طرف مقابل در یک بسته ادغام شود و تعداد این بسته‌ها به سه تا کاهش یابد.

دقیقاً مشابه با تماسهای تلفنی که در آن هر دو نفر می توانند با خداحافظی، گوشی تلفن خود را بگذارند، دو پروسه انتهایی یک اتصال نیز ممکن است بطور همزمان قطعه ای حامل $FIN=1$ بفرستند. هر یک از این تقاضاهای قطع اتصال به روش معمولی تصدیق شده و به اتصال خاتمه داده می شود.

برای آن که مشکل «دو سپاه» (بخش ۳-۲۶) پیش نیاید، از تایمر نیز استفاده شده است. اگر پاسخ به FIN به مدت دو برابر طول عمر حداکثر هر بسته دریافت نشود، فرستنده FIN ، بطور یک جانبه به اتصال خاتمه خواهد داد. طرف مقابل نیز عاقبت متوجه خواهد شد که هیچکسی در طرف مقابل به او گوش نمی دهد و مهلت او نیز منقضی می شود و به اتصال خاتمه خواهد داد. اگرچه این راهکار چندان کامل و مطلوب نیست ولیکن از آنجایی که از لحاظ تئوری هیچ راهکار مطمئن و کاملی وجود ندارد مجبور به برگزیدن این راهکار هستیم. البته این مشکلات در عمل به ندرت بروز می کند.

۷-۵-۶ مدل سازی فرآیند مدیریت اتصال در TCP

مراحل لازم برای برقراری و ختم اتصال در TCP را می توان در قالب یک «ماشین حالت محدود» (Finite State Machine) با یازده وضعیت مختلف که در جدول ۳۲-۶ فهرست شده، نشان داد. در هر «وضعیت» وقوع برخی از «رخدادها» معتبر هستند. وقتی یک رخداد معتبر حادث می شود سلسله ای از «کنشها» (Actions) انجام خواهد شد. برای برخی دیگر از رخدادها نیز، فقط به گزارش خطا پسند می شود.

هر اتصال از وضعیت CLOSED آغاز می شود. اگر یکی از رخدادهای «بازکردن غیرفعال» (با اجرای تابع اولیه LISTEN) یا «بازکردن فعال»^۱ (با اجرای تابع اولیه CONNECT) رخ بدهد، اتصال از این «وضعیت» خارج خواهد شد. اگر طرف مقابل اتصال دقیقاً در وضعیت معکوس قرار داشته باشد یک اتصال برقرار شده و وضعیت جدید اتصال، ESTABLISHED خواهد شد. قطع یک اتصال می تواند توسط هر یک از طرفین شروع شود. وقتی فرآیند قطع اتصال تکمیل شد، اتصال به وضعیت قبلی CLOSED باز خواهد گشت.

توصیف	وضعیت (حالت)
هیچ اتصالی فعال یا معلق (و منتظر) نیست.	CLOSED
سرویس دهنده منتظر یک تماس ورودی (تقاضای اتصال) است.	LISTEN
یک تقاضای برقراری اتصال دریافت شده است. منتظر ACK آن است.	SYN RCVD
برنامه کاربردی (مشری) شروع به ایجاد یک اتصال کرده است.	SYN SENT
وضعیت عادی مبادله داده (وضعیت برقراری اتصال)	ESTABLISHED
برنامه کاربردی اعلام کرده کارش به اتمام رسیده است.	FIN WAIT 1
طرف مقابل نیز با قطع اتصال و خاتمه تماس موافقت کرده است.	FIN WAIT 2
حالت انتظار برای آنکه تمام بسته های سرگردان از بین بروند.	TIMED WAIT
طرفین بطور همزمان سعی در بستن اتصال کرده اند.	CLOSING
طرف مقابل مراحل قطع ارتباط را آغاز کرده است.	CLOSE WAIT
حالت انتظار برای آنکه تمام بسته های سرگردان ACK از بین بروند.	LAST ACK

شکل ۳۲-۶. وضعیتهایی که در مدل «ماشین حالت محدود» از مدیریت اتصال TCP بکار می رود.

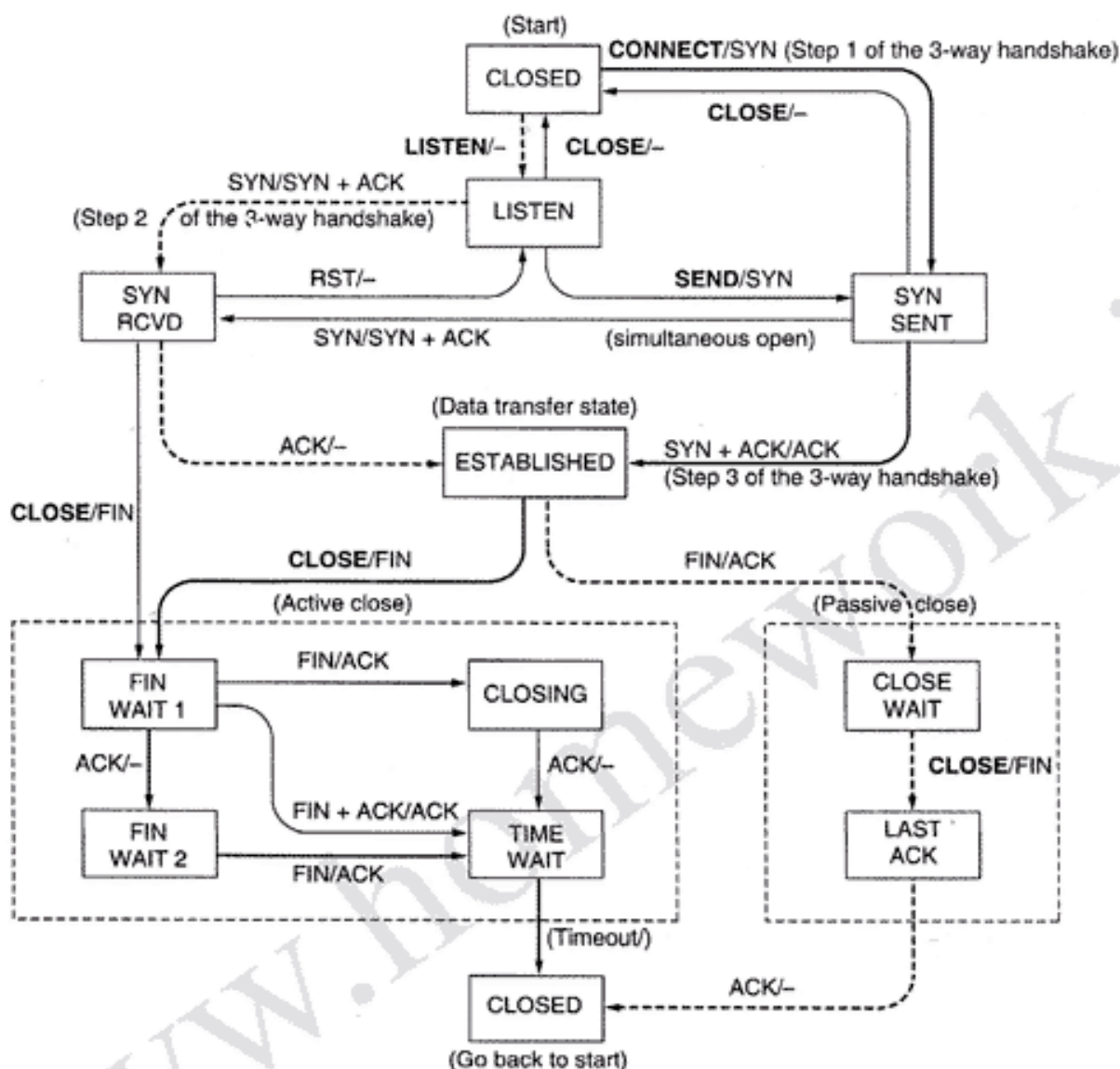
«ماشین حالت محدود» برای TCP، در شکل ۶-۳۳ به تصویر کشیده شده است. حالتی که در آن ماشین مشتری بطور فعال به یک سرویس دهنده غیر فعال متصل می شود با خطوط تیره رنگ نشان داده شده است: خطوط توپر برای تقاضای مشتری و نقطه چین برای پاسخ سرویس دهنده. خطوط کم رنگ، توالی رخدادهای نامتعارف را نشان می دهند. کنار هر خط در شکل ۶-۳۳ یک زوج مشخصه به صورت event/action (رخداد/کنش) نوشته شده است. هر رخداد می تواند ناشی از یک فراخوانی سیستمی باشد که توسط کاربر آغاز می شود (مثل فراخوانیهای LISTEN، CONNECT، SEND یا CLOSE)، یا در اثر دریافت یک قطعه TCP (مثل SYN، FIN، ACK یا RST) بروز کند و یا در اثر انقضای مهلت تایمر حادث شود. «کنشی» که در حین بروز یک رخداد انجام می شود، ارسال یک قطعه کنترلی (مثل SYN، FIN یا RTS) است یا هیچ کاری صورت نمی گیرد (که در شکل با علامت - مشخص شده است). توضیحات لازم درون پرانتز نشان داده شده است.

برای فهم دیاگرام، بهتر آنست که ابتدا مسیر منشعب از مشتری (خطوط توپر ضخیم) و بعد از آن مسیر سرویس دهنده (خطوط ضخیم نقطه چین) دنبال شود. وقتی برنامه کاربردی بر روی ماشین مشتری تقاضای CONNECT صادر می کند، «واحد TCP» در آن ماشین، ابتدا یک رکورد برای آن اتصال ایجاد کرده و پس از علامتگذاری اتصال در وضعیت SYNSENT یک قطعه SYN ارسال می نماید. دقت کنید که ممکن است چندین اتصال از طرف چند برنامه کاربردی بطور همزمان باز شده (یا در حال باز شدن) باشد فلذا به ازای هر اتصال یک «رکورد وضعیت» مستقل ایجاد و وضعیت آن اتصال در رکورد مربوطه درج می شود. وقتی قطعه کنترلی حاوی SYN+ACK دریافت می شود، TCP با ارسال ACK نهایی، فرآیند دست تکانی سه مرحله ای را تکمیل کرده و به ESTABLISHED تغییر وضعیت می دهد. در این وضعیت می توان داده فرستاد یا دریافت کرد.

وقتی کار برنامه کاربردی به پایان رسید، تابع اولیه CLOSE را اجرا می کند. این کار موجب خواهد شد که «واحد انتقال TCP» یک قطعه کنترلی FIN فرستاده و منتظر ACK مربوطه بماند. (مستطیل نقطه چین با عنوان active close نشان داده شده است). وقتی ACK مربوطه دریافت شود، وضعیت اتصال به حالت FIN WAIT2 تغییر کرده و یک جهت از اتصال بسته خواهد شد. وقتی طرف مقابل نیز اتصال را ببندد، یک بسته FIN از راه می رسد و وصول آن تصدیق می شود. حال اگرچه هر دو طرف اتصال را بسته اند ولیکن TCP بایستی به اندازه دو برابر طول عمر حداکثر بسته ها صبر کند تا تمام بسته هایی که احتمالاً از اتصال قبلی در شبکه سرگردان هستند از بین بروند. به محض آن که مهلت چنین تایمیری به پایان رسید، TCP رکورد اتصال مربوطه را حذف خواهد کرد.

حال مدیریت اتصال را از دیدگاه سرویس دهنده بررسی می نماییم: سرویس دهنده با انجام عمل LISTEN منتظر می ماند تا کسی تماس بگیرد. وقتی یک SYN وارد می شود، دریافت آن بلافاصله تصدیق شده و سرویس دهنده به وضعیت SYN RCVD وارد می شود. وقتی SYN-ACK ارسال سرویس دهنده توسط مشتری اعلام وصول شد، فرآیند دست تکانی سه مرحله ای تکمیل می گردد و سرویس دهنده به وضعیت ESTABLISHED می رود. انتقال داده ها اکنون می تواند شروع شود.

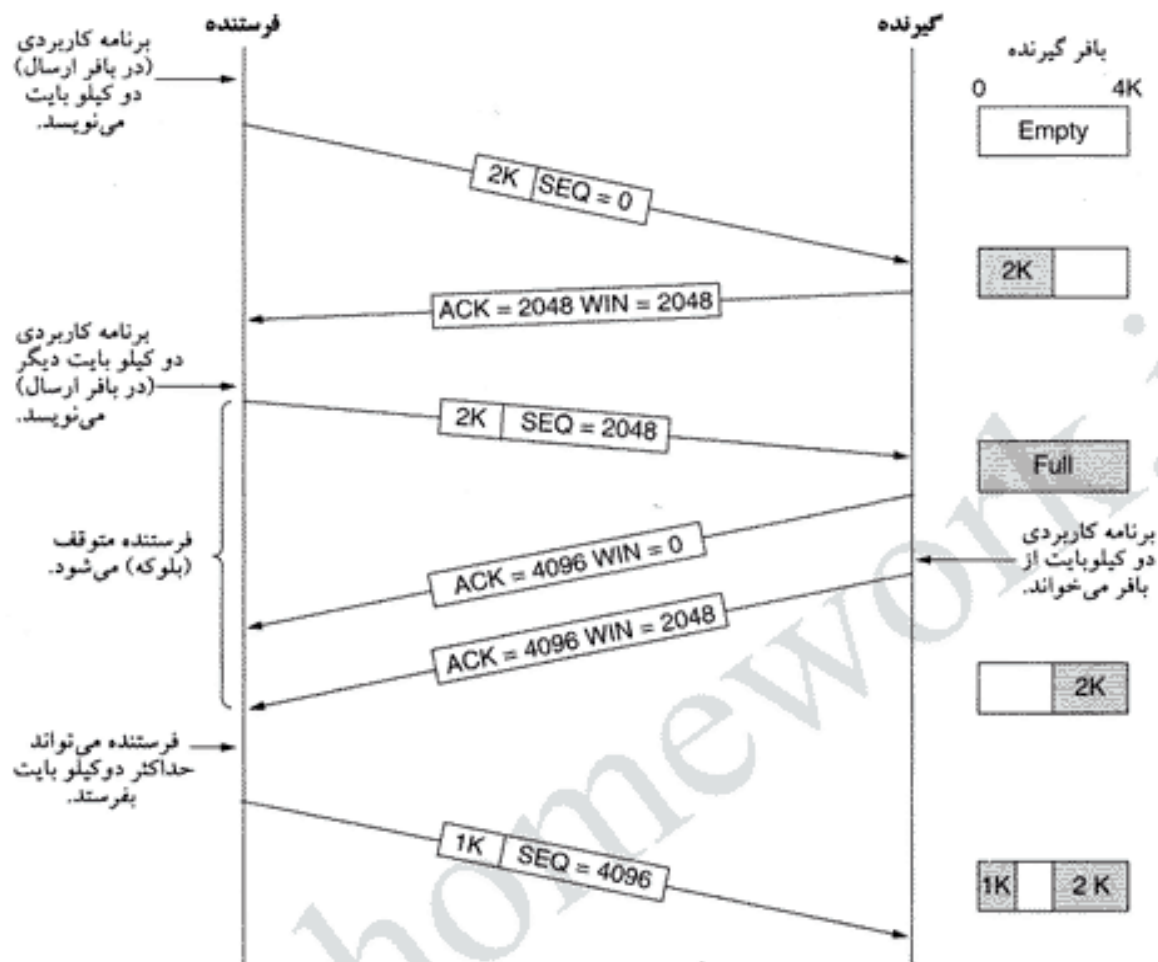
وقتی کار مشتری به انجام رسید، اقدام به صدور فرمان CLOSE می کند. این کار موجب می شود که سرویس دهنده، FIN دریافت کند. (مستطیل نقطه چین، خاتمه غیر فعال^۱ را نشان می دهد). در این لحظه به سرویس دهنده سیگنال داده می شود. هر گاه او نیز فرمان CLOSE را صادر کند، یک قطعه حاوی FIN برای مشتری ارسال می شود. وقتی مشتری دریافت FIN را تأیید کرد، سرویس دهنده به اتصال مربوطه خاتمه داده و رکورد متناظر با آن اتصال را از جدول خود حذف می نماید.



شکل ۶-۳۳. «ماشین حالت محدود» برای مدیریت اتصال TCP. خطوط ضخیم توپر مسیر طبیعی عملکرد مشتری را نشان می دهد. خطوط ضخیم نقطه چین مسیر طبیعی عملکرد سرورس دهنده را نشان می دهد. خطوط نازک توپر بروز رخدادهای نامعمول را نشان می دهد. بر روی هر «گذار» (Transition) برجسی وجود دارد که «رخداد» و «کنش» مربوطه را مشخص می کند.

۸.۵.۶ سیاستهای انتقال در TCP

همانگونه که قبلاً اشاره شد، مدیریت پنجره در TCP، وابستگی مستقیمی به تصدیق دریافت داده ها (Ack) ندارد (در حالی که در اغلب پروتکل های لایه پیوند داده اینگونه است). به عنوان مثال در شکل ۶-۳۴ فرض کنید که گیرنده ۴۰۹۶ بایت فضای بافر در اختیار دارد. اگر فرستنده یک قطعه ۲۰۴۸ بایتی ارسال کند و به درستی و بدون خطا دریافت شود، گیرنده بلافاصله وصول آنرا تأیید خواهد کرد. با این وجود، از آنجایی که پس از دریافت این قطعه، فقط ۲۰۴۸ بایت از فضای بافر خالی است (مگر آن که برنامه کاربردی بخشی از داده ها را از بافر بردارد) فلذا به طرف مقابل خود اعلام می دارد که از بایت بعدی (که شماره آن در فیلد Ack.No. مشخص شده) فقط حق ارسال ۲۰۴۸ بایت داده دارد. یا بعبارت فنی اعلام می کند، پنجره ۲۰۴۸ بایتی از شماره بایتی که در فیلد ۳۲ بیتی



شکل ۶-۳۴. مدیریت پنجره در TCP.

Ack. No. مشخص شده، آغاز می‌گردد.

حال فرستنده، ۲۰۴۸ بایت داده دیگر ارسال می‌نماید و دریافت آنها نیز تأیید می‌شود ولیکن، اندازه پنجره، صفر اعلام می‌شود. فرستنده باید متوقف شود و آنقدر منتظر بماند تا پروسه کاربردی گیرنده داده‌ها، مقداری داده از بافر بردارد و TCP بتواند پنجره بزرگتری را اعلام کند.

وقتی اندازه پنجره صفر اعلام شده، فرستنده عموماً نمی‌تواند قطعه دیگری ارسال کند مگر در دو مورد استثنا: اول «داده‌های اضطراری» (Urgent Data)؛ برای آنکه به کاربر اجازه بدهیم مثلاً پروسه کاربردی اجرا شده بر روی ماشین راه دور را از بین ببرد.^۱ دوم آن که فرستنده ممکن است قطعه‌ای حاوی یک بایت داده ارسال کند تا گیرنده وادار شود اندازه پنجره خود و شماره ترتیب بایتی را که از آن به بعد منتظر دریافت است، از نو اعلام نماید. استاندارد TCP، از این گزینه برای اجتناب از بروز بن‌بست بهره گرفته است تا در صورت عدم اعلام طول پنجره، پروسه متوقف شده در سمت گیرنده، تا ابد منتظر نماند.

فرستنده ملزم نیست که به محض تحویل گرفتن داده‌ها از برنامه کاربردی، فوراً آنها را ارسال نماید. گیرنده داده‌ها نیز مجبور نیست به محض دریافت و در اسرع زمان، دریافت آنها را تصدیق نماید. به عنوان مثال در شکل ۶-۳۴ وقتی اولین دو کیلو بایت از برنامه کاربردی دریافت شد، TCP با اطلاع از آن که ۴ کیلو بایت فضای بافر در

۱. به اصطلاح یونیکس kill کند.

اختیار دارد، می تواند ارسال آن را به تأخیر بیندازد تا ۲ کیلوبایت بعدی نیز دریافت شود و هر چهار کیلوبایت را یکجا ارسال نماید. این آزادی عمل می تواند به افزایش کارایی TCP کمک کند.

حال یک اتصال Telnet را مد نظر قرار بدهید که با فشار هر کلید در «ویرایشگر محاوره ای» (Interactive Editor)، از راه دور باید واکنش نشان بدهد. در بدترین حالت، وقتی که یک کاراکتر تکی جهت ارسال به «واحد انتقال TCP» تحویل می شود، TCP یک قطعه ۲۱ بایتی برای آن تشکیل داده و آن را به IP می دهد. IP نیز آن را به صورت یک دیتاگرام ۴۱ بایتی ارسال می نماید. (۲۰ بایت سرآیند TCP + ۲۰ بایت سرآیند IP + ۱ بایت کاراکتر ارسالی). در سمت گیرنده، TCP بلافاصله وصول آن را با ارسال یک دیتاگرام ۴۰ بایتی تصدیق می کند. بعداً وقتی برنامه ویرایشگر این بایت را می خواند، TCP بار دیگر مقدار جدید پنجره خود را که فقط یک واحد افزایش داشته، اعلام می دارد. این بسته نیز ۴۰ بایتی است. در آخر نیز وقتی برنامه ویرایشگر، کاراکتر ارسالی را پردازش کرد، آن را در قالب یک بسته ۴۱، بازگشت می دهد. (یا به عبارتی آن را Echo می کند). بدین نحو، به ازای هر کاراکتر تایپ شده، ۱۶۴ بایت از پهنای باند مصرف و جمعاً چهار قطعه TCP مبادله می شود. وقتی پهنای باند ارزشمند و محدود باشد این روش، بهیچوجه کارآمد نیست.

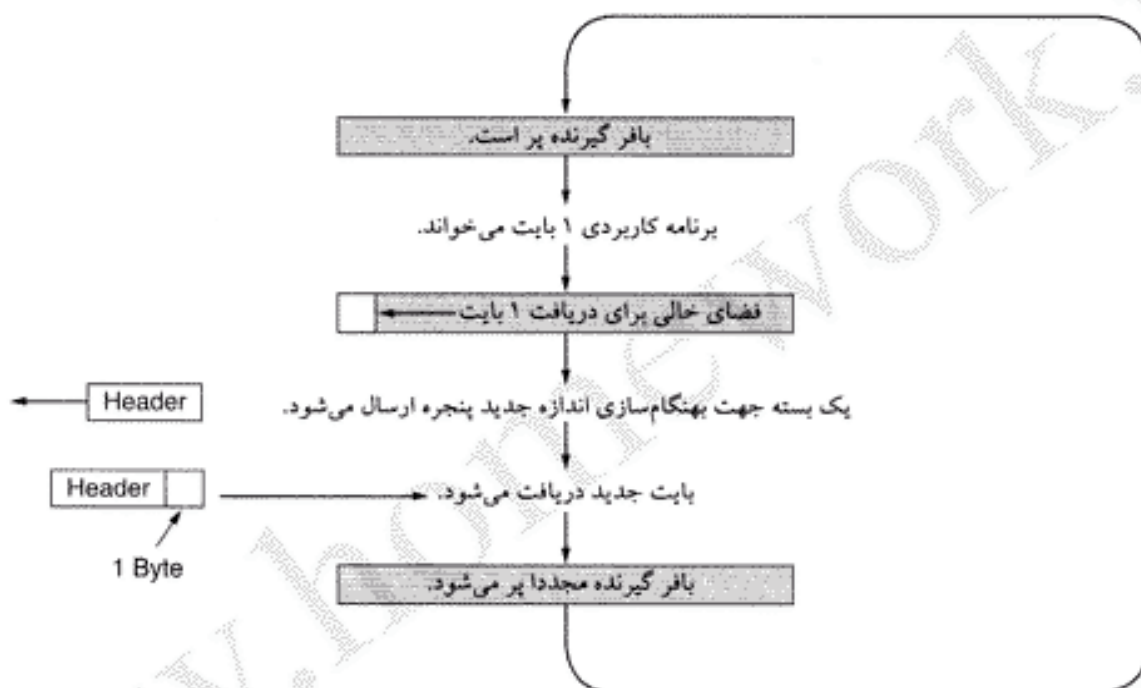
یک راهکار که در بسیاری از پیاده سازیهای عملی TCP برای بهینه سازی این وضعیت مورد استفاده قرار گرفته، آنست که اعلام وصول داده ها و اعلام اندازه پنجره، به مدت ۵۰۰ میلی ثانیه به تعویق بیفتد، بدین امید که داده هایی جهت ارسال تولید و در بسته ای که باید ارسال می شد به رایگان حمل شوند. با فرض آنکه برنامه ویرایشگر در مثال بالا، هر کاراکتر را پس از ۵۰۰ میلی ثانیه بازگشت بدهد (Echo کند)، فقط به ارسال یک بسته ۴۱ بایتی برای کاربر نیاز است و تعداد بسته ها و میزان مصرف پهنای باند به نصف کاهش می یابد.

اگرچه این قاعده بار تحمیل شده توسط گیرنده بر روی شبکه را کاهش می دهد ولیکن هنوز هم فرستنده بسیار ناکارآمد عمل می کند چرا که برای ارسال یک بایت، ۴۱ بایت منتقل می شود. یک روش برای کاهش مصرف پهنای باند، «الگوریتم ناگل» (Nagle's Algorithm) نام دارد. (Nagle, 1984) آنچه که ناگل پیشنهاد کرد ساده بود: وقتی داده ها به صورت تک بایتی جهت ارسال تحویل فرستنده می شوند، فرستنده اولین بایت را ارسال می نماید بقیه آنها را تا وقتی که دریافت همین یک بایت تأیید شود، بافر می کند. پس از اعلام وصول اولین بایت، فرستنده تمام کاراکترهای بافر شده را در قالب یک قطعه TCP به یکباره ارسال می کند و مجدداً تا اعلام وصول آن، به بافر کردن کاراکترها ادامه می دهد. با این روش، اگر سرعت تایپ کاربر زیاد و شبکه بسیار کند باشد، تعداد قابل توجهی کاراکتر در قالب یک قطعه TCP ارسال می شود و پهنای باند مورد نیاز، کاهشی چشمگیر خواهد داشت. مضاف بر این، «الگوریتم ناگل» اجازه می دهد که بسته جدید فقط زمانی ارسال شود که داده ها نیمی از فضای پنجره را پر کرده باشند و یا از طول مجاز یک قطعه TCP بیشتر شده باشد.

«الگوریتم ناگل» بطور گسترده ای در پیاده سازیهای عملی TCP بکار گرفته شده است ولیکن در برخی از مواقع، غیرفعال کردن آن مفیدتر است. خصوصاً وقتی که از برنامه کاربردی X Window در اینترنت استفاده می شود، حرکت ماوس باید به کامپیوتر راه دور ارسال شود. (سیستم X Window یک سیستم گرافیکی مبتنی بر پنجره است که در اغلب سیستمهای یونیکس از آن استفاده می شود. کاربران راه دور می توانند از آن جهت ورود از راه دور به سیستم یونیکس و تعامل با آن بهره بگیرند). استفاده از الگوریتم ناگل موجب می شود که TCP ارسال حرکات ماوس را جمع آوری کرده تا آنها را به یکباره ارسال نماید و این کار حرکت ماوس را غیرطبیعی جلوه می دهد و اسباب ناخشنودی کاربران را فراهم می آورد.

یکی دیگر از مشکلاتی که می تواند کارایی TCP را کاهش بدهد، «سندرم پنجره ناموزون» نام دارد. (Silly Window Syndrome, Clark, 1982) این مسئله زمانی رخ می دهد که داده ها در قالب بلوکهای بزرگ به

واحد انتقال TCP تحویل شود ولیکن برنامه کاربردی گیرنده در طرف مقابل، داده‌ها را به صورت بایت به بایت بخواند! برای درک بهتر این مشکل به شکل ۶-۳۵ نگاه کنید. در ابتدا بافر TCP در سمت گیرنده پر است و فرستنده از این موضوع اطلاع دارد. (به عبارت دیگر اندازه پنجره صفر اعلام شده است). سپس برنامه کاربردی از استریم TCP (یا به عبارتی از بافر)، یک کاراکتر می‌خواند. این کار موجب می‌شود که TCP با ارسال بسته‌ای مقدار جدید پنجره خود را به اطلاع فرستنده برساند و به او تفهیم کند که اجازه ارسال فقط یک بایت دارد! فرستنده اطاعت کرده و یک بایت ارسال می‌کند. حال بافر مجدداً پر می‌شود و ضمن تصدیق وصول این یک بایت، اندازه پنجره صفر اعلام می‌شود. این رفتار می‌تواند تا بی‌نهایت ادامه داشته باشد.



شکل ۶-۳۵. سندروم پنجره ناموزون (Silly Window Syndrome).

راه حل پیشنهادی «کلارک» (Clark) آن بود که گیرنده برای یک بایت، مقدار جدید اندازه پنجره خود را اعلام ننماید؛ در عوض باید آنقدر منتظر شود تا فضای موجود در بافر به حد متناسبی برسد، آنگاه این مقدار اعلام گردد. به ویژه، گیرنده نباید مقدار جدید اندازه پنجره خود را اعلام کند مگر آن که قادر باشد یک قطعه داده را با طولی که در ابتدای برقراری اتصال به پذیرش آن متعهد شده، در بافر خود بپذیرد یا آن که حداقل نیمی از بافرش خالی شده باشد.

مضاف بر این، فرستنده می‌تواند با نفرستادن قطعات کوچک، به کاهش مشکل کمک کند: فرستنده باید سعی کند که قبل از ارسال یک قطعه آنقدر منتظر بماند تا داده‌های کافی متناسب با اندازه یک قطعه کامل یا معادل با نصف بافر گیرنده جمع‌آوری شود. (اندازه بافر گیرنده را می‌توان براساس اعلام‌های متوالی اندازه پنجره، تخمین زد).

الگوریتم ناگل و راه حل کلارک برای درمان «سندروم پنجره ناموزون»، مکمل یکدیگر هستند. ناگل سعی می‌کند مشکلی را حل کند که در اثر تحویل بایت به بایت داده‌ها توسط برنامه کاربردی به TCP پدید می‌آید. کلارک نیز سعی می‌کند عکس این مشکل را یعنی وقتی که برنامه کاربردی داده‌های خود را از TCP بایت به بایت

تحويل می گیرد، حل و فصل نماید. هر دوی این راهکارها معتبرند و در کنار هم کار می کنند. هدف آن است که فرستنده، قطعات کوچک نفرستد و گیرنده نیز قطعات را در اندازه کوچک تحويل نگیرد.

در سمت گیرنده، TCP می تواند به غیر از بکارگیری روش کلارک، به گونه دیگری نیز برای بهبود کارایی اقدام نماید. همانند فرستنده، گیرنده نیز می تواند داده ها را بافر نماید؛ یعنی درخواست برنامه کاربردی جهت خواندن داده ها را آنقدر معلق نگاه دارد تا آن که یک توده بزرگ داده جمع آوری شود. انجام این کار تعداد فراخوانیهای TCP را کاهش داده و طبعاً سربار سیستم کمتر می شود. البته این کار زمان تایمر و تأخیر اجرای فرمان READ را افزایش خواهد داد ولیکن برای کاربردهایی نظیر انتقال فایل، کارایی بیشتر ارجح تر از زمان پاسخ سریع است.

مورد دیگری که گیرنده با آن روبروست، دریافت قطعات داده نامرتب است. گیرنده می تواند بر حسب شرایط بسته هایی را که خارج از ترتیب می رسند، نگاه دارد یا آنها را دور بریزد. البته وصول داده ها را می توان فقط زمانی اعلام کرد که همه داده ها تا شماره ای که اعلام می شود دریافت شده باشد. اگر گیرنده قطعات ۰، ۱، ۲، ۳، ۴، ۵، ۶ و ۷ را (به استثنای ۳) دریافت کند تنها می تواند دریافت داده ها تا آخرین بایت قطعه شماره ۲ را تصدیق نماید. وقتی مهلت فرستنده منقضی می شود، تمام قطعات از شماره ۳ به بعد از نو ارسال می شوند. اگر گیرنده، قطعات ۴ تا ۷ را بافر کرده باشد به محض دریافت قطعه ۳ می تواند دریافت تمام بایتهای را تا آخر قطعه هفتم، تأیید کند.

۹.۵.۶ کنترل ازدحام در TCP

هر گاه بار تحويل شده به شبکه بیش از ظرفیتی باشد که می تواند از عهده آن برآید، ازدحام پدید خواهد آمد. اینترنت نیز از این مشکل مستثنی نیست. در این بخش به تشریح الگوریتمهایی خواهیم پرداخت که در طول ربع قرن گذشته برای حل و فصل مشکل ازدحام توسعه یافته اند. اگرچه لایه شبکه نیز در مدیریت ازدحام می کوشد ولی بار سنگین این مسئولیت بیشتر بر عهده TCP می باشد چرا که راه حل واقعی رفع ازدحام، کاهش نرخ ارسال داده ها در این لایه است.

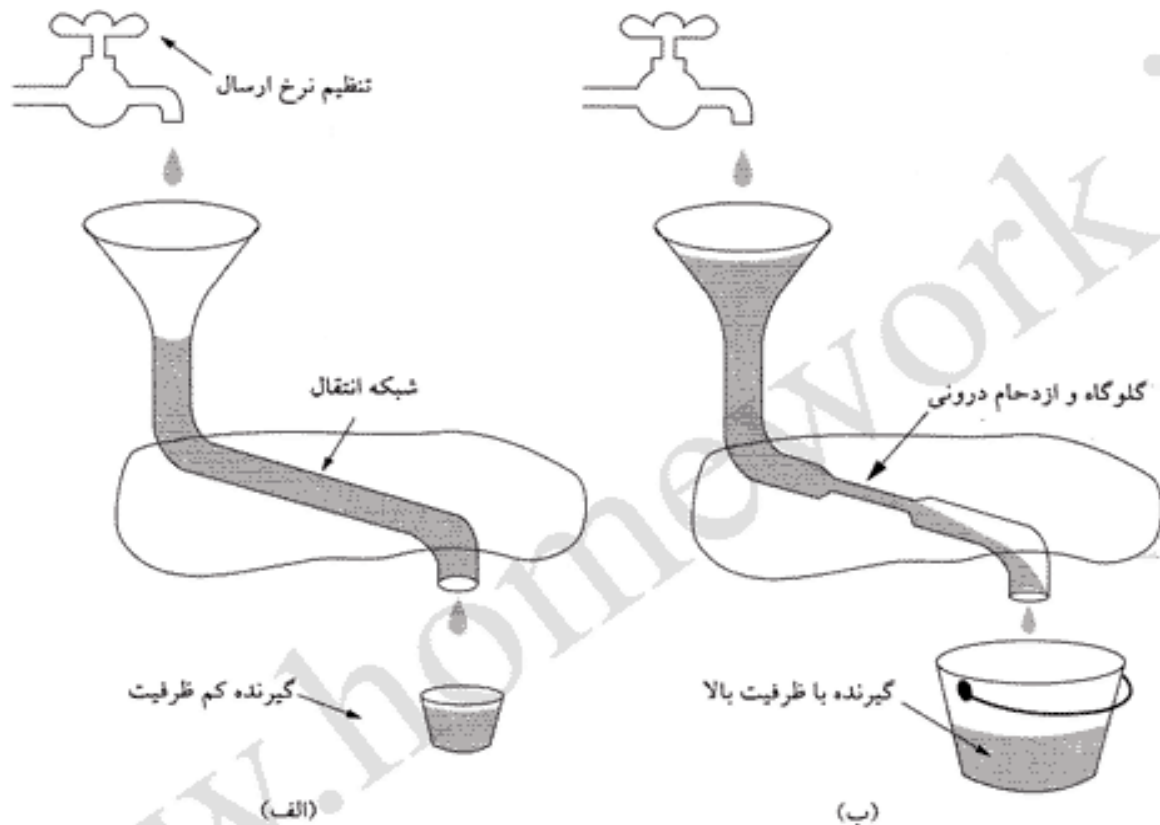
از دیدگاه تئوری، با بکارگیری یک اصل از دانش فیزیک می توان به مدیریت ازدحام پرداخت: «اصل بقای بسته ها»! ایده اصلی مبنی بر آن است که وقتی بسته قبلی از شبکه خارج شد (یعنی به مقصد تحويل گردید)، بسته جدیدی به شبکه تزریق شود. TCP سعی می کند با تنظیم پویا و خودکار اندازه پنجره، بدین هدف نایل آید.

اولین گام در مدیریت ازدحام، تشخیص آن است. در روزگاران گذشته، تشخیص ازدحام بسیار دشوار بود. در آن دوران، به دو دلیل بسته ای از دست می رفت و مهلت فرستنده منقضی می شد: (۱) نویز روی خطوط انتقال (که می توانست بسته ها را نابود کند) (۲) حذف بسته ها توسط مسیریاب دچار ازدحام. تشخیص آنکه کدامیک از عوامل فوق منجر به از دست رفتن یک بسته شده، چندان ساده نبود.

امروزه، از بین رفتن بسته ها در اثر خطای خطوط انتقال پدیده ای بسیار نادر است زیرا اغلب شاهراههای ارتباطی از جنس فیبرنوری هستند. (هر چند شبکه های بی سیم داستان دیگری دارد.) طبعاً می توان نتیجه گرفت که انقضای مهلت ارسال در شبکه اینترنت ناشی از ازدحام است. تمام الگوریتمهای TCP در اینترنت فرض را بر آن گذاشته اند که انقضای مهلت (Timeout) و عدم وصول بسته به دلیل بروز ازدحام بوده است و به همین دلیل TCP بر زمانهای Timeout (یعنی زمان انقضای مهلت اعلام وصول هر بسته) به دقت نظارت می کند.

قبل از تشریح واکنش TCP در برخورد با ازدحام، ابتدا بررسی می کنیم که این پروتکل چه تلاشی در پیشگیری از بروز آن می کند. پس از برقراری یک اتصال، بایستی اندازه مناسبی برای پنجره انتخاب شود. گیرنده می تواند اندازه پنجره را بر حسب بافر در اختیار خود، تعیین نماید. اگر فرستنده به اندازه پنجره گیرنده پایبند باشد مشکلی از بابت سرریز شدن بافر طرف گیرنده پیش نخواهد آمد ولیکن هنوز هم احتمال بروز مشکلات ناشی از ازدحام درون یک شبکه، وجود دارد.

در شکل ۶-۳۶، این مشکل را با تعبیر هیدرولیکی آن به تصویر کشیده ایم. در شکل ۶-۳۶-الف، یک لوله قطور را مشاهده می‌کنیم که به یک گیرنده کم حجم منتهی شده است. مادامی که فرستنده بیش از ظرفیت سطل، آب تولید و ارسال نکند، سطل هم سرریز نخواهد شد. در شکل ۶-۳۶-ب، عامل محدودکننده، ظرفیت سطل نیست بلکه ظرفیت داخلی حمل زیر شبکه، ایجاد محدودیت کرده است. اگر حجم زیاد آب با سرعت بالا به قیف وارد شود، آن را سریعاً پر کرده و آب به هدر خواهد رفت (در این حالت به دلیل سرریزی قیف).



شکل ۶-۳۶. (الف) یک شبکه سریع که یک گیرنده با ظرفیت کم را تغذیه می‌کند. (ب) یک شبکه کند که یک گیرنده با ظرفیت بالا را تغذیه می‌کند.

راه حل اینترنت آن است که عوامل بروز این دو مشکل بالقوه را پذیرفته و با هر یک از آنها بطور مجزا و مستقل برخورد کنیم.^۱ برای این کار هر فرستنده دو پنجره ایجاد می‌نماید: پنجره اول که براساس اعلام گیرنده طرف مقابل ایجاد می‌شود و پنجره دوم، «پنجره ازدحام» (Congestion Window). هر یک از این پنجره‌ها تعداد بایتهایی را مشخص می‌کنند که فرستنده می‌تواند ارسال کند. تعداد بایتهایی که فرستنده مجاز به ارسال آنهاست، مینیمم اندازه این دو پنجره است. بنابراین اندازه مؤثر پنجره، مقدار مینیمم آنچیزی است که فرستنده فکر می‌کند صحیح است و آنچه که گیرنده فکر می‌کند آن باید باشد! مثلاً اگر گیرنده عنوان کند که «هشت کیلوبایت بفرست» ولی فرستنده بداند که ارسال بیش از چهار کیلوبایت شبکه را دچار انسداد می‌کند، چهار کیلوبایت ارسال خواهد کرد. برعکس اگر گیرنده اعلام کند که «هشت کیلوبایت بفرست» و فرستنده نیز بداند که ارسال تا ۳۴ کیلوبایت بلامانع است، فقط هشت کیلوبایت درخواستی را ارسال خواهد کرد.

۱. تفکیک عامل «ظرفیت شبکه» از «ظرفیت گیرنده».

وقتی اتصال برقرار می شود، فرستنده، اندازه «پنجره ازدحام» را با طول حداکثر هر قطعه که در حین اتصال توافق شده، مقداردهی اولیه می کند؛ سپس یک قطعه با طول حداکثر می فرستد. اگر اعلام وصول این قطعه قبل از انقضای مهلت مقرر دریافت شد، اندازه پنجره ازدحام را به اندازه طول حداکثر قطعه اضافه می کند و دفعه بعدی معادل دو قطعه ارسال می نماید. مادامی که پس از ارسال هر قطعه دریافت آن تصدیق می شود، به اندازه «پنجره ازدحام» معادل با طول حداکثر هر قطعه اضافه خواهد شد. وقتی اندازه پنجره ازدحام معادل با طول n قطعه باشد و تمام قطعات ارسالی سر موعد اعلام وصول شوند به پنجره ازدحام معادل با طول کل n قطعه (برحسب بایت) اضافه خواهد شد. کوتاه سخن آن که اگر در هر بار ارسال (معادل با n قطعه) تمام آنها اعلام وصول شوند، طول پنجره ازدحام دو برابر می شود.

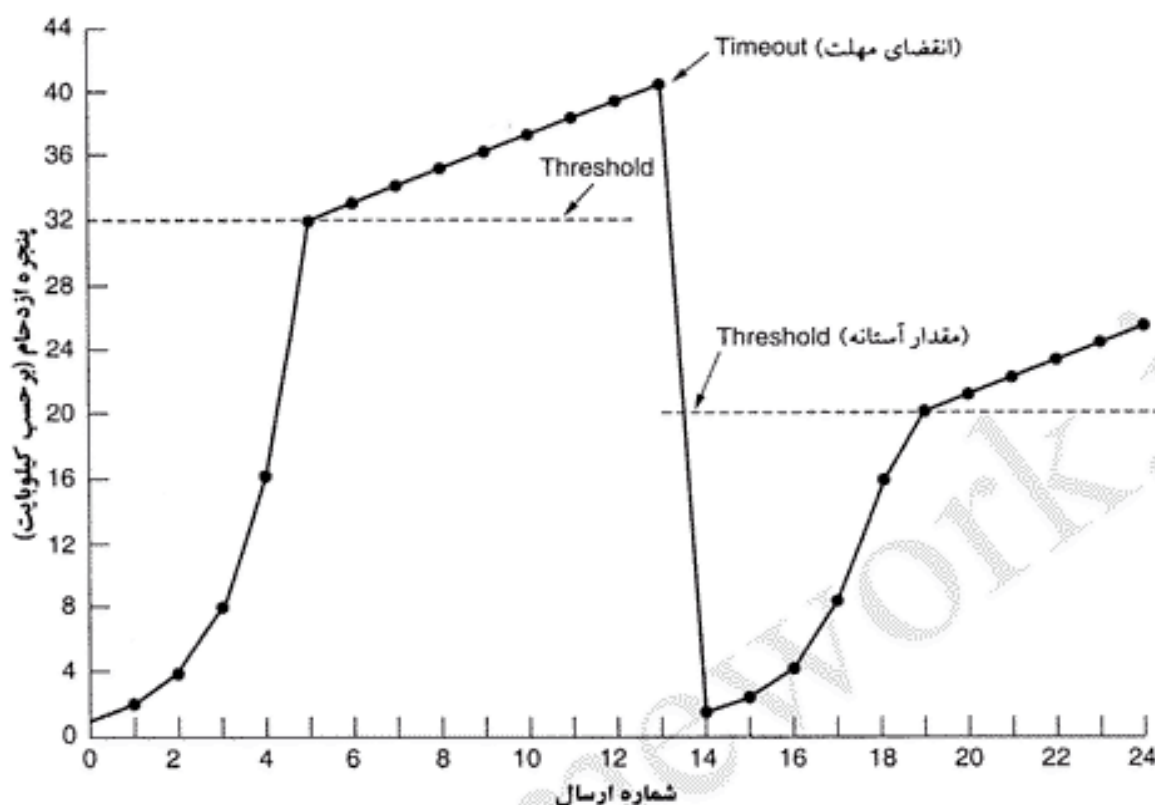
اندازه «پنجره ازدحام» آنقدر به صورت نمایی رشد می کند تا آنکه پس از ارسال قطعات، یا وصول آنها تصدیق نشود و یا آنکه به اندازه پنجره گیرنده برسد. به عنوان مثال اگر اندازه پنجره ۱۰۲۴ بایتی، ۲۰۴۸ بایتی، ۴۰۹۶ بایتی به درستی عمل کند ولی ارسال چندین قطعه معادل با ۸۰۹۶ بایت، با مشکل انقضای مهلت اعلام وصول (Timeout) مواجه شود برای اجتناب از ازدحام، اندازه پنجره به ۴۰۹۶ بایت تنظیم می شود. مادامی که پنجره ازدحام بر روی ۴۰۹۶ ثابت می ماند حتی اگر طول پنجره اعلام شده توسط گیرنده از ۴۰۹۶ بیشتر باشد، فرستنده هیچگاه قطعاتی را که مجموع طول آنها از ۴۰۹۶ بیشتر می شود نخواهد فرستاد. این الگوریتم اصطلاحاً «شروع آهسته» (Slow Start) نام گرفته ولیکن هرگز کند عمل نمی کند زیرا روند آن نمایی است. (Jacobson, 1988) تمام پیاده سازیهای TCP، ملزم به حمایت از این الگوریتم هستند.

حال اجازه بدهید به الگوریتم کنترل ازدحام در اینترنت بپردازیم: در اینترنت به غیر از پنجره گیرنده و پنجره ازدحام، از پارامتر سومی به نام «آستانه» (Threshold) استفاده می شود. مقدار اولیه این پارامتر 64KByte است. وقتی پس از ارسال قطعاتی، مهلت اعلام وصول آنها منقضی شود، پارامتر «آستانه» به نصف مقدار پنجره ازدحام تنظیم شده و مقدار پنجره ازدحام مجدداً به مقدار طول حداکثر یک قطعه بر می گردد. حال مجدداً الگوریتم «شروع آهسته» (Slow Start) برای تعیین اندازه مناسب طول پنجره ازدحام شروع به کار می نماید، با این تفاوت که رشد نمایی به محض رسیدن به مقدار آستانه، متوقف می شود. پس از رسیدن به نقطه آستانه، ارسال موفق یک قطعه اندازه پنجره را دو برابر نمی کند بلکه آن را به صورت خطی افزایش می دهد. طبقاً این الگوریتم حدس می زند که کاهش طول پنجره به نصف معقول است، فلذا از این نقطه شروع کرده و اندازه آنرا به تدریج افزایش می دهد.

برای آن که مشخص شود این الگوریتم کنترل ازدحام چگونه کار می کند به شکل ۶-۳۷ نگاه کنید. در اینجا اندازه حداکثر هر قطعه، ۱۰۲۴ بایت است. در ابتدا، اندازه «پنجره ازدحام»، ۶۴ کیلوبایت بوده ولی به دلیل انقضای مهلت اعلام وصول قطعه ها (Timeout)، مقدار آستانه به ۳۲ کیلوبایت تنظیم شده و اندازه پنجره ازدحام برای ارسال شماره صفر به مقدار ۱۰۲۴ کاهش یافته است. از آن به بعد پنجره ازدحام بطور نمایی رشد کرده تا به مقدار آستانه (یعنی 32KB) رسیده است. پس از آن رشد اندازه پنجره ازدحام به صورت خطی بوده است.

ارسال سیزدهم موفق نبوده است و مجدداً مشکل عدم اعلام وصول داده ها در مهلت مقرر، بروز کرده است. (بروز مشکل در این لحظه چندان دور از ذهن نیست.) لذا مجدداً مقدار آستانه به نصف اندازه پنجره فعلی تنظیم شده است. (در اینجا اندازه پنجره ۴۰ کیلوبایت بوده فلذا نصف آن، ۲۰ کیلوبایت می شود.) بار دیگر الگوریتم «شروع آهسته» کار خود را آغاز می کند. از آنجایی که در ارسال چهاردهم تا بیستم، پیغام ACK برگشته لذا در هر ارسال اندازه پنجره ازدحام دو برابر و از آن به بعد رشد پنجره خطی شده است.

اگر هیچگاه مهلت اعلام وصول منقضی نشود، پنجره ازدحام، رشد خود را آنقدر ادامه می دهد تا به اندازه پنجره گیرنده برسد. در این نقطه، رشد پنجره متوقف شده و مادامی که طول پنجره گیرنده تغییر نکند و مهلت اعلام



شکل ۶-۳۷. مثالی از الگوریتم کنترل ازدحام در اینترنت.

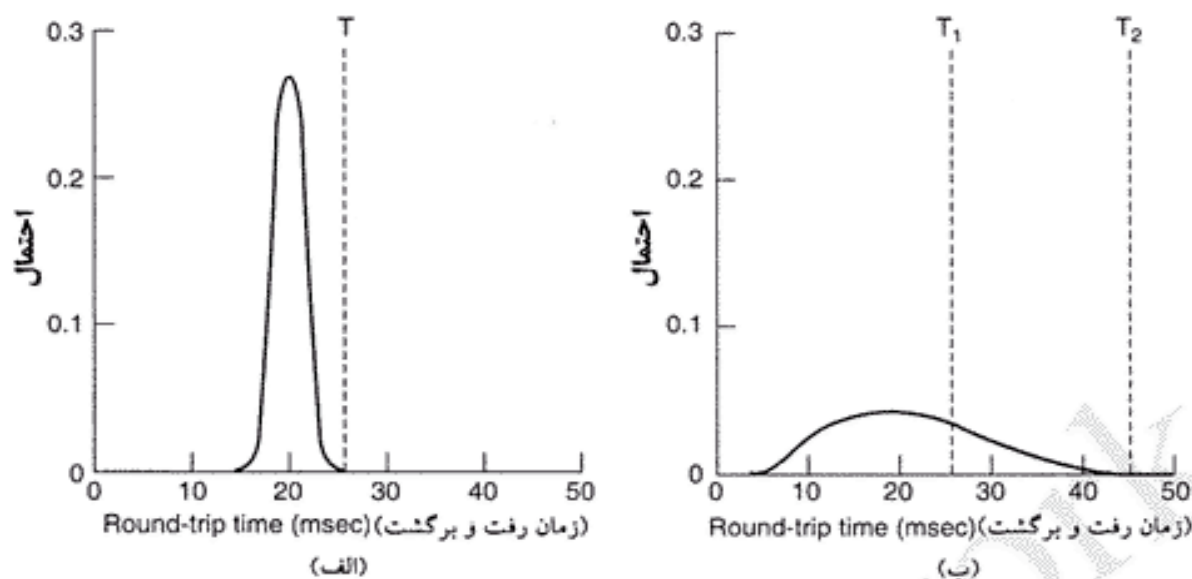
وصول قطعات ارسالی منقضی نشود، اندازه پنجره ازدحام ثابت خواهد ماند. منصف بر این اگر پیغام ICMP SOURCE QUENCH دریافت و تحویل TCP شود این رخداد نیز به مثابه انقضای مهلت تلقی می شود. راهکار جدیدتر در RFC 3168 تشریح شده است.

۱۰-۵۶ مدیریت تایمرها در TCP

TCP برای آن که بتواند کارش را بدرستی انجام بدهد از چندین تایمر بهره می گیرد. مهمترین آنها، «تایمر ارسال مجدد» (Retransmission Timer) است. وقتی قطعه ای ارسال می شود این تایمر شروع به کار می کند. اگر قبل از آن که مهلت این تایمر منقضی گردد، قطعه ارسالی اعلام وصول شود (یعنی Ack آن برگردد)، تایمر متوقف می شود ولیکن اگر وصول داده ها قبل از صفر شدن مقدار این تایمر تأیید نشود، این قطعه از نو ارسال و تایمر مجدداً راه اندازی می گردد. حال سؤال مهمی پیش می آید: زمان انقضای مهلت چقدر باید باشد؟

این مسئله در لایه انتقال از شبکه اینترنت در مقایسه با همین مسئله در پروتکل های لایه پیوند داده (فصل سوم)، دشوارتر و پیچیده تر است. در لایه پیوند داده، تأخیرها کاملاً قابل تخمین هستند (یا به عبارتی واریانس تأخیر پایین است) فلذا به نحوی که در شکل ۶-۳۸ الف مشاهده می شود می توان تایمر را به همان مقداری تنظیم کرد که انتظار می رود اعلام وصول بسته ها (یعنی Ack) قبل از این زمان برگردد. از آنجایی که در لایه پیوند داده ها، پیغام های Ack با تأخیر پیش بینی نشده مواجه نمی گردد (زیرا مشکل ازدحام وجود ندارد) فلذا عدم دریافت Ack، عموماً به معنای نابودی فریم ارسالی یا نابودی Ack بازگشتی تلقی می شود.

TCP با محیطی کاملاً متفاوت مواجه است. «تابع چگالی احتمال زمان بازگشت Ack» به جای آن که شبیه به شکل ۶-۳۸ الف باشد بیشتر شبیه به شکل ۶-۳۸ ب است. تعیین زمان رفت قطعه داده و بازگشت Ack آن



شکل ۶-۳۸. (الف) چگالی احتمال زمان دریافت پیغامهای تصدیق وصول (ACK) در لایه پیوند داده. (ب) چگالی احتمال زمان دریافت پیغامهای تصدیق وصول در TCP (لایه انتقال).

(Round Trip Time) پیچیده است. حتی اگر این زمان مشخص باشد، تصمیم‌گیری در خصوص مهلت دریافت Ack (یعنی مقدار اولیه تایمر) بسیار دشوار است. اگر زمان انقضای مهلت (یا به عبارتی مقدار اولیه تایمر ارسال مجدد) کوتاه در نظر گرفته شود (مثلاً مقدار T_1 در شکل ۶-۳۸-ب) آنگاه برخی از قطعات داده، بیهوده ارسال مجدد شده و اینترنت بایسته‌های زائد شلوغ می‌شود. برعکس اگر این مقدار بسیار طولانی در نظر گرفته شود (مثلاً مقدار T_2) آنگاه وقتی بسته‌ای از بین می‌رود به دلیل تأخیر بسیار زیادی که در ارسال مجدد بسته پیش می‌آید، کارایی بشدت افت می‌کند. مضاف بر این مقدار میانگین و واریانس زمان اعلام وصول بسته‌ها می‌تواند در عرض چند ثانیه تغییر جدی داشته باشد. (این تغییر می‌تواند در اثر بروز ازدحام یا رفع آن باشد).

راه حل نهایی، استفاده از یک الگوریتم کاملاً پویا (Dynamic) است به گونه‌ای که مدام مهلت بازگشت Ack بسته‌ها را تخمین زده و تنظیم نماید. این کار براساس اندازه‌گیری دائمی کارایی شبکه انجام می‌گیرد. الگوریتمی که عموماً در TCP بکار گرفته می‌شود توسط «ژاکوبسن» (۱۹۸۸) پیشنهاد شده و به نحو زیر کار می‌کند: TCP برای هر اتصال یک متغیر به نام RTT نگاه می‌دارد که مقدار آن بهترین تخمین از زمان رفت بسته به مقصد مورد نظر و بازگشت Ack آن می‌باشد. وقتی یک قطعه داده ارسال می‌گردد یک تایمر شروع به کار می‌کند تا اولاً مشخص شود بازگشت Ack چقدر طول می‌کشد، ثانیاً در صورتی که مهلت مقرر منقضی شد آنرا از نو ارسال نماید. اگر قبل از انقضای مهلت مقرر، دریافت قطعه تصدیق شود، TCP مقدار زمان رفت قطعه و برگشت Ack آنرا، (بکمک مقدار فعلی تایمر RTT) اندازه‌گیری می‌کند. این مقدار را M بنامید. سپس براساس M، مقدار جدید RTT را طبق فرمول زیر به‌نگام می‌نماید:

$$RTT = \alpha \cdot RTT + (1 - \alpha) \cdot M$$

در فرمول فوق، α ضریب هموارسازی (Smoothing Factor) است و تعیین می‌کند که مقدار قبلی RTT با چه وزنی در محاسبه مقدار جدید حضور دارد. α عموماً 7/8 است.

حتی با در اختیار داشتن مقدار RTT، انتخاب مقدار مناسب برای تایمر «ارسال مجدد» چندان ساده نیست. عموماً TCP مقدار تایمر را $\beta \cdot RTT$ در نظر می‌گیرد ولیکن مسئله اصلی انتخاب β است. در پیاده‌سازیهای اولیه

TCP، مقدار β همیشه ۲ در نظر گرفته می‌شد ولیکن تجربه نشان داد که انتخاب مقدار ثابت برای β مقدار مناسب و منعطفی نیست زیرا با افزایش واریانس، به درستی جواب نمی‌دهد.

در سال ۱۹۹۸ «ژاکوبسن» پیشنهاد کرد که β متناسب با «انحراف معیار تابع چگالی احتمال زمان دریافت پیامهای Ack»^۱ در نظر گرفته شود. بدین طریق اگر واریانس بالا باشد، β نیز زیاد خواهد بود (و بالعکس). همچنین او پیشنهاد کرد که برای بدست آوردن مقدار تخمینی انحراف معیار از «میانگین انحراف» استفاده شود. به همین دلیل در الگوریتم او به متغیر جدیدی به نام D نیاز است که مقدار تخمینی و هموارشده (Smoothed) «انحراف» را نگه می‌دارد: وقتی پیام Ack یک قطعه داده دریافت می‌شود، اختلاف بین مقدار مورد انتظار (یعنی RTT) و مقدار اندازه‌گیری شده (یعنی M) به صورت $|RTT-M|$ محاسبه می‌شود. سپس مقدار هموار شده D طبق رابطه زیر محاسبه می‌شود:

$$D = \alpha.D + (1 - \alpha).|RTT-M|$$

مقدار α در فرمول بالا می‌تواند با مقدار α در فرمول RTT یکسان باشد یا فرق کند. اگرچه مقدار D دقیقاً معادل با انحراف معیار نیست ولیکن به آن نزدیک است و برای محاسبات ما کفایت می‌کند. ژاکوبسن نشان داد که می‌توان فرمول بالا را با جمع، تفریق و شیفت چند عدد صحیح، پیاده‌سازی کرد. اغلب پیاده‌سازیهای عملی TCP از همین الگوریتم بهره گرفته‌اند و زمان انقضای مهلت (یعنی مقدار تایمر ارسال مجدد) را به صورت زیر تنظیم می‌کنند:

$$\text{Timeout} = RTT + 4 \times D$$

انتخاب ضریب ۴ اختیاری است ولیکن استفاده از این عدد دو مزیت دارد: اول آن که ضرب یک عدد صحیح در ۴ را می‌توان با یک عمل شیفت (دو بیت شیفت به چپ) انجام داد. دوم آن که از انقضای مهلت و ارسال مجدد و غیر ضروری جلوگیری می‌کند زیرا کمتر از یک درصد از بسته‌ها با تأخیری بیش از چهار برابر انحراف معیار دریافت می‌شوند. [به عبارت دیگر احتمال آن که بسته‌ای پیاده‌سازی ارسال مجدد شود کمتر از یک درصد است.] (در واقع، پیشنهاد ژاکوبسن عدد ۲ بود ولی تجربه نشان داد که عدد ۴، کارآمدتر و بهینه‌تر است).

مشکلی که در محاسبه پویای RTT بروز می‌کند آن است که وقتی پس از ارسال یک قطعه، مهلت اعلام وصول آن منقضی و قطعه از نو ارسال شد چه باید کرد؟ وقتی که پس از ارسال مجدد یک قطعه، Ack آن دریافت شد روشن نیست که آیا این Ack مربوط به قطعه اول است که دیر رسیده، یا آنکه واقعاً مربوط به قطعه دوم است. حدس اشتباه در این مورد می‌تواند، منجر به غلط شدن نتیجه تخمین RTT شود. شخصی به نام Phil Karn به این مسئله پی برد. وی یک آماتور علاقمند به سیستمهای رادیویی است که می‌خواست بسته‌های TCP/IP را به کمک کانالهای رادیویی که بشدت نامطمئن و توأم با خطا هستند ارسال نماید. (در شرایط خوب فقط نیمی از بسته‌ها سالم دریافت می‌شوند!) وی یک پیشنهاد ساده ارائه داد: برای بسته‌هایی که از نو ارسال می‌شوند، مقدار RTT بهنگام نگردد. در عوض مهلت دریافت پیام Ack، به دو برابر افزایش یابد. این اصلاحیه به نام الگوریتم Karn مشهور است و در پیاده‌سازی TCP از آن استفاده می‌شود.

«تایمر ارسال مجدد» (Retransmission Timer)، تنها تایمری نیست که TCP از آن استفاده می‌کند. تایمر دومی نیز به نام Persistent Timer وجود دارد. از این تایمر برای پیشگیری از بن‌بست ذیل استفاده می‌شود: فرض کنید گیرنده با ارسال Ack اندازه پنجره خود را به فرستنده صفر اعلام کرده و از او می‌خواهد که منتظر بماند. بعداً گیرنده اندازه جدید پنجره خود را بهنگام‌سازی و اعلام می‌کند ولیکن بسته حاوی مقدار جدید از بین می‌رود. حال گیرنده و فرستنده هر دو در انتظار یکدیگر به سر می‌برند. برای آن که این انتظار تا ابد ادامه نیابد، به محض

۱. Standard Deviation of Acknowledgement Arrival Time Probability Density Function

انقضای مهلت تایمر، فرستنده با ارسال بسته‌ای به گیرنده، سعی می‌کند او را بیازماید. در پاسخ بدین بسته، اندازه پنجره اعلام خواهد شد. اگر این مقدار کم‌تر از صفر باشد، تایمر مجدداً تنظیم شده و این روند تکرار می‌شود ولی اگر این مقدار غیر صفر باشد فرستنده می‌تواند داده‌های خود را ارسال نماید.

تایمر سومی که در برخی از پیاده‌سازیهایی عملی TCP از آن استفاده شده، Keepalive Timer نام دارد. وقتی اتصالی برای مدت زمان طولانی بیکار بماند و مهلت این تایمر منقضی شود، یکی از طرفین سعی می‌کند فعال بودن طرف مقابل خود را بررسی کند. اگر طرف مقابل پاسخی ندهد، اتصال قطع می‌شود. این ویژگی اندکی مناقشه‌برانگیز شده است زیرا اولاً سربار تحمیل می‌کند و ثانیاً ممکن است به دلیل وقفه موقت در شبکه به یک اتصال سالم و صحیح خاتمه داده شود.

آخرین تایمر مورد استفاده در هر اتصال TCP، تایمریست که در وضعیت TIMEED WAIT (شکل ۶-۳۳) در حین بستن یک اتصال، مورد استفاده قرار می‌گیرد. پس از آنکه یک اتصال قطع می‌شود به کمک این تایمر به مدت دو برابر حداکثر طول عمر بسته‌ها، اجازه ایجاد اتصالی با همین شماره پورت را نخواهد داد تا اطمینان حاصل شود که بسته‌های سرگردان (حاصل از اتصال قبل) کاملاً از بین رفته‌اند.

۱۱-۵-۶ TCP و UDP بی‌سیم

از دیدگاه تنوری، پروتکل‌های لایه انتقال باید مستقل از تکنولوژی بکار رفته در لایه شبکه باشند. به ویژه، نباید برای TCP اهمیت داشته باشد که IP بر روی فیبرنوری کار می‌کند یا بر روی کانال‌های رادیویی، ولیکن در عمل این موضوع اهمیت دارد زیرا پیاده‌سازی عملی TCP اغلب بر مبنای مفروضاتی بهینه‌سازی می‌شود که فقط برای شبکه‌های سیمی صادق هستند و برای شبکه‌های بی‌سیم با شکست مواجه می‌گردد. نادیده انگاشتن ویژگیهای انتقال بی‌سیم می‌تواند به نوعی از پیاده‌سازی TCP منتهی شود که از دیدگاه منطقی درست است ولی عملاً کارایی بسیار پایینی دارد.

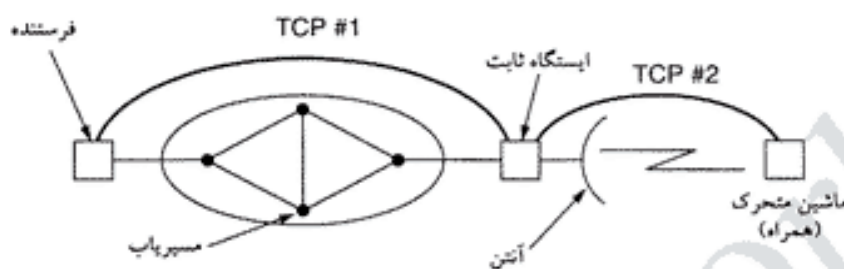
بنیادی‌ترین مشکل در الگوریتم کنترل ازدحام بروز می‌کند. امروزه تقریباً در تمام پیاده‌سازیهایی عملی TCP فرض شده که انقضای مهلت (Timeouts) در اثر ازدحام بوده است و طبعاً خطای کانال انتقال را نادیده می‌گیرند. در نتیجه وقتی تایمر به صفر می‌رسد و مهلت دریافت Ack منقضی می‌شود، TCP سرعت خود را پایین آورده و با شدت کمتری ارسال می‌کند. ایده‌ای که در پشت این راهکار نهفته است کاهش بار شبکه و کم کردن ازدحام می‌باشد.

متأسفانه لینک‌های ارتباطی بی‌سیم بسیار غیر قابل اعتماد و توأم با خطا هستند و همیشه برخی از بسته‌ها را از بین می‌برند. در اینجا راهکار مناسب در مواجهه با بسته‌های نابود شده آنست که در اسرع وقت از نو ارسال شوند. کاهش سرعت ارسال، وضع را به مراتب بدتر می‌کند. اگر مثلاً بیست درصد از کل بسته‌ها از بین بروند، وقتی فرستنده ۱۰۰ بسته در ثانیه ارسال می‌کند درصد مفید آن ۸۰ بسته در ثانیه خواهد بود. در چنین حالتی کاهش نرخ ارسال بسته به ۵۰ بسته در ثانیه، ظرفیت مفید خروجی را به ۴۰ بسته در ثانیه کاهش خواهد داد.

در نتیجه وقتی بسته‌ای در یک شبکه سیمی از دست می‌رود باید سرعت ارسال فرستنده کاهش یابد ولیکن وقتی همین اتفاق در یک شبکه بی‌سیم می‌افتد نه تنها فرستنده نباید سرعت خود را پایین بیاورد بلکه باید تلاش بیشتری در ارسال سریع و مجدد آنها داشته باشد. وقتی که فرستنده از ماهیت شبکه خود آگاه نیست تصمیم‌گیری درست دشوار است.

بسیار اتفاق می‌افتد که مسیر بین فرستنده و گیرنده ناهمگون است یعنی ممکن است ۱۰۰۰ کیلومتر از مسیر در شبکه‌ای سیمی و ۱ کیلومتر آخر بی‌سیم باشد. در چنین حالتی تصمیم‌گیری در خصوص مهلت دریافت Ack به مراتب دشوارتر است زیرا محل بروز مشکل اهمیت دارد. [اگر بسته‌ای در شبکه سیمی از بین رفته باشد ناشی از

مشکل ازدحام و اگر در شبکه بی سیم خراب شده باشد ناشی از خطای کانال بوده است. [برای حل این مشکل راه حلی توسط Bakne و Badrinath (۱۹۹۵) ارائه شده که «TCP غیرمستقیم» (indirect TCP) نام دارد. در این روش، یک اتصال TCP به نحوی که در شکل ۶-۳۹ دیده می شود به دو اتصال مجزا تقسیم می شود: اتصال اول بین فرستنده و ایستگاه ثابت (Base Station) برقرار می شود. اتصال دوم بین ایستگاه ثابت و گیرنده شکل می گیرد. ایستگاه ثابت به سادگی بسته ها را در هر دو جهت، بین این دو اتصال کپی می کند. [به عبارتی از اتصال اول دریافت و بر روی اتصال دوم ارسال می کند و بالعکس]



شکل ۶-۳۹. تقسیم یک اتصال TCP به دو اتصال مجزا.

مزیت این روش آن است که هر دو اتصال در شبکه ای همگن شکل می گیرند. انقضای مهلت در اتصال اول می تواند از سرعت ارسال بکاهد ولیکن بروز همین مشکل در اتصال دوم می تواند به افزایش سرعت فرستنده بینجامد. پارامترهای دیگر نیز براساس محیط هر یک از این دو اتصال به صورت مستقل و پویا تنظیم می شوند. اشکال این روش نیز آن است که مفهوم TCP را نقض می کند. از آنجایی که هر یک از بخشهای یک اتصال خودش یک اتصال کامل است فلذا وقتی دریافت داده ها به فرستنده اعلام می شود به معنای آن نیست که گیرنده حقیقی آنها را دریافت کرده است بلکه به معنای دریافت آنها توسط ایستگاه ثابت می باشد.

راه حل دیگری نیز توسط Balakrishnan و همکاران او پیشنهاد شده که مفهوم TCP را نقض نمی کند. (۱۹۹۵) روش آنها مستلزم ایجاد تغییرات کوچکی در کد اجرایی لایه شبکه در ایستگاه ثابت است. یکی از تغییرات، افزودن یک «عامل تحقیق و تفحص» (Snooping Agent) به لایه شبکه از ایستگاه ثابت می باشد؛ این عامل تمام قطعات TCP را که به سوی یک ماشین متحرک بی سیم می روند یا بسته های حاوی Ack بازگشتی از آنها تشخیص داده و موقتاً در حافظه نهان (cache) خود ذخیره می نماید. اگر «عامل تحقیق و تفحص» متوجه شود که قطعه ای برای ماشین متحرک ارسال شده ولی Ack آن در مدت زمان معقول و کوتاهی برنگشته، آن قطعه را بدون اطلاع دادن به فرستنده آن از نو ارسال می کند. همچنین وقتی یک Ack تکراری از ماشین متحرک می بیند متوجه می شود که این ماشین چیزی را از دست داده است. Ack های تکراری نیز در «عامل تحقیق و تفحص» حذف می شوند تا ماشین مبدا، دریافت آنها را اشتباهاً بمعنای بروز ازدحام تعبیر نکند. در این روش سعی می شود ناهمگونی شبکه ها به نحو زیرکانه ای از چشم طرفین پنهان بماند.

یک اشکال در این روش نامرئی آنست که اگر لینک بی سیم بسیار پر خطا باشد و درصد بالایی از بسته ها را نابود کند، ممکن است ماشین مبدا به دلیل عدم دریافت Ack در مهلت مقرر، الگوریتم کنترل ازدحام خود را فراخوانی و اعمال نماید. در روش «TCP غیرمستقیم» الگوریتم کنترل ازدحام هرگز شروع نخواهد شد مگر آنکه واقعاً در بخش سیمی شبکه ازدحام رخ داده باشد.

در مقاله Balakrishnan و همکاران او، راه حلی نیز برای مشکل از بین رفتن بسته های ماشین متحرک (بی سیم) ارائه شده است: وقتی ایستگاه ثابت متوجه می شود که در بین داده های ارسالی از ماشین متحرک یک

قطعه جا افتاده وجود دارد، با استفاده از یک گزینه جدید در فیلد Option از قطعه TCP، تقاضایی را تولید کرده و جهت دریافت این قطعه جا افتاده برای ماشین متحرک می فرستد. به کمک این اصلاحیه، لینک بی سیم در دو جهت قابل اعتمادتر می شود بدون آن که مبداء چیزی در این خصوص بداند و بی آن که مفهوم واقعی TCP نقض شود. اگرچه UDP از مشکلاتی که TCP با آن مواجه است، رنج نمی برد ولی UDP نیز در محیط های بی سیم مشکلات خاص خود را دارد. مشکل عمده آن است که برنامه های کاربردی استفاده کننده از UDP انتظار قابلیت اعتماد بالا دارند. اگرچه UDP تحویل داده ها را تضمین نکرده ولیکن انتظار می رود که حتی الامکان درست عمل کند. [یعنی درصد بسیار ناچیزی از داده ها از بین برود.] در محیط بی سیم، UDP بسیار نامطمئن و پرخطا عمل خواهد کرد. اگر آن دسته از برنامه های کاربردی که می توانند پیام های UDP گم شده را تشخیص داده و بازیابی کنند، از محیطی که در آن احتمال از بین رفتن بسته ها ناچیز است به محیطی که بخشی از داده ها بطور دائم از دست می روند، منتقل شوند منجر به کاهش بحرانی کارایی آنها خواهد شد.

ارتباط بی سیم، جدای از مسئله کارایی، در زمینه های دیگری نیز تأثیر منفی دارد. به عنوان مثال چگونگی پیدا کردن یک چاپگر محلی و برقراری اتصال با آن، خود یک مسئله است. یا مشکل دیگر آنست که چگونه می توان به صفحه وب محلی در سلول جاری دسترسی پیدا کرد در حالی که نام آن را نمی دانیم. معمولاً طراحان صفحه وب [در محیط های شبکه محلی] فرض را بر آن می گذارند که پهنای باند موجود فراوان است. حال اگر در هر صفحه وب لوگویی قرار داده شود که مراجعه به آن در یک شبکه بی سیم مثلاً ده ثانیه طول می کشد، خوشایند کاربر نخواهد بود.

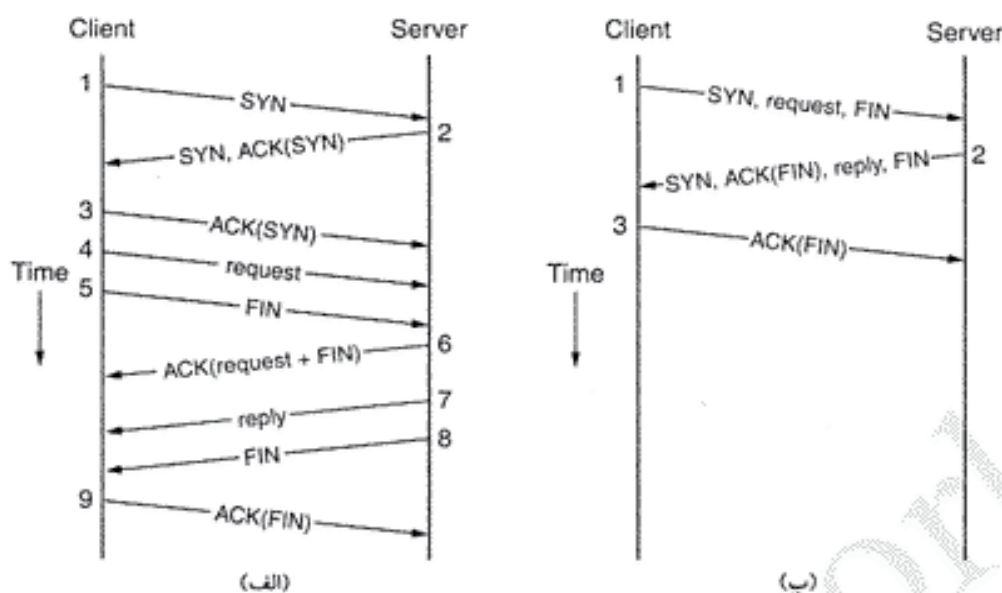
شبکه های بی سیم در حال رواج هستند و مشکل اجرای TCP بر روی اینگونه شبکه ها روز به روز حادتر می شود. در این خصوص کارهای دیگری نیز انجام شده که گزارش آنها در مراجع زیر در دسترس است:

Barakat et al., 2000; Ghani and Dixit, 1999; Huston, 2001; and Xylomenos et al., 2001)

۱۲-۵-۶ TCP تراکنشی (Transactional TCP)

قبلاً در این فصل به موضوع فراخوانی پروسیجرهای راه دور (RPC) به عنوان روشی برای پیاده سازی سیستم های سرویس دهنده/مشری، نگاهی انداختیم. اگر «تقاضا» (request) و «پاسخ» (Reply) آنقدر کوچک باشند که در یک بسته واحد جا بگیرند و تقاضاهای متوالی، مستقل از هم باشند، بسادگی می توان از UDP بهره گرفت. ولیکن اگر این شرایط برقرار نباشد، استفاده از UDP چندان مفید نخواهد بود. به عنوان مثال اگر پاسخ به یک تقاضا، داده ای بسیار بزرگ باشد، قطعات داده باید شماره گذاری شده و مکانیزمی جهت ارسال مجدد قطعات از بین رفته، ابداع و پیاده سازی شود. در نتیجه، برنامه کاربردی ملزم به پیاده سازی عملیات TCP خواهد بود. روشن است که این رویکرد جالب نیست ولی استفاده از TCP به جای UDP نیز فایده ای ندارد. مشکل استفاده از TCP، کارایی آن است. اگر از RPC بر روی TCP استفاده شده باشد، توالی بسته هایی که بین سرویس دهنده و مشری (برای انجام یک فراخوانی راه دور) مبادله می شوند در شکل ۶-۴۰ الف نشان داده شده است. در بهترین حالت باید ۹ بسته مبادله شود. این نه بسته عبارتند از:

۱. برنامه مشری، یک بسته SYN به منظور برقراری اتصال برای سرویس دهنده ارسال می کند.
۲. سرویس دهنده در پاسخ به دریافت بسته SYN، یک بسته ACK بازپس می فرستد.
۳. برنامه مشری فرآیند دست تکانی سه مرحله ای را تکمیل می نماید.
۴. برنامه مشری، تقاضای اصلی خود را ارسال می دارد.
۵. برنامه مشری، بسته FIN را جهت اعلام خاتمه کار خود می فرستد.



شکل ۶-۴۰. (الف) عمل RPC بکمک TCP معمولی. (ب) عمل RPC بکمک T/TCP.

۶. سرویس دهنده دریافت بسته حاوی تقاضا و FIN را تصدیق می کند. (با ارسال ACK)

۷. سرویس دهنده پاسخ مورد نظر مشتری را باز می گرداند.

۸. سرویس دهنده یک بسته FIN می فرستد تا خاتمه کار خود را اعلام نماید.

۹. مشتری، دریافت بسته FIN متعلق به سرویس دهنده را تصدیق می کند.

دقت کنید که این ۹ مرحله در بهترین حالت ممکن اتفاق می افتد. در بدترین حالت، تقاضای مشتری و بسته FIN بطور جداگانه اعلام وصول می شود و به همین نحو پاسخ سرویس دهنده و بسته FIN آن بطور مجزا ارسال می گردد.

سوآلی که مطرح می شود آنست که آیا راهی وجود دارد که کارایی و سرعت RPC مبتنی بر UDP (دقیقاً با دو پیام) و قابلیت اعتماد TCP را با هم تلفیق کرد. پاسخ به این سؤال این است: تقریباً! این کار را می توان با یک گونه آزمایشی از TCP که اصطلاحاً T/TCP (Transactional TCP) نامیده می شود انجام داد. شرح T/TCP در RFC 1379 و RFC 1644 آمده است.

ایده اصلی در T/TCP آن است که روند استاندارد برقراری یک اتصال را به گونه ای اصلاح کنیم که بتوان در حین برقراری اتصال، داده ها را نیز ارسال کرد. پروتکل T/TCP در شکل ۶-۴۰-ب نشان داده شده است. اولین بسته ارسالی توسط مشتری، حاوی بیت SYN، اصل تقاضا و بیت FIN است. در حقیقت می گوید: «مایل به برقراری یک اتصال هستم، این هم داده های من، کارم نیز به اتمام رسید!»

وقتی سرویس دهنده، این تقاضا را دریافت می کند، پاسخ آن را پیدا یا محاسبه کرده و سپس چگونگی ارسال آن را انتخاب می نماید: اگر پاسخ در یک بسته واحد جا بگیرد، بسته ای را که در شکل ۶-۴۰-ب نشان داده شده، ارسال می دارد. در حقیقت اعلام می کند: «دریافت FIN شما را تأیید می کنم، این هم پاسخ شما، کار من هم تمام شد!» در اینجا، مشتری دریافت FIN سرویس دهنده را تصدیق کرده و پروتکل با مبادله سه پیام خاتمه می یابد. ولیکن اگر پاسخ بزرگتر از یک بسته باشد، سرویس دهنده می تواند بیت FIN را فعال نکند و چندین بسته بفرستد. سپس در آخرین بسته FIN را فعال و ارسال نماید.

اشاره به این نکته ارزشمند است که T/TCP تنها نسخه بهبود یافته TCP برای عملیات تراکنشی نیست.

پیشنهاد دیگر SCTP (Stream Control Transmission Protocol) است. از ویژگیهای آن می توان به موارد ذیل اشاره کرد: (۱) حفظ مرز پیام (۲) پشتیبانی از چندین حالت تحویل (مثل تحویل نامرتب) (۳) حمایت از Multihoming (ماشینهای مقصد پشتیبان) (۴) اعلام وصول بسته ها به صورت انتخابی (Selective Repeat). (Stewart and Metz, 2001) با این حال وقتی یک نفر پیشنهاد ایجاد تغییر در پروتکلی را که برای سالها به خوبی کار کرده، می دهد یک مناقشه عظیم بین طرفداران این دو نظریه در می گیرد: «افرادی که معتقدند باید برای پاسخ به نیاز کاربران ویژگیهای بیشتری را به پروتکل افزود» و «افرادی که معتقدند تا وقتی پروتکل به بن بست نخورده نباید آن را اصلاح کرد»

۶-۶ مسائل مرتبط با کارایی^۱

مسائل مرتبط با کارایی در شبکه های کامپیوتری از اهمیت ویژه ای برخوردارند. وقتی صدها یا هزاران کامپیوتر به هم متصل می شوند، تعامل پیچیده آنها با تبعات غیرمنتظره ای همراه است. اکثراً این پیچیدگی منجر به کارایی بسیار ضعیف شبکه می شود و هیچکس هم نمی داند علت چیست. در بخشهای آتی به بررسی مواردی خواهیم پرداخت که به کارایی شبکه مربوط می شود تا ببینیم که چه مشکلاتی وجود دارد و با آنها چه باید کرد.

متأسفانه، تشخیص کارایی شبکه بیشتر یک هنر است تا یک علم! تئوری کمی وجود دارد که بتوان در عمل از آن بهره گرفت. بهترین کاری که می توانیم انجام بدهیم آنست که برخی از قواعد تجربی با پشتوانه محکم و مثالهایی از دنیای واقعی را معرفی کنیم. تعمداً این مبحث را تا اینجا به تعویق انداختیم تا پس از مطالعه TCP بتوانیم از آن به عنوان مثال استفاده نماییم.

لایه انتقال تنها نقطه بروز مشکلات مرتبط با کارایی نیست. در فصل قبلی برخی از این مشکلات را در لایه شبکه بررسی کردیم. علیرغم این، لایه شبکه بیشتر درگیر مسائل مسیریابی و کنترل ازدحام است. موارد مرتبط با سیستمهای نهایی به لایه انتقال بر می گردد لذا این فصل جایگاه مناسبی برای بررسی مشکلات مرتبط با کارایی است. در پنج بخش بعدی به پنج جنبه از کارایی شبکه نگاهی خواهیم انداخت:

۱. مشکلات کارایی
۲. اندازه گیری کارایی شبکه
۳. طراحی سیستم برای رسیدن به کارایی بهتر
۴. پردازش سریع TPDU
۵. پروتکلهایی برای شبکه های کارآمد در آینده

برای هر واحد اطلاعات که توسط لایه انتقال مبادله می شود، نیاز به یک اسم عمومی داریم. واژه بکار رفته در TCP یعنی «قطعه» (Segment) همراه کننده است و در هیچ جای دیگر به غیر از پروتکل TCP از این واژه استفاده نشده است. واژه های بکار رفته در ATM (مثل CS-PDU، SAR-PDU و CPCS-PDU) نیز خاص شبکه ATM هستند. واژه «بسته» در لایه شبکه و واژه «پیام» در لایه کاربرد بکار می روند. به دلیل فقدان یک استاندارد و اجماع عمومی، ما از همان واژه TPDU استفاده خواهیم کرد. وقتی بخواهیم به بسته و TPDU درون آن بطور همزمان اشاره کنیم واژه کلی «بسته» را بکار خواهیم برد مثلاً وقتی می گوییم: «CPU باید آنقدر سریع باشد که بتواند بسته های ورودی را به صورت بی درنگ پردازش کند» منظورمان هم بسته لایه شبکه و هم TPDU جاسازی شده در آن می باشد.

۱۶-۶ مشکلات کارایی در شبکه های کامپیوتری

منشاء برخی از مشکلات کارایی مثل ازدحام به استفاده بیش از حد از منابع موجود شبکه برمی گردد. اگر به ناگاه ترافیکی بیش از توان و ظرفیت مسیریاب، به آن تحویل داده شود ازدحام پدید آمده و موجب کاهش کارایی می شود. در فصل قبلی مفصلاً به موضوع ازدحام پرداختیم.

همچنین وقتی که منابع شبکه از لحاظ ساختاری توازن و تعادل نداشته باشند کارایی کاهش خواهد یافت. به عنوان مثال اگر یک خط ارتباطی یک گیگابیتی به یک PC معمولی متصل شده باشد پردازنده ضعیف این کامپیوتر قادر نخواهد بود که بسته های ورودی را به سرعت پردازش کند و برخی از آنها از دست می رود. بسته های از دست رفته نهایتاً از نو ارسال می شوند و طبعاً تأخیر افزایش می یابد، پهنای باند هدر می رود و در مجموع کارایی شبکه افت می کند.

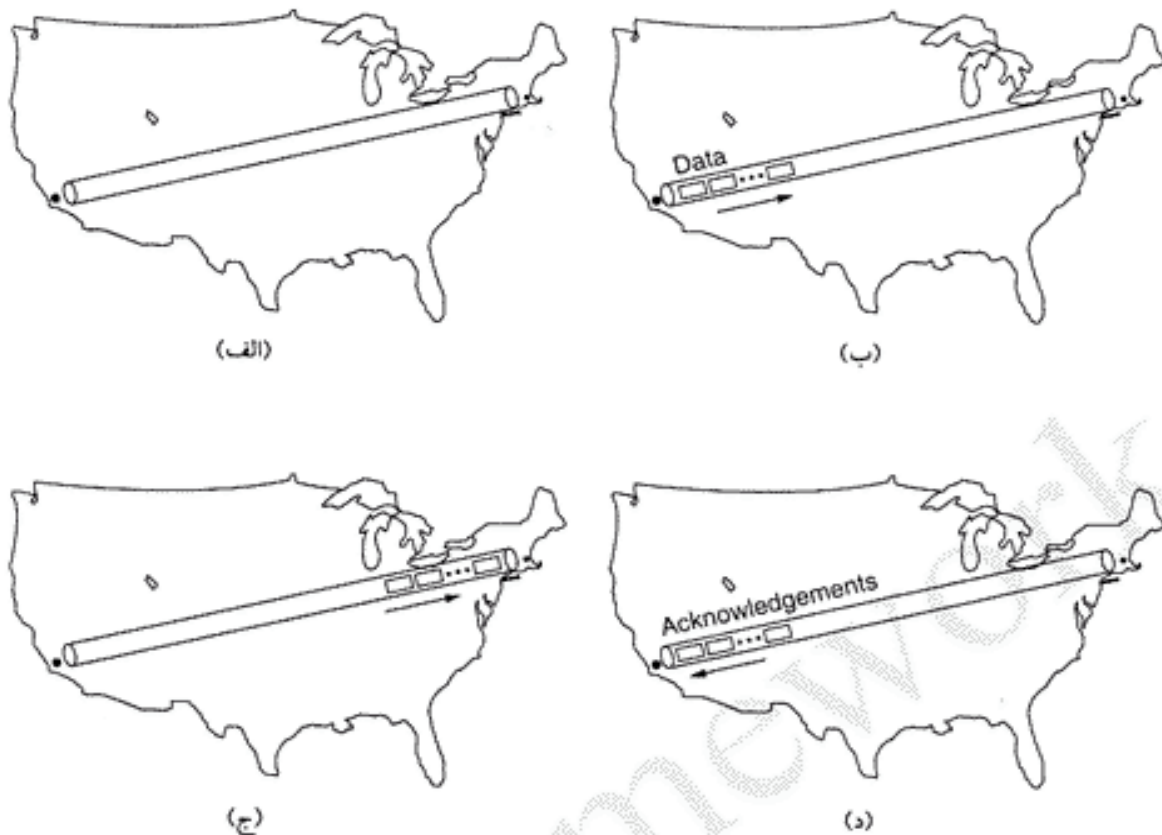
بار بیش از حد (Overload) نیز ممکن است بطور لحظه ای اتفاق افتاده و کارایی شبکه را کاهش بدهد. به عنوان مثال اگر یک TPDU حاوی یک پارامتر اشتباه باشد (مثلاً شماره پورت مقصد آن صحیح نباشد)، در بسیاری حالات، گیرنده یک گزارش خطا باز می گرداند. حال در نظر بگیرید که اگر یک TPDU حاوی پارامتر اشتباه به صورت پخش فراگیر (Broadcast) برای ۱۰۰۰۰ ماشین ارسال شود چه اتفاقی می افتد: ممکن است همه آنها یک پیغام خطا برگردانند! این اتفاق منجر به بروز «طوفان پخش فراگیر» (Broadcast Storm) شده و می تواند شبکه را فلج کند. UDP از این مشکل رنج می برد مگر آن که این پروتکل به نحوی تغییر کند که در مواجهه با TPDU های اشتباه که به صورت فراگیر پخش شده اند پیغام خطا برنگرداند.

مثال دوم از تحمیل بار بیش از اندازه به شبکه (بطور همزمان)، پس از قطع برق اتفاق می افتد. وقتی برق مجدداً بر می گردد همه ماشینها بطور همزمان از طریق ROM خود سعی می کنند از نو راه اندازی شوند. عموماً در روال راه اندازی، هر ماشین شبکه در ابتدا به برخی از سرویس دهنده ها (مثل DHCP) مراجعه می کند تا اول هویت خود را بشناسد؛ پس از آن برای دریافت سیستم عامل به سرویس دهنده فایل مراجعه می نماید. اگر صدها ماشین به طور همزمان این کار را انجام بدهند، احتمال دارد سرویس دهنده در زیر چنین باری از کار بیفتد.

حتی اگر بار، بیش از حد و همزمان اتفاق نیفتد و منابع شبکه نیز بقدر کفایت در اختیار باشد باز هم ممکن است به دلیل عدم تنظیم صحیح سیستم، کارایی شبکه کاهش یابد. به عنوان مثال اگر ماشینی دارای پردازنده بسیار سریع و حجم انبوه حافظه باشد ولیکن به قدر کافی برای فضای بافر حافظه اختصاص داده نشود ممکن است کمبود بافر به از دست رفتن TPDU بینجامد. همچنین اگر الگوریتم زمان بندی پروسه ها، اولویت بالایی را به پردازش TPDU ها ندهد باز هم ممکن است برخی از بسته ها از بین بروند.

مشکل دیگر، تنظیم صحیح مهلت تایمرهاست. وقتی یک TPDU ارسال می گردد برای آن که بتوان از گم شدن آن آگاه شد یک تایمر برای آن تنظیم می شود. اگر مهلت این تایمر کوتاه در نظر گرفته شود، ارسالهای مجدد بیهوده اتفاق می افتد و پهنای باند کانال را هدر می دهد. اگر هم این زمان طولانی در نظر گرفته شود در هنگام از دست رفتن یک TPDU تأخیری مورد تحمیل می شود. پارامتر دیگری که باید تنظیم شود مهلت تایمری است که وقتی داده ای دریافت می شود و نمی خواهیم برای آن ACK مجزا بفرستیم باید به آن مدت صبر کرد تا داده ای جهت ارسال تولید شود و بتوان ACK را به آنها ضمیمه نمود. اگر این زمان انتظار زیاد در نظر گرفته شود، ارسالهای تکراری و بیهوده اتفاق می افتد.

شبکه های گیگابیتی مشکلات جدیدی در کارایی شبکه به همراه آورده اند. به عنوان مثال در نظر بگیرید که یک توده ۶۴ کیلوبایتی داده از سن دیه گو به بوستون ارسال می شود تا در بافر ۶۴ کیلوبایتی گیرنده قرار بگیرد. فرض کنید تأخیر انتشار این لینک (با احتساب سرعت نور در فیبر) معادل ۲۰ میلی ثانیه باشد. در لحظه $t=0$ خط خالی



شکل ۴۱-۶. وضعیت ارسال یک مگابایت داده از سن دیه گو به بوستون. (الف) در لحظه $t=0$. (ب) پس از گذشت ۵۰۰ میکروثانیه. (ج) پس از گذشت ۲۰ میلی ثانیه. (د) پس از گذشت ۴۰ میلی ثانیه.

است و به نحوی که در شکل ۴۱-۶ الف می بینید داده ای بر روی خط ارسال نشده است. حدود ۵۰۰ میکروثانیه بعد، تمام TPDUs حاوی ۶۴ کیلوبایت داده بر روی فیبر قرار گرفته اند و ارسال آنها خاتمه یافته است. حالا اولین TPDUs ارسالی جایی در نزدیکی براولی (Brawley) است! (شکل ۴۱-۶ ب) با این حال فرستنده باید ارسال خود را تا اعلام مجدد اندازه پنجره متوقف کند چرا که متناسب با اندازه پنجره گیرنده، داده ارسال شده است.

پس از ۲۰ میلی ثانیه، اولین TPDUs به بوستون می رسد و طبق شکل ۴۱-۶ ج اعلام وصول آنها آغاز می شود. ۴۰ میلی ثانیه پس از شروع، اولین پیغام ACK به فرستنده باز می گردد و دومین توده داده را می توان ارسال کرد. از آنجایی که در طی ۴۰ میلی ثانیه، فقط ۵/۰ میلی ثانیه از خط انتقال استفاده شده، کارایی آن حدود ۱/۲۵ درصد است. چنین وضعیتی برای پروتکل های قدیمی که بر روی خطوط گینگابیتی اجرا می شوند، کاملاً طبیعی است.

یکی از کمیت های بسیار مهمی که در هنگام تحلیل کارایی شبکه باید به خاطر داشته باشید «حاصل ضرب پهنای باند در تأخیر» (Bandwidth-Delay Product) است. این کمیت را می توان با ضرب مقدار پهنای باند (برحسب بیت بر ثانیه) در زمان تأخیر رفت و برگشت خط (برحسب ثانیه) محاسبه کرد. این حاصل ضرب، ظرفیت خط لوله بین فرستنده و گیرنده و بالعکس را برحسب بیت تعیین می کند.

به عنوان مثال در شکل ۴۱-۶ حاصل ضرب پهنای باند در تأخیر، معادل ۴۰ میلیون بیت است. یعنی فرستنده مجبور است یک توده ۴۰ میلیون بیتی از داده ها را بفرستد تا اولین پیغام ACK باز گردد. این تعداد بیت کل خط را

در دو جهت پر می‌کند.^۱ این همان دلیلی است که راندمان خط برای ارسال نیم میلیون بیت (۶۴ کیلوبایت) داده به مقدار ۱/۲۵ درصد کاهش می‌یابد زیرا فقط ۱/۲۵ درصد از حجم ۴۰ میلیون بیتی ارسال و متوقف شده است. نتیجه‌ای که می‌توان گرفت آنست که برای کارایی خوب، پنجره گیرنده حداقل باید به بزرگی «حاصلضرب پهنای باند در تأخیر» باشد. (حتی ترجیحاً بیشتر زیرا ممکن است گیرنده نتواند فوراً پاسخ بدهد.) برای خطوط گیگابیتی بین قاره‌ای به حداقل ۵ مگابایت فضای حافظه نیاز است.

اگر کارایی خط برای ارسال یک مگابایت داده اینقدر پایین باشد تصور کنید که برای یک تقاضای کوتاه چند صد بیتی چقدر است! اگر وقتی اولین مشتری منتظر دریافت پاسخ از طرف مقابل است از خط استفاده دیگری نشود، خط یک گیگابیتی هیچ مزیتی بر خط یک مگابیتی بر ثانیه نخواهد داشت؛ فقط گرانتر است.

مشکل دیگر کارایی، که برای کاربردهای حساس به زمان مثل صدا و تصویر رخ می‌دهد، «لرزش» (Jitter) است. کوتاه بودن زمان متوسط انتقال کافی نیست؛ انحراف معیار این زمان نیز باید پایین باشد. رسیدن به زمان متوسط انتقال کم و انحراف معیار پایین مستلزم تلاشهای مهندسی بسیار جدی است.

۲-۶-۶ اندازه گیری کارایی شبکه

وقتی شبکه ضعیف و ناکارآمد عمل می‌کند کاربران آن شروع به شکوه و گلایه کرده و خواهان بهبود کارایی می‌شوند. برای بهبود کارایی، اپراتورها باید اول تعیین کنند که مشکل از کجا منشأ می‌گیرد. بمنظور پیگیری و تشخیص مشکل بوجود آمده، آنها مجبور به انجام پاره‌ای محاسبات و اندازه گیری هستند. در این بخش به موضوع اندازه گیری کارایی شبکه نگاهی می‌اندازیم. مباحث ذیل برگرفته از پژوهشهای Mogul (۱۹۹۳) می‌باشد. روال بهبود کارایی شبکه شامل مراحل زیر است:

۱. پارامترهای شبکه مورد نظر و کارایی آن را اندازه گیری کنید.
 ۲. سعی کنید وضعیت فعلی شبکه را ارزیابی نمایید.
 ۳. یکی از پارامترها را تغییر دهید.
- این مراحل آنقدر تکرار می‌شوند تا کارایی شبکه به حد مطلوب برسد یا روشن شود که آخرین حد کارایی همین است.
- اندازه گیریها را می‌توان به روشهای متعدد و در موقیعتهای متفاوت انجام داد (به صورت فیزیکی یا در پشته پروتکل). یکی از اساسی ترین نوع اندازه گیری آن است که تایمری را در ابتدای انجام یک عمل روشن کرده و بررسی کنیم که آن عمل چقدر طول می‌کشد. به عنوان مثال دانستن زمانی که طول می‌کشد تا پس از ارسال TPDU، دریافت آن تأیید شود، بسیار اهمیت دارد. اندازه گیریهای دیگری را نیز می‌توان به کمک شمارنده‌ها انجام داد (مثلاً تعداد TPDUهایی که از دست رفته‌اند). یکی دیگر از پارامترهایی که دانستن آن اهمیت دارد تعداد بایتهایی است که در یک مدت زمان معین پردازش می‌شوند.
- اندازه گیری کارایی و پارامترهای شبکه پیچیدگیهای خاص خود را دارد. در زیر به برخی از آنها اشاره می‌کنیم. در هر گونه تلاش سیستماتیک برای اندازه گیری کارایی شبکه باید موارد ذیل را مد نظر داشته باشید:

مطمئن شوید که تعداد نمونه‌های آماری به قدر کافی زیاد است

فقط به اندازه گیری زمان ارسال یک TPDU اکتفا نکنید بلکه اندازه گیریهای خود را مثلاً یک میلیون بار تکرار کرده و میانگین آن را بدست بیاورید. در اختیار داشتن تعداد بسیار زیاد نمونه‌های آماری، عدم قطعیت مقدار میانگین و

۱. به عبارتی پس از ارسال ۴۰ میلیون بیت بر روی یک خط 1Gbps تازه اولین پیغام ACK باز خواهد گشت. فرستنده باید قادر به ارسال و بافر کردن چنین حجمی از داده باشد. —

انحراف معیار اندازه گیری شده را کاهش خواهد داد. میزان «عدم قطعیت» به کمک فرمولهای استاندارد آماری قابل محاسبه است.

دقت داشته باشید که نمونه های آماری به صورت جامع انتخاب شده اند

ایده آل آنست که مثلاً یک میلیون بار اندازه گیری و نمونه برداری، در زمانهای مختلف یک روز و در طی هفته تکرار شود تا تأثیر بار سیستم در زمانهای متفاوت بر روی کمیت اندازه گیری شده، مشخص گردد. به عنوان مثال اندازه گیری ازدحام در لحظه ای که ازدحام وجود ندارد، چندان مفید نیست. گاهی اوقات نتایج بدست آمده در بدو امر غیرطبیعی به نظر می رسند (مثل ازدحام سنگین در خلال ساعت ۱۰ تا ۲ و عدم ازدحام در حول و حوش ظهر وقت ناهار - که کاربران کار خود را رها کرده اند!).

وقتی از ساعت نه چندان دقیق استفاده می کنید مراقب نتیجه اندازه گیریها باشید

ساعت کامپیوترها بدین نحو کار می کنند که در فواصل زمانی معین به یک یا چند شمارنده، یک واحد می افزایند. به عنوان مثال تایمر یک میلی ثانیه ای در هر میلی ثانیه، یک واحد به شمارنده اضافه می کند. استفاده از چنین تایمری برای اندازه گیری رخدادهایی که کمتر از یک میلی ثانیه طول می کشد امکان پذیر است ولی مستلزم دقت زیاد است. (البته بعضی از کامپیوترها ساعت دقیقتری دارند).

برای محاسبه زمان ارسال یک TPDU، بایستی ساعت سیستم (که مثلاً دقت آن میلی ثانیه است) در ابتدای شروع «کد برنامه واحد انتقال» و همچنین به محض خاتمه کار خوانده شود. اگر زمان واقعی ارسال یک TPDU، ۳۰۰ میکروثانیه باشد، اختلاف بین دو زمان خوانده شده یا صفر است یا یک، که هر دوی آنها اشتباه است. ولیکن اگر اندازه گیری زمان برای ارسال یک میلیون TPDU انجام گیرد و زمان کل بر عدد یک میلیون تقسیم شود، میانگین زمان دقتی حدود یک میکروثانیه خواهد داشت.

مطمئن باشید که در خلال آزمایشات هیچ چیز پیش بینی نشده ای وجود ندارد

انجام اندازه گیری بر روی سیستم یک دانشگاه در روزی که مثلاً چندین پروژه آزمایشگاهی عظیم تحت آزمایش است می تواند نتیجه کاملاً متفاوتی با اندازه گیریهای روز بعد داشته باشد. به دلیل مشابه، اگر چندین پژوهشگر در حین آزمایشهای آماری شما تصمیم به اجرای یک کنفرانس ویدیویی بر روی شبکه بگیرند ممکن است شما را با نتایج همراه کننده ای مواجه سازند. بهترین کار آن است که بررسیها را بر روی یک سیستم بیکار انجام بدهید و باری را که می خواهید بر روی شبکه یا سیستم بگذارید خودتان ایجاد کنید. البته این راهکار هم ممکن است معضلات خود را داشته باشد. شاید با خود بیندیشید که هیچکس در ساعت ۳ بامداد از شبکه استفاده نمی کند در حالی که امکان دارد در همین زمان برنامه هایی که بطور خودکار نسخه ای پشتیبان از دیسک را بر روی نوار مغناطیسی منتقل می کنند در حال کار باشند. مضاف بر این ممکن است ترافیک سنگینی از وبسایت شما به نقطه ای از جهان (که ساعت محلی آنها متفاوت است) در حال انتقال باشد.

فرآیند ذخیره در حافظه نهان (Caching) می تواند صحت نتایج اندازه گیری را به خطر بیندازد

روش بدیهی برای اندازه گیری زمان انتقال فایل آن است که یک فایل بزرگ را باز کنید، کل آن بخوانید و نهایتاً آنرا ببندید و سپس زمان کل را محاسبه نمایید. برای بالا بردن دقت محاسبه، باید این کار چندین بار تکرار شده و میانگین گرفته شود. مشکل اینجاست که امکان دارد، سیستم این فایل را در حافظه نهان ذخیره کرده باشد؛ بنابراین فقط اولین اندازه گیری با ترافیک شبکه سر و کار دارد و بقیه اندازه گیریها کاملاً غلط هستند چرا که از بافر محلی خوانده می شوند. نتیجه ای که از چنین آزمایشی بدست می آید به کل فاقد اعتبار است. (مگر آن که خواسته باشید

کارآیی حافظه نهان (cache - را اندازه گیری کنید)

اغلب می توانید با سرریز کردن بافر آن را دور بزنید. به عنوان مثال اگر فضای حافظه نهان ۱۰ مگابایت باشد در حلقه آزمایش می توانید دو فایل ده مگابایتی را باز کنید؛ سپس آنها را پشت سرهم بخوانید و نهایتاً آنها را ببندید تا در هر بار که یکی از فایلها آزمایش می شود، میزان استفاده از حافظه نهان (cache) به صفر برسد. با این وجود توصیه می شود با احتیاط این کار را انجام دهید مگر آن که کاملاً با الگوریتم caching خود آشنا باشید.

بافرسازی داده ها نیز نتیجه مشابهی دارد. مشهور است که یکی از برنامه های رایج اندازه گیری کارآیی TCP/IP، کارآیی UDP را بیشتر از ظرفیت ممکن خط، گزارش می کند! چرا این اتفاق افتاده است؟ دلیل این است که پس از فراخوانی UDP به محض آن که پیام ارسالی تحویل هسته سیستم عامل می شود، آن پیام در انتهای صف ارسال قرار گرفته و کنترل اجرا به برنامه فراخواننده بر می گردد. اگر فضای بافر به مقدار کافی وجود داشته باشد حتی هزار بار فراخوانی UDP به معنای ارسال آنها نخواهد بود. بیشتر آنها هنوز در اختیار هسته سیستم عامل هستند ولی ابزار محاسبه کارآیی فکر می کند که واقعاً ارسال شده اند.

دقیقاً بدانید که چه چیزی را اندازه گیری می کنید

وقتی زمان خوانده شدن یک فایل راه دور را اندازه می گیرید، اندازه گیری شما به شبکه، سیستم عامل دو طرف (سرویس دهنده و مشتری)، سخت افزار واسط شبکه، برنامه های راه انداز آنها (drivers) و عوامل دیگر بستگی دارد. در صورتی که اندازه گیریها بدقت انجام شده باشد، زمان انتقال فایل براساس پیکربندی فعلی بدست می آید. اگر هدف نهایی شما تنظیم همین پیکربندی است این اندازه گیری مفید خواهد بود ولیکن اگر همین اندازه گیری را بر روی سه سیستم انجام می دهید تا براساس آن یک کارت شبکه مناسب برای خرید انتخاب کنید، نتایج بدست آمده می تواند کاملاً فاقد اعتبار باشد چرا که مثلاً برنامه راه انداز کارت شبکه نامناسب بوده و فقط از ده درصد کارآیی کارت شبکه، بهره برداری کرده است!

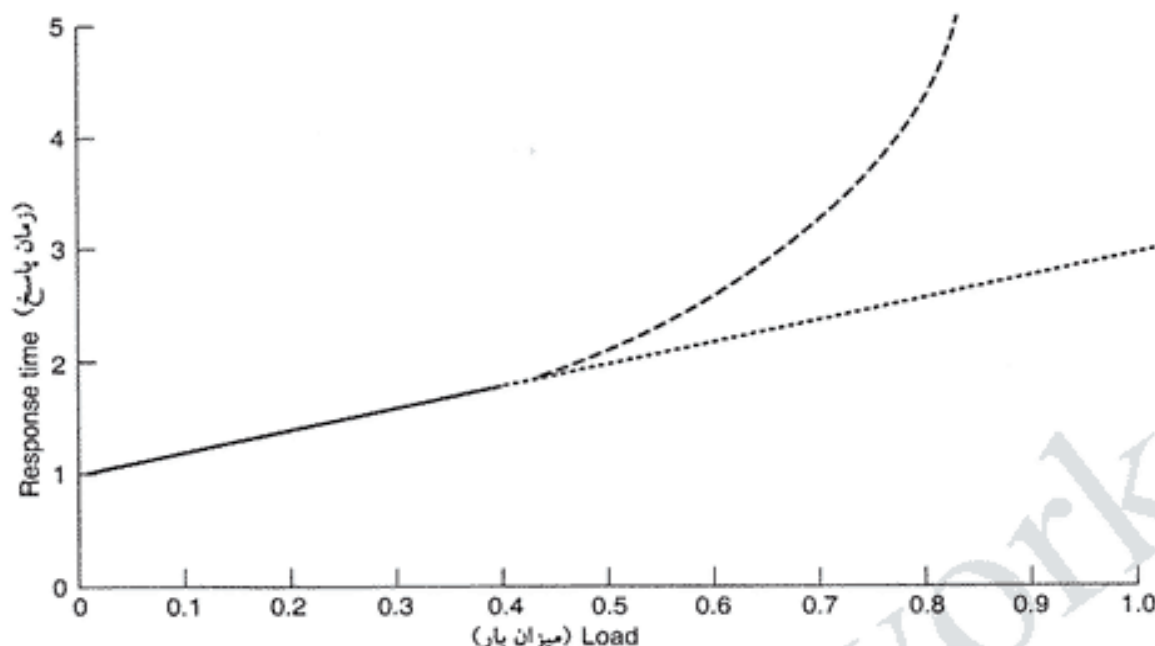
در خصوص نتایج حاصل از برون یابی دقت (Extrapolating) داشته باشید

فرض کنید که اندازه گیری پارامتری مثل زمان پاسخ را با ایجاد بار شبیه سازی شده از صفر (بیکار) تا $0.4/40$ درصد کل ظرفیت) اندازه گیری کرده اید و یک نمودار شبیه به خط توپر در شکل ۶-۴۲ بدست آورده اید. براساس برون یابی خطی، نتیجه ای اشتباه مثل خط نقطه چین بدست می آید. در حالی که نتایج آزمایشاتی که با صاف سرو کار دارند از عاملی به صورت $1/(1-p)$ تأثیر می پذیرند که در آن p ، پارامتر بار (Load) است لذا مقادیر واقعی شبیه به نمودار خط چین هستند که شیب بیشتری نسبت به نمودار خطی دارد.

۶-۳ طراحی سیستم برای کارآیی بهتر

اندازه گیری و بهره برداری از نتایج محاسبات اغلب می تواند کارآیی را تا حد قابل توجهی افزایش بدهد ولیکن نمی تواند از همان ابتدا جایگزین طراحی خوب شود. کارآیی یک شبکه با طراحی ضعیف را فقط تا حدی می توان بهبود داد. پس از آن باید شبکه را از نو طراحی کرد.

در این بخش قواعدی را برای طراحی ارائه می دهیم که مبتنی بر تجارب گسترده از شبکه های متعدد است. این قواعد مرتبط با طراحی سیستم هستند نه طراحی شبکه، زیرا نرم افزار و سیستم عامل، اغلب نقش مهمتری در مقایسه با مسیر یابها و کارتهای واسط شبکه در کارآیی ایفاء می کنند. بیشتر نظریاتی که ارائه می شود برای طراحان شبکه یک دانش پایه عادی و حاصل تجربیاتی است که نسل به نسل منتقل شده و تکامل یافته است. این نظریات برای اولین بار توسط Mogul (۱۹۹۳) به صورت مدون ارائه شد که ما نیز از وی پیروی می کنیم. در این زمینه مرجع (Metcalfe, 1993) نیز برای مطالعه مناسب است.



شکل ۶-۴۲. «زمان پاسخ» بعنوان تابعی از «بار».

قانون شماره ۱: سرعت CPU از سرعت شبکه مهمتر است

تجارب طولانی مدت نشان داده که تقریباً در تمام شبکه ها، سر بار سیستم عامل و پشته پروتکلی از زمان واقعی استفاده از کانال بیشتر است.^۱ به عنوان مثال از دیدگاه تئوری زمان لازم برای یک عمل RPC (یعنی فراخوانی پروسیجر از راه دور) بر روی شبکه اترنت حداقل ۱۰۲ میکروثانیه است. (شامل انتقال یک تقاضای ۶۴ بیتی و پاسخ ۶۴ بیتی). در عمل غلبه کردن بر سر بار تحمیلی توسط نرم افزار و رساندن زمان RPC به همین حدود موفقیت بزرگی محسوب می شود.

به دلیل مشابه، بزرگترین مشکل کار کردن در سرعت 1Gbps، استخراج بیتها از بافر کاربر و انتقال آن بر روی فیبر نوری و همچنین در اختیار داشتن پردازنده ای است که در سمت مقابل بتواند با سرعت کافی آنها را دریافت کند. کوتاه سخن آن که اگر سرعت CPU را دو برابر کنید اغلب توان خروجی نیز به نزدیکی دو برابر می رسد. دو برابر کردن ظرفیت شبکه معمولاً نتیجه ای در پی ندارد چرا که گلوگاه اصلی در ماشینهای میزبان است.

قانون شماره ۲: کاهش تعداد بسته ها برای کاهش سر بار نرم افزار

پردازش یک TPDU به میزان معینی سر بار به ازای هر TPDU (مثلاً برای پردازش سرآیند) و همچنین به میزان معینی سر بار به ازای هر بایت (مثلاً برای محاسبه کد کشف خطای Checksum) به CPU تحمیل می کند.^۲ وقتی یک میلیون بایت ارسال می شود سر بار تحمیلی آن به ازای این تعداد بایت ثابت است و به اندازه TPDU ها بستگی ندارد. ولیکن سرباری که بدلیل استفاده از TPDU های ۱۲۸ بیتی تحمیل می شود ۳۲ برابر نسبت به زمانی که از TPDU های ۲ کیلوبایتی استفاده می گردد، بیشتر خواهد بود.

مضاف بر سر بار TPDU در لایه انتقال، در لایه های زیرین هم سر بار قابل توجهی تحمیل می شود. هر بسته دریافتی یک «وقفه» (interrupt) ایجاد می کند. در پردازنده های مدرن مبتنی بر معماری Pipeline، بروز وقفه (۱)

۱. عبارتی تأخیر تحمیلی توسط نرم افزار می تواند بیشتر از تأخیر ناشی از انتقال باشد. -م

۲. $\text{Overhead per TPDU} / \text{Overhead per Byte}$

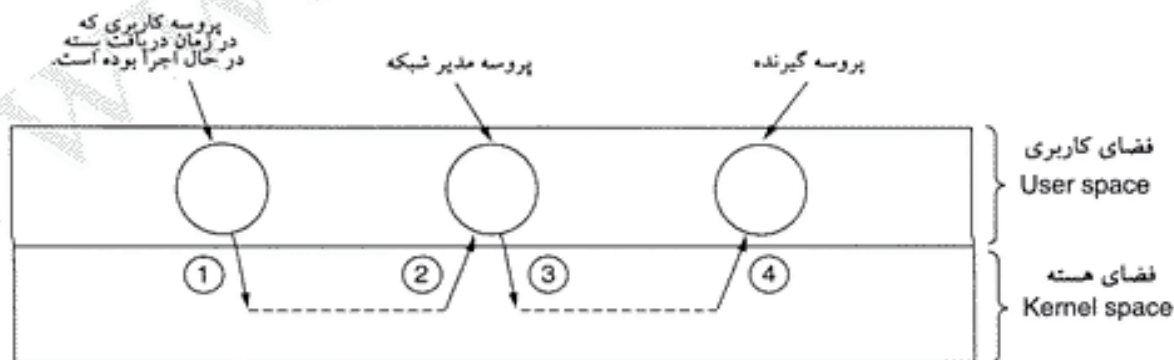
«خط لوله پردازنده» را قطع می‌کند، (۲) حافظه نهان (cache) پردازنده را در هم می‌ریزد، (۳) نیاز به تغییر روال کار مدیریت حافظه (یعنی عمل تعویض فضای کاری حافظه) دارد (۴) CPU را وادار به ذخیره‌سازی تعدادی رجیستر می‌کند. کاهش تعداد TPDUهای ارسالی با ضریب n ، تعداد وقفه‌ها و میزان سربار بسته‌ها را با ضریب n کاهش خواهد داد.

این واقعیت بیانگر آنست که برای کاهش وقفه‌ها در طرف مقابل، باید قبل از ارسال، ابتدا مقدار قابل توجهی داده، جمع‌آوری شود. الگوریتم Nagle و راه حل Clark که در بخشهای قبلی تشریح شدند بدین منظور مفید هستند.

قانون شماره ۳: کاهش تعداد «تعویض متن» (Context Switch)

انجام عملیات «تعویض متن» (مثلاً از حالت هسته به حالت کاربری) از لحاظ سربار تحمیلی مهلک است! این کار تمام ویژگیهای بد وقفه‌ها را دارد و بدترین آنها این است که دنباله‌ای طولانی از محتوای اولیه حافظه نهان CPU از دست می‌رود. داشتن یک پروسیجر کتابخانه‌ای که داده‌های ارسالی را تا رسیدن به حجم قابل توجه بافر کند، به کاهش دفعات «تعویض متن» کمک خواهد نمود. هم‌طور در سمت گیرنده، TPDUهای کوچک دریافتی باید جمع‌آوری شده و بطور یکجا تحویل کاربر گردد تا دفعات «تعویض متن» کاهش یابد.

در بهترین حالت، یک بسته ورودی موجب می‌شود که سیستم عامل مجبور به «تعویض متن» از حالت کاربر به حالت هسته^۱ باشد. سپس باید از هسته به پروسه دریافت‌کننده این بسته ورودی تغییر حالت بدهد. متأسفانه در بسیاری از سیستمهای عامل به دفعات بیشتری «تعویض متن» نیاز است. به عنوان مثال، اگر مدیر شبکه پروسه خاصی را در فضای کاربری^۲ اجرا کرده باشد، دریافت یک بسته موجب می‌شود که سیستم عامل ابتدا از کاربر فعلی به هسته، تغییر وضعیت (و تعویض متن) بدهد، سپس تغییر وضعیتی دیگر از هسته به پروسه مدیر شبکه، سپس تغییری دیگر از پروسه مدیر شبکه به هسته و نهایتاً از هسته به پروسه تحویل گیرنده بسته. این توالی در شکل ۴۳-۶ نشان داده شده است. تمام این عملیات تعویض متن که به ازای ورود هر بسته انجام می‌شود، زمان CPU را تا حد بسیار زیادی هدر خواهد داد و تأثیر مخربی بر روی کارایی شبکه خواهد گذاشت.



شکل ۴۳-۶. چهار بار تعویض متن برای رسیدن یک بسته به پروسه مدیر شبکه در فضای کاربری.

قانون شماره ۴: حداقل کردن دفعات کپی برداری

تعداد کپی برداری حتی از تعداد دفعات تعویض متن هم بدتر است. این موضوع که بخواهیم یک بسته ورودی را قبل از تحویل محتوای درونی آن به پروسه نهایی، سه یا چهار بار کپی کنیم، امری نامعمول و عجیب نیست. معمولاً

۲. User Space.

۱. User mode to Kernel mode Context Switch.

پس از دریافت یک بسته توسط کارت واسط شبکه و ذخیره موقت آن در یک بافر سخت افزاری ویژه، بسته به درون بافر هسته سیستم عامل کپی می شود. از آنجا نیز به بافر لایه شبکه و از آنجا به بافر لایه انتقال و نهایتاً به بافر پروسه کاربردی گیرنده منتقل می شود.

یک سیستم عامل هوشمند قادر به کپی یک کلمه در آن واحد است ولیکن ممکن است برای کپی یک کلمه حتی به ۵ دستورالعمل نیاز باشد. (شامل بار کردن کلمه در رجیستر، ذخیره آن در موقعیت جدید، اضافه کردن به رجیستر شاخص - index register تست برای تشخیص پایان داده ها و انشعاب شرطی) سه بار کپی کردن هر بسته در شرایطی که انتقال هر کلمه ۳۲ بیتی با ۵ دستورالعمل انجام می شود به $4 \div 15$ دستورالعمل یعنی حدوداً چهار دستورالعمل به ازای هر بایت، نیاز خواهد داشت. در یک CPU با سرعت 500MIPS، هر دستورالعمل ۲ نانوثانیه زمان می برد لذا سه بار انتقال هر بایت حدوداً ۸ نانوثانیه طول می کشد؛ یعنی حدود ۱ نانوثانیه به ازای هر بیت؛ بدین ترتیب چنین پردازنده ای قادر به پردازش 1-Gbps است. وقتی سر بار ناشی از پردازش سرآیند، مدیریت وقفه، عملیات تعویض متن (Context Switching) را نیز در نظر بگیریم شاید بتوان به سرعت پردازش 500Mbps رسید، در حالی که پردازش اصلی داده ها را نیز به حساب نیاورده ایم. بدیهی است که پردازش داده های شبکه اترنت 10-Gbps که با تمام سرعت ارسال می کند، بدین نحو میسر نخواهد بود.

در حقیقت، شاید با CPU فوق، حتی نتوان از عهده پردازش داده های یک خط 500Mbps که با سرعت تمام در جریان است، برآمد. از طرفی در محاسبات فوق، سرعت ماشین را 500Mbps فرض کرده ایم یعنی می تواند پانصد میلیون دستورالعمل را در هر ثانیه اجرا کند. در دنیای واقعی، ماشینها فقط زمانی می توانند با چنین سرعتی کار کنند که به حافظه رجوع نداشته باشند. عملیات حافظه عموماً ده بار کندتر از دستورالعملهای رجیستر به رجیستر اجرا می شوند. (به عبارتی ۲۰ نانوثانیه برای هر دستورالعمل حافظه). اگر بیست درصد از دستورات نیازمند رجوع به حافظه باشند (به عبارتی به بیرون از حافظه نهان رجوع کنند) که در خصوص ورود یک بسته کاملاً محتمل است، متوسط زمان اجرای دستورالعملها ۵/۶ نانوثانیه خواهد بود. $(0.8 \times 2 + 0.2 \times 20)$ اگر برای پردازش هر بایت چهار دستورالعمل اجرا شود، به $4 \div 22$ نانوثانیه برای پردازش هر بایت یا $8 \div 2$ نانوثانیه برای هر بیت نیاز خواهیم داشت که در این صورت ظرفیت پردازش به 357Mbps کاهش می یابد. اگر پنجاه درصد سر بار اضافی را در محاسبات خود وارد کنیم، این مقدار به 178Mbps می رسد. در اینجا سخت افزار نمی تواند کمک بیشتری بکند. مشکل اینجا است که تعداد کپی هایی که در سیستم عامل انجام می شود زیاد است.

قانون شماره ۵: می توانید پهنای باند بیشتری بخرید ولی تأخیر پایین تر خریدنی نیست

سه قانون بعدی به جای آن که به پردازش پروتکل پردازند در خصوص مسائل ارتباطی شبکه بحث می کنند. قانون اول بیان می کند که اگر پهنای باند بیشتری خواستید قادر به تهیه آن خواهید بود. قرار دادن یک رشته فیبرنوری دیگر در کنار رشته قبلی پهنای باند شما را دو برابر می کند ولیکن تأخیر را کاهش نخواهد داد. کم کردن تأخیر مستلزم بهبود نرم افزار پروتکل، سیستم عامل، یا کارت شبکه است. حتی اگر به تمام این بهبودها نائل شوید، تأخیر شما هرگز از تأخیر انتقال کمتر نخواهد شد. [بطور ذاتی هر کیلومتر سیم ۵ میکروثانیه و هر کیلومتر فیبرنوری ۳/۳ میکروثانیه تأخیر دارد و این تأخیر اجتناب ناپذیر است. -م]

قانون شماره ۶: پیشگیری از بروز ازدحام بهتر از رفع آن است

قطعاً این مثل قدیمی که: «پیشگیری به مراتب بهتر از درمان است» برای مسئله ازدحام در شبکه نیز صادق است. وقتی شبکه ای با ازدحام مواجه می شود بسته هایی از بین می روند، پهنای باند تلف می شود، تأخیرهای نامعقول پدید می آید و تبعاتی از این قبیل. بیرون آمدن و رفع ازدحام دشوار و زمان بر است. بهتر آن است که نگذاریم

از دحام رخ بدهد. پیشگیری از ازدحام همانند تزریق واکسن DTP است: اندکی دردسر دارد ولی از مشکلات بزرگ پیشگیری می‌کند.

قانون شماره ۷: اجتناب از انقضای مهلت

وجود تایمرها در هر شبکه‌ای قطعاً لازم است ولی فقط باید در موارد ضروری از تایمر استفاده کرد و مهلت انقضای زمان تایمرها بایستی حداقل (بهینه) باشند. وقتی تایمری به صفر می‌رسد، عموماً عملیاتی از نو تکرار خواهد شد. اگر تکرار این عملیات واقعاً لازم نباشد منابع سیستم بیهوده هدر می‌رود.

برای اجتناب از انجام عملیات زائد باید زمان تایمرها به دقت تنظیم شده و اندکی با مقدار لازم فاصله داشته باشد، زیرا محافظه‌کاری زیاد و تنظیم مقادیر بزرگ در تایمرها باعث بروز تأخیرات نامعقول در هنگام از دست رفتن TPDU می‌شود. از طرفی اگر تایمر در زمانی که واقعاً موعد آن نیست به صفر برسد، زمان CPU صرف عملیات بیهوده می‌شود، پهنای باند به هدر می‌رود و بی‌هیچ دلیل بار زیادی بر روی دهها مسیر یاب تحمیل می‌کند.

۶-۶ پردازش سریع TPDU

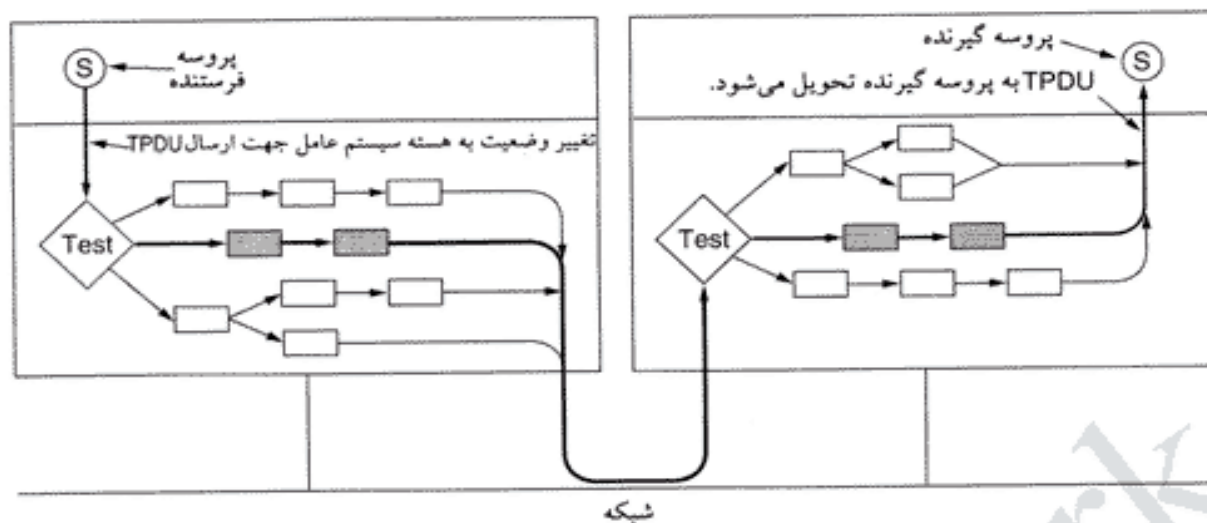
نتیجه‌گیری منطقی از قضایایی که در بالا بدان پرداختیم آنست که مانع اصلی در ایجاد شبکه‌های سریع، نرم‌افزار پروتکل آنهاست. در این بخش روشهایی را مرور می‌کنیم که به این نرم‌افزار سرعت می‌بخشند. برای کسب آگاهی بیشتر به مراجع (Clark et al., 1989; and Chase et al., 2001) مراجعه نمایید.

سرریز پردازش TPDU دو عامل دارد: سرریز تحمیلی به ازای هر TPDU و سرریز تحمیلی به ازای هر بایت. هر دوی آنها را باید کاهش داد. عامل کلیدی در پردازش سریع TPDU آن است که TPDUهای معمولی (حاوی داده) را از مابقی جدا کرده و آنها را به صورت ویژه پردازش نماییم. اگرچه به دنباله‌ای از TPDUهای خاص نیاز است تا بتوان به حالت ESTABLISHED وارد شد ولیکن از اینجا به بعد پردازش TPDU سراسر است و ساده خواهد بود، تا وقتی که یکی از طرفین اقدام به قطع اتصال نماید.

اجازه بدهید فرض را بر آن بگذاریم که طرف فرستنده اقدام به برقراری اتصال کرده و اکنون در حالت ESTABLISHED قرار دارد و می‌خواهد داده‌هایی را بفرستد. برای سادگی بحث فرض می‌کنیم که «واحد انتقال» (Transport Entity) درون هسته سیستم عامل قرار دارد؛ اگرچه ایده‌هایی که مطرح می‌شوند برای زمانی که «واحد انتقال» یک پروسه در فضای کاربری است یا حتی به صورت توابع کتابخانه‌ای درون پروسه فرستنده جای گرفته، نیز صادق هستند. در شکل ۶-۴، پروسه فرستنده داده، به هسته سیستم عامل رجوع می‌کند تا عملیات SEND (ارسال) انجام شود. اولین کاری که واحد انتقال انجام می‌دهد بررسی و تشخیص حالت معمولی است یعنی باید: (۱) اتصال در حالت ESTABLISHED قرار داشته باشد، (۲) هیچیک از طرفین سعی در بستن اتصال نکرده باشند، (۳) TPDU کامل و معتبر باشد و (۴) در سمت گیرنده فضای پنجره به اندازه کافی موجود باشد. اگر تمام شرایط فوق برآورده شود به آزمایش بیشتری نیاز نیست و آن TPDU می‌تواند در مسیر پردازش سریع قرار بگیرد. طبعاً اکثر بسته‌های TPDU از همین مسیر می‌گذرند.

در حالت عادی سرآیند TPDUهای متوالی و حاوی داده، تقریباً مشابه هم هستند.^۱ برای بهره‌برداری مفید از این ویژگی، «نمونه سرآیند» (Header Prototype) درون واحد انتقال کپی می‌شود. در ابتدای مسیر پردازش سریع، سرآیند TPDU با حداکثر سرعت و کلمه به کلمه درون بافر جدید کپی می‌گردد. فیلدهایی که از یک TPDU به TPDU دیگر تغییر می‌کنند درون بافر رونویسی می‌شوند. عموماً مقدار فیلدهایی را که در هر TPDU

۱. از لحاظ مقدار «تقریباً» مشابهند و گرنه از لحاظ ساختار سرآیند یقیناً مثل هم هستند. -م



شکل ۴۴-۶. مسیر پردازش سریع بسته ها از فرستنده به گیرنده با خط تیره تر مشخص شده است. مراحل پردازش نیز بصورت خاکستری نشان داده شده است.

تغییر می کنند (همانند فیلد شماره ترتیب و شماره Ack)، به سادگی می توان از «تغییرهای حالت»، موجود در واحد انتقال بدست آورد. سپس یک اشاره گر به سرآیند کامل TPDU، به همراه اشاره گر دومی به داده های کاربر، تحویل لایه شبکه می شود. در لایه شبکه نیز استراتژی مشابهی دنبال می شود. (استراتژی لایه شبکه در شکل ۴۴-۶ نشان داده نشده است). در آخر، لایه شبکه بسته حاصله را جهت انتقال به لایه پیوند داده تحویل می دهد.^۱

برای آن که ببینیم این روال در عمل چگونه کار می کند، TCP/IP را در نظر می گیریم. در شکل ۴۵-۶ الف سرآیند TCP نشان داده شده است. فیلدهایی که در TPDUs متوالی مشابه هستند (یعنی مقدار آنها تغییر نمی کند) به صورت خاکستری نشان داده شده اند. تمام کاری که «واحد انتقال» انجام می دهد آن است که (۱) این پنج کلمه را از درون «نمونه سرآیند» (Header Prototype) به بافر خروجی منتقل کند؛ (۲) شماره ترتیب بعدی را در محل مربوطه قرار بدهد (از طریق انتقال یک کلمه در حافظه)، (۳) کد جمع کنترلی (Checksum) را محاسبه کند و (۴) به شماره ترتیب در حافظه اضافه نماید. سپس سرآیند تنظیم شده و داده ها تحویل پروسیجر IP می شود تا بطور طبیعی ارسال گردد. نرم افزار IP نیز «نمونه سرآیند» پنج کلمه ای خود را (شکل ۴۵-۶ ب) به درون بافر کپی می کند، فیلد Identification را مقداردهی و کد جمع کنترلی آن را محاسبه می نماید. اکنون بسته، آماده ارسال است.

حال ببینیم، مسیر پردازش سریع در طرف گیرنده از شکل ۴۴-۶ چگونه کار می کند. در مرحله ۱ ابتدا «رکورد اتصال» مربوط به TPDU ورودی پیدا می شود. در نرم افزار TCP، رکورد اتصال می تواند در یک «جدول درهم سازی» (Hash Table) ذخیره شده و کلید آن برحسب تابعی ساده از آدرس IP و دو شماره پورت محاسبه گردد. به محض پیدا شدن موقعیت رکورد متناظر با بسته ورودی، ابتدا آدرسهای IP و شماره های پورت آن با این رکورد مقایسه می شود تا صحت رکورد پیدا شده تأیید گردد.

۱. اگر بخواهیم عملکرد «مسیر پردازش سریع» را به زبان ساده بیان کنیم بدین نحو عمل می کند که اگر TPDU از نوع معمولی و حاوی داده باشد، برای آن از قبل یک سرآیند ایجاد می شود که خوشبختانه برای هر TPDU جدید فقط تعداد کمی از فیلدهای آن تغییر می کند. بلافاصله پس از تنظیم سرآیند هر TPDU، اشاره گر محل شروع این سرآیند و اشاره گر محل شروع داده های کاربر، تحویل لایه شبکه می شود. بنابراین بجای انتقال و کپی یک بسته کامل، اشاره گرها بین دو لایه رد و بدل می شوند. -م

Source port		Destination port	
Sequence number			
Acknowledgement number			
Len	Unused		Window size
Checksum		Urgent pointer	

VER.	IHL	TOS	Total length	
Identification				Fragment offset
TTL		Protocol	Header checksum	
Source address				
Destination address				

(الف)

(ب)

شکل ۶-۴۵. (الف) سرآیند TCP (ب) سرآیند IP. در هر دو سرآیند، فیلدهای خاکستری

بدون هیچ تغییری از «نمونه سرآیند» (Header Prototype) موجود گرفته می‌شود.

یک بهینه‌سازی که می‌تواند سرعت جستجوی «رکورد اتصال» را افزایش بدهد آن است که همیشه اشاره‌گر آخرین رکوردی که از آن استفاده شده، نگهداری گردد و جستجو از آن نقطه آغاز شود. کلارک و همکارانش (Clark et al. 1989) این بهینه‌سازی را آزمایش و مشاهده کردند که احتمال موفقیت سریع در جستجو به بیش از ۹۰ درصد افزایش می‌یابد. روشهای جستجوی دیگر در مرجع (McKenney and Dove; 1992) تشریح شده است.

سپس TPDU بررسی می‌شود تا مشخص گردد که آیا از نوع معمولی است یعنی: اتصال در حالت ESTABLISHED قرار داشته باشد. هیچیک از طرفین تقاضای خاتمه اتصال نداده باشند، TPDU یک بسته کامل و معتبر باشد، هیچیک از بیت‌های پرچم (Flag Bits) در TPDU، تنظیم نشده باشد و شماره ترتیب TPDU دریافتی همان شماره مورد نظر باشد. این بررسی‌ها مستلزم اجرای چندین دستورالعمل ماشین است. اگر این شرایط برآورده شود، یک «پروسسجر پردازش سریع» برای آن TPDU فراخوانی می‌شود.

«مسیر پردازش سریع» رکورد اتصال را به‌نگام‌سازی کرده و داده‌ها را برای پروسه کاربرکپی می‌نماید. در حین کپی، کد جمع‌کنترلی (checksum) داده‌ها را محاسبه می‌کند تا نیاز به یک گذر اضافی برای این عمل نیاز نباشد. اگر کد جمع‌کنترلی صحیح بود، رکورد اتصال به‌نگام‌سازی شده و پیغام Ack بازگردانده می‌شود. الگوی کلی تست سریع سرآیند برای تشخیص آن که آیا سرآیند TPDU، سرآیند مورد انتظار هست یا خیر توسط پروسسجری به نام «پیشگویی سرآیند» (Header Prediction) انجام می‌شود و در بسیاری از پیاده‌سازیهای عملی TCP از آن استفاده شده است. هرگاه این بهینه‌سازی و بهینه‌سازیهای دیگری که در این فصل تشریح کردیم در کنار هم مورد استفاده قرار بگیرد می‌توان به ۹۰ درصد سرعت کپی داده‌ها از حافظه به حافظه رسید. (با فرض آن که شبکه به قدر کافی سریع است).

دو مورد دیگر که به کمک آنها می‌توان بهبود چشمگیری در کارایی بوجود آورد مدیریت بافر و مدیریت تایمرهاست. موضوع مرتبط با مدیریت بافر، اجتناب از کپی برداریهای بی‌مورد است. به این موضوع در بالا اشاره کردیم. مدیریت تایمر اهمیت بیشتری دارد چرا که تقریباً اغلب آنها (اگر بدرستی تنظیم شده باشند) منقضی نمی‌شوند: آنها بدین منظور تنظیم و راه‌اندازی می‌شوند که اگر یک TPDU از بین رفت از نو ارسال شود در حالی که اغلب TPDUها به درستی تحویل و پیغام اعلام وصول آنها نیز به سلامت بر می‌گردند. بنابراین مدیریت بهینه تایمرها بسیار اهمیت دارد.

یک ساختار رایج برای مدیریت تایمرها آن است که از یک «لیست پیوندی» (Linked List) متشکل از

رخدادهای تایمر که بر حسب زمان انقضای آنها مرتب شده، استفاده گردد.^۱ اولین عنصر این لیست پیوندی حاوی یک شمارنده است که تعیین می کند تا انقضای مهلت آن، چند تیک ساعت باقی مانده است. هریک از عناصر متوالی این لیست پیوندی شمارنده ای دارند که مشخص می کند در مقایسه با عنصر قبلی در لیست، چند تیک ساعت بعدتر منقضی می شوند. بنابراین اگر سه تایمر باید به ترتیب پس از ۳، ۱۰ و ۱۲ تیک منقضی شوند مقادیر شمارنده های آنها در لیست پیوندی به ترتیب ۳ و ۷ و ۲ خواهد بود.

در هر تیک ساعت، شمارنده اولین عنصر لیست پیوندی یک واحد کاهش می یابد. هرگاه این شمارنده به صفر برسد، رخداد مربوطه پردازش شده و با حذف آن از لیست، عنصر بعدی در جلوی این لیست قرار می گیرد. بدون آن که نیازی به تغییر در مقدار شمارنده این عنصر باشد. در این روش، حذف و اضافه یک تایمر به لیست پیوندی، عملیاتی پرهزینه است و زمان انجام این دو عمل متناسب با طول لیست پیوندی می باشد.

اگر مقدار حداکثر تایمرها محدود و از قبل معلوم باشد می توان از روش کارآمدتری استفاده کرد. در این روش از آرایه ای به نام «چرخ زمان سنجی» بهره گرفته می شود. این آرایه در شکل ۶-۴۶ نشان داده شده است. هر «برش» (Slot) متناظر با یک تیک ساعت است. در این شکل زمان فعلی، $T=4$ فرض شده است. تایمرها به گونه ای برنامه ریزی شده اند که نسبت به زمان فعلی، پس از ۳، ۱۰ و ۱۲ تیک ساعت منقضی شوند. اگر در همین لحظه به تایمری احتیاج شد که قرار است پس از ۷ تیک ساعت منقضی شود، یک درایه (Entry) برای آن در برش ۱۱ (Slot 11) ایجاد می شود. به روش مشابه اگر نیاز شد که تایمر $T+1$ حذف شود، لیستی که شروع آن در برش ۱۴ مشخص شده جستجو و درایه مورد نظر حذف می شود. دقت کنید که آرایه شکل ۶-۴۶ نمی تواند به غیر از تایمرهایی را که زیر $T+15$ هستند ایجاد کند. [به عبارتی حداکثر زمان قابل تنظیم در تایمرها معادل ۱۵ تیک ثانیه است.]



شکل ۶-۴۶. یک «چرخ زمان سنجی» (Timing Wheel).

۱. یعنی هر یک از عناصر این لیست پیوندی مشخص می کنند که چقدر بعد چه اتفاقی باید بیفتد. -م

وقتی ساعت تیک می زند، اشاره گر زمان فعلی به اندازه یک برش در آرایه جلو می رود. (به صورت چرخشی) اگر درایه ای که اکنون به آن اشاره می شود غیر صفر باشد، تمام تایمرهای تعریف شده در آن پردازش می شوند.^۱ گونه های بی شماری از روش فوق در مرجع (Varghese, Lauck, 1987) بحث شده است.

۵-۶-۶ پروتکل هایی برای شبکه های گیکابیتی

شبکه های گیکابیتی در ابتدای دهه نود در صحنه ظاهر شدند. عموم افراد سعی کردند که از همان پروتکل های قدیمی بر روی شبکه جدید استفاده کنند ولی چیزی نگذشت که مشکلات خود را نشان دادند. در این بخش به برخی از این مشکلات و راهکارهایی که پروتکل های جدید برای حرکت به سمت شبکه های سریعتر برگزیده اند، خواهیم پرداخت.

اولین مشکل آن است که اکثر پروتکل های فعلی، از شماره ترتیب ۳۲ بیتی برای بسته ها، استفاده کرده اند. وقتی اینترنت شروع به کار کرد، خطوط مابین مسیر یابها، غالباً خطوط اجاره ای 56Kbps بودند و اگر یک ماشین با تمام سرعت اقدام به ارسال می کرد بیش از یک هفته طول می کشید تا شماره ترتیب ۳۲ بیتی به مقدار قبلی آن برگردد. برای طراحان TCP، عدد ۲^{۳۲} تقریب بسیار خوبی از بی نهایت تلقی می شد زیرا خطر آن که بسته ای برای یک هفته در زیر شبکه سرگردان باشد و سپس به صورت تکراری دریافت شود عملاً وجود نداشت. در شبکه اترنت 10Mbps، زمان تکرار اعداد ۳۲ بیتی به ۵۷ دقیقه کاهش یافت ولی هنوز قابل تحمل و مدیریت پذیر بود. در اترنت یک گیکابیت بر ثانیه، زمان تکرار اعداد به ۳۴ ثانیه رسید که متأسفانه کمتر از طول عمر حداکثر هر بسته یعنی ۱۲۰ ثانیه است. در اینجا ۲^{۳۲} تقریب خوبی از مقدار بی نهایت نیست و ممکن است در حالی که هنوز بسته های قدیمی در زیر شبکه موجودند بسته هایی با شماره تکراری تولید شده و گیرنده احتمالاً به اشتباه بیفتند. اگرچه در RFC 1323 راهی موقت برای حل این مشکل پیشنهاد شده است.

مشکل از آنجا ناشی می شود که طراحان پروتکل به سادگی فرض کرده اند زمانی که طول می کشد تا از کل فضای ۳۲ بیتی شماره ترتیب استفاده شود از حداکثر طول عمر بسته بیشتر است. در نتیجه (با این فرض) لازم نبود کسی در خصوص وجود بسته هایی با شماره های تکراری (که در اثر برگشت مقادیر ۳۲ بیتی به مقدار قبلی پیش می آید) نگران باشد. در سرعت های گیکابیتی این فرض نانوشته با شکست مواجه می شود.

مشکل دوم آن است که سرعت مخابره داده ها از سرعت پردازش کامپیوترها پیشی گرفته است. (قابل توجه مهندسين کامپیوتر: به میدان رفته و مهندسين مخابرات را شکست بدهید!! ما روی شما حساب می کنیم!) در اوائل دهه هفتاد میلادی، شبکه ARPANET از خطوط 56Kbps و کامپیوترهایی با سرعتی حدود 1 MIPS بهره می گرفت. بسته های اطلاعاتی نیز ۱۰۰۸ بیت بودند و بدین ترتیب ARPANET قادر به تحویل حدود ۵۶ بسته در هر ثانیه بود. هر ماشین میزبان در زمانی حدود ۱۸ میلی ثانیه که به ازای هر بسته مهلت پردازش داشت، می توانست ۱۸۰۰۰ دستورالعمل ماشین را اجرا نماید. البته اختصاص تمام این دستورالعملها به پردازش بسته، کل زمان CPU را مصرف می کند ولیکن اگر فقط ۹۰۰۰ دستورالعمل ماشین به پردازش هر بسته اختصاص داده شود، نیمی از توان CPU برای کارهای اصلی باقی می ماند.

این اعداد و ارقام را با کامپیوترهای 1000 MIPS که بسته های ۱۵۰۰ بیتی را بر روی خطوط گیکابیتی منتقل

۱. به زبان ساده، آرایه فوق را در یک دایره با ۱۶ قطاع تجسم کنید که عقربه ای روی آن حرکت می کند و در هر لحظه در یکی از این قطاعها قرار می گیرد. اگر به فرض نیاز به تایمری داشتید که باید ۵ تیک ثانیه بعد منقضی شود نسبت به مکان فعلی عقربه، ۵ قطاع جلوتر، اشاره گر آن را قرار می دهید. هر بار که عقربه یک قطاع جلو می رود تمام کارهایی که اشاره گرشان درون آن قطاع قرار دارد، انجام می شود. -م

می‌کنند، مقایسه نمایید. در چنین شبکه‌ای نرخ ارسال بیش از ۸۰۰۰۰ بسته در ثانیه است و طبعاً اگر بخواهیم نیمی از توان پردازشی CPU را به بقیه برنامه‌های کاربردی اختصاص بدهیم، بایستی پردازش هر بسته در ۶/۲۵ میکروثانیه تکمیل شود. یک کامپیوتر 1000MIPS در ۶/۲۵ میکروثانیه قادر به اجرای ۶۲۵۰ دستورالعمل است یعنی حدود یک سوم تعداد دستورالعملهایی که ماشینهای ARPANET قادر به اجرای آنها برای هر بسته بودند. مضاف بر این هر دستورالعمل در ماشینهای مدرن RISC، نسبت به دستورالعمل ماشینهای قدیمی تر CISC کار کمتری انجام می‌دهد لذا مشکل حادتر از آن چیزی است که به نظر می‌رسد. نتیجه‌گیری آن است که برای پردازشهایی که در نرم‌افزار پروتکلها انجام می‌شود زمان کمتری در اختیار است فلذا پروتکلها باید ساده‌تر و سریعتر شوند.

مشکل سوم آن است که پروتکل Go Back n بر روی خطوطی که در آنها حاصلضرب پهنای باند در تأخیر بسیار زیاد است، ضعیف عمل می‌کند. به عنوان مثال به یک خط چهار هزار کیلومتری که با سرعت 1-Gbps کار می‌کند، دقت نمایید: زمان تأخیر رفت و برگشت ۴۰ میلی‌ثانیه است و در این زمان فرستنده قادر به ارسال ۵ مگابایت داده می‌باشد؛ طبعاً اگر خطایی گزارش شود مربوط به ۴۰ میلی‌ثانیه قبل بوده است. اگر از پروتکل Go Back n (عقبگرد به اندازه n) استفاده شده باشد، فرستنده نه تنها مجبور است بسته خراب را از نو بفرستد بلکه باید ۵ مگابایت بسته‌های بعد از آن را نیز مجدداً ارسال نماید. به وضوح استفاده از این روش، اتلاف بسیار زیادی در منابع شبکه خواهد بود.

مشکل چهارم آن است که خطوط گیگابیتی تفاوت بنیانی با خطوط مگابیتی دارند: این تفاوت از آنجاست که خطوط طولانی گیگابیتی نسبت به «تأخیر» محدودیت دارند در حالی که دیگری محدودیت «پهنای باند» دارد. در شکل ۶-۴۷ زمانی که طول می‌کشد یک فایل یک مگابیتی از طریق خطی ۴۰۰۰ کیلومتری با نرخهای متفاوت ارسال شود، نشان داده شده است. تا سرعت 1-Mbps، زمان انتقال متأثر از نرخ ارسال بیتهاست. در سرعت 1-Gbps، تأخیر ۴۰ میلی‌ثانیه‌ای رفت و برگشت بر زمان ۱ میلی‌ثانیه‌ای انتقال بیتها بر روی فیبرنوری غالب می‌شود. از اینجا به بعد افزایش پهنای باند تأثیر چندانی در تأخیر انتقال ندارد.

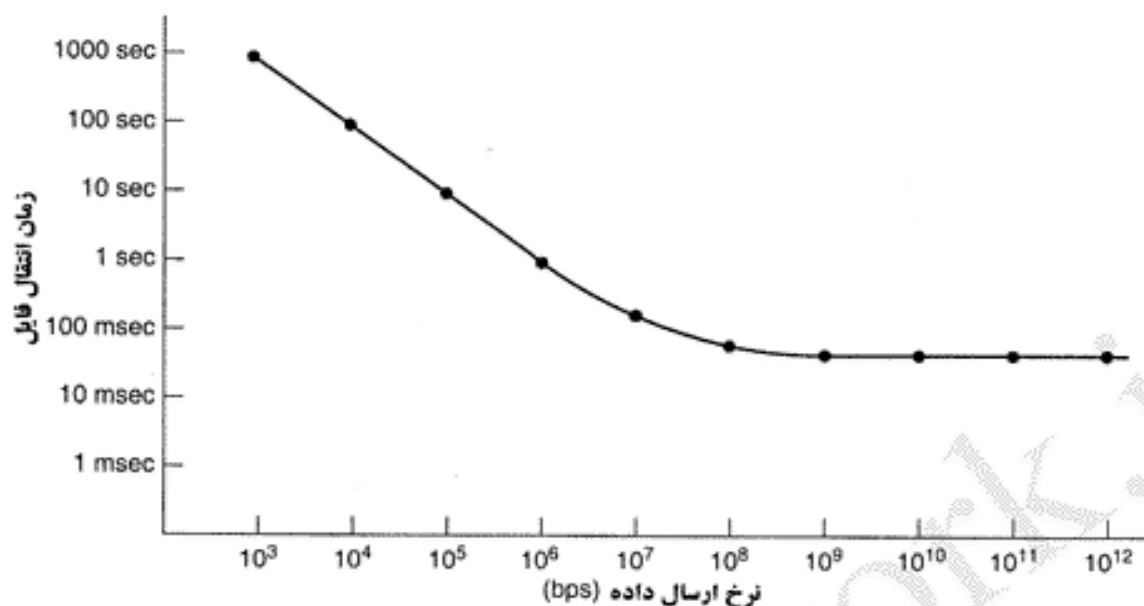
شکل ۶-۴۷ نکات مایوس‌کننده‌ای برای پروتکلهای شبکه دارد. این شکل بیانگر آن است که پروتکلهای «توقف و انتظار» (Stop & Wait) همانند RPC از لحاظ کارایی محدودیت ذاتی دارند. این محدودیت به خاطر سرعت نور تحمیل می‌شود و هیچگونه پیشرفت تکنولوژیک در فیزیک نور نمی‌تواند بر این محدودیت فائق آید. مشکل پنجم که اشاره به آن خالی از لطف نیست ریشه در مسائل تکنولوژیک، پروتکل یا مسائلی بدین شکل ندارد بلکه ناشی از کاربردهای جدید شبکه است. در یک بیان ساده در بسیاری از کاربردهای شبکه گیگابیتی (مثل کاربردهای چند رسانه‌ای) «واریانس زمان دریافت بسته‌ها»، اهمیتی همسنگ با «تأخیر متوسط دریافت» آنها دارد. تحویل آهسته ولی یکنواخت بسته‌ها ارجحتر از تحویل سریع ولی ناگهانی آنهاست.

پس از معرفی مشکلات اجازه بدهید به روشهای حل و فصل آنها پردازیم. ابتدا تذکراتی کلی ارائه می‌کنیم و سپس به مکانیزمهای پروتکل، ساختار بسته‌ها و نرم‌افزار پروتکل نگاهی خواهیم انداخت.

یک اصل اساسی که تمام طراحان شبکه باید در نظر داشته باشند آن است که:

«طراحی باید مبتنی بر سرعت بالا باشد نه برای بهینه‌سازی پهنای باند»

پروتکلهای قدیمی تر اغلب بدان منظور طراحی شده‌اند که تعداد بیتهای ارسالی بر روی کانال به حداقل برسد و بدین منظور از فیلدهای کوچک و ادغام چند فیلد در یک بایت یا کلمه بهره گرفته‌اند. امروزه پهنای باند فراوانی در اختیار است و میزان پردازش در پروتکل به یک مشکل تبدیل شده است، فلذا پروتکلها باید برای به حداقل رساندن حجم پردازش لازم، طراحی شوند. طراحان IPv6 به روشنی این اصل اساسی را درک کرده بودند.



شکل ۶-۴۷. زمان انتقال و تصدیق وصول یک فایل یک مگابیتی از طریق یک خط ۴۰۰۰ کیلومتری.

یک راهکار اغواکننده برای رسیدن به سرعت بالا، ساختن کارتهای واسط شبکه به کمک سخت افزار سریع است. [یعنی بخشی از پروتکل های نرم افزاری توسط سخت افزار شبکه پیاده سازی شود.] مشکل در پیش گرفتن این استراتژی آن است که بدون داشتن یک پروتکل بسیار ساده، کارت سخت افزاری به خودی خود تبدیل به یک برد اضافی با یک CPU مستقل و یک برنامه خاص می شود. برای آنکه «کمک پردازنده شبکه» ارزانتر از پردازنده اصلی تمام شود، اغلب از CPU کندتر استفاده می شود. در نتیجه اگر خود پروتکل ساده نباشد، CPU سریع باید آنقدر منتظر بماند تا CPU دیگر (CPU کندتر) کار خود را انجام بدهد. این ایده که CPU سریع در حین انتظار به کارهای دیگری پردازد بیشتر به یک افسانه شبیه است و مشکلات خاص خود را دارد. مضاف بر این وقتی دو CPU همه منظوره، با یکدیگر در ارتباط باشند شرایط رقابتی (Race) رخ می دهد و به پروتکل های پیچیده ای برای هماهنگ کردن آنها با یکدیگر نیاز خواهد بود. عموماً بهترین راهکار، ساده تر کردن پروتکل و محول نمودن کار به CPU اصلی است.

حال ببینیم مسئله فیدبک [یعنی اعلام طول پنجره یا هر مورد مشابه توسط سمت مقابل] در پروتکل های بسیار سریع چگونه حل و فصل می شود. به دلیل وجود حلقه با تأخیر (نسبتاً) بالا، حتی الامکان باید از ایجاد فیدبک اجتناب شود زیرا زمان زیادی طول می کشد تا سیگنال گیرنده به فرستنده برگردد. مثالی از فیدبک، اعمال نظارت و کنترل نرخ ارسال با استفاده از پروتکل پنجره لغزان می باشد. برای رهایی از تأخیر زیادی که در اعلام طول پنجره گیرنده به فرستنده همیشه وجود دارد، در شبکه های سریع بهتر است از یک پروتکل مبتنی بر نرخ پایه (و بدون فیدبک) بهره گرفته شود. در چنین پروتکلی فرستنده می تواند به هر اندازه که تمایل دارد داده بفرستد ولی حق ندارد با نرخ بیشتر از مقداری که قبلاً توافق کرده اند، ارسال نماید.

مثال دیگری از فیدبک، الگوریتم «شروع آهسته ژاکوبسن» است. این الگوریتم چندین آزمایش انجام می دهد تا مشخص کند که شبکه از عهده چه مقدار داده بر می آید. [بخش ۵-۹] در شبکه های بسیار سریع، انجام چند آزمایش (و حتی یکی دو آزمایش) برای ارزیابی چگونگی پاسخ شبکه، بخش بسیار بزرگی از بهنای باند را تلف می کند. الگوی کارآمدتر آن است که در همان ابتدای برقراری اتصال، فرستنده، گیرنده و شبکه همگی منابع مورد نیاز را رزرو نمایند. رزرو کردن منابع از قبل، این حسن بزرگ را دارد که ساده تر می توان مقدار لرزش (Jitter) را

کاهش داد. کوتاه سخن آن که حرکت به سوی شبکه های بسیار سریع، طراحی پروتکلها را اجباراً به طرف اتصال گرا شدن یا چیزی شبیه به آن سوق می دهد. البته اگر در آینده، پهنای باند آنقدر فراوان و کم اهمیت شود که کسی نگران هدر رفتن مقدار قابل توجهی از آن نباشد، اصول طراحی نیز بسیار متفاوت خواهد بود.

ساختار و قالب هر بسته در شبکه های گیگابیتی حائز اهمیت است. در سرآیند هر بسته حتی الامکان باید تعداد بسیار کمی فیلد تعریف شده باشد تا زمان پردازش آن کاهش یابد. در ضمن این فیلدها باید بقدر کافی بزرگ باشند تا کار خود را بدون محدودیت و نیاز به پردازشهای اضافی انجام بدهند. همچنین هر فیلد باید به صورت کلمه (یعنی ۱ بایتی یا ۲ و ۴ و ۸ و ۱۶ بایتی) تعریف شوند تا پردازش آنها بسیار ساده باشد. منظورمان از فیلد «به قدر کافی بزرگ» آن است که مشکلاتی نظیر برگشت شماره ترتیب به مقادیر تکراری یا ناتوانی گیرنده از اعلام اندازه بزرگ پنجره خود و مسائلی از این قبیل پدید نیاید.

بعلاوه بایستی برای بخش سرآیند و بخش داده ها در هر بسته، دو کد کشف خطای مجزا و مستقل محاسبه و درج شود، به دو دلیل: اول آن که در صورت عدم نیاز به نظارت بر خطای داده ها [مثلاً برای داده های صدا و تصویر]، بتوان فقط سرآیند بسته را از لحاظ صحت مقادیر بررسی کرد. دوم آن که بتوان قبل از کپی کردن داده ها در فضای پروسه کاربر از صحت سرآیند مطمئن شد. مطلوب آن است که کنترل صحت بخش داده در خلال کپی کردن آنها در فضای پروسه کاربر انجام شود ولیکن اگر سرآیند بسته صحیح نباشد ممکن است داده ها به اشتباه برای پروسه دیگری کپی شوند. برای اجتناب از کپی اشتباهی داده ها، داشتن دو کد کشف خطای مستقل الزامی است. در این صورت می توان بررسی صحت بخش داده را به زمان کپی آن موکول کرد.

طول حداکثر داده ها در هر بسته باید بزرگ باشد تا حتی در مواجهه با تأخیر زیاد، کارایی عملیات بالا باشد. هر چه طول داده ها بیشتر باشد درصدی از پهنای باند که صرف سرآیند هر بسته می شود کاهش خواهد یافت. بسته های ۱۵۰۰ بایتی برای شبکه های گیگابیتی بسیار کوچکند.

ویژگی ارزشمند دیگر آن است که فرستنده بتواند به همراه تقاضای برقراری اتصال، مقداری داده نیز بفرستد. بدین ترتیب در زمان رفت و برگشت صرفه جویی می شود.

در آخر، چند کلمه صحبت در خصوص نرم افزار پروتکل خالی از لطف نیست. اندیشه مثبت بر موفقیت آمیز بودن عمل متمرکز می شود در حالی که بسیاری از پروتکلهای قدیمی بیشتر بر این محور تکیه دارند که اگر خطایی رخ داد (مثلاً بسته ای در بین راه از بین رفت) چه باید کرد. برای آن که اجرای پروتکلها سریع شود، طراح باید هدف خود را بر آن بگذارد که وقتی همه چیز به خوبی پیش می رود به حداقل پردازش نیاز باشد. سپس بدان پردازد که چگونه می توان در هنگام بروز خطا حجم پردازشها را به حداقل رساند. [اولویت با حداقل بودن پردازش در شرایط عادی است چرا که شرایط غیرعادی به ندرت پیش می آید. -م]

مسئله دیگر در طراحی نرم افزار پروتکلها به حداقل رساندن زمان کپی برداری [انتقال داده ها در حافظه] است. همانگونه که قبلاً دیدیم، کپی برداری از داده ها عامل عمده تحمیل سربار است. آرمانی آنست که سخت افزار، بسته ورودی را در قالب یک بلوک یکپارچه و به صورت یکجا به حافظه منتقل نماید. سپس نرم افزار باید این بسته را تنها با یک عمل کپی بلوکی به بافر کاربر منتقل کند. براساس چگونگی عملکرد حافظه نهان CPU ممکن است اجتناب از حلقه تکرار در برنامه نویسی به منظور کپی داده ها، مطلوبتر باشد. به عبارت دیگر برای کپی کردن ۱۰۲۴ بایت داده، بهتر آن است که به جای حلقه از ۱۰۲۴ دستورالعمل MOVE پشت سرهم (یا ۱۰۲۴ جفت دستورالعمل LOAD و STORE) استفاده شود! به لحاظ اهمیت حیاتی، زیربرنامه کپی بایستی با کدهای اسمبلی نوشته شود مگر آن که بتوان به طریقی کامپایلر را به منظور تولید کد بهینه و سریع تنظیم کرد.

۷-۶ خلاصه

لایه انتقال کلید آشنایی با پروتکل‌های لایه‌ای است. این لایه خدمات متنوعی را عرضه می‌کند ولی مهمترین آنها ایجاد یک استریم مطمئن، انتها به انتها و اتصال‌گرا بین گیرنده و فرستنده به منظور انتقال دنباله‌ای از بایت‌هاست. دسترسی به خدمات این لایه از طریق یکسری توابع اولیه و پایه صورت می‌گیرد که این توابع برقراری اتصال، استفاده از آن (جهت ارسال و دریافت) و خاتمه اتصال را امکان‌پذیر کرده‌اند. یکی از واسطه‌های رایج برای لایه انتقال با نام «سوکت‌های برکلی» (Berkeley Socket) ارائه شده است.

پروتکل‌های لایه انتقال باید بتوانند بر اتصالانی که در یک شبکه نامطمئن ایجاد می‌شوند مدیریت کنند. برقراری اتصال از آن جهت پیچیده است که امکان دارد بسته‌های تکراری و معلق در زیرشبکه، در لحظه‌ای به گیرنده برسند که او را به اشتباه بیندازند. برای حل و فصل این مشکل، باید برای برقراری یک اتصال از روش دست‌تکانی سه‌مرحله‌ای (3-Way Handshake) استفاده شود. ختم یک اتصال ساده‌تر از برقراری آن است ولیکن به دلیل «مشکل دو سپاه» (Two-Army Problem) چندان هم پیش پا افتاده نیست.

حتی وقتی که لایه شبکه کاملاً مطمئن و بدون خطاست باز هم لایه انتقال کار زیادی باید انجام بدهد. این لایه باید وظیفه ارائه خدمات اولیه (توابع اولیه)، مدیریت اتصالات و تایمرها را بر عهده بگیرد و «اعتبار» تخصیص بدهد.

ایترنت در لایه انتقال دو پروتکل دارد: UDP و TCP. پروتکلی بدون اتصال است که در حقیقت همان خصوصیات پروتکل IP را دارد، با این ویژگی اضافی که بتوان بسته‌های IP را که همگی دارای آدرس مشترک هستند، بین پروسه‌های یک ماشین مالتی‌پلکس و دی‌مالتی‌پلکس کرد. از UDP می‌توان برای تعامل بین سرویس‌دهنده و مشتری بهره گرفت (مثلاً به کمک RPC). همچنین می‌توان از UDP برای ساختن پروتکل‌های بی‌درنگ همانند RTP استفاده کرد.

پروتکل اصلی لایه انتقال در اینترنت، TCP است. این پروتکل یک استریم از بایتها را (به صورت دو طرفه) بین دو پروسه ایجاد کرده و در اختیار می‌گذارد. به یک واحد اطلاعاتی که توسط TCP تولید می‌شود اصطلاحاً «قطعه» (Segment) گفته می‌شود. هر قطعه دارای یک سرآیند ۲۰ بایتی است. قطعات ارسالی ممکن است توسط مسیرپایه‌های اینترنت قطعه قطعه شوند فلذا ماشین میزبان باید آمادگی بازسازی آنها را داشته باشد. کارهای بسیار زیادی نیز برای بهینه سازی کارایی TCP صورت گرفته و بدین منظور از الگوریتم‌هایی نظیر «ناگل»، «کلارک»، «ژاکوبسن»، «کارن» استفاده شده است. لینک‌های بی‌سیم پیچیدگیهای متعددی را به TCP تحمیل می‌کنند. «TCP تراکنشی» (Transaction TCP) گونه توسعه یافته TCP است که تعامل سریع بین سرویس‌دهنده و مشتری و کاهش تعداد بسته‌ها را بر عهده دارد.

کارایی شبکه عموماً متأثر از سربار پروتکل و پردازش TPDU است و در سرعتهای بالا این وضعیت به مراتب بدتر می‌شود. پروتکلها بایستی به نحوی طراحی شوند که تعداد TPDUها، دفعات تعویض متن (Context Switch) و دفعات کپی‌برداری از TPDU را به حداقل برساند. برای شبکه‌های گیگابیتی، پروتکل‌های ساده مورد توجه هستند.

مسائل

۱. در مثالی که در شکل ۶-۲ در خصوص عملکردهای پایه انتقال (Primitives) ارائه کردیم، عمل LISTEN یک نوع فراخوانی متوقف‌کننده (Blocking) است. آیا این رفتار واقعاً لازم بوده است؛ اگر نه، شرح بدهید که چگونه می‌توان از یک تابع غیرمتوقف‌کننده (nonblocking) به جای آن استفاده کرد و این روش چه مزیتی

بر الگویی که در متن تشریح شد، دارد؟

۲. در مدل شکل ۶-۴ فرض بر آنست که احتمالاً برخی از بسته‌ها در لایه شبکه از بین می‌روند و هر بسته باید بطور مستقل اعلام وصول شود. فرض کنید که لایه شبکه صد در صد مطمئن و بدون خطاست و هیچ بسته‌ای از دست نمی‌رود. در این حالت چه تغییری در شکل ۶-۴ لازم است؟
۳. در هر دو بخش از کُد برنامه شکل ۶-۶ بدین نکته اشاره شده که مقدار `SERVER_PORT` باید در سرویس دهنده و مشتری یکسان باشد. چرا این موضوع اینقدر اهمیت دارد؟
۴. فرض کنید که برای تولید مقدار اولیه برای شماره ترتیب از روش مبتنی بر ساعت استفاده شده و شمارنده ساعت ۱۵ بیتی است. ساعت در هر ۱۰۰ میلی ثانیه یکبار تیک می‌زند و حداکثر طول عمر بسته‌ها ۶۰ ثانیه است. در چه مواقعی به هماهنگ‌سازی دوباره (Resynchronization) نیاز است: (پاسخ خود را برای موارد ذیل محاسبه کنید)

الف) در بدترین حالت

ب) وقتی که برای ارسال داده‌ها در هر دقیقه ۲۴۰ شماره ترتیب مصرف می‌شود.

۵. چه لزومی دارد که حداکثر طول عمر بسته‌ها یعنی T، باید آنقدر طولانی باشد تا مطمئن شویم که نه تنها بسته بلکه حتی پیغامهای اعلام وصول آنها (ACK) نیز از بین رفته‌اند؟
۶. تصور کنید که برای برقراری اتصال به جای روش «دست‌تکانی سه مرحله‌ای» از «دست‌تکانی دو مرحله‌ای» استفاده شده باشد. (به عبارت دیگر به پیام سوم نیازی نباشد.) آیا امکان بروز بن‌بست وجود دارد. یا نشان بدهید که بن‌بستی بوجود نمی‌آید یا مثالی از بن‌بست ارائه بدهید.
۷. مسئله «دو سپاه» را به صورت عمومی «n سپاه» (n-Army) در نظر بگیرید که فقط توافق دو سپاه آبی برای حمله، پیروزی آنها را تضمین می‌کند. آیا پروتکلی وجود دارد که براساس آن سپاه آبی به یقین پیروز میدان شود؟
۸. مشکل جبران از کارافتادگی یک ماشین میزبان و بازگرداندن آن به حالت طبیعی را مد نظر قرار بدهید؛ (شکل ۶-۱۸). هرگاه بتوان فاصله زمانی بین نوشتن داده‌ها در پروسه و ارسال پیغام اعلام وصول (ACK) یا برعکس را کم کرد، دو تا از بهترین استراتژیهای فرستنده و گیرنده که احتمال شکست پروتکل را به حداقل می‌رساند، کدامند؟
۹. آیا در واحد انتقال معرفی شده در شکل ۶-۲۰ امکان بروز بن‌بست وجود دارد؟
۱۰. شخصی که واحد انتقال شکل ۶-۲۰ را پیاده‌سازی کرده است تصمیم گرفته که در پروسه `sleep` یک شمارنده بگذارد تا در خصوص آرایه `conn` آمارگیری کند. از جمله این آمارها، تعداد اتصالهایی است که در هر یک از هفت حالت ممکن قرار دارد. ($n_i, i=1,2,\dots,7$) پس از نوشتن برنامه‌ای مفصل به زبان فرترن برای تحلیل داده‌های جمع‌آوری شده، برنامه‌نویس ما مشاهده می‌کند که رابطه $\sum n_i = \text{MAX_CONN}$ همیشه صادق است. آیا روابط این چنین دیگری (که حاصل ثابتی داشته باشند) براساس این هفت «متغیر حالت» وجود دارد؟ [حالات هفتگانه n_i را در شکل ۶-۲۱ مرور کنید. -م]
۱۱. اگر یک کاربر با استفاده از واحد انتقال نشان داده شده در شکل ۶-۲۰ یک پیام با طول صفر بفرستد چه اتفاقی می‌افتد؟
۱۲. تمام رخدادهایی را که ممکن است در واحد انتقال شکل ۶-۲۰ اتفاق بیفتد، در نظر بگیرید. حال بگویید که آیا این رخدادها هنگامی که کاربر در حالت `sending` متوقف مانده است باز هم معتبرند؟

۱۳. مزایا و معایب استفاده از روش مبتنی بر اعتبار را در مقایسه با پروتکل های پنجره لغزان تشریح کنید.
۱۴. چرا به وجود UDP نیاز است؟ آیا اگر پروسه های کاربری داده های خود را در درون بسته های IP جاسازی و ارسال کند کافی نیست؟
۱۵. یک پروتکل بسیار ساده لایه کاربرد را در نظر بگیرید که بر روی UDP کار می کند و امکان آنرا فراهم آورده تا مشتری بتواند یک فایل را از سرویس دهنده راه دور دریافت نماید. آدرس سرویس دهنده (یعنی آدرس IP و پورت) کاملاً مشخص است. مشتری ابتدا یک تقاضا حاوی نام فایل درخواستی برای سرویس دهنده می فرستد. سرویس دهنده در پاسخ دنباله ای از بسته های داده را که هر یک بخشی از فایل درخواستی را حمل می کنند، برای مشتری ارسال می دارد. برای اطمینان از صحت داده ها و همچنین حفظ ترتیب بسته های دریافتی، سرویس دهنده و مشتری از پروتکل «توقف و انتظار» (Stop & Wait) بهره گرفته اند. صرف نظر از مسئله کارایی، آیا مشکل دیگری در این پروتکل می بینید؟ به دقت در مورد امکان از کار افتادن (crashing) پروسه ها فکر کنید.
۱۶. یک برنامه مشتری، از طریق یک فیبرنوری به طول ۱۰۰ کیلومتر و سرعت یک گیگابیت بر ثانیه، تقاضایی ۱۲۸ بایتی (RPC) برای سرویس دهنده می فرستد. کارایی خط در خلال این فراخوانی از راه دور چقدر است؟
۱۷. وضعیت توصیف شده در مسئله قبلی را مد نظر قرار بدهید. حداقل زمان پاسخ سرویس دهنده را برای خطوط 1-Gbps و 1-Mbps محاسبه نمایید. چه نتیجه ای می توانید بگیرید؟
۱۸. UDP و TCP هر دو برای مشخص کردن پروسه تحویل گیرنده پیام از شماره پورت استفاده می کنند. دو دلیل بیاورید که چرا این دو پروتکل برای مشخص کردن پروسه ها، شناسه های جدیدی تعریف کرده اند (یعنی شماره پورتها) و از شناسه پروسه (Process ID) که از قبل [در هسته سیستم عامل] وجود دارد استفاده نمی کنند؟
۱۹. حداقل اندازه کل یک بسته TCP (شامل سر بار IP و TCP، بدون در نظر گرفتن سر بار لایه پیوند داده) چقدر است؟
۲۰. فرایند قطعه قطعه کردن دیتاگرام و بازسازی آنها بر عهده IP است و از دید TCP مخفی می ماند. آیا این قضیه بدین معنا تلقی می شود که TCP نباید نگران تحویل نامرتب داده ها باشد؟
۲۱. برای انتقال صدا با کیفیت CD، از پروتکل RTP استفاده می شود و باید در هر ثانیه ۴۴۱۰۰ جفت نمونه ۱۶ بیتی ارسال شود. (هر یک از این جفت نمونه ها برای یکی از کانالهای استریو است). RTP در هر ثانیه باید چند بسته ارسال کند؟
۲۲. آیا می توان کد اجرایی RTP را در کنار UDP درون هسته سیستم عامل قرار داد؟ پاسخ خود را شرح بدهید.
۲۳. به یک پروسه بر روی ماشین میزبان ۱ شماره پورت p انتساب داده شده است. همچنین به پروسه دیگری بر روی ماشین ۲، شماره پورت q منتسب شده است. آیا این امکان وجود دارد که در یک زمان دو یا چند اتصال TCP بین این دو پورت برقرار شود؟
۲۴. در شکل ۶-۲۹ می بینیم که به غیر از فیلد ۳۲ بیتی Acknowledgement، در چهارمین کلمه یک بیت دیگر با نام ACK وجود دارد. آیا این بیت واقعاً کاری انجام می دهد؟ چرا بله و چرا نه؟
۲۵. حداکثر طول فیلد حمل داده (Payload) در یک قطعه TCP، ۶۵۴۹۵ بایت است. این عدد عجیب از کجا آمده است؟

۲۶. در شکل ۶-۳۳، دو راه وارد شدن به حالت SYN RCVD را تشریح نمایید.
۲۷. اشکال بالقوه «الگوریتم ناگل» (Nagle) را در بکارگیری آن بر روی شبکه ای که شدیداً با مشکل ازدحام مواجه شده، تشریح کنید.
۲۸. تأثیر استفاده از «الگوریتم شروع آهسته» (Slow start algorithm) را بر روی خطی بدون ازدحام و با تأخیر رفت و برگشت ده میلی ثانیه، مد نظر قرار بدهید. پنجره دریافت 24KB و حداکثر طول هر قطعه 2KB است. چقدر طول می کشد تا به اندازه یک پنجره کامل (یعنی ۲۴KB)، داده ارسال شود؟
۲۹. فرض کنید که «پنجره ازدحام TCP» به 18KB تنظیم شده باشد و انقضای مهلت تایمر رخ بدهد. اگر چهار ارسال بعدی موفق باشد اندازه پنجره چقدر است؟ (فرض کنید که حداکثر طول قطعه 1KB می باشد).
۳۰. فرض نمایید زمان رفت و برگشت (یعنی RTT)، فعلاً ۳۰ میلی ثانیه است و سه پیغام ACK بعدی به ترتیب پس از ۲۶، ۳۲ و ۲۴ میلی ثانیه دریافت شوند. با فرض $\alpha = 0.9$ در الگوریتم ژاکوبسن، مقدار جدید RTT چقدر است؟
۳۱. یک ماشین مبتنی بر TCP، به اندازه یک پنجره ۶۵۵۳۵ بایتی بر روی یک کانال 1-Gbps، داده می فرستد. کانال فقط در یک جهت ده میلی ثانیه تأخیر دارد. حداکثر توان خروجی (Throughput) قابل حصول چقدر است؟ کارایی خط چقدر است؟
۳۲. فرض کنید حداکثر طول عمر بسته ها ۱۲۰ ثانیه است. یک ماشین میزبان حداکثر با چه سرعتی می تواند بسته های TCP با ۱۵۰۰ بایت داده را بفرستد در حالی که خطر تکرار شماره ترتیب تهدیدکننده نباشد؟ سربار ناشی از سرآیند TCP، IP و اترنت را هم در نظر بگیرید. فرض کنید که فریمهای اترنت را می توان پیایی و بی وقفه فرستاد.
۳۳. در یک شبکه اندازه حداکثر هر TPDU، ۱۲۸ بایت و حداکثر طول عمر آن ۳۰ ثانیه است. اگر شماره ترتیب هر TPDU هشت بیتی باشد، حداکثر نرخ ارسال در هر اتصال را محاسبه نمایید.
۳۴. فرض کنید زمان دریافت یک TPDU را اندازه گیری می کنید. وقتی یک وقفه رخ می دهد (وقفه دریافت بسته) شما بلافاصله ساعت سیستم را برحسب میلی ثانیه می خوانید. وقتی TPDU کاملاً پردازش شد مجدداً اقدام به خواندن ساعت می کنید. این کار را یک میلیون بار تکرار کرده اید و ۲۷۰۰۰۰ بار زمان را صفر و ۷۳۰۰۰۰ بار زمان را ۱ میلی ثانیه بدست آورده اید. با این اندازه گیریها زمان دریافت یک TPDU چقدر است؟
۳۵. یک CPU دستورالعملهای ماشین را با سرعت 1000-MIPS اجرا می کند. داده ها را می توان به صورت کلمات ۶۴ بیتی کپی کرد و کپی هر کلمه معادل زمان اجرای ۱۰ دستورالعمل طول می کشد. اگر هر بسته ورودی نیاز به چهار بار کپی برداری داشته باشد، آیا این سیستم قادر است از عهده یک خط 1-Gbps برآید؟ برای سادگی فرض کنید تمام دستورالعملهای ماشین حتی آنهایی که از حافظه می خوانند یا در آن می نویسند، همگی با سرعت 1000-MIPS اجرا شوند.
۳۶. برای رهایی از مشکل تکراری شدن شماره ترتیب بسته ها (در حالی که ممکن است بسته های قدیمی هنوز در زیر شبکه سرگردان باشند) می توان از شماره های ۶۴ بیتی استفاده کرد. با این وجود از دیدگاه تئوری می توان بکمک فیبرنوری به نرخ 75Tbps (75000 Gbps) رسید. حداکثر طول عمر بسته ها چقدر باشد که در شبکه های آینده با سرعت 75Tbps شماره های ترتیب ۶۴ بیتی تکرار نشوند؟ (فرض کنید که همانند TCP، هر بایت دارای یک شماره ترتیب است).

۳۷. یکی از مزایای استفاده از RPC بر روی UDP را نسبت به استفاده از «TCP تراکنشی» بیان کنید. یک مزیت T/TCP را نسبت به RPC عنوان نمایید.
۳۸. در شکل ۶-۴۰ الف مشاهده می کنید که برای تکمیل یک فراخوانی راه دور (RPC) به مبادله حداقل ۹ بسته نیاز است. آیا وضعیت دیگری وجود دارد که دقیقاً به ۱۰ بسته نیاز باشد؟
۳۹. در بخش ۶-۵ محاسبه کردیم که خطی یک گیگابیتی ۸۰۰۰۰ بسته در هر ثانیه به درون ماشین میزبان منتقل می کند. همچنین فرض کردیم که برای پردازش هر بسته فقط ۶۲۵۰ دستورالعمل اجرا گردد و نیم دیگر از توان پردازشی CPU برای کاربردهای دیگر کنار گذاشته شود. این محاسبات بر مبنای بسته های ۱۵۰۰ بایتی بود. همین محاسبات را برای بسته های ۱۲۸ بایتی (هم اندازه بسته های ARPANET) انجام بدهید. در هر دو حالت فرض را بر آن بگذارید که سربار کل در اندازه بسته لحاظ شده است. [یعنی نیازی به اضافه کردن طول سرآیندها به مقادیر ۱۲۸ یا ۱۵۰۰ نیست.]
۴۰. برای شبکه ای 1-Gbps که طول کانال آن ۴۰۰۰ کیلومتر است، عامل اصلی محدودیت تأخیر خط است نه پهنای باند. حال یک MAN را در نظر بگیرید که میانگین فاصله مبدا و مقصد، ۲۰ کیلومتر است. در چه نرخ ارسالی تأخیر رفت و برگشت (ناشی از سرعت نور) معادل با تأخیر انتقال یک بسته یک کیلوبایتی است؟ [تأخیر انتشار ناشی از سرعت نور است در حالیکه تأخیر انتقال متأثر از نرخ ارسال]
۴۱. حاصلضرب پهنای باند در تأخیر را برای شبکه های زیر محاسبه کنید: (۱) T1 (با نرخ 1.5Mbps) (۲) اترنت (10Mbps) T3 (۳) (45Mbps) STS-3 (۴) (155 Mbps). زمان رفت و برگشت یعنی RTT را ۱۰۰ میلی ثانیه فرض کنید. به خاطر داشته باشید که در سرآیند TCP فقط ۱۶ بیت جهت اعلام طول پنجره رزرو شده است. از محاسبات خود به چه نتیجه ای می رسید؟
۴۲. حاصلضرب پهنای باند در تأخیر یک کانال ماهواره ای همگام با زمین (ماهواره نوع GEO) چقدر است؟ اگر تمام بسته ها با احتساب سربار، ۱۵۰۰ بایتی باشند اندازه فیلد پنجره در هر بسته باید چقدر باشد؟
۴۳. سرویس دهنده فایل نشان داده شده در شکل ۶-۶ بسیار ضعیف عمل می کند و می توان بهبودهایی را در آن ایجاد کرد. اصلاحات زیر را در آن اعمال نمایید:
- الف) به برنامه مشتری آرگومان سومی اضافه کنید که محدوده بایتهایی که باید ارسال شوند را مشخص کند.
- ب) به برنامه مشتری آرگومان پرچم w- را اضافه کنید تا بتوان فایلی را بر روی سرویس دهنده نوشت.
۴۴. برنامه شکل ۶-۲۰ را به نحوی اصلاح کنید که عملیات جبران خطا (Error Recovery) را نیز انجام بدهد. یک بسته نوع جدید به نام reset تعریف کنید که فقط زمانی دریافت می شود که طرفین اقدام به برقراری اتصال کرده باشند ولی به هر دلیلی هیچیک از طرفین آن را قطع نکرده باشند. این رخداد که بطور همزمان در طرفین یک اتصال بروز می کند بدین معناست تمام بسته هایی که قبلاً ارسال شده اند یا دریافت گردیده اند و یا از بین رفته اند ولیکن به هر حال در زیر شبکه نیستند.
۴۵. برنامه ای بنویسید که به جای سیستم مبتنی بر اعتبار (credit) که در واحد انتقال شکل ۶-۲۰ از آن استفاده شد، مدیریت بافرها را مبتنی بر پروتکل پنجره لغزان (برای کنترل جریان) شبیه سازی کند. باید پروسه های لایه بالاتر بتوانند اتصالاتی را باز کنند، داده بفرستند و نهایتاً اتصال را ببندند. برای ساده شدن برنامه فرض کنید که کل داده ها در یک جهت و از ماشین A به ماشین B جریان دارند. در برنامه خود استراتژیهای مختلف تخصیص بافر را در ماشین B پیازمانید، مثلاً روش تخصیص پیشاپیش بافر برای هر اتصال را با روش

معمولی بافرسازی (یعنی استفاده از یک بافر بزرگ) مقایسه کرده و توان خروجی آن دو مقایسه کنید.

۴۶. یک سیستم گپ زنی (chat) طراحی کنید که امکان گفتگو و محاوره چندین گروه از کاربران را فراهم کند. این سیستم از سه بخش تشکیل شده است: (۱) هماهنگ کننده گفتگو (Chat Coordinator) (۲) سرویس دهنده گفتگو (Chat Server) (۳) برنامه مشتری (Chat Client). سیستم هماهنگ کننده گفتگو در آدرس شناخته شده ای از شبکه قرار گرفته است. سیستم هماهنگ کننده با برنامه های مشتری به روش UDP تبادل داده می کند و برای هر نشستی که ایجاد می شود یک سرویس دهنده گفتگو (Chat Server) تنظیم می کند. یعنی به ازای هر نشست یک سرویس دهنده گفتگو وجود دارد. [یک نشست را مجموعه ای از کاربران در نظر بگیرید که حول یک موضوع خاص گفتگو می کنند.] از دیگر وظائف سیستم هماهنگ کننده، نگهداری یک دایرکتوری از نشستهای موجود است. سرویس دهنده گفتگو برای ارتباط با هر مشتری آن نشست، از TCP بهره می گیرد. برنامه مشتری نیز به کاربران اجازه می دهد که یک نشست ایجاد کنند، به یک نشست موجود بپیوندند یا یک نشست را ترک کنند. کد برنامه سیستم هماهنگ کننده، سرویس دهنده و مشتری را طراحی و پیاده سازی نمایید.

لایه کاربرد



بعد از به سرانجام رساندن همه مقدمات، اکنون به لایه ای رسیده ایم که تمام کاربردهای شبکه در آن قرار دارد: لایه کاربرد (application layer). لایه های زیرین لایه کاربرد فقط برای سرویس دادن به این لایه هستند، و هیچ کار واقعی برای کاربران انجام نمی دهند. در این فصل با کاربردهای واقعی شبکه آشنا خواهیم شد. با این حال در لایه کاربرد هم به پروتکل های پشتیبانی کننده، پروتکل هایی که بار وظایف را بر گردن می گیرند، نیاز داریم. به همین دلیل، برای شروع یکی از این پروتکل ها را بررسی خواهیم کرد. این پروتکل که نام آن DNS است، نامگذاری در اینترنت را بر عهده دارد. پس از آن، سه تا از کاربردهای واقعی شبکه را مورد بررسی قرار خواهیم داد: پست الکترونیک (ایمیل)، تارنمای جهانی (یا باختصار، وب)، و چند رسانه ای.

۱-۷ سیستم نام ناحیه - DNS

با اینکه از نظر تئوری برنامه ها می توانند برای تماس با کامپیوترها، صندوق های پستی و منابع دیگر از آدرس شبکه (مثلاً، IP) آنها استفاده کنند، حفظ کردن این قبیل آدرسها برای افراد دشوار است. همچنین اگر آدرس ایمیل «حسن» hasan@128.111.24.41 باشد، و ISP یا سازمان متبوع وی تصمیم بگیرد کامپیوتر سرویس دهنده پست الکترونیک خود را به آدرس IP دیگری منتقل کند، آدرس ایمیل «حسن» نیز عوض خواهد شد. به همین دلیل برای تفکیک نام ماشینها از آدرس آنها، استفاده از نامهای معمولی باب شد. به این ترتیب، آدرس ای بل «حسن» چیزی شبیه hasan@art.ucsb.edu خواهد شد. با این وجود، کامپیوترها فقط آدرسهای عددی را می فهمند، پس باید مکانیزمی برای تبدیل اسامی معمولی به آدرسهای شبکه فراهم کنیم. در این قسمت روش تبدیل نامهای معمولی به آدرس عددی را در اینترنت بررسی خواهیم کرد.

سالها قبل در آرپانت فایلی وجود داشت بنام hosts.txt، که نام کامپیوترها و آدرس IP آنها در این فایل لیست می شد. کامپیوترهای شبکه هر شب این فایل را از جایی که قرار داشت، می خواندند و خود را به روز می کردند. برای شبکه ای با دهها (و یا صدها) کامپیوتر این روش بخوبی کار می کرد.

ولی وقتی تعداد کامپیوترهای شبکه از مرز هزاران PC و کامپیوتر بزرگ گذشت، همه دریافتند که این روش دیگر جوابگو نیست. اولین دلیل آن بود که اندازه چنین فایلی بشدت بزرگ می شد، ولی از آن مهمتر مشکل نامهای تکراری بود که ضرورت یک مدیریت مرکزی را اجتناب ناپذیر می کرد (چیزی که بزرگی و بار شبکه آنها ناممکن می کرد). برای غلبه بر این مشکلات بود که DNS (سیستم نام ناحیه - Domain Name System) اختراع شد.

ایده اصلی DNS یک روش نامگذاری سلسله مراتبی بر اساس ناحیه ها بود، که بصورت یک پایگاه اطلاعاتی

توزیع یافته پیاده سازی می شد. هدف اولیه این سیستم تبدیل نام کامپیوترها و آدرسهای ایمیل (پست الکترونیک) به آدرسهای IP بود، ولی می توانست کاربردهای دیگری هم داشته باشد. DNS در RFC 1034 و RFC 1035 تعریف شده است.

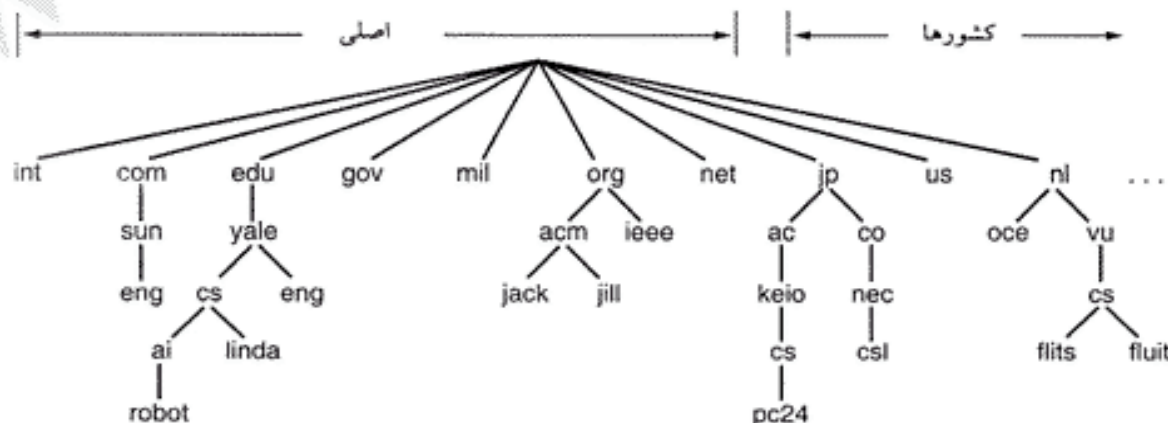
روش کار DNS خیلی خلاصه چنین است: برای تبدیل یک نام به آدرس IP، برنامه یک تابع کتابخانه ای بنام تبدیل کننده (resolver) را فراخوانی می کند، و نام مورد نظر را بصورت پارامتر به آن می دهد. تابع *gethostbyname* در شکل ۶-۶ نمونه ای از یک تبدیل کننده است. تبدیل کننده یک بسته UDP به سرویس دهنده DNS محلی می فرستد، که این DNS آدرس IP معادل نام خواسته شده را یافته و به تبدیل کننده برمی گرداند، که آن هم بنوبه خود آدرس را به برنامه فراخوانی کننده تحویل می دهد. برنامه هم پس از بدست آوردن آدرس IP کامپیوتر مقصد، می تواند با آن ارتباط TCP برقرار کرده یا بسته های UDP به آن بفرستد.

۱-۷ فضای نام DNS

مدیریت مجموعه ای بزرگ و دائماً در تغییر از نامها بهیچوجه کار ساده ای نیست. در سیستم پست، مدیریت نامها از طریق اجبار افراد به نوشتن نام کشور، استان (یا ایالت)، شهر، خیابان و شماره پلاک مقصد انجام می شود. با این روش دیگر «پلاک ۷، خیابان آزادی، تهران» هرگز با «پلاک ۷، خیابان آزادی، اصفهان» اشتباه نخواهد شد. DNS هم به همین روش کار می کند.

اینترنت به بیش از ۲۰۰ ناحیه سطح بالا (top-level domain) (که هر کدام تعداد زیادی کامپیوتر را در برمی گیرند) تقسیم شده است. هر ناحیه به چندین زیرناحیه (subdomain)، و آنها نیز بنوبه خود به زیرناحیه های کوچکتر، تقسیم می شوند. این سلسله مراتب را می توان بصورت یک درخت نمایش داد (شکل ۷-۱ را ببینید). ناحیه هایی که زیرناحیه ندارند، برگهای این درخت را تشکیل می دهند. هر یک از این برگها می تواند یک کامپیوتر، یک شبکه کوچک (با چند کامپیوتر)، و یا شرکتی بزرگ (با هزاران کامپیوتر) باشد.

ناحیه های سطح بالا بر دو گونه اند: عمومی و کشورها. ناحیه های عمومی اولیه عبارت بودند از: *com* (مخفف commercial، تجاری)، *edu* (مخفف educational، مؤسسات آموزشی)، *gov* (مخفف government، ادارات دولتی)، *int* (مخفف international، مؤسسات بین المللی)، *mil* (مخفف military، نظامی)، *net* (مخفف network provider، شرکتهای خدمات شبکه و اینترنت)، و *org* (مخفف organization، مؤسسات غیرانتفاعی). هر کشور نیز دارای یک ناحیه خاص (در ناحیه کشورها) است، که در استاندارد ISO 3166 تعریف شده است.



شکل ۷-۱. بخشی از فضای نام ناحیه در اینترنت.

در نوامبر ۲۰۰۰، ICANN چهار ناحیه سطح بالای جدید را برای مصارف عمومی تصویب کرد: *biz* (مخفف *businesses*، مشاغل)، *info* (مخفف *information*، شرکتهای اطلاعاتی)، *name* (نام افراد)، و *pro* (مخفف *profession*، صاحبان حرف مانند وکلا و پزشکان). علاوه بر آن، سه ناحیه سطح بالای تخصصی نیز معرفی شد، که برخی از صنایع خاص می توانند از آنها استفاده کنند؛ این سه ناحیه عبارتند از: *aero* (مخفف *aerospace*، صنایع هوا-فضا)، *coop* (مخفف *co-operatives*، تعاونی ها)، و *museum* (موزه ها). به احتمال زیاد در آینده ناحیه های سطح بالای دیگری نیز اضافه خواهند شد.

از طرف دیگر هر چه اینترنت بیشتر تجاری می شود، بحث و جدل هم بالاتر می گیرد. مثلاً همین ناحیه *pro* را در نظر بگیرید. این ناحیه برای صاحبان حرف که صلاحیت آنها تأیید شده باشد، در نظر گرفته شده است. اما حرفه ای کیست؟ و چه کسی باید صلاحیت ها را تأیید کند؟ یک پزشک یا وکیل مسلماً حرفه ایست، اما یک عکاس، معلم پیانو، شعبده باز، لوله کش، آرایشگر، خالکوب، آدمکش حرفه ای و یا فاحشه چطور؟ آیا اینها هم حرفه اند، و شایسته دریافت ناحیه *pro*؟ و اگر پاسخ مثبت است، چه کسی صلاحیت داوطلبان را تأیید می کند؟

در کل، گرفتن ناحیه سطح دوم در یک ناحیه سطح بالا، مانند *name-of-company.com*، ساده است: تنها کاری که باید کرد مراجعه به پایگاه اطلاعاتی ناحیه سطح بالا (در اینجا *com*) و اطمینان از آزاد بودن نام مورد نظر است. اگر مشکلی وجود نداشته باشد، درخواست کننده پول کمی بابت هزینه سالیانه پرداخته، و آن نام را بدست می آورد. می توان به جرأت گفت که، اکنون تمام نامهای بامعنای انگلیسی در ناحیه *com* گرفته شده اند (اگر باور ندارید، امتحان کنید!).

نام هر ناحیه بصورت مسیری رو به بالا و به سمت یک ریشه (که نامی ندارد) مشخص می شود. اجزای این نام با نقطه (که «دات» تلفظ می شود) از هم جدا می شوند. برای مثال، نام ناحیه بخش مهندسی شرکت سان میکروسیستمز می تواند *eng.sun.com* باشد (در حالیکه همین نام در سیستم عامل یونیکس بصورت *com/sun/eng* نوشته می شود). دقت کنید که با این روش نامگذاری دیگر نام ناحیه بخش مهندسی سان میکروسیستمز با دانشکده مهندسی دانشگاه ییل (که مثلاً *eng.yale.edu* است) اشتباه نخواهد شد.

نام ناحیه می تواند مطلق (*absolute*) یا نسبی (*relative*) باشد. یک نام مطلق همیشه به نقطه ختم می شود (مانند *eng.sun.com*)، در حالیکه نامهای نسبی چنین نیستند. نامهای نسبی بدون توجه به جایی که بکار رفته اند، معنی نمی دهند. در هر دو حالت، یک نام ناحیه به گرهی خاص در درخت نامها (و تمام گرههای ذیل آن) اشاره می کند.

کوچکی یا بزرگی حروف در نام ناحیه بی تأثیر است، بعبارت دیگر *edu*، *Edu* و *EDU* همگی یک معنی می دهند. هر جزء از نام ناحیه حداکثر ۶۳ حرف، و کل مسیر نام ناحیه حداکثر ۲۵۵ حرف می توانند داشته باشند. برای قرار دادن یک ناحیه در درخت نامهای ناحیه دو روش وجود دارد. برای مثال، ناحیه *cs.yale.edu* می تواند در ذیل شاخه کشور آمریکا (*us*) و بصورت *cs.yale.ct.us* هم ثبت شود. در کل، اغلب سازمانها و شرکتهای در آمریکا ترجیح می دهند از ناحیه های عمومی استفاده کنند، در حالیکه در خارج از آمریکا بیشتر از نام کشور بعنوان ناحیه سطح بالا استفاده می شود. هیچ معنی برای ثبت نام ناحیه در شاخه های متعدد وجود ندارد، با این حال چنین گرایشی بجز در شرکتهای چندملیتی (مانند شرکت سونی که ناحیه های *sony.com* و *sony.nl* را ثبت کرده) وجود ندارد.

هر ناحیه ناحیه های ذیل خود را کنترل می کند. برای مثال، کشور ژاپن دارای ناحیه های *ac.jp* و *co.jp* است، که برترتیب مشابه *edu* و *com* هستند؛ در حالیکه در هلند چنین تقسیم بندی وجود ندارد، و تمام ناحیه ها ذیل *nl* قرار دارند. برای مثال، ناحیه های زیر همگی دانشکده های کامپیوتر در کشور مربوطه هستند:

۱. cs.yale.edu (دانشگاه ییل، ایالات متحده آمریکا)

۲. cs.vu.nl (دانشگاه فریژه، هلند)

۳. cs.keio.ac.jp (دانشگاه کی یو، ژاپن)

برای ایجاد یک زیر ناحیه، مجوز ناحیه بالاتر مورد نیاز است. برای مثال اگر گروه VLSI در دانشکده کامپیوتر دانشگاه ییل تأسیس شود، و بخواهد به نام `vlsi.cs.yale.edu` شناخته شود، باید مجوزهای لازم را از مسئول `cs.yale.edu` کسب کند. بهمین ترتیب اگر دانشگاه جدیدی، مثلاً دانشگاه شمالی داکوتای جنوبی، تأسیس شود، و بخواهد ناحیه `unsd.edu` را برای خود ثبت کند، باید هماهنگیهای لازم را با مسئول ناحیه `edu` بعمل آورد. قرار دادن مسئولیت زیر ناحیه ها بر عهده ناحیه بالاتر باعث می شود تا هیچ دو ناحیه ای هم نام نشوند. همین که یک ناحیه ایجاد شد (مانند، `unsd.edu`)، دیگر می تواند بدون نظارت ناحیه های بالاتر به ایجاد زیر ناحیه های دلخواه خود (مثلاً، `cs.unsd.edu`) بپردازد.

نامگذاری ناحیه ها امری سازمانی است نه فیزیکی. برای مثال، دانشکده های کامپیوتر و مهندسی برق یک دانشگاه می توانند ناحیه های کاملاً مستقلی داشته باشند، حتی اگر در یک ساختمان باشند و از شبکه LAN واحدی استفاده کنند. از طرف دیگر، بخشهای مختلف یک دانشکده به یک ناحیه تعلق دارند، حتی اگر از نظر فیزیکی از هم جدا باشند.

۲-۱-۷ رکوردهای منابع

هر ناحیه، خواه ناحیه ای سطح بالا یا ناحیه ای با یک کامپیوتر، دارای تعدادی رکورد منابع (resource record) است. برای یک کامپیوتر، متداولترین رکورد منبع آدرس IP آن است، اما انواع دیگری از رکوردهای منابع می تواند وجود داشته باشد. وقتی یک تبدیل کننده نام ناحیه را به DNS می دهد، چیزی که دریافت می کند تمام رکوردهای منابع وابسته به آن نام است. بنابراین، اصلی ترین وظیفه یک DNS تبدیل نام ناحیه به رکوردهای منابع است. هر رکورد منبع پنج بخش دارد. با اینکه رکوردهای منابع را می توان برای کارایی بهتر بصورت باینری در آورد، ولی در اغلب مواقع این رکوردها بصورت متنی (ASCII) - یک رکورد در هر خط - نگهداری می شوند. فرمت یک رکورد منبع مانند زیر است:

Domain_name	Time_to_live	Class	Type	Value
-------------	--------------	-------	------	-------

که در آن `Domain_name` نام ناحیه ایست که این رکورد متعلق به آن است. معمولاً هر ناحیه تعداد زیادی رکورد دارد، و در هر پایگاه داده اطلاعات چندین ناحیه نگهداری می شود. در نتیجه این فیلد کلید اصلی جستجو در پایگاه داده DNS است. (ترتیب قرار گرفتن رکوردها در پایگاه داده اهمیتی ندارد).

فیلد `Time_to_live` مشخص می کند که این رکورد چقدر دوام می آورد. این فیلد در رکوردهای با دوام مقدار زیادی دارد، مثلاً ۸۶۴۰۰ (تعداد ثانیه های یک شبانه روز)؛ و بر عکس، اطلاعات کم دوام عمر کوتاهی دارد، مثلاً ۶۰ (یک دقیقه). در بحث حافظه نهان به این موضوع برخواهیم گشت.

فیلد سوم هر رکورد منبع `Class` است. برای اطلاعات اینترنتی این فیلد همیشه `IN` است؛ برای اطلاعات غیر اینترنتی از کدهای دیگری هم می توان استفاده کرد (که البته بندرت دیده می شوند). فیلد `Type` نوع رکورد منبع را مشخص می کند. مهمترین انواع رکوردهای منابع را در شکل ۲-۷ ملاحظه می کنید.

رکورد `SOA` نام منبع اصلی اطلاعات منطقه (zone)، آدرس ایمیل سرپرست ناحیه، شماره سریال منحصر به فرد آن، و مشخصات دیگر ناحیه را مشخص می کند.

نوع	مفهوم	مقدار
SOA	Start of Authority	پارامترهای منطقه
A	IP address of a host	عدد صحیح ۳۲ بیتی
MX	Mail exchange	تقدم دریافت ایمیل
NS	Name Server	نام سرویس دهنده ناحیه
CNAME	Canonical name	نام ناحیه
PTR	Pointer	نام مستعار برای آدرس IP
HINFO	Host description	مشخصات CPU و سیستم عامل
TXT	Text	متن تفسیر نشده

شکل ۷-۲. انواع رکوردهای منابع اصلی DNS برای IPv4.

مهمترین نوع رکورد منبع، رکورد A (آدرس) است. هر رکورد A یک آدرس IP ۳۲ بیتی را در خود نگه می‌دارد. هر کامپیوتر اینترنت باید حداقل یک آدرس IP داشته باشد، تا کامپیوترهای دیگر بتوانند با آن تماس بگیرند. برخی از کامپیوترها دو یا چند آدرس IP دارند، که برای هر آدرس IP چنین کامپیوتری باید یک رکورد A وجود داشته باشد. DNS را می‌توان بگونه‌ای پیکربندی کرد که در میان این رکوردها بچرخد، یعنی برای اولین درخواست اولین رکورد را برگرداند، برای درخواست دوم دومین رکورد را، و الی آخر.

رکورد مهم بعدی رکورد MX است. این رکورد آدرس سرویس دهنده پست الکترونیک (ایمیل) ناحیه را مشخص می‌کند. اختصاص یک نوع رکورد خاص به سرویس دهنده پست الکترونیک بدین خاطر است که تمام کامپیوترها چنین قابلیتی (دریافت ایمیل) ندارند. برای مثال، اگر کسی بخواهد به `bill@microsoft.com` ایمیل بفرستد، باید آدرس سرویس دهنده پست الکترونیک `microsoft.com` را پیدا کند. این اطلاعات را رکورد MX عرضه می‌کند.

رکورد NS سرویس دهنده نام (name server) را مشخص می‌کند. برای مثال، معمولاً هر پایگاه داده DNS یک رکورد NS برای ناحیه‌های سطح بالا دارد. (باز هم به این موضوع برمی‌گردیم.)

از رکورد CNAME می‌توان برای ایجاد نامهای مستعار (alias) استفاده کرد. برای مثال، فرض کنید دوستی بنام پاول در دانشکده مهندسی کامپیوتر دانشگاه MIT دارید، و می‌خواهید برای وی ایمیل بفرستید. فقط می‌دانید که نام وی در شبکه این دانشگاه paul است، اما آدرس ایمیل کامل او را ندارید. حدس می‌زنید که ناحیه دانشکده مزبور `cs.mit.edu` باشد، اما مسئولین این دانشکده (شاید از روی کج سلیقه‌گی) نام `lcs.mit.edu` را برای ناحیه خود را انتخاب کرده‌اند. اگر ایمیلی به آدرس `paul@cs.mit.edu` بفرستید، مسلماً برگشت خواهد خورد؛ ولی اگر مسئولین این دانشکده کمی هوشیاری بخرج دهند، با تعریف یک نام مستعار می‌توانند تا حدی اشتباه خود را جبران کنند - برای این منظور می‌توان از یک رکورد CNAME مانند زیر استفاده کرد:

```
cs.mit.edu      86400      IN      CNAME    lcs.mit.edu
```

رکورد PTR هم، مانند CNAME، به یک نام دیگر اشاره می‌کند. اما برخلاف CNAME (که در واقع یک ماکرو است)، رکورد PTR یک نوع داده معمولی DNS است که تفسیر آن به محتویات این رکورد بستگی دارد. در عمل، تقریباً همیشه از این رکورد برای جستجوی معکوس (reverse lookup) - تبدیل آدرس IP به نام ماشین - استفاده می‌شود.

رکورد HINFO اطلاعات مربوط به نوع ماشین و سیستم عامل آنرا برمی‌گرداند. از رکورد TXT هم می‌توان

برای برگرداندن اطلاعات متنی اضافی به کاربران استفاده کرد. رکوردهای *HINFO* و *TXT* فقط برای راحتی کاربران تعبیه شده اند، و الزامی نیستند. اغلب اوقات چنین اطلاعاتی وجود ندارد، و اگر هم وجود داشته باشد، نمی توان به آن کاملاً اطمینان کرد.

آخرین فیلد رکورد منبع، فیلد *Value* (مقدار) است. این فیلد می تواند یک عدد، نام ناحیه، یا یک رشته متنی باشد. طرز وارد کردن این مقدار به نوع آن بستگی دارد (در شکل ۷-۲ توضیح کوتاهی درباره نوع هر فیلد آورده شده است).

برای آشنایی بیشتر با اطلاعاتی که در پایگاه داده *DNS* یک ناحیه می توان یافت، شکل ۷-۳ را ببینید. در این شکل قسمتی از پایگاه داده نیمه فرضی ناحیه ای بنام *cs.vu.nl* (شکل ۷-۱) نشان داده شده است. در این پایگاه داده هفت نوع رکورد منبع وجود دارد.

اولین خط غیر توضیحی شکل ۷-۳ مقداری اطلاعات اولیه درباره ناحیه *cs.vu.nl* می دهد، که فعلاً با آنها کاری نداریم. دو خط بعدی مقداری اطلاعات متنی درباره این ناحیه (و محل آن) می دهند. پس از آن دو کامپیوتری که مسئول دریافت ایمیل های ناحیه *cs.vu.nl* هستند، مشخص شده اند. اولین کامپیوتری که ایمیل ها باید به آن فرستاده شوند، *zephyr* نام دارد؛ و اگر *zephyr* جواب نداد، نوبت به *top* می رسد.

بعد از یک خط خالی (که فقط برای خوانا تر کردن فایل است)، رکوردهایی آمده اند که می گویند *flits* یک کامپیوتر *Sun* با سیستم عامل *UNIX* است، و دو آدرس *IP* دارد (130.37.16.112 و 192.31.231.165). پس از آن سه ماشین برای دریافت ایمیل های *flits.cs.vu.nl* مشخص شده است: اولین آنها طبیعتاً خود *flits* است، ولی

; Authoritative data for *cs.vu.nl*

<i>cs.vu.nl.</i>	86400	IN	SOA	star boss (952771,7200,7200,2419200,86400)
<i>cs.vu.nl.</i>	86400	IN	TXT	"Divisie Wiskunde en Informatica."
<i>cs.vu.nl.</i>	86400	IN	TXT	"Vrije Universiteit Amsterdam."
<i>cs.vu.nl.</i>	86400	IN	MX	1 <i>zephyr.cs.vu.nl.</i>
<i>cs.vu.nl.</i>	86400	IN	MX	2 <i>top.cs.vu.nl.</i>
<i>flits.cs.vu.nl.</i>	86400	IN	HINFO	<i>Sun Unix</i>
<i>flits.cs.vu.nl.</i>	86400	IN	A	130.37.16.112
<i>flits.cs.vu.nl.</i>	86400	IN	A	192.31.231.165
<i>flits.cs.vu.nl.</i>	86400	IN	MX	1 <i>flits.cs.vu.nl.</i>
<i>flits.cs.vu.nl.</i>	86400	IN	MX	2 <i>zephyr.cs.vu.nl.</i>
<i>flits.cs.vu.nl.</i>	86400	IN	MX	3 <i>top.cs.vu.nl.</i>
<i>www.cs.vu.nl.</i>	86400	IN	CNAME	<i>star.cs.vu.nl</i>
<i>ftp.cs.vu.nl.</i>	86400	IN	CNAME	<i>zephyr.cs.vu.nl</i>

<i>rowboat</i>	IN	A	130.37.56.201
	IN	MX	1 <i>rowboat</i>
	IN	MX	2 <i>zephyr</i>
	IN	HINFO	<i>Sun Unix</i>

<i>little-sister</i>	IN	A	130.37.62.23
	IN	HINFO	<i>Mac MacOS</i>

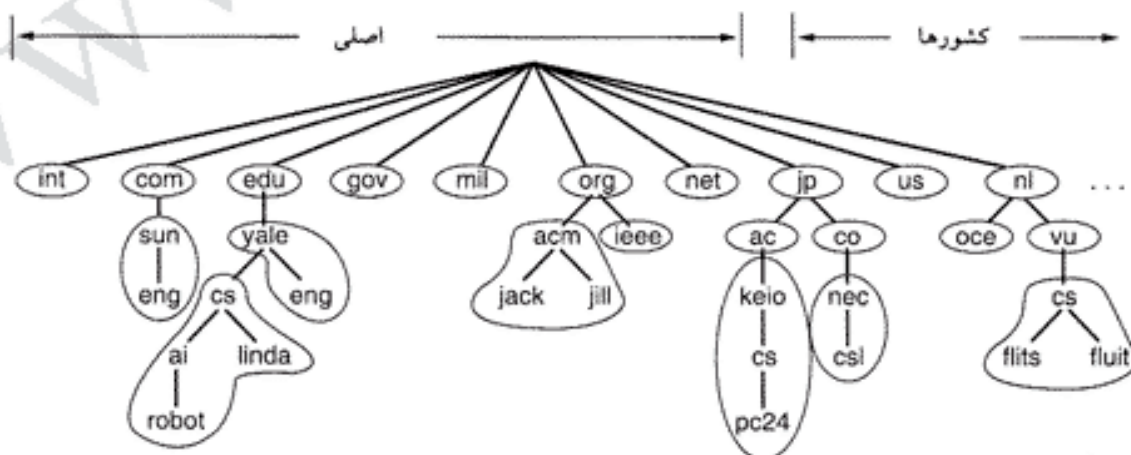
<i>laserjet</i>	IN	A	192.31.231.216
	IN	HINFO	"HP Laserjet IIISi" Proprietary

شکل ۷-۳. قسمتی از پایگاه داده *DNS* در ناحیه *cs.vu.nl*.

اگر این ماشین خاموش بود، *zephyr* و *top* گزینه‌های بعدی خواهند بود. بعد از آن یک نام مستعار برای ماشین *star.cs.vu.nl* تعریف شده است: *www.cs.vu.nl*. با تعریف این نام مستعار، کاربران می‌توانند بدون دغدغه تغییر آدرس صفحه وب *cs.vu.nl* به آن مراجعه کنند. همین کار برای *ftp.cs.vu.nl* نیز انجام شده است. در چهار خط بعدی رکوردهای منبع کامپیوتری بنام *rowboat.cs.vu.nl* تعریف شده‌اند. این اطلاعات شامل آدرس IP، محل دریافت ایمیل (اولیه و ثانویه)، و اطلاعاتی درباره خود ماشین است. پس از آن یک کامپیوتر MAC (بدون قابلیت دریافت ایمیل)، و بدنبال آن یک چاپگر لیزری که به اینترنت متصل است، تعریف شده‌اند. چیزی که در اینجا نشان داده نشده (و در واقع در این فایل هم نیست)، آدرس IP ناحیه‌های سطح بالا است. از آنجائیکه این کار در حوزه مسئولیت ناحیه *cs.vu.nl* نیست، در این فایل هم رکوردی برای آن وجود ندارد. این قبیل اطلاعات را کامپیوترهایی بنام سرویس‌دهنده‌ریشه (root server) ارائه می‌کنند، که با هر بار اجرای سرویس‌دهنده DNS آدرس آنها در حافظه حافظه نهان DNS بار می‌شود. تعداد سرویس‌دهنده‌های ریشه در حدود ۱۲ تاست که در سراسر دنیا پراکنده‌اند، و آدرس IP تمام ناحیه‌های سطح بالا را دارند. بنابراین، اگر ماشینی آدرس IP حداقل یکی از این سرویس‌دهنده‌های ریشه را داشته باشد، می‌تواند نام هر ناحیه‌ای را پیدا کند.

۳-۱-۷ سرویس‌دهنده نام

از نظر تنوری، برای نگهداری تمام اطلاعات DNS و پاسخ دادن به درخواست‌ها یک سرویس‌دهنده DNS کافیست. اما در عمل، بار کاری چنین کامپیوتری آنقدر سنگین خواهد شد که عملاً آنرا بلااستفاده می‌کند. علاوه بر آن، اگر این کامپیوتر از کار بیفتد، تمام اینترنت هم با آن به خواب خواهد رفت. برای اجتناب از چنین وضعیتی، فضای نام DNS به چندین منطقه (zone) با مرزهای مشخص و غیرمشترک تقسیم شده است. در شکل ۷-۴ یکی از راههای تقسیم فضای نام شکل ۷-۱ را مشاهده می‌کنید. هر منطقه شامل بخشی از درخت DNS است، و سرویس‌دهنده‌های نام آنرا در خود نگه می‌دارد. معمولاً هر منطقه دارای یک سرویس‌دهنده نام (name server) اولیه است که اطلاعاتش را از فایلی روی دیسک خود می‌گیرد، و یک یا چند سرویس‌دهنده نام ثانویه نیز دارد که آنها اطلاعات خود را از سرویس‌دهنده نام اولیه می‌گیرند. برای بالا بردن ضریب اطمینان، می‌توان تعدادی از سرویس‌دهنده‌های نام یک منطقه را خارج از آن منطقه مستقر کرد.



شکل ۷-۴. فضای نام DNS به منطقه‌های مختلف تقسیم شده است.

تعیین مرزهای یک منطقه بر عهده سرپرست آن است. تصمیم‌گیری در این باره تا حد زیادی به تعداد سرویس‌دهنده‌های نام منطقه و محل استقرار آنها بستگی دارد. برای مثال، در شکل ۷-۴، دانشگاه پیل دارای یک

سرویس دهنده نام برای ناحیه های *yale.edu* و *eng.yale.edu* است، اما ناحیه *cs.yale.edu* در منطقه دیگری قرار دارد. چنین تصمیماتی بیشتر به تمایل ناحیه ها برای کنترل مستقیم منطقه خود بستگی دارد. در مثال فوق، ناحیه *cs.yale.edu* منطقه ای مستقل است، در حالی که *eng.yale.edu* چنین نیست.

وقتی یک تبدیل کننده می خواهد آدرس ناحیه ای را بداند، ابتدا درخواست خود را به سرویس دهنده های نام محلی خود می دهد. اگر این ناحیه در محدوده قانونی سرویس دهنده نام مزبور بود (مانند *ai.cs.yale.edu* که در قلمرو *cs.yale.edu* است)، سرویس دهنده نام رکوردهای منبع معتبر را به آن برمی گرداند. یک رکورد معتبر (authoritative record) رکوردی است که مستقیماً از سرپرست ناحیه منشأ می گیرد، و بنابراین همیشه صحیح و معتبر است (بر خلاف رکوردهای ذخیره شده - cached record - که تاریخ اعتبار آنها می تواند منقضی شده باشد). ولی اگر آن ناحیه در قلمرو سرویس دهنده های محلی نباشد، سرویس دهنده نام این درخواست را به سرویس دهنده نام سطح بالای ناحیه مزبور می فرستد. برای روشنتر شدن مطلب، به مثالی در شکل ۵-۷ توجه کنید. در اینجا یک تبدیل کننده در ناحیه *flits.cs.vu.nl* می خواهد آدرس IP ماشین سرویس دهنده ناحیه *linda.cs.yale.edu* را بداند. در مرحله ۱، این تبدیل کننده درخواست خود را به سرویس دهنده نام محلی، یعنی *cs.vu.nl*، می فرستد. این درخواست شامل نام ناحیه مورد نظر (رکوردهای نوع *A* و کلاس *IN*) می باشد.



شکل ۵-۷. مراحل جستجوی نام ناحیه.

اجازه دهید فرض کنیم سرویس دهنده نام محلی تا بحال چیزی از ناحیه *linda.cs.yale.edu* نشنیده و درباره آن هیچ اطلاعاتی ندارد. این سرویس دهنده می تواند از همسایه های خود در این باره پرس و جو کند، ولی اگر آنها هم بی اطلاع بودند، یک بسته UDP به سرویس دهنده نام *edu-server.net* (که آدرس آنرا در حافظه اش دارد) می فرستد (شکل ۵-۷ را ببینید). احتمال کمی هست که این سرویس دهنده آدرس *linda.cs.yale.edu* (یا حتی *cs.yale.edu*) را بداند، ولی حتماً بچه های خودش را می شناسد، پس درخواست را به سرویس دهنده نام *yale.edu* منتقل می کند (مرحله ۳). سرویس دهنده *yale.edu* هم درخواست را به *cs.yale.edu* هدایت می کند (مرحله ۴)، که باید اطلاعات معتبر را در اختیار داشته باشد. از آنجائیکه این مسیر از مسیرهای مختلفی عبور کرده، رکوردهای درخواستی هم باید از همان مسیر به *flits.cs.vu.nl* برگردند (مراحل ۵ تا ۸).

وقتی این رکوردها به *cs.vu.nl* رسید، در حافظه نهان آنجا ذخیره می شود تا در دفعات بعد مورد استفاده قرار گیرد. ولی این اطلاعات معتبر نیستند، چون هر تغییری که در *cs.yale.edu* داده شود بطور خودکار در حافظه نهان DNS هایی که این رکوردها در آنجا وجود دارد، پخش نخواهد شد. به همین دلیل، آیتمهای حافظه نهان نباید عمری طولانی داشته باشند؛ و علت قرار دادن فیلد *Time_to_live* در رکوردهای منبع نیز همین است. این فیلد به سرویس دهنده نام می گوید که تا چه مدتی می تواند رکورد را در حافظه نهان خود نگه دارد. اگر یک ماشین IP خود را سالها حفظ می کند، نگه داشتن آن برای ۱ روز در حافظه نهان DNS چندان غیرمنطقی نیست. اما اطلاعات ناپایدارتر را بهتر است بیش از چند ثانیه (یا حداکثر ۱ دقیقه) در حافظه نهان نگه نداریم.

به پرس و جو هایی که به طریق بالا عمل می کنند، جستجوی بازگشتی (recursive query) می گویند، چون هر سرویس دهنده ای که اطلاعات خواسته شده را نداشته باشد، آنرا به سرویس دهنده بالاتر هدایت کرده و جواب را

باز می‌گرداند. روش جستجوی دیگری نیز وجود دارد: سرویس دهنده‌ای که اطلاعات درخواست شده را ندارد، خود به سرویس دهنده بالاتر مراجعه نمی‌کند، بلکه آدرس آنرا به درخواست کننده برمی‌گرداند. برخی از سرویس دهنده‌های DNS قادر به انجام جستجوی بازگشتی نیستند، و همیشه آدرس سرویس دهنده بعدی را برمی‌گردانند.

اگر یک مشتری DNS در زمان مقرر پاسخ خود را دریافت نکند، سراغ سرویس دهنده DNS بعدی خواهد رفت. این اتفاق بیشتر در مواردی رخ می‌دهد که سرویس دهنده DNS بدایلی از مدار خارج شده باشد. با اینکه DNS کار ساده‌ای انجام می‌دهد (تبدیل نام به آدرس IP)، اما کارکرد صحیح آن در اینترنت اهمیت حیاتی دارد. DNS هیچ کمکی برای یافتن افراد، منابع، سرویسها، اشیاء و چیزهایی از این قبیل نمی‌تواند بکند؛ برای این کارها سرویس دیگری بنام LDAP (پروتکل سبک دسترسی دایرکتوری - Light-weight Directory Access Protocol) تعریف شده است. این سرویس شکل ساده شده سرویس دایرکتوری OSI X.500 است، که در استاندارد RFC 2251 تشریح شده است. در LDAP (که می‌توان آنرا «دفتر تلفن اینترنتی» بشمار آورد)، اطلاعات بصورت درختی منظم شده‌اند، و امکانات فراوانی برای جستجو در این درخت تعبیه شده است. در این کتاب بیش از این درباره LDAP صحبت نخواهیم کرد؛ برای کسب اطلاعات بیشتر می‌توانید به (Weltman and Dahbura, 2000) مراجعه کنید.

۲-۷ پُست الکترونیک

بیش از دو دهه است که پُست الکترونیک، یا آنطور که هوادارانش می‌گویند ایمیل (e-mail)، در صحنه حضور دارد. تا سال ۱۹۹۰، این سرویس بیشتر در دانشگاهها و مراکز علمی وجود داشت، ولی وقتی در این سال بصورت سرویسی عمومی در آمد، با چنان سرعتی رشد کرد که در طی یک دهه تعداد نامه‌های الکترونیکی فرستاده شده از تعداد نامه‌های کاغذی فراتر رفت.

ایمیل، مانند سایر روشهای ارتباطی، دارای قواعد و شیوه‌های خاص خود است. جاذبه ایمیل بسیار بالاست، بطوریکه حتی آنهاییکه بندرت نامه‌های معمولی می‌نویسند، در نوشتن نامه‌های الکترونیکی (حتی به مقامات رسمی و سطح بالا) تردیدی بخود راه نمی‌دهند.

نامه‌های الکترونیکی پُر از کلماتیست که قبلاً در هیچ کجا دیده نشده‌اند: BTW (By The Way - راستی)، ROTFL (Rolling On The Floor Laughing - از خنده غش کردم)، و IMHO (In My Humble Opinion - به نظر من ناقابل) از آن نمونه‌اند. بسیاری افراد نیز در ایمیلهای خود از علائم خاصی موسوم به خندانک (smiley) یا احساس‌نما (emoticon) استفاده می‌کنند. در شکل ۶-۷ تعدادی از معروفترین این خندانک‌ها (و معنای آنها) را می‌بینید. اگر می‌خواهید بهتر متوجه معنای این علائم شوید، کتاب را ۹۰° در جهت

مفهوم	خندانک	مفهوم	خندانک	مفهوم	خندانک
دماغ گنده	: +)	عملینکن	= : -)	من خوشحالم	: -)
غیب بزرگ	: -)	عمو سام	=) : -)	من غمگین / ناراحتم	: - (
سبیلو	: - {	پاپا نول	* < : -)	من بی تفاوتم	: -
ژولیده مو	# : -)	کودن / احمق	< : - (من چشمک می‌زنم	: -)
عینکی	8 -)	استرالیایی	(: -)	من خمیازه می‌کشم	: - (O)
با هوش	C : -)	مرد فلکی	: -) X	حالم به هم خورد	: - (*)

شکل ۶-۷. چند خندانک. اینها جزء امتحان نهایی نیستند (-):

عقره های ساعت بچرخانید. برای دیدن تعداد زیادی از این قبیل خندانک ها به (Sanderson and Dougherty, 1993) مراجعه کنید.

اولین سیستم ایمیل فقط یک پروتکل ساده انتقال فایل (file transfer) بود، که آدرس گیرنده در خط اول پیام (فایل) نوشته می شد. با گذشت زمان محدودیت های این روش آشکارتر شد، که برخی از آنها عبارت بودند از:

۱. فرستادن یک پیام به چند نفر مشکل بود. این اشکال بیشتر مدیران را آزار می داد، چون آنها میل داشتند پیامهای خود را به تمام افراد زیر دست خود بفرستند.

۲. پیامها هیچگونه ساختار داخلی نداشتند، و بهمین دلیل پردازش کامپیوتری آنها مشکل بود. برای مثال، اگر پیامی از طرف یک شخص واسطه هدایت یا فرستاده می شد، استخراج قسمت هدایت شده مشکل بود.

۳. فرستنده نامه هرگز نمی توانست بداند پیامش به گیرنده رسیده یا نه.

۴. اگر کسی قصد داشت برای مدتی به مرخصی برود و می خواست در این مدت نامه های وارده به دست منشی اش برسد، کار ساده ای نبود.

۵. واسطه کاربر (جایی که نامه را می نوشت) با قسمت ارسال نامه یکپارچه نبود. کاربر باید ابتدا نامه را می نوشت، و برای ارسال آن برنامه ادیتور را ترک می کرد، و به قسمت انتقال فایل می رفت.

۶. نامه ها فقط متن بود؛ ارسال تصویر، طرح، صدا و مانند آنها ممکن نبود.

بتدریج سیستمهای ایمیل بهتری عرضه شد. در سال ۱۹۸۲، آرپانت سیستم ایمیل پیشنهادی خود را در RFC 821 (پروتکل انتقال) و RFC 822 (فرمت پیام) ارائه کرد. این پیشنهادها با تغییراتی اندک با عنوان RFC 2821 و RFC 2822 به استاندارد اینترنت تبدیل شد - اما هنوز هم استاندارد ایمیل اینترنت را با نام RFC 822 می شناسند. در ۱۹۸۴، CCITT توصیه ای بنام X.400 ارائه کرد. بعد از دو دهه، اغلب سیستمهای ایمیل همچنان به RFC 822 پایبند هستند، و X.400 عملاً کنار گذاشته شده است. این که چگونه سیستمی به عظمت X.400 که تمام مقامات رسمی استاندارد، شرکتهای مخابرات سراسر دنیا، دولتها و بسیاری از صنایع کامپیوتری پشتیبان آن بودند، مغلوب سیستمی که چند دانشجوی کامپیوتر آنرا نوشته اند، می شود بیشتر به داستان داوود و گولیات شبیه است. علت موفقیت RFC 822 خوبی آن نبود، بلکه این X.400 بود که چنان پیچیده و بد طراحی شده بود که پیاده سازی آن را عملاً غیر ممکن می کرد. انتخاب بین یک سیستم ساده ولی کاری (مانند RFC 822) و سیستمی فوق العاده جالب که در عمل کار نمی کرد (مانند X.400) چندان دشوار نبود. این عبرت تاریخ است.

۱-۲-۷ معماری و سرویسها

در این قسمت خواهید دید که یک سیستم ایمیل چه کاری می تواند انجام دهد، و سازماندهی آن چگونه است. هر سیستم ایمیل دارای دو زیرسیستم است: عامل کاربر (user agent)، که به افراد اجازه می دهد پیامهای خود را بفرستند و پیامهای رسیده را بخوانند، و عامل انتقال پیام (message transfer agent)، که پیامها را به دست گیرنده می رساند. عامل کاربر برنامه ایست (با ظاهر معمولی) روی کامپیوتر محلی کاربر، که با سیستم ایمیل بر هم کنش دارد. در حالیکه عامل انتقال پیام معمولاً یک سرویس (service یا daemon) است که در پس زمینه اجرا می شود، و وظیفه آن انتقال پیام در سیستم ایمیل است.

یک سیستم ایمیل، معمولاً، پنج کارکرد اصلی دارد، که آنها را در زیر توضیح می دهیم.

تصنیف: نوشتن پیام و جواب آن. با اینکه از هر ادیتوری می توان برای نوشتن پیامها استفاده کرد، ولی اغلب سیستمهای ایمیل دارای ادیتور خاص خود هستند که بسیاری از کارها (از جمله نوشتن آدرس، و سرآیند ایمیل) را بطور خودکار انجام می دهند. برای مثال، وقتی می خواهید به یک نامه جواب بدهید، سیستم ایمیل می تواند آدرس فرستنده نامه را بطور خودکار استخراج کرده و در فیلد گیرنده جوابیه (reply) قرار دهد.

انتقال: فرستادن پیام از فرستنده به گیرنده. این فرآیند سه مرحله دارد: تماس با ماشین گیرنده (یا یک ماشین واسطه)، فرستادن پیام، و قطع ارتباط. سیستم ایمیل این کارها را بطور خودکار و بدون دخالت کاربر انجام می دهد. گزارش دهی: مطلع کردن کاربر از سرنوشت پیام فرستاده شده. نامه تحویل شد؟ گیرنده آنرا قبول نکرد؟ در راه گم شد؟ در برخی مواقع اطمینان از رسیدن نامه بدست گیرنده اهمیت حیاتی و تبعات قانونی دارد (مانند احضاریه های دادگاه).

نمایش: پیامهای رسیده باید بگونه ای مناسب در معرض دید کاربر قرار گیرند، تا وی بتواند بر راحتی آنها را بخواند. گاهی لازم است برای خواندن محتویات برخی نامه ها (مانند نامه هایی که پیوست صوتی یا تصویری دارند) از برنامه های کمکی استفاده شود. برخی از سیستمهای ایمیل نیز نامه ها را بگونه ای خاص فرمت کرده و نمایش می دهند.

بایگانی: نگهداری نامه های رسیده. سرنوشت پیامهای رسیده متفاوت است: برخی پیامها حتی قبل از خوانده شدن دور انداخته می شوند، برخی فقط یک بار ارزش خواندن دارند، و برخی دیگر را باید حتماً ذخیره کرد. یک سیستم ایمیل باید بتواند نامه ها را به انحاء مختلف پردازش کند.

علاوه بر این سرویسهای اصلی، برخی از سیستمهای ایمیل (مخصوصاً سیستمهای رسمی) دارای ویژگیهای پیشرفته دیگری نیز هستند، که در زیر به برخی از آنها اشاره می کنیم.

وقتی یک فرد از محلی نقل مکان می کند (یا برای مدتی به مأموریت می رود)، باید بتوان پیامهای وی را به محل جدید هدایت کرد (forward). سیستم ایمیل باید بتواند این کار را بصورت خودکار انجام دهد.

در اکثر سیستمها کاربران اجازه دارند برای ذخیره کردن پیامهای خود صندوق پستی (mailbox) داشته باشند. سیستم ایمیل باید فرمانهایی برای ایجاد، مدیریت و یا از بین بردن این صندوقها داشته باشد.

اغلب مدیران نیاز دارند تا یک پیام را به افراد متعددی (کارمندان، مشتریان، یا شرکتهای طرف قرارداد) بفرستند. از اینجا بود که ایده لیست پستی (mailing list) - که در واقع لیستی از آدرسهای ایمیل است - پیدا شد. وقتی پیامی به یک لیست پستی فرستاده می شود، تمام افراد لیست کپی های کاملاً یکسانی از آن پیام دریافت خواهند کرد.

ویژگیهای دیگر عبارتند از: کپی (CC)، کپی ناشناس (BCC)، ایمیل با اولویت زیاد، ایمیل سری (رمز شده)، گیرنده جانشین (وقتی گیرنده اصلی در دسترس نبود)، و تحویل نامه های رئیس به منشی.

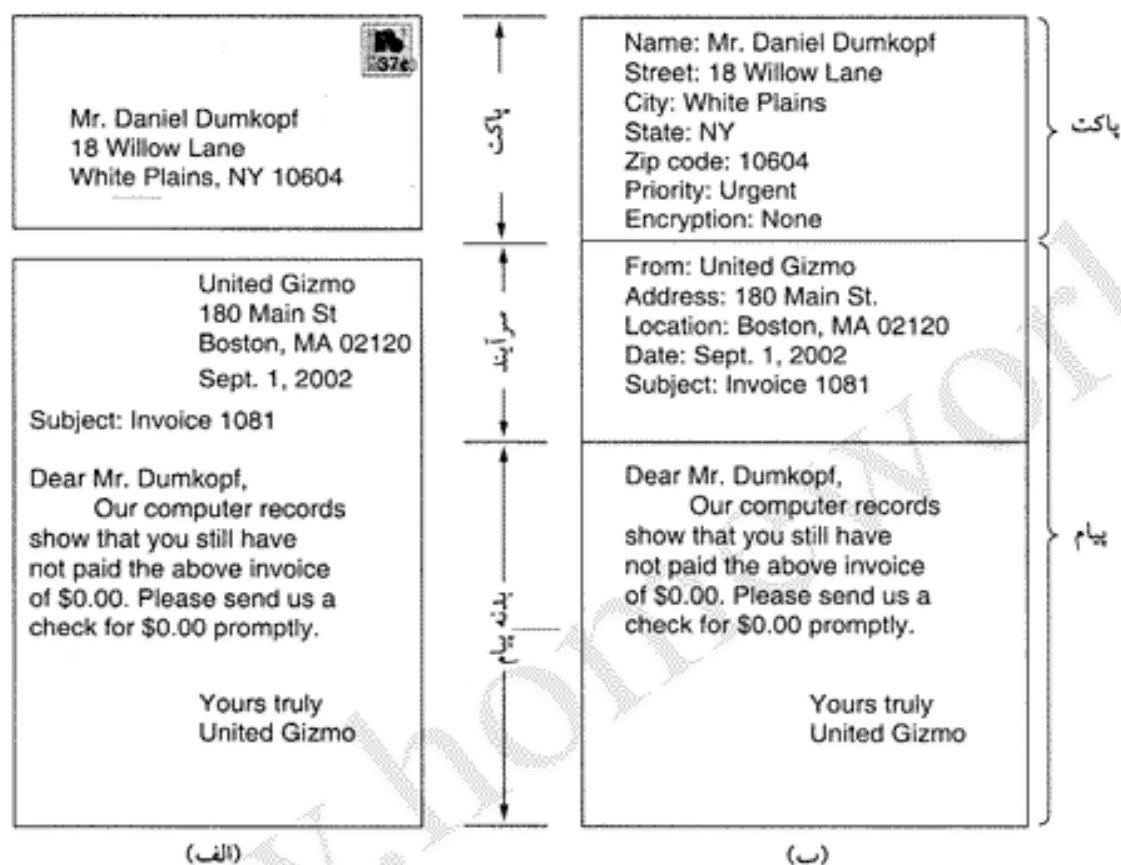
امروزه ایمیل کاربرد گسترده ای برای ارتباطات داخلی شرکتها و سازمانها دارد. ایمیل می تواند گروه گسترده ای از افراد (حتی آنهایی که بسیار از یکدیگر دور هستند) را در یک پروژه گرد آورد. ایمیل با حذف اکثر تمایزها (مانند مقام، سن، و جنسیت) باعث تمرکز روی اهداف می شود. با ایمیل، ایده درخشان یک کارآموز ساده اهمیت فزونی نسبت به ایده های (اکثراً احمقانه) مدیر عامل خواهد یافت.

ایده کلیدی در سیستم ایمیل تمایز بین پاکت و محتویات نامه است. پاکت نامه پیام را در خود جای می دهد، و شامل اطلاعاتی از قبیل آدرس گیرنده، اولویت و سطح امنیتی آن می شود، که معمولاً ارتباطی با محتویات نامه ندارند. عامل انتقال پیام از اطلاعات این پاکت برای جابجایی صحیح پیام استفاده می کند (درست مثل اداره پست). محتویات پاکت دو بخش دارد: سرآیند (header) و بدنه (body). سرآیند شامل اطلاعات کنترلیست که عامل کاربر از آنها برای کار خود استفاده می کند. بدنه بخشی از پیام است که به گیرنده مربوط می شود. در شکل ۷-۷ رابطه پاکت و پیام نشان داده شده است.

۷-۲-۲ عامل کاربر

همانطور که گفتیم، سیستمهای ایمیل دو بخش اصلی دارند: عامل کاربر، و عامل انتقال پیام. در این قسمت بخش

اول (یعنی، عامل کاربر) را بررسی خواهیم کرد. عامل کاربر یک برنامه معمولیست (و گاهی به آن نامه خوان - mail reader - نیز می گویند)، که می تواند در نوشتن پیامها، خواندن نامه های رسیده، پاسخ به آنها و کارهایی از این قبیل به کاربر کمک کند. طیف وسیع و متنوعی از برنامه های عامل کاربر وجود دارد، اما کارکرد اصلی آنها بسیار شبیه یکدیگر است.



شکل ۷-۷. پاکت و پیام در (الف) پست کاغذی، (ب) پست الکترونیک.

فرستادن ایمیل

برای فرستادن یک ایمیل، کاربر باید ابتدا متن نامه را نوشته و آدرس گیرنده را هم مشخص کند. برای نوشتن نامه می توان از ادیتورها یا واژه پردازهای مستقل، و یا ادیتوری که به همراه عامل کاربر می آید، استفاده کرد. آدرس گیرنده باید با فرمتی باشد که برای عامل کاربر آشناست؛ بسیاری از آنها آدرسها را با فرمت `user@dns-address` می پذیرند. با فرمت نامهای DNS در ابتدای همین فصل آشنا شدید.

غیر از آدرسهای DNS فرمتهای دیگری هم برای آدرسهای ایمیل وجود دارد، که بویژه نوع X.400 آن قابل توجه است. آدرسهای X.400 تفاوت قابل توجهی با آدرسهای DNS دارند. هر آدرس X.400 مجموعه ایست از زوجهای `attribute = value` که با / از هم جدا می شوند:

/C=US/ST=MASSACHUSETTS/L=CAMBRIDGE/PA=360 MEMORIAL DR./CN=KEN SMITH/

در این آدرس بترتیب از چپ براست کشور (C)، ایالت (ST)، محل (L)، آدرس (PA)، و نام شخص (CN) نوشته می شود. مشخصات دیگری (مانند شرکت و شغل) نیز وجود دارد، که با استفاده از آنها می توانید پیام خود را حتی به فردی که آدرس ایمیل او را نمی دانید، بفرستید. با اینکه آدرسهای X.400 بسیار غریب تر از نامهای

DNS هستند، اما اغلب سیستمهای ایمیل اجازه می دهند تا هر آدرس یک نام مستعار ساده (موسوم به nickname) داشته باشد. بدین ترتیب دیگر لازم نیست کاربر آدرس ایمیل (حتی X.400) را بطور کامل وارد کند. اغلب سیستمهای ایمیل از لیست پستی پشتیبانی می کنند، بنابراین می توان یک نامه را یکباره برای تعداد زیادی از افراد فرستاد. اگر لیست پستی بصورت محلی نگهداری شود، ایمیل در همان مبدأ بین افراد آن لیست توزیع خواهد شد. ولی اگر لیست پستی در محل دیگری نگهداری شود، پیام در آنجا پخش می شود. برای مثال، اگر گروهی از طرفداران حیات وحش یک لیست پستی بنام *birders* در دانشگاه آریزونا داشته باشند (*birders@meadowlark.arizona.edu*)، نامه ای که به این آدرس فرستاده شده باشد، فقط بعد از رسیدن به دانشگاه آریزونا بین افراد آن توزیع خواهد شد. آنهایی که در یک لیست پستی قرار دارند، بهیچوجه متوجه این مطلب نخواهند شد؛ آنها هم مانند سایر افراد پیامهایی از یک فرستنده مشخص دریافت می کنند.

خواندن ایمیل

معمولاً، وقتی عامل کاربر کار خود را شروع می کند، قبل از هر چیز صندوق پستی کاربر را چک کرده و نامه های رسیده را از آنجا برمی دارد. پس از آن، تعداد نامه های رسیده را به کاربر اعلام کرده، و احتمالاً آنها را بصورت خلاصه (شامل اطلاعاتی از قبیل فرستنده، تاریخ دریافت، و موضوع نامه) به کاربر نمایش می دهد، و سپس منتظر اقدام بعدی کاربر می ماند.

بعنوان مثال، یک سناریوی ساده را در زیر بررسی می کنیم. بعد از شروع برنامه عامل کاربر، اولین اقدام معمولاً نمایش خلاصه ای از نامه های رسیده است. این خلاصه می تواند چیزی شبیه شکل ۷-۸ باشد: هر خط مشخصات یک پیام را نشان می دهد. در این مثال، هشت پیام در صندوق پستی وجود دارد.

موضوع	فرستنده	بایت	پرچم	#
Changes to MINIX	asw	1030	K	1
Not all Trudys are nasty	trudy	6348	KA	2
Request for information	Amy N. Wong	4519	K F	3
Bioinformatics	bal	1236		4
Material on peer-to-peer	kaashoek	104110		5
Re: Will you review a grant proposal	Frank	1223		6
Our paper has been accepted	guido	3110		7
Re: My student's visit	dmr	1204		8

شکل ۷-۸ نمایش محتویات صندوق پستی.

در هر خط اطلاعات مختلفی دیده می شود. در یک سیستم ایمیل ساده این اطلاعات ثابت است، ولی در سیستمهای پیچیده تر کاربر می تواند نحوه نمایش اطلاعات را بنا به میل خود تغییر دهد (و این تنظیمات را در فایل بنام پروفایل کاربر - user profile - ذخیره کند). در مثال بالا، اولین فیلد شماره پیام است. فیلد دوم، *Flags*، مقادیر مختلفی می تواند بگیرد: *K* یعنی این پیام آرشیوی است (جدید نیست و قبلاً خوانده شده)؛ *A* یعنی به این پیام پاسخ داده شده است؛ *F* یعنی این پیام به فرد دیگری هدایت شده است (forward). پرچمهای دیگری نیز در این فیلد می تواند وجود داشته باشد، که هر کدام معنای خاص خود را دارند.

فیلد سوم طول پیام، و فیلد چهارم فرستنده آنرا نشان می دهند. از آنجائیکه این فیلد از نامه رسیده استخراج

می‌شود، محتویات آن به نامه فرستاده شده بستگی دارد. و بالاخره، در فیلد موضوع خلاصه‌ای از محتویات نامه را می‌بینید. سعی کنید همیشه نامه‌هایتان «موضوع» داشته باشد، چون تجربه نشان داده که نامه‌های بدون موضوع پراحتی نادیده گرفته می‌شوند.

بعد از خواندن خلاصه نامه، کاربر می‌توان اقدامات مختلفی روی آن انجام دهد: نامه را باز کند، آنرا حذف کند، به نامه جواب دهد، آنرا برای کس دیگری بفرستد، و مانند آن. برای هر یک از این اعمال، فرمان خاصی در برنامه عامل کاربر وجود دارد.

سیستمهای ایمیل امروزی خیلی بیش از انتقال فایل ساده‌اند. مدیریت حجم زیادی از نامه‌ها با برنامه‌های امروزی کار چندان دشواری نیست؛ و برای افرادی که در سال هزاران ایمیل رد و بدل می‌کنند، این مزیت کوچکی نیست.

۳-۲-۷ فرمت پیامها

اجازه دهید کمی هم درباره فرمت پیامهای ایمیل صحبت کنیم. ابتدا به ایمیلهای متنی ساده با فرمت RFC 822 می‌پردازیم، و پس از آن درباره الحاقات چندرسانه‌ای در RFC 822 توضیح خواهیم داد.

RFC 822

هر پیام یک پاکت ساده (که مشخصات آن در RFC 821 آمده) دارد، علاوه تعدادی سرآیند، یک خط خالی، و بعد از آن متن یا بدنه پیام. هر فیلد سرآیند عبارتست از یک خط متن ASCII مشتمل بر: نام فیلد، علامت :، و مقدار فیلد (که البته برخی از فیلدها می‌توانند مقدار نداشته باشند). در استاندارد RFC 822 که بیش از دو دهه از عمر آن می‌گذرد، تمایز آشکاری بین فیلدهای پاکت نامه و فیلدهای سرآیند وجود ندارد. با اینکه در RFC 2822 این مشکل مرتفع شده، ولی بعلت رواج گسترده آن در عمل چنین اتفاقی نیفتاده است. در حالت عادی، عامل انتقال پیام با استخراج اطلاعات از نامه‌ای که از عامل کاربر دریافت می‌کند، پاکت نامه را می‌سازد، و بهمین دلیل پاکت و نامه با هم مخلوط می‌شوند.

مهمترین فیلدهای سرآیند که نقش مهمی در انتقال پیام دارند، در شکل ۷-۹ نشان داده شده است. در فیلد TO: آدرس DNS گیرنده اصلی پیام نوشته می‌شود. نوشتن چندین گیرنده در این فیلد مجاز است. در فیلد Cc: آدرس گیرنده‌های ثانویه پیام نوشته می‌شود. از نظر تحویل پیام در سیستم ایمیل، تفاوتی بین گیرنده‌های اصلی و ثانویه وجود ندارد؛ این فیلد بیشتر برای انسانها مهم است تا برای ماشین. اصطلاح Cc (کپی کربنی) قدری قدیمی است، چون در کامپیوترها چیزی بنام کاغذ کپی وجود ندارد، اما این اصطلاح دیگر کاملاً جا افتاده است. فیلد Bcc: (کپی کربنی ناپیدا) شبیه Cc: است، با این تفاوت که این فیلد در نامه‌های کپی شده حذف می‌شود، و گیرنده نامه از هویت سایر گیرنده‌ها (و حتی وجود چنین گیرنده‌هایی) مطلع نخواهد شد.

سرآیند	مفهوم
To:	آدرس ایمیل گیرنده های اصلی
Cc:	آدرس ایمیل گیرنده های ثانویه (کپی)
Bcc:	آدرس ایمیل گیرنده های کپی های ناشناس
From:	فرستنده پیام
Sender:	آدرس ایمیل فرستنده
Received:	هر عامل انتقال واسطه در بین راه مشخصات خود را اضافه می کند
Return-Path:	می توان از آن برای مشخص کردن مسیر برگشت به فرستنده استفاده کرد

شکل ۷-۹. فیلدهای سرآیند RFC 822 برای انتقال پیام.

دو فیلد بعدی، یعنی *From:* و *Sender:*، برترتیب نویسنده و فرستنده نامه را مشخص می‌کنند. این دو الزاماً یکی نیستند (ولی اغلب چنین است). برای مثال، نویسنده نامه می‌تواند مدیر عامل باشد (*From:*)، ولی منشی شرکت آنرا بفرستد (*Sender:*)، فیلد *From:* حتماً باید پُر شود، ولی فیلد *Sender:* (اگر با *From:* یکی باشد) می‌تواند خالی رها شود. اگر احتمال می‌دهید نامه بدست گیرنده نمی‌رسد و برگشت می‌خورد، حتماً فیلد *Sender:* را پُر کنید، چون نامه‌های غیرقابل تحویل به این آدرس برگشت داده می‌شوند.

وقتی یک نامه از واسطه‌های مختلفی عبور می‌کند تا به دست گیرنده برسد، نام هر واسطه در یک فیلد *Received:* جداگانه نوشته می‌شود. در این فیلد نام عامل گیرنده، تاریخ و زمان دریافت پیام، و اطلاعات دیگر ثبت می‌شود. از این اطلاعات می‌توان برای رفع اشکالاتی که در طی تحویل نامه‌ها پیش می‌آید، استفاده کرد. فیلد *Return-Path:* که توسط آخرین عامل انتقال پیام اضافه می‌شود، نشان می‌دهد که مسیر برگشت به فرستنده چگونه است. از نظر تنوری این اطلاعات باید شامل تمام سرآیندهای *Received:* (بجز صندوق پستی فرستنده) باشد، ولی بندرت چنین است و معمولاً فقط آدرس فرستنده در آن نوشته می‌شود.

علاوه بر فیلدهای شکل ۷-۹، پیامهای RFC 822 دارای سرآیندهای دیگری نیز هستند که به بیشتر کار عامل کاربر یا شخص گیرنده می‌آیند. در شکل ۷-۱۰ برخی از این سرآیندها را می‌بینید. کارکرد اغلب این فیلدها از روی نامشان پیداست، و نیازی به توضیح زیاد ندارند.

مفهوم	سرآیند
زمان و تاریخ ارسال	Date:
آدرس ایمیل برای پاسخ نامه	Reply-To:
عدد منحصر بفرد شناسایی پیام	Message-Id:
شماره پیامی که این پاسخ پیام پاسخ آن است	In-Reply-To:
سایر شماره های مربوطه	References:
کلمات کلیدی انتخاب شده توسط کاربر	Keywords:
موضوع پیام	Subject:

شکل ۷-۱۰. برخی از فیلدهای سرآیند RFC 822.

فیلد *Reply-To:* برای مواقعیست که نویسنده و گیرنده نامه هیچکدام نمی‌خواهند گیرنده پاسخ نامه باشند. برای مثال، وقتی مدیر بازاریابی نامه‌ای درباره محصولات جدید شرکت به یک مشتری می‌نویسد، و منشی هم آنرا می‌فرستد، فیلد *Reply-To:* می‌تواند به آدرس قسمت فروش شرکت، که پاسخگوی سفارشات هستند، اشاره کند. این فیلد برای مواردی که فرستنده دو آدرس ایمیل دارد، و مایل است پاسخها را از طریق آدرس دیگرش بگیرد، نیز مفید است.

استاندارد RFC 822 اجازه می‌دهد تا کاربران سرآیندهای دلخواهشان را به نامه‌ها اضافه کنند، مشروط باینکه این سرآیندها با X- شروع شوند (هیچیک از سرآیندهای رسمی این استاندارد با X- شروع نمی‌شوند، و در آینده نیز نخواهند شد). این قبیل سرآیندها می‌توانند اطلاعات اضافی را با خود حمل کنند.

بعد از سرآیند، بدنه نامه می‌آید. کاربران می‌توانند هر چیزی در این بدنه بنویسند. برخی افراد در انتهای نامه‌هایشان اختتامیه‌های ماهرانه‌ای می‌آورند، مانند اشکال کارتونی جالب و خنده‌دار، کلمات قصار مشاهیر، و یا عبارات قانونی سلب مسئولیت (مثلاً، «شرکت فلان و بهمان هیچگونه مسئولیتی را درباره محتویات این نامه نمی‌پذیرد.»).

MIME - الحاقات چند منظوره پست اینترنت

در روزهای اولیه آرپانت، ایمیل ها فقط متن ساده بود که به زبان انگلیسی و با فرمت ASCII نوشته می شد. استاندارد RFC 822 با این وضعیت هیچ مشکلی نداشت: کاربر می توانست هر چیزی که می خواست در بدنه نامه بنویسد. اما این روش دیگر برای دنیای امروز کافی نیست. برخی از مشکلات ذاتی این سیستم عبارتند از:

۱. ارسال پیام به زبانهایی که اعراب دارند (مانند فرانسه و آلمانی).
۲. ارسال پیام به زبانهای غیر لاتین (مانند عربی و روسی).
۳. ارسال پیام به زبانهای غیرالفبایی (مانند چینی و ژاپنی).
۴. ارسال پیامهایی که اصلاً متن نیستند (مانند صدا و تصویر).

راه حل این مشکلات در RFC 1341 ارئه شد، و بعدها در RFC 2045-49 به روز در آمد. این راه حل که MIME (الحاقات چند منظوره پست اینترنت - Multipurpose Internet Mail Extensions) نام دارد، امروزه بطور گسترده ای رواج یافته است.

ایده اصلی MIME عبارتست از: ادامه استفاده از فرمت RFC 822، و اضافه کردن ساختاری جدید به بدنه پیام و تعریف قواعد درج پیامهای غیرمتنی. پیامهای MIME با برنامه ها و پروتکل های موجود ایمیل کاملاً سازگارند، چون از استاندارد RFC 822 تخطی نمی کنند. فقط برنامه های عامل کاربر باید عوض شوند، که این کار را هم کاربران می توانند براحتی انجام دهند.

MIME پنج سرآیند جدید تعریف می کند، که آنها را در شکل ۷-۱۱ مشاهده می کنید. اولین سرآیند به عامل کاربر دریافت کننده پیام می گوید که با یک پیام MIME سروکار دارد، و ویرایش آن هم اعلام می شود. هر پیامی که سرآیند *MIME-Version:* نداشته باشد، متن ساده تلقی شده و به همان طریق پردازش می شود.

مفهوم	سرآیند
ویرایش MIME پیام	<i>MIME-Version:</i>
جمله ای دوباره محتویات پیام	<i>Content-Description:</i>
عدد منحصر بفرد شناسایی محتویات پیام	<i>Content-Id:</i>
نحوه کدگذاری پیام	<i>Content-Transfer-Encoding:</i>
نوع و فرمت محتویات پیام	<i>Content-Type:</i>

شکل ۷-۱۱. سرآیندهای اضافی MIME در RFC 822.

سرآیند *Content-Description:* یک عبارت متنی است که می گوید چه چیزی در پیام وجود دارد. از روی این سرآیند است که گیرنده تشخیص می دهد آیا محتویات پیام ارزش خواندن دارد یا خیر. برای مثال، اگر سرآیند *Content-Description:* بگوید: «این عکس یک موش است»، و گیرنده علاقه ای به دیدن عکس موش نداشته باشد، برنامه پیام را دور انداخته و تلاشی برای نمایش این عکس نخواهد کرد.

سرآیند *Content-Id:* محتویات پیام را مشخص می کند. فرمت این سرآیند مانند *Message-Id:* است. سرآیند *Content-Transfer-Encoding:* نحوه کد شدن محتویات پیام (برای گذر از شبکه هایی که فقط به متن ساده اجازه عبور می دهند) را مشخص می کند. پنج نوع کدگذاری پیام وجود دارد. نوع اول فقط متن ساده ASCII است. کاراکترهای ASCII هفت بیتی هستند، و تمام پروتکل های ایمیل می توانند خطوط متن را (مشروط بر اینکه هر خط از ۱۰۰۰ حرف تجاوز نکند) منتقل کنند.

نوع دوم در واقع همان نوع قبلیست، که فقط از کاراکترهای ASCII ۸ بیتی (از ۰ تا ۲۵۵) استفاده می کند.

برخی از بخشهای اینترنت از این کُد برای ارسال متن (و نمایش کاراکترهای خاص) استفاده می کنند. این کُدگذاری در پروتکل های ایمیل اینترنتی مجاز نیست (و این تعریف هم باعث مجاز شدن آن نمی شود)، ولی حداقل توضیح می دهد که اشکال کار از کجاست. پیامهای دارای کُدگذاری ۸ بیتی هم محدود به خط های ۱۰۰۰ حرفی هستند. بدتر از آن پیامهایی هستند که کُدگذاری باینری دارند. اینها فایل های باینری هستند، که نه تنها از تمام حالات ممکنه ۸ بیت استفاده می کنند، بلکه به محدودیت ۱۰۰۰ بایت نیز وقعی نمی گذارند. برنامه های اجرایی در این دسته قرار می گیرند. هیچ تضمینی وجود ندارد که پیامهای باینری درست به مقصد برسند، ولی بسیاری افراد همچنان کار خودشان را می کنند.

یکی از روشهای کُدگذاری صحیح پیامهای باینری روش کُدگذاری base64 (که گاهی ASCII armor نیز گفته می شود) است. در این روش، هر دسته ۲۴ بیتی به چهار واحد ۶ بیتی تقسیم شده، و هر واحد بعنوان یک کاراکتر معتبر ASCII فرستاده می شود. در این روش "A" معادل ۰ است، "B" معادل ۱، ... و بالاخره "Z" معادل ۲۵؛ پس از آن ۲۶ حرف کوچک انگلیسی می آیند، و پس از آن ارقام ۰ تا ۹؛ ۶۲ و ۶۳ نیز بترتیب معادل + و / هستند. توالیهای = و = بترتیب نشان می دهند که آخرین گروه ۸ یا ۱۶ بیتی است. کاراکترهای «برگشت سر خط» (carriage return) و «خط بعدی» (line feed) نیز بکلی نادیده گرفته می شود، بنابراین می توان برای کوتاه کردن خطوط از آنها استفاده کرد. با این روش می توان فایل های باینری را بطور صحیح ارسال کرد.

برای پیامهایی که تقریباً بطور کامل ASCII هستند و فقط چند کاراکتر غیر ASCII دارند، کُدگذاری base64 کارایی مطلوبی ندارد. برای این قبیل پیامها از کُدگذاری quoted-printable استفاده می شود. این در واقع همان روش ASCII ۷ بیتی است، که در آن کاراکترهای بالای ۱۲۷ با علامت = و یک عدد هگزادسیمال دو رقمی مشخص می شوند.

اطلاعات باینری همواره باید با یکی از روشهای base64 یا quoted-printable فرستاده شوند. اگر دلیل موجهی برای استفاده نکردن از هر یک از روشها وجود داشته باشد، می توان از کُدگذاریهای خاص (که با سرآیند Content-Transfer-Encoding مشخص می شوند) استفاده کرد.

آخرین آیتم شکل ۷-۱۱ در واقع مهمترین سرآیند MIME است. این سرآیند حاصلت واقعی محتویات پیام را مشخص می کند. در RFC 2045 هفت نوع (type) تعریف شده، که هر کدام از آنها می توانند چندین زیرنوع (subtype) داشته باشند. نوع و زیرنوع با یک / از هم جدا می شوند:

Content-Type: video/mpeg

زیرنوع باید صریحاً در سرآیند مشخص شود: هیچ مقداری بعنوان پیش فرض وجود ندارد. لیست اولیه نوعها و زیرنوع های RFC 2045 را در شکل ۷-۱۲ ملاحظه می کنید. از آن زمان تاکنون آیتمهای دیگری اضافه شده است، و در آینده باز هم اضافه خواهد شد.

اجازه دهید این لیست را مختصراً بررسی کنیم. نوع text همان متن ASCII ساده است. زیرنوع text/plain به گیرنده می گوید پیام را بهمان صورتی که دریافت کرده (بدون هیچ فرمت یا پردازشی) نمایش دهد. این گزینه اجازه می دهد تا پیامهای معمولی بدون هیچ اقدام اضافه ای به پیامهای MIME تبدیل شوند.

زیرنوع text/enriched اجازه می دهد تا از زبانهای علامتگذاری ساده برای فرمت کردن متن (تعیین فونت، اندازه، رنگ و صفحه بندی) استفاده شود. این زیرنوع زیرمجموعه ای از SGML (زبان علامتگذاری عمومی) می استاندارد - که زبان استاندارد وب یعنی HTML نیز جزیی از آن محسوب می شود) است. برای مثال پیام

The <bold> time </bold> has come the <italic> walrus </italic> said ...

به این صورت به نمایش در خواهد آمد:

نوع	زیر نوع	توضیح
Text	Plain	متن فرمت نشده
	Enriched	متن یا فرمت ساده
Image	Gif	تصاویر با فرمت GIF
	Jpeg	تصاویر با فرمت JPEG
Audio	Basic	صدا
Video	Mpeg	تنظیم با فرمت MPEG
Application	Octet-stream	توالی بایت تفسیر نشده
	Postscript	سند چاپی با فرمت پست اسکریپت
Message	Rfc822	پیام RFC 822
	Partial	سند چند تکه شده
	External-body	پیام باید از اینترنت گرفته شود
Multipart	Mixed	پیامی با چند بخش مستقل
	Alternative	یک پیام با فرمت های مختلف
	Parallel	بخش های پیام باید همزمان دیده شوند
	Digest	هر بخش یک پیام RFC 822 کامل است

شکل ۷-۱۲. نوع ها و زیر نوع های MIME تعریف شده در RFC 2045.

The time has come the walrus said ...

فرمت کردن پیام بطور کامل بر عهده سیستم گیرنده است، و نباید انتظار داشته باشید چنین پیامی در تمام سیستمها یکسان (و آنطوری که شما تصور می کنید) دیده شود. گاهی یک سیستم معنای کاری را که شما خواسته اید نمی فهمد، و بجای آن کار دیگری انجام می دهد.

بعد از رواج وب، زیر نوع جدیدی بنام *text/html* (در RFC 2854) اضافه شد، که اجازه می داد صفحات وب از طریق ایمیل های RFC 822 فرستاده شوند. در RFC 3023 نیز زیر نوع دیگری (*text/xml*) برای ارسال پیام های XML تعریف شده است. در ادامه این فصل با HTML و XML بیشتر آشنا خواهید شد.

نوع MIME بعدی *image* است، که برای ارسال تصاویر ثابت بکار می رود. امروزه فرمت های بسیار متنوعی برای ذخیره و ارسال کردن تصاویر (با فشرده سازی یا بدون آن) وجود دارد، که از میان آنها فرمت های GIF و JPEG بسیار معروفند و تقریباً تمام مرورگرهای اینترنت از آنها پشتیبانی می کنند. (البته بعدها فرمت های دیگری نیز به این لیست اضافه شد.)

نوع های *audio* و *video* به ترتیب برای ارسال صدا و تصاویر متحرک هستند. توجه داشته باشید که نوع *video* فقط برای ارسال اطلاعات تصویری است، و اگر فایل شما دارای تراک صوتی هم باشد، باید آنها را جداگانه بفرستید. اولین فرمت ویدئویی که قابلیت ارسال از طریق ایمیل را پیدا کرد، فرمت MPEG (گروه تخصصی تصاویر متحرک - Moving Picture Expert Group) بود؛ بعدها فرمت های دیگر نیز اضافه شد. علاوه بر زیر نوع *audio/basic*، در RFC 3003 زیر نوع دیگری (*audio/mpeg*) برای ارسال فایل های صوتی MP3 تعریف شده است.

هر نوع فایل باینری دیگری که در دستجات بالا قرار نگیرد (و سیستم ایمیل نداند آنرا چگونه پردازش کند)، در ذیل نوع *application* دسته بندی خواهد شد. زیر نوع *octet-stream* فقط استریمی از بایتهاست (و سیستم هیچگونه تفسیری روی آنها نخواهد کرد). عامل کاربر بعد از دریافت این استریم، تنها کاری که می تواند انجام دهد ذخیره کردن آن بصورت یک فایل است - پس باید نام فایل را از کاربر بگیرد. پردازش های بعدی نیز بر عهده کاربر است.

زیرنوع تعریف شده دیگر *postscript* است، که به زبان پست اسکریپت (زبان توصیف صفحات چاپی، از شرکت Adobe) مربوط می شود. بسیاری از چاپگرهای امروزی دارای مفسرهای پست اسکریپت هستند. با اینکه عامل کاربر می تواند تفسیر فایل های پست اسکریپت را بر عهده برنامه های خارجی بگذارد، اما این کار خالی از خطر نیست. پست اسکریپت یک زبان برنامه نویسی کامل است، و یک برنامه نویس (خودآزار) می تواند با صرف وقت کافی حتی یک کامپایلر C یا سیستم مدیریت پایگاه داده با آن بنویسد. عامل کاربر برای نمایش محتویات فایل پست اسکریپت، در واقع برنامه ای را که در دل پیام فرستاده شده اجرا می کند. چنین برنامه ای حتی می تواند (در کنار نمایش یک متن ساده) فایل های کاربر را تغییر داده، یا آنها را پاک کند (و یا دهها کار ناجور دیگر انجام دهد).

نوع *message* اجازه می دهد تا یک پیام را بطور کامل در دل پیام دیگر جای دهیم. این نوع بویژه برای هدایت پیامها (*message forwarding*) مفید است. برای قرار دادن یک ایمیل RFC 822 در دل پیام دیگر، می توان از زیرنوع *message/rfc822* استفاده کرد.

با زیرنوع *partial* می توان یک پیام را چند تکه کرده، و هر تکه را در دل یک ایمیل جداگانه قرار داد (که این برای پیامهای خیلی بزرگ مناسب است). در مقصد سیستم ایمیل می تواند با استفاده از پارامترهای این زیرنوع، پیام تکه تکه شده را دوباره به هم بچسباند.

بالاخره، از زیرنوع *external-body* می توان برای پیامهای بسیار بزرگ (مانند فیلمهای ویدئویی) استفاده کرد؛ یعنی بجای قرار دادن فایل MPEG در دل پیام، آدرس FTP آن نامه در نوشته می شود، و عامل کاربر می تواند در موقع نیاز به این آدرس مراجعه کرده و فایل را بخواند. این زیرنوع بخصوص در مواردی که بخواهیم فایل بزرگی را برای یک لیست پستی بفرستیم، بسیار ایده آل است، چون احتمال اینکه همه اعضا لیست پستی این فایل را بخواهند چندان زیاد نیست (فرستادن آگهی های تبلیغاتی یکی از این موارد است).

نوع آخر *multipart* است، که اجازه می دهد یک پیام چندین بخش داشته باشد (بخشهایی که ابتدای و انتهای آنها کاملاً مشخص است). در زیرنوع *mixed* این بخشها می توانند کاملاً متفاوت باشند، بدون اینکه نیازی به ساختار اضافی وجود داشته باشد. در بسیاری از برنامه های ایمیل یک پیام ساده می تواند چندین پیوست (*attachment*) از انواع مختلف داشته باشد، که این پیوستها با استفاده از نوع *multipart* فرستاده می شوند.

بر خلاف *multipart*، در زیرنوع *alternative* یک پیام واحد به چندین فرمت (مانند متن ساده، متن فرمت دار، و یا پست اسکریپت) فرستاده می شود، که عامل کاربر گیرنده می تواند بسته به امکانات خود از یکی از این انواع استفاده کند. این انواع باید از ساده ترین به پیچیده ترین مرتب شوند، تا حتی کاربران غیر MIME بتوانند پیامها را بخوانند.

از زیرنوع *alternative* برای ارسال پیام به زبانهای مختلف هم می توان استفاده کرد. (سنگ روزتا را می توان یکی از قدیمی ترین پیامهای *multipart/alternative* دانست - سنگ روزتا سنگ نبشته ای است که توسط سپاهیان ناپلئون در منطقه ای به همین نام در مصر کشف شد، و در آن یک پیام واحد به زبانهای هیروگلیف و یونانی نوشته شده بود؛ با استفاده از همین سنگ نبشته بود که شامپولئون باستان شناس فرانسوی توانست معمای خط هیروگلیف را بعد از قرن ها کشف کند).

در شکل ۷-۱۳ یک پیام *multipart* را ملاحظه می کنید. در این مثال، یک پیام تبریک تولد به دو صورت متن و صوت فرستاده شده است. اگر کامپیوتر گیرنده کارت صوتی داشته باشد، برنامه عامل کاربر فایل صوتی *birthday.snd* را از شبکه گرفته، و پخش می کند. اما اگر چنین نباشد، فقط متن شعر را نمایش خواهد داد. دقت کنید که بخشهای پیام با دو - (خط تیره) و رشته ای از حروف (که نرم افزار آنها را تولید کرده، و در قسمت *boundary* مشخص شده) از هم جدا می شوند.

همچنین توجه کنید که سرآیند *Content-Type* در سه نقطه از این پیام آمده است. در بالای پیام، سرآیند

From: elinor@abcd.com
 To: carolyn@xyz.com
 MIME-Version: 1.0
 Message-Id: <0704760941.AA00747@abcd.com>
 Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
 Subject: Earth orbits sun integral number of times

This is the preamble. The user agent ignores it. Have a nice day:

--qwertyuiopasdfghjklzxcvbnm
 Content-Type: text/enriched

Happy birthday to you
 Happy birthday to you
 Happy birthday dear <bold> Carolyn </bold>
 Happy birthday to you

--qwertyuiopasdfghjklzxcvbnm
 Content-Type: message/external-body;
 access-type="anon-ftp";
 site="bicycle.abcd.com";
 directory="pub";
 name="birthday.snd"

content-type: audio/basic
 content-transfer-encoding: base64
 --qwertyuiopasdfghjklzxcvbnm--

شکل ۷-۱۳. یک پیام *multipart*، محتوی متن و صوت.

Content-Type می گوید که این یک پیام *multipart/alternative* است. در دو *Content-Type* بعدی نوع و زیرنوع هر بخش مشخص می شود. در آخرین *Content-Type* نیز نگذگذاری فایل صوتی را مشخص کرده ایم، چون هر چیزی که متن ASCII ۷ بیتی نباشد، باید دارای نگذگذاری مشخص باشد. پیامهای *multipart* دارای دو زیرنوع دیگر نیز می توانند باشند. از زیرنوع *parallel* وقتی استفاده می کنیم که بخواهیم تمام بخشهای پیام همزمان «مشاهده» شوند. برای مثال، فیلمهای ویدئویی دارای کانالهای تصویری و صوتی مجزا هستند، که باید همزمان پخش شوند (نه پدنبال هم).

زیرنوع *digest* وقتی بکار برده می شود که بخواهیم تعدادی پیام را در یک پیام مرکب بسته بندی کنیم. برای مثال، در گروههای مباحثه (discussion group) اینترنتی معمولاً چندین پیام که از اعضای مختلف جمع آوری شده، در یک پیام *multipart/digest* به سایر اعضای گروه فرستاده می شود.

۷-۲-۷ انتقال پیام

سیستم انتقال پیام (message transfer) با ارسال پیام از فرستنده به گیرنده سروکار دارد. ساده ترین راه برای این کار، برقراری یک اتصال مستقیم از ماشین فرستنده به ماشین گیرنده، و انتقال پیام است. بعد از بررسی این روش، به مواردی می پردازیم که چنین کاری امکان ندارد، و سپس برای آن موارد نیز راه حل هایی نشان خواهیم داد.

SMTP - پروتکل ساده انتقال نامه

در اینترنت، انتقال ایمیل با برقراری یک اتصال TCP از ماشین مبدأ به پورت 25 ماشین مقصد صورت می گیرد. برنامه ای که به این پورت گوش می کند، دیمون SMTP (پروتکل ساده انتقال نامه - Simple Mail Transfer Protocol) نام دارد. این دیمون اتصالات ورودی را پذیرفته، و پیامها را در صندوق پستی مربوطه کپی می کند. اگر گیرنده نتواند پیامی را تحویل گیرد، یک گزارش خطا حاوی اولین بخش از پیام مزبور به فرستنده برمی گرداند. SMTP یک پروتکل ساده ASCII است. بعد از برقراری اتصال TCP، ماشین فرستنده (که نقش مشتری را بازی می کند) منتظر می ماند تا ماشین گیرنده (که نقش سرویس دهنده را بازی خواهد کرد) شروع به صحبت کند. در شروع، سرویس دهنده یک خط متن فرستاده و ضمن معرفی خود، اعلام می کند که آیا آماده دریافت ایمیل هست یا خیر. اگر سرویس دهنده آماده نباشد، مشتری ارتباط را قطع کرده، و بعداً دوباره سعی خواهد کرد. اگر سرویس دهنده آماده دریافت باشد، مشتری اعلام می کند که پیام از چه کسی می آید و به چه کسی باید تحویل شود. اگر چنین دریافت کننده ای در ماشین سرویس دهنده وجود داشته باشد، سرویس دهنده از مشتری می خواهد که پیام را بفرستد. پس از آن که مشتری پیام را فرستاد، سرویس دهنده دریافت آن را تصدیق می کند. سرویس دهنده هیچ تلاشی برای چک کردن جمع تطبیقی انجام نخواهد داد، چون TCP سالم بودن ارتباط را تضمین می کند. اگر باز هم پیامی وجود داشته باشد، پس از آن فرستاده می شود. وقتی تمام ایمیل ها (در هر دو جهت) مبادله شد، ارتباط قطع می شود. در شکل ۷-۱۴ مکالمه سرویس دهنده و مشتری (منجمله گدهای عددی SMTP) برای ارسال ایمیل شکل ۷-۱۳ را ملاحظه می کنید. خطوطی که توسط مشتری ارسال شده اند، را با C، و آنهایی که توسط سرویس دهنده فرستاده شده اند، را با S مشخص کرده ایم.

کمی توضیح درباره شکل ۷-۱۴ می تواند مفید باشد. اولین پیام مشتری HELO است. این کلمه در واقع مخفف چهارحرفی «سلام» (HELLO) است، و پیداست که از همه چهارحرفی ها به آن شبیه تر است. اینکه چرا باید از کلمه های چهارحرفی استفاده کنیم، در غبار زمان گم شده، ولی این رسم همچنان پا برجاست. از آنجائیکه فقط یک گیرنده وجود دارد، در اینجا فقط یک دستور RCPT دیده می شود، ولی می توان در آن واحد یک پیام را به چندین گیرنده فرستاد (که قبول یا رد درخواست برای هر یک جداگانه انجام خواهد شد). با اینکه دستورات چهارحرفی از طرف مشتری ثابت و مشخص هستند، پاسخ سرویس دهنده فقط یک سری گدهای عددی است. در واقع همین گدهاست که اهمیت دارد، و هر سیستم می تواند برای هر کد توضیح خاص خود را داشته باشد.

اجازه دهید برای درک بهتر پروتکل SMTP، کمی درباره فرآیند کار توضیح دهیم. قبل از هر چیز سراغ کامپیوتری بروید که به اینترنت دسترسی دارد. سپس دستور زیر را در خط فرمان وارد کنید (در این مثال فرض کرده ایم سیستم عامل یونیکس است)

```
telnet mail.isp.com 25
```

(بجای mail.isp.com آدرس IP یا نام DNS سیستم ایمیل خود را وارد کنید). در سیستمهای ویندوز، پنجره Start|Run را باز کرده و دستور فوق را در آنجا وارد کنید. این دستور یک اتصال telnet (یعنی، TCP) با پورت 25 ماشین مشخص شده برقرار می کند. پورت 25 پورت SMTP است (شکل ۶-۲۷ را ببینید). پاسخی که دریافت خواهید کرد، احتمالاً چیزی شبیه زیر است:

```
Trying 192.30.200.66...
Connection to mail.isp.com
Escape character is '^]'
220 mail.isp.com Smail #74 ready at Thu, 25 Mar 2003 13:26 +0200
```

```

S: 220 xyz.com SMTP service ready
C: HELO abcd.com
S: 250 xyz.com says hello to abcd.com
C: MAIL FROM: <elinor@abcd.com>
S: 250 sender ok
C: RCPT TO: <carolyn@xyz.com>
S: 250 recipient ok
C: DATA
S: 354 Send mail; end with "." on a line by itself
C: From: elinor@abcd.com
C: To: carolyn@xyz.com
C: MIME-Version: 1.0
C: Message-Id: <0704760941.AA00747@abcd.com>
C: Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
C: Subject: Earth orbits sun integral number of times
C:
C: This is the preamble. The user agent ignores it. Have a nice day.
C:
C: --qwertyuiopasdfghjklzxcvbnm
C: Content-Type: text/enriched
C:
C: Happy birthday to you
C: Happy birthday to you
C: Happy birthday dear <bold> Carolyn </bold>
C: Happy birthday to you
C:
C: --qwertyuiopasdfghjklzxcvbnm
C: Content-Type: message/external-body;
C:     access-type="anon-ftp";
C:     site="bicycle.abcd.com";
C:     directory="pub";
C:     name="birthday.snd"
C:
C: content-type: audio/basic
C: content-transfer-encoding: base64
C: --qwertyuiopasdfghjklzxcvbnm
C: .
S: 250 message accepted
C: QUIT
S: 221 xyz.com closing connection

```

شکل ۷-۱۴. انتقال یک پیام از *elinor@abcd.com* به *carolyn@xyz.com*.

سه خط اول مربوط به برنامه telnet هستند، و می گویند چه اتفاقی در حال افتادن است. خط آخر از سرویس دهنده SMTP آمده، و آمادگی آنرا برای دریافت ایمیل از طرف شما اعلام می کند. برای اینکه ببینید سرویس دهنده ایمیل چه فرمانهایی را قبول می کند، دستور زیر را وارد کنید

HELP

از اینجا به بعد با چیزی شبیه شکل ۷-۱۴ روبرو خواهید بود. پروتکل های ASCII در اینترنت بسیار رایج هستند، چوت تست و دیباگ آنها بسیار ساده است. فرمان دادن به این پروتکلها بسیار آسان است، و پاسخ آنها را نیز براحتی می توان درک کرد.

با اینکه پروتکل SMTP بخوبی تعریف شده است، اما برخی مشکلات کوچک نیز می تواند بروز کند. یکی از این مشکلات طول پیام است: در برخی از سیستم های ایمیل قدیمی طول پیام نباید از 64 KB تجاوز کند. مشکل دیگر زمانهای متفاوت انتظار برای پاسخ در دو سمت مقابل است. اگر زمان انتظار برای دریافت پاسخ (timeout) در سمت سرویس دهنده و مشتری متفاوت باشد، این احتمال هست که یکی از آنها تسلیم شده (در حالیکه دیگری هنوز منتظر است) و ارتباط را بطور نامنتظره قطع کند. مشکل دیگر بروز توفانهای ایمیل است. اگر ماشین ۱ دارای یک لیست پستی بنام A، و ماشین ۲ دارای یک لیست پستی بنام B باشند، و هر لیست عضو لیست مقابل باشد، توفانی پایان ناپذیر از ایمیل های تکراری به راه خواهد افتاد (که فقط با دخالت سرپرست سیستم می تواند قطع شود).

برای حل این قبیل مشکلات، ویرایش گسترش یافته SMTP (موسوم به ESMTP) در RFC 2821 تعریف شده است. اگر یک مشتری بخواهد بجای SMTP از این پروتکل استفاده کند، باید در شروع کار بجای HELO دستور EHLO را بکار ببرد. اگر دستور EHLO از طرف سرویس دهنده پذیرفته نشود، مشتری می فهمد که با یک سرویس دهنده SMTP معمولی سروکار دارد و باید از همان روش سابق استفاده کند. اما اگر EHLO پذیرفته شد، مشتری اجازه دارد دستورات این پروتکل را بکار ببرد.

۵-۲-۷ تحویل نهایی

تا اینجا فرض ما بر این بود که تمام کاربران شبکه ماشینهایی با قابلیت ارسال و دریافت ایمیل دارند. همانطور که دیدید، فرستنده باید یک اتصال TCP به ماشین گیرنده برقرار کرده، و ایمیل خود را به آن بفرستد. این روش سالها بخوبی کار می کرد، چون تمام ماشینهای آرپانت (و بعدها اینترنت) کامپیوترهایی بودند که همیشه روی خط بودند و می توانستند اتصالات TCP را بپذیرند.

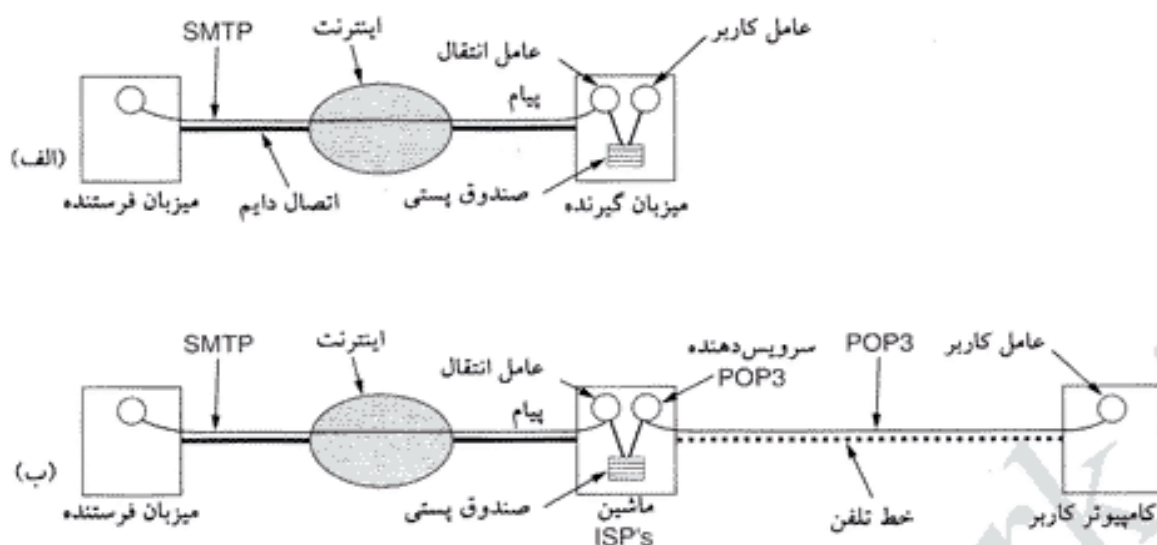
این وضعیت با ظهور کاربرانی که برای دسترسی اینترنت مجبور بودند از طریق مودم و با واسطه یک ISP اقدام کنند، تغییر کرد. مشکل این بود که: اگر در لحظه ای که Elinor می خواهد برای Carolyn ایمیل بفرستد، Carolyn روی خط نباشد، چه خواهد شد؟ در این حالت Elinor قادر به برقراری ارتباط TCP با Carolyn نیست، و نمی تواند پروتکل SMTP را اجرا کند.

ساده ترین راه حل آن است که یک عامل انتقال پیام در محل ISP ایجاد شود، و ایمیلها را پس از دریافت از فرستنده در صندوق پستی گیرنده (که در همان ISP قرار دارد) ذخیره کند. از آنجائیکه این عامل انتقال پیام می تواند ۲۴ ساعته روی خط باشد، همیشه می توان به آن ایمیل فرستاد.

POP3

متأسفانه این راه حل یک مشکل کوچک دارد: کاربران چگونه باید ایمیل های خود را از عامل انتقال مستقر در ISP بگیرند؟ برای حل این مسئله به پروتکل جدیدی نیاز داریم که PC کاربر را به یک عامل انتقال پیام (از ISP) تبدیل کند. یکی از این پروتکل ها، که در RFC 1939 تعریف شده، POP3 (پروتکل دفتر پستی ویرایش ۳ - Post Office Protocol Version 3) نام دارد.

به شکل ۷-۱۵ نگاه کنید؛ در تصویر (الف) وضعیتی که باید وجود داشته باشد، را می بینید: فرستنده و گیرنده



شکل ۷-۱۵. (الف) وضعیتی که فرستنده و گیرنده دسترسی دائم به اینترنت دارند، و عامل کاربر و عامل انتقال پیام هر دو روی یک ماشین اجرا می شوند. (ب) خواندن ایمیل در حالتی که گیرنده از طریق تلفن و با واسطه یک ISP به اینترنت وصل می شود.

هر دو دائماً روی خط هستند. در شکل ۷-۱۵ (ب) اوضاع به این خوبی نیست: فرستنده همیشه روی خط است، ولی گیرنده چنین نیست.

پروتکل POP3 کار خود را زمانی شروع می کند که کاربر برنامه ایمیل خوان را باز می کند. برنامه ایمیل خوان با ISP تماس گرفته (البته اگر این تماس از قبل وجود نداشته باشد)، و یک اتصال TCP به پورت 110 ماشین عامل انتقال پیام برقرار می کند. بعد از برقراری ارتباط، پروتکل POP3 سه مرحله را طی می کند:

۱. احراز هویت (authorization).
۲. تراکنش (transaction).
۳. به روز در آوردن (update).

در مرحله احراز هویت کاربر باید هویت واقعی خود را به عامل انتقال پیام بشناساند. در مرحله تراکنش، کاربر ایمیل های خود را از صندوق پستی خوانده، و آنها را برای حذف شدن علامتگذاری می کند. این ایمیل ها در مرحله بعد (به روز در آوردن) حذف می شوند. برای دیدن مراحل کار، فرمان زیر را اجرا کنید:

```
telnet mail.isp.com 110
```

(که در آن *mail.isp.com* نام DNS سرویس دهنده ایمیل ISP است.) برنامه telnet یک اتصال TCP با پورت 110 (که سرویس دهنده POP3 به آن گوش می کند) برقرار می سازد. بعد از برقراری ارتباط، سرویس دهنده یک پیام ASCII فرستاده، و آمادگی خود را اعلام می کند. این پیام معمولاً با OK+ شروع می شود، و جمله کوتاه دیگری بعد از آن می آید. در شکل ۷-۱۶ مکالمه یک مشتری با سرویس دهنده POP3 را ملاحظه می کنید (در اینجا هم پیامهای مشتری را با C: و پاسخهای سرویس دهنده را با S: مشخص کرده ایم).

در مرحله احراز هویت، مشتری ابتدا نام کاربر (username) و سپس کلمه عبور (password) خود را می فرستد. اگر مشتری بتواند این مرحله را با موفقیت پشت سر گذارد، می تواند با ارسال دستور LIST فهرستی از پیامهای موجود در صندوق پستی خود را دریافت کند (یک پیام در هر خط، که طول آنها نیز مشخص شده است).

پایان این لیست با یک نقطه (.) مشخص می شود.

```

S: +OK POP3 server ready
C: USER carolyn
S: +OK
C: PASS vegetables
S: +OK login successful
C: LIST
S: 1 2505
S: 2 14302
S: 3 8122
S: .
C: RETR 1
S: (sends message 1)
C: DELE 1
C: RETR 2
S: (sends message 2)
C: DELE 2
C: RETR 3
S: (sends message 3)
C: DELE 3
C: QUIT
S: +OK POP3 server disconnecting

```

شکل ۷-۱۶. استفاده از POP3 برای گرفتن پیامهای ایمیل.

پس از آن مشتری می تواند برای دریافت پیامها از فرمان *RETR* استفاده کرده، و آنها را با *DELE* برای حذف علامتگذاری کند. وقتی تمام پیامها دریافت (و احتمالاً برای حذف علامتگذاری) شدند، مشتری می تواند با فرمان *QUIT* وارد مرحله بعدی (به روز در آوردن) شود. بعد از آن که سرویس دهنده پیامهای علامتگذاری شده را حذف کرد، یک پیام تصدیق به مشتری فرستاده و ارتباط TCP را قطع می کند.

با اینکه پروتکل POP3 می تواند ایمیل ها را بصورت دسته جمعی یا تکی گرفته و آنها را روی سرویس دهنده باقی بگذارد، اغلب برنامه های ایمیل همه چیز را از روی سرویس دهنده خوانده و سپس صندوق پستی را خالی می کنند. در این حالت تنها کپی پیامها در دست خود کاربر است، و اگر روزی کامپیوتر وی صدمه ببیند، همه ایمیل ها از بین خواهند رفت.

اجازه دهید یک بار دیگر روش ارسال و دریافت ایمیل را برای کاربران ISP بطور خلاصه مرور کنیم. Elinor در برنامه ایمیل خود (یعنی، عامل کاربر) یک نامه برای Carolyn می نویسد، و روی دکمه Send کلیک می کند. برنامه ایمیل این پیام را گرفته و به عامل انتقال پیام ISP (که Elinor مشتری آن است) تحویل می دهد. عامل انتقال پیام آدرس گیرنده نامه (*carolyn@xyz.com*) را خوانده، و از یک DNS کمک می گیرد تا رکورد *MX* ناحیه *xyz.com* را پیدا کند. در پاسخ به این درخواست، DNS نام DNS سرویس دهنده ایمیل ناحیه *xyz.com* را برمی گرداند. عامل انتقال پیام دوباره به کمک همان DNS آدرس IP این سرویس دهنده را درخواست می کند. پس از دریافت این آدرس IP، عامل انتقال پیام یک اتصال TCP به پورت 25 (سرویس دهنده SMTP) برقرار می کند، و با استفاده از دستورات SMTP (مانند آنچه در شکل ۷-۱۴ دیدید) پیام را به صندوق پستی Carolyn فرستاده، و سپس ارتباط TCP را قطع می کند.

پس از مدتی، Carolyn (از خواب بیدار شده و) PC خود را روشن کرده، و مثل همیشه قبل از هر کاری به ISP وصل می شود تا ایمیل هایش را چک کند. برنامه ایمیل در بدو شروع یک اتصال TCP به پورت 110 سرویس دهنده ایمیل ISP (سرویس دهنده POP3) برقرار می کند. نام DNS یا آدرس IP این ماشین در همان بدو عضویت Carolyn در ISP در اختیار وی قرار داده می شود، تا آنرا در برنامه ایمیل خود وارد کند (و معمولاً نیازی به پا در میانی DNS نیست). پس از برقراری اتصال به پورت 110، برنامه ایمیل پروتکل POP3 را اجرا کرده، و (مانند آنچه در شکل ۷-۱۶ دیدید) پیامهای رسیده را از صندوق پستی Carolyn خوانده و در کامپیوتر وی ذخیره می کند (در پایان کار هم، اتصال TCP را قطع می کند). در اینجا می توان ارتباط تلفنی با ISP را هم قطع کرد (که البته نباید در این کار عجله کنید، چون برای فرستادن پاسخ نامه ها باز هم به آن احتیاج پیدا خواهد کرد).

IMAP

برای کاربری که فقط با یک ISP کار می کند و همیشه هم از یک PC به آن وصل می شود، POP3 بهترین گزینه است (بخصوص که ساده و کارآمد هم هست). ولی در دنیای کامپیوتر اصلی بدیهیست که می گوید، «وقتی چیزی دارد خوب کار می کند، بلافاصله یکی پیدا می شود که بیشتر می خواهد.» برای ایمیل هم همین اتفاق افتاد. برای مثال، خیلی از مردم هستند که فقط یک آدرس ایمیل دارند، و می خواهند هر کجا که هستند (در خانه، در مدرسه، در محل کار، و یا در سفر) با همان آدرس کار کنند. با اینکه POP3 می تواند از عهده این کار بر آید، اما کاربر بزودی متوجه می شود که ایمیل هایش روی چندین کامپیوتر پراکنده شده است (کامپیوترهایی که شاید بعضی از آنها حتی متعلق به وی نباشد).

این مشکل POP3 منجر به ارائه راه حلی بنام IMAP (پروتکل دسترسی پیام اینترنتی - Internet Message Access Protocol) شد، که در RFC 2060 تعریف شده است. برخلاف POP3 که اساساً فرض می کند کاربر تمام پیامهایش را به کامپیوتر خود منتقل کرده و سپس ارتباط با اینترنت را قطع می کند، IMAP پیامها را برای همیشه روی کامپیوتر سرویس دهنده نگه داشته و آنها را در چندین صندوق پستی حفظ می کند. IMAP دارای مکانیزمهای پیشرفته ای برای خواندن پیامهاست، که حتی اجازه می دهند کاربر فقط بخشهایی از یک پیام بخواند؛ فقط تصور کنید که یک پیام مهم برای کاربر بیچاره ما - که با یک مودم کند عهده بوق به اینترنت وصل می شود - آمده، و این پیام ضمن داشتن متنی مهم دارای یک پیوست بزرگ نیز هست - IMAP به کاربر ما اجازه می دهد تا فقط متن پیام را خوانده و از خبر پیوست آن بگذرد. از آنجائیکه در IMAP فرض بر آن است که پیامها به کامپیوتر کاربر منتقل نمی شوند، مکانیزمهایی برای نوشتن ایمیل، از بین بردن آنها، و یا مدیریت پیامهای رسیده (مانند دسته بندی آنها بر حسب فرستنده) روی سرویس دهنده ایمیل در نظر گرفته شده است.

یکی از قابلیت های جالب IMAP دسته بندی و نمایش پیامهای رسیده بر حسب فرستنده ایمیل - یا ویژگیهای دیگر - است (بر خلاف شکل ۷-۸ که تمام ایمیل ها پشت سر هم ردیف می شوند). IMAP (برخلاف POP3) فقط پروتکلی برای دریافت ایمیل نیست، بلکه می تواند ارسال نامه ها را هم انجام دهد. روش کار IMAP بسیار شبیه POP3 (شکل ۷-۱۶) است، با این تفاوت که دستورات بسیار متنوعتری دارد. سرویس دهنده IMAP به پورت 143 گوش می کند. در شکل ۷-۱۷ مقایسه ای بین POP3 و IMAP آورده شده است. اما همین جا باید تذکر داد که تمام ISP ها (و همچنین برنامه های ایمیل) از هر دو پروتکل پشتیبانی نمی کنند (هنگام انتخاب ISP و برنامه ایمیل دقت کنید که از چه پروتکل هایی پشتیبانی می کنند).

امکانات سیستم تحویل نامه

اغلب سیستم های ایمیل، صرف نظر از اینکه از POP3 یا IMAP برای گرفتن نامه ها استفاده کنید، امکاناتی برای

پردازش پیامها دارند. یکی از این امکانات فیلتر کردن پیامهاست. این فیلترها قواعدی هستند که روی ایمیلهای رسیده عمل می کنند. هر قاعده یک شرط دارد و عملی را انجام می دهد. برای مثال، یک قاعده می تواند بگوید «اگر پیامی از رئیس رسید، آنرا در صندوق شماره ۱ قرار بده» یا «اگر پیام از گروهی مشخصی از دوستان بود، آنرا در صندوق شماره ۲ قرار بده» و یا «اگر کلمه خاصی در موضوع پیام بود، آنرا بکلی دور بینداز».

ویژگی	POP3	IMAP
سطح تعریف پروتکل	RFC 1939	RFC 2060
پورت	110	143
محل ذخیره شدن ایمیل	کاربر PC	سرویس دهنده
محل خوانده شدن ایمیل	خارج خط	روی خط
زمان اتصال	کم	زیاد
استفاده از منابع سرویس دهنده	حداقل	گسترده
صندوق پستی های متعدد	خیر	بلی
مسئول گرفتن پشتیبان	کاربر	ISP
مناسب برای کاربران سیار	خیر	بلی
کنترل بار کردن محتویات	کم	زیاد
بار کردن قسمتی از پیامها	خیر	بلی
مشکل محدودیت دیسک	خیر	گاهی
پایه سازی ساده است	بلی	خیر
پشتیبانی گسترده	بلی	در حال و شد

شکل ۷-۱۷. مقایسه ای بین POP3 و IMAP.

برخی از ISP ها فیلترهایی دارند که بطور خودکار ایمیلهای رسیده را به دو دسته «مهم» و «هرز» (spam یا junk - نامه هایی که فقط بدرد سطل آشغال می خورند) تقسیم کرده، و آنها را در صندوقهای مخصوص قرار می دهند. این فیلترها ابتدا با چک کردن فرستنده نامه ها شروع می کنند (تا مطمئن شوند از مردم آزارهای حرفه ای نباشند). اگر چند صد کاربر یک ISP نامه هایی با یک موضوع دریافت کنند، فرستنده آن با احتمال زیاد یک هرزنویس است. برای تشخیص این قبیل نامه ها تکنیکهای دیگری نیز وجود دارد.

یک دیگر از امکانات سیستمهای تحویل ایمیل هدایت (موقتی) نامه های رسیده برای یک کاربر به آدرسی دیگر است. این آدرس حتی می تواند شماره ای در یک سیستم فراخوان (pager) باشد، که در این حالت کاربر بلافاصله بعد از دریافت هر ایمیل موضوع آنرا در دستگاه فراخوان خود مشاهده خواهد کرد.

امکان دیگر پاسخ خودکار در صورت عدم حضور در محل است (که به آن دیمون تعطیلات - vacation daemon - می گویند). با این ویژگی فرد می تواند هنگام رفتن به تعطیلات یا مأموریت بطور خودکار به ایمیلهای رسیده پاسخی مانند زیر بدهد:

سلام، من تا ۲۴ ام آگوست در تعطیلات تابستانی هستم. امیدوارم شما هم تعطیلات خوبی داشته باشید.

در این قبیل پیامها حتی می توانید مشخص کنید که در صورت اضطرار با کجا باید تماس گرفته شود. در برخی از سیستمهای ایمیل، دیمون تعطیلات می تواند تشخیص دهد که قبلاً برای چه کسانی پیام فرستاده، تا از ارسال پیام تکراری برای آنها اجتناب شود. برخی از این دیمون ها حتی می توانند تشخیص دهند که نامه وارده به یک لیست پستی فرستاده شده، که در این صورت از فرستادن جواب آماده خودداری خواهند کرد.

نویسنده این کتاب شخصاً با یک سیستم پاسخ خودکار فوق العاده جالب از جانب شخصی روبرو شده است،

که ادعا می کرد روزی ۶۰۰ ایمیل می گیرد. (اجازه دهید برای حفظ حریم شخصی افراد، این فرد را با نام مستعار جان معرفی کنیم).

جان یک رویات ایمیل در کامپیوتر خود نصب کرده که تمام پیامهای رسیده را چک می کند. اگر این ایمیل از کسی باشد که قبلاً با جان تماس نداشته (و عبارت دیگر تازه وارد باشد)، پیام خودکاری برای وی ارسال می شود که (ضمن عذرخواهی از عدم امکان پاسخ فردی) سندی حاوی تمام اطلاعات مورد نیاز (از جمله آدرس، شماره تلفن، شماره فکس، و طریقه تماس با شرکت جان) در آن آمده است. از اطلاعات مفصل دیگری که جان درباره خود در این پاسخ خودکار آورده حرفی نمی زنم، اما بنظر می رسد که روش جان یکی از نمونه های افراط در استفاده از امکانات باشد.

پست وب

آخرین مبحثی که در سیستمهای ایمیل می توان به آن اشاره کرد، پست وب (Webmail) است. همانطور که شاید قبلاً دیده باشید، برخی از سایت های معروف و بزرگ مانند هات میل (Hotmail.com) و یاهو (Yahoo.com) سرویسهای ایمیل مجانی در اختیار بازدیدکنندگان قرار می دهند. در این سیستمها یک عامل انتقال پیام معمولی وجود دارد، که به پورت 25 (پورت SMTP) گوش می کند. برای وصل شدن به، مثلاً، هات میل باید ابتدا رکورد MX آنرا بدست آورید؛ در یک سیستم یونیکس می توانید از دستور زیر استفاده کنید:

```
host -a -v hotmail.com
```

پس از آن، با فرض اینکه سرویس دهنده ایمیل هات میل `mx10.hotmail.com` نام داشته باشد، می توانید با دستور

```
telnet mx10.hotmail.com 25
```

یک اتصال TCP به پورت 25 آن برقرار کرده، و از فرمانهای SMTP برای ارسال پیامهای خود استفاده کنید. (تا اینجا که چیز غیرعادی وجود ندارد؛ فقط مواظب باشید که سر این کامپیوترها خیلی شلوغ است، و معمولاً باید چند بار سعی کنید تا بتوانید با آنها تماس بگیرید).

بخش جالب در این سرویسها قسمت تحویل نامه است. معمولاً وقتی وارد صفحه وب سرویس ایمیل می شوید، فرمی ظاهر می شود که باید نام کاربر و کلمه عبور خود را در آن وارد کنید. وقتی دکمه Sign In را کلیک می کنید، این اطلاعات به سرویس دهنده فرستاده شده و در آنجا با اطلاعات موجود تطبیق داده می شود. اگر هویت شما تأیید شود، سرویس دهنده صندوق پستی را یافته و محتویات آنرا بصورت یک صفحه ای شبیه شکل ۷-۸، ولی با فرمت HTML، نمایش می دهد. اغلب امکانات یک برنامه ایمیل (مانند خواندن نامه، نوشتن نامه، و حذف آنها) در این صفحه وب نیز وجود دارد.

۳-۷ تارنمای جهانی - وب

تارنمای جهانی (World Wide Web)، یا بطور مختصر وب، ساختاریست برای دسترسی به سند های پیوند شده (linked documents) در اینترنت. ظرف ده سال، وب از یک ابزار ارتباطی فیزیکدانها به چیزی تبدیل شده که بسیاری از مردم آنرا همان «اینترنت» می دانند. علت اصلی محبوبیت وب در ظاهر گرافیکی آن ریشه دارد، که باعث شده تا میلیون ها کاربر تازه کار بتوانند پس ادگی از آن استفاده کنند. حجم فوق العاده اطلاعات موجود در وب را نیز می توان یکی دیگر از علل موفقیت آن دانست.

وب (که به WWW نیز معروف است) به سال ۱۹۸۹ در مرکز اروپایی فیزیک هسته ای موسوم به CERN متولد شد. در این مرکز دهها تیم تحقیقاتی از سراسر اروپا به کار روی فرضیه های فیزیک ذرات مشغول هستند.

اغلب این آزمایشات چنان پیچیده اند که دهها دانشمند از چندین کشور مختلف سالها روی آنها کار می کنند. همین پراکندگی دانشمندان این مرکز در کشورهای مختلف، و لزوم ارتباط پیوسته آنها منجر به تولد وب شد. ایده سندهای پیوند شده را اولین بار یکی از فیزیکدانان CERN بنام تیم برنرزیلی در مارس ۱۹۸۹ مطرح کرد. اولین نمونه این برنامه ۱۸ ماه بعد عملیاتی شد، و در کنفرانس Hypertext '91 که در دسامبر ۱۹۹۱ در سان آنتونیو، تگزاس بر پا شده بود، به نمایش در آمد.

این نمایش و قابلیت های بالقوه یک مرورگر آبرمتن (hypertext browser) نظر محققان را بخود جلب کرد، و یکی از همین افراد بنام مارک آندرسن از دانشگاه ایلینویز اولین مرورگر گرافیکی را در فوریه ۱۹۹۳ به بازار عرضه کرد - این مرورگر موزائیک (Mosaic) نام داشت. موزائیک چنان موفقیتی بدست آورد که یک سال بعد آندرسن شرکتی بنام نت اسکپ (Netscape)، با هدف توسعه نرم افزارهای وب، تأسیس کرد. وقتی نت اسکپ در سال ۱۹۹۵ وارد بورس شد، سرمایه گذاران به تصور تولد یک میکروسافت دیگر ۱/۵ میلیارد دلار بابت خرید سهام آن پول پرداخت کردند - و این همه پول (که یک رکورد محسوب می شد) در شرایطی پرداخت شد که نت اسکپ شرکت کوچکی بود با فقط یک محصول، و حتی اعلام کرده بود که تا مدتها هیچگونه سوددهی نخواهد داشت. طی سه سال بعد نت اسکپ (با مرورگر ناویگیتور - Navigator) و میکروسافت (با مرورگر اینترنت اکسپلورر - Internet Explorer) درگیر جنگی بودند که به «جنگ مرورگرها» معروف شد، و در آن هر شرکت سعی می کرد با ارائه محصول بهتر دیگری را از میدان بدر کند. در سال ۱۹۹۸، شرکت America Online نت اسکپ را به قیمت ۴/۲ میلیارد دلار خرید، و به عمر کوتاه آن بعنوان یک شرکت مستقل پایان داد.

در ۱۹۹۴، CERN و M.I.T موافقتنامه ای برای تأسیس کنسرسیوم WWW (که به W3C معروف است) امضا کردند. وظیفه این سازمان توسعه وب، استاندارد کردن پروتکلها، و تسهیل ارتباط بین سایت ها بود، و برنرزیلی نیز بعنوان اولین رئیس آن انتخاب شد. از آن زمان تاکنون صدها دانشگاه و شرکت بزرگ به این کنسرسیوم پیوسته اند. با اینکه صدها کتاب خوب درباره وب چاپ شده، اما بهترین محل برای کسب اطلاعات درباره وب خود وب است. آدرس صفحه اصلی سایت کنسرسیوم وب www.w3.org است. در این سایت می توانید صدها صفحه، سند و لینک درباره کنسرسیوم وب و فعالیتهای آن ببینید.

۱-۳-۷ بررسی ساختاری

از دید یک کاربر، وب عبارتست از مجموعه ای فوق العاده بزرگ از میلیونها سند یا صفحه وب (web page). هر صفحه وب می تواند لینکهایی به صفحات دیگر (در سراسر دنیا) داشته باشد، که برای رفتن به این صفحات کافیست روی لینک آنها کلیک کنید. این کار را می توان بدفعات نامحدود تکرار کرد. ایده صفحاتی که به یکدیگر اشاره می کنند، و اکنون آن را با عنوان آبرمتن (hypertext) می شناسیم، مدتها قبل از اختراع اینترنت و در سال ۱۹۴۵ توسط وانوار بوش (استاد مهندسی برق دانشگاه M.I.T) ابداع شده بود.

برای دیدن صفحات وب از برنامه ای بنام مرورگر (browser) استفاده می کنیم؛ معروفترین مرورگرهای موجود اینترنت اکسپلورر و نت اسکپ ناویگیتور هستند. مرورگر بعد از آوردن صفحه خواسته شده از وب، محتویات آنرا تفسیر کرده و با فرمت صحیح نمایش می دهد. در شکل ۷-۱۸ (الف) نمونه ای از یک صفحه وب ساده را ملاحظه می کنید. مانند بسیاری از صفحات وب، این صفحه هم با یک عنوان ساده شروع شده، و بعد از مقداری اطلاعات (محتویات صفحه) به یک آدرس ایمیل ختم می شود. مرورگر لینکهایی موجود در صفحه را، که به آنها آبرلینک (hyperlink) گفته می شود، بگونه ای از سایر قسمتهای متن متمایز می کند (با زیرخط، رنگ متفاوت، و یا هر دو). برای دنبال کردن یک لینک، کرسر ماوس را روی لینک موردنظر برده (که معمولاً با اینکار شکل کرسر عوض

می شود، و کلیک کنید. با اینکه مرورگرهای کاملاً متنی هم وجود دارد (مانند لینکس - Lynx)، اما به علت محبوبیت فراگیر مرورگرهای گرافیکی در ادامه این بخش فقط درباره آنها صحبت خواهیم کرد. (اخیراً مرورگرهای کلامی نیز توسعه داده شده اند، که به دستورات شفاهی کاربر عکس العمل نشان می دهند.)

WELCOME TO THE UNIVERSITY OF EAST PODUNK'S WWW HOME PAGE

- Campus Information
 - ☐
 - ☐
 - ☐
 - ☐
- Academic Departments
 - ☐
 - ☐
 - ☐
 - ☐
 - ☐

Webmaster@eastpodunk.edu

(الف)

THE DEPARTMENT OF ANIMAL PSYCHOLOGY

- - Personnel
 - ☐
 - ☐
 - ☐
 - - Our most popular courses
 - ☐
 - ☐
 - ☐
 - ☐

Webmaster@animalpsyc.eastpodunk.edu

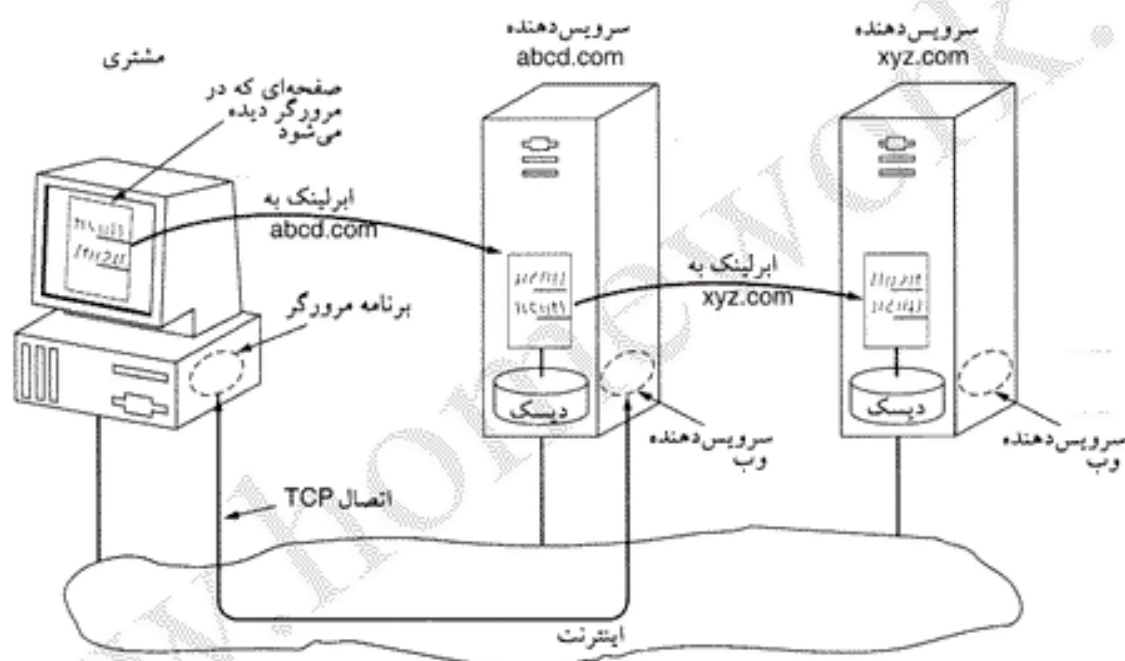
(ب)

شکل ۷-۱۸. (الف) یک صفحه وب. (ب) وقتی کاربر روی لینک Department of Animal Psychology کلیک کند، به اینجا خواهد رسید.

اگر به روانشناسی حیوانات علاقمند باشید، می توانید روی لینک Department of Animal Psychology کلیک کنید تا به این صفحه برسید. یا این کار، مرورگر صفحه مشخص شده را از سایت وب آورده و نمایش می دهد (شکل ۷-۱۸ ب). در این صفحه هم جملاتی که زیرخطدار دارند لینک هستند، و با کلیک کردن آنها می توان به صفحات دیگر رفت. صفحه ای که یک لینک به آن اشاره می کند، می تواند روی همان ماشین باشد یا در

کامپیوتری آن سوی دنیا - حدس آن برای کاربر ممکن نیست، چون مرورگر بدون کمک وی صفحات را می آورد. اگر بعد از دیدن چند صفحه به صفحه اصلی برگردید، لینکهایی که دنبال شده اند را با رنگ دیگری مشاهده خواهید کرد. دقت کنید که با کلیک کردن جمله *Campus Information* در صفحه اصلی هیچ اتفاقی نخواهد افتاد، چون این یک لینک نیست (زیرخط ندارد).

طرز کار مدل وب در شکل ۷-۱۹ نشان داده شده است. در اینجا، مرورگر در حال نمایش یک صفحه روی کامپیوتر مشتری است. وقتی کاربر روی یک جمله که به صفحه ای روی ماشین *abcd.com* لینک شده کلیک می کند، مرورگر لینک را دنبال کرده و از ماشین *abcd.com* می خواهد که آن صفحه را برایش بفرستد. اگر در همین صفحه لینک دیگری به یک صفحه در ماشین *xyz.com* وجود داشته باشد و کاربر آنرا کلیک کند، مرورگر درخواست خود را به سرویس دهنده *xyz.com* می فرستد (و الی آخر).



شکل ۷-۱۹. بخشی از مدل وب.

سمت مشتری

اجازه دهید فعل و انفعالات سمت مشتری (client-side) را با جزئیات بیشتر بررسی کنیم. یک مرورگر در واقع برنامه ایست که می تواند حرکات ماوس و کلیک های آنرا تشخیص داده و صفحات وب را نمایش دهد. وقتی یک آیتم انتخاب می شود، مرورگر آبریلینک را دنبال کرده و صفحه وب را می آورد. بنابراین، آبریلینک ها باید راهی برای نامگذاری و مشخص کردن صفحات وب داشته باشند. صفحات وب با استفاده از URL (پاینده همسان منابع - Uniform Resource Locator) نامگذاری می شوند. در زیر یک URL نوعی را می بینید:

<http://www.abcd.com/products.html>

در قسمتهای آینده بیشتر درباره URL ها توضیح خواهیم داد. فعلاً کافایت بدانید که، یک URL سه بخش دارد: نام پروتکل (*http*)، نام DNS ماشینی که صفحه روی آن قرار دارد (*www.abcd.com*)، و نام فایل صفحه وب (*products.html*).

وقتی کاربر روی آبریلینک کلیک می کند، مرورگر بایستی مراحل را برای آوردن صفحه وب طی کند. فرض

کنید کاربر بعد از کمی گشت و گذار در وب به لینک صفحه اصلی ITU (که URL آن <http://www.itu.org/home/index.html> است) رسیده و می خواهد آنرا ببیند. اجازه دهید ببینیم بعد از انتخاب این لینک، مرورگر چه کارهایی انجام می دهد.

۱. مرورگر با بررسی آیتم انتخاب شده، URL آنرا بدست می آورد.
۲. مرورگر از DNS خود آدرس IP سایت www.itu.org را می پرسد.
۳. DNS آدرس 156.106.192.32 را برمی گرداند.
۴. مرورگر یک اتصال TCP به پورت 80 ماشین 156.106.192.32 برقرار می کند.
۵. سپس روی این لینک درخواستی برای فایل [/home/index.html](http://home/index.html) به آن می فرستد.
۶. سرورس دهنده www.itu.org فایل [/home/index.html](http://home/index.html) را به مرورگر می گرداند.
۷. اتصال TCP قطع می شود.
۸. مرورگر متن صفحه [/home/index.html](http://home/index.html) را نمایش می دهد.
۹. مرورگر تصاویر صفحه را هم آورده و نمایش می دهد.

اغلب مرورگرها مراحل کار خود را در خط وضعیت (پانین پنجره مرورگر) نشان می دهند. بدین ترتیب در مواقعی که مشکلی پیش می آید، کاربر می تواند ببیند که کدام مرحله مشکل ساز شده است (DNS دیر جواب می دهد، سرورس دهنده www سرش شلوغ است، یا ترافیک شبکه زیاد است).

برای آن که مرورگر بتواند صفحات وب را نمایش دهد، باید فرمت آنها را بداند. بهمین دلیل تمام صفحات وب با یک زبان استاندارد بنام HTML نوشته می شوند، تا هر مرورگری بتواند آنها را تفسیر کند. در ادامه درباره زبان HTML بیشتر صحبت خواهیم کرد.

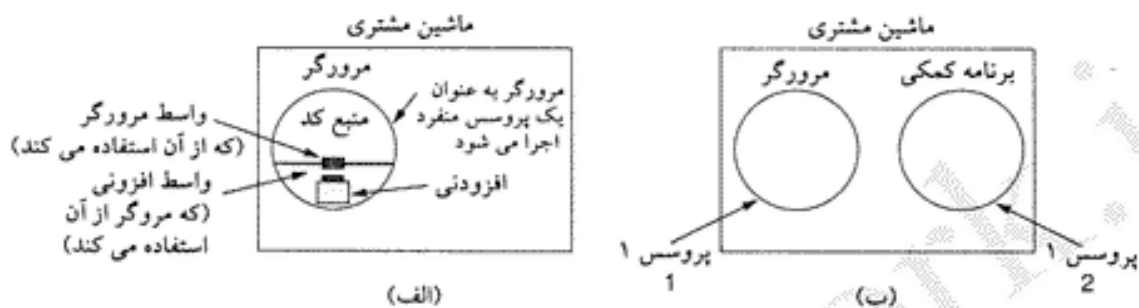
با اینکه مرورگرها اساساً چیزی نیستند جز مفسر فایل های HTML، اغلب آنها دکمه های دیگری نیز دارند که به کاربر در ویگردی (web surfing) کمک می کند. دکمه های عقب (Back - برای برگشت به صفحه قبلی)، جلو (Forward - برای برگشت به صفحه بعدی) و خانه (Home - برای برگشت به صفحه شروع) در اغلب مرورگرها وجود دارد. بسیاری از مرورگرها دارای دکمه ها یا منو هایی برای علامتگذاری صفحات (بهمنظور ردیابی سریع آنها) نیز هستند. ذخیره و چاپ صفحات وب از دیگر امکانات مرورگرهاست، و در ضمن کاربر می تواند محیط کار مرورگر را مطابق نیاز خود تنظیم کند.

صفحات وب علاوه بر متن معمولی و آبرلینک، می توانند آیکون، طرح، و تصویر نیز داشته باشند، که هر یک از آنها می تواند به صفحات دیگر لینک شده باشد (و با کلیک کردن آنها، مرورگر صفحه مشخص شده را باز خواهد کرد). حتی در برخی موارد قسمتهای مختلف یک تصویر می تواند به صفحات مختلفی لینک شده باشد. تمام صفحات وب HTML نیستند؛ یک صفحه وب می تواند حاوی سند PDF، تصاویری با فرمتهای GIF و JPEG، موسیقی با فرمت MP3، ویدئو با فرمت MPEG، و یا صداها نوع فایل دیگر باشد. از آنجائیکه یک صفحه استاندارد HTML می تواند لینکهایی به هر کدام از این فایلها داشته باشد، مرورگر در برخورد با صفحاتی که فرمت آنها را نمی شناسد، مشکل خواهد داشت.

بجای اینکه هر روز مرورگرهای بزرگتری بسازیم که بتوانند فرمتهای جدید را بخوانند، اکثر مرورگرها از روش کلی تر و مؤثرتری استفاده می کنند. وقتی سرورس دهنده وب صفحه ای را برمی گرداند، اطلاعات اضافه ای را نیز درباره آن می دهد، که این اطلاعات شامل نوع MIME صفحه هم می شود (شکل ۷-۱۲ را ببینید). صفحات text/html (و چند نوع ساده دیگر) مستقیماً نمایش داده می شوند. اگر فرمت فایل یکی از این انواع شناخته شده نباشد، مرورگر برای نمایش آن با جدول انواع MIME مشورت می کند. در این جدول برای هر نوع MIME یک

نمایش دهنده (viewer) معرفی شده است.

نمایش دهنده ها بر دو نوعند: افزودنی ها، و برنامه های کمکی. افزودنی (plug-in) یک قطعه کد است، که مرورگر آنرا از روی دیسک خوانده و به قابلیت های خود اضافه می کند (شکل ۷-۲۰ الف). از آنجائیکه افزودنی در داخل مرورگر اجرا می شود، به صفحه وب دسترسی داشته و می تواند ظاهر آنرا عوض کند. بعد از آن که افزودنی کارش را انجام داد (معمولاً بعد از اینکه کاربر به صفحه دیگری رفت)، از حافظه مرورگر پاک می شود.



شکل ۷-۲۰. (الف) یک افزودنی. (ب) یک برنامه کمکی.

هر مرورگر دارای مجموعه ای از روالهاست که افزودنی باید آنها را پیاده سازی کند تا مرورگر بتواند با آن تماس برقرار کند. برای مثال، روالی باید وجود داشته باشد که مرورگر بتواند از طریق آن داده ها را به افزودنی بدهد. به این مجموعه از روالها واسط افزودنی (plug-in interface) گفته می شود، و برای هر مرورگر باید واسطی خاص نوشته شود. علاوه بر آن، مرورگر نیز تعدادی از روالهای خود را در اختیار افزودنی می گذارد تا بتواند به آن سرویس بدهد (روالهای تخصیص حافظه، نمایش وضعیت، و گرفتن پارامترها از این دسته اند). به این روالها واسط مرورگر (browser interface) می گویند.

قبل از استفاده از افزودنی باید آنرا نصب کرد، و این کار بر عهده کاربر است (که به سایت وب افزودنی رفته، آنرا بار کرده و نصب کند). در ویندوز، این فایل معمولاً یک فایل فشرده خود-استخراج (با پسوند .exe) است. وقتی کاربر روی این فایل دو-کلیک کند، برنامه کوچکی که در دل فایل فشرده تعبیه شده، اجرا شده و بعد از باز کردن فایل افزودنی آنرا در دایرکتوری افزودنی های مرورگر کپی می کند. سپس ارتباط ارگانیک نوع MIME افزودنی با آن را برقرار می کند. در سیستم های یونیکس، این فرآیند اغلب بر عهده یک اسکریپت پوسته (shell script) گذاشته می شود.

روش دیگر گسترش دادن قابلیت های مرورگر استفاده از برنامه کمکی (helper application) است؛ این یک برنامه کامل است که بعنوان یک پروسس مستقل اجرا می شود (شکل ۷-۲۰ ب). از آنجائیکه برنامه کمکی یک برنامه مستقل است، هیچ واسطی در اختیار مرورگر نمی گذارد و از سرویس های آن هم استفاده نمی کند. بجای آن، نام فضای موقتی که فایل در آن ذخیره شده را از مرورگر گرفته، محتویات صفحه را خوانده و نمایش می دهد. برنامه های کمکی معمولاً برنامه های بزرگ و مستقلی مانند Adobe Acrobat Reader (برای نمایش فایل های PDF) و Microsoft Word (برای نمایش فایل های DOC) هستند. برخی از برنامه های کمکی برای اجرا شدن به یک افزودنی کوچک نیاز دارند.

اغلب برنامه های کمکی از نوع MIME application استفاده می کنند. در این نوع MIME زیرنوع های بسیاری تعریف شده، که application/pdf (برای فایل های PDF) و application/msword (برای فایل های DOC) از آن جمله اند. بدین ترتیب، یک URL می تواند مستقیماً به فایل PDF یا DOC اشاره کند، و وقتی کاربر چنین

لینکی را کلیک کند، Acrobat یا Word بطور خودکار اجرا شده و محتویات فایل را نمایش می دهند. با این روش می توان بدون کوچکترین تغییر در مرورگر، کاری کرد که بتواند انواع نامحدودی از فایلها را نمایش دهد. اغلب سرویس دهنده های وب برای خواندن صداها نوع زیرنوع MIME پیکربندی شده اند، که روز به روز هم بر تعداد آنها افزوده می شود. البته برنامه های کمکی به نوع application محدود نیستند؛ برای مثال، برنامه Adobe Photoshop از زیرنوع image/x-photoshop و برنامه RealOne Player از زیرنوع audio/mp3 استفاده می کنند.

وقتی برنامه ای در ویندوز نصب می شود، خود را برای نمایش فایل MIME وابسته ثبت می کند. این مکانیزم وقتی چند برنامه برای نمایش یک نوع فایل (مثلاً، video/mpg) وجود داشته باشد، می تواند مشکل ساز شود (معمولاً برنامه ای که آخر نصب شده نمایش فایل را به انحصار خود در می آورد). در نتیجه، نصب برنامه های جدید می تواند رفتار مرورگر را در نمایش فایلها تغییر دهد.

در یونیکس، معمولاً این فرآیند خودکار نیست، و این کاربر است که فایلهای پیکربندی را به روز در می آورد. این روش کار بیشتری می برد، ولی دردسر آن هم کمتر است.

مرورگرها (علاوه بر صفحات وب) فایلهای محلی را نیز می توانند باز کنند، اما از آنجائیکه این فایلها دارای نوع MIME نیستند، مرورگر باید راهی برای تشخیص برنامه کمکی یا افزودنی نمایش دهنده آنها داشته باشد. برای این کار معمولاً از پسوند نام فایل استفاده می شود. برخی مرورگرها (علاوه بر نوع MIME و پسوند فایل) از محتویات خود فایل نیز برای تشخیص نوع آن استفاده می کنند. برای مثال، اینترنت اکسپلورر بیش از نوع MIME به پسوند فایل متکی است.

در این حالت هم برنامه های مختلفی که قادر به نمایش یک نوع فایل هستند، بر سر باز کردن آن با هم رقابت می کنند. در برنامه های حرفه ای کاربر می تواند وابستگی برنامه به انواع MIME را فعال یا غیرفعال کند. اما برنامه های زیادی هم هستند که هیچ اهمیتی به این مسائل نمی دهند، و خیلی ساده وابستگی فایل را به خود اختصاص می دهند.

توانایی مرورگر در نمایش انواع مختلف فایلها بسیار جالب است، اما در عین حال می تواند دردسر ساز باشد. برای مثال، وقتی اینترنت اکسپلورر یک فایل exe را می آورد، متوجه می شود که این فایل یک برنامه است و هیچ برنامه کمکی برای آن ندارد؛ محتملترین گزینه اجرای این برنامه است. اما این می تواند یک رخنه امنیتی بزرگ باشد. تنها کاری که یک سایت وب خرابکار باید انجام دهد، پنهان کردن لینک گد و ویروس پشت تصویری از یک ستاره سینما یا قهرمان ورزشی است. کاربر از همه جایی خبر روی تصویر مورد علاقه اش کلیک می کند، و با این کار ویروس مخرب را وارد کامپیوتر خود کرده و اجرا می کند. البته می توان اینترنت اکسپلورر را بگونه ای تنظیم کرد که هر برنامه ای را بطور خودکار اجرا نکند (و یا حداقل قبل از اجرای برنامه تأیید کاربر را بگیرد)، ولی بسیاری از کاربران مبتدی نمی دانند چگونه باید چنین تنظیمی را انجام دهند.

در سیستمهای یونیکس هم این اتفاق می تواند بیفتد، مشروط بر اینکه کاربر آگاهانه پوسته را بعنوان یک برنامه کمکی تعریف کرده باشد. خوشبختانه، این کار بقدری پیچیده است که احتمال اینکه تصادفی چنین اتفاقی بیفتد، بسیار ناچیز است (در واقع، فقط عده کمی هستند که می توانند چنین کاری را انجام دهند).

سمت سرویس دهنده

کمی هم از فعل و انفعالات سمت سرویس دهنده بگوئیم. همانطور که قبلاً دیدید، وقتی کاربر روی یک URL (یا آبرلینک) کلیک می کند، مرورگر هر آنچه را که بین http:// و / بعدی قرار دارد یک نام DNS تلقی کرده، و بدنبال آدرس IP آن می رود. بعد از بدست آوردن آدرس IP سرویس دهنده، مرورگر یک اتصال TCP به پورت 80 آن

برقرار می‌کند و بقیه URL را (که نام فایل صفحه وب است) به سرویس دهنده می‌فرستد؛ سرویس دهنده هم در جواب صفحه خواسته شده را به مرورگر برمی‌گرداند.

یک سرویس دهنده وب (با کمی تسامح) شبیه سرویس دهنده شکل ۶-۶ است. در هر دوی آنها مراحل که سرویس دهنده (در حلقه اصلی خود) طی می‌کند، مانند زیر است:

۱. اتصال TCP را از مشتری (مرورگر) قبول می‌کند.
۲. نام فایل درخواست شده را می‌گیرد.
۳. فایل را (از روی دیسک) می‌خواند.
۴. فایل را به مشتری برمی‌گرداند.
۵. اتصال TCP را قطع می‌کند.

سرویس دهنده‌های وب مدرن ویژگیهای بسیار بیشتری دارند، ولی وظیفه اصلی آنها همان است که در بالا گفتیم. یکی از مشکلات روش فوق اینست که سرویس دهنده برای هر درخواست باید به دیسک مراجعه کند. در نتیجه تعداد درخواستهایی که یک سرویس دهنده وب می‌تواند پاسخ دهد، به توانایی آن در مراجعه به دیسک بستگی دارد. زمان دسترسی در جدیدترین دیسکهای SCSI در حدود 5 msec است، و این یعنی حداکثر 200 درخواست در ثانیه (که البته اگر فایل بزرگ باشد، کمتر هم خواهد شد). این مقدار حتی برای یک سرویس دهنده وب متوسط هم بسیار کم است.

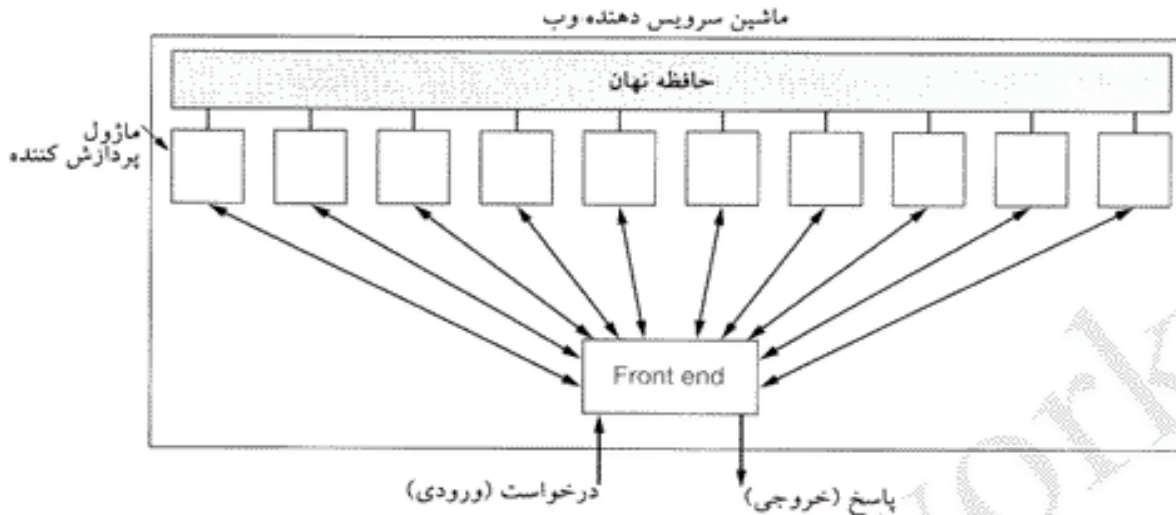
ساده‌ترین روش بهبود پاسخ سرویس دهنده وب (که در تمام سرویس دهنده‌های وب هم از آن استفاده می‌شود)، نگهداری آخرین n فایل درخواست شده در حافظه RAM است. سرویس دهنده وب قبل از رفتن به سراغ دیسک، این حافظه‌نهان (cache) را چک می‌کند. اگر فایل خواسته شده در حافظه نهان باشد، از همان جا برداشته شده و به مشتری تحویل داده می‌شود (و دسترسی دیسک انجام نخواهد شد). با اینکه این تکنیک به حافظه زیادی نیاز دارد و زمانی هم صرف جستجوی حافظه نهان می‌شود، اما تقریباً همیشه سریعتر از دسترسی مستقیم دیسک است.

قدم بعدی برای سریعتر کردن سرویس دهنده وب، طراحی آن بصورت چندرسمانی (multithreaded) است. در این تکنیک سرویس دهنده وب یک ماژول جلودار (front-end) - که درخواستهای رسیده را می‌پذیرد و k ماژول پردازش‌کننده (processing) دارد (شکل ۷-۲۱ را ببینید). تمام این $k + 1$ ریسمان متعلق به یک پروسس هستند، بنابراین به فضای آدرس آن دسترسی دارند. وقتی یک درخواست جدید به سرویس دهنده وب می‌رسد، ماژول جلودار آن را پذیرفته، مشخصات آنرا ثبت می‌کند، و سپس به یکی از ماژولهای پردازش‌کننده تحویل می‌دهد. (تکنیک دیگری نیز وجود دارد که در آن ماژول جلودار حذف شده، و ماژولهای پردازش‌کننده مستقیماً درخواستها را دریافت می‌کنند. اما در این روش بایستی مکانیزمی برای جلوگیری از تداخل پروسسها در نظر گرفته شود.)

ماژول پردازش‌کننده ابتدا حافظه نهان را چک می‌کند تا ببیند که فایل موردنظر در آنجا هست یا خیر. اگر آنجا باشد، اشاره‌گری را که به آن فایل اشاره می‌کند، به روز در می‌آورد. اگر در حافظه نهان نباشد، سراغ دیسک می‌رود و مقداری از آن را در حافظه نهان کپی می‌کند (که با احتمال زیاد مقداری از محتویات قدیمی حافظه نهان را از بین می‌برد). بعد از کپی کردن فایل در حافظه نهان، آنرا برای مشتری می‌فرستد.

مزیت این روش آن است که در زمانهایی که یکی از پروسسها در حال خواندن دیسک است (و کاری با CPU ندارد)، سایر پروسسها می‌توانند با استفاده از حافظه نهان به درخواستهای رسیده پاسخ دهند. برای بهبود واقعی کارایی در سیستمهای چندرسمانی، لازم است که ماشین سرویس دهنده وب بجای یک دیسک چندین دیسک (و

چندین کنترلر دیسک) داشته باشد. با داشتن k دیسک در سرویس دهنده ای با k ریسمان، کارایی سیستم k برابر سیستمهای یک دیسکی خواهد بود.



شکل ۷-۲۱. یک سرویس دهنده وب چندریسمانی، با یک ماژول جلودار و چند ماژول پردازش کننده.

از نظر تئوری، یک سیستم تک ریسمانی با k دیسک نیز می تواند به همان میزان از کارایی دست یابد، ولی چنین سیستمی بسیار پیچیده تر خواهد بود، چون ریسمانی که در حال خواندن دیسک است، هیچ کار دیگری نمی تواند انجام دهد (در حالیکه این محدودیت در سیستمهای چندریسمانی وجود ندارد).

سرویس دهنده های وب جدید فقط فایل عرضه نمی کنند، بلکه کارهای بیشتری (که می توانند بسیار پیچیده باشند) انجام می دهند. به همین دلیل، در اغلب سرویس دهنده ها هر ماژول پردازش کننده (پس از دریافت درخواست از ماژول جلودار، و بسته به نوع درخواست) مراحل مختلفی را طی می کند.

۱. تعیین نام صفحه وب خواسته شده.
۲. احراز هویت مشتری.
۳. کنترل دسترسی مشتری.
۴. کنترل دسترسی صفحه وب.
۵. چک کردن حافظه نهان.
۶. آوردن صفحه خواسته شده از دیسک.
۷. تعیین نوع MIME فایل و نوشتن آن در پاسخ مشتری.
۸. انجام جنبه های دیگر درخواست.
۹. برگرداندن پاسخ به مشتری.
۱۰. نوشتن یک رکورد در دفتر ثبت وقایع (log) سرویس دهنده.

شاید فکر کنید مرحله ۱ اضافیست، چون نام صفحه وب همیشه در درخواست مشتری قید می شود، اما موارد زیادی هست که نام صفحه در درخواست مشتری وجود ندارد. برای مثال، <http://www.cs.vu.nl> یک URL معتبر است که نام صفحه ندارد. در اینجا باید یک صفحه پیش فرض وجود داشته باشد که سرویس دهنده (در صورت

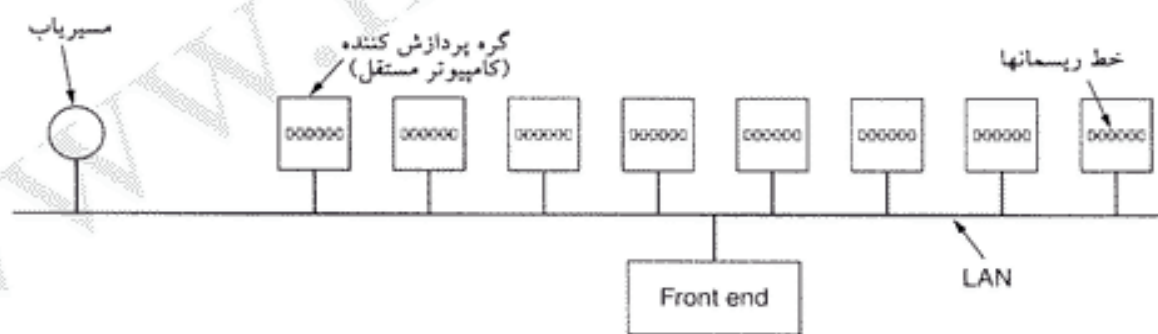
عدم وجود نام صفحه) از آن استفاده کند. در سرویس دهنده های وب جدید حتی می توان زبان پیش فرض (مانند انگلیسی، یا ایتالیایی) هم داشت، که صفحات وب (در صورت وجود) به این زبان برگردانده می شوند. بعلت وجود این مکانیزمهای پیش فرض، آوردن نام صفحات دیگر چندان الزامی نیست.

در مرحله ۲ هویت کاربر تأیید می شود. این مرحله بیشتر برای صفحاتی است که در معرض دسترسی عموم قرار ندارند، و فقط افراد خاصی می توانند از آنها استفاده کنند. در قسمتهای آینده یکی از روشهای احراز هویت مشتری را مورد بررسی قرار خواهیم داد.

در مرحله ۳ مجوزهای کاربر در دسترسی به صفحه خواسته شده چک می شود. مرحله بعد (مرحله ۴) این قبیل مجوزها را نسبت به خود صفحه بررسی می کند. مجوزهای دسترسی صفحات در فایل های خاصی (در همان دایرکتوری که صفحه در آن قرار دارد) تعیین می شود - فایل *htaccess* یکی از این نوع فایلهاست. صفحه خواسته شده در مراحل ۵ و ۶ گرفته می شود. روتین مرحله ۶ باید بگونه ای طراحی شود که قادر به کار همزمان با چندین دیسک باشد.

در مرحله ۷ نوع MIME فایل (از روی پسوند فایل، عبارتی در ابتدای فایل، یا از طریق فایل های پیکربندی) تعیین می شود. در مرحله ۸ کارهای متفرقه ای از قبیل ایجاد پروفایل کاربر یا جمع آوری داده های آماری، صورت می گیرد. صفحه در مرحله ۹ برای مشتری فرستاده شده، و در مرحله ۱۰ اقدامات انجام شده در یک فایل ثبت می شود (که این اطلاعات معمولاً بدرد سرپرست سایت می خورد).

اگر تعداد درخواستهای رسیده در واحد زمان بسیار زیاد باشد، CPU (صرفنظر از تعداد دیسکهای موازی) نمی تواند از عهده انجام آنها برآید. راه حل این مشکل استفاده از کامپیوترهای متعدد برای توزیع بار سرویس دهنده است. این مدل را که مزرعه سرویس دهنده (server farm) نام دارد، در شکل ۷-۲۲ ملاحظه می کنید. در این روش هم یک مازول جلودار وجود دارد که درخواستها را گرفته، ولی این بار (بجای پروسسهای مختلف) در گره ها یا کامپیوترهای مختلف - که هر کدام می توانند ماشینهای چندریسمانی با چندین دیسک باشند - توزیع می کند.

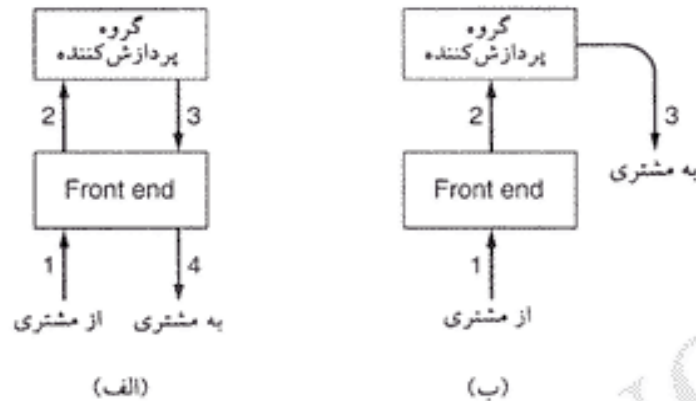


شکل ۷-۲۲. مزرعه سرویس دهنده.

یکی از مشکلات مزرعه سرویس دهنده این است که (بعلت جدا بودن حافظه کامپیوترها) حافظه نهان واحد نمی تواند وجود داشته باشد - مگر اینکه از کامپیوتری با چندین CPU و حافظه مشترک استفاده کنیم. برای غلبه بر این مشکل و بالا بردن کارایی سیستم، مازول جلودار می تواند مسیر درخواستهای مختلف را ثبت کرده و درخواستهای بعدی مرتبط با یک صفحه مشخص را به همان کامپیوتر قبلی بفرستد. این تکنیک باعث می شود تا هر گره (کامپیوتر) به ارائه صفحات خاصی اختصاص یابد، و از هرز رفتن حافظه نهان جلوگیری شود.

مشکل دیگر مزرعه سرویس دهنده این است که اتصالهای TCP به مازول جلودار می رسند، و پاسخ نیز باید از همان مسیر برگردد. به شکل ۷-۲۳ (الف) نگاه کنید: در اینجا درخواست ورودی (1) و پاسخ سرویس دهنده وب

(4) هر دو از مازول جلودار عبور کرده اند. برای دور زدن این مشکل، گاهی از تکنیکی بنام پاس دادن TCP (TCP handoff) استفاده می شود. در این روش اتصال TCP مشتری نیز به گره پردازش کننده منتقل می شود، تا این گره بتواند پاسخ را مستقیماً به مشتری برگرداند - مسیر (3) در شکل ۷-۲۳ (ب). کل فرآیند پاس دادن TCP از دید مشتری مخفی است.



شکل ۷-۲۳. (الف) فرآیند عادی درخواست-پاسخ. (ب) درخواست-پاسخ با استفاده از تکنیک پاس دادن TCP.

URL

بارها گفته ایم که یک صفحه وب می تواند لینکهایی به صفحات دیگر داشته باشد؛ اکنون خواهید دید که این کار چگونه صورت می گیرد. وقتی وب اختراع می شد، بلافاصله روشن شد که صفحات وب باید مکانیزمی برای نامگذاری و مکان یابی صفحاتی که به آنها لینک دارند، داشته باشند. این مکانیزم، بویژه، باید به سه سوال اساسی درباره صفحه انتخاب شده جواب دهد:

۱. نام این صفحه چیست؟
۲. کجا قرار دارد؟
۳. چگونه می توان آنرا خواند؟

اگر هر صفحه دارای یک نام منحصر بفرد باشد، پیدا کردن آن قاعدتاً نباید مشکلی داشته باشد. اما این راه حل هم مشکل ما را برطرف نخواهد کرد. با یک مثال موضوع روشنتر خواهد شد. در ایالات متحده آمریکا هر فرد یک شماره تأمین اجتماعی منحصر بفرد دارد، اما آیا فقط با داشتن این شماره می توان فرد مورد نظر را پیدا کرد. آدرس این فرد کجاست؟ با چه زبانی باید با او تماس گرفت؟ همین مسائل برای وب هم مصداق دارند.

راه حلی که برای این موضوع پیدا شد، به هر سه سوال فوق یکجا جواب می دهد. در این راه حل، هر صفحه وب یک URL (یابنده همسان منابع - Uniform Resource Locator) دارد، که آنرا بگونه ای منحصر بفرد در تمام دنیا مشخص می کند. هر URL سه بخش دارد: پروتکل (سوال ۳)، نام DNS ماشینی که صفحه روی آن قرار دارد (سوال ۲)، و نام صفحه در آن ماشین (سوال ۱). بعنوان نمونه در سایت وب مؤلف کتاب چندین فایل ویدئویی از شهر آمستردام و دانشگاه آن وجود دارد، که در صفحه ای با URL زیر قرار دارند:

<http://www.cs.vu.nl/video/index-en.html>

سه بخش این URL عبارتند از: پروتکل (*http*)، نام DNS ماشین سرویس دهنده (*www.cs.vu.nl*)، و نام صفحه وب (*video/index-en.html*) که با / از هم جدا شده اند. نام صفحه وب عبارتست از مسیر نسبی فایل در دایرکتوری وب پیش فرض کامپیوتر *cs.vu.nl*.

در اغلب سایتهای وب از نامهای کوتاه شده برای فایلها استفاده می شود. اگر نام فایل در URL وجود نداشته باشد، معمولاً صفحه اصلی سایت (home page) برگردانده می شود. اگر فقط دایرکتوری صفحه مشخص شده باشد، سرویس دهنده وب فایل پیش فرض آن دایرکتوری (که معمولاً `index.html` نام دارد) را به مشتری برمی گرداند. در برخی از سرویس دهنده های وب نیز از نامهای کوتاهی مانند `~user/` برای مشخص کردن دایرکتوری اختصاصی اشخاص استفاده می شود. بعنوان مثال، برای دسترسی به صفحه اصلی فضای وب مؤلف می توانید از URL زیر استفاده کنید:

`http://www.cs.vu.nl/~ast/`

اکنون اجازه دهید ببینیم آبرمتن (hypertext) چگونه کار می کند. برای اینکه یک عبارت قابل کلیک باشد، نویسنده متن باید دو چیز را مشخص کند: متنی که باید دیده شود، و URL صفحه ای که بعد از کلیک شدن عبارت باید باز شود (بعداً در همین فصل روش ایجاد چنین عبارت هایی را خواهید دید). وقتی این عبارت کلیک شود، مرورگر آدرس سرویس دهنده صفحه را با استفاده از DNS پیدا کرده، یک اتصال TCP به آن برقرار می کند و نام فایل و پروتکل را به سرویس دهنده می دهد. سرویس دهنده هم صفحه خواسته شده را برمی گرداند. یکی از ویژگی های URL این است که می توان براحتی پروتکل مورد استفاده را عوض کرده و به منابع مختلف دسترسی پیدا کرد. در حقیقت تعداد زیادی از این پروتکلها تعریف شده، که برخی از معروفترین آنها را در شکل ۷-۲۴ ملاحظه می کنید.

مثال	کاربرد	نام
<code>http://www.cs.vu.nl/~ast/</code>	آبرمتن (HTML)	http
<code>ftp://ftp.cs.vu.nl/pub/minix/README</code>	FTP	ftp
<code>file:///usr/suzanne/prog.c</code>	فایل محلی	file
<code>news:comp.os.minix</code>	گروه خبری	news
<code>news:AA0134223112@cs.utah.edu</code>	مقاله خبری	news
<code>gopher://gopher.tc.umn.edu/11/Libraries</code>	گوفر	gopher
<code>mailto:JohnUser@acm.org</code>	ارسال ایمیل	mailto
<code>telnet://www.w3.org:80</code>	ورود از راه دور	telnet

شکل ۷-۲۴. چند URL معروف.

اجازه دهید این فهرست را مختصراً بررسی کنیم. پروتکل `http` زبان وب است، زبانی که سرویس دهنده های وب با آن صحبت می کنند. HTTP مخفف HyperText Transfer Protocol (پروتکل انتقال آبرمتن) است. بعداً درباره این پروتکل بیشتر صحبت خواهیم کرد.

پروتکل `ftp` (پروتکل انتقال فایل - File Transfer Protocol) برای انتقال فایل در اینترنت مورد استفاده قرار می گیرد. FTP بیش از دو دهه است که در اینترنت حضور دارد، و پروتکلی کاملاً جا افتاده است. در سراسر دنیا تعداد زیادی سرویس دهنده FTP وجود دارد که کاربران اینترنت می توانند وارد آنها شده و فایل های موجود در آنها را بار کنند. وب تغییر چندانی در این وضعیت نداد، فقط کارها را ساده تر کرد (چون واسط کاربر FTP کمی کهنه و قدیمی شده بود). FTP بسیار قویتر از HTTP است، چون اجازه می دهد کاربر ماشین A فایل را از ماشین B به ماشین C منتقل کند.

مرورگرها می توانند با استفاده از پروتکل `file` مستقیماً با فایلها نیز (مانند صفحات وب) کار کنند، البته فقط با فایل های محلی (روی همان کامپیوتر) نه فایل های کامپیوترهای دیگر. روش کار شبیه FTP است، با این تفاوت که

نیازی به سرویس دهنده FTP ندارد.

مدتها قبل از اینکه اینترنت بدنیا بیاید، یک سیستم خبری وجود داشت بنام USENET. این سیستم میلیونها کاربر را در بیش از ۳۰,۰۰۰ گروه خبری (newsgroup) گرد هم آورده بود، کاربرانی که درباره موضوعات مختلف و متنوع بحث کرده و مقاله رد و بدل می کردند. برای ارتباط با این گروههای خبری می توان از پروتکل news استفاده کرد (یعنی، مرورگرها می توانند خبرخوان هم باشند، و در واقع بعضی از آنها حتی از برنامه های تخصصی خبرخوان نیز بهتر هستند).

پروتکل news از دو فرمت پشتیبانی می کند. در فرمت اول بعد از مشخص کردن گروه خبری، می توان از میان فهرست مقالات آن مقاله مورد نظر را انتخاب کرد. در فرمت دوم شماره شناسایی مقاله را باید مشخص کرد (AA0134223112@cs.utah.edu در شکل ۷-۲۴). مرورگرها برای آوردن مقالات گروه خبری از پروتکل NNTP (پروتکل انتقال اخبار شبکه - Network News Transfer Protocol) استفاده می کنند. در این کتاب درباره NNTP صحبت نخواهیم کرد، فقط کافایت یادآور شویم که این پروتکل تا حدی شبیه SMTP است و مانند آن عمل می کند.

از پروتکل gopher در سیستمهای گوفر استفاده می شود؛ این سیستم در دانشگاه مینه سوتا اختراع شد، و نام آن از تیم ورزشی این دانشگاه (گوفرهای طلایی - گوفر نوعی موش خرما بزرگ است، و در ضمن لقبی است که به اهالی ایالت مینه سوتا نیز داده شده) گرفته شده است. گوفر سالها قبل از وب وجود داشت، و نوعی سیستم بازیابی اطلاعات محسوب می شود (که متأسفانه فقط متنی است، و قابلیت های گرافیکی ندارد). این سیستم سالهاست که منسوخ شده، و دیگر بندرت کسی از آن استفاده می کند.

دو پروتکل آخر شکل ۷-۲۴ ارتباطی با صفحات وب ندارند، و کاربردهای دیگری دارند. پروتکل mailto اجازه می دهد که کاربر از داخل مرورگر ایمیل بفرستد. برای این کار کافایت روی عبارتی که URL آن `mailto:somebody@abcd.com` است، کلیک کنید. اغلب مرورگرها برای ارسال ایمیل به این آدرس از برنامه ایمیل پیش فرض سیستم استفاده می کنند. پروتکل telnet نیز برای ایجاد ارتباط با کامپیوترهای دیگر بکار می رود، و طرز کار آن بسیار شبیه برنامه telnet است.

استفاده از پروتکل های مختلف به کاربر اجازه می دهد تا یک برنامه واحد (مرورگر وب) را برای کارهای مختلف بکار گیرد. اگر نمی دانستیم که وب و مرورگر را یک فیزیکدان اختراع کرده، قطعاً آنها دستبخت بخش تبلیغات یک شرکت بزرگ نرم افزاری تصور می کردیم.

با وجود تمام این ویژگی های جالب URL، رشد سرسام آور وب ضعف ذاتی آن را آشکار کرد: هر URL به یک سرویس دهنده مشخص اشاره می کند. برای کاهش ترافیک صفحاتی که تعداد مراجعان آن زیاد است، بهتر است یک URL چندین سرویس دهنده داشته باشد. اما مشکل اینجاست که، URL ها هیچ راهی ندارند که بدون ذکر آدرس دقیق صفحه وب، به آن اشاره کنند. برای مثال، یک URL نمی تواند بگوید: «من صفحه xyz را می خواهم، اهمیتی هم نمی دهم آنها از کجا می آوری.» برای حل این مشکل و امکان تکثیر صفحات وب، IETF در حال کار روی سیستمی از URN (نام جهانی منابع - Universal Resource Name) هاست. در کل، URN را می توان یک URL عمومی دانست. تحقیقات در این زمینه همچنان ادامه دارد، اما پیشنهادهایی تحت RFC 2141 نیز ارائه شده است.

بدون حالتی و کوکی

همانطور که بارها خاطر نشان کردیم، وی اساساً بدون حالت (stateless) است، و چیزی مانند ورود به سیستم (login) در آن وجود ندارد. مرورگر درخواست خود را به سرویس دهنده می فرستد، سرویس دهنده هم یک فایل

به آن برمی گرداند، و بعد کلی فراموش می کند که اصلاً چنین مشتری را دیده است. در اوایل کار، زمانیکه وب فقط یک محیط عمومی بود، این مدل کاملاً کفایت می کرد. ولی با گسترش کاربردهای وب این موضوع مشکل ساز شد. بعنوان مثال، برای استفاده از برخی سایتهای وب کاربران باید ثبت نام کنند، و احیاناً پولی بپردازند. این وضعیت سئوالی را پیش می آورد: «سرویس دهنده وب چگونه باید بین درخواستهای کاربرانی که ثبت نام کرده اند، و کاربران عادی فرق قائل شود؟» مثال دیگر مربوط به تجارت الکترونیک (e-commerce) می شود: «کاربری در یک فروشگاه الکترونیکی می چرخد و اجناس مورد نیازش را در یک سبد خرید الکترونیکی (shopping cart) قرار می دهد؛ سئوال اینست که سرویس دهنده وب چگونه محتویات هر سبد خرید را ردگیری می کند؟» مثال سوم به سایتهایی که اجازه می دهند کاربر فضاهای اختصاصی داشته باشد (مانند Yahoo)، مربوط می شود: «سرویس دهنده وب چگونه کاربران را شناسایی می کند، و بخاطر می سپارد که هر کاربر صفحه اش را چگونه تنظیم کرده است؟»

در نگاه اول شاید فکر کنید که سرویس دهنده وب می تواند از آدرس IP کاربر برای شناسایی وی استفاده کند. اما این روش کار نمی کند. اول اینکه، بسیاری کاربران از کامپیوترهای مشترک استفاده می کنند (به ویژه کارمندان شرکتها)، و آدرس IP فقط بدرد شناسایی کامپیوترها می خورد، نه کاربران. دیگر اینکه، اغلب ISPها از تکنیک NAT استفاده می کنند، و تمام بسته هایی که از این ISPها خارج می شود یک آدرس IP دارند. از دیدگاه سرویس دهنده وب تمام کاربران این ISP یک آدرس IP بیشتر ندارند.

برای حل این مشکل، نت اسکپ تکنیکی اختراع کرد بنام کوکی (cookie)، که انتقادات زیادی به آن وارد شد. این نام از نوعی برنامه خاص گرفته شده، که کاری را انجام می دهد و آنرا در جایی ثبت می کند، تا بعدها بتواند دوباره بیاد بیاورد چه کار کرده است (در سیستم عامل هم چنین مفهومی بکرات مورد استفاده قرار می گیرد). کوکی بعدها در RFC 2109 جنبه رسمی بخود گرفت.

وقتی مشتری یک صفحه وب درخواست می کند، سرویس دهنده مقداری اطلاعات اضافی همراه آن می فرستد که کوکی هم می تواند جزئی از آن باشد. کوکی معمولاً یک فایل کوچک (حداکثر 4 KB) یا یک رشته متنی است. مرورگر این کوکی ها را در یک دایرکتوری خاص روی کامپیوتر مشتری ذخیره می کند (البته اگر کاربر این ویژگی را غیرفعال نکرده باشد). کوکی ها فقط فایل یا رشته های متنی هستند، نه برنامه های اجرایی. در حقیقت، کوکی حتی می تواند حاوی کد یک ویروس باشد، اما از آنجائیکه کوکی ها اساساً داده تلقی می شوند، چنین ویروسی هیچ راهی برای اجرا شدن در کامپیوتر مشتری (و صدمه زدن به آن) ندارد. با این حال، همیشه احتمال آن هست که بالاخره یک هکر بتواند راهی برای این کار پیدا کند.

هر کوکی می تواند تا پنج فیلد داشته باشد (شکل ۷-۲۵ را ببینید). فیلد Domain مشخص می کند که کوکی از کجا آمده است. مرورگرها مکانیزمهایی دارند تا مطمئن شوند که سرویس دهنده دروغ نگفته است. هر ناحیه می تواند حداکثر ۲۰ کوکی (برای هر مشتری) در یک کامپیوتر ذخیره کند. فیلد Path مشخص می کند که کدام بخش از سیستم فایل سرویس دهنده وب می تواند از کوکی استفاده کند. این فیلد اغلب / است، که به معنای کل سیستم فایل می باشد.

ایمپی	زمان انقضا	محتویات	مسیر	ناحیه
بلی	15-10-02 17:00	CustomerID=497793521	/	toms-casino.com
غیر	11-10-02 14:22	Cart=1-00501;1-07031;2-13721	/	joes-store.com
غیر	31-12-10 23:59	Prefs=Stk:SUNW+ORCL;Spt:Jets	/	aportal.com
غیر	31-12-12 23:59	UserID=3627239101	/	sneaky.com

شکل ۷-۲۵. چند نمونه کوکی.

فیلد *Content* به شکل $name = value$ است، که *name* و *value* می توانند هر چیزی (که سرویس دهنده می خواهد) باشند. این فیلد نیست که محتویات کوکی در آن ذخیره می شود.

فیلد *Expires* مشخص می کند که کوکی تا چه زمانی اعتبار دارد. اگر در این فیلد مقدار وجود نداشته باشد، مرورگر هنگام خروج کوکی را دور می اندازد. به این قبیل کوکی ها کوکی بی دوام (*nonpersistent cookie*) می گویند. اگر فیلد *Expires* کوکی تاریخ و ساعت داشته باشد، به آن بادوام (*persistent*) می گویند، و کوکی تا این زمان در سیستم حفظ خواهد شد. زمان انقضا بوقت گرینویچ است. اگر سرویس دهنده بخواهد یک کوکی را از روی سیستم مشتری پاک کند، کافیه همان کوکی را دوباره با تاریخ انقضایی در گذشته بفرستد.

و بالاخره فیلد *Secure* می گوید که مرورگر باید کوکی را فقط به یک سرویس دهنده امن (*secure*) پس بفرستد. این ویژگی بیشتر برای تجارت الکترونیک، عملیات بانکی و سایر کاربردهایی که به ایمنی بالا نیاز دارند، مورد استفاده قرار می گیرد.

دیدید که سرویس دهنده چطور کوکی ها را به مشتری می فرستد، اما طرز کار آنها چگونه است؟ درست قبل از اینکه مرورگر درخواستی را به یک سرویس دهنده وب بفرستد، دایرکتوری کوکی ها را چک می کند تا ببیند آیا ناحیه مقصد در آنجا کوکی دارد یا خیر. اگر چنین باشد، مرورگر تمام آن کوکی ها را همراه با درخواست خود به سرویس دهنده وب می فرستد. وقتی سرویس دهنده این کوکی ها را دریافت کرد، خود می داند با آنها چه کند.

اجازه دهید چند نمونه از کاربرد کوکی ها را بررسی کنیم. در شکل ۷-۲۵، اولین کوکی متعلق به ناحیه *toms-casino.com* است، و به کار شناسایی کاربر می آید. وقتی کاربر به خیال بردن مقداری پول وارد این سایت می شود، سرویس دهنده با استفاده از این کوکی وی را شناسایی می کند. برای مثال، سرویس دهنده وب می تواند به کمک این عدد مشخصات کاربر را از پایگاه داده خود استخراج کرده، و صفحه را به شکل مناسب (بسته به علائق قبلی کاربر) به نمایش در آورد.

کوکی دوم متعلق به سایت *joes-store.com* است. در این سناریو کاربر ما مشغول چرخیدن در یک فروشگاه مجازی و انتخاب کالاست. وقتی کاربر کالای مناسبی را یافت، روی آن کلیک می کند؛ با این کار، سرویس دهنده وب یک کوکی به کامپیوتر مشتری می فرستد، که در آن تعداد اجناس انتخاب شده و شماره آنها قید شده است. هر بار که کاربر صفحه جدیدی را در این فروشگاه باز می کند (و در واقع درخواست جدیدی می فرستد)، این کوکی را هم به همراه آن به سرویس دهنده وب برمی گرداند. در مثال شکل ۷-۲۵ سه قلم جنس در سبد خرید مشتری وجود دارد، که از سومی دو عدد انتخاب شده است. در پایان هم وقتی مشتری پای صندوق می رود، سرویس دهنده با استفاده از همین کوکی می تواند قیمت کل خریدهای مشتری را به وی اعلام کند.

کوکی سوم متعلق به یکی از درگاه های وب (*web portal*) است. وقتی کاربر می خواهد وارد این سایت شود، مرورگر این کوکی را به همراه درخواست وی به سرویس دهنده وب می فرستد. سرویس دهنده وب هم صفحه ای به کاربر برمی گرداند که در آن نرخ سهام شرکت های *Sun Microsystems* و *Oracle*، و نتایج تیم فوتبال *New Yprk Jets* نشان داده شده است. از آنجائیکه یک کوکی می تواند تا 4 KB باشد، کاربر می تواند فهرست بلندبالایی از علائق خود را در این سایت ثبت کند.

یک سرویس دهنده وب می تواند از کوکی فقط برای مصارف داخلی خودش استفاده کند، مثلاً تعداد کاربرانش را بداند، و یا بداند که هر بازدیدکننده قبل از ترک سایت چند صفحه را دیده است. وقتی اولین بازدیدکننده وارد سایت می شود، هنوز هیچ کوکی از آن در کامپیوترش ندارد، پس سرویس دهنده یک کوکی که در آن عبارت $Counter = 1$ نوشته شده به مشتری می فرستد. کلیک بعدی کاربر باعث می شود تا همین کوکی به سرویس دهنده برگردد. سرویس دهنده هم مقدار *Counter* را یکی افزایش داده، و دوباره به مشتری برمی گرداند. بدین ترتیب

سرویس دهنده می تواند آماری از تعداد بازدیدکنندگان سایت (و اینکه هر کدام چند صفحه را دیده اند) بدست آورد. امکان سوء استفاده از کوکی ها نیز وجود دارد. در تئوری، مرورگر کوکی ها را فقط به سایتی که به آن تعلق دارند می فرستد، ولی برخی از هکرها توانسته اند با استفاده از باگهایی که در مرورگرها وجود دارد، کوکی هایی را که متعلق به آنها نیست بدست آورند. از آنجائیکه برخی از سایتهای تجارت الکترونیک شماره کارت اعتباری مشتریان خود را در کوکی ها ذخیره می کنند، می توانید خطرات بالقوه این وضعیت را بروشنی دریابید.

نقطه ضعف دیگر کوکی ها (که بحثهای زیادی را هم بدنبال داشته) امکان استفاده از آنها برای جمع آوری مخفیانه اطلاعات درباره کاربران و عاداتی و بگردی آنهاست. برای مثال، شرکت تبلیغاتی Sneaky Ads سایتهای معروف تماس گرفته و از آنها می خواهد که (در ازای دریافت اجرت) اعلان تبلیغاتی این شرکت را بالای سایت خود نصب کنند (و همانطور که می دانید این بزرگترین منبع درآمد سایتهای وب است). اما Sneaky Ads بجای دادن تصویر GIF یا JPEG آگهی تبلیغاتی خود، یک URL به این شرکتها می دهد تا در صفحات خود قرار دهند. هر URL که این شرکت به طرفهای خود می دهد، یک شماره منحصر بفرد دارد، مانند زیر

<http://www.sneaky.com/382674902342.gif>

وقتی کاربر وارد صفحه یکی از این سایتها (مثلاً، صفحه P) می شود، مرورگر فایل HTML آنرا می خواند، و بعد از آنکه متوجه شد در آن یک لینک به تصویری در سایت www.sneaky.com وجود دارد، درخواستی برای خواندن این تصویر به سایت مزبور می فرستد. سرویس دهنده این سایت به همراه تصویر آگهی، یک کوکی با شماره شناسایی منحصر بفرد 3627239101 به مرورگر پس می فرستد (شکل ۷-۲۵ را ببینید). سپس، سایت Sneaky بازدید کاربر از صفحه P را ثبت می کند (و این کاری ساده است، چون سرویس دهنده وب می داند که فایل [382674902342.gif](http://www.sneaky.com/382674902342.gif) مربوط به صفحه P است). البته احتمالاً آگهی همان است، فقط نام فایل آن در هر سایت فرق می کند.

بعدها وقتی کاربر وارد سایت دیگری که آگهی شرکت Sneaky Ads در آن قرار دارد، می شود مرورگر فایل HTML آنرا خوانده و سپس از سایت www.sneaky.com درخواست ارسال تصویر (مثلاً [493654919923.gif](http://www.sneaky.com/493654919923.gif)) را می کند. اما از آنجائیکه مرورگر یک کوکی از سایت www.sneaky.com دارد، این کوکی را نیز همراه درخواست خود می فرستد؛ و اکنون Sneaky Ads دومین صفحه ای را که کاربر ما از آن بازدید کرده، می شناسد.

به مرور زمان، شرکت Sneaky Ads اطلاعات کاملی درباره تعداد زیادی از کاربران و عاداتی و بگردی آنها جمع آوری می کند (بدون اینکه آنها حتی یکبار روی آگهی های این شرکت کلیک کرده باشند!). البته این شرکت هنوز نام کاربر را نمی داند (اگرچه آدرس IP او را می داند، و همین برای بدست آوردن اطلاعات بعدی کافیست) - و اگر کاربر بیچاره حتی برای یک بار نامش را در اختیار یکی از شرکتهای همکار Sneaky Ads گذاشته باشد، که دیگر کار تمام است. امروزه فروش اطلاعات کاربران اینترنت و علائق آنها (به طالبان این قبیل اطلاعات) یکی از منابع سرشار درآمد در وب محسوب می شود. بدترین جنبه این نوع جمع آوری اطلاعات آنست که کاربر حتی روحش از آن خبر ندارد، و فکر می کند چون روی هیچ آگهی اینترنتی کلیک نکرده کاملاً در امان است!

و اگر Sneaky Ads بخواهد پلیدی را به نهایت برساند، حتی لازم نیست یک آگهی کلاسیک بدهد؛ یک آگهی که فقط یک پیکسل (آن هم به رنگ زمینه سایت!) باشد، برای کار او کافیست (مرورگر باز هم مجبور است برای این تصویر یک پیکسلی به سایت www.sneaky.com برود، و کوکی را بگیرد).

برخی از کاربران (برای حفظ حریم شخصی خود) مرورگر را طوری پیکربندی می کنند که تمام کوکی ها را رد کنند. اما این کار در عملکرد سایتهای معتبری که به کوکی وابسته اند، نیز اختلال ایجاد می کند. برای حل این مشکل می توان از نرم افزارهایی که به کوکی خور (cookie-eating) معروفند، استفاده کرد. این نرم افزارها تمام کوکی هایی

را که به یک کامپیوتر می شوند، گرفته و فقط آنهایی را ذخیره می کنند که از نظر کاربر مجاز شناخته شده باشند. مرورگرهای جدید هم به کاربران امکان کنترل کامل کوکی ها را می دهند.

۲-۳-۷ سندهای وب استاتیک

اساس وب عبارتست از انتقال صفحات وب از سرویس دهنده به مشتری. صفحات وب، در ساده ترین شکل خود، استاتیک (ثابت و ایستا) هستند و فقط روی سرویس دهنده منتظرند تا کسی بیاید و آنها را بردارد. با این تعریف، حتی یک فیلم ویدئویی نیز استاتیک است، چون چیزی نیست جز یک فایل ساده. در این قسمت صفحات وب استاتیک را بتفصیل مورد بررسی قرار خواهیم داد.

HTML

صفحات وب با زبانی بنام HTML (زبان علامتگذاری اَیترمتن - HyperText Markup Language) نوشته می شوند. بنا بر HTML می توان متن، گرافیک و لینک به صفحات وب اضافه کرد. HTML یک زبان علامتگذاریست، یعنی زبانی که نشان می دهد سند چگونه باید فرمت شود. «علامتگذاری» اصطلاحیست قدیمی که به دوران چاپ با حروف سربی برمی گردد، و در آن ویراستار با علامتگذاری متن نوشته به حروفچین نشان می داد که چگونه (با چه حروف و اندازه ای) باید صفحه چاپی را بچیند. در زبانهای علامتگذاری فرمانهای مشخصی برای فرمت کردن سند وجود دارد. برای مثال، در HTML علامت `` فرمان شروع فرمت بافونت ضخیم، و `` فرمان پایان فونت ضخیم است. مزیت زبان علامتگذاری اینست که نوشتن مرورگر برای آن ساده است، چون فقط کافیست مرورگر فرمانهای علامتگذاری را بداند. زبانهای TeX و troff هم جزء زبانهای علامتگذاری معروف هستند.

با نوشتن فرمانهای علامتگذاری در فایل HTML و استاندارد کردن این فرمانها، تمام مرورگرها می توانند صفحات وب را خوانده و فرمت کنند. این ویژگی اهمیت بسیار زیادی در نمایش صحیح صفحات وب دارد، چون ممکنست یک صفحه وب در کامپیوتری با وضوح 1600×1200 پیکسل و رنگ 24-bit ایجاد شده باشد، ولی کامپیوتر بازدیدکننده فقط وضوحی معادل 640×480 پیکسل و رنگ 8-bit داشته باشد.

در این قسمت HTML را بطور مختصر بررسی خواهیم کرد. یک سند HTML را می توان با هر ادیتوری نوشت، اما برای نوشتن صفحات HTML برنامه های خاصی نیز طراحی شده است، که امکانات بیشتری در اختیار فرد قرار می دهند (و در ضمن دست او را در ریزه کاریها می بندند).

هر صفحه وب یک سر (head) و یک بدنه (body) دارد، که بین برچسبهای `<html>` و `</html>` قرار می گیرند، اگرچه اغلب مرورگرها نبودن این دو برچسب را نادیده می گیرند (برچسب - tag - فرمان فرمت HTML است که داخل `< >` نوشته می شود). همانطور که در شکل ۷-۲۶ (الف) می بینید، قسمت سر بین برچسبهای `<head>` و `</head>`، و قسمت بدنه بین برچسبهای `<body>` و `</body>` محصور می شوند. دستورات HTML داخل این برچسبها نوشته می شود. اغلب فرمانهای HTML دارای چنین شکلی هستند، یعنی با `<something>` شروع، و به `</something>` ختم می شوند. در اغلب مرورگرها فرمانی وجود دارد بنام VIEW SOURCE (یا چیزی شبیه آن)، که به کمک آن می توان (بجای خروجی فرمت شده) محتویات فایل HTML را دید.

نوع حروف در برچسبهای HTML تفاوتی ندارد، بعبارت دیگر برچسبهای `<head>` و `<HEAD>` یکی هستند؛ البته در استانداردهای جدید فقط می توان از حروف کوچک برای نوشتن برچسبها استفاده کرد. فرمت خود فایل HTML هیچ تأثیری روی خروجی آن ندارد، چون مرورگرها مجبورند صفحات را در کامپیوترهای مختلف

نمایش دهند، و بهمین دلیل فاصله‌ها و فضاها را حذف کرده و صفحه را متناسب با تنظیمات کامپیوتر مقصد از نو فرمت می‌کنند. البته نویسنده فایل HTML می‌تواند از فاصله و فضای خالی برای خوانا تر کردن آن استفاده کند (که اتفاقاً چیز بسیار خوبی هم هست). در نتیجه، برای فاصله انداختن بین پاراگرافها نمی‌توانید از Enter استفاده کنید: برای این کار برچسب مخصوصی وجود دارد. بعضی از برچسبها دارای پارامترهای نام‌دار (named parameter) هستند، که به صفت (attribute) معروفند.

```
<html>
<head> <title> AMALGAMATED WIDGET, INC. </title> </head>
<body> <h1> Welcome to AWI's Home Page </h1>
 <br>
We are so happy that you have chosen to visit <b> Amalgamated Widget's</b>
home page. We hope <i> you </i> will find all the information you need here.
<p>Below we have links to information about our many fine products.
You can order electronically (by WWW), by telephone, or by fax. </p>
<hr>
<h2> Product information </h2>
<ul>
<li> <a href="http://widget.com/products/big"> Big widgets </a>
<li> <a href="http://widget.com/products/little"> Little widgets </a>
</ul>
<h2> Telephone numbers </h2>
<ul>
<li> By telephone: 1-800-WIDGETS
<li> By fax: 1-415-765-4321
</ul>
</body>
</html>
```

(الف)



(ب)

شکل ۷-۲۶. (الف) کد HTML برای صفحه وب نمونه (ب) صفحه قالب‌دهی شده بر روی مرورگر.

مثلاً، در فرمان

``

برچسب `` دو پارامتر بنامهای `src` و `alt` دارد، که مقدار آنها بترتیب `abc` و `foobar` است. در استاندارد HTML فهرستی از پارامترهای مجاز هر برچسب و مفهوم آنها داده شده است. از آنجائیکه این پارامترها نامدار هستند، ترتیب نوشتن آنها اهمیتی ندارد.

از نظر فنی، سند های HTML با مجموعه کاراکتر ISO 8859-1 Latin-1 نوشته می شوند، ولی از آنجائیکه اغلب کاربران از صفحه کلید های ASCII استفاده می کنند، برای نمایش کاراکترهای خاص (مانند `è`) می توان از توالی گریز (escape sequence) کمک گرفت. این کاراکترها (که فهرست آنها در استاندارد HTML آمده) با `&` شروع و به `;` ختم می شوند. برای مثال، توالی گریز ` ` معادل یک فاصله (space)، توالی گریز ``` معادل `è`، و توالی گریز `´` معادل `é` است. از آنجائیکه `<`، `>` و `&` در فایل های HTML معنای خاصی دارند، برای نمایش آنها نیز باید از توالی های گریز (بترتیب `<`، `>` و `&`) استفاده کرد.

مهمترین چیزی که در قسمت سر نوشته می شود عنوان صفحه وب است، که بین برچسب های `<title>` و `</title>` قرار می گیرد (البته برخی اطلاعات دیگر، که به اطلاعات متا معروفند، نیز در قسمت سر نوشته می شوند). هر چیزی که در برچسب `<title>` نوشته شود، در خود صفحه وب دیده نخواهد شد، بلکه در میله عنوان پنجره مرورگر ظاهر می شود.

اجازه دهید نگاهی به قسمتهای دیگر شکل ۷-۲۶ بیندازیم. برچسب هایی که در این فایل بکار رفته (و چند برچسب دیگر) را در شکل ۷-۲۷ ملاحظه می کنید. برای نوشتن تیتیر مطالب می توانید از برچسب `<h1>`، که n عددی بین ۱ تا ۶ است، استفاده کنید: `<h1>` بزرگترین تیتیر، و `<h6>` کوچکترین تیتیر را تولید می کنند. فرمت کردن تیتیرها (اندازه و نوع فونت یا رنگ آنها) بر عهده مرورگر است، و معمولاً تیتیرهای درشت تر اعداد کوچکتری دارند. برچسب `<h1>` که بزرگترین تیتیر را تولید می کند، دارای بیشترین فاصله خالی در بالا و پائین نیز هست، در حالیکه تیتیرهای کوچکتر فاصله کمتری با خطوط دیگر دارند.

برچسب	مفهوم
<code><html> ... </html></code>	صفحه وب را تعریف می کند
<code><head> ... </head></code>	محتویات سر صفحه را مشخص می کند
<code><title> ... </title></code>	عنوان صفحه را تعیین می کند (که البته در خود صفحه دیده نمی شود)
<code><body> ... </body></code>	بدنه صفحه را مشخص می کند
<code><h n> ... </h n></code>	سطح (اندازه) عنوان را تیتیر را مشخص می کند
<code> ... </code>	محتویات برچسب را ضخیم می کند
<code><i> ... </i></code>	محتویات برچسب را کج می کند
<code><center> ... </center></code>	محتویات برچسب را وسط صفحه وب قرار می دهد
<code> ... </code>	یک لیست غیر منظم (گلوله دار) می سازد
<code> ... </code>	یک لیست شماره دار می سازد
<code> ... </code>	آینمهای لیست را مشخص می کند
<code>
</code>	یک شکست خط در صفحه ایجاد می کند
<code><p></code>	یک پاراگراف جدید می سازد
<code><hr></code>	یک خط افقی روی صفحه رسم می کند
<code></code>	یک تصویر روی صفحه نمایش می دهد
<code> ... </code>	یک لینک را تعریف می کند

شکل ۷-۲۷. چند برچسب HTML، برخی از این برچسبها می توانند پارامتر داشته باشند.

برچسبهای `` و `<i>` به ترتیب برای ضخیم و کج کردن فونت بکار می‌روند. اگر مرورگر نتواند فونتهای ضخیم یا کج را نمایش دهد، معمولاً آنها بگونه‌ای دیگر (رنگ متفاوت یا نگاتیو کردن حروف) از سایر قسمت‌ها متمایز خواهد کرد.

در HTML مکانیزم‌های مختلفی برای ایجاد لیست (از جمله لیستهای تودرتو) وجود دارد. لیست‌ها با برچسب `` یا `` شروع می‌شوند، که در هر دو مورد برای مشخص کردن آیتمهای لیست از برچسب `` استفاده می‌کنیم. برچسب `` یک لیست مرتب‌نشده (unordered list) می‌سازد، که آیتمهای آن با گلوله (bullet) مشخص می‌شوند. برچسب `` یک لیست مرتب‌شده (ordered list) می‌سازد، که آیتمهای آن توسط مرورگر شماره‌گذاری خواهند شد.

برچسبهای `
`، `<p>` و `<hr>` بین قسمت‌های مختلف متن فاصله می‌اندازند. برای فرمت کردن دقیق متن می‌توان از شیوه‌نامه (style sheet) نیز استفاده کرد. برچسب `
` باعث می‌شود تا ادامه متن از سر خط شروع شود (معمولاً مرورگرها بعد از `
` خط خالی اضافه نمی‌کنند). برچسب `<p>` پاراگراف جدیدی را شروع می‌کند و بین آنها یک خط فاصله می‌اندازد (و حتی ممکنست تورفتگی - indent - ابتدای پاراگراف را هم رعایت کند). از نظر تئوری یک پاراگراف باید با `<p>` شروع و به `</p>` ختم شود، ولی برچسب `<p>` بندرت بکار برده می‌شود، و حتی اغلب آنهایی که صفحات وب می‌نویسند از وجود آن خبر ندارد. برچسب `<hr>` نیز یک خط افقی روی صفحه رسم می‌کند.

صفحات وب می‌توانند تصویر نیز داشته باشند. یک برچسب `` تصویر مشخص شده را در همان نقطه‌ای که این برچسب نوشته شده، نمایش می‌دهد. برچسب `` می‌تواند پارامترهای متعددی بگیرد. پارامتر `src` همان URL تصویر موردنظر است. فرمت این تصاویر جزئی از استاندارد HTML نیست، بلکه جزء قابلیت‌های مرورگر محسوب می‌شود. اغلب مرورگرها می‌توانند تصاویر GIF و JPEG را نمایش دهند. مرورگرها می‌توانند از فرمت‌های دیگر نیز پشتیبانی کنند، ولی این یک شمشیر دو لبه است؛ اگر عادت کنید از فرمت‌های دیگر (مثلاً، BMP) در صفحات وب خود استفاده کنید، شاید بعضی از کاربران (که مرورگرهای متفاوتی دارند) نتوانند خلاقیت هنری شما را تحسین کنند!

برخی دیگر از پارامترهای `` عبارتند از: `align` که تصویر را با خط کرسی متن تراز می‌کند (مقادیر `top`، `middle`، و `bottom` به ترتیب معادل تراز بالا، وسط و پایین هستند)؛ `alt` که متن جایگزین تصویر را (وقتی کاربر تصاویر را غیرفعال کرده باشد) مشخص می‌کند؛ و `ismap` که نشان می‌دهد تصویر دارای نقاط فعال (قابل کلیک) است. برای ایجاد آبرلینک در صفحات وب از برچسب `<a>...` استفاده می‌کنیم. برچسب `<a>` هم پارامترهای مختلفی دارد، از جمله `href` (URL مقصد لینک)، و `name` (نام آبرلینک). متن (یا تصویری) که بین `<a>` و `` قرار دارد، در صفحه وب دیده خواهد شد، و کاربر با کلیک کردن این متن (یا تصویر) می‌تواند به صفحه مقصد لینک برود. در زیر نمونه‌ای از یک آبرلینک را می‌بینید.

```
<a href="http://www.nasa.gov">NASA's home page</a>
```

که کاربر آنرا در مرورگر به این صورت خواهد دید:

NASA's home page

و اگر کاربر کنجکاو ما به ناسا علاقمند باشد و روی این لینک کلیک کند، مرورگر بلافاصله صفحه `http://www.nasa.gov` را آورده و نمایش می‌دهد. مثال زیر شکل دیگری از همان لینک بالاست:

```
<a href="http://www.nasa.gov"></a>
```

در اینجا بجای عبارت NASA's home page از تصویر یک شاتل فضایی استفاده کرده‌ایم، و کاربر با کلیک

کردن این تصویر به همان صفحه <http://www.nasa.gov> خواهد رفت. اگر هم کاربر نمایش تصاویر را در مرورگر غیرفعال کرده باشد، کلمه NASA نشان می دهد که موضوع از چه قرار است.

برچسب `<a>` پارامتری دارد بنام `name`، که به کمک آن می توان لینک را به وسط یک صفحه نشانه رفت. با این پارامتر می توان صفحاتی بصورت فهرست مطالب ایجاد کرد، که کلیک کردن آیتمهای آن باعث رفتن کاربر به قسمت مربوطه در همان صفحه می شود.

استاندارد HTML از تغییر و تکامل در امان نبوده است. در ویرایشهای HTML 1.0 و HTML 2.0 جدول (table) وجود نداشت، ولی از HTML 3.0 این عنصر هم به صفحات وب اضافه شد. هر جدول HTML تعدادی سطر دارد، که خود از یک یا چند سلول (cell) تشکیل شده است. در یک سلول می توان هر چیزی قرار دارد: متن، عدد، تصویر، و حتی یک جدول دیگر. سلولها را می توان در هم ادغام کرد، برای مثال می توان با ادغام چند سلول برای جدول تیترا عنوان درست کرد. جدولها حرف آخر را در فرمت کردن صفحات وب (و کنترل خصوصیات ظاهری آن) می زنند.

در شکل ۷-۲۸ (الف) تعریف یک جدول HTML، و در شکل ۷-۲۸ (ب) خروجی آنرا می بینید. این مثال فقط برخی از ویژگیهای ابتدایی جدول را نشان می دهد. جدولها با برچسب `<table>` شروع می شوند، و با دادن پارامترهای دلخواه می توان ویژگیهای کلی آنها را تعیین کرد.

با استفاده از برچسب `<caption>` می توان یک تیترا کلی به جدول داد. هر سطر جدول با برچسب `<tr>` (مخفف Table Row) شروع می شود، و برای مشخص کردن محتویات سلولها از برچسبهای `<th>` (مخفف Table Header) یا `<td>` (مخفف Table Data) استفاده می شود (مرورگر برچسبهای `<th>` و `<td>` را بگونه ای متفاوت فرمت می کند). جدولها دارای صفات متعددی دیگری (از جمله نوع تراز افقی و عمودی سلولها، حاشیه آنها، و ادغام سلولها در یکدیگر) نیز هستند.

در HTML 4.0 قابلیتهای جدیدی به این زبان اضافه شده است، که از میان آنها می توان به امکانات جدید برای سهولت و بگردی معلولین، قابلیت استفاده از شیء (object) در صفحات وب، پشتیبانی از زبانهای اسکریپت نویسی (برای ایجاد محتویات دینامیک) اشاره کرد.

در سایتهای بزرگ وب که طراحان متعددی در ایجاد صفحات آن شرکت دارند، یکی از مهمترین نکات حفظ یکدستی و یکنواختی ظاهر صفحات است. این مشکل را می توان با استفاده از شیوه نامه (style sheet) حل کرد. در این تکنیک طراحان صفحات وب بجای شیوه های فیزیکی (مانند فونت ضخیم یا کج)، از شیوه های منطقی مانند `<dn>` (تعریف)، `` (تأکید خفیف)، `` (تأکید شدید)، و `<var>` (متغیرهای برنامه) استفاده می کنند. این شیوه های منطقی در یک شیوه نامه (که در ابتدای هر صفحه به آن ارجاع می شود) تعریف می شوند. با استفاده از شیوه نامه، سرپرست تیم طراحی می تواند ظاهر یکنواخت کلیه صفحات را تضمین کند. برای مثال، اگر روزی تصمیم گرفته شود که برچسب `` از فونت 14 کج آبی به فونت 18 ضخیم صورتی پُررنگ تبدیل شود، فقط کافیست تعریف این برچسب در شیوه نامه تغییر کند. شیوه نامه ها را می توان معادل فایل های `#include` در برنامه های C (که محل تعریف ماکروهاست) دانست.

فرم

اولین ویرایش HTML (یعنی HTML 1.0) اساساً یک طرفه بود: کاربران می توانستند صفحات را درخواست کنند، ولی فرستادن اطلاعات به سرویس دهنده وب بسیار مشکل بود. با رشد کاربردهای تجاری وب، تقاضا برای ترافیک دو طرفه بشدت فزونی گرفت. برای مثال، شرکتهای بسیاری میل داشتند از طریق وب هم سفارش بگیرند، یا شرکتهای نرم افزاری می خواستند محصولات خود را بصورت الکترونیکی بفروشند، و دهها مورد مانند آن.

```

<html>
<head> <title> A sample page with a table </title> </head>
<body>
<table border=1 rules=all>
<caption> Some Differences between HTML Versions </caption>
<col align=left>
<col align=center>
<col align=center>
<col align=center>
<tr> <th>Item <th>HTML 1.0 <th>HTML 2.0 <th>HTML 3.0 <th>HTML 4.0 </tr>
<tr> <th> Hyperlinks <td> x <td> x <td> x <td> x </tr>
<tr> <th> Images <td> x <td> x <td> x <td> x </tr>
<tr> <th> Lists <td> x <td> x <td> x <td> x </tr>
<tr> <th> Active Maps and Images <td> &nbsp; <td> x <td> x <td> x </tr>
<tr> <th> Forms <td> &nbsp; <td> x <td> x <td> x </tr>
<tr> <th> Equations <td> &nbsp; <td> &nbsp; <td> x <td> x </tr>
<tr> <th> Toolbars <td> &nbsp; <td> &nbsp; <td> x <td> x </tr>
<tr> <th> Tables <td> &nbsp; <td> &nbsp; <td> x <td> x </tr>
<tr> <th> Accessibility features <td> &nbsp; <td> &nbsp; <td> &nbsp; <td> x </tr>
<tr> <th> Object embedding <td> &nbsp; <td> &nbsp; <td> &nbsp; <td> x </tr>
<tr> <th> Scripting <td> &nbsp; <td> &nbsp; <td> &nbsp; <td> x </tr>
</table>
</body>
</html>

```

(الف)

برخی از تفاوت‌های در ویرایش HTML

آیتم	HTML 1.0	HTML 2.0	HTML 3.0	HTML 4.0
ایر لیتک	X	X	X	X
تصویر	X	X	X	X
لیست	X	X	X	X
تصویر یا نقش فعال		X	X	X
فرم		X	X	X
معادله			X	X
میله ابزار			X	X
جدول			X	X
ویژگی معلولین				X
قرار دادن شیء در صفحه				X
اسکرپت نویسی				X

(ب)

شکل ۷-۲۸. (الف) یک جدول HTML. (ب) خروجی جدول.

این تقاضاها باعث شد تا فرم (form) به ویرایش بعدی یعنی HTML 2.0 اضافه شود. فرم می‌تواند فیلدها (یا دکمه‌هایی) داشته باشد، که کاربر آنها را پُر کرده و به سرویس دهنده برگرداند. برای این منظور از برچسبی بنام `<input>` استفاده می‌کنیم، که دارای پارامترهای متعددی برای کنترل ظاهر و عملکرد آن است. رایجترین فرمها معمولاً دارای یک یا چند فیلد برای وارد کردن متن، یک یا چند جعبه برای انتخاب کردن، نگاشتهای فعال، و دکمه‌های `submit` باشند. در شکل ۷-۲۹ برخی از خصوصیات فرم نشان داده شده است.

اجازه دهید برای آشنایی بیشتر با فرمها، این مثال را دقیقتر بررسی کنیم. هر فرم با برچسبهای `<form>` و `</form>` مشخص می‌شود. در داخل یک فرم می‌توان از تمام برچسبهای HTML استفاده کرد، اما هر متنی که خارج این برچسبها نوشته شود، بهمان صورت دیده خواهد شد. سه نوع جعبه ورودی (input box) وجود دارد که می‌توان از آنها در فرمها استفاده کرد.

در شکل ۷-۲۹، اولین جعبه ورودی بعد از کلمه "Name" آمده است. این جعبه ورودی رشته‌ای بطول حداکثر ۴۶ کاراکتر می‌گیرد، و آنرا در متغیری بنام `customer` ذخیره می‌کند. برچسب `<p>` هم باعث می‌شود که مرورگر فیلدهای بعدی را در خط بعد نشان دهد. با استفاده از برچسب `<p>` طراح صفحه می‌تواند ظاهر آنرا بصورت دلخواه کنترل کند.

خط بعدی فرم یک آدرس (بطول ۴۰ کاراکتر) از کاربر می‌گیرد. فیلدهای شهر، استان، و کشور هم پدنبال آن آمده‌اند؛ بین این فیلدها `<p>` وجود ندارد، بنابراین مرورگر آنها را در یک خط نشان می‌دهد. تا آنجا که به مرورگر مربوط است، این پاراگراف شش آیتم دارد (سه رشته متن، و سه جعبه ورودی)، که باید آنها را یک خط نمایش دهد (و اگر نتوانست به خط بعدی می‌رود). اگر وضوح تصویر مانیتور زیاد باشد، این شش آیتم در یک خط دیده خواهند شد، اما اگر وضوح آن پائین باشد، احتمالاً به دو خط شکسته می‌شوند (حتی ممکنست کلمه "Country" در آخر خط، و جعبه ورودی مقابل آن در ابتدای خط بعدی بیفتد).

خط بعدی شماره کارت اعتباری و تاریخ انقضای آنرا از کاربر می‌گیرد (البته ارسال این قبیل اطلاعات روی اینترنت فقط باید در شرایط کاملاً امن صورت گیرد - در فصل ۸ در این باره بیشتر صحبت خواهیم کرد).

بعد از فیلد تاریخ انقضا، نوع دیگری از جعبه ورودی را می‌بینید: دکمه رادیویی (`radio button`). از دکمه رادیویی وقتی استفاده می‌شود که بخواهیم انتخاب کاربر را به دو یا چند گزینه محدود کنیم (مانند رادیوی ماشین که با زدن هر دکمه می‌توان یک ایستگاه از پیش تعیین شده را انتخاب کرد). کاربر می‌تواند برای انتخاب هر یک از این دکمه‌ها روی آن کلیک کند (و یا از صفحه کلید استفاده کند). انتخاب هر دکمه، باعث می‌شود که دکمه‌های دیگر همان گروه خاموش شوند (نمایش ظاهری دکمه‌های خاموش و روشن بر عهده مرورگر است). گروه‌بندی دکمه‌های رادیویی (و ایجاد ارتباط منطقی بین آنها) توسط صفت `name` صورت می‌گیرد. برای مثال، Widget size گروه مستقلیست که آن هم دو دکمه دارد.

برای مشخص کردن اینکه در هر گروه کدام دکمه روشن است، از پارامتر `value` استفاده می‌کنیم. برای مثال، در گروه اول مقدار متغیر `cc` ("mastercard" یا "visacard") نشان می‌دهد که کاربر کدامیک از دکمه‌های M/C یا Visa را انتخاب کرده است.

بعد از این دکمه‌های رادیویی، نوع دیگری از جعبه ورودی را می‌بینید: جعبه چک (`checkbox`). یک جعبه چک نیز مانند دکمه رادیویی می‌تواند دو حالت داشته باشد: روشن، خاموش. اما برخلاف دکمه‌های رادیویی، هر جعبه چک می‌تواند مستقلاً خاموش یا روشن باشد. برای مثال، وقتی می‌خواهید از طریق وب سفارش پیتزا بدهید، می‌توانید قارچ، گوشت و پیاز را همزمان بعنوان ترکیبات یک پیتزا انتخاب کنید، ولی امکان انتخاب همزمان اندازه‌های کوچک، متوسط و بزرگ را برای یک پیتزا ندارید. بهمین دلیل ترکیبات پیتزا را با استفاده از جعبه چک نشان می‌دهند، و اندازه آنرا با دکمه‌های رادیویی.

```

<html>
<head> <title> AWI CUSTOMER ORDERING FORM </title> </head>
<body>
<h1> Widget Order Form </h1>
<form ACTION="http://widget.com/cgi-bin/widgetorder" method=POST>
<p> Name <input name="customer" size=46> </p>
<p> Street Address <input name="address" size=40> </p>
<p> City <input name="city" size=20> State <input name="state" size =4>
Country <input name="country" size=10> </p>
<p> Credit card # <input name="cardno" size=10>
Expires <input name="expires" size=4>
M/C <input name="cc" type=radio value="mastercard">
VISA <input name="cc" type=radio value="visacard"> </p>
<p> Widget size Big <input name="product" type=radio value="expensive">
Little <input name="product" type=radio value="cheap">
Ship by express courier <input name="express" type=checkbox> </p>
<p><input type=submit value="submit order"> </p>
Thank you for ordering an AWI widget, the best widget money can buy!
</form>
</body>
</html>

```

(الف)

(ب)

شکل ۷-۲۹. (الف) یک فرم HTML. (ب) خروجی فرمت شده فرم.

در مواردی که تعداد گزینه‌های موجود برای یک انتخاب بسیار زیاد باشد، استفاده از دکمه‌های رادیویی قدری مشکل است. در این موارد می‌توان از برچسب `<select>` استفاده کرد: این برچسب با پارامتر *multiple* مانند

جعبه چک عمل می کند، و بدون آن مانند دکمه رادیویی. در اغلب مرورگرها برچسب `<select>` بصورت یک منوی بازشو (drop-down menu) نمایش داده می شود.

تا اینجا دو نوع `<input>` را دیدید: `radio` و `checkbox`. البته نوع سوم را هم قبل از آن دیده بودید: نوع `text`؛ ولی از آنجائیکه این نوع پیش فرض `<input>` است، استفاده از صفت `type = text` ضرورتی ندارد. انواع دیگر جعبه ورودی عبارتند از: `password` و `textarea`. جعبه `password` درست مانند `text` است، با این تفاوت که هر چیزی که در آن نوشته شود، فقط * نشان می دهد. جعبه `textarea` نیز مانند `text` است، فقط چندخطی است.

در آخرین قسمت از فرم شکل ۷-۲۹ یک دکمه `submit` می بینید. وقتی کاربر این دکمه را کلیک کند، اطلاعاتی که در سایر قسمت های فرم وارد کرده به کامپیوتری که فرم روی آن قرار دارد، فرستاده می شود. در اینجا پارامتر `value` عبارتست که روی دکمه `submit` دیده خواهد شد. دکمه `submit` تنها جعبه ورودی است که باید `value` داشته باشد، در سایر جعبه ها این پارامتر اختیاری است (چون کاربر می تواند متن موجود در آنها را بدلتخواه تغییر دهد). با کلیک شدن دکمه `submit`، مرورگر تمام اطلاعات فرم را در یک خط ترکیب کرده و به سرویس دهنده برمی گرداند. این فیلدها با & از هم جدا می شوند، و اگر در آنها فاصله وجود داشته باشد، مرورگر بجای آن + قرار می دهد. در شکل ۷-۳۰ نمونه ای از مقدار برگشتی فرم ۷-۲۹ به سرویس دهنده را می بینید (تمام این اطلاعات در واقع فقط یک خط است، که بدلیل محدودیت عرض صفحه به چند خط شکسته شده است).

```
customer=John+Doe&address=100+Main+St.&city=White+Plains&
state=NY&country=USA&cardno=1234567890&expires=6/98&cc=mastercard&
product=cheap&express=on
```

شکل ۷-۳۰. ورودیهای کاربر که مرورگر به سرویس دهنده برمی گرداند.

(اگر یک جعبه چک انتخاب نشده باشد، مرورگر آنرا به سرویس دهنده نمی فرستد.) تفسیر اطلاعات رسیده بر عهده سرویس دهنده است - در این باره بعداً بیشتر صحبت خواهیم کرد.

XSL و XML

HTML، با فرم یا بدون آن، هیچگونه ساختاری برای صفحات وب فراهم نمی آورد. در HTML محتوای صفحه و فرمانهای فرمت کننده نیز مخلوط هستند. با گسترش روزافزون تجارت الکترونیک (و سایر کاربردها)، نیاز به نوعی ساختار برای صفحات وب (و تفکیک محتوا و فرمت) بالا گرفت. برای مثال، برنامه ای که می خواهد در وب بدنبال بهترین قیمت یک کتاب (یا CD) جستجو کند، باید صفحات متعددی را خوانده و در آنها بدنبال عنوان کتاب و قیمت آن بگردد. وقتی صفحات وب با HTML نوشته شده باشند، برای چنین برنامه ای دشوار است تشخیص دهد عنوان کتاب کجاست و قیمت آن کجا.

به همین دلیل کنسرسیوم W3C استاندارد جدیدی برای HTML توسعه داده، که به کمک آن می توان به صفحات وب ساختاری مناسب برای پردازش خودکار داد. برای این هدف دو زبان جدید نیز توسعه داده شده است. اولی، XML (زبان علامتگذاری قابل توسعه - eXtensible Markup Language)، محتویات صفحه وب را بصورت ساخت یافته توصیف می کند، و دومی، XSL (زبان شیوه نویسی قابل توسعه - eXtensible Style Language)، برای توصیف مستقل فرمت این محتواست. هر دوی این زبانها بسیار مفصل و پیچیده اند، و بحث این قسمت فقط ایده ای از طرز کار آنها به شما خواهد داد.

مثال شکل ۷-۳۱ را در نظر بگیرید. در این مثال ساختاری بنام `book_list`، برای نگهداری مشخصات کتابها، تعریف شده است. هر کتاب سه فیلد دارد: عنوان، مؤلف، و سال انتشار. این ساختار بسیار ساده است، اما می توان ساختارهای پیچیده تری هم ایجاد کرد (مانند کتابهایی که چند مؤلف دارند، کتابهایی که CD پیوست دارند، و یا URL سایتهای عرضه و فروش کتاب).

در این مثال هر فیلد فقط یک قسمت دارند، اما فیلدها را می توان به فیلدهای کوچکتری نیز تقسیم کرد. بعنوان مثال، می توان فیلد `<author>` را به نام و نام خانوادگی تقسیم کرد، تا جستجوهای دقیقتر ممکن شود (این تقسیم را می توان تا هر درجه ای از عمق انجام داد):

```
<author>
  <first_name>Andrew</first_name>
  <last_name>Tanenbaum</last_name>
</author>
```

تمام کاری که فایل ۷-۳۱ انجام می دهد، ایجاد فهرستی از سه کتاب است. این فایل هیچ حرفی درباره نحوه نمایش این اطلاعات نمی زند. برای فرمت کردن این اطلاعات به فایل دیگری نیاز داریم: فایل `book_list.xsl`، که حاوی تعریفهای XSL لازم برای فرمت کردن فهرست `book_list.xml` است. این فایل یک شیوه نامه است که نحوه فرمت کردن صفحه وب را بیان می کند. (البته روشهای دیگری برای فرمت کردن فایل های XML وجود دارد - مانند تبدیل XML به HTML - که از حوزه بحث این کتاب خارج است.)

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="b5.xsl"?>
<book_list>
<book>
  <title> Computer Networks, 4/e </title>
  <author> Andrew S. Tanenbaum </author>
  <year> 2003 </year>
</book>
<book>
  <title> Modern Operating Systems, 2/e </title>
  <author> Andrew S. Tanenbaum </author>
  <year> 2001 </year>
</book>
<book>
  <title> Structured Computer Organization, 4/e </title>
  <author> Andrew S. Tanenbaum </author>
  <year> 1999 </year>
</book>
</book_list>
```

شکل ۷-۳۱. یک صفحه وب که با XML نوشته شده است.

در شکل ۷-۳۲ یک فایل XSL نمونه برای فرمت کردن فایل ۷-۳۱ را ملاحظه می کنید. بعد از چند تعریف لازم (مانند URL استاندارد XSL)، برچسبهای `<html>` و `<body>` آمده اند، که اینها (مانند همیشه) شروع صفحه وب را مشخص می کنند. پس از آن جدولی با یک تیر و سه ستون تعریف کرده ایم. دقت کنید که در این

جدول علاوه بر برچسبهای <th> از برچسبهای </th> نیز استفاده کرده ایم، کاری که قبلاً نمی کردیم. استانداردهای XML و XSL بسیار سختگیرتر از HTML هستند، و رعایت اصول نوشتن برچسبها در آنها لازم است، حتی اگر مرورگر قادر به تشخیص نیت طراح صفحه وب باشد. مرورگری که فایل های غلط XML و XSL را قبول کند (و خود اشکالات آنها را رفع کند)، در امتحانات سازگاری نمره قبولی نخواهد گرفت - مرورگرها فقط می توانند خطاها را اعلام کنند. این مقررات سختگیرانه برای مقابله با افراد تنبلی که وب را پر از صفحات شلخته کرده اند، لازم است.

```
<?xml version='1.0'?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:template match="/">
<html>
<body>
<table border="2">
  <tr>
    <th> Title</th>
    <th> Author</th>
    <th> Year </th>
  </tr>

  <xsl:for-each select="book_list/book">
    <tr>
      <td> <xsl:value-of select="title"/> </td>
      <td> <xsl:value-of select="author"/> </td>
      <td> <xsl:value-of select="year"/> </td>
    </tr>
  </xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

شکل ۷-۳۲. یک شیوه نامه XSL.

دستور

```
<xsl:for-each select="book_list/book">
```

معادل دستور for در زبان C است. این دستور (که به </xsl:for-each> ختم می شود) در یک حلقه کتابها را بترتیب می خواند. در هر تکرار این حلقه پنج فرمان HTML در خروجی نوشته می شود: <tr>، عنوان کتاب، مؤلف، سال انتشار کتاب، و </tr>. پس از اتمام حلقه و بستن جدول، برچسبهای </body> و </html> نیز در خروجی نوشته می شوند. حاصل کار یک جدول HTML است، که کاملاً شبیه جدولهای معمولی است. اما با این فرمت هر برنامه ای می تواند فایل XML را آنالیز کرده، و کتابهای موردنظر را استخراج کند. تأکید بر این نکته ضروریست که، اگر چه در فایل XSL نوعی حلقه وجود دارد ولی صفحات XML و XSL همچنان استاتیک هستند، چون فقط برای فرمت کردن صفحه وب بکار می آیند. البته مرورگر باید توانایی تفسیر فایل های XML و XSL را داشته باشد، ولی خبر خوب اینست که اغلب مرورگرهای امروزی چنین قابلیتی دارند. آینده XSL و اینکه آیا می تواند جایگزین شیوه نامه ها شود، هنوز در پرده ای از ابهام قرار دارد.

با اینکه روش کار را نشان ندادیم، XML به طراح سایت وب اجازه می‌دهد تا با تعریف فایل‌های ساختاری و ضمیمه کردن آنها به صفحات وب، سایت‌هایی پیچیده‌ای ایجاد کند. کتاب‌های متنوع و بسیار خوبی در این زمینه نوشته شده، که برای نمونه می‌توان به (Livingston, 2002; and Williamson, 2001) اشاره کرد. قبل از پایان دادن به بحث XML و XSL، بد نیست به جنگ ایدئولوژیکی که بین کنسرسیوم WWW و طراحان وب در گرفته، هم اشاره کنیم. هدف اولیه HTML مشخص کردن ساختار سند بود، نه ظاهر آن. برای مثال، فرمان

```
<h1>Deborah's Photos</h1>
```

فقط به مرورگر می‌گوید که این یک تیتراست، اما هیچ چیز درباره فرمت (نوع فونت، رنگ و یا اندازه) آن نمی‌گوید. این وظیفه بر عهده مرورگر گذاشته شده است. اما از آنجائیکه بسیاری از طراحان وب مایلند ظاهر صفحه را خود کنترل کنند، برای این کار برچسب‌های جدیدی به HTML اضافه شد:

```
<font face="helvetica" size="24" color="red">Deborah's Photos</font>
```

همچنین فرمان‌هایی برای کنترل دقیق محل اشیاء روی صفحه در نظر گرفته شد. اما مشکل این روش آن است که صفحاتی که با این دستورات نوشته می‌شوند، همه جا یکسان دیده نمی‌شوند (صفحه‌ای که در کامپیوتر طراح آن بسیار قشنگ دیده می‌شود، ممکنست در جای دیگر یک افتضاح واقعی باشد). XML تلاشی بود برای بازگشت به آن ایده‌های اولیه: مشخص کردن ساختار صفحه، نه ظاهر آن. اما هر ایده خوبی می‌تواند مورد سوءاستفاده قرار گیرد (در این حرف شک نکنید!).

علاوه بر توصیف صفحات وب، XML کاربردهای دیگری نیز دارد. برای مثال، می‌توان از XML بعنوان رابط بین برنامه‌های کاربردی استفاده کرد: SOAP (پروتکل ساده دسترسی شیء - Simple Object Access Protocol) یکی از راه‌های انجام RPC (فراخوانی از راه دور - Remote Procedure Call) بین برنامه‌ها (مستقل از زبان و سیستم عامل) است. در این روش، مشتری درخواست خود را بصورت یک پیام XML در آورده و با استفاده از پروتکل HTTP به سرویس دهنده می‌فرستد؛ پاسخ سرویس دهنده هم بصورت پیام XML به مشتری برگشت داده می‌شود. با این تکنیک برنامه‌هایی که روی سیستم عامل‌های متفاوت هستند، می‌توانند با یکدیگر ارتباط برقرار کنند.

XHTML

HTML برای پاسخ به نیازهای جدید به تحول خود ادامه می‌دهد. از هم اکنون می‌توان حدس زد که در آینده صفحات وب فقط به PC ها محدود نمی‌مانند، و راه خود را به دستگاه‌های تلفن همراه و PDA ها باز خواهند کرد. این قبیل دستگاه‌ها حافظه کمی دارند، و نمی‌توانند مرورگرهای حجیم و سنگین امروزی (که قسمت اعظم کد آنها صرف حدس زدن و رفع و رجوع خرابکاری طراح صفحه وب می‌شود) را اجرا کنند. به همین دلیل، بعد از HTML 4 (بجای HTML 5) زبان جدیدی بنام XHTML (HTML توسعه یافته - eXtended HTML) به بازار آمد، که بسیار هم ناخن خشک است. این زبان جدید اساساً همان HTML 4 است، که با XML فرمول بندی شده است. عبارت دیگر، در این زبان برچسبی مانند `<h1>` هیچ معنای خاصی ندارد، مگر اینکه در یک فایل XSL تعریف شده باشد. XHTML استاندارد جدید وب است، و اگر می‌خواهید بالاترین قابلیت جابجایی را در وب داشته باشید، باید از آن استفاده کنید.

بین HTML 4 و XHTML شش تفاوت عمده (و چند تفاوت جزئی) وجود دارد، که در اینجا آنها را فهرست وار مرور خواهیم کرد. اول اینکه، صفحات و مرورگرهای XHTML باید استانداردهای آن را دقیقاً رعایت کنند (دیگر جایی برای بنجل‌ها نیست!). XHTML این ویژگی را از XML ارث برده است.

دوم، تمام برچسب ها و صفت های XHTML باید با حروف کوچک نوشته شوند. برچسبی مثل `<HTML>` دیگر در XHTML معتبر نیست، و باید آنرا بصورت `<html>` نوشت. دستور `` نیز غلط است، چون XHTML صفتی بنام SRC نمی شناسد. سوم، حذف برچسبهای انتهایی (حتی برچسبی مثل `</p>`) قدغن است. در برچسبهایی که ذاتاً انتها ندارند (مانند `
`، `<hr>`، و ``)، نیز باید از `</>` استفاده کرد:

```

```

چهارم، مقدار تمام صفت ها باید در داخل گیومه نوشته شود (حتی اگر عدد باشد). مثلاً، دستور

```

```

مجاز نیست، چون 500 در داخل "" قرار ندارد.

پنجم، تودرتو کردن برچسبها باید بدرستی انجام شود. در گذشته این کار لزومی نداشت، چون مرورگر منظور طراح صفحه را حدس می زد. برای مثال، 4 HTML دستور زیر را بدرستی اجرا می کند:

```
<center><b>Vacation Pictures</center></b>
```

اما این دستور در XHTML مجاز نیست (جای `</center>` و `` باید عوض شود).

و بالاخره ششم اینکه، نوع هر سند باید دقیقاً (در ابتدای آن) مشخص شده باشد. (برای دیدن سایر تفاوت های XHTML و HTML می توانید به سایت www.w3c.org مراجعه کنید).

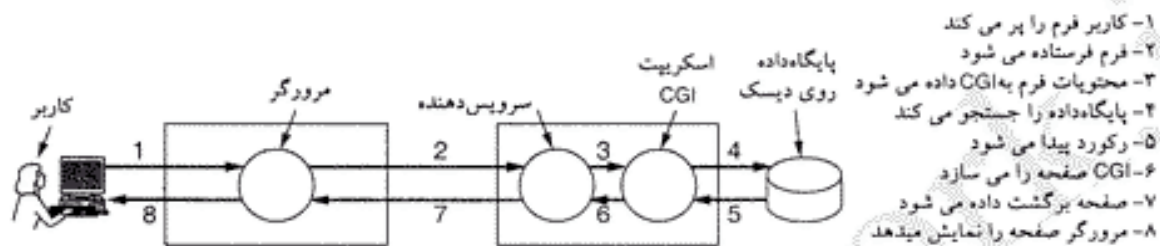
۳-۳-۷ سندهای وب دینامیک

تا اینجا با مدلی که در شکل ۶-۶ دیدید سروکار داشتیم: مشتری فایلی را از سرویس دهنده درخواست می کند، و سرویس دهنده این فایل را برای او می فرستد. در روزهای اولیه وب اوضاع همیشه بر همین منوال بود، یعنی صفحه ها ثابت بودند و هیچ تغییری نمی کردند. اما در سالهای اخیر روز به روز بر تعداد صفحاتی که محتویات آنها بصورت دینامیک تولید می شود (و از قبل فایلی روی دیسک سرویس دهنده وجود ندارد)، اضافه شده است. تولید محتوا می تواند در دو نقطه صورت گیرد: سمت سرویس دهنده، و سمت مشتری. اجازه دهید هر یک از آنها را بررسی کنیم.

تولید صفحات وب دینامیک سمت - سرویس دهنده

برای اینکه ببینید اساساً چرا تولید محتوا در سمت سرویس دهنده احتیاج است، فرمهایی را که در قسمت قبل درباره آنها صحبت کردیم، در نظر بگیرید. وقتی کاربر فرم را پُر کرده و دکمه `submit` را کلیک می کند، پیامی به سرویس دهنده فرستاده شده و مقدار فیلدهای فرم را به آن اعلام می کند. این پیام هیچ چیز درباره اینکه سرویس دهنده چه فایلی باید برگرداند، نمی گوید - و در واقع سرویس دهنده قبل از هر کاری باید این اطلاعات را (توسط یک برنامه یا اسکریپت) پردازش کند. معمولاً از اطلاعات فرم برای جستجوی یک پایگاه داده (روی سرویس دهنده) و ایجاد یک صفحه HTML خاص استفاده می شود. برای مثال، وقتی در یک برنامه تجارت الکترونیک کاربر دکمه `PROCEED TO CHECKOUT` را کلیک می کند، مرورگر کوکی محتوی اقلام موجود در سبد خرید کاربر را هم به همراه آن به سرویس دهنده می فرستد، و این سرویس دهنده است که باید با استفاده از اطلاعات این کوکی صفحه HTML لازم را ایجاد کند. بعنوان مثال، این صفحه می تواند اقلام انتخاب شده را به کاربر نشان دهد، و ضمن نمایش اطلاعات دیگر (از قبیل آدرس خریدار، و شماره کارت اعتباری) تأیید نهایی وی را برای شروع عملیات مالی اخذ کند. در شکل ۷-۳۳ مراحل پردازش اطلاعات فرم HTML نشان داده شده است. روش سنتی پردازش فرمها و دیگر صفحات تعاملی وب سیستمی است بنام CGI (واسط مشترک دروازه -

CGI (Common Gateway Interface). یک واسطه استاندارد شده است که به سرویس دهنده وب اجازه می دهد تا با برنامه ها یا اسکریپت های پشت صحنه (که می توانند اطلاعات ورودی را از فرمها گرفته، و صفحات HTML تولید کنند) ارتباط برقرار کند. برنامه های پشت صحنه معمولاً به زبان اسکریپت نویسی پِریل (Perl) نوشته می شوند، چون نوشتن آنها ساده تر و سریعتر است (البته اگر پِریل بلد باشید). این قبیل برنامه ها را معمولاً در یک دایرکتوری بنام *cgi-bin* ذخیره می کنند (که در اغلب URL ها می توانید این موضوع را ببینید). زبانهای اسکریپت نویسی دیگری هم هست (مانند پایتون - Python)، که می توان بجای پِریل از آنها استفاده کرد.



شکل ۷-۳۳. مراحل پردازش اطلاعات یک فرم HTML.

برای دیدن طرز کار CGI، فرض کنید شرکت Truly Great Products Company محصولات خود را بدون کارت ضمانت نامه می فروشد، ولی از خریداران دعوت می کند که برای پُر کردن کارت ضمانت نامه به سایت www.tgpc.com مراجعه کنند. وقتی خریدار به این صفحه مراجعه می کند، لینک زیر را در آن می بیند:

Click here to register your product

این لینک به اسکریپت www.tgpc.com/cgi-bin/reg.perl اشاره می کند. وقتی این اسکریپت بدون هیچ پارامتری اجرا شود، یک صفحه HTML شامل فرم ضمانت نامه به کاربر برمی گرداند. وقتی کاربر بعد از پُر کردن فرم دکمه *submit* را کلیک کند، پیامی حاوی اطلاعات فرم به اسکریپت *reg.perl* فرستاده می شود. اسکریپت *reg.perl* پس از پردازش اطلاعات فرم، یک رکورد برای مشتری جدید در پایگاه داده مشتریان ایجاد کرده، و یک صفحه HTML حاوی شماره ضمانت نامه محصول و تلفن پشتیبانی پس از فروش را به کاربر برمی گرداند. این تنها راه پردازش دینامیک اطلاعات نیست، اما متداولترین روش است. صدها کتاب درباره اسکریپت های CGI و برنامه نویسی پِریل نوشته شده، که بعنوان نمونه می توانیم به Hanegan, 2001; Lash, 2002; and Meltzer and Michalski, 2001 اشاره کنیم.

اسکریپت های CGI تنها روش تولید محتویات دینامیک در سمت سرویس دهنده نیست. روش متداول دیگر نوشتن اسکریپت های کوچک در داخل صفحات HTML و اجرای این اسکریپت ها توسط سرویس دهنده است. یکی از زبانهای رایج برای نوشتن این قبیل اسکریپت ها PHP (صفحه خانگی شخصی - Personal Home Page) نام دارد. البته سرویس دهنده باید این زبان را بشناسد تا بتواند اسکریپت های آنرا اجرا کند (درست مثل مرورگر که باید XML را بشناسد تا بتواند صفحات XML را نمایش دهد). صفحاتی که حاوی کد PHP هستند، معمولاً بجای *htm* یا *html* پسوند *php* دارند.

در شکل ۷-۳۴ یک اسکریپت کوچک PHP را ملاحظه می کنید؛ این اسکریپت با هر سرویس دهنده ای که PHP روی آن نصب شده باشد، کار می کند. در این صفحه (علاوه بر برجسبهای معمول HTML) یک دستور PHP که در داخل برجسب `<?php ... ?>` قرار دارد، می بینید. این صفحه به کاربر اعلام می کند که از وی چه می داند. این قبیل اطلاعات معمولاً توسط مرورگر (به همراه کوکی ها) به سرویس دهنده فرستاده می شوند (و

سرویس دهنده آنها را از طریق متغیر `HTTP_USER_AGENT` بدست می آورد. اگر این صفحه در فایل بنام `test.php` (در سایت `abcd.com`) قرار داشته باشد، و کاربر روی لینک `www.abcd.com/test.php` کلیک کند، صفحه ای دریافت می کند که نوع مرورگر، زبان و سیستم عاملش را به وی اعلام خواهد کرد.

```
<html>
<body>
<h2> This is what I know about you </h2>
<?php echo $HTTP_USER_AGENT ?>
</body>
</html>
```

شکل ۷-۳۴. یک صفحه HTML با دستورات PHP.

PHP از CGI ساده تر، و بویژه برای پردازش فرمها مناسبتر است. برای آشنایی بیشتر با طرز کار PHP مثال شکل ۷-۳۵ (الف) را در نظر بگیرید. در این شکل یک فرم HTML می بینید؛ تنها تفاوت این فرم با فرمهای قبلی در خط اول آن است: این فرم هنگام کلیک شدن دکمه `submit` فایل بنام `action.php` را اجرا می کند. در این فرم دو جعبه ورودی از نوع `text` وجود دارد، که یکی نام و دیگری سن کاربر را می پرسد. بعد از کلیک شدن دکمه `submit` و ارسال محتویات فرم به سرویس دهنده (مانند آنچه در شکل ۷-۳۰ دیدید)، سرویس دهنده مقدار فیلد اول را در متغیری بنام `name` و مقدار فیلد دوم را در متغیری بنام `age` قرار می دهد. پس از آن فایل `action.php` (شکل ۷-۳۵ ب) را پردازش کرده، و دستورات PHP آن را اجرا می کند. اگر کاربر در فیلدهای فرم ۷-۳۵ (الف) بترتیب "Barbara" و "24" را بعنوان نام و سن خود وارد کرده باشد، فایلی شبیه شکل ۷-۳۵ (ج) به وی برگردانده خواهد شد. همانطور که می بینید، پردازش فرمهای HTML با PHP بسیار ساده شده است.

با اینکه PHP نسبتاً آسان است، اما زبانی بسیار قوی برای ارتباط با پایگاه داده محسوب می شود. متغیرها، رشته ها، آرایه ها، و اغلب ساختارهای کنترلی PHP شباهت زیادی با C دارد (و I/O آن نیز بسیار قوی است). کد PHP باز است و همه جا می توان آنرا مجانی بدست آورد. این زبان بویژه برای کار روی آپاچی (Apache) - یکی از پُرطرفدارترین سرویس دهنده های وب (طراحی شده است. برای اطلاعات بیشتر درباره PHP به Valade, 2002) مراجعه کنید.

تا اینجا دو روش مختلف برای ایجاد صفحات HTML دینامیک را بررسی کردیم: CGI و PHP. تکنیک سومی نیز وجود دارد، که JSP (صفحات سرویس دهنده جاوا - JavaServer Pages) نام دارد؛ این تکنیک مشابه PHP است، با این تفاوت که بجای PHP از زبان برنامه نویسی جاوا (Java) استفاده می کند (صفحاتی که با این روش نوشته شده اند، پسوند `jsp` دارند). تکنیک چهارم یعنی ASP (صفحات فعال سرویس دهنده - Active Server Pages) معادلیست برای PHP و JSP از شرکت میکروسافت. در این تکنیک از زبان برنامه نویسی VBScript (که آن هم از محصولات میکروسافت است) استفاده می شود (صفحات ASP پسوند `asp` دارند). انتخاب یکی از تکنیکهای PHP، JSP یا ASP بیشتر از آن که جنبه فنی داشته باشد، سیاسی است (دعای همیشگی طرفداران کد باز، سان و میکروسافت)، چون همه آنها تقریباً شبیه هم هستند. به مجموعه تکنیکهای ایجاد محتویات دینامیک وب گاهی HTML دینامیک نیز گفته می شود.

تولید صفحات وب دینامیک سمت-مشتری

تکنیکهای CGI، PHP، JSP، و ASP همگی برای پردازش اطلاعات فرمها (و ارتباط با پایگاه داده) در سمت

```

<html>
<body>
<form action="action.php" method="post">
<p> Please enter your name: <input type="text" name="name"> </p>
<p> Please enter your age: <input type="text" name="age"> </p>
<input type="submit">
</form>
</body>
</html>

```

(الف)

```

<html>
<body>
<h1> Reply: </h1>
Hello <?php echo $name; ?>.
Prediction: next year you will be <?php echo $age + 1; ?>
</body>
</html>

```

(ب)

```

<html>
<body>
<h1> Reply: </h1>
Hello Barbara.
Prediction: next year you will be 25
</body>
</html>

```

(د)

شکل ۷-۳۵. (الف) یک فرم HTML، (ب) اسکریپت PHP برای پردازش این فرم.

(ج) خروجی اسکریپت PHP برای ورودی‌های "Barbara" و "24".

سرویس دهنده هستند. این تکنیک‌ها با پردازش اطلاعات فرم و جستجو در پایگاه داده، صفحات HTML مناسب را تولید و به مشتری برمی‌گردانند. اما هیچکدام از آنها نمی‌توانند حرکات ماوس را تشخیص داده، و یا مستقیماً با کاربر ارتباط برقرار کنند. برای این کار باید از اسکریپتهایی که در دل صفحات HTML نوشته شده و در کامپیوتر مشتری اجرا می‌شوند، استفاده کرد. پای این قبیل اسکریپتها، که با برچسب <script> مشخص می‌شوند، از HTML 4.0 به وب باز شد. متداولترین زبان اسکریپت‌نویسی سمت مشتری جاوااسکریپت (JavaScript) است، پس ما هم همین زبان را مختصراً بررسی خواهیم کرد.

جاوااسکریپت یک زبان اسکریپت‌نویسی است، که ارتباط دوری با جاوا دارد، اما مسلماً جاوا نیست. مانند سایر زبانهای اسکریپت‌نویسی، جاوااسکریپت زبانی سطح-بالا است. برای مثال، فقط با یک خط می‌توان یک جعبه دیالوگ نمایش داد، ورودی کاربر را گرفت، و آنرا در یک متغیر ذخیره کرد. این ویژگی جاوااسکریپت را برای ایجاد صفحات وب تعاملی (interactive) بسیار مناسب کرده است. از طرف دیگر، این واقعیت که این زبان هنوز

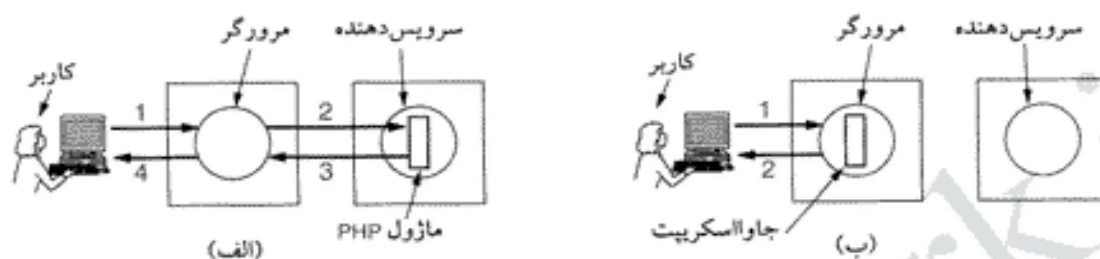
استاندارد نشده (و با سرعتی بیشتر از یک مگس گرفتن آمدن در ماشین اشعه X جهش پیدا می کند)، نوشتن یک برنامه جاوااسکریپت که بتواند روی هر سیستمی اجرا شود را فوق العاده مشکل کرده است. برنامه جاوااسکریپت شکل ۷-۳۶ را در نظر بگیرید (این برنامه هم مانند شکل ۷-۳۵ الف) نام و سن کاربر را گرفته، و پیش بینی می کند که وی سال آینده چند سال خواهد داشت!). قسمت `<body>` تقریباً با برنامه PHP یکسان است؛ تنها جایی که تفاوت کرده، قسمت تعریف دکمه `submit` است. در اینجا، صفت `onclick` به مرورگر می گوید که هنگام کلیک شدن دکمه باید اسکریپت `response` را اجرا کرده، و فرم `form` را بعنوان پارامتر به آن بدهد. اما تعریف تابع جاوااسکریپت `response` در قسمت `<head>` این صفحه کاملاً جدید است. این تابع ابتدا مقدار فیلد `name` فرم را استخراج کرده و آنرا به همان صورت در متغیری بنام `person` ذخیره می کند؛ سپس مقدار فیلد `age` فرم را هم خوانده، و پس از تبدیل آن به عدد (با تابع `eval`) و اضافه کردن 1 به آن، آنرا در متغیر `years` ذخیره می کند. پس از آن یک سند (صفحه وب) برای خروجی باز کرده، با دستور `writeln` چهار خط در آن می نویسد، و آنرا می بندد. بعد از کامل شدن این صفحه، مرورگر آنرا مانند یک فایل HTML معمولی نمایش می دهد.

```
<html>
<head>
<script language="javascript" type="text/javascript">
function response(test_form) {
    var person = test_form.name.value;
    var years = eval(test_form.age.value) + 1;
    document.open();
    document.writeln("<html> <body>");
    document.writeln("Hello " + person + ".<br>");
    document.writeln("Prediction: next year you will be " + years + ".");
    document.writeln("</body> </html>");
    document.close();
}
</script>
</head>
<body>
<form>
Please enter your name: <input type="text" name="name">
<p>
Please enter your age: <input type="text" name="age">
<p>
<input type="button" value="submit" onclick="response(this.form)">
</form>
</body>
</html>
```

شکل ۷-۳۶. استفاده از جاوااسکریپت برای پردازش فرم.

درک این نکته بسیار مهم است، که با وجود شباهت برنامه های ۷-۳۵ و ۷-۳۶، آنها بطریق کاملاً متفاوتی پردازش می شوند. در شکل ۷-۳۵، بعد از آن که کاربر دکمه `submit` را کلیک کرد، مرورگر اطلاعات فرم را در رشته ای مانند شکل ۷-۳۵ جمع آوری کرده، و به سرویس دهنده ای که صفحه روی آن قرار دارد، می فرستد. سرویس دهنده بعد از دیدن نام فایل PHP، این اسکریپت را اجرا می کند. اسکریپت PHP مزبور نیز با استفاده از اطلاعات فرم، یک صفحه HTML ایجاد کرده و به مشتری برمی گرداند. اما وقتی در شکل ۷-۳۶ دکمه `submit`

کلیک می‌شود، مرورگر اسکریپت درون آنرا اجرا می‌کند - تمام کارها در داخل مرورگر انجام می‌شود، و هیچ تماسی با سرویس‌دهنده وجود ندارد. بهمین دلیل نتیجه کار معمولاً بلافاصله است، برخلاف اسکریپت PHP که رفت و برگشت صفحه ممکنست چندین ثانیه طول بکشد. تفاوت اسکریپت سمت سرویس‌دهنده و سمت-مشری در شکل ۷-۳۷ نشان داده شده است. در هر دو شکل، مرحله ۱ گرفتن اطلاعات از کاربر است؛ تفاوت این دو روش در شکل بخوبی مشخص است.



شکل ۷-۳۷. (الف) اسکریپت سمت سرویس‌دهنده با PHP. (ب) اسکریپت سمت-مشری با جاوااسکریپت.

اما این تفاوت بدان معنا نیست که جاوااسکریپت بهتر از PHP است - آنها کاربردهای کاملاً متفاوتی دارند. PHP (و زبانهای مشابه آن) اساساً برای پردازش اطلاعات پایگاه داده روی وب مناسب هستند، در حالیکه جاوااسکریپت بدرد نوشتن برنامه‌های تعاملی می‌خورد. نوشتن صفحه‌ای که PHP و جاوااسکریپت را با هم داشته باشد، کاملاً ممکن است چون آنها کارهای متفاوتی انجام می‌دهند. جاوااسکریپت یک زبان کامل است با تمام تواناییهای C و جاوا، و علاوه بر آن امکانات خاصی نیز برای پردازش صفحات وب (مانند کار با پنجره‌ها و فریم‌ها، ست کردن و خواندن کوکی، پردازش فرم و لینک) دارد. در شکل ۷-۳۸ یک برنامه جاوااسکریپت می‌بینید که در آن از یک تابع بازگشتی (recursive) استفاده شده است.

```
<html>
<head>
<script language="javascript" type="text/javascript">
function response(test_form) {
    function factorial(n) {if (n == 0) return 1; else return n * factorial(n - 1);}
    var r = eval(test_form.number.value);    // r = typed in argument
    document.myform.mytext.value = "Here are the results.\n";
    for (var i = 1; i <= r; i++)    // print one line from 1 to r
        document.myform.mytext.value += (i + "! = " + factorial(i) + "\n");
}
</script>
</head>
<body>
<form name="myform">
Please enter a number: <input type="text" name="number">
<input type="button" value="compute table of factorials" onclick="response(this.form)">
<p>
<textarea name="mytext" rows=25 cols=50> </textarea>
</form>
</body>
</html>
```

شکل ۷-۳۸. یک برنامه جاوااسکریپت برای محاسبه و چاپ فاکتوریل.

جاوااسکریپت همچنین می تواند حرکت ماوس روی اشیاء مختلف صفحه را دنبال کند. بدین ترتیب می توان کاری کرد که وقتی کاربر ماوس را روی قسمتهای خاصی از صفحه می برد، اتفاق خاصی بیفتد (مثلاً، باز شدن یک منو، عوض شدن تصویر یا رنگ نوشته ها). این قبیل برنامه ها (که نمونه ای از آنرا در شکل ۷-۳۹ می بینید) شادابی و سرزندگی خاصی به صفحات وب می دهند.

```
<html>
<head>
<script language="javascript" type="text/javascript">
if (!document.myurl) document.myurl = new Array();
document.myurl[0] = "http://www.cs.vu.nl/ ast/im/kitten.jpg";
document.myurl[1] = "http://www.cs.vu.nl/ ast/im/puppy.jpg";
document.myurl[2] = "http://www.cs.vu.nl/ ast/im/bunny.jpg";
function pop(m) {
    var urx = "http://www.cs.vu.nl/ ast/im/cat.jpg";
    popupwin = window.open(document.myurl[m], "mywind", "width=250,height=250");
}
</script>
</head>
<body>
<p> <a href="#" onmouseover="pop(0); return false;" > Kitten </a> </p>
<p> <a href="#" onmouseover="pop(1); return false;" > Puppy </a> </p>
<p> <a href="#" onmouseover="pop(2); return false;" > Bunny </a> </p>
</body>
</html>
```

شکل ۷-۳۹. یک صفحه وب تعاملی، که به حرکات ماوس عکس العمل نشان می دهد.

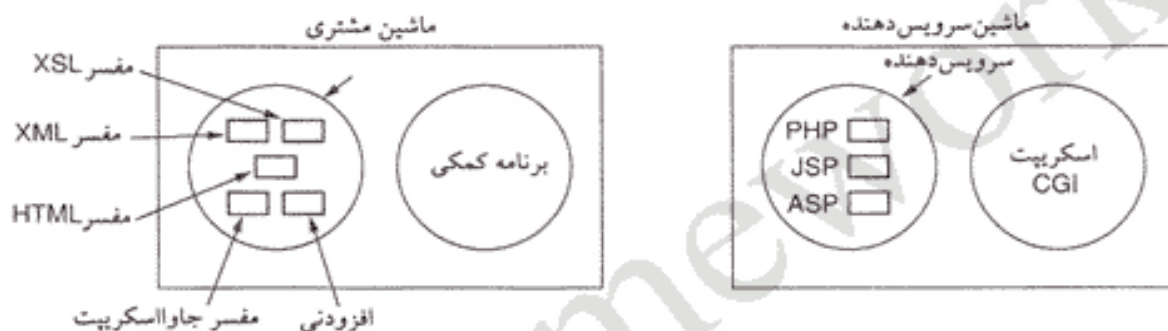
جاوااسکریپت تنها راه نوشتن صفحات وب تعاملی نیست؛ روش دیگر استفاده از اپلت (applet) است. اپلت عبارتست از یک برنامه کوچک جاوا، که برای یک ماشین مجازی بنام JVM (ماشین مجازی جاوا - Java Virtual Machine) کامپایل می شود. این اپلت ها را می توان (با استفاده از برچسب <applet>) در صفحات وب قرار داد، و مرورگرهایی که JVM را داشته باشند، آنها را اجرا خواهند کرد. از آنجائیکه اپلت های جاوا بطور مستقیم اجرا نمی شوند، مفسر جاوا می تواند جلوی اعمال مخرب آنها را بگیرد؛ حداقل در تئوری که اینطور گفته می شود. اما در عمل برنامه نویسان باهوش جاوا نارسایی های بیشماری در سیستم I/O جاوا شناسایی کرده اند، که می توان به کمک آنها در سیستم قربانی نفوذ کرد.

میکروسافت در پاسخ به اپلت های جاوا (که از ابداعات شرکت سان میکروسیستمز است) کنترل اکتیوایکس (ActiveX control) را معرفی کرد. کنترل های اکتیوایکس برنامه های کامپایل شده پتیوم هستند، که مستقیماً روی سخت افزار کامپیوتر مشتری اجرا می شوند. سرعت و توانایی های کنترل های اکتیوایکس بسیار بیشتر و بهتر از اپلت های جاواست، چون آنها برنامه های واقعی هستند. وقتی مرورگر اینترنت اکسپلورر در صفحه وب به یک کنترل اکتیوایکس برخورد می کند، آنرا بار کرده، و (پس از تعیین هویت) اجرا می کند. اما، همانطور که می توان حدس زد، بار کردن و اجرای برنامه های خارجی مشکلات امنیتی زیادی به همراه دارد، که در فصل آینده به آنها اشاره خواهیم کرد.

از آنجائیکه برنامه های جاوا و جاوااسکریپت روی (تقریباً) تمام مرورگرها اجرا می شوند، بهترین گزینه برای نوشتن صفحات تعاملی وب هستند، ولی اگر فقط مرورگرهای میکروسافت را هدف قرار داده اید، اکتیوایکس هم

به لیست امکانات شما اضافه خواهد شد. بطور کلی، نوشتن برنامه های جاوااسکریپت از همه ساده تر است، اپلت های جاوا سریعتر اجرا می شوند، و کنترل های اکتیوایکس از هر دوی آنها سریعترند. از طرف دیگر، از آنجائیکه اکثر قریب به اتفاق مرورگرها از یک JVM یکسان استفاده می کنند (در حالیکه پیاده سازی جاوااسکریپت در هیچ دو مرورگری یکسان نیست)، قابلیت انتقال اپلت های جاوا بیشتر از برنامه های جاوااسکریپت است. در زمینه برنامه نویسی جاوااسکریپت کتابهای متعدد و مفصلی وجود دارد، که از میان آنها می توان به (Easttom, 2001; Harris, 2001; and McFedries, 2001) اشاره کرد.

قبل از خاتمه دادن به بحث صفحات وب دینامیک، اجازه دهید یک بار دیگر آنچه را تا اینجا دیدیم مرور کنیم. با استفاده از اسکریپت ها می توان صفحات وب را بصورت کاملاً دینامیک در سرویس دهنده ایجاد کرد. این صفحات در کامپیوتر مشتری درست مثل صفحات معمولی HTML نمایش داده می شوند، و هیچ تفاوتی با آنها نخواهند داشت. اسکریپت ها را می توان با پرل، PHP، JSP، یا ASP نوشت (شکل ۷-۴۰ را ببینید).



شکل ۷-۴۰. روشهای مختلف ایجاد محتویات دینامیک و نمایش آنها.

تولید محتویات دینامیک در سمت مشتری نیز امکانپذیر است. صفحات وب را می توان با XML نوشت، و سپس برای تبدیل آنها به HTML از فایل های XSL استفاده کرد. با برنامه های جاوااسکریپت هم می توان محاسبات مورد نیاز را انجام داد. و بالاخره، با استفاده از برنامه های کمکی می توان محتویات صفحه را بطرق مختلف فرمت کرد.

۷-۳-۴ پروتکل انتقال آبرمتن - HTTP

پروتکل انتقال در سراسر وب HTTP (پروتکل انتقال آبرمتن - HyperText Transfer Protocol) است. این پروتکل مشخص می کند که مشتری چه چیزهایی می تواند به سرویس دهنده بفرستد، و سرویس دهنده چه پاسخهایی می تواند به آن بدهد. درخواست ها از نوع متنی (ASCII) هستند، و پاسخ سرویس دهنده یکی از انواع RFC 822 MIME، تمام مشتری ها و سرویس دهنده ها باید از این پروتکل پیروی کنند. پروتکل HTTP در RFC 2616 تعریف شده است. در این قسمت مهمترین ویژگی های این پروتکل را مورد بررسی قرار خواهیم داد.

اتصال

مرورگرها معمولاً از طریق اتصال TCP به پورت 80 سرویس دهنده با آن ارتباط برقرار می کند، اگرچه این الزامی رسمی نیست. خوبی اتصال TCP آنست که مرورگر یا سرویس دهنده هیچکدام لازم نیست نگران گم شدن پیامها، پیامهای تکراری یا خیلی بلند، و یا برگرداندن تصدیق دریافت باشند، چون تمام این کارها را TCP انجام می دهد. در HTTP 1.0، بعد از برقراری اتصال یک درخواست فرستاده شده و یک پاسخ دریافت می شود، و پس از آن اتصال قطع خواهد شد. در آن زمانی که صفحات HTML فقط متن بودند، این روش کاملاً کفایت می کرد. اما

خیلی زود صفحات وب پر شد از تصویر، آیکون و چیزهایی مانند آن، که برقرای یک اتصال TCP برای انتقال هر کدام از آنها اصلاً مقرون بصرفه نبود.

برای حل این مشکل HTTP 1.1 عرضه شد، که از اتصال پایدار (persistent connection) پشتیبانی می کرد. اتصال پایدار، اتصالیست که می توان روی آن چندین بار درخواست و پاسخ ردوبدل کرد. بدین ترتیب، سربراره TCP برای هر صفحه وب بسیار کمتر خواهد شد. در این روش امکان استفاده از تکنیک خطلوله (pipeline - ارسال درخواست ۲ قبل از برگشت پاسخ درخواست ۱) هم وجود دارد، که سرعت بار کردن صفحات را بسیار بهتر می کند.

متد

با اینکه HTTP برای استفاده در وب طراحی شد، اما طراحان آن (هوشمندانه) نگاهی به آینده برنامه های شیء-گرا نیز داشتند. به همین دلیل در HTTP عملیاتی غیر از درخواست صفحات وب نیز پیش بینی شده است، که به آنها متد (method) می گویند. از همین جاست که پروتکل SOAP امکان وجود پیدا کرد. هر درخواست HTTP یک (یا چند) خط متن ASCII است، که با نام متد شروع می شود. در شکل ۷-۴۱ متدهایی که HTTP پشتیبانی می کند، را می بینید. اضافه کردن متدهای جدید به HTTP (برای انجام کارهای جدید) نیز امکانپذیر است. نام متدها در HTTP به نوع حروف حساس است، بنابراین متد GET را نمی توان بصورت get نوشت.

متد	توضیح
GET	تقاضای خواندن صفحه وب
HEAD	تقاضای خواندن سر صفحه وب
PUT	تقاضای ذخیره کردن صفحه وب
POST	اضافه کردن به یک منبع نام دار (یعنی صفحه وب)
DELETE	حذف صفحه وب
TRACE	برگرداندن درخواست ورودی
CONNECT	برای آینده رزرو شده است
OPTIONS	جستجوی گزینه های خاص

شکل ۷-۴۱. متدهای HTTP.

برای درخواست ارسال یک صفحه وب از سرویس دهنده (یا هر شیء دیگر، و عبارت کلی تر، یک فایل) از متد GET استفاده می کنیم. صفحه ارسال شده بصورت MIME گُذ می شود. بیشترین درخواستها از سرویس دهنده های وب همین متد GET است، که شکل کلی آن چنین است:

GET filename HTTP/1.1

که در آن filename نام شیء موردنظر، و 1.1 ویرایش پروتکل مورداستفاده است.

متد HEAD فقط سرآیند صفحه (و نه خود آن) را درخواست می کند. از این متد می توان برای تست به روز بودن صفحات، و یا تست معتبر بودن URL ها استفاده کرد.

متد PUT عکس GET است: این متد یک صفحه را به سرویس دهنده می فرستد. محتویات صفحه موردنظر (که می تواند بصورت MIME گُذ شده باشد) در بدنه متد PUT می آید؛ در این متد سرآیند احراز هویت (authentication) - برای اثبات اینکه مشتری اجازه نوشتن محتویات در سرویس دهنده را دارد - نیز می تواند وجود داشته باشد.

متد *POST* تا حدی شبیه *PUT* است، با این تفاوت که محتویات جدید به انتهای داده‌های قبلی اضافه می‌شود. از این متد اغلب برای ارسال پیام به گروه‌های خبری یا سیستم‌های BBS (Bulletin Board System) استفاده می‌شود. در عمل، متدهای *PUT* و *POST* کاربرد بسیار محدودی دارند.

متد *DELETE* همان کاری را می‌کند، که از آن انتظار می‌رود: حذف صفحه (البته برای این کار هم مانند *PUT* داشتن مجوز ضروریست). هیچ تضمینی نیست که متد *DELETE* با موفقیت انجام شود، چون بسیار امکان دارد که وقتی سرویس دهنده *HTTP* می‌خواهد صفحه را حذف کند، فایل در وضعیتی باشد که حذف آن ممکن نباشد.

متد *TRACE* برای دیباگ بکار می‌رود: این متد به سرویس دهنده می‌گوید که درخواست را به همان شکلی که دریافت کرده، برگرداند. در مواردی که یک درخواست بدرستی پاسخ داده نمی‌شود، و می‌خواهیم بدانیم علت آن چیست، متد *TRACE* می‌تواند نشان دهد که سرویس دهنده درخواست را چگونه دریافت کرده است.

متد *CONNECT* در حال حاضر استفاده نمی‌شود، و برای آینده پیش‌بینی شده است.

متد *OPTIONS* اجازه می‌دهد تا مشتری اطلاعاتی از سرویس دهنده (یا یکی از فایل‌های آن) بدست آورد. هر درخواست یک پاسخ شامل یک خط وضعیت و احتمالاً مقداری اطلاعات دیگر (که می‌تواند صفحه وب یا بخشی از آن باشد) دریافت می‌کند. خط وضعیت (status line) شامل یک کد سه رقمی است که نشان می‌دهد آیا درخواست انجام شده، و اگر نشده، چرا. رقم اول این کد برای دسته‌بندی پاسخها بکار می‌رود؛ همانطور که در شکل ۷-۴۲ می‌بینید، پنج نوع پاسخ وجود دارد. کدهای 1xx در عمل بتدرت مورد استفاده قرار می‌گیرند. کدهای 2xx می‌گویند که درخواست پذیرفته شده، و اطلاعات خواسته شده در حال فرستاده شدن است. کدهای 3xx مشتری را بجای دیگری (یک URL دیگر، یا حافظه نهان خود مشتری) حواله می‌دهند. کدهای 4xx حاکی از عدم موفقیت درخواست (به علت خطا در درخواست مشتری، یا عدم وجود صفحه خواسته شده) هستند. کدهای 5xx هم حاکی از آن هستند که خود سرویس دهنده با خطا مواجه شده است (خطا در اجرای کد خواسته شده، یا ترافیک بیش از حد و عدم توانایی در پاسخ به موقع).

سرآیند پیام

معمولاً بدنبال خط درخواست (خطی که مثلاً متد *GET* در آن آمده) خط‌های دیگری (با اطلاعات بیشتر) نیز می‌آیند. به این خط‌ها (که می‌توان آنها را شبیه پارامتر دانست) سرآیند درخواست (request header) می‌گویند. پاسخها هم می‌توانند سرآیند پاسخ (response header) داشته باشند. برخی از این سرآیندها (که تعدادی از آنها را در شکل ۷-۴۳ می‌بینید) در هر دو جهت کاربرد دارند.

کد	مفهوم	مثال
1xx	Information	۱۰۰ = سرویس دهنده یا درخواست مشتری موافق است
2xx	Success	۲۰۰ = درخواست موفق بوده است، ۲۰۴ = محتویات موجود نیست
3xx	Redirection	۳۰۱ = صفحه جابجا شده، ۳۰۴ = صفحه موجود در حافظه نهان همچنان معتبر است
4xx	Client error	۴۰۳ = صفحه ممنوع، ۴۰۴ = پیدا نشد
5xx	Server error	۵۰۰ = خطای داخلی سرویس دهنده، ۵۰۳ = دوباره سعی کنید

شکل ۷-۴۲. کدهای پاسخ سرویس دهنده.

با سرآیند *User-Agent* مشتری می‌تواند اطلاعاتی درباره مرورگر و سیستم عامل خود (و چیزهایی از این قبیل) به سرویس دهنده بدهد. در اسکریپت شکل ۷-۳۴ دیدید که سرویس دهنده چگونه می‌تواند با استفاده از این سرآیند اطلاعات جالبی از مشتری نمایش دهد.

چهار سرآیند *Accept* به سرویس دهنده می گویند که مشتری چه چیزهایی را می تواند بپذیرد. اولین *Accept* نوع MIME موردپذیرش مشتری (مثلاً، *text/html*) را مشخص می کند، و دومی مجموعه کاراکتر آن را (مثلاً، ISO-8859-5 یا Unicode-1-1). سومین *Accept* نوع فشرده سازی (مثل، *gzip*)، و چهارمی زبان مشتری (مثلاً، ایتالیایی یا اسپانیایی) را اعلام می کنند (تا اگر سرویس دهنده امکان انتخاب زبان را فراهم کرده باشد، صفحه مناسب را بفرستد). اگر سرویس دهنده نتواند خواسته های مشتری را اجابت کند، یک کد خطا برمی گرداند.

سرآیند	نوع	محتویات
User-Agent	درخواست	اطلاعات درباره مرورگر و سیستم عامل آن
Accept	درخواست	نوع صفحاتی که مشتری می تواند بپذیرد
Accept-Charset	درخواست	مجموعه کارکترهای قابل قبول مشتری
Accept-Encoding	درخواست	کد گذاری های صفحه قابل قبول مشتری
Accept-Language	درخواست	زبانهای طبیعی قابل قبول مشتری
Host	درخواست	نام DNS سرویس دهنده
Authorization	درخواست	کوکی را به سرویس دهنده باز پس می فرستد
Cookie	درخواست	لیست احراز هویت مشتری
Date	هر دو	زمان و تاریخ ارسال پیام
Upgrade	هر دو	پروتکلی که فرستنده می خواهد به آن ارتقاء دهد
Server	پاسخ	اطلاعاتی درباره سرویس دهنده
Content-Encoding	پاسخ	نحوه کدگذاری محتویات صفحه
Content-Language	پاسخ	زبان طبیعی صفحه
Content-Length	پاسخ	طول صفحه (بر حسب بایت)
Content-Type	پاسخ	نوع MIME صفحه
Last-Modified	پاسخ	زمان و تاریخ آخرین تغییر صفحه
Location	پاسخ	فرمان به مشتری برای ارسال درخواست به جای دیگر
Accept-Ranges	پاسخ	سرویس دهنده محدوده بایت را پذیرفت
Set-Cookie	پاسخ	سرویس دهنده از مشتری می خواهد یک کوکی ذخیره کند

شکل ۷-۴۳. چند سرآیند پیام HTTP.

سرآیند *Host* نام سرویس دهنده را مشخص می کند، و از URL گرفته می شود. این سرآیند اجباریست، چون ممکنست چندین نام DNS دارای یک آدرس IP مشترک باشند، و سرویس دهنده باید بداند که درخواست مربوط به کدام میزبان است.

سرآیند *Authorization* برای صفحاتی که حفاظت شده هستند، لازم است (در این قبیل موارد، مشتری باید ثابت کند که اجازه دریافت صفحه خواسته شده را دارد).

با اینکه کوکی ها در RFC 2109 تعریف شده اند، در RFC 2616 نیز دو سرآیند برای کوکی ها در نظر گرفته شده است. مشتری برای ارسال کوکی به سرویس دهنده ای که قبلاً کوکی را از آن گرفته، از سرآیند *Cookie* استفاده می کند. سرآیند *Date* یکی از سرآیندهای دو طرفه است، و برای مشخص کردن زمان ارسال پیام بکار می رود. سرآیند *Upgrade* برای تسهیل فرآیند ارتقاء (ویرایشهای ناسازگار) پروتکل HTTP در نظر گرفته شده است. این سرآیند به مشتری (یا سرویس دهنده) اجازه می دهد تا اعلام کند از چه ویرایشی پشتیبانی می کند.

سرآیندهای بعدی همگی خاص سرویس دهنده هستند، که در پاسخها از آنها استفاده می کند. با اولین آنها، یعنی *Server*، سرویس دهنده خود را معرفی کرده و برخی از مشخصاتش را اعلام می کند.

چهار سرآیند بعدی، که همگی با *Content-* شروع می شوند، به سرویس دهنده اجازه می دهند تا صفحه ای را

که در حال فرستادن آن است، توصیف کند.

سرآیند *Last-Modified* مشخص می‌کند که تاریخ آخرین تغییر صفحه چه زمانی بوده است. این سرآیند نقش مهمی در عملکرد حافظه نهان صفحات وب در مشتری دارد.

سرویس دهنده با استفاده از سرآیند *Location* به مشتری می‌گوید که باید به URL دیگری مراجعه کند (احتمالاً برای اینکه صفحه خواسته شده به جای دیگری منتقل شده، یا هدایت مشتری به سرویس دهنده‌های کم ترافیک‌تر). در مواردی هم که یک شرکت دارای شعبه‌های متعددی در سراسر دنیا است، با استفاده از این سرآیند مشتری را به سرویس دهنده‌ای که خاص منطقه وی در نظر گرفته شده، هدایت می‌کند. سرویس دهنده می‌تواند برای تشخیص مکان جغرافیایی مشتری از آدرس IP یا زبان درخواستی وی استفاده کند.

گاهی که یک صفحه خیلی بزرگ باشد، مشتریهای کوچک مایلند آنرا بصورت قطعه قطعه دریافت کنند. برخی از سرویس دهنده‌ها می‌توانند تعداد بایتهای درخواستی یک مشتری را پذیرفته، و فقط همان مقدار را برای وی بفرستند. سرویس دهنده با سرآیند *Accept-Range* آمادگی خود را برای ارسال جزئی صفحات اعلام می‌کند.

دومین سرآیند کوکی، یعنی *Set-Cookie*، وسیله‌ایست که سرویس دهنده به کمک آن کوکی‌ها را به مشتری می‌فرستد. مشتری ملزم است این کوکی‌ها را ذخیره کرده، و همراه با درخواستهای بعدی به سرویس دهنده پس بفرستد.

نمونه‌ای از کاربرد HTTP

از آنجائیکه HTTP یک پروتکل متنی است، هیچ نیازی نیست مرورگر باشید تا بتوانید با یک سرویس دهنده وب تماس بگیرید: فقط کافیست یک اتصال TCP به پورت 80 سرویس دهنده داشته باشید. اکیداً به خواننده توصیه می‌کنیم این قسمت را شخصاً امتحان کند (البته UNIX برای این منظور بهتر است، چون برخی از سیستمهای دیگر وضعیت اتصال را برنمی‌گردانند). برای شروع فرمان زیر را وارد کنید:

```
telnet www.ietf.org 80 >log
GET /rfc.html HTTP/1.1
Host: www.ietf.org
close
```

این فرمان یک اتصال TCP به پورت 80 سرویس دهنده وب IETF (www.ietf.org) برقرار می‌کند. نتیجه کار به فایل *log* هدایت می‌شود، تا بعداً بتوانیم آنرا بهتر بررسی کنیم. پس از آن فرمان *GET* (به همراه نام فایل و پروتکل) می‌آید، و خط بعدی سرآیند اجباری *Host* است. خط خالی بعدی نیز اجباریست: این خط خالی به سرویس دهنده می‌گوید که ارسال سرآیندها از طرف ما تمام شده است. با فرمان *close* هم به برنامه telnet می‌گوئیم که ارتباط را قطع کند.

فایل *log* یک فایل متنی است، که با هر ادیتوری می‌توان آنرا مشاهده کرد. ابتدای این فایل باید چیزی شبیه شکل ۷-۴۴ باشد (البته اگر IETF تازگیها آنرا عوض نکرده باشد).

سه خط اول خروجی برنامه telnet هستند، نه سرویس دهنده www.ietf.org. خط چهارم، که با HTTP/1.1 شروع شده، پاسخ IETF است و می‌گوید که این سرویس دهنده مایل است با پروتکل HTTP/1.1 با شما صحبت کند. پس از آن تعدادی سرآیند، و سپس محتویات صفحه www.ietf.org/rfc.html می‌آید. قبلاً همه این سرآیندها را دیده‌اید، بجز دو تا از آنها: سرآیند *ETag* (که یک شماره منحصر بفرد برای شناسایی صفحه، مخصوص حافظه نهان، است)، و *X-Pad* (که جزء سرآیندهای استاندارد نیست، و احتمالاً برای مقابله با مشکلات برخی از مرورگرها بکار می‌آید).

۵-۳-۷ بهبود کارایی

احتمالاً بزرگترین نقطه ضعف وب همان محبوبیت آن است. سرویس دهنده‌ها، مسیر یابها، و خطوط مخابراتی

```
Trying 4.17.168.6...
Connected to www.ietf.org.
Escape character is '^]'.
HTTP/1.1 200 OK
Date: Wed, 08 May 2002 22:54:22 GMT
Server: Apache/1.3.20 (Unix) mod_ssl/2.8.4 OpenSSL/0.9.5a
Last-Modified: Mon, 11 Sep 2000 13:56:29 GMT
ETag: "2a79d-c8b-39bce48d"
Accept-Ranges: bytes
Content-Length: 3211
Content-Type: text/html
X-Pad: avoid browser bug
```

```
<html>
<head>
<title>IETF RFC Page</title>

<script language="javascript">
function url() {
var x = document.form1.number.value
if (x.length == 1) {x = "000" + x }
if (x.length == 2) {x = "00" + x }
if (x.length == 3) {x = "0" + x }
document.form1.action = "/rfc/rfc" + x + ".txt"
document.form1.submit
}
</script>

</head>
```

شکل ۷-۴۴. قسمت ابتدایی خروجی www.ietf.org/rfc.html

امروزه با ترافیک شدیدی روبرو هستند، تا آنجا که خیلی ها WWW را World Wide Wait (انتظاری پایان جهانی) می خوانند. برای مقابله با این مشکل، محققان تکنیکهای مختلفی برای بالا بردن کارایی وب ابداع کرده اند. در این قسمت با برخی از این تکنیکها آشنا خواهید شد: حافظه نهان، تکثیر سرویس دهنده، و شبکه های تحویل محتوا.

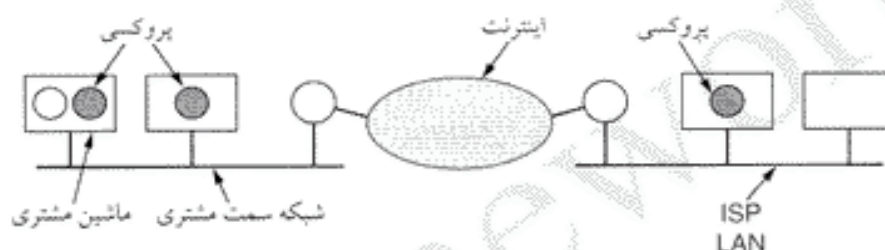
حافظه نهان

یکی از ساده ترین روشهای بهبود کارایی وب، ذخیره کردن صفحات برای مراجعه مجدد به آنهاست. این تکنیک بویژه برای صفحاتی که بازدیدکننده زیادی دارند (مانند www.yahoo.com یا www.cnn.com)، مناسب است. ذخیره کردن صفحات وب برای مراجعات بعدی به حافظه نهان (caching) معروف است. این حافظه نهان معمولاً از طریق یک واسطه، موسوم به پروکسی (proxy)، بکار گرفته می شود، و مرورگر بجای مراجعه مستقیم به سرویس دهنده، صفحه مورد نظر را از این پروکسی درخواست می کند. اگر پروکسی این صفحه را در حافظه نهان خود داشته باشد، بلافاصله آنرا به مرورگر برمی گرداند. ولی اگر نداشته باشد، صفحه را از سرویس دهنده گرفته، و (بعد از ذخیره کردن در حافظه نهان خود) به مشتری تحویل می دهد.

در رابطه با حافظه نهان دو سوال مهم وجود دارد:

۱. چه کسی باید این کار را انجام دهد؟
۲. صفحات چه مدت باید نگه داشته شوند؟

سوال اول جوابهای مختلفی می تواند داشته باشد. معمولاً PC ها هر کدام یک پروکسی دارند، که به کمک آن می توانند صفحاتی را که قبلاً دیده اند بسرعت پیدا کنند. در شبکه های محلی هم اغلب یک ماشین به این کار اختصاص داده می شود، تا اگر یکی از کاربران بخواهد صفحه ای را ببیند که کاربر دیگری قبلاً آنرا دیده است، نیازی به مراجعه مستقیم به سرویس دهنده نباشد و بتوان آنرا از حافظه نهان پروکسی در اختیار وی گذاشت. اکثر ISP ها هم برای سرعت دادن به دسترسی اینترنت کاربران خود از پروکسی استفاده می کنند. همه این پروکسی ها همزمان با هم کار می کنند، بنابراین درخواستها ابتدا به اولین پروکسی می رسد. اگر این پروکسی صفحه را نداشته باشد، سراغ پروکسی شبکه محلی می رود، و اگر آن هم صفحه را نداشته باشد، نوبت به پروکسی ISP می رسد. این آخری بالاخره هر طور که هست (از حافظه نهان خودش، از حافظه نهان سطح بالاتر، و یا مستقیماً از سرویس دهنده) صفحه را به درخواست کننده تحویل می دهد. به چنین روشی که در آن پروکسی ها بصورت زنجیره ای به هم متصلند، حافظه نهان سلسله مراتبی (hierarchical chacing) گفته می شود (شکل ۷-۴۵ را ببینید).



شکل ۷-۴۵. حافظه نهان سلسله مراتبی با سه پروکسی.

پاسخ سوال دوم کمی پیچیده تر است. برخی از صفحات اصلاً نباید در حافظه نهان ذخیره شوند. برای مثال، صفحه ای که قیمت سهام ۵۰ شرکت فعال بورس را نشان می دهد، هر ثانیه تغییر می کند. اگر چنین صفحه ای از حافظه نهان به مشتری داده شود، اطلاعات آن کهنه و غیر قابل اعتماد خواهد بود. از طرف دیگر، همین که بازار بورس تعطیل شود، اطلاعات این صفحه تا فردا (که بورس دوباره باز شود) معتبر خواهد بود. همانطور که می بینید، قابلیت ذخیره سازی یک صفحه در حافظه نهان می تواند بشدت به زمان وابسته باشد.

نکته کلیدی در تعیین مدت نگهداری صفحات در حافظه نهان این است که کاربران تا چه حد می توانند کهنگی صفحات را تحمل کنند (از آنجائیکه این صفحه ها روی دیسک ذخیره می شوند، مقدار آنها چندان اهمیتی ندارد). اگر یک پروکسی صفحات را مدت زیادی در حافظه نهان خود نگه ندارد (و آنها را بسرعت دور بیندازد)، صفحات آن بندرت کهنه می شوند، ولی از طرف دیگر کارایی خوبی نیز نخواهد داشت (چون صفحات کمی را از حافظه نهان به مشتری ها می دهد). اما اگر صفحات را مدت زیادی نگه دارد، اکثر درخواستها را از این حافظه نهان پاسخ می دهد، ولی به قیمت کهنه شدن آنها.

دو روش برای حل این مشکل وجود دارد. روش اول از نوعی شهود و گمان برای حدس زدن اینکه هر صفحه چه مدت باید در حافظه نهان نگه داشته شود، استفاده می کند. یکی از آنها بر پایه فیلد Last-Modified سرآیند صفحه استوار است (شکل ۷-۴۳ را ببینید). اگر صفحه ای یک ساعت پیش تغییر کرده باشد، پروکسی آنرا یک ساعت در حافظه نهان خود نگه می دارد. اگر صفحه ای آخرین بار یک سال پیش تغییر کرده باشد، پیداست که اطلاعات آن بسیار ثابت و پایدار است (مثلاً، صفحه ای که درباره تاریخ روم باستان است)، و می توان آنرا یک سال دیگر هم در حافظه نهان نگه داشت، بدون اینکه نگرانی زیادی درباره کهنه شدن اطلاعات آن وجود داشته باشد. با اینکه روش شهودی فوق غالباً در عمل خوب کار می کند، ولی گاهی هم صفحات کهنه برمی گرداند.

رہیافت دیگر (که بر اساس یکی از ویژگیهای مدیریت حافظه نهان در RFC 2616 بنا نهاده شده) قدری گرانتر است، ولی احتمال برگرداندن صفحات کهنه را از بین می برد. یکی از سودمندترین این ویژگیها سرآیند *If-Modified-Since* است، که پروکسی می تواند به سرویس دهنده بفرستد. در اینجا پروکسی نام صفحه موردنظر و تاریخ آخرین تغییر آنرا (از سرآیند *Last-Modified*) به سرویس دهنده می فرستد. اگر صفحه مزبور از این تاریخ به بعد تغییر نکرده باشد، سرویس دهنده پیام کوتاهی با مضمون *Not Modified* برمی گرداند (کد وضعیت 304، جدول ۷-۴۲)، که به پروکسی می گوید «استفاده از صفحه موجود در حافظه نهان بی اشکال است». اما اگر صفحه بعد از این تاریخ تغییر کرده باشد، سرویس دهنده صفحه جدید را برمی گرداند. همانطور که می بینید، در این روش پروکسی همیشه باید با سرویس دهنده تماس بگیرد، که البته در مواقعی که اطلاعات صفحه موجود در حافظه نهان همچنان معتبر باشد، پاسخ سرویس دهنده بسیار کوتاه خواهد بود.

این دو روش را می توان با سانی با هم ترکیب کرد. برای مدت زمان ΔT بعد از گرفتن صفحه، پروکسی صفحه را مستقیماً از حافظه نهان به درخواست کنندگان می دهد. اما پس از آنکه مدتی گذشت، پروکسی با استفاده از پیام *If-Modified-Since* اعتبار صفحه را چک می کند. انتخاب ΔT برای هر صفحه هم نیاز به نوعی شهود و گمان دارد، که باز هم به فیلد *Last-Modified* متکیست.

صفحات دینامیک (مانند آنهایی که اسکریپت PHP دارند) هرگز نباید در حافظه نهان ذخیره شوند، چون پارامترهای آنها هر بار تغییر می کند. برای حل این مشکل، سرویس دهنده با استفاده از یک مکانیزم خاص به تمام پروکسیهایی که بین آن و مشتری قرار دارند، دستور می دهد که بدون اطمینان از اعتبار و تازگی محتویات صفحه آنرا به مشتری ندهند. از این مکانیزم برای صفحاتی که محتویات آنها با سرعت تغییر می کند، نیز می توان استفاده کرد. در RFC 2616 مکانیزمهای مختلفی برای کنترل حافظه نهان تعریف شده است.

روش دیگری که برای بهبود کارایی وب وجود دارد، حافظه نهان پیشگو (proactive caching) است. در این روش، وقتی پروکسی صفحه ای را می آورد، تمام لینکهای موجود در آنرا بررسی کرده، و بطور خودکار آن صفحات را هم بار می کند، تا اگر به آنها نیاز شد، از قبل آمادگی داشته باشد. این تکنیک سرعت دسترسی به صفحات را بسیار بالا می برد، ولی ترافیک خطوط ارتباطی را هم بشدت افزایش خواهد داد (آن هم برای خواندن صفحاتی که شاید هرگز نیازی به آنها نباشد).

ذخیره کردن صفحات وب در حافظه نهان بهیچوجه موضوع ساده ای نیست، و می توان ساعتها درباره آن صحبت کرد. در این زمینه کتابهای متعددی نوشته شده، که از میان آنها می توان به (Rabinovich and Spatscheck, 2002; and Wessels, 2001) اشاره کرد. اما اجازه دهید ما این بحث را در همین جا خاتمه دهیم، و سراغ مبحث بعدی برویم.

تکثیر سرویس دهنده

حافظه نهان یک تکنیک بهبود کارایی سمت مشتری است، ولی تکنیکهای سمت سرویس دهنده نیز وجود دارند. یکی از متداولترین این تکنیکها تکثیر (replication) محتویات سرویس دهنده در نقاط دور از هم است. به این تکنیک گاهی آینه ای کردن (mirroring) نیز گفته می شود.

در ساده ترین شکل، صفحه اصلی یک سایت آینه ای دارای لینکهایی به نقاط مختلف (شمال، جنوب، شرق، و غرب) است. کاربری که روی یکی از این لینکها کلیک می کند، با توجه به ناحیه ای که در آن زندگی می کند، به سرویس دهنده مربوطه هدایت می شود. از آن پس نیز تمام درخواستهای وی به این سرویس دهنده می روند. سایتهای آینه ای معمولاً سایتهایی کاملاً ثابت و استاتیک هستند، و برای مدتهای مدید تغییر نمی کنند. معمولاً محتویات سایت اصلی کمابیش در سایت های آینه ای نیز کپی می شود (باستثنای آن بخشهایی که منطقی است حذف

شوند - برای مثال، مطالب مربوط به بخاری و وسایل گرم‌کننده در مناطق گرمسیر، یا مطالب مربوط به لباس شنا در مناطق قطبی، محلی از اعراب ندارد).

یکی از پدیده‌هایی که متأسفانه در وب بسیار دیده می‌شود، شهرت ناگهانی است: چه بسیار سایتهایی که مدتها ناشناخته، بی‌تماشاجی و راکد بوده‌اند، و ناگهان یک شبه تبدیل به مرکز توجه تمام دنیا شده‌اند. برای مثال، تا روز ششم نوامبر سال ۲۰۰۰ سایت دولت محلی فلوریدا (www.dos.state.fl.us) بزحمت روزی چند بازدیدکننده داشت. اما روز هفتم نوامبر که انتخابات ریاست جمهوری ایالات متحده بر سر چند هزار رأی حوزه‌های پرت و دورافتاده این ایالت به جنجال کشیده شد، این سایت تبدیل به یکی از پربیننده‌ترین سایتهای دنیا شد (و حتی برای مدتی بین پنج سایت رده اول قرار گرفت). لازم به گفتن نیست که این سایت تحمل چنین استقبالی را نداشت، و زیر بار این فشار سرعت از پا درآمد.

چیزی که یک سایت وب در این قبیل موارد لازم دارد، اینست که بتواند بطور خودکار خود را در محلهای مختلف تکثیر کرده، و تا پایان شرایط اضطراری آنها را فعال نگه دارد، و بعد از عبور طوفان آنها را خاموش کند. البته برای چنین کاری، یک سایت باید از قبل با شرکتهای میزبانی وب (Web hosting) هماهنگیهای لازم را انجام داده باشد.

یک استراتژی انعطاف‌پذیرتر ایجاد نسخه‌های تکثیری دینامیک برای تک تک صفحات (بر اساس تقاضاهای رسیده برای هر صفحه) است. در (Pierre et al., 2001; and Pierre et al., 2002) تحقیقاتی که در این زمینه انجام شده، گزارش شده است.

شبکه‌های تحویل محتوا

یکی از نقاط درخشان کاپیتالیسم این است که بالاخره یک نفر فهمید چگونه می‌توان از World Wide Wait (انتظار بی‌پایان برای گرفتن صفحات وب) پول در آورد. چگونه؟! شرکتهایی بنام CDN (شبکه تحویل محتوا - Content Delivery Network) به شرکتهای تولیدکننده محتوا (مانند سایتهای موسیقی، روزنامه‌ها، و دیگر جاهایی که مایلند محصولات خود را سرعت به بازار عرضه کنند) پیشنهاد کردند که محصولات آنها را سریعاً و در ازای مبلغی بعنوان حق اشتراک (یا حق مصرف) به دست مصرف‌کنندگان برسانند. بعد از آن که قرارداد بسته شد، صاحب کالا آنرا برای پردازش اولیه و توزیع در اختیار CDN می‌گذارد.

سپس، CDN با تعداد زیادی از ISP ها صحبت کرده، و با آنها (در ازای پرداخت پول) برای در اختیار گرفتن سرویس‌دهنده‌هایی که محتویات در آنها ذخیره خواهد شد، به توافق می‌رسد. این نه تنها منبع درآمدی برای CDN است، بلکه ISP را هم به نان و نوایی می‌رساند، چون مشتریان آن می‌توانند سرعت به مطالب دلخواهشان دسترسی پیدا کنند (و این در دنیای پر رقابت خدمات اینترنتی، یعنی موفقیت). بهمین دلیل ISP ها برای بستن قرارداد با CDN ها سر و دست می‌شکنند، و بعضی از این CDN ها متجاوز از ۱۰,۰۰۰ سرویس‌دهنده در سراسر دنیا دارند. وقتی محتویات روی هزاران سرویس‌دهنده پخش شده باشد، کیفیت دسترسی کاربران بشدت بالا خواهد رفت. اما برای آن که این سیستم توزیع شده بتواند بخوبی کار کند، باید مکانیزمی برای تغییر مسیر کاربران به نزدیکترین ISP (و ترجیحاً همان ISP که از آن سرویس می‌گیرند) وجود داشته باشد. این تغییر مسیر بایستی بدون هرگونه دستکاری در زیرساخت‌های اینترنت (از قبیل DNS ها) انجام شود. در زیر طرز کار آکامای (Akamai)، بزرگترین CDN دنیا، را بصورت ساده شده بررسی خواهیم کرد.

کار از آنجایی آغاز می‌شود که تولیدکننده محتوا سایت وب خود را تحویل CDN می‌دهد. در این مرحله، CDN یک پردازش اولیه روی تک تک صفحات سایت انجام داده، و تمام URL های آنرا عوض می‌کند. مدل کاری نهفته در پشت این استراتژی آنست که سایت وب تولیدکننده محتوا چند صفحه ساده HTML است، که

لینکهایی به فایل های بزرگتر (مانند تصویر، صدا و ویدئو) دارد. این صفحات تغییر یافته روی سرویس دهنده سایت تولیدکننده محتوا می مانند، و فقط فایل های تصویر، صدا و ویدئو است که به سرویس دهنده های CDN منتقل می شود. برای درک بهتر روش کار، صفحه اصلی سایت وب Furry Video (شکل ۷-۴۶ الف) را در نظر بگیرید. این صفحه بعد از پردازش اولیه توسط CDN به شکل ۷-۴۶ ب) در می آید، و با نام www.furryvideo.com/index.html در سرویس دهنده Furry Video ذخیره می شود.

```
<html>
<head> <title> Furry Video </title> </head>
<body>
<h1> Furry Video's Product List </h1>
<p> Click below for free samples. </p>
<a href="bears.mpg"> Bears Today </a> <br>
<a href="bunnies.mpg"> Funny Bunnies </a> <br>
<a href="mice.mpg"> Nice Mice </a> <br>
</body>
</html>
```

(الف)

```
<html>
<head> <title> Furry Video </title> </head>
<body>
<h1> Furry Video's Product List </h1>
<p> Click below for free samples. </p>
<a href="http://cdn-server.com/furryvideo/bears.mpg"> Bears Today </a> <br>
<a href="http://cdn-server.com/furryvideo/bunnies.mpg"> Funny Bunnies </a> <br>
<a href="http://cdn-server.com/furryvideo/mice.mpg"> Nice Mice </a> <br>
</body>
</html>
```

(ب)

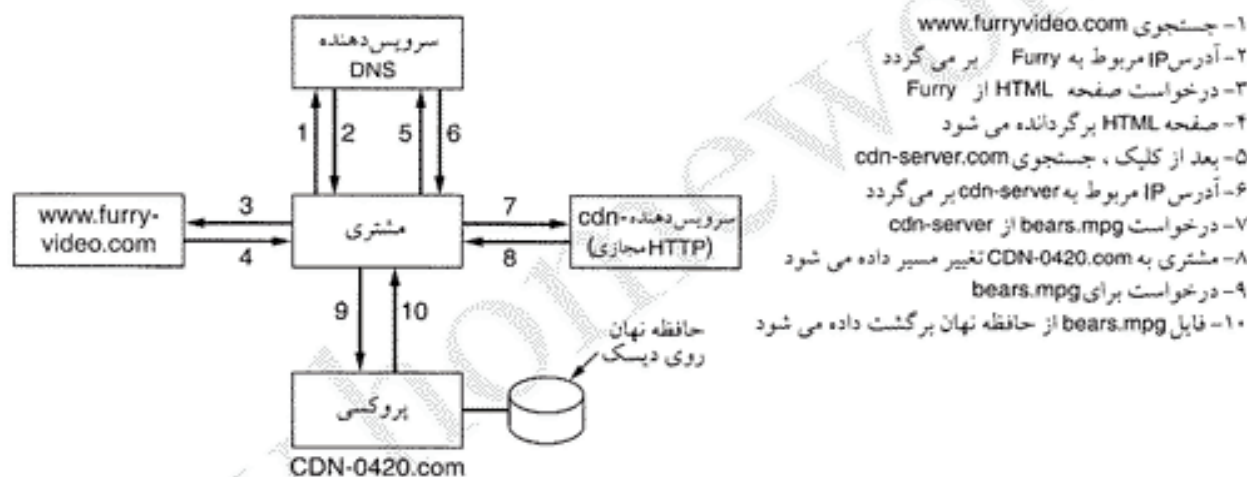
شکل ۷-۴۶. (الف) صفحه وب اولیه، (ب) همان صفحه پس از تغییر در CDN.

وقتی کاربر URL این سایت را وارد می کند، DNS طبق روال معمول آدرس IP سایت www.furryvideo.com را برمی گرداند، و مرورگر صفحه index.html را از سرویس دهنده Furry Video می خواند. اما وقتی روی هر یک از لینک های این صفحه کلیک شود، مرورگر آدرس سرویس دهنده cdn-server.com را از DNS می پرسد، فایل مورد نظر را از این سرویس دهنده درخواست می کند، و منتظر می ماند تا سرویس دهنده cdn-server.com این فایل را برگرداند.

اما این اتفاق نمی افتد، چون cdn-server.com چنین فایلی را ندارد. در اینجا CDN نقش یک سرویس دهنده HTTP تقلبی را بازی می کند، و با بررسی درخواست رسیده می فهمد که تقاضا مربوط به کدام صفحه از کدام تولیدکننده محتواست. همچنین با بررسی آدرس IP درخواست کننده (و جستجو در پایگاه اطلاعاتی خود) در می یابد که کاربر در کدام ناحیه جغرافیایی قرار دارد. با داشتن این اطلاعات، CDN می تواند تصمیم بگیرد که کدام سرویس دهنده محتوا مناسبترین گزینه برای کاربر مورد نظر است. این تصمیم گیری چندان هم که بنظر می آید ساده نیست، چون ممکنست نزدیکترین محل از نظر جغرافیایی نزدیکترین محل از نظر توپولوژی شبکه نباشد، و یا نزدیکترین سرویس دهنده از نظر توپولوژی شبکه در آن لحظه ترافیک بالایی داشته باشد. بعد از انتخاب

سرویس دهنده مناسب، CDN یک پیام با کد 301 که URL نزدیکترین محل در فیلد Location آن مشخص شده، به مشتری برمیگرداند. برای این مثال فرض می‌کنیم که URL نزدیکترین محل به مشتری www.CDN-0420.com/furryvideo/bears.mpg است. مرورگر پس از دریافت این URL به سراغ آن رفته، و فایل bears.mpg را طبق روال معمول می‌خواند.

مراحل کار در شکل ۷-۴۷ نشان داده شده است. اولین مرحله تعیین آدرس IP سایت www.furryvideo.com، و پس از آن آوردن صفحه index.html و نمایش آن است. این صفحه سه لینک به cdn-server.com دارد (شکل ۷-۴۶ ب). وقتی (مثلاً) لینک اول انتخاب شود، DNS آدرس آنرا جستجو کرده (مرحله ۵) و برمیگرداند (مرحله ۶). با ارسال درخواست فایل bears.mpg به cdn-server.com (مرحله ۷)، به مشتری گفته می‌شود که باید به CDN-0420.com برود (مرحله ۸). وقتی مشتری به این محل مراجعه می‌کند (مرحله ۹)، پروکسی CDN-0420.com فایل مزبور را به وی تحویل می‌دهد (مرحله ۱۰). آنچه که باعث می‌شود این مکانیزم کار کند، مرحله ۸ است: جایی که یک سرویس دهنده HTTP قلابی مشتری را به نزدیکترین پروکسی CDN تغییر مسیر می‌دهد.



شکل ۷-۴۷. مراحل پیدا کردن URL وقتی پای CDN در میان است.

سرویس دهنده CDN (که مشتری از آنجا سر در می‌آورد) معمولاً یک پروکسی با حافظه نهان بسیار بزرگ است (که قسمت اعظم محتویات در آن قرار دارند). با این تمهید (استفاده از پروکسی بجای یک سرویس دهنده وب معمولی) کارایی سیستم بنحو قابل توجهی بالا می‌رود. برای کسب اطلاعات بیشتر درباره شبکه‌های تحویل محتوا به (Hull, 2002; and Rabinovich and Spatscheck, 2002) مراجعه کنید.

۶-۳-۷ وب بیسیم

این روزها تقاضای زیادی برای دستگاههای کوچک بیسیم که توانایی کار با وب را داشته باشند، وجود دارد. در حقیقت، اولین قدمهای تجربی در این زمینه قبلاً برداشته شده است. شکی نیست که در سالهای آینده تغییرات زیادی را در این زمینه شاهد خواهیم بود، ولی جا دارد که ایده‌های اساسی وب بیسیم را بررسی کنیم، تا دریابیم کجا هستیم و به کجا می‌رویم. در این قسمت بررسی خود را روی دو سیستمی که زودتر از همه به بازار آمدند، متمرکز خواهیم کرد: WAP و I-Mode.

WAP

با رواج روزافزون اینترنت و تلفن همراه، دور نبود روزی که ایده ترکیب این دو تکنولوژی مطرح شود. ایده چنین سیستمی اولین بار توسط کنسرسیومی از شرکتهای نوکیا، اریکسون، موتورولا، و Unwired.com

Planet سابق) پیش کشیده شد، و اکنون صدها شرکت دیگر آنرا پشتیبانی می کنند. این سیستم اکنون با نام WAP (پروتکل کاربردهای بیسیم - Wireless Application Protocol) شناخته می شود.

دستگاه WAP می تواند یک تلفن همراه پیشرفته، یک PDA، و یا یک کامپیوتر سفری ساده باشد (استاندارد WAP محدودیتی برای نوع دستگاه قائل نشده است). WAP در واقع به زیرساختهای بیسیم دیجیتال موجود متکی است. کاربر می تواند از طریق لینکهای بیسیم به دروازه WAP (WAP Gateway) وصل شده، و درخواست خود برای صفحات وب را ارسال کند. اولین جایی که برای این درخواست چک می شود، حافظه نهان دروازه است: اگر صفحه در حافظه نهان دروازه موجود باشد، به کاربر برگردانده می شود؛ اگر نباشد، از اینترنت (که ارتباط دروازه با آن از طریق کابل یا فیبر است) گرفته شده، و به کاربر داده می شود. به این ترتیب، WAP 1.0 اساساً یک سیستم سونچینگ مداری بود که هزینه آن بر اساس زمان مکالمه دریافت می شد؛ البته مصرف کنندگان هم از چنین چیزی (آن هم برای دیدن صفحات وب روی ماینیتر کوچک تلفنهای همراه) خوششان نیامد، و WAP 1.0 شکست خورد (که البته این شکست علت های دیگری هم داشت). با این حال، ایده WAP (و رقیب آن، I-Mode) هنوز شکست نخورده، و بنظر می رسد WAP 2.0 بتواند با موفقیت همراه باشد. از آنجائیکه WAP 1.0 اولین تلاش برای پیاده سازی اینترنت بیسیم بود، نگاه مختصری به آن نمی تواند خالی از فایده باشد.

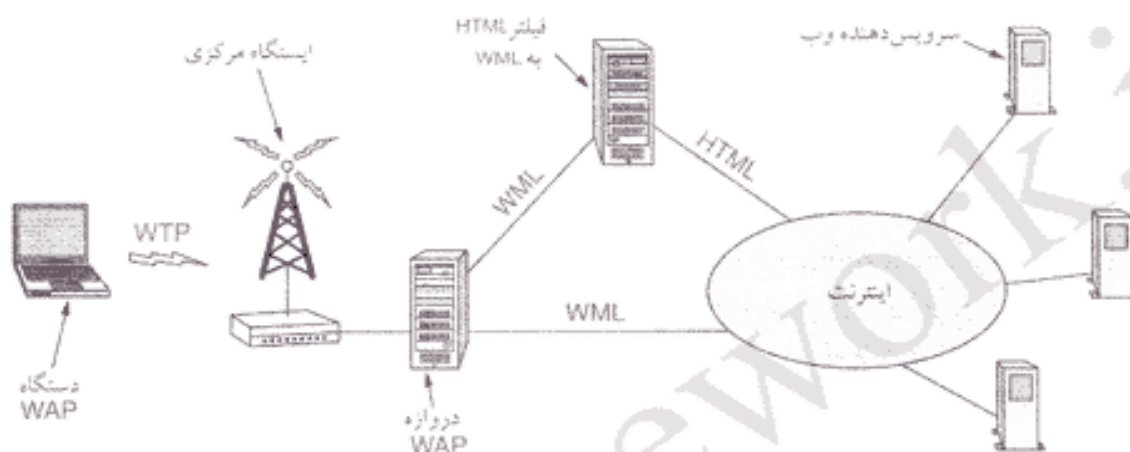
WAP اساساً یک پشته پروتکل است برای دسترسی وب، که برای وسایلی با پهنای باند کم، CPU کند، مقدار حافظه کم، و ماینیتر کوچک بهینه شده است. این رهیافت که آشکارا با استاندارد PC های امروزی متفاوت است، منجر به پروتکلی متفاوت نیز شده است. لایه های پروتکل WAP را در شکل ۷-۴۸ ملاحظه می کنید.

محیط برنامه های کاربردی بیسیم
پروتکل نشست بیسیم
پروتکل تبادل بیسیم
ایمنی لایه انتقال بیسیم
پروتکل دیتاگرام بیسیم
لایه منتقل کننده (GSM, CDMA, D-AMPS, GPRS, etc.)

شکل ۷-۴۸. پشته پروتکل WAP.

پاینترین لایه (و در واقع لایه فیزیکی) از تمام سیستمهای تلفن همراه موجود (از جمله GSM، D-AMPS، و CDMA) پشتیبانی می کند. در WAP 1.0 نرخ داده 9600 bps است. بالای این لایه پروتکل دیتاگرام، WDP (پروتکل دیتاگرام بیسیم - Wireless Datagram Protocol)، که اساساً همان UDP است، قرار دارد. پس از آن لایه ای برای ایمنی داده ها (که در مخابرات بیسیم الزامیست) می آید. این لایه، WTLS (ایمنی لایه انتقال بیسیم - Wireless Transport Layer Security)، زیرمجموعه ایست از SSL (که از ابداعات نت اسکپ می باشد). سپس لایه تبادل، WTP (پروتکل تبادل بیسیم - Wireless Transaction Protocol)، می آید که وظیفه آن مدیریت درخواستها و پاسخهاست. این لایه جایگزین TCP شده، که بدلیل کارایی پائین نمی توان از آن در ارتباطات بیسیم استفاده کرد. پس از آن یک لایه نشست می آید، که شبیه HTTP/1.1 است، ولی بمنظور کارایی بهتر تغییراتی در آن صورت گرفته است. و بالاخره، در بالای همه اینها یک مینی مرورگر - در لایه WAE (محیط کاربردی بیسیم - Wireless Application Environment) - قرار می گیرد.

بدون شک یکی از علل عدم پذیرش WAP (علاوه بر هزینه بالا) این واقعیت است که WAP از HTML استفاده نمی‌کند: WAE از یک زبان علامتگذاری خاص بنام WML (زبان علامتگذاری بیسیم - Wireless Markup Language) استفاده می‌کند، که به XML متکی است. در نتیجه، یک دستگاه WAP فقط صفحاتی را می‌تواند بخواند که قبلاً به WML تبدیل شده باشند. از آنجائیکه این ویژگی کاربرد WAP را بشدت محدود خواهد کرد، در معماری WAP فیلترهایی برای تبدیل آنی HTML به WML در نظر گرفته شده است (شکل ۴۹-۷ را ببینید).



شکل ۴۹-۷. معماری WAP.

WAP، با وجود تمام شایستگی‌هایش، شاید کمی از زمانه جلوتر بود. وقتی WAP برای اولین بار راه‌اندازی شد، خارج از W3C کمتر کسی XML را می‌شناخت، و بهمین دلیل همه جا فریاد زدند: «WAP از HTML پشتیبانی نمی‌کند». شاید بهتر بود می‌گفتند: «WAP از استاندارد جدید HTML پشتیبانی می‌کند». ولی وقتی ضربه وارد آمد، دیگر برای جبران آن دیر شده بود، و WAP 1.0 هرگز نتوانست دوباره کمر راست کند. قبل از اینکه داستان WAP را ادامه دهیم، نگاهی به نزدیکترین رقیب آن، یعنی I-Mode، خواهیم داشت.

I-Mode

در همان زمانیکه کنسرسیوم چندملیتی شرکتهای کامپیوتری و مخابراتی در تلاش بودند تا یک استاندارد باز برای HTML توسعه دهند، ژاپنی‌ها بیکار نشسته بودند. در آنجا خانمی بنام ماری ماتسونوگا رهیافت جدیدی برای وب بیسیم اختراع کرد، و نام آنرا I-Mode (Information-Mode) گذاشت. او توانست شرکت مخابرات بیسیم ژاپن (که زیرمجموعه وزارت تلفن ژاپن محسوب می‌شد) را متقاعد کند که اختراعش مفید است، و در فوریه ۱۹۹۹ NTT DoCoMo (که به زبان ژاپنی معادل عبارت «شرکت تلفن و تلگراف ژاپن: هر جاکه هستید» است) رسماً شروع بکار کرد. بعد از سه سال، این سرویس اکنون متجاوز از ۲۵ میلیون مشترک دارد، که به ۴۰,۰۰۰ سایت وب I-Mode دسترسی دارند. مدیران این سیستم (در غیاب WAP) بیشترین لاف‌ها را درباره موفقیت مالی آن زده‌اند. اما اجازه دهید ببینیم I-Mode چیست و چگونه کار می‌کند.

سیستم I-Mode دارای سه مؤلفه اصلی است: یک سیستم انتقال جدید، یک گوشی جدید، و یک زبان جدید برای طراحی صفحات وب. سیستم انتقال از دو شبکه مجزا تشکیل شده است: شبکه تلفن همراه سونیچینگ بسته‌ای موجود (که تا حدی شبیه D-AMPS است)، و یک شبکه سونیچینگ مداری که اختصاصاً برای سرویس I-Mode ایجاد شده است. سرویس I-Mode از شبکه سونیچینگ بسته‌ای استفاده می‌کند و ارتباط آن دائمی

است (مانند ADSL یا کابل). بنابراین چیزی بنام هزینه اتصال در آن وجود ندارد، و بجای آن هزینه مشترک بر اساس بسته های فرستاده شده محاسبه می شود. در حال حاضر امکان استفاده همزمان از هر دو شبکه وجود ندارد. گوشی I-Mode شبیه گوشی های معمولی تلفن همراه است، با یک صفحه مانیتور بزرگتر. حتی NTT DoCoMo در تبلیغات خود دستگاه های I-Mode را نسل جدید و پیشرفته تلفن های همراه معرفی می کند، نه ترمینال های بیسیم وب (چیزی که در واقع هستند). بسیاری از مشترکان این سیستم حتی نمی دانند که روی اینترنت هستند. اغلب آنها تصور می کنند که دستگاه I-Mode فقط یک تلفن همراه پیشرفته است. از آنجائیکه I-Mode فقط یک سرویس است، کاربران امکان برنامه نویسی با گوشی خود را ندارند (با اینکه گوشی آنها از کامپیوترهای سال ۹۵ قویتر است، و احتمالاً حتی می تواند ویندوز ۹۵ یا یونیکس را اجرا کند).

وقتی یک گوشی I-Mode روشن می شود، فهرستی از سرویس های رسمی و تأیید شده به کاربر ارائه می کند. تعداد این سرویس ها بیش از ۱۰۰۰ تا است، که به ۲۰ دسته تقسیم شده اند. هر سرویس که در واقع یک سایت I-Mode کوچک است، توسط یک شرکت مستقل اداره می شود. برخی از مهمترین دسته هایی که در این منو ظاهر می شوند، عبارتند از: ایمیل، اخبار، وضع آب و هوا، ورزش، بازی، خرید، نقشه، طالع بینی، سرگرمی، مسافرت، راهنمای اعمال مذهبی، انواع زنگ های تلفن، دستورات آشپزی، قمار، بانکداری خانگی، و قیمت های بورس. این سرویس ها تا حد زیادی نوجوانان و جوانان (با سنی حدود ۲۰ سال) را هدف گرفته اند، افرادی که عاشق اسباب بازی های الکترونیکی (مخصوصاً اسباب بازی های رنگارنگ) هستند. این موضوع که بیش از ۴۰ شرکت فقط زنگ تلفن می فروشند، می تواند گویای حقایق زیادی باشد. در اینجا هم محبوبترین سرویس ایمیل است، که اجازه می دهد پیام هایی تا ۵۰۰ بایت رد و بدل شود (نسبت به سرویس SMS با محدودیت ۱۶۰ بایتی، پیشرفت چشمگیری محسوب می شود). بازی نیز یکی دیگر از سرویس های پرطرفدار I-Mode است.

بیش از ۴۰,۰۰۰ سایت وب I-Mode نیز وجود دارد، که کاربر باید URL آنها را وارد کند (و از طریق منو قابل دسترسی نیستند). منوی رسمی I-Mode بسیار شبیه دروازه های وب (مانند سایت Yahoo!) است، که به کاربر اجازه می دهند تا بدون وارد کردن URL و فقط با یک کلیک به سایت های مختلف دسترسی پیدا کند.

سرویس های رسمی I-Mode تحت کنترل شدید NTT DoCoMo قرار دارند. برای آن که یک سرویس در منوی رسمی I-Mode قرار گیرد، باید شرایط مختلفی را برآورده کند. برای مثال، یک سرویس نباید تأثیرات منفی اجتماعی داشته باشد، فرهنگ های لغت ژاپنی-انگلیسی باید به مقدار کافی لغت داشته باشند، سرویس های زنگ تلفن باید بطور مرتب زنگ های جدیدی عرضه کنند، و هیچ سایتی نباید مطالب بیهوده و سبک (یا چیزی علیه NTT DoCoMo) ارائه کند (Frengle, 2002). از طرف دیگر، سایت های اینترنتی I-Mode مجازند هر کاری می خواهند بکنند.

مدل تجاری I-Mode تفاوت اساسی با اینترنت دارد. هزینه اشتراک I-Mode فقط چند دلار در ماه است. از آنجائیکه در این سیستم هزینه ها بر اساس بسته های ارسال شده محاسبه می شود، در واقع با این شارژ اولیه کار چندانی نمی تواند کرد. کاربر می تواند هزینه ماهیانه ثابت بیشتری (برای تعداد بسته بیشتر) بپردازد، و در شارژ بسته ها صرفه جویی کند، چون با بالا رفتن پرداخت ثابت ماهیانه تعداد بسته هایی که می تواند بفرستد، بصورت تصاعدی بالا خواهد رفت. اگر وسط ماه بسته های مجانی شما تمام شود، می تواند بصورت بر-خط بسته های بیشتری بخرد.

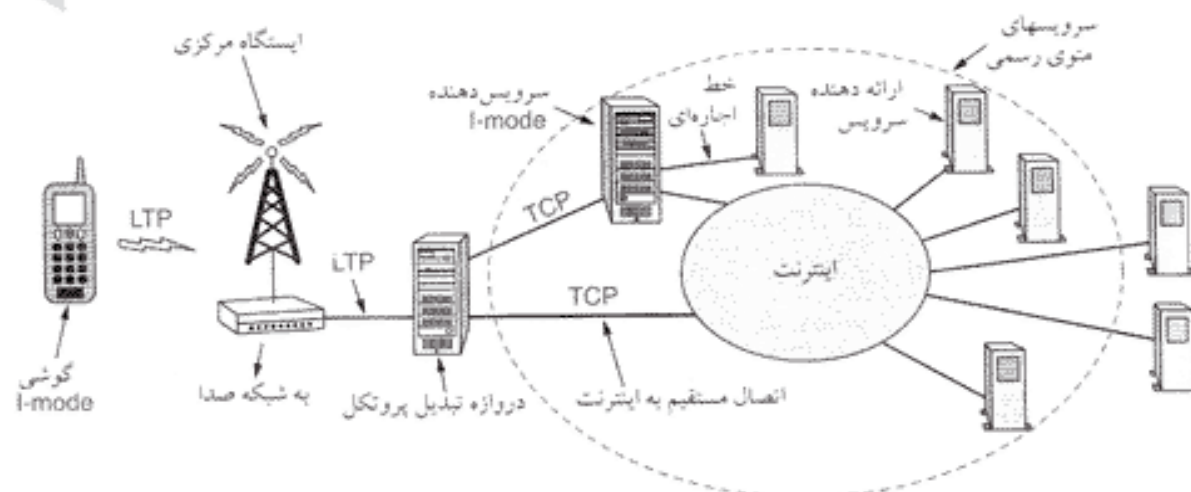
برای استفاده از یک سرویس باید مشترک آن شوید، کاری که فقط با یک کلیک روی آن و وارد کردن کد PIN خود می توانید انجام دهید. اغلب سرویس های رسمی هزینه ای معادل ۱ تا ۲ دلار در ماه دارند. NTT DoCoMo این پول را در صورت حساب تلفن شما منظور می کند، و بعد از کسر ۹٪ سهم خود بقیه (۹۱٪) را به شرکت ارائه کننده سرویس مسترد می کند. اگر یک سرویس غیررسمی ۱,۰۰۰,۰۰۰ مشترک داشته باشد، مجبور است هر ماه ۱,۰۰۰,۰۰۰ صورت حساب ۱ تا ۲ دلاری صادر کرده و برای مشتریان خود بفرستد. در حالیکه اگر سرویس خود را

بصورت رسمی در آورد، NTT DoCoMo عملیات بانکی را بجای آن انجام داده و بازای آن ۹۰,۰۰۰ دلار از کل مبلغ کم می کند. صرفه جویی در هزینه های بانکی وصول حق اشتراک از مشتریان انگیزه ای بسیار جدی برای تمایل شرکتها به رسمی شدن است (و NTT DoCoMo هم که به حق خود می رسد). رسمی شدن احتمال جذب مشتری را هم افزایش می دهد، چون سایت شما وارد منوی اولیه I-Mode خواهد شد. این وسط فقط کاربران هستند که بجای یک صورتحساب باید برای چهار چیز پول بدهند: پول تماسهای تلفنی معمولی، حق اشتراک I-Mode، حق اشتراک سرویسها، و هزینه بسته های اضافی.

علیرغم اقبال گسترده در ژاپن، هنوز مشخص نیست که I-Mode بتواند در اروپا و آمریکا نیز به چنان موفقیتی دست یابد، چون وضعیت اجتماعی ژاپن تفاوت زیادی با جوامع غربی دارد. اول از همه اینکه، اکثر مشتریان بالقوه این سیستم در غرب (نوجوانان، دانشجویان، و کارمندان) دارای PC های بزرگ در خانه با دسترسی اینترنت پرسرعت (حداقل 56 kbps) هستند، چیزی که در ژاپن چندان مرسوم نیست (اول بعلت فضای کوچک خانه های ژاپنی، و دوم بدلیل نرخهای کمرشکن NTT برای سرویسهای تلفن - ۷۰۰ دلار برای نصب یک خط تلفن، و ساعتی ۱/۵ دلار هزینه تماسهای شهری). در ژاپن خیلی از افراد فقط از طریق I-Mode به اینترنت دسترسی دارند. دوم اینکه، در غرب کسی عادت ندارد برای هر سایتی که می خواهد برود، ۱ دلار حق اشتراک بدهد (پرداخت پول برای هر MB خواندن صفحات که دیگر بماند!). اغلب ISP های غربی (تحت فشار مشتریان خود) هزینه دسترسی اینترنت را به یک شارژ ثابت ماهیانه (مستقل از مصرف واقعی آنان) تقلیل داده اند.

سوم اینکه، اغلب ژاپنی ها در زمان رفت و آمد به محل کار یا مدرسه (و در قطار) از I-Mode استفاده می کنند، در حالیکه در اروپا افراد کمی برای رفت و آمد از قطار استفاده می کنند (و این پدیده در آمریکا حتی از اروپا هم نادرتر است). استفاده از I-Mode در خانه، آن هم کنار یک ماینیو ۱۷ اینچی، با یک خط 1-Mbps ADSL (بدون اینکه نیازی باشد نگران حجم اطلاعات رد و بدل شده باشید) چندان با عقل سلیم جور در نمی آید. با این حال، هیچکس چنین محبوبیتی را برای تلفنهای همراه هم پیش بینی نمی کرد، پس شاید در غرب هم جایی برای I-Mode باشد.

همانطور که قبلاً هم گفتیم، گوشی های I-Mode برای ارتباطات صدا از شبکه موجود سوئیچینگ مداری، و برای ارتباطات داده از یک شبکه سوئیچینگ بسته ای جدید استفاده می کنند. شبکه داده بر اساس CDMA، با بسته های ۱۲۸ بیتی و با نرخ 9600 bps کار می کند (شکل ۷-۵ را ببینید). گوشی با استفاده از LPT (پروتکل سبک انتقال - Lightweight Transport Protocol) با دروازه تبدیل پروتکل (protocol conversion gateway)



شکل ۷-۵. ساختار شبکه داده I-Mode: پروتکل های انتقال.

ارتباط می گیرد. این دروازه توسط یک فیبر نوری پهن باند به سرویس دهنده I-Mode (که به تمام سرویسها متصل است) وصل می شود. وقتی کاربر یکی از سرویسهای رسمی را انتخاب می کند، این درخواست به سرویس دهنده I-Mode فرستاده می شود، که (برای بهبود کارایی شبکه) اغلب صفحه ها را در حافظه نهان خود دارد. درخواست برای سایتی که جزء منوی رسمی نیستند، مستقیماً به اینترنت فرستاده می شود.

گوشی های فعلی I-Mode با 100-MHz CPU کار می کنند، و چندین مگابایت حافظه ROM، چیزی حدود یک مگابایت RAM و یک صفحه مانیتور کوچک دارند. وضوح مانیتور I-Mode بایستی حداقل 72×94 پیکسل باشد، ولی دستگاههایی با وضوح 120×160 پیکسل نیز عرضه شده اند. این مانیتورها از رنگ 8-bit پشتیبانی می کنند، که قادرست 256 رنگ مختلف را نمایش دهد. با اینکه این تعداد رنگ برای نمایش عکس کافی نیست، اما طرح و تصاویر کارتونی را بخوبی نشان می دهد. گوشی های I-Mode (طبعاً) ماوس ندارند، و حرکت بین آیتمهای صفحه به کمک کلیدها صورت می گیرد.

ساختار نرم افزاری I-Mode را در شکل ۷-۵۱ ملاحظه می کنید. در پائین ترین لایه یک سیستم عامل بی درنگ (real-time) قرار دارد، که سخت افزار را کنترل می کند. پس از آن ماژول ارتباط با شبکه می آید، که از پروتکل LPT اختصاصی NTT DoCoMo استفاده می کند. بالای این لایه یک سیستم مدیریت پنجره (window manager) قرار دارد، که متن و گرافیک های ساده (فایل های GIF) را نمایش می دهد. البته با مانیتوری به ابعاد حداکثر 120×160 پیکسل چیز زیادی برای مدیریت کردن وجود ندارد.

User interaction module (ماژول تعامل با کاربر)		
Plug-ins	cHTML interpreter	Java
Simple window manager (مدیر پنجره ساده)		
Network communication (ارتباطات شبکه)		
Real-time operating system (سیستم عامل زمان واقعی)		

شکل ۷-۵۱. ساختار نرم افزاری I-Mode.

لایه چهارم شامل مفسر صفحات وب (یعنی همان مرورگر) است؛ I-Mode فقط از زیرمجموعه ای از HTML بنام cHTML (HTML فشرده - compact HTML) که بر اساس HTML 1.0 قرار دارد پشتیبانی می کند. برنامه های کمکی و افزودنی (مانند مرورگرهای معمولی) نیز در این لایه قرار می گیرند. یکی از برنامه های کمکی I-Mode مفسری بر اساس ویرایش اصلاح شده JVM است. و بالاخره، در بالای همه لایه ها ماژول تعامل با کاربر (user interaction) قرار گرفته است، که وظیفه آن ارتباط با کاربر از طریق صفحه کلید و مانیتور می باشد. اجازه دهید نگاه دقیقتری به cHTML بیندازیم. همانطور که گفتیم، cHTML تقریباً همان HTML 1.0 است، که برای انطباق با گوشی های تلفن همراه تغییراتی در آن صورت گرفته است. استاندارد cHTML از طرف W3C برای نظرخواهی ارائه شده، ولی از آنجائیکه W3C علاقه چندانی به آن نشان نداده، به احتمال زیاد همچنان بصورت محصول اختصاصی NTT DoCoMo باقی خواهد ماند.

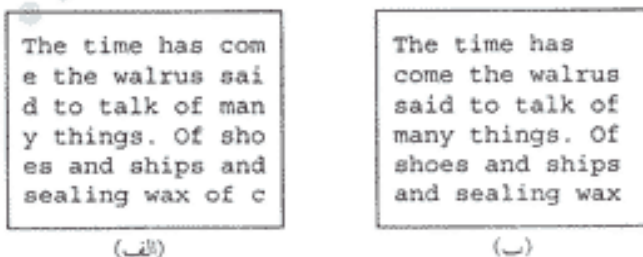
اکثر برچسبهای HTML مانند <html>، <head>، <title>، <body>، <hr>، <center>، ، ، <menu>، ،
، <p>، ، <form>، و <input> در cHTML هم وجود دارند، ولی و <i> حذف شده اند.

یدی بنام tel به برچسب <a> (لینک به صفحات دیگر) اضافه شده، که با آن می توان با یک شماره تلفن تماس گرفت. از یک جهت tel شبیه mailto است، که وقتی کاربر آنرا انتخاب می کند، مرورگر با اجرای برنامه ایمیل به کاربر اجازه می دهد به آدرس مشخص شده پیام بفرستد؛ با این تفاوت که در اینجا مرورگر شماره تلفن را می گیرد. بعنوان مثال، با این صفت می توان یک دفترچه تلفن تصویری ساخت، که وقتی روی هر یک از تصاویر کلیک می کنید، گوشی I-Mode شماره مربوطه را بگیرد. (URL های تلفنی در RFC 2806 مورد بحث قرار گرفته اند).

مرورگر cHTML محدودیتهای دیگری نیز دارد، مثلاً از جاوا اسکریپت، فریم، شیوه نامه، و رنگ یا تصویر زمینه پشتیبانی نمی کند. مرورگرهای I-Mode تصاویر JPEG را هم نمایش نمی دهند، چون قدرت کافی برای خارج کردن سریع این تصاویر از حالت فشرده ندارند. صفحات cHTML می توانند اپلت جاوا داشته باشند، ولی اندازه آنها (بدلیل کم بودن سرعت انتقال روی لینکهای هوایی) به 10 KB محدود است.

با اینکه NTT DoCoMo برخی از برچسبهای HTML را حذف کرده، ولی برچسبهای جدیدی را هم به آن اضافه کرده است. یکی از این برچسبها <blink> است، که متن را بصورت چشمک زن درمی آورد. شاید این برچسب جانشین (که در cHTML حذف شده) باشد، ولی بهر حال این اقدام NTT DoCoMo با استاندارد HTML که به ظاهر صفحات وب بی توجه است، همخوانی چندانی ندارد. برچسب جدید دیگر <marquee> است، که متن را در عرض صفحه حرکت می دهد.

به برچسب
 نیز صفت جدیدی بنام align اضافه شده، که برای نمایش مرتب کلمات روی مانیتورهای ۶ سطر x ۱۶ حرف در نظر گرفته شده است. در چنین مانیتورهایی احتمال زیادی هست که کلمات از وسط شکسته شوند (شکل ۷-۵۲ الف را ببینید). صفت align کمک می کند تا این مشکل تا حد زیادی رفع شود (مانند آنچه در شکل ۷-۵۲ ب می بینید). البته جالبست بدانید که در زبان ژاپنی چیزی بنام شکستن کلمات وجود ندارد، چون هر علامت کانجی (kanji - الفبای ژاپنی) - که در یک سلول ۱۰ x ۹ پیکسل یا ۱۲ x ۱۲ پیکسل نوشته می شود - معادل یک کلمه در زبان انگلیسی است. در زبان ژاپنی یک خط متن می تواند از هر نقطه ای شکسته شود.



شکل ۷-۵۲. آشفته گی در صفحه مانیتور ۶ x ۱۶.

صفت جد با اینکه زبان ژاپنی دهها هزار کانجی دارد، NTT DoCoMo زبان تصویری جدیدی بنام اموجی (emoji) با ۱۶۶ علامت اختراع کرده است، که تا حدی شبیه خندانک های شکل ۷-۶ هستند. در این زبان علامتهای زیبایی برای نمادهای طالع بینی، آبجو، همبرگر، پارک تفریحات، روز تولد، تلفن همراه، سگ، گربه، کریسمس، قلب شکسته، بوسه، خلق و خو، خواب، و البته زیبا و دلغریب وجود دارد.

ویژگی دیگر cHTML امکان انتخاب لینکهای صفحه وب با استفاده از صفحه کلید است (که البته در جایی که ماوس وجود ندارد، ویژگی بسیار مهمی است). در شکل ۷-۵۳ نمونه ای از یک صفحه cHTML که در آن از این ویژگی استفاده شده، را مشاهده می کنید.

با اینکه I-Mode در سمت مشتری با محدودیتهای زیادی مواجه است، در سمت سرویس دهنده هیچ

```

<html>
<body>
<h1> Select an option </h1>
<a href="messages.shtml" accesskey="1"> Check voicemail </a> <br>
<a href="mail.shtml" accesskey="2"> Check e-mail </a> <br>
<a href="games.shtml" accesskey="3"> Play a game </a>
</body>
</html>

```

شکل ۷-۵۳. نمونه ای از یک فایل cHTML.

محدودیتی ندارد. سرویس دهنده های I-Mode کامپیوترهای قدرتمندی هستند که از CGI، PHP، JSP، ASP (و هر چیزی که یک سرویس دهنده وب معمولی می تواند داشته باشد) پشتیبانی می کنند.

در شکل ۷-۵۴ مقایسه ای بین سیستمهای نسل اول WAP و I-Mode بعمل آمده است. با آنکه برخی از این تفاوتها کوچک بنظر می رسند، ولی در جای خود اهمیت زیادی دارند. برای مثال، اغلب نوجوانان ۱۵ ساله کارت اعتباری ندارند، بنابراین امکان واریز هزینه های خرید الکترونیکی به صورت حساب ماهیانه تلفن می تواند عامل تعیین کننده ای در جلب این قبیل مشتریان داشته باشد.

برای کسب اطلاعات بیشتر درباره I-Mode می توانید به (Frengle, 2002; and Vacca, 2002) مراجعه کنید.

ویژگی	WAP	I-mode
ماهیت	پشته پونکل	سرویس
دستگاه	گوشی، PDA، کامپیوتری	گوشی
دسترسی	تلفنی	همیشه برقرار
شبکه زیرین	سوئیچینگ مداری	سوئیچینگ مداری و بسته ای
نرخ داده	9600 bps	9600 bps
صفحه نمایش	سیاه و سفید	رنگی
زبان علامتگذاری	WML (XML application)	cHTML
زبان اسکریپت نویسی	WML script	ندارد
هزینه	بر دقیقه	پربسته
پرداخت خریده ها	کارت اعتباری	صورتحساب تلفن
علامت تصویری	خیر	بلی
استاندارد سازی	استاندارد باز مجمع WAP	NTT DoCoMo
محل استفاده	اروپا، ژاپن	ژاپن
کاربران نوعی	تاجران و صنعتگران	افراد جوان

شکل ۷-۵۴. مقایسه ای بین سیستمهای نسل اول WAP و I-Mode.

نسل دوم وب بیسیم

WAP 1.0، که بر اساس استانداردهای پذیرفته شده بین المللی طراحی شده بود، ابزاری جدی برای افراد پرتحرک محسوب می شد. اما، متأسفانه شکست خورد. در مقابل، I-Mode بازبچه ای بود برای نوجوانان ژاپنی، که هیچ استاندارد هم پشت آن نبود. در میان تعجب همگان، I-Mode یک موفقیت تجاری بزرگ بود. بعد چه شد؟ نسل اول وب بیسیم درسهایی برای همه داشت. کنسرسیوم WAP فهمید که محتوا از همه چیز مهمتر است. اگر تعداد زیادی سایت وب که به زبان شما حرف بزنند نداشته باشید، باخته اید. از آن سو، NTT DoCoMo هم

فهمید که یک سیستم اختصاصی بسته (که فقط برای گوشی‌های کوچک و فرهنگ ژاپنی طراحی شده باشد) نمی‌تواند شانس برای جهانی شدن داشته باشد. هر دو طرف هم فهمیدند که، برای متقاعد کردن سایتهای وب برای حرف زدن به فرمت شما، باید زبانی داشته باشید باز، محکم و مطابق با استانداردهای جهانی. در دنیای تجارت جنگ فرمتها چیز خوبی نیست.

هر دو سرویس در آستانه ورود به نسل دوم خود هستند. از آنجائیکه WAP 2.0 زودتر به بازار آمده، ما هم آنرا بررسی خواهیم کرد. WAP 1.0 چیزهای خوبی داشت، که WAP 2.0 آنها را حفظ کرده است. یکی اینکه، WAP می‌تواند روی شبکه‌های مختلف کار کند. نسل اول از شبکه‌های سوئیچینگ مداری استفاده می‌کرد، ولی همیشه می‌توانست با سوئیچینگ بسته‌ای هم کار کند. نسل دوم به احتمال زیاد از سوئیچینگ بسته‌ای (مثلاً، GPRS) استفاده خواهد کرد. دیگر اینکه، WAP از دستگاههای متنوعی (از تلفنهای همراه گرفته تا کامپیوترهای کتابی قوی) پشتیبانی می‌کرد، و هنوز هم می‌کند.

WAP 2.0 ویژگیهای جدیدی هم دارد، که مهمترین آنها عبارتند از:

۱. پشتیبانی از مدل پوش (دادن)، در کنار مدل قدیمی پول (گرفتن).
۲. یکپارچه کردن سرویسهای تلفنی در برنامه‌های کاربردی.
۳. پیام‌رسانی چندرسانه‌ای.
۴. داشتن ۲۶۴ علامت تصویری.
۵. ارتباط با وسایل ذخیره‌سازی.
۶. پشتیبانی از افزودنی‌های مرورگر.

مدل پول (pull) برای همه شناخته شده است: مشتری صفحه‌ای را درخواست می‌کند، و آنرا می‌گیرد. مدل پوش (push) یعنی فرستادن (دادن) اطلاعات به مشتری بدون اینکه درخواست کرده باشد، مانند ارسال پیوسته قیمت سهام یا هشدارهای ترافیکی به کاربر.

مدتهاست که صدا و داده در حال یکی شدن هستند، و WAP 2.0 به طرق مختلف از آنها پشتیبانی می‌کند. یک نمونه که قبلاً به آن اشاره کردیم، ایجاد دفترچه تلفن تصویری با I-Mode است. WAP 2.0، علاوه بر ایمیل و تلفن، از پیام‌رسانی چندرسانه‌ای (multimedia messaging) هم پشتیبانی می‌کند.

محبوبیت فوق‌العاده کاراکترهای اموجی در I-Mode، کنسرسیوم WAP را تشویق کرد که ۲۶۴ کاراکتر اموجی در WAP 2.0 بگنجاند. این کاراکترها در زمینه‌های متنوعی از قبیل حیوانات، وسایل خانگی، لباس، احساسات، غذا، بدن انسان، جنسیت، نقشه، موسیقی، گیاهان، ورزش، وقت، آب و هوا، ابزار، وسایل نقلیه، و اسلحه طراحی شده‌اند. جالبست بدانید که در استاندارد WAP 2.0 فقط نام این علائم تصویری تعریف شده، و اقدامی برای طراحی شکل آنها بعمل نیامده است، چون برخی از آنها (مانند بوسیدن و در آغوش گرفتن) در فرهنگهای مختلف معانی بسیار متفاوتی دارند. این مشکل در I-Mode وجود نداشت، چون فقط برای یک کشور طراحی شده بود. پشتیبانی از وسایل ذخیره‌سازی بدان معنا نیست که هر تلفن WAP 2.0 یک هارد دیسک بزرگ خواهد داشت؛ Flash ROM نیز نوعی وسیله ذخیره‌سازی است. یک دوربین بیسیم WAP می‌تواند عکسهای گرفته شده را موقتاً روی یک Flash ROM ذخیره کند، تا زمان فرستادن بهترین آنها به اینترنت فرا برسد.

و بالاخره، مرورگرهای WAP 2.0 می‌توانند با استفاده از افزودنی‌ها (plug-in) قابلیت‌های خود را توسعه دهند. در WAP 2.0 یک زبان اسکریپت‌نویسی نیز پیش‌بینی شده است.

تفاوتهای فنی مختلفی نیز بین WAP 1.0 و WAP 2.0 وجود دارد، که پشته پروتکل و زبان علامتگذاری از مهمترین آنهاست. WAP 2.0 همچنان به پشتیبانی از پشته پروتکل قدیمی شکل ۷-۴۸ ادامه می‌دهد، ولی از استاندارد اینترنت یعنی TCP و HTTP/1.1 نیز پشتیبانی می‌کند. پروتکل TCP در WAP 2.0 دارای چهار تفاوت

جزئی (ولی سازگار) با TCP استاندارد است: (۱) استفاده از پنجره های ثابت 64-KB، (۲) نداشتن شروع آهسته، (۳) سقف ۱۵۰۰ بایتی برای MTU، و (۴) تفاوت جزئی در الگوریتم ارسال مجدد (این تغییرات برای ساده تر شدن کد پروتکل انجام شده اند). TLS یک پروتکل ایمنی لایه انتقال (Transport Layer Security) است، که توسط IETF استاندارد شده است (برای توضیحات بیشتر به فصل ۸ مراجعه کنید). بسیاری از دستگاه های WAP 2.0 همزمان از هر دو پشته پروتکل پشتیبانی خواهند کرد (شکل ۷-۵۵ را ببینید).

XHTML	
WSP	HTTP
WTP	TLS
WTLS	TCP
WDP	IP
Bearer layer (لایه منتقل کننده)	Bearer layer (لایه منتقل کننده)
WAP 1.0 protocol stack	WAP 2.0 protocol stack

شکل ۷-۵۵. WAP 2.0 از دو پشته پروتکل پشتیبانی می کند.

تفاوت دیگر WAP 2.0 با WAP 1.0 در زبان علامتگذاری آنهاست. WAP 2.0 از XHTML ساده (که برای دستگاه های کوچک بیسیم طراحی شده) پشتیبانی می کند. از آنجائیکه NTT DoCoMo هم قبول کرده که از XHTML ساده پشتیبانی کند، با نوشتن صفحات خود به این زبان می توانید مطمئن باشید که آنها در اینترنت و دستگاه های بیسیم بخوبی دیده خواهند شد. این تصمیم به جنگ فرمتها (که رشد وب بیسیم را دچار وقفه کرده بود) خاتمه خواهد داد.

شاید بد نباشد چند کلمه ای هم درباره XHTML ساده بگوئیم. این زبان برای تلفن های همراه، تلویزیون ها، دستگاه های PDA، ماشین های فروش کالا، فراخوان ها، اتومبیل ها، کنسول های بازی، و حتی ساعت در نظر گرفته شده است. به همین دلیل در این ویرایش از شیوه نامه، اسکرپت، یا فریم خبری نیست، ولی اکثر برچسب های استاندارد وجود دارند. این برچسب ها به ۱۱ مازول تقسیم شده اند، که برخی اجباری و برخی دیگر اختیاری اند (تمام این مازولها در XML تعریف شده اند). این مازولها را، همراه با مثالهایی از هر کدام، در شکل ۷-۵۶ ملاحظه می کنید. برای کسب اطلاعات بیشتر درباره این برچسب ها می توانید به www.w3.org مراجعه کنید.

علیرغم توافق WAP و I-Mode بر سر استفاده از XHTML ساده، رقیب جدیدی آنها را تهدید می کند: 802.11. نسل دوم وب بیسیم با سرعت 384 kbps کار می کند، که از 9600 bps نسل اول خیلی بهتر است، ولی در مقایسه با سرعت 11 Mbps یا 54 Mbps دستگاه های 802.11 رنگ می بازد. البته 802.11 همه جا حضور ندارد، ولی با تصمیم صاحبان رستورانها، هتلها، فروشگاهها، شرکتها، فرودگاهها، ایستگاههای اتوبوس، موزه ها، دانشگاهها، بیمارستانها، و بسیاری جاهای دیگر برای نصب ایستگاههای مرکزی 802.11 و دادن امکان دسترسی اینترنت به کارمندان و مشتریان خود، پوشش کافی در مناطق شهری بوجود خواهد آمد، و آن وقت کافیس به نزدیکترین کافه تریا بروید تا بتوانید ضمن خوردن یک فنجان قهوه، ایمیل های خود را هم چک کنید. دور نیست روزی که مغازه ها (کنار آرم کارتهای اعتباری قابل قبول) برای جلب مشتری بیشتر آرم 802.11 را در ویتترین یا کنار در ورودی خود قرار دهند، و مردم هم (مثل صحرانشینان که بدنبال آب بیابانها را زیر پامی گذاشتند) بدنبال 802.11 از این خیابان به آن خیابان سرگردان شوند.

برچسب های نمونه	کارکرد	الزامی؟	ماژول
body, head, html, title	ساختار سند	بلی	ساختاری
br, code, dfn, em, hn, kbd, p, strong	اطلاعات	بلی	متن
a	ایرلینک	بلی	ابر متن
dl, dt, dd, ol, ul, li	لیست	بلی	لیست
form, input, label, option, textarea	فرم	خیر	فرم
caption, table, td, th, tr	جدول	خیر	جدول
img	تصویر	خیر	تصویر
object, param	اپلت، نقشه و غیره	خیر	شیء
meta	اطلاعات اضافی	خیر	اطلاعات اضافی
link	شبه <a>	خیر	لینک
base	نقطه شروع URL	خیر	مبنا

شکل ۷-۵۶. ماژول ها و برچسب های اصلی XHTML.

شاید رستورانها و کافه تریاها خیلی زود برای نصب ایستگاههای مرکزی 802.11 دست بکار شوند، ولی بنظر نمی رسد کشاورزان به این زودی ها چنین کاری بکنند، بهمین دلیل پوشش یکپارچه 802.11 به مناطق شهری (و حداکثر حومه آنها) محدود خواهد ماند (بیاد بیاورید که بُرد 802.11 در بهترین حالت چند صد متر بیشتر نیست). شاید در آینده دستگاههایی به بازار بیایند که در صورت یافتن سیگنال 802.11 از آن استفاده کنند، و وقتی به جایی رسیدند که این پوشش وجود نداشت، بطور خودکار به WAP سوئیچ کنند.

۴-۷ چند رسانه ای

وب بیسیم یکی از تکنولوژیهای جدید و مهیج است، ولی تنها تکنولوژی نیست. برای خیلی ها چند رسانه ای (multimedia) جام مقدس شبکه است. وقتی اسم چند رسانه ای می آید، همه چشم ها خیره می شود (چون برای هر کس چیزی دارد). از آنجائیکه چند رسانه ای به پهنای باند زیادی نیاز دارد، فعلاً فقط روی شبکه های ثابت کار می کنند، و برای دیدن آن روی لینکهای بیسیم باید چند سال دیگر منتظر ماند.

از نظر لغوی، چند رسانه ای یعنی دو یا چند رسانه. حتی ناشر همین کتاب حاضر هم می تواند برای آن بعنوان کتابی چند رسانه ای تبلیغ کند، چون هم متن دارد هم تصویر (شکل). با این حال، وقتی اغلب مردم این اصطلاح را بکار می برند، منظورشان ترکیبی از چند رسانه پیوسته (continuous media) - رسانه هایی که می توان آنها در یک فاصله زمانی مشخص، به همراه نوعی تعامل با کاربر، پخش کرد - است. در عمل، چند رسانه ای بیشتر به ترکیبی از صدا و ویدئو (تصاویر متحرک) اطلاق می شود.

با این همه، بسیاری از مردم به صدای تنها (مثلاً، تلفن یا رادیوی اینترنتی) هم چند رسانه ای می گویند، که پیداست چنین نیست. نام رسانه جویباری (streaming media) برای این قبیل رسانه ها مناسبتر است، ولی اجازه دهید ما هم با افکار عمومی همراهی کنیم و آنرا همان چند رسانه ای بخوانیم. در قسمتهای آینده خواهید دید که کامپیوترها چگونه صدا و ویدئو را پردازش می کنند، چگونه آنها را فشرده می کنند، و چگونه می توان از این تکنولوژیها استفاده کرد. برای یک بحث مفصل (سه جلدی) درباره شبکه های چند رسانه ای کتابهای (Steinmetz and Nahrstedt, 2002; Steinmetz and Nahrstedt, 2003; and Steinmetz and Nahrstedt, 2003b)

بینید.

۱-۴-۷ مقدمه ای بر صدای دیجیتال

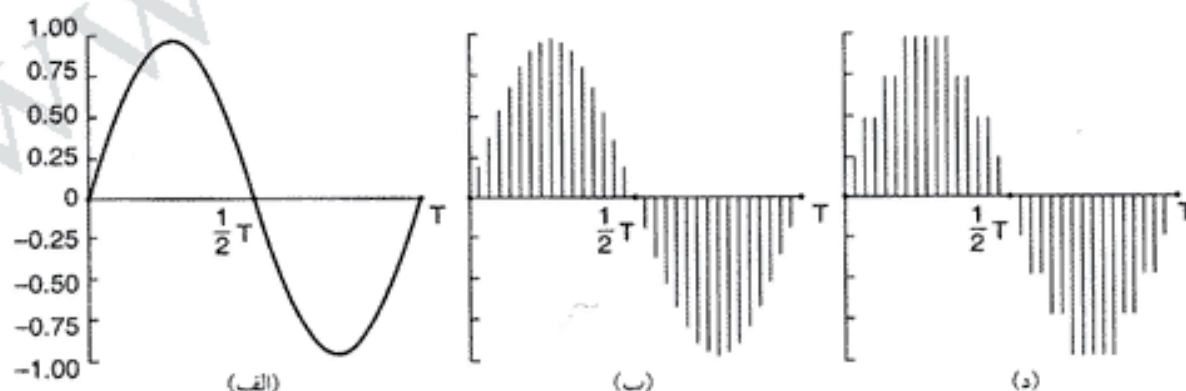
صدا یک موج فشاری یک بُعدی است. وقتی این موج وارد گوش می شود، پرده گوش را به ارتعاش در آورده و باعث می شود تا استخوانهای ریز گوش میانی هم به ارتعاش در آیند. ارتعاش استخوانهای گوش میانی باعث ایجاد سیگنالهای عصبی در گوش داخلی شده، و این سیگنال به مغز می رود. درک ما از صدا توسط مغز (و با تفسیر این سیگنالهای عصبی) انجام می شود. به همین ترتیب، وقتی موج صوتی به میکروفون برخورد می کند، میکروفون آنرا به یک سیگنال الکتریکی (که دامنه آن تابعی از زمان است) تبدیل می کند. پردازش، ذخیره سازی، انتقال و باز تولید این سیگنالها یکی از مهمترین زمینه های تحقیقاتی سیستمهای چند رسانه ای است.

محدوده فرکانسی گوش انسان بین 20 Hz تا 20,000 Hz است. برخی از حیوانات (هویزه سگ و خفاش) می توانند فرکانسهای بالاتر را نیز بشنوند. توانایی گوش در شنیدن اصوات لگاریتمی است، بنابراین نسبت دو صوت با قدرتهای A و B بصورت dB (دسی بل) با فرمول زیر بیان می شود:

$$\text{dB} = 10 \log_{10}(A/B)$$

اگر حد پائین شنوایی (فشاری معادل 0.0003 dyne/cm^2) در فرکانس 1-kHz سینوسی را 0 dB تعریف کنیم، صحبت معمولی حدود 5 dB، و آستانه درد معادل 120 dB است (یعنی تفاوتی در حد ۱ میلیون برابر). حساسیت گوش به صداهای گذرا (در حد چند میلی ثانیه) بسیار شگفت آور است (حتی چشم نیز نورهایی در این فاصله کوتاه را تشخیص نمی دهند). در نتیجه، لرزش صدا (حتی برای چند میلی ثانیه) می تواند در کیفیت انتقال چند رسانه ای اثر بگذارد، در حالیکه چنین لرزشهایی روی کیفیت تصاویر تقریباً بی تأثیر است.

برای تبدیل صدا به سیگنال دیجیتال می توان از یک ADC (مبدل آنالوگ دیجیتال - Analog Digital Converter) استفاده کرد. مبدل ADC ولتاژ الکتریکی را بعنوان ورودی گرفته، و یک خروجی دودویی (باینری) تولید می کند. شکل ۵۷-۷ (الف) یک موج سینوسی را نشان می دهد. برای نمایش دیجیتالی این سیگنال، می توانیم در فواصل ΔT ثانیه از آن نمونه برداری کنیم (شکل ۵۷-۷ ب). اگر موج صوتی سینوسی خالص نباشد ولی بتوان آنرا بصورت مجموع خطی چند موج سینوسی (با بیشترین فرکانس f) نمایش داد، قضیه نایکویست (فصل ۲) می گوید که فرکانس $2f$ برای نمونه برداری آن کافیست. نمونه برداری با فرکانسهای بالاتر اطلاعات بیشتری بدست نمی دهد، چون فرکانسهایی که این نمونه برداری می تواند آشکار کند، در موج اصلی وجود ندارند.



شکل ۵۷-۷. (الف) یک موج سینوسی. (ب) نمونه برداری از موج سینوسی. (ج) کوانتیزه کردن نمونه ها بصورت ۴ بیتی.

نمونه های دیجیتالی هرگز دقیق و کامل نیستند. در مثال شکل ۵۷-۷ (ج) فقط ۹ مقدار مجاز (از -1.00 تا +1.00 با گامهای 0.25) وجود دارد (و همانطور که می دانید، برای نمایش ۹ مقدار به ۴ بیت دودویی نیاز داریم). با

نمونه‌های ۸ بیتی می‌توان ۲۵۶ مقدار مختلف را ثبت کرد؛ و با نمونه‌های ۱۶ بیتی می‌توانیم ۶۵,۵۳۶ مقدار مجزا داشته باشیم. خطایی که در اثر محدود بودن تعداد بیتها برای نمایش هر نمونه بوجود می‌آید، به نویز کوانتیزه کردن (quantization noise) معروف است. اگر این خطا زیاد باشد، گوش می‌تواند آنرا تشخیص دهد.

تلفن و دیسکهای صوتی دو نمونه آشنا از کوانتیزه کردن صوت هستند. مدولاسیون کُد پالس (PCM)، که در سیستمهای تلفن بکار می‌رود، از نمونه‌های ۸ بیتی استفاده می‌کند و در هر ثانیه ۸۰۰۰ نمونه برمی‌دارد. در آمریکای شمالی و ژاپن، ۷ بیت برای داده و ۱ بیت برای کنترل بکار می‌رود، در حالیکه در اروپا از هر ۸ بیت برای داده استفاده می‌شود. نرخ داده در این سیستمها بترتیب 56,000 bps و 64,000 bps است. با نرخ نمونه‌برداری 8000 samples/sec فرکانسهای بالاتر از 4 kHz از دست می‌روند.

نرخ نمونه‌برداری در دیسکهای صوتی 44,100 samples/sec است، که برای آشکارسازی فرکانسهای تا 22,050 Hz کافیست (که برای آدمهای موزیک دوست خوب است، ولی سگ‌ها بهیچوجه از آن راضی نخواهند بود). در این دیسکها، نمونه‌ها ۱۶ بیتی (با توزیع خطی روی محدوده دامنه) هستند. توجه کنید که ۱۶ بیت فقط برای نمایش ۶۵,۵۳۶ مقدار کافیست، در حالیکه بین پائینترین و بالاترین حد شنوایی انسان حداقل ۱,۰۰۰,۰۰۰ گام قابل شنیدن وجود دارد؛ بهمین دلیل، حتی در این دیسکها هم مقداری نویز کوانتیزه کردن وجود دارد. با 44,100 نمونه ۱۶ بیتی در هر ثانیه، دیسکهای صوتی به پهنای باند 705.6 kbps برای اصوات مونو، و 1.411 Mbps برای اصوات استریو نیاز دارند. همانطور که می‌بینید، برای انتقال صوت غیرفشرده با کیفیت CD به یک کانال کامل T1 نیاز داریم (در مورد ویدئو که وضع از این هم بدتر است - قسمت بعد را ببینید).

کامپیوترها می‌توانند صدای دیجیتال را براحتی پردازش کنند. امروزه برنامه‌های بسیاری برای ضبط، پخش، ادیت، میکس و ذخیره کردن امواج صوتی در بازار یافت می‌شود، و تقریباً تمام حرفه‌ایها برای ضبط و ادیت کردن صوت از سیستمهای دیجیتال استفاده می‌کنند.

صحبت یکی دیگر از زمینه‌های مهم در صدای دیجیتال است. صحبت انسانها معمولاً در محدوده 600 Hz تا 6000 Hz است. حروف صدادار و بی‌صدا هر کدام ویژگیهای خاص خود دارند. حروف صدادار وقتی ایجاد می‌شوند که مجرای صوتی باز است و تشدید صورت می‌گیرد، و فرکانس آن به اندازه و شکل حنجره، و محل قرار گرفتن زبان و آرواره فرد بستگی دارد. اصوات صدادار تناوبی در حدود 30 msec دارند. حروف بی‌صدا زمانی ایجاد می‌شوند که مجرای صوتی تقریباً مسدود است، و معمولاً بدون تناوب و شکل خاصی هستند.

در برخی از سیستمهای تولید صحبت (بجای ترکیب شکل موج واژه‌ها) از مدل‌های ساده شده مجرای صوتی (با پارامترهای محدودتری از قبیل اندازه و شکل حفره‌های صوتی) استفاده می‌شود. طرز کار این سیستمها در حیطه بحث کتاب حاضر نیست.

۲-۴-۷ فشرده‌سازی صدا

همانطور که دیدید برای انتقال صدای استریو با کیفیت CD به پهنای باند 1.411 Mbps نیاز داریم، پس پیداست که برای انتقال این رسانه روی اینترنت باید آنرا تا حد زیادی فشرده کنیم. برای این کار الگوریتمهای فشرده‌سازی مختلفی توسعه داده شده‌اند، که شاید محبوبترین آنها صدای MPEG باشد. این الگوریتم دارای سه لایه (نوع) مختلف است، که لایه سوم (MP3 - MPEG audio layer 3) از همه قویتر و معروفتر است. حجم زیادی از موسیقی با این فرمت روی اینترنت وجود دارد، که البته همه آنها قانونی نیستند، و همین منجر به دعوای قانونی بسیاری بین متخلفان و صاحبان این آثار شده است. فرمت MP3 در واقع بخش صوتی استاندارد فشرده‌سازی ویدئویی MPEG است، که در قسمتهای آینده درباره آن هم صحبت خواهیم کرد.

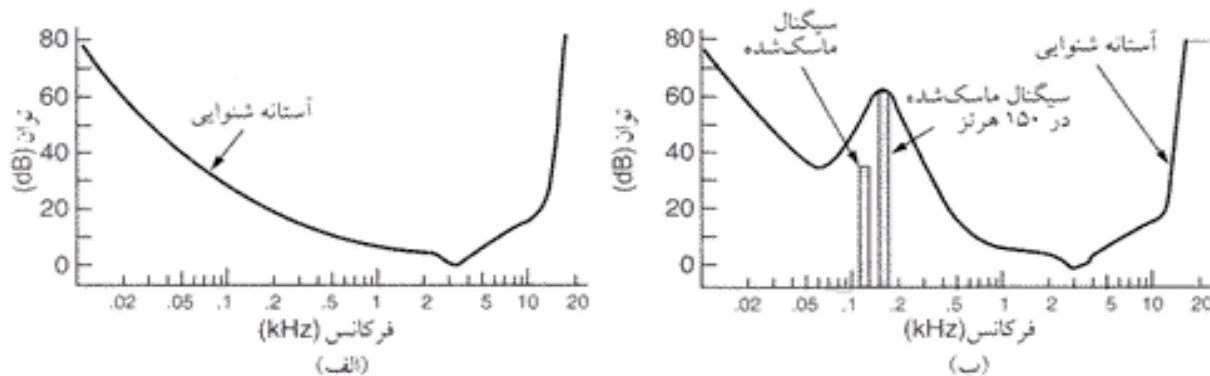
فشرده‌سازی صدا را به دو روش می‌توان انجام داد. در کُدگذاری شکل موج (waveform coding) سیگنال

صوتی بصورت ریاضی و با استفاده از تبدیل فوریه، به فرکانسهای تشکیل دهنده آن تجزیه می شود. در شکل ۱-۲ (الف) نمونه ای از یک موج و دامنه های فوریه آنرا می بینید. سپس دامنه هر مؤلفه به ساده ترین شکل ممکن کُد می شود. هدف از این کار بازسازی شکل موج با حداقل بیت های ممکن است.

روش دیگر، کُدگذاری ادراکی (perceptual coding)، سیگنال ورودی را با استفاده از نقائص سیستم صوتی انسان بگونه ای کُد می کند که شنونده متوجه تفاوت آنها نشود (حتی اگر این سیگنالها در اسیلوسکوپ بهیچوجه یکسان دیده نشوند). کُدگذاری ادراکی بر اساس تحقیقات روان شنوایی (psychoacoustics) - دانش درک انسانها از صوت - بنا نهاده شده است. فرمت MP3 از کُدگذاری ادراکی استفاده می کند.

نکته کلیدی در کُدگذاری ادراکی این است که برخی از اصوات می توانند صداهای دیگر را پوشانند. فرض کنید در یک روز تابستانی در هوای آزاد مشغول اجرای یک کنسرت فلوت هستید. ناگهان وسط کنسرت شما چند متر آن طرفتر کارگران شهرداری چکشهای بادی خود را روشن کرده و مشغول کندن آسفالت خیابان می شوند. در این حال دیگر هیچکس نمی تواند صدای فلوت شما را بشنود، چون صدای چکشهای بادی نوای فلوت را پوشانده است. اگر بخواهید همین کنسرت را به جای دیگر مخابره کنید، فقط کافیست صدای چکشهای بادی را ارسال کنید، چون به هر حال کسی صدای فلوت را نخواهد شنید. به این پدیده - پوشانده شدن صدای ملایم یک باند فرکانسی توسط صدای خشن یک باند فرکانسی دیگر - ماسک فرکانسی (frequency masking) می گویند. در حقیقت، حتی بعد از خاموش شدن چکشهای بادی باز هم تا مدتی شنوندگان قادر به شنیدن صدای فلوت نخواهند بود، چون گوش (برای جلوگیری از صدمه دیدن سیستم شنوایی) بهره شنوایی خود را با شروع صدای چکشها پائین می آورد، و بازگشت آن به حالت قبل (بعد از خاموش شدن چکشها) مدتی طول خواهد کشید. به این پدیده هم ماسک موقتی (temporal masking) گفته می شود.

برای دیدن تأثیر کمی این پدیده ها، یک آزمایش ترتیب می دهیم. در یک اتاق کاملاً ساکت، فردی گوشی های متصل به کارت صوتی کامپیوتر را به گوش می گذارد. کامپیوتر یک موج سینوسی کم قدرت با فرکانس 100 Hz تولید می کند، و بتدریج قدرت آنرا بالا می برد. به این فرد گفته ایم که به محض شنیدن صدا کلیدی را بزند. در این لحظه کامپیوتر قدرت صدا را ثبت کرده، و همین کار را با فرکانسهای 200 Hz، 300 Hz و ... (تا رسیدن به آستانه شنوایی انسان) تکرار می کند. با تکرار این آزمایش برای افراد مختلف و محاسبه متوسط قدرت لازم برای شنیده شدن صدا در هر فرکانس، نمودار لگاریتمی شکل ۷-۵۸ (الف) را رسم کرده ایم. این نمودار بروشنی نشان می دهد که هیچ نیازی به کُد کردن فرکانسهای که قدرت آنها زیر آستانه شنوایی انسان باشد، نیست. برای مثال، اگر قدرت یک سیگنال 100 Hz فقط 20 dB باشد، می توان آنرا حذف کرد بدون اینکه تأثیری روی کیفیت خروجی بگذارد، چون طبق شکل ۷-۵۸ (الف) این سیگنال زیر آستانه شنوایی انسانها قرار دارد.



اکنون آزمایش دیگری می‌کنیم. در اینجا هم کامپیوتر همان تست قبل را اجرا می‌کند، ولی این بار یک موج سینوسی ثابت با فرکانس (مثلاً) 150 Hz در زمینه فرکانس تست پخش می‌شود. همانطور که به روشنی می‌توان دید (شکل ۷-۵۸ ب)، آستانه شنوایی فرکانسهای نزدیک 150 Hz بالاتر رفته است.

نتیجه این آزمایش آن است که، با دنبال کردن فرکانسهای نزدیک به هم می‌توان سیگنالهای ضعیفتر را حذف کرده، و در تعداد بیت‌های مورد نیاز صرفه‌جویی کرد. شکل ۷-۵۸ (ب) نشان می‌دهد که حتی اگر فرکانسهای 125-Hz را بکلی حذف کنیم، هیچ گوشی متوجه تفاوت آن نخواهد شد. حتی اگر در جایی سیگنال قویتر متوقف شود، باز هم می‌توانیم تا مدتی به حذف سیگنال ضعیفتر ادامه دهیم، چون برگشت گوش به حالت نرمال کمی طول می‌کشد (اثر ماسک موقتی). فرمت MP3 نیز در اساس چیزی نیست جز تبدیل فوریه صدا به سیگنالهای سینوسی و حذف تمام سیگنالهایی که ماسک شده‌اند.

با این زمینه‌تئوریک، اکنون می‌توانیم نشان دهیم که کدگذاری چگونه انجام می‌شود. برای فشرده‌سازی صدا، از شکل موج ورودی با نرخهای 32 kHz، 44.1 kHz، یا 48 kHz نمونه‌برداری می‌شود. نمونه‌برداری را می‌توان روی یک یا دو کانال به طرق زیر انجام داد:

۱. تک‌صدایی (یک استریو ورودی).
۲. تک‌صدایی دوبل (مثلاً، حاشیه صوتی انگلیسی و ژاپنی).
۳. استریوی منفصل (فشرده‌سازی جداگانه هر کانال).
۴. استریوی متصل (استفاده کامل از افزونگی بین کانالها).

ابتدا، نرخ بیت خروجی انتخاب می‌شود. با الگوریتم MP3 می‌توان یک CD استریوی را که رول را تا 96 kbps فشرده کرد، بطوریکه حتی دو آتشه‌ترین طرفداران این موسیقی هم متوجه افت کیفیت نشوند. برای یک کنسرت پیانو به حداقل 128 kbps نیاز داریم، چون نسبت سیگنال به-نویز موسیقی را که رول بسیار بالاتر از پیانو است. اگر نرخ پائینتری انتخاب کنیم، کیفیت صدا قدری افت خواهد کرد.

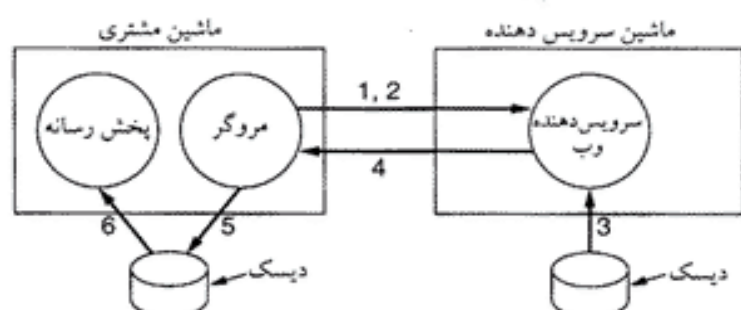
سپس، نمونه‌برداری در گروههای ۱۱۵۲ تایی (تقریباً 26 msec) انجام می‌شود. هر گروه با عبور از ۳۲ فیلتر دیجیتال به ۳۲ باند فرکانسی تفکیک می‌شود. همزمان، موج ورودی به یک مدل روان‌شنوایی داده می‌شود تا فرکانسهای ماسک شده مشخص شوند. پس از آن ۳۲ باند فرکانسی باز هم تبدیل می‌شوند، تا طیف فرکانسی دقیقتری بدست آید. در مرحله بعد، تعداد بیت‌های موجود بین این باندها تقسیم می‌شود (با این معیار که به توانهای طیفی ماسک‌نشده بیت‌های بیشتری برسد، باندهای ماسک‌نشده کم‌قدرت تر بیت‌های کمتری بگیرند، و باندهای ماسک‌شده اصلاً چیزی نگیرند). در پایان هم، بیت‌های حاصله با استفاده از روش هافمن - اختصاص کدهای کوتاه‌تر به اعدادی که بیشتر تکرار شده‌اند، و اختصاص کدهای بلندتر به اعداد کمتر تکرار شده - کد می‌شوند. در واقع، قصه فشرده‌سازی مفصلتر از اینهاست. تکنیکهای مختلفی برای کاهش نویز، یکنواخت‌سازی، و بهره‌گیری از افزونگی بین کانالها وجود دارد، که از حوزه بحث این کتاب خارج هستند. برای کسب اطلاعات بیشتر در زمینه ریاضیات فشرده‌سازی صدا به (Pan, 1995) مراجعه کنید.

۳-۴-۷ صدای جویباری

در این قسمت سه تا از کاربردهای صدای دیجیتال را مورد بررسی قرار می‌دهیم. اولین آنها صدای جویباری (streaming audio) است: شنیدن صدا بصورت پیوسته روی اینترنت. در قسمتهای بعد با رادیوی اینترنتی و صدا روی IP (یا همان تلفن اینترنتی) آشنا خواهید شد.

اینترنت پُر است از سایتهای موسیقی، که کاربر می‌تواند با کلیک کردن روی لینک هر ترانه به آن گوش کند.

بعضی از این سایتها مجانی هستند (گروههای تازه کاری که بدنبال شهرت اند)، و برای بعضی دیگر باید پول داد (البته روی این سایتها هم نمونه های مجانی یافت می شود). ساده ترین روش پخش موسیقی را در شکل ۷-۵۹ ملاحظه می کنید.



- ۱- برقراری اتصال TCP
- ۲- ارسال درخواست HTTP GET
- ۳- سرویس دهنده فایل را از دیسک می خواند
- ۴- فایل فرستاده می شود
- ۵- مرورگر فایل را روی دیسک می نویسد
- ۶- برنامه پخش فایل را از دیسک خوانده و پخش می کند

شکل ۷-۵۹. یک روش ساده برای پخش موسیقی در صفحات وب.

کار با کلیک کردن روی لینک موسیقی دلخواه شروع می شود، و در اینجا مرورگر وارد صحنه می شود. در مرحله ۱ مرورگر یک اتصال TCP به سرویس دهنده وب هدف برقرار می کند، و در مرحله ۲ با فرمان *GET* فایل موسیقی را از آن درخواست می کند. سپس (مراحل ۳ و ۴)، سرویس دهنده فایل خواسته شده را (که می تواند با فرمت MP3 یا هر فرمت دیگری باشد) از روی دیسک خوانده و برای مرورگر می فرستد. اگر این فایل از حجم حافظه سرویس دهنده بزرگتر باشد، آنرا بصورت قطعه قطعه خواهد فرستاد.

مرورگر با استفاده از نوع MIME فایل دریافت شده (مثلاً، *audio/mp3*)، تشخیص می دهد که چگونه باید آنرا پخش کند. مرورگرها معمولاً برای این کار از برنامه های کمکی مانند *Windows Media Player*، *RealOne Player*، یا *Winamp* کمک می گیرند. از آنجائیکه متداولترین روش ارتباط با برنامه های کمکی نوشتن فایل روی محلی موقتی است، مرورگر ابتدا فایل را روی دیسک ذخیره می کند (مرحله ۵). سپس، برنامه پخش موسیقی را اجرا کرده و نام فایل را به آن می دهد. با این کار برنامه پخش موسیقی شروع به خواندن فایل از روی دیسک، و پخش آن می کند (مرحله ۶).

این روش هیچ اشکالی ندارد و در عمل کار هم می کند، ولی مشکل اینجاست که تمام فایل باید قبل از شروع پخش خوانده شود. اگر یک فایل موسیقی 4 MB باشد (اندازه ای معمولی برای فایل های MP3)، و کاربر با مودم 56 kbps به اینترنت متصل شده باشد، قبل از شروع پخش موسیقی باید ۱۰ دقیقه انتظار بکشد. اغلب موسیقی دوستان تحمل چنین انتظاری را ندارند.

برخی از سایت های وب برای حل این مشکل (بدون آنکه روش دسترسی به موسیقی را تغییر دهند) از روش ذیل استفاده می کنند. در این روش، لینک صفحه وب مستقیماً به فایل موسیقی اشاره نمی کند، بلکه یک متافایل (*metafile*) - فایل بسیار کوچکی که فقط نام فایل موسیقی در آن است) را مشخص می کند. یک متافایل چنین شکلی دارد:

`rtsp://joes-audio-server/song-0025.mp3`

وقتی مرورگر این فایل یک خطی را دریافت کرد، طبق معمول آنرا را روی دیسک نوشته و پس از اجرای برنامه کمکی پخش موسیقی، نام فایل مزبور را به آن می دهد. برنامه پخش بعد از خواندن فایل متوجه می شود که این یک URL است، بنابراین به سرویس دهنده *joes-audio-server* متصل شده و آهنگ مشخص شده را درخواست می کند. توجه کنید که در اینجا دیگر مرورگر نقشی ندارد.

در اغلب موارد، سرویس دهنده مشخص شده در متافایل همان سرویس دهنده وب نیست، و در واقع حتی یک

سرویس دهنده HTTP هم نیست، بلکه نوع خاصی از سرویس دهنده رسانه (media server) است. در مثال بالا، سرویس دهنده رسانه از پروتکل RTSP (پروتکل جویباری بی درنگ - Real Time Streaming Protocol) استفاده می کند (به نام پروتکل *rtsp* در ابتدای URL دقت کنید). این پروتکل در RFC 2326 تعریف شده است. برنامه پخش چهار وظیفه اصلی دارد:

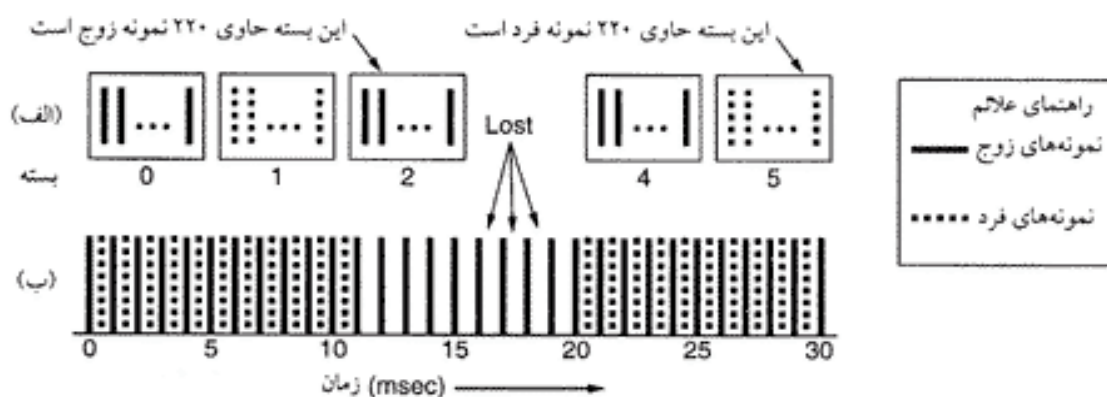
۱. مدیریت واسط کاربر.
۲. مقابله با خطاهای انتقال.
۳. باز کردن (از حالت فشرده خارج کردن) موسیقی.
۴. حذف لرزش ها.

اغلب برنامه های پخش امروزی دارای ظاهری بسیار جالب و شکیل هستند، و از دستگاههای پخش استریوی مدل بالا (با تمام دکمه، عقربه ها، و چراغهای رنگارنگ) تقلید می کنند. بعضی از آنها حتی اجازه می دهند تا شکل ظاهری برنامه را (که به پوست - skin - معروف است) به سلیقه خود تغییر دهید. این یکی از وظایف برنامه پخش در رابطه با واسط کاربر است.

وظیفه دوم برنامه پخش مقابله با خطاهای انتقال است. برنامه های پخش بی درنگ بندرت از TCP برای انتقال موسیقی استفاده می کنند، چون مدیریت خطا در TCP می تواند باعث ایجاد وقفه های آزار دهنده در موسیقی شود. پروتکل مرسوم برای انتقال موسیقی پروتکل RTP است، که آنرا در فصل ۶ دیدید. مانند اغلب پروتکل های بی درنگ، RTP هم لایه ایست بر فراز UDP، بنابراین گم شدن بسته ها کاملاً محتمل است.

در برخی از موارد، برای بهبود مدیریت خطا از روشهای یک درمیانی (interleaving) استفاده می شود. برای مثال، هر بسته می تواند حاوی ۲۲۰ نمونه استریو (یک زوج ۱۶ بیتی) معادل ۵ msec موسیقی باشد، ولی پروتکل انتقال بجای آن ابتدا ۱۰ msec از نمونه های فرد را در یک بسته، و بدنبال آن ۱۰ msec از نمونه های زوج را در بسته بعدی می فرستد. در این روش اگر یک بسته گم شود، شکاف ۵ msec در موسیقی بوجود نمی آید، و برنامه پخش می تواند با استفاده از تکنیکهای درون یابی زوجهای گم شده را تخمین بزند.

در شکل ۷-۶۰ تکنیک یک درمیانی را ملاحظه می کنید. در اینجا هر بسته شامل نمونه های فرد یا زوج فاصله زمانی ۱۰ msec است. در نتیجه، از دست رفتن بسته ۳ باعث ایجاد شکاف در موسیقی نمی شود، بلکه فقط کیفیت پخش بصورت موقتی و زودگذر افت خواهد کرد. برنامه پخش برای حفظ پیوستگی موسیقی، با استفاده از تکنیکهای ریاضی درون یابی (interpolating) نمونه های گم شده را (از روی نمونه های قبلی و بعدی) تخمین

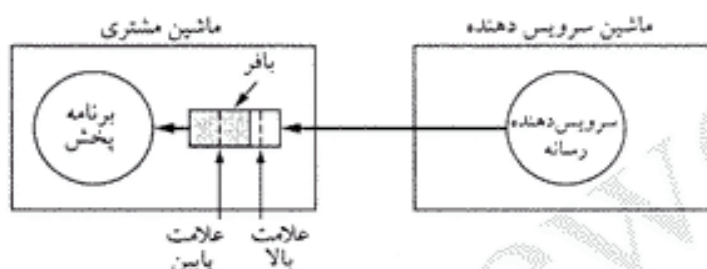


شکل ۷-۶۰. وقتی نمونه ها بصورت یک درمیان فرستاده شوند، گم شدن یک بسته بجای ایجاد شکاف در موسیقی فقط باعث کاهش موقتی کیفیت آن خواهد شد.

می زند. البته این روش خاص فقط با نمونه های فشرده نشده کار می کند، ولی بخوبی نشان می دهد که روشهای هوشمندانه چگونه می توانند به بهبود کیفیت پخش کمک کنند. (در RFC 3119 برای صدای فشرده شده نیز روشی ارائه شده است.)

وظیفه سوم برنامه پخش باز کردن (decompress) موسیقی است. با آن که این تکنیک نسبتاً ساده است، اما به محاسبات زیادی نیاز دارد.

وظیفه چهارم برنامه پخش، حذف لرزش (jitter)، جزء آزاردهنده ترین وظایف همه سیستمهای پی درنگ است. تمام سیستمهای صدای جویباری قبل از شروع پخش، ۱۰ الی ۱۵ ثانیه موسیقی را بافر می کنند (شکل ۷-۶۱ را ببینید). بطور ایده آل، سرویس دهنده این بافر را با همان سرعتی که برنامه پخش از آن می خواند، پُر می کند. اما در واقعیت ممکنست چنین چیزی عملی نباشد، پس به نوعی حلقه فیدبک نیاز داریم.



شکل ۷-۶۱. برنامه پخش، بسجای پخش مستقیم موسیقی از شبکه، ورودی سرویس دهنده رسانه را بافر کرده، و موسیقی را از این بافر اجرا می کند.

برای پُر نگه داشتن بافر از دو روش می توان استفاده کرد. در روش سرویس دهنده پول (pull server)، تا زمانی که بافر جا دارد، برنامه پخش به درخواست از سرویس دهنده رسانه برای ارسال بسته های بعدی ادامه می دهد. هدف برنامه پخش آن است که بافر را تا حد امکان پُر نگه دارد. عیب بزرگ این روش ارسال درخواستهای غیرضروری از طرف برنامه پخش است. سرویس دهنده می داند که باید تمام فایل را بفرستد، پس درخواست پخش کننده دیگر برای چیست؟ به همین دلیل از روش فوق بندرت استفاده می شود.

در روش دیگر، سرویس دهنده پوش (push server)، برنامه پخش فقط یک بار درخواست PLAY را به سرویس دهنده رسانه می فرستد، و این سرویس دهنده به وظیفه خود (فرستادن پیوسته داده ها) عمل می کند. در اینجا دو احتمال پیش می آید: سرویس دهنده رسانه با سرعت معمولی پخش هماهنگ یا از آن سریعتر است. در هر دو حالت، قبل از شروع پخش مقداری داده بافر می شود. اگر سرعت سرویس دهنده همپای پخش باشد، داده از آن وارد بافر شده و از سمت دیگر توسط برنامه پخش خوانده می شود. اگر همه چیز خوب پیش رود، مقدار داده درون بافر همیشه ثابت می ماند. این ساده ترین حالت است، چون لازم نیست هیچ پیام کنترلی (در هر دو جهت) فرستاده شود. حالت دیگر اینست که سرویس دهنده پوش داده را با سرعتی بیشتر از آنچه برنامه پخش می خواند، بفرستد. در این وضعیت سرویس دهنده همیشه می تواند خود را به برنامه پخش برساند، حتی اگر گاهی (بدلایلی) از آن عقب بماند. اما این حالت (عجله سرویس دهنده برای عقب نماندن)، می تواند منجر به رونویسی بافر (buffer overrun) - نوشتن روی داده هایی که هنوز خوانده نشده اند - شود.

برنامه پخش برای جلوگیری از این وضعیت می تواند دو علامت حد پایین و حد بالا در بافر تعریف کند. سرویس دهنده آنقدر داده می فرستد که بافر تا علامت حد بالا پُر شود؛ در اینجا پخش کننده به سرویس دهنده پیام

می دهد که فرستادن را متوقف کند. از آنجائیکه سرویس دهنده قبل از دریافت پیام توقف همچنان به ارسال داده ادامه می دهد، فاصله علامت حد بالای بافر تا ظرفیت نهایی آن باید از حاصلضرب پهنای باند در تأخیر انتشار شبکه بیشتر باشد. پخش کننده حتی بعد از توقف ارسال سرویس دهنده به خالی کردن بافر ادامه می دهد، و وقتی مقدار بافر به علامت حد پائین رسید، به آن اطلاع می دهد که ارسال داده را از سر گیرد. محل علامت حد پائین بافر باید بگونه ای انتخاب شود که بافر تا رسیدن داده های جدید خالی نشود (buffer underrun). برای آن که برنامه پخش بتواند سرویس دهنده رسانه را کنترل کند، به پروتکلی مانند RTSP نیاز دارد. این پروتکل و مکانیزمهای کنترلی آن در RFC 2326 تعریف شده است (البته این پروتکل با RTP تفاوت دارد). در شکل ۶۲-۷ فرمانهای اصلی RTSP را ملاحظه می کنید.

فرمان	عملکرد سرویس دهنده
DESCRIBE	پارامترها را لیست می کند
SETUP	یک کانال بین پخش کننده و سرویس دهنده برقرار می کند
PLAY	شروع به ارسال داده می کند
RECORD	شروع به دریافت داده می کند
PAUSE	ارسال را موقتاً قطع می کند
TEARDOWN	کانال را رها می کند

شکل ۶۲-۷. فرمانهای RTSP از پخش کننده به سرویس دهنده.

۴-۷-۴ رادیوی اینترنتی

همین که امکان اسال صدای جویباری (پیوسته) روی اینترنت فراهم شد، ایستگاههای رادیویی به این فکر افتادند که علاوه بر روش معمول از طریق اینترنت هم برنامه پخش کنند. کمی بعد اولین ایستگاههای رادیویی دانشگاهی روی اینترنت شروع بکار کردند؛ و بعد از آن هم ایستگاههای دانشجویی راه افتادند. با تکنولوژی امروزی، تقریباً هر کسی می تواند برای خود یک ایستگاه رادیویی داشته باشد. ایستگاههای رادیویی اینترنتی موضوع بسیار نو و پویایی هستند، ولی ارزش آنها دارد که کمی درباره آن حرف بزنیم.

دو رهیافت کلی برای رادیوی اینترنتی (Internet radio) وجود دارد. در رهیافت اول برنامه ها روی دیسک ضبط می شوند، و شنوندگان می توانند برنامه های دلخواه را از آرشیو بار کرده و به آن گوش کنند. در حقیقت این فقط شکل دیگری از صدای جویباری است، که در قسمت قبل توضیح دادیم. همچنین می توان برنامه های زنده رادیویی را روی دیسک ضبط کرد، بگونه ای که آرشیو فقط نیم ساعت از برنامه زنده عقب باشد. مزیت این روش سادگی آن است، و می توان از تکنیکهایی که قبلاً تشریح کردیم برای پیاده سازی آن استفاده کرد.

رهیافت دیگر پخش برنامه زنده روی اینترنت است. برخی از ایستگاههای رادیویی همزمان با پخش معمولی، برنامه های خود را روی اینترنت نیز پخش می کنند، ولی تعداد ایستگاههایی که فقط پخش اینترنتی دارند روز به روز در حال افزایش است. برخی از تکنیکهای صدای جویباری در اینجا نیز قابل استفاده است، ولی چند تفاوت کلیدی هم وجود دارد.

یکی از نکاتی که در هر دو سیستم یکسان است، لزوم بافر کردن داده ها برای حذف لرزش است. با بافر کردن ۱۰ الی ۱۵ ثانیه صدا می توان تا حد زیادی با قطع و وصلهای اجتناب ناپذیر شبکه مقابله کرد. مادامیکه بسته ها به موقع برسند، زمان رسیدن آنها چندان اهمیتی ندارد.

یکی از تفاوت های مهم صدای جویباری و رادیوی اینترنتی این است که سرویس دهنده رسانه می تواند داده ها را سریعتر از آنچه پخش کننده پخش می کند، بفرستد - چون پخش کننده می تواند با رسیدن بافر به علامت حد بالا

سرویس دهنده را وادار به توقف کند (یا از آن بخواهد در این مدت بسته های گم شده را دوباره ارسال کند). اما رادیوی اینترنتی نمی تواند داده ها را سریعتر از آنچه در حال تولید و پخش است، بفرستد.

تفاوت دیگر این است که یک رادیوی پخش زنده اینترنتی معمولاً هزاران شنونده همزمان دارد، در حالیکه صدای جویباری اساساً سیستمی نقطه-به-نقطه است. در این حالت، رادیوی اینترنتی باید از پروتکل های چندپخشی RTP/RTSP استفاده کند. این مؤثرترین روش پخش اینترنتی است.

اما در عمل رادیوهای اینترنتی به این روش کار نمی کنند. آنچه که معمولاً اتفاق می افتد این است که کاربر یک اتصال TCP به ایستگاه رادیویی برقرار کرده، و محتویات را روی این اتصال دریافت می کند. البته این روش مشکلاتی هم دارد، مثلاً وقتی پنجره پُر می شود یا بسته ها گم می شوند، پخش دچار وقفه خواهد شد.

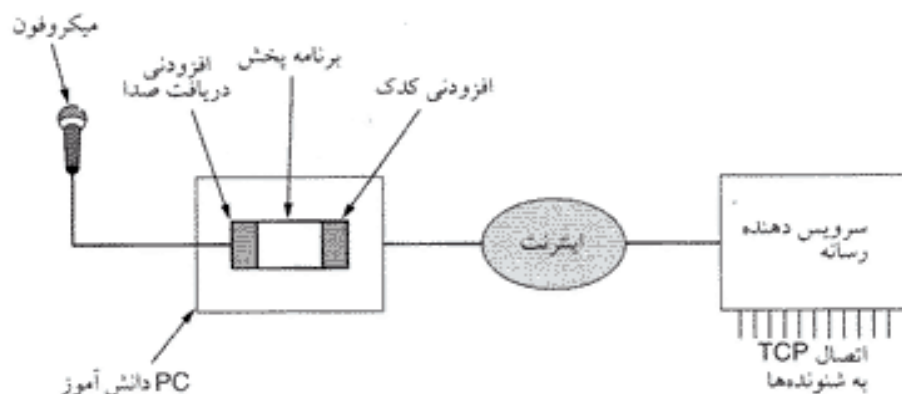
سه دلیل عمده برای استفاده از پروتکل تکپخشی TCP بجای پروتکل چندپخشی RTP وجود دارد. اول اینکه، اغلب ISP ها از چندپخشی پشتیبانی نمی کنند، و در واقع این گزینه عملی نیست. دوم اینکه، پروتکل RTP کمتر از TCP شناخته شده است، و کادر فنی رادیوهای اینترنتی (که معمولاً شرکت های کوچکی بیش نیستند) مایلند با همان TCP معروف کار کنند. و سوم اینکه، معمولاً افراد بیشتر در محل کار به رادیوهای اینترنتی گوش می کنند، و این محل ها هم اغلب پشت دیوار آتش (firewall) قرار دارند. این دیوارهای آتش معمولاً فقط به پروتکل های SMTP (پورت 25 - برای ایمیل)، UDP (پورت 53 - برای DNS)، و HTTP (پورت 80 - برای وب) اجازه عبور می دهند، و جلوی هر چیز دیگر (از جمله RTP) را می گیرند. بنابراین، برای رادیوهای اینترنتی بهترین روش آنست که خود را یک سرویس دهنده HTTP (که با اتصال TCP کار می کند) جا بزنند. البته با این کار برنامه های چندرسانه ای کارایی خود را بشدت از دست می دهند.

از آنجائیکه رادیوی اینترنتی موجودی نوپاست، جنگ فرمتها در آن با شدت تمام ادامه دارد. فرمت های از آنجائیکه RealAudio، Windows Media Audio، و MP3 بشدت در تلاشند تا خود را بعنوان فرمت غالب رادیوی اینترنتی معرفی کنند. اخیراً فرمت دیگری بنام Vorbis نیز وارد این معرکه شده، که از نظر فنی شبیه MP3 است، ولی منبع باز (open source) است و آنقدر با MP3 فرق دارد که نیازی به رعایت حق اختراع آن نداشته باشد.

رادیوهای اینترنتی معمولاً یک صفحه وب دارند که فهرست برنامه های آن (و اطلاعات متفرقه دیگر، باضافه مقدار زیادی آگهی) را نمایش می دهد. اغلب این رادیوها در حاضر از فرمت های مختلفی پشتیبانی می کنند، که کاربر می تواند بدخواه خود یکی از آنها را انتخاب کند. برای هر یک از این فرمتها یک آیکون جداگانه وجود دارد، که به متافایل مربوطه مرتبط است.

وقتی کاربر روی یکی از آیکونها کلیک کند، متافایل مربوطه را دریافت می کند. مرورگر با استفاده از اطلاعات این متافایل می تواند برنامه کمکی مناسب را انتخاب کند. سپس این فایل را روی دیسک نوشته، و نام فایل را به برنامه کمکی می دهد. برنامه پخش این متافایل را خوانده، و URL مقصد را (که معمولاً پروتکل آن http است، تا بتواند دیوار آتش را دور بزند) از آن استخراج می کند؛ سپس به سرویس دهنده وصل شده، و مانند یک رادیو شروع به کار می کند. پروتکل http برای صدا (که فقط یک استریم است) کاملاً کفایت می کند، ولی اگر پای ویدئو (که حداقل دو استریم مستقل دارد) در میان باشد، دیگر کاری از http ساخته نیست و حتماً باید از چیزی شبیه rtsp استفاده کرد.

جذابترین جنبه رادیوی اینترنتی سادگی آن است، بطوری که حتی یک دانش آموز دبیرستانی هم می تواند رادیوی اینترنتی راه بیندازد. در شکل ۷-۶۳ ساده ترین پیکربندی یک رادیوی اینترنتی را می بینید. همانطور که می بینید، یک رادیوی اینترنتی در واقع چیزی نیست جز یک PC معمولی با کارت صوتی و میکروفون. نرم افزار هم فقط یک برنامه پخش رسانه (مانند Winamp) است، باضافه افزودنی های لازم برای گرفتن صدای میکروفون و تبدیل آن به فرمت مناسب (مانند MP3 یا Vorbis).



شکل ۷-۶۳. یک رادیوی اینترنتی دانش آموزی.

استریم صدای خروجی، سپس، به یک سرویس دهنده HTTP روی اینترنت فرستاده می شود تا در اختیار شنوندگان قرار گیرد. این قبیل سرویس دهنده ها معمولاً تعداد زیادی ایستگاه رادیویی را میزبانی می کنند، و حتی صفحه ای دارند که نشان می دهد در لحظه حاضر کدام ایستگاهها در حال پخش برنامه اند. شنونده علاقمند به این سرویس دهنده وصل شده، ایستگاه مورد نظر را انتخاب کرده و محتویات ایستگاه را از طریق اتصال TCP دریافت می کند. بسته های نرم افزاری مختلفی برای مدیریت یک ایستگاه رادیوی اینترنتی (از ابتدا تا انتها) وجود دارد، که از میان آنها می توان به icecast (که نرم افزاری با استاندارد منبع باز است) اشاره کرد. سرویس دهنده های زیادی هم هستند، که در ازای دریافت مبالغ ناچیزی رادیوی شما را میزبانی می کنند.

۷-۵-۴ صدور IP

آن قدیمها شبکه تلفن عمومی بیشتر برای صدا بکار می رفت، و کمی هم داده روی آن منتقل می شد. اما ترافیک داده روز به روز افزایش یافت، تا اینکه در سال ۱۹۹۹ ترافیک داده و صدا مساوی شد (از آنجائیکه صدا روی ترانک های شبکه بصورت دیجیتال منتقل می شود، می توان آنرا هم با بیت/ثانیه اندازه گرفت). در سال ۲۰۰۲ ترافیک داده بسیار بیشتر از ترافیک صدا بود، و رشد نمایی آن همچنان ادامه دارد (در حالیکه ترافیک صدا رشد ثابتی معادل ۵٪ در سال دارد).

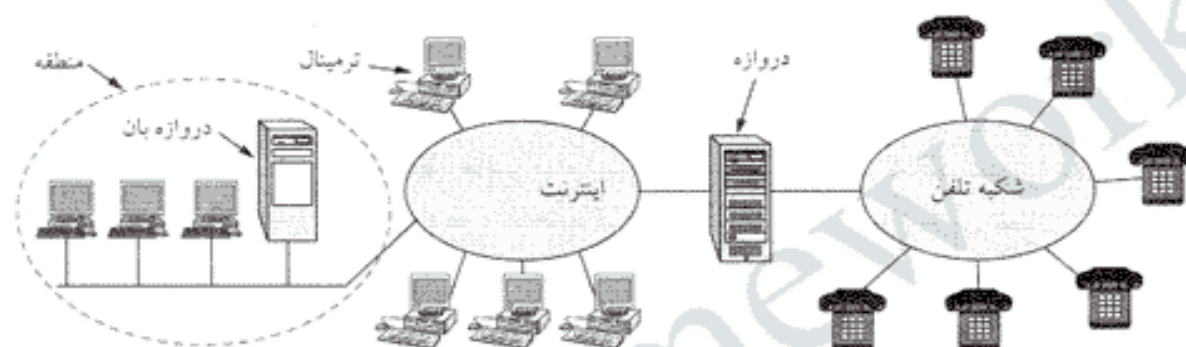
یکی از نتایج این وضعیت آن بود که بسیاری از اپراتورهای شبکه های سوئیچینگ بسته ای علاقمند شدند تا صدا را هم روی شبکه های داده منتقل کنند. فشاری که این کار به شبکه های داده می آورد چندان مهم نیست، چون این شبکه ها برای انتقال حجم عظیمی از اطلاعات طراحی شده اند. صورت حساب تلفن اغلب افراد بسیار بیشتر از صورت حساب اینترنت آنهاست، به همین دلیل اپراتورهای شبکه داده تلفن اینترنتی (Internet telephony) را یکی از راههای کسب سود سرشار ارزیابی کردند (بدون اینکه نیاز به سرمایه گذاری زیادی داشته باشد). بدین ترتیب بود که تلفن اینترنتی (یا همان صدور IP - voice over IP) متولد شد.

H.323

یک چیز از همان اول همه روشن بود: اگر هر کسی ساز خودش را بزند، این سیستم هرگز کار نخواهد کرد. برای اجتناب از چنین وضعیتی، تعدادی از شرکتهای علاقمند تحت نظارت ITU گرد آمدند، تا استاندارد برای تلفن اینترنتی وضع کنند. در سال ۱۹۹۶، ITU توصیه نامه H.323 را با عنوان «سیستمها و تجهیزات تلفن بصری برای شبکه های محلی بدون تضمین کیفیت سرویس» ارائه کرد (این نامگذاری ها فقط از شرکتهای تلفن برمی آید!).

این استاندارد در سال ۱۹۹۸ مورد بازنگری قرار گرفت، و همین ویرایش H.323 بود که اولین سیستم تلفن اینترنتی بر مبنای آن ساخته شد.

H.323 بیش از آن که یک پروتکل خاص باشد، تعریفی است از معماری تلفن اینترنتی. این استاندارد بجای آن که خودش چیزی را تعریف کند، به تعدادی استانداردهای جانبی برای کُد کردن صدا، برقراری تماس، سیگنالینگ، انتقال داده، و زمینه های دیگر ارجاع کرده است. مدل کلی استاندارد H.323 را در شکل ۷-۶۴ ملاحظه می کنید. در قلب این مدل یک دروازه (gateway) قرار دارد، که اینترنت را به شبکه تلفن وصل می کند. این مدل در سمت اینترنت به زبان پروتکل های H.323 صحبت می کند، و در سمت شبکه تلفن به زبان پروتکل های PSTN. در این مدل به دستگاه های مخابراتی ترمینال (terminal) گفته می شود. هر شبکه محلی می تواند دارای یک دروازه بان (gatekeeper) باشد، که ارتباطات آن منطقه (zone) را کنترل می کند.



شکل ۷-۶۴. مدل H.323 برای تلفن اینترنتی.

یک شبکه تلفن به تعدادی پروتکل نیاز دارد، که اولین آنها پروتکلی برای کُد کردن صدا است. سیستم PCM، که در فصل ۲ بررسی کردیم، در توصیه نامه ITU G.711 تعریف شده است. این سیستم هر کانال صوتی را با نرخ نمونه برداری 8000 samples/sec (با نمونه های ۸ بیتی) کُد کرده و یک خروجی فشرده نشده 64 kbps بدست می دهد. تمام سیستم های H.323 باید از G.711 پشتیبانی کنند، ولی پروتکل های فشرده سازی صدای دیگر را هم می توان بکار برد (اگر چه الزامی نیست). این پروتکلها از الگوریتم های مختلف فشرده سازی استفاده می کنند، و نسبت های متغیری از «کیفیت/پهنای باند» بدست می دهند. برای مثال، G.723.1 یک بلوک ۲۴۰ نمونه ای (۲۴۰ بیتی - که معادل 30 msec صداست) را گرفته و با استفاده از کُد کردن پیشگویانه (predictive coding) به ۲۴ یا ۲۰ بایت تقلیل می دهد. نرخ خروجی این الگوریتم 6.4 kbps یا 5.3 kbps (ضریب کاهش ۱/۱۰ یا ۱/۱۲) می باشد، ضمن اینکه کیفیت صدا را هم تا حد بسیار خوبی حفظ می کند. کُدک های دیگری نیز برای کُد کردن صدا وجود دارند. از آنجائیکه الگوریتم های فشرده سازی مختلفی وجود دارند، ترمینالها باید بتوانند بطریقی پروتکل فشرده سازی بکار رفته را بین خود مشخص کنند. این کار بر عهده پروتکل H.245 گذاشته شده است. این پروتکل همچنین نرخ بیت اتصال را تعیین می کند. برای کنترل کانال های RTP هم به پروتکل RTCP نیاز داریم. همچنین پروتکلی می خواهیم که کارهایی از قبیل برقراری یا قطع اتصال، ارسال بوق آزاد تلفن، تأمین زنگ های تلفن (در حالت های مختلف)، و مانند اینها را انجام دهد. برای این کار از پروتکل ITU Q.931 استفاده می کنیم. ترمینالها برای ارتباط با دروازه بان (اگر وجود داشته باشد) هم به یک پروتکل نیاز دارند، که H.225 را برای این منظور بکار می بریم. این پروتکل کانال بین PC و دروازه بان را، که کانال RAS (Registration/Admission/Status) نام دارد، کنترل می کند. این کانال اجازه می دهد تا ترمینال به منطقه وارد شده یا آنرا ترک کند، پهنای باند را درخواست

کرده یا پس بدهد، وضعیت خود را به روز در آورد، و بسیاری کارهای دیگر. بالاخره، پروتکلی می‌خواهیم برای انتقال داده‌ها، و برای این منظور از RTP (که مدیریت آن بر عهده RTCP است) استفاده می‌کنیم. موقعیت پروتکل‌های مختلف H.323 را در شکل ۶۵-۷ ملاحظه می‌کنید.

صدا	کنترل			
G.7xx	RTCP	H.2250 (RAS)	Q.931 (سیگنالینگ تماس)	H.245 (کنترل تماس)
RTP				
UDP			TCP	
IP				
پروتکل پیوند داده				
پروتکل لایه فیزیکی				

شکل ۶۵-۷. پشته پروتکل H.323.

برای اینکه ببینید همه این پروتکل‌ها چگونه با هم کار می‌کنند، ترمینالی (که یک PC در شبکه‌ای یا دروازه‌بان است) را در نظر بگیرید که می‌خواهد با تلفن راه دور تماس بگیرد. این PC ابتدا باید دروازه‌بان LAN را پیدا کند، بنابراین یک بسته UDP «کشف دروازه‌بان» روی پورت 1718 در تمام شبکه پخش می‌کند. وقتی دروازه‌بان پاسخ داد، PC آدرس IP آنرا یاد می‌گیرد. اکنون PC باید خود را به دروازه‌بان شناساند، که برای این منظور یک پیام RAS (در یک بسته UDP) به آن می‌فرستد (مرحله ثبت نام - Registration). بعد از آنکه دروازه‌بان شبکه PC را قبول کرد، PC یک پیام پذیرش RAS فرستاده (مرحله پذیرش - Admission)، و تقاضای پهنای باند می‌کند. فقط بعد از اختصاص پهنای باند از سوی دروازه‌بان است، که تماس تلفنی می‌تواند شروع شود. این مرحله برای تضمین حداقل کیفیت سرویس صورت می‌گیرد، تا دروازه‌بان بتواند تعداد تماس‌های همزمان را کنترل کرده و از فشار بیش از حد به خطوط خروجی جلوگیری کند.

پس از آن، PC برای شروع تماس تلفنی یک اتصال TCP به دروازه‌بان برقرار می‌کند. از آنجائیکه برای برقراری تماس به پروتکل‌های شبکه تلفن نیاز است و این پروتکل‌ها هم اتصال-گرا هستند، فقط از TCP می‌توان برای این منظور استفاده کرد. از طرف دیگر، چون در شبکه تلفن چیزی شبیه RAS وجود ندارد، در انتخاب نوع پروتکل آن قسمت (UDP یا TCP) آزادی عمل داریم؛ و بعلاوه کمتر بودن سرباره UDP، طراحان H.323 آنرا انتخاب کرده‌اند.

پس از گرفتن پهنای باند لازم، PC می‌تواند یک پیام Q.931 SETUP روی اتصال TCP بفرستد، و شماره تلفن مقصد (یا آدرس IP و شماره پورت، اگر مقصد یک کامپیوتر باشد) را به دروازه‌بان بدهد. دروازه‌بان هم برای تصدیق دریافت درخواست کاربر، یک پیام Q.931 CALL PROCEEDING به آن برمی‌گرداند. سپس، دروازه‌بان پیام SETUP را به دروازه هدایت می‌کند.

دروازه، که نیمی کامپیوتر و نیمی سوئیچ تلفن است، شماره تلفن خواسته شده را می‌گیرد. ایستگاه پایانی که تلفن در آن قرار دارد، به شماره مزبور زنگ می‌زند و یک پیام Q.931 ALERT هم به PC مبدأ می‌فرستد تا نشان دهد که زنگ خوردن شروع شده است. وقتی طرف مقابل گوشی را برمی‌دارد، ایستگاه پایانی یک پیام Q.931 CONNECT به مبدأ برمی‌گرداند تا بگوید که ارتباط برقرار شده است.

بعد از برقراری تماس دروازه بان دیگر کاری ندارد و از مدار خارج می شود، ولی دروازه همچنان به کار خود ادامه خواهد داد. بسته های بعدی دروازه بان را دور زده، و مستقیماً به آدرس IP دروازه فرستاده می شوند. در این نقطه دیگر فقط یک کانال ارتباطی معمولی بین مبدأ و مقصد وجود دارد، که چیزی نیست جز اتصال لایه های فیزیکی برای انتقال بیت ها، بدون هیچ اطلاعی از ماهیت واقعی کار.

در این مرحله برای تعیین پارامترهای تماس تلفنی از پروتکل H.245 استفاده می شود. این پروتکل از کانالهای H.245 استفاده می کند، که همیشه باز هستند. هر دو طرف ابتدا روی این کانال مقدمات خود را اعلام می کند، برای مثال نوع تماس (مثلاً، ویدئو - چون H.323 قادر به انتقال ویدئو نیز هست - یا تماس کنفرانسی)، نوع کدک، و غیره. همین که دو طرف از تواناییهای یکدیگر با خبر شدند، دو کانال یک طرفه داده (با تعیین نوع کدک و سایر پارامترها) بین آنها ایجاد می شود. از آنجائیکه امکان دارد قابلیت های دو طرف یکسان نباشد، کانالهای رفت و برگشت می توانند کاملاً متفاوت باشند. بعد از پایان مرحله مذاکره بر سر پارامترها، انتقال داده با استفاده از RTP می تواند شروع شود، که برای مدیریت آن از RTCP کمک می گیریم (این پروتکل نقش مهمی در کنترل ازدحام دارد). اگر ویدئو نیز وجود داشته باشد، سنکرون کردن صدا و تصویر هم بر عهده RTCP خواهد بود. انواع کانالهای ممکن را در شکل ۶۶-۷ ملاحظه می کنید. در پایان تماس نیز، برای قطع ارتباط از سیگنالهای Q.931 استفاده می شود.



شکل ۶۶-۷. کانال های منطقی بین دو طرف تماس.

بعد از قطع ارتباط، PC تماس گیرنده یک پیام RAS دیگر به دروازه بان می فرستد تا پهنای باند اختصاص داده شده را به آن پس بدهد (یا تماس دیگری بپذیرد).

درباره کیفیت سرویس (Quality of Service - QoS)، که نقش اساسی در موفقیت VOIP دارد، هیچ حرفی نزدیم. علت آن است که QoS جزء وظایف H.323 نیست. اگر شبکه موجود بتواند (با استفاده از تکنیکهای فصل ۵) ارتباطی پایدار و بدون لرزش بین PC تماس گیرنده و دروازه برقرار کند، QoS تماس خوب خواهد بود؛ در غیر اینصورت، خیر. شبکه تلفن از PCM استفاده می کند، و هیچوقت لرزش ندارد.

SIP - پروتکل آغازنشست

H.323 توسط ITU طراحی شد، و بسیاری از اینترنتی ها آنرا یک محصول تلفنی دیگر (بزرگ، پیچیده، و انعطاف ناپذیر) یافتند. به همین دلیل، IETF کمیته ای تشکیل داد تا روشی ساده تر و مدولارتر برای VOIP طراحی کند. ماحصل کار این کمیته SIP (پروتکل آغازنشست - Session Initiation Protocol) بود که در RFC 3261 تعریف شده است. در این پروتکل نحوه برقراری تماسهای تلفن اینترنتی، کنفرانس ویدئویی، و ارتباطات چندرسانه ای دیگر تشریح شده است. برخلاف H.323 که مجموعه ایست از چند پروتکل، SIP یک ماژول منفرد

است، ولی بگونه‌ای طراحی شده که بتواند با برنامه‌های اینترنتی موجود کار کند. برای مثال، در این پروتکل شماره‌های تلفن بصورت URL تعریف شده‌اند، و می‌توان آنها را روی صفحه‌های وب قرار داد؛ با کلیک کردن این لینک‌ها فرآیند برقراری تماس شروع می‌شود (درست به همان شکلی که لینکهای *mailto* کار می‌کنند). پروتکل SIP می‌تواند نشست‌های دو-طرفه (تماسهای تلفنی معمولی)، نشست‌های چند-طرفه (کنفرانس تلفنی - که تمام شرکت‌کنندگان می‌توانند حرف بزنند و حرف دیگران را بشنوند)، و نشست‌های چندپخش (یک گوینده و چندین شنونده) برقرار کند. در این نشست‌ها می‌توان صدا، تصویر و داده منتقل کرد (که این آخری به درد بازیهای گروهی روی اینترنت می‌خورد). البته SIP فقط شروع، کنترل و قطع نشست‌ها را بر عهده دارد، و انتقال داده را پروتکل‌های دیگر، مانند RTP/RTCP، انجام می‌دهند. از آنجائیکه SIP یک پروتکل لایه کاربرد است، می‌تواند روی TCP یا UDP نیز اجرا شود.

پروتکل SIP سرویسهای متنوعی ارائه می‌کند، از جمله یافتن تماس‌شونده (اگر در خانه نباشد)، تعیین قابلیت‌های تماس‌شونده، و مکانیزمهایی برای شروع و قطع ارتباط. در ساده‌ترین شکل، SIP یک نشست بین کامپیوتر تماس‌گیرنده و کامپیوتر تماس‌شونده برقرار می‌کند، و ما هم ابتدا همین فرآیند را بررسی خواهیم کرد. در SIP شماره تلفن‌ها با یک URL (که از پروتکل *sip* استفاده می‌کند) مشخص می‌شوند؛ مثلاً، *sip:ilse@cs.university.edu* شماره تلفن کاربری بنام *ilse* در ناحیه *cs.university.edu* است. در URL‌های SIP می‌توان از آدرسهای IPv4، آدرسهای IPv6، یا شماره تلفن واقعی استفاده کرد.

SIP یک پروتکل متنی (بر اساس مدل HTTP) است، که در آن نام متد در خط اول می‌آید، و بدنبال آن پارامترهای مورد نیاز فرستاده می‌شوند. بسیاری از سرآیندهای SIP از MIME گرفته شده‌اند، تا این پروتکل بتواند با برنامه‌های اینترنتی موجود کار کند. در شکل ۷-۶۷ شش تا از متدهای SIP، که در مشخصه اصلی آن تعریف شده‌اند، را ملاحظه می‌کنید.

متد	توضیح
INVITE	درخواست برقراری یک نشست
ACK	تایید شروع نشست
BYE	درخواست پایان نشست
OPTIONS	پرس و جو درباره قابلیت‌های میزبان
CANCEL	لغو درخواست معلق مانده
REGISTER	دادن اطلاعات مکان مشتری به سرویس دهنده

شکل ۷-۶۷. متدهای اصلی SIP.

برای برقراری یک نشست دو روش وجود دارد: برقراری اتصال TCP به تماس‌شونده و ارسال یک پیام *INVITE* به آن؛ ارسال پیام *INVITE* در یک بسته UDP. در هر دو حالت، سرآیند خط دوم و خطهای بعدی ساختار بدنه پیام (شامل قابلیت‌های تماس‌گیرنده، نوع رسانه، و فرمت‌های آن) را مشخص می‌کنند. اگر تماس‌شونده دعوت به تماس را قبول کند، یک کد پاسخ شبیه HTTP (کدهای سه رقمی، که در اینجا هم 200 نشانه قبول درخواست است) برمی‌گرداند. پس از این کد پاسخ، تماس‌شونده می‌تواند اطلاعات مربوط به خود (قابلیت‌ها، نوع رسانه، و فرمت‌ها) را بفرستد.

برای برقراری هر تماس سه پیام باید رد و بدل شود، بنابراین تماس‌گیرنده یک پیام *ACK* به تماس‌شونده برمی‌گرداند تا نشان دهد که پذیرش وی را دریافت کرده است؛ این پایان پروتکل برقراری نشست است.

هر یک از طرفین تماس می توانند برای قطع ارتباط پیام *BYE* بفرستند. وقتی طرف مقابل این پیام را تصدیق کرد، نشست خاتمه خواهد یافت.

هر ماشین می تواند برای تعیین قابلیت های خود (و اینکه اصلاً قادر به برقراری تماس *VOIP* هست یا نه) از متد *OPTIONS* استفاده کند، و معمولاً این کار را قبل از شروع نشست انجام می دهد.

پروتکل *SIP* از متد *REGISTER* برای یافتن افرادی که در محل مورد انتظار نیستند، استفاده می کند. هر ماشین پیام *REGISTER* خود را به یک سرویس دهنده مکان *SIP* (*SIP location server*)، که محل افراد را تعقیب می کند، می فرستد. وقتی کسی می خواهد با شما تماس بگیرد، می تواند به کمک این سرویس دهنده محل فعلی شما را پیدا کند. روش کار را در شکل ۷-۶۸ می بینید. در اینجا، تماس گیرنده پیام *INVITE* خود را به یک پروکسی فرستاده است (هدف از یکارگیری پروکسی برداشتن وظیفه یافتن مکان مخاطب از دوش کامپیوتر تماس گیرنده است). این پروکسی به کمک سرویس دهنده مکان محل فرد مورد نظر را یافته، و پیام *INVITE* را به آن می فرستد (تمام پیام های بعدی هم با واسطه همین پروکسی رد و بدل خواهند شد). پیام های *LOOKUP* و *REPLY* جزء *SIP* نیستند، و از هر پروتکلی (که با سرویس دهنده مکان مورد استفاده سازگار باشد) می توان برای این منظور استفاده کرد.



شکل ۷-۶۸. استفاده از سرویس دهنده پروکسی با *SIP*.

پروتکل *SIP* ویژگی های دیگری (از جمله انتظار مکالمه، پیگیری تماس، رمزنگاری، و احراز هویت) نیز دارد که در اینجا به آنها نخواهیم پرداخت. اگر دروازه مناسب بین اینترنت و شبکه تلفن وجود داشته باشد، *SIP* می تواند بین کامپیوتر و تلفن های معمولی نیز تماس برقرار کند.

مقایسه *SIP* و *H.323*

SIP و *H.323* شباهت ها و تفاوت های بسیاری دارند. هر دوی آنها می توانند تماس های مستقیم یا کنفرانسی بین کامپیوترها و تلفن های معمولی برقرار کنند. هر دو از مذاکره برای تعیین پارامترهای تماس، رمزنگاری، و پروتکل های *RTP/RTCP* پشتیبانی می کنند. فهرستی از شباهت ها و تفاوت های این دو را در شکل ۷-۶۹ ملاحظه می کنید.

با وجود شباهت های ظاهری، این دو پروتکل از نظر فلسفه وجودی بسیار با هم متفاوتند. *H.323* پروتکلی است بزرگ، پیچیده و استاندارد صنعت تلفن، که دقیقاً مشخص کرده چه چیزهایی مجازند و چه چیزهایی ممنوع. در این ره یافت همه چیز کاملاً مشخص و تعریف شده، و ارتباط بین سیستم های مختلف بسادگی امکان پذیر است. بهایی که برای این سادگی باید پرداخت، عبارتست از بزرگی، پیچیدگی و انعطاف ناپذیری، که انطباق این پروتکل با نیازهای آینده را دشوار کرده است.

SIP	H.323	آیتم
IETF	ITU	مستول طراحی
تا حد زیاد	بلی	سازگاری با PSTN
بلی	خیر	سازگاری با اینترنت
ماژولار	یکپارچه	معماری
فقط برقراری تماس	پشته پروتکل کامل	کامل بودن
بلی	بلی	مذاکره پارامترها
SIP روی TCP یا UDP	TCP روی Q.931	سیگنالینگ تماس
متنی	باینری	فرمت پیام
RTP/RTCP	RTP/RTCP	انتقال رسانه
بلی	بلی	تماس چند طرفه
بلی	بلی	کنفرانس چند رسانه ای
URL	شماره تلفن با میزبان	آدرس دهی
صریح یا بعد از انقضای زمان	صریح یا رها کردن TCP	پایان تماس
بلی	خیر	پیام رسانی فوری
بلی	بلی	رمزنگاری
۲۵۰ صفحه	۱۴۰۰ صفحه	اندازه استاندارد
متوسط	بزرگ و پیچیده	پیاده سازی
در حال رشد	کاربرد گسترده	وضعیت فعلی

شکل ۷-۶۹. مقایسه ای بین H.323 و SIP.

از طرف دیگر، SIP یک پروتکل سبک و معمولی اینترنتی است که با مبادله پیامهای متنی کار می کند، و سازگاری خوبی با پروتکل های موجود اینترنت دارد، ولی در زمینه ارتباط با پروتکل های سیگنالینگ تلفن چندان قوی نیست. از آنجائیکه IETF مدل VOIP خود را بصورت کاملاً مدولار تعریف کرده، این پروتکل انعطاف پذیری بالایی دارد و براحتی می توان آنرا با نیازهای آتی انطباق داد. نقطه ضعف این رهیافت مشکلات ناشی از ناسازگاری سیستمهاست، که IETF سعی کرده با برپایی سمینارهای متعدد و تبادل آراء بین سازندگان مختلف آنرا به حداقل برساند.

صداروی IP (VOIP) مبحثی نو و در حال تحول است. کتابهای متعددی در این زمینه نوشته شده، که از میان آنها می توان به (Colins, 2001; Davidson and Peters, 2000; Kumar et al., 2001; and Wright, 2001) اشاره کرد. در شماره May/June 2002 مجله *Internet Computing* نیز چندین مقاله در زمینه VOIP ارائه شده است.

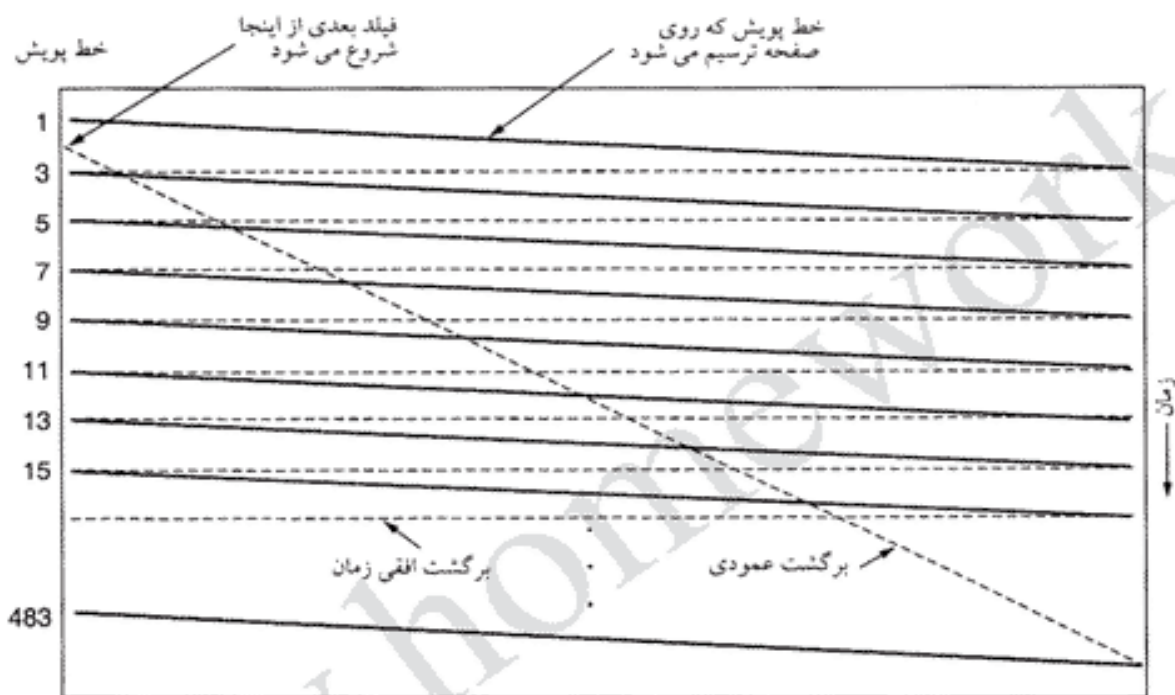
۶-۴-۷ مقدمه ای بر ویدئو

تا اینجا درباره گوش صحبت کردیم؛ اکنون وقت آن است که به چشم پردازیم (نگران نباشید بعد از آن دیگر سراغ بینی نخواهیم رفت!). یکی از ویژگیهای چشم انسان این است که وقتی تصویری روی شبکه می افتد، این تصویر برای چند هزارم ثانیه باقی می ماند. اگر یک تصویر را خط به خط رسم کنیم بطوریکه تمام این خطها در کمتر از یک پنجاهم ثانیه رسم شوند، چشم متوجه قطعه قطعه بودن تصویر نخواهد شد. تمام وسایل تصویری (از جمله تلویزیون) از این ویژگی برای تولید تصاویر متحرک استفاده می کنند.

سیستمهای آنالوگ

برای درک بهتر ویدئو، بهتر است از ساده ترین وسیله یعنی یک تلویزیون سیاه-سفید قدیمی کمک بگیریم.

برای تبدیل یک تصویر دو-بعدی به سیگنالی یک-بعدی (تابعی از ولتاژ بر حسب زمان)، دوربین تلویزیونی با شروع از بالای تصویر آن را به وسیله یک پرتو الکترونی از چپ برآست اسکن کرده و بتدریج پائین می آید، و در هر لحظه شدت نور را ثبت می کند. وقتی پرتو الکترونی به انتهای اسکن (که فریم نامیده می شود) رسید، دوباره به نقطه شروع برمی گردد. خروجی دوربین همین سیگنال (تابع شدت نور بر حسب زمان) است، و گیرنده با تکرار فرآیند اسکن دوباره تصویر را می سازد. روش اسکن کردن تصویر (که در دوربین و گیرنده یکسان است) در شکل ۷-۷۰ نشان داده شده است.



شکل ۷-۷۰. الگوی اسکن کردن تصویر در سیستم ویدئویی NTSC.

پارامترهای اسکن کردن تصویر از کشوری به کشور دیگر فرق می کند. در سیستمهای ویدئویی آمریکای شمالی، جنوبی و ژاپن از اسکن ۵۲۵ خطی، نسبت افقی-به-عمودی ۴:۳، و سرعت ۳۰ فریم بر ثانیه استفاده می شود. سیستمهای اروپایی اسکن ۶۲۵ خطی، نسبت افقی-به-عمودی ۴:۳، و سرعت ۲۵ فریم بر ثانیه را بکار می برند. در هر دو سیستم، برای چهارگوش کردن تصویر در لامپهای گرد قدیمی، چند خط از بالا و چند خط از پائین اصلاً نمایش داده نمی شود؛ در NTSC (که ۵۲۵ خط دارد) فقط ۴۸۳ خط نشان داده می شود، و در سیستمهای اروپایی (PAL/SECAM) ۵۷۶ خط (از ۶۲۵ خط). در مدت برگشت پرتو الکترونی به نقطه شروع این پرتو خاموش است (تاروی تصویر خط نیندازد)، و بسیاری از کشورها (بویژه در اروپا) از این فرصت برای ارسال اطلاعات متنی (اخبار، وضع هوا، اخبار ورزشی، قیمت سهام، و غیره) استفاده می کنند؛ این سرویس به تله تکست (TeleText) معروف است.

با اینکه ۲۵ فریم بر ثانیه برای نمایش تصاویر متحرک کافیست، اما برخی افراد (بویژه سالخورده گان) در این سرعت احساس می کنند تصاویر چشمک می زند (چون مدت دوام تصویر روی شبکیه آنها کمتر از افراد معمولی است). برای حل این مشکل، بجای بالا بردن نرخ فریم (که باعث هدر رفتن پهنای باند خواهد شد) از تکنیک متفاوتی استفاده می شود. در این تکنیک (بجای نمایش خطوط اسکن متوالی)، ابتدا خطوط فرد و سپس خطوط

زوج نمایش داده می شوند. بدین ترتیب، در هر بار حرکت پرتو الکترونی از بالا تا پایین صفحه فقط نیمی از یک فریم کامل نشان داده می شود؛ این نیم فریم را یک فیلد (field) می گویند. آزمایشات نشان داده که با وجود احساس چشمک زدن تصویر در سرعت ۲۵ فریم بر ثانیه، این احساس در سرعت ۵۰ فیلد بر ثانیه از بین می رود. این تکنیک خط-در-میانی (interlacing) نام دارد. به تلویزیونها یا ویدئوهای غیر خط-در-میانی پیشرونده (progressive) گفته می شود. توجه کنید که، با اینکه فیلمهای سینمایی با سرعت ۲۴ فریم بر ثانیه پخش می شوند، ولی در آنجا هر تصویر به مدت $1/24$ ثانیه بطور کامل دیده می شود.

ویدئوی رنگی از همان الگوی اسکن تک رنگ (سیاه-سفید) استفاده می کند، ولی در آن برای هر رنگ اصلی یک پرتو الکترونی مستقل وجود دارد که بطور هماهنگ عمل می کنند. رنگهای اصلی عبارتند از: قرمز، سبز، آبی (RGB). همانطور که می دانید، هر رنگی را می توان با برهم نهی خطی رنگهای اصلی (با شدت های مناسب) ایجاد کرد. با این حال، برای انتقال تصاویر رنگی روی یک کانال، باید سیگنالهای رنگ را در یک سیگنال مرکب (composite signal) ترکیب کرد.

وقتی تلویزیون رنگی اختراع شد، تکنیکهای متفاوتی برای نمایش رنگ وجود داشت، و هر کشور یکی از این تکنیکها را برای خود انتخاب کرد، که این منجر شد به سیستمهای تلویزیون رنگی ناسازگار (وضعیتی که همچنان ادامه دارد). (توجه داشته باشید که این قضیه هیچ ارتباطی با دعوای VHS، بتاماکس، و P2000 - که سیستمهای ضبط تصاویر رنگی هستند - ندارد.) اما در تمام کشورها یک الزام قانونی وجود داشت: تصاویر رنگی باید روی گیرنده های سیاه-سفید هم قابل دریافت باشند. به همین دلیل کُد کردن جداگانه سیگنالهای RGB (که ساده ترین روش ارسال سیگنالهای رنگی است) عملاً کنار گذاشته شد. (البته RGB کارآمدترین روش کُد کردن سیگنالهای رنگی نیست.)

اولین سیستم رنگی در آمریکا توسط «کمیته ملی استانداردهای تلویزیون» (National Television Standards Committee) استاندارد شد، و نام خود را هم به آن داد: NTSC. تلویزیون رنگی سالها بعد وارد اروپا شد، زمانی که تکنولوژی آن پیشرفت قابل توجهی کرده بود، به همین دلیل سیستمهای اروپایی از نظر رنگ و مقاومت در برابر نویز بسیار بهتر از سیستمهای آمریکایی بودند. در اروپا دو سیستم تلویزیون رنگی بکار گرفته شد: SECAM (Sequential Couleur Avec Memoire) که در فرانسه و اروپای شرقی بکار گرفته شد، و PAL

(Phase Alternating Line) که در بقیه اروپا از آن استفاده می شود. تفاوت کیفیت رنگ بین NTSC و PAL/SECAM چنان فاحش است، که NTSC را به استهزاء «هر دفعه به یک رنگ» (Never Twice the Same Color) هم می گویند.

برای آن که سیگنالهای RGB روی تلویزیونهای سیاه-سفید هم قابل دریافت باشند، در هر سه سیستم سیگنالهای RGB بصورت خطی - با نسبتهای مختلف - در یک سیگنال روشنایی (luminance) و دو سیگنال رنگ (chrominance) ترکیب می شوند. جالبست بدانید که حساسیت چشم انسان نسبت به سیگنالهای روشنایی بسیار بیشتر از سیگنالهای رنگ است، بنابراین دقت چندان در ارسال سیگنالهای رنگ لازم نیست. سیگنال روشنایی با همان فرکانس تلویزیونهای سیاه-سفید قدیمی پخش می شود، تا آنها هم بتوانند تصاویر را دریافت کنند؛ سیگنالهای رنگ نیز با فرکانسهای بالاتر (در باندی باریک) پخش می شوند. در برخی از تلویزیونها کنترلهایی بنام روشنایی، رنگ و غلظت رنگ وجود دارد، که در واقع این سیگنالها را کنترل می کنند. درک روشنایی و رنگ برای فهم تکنیکهای فشرده سازی ویدئو ضروری است.

در سالهای اخیر سر و صدای زیاد در اطراف HDTV (تلویزیون با وضوح بالا - High Definition TeleVision) بر پا شده است. این تلویزیونها با (تقریباً) دو برابر کردن تعداد خطوط اسکن، تصاویر بسیار بهتری تولید می کنند. آمریکا، اروپا و ژاپن همگی سیستمهای HDTV خاص خود را توسعه داده اند، که هیچکدام با دیگری سازگار نیست. (انتظار دیگری داشتید؟) اصول کار HDTV (اسکن، روشنایی، رنگ، و غیره) با سیستمهای موجود یکسان است، ولی نسبت افقی-به-عمودی در همه آنها 16:9 می باشد. این فرمت برای نمایش فیلمهای سینمایی (که روی فیلمهای 35 mm با نسبت 3:2 ضبط می شوند) بسیار مناسبتر است.

سیستمهای دیجیتال

ساده ترین راه نمایش ویدئوی دیجیتال عبارتست از توالی فریمهایی که هر کدام از تعدادی پیکسل (pixel) با آرایش مستطیلی تشکیل شده اند. هر پیکسل را می توان با یک بیت (سیاه یا سفید) نشان داد. کیفیت این سیستم شبیه ارسال تصویر با فکس است (فوق العاده وحشتناک!).

اگر از ۸ بیت برای نمایش هر پیکسل استفاده کنیم، می توانیم ۲۵۶ سایه خاکستری را نشان دهیم. کیفیت سیاه-سفید چنین سیستمی نسبتاً خوب است. در سیستمهای رنگی خوب، برای هر یک از رنگهای RGB از ۸ بیت جداگانه استفاده می شود، اگر چه تمام این سیستمها رنگها را هنگام ارسال به یک سیگنال مرکب تبدیل می کنند. استفاده از ۲۴ بیت برای هر پیکسل تعداد رنگهای ممکن را به حدود ۱۶ میلیون محدود می کند، ولی هیچ چشمی قادر به تشخیص این تعداد رنگ نیست (بیشتر که جای خود دارد). تصاویر رنگی دیجیتال با استفاده از سه پرتو اسکن کننده (یک پرتو برای هر رنگ) تولید می شوند. روش کار شبیه شکل ۷-۷ است، با این تفاوت که پیکسلهای منفرد جای خطوط پیوسته را گرفته اند.

در ویدئوی دیجیتال هم (مانند آنالوگ) برای ایجاد تصاویر متحرک باید حداقل ۲۵ فریم بر ثانیه نمایش داده شود. اما از آنجائیکه مانیتورهای امروزی قادرند تا ۷۵ تصویر در هر ثانیه نمایش دهند، نیازی به استفاده از تکنیک خط-در-میان نیست (و معمولاً هم استفاده نمی شود). وقتی بتوانیم هر فریم را سه بار پشت سر هم روی مانیتور رسم کنیم، دیگر چیزی بنام چشمک (flicker) وجود نخواهد داشت.

به تفاوت این دو مفهوم توجه کنید: نرمی حرکت به تعداد تصاویر مختلف در هر ثانیه بستگی دارد، در حالیکه چشمک به تعداد دفعات ترسیم یک تصویر روی صفحه وابسته است. در یک تصویر ثابت که با سرعت ۲۰ فریم بر ثانیه نمایش داده می شود، هیچ مشکلی بنام نرمی حرکت وجود ندارد، ولی همین تصویر دارای لرزش و چشمک است، چون هر تصویر برای مدت زمان کافی روی شبکه نقش نمی بندد. حال اگر فیلمی با سرعت ۲۰ فریم بر ثانیه نمایش پخش شود، ولی هر تصویر ۴ بار روی صفحه مانیتور رسم شود، تصاویر پخش شده بدون چشمک خواهند بود ولی حرکت نرم نیست.

اهمیت این دو پارامتر وقتی بیشتر روشن می شود که پهنای باند لازم برای ارسال تصاویر ویدئویی دیجیتال روی شبکه را در نظر بگیریم. مانیتورهای امروزی همچنان دارای نسبت افقی-به-عمودی 4:3 هستند، چون باید از لامپهای تلویزیونی که تولید انبوه شده اند، استفاده کنند. وضوح این مانیتورها اغلب 1024×768 ، 1280×960 یا 1600×1200 است. حتی کمترین وضوح (1024×768) با ۲۴ بیت بر پیکسل، و سرعت ۲۵ فریم بر ثانیه به پهنای باندی معادل 472 Mbps نیاز دارد. این پهنای باند معادل کاریر SONET OC-12 است، و فعلاً که قرار نیست به هر خانه یک خط OC-12 بکشند. اگر بخواهیم برای حذف چشمک این سرعت را دو برابر کنیم، که اوضاع خیلی خرابتر خواهد شد. البته برای حذف چشمک می توان فریمها را در تلویزیون ذخیره کرده و هر فریم را ۲ بار روی صفحه رسم کرد. اما مشکل اینست که تلویزیونهای معمولی حافظه ندارند، و اگر هم داشتند، سیگنالهای آنالوگ را نمی توان بدون تبدیل به دیجیتال در حافظه ذخیره کرد (و همه اینها یعنی هزینه اضافی).

۷-۴-۷ فشرده سازی ویدئو

تا اینجا فهمیدیم که به ارسال تصاویر ویدئویی فشرده نشده حتی نباید فکر کرد؛ تنها امیدمان اینست که بتوانیم تصاویر ویدئویی را به میزان زیاد فشرده کنیم. خوشبختانه کار تحقیقاتی گسترده در چند دهه گذشته باعث شد تا ارسال تصاویر ویدئویی فشرده روی شبکه ممکن شود. در این قسمت خواهید دید که فشرده کردن تصاویر ویدئویی چگونه انجام می شود.

تمام سیستمهای مبتنی بر داده های فشرده دو بخش دارند: یک الگوریتم فشرده سازی در مبدأ، و یکی برای عکس آن در مقصد. در ادبیات فنی به این الگوریتمها بترتیب گذ کردن (encoding) و دیگد کردن (decoding) می گویند؛ ما هم از همین اصطلاحات استفاده خواهیم کرد.

در الگوریتمهای فشرده سازی نوعی عدم تقارن ذاتی وجود دارد، که برای درک بهتر این فرآیند از اهمیت زیادی برخوردار است. اول اینکه، یک سند چندرسانه ای (مثلاً، یک فیلم سینمایی) فقط یک بار کُد شده و روی سرویس دهنده قرار داده می شود، ولی دیگد شدن آن هزاران بار (هر بار که یکی از کاربران می خواهد آنرا تماشا کند) اتفاق می افتد. این بدان معناست که الگوریتم کُد کردن می تواند کُند و متکی به سخت افزارهای گران قیمت باشد، مشروط باینکه الگوریتم دیگد کردن سریع بوده و نیازی به سخت افزارهای گران قیمت نداشته باشد. برای شرکت های پخش چندرسانه ای دو هفته کرایه کردن یک اَبَر کامپیوتر (و کُد کردن تمام فیلمهایی که دارند) چندان دور از منطق نیست، ولی نمی توان از کاربران انتظار داشت برای تماشای یک فیلم ویدئویی دو ساعت اَبَر کامپیوتر اجاره کنند. در بسیاری از سیستمهای فشرده سازی الگوریتم کُد کردن، به قیمت سریع و ساده شدن عملیات دیگد، پیچیده و وقت گیر طراحی شده اند.

از طرف دیگر، در سیستمهای چندرسانه ای بی درنگ (real-time multimedia) - مانند کنفرانس ویدئویی - کُند بودن فرآیند کُد کردن نیز غیر قابل قبول است. در اینجا کُد کردن باید در لحظه و بصورت بی درنگ انجام شود. در نتیجه، چنین سیستمهایی باید از الگوریتمهای متفاوتی استفاده کنند.

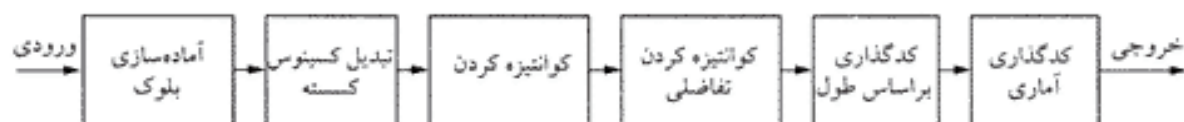
عدم تقارن دیگر در فرآیند کُد/دیگد اینست که آنها نباید الزاماً عکس یکدیگر باشند. این بدان معناست که در سیستمهای چندرسانه ای الزامی نیست فایلی که در مقصد دیگد می شود دقیقاً همان فایلی باشد که در مبدأ کُد شده، و می تواند چیزهایی را از دست بدهد. وقتی خروجی دیگد شده دقیقاً همان ورودی کُد شده نباشد، اصطلاحاً گفته می شود که سیستم تلفات دار (lossy) است. در مقابل، سیستمی که خروجی دقیقاً معادل ورودی باشد، بدون تلفات (lossless) نامیده می شود. سیستمهای تلفات دار از اهمیت زیادی برخوردارند، چون از دست دادن مقدار کمی از اطلاعات به فشرده سازی بسیار بالایی منجر می شود.

استاندارد JPEG

ویدئو عبارتست از توالی چند تصویر (بعلاوه صدا). اگر بتوانیم برای کُد کردن تصاویر ثابت الگوریتم مناسبی پیدا کنیم، می توانیم آنرا برای تصاویر متحرک (ویدئو) نیز بکار ببریم. امروزه الگوریتمهای خوبی برای فشرده کردن تصاویر ثابت وجود دارد، پس اجازه دهید از همانها شروع کنیم. استاندارد Joint Photographic Experts Group (تحت نظارت ITU، ISO و IEC) و چند سازمان دیگر) برای فشرده کردن تصاویر غیرگرافیکی (بویژه، عکس) توسعه داده شد. اهمیت این الگوریتم در آنجاست که مهمترین استاندارد فشرده سازی تصاویر متحرک یعنی MPEG، در واقع چیزی نیست جز کُد کردن فریمهای متوالی با JPEG (باضافه چند ویژگی دیگر برای فشرده کردن بین فریمها و تشخیص حرکت). JPEG در استاندارد ISO 10918 تعریف شده است.

الگوریتم JPEG دارای چهار حالت و چندین گزینه است (و در واقع بیشتر به یک لیست خرید می ماند، تا

الگوریتم فشرده سازی). اما آنچه که ما به آن علاقه داریم (و در شکل ۷-۷۱ می بینید)، حالت متوالی تلفات دار (lossy sequential) است. در اینجا برای سادگی بیشتر بحث، فقط به روش کد کردن تصاویر ویدئویی 24-bit RGB در JPEG توجه می کنیم، و جزئیات دیگر را نادیده می گیریم.



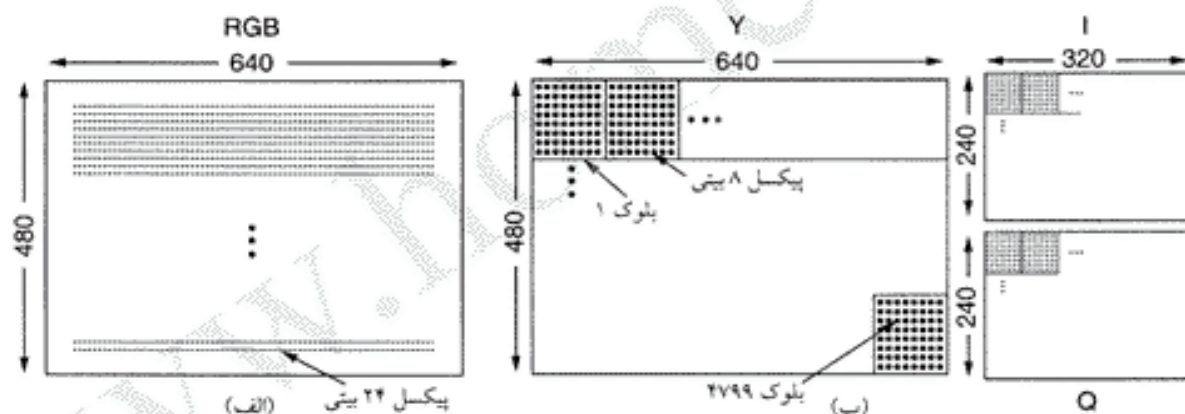
شکل ۷-۷۱. عملکرد JPEG در حالت متوالی تلفات دار.

مرحله اول کد کردن تصویر با JPEG، آماده سازی بلوک (block preparation) است. برای سادگی بحث فرض می کنیم که ورودی JPEG یک تصویر RGB 640×480 با وضوح 24 bits/pixel است (شکل ۷-۷۲ الف). از آنجائیکه فشرده سازی روشنایی و رنگ نتایج بهتری دارد، ابتدا سیگنال روشنایی، Y ، و سیگنالهای رنگ، I و Q ، تصویر را (برای سیستم NTSC) با استفاده از فرمولهای زیر محاسبه می کنیم:

$$Y = 0.30R + 0.59G + 0.11B$$

$$I = 0.60R - 0.28G - 0.32B$$

$$Q = 0.21R - 0.52G + 0.31B$$



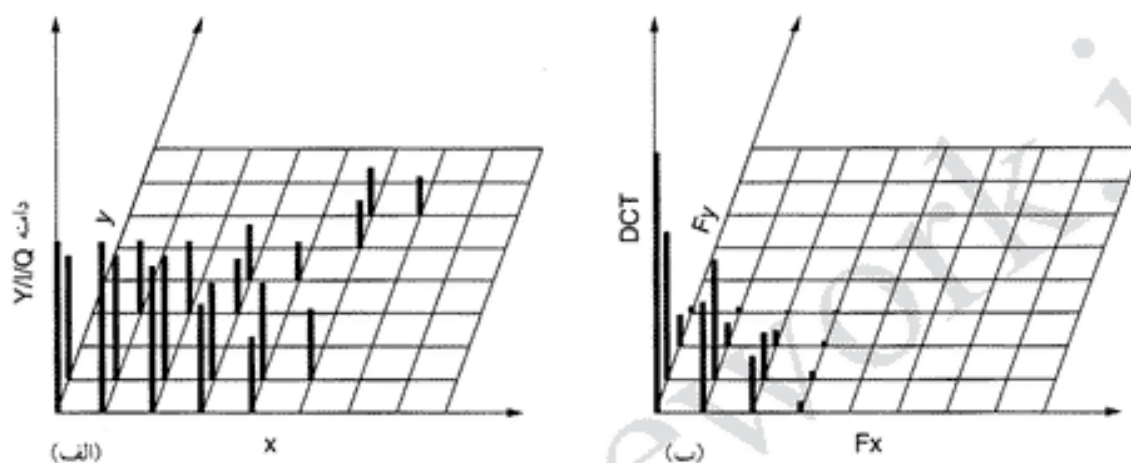
شکل ۷-۷۲. (الف) ورودی RGB. (ب) بعد از آماده سازی بلوک.

در PAL سیگنالهای رنگ I و Q نامیده می شوند و ضرایب متفاوتی دارند، ولی ایده اصلی همان است؛ SECAM هم که با هر دوی آنها فرق دارد.

برای Y ، I و Q ماتریسهای جداگانه ای (با مقادیر 0 تا 255) ایجاد می شود. سپس، تصویر به بلوکهای مربعی 4×4 پیکسلی تقسیم شده، و I و Q متوسط آنها محاسبه می شود (با این کار تصویر ورودی به 320×240 تبدیل می شود). این کاهش با تلفات همراه است، اما از آنجائیکه چشم انسان به روشنایی حساس تر است تا رنگ، متوجه آن نخواهد شد. ضریب کاهش تا اینجا ۱ به ۲ است (یعنی اندازه فایل نصف شده است). حال، از تمام عناصر هر سه ماتریس عدد ۱۲۸ کم می شود، که با اینکار 0 به عدد میانه در محدوده اعداد ماتریس ها تبدیل خواهد شد. و در آخر، تمام ماتریس ها به بلوکهای 8×8 تقسیم می شوند. با این کار ماتریس Y دارای 4800 بلوک، و دو ماتریس دیگر هر یک دارای 1200 بلوک خواهند بود (شکل ۷-۷۲ ب را ببینید).

در مرحله دوم JPEG، روی تمام 7200 ماتریس بصورت جداگانه تبدیل کسینوسی گسته (Discrete - DCT)

Cosine Transformation) انجام می شود. خروجی هر DCT یک ماتریس 8×8 از ضرایب DCT است، که عنصر $(0, 0)$ هر تبدیل DCT مقدار متوسط آن بلوک است. عناصر دیگر نشان می دهند که در هر فرکانس فضایی چه مقدار انرژی طیفی وجود دارد. از نظر تئوری، DCT یک الگوریتم بدون تلفات است، ولی گرد شدن اعداد در محاسبات اعشاری و مثلثاتی عملاً باعث مقداری اتلاف اطلاعات خواهد شد. معمولاً مقدار عناصر ماتریس با دور شدن از مبدأ مختصات $(0, 0)$ بسرعت تحلیل می روند (به شکل ۷-۷۳ نگاه کنید).



شکل ۷-۷۳. (الف) یکی از بلوکهای ماتریس Y . (ب) ضرایب DCT.

بعد از پایان DCT، JPEG وارد مرحله سوم یعنی کوانتیزه کردن (quantization) می شود، که در آن ضرایب کم اهمیت تر DCT دور انداخته می شوند. در این تبدیل تلفات دار، عناصر ماتریس 8×8 ضرایب DCT بر وزنی که از یک جدول (جدول کوانتیزه کردن) گرفته می شود، تقسیم می شوند. اگر تمام وزن ها 1 باشند، تبدیل هیچ کاری انجام نمی دهد؛ ولی اگر وزن ها خیلی از مبدأ دور باشند، فرکانسهای فضایی بالاتر دور انداخته می شوند. به مثال شکل ۷-۷۴ نگاه کنید. در این شکل سه جدول می بینید: ماتریس DCT اولیه، جدول کوانتیزه کردن، و جدولی که از تقسیم عناصر ماتریس DCT بر عناصر متناظر در جدول کوانتیزه کردن بدست آمده است. مقادیر جدول کوانتیزه کردن جزئی از استاندارد JPEG نیست، و هر برنامه تبدیل به فرمت JPEG باید خود آنرا داشته باشد (و بوسیله آن مقدار تلفات اطلاعات را کنترل کند).

ضرایب DCT								جدول کوانتیزه کردن								ضرایب کوانتیزه شده							
150	80	40	14	4	2	1	0	1	1	2	4	8	16	32	64	150	80	20	4	1	0	0	0
92	75	36	10	6	1	0	0	1	1	2	4	8	16	32	64	92	75	18	3	1	0	0	0
52	38	26	8	7	4	0	0	2	2	2	4	8	16	32	64	26	19	13	2	1	0	0	0
12	8	6	4	2	1	0	0	4	4	4	4	8	16	32	64	3	2	2	1	0	0	0	0
4	3	2	0	0	0	0	0	8	8	8	8	8	16	32	64	1	0	0	0	0	0	0	0
2	2	1	1	0	0	0	0	16	16	16	16	16	16	32	64	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	32	32	32	32	32	32	32	64	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	64	64	64	64	64	64	64	64	0	0	0	0	0	0	0	0

شکل ۷-۷۴. محاسبه ضرایب کوانتیزه DCT.

در مرحله چهارم، بجای عنصر $(0, 0)$ هر بلوک (گوشه چپ-بالا) تفاضل آن با همین عنصر از جدول قبل نوشته می شود. از آنجائیکه این عناصر مقدار متوسط هر بلوک هستند، اختلاف آنها زیاد نیست و عددی که بدست

می آید نسبتاً کوچک است. فقط روی این عنصر است که تفاضل محاسبه می شود، نه عناصر دیگر. به عنصر $(0, 0)$ هر جدول مؤلفه DC (و به سایر عناصر، مؤلفه AC) گفته می شود.

در مرحله پنجم تمام 64 عنصر جدول با روش گذردن run-length خطی می شوند. برای این کار از اسکن زیگزاگی جدول استفاده می شود (شکل ۷-۷۵)، چون در اسکن افقی عمودی 0 ها کنار هم قرار نمی گیرند. در مثال شکل ۷-۷۵، اسکن زیگزاگی باعث شده تا ۳۸ صفر متوالی در انتهای ماتریس بدست آید. این عدد را می توان بطور خلاصه «۳۸ صفر» نامید (و جدول را خیلی کوچک کرد).

150	80	20	4	1	0	0	0
92	78	18	8	1	0	0	0
26	19	13	2	1	0	0	0
3	2	2	1	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

شکل ۷-۷۵. روش خطی کردن عناصر ماتریس کوانتیزه شده.

پس از اجرای مرحله پنجم، لیستی از اعداد (در فضای تبدیل) بدست می آید که نماینده تصویر فشرده شده هستند. در مرحله آخر برای کوچک کردن هر چه بیشتر این لیست، به اعدادی که بیشتر تکرار شده اند کدهای کوچکتر داده می شود، و به اعداد با تکرار کمتر کدهای بلندتر (کدهای فم). (کدهای فم).

شاید JPEG پیچیده بنظر برسد، چون در واقع پیچیده هم هست. اما از آنجائیکه ضریب فشرده سازی در آن بسیار بالاست (نزدیک 20:1)، کاربرد گسترده ای پیدا کرده است. برای دیگد کردن تصاویر JPEG، الگوریتم بالا بصورت معکوس انجام می شود. JPEG تا حد زیادی متقارن است: دیگد کردن تقریباً به همان اندازه گد کردن وقت می برد. اما همانطور که خواهید دید، گدهایی هم هستند که بشدت نامتقارنند.

استاندارد MPEG

بالاخره به اصل مطلب رسیدیم: استانداردهای MPEG (Motion Picture Experts Group). این استانداردها از سال ۱۹۹۳ بصورت بین المللی برای فشرده کردن ویدئو بکار برده می شوند، چون می توانند صدا و تصویر را با هم فشرده کنند (و ویدئو هم ترکیبی است از صدا و تصویر). تا اینجا فشرده سازی صدا و تصویر ثابت را دیدید. پس اجازه دهید فشرده سازی ویدئو را بررسی کنیم.

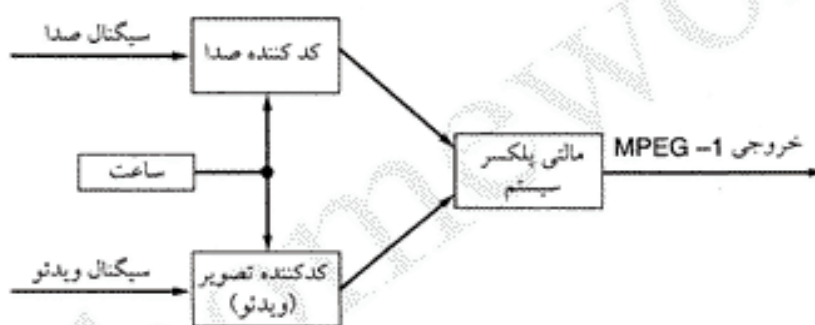
اولین استاندارد که نهایی شد، MPEG-1 بود (ISO 11172). هدف این استاندارد تولید خروجی با کیفیت ضبط ویدئو (وضوح 352×240 در سیستم NTSC) با نرخ انتقال 1.2 Mbps بود. ویدئوی 352×240 با رنگ 24 bit/pixel و سرعت 25 frame/sec به پهنای باند 50.7 Mbps نیاز دارد، پس تقلیل آن به 1.2 Mbps یعنی فشرده سازی 40:1، و این اصلاً کار ساده ای نیست. از MPEG-1 برای انتقال ویدئو روی کابل زوج تابیده (در مسافتهای نه چندان زیاد)، و ذخیره کردن فیلم روی CD-ROM استفاده می شود.

استاندارد بعدی این خانواده MPEG-2 بود (ISO 13818)، که برای فشرده سازی ویدئو با کیفیت پخش به

پهنای باند 4-6 Mbps (معادل پهنای باند پخش NTSC یا PAL) طراحی شده بود. بعدها وضوح تصویر در MPEG-2 افزایش یافت، و HDTV را هم در بر گرفت. این فرمت امروزه بسیار رواج دارد، و از آن در DVD و تلویزیون ماهواره ای دیجیتال استفاده می شود.

اصول کار در MPEG-1 و MPEG-2 یکسان است، و آنها فقط در جزئیات با هم فرق دارند. در واقع MPEG-2 همان MPEG-1 است، که ویژگیهای دیگری (از قبیل فرمت ها و روشهای کد کردن) به آن اضافه شده است. ما هم ابتدا MPEG-1 و سپس MPEG-2 را بررسی خواهیم کرد.

MPEG-1 سه قسمت دارد: صدا، ویدئو، و سیستم که دو قسمت قبلی را یکپارچه می کند (شکل ۷-۷۶ را ببینید). صدا و ویدئو بصورت جداگانه و مستقل کد می شوند، که این کار مشکل سنکرون کردن آنها در گیرنده را پیش می آورد. این مشکل با استفاده از یک ساعت سیستم 90-kHz (که وقت فعلی را به هر دو قسمت صدا و ویدئو می دهد) حل شده است. اینها اعداد ۳۳ بیتی هستند، بنابراین یک فیلم می تواند بدون هیچ مشکلی حتی ۲۴ ساعت طول بکشد (بدون اینکه مسئله صفر شدن تایمر پیش آید). این برچسبهای زمانی در خروجی گذشته هم نوشته می شوند، و گیرنده می تواند از آنها برای سنکرون کردن صدا و ویدئو استفاده کند.



شکل ۷-۷۶. سنکرون کردن صدا و ویدئو در MPEG-1.

اکنون اجازه دهید فشرده سازی ویدئو در MPEG-1 را بررسی کنیم. در هر فیلم دو نوع افزونگی وجود دارد: فضایی و موقتی؛ MPEG-1 از هر دوی آنها استفاده می کند. برای بکارگیری افزونگی فضایی، هر فریم بطور جداگانه با JPEG کد می شود. این رهیافت بیشتر در مواقعی بکار برده می شود که (علاوه بر پخش فیلم) به تک تک فریمها نیز (برای کارهایی مانند تدوین فیلم) احتیاج داشته باشیم. با این روش می توان به پهنای باند 8-10 Mbps دست یافت.

برای رسیدن به فشردگی بیشتر می توان از این واقعیت که در یک فیلم فریمهای متوالی تقریباً یکسان هستند، بهره گرفت. البته مقدار کاهش حاصله از این ویژگی آنچنان که در نگاه اول بنظر می رسد زیاد نیست، چون در اغلب فیلمها هر ۳ یا ۴ ثانیه (بطور متوسط ۷۵ فریم) صحنه بکلی عوض می شود. اما همین توالیهای ۷۵ فریمی تقریباً یکسان هم به کاهش قابل ملاحظه ای (در مقایسه با JPEG) منجر خواهد شد.

در صحنه هایی که زمینه یا دوربین ثابت است و هنرپیشه ها حرکت کمی دارند، تقریباً تمام پیکسلها در فریمهای متوالی شبیه هم هستند. در این حالت، محاسبه تفاضل دو فریم و اجرای الگوریتم JPEG روی این تفاضل، بخوبی کار خواهد کرد. اما در صحنه هایی که حرکت دوربین زیاد است، این تکنیک به هیچ دردی نخواهد خورد، و باید راهی پیدا کرد که حرکت شدید صحنه را جبران کند. این دقیقاً همان کاری است که MPEG انجام می دهد؛ و تفاوت MPEG و JPEG نیز در همین جاست.

در خروجی MPEG-1 چهار نوع فریم می تواند وجود داشته باشد:

۱. فریمهای I (Intracoded): تصاویر ثابت و مستقل JPEG.
۲. فریمهای P (Predictive): تفاوت بلوک-به-بلوک با فریم قبلی.
۳. فریمهای B (Bidirectional): تفاوت های بین فریم قبلی و بعدی.
۴. فریمهای D (DC-coded): متوسط بلوک ها برای جلوگیری از افتادن سریع فیلم (FF).

فریمهای I تصاویر ثابت با فرمت JPEG هستند، که همچنین از وضوح کامل روشنایی و وضوح نیمه کامل در هر محور رنگ استفاده می کنند. سه دلیل برای قرار دادن فریمهای I در استریم خروجی وجود دارد. اول اینکه، از MPEG-1 در سیستمهای چندپخش (multicast)، که تعداد زیادی بیننده مستقل دارند، نیز استفاده می شود. اگر هر فریم به فریم قبلی (همینطور تا اولین فریم فیلم) وابسته باشد، کسی که وسط فیلم تلویزیون خود را روشن کرده (و اولین فریم را از دست داده)، هیچ چیز نمی تواند ببیند (چون در واقع مبنایی برای دیگد کردن فریمها ندارد). دوم، اگر یکی از فریمها خراب شود، دیگر نمی توان فریمهای بعدی را دیگد کرد. سوم اینکه، بدون فریمهای I کار گیرنده برای جلو یا عقب رفتن سریع (FF یا Rewind) بسیار مشکل خواهد شد، چون مجبور است تک تک فریمها را دیگد کند. به این دلایل، در هر ثانیه یک یا دو فریم I در خروجی قرار داده می شود.

بر خلاف فریمهای I، فریمهای P فقط اختلاف بین فریمها را کد می کنند. فریمهای P از ایده ماکروبلوک (macroblock) - ماتریسهای 16×16 پیکسل در فضای روشنایی، و ماتریسهای 8×8 پیکسل در فضای رنگ - استفاده می کنند. هر ماکروبلوک با جستوی بلوک مشابه در فریم قبلی (برای یافتن تشابه یا اختلاف) کد می شود. شکل ۷-۷۷ نمونه ای که در آن فریمهای P بکار می آیند، را نشان می دهد. در اینجا سه فریم متوالی می بینید که زمینه صحنه در آنها یکسان است، و فقط جای هنرپیشه عوض شده است. در این فریمها، ماکروبلوک های زمینه صحنه دقیقاً یکسان مانده، و فقط ماکروبلوک های مربوط به هنرپیشه عوض شده و باید کد شود.



شکل ۷-۷۷. سه فریم متوالی.

استاندارد MPEG-1 هیچ حرفی درباره روش جستجو، عمق جستجو، و یا مفهوم یکسان بودن، نمی زند - تمام اینها برعهده نویسنده برنامه گذاشته شده است. برای مثال، در یک برنامه ممکنست جستجوی ماکروبلوک در مکان فعلی در فریم قبلی، و در تمام آفست های $\pm \Delta x$ (در جهت x) و $\pm \Delta y$ (در جهت y) صورت گیرد، و تعداد نقاط یکسان از نظر روشنایی محاسبه شود. مکانی با مقدار بیشتر (مشروط باینکه از یک آستانه تعریف شده بالاتر باشد) بعنوان برنده انتخاب می شود. در غیراینصورت، گفته می شود که ماکروبلوک گم شده است. البته الگوریتمهای بسیار بهتری نیز برای این منظور وجود دارد.

اگر ماکروبلوک پیدا شود، اختلاف (روشنایی و رنگ) آن با فریم قبلی محاسبه شده، و سپس با الگوریتم JPEG (تبدیل DCT، کوانتیزه کردن، کد run-length، و کد هافمن) کد می شود. مقدار این ماکروبلوک در استریم خروجی بصورت بردار حرکت آن (شامل مقدار و جهت جابجایی) ثبت می شود. اگر یک ماکروبلوک در فریم قبلی رجوع نداشته باشد، بصورت عادی (فریم I) کد می شود.

این الگوریتم بشدت نامتقارن است. برنامه آزاد است هر مکان قابل قبولی در فریم قبلی را که بخواهد (برای

یافتن ماکروبلوک موردنظر) امتحان کند. با این روش استریم MPEG-1 می تواند به ضریب فشرده سازی بالایی دست پیدا کند، ولی در ضمن زمان کُد کردن هم بالا خواهد رفت. همانطور که می توان حدس زد، این رهیافت برای کُد کردن فیلمهای کتابخانه ای مناسب است، ولی اصلاً بدرد ویدئوکنفرانس نمی خورد.

برنامه نویسی در اتخاذ تصمیم برای تعریف مفهوم «ماکروبلوک یکسان» نیز آزاد است. این آزادی دست برنامه نویسی را برای انتخاب نقطه تعادل بین سرعت و کیفیت (در عین اطمینان از اینکه خروجی همواره با MPEG-1 سازگار است) باز می گذارد. در هر حال، خروجی یا ماکروبلوک JPEG شده است، یا JPEG شده اختلاف آن با فریم قبلی.

تا اینجا، دیگد کردن MPEG-1 ساده است. دیگد کردن فریمهای I که هیچ فرقی با تصاویر ثابت JPEG ندارد. برای دیگد کردن فریمهای P، برنامه فریم قبلی را در یک بافر ذخیره کرده و در بافر دیگر فریم جدیدی بر اساس ماکروبلوکهای کامل و ماکروبلوکهای اختلافها با فریم قبلی می سازد. فریم جدید بلوک به بلوک ساخته می شود. فریمهای B شبیه فریمهای P هستند، با این تفاوت که فریم مبنای آنها می تواند (بجای فریم قبلی) فریم بعدی نیز باشد. این آزادی کمک زیادی به جبران حرکت صحنه ها می کند، بویژه وقتی اشیاء روی صحنه از جلو یا پشت اشیاء دیگر عبور می کنند. برای دیگد کردن فریمهای B، برنامه باید سه فریم را در آن واحد در حافظه داشته باشد: فریم قبلی، فریم فعلی، و فریم بعدی. با آنکه فریمهای B ضریب فشرده سازی را بالا می برند، همه برنامه های MPEG از آنها پشتیبانی نمی کنند.

فریمهای D فقط برای نمایش تصویری با وضوح پائین هنگام جلو یا عقب رفتن سریع فیلم (FF یا Rewind) بکار برده می شوند. دیگد کردن MPEG-1 با سرعت پخش معمولی باندازه کافی دشوار هست، انجام همین کار با سرعت ۱۰ برابر که دیگر جای خود دارد. به همین دلیل در سرعتهای بالا از فریمهای D (که وضوح کمتری دارند) استفاده می شود. هر فریم D در واقع فقط مقدار متوسط هر بلوک (بدون هر گونه کُد کردن اضافی) است، که نمایش آنرا در زمان واقعی آسان می کند. با این ویژگی می توان یک فیلم را با سرعت زیاد بدنبال صحنه موردنظر جستجو کرد. فریمهای D معمولاً درست قبل از فریمهای I قرار داده می شوند، تا وقتی حرکت سریع فیلم متوقف شد، بتوان نمایش را با سرعت معمولی ادامه داد.

اجازه دهید بعد از MPEG-1، سراغ MPEG-2 برویم. روش کُد کردن MPEG-2 اساساً شبیه MPEG-1 است، با این تفاوت که MPEG-2 از فریمهای D پشتیبانی نمی کند. همچنین، در DCT بجای ماتریسهای 8×8 از بلوکهای 10×10 استفاده می کند، که با این کار تعداد ضرایب ۵۰ درصد بیشتر شده و کیفیت بالاتر می رود. از آنجائیکه MPEG-2 برای پخش تلویزیونی و DVD طراحی شده، از تصاویر خط-در-میان و پیشرونده پشتیبانی می کند (در حالیکه MPEG-1 فقط از تصاویر پیشرونده پشتیبانی می کند). بین این دو استاندارد تفاوتهای کوچک دیگری نیز وجود دارد.

بجای یک وضوح ثابت، MPEG-2 از چهار وضوح پشتیبانی می کند: کم (352×240)، اصلی (720×480)، بالا-۱۴۴۰ (1440×1152)، و بالا (1920×1080). وضوح 352×240 برای دستگاههای VCR (و سازگاری با MPEG-1) در نظر گرفته شده است. وضوح اصلی برای پخش NTSC بکار می رود. دو وضوح دیگر برای HDTV در نظر گرفته شده اند. MPEG-2 در وضوحهای بالا معمولاً با سرعت 4-8 Mbps اجرا می شود.

۸-۴-۷ پخش فیلم بر حسب تقاضا

پخش فیلم بر حسب تقاضا (Video on demand - VOD) چیزی شبیه مغازه کرایه فیلمهای ویدئویی است. در آنجا مشتری یکی از فیلمهای موجود را انتخاب کرده و برای تماشا به منزل می برد. ولی در اینجا نیازی به مراجعه به

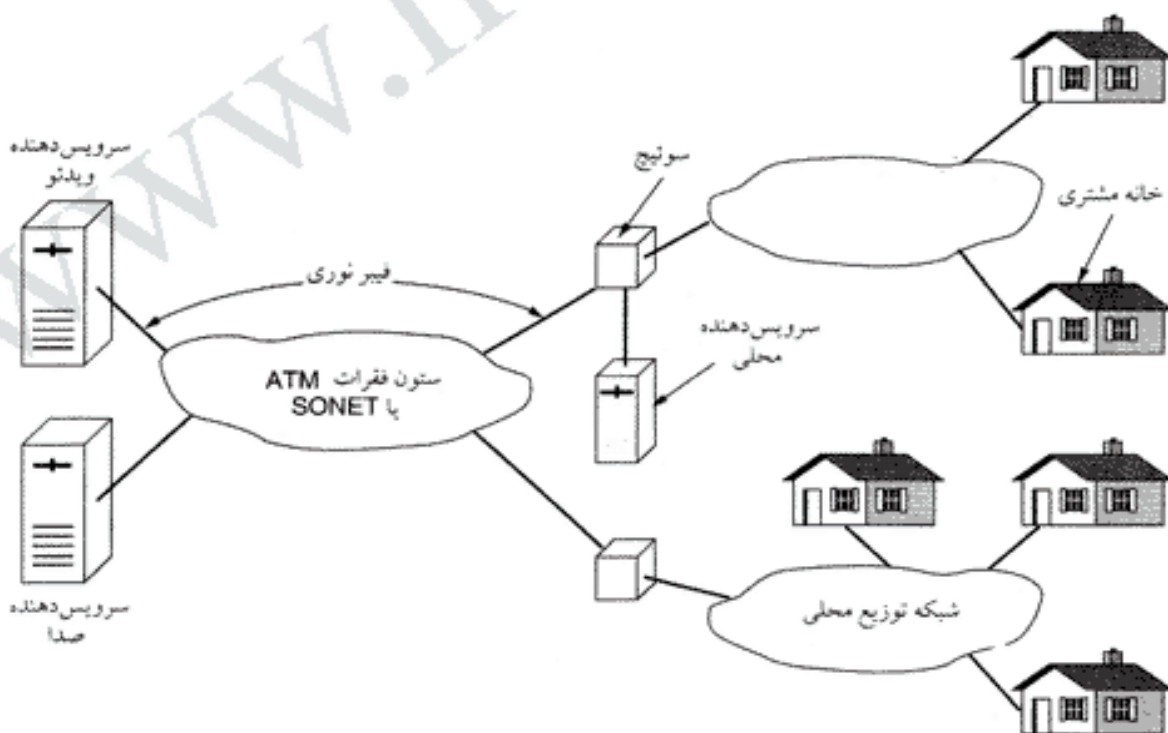
مغازه نیست و انتخاب فیلم از طریق دستگاه کنترل از راه دور تلویزیون انجام شده، و پخش آن هم بلافاصله شروع می شود. لازم به گفتن نیست که، پیاده سازی VOD از تعریف آن کمی مشکلتر است.

آیا پخش فیلم بر حسب تقاضا شبیه کرایه فیلم ویدئویی است، یا انتخاب کانال در تلویزیونهای کابلی؟ پاسخ این سوال تبعات فنی مهمی دارد. وقتی یک فیلم ویدئویی کرایه می کنید، می توانید وسط فیلم آنرا متوقف کرده، و بعد از خوردن یک فنجان چای یا جواب دادن به تلفن، ادامه آنرا از همانجایی که متوقف شده بود، تماشا کنید. اما بینندگان کانالهای تلویزیونی نمی توانند چنین کاری بکنند.

اگر VOD بخواهد رقابت مؤثری با مغازه های کرایه فیلم داشته باشد، باید به مشتری اجازه دهد فیلم را بدوخواه خود متوقف کرده، و یا عقب و جلو ببرد. چنین کاری مستلزم آن است که برای هر مشتری یک کپی اختصاصی از فیلم پخش شود.

اما اگر VOD را فقط نوعی تلویزیون پیشرفته فرض کنیم، آنگاه کافیسیت فیلمهای پُرطرفدار را در فواصل ۱۰ دقیقه ای (و بدون توقف تا آخر) پخش کنیم. در این حالت، مشتری برای دیدن این فیلم فقط کافیسیت (حداکثر) ۱۰ دقیقه صبر کند. با اینکه متوقف کردن چنین فیلمی وجود ندارد، اما اگر وسط آن کاری برایتان پیش آمد، کافیسیت به کانال دیگری (که ۱۰ دقیقه عقبتر است) رفته و فیلم را از جایی که از دست داده بودید، تماشا کنید. (تکرار چیز خوبی نیست، ولی مسلماً بهتر از ندیدن قسمتهایی از فیلم است.) این روش را «تقریباً VOD» می نامند. هزینه پیاده سازی چنین سیستمی بسیار کمتر است، چون می توان هر فیلم را برای عده زیادی پخش کرد. تفاوت VOD و «تقریباً VOD» مثل مسافرت با اتومبیل شخصی و اتوبوس است.

تماشای فیلم با VOD (یا «تقریباً VOD») یکی از امکانات بالقوه شبکه های پهن باند امروزی است، که در شکل ۷-۷۸ یکی از مدل های متداول آنرا می بینید. در مرکز این سیستم یک شبکه با پهنای باند زیاد (ملی یا بین المللی) قرار دارد، که هزاران شرکت توزیع کننده (شرکتهای تلفن، تلویزیون کابلی و مانند آنها) به آن متصلند.



شکل ۷-۷۸. یک سیستم پخش فیلم بر حسب تقاضا (VOD).

انشعابات این سیستم از طرف دیگر به هزاران خانه وارد شده، و در آنجا به ترمینالهای هوشمند (که در واقع کامپیوترهای تخصصی هستند) متصل می‌شود.

تعداد زیادی شرکت سرویس دهنده نیز با فیبرهای نوری پُر ظرفیت به ستون فقرات شبکه وصل هستند، که برخی از آنها سرویسهای عمومی مانند «پول‌بده-تماشاکن» (pay-per-view) یا «پول‌بده-گوش‌کن» (pay-per-hear)، یا سرویسهای تخصصی مانند خرید از خانه (home shopping) ارائه می‌کنند. چنین شبکه‌ای می‌تواند سرویسهایی مانند اخبار، ورزش، فیلم و سریال، دسترسی وب، و هزاران امکان بالقوه دیگر در اختیار کاربران قرار دهد.

در این سیستم سرویس دهنده‌های کوچکتری در نزدیکی مشتریان تعبیه می‌شود (local spooling server)، که ویدئوهای درخواستی را ذخیره می‌کنند تا ترافیک شبکه در ساعات اوج مصرف کمتر شود. این کارها چگونه باید انجام شوند و چه کسی باید آنها را انجام دهد؟ هنوز بحث‌های سختی در جریان است. در اینجا ما فقط به قسمتهای اصلی سیستم می‌پردازیم: سرویس دهنده‌های ویدئو (video server) و شبکه توزیع (distribution network).

سرویس دهنده‌های ویدئو

برای ایجاد یک سیستم VOD (یا «تقریباً VOD») به سرویس دهنده‌های ویدئو، که بتوانند تعداد زیادی فیلم سینمایی را ذخیره و همزمان پخش کنند، نیاز داریم. تعداد کل فیلمهای سینمایی که تاکنون ساخته شده، چیزی در حدود ۶۵,۰۰۰ تخمین زده شده است (Minoli, 1995). یک فیلم معمولی با فرمت MPEG-2 چیزی در حدود 4 GB حجم خواهد داشت، بنابراین برای ذخیره کردن ۶۵,۰۰۰ فیلم به 260 TB (= ترابایت) نیاز داریم. اگر فیلمها و سریالهای قدیمی تلویزیونی، فیلمهای خبری و ورزشی، و کانالوهای ویدئویی را هم به آن اضافه کنیم، آن وقت متوجه می‌شوید که مشکل کجاست!

ارزانترین وسیله برای ذخیره کردن حجم زیادی از اطلاعات، نوار مغناطیسی است - و بنظر می‌رسد در آینده نزدیک وسیله ارزانتری به بازار نخواهد آمد. روی یک نوار مغناطیسی 200-GB می‌توان ۵۰ فیلم MPEG-2 (با هزینه ۱ تا ۲ دلار برای هر فیلم) ذخیره کرد. امروزه سرویس دهنده‌های بزرگ ویدئویی که می‌توانند هزاران نوار مغناطیسی را در خود نگه دارند، و برای جابجا کردن فیلمها به بازوهای روباتیک مجهز هستند، به بازار آمده‌اند. مشکل این سیستمها زمان دسترسی به فیلمها (به‌خصوص فیلم پنجاهم)، سرعت انتقال، و محدودیت تعداد دستگاههای پخش است (برای پخش همزمان n فیلم، به n دستگاه پخش نیاز داریم).

خوشبختانه، تجربه مغازه‌های کرایه ویدئو، کتابخانه‌های عمومی، و سازمانهای دیگر نشان می‌دهد که تمام آیتمها دارای محبوبیت یکسانی نیستند. طبق یک فرمول تجربی، اگر N فیلم داشته باشیم، احتمال درخواست برای k امین فیلم محبوب تقریباً C/k است - که در آن C بصورت زیر محاسبه می‌شود:

$$C = 1/(1 + 1/2 + 1/3 + 1/4 + 1/5 + \dots + 1/N)$$

طبق این فرمول (که به قانون زیف - Zipf law - معروف است) محبوبترین فیلم هفت برابر فیلم هفتم طرفدار دارد (Zipf, 1994). با استفاده از این واقعت می‌توان به یک مدل ذخیره‌سازی سلسله مراتبی دست یافت (شکل ۷-۷۹ را ببینید). در این مدل، بالا رفتن در هرم باعث افزایش کارایی (سرعت دسترسی، و سرعت انتقال) می‌شود. گزینه بعدی برای ذخیره‌سازی فیلمهای ویدئویی، دیسک نوری است. دیسکهای DVD در حال حاضر ظرفیتی معادل 4.7 GB دارند، که برای ذخیره کردن یک فیلم MPEG-2 کافیست، ولی نسل بعدی DVD برای ۲ فیلم جای کافی خواهد داشت. با آن که زمان جستجو در دیسکهای نوری در مقایسه با نوار مغناطیسی بیشتر است



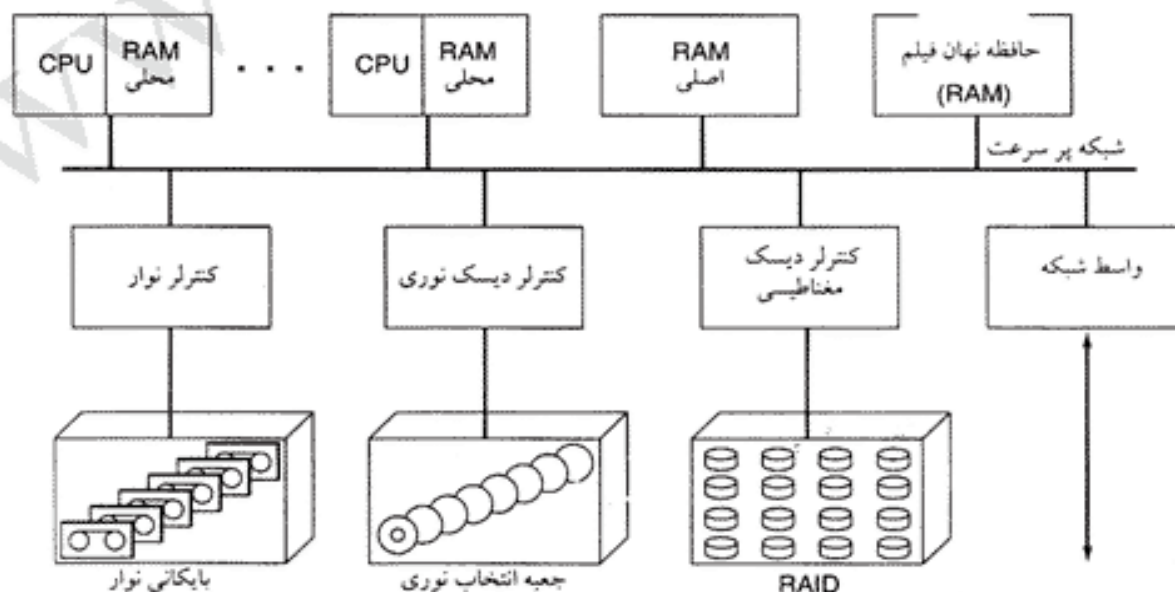
شکل ۷-۷۹. سلسله مراتب ذخیره سازی در سرویس دهنده های ویدئو.

(50 msec در مقابل 5 msec)، ولی آنها ارزانتر و مقاومتر هستند، و بزودی شاهد سرویس دهنده های ویدئو با ظرفیت هزاران DVD در بازار خواهیم بود.

در مرحله بعد دیسکهای مغناطیسی قرار دارند. این دیسکها با زمان دسترسی پائین (5 msec)، نرخ انتقال بالا (320 MB/sec در دیسکهای SCSI) و ظرفیت زیاد (> 100 MB) گزینه خوبی برای ذخیره کردن فیلمهای پربیننده هستند. عیب عمده این دیسکها قیمت بالای آنهاست.

در بالاترین نقطه هرم شکل ۷-۷۹ حافظه RAM قرار دارد. با اینکه قیمت RAM در سالهای اخیر بشدت کاهش پیدا کرده، ولی ذخیره کردن یک فیلم MPEG-2 به ۲۰۰ دلار حافظه RAM نیاز دارد - و برای ذخیره کردن فقط ۱۰۰ فیلم باید ۲۰,۰۰۰ دلار هزینه کنیم (که البته برای یک سرویس دهنده ویدئوی سطح بالا چندان زیاد و غیر عملی نیست).

از آنجائیکه یک سرویس دهنده ویدئو اساساً دستگاهیست برای I/O بی درنگ در حجمهای بالا، سخت افزار و نرم افزار آن بایستی تفاوت اساسی با کامپیوترهای ویندوز و یونیکس معمولی داشته باشد. در شکل ۷-۸۰ معماری سخت افزاری یک سرویس دهنده ویدئوی نوعی رامی بینید. قسمتهای مختلف این سرویس دهنده عبارتند از: یک یا چند CPU سریع (با مقداری RAM اختصاصی برای هر کدام)، یک حافظه اصلی مشترک، یک حافظه نهان بزرگ از نوع RAM برای فیلمهای پربیننده، ترکیبی از وسایل ذخیره سازی متنوع برای ذخیره کردن فیلمها، و سخت افزار شبکه (معمولاً ارتباط فیبر نوری به ستون فقرات ATM یا SONET با سرعت OC-12 یا بالاتر) - که این قسمتها با یک باس فوق سریع (حداقل 1 GB/sec) به یکدیگر متصل می شوند.



شکل ۷-۸۰. معماری سخت افزاری یک سرویس دهنده ویدئو.

نرم افزار سرویس دهنده ویدئو خود داستان دیگریست. در این سیستم، وظیفه CPU ها عبارتست از: گرفتن درخواست مشتریان، پیدا کردن فیلمها، منتقل کردن فیلم بین قسمتهای مختلف، نگهداری صورتحساب مشتریان، و مانند آن. زمان در برخی از این کارها نقش حیاتی دارد، بهمین دلیل باید از سیستم عامل بی درنگ استفاده کنیم (البته نه در همه CPU ها). در این سیستمها، هر وظیفه به بخشهای کوچکتر تقسیم می شود، و هر بخش باید در زمان معین به پایان برسد. برای زمانبندی این وظایف می توان از الگوریتمهایی مانند «نزدیکترین بن بست درنوبت بعد» (nearest deadline next) یا «نرخ یکنواخت» (rate monotonic) استفاده کرد (Liu and Layland, 1973).

این نرم افزار همچنین خصلت واسط سمت مشتری (سرویس دهنده های بینایی و گیرنده) را نیز تعیین می کند. دو نوع واسط کاربر بیشتر رواج دارند. اولی یک سیستم فایل معمولی است، که مشتری می تواند فایلهای مورد نظرش را باز کرده، بخواند، بنویسد، و سپس ببیند. چنین واسطی می تواند سیستم فایل شبیه یونیکس داشته باشد (البته با در نظر داشتن مشکلاتی که از ساختار سلسله مراتبی سیستم و بی درنگ بودن آن ناشی می شود).

واسط کاربر دوم به دستگاههای ضبط ویدئو شبیه است، با فرمانهایی مانند باز کردن فیلم، پخش، توقف، سریع به جلو، و سریع به عقب. تفاوت این واسط با قبلی (شبیه یونیکس) این است که به محض شروع پخش فیلم، سرویس دهنده (بدون نیاز به فرمان اضافی) داده ها را به سمت مشتری پمپ می کند.

قلب سرویس دهنده ویدئو نرم افزار «مدیریت دیسک» (disk management) است. این نرم افزار دو وظیفه اصلی دارد: انتقال فیلم از نوار مغناطیسی یا دیسک نوری به دیسک مغناطیسی، و انجام بموقع درخواست های خواندن دیسک. وظیفه اول (یعنی انتقال فیلم) نقش مهمی در افزایش کارایی سیستم دارد.

برای سازماندهی دیسکها دو روش وجود دارد: مزرعه دیسک (disk farm)، و آرایه دیسک (disk array). در روش اول، مزرعه دیسک، روی هر دیسک چند فیلم ذخیره می شود (که برای کارایی و اطمینان بیشتر می توان هر فیلم را روی چند دیسک - حداقل دو تا - ذخیره کرد). در روش دوم، آرایه دیسک یا RAID (آرایه افزونه از دیسکهای ارزان - Redundant Array of Inexpensive Disks)، هر فیلم روی چندین دیسک مختلف پخش می شود. به این روش - که در آن فیلم به n نوار باریک تقسیم شده، و این نوارها روی دیسکهای 0 تا $n-1$ ذخیره می شود - نوار کردن (striping) گفته می گویند.

یک آرایه دیسکهای نواری چندین مزیت نسبت به مزرعه دیسک دارد. اول، همه n دیسک را می توان بطور همزمان خواند یا نوشت، و این کارایی سیستم را n برابر می کند. دوم، می توان با قرار دادن یک دیسک اضافی در هر گروه n تایی از دیسکها، و نوشتن حاصل XOR تمام نوارهای اصلی روی این دیسک، آنها را در مقابل خرابی محافظت کرد. و بالاخره، متعادل کردن بار بطور خودکار انجام خواهد شد، و نیازی نیست فیلمهای پُریپینده را بصورت دستی بین دیسکهای مختلف پخش کنیم. از طرف دیگر، آرایه دیسک تکنیکی پیچیده است، و اگر چندین دیسک با هم خراب شوند، دسترسی به کل فیلمها غیرممکن خواهد شد. پیاده سازی ویژگیهای «سریع به جلو» یا «سریع به عقب» (که جزء الزامات واسط نوع دوم هستند) نیز در این سیستم بسیار دشوار است.

وظیفه دیگر نرم افزار مدیریت دیسک دادن سرویس بی درنگ به درخواستهای مشتریان است. تا چند سال پیش چنین وظیفه ای مستلزم طراحی الگوریتمهای پیچیده زمانبندی دیسک بود، ولی امروزه با کاهش شدید قیمت RAM روشهای بسیار ساده تری ممکن شده است. برای اینکه بتوان هر استریم را بصورت بی درنگ به مشتری فرستاد، برای هر فیلم بافری از ۱۰ ثانیه ویدئو (معادل 5 MB) در RAM نگه داشته می شود. این بافر توسط واسط دیسک پُر، و توسط واسط شبکه خالی می شود. با 500 MB RAM می توان ۱۰۰ استریم را به این شکل (مستقیماً از حافظه) سرویس داد. البته برای اینکه زیرسیستم دیسک بتواند این بافرها را بطور پیوسته تغذیه کند، باید سرعتی معادل 50 MB/sec داشته باشد، که این ویژگی با یکارگیری دیسکهای SCSI جدید بسادگی امکانپذیر است.

شبکه توزیع

شبکه توزیع (distribution network) عبارتست از مجموعه سونیچها و خطوط ارتباطی بین مبدأ و مقصد. همانطور که در شکل ۷-۷۸ دیدید، در این شبکه یک ستون فقرات شبکه وجود دارد، که به شبکه توزیع محلی متصل است. معمولاً سونیچینگ فقط در ستون فقرات وجود دارد، نه در شبکه توزیع محلی.

مهمترین چیزی که ستون فقرات باید داشته باشد، پهنای باند زیاد است. پائین بودن میزان لرزش (jitter - وقفه های کوچک و ناخوشایندی که در اثر ترافیک زیاد در پخش صدا و تصویر رخ می دهد) از دیگر الزامات ستون فقرات است، ولی از آنجائیکه ضعیفترین کامپیوترهای امروزی نیز می توانند حداقل ۱۰ ثانیه ویدئو با کیفیت MPEG-2 را بافر کنند، این ویژگی اهمیت سابق خود را از دست داده است.

بی نظمی عجیبی بر شبکه های محلی حکمفرماست، چون شرکتهای زیادی سعی می کنند تا خدمات متنوع خود را به مشتریان بفروشند. شرکتهای تلفن، تلویزیون کابلی (و اخیراً شرکتهای برق) متقاعد شده اند که برای برنده شدن در این میدان رقابت باید نفر اول باشند. در نتیجه، هر روز تکنولوژی جدیدی وارد بازار مصرف می شود. در ژاپن حتی شرکتهای فاضلاب وارد تجارت اینترنت شده اند، با این استدلال که گسترده ترین شبکه خطوط لوله متعلق به آنهاست، و می توانند فیبرهای نوری را به هر خانه ای بکشند (فقط باید دقت کنند کابلهای آنها از کجا سر در می آورند). چهار روش توزیع محلی VOD عبارتند از: ADSL، FTTC، FTTH، و HFC - اجازه دهید آنها را بررسی کنیم.

شرکتهای تلفن اولین بار با تکنولوژی ADSL وارد میدان رقابت شبکه های توزیع محلی شدند. در فصل ۲ درباره تکنولوژی ADSL صحبت کردیم، و نیازی به تکرار آن نیست. ایده اصلی ADSL استفاده از سیمهای مسی است که تقریباً به هر خانه ای (در اروپا، آمریکا و ژاپن) کشیده شده اند. اگر می شد از این سیمها برای پخش فیلم استفاده کرد، که نان شرکتهای تلفن توی روغن بود! ولی مشکل اینجاست که سیمهای مسی در فواصل ۱۰ کیلومتری حتی برای پخش MPEG-1 مناسب نیستند، چه رسد به MPEG-2. فیلمهای تمام رنگی با وضوح بالا به پهنای باند 4-8 Mbps (بسته به کیفیت مورد نظر) نیاز دارند، و ADSL (جز در مسافتهای بسیار کوتاه) نمی تواند چنین سرعتی ارائه کند.

دومین طرح شرکتهای تلفن FTTC (فیبر نوری تا کوچه - Fiber To The Curb) است. در FTTC، شرکت تلفن از ایستگاه پایانی (end office) یک رشته فیبر نوری به هر محله می کشد، و آنرا به دستگاهی موسوم به ONU (واحد شبکه نوری - Optical Network Unit) وصل می کند. به هر ONU می توان تا ۱۶ زوج سیم مسی تلفن وصل کرد. با این تمهید، طول کابلهای مسی آنقدر کوتاه می شود که دستیابی به T1 یا T2 دو - طرفه همزمان (که بترتیب برای MPEG-1 و MPEG-2 مناسبند) را ممکن می سازد. این سرویس (بعلمت مقارن بودن) برای ویدئو کنفرانس نیز کاملاً مناسب است.

سومین راه حل شرکتهای تلفن کشیدن فیبر نوری تا در خانه مشتریان است، که FTTH (فیبر نوری تا خانه) نام دارد. در این طرح، هر فرد می تواند یک کاربر OC-1، OC-3، یا حتی بیشتر خانه اش داشته باشد. FTTH بسیار گران است و بزودی عملی نخواهد شد، ولی می توان تصور کرد که امکانانی در اختیار مشتریان می گذارد. در شکل ۷-۶۳ دیدید که چگونه هر فردی می تواند یک ایستگاه رادیویی شخصی راه بیندازد؛ با داشتن FTTH راه اندازی ایستگاه تلویزیون شخصی چندان دور از ذهن نیست. هر سه طرح ADSL، FTTC و FTTH روشهایی نقطه-به-نقطه هستند، و این از شرکتهای تلفن (که به این نوع ارتباطات عادت دارند) چندان بعید نیست.

رهیافت HFC (آمیخته فیبر - کوکس - Hybrid Fiber Coax) که از سوی شرکتهای تلویزیون - رانه شده.

کلی با سه روش قبلی متفاوت است (به شکل ۲-۴۷ الف نگاه کنید). جریان از این قرار است: شرکت های تلویزیون کابلی در حال تعویض کابل های کواکس 300-450 MHz (با ظرفیت ۵۰ تا ۷۵ کانال 6-MHz) با کابل های کواکس 750-MHz (که ۱۲۵ کانال 6-MHz ظرفیت دارند) هستند، که فقط ۷۵ تا از این کانالها برای پخش تلویزیونی معمولی (آنالوگ) مورد استفاده قرار می گیرند.

اگر هر یک از ۵۰ کانال باقیمانده را با QAM-256 مدوله کنیم، پهنای باندی معادل 40 Mbps برای هر کانال (و در مجموع 2 Gbps) بدست می آوریم. با نزدیکتر کردن جعبه تقسیم سیستم به خانه های مشتریان، می توان به هر ۵۰۰ خانه یک کابل رساند. با یک حساب سرانگشتی می توان دید که به هر خانه پهنای باندی معادل 4 Mbps می رسد، که برای پخش فیلم های MPEG-2 کافیست.

هیجان انگیز است، نه؟ اما این کار مستلزم آن است که شرکت های تلویزیون کابلی تمام کابل های قدیمی را با کابل های 750 MHz جایگزین کنند، جعبه تقسیم های جدید نصب کنند، و تمام تقویت کننده های یکطرفه را هم جمع کنند - و این یعنی عوض کردن کل سیستم تلویزیون کابلی. هزینه این کار با ایجاد یک زیرساخت کامل FTTC قابل مقایسه است. در هر دو سیستم، شرکت های عمل کننده مجبورند به هر کوچه و خیابانی فیبر نوری بکشند، و در انتهای هر رشته فیبر یک مبدل نوری-الکتریکی نصب کنند. تنها فرق آنها در اینست که، در FTTC به هر خانه یک زوج سیم تابیده مستقل می رود، ولی در HFC یک کابل کواکس مشترک بین تمام خانه ها کشیده می شود. همانطور که می بینید، این دو سیستم آنقدرها که صاحبان آنها ادعا می کنند، با هم متفاوت نیستند.

با این حال یک تفاوت اساسی هست که باید به آن اشاره کرد: HFC از یک رسانه مشترک (بدون هیچگونه سوئیچینگ و مسپردگی) استفاده می کند. هر اطلاعاتی که روی این کابل فرستاده شود، بطور بالقوه بوسیله تمام مشترکان قابل دریافت است. اما FTTC سیستمی است مبتنی بر سوئیچینگ، و چنین چیزی در آن اتفاق نمی افتد. اگر متولیان HFC بخواهند کسی بدون پرداخت پول قادر به تماشای فیلم های پخش شده نباشد، باید از نوعی رمزنگاری استفاده کنند. در FTTC هیچ نیازی به رمزنگاری و اقدامات امنیتی اضافی نیست، و تنها حاصل آن افزایش پیچیدگی و افت کارایی سیستم خواهد بود. از دیدگاه شرکت های پخش رسانه، رمزنگاری ایده خوبیست یا خیر؟ اغلب شرکت های تلفن چنین کاری نمی کنند، با این توضیح که مایل به افت کیفیت نیستند (ولی در واقع برای ضرر زدن به رقبای HFC).

بعد از این سیستم های توزیع، نوبت به سرویس دهنده های ویدئوی محلی می رسد. آنها در واقع نسخه کوچکی از سرویس دهنده های ویدئو (که در بالا توضیح دادیم) هستند. نقش اصلی این سرویس دهنده ها کاستن از ترافیک ستون فقرات است.

از این سرویس دهنده های محلی می توان برای رزرو کردن فیلم ها استفاده کرد. اگر مشتریان از مدتی قبل تمایل خود را برای دیدن فیلمی به شرکت پخش رسانه اعلام کنند، این فیلم می تواند در ساعات خلوت شبکه در سرویس دهنده محلی بار شود. این روش می تواند به کاهش قیمت ها نیز منجر شود. برای مثال، اپراتور شبکه می تواند برای ساعات کم مصرف در تعرفه های خود تخفیف دهد.

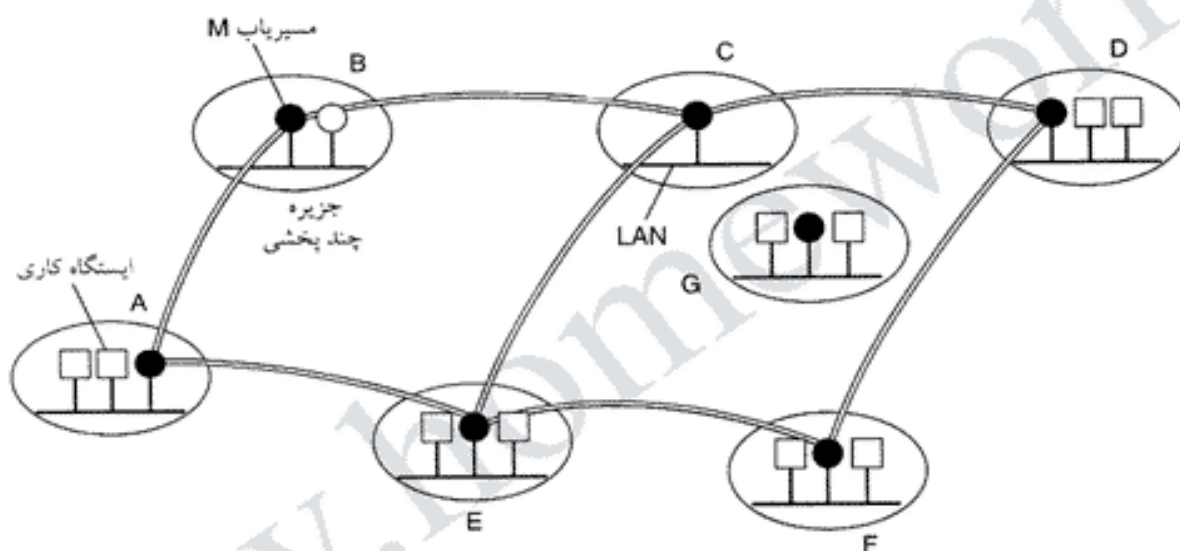
۹-۴-۷ ستون فقرات چندپخش - MBone

در همان حالیکه شرکت های تلفن و تلویزیون کابلی در حال طرح نقشه های بزرگ برای آینده «VOD دیجیتال» هستند، جامعه اینترنت به آرامی در حال پیاده سازی سیستم چندرسانه ای دیجیتال خاص خود است: MBone (ستون فقرات چندپخش - Multicast Backbone). در این قسمت خواهیم دید که MBone چیست، و چگونه کار می کند.

MBone را می توان تلویزیون اینترنتی دانست. برخلاف VOD (که تأکید روی تماس کاربر و پخش فیلم های از پیش فشرده شده از روی یک سرویس دهنده ویدئو است)، MBone بیشتر برای پخش ویدئوی دیجیتالی زنده

از سراسر دنیا و از طریق اینترنت مورد استفاده قرار می گیرد. این سیستم از سال ۱۹۹۲ عملیاتی شده، و کنفرانسهای علمی، از جمله گردهمایی های IETF، و رخدادهای مهم علمی (مانند پرتاب شاتل فضایی) از طریق آن پخش شده است. یکی از کنسرت های گروه رولینگ استونز، و بخشهایی از جشنواره فیلم کن نیز از طریق MBone پخش شده اند (البته در اینکه بتوان اینها را رخدادهای مهم علمی بشمار آورد، جای بحث است).

از نظر فنی، MBone شبکه ایست مجازی روی اینترنت، که تشکیل شده است از تعدادی جزیره چندپخش (multicast island) که بوسیله چند تونل (tunnel) به هم متصل شده اند (شکل ۷-۸۱ را ببینید). در این شکل، MBone شش جزیره دارد (A تا F)، که بوسیله هفت تونل به هم وصل شده اند. هر جزیره (که یک LAN، یا چند LAN متصل به هم است) از چندپخش سخت افزاری به کامپیوترهای میزبان خود پشتیبانی می کند. بسته های MBone از طریق تونل ها بین جزایر منتشر می شود. شاید روزی در آینده، که تمام مسیرهای اینترنت توانایی جابجایی ترافیک چندپخش را بدست آورند، دیگر نیازی به این ساختار نباشد، ولی فعلاً که کار ما را راه می اندازد.



شکل ۷-۸۱ MBone تشکیل شده است از جزایر چندپخش، که بوسیله تونل به هم متصل شده اند.

هر جزیره دارای یک یا چند مسیریاب ویژه موسوم به مسیریاب چندپخش (multicast router - mrouter) است. برخی از اینها مسیریابهای معمولی هستند، ولی برخی دیگر کامپیوترهای یونیکس هستند که نرم افزار خاصی را (در سطح root) اجرا می کنند. مسیریابهای چندپخش از نظر منطقی به تونل ها متصل می شوند. بسته های MBone در داخل بسته های IP پیچیده شده و بصورت بسته های تک پخش (unicast) معمولی به آدرس مسیریاب چندپخش مقصد ارسال می شوند.

تونل ها را باید بصورت دستی پیکربندی کرد. معمولاً، تونل روی یک مسیر فیزیکی اجرا می شود، ولی این الزامی نیست. مثلاً، اگر مسیر فیزیکی یک تونل از بد حادثه دچار مشکل شود، مسیریابهای چندپخش (که از این تونل استفاده می کنند) حتی از این موضوع مطلع نخواهند شد، چون اینترنت بطور خودکار ترافیک بسته های IP را به مسیرهای دیگر هدایت خواهد کرد.

وقتی یک جزیره جدید می خواهد به MBone وصل شود (مانند G در شکل ۷-۸۱)، سرپرست آن با ارسال پیام به لیست پستی MBone حضور خود را اعلام می کند. پس از آن سرپرستان سایتهای مجاور با وی تماس می گیرند، تا وی بتواند تونل های مورد نیاز را پیکربندی کند. حتی گاهی اوقات تونل های موجود برای بهینه کردن

توپولوژی شبکه و بهره‌گیری از جزیره جدید التأسيس، آرایش خود را عوض می‌کنند. چون آنها که واقعاً وجود خارجی ندارند، و چیزی نیستند جز چند جدول در حافظه مسیریابها (که آنها را هم براحتی می‌توان اضافه، حذف، و یا جابجا کرد). معمولاً هر کشور متصل به Mbone دارای یک ستون فقرات و تعدادی جزیره متصل به این ستون فقرات است. با عبور یک یا دو تونل از اقیانوس اطلس، Mbone به شبکه‌ای جهانی تبدیل شده است.

بنابراین، Mbone صرف‌نظر از تعداد آدرسهای چندپخشی (و تعداد کسانی که به آن گوش می‌کنند، یا آنرا تماشا می‌کنند) چیزی نیست جز چند جزیره و تونل، که کاملاً شبیه یک زیرشبکه معمولی (فیزیکی) است، بنابراین می‌توان از الگوریتمهای معمولی مسیریابی برای آن استفاده کرد. به همین دلیل، Mbone در شروع کار از یک الگوریتم مسیریابی بر اساس الگوریتم بردار فاصله بل من-فور، بنام DVMRP (پروتکل مسیریابی چندپخشی بردار فاصله - Distance Vector Multicast Routing Protocol)، استفاده کرد. برای مثال، در شکل ۷-۸۱ جزیره C برای ارسال بسته‌های خود به جزیره A می‌تواند از طریق B یا E (و یا حتی D) اقدام کند. این جزیره برای انتخاب مسیر مناسب، فاصله هر گره تا A را (از خود آن گره) گرفته و آنها را با هم جمع می‌کند. با این روش، هر جزیره می‌تواند بهترین مسیر به جزیره‌های دیگر را محاسبه کند. اما (همانطور که بزودی خواهید دید) مسیرها واقعاً به این شکل استفاده نمی‌شوند.

ابتدا اجازه دهید ببینیم اصلاً چندپخشی چگونه اتفاق می‌افتد. برای این که یک کامپیوتر بتواند صدا یا فیلم پخش کند، ابتدا باید یک آدرس چندپخشی کلاس D بگیرد. آدرسهای کلاس D مانند فرکانس ایستگاه رادیویی عمل می‌کنند، و همه آنها در یک پایگاه داده واحد نگهداری می‌شوند. یک کامپیوتر می‌تواند به هر آدرس چندپخشی که مایل است گوش کند، درست مثل تنظیم موج رادیو.

مسیریابهای چندپخشی بصورت متناوب بسته‌های پخشی IGMP در جزیره خود منتشر می‌کنند، تا ببینند چه کسی به کدام کانالها علاقه دارد. میزبانهایی که بخواهند آن کانال را دریافت کنند، در پاسخ یک بسته IGMP برمی‌گردانند. این پاسخها هم به تناوب فرستاده می‌شوند، تا شبکه محلی در این بسته‌ها غرق نشود. هر مسیریاب چندپخشی جدولی دارد که نشان می‌دهد چه کانالهایی را باید روی LAN خود پخش کند (پخش کانالهایی که هیچ بیننده‌ای ندارند، چیزی جز اتلاف پهنای باند نیست).

برنامه‌های چندپخشی بطریق ذیل روی Mbone منتشر می‌شوند. وقتی منبع صدا یا ویدئو بسته جدیدی تولید می‌کند، آن بسته را (به کمک سخت‌افزار چندپخشی) در جزیره خود پخش می‌کند. مسیریاب چندپخشی این بسته‌ها را گرفته، و روی تمام تونلهایی که به آنها وصل است، پخش می‌کند.

هر مسیریاب چندپخشی که بسته‌ای دریافت می‌کند، ابتدا چک می‌کند که آیا این بسته بهترین مسیر را طی می‌کند (هر مسیریاب بهترین مسیرها برای تمام گره‌ها را در جدولی نگه می‌دارد). اگر بسته در بهترین مسیر خود به این مسیریاب رسیده باشد، مسیریاب آنرا روی تمام تونلهای خروجی (تمام تونلهای منتهای تونلی که بسته از آن وارد شده) کپی می‌کند. اگر بسته در بهترین مسیر نباشد، مسیریاب آنرا دور می‌اندازد. برای مثال، اگر در شکل ۷-۸۱ جدول مسیریاب C بگوید که بهترین مسیر به A از گره B می‌گذرد، و یک بسته چندپخشی از A و از طریق B به C برسد، مسیریاب C این بسته را به تونلهای متصل به D و E کپی می‌کند. اما اگر یک بسته از E طریق A به C رسیده باشد (که طبق جدول، بهترین مسیر نیست)، مسیریاب C آنرا دور می‌اندازد. اگر بخاطر داشته باشید، این همان الگوریتم هدایت در مسیر معکوس است (که در فصل ۵ دیدید)، و با اینکه کاملترین روش نیست ولی الگوریتمی ساده و نسبتاً خوب محسوب می‌شود.

برای جلوگیری از ازدحام در اینترنت، علاوه بر استفاده از الگوریتم هدایت در مسیر معکوس، بسته‌های IP به TTL (زمان‌زنده ماندن - Time To Live) نیز مجهز می‌شوند تا برای همیشه در اینترنت سرگردان نشوند. این

مقدار توسط منبع ارسال بسته چندپخشی تعیین می شود. هر تونل Mbone دارای یک وزن خاص است، و اگر بسته ای بخواهد از یک تونل عبور کند، باید وزن کافی داشته باشد - در غیر اینصورت دور انداخته خواهد شد. برای مثال، اگر به تونل بین قاره ای (که از اقیانوس اطلس عبور می کند) وزن 128 بدهیم، و بخواهیم یک بسته چندپخشی را در همان قاره مبدأ محبوس کنیم و نگذاریم به قاره دیگر برود، کافیسست فیلد TTL آنرا به 127 (یا کمتر) ست کنیم. وقتی یک بسته از تونلی عبور می کند، باندازه وزن تونل از TTL آن کم خواهد شد.

با اینکه الگوریتم مسیریابی فوق بخوبی کار می کند، محققان زیادی برای بهبود آن تلاش می کنند. در یکی از این پیشنهادات از ایده مسیریابی بردار فاصله، ضمن تقسیم سلسله مراتبی سایتهای Mbone به مناطق مختلف، بهره گرفته شده است (Thyagarajan and Deering, 1995). پیشنهاد دیگر استفاده از مسیریابی حالت لینک، بجای مسیریابی بردار فاصله، است. بویژه، یکی از گروههای کاری IETF تغییراتی در پروتکل OSPF داده تا برای ایجاد یک سیستم چندپخشی خودمختار (Multicast AS) مناسب شود. این پروتکل OSPF چندپخشی (یا MOSPF) نام دارد (Moy, 1994). در MOSPF، مسیریاب علاوه بر اطلاعات معمولی مسیریابی، نقشه کامل جزیره ها و تونلهای چندپخشی را هم نگه می دارد. با استفاده از این اطلاعات، پیدا کردن بهترین مسیر از هر جزیره به جزیره دیگر کاری ساده و سر راست خواهد بود. الگوریتم دایکسترا (Dijkstra) یکی از الگوریتمهایی است که می توان از آن استفاده کرد.

زمینه دیگر تحقیقات مسیریابی بین-AS (بین سیستمهای خودمختار) است. در اینجا نیز یکی از گروههای کاری IETF الگوریتمی بنام PIM (چندپخشی مستقل از پروتکل - Protocol Independent Multicast) توسعه داده است. این پروتکل دو ویرایش دارد: یکی برای مناطق شلوغ (جزیره های پُر بیننده)، بنام PIM-DM؛ و دیگری برای مناطق خلوت (جزیره های کم بیننده)، بنام PIM-SM. هر دو ویرایش، بجای ایجاد لایه های توپولوژی خاص مانند کاری که DVMRP و MOSPF انجام می دهند، از جدولهای مسیریابی تک پخشی استاندارد استفاده می کنند. در PIM-DM مسیرهای زائد و بی مصرف حذف می شوند. روش حذف شاخه های اضافی چنین است. وقتی یک بسته چندپخشی از تونل «اشتباه» به یک گره می رسد، این گره از طریق همان تونل یک بسته حذف (purge packet) به فرستنده برمی گرداند، و از وی می خواهد که تا دیگر بسته ای در آن مسیر به وی ندهد. وقتی همین گره یک بسته «صحیح» دریافت می کند، آنرا به تمام تونلهایی که قبلاً خودشان را (برای آن مسیر) حذف نکرده اند، کپی می کند. اگر تمام تونلها خود را حذف کرده باشند، و در همان جزیره هم شنونده یا بیننده ای برای آن بسته وجود ندارد، مسیریاب چندپخشی یک بسته حذف روی تونل «صحیح» برمی گرداند. بدین ترتیب، چندپخشی همیشه بطور خودکار خود را با بهترین مسیرها منطبق می کند.

طرز کار PIM-SM، که در RFC 2362 تعریف شده، متفاوت است. در اینجا هدف آن است که برای یک کنفرانس سه نفره روی یک آدرس کلاس D در برکلی آمریکا، تمام مسیرهای اینترنت مشغول نشود. در این رهیافت از مفهومی بنام نقطه قرار (rendezvous point) استفاده می شود. هر فرستنده در یک گروه چندپخشی PIM-SM بسته های خود را به یک نقطه قرار می فرستد، و هر سایتی که به محتویات این فرستنده علاقمند باشد، یک تونل به آن نقطه قرار ایجاد می کند. با این روش، ترافیک PIM-SM از حالت چندپخشی خارج شده و بصورت تک پخشی درمی آید. محبوبیت PIM-SM روز به روز در حال افزایش است، و Mbone بتدریج به آن سمت حرکت می کند (و MOSPF کم کم کاربرد خود را از دست می دهد). البته خود Mbone هم بتدریج به نقطه اشباع و رکود نزدیک می شود، و بنظر می رسد هرگز نتواند به موفقیت خیره کننده ای دست یابد.

حتی اگر Mbone موفقیت آنچنانی بدست نیاورد، شبکه های چند رسانه ای همچنان یکی از فیلدهای مهیج و رشدیابنده اینترنت باقی خواهند ماند. هر روز تکنولوژی ها و برنامه های جدیدی وارد بازار می شوند، و چندپخشی

و کیفیت سرویس در حال نزدیک شدن به یکدیگر هستند (Striegel and Manimaran, 2002). چندپخشى بیسیم یکی دیگر از زمینه‌های پرتوجه و داغ است (Gossain et al., 2002). چندپخشى و تمام زمینه‌های مرتبط با آن احتمالاً تا سالهای آینده در مرکز توجه باقی خواهند ماند.

۵-۷ خلاصه

برای نامگذاری در اینترنت از یک سیستم سلسله‌مراتبی بنام سیستم نام ناحیه (DNS) استفاده می‌شود. در بالاترین نقطه این هرم ناحیه‌های شناخته شده، مانند edu، com و نزدیک به ۲۰۰ ناحیه کشوری، قرار دارد. DNS یک پایگاه داده توزیع شده است، که سرویس دهنده‌های آن در تمام دنیا پخش هستند. وظیفه DNS تبدیل نام ناحیه به آدرس IP است.

یکی از پُرطرفدارترین کاربردهای اینترنت ایمیل (email) است، و امروزه همه، از بچه‌های مدرسه‌ای گرفته تا پدربزرگها و مادربزرگها، از آن استفاده می‌کنند. اغلب سیستمهای ایمیل امروزی با استانداردهای تعریف شده در RFC 2821 و RFC 2822 کار می‌کنند. در این سیستمها مشخصات پیام به کمک سرآیندهای متنی (ASCII) تعیین می‌شود، و برای ارسال محتویات پیام می‌توان از انواع داده MIME استفاده کرد. ارسال پیام با برقراری یک اتصال TCP به پورت 25 کامپیوتر مقصد، به کمک پروتکلی بنام SMTP، انجام می‌شود.

شبکه تارنمای جهانی (وب) نیز تقریباً به همان اندازه ایمیل طرفدار دارد. وب سیستمی است از سندهای لینک شده به یکدیگر؛ این سندها به زبان HTML نوشته می‌شوند. در وب محتویات دینامیک نیز، در هر دو نوع سمت سرویس دهنده (PHP، JSP، و ASP) و سمت مشتری (JavaScript، و VBScript)، وجود دارند. برای دیدن صفحات وب از برنامه‌هایی موسوم به مرورگر، که کار آنها اتصال به سرویس دهنده، دریافت صفحه وب و نمایش آن است، استفاده می‌شود. برای افزایش کارایی وب تکنیکهای مختلفی، مانند حافظه نهان، تکثیر سرویس دهنده و شبکه‌های تحویل محتوا، توسعه داده شده است.

وب بیسیم تازه در آغاز راه است. اولین آنها عبارتند از WAP و I-Mode، که پهنای باند کمی دارند و صفحه نمایش آنها نیز کوچک است؛ ولی نسل بعدی قویتر خواهد بود.

چند رسانه‌ای (صدا و تصویر دیجیتال) یکی دیگر از ستاره‌های در حال طلوع اینترنت است. صدای دیجیتال (صدای جویباری، صدا روی IP، و رادیوی اینترنتی) بدلیل نیاز به پهنای باند کمتر مدتهاست وارد بازار شده، و همچنان در حال رشد است. پخش فیلم بر حسب تقاضا نیز طرفداران زیادی دارد، و در آینده از آن بیشتر خواهید شنید. یکی دیگر از بازیگران این صحنه MBone (سرویس تلویزیون دیجیتال اینترنتی) هنوز در مراحل تجربی بسر می‌برد.

مسائل

۱. اغلب کامپیوترهای تجاری سه شناسه جهانی و منحصر بفرد دارند. آنها چیستند؟
۲. با توجه به اطلاعات داده شده در شکل ۷-۳، کامپیوتر little-sister.cs.vu.nl جزء کلاس A است، یا B یا C؟
۳. در شکل ۷-۳ بعد از rowboat نقطه وجود ندارد. چرا؟
۴. حدس بزنید خندانک X- (که گاهی به شکل #-: هم نوشته می‌شود) چه معنایی دارد.
۵. DNS بجای TCP از UDP استفاده می‌کند، و اگر بسته‌ای گم شود، بازیابی آن بطور خودکار ممکن نیست. آیا این مشکل ساز نیست؟ اگر هست، راه حل آن چیست؟
۶. بسته‌های UDP، علاوه بر احتمال گم شدن، محدودیت طول نیز دارند (حداکثر ۵۷۶ بایت). اگر یک نام

- DNS بلندتر از این باشد، چه باید کرد؟ آیا می توان آنرا در دو بسته فرستاد؟
۷. آیا ماشینی با یک نام DNS می تواند چند آدرس IP داشته باشد؟ چگونه؟
۸. آیا یک کامپیوتر می تواند دو نام DNS (در دو ناحیه کاملاً جدا) داشته باشد؟ اگر جواب مثبت است، یک مثال بزنید. اگر نه، چرا؟
۹. در سالهای اخیر تعداد شرکتهایی که سایت وب دارند، بصورت انفجاری افزایش یافته، و اغلب این شرکتها هم سایت خود را در ناحیه com ثبت کرده اند، که باعث فشار بسیار زیاد به سرویس دهنده های سطح بالا در این ناحیه شده است. آیا برای رفع این مشکل (بدون تغییر دادن ساختار سیستم و معرفی ناحیه جدید) راهی می شناسید؟ اگر این راه حل به تغییر در سمت مشتری متکی باشد، اشکالی ندارد.
۱۰. در برخی از سیستمهای ایمیل فیلدی در سرآیند پیامها وجود دارد بنام: Content Return - این فیلد مشخص می کند که اگر گیرنده در مقصد شناسایی نشد، بدنه پیام برگشت داده شود یا خیر. این فیلد در کدام قسمت قرار می گیرد: پاکت نامه، یا سرآیند آن؟
۱۱. سیستمهای پست الکترونیک به نوعی دایرکتوری نیاز دارند، تا بتوانند اشخاص را به کمک آن پیدا کنند. برای ایجاد چنین دایرکتوری، مشخصات افراد باید به قسمتهای کوچکتر (نام، نام خانوادگی، و مانند آن) شکسته شود. درباره مشکلات تدوین استاندارد بین المللی چنین دایرکتوری بحث کنید.
۱۲. آدرس ایمیل افراد عبارتست از: نام ورود فرد به سیستم @ نام ناحیه DNS با یک رکورد MX. این نام ورود می تواند هر چیزی (نام کوچک، نام خانوادگی، و یا هر ترکیبی از آن با حروف و اعداد دیگر) باشد. در یک شرکت تعداد زیادی از ایمیل ها به هدر می رود، چون افراد نام ورود به سیستم گیرنده ها را نمی دانند. آیا راهی برای حل این مشکل بدون تغییر دادن DNS وجود دارد؟ اگر بله، چگونه؟ اگر خیر، چرا؟
۱۳. اندازه یک فایل باینری ۳۰۷۲ بایتی بعد از تبدیل به کد base64 (با یک جفت CR+LF بعد از هر ۸۰ بایت) چقدر خواهد شد؟
۱۴. روش نگذاری quoted-printable را در نظر بگیرید. یکی از مشکلات این روش را که در متن کتاب به آن اشاره نشده، نام برده و درباره راه حل آن بحث کنید.
۱۵. پنج نوع داده MIME را که در کتاب نیامده، نام ببرید. برای این کار می توانید در اینترنت جستجو کنید.
۱۶. فرض کنید می خواهید یک فایل MP3 را از طریق ایمیل به دوست خود بفرستید، ولی ISP دوست شما روی ایمیل های وارده محدودیت حجم 1 MB اعمال کرده، در حالیکه فایل شما 4 MB است. آیا راهی برای غلبه بر این مشکل (با توجه به RFC 822 و MIME) وجود دارد؟
۱۷. فرض کنید فردی یک دیمون تعطیلات برای خود ایجاد کرده، و درست قبل از خروج یک ایمیل به دوست خود می فرستد. متأسفانه این دوست هم برای یک هفته به مرخصی رفته و یک دیمون تعطیلات برای خود ست کرده است. چه اتفاقی می افتد؟ آیا این ایمیل ها تا برگشتن یکی از این دو نفر مدام بین دو سیستم پاس کاری خواهد شد؟
۱۸. در هر استاندارد، مانند RFC 822، قواعد گرامری (حتی ساده ترین آنها) باید به دقت تعریف شوند تا سیستمهای مختلف بتوانند با یکدیگر کار کنند. در سرآیندهای SMTP می توان از فاصله سفید (white space - هر یک از کاراکترهای فاصله، Enter یا Tab) بین توکن ها استفاده کرد. دو جایگزین ممکن دیگر برای فاصله بین توکن ها چیست؟
۱۹. دیمون تعطیلات بخشی از عامل کاربر است، یا عامل انتقال پیام؟ البته برای ست کردن دیمون تعطیلات از

- عامل کاربر استفاده می کنیم، ولی آیا پاسخها را همین عامل برمی گرداند؟ توضیح دهید.
۲۰. پروتکل POP3 اجازه می دهد تا کاربران ایمیل های خود را از یک صندوق پستی راه دور دریافت کنند. آیا این بدان معناست که فرمت داخلی صندوقهای POP3 باید استاندارد باشد، تا سیستمهای مختلف بتوانند با آن کار کنند؟ توضیح دهید.
۲۱. پروتکل های POP3 و IMAP از دیدگاه یک ISP تفاوت فاحشی دارند: کاربران POP3 معمولاً صندوق پستی خود را خالی می کنند، در حالیکه کاربران IMAP نامه های خود را برای مدتی نامحدود روی سرور می دهند. شما کدام روش را به یک ISP پیشنهاد می کنید؟ دلایل خود را توضیح دهید.
۲۲. پست وب (Webmail) از POP3 استفاده می کند، یا IMAP، یا هیچکدام؟ چرا؟ اگر پاسخ منفی است، روش آن به کدامیک نزدیکتر است؟
۲۳. هنگام ارسال صفحات وب هم از سرآیندهای MIME استفاده می شود. چرا؟
۲۴. چه زمانی از برنامه های کمکی برای دیدن محتویات صفحه وب استفاده می شود؟ مرورگر چگونه تشخیص می دهد از کدام برنامه باید استفاده کند؟
۲۵. آیا امکان دارد کلیک کردن روی یک لینک در مرورگرهای نت اسکپ و اینترنت اکسپلورر باعث اجرای برنامه های کمکی متفاوتی شود (در حالیکه در آن لینک یک نوع MIME مشخص شده است)؟ توضیح دهید.
۲۶. در شکل ۷-۲۱ یک سرور دهنده وب چندرسمانی را ملاحظه می کنید. در این سرور دهنده گرفتن درخواست کاربر و چک کردن حافظه نهان $500 \mu\text{sec}$ طول می کشد. در نیمی از مواقع فایل درخواستی در حافظه نهان پیدا شده، و بلافاصله برگردانده می شود. در نیمی دیگر، این مازول باید برای خواندن دیسک 9 msec دیگر صبر کند. (با فرض اینکه دیسک گلوگاه سیستم نباشد) برای استفاده کامل از ظرفیت CPU سرور دهنده چند مازول باید اجرا کند؟
۲۷. در URL های استاندارد http فرض بر این است که سرور دهنده به پورت 80 گوش می کند، ولی این بهیچوجه اجباری نیست. روشی طراحی کنید که URL بتواند به پورتهای غیراستاندارد هم دسترسی پیدا کند.
۲۸. URL ها می توانند علاوه بر نامهای DNS مستقیماً از آدرس IP هم استفاده کنند (مانند، `http://192.31.231.66/index.html`). مرورگر چگونه می تواند تشخیص دهد که بعد از `http://` یک نام DNS آمده یا آدرس IP؟
۲۹. فرض کنید فردی در دپارتمان کامپیوتر دانشگاه استغفورد برنامه ای نوشته و می خواهد آنرا از طریق FTP توزیع کند. محل این فایل در کامپیوتر سرور دهنده `ftp/pub/freebits/newprog.c` است - URL آن چه می تواند باشد؟
۳۰. در شکل ۷-۲۵، سایت `www.portal.com` تنظیمات کاربر را در یک کوکی نگه می دارد. مشکل این روش آن است که کوکی به 4 KB محدود است، و ممکنست برای نگهداری تنظیمات کاربر کافی نباشد. روشی طراحی کنید که این محدودیت را نداشته باشد.
۳۱. یک بانک تجاری «بانک تنبل ها» که می خواهد مشتریان تنبل خود را هم راضی نگه دارد، سیستمی طراحی کرده که بعد از وارد شدن مشتری به سیستم (با نام کاربر و کلمه رمز) یک شماره شناسایی بصورت کوکی روی کامپیوتر وی ذخیره می کند تا کاربر مجبور نباشد هر بار نام کاربر و کلمه رمز خود را وارد کند. چه نظری

- در باره این ایده دارید؟ آیا عملی است؟ آیا ایده خوبیست؟
۳۲. در شکل ۷-۲۶، پارامتر *ALT* برچسب `` ست شده است. تحت چه شرایطی مرورگر از این پارامتر استفاده می کند (و چگونه)؟
۳۳. چگونه می توان یک تصویر را در HTML قابل کلیک کرد؟ مثالی بزنید.
۳۴. با استفاده از برچسب `<a>` برای کلمه ACM لینکی به آدرس <http://www.acm.org> تعریف کنید.
۳۵. یک فرم سفارش خرید برای شرکت «اینترنت همبرگر» طراحی کنید، که مشتریان آن بتوانند از طریق اینترنت همبرگر خریداری کنند. در این فرم مشتری مشخصات خود (نام، آدرس، و شهر محل سکونت) و البته اندازه همبرگر (بزرگ یا بسیار بزرگ) و نوع پنیر آنرا وارد می کند. پول همبرگرها هنگام تحویل نقداً دریافت می شود، و امکان خرید با کارت اعتباری وجود ندارد.
۳۶. فرمی طراحی کنید که کاربر دو عدد را وارد کرده، و وقتی دکمه Submit را کلیک کرد، سرویس دهنده جمع آنها را برگرداند. برای نوشتن اسکریپت سمت سرویس دهنده از PHP استفاده کنید.
۳۷. برای هر یک از برنامه های زیر، مشخص کنید که آیا امکان استفاده از PHP یا جاوااسکریپت وجود دارد. و کدامیک بهتر است.
- (الف) نمایش یک تقویم برای هر یک از ماههای بعد از سپتامبر ۱۷۵۲.
- (ب) نمایش جدول پروازهای آمستردام به نیویورک.
- (ج) رسم نمودار چند جمله ای با استفاده از ضرایبی که کاربر وارد می کند.
۳۸. برنامه ای با جاوااسکریپت بنویسید، که یک عدد بزرگتر از ۲ را گرفته و مشخص کند این عدد اول است یا خیر. توجه کنید که جاوااسکریپت دارای ساختارهای `if` و `while` (شبه C و جاوا) است. عملگر باقیمانده در جاوااسکریپت `%` است. اگر به جذر x نیاز داشتید، می توانید از تابع `Math.sqrt(x)` استفاده کنید.
۳۹. صفحه HTML زیر را در نظر بگیرید:
- ```
<html> <body>
Click here for info
</body> </html>
```
- وقتی کاربر این لینک را کلیک می کند، یک اتصال TCP به سرویس دهنده برقرار شده و چند خط به آن فرستاده می شود. این خط ها را بنویسید.
۴۰. از سرآیند *If-Modified-Since* می توان برای تعیین اعتبار صفحه وب استفاده کرد. هر درخواست می تواند شامل مختلف (تصویر، صدا، ویدئو و البته HTML) باشد. کارایی این تکنیک را در مورد تصاویر JPEG و HTML مقایسه کنید.
۴۱. در روزهایی که مسابقات مهم ورزشی برگزار می شود، تعداد مراجعات به سایت رسمی آن رویداد افزایش چشمگیری پیدا می کند. آیا این اتفاق شبیه انتخابات فلوریدا در سال ۲۰۰۰ است؟ توضیح دهید.
۴۲. آیا یک ISP منفرد می تواند بعنوان CDN عمل کند؟ اگر بله، چگونه؟ اگر خیر، چرا؟
۴۳. در چه شرایطی استفاده از CDN ایده مناسبی نیست؟
۴۴. ترمینالهای بیسیم وب پهنای باند کمی دارند، و به همین دلیل کد کردن مناسب اهمیت بیشتری می یابد. روشی با کارایی مناسب برای انتقال متن انگلیسی روی لینکهای بیسیم به دستگاههای WAP طراحی کنید. فرض کنید دستگاه WAP چندین مگابایت ROM و CPU نسبتاً قوی دارد. راهنمایی: از ایده زبان ژاپنی، که در آن هر علامت معادل یک کلمه است، کمک بگیرید.

۴۵. یک دیسک فشرده صوتی ظرفیتی معادل 650 MB دارد. آیا در این دیسکها از فشرده سازی استفاده می شود؟ توضیح دهید.
۴۶. در شکل ۷-۵۷ (ج)، بدلیل استفاده از نمونه های ۴ بیتی برای نمایش ۹ سیگنال نویز کوانتیزه کردن رخ می دهد. اولین نمونه، در 0، دقیق است ولی چند نمونه بعدی دقیق نیستند. درصد خطا در نقاط 1/32، 2/32 و 3/32 چقدر است؟
۴۷. آیا از مدل روان شنوایی می توان برای کاستن از پهنای باند مورد نیاز تلفن اینترنتی استفاده کرد؟ اگر بله، تحت چه شرایطی؟ اگر خیر، چرا؟
۴۸. فاصله یک سرویس دهنده صدای جویباری با پخش کننده معادل 50 msec، و سرعت ارسال آن 1 Mbps است. اگر پخش کننده دارای بافری 1 MB باشد، درباره علامتهای حد-بالا و پائین چه می توان گفت؟
۴۹. مزیت الگوریتم یک درمیانی (شکل ۷-۶۰) آن است که در صورت گم شدن چند بسته خللی در پخش پیش نمی آید. اما کاربرد این الگوریتم در تلفن اینترنتی دارای یک عیب کوچک نیز هست. آن چیست؟
۵۰. آیا VOIP نیز مانند صدای جویباری با دیوار آتش (فایروال) مشکل دارد؟ توضیح دهید.
۵۱. نرخ انتقال تصویر غیر فشرده  $800 \times 600$  با رنگ 8 bit و با سرعت 40 frames/sec چقدر است؟
۵۲. آیا ۱ بیت خطا در یک فریم MPEG می تواند روی چند فریم تأثیر بگذارد؟ توضیح دهید.
۵۳. یک سرویس دهنده ویدئو با ۱۰۰,۰۰۰ مشتری را در نظر بگیرید، که هر مشتری در ماه دو فیلم تماشا می کند. نیمی از این فیلمها باید در ساعت ۸ بعد از ظهر پخش شود. این سرویس دهنده در هر لحظه (در همان ساعت) چند فیلم باید پخش کند؟ اگر هر فیلم به 4 Mbps پهنای باند نیاز داشته باشد، این سرویس دهنده به چند خط OC-12 نیاز دارد؟
۵۴. فرض کنید قانون زیف در مورد یک سرویس دهنده ویدئو با ۱۰,۰۰۰ فیلم مصداق دارد. اگر این سرویس دهنده ۱۰۰۰ تا از فیلمهای پُریننده تر را روی دیسک و ۹۰۰۰ فیلم دیگر را روی CD نگه دارد، چند درصد از درخواستها به دیسک مراجعه می کند؟ (عبارت آنرا بدست آورید.) برای ارزیابی عددی این فرمول، یک برنامه کوچک بنویسید.
۵۵. برخی از متقلبان اینترنتی اقدام به ثبت نامهای اینترنتی نزدیک به نامهای معروف می کنند (مانند [www.microsoft.com](http://www.microsoft.com) که فقط یک حرف آن با [www.microsoft.com](http://www.microsoft.com) جابجا شده است). حداقل پنج تا از این نامهای تقلبی را فهرست کنید.
۵۶. بسیاری از افراد نامهای DNS بصورت [www.word.com](http://www.word.com) که در آن word یکی از کلمات رایج است، را ثبت کرده اند. رای هر یک از دسته های زیر پنج سایت (بهمراه مختصری از مشخصات آنها) فهرست کنید: حیوانات، غذاها، لوازم خانگی، و اندامهای بدن.
۵۷. چرا علامت دلخواه اموجی  $12 \times 12$  طراحی کنید. برای مثال، سعی کنید کلمات پسر، دختر، معلم، و سیاستمدار را نمایش دهید.
۵۸. یک سرویس دهنده POP3 بنویسید که فرمانهای زیر را قبول کند: `RETR`، `LIST`، `PASS`، `USER`، `QUIT` و `DELE`.
۵۹. سرویس دهنده شکل ۶-۶ را با استفاده از فرمان `HTTP 1.1 GET` به یک سرویس دهنده واقعی وب تبدیل کنید. این سرویس دهنده همچنین باید پیامهای `Host` را قبول کند. این سرویس دهنده باید یک حافظه نهان داشته باشد، و فایلهایی را که اخیراً بازدید شده اند در این حافظه نهان نگه دارد.

# امنیت شبکه



در چند دهه ابتدایی پیدایش، از شبکه های کامپیوتری بیشتر توسط پژوهشگران دانشگاه و برای ارسال نامه های الکترونیکی و یا توسط کارمندان شرکتها برای به اشتراک گذاری چاپگر، استفاده می شد. در چنین شرایطی، امنیت شبکه از اهمیت چندانی برخوردار نبود. اما اکنون که میلیون ها تن از شهروندان عادی از شبکه ها برای انجام عملیات بانکی، معاملات یا پر کردن اظهارنامه های مالیاتی خود استفاده می کنند، امنیت شبکه به عنوان یک مسئله بالقوه و عمده پدیدار شده است. در این فصل از چندین زاویه به مطالعه امنیت شبکه خواهیم پرداخت، اشکالات و موانع امنیت را متذکر شده و چندین الگوریتم و پروتکل را که شبکه ها را امن تر و قابل اطمینان می کنند، تشریح خواهیم نمود.

«امنیت شبکه» در برگیرنده عناوین و موارد بسیار گسترده است و با مشکلات و معضلات متعددی سروکار دارد. در یک عبارت ساده می توان امنیت را «اطمینان از عدم دسترسی افراد فضول و جلوگیری از دستکاری در پیام های محرمانه دیگران» تعبیر کرد. همچنین می توان امنیت را در ارتباط با افرادی تعبیر کرد که تلاش می کنند به سرویس های راه دور در شبکه دسترسی پیدا کنند در حالی که مجوز استفاده از آنها را ندارند؛ یا به روش هایی اطلاق می شود که بتوان صحت پیام هایی که مثلاً از اداره اخذ مالیات (IRS) می رسد و اعلام می کند: «حداکثر تا جمعه مبلغ اعلام شده را واریز کنید» را تأیید کرد و تشخیص داد که این پیام واقعاً از اداره مالیات آمده نه از مافیا! همچنین می توان امنیت را در خصوص پیشگیری از دخل و تصرف و یا پاسخ جعلی به پیام های قانونی دیگران و مقابله با افرادی که پس از ارسال پیام سعی در انکار آنها می کنند، تعبیر کرد.

منشاء اغلب مشکلاتی که به صورت عمده برای امنیت شبکه ها به وجود می آید افرادی هستند که سعی در کسب درآمد نامشروع، جلب توجه یا آزاررسانی به دیگران دارند. در شکل ۸-۱ فهرستی از افراد که بطور عام مرتکب جرم های امنیتی می شوند و انگیزه های آنان درج شده است. با بررسی این جدول روشن خواهد شد که تضمین امنیت شبکه، مقوله ای فراتر از رفع اشکالات برنامه نویسی است. در این خصوص باید تمهیداتی برای پیشگیری از حمله دشمنانی اندیشیده شود که غالباً افرادی باهوش و کوشا هستند و گاهی اوقات سازمان یافته و با برنامه ریزی قبلی اقدام به حمله می کنند. البته همیشه عملیات مخرب بر علیه شبکه توسط یک گروه خاص و خطرناک انجام نمی شود؛ بررسی پرونده های اداره پلیس نشان می دهد که بسیاری از حملات بر علیه شبکه توسط عوامل خارجی و به واسطه نفوذ از طریق خط تلفن نبوده است بلکه توسط عوامل مغرض داخلی انجام گرفته است. بنابراین طراحی سیستم های امنیتی باید با در نظر داشتن این حقیقت انجام شود.

تهدیدکنندگان امنیت	هدف
دانشجو	تفریح کردن از طریق تجسس در نامه های دیگران
کراکر (Cracker)	به منظور آزمایش سیستم امنیت متعلق به شخص (یا گروه) خاص یا سرقت اطلاعات
نماینده فروش	برای اطلاع از طرحهای استراتژیک حریف در خصوص بازار خرید و فروش
کارمند اخراجی	برای انتقام گیری
حسابداران	برای اختلاس پول از یک شرکت
دلال سهام	برای انکار وعده هایی که به یک مشتری داده شده است (از طریق email)
کلاه بردار	برای سرقت شماره های کارتهای اعتباری جهت خرید
جاسوس	برای اطلاع از اسرار نظامی یا صنعتی دشمن
تروریستها	برای سرقت اسرار جنگهای میکروبی

شکل ۸-۱. فهرست برخی از افراد که منجر به مشکلات امنیتی می شوند و انگیزه های آنها.

مشکلات امنیت شبکه بطور کلی به چهار رده نزدیک و مرتبط به هم تقسیم بندی می شوند: (۱) سری مانند اطلاعات<sup>۱</sup>، (۲) احراز هویت کاربران<sup>۲</sup>، (۳) غیرقابل انکار بودن پیامها<sup>۳</sup>، (۴) نظارت بر صحت اطلاعات<sup>۴</sup>.

سری مانند اطلاعات که گاه «محرمانه نگاهداری اطلاعات» (Confidentiality) نیز نامیده می شود، متضمن انجام عملیاتی است که اطلاعات را از دسترس کاربران غیرمجاز و بیگانه دور نگاه می دارد. این همان مفهومی است که در ذهن مردم عادی در خصوص امنیت شبکه تداعی می شود. «احراز هویت» عبارتست از تایید هویت طرف مقابل ارتباط قبل از آنکه اطلاعات حساس در اختیار او قرار بگیرد یا در معاملات تجاری شرکت داده شود.

مقوله «غیرقابل انکار بودن پیامها» (Nonrepudiation) با امضاهای دیجیتالی سر و کار دارد و به اطلاعات و مستندات، هویت حقوقی اعطاء می کند: وقتی مشتری شما که قبلاً به صورت الکترونیک یک میلیون عدد از یک کالای کوچک به قیمت ۸۹ سنت سفارش داده و بعداً ادعا می کند قیمت کالا ۶۹ سنت بوده، چگونه می توان خلاف ادعای او را ثابت کرد؟ شاید حتی او ادعا کند هرگز چنین سفارش خریدی نداده است.

نهایتاً چگونه می توان مطمئن شد که پیامی که شما دریافت کرده اید، دقیقاً همان پیامی است که در اصل فرستاده شده و یک دشمن بدخواه در حین انتقال پیام آن را دستکاری و تحریف نکرده است.

تمام موارد فوق الذکر (سری مانند اطلاعات، احراز هویت، غیرقابل انکار بودن پیامها و نظارت بر صحت اطلاعات) در سیستمهای سنتی و معمولی پیرامونمان نیز وجود دارد، البته با تفاوتهای قابل توجه؛ عملیات محرمانه نگاهداشتن و نظارت بر صحت اطلاعات با اتکاء به پست سفارشی و لاک و مهر کردن مستندات انجام می شود. همچنین عموم افراد اغلب قادرند تفاوت بین اصل یک سند و تصویر آن سند را تشخیص بدهند. به عنوان یک آزمایش تصویر (کپی شده) یک چک معتبر بانکی را تهیه کرده و سعی در نقد کردن اصل چک در روز دوشنبه بنمائید. حال، تلاش کنید تصویر کپی شده چک را در روز سه شنبه نقد کنید تا تفاوت بین رفتاری که با شما می شود را عیناً مشاهده نمائید!!! در بانکداری الکترونیک، اصل و کپی چکهای الکترونیکی تفاوتی با هم نداشته و غیرقابل تشخیص است. لذا چگونگی برخورد بانک با اینگونه چکها جای تأمل دارد.

در دنیای پیرامون ما، افراد از طریق تشخیص چهره یک فرد، صدا یا دستخط، او را احراز هویت می‌کنند. در عملیات اداری، تأیید هویت اشخاص از طریق امضای آنها در برگه‌های حقوقی یا مهرهای برجسته و نظائر آن انجام می‌شود. هرگونه تلاش برای جعل یا دستکاری در اسناد، با مقایسه دستخط یا امضاء، قابل کشف است. این گزینه‌ها در دنیای الکترونیک قابل اعمال نیستند و بدیهی است که به راهکارهای دیگری نیاز است. قبل از آن که به ماهیت تک‌تک راه‌حلهای تضمین امنیت اطلاعات بپردازیم، بررسی تمهیدات و راهکارهای امنیتی که در هر یک از لایه‌های پشته پروتکلی شبکه (Protocol Stack) ممکن و قابل اعمال است، خالی از لطف نخواهد بود.

رعایت نکات و تمهیدات امنیتی فقط در یک نقطه خاص متمرکز نیست. در هر لایه از معماری شبکه، باید نکات و موارد امنیتی مدنظر قرار گرفته و بدقت رعایت شود: در لایه فیزیکی (Physical Layer) برای جلوگیری از ایجاد انشعاب در سیم و پیشگیری از استراق سمع سیگنال (Wire tapping)، می‌توان سیمها را در درون یک لوله محافظ جاسازی و لوله را با یک گاز تحت فشار پر کرد. در این حالت هرگونه تلاش در نقب زدن به این لوله و سوراخ کردن آن منجر به تخلیه گاز، کاهش فشار لوله و به صدا در آمدن زنگهای هشدار خواهد شد. در برخی از سیستمهای نظامی از این روش استفاده شده است.

در لایه پیوند داده‌ها (Data Link)، بسته‌های ارسالی بر روی خطوط نقطه به نقطه (Point To Point) قبل از خروج از ماشین مبدا، رمزنگاری شده و به محض ورود به ماشین مقصد رمزگشایی می‌شود. تمام جزئیات کار در سطح لایه پیوند داده‌ها پیاده‌سازی و انجام می‌شود و لایه‌های فوقانی به آنچه که در این لایه اتفاق می‌افتد بی‌توجه هستند و بهیچوجه درگیر جزئیات آن نخواهند شد. این راه‌حل در مواقعی که بسته‌های حامل داده مجبور به گذر از چند مسیر یاب باشند کارآیی خود را از دست خواهد داد زیرا اطلاعات در بدو ورود به هر مسیر یاب رمزگشایی شده و بمحض خروج از آن مجدداً رمزنگاری می‌شوند فلذا این خطر وجود دارد که در یکی از مسیر یابهای بین راه به اطلاعات حمله شود. از طرف دیگر با این روش نمی‌توان از نشستها (Sessions) حفاظت و مراقبت کرد (به عنوان مثال مراقبت از یک نشست Online برای خرید از طریق کارت اعتباری ممکن نیست) و در هر یک از مسیر یابهای واقع بر روی مسیر احتمال دستکاری یا استراق سمع اطلاعات وجود خواهد داشت. علیرغم این کمبودها، روش «رمزنگاری لینک» (Link Encryption) را می‌توان به سادگی به هر شبکه افزود و بیشتر مواقع نیز سودمند است.

در لایه شبکه (Network Layer) می‌توان برای نظارت مؤثر بر ورود و خروج بسته‌های مجاز و تشخیص بسته‌های غیرمجاز (حامل داده‌های مخرب)، «دیوار آتش» (Firewall) نصب کرد. در ضمن در این لایه می‌توان از پروتکل IPsec (IP Security) استفاده کرد.

در لایه انتقال (Transport Layer) کل اتصال (Connection) بین مبدا و مقصد، می‌تواند رمزنگاری شود. به عبارت بهتر تمام داده‌های در حال تبادل، در پروسه مبدا رمز شده و در پروسه مقصد از رمز خارج خواهد شد؛ (به این روش امنیت انتها به انتها یا End-to-End گفته می‌شود). برای تضمین حداکثر امنیت، اعمال «امنیت انتها به انتها» الزامی است.

در آخر، مواردی نظیر احراز هویت کاربران و غیرقابل انکار بودن محتوای پیامها (Nonrepudiation) فقط در لایه کاربرد قابل اعمال و پیاده‌سازی است.

از آنجایی که تضمین امنیت اطلاعات دقیقاً در یک لایه خاص قرار نمی‌گیرد لذا نمی‌شد مفاد آنرا در ادامه هیچیک از فصول این کتاب گنجانده؛ بهمین دلیل یک فصل اختصاصی و مجزا به آن اختصاص داده‌ایم. این فصل طولانی، فنی و بنیادی است و همچنین ممکن است در نگاه اول اندکی گسیخته و نامربوط به نظر

برسد. بررسی مستندات به خوبی نشان می دهد که بسیاری از شکستها در حریم امنیتی شبکه هایی مثل بانکها، بیشتر ناشی از عواملی نظیر بی لیاقتی و سهل انگاری کارمندان، تمهیدات و روالهای امنیتی ناکافی، تقلب و کلاهبرداریهای داخلی بوده است تا آنکه از یک نقشه جنایتکارانه دقیق، ایجاد انشعاب در خطوط تلفن، سرقت و رمزگشایی اطلاعات منشاء گرفته باشد. اگر شخصی که یک کارت عابربانک (ماشین خودپرداز ATM) را در خیابان پیدا می کند براحتمی بتواند به یکی از شعب مربوطه مراجعه نماید و با اعلام آنکه شماره PIN (شماره رمز شخصی) را فراموش کرده در همان شعبه شماره جدیدی دریافت کند (تحت عنوان ارتباط خوب و دوستانه با مشتریان!) آنگاه استفاده از الگوهای رمزنگاری و مکانیزمهای امنیتی، دیگر هیچ خاصیت و کاربردی در حفظ امنیت اطلاعات نخواهند داشت. کتاب Ross Anderson در این خصوص حقیقتاً هوشیارکننده و هشداردهنده است؛ مستندات کتاب مذکور نشان می دهد که در صدها نمونه از شکستهای امنیتی و نقض حریم شبکه ها در صنایع و سازمانهای متعدد، تقریباً تمام آنها (در یک عبارت مودبانه) ناشی از سهل انگاری در عملکرد و بی دقتی در رعایت نکات امنیتی بوده است. (Anderson, 2001) علیرغم این مسئله، خوشبین هستیم که با گسترش روزافزون تجارت الکترونیکی، موسسات و شرکتها در روالهای عملیاتی خود تجدیدنظر کرده، شکافهای امنیتی سیستمهای خود را برطرف نموده و جنبه های فنی «امنیت» را سرلوحه کار خود قرار بدهند.

به غیر از امنیت در سطح لایه فیزیکی (که به صورت مکانیکی و از طریق لوله های محتوی گاز تحت فشار انجام می شود) در بقیه لایه ها تقریباً تمام مکانیزمهای امنیتی مبتنی بر اصول رمزنگاری است. به همین دلیل مطالعات خود در خصوص امنیت را با تشریح مبسوط مبانی رمزنگاری آغاز خواهیم کرد. در بخش ۱.۸ نگاهی بر اصول اولیه آن خواهیم داشت. در بخش ۲.۸ تا بخش ۵.۸ به برخی از الگوریتمهای اساسی و ساختمان داده های مورد استفاده در رمزنگاری خواهیم پرداخت. سپس به صورت مبسوط تشریح خواهیم کرد که چگونه این مفاهیم برای تأمین امنیت در شبکه های کامپیوتری بکار گرفته می شوند. نهایتاً فصل را با خلاصه ای از نظریات در خصوص «تکنولوژی و جامعه» (Technology & Society) جمع بندی خواهیم کرد.

قبل از شروع، به مواردی که در این فصل بدانها نخواهیم پرداخت اشاره می کنیم. سعی کرده ایم در این فصل به مواردی در خصوص امنیت بپردازیم که مستقیماً در ارتباط با معماری شبکه است فلذا مواردی را که به سیستم عامل و برنامه های کاربردی مربوط می شود، بررسی نکرده ایم. به عنوان مثال در این فصل مطلبی درخصوص احراز هویت کاربران به روش بیومتریک، استراتژیهای امنیت کلمه عبور، «حمله از نوع سرریز کردن بافر» و «اسبهای تروا» (Trojan Horses)، «تقلب در ورود به سیستم» (Login Spoofing)، بمبهای منطقی، ویروسها، کرمها و نظایر آن وجود ندارد. تمام این موارد به تفصیل در فصل نهم از کتاب «سیستم عامل مدرن» (تانیام، ۲۰۰۱) بررسی شده اند. علاقمندان می توانند برای مطالعه بر روی این جوانب از امنیت، به کتاب فوق مراجعه نمایند. حال بیایید خط سیر خود را آغاز کنیم.

## ۱.۸ رمزنگاری

کلمه Cryptography (رمزنگاری) برگرفته از لغات یونانی به معنای «محر» - نوشتن متون است. رمزنگاری پیشینه ای طولانی و درخشان دارد که به هزاران سال قبل بر می گردد. در این بخش برخی نکات برجسته از تاریخ رمزنگاری را بعنوان اطلاعات عمومی (که دانستن آنها برای ادامه بخشهای بعد مفید خواهد بود)، بررسی خواهیم کرد. برای آشنایی با پیشینه دقیق و کامل رمزنگاری، کتاب Kahn (۱۹۹۵) جهت مطالعه توصیه می شود؛ برای تحلیل جامع مفاهیم مدرن و پیشرفته در امنیت و آشنایی با الگوریتمهای رمزنگاری، پروتکلها و برنامه های کاربردی به کتاب Kaufman (۲۰۰۲) مراجعه کنید. برای آشنایی با جزئیات ریاضی این روشها کتاب Stinson

(۲۰۰۲) را ببینید. برای بررسی این روشها، بدون جزئیات ریاضی، به کتاب Burnett & Paine (۲۰۰۱) مراجعه کنید.

متخصصین رمزنگاری بین «رمز» (Cipher) و «کُد» (Code) تمایز قائل می‌شوند. «رمز» عبارتست از تبدیل کاراکتر به کاراکتر یا بیت به بیت بدون آن که به محتویات زبان شناختی (ادبیات) آن پیام توجه شود. در طرف مقابل، «کُد» تبدیلی است که کلمه‌ای را با یک کلمه یا علامت (سمبول) دیگر جایگزین می‌کند. امروزه از کدها استفاده چندانی نمی‌شود اگرچه استفاده از آن پیشینه طولانی و پر سابقه‌ای دارد. موفقترین «کُد»هایی که تاکنون ابداع شده‌اند توسط ارتش ایالات متحده و در خلال جنگ جهانی دوم در اقیانوس آرام بکار گرفته شد. آنها از لهجه محلی Navajo در میان سرخپوستان الهام گرفته و برای عبارات و کلمات نظامی به سادگی از لغات خاص این زبان محلی استفاده کردند؛ به عنوان مثال عبارت chay-dagahi-nail-tsaide (در زبان محلی سرخپوستان به معنای «کُشنده لاک‌پشت») رمزی برای سلاح ضد تانک بود. زبان Navajo لهجه‌ای بسیار آهنگین و بشدت پیچیده است و هیچ ادبیات نوشتاری و الفبای خطی ندارد و یک فرد ژاپنی (آن هم در جنگ جهانی دوم) هیچ چیزی در مورد آن نمی‌دانست.

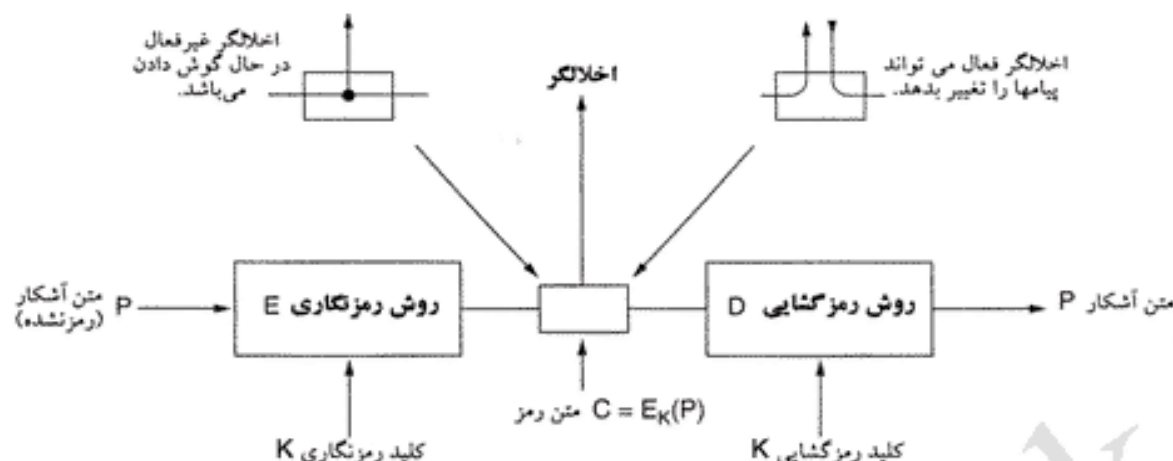
در سپتامبر ۱۹۴۵ در مجمع متفقین در سن دیه‌گو این «کُد» با بیان گزارش ذیل توصیف شد: «برای سه سال متوالی، هر گاه ناوگان دریایی در خشکی پهلوی می‌گرفت، آنچه که جاسوسان ژاپنی (از بی‌سیم) استراق سمع می‌کردند یک صدای نامفهوم و شلوغ بود که با دیگر اصوات درهم آمیخته و در نتیجه صدایی شبیه به لحن راهبان تبت یا صدای یک بطری آبجوش که آب آن در حال خالی شدن باشد، می‌شنیدند!!» ژاپنی‌ها هرگز نتوانستند این کد را بشکنند؛ پس از جنگ بسیاری از افرادی که مبادله پیامهای سری جنگ را به زبان رمزی Navajo بر عهده داشتند بخاطر خدمات شایان و شجاعتشان در طول جنگ، مفتخر به دریافت نشانهای عالی نظامی شدند. ارتش ایالات متحده توانست کدهای رمز ژاپنی‌ها را بشکند در حالی که ژاپنی‌ها نتوانستند کد Navajo را بشکنند و این حقیقت، نقش بسیار مهمی در پیروزیهای آمریکاییان در جنگ اقیانوس آرام (با نیروی دریایی ژاپن) ایفاء کرد.

### ۱-۱-۸ مقدمه‌ای بر رمزنگاری

از دیدگاه تاریخ، چهار گروه از مردم در شکل‌گیری هنر رمزنگاری دخیل بوده‌اند: «نظامیان»، «هیئت‌های سیاسی»، «خاطره‌نویسان/واقعه‌نگاران» و «عشاق!». از بین اینها نظامیان نقش بسیار مهمتری دارند و در طول قرن‌ها به تکوین این شاخه از علم پرداخته‌اند. سابقاً در مؤسسات نظامی، پیامهایی که باید رمزنگاری می‌شدند به یک کارمند (منشی) دون پایه و حقوق‌بگیر تحویل می‌شد تا آنها را رمز و ارسال کند. حجم عظیم پیامهایی که در طی یک روز باید رمز و ارسال می‌شد مانع از آن بود که بتوان این کار خطیر را بر عهده معدود متخصصین خبره حاضر در یک مؤسسه گذاشت.

تا زمان ابداع کامپیوترها، در عرصه یک جنگ واقعی و با تجهیزات اندک، بزرگترین نقطه ضعف استراتژی رمزنگاری آن بود که همه چیز به توانائی و سرعت عمل کارمند رمزنگار پیام، وابسته و منوط می‌شد. محدودیت دیگر آن بود که نمی‌شد براحتی و سریع یک روش رمزنگاری را به روشی دیگر تغییر داد زیرا این کار مستلزم بازآموزی جمع کثیری از منشیان و کارمندان رمزنگار بود. از طرفی این خطر نیز وجود داشت که یکی از منشیان رمزنگار، دستگیر شده و روش رمزنگاری فاش گردد لذا باید این امکان مهیا می‌شد که به محض احساس لزوم، روش رمزنگاری تغییر کند. این مشکلات متناقض، منجر به پیدایش مدل شکل ۸-۲ شد.

پیامی که باید رمزنگاری شود، «متن آشکار» (Plaintext) نامیده می‌شود و توسط یک تابع خاص با پارامتری بنام «کلید» (Key) به متن رمز، تبدیل می‌گردد. نتیجه فرآیند رمزنگاری که «متن رمز» (Ciphertext) نامیده می‌شود بر روی کانال منتقل خواهد شد. فرض کنیم که دشمن یا اخلاک‌گر (Intruder) متن رمز شده را به صورت کامل



شکل ۸-۲. مدل رمزنگاری (برای روشهای رمزنگاری با کلید متقارن).

می شوند و آن را در اختیار می گیرد. به هر حال او برخلاف گیرنده اصلی، سراسیمه قادر به رمزگشایی پیام و بهره برداری از آن نخواهد بود زیرا کلید رمز را نمی داند. برخی اوقات یک اخلالگر غیر فعال (Passive Intruder) نه تنها قادر است به جریان اطلاعات بر روی کانال مخابراتی گوش بدهد بلکه می تواند آنها را در جایی ثبت کرده و بعداً آن را بارها به جریان بیندازد؛ در مقابل یک اخلالگر فعال (Active Intruder) می تواند پیام مورد نظر خود را در داخل یک پیام مجاز و معتبر جاسازی کند یا در آن دستکاری نماید. هنر شکستن رمز بدون در اختیار داشتن کلید آن، «علم تحلیل رمز» (Cryptanalysis) نام دارد؛ به هنر ابداع روشهای رمزنگاری جدید «علم رمزنگاری» (Cryptology) اطلاق می شود.

در اختیار داشتن یک نماد و فرمول ریاضی که ارتباط بین متن آشکار، متن رمز شده و کلید رمز را مشخص کند بسیار مفید خواهد بود. ما از نماد  $C = E_K(P)$  استفاده خواهیم کرد، بدین معنا که عملیات رمزنگاری بر روی متن آشکار  $P$  توسط کلید رمز  $K$  انجام شده و متن رمز شده  $C$  بدست آمده است. به روش مشابه، فرمول  $P = D_K(C)$  عمل رمزگشایی متن رمز شده توسط کلید  $K$  را (به منظور استخراج اصل پیام) توصیف می کند. بنابراین داریم:

$$D_K(E_K(P)) = P$$

این نماد بیانگر آن است که  $E$  و  $D$  توابع ریاضی و معکوس یکدیگر هستند. تنها نکته قابل اشاره آنست که این توابع دارای دو پارامتر هستند، اگرچه کلید رمز  $K$  را که در حقیقت یکی از پارامترهای این توابع است به صورت پانویس برای  $E$  یا  $D$  نشان داده ایم تا تمایز آن از پیام مشخص باشد.

یکی از قواعد اساسی در علم رمزنگاری آن است که شخص باید فرض را بر آن بگذارد که دیگران [از جمله تحلیلگران رمز و رمزشکنها] الگوریتم بکار رفته در عملیات رمزنگاری را می دانند. به عبارت دیگر شخص رمزشکن، روش رمزنگاری یعنی تابع  $E$  و روش رمزگشایی یعنی تابع  $D$  در شکل ۸-۲ را می داند و آنچه از او پنهان نگاه داشته می شود فقط کلید رمز ( $K$ ) است. میزان نیرو و تلاشی که باید برای ابداع، آزمایش و نصب یک الگوریتم رمزنگاری جدید (در صورت فاش شدن روش قبلی) انجام بگیرد بقدری زیاد است که محرمانه نگهداشتن روش رمزنگاری عملاً ممکن نیست. تصور آنکه الگوریتم رمزنگاری می تواند سری و مخفی بماند (در حالی که ممکن نیست) خطرات و زیانهای بیشتر از منافع آن دارد.

اینجاست که «کلید رمز» وارد قضیه می شود. «کلید رمز» یک رشته کاراکتری نسبتاً کوتاه است که پیام براساس آن رمز می شود. برخلاف آن که روش رمزنگاری ممکن است هر چند سال یکبار تغییر کند، کلید رمز می تواند بر

طبق نیاز و به دفعات عوض شود. بنابراین مدل پایه سیستمهای رمزنگاری، مدلی است پایدار (ثابت) که همه از عملکرد و الگوریتم آن مطلعند و فقط با یک کلید محرمانه و قابل تغییر کار می‌کند. این نظریه که «تحلیل‌گر رمز» (رمزشکن / Cryptanalyst) از الگوریتم رمزنگاری آگاه است و سری ماندن یک پیام صرفاً به مخفی ماندن کلید رمز وابسته است «اصل کِرکُهف» نامیده می‌شود که توسط یکی از رمزنگاران ارتش فلاندرز به نام Auguste Kerckhoff در سال ۱۸۸۳ بیان شده است. بنابراین داریم:

**اصل کِرکُهف:** تمام الگوریتمهای رمزنگاری باید آشکار و عمومی باشند و تنها کلیدهای رمز، مخفی و محرمانه هستند.

شاید نتوان حق این مطلب را که «تکیه بر مخفی ماندن الگوریتم رمزنگاری اشتباه است»، بدرستی ادا کرد. تلاش برای سری نگه داشتن الگوریتم رمزنگاری که در عُرف عامه به اصطلاح «امنیت در سایه گمنامی و ابهام» (Security by Obscurity) مشهور است، هرگز محقق نخواهد شد. با عمومی سازی یک الگوریتم، طراح یک الگوریتم رمزنگاری می‌تواند از نیروی عظیم متخصصین رمزنگاری که مشتاق به شکستن یک سیستم هستند، مشورت بگیرد و آنها را به مبارزه بطلبد؛ آنها نیز می‌توانند در خصوص تلاشهایی که برای درهم شکستن یک سیستم رمزنگاری مصروف کرده‌اند مقاله بنویسند و خبرگی و هوش خود را به رخ بکشند! هرگاه متخصصین کثیری سعی در شکستن یک الگوریتم کردند و با گذشت پنج سال پس از انتشار عمومی آن هیچ گزارشی مبنی بر موفقیت آنان مشاهده نشد، آن الگوریتم احتمالاً بقدر کافی سخت و محکم بوده است!

از آنجایی که سری ماندن پیامها وابسته به کلید است لذا طول کلید یکی از نکات بسیار مهم در طراحی الگوریتمهای رمزنگاری است. به عنوان مثالی ساده، یک قفل رمزدار ترکیبی [مثل قفل بعضی از کیفهای شخصی] را در نظر بگیرید. قاعده کلی برای باز شدن این قفل آن است که چند رقم را به ترتیب وارد کنید. همه این موضوع [الگوریتم] را می‌دانند و لیکن کلید رمز محرمانه است. کلید رمز دو رقمی تنها صد حالت مختلف دارد. کلید سه رقمی معادل با هزار و کلید شش رقمی معادل یک میلیون حالت مختلف است. هر چه طول یک کلید بزرگتر باشد، حجم عملیات سعی و خطائی که رمزشکن برای دسترسی به کلید رمز باید انجام بدهد زیادتر خواهد بود.<sup>۱</sup> حجم عملیات (Work Factor) برای شکستن یک سیستم رمز از طریق آزمون تمام فضای حالات کلید، برحسب طول کلید به صورت نمائی رشد خواهد کرد. سری ماندن و امنیت پیامها با داشتن یک الگوریتم بسیار قدرتمند (ولی آشکار و همگانی) به همراه یک کلید طولانی تضمین می‌شود. برای آن که نگذارید برادر کوچک شما نامه‌های الکترونیکی شما را بخواند یک کلید ۶۴ بیتی (هشت کاراکتری) کفایت می‌کند. برای انجام عملیات معمول اقتصادی باید از کلیدی با حداقل ۱۲۸ بیت استفاده شود. برای حراست از پیامهای سری دولتی به کلیدهایی با طول حداقل ۲۵۶ بیت یا حتی بیشتر نیاز است.

از دیدگاه یک تحلیل‌گر رمز (رمزشکن)، مسئله کشف رمز سه شیق اساسی را در بر می‌گیرد: (۱) هرگاه او فقط توده‌ای از متن رمز شده (بدون هیچ متن آشکار و بدون کلید) در اختیار داشته باشد با مسئله‌ای به نام «صرفاً متن رمز شده»<sup>۲</sup> مواجه است. (۲) وقتی رمزشکن بخشی از متن آشکار را به همراه معادل رمز شده آن، در اختیار دارد، اصطلاحاً با مسئله «متن آشکار و شناخته شده»<sup>۳</sup> مواجه است. (۳) نهایتاً هرگاه رمزشکن قادر باشد هر قسمت دلخواه از یک متن آشکار را رمز کند اصطلاحاً با مسئله «متن آشکار و انتخابی»<sup>۴</sup> مواجه است. هرگاه به یک رمزشکن اجازه داده شود تا پیرسد مثلاً حاصل رمزنگاری رشته ABCDEFGHIJKL چیست به سادگی قادر

۱. به حجم عملیات لازم برای کشف کلید رمز به روش سعی و خطا، اصطلاحاً Work Factor گفته می‌شود. -م.

۲. Chosen Plaintext

۳. Known Plaintext

۴. Ciphertext Only

خواهد بود تا رمزهای معمولی را بشکنند و کلید رمز را بدست بیاورد.

نواموزان حرفه رمزنگاری به اشتباه می اندیشند که هرگاه یک متن رمز شده بتواند در مقابل حمله نوع «صرفاً متن رمز شده» استقامت کند و فاش نشود، آن سیستم رمز مطمئن است.<sup>۱</sup> این فرض کاملاً خام و ناشیانه است زیرا در بسیاری از حالات، رمزشکن قادر است حدسهای درست و موثقی را در مورد برخی از قسمتهای یک متن رمز شده آزمایش کند.<sup>۲</sup> به عنوان مثال اولین جمله ای که در حین برقراری ارتباط شخص از راه دور [مثلاً با یک سرویس دهنده TelNet]، ارسال می شود معمولاً کلمه login: (به صورت رمز شده) است. حال رمزشکن پاره ای متن رمز شده به همراه معادل رمز نشده آن را در اختیار دارد؛ کار او نسبتاً ساده و سراسر است. برای رسیدن به امنیت کامل، طراح الگوریتم رمزنگاری باید محافظه کار بوده و مطمئن شود که سیستم رمز او بگونه ای است که حتی در صورتی که حریف رمزشکن، معادل رمز شده یک متن آشکار را در اختیار داشته باشد باز هم قادر به شکستن رمز و بدست آوردن کلید رمز نیست.<sup>۳</sup>

روشهای رمزنگاری بطور کلی به دو رده تقسیم می شوند: (۱) رمزهای جانشینی (Substitution) (۲) رمزهای جایگشتی (Transposition). در اینجا به عنوان مقدمه ای بر روشهای مدرن رمزنگاری، مختصراً به این دو روش خواهیم پرداخت.

## ۲-۱-۸ رمزهای جانشینی (Substitution Cipher)

در رمزنگاری جانشینی هر حرف یا گروهی از حروف با یک حرف یا گروهی دیگر از حروف جایجا می شوند تا شکل پیام بهم بریزد. یکی از قدیمی ترین رمزهای شناخته شده روش رمزنگاری سزار است که ابداع آن به ژولیوس سزار نسبت داده می شود. در این روش حرف a به D تبدیل می شود، b به E، c به F و به همین ترتیب تا z که با C جایگزین می گردد. به عنوان مثال در این روش، کلمه attack به DWWDFN تبدیل می شود. در مثالها متن پیام متشکل از حروف کوچک فرض شده و متن رمز شده صرفاً شامل حروف بزرگ انگلیسی است.

یک حالت عمومی و ساده از رمزنگاری سزار آن است هر حرف الفباء از متن اصلی با حرفی که در جدول الفباء k حرف بعدتر قرار گرفته جایجا شود. (روش Shift by k) در این روش کلید رمز عدد k خواهد بود و براساس آن حروف یک متن به صورت چرخشی (Circular) با حرف kام بعد از خودش جایگزین می شود. روش رمزنگاری سزار شاید در آن زمانها کسی را فریفته باشد ولی امروزه نمی تواند هیچکس را گول بزند. بهبود بعدی این روش آن است که هر حرف در متن اصلی با یک حرف دلخواه جانشین شود، یعنی ۲۶ حرف جدول الفباء به حروف دیگری در همان جدول نگاشته شود. به عنوان مثال از نگاشت زیر می توان برای رمزنگاری جانشینی استفاده کرد:

متن آشکار    abcdefghijklmnopqrstuvwxyz  
متن رمز شده    QWERTYUIOPASDFGHJKLZXCVBNM

هر سیستم رمزنگاری که در آن یک سمبول با سمبول دیگر جایگزین می شود اصطلاحاً «سیستم جانشینی

۱. بدین خیال که چون رمزشکن فقط متن رمز شده را در اختیار دارد و چیزی از متن اصلی نمی داند بنابراین قادر به آزمون تمام حالات مختلف کلید رمز نیست و بالطبع کلید فاش نخواهد شد. - م

۲. به عبارت دیگر اگرچه به زعم دیگران او از متن رمز شده چیزی نمی داند ولی رمزشکن قادر است بخشهای کوچکی از یک متن را حدس بزند. م

۳. بعبارت روشتر سیستم رمز باید بنحوی طراحی شود که حتی اگر یک متن رمز شده و معادل رمز نشده آنرا به رمزشکن بدهید باز هم نتواند کلید رمز شما را پیدا کند. - م

تک حرفی» (Monoalphabetic Substitution) گفته می‌شود که در آن کلید رمز یک رشته ۲۶ کاراکتری است و نگاشت جدول الفباء را مشخص می‌نماید. بر اساس کلید رمز مثال بالا، کلمه attack به QZZQEA تبدیل خواهد شد.

در نگاه اول این سیستم رمزنگار مطمئن به نظر می‌رسد زیرا اگرچه رمز شکن روش عمومی جانشینی حروف را می‌داند ولی نمی‌داند از بین 26! حالت مختلف (معادل با  $4 \times 10^{26}$  حالت) کدامیک کلید رمز است. برخلاف رمز سزار، آزمایش تمام حالات مختلف کلید غیرممکن است زیرا اگر هر یک از حالات کلمه رمز در یک نانو ثانیه آزمایش شود، بررسی تمام حالات کلید توسط چنین کامپیوتری  $10^{10}$  سال طول خواهد کشید.

در روش فوق علیرغم آنکه آزمایش تمام حالات یک کلید ممکن نیست ولی حتی برای یک قطعه متن رمز شده کوچک، رمز متن به سادگی شکسته خواهد شد. در حمله اصولی به این سیستم رمز از ویژگیهای آماری زبانهای طبیعی بهره گرفته شده است. به عنوان مثال در زبان انگلیسی حرف *e* بیشترین تکرار را در متون معمولی دارد؛ به دنبال آن حرف *t*، سپس *a*، *o*، *n* و *i* در رتبه‌های بعدی قرار می‌گیرند. ترکیبات دو حرفی که اصطلاحاً digram نامیده می‌شوند به ترتیب بیشترین تکرار عبارتند از: (۱) *th* (۲) *in* (۳) *er* (۴) *re* (۵) *an* و بهمین ترتیب. ترکیبات سه حرفی حروف انگلیسی (Trigram) به ترتیب بیشترین تکرار عبارتند از: (۱) *the* (۲) *ing* (۳) *and* و (۴) *ion*

تحلیل‌گر رمز (رمز شکن) برای شکستن سیستم رمزنگاری «جانشینی تک حرفی» (Monoalphabetic) با شمارش حروف متن رمز شده و محاسبه تکرار نسبی هر حرف شروع می‌کند؛ سپس حرفی را که دارای بیشترین تکرار است با *e* و حرف پر تکرار بعدی را با *t* جایگزین می‌کند. حال او می‌تواند با در نظر داشتن سه حرفی *the* به دنبال سه حرفی‌های *tXe* در متن رمز شده بگردد؛ به احتمال قوی *X* معادل با *h* است. سپس به روش مشابه به سراغ چهار حرفی‌های *thYt* می‌گردد. به احتمال زیاد *Y* معادل با *a* است. با اطلاعاتی که بدست آمده است او به دنبال سه حرفیهای مکرر با الگوی *aZW* می‌گردد که احتمالاً معادل با *and* خواهد بود. با حدس زدن بقیه یک حرفیها، دو حرفیها و سه حرفیهای تکراری و با شناخت از حروف صدا دار و بی صدا و چگونگی ترکیب آنها در کلمه، رمز شکن می‌تواند متن اصلی را به روش سعی و خطا (حرف به حرف) بدست آورد. یک روش دیگر برای شکستن رمز متون آن است که یک کلمه یا یک عبارت کامل حدس زده شود. به عنوان مثال به متن رمز شده زیر که متعلق به یک شرکت حسابرسی است (و به صورت دسته‌های پنج کاراکتری نشان داده شده است) دقت کنید:

CTBMN BYCTC BTJDS QXBNS GSTJC BTSWX CTQTZ CQVUJ  
QJSGS TJQZZ MNQJS VLNSX VSZJU JDSTS JQUUS JUBXJ  
DSKSU JSNTK BGAQJ ZBGYQ TLCTZ BNYBN QJSW

احتمال وجود کلمه «financial» در پیامهای یک شرکت حسابرسی زیاد است. با استفاده از این ویژگی که کلمه financial دارای دو حرف «i» با فاصله چهار حرف از یکدیگر است، در متن رمز شده به جستجوی حروف مشابه با فاصله چهار حرف از یکدیگر می‌پردازیم. با جستجو در متن فوق به ۱۲ مورد تطابق در موقعیتهای: ۶، ۱۵، ۲۷، ۳۱، ۴۲، ۴۸، ۵۶، ۶۶، ۷۰، ۷۱، ۷۶ و ۸۲ بر می‌خوریم. ولیکن از بین این دوازده مورد فقط موارد ۳۱ و ۴۲ هستند که در آنها حرف بعدی (متناظر با *n* از کلمه financial) با فاصله یک حرف تکرار شده و مطابقت دارد. از بین این دو مورد نیز فقط مورد ۳۱ است که با تکرار حرف *a* با فاصله سه حرف مطابقت دارد؛ لذا می‌توان متوجه شد که در این متن کلمه financial در موقعیت ۳۰ از متن رمز شده قرار گرفته است. پیدا شدن این کلمه، نقطه شروع خوبی برای پیدا کردن کلید رمز با کمک ویژگیهای آماری حروف در زبان انگلیسی خواهد بود.

## ۳-۱-۸ رمزنگاری جایگشتی (Transposition)

رمزنگاری جانشینی ترتیب سمبولهای یک متن را حفظ می کند ولی (صرفاً) شکل سمبولها را تغییر می دهد. برعکس، «رمزنگاری جایگشتی» ترتیب حروف متن را بهم می ریزد ولیکن شکل آنها را تغییر نخواهد داد. در شکل ۳-۸ یک روش عمومی از رمزنگاری جایگشتی که در آن ترتیب سمبولها بصورت ستونی بهم ریخته می شود نشان داده شده است. کلید رمز یک کلمه یا عبارت [انگلیسی] است که هیچ حرف تکراری ندارد. در این مثال کلید رمز کلمه MEGABUCK انتخاب شده است. کاربرد کلید رمز آنست که ستونها شماره گذاری شود. شماره هر ستون براساس ترتیب الفبایی هر حرف کلید نسبت به جدول الفبای انگلیسی تعیین می شود. به عنوان مثال ستون چهارم شماره ۱ است (حرف A)، ستون پنجم شماره ۲ (حرف B)، ستون هفتم شماره ۳ (حرف C) و ستون دوم شماره ۴ (حرف E) و به همین ترتیب. متن اصلی به صورت افقی (سطری) در ذیل کلید رمز نوشته می شود. سپس در صورت لزوم تعدادی حرف به آخرین سطر اضافه می شود تا ماتریس مربوطه پر شود. متن رمز شده بر اساس شماره ستونها به صورت عمودی خوانده شده و به هم متصل می شود. ترتیب خواندن ستونها، از ستون با کمترین شماره به بزرگترین شماره است.

```

M E G A B U C K
7 4 5 1 2 8 3 6
p l e a s e t r
a n s f e r o n
e m i l l i o n
d o l l a r s t
o m y s w i s s
b a n k a c c o
u n t s i x t w
o t w o a b c d

```

متن آشکار

```

pleasetransferonemilliondollarsto
myswissbankaccountsixtwo

```

متن رمز شده

```

AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUEIRICXB

```

شکل ۳-۸. رمزنگاری جایگشتی.

برای شکستن رمز فوق، تحلیلگر رمز ابتدا باید مطمئن شود که آیا واقعاً با یک متن رمز شده به روش جایگشت روبرو است یا خیر؟ با بررسی تکرار حروف N، I، O، A، T، E و نظائر آن بسادگی می توان مطمئن شد که متن الگویی شبیه به یک متن معمولی رمز نشده دارد. در این صورت روشن است که متن رمز شده از نوع جایگشتی است زیرا در این روش شکل هر حرف تغییر نمی کند بلکه فقط جای آن در متن عوض می شود و این کار در آمار حروف و میزان تکرار آنها تأثیری ندارد.

گام بعدی آن است که تعداد ستونها حدس زده شود. در بیشتر مواقع می توان برخی از کلمات یا عبارات یک متن را حدس زد. به عنوان مثال فرض کنید رمز شکن ما به وجود عبارت milliondollars در جایی از متن اصلی مشکوک است. دقت کنید که دو حرفی های MO، IL، LL، LA، IR و OS در متن رمز، دیده می شود که ناشی از شکسته و چیده شدن عبارت فوق به صورت ستونی است. حرف O در متن رمز شده پشت M ظاهر شده است. (یعنی در اثر چیده شدن عبارت فوق در سطرها، این دو حرف در ستون چهارم کنار هم قرار گرفته اند.) «حرفی که در متن رمز شده کنار هم قرار می گیرند در متن اصلی به اندازه «طول کلید» از هم فاصله دارند»، اگر طول کلید به جای ۸، هفت در نظر گرفته می شد، پس از رمز شدن عبارت milliondollars، دو حرفی های MD، IO، LL، LL، OR، IA و NS بوجود می آمدند. در حقیقت بسته به طول کلید مجموعه متفاوتی از دو حرفی ها در متن رمز شده ظاهر می شود. تحلیلگر رمز می تواند با آزمایش طولهای مختلف کلید، به سادگی طول کلید را بدست بیاورد.

گام آخر، بدست آوردن ترتیب ستونها است. وقتی تعداد ستونها  $(k)$  کم باشد به سادگی می توان تمام  $k \times (k-1)$  حالت مختلف زوج ستونها را آزمود تا ببینیم آیا میزان تکرار دو حرفها و سه حرفهای این دو ستون با شرایط آماری یک متن معمولی مطابقت دارد یا خیر؟ حالتی از ترکیب ستونها که براساس آن ترکیب، نتیجه رمزگشایی بیشترین تطبیق را با متون معمولی داشته باشد به عنوان زوج ستون متوالی در نظر گرفته می شود. حال یکایک ستونها به عنوان ستونهای بعدی این زوج رمزگشایی شده، آزمایش و تحلیلهای آماری بر روی آنها انجام می شود و بهترین ستون به عنوان ستون بعدی آن در نظر گرفته می شود و این کار ادامه می یابد. ستون قبلی یک ستون نیز به همین روش (تحلیل دو حرفی و سه حرفی) امکان پذیر است. این فرآیند آنقدر تکرار می شود تا ترکیب درست و مورد تایید به دست بیاید. عمل رمزشکنی در نقطه ای با اقبال و موفقیت رو برو خواهد شد که یک کلمه یا یک عبارت ظاهر گردد. (به عنوان مثال هر گاه در آزمون یک ترکیب خاص از ستونها، کلمه million ظاهر شود می توان ترتیب واقعی ستونها و اشتباهاتی که قبلاً در حدسها وجود داشته را روشن کرد).

برخی از سیستمهای رمزنگاری جایگشتی، یک بلوک از کاراکترها با طول ثابت را از ورودی دریافت کرده و یک بلوک رمز شده (با طول ثابت) در خروجی تولید می کنند. در این گونه از روشها فهرست کامل جایگشتهای ورودی (که متن رمز شده خروجی را تولید می کند) مشخص است. به عنوان مثال سیستم رمز شکل ۸-۳ را می توان در قالب یک رمزنگار با بلوکهای ورودی ۶۴ بیتی تصور کرد که فهرست جایگشتها عبارتست از ۴، ۱۲، ۲۰، ۲۸، ۳۶، ۴۴، ۵۲، ۶۰، ۵، ۱۳، ۲۱، ... و ۶۲. به عبارت دیگر چهارمین کاراکتر ورودی (یعنی a) اولین کاراکتر خروجی این رمزنگار خواهد بود.

### ۴-۱-۸ رمز One-Time Pads (بیم ریزی محتوی پیام)

ایجاد یک سیستم رمزنگاری که غیرقابل شکستن باشد حقیقتاً کار ساده ایست و روش آن نیز از دهها سال قبل شناخته شده است: ابتدا یک رشته بیت تصادفی را به عنوان کلید انتخاب کنید. سپس متن اصلی (رمز نشده) را به یک رشته بیت متوالی تبدیل نمایند؛ (مثلاً با الحاق بیتهای کد آسکی هر کاراکتر) نهایتاً این دو رشته را بیت به بیت با یکدیگر XOR (Exclusive OR) کنید. رشته بیت حاصل، متن رمز شده شماست که هرگز قابل شکستن نیست زیرا در صورتی که متن رمز شده به قدر کافی بزرگ باشد هر حرف در این متن به یک نسبت تکرار خواهد شد، همچنین میزان تکرار تمام دو حرفها و سه حرفها (Digram/Trigram) در متن رمز، مشابه خواهد بود.<sup>۱</sup> این روش که به نام One-Time Pad مشهور است در برابر تمام حملات فعلی یا حملات احتمالی آینده مصون خواهد ماند و میزان توان محاسباتی و هوش رمزشکن هیچ تأثیری در شکستن آن نخواهد داشت. دلیل منطقی شکست ناپذیری این روش رمزنگاری، از «نظری اطلاعات» (Information Theory) استنتاج می شود: در صورت انتخاب کلید کاملاً تصادفی، هیچ اطلاعاتی از پیام اصلی در پیام رمز شده باقی نخواهد ماند زیرا تمام حروف و سمبولها با احتمال وقوع مشابه در متن رمز شده تکرار خواهند شد.

مثالی از چگونگی رمزنگاری در روش One-Time Pad در شکل ۸-۴ مشاهده می شود. ابتدا پیام ۱ (جمله "I Love You") کاراکتر به کاراکتر به کدهای اسکی هفت بیتی تبدیل می شوند. سپس یک کلید رمز تصادفی [که از این به بعد آن را Pad می نامیم] انتخاب و با پیام XOR می شود تا متن رمز شده بدست آید. یک رمزشکن باید تمام حالات مختلف رشته Pad را آزمایش کند تا ببیند به ازای هر Pad چه متنی حاصل می شود که البته باز هم موفق به یافتن متن اصلی نخواهد شد زیرا به عنوان مثال اگر Pad شماره ۲ با پیام اول XOR شود رشته ای را حاصل خواهد

۱. به عبارت فنی و دقیق تر اگر کلید رمز مناسب انتخاب شود متن رمز شده هیچیک از خصوصیات آماری یک متن معمولی را نخواهد داشت و رمزشکن هیچ راهی برای تحلیل متن و کشف رمز نخواهد داشت. -م

کرد که آن هم متن معمولی و معادل با متن "Elvis Lives" است و احتمالاً به عنوان متن واقعی تلقی و باور خواهد شد که البته اشتباه است. به عبارت بهتر برای هر متن یازده کاراکتری مثال فوق، یک Pad وجود دارد [که پس از XOR شدن با متن رمز] عبارت Elvis Lives (یا هر متن دلخواه دیگر) را تولید خواهد کرد. بنابراین وقتی می‌گوییم که یک متن پس از XOR شدن با یک Pad دلخواه، در خصوص متن اصلی هیچ اطلاعاتی در خود ندارد منظورمان آن است که می‌توانید از متن رمز شده هر پیامی به غیر از پیام اصلی و با طول مشابه استخراج کنید.

پیام ۱:	1001001 0100000 1101100 1101111 1110110 1100101 0100000 1111001 1101111 1110101 0101110
Pad 1:	1010010 1001011 1110010 1010101 1010010 1100011 0001011 0101010 1010111 1100110 0101011
متن رمز:	0011011 1101011 0011110 0111010 0100100 0000110 0101011 1010011 0111000 0010011 0000101
Pad 2:	1011110 0000111 1101000 1010011 1010111 0100110 1000111 0111010 1001110 1110110 1110110
متن آشکار ۲:	1000101 1101100 1110110 1101001 1110011 0100000 1101100 1101001 1110110 1100101 1110011

**شکل ۸-۴.** استفاده از روش One-Time Pad برای رمزنگاری و این امکان که می‌توان

بکمک برخی از Pad ها در متن دلخواهی را از این رشته رمز استخراج کرد.

روش رمزنگاری One-Time Pad اگرچه از دیدگاه تئوری عالی و امن به نظر می‌رسد ولیکن در عمل با اشکالات عمده‌ای مواجه است. به عنوان اولین اشکال، کلید را نمی‌توان بخاطر سپرد و هم گیرنده و هم فرستنده پیام، باید آنرا به صورت نوشته شده با خود حمل کنند. اگر یکی از این دو طرف در معرض حمله فیزیکی یا سرقت کلید قرار داشته باشند، کلیدهایی که در جایی یادداشت شده هرگز مطلوب و قابل اعتماد نخواهد بود. همچنین حجم کل داده‌هایی که می‌تواند ارسال شود به طول کلید مورد استفاده بستگی دارد. مشکل دیگر در این روش، حساسیت به کاراکترهای جا افتاده (گمشده) یا اضافی است زیرا اگر یک کاراکتر از درون متن حذف یا به درون آن اضافه شود از آن محل به بعد، متن قابل رمزگشایی نخواهد بود لذا اگر به هر دلیلی گیرنده و فرستنده هماهنگی خود را از دست بدهند (یا به عبارت بهتر از حالت سنکرون خارج شوند) از آن لحظه به بعد تمام داده‌های رمزگشایی شده غیر قابل استفاده و آشغال خواهد بود.

با ابداع کامپیوترها، ممکن است استفاده از روش رمزنگاری One-Time Pad در برخی از برنامه‌های کاربردی امکان‌پذیر و عملی باشد. (به عنوان مثال) کلید می‌تواند بر روی یک DVD خاص حاوی چندین گیگابایت اطلاعات، ذخیره گردد؛ اگر کلید رمز در پوشش چند دقیقه فیلم ویدیویی بر روی DVD جاسازی شود شک برانگیز نخواهد بود. البته در شبکه‌هایی با سرعت گیگابیت بر ثانیه تغییر DVD مثلاً در هر سی ثانیه (بمنظور تغییر کلید رمز) کاری ملال‌آور و غیرممکن است. در ضمن DVD های حامل کلیدهای رمز باید توسط شخص فرستنده اطلاعات و قبل از ارسال داده‌ها، به گیرنده تسلیم شود، که این کار بشدت از عملی بودن روش خواهد کاست.<sup>۱</sup>

### رمزنگاری کوآنتومی

راه حل جالبی برای مشکل انتقال کلید رمز به روش One-Time Pad بر روی شبکه وجود دارد؛ ابداع این راه حل منشاء عجیب و دور از ذهنی دارد: «مکانیک کوآنتومی»! هر چند این روش هنوز در مراحل تستهای آزمایشگاهی بسر می‌برد ولی آزمایشات اولیه امیدوارکننده بوده است. اگر این روش بتواند تکمیل شده و کارایی آن تضمین گردد، در آینده احتمالاً تمام سیستمهای رمزنگاری براساس روش One-Time Pad شکل خواهد گرفت زیرا این

۱. بزرگترین مشکل روش رمزنگاری One-Time Pad را طولانی بودن طول کلید و حمل و نقل آن، در نظر بگیرید. - ۳

روش کاملاً امن و غیر قابل شکستن است. در زیر بطور مختصر توضیح خواهیم داد که روش رمزنگاری کوآنتومی چگونه کار می کند. بطور خاص پروتکل BB84 را بررسی خواهیم کرد. (BB84 ابتدای نام ابداع کنندگان و سال انتشار مقاله آنهاست - Bennet and Brassard, ۱۹۸۴)

یک کاربر مثل آلینس مایل است با کاربر دوم (مثلاً باب) یک Pad طولانی ایجاد و از آن به عنوان کلید رمز استفاده کند. آلینس و باب که اصطلاحاً طرفین اصلی - Principals - نامیده می شوند بازیگران اصلی داستان ما هستند. به عنوان مثال باب یک بانکدار و آلینس یکی از مشتریان بانک است که می خواهد با باب مراودات تجاری داشته باشد. در چند دهه گذشته در مقالات و کتب رمزنگاری، اسامی «آلینس» و «باب» عموماً به عنوان طرفین مجاز و بازیگران اصلی سناریوهای امنیتی انتخاب شده است. رمزنگارها به سنت پایبند هستند! اگر به جای این دو نام، نامهای آندی و باربارا را به عنوان بازیگران سناریومان انتخاب می کردیم هیچکس مطالب این فصل را باور نمی کرد! پس بگذارید ما هم از این سنت تبعیت کنیم.

اگر آلینس و باب بتوانند یک Pad طولانی به عنوان کلید رمز، ایجاد کنند، قادرند با استفاده از آن، داده ها را به صورت مطمئن و امن مبادله نمایند. سؤال آن است که آنها چگونه می توانند بدون رد و بدل کردن فیزیکی کلید رمز (مثلاً با DVD)، Pad ها را ایجاد و مبادله کنند؟ می توانیم فرض کنیم که آلینس و باب در دو سمت یک فیبرنوری قرار گرفته اند و می توانند پالسهای نوری را ارسال یا دریافت نمایند ولیکن به فرض یک متجاوز به نام تروودی توانسته فیبرنوری را قطع کرده و در آن یک انشعاب فعال ایجاد کند. بدین ترتیب تروودی قادر است تمام بیتها را در دو جهت استراق سمع نماید. او همچنین می تواند پیامهای غلط برای دو طرف ارسال کند. این شرایط اگرچه برای باب و آلینس ناامیدکننده و خطرناک به نظر می رسد ولیکن رمزنگاری کوآنتومی می تواند روزنه امید برای حل این مشکل باشد.

رمزنگاری کوآنتومی بر این اصل استوار است که نور در قالب بسته های کوچکی به نام فوتون با ویژگیهای خاص و عجیب، جابجا می شود. به علاوه وقتی نور از یک فیلتر پلاریزه کننده عبور می کند، پلاریزه (قطبی) می شود. نور پلاریزه شده برای عکاسان یا آنهایی که عینک آفتابی می زنند بخوبی ملموس است. اگر یک پرتو نوری (یا به عبارتی جریان فوتونها) از یک فیلتر پلاریزه کننده عبور کند، تمام فوتونهای پرتو خارج شده از فیلتر، در راستای محور فیلتر (محور عمودی) پلاریزه می شوند. حال اگر این پرتو مجدداً از فیلتر پلاریزه کننده دوم عبور نماید «شدت نور» پرتوی خروجی، متناسب با مربع کسینوس زاویه بین محورهای عمودی دو فیلتر ( $\cos^2\phi$ ) خواهد بود. اگر محورهای دو فیلتر بر هم عمود باشند هیچ یک از فوتونها از بین این دو فیلتر عبور نخواهند کرد. البته زاویه مطلق دو فیلتر در فضای سه بعدی اصلاً مهم نیست بلکه فقط زاویه نسبی محورهای دو فیلتر پلاریزه کننده در محاسبه وارد می شود.

برای تولید یک Pad به عنوان کلید رمز، آلینس نیاز به تنظیم دو زوج فیلتر پلاریزه کننده دارد. زوج فیلتر اول، شامل یک فیلتر عمودی و یک فیلتر افقی است. این انتخاب اصطلاحاً «دستگاه مستقیم» (Rectilinear Basis) نامیده می شود. زوج فیلتر دوم مشابه با قبلی است با این تفاوت که ۴۵ درجه چرخیده است یعنی یکی از فیلترها در امتداد قطر گوشه پایین سمت چپ به گوشه بالا سمت راست قرار گرفته و دیگری عمود بر آن یعنی در امتداد گوشه پایین سمت راست به گوشه بالا سمت چپ قرار دارد. زوج فیلتر دوم اصطلاحاً «دستگاه مورب» (Diagonal Basis) نامیده می شود. بدین ترتیب آلینس دارای دو دستگاه مبنا است که می تواند به انتخاب خود پرتوی نور را از هر کدام از این دستگاههای مبنا (که هر یک شامل دو فیلتر است) عبور بدهد. البته در دنیای واقعی آلینس دارای چهار فیلتر پلاریزه کننده مجزا نیست بلکه فقط یک قطعه بلور خاص (کریستال پلاریزه کننده) وجود دارد که می تواند با سرعت بسیار بالا به هر یک از فیلترهای مورد نظر سوئیچ کند و زاویه پلاریزاسیون را عوض

نماید.<sup>۱</sup> در سمت مقابل، باب نیز از چنین ابزاری استفاده می کند. این واقعیت که آلیس و باب هر کدام دارای دو دستگاه مبنا هستند در رمزنگاری کوآنتومی بسیار اساسی و حیاتی است.

آلیس برای هر یک از دستگاههای مبنا یکی از جهتها را بیت صفر و دیگری را بیت یک فرض می کند. در مثالی که در ادامه مطرح کرده ایم فرض شده که آلیس پلاریزاسیون راستای عمود را بیت صفر و راستای افق را بیت ۱ قرارداد کرده است. همچنین پلاریزاسیون راستای ۴۵ درجه (↗) بیت صفر و راستای ۱۳۵ درجه (↖) بیت ۱ فرض شده است. آلیس قرارداد انتخابی خود را برای باب می فرستد.

حال آلیس یک Pad برای خود انتخاب می کند که این انتخاب می تواند توسط یک مولد اعداد تصادفی انجام شود (موضوعی که به خودی خود مقوله ای پیچیده محسوب می شود). او این Pad را بیت به بیت برای باب می فرستد در حالی که برای ارسال هر بیت بطور تصادفی از یکی از دستگاههای مبنا استفاده می کند.<sup>۲</sup>

برای ارسال یک بیت، تفنگ تولید فوتون قادر است فوتونی را منتشر کند که پلاریزاسیون آن کاملاً منطبق با دستگاه مبنا آلیس و به اختیار او باشد. به عنوان مثال او می تواند بطور متوالی دستگاه مبنای پلاریزاسیون را تغییر داده و دستگاه مورب (Diagonal)، دستگاه مستقیم (Rectilinear) را انتخاب نماید. مثلاً آلیس می تواند برای ارسال یک Pad نظیر 1001110010100110، طبق قرارداد فوق فوتونهای پلاریزه شده شکل ۸-۵ الف را ارسال نماید. با داشتن یک Pad معلوم و مشخص بودن توالی دستگاههای مبنا، پلاریزاسیون مورد استفاده برای هر بیت به صورت یکتا مشخص می شود. «بیتهایی که در قالب یک فوتون منفرد و در یک زمان مشخص ارسال می شوند اصطلاحاً کویت (ها) (Qubit(s) نامیده می شوند.»

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
شماره بیت	0	1	0	1	1	1	0	0	1	0	1	0	0	1	1	0
بیت های داده (الف)	↖	↑	↑	↖	→	↖	↗	↗	→	↑	→	↗	↗	→	→	↑
دستگاه مبنای باب (ب)	+	+	×	×	×	+	+	×	+	×	+	×	×	×	+	×
آنچه باب دریافت می کند (ج)	↑	↑	↗	↖	↖	↑	→	↗	→	↗	→	↗	↗	↖	→	↗
آیا مبناها صحیح بوده اند؟ (د)	No	Yes	No	Yes	No	No	No	Yes	Yes	No	Yes	Yes	Yes	No	Yes	No
One-time pad (ه)		0		1				0	1		1	0	0		1	
دستگاه مبنای ترودی (و)	×	+	+	×	+	+	×	+	+	×	×	+	×	+	×	×
pad ترودی (ز)	x	0	x	1	x	x	x	?	1	x	?	?	0	x	?	x

شکل ۸-۵. مثالی از رمزنگاری کوآنتومی.

۱. به عبارت بهتر یک کریستال به صورت الکترونیکی به گونه ای تنظیم می شود که رفتار هر یک از فیلترهای مورد نظر را بروز بدهد. امروزه کریستالهای پلاریزه کننده الکترونیکی نور موجودند. -م.

۲. یعنی در ارسال PAD به صورت «کاملاً تصادفی» برای بیت ۱ یکی از پلاریزاسیونهای ↗ یا ↑ و برای بیت صفر یکی از پلاریزاسیونهای ↖ یا ← انتخاب می شود. -م.

باب نمی داند که از چه دستگاه مینا [برای دریافت فوتونها] استفاده کند، لذا برای دریافت هر فوتون یکی از دستگاههای قراردادی خود را به صورت تصادفی در نظر گرفته و آن را بکار می گیرد، مثلاً به گونه ای که در شکل ۵-۸-ب مشاهده می کنید. اگر او تصادفاً برای یک بیت، دستگاه مبنایی مطابق با دستگاه مبنای پلاریزاسیون آلیس انتخاب کرده باشد، بیت صحیح دریافت خواهد کرد ولیکن اگر دستگاه مبنای او اشتباه باشد آنگاه به صورت کاملاً تصادفی و با احتمال مساوی یکی از بیتهای ۱ یا صفر دریافت خواهد شد زیرا طبق نظریه مکانیک کوانتومی هر گاه یک فوتون به یک فیلتر پلاریزه کننده با اختلاف زاویه ۴۵ درجه (نسبت به زاویه پلاریزاسیون خود فوتون) برخورد کند بطور تصادفی و با احتمال مساوی، یا به زاویه پلاریزاسیون فیلتر و یا به زاویه قائم بر پلاریزاسیون فیلتر، پخش خواهد کرد. این ویژگی فوتونها در مکانیک کوانتومی یکی از اصول اساسی به شمار می آید. بدین ترتیب هر گاه دستگاه مبنای انتخاب شده توسط باب صحیح نباشد برخی از بیتهای دریافتی صحیح و برخی دیگر اشتباه هستند ولیکن باب نمی داند که کدام صحیح و کدام غلط هستند. نتیجه دریافت بیتها توسط باب در شکل ۵-۸-ب نشان داده شده است.

باب چگونه می تواند متوجه شود که کدام مینا درست و کدام غلط بوده است؟ او به سادگی و به صورت آشکار به آلیس اعلام می کند که دستگاههای مبنای انتخابی او برای هر بیت چه بوده اند و آلیس هم در پاسخ به او خواهد گفت که کدام صحیح و کدام یک غلط است؛ (به گونه ای که در شکل ۵-۸-د ملاحظه می کنید). با این اطلاعات طرفین می توانند یک رشته بیت، مطابق با شکل ۵-۸-ه از حدسهای درست بسازند. بطور میانگین این رشته بیت، نیمی از کل رشته بیت Pad را تشکیل می دهد ولی از آنجایی که طرفین آن را می دانند می تواند به عنوان کلید رمز (یا همان Pad) انتخاب شود. تمام کاری که آلیس مجبور است انجام بدهد آن است که طول Pad انتخابی در ابتدای کار از دو برابر طول مورد نظر بیشتر باشد تا پس از دریافت، طول رشته Pad به اندازه مطلوب و مورد باشد. بدین نحو مسئله حل شده است.

اما لحظه ای تأمل کنید؛ در این میان ترویدی را فراموش کرده ایم. فرض کنید او کنجکاو است بداند که آلیس چه می کند، لذا فیبرنوری را قطع می کند و مدار آشکارساز (Detector) و فرستنده خود را در میانه فیبر جاسازی می کند، [و بدین ترتیب یک انشعاب فعال برای استراق سمع پدید می آورد]. ترویدی نمی داند برای هر فوتون حامل داده چه مبنایی را [برای پلاریزاسیون] در نظر بگیرد. بهترین کاری که می تواند انجام بدهد آن است که برای هر فوتون یک مبنای تصادفی انتخاب کند، دقیقاً همانند کاری که باب می کند. مثالی از انتخابهای تصادفی او در شکل ۵-۸-و نشان داده شده است. بعداً وقتی باب به آلیس (به صورت آشکار) گزارش می دهد که مبنای انتخابی او چه بوده و آلیس نیز به او می گوید که کدام صحیح و کدام غلط است، ترویدی نیز می داند که انتخابهای او کدام صحیح و کدام غلط هستند. مطابق با شکل ۵-۸-و بیتهای ۰ و ۱ و ۲ و ۳ و ۴ و ۶ و ۸ و ۱۲ و ۱۳ را صحیح بدست آورده است ترویدی از پاسخ آلیس به باب متوجه می شود که فقط بیتهای ۱ و ۳ و ۷ و ۸ و ۱۰ و ۱۱ و ۱۲ و ۱۴ جزو Pad (کلید رمز) هستند. برای چهار تا از این بیتها یعنی (۱۲ و ۸ و ۳ و ۱) او حدس درستی زده است و بیتهای صحیحی بدست آورده است. برای چهار بیت دیگر یعنی (۱۴ و ۱۱ و ۱۰ و ۷) او مبنای صحیحی نداشته است و طبیعتاً نمی داند که بیت ارسال شده چه بوده است. بنابراین مطابق با شکل ۵-۸-و باب می داند که Pad او با رشته بیت 01011001 آغاز می شود در حالی که آنچه ترویدی از این Pad در اختیار دارد مطابق با شکل ۵-۸-ز رشته 011?1?0? است.

البته آلیس و باب هر دو آگاهند که ممکن است ترویدی بخشی از Pad آنها را بدست بیاورد، به همین دلیل مایلند اطلاعاتی که ترویدی دارد را کاهش بدهند. لذا آنها می توانند با انجام عملیات تبدیل (Transformation) بر روی Pad این کار را انجام بدهند. به عنوان مثال آنها می توانند Pad طولانی خود را به بلوکهای ۱۰۲۴ بیتی تقسیم

کرده و سپس آن را به توان دو برسانند تا به عددی  $2^{48}$  بیتی تبدیل شود؛ سپس بلوکهای  $2^{48}$  بیتی به هم چسبیده و Pad اصلی (کلید رمز) را تشکیل می‌دهند. در این صورت ترودی با داشتن اطلاعات جزئی از بلوکهای  $2^{48}$  بیتی [زیرا نیمی از بیتها را به درستی نمی‌داند] قادر نیست توان دوی آن را محاسبه کند و بنابراین چیزی از Pad واقعی نخواهد فهمید. انجام تبدیل بر روی Pad اولیه، (به منظور کاهش اطلاعات ترودی)، اصطلاحاً «تشدید پنهان‌سازی»<sup>۱</sup> نامیده می‌شود. در دنیای عمل، به جای آن که بلوکهای Pad به توان دو برسد، عملیات پیچیده‌ای بر روی بلوکهای ورودی انجام می‌شود تا هر بیت خروجی به هر بیت ورودی وابستگی داشته باشد. [بدین ترتیب هر گونه حدس غلط در مورد یک بیت، پس از انجام عملیات تبدیل منجر به خطاهای متعدد در Pad نهایی خواهد شد.]

ترودی بی‌نواانه تنها هیچ حدسی در مورد Pad در اختیار ندارد بلکه حضور او [و استراق سمع داده‌ها] مهم و محرمانه نیست. گذشته از این، او باید بیتهایی را که از آلیس دریافت می‌کند مجدداً برای باب تقویت و ارسال (رله) نماید تا وانمود کند که باب در حال محاوره مستقیم با آلیس است [تا حضورش کماکان مخفی بماند]. مشکل آن است که بهترین کاری که ترودی می‌تواند انجام بدهد آن است که کویتهای دریافتی (Qubits) را مطابق با پلاریزاسیون فرضی خودش برای باب ارسال کند و چون بطور میانگین نیمی از آنچه ارسال می‌کند [به دلیل اشتباه در دستگاه مبنای پلاریزاسیون] غلط است لذا حجم بیتهای اشتباه در Pad متعلق به باب بسیار زیاد خواهد شد.

وقتی آلیس شروع به ارسال داده‌ها می‌کند آن را با «کدهای تصحیح خطای پیشرونده»<sup>۲</sup> کدگذاری می‌نماید. از دیدگاه باب یک بیت خطا در Pad معادل با یک بیت خطای انتقال در داده‌ها تلقی می‌شود. به هر حال او به دلیل خطا در Pad یک بیت را به اشتباه دریافت می‌کند. اگر بیتهای تصحیح خطا به همراه داده ارسال شوند او قادر خواهد بود علیرغم وجود خطا، اصل پیام را بازسازی نماید و در عین حال براحتی می‌تواند تعداد خطاهای تصحیح شده را بشمارد. اگر تعداد آنها از حد انتظار بیشتر باشد باب متوجه خواهد شد که ترودی (به عنوان شخص ثالث و مزاحم) در کانال انشعاب ایجاد کرده است و در این حالت می‌تواند طبق دستور عمل نماید؛ (مثلاً به آلیس بگوید که به کانال رادیویی سوئیچ کند یا مستقیماً با پلیس تماس بگیرد و...) البته اگر ترودی قادر باشد یک فوتون را عیناً (مشابه با فوتون دریافتی) تولید نماید می‌تواند فوتونی را دریافت و عین آن را برای باب ارسال کند و بدین ترتیب حضورش کشف نخواهد شد ولیکن هنوز هیچ راهی برای «تولید مثل»<sup>۳</sup> فوتونها شناخته نشده است. ولیکن حتی اگر ترودی بتواند فوتونها را تولید مثل نماید از ارزشهای رمزنگاری کوآنتومی برای ایجاد Pad (کلید رمز) کاسته نخواهد شد.

اگرچه نشان داده شده که رمزنگاری کوآنتومی بر روی یک فیبرنوری به طول ۶۰ کیلومتر بخوبی کار می‌کند ولیکن ابزارهای لازم بسیار گران و پیچیده هستند. این نظریه هنوز هم امیدبخش است. برای اطلاعات بیشتر در خصوص رمزنگاری کوآنتومی به کتاب Mullins (۲۰۰۲) مراجعه کنید.

## ۵-۱-۸ دواصل اساسی در رمزنگاری

هر چند در صفحاتی که پیش رو دارید چندین سیستم رمزنگاری مختلف را بررسی خواهیم کرد ولیکن فهم دو اصل اساسی که تمام آنها بر آن استوار هستند اهمیت فراوان دارد.

### افزونگی (Redundancy)

اولین اصل آن است که تمام پیامهای رمز شده باید شامل مقداری «افزونگی» [داده‌های زائد] باشند؛ به عبارت دیگر

لزو می ندارد که اطلاعات واقعی به همانگونه که هستند رمز و ارسال شوند.<sup>۱</sup> یک مثال می تواند به فهم دلیل این نیاز کمک کند. فرض کنید یک شرکت به نام TCP (The Couch Potato) با ۶۰۰۰۰ کالا، از طریق سیستم پست الکترونیکی سفارش خرید می پذیرد. برنامه نویسان شرکت TCP به خیال آن که برنامه های مؤثر و کارآمدی می نویسند، پیامهای سفارش کالا را مشتمل بر ۱۶ بایت نام مشتری و به دنبال آن سه بایت فیلد داده (شامل یک بایت برای تعداد کالا و دو بایت برای شماره کالا) در نظر می گیرند که سه بایت آخر توسط یک کلید بسیار طولانی رمزنگاری می شود و این کلید را فقط مشتری و شرکت TCP می داند.

در نگاه اول ممکن است این طرح مطمئن و امن به نظر برسد، از آن جهت که یک اخلالگر غیرفعال (Passive Intruder) به هیچوجه قادر به رمزگشایی اطلاعات نخواهد بود. ولی متأسفانه در این منطق یک اشتباه اساسی وجود دارد که عملاً آن را به طرحی غیرقابل استفاده تبدیل می کند. فرض کنید یک کارمند اخراجی کینه جو می خواهد شرکت TCP را تنبیه کند. لذا قبل از ترک شرکت فهرست مشتریان این شرکت را با خود به همراه می برد. [فهرست مشتریان محرمانه و سری نیست و رمزنگاری نیز نمی شود.] او در طول شب برنامه ای می نویسد تا با استفاده از نامهای واقعی مشتریان سفارشات جعلی و دروغین بدهد. از آنجایی که او فهرست کلیدهای رمز را در اختیار ندارد لذا در سه بایت آخر مقادیری تصادفی قرار می دهد و بدین طریق صدها سفارش جعلی برای شرکت TCP ارسال می کند.

وقتی این پیامها دریافت می شوند کامپیوتر شرکت TCP ابتدا کلید مربوط به هر مشتری را پیدا می کند تا بدنه پیام را رمزگشایی کند. از آنجایی که متأسفانه تمام مقادیر این سه بایت معتبر هستند [یعنی مقادیر تصادفی درون آن پس از رمزگشایی به یک مقدار تصادفی ولی معتبر تبدیل می شود] بنابراین کامپیوتر، شروع به چاپ سفارشهای خرید و صورتحساب می کند. گرچه شاید عجیب به نظر برسد که یک مشتری ۸۳۷ عدد تاب برای بچه ها یا ۵۴۰ جفجغه سفارش بدهد ولی براساس دانشی که یک کامپیوتر دارد شاید خریدار، این مقدار تاب را برای ساخت یک شهر بازی در نظر داشته است! بدین ترتیب یک اخلالگر فعال (کارمند اخراجی) توانسته مشکلات بزرگی را برای شرکت TCP ایجاد کند هر چند خود اخلالگر نمی تواند بفهمد پیامهایی که کامپیوتر او تولید کرده چه هستند!

این مسئله را می توان با اضافه کردن مقداری افزونگی به تمام پیامها حل کرد. به عنوان مثال اگر فیلد سفارش در هر پیام از ۳ بایت به ۱۲ بایت افزایش یابد و نه بایت اول آن باید صفر باشد، آنگاه حمله فوق عملی نخواهد بود زیرا کارمند اخراجی [یا تولید اعداد تصادفی ۱۲ بایتی] قادر نخواهد بود یک حجم عظیم از سفارشهای معتبر تولید نماید.<sup>۳</sup>

حقیقت قضیه آن است که تمام پیامها باید مقدار قابل توجهی افزونگی (Redundancy) داشته باشند بگونه ای که یک اخلالگر فعال (Active Intruder) نتواند پیامهای تصادفی بی معنا تولید و ارسال کند و باعث شود این پیامها به عنوان پیامهای معتبر تفسیر شوند.

با این وجود اضافه کردن مقداری «افزونگی» به پیامها باعث می شود که رمزشکنها ساده تر بتوانند رمز پیامها را بشکنند. فرض کنید که بازار سفارش اجناس از طریق پست الکترونیکی بسیار گرم و رقابتی باشد و رقیب اصلی

۱. یعنی یک رشته رمز شده نباید پس از رمزگشایی معادل با اصل پیام باشد بلکه باید داده های زائد و حساب شده ای در درون آن جاسازی شده باشد. -م.

۲. اصطلاح The Couch Potato واژه ای فکاهی است که معنای کار بیهوده و مسخره دارد و در فارسی در افواه عامه شرکت کشک سانی ترجمه می شود!!!

۳. زیرا آن دسته از اعداد ۱۲ بایتی که پس از رمزگشایی، ۹ بایت اول آنها دقیقاً صفر شود اصلاً مشخص نیستند و تولید آنها به روش سعی و خطا در فضای ۱۲ بایتی ممکن نخواهد بود. -م.

شرکت TCP شرکت Sofa Tuber باشد که بشدت علاقمند است بداند شرکت TCP چند جغجغه می فروشد! در این راستا از خط تلفن شرکت TCP دزدانه انشعاب گرفته است و اطلاعات خط را استراق سمع می کند. در طرح اصلی با سه بایت در بدنه پیام، شکستن رمز و بدست آوردن کلید تقریباً غیر ممکن است زیرا پس از حدس زدن یک کلید، رمز شکن هیچ راهی برای اثبات صحت حدس خود ندارد چرا که تقریباً تمام پیامها معتبرند. در طرح جدید با پیامهای ۱۲ بیتی، رمز شکن به سادگی می تواند پیامهای معتبر را از پیامهای غیر معتبر تشخیص بدهد. [زیرا از بین حدسهایی که در مورد کلید رمز آزمایش می کند حدسی درست است که پیامی با ۹ بایت صفر در ابتدای آن تولید کند.] بهر حال وجود افزونگی الزامی است و داریم:

### اصل اساسی ۱ در رمزنگاری: پیامها باید شامل مقداری افزونگی باشند.

به عبارت دیگر پس از رمزگشایی پیام، گیرنده باید بتواند پیامهای معتبر را با یک بررسی و محاسبه ساده [از پیامهایی که به صورت تصادفی تولید شده اند] تشخیص بدهد. این افزونگی بدان جهت نیاز است که از ارسال پیامهای بی ارزش اخلالگران و فریب خوردن گیرنده در رمزگشایی پیامها و پردازش آنها جلوگیری شود. ولیکن همین افزونگی، شکستن سیستم رمز توسط اخلالگران غیر فعال را ساده تر می سازد؛ لذا در اینجا یک تناقض پدید می آید. بعلاوه افزونگی اطلاعات نباید در قالب اضافه کردن تعداد  $n$  تا صفر به ابتدا یا انتهای پیام، انجام بگیرد چرا که اجرای الگوریتمهای رمزنگاری بر روی چنین پیامهایی، نتایج قابل پیش بینی برخواهد گرداند و کار رمز شکن را ساده تر می کند. اضافه کردن یک چند جمله ای CRC به جای دنباله ای از صفرها بهتر خواهد بود زیرا از یک طرف گیرنده اصلی براحتی می تواند صحت پیامها را بررسی کند و از طرف دیگر رمز شکن را با حجم کار بسیار زیاد مواجه می کند [تا عمل رمز شکنی ناموفق باشد]. حتی بهتر از آن استفاده از «توابع درهم سازی رمز» (Hash) است، مفهومی که بعداً آن را بررسی خواهیم کرد.

اگر برای لحظاتی به رمزنگاری کوآنتومی برگردیم، می توانیم به نقشی که «افزونگی» در آن سیستم ایفاء می کند پی ببریم. به دلیل استراق سمع فوتونها توسط ترودی، برخی از بیتها در Pad انتخابی باب، غلط خواهد بود. بنابراین باب نیازمند به افزونگی در پیامهای دریافتی خود است تا بتواند خطاهای موجود در آن را تشخیص بدهد. یک روش خام و بسیار ضعیف برای ایجاد افزونگی در پیام آن است که پیام دو بار تکرار شود. اگر دو نسخه تکراری پیام مشابه نباشند، باب متوجه خواهد شد که یا فیبرنوری بشدت نویزی است و یا آن که شخصی در حال استراق سمع از روی کانال انتقال است. البته دو بار ارسال کردن یک پیام واحد بسیار غیر منطقی و بیهوده است لذا اضافه کردن «کدهای همینگ» (Hamming Code) یا کدهای «رید سولومون» (Reed-Solomon Code) راه مؤثر و کارآمدتری برای کشف و تصحیح خطا محسوب می شود. ولی بهر حال روشن است که به منظور تشخیص پیامهای معتبر از پیامهای ساختگی به مقداری افزونگی نیاز خواهد بود، مخصوصاً وقتی پای یک اخلالگر فعال (Active Intruder) در میان است.

### تازگی پیامها (Freshness)

دومین اصل اساسی در رمزنگاری آن است که باید محاسباتی صورت بگیرد تا مطمئن شویم هر پیام دریافتی تازه و جدید است یا به عبارتی اخیراً فرستاده شده است. این بررسی برای جلوگیری از ارسال مجدد پیامهای قدیمی توسط یک اخلالگر فعال، الزامی است. اگر چنین بررسیهایی انجام نشود کارمند اخراجی ما [در نمایشنامه قبلی] قادر است با ایجاد یک انشعاب مخفی از خط تلفن، پیامهای معتبری را که قبلاً ارسال شده، مکرراً ارسال نماید. [حتی اگر نداند محتوای آن چیست.] این نظریه را می توان در قالب زیر بازگو کرد:

اصل اساسی ۲ در رمزنگاری: به روشهایی نیاز است تا از حملات منجر به تکرار پیام جلوگیری شود.

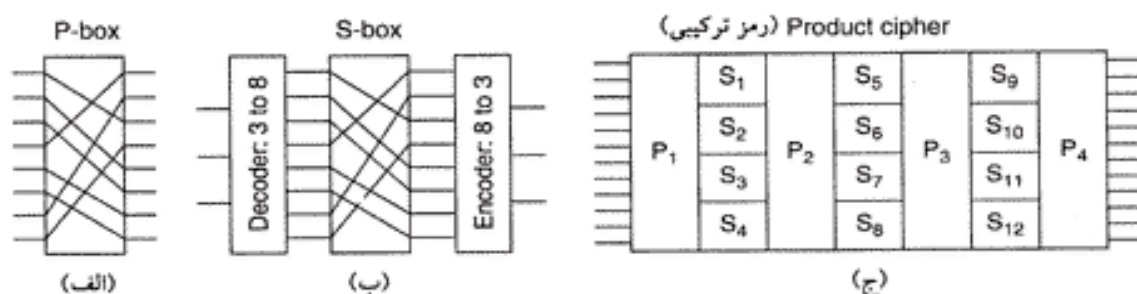
یک چنین محاسبه‌ای را می‌توان با قرار دادن یک «مهر زمان» (Timestamp) در پیامها پیش‌بینی کرد به نحوی که پیام مثلاً برای ده ثانیه معتبر باشد. گیرنده پیام می‌تواند آن را برای حدود ده ثانیه نگه دارد تا بتواند پیامهای جدید را با آن مقایسه کرده و نسخه‌های تکراری را حذف نماید. پیامهایی که بعد از ده ثانیه دریافت شوند کنار گذاشته می‌شوند؛ بدین ترتیب پیامهای تکراری که دارای مهر زمان هستند، به عنوان پیامهای قدیمی شناخته و حذف خواهند شد. به غیر از روش مهر زمان (Timestamp)، روشهای دیگری برای ارزیابی تازگی پیام وجود دارد که در ادامه تشریح خواهند شد.

## ۲-۸ الگوریتمهای رمزنگاری با کلید متقارن (Symmetric-Key)

روشهای پیشرفته و پیچیده رمزنگاری از اصول و قواعدی مشابه با رمزنگاری سستی (مثل روشهای جانشینی و جایگشتی) بهره گرفته‌اند درحالی‌که که راهکارها متفاوت هستند. در قدیم رمزنگاران از الگوریتمهای ساده‌ای استفاده می‌کردند در حالی که امروزه عکس این موضوع صادق است: هدف آن است که یک الگوریتم به قدری پیچیده و بغرنج طراحی شود که حتی اگر رمزشکن توده عظیمی از متن رمز شده را به انتخاب خود در اختیار بگیرد، بدون کلید هرگز نتواند چیزی از بطن آن استخراج کند.

اولین گروه الگوریتمهای پیشرفته رمزنگاری که در درس امروز به بررسی آنها خواهیم پرداخت «الگوریتمهای با کلید متقارن» نام دارد زیرا این الگوریتمها چه برای رمزنگاری و چه برای رمزگشایی از یک کلید مشابه استفاده می‌کنند. شکل ۸-۲ عملکرد یک الگوریتم با کلید متقارن را به تصویر کشیده است. بطور خاص بر روی رمزهای بلوکی متمرکز خواهیم شد که در آنها یک بلوک  $n$  بیتی از «متن آشکار» تحویل الگوریتم شده و براساس کلید تبدیلاتی بر روی آن انجام و یک بلوک  $n$  بیتی «رمز» بدست می‌آید.

الگوریتمهای رمزنگاری را می‌توان هم به صورت سخت‌افزاری (به منظور سرعت بالاتر) و هم به صورت نرم‌افزاری (برای انعطاف‌پذیری بیشتر) پیاده‌سازی کرد. اگرچه توجه ما بیشتر معطوف به الگوریتمها و پروتکلهای رمزنگاری، مستقل از پیاده‌سازی واقعی آنهاست ولی توضیحی کوتاه در خصوص سخت‌افزار رمزنگاری بعضاً جالب و قابل توجه خواهد بود. روشهای جانشینی و جایگشتی می‌توانند با یک مدار ساده الکترونیکی پیاده‌سازی شوند. شکل ۸-۶ الف ابزاری را نشان می‌دهد که به نام P-box (مخفف Permutation یا جایگشت) مشهور است و برای جایگشت بیتهای یک ورودی هشت بیتی کاربرد دارد. در این ساختار اگر بیتهای ورودی را از بالا به پایین با شماره‌های 01234567 شماره‌گذاری کنیم، خروجی این P-box خاص، 36071245 خواهد بود. با سیم‌بندی و برنامه‌ریزی درونی، این P-box قادر است هر گونه جایگشت بیتی را عملاً با سرعتی نزدیک به سرعت نور انجام بدهد؛ چرا که هیچگونه محاسبه‌ای لازم نیست و فقط تأخیر انتشار سیگنال وجود دارد. این طراحی از اصل کرکف تبعیت می‌کند یعنی: حمله‌کننده از روش عمومی جایگشت بیتها مطلع است. آنچه که او از آن خبر ندارد آن است که کدام بیت به کدام بیت نگاشته می‌شود؛ کلید رمز همین است.



شکل ۸-۶. عناصر پایه در رمزنگاری ترکیبی. (الف) P-Box (ب) S-Box (ج) ترکیب این دو.

عمل جانشینی به گونه ای که در شکل ۸-۶-ب نشان داده شده توسط بلوکی به نام S-box انجام می شود. در این مثال یک ورودی سه بیتی به بلوک S-box وارد شده و یک رمز سه بیتی از آن خارج می شود. ورودی سه بیتی، یکی از هشت خط خروجی بخش اول را فعال کرده و آن را به ۱ تنظیم می کند در حالی که بقیه خطوط صفر هستند. بخش دوم یک P-box است. سومین بخش، خط انتخاب شده ورودی را مجدداً در سه بیت گد می کند. منطبق با سیم بندی مثال ۸-۶-ب اگر هشت عدد اکتال (مبنای هشت) ورودی را به ترتیب ۱۲۳۴۵۶۷ در نظر بگیریم، توالی خروجی به ترتیب عبارتند از ۲۴۵۰۶۷۱۳. به عبارت دیگر کد صفر با ۲ جابجا می شود، ۱ با ۴ جابجا می شود و به همین ترتیب. در این ساختار نیز با سیم بندی مناسب در بخش P-box از S-box، هر گونه جانشینی دلخواه ممکن و میسر خواهد بود. به علاوه چنین ابزاری را می توان با سخت افزار پیاده کرد و سرعت بسیار بالایی را بدست آورد زیرا بلوکهای کدکننده (Encoder) و دیکودکننده (Decoder) فقط دارای یک یا دو گیت با تأخیر بسیار ناچیز (کسری از نانوثانیه) است و تأخیر انتشار بخش P-box می تواند کمتر از یک پیکوثانیه باشد.

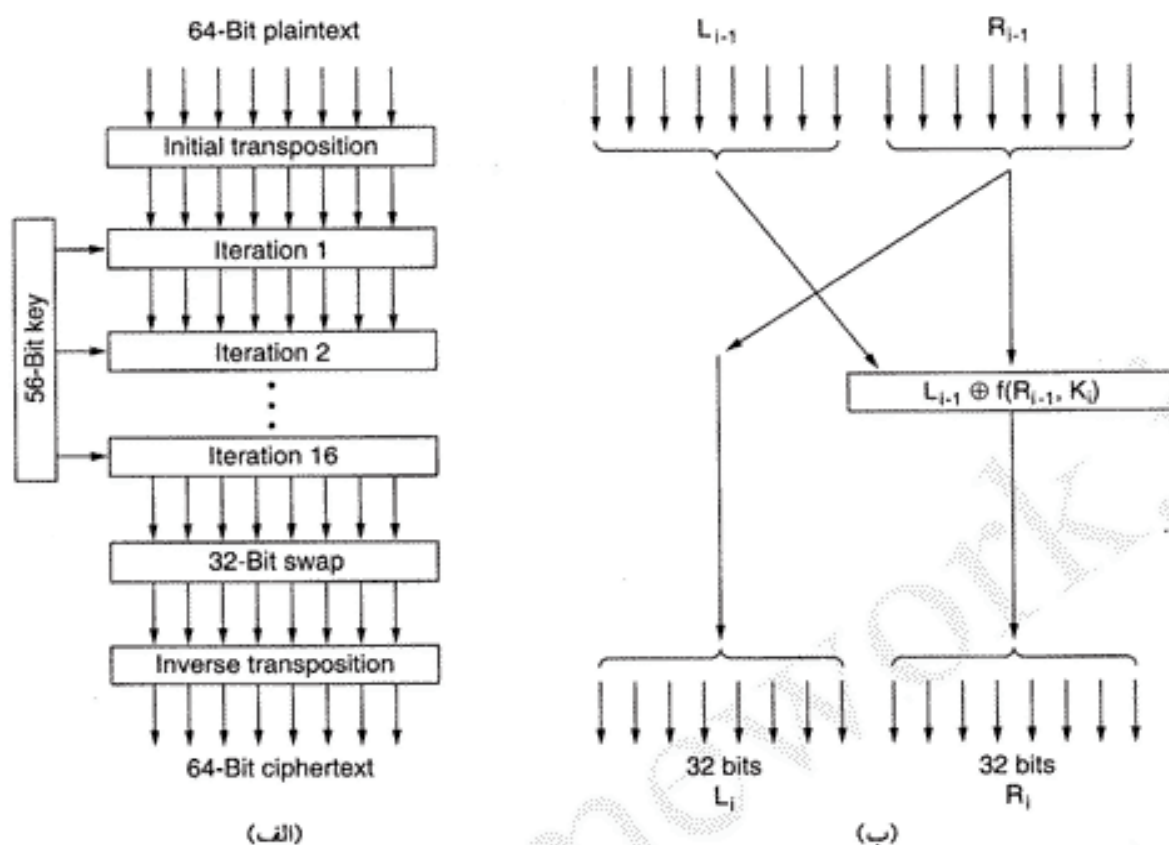
قدرت واقعی این عناصر پایه، زمانی مشخص می شود که بخواهیم با ترکیب تعدادی از این بلوکها همانند شکل ۸-۶-ج، یک سیستم رمزنگار ترکیبی ایجاد نماییم. در این مثال در مرحله اول، به ۱۲ خط ورودی جایگشت داده می شود ( $P_1$ ). در تئوری می توان بلوکی داشت که مستقیماً یک عدد ۱۲ بیتی را به عدد ۱۲ بیتی دیگر بنگارد ولی در عمل چنین ابزاری به  $2^{12}$  یعنی ۴۰۹۶ اتصال متقاطع (در بخش دوم) نیاز خواهد داشت. بجای آن، ورودیها به چهار دسته سه بیتی شکسته شده و هر یک از این سه بیت به صورت مستقل جایگشت داده می شود. اگرچه این روش کمتر معمول است ولیکن در نوع خود قدرتمند بشمار می رود. با قرار دادن تعداد زیاد و کافی از این بلوکها در ساختار ترکیبی فوق، می توان سیستمی را پیاده کرد که خروجی آن (به عنوان تابعی از ورودی) به قدر کافی پیچیده و بهم ریخته باشد.

امروزه سیستمهای رمزنگار ترکیبی که بر روی k بیت ورودی عمل کرده و k بیت خروجی تولید می کنند بسیار رایج هستند. بطور معمول k بین ۱۶ تا ۲۵۶ متغیر است. یک سخت افزار رمزنگار ترکیبی برخلاف شکل ۸-۶-ج به جای هفت بخش حداقل هجده بخش فیزیکی متوالی دارد. پیاده سازی نرم افزاری این روش، در قالب یک حلقه با حداقل هشت تکرار، برنامه نویسی می شود و در هر تکرار عملکرد یکی از S-boxها بر روی یک بلوک ۶۴ تا ۲۵۶ بیتی انجام و سپس عملیات جایگشت محاسبه می شود. اغلب در شروع، یک جایگشت ابتدایی بر روی داده ها انجام می شود سپس عملیات رمزنگاری انجام شده و در پایان نیز یک مرحله جایگشت تکمیلی انجام می گیرد. در ادبیات رمزنگاری هر مرحله تکرار یک «دور» (Round) نامیده می شود.

## ۱-۲-۸ رمزنگاری (Data Encryption Standard) DES

در ژانویه ۱۹۷۷، دولت ایالات متحده یک سیستم رمزنگاری ترکیبی را که توسط IBM طراحی شده بود به عنوان استاندارد رمزنگاری اطلاعات و اسناد رسمی و طبقه بندی نشده خود پذیرفت. از این رمز یعنی DES، بطور گسترده ای توسط صنایع در محصولات امنیتی استفاده شد. البته این سیستم با شکل اولیه و اصلی آن چندان امن نیست ولی شکل اصلاح شده آن هنوز هم سودمند است. در این مبحث به گونه ای عملکرد DES را تشریح خواهیم کرد.

طرح کلی سیستم DES در شکل ۸-۷-الف نشان داده شده است. متن آشکار در قالب بلوکهای ۶۴ بیتی (۸ کاراکتری) رمزنگاری می شود و نهایتاً متن ۶۴ بیتی رمز در خروجی بدست می آید. این الگوریتم رمزنگاری که دارای یک پارامتر ۵۶ بیتی به عنوان کلید است، عملیات لازم را در ۱۹ مرحله مجزا انجام می دهد. اولین مرحله، شامل یک جایگشت بر روی متن ۶۴ بیتی ورودی است که این جایگشت مستقل از کلید رمز است [یعنی کلید در این مرحله وارد نخواهد شد و جدول جایگشت ثابت است]. آخرین مرحله (مرحله نوزدهم) دقیقاً عکس این



شکل ۸-۷. استاندارد DES (الف) الگوی کلی (ب) جزئیات یکی از مراحل تکرار. علامت  $\oplus$  بمعنای XOR است.

جایگشت انجام می‌گیرد. مرحله قبل از آخر نیز، جابجایی ۳۲ بیت سمت چپ با ۳۲ بیت سمت راست می‌باشد. ۱۶ مرحله باقیمانده از لحاظ عملکرد دقیقاً مشابهند، با این تفاوت که در هر مرحله، از پارامتری متفاوت که براساس کلید تعیین می‌گردد، استفاده شده است. الگوریتم به گونه‌ای طراحی شده که اجازه می‌دهد رمزگشایی اطلاعات توسط همان کلیدی انجام شود که رمزنگاری نیز توسط آن انجام شده بود؛ خصوصیتی که در تمام الگوریتمهای با کلید متقارن مورد نیاز است. مراحل رمزگشایی اطلاعات دقیقاً برعکس مراحل رمزنگاری است.

عملکرد یکی از مراحل میانی، در شکل ۸-۷-ب نشان داده شده است. در هر مرحله، دو ورودی ۳۲ بیتی وارد و دو رشته ۳۲ بیتی خروجی، تولید می‌شود. رشته ۳۲ بیتی سمت چپ در خروجی دقیقاً مشابه رشته ۳۲ بیتی سمت راست است. رشته ۳۲ بیتی سمت راست حاصل عمل XOR رشته ورودی سمت چپ با تابعی از رشته سمت راست و کلید این مرحله یعنی  $K_i$  است. تمام پیچیدگی روش، در این تابع نهفته است.

تابع  $f$  شامل چهار گام متوالی است: ابتدا با توسعه عدد ۳۲ بیتی  $R_{i-1}$  (رشته ۳۲ بیتی سمت راست ورودی) یک عدد ۴۸ بیتی به نام  $E$  بدست می‌آید که براساس یک جایگشت ساده و تکرار برخی از بیتها انجام می‌شود.<sup>۱</sup> در مرحله دوم  $E$  با یکدیگر XOR می‌شوند.  $[E]$  حاصل مرحله قبل و  $K_i$  کلید این مرحله است که از کلید ۵۶ بیتی اصلی بدست می‌آید. خروجی این مرحله به هشت گروه شش بیتی تقسیم شده و هر یک از آنها به یک S-box خاص وارد می‌شوند. هر یک از ۶۴ حالت ممکن ورودی، توسط S-box به یک خروجی چهار بیتی نگاشته می‌شود. نهایتاً هشت خروجی چهار بیتی (جمعاً ۳۲ بیت) از درون یک P-box (بمنظور جایگشت بیتی) گذر داده می‌شود.

۱. به ازای هر ۴ بیت، دو بیت تکراری (به ابتدا و انتهای آن چهار بیت) اضافه می‌شود. رجوع کنید به کتاب استالینگ. -م.

در هر یک از ۱۶ مرحله، الگوریتم فوق ثابت است ولی کلید متفاوتی بکار گرفته می شود. قبل از آن که الگوریتم آغاز شود، یک جایگشت ۵۶ بیتی بر روی کلید اعمال می شود [تا بعداً از این کلید ۵۶ بیتی، شانزده کلید رمز ۴۸ بیتی برای هر مرحله بدست آید]. دقیقاً قبل از شروع هر مرحله، کلید به دو بخش ۲۸ بیتی تقسیم شده و هر یک از این بخشها برحسب شماره مرحله، به سمت چپ شیفت گردشی (Rotation) داده می شوند. [مثلاً برای کلید مرحله پنجم هر بخش ۲۸ بیتی، پنج بیت به سمت چپ شیفت گردشی داده می شود]. پس از این مرحله برای محاسبه هر کلید یعنی  $K_i$  یک جایگشت ۵۶ بیتی دیگر بر روی آن اعمال شده و نهایتاً یک زیرمجموعه چهل و هشت بیتی از این ۵۶ بیت استخراج و در هر مرحله از آن استفاده می شود.

تکنیکی که برخی از اوقات برای قدرتمندتر کردن سیستم DES بکار می رود اصطلاحاً «سفیدسازی» (Whitening) نام گرفته است. برای این کار هر بلوک ۶۴ بیتی ورودی به سیستم DES، ابتدا با یک کلید ۶۴ بیتی تصادفی XOR می شود؛ سپس خروجی DES مجدداً با کلید دوم XOR می گردد. عمل «سفیدسازی» به سادگی برگشت پذیر است و برای این کار عکس عملیات فوق انجام می شود (به شرط آن که کلید سفیدسازی در اختیار گیرنده باشد). از آنجایی که این تکنیک از دو کلید ۶۴ بیتی اضافی استفاده می کند، طول کلیدها افزایش خواهد یافت، لذا فضای جستجوی کلید (به روش سعی و خطا) را افزایش داده و رمزشکنی آن را بسیار وقتگیر خواهد کرد. دقت کنید که برای تمام بلوکهای ۶۴ بیتی از یک کلید سفیدسازی مشترک استفاده می شود. (به عبارت بهتر تنها یک کلید سفیدسازی وجود دارد).

از زمانی که DES معرفی و بکار گرفته شد بحث و مناقشات گسترده ای پیرامون آن در گرفت. این سیستم مبتنی بر یک روش رمزنگاری به نام «لوسیفر» (Lucifer) است که IBM آن را طراحی و ثبت کرده بود با این تفاوت که IBM در آن از کلید رمز ۱۲۸ بیتی به جای ۵۶ بیتی استفاده می کرد. وقتی دولت فدرال ایالات متحده خواست که یک سیستم رمزنگاری را برای پرونده های طبقه بندی نشده استاندارد کند، IBM را دعوت کرد تا روش خود را برای NSA، تشریح کند. (آژانس امنیت ملی یا NSA در دولت ایالات متحده، بزرگترین مجموعه ریاضیدانان و رمزشکنهای دنیاست). NSA گروهی به شدت سری است تا جایی که در مورد آن یک لطیفه وجود دارد:

س. NSA مخفف چیست؟

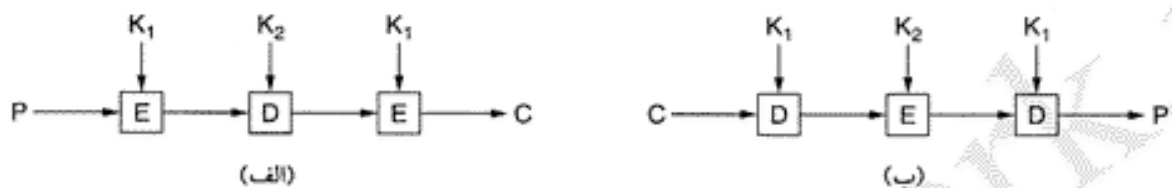
ج. مخفف No Such Agency به معنای «اصلاً چنین آژانسی وجود ندارد!» اما در واقع NSA مخفف کلمات National Security Agency (آژانس امنیت ملی) است.

پس از تشریح روش، IBM کلید ۱۲۸ بیتی را به ۵۶ بیت کاهش داد و تصمیم گرفت فرآیند طراحی DES را محرمانه نگه دارد. بسیاری از افراد شک کردند که شاید دلیل کاهش طول کلید آن بوده که NSA بتواند در صورت لزوم رمز DES را بشکند در حالی که هیچ سازمانی با بودجه کمتر قادر به چنین کاری نباشد. از طرفی محرمانه بودن طرح، احتمالاً بدان دلیل بوده که در آن یک رخنه عمدی گذاشته شده تا NSA راحتتر بتواند رمز DES را بشکند. وقتی یکی از ماموران NSA به صورت محرمانه و غیررسمی از IEEE خواست که برنامه کنفرانس رمزنگاری خود را لغو نماید تا کار عموم را برای رمزشکنی راحتتر از قبل نکند، دامنه این شایعات قوت گرفت. بعداً NSA همه چیز را انکار کرد!

در سال ۱۹۷۷ دو محقق رمزنگاری از دانشگاه استنفورد به نامهای «دیفی» و «هلمن» ماشینی برای شکستن رمز DES طراحی کردند و تخمین زدند که این ماشین با هزینه ای حدود بیست میلیون دلار قابل ساخت است. این ماشین می توانست با داشتن یک قطعه کوچک از متن آشکار و متن رمز شده، کلید رمز را از بین  $2^{56}$  حالت مختلف، در زمانی کمتر از یک روز پیدا کند. امروزه چنین ماشینی زیر یک میلیون دلار قیمت دارد.

## Triple DES (رمزنگاری سه گانه)

در همان آوان ۱۹۷۹، IBM به این حقیقت پی برده بود که طول کلید در سیستم DES بیش از اندازه کوتاه است و روشی برای افزایش مؤثر طول کلید با استفاده از «رمزنگاری سه گانه» ابداع کرد. (Tuchman 1979) روش انتخابی آنها که بعداً در قالب استاندارد بین المللی ۸۷۳۲ (IS 8732) معرفی شد، در شکل ۸-۸ نشان داده شده است. در این شکل از دو کلید و سه مرحله پردازش استفاده شده است. در مرحله اول، متن اصلی با کلید  $K_1$  و مبتنی بر روش معمولی DES رمزنگاری می شود. در مرحله بعدی DES در حالت رمزگشایی ولی با کلید  $K_2$  بر روی نتیجه مرحله قبل اعمال می شود. نهایتاً بار دیگر رمزنگاری DES با کلید  $K_1$  (کلید اول) انجام می شود.



شکل ۸-۸ (الف) سه بار رمزنگاری یکمک استاندارد DES. (ب) رمزگشایی.

این سبک طراحی بلافاصله دو سؤال را به پیش می کشد. اول آن که چرا به جای سه کلید فقط از دو کلید رمز استفاده شده است؟ ثانیاً چرا به جای سه بار رمزنگاری متوالی (اصطلاحاً  $EEE$ )<sup>۱</sup> از روش «رمزنگاری-رمزگشایی-رمزنگاری» (اصطلاحاً  $EDE$ )<sup>۲</sup> استفاده شده است؟

دلیل آن که فقط از دو کلید استفاده شده است آن است که حتی وسواسی ترین رمزنگاران اذعان دارند که برای کاربردهای تجاری کنونی یک کلید ۱۱۲ بیتی بخوبی کفایت می کند و مطمئن است. (در بین رمزنگاران، وسواس و تردید یک ویژگی ممتاز تلقی می شود نه یک اشکال!) حرکت به سمت کلید ۱۶۸ بیتی [۳ کلید ۵۶ بیتی] در مقایسه با بهره واقعی آن، مشکلات سربار زیادتری در خصوص مدیریت و حمل و نقل کلید اضافی ایجاد خواهد کرد.

دلیل استفاده از روش «رمزنگاری-رمزگشایی-رمزنگاری» ( $EDE$ ) سازگار ماندن آن با سیستم تک کلیدی DES بوده است. هر دو عمل رمزنگاری یا رمزگشایی در حقیقت نگاشت یک عدد ۶۴ بیتی به عدد ۶۴ بیتی دیگر هستند. استفاده از روش  $EDE$  به جای  $EEE$  این امتیاز بزرگ را دارد که یک کامپیوتر که مبتنی بر «DES سه گانه» عمل می کند، براحتی قادر است با انتخاب  $K_1 = K_2$  این سیستم را به DES معمولی تبدیل کرده و با ماشینهایی که سیستم رمزنگاری آنها DES تک کلیدی است محاوره داشته باشد. در آن زمان، این ویژگی اجازه می داد که سیستم «DES سه گانه» به تدریج در محیطها جا بیفتد و با سیستمهای رمزنگاری قدیمی سازگار باشد؛ این خصوصیت اگرچه برای رمزنگاران دانشگاهی اهمیتی ندارد ولی برای شرکت IBM و مشتریان آن بسیار حیاتی بود!

## ۲-۲-۸ استاندارد پیشرفته رمزنگاری: AES

در حالی که DES آرام آرام به پایان عمر خود نزدیک می شد (حتی با ابداع DES سه گانه)، برای NIST<sup>۳</sup> در ایالات متحده (که مسئولیت بهبود استانداردهای دولت فدرال آمریکا را بر عهده گرفته است)، مسجل شد که دولت به یک استاندارد رمزنگاری جدید برای اسناد طبقه بندی نشده خود نیاز مبرم دارد. NIST به فراست از دشمنان DES آگاه بود و نیک می دانست که اگر به یکباره استاندارد جدیدی را معرفی نماید همه آنانی که دستی در رمزنگاری دارند ناخودآگاه فرض را بر آن می گذارند که باز هم آژانس سرویسهای محرمانه ایالات متحده، (NSA) یک رخنه<sup>۴</sup> در

۲. Encrypt-Decrypt-Encrypt

۴. در پشتی یا Backdoor

۱. Encrypt-Encrypt-Encrypt

۳. National Institute of Standard Technology

این سیستم باقی گذاشته است و هر چیزی که با آن رمز شود توسط NSA رمزگشایی و خوانده خواهد شد. در این شرایط هیچکس از استاندارد جدید استفاده نمی کرد و به احتمال زیاد استاندارد جدید نیز در یک مرگ آرام رو به زوال می رفت!

بدین ترتیب NIST در فضای بوروکراسی دولتی، راهکار بسیار جالب و متفاوتی را اتخاذ کرد: او از یک رقابت انتخاباتی در بین رمزنگاران بهره گرفت. در ژانویه ۱۹۹۷ از تمام محققین رمزنگاری دنیا دعوت شد که پیشنهادات خود را برای تدوین یک استاندارد جدید که AES نامگذاری شده بود (استاندارد پیشرفته رمزنگاری) ارسال نمایند. شرایط شرکت در این رقابت عبارت بودند از:

۱. الگوریتم پیشنهادی باید یک سیستم رمز متقارن و بلوکی باشد.
۲. جزئیات طراحی باید مشخص و عمومی باشد.
۳. باید از کلیدهای ۱۲۸، ۱۹۲ و ۲۵۶ بیتی حمایت شود.
۴. پیاده سازی سخت افزاری و نرم افزاری الگوریتم ممکن باشد.
۵. الگوریتم باید عمومی (غیرانحصاری) یا تحت قوانین غیرانحصاری ثبت شده باشد.<sup>۱</sup>

پانزده طرح پیشنهادی قابل توجه ارائه گردید و در همین راستا یک کنفرانس همگانی ترتیب داده شد تا در آن طرحها ارائه شود و شرکت کنندگان تشویق شوند تا اشکالات آنها را یافته و تحلیل کنند. در آگوست ۱۹۹۸، NIST براساس ویژگیهای «امنیت»، «کارایی»، «سادگی»، «قابلیت انعطاف» و «فضای حافظه مورد نیاز برای پیاده سازی» (که در سیستمهای درونکار - Embedded - بسیار مهم است) پنج طرح برگزیده را انتخاب و معرفی کرد. کنفرانسهای زیادی برگزار شد و هر کسی تیری در تاریکی انداخته بود! عاقبت در کنفرانس نهایی یک رای گیری آزادانه انجام شد. برگزیدگان نهایی و امتیازات آنها به ترتیب زیر بود:

- |             |                                                       |
|-------------|-------------------------------------------------------|
| ۱. Rijndael | (توسط John Daemen و Vincent Rijmen) ۸۶ رأی            |
| ۲. Serpent  | (توسط Ross Anderson, Eli Biham و Lars Knudsen) ۵۹ رأی |
| ۳. Twofish  | (توسط تیمی به سرپرستی Bruce Schneier) ۳۱ رأی          |
| ۴. RC6      | (توسط آزمایشگاه RSA) ۲۳ رأی                           |
| ۵. MARS     | (توسط IBM) ۱۳ رأی                                     |

در اکتبر ۲۰۰۰، NIST اعلام کرد که او هم به روش Rijndael رأی می دهد و در نوامبر ۲۰۰۱ روش Rijndael استاندارد دولت ایالات متحده شد و در سند استاندارد FIPS 197 ثبت گردید. به دلیل فضای آزاد شگفت انگیز حاکم بر این رقابت و همچنین ویژگیهای فنی برتر Rijndael و با توجه بدان که تیم پیشنهاددهنده دو رمزنگار جوان بلژیکی بودند (که شائبه وجود رخنه مورد نظر NSA در آن را از اذهان می زداید)، انتظار می رود که Rijndael لااقل برای یک دهه استاندارد رمزنگاری کل دنیا شود. روش Rijndael کم و بیش به صورت «رایج» دال<sup>۱</sup> تلفظ می شود و از نام خانوادگی ابداع کنندگان آن (Rijmen & Daemon) گرفته شده است.

Rijndael از کلید و بلوکهای داده ۱۲۸ یا ۲۵۶ بیتی (در قطعات ۳۲ بیتی) حمایت می کند؛ طول کلید و طول بلوکهای داده می تواند مستقل از هم انتخاب شود. با این وجود استاندارد AES بیان می کند که اندازه بلوک داده باید صرفاً ۱۲۸ بیتی باشد ولی طول کلید می تواند یکی از سه حالت ۱۲۸، ۱۹۲ و ۲۵۶ انتخاب شود. برای کسی که همواره از کلیدهای ۱۹۲ بیتی استفاده می کند استفاده از دو گزینه دیگر AES یعنی یک کلید ۱۲۸ بیتی با بلوک داده ۱۲۸ بیتی و کلید ۲۵۶ بیتی با بلوک داده ۱۲۸ بیتی، او را [در خصوص استفاده از آنها] به تردید خواهد افکند.

۱. یعنی امتیاز آن متعلق به هیچ کس نباشد.

در بحثی که در ادامه، پیرامون این الگوریتم خواهیم داشت فقط گزینه ۱۲۸/۱۲۸ (کلید رمز ۱۲۸ بیتی / بلوک داده ۱۲۸ بیتی) را بررسی کرده ایم زیرا احتمالاً در کاربردهای معمول دنیای اقتصاد، کلید ۱۲۸ بیتی جا خواهد افتاد. کلید ۱۲۸ بیتی یک فضای حالت با  $2^{128}$  ( $= 3 \times 10^{38}$ ) حالت مختلف ایجاد می کند. حتی اگر NSA سفارش ساخت ماشینی با یک میلیارد پردازنده موازی بدهد و هر یک از پردازنده ها بتوانند یک کلید را در یک پیکوثانیه ( $10^{-12}$  Sec) آزمایش کنند، آزمایش تمام این کلیدها حدود  $10^{10}$  سال طول خواهد کشید. در آن زمان خورشید خاموش گشته است و مردم مجبورند نتیجه کار را در زیر نور شمع بخوانند!!

### Rijndael

از دیدگاه ریاضی، رمزنگاری Rijndael مبتنی بر «نظریه میدان گالوا» است که به آن ویژگیهای امنیتی قابل اثبات و مطمئنی بخشیده است. با این وجود می توان آن را با کد زبان C بررسی کرد بدون آن که وارد جزئیات ریاضی آن شد.

همانند DES، رمزنگار Rijndael نیز از روشهای جانشینی (Substitution) و جایگشتی (Permutation) استفاده کرده است. همچنین کل مراحل از چندین «دوره» (Round) تشکیل شده است. تعداد «دوره» بستگی به طول کلید و اندازه بلوک داده خواهد داشت: از ۱۰ دور برای کلید ۱۲۸ بیتی با بلوک داده ۱۲۸ بیتی تا ۱۴ دور برای بزرگترین کلید [۲۵۶ بیتی] و بزرگترین بلوک داده [۲۵۶ بیتی].<sup>۱</sup> ولی برخلاف DES، عملیات بر روی بایتها انجام می گیرد نه بیتها؛ بدین ترتیب پیاده سازی سخت افزاری یا نرم افزاری آن ساده تر و کارآمدتر خواهد بود. کلیات کد این الگوریتم در شکل ۸-۹ آورده شده است.

```
#define LENGTH 16 /* # bytes in data block or key */
#define NROWS 4 /* number of rows in state */
#define NCOLS 4 /* number of columns in state */
#define ROUNDS 10 /* number of iterations */
typedef unsigned char byte; /* unsigned 8-bit integer */

rijndael(byte plaintext[LENGTH], byte ciphertext[LENGTH], byte key[LENGTH])
{
 int r; /* loop index */
 byte state[NROWS][NCOLS]; /* current state */
 struct {byte k[NROWS][NCOLS];} rk[ROUNDS + 1]; /* round keys */
 expand_key(key, rk); /* construct the round keys */
 copy_plaintext_to_state(state, plaintext); /* init current state */
 xor_roundkey_into_state(state, rk[0]); /* XOR key into state */

 for (r = 1; r <= ROUNDS; r++) {
 substitute(state); /* apply S-box to each byte */
 rotate_rows(state); /* rotate row i by i bytes */
 if (r < ROUNDS) mix_columns(state); /* mix function */
 xor_roundkey_into_state(state, rk[r]); /* XOR key into state */
 }
 copy_state_to_ciphertext(ciphertext, state); /* return result */
}
```

شکل ۸-۹. الگوی کلی برنامه Rijndael.

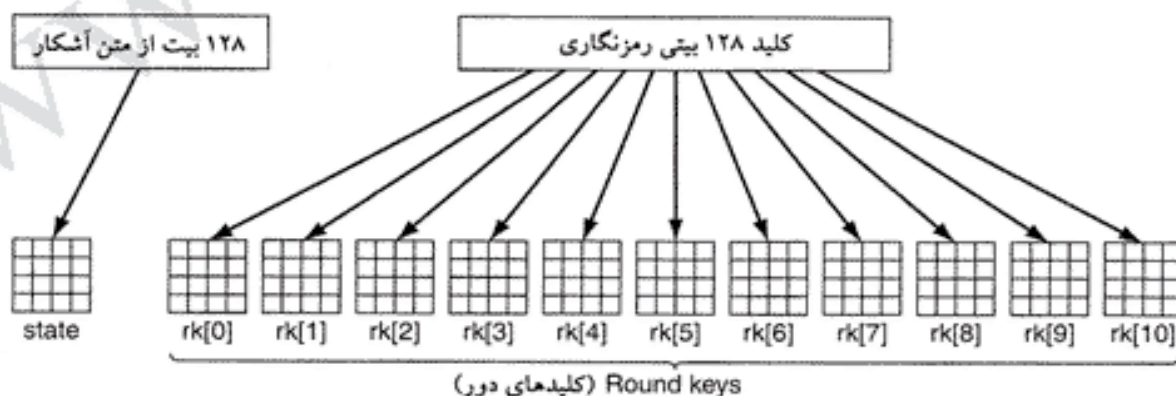
۱. دقت کنید که در رمزنگاری و این دال طول بلوکهای داده می تواند ۲۵۶ بیتی نیز باشد و آنکه در بخش قبلی در مورد AES عنوان شد (که طول بلوک داده باید «صرفاً ۱۲۸ بیتی» باشد) استاندارد NITS است نه الزام طراحان آن م.

تابع *rijndael* سه پارامتر دارد که عبارتند از: (۱) *plaintext*: یک آرایه ۱۶ بیتی محتوی داده های ورودی؛ (۲) *ciphertext*: یک آرایه ۱۶ بیتی که نهایتاً محتوی خروجی رمز شده را باز می گرداند. (۳) *key*: کلید رمز ۱۶ بیتی (۱۲۸ بیتی). در خلال پردازش، حالت فعلی داده ها در یک آرایه دو بعدی به نام *state* که اندازه آن  $NROWS \times NCOLS$  است حفظ می شود. برای بلوکهای ۱۲۸ بیتی داده این آرایه  $4 \times 4$  (یعنی ۱۶ بیتی) است. در این ۱۶ بایت، یک بلوک ۱۲۸ بیتی داده قابل ذخیره خواهد بود.

در ابتدا آرایه *state* با داده های رمز نشده ورودی، مقداردهی اولیه می شود و سپس در هر مرحله از محاسبات مقدار آن تغییر خواهد کرد. در برخی از مراحل فقط جانشینی بایت به بایت انجام می شود و در برخی دیگر بر روی محتویات این آرایه جایگشت بایتها انجام می شود. البته (بغیر از جانشینی و جایگشت) تبدیلات دیگری نیز بر روی بایتها انجام می گیرد. در نهایت محتویات آرایه *state* به عنوان متن رمز شده باز خواهد گشت.

این برنامه با توسعه کلید ۱۲۸ بیتی به یازده آرایه متفاوت و هم اندازه با آرایه *state*، کار خود را آغاز می کند؛ این یازده کلید در آرایه *rk* ذخیره می شوند. *rk* آرایه ای از استراکچر است که به زبان C تعریف شده و هر عضو این آرایه خودش از نوع یک آرایه *state* ( $4 \times 4$  بیتی) است. یکی از این کلیدها در بدو شروع محاسبات مورد استفاده قرار می گیرد و از ده تای دیگر در خلال ده دور پردازش بهره گرفته می شود. به هر یک از این ده کلید، «کلید دور» (Round Key) گفته می شود. محاسبه و استخراج کلیدهای دور بسیار پیچیده است و برای پرهیز از پیچیده شدن اصل موضوع در اینجا بدان نخواهیم پرداخت؛ چراکه از عمومیت کار کاسته نخواهد شد و تشریح آن برای اصل قضیه محوری نیست. فقط به ذکر این نکته بسنده می کنیم که کلیدهای هر «دور» براساس چرخش کلید (Rotation) و XOR کردن آن با گروههای خاصی از بیتهای خود کلید انجام می شود. برای تشریح کامل روش به مرجع (Daemen & Rijmen 2002) مراجعه نمایید.

گام بعدی آن است که متن اصلی در درون آرایه *state* کپی شود تا در خلال ده دور متوالی پردازش شود. عمل کپی درون این آرایه به صورت ستونی انجام می شود یعنی اولین چهار بایت، در ستون اول (ستون شماره صفر)، چهار بایت دوم در ستون دوم (ستون شماره ۱) کپی می شود و به همین ترتیب ادامه می یابد. شماره گذاری ستونها و سطرها از شماره صفر شروع شده است در حالی که مراحل پردازش (دورها) از شماره یک شماره گذاری می شوند. مراحل تنظیم مقداردهی مقدماتی آرایه های  $4 \times 4$  در شکل ۸-۱۰ نشان داده شده است.



شکل ۸-۱۰. ایجاد آرایه های *state* و *rk*.

قبل از شروع محاسبات اصلی، یک کار دیگر نیز انجام می شود: *rk[0]* با آرایه *state* بایت به بایت XOR می شود. به عبارت دیگر هر یک از ۱۶ بایت آرایه *state*، با مقدار حاصل از XOR خودش با بایت متناظر در *rk[0]* تعویض می گردد.

حال زمان شروع محاسبات اصلی فرا رسیده است. حلقه ده بار تکرار می شود (یکبار به ازای هر دور) و در هر تکرار محتوای  $state$  تغییر می کند. هر دور شامل چهار گام است: در گام اول محتوای  $state$  بایت به بایت با مقادیر جدید «جانشین» (Substitute) می شود. در حقیقت محتوای هر بایت به عنوان اندیس ورودی به یک S-box اعمال می شود تا خروجی معادل با خود را تولید نماید. این مرحله یک سیستم جانشینی ساده و کارا کتر به کارا کتر (monoalphabetic) است که در ابتدای این فصل بدان پرداختیم. برخلاف DES که چندین S-box مختلف دارد، در سیستم Rijndael فقط یک S-box وجود دارد.

گام دوم از هر دور، چهار سطر آرایه  $state$  را به سمت چپ می چرخاند: سطر شماره صفر، صفر بایت می چرخد (یعنی تغییر نمی کند)، سطر شماره یک، یک بایت به سمت چپ می چرخد، سطر شماره دو، دو بایت و سطر شماره سه، سه بایت. در این گام محتوای بلوک فعلی آرایه بهم ریخته می شود که معادل با بلوک جایگشت در شکل ۸-۶-الف است.

در گام سوم هر ستون (از آرایه  $state$ ) بطور مستقل از دیگری درهم ریخته می شود. این عمل براساس ضرب ماتریسی انجام می گیرد، بدین نحو که ستون فعلی در یک ماتریس ثابت ضرب شده و ستون جدید را تولید می کند. عمل ضرب ماتریس مبتنی بر نظریه «میدان محدود گالوا»<sup>۱</sup> یعنی  $GF(2^8)$  انجام می شود. اگرچه این فرآیند ممکن است پیچیده به نظر برسد ولی یک الگوریتم ساده در این خصوص وجود دارد که در آن هر یک از ستونهای جدید براساس دو جستجو در جدول نگاشت<sup>۲</sup> و سه عمل XOR محاسبه می شوند. برای کسب اطلاعات تفصیلی به مرجع (Rijndael, Daemen; 2002) مراجعه کنید.

نهایتاً در گام چهارم «کلید دور» با آرایه  $state$  بایت به بایت XOR می شود.

از آنجایی که یکایک مراحل به سادگی برگشت پذیر هستند، لذا عمل رمزگشایی با اجرای برعکس الگوریتم (از آخر به اول) انجام می شود. با این وجود یک راه زیرکانه وجود دارد که در آن عمل رمزگشایی با اجرای همان الگوریتم رمزنگاری ولی با جداول متفاوت انجام می گیرد.

این الگوریتم هم امنیت بسیار بالا و هم سرعت بسیار عالی را تضمین می کند. پیاده سازی نرم افزاری آن بر روی یک ماشین 2-GHz می تواند عمل رمزنگاری هفتصد مگابیت داده در ثانیه را به صورت بلا درنگ انجام بدهد که برای رمزنگاری صد کانال ویدیویی MPEG-2 به صورت همزمان کفایت می کند. پیاده سازی سخت افزاری، از این هم سریعتر خواهد بود.

### ۳-۲-۸ حالات رمز (Cipher Modes)

علیرغم تمام پیچیدگیها، AES (یا DES) هر سیستم رمزنگار که بر روی بلوک محدودی از داده ها عمل می کند) براساس سیستم رمز جانشینی بنیان گذاشته شده است که در آن یک بلوک بزرگ از کاراکترها (۱۲۸ بیتی در AES و ۶۴ بیتی در DES) با یک بلوک جدید جایگزین می شود. هرگاه بلوکهای مشابه از اطلاعات رمز نشده به ورودی این سیستم اعمال شود بلوکهای رمز شده یکسانی تولید خواهد شد. مثلاً اگر شما متن abcdefgh را با یک کلید مشابه در سیستم DES صد بار رمز کنید، صد نتیجه یکسان خواهید گرفت. یک رمز شکن می تواند از این ویژگی برای واژگون کردن رمز (استخراج اطلاعات بدون در اختیار داشتن کلید) سوء استفاده کند.

#### حالت کتابچه رمز (Electronic Code Book Mode)

برای آن که ببینیم در رمزهای جانشینی چگونه از خصوصیت فوق الذکر برای شکستن رمز سوء استفاده می شود،

سیستم رمز «DES سه گانه» (Triple DES) را بررسی می کنیم زیرا نشان دادن بلوکهای ۶۴ بیتی داده از بلوکهای ۱۲۸ بیتی ساده تر است ولیکن سیستم AES نیز دقیقاً همین مشکل را دارد. ساده ترین راه برای رمزنگاری یک قطعه طولانی داده آن است که به قطعات متوالی هشت بایتی (۶۴ بیتی) تقسیم شده و هر یک از این قطعات با کلید مشابه، یکی پس از دیگری رمزنگاری شوند. آخرین قطعه، ممکن است نیاز به اضافه کردن اطلاعات زائد (تا رسیدن به ۸ بایت) داشته باشد. این روش (یعنی قطعه قطعه کردن داده ها به بخشهایی با طول ثابت) ECB Mode<sup>۱</sup> نامیده می شود که مشابه با یک سیستم قدیمی رمزنگاری است که در آن کلمات یک متن تفکیک شده و به جای آنها بکمک کتابچه رمز یک کد پنج رقمی (در مبنای ده) جایگزین می شد تا متن رمز شده را ایجاد نماید.

در شکل ۸-۱۱، ابتدای یک فایل کامپیوتری را مشاهده می کنید که در آن فهرستی از پادشاهای سالانه یک شرکت که قرار است به کارمندان خود اعطاء کند، درج شده است. این فایل از رکوردهای متوالی ۳۲ بایتی تشکیل شده و به ازای هر کارمند یک رکورد، طبق قالب ذیل ذخیره شده است: ۱۶ بایت برای نام، ۸ بایت برای رده شغلی و ۸ بایت برای پاداش. فایل مربوطه که جمعاً شامل ۱۶ بلوک ۸ بایتی است که از شماره صفر تا ۱۵ شماره گذاری شده و بکمک «سیستم DES سه گانه» (Triple DES) رمزنگاری شده است.

نام	جایگاه شغلی	پاداش
A d a m s , L e s l i e	C l e r k	\$ 1 0
B l a c k , R o b i n	B o s s	\$ 5 0 0 0 0 0 0 0
C o l l i n s , K i m	M a n a g e r	\$ 1 0 0 0 0 0 0 0
D a v i s , B o b b i e	J a n i t o r	\$ 5

Bytes ← 16 8 8

شکل ۸-۱۱. متن اصلی از یک فایل که در قالب ۱۶ بلوک پروش DES رمز شده است.

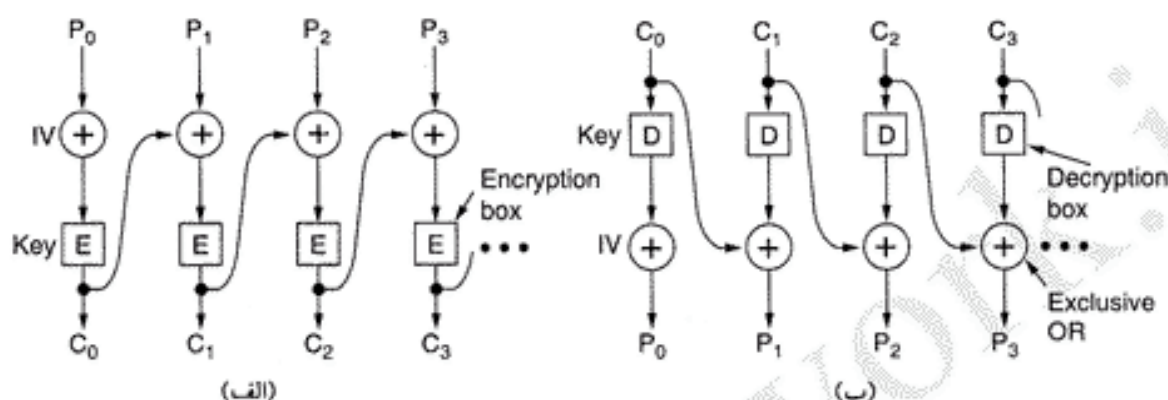
فرض کنید Leslie با رئیسش مشکل دارد و طبعاً انتظار دریافت پاداش چندانی در سر نمی پرورد. در طرف مقابل Kim مورد توجه و عنایت رئیس است و همه این را می دانند. Leslie می تواند به این فایل بعد از رمز شدن ولی قبل از ارسال به بانک دسترسی پیدا کند. آیا Leslie قادر است وضعیت نامناسب پاداش خود را (در حالی که اطلاعات درون فایل رمز شده است) اصلاح نماید؟

هیچ مشکلی برای این کار وجود ندارد! تمام کاری که Leslie باید انجام بدهد آن است که کپی بلوک دوازدهم از متن رمز شده (که شامل پاداش Kim است) را استخراج کرده و آنرا با بلوک چهارم (رکورد پاداش خودش) عوض کند. حتی بدون آن که Leslie بداند در بلوک دوازدهم چه چیزی درج شده است، قطعاً انتظار ایام بسیار شادتری را در کریسمس این سال خواهد داشت! (البته او می تواند بلوک رمز شده هشتم - پاداش رئیس - را برای خودش کپی کند ولیکن به احتمال زیاد قضیه فاش خواهد شد! گذشته از آن Leslie آدم حریص و زیاده طلبی نیست!)

#### حالت زنجیره سازی بلوکهای رمز (Cipher Block Chaining Mode)

برای ختنی کردن این نوع حملات، بلوکهای داده می توانند به صورت زنجیره ای رمز شوند به گونه ای که تغییر یا جابجایی در یک بلوک (همانند کاری که Leslie انجام داد) باعث شود متن رمزگشایی شده از محل دستکاری، به بلوکهای آشغال و بی معنی تبدیل گردد. یکی از روشهای زنجیره سازی، اصطلاحاً روش

Cipher Block Chaining نام دارد. در این روش که در شکل ۸-۱۲ نشان داده شده هر بلوک از اطلاعات اصلی، پیش از رمزنگاری با بلوک رمز شده قبل از خودش XOR می شود. بدین ترتیب بلوکهای یکسان متن به بلوکهای رمز شده مشابه تبدیل نخواهد شد و رمزنگاری از حالت «جانشینی بلوک» خارج خواهد گردید. اولین بلوک با یک مقدار تصادفی به نام IV (Initialization Vector)، XOR می شود و به صورت آشکار به همراه داده های رمز شده ارسال می گردد.

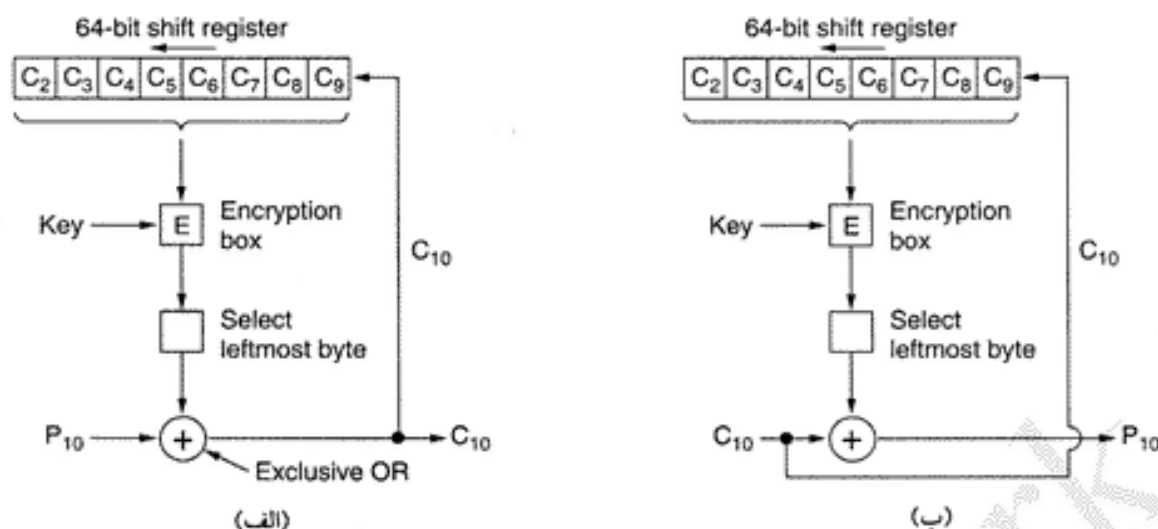


شکل ۸-۱۲. زنجیره سازی بلوکهای رمز. (الف) رمزنگاری. (ب) رمزگشایی.

با بررسی شکل ۸-۱۲ به سادگی می توان چگونگی زنجیره سازی بلوکهای رمز را مشاهده و بررسی کرد. کار را با محاسبه  $C_0 = E(P_0 \text{ XOR } IV)$  شروع می کنیم. سپس بلوک بعدی رمز را با محاسبه  $C_1 = E(P_1 \text{ XOR } C_0)$  بدست آورده و به همین ترتیب ادامه می دهیم. دقت داشته باشید که عمل رمزگشایی بلوک  $i$ ، منوط به آن است که تمام بلوکهای صفر تا  $i-1$  از رمز خارج شده باشند بدین ترتیب بلوکهای مشابه در متن اصلی بسته به محل قرار گرفتنشان، بلوکهای رمز شده کاملاً متفاوتی را ایجاد خواهند کرد. اگر تبدیل یا تغییری در فایل رمز شده انجام گیرد (از نوعی که Leslie در مثال بالا انجام داد)، پس از رمزگشایی فایل، از محل دستکاری به بعد بلوکهای معنی و نامعتبر خواهند بود. برای یک کار آگاه امنیتی زیرک، محلی که از آنجا به بعد این ناهنجاری بوجود آمده (یعنی محلی که پس از رمزگشایی، بلوکهای معنی شده اند) می تواند نقطه شروع خوبی برای آغاز بازرسی و پیگیری باشد! زنجیره سازی بلوکهای رمز [گذشته از خشتی کردن حملاتی نظیر آنچه که در بالا اشاره شد] این حسن بزرگ را دارد که بلوکهای متن، بلوکهای رمز یکسان تولید نخواهند کرد و بالطبع کار تحلیل گر رمز [برای شکستن رمز] بسیار سخت تر خواهد شد. در حقیقت این حسن دلیل اصلی استفاده از آن می باشد.

#### حالت فیدبک رمز (Cipher Feedback Mode)

با وجود این، زنجیره سازی بلوکهای رمز یک اشکال دارد و آن هم این که تا موقعی که یک بلوک ۶۴ بیتی بطور کامل دریافت نشود، رمزگشایی آن [و بلوکهای بعدی حتی در صورت دریافت] ممکن نخواهد بود. استفاده از این روش در یک ترمینال محاوره ای که در آن جا کاربران می توانند خطوط یا فرامین کوتاه تر از هشت کاراکتر درج و ارسال کنند و برای دریافت پاسخ منتظر بمانند، چندان مناسب نیست. برای رمزنگاری بایت به بایت (به نحوی که در شکل ۸-۱۳ می بینید) از روش Cipher Feedback Mode مبتنی بر رمزنگاری «DES سه گانه» (Triple DES) استفاده می شود. برای روش AES نیز دقیقاً همین ایده کارساز خواهد بود با این تفاوت که در آنجا شیفت رجیسترها ۱۲۸ بیتی هستند. در این شکل وضعیت ماشین رمزنگار در حالتی نشان داده شده که قبل از آن بایتهای صفر تا ۹، رمز و ارسال شده اند. وقتی بایت دهم از راه می رسد، (مطابق با شکل ۸-۱۳-الف) الگوریتم DES بر روی محتوای ۶۴ بیتی موجود در شیفت رجیستر اعمال شده و کدهای رمز ۶۴ بیتی در خروجی آماده



شکل ۸-۱۳. حالت فیدبک رمز. (الف) رمزنگاری. (ب) رمزگشایی.

می شوند. سپس بایت سمت چپ این متن رمزی استخراج و با بایت دهم تازه وارد [یعنی  $P_{10}$ ] XOR می شود و بایت حاصل یعنی  $C_{10}$  بر روی خط انتقال ارسال خواهد شد. بعلاوه وقتی  $C_{10}$  ارسال شد، شیفت رجیستر، بایتها را به سمت چپ شیفت داده و  $C_{10}$  در سمت راست شیفت رجیستر قرار می گیرد. دقت کنید که محتوای فعلی شیفت رجیستر به پیشینه کل متن ارسالی وابسته خواهد بود، بدین ترتیب یک الگوی تکراری در متن اصلی به صورت گوناگون در متن رمز شده نگاشته می شود. دقیقاً همانند روش زنجیره سازی بلوکهای متن، در این روش نیز برای شروع گردش عملیات به یک مقدار اولیه (Initialization Vector) نیاز خواهد بود.

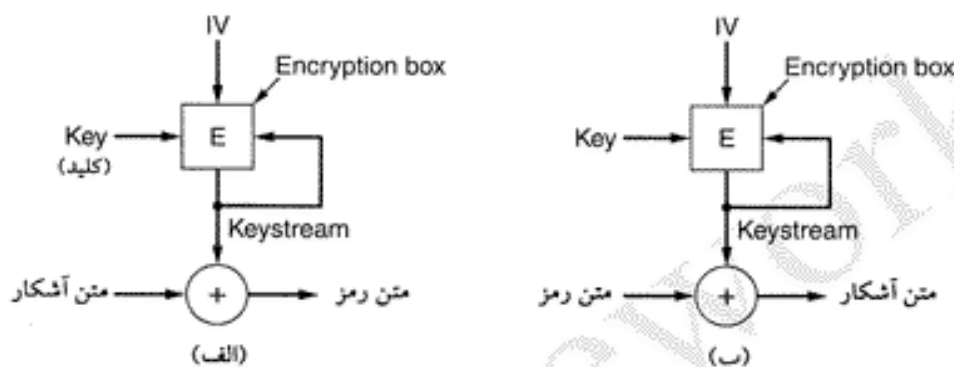
رمزگشایی در روش Cipher Feedback Mode دقیقاً مشابه با عملیات رمزنگاری داده ها است. در اصل محتوای شیفت رجیستر به جای رمزگشایی مجدداً رمزنگاری می شود زیرا هر گاه خروجی رمز شده شیفت رجیستر، با  $C_{10}$  XOR شود  $P_{10}$  را نتیجه خواهد داد چراکه در ابتدا برای بدست آمدن  $C_{10}$  خروجی رمز شده شیفت رجیستر با  $P_{10}$  XOR شده بود [و XOR مجدد آن با  $C_{10}$  کاراکتر  $P_{10}$  را نتیجه خواهد داد]. تا زمانی که دو رجیستر مشابه هم هستند عمل رمزگشایی به درستی انجام می شود. این مفهوم در شکل ۸-۱۳ ب نشان داده شده است.

مشکلی که در روش Cipher Feedback Mode وجود دارد آن است که هر گاه یک بیت از متن رمز شده در خلال ارسال وارونه شود، تا زمانی که این بیت خراب در درون شیفت رجیستر قرار دارد نتیجه رمزگشایی غلط خواهد بود و بدین ترتیب به ازای هر بیت خطا، هشت بایت اشتباه در خروجی پدیدار خواهد شد. هر گاه بیت اشتباه از شیفت رجیستر بیرون بیفتد مجدداً خروجی صحیح تولید خواهد شد. بنابراین تأثیر یک بیت اشتباه کاملاً محلی است و به مابقی پیام سرایت نخواهد کرد بلکه فقط بیتهایی به پهنای شیفت رجیستر را خراب می کند.

### Stream Cipher Mode

علیرغم محلی بودن اثر یک بیت خطا در روش قبل، برنامه های کاربردی خاصی وجود دارند که در آنها سرایت خطای یک بیت به ۶۴ بیت، تأثیر مخرب بسیار زیاد، بجا می گذارد. برای این نوع از برنامه های کاربردی گزینه چهارمی به نام Stream Cipher Mode وجود دارد. این روش با رمز کردن یک مقدار اولیه به نام بردار (Initialization Vector) IV توسط یک کلید کارش را شروع می کند تا یک بلوک خروجی بدست آید. بلوک

خروجی مجدداً رمز می‌شود تا بلوک دوم بدست آید. بلوک دوم نیز مجدداً رمز شده تا بلوک سوم بدست آید و این روال ادامه خواهد یافت. دنباله (طولانی و اختیاری) بلوکهای خروجی که اصطلاحاً Keystream نامیده می‌شود با بلوکهای متن، XOR می‌گردند و بدین ترتیب همانند آنچه که در شکل ۸-۱۴ ملاحظه می‌کنید متن رمز می‌شود.<sup>۱</sup> دقت کنید که IV فقط در مرحله اول مورد استفاده قرار می‌گیرد؛ پس از آن، خروجی هر مرحله رمز می‌شود تا Keystream جدید بدست آید. همچنین توجه کنید که Keystream مستقل از داده‌هاست لذا حتی می‌توان Keystream هر مرحله را با داشتن کلید و بردار IV پیشاپیش محاسبه کرد و بنابراین Keystream نسبت به خطاهایی که در حین انتقال ممکن است اتفاق بیفتد حساس نیست.<sup>۲</sup> عمل رمزگشایی در شکل ۸-۱۴ ب نشان داده شده است.



شکل ۸-۱۴. حالت استریم رمز (Stream Cipher Mode). (الف) رمزنگاری. (ب) رمزگشایی.

رمزگشایی در سمت گیرنده با تولید Keystream‌های مشابه انجام می‌شود. از آنجایی که Keystream فقط به IV و کلید رمز وابسته است، لذا از خطاهای احتمالی که در حین انتقال برای برخی از بیت‌های متن رمز شده اتفاق می‌افتد، تأثیر نمی‌گیرد. بدین ترتیب یک بیت خطا در حین انتقال متن رمز شده، فقط و فقط یک بیت خطا در متن رمزگشایی شده ایجاد خواهد کرد.

نکته حیاتی آنست که در این روش هیچگاه نباید از زوج (کلید، بردار IV) مشابه استفاده شود زیرا این کار منجر به تولید Keystream‌های یکسان خواهد شد. استفاده از Keystream‌های مشابه، متن رمز شده را در معرض آسیب حمله *Keystream reuse attack* قرار می‌دهد: فرض کنید یک بلوک از متن اصلی، مثلاً  $P_0$ ، طبق قاعده  $P_0 \text{ XOR } K_0$  رمز شده باشد. [که در آن  $P_0$  بلوک اول از متن اصلی و  $K_0$  همان Keystream تولید شده در مرحله اول است.] همچنین فرض کنید بعداً برای متن جدید  $Q$  نیز از همین Keystream استفاده شود. بدین ترتیب بلوک  $Q_0$  از متن دوم نیز طبق قاعده  $Q_0 \text{ XOR } K_0$  رمز می‌شود. یک رمز شکن که توانسته بلوکهای رمز شده پیامهای  $P$  و  $Q$  را جداگانه در اختیار بگیرد، این دو بلوک را با هم XOR می‌کند و با اینکار کلید از میان می‌رود.<sup>۳</sup> اگر یکی از بلوکهای  $P_0$  یا  $Q_0$  معلوم باشد یا حدس زده شود، دیگری هم به سادگی بدست می‌آید. به هر تقدیر می‌توان به حاصل XOR دو متن اصلی، طبق روشها و ویژگیهای آماری حمله کرد و آنها را آشکار ساخت. به عنوان مثال در متن انگلیسی، دو کاراکتر فاصله خالی (Space) بیشترین احتمال XOR شدن را دارند. پس از آن XOR شدن

۱. در حقیقت این روش همانند روش رمزنگاری One Time Pad که در ابتدای این فصل تشریح شد عمل می‌کند با این تفاوت که Pad به صورت خودکار، توسط کلید و IV تولید می‌شود. -م

۲. در حقیقت رمزنگاری پیام فقط یک عمل XOR ساده با Keystream‌ها در هر مرحله است. -م

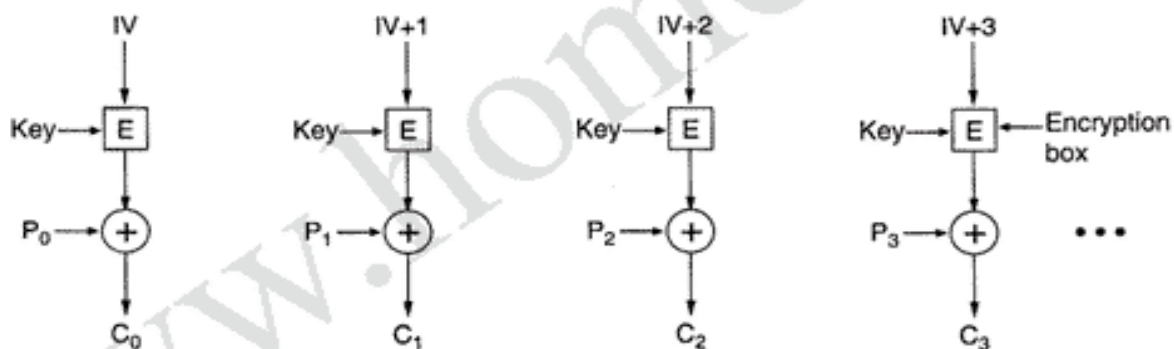
۳. عمل XOR این دو بلوک رمز عبارت است از  $(P_0 \text{ XOR } K_0) \text{ XOR } (Q_0 \text{ XOR } K_0)$  که معادل  $P_0 \text{ XOR } Q_0$  خواهد شد. -م

حرف 'e' با Space بالاترین احتمال را دارد و به همین ترتیب، کوتاه سخن آنکه، با در اختیار داشتن حاصل XOR شده دو متن، رمز شکن اقبال بسیار بلندی برای آشکار کردن هر دوی آنها خواهد داشت.

#### حالت شمارنده (Counter Mode)

یکی از مشکلاتی که تمام روشهای قبل، به استثنای روش Electronic Code Book، دارند آنست که دسترسی تصادفی به داده های رمز شده ممکن نیست. به عنوان مثال فرض کنید یک فایل بر روی شبکه ارسال و به صورت رمز شده بر روی دیسک ذخیره گردد. ذخیره فایلها به صورت رمز شده از این دیدگاه که ماشین گیرنده فایل یک کامپیوتر کیفی است و احتمال دارد سرقت شود، کاملاً منطقی است. ذخیره فایلها به صورت رمز شده خطر فاش شدن اطلاعات را کاهش خواهد داد مخصوصاً زمانی که احتمال دارد یک کامپیوتر بدست افراد ناپاب بیفتد.

ولیکن در عمل، دسترسی به فایل های روی دیسک و بالاخص پایگاههای اطلاعاتی، به صورت غیر ترتیبی (nonsequential) انجام می شود. برای دسترسی به یک بلوک تصادفی از فایلی که به روش «زنجیره سازی بلوکها»<sup>۱</sup> رمزنگاری شده است باید ابتدا تمام بلوکهای قبل از آن رمزگشایی شوند که این فرآیند (از لحاظ زمان دسترسی) بسیار پرهزینه است.<sup>۲</sup> به همین دلیل روش دیگری ابداع شده که ساختار آنرا در شکل ۸-۱۵ مشاهده می کنید. در این روش متن اصلی به صورت مستقیم رمز نمی شود بلکه یک بردار اولیه (IV) که با یک عدد صحیح جمع می شود توسط کلید رمز شده و نتیجه آن با متن اصلی XOR می شود. برای هر بلوک جدید یک واحد به IV اضافه می شود فلذا دسترسی به هر بلوک از داده در هر کجای آن بدون نیاز به رمزگشایی بلوکهای قبلی آن، میسر خواهد بود.



شکل ۸-۱۵. رمزنگاری در حالت شمارنده (Counter Mode)

اگرچه روش Counter Mode سودمند است ولیکن از یک ضعف اساسی رنج می برد که اشاره به آن خالی از لطف نیست. فرض کنید که در آینده [برای رمزنگاری فایلی دیگر] از کلیدی مشابه K استفاده شود (با IV مشابه ولی با متنی متفاوت) و یک رمز شکن این دو متن رمز شده را بدست بیاورد. چون کلید و IV مشابهند بنابراین Keystream ها [یعنی کلیدهایی که برای هر بلوک جدید محاسبه و با آن XOR می شوند] یکسانند و طبعاً سیستم رمز Counter Mode در معرض حمله Keystream Reuse Attack که در بخش قبلی تشریح شد، قرار می گیرد. رمز شکن با XOR کردن یکایک بلوکهای این دو متن، کلید رمز را از میان برمی دارد و حاصل XOR شده دو متن را بدست می آورد. این ضعف بدان معنا نیست که روش Counter Mode، نظریه بدی است، بلکه بدین مفهوم

۱. Cipher Block Chaining.

۲. به عبارت بهتر دسترسی مستقیم به یک بلوک از چنین فایلی هرگز میسر نیست مگر آن که تمام بلوکهای قبل از آن، از رمز خارج شده باشند. -م

است که چه کلید و چه بردار IV باید مستقل از هم و به صورت کاملاً تصادفی انتخاب شود. در این صورت وقتی IV انتخابی متفاوت باشد، حتی اگر از یک کلید مشابه به صورت اتفاقی دو بار استفاده شود، متن رمز شده امن خواهد ماند.

## ۸-۲-۴ رمزهای دیگر

DES و Rijndael، مشهورترین الگوریتمهای رمزنگاری با کلید متقارن هستند ولیکن باید به این نکته دقت داشت که روشهای متعدد دیگری برای رمزنگاری با کلید متقارن ابداع شده است. برخی از این الگوریتمها در درون تولیدات مختلف [مثل کارتهای هوشمند، کدکنندههای ویدیویی یا برخی از نرم افزارها] پیاده سازی شده اند. فهرست رایجترین این روشها را در شکل ۸-۱۶ می بینید.

نام رمز	ابداع کننده	طول کلید	توضیح
Blowfish	Bruce Schneier	1-448 bits	قدیمی است و کند عمل می کند.
DES	IBM	56 bits	برای کاربردهای امروزی بسیار ضعیف است.
IDEA	Massey and Xuejia	128 bits	روش مناسبی است ولی امتیاز آن ثبت شده
RC4	Ronald Rivest	1-2048 bits	احتیاط: برخی از کلیدها ضعیف عمل می کنند!
RC5	Ronald Rivest	128-256 bits	روش مناسبی است ولی امتیاز آن ثبت شده
Rijndael	Daemen and Rijmen	128-256 bits	بهترین گزینه ممکن
Serpent	Anderson, Biham, Knudsen	128-256 bits	بسیار محکم و قوی
Triple DES	IBM	168 bits	دومین گزینه مناسب
Twofish	Bruce Schneier	128-256 bits	بسیار قوی است و کاربرد گسترده ای دارد.

شکل ۸-۱۶. برخی از الگوریتمهای رایج رمزنگاری با کلید متقارن

## ۸-۲-۵ تحلیل رمز (رمزشکنی)

قبل از آن که موضوع رمزنگاری با کلید متقارن را خاتمه بدهیم، اشاره به چهار روش توسعه یافته در تحلیل رمز (رمزشکنی) ارزشمند خواهد بود. اولین روش، «تحلیل رمز به روش تفاضلی» (Differential Cryptanalysis) است. (توسط Biham و Shamir، ۱۹۹۳) این روش می تواند برای حمله به هر سیستم رمز بلوکی به کار گرفته شود. در اینجا، کار تحلیل رمز با انتخاب دو بلوک متن که در تعداد بسیار کمی بیت با هم اختلاف دارند، آغاز می شود؛ سپس با اعمال آنها در الگوریتم رمزنگاری، اتفاقات حاصل از اجرای هر مرحله از الگوریتم، بدقت بررسی می شود. در بسیاری از حالات، برخی از الگوهای بیتی نسبت به الگوهای دیگر بیشتر تکرار می شوند و این نتیجه و مشاهده می تواند در هدایت حمله مبتنی بر احتمال و آمار بسیار مؤثر باشد.

دومین روش، «تحلیل رمز خطی»<sup>۱</sup> است (Matsui, 1994) که می تواند رمز DES را با آزمایش  $2^{43}$  حالت شناخته شده متن، بشکند. این روش با XOR کردن بیتهایی مشخص از متن اصلی و متن رمز شده با یکدیگر و آزمایش مجدد الگوی بدست آمده انجام می شود. هر گاه این روال به تعداد دفعات زیاد تکرار شود، نیمی از بیتها باید صفر و نیم دیگر یک باشند.<sup>۲</sup> ولیکن اغلب نتیجه بدست آمده، نسبت به یکی از جهتها (یعنی بیتهای صفر و

۱. Linear Cryptanalysis

۲ طبق نظریه احتمال، هر گاه متن ورودی هیچگاه به کلید وابستگی آماری نداشته باشد، احتمال ظاهر شدن صفرها و یکها مساوی ۰/۵ است. -م

یک) «بایاس» پیدا می کند؛<sup>۱</sup> این بایاس حتی اگر کوچک باشد می تواند برای کاهش Work Factor (جستجوی فضای کلید) مورد سوء استفاده واقع شود. برای شرح کامل این روش به مقاله Matsui مراجعه نمایید. سومین روش برای تحلیل رمز، استفاده از میزان توان الکتریکی مصرفی پردازنده برای یافتن کلید سری است. بطور معمول کامپیوترها از ولتاژ ۳ ولت برای نمایش بیت یک و از ولتاژ صفر برای بیت صفر استفاده می کنند. بنابراین پردازش بیت ۱ انرژی الکتریکی بیشتری نسبت به بیت صفر مصرف می کند. اگر یک الگوریتم رمزنگاری شامل حلقه ای باشد که در آن بیت های کلید به ترتیب پردازش می شوند، رمز شکن سیگنال ساعت n گیکاهرتزی را به مقدار کمتری کاهش داده (مثلاً صد مگاهرتز) و یک گیره کوچک از نوع سوسماری را به پایه های تغذیه و زمین CPU متصل کرده و بر توان مصرفی پردازنده در حین اجرای هر دستورالعمل نظارت می کند. استنتاج کلید رمز از این اطلاعات بسیار ساده خواهد بود. برای خشتی کردن این نوع از رمز شکنی، می توان الگوریتم رمزنگاری را به زبان اسمبلی برنامه نویسی کرده و مطمئن شد که توان مصرفی پردازنده مستقل از بیت های کلید و همچنین «کلیدهای دور» (Round Keys) است.

چهارمین روش، «تحلیل زمانی» (Time Analysis) است. الگوریتم های رمزنگاری سرشار از دستورات if then else هستند که در حقیقت بیت هایی از کلیدهای هر دور را آزمایش می کنند. هرگاه دستورات پس از then و else از لحاظ زمان اجرا، متفاوت باشند، با پایین آوردن سرعت سیگنال ساعت پردازنده و اندازه گیری زمان اجرای هر مرحله، استنتاج کلیدهای هر دور ممکن خواهد بود. وقتی یکایک کلیدهای هر دور از اجرای الگوریتم بدست آمد می توان کلید اصلی را محاسبه کرد. تحلیل توان مصرفی و تحلیل های زمانی را می توان بطور همزمان بکار گرفت تا کار کشف کلید رمز ساده تر شود. اگرچه تحلیل توان و زمان ممکن است عجیب به نظر برسد ولی در واقع این دو روش قادرند هر گونه سیستم رمز را که در مقابل چنین حمله ای، مقاوم طراحی نشده باشد، بشکنند.

### ۳-۸ الگوریتم های کلید عمومی (Public Key)

همواره توزیع و مبادله کلید رمز (Key Distribution) یکی از مشکلات سیستم های رمزنگاری بوده است. فارغ از آن که یک سیستم رمزنگاری چقدر قدرتمند و محکم است، هرگاه یک اخلالگر بتواند کلید رمز را سرقت کند، کل سیستم بی ارزش خواهد شد. رمز شکنها همیشه از روشهایی که در آنها کلید رمزنگاری و رمزگشایی یکسان است (یا از طریق یکدیگر قابل محاسبه هستند) قلباً استقبال می کنند. در این روشها بالاخره باید کلیدها بین کاربران سیستم توزیع شود. در همین نقطه به نظر می رسد که یک اشکال ذاتی و درونی وجود دارد. از یک طرف این کلیدها باید در مقابل سرقت حفاظت شوند و از طرف دیگر باید بین کاربران توزیع شوند، بنابراین نمی توان از این کلیدها در گاوصندوق نگهداری کرد!

در سال ۱۹۷۶، دو پژوهشگر در دانشگاه استنفورد به نامهای دیفی و هلمن (۱۹۷۶) یک سیستم رمز کاملاً جدید را پیشنهاد کردند که در آن کلیدهای رمزنگاری و رمزگشایی متفاوت بودند و با در اختیار داشتن کلید رمزنگاری عملاً نمی شد کلید رمزگشایی را استنتاج کرد. در طرح پیشنهادی این دو نفر، الگوریتم رمزنگاری E (با کلید e) و الگوریتم رمزگشایی D (با کلید d)، باید سه نیاز را برآورده می کرد. این نیازها را می توان به سادگی به صورت زیر توصیف کرد:

$$D(E(P))=P \quad ۱.$$

۲. استنتاج d (کلید رمزگشایی) از روی e (کلید رمزنگاری) بی نهایت مشکل باشد.

۳. E از طریق مکانیزم «حمله با متن های انتخابی و شناخته شده» شکسته نشود.

۱. یعنی احتمال وقوع یکی از بیتها از ۰/۵ کمتر و دیگری از ۰/۵ بیشتر می شود. -م

اولین نیاز بیانگر آن است که هر گاه الگوریتم رمزگشایی  $D$  را بر روی یک متن رمز شده یعنی  $E(P)$  اعمال کنیم مجدداً اصل پیام  $P$  را بدست بیاوریم. بدون این ویژگی گیرنده مجاز نیز قادر به رمزگشایی متن رمزی نخواهد بود. نیاز دوم به قدر کافی گویاست و احتیاجی به توضیح اضافی ندارد. نیاز سوم به نحوی که بعداً خواهیم دید از آن جهت است که یک رمز شکن ممکن است الگوریتم را با استفاده از متنهای شناخته شده بیازماید و به روش سعی و خطا متن رمز شده را بشکند. با این سه شرط دلیلی وجود ندارد که کلید رمزنگاری را نتوان به صورت عمومی در اختیار همه قرار داد.

روش کار بدین نحوست که یک شخص مثلاً آلیس، وقتی تمایل دارد پیامهای محرمانه دریافت کند باید ابتدا دو الگوریتم منطبق با شرایط فوق ابداع کند. الگوریتم و کلید رمزنگاری آلیس به صورت عمومی و آشکار اعلام می شود. آلیس حتی می تواند کلید عمومی [برای رمزنگاری] را در صفحه اصلی از وبسایت خودش به همه اعلام کند. ما از نماد  $E_A$  به معنای الگوریتم رمزنگاری با پارامتر  $A$  یعنی کلید عمومی آلیس، استفاده می کنیم. همچنین از نماد  $D_A$  به معنای الگوریتم رمزگشایی با پارامتر  $A$  یعنی کلید خصوصی آلیس، استفاده می نماییم. باب نیز دقیقاً همین کار را می کند،  $E_B$  را به صورت عمومی آشکار می کند در حالی که  $D_B$  را به صورت سری نزد خود نگهداری می کند.

حال ببینیم مشکل برقراری یک کانال مطمئن بین آلیس و باب که هیچ ارتباط قبلی با هم نداشته اند چگونه حل می شود. فرض شده کلید رمزنگاری آلیس یعنی  $E_A$  و کلید رمزنگاری باب یعنی  $E_B$  در فایل های قابل خواندن [و به صورت آشکار] قرار دارد. آلیس اولین پیام خود یعنی  $P$  را می گیرد و  $E_B(P)$  را محاسبه کرده و نتیجه را برای باب می فرستد. باب با اعمال کلید سری خود یعنی  $D_B$  آنرا رمزگشایی می کند. (به عبارت دیگر  $D_B(E_B(P))$  را محاسبه می کند). هیچ شخص دیگری نمی تواند از پیام رمزنگاری شده بهره برداری کند چرا که سیستم رمزنگاری بسیار قدرتمند فرض شده و استنتاج  $D_B$  (کلید رمزگشایی) از کلید رمزنگاری  $E_B$  بسیار مشکل و غیر عملی است. برای ارسال پاسخ پیام یعنی  $R$ ، باب  $E_A(R)$  را ارسال می کند. حال آلیس و باب می توانند به صورت مطمئن با یکدیگر مبادله پیام نمایند، بدون آن که کلیدهای سری آنها را غیر از خودشان کسی بداند.

شاید اشاره به چند اصطلاح در خصوص این روش مفید باشد. رمزنگاری با کلید عمومی (Public Key Cryptography) ایجاب می کند که هر کاربر دو کلید داشته باشد: یک کلید عمومی (Public Key) که تمام دنیا برای ارسال پیام به کاربر، از آن استفاده می کنند و یک کلید خصوصی (Private Key) که کاربر برای رمزگشایی پیامها بدان احتیاج دارد. از این به بعد، به کلمات از اصطلاحات کلید عمومی و خصوصی استفاده خواهیم کرد تا از «کلید سری» (Secret Key) که در سیستمهای رمزنگاری با کلید متقارن کاربرد دارد تمیز داده شود.

### RSA ۱۳-۸

تنها کاری که باید انجام شود یافتن الگوریتمی است که سه نیاز اشاره شده را برآورده نماید. به دلیل محاسن بالقوه ای که رمزنگاری با کلید عمومی در بردارد، بسیاری از پژوهشگران در این زمینه بشدت فعالیت می کنند و تاکنون چندین الگوریتم مناسب تدوین و معرفی شده است. یکی از الگوریتمهای خوب، توسط گروهی در دانشگاه MIT (به سرپرستی ری وست، ۱۹۷۸) ابداع شد. این روش که به نام RSA مشهور است از حرف ابتدایی اسامی مخترعین آن، ری وست، شامیر و آدلמן (Rivest, Shmir, Adelman) گرفته شده است. روش RSA برای حدود ربع قرن در مقابل تلاشهای فراوان برای شکستن آن، دوام آورده و یک روش بسیار قدرتمند تلقی می شود. بسیاری از روشهای عملی امنیت، براساس RSA هستند. بزرگترین اشکال این روش آن است که برای رسیدن به بالاترین درجه امنیت، به کلید رمزی با حداقل ۱۰۲۴ بیت احتیاج است (برخلاف الگوریتمهای با کلید

متقارن ۱۲۸ بیتی) که این موضوع الگوریتم را بسیار کند می کند.

روش RSA بر یک سری از اصول اساسی در نظریه اعداد استوار است. ما چکیده این روش را در همین جا بیان می کنیم؛ برای تفصیل بیشتر از مقالات کمک بگیرید.

۱. دو عدد اول بسیار بزرگ  $p$  و  $q$  را انتخاب نمایید. (عموماً ۱۰۲۴ بیتی)

۲. حاصل  $n = p \times q$  و  $Z = (p-1)(q-1)$  را بدست بیاورید.

۳. عددی انتخاب کنید که نسبت به  $Z$  اول باشد و آنرا  $d$  بنامید.

۴.  $e$  را به گونه ای پیدا کنید که  $exd \bmod Z = 1$  برقرار باشد.

با این پارامترها که پیشاپیش محاسبه می شود، آماده شروع رمزنگاری هستیم: متن اصلی (که به صورت رشته ای از بیتها تلفی می شود) ابتدا به تعدادی بلوک تقسیم می گردد، به نحوی که هر بلوک  $P$  در بازه  $0 \leq P < n$  قرار بگیرد. این کار را با تقسیم متن به بلوکهای  $k$  بیتی که در آن  $k$  بزرگترین عدد صحیحی است که در رابطه  $2^k < n$  صدق می کند، انجام بدهید. ( $n = p \times q$ )

برای رمزنگاری پیام  $P$ ،  $C = P^e \bmod n$  را محاسبه نمایید. برای رمزگشایی نیز  $P = C^d \bmod n$  را حساب کنید. براحتی قابل اثبات است که توابع رمزگشایی و رمزنگاری، توابع معکوس یکدیگر هستند. برای رمزنگاری فقط به  $e$  و  $n$  احتیاج دارید. برای رمزگشایی به  $d$  و  $n$  نیازمندید. بنابراین کلید عمومی متشکل از  $(e$  و  $n$ ) است و کلید خصوصی  $(d$  و  $n$ ) خواهد بود.

امنیت این روش از آنجا ناشی شده که تجزیه اعداد بسیار بزرگ به عوامل اول بسیار دشوار است. اگر رمزشکن بتواند عدد  $n$  را به عوامل اول تجزیه کند، قادر خواهد بود  $p$  و  $q$  را پیدا کرده و از این راه  $Z$  را محاسبه نماید. با در اختیار داشتن  $Z$  و  $e$  می توان توسط الگوریتم اقلیدس  $d$  را پیدا کرد. خوشبختانه، ریاضی دانها از حدود سیصد سال قبل برای تجزیه اعداد بزرگ [به عوامل اول] تلاش کرده اند و شواهد حاکی از آن است که این کار بسیار مشکل می باشد.

براساس گزارش ری وست و گروه ابداع کننده RSA، تجزیه یک عدد ۵۰۰ رقمی به روش «تکرار و آزمون» حدود ۱۰<sup>۲۵</sup> سال طول می کشد. در هر حال آنها فرض را بر آن گذاشتند که بهترین الگوریتم و سریعترین کامپیوتر هر دستورالعمل را در یک میکروثانیه انجام بدهد. حتی اگر کامپیوترها در هر دهه به صورت نمایی سریعتر شوند باز هم تا زمانی که تجزیه یک عدد ۵۰۰ رقمی امکان پذیر شود، قرنهای باقی مانده است و در آن زمان هم می توان برای امن کردن RSA،  $p$  و  $q$  را بزرگتر انتخاب کرد!!

یک مثال بسیار ساده آموزشی از عملکرد الگوریتم RSA در شکل ۸-۱۷ نشان داده شده است. در این مثال  $p$  را ۳ و  $q$  را ۱۱ فرض کرده ایم که نتیجه می دهد:  $n = 33$  و  $Z = 20$ . یک مقدار مناسب برای  $d$ ، عدد ۷ است چرا که ۷ و ۲۰ هیچ عامل مشترکی ندارند. طبق این انتخاب باید عدد  $e$  را به نحوی پیدا کرد که در رابطه  $7 \times e \bmod 20 = 1$  صدق نماید، که عدد ۳ را نتیجه می دهد. کدهای رمز شده  $C$  برای یک پیام مثل  $P$  از رابطه  $C = P^3 \bmod 33$  بدست می آید. متن رمز شده را می توان طبق قاعده  $P = C^7 \bmod 33$  از رمز خارج نمود. در این شکل برای نمونه، مراحل رمزنگاری و رمزگشایی کلمه "SUZANNE" نشان داده شده است.

به دلیل آن که اعداد اول انتخابی در این مثال بسیار کوچک است، لذا  $P$  باید کمتر از ۳۳ باشد و بدین ترتیب هر بلوک از متن فقط می تواند شامل یک تک کاراکتر باشد. نتیجه رمز، یک جانشینی تک کاراکتری خواهد بود که هیچوجه مؤثر نیست. در عوض اگر  $p$  و  $q$  را اعدادی در حدود ۲۵۱۲ انتخاب کرده بودیم، عدد  $n$  را از مرتبه ۲<sup>۱۰۲۴</sup> می داشتیم و هر بلوک از متن می توانست تا ۱۰۲۴ بیت یعنی ۱۲۸ کاراکتر هشت بیتی باشد. (برخلاف بلوکهای هشت کاراکتری در DES یا بلوکهای ۱۶ کاراکتری در AES).

متن آشکار (P)			متن رمز (C)		پس از رمزگشایی	
Symbolic	Numeric	$P^3$	$P^3 \pmod{33}$	$C^7$	$C^7 \pmod{33}$	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E

محاسبات فرستنده

محاسبات گیرنده

شکل ۸-۱۷. مثالی از الگوریتم RSA.

باید بدین نکته اشاره کرد که در RSA نیز شبیه به الگوریتم متقارن در حالت ECB<sup>۱</sup>، هرگاه ورودیه‌های مشابه به الگوریتم اعمال شود، نتیجه خروجی یکسان خواهد بود. بدین ترتیب برای رمزنگاری داده‌ها گونه‌ای از «زنجیره‌سازی» مورد نیاز است، ولیکن در عمل اکثر سیستم‌های مبتنی بر رمزنگاری کلید عمومی، از RSA فقط برای «توزیع کلیدهای نشست» بهره می‌گیرند تا یکمک آن کلید جدیدی تولید و رد و بدل گردد و پس از آن از الگوریتم متقارنی مثل «DES سه گانه» یا AES استفاده شود.<sup>۲</sup> زیرا استفاده از RSA برای رمزنگاری حجم عظیم اطلاعات، بسیار کند عمل می‌کند و به همین دلیل عموماً از آن فقط برای توزیع کلید استفاده می‌شود.

### ۸-۳-۲ الگوریتم‌های کلید عمومی دیگر

اگرچه از RSA به صورت گسترده‌ای استفاده می‌شود ولی این بدان معنا نیست که تنها الگوریتم شناخته شده کلید عمومی است. در حقیقت اولین الگوریتم کلید عمومی، الگوریتم Knapsack «کوله‌پشتی» است. (مِرکل و هِلْمَن؛ ۱۹۷۸) ایده این روش مبتنی بر آن است که شخصی تعداد فراوانی شیء در اختیار دارد که هر یک از لحاظ وزنی با دیگری متفاوت است. صاحب آنها، پیام خود را با انتخاب محرمانه یک زیرمجموعه از این اشیاء و ریختن آنها در یک کوله‌پشتی، کدگذاری می‌کند. وزن کل اشیاء درون کوله‌پشتی و همچنین فهرست تمام اشیاء ممکن، عمومی و آشکار است در حالی که فهرست دقیق اشیاء درون کوله‌پشتی محرمانه و سری است. در ابتدا گمان می‌رفت که با افزودن مجموعه‌ای از محدودیتها، استخراج فهرست اشیاء درون کیسه با فرض در اختیار داشتن وزن کل، در عمل غیرممکن خواهد بود و می‌تواند پایه یک «الگوریتم کلید عمومی» را تشکیل دهد.

مخترع این الگوریتم «رالف مِرکل»، کاملاً اطمینان داشت که این الگوریتم قابل شکستن نیست و به همین دلیل اعلام کرد که به هر کس که بتواند آن را بشکند صد دلار جایزه می‌دهد. ادی شامیر (Adi Shamir) یا همان S در نام RSA سریعاً آن را شکست و جایزه را گرفت. مِرکل بی‌درنگ الگوریتم را قویتر کرد و برای کسی که بتواند نسخه جدید را بشکند هزار دلار پیشنهاد کرد. رونالد ری‌وست (Ronald Rivest) یا همان R در RSA سریعاً نسخه جدید را نیز شکست و صاحب این جایزه شد. مِرکل هرگز جرات نکرد برای نسخه جدیدتر الگوریتم خود، ده هزار دلار پیشنهاد کند و بدین نحو «A» (یعنی Leonard Adelman) سرش بی‌کلاه ماند! الگوریتم Knapsack امن تلقی نمی‌شود و هیچگاه در عمل مورد استفاده قرار نمی‌گیرد.

۱. Electronic Code Book

۲. به عبارت ساده‌تر، RSA فقط در هنگام شروع یک نشست و آنهم برای مبادله یک کلید مشترک و تصادفی کاربرد دارد و پس از این مرحله از الگوریتم رمزنگاری متقارن استفاده می‌شود. -سم

یکی دیگر از شماهای کلید عمومی، بر پیچیدگی محاسبه لگاریتم گسسته (Discrete Logarithm) بنا نهاده شده است. الگوریتمهایی که از این قاعده استفاده کرده اند توسط El Gamal (۱۹۸۵) و Schnorr (۱۹۹۱) ابداع شده اند.

چند شمای دیگر نیز وجود دارد؛ مثلاً یک دسته از روشها مبتنی بر منحنی های بیضوی هستند (Menezes and Vanstone, 1993) ولیکن دو دسته از مهمترین الگوریتمهای کلید عمومی، براساس پیچیدگی و دشواری تجزیه اعداد اول بسیار بزرگ یا محاسبه لگاریتم گسسته اعداد بزرگ بنا شده اند. این مسائل حقیقتاً لاینحل به نظر می رسند و ریاضی دانها سالها بر روی آنها کار کرده اند ولی هیچ راهی برای رخنه در آن نیافته اند.

## ۸-۴ امضاهای دیجیتالی

احراز هویت و تعیین اعتبار بسیاری از اسناد حقوقی، بازرگانی و نظائر آن، براساس وجود یا عدم وجود امضای مجاز و دستنویس در ذیل آنها، انجام می شود و طبعاً تصویر این اسناد ارزش قانونی ندارد. برای سیستمهای پیام رسان کامپیوتری که جانشین گردش کاغذ و اسناد خطی شده اند نیز باید روشی پیدا شود تا بتوان آنها را به گونه ای امضاء کرد که هرگز قابل جعل نباشد.

مسئله ابداع یک روش جایگزین به جای امضاهای دستنویس یکی از موضوعات دشوار به حساب می آید. در اصل به سیستمی نیاز است که براساس آن یک طرف بتواند پیامی امضاء شده را برای طرف دیگر بفرستد به گونه ای که شرایط زیر به درستی احراز شود:

۱. گیرنده بتواند هویت شخص فرستنده پیام را بررسی کند.
۲. فرستنده بعداً نتواند محتوای پیام ارسالی خود را انکار کند.
۳. گیرنده نیز نتواند پیامهای جعلی برای خود بسازد. [و ارسال آنها را به دیگران نسبت بدهد.]

نیاز اول در سیستمهای اقتصادی و مالی حیاتی است؛ وقتی یک مشتری بانک از طریق کامپیوتر، سفارش خرید یک تَن طلا می دهد (۱) کامپیوتر بانک باید توانایی آنرا داشته باشد تا از هویت واقعی کامپیوتر سفارش دهنده، یقین حاصل کرده و مطمئن شود که سفارش دهنده همانی است که ادعا می کند و باید هزینه سفارش از حساب او کسر شود. به عبارت دیگر بانک باید مشتری خود را احراز هویت کند، (و در سمت مقابل، مشتری نیز از هویت بانک مطمئن شود).

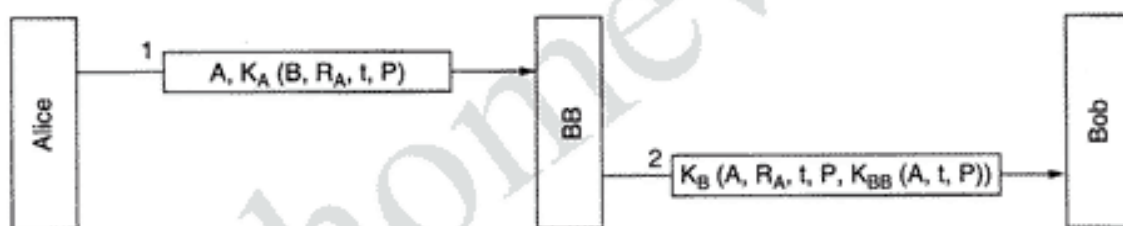
نیاز دوم بدان جهت حیاتی است که از بانک در مقابل کلاهبرداری محافظت نماید. فرض کنید بانک، یک تَن طلای سفارش داده شده را تهیه کرده ولی به ناگاه قیمت طلا بشدت سقوط می کند. مشتری بخت برگشته ممکن است ادعا کند که هرگز سفارش خرید یک تَن طلا نداده است. وقتی بانک پیام ارسالی را به دادگاه تسلیم می کند، ممکن است مشتری ارسال آن را تکذیب نماید. داشتن این ویژگی که هیچیک از طرفین یک قرارداد امضاء شده، نتوانند مفاد آن را تکذیب کنند اصطلاحاً "nonrepudiation" یا «غیر قابل انکار بودن» پیام نامیده می شود. الگوهای امضای دیجیتال که آنها را معرفی خواهیم کرد این قابلیت را فراهم می سازند.

سومین نیاز برای حفاظت از مشتری در مقابل کلاهبرداری است مثلاً در سناریوی قبل وقتی قیمت طلا پس از خرید بشدت افزایش می یابد شاید بانک بخواهد پیام ارسالی مشتری را دستکاری کرده و وزن طلای سفارشی را از یک تَن به یک شمش کاهش بدهد! در این سناریوی کلاهبرداری، بانک سود باقیمانده طلا را برای خود بر می دارد!!

## ۸-۱۸ امضاهای دیجیتالی با کلید متقارن

یکی از روشهای ساماندهی امضاهای دیجیتالی آنست که یک مرکز معتبر و مجاز گواهی امضاء داشته باشیم که همه را می شناسد و مورد اعتماد همه نیز هست؛ آن را اصطلاحاً BB (برادر بزرگتر یا Big Brother) می نامیم. هر کاربر برای خود یک کلید رمز سری (Secret Key) انتخاب کرده و شخصاً به اداره BB مراجعه و آن را ثبت می نماید. بدین ترتیب کاربری مثل آلیس، فقط خودش و BB کلید سری و توافق شده  $K_A$  را می دانند؛ بهمین روال، دیگر کاربران نیز کلید خودشان را در BB ثبت می نمایند.

وقتی آلیس بخواهد پیام امضاء شده خود یعنی P را برای کاربرداز بانک خود (یعنی باب) بفرستد،  $K_A(B, R_A, t, P)$  را تولید می کند که در آن B، مشخصه شناسایی باب (Bob ID)،  $R_A$  یک عدد تصادفی که توسط آلیس انتخاب شده، t مهر زمان برای اطمینان از جدید و تازه بودن آن، P اصل پیام و  $K_A(B, R_A, t, P)$  نتیجه رمزنگاری مجموعه این چهار آیتم توسط کلید سری آلیس یعنی  $K_A$  است. سپس او این داده های رمز شده را طبق شکل ۸-۱۸ برای BB می فرستد. BB متوجه می شود که پیام از آلیس است لذا آن را با کلید سری آلیس رمزگشایی می کند و به نحوی که در شکل نشان داده شده آنرا مجدداً رمز کرده و برای باب می فرستد. پیام ارسالی به باب شامل اصل پیام آلیس و یک پیام امضاء شده  $K_{BB}(A, t, P)$  است. حال باب می تواند درخواست آلیس را با اطمینان خاطر انجام بدهد.



شکل ۸-۱۸. امضاهای دیجیتالی به کمک مرکز گواهی امضاء (BB).

اگر بعداً آلیس ارسال پیام را انکار کرد چه اتفاقی می افتد؟ طبعاً هر کسی می تواند از دیگری ادعای خسارت کند! (حداقل در ایالات متحده این گونه است!) سرانجام وقتی مورد در دادگاه مطرح می شود و آلیس قویاً پیام ارسالی خود به باب را تکذیب می کند، قاضی از باب می پرسد که چگونه ادعا می کند پیام ارسالی آلیس حقیقتاً توسط خود آلیس ارسال شده و از شخص ثالثی مثل ترویدی نیست. باب ادعا می کند که BB هرگز پیامی را از آلیس قبول نخواهد کرد مگر آن که با کلید شخص آلیس یعنی  $K_A$  رمز شده باشد لذا امکان آن که شخص ثالثی بتواند پیامی جعلی را از طرف آلیس بفرستد و BB آن را کشف نکند وجود نخواهد داشت.

سپس باب مدرک  $K_{BB}(A, t, P)$  را به دادگاه تسلیم کرده و بیان می کند که این همان پیامی است که توسط BB امضاء شده است و ارسال پیام P توسط آلیس به باب را اثبات می کند. قاضی از BB (که مورد اعتماد همه است) می خواهد که این مدرک را رمزگشایی کند. وقتی BB صحت ادعای باب را تأیید کرد، دادگاه به نفع باب رأی خواهد داد و پرونده مختومه خواهد شد.

یکی از مسائلی که در پروتکل امضاء در شکل ۸-۱۸ به صورت بالقوه وجود دارد آن است که ترویدی ممکن است یک پیام ارسالی از آلیس را استراق سمع کرده و آن را بعداً از طرف آلیس به باب بفرستد. برای کاهش این مشکل برای هر پیام از «مهر زمان» (Time Stamp) استفاده می شود. به علاوه، باب می تواند با بررسی  $R_A$  اثبات کند که آیا چنین پیامی را قبلاً نیز دریافت کرده است یا خیر. اگر پیام تکراری بود آن را حذف خواهد کرد. دقت کنید که براساس مهر زمان باب می تواند پیامهای قدیمی را حذف کند. برای پیشگیری از تکرار آنی و بلادرنگ پیام، باب

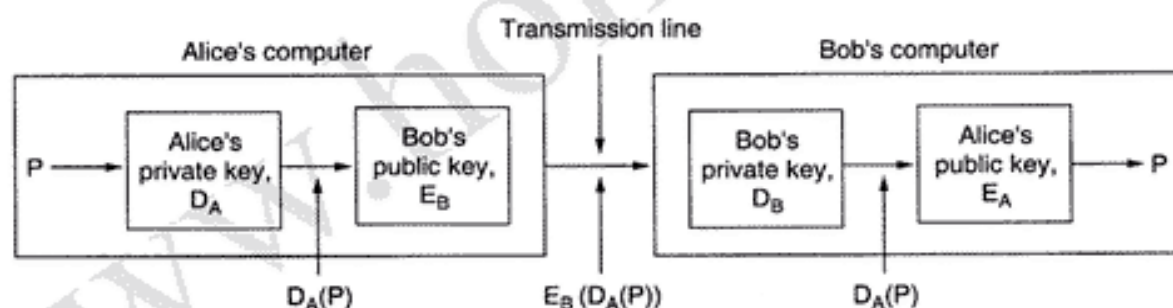
$R_A$  را بررسی می کند و اگر پیامی با  $R_A$  تکراری دریافت گردد حذف می شود. اگر باب از این دو مورد مطمئن شد می تواند فرض کند که این تقاضا معتبر و جدید است.

## ۲-۴-۸ امضاهای با کلید عمومی

مشکل ساختاری در یکارگیری رمزنگاری با کلید متقارن برای امضاهای دیجیتالی، آن است که همه باید به BB (مرکز گواهی امضاء) اعتماد کنند. در ضمن BB قادر است تمام پیامهای امضاء شده را بخواند. منطقی ترین کاندیداهای راه اندازی مرکز گواهی امضاء، دولت، بانکها، سازمانهای حساسی و کانون وکلاء هستند. متأسفانه هیچیک از این سازمانها مورد اعتماد و وفاق عموم شهروندان نیستند. بنابراین، امکان امضای مستندات به گونه ای که به هیچ مرکز گواهی امضاء نیاز نباشد، مورد بسیار جالبی است.

خوشبختانه، رمزنگاری با کلید عمومی می تواند در این زمینه نقش بسیار مؤثر و مثبتی ایفاء کند. فرض را بر آن می گذاریم که الگوریتمهای رمزنگاری و رمزگشایی دارای این خصوصیت است که  $E(D(P))=P$  و همچنین  $D(E(P))=P$  (RSA) دارای این ویژگی هست و چنین فرضی دور از واقعیت نخواهد بود. با فرض وجود این ویژگی، آلیس می تواند متن رمز و امضا شده  $P$  را به صورت  $E_B(D_A(P))$  برای باب بفراستد.<sup>۲</sup> دقت داشته باشید که آلیس فقط و فقط خودش کلید خصوصی خود یعنی  $D_A$  را می داند؛ همچنین کلید عمومی باب یعنی  $E_B$  را در اختیار دارد بنابراین ایجاد پیام فوق برای آلیس ممکن خواهد بود.

وقتی باب پیام را دریافت می کند، ابتدا آن را با کلید خصوصی خود رمزگشایی کرده و  $D_A(P)$  را مطابق با شکل ۸-۱۹ بدست می آورد. او این متن را در جای امنی ذخیره می کند [برای استفاده احتمالی در دادگاه] و سپس کلید عمومی آلیس یعنی  $E_A$  را بر روی آن اعمال کرده و متن اصلی را بدست می آورد.



شکل ۸-۱۹. امضاهای دیجیتالی با استفاده از رمزنگاری کلید عمومی.

برای آن که ببینیم این ساختار چگونه نیازهای امضای دیجیتالی را برآورده می کند، فرض کنید بعداً آلیس، ارسال پیام  $P$  به باب را انکار کند. وقتی این مورد در دادگاه مطرح می شود، باب هر دو پیام  $P$  و  $D_A(P)$  را به دادگاه تسلیم می کند. قاضی می تواند به سادگی و با اعمال کلید عمومی آلیس یعنی  $E_A$  بررسی کند که آیا باب، پیام اصلی و معتبر را دارد یا خیر! از آنجایی که باب، کلید خصوصی آلیس را در اختیار ندارد لذا نمی تواند  $D_A(P)$  را به صورت جعلی تولید کند و قطعاً آن را به همین صورت دریافت کرده و بنابراین کسی جز آلیس آن را نفرستاده است. آلیس در طی دوران حبس خود به جرم سوگند دروغ و کلاهبرداری، به قدر کافی وقت دارد که به طرح یک الگوریتم جدید رمزنگاری با کلید عمومی بپردازد!!

۱. یعنی فرض شده متن رمزنگاری شده با کلید عمومی، یکمک کلید خصوصی قابل رمزگشایی است و بالعکس متن رمزنگاری شده با کلید خصوصی، یکمک کلید عمومی قابل رمزگشایی است. -م
۲. یعنی ابتدا پیام را با کلید خصوصی خودش رمز کند و نتیجه را مجدداً با کلید عمومی باب رمز و ارسال نماید. -م

اگرچه بکارگیری رمزنگاری با کلید عمومی در طراحی امضاهای دیجیتال، الگوی بسیار جالبی است ولی از مشکلاتی رنج می برد که ناشی از خود الگوریتم نیست بلکه مربوط به محیطی است که در آن عمل می کند. فقط زمانی باب می تواند ثابت کند که پیام ارسالی متعلق به آلیس است که  $D_A$  (کلید خصوصی آلیس) محرمانه و سری باقی بماند. اگر آلیس کلید خصوصی خود را لو بدهد، هیچ دلیل محکمه پسندی باقی نخواهد ماند زیرا هر کسی حتی خود باب می توانسته چنین پیامی را ارسال نماید.

به عنوان مثال، مشکل زمانی بروز می کند که باب دلال سهام آلیس باشد. آلیس به باب می گوید که برایش سهام یا اوراق بهادار خاصی را بخرد. بلافاصله پس از این سفارش قیمتها بشدت سقوط می کند. آلیس برای آن که بتواند پیام سفارش خود به باب را انکار کند، سریعاً با پلیس تماس گرفته و ادعا می کند که خانه اش مورد سرقت واقع شده و کامپیوتر محتوی کلید رمز دزدیده شده است. بسته به قوانین حاکم بر کشور یا ایالت آلیس، او ممکن است از لحاظ قانونی از خود سلب مسئولیت کند یا برعکس مسئول خسارت خود باشد، بالاخص اگر او ادعا کند موضوع سرقت خانه اش را به دلیل آن که سر کار بوده چندین ساعت بعد متوجه شده است!

مشکل دیگر در این ساختار آن است که اگر آلیس تصمیم بگیرد کلید رمز خود را عوض کند چه اقدامی باید انجام بدهد؟ این کار کاملاً قانونی است و حتی شاید انجام آن به صورت دوره ای ایده بسیار خوبی هم باشد. اگر در دادگاهی که بعداً تشکیل می شود، به نحوی که در بالا اشاره کردیم قاضی کلید عمومی فعلی آلیس یعنی  $E_A$  جدید را بر روی  $D_A(P)$  اعمال نماید، متوجه می شود که  $P$  بدست نمی آید. باب در چنین شرایطی حسابی گیج و درمانده خواهد شد.

در اصل، از هر الگوریتم رمزنگاری با کلید عمومی می توان برای امضاهای دیجیتال بهره گرفت ولیکن استاندارد غیررسمی صنعت، رمزنگاری RSA است و بسیاری از محصولات امنیتی از آن بهره گرفته اند. با این وجود در سال ۱۹۹۱، NIST استاندارد جدیدی برای امضاهای دیجیتال به نام DSS<sup>۱</sup> پیشنهاد کرد که مبتنی بر الگوریتم رمزنگاری کلید عمومی El Gamal بود. الگوریتم El Gamal امنیت ذاتی خود را از دشوار بودن محاسبه لگاریتم گسسته (به جای تجزیه اعداد بزرگ به عوامل اول) کسب کرده است. طبق معمول وقتی دولت سعی در تحمیل یک استاندارد رمزنگاری می کند، غوغایی پیرامون آن به پا می شود. از DSS در موارد زیر انتقاد شده است:

۱. بسیار محرمانه و سری است (NSA در این پروتکل از الگوریتم El Gamal استفاده کرده است).
  ۲. بسیار کند عمل می کند. (برای بررسی امضاء ده تا چهل برابر از RSA کندتر عمل می نماید).
  ۳. خیلی جدید است. (الگوریتم El Gamal هنوز به قدر کافی تحلیل و بررسی نشده است).
  ۴. خیلی ناامن به نظر می رسد. (از کلید ۵۱۲ بیتی با طول ثابت استفاده شده است).
- در بازبینی های بعدی، چهارمین مورد اشکال با افزایش طول کلید به ۱۰۲۴ بیت، برطرف شد ولیکن همچنان دو اشکال اول باقی است.

### ۳-۸ خلاصه پیامها (Message Digests)

یک انتقاد که به روش امضاء دیجیتال وارد است، آنست که اغلب دو عمل متفاوت و مجزا را درهم آمیخته و باهم انجام می دهند: «احراز هویت» و «سری ماندن پیام». بسیاری از اوقات احراز هویت لازم است در حالی که به سری ماندن محتوای پیام نیازی نیست. سیستمهایی که فقط سرویس «احراز هویت» ارائه می کنند و در مورد مبادله سری

پیام کاری انجام نمی دهند (گذشته از سرعت و کارایی بیشتر) راحتتر مجوز صدور می گیرند.<sup>۱</sup> در زیر الگویی را برای «احراز هویت» بررسی می کنیم که نیازی به رمزنگاری کل پیام ندارد.

این ساختار مبتنی بر تابع Hash یک مرحله ای است (One-Way Hash Function) که یک قطعه طولانی از متن اصلی را گرفته و یک رشته بیتی با طول ثابت از آن محاسبه و استخراج می کند. حاصل این «تابع درهم سازی»، اغلب به نام «خلاصه پیام»<sup>۲</sup> یا MD<sup>۳</sup> شناخته می شود و دارای چهار ویژگی زیر است:

۱. با داشتن P مفروض، محاسبه MD(P) بسیار ساده است.
۲. با داشتن MD(P)، عملاً پیدا کردن P غیرممکن است.
۳. با داشتن P مفروض نمی توان یک P' پیدا کرد به نحوی که  $MD(P) = MD(P')$
۴. تغییر در ورودی حتی به اندازه یک بیت، خروجی کاملاً متفاوتی ایجاد خواهد کرد.

برای برآورده کردن ویژگی سوم، طول رشته Hash\*، باید حداقل ۱۲۸ بیت یا ترجیحاً بیشتر باشد. برای برآورده کردن نیاز چهارم، رشته Hash باید بر خلاف الگوریتمهای رمزنگاری با کلید متقارن [که بر روی بلوکهای کوچک عمل می کنند] براساس تمام بیتهای درهم شده پیام محاسبه شود.

محاسبه «خلاصه پیام» براساس قطعه ای از یک متن، بسیار سریعتر از رمزنگاری آن توسط الگوریتمهای کلید عمومی است و بدین ترتیب محاسبه MD می تواند برای افزایش سرعت الگوریتمهای امضای دیجیتال مؤثر واقع شود. برای آن که ببینیم این ساختار چگونه کار می کند مجدداً به پروتکل امضاء در شکل ۸-۱۸ دقت کنید. به جای امضای P با ارسال  $K_{BB}(A, t, P)$ ، مرکز گواهی امضاء (یعنی BB)، تابع MD را بر روی P اعمال کرده و یک رشته نسبتاً کوتاه MD(P) را بدست می آورد و با جاسازی آن در  $K_{BB}(A, t, MD(P))$  [به جای  $K_{BB}(A, t, P)$  که طولانی است] آنرا برای باب ارسال می کند.

هرگاه یک دعوای حقوقی بوجود بیاید، باب می تواند P و  $K_{BB}(A, t, MD(P))$  را به دادگاه عرضه کند. پس از آن که مرکز گواهی آنرا برای قاضی رمزگشایی کرد، باب MD(P) را به عنوان مدرک در اختیار خواهد داشت که می تواند صحت پیام P را تأیید کند. همچنین از آنجایی که عملاً باب قادر نخواهد بود که پیامی جعلی پیدا کند که رشته Hash آن [حاصل MD]، معادل با رشته Hash پیام واقعی باشد لذا باب نخواهد توانست دادگاه را در خصوص صحت پیام جعلی خود متقاعد کند. با استفاده از روش «خلاصه پیام» هم در زمان رمزنگاری کل پیام و هم در هزینه انتقال پیام بر روی شبکه صرفه جویی خواهد شد.

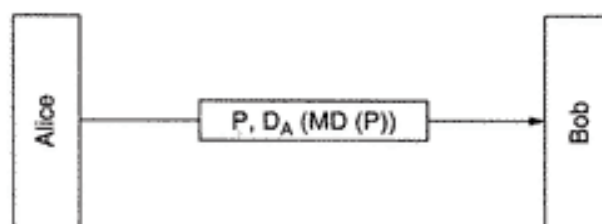
پایه سازی روش «خلاصه پیام» به نحوی که در شکل ۸-۲۰ نشان داده شده، براساس سیستمهای کلید عمومی نیز عملیست. در اینجا آلیس ابتدا خلاصه پیام خود را محاسبه می نماید، سپس آن را به جای اصل پیام امضاء کرده و نهایتاً اصل پیام [رمز نشده] را به همراه MD رمز شده برای باب می فرستد.<sup>۴</sup> اگر شخص ثالثی مثل ترویدی P را دستکاری کند، باب وقتی MD(P) را محاسبه می کند متوجه جعلی بودن پیام خواهد شد.

۱. به خاطر داشته باشید که در بسیاری از کشورها صدور نرم افزارهای رمزنگاری یا سیستمهای مبتنی بر مبادله سوزی پیام کلاً ممنوع است و مجازاتهای زندان (در فرانسه تا ۱۳ سال) دارد!! -م.

۲. اگر بخواهیم در دو جمله «خلاصه پیام» یا همان Message Digest را بیان کنیم عبارتست از: «یک رشته رمزی کوتاه که از درون یک پیام استخراج و پس از رمزنگاری به همراه پیام رمز نشده ارسال می شود. حتی اگر یک بیت از پیام اصلی دستکاری شود رشته خلاصه پیام جدید با این رشته یکسان نخواهد بود.» \* رشته Hash، در حقیقت همان رشته ای است که از درون پیام استخراج و به صورت رمز همراه آن ارسال می شود تا براساس آن بتوان سلامت پیام و هویت صاحب آنرا بررسی کرد. هر جا صحبت از Hash شد منظور همین رشته بیت است. -م

۳. Message Digest

۴. منظور از امضای «خلاصه پیام» یا MD، رمزنگاری آن توسط کلید خصوصی صاحب پیام می باشد. -م



شکل ۸-۲۰. امضاهای دیجیتالی با استفاده از «خلاصه پیام».

## MD5

توابع متنوعی برای تولید «خلاصه پیام» پیشنهاد شده است که رایج ترین آنها MD5 (ری وست، ۱۹۹۲) و SHA-1 (NIST، ۱۹۹۳) است. MD5 پنجمین روش پیاده سازی «خلاصه پیام» است که همگی توسط رونالد ری وست (سرپرست گروه ابداع کننده RSA) طراحی شده اند. این روش با درهم فشردن تمام بیتها، طبق رابطه ای بسیار پیچیده، MD را به نحوی محاسبه می کند که یکایک بیتهای خروجی از یکایک بیتهای ورودی [متن اصلی] تأثیر می پذیرند. بطور کاملاً مجمل، این روش ابتدا آنقدر بیتهای بی اهمیت به متن اصلی اضافه می کند که طول آخرین بلوک، فقط و فقط ۴۴۸ بیت باشد. سپس طول واقعی پیام در یک فیلد ۶۴ بیتی به پیام اضافه می شود تا نهایتاً تعداد صحیحی از بلوکهای ۵۱۲ بیتی بدست آید. [یعنی طول کل متن اصلی ضربی از ۵۱۲ شود]. سپس در تکمیل مراحل پیش پردازش، یک بافر ۱۲۸ بیتی با یک عدد ثابت، مقداردهی اولیه می گردد.

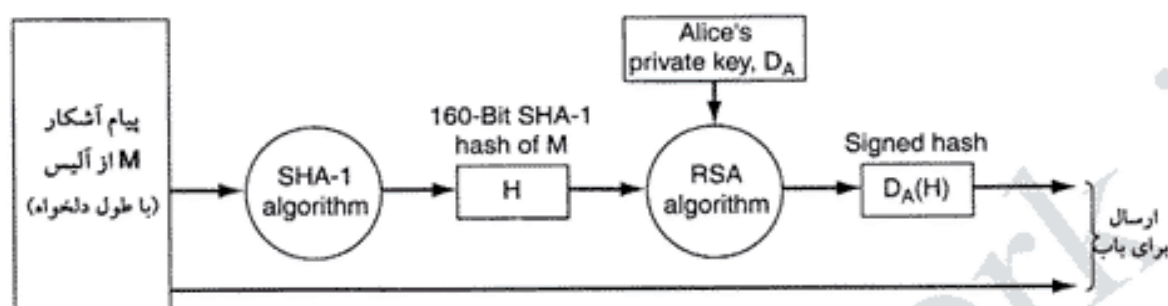
حال محاسبات آغاز می شود. در هر دور از محاسبه یک بلوک ۵۱۲ بیتی از متن ورودی استخراج و طبق یک رابطه خاص با بافر ۱۲۸ بیتی ادغام [مخلوط] می شود. برای اندازه گیری بهتر، یک جدول که با مقادیر تابع Sin مقداردهی شده نیز در آن تزیق می شود. استفاده از تابع شناخته شده ای مثل Sin از آن جهت نبوده که مقادیر آن تصادفی تر از مولد اعداد تصادفی است بلکه فقط بدین موضوع کمک کرده که جلوی هر گونه سوءظن در خصوص طراحی یک رخنه پنهان (Back door) درون الگوریتم گرفته شود. بخاطر داشته باشید که وقتی IBM از تشریح S-boxها در سیستم DES و ماهیت آنها طفره رفت، شایعات بسیار زیادی پیرامون آن در گرفت. ری وست (مخترع MD-5) می خواست از هر گونه شایعه ای جلوگیری کند. در ادامه، چهار دور پردازش بر روی هر بلوک ورودی انجام می شود. این فرآیند آنقدر تکرار می شود تا محاسبه تمام بلوکهای ورودی خاتمه یابد. محتوای بافر ۱۲۸ بیتی، Message Digest یا همان خلاصه «پیام» را تشکیل می دهد.

اکنون بیش از یک دهه است که MD5 معرفی شده و بسیاری از افراد سعی در حمله به آن داشته اند. اگرچه تعدادی نقطه ضعف در آن پیدا شده ولی برخی از مراحل درونی الگوریتم نگذاشته تا این الگوریتم درهم شکسته شود. با این وجود اگر این چند مرحله باقیمانده که حصارهای نهایی MD5 محسوب می گردند شکسته شود، MD5 فرو می باشد ولی علیرغم این موضوع، در زمان نوشتن این کتاب، MD5 هنوز هم به طور گسترده ای رایج است.

## SHA-1

یکی دیگر از توابع مهم «خلاصه پیام» تابع SHA-1 است که توسط NSA (آژانس امنیت ملی در آمریکا) ابداع شده و توسط NIST (اداره استاندارد ها و فناوریهای مدرن آمریکا) به شماره FIPS 180-1 به ثبت رسیده است. همانند MD5، SHA-1 نیز بلوکهای ورودی را در قالب بلوکهای ۵۱۲ بیتی پردازش می کند ولی برخلاف MD5 یک رشته

۱۶۰ بیتی (به عنوان خلاصه پیام یا Message Digest) تولید می کند. روش عمومی ارسال یک پیام غیر محرمانه ولی امضاء شده از آلیس به باب در شکل ۸-۲۱ نشان داده شده است. در این شکل، پیام آلیس به الگوریتم SHA-1 تحویل می شود تا یک رشته «درهم شده» ۱۶۰ بیتی<sup>۱</sup> بدست آید. سپس آلیس این رشته ۱۶۰ بیتی را با استفاده از کلید خصوصی RSA خود رمز (امضاء) می کند و نهایتاً پیام اصلی رمز نشده و رشته Hash امضاء شده (رمز شده) را برای باب می فرستد.



شکل ۸-۲۱. استفاده از SHA-1 و RSA برای امضای پیامهای غیر محرمانه.

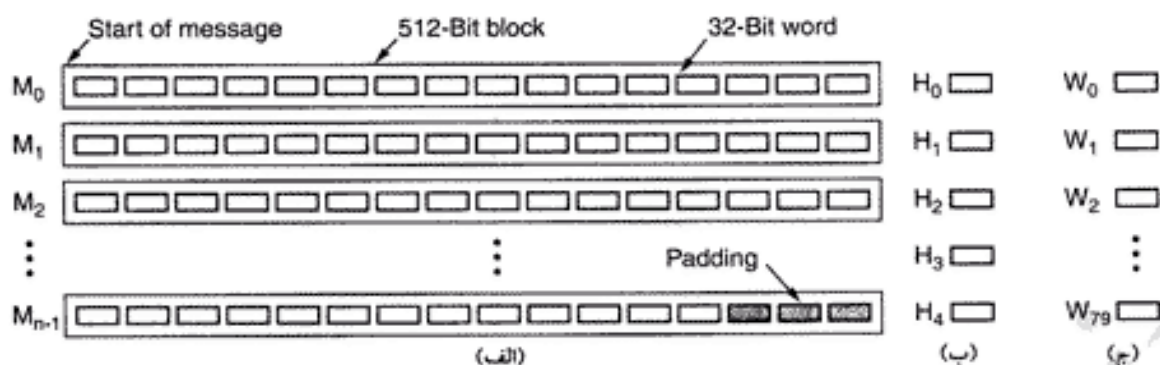
پس از دریافت، باب (بدون توجه به امضای پیام) شخصاً رشته Hash را برای پیام محاسبه می کند و از طرف دیگر با کلید عمومی آلیس، Hash ارسالی را بازگشایی می کند. اگر این دو رشته بیت با هم یکسان بودند پیام ارسالی معتبر و حقیقی است. از آنجایی که شخص ثالثی مثل ترویدی قادر نیست پیام ارسالی را در حین گذر از شبکه تغییر داده و برای آن Hash جدید تولید کند [چون کلید خصوصی آلیس را نمی داند] لذا باب به سادگی هر گونه تغییری را که توسط ترویدی در پیام ایجاد شده باشد، کشف خواهد کرد. برای پیامهایی که صحت آنها حیاتی است ولی محتوای آنها محرمانه نیست، روش شکل ۸-۲۱ کاربرد زیادی دارد. این ساختار ضمن هزینه محاسباتی بسیار پایین، تضمین می کند که هر گونه دستکاری در پیامها حین گذر از شبکه، با احتمال بسیار بالایی کشف شود.

حال اجمالاً ببینیم که SHA-1 چگونه کار می کند. SHA-1 کار را با اضافه کردن یک بیت ۱ در انتهای پیام و سپس تعدادی بیت صفر در ادامه آن آغاز می کند تا طول پیام ضربی از ۵۱۲ بیت شود. سپس یک عدد ۶۴ بیتی که طول حقیقی پیام (قبل از اضافه کردن بیتهای صفر و یک) را نشان می دهد با ۶۴ بیت انتهایی OR می شود. در شکل ۸-۲۲، یک پیام در قالب بلوکهای ۵۱۲ بیتی به همراه بیتهای اضافه شده به آن نشان داده شده است که محل قرار گرفتن بیتهای اضافی (و فیلد ۶۴ بیتی) در انتهای پیام در سمت راست (پایین ترین سطر) مشاهده می شود. در کامپیوترها این ساختار شبیه به ماشینهای Big-Endian<sup>۲</sup> مثل SPARC است ولی فارغ از نوع ماشین، SHA-1 داده های اضافی (Pad) را به انتهای پیام می چسباند. در خلال محاسبه، SHA پنج متغیر ۳۲ بیتی  $H_0$  تا  $H_4$  را در حافظه نگهداری می کند که رشته Hash [۱۶۰ بیتی] در آنجا جمع آوری و محاسبه می شود. در ادامه بلوکهای ۵۱۲ بیتی  $M_0$  تا  $M_{n-1}$  به ترتیب پردازش می شوند. برای بلوک جاری ابتدا ۱۶ کلمه ۳۲ بیتی [معادل ۵۱۲ بیت] به نحوی که در شکل ۸-۲۲-ج نشان داده شده در یک آرایه کمکی ۸۰ کلمه ای به نام W کپی می شود. ۶۴ کلمه باقیمانده از آرایه W طبق فرمول زیر محاسبه و پر می شوند:

$$W_i = S^1(W_{i-3} \text{ XOR } W_{i-8} \text{ XOR } W_{i-14} \text{ XOR } W_{i-16}) \quad (16 \leq i \leq 79)$$

۱. 160 Bits SHA-1 Hash

۲. ماشینهای Big Endian ماشینهایی هستند که برای ذخیره سازی کلمات ۲، ۴ یا ۸ بیتی در حافظه ابتدا بایت پر ارزش را و سپس بایتهای کم ارزش را ذخیره می کنند. ماشینهای سری x86 برعکس عمل می کنند.



شکل ۸-۲۲. (الف) به پیام آنقدر داده‌های زائد اضافه می‌شود تا طول آن ضربی از ۵۱۲ شود. (ب) متغیرهای خروجی. (ج) آرایه‌ای از کلمات ۴ بایتی.

در فرمول فوق،  $S^b(W)$  به معنای شیفت چرخشی به سمت چپ در کلمه ۳۲ بیتی  $W$  و به اندازه  $b$  بیت است. حال پنج متغیر جدید  $A$  تا  $E$  به ترتیب با مقادیر  $H_0$  تا  $H_4$  مقداردهی می‌شوند. محاسباتی را که در ادامه بر روی  $A$  تا  $E$  انجام می‌شود می‌توان به صورت شبه کد  $C$  نشان داد:

```
for (i = 0; i < 80; i++) {
 temp = S5(A) + fi(B, C, D) + E + Wi + Ki;
 E = D; D = C; C = S30(B); B = A; A = temp;
}
```

در کد بالا  $K_i$ ها ثابت‌هایی هستند که در استاندارد تعریف شده‌اند. تابع مخلوط‌سازی  $f_i$  نیز به ترتیب زیر تعریف شده است:

$$\begin{aligned} f_i(B, C, D) &= (B \text{ AND } C) \text{ OR } (\text{NOT } B \text{ AND } D) & (0 \leq i \leq 19) \\ f_i(B, C, D) &= B \text{ XOR } C \text{ XOR } D & (20 \leq i \leq 39) \\ f_i(B, C, D) &= (B \text{ AND } C) \text{ OR } (B \text{ AND } D) \text{ OR } (C \text{ AND } D) & (40 \leq i \leq 59) \\ f_i(B, C, D) &= B \text{ XOR } C \text{ XOR } D & (60 \leq i \leq 79) \end{aligned}$$

پس از آنکه حلقه هشتاد بار اجرا و تکمیل شد، متغیرهای  $A$  تا  $E$  به ترتیب با  $H_0$  تا  $H_4$  جمع می‌شوند. در اینجا اولین بلوک ۵۱۲ بیتی پردازش شده است و همین روال برای بلوک بعدی آغاز و تکرار می‌شود. آرایه  $W$  با بلوک جدید مقداردهی می‌گردد در حالی که مقادیر  $H_0$  تا  $H_4$ ، از مرحله قبلی [بدون تغییر] حفظ خواهد شد. هر گاه این بلوک نیز پردازش شد، بلوکهای بعدی نیز به همین ترتیب پردازش می‌شوند تا تمام بلوکهای ۵۱۲ بیتی در این ملغمه وارد شوند! وقتی آخرین بلوک پردازش شد، محتوای پنج کلمه ۳۲ بیتی  $H_0$  تا  $H_4$  در آرایه  $H$  به عنوان رشته ۱۶۰ بیتی Hash بدست می‌آید. برنامه کامل SHA-1 به زبان C در RFC 3174 ارائه شده است. نسخه جدید SHA-1 با رشته‌های ۲۵۶، ۳۸۴ و ۵۱۲ بیتی Hash تحت مطالعه و پیاده‌سازی است.

### ۸-۴ حملۀ روز تولد (The birthday Attack)

در دنیای رمزنگاری هیچ چیزی مثل آنچه در ظاهر به نظر می‌رسد، نیست! شاید کسی فکر کند برای شکستن Message Digest به طول  $m$  بیت، باید  $2^m$  حالت مختلف یکی‌یکی آزمایش شود. ولی در حقیقت با استفاده از مفهوم «حملۀ روز تولد» تعداد عملیات آزمایش  $2^{m/2}$  حالت است؛ روشی که توسط Yuval (۱۹۷۹) در مقاله‌ای به

نام "How to Swindle Rabin" منتشر شد. ایده این نوع حمله از تکنیکی منشاء می گیرد که اساتید ریاضی در کلاس درس احتمال، بکار می برند. سؤال این است که: «در یک کلاس چند نفر باید حضور داشته باشند تا احتمال آن که دو نفر در این جمع در یک روز متولد شده باشند بیش از  $\frac{1}{2}$  باشد؟» بسیاری از دانشجویان انتظار دارند جواب این مسئله بیش از ۱۰۰ باشد، ولیکن تئوری احتمال می گوید که این تعداد ۲۳ است. بدون یک تحلیل دقیق و به صورت ذهنی می توان بدین گونه حساب کرد که با ۲۳ نفر  $\frac{23 \times 22}{2} = 253$  زوج مختلف خواهیم داشت که هر یک از این زوجها با احتمال  $\frac{1}{365}$  در یک روز سال به دنیا آمده اند. بنابراین احتمال آن که یک زوج در یک روز به دنیا آمده باشد  $\frac{253}{365}$  و بیش از ۵۰٪ است که با این استدلال عدد ۲۳ چندان عجیب نیست.

بطور عام اگر یک نگاشت بین ورودی و خروجی، با  $n$  ورودی (مردم، پیامها و نظائر آن) و  $k$  حالت مختلف خروجی (تولد، Message Digest و نظائر آن) انجام شود،  $n(n-1)/2$  زوج ورودی خواهیم داشت. اگر  $n(n-1)/2 > k$  باشد احتمال آن که حداقل یک مورد تطابق حاصل شود، بسیار بالا خواهد بود. بنابراین اگر  $n > \sqrt{k}$  باشد احتمال آن که حداقل یک مورد تطابق پیدا شود [مثلاً دو مورد تولد در یک روز، یا تطابق پیام با Message Digest] بالاست. نتیجه فوق بدان معناست که شکستن<sup>۱</sup> یک خلاصه پیام (Message Digest) ۶۴ بیتی با تولید  $2^{32}$  پیام مختلف و مقایسه آنها امکان پذیر است. [حدود چهار میلیارد حالت مختلف]

بگذارید یک مثال عملی را بررسی کنیم. دانشکده علوم کامپیوتر در یک دانشگاه به یک عضو هیئت علمی نیاز دارد که برای این موقعیت دو نفر به نامهای «تام» و «دیک» نامزد شده اند. تام دو سال زودتر استخدام شده و به همین دلیل زودتر از دیک برای مصاحبه دعوت می شود و در این صورت «دیک» این موقعیت را از دست خواهد داد. تام می داند که «مرلین»، مدیر گروه دانشکده، نظر مساعدی به کار او دارد، به همین دلیل از او خواهش می کند یک توصیه نامه برای او خطاب به «دین» (Dean) مسئول گزینش تام، بنویسد. تمام نامه ها محرمانه خواهد ماند. «مرلین» از منشی خود «الین» می خواهد تا طی نامه ای خطاب به «دین» آنچه را مورد نظر اوست بنویسد تا پس از حاضر شدن، آنرا بررسی کرده و از طریق یک رشته ۶۴ بیتی Message Digest امضاء نماید. پس از آن «الین» می تواند نامه را از طریق پست الکترونیکی ارسال کند. از بخت بد تام، «الین» متمایل به انتخاب «دیک» است لذا ابتدا نامه زیر را با ۳۲ گزینه مختلف می نویسد (با مضمون خوب).

Dear Dean Smith,

This [letter | message] is to give my [honest | frank] opinion of Prof. Tom Wilson, who is [a candidate | up] for tenure [now | this year]. I have [known | worked with] Prof. Wilson for [about | almost] six years. He is an [outstanding | excellent] researcher of great [talent | ability] known [worldwide | internationally] for his [brilliant | creative] insights into [many | a wide variety of] [difficult | challenging] problems.

He is also a [highly | greatly] [respected | admired] [teacher | educator]. His students give his [classes | courses] [rave | spectacular] reviews. He is [our | the Department's] [most popular | best-loved] [teacher | instructor].

[In addition | Additionally] Prof. Wilson is a [gifted | effective] fund raiser. His [grants | contracts] have brought a [large | substantial] amount of money into [the | our] Department. [This money has | These funds have] [enabled | permitted] us to [pursue | carry out] many [special | important] programs, [such as | for example] your State 2000 program. Without these

۱. شکستن خلاصه پیام به معنای آنست که: رمز شکن یک رشته خلاصه پیام (Message Digest) داشته باشد و بتواند یک پیام معادل با آن پیدا کند. -م.

funds we would [be unable | not be able] to continue this program, which is so [important | essential] to both of us. I strongly urge you to grant him tenure.

پس از نگارش و تایپ نامه فوق، نامه دومی (با مضمون بد) به صورت زیر می نویسد:

Dear Dean Smith,

This [letter | message] is to give my [honest | frank] opinion of Prof. Tom Wilson, who is [a candidate | up] for tenure [now | this year]. I have [known | worked with] Tom for [about | almost] six years. He is a [poor | weak] researcher not well known in his [field | area]. His research [hardly ever | rarely] shows [insight in | understanding of] the [key | major] problem of [the | our] day.

Furthermore, he is not [respected | admired] [teacher | educator]. His students give his [classes | courses] [poor | bad] reviews. He is [our | the Department's] least popular [teacher | instructor], known [mostly | primarily] within [the | our] Department for his [tendency | propensity] to [ridicule | embarrass] students [foolish | imprudent] enough to ask questions in his classes.

[In addition | Additionally] Tom is a [poor | marginal] fund raiser. His [grants | contracts] have brought only a [meager | insignificant] amount of money into [the | our] Department. Unless new [money is | funds are] quickly located, we may have to cancel some essential programs, such as your State 2000 program. Unfortunately, under these [conditions | circumstances] I cannot in good [conscience | faith] recommend him to you for [tenure | a permanent position].

حال «الن» کامپیوتر خود را به نحوی برنامه ریزی می کند تا در خلال شب تمام ۲۳۲ حالت مختلف «خلاصه پیام» (Message Digest) را برای هر دو نامه محاسبه کند.<sup>۱</sup> زمانی او موفق است که دو پیام فوق دارای «خلاصه پیام» یکسان شوند. اگر نشد او چند گزینه دیگر به پیام دوم می افزاید و در پایان هفته مجدداً آنها را می آزماید. فرض کنید که بالاخره یک مورد تطابق پیدا می کند. نامه خوب را A و نامه بد را B بنامید.

«الن» نامه A را جهت تائید برای «مرلین» می فرستد و نامه B را به صورت محرمانه نزد خود نگاه می دارد و آنرا به هیچ کس نشان نمی دهد. «مرلین» محتوای نامه A را تائید و آنرا از طریق یک «خلاصه پیام» شصت و چهار بیتی امضاء می کند. «الن» مستقلاً به جای نامه اول، نامه دوم (B) را می فرستد، در حالی که امضای دو نامه یکی است.

پس از دریافت نامه و «خلاصه پیام» امضاء شده آن، «دین» (مسئول گزینش) الگوریتم MD را بر روی نامه دوم اعمال کرده و می بیند که با آنچه که «مرلین» امضاء کرده مطابقت دارد فلذا [براساس مضمون نامه] «تام» را بیرون می اندازد. «دین» متوجه نخواهد شد که «الن» به گونه ای برنامه ریزی کرده که دو نامه با MD مشابه ایجاد شود و نامه جعلی دوم را ارسال نموده است. (پایان اختیاری داستان: «الن» موضوع را به «دیک» می گوید. «دیک» را ترس فرا می گیرد و دوستی خود را با او بهم می زند. «الن» بشدت ناراحت می شود و نزد «مرلین» اعتراف می کند. «مرلین» موضوع را به «دین» اطلاع می دهد. «تام» منصب مورد نظر را بدست می آورد!!!!)

اعمال «حملة روز تولد» بر علیه MD5 بسیار مشکل است و حتی اگر در ثانیه یک میلیارد Message Digest متفاوت تولید شود محاسبه MD برای ۲۶۴ حالت مختلف برای دو نامه حدود ۵۰۰ سال طول می کشد و ضمناً

۱. ترکیبات مختلف کلماتی که در متن بصورت ایتالیک نمایش داده شده اند در مجموع حدود ۲۳۲ حالت خواهد بود. -م.

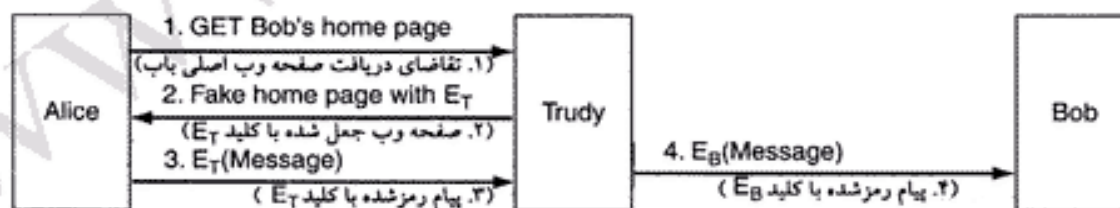
رسیدن به MD مشابه برای دو نامه تضمین شده هم نیست. البته با داشتن ۵۰۰۰ کامپیوتر که بطور موازی با هم کار کنند این زمان به پنج هفته کاهش خواهد یافت. SHA-1 در این مورد بهتر است چرا که رشته Message Digest طولانی تری ایجاد می کند.

## ۵-۸ مدیریت کلیدهای عمومی

«رمزنگاری با کلید عمومی» این امکان را فراهم آورده تا افرادی که از قبل بر روی یک کلید مشترک توافق نداشته اند، بتوانند با یکدیگر محاوره و مبادله اطلاعات داشته باشند. همچنین امضای پیامها را بدون نیاز به حضور یک شخص ثالث و مورد اعتماد، ممکن کرده است. نهایتاً، امضای پیامها با Message Digest، امکان بررسی صحت پیامهای دریافتی را میسر ساخته است.<sup>۱</sup>

با این حال، مشکلی وجود دارد که نگاهی اجمالی بدان خواهیم انداخت: اگر آلیس و باب شناختی از هم نداشته باشند چگونه کلید عمومی یکدیگر را بدست بیاورند تا مبادله اطلاعات را آغاز کنند؟ ساده ترین راه حل (یعنی قرار دادن کلید عمومی در وبسایت شخصی خود) به دلیل زیر کارآمد نخواهد بود: فرض کنید که آلیس می خواهد کلید عمومی باب را بر روی وبسایت او نگاه کند. چه کاری انجام می دهد؟ او URL وبسایت را تایپ می کند. مرورگر (Browser) او از DNS برای پیدا کردن آدرس سایت شخصی باب کمک گرفته و مطابق شکل ۸-۲۳ برای دریافت این صفحه وب، فرمان GET [از فرامین HTTP] را صادر می کند. متأسفانه ترویدی در حال استراق سمع این تقاضا است و در پاسخ به آن سریعاً یک صفحه وب جعلی بر می گرداند که احتمالاً کپی صفحه وب باب است با این تفاوت که کلید عمومی باب با کلید عمومی ترویدی عوض شده است. [این کلید جعلی را  $E_T$  فرض کنید.] وقتی آلیس اولین پیام خود را با  $E_T$  رمز می کند و به گمان خود آن را برای باب می فرستد ترویدی آن را گرفته و با کلید خصوصی خود رمزگشایی می کند و پس از بهره برداری، برای آن که این موضوع کشف نشود پیام را با کلید واقعی باب مجدداً رمز کرده و برای او می فرستد؛ بدون آن که باب آگاه باشد که پیام دریافتی او استراق سمع شده است. از آن بدتر، ترویدی قادر خواهد بود که پیام آلیس را قبل از رمزنگاری مجدد، تغییر نیز بدهد.

به وضوح برای پیشگیری از چنین حملاتی به مکانیزمهایی نیاز است تا مطمئن شویم که کلیدها عمومی به صورت مطمئن مبادله می شوند.



شکل ۸-۲۳. روشی که بر اساس آن ترویدی می تواند رمزنگاری کلید عمومی را با اشکال مواجه کند.

## ۱-۵-۸ گواهینامه ها (Certificates)

در اولین تلاش برای توزیع مطمئن کلیدهای عمومی، می توانیم به یک سازمان مرکزی برای توزیع کلیدها بپندیشیم که به صورت شبانه روزی فعال است و کلیدهای عمومی افراد را در اختیار قرار می دهد. یکی از مشکلات بی شمار این روش آنست که چندان قابل توسعه نیست و مرکز توزیع کلید سریعاً به یک گلوگاه در شبکه تبدیل می شود. [زیرا زیر بار تقاضاها خواهد ماند.] همچنین اگر زمانی این مرکز از کار بیفتد، امنیت اینترنت نیز به ناگاه مختل

۱. به خاطر داشته باشید که Message Digest نیز توسط کلید عمومی فرد امضا کننده پیام، از رمز خارج می شود. -م.

خواهد شد.

به دلایل فوق، عموم افراد راهکار متفاوتی اتخاذ کرده‌اند که در آن نیازی به وجود یک مرکز فعال و تمام وقت نیست. در حقیقت، اجباری برای روی خط (on-line) نگاه داشتن چنین مرکزی وجود ندارد. در عوض، تنها کاری که این مرکز باید انجام بدهد آن است که کلید عمومی متعلق به افراد، شرکتها و سازمانها را «گواهی» کند. امروزه سازمانی که کلیدهای عمومی افراد را گواهی می‌کند اصطلاحاً CA (Certification Authority) نامیده می‌شود.

به عنوان مثال فرض کنید باب می‌خواهد به آلیس و دیگران اجازه بدهد تا به صورت محرمانه با او مبادله داده داشته باشند. او می‌تواند با در دست داشتن پاسپورت یا گواهینامه رانندگی و همچنین کلید عمومی خود به مرکز CA مراجعه کرده و از آنها بخواهد که کلید عمومی او را گواهی کنند. CA برای او یک گواهینامه مشابه با شکل ۸-۲۴ صادر کرده و «رشته SHA-1 Hash»<sup>۱</sup> این گواهینامه را استخراج و آنرا با کلید خصوصی خودش امضاء (رمز) می‌کند. سپس باب هزینه مورد نظر CA را پرداخته و یک دیسک نرم (فلاپی) حاوی گواهینامه و رشته Hash امضاء شده را تحویل می‌گیرد.

I hereby certify that the public key  
19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A  
belongs to  
Robert John Smith  
12345 University Avenue  
Berkeley, CA 94702  
Birthday: July 4, 1958  
Email: bob@superdupernet.com

SHA-1 hash of the above certificate signed with the CA's private key

شکل ۸-۲۴. گونه‌ای از یک گواهینامه و رشته Hash امضاء شده آن.

کار اصلی یک گواهینامه آنست که نام صاحب کلید (نام شخصی، نام شرکت یا نظایر آن) را به همراه کلید عمومی او قید کرده باشد. باب ممکن است تصمیم بگیرد که گواهینامه جدید خودش را در وبسایت شخصی خود قرار داده و در صفحه اصلی یک لینک با این مضمون تعبیه کند: «برای دریافت گواهی دیجیتالی من اینجا کلیک کنید» با کلیک کردن، گواهینامه دیجیتالی و «بلوک امضای CA» (یعنی رشته Hash گواهینامه که توسط کلید خصوصی CA رمز شده است) برای متقاضی ارسال می‌شود.

حال مجدداً سناریوی شکل ۸-۲۳ را برای روش جدید به اجراء می‌گذاریم. وقتی ترودی تقاضای دریافت صفحه وبسایت باب را استراق سمع می‌کند، چه کاری می‌تواند انجام بدهد؟ ترودی می‌تواند بر روی صفحه وب جعلی (که بدروغ برای آلیس می‌فرستد) گواهینامه خودش را ارسال کند در حالی که وقتی آلیس این گواهینامه را مطالعه می‌کند متوجه می‌شود که در حال صحبت با باب نیست زیرا نام باب را در گواهینامه ارسالی مشاهده نمی‌کند. همچنین ترودی می‌تواند صفحه وب متعلق به باب را در حین ارسال دستکاری کرده و کلید عمومی او را با کلید عمومی خودش عوض کند ولیکن، وقتی آلیس الگوریتم SHA-1 را بر روی این کلید عمومی اجرا می‌کند، رشته Hash بدست آمده با رشته Hash که از رمزگشایی امضاء (با کلید عمومی و شناخته شده CA) حاصل می‌شود مطابقت ندارد. از آنجایی که ترودی کلید خصوصی CA را در اختیار ندارد لذا به هیچوجه قادر نخواهد

۱. یعنی خلاصه پیام استخراج شده بروش SHA-1

بود که بلوک امضای دستکاری شده در صفحه وب را رمز کند. بدین ترتیب آلیس می‌تواند مطمئن باشد که کلید عمومی باب را در اختیار دارد نه کلید ترویدی یا کس دیگری را. بنابراین همانگونه که قبلاً قول داده بودیم، در این ساختار لازم نیست که CA برای بررسی گواهینامه همیشه فعال و روی خط (Online) باشد و بدین ترتیب خطر بروز گلوگاه رفع خواهد شد.

هرچند عملکرد اصلی و استاندارد گواهینامه دیجیتال آنست که کلید عمومی شخص را با مشخصات صاحب اصلی آن قید کند، می‌توان از آن بدین نحو نیز استفاده کرد که کلید عمومی به‌مراه «ویژگی» (Attribute) صاحب آن قید شود. به عنوان مثال یک گواهینامه دیجیتال می‌تواند بیان کند که: «این کلید عمومی به شخصی با سن بالای ۱۸ سال تعلق دارد!» این گواهینامه می‌تواند [بدون فاش کردن نام و مشخصات شخص] ثابت کند که صاحب این کلید خردسال نیست و مجاز به استفاده از مفادی است که مناسب بچه‌ها نیستند. عموماً کسی که صاحب چنین گواهینامه‌ای است آن را برای سایتهای وب، اشخاص یا پروسه‌ها می‌فرستد تا در خصوص رده سنی او مطمئن شوند. این سایتهای اشخاص یا پروسه‌ها یک عدد تصادفی تولید و آن را با کلید عمومی موجود در گواهینامه طرف مقابل رمز کرده و برای او می‌فرستند. در صورتی که صاحب گواهینامه توانست آن را رمزگشایی کرده و پس بفرستد، ثابت می‌شود که او همان کسی است که ویژگیهایش در گواهینامه درج شده است. به جای این کار می‌توان از همین عدد تصادفی برای تولید یک کلید نشست جدید و محاوره مطمئن (رمزنگاری شده) بهره گرفت.

یک مثال دیگر که در آن به «گواهینامه حاوی ویژگی» (attribute) نیاز خواهد بود سیستمهای شیء‌گرای توزیع شده هستند. هر «شیء» عموماً دارای چندین «متود» است. مالک شیء می‌تواند برای هر مشتری یک گواهینامه دیجیتال فراهم کند که در آن یک رشته بیت درج شده و مشخص می‌کند آن مشتری، مجاز به فراخوانی چه متودهایی از آن «شیء» است و سپس این رشته بیت را به همراه کلید عمومی مشتری امضاء و ضمیمه می‌کند. بار دیگر، اگر کسی که گواهینامه دارد بتواند ثابت کند که کلید خصوصی خود را در اختیار دارد، اجازه خواهد داشت متودهایی که در این رشته بیت مشخص شده را، بکار بگیرد (بعبارت دیگر فراخوانی و اجراء کند). روش فوق این ویژگی را دارد که لازم نیست مشخصات واقعی مالک گواهینامه مشخص باشد؛ این خصوصیت برای حفظ حریم خصوصی افراد، بسیار مفید است.

## X.509 ۲-۵-۸

اگر هر کسی که می‌خواهد چیزی را امضا کند به CA مراجعه کند و نوع متفاوتی از گواهینامه بگیرد، چیزی نخواهد گذشت که مدیریت اشکال گوناگون و غیراستاندارد گواهینامه‌ها به یک مشکل عمده تبدیل خواهد شد. برای حل این مشکل، استاندارد برای صدور گواهینامه‌های دیجیتال ابداع و توسط ITU تأیید شده است. این استاندارد X.509 نامیده می‌شود و امروزه در اینترنت کاربرد گسترده‌ای دارد. پس از استانداردسازی اولیه در ۱۹۸۸، تاکنون سه نسخه متفاوت از X.509 ارائه شده است.

X.509 بشدت تحت تأثیر فضای حاکم بر OSI بوده است و برخی از بدترین ویژگیهای آن (مثل قواعد نامگذاری و روش کد کردن) را به عاریه گرفته است! خوشبختانه، IETF با X.509 همراهی کرد، کما اینکه تقریباً در تمام زمینه‌ها از آدرس‌دهی ماشینها و پروتکل‌های لایه انتقال گرفته تا قالب نامه‌های الکترونیکی، دخالت و همراهی کرده و عموماً در اغلب آنها از OSI<sup>۱</sup> چشمپوشی نموده و فقط سعی کرده کار را به درستی انجام بدهد. (فارغ از پیچیدگیهای دست و پاگیری که سازمان جهانی استاندارد همیشه تحمیل می‌کند.)

X.509 در اصل روشی برای تعریف و تبیین گواهینامه‌های دیجیتال است. فیلدهای اصلی یک گواهینامه

X.509 در شکل ۸-۲۵ فهرست شده‌اند. توضیحاتی که در این جدول وجود دارد می‌تواند کاربرد کلی هر فیلد را مشخص کند. برای اطلاعات بیشتر لطفاً از RFC 2459 راهنمایی بگیرید.

به عنوان مثال اگر باب در قسمت «وام» از «بانک پول» (Money Bank) مشغول به کار باشد، ممکن است آدرس X.509<sup>۱</sup> او برای درج در گواهینامه به صورت زیر باشد:

/C=US/O=MoneyBank/OU=Loan/CN=Bob/

که در آن C مشخصه کشور، O مشخصه سازمان (Organization)، OU مشخصه قسمت (بخش در سازمان) و CN مشخصه نام عمومی فرد است. دیگر مشخصات گواهینامه نیز به روش مشابه نامگذاری می‌شوند.

یکی از مشکلات ذاتی در نامگذاری X.509 آن است که اگر آلیس تلاش کند تا با کسی به آدرس bob@moneybank.com و دارای گواهینامه X.509، ارتباط برقرار کند، در گواهینامه‌ای که مشاهده می‌کند نام باب به وضوح دیده نمی‌شود. خوشبختانه در نسخه سوم از استاندارد X.509، اجازه داده شده که نامهای DNS (اسامی و آدرسهای نمادین در اینترنت) به جای اسامی X.509 در گواهینامه‌ها درج شود که بدین نحو مشکل فوق حل شده است.

گواهینامه‌های دیجیتالی پس از تدوین، به روش<sup>۲</sup> OSI ASN.1 کدگذاری می‌شود که می‌توانید فکر کنید تعریف و ساختاری شبیه به استراکچر در زبان C دارد، با این تفاوت که از نمادهای عجیب و غریب و بسیار طولانی استفاده شده است. اطلاعات بیشتر در مورد X.509 در (Ford and Baum, 2000) در دسترس می‌باشد.

کاربرد	نام فیلد
نسخه استاندارد X.509 را مشخص می‌کند.	Version
این شماره به همراه نام CA هویت (و اصالت) این گواهینامه را بصورت یکتا مشخص می‌کند.	Serial number
الگوریتم بکاررفته برای امضای گواهینامه را مشخص می‌کند.	Signature algorithm
نام X.509 برای CA (نام مرکز گواهی امضا)	Issuer
زمان شروع و ختم اعتبار گواهینامه	Validity period
موجودیتی که کلید عمومی او در این گواهینامه نایب می‌شود. (مثل شخص یا موسسه)	Subject name
کلید عمومی متعلق به صاحب گواهینامه و مشخصه الگوریتم مورد استفاده او	Public key
یک شماره شناسایی منحصر بفرد و یکتا که هویت صادرکننده گواهینامه را تعیین می‌کند. (اختیاری)	Issuer ID
یک شماره شناسایی منحصر بفرد و یکتا که هویت صاحب گواهینامه را تعیین می‌کند. (اختیاری)	Subject ID
بکمک این فیلد می‌تواند بهر تعداد مشخصات اضافی تعریف کرد.	Extensions
امضای کل گواهینامه (امضا شده با کلید خصوصی CA)	Signature

شکل ۸-۲۵. فیلدهای اصلی در گواهینامه X.509.

### ۳-۵-۸ زیرساخت کلید عمومی (Public Key Infrastructure)

به وضوح، داشتن یک مرکز مجاز صدور گواهی (CA) که تمام گواهینامه‌ها را در دنیا صادر کند، بزودی از کارایی ساقط خواهد شد؛ زیرا در زیر بار زیاد فرو می‌پاشد و در نوع خود یک «نقطه حساس به خرابی» محسوب می‌شود. یک راهکار دیگر آن است که چندین CA (مرکز گواهی) داشته باشیم که هر یک توسط یک سازمان خاص راه‌اندازی شده‌اند ولی همه از یک «کلید خصوصی» (Private Key) مشابه برای امضای گواهینامه‌ها استفاده

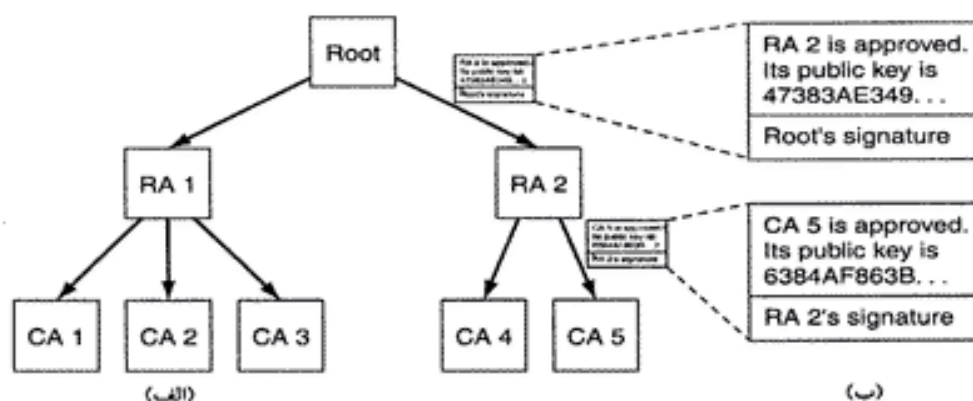
۱. X.500 استاندارد نامگذاری و X.509 استاندارد گواهینامه‌های دیجیتالی است.

۲. Abstract Syntax Notation 1.

کنند. این راهکار اگرچه مسئله بار و مشکل خرابی را رفع می کند ولیکن مشکل جدیدی را بوجود خواهد آورد: «مشکل نشت کردن و فاش شدن کلید». اگر دهها سرویس دهنده پراکنده در سطح دنیا وجود داشته باشد و همه از یک کلید خصوصی مشابه استفاده کنند احتمال سرقت کلید یا لو رفتن آن بشدت افزایش می یابد. از یک طرف توافق بر سر کلید خصوصی مشابه، زیربنای امنیت دنیای الکترونیک را متزلزل می کند و از طرف دیگر داشتن یک سازمان مرکزی صدور گواهینامه (CA) بسیار پرمخاطره است. به علاوه، چه سازمان یا نهادی باید متولی راه اندازی CA شود؟ به سختی بتوان به یک مرکز مجاز جهانی اندیشید که بتواند در سطح کل دنیا مورد وفاق و اعتماد عمومی قرار بگیرد. در بعضی از کشورها، مردم اصرار دارند که این مرکز به دست دولت اداره شود در حالی که در برخی دیگر کشورها، اصرار عمومی آن است که دولت در این خصوص دخالت نکند!

بدین دلایل روشهای متفاوتی برای گواهی کلیدهای عمومی افراد معرفی شده است. این روشها همگی تحت عنوان PKI (Public Key Infrastructure) مشهور هستند. در این بخش مبانی عمومی PKI را به اختصار بررسی می کنیم، هر چند در این خصوص پیشنهادات بسیار زیاد است و جزئیات آن هنوز در حال تکوین و تدوین می باشد.

PKI از چندین مولفه شامل «کاربران»، «CAها یا مراکز صدور گواهی»، «گواهینامه ها» و «دایرکتوریها» تشکیل شده است. آنچه که PKI باید انجام دهد آن است که تمام این مولفه ها را تحت لوای یک ساختار واحد جمع کند و برای مستندات و پروتکل های مختلفی که در این خصوص عرضه شده، استاندارد مدونی تعریف نماید. یک الگوی بسیار ساده از PKI، به نحوی که در شکل ۸-۲۶ نشان داده شده، ساختار سلسله مراتبی است. در این مثال سه سطح را نشان داده ایم که در عمل می تواند بیشتر یا کمتر باشد. بالاترین سطح CA یعنی «ریشه» (Root)، فقط مراکز صدور گواهینامه سطح ۲ را تائید و گواهی می کند؛ مراکز سطح ۲ اصطلاحاً RA<sup>۲</sup> یا «مراکز مجاز منطقه ای» نامیده می شوند زیرا یک ناحیه جغرافیایی مشخص مثل کشور یا قاره را پوشش می دهند. البته این تعاریف، استاندارد و دقیق نیستند و هیچ تعریف دقیقی برای مالکیت هر سطح از این ساختار درختی وجود ندارد. «مراکز مجاز منطقه ای» (RA) هویت مراکز صدور گواهینامه دیجیتالی (CAها) را که به منظور صدور گواهینامه های X.509 برای سازمانها و اشخاص حقیقی راه اندازی شده، گواهی و تائید می کند. وقتی «ریشه»، یک جدید را تائید و گواهی نمود برای آن مرکز، یک گواهینامه X.509 صادر و در اختیار آن RA قرار می دهد که در آن هویت و کلید عمومی آن مرکز، درج و امضاء شده است. به روش مشابه وقتی یک RA یک مرکز CA جدید را تائید می کند، این مرکز قادر است برای عموم افراد و سازمانها، گواهینامه های دیجیتالی شامل کلید عمومی آنها صادر نماید.



شکل ۸-۲۶. (الف) ساختار سلسله مراتبی PKI. (ب) یک زنجیره از گواهینامه ها.

PKI شبیه به این سناریو عمل می‌کند: فرض کنید آلیس برای مبادله اطلاعات با باب به کلید عمومی او نیاز دارد، لذا گواهینامه او را پیدا کرده و مشاهده می‌کند که این گواهینامه توسط CA5 [در ساختار شکل ۸-۲۶] صادر و امضاء شده است. آلیس هرگز در مورد CA5 چیزی نشنیده است. او می‌تواند به CA5 مراجعه کرده و از آن بخواهد که هویت قانونی خود را ثابت کند. در پاسخ، CA5 گواهینامه خودش را که توسط RA2 صادر و کلید عمومی CA5 در آن درج شده است، برای آلیس می‌فرستد. حال با کلید عمومی CA5، او می‌تواند گواهینامه باب را [که توسط CA5 صادر شده] بررسی کند و طبعاً تأیید می‌شود (در حالی که هنوز هویت RA2 مورد تردید است). در مرحله بعدی او به سراغ RA2 رفته و از او می‌خواهد که هویت قانونی خود را اثبات نماید. در پاسخ به این درخواست، یک گواهینامه دیجیتالی دیگر برمی‌گردد که توسط «ریشه» (عالیترین مرکز تأیید گواهینامه‌ها)، امضاء و در آن کلید عمومی RA2 درج شده است. حال آلیس می‌تواند به کلید عمومی باب و هویت او اعتماد کند.

سؤال آخر آنکه آلیس چگونه می‌تواند کلید عمومی «ریشه» را پیدا کند؟ فرض بر آن است که هر کسی کلید عمومی «ریشه» را می‌داند. به عنوان مثال، مرورگر او ممکن است به صورت درونی و در هنگام عرضه، این کلید را در اختیار داشته باشد.

باب جزو افراد ضمیمی است! و نمی‌خواهد که آلیس زحمت زیادی بکشد. او می‌داند که آلیس مجبور است گواهینامه CA5 و RA2 را بررسی کند، لذا برای کم کردن زحمت او، این دو گواهینامه مورد نیاز آلیس را بدست آورده و آنها را نیز به همراه گواهینامه خود برای او ارسال می‌نماید. حال آلیس با دانستن کلید «ریشه» شروع به بررسی گواهینامه سطح بالای RA2 می‌کند و در صورت صحت، با استفاده از کلید عمومی موجود در آن، گواهینامه بعدی یعنی CA5 را بررسی می‌نماید. بدین ترتیب آلیس احتیاجی به برقراری ارتباط با هیچ مرکزی برای بررسی صحت گواهینامه‌ها ندارد. به دلیل اینکه تمام این گواهینامه‌ها امضاء شده هستند براحتی می‌تواند هر گونه تقلب و دستکاری در گواهینامه‌ها را کشف کند. زنجیره گواهینامه‌های دیجیتالی که نهایتاً به «ریشه» ختم می‌شود گاهی «زنجیره اعتماد» (chain of trust) یا «مسیر گواهی» (Certification Path) نامیده می‌شود. از این تکنیک در عمل به صورت گسترده‌ای استفاده شده است.

البته هنوز یک مشکل باقی است و آن هم در خصوص آن است که چه کسی متولی «ریشه» خواهد بود؟ راهکار واقعی آن است که یک «ریشه واحد» نداشته باشیم بلکه تعداد بسیار زیادی «ریشه» (Root) وجود داشته باشد و هر یک برای خود تعدادی RA (مراکز منطقه‌ای) و تعدادی CA (مراکز صدور گواهینامه به افراد) داشته باشند. در حقیقت در مرورگرهای جدید، کلید عمومی پیش از صد «ریشه» گنجانیده شده و به صورت پیش فرض مشخص است؛ به این مجموعه از کلیدهای عمومی اصطلاحاً «لنگرهای اعتماد» (Trust Anchors) گفته می‌شود و بدین ترتیب از تمرکز بر روی یک مرکز واحد و جهانی پیشگیری خواهد شد.

اما مورد دیگری که بروز می‌کند آن است که چگونه عرضه کنندگان مرورگر (شرکتهایی که مرورگرها را طراحی می‌کنند) تصمیم بگیرند که کدامیک از مراکز عالی صدور گواهینامه دیجیتالی (Root) مورد اعتماد و کدام بی‌پشتوانه و مشکوک هستند؟ این مورد به کاربر و سطح آگاهی او بر می‌گردد؛ کاربر خود باید تصمیمی منطقی بگیرد و «لنگرهای اعتماد» (Trust Anchors) عرضه شده همراه با مرورگر را چشم بسته تأیید نکند. اکثر مرورگرها به کاربر اجازه می‌دهند تا کلیدهای ریشه را بررسی کند (عموماً در قالب گواهینامه‌هایی که توسط ریشه امضاء شده است) و آنهایی را که به نظرش مشکوک می‌رسد، حذف کند.

#### فهرستها (Directories)

مورد دیگری که در خصوص ساختار هر PKI وجود دارد آن است که گواهینامه‌ها (و زنجیره‌ای که آنها را به ریشه یا لنگرگاه اعتماد می‌رساند) در کجا ذخیره شوند؟ یک راهکار آن است که هر کاربر شخصاً گواهینامه خودش را

نگهداری کند. این راهکار مطمئن است (زیرا هیچ راهی برای دیگر کاربران وجود ندارد تا بتوانند گواهینامه امضاء شده او را بدون آن که کشف شود دستکاری کنند) ولیکن این روش چندان راحت نیست. راهکار دیگری که پیشنهاد شده آن است که از سرویس دهنده DNS به عنوان فهرست گواهینامه‌ها استفاده شود زیرا قبل از آن که آلیس بتواند با باب ارتباط برقرار کند، احتمالاً از طریق DNS، آدرس IP او را جستجو می‌کند؛ پس چرا DNS زنجیره کامل گواهینامه‌های باب را به همراه آدرس IP او نفرستد؟

برخی فکر می‌کنند این همان راهی است که باید ادامه پیدا کند ولی برخی دیگر ترجیح می‌دهند که یک سرویس دهنده اختصاصی برای مدیریت فهرستها وجود داشته باشد که صرفاً گواهینامه‌های X.509 را مدیریت کند. چنین سرویس دهنده‌ای می‌تواند سرویس جستجوی نام را براساس اسامی X.500 فراهم نماید. به عنوان نمونه چنین سرویس دهنده‌ای از دیدگاه تئوری قادر خواهد بود به پرسشی نظیر این مثال پاسخ بدهد: «فهرستی از افراد که نام آنها آلیس است و در بخش فروش شرکتهایی در آمریکا یا کانادا کار می‌کنند به من بده!» سیستم LDAP می‌تواند نامزد مناسبی برای نگهداری اینگونه اطلاعات باشد.

#### ابطال گواهینامه (Revocation)

دنای واقعی پر از گواهینامه‌های مختلف مثل گواهینامه رانندگی و پاسپورت است. گاهی اوقات این گواهینامه‌ها می‌تواند باطل شود؛ مثلاً رانندگی در حالت مستی یا برخی دیگر از تخلفات منجر به لغو گواهینامه رانندگی می‌گردد. همین مسائل در دنای دیجیتال نیز اتفاق می‌افتد: اعطاکنده گواهینامه به یک شخص یا سازمان ممکن است به دلیل سوءاستفاده از گواهینامه، تصمیم بگیرد که آنرا باطل کند. همچنین ممکن است به دلیل آن که کلید خصوصی صاحب گواهینامه فاش شده یا از آن بدتر کلید خصوصی یک مرکز گواهی امضاء (CA) لو رفته، گواهینامه‌ها باطل شوند. بنابراین یک PKI باید بتواند در صورت لزوم گواهینامه‌ها را لغو کند.

اولین گام در این راستا آن است که هر CA (مرکز صدور گواهینامه) بطور متناوب فهرستی به نام CRL<sup>۱</sup> (فهرست گواهینامه‌های باطل شده) صادر و در آن شماره سریال گواهینامه‌های باطل شده را مشخص نماید. از آنجایی که هر گواهینامه دارای «زمان اعتبار» مشخصی است لذا در CRL شماره سریال گواهینامه‌هایی درج می‌شود که هنوز منقضی نشده‌اند. زیرا پس از انقضای زمان اعتبار، گواهینامه به صورت خودکار اعتبار خود را از دست می‌دهد و تفاوتی بین گواهینامه‌های منقضی شده یا باطل شده وجود ندارد؛ در هر دو حالت گواهینامه‌ها بلااستفاده هستند.

متأسفانه، کاربرانی که قصد بررسی گواهینامه کسی را دارند باید ابتدا فهرست CRL را بدست بیاورند و بررسی کنند که آیا گواهینامه باطل شده است یا خیر. اگر گواهینامه در این فهرست بود نباید از آن استفاده کرد. ولیکن حتی اگر گواهینامه‌ای در این فهرست وجود نداشته باشد، باز هم امکان دارد بعد از تدوین این فهرست باطل شده باشد. بنابراین مطمئن‌ترین راه تعیین اعتبار، آنست که از CA سؤال شود. دفعه بعد هم که قرار است از همان گواهینامه استفاده شود، باز هم باید از CA سؤال کرد زیرا ممکن است چند ثانیه قبل باطل شده باشد.

یکی دیگر از پیچیدگیهای عمل «ابطال»، آنست که باید بتوان یک گواهینامه باطل شده را تجدید اعتبار کرد. به عنوان مثال گواهینامه‌ای که به دلیل عدم پرداخت هزینه مصوب، باطل شده، پس از پرداخت باید تجدید اعتبار شود. نیاز به ابطال یا تجدید اعتبار گواهینامه‌ها، یکی از بهترین ویژگیهای گواهینامه را پایمال می‌کند که همانا عدم مراجعه مکرر به CA بوده است.

فهرست CRL کجا باید ذخیره شود؟ بهترین مکان ذخیره CRL، همان جایی است که اصل گواهینامه‌ها ذخیره

می‌شود. یک راهکار آن است که CA به صورت فعال و متناوب فهرست CRL را منتشر کرده و گواهینامه‌های باطل شده از فهرستهای تمام کاربران حذف شود. اگر از «سرویس دایرکتوری» استفاده نشده باشد، می‌توان CRL را در یک نقطه مناسب از شبکه ذخیره کرد؛ از آنجایی که CRL یک سند امضاء شده است هر گونه دستکاری احتمالی در آن، به سادگی کشف خواهد شد و ذخیره آن در هر نقطه از شبکه خطر امنیتی ندارد.

اگر گواهینامه‌ها مهلت اعتبار بسیار طولانی داشته باشند، CRL نیز بسیار طولانی خواهد شد. به عنوان مثال اگر کارتهای اعتباری، برای پنج سال معتبر باشند تعداد کارتهای باطل شده، نسبت به زمانی که مهلت اعتبار این کارتها سه ماه است صد چندان خواهد بود. یک روش استاندارد برای تعامل با CRL آن است که این فهرست فقط در موارد خاص به صورت یکجا ارسال شود و در عوض فقط آخرین تغییرات در فهرست اعلام گردد. این کار پهنای باند لازم برای توزیع فهرستهای CRL را کاهش خواهد داد.

## ۶-۸ امنیت ارتباطات

تا اینجا مطالعه خود را در خصوص ابزارها و روشهای معمول امنیت به پایان رسانده‌ایم و بسیاری از پروتکلها و تکنیکها را بررسی کرده‌ایم. مابقی این فصل در خصوص آن است که این تکنیکها چگونه در محیط عمل وارد می‌شوند تا امنیت شبکه را تضمین نمایند؛ همچنین در انتهای فصل در خصوص جنبه‌های اجتماعی، امنیتی، نظریاتی را ارائه خواهیم کرد.

در چهار بخش آتی، نگاهی به امنیت ارتباطات (Communication Security) خواهیم انداخت که در خصوص «تحویل مطمئن و سری بیتها از مبدا به مقصد، بدون هیچ تغییر یا دستکاری و همچنین چگونگی جلوگیری از تزریق بیتهای ناخواسته به لینک ارتباطی» بحث می‌کند. این روشها اگرچه تنها موارد امنیتی در سطح شبکه نیستند ولی جزو مهمترین موارد محسوب می‌شوند و بالطبع نقطه شروع خوبی خواهند بود.

### ۱۶-۸ IPsec

IETF از سالها قبل بخوبی دریافته بود که «امنیت در اینترنت» در حال زوال است و برقراری امنیت نیز بسادگی میسر نبود زیرا بر سر نقطه‌ای که باید امنیت در آنجا متمرکز می‌شد مناقشه و جدل وجود داشت. بسیاری از خبرگان امنیت بر این اعتقادند که برای تضمین واقعی امنیت در شبکه، رمزنگاری و بررسی صحت پیامها باید «انتها به انتها» (End to End) انجام شود (به عبارت دیگر در سطح لایه کاربرد)؛ بدین سیاق، پروسه مبدا اقدام به رمزنگاری داده‌ها و تمهیدات حفاظتی کرده و سپس آنها را برای پروسه مقصد ارسال می‌کند؛ این پروسه نیز داده‌ها را رمزگشایی کرده و آنها را از لحاظ صحت بررسی می‌کند. هر گونه دستکاری در داده‌ها مابین این دو پروسه، حتی اگر در سطح سیستم عامل انجام شده باشد، قابل کشف است. اشکال این روش آن است که تمام برنامه‌های کاربردی باید به گونه‌ای تغییر داده شوند که خودشان امنیت مورد نظر خود را تأمین و تضمین نمایند. با این دیدگاه، رویکرد بهتر آن است که وظیفه رمزنگاری داده‌ها به لایه انتقال محول شود یا آن که لایه جدیدی بین لایه انتقال و لایه کاربرد این وظیفه را بر عهده بگیرد؛ در این صورت باز هم امنیت، «انتها به انتها» خواهد بود، بدون آنکه برنامه‌های کاربردی نیاز به تغییر داشته باشند.

دیدگاه مخالف این رویکرد، آنست که کاربران درک صحیحی از امنیت ندارند و قادر به استفاده صحیح از آن نیستند و در ضمن هیچکس نمی‌خواهد برنامه‌های موجود خود را تحت هیچ شرایطی تغییر بدهد، لذا این لایه شبکه است که باید احراز هویت و رمزنگاری بسته‌ها را (بدون آنکه کاربر را درگیر کند) بر عهده بگیرد. پس از سالها کشمکش بین این دو دیدگاه، نظریه پشتیبانی از امنیت در سطح لایه شبکه به یک پیروزی نسبی دست یافت و استانداردهای امنیت در لایه شبکه شکل گرفت. چکیده استدلال آن بود که رمزنگاری در سطح لایه شبکه مانع از

انجام صحیح و مطلوب عملیات کاربران آگاه و مسلط به امنیت نخواهد شد در حالی که به کاربران ناآگاه تا حدی کمک می کند.<sup>۱</sup>

نتیجه این منازعه طرحی بود که IPsec نامیده شد و در مستندات RFC 2401، 2402، 2406 تشریح گشت. از آنجا که تمام کاربران نمی خواهند که از رمزنگاری [بسته ها] استفاده کنند (زیرا از لحاظ زمان پردازش هزینه بالایی دارد)، لذا استفاده از آن اختیاری است. البته برای آن که طرح IPsec عمومیت خود را از دست ندهد و در همان بدو کار کثرت پروتکل بوجود نیاید تصمیم بر آن شد که در تمام حالات رمزنگاری انجام شود ولیکن در عوض، یک الگوریتم «پوچ» (Null Algorithm) (برای آنهایی که نمی خواهند بسته ها رمزنگاری شوند) تعریف گردید. این «الگوریتم پوچ» به دلیل سادگی، راحتی در پیاده سازی و سرعت بسیار بالا در RFC 2410 تشریح و از آن ستایش شده است!!! شاید سریعترین الگوریتم دنیا باشد!

طراحی کامل IPsec متشکل از یک چارچوب کاری (Framework) برای ارائه خدمات چندگانه، شامل تعدادی الگوریتم و مولفه است. دلیل ارائه چندین رده خدمات (Services) آن است که شاید همه نخواهند برای استفاده از تمام آنها هزینه پردازند فلذا این خدمات به صورت انتخابی در اختیار کاربران هستند. خدمات ویژه عبارتند از «ارسال محرمانه بسته ها» (Secrecy)، «تضمین صحت» (Integrity) و حفاظت در مقابل حملاتی که براساس آنها یک بخش از داده ها به صورت تکراری ارسال می شوند<sup>۲</sup> (که در آن اخلالگر سعی می کند یک سری از بسته های مجاز را بصورت تکراری ارسال نماید بدون آن که از محتوای آنها مطلع باشد). تمام خدمات فوق براساس رمزنگاری با کلید متقارن انجام می شود زیرا در سطح لایه شبکه کارایی و سرعت بسیار بالا، کاملاً حیاتی است. دلیل استفاده از چندین الگوریتم رمزنگاری آن بوده که شاید الگوریتمی که امروزه امن به حساب می آید در آینده شکسته شود. وقتی IPsec مستقل از الگوریتم خاص طراحی شده باشد، حتی در صورت شکسته شدن یک الگوریتم در آینده، باز هم قابل استفاده خواهد بود و به حیات خود ادامه می دهد. [یعنی اعتبار IPsec به اعتبار یک الگوریتم رمزنگاری خاص گره نخورده است].

دلیل مولفه های چندگانه ای که این پروتکل دارد آنست که بتوان فقط بر روی یک اتصال TCP متمرکز شد و از داده هایی که بین دو ماشین خاص در شبکه مبادله می شود مراقبت کرد یا آنکه از بین کل مسیربها صرفاً ترافیک بین دو مسیرب امن، رمزنگاری شود.

یکی از جنبه های نسبتاً عجیب IPsec آن است که اگرچه این پروتکل در لایه IP (لایه شبکه) قرار می گیرد ولیکن برخلاف IP، اتصال گرا (Connection Oriented) است. در واقع این مسئله چندان هم عجیب و دور از ذهن نیست زیرا برای ایجاد امنیت باید یک کلید رمز بین ماشینها توافق و ایجاد گردد که در اصل نوعی از «اتصال» محسوب می شود. (زیرا به هماهنگیهای قبلی بین ماشینها نیاز است).

البته هزینه برقراری چنین اتصالی بر روی حجم زیادی از بسته ها سرشکن می شود. یک «اتصال» در عرف IPsec اصطلاحاً SA (Security Association) نامیده می شود. یک SA، اتصالی «یکطرفه» بین دو نقطه پایانی در شبکه است که به آن یک «شناسه امنیت» (Security Identifier) منتسب شده است. اگر نیاز باشد که ترافیک در دو جهت به صورت امن مبادله شود، به دو SA احتیاج است. «شناسه های امنیت» درون بسته هایی که در شبکه و مبتنی بر یک «اتصال» سیر می کنند جاسازی شده و از آن برای جستجوی کلید متناظر و همچنین بدست آوردن اطلاعات مرتبط با بسته های امن ورودی استفاده می شود.

۱. به عبارت بهتر کاربرانی که این سطح از امنیت آنها را راضی نمی کند هیچ مانعی برای پیاده کردن استراتژی های خود در لایه های بالاتر نخواهند داشت و می توانند این سطح از امنیت را نادیده بگیرند؛ در حالی که برای کاربران معمولی بسیار مفید است. م.

۲. Replay Attack

با دیدگاه فنی، IPsec از دو بخش اصلی شکل گرفته است: در بخش اول دو سرآیند (Header) جدید تعریف شده که می‌تواند برای حمل «شناسه امنیت»، داده‌های لازم برای کنترل صحت اطلاعات و داده‌های مرتبط با امنیت استفاده شود. بخش دیگر یعنی ISAKMP<sup>۱</sup> با ایجاد و توزیع کلید رمز سر و کار دارد. ما به دو دلیل ISAKMP را بررسی نخواهیم کرد، زیرا: (۱) پشدد پیچیده است. (۲) پروتکل اصلی آن IKE<sup>۲</sup> اشکالات بنیانی دارد و باید عوض شود. (رجوع کنید به Perlman & Kaufman, 2000)

از IPsec می‌توان در دو حالت استفاده کرد؛ در «حالت انتقال» (Transport Mode) سرآیند IPsec دقیقاً پس از سرآیند بسته IP قرار می‌گیرد و فیلد پروتکل در بسته IP به گونه‌ای مقداری می‌شود که مشخص کند پس از سرآیند بسته IP سرآیند IPsec شروع می‌شود.<sup>۳</sup> سرآیند IPsec، شامل اطلاعات امنیتی نظیر شناسه SA، یک شماره ترتیب جدید و احتمالاً تنظیمات لازم برای بررسی صحت محتوای بسته (Payload) است.

در «حالت تونل» (Tunnel Mode)، کل بسته IP شامل سرآیند و محتوای آن در درون یک بسته جدید با سرآیند متفاوت جاسازی می‌شود. «حالت تونل» زمانی مفید است که انتهای تونل به جای ماشین مقصد به یک نقطه خاص [مثل یک مسیر یاب] ختم شود. در برخی از حالات، انتهای تونل یک ماشین است که نقش «دروازه امنیت» (Security Gateway) را ایفاء می‌کند (مثلاً دیوار آتش یک شرکت). در این حالت، دیوار آتش در حین عبور بسته‌ها، آنها را جاسازی (کپسوله) کرده و طرف دیگر بسته‌ها را باز می‌کند. وقتی تونل به یک ماشین امن در شبکه منتهی شود، ماشینهایی که در درون شبکه محلی آن شرکت واقفند نیازی به IPsec نخواهند داشت. در این حالت فقط دیوار آتش با IPsec سر و کار دارد.<sup>۴</sup> همچنین زمانی که باید مجموعه‌ای از اتصالات TCP به صورت یکجا جمع شده و به صورت جریانی واحد رمزنگاری شوند، «حالت تونل» مفید خواهد بود تا اختلالگر متوجه نشود چند بسته و برای چه کسی ارسال می‌شود. گاهی اوقات دانستن آن که چه مقدار ترافیک به کجا می‌رود، اطلاعات با ارزشی محسوب می‌شود. به عنوان مثال در حین بروز یک بحران نظامی، جریان اطلاعات بین کاخ سفید و پنتاگون سریعاً افت کرده و در عوض حجم ترافیک بین پنتاگون و یکی از مقرهای نظامی در کلرادوی آمریکا به همان اندازه افزایش می‌یابد. حال یک جاسوس اختلالگر می‌تواند از این داده‌ها، اطلاعات با ارزشی بدست بیاورد. مطالعه الگوی جریان بسته‌ها، حتی وقتی رمزنگاری شده هستند، اصطلاحاً «تحلیل ترافیک» (Traffic Analysis) نامیده می‌شود. «حالت تونل» می‌تواند تاحدودی برای خنثی کردن این مشکل کارساز باشد. اشکال «حالت تونل» آن است که باید یک سرآیند اضافی به هر بسته IP افزوده شود و بدین ترتیب اندازه بسته‌ها افزایش خواهد یافت. برعکس، در «حالت انتقال» اندازه بسته چندان تغییر نخواهد کرد.

اولین سرآیند جدید، AH<sup>۵</sup> است. این سرآیند، بررسی صحت داده و غیر تکراری بودن بسته‌ها را ممکن می‌سازد ولی داده‌ها سرّی نیستند (به عبارت دیگر رمزنگاری صورت نمی‌گیرد). کاربرد AH در «حالت انتقال»، در شکل ۸-۲۷ به تصویر کشیده شده است. در پروتکل IPv4 این سرآیند (AH)، بین سرآیند اصلی بسته IP و سرآیند بسته TCP قرار می‌گیرد. در پروتکل IPv6 نیز این سرآیند به عنوان «سرآیند اضافی» (Extension Header) تلقی می‌شود. [به مشخصات IPv6 مراجعه کنید.] ممکن است به نحوی که در شکل مشخص شده، به دلیل الزام در الگوریتم احراز هویت، لازم باشد که به داده‌ها مقداری داده (Pad) اضافه شود.

۱. Internet Security Association and Key Management Protocol

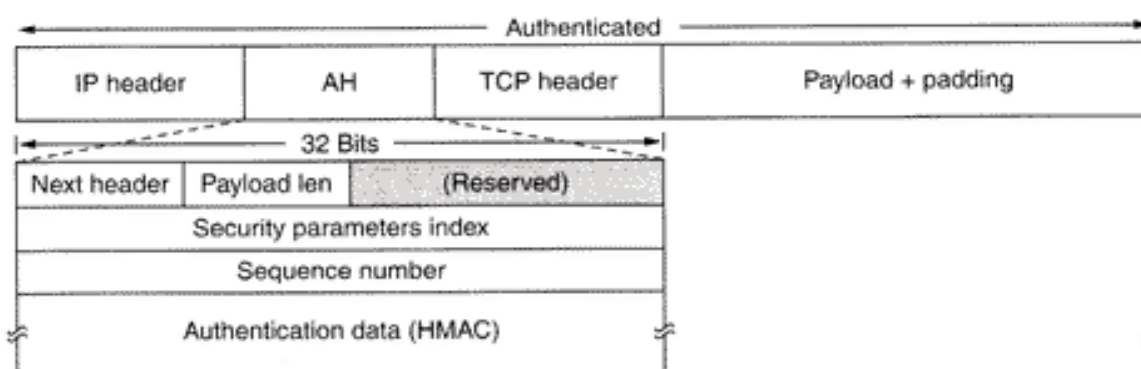
۲. Internet Key Exchange

۳. در فیلد پروتکل از بسته IP شماره پروتکل لایه بالاتر (پردازش‌کننده بسته) درج می‌شود. درج شماره IPsec در این فیلد نشان می‌دهد که بسته IP محتوی یک بسته IPsec است، نه بسته TCP، UDP یا هر پروتکل دیگر. رجوع کنید به فصل ۵-م.

۴. زیرا دیوار آتش به نیابت از همه آنها بسته‌ها را جاسازی و رمزنگاری می‌کند و سمت مقابل، آنها را بازگشایی می‌نماید.

۵. Authentication Header

بنابراین ماشینها، درگیر با IPsec نخواهند بود.



شکل ۸-۲۷. «سرآیند احراز هویت» (Authentication Header) که در «حالت انتقال» و برای IPv4 بکار می آید.

حال بیایید سرآیند AH را بررسی کنیم: فیلد Next Header بدان منظور به کار می رود که مقدار قبلی فیلد Protocol در بسته IP را (قبل از تغییر به مقدار ۵۱<sup>۱</sup>)، حفظ نماید. در اغلب موارد در این فیلد، عدد ۶ قرار می گیرد (به معنای وجود بسته TCP در درون بسته IPsec). فیلد Payload Length، طول سرآیند AH را ۲ واحد کمتر، در مبنای کلمات ۳۲ بیتی نشان می دهد.<sup>۲</sup>

فیلد Security Parameter Index، «شناسه اتصال» است. این فیلد توسط فرستنده تنظیم می شود تا رکورد خاصی را در پایگاه اطلاعاتی ماشین گیرنده مشخص کند. این رکورد شامل کلید مشترک و اطلاعات دیگری در خصوص اتصال است. اگر این پروتکل به جای IETF توسط ITU ابداع شده بود این فیلد به نام Virtual Circuit Number نامگذاری می شد!

فیلد Sequence Number برای شماره گذاری تمام بسته هایی است که بر روی یک SA<sup>۳</sup> ارسال می شوند. تمام بسته ها، حتی آنهایی که به هر دلیل از نو ارسال شده اند یک شماره مستقل و یکتا می گیرند. به عبارت دیگر، ارسال مجدد بسته ای که قبلاً نیز فرستاده شده با شماره جدید انجام می شود (حتی اگر شماره ترتیب آن در بسته TCP یکسان باشد). هدف از این فیلد آن است که «حمله ارسال تکراری» (Replay Attack) کشف شود. این شماره ترتیب هیچگاه از نو به صفر بر نخواهد گشت [اصطلاحاً Wrap around نیست] و هرگاه تمام ۲<sup>۳۲</sup> حالت آن استفاده شد برای ادامه مبادله اطلاعات باید یک SA جدید به وجود بیاید.

نهایتاً به فیلد Authentication Data می رسم که فیلدی با طول متغیر است و امضای دیجیتالی داده های درون بسته در آن قرار می گیرد. وقتی یک SA بوجود آمد، ابتدا طرفین در خصوص الگوریتم امضای دیجیتالی که از آن استفاده خواهند کرد مذاکره و توافق می کنند. در اینجا عموماً از روشهای مبتنی بر کلید عمومی (Public Key) استفاده نمی شود چرا که بسته ها باید بی نهایت سریع پردازش شوند در حالی که روشهای کلید عمومی بسیار کند هستند. از آنجایی که IPsec مبتنی بر رمزنگاری با کلید متقارن است و گیرنده و فرستنده قبل از ایجاد SA، بر روی یک کلید مشترک مذاکره و توافق می کنند لذا از همین کلید برای محاسبه امضای دیجیتالی استفاده می شود. ساده ترین راه برای احراز هویت داده ها آنست که خلاصه درهم شده (Hash) کل بسته به انضمام کلید مشترک استخراج شود و در این فیلد قرار بگیرد. البته کلید مشترک هرگز بر روی خط ارسال نمی شود بلکه فقط برای محاسبه این فیلد به بسته اضافه می شود. به ساختاری شبیه به این روش، اصطلاحاً HMAC<sup>۴</sup> گفته می شود. این

۱. مقدار ۵۱ نشانگر وجود بسته IPsec در درون بسته IP است. م.

۲. طول سرآیند AH متغیر است. م.

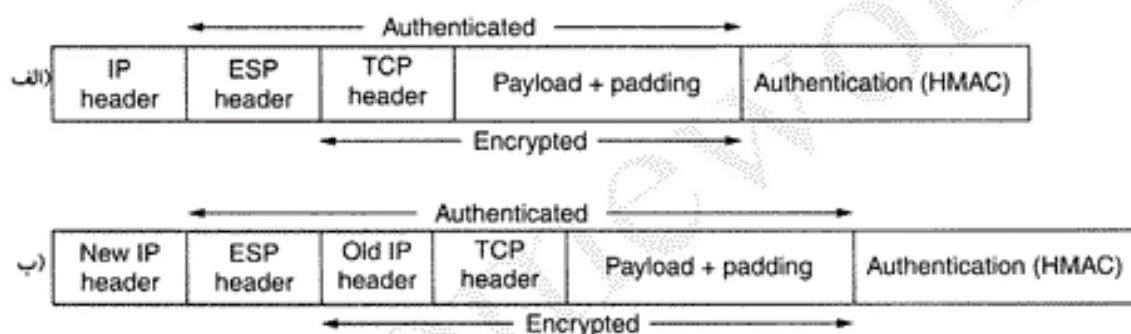
۳. SA را یک نشست یا اتصال یکطرفه فرض کنید. م.

۴. Hashed Message Authentication Code.

روش از نظر حجم محاسبات بسیار سریعتر از آنست که ابتدا SHA-1 اجرا شود و سپس الگوریتم RSA بر روی نتیجه آن اعمال گردد.

سرآیند AH امکان رمزنگاری داده‌ها را فراهم نکرده است و بالطبع زمانی مفید است که فقط صحت داده‌ها اهمیت داشته باشد و نیازی به ارسال محرمانه داده‌ها نباشد. یکی از ویژگیهای ارزشمند AH آن است که نظارت بر صحت برخی از فیلدهای بسته IP را که بهیچوجه در خلال گذر از یک مسیر یاب به مسیر یاب دیگر تغییر نخواهند کرد، در برمی‌گیرد. به عنوان مثال فیلد TTL (Time To Live) در بسته IP در هر گام قطعاً تغییر می‌کند فلذا نمی‌توان آن را در محاسبه کد تایید صحت دخالت داد در حالی که فیلد «آدرس مبدا» (Source Address) می‌تواند در محاسبه کد بررسی صحت دخالت داده شود تا دستکاری اخلالگران در آدرس مبدا بسته، ناممکن باشد.

یکی دیگر از سرآیندهای بسته IPsec، «سرآیند ESP» (Encapsulating Security Payload) است. از این سرآیند به نحوی که در شکل ۸-۲۸ نشان داده شده، چه در «حالت انتقال» و چه در «حالت تونل» استفاده می‌شود.



شکل ۸-۲۸. (الف) سرآیند ESP در حالت انتقال. (ب) سرآیند ESP در حالت تونل.

سرآیند ESP، از دو کلمه ۳۲ بیتی تشکیل شده است. این دو فیلد عبارتند از: Security Parameter Index و Sequence Number که تعریف آنها را در AH دیدیم. کلمه سومی که عموماً به دنبال این دو فیلد می‌آید (ولیکن جزو سرآیند محسوب نمی‌شود) فیلد Initialization Vector است که برای رمزنگاری اطلاعات کاربرد دارد مگر آن که از الگوریتم رمزنگاری «پوچ» (Null) استفاده شده باشد، در این صورت از آن صرف‌نظر می‌شود. ESP برای آزمایش صحت داده‌ها (همانند AH)، HMAC را عرضه کرده است ولیکن به جای آنکه در سرآیند ظاهر شود، مطابق با شکل ۸-۲۸ پس از فیلد حاوی داده (Payload)، قرار می‌گیرد. قرار دادن HMAC در انتهای بسته، برای پیاده‌سازی سخت‌افزاری مفید خواهد بود زیرا HMAC می‌تواند در خلال ارسال بیتها و خروج آنها از کارت واسط شبکه به صورت سخت‌افزاری محاسبه و در نهایت به انتهای بسته ضمیمه شود. دقیقاً این همان دلیلی است که در شبکه اترنت و دیگر شبکه‌های محلی، کد CRC به جای آن که در سرآیند فریم ظاهر شود در انتهای فریم می‌آید. با AH (که در ابتدای هر فریم ظاهر می‌شود) بسته ابتدا بافر شده و امضای دیجیتالی آن قبل از ارسال محاسبه می‌شود؛ بدین ترتیب تعداد بسته‌هایی که می‌توان در واحد زمان ارسال کرد کاهش می‌یابد.

هر آنچه که AH می‌تواند انجام بدهد، نه تنها ESP نیز می‌تواند انجام بدهد بلکه کارایی بیشتر و سرعت بالاتری نیز دارد، پس یک سؤال بوجود می‌آید: چرا با داشتن AH خودمان را به زحمت انداخته‌ایم؟ پاسخ این سؤال، ریشه تاریخی دارد؛ در ابتدا AH فقط صحت داده‌ها را (Integrity) و ESP فقط محرمانه ماندن داده‌ها (Secrecy) را تضمین می‌کرد. بعداً امکان بررسی صحت داده‌ها به ESP افزوده شد ولیکن گروهی که AH را طراحی کرده بودند نمی‌خواستند که بعد از آن همه کار اجازه بدهند AH فنا شود. تنها دلیل قانع‌کننده آنها این بود

که AH بخشی از سرآیند بسته IP را در بررسی صحت (Integrity) بسته دخالت می دهد در حالی که ESP این کار را نمی کند ولی این دلیل، پشتوانه ضعیفی دارد. یک استدلال ضعیف دیگر آن است که محصولاتی که از AH حمایت می کنند ولی از ESP حمایت نمی کنند ممکن است مشکلات کمتری در اخذ مجوز صدور از دولت بگیرند زیرا هیچگونه رمزنگاری انجام نمی شود. [محصولات مبتنی بر رمزنگاری گاه محدودیتهای دولتی برای صدور دارد.] احتمالاً در آینده AH از صحنه خارج خواهد شد.

## ۸-۶-۲ دیوارهای آتش (Firewalls)

این قابلیت که بتوان یک کامپیوتر را در هر کجا به کامپیوتری دیگر در جایی دیگر متصل کرد، به منزله یک سکه دو رو است؛ برای اشخاصی که در منزل هستند گردش در اینترنت بسیار لذت بخش است در حالی که برای مدیران امنیت در شرکتها، یک کابوس وحشتناک به حساب می آید. اغلب شرکتها دارای حجم عظیمی از اطلاعات محرمانه و «روی خط» (Online) هستند، مثل اسرار بازرگانی، طرحهای توسعه محصولات، استراتژیهای فروش، تحلیل های اقتصادی و نظایر آنها. دسترسی رقبا به این اطلاعات می تواند تبعات بسیار سهمگینی داشته باشد.

گذشته از خطر نشت اطلاعات به بیرون و افشای اطلاعات داخلی، خطر نفوذ اطلاعات مخرب به درون نیز وجود دارد. بالاخص ویروسها، کرمها و دیگر آفتهای دیجیتالی، می تواند امنیت را درهم بشکنند، داده های ارزشمند را نابود کنند و وقت بسیار زیادی از مسئول شبکه برای ساماندهی به آسیبهای بجا مانده، تلف نماید. این اطلاعات مخرب توسط کارمندان بی دقتی که مثلاً خواسته اند یک بازی کامپیوتری جدید و جذاب را اجرا کنند، به درون شبکه منتقل می شود.

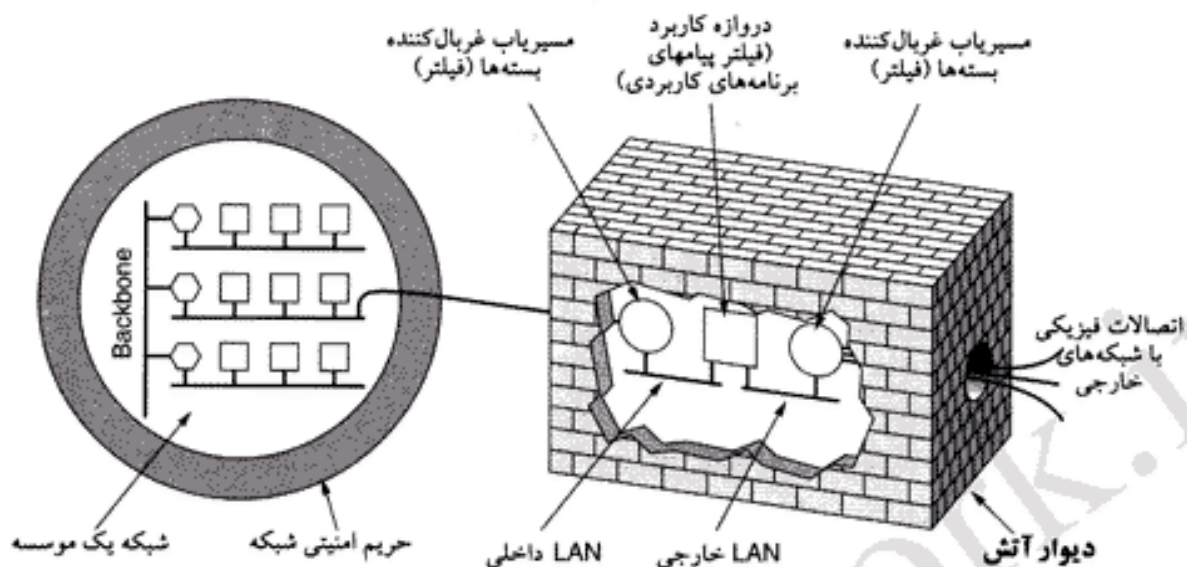
در نتیجه به مکانیزمهایی نیاز است که بتوان بپتتهای «خوب» را از بپتتهای «بد» تفکیک کرد. یک روش آن است که از IPsec استفاده شود. IPsec از داده هایی که بین سایتها در حال تردد هستند بخوبی مراقبت می کند ولیکن هیچ کاری برای پیشگیری از ورود آفتهای دیجیتالی (مثل ویروسها) و اختلالگران به درون شبکه محلی شرکت انجام نمی دهد.

«دیوار آتش» (Firewall) پیاده سازی مدرنی از روش قدیمی و قرون وسطایی حصارهای امنیتی است: خندقی عمیق دور تا دور قلعه خود حفر کنید! این الگو همه را مجبور می کند تا برای ورود یا خروج از قلعه، از یک پل متحرک و واحد بگذرند و بتوان همه را توسط پلیس حراست بازرسی کرد. در دنیای شبکه های کامپیوتری، همین راهکار ممکن خواهد بود: یک شرکت می تواند هر تعداد شبکه محلی داشته باشد که به صورت دلخواه به هم متصل شده اند، اما تمام ترافیک ورودی یا خروجی شرکت صرفاً از طریق یک پل متحرک (همان دیوار آتش) میسر است (شکل ۸-۲۹ را ببینید).

دیوار آتش با پیکربندی شکل ۸-۲۹ دو مولفه دارد: (۱) یک جفت مسیر یاب که عمل غربال سازی بسته ها را انجام می دهند. (Packet Filtering) (۲) دروازه برنامه های کاربردی (Application Gateway). ساختار ساده تری نیز وجود دارد ولیکن حسن بزرگ این طرح آنست که هر بسته باید از دو مرحله غربال سازی (فیلترینگ) و یک مرحله بازرسی محتوایی توسط دروازه، بگذرد. هیچ مسیر دیگری نیز وجود ندارد. خوانندگانی که فکر می کنند فقط یک مرحله بازرسی امنیتی کافی است، به احتمال زیاد اخیراً یک پرواز بین المللی با خطوط هوایی نداشته اند!

هر غربال کننده بسته، یک مسیر یاب استاندارد با برخی از ویژگیهای بیشتر (درخصوص غربال سازی)<sup>۱</sup> است. این قابلیت اجازه می دهد که تمام بسته های ورودی و خروجی بازرسی شوند. بسته هایی که بتوانند برخی از معیارها

۱. تمام مسیر یابهای امروزی قابلیت فیلترینگ را دارند. -م.



شکل ۸-۲۹. یک دیوار آتش شامل دو «فیلترکننده بسته» و یک «دروازه کاربردی».

و شرایط را احراز کنند بطور طبیعی هدایت می شوند و آنهایی که در این بازرسی مردود شوند حذف می گردند. در شکل ۸-۲۹، غربال کننده داخلی (متصل به LAN) بسته های خروجی از شبکه را و غربال کننده بیرونی [متصل به خط ارتباط بیرونی] بسته های ورودی به شبکه را بازرسی می کنند. بسته هایی که بتوانند از اولین مانع عبور کنند، برای بازرسی بیشتر وارد «دروازه برنامه های کاربردی» می شوند. قرار دادن دو غربال کننده این تضمین را می دهد که هیچ بسته ای نتواند قبل از بررسی های ابتدایی و سپس گذر از دروازه، از شبکه خارج یا به آن وارد شود. غربال کننده بسته عموماً بنابر جدولی که توسط مسئول شبکه تنظیم می شود، در خصوص بسته ها تصمیم گیری می کند. در این جدول آدرس مبدا یا آدرس مقصد ماشینهای مجاز و ماشینهای غیرمجاز و همچنین قواعدی در خصوص شرایط تردد بسته ها درج می شود.

در شبکه هایی که عموماً با TCP/IP پیگیری شده اند مبدا و مقصد با آدرس IP و شماره پورت مشخص می شود. شماره پورت مشخص می کند که چه سرویسی مورد نظر است. به عنوان مثال پورت شماره ۲۳ از TCP متعلق به سرویس TelNet، پورت ۷۹ از TCP متعلق به سرویس Finger و پورت ۱۱۹ از TCP متعلق به سرویس خبررسانی یوزنت می باشد. یک شرکت می تواند تمام بسته ها با هر آدرس IP را که با یکی از شماره پورتهای بالا ترکیب شده است، حذف نماید. در این حالت هیچ شخصی در خارج از شرکت قادر نخواهد بود که از طریق TelNet به یک ماشین وارد شود (Log کند) یا از طریق سرویس Finger فهرست افرادی را که در حال کار با شبکه هستند، بدست بیاورد. همچنین یک شرکت باید با حذف بسته های یوزنت، مانع از آن شود که کارمندانش روز خود را با خواندن اخبار بگذرانند.

مسدود و حذف کردن بسته های خروجی از شبکه نیاز به زیرکی بیشتری دارد زیرا اگرچه اکثر سایتها خودشان را مقید به شماره گذاری استاندارد پورتها کرده اند ولی اجباری به انجام این کار نیست.<sup>۱</sup> گذشته از آن، در سرویسهای بسیار مهمی نظیر FTP (پروتکل انتقال فایل) شماره پورت به صورت پویا تعیین می شود. اگرچه

۱. مثلاً اغلب سایتها از پورت ۸۰ صرفاً برای سرویس دهنده وب استفاده می نمایند ولی برخی از افراد برای رد گم کردن شماره پورت ۸۰ را برای سرویس دهنده های دیگر (مثل Telnet یا NetCat) انتخاب کرده اند. -م.

مسدود ساختن اتصالات TCP (از طریق حذف بسته ها) کاری مشکل است، حذف بسته های UDP حتی از آن هم دشوارتر است چراکه هیچ آگاهی اولیه در خصوص آن که بسته چه کاری انجام خواهد داد وجود ندارد. بسیاری از غربال کننده های بسته به گونه ای پیکربندی شده اند که به سادگی ورود و خروج بسته های UDP را در دو جهت مسدود و قدغن کنند.

نیمه دوم یک دیوار آتش «دروازه برنامه های کاربردی یا Application Gateway» است. این قسمت به جای آن که بررسی خود را بر روی مشخصات بسته های خام متمرکز کند در سطح لایه کاربرد عمل می کند. مثلاً دروازه پست الکترونیکی (Mail Gateway)، دروازه ای است که براساس مشخصات هر پیام تصمیم می گیرد که آیا ورود یا خروج آن مجاز است یا خیر! برای هر پیام، «دروازه» با بررسی فیلدهای سرآیند پیام، طول پیام یا حتی محتوای پیام، تصمیم به حذف یا ارسال آن می گیرد. (به عنوان مثال در یک مقرر نظامی، وجود کلماتی نظیر «بمب»، «اتمی» در پیام ممکن است به انجام عملیات ویژه بیانجامد.) مؤسسات آزادند که به هر تعداد «دروازه برنامه کاربردی» برای سرویس دهنده های خود نصب نمایند ولی بطور کلی تردد نامه های الکترونیکی و استفاده از وب جهانی در اغلب سازمانها مجاز شمرده می شود. بقیه سرویسها معمولاً مسدود هستند. ترکیب رمزنگاری و غربال سازی بسته ها می تواند ساختاری را ایجاد کند که در آن امنیت در سطحی محدود و در ازای از دست رفتن سهولت (سهولت کاربری و پیکربندی شبکه) بدست آید.

حتی اگر دیوار آتش به درستی پیکربندی شده باشد، باز هم انبوهی از مشکلات امنیتی باقی خواهد ماند. به عنوان مثال اگر دیوار آتش به گونه ای پیکربندی شده باشد که فقط به بسته های یک شبکه خاص (مثل شبکه متعلق به یکی از کارخانه های شرکت) اجازه تردد بدهد، یک اختلالگر قادر خواهد بود با قرار دادن آدرسهای غلط (آدرس مبدا جعلی) از حصار این بازرسی عبور نماید. اگر یکی از افراد در داخل شبکه بخواهد یک سند محرمانه را خارج نماید می تواند آن را رمز کند و یا حتی به یک تصویر تبدیل کرده و آن را در قالب یک فایل JPEG ارسال نماید تا از هر غربال کننده ای که حتی درون متن را جستجو می کند، بگذرد. هنوز این حقیقت را خاطرنشان نکرده ایم که ۷۰ درصد از کل حملات از درون شبکه و قبل از دیوار آتش صورت می گیرد که به عنوان مثال توسط کارمندان ناراضی هدایت می شود. (Schneier, 2000)

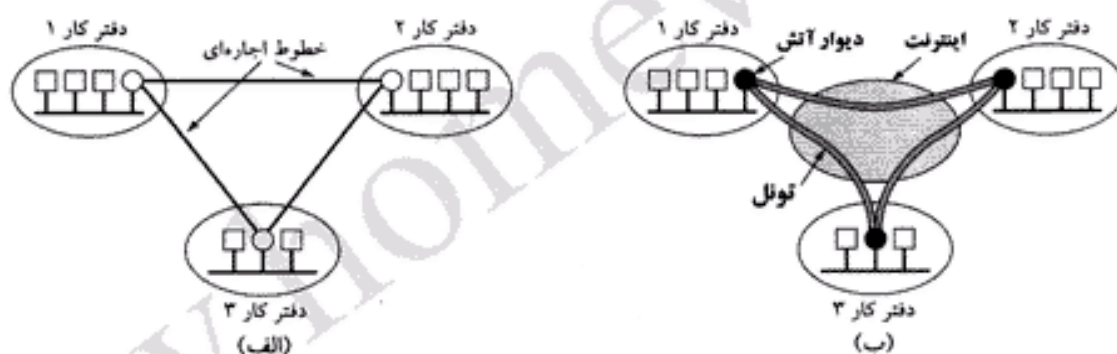
به علاوه، یک رده کامل از حملات وجود دارند که دیوار آتش هیچ کاری در مواجهه با آنها نمی تواند انجام بدهد. ایده اصلی دیوار آتش آن است که جلوی ورود اختلالگران را به شبکه و خروج اطلاعات محرمانه از آن را بگیرد. متأسفانه افرادی هستند که هیچ کاری برایشان بهتر از آن نیست که یک سایت خاص را از کار بیندازند. آنها این کار را با ارسال بسته های مجاز در تعداد بسیار زیاد به یک هدف در شبکه انجام می دهند تا هدف زیر بار بالا در هم بشکند. به عنوان مثال برای زمین گیر کردن یک سایت وب، اختلالگر می تواند حجم بسیار زیادی بسته های TCP SYN را برای برقراری اتصال TCP به سوی آن ماشین روانه کند. سایت مربوط جدولی را (به ازای هر تقاضای اتصال) تخصیص داده و در پاسخ بسته SYN+ACK باز پس می فرستد. اگر اختلالگر هیچ واکنشی به این بسته های پاسخ نشان ندهد، جدول تخصیص داده شده برای چندین ثانیه و تا زمان انقضای مهلت، در حافظه باقی خواهد ماند. اگر اختلالگر بتواند هزاران بسته تقاضای اتصال (TCP SYN) را ارسال کند، جدول مربوطه پر شده و از آن به بعد برقرای هیچ ارتباط مجاز نیز ممکن نخواهد بود. حملاتی که در آنها مقصود اختلالگر به جای سرقت اطلاعات از کار انداختن یک هدف در شبکه است اصطلاحاً «حمله DoS»<sup>۱</sup> نامیده می شود. معمولاً بسته های تقاضا دارای آدرس مبدا غلط هستند و بدین ترتیب براحتی نمی توان اختلالگر را تعقیب کرد.

۱. حمله اختلال در سرویس دهی یا Denial of Service

یک حالت بسیار خطرناکتر آن است که اخلاکگر توانسته باشد به صدها کامپیوتر در هر کجای دنیا نفوذ کند و آنها را برای حمله به یک هدف مشترک در یک زمان مشخص، تحت فرمان خود در آورد. این روش نه تنها قدرت حمله اخلاکگر را افزایش می دهد بلکه احتمال تعقیب و یافتن او نیز کاهش می یابد زیرا بسته هایی به هدف گسیل می شوند که متعلق به ماشین کاربران عادی و غیر مشکوک هستند. به چنین حمله ای اصطلاحاً «حمله DDoS»<sup>۱</sup> گفته می شود؛ مقابله با چنین حمله ای واقعاً مشکل است. حتی اگر ماشین تحت حمله بتواند سریعاً درخواستهای غیر متعارف را از درخواستهای مجاز تشخیص بدهد زمانی طول می کشد تا پردازش و حذف شوند و اگر این درخواستها در ثانیه، از حدی بیشتر شود کل زمان CPU صرف پردازش آنها خواهد شد.

### ۸-۶-۳ شبکه های خصوصی مجازی (VPN)

بسیاری از شرکتها داری دفاتر و کارخانه هایی هستند که در شهرها و گاهی در چندین کشور پراکنده اند. در ایام گذشته، قبل از ایجاد شبکه های عمومی داده، برای اتصال شبکه های پراکنده متعلق به یک شرکت، رایج ترین کار استفاده از خطوط اجاره ای (Leased Line) متعلق به شرکت های مخابرات بود. شبکه ای که از کامپیوترهای یک شرکت و خطوط اجاره ای تلفن تشکیل شده اصطلاحاً «شبکه خصوصی» (Private Network) نامیده می شود. مثالی از یک شبکه خصوصی که سه شبکه را به هم متصل ساخته در شکل ۸-۳۰ الف نشان داده شده است.



شکل ۸-۳۰. (الف) یک شبکه خصوصی با خطوط اجاره ای. (ب) شبکه خصوصی مجازی.

شبکه های خصوصی، بسیار خوب و مطمئن عمل می کنند. اگر خطوط در اختیار شرکت، تماماً اجاره ای باشند، هیچ ترافیکی نمی تواند به بیرون از شرکت منتقل شود و اخلاکگر مجبور است به صورت فیزیکی از خطوط انتقال انشعاب گرفته و بداند متصل شود که انجام این کار نیز ساده نخواهد بود. مشکل بزرگ شبکه های خصوصی آنست که اجاره کردن یک خط T1 [با نرخ ارسال 1.544Mbps] در هر ماه هزاران دلار هزینه دارد؛ خطوط T3 نیز چندین برابر گرانتر هستند. وقتی شبکه های عمومی داده<sup>۲</sup> و بعد از آن اینترنت به صحنه آمد، بسیاری از شرکتها تصمیم گرفتند که انتقال داده های خود (و احتمالاً انتقال صوت) را از طریق شبکه های عمومی موجود انجام بدهند که هزینه ناچیزی دارد ولیکن در عوض امنیت یک شبکه خصوصی را ندارد.

احساس این نیاز به ابداع VPN (شبکه خصوصی مجازی) انجامید که بر روی زیرساخت شبکه عمومی بنا شده است ولی اکثر ویژگیهای یک شبکه خصوصی را عرضه می کند. این گونه شبکه ها از آن جهت «مجازی» نامیده شده اند که صرفاً یک توهم (یا بهتر بگوییم یک مدل انتزاعی) هستند؛ دقیقاً مثل «مدار مجازی» که در آن هیچ مدار واقعی در کار نیست یا «حافظه مجازی» که در آن هیچ حافظه واقعی از نوع RAM وجود ندارد.

اگرچه VPN را می توان بر روی ATM (یا شبکه Frame Relay) پیاده کرد ولیکن عمومیتش روشن آن است که VPN مستقیماً بر روی اینترنت بنا نهاده شود. رایجترین طرح آن است که هر دفتر کار [از یک شرکت] به یک دیوار آتش مجهز شود و به گونه ای که در شکل ۸-۳۰<sup>۱</sup> نشان داده شده یک تونل بین هر دو دفتر از طریق خطوط عمومی اینترنت ایجاد شود. اگر از IPsec برای ایجاد تونل بین این دو دفتر استفاده شده باشد می توان ترافیک جاری بین دو طرف ارتباط را از طریق یک SA<sup>۲</sup> که احراز هویت و رمزنگاری شده است، ارسال کرد و بدین ترتیب صحت و امنیت داده ها تضمین می شود و همچنین ایمنی قابل توجهی در مقابل خطر «تحلیل ترافیک» (که در بخش IPsec بدان اشاره شد) بدست می آید.

وقتی این سیستم فعال گردد، زوج دیوار آتش مستقر در دو شبکه باید ابتدا در خصوص پارامترهای SA شامل نوع حالت [حالت انتقال/حالت تونل]، نوع سرویسها و نوع الگوریتم و کلیدها مذاکره و توافق کنند. بسیاری از دیوارهای آتش قابلیت ایجاد VPN را به صورت درونی در خود دارند اگرچه امروزه برخی از مدل های معمولی مسیریابها نیز همین کار را بخوبی انجام می دهند، ولیکن از آنجایی که دیوارهای آتش ذاتاً در محیط های امن بکار گرفته می شوند بنابراین طبیعی است که تونلهایی داشته باشیم که از یک دیوار آتش شروع و به دیگری ختم می گردند تا تکنیک کاملی بین اینترنت و شبکه یک شرکت ایجاد نماییم. بدین ترتیب دیوارهای آتش، VPN و IPsec به همراه ESP (در حالت تونل) ترکیب طبیعی برای چنین محیط هایی است و در عمل به صورت گسترده ای از این ترکیب استفاده می شود.

پس از آن که SA ایجاد شد، ترافیک می تواند جریان پیدا کند. از دیدگاه مسیریابهایی که در درون ساختار شبکه اینترنت واقع هستند بسته هایی که از تونل VPN منشأ گرفته اند هیچ تفاوتی با بسته های معمولی IP ندارند.<sup>۲</sup> تنها چیزی که بسته های IP حاوی داده های معمولی را از بسته های IP حاوی بسته IPsec جدا می کند سرآیند بسته IPsec است که دقیقاً بعد از سرآیند بسته IP معمولی قرار گرفته است ولی برای مسیریابها این سرآیند اضافی هیچ اهمیتی ندارد و در شرایط عادی تأثیری بر عمل هدایت و مسیریابی نمی گذارد چراکه مسیریابها عموماً به این سرآیند اضافی اعتنایی نمی کنند.

بزرگترین حسن ایجاد شبکه VPN آن است که بطور کلی از تمام نرم افزارهای کاربر مستقل بوده و هیچ تغییری در آنها نیاز نیست و امکانات آن کاملاً نامرئی است: دیوارهای آتش به صورت مستقل SA های لازم را ایجاد و مدیریت می کنند. تنها کسی که از تنظیمات آن اطلاع دارد مسئول شبکه است که طبقاً موظف به پیکربندی و مدیریت دیوارهای آتش می باشد. برای بقیه افراد، این شبکه دقیقاً مشابه با شبکه های خصوصی با خطوط اجاره ای است. برای کسب اطلاعات بیشتر در خصوص VPN مرجع (Brown, 1999; Izzo, 2000) را نگاه کنید.

### ۸-۶-۸ امنیت شبکه های بی سیم

از لحاظ منطقی طراحی سیستمی که کاملاً امن باشد یکمک VPN و دیوار آتش، کاری بسیار ساده است ولیکن در عمل شبیه به یک آبکش، اطلاعات از آن نشت خواهد کرد! این وضعیت زمانی اتفاق می افتد که برخی از ماشینهای شبکه بی سیم بوده و از مخابرات رادیویی استفاده کرده باشند که در این صورت در همه جهات پیرامون دیوار آتش، داده های در حال تبادل قابل شنود هستند. محدوده کاری شبکه ۸۰۲.۱۱ حدود چند صد متر است، لذا هر کسی که بخواهد جاسوسی یک شرکت را بنماید می تواند براحتی خودروی خود را صبح زود در پارکینگ کارمندان پارک کرده و یک کامپیوتر کیفی مجهز به شبکه ۸۰۲.۱۱ را در درون خودرو به گونه ای پیکربندی نماید که هر آنچه را می شنود ذخیره کند. عصر همان روز، دیسک سخت این کامپیوتر سرشار از اطلاعات با ارزش خواهد بود. البته از

۱. Security Association. ۲. در حالت تونل بسته های IPsec در درون یک بسته IP معمولی جاسازی می شوند. سم.

دیدگاه تئوری فرض بر آن است که هیچ فردی به بانک دستبرد نمی زند!! بسیاری از مشکلات امنیتی در شبکه های بی سیم به سازندگان ایستگاههای ثابت<sup>۱</sup> بر می گردد که سعی می کنند محصولاتشان هر چه بیشتر ساده و با محیطی دوستانه باشد. عموماً اگر یک کاربر دستگاه جانبی مورد نیاز برای اتصال به شبکه بی سیم را خریداری و آن را به برق متصل کند، بلافاصله عملیاتی شده و شروع به کار می کند در حالی که هیچگونه امنیتی وجود ندارد و تمام اطلاعات محرمانه در محدوده برد شبکه بی سیم فاش خواهد شد. شبکه بی سیم رویای جاسوسان الکترونیکی را محقق کرده است یعنی: «دسترسی آزاد به داده ها بدون نیاز به انجام هیچ کاری!»

بنابراین باید اشاره کرد که امنیت اطلاعات در شبکه های بی سیم از اهمیت و حساسیت بیشتری نسبت به شبکه های مبتنی بر سیم برخوردار است. در این بخش به روشهایی که سعی در برقراری امنیت در شبکه های بی سیم دارند نگاهی خواهیم انداخت. اطلاعات بیشتر را می توانید در (Nichols and Lekkas, 2002) بیابید.

### امنیت شبکه ۸۰۲.۱۱

استاندارد ۸۰۲.۱۱ پروتکلی برای ایجاد امنیت در سطح لایه پیوند داده ها به نام WEP<sup>۲</sup> معرفی کرده که هدف از طراحی آن، برقراری امنیت در شبکه های بی سیم در سطحی معادل با شبکه های مبتنی بر سیم بوده است. از آنجایی که شبکه های محلی مبتنی بر سیم، به صورت پیش فرض هیچ امنیتی ندارند رسیدن به این هدف بسیار ساده است و قطعاً WEP (به گونه ای که خواهیم دید) به این هدف رسیده است!!!!

وقتی گزینه امنیت (WEP) در شبکه بی سیم ۸۰۲.۱۱ فعال شده باشد، هر ایستگاه دارای کلیدی مشترک با ایستگاه ثابت (Base Station) خواهد بود. چگونگی توزیع این کلیدها در استاندارد WEP تعریف نشده است. ممکن است این کلیدها توسط سازنده سخت افزار بی سیم به صورت کاملاً تصادفی انتخاب و از قبل تنظیم شده باشد. بعداً می توان این کلیدها را تعویض کرد. نهایتاً چه ایستگاه ثابت و چه ماشین کاربر (ایستگاههای متحرک) می توانند از طریق این کلید از قبل تعیین شده، یک کلید تصادفی برای خود انتخاب کرده و پس از رمزنگاری، آنرا برای یکدیگر بفرستند. پس از ایجاد و توافق، این کلید می تواند برای ماهها یا حتی سالها ثابت باقی بماند.

WEP برای رمزنگاری از روش Stream Cipher<sup>۳</sup> و الگوریتم RC4 استفاده می کند. توسط رونالد ری وست (از ابداع کنندگان RSA) طراحی شده و الگوریتم آن محرمانه نگاه داشته شده بود تا آن که در سال ۱۹۹۴ این الگوریتم کشف و در اینترنت اعلام شد! قبلاً هم اشاره کرده بودیم که محرمانه نگه داشتن الگوریتم تقریباً غیرممکن است؛ حتی وقتی هدف آن باشد که ویژگیهای عالی آنرا از دیگران مخفی نگاه داریم! (در مورد RC4 نیز هدف همین بود). الگوریتم RC4 بکار رفته در WEP یک Keystream تولید کرده و آنرا با داده های رمز نشده XOR می کند تا متن رمز شده بدست آید.

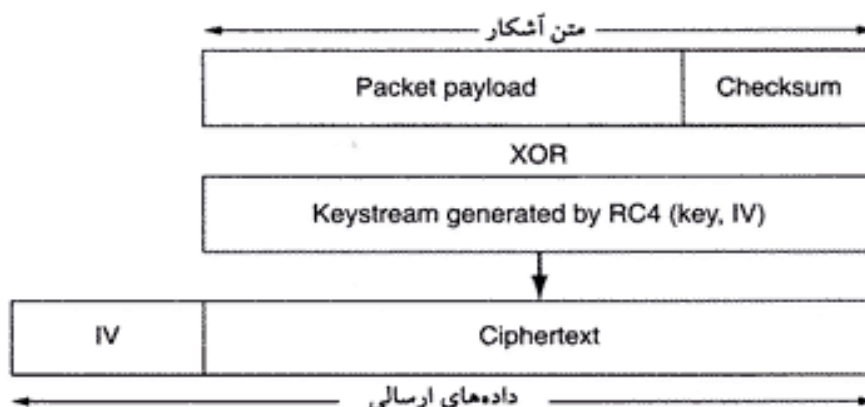
فیلد داده هر بسته اطلاعاتی طبق روشی که در شکل ۸-۳۱ نشان داده شده رمز می شود. ابتدا از اصل پیام با استفاده از یک چند جمله ای CRC-32، یک کد چهار بیتی کشف خطا استخراج و به انتهای داده ها ضمیمه می شود تا مجموع این دو به الگوریتم رمزنگاری تحویل گردد. سپس این داده های رمز نشده با یک Keystream طولانی XOR می شود. حاصل این XOR، داده های رمز شده خواهد بود. IV<sup>۴</sup> مورد نیاز برای شروع در الگوریتم RC4 به همراه داده ها ارسال خواهد شد. وقتی گیرنده بسته ای را دریافت می کند داده های رمزنگاری شده را از درون آن استخراج نموده و براساس IV ارسالی و همچنین کلید مشترک، Keystream را محاسبه کرده و برای رمزگشایی

۱. ایستگاههای ثابت (Base Stations) نقاط دسترسی ایستگاههای رادیویی به شبکه هستند.

۲. Wired Equivalent Privacy.

۳. بخش ۸-۲-۳ را ببینید.

۴. Initialization Vector.



شکل ۸-۳۱. رمزنگاری بسته با استفاده از WEP.

اطلاعات، آن را با محتوای بسته XOR می کند. سپس برای بررسی هر گونه دستکاری در داده ها، کد کشف خطای آن [یعنی CRC] را آزمایش و بررسی می نماید.

اگرچه در نگاه اول این روش بسیار خوب به نظر می رسد ولی راهی برای شکستن آن تدوین و پیشنهاد شده است. (Borisov et al., 2001) در ادامه خلاصه ای از روش درهم شکستن WEP را ارائه می کنیم. اول از همه آنکه در بسیاری از مؤسسات، کلیدی مشترک و یکسان برای همه کاربران تعریف شده که در چنین حالتی یک کاربر می تواند ترافیک تمام کاربران دیگر را براحتی بدست آورده و بخواند. این مسئله دقیقاً مشابه با شبکه اترنت است ولی به هیچوجه امن نیست.

حتی اگر هر کاربر کلیدی مجزا داشته باشد باز هم می توان به WEP حمله کرد. از آنجایی که کلیدها برای مدت زمانی بسیار طولانی تغییر نمی کنند، WEP توصیه کرده که لااقل IV برای هر بسته تغییر کند تا نتوان از طریق حمله Keystream Reuse Attack که شرح آن در بخش ۲-۳ آمد، اطلاعات را رمزگشایی کرد. (تغییر IV فقط توصیه شده ولی اجباری نیست). متأسفانه در بسیاری از کارتهای واسطه شبکه ۸۰۲.۱۱ که برای کامپیوترهای کیفی ارائه شده، وقتی کارت در درون کامپیوتر قرار داده می شود، مقدار IV به صفر تنظیم و به ازای ارسال هر بسته یک واحد IV اضافه می شود. از آنجایی که افراد، این کارت را باز و بسته می کنند، لذا مقدار IV عموماً کم است. اگر ترویدی بتواند تعدادی بسته را که با IV مشابه توسط یک کاربر خاص ارسال شده، جمع آوری کند براحتی قادر خواهد بود محتوای دو بسته را با هم XOR کرده و با حذف کلید احتمالاً رمز آنها را بشکند. (فراموش نکنید که IV به صورت آشکار و به همراه بسته ارسال می شود).

ولیکن حتی اگر کارت شبکه ۸۰۲.۱۱ برای هر بسته یک IV مستقل و تصادفی انتخاب کند باز هم ممکن است از IV مشابه استفاده شود چراکه IV جمعی ۲۴ بیت است و پس از ارسال  $2^{24}$  بسته، از شماره های تکراری استفاده خواهد شد. بدتر از آن، اگر IV به صورت تصادفی انتخاب شود طبق روشی که در بخش ۴-۴-۸ در مورد «حمله روز تولد» اشاره شد، متوسط بسته هایی که باید ارسال شود تا به یک زوج IV مشابه برخورد کنیم حدود ۵۰۰۰ است. ( $2^{24}/2 = 2^{23} = 4096$ ) طبق این استدلال، اگر ترویدی برای چند دقیقه به بسته های در حال مبادله گوش بدهد قادر خواهد بود حداقل دو بسته با IV یکسان و کلید مشابه بدست آورد. با XOR کردن دو بسته کلید حذف شده و حاصل XOR اصل دو پیام، بدست می آید. این رشته بیت را می توان به روشهای مختلف مورد حمله قرار داد تا اصل پیامها بازیابی شود. با اندکی کار بیشتر، Keystream متناظر با آن IV بدست خواهد آمد. ترویدی می تواند کار خود را اندکی ادامه بدهد و یک دیکشنری برای Keystream متناظر با تمام IVها تشکیل دهد. پس از شکسته شدن IV، تمام بسته هایی که در آینده ارسال خواهد شد (یا حتی در گذشته ارسال شده) براحتی قابل رمزگشایی

است. گذشته از آن، چون که IVها به صورت تصادفی بکار می‌روند به محض آن که ترویدی بتواند یک زوج (IV و Keystream) را تعیین کند، قادر خواهد بود بسته‌های دلخواه خود را با آن رمز کرده و به صورت جعلی برای یکی از طرفین بفرستد و بدین ترتیب در مبادله اطلاعات بین کاربر و ایستگاه ثابت مداخله نماید. از دیدگاه تئوری، برای کشف این موضوع، گیرنده باید تمام بسته‌هایی که به ناگاه و با IV مشابه ارسال می‌شوند را بررسی نماید تا از این حمله آگاه گردد ولیکن دو اشکال وجود دارد: اول آن که WEP اجازه چنین کاری را داده است<sup>۱</sup>؛ دوم آنکه هیچ کس چنین بررسی و آزمایشی را انجام نمی‌دهد!

در آخر باید اشاره کنیم که CRC هیچ کار با ارزشی انجام نمی‌دهد چرا که ترویدی قادر است محتوی درون هر بسته را تغییر داده و CRC متناظر با آن را تولید و به آن بیفزاید، بدون آن که لازم باشد برای این کار اطلاعات از رمز خارج شود. کوتاه سخن آن که شکستن ۸۰۲.۱۱ تقریباً ساده و سراسر است. ما به تمام فهرست حملاتی که آقای Borisov کشف کرده نپرداختیم.

در آگوست سال ۲۰۰۱، یک ماه پس از انتشار مقاله Borisov، یک حمله ویرانگر بر علیه WEP توسط Fluhrer، تداوین و گزارش شد. این شخص کشف کرد که بسیاری از کلیدهای بکار رفته برای رمزنگاری دارای ویژگی خاصی هستند که می‌توان از روی Keystream، برخی از بیت‌های کلید را استخراج کرد. اگر این حمله چندین بار تکرار شود، استخراج تمام بیت‌های کلید با تلاش بسیار کمی ممکن خواهد بود. البته به غیر از ارائه تنوریک این ادعا، Fluhrer و گروهش هیچ تلاشی برای شکستن یک شبکه محلی مبتنی بر ۸۰۲.۱۱ نکرده بود.

در مقابل، وقتی یک دانشجوی کارآموز و دو محقق در آزمایشگاه AT&T از حمله گزارش شده توسط Fluhrer آگاه شدند تصمیم گرفتند آن را در عمل آزمایش کنند. پس از یک هفته تلاش، آنها توانستند اولین کلید ۱۲۸ بیتی را بر روی یک محصول واقعی ۸۰۲.۱۱ پیدا کنند، چندین هفته نیز به دنبال یافتن کارتهای سخت‌افزاری ۸۰۲.۱۱، تهیه مجوز خرید و نصب و آزمایش آنها بودند. برنامه‌نویسی لازم فقط دو ساعت طول کشید!

وقتی این پژوهشگران نتایج کار خود را اعلام کردند، CNN گزارشی خبری تحت عنوان "Off-the-shelf Hack Breaks Wireless Encryption" به چاپ رساند که در آن برخی از صنایع، نتایج کار آنها را به مسخره گرفته بودند؛ با ذکر این نکته که کار آنها براساس تئوری Fluhrer بوده و کاملاً بدیهی و بی‌ارزش است. اگرچه استدلال این صنایع از دیدگاه فنی درست است ولی این حقیقت را نمی‌توان نادیده گرفت که تلاش مشترک این دو گروه پرده از یک اشکال اساسی در ۸۰۲.۱۱ برداشت.

در هفتم سپتامبر ۲۰۰۱، IEEE با اقرار به این واقعیت که WEP بطور کامل قابل شکستن است با صدور یک اعلامیه کوتاه شامل شش بند جوابیه‌ای را منتشر ساخت که می‌توان آن را به صورت زیر خلاصه کرد:

۱. ما گفته بودیم که امنیت WEP بهتر از امنیت در شبکه اترنت نیست!!

۲. همین امنیت ناقص بهتر از نبود آنست!

۳. سعی کنید از روشهای امنیتی دیگر بهره بگیرید. (مثلاً امنیت در سطح لایه انتقال)

۴. نسخه بعدی، 802.11i امنیت بالاتری خواهد داشت.

۵. صدور گواهینامه برای محصولات ۸۰۲.۱۱ منوط به استفاده از استاندارد 802.11i خواهد بود.

۶. در تلاش هستیم که روشی برای امن کردن آن تا قبل از معرفی 802.11i ارائه بدهیم.

این قضیه را بدان منظور بررسی کردیم تا خاطرنشان کنیم که رسیدن به امنیت کامل حتی برای خبرگان این فن ساده نیست.

۱. یعنی WEP ارسال چند بسته با IVهای مشابه را منع نکرده است و ممکن است برخی از کارتهای شبکه چنین کنند. -م.

## امنیت در تکنولوژی بلوتوث (Bluetooth)

بلوتوث بُرد کوتاهتری نسبت به شبکه ۸۰۲.۱۱ دارد لذا در این شبکه نمی توان با پارک در پارکینگ یک مؤسسه و استراق سمع داده ها به آن حمله کرد ولی کماکان امنیت بلوتوث مورد مهمی بشمار می رود. به عنوان مثال فرض کنید که کامپیوتر آلیس مجهز به یک صفحه کلید بی سیم مبتنی بر بلوتوث است. اگر در این شبکه امنیت وجود نداشته باشد، ترویدی در دفتر مجاور آلیس، می تواند هر آنچه را آلیس تایپ می کند (حتی نامه های ارسالی او) را بخواند. او همچنین می تواند هر آنچه را که آلیس برای چاپ بر روی «چاپگر بلوتوث» می فرستد بدست آورد (مثل نامه های دریافتی یا گزارشهای محرمانه). خوشبختانه بلوتوث، ساختار امنیتی دقیقی دارد و تلاش می کند تا فعالیتهای اخلالگران دنیا را خنثی کند. در ادامه ویژگیهای اصلی این ساختار را به اختصار بررسی می نماییم.

بلوتوث سه حالت امنیتی متفاوت در محدوده «بدون رمزنگاری» تا «رمزنگاری کامل» و «امکان بررسی صحت داده ها» دارد. همانند شبکه ۸۰۲.۱۱ هر گاه گزینه امنیت غیرفعال شده باشد (که به صورت پیش فرض این گونه است) هیچ امنیتی برای داده ها وجود ندارد. بسیاری از کاربران گزینه امنیت را غیرفعال نگه می دارند تا وقتی که یک اشکال جدی برایشان اتفاق بیفتد؛ پس از آن امنیت را فعال می کنند. در دنیای کشاورزی به این بی احتیاطی «بستن در اصطبل پس از فرار اسب» گفته می شود!!!

بلوتوث امنیت را در چندین لایه عرضه کرده است: در لایه فیزیکی تعویض مستمر فرکانس (Frequency Hopping) امنیت ناچیزی عرضه می کند ولیکن از آنجایی که در ابزارهای مبتنی بر این تکنولوژی قبل از پرش به یک فرکانس خاص باید ترتیب تغییر فرکانس به اطلاع طرفین برسد لذا این تغییر محرمانه نیست و برای اخلالگر قابل شنود است. امنیت واقعی زمانی آغاز می شود که یک دستگاه «پیرو» (Slave) که تازه به شبکه وارد شده (مثل صفحه کلید مبتنی بر بلوتوث) از دستگاه «اصلی» Master (مثل کامپیوتر) تقاضای یک کانال می کند. فرض شده که این دو دستگاه دارای یک کلید مشترک سری هستند که از قبل درون آنها درج شده است. در برخی از حالات این کلید مشترک به صورت سخت افزاری توسط کارخانه سازنده بر روی آن ابزار ذخیره می شود. (به عنوان مثال همانند شماره ای که به صورت پیش فرض بر روی گوشی تلفن همراه وجود دارد). در برخی دیگر از حالات یکی از دستگاهها دارای کلیدی تعبیه شده بر روی سخت افزار است در حالی که کاربر (برای فعال کردن دستگاه) مجبور است این کلید را در قالب عددی دهدی وارد کند. این کلیدهای مشترک اصطلاحاً «کلیدهای عبور» (Passkeys) نامیده می شوند.

برای ایجاد یک کانال، ماشین اصلی (Master) و ماشین «پیرو» (Slave) هر یک بررسی می کنند که آیا دیگری کلید عبور را می داند؟ در این صورت، با یکدیگر در خصوص آنکه (۱) آیا اطلاعات کانال رمزنگاری شود؛ (۲) صحت آنها بررسی شود (۳) یا هر دو کار انجام شود؛ مذاکره و توافق می کنند. سپس یک کلید ۱۲۸ بیتی که برخی از بیهیهای آن عمومی و آشکار است برای نشست انتخاب می نمایند. این نکته که در بلوتوث اجازه داده شده با معلوم بودن برخی از بیتها کلید رمز ضعیف باشد، بدان دلیل است که در برخی از کشورها طبق قوانین دولتی، اجازه بکارگیری یا صدور محصولاتی که در آنها کلید رمز طولانی است و دولت قادر نیست رمز آن را بشکند، ممنوع است.

رمزنگاری در بلوتوث به روشی مبتنی بر Stream Cipher<sup>۱</sup> که  $E_0$  نام دارد، انجام می شود. بررسی صحت اطلاعات نیز به روش SAFER+ انجام می گیرد. این دو روش براساس روش معمولی رمزنگاری با کلید متقارن هستند. SAFER+ برای مسابقه AES (که نهایتاً به پیروزی Rijndael انجامید) ارسال شد ولیکن در همان مراحل مقدماتی از دور مسابقه خارج گردید چراکه از بقیه روشهای پیشنهادی کُندتر بود. طراحی شبکه بلوتوث قبل از

انتخاب برنده AES، پایان یافته بود و گرنه به احتمال زیاد در آن از روش Rijndael استفاده می شد. روش واقعی رمزنگاری بکار رفته در Stream Cipher در شکل ۸-۱۴ نشان داده شده است که در آن متن اصلی با کلید هر مرحله XOR (Stream Key) شده و متن رمز را بدست می دهد. متأسفانه  $E_0$  نیز (شبهه به RC4) می تواند اشکالات اساسی داشته باشد (Jokobson & Wetzel, 2001). اگرچه هنوز  $E_0$  شکسته نشده است (تا زمان نوشتن این کتاب) ولیکن شباهتهای آن با رمز A5/I که اشکال واضح آن ترافیک تلفنهای همراه GSM را در معرض حمله قرار داده است به این نگرانی دامن می زند. گاهی این موضوع افراد را سردرگم و متعجب می کند که چرا در بازی همیشگی موش و گربه بین متخصصین رمزنگار و رمزشکن، بیشتر اوقات رمزشکنها برنده هستند! یکی دیگر از موارد امنیتی آن است که بلوتوث صرفاً «دستگاه» را احراز هویت می کند نه کاربر را، فلذا یک دستگاه دزدیده شده مبتنی بر بلوتوث می تواند به سارق اجازه دسترسی به حساب کاربری و دیگر امکانات صاحب آن دستگاه را بدهد. با این حال، بلوتوث امنیت را در لایه های بالاتر نیز پیاده سازی کرده است؛ بنابراین حتی اگر امنیت در لایه پیوند داده ناپود شود، در لایه های بالاتر باقی خواهد ماند؛ بالاخص در برخی از برنامه های کاربردی این ویژگی مثبت وجود دارد که گاهی برای آن که کاربر بتواند کاری را انجام بدهد از او کد PIN (شماره شناسایی شخصی) مطالبه می کنند.

#### امنیت در WAP 2.0

در اکثر بخشها، مجمع توسعه دهنده WAP از غیراستاندارد بودن پشت پرده پروتکلی WAP 1.0 درس گرفته و به همین دلیل WAP 2.0 در تمام لایه ها به طرز گسترده ای از پروتکل های استاندارد استفاده می کند. از آنجایی که WAP مبتنی بر IP است در لایه شبکه به طور کامل از IPsec حمایت می کند. در لایه انتقال نیز، از یک اتصال TCP به کمک TLS حفاظت می شود. (TLS استاندارد IETF است که در همین فصل بدان خواهیم پرداخت.) در لایه بالاتر از روش «احراز هویت HTTP» که در RFC 2617 تشریح شده است، بخوبی حمایت می شود. وجود یک کتابخانه سیستمی (Library) در سطح لایه کاربرد به منظور رمزنگاری، امکانات کافی جهت کنترل صحت و غیرقابل انکار بودن پیامها را در اختیار برنامه نویسان WAP قرار داده است. از آنجایی که WAP 2.0 مبتنی بر استانداردهای شناخته شده است این شانس بزرگ وجود دارد که سرویسهای امنیتی آن بالاخص سرویسهای احراز هویت، بررسی صحت داده ها، غیرقابل انکار بودن و محرمانه ماندن پیامها، از امنیت 802.11 و Bluetooth بهتر و مطمئن تر باشد.

### ۷-۸ پروتکل های احراز هویت

«احراز هویت» (Authentication) روشی است که براساس آن یک پروسه بررسی می کند که آیا شریک او در یک ارتباط (یعنی پروسه طرف مقابل)، همانی است که باید باشد یا یک نفوذگرسست که خود را به جای طرف واقعی جا زده است. بررسی هویت واقعی یک پروسه راه دور در شرایطی که با اختلالگران فعال و بدخواه روبرو هستیم فرآیندی بسیار دشوار است و به پروتکل های پیچیده مبتنی بر رمزنگاری نیاز دارد. در این بخش برخی از پروتکل های بی شمار احراز هویت را که در شبکه های ناامن مورد استفاده قرار می گیرد، مطالعه خواهیم کرد.

گاهی مردم اصطلاح «احراز هویت» (Authentication) را با «صدور مجوز» (Authorization) اشتباه می گیرند. «احراز هویت» با این سؤال سر و کار دارد که آیا شما حقیقتاً در حال محاوره و تبادل اطلاعات با یک پروسه خاص هستید. «صدور مجوز» با این مقوله سر و کار دارد که یک پروسه، مجوز انجام چه کارهایی را دارد. به عنوان مثال، یک پروسه مشتری با یک سرویس دهنده فایل ارتباط برقرار کرده و اعلام می کند: «من پروسه اسکات» هستم و می خواهم فایل `cookbook.old` را پاک کنم. از دیدگاه سرویس دهنده فایل پاسخ دو سؤال

باید مشخص شود:

۱. آیا این پروسه حقیقتاً پروسه «اسکات» است؟ (احراز هویت)
۲. آیا «اسکات» اجازه حذف فایل *cookbook.old* را دارد؟ (صدور مجوز)

پس از آن که پاسخ این دو سؤال، بدون هیچ ابهامی مثبت ارزیابی شد، عمل درخواستی قابل انجام است. سؤال اول حساس تر و کلیدی تر است. پس از آن که سرویس دهنده فایل متوجه شد که با چه کسی صحبت می کند، بررسی مجوزها در حد یک جستجوی ساده درون جدول یا پایگاه اطلاعاتی محلی است. به همین دلیل ما در این بخش صرفاً بر روی موضوع احراز هویت متمرکز خواهیم شد.

یک مدل عمومی که تمام پروتکل های احراز هویت از آن استفاده می کنند بدین نحو است که مثلاً: آلیس کارش را با ارسال پیامی به باب یا یک «KDC مورد اعتماد»<sup>۱</sup> که صادق و امین همه است، شروع می کند. چندین پیام دیگر بین طرفین و در دو جهت مبادله می شود. ممکن است در حالی که این پیامها در حال مبادله هستند شخص ثالثی مثل ترویدی مشغول استراق سمع، دستکاری در پیام یا تکرار پیامها باشد تا بدین نحو آلیس یا باب را فریب بدهد یا در کار آنها اختلال کند.

علیرغم تمام این کارشکنی ها، وقتی عملکرد پروتکل تکمیل شده باشد، آلیس مطمئن خواهد بود که در حال صحبت با باب است و باب هم اطمینان دارد که با آلیس محاوره دارد. به علاوه در اغلب پروتکلها، دو طرف یک «کلید سری نشست» ایجاد می کنند تا در محاوره خود از آن [برای رمزنگاری] استفاده نمایند. در عمل و به دلایل سرعت و کارایی، تمام ترافیک داده ها در حین نشست به روش رمزنگاری با کلید متقارن (عموماً AES یا DES) رمز می شوند، اگرچه از روش «رمزنگاری کلید عمومی» در پروتکل های احراز هویت برای ایجاد «کلید نشست» در سطح گسترده ای استفاده می شود.

دلیل آنکه برای هر «اتصال» جدید یک کلید تصادفی به عنوان کلید نشست انتخاب می شود آنست که حجم ترافیکی که مستقیماً توسط کلید سری یا کلید عمومی کاربر رمز می گردد حداقل بماند و در نتیجه میزان اطلاعات رمز شده (که در تمام آنها از یک کلید استفاده شده) کاهش یابد. همچنین هرگاه پروسه ای دچار اشکال شده و در هم بشکند (crash کند)، ممکن است «تصویر حافظه» آن پروسه در اختیار افراد ناباب قرار بگیرد و آنها بتوانند کلید اصلی کاربر را از درون آن استخراج کنند.<sup>۲</sup> در این حالت حتی اگر کلیدی فاش شود کلید نشست خواهد بود که آنها ثابت نیست. تمام کلیدهای ثابت و دائمی پس از آن که نشست برقرار شد از حافظه پروسه ها پاک شده و فقط از کلید نشست استفاده می شود.

## ۸-۷-۱ احراز هویت براساس کلید مشترک و سری

در اولین پروتکل احراز هویت، فرض می کنیم که آلیس و باب قبلاً در مورد یک کلید سری به نام  $K_{AB}$  با یکدیگر توافق کرده اند. ممکن است این کلید را با استفاده از تلفن یا از طریق یک شخص معتمد به اطلاع یکدیگر رسانده باشند ولی در هر حال فرض بر آن است که این کلید سری را از طریق شبکه ناامن، برای یکدیگر ارسال نکرده اند. این پروتکل بر اصولی استوار است که تمام پروتکل های دیگر احراز هویت نیز از آن تبعیت می کنند: «یکی از طرفین عددی تصادفی برای دیگری ارسال می کند و طرف مقابل تبدیل خاصی را بر روی آن اعمال کرده و نتیجه را بر می گرداند». چنین پروتکل هایی اصطلاحاً پروتکل های «چالش-پاسخ» (Challenge-Response) نامیده می شوند.

۱. KDC: مرکز توزیع کلید یا Key Distribution Center

۲. در سیستم های عاملی مثل یونیکس هرگاه پروسه ای دچار اشکال شده و crash کند، کل فضای حافظه در اختیار آن پروسه بر روی دیسک سخت ذخیره می شود تا بتوان برای عملیاتی نظیر Recovery از آن بهره گرفت. -م-

در این پروتکل و دیگر پروتکل‌های احراز هویت که در ادامه می‌آیند، از نمادهای زیر استفاده شده است:

$A$  و  $B$  مشخصه‌های شناسایی<sup>۱</sup> آلیس و باب هستند.

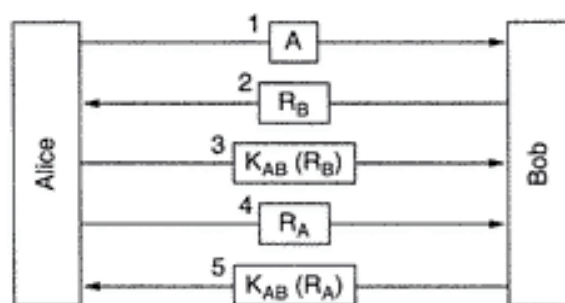
$R_i$  رشته‌های «چالش» (Challenge) هستند که پانویس آنها یعنی  $i$  فرستنده آن را مشخص می‌کند.

$K_i$  کلیدهایی هستند که پانویس آنها یعنی  $i$  صاحب کلید را مشخص می‌نماید.

$K_S$  کلید نشست

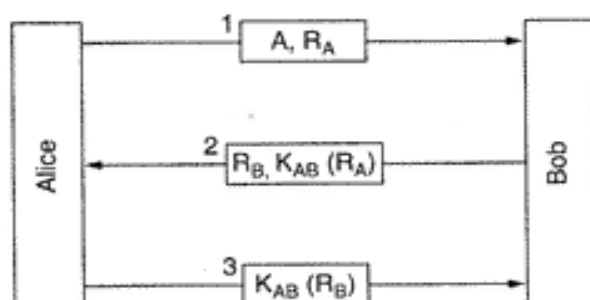
در شکل ۸-۳۲، اولین پروتکل احراز هویت مبتنی بر کلید مشترک و ترتیب مبادله پیامها نشان داده شده است. در پیام ۱، آلیس مشخصه شناسایی خود یعنی  $A$  را به گونه‌ای برای باب می‌فرستد که او بتواند آن را بفهمد (بصورت آشکار و بدون رمزنگاری). البته باب (فعالاً) هیچ راهی برای تشخیص آن که آیا این پیام واقعاً از آلیس آمده یا از شخص ثالثی مثل ترودی، ندارد. به همین دلیل یک عدد تصادفی بسیار بزرگ یعنی  $R_B$  را به عنوان رشته «چالش» (Challenge) انتخاب کرده و آنرا در پیام شماره ۲ بصورت آشکار به آلیس بر می‌گرداند. اعداد تصادفی بکار رفته در پروتکل‌های «چالش-پاسخ» مثل این پروتکل، اصطلاحاً *nonce* نامیده می‌شوند. آلیس پیام شماره ۲ را با کلید مشترک خود رمزنگاری کرده و داده‌های رمز شده یعنی  $K_{AB}(R_B)$  را در پیام شماره ۳ به باب بر می‌گرداند. وقتی باب این پیام را دریافت می‌کند فوراً متوجه می‌شود که این پیام واقعاً از آلیس آمده است زیرا ترودی  $K_{AB}$  را نمی‌داند و طبعاً نمی‌توانسته چنین پیامی را تولید نماید. از آنجایی که  $R_B$  به صورت کاملاً تصادفی و در یک فضای بسیار بزرگ (مثلاً اعداد تصادفی ۱۲۸ بیتی) انتخاب می‌شود لذا احتمال آن که ترودی قبلاً یکبار  $R_B$  و پاسخ آن را مشاهده کرده باشد بسیار بعید است. همچنین تقریباً احتمال آن که او بتواند پاسخ صحیح هر رشته «چالش» را [بدون داشتن کلید] حدس بزند وجود ندارد.

در این جا، باب مطمئن شده که در حال صحبت با آلیس است ولی آلیس از هیچ چیز مطمئن نیست زیرا ممکن است ترودی پیام شماره ۱ را استراق سمع کرده باشد و در پاسخ  $R_B$  را برگرداند. اصلاً ممکن است باب، شب قبل فوت کرده باشد!! برای آن که آلیس بداند که در حال صحبت با چه کسی است عدد تصادفی  $R_A$  را انتخاب و در پیام شماره ۴ آن را برای باب می‌فرستد. وقتی باب پاسخ  $K_{AB}(R_A)$  (یعنی حاصل رمزنگاری  $R_A$  با کلید مشترک) را برگرداند، آلیس نیز متوجه می‌شود که واقعاً با باب صحبت می‌کند. حال اگر این دو بخواهند یک کلید نشست ایجاد کنند، آلیس می‌تواند کلیدی مثل  $K_S$  را انتخاب و آن را با  $K_{AB}$  رمز کرده و برای باب بفرستد.



شکل ۸-۳۲. پروتکل دومرحله‌ای «چالش-پاسخ» جهت احراز هویت.

پروتکل شکل ۸-۳۲ شامل پنج پیام است. حال ببینیم آیا می‌توان با تیزهوشی، تعدادی از این مراحل را حذف کرد. یکی از این راهکارها در شکل ۸-۳۳ نشان داده شده است. در این شکل آلیس به جای آن که منتظر شروع

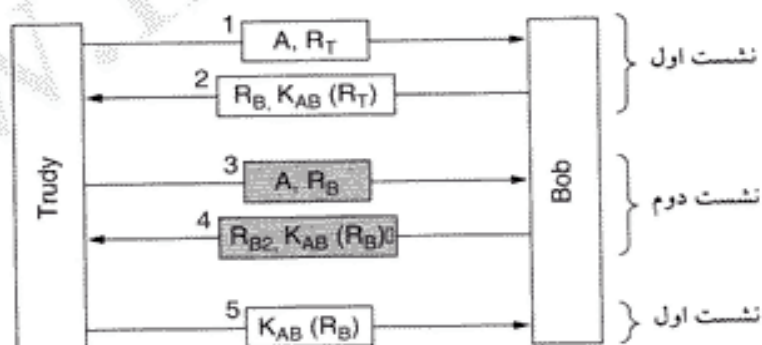


شکل ۸-۳۳. پروتکل دومرحله ای احراز هویت با تعداد مراحل کمتر.

مراحل «چالش-پاسخ» توسط باب شود خودش شخصاً این کار را شروع می کند.<sup>۱</sup> به روش مشابه، باب وقتی به چالش آلیس پاسخ می دهد، رشته چالش خودش یعنی  $R_B$  را نیز برای آلیس می فرستد. بدین ترتیب کل پروتکل به جای پنج مرحله به سه مرحله کاهش می یابد.

آیا پروتکل جدید مزیتی بر پروتکل اصلی دارد؟ به صورت حسی شاید بگوییم کوتاهتر و سریعتر است. متأسفانه اشتباه است! در شرایط خاص ترودی می تواند این پروتکل را با استفاده از تکنیکی نام «حمله بازتاب» (Reflection Attack) در هم بشکند. اگر امکان گشودن چند نشست همزمان با باب وجود داشته باشد، ترودی براحتی خواهد توانست این پروتکل را شکست بدهد. این وضعیت زمانی اتفاق می افتد که به عنوان مثال، باب یک بانک باشد و طبعاً باید بتواند ارتباط همزمان چندین ماشین خودپرداز (Teller Machine) را بپذیرد.

«حمله بازتاب» که توسط ترودی انجام می شود در شکل ۸-۳۴ نشان داده شده است. او حمله خود را با ارسال  $R_T$  و اعلام آن که آلیس است، شروع می کند. باب طبق معمول پاسخ این چالش را به همراه رشته چالش خود یعنی  $R_B$  برمی گرداند. حال ترودی گیر می افتد. او  $K_{AB}(R_B)$  را نمی داند. پس چه کاری می تواند انجام بدهد؟



شکل ۸-۳۴. حمله بازتاب.

او می تواند نشست دومی را با ارسال پیام ۳ شروع نماید و  $R_B$  بدست آمده از مرحله دوم را به عنوان رشته «چالش» خودش برای باب بفرستد. باب در کمال آرامش آن را رمز کرده و به صورت  $K_{AB}(R_B)$  در پیام چهارم برای ترودی می فرستد. در شکل ۸-۳۴، پیامهای نشست دوم را برای تمایز از نشست اول، خاکستری نشان داده ایم. حال ترودی اطلاعاتی را که برای نشست اول کم داشت در اختیار دارد لذا می تواند نشست اول را تکمیل کرده و نشست دوم را ناتمام رها کند. حال باب متقاعد شده که ترودی همان آلیس است لذا وقتی حساب بانکی

۱. یعنی بلافاصله ضمن معرفی خود، رشته چالش یعنی  $R_A$  را برای باب می فرستد.

آلیس را تقاضا می‌کند، باب بی‌هیچ پرسشی اطاعت می‌نماید. وقتی ترویدی مثلاً از باب می‌خواهد که تمام موجودی او را به یک حساب محرمانه در سونیس واریز نماید، باب این کار را بدون اندکی درنگ انجام می‌دهد! پند علمی این داستان آن است که:

«طراحی یک پروتکل صحیح برای احراز هویت دشوارتر از آن است که به نظر می‌رسد.»

چهار قاعده کلی زیر می‌تواند راهنمای خوبی در طراحی پروتکل باشد:

۱. شروع کننده را وادار کنید که قبل از پاسخ‌دهنده، هویت خود را اثبات کند وگرنه باب قبل از آن که ترویدی مدرکی در خصوص هویت خود ارائه داده باشد، اطلاعات با ارزشی را از دست می‌دهد.
۲. شروع کننده و پاسخ‌دهنده را وادار کنید که از کلیدهای متفاوتی برای اثبات هویت خودشان استفاده کنند حتی اگر این کار به معنای تعریف دو کلید مشترک و مستقل  $K_{AB}$  و  $K'_{AB}$  باشد.
۳. شروع کننده و پاسخ‌دهنده را وادار کنید که رشته‌های «چالش» خود را از مجموعه‌های متفاوتی انتخاب نمایند. مثلاً شروع کننده مجبور باشد اعداد زوج را انتخاب کند و پاسخ‌دهنده اعداد فرد را.
۴. پروتکل را در مقابل حملاتی که در اثر نشستهای موازی و همزمان امکان‌پذیر می‌شود، مقاوم کنید زیرا ممکن است اطلاعاتی که از یک نشست بدست می‌آید در دیگری قابل استفاده باشد.

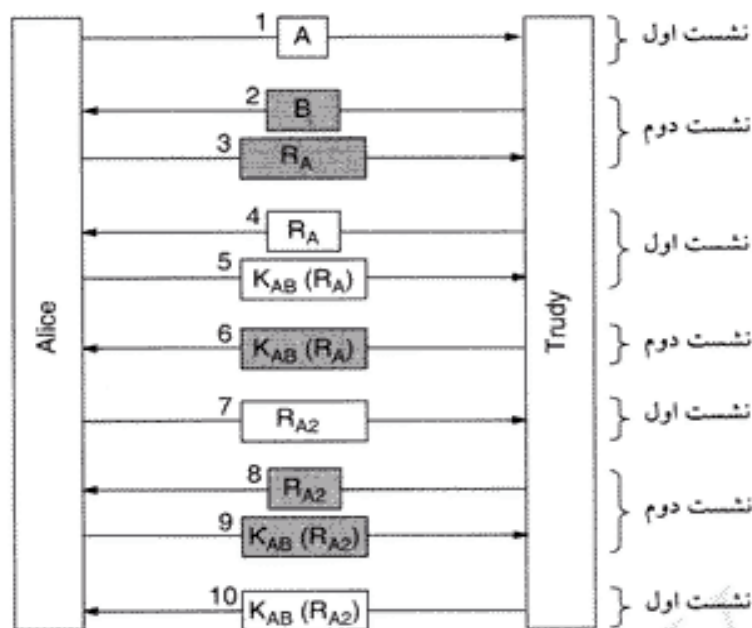
حتی اگر یکی از این چهار قاعده نقض شود، پروتکل به کرات قابل شکستن خواهد بود. در پروتکل سه مرحله‌ای مثال قبل هر چهار قاعده نقض شده بود و بالطبع نتایج خطرناکی به بار خواهد آورد.

حال اجازه بدهید به شکل ۸-۳۲ بازگردیم و نگاهی دقیقتر به آن بیندازیم. آیا این پروتکل مطمئناً در معرض «حمله بازتاب» (Reflection Attack) قرار نمی‌گیرد؟ بستگی دارد! قضیه کمی پیچیده است! ترویدی می‌تواند این پروتکل را با استفاده از حمله بازتاب شکست بدهد زیرا این امکان وجود دارد که او بتواند یک نشست دوم با باب ترتیب داده و او را به نحوی بفریبد تا به پرسشهای مورد نظر او پاسخ گوید. اگر آلیس را یک ماشین خودکار چندمنظوره فرض کنیم نه یک شخص حقیقی، با این ویژگی که بطور همزمان چندین نشست را می‌پذیرد، چه اتفاقی می‌افتد؟ بیایید بررسی کنیم در این وضعیت ترویدی چه می‌تواند بکند.

برای آنکه متوجه شوید که حمله ترویدی به چه نحو خواهد بود به شکل ۸-۳۵ نگاه کنید. آلیس با اعلام مشخصه شناسایی خود برای باب، کارش را شروع می‌کند. ترویدی این پیام را راهزنی و متوقف کرده و با ارسال پیام ۲ خودش را به دروغ باب معرفی و نشست دومی را آغاز می‌کند. باز هم پیامهای نشست دوم به صورت خاکستری نشان داده شده‌اند. آلیس [در پاسخ به پیام ۲] پیامی بدین مضمون می‌فرستد: شما ادعا می‌کنید باب هستید؛ ثابت کنید! در اینجا به نظر می‌رسد که ترویدی گیر افتاده چون نمی‌تواند ثابت کند باب است. ترویدی چه کاری می‌تواند انجام بدهد؟

او به نشست اول بر می‌گردد؛ در آنجا نوبت اوست که رشته «چالش» خود را بفرستد، به همین دلیل  $R_A$  (ارسالی توسط آلیس) را می‌فرستد! آلیس در پاسخ، پیام ۵ را باز می‌گرداند.  $K_{AB}(R_A)$  همان اطلاعاتی است که ترویدی برای ادامه نشست دوم بدان نیاز داشته است، لذا آنرا از طریق نشست دوم برای آلیس می‌فرستد. در اینجا ترویدی در نشست دوم پاسخ موفقیت‌آمیزی را برای آلیس ارسال می‌کند. حال او می‌تواند نشست اول را لغو کرده و باقیمانده مراحل نشست دوم را تکمیل کند و بدین ترتیب یک نشست موفق و مورد تایید با آلیس خواهد داشت. (نشست ۲)

ولیکن ترویدی پلیدتر از این حرفهاست و می‌خواهد مردم آزاری خود را ادامه بدهد!!! به جای آنکه اعدادی قدیمی و بی‌ارزش را برای تکمیل نشست ۲ ارسال کند، منتظر می‌ماند که آلیس در ادامه نشست اول رشته چالش خود یعنی  $R_{A2}$  را برایش بفرستد. اگرچه ترویدی نمی‌داند که در پاسخ به آن چه باید برگرداند ولیکن باز هم از

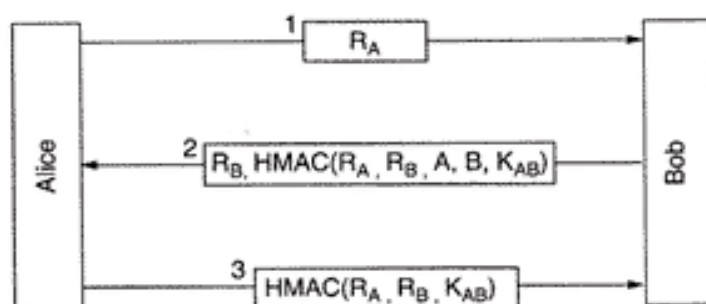


شکل ۸-۳۵. حمله بازتاب بر علیه پروتکل شکل ۸-۳۴.

«حمله بازتاب» بهره می گیرد و  $R_{A2}$  را در پیام هشتم ارسال می نماید. آلیس براحتی آن را رمز کرده و در پیام نهم بر می گرداند. ترودی مجدداً به نشست اول بازگشته و عدد بدست آمده از پیام نهم را در پیام ۱۰ برای آلیس پس می فرستد. در این لحظه ترودی دو نشست آماده و کامل با آلیس ترتیب داده است.

نتیجه این حمله، با حمله به پروتکل سه مرحله ای شکل ۸-۳۴ متفاوت است. در اینجا ترودی دو ارتباط تانید شده و کامل با آلیس دارد در حالی که در مثال قبلی او فقط یک ارتباط تانید شده با باب برقرار کرده بود. در اینجا نیز اگر چهار قاعده اساسی در پروتکل های احراز هویت که قبلاً عنوان کردیم رعایت می شد، چنین حمله ای ممکن نبود. تشریح کامل این نوع از حملات و چگونگی خنثی سازی آنها در مرجع (Bird et.al, 1993) آمده است. آنها نشان داده اند که می توان پروتکل های متفاوتی را ایجاد کرد که صحت عملکردشان قابل اثبات باشد. ساده ترین پروتکل از این گونه، تا حدودی پیچیده است به همین دلیل ما در اینجا رده متفاوتی از پروتکل های احراز هویت را معرفی خواهیم کرد.

پروتکل جدید احراز هویت در شکل ۸-۳۶ نشان داده شده است. (Bird et. al, 1993) در این پروتکل از HMAC<sup>۱</sup> که در توضیح IPsec بدان اشاره کردیم استفاده شده است. آلیس در اولین پیام با ارسال  $R_A$  (به عنوان nonce یا همان رشته چالش) برای باب، کارش را آغاز می کند. باب با انتخاب  $R_B$  برای خود، آن را در قالب رشته درهم شده HMAC برای آلیس پس می فرستد. HMAC به گونه ای سازماندهی شده است که یک ساختمان داده شامل:  $R_A$  (ارسالی توسط آلیس)،  $R_B$  (متعلق به باب و مشخصه های شناسایی هر دو و همچنین کلید مشترک آنها یعنی  $K_{AB}$ ) را در بر می گیرد. این ساختمان داده به روشی مثل SHA-1 درهم شده و در HMAC قرار می گیرد. وقتی آلیس پیام ۲ را دریافت می کند،  $R_A$  (که خودش در ابتدا انتخاب کرده بود)،  $R_B$  که به صورت آشکار برایش ارسال شده بود، دو مشخصه شناسایی (مشخصه خودش و باب) و کلید محرمانه را در اختیار دارد و بدین ترتیب می تواند بطور مستقل HMAC (یعنی رشته Hash) را برای این آیتمها محاسبه نماید. اگر HMAC محاسبه شده با



شکل ۸-۳۶. احراز هویت با استفاده از روش HMAC.

HMAC از سالی توسط باب معادل باشد، آلیس می تواند مطمئن باشد که در حال صحبت با باب است زیرا ترودی  $K_{AB}$  را نمی داند و بالطبع نمی تواند HMAC را به صورت جعلی و دروغین ساخته و آن را از طرف باب ارسال نماید. آلیس نیز در پاسخ، HMAC مربوط به سه آیت  $R_A, R_B, K_{AB}$  را ارسال می کند. (باز هم ترودی نمی تواند این پیام را جعل کند، چون کلید  $K_{AB}$  را در اختیار ندارد).

آیا ترودی می تواند به طریقی این پروتکل را شکست بدهد؟ خیر؛ چون او نمی تواند طبق وقایعی که در شکل ۸-۳۵ اتفاق افتاد، طرفین را وادار کند تا مقادیر دلخواه او را رمزنگاری یا رشته Hash را محاسبه نماید. هر دوی HMACها شامل مقادیری هستند که در کنترل ترودی نیست.

استفاده از HMAC تنها روش بهره گیری از ایده فوق نیست. اغلب به جای محاسبه HMAC از روشی جایگزین استفاده می شود که در آن دنباله آیتها به روش «زنجیره سازی بلوکهای رمز» رمز و ارسال می شوند.

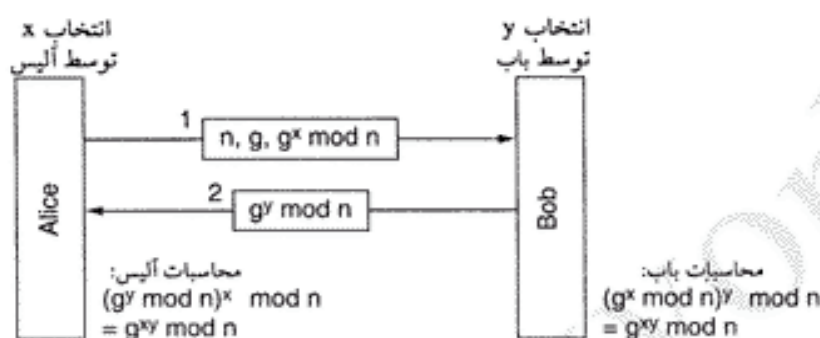
## ۸-۷-۲ ایجاد کلید مشترک: مبادله کلید به روش «دیفی-هلمن»

تا اینجا فرض کرده ایم که آلیس و باب بر روی یک کلید سری توافق کرده اند. حال فرض را بر آن بگذارید که آنها چنین کاری نکرده اند زیرا مثلاً هیچ مرکز جهانی و مورد وفاق PKI برای امضاء و توزیع گواهینامه های دیجیتالی وجود ندارد. در چنین حالتی چگونه این دو نفر می توانند یک کلید مشترک ایجاد کنند. یک راه آن است که آلیس از طریق تلفن با باب تماس گرفته و کلید مورد نظر خود را به او بگوید ولی ممکن است در همان ابتدای کار باب از آلیس سؤال کند که: «از کجا بدانم شما آلیس هستی و ترودی نیستی؟» آنها می توانند یک ملاقات ترتیب داده و هر کدام، گذرنامه یا گواهینامه رانندگی خود و سه کارت اعتباری معتبر و مهم را همراه آورده و پس از تأیید هویت یکدیگر، بر روی یک کلید مشترک توافق نمایند. برای مردم گرفتار، شاید تنظیم قرار ملاقات برای ماهها امکان پذیر نباشد. خوشبختانه راه حلی برای توافق و ایجاد کلید سری بین افراد ناآشنا با یکدیگر وجود دارد که آنها می توانند در روز روشن و حتی ترودی قادر به استراق سمع و ضبط تمام پیامهاست، یک کلید سری ایجاد کنند، هر چند این موضوع ممکن است اندکی غیرقابل باور به نظر برسد.

پروتکلی که به افراد غریبه و ناآشنا با یکدیگر، اجازه می دهد یک کلید مشترک و سری ایجاد کنند، پروتکل «مبادله کلید دیفی - هلمن»<sup>۱</sup> نام دارد و به صورت زیر عمل می کند: آلیس و باب می باید بر روی دو عدد بسیار بزرگ  $n$  و  $g$  توافق کنند که از این دو  $n$  عددی اول است. همچنین  $(n-1)/2$  نیز عددی اول است و شرایط خاصی نیز بر روی  $g$  اعمال می شود. این دو عدد عمومی و غیر سری بوده و هر کسی می تواند آزادانه یک  $n$  و  $g$  انتخاب کرده و به دیگری اعلام کند. در اینجا آلیس یک عدد بزرگ  $x$  (مثلاً ۵۱۲ بیتی) انتخاب کرده و آن را به صورت سری نزد

۱. Diffie-Hellman key exchange; (Diffie and Hellman, 1976)

خود نگاه می دارد. به همین روش باب نیز یک عدد سری مثل  $y$  برای خود انتخاب می کند. آلیس، طبق شکل ۸-۳۷، پروتکل مبادله کلید را با ارسال پیامی شامل آیتمهای  $g$  و  $n$  و  $g^x \bmod n$  آغاز می کند. باب نیز در پاسخ، پیامی را ارسال می کند که در برگیرنده  $g^y \bmod n$  است. حال آلیس، عدد ارسالی توسط باب را در پیمانه  $n$  به توان  $x$  می رساند تا حاصل  $(g^y \bmod n)^x \bmod n$  به دست آید. باب نیز همین کار را به صورت  $(g^x \bmod n)^y \bmod n$  انجام می دهد. طبق روابط حاکم بر نظریه اعداد، هر دو نفر حاصل  $g^{xy} \bmod n$  را بدست خواهند آورد. دقت کنید! آلیس و باب به ناگاه صاحب یک کلید مشترک و سری با فرمول  $g^{xy} \bmod n$  شده اند.

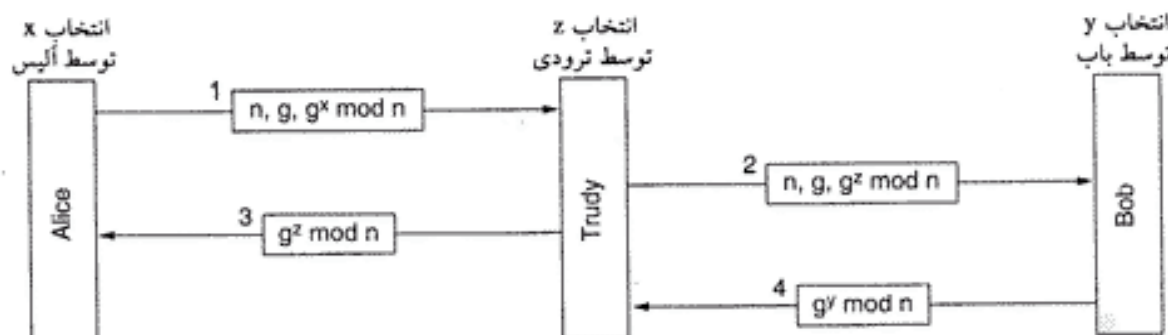


شکل ۸-۳۷. مبادله کلید بروش دیفی-هلمن.

البته در این میان ترویدی هر دو پیام را می بیند بنابراین  $g$  و  $n$  را از پیام اول در اختیار دارد. اگر او بتواند  $x$  و  $y$  را محاسبه کند قادر خواهد بود تا کلید سری را تعیین نماید ولیکن مشکل اینجاست که با داشتن  $g^x \bmod n$  نمی توان  $x$  را پیدا کرد زیرا هیچ الگوریتم عملی برای محاسبه لگاریتم گسسته در پیمانه اعداد اول بسیار بزرگ، تاکنون کشف نشده است.

برای آن که مثال بالا را بهتر تفهیم کنیم از اعداد  $n=47$  و  $g=3$  استفاده می نمایم (این اعداد در عمل بسیار کوچک و غیر قابل استفاده اند). آلیس عدد  $x=8$  و باب عدد  $y=10$  را برای خود انتخاب می کند؛ هر دوی این اعداد سری نگه داشته می شوند. پیام آلیس به باب شامل آیتمهای (۲۸ و ۳ و ۴۷) خواهد بود چرا که حاصل  $3^8 \bmod 47$  معادل ۲۸ است. پیام باب به آلیس عدد ۱۷ است. آلیس مقدار  $17^8 \bmod 47$  را محاسبه می کند که ۴ بدست می آید. باب نیز حاصل  $28^{10} \bmod 47$  را بدست می آورد که باز هم ۴ است. در اینجا باب و آلیس هر دو کلید سری و مشترک ۴ را دارند که به صورت مستقل بدست آمده است. برای بدست آوردن کلید سری، ترویدی مجبور است معادله  $3^x \bmod 47 = 28$  را حل کند که فقط با جستجوی کامل [در محدوده صفر تا  $n-1$ ] امکان پذیر است و اگر  $n$  عددی بسیار بزرگ در حد چند صد بیت باشد، حل این معادله عملاً ممکن نخواهد بود. حتی اگر از ابر کامپیوترهای موازی استفاده شده باشد، تمام الگوریتمهای شناخته شده امروزی برای حل این معادله، نیاز به وقت بسیار زیادی دارند.

علیرغم زیبایی الگوریتم «دیفی - هلمن»، مشکلی در آن وجود دارد: وقتی باب سه آیت (۲۸ و ۳ و ۴۷) را دریافت می کند از کجا بداند که واقعاً از طرف آلیس پیشنهاد شده است نه از طرف ترویدی؟ هیچ راهی برای این تشخیص وجود ندارد! متأسفانه ترویدی می تواند به گونه ای که در شکل ۸-۳۸ نشان داده شده بطور همزمان آلیس و باب را فریب بدهد. در اینجا وقتی آلیس و باب اعداد  $x$  و  $y$  را برای خود انتخاب می کنند، ترویدی نیز  $z$  را برای خود بر می گزیند. حال آلیس پیام اول را برای باب می فرستد ولیکن این پیام در میانه راه توسط ترویدی دریافت و متوقف می شود. ترویدی  $g$  و  $n$  (که عمومی و غیر سری هستند) را به همراه  $g^z \bmod n$  (به جای  $g^x \bmod n$ ) برای



شکل ۸-۳۸. حمله «گروه آتش‌نشان» (یا حمله Man-In-The-Middle).

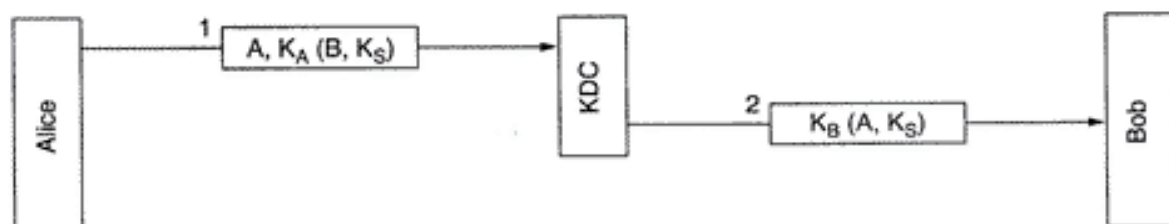
باب می‌فرستد. همچنین پیام سوم را به صورت  $g^z \bmod n$  برای آلیس بر می‌گرداند. بعداً باب پیام چهارم یعنی  $g^y \bmod n$  را برای آلیس می‌فرستد که آن هم توسط ترودی دریافت و حفظ می‌شود. حال هر سه نفر محاسبات پیمانه‌ای (Modular Arithmetic) خود را آغاز می‌کنند. آلیس و ترودی کلید سری  $g^{xz} \bmod n$  را محاسبه می‌کنند. از طرف دیگر باب و ترودی نیز  $g^{yz} \bmod n$  را به عنوان کلید سری خود محاسبه می‌نمایند. آلیس در این خیال است که با باب صحبت می‌کند لذا یک کلید نشست با ترودی ایجاد کرده است. باب نیز به همین گونه فریب می‌خورد. هر پیامی که به صورت رمزنگاری شده توسط آلیس ارسال گردد ابتدا توسط ترودی دریافت، ذخیره و در صورت تمایل دستکاری شده و سپس برای باب ارسال می‌شود. در طرف مقابل نیز همین اتفاق می‌افتد. باب و آلیس با این تصور نادرست که کانالی امن ایجاد کرده‌اند، با یکدیگر تبادل اطلاعات می‌کنند در حالی که ترودی تمام پیامهای آنها را می‌بیند و احیاناً آنها را به میل خود تغییر می‌دهد. این حمله اصطلاحاً به نام «حمله گروه آتش‌نشان» (Bucket Brigade Attack) مشهور است زیرا تقریباً به انتقال سطلهای آب از کامیون به محل آتش‌سوزی شباهت دارد که آتش‌نشانهایی داوطلب در یک صف، سطلهای آب را دست به دست هدایت می‌کنند. این حمله همچنین به نام حمله  $mitm$  (Man in The Middle Attack) شهرت دارد.

### ۸-۷-۳ احراز هویت توسط مرکز توزیع کلید (KDC)

توافق و تنظیم یک کلید مشترک و سری با افراد غریبه به روش فوق تقریباً عملی است ولیکن کامل و بی‌نقص نیست. حتی شاید (بدلیل ضعف امنیتی اشاره شده در بخش قبل و اشکالی که در ادامه بدان می‌پردازیم)، ارزش اجرایی نداشته باشد. شما برای آن که بتوانید با  $n$  نفر محاوره داشته باشید، طبعاً به  $n$  عدد کلید احتیاج خواهید داشت. برای افراد عادی، نگهداری و مدیریت کلیدها واقعاً سخت و پرمخاطره است، بالاخص اگر لازم باشد کلیدها بر روی یک کارت پلاستیکی ذخیره و تحویل شود.

راهکار متفاوتی که وجود دارد آنست که یک «مرکز توزیع کلید» مورد اعتماد و وفای عموم (KDC)، معرفی شود. در این ساختار هر کاربر تنها یک کلید دارد که بین او و KDC مشترک است. احراز هویت و ایجاد کلید نشست، از طریق واسطه KDC انجام می‌شود. در شکل ۸-۳۹ ساده‌ترین پروتکل احراز هویت مبتنی بر KDC نشان داده شده که شامل یک مرکز معتدله توزیع کلید و طرفین ارتباط است.

مبنای نظری این پروتکل بسیار ساده است: آلیس یک کلید نشست  $K_S$  را انتخاب کرده و به KDC اعلام می‌کند که تمایل دارد با استفاده از این کلید با باب محاوره نماید. این پیام با استفاده از کلید سری و مشترک بین آلیس و KDC که آن را  $K_A$  نامیده‌ایم رمز می‌شود. KDC این پیام را رمزگشایی کرده و مشخصه شناسایی باب و کلید نشست را از درون آن استخراج می‌نماید. سپس KDC، پیام جدیدی را می‌سازد و در آن مشخصه شناسایی آلیس و کلید نشست را قرار داده و پس از رمزنگاری برای باب می‌فرستد. رمزنگاری این پیام با کلید  $K_B$  یعنی کلید



شکل ۸-۳۹. اولین پروتکل احراز هویت یکمک KDC.

مشترک بین باب و KDC انجام می شود. وقتی باب این پیام را رمزگشایی کند، متوجه می شود که آلیس تمایل به محاوره با او دارد و کلید رمزی را که برای محاوره باید استفاده شود، بدست می آورد.

این روش احراز هویت بسیار ساده انجام می شود: KDC می داند که پیام قاعداً از طرف آلیس آمده است چراکه هیچکسی قادر به رمزنگاری این پیام با کلید سرّی آلیس نیست؛ (چون کلید آلیس را ندارد). به روش مشابه، باب نیز می داند که پیام ۲ از طرف KDC که مورد اعتماد اوست صادر شده چون هیچکس کلید سرّی او را در اختیار ندارد (مگر KDC).

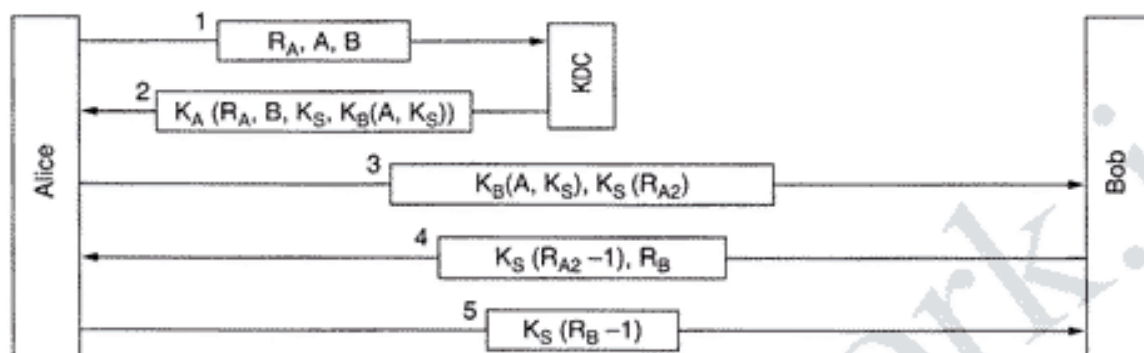
متأسفانه این پروتکل دارای یک اشکال جدّی است. به سناریوی زیر دقت کنید. ترویدی به مقداری پول احتیاج دارد لذا به گونه ای برنامه ریزی می کند که خدماتی متعارف برای آلیس انجام بدهد و سپس به استخدام موقت او در می آید. پس از انجام کار، ترویدی مؤدبانه از آلیس خواهش می کند تا حقوقش را به حساب بانکی او واریز نماید. آلیس با بانکدار خود یعنی باب، یک کلید نشست ایجاد کرده و سپس از باب تقاضا می کند که پول درخواستی را از حساب او به حساب ترویدی واریز نماید.

در این میان، ترویدی به راه و روش قبلی خود یعنی جاسوسی و پرسه زدن در شبکه برمی گردد. او پیام شماره ۲ در شکل ۸-۳۹ و همچنین پیام تقاضای انتقال پول [که در شکل نیامده] را استراق سمع نموده و در جایی کپی می کند. ترویدی بعداً این دو پیام را بار دیگر برای باب تکرار می کند. باب این پیامها را دریافت کرده و با خود می اندیشد که آلیس ترویدی را برای بار دیگر به استخدام خود در آورده است لذا همان مقدار پول را از حساب بانکی آلیس به حساب ترویدی واریز می کند. پس از دریافت پنجاهمین پیام، باب از دفترش بیرون می آید تا ترویدی را پیدا کرده و برای توسعه کار تجارت به او پیشنهاد وام بدهد!!! به این نوع حمله اصطلاحاً «حمله تکرار» (Replay Attack) گفته می شود.

چندین راه حل برای جلوگیری از حمله تکرار وجود دارد. اولین راه حل آن است که هر پیام دارای «مهر زمان» (Timestamp) باشد. در این صورت هرگاه کسی یک پیام قدیمی دریافت کرد می تواند براحتی آنرا حذف کند. یکی از مشکلات این روش آن است که نمی توان در یک شبکه، همه ساعتها را بدقت با هم تنظیم کرد لذا باید به گونه ای برنامه ریزی نمود که مهر زمان در یک بازه مشخص [مثلاً پانزده دقیقه] اعتبار داشته باشد و بدین ترتیب ترویدی در این بازه زمانی مهلت خواهد داشت تا پیامهای مورد نظر خود را تکرار کرده و به منظور خود برسد.

راه حل دوم آن است که در هر پیام یک عدد یا رشته تصادفی چالش (nonce) گذاشته شود و پیامهایی که این عدد یا رشته در آنها تکراری است حذف شوند. مشکل این روش آن است که تمام این اعداد یا رشته ها باید برای همیشه نگهداری شوند مبادا ترویدی پیامی مربوط به پنج سال قبل را به صورت تکراری بفرستد. همچنین اگر ماشینی درهم بشکند و فهرست این اعداد یا رشته ها از دست برود در معرض «حمله تکرار» قرار خواهد گرفت. می توان از ترکیب «مهر زمان» و «الصاق عدد یا رشته تصادفی چالش» (nonce) در پیامها استفاده کرد. بدین ترتیب طول مدتی که باید این اعداد و رشته ها حفظ شوند کاهش پیدا می کند ولیکن پروتکل پیچیده تر خواهد شد.

راهکاری پیچیده‌تر برای آنکه طرفین بتوانند خودشان به کمک KDC، یکدیگر را احراز هویت کنند آنست که از یک پروتکل «چالش-پاسخ» (Challenge/Response) چند مرحله‌ای استفاده شود. مثالی از این نوع، «پروتکل احراز هویت نیدهام-شرودر» (Needham-Schroeder, 1978) است که گونه‌ای از آنرا در شکل ۸-۴۰ می‌بینید.



شکل ۸-۴۰. پروتکل احراز هویت «نیدهام-شرودر».

این پروتکل بدین نحو آغاز می‌شود که آلیس به KDC اعلام می‌کند تمایل به محاوره با باب دارد. این پیام شامل یک عدد تصادفی بزرگ  $R_A$  است. KDC پیام ۲ را بر می‌گرداند که حاوی: عدد تصادفی آلیس (یعنی  $R_A$ )، کلید نشست (یعنی  $K_S$ )، مشخصه شناسایی باب (یعنی  $B$ ) و یک «بلیط» (یعنی  $K_B(A, K_S)$ ) است؛ این بلیط باید دست نخورده برای باب ارسال شود. هدف از  $R_A$  آن است که از غیرتکراری و غیرجعلی بودن پیام ۲ اطمینان حاصل شود. مشخصه شناسایی باب بدان جهت درون این پیام جاسازی شده که اگر ترویدی در میانه راه، پیام شماره یک را دستکاری نموده و مشخصه خود را با مشخصه باب  $B$  عوض کرده باشد، قضیه آشکار شود چراکه در غیر این صورت KDC بلیط را به جای  $K_B$  با  $K_T$  (کلید ترویدی) رمزنگاری کرده و بر می‌گرداند. «بلیط» ابتدا با کلید  $K_B$  رمزنگاری شده و سپس درون پیامی قرار می‌گیرد که بار دیگر با کلید  $K_A$  رمز خواهد شد تا ترویدی بهیچوجه نتواند محتوای این پیام (پیام ۲) را دستکاری کرده و به آلیس تحویل بدهد.

حال آلیس بلیط خود  $(K_B(A, K_S))$  را برای باب می‌فرستد و به همراه آن یک عدد تصادفی بزرگ  $R_{A2}$  را نیز با کلید نشست  $K_S$  رمز کرده و ارسال می‌کند. در پیام چهارم باب حاصل رمزنگاری  $K_S(R_{A2}-1)$  را برای آلیس باز پس می‌فرستد تا ثابت کند که آلیس واقعاً با باب صحبت می‌کند. برگرداندن  $K_S(R_{A2})$  عملی نخواهد بود زیرا ترویدی می‌تواند آن را از پیام سوم استراق سمع کند و به صورت جعلی به آلیس بازگرداند.

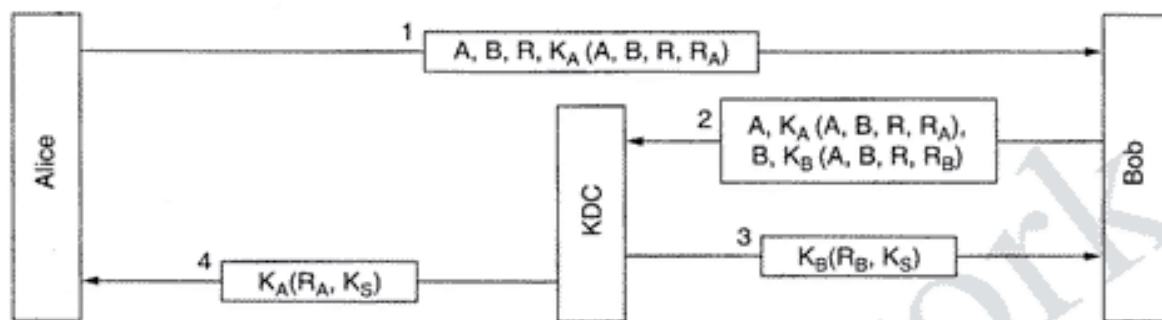
پس از دریافت پیام چهارم آلیس متقاعد می‌شود که با باب واقعی صحبت می‌کند و هیچ پیام جعلی و تکراری دریافت نکرده چراکه او  $R_{A2}$  را در چند میلی‌ثانیه قبل و به صورت تصادفی تولید نموده است. مقصود از ارسال پیام پنجم آن است که آلیس، باب را متقاعد کند که واقعاً آلیس است و پیام سوم، جعلی و تکراری نبوده است. چون هر یک از طرفین رشته‌های «چالش» (Challenge) و «پاسخ» (Response) را مبادله می‌کنند لذا امکان «حمله تکرار» (Replay) متنفی است. [در این ساختار، KDC نیز نقش کوتاه ولی بسیار موثری ایفا می‌کند].

اگرچه این پروتکل بسیار محکم به نظر می‌رسد ولیکن یک ضعف بسیار جزیی دارد. اگر ترویدی بتواند با برنامه‌ریزی یک کلید نشست قدیمی که در گذشته از آن استفاده شده را بنحوی بدست بیاورد<sup>۱</sup> می‌تواند به صورت

۱. منظور از بدست آوردن کلید نشست قدیمی، شکستن رمز نیست بلکه سرقت کلیدی است که مثلاً بدلیل قدیمی بودن حذف نشده و به نحوی لو رفته است. -م.

جعلی از مرحله سوم شروع کرده و نشست جدیدی را با باب شروع کند و او را متقاعد نماید که آلیس است. در اینجا ترویدی می تواند بدون هیچ زحمت اضافی، مثلاً حساب بانکی آلیس را چپاول کند.

برای رفع این اشکال «نیدهام و شرودر» پروتکلی را تدوین کردند. (۱۹۸۷) جالب آن که در همان شماره از مجله علمی که آنها پروتکل خود را به چاپ رساندند دو نفر دیگر به نامهای «اتوی» و «ریس» (Otway & Rees) نیز پروتکلی را تدوین و عرضه کرده بودند که این مشکل را به روشی ساده تر و کوتاه تر حل می کرد. شکل ۸-۴۱ پروتکل «اتوی-ریس» را با تغییر جزئی نشان می دهد.



شکل ۸-۴۱. پروتکل احراز هویت «اتوی-ریس».

در پروتکل اتوی-ریس، آلیس کار خود را با تولید دو عدد تصادفی بزرگ آغاز می کند:  $R$  که به عنوان یک شناسه مشترک بکار می رود و  $R_A$  که آلیس از آن به عنوان رشته «چالش» (Challenge) استفاده می نماید. وقتی باب پیام اول را [طبق شکل] دریافت می کند، با استفاده از بخش رمزنگاری شده پیام آلیس، پیام جدیدی را ساخته و برای KDC می فرستد. در این پیام، باب آیتمی مشابه با بخش رمزنگاری شده پیام اول تولید و به آن می افزاید.  $K_A^1$  و  $K_B$  کلیدهای محرمانه آلیس و باب هستند و آیتمهای  $A$  و  $B$  و  $R$  و  $R_A$  یا  $R_B$  با آنها رمز می شوند.

حال از آنجایی که KDC کلید رمز هر دو نفر را در اختیار دارد، با دریافت این پیام ابتدا بررسی می کند که آیا  $R$  بدست آمده از دو پیام رمزنگاری شده یکی است یا خیر، زیرا ممکن است به دلیل دستکاری در پیام اول توسط ترویدی، با هم مغایرت داشته باشند. اگر هر دو  $R$  با هم یکی بودند KDC متقاعد می شود که پیام ارسالی از باب [که در حقیقت تقاضای احراز هویت به حساب می آید] معتبر است. سپس برای هر دو نفر یک کلید نشست تولید نموده و آن را یکبار به همراه  $R_A$ ، با کلید آلیس رمز کرده و برای آلیس می فرستد و بار دیگر به همراه  $R_B$  با کلید باب رمز کرده و برای باب ارسال می دارد. هر کدام از این پیامها (پیام ۳ و ۴) شامل عدد تصادفی تولید شده توسط گیرنده آن پیام است که ثابت می کند واقعاً KDC آن را تولید کرده و به صورت جعلی یا تکراری توسط ترویدی ارسال نشده است. در این لحظه آلیس و باب دارای کلید مشترکی هستند که توسط KDC پیشنهاد شده و می توانند با اطمینان محاوره خود را آغاز نمایند. در اولین پیامی که بین آنها رد و بدل می شود آنها می توانند بررسی کنند که آیا دیگری  $K_S$  مشابهی دارد یا خیر. [چراکه در غیر این صورت پیامهای ارسالی از رمز خارج نخواهد شد.] بدین ترتیب مراحل احراز هویت تکمیل می شود.

#### ۸-۷-۴ احراز هویت با استفاده از Kerberos

یک پروتکل احراز هویت که در بسیاری از سیستمهای واقعی (مثل ویندوز ۲۰۰۰) بکار گرفته می شود، Kerberos است که براساس گونه ای از پروتکل «نیدهام - شرودر» بنا نهاده شده است. نام Kerberos برگرفته از یک اسطوره

۱. یعنی پیام دوم شامل  $A$  و  $K_A(A, B, R, R_A)$  است که توسط آلیس ارسال شده و در آن دخل و تصرفی نمی شود و همچنین شامل  $K_B(A, B, R, R_B)$  و  $B$  است که خودش آنرا می سازد. -م.

یونانی است که در آن یک سگ چند سر از درب دوزخ نگهبانی می‌کند (که مثلاً نگذارد کسی از آن خارج شود)!! Kerberos در دانشگاه MIT طراحی شده و به کاربران اجازه دسترسی مطمئن به منابع شبکه را می‌دهد. مهمترین تفاوت آن با پروتکل «نیدهام - شرودر» آن است که فرض شده ساعت تمام ایستگاههای شبکه دقیقاً با هم تنظیم شده‌اند. این پروتکل مراحل پیاده‌سازی متعددی را پشت سر گذاشته است. نسخه V4 آن کاربرد بسیار گسترده‌ای در صنعت دارد به همین دلیل، آن را توضیح می‌دهیم. سپس چند کلمه‌ای در خصوص نسخه بعدی آن V5 سخن خواهیم گفت. برای اطلاعات بیشتر به مرجع (Steiner et. al.; 1988) مراجعه کنید.

Kerberos به غیر از آلیس به عنوان ماشین مشتری، با سه ماشین سرویس دهنده دیگر سر و کار دارد:

۱. سرویس دهنده احراز هویت یا AS (Authentication Server): این سرویس دهنده کاربران را در حین ورود به سیستم (Login) بازرسی می‌نماید.

۲. سرویس دهنده صدور بلیط یا TGS (Ticket Granting Server): این سرویس دهنده بلیطهای تایید هویت صادر می‌کند.

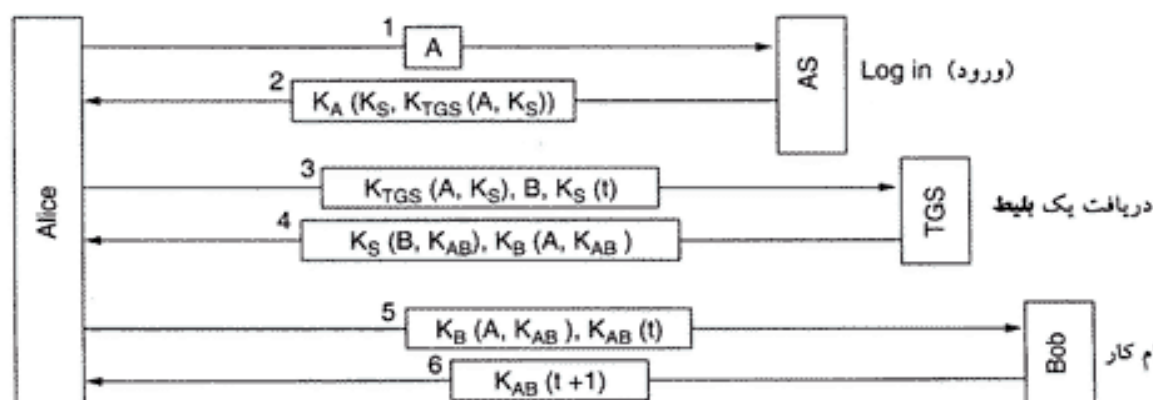
۳. سرویس دهنده باب (یا هر سرویس دهنده‌ای که خدماتی را ارائه می‌کند): این سرویس دهنده کاری را که آلیس از او می‌خواهد انجام می‌دهد.

سرویس دهنده AS شبیه به KDC عمل می‌کند که در آن کلیدهای سری تمام کاربران شبکه نگهداری می‌شود. کار سرویس دهنده TGS آن است که بلیطهای معتبری را برای کاربران صادر کند تا سرویس دهنده‌های واقعی در شبکه متقاعد شوند که حامل بلیط، واقعاً همان کسی است که ادعا می‌کند.

برای شروع یک نشست، آلیس در جلوی هر یک از ایستگاههای عمومی شبکه نشسته و نام خود را درج (تایپ) می‌کند. ایستگاه (Workstation) نام او را به صورت آشکار برای سرویس دهنده AS می‌فرستد. مراحل احراز هویت در شکل ۸-۴۲ نشان داده شده است. آنچه که در پاسخ باز برمی‌گردد یک «کلید نشست» (یعنی  $K_s$ ) و یک «بلیط» یعنی  $K_{TGS}(A, K_s)$  است که باید بعداً به سرویس دهنده TGS تحویل داده شود. این دو آیتم درون یک پیام جاسازی شده و با استفاده از کلید سری آلیس رمز می‌شود فلذا هیچکس جز آلیس نمی‌تواند آن را رمزگشایی کند. وقتی پیام ۲ دریافت شود، ایستگاه از آلیس کلمه عبور او را سؤال می‌کند. سپس از کلمه عبور، کلید رمز آلیس یعنی  $K_A$  تولید می‌شود تا پیام ۲ رمزگشایی شده و کلید نشست و بلیط TGS از درون آن استخراج شود. در این لحظه ایستگاه فوراً کلمه عبور آلیس را از حافظه پاک می‌کند تا بیش از چند میلی ثانیه در حافظه ایستگاه باقی نماند. اگر ترویدی سعی کند با نام آلیس وارد شود (Login کند)، کلمه عبوری که درج می‌کند غلط خواهد بود و ایستگاه این موضوع را براحتی کشف خواهد کرد چراکه بدنه اصلی پیام فقط با کلید آلیس از رمز خارج خواهد شد و بدون این کلید پیام دوم غیرقابل استفاده و نامفهوم خواهد بود.

پس از آن که آلیس به شبکه وارد شد ممکن است بخواهد با سرویس دهنده فایل باب، ارتباط برقرار کند. در اینجا ایستگاه پیام ۳ را به سرویس دهنده TGS می‌فرستد و از او می‌خواهد تابلیتی برای استفاده از سرویس دهنده باب صادر نماید. نکته اساسی در این تقاضا  $K_{TGS}(A, K_s)$  است که با کلید سری سرویس دهنده TGS رمزنگاری شده و محتوای آن ثابت می‌کند که فرستنده بلیط واقعاً آلیس است. سرویس دهنده TGS با ایجاد یک کلید نشست  $K_{AB}$  برای آلیس، به او امکان استفاده از آن را در محاوره با باب می‌دهد. دو نسخه از این کلید برگشت داده می‌شود: اولی با کلید نشست  $K_s$  رمز شده و آلیس می‌تواند آن را استخراج کرده و بخواند. نسخه دوم با کلید سری باب یعنی  $K_B$  رمز می‌شود که هیچکس جز باب نمی‌تواند آن را بخواند.

ترویدی می‌تواند پیام ۳ را دریافت و کپی کند و تلاش نماید تا آن را مجدداً یکبار بگیرد ولیکن تلاش او با وجود مهر زمان ۱ که رمزنگاری نیز شده است، بی‌ثمر خواهد ماند. ترویدی نمی‌تواند مهر زمان درج شده در پیام ۳ را



شکل ۸-۴۲. عملکرد سیستم Kerberos V4.

عوض کرده و زمان جدیدی در آن درج نماید زیرا از کلید نشست  $K_S$  که آلیس بکمک آن با TGS محاوره می کند، بی اطلاع است. حتی اگر ترویدی بلافاصله پیام ۳ را استراق سمع و تکرار کند، کپی پیام چهارم را بدست خواهد آورد که قادر نخواهد بود آن را فوراً رمزگشایی کند. (نه قسمت اول آن را که با  $K_S$  رمز شده و نه قسمت دوم را که با  $K_B$  رمز شده است).

حال آلیس می تواند  $K_{AB}$  را برای باب فرستاده و یک نشست با او ترتیب بدهد. این مرحله نیز به همراه مهر زمان (Timestamp) خواهد بود. پاسخی که آلیس از باب دریافت می کند ثابت خواهد کرد که واقعاً با باب محاوره می کند نه ترویدی. [بدین نحو هویت باب تأیید می شود].

پس از این چند مرحله مبادله پیام (جمعاً ۶ پیام) آلیس می تواند در پوشش کلید رمز  $K_{AB}$  با باب تبادل اطلاعات داشته باشد. هر گاه او تصمیم بگیرد که با یک سرویس دهنده دیگر محاوره نماید (مثلاً سرویس دهنده کارول) از پیام سوم شروع می کند با این تفاوت که به جای مشخصه شناسایی B در این پیام، C را درج می کند. TGS سریعاً برای او بلیطی صادر می کند که به جای  $K_B$  با  $K_C$  رمز شده است و آلیس می تواند آن را برای کارول بفرستد و کارول نیز آن را به عنوان یک سند معتبر که از طرف آلیس ارسال شده می پذیرد.

نکته ارزشمند در این ساختار آنست که آلیس می تواند به تمام سرویس دهنده های موجود در شبکه به روشی مطمئن دسترسی داشته باشد و به هیچ وجه کلمه عبور او بر روی شبکه منتقل نخواهد شد. در حقیقت کلمه عبور او فقط برای چند میلی ثانیه درون ایستگاهی که او در کنار آن قرار دارد، ذخیره می شود. به این نکته دقت کنید که هر سرویس دهنده بعداً احراز هویت خاص خود را انجام می دهد یعنی وقتی آلیس بلیط خود را به سرویس دهنده باب عرضه می کند، تنها کاری که این بلیط انجام می دهد آن است که برای باب ثابت کند که او واقعاً آلیس است. حال کارهایی که آلیس مجاز به انجام آن است صرفاً توسط سرویس دهنده باب تعیین می شود.

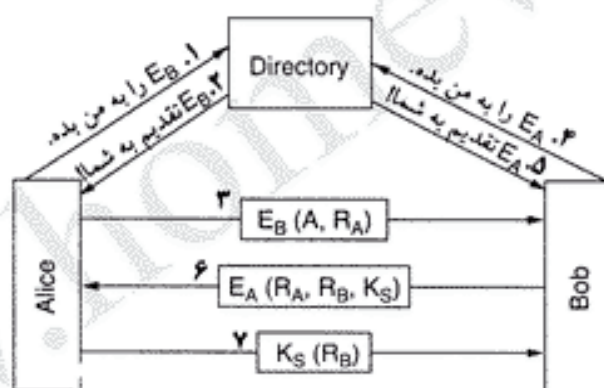
از آنجایی که طراحان Kerberos توقع نداشتند که کل دنیا به یک سرویس دهنده واحد و جهانی احراز هویت اعتماد کنند، لذا این امکان را فراهم کردند که تعدادی ناحیه مستقل وجود داشته باشد و هر ناحیه از یک AS و TGS خاص خود استفاده کند. آلیس برای آن که بتواند بلیطی را برای یک سرویس دهنده در ناحیه ای دور به دست بیاورد، از سرویس دهنده TGS خودش می خواهد که بلیطی برایش صادر کند که مورد پذیرش TGS ناحیه راه دور باشد. اگر TGS ناحیه دور در TGS محلی ثبت شده باشد (به همان روشی که سرویس دهنده های محلی ثبت می شوند)، TGS محلی به آلیس بلیطی می دهد که نزد TGS دیگر معتبر است. او می تواند با این بلیط کار دلخواهش را در آن TGS انجام بدهد مثلاً می تواند از آن TGS بلیط دسترسی به یک سرویس دهنده دیگر در آن ناحیه را دریافت نماید. بهر حال برای آن که طرفین در دو ناحیه مختلف بتوانند کارشان را انجام بدهند باید به TGS

یکدیگر اعتماد داشته باشند.

نسخه V5 از Kerberos، مفصلتر از نسخه V4 است و زحمت و سربار زیادی دارد. همچنین برای توصیف انواع داده (Data Types) از استاندارد OSI ASN.1 (Abstract Syntax Notation 1) استفاده کرده است و در ساختار پروتکل نیز تغییرات اندکی ایجاد شده است. به علاوه، طول عمر بلیطها زیاده‌تر شده و اجازه داده شده بلیطها تجدید شوند. به علاوه پروتکل V5، لااقل در تئوری به DES وابسته نیست (در حالی که V4 این‌گونه است) و از وجود نواحی مختلف و تفویض صدور کلید به سرویس‌دهنده‌های این نواحی حمایت می‌نماید.

### ۵-۷-۸ احراز هویت با استفاده از رمزنگاری با کلید عمومی

می‌توان عملیات احراز هویت را با استفاده از رمزنگاری با کلید عمومی انجام داد. برای شروع، آلیس نیاز دارد که کلید عمومی باب را بدست بیاورد. اگر PKI<sup>۱</sup> (مرکز توزیع کلید عمومی) با ساختار «سرویس‌دهنده دایرکتوری» در جایی از شبکه موجود باشد و گواهی‌نامه‌های کلید عمومی را تحویل بدهد، آلیس می‌تواند به نحوی که در پیام ۱ از شکل ۸-۴۳ نشان داده شده است، از این سرویس‌دهنده در مورد باب سؤال نماید. در پاسخ، پیام ۲ که حاوی گواهی‌نامه X.509 و کلید عمومی باب است، باز می‌گردد. پس از آن که آلیس صحت امضای باب را بررسی و تأیید کرد، برای او پیامی می‌فرستد که در آن مشخصه شناسایی خودش و یک عدد تصادفی ( $R_A$  یا Nonce) با کلید عمومی باب رمز شده است.



شکل ۸-۴۳. احراز هویت متقابل با استفاده از رمزنگاری کلید عمومی.

وقتی باب این پیام را دریافت می‌کند، نمی‌داند که آیا این پیام واقعاً از طرف آلیس ارسال شده یا ترویدی، ولی به همان روش عمل کرده و از «سرویس‌دهنده دایرکتوری» در مورد کلید عمومی آلیس سؤال می‌کند (در پیام ۴) و پاسخ آن را (در پیام ۵) به دست می‌آورد. سپس در پیام ششم  $R_A$  متعلق به آلیس، عدد تصادفی خودش ( $R_B$ ) و یک کلید نشست  $K_S$  پیشنهادی را برای آلیس می‌فرستد. وقتی آلیس پیام ششم را دریافت می‌کند آن را با استفاده از کلید خصوصی خودش رمزگشایی کرده و  $R_A$  خود را درون آن می‌بیند که به او در خصوص هویت باب اطمینان قلبی می‌بخشد. این پیام قطعاً باید از طرف باب آمده باشد چرا که ترویدی هیچ راهی برای تعیین  $R_A$  ندارد. همچنین این پیام تکراری نبوده و جدید است چرا که باب  $R_A$  را [که به صورت تصادفی و غیر تکراری تولید شده است] بازگردانده است. آلیس با برگرداندن پیام هفتم، بر روی کلید نشست توافق می‌کند. وقتی باب،  $R_B$  ارسالی خود را که با کلید نشست رمزنگاری و بازگردانده شده، در پیام هفتم مشاهده می‌کند، متوجه می‌شود که آلیس، پیام ششم را گرفته و  $R_A$  را بررسی کرده است.

۱. بخش ۸-۵ را ببینید. -م.

چگونه ترویدی می تواند این پروتکل را در هم بریزد و در آن رسوخ کند؟ او می تواند پیام شماره ۳ را به صورت جعلی تولید و از طرف آلیس برای باب بفرستد [چرا که کلید عمومی او یعنی  $E_B$  را همه و از جمله ترویدی می دانند] ولی وقتی آلیس در پیام ششم  $R_A$  را مشاهده می کند متوجه می شود که چنین عددی را او نفرستاده و به همین دلیل مراحل کار را ادامه نخواهد داد. ترویدی قادر نخواهد بود پیام هفتم را به صورت جعلی تولید کرده و برای باب بفرستد، چون  $R_B$  و  $K_S$  را نمی داند و به هیچ وجه قادر نخواهد بود آنها را بدون کلید خصوصی آلیس محاسبه و تعیین نماید. لذا او هیچ اقبالی نخواهد داشت.

## ۸-۸ امنیت نامه های الکترونیکی

وقتی یک پیام توسط پست الکترونیکی بین دو سایت راه دور مبادله می شود، بطور معمول آن نامه در طی مسیر خود از دهها ماشین میانی عبور خواهد کرد. هر یک از این ماشینها قادرند آن را بخوانند یا برای استفاده های بعدی ذخیره کنند. برخلاف آنچه که بسیاری از مردم می اندیشند، حریم خصوصی عملاً وجود ندارد ولیکن علیرغم این بسیاری از افراد علاقمندند نامه هایی را که ارسال می کنند فقط گیرنده مورد نظر بخواند نه هیچکس دیگر؛ نه رئیس آنها و نه حتی حکومت. این احساس نیاز بسیاری از گروهها و افراد را ترغیب کرد که اصول رمزنگاری را که تا اینجا بررسی کردیم، بر روی نامه های الکترونیکی اعمال کرده و یک سیستم امن پست الکترونیکی به وجود بیاورند. در بخش بعدی، PGP را مطالعه خواهیم کرد که یک سیستم پست الکترونیکی امن و بسیار رائج است، سپس به اختصار به دو روش PEM و S/MIME نگاهی خواهیم انداخت. برای دسترسی به اطلاعات غنی تر در مورد سیستمهای امن پست الکترونیکی، مرجع (Kaufman et. al. 2002; Schneier, 1995) را ملاحظه نمایید.

## ۱۸-۸ (Pretty Good Privacy) PGP

PGP، به عنوان اولین مثال از سیستم پست الکترونیکی امن، زائیده تفکر شخصی به نام «زیمرمَن» (Phil Zimmermann, 1995) بود. شعار زیمرمَن که یکی از طرفداران حفظ حریم خصوصی افراد است، این بود که: «اگر ایجاد حریم خصوصی و حفظ اسرار مردم قانون شکنی است آنگاه فقط حریم خصوصی قانون شکنان حفظ خواهد شد.» PGP یک بسته نرم افزاری کامل برای امنیت نامه های الکترونیکی است که در سال ۱۹۹۱ منتشر شد و با یک ساختار بسیار ساده کاربری، تمام امکانات ذیل شامل «تدوین نامه های خصوصی» (رمز شده)، «احراز هویت»، «امضای دیجیتالی» و حتی «فشرده سازی اطلاعات» را به صورت یکجا عرضه کرده است. به علاوه، بسته نرم افزاری به همراه کدهای برنامه، بصورت رایگان از طریق اینترنت در دسترس عموم قرار می گیرد. امروزه به دلیل کیفیت بالا، عدم هزینه (قیمت صفر) و وجود نسخه های متفاوت بر روی یونیکس، لینوکس، ویندوز و سیستم عامل مکینتاش (Mac OS)، از آن به صورت گسترده ای استفاده می شود.

PGP داده ها (محتویات نامه) را با استفاده از روشی به نام IDEA<sup>۱</sup> که مبتنی بر رمزنگاری بلوکی است رمز می کند و در آن از کلیدهای ۱۲۸ بیتی و متقارن بهره می گیرد. این روش رمزنگاری در سونیس زمانی ابداع شد که ضعف و اشکالات DES ارزش آن را خدشه دار کرده بود ولی هنوز AES اختراع و معرفی نشده بود. IDEA شبیه به DES و AES است: در چندین دور (Round) بیت های داده را با هم ترکیب می کند ولیکن جزئیات توابع ترکیب بیتها، در مقایسه با DES و AES متفاوت است. در PGP، برای مدیریت کلیدها از RSA و برای بررسی صحت داده ها از MD5 استفاده شده که این روشها را قبلاً معرفی کردیم.

از اولین روز معرفی PGP، جنگ و جدل وسیعی پیرامون آن در گرفت، (Levy, 1993) از آنجایی که زیمرمَن

هیچ اقدامی برای آن که افراد را از توزیع PGP بر روی اینترنت منع کند انجام نداده بود و هر کسی در هر نقطه دنیا می توانست بدان دسترسی داشته باشد، دولت ایالات متحده او را متهم ساخت که قوانین «منع صدور ابزارهای استراتژیک» را نقض کرده است. بازجویی دولت آمریکا از زیرمن پنج سال طول کشید ولی نهایتاً پرونده مختومه شد؛ شاید به دو دلیل: اول آن که زیرمن شخصاً PGP را بر روی اینترنت نگذاشته بود و به همین دلیل وکلای او ادعا کردند که او هیچ چیزی را صادر نکرده است. ثانیاً دولت به این نتیجه رسید که پیروزی در این دادگاه و محکومیت زیرمن بدان معنا تلقی می شود که یک وبسایت که برنامه ای قابل دریافت (Downloadable) در خصوص امنیت ارائه می دهد در شمول قانون منع فروش تجهیزات استراتژیک مثل تانک، زیردریایی، هواپیماهای نظامی و موشکهای هسته ای قرار گرفته است. سالها تبلیغات منفی، احتمالاً هیچ کمکی به آنها نکرده بود و قرار گرفتن سایتهای وب در شمول تجهیزات جنگی یک تبلیغ منفی بزرگ برای دولت بود.

دولت قرار دادن کدهای یک برنامه بر روی وبسایت را در شمول صادرات غیرقانونی قرار داد و بدین ترتیب زیرمن را به مدت پنج سال گرفتار دادگاه کرد در حالیکه اگر کسی کدهای برنامه PGP را به زبان C در یک کتاب تدوین (و آن را با حروف بزرگ و قابل اسکن چاپ می نمود) و سپس آن کتاب را صادر می کرد توسط دولت فقط جریمه می شد چرا که کتاب در رده تجهیزات جنگی قرار نمی گیرد. به هر حال لااقل برای عموسام قدرت شمشیر از قلم بیشتر است.

یکی دیگر از مشكلات حقوقی PGP، مسئله نقض مالکیت امتیاز بود. شرکتی که امتیاز RSA را برای خود ثبت کرده بود (شرکت RSA Security Inc.) اقامه دعوی کرد که PGP با استفاده بدون مجوز از الگوریتم RSA، مرتکب نقض مالکیت حقوق معنوی آن شده است ولی از نسخه ۲.۶ به بعد این مسئله رفع شد. همچنین PGP از الگوریتم IDEA که حق امتیاز آن هم ثبت شده، استفاده کرده است؛ این مسئله نیز مشکلاتی را ایجاد کرد.

از آنجا که کدهای برنامه PGP آزاد است لذا افراد و گروههای مختلف، نسخه های متفاوتی از آن را تولید و عرضه کردند. برخی از این نسخه ها با این هدف طراحی شده اند که قوانین «منع صدور تجهیزات حساس» را دور بزنند و برخی دیگر بر آن متمرکز بوده که از الگوریتمهای ثبت مالکیت شده استفاده نشود و حتی برخی دیگر خواستند آن را در قالب یک محصول تجاری با «کدهای بسته» و غیر آزاد عرضه کنند. اگرچه امروزه قانون صدور تجهیزات حساس، نسبتاً آزادتر شده (البته به غیر از محصولات مبتنی بر AES که به هیچ وجه از ایالات متحده به خارج قابل صدور نیست) و در سپتامبر ۲۰۰۰ نیز حق مالکیت الگوریتم RSA منقضی گردید ولیکن میراث مشکلات و مسائل حقوقی PGP آن شد که نسخه های کاملاً ناسازگار از PGP با نامهای مختلف به جریان بیفتند. توضیح زیر بر روی نسخه کلاسیک PGP که قدیمیترین و ساده ترین نسخه آن است متمرکز خواهد بود. نسخه رایج دیگر آن یعنی Open PGP در RFC 2440 تشریح شده است. همچنین نسخه دیگری به نام GNU Privacy Guard نیز موجود است.

PGP تعمداً به جای ابداع یک روش رمزنگاری جدید از الگوریتمهای موجود استفاده کرده است. این روش، براساس الگوریتمهایی بنیان نهاده شده که تاکنون در مقابل بررسیها و حملات گسترده دوام آورده و شکسته نشده و همچنین آژانسهای دولتی در طراحی آنها تأثیرگذار نبوده اند. برای افرادی که به حکومت اعتماد ندارند این ویژگی، امتیاز بزرگی محسوب می شود.

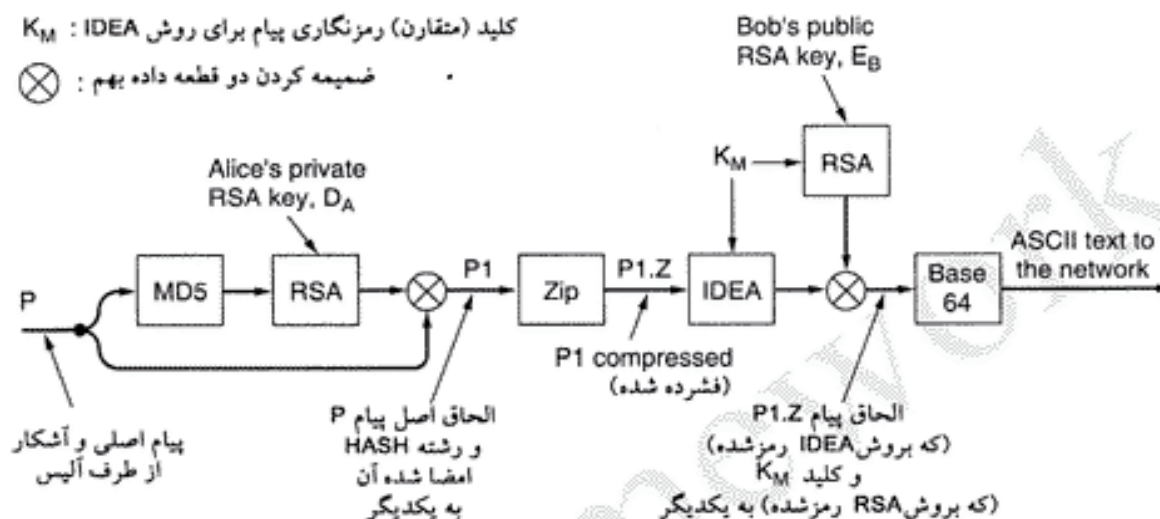
PGP از فشرده سازی متن، سری سازی آن و امضاهای دیجیتالی حمایت می کند و امکانات بسیار جالبی در خصوص مدیریت کلید عرضه کرده است ولی امکانات کمی در خصوص تدوین و ارسال نامه الکترونیکی دارد. PGP را می توان یک «پیش پردازنده» تلقی کرد که متن اصلی را به عنوان ورودی می گیرد و یک متن امضاء شده سری را که در قالب base64 کد شده است، باز می گرداند. البته خروجی آن بعداً می تواند توسط پست الکترونیکی

ارسال شود. برخی از پیاده سازیهای PGP در آخرین مرحله می توانند «عامل کاربر» مثلاً نرم افزاری مثل Outlook را جهت ارسال پیام فراخوانی کنند.

برای آن که ببینیم چگونه کار می کند، اجازه بدهید مثال شکل ۸-۴۴ را بررسی کنیم. در این شکل آلیس می خواهد یک پیام آشکار امضاء شده مثل P را به روشی مطمئن برای باب بفرستد. باب و آلیس هر کدام دارای یک کلید خصوصی ( $D_X$ ) و یک کلید عمومی ( $E_X$ ) در الگوریتم RSA هستند. فعلاً فرض می کنیم هر کدام، کلید عمومی دیگری را می دانند؛ بعداً در خصوص مدیریت کلیدها به اختصار توضیح خواهیم داد.

کلید (متقارن) رمزنگاری پیام برای روش IDEA :  $K_M$

⊗ : ضمیمه کردن دو قطعه داده بهم



شکل ۸-۴۴. عملکرد PGP برای ارسال یک پیام.

آلیس پشت کامپیوترش نشسته و کارش را با فراخوانی برنامه PGP آغاز می کند. PGP ابتدا به روش MD5 پیام P را در هم ریخته (Hash کرده) و یک رشته خلاصه پیام (Message Digest) استخراج نموده و سپس آنرا طبق روش RSA با کلید خصوصی خودش  $D_A$  رمز می کند. بعداً وقتی باب پیام را دریافت کند می تواند رشته Hash را با کلید عمومی آلیس رمزگشایی کرده و صحت آن را بررسی نماید. حتی اگر شخص ثالثی (مثل ترویدی) بتواند رشته Hash را بدست آورده و آن را رمزگشایی کند، بدلیل استحکام روش MD5، این تضمین وجود دارد که نتواند پیام جعلی دیگری را با رشته Hash مشابه، تولید و جایگزین کند. بدین ترتیب در این مرحله صحت و سلامت پیام (Integrity) تضمین می شود.

رشته Hash رمز شده و پیام اصلی به یکدیگر ضمیمه می شوند و یک پیام واحد به نام P1 را ایجاد می کنند. سپس P1 با استفاده از برنامه ZIP که از الگوریتم «لیمپل-زیو» استفاده می کند فشرده می شود. (Lempel - Ziv, 1977). خروجی این مرحله را P1.Z بنامید.

PGP در ادامه، از آلیس یک ورودی تصادفی مطالبه می کند. برای تولید یک کلید ۱۲۸ بیتی بنام  $K_M$  برای الگوریتم رمزنگاری IDEA، از محتویات پیام و سرعت تایپ او (که هر دو تصادفی هستند) استفاده می شود.  $K_M$  در ادبیات PGP کلید نشست نامیده می شود ولی در واقع کلید نشست، اسم بی معنایی است چرا که در ارسال نامه هیچ نشتی ترتیب داده نمی شود). حال پیام P1.Z به روش IDEA و در حالت Feedback Mode<sup>۱</sup>، با کلید  $K_M$  رمزنگاری می شود. مضاف بر این، خود  $K_M$  نیز با استفاده از کلید عمومی باب  $E_B$  رمز می شود. نهایتاً این دو مولفه (یعنی کلید رمز + پیام فشرده و رمز شده) بهم ضمیمه شده و سپس در مبای base64 که در بخش MIME از

۱. بخش ۸-۲۰-۷ را ببینید.

فصل هفتم تشریح کردیم، کدگذاری و تبدیل به متن ASCII می شود. پیام حاصل فقط شامل حروف الفباء، ارقام و سمبولهای '+', '=', '/' خواهد بود بدین معنا که می تواند براحتی در بدنه یک نامه RFC 822 قرار گرفته و بدون هیچ تغییری به مقصد برسد.

وقتی باب پیام را دریافت می کند ابتدا عکس عمل کدگذاری base64 را انجام داده و کلید رمز IDEA را با کلید خصوصی RSA خود، استخراج می نماید. سپس با استفاده از این کلید، بدنه پیام را رمزگشایی می کند تا Plaintext را بدست بیاورد. پس از آن که متن از حالت فشرده خارج شد، باب متن اصلی را از رشته رمزنگاری شده Hash جدا کرده و این رشته را رمزگشایی می کند. اگر رشته Hash متن اصلی با رشته MD5 ارسالی تطابق داشت او از صحت پیام مطمئن شده و یقین حاصل می کند که پیام از طرف آلیس فرستاده شده است.

اشاره به این نکته ارزشمند است که در این ساختار، الگوریتم RSA فقط در دو جا استفاده شده است: (۱) برای رمزنگاری رشته ۱۲۸ بیتی MD5 (۲) برای رمزنگاری کلید ۱۲۸ بیتی IDEA. اگرچه الگوریتم RSA بسیار کند است ولیکن فقط باید ۲۵۶ بیت را رمز کند نه حجم بسیار زیاد اطلاعات را! بعلاوه این ۲۵۶ بیت اطلاعات، مطلقاً تصادفی است و برای حدس کلید، ترویدی به تلاش بسیار زیادی، نیازمند خواهد بود. در عوض حجم سنگین عملیات رمزنگاری بدنه پیام، به روش IDEA که بارها از RSA سریعتر است، انجام می شود. PGP، امنیت، فشرده سازی و امضای دیجیتالی را بطور یکجا عرضه کرده و تمام این عملیات را بسیار سریعتر و مؤثرتر از ساختار پیشنهادی در شکل ۸-۱۹، انجام می دهد.

PGP از کلیدهای RSA با چهار اندازه مختلف حمایت می کند. انتخاب طول مناسب بر عهده کاربر گذاشته شده است. این چهار انتخاب عبارتند از:

۱. کلیدهای علنی (۳۸۴ بیت): امروزه می توان براحتی آن را شکست.
۲. کلیدهای تجاری (۵۱۲ بیت): فقط توسط مؤسسات سه حرفی قابل شکستن است.<sup>۱</sup>
۳. کلیدهای نظامی (۱۰۲۴ بیتی): هیچکس بر روی کره زمین نمی تواند آن را بشکند.
۴. کلیدهای کیهانی (۲۰۴۸ بیتی): هیچکس حتی در سیارات دیگر نمی تواند آن را بشکند!!

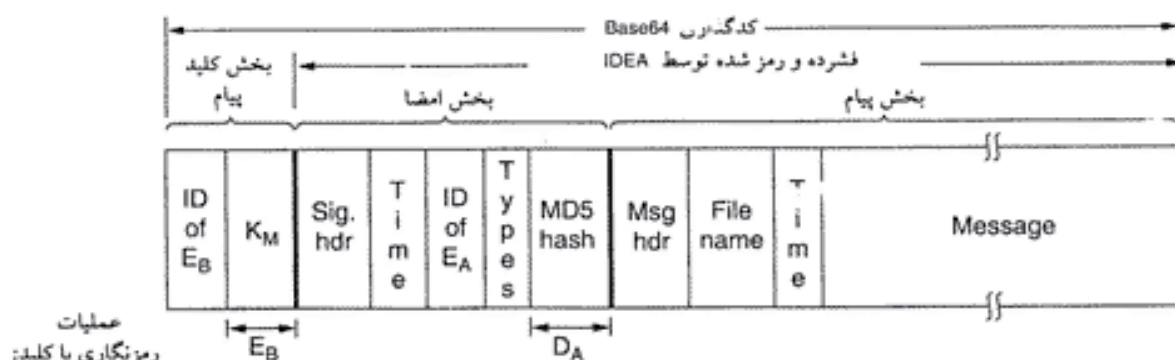
از آنجایی که RSA فقط در دو محاسبه کوچک [یعنی برای رمزنگاری ۲ کلید ۱۲۸ بیتی] بکار گرفته می شود منطقی است که عموم افراد از کلید با طول ۲۰۴۸ بیت استفاده کنند.

قالب یک پیام کلاسیک PGP در شکل ۸-۴۵ نشان داده شده است. البته از قالبهای پیشمار دیگری نیز استفاده می شود. پیام در سه بخش شامل بخش کلید IDEA، بخش امضاء و بخش بدنه پیام سازماندهی شده است. بخش حاوی کلید نه تنها کلید را در بر می گیرد بلکه شماره شناسایی کلید عمومی را نیز شامل می شود. بدین ترتیب کاربران مجاز به داشتن چندین کلید عمومی هستند.

بخش امضاء شامل یک سرآیند خاص است که در اینجا به آن نخواهیم پرداخت. پس از سرآیند امضاء، مهر زمان و سپس شماره شناسایی کلید عمومی فرستنده درج می شود که توسط این کلید عمومی، رشته Hash امضاء دیجیتالی از رمز خارج خواهد شد. سپس در ادامه، یک فیلد Type قرار داده شده که نوع الگوریتم بکار رفته را تعیین می کند (تا مثلاً اجازه بدهد در صورت ابداع MD6 یا RSA2 از آنها استفاده شود!) سپس در انتهای بخش امضاء، رشته کد MD5 پیام قرار می گیرد.

بخش پیام نیز دارای یک فیلد سرآیند است، سپس نام پیش فرض فایل درج می شود تا اگر گیرنده پیام خواست آنرا بر روی دیسک ذخیره کند از این نام استفاده شود. سپس مهر زمان ایجاد پیام، درج شده و نهایتاً پیام اصلی قرار

۱. منظور از مؤسسات سه حرفی CIA, FBI, NSA و امثال آنهاست! س.



شکل ۸-۴۵. قالب پیامهای PGP.

می گیرد.

در PGP به مدیریت کلیدها توجه ویژه ای شده است چرا که پاشنه آشیل تمام سیستمهای امنیتی محسوب می شود! مدیریت کلیدها در PGP بدین نحو انجام می گیرد که هر کاربر به صورت محلی از دو ساختمان داده خاص نگهداری می کند: الف) «دسته کلید خصوصی» ب) «دسته کلید عمومی». «دسته کلید خصوصی» شامل یک یا چند زوج کلید عمومی و همتای خصوصی آنهاست. دلیل آن که از چندین زوج کلید برای هر کاربر حمایت می شود آنست که کاربر بتواند بطور متناوب کلید عمومی خود را عوض کند، یا هر گاه شخصی شک کرد که یکی از کلیدهای خصوصیش فاش شده بلافاصله آن را تغییر بدهد. هر زوج کلید دارای یک شناسه (شماره شناسایی) متناظر است که براساس آن فرستنده پیام به گیرنده اطلاع می دهد که برای رمزگشایی پیام از کدام کلید باید استفاده کند. شناسه کلیدها، ۶۴ بیت کم ارزش هر کلید عمومی است. کاربران خودشان وظیفه دارند که از انتخاب کلیدهایی که ۶۴ بیت کم ارزششان یکسان است خودداری کنند. کلیدهای خصوصی قبل از ذخیره بر روی دیسک با استفاده از یک کلمه عبور (با اندازه طولانی) رمز می شوند تا از خطر سرقت مصون بمانند.

«دسته کلید عمومی» در برگرفته کلیدهای عمومی متعلق به کاربران طرف مقابل ارتباط است. برای رمزنگاری کلیدهای هر پیام، به کلید عمومی گیرنده پیام نیز نیاز است. هر درایه (Entry) در دسته کلیدهای عمومی، نه تنها شامل خود کلید است بلکه شناسه ۶۴ بیتی هر کلید عمومی (۶۴ بیت کم ارزش کلید) و فیلد دیگری که در آن میزان اعتماد کاربر به آن کلید مشخص شده است را نیز در برمی گیرد. در زیر فلسفه این فیلد مشخص شده است. مسئله ای که ممکن است برای PGP کلاسیک بوجود بیاید آنست که فرض کنید کلیدهای عمومی بر روی «بولتن برد» (Bulletin Board) اعلام شده باشد. یک راه برای آنکه ترویدی بتواند نامه های محرمانه باب را بخواند آن است که به این سایت حمله کرده و کلید عمومی او را به دلخواه خود تغییر بدهد. بعداً وقتی آلیس این کلید را به خیال آن که به باب تعلق دارد دریافت می کند، ترویدی خواهد توانست در میانه راه نامه های او را استراق سمع نماید.<sup>۱</sup>

برای پیشگیری از این گونه حملات یا حداقل کاهش تبعات آن، آلیس نیازمند آن است که بداند به چه اندازه می تواند به آیمی که در دسته کلیدهای عمومی، به باب منسوب شده اعتماد کند. فیلد «میزان اعتماد» به همین منظور بکار می رود. اگر آلیس کلید عمومی باب را بر روی یک فلاپی دیسک شخصاً از باب گرفته باشد، می تواند فیلد «میزان اعتماد» این کلید را به مقدار بالایی تنظیم نماید. این روش مدیریت کلید که بصورت غیر متمرکز و تحت نظارت کاربر انجام می شود، PGP را از قید وابستگی به یک PKI متمرکز رها کرده است. علیرغم این، افراد معمولاً کلید عمومی دیگران را با سؤال از یک سرویس دهنده مورد اعتماد توزیع کلید،

۱. حمله «گروه آتش نشان» (bucket brigade) را در بخش ۸-۷-۲ مطالعه نمایید.

بدست می آورند. به همین دلیل پس از آن که X.509 استاندارد شد، PGP در کنار مکانیزم سستی خود از گواهینامه دیجیتالی X.509 نیز حمایت کرد. تمام نسخه های فعلی PGP از گواهینامه های X.509 پشتیبانی می کنند.

### ۲-۸-۸ (Privacy Enhanced Mail) PEM

برخلاف PGP که از ابتدا یک نمایش تکنفره با هنرنمایی زیرمن بود، دومین مثال ما، PEM، یک استاندارد رسمی اینترنت است که در اواخر دهه هشتاد توسعه یافت و در چهار RFC از RFC 1421 تا RFC 1424 تشریح شد. اصول PEM تقریباً با PGP مشابه است: هر دو به منظور احراز هویت و محرمانه نگاه داشتن نامه ها در سیستم های پست الکترونیکی مبتنی بر RFC 822 هستند؛ ولیکن در روشها و فناوری بکار رفته در آن با PGP تفاوت هایی دارد.

پیام های ارسالی توسط PEM در ابتدا به قالب استاندارد تبدیل می شود به نحوی که تمام نامه ها از قواعد و استاندارد مشابهی در خصوص فضاهای خالی (همانند فاصله ها، Tab و...) پیروی کنند. سپس با استفاده از MD2 یا MD5 برای نامه یک رشته Hash بدست می آید. پس از الحاق رشته Hash به پیام، مجموع این دو با استفاده از روش DES رمزنگاری می شود. با در نظر داشتن ضعف کلیدهای ۵۶ بیتی در DES، این انتخاب کاملاً مشکوک به نظر می رسد. سپس پیام رمز شده به روش base64 کدگذاری و برای گیرنده آن ارسال می شود. همانند PGP، هر پیام با استفاده از کلید یک مرحله ای و متقارن رمز شده و آن کلید نیز در کنار پیام جاسازی و ارسال می شود. این کلید با استفاده از روش رمزنگاری RSA یا روش Triple DES (DES سه گانه مبتنی بر مکانیزم EDE) رمز می گردد.

مدیریت کلید در PEM نسبت به PGP سازمان یافته تر است. نیدها با گواهینامه های X.509 که توسط مراکز صدور گواهی -CA- صادر شده اند، تصدیق می شوند. مراکز صدور گواهی در یک ساختار سلسله مراتبی که از مرکزی به نام «ریشه» (Root) شروع می شود، سازماندهی شده اند. حسن این ساختار آنست که ابطال گواهینامه ها با انتشار متناوب فهرست CRL<sup>۱</sup> توسط «ریشه»، بسادگی میسر می باشد.

تنها مشکل PEM آن است که هیچکس تاکنون از آن استفاده نکرده و تقریباً به هیچ پیوسته است! معضل آن بیشتر سیاسی بود: چه کسی و تحت چه شرایطی باید مسئولیت «ریشه» (Root) را بر عهده بگیرد؟ نامزدهای این مسئولیت کم نبودند ولی بسیاری افراد از اعتماد به یک شرکت واحد می ترسند. جدی ترین نامزد این مسئولیت، شرکت RSA بود که می خواست به ازای صدور هر گواهینامه مبلغی را دریافت کند. بعضی مؤسسات از این ایده استقبال نکردند. مخصوصاً از این جهت که دولت ایالات متحده مجاز به استفاده از تمام ابداعات و اختراعات ثبت شده در آن کشور است و همچنین شرکتهای خارج از ایالات متحده عادت داشتند از الگوریتم RSA به رایگان استفاده کنند (شرکت RSA فراموش کرده بود امتیاز استفاده از RSA را در خارج از ایالات متحده به نفع خود ثبت کند) مؤسسات مختلف علاقمند نبودند برای کاری که قبلاً به رایگان انجام می شد به شرکت RSA پول بپردازند. نهایتاً هیچکس برای پذیرش مسئولیت «ریشه» [در ساختار سلسله مراتبی صدور گواهینامه] پیدا نشد و PEM شکست خورد.

### ۳-۸-۸ S/MIME

اقدام بعدی IETF برای امنیت سیستم پست الکترونیکی که S/MIME نامیده شد در اسناد RFC 2632 تا RFC 2643 تشریح شده است. همانند PEM، این روش نیز امکان «احراز هویت»، «تایید صحت اطلاعات»، «سری نگاه داشتن پیام» و «غیرقابل انکار بودن پیام» را تضمین کرده است. همچنین این روش کاملاً قابل انعطاف

۱. CRL: «فهرست گواهینامه های باطل شده»

بوده و از الگوریتمهای رمزنگاری مختلفی پشتیبانی می‌کند. همانگونه که از نام S/MIME مشخص است این روش، استاندارد MIME را تکمیل و امن کرده است فلذا عجیب نیست که از انواع مختلف پیامها حمایت و حفاظت می‌کند.<sup>۱</sup> در این استاندارد تعدادی سرآیند جدید برای MIME تعریف شده که برای عملیاتی نظیر مشخص کردن امضای دیجیتالی پیام، کاربرد دارند.

IETF قطعاً از تجربه PEM چیزهایی را آموخته بود. S/MIME به یک ساختار سلسله‌مراتبی صدور گواهینامه که از «ریشه» شروع شود نیاز ندارد. کاربران می‌توانند به جای یک مؤسسه واحد به عنوان ریشه، مجموعه‌ای از مؤسسات مورد اعتماد صدور گواهینامه (Trust Anchors) را به خدمت بگیرند. هر گاه در سلسله‌مراتب تایید گواهینامه‌های دیجیتالی، گواهینامه‌ای به یکی از این مؤسسات مورد اعتماد کاربر ختم گردد، معتبر فرض می‌شود. S/MIME از پروتکلهای و استانداردهایی استفاده می‌کند که تا اینجا همه آنها را بررسی کرده‌ایم و بیش از این، بدانها نخواهیم پرداخت. برای دسترسی به شرح مبسوط آن به RFCهای مربوطه مراجعه نمایید.

## ۹-۸ امنیت وب

تا اینجا دو زمینه بسیار مهم را که مقوله امنیت در آنها ضروری است، مطالعه کرده‌ایم: امنیت در مخابرات داده‌ها و امنیت در سیستم پست الکترونیکی. می‌توانید اینها را به عنوان سوپ و پیش‌غذا تلقی کنید. حال وقت آن رسیده تا به موضوع اصلی بپردازیم: «امنیت وب». امروزه وب جولانگاه اخلاط‌گران و ترودیهایی است که به کارهای کثیف خود مشغولند. در بخشهای آتی به برخی از موارد و مشکلات مربوط به امنیت وب نگاهی خواهیم انداخت.

بطور تقریبی امنیت وب را می‌توان در سه بخش تقسیم کرد: اول آن که چگونه اشیاء (Objects) و منابع (Resources) به روشی مطمئن نامگذاری شوند؟ دوم آنکه چگونه می‌توان ارتباطی تایید شده و مطمئن برقرار کرد؟ سوم وقتی یک وب‌سایت برای مشتری خود یک قطعه کد قابل اجرا می‌فرستد چه اتفاقی می‌افتد؟ پس از آنکه به تهدیدهای بالقوه در وب نگاهی انداختیم این سه مورد را نیز بررسی خواهیم کرد.

### ۱-۹-۸ تهدیدها

تقریباً هر هفته در روزنامه‌ها و مجلات، مطالبی در خصوص مشکلات امنیتی سایتهای وب می‌خوانید. این وضعیت واقعاً فاجعه است. بیایید مثالهایی از آنچه که در گذشته اتفاق افتاده را بررسی کنیم. اولاً صفحه وب اصلی (Home Page) بسیاری از مؤسسات مورد حمله قرار گرفته و با صفحات وب دلخواه اخلاط‌گران (Crackers) تعویض شده است. (مطبوعات عموماً به کسی که حریم کامپیوترها را درهم می‌شکند، نفوذگر یا Hacker می‌گویند در حالی که بسیاری از برنامه‌نویسان این اصطلاح را برای یک برنامه‌نویس نخبه بکار می‌برند. ما نیز ترجیح می‌دهیم این افراد را اخلاط‌گر یا «Cracker» بنامیم.) سایتهایی که تاکنون درهم شکسته و در آنها اختلال شده شامل سایتهای مشهوری مثل Yahoo، U.S.Army، CIA، NASA و New York Times بوده است. در بسیاری موارد، اخلاط‌گران پس از حمله، جملات و متون مسخره در این سایتها درج کرده‌اند و پس از چند ساعت این سایتها اصلاح شده‌اند.

حال موارد خطرناکتر را بررسی می‌نمائیم. تعداد بی‌شماری از سایتها توسط حمله DoS (حمله نوع اختلال در سرویس‌دهی) از کار افتاده‌اند و اخلاط‌گر موفق شده یک سایت را با ترافیکی سیل‌آسا مواجه کند به نحوی که قادر به پاسخ‌دهی به تقاضاهای مجاز نباشد. گاهی اوقات این حمله توسط تعداد بسیار زیادی از ماشینها بر علیه یک

۱. در فصل قبل دانستید که MIME تقریباً از هر نوع پیامی (با هر قالب و محتوا) حمایت می‌کند. طبعاً S/MIME نیز باید همینگونه باشد. م.

سایت شکل گرفته که این ماشینها خود قربانی حملات قبلی اخلاالگران بوده‌اند<sup>۱</sup> (حمله نوع DDoS<sup>۲</sup>). این نوع حمله به قدری رایج شده که برای هیچکس خبر جدیدی تلقی نمی‌شود ولی سایت مورد حمله گاهی هزاران دلار متضرر می‌گردد.

در سال ۱۹۹۹ یک اخلاالگر سوئدی در سایت وب Microsoft Hotmail رسوخ کرد و با ایجاد یک سایت معادل (Mirror Site) که در آن کاربران نام کاربری و کلمه عبور خود را درج می‌کردند موفق شد نامه‌های جاری و بایگانی شده تمام افراد مراجعه کننده به این سایت را بخواند.

در یک مورد دیگر، اخلاالگر ۱۹ ساله روسی که ماکسیم نام داشت توانست به یک سایت تجارت الکترونیکی نفوذ کند و شماره ۳۰۰۰۰۰ کارت اعتباری را بدزدد. او نزد صاحب سایت رفت و به او گفت که اگر صد هزار دلار به او پرداخت نکنند تمام این شماره‌ها را بر روی اینترنت منتشر خواهد کرد. آنها به خواسته او اعتنایی نکردند و او نیز با لجاجت شماره کارت‌های اعتباری دیگران را بر روی اینترنت گذاشت و ضرر و زیان زیادی به شماری از قربانیان بی‌گناه تحمیل کرد.

در یک حمله دیگر، یک دانشجوی ۲۳ ساله کالیفرنایی گزارشی را برای یک آژانس خبری ارسال کرد و به دروغ عنوان نمود که شرکت «امیولکس» (Emulex Corporation) می‌خواهد در تراز اقتصادی سه ماهه خود، حجم زیان بسیار بالایی را اعلام کند و مسئولان آن بلافاصله استعفا خواهند کرد. در عرض چند ساعت، سهام این شرکت ۶۰ درصد سقوط کرد و سهامداران بیش از دو میلیارد دلار متضرر شدند!! عوامل این تخلف با معامله سهام خود قبل از ارسال این گزارش، حدود یک چهارم میلیون دلار بدست آوردند. اگرچه این حادثه در اثر حمله و نفوذ به یک وب‌سایت نبود ولیکن روشن است که قرار دادن چنین گزارشهایی در صفحه اصلی یک شرکت بزرگ، تأثیر مشابه و وخیمی بجا خواهد گذاشت.

متأسفانه می‌توانیم از اینگونه حملات، صفحات بسیار زیادی مطلب و گزارش بنویسیم ولیکن زمان آن فرا رسیده که به موارد فنی در خصوص امنیت وب پردازیم. برای اطلاعات بیشتر در خصوص انواع مختلف و مستند مشکلات امنیتی وب به مراجع (Anderson, 2001; Garfinkel with Spafford, 2002; Schneier, 2000) مراجعه نمایید. با جستجو در اینترنت نیز تعداد بی‌شماری از موارد مشابه را خواهید یافت.

## ۲-۹-۸ نامگذاری مطمئن

اجازه بدهید با یک مسئله بسیار بنیادی شروع کنیم: آلیس تمایل دارد وب‌سایت متعلق به باب را ببیند. لذا آدرس URL باب را در نرم‌افزار مرورگر خود درج کرده و چند ثانیه بعد یک صفحه وب بر روی نمایشگر او ظاهر می‌شود. آیا این صفحه واقعاً متعلق به باب است؟ ممکن است باشد و احتمال دارد نباشد! ممکن است ترودی مجدداً از حق قدیمی خود استفاده کرده باشد. به عنوان مثال ممکن است او در میانه راه تمام بسته‌های خروجی آلیس را دریافت و بازرسی کند. وقتی ترودی در بین بسته‌ها به یک تقاضای HTTP GET که به سمت وب‌سایت باب روانه شده بر می‌خورد، می‌تواند خود به وب‌سایت باب مراجعه کرده و آن صفحه را دریافت و تغییرات مورد نظر خود را در آن ایجاد کند و صفحه جعلی را به آلیس برگرداند. حال ترودی می‌تواند (مثلاً) قیمت کالاهای فروشگاه الکترونیکی باب را به قدری پایین آورد تا جلب توجه کرده و آلیس را فریب بدهد تا شماره کارت اعتباری خود را جهت خرید کالا از «باب» ارسال نماید.

۱. به ماشینهایی که خود قربانی نفوذ بدخواهان شده‌اند و ناخودآگاه در حمله به یک سایت شرکت می‌کنند اصطلاحاً «ماشینهای

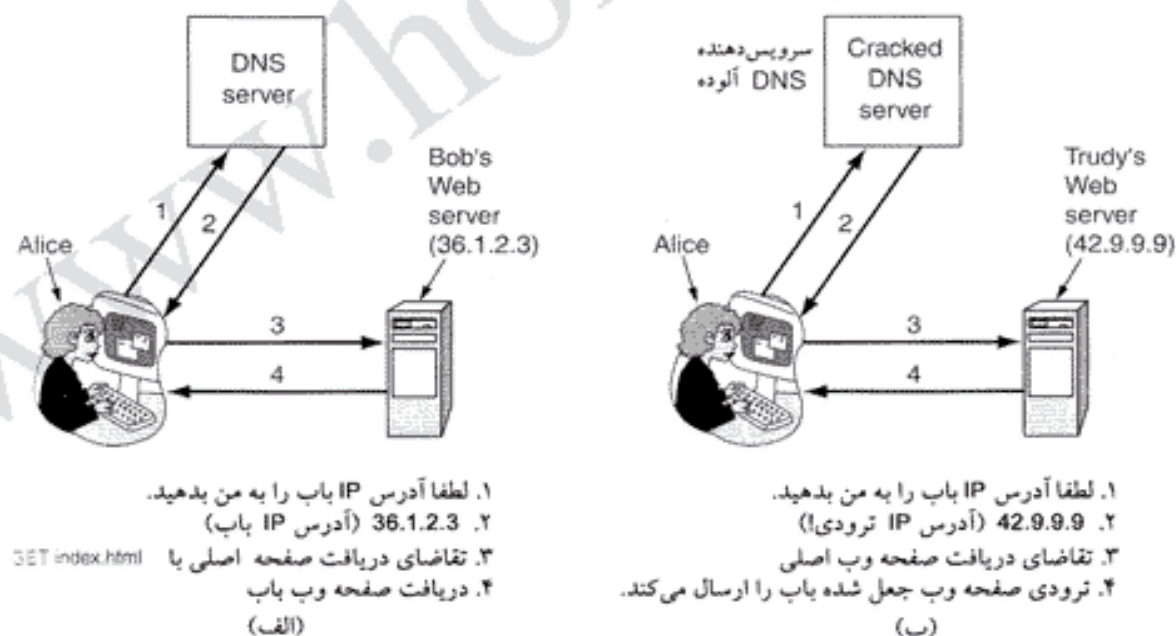
۲. Distributed Denial of Service.

زامبی» (Zombie) گفته می‌شود. -م.

یکی از اشکالات حمله <sup>۱</sup>mitm (از دیدگاه اخلاطگران) آنست که ترودی مجبور خواهد بود در موقعیتی قرار بگیرد تا بتواند ترافیک خروجی آلیس را دریافت کرده و ترافیک ورودی آن را دستکاری و جعل نماید. در عمل او بایا، از خط تلفن آلیس یا باب انشعاب بگیرد چرا که انشعاب از ستون فقرات شبکه ای که با فیبرنوری ایجاد شده بسیار دشوار و ناموفق است. اگرچه ایجاد انشعاب فعال از خطوط تلفن قطعاً ممکن است ولیکن باید کارهای فیزیکی دشواری انجام بشود و چون ترودی گذشته از آنکه زیرک است، تنبل هم هست برای فریب دادن آلیس، راه ساده تری را در پیش می گیرد:

### DNS Spoofing

به عنوان نمونه فرض کنید ترودی قادر به نفوذ در سیستم DNS شده است و توانسته در حافظه نهان DNS (DNS Cache) متعلق به ISP آلیس، تغییر ایجاد کرده و آدرس IP باب (مثلاً 36.1.2.3) را با آدرس IP خودش (42.9.9.9) عوض کند. این دستکاری و تغییر، حمله ذیل را در پی خواهد داشت. برای شروع، عملکرد معمول سیستم DNS را در شکل ۸-۴۶ الف در نظر بگیرید. در این شکل: (۱) آلیس از DNS، آدرس IP باب را سؤال می کند. (۲) پاسخ آن را دریافت می کند. (۳) از باب، صفحه اصلی سایت (Home Page) او را تقاضا می کند. (۴) صفحه را دریافت می نماید. پس از آن که ترودی آدرس IP رکورد DNS باب را با آدرس IP خودش عوض کرد، وضعیت ۸-۴۶ ب را خواهیم داشت. در اینجا وقتی آلیس آدرس IP باب را درخواست می کند، آدرس ترودی را تحویل خواهد گرفت لذا کل ترافیک داده هایی که باید به سمت باب هدایت شود به سوی ترودی می رود. حال ترودی می تواند حمله mitm را پایه ریزی کند بدون آن که نیازی به انشعاب گرفتن از خط تلفن باب یا آلیس داشته باشد. او فقط باید بتواند در سرویس دهنده DNS نفوذ کرده و تنها یک رکورد را تغییر بدهد که بسیار ساده تر از انشعاب گرفتن از خطوط ارتباطی است!



شکل ۸-۴۶. (الف) وضعیت طبیعی. (ب) حمله ای بر اساس دستکاری در داده های DNS و

تغییر در رکورد مربوط به باب.

ترودی چگونه می تواند DNS را فریب داده و دستکاری کند؟ این کار نسبتاً ساده به نظر می رسد! بطور اجمالی، ترودی می تواند سرویس دهنده DNS در ISP آلیس را وادار به ارسال تقاضایی جهت ترجمه آدرس باب نماید. متأسفانه چون DNS از UDP بهره می گیرد، سرویس دهنده DNS هیچ راهی برای بررسی آن که واقعاً چه کسی پاسخ این تقاضا را داده، ندارد. ترودی می تواند از این ویژگی سوء استفاده کرده و پاسخ مورد نظر DNS را جعل و بدین نحو آدرس IP غلطی را در حافظه نهان سرویس دهنده DNS (DNS Cache) تزریق نماید. برای ساده تر شدن بحث فرض می کنیم که سرویس دهنده DNS در ISP آلیس، هیچ درایه ای (Entry) در خصوص آدرس وب سایت باب، *bob.com*، ندارد. البته اگر چنین درایه ای وجود داشته باشد ترودی می تواند آنقدر صبر کند تا زمان اعتبار آن منقضی شود و بعداً تلاش خود را از سر بگیرد. (با از روشهای دیگر استفاده کند).

ترودی حمله خود را با ارسال تقاضای ترجمه آدرس *bob.com* به سمت سرویس دهنده DNS در ISP آلیس، آغاز می نماید. از آنجایی که هیچ درایه ای برای این نام در حافظه DNS وجود ندارد، سرویس دهنده DNS از سرویس دهنده سطح بالا یعنی سرویس دهنده *com*، در خصوص این نام سؤال می کند. حال ترودی در نقش سرویس دهنده DNS سطح بالا حمله کرده و یک پاسخ جعلی و دروغین با این مضمون برای DNS می فرستد: «نام *bob.com* معادل با 42.9.9.9 است!!» در حالی که این آدرس IP متعلق به خود اوست. اگر پاسخ جعلی و غلط او زودتر به ISP آلیس برسد در حافظه سرویس دهنده DNS درج می شود و پاسخ واقعی که بعداً می رسد به عنوان پاسخی بیجا و سرزده کنار گذاشته خواهد شد. فریب دادن سرویس دهنده DNS به نحوی که آدرس IP جعلی و دروغین را در حافظه نهان خود درج کند اصطلاحاً «DNS Spoofing» نامیده می شود.<sup>۱</sup> به حافظه نهان که در آن آدرسهای IP غلط درج شده است اصطلاحاً «حافظه نهان سمی» (Poisoned Cache) گفته می شود.

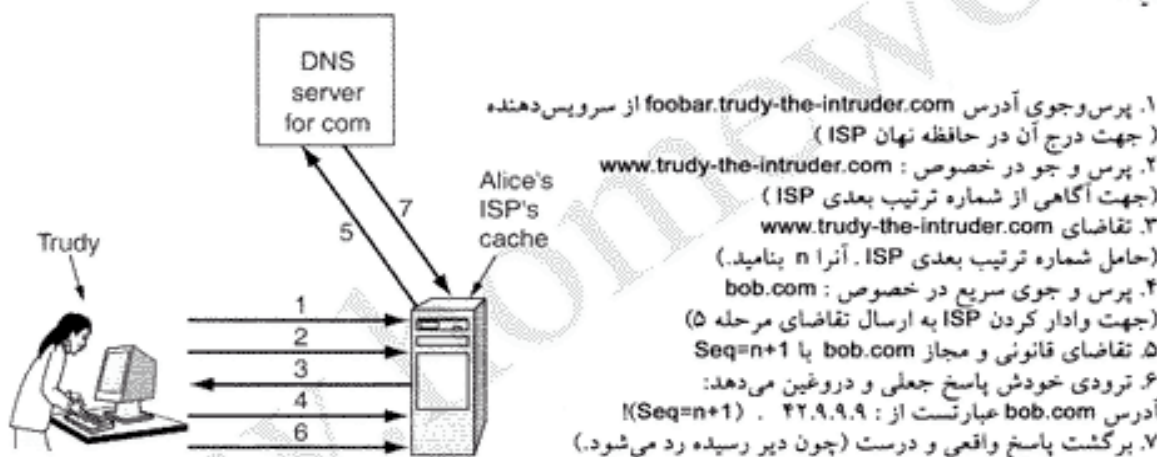
البته انجام این کارها با دشواریهایی روبروست: اول آن که ISP آلیس، بررسی می کند تا ببیند آیا بسته پاسخ دریافتی، آدرس IP سرویس دهنده معتبر و واقعی سطح بالا را در خود حمل می کند یا خیر؟ از آنجایی که ترودی می تواند هر چه که خواست در فیلد آدرس IP بسته پاسخ قرار بدهد لذا قادر است این بررسی را ناکام بگذارد زیرا آدرسهای IP سرویس دهنده های سطح بالا شناخته شده هستند و او می تواند از این آدرسها استفاده کرده و پاسخ جعلی خود را با این آدرسها بفرستد.

دوم آنکه سرویس دهنده DNS برای تشخیص اینکه کدام پاسخ متعلق به کدام تقاضا بوده است برای هر بسته تقاضا یک شماره ترتیب در نظر می گیرد. برای آن که ISP آلیس فریب داده شود، ترودی مجبور است شماره ترتیب تقاضای جاری را بداند. ساده ترین راه برای فهمیدن شماره تقاضای جاری آنست که ترودی برای خودش یک نام حوزه مثل *trudy-the-intruder.com* ثبت کند. فرض کنیم آدرس IP او 42.9.9.9 است. ترودی همچنین یک سرویس دهنده DNS برای آدرس ثبت شده خودش به نام *dns.trudy-the-intruder.com* ایجاد می کند. البته آدرس IP این سرویس دهنده نیز 42.9.9.9 است چراکه ترودی یک ماشین بیشتر در اختیار ندارد. حال باید ISP آلیس را از وجود DNS خودش مطلع نماید. این کار بسیار ساده است؛ تمام کاری که باید انجام شود آنست که از ISP آلیس، در مورد نامی مثل *foobar.trudy-the-intruder.com* سؤال نماید. این کار ISP آلیس را وادار می کند تا از سرویس دهنده سطح بالای *com*، در خصوص نام حوزه جدید ترودی سؤال کند.

پس از آنکه آدرس *dns.trudy-the-intruder.com* در حافظه نهان DNS آلیس درج شد حمله اصلی می تواند شروع شود. ترودی تقاضایی به سمت DNS متعلق به ISP آلیس می فرستد و ترجمه نام *www.trudy-the-intruder.com* را خواستار می شود. این ISP طبق معمول یک بسته پرسش به سمت سرویس

۱. یعنی DNS متعلق به ISP آلیس را وادار به پرسش در خصوص نام *bob.com* می کند و بلافاصله خودش به این پرسش جواب غلط می دهد؛ چون DNS از UDP بهره می گیرد تشخیص هویت پاسخ دهنده میسر نیست. سم.

دهنده ترودی ارسال می کند. این بسته تقاضا شماره ترتیب مورد نیاز ترودی را با خود حمل می نماید. ترودی بلافاصله از ISP آلیس در مورد نام حوزه باب (bob.com) سؤال می کند. سپس به سرعت به سؤال خودش پاسخ داده و بسته های جعلی را که در ظاهر به نظر می رسد از سرویس دهنده های سطح بالا رسیده برای آن ISP می فرستد. شماره ترتیب این بسته های جعلی یکی بیشتر از بسته ای است که چند لحظه قبل دریافت شده بود. البته برای اطمینان بیشتر ترودی می تواند یک بسته پاسخ با شماره ترتیب دو واحد بیشتر یا دهها بسته با شماره ترتیب متوالی (با شماره بیشتر) بفرستد. بالاخره یکی از آنها با شماره ترتیب تقاضای ارسالی منطبق خواهد شد و بقیه حذف خواهند گردید. وقتی پاسخ جعلی توسط DNS آلیس دریافت شد در حافظه نهان ذخیره می شود. هرگاه در آینده پاسخ واقعی دریافت شود با توجه به آن که قبلاً پاسخ جعلی دریافت شده، کنار گذاشته خواهد شد. حال وقتی آلیس آدرس bob.com را جستجو می کند به او گفته می شود که از آدرس 42.9.9.9 استفاده کند که همان آدرس ترودی است. بدین ترتیب ترودی موفق شده از اتاق محل زندگی خود، حمله mitm را بر علیه آلیس پی ریزی و اجرا نماید. گامهای بعدی این حمله در شکل ۸-۴۷ نشان داده شده است. بدتر از همه، روش فوق تنها راه فریب دادن DNS نیست. راههای دیگری هم هستند که ترودی می تواند در صورت عدم موفقیت آنها را نیز بیازماید.



شکل ۸-۴۷. چگونگی فریب دادن ISP آلیس توسط ترودی.

### Secure DNS

این حمله خاص را می توان بدین نحو خنثی کرد که DNS بجای استفاده از شماره ترتیب شمارشی برای بسته های پرسش و پاسخ از شماره های کاملاً تصادفی بهره بگیرد ولی به نظر می رسد که هرگاه یک رخنه نفوذ مسدود شود، در کنار آن رخنه دیگری در سیستم ایجاد می شود. مشکل اصلی آن است که DNS در زمانی طراحی شده که اینترنت فقط یک ابزار تحقیقاتی و آن هم در چند صد دانشگاه بیشتر نبود و هیچیک از افراد نظیر آلیس، باب یا ترودی به این محفل دعوت نشده بودند. در آن زمان امنیت، مورد مهمی به حساب نمی آمد؛ مهمترین مورد آن بود که اینترنت در همه حال کار کند. در طول این سالها شرایط حاکم بر محیط اینترنت پشت تغییر کرد؛ بهمین دلیل در سال ۱۹۹۴، IETF گروهی را مأمور ساخت تا DNS را بطور پنیادی امن نمایند. این پروژه به نام DNSsec شناخته می شود و نتیجه آن در RFC 2535 ارائه شده است. متأسفانه از DNSsec در ترکیب شبکه ها به صورت گسترده و کامل استفاده نمی شود و به همین دلیل هنوز هم بسیاری از سرویس دهنده های DNS نسبت به حمله DNS Spoofing آسیب پذیر هستند.

DNSsec، از دیدگاه مفهومی فوق العاده ساده است. این سرویس دهنده بر اصول رمزنگاری با کلید عمومی

استوار است. هر ناحیه در DNS (DNS Zone) (با دیدگاه شکل ۷-۴) دارای یک زوج کلید عمومی و خصوصی است. تمام اطلاعاتی که توسط سرویس دهنده DNS فرستاده می شود قبل از ارسال، با کلید خصوصی «ناحیه مبدأ»<sup>۱</sup> امضاء شده و بدین ترتیب گیرنده آنها می تواند هویت این اطلاعات را بررسی نماید.

DNSsec سه دسته خدمات اساسی زیر را ارائه می کند:

۱. اثبات آن که داده ها از کجا منشأ گرفته اند و به چه کسی متعلقند.

۲. توزیع کلید عمومی

۳. احراز هویت تقاضاها و تعاملاتی که با سرویس دهنده صورت می گیرد.

اصلی ترین سرویس همان مورد اول است که براساس آن بررسی می شود که اطلاعات برگشتی از DNS واقعاً متعلق به چه کسی است. مورد دوم برای ذخیره و بازیابی مطمئن کلیدهای عمومی کاربرد دارد. سومین مورد بدان جهت نیاز است تا بتوان حمله نوع تکرار (Replay Attack) و حمله DNS Spoofing را خنثی کرد. دقت کنید که در DNSsec خدمات رمزنگاری داده ها ارائه نمی شود زیرا تمام اطلاعات DNS، عمومی و غیرمحرمانه است. انتظار می رود مراحل جایگزینی DNS با DNSsec معمولی چندین سال طول بکشد لذا سرویس دهنده DNSsec باید این قابلیت اساسی را داشته باشد که با سرویس دهنده های معمولی و ناامن نیز کار کند و طبقاً پروتکل DNSsec در مقایسه با DNS، تغییر بنیادی و ناسازگار نداشته است. اجازه بدهید نگاهی به این پروتکل بیندازیم. رکوردهای DNS در گروههایی به نام RRSet (Resource Record Sets) دسته بندی شده اند. تمام رکوردهایی که «نام دامنه»، «نوع» و «کلاس» مشابه دارند در یک مجموعه RRSet قرار می گیرند.<sup>۲</sup> یک مجموعه RRSet ممکن است شامل چندین رکورد A (رکوردهای آدرس IP) باشد چرا که مثلاً یک سرویس دهنده ممکن است دارای یک آدرس IP اولیه و یک آدرس ثانویه باشد. در هر مجموعه RRSet چندین رکورد نوع جدید (که در ادامه بررسی خواهند شد) قرار می گیرد. همچنین برای هر مجموعه RRSet یک رشته Hash (که صحت آن مجموعه را تأیید می کند) به یکی از روشهای معمول مثل SHA-1 یا MD5 محاسبه می شود و نهایتاً با استفاده از کلید خصوصی صاحب آن ناحیه (Zone) به روشی مثل RSA رمزنگاری می گردد. کوچکترین واحد اطلاعات که برای مشتری ارسال می شود یک مجموعه امضاء شده RRSet است. پس از دریافت RRSet امضاء شده، مشتری (Client) می تواند بررسی کند که آیا واقعاً با کلید خصوصی ناحیه مبدأ امضاء شده است؟ اگر امضای آن منطبق بود داده ها پذیرفته می شوند. از آنجایی که هر مجموعه RRSet امضای خودش را به همراه دارد می تواند در هر جایی ذخیره (cache) شود، حتی بر روی سرویس دهنده های غیر قابل اعتماد؛ بدون آن که خدشهای به امنیت آنها وارد گردد.

DNSsec چندین نوع رکورد جدید معرفی کرده است. اولین آنها رکورد نوع KEY است. این رکورد کلید عمومی یک «ناحیه» (Zone) یا کاربر یا ماشین میزبان (Host) و همچنین الگوریتم رمزنگاری بکار رفته برای امضاء، پروتکل انتقال و تعدادی مشخصه دیگر را نگهداری می کند. کلید عمومی به صورت آشکار (رمز نشده) ذخیره می شود. از گواهینامه های X.509 به دلیل حجم بزرگ استفاده نشده است. در رکورد نوع KEY فیلد مشخص کننده الگوریتم، برای امضاهای مبتنی بر MD5/RSA به عدد ۱ تنظیم می شود (انتخاب ارجح) و مقادیر دیگر، ترکیب الگوریتمهای مورد نظر را مشخص می کند. فیلد پروتکل می تواند استفاده از IPsec یا هر پروتکل امنیتی دیگر را (در صورت وجود) تعیین نماید.

دومین نوع جدید رکوردها، رکورد SIG است. این رکورد، رشته Hash امضاء شده یک ناحیه (Zone) را

۲. در خصوص این واژه ها به فصل هفتم مراجعه کنید. -م.

۱. Originating Zone.

مشخص می نماید. (رشته Hash براساس الگوریتمی که در رکورد KEY مشخص شده، امضاء می گردد). این امضاء بر روی تمام رکوردهای یک مجموعه RRSet شامل رکوردهای KEY (به استثنای خودش)، اعمال می شود. رکورد SIG، زمان شروع اعتبار و زمان انقضای امضاء و همچنین نام امضاءکننده و چند آیت دیگر را نیز تعیین می کند.

طراحی DNSsec به گونه ای است که می توان کلیدهای خصوصی را به صورت جدا و در ماشینی غیرمتصل به شبکه (به صورت offline) نگهداری کرد. هر روز یا هر دو روز یکبار، محتویات پایگاه داده DNS به صورت دستی (مثلاً توسط CD-ROM) بر روی یک ماشین غیرمتصل به شبکه که کلیدهای خصوصی را نگه می دارد منتقل می گردد. در آنجا تمام مجموعه های RRSet امضاء شده و برای هر مجموعه، یک رکورد SIG (امضاء) تولید و نتیجه توسط CD-ROM به ماشین اصلی برگردانده می شود. بدین ترتیب می توان کلیدهای خصوصی را بر روی یک CD-ROM ذخیره کرد و CD-ROM به جز در مواقعی که باید مجموعه های RRSet را امضاء کرد در جای مطمئنی نگهداری شود. پس از امضاء، حافظه و دیسک آن ماشین کاملاً پاک شده و CD-ROM به محل امن خود برگردانده می شود. در این روال، امنیت الکترونیکی به امنیت فیزیکی کاهش می یابد و پیچیدگی چندانی ندارد. این روش که در آن مجموعه های RRSet از قبل امضاء می شوند، سرعت فرآیند پاسخ به تقاضاها را بشدت افزایش می دهد چراکه لازم نیست در حین پاسخ و ارسال هیچگونه عمل رمزنگاری انجام شود.

وقتی ماشین مشتری، یک مجموعه RRSet امضاء شده را دریافت می کند ابتدا باید کلید عمومی آن ناحیه را اعمال کرده و رشته Hash را بدست بیاورد؛ سپس خودش رشته Hash را مستقلاً محاسبه کرده و این دو مقدار را با هم مقایسه کند. اگر این دو مقدار یکسان بود، داده ها معتبر فرض می شود. این روال یک سؤال ایجاد می کند و آن هم این که مشتری چگونه کلید عمومی هر ناحیه را بدست آورده و بدان اعتماد کند؟ یک راه حل آن است که این کلیدها از یک سرویس دهنده معتبر و مورد اعتماد و براساس یک اتصال امن نظیر IPsec اخذ شود.

با این وجود، در عمل فرض شده که ماشینهای مشتری به گونه ای پیکربندی شده اند که کلیدهای عمومی حوزه های سطح بالا را به صورت پیش فرض در اختیار دارند. حال اگر آلیس بخواهد به وب سایت باب مراجعه کند می تواند از DNS بخواهد که مجموعه RRSet متناظر با bob.com را در اختیار او بگذارد که در آن آدرس IP و همچنین رکورد KEY (شامل کلید عمومی باب) مشخص شده است. مجموعه RRSet توسط مسئول حوزه سطح بالای com. امضاء می شود لذا آلیس می تواند اعتبار آن را بررسی کند زیرا کلید عمومی حوزه com. را همه می دانند و نیازی به سؤال نیست. یک مثال از رکوردهایی که می تواند در یک مجموعه RRSet وجود داشته باشد در شکل ۸-۴۸ نشان داده شده است.

حال با در اختیار داشتن نسخه ای معتبر از کلید عمومی باب، آلیس می تواند از سرویس دهنده DNS متعلق به باب، در خصوص آدرس IP معادل با www.bob.com سؤال کند. اگر ترویدی به نحوی برنامه ریزی کند که تعدادی RRSet را به صورت جعلی تشکیل داده و در حافظه نهان یک سرویس دهنده DNSsec تزریق نماید، آلیس قادر

Domain name	Time to live	Class	Type	Value
bob.com.	86400	IN	A	36.1.2.3
bob.com.	86400	IN	KEY	3682793A7B73F731029CE2737D...
bob.com.	86400	IN	SIG	86947503A8B848F5272E53930C...

شکل ۸-۴۸. مثالی از رکوردهای RRSet برای نام حوزه bob.com. رکورد KEY حاوی کلید عمومی باب است. رکورد SIG حاوی رشته HASH امضاء شده توسط سرویس دهنده سطح بالای com. است که برای رکوردهای نوع A و KEY محاسبه شده تا بتوان اعتبار و هویت این رکوردها را بررسی کرد.

خواهد بود به دلیل فقدان اعتبار، این موضوع را کشف کند چرا که رکورد SIG (امضاء) با مجموعه RRSet مطابقت ندارد و غلط است.

با این وجود DNSsec یک مکانیزم مبتنی بر رمزنگاری برای تطبیق بسته های پاسخ متناظر با تقاضاهای ارسالی ارائه کرده است تا از پی ریزی حمله DNS Spoofing توسط ترویدی (مبتنی بر شکل ۸-۴۷) جلوگیری شود. در این مکانیزم اختیاری (که antispoof نام دارد) به هر بسته پاسخ یک رشته Hash اضافه می شود که این Hash از بسته تقاضا استخراج شده و سپس با کلید خصوصی پاسخ دهنده، امضا (رمز) می شود. از آنجایی که ترویدی کلید خصوصی سرویس دهنده سطح بالای com را نمی داند لذا قطعاً نخواهد توانست پاسخ ارسالی توسط این سرویس دهنده به ISP آلیس را جعل نماید. البته او می تواند بسته جعلی خود را برای آلیس بفرستد ولیکن به دلیل امضای اشتباهی که دارد، حذف خواهد شد.

DNSsec از چندین رکورد دیگر نیز حمایت می کند. به عنوان مثال رکورد CERT می تواند برای ذخیره و نگهداری گواهی نامه هایی مثل X.509 مورد استفاده قرار بگیرد. این رکورد بدان دلیل عرضه شده که برخی افراد علاقمندند DNS خود را در ساختار PKI<sup>۱</sup> قرار بدهند؛ اگرچه هنوز در عمل چنین کاری انجام نشده است. در اینجا توضیحات خود در خصوص DNSsec را به پایان می رسانیم. برای تفصیل بیشتر از RFC 2535 کمک بگیرید.

#### «اسامی خود گواهی» (Self-Certifying Names)

استفاده از سرویس دهنده امن DNS (DNSsec) تنها راه اطمینان کردن به اسامی نیست. یک راهکار کاملاً متفاوت در Secure File System (سیستم مطمئن فایل) بکار گرفته می شود. (Mazieres et al., 1999) پژوهشگران این پروژه، یک سیستم فایل مطمئن، جهانی و قابل گسترش طراحی کردند، بدون آنکه نیاز به تغییری در DNS باشد و حتی بدون استفاده از گواهی نامه های دیجیتالی یا بدون وجود هیچگونه PKI کار کنند. در این بخش نشان خواهیم داد که چگونه می توان نظریات آنان را در وب اعمال کرد. بر این اساس، در توضیحات این بخش به جای استفاده از اصطلاحات و واژه های بکار رفته در مقاله آنان، از اصطلاحات وب استفاده شده است؛ ولیکن برای احتراز از پیچیده شدن احتمالی بحث، باید اشاره کرد که این ساختار هنوز در سیستم وب بکار گرفته نشده ولی برای رسیدن به امنیت بالا در وب، این روش ممکن و مناسب است اگرچه قبل از معرفی و بکارگیری، باید تغییراتی اساسی در نرم افزارهای مرتبط با وب ایجاد شود.

برای شروع فرض می کنیم که هر سرویس دهنده وب دارای یک جفت کلید عمومی و خصوصی است. دلیل این فرض آنست که در هر URL یک رشته Hash برای بررسی صحت نام سرویس دهنده و صحت کلید عمومی آن، (به عنوان بخشی از URL) درج می شود. به عنوان مثال در شکل ۸-۴۹، URL یک فایل تصویر متعلق به باب را می بینیم. این آدرس طبق معمول با گزینه http شروع شده که نام پروتکل<sup>۲</sup> را تعیین می کند، سپس در ادامه، آدرس DNS سرویس دهنده باب (یعنی www.bob.com) درج شده است. بعد از آن یک علامت کالون (:) و سپس ۳۲ کاراکتر به عنوان رشته Hash اضافه شده است. در انتهای این رشته، باز هم طبق روش معمول، نام و موقعیت فایل می آید. به غیر از رشته Hash، بقیه URL استاندارد و معمولی است. با وجود رشته Hash، این URL اصطلاحاً «خود گواهی» می شود؛ (خودش صحت خود را تأیید می کند).

سؤال بدیهی آن است که: رشته Hash برای چیست؟ پس از الحاق نام نمادین و کلید عمومی یک سرویس دهنده به یکدیگر، الگوریتم SHA-1 بر روی آن اعمال می شود تا رشته ۱۶۰ بیتی Hash بدست آید. در

Server	SHA-1 (Server, Server's Public key)	File name
http://www.bob.com:2g5hd8bfjkc7mf6hg8dgany23xds4pe6/photos/bob.jpg		

شکل ۸-۴۹. یک «URL خودگواهی» شامل رشته HASH از نام و کلید عمومی سرویس دهنده.

الگوی فوق رشته Hash به صورت دنباله ای از ۳۲ رقم و حروف کوچک نشان داده شده است با این استثنا که از حروف 'l' و 'o' و ارقام '1' و '0' به دلیل مشابهت در شکل نمایش و احتمال خطا در حین وارد کردن URL صرف نظر شده است، بنابراین جمعاً ۳۲ حرف و رقم باقی می ماند؛ (۲۴ حرف کوچک و ۸ رقم). با این سی و دو کاراکتر می توان پنج بیت را کدگذاری کرد. بدین ترتیب با یک رشته ۳۲ کاراکتری می توان جمعاً ۱۶۰ بیت رشته SHA-1 را کد نمود؛ (۳۲ کاراکتر هر کاراکتر معادل ۵ بیت) البته در واقع، الزامی در استفاده از Hash نیست و می توان به جای آن خود کلید عمومی را قرار داد؛ حُسن استفاده از رشته Hash آنست که طول نام (که رشته Hash را در بر می گیرد) کاهش یابد.

ساده ترین روش (ولی در عین حال نامناسب ترین روش از لحاظ سهولت) برای مراجعه به فایل تصویر باب، آنست که آلیس رشته شکل ۸-۴۹ را در مرورگر خود درج نماید. مرورگر پیامی برای سرویس دهنده وب متعلق به باب فرستاده و کلید عمومی او را مطالبه می کند. وقتی کلید عمومی باب دریافت شد مرورگر، نام سرویس دهنده و کلید عمومی را بهم چسبانده و الگوریتم Hash را بر روی آن اعمال می کند. اگر رشته Hash حاصل شده با رشته Hash سی و دو کاراکتری در URL مطابقت داشت، مرورگر مطمئن خواهد شد که کلید عمومی ارسالی واقعاً متعلق به باب است. حتی اگر ترویدی بتواند در میانه راه این تقاضا را دریافت کرده و یک پاسخ جعلی برگرداند نمی تواند هیچ کلید عمومی که رشته Hash آن با رشته درج شده در URL مطابقت داشته باشد پیدا کند لذا هرگونه دخالت او در جعل پاسخ، به سادگی کشف خواهد شد. پس از دریافت کلید عمومی باب، می توان آن را برای استفاده های آتی در حافظه نهان (cache) ذخیره کرد.

حال آلیس باید بررسی کند که آیا باب واقعاً کلید خصوصی خود را (متناظر با کلید عمومی ارسال شده) در اختیار دارد یا خیر؟ او یک پیام می سازد که در برگیرنده یک کلید نشست AES، یک عدد تصادفی (nonce) و یک مهر زمان است. سپس او این پیام را با کلید عمومی باب رمز کرده و آن را برای او می فرستد. از آنجایی که فقط باب کلید خصوصی متناظر با این کلید عمومی را در اختیار دارد بنابراین تنها او قادر به رمزگشایی آن و بازگرداندن عدد تصادفی (nonce) با کلید AES است. در صورتی که عدد تصادفی که با کلید AES رمز و برگردانده شده با آنچه که توسط آلیس ارسال شده یکسان باشد، او می تواند مطمئن شود که باب صحبت می کند. همچنین در اینجا آلیس و باب دارای یک کلید نشست AES مشترک هستند که می توانند برای تقاضاهای GET بعدی و پاسخهای ارسالی از آن استفاده کنند.

پس از آن که آلیس فایل تصویر متعلق به باب (یا هر صفحه وب دیگر) را بدست آورد می تواند آدرس URL آن را در دفترچه یادداشت مرورگر خود ذخیره نماید، بدین ترتیب او مجبور نخواهد بود در آینده، URL کامل را تایپ کند. بعلاوه URL های جاسازی شده در درون صفحات وب نیز می توانند از نوع «خودگواهی» باشند، بنابراین فقط با یک کلیک می توان از آنها به صورت معمولی استفاده کرد؛ البته بشرطی که صفحه وب برگشتی واقعی و دست نخورده باشد. یک روش دیگر برای آن که نیازی به درج URL های «خودگواهی» نباشد آنست که این اسامی را از طریق یک سرویس دهنده مورد اعتماد (که دارای گواهینامه X.509 امضا شده توسط CA<sup>۱</sup> باشد) و

با برقراری یک اتصال امن (Secure Connection) دریافت نمائیم.

روش دیگر برای دریافت URLهای خودگواهی آنست که با وارد کردن «URL خودگواهی» یک موتور جستجوی مورد اعتماد و مطمئن (فقط برای یکبار) به آن متصل شده و به روش تشریح شده در بالا یک ارتباط مطمئن و احراز هویت شده با آن موتور جستجوی امن برقرار نمائیم. حال می توان این موتور جستجو را در خصوص یک URL مورد سؤال قرار داد و طبقاً نتیجه کار، یک صفحه وب امضاء شده خواهد بود که درون آن تمام URLها از نوع «خودگواهی» هستند و می توان بر روی آنها کلیک کرد، بدون آن که نیازی به درج رشته های طولانی باشد.

حال ببینیم این روش چگونه می تواند در مقابل حمله DNS Spoofing که توسط ترویدی انجام می شود مقاومت نماید: اگر ترویدی به نحوی برنامه ریزی کند تا حافظه نهان ISP متعلق به آلیس آلوده شود، تقاضای آلیس برای دریافت یک صفحه وب، به جای آن که توسط باب دریافت شود تحویل ترویدی خواهد شد. حال طبق پروتکل فوق، مرورگر آلیس در اولین پیام، از ترویدی می خواهد که کلید عمومی خود را ارائه بدهد. اگر ترویدی کلید عمومی خود را برگرداند، آلیس فوراً حمله را کشف خواهد کرد چراکه رشته Hash موجود در URL خودگواهی (که مبتنی بر کلید عمومی باب است)، با رشته Hash محاسبه شده به روش SHA-1، منطبق نیست. اگر ترویدی کلید عمومی باب را ارائه بدهد، آلیس قادر نخواهد بود حمله را کشف کند ولیکن از آنجایی که پیامهای بعدی را براساس کلید عمومی باب رمزنگاری و ارسال می کند لذا ترویدی پس از دریافت این پیامها هیچ راهی برای رمزگشایی و استخراج کلید AES و عدد تصادفی درون پیام نخواهد داشت. در اینجا اگرچه حمله DNS Spoofing سودی برای ترویدی (جهت بهره برداری از اطلاعات ارسالی) ندارد ولیکن می تواند برای حمله «اخلال در سرویس دهی» (Denial of Service) مؤثر باشد، زیرا ترویدی کاری کرده که آلیس نتواند با باب ارتباط برقرار کند.

### ۳-۹-۸ SSL: لایه سوکت های امن

نامگذاری مطمئن اگرچه شروع خوبی است ولیکن برای امنیت وب امکانات بسیار زیادتری وجود دارد. گام بعدی در امنیت وب برقراری اتصال مطمئن (Secure Connection) است. حال ببینیم اتصالات امن چگونه بوجود می آیند. در بدو زمانی که وب در صحنه اجتماع پدیدار گردید، از آن فقط برای توزیع صفحات وب ایستا استفاده می شد. با این وجود در اندک زمانی، بسیاری از شرکتها به صرافت افتادند تا از وب برای معاملات اقتصادی مثل داد و ستد کالا با کارتهای اعتباری، بانکداری Online و خرید و فروش الکترونیکی سهام استفاده کنند. این نوع کاربردها نیاز به ایجاد «اتصال امن» (Secure Connection) را دامن زدند. در سال ۱۹۹۵ شرکت نت اسکپ (Netscape) عرضه کننده مرورگر مهم دنیا، با معرفی یک بسته امنیتی به نام SSL (Secure Socket Layer) به این نیاز پاسخ داد. امروزه از این نرم افزار و پروتکل معرفی شده آن، بطور فزاینده ای استفاده می شود و حتی مرورگر IE مایکروسافت نیز از آن حمایت می کند، لذا بجاست که آنرا با اندکی شرح بیشتر بررسی کنیم.

«SSL یک اتصال مطمئن با مشخصات زیر، بین دو سوکت ایجاد می نماید:

۱. امکان مذاکره و توافق پارامترها بین سرویس دهنده و مشتری
۲. احراز هویت سرویس دهنده و مشتری بطور مستقل و مجزا
۳. مخابره سازی و رمزنگاری شده داده ها
۴. مراقبت از صحت و سلامت داده ها

۱. صفحاتی که فقط ارائه دهنده اطلاعات هستند و هیچگونه دریافت اطلاعات ندارند. سم

قبلاً هر یک از موضوعات فوق را به صورت جداگانه بررسی کرده ایم و نیازی به شرح مبسوط آنها نیست. محل قرار گرفتن لایه SSL در پشته پروتکلی TCP/IP، در شکل ۸-۵۰ نشان داده شده است. در حقیقت SSL یک لایه جدید بین لایه کاربرد و لایه انتقال است که تقاضاهای مرورگر را گرفته و آنها را از طریق لایه TCP برای سرویس دهنده ارسال می کند. پس از آن که یک اتصال مطمئن ایجاد شد، مهمترین وظیفه SSL، انجام عملیات فشرده سازی و رمزنگاری اطلاعات (و بالعکس) خواهد بود. وقتی از HTTP بر روی SSL استفاده می شود، اصطلاحاً آنرا HTTPS (Secure HTTP) می نامند، هرچند هیچ تفاوتی با پروتکل استاندارد HTTP ندارد؛ فقط در لایه زیرین تغییراتی ایجاد شده است. البته برخی اوقات به جای پورت استاندارد ۸۰، HTTPS بر روی پورت جدید ۴۴۳ در دسترس قرار می گیرد. از آنجایی که SSL یک لایه مستقل بر روی TCP است لذا استفاده از آن به مرورگرهای وب محدود نشده و دیگر برنامه ها نیز می توانند از امکانات آن استفاده کنند ولیکن عمومی ترین کاربرد آن در وب است.

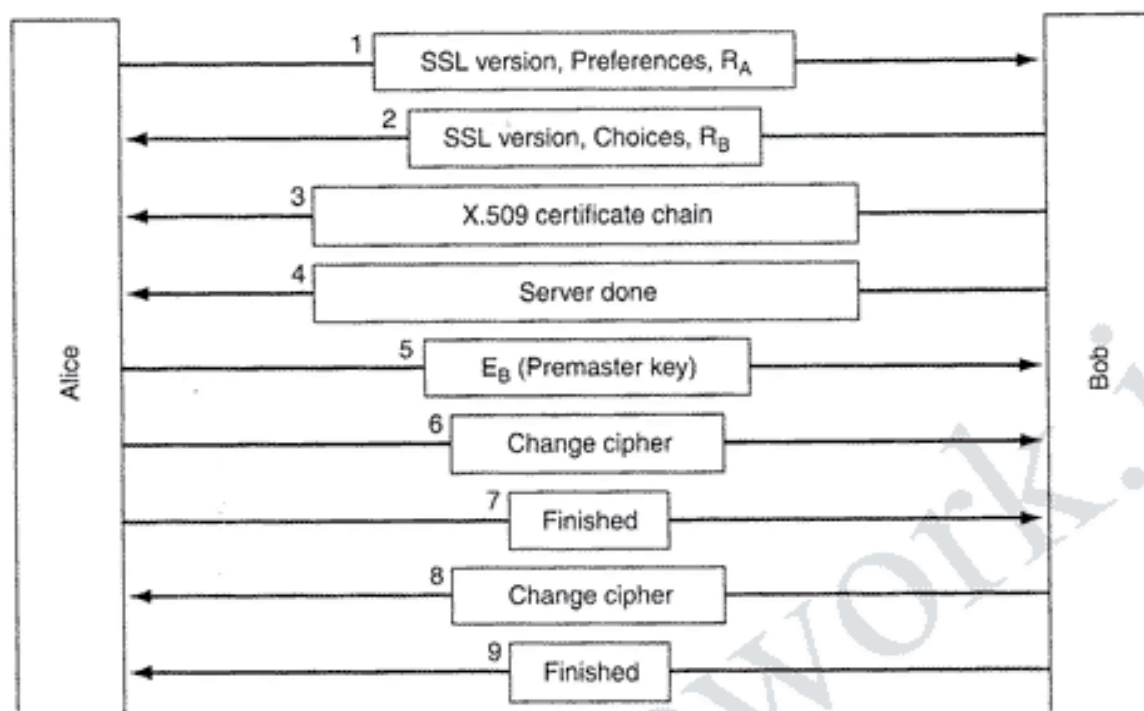
Application (HTTP)
Security (SSL)
Transport (TCP)
Network (IP)
Data link (PPP)
Physical (modem, ADSL, cable TV)

شکل ۸-۵۰. لایه ها (و پروتکلها) برای یک کاربر خانگی که از طریق SSL به مرور وب می پردازد.

پروتکل SSL از چندین نسخه مختلف، گذر کرده و متکامل شده است. در زیر ما فقط نسخه ۳ آنرا بررسی می کنیم چراکه بطور فزاینده ای از آن استفاده می شود و نسخه های قبلی عملاً منسوخ شده اند. SSL از چندین الگوریتم و گزینه های مختلف حمایت می کند. گزینه ها شامل استفاده یا عدم استفاده از فشرده سازی، تعیین نوع الگوریتم رمزنگاری و مواردی در خصوص محدودیت صادرات محصولات مبتنی بر رمزنگاری است. آخرین مورد یعنی گزینه های مرتبط با محدودیتهای صادرات بدان منظور بوده تا از امکانات پیشرفته رمزنگاری (که طبق قوانین ایالات متحده، صدور آنها به خارج محدود است) فقط زمانی استفاده شود که طرفین یک اتصال، صرفاً در ایالات متحده مستقر باشند. در بقیه موارد طول کلید به ۴۰ بیت محدود شده است که بسیاری از رمزنگاران این طول کلید را یک شوخی تلقی می کنند. شرکت نت اسکپ برای آن که بتواند مجوز صدور محصول خود را از دولت ایالات متحده اخذ کند مجبور بود این محدودیت را اعمال نماید.

SSL از دو «زیر پروتکل» (subprotocol) تشکیل شده است: یکی برای ایجاد اتصال امن و دیگری برای بکارگیری از آن جهت مبادله اطلاعات. اجازه بدهید در ابتدا ببینیم که چگونه اتصالات امن بوجود می آید. زیر پروتکل ایجاد اتصال، در شکل ۸-۵۱ نشان داده شده است. وقتی آلیس تقاضای برقراری یک ارتباط امن با باب را می دهد، این زیر پروتکل کار خود را با ارسال پیام ۱ شروع می کند. این پیام نسخه SSL مورد استفاده توسط آلیس و همچنین گزینه های انتخابی او نظیر تمایل به فشرده سازی و الگوریتم مورد نظر او جهت رمزنگاری را مشخص می کند. این پیام همچنین شامل یک عدد تصادفی بزرگ  $R_A$  است که به عنوان رشته چالش (nonce) بعداً از آن استفاده خواهد شد.

حال نوبت باب است. در پیام ۲، باب از بین گزینه های پیشنهادی که آلیس از آنها حمایت می کند، گزینه هایی را انتخاب کرده و این انتخابها را به همراه عدد تصادفی  $R_B$  و نسخه SSL مورد استفاده، برای آلیس می فرستد. سپس



شکل ۸-۵۱. نسخه ساده شده از زیرپروتکل برقراری اتصال در SSL.

در پیام سوم باب گواهینامه دیجیتالی خود را که در آن کلید عمومی او درج شده، برای آلیس می‌فرستد. اگر این گواهینامه توسط یک مرکز معتبر، امضا نشده باشد، باب زنجیره‌ای از گواهینامه‌ها را که نهایتاً به یک مرکز معتبر ختم می‌شود، برای آلیس می‌فرستد. تمام مرورگرها و از جمله مرورگر آلیس، دارای حدود صد کلید عمومی از مراکز شناخته شده و مجاز گواهی امضاء هستند که باب باید زنجیره گواهینامه‌ها را به نحوی برای آلیس بفرستد تا نهایتاً به یکی از این صد مرکز ختم شود.<sup>۱</sup> بدین ترتیب آلیس قادر خواهد بود تا صحت کلید عمومی باب را بررسی نماید. در اینجا باب ممکن است پیامهای دیگری برای آلیس ارسال کند (مثل تقاضای گواهینامه دیجیتالی آلیس). پس از انجام این عملیات او پیام ۴ را برای آلیس فرستاده و به او اعلام می‌کند که نوبت اوست.

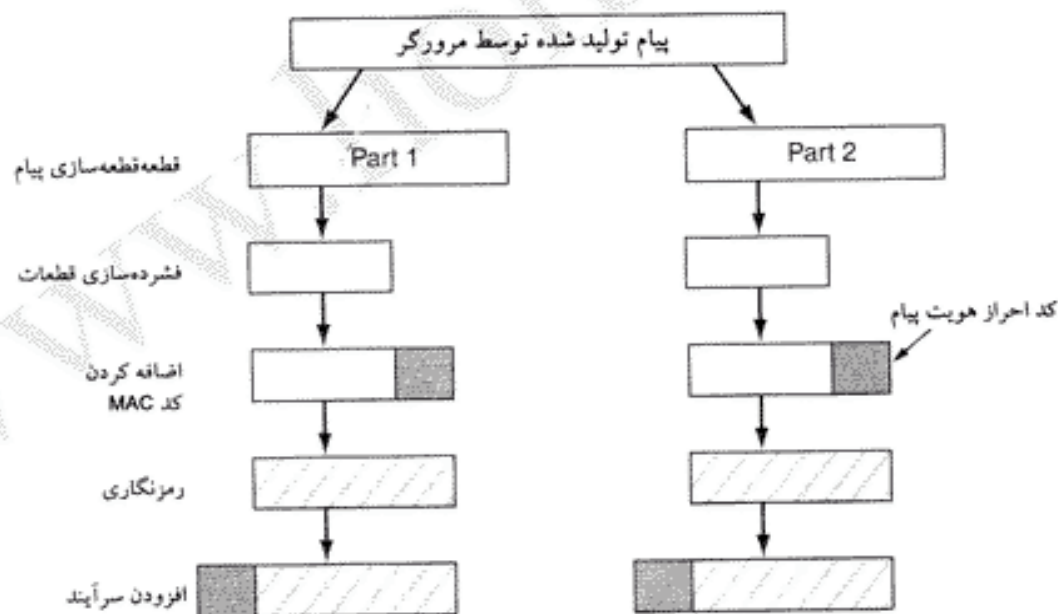
آلیس در پیام ۵، با انتخاب یک «شاه کلید» اولیه ۳۸۴ بیتی و ارسال آن برای باب به او پاسخ می‌دهد در حالی که این شاه کلید با کلید عمومی باب رمزنگاری شده است. کلید اصلی نشست که بعداً برای رمزنگاری اطلاعات استفاده خواهد شد از ترکیب شاه کلید و  $R_A$  و  $R_B$  به روش پیچیده‌ای استخراج می‌گردد. پس از دریافت پیام ۵، آلیس و باب هر دو قادرند کلید اصلی نشست را محاسبه نمایند. به همین دلیل آلیس در پیام ۶ به باب اعلام می‌کند که از سیستم رمز جدید (با کلید جدید) استفاده کند. در پیام ۷، خاتمه زیرپروتکل ایجاد اتصال اعلام می‌شود. باب نیز در پاسخ، آلیس را تایید و ختم زیرپروتکل را اعلام می‌کند. (پیامهای ۸ و ۹)

در اینجا اگرچه آلیس از باب مطمئن است ولی باب نمی‌داند که آلیس کیست (مگر آن که آلیس دارای یک کلید عمومی و یک گواهینامه دیجیتالی معتبر باشد). بنابراین اولین پیامی که ممکن است باب پس از ایجاد اتصال برای آلیس بفرستد آن است.<sup>۲</sup> از او تقاضا کند با ارائه نام ورود و کلمه عبور، Login نماید. پروتکل لازم برای عملیات Login، خارج از محدوده این کتاب است. پس از تکمیل عملیات Login، انتقال اطلاعات می‌تواند آغاز شود. به گونه‌ای که قبلاً اشاره شد، SSL از چندین الگوریتم رمزنگاری حمایت می‌کند. در قدرتمندترین روش،

۱. برای آشنایی با زنجیره گواهینامه‌های دیجیتالی به بخش ۸-۵-۳ مراجعه کنید

برای رمزنگاری از triple DES با سه کلید مجزا و برای بررسی صحت پیامها از SHA-1 استفاده می شود. ترکیب این دو پروتکل نسبتاً کند عمل می کند و اغلب برای عملیات بانکداری یا کاربردهایی که در آنها امنیت بالا حیاتی است مورد استفاده قرار می گیرد. برای عملیات معمولی تجارت الکترونیکی، از رمزنگاری RC4 با کلید ۱۲۸ بیتی و روش MD5 برای تأیید صحت اطلاعات استفاده می شود. RC4 کلید ۱۲۸ بیتی را به عنوان نقطه شروع گرفته و برای استفاده در الگوریتم، آنرا به یک عدد بزرگ بسط می دهد و سپس از آن برای تولید Keystream بهره می گیرد. این Keystream (به نحوی که در بخش ۸-۲-۳ تشریح شد) با داده های ارسالی، XOR می شود تا متن رمز شده (به روش Stream Cipher) بدست بیاید؛ (شکل ۸-۵۲). نسخه صادراتی SSL نیز از کلیدهای ۱۲۸ بیتی RC4 استفاده می کند ولیکن ۸۸ بیت از آن عمومی و آشکار است تا بتوان رمز آن را براحتی شکست. (زیرا صدور محصولات SSL از ایالات متحده، فقط با کلید ۴۰ بیتی مجاز است).

به نحوی که در شکل ۸-۵۲ نشان داده شده، برای انتقال واقعی اطلاعات، از زیرپروتکل دوم SSL استفاده می شود. پیامهای ارسالی از مرورگر، ابتدا به قطعات ۱۶ کیلوبایتی شکسته می شود. اگر گزینه فشرده سازی فعال شده باشد، هر قطعه به صورت مستقل فشرده می شود. سپس یک کلید سری که از ترکیب دو عدد تصادفی (nonce) و شاه کلید اولیه بدست می آید (موقتاً) به متن فشرده شده ضمیمه می شود و براساس الگوریتم مورد توافق (معمولاً MD5)، از آن یک رشته Hash استخراج می گردد. این رشته Hash به عنوان MAC<sup>۱</sup> (کد تأیید صحت پیام) به قطعه فشرده شده ضمیمه می شود. سپس مجموعه این دو با استفاده از الگوریتم متقارن مورد توافق (معمولاً با XOR کردن داده ها با Keystream های مبتنی بر الگوریتم RC4)، رمزنگاری می شود. نهایتاً به قطعه حاصل یک سرآیند اضافه شده و نتیجه بر روی اتصال TCP ارسال می گردد.



شکل ۸-۵۲. انتقال داده با استفاده از SSL.

در استفاده از RC4 یک مورد احتیاط وجود دارد: از آنجایی که اثبات شده در الگوریتم RC4، کلیدهای ضعیف و نامناسبی وجود دارند که از طریق آنها می توان رمز متن را شکست، لذا امنیت SSL وقتی که در آن از RC4 استفاده شده بسیار متزلزل و شکننده است. مرورگرهایی به کاربر اجازه می دهند شخصاً الگوریتم مورد

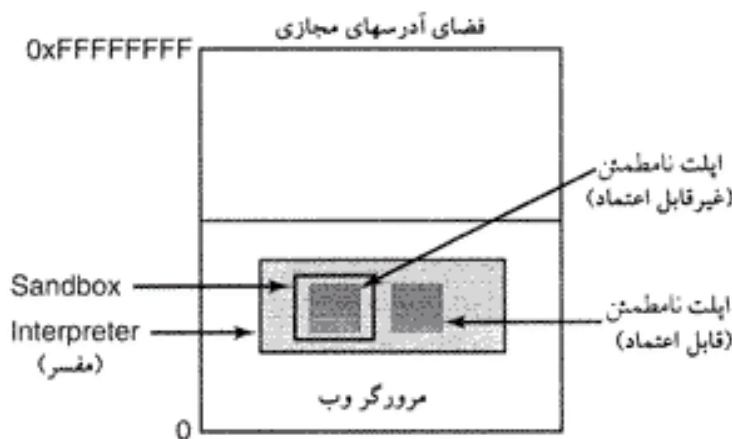
نظرش را انتخاب نماید باید به نحوی پیکربندی شوند تا همیشه از روش رمزنگاری Triple DES با کلیدهای ۱۶۸ بیتی و روش SHA-1 استفاده کنند اگرچه این ترکیب بسیار کندتر از ترکیب RC4 و MD5 عمل می کند. اشکال دیگر SSL آن است که طرفین ارتباط ممکن است گواهینامه دیجیتالی نداشته باشند یا حتی در صورت دارا بودن گواهینامه، همیشه صحت و تطابق کلیدهای مورد استفاده را بررسی نکنند. در سال ۱۹۹۶ شرکت نت اسکایپ، SSL را برای استانداردسازی به IETF ارائه کرد. نتیجه کار TLS (Transprot Layer Security) بود که در RFC 2246 تشریح شده است. تغییراتی که در SSL ایجاد شد نسبتاً کم بود ولیکن همین تغییرات ناچیز کافی بود تا SSL نسخه ۳ با TLS سازگار نباشد. به عنوان مثال روشی که براساس آن از شاه کلید اولیه و اعداد تصادفی (nonce)، کلید نشست استخراج می شود تغییر داده شد تا کلید قوی تر و محکم تر شود. (رمزشکنی آن سخت تر شود). TLS گاهی به عنوان نسخه 3.1 از SSL نیز نامیده می شود. اولین پیاده سازی TLS در سال ۱۹۹۹ عرضه شد ولی هنوز مشخص نیست که عملاً چه زمانی جایگزین SSL خواهد شد، هرچند از SSL نسبتاً قویتر است. البته اشکال ضعیف بودن کلیدهای RC4 هنوز هم در TLS وجود دارد.

### ۸-۹-۴ امنیت کدهای متحرک

«نامگذاری مطمئن» و «ایجاد اتصالات امن»، دو زمینه مهم و مرتبط با امنیت وب است ولی هنوز موارد متعدد دیگری وجود دارد. در روزهای اولیه، صفحات وب فقط فایل های HTML ایستا بودند و درون آنها هیچگونه کد اجرایی وجود نداشت. امروزه اغلب صفحات وب، در برگیرنده برنامه های کوچک اجرایی شامل اپلت های جاوا، کنترلرهای Active X و اسکریپت های جاوا هستند. دریافت و اجرای چنین کدهای متحرکی، خطرات امنیتی بسیار عظیمی دارد لذا باید روشهای مختلفی طراحی و ابداع شود تا این مخاطرات کاهش یابد. در این بخش برخی از موارد مرتبط با کدهای متحرک و روشهای برخورد با مخاطرات آن را بطور اجمالی مرور خواهیم کرد.

#### امنیت اپلت های جاوا

اپلت های جاوا، برنامه های کوچک جاوا هستند که برای اجرا بر روی یک «ماشین مجازی مبتنی بر پشته» (Stack Oriented) به نام JVM (Java Virtual Machine) ترجمه (کامپایل) می شوند. این برنامه ها می توانند درون یک صفحه وب قرار گرفته و به همراه آن صفحه، بارگذاری و اجرا شوند. به گونه ای که در شکل ۸-۵۳ نشان داده شده است پس از بارگذاری صفحه وب، اپلتها جهت اجرا تحویل مفسر JVM (تعبیه شده در درون مرورگر) می شوند.



شکل ۸-۵۳. اپلتها می توانند توسط مرورگر تفسیر و اجرا شوند.

یک مزیت اجرای کدهای مفسری (Interpreted Code) در مقایسه با کدهای ترجمه شده (Compiled Code) آن است که هر دستورالعمل قبل از اجرا توسط مفسر بررسی می شود. این قابلیت، فرصت بررسی اعتبار آدرس هر دستورالعمل را به مفسر فرمان می دهد. به علاوه هرگونه فراخوانی سیستمی (System Call) توسط مفسر دریافت و قبل از اجرا تفسیر می شود. چگونگی برخورد با این فراخوانی های سیستمی، جوهره سیاستهای امنیتی را تشکیل می دهد. به عنوان مثال هرگاه یک اپلت قابل اعتماد باشد (از دیسک محلی خود کامپیوتر دریافت و اجرا شده باشد) هرگونه فراخوانی سیستمی بدون چون و چرا اجرا می شود ولیکن اگر اپلت مورد اعتماد نباشد (از طریق اینترنت دریافت شده باشد) باید در یک شرایط کاملاً بسته و حفاظت شده (که Sandbox نامیده می شود) اجرا گردد تا بتوان بر رفتار آن، اعمال محدودیت کرد و از هرگونه تلاش برای دسترسی به منابع سیستم جلوگیری شود.

هرگاه یک اپلت تلاش کند تا از منابع سیستمی ماشین بهره بگیرد، فراخوانی سیستمی آن اپلت، به منظور تائید بلافاصله تحویل یک «ناظر امنیت» (Security Monitor) می شود. «ناظر امنیت» براساس سیاستهای امنیتی که به صورت محلی تدوین شده تصمیم می گیرد که آیا باید به آن فراخوانی اجازه اجرا بدهد یا آنرا رد کند. بدین ترتیب فقط امکان دسترسی به برخی از منابع سیستمی وجود دارد نه همه آنها. متأسفانه حقیقت آنست که این مدل امنیتی بد عمل می کند و اشکالات آن همیشه به ناگاه پدیدار می شود.

#### Active X

کنترل های اکتیواکس، برنامه های اجرایی و دودویی برای ماشینهای پنتیوم هستند که می توانند درون صفحات وب جاسازی شوند. وقتی مرورگر به یکی از آنها بر می خورد، ابتدا بازرسی و آزمایشهایی انجام می شود تا ببیند آیا قابل اجرا است؛ در صورتی که بتواند این آزمون را با موفقیت بگذراند اجرا می شود. این کنترلها به هیچ وجه تفسیر (Interpret) و نظارت نمی شوند لذا به اندازه برنامه معمولی کاربران قدرت اجرایی دارند و می توانند آسیبهای بالقوه خطرناکی را به سیستم تحمیل کنند.

روشی که شرکت مایکروسافت برای تصمیم گیری در خصوص اجرا یا عدم اجرای کنترل های اکتیواکس برگزیده است مبتنی بر روشی به نام «امضای کد» (Code Signing) است. هر کنترل اکتیواکس یک امضای دیجیتالی با خود به همراه دارد که در حقیقت یک رشته Hash از کد اجرایی است که توسط کلید خصوصی طراح آن، رمزنگاری و امضاء می شود. وقتی یک کنترل اکتیواکس ظاهر می شود، مرورگر ابتدا امضای آن را بررسی می کند تا مطمئن شود که در خلال انتقال دستکاری نشده است. اگر امضای آن صحیح بود مرورگر جدول داخلی خود را بررسی می کند تا ببیند آیا پدیدآورنده آن برنامه قابل اعتماد است؟ یا آن که زنجیره گواهی نامه های آن، به یک تولیدکننده مورد اعتماد ختم می شود یا خیر؟ اگر پدیدآورنده اکتیواکس مورد اعتماد باشد، برنامه اجرا می شود، در غیر این صورت اجرا نخواهد شد. سیستمی که مایکروسافت برای بررسی کنترل های اکتیواکس پیاده کرده به نام Authenticode مشهور است.

مقایسه روشهای بکار رفته در جاوا و اکتیواکس مفید خواهد بود. در روش بکار رفته در جاوا هیچ تلاشی برای آن که مشخص شود چه کسی اپلت را نوشته، انجام نمی شود. در عوض یک «مفسر زمان اجرا»، این اطمینان را فراهم می آورد که اپلت کارهایی را که صاحب ماشین اجازه نداده، انجام نمی دهد. برعکس، در اکتیواکس با اطمینان به امضای کدهای اجرایی، بر عملکرد و رفتار کد در حال اجرا نظارت نمی شود. اگر آن برنامه از منبع قابل اعتماد دریافت شده و در حین گذر از شبکه دستکاری نشده باشد فوراً اجرا می شود. هیچ تلاشی نیز برای بررسی اینکه آیا کدهای برنامه مخرب هستند یا نه انجام نمی گیرد. اگر برنامه نویسی اصلی اکتیواکس (که دارای گواهی نامه معتبر است و مورد اعتماد تلقی می شود)، در نظر داشته باشد که کد او کل دیسک سخت ماشین را نابود کرده و سپس

Flash ROM کامپیوتر را پاک کند تا دیگر بوت نشود، گداو بی چون و چرا اجرا شده و کل کامپیوتر را نابود خواهد کرد. (مگر آن که گزینه Active X در تنظیمات مرورگر غیرفعال شده باشد.)

بسیاری از افراد در خصوص اعتماد به شرکتهای ناشناخته تولید نرم افزار نگران و بدبین هستند. یک برنامه نویس در سیاتل آمریکا برای آن که این اشکال را نشان بدهد یک شرکت نرم افزاری بوجود آورد و یک گواهینامه اعتماد برای خود تهیه کرد؛ انجام این کار ساده است. سپس یک کنترل اکتیواکس نوشت که کامپیوتر را خاموش (shutdown) می کرد. او این برنامه را در سطح وسیعی در اینترنت منتشر ساخت. این اکتیواکس ماشینهای زیادی را خاموش کرد ولی هیچ آسیبی به آنها نرساند و بلافاصله می شد ماشین را از نو راه اندازی نمود. واکنش رسمی به کار او آن بود که گواهینامه اش را برای این کنترل اکتیواکس خاص باطل کردند و بدین نحو به این داستان خجالت آور پایان داده شد ولیکن هنوز زمینه بروز چنین مشکلاتی برای سوء استفاده برنامه نویسان شرور وجود دارد. (Garfinkel with Spafford, 2002) از آنجایی که هیچگاه نمی توان بر هزاران شرکت تولید نرم افزار که ممکن است کدهای متحرک بنویسند نظارت کرد، لذا روش امضای کدهای اجرایی، فاجعه ای است که هر لحظه در شرف وقوع خواهد بود.

#### اسکرپتهای جاوا

اسکرپتهای جاوا ذاتاً دارای هیچ مدل امنیتی رسمی و شناخته شده ای نیستند و پیاده سازیهای ناامن و نفوذپذیر آنها پیشینه ای طولانی دارد. هر تولیدکننده نرم افزار از یک دیدگاه خاص به امنیت می پردازد. به عنوان مثال نسخه دوم از مرورگر نت اسکپ از مدلی شبیه به مدل جاوا استفاده می کند (یعنی مبتنی بر نظارت و کنترل فراخوانیها)، در حالی که در نسخه چهارم به سمت مدل امضای کدها حرکت کرده است.

مشکل اساسی آنست که اجازه اجرای کدهای بیگانه و ناشناخته بر روی ماشین شما، نوعی خوشامدگویی به دردسر و گرفتاری است. از دیدگاه امنیت، این کار همانند آن است که یک دزد را به خانه خود دعوت کرده و سپس تلاش کنید او را به دقت تحت نظر بگیرید مبادا از آشپزخانه به اتاق نشیمن فرار کند. اگر اتفاق غیرمنتظره ای بیفتد یا آن که برای لحظاتی غفلت کنید، حوادث ناخوشایندی می تواند رخ بدهد. گرایش به کدهای اجرایی و متحرک مثل اسکرپتها از آنجایی ناشی می شود که این کدها امکان نمایش گرافیکهای متحرک و تعاملات سریعتر و بهینه تری را فراهم می کنند و بسیاری از طراحان وب سایت وب فکر می کنند وجود این امکانات از امنیت مهمتر است بالاخص وقتی ماشین دیگران در معرض خطر باشد!!

#### ویروسها

ویروسها نوع دیگری از کدهای متحرک هستند. برخلاف مثالهای فوق، هیچکس ویروسها را به ماشین خود دعوت و منتقل نمی کند. تفاوت بین یک ویروس و کدهای متحرک معمولی آن است که ویروسها به گونه ای نوشته می شوند که خود را تکثیر کنند. وقتی یک ویروس از طریق صفحات وب، ضمیمه های نامه الکترونیکی یا روشهای دیگر به یک ماشین می رسد، کار خود را با آلوده کردن برنامه های اجرایی روی دیسک آغاز می کند، وقتی یکی از این برنامه ها اجرا شود، کنترل اجرا به ویروس منتقل شده و آن ویروس مجدداً تلاش می کند به روشهایی مثل ارسال کپی خودش از طریق پست الکترونیکی (به آدرسهای که قربانی در کامپیوترش یادداشت کرده)، خود را تکثیر کند. برخی از ویروسها بوت سکتور دیسک سخت را آلوده می کنند به نحوی که وقتی ماشین بوت می شود، ویروس نیز بلافاصله اجرا می گردد. ویروسها به یک مشکل عمده برای شبکه اینترنت تبدیل شده اند و میلیاردها دلار خسارت به بار می آورند. شاید نسل جدید سیستمهای عامل که براساس تکنولوژی ریزهسته امن (Secure Microkernel) و تفکیک دقیق و محکم کاربران، پرونده ها و منابع بنا نهاده شده است به کاهش این مشکل کمک کند.

## ۱۰-۸ زمینه ها و پی آمدهای اجتماعی

اینترنت و تکنولوژی امنیت آن، موضوعی است که در آن موارد اجتماعی، سیاستهای عمومی و تکنولوژی با یکدیگر تلاقی می کنند و اغلب تبعات گسترده و عظیمی دارند. در زیر به اختصار سه موضوع را بررسی خواهیم کرد: «حریم خصوصی افراد» (Privacy)، «آزادی بیان» و «حقوق مالکیت معنوی» (Copy right). بدیهی است که فقط می توانیم این موارد را به صورت سطحی بررسی نماییم. برای مطالعه بیشتر در این خصوص به این مراجع مراجعه کنید: (Anderson, 2001; Garfinkel with Spafford, 2002; Schneier, 2000) در خصوص این موضوعات اینترنت سرشار از مطلب و مقاله است. فقط کافی است کلمات "Privacy" یا "Censorship" (سانسور) یا "Copyright" را در یک موتور جستجو، تایپ کنید و نتایج جستجو را دنبال نمایید. همچنین می توانید به وب سایت همین کتاب مراجعه کنید تا در این خصوص، چندین لینک به سایتهای مفید بدست بیاورید.

### ۱۰-۸-۱ حریم خصوصی افراد (Privacy)

آیا افراد از حق داشتن حریم خصوصی بهره مند هستند؟ سؤال بسیار خوبی است! در چهارمین اصلحیه قانون اساسی در ایالات متحده آمریکا آمده که حکومت بدون دلیل موجه و قانونی حق جستجوی منازل مردم، نوشته ها و آثار آنها را ندارد و شرایط صدور چنین مجوزی بسیار محدود و خاص در نظر گرفته شده است. لذا حداقل در ایالات متحده، بیش از ۲۰۰ سال است که حفظ حریم خصوصی افراد به یک قاعده عمومی تبدیل شده است.

در طول چند دهه گذشته چیزهایی تغییر کرده اند که هم کار حکومت در جاسوسی از شهروندان را ساده تر کرده و هم این امکان را برای شهروندان فراهم آورده که جلوی این جاسوسی را بگیرند! در قرن هجدهم، برای آن که دولت بتواند نوشته های یک نفر را تفتیش کند، مجبور بود که یک پلیس اسب سوار را به مزرعه آن شهروند بفرستد و مستندات خاصی را بررسی نماید. این روال بسیار پر دردسر بود. امروزه شرکتهای تلفن و ارائه کنندگان خدمات اینترنت در صورت ارائه مجوز لازم به سادگی امکان ایجاد انشعاب و استراق سمع را در اختیار می گذارند. این امکان، کار پلیسها را راحتتر کرده و خطر سقوط از اسب نیز وجود ندارد!!

رمزنگاری وضعیت را تغییر داده است. هر کسی که یک نسخه از بسته نرم افزاری PGP را دریافت و نصب کند و از کلید رمز مطمئن و مستحکم استفاده نماید می تواند اطمینان داشته باشد که هیچکس در این جهان قادر به خواندن نامه های او نخواهد بود. دولتها و حکومتها از این موضوع به خوبی آگاهند و هرگز خشنود نیستند. حفظ حریم خصوصی افراد به صورت واقعی، بدین معناست که جاسوسی افراد در موارد جنایی نیز بسیار دشوار خواهد بود. همچنین جاسوسی روزنامه نگاران و رقبای سیاسی بسیار سخت تر می شود. بدین دلیل برخی از دولتها بکارگیری یا صدور محصولات رمزنگاری را محدود یا حتی ممنوع کرده اند. به عنوان مثال در فرانسه تا قبل از سال ۱۹۹۹ هرگونه رمزنگاری اطلاعات ممنوع بود مگر آن که کلید رمز به دولت تسلیم شده باشد.

فرانسه تنها نبود. در آوریل ۱۹۹۳ دولت آمریکا اعلام کرد که در نظر دارد یک سخت افزار رمزنگار به نام Cliper Chip را به عنوان استاندارد برای تمام مخابرات شبکه بسازد. البته عنوان شده بود که حریم خصوصی شهروندان محترم شمرده می شود. همچنین اشاره شده بود که این تراشه عرضه شده توسط دولت قادر است تمام ترافیک داده ها را با استفاده از روشی به نام Key escrow رمزگشایی کند؛ این ساختار اجازه می داد تا دولت تمام کلیدهای رمز را در اختیار داشته باشد. با این وجود دولت قول داده بود که فقط با مجوز قانونی به جستجو و استراق سمع داده ها خواهد پرداخت. بدیهی است که خشم عموم برانگیخته شد، طرفداران حفظ حریم خصوصی شهروندان آن را شرم آور و ننگین دانستند در حالی که مجریان قانون آن را ستایش می کردند. عاقبت، دولت

عقب‌نشینی کرد و این نظریه را کنار گذاشت.

حجم بسیار زیادی اطلاعات مفید، در خصوص حریم الکترونیکی افراد در سایت متعلق به سازمان Electronic Frontier Foundation به آدرس [www EFF.org](http://www EFF.org) در دسترس عموم قرار دارد.

#### نامه‌پراکن‌های ناشناس (Anonymous Remailer)

SSL, PGP و تکنولوژیهای دیگر این امکان را فراهم آورده‌اند که دو طرف با یکدیگر به صورت امن و احراز هویت شده و فارغ از شنود یا دخالت شخص ثالث مبادله و مخابره اطلاعات داشته باشند. ولیکن گاهی حریم خصوصی افراد ایجاب می‌کند که احراز هویت وجود نداشته باشد یعنی افراد بتوانند به صورت ناشناس با یکدیگر در ارتباط باشند.

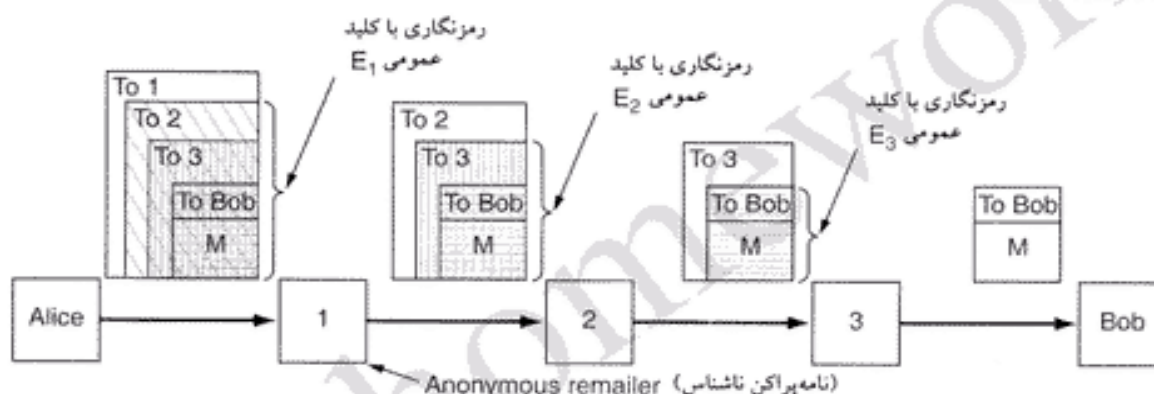
اجازه بدهید به چند مثال پردازیم: اول آن که مخالفان سیاسی که در سیطره رژیمهای استبدادی زندگی می‌کنند اغلب علاقه‌مندند به صورت ناشناس ارتباط برقرار کنند تا از دستگیری و نهایتاً کشته شدن در امان بمانند. دوم آن که خلافتکاری شرکتها، مؤسسات آموزشی، دولتی و دیگر سازمانها اغلب توسط افرادی فاش شده که می‌خواهند برای فرار از انتقام ناشناس بمانند. سوم، افرادی که دارای دیدگاههای اجتماعی، سیاسی یا مذهبی نامتداول یا مخالف هستند علاقه‌مندند از طریق پست الکترونیکی یا گروههای خبری بدون افشای هویت خود با یکدیگر ارتباط برقرار کنند. چهارم، شاید افرادی بخواهند مسائل خود در خصوص بیماریهای روانی و مواردی از این قبیل را به صورت ناشناس در گروههای خبری (newsgroup) بیان کنند. البته مثالهای بی‌شمار دیگری نیز وجود دارد. بیایید به یک مثال خاص پردازیم. در سال ۱۹۹۰ برخی از مخالفین یک اقلیت مذهبی، دیدگاههای خود را از طریق یک سیستم نامه‌پراکن ناشناس (Anonymous Remailer) برای یک گروه خبری در یوزنت ارسال کردند. این سرویس دهنده به کاربران اجازه می‌داد تا برای خود اسم مستعار انتخاب کرده و نامه‌های الکترونیکی خود را برای آن بفرستند؛ آن سرویس دهنده نیز نامه‌ها را با اسم مستعار برای علاقمندان ارسال می‌کرد؛ بدین ترتیب هیچکس نمی‌توانست بگوید که پیام از طرف چه کسی آمده است. در برخی از این مرسولات اطلاعاتی فاش شده بود که این اقلیت مذهبی بعداً ادعا کرد اسرار بازرگانی و اسناد دارای حق مالکیت (Copyright) آنها بوده است. سران این اقلیت مذهبی با گزارش به مراجع قانونی ادعا کردند که اسرار بازرگانی آنها فاش و مالکیت معنوی اسناد نقض شده است و هر دوی این موارد در محلی که این سرویس دهنده قرار داشت جرم محسوب می‌شد. مورد به دادگاه کشیده شد و اپراتور این سرویس دهنده وادار شد اطلاعات مربوط به فهرست اصلی افراد و اسامی مستعار را به دادگاه عرضه کند و بدین ترتیب هویت واقعی افرادی که با این سرویس دهنده مکاتبه‌ای داشتند مشخص شد. (این اولین باری نبود که یک گروه مذهبی از انتشار اسرار درونی خود بر می‌آشت؛ ویلیام تیندال در سال ۱۵۳۹ به دلیل ترجمه انجیل به زبان انگلیسی در آتش خشم سوخت).

بخش قابل ملاحظه‌ای از جامعه اینترنت از بابت نقض حریم خصوصی و افشای اسرار دیگران در ماجرای فوق به خشم آمد. نتیجه‌ای که عموم افراد از این ماجرا گرفتند آن بود که یک سرویس دهنده نامه‌پراکن ناشناس که در آن آدرسهای واقعی نامه‌های الکترونیکی و اسامی مستعار ذخیره می‌گردد (که سیستم نامه‌پراکن نوع یک Type 1 Remailer نامیده می‌شود) ارزش و قابلیت اعتماد چندانی ندارد. این موارد بسیاری از افراد را بر آن داشت تا سیستمهای نامه‌پراکن ناشناس را به گونه‌ای طراحی کنند که بتواند در مقابل حملاتی نظیر احضار و توقیف سرویس دهنده مقاومت کند.

این سیستمهای جدید نامه‌پراکن که اغلب Cypherpunk Remailer نامیده می‌شوند عملکردی شبیه به روال زیر دارند: کاربر پیام الکترونیکی خود را تولید و سرآیند استاندارد RFC 822 را بدان می‌افزاید (البته بدون گزینه From) و سپس آن را با کلید عمومی سرویس دهنده نامه‌پراکن رمز می‌کند و نهایتاً پیام را برای آن نامه‌پراکن

می فرستد. در این سرویس دهنده، سرآیند خارجی RFC 822 حذف و پیام رمزگشایی می شود و سپس برای دیگران ارسال می گردد. سرویس دهنده نامه پراکن، هیچگونه حساب کاربری (Account) تعریف نکرده و هیچ «فایل سوابق» (Log File) ذخیره و نگهداری نمی کند و بدین ترتیب حتی اگر بعداً این سرویس دهنده توقیف شود هیچ ردّی از پیامهایی که از آن گذر کرده اند بجا نمی ماند.

بسیاری از کاربران که علاقه مند و مضّر به ناشناس ماندن هستند به نحوی که در شکل ۸-۵۴ نشان داده شده، پیامهای خود را به صورت زنجیره ای از چندین نامه پراکن عبور می دهند. در این شکل، آلیس تمایل دارد که یک پیام را به صورت کاملاً ناشناس برای باب بفرستد. به همین دلیل از سه سرویس دهنده نامه پراکن استفاده می کند. او پیام M را تنظیم کرده و سرآیند لازم را که آدرس پست الکترونیکی باب جزو آنست به پیام اضافه می کند. سپس کل آن را با کلید عمومی سومین سرویس دهنده نامه پراکن یعنی E3 رمز می کند. (در شکل این پیام به صورت هاشورهای افقی نشان داده شده است.) سپس به پیام سرآیند جدیدی که در برگیرنده آدرس پست الکترونیکی سرویس دهنده سوم است اضافه می کند. این پیام در حقیقت همانی است که در شکل، بین سرویس دهنده ۲ و ۳ مبادله شده است.



شکل ۸-۵۴. چگونگی ارسال پیام ناشناس از آلیس به باب به کمک سه نامه پراکن ناشناس.

سپس پیام جدید را با کلید عمومی سرویس دهنده نامه پراکن دوم رمز می کند. (در شکل این پیام با هاشورهای عمودی نشان داده شده است.) در ادامه سرآیندی حاوی آدرس پست الکترونیکی سرویس دهنده دوم به متن حاصل می افزاید. این پیام در شکل ۸-۵۴ بین سرویس دهنده ۱ و ۲ نشان داده شده است. نهایتاً او کل پیام را با کلید عمومی سرویس دهنده نامه پراکن ۱ رمز کرده و آدرس پست الکترونیکی او را به آن اضافه می کند. این پیام همانی است که بین آلیس و سرویس دهنده اول در شکل مبادله می شود و پیام واقعی که انتقال داده خواهد شد همین پیام است.

وقتی پیام به اولین سرویس دهنده نامه پراکن بر می خورد، سرآیند بیرونی آن جدا شده و متن درون آن رمزگشایی و برای سرویس دهنده دوم ارسال می شود. همین مراحل در دو سرویس دهنده بعدی اتفاق می افتد. اگرچه تعقیب ردّ پیامها برای هر کس بی نهایت دشوار است ولیکن برای اطمینان بیشتر، بسیاری از این نامه پراکنها اقدامات احتیاطی اضافه تری را انجام می دهند. به عنوان مثال ممکن است پیام را به صورت تصادفی برای مدتی معطل کنند یا اطلاعات زائدی را به انتهای پیام بچسبانند یا آنها را حذف کنند یا ترتیب پیامها را عوض نمایند تا هیچکس براحتی نتواند تشخیص بدهد کدام پیام خروجی متناظر با کدام پیام ورودی است و به کدام سرویس دهنده نامه پراکن می رود و بدین ترتیب حمله ای که براساس تحلیل ترافیک صورت می گیرد خنثی خواهد شد. برای شرح بیشتر در خصوص سیستم پست الکترونیکی ناشناس و مدرن به مراجع (Mazieres and Kaashoek, 1998) مراجعه نمایند.

ناشناس ماندن (Anonymity) فقط محدود به نامه‌های الکترونیکی نیست. برای گشت و گذار ناشناس در وب نیز سرویس‌هایی وجود دارد. کاربر، مرورگر خود را به گونه‌ای تنظیم می‌کند تا از سرویس دهنده Anonymizer به عنوان پراکسی استفاده نماید. از آن پس تمام تقاضاهای HTTP به سوی این سرویس دهنده ارسال می‌شود و این سرویس دهنده به نیابت از کاربر صفحات را تحویل گرفته و به او بر می‌گرداند. تمام وب‌سایت‌های دنیا، این سرویس دهنده را (یعنی Anonymizer را) مبداء اصلی تقاضای بیننده کاربر اصلی را، تا زمانی که سرویس دهنده Anonymizer از نگهداری «فایل سوابق» (Log) اجتناب ورزد هیچکس نمی‌تواند تعیین کند چه کسی تقاضای کدام صفحه را داده است.

## ۸-۱-۲ آزادی بیان

رعایت حریم خصوصی (Privacy) خواسته افرادی است که تمایل دارند آنچه دیگران می‌توانند در ارتباط با آنها ببینند و بدانند محدود باشد. یکی دیگر از موارد مهم اجتماعی، «آزادی بیان» و متضاد آن «سانسور عقاید» است. حکومت‌ها می‌خواهند آنچه را که افراد می‌توانند بخوانند یا منتشر کنند، محدود باشد. وب که حاوی میلیون‌ها صفحه حاوی اطلاعات است به بهشت سانسورگران تبدیل شده است. بسته به ماهیت و ایدئولوژی یک ملت، مفاد ممنوعه در وب می‌تواند شامل موارد ذیل باشد:

۱. مفادی که برای کودکان و نوجوانان مناسب نیست.
۲. تنازعات قومی، نژادی، مذهبی و گروهی
۳. اطلاعاتی شامل آموزش جرم و جنایت
۴. اسرار ساخت جنگ‌افزارهای کشتار جمعی
۵. تبلیغ مسائل ضدملی و ضداجتماعی

واکنش معقول آن است که سایت‌های نامناسب باید ممنوع و توسعه‌دهندگان آن تحت پیگرد قرار بگیرند.

ولیکن برخی از بایدها و نبایدها در تعارض با یکدیگر قرار می‌گیرند و تفکر همه افراد، ملتها و حکومت‌ها یکی نیست. به عنوان مثال در نوامبر ۲۰۰۰ دادگاهی در فرانسه به سایت یاهو مستقر در فرانسه اخطار داد که اجازه ندهد شهروندان فرانسوی سایت حراج خاطرات نازیها را ببینند زیرا در اختیار داشتن چنین مواردی طبق قانون فرانسه جرم محسوب می‌شود. چنین مواردی بسیار فراوان است. هر کشور قوانین و قواعد خاص خود را دارد و قوانین آن با آنچه که در وب جریان دارد همسو نیست. در طرف مقابل گستره وب جهانی است و نمی‌توان هیچ قانون واحدی را بر آن حاکم کرد. از طرفی امروزه سرویس‌هایی عرضه شده که اگرچه در مقابل پدیده سانسور قرار می‌گیرند ولیکن می‌تواند در هم شکننده حریم اخلاقی، اجتماعی یا ملی کشورها محسوب شود. نوعی از این سرویس که اصطلاحاً «سرویس ازلی» یا Eternity Service نام دارد در ابتدا به عنوان سیستمی مقاوم در برابر سانسور پیشنهاد شد. (Anderson, 1996) بعداً سیستم‌های کاملتر دیگری پیشنهاد و بعضاً پیاده‌سازی شدند. به این سرویس دهنده‌ها امکاناتی نظیر رمزنگاری، ناشناس ماندن افراد و تحمل خطا و خرابی (Fault Tolerance) اضافه شده است. فایل‌هایی که باید ذخیره شوند به قطعاتی تقسیم شده و این قطعات بر روی سرویس دهنده‌های متعدد ذخیره می‌گردد. در این سیستم‌ها برخی از فایل‌ها تا مدت زمان خاصی حتی توسط صاحب آن قابل تغییر و حذف نیستند و چون دارای پشتیبان‌های متعدد (Backup) هستند حتی در صورت از کار افتادن یا توقیف یکی از آنها، سرویس دهنده‌های دیگر، عرضه اطلاعات را ادامه خواهند داد. برخی از سیستم‌های پیشنهادی در این خصوص عبارتند از: Freenet (Clark, 2002)، PASIS (Wylie, 2000) و Publius (Waldman, 2000). موارد دیگر در مرجع (Serjantov, 2002) گزارش شده‌اند.

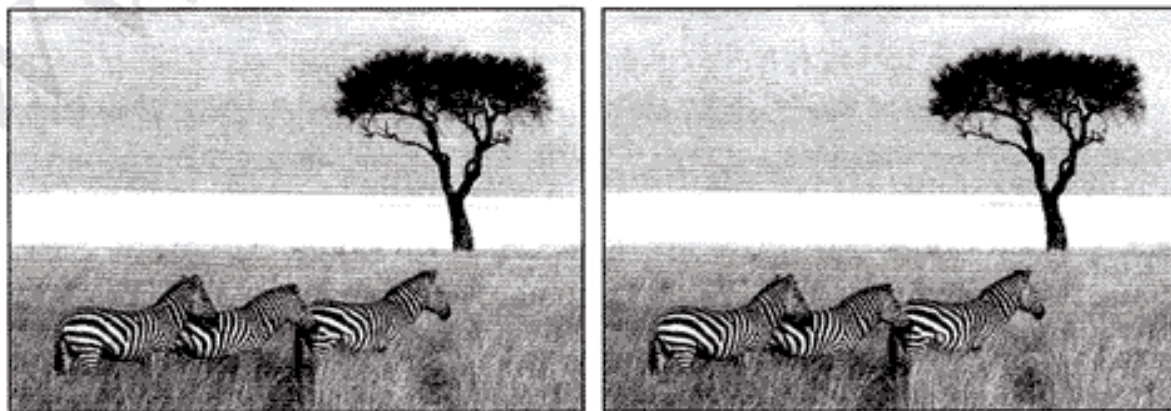
بسیاری از کشورها به صورت فزاینده ای در تلاش هستند تا صدور محصولات غیرعینی و غیرفیزیکی را که بیشتر در برگیرنده سایت های وب، نرم افزار، مقالات علمی، نامه های الکترونیکی و مشاوره های تلفنی هستند، قانونمند و محدود کنند. حتی انگلستان که برای قرن ها مدعی آزادی بیان بوده، بطور جدی در حال بررسی قوانین محدودکننده و سرسختانه ای است که در آن به عنوان مثال مباحثات بین یک استاد بریتانیایی و دانشجوی خارجی او در دانشگاه کمبریج در شمول این قانون قرار گرفته و نیاز به مجوز از دولت دارد. (Anderson, 2002) بدیهی است که چنین سیاست هایی مناقشه برانگیز هستند.

### استیگانوگرافی<sup>۱</sup> (پوشیده نویسی)

در کشورهایی که حجم سانسور بسیار زیاد است مخالفان اغلب سعی می کنند برای فرار از آن از تکنولوژی استفاده نمایند. رمزنگاری اجازه می دهد که پیامها (حتی اگر قانونی هم نباشند) ارسال شوند ولیکن اگر حکومت، آلیس را شخصی بد و مخرب بینگارد، افرادی مثل باب که با او مراوده دارند را نیز هم فکر او تلقی می کند. سرویس دهنده «نامه پراکن ناشناس» (Anonymous Remailer) می تواند به آنها کمک کند ولی اگر چنین سرویس دهنده هایی تحریم و مسدود شده باشند و یا ارسال پیام به یکی از آنها در خارج، نیاز به مجوز دولت داشته باشد، چندان مفید نخواهند بود؛ ولی وب راه گشا است.

افرادی که می خواهند به صورت سری با یکدیگر ارتباط داشته باشند اغلب سعی می کنند این ارتباط را به هر نحوی پنهان نگاه دارند. علم مخفی کردن پیامها اصطلاحاً «استیگانوگرافی» نامیده می شود که برگرفته از دو کلمه یونانی به معنای «پوشیده نویسی» است. در حقیقت در ابتدا یونانیان باستان خود از این روش استفاده کرده اند. «هروودوت» مورخ یونانی از یک ژنرال ارتش یاد کرده که سر پیام رسان خود را تراشید و پیام را بر روی پوست سر او خالکوبی کرد؛ سپس صبر کرد تا قبل از اعزام او به مأموریت، موهای او رشد کنند! تکنیک های مدرن از لحاظ مفهوم مشابه همین روش هستند!

به عنوان یک مثال از استیگانوگرافی به شکل ۸-۵۵-الف دقت کنید. این عکس توسط مؤلف کتاب در کنیا گرفته شده است و در آن سه گورخر در کنار درخت افاقیا دیده می شوند. تصویر ۸-۵۵-ب نیز مشابه با عکس قبلی به نظر می رسد ولیکن این عکس جذابیتهای دیگری هم دارد! این عکس حامل متن کامل پنج نمایشنامه از شکسپیر است که مخفیانه درون آن جاسازی شده است، شامل: «هملت»، «شاه لیر»، «مکبث»، «تاجر ونیزی» و «ژولیس سزار». مجموع این نمایشنامه ها جمعاً ۷۰۰ کیلوبایت متن هستند.



(الف)

(ب)

شکل ۸-۵۵. (الف) سه گورخر و یک درخت! (ب) سه گورخر و یک درخت و متن کامل ۵ نمایشنامه از شکسپیر!!

این مخفی سازی چگونه انجام می شود؟ ابعاد تصویر رنگی و اصلی  $1024 \times 768$  نقطه (پیکسل) است. هر پیکسل شامل سه عدد هشت بیتی است که هر یک شدت رنگهای قرمز، سبز و آبی را در هر نقطه تصویر مشخص می کنند. از ترکیب این سه رنگ (با شدتهای متفاوت) رنگ هر نقطه بدست می آید. در روش کدگذاری مخفی از کم ارزش ترین بیت هر یک از سه مقدار رنگهای RGB به عنوان «کانالهای مخفی» (Covert Channel) استفاده می شود. بنابراین هر پیکسل فضایی معادل ۳ بیت برای جاسازی اطلاعات سری در اختیار می گذارد؛ (یک بیت در مقدار قرمز، یکی در آبی و یکی در سبز). در تصویری به ابعاد فوق مجموعاً  $1024 \times 768 \times 3$  بیت (معادل  $294912$  بایت) از اطلاعات سری را می توان جاسازی کرد.

متن کامل این پنج نمایشنامه به همراه یک توضیح کوتاه جمعاً حدود  $734891$  بایت است. این متن ابتدا با استفاده از الگوریتم استاندارد فشرده سازی، به  $274$  کیلوبایت فشرده شده و سپس نتیجه، با استفاده از الگوریتم IDEA رمزنگاری و در کم ارزش ترین بیت از مقادیر رنگها ذخیره شده است. به گونه ای که مشاهده می شود (یا به عبارت بهتر به گونه ای که مشاهده نمی شود!!) وجود این اطلاعات کاملاً غیر قابل رویت است. حتی در تصویر بزرگ شده و تمام رنگی این عکس باز هم چیزی قابل رویت نیست. چشم نمی تواند تفاوت بین رنگهای ۲۱ بیتی یا ۲۴ بیتی را تشخیص بدهد.

البته مشاهده دو تصویر فوق به صورت سیاه و سفید و با دقت پایین در این کتاب، نمی تواند در مورد قدرت این تکنیک قضاوت کند. برای آن که عملکرد استیگانوگرافی را بهتر احساس کنید، مؤلف یک نمونه نمایشی از استیگانوگرافی، شامل تصویر تمام رنگی با دقت بالا از شکل ۸-۵۵-ب را به همراه پنج نمایشنامه جاسازی شده، عرضه کرده است. این نمونه نمایشی شامل یک ابزار برای جاسازی و استخراج متن در تصویر است که می توانید آن را در وبسایت این کتاب بدست بیاورید.

برای استفاده از استیگانوگرافی به منظور محاوره مخفیانه، معاندین می توانند یک وبسایت ایجاد کنند که سرشار از تصاویر مجاز و قانونی باشد. در حالی که پیامهای مخفی در آن جاسازی شده است. اگر پیامها ابتدا فشرده و سپس رمزنگاری شوند حتی در صورتی که کسی به وجود پیام مخفی در آن شک کند قطعاً برای او تشخیص پیام از نویز سفید تصویر، بسیار دشوار خواهد بود.

تصاویر بهیچوجه تنها حامل پیامهای مخفی نیستند. فایل های صوتی نیز بخوبی کارایی دارند. فایل های ویدیویی دارای پهنای باند بسیار عظیمی برای پنهان سازی اطلاعات هستند. حتی ترکیب چیده شدن عناصر (Layout) و ترتیب برچسبها در فایل HTML (HTML Tags) نیز می تواند حامل اطلاعات باشد!

استیگانوگرافی تنها برای حمل اطلاعات مخفی نیست و کاربردهای دیگری هم دارد. یکی از کاربردهای عمومی آن می تواند این باشد که صاحب حقوقی یک عکس پیامهایی سری در درون یک تصویر جاسازی کند. هر گاه چنین تصویری دزدیده شده و در یک وبسایت قرار داده شود، مالک قانونی آن می تواند این پیام محرمانه و سری را برای اثبات مالکیت آن، به دادگاه عرضه کند. به این تکنیک اصطلاحاً نشانه گذاری (Watermarking) گفته می شود و در مرجع (Piva et al., 2002) تشریح شده است.

برای بدست آوردن اطلاعات بیشتر در خصوص استیگانوگرافی به مراجع ذیل مراجعه کنید:

(Artz, 2001; Johnson and Jajoda, 1998; Katzenbeisser and Petitcolas, 2000; and Wayner, 2002)

### ۸-۱۰-۳ مالکیت معنوی (Copyright)

رعایت حریم خصوصی افراد و پدیده سانسور موضوعاتی هستند که سیاستهای عمومی و تکنولوژی را رو در رو قرار داده اند. مورد سوم مالکیت معنوی آثار است. مالکیت معنوی (Copyright) بدین معناست که امتیاز

بهره برداری از عواید یک اثر برای مدتی برای آفرینندگان آن (شامل نویسندگان، هنرمندان، آهنگسازان، نوازندگان، عکاسان، سینماگران) محفوظ بماند که این دوره زمانی می تواند ۵۰ تا ۷۵ سال پس از عمر صاحب اثر باشد. پس از آن که طول دوره این امتیاز که به صورت قانونی ثبت می شود به سر آمد، آن اثر همگانی تلقی شده و همه می توانند به دلخواه از آن بهره ببرند یا آن را بفروشند. به عنوان مثال پروژه گوتنبرگ یا آثار شکسپیر امروزه همگانی تلقی می شوند و به رایگان در وب در دسترس هستند. در سال ۱۹۹۸ کنگره آمریکا طول زمان مالکیت آثار را به درخواست هالیوود که مدعی شده بود اگر افزایش نیابد هیچکس قادر نخواهد بود چیزی خلق کند، ۲۰ سال اضافه کرد.

مناقشه بر سر موضوع مالکیت آثار، زمانی اوج گرفت که تعداد مشترکین شرکت Napster (ارائه دهنده خدمات مبادله رایگان موزیک) به ۵۰ میلیون نفر رسید. از آنجایی که Napster هیچگونه عمل کپی فایل های موزیک را انجام نمی داد دادگاه به طرح این اتهام پرداخت که در اختیار گذاشتن یک بانک اطلاعاتی از اینکه چه کسی چه موزیک هایی را در اختیار دارد معاونت در جرم تلقی می شود و Napster بدین ترتیب به وقوع جرم کمک کرده است. اگرچه هیچکس مخالف قانون مالکیت معنوی آثار نیست (هرچند برخی ادعا می کنند طول دوره و جزئیات سختگیرانه آن زیاد است) ولیکن دور جدیدی از به اشتراک گذاری موزیک، یک مناقشه عظیم را دامن زده است.

به عنوان مثال شبکه ای نقطه به نقطه را در نظر بگیرید که در آن افراد فایل های قانونی خود را (شامل موزیک های همگانی، ویدیو های شخصی، اعلان های مذهبی که در آن اسراری وجود ندارد) و شاید چند فایل که حق مالکیت آن انحصاری است را به اشتراک گذاشته اند. فرض کنید افراد به صورت دائم از طریق یک خط ADSL یا کابل، به شبکه وصل شده اند. هر ماشین یک فهرست از آنچه بر روی دیسک سخت او موجود است و همچنین فهرستی از بقیه مشترکین را در اختیار دارد. هر کسی که به دنبال یک مورد خاص می گردد یکی از اعضای این فهرست را انتخاب کرده و بررسی می کند که آیا او این آیتم را در اختیار دارد یا خیر؟ اگر نداشت او می تواند در فهرست یکایک اعضا بررسی کند. پس از پیدا شدن آیتم مورد نظر، متقاضی می تواند آن را کپی نماید.

اگر آنچه که به اشتراک گذاشته می شود تحت حمایت قانون مالکیت معنوی باشد، آنهایی که چنین آیتم هایی را برداشت می کنند مرتکب قانون شکنی شده اند. (هر چند مشخص نیست وقتی که انتقال چنین آیتم هایی به صورت بین المللی انجام می شود، چه قانونی قابل اعمال و استناد خواهد بود.) با کسی که چنین زمینه ای را فراهم کرده چه باید کرد؟ آیا این جرم است که یک موزیک را که شما بهای آن را پرداخته و بر روی دیسک سخت خود ذخیره کرده اید، دیگران پیدا و دریافت کنند؟ اگر مثلاً شما درب اتاق خود را قفل نکرده باشید و یک دزد بتواند از روی یکی از کتابهای شما کپی بگیرد، آیا شما در جرم نقض حقوق قانونی آن کتاب، گناهکار هستید؟

مناقشات بسیار گسترده ای پیرامون مالکیت معنوی آثار در گرفته است و نزاع شدیدی بین هالیوود و شرکتهای کامپیوتری بوجود آمده است. هالیوود می خواهد تا قوانین حفاظت از آثار، سختگیرانه و سنگین تر شود و شرکتهای کامپیوتری نیز نمی خواهند مأمور حفاظت از منافع هالیوود باشند.

در اکتبر ۱۹۹۸، کنگره آمریکا قانونی به نام DMCA (Copyright Act Digital Millennium) را از تصویب گذراند که براساس آن هر گونه فریبکاری و حیل پردازی برای نقض مالکیت آثار در پوششهای به ظاهر قانونی یا انتشار چنین راههایی برای فرار از قانون، جرم محسوب می شود. چنین قانونی در اروپای متحد نیز از تصویب گذشت. در حالی که بسیاری از افراد آگاه نیستند که در شرق دور نسخه برداری از آثار مجاز شمرده می شود! و خوشبینانه بدان می اندیشند که قانون DMCA می تواند بین حقوق مالکین و پدید آورندگان آثار و حقوق عمومی یک توازن ایجاد کند.

اشاره به موردی دیگر خالی از لطف نیست. در سپتامبر ۲۰۰۰، یک کنسرسیوم از صنایعی که در موزیک فعالیت می کردند سیستمی غیرقابل نفوذ برای فروش موزیک (به صورت On-line) را سازماندهی و ایجاد کرده و سپس از رقبا خواستند تا افراد را به شکستن این سیستم دعوت نمایند (که عملی کاملاً قانونی برای هر سیستم امنیتی جدید است چراکه به آشکار شدن اشکالات سیستم کمک می کند). یک گروه از محققین امنیت سیستم از چندین دانشگاه به سرپرستی پروفیسور ادوارد فلتن از دانشگاه پرینستون بدین چالش علمی وارد شده و این سیستم را درهم شکستند. سپس مقاله ای در خصوص یافته های خود نوشته و آن را برای کنفرانس امنیت USENIX ارسال کردند. قبل از آن که موعد ارائه این مقاله فرا برسد، فلتن نامه ای از انجمن صنایع ضبط و پخش موزیک در آمریکا دریافت کرد که او را تهدید کرده بودند در صورت انتشار مقاله، بر علیه او طبق قانون DMCA ادعای خسارت خواهند کرد.

در پاسخ، فلتن از دادگاه فدرال کسب تکلیف کرد که آیا تألیف مقالات علمی در خصوص امنیت سیستمها قانونی است یا خیر؟ این انجمن از ترس آن که دادگاه بر علیه آنها کاری انجام بدهد دست از تهدید فلتن برداشتند و پرونده مختومه شد. شکی نیست که این صنایع اشکال کار را در خودشان می دیدند: از یک طرف افراد را دعوت به شکستن سیستم کرده و از طرف دیگر آنهایی که چالش آنها را پذیرفته بودند تهدید به ادعای خسارت و شکایت کردند. پس از آن که تهدید رفع شد مقاله فوق الذکر منتشر گردید.

مورد دیگری که به بحث ما مرتبط است بسط «نظریه استفاده جوانمردانه از آثار» (Fair Use Doctrine) است که توسط قانونگذاران قوه قضاییه در بسیاری از کشورها وضع شده است. این نظریه بیان می کند که خریداران یک اثر که حقوق معنوی آن متعلق به دیگران است، اجازه دارند طبق ضوابط خاصی از آن کار نسخه برداری کنند یا بخشهایی از آن را در جهت مقاصد علمی نقل قول کنند، مفاد آن را در دانشگاه یا مدارس تدریس نمایند و حتی برای اطمینان خاطر از آنکه در صورتی خرابی نسخه اصلی یک اثر چیزی از دست ندهند، از آن چندین نسخه کپی تهیه کنند. برای بررسی آن که آیا از یک اثر، استفاده جوانمردانه می شود یا نه، باید معیارهای زیر ارزیابی شود: (۱) آیا استفاده از آن اهداف تجاری دارد. (۲) چند درصد از کل آن نسخه برداری می شود. (۳) نسخه برداری از آن بر فروش آن اثر چقدر تأثیر منفی دارد. از آنجایی که قانون DMCA و قوانین مشابه در اروپا، هرگونه روشهای زیرکانه و فریبکارانه برای پایمال کردن حقوق معنوی آثار را غیرمجاز می داند، استفاده جوانمردانه و طبیعی با معیارهای فوق الذکر را نیز غدغن کرده است.

زمانی که که قانون DMCA تلاش داشت بین حقوق قانونی پدیدآورندگان آثار و حقوق استفاده کنندگان توازن معقول ایجاد کند، طرح جدیدی به رهبری اپتل و مایکروسافت به نام TCPA<sup>۱</sup> ارائه شد. نظریه آن بود که یک تراشه CPU و سیستم عامل آن، بدقت بر رفتار و عملکرد کاربران نظارت داشته باشد (مثل اجرای یک موزیک یا نرم افزار که به صورت غیرقانونی کپی شده است) و جلوی اعمال غیرمجاز کاربر را بگیرد. این سیستم حتی به مالکان نرم افزارها یا دیگر کالاهای الکترونیکی اجازه می دهد تا در هر زمان که صلاح دیدند از راه دور به PC کاربران سرکشی کرده و قواعد استفاده از محصولاتشان را تغییر بدهند. بدیهی است که تبعات اجتماعی چنین طرحی، بسیار زیاد خواهد بود. اگرچه توجه صاحبان صنایع به موضوع امنیت پسندیده و قابل تحسین است ولی بسیار ناگوار است که آنها به جای پرداختن به مسئله ویروسها، کراکرها (Crackers)، اختلالگران و دیگر موارد امنیتی که مردم با آن دست به گریبانند، تمام تلاش خود را مصروف اجرای قانون حمایت از پدیدآورندگان آثار کرده اند!!

کوتاه سخن آن که قانونگذاران و وکلا در سالهای آتی نیز درگیر مسئله توازن بین حقوق معنی دهنده و حقوق

پدیدآورندگان آثار خواهند بود. دنیای الکترونیکی امروز بی شباهت به صحنه جنگ نیست: گروهی به جان گروه دیگر می افتند، یکدیگر را لگدمال می کنند، کارشان به دادگاه می کشد و خوشبختانه سرانجام کار، اکثراً مصالحه می کنند و این روال ادامه خواهد یافت تا در آخر تکنولوژی جدید از راه برسد.

## ۱۱-۸ خلاصه

رمزنگاری، ابزاری مؤثر برای محرمانه نگاه داشتن اطلاعات و اطمینان از صحت و هویت آنهاست. سیستمهای رمزنگاری جدید براساس قانون یکرکف بنا شده اند یعنی الگوریتم یکار رفته در آنها آشکار و عمومی و فقط کلید رمز سری است. بسیاری از الگوریتمهای رمزنگاری از تبدیلهای پیچیده ریاضی شامل عملیات جانشینی و جایگشتی برای تبدیل متن به رمز بهره گرفته اند. ولیکن هرگاه رمزنگاری کوانتومی بتواند در محیط عمل وارد شود، روش One-Time Pad یک سیستم رمزنگاری واقعاً غیرقابل شکست عرضه خواهد کرد.

الگوریتمهای رمزنگاری را می توان به دو دسته تقسیم کرد: الگوریتمهای با کلید متقارن و الگوریتمهای با کلید عمومی. الگوریتمهای با کلید متقارن، بیتهای متن را در چندین مرحله و براساس پارامترهای مشتق شده از کلید اصلی، ترکیب و مخلوط می کنند تا در نتیجه متن رمز شده بدست آید. در حال حاضر الگوریتمهای Triple DES و Rijndael (AES) مشهورترین الگوریتمهای با کلید متقارن هستند. از این الگوریتمها می توان در حالتهای Counter Mode، Stream Cipher Mode، Chaining Mode، Cipher Block، Electronic Code Book و نظایر آن بهره گرفت.

الگوریتمهای رمزنگاری با کلید عمومی این ویژگی را دارند که کلیدهای رمزنگاری و رمزگشایی متفاوت از یکدیگر هستند و نمی توان با داشتن کلید رمزنگاری، کلید رمزگشایی را استخراج کرد. این ویژگی اجازه می دهد تا بتوان کلیدهای عمومی را منتشر کرد. اصلی ترین الگوریتم کلید عمومی، RSA است که قدرت خود را از آنجایی بدست آورده که تجزیه اعداد بزرگ به عوامل اول بسیار بسیار دشوار است.

اسناد قانونی، تجاری و نظایر آن نیازمند امضاء هستند. بر این اساس روشهای متنوعی برای امضای دیجیتالی ابداع شده که در آن، هم از الگوریتمهای رمزنگاری با کلید متقارن و هم از روشهای کلید عمومی بهره گرفته شده است. بطور معمول، ابتدا از پیامهایی که باید امضا شوند با استفاده از روشهایی مثل MD5 یا SHA-1 یک رشته Hash (رشته خلاصه و درهم شده پیام) استخراج شده و سپس این رشته به جای متن اصلی رمزنگاری می شود. مدیریت کلیدهای عمومی با استفاده از گواهینامه های دیجیتالی که در آن هویت شخص و کلید عمومی او درج شده، قابل انجام است. گواهینامه های دیجیتالی توسط مراکز معتمد مردم یا اشخاص حقیقی تانید می شوند. «ریشه» (یعنی Root یا عالیترین مرکز گواهی امضاء) باید پیشاپیش مشخص باشد ولیکن اغلب مرورگرها گواهینامه دیجیتالی بسیاری از مراکز اصلی گواهی امضاء را به صورت درونی در اختیار دارند.

ابزارهای رمزنگاری می توانند برای امن کردن ترافیک جاری بر روی شبکه به کار گرفته شوند. IPsec در سطح لایه شبکه عمل می کند و بسته هایی را که از یک ماشین به ماشین دیگر روانه می شوند، رمزنگاری می کند. دیوارهای آتش می توانند بر ورود و خروج اطلاعات یک سازمان نظارت کنند که این کار اغلب براساس نوع پروتکل (شماره پروتکل لایه انتقال) و شماره پورت انجام می گیرد. شبکه های VPN می توانند شبکه های قدیمی که مبتنی بر خطوط اجاره ای و انحصاری بودند را با سطح امنیت مورد نظر شبیه سازی کنند. نهایتاً شبکه های بی سیم نیاز به امنیت خوب و کافی دارند در حالی که شبکه WEP 802.11 چنین امنیتی را فراهم نکرده است؛ اگرچه 802.11i خواهد توانست این ویژگی را بهبود بخشد.

وقتی دو طرف با یکدیگر نشستی را ترتیب می دهند، در ابتدا مجبور هستند یکدیگر را تانید هویت کنند و یک

کلید نشست ایجاد نمایند. در این خصوص، پروتکل های احراز هویت متفاوتی وجود دارد، شامل: روش مبتنی بر یک شخص ثالث و معتمد، روش دیفی-هلمن، روش Kerberos و روش رمزنگاری کلید عمومی. امنیت نامه های الکترونیکی را می توان براساس ترکیب روشهایی که در این فصل معرفی شدند، تأمین کرد. به عنوان مثال در PGP ابتدا پیامها فشرده شده و سپس توسط روش IDEA رمز می شوند. کلید رمز IDEA، یکمک کلید عمومی گیرنده پیام، رمز می شود و به همراه پیام ارسال می گردد. بعلاوه برای بررسی صحت پیامها، از درون آن یک رشته Hash استخراج شده و امضاء می شود.

امنیت وب نیز یکی از عناوین مهم در امنیت شبکه است که با مقوله «نامگذاری امن» آغاز می شود. DNSsec روشی است که نامها خودشان صحت خود را گواهی کنند و بدین ترتیب از حمله DNSspoofing پیشگیری می شود. بسیاری از وبسایت های تجارت الکترونیکی برای برقراری نشست های امن و احراز هویت شده بین مشتری و سرویس دهنده، از SSL بهره می گیرند. روش های متنوعی نیز برای برخورد مطمئن با کدهای متحرک (نظیر اسکریپتها و اکتیواکس) ابداع شده است.

اینترنت موارد بسیار متعددی را بوجود آورده که تکنولوژی و سیاست های عمومی را روبروی هم قرار داده است. برخی از موضوعات در این مقوله عبارتند از «حریم خصوصی افراد»، «آزادی بیان» و «مالکیت معنوی آثار».

## مسائل

۱. رمز قطعه کد تک حرفی (Monoalphabetic) زیر را بشکنید. متن اصلی فقط از حروف الفباء تشکیل شده و یک قطعه ادبی از آثار مشهور لوئیس کارول است.

sok pztk z stk kfd uamkdim eitdx sdruid pd fzld uoi efzk  
rui mubd ur om zid uok ur sidzkh zhx zyy ur om zid rzk  
hu foia mztz kfd ezindhkdi kfda kfzhgdx fitb boef rui kfzk

۲. رمز قطعه کد زیر را که به صورت جایگشت ستونی رمزنگاری شده، بشکنید. این متن از یک کتاب معمولی رشته کامپیوتر انتخاب شده و طبعاً کلمه computer محتمل ترین کلمه درون آن است. متن اصلی فقط از حروف الفباء انگلیسی تشکیل شده و فاصله خالی درون آن نیست. فقط برای سادگی در خواندن، متن رمز شده به صورت دسته های پنج حرفی نشان داده شده است (فاصله خالی را در حین محاسبات خود حذف نمایید).

aauan cvlre runnn dlme aeepb ytust iceat nrmey iicgo gorch srsoc  
nntii imiha oofpa gsivt tpsit lbofr otoex

۳. یک رشته بیت One-Time Pad ۷۷ بیتی برای متن رمز شده شکل ۸-۴ پیدا کنید تا متن "Donald Duck" را تولید نماید.

۴. رمزنگاری کوانتومی به یک تفنگ فوتونی نیاز دارد تا بتواند در صورت نیاز، یک تک فوتون حامل بیت ۱ تولید نماید. محاسبه نماید که بر روی یک فیبرنوری صد گیگاهرتز یک تک بیت حامل چه تعداد فوتون است. فرض کنید که طول یک فوتون معادل طول موج آنست که در این مسئله یک میکرون فرض شده است. سرعت نور در یک فیبرنوری را ۲۰ سانتی متر در نانو ثانیه (20Cm/nsec) در نظر بگیرید.

۵. وقتی از رمزنگاری کوانتومی استفاده می شود اگر ترویدی بتواند فوتون های نوری را دریافت و بازتولید نماید برخی از بیتها را اشتباه دریافت خواهد کرد و همچنین خطاهایی را در رشته بیت One-Time Pad متعلق به باب بوجود خواهد آورد. بطور متوسط چه نسبی از رشته بیت باب خراب خواهد شد؟

۶. یکی از اصول پایه رمزنگاری بیان می‌کند که تمام پیامها باید افزونگی داشته باشد. ولیکن از طرفی با این موضوع آشنا شدیم که وجود افزونگی به اختلالگر کمک می‌کند تا صحت حدس خود در مورد کلید رمز را بررسی نماید. حال به دو نوع افزونگی زیر دقت نمائید: اول آن که در  $n$  بیت از متن اصلی، یک الگوی شناخته شده قرار داده شود. دوم آن که در  $n$  بیت نهایی پیام، یک رشته Hash (استخراج شده از پیام)، قرار داده شود. آیا این دو رویکرد از دیدگاه امنیت، معادل و یکسان هستند؟ پاسخ خود را تشریح کنید.
۷. در شکل ۸-۶ (سمت راست) P-Box ها و S-box ها به صورت متناوب و یک در میان قرار گرفته‌اند. اگرچه این ساختار به ظاهر خوب و مناسب به نظر می‌رسد، آیا برای تضمین امنیت بیشتر بهتر نیست که ابتدا تمام P-Box ها و سپس تمام S-box ها قرار بگیرند؟
۸. حمله‌ای را بر علیه سیستم DES ترتیب بدهید با این دانش قبلی که متن رمز شده صرفاً از حروف الفبای انگلیسی بزرگ، فاصله خالی، کاما، نقطه اعشار، سمی کالون، و کاراکترهای Line Feed و Carriage Return تشکیل شده است. هیچ چیزی در مورد بیت‌های توازن (Parity Bits) در متن اصلی مشخص نیست.
۹. در این فصل محاسبه کردیم برای شکستن رمز استاندارد AES با کلید ۱۲۸ بیتی، یک ماشین رمز شکن با یک میلیارد پردازنده که می‌تواند در هر پیکوثانیه ( $10^{-12}$  sec) یک کلید را آزمایش کند، به حدود  $10^{10}$  سال، زمان نیاز خواهد داشت. ولیکن در ماشینهای فعلی که حداکثر می‌توانند تا حدود  $10^{24}$  پردازنده داشته باشند، برای شکستن رمز AES باید کارایی آنها تا  $10^{15}$  بار بهتر شود. اگر قانون Moore (که بیان می‌کند قدرت پردازش کامپیوترها هر ۱۸ ماه دو برابر می‌شود) روند خود را ادامه بدهد، چند سال طول می‌کشد تا چنین ماشینینی ساخته شود؟
۱۰. AES از کلیدهای ۲۵۶ بیتی حمایت می‌کند. در AES-256 چند کلید می‌تواند وجود داشته باشد؟ بررسی کنید که آیا می‌توانید عددی معادل با این عدد در فیزیک، شیمی یا نجوم پیدا کنید. از اینترنت برای جستجوی اعداد بسیار بزرگ بهره بگیرید و نتیجه‌گیری خود از این تحقیق را ارائه بدهید.
۱۱. فرض کنید که پیامی با استفاده از DES و در حالت زنجیره‌سازی بلوکها (Block Chaining) رمزنگاری شده است. در حین انتقال یک بیت از متن رمز شده در بلوک  $C_i$  تصادفاً از صفر به یک تبدیل شده است. در اثر این خطا چقدر از متن اصلی (پس از رمزگشایی) خراب و بلااستفاده خواهد شد؟
۱۲. حالا مجدداً سیستم DES در حالت زنجیره‌سازی بلوکها را در نظر بگیرید. فقط به جای آنکه یک بیت صفر در حین انتقال به ۱ تبدیل شود یک بیت صفر اضافی تصادفاً در لابلای متن رمز شده بعد از بلوک  $C_i$  اضافه می‌شود. در اثر این خطا (پس از رمزگشایی) چقدر از متن اصلی خراب و بلااستفاده خواهد شد؟
۱۳. روش زنجیره‌سازی بلوکها (Block Chaining) را با روش Cipher Feedback Mode برحسب تعداد عملیات رمزنگاری مورد نیاز برای انتقال یک فایل بزرگ مقایسه نمائید. کدامیک از این روشها کارآمدتر و سریعتر است و چقدر؟
۱۴. در سیستم رمزنگاری کلید عمومی RSA، کاراکترهای  $a$  با عدد ۱،  $b$  با عدد ۲،  $c$  با عدد ۳ و به همین ترتیب کدگذاری شده‌اند:
- الف) اگر  $p=7$  و  $q=11$  باشد، پنج مقدار معتبر برای  $d$  بیابید.
- ب) اگر  $p=13$  و  $q=31$  و  $d=7$  باشد،  $e$  را پیدا کنید.
- ج) با داشتن  $p=5$  و  $q=11$  و  $d=27$ ، ابتدا  $e$  را یافته و سپس متن "abcdefghij" را رمز کنید.
۱۵. فرض کنید کاربری به نام ماریا متوجه می‌شود که کلید خصوصی RSA او یعنی  $(d_1, n_1)$  دقیقاً معادل با کلید

عمومی RSA از یک کاربر دیگر به نام فرانسیس با کلید  $(e_2, n_2)$  است. به عبارت دیگر  $d_1 = e_2$  و  $n_1 = n_2$  است. آیا ماریا باید کلیدهای عمومی و خصوصی خود را تغییر بدهد؟ پاسخ خود را تشریح کنید.

۱۶. به سیستم رمزنگاری نشان داده شده در شکل ۸-۱۵ (یعنی روش Counter Mode) دقت کنید با این تفاوت که IV را در این شکل معادل صفر در نظر بگیرید. آیا عموماً استفاده از صفر برای IV، امنیت این سیستم رمز را به خطر می اندازد؟

۱۷. پروتکل امضای دیجیتالی نشان داده شده در شکل ۸-۱۸ دارای این ضعف است که اگر ماشین باب به ناگاه مختل شود (اصطلاحاً crash کند) تمام محتویات RAM از دست خواهد رفت. این مسئله منجر به بروز چه اشکالی می شود و او چگونه می تواند از بروز این اشکال پیشگیری نماید.

۱۸. در شکل ۸-۲۰ می بینیم که چگونه آلیس پیامی امضاء شده را برای باب می فرستد. اگر ترویدی متن P را کلاً عوض کند باب متوجه خواهد شد. به نظر شما اگر ترویدی هم P و هم امضای آن را عوض کند چه اتفاقی می افتد؟

۱۹. امضاهای دیجیتالی دارای یک ضعف بالقوه هستند که از تنبلی کاربران ناشی می شود. در معاملات تجارت الکترونیکی پیش نویس یک قرارداد تهیه شده و از کاربر خواسته می شود تا رشته SHA-1 Hash متناظر با آن قرارداد را با کلید خصوصی خود امضاء نماید. اگر کاربر بررسی نکند که آیا حقیقتاً رشته Hash که آنرا امضاء می کند متناظر با قرارداد مورد نظر اوست ممکن است سهواً Hash یک قرارداد دیگر را امضاء کند. فرض کنید مافیا سعی می کند از این اشکال سوءاستفاده کرده و پول بدست بیاورد. آنها یک وبسایت که کاربران برای ورود به آن مجبورند پول بپردازند ایجاد کرده و از مشتریان خود می خواهند که شماره کارت اعتباری خود را وارد نمایند. سپس قراردادی را برای مشتری می فرستند تا آنرا امضاء کند، با این نیت که اکثر کاربران بدون بررسی آنکه آیا Hash مربوط به قرارداد متناظر و متعلق به پیام هست یا خیر، آن را امضاء می کنند. نشان بدهید که چگونه مافیا می تواند مقداری الماس از یک جواهرفروش در اینترنت بخرد و قیمت آن را بر گردن یک کاربر که کسی به او مشکوک نخواهد شد بیندازد؟

۲۰. یک کلاس ریاضی ۲۰ دانش آموز دارد. احتمال آن که حداقل دو دانش آموز در یک روز از سال به دنیا آمده باشند چقدر است؟ فرض کنید هیچکس در سال کبیسه به دنیا نیامده باشد و روزهای مختلف تولد ۳۶۵ حالت بیشتر نیست.

۲۱. در سناریوی بخش ۸-۴-۴ پس از آن که الن نزد مرلین اعتراف کرد که در خصوص رسمی شدن تام در منصب هیئت علمی دانشگاه فریبکاری کرده است، مرلین برای پیشگیری از این مسئله سعی می کند ضمن دیکته کردن پیامهای آتی خود از طریق یک ماشین خاص، منشی خود را نیز عوض کند. سپس مرلین به گونه ای برنامه ریزی می کند تا از آن به بعد پیامهای تایپ شده را پس از تایپ بدقت بررسی کند تا مطمئن شود کلمات متن بدقت و صحیح تایپ شده باشند. آیا منشی جدید هنوز هم می تواند از «حمله روز تولد» (Birthday Attack) برای جعل پیامها بهره بگیرد و اگر می تواند چگونه؟ (راهنمایی: این کار ممکن است).

۲۲. به تلاش ناموفق آلیس در دریافت کلید عمومی باب در شکل ۸-۲۳ دقت نمایید. فرض کنید باب و آلیس از قبل بر روی یک کلید سری و مشترک توافق کرده اند ولی باز هم آلیس به کلید عمومی باب نیاز دارد. آیا در چنین حالتی روشی برای دریافت مطمئن این کلید وجود دارد؟ اگر وجود دارد چگونه؟

۲۳. آلیس می خواهد با باب به کمک کلید عمومی، تبادل اطلاعات داشته باشد. او یک اتصال با کسی که فکر می کند باب است برقرار می نماید و از طرف مقابل می خواهد که کلید عمومی خود را بفرستد؛ طرف مقابل نیز کلید عمومی خود را به صورت آشکار و به همراه گواهینامه X.509 خود که توسط مرکز عالی CA امضاء

- شده، ارسال می کند. آلیس نیز یک کلید عمومی که توسط مرکز CA امضاء شده در اختیار دارد. چند مرحله باید انجام شود تا آلیس اطمینان حاصل کند که طرف مقابل او واقعاً باب است. فرض کنید باب نیز از هویت کسی که با او در حال محاوره است مطمئن نیست. (مثلاً باب یک سرویس دهنده عمومی است).
۲۴. فرض کنید که یک سیستم از PKI مبتنی بر ساختار «سلسله مراتب مراکز CA» بهره می گیرد. آلیس می خواهد که با باب ارتباط برقرار کند و به همین دلیل پس از ایجاد ارتباط، گواهینامه دیجیتالی باب را که توسط یک CA با نام X امضاء شده دریافت می کند. فرض کنید او هیچ چیزی در مورد مرکز X نمی داند. آلیس باید چه مرحله برای بررسی هویت کسی که با او صحبت می کند، پشت سر بگذارد.
۲۵. آیا در یک ماشین که در پشت جعبه NAT (NAT BOX) قرار گرفته می توان از IPsec (با استفاده از AH و در حالت انتقال) بهره گرفت؟
۲۶. یک مزیت استفاده از HMAC به جای RSA، برای امضای رشته های SHA-1 Hash را عنوان کنید؟
۲۷. یک دلیل بیاورید که چرا دیوار آتش ممکن است برای بررسی ترافیک ورودی پیکربندی شود؛ همچنین یک دلیل بیاورید که چرا دیوار آتش ممکن است برای بررسی ترافیک خروجی پیکربندی شود؛ آیا فکر می کنید این بررسیها احتمال موفقیت دارد؟
۲۸. قالب بسته WEP در شکل ۸-۳۱ نشان داده شده است. فرض کنید که کد کشف خطای ۳۲ بیتی در این بسته، با XOR کردن کلمات ۳۲ بیتی بخش داده (Payload) محاسبه شود. همچنین فرض کنید که برای رفع مشکلات RC4، از یک روش قدرتمند مبتنی بر Stream Cipher (بخش ۸-۲-۳) استفاده شود و IV به ۱۲۸ بیت توسعه یابد. آیا راهی وجود دارد که یک اخلالگر بتواند اطلاعات را استراق سمع یا دستکاری کند بدون آن که کشف شود؟
۲۹. فرض کنید که یک سازمان برای اتصال چندین سایت از طریق اینترنت به روش امن، از VPN بهره گرفته باشد. آیا لازم است کاربری مثل جین که می خواهد در درون همین سازمان با کاربری دیگر به نام مری ارتباط برقرار کند از رمزنگاری یا مکانیزمهای امنیتی استفاده کند؟
۳۰. یکی از پیامهای پروتکل ۸-۳۴ را به گونه ای تغییر دهید تا در مقابل حمله بازتاب (Reflection Attack) نفوذپذیر شود. تشریح کنید که این تغییر شما به چه نحو کار می کند.
۳۱. برای ایجاد یک کلید سری بین آلیس و باب از روش «مبادله کلید دیفی-هلمن» استفاده شده است. آلیس آیتماهای (۱۹۱ و ۳ و ۷۱۹) را برای باب می فرستد. باب با (۵۴۳) پاسخ می دهد. عدد سری و محرمانه آلیس یعنی  $x$  معادل ۱۶ است. کلید سری و مشترک چیست؟
۳۲. اگر آلیس و باب هیچگاه یکدیگر را ملاقات نکرده و هیچ گواهینامه دیجیتالی یا کلید سری در اختیار نداشته باشند، می توانند توسط الگوریتم دیفی-هلمن یک کلید مشترک و سری ایجاد نمایند. تشریح کنید که در این الگوریتم چرا مقابله با حمله نوع man-in-the-middle بسیار دشوار است؟
۳۳. در پروتکل شکل ۸-۳۹ چرا مشخصه A به صورت آشکار (رمز نشده) به همراه کلید رمز شده نشست ارسال می شود؟
۳۴. در پروتکل شکل ۸-۳۹ اشاره کردیم که اگر هر قطعه متن پیام با ۳۲ بیت صفر شروع شود یک تهدید امنیتی به وجود خواهد آمد. فرض کنید که هر پیام با یک شماره تصادفی که به ازای هر کاربر تولید می شود و یک کلید سری که فقط برای کاربر و KDC مشخص است، شروع شود. آیا این ساختار مشکل حمله از طریق «متون شناخته شده» (Known Plaintext) را حل می کند؟ چرا؟
۳۵. در پروتکل «نیدهام-شرودر»، آلیس دو رشته چالش  $R_A$  و  $R_{A2}$  تولید می کند. این کار بی مورد و زائد به نظر

- می‌رسد. آیا یکی از آنها کافی نیست؟
۳۶. فرض کنید سازمانی برای احراز هویت کاربران از روش Kerberos استفاده کرده باشد. از دیدگاه «توانایی دسترسی به سرویسهای شبکه» و «امنیت»، چه اتفاقی می‌افتد اگر AS و TGS از کار بیفتند؟
۳۷. در پروتکل احراز هویت با کلید عمومی (شکل ۸-۴۳)، در پیام  $V$ ،  $R_B$  با کلید  $K_S$  رمزنگاری شده است. آیا این رمزنگاری لازم بوده و آیا می‌تواند به صورت رمز نشده و آشکار برگشت داده شود؟ پاسخ خود را شرح دهید.
۳۸. ترمینالهای فروش که از کارتهای اعتباری مغناطیسی و کدهای PIN استفاده می‌کنند یک اشکال جدی دارند: یک فروشنده متقلب می‌تواند دستگاه کارت‌خوان خود را به گونه‌ای دستکاری کند تا بتواند اطلاعات درون کارت و همچنین PIN افراد را بدست آورده و در جایی ذخیره نماید و از آنها برای معاملات آنی خود سوءاستفاده کند. نسل آینده ترمینالهای فروش از کارتهایی استفاده می‌کنند که بر روی آنها یک CPU کامل، صفحه کلید و یک نمایشگر کوچک قرار گرفته است. پروتکلی برای این سیستم ابداع کنید تا هیچ فروشنده بدخواهی نتواند آن را بشکند.
۳۹. دو دلیل بیاورید که چرا PGP پیامها را فشرده می‌کند.
۴۰. با فرض آن که همه در اینترنت از PGP استفاده کرده باشند، آیا یک پیام PGP می‌تواند برای هر آدرس دلخواه در اینترنت ارسال شود و به راحتی توسط تمام گیرندگان آن رمزگشایی گردد؟ پاسخ خود را تشریح کنید.
۴۱. در حمله‌ای که در شکل ۸-۴۷ نشان داده شده یک مرحله باقیمانده است. البته این مرحله برای موفقیت در DNS spoofing الزامی نیست ولیکن انجام چنین مرحله‌ای احتمال آن که پس از انجام عملیات شک دیگران برانگیخته شود را کاهش خواهد داد. به نظر شما این مرحله باقیمانده چیست؟
۴۲. پیشنهاد شده که برای خنثی کردن DNS spoofing که با استفاده از تخمین و تعیین ID عملی می‌شود، به جای استفاده از شماره‌ای که IDهای متوالی تولید می‌کند، از ID تصادفی استفاده شود. جنبه‌های امنیتی این روش را تشریح کنید.
۴۳. در پروتکل انتقال داده SSL، دو عدد (nonce) و یک شاه کلید اولیه بکار رفته است. این اعداد (nonce) چه مقادیری و چه کاربردی دارند؟
۴۴. تصویر شکل ۸-۵۵-ب در برگیرنده متن اسکی پنج نمایشنامه از شکسپیر است. آیا می‌توان به جای متن، یک قطعه موزیک را درون این تصویر از گورخرها پنهان کرد؟ اگر بله این کار چگونه ممکن است و چقدر موزیک می‌توان در آن ذخیره کرد؟ اگر جواب منفی است چرا؟
۴۵. آلیس یکی از کاربران دائمی یک «نامه‌پراکن ناشناس» از نوع Anonymous Remailer 1 بود. او پیامهای زیادی برای گروه خبری مورد علاقه‌اش alt.fanclub.alice می‌فرستاد و همه می‌دانستند که این پیامها از طرف آلیس می‌آید چرا که همه پیامهایش با یک اسم مستعار مشابه می‌رسیدند. با فرض آن که سرویس‌دهنده نامه‌پراکن (Remailer) به درستی کار کرده باشد، ترویدی نمی‌توانسته خودش را به جای آلیس جا بزند. پس از آن که این سرویس‌دهنده نامه‌پراکن (Remailer) از کار افتاد، آلیس به یک سرویس‌دهنده Cypherpunk Remailer تغییر سرویس‌دهنده داد و با این سرویس‌دهنده ارسالهای خود به گروه خبری را از سر گرفت. راهی ابداع کنید تا ترویدی نتواند خود را به جای آلیس جا زده و به جای او پیامهایی را برای گروه خبری ارسال نماید.
۴۶. در اینترنت به دنبال یک مورد جالب در خصوص موضوع «حفظ حریم خصوصی افراد» (Privacy) بگردید

و یک گزارش یک صفحه ای تهیه کنید.

۴۷. در اینترنت بدنبال چند مورد قضایی در خصوص «مالکیت حقوق معنوی آثار» (Copyright) جستجو کرده و خلاصه یافته های خود را در یک صفحه، گزارش نمایید.

۴۸. برنامه ای بنویسید که ورودی خود را با XOR کردن آن با یک کلید Keystream، رمز نماید. یک مولد اعداد تصادفی مناسب بنویسید (یا پیدا کنید) تا بتوانید Keystream تولید نمایید. برنامه شما باید همانند یک فیلتر عمل کرده و متن اصلی را از طریق ورودی استاندارد دریافت نماید و نتیجه رمز شده را به خروجی استاندارد بفرستد؛ (و بالعکس برای رمزگشایی همینطور عمل کند). برنامه فقط باید یک پارامتر از ورودی دریافت کند که آن هم کلیدی است که از آن بعنوان نقطه شروع مولد عدد تصادفی (Seed) استفاده می شود.

۴۹. پروسیجری بنویسید که رشته SHA-1 Hash متناظر با یک بلوک داده را محاسبه نماید. این پروسیجر باید دو پارامتر داشته باشد: یک اشاره گر به بافر ورودی و یک اشاره گر به یک بافر بیست بایتی خروجی. برای آن که شرح دقیق و جزئیات SHA-1 را بررسی کنید، در اینترنت FIPS 180-1 را جستجو نمایید که شامل شرح دقیق این استاندارد است.