```
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 خیلی‌ها WWW را World Wide Wait (انتظاری پایان جهانی) می‌خوانند. برای مقابله با این مشکل، محققان تکنیکهای مختلفی برای بالا بردن کارایی وب ابداع کرده‌اند. در این قسمت با برخی از این تکنیکها آشنا خواهید شد: حافظهٔ نهان، تکثیر سرویس‌دهنده، و شبکه‌های تحویل محتوا.

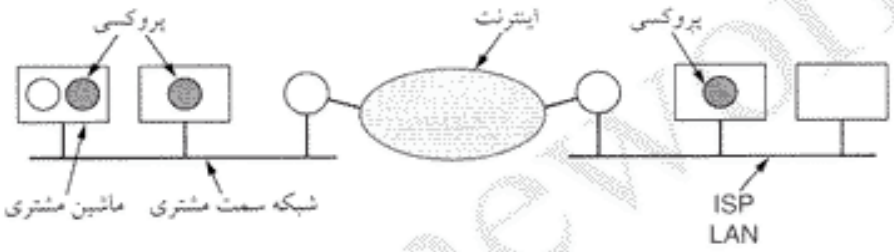
حافظهٔ نهان

یکی از ساده‌ترین روشهای بهبود کارایی وب، ذخیره کردن صفحات برای مراجعهٔ مجدد به آنهاست. این تکنیک بویژه برای صفحاتی که بازدیدکنندهٔ زیادی دارند (مانند، *www.yahoo.com* یا *www.cnn.com*)، مناسب است. ذخیره کردن صفحات وب برای مراجعات بعدی به **حافظهٔ نهان** (caching) معروف است. این حافظهٔ نهان معمولاً از طریق یک واسطه، موسوم به **پروکسی** (proxy) ، بکار گرفته می‌شود، و مرورگر بجای مراجعهٔ مستقیم به سرویس‌دهنده، صفحهٔ موردنظر را از این پروکسی درخواست می‌کند. اگر پروکسی این صفحه را در حافظهٔ نهان خود داشته باشد، بلافاصله آنرا به مرورگر برمی‌گرداند. ولی اگر نداشته باشد، صفحه را از سرویس‌دهنده گرفته، و (بعد از ذخیره کردن در حافظهٔ نهان خود) به مشتری تحویل می‌دهد.

در رابطه با حافظهٔ نهان دو سئوال مهم وجود دارد:

۱. چه کسی باید این کار را انجام دهد؟
۲. صفحات چه مدت باید نگه داشته شوند؟

سئوال اول جوابهای مختلفی می‌تواند داشته باشد. معمولاً PC ها هر کدام یک پروکسی دارند، که به کمک آن می‌توانند صفحاتی را که قبلاً دیده‌اند بسرعت پیدا کنند. در شبکه‌های محلی هم اغلب یک ماشین به این کار اختصاص داده می‌شود، تا اگر یکی از کاربران بخواهد صفحه‌ای را ببیند که کاربر دیگری قبلاً آنرا دیده است، نیازی به مراجعهٔ مستقیم به سرویس‌دهنده نباشد و بتوان آنرا از حافظهٔ نهان پروکسی در اختیار وی گذاشت. اکثر ISP ها هم برای سرعت دادن به دسترسی اینترنت کاربران خود از پروکسی استفاده می‌کنند. همهٔ این پروکسی‌ها همزمان با هم کار می‌کنند، بنابراین درخواستها ابتدا به اولین پروکسی می‌رسد. اگر این پروکسی صفحه را نداشته باشد، سراغ پروکسی شبکهٔ محلی می‌رود، و اگر آن هم صفحه را نداشته باشد، نوبت به پروکسی ISP می‌رسد. این آخری بالاخره هر طور که هست (از حافظهٔ نهان خودش، از حافظهٔ نهان سطح بالاتر، و یا مستقیماً از سرویس‌دهنده) صفحه را به درخواست‌کننده تحویل می‌دهد. به چنین روشی که در آن پروکسی‌ها بصورت زنجیره‌ای به هم متصلند، **حافظهٔ نهان سلسله‌مراتبی** (hierarchical chacing) گفته می‌شود (شکل ۷-۴۵ را ببینید).



شکل ۷-۴۵. حافظهٔ نهان سلسله‌مراتبی با سه پروکسی.

پاسخ سئوال دوم کمی پیچیده‌تر است. برخی از صفحات اصلاً نباید در حافظهٔ نهان ذخیره شوند. برای مثال، صفحه‌ای که قیمت سهام ۵۰ شرکت فعال بورس را نشان می‌دهد، هر ثانیه تغییر می‌کند. اگر چنین صفحه‌ای از حافظهٔ نهان به مشتری داده شود، اطلاعات آن کهنه و غیرقابل‌اعتماد 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 می‌بینید، قابلیت ذخیره‌سازی یک صفحه در حافظهٔ نهان می‌تواند بشدت به زمان وابسته باشد.

نکتهٔ کلیدی در تعیین مدت نگهداری صفحات در حافظهٔ نهان این است که کاربران تا چه حد می‌توانند کهنگی صفحات را تحمل کنند (از آنجائیکه این صفحه‌ها روی دیسک ذخیره می‌شوند، مقدار آنها چندان اهمیتی ندارد). اگر یک پروکسی صفحات را مدت زیادی در حافظهٔ نهان خود نگه ندارد (و آنها را بسرعت دور بیندازد)، صفحات آن بندرت کهنه می‌شوند، ولی از طرف دیگر کارایی خوبی نیز نخواهد داشت (چون صفحات کمی را از حافظهٔ نهان به مشتری‌ها می‌دهد). اما اگر صفحات را مدت زیادی نگه دارد، اکثر درخواستها را از این حافظهٔ نهان پاسخ می‌دهد، 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 حافظهٔ نهان نگه داشته شود، استفاده می‌کند. یکی از آنها بر پایهٔ فیلد *Last-Modified* سرآیند صفحه استوار است (شکل ۷-۴۳ را ببینید). اگر صفحه‌ای یک ساعت پیش تغییر کرده باشد، پروکسی آنرا یک ساعت در حافظهٔ نهان خود نگه می‌دارد. اگر صفحه‌ای آخرین بار یک سال پیش تغییر کرده باشد، پیداست که اطلاعات آن بسیار ثابت و پایدار است (مثلاً، صفحه‌ای که دربارهٔ تاریخ روم باستان است)، و می‌توان آنرا یک سال دیگر هم در حافظهٔ نهان نگه داشت، بدون اینکه نگرانی زیادی دربارهٔ کهنه شدن اطلاعات آن وجود داشته باشد. با اینکه روش شهودی فوق غالباً در عمل خوب کار می‌کند، ولی گاهی هم صفحات کهنه برمی‌گرداند.

رهیافت دیگر (که بر اساس یکی از ویژگیهای مدیریت حافظهٔ نهان در RFC 2616 بنا نهاده شده) قدری گرانتر است، ولی احتمال برگرداندن صفحات کهنه را از بین می‌برد. یکی از سودمندترین این ویژگیها سرآیند *If-Modified-Since* است، که پروکسی می‌تواند به سرویس‌دهنده بفرستد. در اینجا پروکسی نام صفحهٔ موردنظر و تاریخ آخرین تغییر آنرا (از سرآیند *Last-Modified*) به سرویس‌دهنده می‌فرستد. اگر صفحهٔ مزبور از این تاریخ به بعد تغییر نکرده باشد، سرویس‌دهنده پیام کوتاهی با مضمون *Not Modified* برمی‌گرداند (کُد وضعیت 304، جدول ۷-۴۲)، که به پروکسی می‌گوید «استفاده از صفحهٔ موجود در حافظهٔ نهان بی‌اشکال است.» اما اگر صفحه بعد از این تاریخ تغییر کرده باشد، سرویس‌دهنده صفحهٔ جدید را برمی‌گرداند. همانطور که می‌بینید، در این روش پروکسی همیشه باید با سرویس‌دهنده تماس بگیرد، که البته در مواقعی که اطلاعات صفحهٔ موجود در حافظهٔ نهان همچنان معتبر باشد، پاسخ سرویس‌دهنده بسیار کوتاه خواهد بود.

این دو روش را می‌توان بآسانی با هم ترکیب کرد. برای مدت زمان ΔT بعد از گرفتن صفحه، پروکسی صفحه را مستقیماً از حافظهٔ نهان به درخواست‌کنندگان می‌دهد. اما پس از آنکه مدتی گذشت، پروکسی با استفاده از پیام *If-Modified-Since* اعتبار صفحه را چک می‌کند. 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 حافظهٔ نهان ذخیره شوند، چون پارامترهای آنها هر بار تغییر می‌کند. برای حل این مشکل، سرویس‌دهنده با استفاده از یک مکانیزم خاص به تمام پروکسیهایی که بین آن و مشتری قرار دارند، دستور می‌دهد که بدون اطمینان از اعتبار و تازگی محتویات صفحه آنرا به مشتری ندهند. از این مکانیزم برای صفحاتی که محتویات آنها بسرعت تغییر می‌کند، نیز می‌توان استفاده کرد. در RFC 2616 مکانیزمهای مختلفی برای کنترل حافظهٔ نهان تعریف شده است.

روش دیگری که برای بهبود کارایی وب وجود دارد، **حافظهٔ نهان پیشگو** (proactive caching) است. در این روش، وقتی پروکسی صفحه‌ای را می‌آورد، تمام لینکهای موجود در آنرا بررسی کرده، و بطور خودکار آن صفحات را هم بار می‌کند، تا اگر به آنها نیاز شد، از قبل آمادگی داشته باشد. این تکنیک سرعت دسترسی به صفحات را بسیار بالا می‌برد، 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 حافظهٔ نهان بهیچوجه موضوع ساده‌ای نیست، و می‌توان ساعتها دربارهٔ آن صحبت کرد. در این زمینه کتابهای متعددی نوشته شده، که از میان آنها می‌توان 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 سرویس‌دهنده

حافظهٔ نهان یک تکنیک بهبود کارایی سمت-مشتری است، ولی تکنیکهای سمت-سرویس‌دهنده نیز وجود دارند. یکی از متداولترین این تکنیکها **تکثیر** (replication) محتویات سرویس‌دهنده در نقاط دور از هم است. به این تکنیک گاهی **آینه‌ای‌کردن** (mirroring) نیز گفته می‌شود.

در ساده‌ترین شکل، صفحهٔ اصلی یک سایت آینه‌ای دارای لینکهایی به نقاط مختلف (شمال، جنوب، شرق، و غرب) است. کاربری که روی یکی از این لینکها کلیک می‌کند، با توجه به ناحیه‌ای که در آن زندگی می‌کند، به سرویس‌دهندهٔ مربوطه هدایت می‌شود. از آن پس نیز تمام درخواستهای وی به این سرویس‌دهنده می‌روند.

سایتهای آینه‌ای معمولاً سایتهایی کاملاً ثابت و استاتیک هستند، و برای مدتهای مدید تغییر نمی‌کنند. معمولاً محتویات سایت اصلی کمابیش در سایت‌های آینه‌ای نیز کپی می‌شود (باستثنای آن بخشهایی که منطقاً باید 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، بی‌تماشاچی 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 دورافتادهٔ این ایالت به جنجال کشیده شد، این سایت تبدیل به یکی از پربیننده‌ترین سایتهای دنیا شد (و حتی برای مدتی بین پنج سایت ردهٔ اول قرار گرفت). لازم به گفتن نیست که این سایت تحمل چنین استقبالی را نداشت، و زیر بار این فشار بسرعت از پا در آمد.

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 **میزبانی‌وب** (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 (**شبکهٔ تحویل محتوا** - Content Delivery Network) به شرکتهای تولیدکنندهٔ محتوا (مانند سایتهای موسیقی، روزنامه‌ها، و دیگر جاهایی که مایلند محصولات خود را بسرعت 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 بسرعت به مطالب دلخواهشان دسترسی پیدا کنند (و این در دنیای پر رقابت خدمات اینترنتی، یعنی موفقیت). بهمین دلیل ISP ها برای بستن قرار داد 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 توزیع‌شده 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 تولیدکنندهٔ 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 تولیدکنندهٔ محتوا چند صفحهٔ سادهٔ HTML است، که

لینکهایی به فایلهای بزرگتر (مانند تصویر، صدا و ویدئو) دارد. این صفحات تغییر یافته روی سرویس‌دهندهٔ سایت تولیدکنندهٔ محتوا می‌مانند، و فقط فایلهای تصویر، صدا و ویدئو است که به سرویس‌دهنده‌های CDN منتقل می‌شود.

برای درک بهتر روش کار، صفحهٔ 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 می‌آید، و با نام *www.furryvideo.com/index.html* در سرویس‌دهندهٔ Furry Video ذخیره می‌شود.

```
<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)

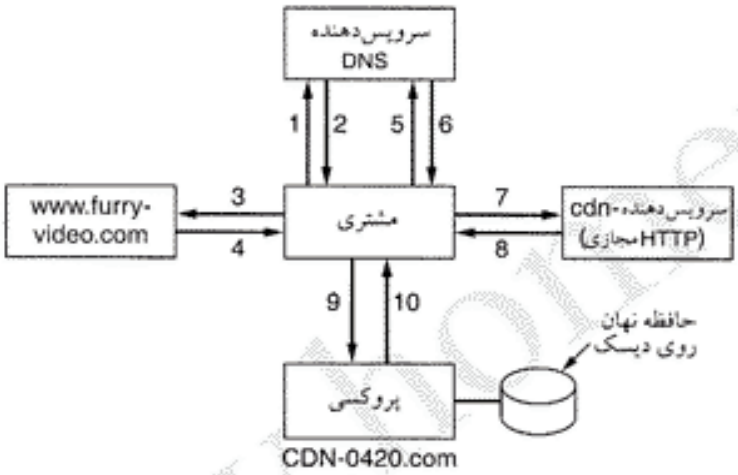
شکل ۷-۴۶. (الف) صفحهٔ وب اولیه. (ب) همان صفحه پس از تغییر در CDN .

وقتی کاربر URL این سایت را وارد می‌کند، DNS طبق روال معمول آدرس IP سایت *www.furryvideo.com* را برمی‌گرداند، و مرورگر صفحهٔ *index.html* را از سرویس‌دهندهٔ Furry Video می‌خواند. اما وقتی روی هر یک از لینکهای این صفحه کلیک شود، مرورگر آدرس سرویس‌دهندهٔ *cdn-server.com* را از DNS می‌پرسد، فایل موردنظر را از این سرویس‌دهنده درخواست می‌کند، و منتظر می‌ماند تا سرویس‌دهندهٔ *cdn-server.com* این فایل را برگرداند.

اما این اتفاق نمی‌افتد، چون *cdn-server.vom* چنین فایلی را ندارد. در اینجا CDN نقش یک سرویس‌دهندهٔ HTTP تقلبی را بازی می‌کند، و با بررسی درخواست رسیده می‌فهمد که تقاضا مربوط به کدام صفحه از کدام تولیدکنندهٔ محتواست. همچنین با بررسی آدرس IP درخواست کننده (و جستجو در پایگاه اطلاعاتی خود) در می‌یابد که کاربر در کدام ناحیهٔ جغرافیایی قرار دارد. با داشتن این اطلاعات، CDN می‌تواند تصمیم بگیرد که کدام سرویس‌دهندهٔ محتوا مناسبترین گزینه برای کاربر موردنظر است. این تصمیم‌گیری چندان هم که بنظر می‌آید ساده نیست، چون ممکنست نزدیکترین محل از نظر جغرافیایی نزدیکترین محل از نظر توپولوژی شبکه نباشد، و یا نزدیکترین سرویس‌دهنده از نظر توپولوژی شبکه در آن لحظه ترافیک بالایی داشته باشد. بعد از انتخاب

سرویس‌دهندهٔ مناسب، CDN یک پیام با کُد 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 صفحهٔ *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 (مرحلهٔ ۵) و برمی‌گرداند (مرحلهٔ ۶). با ارسال درخواست فایل *bears.mpg* به *cdn-server.com* (مرحلهٔ ۷)، به مشتری گفته می‌شود که باید به *CDN-0420.com* برود (مرحلهٔ ۸). وقتی مشتری به این محل مراجعه می‌کند (مرحلهٔ ۹)، پروکسی *CDN-0420.com* فایل مزبور را به وی تحویل می‌دهد (مرحلهٔ ۱۰). آنچه که باعث می‌شود این مکانیزم کار کند، مرحلهٔ ۸ است: جایی که یک سرویس‌دهندهٔ HTTP قلابی مشتری را به نزدیکترین پروکسی CDN تغییر مسیر می‌دهد.



شکل ۷-۴۷. مراحل پیدا کردن URL وقتی پای CDN در میان است.

سرویس‌دهندهٔ CDN (که مشتری از آنجا سر در می‌آورد) معمولاً یک پروکسی با حافظهٔ نهان بسیار بزرگ است (که قسمت اعظم محتویات در آن قرار دارند). با این تمهید (استفاده از پروکسی بجای یک سرویس‌دهندهٔ وب معمولی) کارایی سیستم بنحو قابل توجهی بالا می‌رود. برای کسب اطلاعات بیشتر دربارهٔ شبکه‌های تحویل محتوا به (Hull, 2002; and Rabinovich and Spatscheck, 2002) مراجعه کنید.

۷-۳-۶ 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 ایدهٔ ترکیب این دو تکنولوژی مطرح شود. ایدهٔ چنین سیستمی اولین بار توسط کنسرسیومی از شرکتهای نوکیا، اریکسون، موتورولا، و phone.com (Unwired

Planet سابق) پیش کشیده شد، و اکنون صدها شرکت دیگر آنرا پشتیبانی می‌کنند. این سیستم اکنون با نام WAP (پروتکل کاربردهای بیسیم – Wireless Application Protocol) شناخته می‌شود.

دستگاه WAP می‌تواند 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 می‌تواند از طریق لینکهای بیسیم به دروازهٔ WAP (WAP Gateway) وصل شده، و درخواست خود برای صفحات وب را ارسال کند. اولین جایی که برای این درخواست چک می‌شود، حافظهٔ نهان دروازه است: اگر صفحه در حافظهٔ نهان دروازه موجود باشد، به کاربر برگردانده می‌شود؛ اگر نباشد، از اینترنت (که ارتباط دروازه با آن از طریق کابل یا فیبر است) گرفته شده، و به کاربر داده می‌شود. به این ترتیب، WAP 1.0 اساساً یک سیستم سوئیچینگ مداری بود که هزینهٔ آن بر اساس زمان مکالمه دریافت می‌شد؛ البته مصرف‌کنندگان هم از چنین چیزی (آن هم برای دیدن صفحات وب روی مانیتور کوچک تلفنهای همراه) خوششان نیامد، و WAP 1.0 شکست خورد (که البته این شکست علتهای دیگری هم داشت). با این حال، ایدهٔ WAP (و رقیب آن، I-Mode) هنوز شکست نخورده، و بنظر می‌رسد WAP 2.0 بتواند با موفقیت همراه باشد. از آنجائیکه WAP 1.0 اولین تلاش برای پیاده‌سازی اینترنت بیسیم بود، نگاه مختصری به آن نمی‌تواند خالی از فایده باشد.

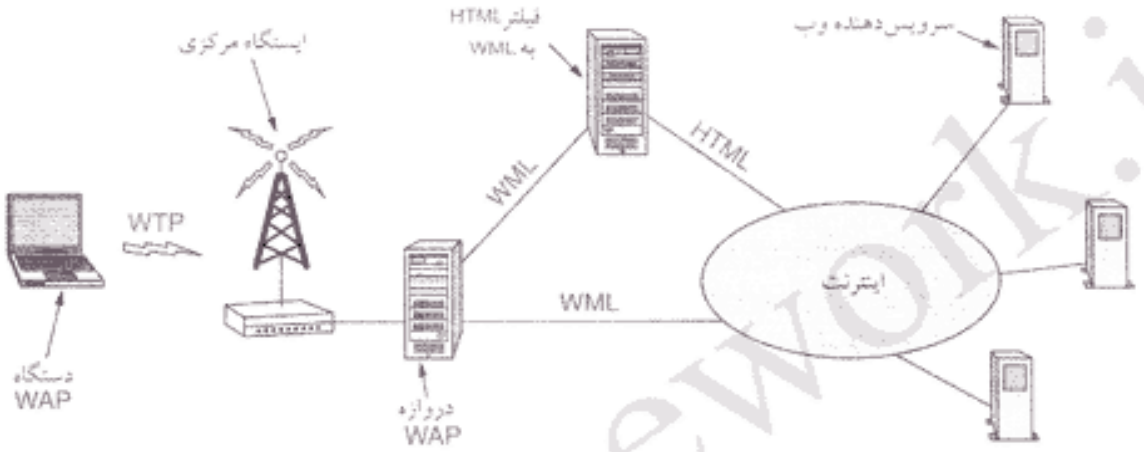
WAP اساساً یک پشتهٔ پروتکل است برای دسترسی وب، که برای وسایلی با پهنای باند کم، CPU کُند، مقدار حافظهٔ کم، و مانیتور کوچک بهینه شده است. این رهیافت که آشکارا با استاندارد PC های امروزی متفاوت است، منجر به پروتکلی متفاوت نیز شده است. لایه‌های پروتکل WAP را در شکل ۷–۴۸ ملاحظه می‌کنید.

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. پشتهٔ پروتکل WAP .

پائینترین لایه (و در واقع لایهٔ فیزیکی) از تمام سیستمهای تلفن همراه موجود (از جمله GSM ، D-AMPS ، و CDMA) پشتیبانی می‌کند. 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 لایه‌ای برای ایمنی داده‌ها (که در مخابرات بیسیم الزامیست) می‌آید. این لایه، WTLS (ایمنی لایهٔ انتقال بیسیم – Wireless Transport Layer Security)، زیرمجموعه‌ایست از SSL (که از ابداعات نت‌اسکیپ می‌باشد). سپس لایهٔ تبادل، WTP (پروتکل تبادل بیسیم – Wireless Transaction Protocol)، می‌آید که وظیفهٔ آن مدیریت درخواستها و پاسخهاست. این لایه جایگزین TCP شده، که بدلیل کارایی پائین نمی‌توان از آن در ارتباطات بیسیم استفاده کرد. پس از آن یک لایهٔ نشست می‌آید، که شبیه HTTP/1.1 است، ولی بمنظور کارایی بهتر تغییراتی در آن صورت گرفته است. و بالاخره، در بالای همهٔ اینها یک مینی مرورگر – در لایهٔ WAE (محیط کاربردی بیسیم – Wireless Application Environment) – قرار می‌گیرد.

بدون شک یکی از علل عدم پذیرش WAP (علاوه بر هزینهٔ 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 ۷–۴۹ را ببینید).



شکل ۷–۴۹. 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 ننشسته بودند. در آنجا خانمی بنام ماری ماتسوناگا رهیافت جدیدی برای وب بیسیم اختراع کرد، و نام آنرا I-Mode (Information-Mode) گذاشت. او توانست شرکت مخابرات بیسیم ژاپن (که زیرمجموعهٔ 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: هرجاکه هستید» است) رسماً شروع بکار کرد. بعد از سه سال، این سرویس اکنون متجاوز از ۳۵ میلیون مشترک دارد، که به ۴۰,۰۰۰ سایت وب I-Mode دسترسی دارند. مدیران این سیستم (در غیاب WAP) بیشترین لاف‌ها را دربارهٔ موفقیت مالی آن زده‌اند. اما اجازه دهید ببینیم I-Mode چیست و چگونه کار می‌کند.

سیستم I-Mode دارای سه مؤلفهٔ اصلی است: یک سیستم انتقال جدید، یک گوشی جدید، و یک زبان جدید برای طراحی صفحات وب. سیستم انتقال از دو شبکهٔ مجزا تشکیل شده است: شبکهٔ تلفن همراه سوئیچینگ بسته‌ای موجود (که تا حدی شبیه D-AMPS است)، و یک شبکهٔ سوئیچینگ مداری که اختصاصاً برای سرویس I-Mode ایجاد شده است. سرویس I-Mode از شبکهٔ سوئیچینگ بسته‌ای استفاده می‌کند و ارتباط آن دائمی

است (مانند ADSL یا کابل). بنابراین چیزی بنام هزینهٔ اتصال در آن وجود ندارد، و بجای آن هزینهٔ مشترک بر اساس بسته‌های فرستاده شده محاسبه می‌شود. در حال حاضر امکان استفادهٔ همزمان از هر دو شبکه وجود ندارد.

گوشی I-Mode شبیه گوشی‌های معمولی تلفن همراه است، با یک صفحهٔ مانیتور بزرگتر. حتی NTT DoCoMo در تبلیغات خود دستگاههای I-Mode را نسل جدید و پیشرفتهٔ تلفنهای همراه معرفی می‌کند، نه ترمینالهای بیسیم وب (چیزی که در واقع هستند). بسیاری از مشترکان این سیستم حتی نمی‌دانند که روی اینترنت هستند. اغلب آنها تصور می‌کنند که دستگاه I-Mode فقط یک تلفن همراه پیشرفته است. از آنجائیکه I-Mode فقط یک سرویس است، کاربران امکان برنامه‌نویسی با گوشی خود را ندارند (با اینکه گوشی آنها از کامپیوترهای سال ۹۵ قویتر است، و احتمالاً حتی می‌تواند ویندوز ۹۵ یا یونیکس را اجرا کند).

وقتی یک گوشی I-Mode روشن می‌شود، فهرستی از سرویسهای رسمی و تأئید شده به کاربر ارائه می‌کند. تعداد این سرویسها بیش از ۱۰۰۰ تاست، که به ۲۰ دسته تقسیم شده‌اند. هر سرویس که در واقع یک سایت I-Mode کوچک است، توسط یک شرکت مستقل اداره می‌شود. برخی از مهمترین دسته‌هایی که در این منو ظاهر می‌شوند، عبارتند از: ایمیل، اخبار، وضع آب و هوا، ورزش، بازی، خرید، نقشه، طالع‌بینی، سرگرمی، مسافرت، راهنمای اعمال مذهبی، انواع زنگهای تلفن، دستورات آشپزی، قمار، بانکداری خانگی، و قیمتهای بورس. این سرویسها تا حد زیادی نوجوانان و جوانان (با سنی حدود ۲۰ سال) را هدف گرفته‌اند، افرادی که عاشق اسباب‌بازیهای الکترونیکی (مخصوصاً اسباب‌بازیهای رنگارنگ) هستند. این موضوع که بیش از ۴۰ شرکت فقط زنگ تلفن می‌فروشند، می‌تواند گویای حقایق زیادی باشد. در اینجا هم محبوبترین سرویس ایمیل است، که اجازه می‌دهد پیامهایی تا ۵۰۰ بایت رد وبدل شود (نسبت به سرویس SMS با محدودیت ۱۶۰ بایتی، پیشرفت چشمگیری محسوب می‌شود). بازی نیز یکی دیگر از سرویسهای 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 دروازه‌های وب (مانند سایت Yahoo!) است، که به کاربر اجازه می‌دهند تا بدون وارد کردن URL و فقط با یک کلیک به سایتهای مختلف دسترسی پیدا کند.

سرویسهای 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، فرهنگهای لغت ژاپنی-انگلیسی باید به مقدار کافی لغت داشته باشند، سرویسهای زنگ تلفن باید بطور مرتب زنگهای جدیدی عرضه کنند، و هیچ سایتی نباید مطالب بیهوده و سبک (یا چیزی علیه NTT DoCoMo) ارائه کند (Frengle, 2002) . از طرف دیگر، سایتهای اینترنتی I-Mode مجازند هر کاری می‌خواهند بکنند.

مدل تجاری I-Mode تفاوت اساسی با اینترنت دارد. هزینهٔ اشتراک I-Mode فقط چند دلار در ماه است. از آنجائیکه در این سیستم هزینه‌ها بر اساس بسته‌های ارسال شده محاسبه می‌شود، در واقع با این شارژ اولیه کار چندانی نمی‌تواند کرد. کاربر می‌تواند هزینهٔ ماهیانهٔ ثابت بیشتری (برای تعداد بستهٔ بیشتر) بپردازد، و در شارژ بسته‌ها صرفه‌جویی کند، چون با بالا رفتن پرداخت ثابت ماهیانه تعداد بسته‌هایی که می‌تواند بفرستد، بصورت تصاعدی بالا خواهد رفت. اگر وسط ماه بسته‌های مجانی شما تمام شود، می‌تواند بصورت برخط بسته‌های بیشتری بخرید.

برای استفاده از یک سرویس باید مشترک آن شوید، کاری که فقط با یک کلیک روی آن و وارد کردن کُد PIN خود می‌توانید انجام دهید. اغلب سرویسهای رسمی هزینه‌ای معادل ۱ تا ۲ دلار در ماه دارند. NTT DoCoMo این پول را در صورتحساب تلفن شما منظور می‌کند، و بعد از کسر ۹٪ سهم خود بقیه (۹۱٪) را به شرکت ارائه‌کنندهٔ سرویس مسترد می‌کند. اگر یک سرویس غیررسمی ۱,۰۰۰,۰۰۰ مشترک داشته باشد، مجبور است هر ماه ۱,۰۰۰,۰۰۰ صورتحساب ۱ تا ۲ دلاری صادر کرده و برای مشتریان خود بفرستد. در حالیکه اگر سرویس خود را

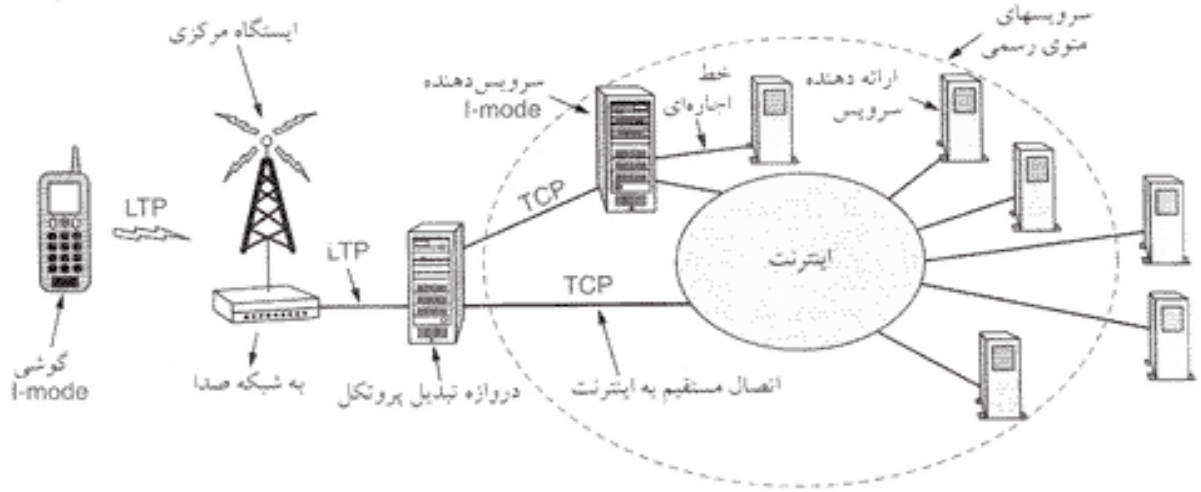
بصورت رسمی در آورد، NTT DoCoMo عملیات بانکی را بجای آن انجام داده و بازای آن ۹۰,۰۰۰ دلار از کل مبلغ کم می‌کند. صرفه‌جویی در هزینه‌های بانکی وصول حق اشتراک از مشتریان انگیزه‌ای بسیار جدی برای تمایل شرکتها به رسمی شدن است (و NTT DoCoMo هم که به حق خود می‌رسد!). رسمی شدن احتمال جذب مشتری را هم افزایش می‌دهد، چون سایت شما وارد منوی اولیهٔ 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 هزینهٔ بسته‌های 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 بالقوهٔ این سیستم در غرب (نوجوانان، دانشجویان، و کارمندان) دارای PC های بزرگ در خانه با دسترسی اینترنت پُرسرعت (حداقل 56 kbps) هستند، چیزی که در ژاپن چندان مرسوم نیست (اول بعلت فضای کوچک خانه‌های ژاپنی، و دوم بدلیل نرخهای کمرشکن NTT برای سرویسهای تلفن - ۷۰۰ دلار برای نصب یک خط تلفن، و ساعتی ۱/۵ دلار هزینهٔ 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 می‌خواهد برود، ۱ دلار حق اشتراک بدهد (پرداخت پول برای هر MB خواندن صفحات که دیگر بماند!). اغلب ISP های غربی (تحت فشار مشتریان خود) هزینهٔ دسترسی اینترنت را به یک شارژ ثابت ماهیانه (مستقل از مصرف واقعی آنان) تقلیل داده‌اند.

سوم اینکه، اغلب ژاپنی‌ها در زمان رفت و آمد به محل کار یا مدرسه (و در قطار) از I-Mode استفاده می‌کنند، در حالیکه در اروپا افراد کمی برای رفت و آمد از قطار استفاده می‌کنند (و این پدیده در آمریکا حتی از اروپا هم نادرتر است). استفاده از I-Mode در خانه، آن هم کنار یک مانیتور 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 نمی‌آید. با این حال، هیچکس چنین محبوبیتی را برای تلفنهای همراه هم پیش‌بینی نمی‌کرد، پس شاید در غرب هم جایی برای I-Mode باشد.

همانطور که قبلاً هم گفتیم، گوشی‌های I-Mode برای ارتباطات صدا از شبکهٔ موجود سوئیچینگ مداری، و برای ارتباطات داده از یک شبکهٔ سوئیچینگ بسته‌ای جدید استفاده می‌کنند. شبکهٔ داده بر اساس CDMA ، با بسته‌های ۱۲۸ بایتی و با نرخ 9600 bps کار می‌کند (شکل ۷-۵۰ را ببینید). گوشی با استفاده از LPT (**پروتکل سبک انتقال** - Lightweight Transport Protocol) با **دروازهٔ تبدیل پروتکل** (protocol conversion gateway)

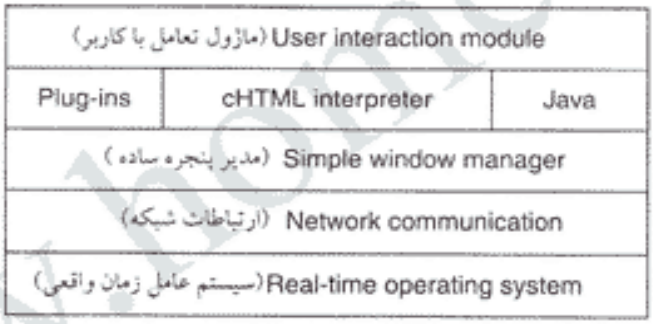


شکل ۷-۵۰. ساختار شبکهٔ دادهٔ I-Mode : پروتکلهای انتقال.

ارتباط می‌گیرد. این دروازه توسط یک فیبر نوری پهن‌باند به سرویس‌دهندهٔ I-Mode (که به تمام سرویسها متصل است) وصل می‌شود. وقتی کاربر یکی از سرویسهای رسمی را انتخاب می‌کند، این درخواست به سرویس‌دهندهٔ I-Mode فرستاده می‌شود، که (برای بهبود کارایی شبکه) اغلب صفحه‌ها را در حافظهٔ نهان خود دارد. درخواست برای سایتهایی که جزء منوی رسمی نیستند، مستقیماً به اینترنت فرستاده می‌شود.

گوشی‌های فعلی I-Mode با 100-MHz CPU کار می‌کنند، و چندین مگابایت حافظهٔ ROM ، چیزی حدود یک مگابایت RAM و یک صفحهٔ مانیتور کوچک دارند. وضوح مانیتور I-Mode بایستی حداقل 94 × 72 پیکسل باشد، ولی دستگاههایی با وضوح 160 × 120 پیکسل نیز عرضه شده‌اند. این مانیتورها از رنگ 8-bit پشتیبانی می‌کنند، 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 می‌دهد. گوشی‌های I-Mode (طبعاً) ماوس ندارند، و حرکت بین آیتمهای صفحه به کمک کلیدها صورت می‌گیرد.

ساختار نرم‌افزاری I-Mode را در شکل ۷-۵۱ ملاحظه می‌کنید. در پائین‌ترین لایه یک سیستم عامل بی‌درنگ (real-time) قرار دارد، که سخت‌افزار را کنترل می‌کند. پس از آن ماژول ارتباط با شبکه می‌آید، که از پروتکل LPT اختصاصی NTT DoCoMo استفاده می‌کند. بالای این لایه یک سیستم مدیریت پنجره (window manager) قرار دارد، که متن و گرافیک‌های ساده (فایلهای GIF) را نمایش می‌دهد. البته با مانیتوری به ابعاد حداکثر 160 × 120 پیکسل چیز زیادی برای مدیریت کردن وجود ندارد.



شکل ۷-۵۱. ساختار نرم‌افزاری I-Mode .

لایهٔ چهارم شامل مفسر صفحات وب (یعنی همان مرورگر) است؛ I-Mode فقط از زیرمجموعه‌ای از HTML بنام cHTML (HTML فشرده – compact HTML : که بر اساس HTML 1.0 قرار دارد) پشتیبانی می‌کند. برنامه‌های کمکی و افزودنی (مانند مرورگرهای معمولی) نیز در این لایه قرار می‌گیرند. یکی از برنامه‌های کمکی I-Mode مفسری بر اساس ویرایش اصلاح‌شدهٔ JVM است. و بالاخره، در بالای همهٔ لایه‌ها ماژول تعامل باکاربر (user interaction) قرار گرفته است، که وظیفهٔ آن ارتباط با کاربر از طریق صفحه کلید و مانیتور می‌باشد.

اجازه دهید نگاه دقیقتری به cHTML بیندازیم. همانطور که گفتیم، cHTML تقریباً همان HTML 1.0 است، که برای انطباق با گوشی‌های تلفن همراه تغییراتی در آن صورت گرفته است. استاندارد cHTML از طرف W3C برای نظرخواهی ارائه شده، ولی از آنجائیکه W3C علاقهٔ 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> ، <h*n*> ، ، ، ، <menu> ، ،
 ، <p> ، <hr> ، <img> ، <form> ، و <input> در cHTML هم وجود دارند، ولی <b> و <i> حذف شده‌اند.

یدی بنام *tel* به برچسب <a> (لینک به صفحات دیگر) اضافه شده، که با آن می‌توان با یک شماره تلفن تماس گرفت. از یک جهت *tel* شبیه *mailto* است، که وقتی کاربر آنرا انتخاب می‌کند، مرورگر با اجرای برنامهٔ ایمیل به کاربر اجازه می‌دهد به آدرس مشخص شده پیام بفرستد؛ با این تفاوت که در اینجا مرورگر شماره تلفن را می‌گیرد. بعنوان مثال، با این صفت می‌توان یک دفترچه تلفن تصویری ساخت، که وقتی روی هر یک از تصاویر کلیک می‌کنید، گوشی I-Mode شمارهٔ مربوطه را بگیرد. (URL های تلفنی در RFC 2806 مورد بحث قرار گرفته‌اند.)

مرورگر cHTML محدودیتهای دیگری نیز دارد، مثلاً از جاواسکریپت، فریم، شیوه‌نامه، و رنگ یا تصویر زمینه پشتیبانی نمی‌کند. مرورگرهای I-Mode تصاویر JPEG را هم نمایش نمی‌دهند، چون قدرت کافی برای خارج کردن سریع این تصاویر از حالت فشرده ندارند. صفحات cHTML می‌توانند اپلت جاوا داشته باشند، ولی اندازهٔ 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 چشمک‌زن درمی‌آورد. شاید این برچسب جانشین <b> (که در cHTML حذف شده) باشد، ولی بهر حال این اقدام NTT DoCoMo با استاندارد HTML که به ظاهر صفحات وب بی‌توجه است، همخوانی چندانی ندارد. برچسب جدید دیگر <marquee> است، که متن را در عرض صفحه حرکت می‌دهد.

به برچسب
 نیز صفت جدیدی بنام *align* اضافه شده، که برای نمایش مرتب کلمات روی مانیتورهای ۶سطر × 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 می‌کند تا این مشکل تا حد زیادی رفع شود (مانند آنچه در شکل ۷-۵۲ ب می‌بینید). البته جالبست بدانید که در زبان ژاپنی چیزی بنام شکستن کلمات وجود ندارد، چون هر علامت کانجی (kanji - الفبای ژاپنی) - که در یک سلول 10 × 9 پیکسل یا 12 × 12 پیکسل نوشته می‌شود - معادل یک کلمه در زبان انگلیسی است. در زبان ژاپنی یک خط متن می‌تواند از هر نقطه‌ای شکسته شود.

```
The time has com
e the walrus sai
d to talk of man
y things. Of sho
es and ships and
sealing wax of c
```

(الف)

```
The time has
come the walrus
said to talk of
many things. Of
shoes and ships
and sealing wax
```

(ب)

شکل ۷-۵۲. آشفتگی در صفحهٔ مانیتور 6 × 16 .

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 خندانک‌های شکل ۷-۶ هستند. در این زبان علامتهای زیبایی برای نمادهای طالع‌بینی، آبجو، همبرگر، پارک تفریحات، روز تولد، تلفن همراه، سگ، گربه، کریسمس، قلب شکسته، بوسه، خلق و خو، خواب، و البته زیبا و دلفریب وجود دارد.

ویژگی دیگر cHTML امکان انتخاب لینکهای صفحهٔ وب با استفاده از صفحه کلید است (که البته در جایی که ماوس وجود ندارد، ویژگی بسیار مهمی است). در شکل ۷-۵۳ نمونه‌ای از یک صفحهٔ cHTML که در آن از این ویژگی استفاده شده، را مشاهده می‌کنید.

با اینکه I-Mode در سمت مشتری با محدودیتهای زیادی مواجه است، در سمت سرویس‌دهنده هیچ

```
<html>
<body>
<h1> Select an option </h1>
<a href="messages.chtml" accesskey="1"> Check voicemail </a> <br>
<a href="mail.chtml" accesskey="2"> Check e-mail </a> <br>
<a href="games.chtml" accesskey="3"> Play a game </a>
</body>
</html>
```

شکل ۷-۵۳. نمونه‌ای از یک فایل cHTML .

محدودیتی ندارد. سرویس‌دهنده‌های I-Mode کامپیوترهای قدرتمندی هستند که از CGI ، Perl ، PHP ، JSP ، ASP (و هر چیزی که یک سرویس‌دهندهٔ وب معمولی می‌تواند داشته باشد) پشتیبانی می‌کنند.

در شکل ۷-۵۴ مقایسه‌ای بین سیستمهای نسل اول WAP و I-Mode بعمل آمده است. با آنکه برخی از این تفاوتها کوچک بنظر می‌رسند، ولی در جای خود اهمیت زیادی دارند. برای مثال، اغلب نوجوانان ۱۵ ساله کارت اعتباری ندارند، بنابراین امکان واریز هزینه‌های خرید الکترونیکی به صورتحساب ماهیانهٔ تلفن می‌تواند عامل تعیین‌کننده‌ای در جلب این قبیل مشتریان داشته باشد.

برای کسب اطلاعات بیشتر دربارهٔ I-Mode می‌توانید به (Frengle, 2002; and Vacca, 2002) مراجعه کنید.

I-mode	WAP	ویژگی
سرویس	پشته پونکل	ماهیت
گوشی	گوشی، PDA ، کامپیوتری	دستگاه
همیشه برقرار	تلفنی	دسترسی
سوئیچینگ مداری و بسته ای	سوئیچینگ مداری	شبکه زیرین
9600 bps	9600 bps	نرخ داده
رنگی	سیاه و سفید	صفحه نمایش
cHTML	WML (XML application)	زبان علامتگذاری
ندارد	WML script	زبان اسکریپت نویسی
بربسته	بر دقیقه	هزینه
صورتحساب تلفن	کارت اعتباری	پرداخت خریدها
بلی	خیر	علایم تصویری
در مالکیت NTT DoCoMo	استاندارد باز مجمع WAP	استاندارد سازی
ژاپن	اروپا، ژاپن	محل استفاده
افراد جوان	تاجران و صنعتگران	کاربران نوعی

شکل ۷-۵۴. مقایسه‌ای بین سیستمهای نسل اول WAP و I-Mode .

نسل دوم وب بیسیم

WAP 1.0 ، که بر اساس استانداردهای پذیرفته‌شدهٔ بین‌المللی طراحی شده بود، ابزاری جدی برای افراد پُرتحرک محسوب می‌شد. اما، متأسفانه شکست خورد. در مقابل، I-Mode بازیچه‌ای بود برای نوجوانان ژاپنی، که هیچ استانداردی 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، باخته‌اید. از آن سو، NTT DoCoMo هم

فهمید که یک سیستم اختصاصی بسته (که فقط برای گوشی‌های کوچک و فرهنگ ژاپنی طراحی شده باشد) نمی‌تواند شانسی 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 پیوستهٔ 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.

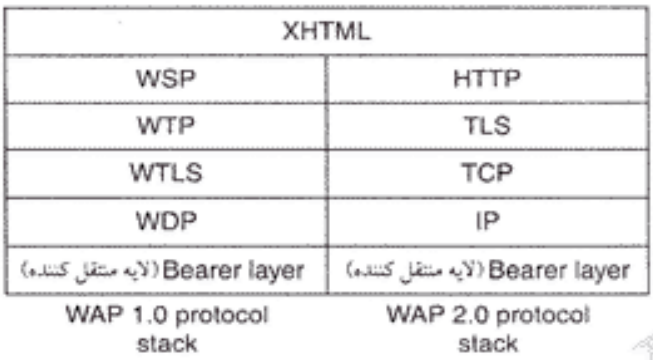
محبوبیت فوق‌العادهٔ کاراکترهای اموجی در I-Mode، کنسرسیوم WAP را تشویق کرد که ۲۶۴ کاراکتر اموجی در WAP 2.0 بگنجاند. این کاراکترها در زمینه‌های متنوعی از قبیل حیوانات، وسایل خانگی، لباس، احساسات، غذا، بدن انسان، جنسیت، نقشه، موسیقی، گیاهان، ورزش، وقت، آب و هوا، ابزار، وسایط نقلیه، و اسلحه طراحی شده‌اند. 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 وسیلهٔ 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 پشتهٔ 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 پنجره‌های ثابت 64-KB ، (۲) نداشتن شروع آهسته، (۳) سقف ۱۵۰۰ بایتی برای MTU ، و (۴) تفاوت جزئی در الگوریتم ارسال مجدد (این تغییرات برای ساده‌تر شدن کُد پروتکل انجام شده‌اند). TLS یک پروتکل ایمنی لایهٔ انتقال (Transport Layer Security) است، که توسط IETF استاندارد شده است (برای توضیحات بیشتر به فصل ۸ مراجعه کنید). بسیاری از دستگاههای WAP 2.0 همزمان از هر دو پشته پروتکل پشتیبانی خواهند کرد (شکل ۷-۵۵ را ببینید).



شکل ۷-۵۵. WAP 2.0 از دو پشته پروتکل پشتیبانی می‌کند.

تفاوت دیگر WAP 2.0 با WAP 1.0 در زبان علامتگذاری آنهاست. WAP 2.0 از XHTML ساده (که برای دستگاههای کوچک بیسیم طراحی شده) پشتیبانی می‌کند. از آنجائیکه NTT DoCoMo هم قبول کرده که از XHTML ساده پشتیبانی کند، با نوشتن صفحات خود به این زبان می‌توانید مطمئن باشید که آنها در اینترنت و دستگاههای 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 کلمه‌ای هم دربارهٔ XHTML ساده بگوئیم. این زبان برای تلفنهای همراه، تلویزیونها، دستگاههای PDA ، ماشینهای فروش کالا، فراخوان‌ها، اتومبیل‌ها، کنسولهای بازی، و حتی ساعت در نظر گرفته شده است. بهمین دلیل در این ویرایش از شیوه‌نامه، اسکریپت، یا فریم خبری نیست، ولی اکثر برچسب‌های استاندارد وجود دارند. این برچسب‌ها به ۱۱ ماژول تقسیم شده‌اند، که برخی اجباری و برخی دیگر اختیاری‌اند (تمام این ماژولها در XML تعریف شده‌اند). این ماژولها را، همراه با مثالهایی از هر کدام، در شکل ۷-۵۶ ملاحظه می‌کنید. برای کسب اطلاعات بیشتر دربارهٔ این برچسب‌ها می‌توانید به *www.w3.org* مراجعه کنید.

علیرغم توافق WAP و I-Mode بر سر استفاده از XHTML ساده، رقیب جدیدی آنها را تهدید می‌کند: 802.11 . نسل دوم وب بیسیم با سرعت 384 kbps کار می‌کند، که از 9600 bps نسل اول خیلی بهتر است، ولی در مقایسه با سرعت 11 یا 54 Mbps دستگاههای 802.11 رنگ می‌بازد. البته 802.11 همه جا حضور ندارد، ولی با تصمیم صاحبان رستورانها، هتلها، فروشگاهها، شرکتها، فرودگاهها، ایستگاههای اتوبوس، موزه‌ها، 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 کافیست به نزدیکترین کافه تریا بروید تا بتوانید ضمن خوردن یک فنجان قهوه، ایمیلهای خود را هم چک کنید. دور نیست روزی که مغازه‌ها (کنار آرم کارتهای اعتباری قابل قبول) برای جلب مشتری بیشتر آرم 802.11 را در ویترین یا کنار در ورودی خود قرار دهند، و مردم هم (مثل صحرانشینان که بدنبال آب بیابانها را زیر پا می‌گذاشتند) 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, h*n*, kbd, p, strong	اطلاعات	بلی	متن
a	ابرلینک	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	شی ء
meta	اطلاعات اضافی	خیر	اطلاعات اضافی
link	شبیه <a>	خیر	لینک
base	نقطه شروع URL	خیر	مبنا

شکل ۷-۵۶. ماژول‌ها و برچسب‌های اصلی XHTML.

شاید رستورانها و کافه‌تریاها خیلی زود برای نصب ایستگاههای مرکزی 802.11 دست بکار شوند، ولی بنظر نمی‌رسد کشاورزان به این زودی‌ها چنین کاری بکنند، بهمین دلیل پوشش یکپارچهٔ 802.11 به مناطق شهری (و حداکثر حومهٔ 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.

۷-۴ چندرسانه‌ای

وب بیسیم یکی از تکنولوژیهای جدید و مهیج است، ولی تنها تکنولوژی نیست. برای خیلی‌ها چندرسانه‌ای (multimedia) جام مقدس شبکه است. وقتی اسم چندرسانه‌ای می‌آید، همهٔ چشم‌ها خیره می‌شود (چون برای هر کس چیزی دارد). از آنجائیکه چندرسانه‌ای به پهنای باند زیادی نیاز دارد، فعلاً فقط روی شبکه‌های ثابت کار می‌کند، و برای دیدن آن روی لینکهای بیسیم باید چند سال دیگر منتظر ماند.

از نظر لغوی، چندرسانه‌ای یعنی دو یا چند رسانه. حتی ناشر همین کتاب حاضر هم می‌تواند برای آن بعنوان کتابی چندرسانه‌ای تبلیغ کند، چون هم متن دارد هم تصویر (شکل). با این حال، وقتی اغلب مردم این اصطلاح را بکار می‌برند، منظورشان ترکیبی از چند رسانهٔ پیوسته (continuous media) - رسانه‌هایی که می‌توان آنها در یک فاصلهٔ زمانی مشخص، بهمراه نوعی تعامل با کاربر، پخش کرد - است. در عمل، چندرسانه‌ای بیشتر به ترکیبی از صدا و ویدئو (تصاویر متحرک) اطلاق می‌شود.

با این همه، بسیاری از مردم به صدای تنها (مثلاً، تلفن یا رادیوی اینترنتی) هم چندرسانه‌ای می‌گویند، که پیداست چنین نیست. نام رسانهٔ جویباری (streaming media) برای این قبیل رسانه‌ها مناسبتر است، ولی اجازه دهید ما هم با افکار عمومی همراهی کنیم و آنرا همان چندرسانه‌ای بخوانیم. در قسمتهای آینده خواهید دید که کامپیوترها چگونه صدا و ویدئو را پردازش می‌کنند، چگونه آنها را فشرده می‌کنند، و چگونه می‌توان از این تکنولوژیها استفاده کرد. برای یک بحث مفصل (سه جلدی) دربارهٔ شبکه‌های چندرسانه‌ای کتابهای Steinmetz (and Nahrstedt, 2002; Steinmetz and Nahrstedt, 2003; and Steinmetz and Nahrstedt, 2003b) را ببینید.

۷-٤-۱ مقدمه‌ای بر صدای دیجیتال

صدا یک موج فشاری یک بُعدی است. وقتی این موج وارد گوش می‌شود، پردهٔ گوش را به ارتعاش در آورده و باعث می‌شود 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 می‌رود. درک ما از صدا توسط مغز (و یا تفسیر این سیگنالهای عصبی) انجام می‌شود. به همین ترتیب، وقتی موج صوتی به میکروفون برخورد می‌کند، میکروفون آنرا به یک سیگنال الکتریکی (که دامنهٔ آن تابعی از زمان است) تبدیل می‌کند. پردازش، ذخیره‌سازی، انتقال و بازتولید این سیگنالها یکی از مهمترین زمینه‌های تحقیقاتی سیستمهای چندرسانه‌ای است.

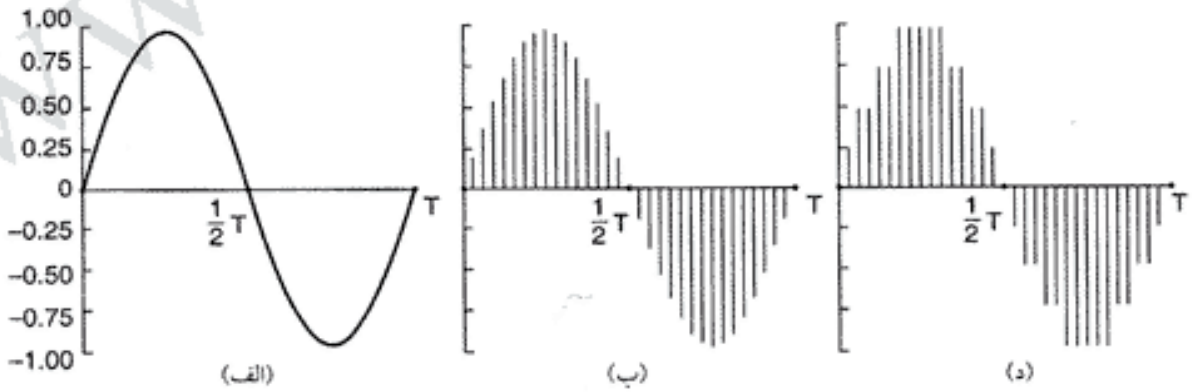
محدودهٔ فرکانسی گوش انسان بین 20 Hz تا 20,000 Hz است. برخی از حیوانات (بویژه سگ و خفاش) می‌توانند فرکانسهای بالاتر را نیز بشنوند. توانایی گوش در شنیدن اصوات لگاریتمی است، بنابراین نسبت دو صوت با قدرتهای A و B بصورت dB (دسی‌بل) با فرمول زیر بیان می‌شود:

$$dB = 10\log_{10}(A/B)$$

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 آستانهٔ درد معادل 120 dB است (یعنی تفاوتی در حد ۱ میلیون برابر).

حساسیت گوش به صداهای گذرا (در حد چند میلی‌ثانیه) بسیار شگفت‌آور است (حتی چشم نیز نورهایی در این فاصلهٔ کوتاه را تشخیص نمی‌دهند). در نتیجه، لرزش صدا (حتی برای چند میلی‌ثانیه) می‌تواند در کیفیت انتقال چندرسانه‌ای اثر بگذارد، در حالیکه چنین لرزشهایی روی کیفیت تصاویر تقریباً بی‌تأثیر است.

برای تبدیل صدا به سیگنال دیجیتال می‌توان از یک ADC (مبدل آنالوگ‌دیجیتال – Analog Digital Converter) استفاده کرد. مبدل ADC ولتاژ الکتریکی را بعنوان ورودی گرفته، و یک خروجی دودویی (باینری) تولید می‌کند. شکل ۷-۵۷(الف) یک موج سینوسی را نشان می‌دهد. برای نمایش دیجیتالی این سیگنال، می‌توانیم در فواصل ΔT ثانیه از آن نمونه‌برداری کنیم (شکل ۷-۵۷ ب). اگر موج صوتی سینوسی خالص نباشد ولی بتوان آنرا بصورت مجموع خطی چند موج سینوسی (با بیشترین فرکانس f) نمایش داد، قضیهٔ نایکوئیست (فصل ۲) می‌گوید که فرکانس $2f$ برای نمونه‌برداری آن کافیست. نمونه‌برداری با فرکانسهای بالاتر اطلاعات بیشتری بدست نمی‌دهد، چون فرکانسهایی که این نمونه‌برداری می‌تواند آشکار کند، در موج اصلی وجود ندارند.



شکل ۷-۵۷. (الف) یک موج سینوسی. (ب) نمونه‌برداری از موج سینوسی. (ج) کوانتیزه کردن نمونه‌ها بصورت ۴ بیتی.

نمونه‌های دیجیتالی هرگز دقیق و کامل نیستند. در مثال شکل ۷-۵۷ (ج) فقط ۹ مقدار مجاز (از 1.00- تا 1.00+ باگامهای 0.25) وجود دارد (و همانطور که می‌دانید، برای نمایش ۹ مقدار به ۴ بیت دودویی نیاز داریم). با

نمونه‌های ۸بیتی 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 **نویزکوانتیزه‌کردن** (quantization noise) معروف است. اگر این خطا زیاد باشد، گوش می‌تواند آنرا تشخیص دهد.

تلفن و دیسکهای صوتی دو نمونهٔ آشنا از کوانتیزه کردن صوت هستند. مدولاسیون کُد پالس (PCM)، که در سیستمهای تلفن بکار می‌رود، از نمونه‌های ۸بیتی 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 فرکانسهایی 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 محدودهٔ 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 نمونهٔ ۱۶بیتی 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 باکیفیت 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 محدودهٔ 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 آروارهٔ 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 شکل‌موج واژه‌ها) از مدلهای ساده شدهٔ مجرای صوتی (با پارامترهای محدودتری از قبیل اندازه و شکل حفره‌های صوتی) استفاده می‌شود. طرز کار این سیستمها در حیطهٔ بحث کتاب حاضر نیست.

۲.۷.۴ فشرده‌سازی صدا

همانطور که دیدید برای انتقال صدای استریو باکیفیت 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 لایهٔ سوم (MPEG audio layer 3 - MP3) از همه قویتر و معروفتر است. حجم زیادی از موسیقی با این فرمت روی اینترنت وجود دارد، که البته همهٔ آنها قانونی نیستند، و همین منجر به دعواهای 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 دربارهٔ آن هم صحبت خواهیم کرد.

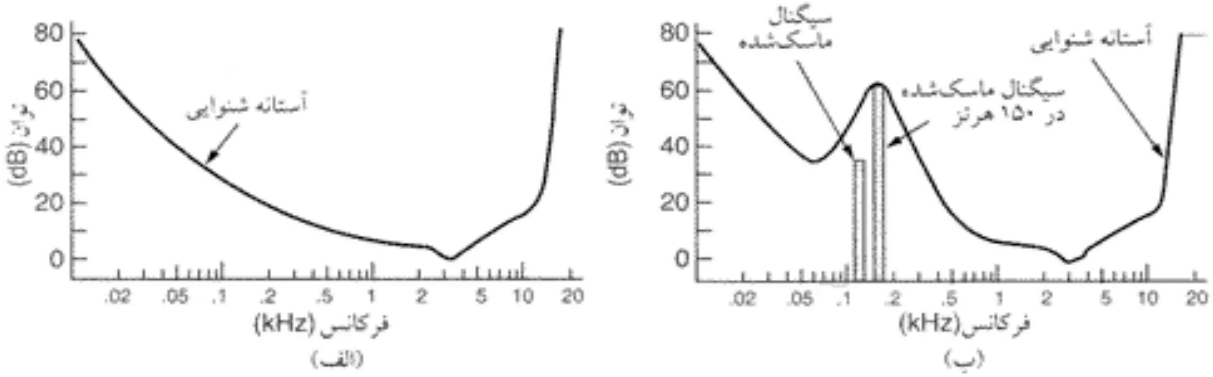
فشرده‌سازی صدا را به دو روش می‌توان انجام داد. در **کُدگذاری شکل‌موج** (waveform coding) سیگنال

صوتی بصورت ریاضی و با استفاده از تبدیل فوریه، به فرکانسهای تشکیل‌دهندهٔ آن تجزیه می‌شود. در شکل ۲-۱ (الف) نمونه‌ای از یک موج و دامنه‌های فوریهٔ آنرا می‌بینید. سپس دامنهٔ هر مؤلفه به ساده‌ترین شکل ممکن کُد می‌شود. هدف از این کار بازسازی شکل‌موج با حداقل بیت‌های ممکن است.

روش دیگر، **کُدگذاری ادراکی** (perceptual coding) ، سیگنال ورودی را با استفاده از نقائص سیستم صوتی انسان بگونه‌ای کُد می‌کند که شنونده متوجه تفاوت آنها نشود (حتی اگر این سیگنالها در اسیلوسکوپ بهیچوجه یکسان دیده نشوند). کُدگذاری ادراکی بر اساس تحقیقات **روان‌شنوایی** (psychoacoustics) – دانش درک انسانها از صوت – بنا نهاده شده است. فرمت MP3 از کُدگذاری ادراکی استفاده می‌کند.

نکتهٔ کلیدی در کُدگذاری ادراکی این است که برخی از اصوات می‌توانند صداهای دیگر را **بپوشانند**. 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 می‌شوند. در این حال دیگر هیچکس نمی‌تواند 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 – **ماسک‌فرکانسی** (frequency masking) می‌گویند. 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) بهرهٔ شنوایی خود را با شروع صدای چکشها پائین می‌آورد، و بازگشت آن به حالت قبل (بعد از خاموش شدن چکشها) مدتی طول خواهد کشید. به این پدیده هم **ماسک‌موقتی** (temporal masking) گفته می‌شود.

برای دیدن تأثیر کمی این پدیده‌ها، یک آزمایش ترتیب می‌دهیم. در یک اتاق کاملاً ساکت، فردی گوشی‌های متصل به کارت صوتی کامپیوتر را به گوش می‌گذارد. کامپیوتر یک موج سینوسی کم قدرت با فرکانس 100 Hz تولید می‌کند، و بتدریج قدرت آنرا بالا می‌برد. به این فرد گفته‌ایم 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 آستانهٔ شنوای انسان) تکرار می‌کند. با تکرار این آزمایش برای افراد مختلف و محاسبهٔ متوسط قدرت لازم برای شنیده شدن صدا در هر فرکانس، نمودار لگاریتمی شکل ۷-۵۸(الف) را رسم کرده‌ایم. این نمودار بروشنی نشان می‌دهد که هیچ نیازی به کُد کردن فرکانسهایی که قدرت آنها زیر آستانهٔ شنوایی انسان باشد، نیست. برای مثال، اگر قدرت یک سیگنال 100 Hz فقط 20 dB باشد، می‌توان آنرا حذف کرد بدون اینکه تأثیری روی کیفیت خروجی بگذارد، چون طبق شکل ۷-۵۸ (الف) این سیگنال زیر آستانهٔ شنوایی انسانها قرار دارد.



شکل ۷-۵۸. (الف) نمودار آستانهٔ شنوایی بر حسب فرکانس. (ب) اثر ماسک فرکانسی.

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 زمینهٔ فرکانس تست پخش می‌شود. همانطور که به روشنی می‌توان دید (شکل ۷-۵۸ب)، آستانهٔ شنوایی فرکانسهای نزدیک 150 Hz بالاتر رفته است.

نتیجهٔ این آزمایش آن است که، با دنبال کردن فرکانسهای نزدیک به هم می‌توان سیگنالهای ضعیفتر را حذف کرده، و در تعداد بیت‌های موردنیاز 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 فوریهٔ صدا به سیگنالهای سینوسی و حذف تمام سیگنالهایی که ماسک شده‌اند.

با این زمینهٔ تئوریک، اکنون می‌توانیم نشان دهیم که کُدگذاری 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 استریم ورودی).
۲. تک‌صدایی دوبل (مثلاً، حاشیهٔ صوتی انگلیسی و ژاپنی).
۳. استریوی منفصل (فشرده‌سازی جداگانهٔ هر کانال).
۴. استریوی متصل (استفادهٔ کامل از افزونگی بین‌کانالها).

ابتدا، نرخ بیت خروجی انتخاب می‌شود. با الگوریتم MP3 می‌توان یک CD استریوی راک اند رول را تا 96 kbps فشرده کرد، بطوریکه حتی دوآتشه‌ترین طرفداران این موسیقی هم متوجه افت کیفیت نشوند. برای یک کنسرت پیانو به حداقل 128 kbps نیاز داریم، چون نسبت سیگنال-به-نویز موسیقی راک اند 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 مرحلهٔ بعد، تعداد بیت‌های موجود بین این باندها تقسیم می‌شود (با این معیار که به توانهای طیفی ماسک‌نشده بیت‌های بیشتری برسد، باندهای ماسک‌نشدهٔ کم‌قدرت‌تر بیت‌های کمتری بگیرند، و باندهای ماسک‌شده اصلاً چیزی نگیرند). در پایان هم، بیت‌های حاصله با استفاده از روش هافمن - اختصاص کُدهای کوتاهتر به اعدادی که بیشتر تکرار شده‌اند، و اختصاص کُدهای بلندتر به اعداد کمتر تکرار شده - کُد می‌شوند.

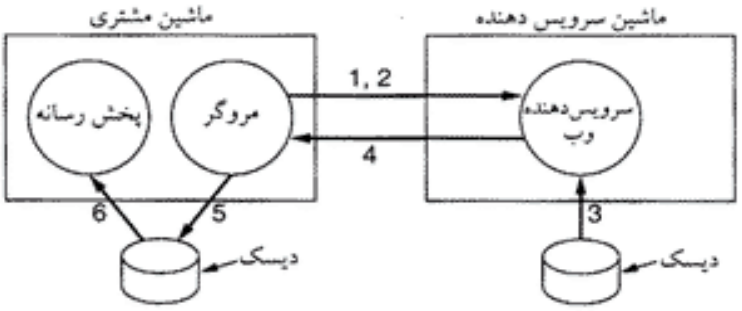
در واقع، قصهٔ فشرده‌سازی مفصلتر از اینهاست. تکنیکهای مختلفی برای کاهش نویز، یکنواخت‌سازی، و بهره‌گیری از افزونگی بین‌کانالها وجود دارد، که از حوزهٔ بحث این کتاب خارج هستند. برای کسب اطلاعات بیشتر در زمینهٔ ریاضیات فشرده‌سازی صدا به (Pan, 1995) مراجعه کنید.

۳.٤.۷ صدای جویباری

در این قسمت سه تا از کاربردهای صدای دیجیتال را مورد بررسی قرار می‌دهیم. اولین آنها صدای‌جویباری (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 تازه‌کاری که بدنبال شهرت‌اند)، و برای بعضی دیگر باید پول داد (البته روی این سایتها هم نمونه‌های مجانی یافت می‌شود). ساده‌ترین روش پخش موسیقی را در شکل ۷-۵۹ ملاحظه می‌کنید.



۱- برقراری اتصال TCP
۲- ارسال درخواست HTTP GET
۳- سرویس دهنده فایل را از دیسک می‌خواند
۴- فایل فرستاده می‌شود
۵- مرورگر فایل را روی دیسک می‌نویسد
۶- برنامه پخش فایل را از دیسک خوانده و پخش می‌کند

شکل ۷-۵۹. یک روش ساده برای پخش موسیقی در صفحات وب.

کار با کلیک کردن روی لینک موسیقی دلخواه شروع می‌شود، و در اینجا مرورگر وارد صحنه می‌شود. در مرحلهٔ ۱ مرورگر یک اتصال TCP به سرویس‌دهندهٔ وب هدف برقرار می‌کند، و در مرحلهٔ ۲ با فرمان *GET* فایل موسیقی را از آن درخواست می‌کند. سپس (مراحل ۳ و ۴)، سرویس‌دهنده فایل خواسته شده را (که می‌تواند با فرمت MP3 یا هر فرمت دیگری باشد) از روی دیسک خوانده و برای مرورگر می‌فرستد. اگر این فایل از حجم حافظهٔ سرویس‌دهنده بزرگتر باشد، آنرا بصورت قطعه قطعه خواهد فرستاد.

مرورگر با استفاده از نوع MIME فایل دریافت‌شده (مثلاً *audio/mp3*)، تشخیص می‌دهد که چگونه باید آنرا پخش کند. مرورگرها معمولاً برای این کار از برنامه‌های کمکی مانند RealOne Player ، Windows Media Player ، یا Winamp کمک می‌گیرند. از آنجائیکه متداولترین روش ارتباط با برنامه‌های کمکی نوشتن فایل روی محلی موقتی است، مرورگر ابتدا فایل را روی دیسک ذخیره می‌کند (مرحلهٔ ۵). سپس، برنامهٔ پخش موسیقی را اجرا کرده و نام فایل را به آن می‌دهد. با این کار برنامهٔ پخش موسیقی شروع به خواندن فایل از روی دیسک، و پخش آن می‌کند (مرحلهٔ ۶).

این روش هیچ اشکالی ندارد و در عمل کار هم می‌کند، ولی مشکل اینجاست که تمام فایل باید قبل از شروع پخش خوانده شود. اگر یک فایل موسیقی 4 MB باشد (اندازه‌ای معمولی برای فایلهای MP3)، و کاربر با مودم 56 kbps به اینترنت متصل شده باشد، قبل از شروع پخش موسیقی باید ۱۰ دقیقه انتظار بکشد. اغلب موسیقی‌دوستان تحمل چنین انتظاری را ندارند.

برخی از سایتهای وب برای حل این مشکل (بدون آنکه روش دسترسی به موسیقی را تغییر دهند) از روش ذیل استفاده می‌کنند. در این روش، لینک صفحهٔ وب مستقیماً به فایل موسیقی اشاره نمی‌کند، بلکه یک متافایل (metafile - فایل بسیار کوچکی که فقط نام فایل موسیقی در آن است) را مشخص می‌کند. یک متافایل چنین شکلی دارد:

```
rtsp://joes-audio-server/song-0025.mp3
```

وقتی مرورگر این فایل یک‌خطی را دریافت کرد، طبق معمول آنرا روی دیسک نوشته و پس از اجرای برنامهٔ کمکی پخش موسیقی، نام فایل مزبور را به آن می‌دهد. برنامهٔ پخش بعد از خواندن فایل متوجه می‌شود که این یک URL است، بنابراین به سرویس‌دهندهٔ *joes-audio-server* متصل شده و آهنگ مشخص شده را درخواست می‌کند. توجه کنید که در اینجا دیگر مرورگر نقشی ندارد.

در اغلب موارد، سرویس‌دهندهٔ مشخص شده در متافایل همان سرویس‌دهندهٔ وب نیست، و در واقع حتی یک

سرویس‌دهندهٔ HTTP هم نیست، بلکه نوع خاصی از **سرویس‌دهندهٔ رسانه** (media server) است. در مثال بالا، سرویس‌دهندهٔ رسانه از پروتکل RTSP (**پروتکل جویباری بی‌درنگ** – Real Time Streaming Protocol) استفاده می‌کند (به نام پروتکل *rtsp* در ابتدای URL دقت کنید). این پروتکل در RFC 2326 تعریف شده است.

برنامهٔ پخش چهار وظیفهٔ اصلی دارد:

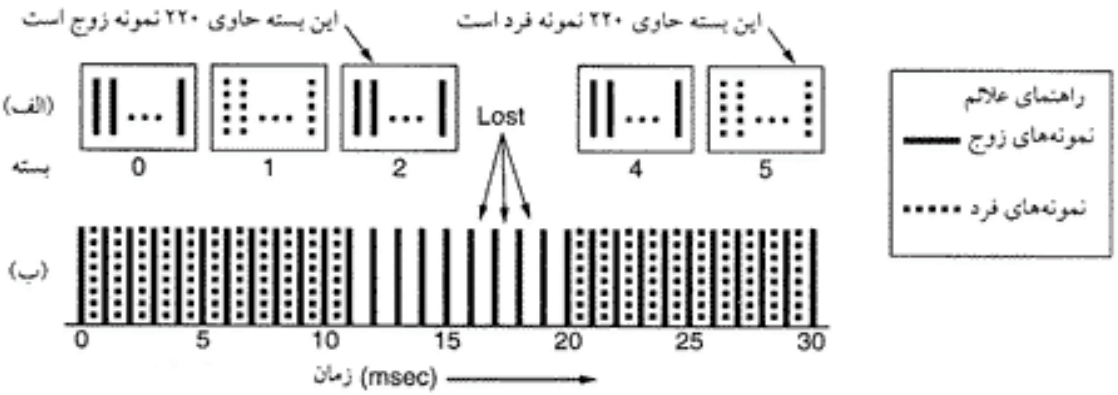
۱. مدیریت واسط کاربر.
۲. مقابله با خطاهای انتقال.
۳. باز کردن (از حالت فشرده خارج کردن) موسیقی.
۴. حذف لرزش‌ها.

اغلب برنامه‌های پخش امروزی دارای ظاهری بسیار جالب و شکیل هستند، و از دستگاههای پخش استریوی مدل بالا (با تمام دکمه، عقربه‌ها، و چراغهای رنگارنگ) تقلید می‌کنند. بعضی از آنها حتی اجازه می‌دهند تا شکل ظاهری برنامه را (که به پوست – skin – معروف است) به سلیقهٔ خود تغییر دهید. این یکی از وظایف برنامهٔ پخش در رابطه با واسط کاربر است.

وظیفهٔ دوم برنامهٔ پخش مقابله با خطاهای انتقال است. برنامه‌های پخش بی‌درنگ بندرت از TCP برای انتقال موسیقی استفاده می‌کنند، چون مدیریت خطا در TCP می‌تواند باعث ایجاد وقفه‌های آزاردهنده در موسیقی شود. پروتکل مرسوم برای انتقال موسیقی پروتکل RTP است، که آنرا در فصل ۶ دیدید. مانند اغلب پروتکلهای بی‌درنگ، RTP هم لایه‌ایست بر فراز UDP ، بنابراین گم شدن بسته‌ها کاملاً محتمل است.

در برخی از موارد، برای بهبود مدیریت خطا از روشهای **یک‌درمیانی** (interleaving) استفاده می‌شود. برای مثال، هر بسته می‌تواند حاوی ۲۲۰ نمونهٔ استریو (یک زوج ۱۶ بیتی) معادل 5 msec موسیقی باشد، ولی پروتکل انتقال بجای آن ابتدا 10 msec از نمونه‌های فرد را در یک بسته، و بدنبال آن 10 msec از نمونه‌های زوج را در بستهٔ بعدی می‌فرستد. در این روش اگر یک بسته گم شود، شکاف 5 msec در موسیقی بوجود نمی‌آید، و برنامهٔ پخش می‌تواند با استفاده از تکنیکهای درون‌یابی زوجهای گم شده را تخمین بزند.

در شکل ۷-۶۰ تکنیک یک‌درمیانی را ملاحظه می‌کنید. در اینجا هر بسته شامل نمونه‌های فرد یا زوج فاصلهٔ زمانی 10 msec است. در نتیجه، از دست رفتن بستهٔ ۳ باعث ایجاد شکاف در موسیقی نمی‌شود، بلکه فقط کیفیت پخش بصورت موقتی و زودگذر افت خواهد کرد. برنامهٔ پخش برای حفظ پیوستگی موسیقی، با استفاده از تکنیکهای ریاضی **درون‌یابی** (interpolating) نمونه‌های گم شده را (از روی نمونه‌های قبلی و بعدی) تخمین

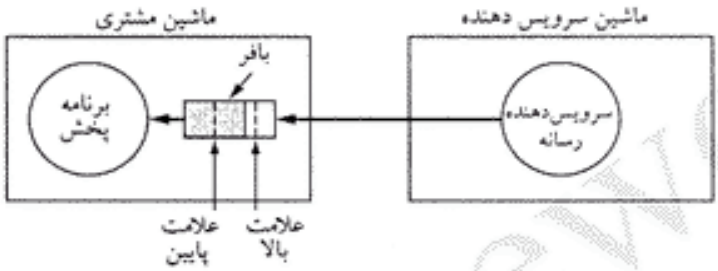


شکل ۷-۶۰. وقتی نمونه‌ها بصورت یک‌درمیان فرستاده شوند، گم شدن یک بسته بجای ایجاد شکاف در موسیقی فقط باعث کاهش موقتی کیفیت آن خواهد شد.

می‌زند. البته این روش خاص فقط با نمونه‌های فشرده‌نشده کار می‌کند، ولی بخوبی نشان می‌دهد که روشهای هوشمندانه چگونه می‌توانند به بهبود کیفیت پخش کمک کنند. (در RFC 3119 برای صدای فشرده‌شده نیز روشی ارائه شده است.)

وظیفهٔ سوم برنامهٔ پخش باز کردن (decompress) موسیقی است. با آن که این تکنیک نسبتاً ساده است، اما به محاسبات زیادی نیاز دارد.

وظیفهٔ چهارم برنامهٔ پخش، حذف لرزش (jitter) ، جزء آزاردهنده‌ترین وظایف همهٔ سیستمهای بی‌درنگ است. تمام سیستمهای صدای جویباری قبل از شروع پخش، ۱۰ الی ۱۵ ثانیه موسیقی را بافر می‌کنند (شکل ۷-۶۱ را ببینید). بطور ایده‌آل، سرویس‌دهنده این بافر را با همان سرعتی که برنامهٔ پخش از آن می‌خواند، پُر می‌کند. اما در واقعیت ممکنست چنین چیزی عملی نباشد، پس به نوعی حلقهٔ فیدبک نیاز داریم.



شکل ۷-۶۱. برنامهٔ پخش، بجای پخش مستقیم موسیقی از شبکه، ورودی سرویس‌دهندهٔ رسانه را بافر کرده، و موسیقی را از این بافر اجرا می‌کند.

برای پُر نگه داشتن بافر از دو روش می‌توان استفاده کرد. در روش **سرویس‌دهندهٔ پول** (pull server) ، تا زمانی که بافر جا دارد، برنامهٔ پخش به درخواست از سرویس‌دهندهٔ رسانه برای ارسال بسته‌های بعدی ادامه می‌دهد. هدف برنامهٔ پخش آن است که بافر را تا حد امکان پُر نگه دارد. عیب بزرگ این روش ارسال درخواستهای غیرضروری از طرف برنامهٔ پخش است. سرویس‌دهنده می‌داند که باید تمام فایل را بفرستد، پس درخواست پخش‌کننده دیگر برای چیست؟ به همین دلیل از روش فوق بندرت استفاده می‌شود.

در روش دیگر، **سرویس‌دهندهٔ پوش** (push server) ، برنامهٔ پخش فقط یک بار درخواست *PLAY* را به سرویس‌دهندهٔ رسانه می‌فرستد، و این سرویس‌دهنده به وظیفهٔ خود (فرستادن پیوستهٔ داده‌ها) عمل می‌کند. در اینجا دو احتمال پیش می‌آید: سرویس‌دهندهٔ رسانه با سرعت معمولی پخش هماهنگ یا از آن سریعتر است. در هر دو حالت، قبل از شروع پخش مقداری داده بافر می‌شود. اگر سرعت سرویس‌دهنده همپای پخش باشد، داده از آن وارد بافر شده و از سمت دیگر توسط برنامهٔ پخش خوانده می‌شود. اگر همه چیز خوب پیش رود، مقدار دادهٔ درون بافر همیشه ثابت می‌ماند. این ساده‌ترین حالت است، چون لازم نیست هیچ پیام کنترلی (در هر دو جهت) فرستاده شود.

حالت دیگر اینست که سرویس‌دهندهٔ پوش داده را با سرعتی بیشتر از آنچه برنامهٔ پخش می‌خواند، بفرستد. در این وضعیت سرویس‌دهنده همیشه می‌تواند خود را به برنامهٔ پخش برساند، حتی اگر گاهی (بدلایلی) از آن عقب بماند. اما این حالت (عجلهٔ سرویس‌دهنده برای عقب نماندن)، می‌تواند منجر به رونویسی بافر (buffer overrun) – نوشتن روی داده‌هایی که هنوز خوانده نشده‌اند – شود.

برنامهٔ پخش برای جلوگیری از این وضعیت می‌تواند دو علامت **حدپائین** و **حدبالا** در بافر تعریف کند. سرویس‌دهنده آنقدر داده می‌فرستد که بافر تا علامت حد بالا پُر شود؛ در اینجا پخش‌کننده به سرویس‌دهنده پیام

می‌دهد که فرستادن را متوقف کند. از آنجائیکه سرویس‌دهنده قبل از دریافت پیام توقف همچنان به ارسال داده ادامه می‌دهد، فاصلهٔ علامت حد بالای بافر تا ظرفیت نهایی آن باید از حاصلضرب پهنای باند در تأخیر انتشار شبکه بیشتر باشد. پخش‌کننده حتی بعد از توقف ارسال سرویس‌دهنده به خالی کردن بافر ادامه می‌دهد، و وقتی مقدار بافر به علامت حد پائین رسید، به آن اطلاع می‌دهد که ارسال داده را از سر گیرد. محل علامت حد پائین بافر باید بگونه‌ای انتخاب شود که بافر تا رسیدن داده‌های جدید خالی نشود (buffer underrun) .

برای آن که برنامهٔ پخش بتواند سرویس‌دهندهٔ 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 ۷-۶۲ فرمانهای اصلی RTSP را ملاحظه می‌کنید.

فرمان	عملکرد سرویس‌دهنده
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 موقتا قطع می کند
TEARDOWN	کانال را رها می کند

شکل ۷-۶۲. فرمانهای RTSP از پخش‌کننده به سرویس‌دهنده.

۷-۴-۴ رادیوی اینترنتی

همین که امکان ارسال 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 می‌تواند برای خود یک ایستگاه رادیویی داشته باشد. ایستگاههای رادیویی اینترنتی موضوع بسیار نو و پویایی هستند، ولی ارزش آنرا دارد که کمی دربارهٔ آن حرف بزنیم.

دو رهیافت کلی برای رادیوی‌اینترنتی (Internet radio) وجود دارد. در رهیافت اول برنامه‌ها روی دیسک ضبط می‌شوند، و شنوندگان می‌توانند برنامه‌های 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 می‌توان برنامه‌های زندهٔ رادیویی را روی دیسک ضبط کرد، بگونه‌ای که آرشیو فقط نیم ساعت از برنامهٔ زنده عقب باشد. مزیت این روش سادگی آن است، و می‌توان از تکنیکهایی که قبلاً تشریح کردیم برای پیاده‌سازی آن استفاده کرد.

رهیافت دیگر پخش برنامهٔ زنده روی اینترنت است. برخی از ایستگاههای رادیویی همزمان با پخش معمولی، برنامه‌های خود را روی اینترنت نیز پخش می‌کنند، 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 داده‌ها برای حذف لرزش است. با بافر کردن ۱۰ الی ۱۵ ثانیه صدا می‌توان تا حد زیادی با قطع و وصلهای اجتناب‌ناپذیر شبکه مقابله کرد. مادامیکه بسته‌ها به موقع برسند، زمان رسیدن آنها چندان اهمیتی ندارد.

یکی از تفاوتهای مهم صدای جویباری و رادیوی اینترنتی این است که سرویس‌دهندهٔ رسانه می‌تواند داده‌ها را سریعتر از آنچه پخش‌کننده پخش می‌کند، بفرستد - چون پخش‌کننده می‌تواند با رسیدن بافر به علامت حد بالا

سرویس‌دهنده را وادار به توقف کند (یا از آن بخواهد در این مدت بسته‌های گم شده را دوباره ارسال کند). اما رادیوی اینترنتی نمی‌تواند داده‌ها را سریعتر از آنچه در حال تولید و پخش است، بفرستد.

تفاوت دیگر این است که یک رادیوی پخش زندهٔ اینترنتی معمولاً هزاران شنوندهٔ همزمان دارد، در حالیکه صدای جویباری اساساً سیستمی نقطه-به-نقطه است. در این حالت، رادیوی اینترنتی باید از پروتکلهای چندپخشی RTP/RTSP استفاده کند. این مؤثرترین روش پخش اینترنتی است.

اما در عمل رادیوهای اینترنتی به این روش کار نمی‌کنند. آنچه که معمولاً اتفاق می‌افتد این است که کاربر یک اتصال TCP به ایستگاه رادیویی برقرار کرده، و محتویات را روی این اتصال دریافت می‌کند. البته این روش مشکلاتی هم دارد، مثلاً وقتی پنجره پُر می‌شود یا بسته‌ها گم می‌شوند، پخش دچار وقفه خواهد شد.

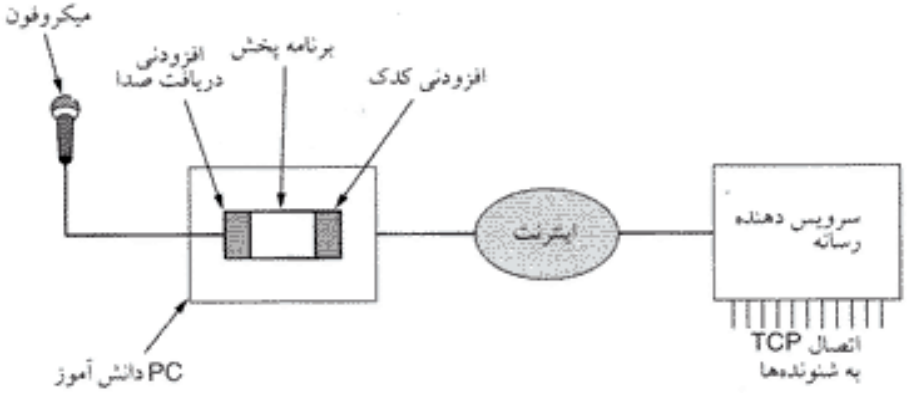
سه دلیل عمده برای استفاده از پروتکل تک‌پخشی TCP بجای پروتکل چندپخشی RTP وجود دارد. اول اینکه، اغلب ISP ها از چندپخشی پشتیبانی نمی‌کنند، و در واقع این گزینه عملی نیست. دوم اینکه، پروتکل RTP کمتر از TCP شناخته شده است، و کادر فنی رادیوهای اینترنتی (که معمولاً شرکتهای کوچکی بیش نیستند) مایلند با همان TCP معروف کار کنند. و سوم اینکه، معمولاً افراد بیشتر در محل کار به رادیوهای اینترنتی گوش می‌کنند، و این محل‌ها هم اغلب پشت دیوارآتش (firewall) قرار دارند. این دیوارهای آتش معمولاً فقط به پروتکلهای SMTP (پورت 25 - برای ایمیل)، UDP (پورت 53 - برای DNS)، و HTTP (پورت 80 - برای وب) اجازهٔ عبور می‌دهند، و جلوی هر چیز دیگر (از جمله RTP) را می‌گیرند. بنابراین، برای رادیوهای اینترنتی بهترین روش آنست که خود را یک سرویس‌دهندهٔ HTTP (که با اتصال TCP کار می‌کند) جا بزنند. البته با این کار برنامه‌های چندرسانه‌ای کارایی خود را بشدت از دست می‌دهند.

از آنجائیکه رادیوی اینترنتی موجودی نو پاست، جنگ فرمتها در آن با شدت تمام ادامه دارد. فرمتهای 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 منبع‌باز (open source) است و آنقدر با MP3 فرق دارد که نیازی به رعایت حق اختراع آن نداشته باشد.

رادیوهای اینترنتی معمولاً یک صفحهٔ وب دارند که فهرست برنامه‌های آن (و اطلاعات متفرقهٔ دیگر، باضافهٔ مقدار زیادی آگهی) را نمایش می‌دهد. اغلب این رادیوها در حاضر از فرمتهای مختلفی پشتیبانی می‌کنند، که کاربر می‌تواند بدلخواه 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 می‌کند. مرورگر با استفاده از اطلاعات این متافایل می‌تواند برنامهٔ کمکی مناسب را انتخاب کند. سپس این فایل را روی دیسک نوشته، و نام فایل را به برنامهٔ کمکی می‌دهد. برنامهٔ 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 می‌کند؛ سپس به سرویس‌دهنده وصل شده، و مانند یک رادیو شروع به کار می‌کند. پروتکل *http* برای صدا (که فقط یک استریم است) کاملاً کفایت می‌کند، 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 جنبهٔ رادیوی اینترنتی سادگی آن است، بطوری که حتی یک دانش‌آموز دبیرستانی هم می‌تواند رادیوی اینترنتی راه بیندازد. در شکل ۷-۶۳ ساده‌ترین پیکربندی یک رادیوی اینترنتی را می‌بینید. همانطور که می‌بینید، یک رادیوی اینترنتی در واقع چیزی نیست جز یک PC معمولی با کارت صوتی و میکروفون. نرم‌افزار هم فقط یک برنامهٔ پخش رسانه (مانند Winamp) است، باضافهٔ افزودنی‌های لازم برای گرفتن صدای میکروفون و تبدیل آن به فرمت مناسب (مانند MP3 یا Vorbis).



شکل ۷-۶۳. یک رادیوی اینترنتی دانش‌آموزی.

استریم صدای خروجی، سپس، به یک سرویس‌دهندهٔ HTTP روی اینترنت فرستاده می‌شود تا در اختیار شنوندگان قرار گیرد. این قبیل سرویس‌دهنده‌ها معمولاً تعداد زیادی ایستگاه رادیویی را میزبانی می‌کنند، و حتی صفحه‌ای دارند که نشان می‌دهد در لحظهٔ حاضر کدام ایستگاهها در حال پخش برنامه‌اند. شنوندهٔ علاقمند به این سرویس‌دهنده وصل شده، ایستگاه موردنظر را انتخاب کرده و محتویات ایستگاه را از طریق اتصال TCP دریافت می‌کند. بسته‌های نرم‌افزاری مختلفی برای مدیریت یک ایستگاه رادیوی اینترنتی (از ابتدا تا انتها) وجود دارد، که از میان آنها می‌توان به icecast (که نرم‌افزاری با استاندارد منبع باز است) اشاره کرد. سرویس‌دهنده‌های زیادی هم هستند، که در ازای دریافت مبالغ ناچیزی رادیوی شما را میزبانی می‌کنند.

۷-۴-۵ صداروی IP

آن قدیمها شبکهٔ تلفن عمومی بیشتر برای صدا بکار می‌رفت، و کمی هم داده روی آن منتقل می‌شد. اما ترافیک داده روز به روز افزایش یافت، تا اینکه در سال ۱۹۹۹ ترافیک داده و صدا مساوی شد (از آنجائیکه صدا روی ترانک‌های شبکه بصورت دیجیتال منتقل می‌شود، می‌توان 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).

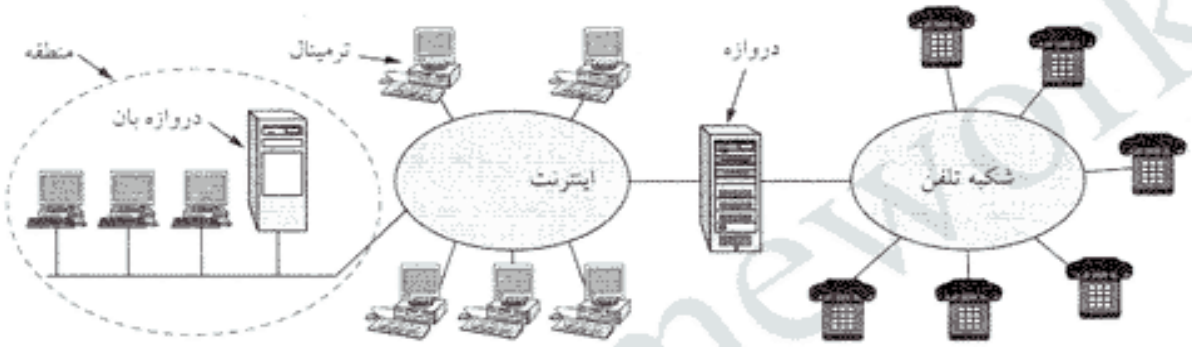
یکی از نتایج این وضعیت آن بود که بسیاری از اپراتورهای شبکه‌های سوئیچینگ بسته‌ای علاقمند شدند تا صدا را هم روی شبکه‌های داده منتقل کنند. فشاری که این کار به شبکه‌های داده می‌آورد چندان مهم نیست، چون این شبکه‌ها برای انتقال حجم عظیمی از اطلاعات طراحی شده‌اند. صورتحساب تلفن اغلب افراد بسیار بیشتر از صورتحساب اینترنت آنهاست، بهمین دلیل اپراتورهای شبکهٔ داده تلفن‌اینترنتی (Internet telephony) را یکی از راههای کسب سود سرشار ارزیابی کردند (بدون اینکه نیاز به سرمایه‌گذاری زیادی داشته باشد). بدین ترتیب بود که تلفن اینترنتی (یا همان صداروی IP – voice over IP) متولد شد.

H.323

یک چیز از همان اول برای 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 استانداردی برای تلفن اینترنتی وضع کنند. در سال ۱۹۹۶، ITU توصیه‌نامهٔ H.323 را با عنوان «سیستمها و تجهیزات تلفن بصری برای شبکه‌های محلی بدون تضمین کیفیت سرویس» ارائه کرد (این نامگذاری‌ها فقط از شرکتهای تلفن برمی‌آید!).

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 زمینه‌های دیگر ارجاع کرده است. مدل کلی استاندارد H.323 را در شکل ۷-۶۴ ملاحظه می‌کنید. در قلب این مدل یک دروازه (gateway) قرار دارد، که اینترنت را به شبکهٔ تلفن وصل می‌کند. این مدل در سمت اینترنت به زبان پروتکلهای H.323 صحبت می‌کند، و در سمت شبکهٔ تلفن به زبان پروتکلهای PSTN . در این مدل به دستگاههای مخابراتی ترمینال (terminal) گفته می‌شود. هر شبکهٔ محلی می‌تواند دارای یک دروازه‌بان (gatekeeper) باشد، که ارتباطات آن منطقه (zone) را کنترل می‌کند.

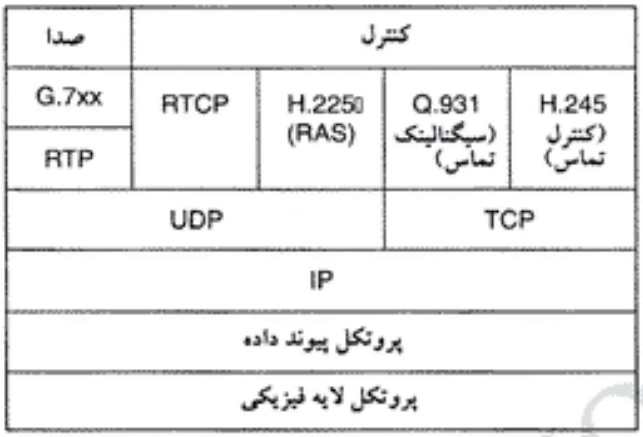


شکل ۷-۶۴. مدل H.323 برای تلفن اینترنتی.

یک شبکهٔ تلفن به تعدادی پروتکل نیاز دارد، که اولین آنها پروتکلی برای کُد کردن صدا است. سیستم PCM ، که در فصل ۲ بررسی کردیم، در توصیه‌نامهٔ ITU G.711 تعریف شده است. این سیستم هر کانال صوتی را با نرخ نمونه‌برداری 8000 samples/sec (با نمونه‌های ۸ بیتی) کُد کرده و یک خروجی فشرده‌نشدهٔ 64 kbps بدست می‌دهد. تمام سیستمهای H.323 باید از G.711 پشتیبانی کنند، ولی پروتکلهای فشرده‌سازی صدای دیگر را هم می‌توان بکار برد (اگر چه الزامی نیست). این پروتکلها از الگوریتمهای مختلف فشرده‌سازی استفاده می‌کنند، و نسبتهای متغیری از «کیفیت/پهنای باند» بدست می‌دهند. برای مثال، G.723.1 یک بلوک ۲۴۰ نمونه‌ای (۲۴۰ بایتی – که معادل 30 msec صداست) را گرفته و با استفاده از **کُدکردن پیشگویانه** (predictive coding) به ۲۴ یا ۲۰ بایت تقلیل می‌دهد. نرخ خروجی این الگوریتم 6.4 kbps یا 5.3 kbps (ضریب کاهش ۱/۱۰ یا ۱/۱۲) می‌باشد، ضمن اینکه کیفیت صدا را هم تا حد بسیار خوبی حفظ می‌کند. کُدک‌های دیگری نیز برای کُد کردن صدا وجود دارند.

از آنجائیکه الگوریتمهای فشرده‌سازی مختلفی وجود دارند، ترمینالها باید بتوانند بطریقی پروتکل فشرده‌سازی بکار رفته را بین خود مشخص کنند. این کار بر عهدهٔ پروتکل H.245 گذاشته شده است. این پروتکل همچنین نرخ بیت اتصال را تعیین می‌کند. برای کنترل کانالهای RTP هم به پروتکل RTCP نیاز داریم. همچنین پروتکلی می‌خواهیم که کارهایی از قبیل برقراری یا قطع اتصال، ارسال بوق آزاد تلفن، تأمین زنگهای تلفن (در حالتهای مختلف)، و مانند اینها را انجام دهد. برای این کار از پروتکل ITU Q.931 استفاده می‌کنیم. ترمینالها برای ارتباط با دروازه‌بان (اگر وجود داشته باشد) هم به یک پروتکل نیاز دارند، که H.225 را برای این منظور بکار می‌بریم. این پروتکل کانال بین PC و دروازه‌بان را، که کانال RAS (Registration/Admission/Status) نام دارد، کنترل می‌کند. این کانال اجازه می‌دهد 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 عهدهٔ RTCP است) استفاده می‌کنیم. موقعیت پروتکلهای مختلف H.323 را در شکل ۷-۶۵ ملاحظه می‌کنید.



شکل ۷-۶۵. پشته پروتکل H.323 .

برای اینکه ببینید همهٔ این پروتکلها چگونه با هم کار می‌کنند، ترمینالی (که یک PC در شبکه‌ای با 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 بستهٔ 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 بشناساند، که برای این منظور یک پیام RAS (در یک بستهٔ UDP) به آن می‌فرستد (مرحلهٔ ثبت نام - Registration). بعد از آنکه دروازه‌بان شبکه PC را قبول کرد، PC یک پیام پذیرش RAS فرستاده (مرحلهٔ 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.

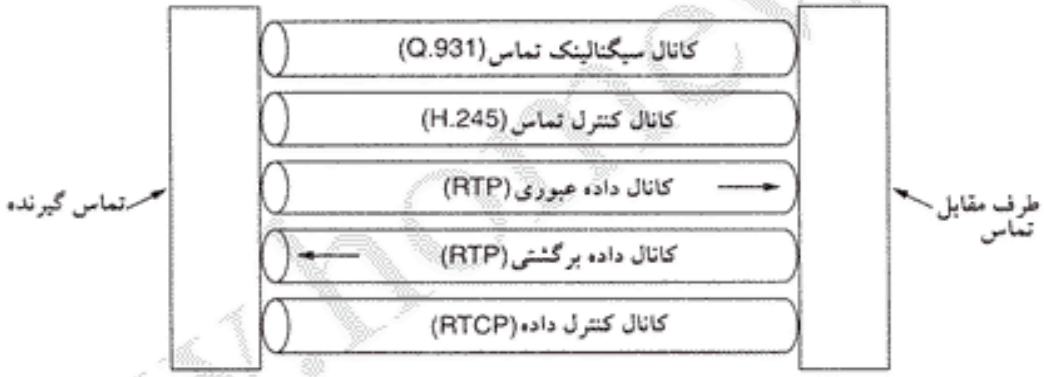
پس از آن، PC برای شروع تماس تلفنی یک اتصال TCP به دروازه‌بان برقرار می‌کند. از آنجائیکه برای برقراری تماس به پروتکلهای شبکهٔ تلفن نیاز است و این پروتکلها هم اتصال-گرا هستند، فقط از TCP می‌توان برای این منظور استفاده کرد. از طرف دیگر، چون در شبکهٔ تلفن چیزی شبیه RAS وجود ندارد، در انتخاب نوع پروتکل آن قسمت (UDP یا TCP) آزادی عمل داریم؛ و بعلت کمتر بودن سربارهٔ UDP ، طراحان H.323 آنرا انتخاب کرده‌اند.

پس از گرفتن پهنای باند لازم، PC می‌تواند یک پیام *SETUP* Q.931 روی اتصال TCP بفرستد، و شماره تلفن مقصد (یا آدرس IP و شمارهٔ پورت، اگر مقصد یک کامپیوتر باشد) را به دروازه‌بان بدهد. دروازه‌بان هم برای تصدیق دریافت درخواست کاربر، یک پیام *CALL PROCEEDING* Q.931 به آن برمی‌گرداند. سپس، دروازه‌بان پیام *SETUP* را به دروازه هدایت می‌کند.

دروازه، که نیمی کامپیوتر و نیمی سوئیچ تلفن است، شمارهٔ تلفن خواسته شده را می‌گیرد. ایستگاه پایانی که تلفن در آن قرار دارد، به شمارهٔ مزبور زنگ می‌زند و یک پیام *ALERT* Q.931 هم به PC مبدأ می‌فرستد تا نشان دهد که زنگ خوردن شروع شده است. وقتی طرف مقابل گوشی را برمی‌دارد، ایستگاه پایانی یک پیام *CONNECT* Q.931 به مبدأ برمی‌گرداند تا بگوید که ارتباط برقرار شده است.

بعد از برقراری تماس دروازه‌بان دیگر کاری ندارد و از مدار خارج می‌شود، ولی دروازه همچنان به کار خود ادامه خواهد داد. بسته‌های بعدی دروازه‌بان را دور زده، و مستقیماً به آدرس IP دروازه فرستاده می‌شوند. در این نقطه دیگر فقط یک کانال ارتباطی معمولی بین مبدأ و مقصد وجود دارد، که چیزی نیست جز اتصال لایه‌های فیزیکی برای انتقال بیت‌ها، بدون هیچ اطلاعی از ماهیت واقعی کار.

در این مرحله برای تعیین پارامترهای تماس تلفنی از پروتکل H.245 استفاده می‌شود. این پروتکل از کانالهای H.245 استفاده می‌کند، که همیشه باز هستند. هر دو طرف ابتدا روی این کانال مقدورات خود را اعلام می‌کند، برای مثال نوع تماس (مثلاً، ویدئو - چون H.323 قادر به انتقال ویدئو نیز هست - یا تماس کنفرانسی)، نوع کُدک، و غیره. همین که دو طرف از توانائیهای یکدیگر با خبر شدند، دو کانال یک طرفهٔ داده (با تعیین نوع کُدک و سایر پارامترها) بین آنها ایجاد می‌شود. از آنجائیکه امکان دارد قابلیتهای دو طرف یکسان نباشد، کانالهای رفت و برگشت می‌توانند کاملاً متفاوت باشند. بعد از پایان مرحلهٔ مذاکره بر سر پارامترها، انتقال داده با استفاده از RTP می‌تواند شروع شود، که برای مدیریت آن از RTCP کمک می‌گیریم (این پروتکل نقش مهمی در کنترل ازدحام دارد). اگر ویدئو نیز وجود داشته باشد، سنکرون کردن صدا و تصویر هم بر عهدهٔ RTCP خواهد بود. انواع کانالهای ممکن را در شکل ۷-۶۶ ملاحظه می‌کنید. در پایان تماس نیز، برای قطع ارتباط از سیگنالهای Q.931 استفاده می‌شود.



شکل ۷-۶۶. کانال‌های منطقی بین دو طرف تماس.

بعد از قطع ارتباط، PC تماس‌گیرنده یک پیام RAS دیگر به دروازه‌بان می‌فرستد تا پهنای باند اختصاص داده شده را به آن پس بدهد (یا تماس دیگری بگیرد).

دربارهٔ **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 شبکهٔ موجود بتواند (با استفاده از تکنیکهای فصل ۵) ارتباطی پایدار و بدون لرزش بین PC تماس‌گیرنده و دروازه برقرار کند، QoS تماس خوب خواهد بود؛ در غیر اینصورت، خیر. شبکهٔ تلفن از PCM استفاده می‌کند، و هیچوقت لرزش ندارد.

SIP - پروتکل آغاز نشست

H.323 توسط ITU طراحی شد، و بسیاری از اینترنتی‌ها آنرا یک محصول تلفنی دیگر (بزرگ، پیچیده، و انعطاف‌ناپذیر) یافتند. بهمین دلیل، IETF کمیته‌ای تشکیل داد تا روشی ساده‌تر 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 نحوهٔ برقراری تماسهای تلفن اینترنتی، کنفرانس ویدئویی، و ارتباطات چندرسانه‌ای دیگر تشریح شده است. برخلاف H.323 که مجموعه‌ایست 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 چندپخشی (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 لایهٔ 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 ناحیهٔ *cs.university.edu* است. در URL های 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 مشخصهٔ 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 .

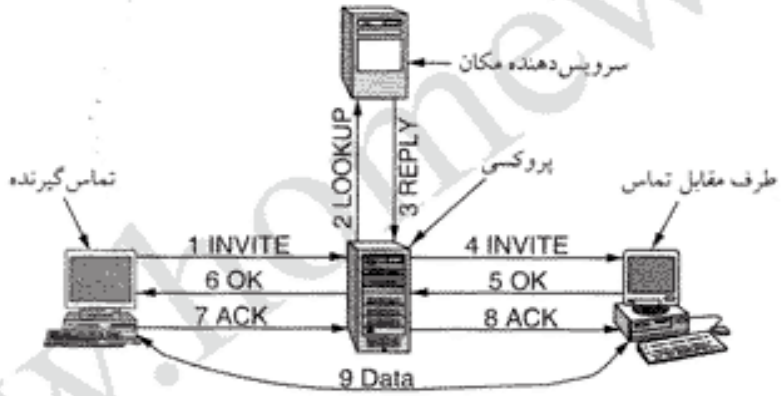
برای برقراری یک نشست دو روش وجود دارد: برقرای اتصال TCP به تماس‌شونده و ارسال یک پیام *INVITE* به آن؛ ارسال پیام *INVITE* در یک بستهٔ UDP . در هر دو حالت، سرآیند خط دوم و خطهای بعدی ساختار بدنهٔ پیام (شامل قابلیتهای تماس‌گیرنده، نوع رسانه، و فرمتهای آن) را مشخص می‌کنند. اگر تماس‌شونده دعوت به تماس را قبول کند، یک کُد پاسخ شبیه HTTP (کُدهای سه رقمی، که در اینجا هم 200 نشانهٔ قبول درخواست است) برمی‌گرداند. پس از این کُد 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 می‌توانند برای قطع ارتباط پیام *BYE* بفرستند. وقتی طرف مقابل این پیام را تصدیق کرد، نشست خاتمه خواهد یافت.

هر ماشین می‌تواند برای تعیین قابلیتهای خود (و اینکه اصلاً قادر به برقرای تماس VOIP هست یا نه) از متد *OPTIONS* استفاده کند، و معمولاً این کار را قبل از شروع نشست انجام می‌دهد.

پروتکل SIP از متد *REGISTER* برای یافتن افرادی که در محل مورد انتظار نیستند، استفاده می‌کند. هر ماشین پیام *REGISTER* خود را به یک سرویس‌دهندهٔ مکان SIP (SIP location server)، که محل افراد را تعقیب می‌کند، می‌فرستد. وقتی کسی می‌خواهد با شما تماس بگیرد، می‌تواند به کمک این سرویس‌دهنده محل فعلی شما را پیدا کند. روش کار را در شکل ۷-۶۸ می‌بینید. در اینجا، تماس‌گیرنده پیام *INVITE* خود را به یک پروکسی فرستاده است (هدف از بکارگیری پروکسی برداشتن وظیفهٔ یافتن مکان مخاطب از دوش کامپیوتر تماس‌گیرنده است). این پروکسی به کمک سرویس‌دهندهٔ مکان محل فرد مورد نظر را یافته، و پیام *INVITE* را به آن می‌فرستد (تمام پیامهای بعدی هم با واسطهٔ همین پروکسی رد و بدل خواهند شد). پیامهای *LOOKUP* و *REPLY* جزء SIP نیستند، و از هر پروتکلی (که با سرویس‌دهندهٔ مکان مورد استفاده سازگار باشد) می‌توان برای این منظور استفاده کرد.



شکل ۷-۶۸. استفاده از سرویس‌دهندهٔ پروکسی با SIP .

پروتکل SIP ویژگیهای دیگری (از جمله انتظار مکالمه، پیگیری تماس، رمزنگاری، و احراز هویت) نیز دارد که در اینجا به آنها نخواهیم پرداخت. اگر دروازهٔ مناسب بین اینترنت و شبکهٔ تلفن وجود داشته باشد، SIP می‌تواند بین کامپیوتر و تلفنهای معمولی نیز تماس برقرار کند.

مقایسهٔ H.323 و SIP

H.323 و SIP شباهتها و تفاوتهای بسیاری دارند. هر دوی آنها می‌توانند تماسهای مستقیم یا کنفرانسی بین کامپیوترها و تلفنهای معمولی برقرار کنند. هر دو از مذاکره برای تعیین پارامترهای تماس، رمزنگاری، و پروتکلهای RTP/RTCP پشتیبانی می‌کنند. فهرستی از شباهتها و تفاوتهای این دو را در شکل ۷-۶۹ ملاحظه می‌کنید.

با وجود شباهتهای ظاهری، این دو پروتکل از نظر فلسفهٔ 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 رهیافت همه چیز کاملاً مشخص و تعریف شده، و ارتباط بین سیستمهای مختلف بسادگی امکانپذیر است. بهایی که برای این سادگی باید پرداخت، عبارتست از بزرگی، پیچیدگی و انعطاف‌ناپذیری، که انطباق این پروتکل با نیازهای آینده را دشوار کرده است.

آیتم	H.323	SIP
مسئول طراحی	ITU	IETF
سازگاری با PSTN	بلی	تا حد زیاد
سازگاری با اینترنت	خیر	بلی
معماری	یکپارچه	ماژولار
کامل بودن	پشته پروتکل کامل	فقط برقراری تماس
مذاکره پارامترها	بلی	بلی
سیگنالینگ تماس	Q.931 رویTCP	SIP روی TCP یا UDP
فرمت پیام	باینری	متنی
انتقال رسانه	RTP/RTCP	RTP/RTCP
تماس چند طرفه	بلی	بلی
کنفرانس چند رسانه ای	بلی	بلی
آدرس دهی	شماره تلفن یا میزبان	URL
پایان تماس	صریح یا رها کردنTCP	صریح یا بعد از انقضای زمان
پیام رسانی فوری	خیر	بلی
رمزنگاری	بلی	بلی
اندازه استاندارد	۱۴۰۰ صفحه	۲۵۰ صفحه
پیاده سازی	بزرگ و پیچیده	متوسط
وضعیت فعلی	کاربرد گسترده	درحال رشد

شکل ۷-۶۹ مقایسه‌ای بین H.323 و SIP .

از طرف دیگر، SIP یک پروتکل سبک و معمولی اینترنتی است که با مبادلهٔ پیامهای متنی کار می‌کند، و سازگاری خوبی با پروتکلهای موجود اینترنت دارد، ولی در زمینهٔ ارتباط با پروتکلهای سیگنالینگ تلفن چندان قوی نیست. از آنجائیکه IETF مدل VOIP خود را بصورت کاملاً مدولار تعریف کرده، این پروتکل انعطاف‌پذیری بالایی دارد و براحتی می‌توان 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.

صدا روی IP (VOIP) مبحثی نو و در حال تحول است. کتابهای متعددی در این زمینه نوشته شده، که از میان آنها می‌توان به (Colins, 2001; Davidson and Peters, 2000; Kumar et al., 2001; and Wright, 2001) اشاره کرد. در شمارهٔ May/June 2002 مجلهٔ *Internet Computing* نیز چندین مقاله در زمینهٔ VOIP ارائه شده است.

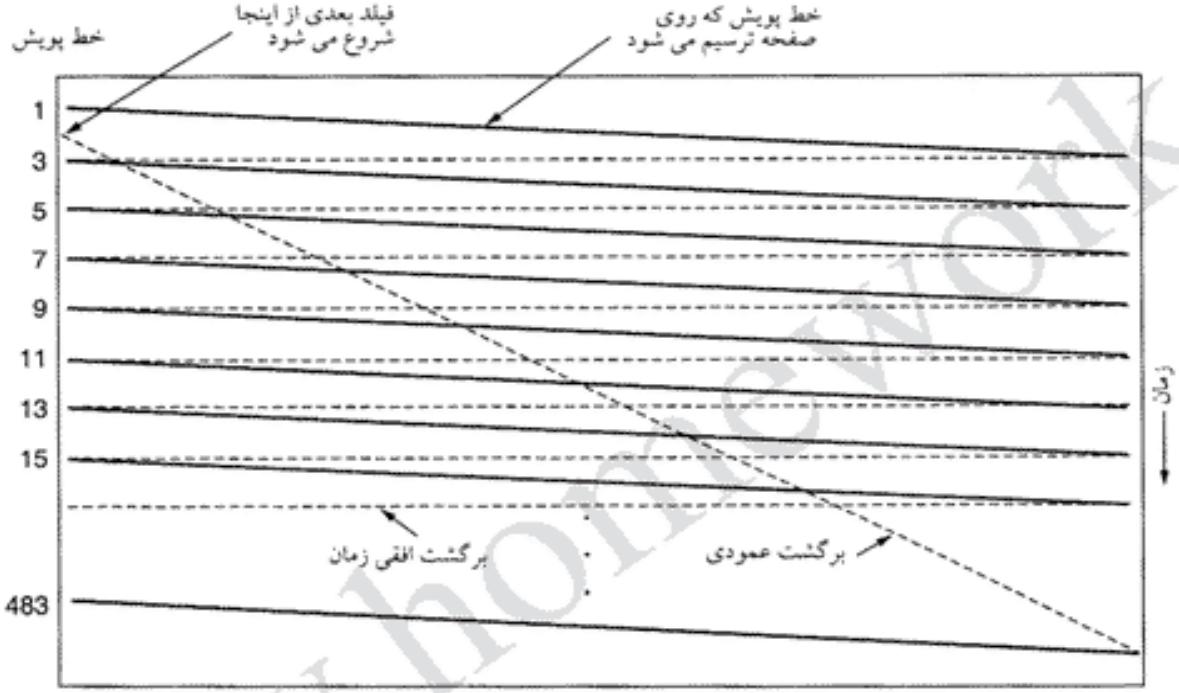
۶-۴-۷ مقدمه‌ای بر ویدئو

تا اینجا دربارهٔ گوش صحبت کردیم؛ اکنون وقت آن است که به چشم بپردازیم (نگران نباشید بعد از آن دیگر سراغ بینی نخواهیم رفت!). یکی از ویژگیهای چشم انسان این است که وقتی تصویری روی شبکیه می‌افتد، این تصویر برای چند هزارم ثانیه باقی می‌ماند. 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 می‌کنند.

سیستمهای آنالوگ

برای درک بهتر ویدئو، بهتر است از ساده‌ترین وسیله یعنی یک تلویزیون سیاه-سفید قدیمی کمک بگیریم.

برای تبدیل یک تصویر دو-بُعدی به سیگنالی یک-بُعدی (تابعی از ولتاژ بر حسب زمان)، دوربین تلویزیونی با شروع از بالای تصویر آن را به وسیلهٔ یک پرتو الکترونی از چپ براست اسکن کرده و بتدریج پائین می‌آید، و در هر لحظه شدت نور را ثبت می‌کند. وقتی پرتو الکترونی به انتهای اسکن (که فریم نامیده می‌شود) رسید، دوباره به نقطهٔ شروع برمی‌گردد. خروجی دوربین همین سیگنال (تابع شدت نور بر حسب زمان) است، و گیرنده با تکرار فرآیند اسکن دوباره تصویر را می‌سازد. 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 می‌کند. در سیستمهای ویدئویی آمریکای شمالی، جنوبی و ژاپن از اسکن ۵۲۵ خطی، نسبت افقی-به-عمودی 4:3 ، و سرعت ۳۰ فریم بر ثانیه استفاده می‌شود. سیستمهای اروپایی اسکن ۶۲۵ خطی، نسبت افقی-به-عمودی 4:3 ، و سرعت ۲۵ فریم بر ثانیه را بکار می‌برند. در هر دو سیستم، برای چهارگوش کردن تصویر در لامپهای گرد قدیمی، چند خط از بالا و چند خط از پائین اصلاً نمایش داده نمی‌شود؛ در NTSC (که ۵۲۵ خط دارد) فقط ۴۸۳ خط نشان داده می‌شود، و در سیستمهای اروپایی (PAL/SECAM) ۵۷۶ خط (از ۶۲۵ خط). در مدت برگشت پرتو الکترونی به نقطهٔ شروع این پرتو خاموش است (تا روی 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 می‌کنند؛ این سرویس به تله‌تکست (TeleText) معروف است.

با اینکه ۲۵ فریم بر ثانیه برای نمایش تصاویر متحرک کافیست، اما برخی افراد (بویژه سالخورده‌گان) در این سرعت احساس می‌کنند تصاویر چشمک می‌زند (چون مدت دوام تصویر روی شبکیهٔ 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 می‌شود. در این تکنیک (بجای نمایش خطوط اسکن متوالی)، ابتدا خطوط فرد و سپس خطوط

زوج نمایش داده می‌شوند. بدین ترتیب، در هر بار حرکت پرتو الکترونی از بالا تا پائین صفحه فقط نیمی از یک فریم کامل نشان داده می‌شود؛ این نیم‌فریم را یک فیلد (field) می‌گویند. 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 می‌رود. این تکنیک خط-در-میانی (interlacing) نام دارد. به تلویزیونها یا ویدئوهای غیر خط-در-میانی پیشرونده (progressive) گفته می‌شود. توجه کنید که، با اینکه فیلمهای سینمایی با سرعت ۲۴ فریم بر ثانیه پخش می‌شوند، ولی در آنجا هر تصویر به مدت ۱/۲۴ ثانیه بطور کامل دیده می‌شود.

ویدئوی رنگی از همان الگوی اسکن تک‌رنگ (سیاه-سفید) استفاده می‌کند، ولی در آن برای هر رنگ اصلی یک پرتو الکترونی مستقل وجود دارد که بطور هماهنگ عمل می‌کنند. رنگهای اصلی عبارتند از: قرمز، سبز، آبی (RGB) . همانطور که می‌دانید، هر رنگی را می‌توان با برهم‌نهی خطی رنگهای اصلی (با شدت‌های 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 گیرنده‌های سیاه-سفید هم قابل دریافت باشند. به همین دلیل کُد کردن جداگانهٔ سیگنالهای RGB (که ساده‌ترین روش ارسال سیگنالهای رنگی است) عملاً کنار گذاشته شد. (البته RGB کارآمدترین روش کُد کردن سیگنالهای رنگی نیست.)

اولین سیستم رنگی در آمریکا توسط «کمیتهٔ 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 سیستمهایی 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 بقیهٔ اروپا از آن استفاده می‌شود. 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 می‌گویند.

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 می‌شوند. جالبست بدانید که حساسیت چشم انسان نسبت به سیگنالهای روشنایی بسیار بیشتر از سیگنالهای رنگ است، بنابراین دقت چندانی در ارسال سیگنالهای رنگ لازم نیست. سیگنال روشنایی با همان فرکانس تلویزیونهای سیاه-سفید قدیمی پخش می‌شود، تا آنها هم بتوانند تصاویر را دریافت کنند؛ سیگنالهای رنگ نیز با فرکانسهای بالاتر (در باندی باریک) پخش می‌شوند. در برخی از تلویزیونها کنترلهایی بنام روشنایی، رنگ و غلظت رنگ وجود دارد، که در واقع این سیگنالها را کنترل می‌کنند. درک روشنایی و رنگ برای فهم تکنیکهای فشرده‌سازی ویدئو ضروری است.

در سالهای اخیر سر و صدای زیاد در اطراف HDTV (تلویزیون باوضوح بالا – High Definition TeleVision) بر پا شده است. این تلویزیونها با (تقریباً) دو برابر کردن تعداد خطوط اسکن، تصاویر بسیار بهتری تولید می‌کنند. آمریکا، اروپا و ژاپن همگی سیستمهای HDTV خاص خود را توسعه داده‌اند، 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 همهٔ آنها 16:9 می‌باشد. این فرمت برای نمایش فیلمهای سینمایی (که روی فیلمهای 35 mm با نسبت 3:2 ضبط می‌شوند) بسیار مناسبتر است.

سیستمهای دیجیتال

ساده‌ترین راه نمایش ویدئوی دیجیتال عبارتست از توالی فریمهایی که هر کدام از تعدادی پیکسل (pixel) با آرایش مستطیلی تشکیل شده‌اند. هر پیکسل را می‌توان با یک بیت (سیاه یا سفید) نشان داد. کیفیت این سیستم شبیه ارسال تصویر با فکس است (فوق‌العاده وحشتناک!).

اگر از ۸ بیت برای نمایش هر پیکسل استفاده کنیم، می‌توانیم ۲۵۶ سایهٔ 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 می‌شود، اگر چه تمام این سیستمها رنگها را هنگام ارسال به یک سیگنال مرکب تبدیل می‌کنند. استفاده از ۲۴ بیت برای هر پیکسل تعداد رنگهای ممکن را به حدود ۱۶ میلیون محدود می‌کند، ولی هیچ چشمی قادر به تشخیص این تعداد رنگ نیست (بیشتر که جای خود دارد). تصاویر رنگی دیجیتال با استفاده از سه پرتو اسکن‌کننده (یک پرتو برای هر رنگ) تولید می‌شوند. روش کار شبیه شکل ۷-۷۰ است، با این تفاوت که پیکسلهای منفرد جای خطوط پیوسته را گرفته‌اند.

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-میانی نیست (و معمولاً هم استفاده نمی‌شود). 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 می‌شود، هیچ مشکلی بنام نرمی حرکت وجود ندارد، ولی همین تصویر دارای لرزش و چشمک است، چون هر تصویر برای مدت زمان کافی روی شبکیه نقش نمی‌بندد. حال اگر فیلمی با سرعت ۲۰ فریم بر ثانیه نمایش پخش شود، ولی هر تصویر ۴ بار روی صفحهٔ مانیتور رسم شود، تصاویر پخش شده بدون چشمک خواهند بود ولی حرکت نرم نیست.

اهمیت این دو پارامتر وقتی بیشتر روشن می‌شود 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 شده‌اند، استفاده کنند. وضوح این مانیتورها اغلب 768 × 1024 ، 960 × 1280 یا 1200 × 1600 است. حتی کمترین وضوح (768 × 1024) 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 می‌توان 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 نمی‌توان بدون تبدیل به دیجیتال در حافظه ذخیره کرد (و همهٔ اینها یعنی هزینهٔ اضافی).

۷-٤-۷ فشرده‌سازی ویدئو

تا اینجا فهمیدیم که به ارسال تصاویر ویدئویی فشرده‌نشده حتی نباید فکر کرد؛ تنها امیدمان اینست که بتوانیم تصاویر ویدئویی را به میزان زیاد فشرده کنیم. خوشبختانه کار تحقیقاتی گسترده در چند دههٔ گذشته باعث شد تا ارسال تصاویر ویدئویی فشرده روی شبکه ممکن شود. در این قسمت خواهید دید که فشرده کردن تصاویر ویدئویی چگونه انجام می‌شود.

تمام سیستمهای مبتنی بر داده‌های فشرده دو بخش دارند: یک الگوریتم فشرده‌سازی در مبدأ، و یکی برای عکس آن در مقصد. در ادبیات فنی به این الگوریتمها بترتیب **کُدکردن** (encoding) و **دیکُدکردن** (decoding) می‌گویند؛ ما هم از همین اصطلاحات استفاده خواهیم کرد.

در الگوریتمهای فشرده‌سازی نوعی عدم تقارن ذاتی وجود دارد، که برای درک بهتر این فرآیند از اهمیت زیادی برخوردار است. اول اینکه، یک سند چندرسانه‌ای (مثلاً، یک فیلم سینمایی) فقط یک بار کُد شده و روی سرویس‌دهنده قرار داده می‌شود، ولی دیکُد شدن آن هزاران بار (هر بار که یکی از کاربران می‌خواهد آنرا تماشا کند) اتفاق می‌افتد. این بدان معناست که الگوریتم کُد کردن می‌تواند کُند و متکی به سخت‌افزارهای گران‌قیمت باشد، مشروط باینکه الگوریتم دیکُد کردن سریع بوده و نیازی به سخت‌افزارهای گران‌قیمت نداشته باشد. برای شرکتهای پخش چندرسانه‌ای دو هفته کرایه کردن یک آبرکامپیوتر (و کُد کردن تمام فیلمهایی که دارند) چندان دور از منطق نیست، ولی نمی‌توان از کاربران انتظار داشت برای تماشای یک فیلم ویدئویی دو ساعت آبرکامپیوتر اجاره کنند. در بسیاری از سیستمهای فشرده‌سازی الگوریتم کُد کردن، به قیمت سریع و ساده شدن عملیات دیکُد، پیچیده و وقت‌گیر طراحی شده‌اند.

از طرف دیگر، در سیستمهای **چندرسانه‌ای بی‌درنگ** (real-time multimedia) – مانند کنفرانس ویدئویی – کُند بودن فرآیند کُد کردن نیز غیرقابل‌قبول است. در اینجا کُد کردن باید درلحظه و بصورت بی‌درنگ انجام شود. در نتیجه، چنین سیستمهایی باید از الگوریتمهای متفاوتی استفاده کنند.

عدم تقارن دیگر در فرآیند کُد/دیکُد اینست که آنها نباید الزاماً عکس یکدیگر باشند. این بدان معناست که در سیستمهای چندرسانه‌ای الزامی نیست فایلی که در مقصد دیکُد می‌شود دقیقاً همان فایلی باشد که در مبدأ کُد شده، و می‌تواند چیزهایی را از دست بدهد. وقتی خروجی دیکُد شده دقیقاً همان ورودی کُد شده نباشد، اصطلاحاً گفته می‌شود که سیستم **تلفات‌دار** (lossy) است. در مقابل، سیستمی که خروجی دقیقاً معادل ورودی باشد، **بدون‌تلفات** (lossless) نامیده می‌شود. سیستمهای تلفات‌دار از اهمیت زیادی برخوردارند، چون از دست دادن مقدار کمی از اطلاعات به فشرده‌سازی بسیار بالایی منجر می‌شود.

استاندارد JPEG

ویدئو عبارتست از توالی چند تصویر (بعلاوهٔ صدا). اگر بتوانیم برای کُد کردن تصاویر ثابت الگوریتم مناسبی پیدا کنیم، می‌توانیم آنرا برای تصاویر متحرک (ویدئو) نیز بکار ببریم. امروزه الگوریتمهای خوبی برای فشرده‌کردن تصاویر ثابت وجود دارد، پس اجازه دهید از همانها شروع کنیم. استاندارد JPEG (Joint Photographic Experts Group) توسط گروهی از متخصصان عکاسی (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 فشرده‌سازی تصاویر متحرک یعنی MPEG ، در واقع چیزی نیست جز کُد کردن فریمهای متوالی با JPEG (باضافهٔ چند ویژگی دیگر برای فشرده‌کردن بین فریمها و تشخیص حرکت). JPEG در استاندارد ISO 10918 تعریف شده است.

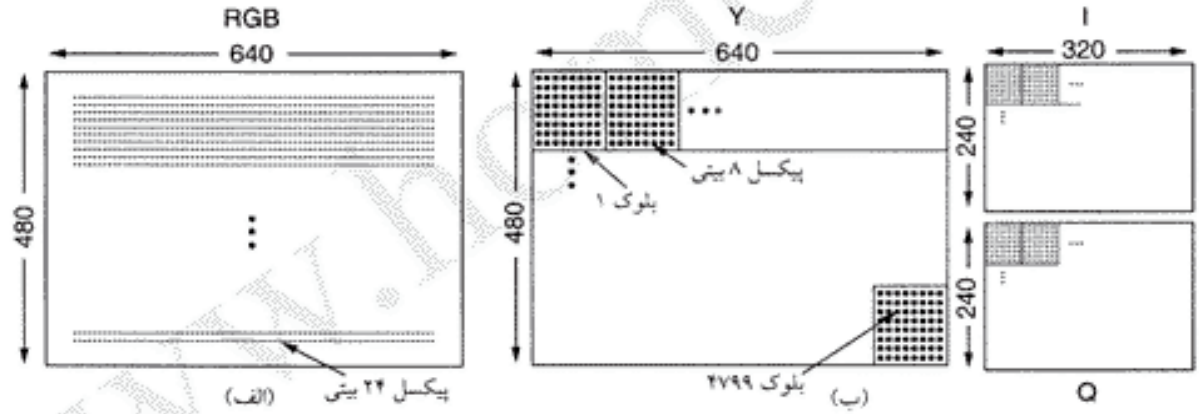
الگوریتم JPEG دارای چهار حالت و چندین گزینه است (و در واقع بیشتر به یک لیست خرید می‌ماند، تا

الگوریتم فشرده‌سازی). اما آنچه که ما به آن علاقه داریم (و در شکل ۷-۷۱می‌بینید)، حالت متوالی تلفات‌دار (lossy seqential) است. در اینجا برای سادگی بیشتر بحث، فقط به روش کُد کردن تصاویر ویدئویی 24-bit RGB در JPEG توجه می‌کنیم، و جزئیات دیگر را نادیده می‌گیریم.

شکل ۷-۷۱. عملکرد JPEG در حالت متوالی تلفات‌دار.

مرحلهٔ اول کُد کردن تصویر با JPEG ، **آماده‌سازی بلوک** (block preparation) است. برای سادگی بحث فرض می‌کنیم که ورودی JPEG یک تصویر RGB 480 × 640 با وضوح 24 bits/pixel است (شکل ۷-۷۲ الف). از آنجائیکه فشرده‌سازی 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 می‌کنیم:

$$Y = 0.30R + 0.59G + 0.11B$$
$$I = 0.60R - 0.28G - 0.32B$$
$$Q = 0.21R - 0.52G + 0.31B$$



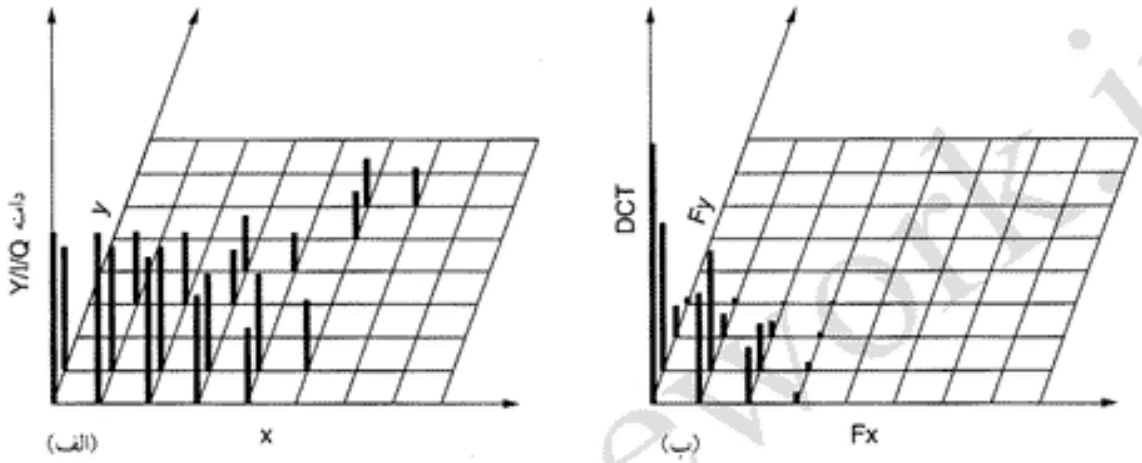
شکل ۷-۷۲. (الف) ورودی RGB . (ب) بعد از آماده‌سازی بلوک.

در PAL سیگنالهای رنگ V و U نامیده می‌شوند و ضرایب متفاوتی دارند، ولی ایدهٔ اصلی همان است؛ SECAM هم که با هر دوی آنها فرق دارد.

برای Y ، I و Q ماتریسهای جداگانه‌ای (با مقادیر 0 تا 255) ایجاد می‌شود. سپس، تصویر به بلوکهای مربعی ۴ پیکسلی تقسیم شده، و I و Q متوسط آنها محاسبه می‌شود (با این کار تصویر ورودی به 240 × 320 تبدیل می‌شود). این کاهش با تلفات همراه است، اما از آنجائیکه چشم انسان به روشنایی حساس‌تر است تا رنگ، متوجه آن نخواهد شد. ضریب کاهش تا اینجا ۱ به ۲ است (یعنی اندازهٔ فایل نصف شده است). حال، از تمام عناصر هر سه ماتریس عدد ۱۲۸ کم می‌شود، که با اینکار 0 به عدد میانه در محدودهٔ اعداد ماتریس‌ها تبدیل خواهد شد. و در آخر، تمام ماتریس‌ها به بلوکهای 8 × 8 تقسیم می‌شوند. با این کار ماتریس Y دارای 4800 بلوک، و دو ماتریس دیگر هر یک دارای 1200 بلوک خواهند بود (شکل ۷-۷۲ب را ببینید).

در مرحلهٔ دوم JPEG ، روی تمام 7200 ماتریس بصورت جداگانه **تبدیل کسینوسی گسسته** (Discrete – DCT

Cosine Transformation) انجام می‌شود. 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 می‌دهند که در هر فرکانس فضایی چه مقدار انرژی طیفی وجود دارد. از نظر تئوری، DCT یک الگوریتم بدون‌تلفات 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 می‌روند (به شکل ۷-۷۳ نگاه کنید).



شکل ۷-۷۳. (الف) یکی از بلوکهای ماتریس Y . (ب) ضرایب DCT .

بعد از پایان DCT ، JPEG وارد مرحلهٔ سوم یعنی **کوانتیزه‌کردن** (quantization) می‌شود، که در آن ضرایب کم‌اهمیت‌تر DCT دور انداخته می‌شوند. در این تبدیل تلفات‌دار، عناصر ماتریس 8 × 8 ضرایب DCT بر وزنی که از یک جدول (جدول کوانتیزه کردن) گرفته می‌شود، تقسیم می‌شوند. اگر تمام وزن‌ها 1 باشند، تبدیل هیچ کاری انجام نمی‌دهد؛ ولی اگر وزن‌ها خیلی از مبدأ دور باشند، فرکانسهای فضایی بالاتر دور انداخته می‌شوند.

به مثال شکل ۷-۷۴ نگاه کنید. در این شکل سه جدول می‌بینید: ماتریس DCT اولیه، جدول کوانتیزه کردن، وجدولی که از تقسیم عناصر ماتریس DCT بر عناصر متناظر در جدول کوانتیزه کردن بدست آمده است. مقادیر جدول کوانتیزه کردن جزیی از استاندارد JPEG نیست، و هر برنامهٔ تبدیل به فرمت JPEG باید خود آنرا داشته باشد (و بوسیلهٔ آن مقدار تلفات اطلاعات را کنترل کند).

ضرایب DCT

150	80	40	14	4	2	1	0
92	75	36	10	6	1	0	0
52	38	26	8	7	4	0	0
12	8	6	4	2	1	0	0
4	3	2	0	0	0	0	0
2	2	1	1	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0

جدول کوانتیزه‌کردن

1	1	2	4	8	16	32	64
1	1	2	4	8	16	32	64
2	2	2	4	8	16	32	64
4	4	4	4	8	16	32	64
8	8	8	8	8	16	32	64
16	16	16	16	16	16	32	64
32	32	32	32	32	32	32	64
64	64	64	64	64	64	64	64

ضرایب کوانتیزه‌شده

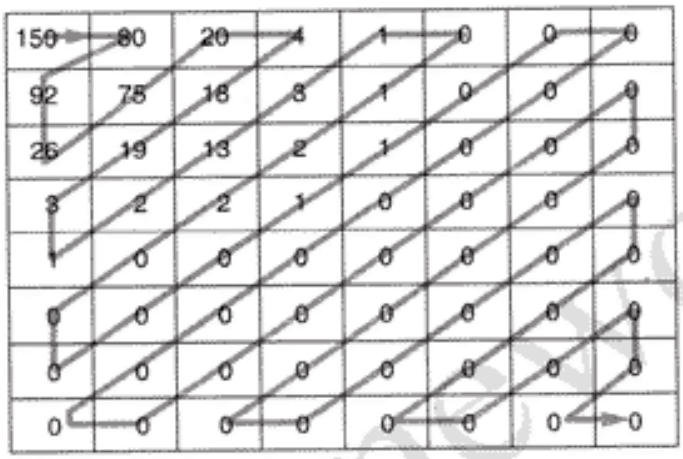
150	80	20	4	1	0	0	0
92	75	18	3	1	0	0	0
26	19	13	2	1	0	0	0
3	2	2	1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

شکل ۷-۷۴. محاسبهٔ ضرایب کوانتیزهٔ DCT .

در مرحلهٔ چهارم، بجای عنصر (0, 0) هر بلوک (گوشهٔ چپ-بالا) تفاضل آن با همین عنصر از جدول قبل نوشته می‌شود. از آنجائیکه این عناصر مقدار متوسط هر بلوک هستند، اختلاف آنها زیاد نیست و عددی که بدست

می‌آید نسبتاً کوچک است. فقط روی این عنصر است که تفاضل محاسبه می‌شود، نه عناصر دیگر. به عنصر (0, 0) هر جدول مؤلفهٔ DC (و به سایر عناصر، مؤلفهٔ AC) گفته می‌شود.

در مرحلهٔ پنجم تمام 64 عنصر جدول با روش کُدکردن run-length خطی می‌شوند. برای این کار از اسکن زیگزاگی جدول استفاده می‌شود (شکل ۷-۷۵)، چون در اسکن افقی-عمودی 0 ها کنار هم قرار نمی‌گیرند. در مثال شکل ۷-۷۵، اسکن زیگزاگی باعث شده تا ۳۸ صفر متوالی در انتهای ماتریس بدست آید. این ۳۸ عدد را می‌توان بطور خلاصه «۳۸ صفر» نامید (و جدول را خیلی کوچک کرد).



شکل ۷-۷۵. روش خطی کردن عناصر ماتریس کوانتیزه شده.

پس از اجرای مرحلهٔ پنجم، لیستی از اعداد (در فضای تبدیل) بدست می‌آید که نمایندهٔ تصویر فشرده شده هستند. در مرحلهٔ آخر برای کوچک کردن هر چه بیشتر این لیست، به اعدادی که بیشتر تکرار شده‌اند کُدهای کوچکتر داده می‌شود، و به اعداد با تکرار کمتر کُدهای بلندتر (**کُدهافمن**).

شاید JPEG پیچیده بنظر برسد، چون در واقع پیچیده هم هست. اما از آنجائیکه ضریب فشرده‌سازی در آن بسیار بالاست (نزدیک 20:1)، کاربرد گسترده‌ای پیدا کرده است. برای دیکُد کردن تصاویر JPEG ، الگوریتم بالا بصورت معکوس انجام می‌شود. JPEG تا حد زیادی متقارن است: دیکُد کردن تقریباً به همان اندازهٔ کُد کردن وقت می‌برد. اما همانطور که خواهید دید، کُدهایی هم هستند که بشدت نامتقارنند.

استاندارد MPEG

بالاخره به اصل مطلب رسیدیم: استانداردهای MPEG (Motion Picture Experts Group) . این استانداردها از سال ۱۹۹۳ بصورت بین‌المللی برای فشرده‌کردن ویدئو بکار برده می‌شوند، چون می‌توانند صدا و تصویر را با هم فشرده کنند (و ویدئو هم ترکیبی است از صدا و تصویر). تا اینجا فشرده‌سازی صدا و تصویر ثابت را دیدید. پس اجازه دهید فشرده‌سازی ویدئو را بررسی کنیم.

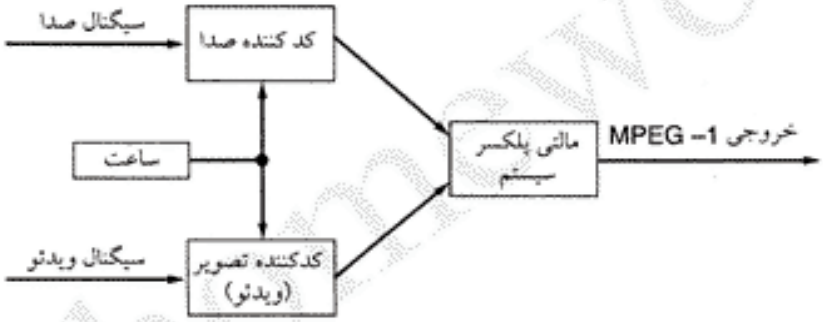
اولین استانداردی که نهایی شد، MPEG-1 بود (ISO 11172) . هدف این استاندارد تولید خروجی با کیفیت ضبط ویدئو (وضوح 240 × 352 در سیستم NTSC) با نرخ انتقال 1.2 Mbps بود. ویدئوی 240 × 352 با رنگ 24 bit/pixel و سرعت 25 frame/sec به پهنای باند 50.7 Mbps نیاز دارد، پس تقلیل آن به 1.2 Mbps یعنی فشرده‌سازی 40:1 ، و این اصلاً کار ساده‌ای نیست. از MPEG-1 برای انتقال ویدئو روی کابل زوج تابیده (در مسافتهای نه چندان زیاد)، و ذخیره کردن فیلم روی CD-ROM استفاده می‌شود.

استاندارد بعدی این خانواده MPEG-2 بود (ISO 13818)، که برای فشرده‌سازی دیدئو 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 ماهواره‌ای دیجیتال استفاده می‌شود.

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 فرمت‌ها و روشهای کُد 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 می‌کند (شکل ۷-۷۶ را ببینید). صدا و ویدئو بصورت جداگانه و مستقل کُد می‌شوند، که این کار مشکل سنکرون کردن آنها در گیرنده را پیش می‌آورد. این مشکل با استفاده از یک ساعت سیستم 90-kHz (که وقت فعلی را به هر دو قسمت صدا و ویدئو می‌دهد) حل شده است. اینها اعداد ۳۳ بیتی هستند، بنابراین یک فیلم می‌تواند بدون هیچ مشکلی حتی ۲۴ ساعت طول بکشد (بدون اینکه مسئلهٔ صفر شدن تایمر پیش آید). این برچسبهای زمانی در خروجی کُدشده هم نوشته می‌شوند، و گیرنده می‌تواند از آنها برای سنکرون کردن صدا و ویدئو استفاده کند.



شکل ۷-۷۶. سنکرون کردن صدا و ویدئو در MPEG-1.

اکنون اجازه دهید فشرده‌سازی 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 می‌کند. برای بکارگیری افزونگی فضایی، هر فریم بطور جداگانه با JPEG کُد می‌شود. این رهیافت بیشتر در مواقعی بکار برده می‌شود که (علاوه بر پخش فیلم) به تک تک فریمها نیز (برای کارهایی مانند تدوین فیلم) احتیاج داشته باشیم. با این روش می‌توان به پهنای باند 8-10 Mbps دست یافت.

برای رسیدن به فشردگی بیشتر می‌توان 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 می‌رسد زیاد نیست، چون در اغلب فیلمها هر ۳ یا ۴ ثانیه (بطور متوسط ۷۵ فریم) صحنه بکلی عوض می‌شود. اما همین توالیهای ۷۵ فریمی تقریباً یکسان هم به کاهش قابل‌ملاحظه‌ای (در مقایسه با JPEG) منجر خواهد شد.

در صحنه‌هایی که زمینه یا دوربین ثابت است و هنرپیشه‌ها حرکت کمی دارند، تقریباً تمام پیکسلها در فریمهای متوالی شبیه هم هستند. در این حالت، محاسبهٔ تفاضل دو فریم و اجرای الگوریتم JPEG روی این تفاضل، بخوبی کار خواهد کرد. اما در صحنه‌هایی 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 می‌دهد؛ و تفاوت MPEG و JPEG نیز در همین جاست.

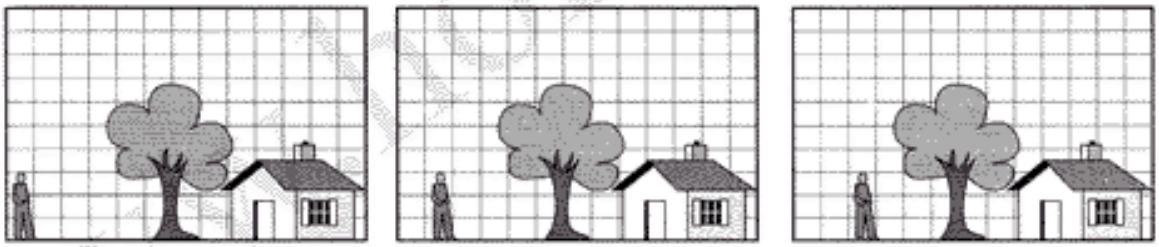
در خروجی MPEG-1 چهار نوع فریم می‌تواند وجود داشته باشد:

۱. فریمهای I (Intracoded): تصاویر ثابت و مستقل JPEG .
۲. فریمهای P (Predictive): تفاوت بلوک-به-بلوک با فریم قبلی.
۳. فریمهای B (Bidirectional): تفاوتهای بین فریم قبلی و بعدی.
۴. فریمهای D (DC-coded): متوسط بلوک‌ها برای جلو رفتن 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 می‌کنند. سه دلیل برای قرار دادن فریمهای I در استریم خروجی وجود دارد. اول اینکه، از MPEG-1 در سیستمهای چندپخشی (multicast) ، که تعداد زیادی بینندهٔ مستقل دارند، نیز استفاده می‌شود. 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 نمی‌تواند ببیند (چون در واقع مبنایی برای دیکُد کردن فریمها ندارد). دوم، اگر یکی از فریمها خراب شود، دیگر نمی‌توان فریمهای بعدی را دیکُد 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 دیکُد کند. به این دلایل، در هر ثانیه یک یا دو فریم I در خروجی قرار داده می‌شود.

بر خلاف فریمهای I ، فریمهای P فقط اختلاف بین فریمها را کُد می‌کنند. فریمهای P از ایدهٔ ماکروبلوک (macroblock) - ماتریسهای 16 × 16 پیکسل در فضای روشنایی، و ماتریسهای 8 × 8 پیکسل در فضای رنگ - استفاده می‌کنند. هر ماکروبلوک با جستوی بلوک مشابه در فریم قبلی (برای یافتن تشابه یا اختلاف) کُد می‌شود.

شکل ۷-۷۷نمونه‌ای که در آن فریمهای P بکار می‌آیند، را نشان می‌دهد. در اینجا سه فریم متوالی می‌بینید که زمینهٔ صحنه در آنها یکسان است، و فقط جای هنرپیشه عوض شده است. در این فریمها، ماکروبلوک‌های زمینهٔ صحنه دقیقاً یکسان مانده، و فقط ماکروبلوک‌های مربوط به هنرپیشه عوض شده و باید کُد شود.



شکل ۷-۷۷. سه فریم متوالی.

استاندارد MPEG-1 هیچ حرفی دربارهٔ روش جستجو، عمق جستجو، و یا مفهوم یکسان بودن، نمی‌زند - تمام اینها برعهدهٔ نویسندهٔ برنامه گذاشته شده است. برای مثال، در یک برنامه ممکنست جستجوی ماکروبلوک در مکان فعلی در فریم قبلی، و در تمام آفست‌های $\pm\Delta x$ (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 آستانهٔ تعریف شده بالاتر باشد) بعنوان برنده انتخاب می‌شود. در غیراینصورت، گفته می‌شود 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، کُد run-length ، و کُد هافمن) کُد می‌شود. مقدار این ماکروبلوک در استریم خروجی بصورت بردار حرکت آن (شامل مقدار و جهت جابجایی) ثبت می‌شود. اگر یک ماکرو بلوک در فریم قبلی وجود نداشته باشد، بصورت عادی (فریم I) کُد می‌شود.

این الگوریتم بشدت نامتقارن است. برنامه آزاد است هر مکان قابل‌قبولی در فریم قبلی را که بخواهد (برای

یافتن ماکروبلوک موردنظر) امتحان کند. با این روش استریم MPEG-1 می‌تواند به ضریب فشرده‌سازی بالایی دست پیدا کند، ولی در ضمن زمان کُد کردن هم بالا خواهد رفت. همانطور که می‌توان حدس زد، این رهیافت برای کُد کردن فیلمهای کتابخانه‌ای مناسب است، ولی اصلاً بدرد ویدئوکنفرانس نمی‌خورد.

برنامه‌نویس در اتخاذ تصمیم برای تعریف مفهوم «ماکروبلوک یکسان» نیز آزاد است. این آزادی دست برنامه‌نویس را برای انتخاب نقطهٔ تعادل بین سرعت و کیفیت (در عین اطمینان از اینکه خروجی همواره با MPEG-1 سازگار است) باز می‌گذارد. در هر حال، خروجی یا ماکروبلوک JPEG شده است، یا JPEG شدهٔ اختلاف آن با فریم قبلی.

تا اینجا، دیکُد کردن MPEG-1 ساده است. دیکُد کردن فریمهای I که هیچ فرقی با تصاویر ثابت JPEG ندارد. برای دیکُد 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 می‌سازد. فریم جدید بلوک به بلوک ساخته می‌شود.

فریمهای B شبیه فریمهای P هستند، با این تفاوت که فریم مبنای آنها می‌تواند (بجای فریم قبلی) فریم بعدی نیز باشد. این آزادی کمک زیادی به جبران حرکت صحنه‌ها می‌کند، بویژه وقتی اشیاء روی صحنه از جلو یا پشت اشیاء دیگر عبور می‌کنند. برای دیکُد 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 فشرده‌سازی را بالا می‌برند، همهٔ برنامه‌های MPEG از آنها پشتیبانی نمی‌کنند.

فریمهای D فقط برای نمایش تصویری با وضوح پائین هنگام جلو یا عقب رفتن سریع فیلم (FF یا Rewind) بکار برده می‌شوند. دیکُد کردن MPEG-1 با سرعت پخش معمولی باندازهٔ کافی دشوار هست، انجام همین کار با سرعت ۱۰ برابر که دیگر جای خود دارد. بهمین دلیل در سرعتهای بالا از فریمهای D (که وضوح کمتری دارند) استفاده می‌شود. هر فریم D در واقع فقط مقدار متوسط هر بلوک (بدون هر گونه کُد کردن اضافی) است، که نمایش آنرا در زمان واقعی آسان می‌کند. با این ویژگی می‌توان یک فیلم را با سرعت زیاد بدنبال صحنهٔ موردنظر جستجو کرد. فریمهای D معمولاً درست قبل از فریمهای I قرار داده می‌شوند، 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 نمی‌کند. همچنین، در DCT بجای ماتریسهای 8 × 8 از بلوکهای 10 × 10 استفاده می‌کند، که با این کار تعداد ضرایب ۵۰ درصد بیشتر شده و کیفیت بالاتر می‌رود. از آنجائیکه MPEG-2 برای پخش تلویزیونی و DVD طراحی شده، از تصاویر خط-در-میانی و پیشرونده پشتیبانی می‌کند (در حالیکه MPEG-1 فقط از تصاویر پیشرونده پشتیبانی می‌کند). بین این دو استاندارد تفاوتهای کوچک دیگری نیز وجود دارد.

بجای یک وضوح ثابت، MPEG-2 از چهار وضوح پشتیبانی می‌کند: کم (240 × 352)، اصلی (480 × 720)، بالا-۱۴۴۰ (1152 × 1440)، و بالا (1080 × 1920). وضوح 240 × 352 برای دستگاههای VCR (و سازگاری با MPEG-1) در نظر گرفته شده است. وضوح اصلی برای پخش NTSC بکار می‌رود. دو وضوح دیگر برای HDTV در نظر گرفته شده‌اند. MPEG-2 در وضوحهای بالا معمولاً با سرعت 4-8 Mbps اجرا می‌شود.

۸.۴.۷ پخش فیلم بر حسب تقاضا

پخش فیلم برحسب تقاضا (Video on demand - VOD) چیزی شبیه مغازهٔ کرایهٔ فیلمهای ویدئویی است. در آنجا مشتری یکی از فیلمهای موجود را انتخاب کرده و برای تماشا به منزل می‌برد. ولی در اینجا نیازی به مراجعه به

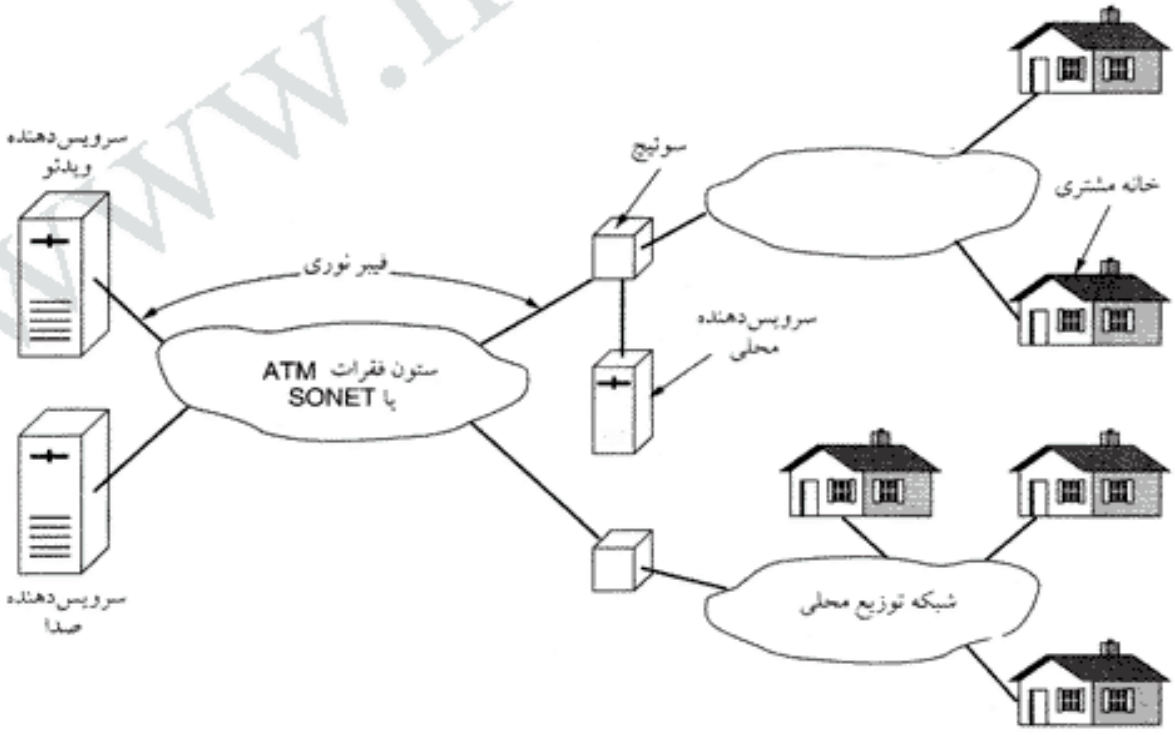
مغازه نیست و انتخاب فیلم از طریق دستگاه کنترل از راه دور تلویزیون انجام شده، و پخش آن هم بلافاصله شروع می‌شود. لازم به گفتن نیست که، پیاده‌سازی VOD از تعریف آن کمی مشکلتر است.

آیا پخش فیلم بر حسب تقاضا شبیه کرایهٔ فیلم ویدئویی است، یا انتخاب کانال در تلویزیونهای کابلی؟ پاسخ این سئوال تبعات فنی مهمی دارد. وقتی یک فیلم ویدئویی کرایه می‌کنید، می‌توانید وسط فیلم آنرا متوقف کرده، و بعد از خوردن یک فنجان چای یا جواب دادن به تلفن، ادامهٔ آنرا از همانجایی که متوقف شده بود، تماشا کنید. اما بینندگان کانالهای تلویزیونی نمی‌توانند چنین کاری بکنند.

اگر VOD بخواهد رقابت مؤثری با مغازه‌های کرایهٔ فیلم داشته باشد، باید به مشتری اجازه دهد فیلم را بدلخواه 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 کافیست فیلمهای پُرطرفدار را در فواصل ۱۰ دقیقه‌ای (و بدون توقف تا آخر) پخش کنیم. در این حالت، مشتری برای دیدن این فیلم فقط کافیست (حداکثر) ۱۰ دقیقه صبر کند. با اینکه متوقف کردن چنین فیلمی وجود ندارد، اما اگر وسط آن کاری برایتان پیش آمد، کافیست 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» می‌نامند. هزینهٔ پیاده‌سازی چنین سیستمی بسیار کمتر است، چون می‌توان هر فیلم را برای عدهٔ 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 بالقوهٔ شبکه‌های پهن‌باند امروزی است، که در شکل ۷-۷۸ یکی از مدلهای متداول آنرا می‌بینید. در مرکز این سیستم یک شبکهٔ با پهنای باند زیاد (ملی یا بین‌المللی) قرار دارد، که هزاران شرکت توزیع‌کننده (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 سرویس‌دهنده نیز با فیبرهای نوری پُرظرفیت به ستون‌فقرات 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 خریدازخانه (home shopping) ارائه می‌کنند. چنین شبکه‌ای می‌تواند سرویسهایی مانند اخبار، ورزش، فیلم و سریال، دسترسی وب، و هزاران امکان بالقوهٔ دیگر در اختیار کاربران قرار دهد.

در این سیستم سرویس‌دهنده‌های 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: **سرویس‌دهنده‌های ویدئو** (video server) و **شبکهٔ توزیع** (distribution network) .

سرویس‌دهنده‌های ویدئو

برای ایجاد یک سیستم VOD (یا «تقریباً VOD») به سرویس‌دهنده‌های 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 (TB = ترابایت) نیاز داریم. اگر فیلمها و سریالهای قدیمی تلویزیونی، فیلمهای خبری و ورزشی، و کاتالوگهای ویدئویی را هم به آن اضافه کنیم، آن وقت متوجه می‌شوید که مشکل کجاست!

ارزانترین وسیلهٔ برای ذخیره کردن حجم زیادی از اطلاعات، نوار مغناطیسی است - و بنظر می‌رسد در آیندهٔ نزدیک وسیلهٔ ارزانتری به بازار نخواهد آمد. روی یک نوار مغناطیسی 200-GB می‌توان ۵۰ فیلم MPEG-2 (با هزینهٔ ۱ تا ۲ دلار برای هر فیلم) ذخیره کرد. امروزه سرویس‌دهنده‌های 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 (بخصوص فیلم پنجاهم)، سرعت انتقال، و محدودیت تعداد دستگاههای پخش است (برای پخش همزمان n فیلم، به n دستگاه پخش نیاز داریم).

خوشبختانه، تجربهٔ مغازه‌های کرایهٔ 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:

$$C = 1/(1 + 1/2 + 1/3 + 1/4 + 1/5 + \cdots + 1/N)$$

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 واقعیت 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.

گزینهٔ 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

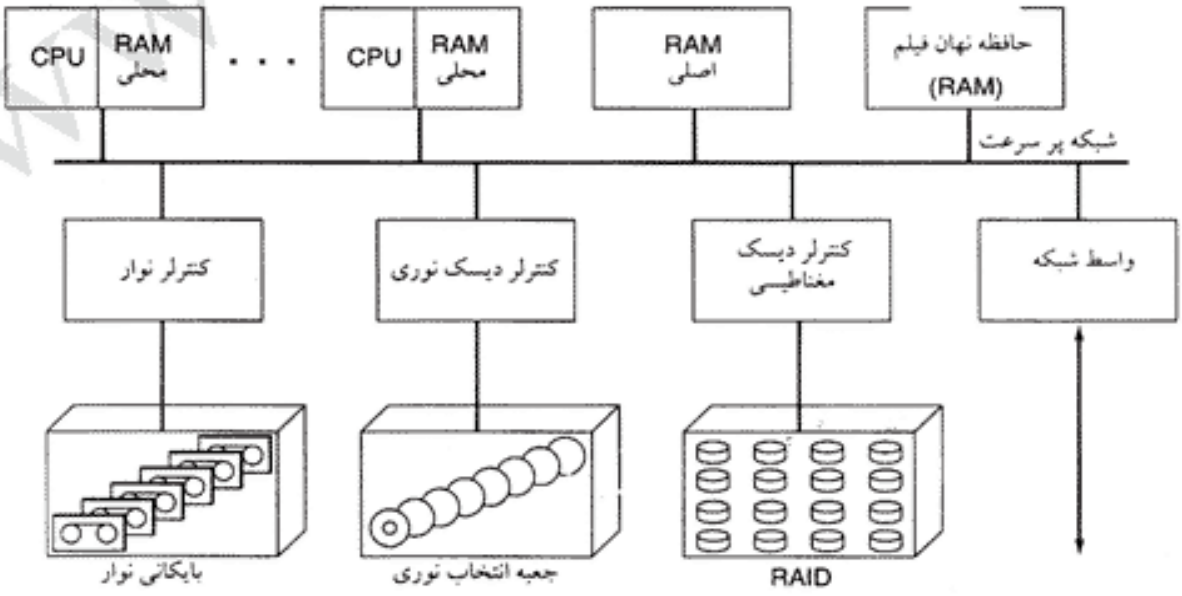
شکل ۷-۷۹. سلسله مراتب ذخیره‌سازی در سرویس‌دهنده‌های ویدئو.

(50 msec در مقابل 5 msec)، ولی آنها ارزانتر و مقاومتر هستند، و بزودی شاهد سرویس‌دهنده‌های ویدئو با ظرفیت هزاران DVD در بازار خواهیم بود.

در مرحلهٔ 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 <) گزینهٔ خوبی برای ذخیره کردن فیلمهای پُربیننده هستند. عیب عمدهٔ این دیسکها قیمت بالای آنهاست.

در بالاترین نقطهٔ هرم شکل ۷-۷۹ حافظهٔ RAM قرار دارد. با اینکه قیمت RAM در سالهای اخیر بشدت کاهش پیدا کرده، ولی ذخیره کردن یک فیلم MPEG-2 به ۲۰۰ دلار حافظهٔ RAM نیاز دارد - و برای ذخیره کردن فقط ۱۰۰ فیلم باید ۲۰,۰۰۰ دلار هزینه کنیم (که البته برای یک سرویس‌دهندهٔ ویدئوی سطح بالا چندان زیاد و غیرعملی نیست).

از آنجائیکه یک سرویس‌دهندهٔ ویدئو اساساً دستگاهیست برای I/O بی‌درنگ در حجمهای بالا، سخت‌افزار و نرم‌افزار آن بایستی تفاوت اساسی با کامپیوترهای ویندوز و یونیکس معمولی داشته باشد. در شکل ۷-۸۰ معماری سخت‌افزاری یک سرویس‌دهندهٔ ویدئوی نوعی را می‌بینید. قسمتهای مختلف این سرویس‌دهنده عبارتند از: یک یا چند CPU سریع (با مقداری RAM اختصاصی برای هر کدام)، یک حافظهٔ اصلی مشترک، یک حافظهٔ نهان بزرگ از نوع RAM برای فیلمهای پُربیننده، ترکیبی از وسایل ذخیره‌سازی متنوع برای ذخیره کردن فیلمها، و سخت‌افزار 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 می‌شوند.



شکل ۷-۸۰. معماری سخت‌افزاری یک سرویس‌دهندهٔ ویدئو.

نرم‌افزار سرویس‌دهندهٔ ویدئو خود داستان دیگریست. در این سیستم، وظیفهٔ 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 بی‌درنگ استفاده کنیم (البته نه در همهٔ CPU‌ها). در این سیستمها، هر وظیفه به بخشهای کوچکتر تقسیم می‌شود، و هر بخش باید در زمان معین به پایان برسد. برای زمانبندی این وظایف می‌توان از الگوریتمهایی مانند «نزدیکترین بن‌بست‌درنوبت‌بعد» (nearest deadline next) یا «نرخ‌یکنواخت» (rate monotonic) استفاده کرد (Liu and Layland, 1973).

این نرم‌افزار همچنین خصلت واسط سمت مشتری (سرویس‌دهنده‌های بینابینی و گیرنده) را نیز تعیین می‌کند. دو نوع واسط کاربر بیشتر رواج دارند. اولی یک سیستم فایل معمولی است، که مشتری می‌تواند فایلهای موردنظرش را باز کرده، بخواند، بنویسد، و سپس ببندد. چنین واسطی می‌تواند سیستم فایلی شبیه یونیکس داشته باشد (البته با در نظر داشتن مشکلاتی که از ساختار سلسله مراتبی سیستم و بی‌درنگ بودن آن ناشی می‌شود).

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، سرویس‌دهنده (بدون نیاز به فرمان اضافی) داده‌ها را به سمت مشتری پمپ می‌کند.

قلب سرویس‌دهندهٔ ویدئو نرم‌افزار «مدیریت‌دیسک» (disk management) است. این نرم‌افزار دو وظیفهٔ اصلی دارد: انتقال فیلم از نوار مغناطیسی یا دیسک نوری به دیسک مغناطیسی، و انجام بموقع درخواست‌های خواندن دیسک. وظیفهٔ اول (یعنی انتقال فیلم) نقش مهمی در افزایش کارایی سیستم دارد.

برای سازماندهی دیسکها دو روش وجود دارد: **مزرعهٔ‌دیسک** (disk farm) ، و **آرایهٔ‌دیسک** (disk array) . در روش اول، مزرعهٔ دیسک، روی هر دیسک چند فیلم ذخیره می‌شود (که برای کارایی و اطمینان بیشتر می‌توان هر فیلم را روی چند دیسک - حداقل دو تا - ذخیره کرد). در روش دوم، آرایهٔ دیسک یا RAID (**آرایهٔ‌افزونه‌از دیسکهای‌ارزان** - Redundant Array of Inexpensive Disks)، هر فیلم روی چندین دیسک مختلف پخش می‌شود. به این روش - که در آن فیلم به n نوار باریک تقسیم شده، و این نوارها روی دیسکهای 0 تا $n-1$ ذخیره می‌شود - **نوارکردن**(striping) گفته می‌گویند.

یک آرایهٔ دیسکهای نواری چندین مزیت نسبت به مزرعهٔ دیسک دارد. اول، همهٔ n دیسک را می‌توان بطور همزمان خواند یا نوشت، و این کارایی سیستم را n برابر می‌کند. دوم، می‌توان 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 پُربیننده را بصورت دستی بین دیسکهای مختلف پخش کنیم. از طرف دیگر، آرایهٔ دیسک تکنیکی پیچیده است، و اگر چندین دیسک با هم خراب شوند، دسترسی به کل فیلمها غیرممکن خواهد شد. پیاده‌سازی ویژگیهای «سریع به جلو» یا «سریع به عقب» (که جزء الزامات واسط نوع دوم هستند) نیز در این سیستم بسیار دشوار است.

وظیفهٔ دیگر نرم‌افزار مدیریت دیسک دادن سرویس بی‌درنگ به درخواستهای مشتریان است. تا چند سال پیش چنین وظیفه‌ای مستلزم طراحی الگوریتمهای پیچیدهٔ زمانبندی دیسک بود، ولی امروزه با کاهش شدید قیمت RAM روشهای بسیار ساده‌تری ممکن شده است. برای اینکه بتوان هر استریم را بصورت بی‌درنگ به مشتری فرستاد، برای هر فیلم بافری از ۱۰ ثانیه ویدئو (معادل 5 MB) در RAM نگه داشته می‌شود. این بافر توسط واسط دیسک پُر، و توسط واسط شبکه خالی می‌شود. با 500 MB RAM می‌توان 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 بکارگیری دیسکهای SCSI جدید بسادگی امکانپذیر است.

شبکهٔ توزیع

شبکهٔ توزیع (distribution network) عبارتست از مجموعهٔ سوئیچها 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 شبکهٔ توزیع محلی متصل است. معمولاً سوئیچینگ فقط در ستون فقرات وجود دارد، نه در شبکهٔ توزیع محلی.

مهمترین چیزی که ستون فقرات باید داشته باشد، پهنای باند زیاد است. پائین بودن میزان لرزش (jitter – وقفه‌های کوچک و ناخوشایندی که در اثر ترافیک زیاد در پخش صدا و تصویر رُخ می‌دهد) از دیگر الزامات ستون فقرات است، ولی از آنجائیکه ضعیفترین کامپیوترهای امروزی نیز می‌توانند حداقل ۱۰ ثانیه ویدئو با کیفیت MPEG-2 را بافر کنند، این ویژگی اهمیت سابق خود را از دست داده است.

بی‌نظمی عجیبی بر شبکه‌های محلی حکمفرماست، چون شرکتهای زیادی سعی می‌کنند تا خدمات متنوع خود را به مشتریان بفروشند. شرکتهای تلفن، تلویزیون کابلی (و اخیراً شرکتهای برق) متقاعد شده‌اند که برای برنده شدن در این میدان رقابت باید نفر اول باشند. در نتیجه، هر روز تکنولوژی جدیدی وارد بازار مصرف می‌شود. در ژاپن حتی شرکتهای فاضلاب وارد تجارت اینترنت شده‌اند، با این استدلال که گسترده‌ترین شبکهٔ خطوط لوله متعلق به آنهاست، و می‌توانند فیبرهای نوری را به هر خانه‌ای بکشند (فقط باید دقت کنند کابلهای آنها از کجا سر در می‌آورد!). 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 شبکه‌های توزیع محلی شدند. در فصل ۲ دربارهٔ تکنولوژی ADSL صحبت کردیم، و نیازی به تکرار آن نیست. ایدهٔ اصلی ADSL استفاده از سیمهای مسی است که تقریباً به هر خانه‌ای (در اروپا، آمریکا و ژاپن) کشیده شده‌اند. اگر می‌شد 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 موردنظر) نیاز دارند، و ADSL (جز در مسافتهای بسیار کوتاه) نمی‌تواند چنین سرعتی ارائه کند.

دومین طرح شرکتهای تلفن FTTC (فیبرنوری تاکوچه – Fiber To The Curb) است. در FTTC ، شرکت تلفن از ایستگاه پایانی (end office) یک رشته فیبر نوری به هر محله می‌کشد، و آنرا به دستگاهی موسوم به ONU (واحدشبکهٔ نوری – Optical Network Unit) وصل می‌کند. به هر ONU می‌توان تا ۱۶ زوج سیم مسی تلفن وصل کرد. با این تمهید، طول کابلهای مسی آنقدر کوتاه می‌شود که دستیابی به T1 یا T2 دو-طرفهٔ همزمان (که بترتیب برای MPEG-1 و MPEG-2 مناسبند) را ممکن می‌سازد. این سرویس (بعلت متقارن بودن) برای ویدئوکنفرانس نیز کاملاً مناسب است.

سومین راه حل شرکتهای تلفن کشیدن فیبر نوری تا در خانهٔ مشتریان است، که FTTH (فیبرنوری تا خانه Fiber To The Home) نام دارد. در این طرح، هر فرد می‌تواند یک کاربر OC-1 ، OC-3 ، یا حتی بیشتر خانه‌اش داشته باشد. FTTH بسیار گران است و بزودی عملی نخواهد شد، ولی می‌توان تصور کرد که چه امکاناتی در اختیار مشتریان می‌گذارد. در شکل ۷-۶۳ دیدید که چگونه هر فردی می‌تواند یک ایستگاه رادیویی شخصی راه بیندازد؛ با داشتن FTTH راه‌اندازی 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 (آمیختهٔ فیبر-کواکس – Hybrid Fiber Coax) که از سوی شرکتهای تلویزیون کابلی ارائه شده

کلی با سه روش قبلی متفاوت است (به شکل ۲-۴۷الف نگاه کنید). جریان از این قرار است: شرکتهای تلویزیون کابلی در حال تعویض کابلهای کواکس 300-450 MHz (با ظرفیت ۵۰ تا ۷۵ کانال 6-MHz) با کابلهای 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 می‌گیرند.

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 می‌آوریم. با نزدیکتر کردن جعبه تقسیم سیستم به خانه‌های مشتریان، می‌توان به هر ۵۰۰ خانه یک کابل رساند. با یک حساب سرانگشتی می‌توان دید که به هر خانه پهنای باندی معادل 4 Mbps می‌رسد، که برای پخش فیلمهای MPEG-2 کافیست.

هیجان‌انگیز است، نه؟ اما این کار مستلزم آن است که شرکتهای تلویزیون کابلی تمام کابلهای قدیمی را با کابلهای 750 MHz جایگزین کنند، جعبه تقسیمهای جدید نصب کنند، و تمام تقویت‌کننده‌های یکطرفه را هم جمع کنند - و این یعنی عوض کردن کل سیستم تلویزیون کابلی. هزینهٔ این کار با ایجاد یک زیرساخت کامل FTTC قابل مقایسه است. در هر دو سیستم، شرکتهای عمل‌کننده 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 می‌رود، ولی در HFC یک کابل کواکس مشترک بین تمام خانه‌ها کشیده می‌شود. همانطور که می‌بینید، این دو سیستم آنقدرها که صاحبان آنها ادعا می‌کنند، با هم متفاوت نیستند.

با این حال یک تفاوت اساسی هست که باید به آن اشاره کرد: HFC از یک رسانهٔ مشترک (بدون هیچگونه سوئیچینگ و مسیردهی) استفاده می‌کند. هر اطلاعاتی که روی این کابل فرستاده شود، بطور بالقوه بوسیلهٔ تمام مشترکان قابل دریافت است. اما FTTC سیستمی است مبتنی بر سوئیچینگ، و چنین چیزی در آن اتفاق نمی‌افتد. اگر متولیان HFC بخواهند کسی بدون پرداخت پول قادر به تماشای فیلمهای 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 شرکتهای پخش رسانه، رمزنگاری ایدهٔ خوبیست یا خیر؟ اغلب شرکتهای تلفن چنین کاری نمی‌کنند، با این توضیح که مایل به افت کیفیت نیستند (ولی در واقع برای ضرر زدن به رقبای HFC).

بعد از این سیستمهای توزیع، نوبت به سرویس‌دهنده‌های ویدئوی محلی می‌رسد. آنها در واقع نسخهٔ کوچکی از سرویس‌دهنده‌های ویدئو (که در بالا توضیح دادیم) هستند. نقش اصلی این سرویس‌دهنده‌ها کاستن از ترافیک ستون فقرات است.

از این سرویس‌دهنده‌های محلی می‌توان برای رزرو کردن فیلمها استفاده کرد. اگر مشتریان از مدتی قبل تمایل خود را برای دیدن فیلمی به شرکت پخش رسانه اعلام کنند، این فیلم می‌تواند در ساعات خلوت شبکه در سرویس‌دهندهٔ محلی بار شود. این روش می‌تواند به کاهش قیمتها نیز منجر شود. برای مثال، اپراتور شبکه می‌تواند برای ساعات کم مصرف در تعرفه‌های خود تخفیف دهد.

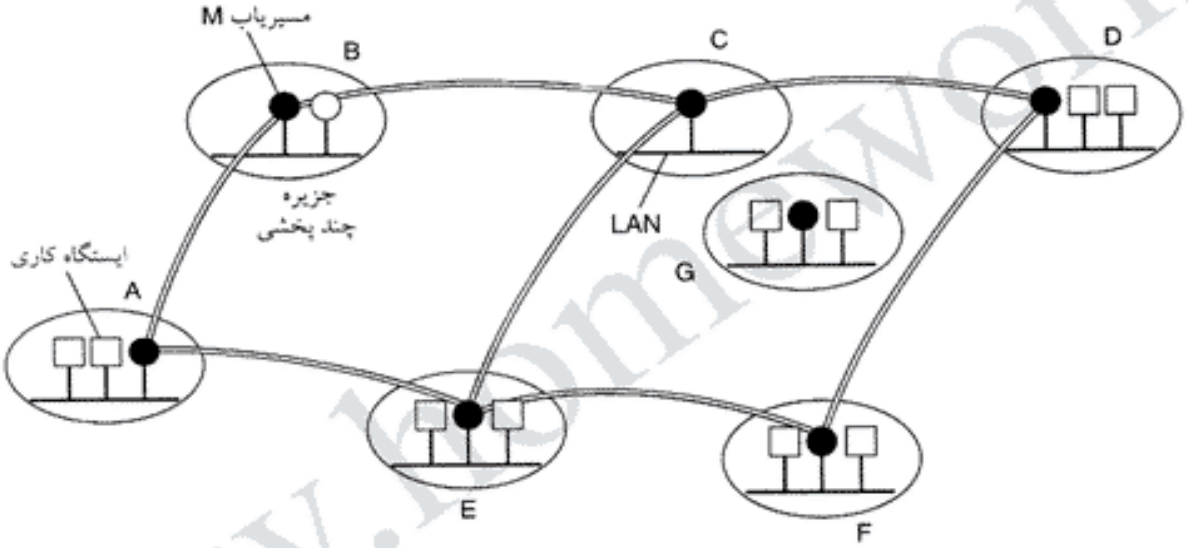
۹-۴-۷ ستون‌فقرات چندپخشی - MBone

در همان حالیکه شرکتهای تلفن و تلویزیون کابلی در حال طرح نقشه‌های بزرگ برای آیندهٔ «VOD دیجیتال» هستند، جامعهٔ اینترنت به آرامی در حال پیاده‌سازی سیستم چندرسانه‌ای دیجیتال خاص خود است: MBone (ستون فقرات چندپخشی - Multicast Backbone). در این قسمت خواهید دید که MBone چیست، و چگونه کار می‌کند.

MBone را می‌توان تلویزیون اینترنتی دانست. بر خلاف VOD (که تأکید روی تماس کاربر و پخش فیلمهای از پیش فشرده‌شده از روی یک سرویس‌دهندهٔ ویدئو است)، MBone بیشتر برای پخش ویدئوی دیجیتالی زنده

از سراسر دنیا و از طریق اینترنت مورد استفاده قرار می‌گیرد. این سیستم از سال ۱۹۹۲ عملیاتی شده، و کنفرانسهای علمی، از جمله گردهمایی‌های IETF ، و رخدادهای مهم علمی (مانند پرتاب شاتل فضایی) از طریق آن پخش شده است. یکی از کنسرتهای گروه رولینگ استونز، و بخشهایی از جشنوارهٔ فیلم کن نیز از طریق MBone پخش شده‌اند (البته در اینکه بتوان اینها را رخدادهای مهم علمی بشمار آورد، جای بحث است).

از نظر فنی، MBone شبکه‌ایست مجازی روی اینترنت، که تشکیل شده است از تعدادی **جزیرهٔ چندپخشی** (multicast island) که بوسیلهٔ چند **تونل** (tunnel) به هم متصل شده‌اند (شکل ۷-۸۱ را ببینید). در این شکل، MBone شش جزیره دارد (A تا F)، که بوسیلهٔ هفت تونل به هم وصل شده‌اند. هر جزیره (که یک LAN ، یا چند LAN متصل به هم است) از چندپخشی سخت‌افزاری به کامپیوترهای میزبان خود پشتیبانی می‌کند. بسته‌های MBone از طریق تونل‌ها بین جزایر منتشر می‌شود. شاید روزی در آینده، که تمام مسیرهای اینترنت توانایی جابجایی ترافیک چندپخشی را بدست آورند، دیگر نیازی به این ساختار نباشد، ولی فعلاً که کار ما را راه می‌اندازد.



شکل ۷-۸۱ MBone تشکیل شده است از جزایر چندپخشی، که بوسیلهٔ تونل به هم متصل شده‌اند.

هر جزیره دارای یک یا چند مسیریاب ویژه موسوم به **مسیریاب چندپخشی** (mrouter - multicast router) است. برخی از اینها مسیریابهای معمولی هستند، ولی برخی دیگر کامپیوترهای یونیکس بیش نیستند که نرم‌افزار خاصی را (در سطح root) اجرا می‌کنند. مسیریابهای چندپخشی از نظر منطقی به تونل‌ها متصل می‌شوند. بسته‌های MBone در داخل بسته‌های IP پیچیده شده و بصورت بسته‌های **تک‌پخشی** (unicast) معمولی به آدرس مسیریاب چندپخشی مقصد ارسال می‌شوند.

تونل‌ها را باید بصورت دستی پیکربندی کرد. معمولاً، تونل روی یک مسیر فیزیکی اجرا می‌شود، ولی این الزامی نیست. مثلاً، اگر مسیر فیزیکی یک تونل از بد حادثه دچار مشکل شود، مسیریابهای چندپخشی (که از این تونل استفاده می‌کنند) حتی از این موضوع مطلع نخواهند شد، چون اینترنت بطور خودکار ترافیک بسته‌های IP را به مسیرهای دیگر هدایت خواهد کرد.

وقتی یک جزیرهٔ جدید می‌خواهد به MBone وصل شود (مانند *G* در شکل ۷-۸۱)، سرپرست آن با ارسال پیام به لیست پستی MBone حضور خود را اعلام می‌کند. پس از آن سرپرستان سایتهای مجاور با وی تماس می‌گیرند، تا وی بتواند تونل‌های موردنیاز را پیکربندی کند. حتی گاهی اوقات تونل‌های موجود برای بهینه کردن

توپولوژی شبکه و بهره‌گیری از جزیرهٔ جدیدالتأسیس، آرایش خود را عوض می‌کنند. چون آنها که واقعاً وجود خارجی ندارند، و چیزی نیستند جز چند جدول در حافظهٔ مسیریابها (که آنها را هم براحتی می‌توان اضافه، حذف، و یا جابجا کرد). معمولاً هر کشور متصل به MBone دارای یک ستون‌فقرات و تعدادی جزیرهٔ 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 زیرشبکهٔ 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 فاصلهٔ بل‌من-فورد، بنام DVMRP (**پروتکل مسیریابی چندپخشی بردار فاصله** – Distance Vector Multicast Routing Protocol)، استفاده کرد. برای مثال، در شکل ۷-۸۱ جزیرهٔ C برای ارسال بسته‌های خود به جزیرهٔ A می‌تواند از طریق B یا E (و یا حتی D) اقدام کند. این جزیره برای انتخاب مسیر مناسب، فاصلهٔ 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 همهٔ آنها در یک پایگاه دادهٔ 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 جزیرهٔ خود منتشر می‌کنند، تا ببینند چه کسی به کدام کانالها علاقه دارد. میزبانهایی که بخواهند آن کانال را دریافت کنند، در پاسخ یک بستهٔ IGMP برمی‌گردانند. این پاسخها هم به تناوب فرستاده می‌شوند، تا شبکهٔ 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 بستهٔ جدیدی تولید می‌کند، آن بسته را (به کمک سخت‌افزار چندپخشی) در جزیرهٔ 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 تونلها منهای 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 بستهٔ 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 از 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 (زمان‌زنده‌ماندن – Time To Live) نیز مجهز می‌شوند تا برای همیشه در اینترنت سرگردان نشوند. این

مقدار توسط منبع ارسال بستهٔ چندپخشی تعیین می‌شود. هر تونل MBone دارای یک وزن خاص است، و اگر بسته‌ای بخواهد از یک تونل عبور کند، باید وزن کافی داشته باشد - در غیر اینصورت دور انداخته خواهد شد. برای مثال، اگر به تونل بین قاره‌ای (که از اقیانوس اطلس عبور می‌کند) وزن 128 بدهیم، و بخواهیم یک بستهٔ چندپخشی را در همان قارهٔ مبدأ محبوس کنیم و نگذاریم به قارهٔ دیگر برود، کافیست فیلد TTL آنرا به 127 (یا کمتر) ست کنیم. وقتی یک بسته از تونلی عبور می‌کند، باندازهٔ وزن تونل از TTL آن کم خواهد شد.

با اینکه الگوریتم مسیریابی فوق بخوبی کار می‌کند، محققان زیادی برای بهبود آن تلاش می‌کنند. در یکی از این پیشنهادات از ایدهٔ 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، نقشهٔ کامل جزیره‌ها و تونلهای چندپخشی را هم نگه می‌دارد. با استفاده از این اطلاعات، پیدا کردن بهترین مسیر از هر جزیره به جزیرهٔ دیگر کاری ساده و سر راست خواهد بود. الگوریتم دایکسترا (Dijkstra) یکی از الگوریتمهایی است که می‌توان از آن استفاده کرد.

زمینهٔ 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 Muliticast) توسعه داده است. این پروتکل دو ویرایش دارد: یکی برای مناطق شلوغ (جزیره‌های پُربیننده)، بنام PIM-DM ؛ و دیگری برای مناطق خلوت (جزیره‌های کم‌بیننده)، بنام PIM-SM . هر دو ویرایش، بجای ایجاد لایه‌های توپولوژی خاص مانند کاری که DVMRP و MOSPF انجام می‌دهند، از جدولهای مسیریابی تک‌پخشی استاندارد استفاده می‌کنند.

در PIM-DM مسیرهای زائد و بی‌مصرف حذف می‌شوند. روش حذف شاخه‌های اضافی چنین است. وقتی یک بستهٔ چندپخشی از تونل «اشتباه» به یک گره می‌رسد، این گره از طریق همان تونل یک **بستهٔ حذف** (purne packet) به فرستنده برمی‌گرداند، و از وی می‌خواهد که تا دیگر بسته‌ای در آن مسیر به وی ندهد. وقتی همین گره یک بستهٔ «صحیح» دریافت می‌کند، آنرا به تمام تونلهایی که قبلاً خودشان را (برای آن مسیر) حذف نکرده‌اند، کپی می‌کند. اگر تمام تونلها خود را حذف کرده باشند، و در همان جزیره هم شنونده یا بیننده‌ای برای آن بسته وجود ندارد، مسیریاب چندپخشی یک بستهٔ حذف روی تونل «صحیح» برمی‌گرداند. بدین‌ترتیب، چندپخشی همیشه بطور خودکار خود را با بهترین مسیرها منطبق می‌کند.

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 **نقطهٔ قرار** (rendezvous point) استفاده می‌شود. هر فرستنده در یک گروه چندپخشی PIM-SM بسته‌های خود را به یک نقطهٔ قرار می‌فرستد، و هر سایتی که به محتویات این فرستنده علاقمند باشد، یک تونل به آن نقطهٔ قرار ایجاد می‌کند. با این روش، ترافیک PIM-SM از حالت چندپخشی خارج شده و بصورت تک‌پخشی درمی‌آید. محبوبیت PIM-SM روز به روز در حال افزایش است، و MBone بتدریج به آن سمت حرکت می‌کند (و MOSPF کم‌کم کاربرد خود را از دست می‌دهد). البته خود MBone هم بتدریج به نقطهٔ اشباع و رکود نزدیک می‌شود، و بنظر می‌رسد هرگز نتواند به موفقیت خیره‌کننده‌ای دست یابد.

حتی اگر MBone موفقیت آنچنانی بدست نیاورد، شبکه‌های چندرسانه‌ای همچنان یکی از فیلدهای مهیج و رشدیابندهٔ اینترنت باقی خواهند ماند. هر روز تکنولوژی‌ها و برنامه‌های جدیدی وارد بازار می‌شوند، و چندپخشی

و کیفیت سرویس در حال نزدیک شدن به یکدیگر هستند (Striegel and Manimaran, 2002) . چندپخشی بیسیم یکی دیگر از زمینه‌های پُرتوجه و داغ است (Gossain et al., 2002) . چندپخشی و تمام زمینه‌های مرتبط با آن احتمالاً تا سالهای آینده در مرکز توجه باقی خواهند ماند.

۷-۵ خلاصه

برای نامگذاری در اینترنت از یک سیستم سلسله‌مراتبی بنام سیستم نام ناحیه (DNS) استفاده می‌شود. در بالاترین نقطهٔ این هرم ناحیه‌های شناخته شده، مانند *com* ، *edu* و نزدیک به ۲۰۰ ناحیهٔ کشوری، قرار دارد. DNS یک پایگاه‌دادهٔ توزیع شده است، که سرویس‌دهنده‌های آن در تمام دنیا پخش هستند. وظیفهٔ 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 دادهٔ MIME استفاده کرد. ارسال پیام با برقراری یک اتصال TCP به پورت 25 کامپیوتر مقصد، به کمک پروتکلی بنام SMTP ، انجام می‌شود.

شبکهٔ تارنمای جهانی (وب) نیز تقریباً به همان اندازهٔ 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-سرویس‌دهنده (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 سرویس‌دهنده، دریافت صفحهٔ وب و نمایش آن است، استفاده می‌شود. برای افزایش کارایی وب تکنیکهای مختلفی، مانند حافظهٔ نهان، تکثیر سرویس‌دهنده 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 صفحهٔ نمایش آنها نیز کوچک است؛ ولی نسل بعدی قویتر خواهد بود.

چندرسانه‌ای (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 شناسهٔ جهانی و منحصربفرد 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؟
٤. 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 مشکل‌ساز 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 می‌توان آنرا در دو بسته فرستاد؟

۷. آیا ماشینی با یک نام DNS می‌تواند چند آدرس IP داشته باشد؟ چگونه؟

۸. آیا یک کامپیوتر می‌تواند دو نام DNS (در دو ناحیهٔ 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 ناحیهٔ *com* ثبت کرده‌اند، که باعث فشار بسیار زیاد به سرویس‌دهنده‌های سطح-بالا در این ناحیه شده است. آیا برای رفع این مشکل (بدون تغییر دادن ساختار سیستم و معرفی ناحیهٔ جدید) راهی می‌شناسید؟ 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 می‌کند که اگر گیرنده در مقصد شناسایی نشد، بدنهٔ پیام برگشت داده شود یا خیر. این فیلد در کدام قسمت قرار می‌گیرد: پاکت نامه، یا سرآیند آن؟

۱۱. سیستمهای پُست‌الکترونیک 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. دربارهٔ مشکلات تدوین استاندارد بین‌المللی چنین دایرکتوری بحث کنید.

۱۲. آدرس ایمیل افراد عبارتست از: نام ورود فرد به سیستم @ نام ناحیهٔ DNS با یک رکورد *MX* . این نام ورود می‌تواند هر چیزی (نام کوچک، نام خانوادگی، و یا هر ترکیبی از آن با حروف و اعداد دیگر) باشد. در یک شرکت تعداد زیادی از ایمیل‌ها به هدر می‌رود، چون افراد نام ورود به سیستم گیرنده‌ها را نمی‌دانند. آیا راهی برای حل این مشکل بدون تغییر دادن DNS وجود دارد؟ اگر بله، چگونه؟ اگر خیر، چرا؟

۱۳. اندازهٔ یک فایل باینری ۳۰۷۲بایتی بعد از تبدیل به کُد base64 (با یک جفت CR+LF بعد از هر ۸۰بایت) چقدر خواهد شد؟

۱۴. روش کُدگذاری quoted-printable را در نظر بگیرید. یکی از مشکلات این روش را که در متن کتاب به آن اشاره نشده، نام برده و دربارهٔ راه حل آن بحث کنید.

۱۵. پنج نوع دادهٔ MIME را که در کتاب نیامده، نام ببرید. برای این کار می‌توانید در اینترنت جستجو کنید.

۱۶. فرض کنید می‌خواهید یک فایل MP3 را از طریق ایمیل به دوست خود بفرستید، ولی ISP دوست شما روی ایمیلهای 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 می‌فرستد. متأسفانه این دوست هم برای یک هفته به مرخصی رفته و یک دیمون تعطیلات برای خود ست کرده است. چه اتفاقی می‌افتد؟ آیا این ایمیل‌ها تا برگشتن یکی از این دو نفر مدام بین دو سیستم پاس کاری خواهد شد؟

۱۸. در هر استانداردی، مانند RFC 822 ، قواعد گرامری (حتی ساده‌ترین آنها) باید به دقت تعریف شوند تا سیستمهای مختلف بتوانند با یکدیگر کار کنند. در سرآیندهای SMTP می‌توان از فاصلهٔ سفید (white space – هر یک از کاراکترهای فاصله، *Enter* یا *Tab*) بین توکن‌ها استفاده کرد. دو جایگزین ممکن دیگر برای فاصلهٔ بین توکن‌ها چیست؟

۱۹. دیمون تعطیلات بخشی از عامل کاربر است، یا عامل انتقال پیام؟ البته برای ست کردن دیمون تعطیلات از

عامل کاربر استفاده می‌کنیم، ولی آیا پاسخها را همین عامل برمی‌گرداند؟ توضیح دهید.

۲۰. پروتکل POP3 اجازه می‌دهد تا کاربران ایمیلهای 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. پروتکلهای POP3 و IMAP از دیدگاه یک ISP تفاوت فاحشی دارند: کاربران POP3 معمولاً صندوق پستی خود را خالی می‌کنند، در حالیکه کاربران IMAP نامه‌های خود را برای مدتی نامحدود روی سرویس‌دهنده نگه می‌دارند. شما کدام روش را به یک ISP پیشنهاد می‌کنید؟ دلایل خود را توضیح دهید.

۲۲. پُست وب (Webmail) از POP3 استفاده می‌کند، 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 می‌شود. چرا؟

۲۴. چه زمانی از برنامه‌های کمکی برای دیدن محتویات صفحهٔ وب استفاده می‌شود؟ مرورگر چگونه تشخیص می‌دهد از کدام برنامه باید استفاده کند؟

۲۵. آیا امکان دارد کلیک کردن روی یک لینک در مرورگرهای نت‌اسکیپ و اینترنت اکسپلورر باعث اجرای برنامه‌های کمکی متفاوتی شود (در حالیکه در آن لینک یک نوع MIME مشخص شده است)؟ توضیح دهید.

۲۶. در شکل ۷-۲۱ یک سرویس‌دهندهٔ وب چندریسمانی را ملاحظه می‌کنید. در این سرویس‌دهنده گرفتن درخواست کاربر و چک کردن حافظهٔ نهان 500 μsec طول می‌کشد. در نیمی از مواقع فایل درخواستی در حافظهٔ نهان پیدا شده، و بلافاصله برگردانده می‌شود. در نیمی دیگر، این ماژول باید برای خواندن دیسک 9 msec دیگر صبر کند. (با فرض اینکه دیسک گلوگاه سیستم نباشد) برای استفادهٔ کامل از ظرفیت CPU سرویس‌دهنده چند ماژول باید اجرا کند؟

۲۷. در URL های استاندارد *http* فرض بر این است که سرویس‌دهنده به پورت 80 گوش می‌کند، ولی این بهیچوجه اجباری نیست. روشی طراحی کنید که URL بتواند به پورتهای غیراستاندارد هم دسترسی پیدا کند.

۲۸. URL ها می‌توانند علاوه بر نامهای DNS مستقیماً از آدرس IP هم استفاده کنند (مانند، *http://192.31.231.66/index.html*). مرورگر چگونه می‌تواند تشخیص دهد که بعد از *http://* یک نام DNS آمده یا آدرس IP ؟

۲۹. فرض کنید فردی در دپارتمان کامپیوتر دانشگاه استنفورد برنامه‌ای نوشته و می‌خواهد آنرا از طریق FTP توزیع کند. محل این فایل در کامپیوتر سرویس‌دهنده *ftp/pub/freebits/newprog.c* است – URL آن چه می‌تواند باشد؟

۳۰. در شکل ۷-۲۵، سایت *www.portal.com* تنظیمات کاربر را در یک کوکی نگه می‌دارد. 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 تنبل‌ها») که می‌خواهد مشتریان تنبل خود را هم راضی نگه دارد، سیستمی طراحی کرده که بعد از وارد شدن مشتری به سیستم (با نام کاربر و کلمهٔ رمز) یک شمارهٔ شناسایی بصورت کوکی روی کامپیوتر وی ذخیره می‌کند تا کاربر مجبور نباشد هر بار نام کاربر و کلمهٔ رمز خود را وارد کند. چه نظری

دربارهٔ این ایده دارید؟ آیا عملی است؟ آیا ایدهٔ خوبیست؟

۳۲. در شکل ۷-۲۶، پارامتر *ALT* برچسب <img> ست شده است. تحت چه شرایطی مرورگر از این پارامتر استفاده می‌کند (و چگونه)؟

۳۳. چگونه می‌توان یک تصویر را در HTML قابل‌کلیک کرد؟ مثالی بزنید.

۳۴. با استفاده از برچسب <a> برای کلمهٔ 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 اندازهٔ همبرگر (بزرگ یا بسیار بزرگ) و نوع پنیر آنرا وارد می‌کند. پول همبرگرها هنگام تحویل نقداً دریافت می‌شود، و امکان خرید با کارت اعتباری وجود ندارد.

۳۶. فرمی طراحی کنید که کاربر دو عدد را وارد کرده، و وقتی دکمهٔ Submit را کلیک کرد، سرویس‌دهنده جمع آنها را برگرداند. برای نوشتن اسکریپت سمت-سرویس‌دهنده از PHP استفاده کنید.

۳۷. برای هر یک از برنامه‌های 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 چندجمله‌ای با استفاده از ضرایبی که کاربر وارد می‌کند.

۳۸. برنامه‌ای 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 (شبیه C و جاوا) است. عملگر باقیمانده در جاوااسکریپت % است. اگر به جذر x نیاز داشتید، می‌توانید از تابع *Math.sqrt(x)* استفاده کنید.

۳۹. صفحهٔ HTML زیر را در نظر بگیرید:

```
<html> <body>
<a href="www.info-source.com/welcome.html">Click here for info</a>
</body> </html>
```

وقتی کاربر این لینک را کلیک می‌کند، یک اتصال TCP به سرویس‌دهنده برقرار شده و چند خط به آن فرستاده می‌شود. این خطها را بنویسید.

۴۰. از سرآیند *If-Modified-Since* می‌توان برای تعیین اعتبار صفحهٔ وب استفاده کرد. هر درخواست می‌تواند 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 می‌شود، تعداد مراجعات به سایت رسمی آن رویداد افزایش چشمگیری پیدا می‌کند. آیا این اتفاق شبیه انتخابات فلوریدا در سال ۲۰۰۰ است؟ توضیح دهید.

۴۲. آیا یک ISP منفرد می‌تواند بعنوان CDN عمل کند؟ اگر بله، چگونه؟ اگر خیر، چرا؟

۴۳. در چه شرایطی استفاده از CDN ایدهٔ مناسبی نیست؟

۴۴. ترمینالهای بیسیم وب پهنای باند کمی دارند، و بهمین دلیل کُد کردن مناسب اهمیت بیشتری می‌یابد. 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 ایدهٔ زبان ژاپنی، که در آن هر علامت معادل یک کلمه است، کمک بگیرید.

۴۵. یک دیسک فشردهٔ صوتی ظرفیتی معادل 650 MB دارد. آیا در این دیسکها از فشرده‌سازی استفاده می‌شود؟ توضیح دهید.

۴۶. در شکل ۷-۵۷ (ج)، بدلیل استفاده از نمونه‌های ۴ بیتی برای نمایش ۹ سیگنال نویز کوانتیزه کردن رُخ می‌دهد. اولین نمونه، در 0 ، دقیق است ولی چند نمونهٔ بعدی دقیق نیستند. درصد خطا در نقاط 1/32، 2/32 و 3/32 چقدر است؟

۴۷. آیا از مدل روان‌شنوایی می‌توان برای کاستن از پهنای باند موردنیاز تلفن اینترنتی استفاده کرد؟ اگر بله، تحت چه شرایطی؟ اگر خیر، چرا؟

۴۸. فاصلهٔ یک سرویس‌دهندهٔ صدای جویباری با پخش‌کننده معادل 50 msec ، و سرعت ارسال آن 1 Mbps است. اگر پخش‌کننده دارای بافری 1 MB باشد، دربارهٔ علامتهای حد-بالا و پائین چه می‌توان گفت؟

۴۹. مزیت الگوریتم یک‌درمیانی (شکل ۷-۶۰) آن است که در صورت گم شدن چند بسته خللی در پخش پیش نمی‌آید. 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 غیرفشردهٔ 600 × 800 با رنگ 8 bit و با سرعت 40 frames/sec چقدر است؟

۵۲. آیا ۱ بیت خطا در یک فریم MPEG می‌تواند روی چند فریم تأثیر بگذارد؟ توضیح دهید.

۵۳. یک سرویس‌دهندهٔ ویدئو با ۱۰۰,۰۰۰ مشتری را در نظر بگیرید، که هر مشتری در ماه دو فیلم تماشا می‌کند. نیمی از این فیلمها باید در ساعت ۸ بعد از ظهر پخش شود. این سرویس‌دهنده در هر لحظه (در همان ساعت) چند فیلم باید پخش کند؟ اگر هر فیلم به 4 Mbps پهنای باند نیاز داشته باشد، این سرویس‌دهنده به چند خط OC-12 نیاز دارد؟

۵۴. فرض کنید قانون زیف در مورد یک سرویس‌دهندهٔ ویدئو با ۱۰,۰۰۰ فیلم مصداق دارد. اگر این سرویس‌دهنده ۱۰۰۰ تا از فیلمهای پُربیننده‌تر را روی دیسک و ۹۰۰۰ فیلم دیگر را روی CD نگه دارد، چند درصد از درخواستها به دیسک مراجعه می‌کند؟ (عبارت آنرا بدست آورید.) برای ارزیابی عددی این فرمول، یک برنامهٔ کوچک بنویسید.

۵۵. برخی از متقلبان اینترنتی اقدام به ثبت نامهای اینترنتی نزدیک به نامهای معروف می‌کنند (مانند *www.microsfot.com* که فقط یک حرف آن با *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* که در آن *word* یکی از کلمات رایج است، را ثبت کرده‌اند. برای هر یک از دسته‌های زیر پنج سایت (بهمراه مختصری از مشخصات آنها) فهرست کنید: حیوانات، غذاها، لوازم خانگی، و اندامهای بدن.

۵۷. یک علامت دلخواه اموجی 12 × 12 طراحی کنید. برای مثال، سعی کنید کلمات پسر، دختر، معلم، و سیاستمدار را نمایش دهید.

۵۸. یک سرویس‌دهندهٔ POP3 بنویسید که فرمانهای زیر را قبول کند: *USER* ، *PASS* ، *LIST* ، *RETR* ، *DELE* ، و *QUIT* .

۵۹. سرویس‌دهندهٔ شکل ۶-۶ را با استفاده از فرمان HTTP 1.1 GET به یک سرویس‌دهندهٔ واقعی وب تبدیل کنید. این سرویس‌دهنده همچنین باید پیامهای *Host* را قبول کند. این سرویس‌دهنده باید یک حافظهٔ نهان داشته باشد، و فایلهایی را که اخیراً بازدید شده‌اند در این حافظهٔ نهان نگه دارد.

۸ امنیت شبکه

در چند دههٔ ابتدایی پیدایش ، از شبکه‌های کامپیوتری بیشتر توسط پژوهشگران دانشگاه و برای ارسال نامه‌های الکترونیکی و یا توسط کارمندان شرکتها برای به اشتراک‌گذاری چاپگر، استفاده می‌شد. در چنین شرایطی، امنیت شبکه از اهمیت چندانی برخوردار نبود. اما اکنون که میلیونها تن از شهروندان عادی از شبکه‌ها برای انجام عملیات بانکی، معاملات یا پر کردن اظهارنامه‌های مالیاتی خود استفاده می‌کنند، امنیت شبکه به عنوان یک مسئلهٔ بالقوه و عمده پدیدار شده است. در این فصل از چندین زاویه به مطالعهٔ امنیت شبکه خواهیم پرداخت، اشکالات و موانع امنیت را متذکر شده و چندین الگوریتم و پروتکل را که شبکه‌ها را امن‌تر و قابل‌اطمینان می‌کنند، تشریح خواهیم نمود.

«امنیت شبکه» در برگیرندهٔ عناوین و موارد بسیار گسترده است و بامشکلات و معضلات متعددی سروکار دارد. در یک عبارت ساده می‌توان امنیت را «اطمینان از عدم دسترسی افراد فضول و جلوگیری از دستکاری در پیامهای محرمانه دیگران» تعبیر کرد. همچنین می‌توان امنیت را در ارتباط با افرادی تعبیر کرد که تلاش می‌کنند به سرویسهای راه دور در شبکه دسترسی پیدا کنند در حالی که مجوز استفاده از آنها را ندارند؛ یا به روشهایی اطلاق می‌شود که بتوان صحت پیامهایی که مثلاً از ادارهٔ اخذ مالیات (IRS) می‌رسد و اعلام می‌کند: «حداکثر تا جمعه مبلغ اعلام شده را واریز کنید» را تائید کرد و تشخیص داد که این پیام واقعاً از ادارهٔ مالیات آمده نه از مافیا! همچنین می‌توان امنیت را در خصوص پیشگیری از دخل و تصرف ویا پاسخ جعلی به پیامهای قانونی دیگران و مقابله با افرادی که پس از ارسال پیام سعی در انکار آنها می‌کنند ، تعبیر کرد.

منشاء اغلب مشکلاتی که به صورت عمدی برای امنیت شبکه‌ها به وجود می‌آید 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 جرمهای امنیتی می‌شوند و انگیزه‌های آنان درج شده است. با بررسی این جدول روشن خواهد شد که تضمین امنیت شبکه، مقوله‌ای فراتر از رفع اشکالات برنامه‌نویسی است. در این خصوص باید تمهیداتی برای پیشگیری از حملهٔ دشمنانی اندیشیده شود که غالباً افرادی باهوش و کوشا هستند و گاهی اوقات سازمان‌یافته و با برنامه‌ریزی قبلی اقدام به حمله می‌کنند. البته همیشه عملیات مخرب بر علیه شبکه توسط یک گروه خاص و خطرناک انجام نمی‌شود؛ بررسی پرونده‌های ادارهٔ پلیس نشان می‌دهد که بسیاری از حملات بر علیه شبکه توسط عوامل خارجی و به واسطهٔ نفوذ از طریق خط تلفن نبوده است بلکه توسط عوامل مغرض داخلی انجام گرفته است. بنابراین طراحی سیستمهای امنیتی باید با در نظر داشتن این حقیقت انجام شود.

تهدیدکنندگان امنیت	هدف
دانشجو	تفریح کردن از طریق تجسس در نامه‌های دیگران
کِراکر (Cracker)	به منظور آزمایش سیستم امنیت متعلق به شخص (یا گروه) خاص یا سرقت اطلاعات
نمایندهٔ فروش	برای اطلاع از طرحهای استراتژیک حریف در خصوص بازار خرید و فروش
کارمند اخراجی	برای انتقام‌گیری
حسابداران	برای اختلاس پول از یک شرکت
دلال سهام	برای انکار وعده‌هایی که به یک مشتری داده شده است (از طریق email)
کلاه‌بردار	برای سرقت شماره‌های 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 می‌شوند و انگیزه‌های آنها.

مشکلات امنیت شبکه بطور کلی به چهار ردهٔ نزدیک و مرتبط به هم تقسیم‌بندی می‌شوند: (۱) **سرّی ماندن اطلاعات**[۱]، (۲) **احراز هویت کاربران**[۲]، (۳) **غیرقابل انکار بودن پیامها**[۳]، (۴) **نظارت بر صحّت اطلاعات**[۴].

سری ماندن اطلاعات که گاه «محرمانه نگاهداری اطلاعات» (Confidentiality) نیز نامیده می‌شود، متضمن انجام عملیاتی است که اطلاعات را از دسترس کاربران غیرمجاز و بیگانه دور نگاه می‌دارد. این همان مفهومی است که در ذهن مردم عادی در خصوص امنیت شبکه تداعی می‌شود.

«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.

مقولهٔ «غیرقابل انکار بودن پیامها» (Nonrepudiation) با امضاهای دیجیتالی سر و کار دارد و به اطلاعات و مستندات، هویت حقوقی اعطاء می‌کند: وقتی مشتری شما که قبلاً به صورت الکترونیک یک میلیون عدد از یک کالای کوچک به قیمت ۸۹ سنت سفارش داده و بعداً ادعا می‌کند قیمت کالا ۶۹ سنت بوده، چگونه می‌توان خلاف ادعای او را ثابت کرد؟ شاید حتی او ادعا کند هرگز چنین سفارش خریدی نداده است.

نهایتاً چگونه می‌توان مطمئن شد که پیامی که شما دریافت کرده‌اید، 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 فوق‌الذکر (سری ماندن 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 انکاء به پست سفارشی و لاک و مهر کردن مستندات انجام می‌شود.

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. حالا، تلاش کنید تصویر کپی شدهٔ چک را در روز سه‌شنبه نقد کنید تا تفاوت بین رفتاری که با شما می‌شود 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.

۱. Secracy ۲. Authentication ۳. Nonrepudiation ۴. Integrity Control

در دنیای پیرامون ما، افراد از طریق تشخیص چهرهٔ یک فرد، صدا یا دستخط ، او را احراز هویت می‌کنند. در عملیات اداری، تأئید 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 مقایسهٔ 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 پشتهٔ پروتکلی شبکه (Protocol Stack) ممکن و قابل اِعمال است، خالی از لطف نخواهد بود.

رعایت نکات و تمهیدات امنیتی فقط در یک نقطهٔ خاص متمرکز نیست. در هر لایه از معماری شبکه، باید نکات و موارد امنیتی مدّنظر قرار گرفته و بدقت رعایت شود: در لایهٔ فیزیکی (Physical Layer) برای جلوگیری از ایجاد انشعاب در سیم و پیشگیری از استراق سمع سیگنال (Wire tapping) ، می‌توان سیمها را در درون یک لولهٔ محافظ جاسازی و لوله را با یک گاز تحت فشار پر کرد. در این حالت هر گونه تلاش در نقب زدن به این لوله و سوراخ کردن آن منجر به تخلیهٔ 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 لایهٔ پیوند داده‌ها (Data Link)، بسته‌های ارسالی بر روی خطوط نقطه به نقطه (Point To Point) قبل از خروج از ماشین مبداء ، رمزنگاری شده و به محض ورود به ماشین مقصد رمزگشایی می‌شود. تمام جزئیات کار در سطح لایهٔ 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 مسیریاب باشند کارآیی خود را از دست خواهد داد زیرا اطلاعات در بدو ورود به هر مسیریاب رمزگشایی شده و بمحض خروج از آن مجدداً رمزنگاری می‌شوند فلذا این خطر وجود دارد که در یکی از مسیریابهای 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 مسیریابهای واقع برروی 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 لایهٔ 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 لایهٔ انتقال (Transport Layer) کل اتصال (Connection) بین مبداء و مقصد، می‌تواند رمزنگاری شود. به عبارت بهتر تمام داده‌های در حال تبادل، در پروسهٔ مبداء رمز شده و در پروسهٔ مقصد از رمز خارج خواهد شد؛ (به این روش امنیت انتها به انتها یا End-to-End گفته می‌شود). برای تضمین حداکثر امنیت، اِعمال «**امنیت انتهابه‌انتها**» الزامی است.

در آخر، مواردی نظیر احراز هویت کاربران و غیرقابل انکار بودن محتوای پیامها (Nonrepudiation) فقط در لایهٔ کاربرد قابل اعمال و پیاده‌سازی است.

از آنجایی که تضمین امنیت اطلاعات دقیقاً در یک لایهٔ خاص قرار نمی‌گیرد لذا نمی‌شد مفاد آنرا در ادامهٔ هیچیک از فصول این کتاب گنجاند؛ 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 می‌دهد که بسیاری از شکستها در حریم امنیتی شبکه‌هایی مثل بانکها، بیشتر ناشی از عواملی نظیر بی‌لیاقتی و سهل‌انگاری کارمندان، تمهیدات و روالهای امنیتی ناکافی، تقلب و کلاه‌برداریهای داخلی بوده است تا آنکه از یک نقشه جنایتکارانهٔ 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 می‌کند براحتی بتواند به یکی از شعب مربوطه مراجعه نماید و با اعلام آنکه شمارهٔ PIN (شمارهٔ رمز شخصی) را فراموش کرده در همان شعبه شمارهٔ 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 می‌دهد که در صدها نمونه از شکستهای امنیتی و نقض حریم شبکه‌ها در صنایع و سازمانهای متعدد، تقریباً تمام آنها (در یک عبارت مودبانه) ناشی از سهل‌انگاری در عملکرد و بی‌دقتی 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 جنبه‌های فنی «امنیت» را سرلوحه کار خود قرار بدهند.

به غیر از امنیت در سطح لایه فیزیکی (که به صورت مکانیکی و از طریق لوله‌های محتوی گاز تحت فشار انجام می‌شود) در بقیهٔ لایه‌ها 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 ۸ـ۱ نگاهی بر اصول اولیهٔ آن خواهیم داشت. در بخش ۸ـ۲ تا بخش ۸ـ۵ به برخی از الگوریتمهای اساسی و ساختمان داده‌های مورد استفاده در رمزنگاری خواهیم پرداخت. سپس به صورت مبسوط تشریح خواهیم کرد که چگونه این مفاهیم برای تأمین امنیت در شبکه‌های کامپیوتری بکار گرفته می‌شوند. نهایتاً فصل را با خلاصه‌ای از نظریات در خصوص «تکنولوژی و جامعه» (Technology & Society) جمع‌بندی خواهیم کرد.

قبل از شروع، به مواردی که در این فصل بدانها نخواهیم پرداخت اشاره می‌کنیم. سعی کرده‌ایم 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 برنامه‌های کاربردی مربوط می‌شود ، بررسی نکرده‌ایم. به عنوان مثال در این فصل مطلبی درخصوص احراز هویت کاربران به روش بیومتریک، استراتژیهای امنیت کلمهٔ عبور، **«حملهٔ 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» (تاننبام،۲۰۰۱)بررسی شده‌اند. علاقمندان می‌توانند برای مطالعه بر روی این جوانب از امنیت، به کتاب فوق مراجعه نمایند. حال بیائید خط سیر خود را آغاز کنیم.

۸ـ۱ رمزنگاری

کلمهٔ Cryptography (رمزنگاری) برگرفته از لغات یونانی به معنای «محر .'- نوشتن متون» است. رمزنگاری پیشینه‌ای طولانی و درخشان دارد که به هزاران سال قبل بر می‌گردد. در این بخش برخی نکات برجسته از تاریخ رمزنگاری را بعنوان اطلاعات عمومی (که دانستن آنها برای ادامهٔ بخشهای بعد مفید خواهد بود)، بررسی خواهیم کرد. برای آشنایی با پیشینهٔ دقیق و کامل رمزنگاری، کتاب Kahn (۱۹۹۵) جهت مطالعه توصیه می‌شود؛ برای تحلیل جامع مفاهیم مدرن و پیشرفته در امنیت و آشنایی با الگوریتمهای رمزنگاری، پروتکلها و برنامه‌های 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، «کد» تبدیلی است که کلمه‌ای را با یک کلمه یا علامت (سمبول) دیگر جایگزین می‌کند. امروزه از کدها استفادهٔ چندانی نمی‌شود اگرچه استفاده از آن پیشینهٔ طولانی و پر سابقه‌ای دارد. موفقترین «کد»هایی که تاکنون ابداع شده‌اند توسط ارتش ایالات متحده و در خلال جنگ جهانی دوم در اقیانوس آرام بکار گرفته شد. آنها از لهجهٔ 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-tsaidi (در زبان محلی سرخپوستان به معنای کُشنده‌لاک‌پشت!) 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.

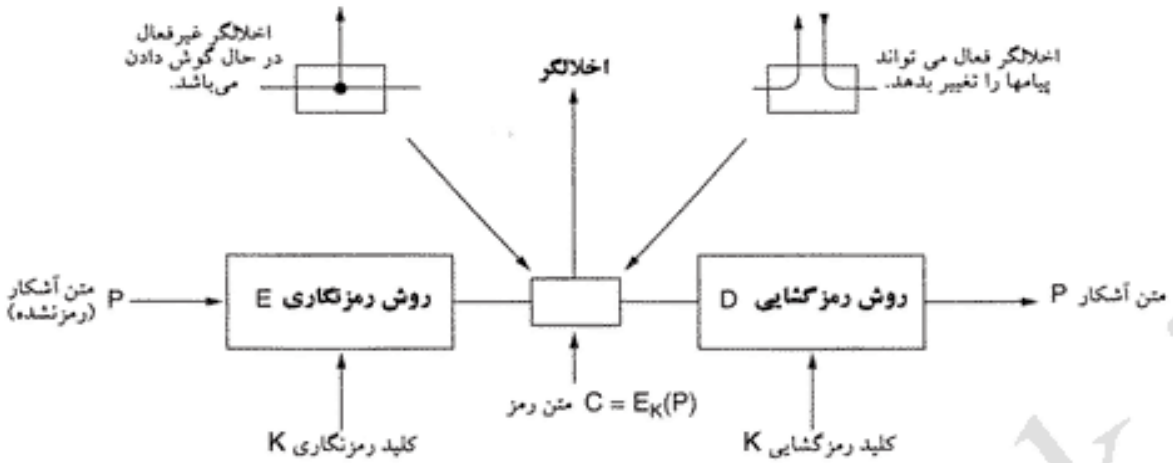
در سپتامبر ۱۹۴۵ در مجمع متفقین در سَن‌دیه‌گو این «کُد» با بیان گزارش ذیل توصیف شد: «برای سه سال متوالی، هر گاه ناوگان دریایی در خشکی پهلو می‌گرفت، آنچه که جاسوسان ژاپنی (از بی‌سیم) استراق سمع می‌کردند یک صدای نامفهوم و شلوغ بود که با دیگر اصوات درهم آمیخته و در نتیجه صدایی شبیه به لحن راهبان تبت یا صدای 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 مبادلهٔ پیامهای سرّی 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 کُدهای رمز ژاپنی‌ها را بشکند در حالی که ژاپنی‌ها نتوانستند کُد Navajo را بشکنند و این حقیقت، نقش بسیار مهمی در پیروزیهای آمریکاییان در جنگ اقیانوس آرام (با نیروی دریائی 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 «عشّاق!». از بین اینها نظامیان نقش بسیار مهمتری دارند و در طول قرنها به تکوین این شاخه از علم پرداخته‌اند. سابقاً در مؤسسات نظامی، پیامهائی که باید رمزنگاری می‌شدند به یک کارمند (منشی) دون پایه و حقوق‌بگیر تحویل می‌شد تا آنها را رمز و ارسال کند. حجم عظیم پیامهائی که در طی یک روز باید رمز و ارسال می‌شد مانع از آن بود که بتوان این کار خطیر را بر عهدهٔ معدود متخصصین خبرهٔ حاضر در یک مؤسسه گذاشت.

تا زمان ابداع کامپیوترها، در عرصهٔ یک جنگ واقعی و با تجهیزات اندک، بزرگترین نقطهٔ 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. نتیجهٔ فرآیند رمزنگاری که «**متن رمز**» (Ciphertext) نامیده می‌شود بر روی کانال منتقل خواهد شد. فرض کنیم که دشمن یا اخلالگر (Intruder) متن رمز شده را به صورت کامل



شکل ۸-۲. مدل رمزنگاری (برای روشهای رمزنگاری با کلید متقارن).

می‌شنود و آن را در اختیار می‌گیرد. به هر حال او برخلاف گیرندهٔ اصلی، براحتی قادر به رمزگشایی پیام و بهره‌برداری از آن نخواهد بود زیرا کلید رمز را نمی‌داند. برخی اوقات یک اخلالگر غیرفعال (Passive Intruder) نه تنها قادر است به جریان اطلاعات بر روی کانال مخابراتی گوش بدهد بلکه می‌تواند آنها را در جایی ثبت کرده و بعداً آن را بارها به جریان بیندازد؛ در مقابل یک اخلالگر فعّال (Active Intruder) می‌تواند 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**» (Cryptoanalysis) نام دارد؛ به هنر ابداع روشهای رمزنگاری جدید «**علم رمزنگاری**» (Cryptology) اطلاق می‌شود.

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 رمزشدهٔ C بدست آمده است. به روش مشابه، فرمول $P=D_K(C)$ عمل رمزگشایی متن رمزشده توسط کلید K را (به منظور استخراج اصل پیام) توصیف می‌کند. بنابراین داریم:

$$D_K(E_K(P))=P$$

این نماد بیانگر آن است که E و D توابع ریاضی و معکوس یکدیگر هستند. تنها نکتهٔ 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 داده‌ایم 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 می‌دانند. به عبارت دیگر شخص رمزشکن، روش رمزنگاری یعنی تابع E و روش رمزگشایی یعنی تابع D در شکل ۸-۲ را می‌داند و آنچه از او پنهان نگاه داشته می‌شود 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 می‌تواند سری و مخفی بماند (در حالی که ممکن نیست) خطرات و زیانهائی بیشتر از منافع آن دارد.

اینجاست که «**کلید رمز**» وارد قضیه می‌شود. «کلید رمز» یک رشتهٔ کاراکتری نسبتاً کوتاه است که پیام براساس آن رمز می‌شود. برخلاف آن که روش رمزنگاری ممکن است هر چند سال یکبار تغییر کند، کلید رمز می‌تواند 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 کِرکهُف**» نامیده می‌شود که توسط یکی از رمزنگاران ارتش فلاندرز به نام Auguste Kerckhoff در سال ۱۸۸۳ بیان شده است. بنابراین داریم:

***اصل کِرکُهُف:* تمام الگوریتمهای رمزنگاری باید آشکار و عمومی باشند و تنها کلیدهای رمز، مخفی و محرمانه هستند.**

شاید نتوان حقّ 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 سایهٔ گمنامی و ابهام**» (Security by Obscurity) مشهور است، هرگز محقق نخواهد شد. با عمومی‌سازی 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! هر گاه 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. قاعدهٔ 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.[1] 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 شِق اساسی را در بر می‌گیرد: (۱) هر گاه 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**»[2] مواجه است. (۲) وقتی رمزشکن بخشی از متن آشکار را به همراه معادل رمز شدهٔ آن ، در اختیار دارد، اصطلاحاً با مسئله «**متن آشکار و شناخته شده**»[3] مواجه است. (۳) نهایتاً هر گاه رمزشکن قادر باشد هر قسمت دلخواه از یک متن آشکار را رمز کند اصطلاحاً با مسئله «**متن آشکار و انتخابی**»[4] مواجه است. هرگاه به یک رمزشکن اجازه داده شود تا بپرسد 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. Ciphertext Only ۳. Known Plaintext ۴. Chosen Plaintext

خواهد بود تا رمزهای معمولی را بشکنند و کلید رمز را بدست بیاورد.

نوآموزان حرفهٔ رمزنگاری به اشتباه می‌اندیشند که هرگاه یک متن رمز شده بتواند در مقابل حملهٔ نوع «صرفاً متن رمز شده» استقامت کند و فاش نشود، آن سیستم رمز مطمئن است.[۱] 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.[۲] به عنوان مثال اولین جمله‌ای که در حین برقراری ارتباط شخص از راه دور [مثلاً با یک سرویس‌دهنده TelNet] ، ارسال می‌شود معمولاً کلمهٔ login: (به صورت رمز شده) است. حال رمزشکن پاره‌ای متن رمز شده به همراه معادل رمز نشدهٔ آن را در اختیار دارد؛ کار او نسبتاً ساده و سرراست است. برای رسیدن به امنیت کامل، طراح الگوریتم رمزنگاری باید محافظه‌کار بوده و مطمئن شود که سیستم رمز او بگونه‌ای است که حتی در صورتی که حریف رمزشکن ، معادل رمز شدهٔ یک متن آشکار را در اختیار داشته باشد باز هم قادر به شکستن رمز و بدست آوردن کلید رمز نیست.[۳]

روشهای رمزنگاری بطور کلی به دو رده تقسیم می‌شوند: (۱)رمزهای جانشینی (Substitution) (۲)رمزهای جایگشتی (Transposition) . در اینجا به عنوان مقدمه‌ای بر روشهای مدرن رمزنگاری، مختصراً به این دو روش خواهیم پرداخت.

۸-۱-۲ رمزهای جانشینی (Substitution Cipher)

در رمزنگاری جانشینی هر حرف یا گروهی از حروف با یک حرف یا گروهی دیگر از حروف جابجا می‌شوند تا شکل پیام بهم بریزد. یکی از قدیمی‌ترین رمزهای شناخته شده روش رمزنگاری سزار است که ابداع آن به ژولیوس سزار نسبت داده می‌شود. در این روش حرف a به D تبدیل می‌شود، b به E ، c به F و به همین ترتیب تا z که با C جایگزین می‌گردد. به عنوان مثال در این روش، کلمهٔ attack به DWWDFN تبدیل می‌شود. 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 جابجا 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 می‌شود. روش رمزنگاری سزار شاید در آن زمانها کسی را فریفته باشد ولی امروزه نمی‌تواند 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 می‌توان برای رمزنگاری جانشینی استفاده کرد:

متن آشکار a b c d e f g h i j k l m n o p q r s t u v w x y z

متن رمز شده Q W E R T Y U I O P A S D F G H J K L Z X C V B N M

هر سیستم رمزنگاری که در آن یک سمبول با سمبول دیگر جایگزین می‌شود اصطلاحاً «**سیستم جانشینی**

---

۱. بدین خیال که چون رمزشکن فقط متن رمز شده را در اختیار دارد و چیزی از متن اصلی نمی‌داند 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 رمزشده چیزی نمی‌داند ولی رمزشکن قادر است بخشهای کوچکی از یک متن را حدس بزند. م

۳. بعبارت روشنتر سیستم رمز باید بنحوی طراحی شود که که حتی اگر یک متن رمزشده و معادل رمزنشدهٔ آنرا به رمزشکن بدهید بازهم نتواند کلید رمز شما را پیدا کند. -م

تک‌حرفی» (Monoalphabetic Substitution) گفته می‌شود که در آن کلید رمز یک رشتهٔ ۲۶ کاراکتری است و نگاشت جدول الفباء را مشخص می‌نماید. بر اساس کلید رمزِ مثال بالا، کلمهٔ attack به QZZQEA تبدیل خواهد شد.

در نگاه اول این سیستم رمزنگار مطمئن به نظر می‌رسد زیرا اگرچه رمزشکن روش عمومی جانشینی حروف را می‌داند ولی نمی‌داند از بین !26 حالت مختلف (معادل با ۴×۱۰^{۲۶} حالت) کدامیک کلید رمز است. برخلاف رمز سزار، آزمایش تمام حالات مختلف کلید غیرممکن است زیرا اگر هر یک از حالات کلمهٔ رمز در یک نانوثانیه آزمایش شود، بررسی تمام حالات کلید توسط چنین کامپیوتری $۱۰^{۱۰}$ سال طول خواهد کشید.

در روش فوق علیرغم آنکه آزمایش تمام حالات یک کلید ممکن نیست ولی حتی برای یک قطعه متن رمز شدهٔ کوچک، رمز متن به سادگی شکسته خواهد شد. در حملهٔ 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 ***o***، ***a***، ***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 (رمزشکن) برای شکستن سیستم رمزنگاری «جانشینی تک‌حرفی» (Monoalphabetic) با شمارش حروف متن رمز شده و محاسبهٔ 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 بقیهٔ یک‌حرفیها، دوحرفیها و سه‌حرفیهای تکراری و با شناخت از حروف صدادار و بی‌صدا و چگونگی ترکیب آنها در کلمه، رمزشکن 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 شدهٔ 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 کلمهٔ «financial» در پیامهای یک شرکت حسابرسی زیاد است. با استفاده از این ویژگی که کلمهٔ financial دارای دو حرف «***i***» با فاصلهٔ چهار حرف از یکدیگر است، در متن رمزشده به جستجوی حروف مشابه با فاصلهٔ 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 کلمهٔ financial) با فاصلهٔ یک حرف تکرار شده و مطابقت دارد. از بین این دو مورد نیز فقط مورد ۳۱ است که با تکرار حرف ***a*** با فاصلهٔ سه حرف مطابقت دارد؛ لذا می‌توان متوجه‌شد که در این متن کلمهٔ financial در موقعیت ۳۰ از متن رمز شده قرار گرفته است. پیدا شدن این کلمه، نقطهٔ شروع خوبی برای پیدا کردن کلید رمز با کمک ویژگیهای آماری حروف در زبان انگلیسی خواهد بود.

۸-۱-۳ رمزنگاری جایگشتی (Transposition)

رمزنگاری جانشینی ترتیب سمبولهای یک متن را حفظ می‌کند ولی (صرفاً) شکل سمبولها را تغییر می‌دهد. برعکس، «رمزنگاری جایگشتی» ترتیب حروف متن را بهم می‌ریزد ولیکن شکل آنها را تغییر **نخواهد داد**. در شکل ۸-۳ یک روش عمومی از رمزنگاری جایگشتی که در آن ترتیب سمبولها بصورت ستونی بهم ریخته می‌شود نشان داده شده است. کلید رمز یک کلمه یا عبارت [انگلیسی] است که هیچ حرف تکراری ندارد. در این مثال کلید رمز کلمهٔ MEGABUCK انتخاب شده است. کاربرد کلید رمز آنست که ستونها شماره‌گذاری شود. شمارهٔ هر ستون براساس ترتیب الفبایی هر حرف کلید نسبت به جدول الفبای انگلیسی تعیین می‌شود. به عنوان مثال ستون چهارم شمارهٔ ۱ است (حرف A)، ستون پنجم شماره ۲ (حرف B)، ستون هفتم شمارهٔ ۳ (حرف C) و ستون دوم شمارهٔ ۴ (حرف E) و به همین ترتیب. متن اصلی به صورت افقی (سطری) در ذیل کلید رمز نوشته می‌شود. سپس در صورت لزوم تعدادی حرف به آخرین سطر اضافه می‌شود تا ماتریس مربوطه پر شود. متن رمز شده بر اساس شمارهٔ ستونها به صورت عمودی خوانده شده و به هم متصل می‌شود. 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
myswissbankaccountsixtwotwo

متن رمزشده

AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUOERIRICXB

شکل ۸-۳. 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 E، T، A، O، I، N و نظائر آن بسادگی می‌توان مطمئن شد که متن الگوئی شبیه به یک متن معمولی رمزنشده دارد. در این صورت روشن است که متن رمز شده از نوع جایگشتی است زیرا در این روش شکل هر حرف تغییر نمی‌کند بلکه فقط جای آن در متن عوض می‌شود 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 می‌توان برخی از کلمات یا عبارات یک متن را حدس زد. به عنوان مثال فرض کنید رمزشکن ما به وجود عبارت milliondollars در جایی از متن اصلی مشکوک است. دقت کنید که دو حرفی‌های MO، IL، LL، LA، IR و OS در متن رمز، دیده می‌شود 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 گرفته‌اند.) «حروفی که در متن رمز شده کنار هم قرار می‌گیرند در متن اصلی به اندازه «طول کلید» از هم فاصله دارند». اگر طول کلید به جای ۸، هفت در نظر گرفته می‌شد، پس از رمز شدن عبارت milliondollars ، دو حرفی‌های MD، IO، LL، LL، IA، OR و NS بوجود می‌آمدند. در حقیقت بسته به طول کلید مجموعهٔ متفاوتی از دو حرفی‌ها در متن رمز شده ظاهر می‌شود. تحلیل‌گر رمز می‌تواند 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 می‌توان تمام $k\times(k-1)$ حالت مختلف زوج ستونها را آزمود تا ببینیم آیا میزان تکرار دو حرفیها و سه حرفیهای این دو ستون با شرایط آماری یک متن معمولی مطابقت دارد یا خیر؟ حالتی از ترکیب ستونها که براساس آن ترکیب، نتیجهٔ رمزگشایی بیشترین تطبیق را با متون معمولی داشته باشد به عنوان زوج ستون متوالی در نظر گرفته می‌شود. حال یکایک ستونها به عنوان ستونهای بعدی این زوج رمزگشایی شده، آزمایش و تحلیلهای آماری بر روی آنها انجام می‌شود و بهترین ستون به عنوان ستون بعدی آن در نظر گرفته می‌شود و این کار ادامه می‌یابد. ستون قبلی یک ستون نیز به همین روش (تحلیل دو حرفی و سه حرفیها) امکان‌پذیر است. این فرآیند آنقدر تکرار می‌شود تا ترکیب درست و مورد تائید به دست بیاید. عمل رمزشکنی در نقطه‌ای با اقبال و موفقیت روبرو خواهد شد که یک کلمه یا یک عبارت ظاهر گردد. (به عنوان مثال هر گاه در آزمون یک ترکیب خاص از ستونها، کلمهٔ million ظاهر شود می‌توان 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 می‌کنند. در این گونه از روشها فهرست کامل جایگشتهای ورودی (که متن رمز شدهٔ خروجی را تولید می‌کند) مشخص است. به عنوان مثال سیستم رمز شکل ۸-۳ را می‌توان در قالب یک رمزنگار با بلوکهای ورودی ۶۴بایتی 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.

۸-۱-٤ رمز One-Time Pads (بهم ریزی محتوی پیام)

ایجاد یک سیستم رمزنگاری که غیرقابل شکستن باشد حقیقتاً کار ساده‌ایست 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 نمائید؛ (مثلاً با الحاق بیتهای کد اَسکی هر کاراکتر) نهایتاً این دو رشته را بیت به بیت با یکدیگر XOR (Exclusive OR) کنید. رشته بیت حاصل، متن رمز شدهٔ 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 دوحرفیها و سه‌حرفیها (Digram/Trigram) در متن رمز، مشابه خواهد بود.[1] 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 «**تئوری اطلاعات**» (Information Theory) استنتاج می‌شود: 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 می‌شود. ابتدا پیام ۱ (جملهٔ "I Love You") کاراکتر به کاراکتر به کدهای اسکی هفت بیتی تبدیل می‌شوند. سپس یک کلید رمز تصادفی [که از این به بعد آن را Pad می‌نامیم] انتخاب و با پیام XOR می‌شود تا متن رمز شده بدست آید. یک رمزشکن باید تمام حالات مختلف رشتهٔ Pad را آزمایش کند تا ببیند به ازای هر Pad چه متنی حاصل می‌شود که البته باز هم موفق به یافتن متن اصلی نخواهد شد زیرا به عنوان مثال اگر Pad شمارهٔ ۲ با پیام اول XOR شود رشته‌ای را حاصل خواهد

---

۱. به عبارت فنی و دقیق‌تر 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 Padها در متن دلخواهی را از این رشتهٔ 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 فرستندهٔ پیام، باید آنرا به صورت نوشته شده با خود حمل کنند. اگر یکی از این دو طرف در معرض حملهٔ 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 رمزگشایی‌شده غیرقابل 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 خاصّ حاوی چندین گیگابایت اطلاعات، ذخیره گردد؛ اگر کلید رمز در پوشش چند دقیقه فیلم ویدیویی بر روی DVD جاسازی شود شک‌برانگیز 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 DVDهای حامل کلیدهای رمز باید توسط شخص فرستندهٔ اطلاعات و قبل از ارسال داده‌ها، به گیرنده تسلیم شود، که این کار بشدت از عملی بودن روش خواهد کاست.[۱]

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 کارآیی 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. -م

روش کاملاً امن و غیرقابل شکستن است. در زیر بطور مختصر توضیح خواهیم داد که روش رمزنگاری کوآنتومی چگونه کار می‌کند. بطور خاص پروتکل BB84 را بررسی خواهیم کرد. (BB84 ابتدای نام ابداع‌کنندگان و سال انتشار مقالهٔ آنهاست ـ Bennet and Brassard ، ۱۹۸۴)

یک کاربر مثل آلیس مایل است با کاربر دوم (مثلاً باب) یک Pad طولانی ایجاد و از آن به عنوان کلید رمز استفاده کند. آلیس و باب که اصطلاحاً طرفین اصلی ـ Principals ـ نامیده می‌شوند بازیگران اصلی داستان ما هستند. به عنوان مثال باب یک بانکدار و آلیس یکی از مشتریان بانک است که می‌خواهد با باب مراودات تجاری داشته باشد. در چند دههٔ گذشته در مقالات و کتب رمزنگاری، اسامی «آلیس» و «باب» عموماً به عنوان طرفین مُجاز 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 می‌کردیم هیچکس مطالب این فصل را باور نمی‌کرد! پس بگذارید ما هم از این سنّت تبعیت کنیم.

اگر آلیس و باب بتوانند یک Pad طولانی به عنوان کلید رمز، ایجاد کنند، قادرند با استفاده از آن ، داده‌ها را به صورت مطمئن و امن مبادله نمایند. سؤال آن است که آنها چگونه می‌توانند بدون ردّ و بدل کردن فیزیکی کلید رمز (مثلاً با DVD)، Padها را ایجاد و مبادله کنند؟ می‌توانیم فرض کنیم که آلیس و باب در دو سمت یک فیبرنوری قرار گرفته‌اند و می‌توانند پالسهای نوری را ارسال یا دریافت نمایند ولیکن به فرض یک متجاوز به نام ترودی توانسته فیبرنوری را قطع کرده و در آن یک انشعاب فعال ایجاد کند. بدین ترتیب ترودی قادر است تمام بیتها را در دو جهت استراق سمع نماید. او همچنین می‌تواند پیامهای غلط برای دو طرف ارسال کند. این شرایط اگرچه برای باب و آلیس ناامیدکننده و خطرناک به نظر می‌رسد ولیکن رمزنگاری کوآنتومی می‌تواند روزنهٔ امیدی برای حل این مشکل باشد.

رمزنگاری کوآنتومی بر این اصل استوار است که نور در قالب بسته‌های کوچکی به نام فوتون با ویژگیهای خاصّ و عجیب، جابجا می‌شود. به علاوه وقتی نور از یک فیلتر پلاریزه‌کننده عبور می‌کند، پلاریزه (قطبی) می‌شود. نور پلاریزه شده برای عکاسان یا آنهایی که عینک آفتابی می‌زنند 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 می‌شوند. حال اگر این پرتو مجدداً از فیلتر پلاریزه‌کنندهٔ دوم عبور نماید «شدت نور» پرتوی خروجی، متناسب با مربع کسینوس زاویهٔ 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 زاویهٔ مطلق دو فیلتر در فضای سه بعدی اصلاً مهم نیست بلکه فقط زاویهٔ نسبی محورهای دو فیلتر پلاریزه کننده در محاسبه وارد می‌شود.

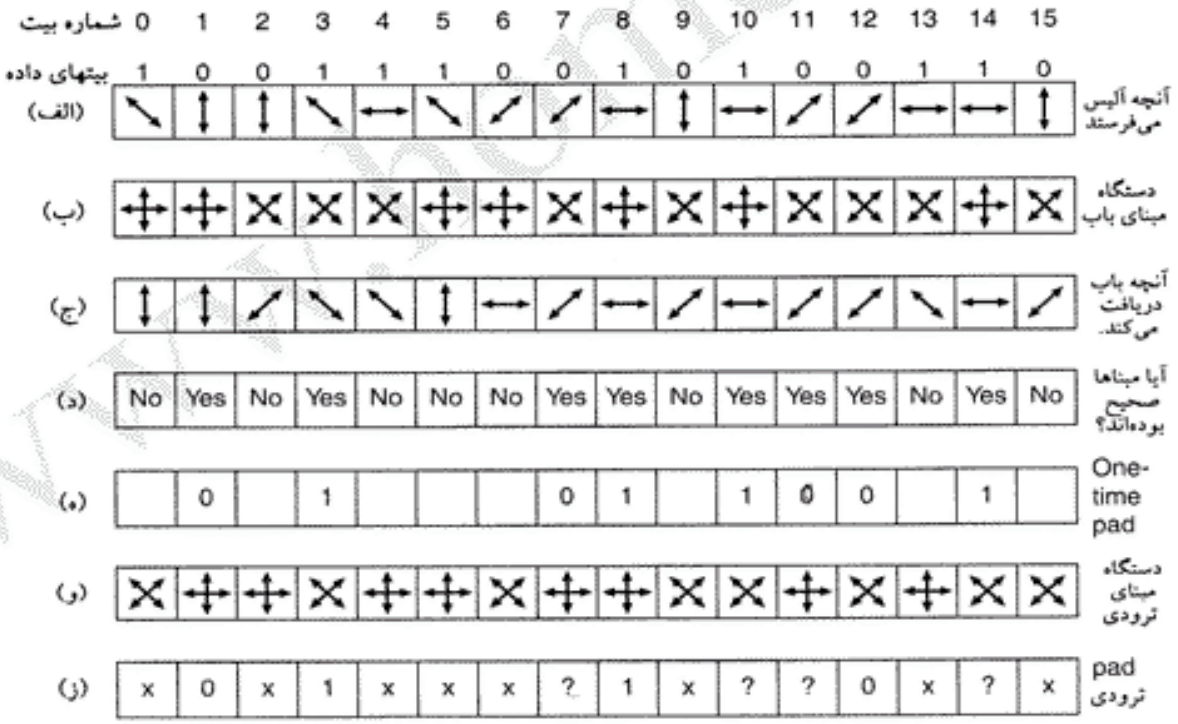
برای تولید یک Pad به عنوان کلید رمز، آلیس نیاز به تنظیم دو زوج فیلتر پلاریزه کننده دارد. زوج فیلتر اول، شامل یک فیلتر عمودی و یک فیلتر افقی است. این انتخاب اصطلاحاً «**دستگاه مستقیم**» (Rectilinear Basis) نامیده می‌شود. زوج فیلتر دوم مشابه با قبلی است با این تفاوت که ۴۵ درجه چرخیده است یعنی یکی از فیلترها در امتداد قطر گوشهٔ پایین سمت چپ به گوشهٔ بالا سمت راست قرار گرفته و دیگری عمود بر آن یعنی در امتداد گوشهٔ پایین سمت راست به گوشه بالا سمت چپ قرار دارد. زوج فیلتر دوم اصطلاحاً «**دستگاه مورب**» (Diagonal Basis) نامیده می‌شود. بدین ترتیب آلیس دارای دو دستگاه مبنا است که می‌تواند 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 آلیس دارای چهار فیلتر پلاریزه‌کنندهٔ مجزا نیست بلکه فقط یک قطعه بلور خاص (کریستال پلاریزه کننده) وجود دارد که می‌تواند با سرعت بسیار بالا به هر یک از فیلترهای مورد نظر سوئیچ کند و زاویهٔ پلاریزاسیون را عوض

نماید.[1] در سمت مقابل، باب نیز از چنین ابزاری استفاده می‌کند. 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 می‌کند. در مثالی که در ادامه مطرح کرده‌ایم 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 می‌فرستد.

حال آلیس یک Pad برای خود انتخاب می‌کند که این انتخاب می‌تواند توسط یک مولد اعداد تصادفی انجام شود (موضوعی که به خودی خود مقوله‌ای پیچیده محسوب می‌شود). او این Pad را بیت به بیت برای باب می‌فرستد در حالی که برای ارسال هر بیت بطور تصادفی از یکی از دستگاههای مبنا استفاده می‌کند.[2]

برای ارسال یک بیت، تفنگ تولید فوتون قادر است فوتونی را منتشر کند که پلاریزاسیون آن کاملاً منطبق با دستگاه مبنای آلیس و به اختیار او باشد. به عنوان مثال او می‌تواند بطور متوالی دستگاه مبنای پلاریزاسیون را تغییر داده و دستگاه مورب (Diagonal)، دستگاه مستقیم (Rectilinear) را انتخاب نماید. مثلاً آلسی می‌تواند برای ارسال یک Pad نظیر 1001110010100110 ، طبق قرارداد فوق فوتونهای پلاریزه شدهٔ 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 می‌شود. «بیتهایی که در قالب یک فوتون منفرد و در یک زمان مشخص ارسال می‌شوند اصطلاحاً کوبیت(ها) (Qubit(s)) نامیده می‌شوند.»



شکل ۸-۵. مثالی از رمزنگاری کوآنتومی.

---

۱. به عبارت بهتر یک کریستال به صورت الکترونیکی به گونه‌ای تنظیم می‌شود که رفتار هر یک از فیلترهای مورد نظر را بروز بدهد. امروزه کریستالهای پلاریزه کنندهٔ 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 می‌شود. -م.

باب نمی‌داند که از چه دستگاه مبنا [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 می‌گیرد، مثلاً به گونه‌ای که در شکل ۸-۵-ب مشاهده می‌کنید. 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 نظریهٔ مکانیک کوآنتومی هر گاه یک فوتون به یک فیلتر پلاریزه کننده با اختلاف زاویهٔ ۴۵ درجه (نسبت به زاویهٔ پلاریزاسیون خود فوتون) برخورد کند بطور تصادفی و با احتمال مساوی، یا به زاویه پلاریزاسیون فیلتر و یا به زاویهٔ قائم بر پلاریزاسیون فیلتر، پرش خواهد کرد. این ویژگی فوتونها در مکانیک کوآنتومی یکی از اصول اساسی به شمار می‌آید. بدین ترتیب هر گاه دستگاه مبنای انتخاب شده توسط باب صحیح نباشد برخی از بیتهای دریافتی صحیح و برخی دیگر اشتباه هستند ولیکن باب نمی‌داند که کدام صحیح و کدام غلط هستند. نتیجهٔ دریافت بیتها توسط باب در شکل ۸-۵ نشان داده شده است.

باب چگونه می‌تواند متوجه شود که کدام مبنا درست و کدام غلط بوده است؟ او به سادگی و به صورت آشکار به آلیس اعلام می‌کند که دستگاههای مبنای انتخابی او برای هر بیت چه بوده‌اند و آلیس هم در پاسخ به او خواهد گفت که کدام صحیح و کدام یک غلط است؛ (به گونه‌ای که در شکل ۸-۵-د ملاحظه می‌کنید). با این اطلاعات طرفین می‌توانند یک رشته بیت، مطابق با شکل ۸-۵-ه از حدسهای درست بسازند. بطور میانگین این رشتهٔ بیت، نیمی از کل رشته بیت Pad را تشکیل می‌دهد ولی از آنجایی که طرفین آن را می‌دانند می‌تواند 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 رشتهٔ Pad به اندازهٔ مطلوب و مورد باشد. بدین نحو مسئله حل شده است.

اما لحظه‌ای تأمل کنید؛ در این میان ترودی را فراموش کرده‌ایم. فرض کنید او کنجکاو است بداند که آلیس چه می‌کند، لذا فیبرنوری را قطع می‌کند و مدار آشکارساز (Detector) و فرستندهٔ خود را در میانهٔ فیبر جاسازی می‌کند، [و بدین ترتیب یک انشعاب فعال برای استراق سمع پدید می‌آورد]. ترودی نمی‌داند برای هر فوتون حامل داده چه مبنایی را [برای پلاریزاسیون] در نظر بگیرد. بهترین کاری که می‌تواند انجام بدهد آن است که برای هر فوتون یک مبنای تصادفی انتخاب کند، دقیقاً همانند کاری که باب می‌کند. مثالی از انتخابهای تصادفی او در شکل ۸-۵-و نشان داده شده است. بعداً وقتی باب به آلیس (به صورت آشکار) گزارش می‌دهد که مبناهای انتخابی او چه بوده و آلیس نیز به او می‌گوید که کدام صحیح و کدام غلط است، ترودی نیز می‌داند که انتخابهای او کدام صحیح و کدام غلط هستند. مطابق با شکل ۸-۵ او بیتهای ۰ و ۱ و ۲ و ۳ و ۴ و ۶ و ۸ و ۱۲ و ۱۳ را صحیح بدست آورده است ترودی از پاسخ آلیس به باب متوجه می‌شود 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 نمی‌داند که بیت ارسال شده چه بوده است. بنابراین مطابق با شکل ۸-۵-ه باب می‌داند که Pad او با رشته بیت 01011001 آغاز می‌شود در حالی که آنچه ترودی از این Pad در اختیار دارد مطابق با شکل ۸-۵-ز رشتهٔ 01?1??0? است.

البته آلیس و باب هر دو آگاهند که ممکن است ترودی بخشی از Pad آنها را بدست بیاورد، به همین دلیل مایلند اطلاعاتی که ترودی دارد را کاهش بدهند. لذا آنها می‌توانند با انجام عملیات تبدیل (Transformation) بر روی Pad این کار را انجام بدهند. به عنوان مثال آنها می‌توانند Pad طولانی خود را به بلوکهای ۱۰۲۴ بیتی تقسیم

کرده و سپس آن را به توان دو برسانند تا به عددی ۲۰۴۸ بیتی تبدیل شود؛ سپس بلوکهای ۲۰۴۸ بیتی به هم چسبیده و Pad اصلی (کلید رمز) را تشکیل می‌دهند. در این صورت ترودی با داشتن اطلاعات جزیی از بلوکهای ۱۰۲۴ 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**»[1] 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 بی‌نوا! نه 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 محاورهٔ مستقیم با آلیس است [تا حضورش کماکان مخفی بماند.] مشکل آن است که بهترین کاری که ترودی می‌تواند انجام بدهد آن است که کوبیتهای 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**»[2] 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»[3] 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.

۸-۱-۵ دو اصل 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

---

۱. Privacy Amplification ۲. Forward Error Correcting Code ۳. Cloning

لزومی ندارد که اطلاعات واقعی به همانگونه که هستند رمز و ارسال شوند.[۱] یک مثال می‌تواند به فهم دلیل این نیاز کمک کند. فرض کنید یک شرکت به نام TCP[۲] (The Couch Potato) با ۶۰۰۰۰ کالا، از طریق سیستم پست الکترونیکی سفارش خرید می‌پذیرد.برنامه‌نویسان شرکت TCP به خیال آن که برنامه‌های مؤثر و کارآمدی می‌نویسند، 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 شمارهٔ کالا) در نظر می‌گیرند که سه بایت آخر توسط یک کلید بسیار طولانی رمزنگاری می‌شود و این کلید را فقط مشتری و شرکت TCP می‌داند.

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 می‌کند. فرض کنید یک کارمند اخراجی کینه‌جو می‌خواهد شرکت TCP را تنبیه کند. لذا قبل از ترک شرکت فهرست مشتریان این شرکت را با خود به همراه می‌برد. [فهرست مشتریان محرمانه و سرّی نیست و رمزنگاری نیز نمی‌شود.] او در طول شب برنامه‌ای می‌نویسد 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 می‌دهد و بدین طریق صدها سفارش جعلی برای شرکت TCP ارسال می‌کند.

وقتی این پیامها دریافت می‌شوند کامپیوتر شرکت TCP ابتدا کلید مربوط به هر مشتری را پیدا می‌کند تا بدنهٔ 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 می‌شود] بنابراین کامپیوتر، شروع به چاپ سفارشهای خرید و صورتحساب می‌کند. گرچه شاید عجیب به نظر برسد که یک مشتری ۸۳۷ عدد تاب برای بچه‌ها یا ۵۴۰ جغجغه سفارش بدهد ولی براساس دانشی که یک کامپیوتر دارد شاید خریدار، این مقدار تاب را برای ساخت یک شهربازی در نظر داشته است! بدین ترتیب یک اخلالگر فعال (کارمند اخراجی) توانسته مشکلات بزرگی را برای شرکت TCP ایجاد کند هر چند خود اخلالگر نمی‌تواند بفهمد پیامهایی که کامپیوتر او تولید کرده چه هستند!

این مسئله را می‌توان 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 حملهٔ فوق عملی نخواهد بود زیرا کارمند اخراجی [با تولید اعداد تصادفی ۱۲ بایتی] قادر نخواهد بود یک حجم عظیم از سفارشهای معتبر تولید نماید.[۳]

حقیقت قضیه آن است که تمام پیامها باید مقدار قابل توجهی افزونگی (Redundancy) داشته باشند بگونه‌ای که یک اخلالگر فعال (Active Intruder) نتواند پیامهای تصادفی بی‌معنا 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 می‌شود که رمزشکنها ساده‌تر 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 رشتهٔ رمز شده نباید پس از رمزگشایی معادل با اصل پیام باشد بلکه باید داده‌های زائد و حساب شده‌ای در درون آن جاسازی شده باشد. -م.

۲ اصطلاح The Couch Potato واژه‌ای فکاهی است که معنای کار بیهوده و مسخره دارد و در فارسی در افواه عامه شرکت‌کشک‌سانی ترجمه می‌شود!!!

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 می‌فروشد! در این راستا از خط تلفن شرکت TCP دزدانه انشعاب گرفته است و اطلاعات خط را استراق سمع می‌کند. در طرح اصلی با سه بایت در بدنهٔ پیام، شکستن رمز و بدست آوردن کلید تقریباً غیرممکن است زیرا پس از حدس زدن یک کلید، رمزشکن هیچ راهی برای اثبات صحت حدس خود ندارد چرا که تقریباً تمام پیامها معتبرند. در طرح جدید با پیامهای ۱۲ بایتی ، رمزشکن به سادگی می‌تواند پیامهای معتبر را از پیامهای غیر معتبر تشخیص بدهد. [زیرا از بین حدسهایی که در مورد کلید رمز آزمایش می‌کند حدسی درست است که پیامی با ۹ بایت صفر در ابتدای آن تولید کند.] بهرحال 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 محاسبهٔ ساده [از پیامهایی که به صورت تصادفی تولید شده‌اند] تشخیص بدهد. این افزونگی بدان جهت نیاز است که از ارسال پیامهای بی‌ارزش اخلالگران و فریب خوردن گیرنده در رمزگشایی پیامها و پردازش آنها جلوگیری شود. ولیکن همین افزونگی، شکستن سیستم رمز توسط اخلالگران غیرفعال را ساده‌تر می‌سازد؛ لذا در اینجا یک تناقض پدید می‌آید. بعلاوه افزونگی اطلاعات نباید در قالب اضافه کردن تعداد n تا صفر به ابتدا یا انتهای پیام، انجام بگیرد چراکه اجرای الگوریتمهای رمزنگاری بر روی چنین پیامهایی، نتایج قابل پیش‌بینی برخواهد گرداند و کار رمزشکن را ساده‌تر می‌کند. اضافه کردن یک چندجمله‌ای CRC به جای دنباله‌ای از صفرها بهتر خواهد بود زیرا از یک طرف گیرندهٔ اصلی براحتی می‌تواند صحت پیامها را بررسی کند و از طرف دیگر رمزشکن را با حجم کار بسیار زیاد مواجه می‌کند [تا عمل رمزشکنی ناموفق باشد]. حتی بهتر از آن استفاده از «توابع درهم‌سازی رمز» (Hash) است، مفهومی که بعداً آن را بررسی خواهیم کرد.

اگر برای لحظاتی به رمزنگاری کوآنتومی برگردیم، می‌توانیم به نقشی که «افزونگی» در آن سیستم ایفاء می‌کند 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 نسخهٔ 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 غیرمنطقی 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 می‌شود. ولی بهرحال 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 مجدّد پیامهای قدیمی توسط یک اخلالگر فعال، الزامی است. اگر چنین بررسیهایی انجام نشود کارمند اخراجی ما [در نمایشنامهٔ 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 می‌توان در قالب زیر بازگو کرد:

اصل اساسی ۲ در رمزنگاری : به روشهایی نیاز است تا از حملات منجر به تکرار پیام جلوگیری شود.

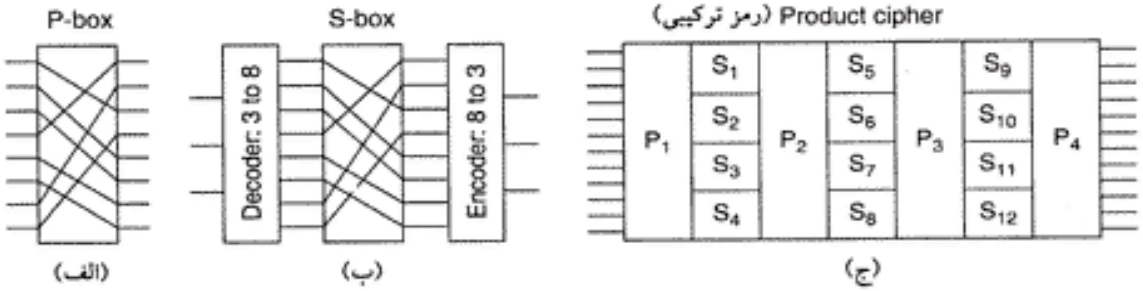
یک چنین محاسبه‌ای را می‌توان با قرار دادن یک «**مُهر زمان**» (Timestamp) در پیامها پیش‌بینی کرد به نحوی که پیام مثلاً برای ده ثانیه معتبر باشد. گیرندهٔ پیام می‌تواند آن را برای حدود ده ثانیه نگه دارد تا بتواند پیامهای جدید را با آن مقایسه کرده و نسخه‌های تکراری را حذف نماید. پیامهایی که بعد از ده ثانیه دریافت شوند کنار گذاشته می‌شوند؛ بدین ترتیب پیامهای تکراری که دارای مُهر زمان هستند، به عنوان پیامهای قدیمی شناخته و حذف خواهند شد. به غیر از روش مُهر زمان (Timestamp)، روشهای دیگری برای ارزیابی تازگی پیام وجود دارد که در ادامه تشریح خواهند شد.

۸-۲ الگوریتمهای رمزنگاری با کلید متقارن (Symmetric-Key)

روشهای پیشرفته و پیچیدهٔ رمزنگاری از اصول و قواعدی مشابه با رمزنگاری سنتی (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 تودهٔ 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 پیشرفتهٔ 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 سادهٔ الکترونیکی پیاده‌سازی شوند. شکل ۸-۶-الف ابزاری را نشان می‌دهد که به نام P-box (P 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 خاصّ، 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 کرکهف 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 گونه‌ای که در شکل ۸-۶-ب نشان داده شده توسط بلوکی به نام S-box انجام می‌شود. در این مثال یک ورودی سه بیتی به بلوک S-box وارد شده و یک رمز سه بیتی از آن خارج می‌شود. ورودی سه بیتی، یکی از هشت خط خروجی بخش اول را فعال کرده و آن را به ۱ تنظیم می‌کند در حالی که بقیهٔ خطوط صفر هستند. بخش دوم یک P-box است. سومین بخش، خط انتخاب شدهٔ ورودی را مجدداً در سه بیت کُد می‌کند. منطبق با سیم‌بندی مثال ۸-۶-ب اگر هشت عدد اکتال (مبنای هشت) ورودی را به ترتیب ۰۱۲۳۴۵۶۷ در نظر بگیریم، توالی خروجی به ترتیب عبارتند از ۲۴۵۰۶۷۱۳. به عبارت دیگر کد صفر با ۲ جابجا می‌شود، ۱ با ۴ جابجا می‌شود و به همین ترتیب. در این ساختار نیز با سیم‌بندی مناسب در بخش P-box از S-box، هر گونه جانشینی دلخواه ممکن و میسر خواهد بود. به علاوه چنین ابزاری را می‌توان با سخت‌افزار 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 می‌تواند کمتر از یک پیکو ثانیه باشد.

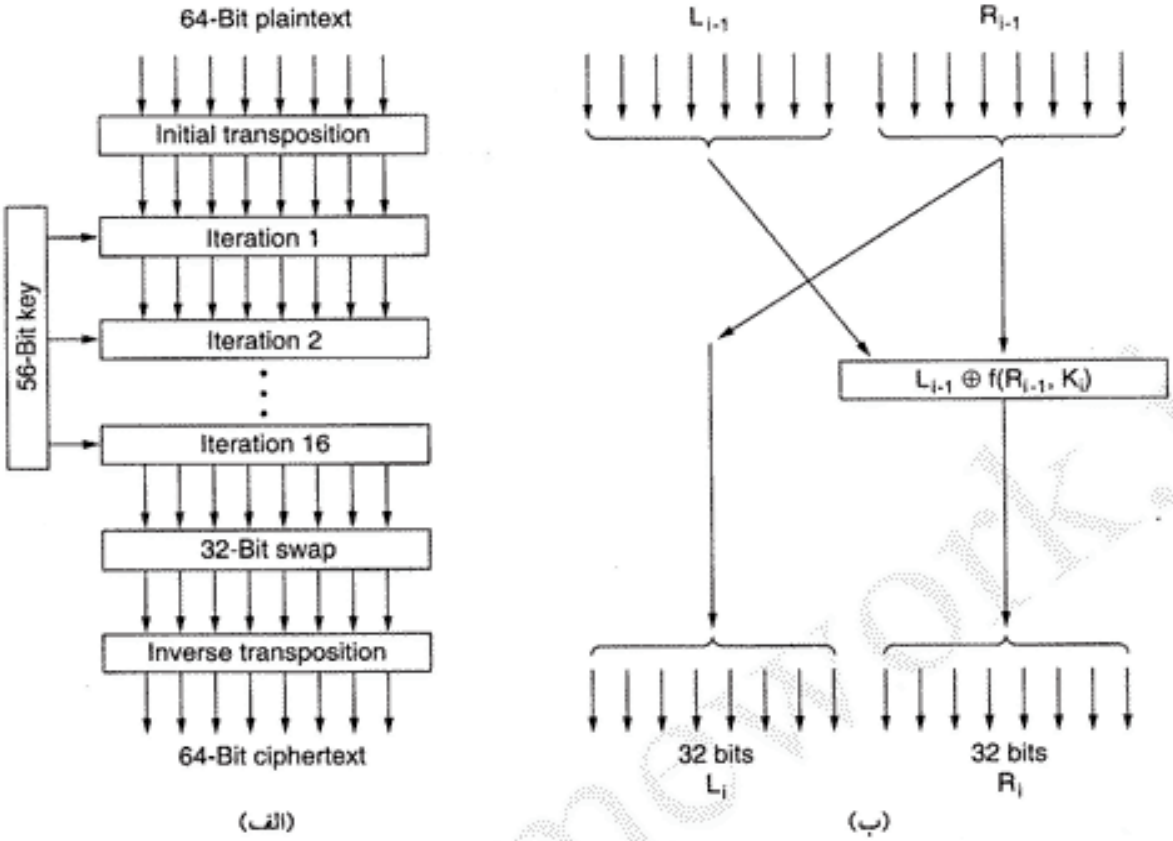
قدرت واقعی این عناصر پایه، زمانی مشخص می‌شود که بخواهیم با ترکیب تعدادی از این بلوکها همانند شکل ۸-۶-ج، یک سیستم رمزنگار ترکیبی ایجاد نماییم. در این مثال در مرحلهٔ اول، به ۱۲ خط ورودی جایگشت داده می‌شود (P_1). در تئوری می‌توان بلوکی داشت که مستقیماً یک عدد ۱۲ بیتی را به عدد ۱۲ بیتی دیگر بنگارد ولی در عمل چنین ابزاری به 2^{12} یعنی ۴۰۹۶ اتصال متقاطع (در بخش دوم) نیاز خواهد داشت. بجای آن، ورودیها به چهار دستهٔ سه بیتی شکسته شده و هر یک از این سه بیت به صورت مستقل جایگشت داده می‌شود. اگرچه این روش کمتر معمول است ولیکن در نوع خود قدرتمند بشمار می‌رود. با قرار دادن تعداد زیاد و کافی از این بلوکها در ساختار ترکیبی فوق، می‌توان 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 می‌کنند بسیار رایج هستند. بطور معمول k بین ۱۶ تا ۲۵۶ متغیر است. یک سخت افزاز رمزنگار ترکیبی برخلاف شکل ۸-۶-ج به جای هفت بخش حداقل هجده بخش فیزیکی متوالی دارد. پیاده‌سازی نرم‌افزاری این روش، در قالب یک حلقه با حداقل هشت تکرار، برنامه‌نویسی می‌شود و در هر تکرار عملکرد یکی از S-boxها بر روی یک بلوک ۶۴ تا ۲۵۶ بیتی انجام و سپس عملیات جایگشت محاسبه می‌شود. اغلب در شروع، یک جایگشت ابتدایی بر روی داده‌ها انجام می‌شود سپس عملیات رمزنگاری انجام شده و در پایان نیز یک مرحله جایگشت تکمیلی انجام می‌گیرد. در ادبیات رمزنگاری هر مرحلهٔ تکرار یک «دور» (Round) نامیده می‌شود.

۸-۲-۱ رمزنگاری DES (Data Encryption Standard)

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 طبقه‌بندی نشدهٔ خود پذیرفت. از این رمز یعنی DES، بطور گسترده‌ای توسط صنایع در محصولات امنیتی استفاده شد. البته این سیستم با شکل اولیه و اصلی آن چندان امن نیست ولی شکل اصلاح شدهٔ آن هنوز هم سودمند است. در این مبحث چگونگی 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 می‌شود و نهایتاً متن ۶۴ بیتی رمز در خروجی بدست می‌آید. این الگوریتم رمزنگاری که دارای یک پارامتر ۵۶ بیتی به عنوان کلید است، عملیات لازم را در ۱۹ مرحلهٔ مجزا انجام می‌دهد. 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 (مرحلهٔ نوزدهم) دقیقاً عکس این



شکل ۸-۷. استاندارد DES (الف) الگوی کلی (ب) جزئیات یکی از مراحل تکرار. علامت ⊕ بمعنای XOR است.

جایگشت انجام می‌گیرد. مرحله قبل از آخر نیز، جابجایی ۳۲ بیت سمت چپ با ۳۲ بیت سمت راست می‌باشد. ۱۶ مرحلهٔ 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 ۳۲بیتی وارد و دو رشتهٔ ۳۲ بیتی خروجی، تولید می‌شود. رشتهٔ ۳۲ بیتی سمت چپ در خروجی دقیقاً مشابه رشتهٔ ۳۲ بیتی سمت راست است. رشتهٔ ۳۲بیتی سمت راست حاصل عمل XOR رشته ورودی سمت چپ با تابعی از رشتهٔ 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 توسعهٔ عدد ۳۲بیتی R_{i-1} (رشتهٔ ۳۲بیتی سمت راست ورودی) یک عدد ۴۸ بیتی به نام E بدست می‌آید که براساس یک جایگشت ساده و تکرار برخی از بیتها انجام می‌شود.[1] در مرحلهٔ دوم E و K_i با یکدیگر XOR می‌شوند. [E حاصل مرحلهٔ قبل و K_i کلید این مرحله است که از کلید ۵۶بیتی 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 می‌شود. قبل از آن که الگوریتم آغاز شود، یک جایگشت ۵۶ بیتی بر روی کلید اعمال می‌شود [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 شمارهٔ مرحله، به سمت چپ شیفت گردشی (Rotation) داده می‌شوند. [مثلاً برای کلید مرحلهٔ پنجم هر بخش ۲۸ بیتی، پنج بیت به سمت چپ شیفت گردشی داده می‌شود.] پس از این مرحله برای محاسبهٔ هر کلید یعنی K_i^+ یک جایگشت ۵۶ بیتی دیگر بر روی آن اعمال شده و نهایتاً یک زیرمجموعهٔ چهل و هشت بیتی از این ۵۶ بیت استخراج و در هر مرحله از آن استفاده می‌شود.

تکنیکی که برخی از اوقات برای قدرتمندتر کردن سیستم DES بکار می‌رود 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 می‌شود؛ سپس خروجی DES مجدداً با کلید دوم XOR می‌گردد. عمل «سفیدسازی» به سادگی برگشت‌پذیر است و برای این کار عکس عملیات فوق انجام می‌شود (به شرط آن که کلید سفیدسازی در اختیار گیرنده باشد). از آنجایی که این تکنیک از دو کلید ۶۴ بیتی اضافی استفاده می‌کند، 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 می‌شود. (به عبارت بهتر تنها یک کلید سفیدسازی وجود دارد.)

از زمانی که DES معرفی و بکار گرفته شد بحث و مناقشات گسترده‌ای 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 می‌کرد. وقتی دولت فدرال ایالات متحده خواست که یک سیستم رمزنگاری را برای پرونده‌های طبقه‌بندی 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 مجموعهٔ ریاضیدانان و رمزشکنهای دنیاست). NSA گروهی به شدت سرّی است تا جایی که در مورد آن یک لطیفه وجود دارد:

س. NSA مخفف چیست؟

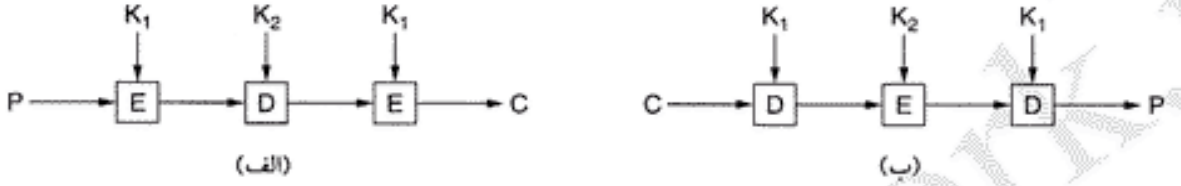
ج. مخفف No Such Agency به معنای «اصلاً چنین آژانسی وجود ندارد!» اما در واقع NSA مخفف کلمات National Security Agency (آژانس امنیت ملی) است.

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 بودجهٔ کمتر قادر به چنین کاری نباشد. از طرفی محرمانه بودن طرح، احتمالاً بدان دلیل بوده که در آن یک رخنهٔ 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 برنامهٔ کنفرانس رمزنگاری خود را لغو نماید تا کار عموم را برای رمزشکنی راحتتر از قبل نکند، دامنهٔ این شایعات قوت گرفت. بعداً NSA همه چیز را انکار کرد!

در سال ۱۹۷۷ دو محقق رمزنگاری از دانشگاه استَنفورد به نامهای «دیفی» و «هِلمن» ماشینی برای شکستن رمز DES طراحی کردند و تخمین زدند که این ماشین با هزینه‌ای حدود بیست میلیون دلار قابل ساخت است. این ماشین می‌توانست با داشتن یک قطعهٔ 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 بین‌المللی 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 مرحلهٔ اول، متن اصلی با کلید K_1 و مبتنی بر روش معمولی DES رمزنگاری می‌شود. در مرحلهٔ بعدی DES در حالت **رمزگشایی** ولی با کلید K_2 بر روی نتیجهٔ مرحلهٔ قبل اعمال می‌شود. نهایتاً بار دیگر رمزنگاری DES با کلید K_1 (کلید اول) انجام می‌شود.



شکل ۸-۸ (الف) سه بار رمزنگاری بکمک استاندارد DES. (ب) رمزگشایی.

این سبک طراحی بلافاصله دو سؤال را به پیش می‌کشد. 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*[1]) از روش «رمزنگاری-رمزگشایی-رمزنگاری» (اصطلاحاً EDE[2]) استفاده شده است؟

دلیل آن که فقط از دو کلید استفاده شده است آن است که حتی وسواسی‌ترین رمزنگاران اذعان دارند که برای کاربردهای تجاری کنونی یک کلید ۱۱۲ بیتی بخوبی کفایت می‌کند و مطمئن است. (در بین رمزنگاران، وسواس و تردید یک ویژگی ممتاز تلقی می‌شود نه یک اشکال!) حرکت به سمت کلید ۱۶۸ بیتی [۳کلید ۵۶بیتی] در مقایسه با بهرهٔ 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 عددِ ۶۴بیتی به عددِ ۶۴بیتی دیگر هستند. استفاده از روش EDE به جای EEE این امتیاز بزرگ را دارد که یک کامپیوتر که مبتنی بر «DES سه گانه» عمل می‌کند، 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 می‌داد که سیستم «DES سه‌گانه» 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!

۸-۲-۲ استاندارد پیشرفتهٔ رمزنگاری: AES

در حالی که DES آرام آرام به پایان عمر خود نزدیک می‌شد (حتی با ابداع DES سه گانه)، برای NIST[3] در ایالات متحده (که مسئولیت بهبود استانداردهای دولت فدرال آمریکا را برعهده گرفته است)، مسجّل شد که دولت به یک استاندارد رمزنگاری جدید برای اسناد طبقه‌بندی نشده خود نیاز مبرم دارد. NIST به فراست از دشمنان DES آگاه بود و نیک می‌دانست که اگر به یکباره استاندارد جدیدی را معرفی نماید همهٔ آنانی که دستی در رمزنگاری دارند ناخودآگاه فرض را بر آن می‌گذارند که باز هم آژانس سرویسهای محرمانهٔ ایالات متحده، (NSA) یک **رخنه**[4] در

---

۱. Encrypt-Encrypt-Encrypt
۲. Encrypt-Decrypt-Encrypt
۳. National Institute of Standard Technology
۴. در پشتی یا Backdoor

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 نمی‌کرد و به احتمال زیاد استاندارد جدید نیز در یک مرگ آرام رو به زوال می‌رفت!

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 پیشرفتهٔ 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. پیاده‌سازی سخت‌افزاری و نرم‌افزاری الگوریتم ممکن باشد.
۵. الگوریتم باید عمومی (غیرانحصاری) یا تحت قوانین غیرانحصاری ثبت شده باشد.[۱]

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»، «کارآیی»، «سادگی»، «قابلیت انعطاف» و «فضای حافظهٔ مورد نیاز برای پیاده‌سازی» (که در سیستمهای درونکار ـ Embedded ـ 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 رای‌گیری 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 می‌دهد و در نوامبر ۲۰۰۱ روش Rijndael استاندارد دولت ایالات متحده شد و در سند استاندارد FIPS 197 ثبت گردید. به دلیل فضای آزاد شگفت‌انگیز حاکم بر این رقابت و همچنین ویژگیهای فنی برتر Rijndael و با توجه بدان که تیم پیشنهاددهنده دو رمزنگار جوان بلژیکی بودند (که شائبهٔ وجود رخنهٔ مورد نظر NSA در آن را از اذهان می‌زداید)، انتظار می‌رود که Rijndael لااقل برای یک دهه استاندارد رمزنگاری کل دنیا شود. روش Rijndael کم و بیش به صورت «رایـن‌دال» تلفظ می‌شود و از نام خانوادگی ابداع‌کنندگان آن (Rijmen & Daemon) گرفته شده است.

Rijndael از کلید و بلوکهای دادهٔ ۱۲۸ یا ۲۵۶ بیتی (در قطعات ۳۲ بیتی) حمایت می‌کند؛ طول کلید و طول بلوکهای داده می‌تواند مستقل از هم انتخاب شود. با این وجود استاندارد AES بیان می‌کند که اندازهٔ بلوک داده باید صرفاً ۱۲۸ بیتی باشد ولی طول کلید می‌تواند یکی از سه حالت ۱۲۸، ۱۹۲ و ۲۵۶ انتخاب شود. برای کسی که همواره از کلیدهای ۱۹۲ بیتی استفاده می‌کند استفاده از دو گزینهٔ دیگر AES یعنی یک کلید ۱۲۸ بیتی با بلوک دادهٔ ۱۲۸ بیتی و کلید ۲۵۶ بیتی با بلوک دادهٔ 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 گزینهٔ ۱۲۸/۱۲۸ (کلید رمز ۱۲۸ بیتی / بلوک دادهٔ ۱۲۸ بیتی) را بررسی کرده‌ایم زیرا احتمالاً در کاربردهای معمول دنیای اقتصاد، کلید ۱۲۸ بیتی جا خواهد افتاد. کلید ۱۲۸ بیتی یک فضای حالت با $۲^{۱۲۸}$ ($=۳\times۱۰^{۳۸}$) حالت مختلف ایجاد می‌کند. حتی اگر NSA سفارش ساخت ماشینی با یک میلیارد پردازندهٔ موازی بدهد و هر یک از پردازنده‌ها بتوانند یک کلید را در یک پیکوثانیه ($۱۰^{-۱۲}$Sec) آزمایش کنند، آزمایش تمام این کلیدها حدود $۱۰^{۱۰}$ سال طول خواهد کشید. در آن زمان خورشید خاموش گشته است و مردم مجبورند نتیجهٔ کار را در زیر نور شمع بخوانند!!

Rijndael

از دیدگاه ریاضی، رمزنگاری Rijndael مبتنی بر «نظریهٔ میدان گالوا» است که به آن ویژگیهای امنیتی قابل اثبات و مطمئنی بخشیده است. با این وجود می‌توان 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 «دور» (Round) تشکیل شده است. تعداد «دور» بستگی به طول کلید و اندازهٔ بلوک داده خواهد داشت: از ۱۰ دور برای کلید ۱۲۸ بیتی با بلوک دادهٔ ۱۲۸ بیتی تا ۱۴ دور برای بزرگترین کلید [۲۵۶ بیتی] و بزرگترین بلوک داده [۲۵۶ بیتی].[۱] ولی برخلاف DES، عملیات بر روی بایتها انجام می‌گیرد نه بیتها؛ بدین ترتیب پیاده‌سازی سخت افزازی یا نرم‌افزاری آن ساده‌تر و کارآمدتر خواهد بود. کلیّات کُد 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 برنامهٔ Rijndael.

---

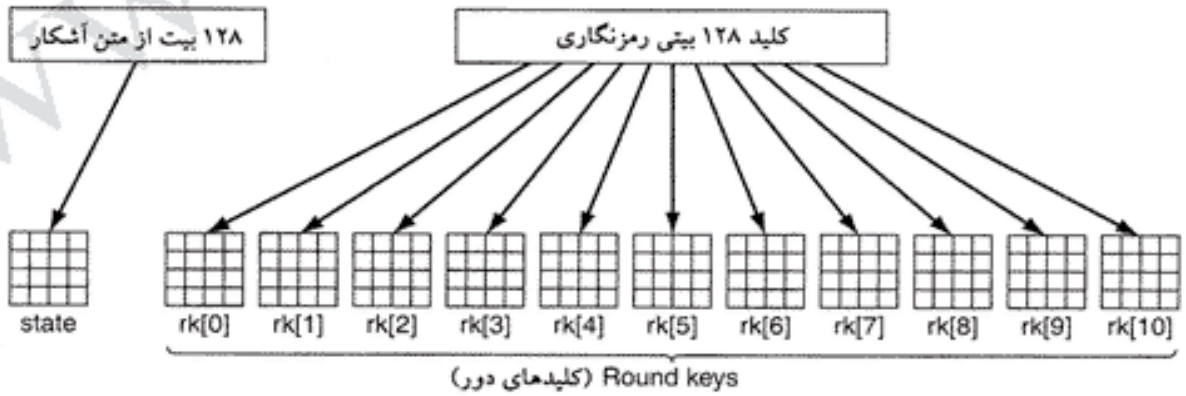
۱. دقت کنید که در رمزنگاری رایِن‌دال طول بلوکهای داده می‌تواند ۲۵۶ بیتی نیز باشد و آنکه در بخش قبلی در مورد AES عنوان شد (که طول بلوک داده باید «صرفاً ۱۲۸ بیتی» باشد) استاندارد NITS است نه الزام طراحان آن -م.

تابع *rijndael* سه پارامتر دارد که عبارتند از: (۱) *plaintext*: یک آرایه ۱۶ بایتی محتوی داده‌های ورودی؛ (۲) *ciphertext*: یک آرایهٔ ۱۶ بایتی که نهایتاً محتوی خروجی رمز شده را باز می‌گرداند. (۳) *key* کلید رمز ۱۶ بایتی (۱۲۸ بیتی). در خلال پردازش، حالت فعلی داده‌ها در یک آرایهٔ دوبعدی به نام *state* که اندازهٔ آن *NROWS×NCOLS* است حفظ می‌شود. برای بلوکهای ۱۲۸ بیتی داده این آرایه ۴×۴ (یعنی ۱۶ بایتی) است. در این ۱۶ بایت، یک بلوک ۱۲۸ بیتی داده قابل ذخیره خواهد بود.

در ابتدا آرایهٔ *state* با داده‌های رمز نشدهٔ ورودی، مقداردهی اولیه می‌شود و سپس در هر مرحله از محاسبات مقدار آن تغییر خواهد کرد. در برخی از مراحل فقط جانشینی بایت به بایت انجام می‌شود و در برخی دیگر بر روی محتویات این آرایه جایگشت بایتها انجام می‌شود. البته (بغیر از جانشینی و جایگشت) تبدیلات دیگری نیز بر روی بایتها انجام می‌گیرد. در نهایت محتویات آرایهٔ *state* به عنوان متن رمز شده باز خواهد گشت.

این برنامه با توسعهٔ کلید ۱۲۸ بیتی به یازده آرایهٔ متفاوت و هم اندازه با آرایهٔ *state* کار خود را آغاز می‌کند؛ این یازده کلید در آرایهٔ *rk* ذخیره می‌شوند. *rk* آرایه‌ای از استراکچر است که به زبان C تعریف شده و هر عضو این آرایه خودش از نوع یک آرایهٔ *state* (۴×۴ بایتی) است. یکی از این کلیدها در بدو شروع محاسبات مورد استفاده قرار می‌گیرد و از ده تای دیگر در خلال ده دور پردازش بهره گرفته می‌شود. به هر یک از این ده کلید، «**کلید دور**» (Round Key) گفته می‌شود. 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 می‌کنیم که کلیدهای هر «دور» براساس چرخش کلید (Rotation) و XOR کردن آن با گروههای خاصی از بیتهای خودِ کلید انجام می‌شود. برای تشریح کامل روش به مرجع (Daemen & Rijmen 2002) مراجعه نمائید.

گام بعدی آن است که متن اصلی در درون آرایهٔ *state* کپی شود تا در خلال ده دور متوالی پردازش شود. عمل کپی درون این آرایه به صورت ستونی انجام می‌شود یعنی اولین چهار بایت، در ستون اول (ستون شمارهٔ صفر)، چهار بایت دوم در ستون دوم (ستون شمارهٔ ۱) کپی می‌شود و به همین ترتیب ادامه می‌یابد. شماره‌گذاری ستونها و سطرها از شمارهٔ صفر شروع شده است در حالی که مراحل پردازش (دورها) از شمارهٔ یک شماره‌گذاری می‌شوند. مراحل تنظیم مقداردهی مقدماتی آرایه‌های ۴×۴ در شکل ۸-۱۰ نشان داده شده است.



شکل ۸-۱۰. ایجاد آرایه‌های state و rk.

قبل از شروع محاسبات اصلی، یک کار دیگر نیز انجام می‌شود: *rk*[0] با آرایهٔ *state* بایت به بایت *XOR* می‌شود. به عبارت دیگر هر یک از ۱۶ بایت آرایهٔ *state*، با مقدار حاصل از *XOR* خودش با بایت متناظر در *rk*[0] تعویض می‌گردد.

حال زمان شروع محاسبات اصلی فرا رسیده است. حلقه ده بار تکرار می‌شود (یکبار به ازای هر دور) و در هر تکرار محتوای *state* تغییر می‌کند. هر دور شامل چهار گام است: در گام اول محتوای *state* بایت به بایت با مقادیر جدید «جانشین» (Substitute) می‌شود. در حقیقت محتوای هر بایت به عنوان اندیس ورودی به یک S-box اعمال می‌شود تا خروجی معادل با خود را تولید نماید. این مرحله یک سیستم جانشینی ساده و کاراکتر به کاراکتر (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 سطرِ آرایهٔ *state* را به سمت چپ می‌چرخاند: سطر شماره صفر، صفر بایت می‌چرخد (یعنی تغییر نمی‌کند)، سطر شمارهٔ یک، یک بایت به سمت چپ می‌چرخد، سطر شمارهٔ دو، دو بایت و سطر شمارهٔ سه، سه بایت. در این گام محتوای بلوک فعلی آرایه بهم ریخته می‌شود که معادل با بلوک جایگشت در شکل ۸-۶-الف است.

در گام سوم هر ستون (از آرایهٔ *state*) بطور مستقل از دیگری درهم ریخته می‌شود. این عمل براساس ضرب ماتریسی انجام می‌گیرد، بدین نحو که ستون فعلی در یک ماتریس ثابت ضرب شده و ستون جدید را تولید می‌کند. عمل ضرب ماتریس مبتنی بر نظریه «**میدان محدود گالوا**»[1] یعنی $GF(2^8)$ انجام می‌شود. 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[2] و سه عمل XOR محاسبه می‌شوند. برای کسب اطلاعات تفصیلی به مرجع (Rijndael, Daemen; 2002) مراجعه کنید.

نهایتاً در گام چهارم «**کلید دور**» با آرایهٔ *state* بایت به بایت XOR می‌شود.

از آنجایی که یکایک مراحل به سادگی برگشت‌پذیر هستند، لذا عمل رمزگشایی با اجرای برعکس الگوریتم [از آخر به اول] انجام می‌شود. با این وجود یک راه زیرکانه وجود دارد که در آن عمل رمزگشایی با اجرای همان الگوریتم رمزنگاری ولی با جداول متفاوت انجام می‌گیرد.

این الگوریتم هم امنیت بسیار بالا و هم سرعت بسیار عالی را تضمین می‌کند. پیاده‌سازی نرم‌افزاری آن بر روی یک ماشین 2-GHz می‌تواند عمل رمزنگاری هفتصد مگابیت داده در ثانیه را به صورت بلادرنگ انجام بدهد که برای رمزنگاری صد کانال ویدیویی MPEG-2 به صورت همزمان کفایت می‌کند. پیاده‌سازی سخت‌افزاری، از این هم سریعتر خواهد بود.

۸-۲-۳ حالات رمز (Cipher Modes)

علیرغم تمام پیچیدگیها، AES (یا DES یا هر سیستم رمزنگار که بر روی بلوک محدودی از داده‌ها عمل می‌کند) 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 می‌شود. 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 نتیجهٔ یکسان خواهید گرفت. یک رمزشکن می‌تواند از این ویژگی برای واژگون کردن رمز (استخراج اطلاعات بدون در اختیار داشتن کلید) سوءاستفاده کند.

حالت کتابچهٔ رمز (Electronic Code Book Mode)

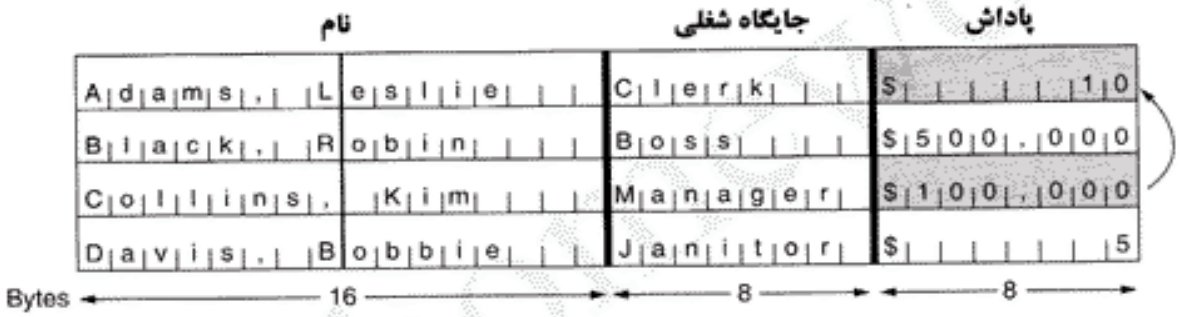
برای آن که ببینیم در رمزهای جانشینی چگونه از خصوصیت فوق‌الذکر برای شکستن رمز سوءاستفاده می‌شود،

---

۱. Finite Galois Field ۲. جستجو از طریق Lookup Table

سیستم رمز «DES سه‌گانه» (Triple DES) را بررسی می‌کنیم زیرا نشان دادن بلوکهای ۶۴ بیتی داده از بلوکهای ۱۲۸ بیتی ساده‌تر است ولیکن سیستم AES نیز دقیقاً همین مشکل را دارد. ساده‌ترین راه برای رمزنگاری یک قطعه طولانی داده آن است که به قطعات متوالی هشت بایتی (۶۴بیتی) 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 ۸بایت) داشته باشد. این روش (یعنی قطعه قطعه کردن داده‌ها به بخشهائی با طول ثابت) ECB Mode[1] نامیده می‌شود که مشابه با یک سیستم قدیمی رمزنگاری است که در آن کلمات یک متن تفکیک شده و به جای آنها بکمک کتابچهٔ رمز یک کد پنج رقمی (در مبنای ده) جایگزین می‌شد تا متن رمز شده را ایجاد نماید.

در شکل ۸-۱۱، ابتدای یک فایل کامپیوتری را مشاهده می‌کنید که در آن فهرستی از پاداشهای سالانهٔ یک شرکت که قرار است به کارمندان خود اعطاء کند، درج شده است. این فایل از رکوردهای متوالی ۳۲بایتی تشکیل شده و به ازای هر کارمند یک رکورد، طبق قالب ذیل ذخیره شده است: ۱۶ بایت برای نام، ۸بایت برای ردهٔ شغلی و ۸بایت برای پاداش. فایل مربوطه که جمعاً شامل ۱۶ بلوک ۸بایتی است که از شمارهٔ صفر تا ۱۵ شماره‌گذاری شده و بکمک «سیستم DES سه‌گانه» (Triple DES) رمزنگاری شده است.



شکل ۸-۱۱. متن اصلی از یک فایل که در قالب ۱۶ بلوک بروش DES رمز شده است.

فرض کنید Leslie با رئیسش مشکل دارد و طبعاً انتظار دریافت پاداش چندانی در سر نمی‌پرورد. در طرف مقابل Kim مورد توجه و عنایت رئیس است و همه این را می‌دانند. Leslie می‌تواند 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 می‌تواند بلوک رمز شدهٔ هشتم ـ پاداش رئیس ـ را برای خودش کپی کند ولیکن به احتمال زیاد قضیه فاش خواهد شد!! گذشته از آن Leslie آدم حریص و زیاده‌طلبی نیست!)

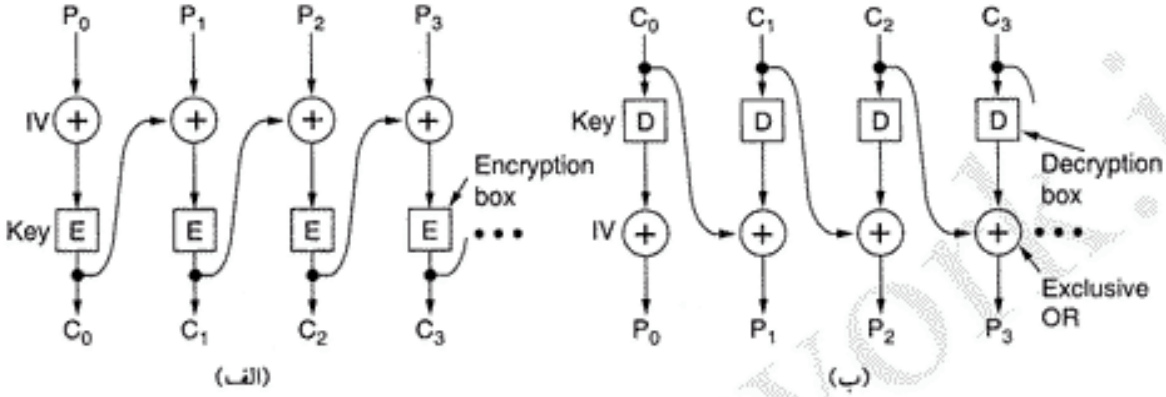
حالت زنجیره‌سازی بلوکهای رمز (Cipher Block Chaining Mode)

برای خنثی کردن این نوع حملات، بلوکهای داده می‌توانند به صورت زنجیره‌ای رمز شوند به گونه‌ای 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 بی‌معنی تبدیل گردد. یکی از روشهای زنجیره‌سازی، اصطلاحاً روش

---

۱. Electronic Code Book Mode

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 شدهٔ قبل از خودش XOR می‌شود. بدین ترتیب بلوکهای یکسان متن به بلوکهای رمز شدهٔ 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 می‌شود و به صورت آشکار به همراه داده‌های رمز شده ارسال می‌گردد.



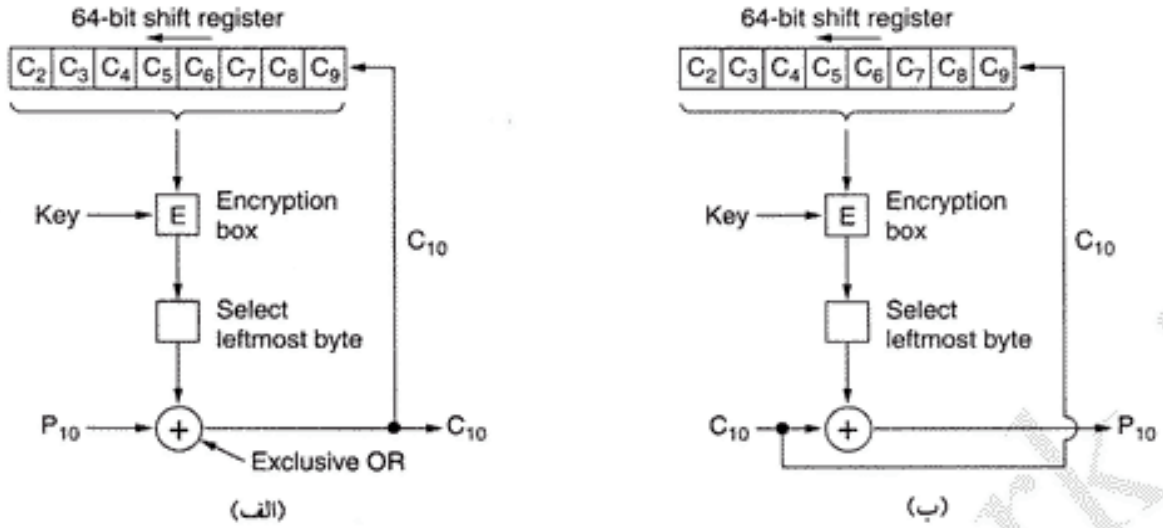
شکل ۸-۱۲. زنجیره سازی بلوکهای رمز. (الف) رمزنگاری. (ب) رمزگشایی.

با بررسی شکل ۸-۱۲ به سادگی می‌توان چگونگی زنجیره‌سازی بلوکهای رمز را مشاهده و بررسی کرد. کار را با محاسبهٔ $C_0=E(P_0 \text{ XOR } IV)$ شروع می‌کنیم. سپس بلوک بعدی رمز را با محاسبهٔ $C_1=E(P_1 \text{ XOR } C_0)$ بدست آورده و به همین ترتیب ادامه می‌دهیم. 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 شدهٔ 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 بلوکها بی‌معنی و نامعتبر خواهند بود. برای یک کارآگاه امنیتی زیرک، محلی که از آنجا به بعد این ناهنجاری بوجود آمده (یعنی محلی که پس از رمزگشایی، بلوکها بی‌معنی شده‌اند) می‌تواند نقطهٔ شروع خوبی برای آغاز بازرسی و پیگیری باشد!

زنجیره سازی بلوکهای رمز [گذشته از خنثی کردن حملاتی نظیر آنچه که در بالا اشاره شد] این حسن بزرگ را دارد که بلوکهای مشابه متن، بلوکهای رمز یکسان تولید نخواهند کرد و بالطبع کار تحلیل‌گر رمز [برای شکستن رمز] بسیار سخت‌تر خواهد شد. در حقیقت این حسن دلیل اصلی استفاده از آن می‌باشد.

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 محاوره‌ای که در آن جا کاربران می‌توانند خطوط یا فرامین کوتاهتر از هشت کاراکتر درج و ارسال کنند و برای دریافت پاسخ منتظر بمانند، چندان مناسب نیست. برای رمزنگاری بایت‌به‌بایت (به نحوی که در شکل ۸-۱۳ می‌بینید) از روش Cipher Feedback Mode مبتنی بر رمزنگاری «DES سه گانه» (Triple DES) استفاده می‌شود. 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 شده‌اند. وقتی بایت دهم از راه می‌رسد، (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.

می‌شوند. سپس بایت سمت چپ این متن رمزی استخراج و با بایت دهم تازه وارد [یعنی P_{10}]، XOR می‌شود 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 می‌گیرد. دقت کنید که محتوای فعلی شیفت رجیستر به پیشینهٔ کل متن ارسالی وابسته خواهد بود، بدین ترتیب یک الگوی تکراری در متن اصلی به صور گوناگون در متن رمز شده نگاشته می‌شود. دقیقاً همانند روش زنجیره سازی بلوکهای متن، در این روش نیز برای شروع گردش عملیات به یک مقدار اولیه (Initialization Vector) نیاز خواهد بود.

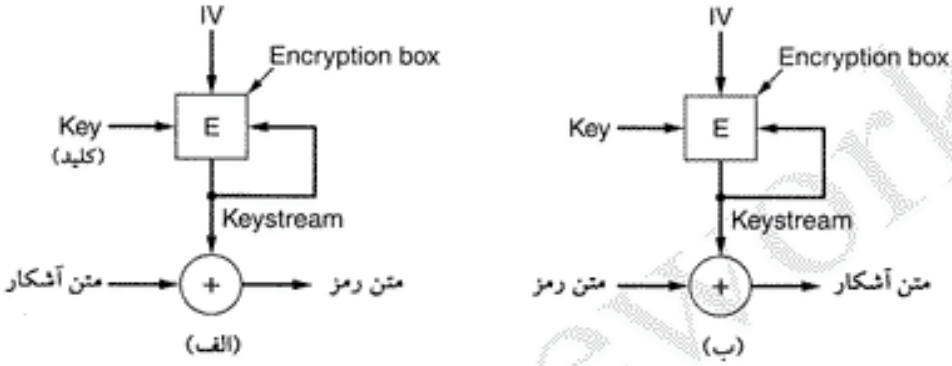
رمزگشایی در روش Cipher Feedback Mode دقیقاً مشابه با عملیات رمزنگاری داده‌ها است. در اصل محتوای شیفت رجیستر به جای رمزگشایی مجدداً رمزنگاری می‌شود زیرا هر گاه خروجی رمز شده شیفت رجیستر، با C_{10} XOR شود P_{10} را نتیجه خواهد داد چراکه در ابتدا برای بدست آمدن C_{10} خروجی رمز شده شیفت رجیستر با P_{10}، 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 می‌شود. 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 نتیجهٔ 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 می‌کند.

Stream Cipher Mode

علیرغم محلی بودن اثر یک بیت خطا در روش قبل، برنامه‌های کاربردی خاصی وجود دارند که در آنها سرایت خطای یک بیت به ۶۴ بیت، تأثیر مخرب بسیار زیاد، بجا می‌گذارد. برای این نوع از برنامه‌های کاربردی گزینهٔ چهارمی به نام Stream Cipher Mode وجود دارد. این روش با رمز کردن یک مقدار اولیه به نام بردار IV (Initialization Vector) توسط یک کلید کارش را شروع می‌کند 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. دنبالهٔ (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.[۱] دقت کنید که IV فقط در مرحلهٔ 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.[۲] 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.

نکتهٔ 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 حملهٔ *Keystream reuse attack* قرار می‌دهد: فرض کنید یک بلوک از متن اصلی، مثلاً P_0 ، طبق قاعدهٔ P_0 XOR K_0 رمز شده باشد. [که در آن P_0 بلوک اول از متن اصلی و K_0 همان Keystream تولید شده در مرحلهٔ 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 قاعدهٔ Q_0 XOR K_0 رمز می‌شود. یک رمزشکن که توانسته بلوکهای رمز شدهٔ پیامهای P و Q را جداگانه در اختیار بگیرد، این دو بلوک را با هم XOR می‌کند و با اینکار کلید از میان می‌رود.[۳] 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

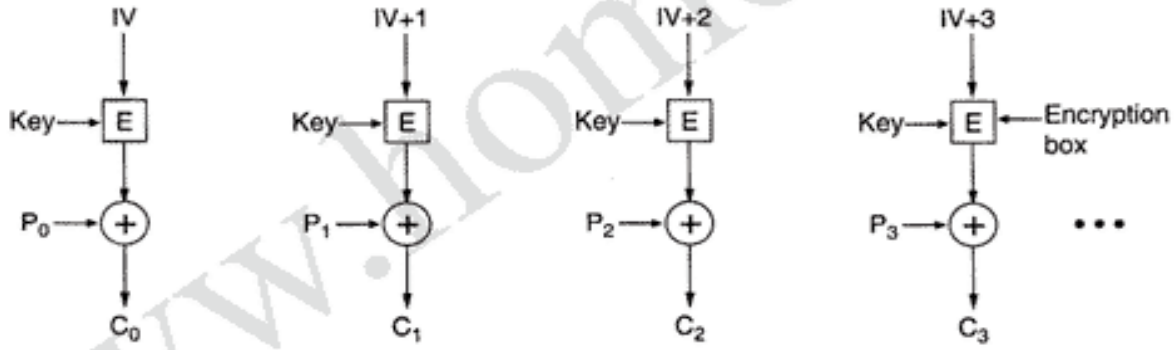
۳. عمل XOR این دو بلوک رمز عبارت است از $(P_0 \text{ XOR } K_0) \text{ XOR } (Q_0 \text{ XOR } K_0)$ که معادل با $P_0 \text{ XOR } Q_0$ خواهد شد. -م

حرف 'e' با Space بالاترین احتمال را دارد و به همین ترتیب. کوتاه سخن آنکه، با در اختیار داشتن حاصل XOR شدهٔ دو متن ، رمزشکن 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 داده‌های 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. ذخیرهٔ فایلها به صورت رمز شده از این دیدگاه که ماشین گیرندهٔ فایل یک کامپیوتر کیفی است و احتمال دارد سرقت شود، کاملاً منطقی است . ذخیرهٔ فایلها به صورت رمز شده خطر فاش شدن اطلاعات را کاهش خواهد داد مخصوصاً زمانی که احتمال دارد یک کامپیوتر بدست افراد ناباب بیفتد.

ولیکن در عمل، دسترسی به فایلهای روی دیسک و بالاخص پایگاههای اطلاعاتی، به صورت غیرترتیبی (nonsequential) انجام می‌شود. برای دسترسی به یک بلوک تصادفی از فایلی که به روش «زنجیره‌سازی بلوکها»[1] رمزنگاری شده است باید ابتدا تمام بلوکهای قبل از آن رمزگشایی شوند که این فرآیند (از لحاظ زمان دسترسی) بسیار پر هزینه است.[2] به همین دلیل روش دیگری ابداع شده که ساختار آنرا در شکل ۸-۱۵ مشاهده می‌کنید. در این روش متن اصلی به صورت مستقیم رمز نمی‌شود بلکه یک بردار اولیه (IV) که با یک عدد صحیح جمع می‌شود توسط کلید رمز شده و نتیجهٔ آن با متن اصلی XOR می‌شود. برای هر بلوک جدید یک واحد به IV اضافه می‌شود 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 می‌برد 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 رمزشکن این دو متن رمز شده را بدست بیاورد. چون کلید و IV مشابهند بنابراین Keystreamها [یعنی کلیدهایی که برای هر بلوک جدید محاسبه و با آن XOR می‌شوند] یکسانند و طبعاً سیستم رمز Counter Mode در معرض حملهٔ *Keystream Reuse Attack* که در بخش قبلی تشریح شد، قرار می‌گیرد. رمزشکن با XOR کردن یکایک بلوکهای این دو متن، کلید رمز را از میان برمی دارد و حاصل XOR شدهٔ دو متن را بدست می‌آورد. این ضعف بدان معنا نیست که روش Counter Mode، نظریهٔ 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.

۸-۲-٤ 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 نرم‌افزارها] پیاده‌سازی شده‌اند. فهرست رایجترین این روشها را در شکل ۸-۱۶ می‌بینید.

نام رمز	ابداع کننده	طول کلید	توضیح
Blowfish	Bruce Schneier	1–448 bits	قدیمی است و کند عمل می‌کند.
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 می‌کنند!
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 گسترده‌ای 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 Cryptoanalysis) است. (توسط Biham و Shamir ، ۱۹۹۳) این روش می‌تواند 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 می‌شود؛ سپس با اعمال آنها در الگوریتم رمزنگاری، اتفاقات حاصل از اجرای هر مرحله از الگوریتم ، بدقت بررسی می‌شود. در بسیاری از حالات ، برخی از الگوهای بیتی نسبت به الگوهای دیگر بیشتر تکرار می‌شوند و این نتیجه و مشاهده می‌تواند در هدایت حملهٔ مبتنی بر احتمال و آمار بسیار مؤثر باشد.

دومین روش ، «**تحلیل رمز خطی**»[1] است (Matsui, 1994) که می‌تواند رمز DES را با آزمایش 2^{43} حالت شناخته شدهٔ 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 می‌شود. هر گاه این روال به تعداد دفعات زیاد تکرار شود، نیمی از بیتها باید صفر و نیم دیگر یک باشند.[2] ولیکن اغلب نتیجه بدست آمده، نسبت به یکی از جهتها (یعنی بیتهای صفر و

---

۱. Linear Cryptanalysis

۲ طبق نظریهٔ 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 می‌کند؛[1] این بایاس حتی اگر کوچک باشد می‌تواند برای کاهش Work Factor (جستجوی فضای کلید) مورد سوءاستفاده واقع شود. برای شرح کامل این روش به مقالهٔ Matsui مراجعه نمائید.

سومین روش برای تحلیل رمز، استفاده از میزان توان الکتریکی مصرفی پردازنده برای یافتن کلید سرّی است. بطور معمول کامپیوترها از ولتاژ ۳ ولت برای نمایش بیت یک و از ولتاژ صفر برای بیت صفر استفاده می‌کنند. بنابراین پردازش بیت ۱ انرژی الکتریکی بیشتری نسبت به بیت صفر مصرف می‌کند. اگر یک الگوریتم رمزنگاری شامل حلقه‌ای باشد که در آن بیتهای کلید به ترتیب پردازش می‌شوند، رمزشکن سیگنال ساعت nگیگاهرتزی را به مقدار کمتری کاهش داده (مثلاً صد مگاهرتز) و یک گیرهٔ کوچک از نوع سوسماری را به پایه‌های تغذیه و زمین CPU متصل کرده و بر توان مصرفی پردازنده در حین اجرای هر دستورالعمل نظارت می‌کند. استنتاج کلید رمز از این اطلاعات بسیار ساده خواهد بود. برای خنثی کردن این نوع از رمزشکنی، می‌توان الگوریتم رمزنگاری را به زبان اسمبلی برنامه‌نویسی کرده و مطمئن شد که توان مصرفی پردازنده مستقل از بیتهای کلید و همچنین «کلیدهای دور» (Round Keys) است.

چهارمین روش، «تحلیل زمانی» (Time Analysis) است. الگوریتمهای رمزنگاری سرشار از دستورات if then else هستند که در حقیقت بیتهایی از کلیدهای هر دور را آزمایش می‌کنند. هرگاه دستورات پس از then و else از لحاظ زمان اجرا، متفاوت باشند، با پایین آوردن سرعت سیگنال ساعت پردازنده و اندازه‌گیری زمان اجرای هر مرحله، استنتاج کلیدهای هر دور ممکن خواهد بود. وقتی یکایک کلیدهای هر دور از اجرای الگوریتم بدست آمد می‌توان کلید اصلی را محاسبه کرد. تحلیل توان مصرفی و تحلیلهای زمانی را می‌توان بطور همزمان بکار گرفت تا کار کشف کلید رمز ساده‌تر 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 حمله‌ای، مقاوم طراحی نشده باشد، بشکنند.

۳-۸ الگوریتمهای کلید عمومی (Public Key)

همواره توزیع و مبادلهٔ کلید رمز (Key Distribution) یکی از مشکلات سیستمهای 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 بی‌ارزش خواهد شد. رمزشکنها همیشه از روشهایی که در آنها کلید رمزنگاری و رمزگشایی یکسان است (یا از طریق یکدیگر قابل محاسبه هستند) قلباً استقبال می‌کنند. در این روشها بالاخره باید کلیدها بین کاربران سیستم توزیع شود. در همین نقطه به نظر می‌رسد 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 نمی‌توان 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 نمی‌شد 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 ر را برآورده می‌کرد. این نیازها را می‌توان به سادگی به صورت زیر توصیف کرد:

۱. $D(E(P))=P$

۲. استنتاج d (کلید رمزگشایی) از روی e (کلید رمزنگاری) بی‌نهایت مشکل باشد.

۳. E از طریق مکانیزم «حمله با متن‌های انتخابی و شناخته شده» شکسته نشود.

---

۱. یعنی احتمال وقوع یکی از بیتها از ۰/۵ کمتر و دیگری از ۰/۵ بیشتر می‌شود. ـم

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 گیرندهٔ 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 رمزشکن 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 می‌شود. آلیس حتی می‌تواند کلید عمومی [برای رمزنگاری] را در صفحهٔ 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 می‌کنیم. همچنین از نماد D_A به معنای الگوریتم رمزگشایی با پارامتر A یعنی کلید خصوصی آلیس، استفاده می‌نمائیم. باب نیز دقیقاً همین کار را می‌کند، E_B را به صورت عمومی آشکار می‌کند در حالی که D_B را به صورت سرّی نزد خود نگهداری می‌کند.

حال ببینیم مشکل برقراری یک کانال مطمئن بین آلیس و باب که هیچ ارتباط قبلی با هم نداشته‌اند چگونه حل می‌شود. فرض شده کلید رمزنگاری آلیس یعنی E_A و کلید رمزنگاری باب یعنی E_B در فایلهای قابل خواندن [و به صورت آشکار] قرار دارد. آلیس اولین پیام خود یعنی P را می‌گیرد و $E_B(P)$ را محاسبه کرده و نتیجه را برای باب می‌فرستد. باب با اعمال کلید سرّی خود یعنی D_B، آنرا رمزگشایی می‌کند. (به عبارت دیگر $D_B(E_B(P))$ را محاسبه می‌کند.) هیچ شخص دیگری نمی‌تواند از پیام رمزنگاری شده بهره‌برداری کند چراکه سیستم رمزنگاری بسیار قدرتمند فرض شده و استنتاج D_B (کلید رمزگشایی) از کلید رمزنگاری E_B بسیار مشکل و غیرعملی است. برای ارسال پاسخ پیام یعنی R، باب $E_A(R)$ را ارسال می‌کند. حال آلیس و باب می‌توانند 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 می‌کند که هر کاربر دو کلید داشته باشد: یک **کلید عمومی** (Public Key) که تمام دنیا برای ارسال پیام به کاربر، از آن استفاده می‌کنند و یک **کلید خصوصی** (Private Key) که کاربر برای رمزگشایی پیامها بدان احتیاج دارد. از این به بعد، به کرّات از اصطلاحات کلید عمومی و خصوصی استفاده خواهیم کرد تا از «**کلید سرّی**» (Secret Key) که در سیستمهای رمزنگاری با کلید متقارن کاربرد دارد تمیز داده شود.

۸-۳-۱ RSA

تنها کاری که باید انجام شود یافتن الگوریتمی است که سه نیاز اشاره شده را برآورده نماید. به دلیل محاسن بالقوه‌ای که رمزنگاری با کلید عمومی در بردارد، بسیاری از پژوهشگران در این زمینه بشدت فعالیت می‌کنند و تاکنون چندین الگوریتم مناسب تدوین و معرفی شده است. یکی از الگوریتمهای خوب، توسط گروهی در دانشگاه MIT (به سرپرستی ری‌وِست، ۱۹۷۸) ابداع شد. این روش که به نام RSA مشهور است از حرف ابتدایی اسامی مخترعین آن ، ری‌وِست، شامیر و آدِلْمن (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 می‌شود. بسیاری از روشهای عملی امنیت، براساس RSA هستند. بزرگترین اشکال این روش آن است که برای رسیدن به بالاترین درجهٔ امنیت، به کلید رمزی با حداقل ۱۰۲۴ بیت احتیاج است (برخلاف الگوریتمهای با کلید

متقارن ۱۲۸ بیتی) که این موضوع الگوریتم را بسیار کُند می‌کند.

روش RSA بر یک سری از اصول اساسی در نظریهٔ اعداد استوار است. ما چکیدهٔ این روش را در همین جا بیان می‌کنیم؛ برای تفصیل بیشتر از مقالات کمک بگیرید.

۱. دو عدد اول بسیار بزرگ p و q را انتخاب نمائید. (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 گونه‌ای پیدا کنید که $e\times d \bmod Z = 1$ برقرار باشد.

با این پارامترها که پیشاپیش محاسبه می‌شود، آمادهٔ شروع رمزنگاری هستیم: متن اصلی (که به صورت رشته‌ای از بیتها تلقی می‌شود) ابتدا به تعدادی بلوک تقسیم می‌گردد، به نحوی که هر بلوک P در بازهٔ $0 <= P < n$ قرار بگیرد. این کار را با تقسیم متن به بلوکهای k بیتی که در آن k بزرگترین عدد صحیحی است که در رابطهٔ $2^k < n$ صدق می‌کند، انجام بدهید. ($n=p\times q$)

برای رمزنگاری پیام P، $C = P^e \bmod n$ را محاسبه نمائید. 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 (n و e) است و کلید خصوصی (n و d) خواهد بود.

امنیت این روش از آنجا ناشی شده که تجزیهٔ 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 می‌توان توسط الگوریتم اقلیدس d را پیدا کرد. خوشبختانه، ریاضی‌دانها از حدود سیصد سال قبل برای تجزیهٔ اعداد بزرگ [به عوامل اول] تلاش کرده‌اند و شواهد حاکی از آن است که این کار بسیار مشکل می‌باشد.

براساس گزارش **ری‌وست** و گروه ابداع کنندهٔ RSA، تجزیهٔ یک عدد ۵۰۰ رقمی به روش «تکرار و آزمون» حدود 10^{25} سال طول می‌کشد. 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 تجزیهٔ یک عدد ۵۰۰ رقمی امکان‌پذیر شود، قرنها باقی مانده است و در آن زمان هم می‌توان برای امن کردن RSA، p و q را بزرگتر انتخاب کرد!!

یک مثال بسیار سادهٔ آموزشی از عملکرد الگوریتم RSA در شکل ۸-۱۷ نشان داده شده است. در این مثال p را ۳ و q را ۱۱ فرض کرده‌ایم که نتیجه می‌دهد: $n = 33$ و $Z = 20$. یک مقدار مناسب برای d، عدد ۷ است چرا که ۲۰ و ۷ هیچ عامل مشترکی ندارند. طبق این انتخاب باید عدد e را به نحوی پیدا کرد که در رابطهٔ $7\times e \bmod 20 = 1$ صدق نماید، که عدد ۳ را نتیجه می‌دهد. کدهای رمز شدهٔ C برای یک پیام مثل P از رابطهٔ $C = P^3 \bmod 33$ بدست می‌آید. متن رمز شده را می‌توان 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 می‌تواند شامل یک تک کاراکتر باشد. نتیجهٔ رمز، یک جانشینی تک‌کاراکتری خواهد بود که بهیچوجه مؤثر نیست. در عوض اگر p و q را اعدادی در حدود 2^{512} انتخاب کرده بودیم، عدد n را از مرتبهٔ 2^{1024} می‌داشتیم و هر بلوک از متن می‌توانست 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 (mod 33)	C^7	C^7 (mod 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[1]، هرگاه ورودیهای مشابه به الگوریتم اعمال شود، نتیجهٔ 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 بکمک آن کلید جدیدی تولید و ردّ وبدل گردد و پس از آن از الگوریتم متقارنی مثل «DES سه‌گانه» یا AES استفاده شود.[2] زیرا استفاده از RSA برای رمزنگاری حجم عظیم اطلاعات، بسیار کُند 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 شدهٔ کلید عمومی است. در حقیقت اولین الگوریتم کلید عمومی، الگوریتم Knapsack «**کوله‌پشتی**» است. (مِرکِل و هِلمن؛ ۱۹۷۸) ایدهٔ 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 محرمانهٔ 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 سرّی است. در ابتدا گمان می‌رفت که با افزودن مجموعه‌ای از محدودیتها، استخراج فهرست اشیاء درون کیسه با فرض در اختیار داشتن وزن کلّ، در عمل غیرممکن خواهد بود و می‌تواند پایهٔ 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 نسخهٔ جدید را بشکند هزار دلار پیشنهاد کرد. رونالد ری‌وست (Ronald Rivest یا همان R در RSA) سریعاً نسخهٔ جدید را نیز شکست و صاحب این جایزه شد. مرکل 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 مبادلهٔ یک کلید مشترک و تصادفی کاربرد دارد و پس‌از این مرحله از الگوریتم رمزنگاری متقارن استفاده می‌شود. -م

یکی دیگر از شماهای کلید عمومی، بر پیچیدگی محاسبهٔ لگاریتم گسسته (Discrete Logarithm) بنا نهاده شده است. الگوریتمهایی که از این قاعده استفاده کرده‌اند توسط El Gamal (۱۹۸۵) و Schnorr (۱۹۹۱) ابداع شده‌اند.

چند شمای دیگر نیز وجود دارد؛ مثلاً یک دسته از روشها مبتنی بر منحنی‌های بیضوی هستند (Menezes and Vanstone, 1993) ولیکن دو دسته از مهمترین الگوریتمهای کلید عمومی، براساس پیچیدگی و دشواری تجزیهٔ اعداد اول بسیار بزرگ یا محاسبهٔ لگاریتم گسستهٔ اعداد بزرگ بنا شده‌اند. این مسائل حقیقتاً لاینحل به نظر می‌رسند و ریاضی‌دانها سالها بر روی آنها کار کرده‌اند ولی هیچ راهی برای رخنه در آن نیافته‌اند.

٤.٨ 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 می‌شود و طبعاً تصویر این اسناد ارزش قانونی ندارد. برای سیستمهای پیام‌رسان کامپیوتری که جانشین گردش کاغذ و اسناد خطی شده‌اند نیز باید روشی پیدا شود تا بتوان آنها را به گونه‌ای امضاء کرد که هرگز قابل جعل نباشد.

مسئله ابداع یک روش جایگزین به جای امضاهای دستنویس یکی از موضوعات دشوار به حساب می‌آید. در اصل به سیستمی نیاز است که براساس آن یک طرف بتواند پیامی امضاء شده را برای طرف دیگر بفرستد به گونه‌ای که شرایط زیر به درستی احراز شود:

۱. گیرنده بتواند هویت شخص فرستندهٔ 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 تُن طلا می‌دهد (!) 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 می‌کند و باید هزینهٔ 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).

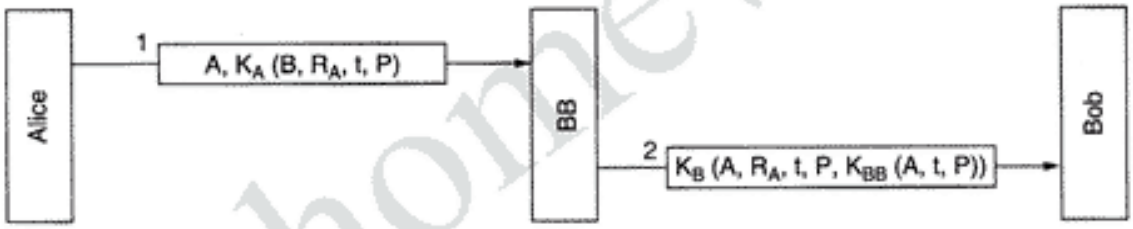
نیاز دوم بدان جهت حیاتی است که از بانک در مقابل کلاه‌برداری محافظت نماید. فرض کنید بانک، یک تن طلای سفارش داده شده را تهیه کرده ولی به ناگاه قیمت طلا بشدت سقوط می‌کند. مشتری بخت برگشته ممکن است ادعا کند که هرگز سفارش خرید یک تن طلا نداده است. وقتی بانک پیام ارسالی را به دادگاه تسلیم می‌کند، ممکن است مشتری ارسال آن را تکذیب نماید. داشتن این ویژگی که هیچیک از طرفینِ یک قراردادِ امضاء شده، نتوانند مفاد آن را تکذیب کنند اصطلاحاً "nonrepudiation" یا «**غیرقابل انکار بودن**» پیام نامیده می‌شود. الگوهای امضای دیجیتال که آنها را معرفی خواهیم کرد این قابلیت را فراهم می‌سازند.

سومین نیاز برای حفاظت از مشتری در مقابل کلاه‌برداری است مثلاً در سناریوی قبل وقتی قیمت طلا پس از خرید بشدت افزایش می‌یابد شاید بانک بخواهد پیام ارسالی مشتری را دستکاری کرده و وزن طلای سفارشی را از یک تن به یک شمش کاهش بدهد! در این سناریوی کلاه‌برداری، بانک سود باقیماندهٔ طلا را برای خود بر می‌دارد!!

۸-۴-۱ امضاهای دیجیتالی باکلید متقارن

یکی از روشهای ساماندهی امضاهای دیجیتالی آنست که یک مرکز معتبر و مجازِ گواهی امضاء داشته باشیم که همه را می‌شناسد و مورد اعتماد همه نیز هست؛ آن را اصطلاحاً BB (برادر بزرگتر یا Big Brother) می‌نامیم. هر کاربر برای خود یک کلید رمز سرّی (Secret Key) انتخاب کرده و شخصاً به ادارهٔ BB مراجعه و آن را ثبت می‌نماید. بدین ترتیب کاربری مثل آلیس، فقط خودش و BB کلید سرّی و توافق شدهٔ K_A را می‌دانند؛ بهمین روال، دیگر کاربران نیز کلید خودشان را در BB ثبت می‌نمایند.

وقتی آلیس بخواهد پیام امضاء شدهٔ خود یعنی P را برای کارپرداز بانک خود (یعنی باب) بفرستد، $K_A(B,R_A,t,P)$ را تولید می‌کند که در آن B، مشخصهٔ 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 $K_A(B,R_A,t,P)$ نتیجهٔ رمزنگاری مجموعهٔ این چهار آیتم توسط کلید سرّی آلیس یعنی K_A است. سپس او این داده‌های رمز شده را طبق شکل ۸-۱۸ برای BB می‌فرستد. BB متوجه می‌شود که پیام از آلیس است لذا آن را با کلید سرّی آلیس رمزگشایی می‌کند و به نحوی که در شکل نشان داده شده آنرا مجدداً رمز کرده و برای باب می‌فرستد. پیام ارسالی به باب شامل اصل پیام آلیس و یک پیام امضاء شدهٔ $K_{BB}(A,t,P)$ است. حال باب می‌تواند 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 می‌افتد؟ طبعاً هر کسی می‌تواند از دیگری ادعای خسارت کند! (حداقل در ایالات متحده این گونه است!) سرانجام وقتی مورد در دادگاه مطرح می‌شود و آلیس قویاً پیام ارسالی خود به باب را تکذیب می‌کند، قاضی از باب می‌پرسد که چگونه ادعا می‌کند پیام ارسالی آلیس حقیقتاً توسط خود آلیس ارسال شده و از شخص ثالثی مثل ترودی نیست. باب ادعا می‌کند 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 می‌کند که این همان پیامی است که توسط BB امضاء شده است و ارسال پیام P توسط آلیس به باب را اثبات می‌کند. قاضی از BB (که مورد اعتماد همه است) می‌خواهد که این مدرک را رمزگشایی کند. وقتی BB صحت ادعای باب را تائید 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 ترودی 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 «**مُهر زمان**» (Time Stamp) استفاده می‌شود. به علاوه، باب می‌تواند 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 می‌تواند پیامهای قدیمی را حذف کند. برای پیشگیری از تکرار آنی و بلادرنگ پیام، باب

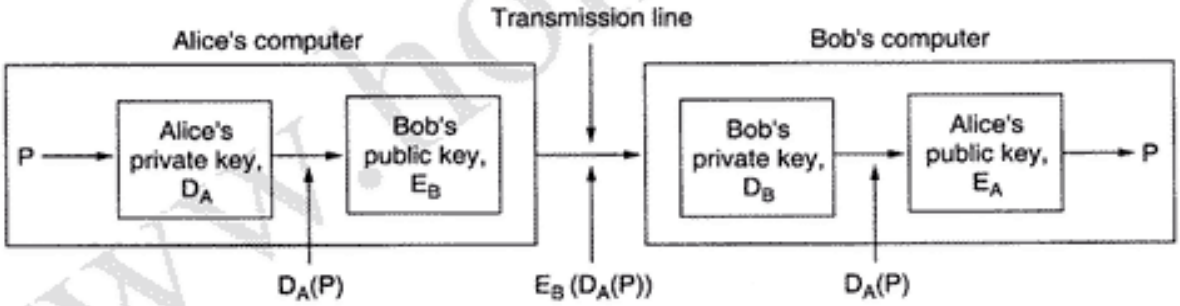
R_A را بررسی می‌کند و اگر پیامی با R_A تکراری دریافت گردد حذف می‌شود. اگر باب از این دو مورد مطمئن شد می‌تواند فرض کند که این تقاضا معتبر و جدید است.

۸-٤-۲ امضاهای باکلید عمومی

مشکل ساختاری در بکارگیری 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. منطقی‌ترین کاندیداهای راه‌اندازی مرکز گواهی امضاء ، دولت، بانکها، سازمانهای حسابرسی و کانون وکلاء هستند. متأسفانه هیچیک از این سازمانها مورد اعتماد و وفاق عموم شهروندان نیستند. بنابراین، امکان امضای مستندات به گونه‌ای که به هیچ مرکز گواهی امضاء نیاز نباشد، مورد بسیار جالبی است.

خوشبختانه، رمزنگاری با کلید عمومی می‌تواند در این زمینه نقش بسیار مؤثر و مثبتی ایفاء کند. فرض را بر آن می‌گذاریم که الگوریتمهای رمزنگاری و رمزگشایی دارای این خصوصیت است که $E(D(P))=P$ و همچنین $D(E(P))=P$.[1] (RSA دارای این ویژگی هست و چنین فرضی دور از واقعیت نخواهد بود.) با فرض وجود این ویژگی، آلیس می‌تواند متن رمز و امضا شدهٔ P را به صورت $E_B(D_A(P))$ برای باب بفرستد.[2] دقت داشته باشید که آلیس فقط و فقط خودش کلید خصوصی خود یعنی D_A را می‌داند؛ 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 می‌کند، ابتدا آن را با کلید خصوصی خود رمزگشایی کرده و $D_A(P)$ را مطابق با شکل ۸-۱۹ بدست می‌آورد. او این متن را در جای امنی ذخیره می‌کند [برای استفادهٔ احتمالی در دادگاه] و سپس کلید عمومی آلیس یعنی E_A را بر روی آن اعمال کرده و متن اصلی را بدست می‌آورد.



شکل ۸-۱۹. امضاهای دیجیتالی با استفاده از رمزنگاری کلید عمومی.

برای آن که ببینیم این ساختار چگونه نیازهای امضای دیجیتالی را برآورده می‌کند، فرض کنید بعداً آلیس، ارسال پیام P به باب را انکار کند. وقتی این مورد در دادگاه مطرح می‌شود، باب هر دو پیام P و $D_A(P)$ را به دادگاه تسلیم می‌کند. قاضی می‌تواند 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 نمی‌توانسته $D_A(P)$ 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، بکمک کلید خصوصی قابل رمزگشایی است و بالعکس متن رمزنگاری شده با کلید خصوصی، بکمک 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 دیجیتالی، الگوی بسیار جالبی است ولی از مشکلاتی رنج می‌برد که ناشی از خود الگوریتم نیست بلکه مربوط به محیطی است که در آن عمل می‌کند. فقط زمانی باب می‌تواند ثابت کند که پیام ارسالی متعلق به آلیس است که D_A (کلید خصوصی آلیس) محرمانه و سرّی باقی بماند. اگر آلیس کلید خصوصی خود را لو بدهد، هیچ دلیل محکمه‌پسندی باقی نخواهد ماند زیرا هر کسی حتی خود باب می‌توانسته چنین پیامی را ارسال نماید.

به عنوان مثال، مشکل زمانی بروز می‌کند که باب دلال سهام آلیس باشد. آلیس به باب می‌گوید که برایش سهام یا اوراق بهادار خاصی را بخرد. بلافاصله پس از این سفارش قیمتها بشدت سقوط می‌کند. آلیس برای آن که بتواند پیام سفارش خود به باب را انکار کند، سریعاً با پلیس تماس گرفته و ادعا می‌کند که خانه‌اش 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 خانه‌اش 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 دوره‌ای ایدهٔ بسیار خوبی هم باشد. اگر در دادگاهی که بعداً تشکیل می‌شود، 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 می‌شود که P بدست نمی‌آید. باب در چنین شرایطی حسابی گیج و درمانده خواهد شد.

در اصل، از هر الگوریتم رمزنگاری با کلید عمومی می‌توان برای امضاهای دیجیتالی بهره گرفت ولیکن استاندارد غیررسمی صنعت، رمزنگاری RSA است و بسیاری از محصولات امنیتی از آن بهره گرفته‌اند. با این وجود در سال ۱۹۹۱، NIST استاندارد جدیدی برای امضاهای دیجیتال به نام DSS[1] پیشنهاد کرد که مبتنی بر الگوریتم رمزنگاری کلید عمومی El Gamal بود. الگوریتم El Gamal امنیت ذاتی خود را از دشوار بودن محاسبهٔ لگاریتم گسسته (به جای تجزیهٔ اعداد بزرگ به عوامل اول) کسب کرده است.

طبق معمول وقتی دولت سعی در تحمیل یک استاندارد رمزنگاری می‌کند، غوغایی پیرامون آن به پا می‌شود. از DSS در موارد زیر انتقاد شده است:

۱. بسیار محرمانه و سرّی است (NSA در این پروتکل از الگوریتم El Gamal استفاده کرده است.)
۲. بسیار کُند عمل می‌کند. (برای بررسی امضاء ده تا چهل برابر از RSA کُندتر عمل می‌نماید.)
۳. خیلی جدید است. (الگوریتم El Gamal هنوز به قدر کافی تحلیل و بررسی نشده است.)
۴. خیلی ناامن به نظر می‌رسد. (از کلید ۵۱۲ بیتی با طول ثابت استفاده شده است.)

در بازبینی‌های بعدی، چهارمین مورد اشکال با افزایش طول کلید به ۱۰۲۴ بیت، برطرف شد ولیکن همچنان دو اشکال اول باقی است.

۳-٤-۸ خلاصهٔ 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 می‌دهند: «احراز هویت» و «سرّی ماندن پیام». بسیاری از اوقات احراز هویت لازم است در حالی که به سرّی ماندن محتوای پیام نیازی نیست. سیستمهایی که فقط سرویس «احراز هویت» ارائه می‌کنند و در مورد مبادلهٔ سرّی

---

۱. Digital Signature Standard

پیام کاری انجام نمی‌دهند (گذشته از سرعت و کارآیی بیشتر) راحتتر مجوز صدور می‌گیرند.[۱] در زیر الگویی را برای «احراز هویت» بررسی می‌کنیم که نیازی به رمزنگاری کل پیام ندارد.

این ساختار مبتنی بر تابع Hash یک مرحله‌ای است (One-Way Hash Function) که یک قطعهٔ طولانی از متن اصلی را گرفته و یک رشتهٔ بیتی با طول ثابت از آن محاسبه و استخراج می‌کند. حاصل این **«تابع درهم‌سازی»**، اغلب به نام **«خلاصهٔ پیام»**[۲] یا MD[۳] شناخته می‌شود و دارای چهار ویژگی زیر است:

۱. با داشتن P مفروض، محاسبهٔ MD(P) بسیار ساده است.

۲. با داشتن MD(P)، عملاً پیدا کردن P غیرممکن است.

۳. با داشتن P مفروض نمی‌توان یک P' پیدا کرد به نحوی که MD(P) = MD(P')

۴. تغییر در ورودی حتی به اندازهٔ 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 می‌کنند] براساس تمام بیتهای درهم شدهٔ پیام محاسبه شود.

محاسبهٔ **«خلاصهٔ پیام»** براساس قطعه‌ای از یک متن، بسیار سریعتر از رمزنگاری آن توسط الگوریتمهای کلید عمومی است و بدین ترتیب محاسبهٔ MD می‌تواند برای افزایش سرعت الگوریتمهای امضای دیجیتالی مؤثر واقع شود. برای آن که ببینیم این ساختار چگونه کار می‌کند 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 رشتهٔ نسبتاً کوتاه MD(P) را بدست می‌آورد 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 می‌کند.

هرگاه یک دعوای حقوقی بوجود بیاید، باب می‌تواند 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 می‌تواند صحت پیام P را تائید کند. همچنین از آنجایی که عملاً باب قادر نخواهد بود که پیامی جعلی پیدا کند که رشتهٔ Hash آن [حاصل MD]، معادل با رشتهٔ Hash پیام واقعی باشد لذا باب نخواهد توانست دادگاه را در خصوص صحت پیام جعلی خود متقاعد کند. با استفاده از روش **«خلاصهٔ پیام»** هم در زمان رمزنگاری کل پیام و هم در هزینهٔ انتقال پیام بر روی شبکه صرفه‌جویی خواهد شد.

پیاده‌سازی روش **«خلاصهٔ پیام»** به نحوی که در شکل ۸-۲۰ نشان داده شده، براساس سیستمهای کلید عمومی نیز عملیست. در اینجا آلیس ابتدا خلاصهٔ پیام خود را محاسبه می‌نماید، سپس آن را به جای اصل پیام امضاء کرده و نهایتاً اصل پیام [رمز نشده] را به همراه MD رمز شده برای باب می‌فرستد.[۴] اگر شخص ثالثی مثل ترودی P را دستکاری کند، باب وقتی MD(P) را محاسبه می‌کند متوجه جعلی بودن پیام خواهد شد.

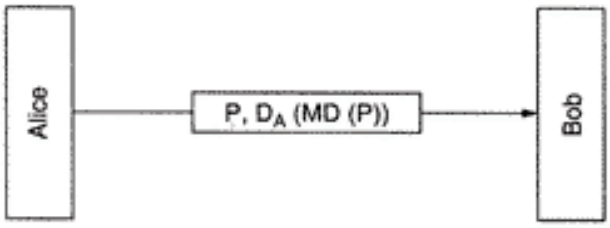
---

۱. به خاطر داشته باشید که در بسیاری از کشورها صدور نرم‌افزارهای رمزنگاری یا سیستمهای مبتنی بر مبادلهٔ سرّی پیام کلاً ممنوع است و مجازاتهای زندان (در فرانسه تا ۱۳ سال) دارد!! -م.

۲. اگر بخواهیم در دو جمله «خلاصهٔ پیام» یا همان Message Digest را بیان کنیم عبارتست از : «یک رشتهٔ رمزی کوتاه که از درون یک پیام استخراج و پس از رمزنگاری بهمراه پیام رمزنشده ارسال می‌شود. حتی اگر یک بیت از پیام اصلی دستکاری شود رشتهٔ خلاصهٔ پیام جدید با این رشته یکسان نخواهد بود.». * رشته Hash، در حقیقت همان رشته‌ای است که از درون پیام استخراج و به صورت رمز همراه آن ارسال می‌شود 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 می‌باشد. -م



شکل ۸-۲۰. امضاهای دیجیتالی با استفاده از «خلاصهٔ پیام».

MD5

توابع متنوعی برای تولید «**خلاصهٔ پیام**» پیشنهاد شده است که رایج‌ترین آنها MD5 (ری وست، ۱۹۹۲) و SHA-1 (NIST ، ۱۹۹۳) است. MD5 پنجمین روش پیاده سازی «**خلاصهٔ پیام**» است که همگی توسط رونالد ری‌وست (سرپرست گروه ابداع کنندهٔ RSA) طراحی شده‌اند. این روش با درهم فشردن تمام بیتها، طبق رابطه‌ای بسیار پیچیده، MD را به نحوی محاسبه می‌کند که یکایک بیتهای خروجی از یکایک بیتهای ورودی [متن اصلی] تأثیر می‌پذیرند. بطور کاملاً مجمل، این روش ابتدا آنقدر بیتهای بی‌اهمیت به متن اصلی اضافه می‌کند که طول آخرین بلوک، فقط و فقط ۴۴۸ بیت باشد. سپس طول واقعی پیام در یک فیلد ۶۴ بیتی به پیام اضافه می‌شود تا نهایتاً تعداد صحیحی از بلوکهای ۵۱۲ بیتی بدست آید. [یعنی طول کل متن اصلی ضریبی از ۵۱۲ شود.] سپس در تکمیل مراحل پیش‌پردازش، یک بافر ۱۲۸ بیتی با یک عدد ثابت، مقداردهی اولیه می‌گردد.

حال محاسبات آغاز می‌شود. در هر دور از محاسبه یک بلوک ۵۱۲ بیتی از متن ورودی استخراج و طبق یک رابطهٔ خاص با بافر ۱۲۸ بیتی ادغام [مخلوط] می‌شود. برای اندازه‌گیری بهتر، یک جدول که با مقادیر تابع Sin مقداردهی شده نیز در آن تزریق می‌شود. استفاده از تابع شناخته شده‌ای مثل Sin از آن جهت نبوده که مقادیر آن تصادفی‌تر از مولد اعداد تصادفی است بلکه فقط بدین موضوع کمک کرده که جلوی هر گونه سوءظن در خصوص طراحی یک رخنهٔ 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. ری‌وست (مخترع MD-5) می‌خواست از هر گونه شایعه‌ای جلوگیری کند. در ادامه، چهار دور پردازش بر روی هر بلوک ورودی انجام می‌شود. این فرآیند آنقدر تکرار می‌شود تا محاسبهٔ تمام بلوکهای ورودی خاتمه یابد. محتوای بافر ۱۲۸ بیتی ، Message Digest یا همان خلاصهٔ «پیام» را تشکیل می‌دهد.

اکنون بیش از یک دهه است که MD5 معرفی شده و بسیاری از افراد سعی در حمله به آن داشته‌اند. 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 مرحلهٔ باقیمانده که حصارهای نهایی MD5 محسوب می‌گردند شکسته شود، MD5 فرو می‌پاشد ولی علیرغم این موضوع، در زمان نوشتن این کتاب، MD5 هنوز هم به طور گسترده‌ای رایج است.

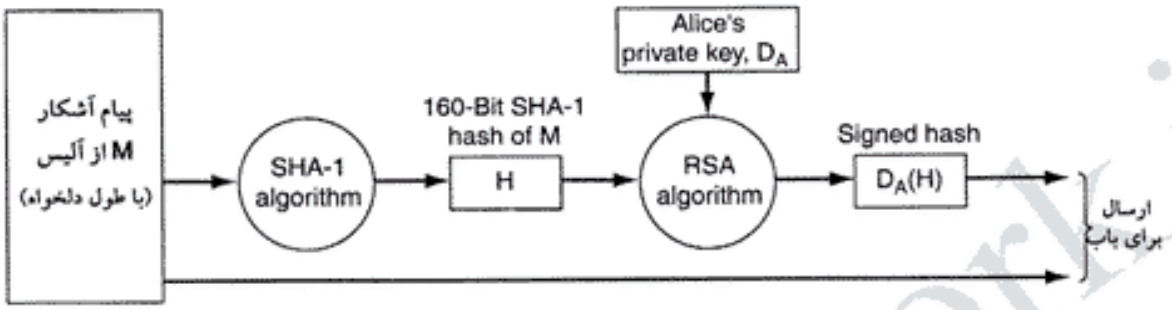
SHA-1[1]

یکی دیگر از توابع مهم «**خلاصهٔ پیام**» تابع SHA-1 است که توسط NSA (آژانس امنیت ملی در آمریکا) ابداع شده و توسط NIST (ادارهٔ استانداردها و فناوریهای مدرن آمریکا) به شمارهٔ FIPS 180-1 به ثبت رسیده است. همانند MD5 ، SHA-1 نیز بلوکهای ورودی را در قالب بلوکهای ۵۱۲ بیتی پردازش می‌کند ولی برخلاف MD5 یک رشتهٔ

---

۱. Secure Hash Algorithm

۱۶۰ بیتی (به عنوان خلاصهٔ پیام یا Message Digest) تولید می‌کند. روش عمومی ارسال یک پیام غیرمحرمانه 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 می‌شود تا یک رشتهٔ «درهم شدهٔ» ۱۶۰ بیتی[1] بدست آید. سپس آلیس این رشتهٔ ۱۶۰ بیتی را با استفاده از کلید خصوصی RSA خود رمز (امضاء) می‌کند و نهایتاً پیام اصلی رمز نشده و رشتهٔ Hash امضاء شده (رمز شده) را برای باب می‌فرستد.



شکل ۸-۲۱. استفاده از SHA-1 و RSA برای امضای پیامهای غیرمحرمانه.

پس از دریافت، باب (بدون توجه به امضای پیام) شخصاً رشتهٔ Hash را برای پیام محاسبه می‌کند و از طرف دیگر با کلید عمومی آلیس، Hash ارسالی را بازگشایی می‌کند. اگر این دو رشته بیت با هم یکسان بودند پیام ارسالی معتبر و حقیقی است. از آنجایی که شخص ثالثی مثل تروودی قادر نیست پیام ارسالی را در حین گذر از شبکه تغییر داده و برای آن Hash جدید تولید کند [چون کلید خصوصی آلیس را نمی‌داند] لذا باب به سادگی هر گونه تغییری را که توسط تروودی 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 هزینهٔ محاسباتی بسیار پایین، تضمین می‌کند که هر گونه دستکاری در پیامها حین گذر از شبکه، با احتمال بسیار بالایی کشف شود.

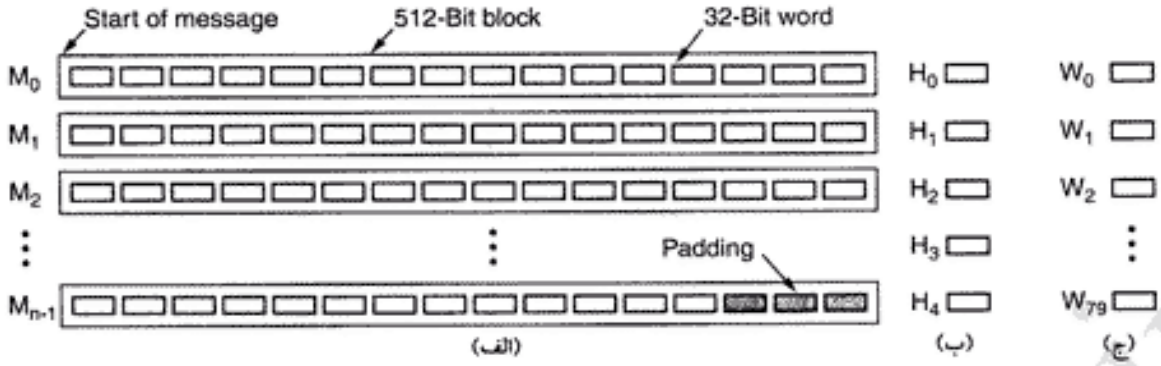
حال اجمالاً ببینیم که SHA-1 چگونه کار می‌کند. SHA-1 کار را با اضافه کردن یک بیت ۱ در انتهای پیام و سپس تعدادی بیت صفر در ادامهٔ آن آغاز می‌کند تا طول پیام ضریبی از ۵۱۲ بیت شود. سپس یک عدد ۶۴ بیتی که طول حقیقی پیام (قبل از اضافه کردن بیتهای صفر و یک) را نشان می‌دهد با ۶۴ بیت انتهایی OR می‌شود. در شکل 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 (پایین‌ترین سطر) مشاهده می‌شود. در کامپیوترها این ساختار شبیه به ماشینهای Big-Endian[2] مثل SPARC است ولی فارغ از نوع ماشین، SHA-1 داده‌های اضافی (Pad) را به انتهای پیام می‌چسباند. در خلال محاسبه، SHA پنج متغیر ۳۲ بیتی H_0 تا H_4 را در حافظه نگهداری می‌کند که رشتهٔ Hash [۱۶۰ بیتی] در آنجا جمع‌آوری و محاسبه می‌شود. در ادامه بلوکهای ۵۱۲ بیتی M_0 تا M_{n-1} به ترتیب پردازش می‌شوند. برای بلوک جاری ابتدا ۱۶ کلمهٔ ۳۲ بیتی [معادل ۵۱۲ بیت] به نحوی که در شکل ۸-۲۲-ج نشان داده شده در یک آرایه کمکی ۸۰ کلمه‌ای به نام W کپی می‌شود. ۶۴ کلمهٔ باقیمانده از آرایهٔ W طبق فرمول زیر محاسبه و پر می‌شوند:

$$W_i = S^1 (W_{i-3} \text{ XOR } W_{i-8} \text{ XOR } W_{i-14} \text{ XOR } W_{i-16}) \qquad (16 <= i =<79)$$

---

۱. 160 Bits SHA-1 Hash

۲. ماشینهای Big Endian ماشینهایی هستند که برای ذخیره سازی کلمات ۲، ۴ یا ۸ بایتی در حافظه ابتدا بایت پرارزش را و سپس بایتهای کم ارزش را ذخیره می‌کنند. ماشینهای سری x86 برعکس عمل می‌کنند.



شکل ۸-۲۲. (الف) به پیام آنقدر داده‌های زائد اضافه می‌شود تا طول آن ضریبی 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 کلمهٔ ۳۲ بیتی W و به اندازهٔ 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 شبه‌کد C نشان داد:

```
for (i = 0 ; i<80 ; i ++) {
    temp = S^5(A)+f_i(B,C,D)+E+W_i+K_i;
    E=D;  D=C  C=S^30(B);  B=A;  A=temp;
}
```

در کُد بالا K_iها ثابتهایی هستند که در استاندارد تعریف شده‌اند. تابع مخلوط‌سازی[1] f_i نیز به ترتیب زیر تعریف شده است:

$f_i(B,C,D)$=(B AND C) OR (NOT B AND D) $(0 \le i \le 19)$
$f_i(B,C,D)$=B XOR C XOR D $(20 \le i \le 39$
$f_i(B,C,D)$=(B AND C) OR (B AND D) OR (C AND D) $(40 \le i \le 59)$
$f_i(B,C,D)$=B XOR C XOR D $(60 \le i \le 79$

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. آرایهٔ W با بلوک جدید مقداردهی می‌گردد در حالی که مقادیر H_0 تا H_4، از مرحلهٔ 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 کلمهٔ ۳۲ بیتی H_0 تا H_4 در آرایهٔ H به عنوان رشتهٔ ۱۶۰ بیتی Hash بدست می‌آید. برنامهٔ کامل SHA-1 به زبان C در RFC 3174 ارائه شده است.

نسخهٔ جدید SHA-1 با رشته‌های ۲۵۶، ۳۸۴ و ۵۱۲ بیتی Hash تحت مطالعه و پیاده‌سازی است.

۸-۴-۴ حمله 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 «**حملهٔ روز تولد**» تعداد عملیات آزمایش $2^{m/2}$ حالت است؛ روشی که توسط Yuval (۱۹۷۹) در مقاله‌ای به

۱. Mixing Function

نام "How to Swindle Rabin" منتشر شد. ایدهٔ این نوع حمله از تکنیکی منشاء می‌گیرد که اساتید ریاضی در کلاس درس احتمال، بکار می‌برند. 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 $\frac{۱}{۲}$ باشد؟» بسیاری از دانشجویان انتظار دارند جواب این مسئله بیش از ۱۰۰ باشد، ولیکن تئوری احتمال می‌گوید که این تعداد ۲۳ است. بدون یک تحلیل دقیق و به صورت ذهنی می‌توان بدین گونه حساب کرد که با ۲۳ نفر $\frac{۲۳\times۲۲}{۲}$=۲۵۳ زوج مختلف خواهیم داشت که هر یک از این زوجها با احتمال $\frac{۱}{۳۶۵}$ در یک روز سال به دنیا آمده‌اند. [بنابراین احتمال آن که یک زوج در یک روز به دنیا آمده باشد $\frac{۲۵۳}{۳۶۵}$ و بیش از ۰/۵ 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 نظایر 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. نتیجهٔ فوق بدان معناست که شکستن [۱] یک خلاصهٔ پیام (Message Digest) ۶۴ بیتی با تولید $۲^{۳۲}$ پیام مختلف و مقایسهٔ آنها امکان‌پذیر است. [حدود چهار میلیارد حالت مختلف]

بگذارید یک مثال عملی را بررسی کنیم. دانشکدهٔ علوم کامپیوتر در یک دانشگاه به یک عضو هیئت علمی نیاز دارد که برای این موقعیت دو نفر به نامهای «تام» و «دیک» نامزد شده‌اند. تام دو سال زودتر استخدام شده و به همین دلیل زودتر از دیک برای مصاحبه دعوت می‌شود و در این صورت «دیک» این موقعیت را از دست خواهد داد. تام می‌داند که «مرلین»، مدیر گروه دانشکده، نظر مساعدی به کار او دارد، به همین دلیل از او خواهش می‌کند یک توصیه‌نامه برای او خطاب به «دین» (Dean) مسئول گزینش تام، بنویسد. تمام نامه‌ها محرمانه خواهد ماند.

«مرلین» از منشی خود «الن» می‌خواهد تا طی نامه‌ای خطاب به «دین» آنچه را مورد نظر اوست بنویسد تا پس از حاضر شدن، آنرا بررسی کرده و از طریق یک رشتهٔ ۶۴ بیتی Message Digest امضاء نماید. پس از آن «الن» می‌تواند نامه را از طریق پست الکترونیکی ارسال کند. از بخت بدِ تام، «الن» متمایل به انتخاب «دیک» است لذا ابتدا نامهٔ زیر را با ۳۲ گزینهٔ مختلف می‌نویسد (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 خلاصهٔ پیام به معنای آنست که: رمزشکن یک رشتهٔ خلاصهٔ پیام (Message Digest) داشته باشد و بتواند یک پیام معادل با آن پیدا کند. ـم.

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 نامهٔ فوق، نامهٔ دومی (با مضمون بد) به صورت زیر می‌نویسد:

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. Unfotunately, 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 برنامه‌ریزی می‌کند تا در خلال شب تمام 2^{32} حالت مختلف «خلاصهٔ پیام» (Message Digest) را برای هر دو نامه محاسبه کند.[1] زمانی او موفق است که دو پیام فوق دارای «خلاصهٔ پیام» یکسان شوند. اگر نشد او چند گزینهٔ دیگر به پیام دوم می‌افزاید و در پایان هفته مجدداً آنها را می‌آزماید. فرض کنید که بالاخره یک مورد تطابق پیدا می‌کند. نامهٔ خوب را A و نامهٔ بد را B بنامید.

«الن» نامهٔ A را جهت تائید برای «مرلین» می‌فرستد و نامهٔ B را به صورت محرمانه نزد خود نگاه می‌دارد و آنرا به هیچ کس نشان نمی‌دهد. «مرلین» محتوای نامهٔ A را تائید و آنرا از طریق یک «خلاصهٔ پیام» شصت و چهار بیتی امضاء می‌کند. «الن» مستقلاً به جای نامهٔ اول، نامهٔ دوم (B) را می‌فرستد، در حالی که امضای دو نامه یکی است.

پس از دریافت نامه و «خلاصهٔ پیام» امضاء شدهٔ آن، «دین» (مسئول گزینش) الگوریتم MD را بر روی نامهٔ دوم اعمال کرده و می‌بیند که با آنچه که «مرلین» امضاء کرده مطابقت دارد فلذا [براساس مضمون نامه] «تام» را بیرون می‌اندازد. «دین» متوجه نخواهد شد که «الن» به گونه‌ای برنامه‌ریزی کرده که دو نامه با MD مشابه ایجاد شود و نامهٔ جعلی دوم را ارسال نموده است. (پایان اختیاری داستان: «الن» موضوع را به «دیک» می‌گوید. «دیک» را ترس فرا می‌گیرد و دوستی خود را با او بهم می‌زند. «الن» بشدت ناراحت می‌شود و نزد «مرلین» اعتراف می‌کند. «مرلین» موضوع را به «دین» اطلاع می‌دهد. «تام» منصب مورد نظر را بدست می‌آورد!!!!!)

اِعمال «**حملهٔ روز تولد**» بر علیه MD5 بسیار مشکل است و حتی اگر در ثانیه یک میلیارد Message Digest متفاوت تولید شود محاسبهٔ MD برای 2^{64} حالت مختلف برای دو نامه حدود ۵۰۰ سال طول می‌کشد و ضمناً

---

۱. ترکیبات مختلف کلماتی که در متن بصورت ایتالیک نمایش داده شده‌اند در مجموع حدود 2^{32} حالت خواهد بود. -م.

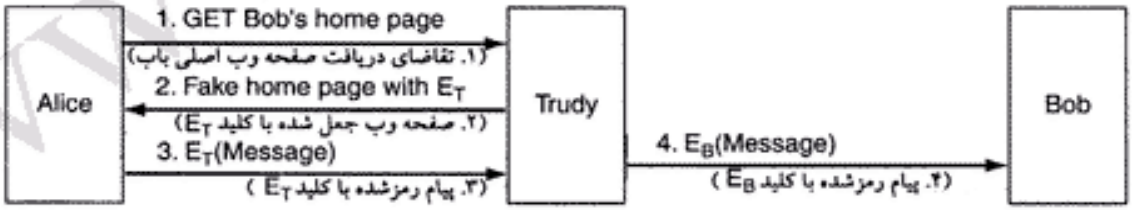
رسیدن به MD مشابه برای دو نامه تضمین شده هم نیست. البته با داشتن ۵۰۰۰ کامپیوتر که بطور موازی با هم کار کنند این زمان به پنج هفته کاهش خواهد یافت. SHA-1 در این مورد بهتر است چرا که رشته Message Digest طولانی‌تری ایجاد می‌کند.

۸-۵ مدیریت کلیدهای عمومی

«رمزنگاری با کلید عمومی» این امکان را فراهم آورده تا افرادی که از قبل بر روی یک کلید مشترک توافق نداشته‌اند، بتوانند با یکدیگر محاوره و مبادلهٔ 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 صحّت پیامهای دریافتی را میسر ساخته است.[1]

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 مبادلهٔ اطلاعات را آغاز کنند؟ ساده‌ترین راه حل (یعنی قرار دادن کلید عمومی در وب‌سایت شخصی خود) به دلیل زیر کارآمد نخواهد بود: فرض کنید که آلیس می‌خواهد کلید عمومی باب را بر روی وب‌سایت او نگاه کند. چه کاری انجام می‌دهد؟ او URL وب‌سایت را تایپ می‌کند. مرورگر (Browser) او از DNS برای پیدا کردن آدرس سایت شخصی باب کمک گرفته و مطابق شکل ۸-۲۳ برای دریافت این صفحهٔ وب، فرمان GET [از فرامین HTTP] را صادر می‌کند. متأسفانه ترودی در حال استراق سمع این تقاضا است و در پاسخ به آن سریعاً یک صفحهٔ وب جعلی بر می‌گرداند که احتمالاً کپی صفحهٔ وب باب است با این تفاوت که کلید عمومی باب با کلید عمومی ترودی عوض شده است. [این کلید جعلی را E_T فرض کنید.] وقتی آلیس اولین پیام خود را با E_T رمز می‌کند و به گمان خود آن را برای باب می‌فرستد ترودی آن را گرفته و با کلید خصوصی خود رمزگشایی می‌کند و پس از بهره‌برداری، برای آن که این موضوع کشف نشود پیام را با کلید واقعی باب مجدداً رمز کرده و برای او می‌فرستد؛ بدون آن که باب آگاه باشد که پیام دریافتی او استراق سمع شده است. از آن بدتر، ترودی 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 می‌شوند.



شکل ۸-۲۳. روشی که بر اساس آن ترودی می‌تواند رمزنگاری کلید عمومی را با اشکال مواجه کند.

۸-۵-۱ گواهینامه‌ها (Certificates)

در اولین تلاش برای توزیع مطمئن کلیدهای عمومی، می‌توانیم به یک سازمان مرکزی برای توزیع کلیدها بیندیشیم که به صورت شبانه‌روزی فعال است و کلیدهای عمومی افراد را در اختیار قرار می‌دهد. یکی از مشکلات بی‌شمار این روش آنست که چندان قابل توسعه نیست و مرکز توزیع کلید سریعاً به یک گلوگاه در شبکه تبدیل می‌شود. [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 امضاکنندهٔ پیام، از رمز خارج می‌شود. ـم.

خواهد شد.

به دلائل فوق، عموم افراد راهکار متفاوتی اتخاذ کرده‌اند که در آن نیازی به وجود یک مرکز فعال و تمام وقت نیست. در حقیقت، اجباری برای روی‌خط (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 مبادلهٔ داده داشته باشند. او می‌تواند با در دست داشتن پاسپورت یا گواهینامهٔ 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 «رشتهٔ SHA-1 Hash»[1] این گواهینامه را استخراج و آنرا با کلید خصوصی خودش امضاء (رمز) می‌کند. سپس باب هزینهٔ مورد نظر CA را پرداخته و یک دیسک نرم (فلاپی) حاوی گواهینامه و رشتهٔ 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 رشتهٔ Hash امضاء شدهٔ 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 گواهینامهٔ جدید خودش را در وب‌سایت شخصی خود قرار داده و در صفحهٔ اصلی یک لینک با این مضمون تعبیه کند: «برای دریافت گواهی دیجیتالی من اینجا کلیک کنید!» با کلیک کردن، گواهینامهٔ دیجیتالی و «**بلوک امضای CA**» (یعنی رشتهٔ 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 صفحهٔ وب‌سایت باب را استراق سمع می‌کند، چه کاری می‌تواند انجام بدهد؟ ترودی می‌تواند بر روی صفحهٔ وب جعلی (که بدروغ برای آلیس می‌فرستد) گواهینامهٔ 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 گواهینامهٔ ارسالی مشاهده نمی‌کند. همچنین ترودی می‌تواند صفحهٔ 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، رشتهٔ Hash بدست آمده با رشتهٔ Hash که از رمزگشایی امضاء (با کلید عمومی و شناخته شدهٔ CA) حاصل می‌شود مطابقت ندارد. از آنجایی که ترودی کلید خصوصی CA را در اختیار ندارد لذا به هیچوجه قادر نخواهد

---

۱. یعنی خلاصهٔ پیام استخراج شده بروش SHA-1

بود که بلوک امضای دستکاری شده در صفحهٔ وب را رمز کند. بدین ترتیب آلیس می‌تواند مطمئن باشد که کلید عمومی باب را در اختیار دارد نه کلید ترودی 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 گواهینامهٔ دیجیتالی 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 گواهینامهٔ دیجیتالی 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 سایتها، 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 محاورهٔ مطمئن (رمزنگاری‌شده) بهره گرفت.

یک مثال دیگر که در آن به «گواهینامهٔ حاوی ویژگی» (attribute) نیاز خواهد بود سیستمهای شیء‌گرای توزیع‌شده هستند. هر «شیء» عموماً دارای چندین «متود» است. مالک شیء می‌تواند برای هر مشتری یک گواهینامهٔ دیجیتالی فراهم کند که در آن یک رشته بیتی درج شده و مشخص می‌کند آن مشتری، مجاز به فراخوانی چه متودهایی از آن «شیء» است و سپس این رشتهٔ 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 رشتهٔ 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، استانداردی برای صدور گواهینامه‌های دیجیتالی ابداع و توسط ITU تأئید شده است. این استاندارد X.509 نامیده می‌شود و امروزه در اینترنت کاربرد گسترده‌ای دارد. پس از استانداردسازی اولیه در ۱۹۸۸، تاکنون سه نسخهٔ 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 کُد کردن) را به عاریه گرفته است!! خوشبختانه، IETF با X.509 همراهی کرد، کما اینکه تقریباً در تمام زمینه‌ها از آدرس‌دهی ماشینها و پروتکلهای لایهٔ انتقال گرفته تا قالب نامه‌های الکترونیکی، دخالت و همراهی کرده و عموماً در اغلب آنها از OSI[1] 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 دیجیتالی است. فیلدهای اصلی یک گواهینامهٔ

---

۱. استاندارد جهانی ISO برای شبکه‌های باز

X.509 در شکل ۸-۲۵ فهرست شده‌اند. توضیحاتی که در این جدول وجود دارد می‌تواند کاربرد کلّی هر فیلد را مشخص کند. برای اطلاعات بیشتر لطفاً از RFC 2459 راهنمایی بگیرید.

به عنوان مثال اگر باب در قسمت «وام» از «بانکِ پول» (Money Bank) مشغول به کار باشد، ممکن است آدرس X.500[1] او برای درج در گواهینامه به صورت زیر باشد:

/C=US/O=MoneyBank/OU=Loan/CN=Bob/

که در آن C مشخصهٔ کشور، O مشخصهٔ سازمان (Organization)، OU مشخصهٔ 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.500 آن است که اگر آلیس تلاش کند تا با کسی به آدرس bob@moneybank.com و دارای گواهینامهٔ X.500 ، ارتباط برقرار کند، در گواهینامه‌ای که مشاهده می‌کند نام باب به وضوح دیده نمی‌شود. خوشبختانه در نسخهٔ سوم از استاندارد X.509، اجازه داده شده که نامهای DNS (اسامی و آدرسهای نمادین در اینترنت) به جای اسامی X.500 در گواهینامه‌ها درج شود که بدین نحو مشکل فوق حل شده است.

گواهینامه‌های دیجیتالی پس از تدوین، به روش OSI ASN.1[2] 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.

نام فیلد	کاربرد
Version	نسخه استاندارد X.509 را مشخص می‌کند.
Serial number	این شماره بهمراه نام CA هویت (و اصالت) این گواهینامه را بصورت یکتا مشخص می‌کند.
Signature algorithm	الگوریتم بکاررفته برای امضای گواهینامه را مشخص می‌کند.
Issuer	نام X.509 برای CA (نام مرکز گواهی امضا)
Validity period	زمان شروع و ختم اعتبار گواهینامه
Subject name	موجودیتی که کلید عمومی او در این گواهینامه تایید می‌شود. (مثل شخص یا موسسه)
Public key	کلید عمومی متعلق به صاحب گواهینامه و مشخصه الگوریتم مورد استفاده او
Issuer ID	یک شماره‌شناسایی منحصر بفرد و یکتا که هویت صادرکننده گواهینامه را تعیین می‌کند. (اختیاری)
Subject ID	یک شماره‌شناسایی منحصر بفرد و یکتا که هویت صاحب گواهینامه را تعیین می‌کند. (اختیاری)
Extensions	بکمک این فیلد می‌تواند بهر تعداد مشخصات اضافی تعریف کرد.
Signature	امضای کل گواهینامه (امضا شده با کلید خصوصی CA)

شکل ۸-۲۵. فیلدهای اصلی در گواهینامهٔ X.509.

۸-۵-۳ 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 کارآیی ساقط خواهد شد؛ زیرا در زیر بار زیاد فرو می‌پاشد و در نوع خود یک **«نقطهٔ 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

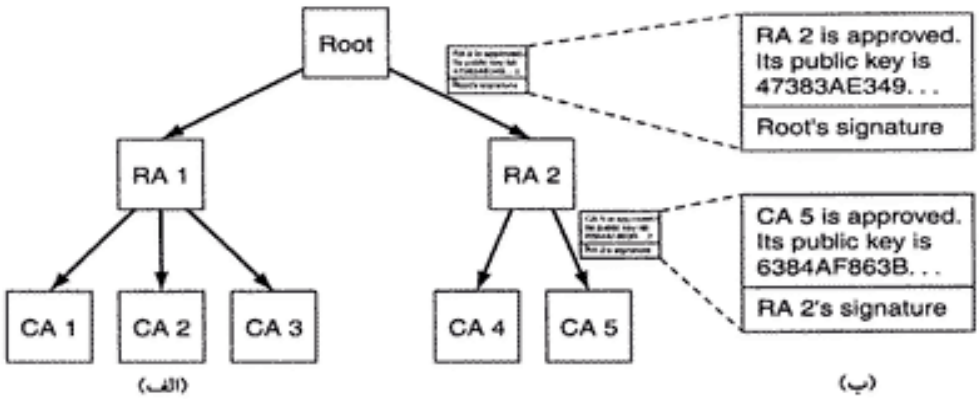
۱. X.500 استاندارد نامگذاری و X.509 استاندارد گواهینامه‌های دیجیتالی است.

۲. Abstract Syntax Notation 1

کنند. این راهکار اگرچه مسئله بار و مشکل خرابی را رفع می‌کند ولیکن مشکل جدیدی را بوجود خواهد آورد: **«مشکل نشت کردن و فاش شدن کلید»**. اگر دهها سرویس‌دهندهٔ 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 می‌یابد. از یک طرف توافق بر سر کلید خصوصی مشابه، زیربنای امنیت دنیای الکترونیک را متزلزل می‌کند و از طرف دیگر داشتن یک سازمان مرکزی صدور گواهینامه (CA) بسیار پرمخاطره است. به علاوه، چه سازمان یا نهادی باید متولی راه‌اندازی 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 دلائل روشهای متفاوتی برای گواهی کلیدهای عمومیِ 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 می‌کنیم، هر چند در این خصوص پیشنهادات بسیار زیاد است و جزئیات آن هنوز در حال تکوین و تدوین می‌باشد.

PKI از چندین مولفه شامل **«کاربران»**، **«CA[1]ها یا مراکز صدور گواهی»**، **«گواهینامه‌ها»** و **«دایرکتوریها»** تشکیل شده است. آنچه که PKI باید انجام دهد آن است که تمام این مولفه‌ها را تحت لوای یک ساختار واحد جمع کند و برای مستندات و پروتکلهای 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 داده‌ایم که در عمل می‌تواند بیشتر یا کمتر باشد. بالاترین سطح CA یعنی **«ریشه»** (Root)، فقط مراکز صدور گواهینامهٔ سطح ۲ را تائید و گواهی می‌کند؛ مراکز سطح ۲ اصطلاحاً RA[2] یا «مراکز مجاز منطقه‌ای» نامیده می‌شوند زیرا یک ناحیهٔ جغرافیایی مشخص مثل کشور یا قاره را پوشش می‌دهند. البته این تعاریف، استاندارد و دقیق نیستند و هیچ تعریف دقیقی برای مالکیت هر سطح از این ساختار درختی وجود ندارد. «مراکز مجاز منطقه‌ای» (RA) هویت مراکز صدور گواهینامهٔ دیجیتالی (CAها) را که به منظور صدور گواهینامه‌های X.509 برای سازمانها و اشخاص حقیقی راه‌اندازی شده، گواهی و تائید می‌کند. وقتی **«ریشه»**، یک RA جدید را تائید و گواهی نمود برای آن مرکز، یک گواهینامهٔ X.509 صادر و در اختیار آن RA قرار می‌دهد 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 می‌کند، این مرکز قادر است برای عموم افراد و سازمانها، گواهینامه‌های دیجیتالی شامل کلید عمومی آنها صادر نماید.



شکل ۸-۲۶. (الف) ساختار سلسله‌مراتبی PKI . (ب) یک زنجیره از گواهینامه‌ها.

۱. Certificate Authority ۲. Regional Authority

PKI شبیه به این سناریو عمل می‌کند: فرض کنید آلیس برای مبادلهٔ اطلاعات با باب به کلید عمومی او نیاز دارد، لذاگواهینامهٔ 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 راثابت کند. در پاسخ، CA5گواهینامهٔ خودش راکه توسط RA2 صادر و کلید عمومی CA5 در آن درج شده است، برای آلیس می‌فرستد. حال باکلید عمومی CA5، او می‌تواند گواهینامهٔ باب را [که توسط CA5 صادر شده] بررسی کند و طبعاً تائید می‌شود (در حالی که هنوز هویت RA2 مورد تردید است.). در مرحلهٔ بعدی او به سراغ RA2 رفته و از او می‌خواهد که هویت قانونی خود را اثبات نماید. در پاسخ به این درخواست، یک‌گواهینامهٔ دیجیتالی دیگر برمی گردد که توسط «ریشه» (عالیترین مرکز تائیدگواهینامه‌ها)، 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 صمیمی است! و نمی‌خواهد که آلیس زحمت زیادی بکشد. او می‌داند که آلیس مجبور است گواهینامهٔ CA5 و RA2 را بررسی کند، لذا برای کم کردن زحمت او، این دو گواهینامهٔ مورد نیاز آلیس را بدست آورده و آنها را نیز به همراه گواهینامهٔ خود برای او ارسال می‌نماید. حال آلیس با دانستن کلید «**ریشه**» شروع به بررسی گواهینامهٔ سطح بالای RA2 می‌کند و در صورت صحّت ، با استفاده از کلید عمومی موجود در آن، گواهینامهٔ بعدی یعنی CA5 را بررسی می‌نماید. بدین ترتیب آلیس احتیاجی به برقراری ارتباط با هیچ مرکزی برای بررسی صحّت گواهینامه‌ها ندارد. به دلیل اینکه تمام این گواهینامه‌ها امضاء شده هستند براحتی می‌تواند هر گونه تقلب و دستکاری در گواهینامه‌ها را کشف کند. زنجیرهٔ گواهینامه‌های دیجیتالی که نهایتاً به «ریشه» ختم می‌شود گاهاً «**زنجیرهٔ 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 «ریشهٔ واحد» نداشته باشیم بلکه تعداد بسیار زیادی «ریشه» (Root) وجود داشته باشد و هر یک برای خود تعدادی RA (مراکز منطقه‌ای) وتعدادی CA (مراکز صدور گواهینامه به افراد) داشته باشند. در حقیقت در مرورگرهای جدید، کلید عمومی بیش از صد «ریشه» گنجانیده شده و به صورت پیش‌فرض 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 عرضه‌کنندگان مرورگر (شرکتهایی که مرورگرها را طراحی می‌کنند) تصمیم بگیرند که‌کدامیک از مراکز عالی صدورگواهینامهٔ 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 تاکلیدهای 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 گواهینامهٔ خودش را

نگهداری کند. این راهکار مطمئن است (زیرا هیچ راهی برای دیگر کاربران وجود ندارد تا بتوانند گواهینامهٔ امضاء شدهٔ او را بدون آن که کشف شود دستکاری کنند) ولیکن این روش چندان راحت نیست. راهکار دیگری که پیشنهاد شده آن است که از سرویس‌دهندهٔ 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 زنجیرهٔ 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 دهندهٔ 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)

دنیای واقعی پر از گواهینامه‌های مختلف مثل گواهینامهٔ رانندگی و پاسپورت است. گاهی اوقات این گواهینامه‌ها می‌توانند باطل شود؛ مثلاً رانندگی در حالت مستی یا برخی دیگر از تخلفات منجر به لغو گواهینامهٔ رانندگی می‌گردد. همین مسائل در دنیای دیجیتال نیز اتفاق می‌افتد: اعطاکنندهٔ 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[1] (فهرست گواهینامه‌های باطل شده) صادر و در آن شمارهٔ سریال گواهینامه‌های باطل شده را مشخص نماید. از آنجایی که هر گواهینامه دارای «زمان اعتبار» مشخصی است لذا در CRL شمارهٔ 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 گواهینامهٔ 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. دفعهٔ 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 گواهینامهٔ باطل شده را تجدید اعتبار کرد. به عنوان مثال گواهینامه‌ای که به دلیل عدم پرداخت هزینهٔ 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 مراجعهٔ مکرّر به CA بوده است.

فهرست CRL کجا باید ذخیره شود؟ بهترین مکان ذخیرهٔ CRL، همان جایی است که اصل گواهینامه‌ها ذخیره

---

۱. Certificate Revocation List

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 نقطهٔ 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 ذخیرهٔ 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.

۸-۶ امنیت ارتباطات

تا اینجا مطالعهٔ 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 سرّی بیتها از مبداء 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 نقطهٔ شروع خوبی خواهند بود.

IPsec ۱-۶-۸

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 «**انتهابه‌انتها**» (End to End) انجام شود (به عبارت دیگر در سطح لایهٔ کاربرد)؛ بدین سیاق، پروسهٔ مبداء اقدام به رمزنگاری داده‌ها و تمهیدات حفاظتی کرده و سپس آنها را برای پروسهٔ مقصد ارسال می‌کند؛ این پروسه نیز داده‌ها را رمزگشایی کرده و آنها را از لحاظ صحّت 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 وظیفهٔ رمزنگاری داده‌ها به لایهٔ انتقال محول شود یا آن که لایهٔ جدیدی بین لایهٔ انتقال و لایهٔ کاربرد این وظیفه را بر عهده بگیرد؛ در این صورت باز هم امنیت، «انتهابه‌انتها» 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 استفادهٔ صحیح از آن نیستند و در ضمن هیچکس نمی‌خواهد برنامه‌های موجود خود را تحت هیچ شرایطی تغییر بدهد، لذا این لایهٔ 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، نظریهٔ پشتیبانی از امنیت در سطح لایهٔ شبکه به یک پیروزی نسبی دست یافت و استانداردهای امنیت در لایه شبکه شکل گرفت. چکیدهٔ استدلال آن بود که رمزنگاری در سطح لایهٔ 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 می‌کند.[۱]

نتیجهٔ این منازعه طرحی بود که IPsec نامیده شد و در مستندات RFC 2401، 2402 ، 2406 تشریح گشت. از آنجا که تمام کاربران نمی‌خواهند که از رمزنگاری [بسته‌ها] استفاده کنند (زیرا از لحاظ زمان پردازش هزینهٔ 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 برآن شد که در تمام حالات رمزنگاری انجام شود ولیکن در عوض، یک الگوریتم «**پوچ**» (Null Algorithm) (برای آنهایی که نمی‌خواهند بسته‌ها رمزنگاری شوند) تعریف گردید. این «**الگوریتم پوچ**» به دلیل سادگی، راحتی در پیاده‌سازی 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 بپردازند فلذا این خدمات به صورت انتخابی در اختیار کاربران هستند. خدمات ویژه عبارتند از «ارسال محرمانه بسته‌ها» (Secracy)، «تضمین صحّت» (Intgrity) و حفاظت در مقابل حملاتی که‌براساس آنها یک بخش از داده‌ها به صورت تکراری ارسال می‌شوند[۲] (که در آن اخلالگر سعی می‌کند یک سری از بسته‌های مجاز را بصورت تکراری ارسال نماید بدون آن که از محتوای آنها مطلع باشد). تمام خدمات فوق براساس رمزنگاری با کلید متقارن انجام می‌شود زیرا در سطح لایهٔ شبکه کارآیی و سرعت بسیار بالا، کاملاً حیاتی است. دلیل استفاده از چندین الگوریتم رمزنگاری آن بوده که شاید الگوریتمی که امروزه امن به حساب می‌آید 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 می‌دهد. [یعنی اعتبار IPsec به اعتبار یک الگوریتم رمزنگاری خاص گره نخورده است.]

دلیل مولفه‌های چندگانه‌ای که این پروتکل دارد آنست که بتوان فقط بر روی یک اتصال TCP متمرکز شد و از داده‌هایی که بین دو ماشین خاص در شبکه مبادله می‌شود مراقبت کرد یا آنکه از بین کل مسیریابها صرفاً ترافیک بین دو مسیریاب امن ، رمزنگاری شود.

یکی از جنبه‌های نسبتاً عجیب IPsec آن است که اگرچه این پروتکل در لایهٔ IP (لایهٔ شبکه) قرار می‌گیرد ولیکن برخلاف IP، اتصال‌گرا (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 می‌شود. (زیرا به هماهنگیهای قبلی بین ماشینها نیاز است.)

البته هزینهٔ برقراری چنین اتصالی بر روی حجم زیادی از بسته‌ها سرشکن می‌شود. یک «**اتصال**» در عُرف IPsec اصطلاحاً **SA** (Security Association) نامیده می‌شود. یک SA، اتصالی «**یکطرفه**» بین دو نقطهٔ پایانی در شبکه است که به آن یک «شناسهٔ امنیت» (Security Identifier) منتسب شده است. اگر نیاز باشد که ترافیک در دو جهت به صورت امن مبادله شود، به دو SA احتیاج است. «شناسه‌های امنیت» درون بسته‌هایی که در شبکه و مبتنی بر یک «اتصال» سیر می‌کنند جاسازی شده و از آن برای جستجوی کلید متناظر و همچنین بدست آوردن اطلاعات مرتبط با بسته‌های امن ورودی استفاده می‌شود.

---

۱. به عبارت بهتر کاربرانی که این سطح از امنیت آنها را راضی نمی‌کند هیچ مانعی برای پیاده کردن استراتژی‌های خود در لایه‌های بالاتر نخواهند داشت و می‌توانند این سطح از امنیت را نادیده بگیرند؛ در حالی که برای کاربران معمولی بسیار مفید است. ـم.

۲. Replay Attack

با دیدگاه فنّی، IPsec از دو بخش اصلی شکل گرفته است: در بخش اول دو سرآیند (Header) جدید تعریف شده که می‌تواند برای حمل «شناسهٔ امنیت» ، داده‌های لازم برای کنترل صحّت اطلاعات و داده‌های مرتبط با امنیت استفاده شود. بخش دیگر یعنی ISAKMP[1] با ایجاد و توزیع کلید رمز سر و کار دارد. ما به دو دلیل ISAKMP را بررسی نخواهیم کرد، زیرا: (۱) بشدت پیچیده است. (۲) پروتکل اصلی آن IKE[2] 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 بستهٔ IP قرار می‌گیرد و فیلد پروتکل در بستهٔ IP به گونه‌ای مقداردهی می‌شود که مشخص کند پس از سرآیند بستهٔ IP سرآیند IPsec شروع می‌شود.[3] سرآیند IPsec ، شامل اطلاعات امنیتی نظیر شناسهٔ SA ، یک شماره ترتیب جدید و احتمالاً تنظیمات لازم برای بررسی صحت محتوای بسته (Payload) است.

در «**حالت تونل**» (Tunnel Mode)، کل بستهٔ IP شامل سرآیند و محتوای آن در درون یک بستهٔ جدید با سرآیند متفاوت جاسازی می‌شود. «حالت تونل» زمانی مفید است که انتهای تونل به جای ماشین مقصد به یک نقطهٔ خاص [مثل یک مسیریاب] ختم شود. در برخی از حالات، انتهای تونل یک ماشین است که نقش «دروازهٔ 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 شبکهٔ محلی آن شرکت واقعند نیازی به IPsec نخواهند داشت. در این حالت فقط دیوار آتش باIPsec سر و کار دارد.[4] 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 اخلالگر 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 اخلالگر می‌تواند از این داده‌ها، اطلاعات با ارزشی بدست بیاورد. مطالعهٔ الگوی جریان بسته‌ها، حتی وقتی رمزنگاری شده هستند، اصطلاحاً «**تحلیل ترافیک**» (Traffic Analysis) نامیده می‌شود. «حالت تونل» می‌تواند تا حدودی برای خنثی کردن این مشکل کارساز باشد. اشکال «حالت تونل» آن است که باید یک سرآیند اضافی به هر بستهٔ IP افزوده شود و بدین ترتیب اندازهٔ بسته‌ها افزایش خواهد یافت. برعکس، در «حالت انتقال» اندازهٔ بسته چندان تغییر نخواهد کرد.

اولین سرآیند جدید، AH[5] است. این سرآیند، بررسی صحت داده و غیرتکراری 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 بستهٔ IP و سرآیند بستهٔ 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 دادهٔ زائد (Pad) اضافه شود.

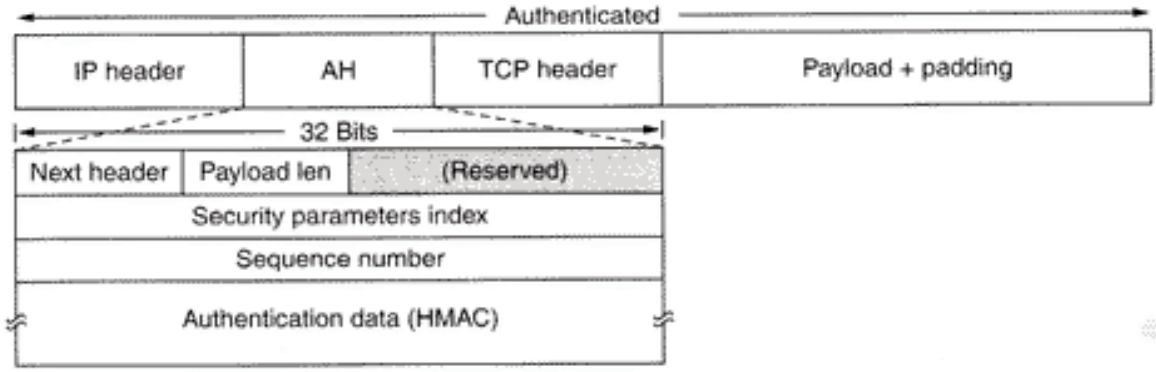
---

۱. Internet Security Association and Key Management Protocol ۲. Internet Key Exchange

۳. در فیلد پروتکل از بستهٔ IP شمارهٔ پروتکل لایهٔ بالاتر (پردازش‌کنندهٔ بسته) درج می‌شود. درج شمارهٔ IPsec در این فیلد نشان می‌دهد که بستهٔ IP محتوی یک بستهٔ IPsec است، نه بستهٔ TCP ، UDP یا هر پروتکل دیگر. رجوع کنید به فصل ۵. -م.

۴. زیرا دیوار آتش به نیابت از همهٔ آنها بسته‌ها را جاسازی و رمزنگاری می‌کند و سمت مقابل، آنها را بازگشایی می‌نماید. بنابراین ماشینها ، درگیر باIPsec نخواهند بود.

۵. Authentication Header



شکل ۸-۲۷. «سرآیند احراز هویت» (Authentication Header) که در «حالت انتقال» و برای IPv4 بکار می‌آید.

حال بیایید سرآیند AH را بررسی کنیم: فیلد Next Header بدان منظور به کار می‌رود که مقدار قبلی فیلد Protocol در بستهٔ IP را (قبل از تغییر به مقدار ۵۱[۱])، حفظ نماید. در اغلب موارد در این فیلد، عدد ۶ قرار می‌گیرد (به معنای وجود بستهٔ TCP در درون بستهٔ IPsec). فیلد Payload Length، طول سرآیند AH را ۲ واحد کمتر، در مبنای کلمات ۳۲ بیتی نشان می‌دهد.[۲]

فیلد Security Parameter Index، «شناسهٔ اتصال» است. این فیلد توسط فرستنده تنظیم می‌شود 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 می‌شد!

فیلد Sequence Number برای شماره‌گذاری تمام بسته‌هایی است که بر روی یک SA[۳] ارسال می‌شوند. تمام بسته‌ها، حتی آنهایی که به هر دلیل از نو ارسال شده‌اند یک شمارهٔ مستقل و یکتا می‌گیرند. به عبارت دیگر، ارسال مجدد بسته‌ای که قبلاً نیز فرستاده شده با شمارهٔ جدید انجام می‌شود (حتی اگر شمارهٔ ترتیب آن در بستهٔ TCP یکسان باشد.) هدف از این فیلد آن است که **«حملهٔ ارسال تکراری»** (Replay Attack) کشف شود. این شمارهٔ ترتیب هیچگاه از نو به صفر بر نخواهد گشت [اصطلاحاً Wrap around نیست] و هر گاه تمام 2^{32} حالت آن استفاده شد برای ادامهٔ مبادلهٔ اطلاعات باید یک SA جدید به وجود بیاید.

نهایتاً به فیلد Authentication Data می‌رسیم که فیلدی با طول متغیر است و امضای دیجیتالی داده‌های درون بسته در آن قرار می‌گیرد. وقتی یک SA بوجود آمد، ابتدا طرفین در خصوص الگوریتم امضای دیجیتالی که از آن استفاده خواهند کرد مذاکره و توافق می‌کنند. در اینجا عموماً از روشهای مبتنی بر کلید عمومی (Public Key) استفاده نمی‌شود چرا که بسته‌ها باید بی‌نهایت سریع پردازش شوند در حالی که روشهای کلید عمومی بسیار کُند 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 می‌کنند لذا از همین کلید برای محاسبهٔ امضای دیجیتالی استفاده می‌شود. ساده‌ترین راه برای احراز هویت داده‌ها آنست که خلاصهٔ درهم‌شدهٔ (Hash) کلّ بسته به انضمام کلید مشترک استخراج شود و در این فیلد قرار بگیرد. البته کلید مشترک هرگز بر روی خط ارسال نمی‌شود بلکه فقط برای محاسبهٔ این فیلد به بسته اضافه می‌شود. به ساختاری شبیه به این روش، اصطلاحاً HMAC[۴] گفته می‌شود. این

---

۱. مقدار ۵۱ نشانگر وجود بستهٔ IPsec در درون بستهٔ IP است. -م.

۲. طول سرآیند AH متغیر است. -م.

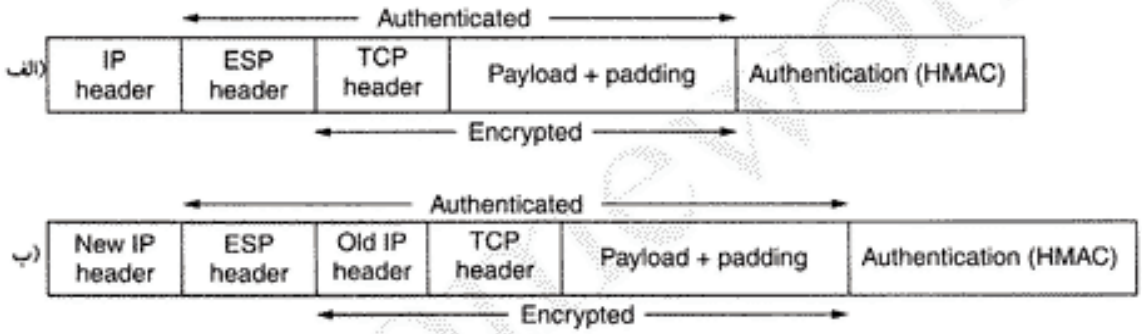
۳. SA را یک نشست یا اتصال یکطرفه فرض کنید. -م.

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 نتیجهٔ آن اعمال گردد.

سرآیند AH امکان رمزنگاری داده‌ها را فراهم نکرده است و بالطبع زمانی مفید است که فقط صحّت داده‌ها اهمیت داشته باشد و نیازی به ارسال محرمانهٔ 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 مسیریاب به مسیریاب دیگر تغییر نخواهند کرد، در برمی‌گیرد. به عنوان مثال فیلد TTL (Time To Live) در بستهٔ IP در هر گام قطعاً تغییر می‌کند فلذا نمی‌توان آن را در محاسبهٔ کُد تایید صحت دخالت داد در حالی که فیلد «آدرس مبداء» (Source Address) می‌تواند در محاسبهٔ کد بررسی صحّت دخالت داده شود تا دستکاری اخلالگران در آدرس مبدأ بسته، ناممکن باشد.

یکی دیگر از سرآیندهای بستهٔ 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 کلمهٔ ۳۲بیتی تشکیل شده است. این دو فیلد عبارتند از: Security Parameter Index و Sequence Number که تعریف آنها را در AH دیدیم. کلمهٔ 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، کُد 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 کارآیی بیشتر و سرعت بالاتری نیز دارد، پس یک سؤال بوجود می‌آید: چرا با داشتن AH خودمان را به زحمت انداخته‌ایم؟ پاسخ این سؤال، ریشهٔ تاریخی دارد؛ در ابتدا AH فقط صحت داده‌ها را (Integrity) و ESP فقط محرمانه ماندن داده‌ها (Secracy) 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 قانع‌کنندهٔ آنها این بود

که AH بخشی از سرآیند بستهٔ IP را در بررسی صحت (Intgrity) بسته دخالت می‌دهد در حالی که ESP این کار را نمی‌کند ولی این دلیل، پشتوانهٔ ضعیفی دارد. یک استدلال ضعیف دیگر آن است که محصولاتی که از AH حمایت می‌کنند ولی از ESP حمایت نمی‌کنند ممکن است مشکلات کمتری در اخذ مجوز صدور از دولت بگیرند زیرا هیچگونه رمزنگاری انجام نمی‌شود. [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 منزلهٔ یک سکهٔ دو رو است؛ برای اشخاصی که در منزل هستند گردش در اینترنت بسیار لذت‌بخش است در حالی که برای مدیران امنیت در شرکتها، یک کابوس وحشتناک به حساب می‌آید. اغلب شرکتها دارای حجم عظیمی از اطلاعات محرمانه و «روی خط» (Online) هستند، مثل اسرار بازرگانی، طرحهای توسعهٔ محصولات، استراتژیهای فروش، تحلیل‌های اقتصادی و نظایر آنها. دسترسی رقبا به این اطلاعات می‌تواند 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، می‌تواند امنیت را درهم بشکند، داده‌های ارزشمند را نابود کند و وقت بسیار زیادی از مسئول شبکه برای ساماندهی به آسیبهای بجا مانده، تلف نماید. این اطلاعات مخرب توسط کارمندان بی‌دقتی که مثلاً خواسته‌اند یک بازی کامپیوتری جدید و جذاب را اجرا کنند، به درون شبکه منتقل می‌شود.

در نتیجه به مکانیزمهایی نیاز است که بتوان بیتهای «خوب» را از بیتهای «بد» تفکیک کرد. یک روش آن است که از IPsec استفاده شود. IPsec از داده‌هایی که بین سایتها در حال تردد هستند بخوبی مراقبت می‌کند ولیکن هیچ کاری برای پیشگیری از ورود آفتهای دیجیتالی (مثل ویروسها) و اخلالگران به درون شبکهٔ محلی شرکت انجام نمی‌دهد.

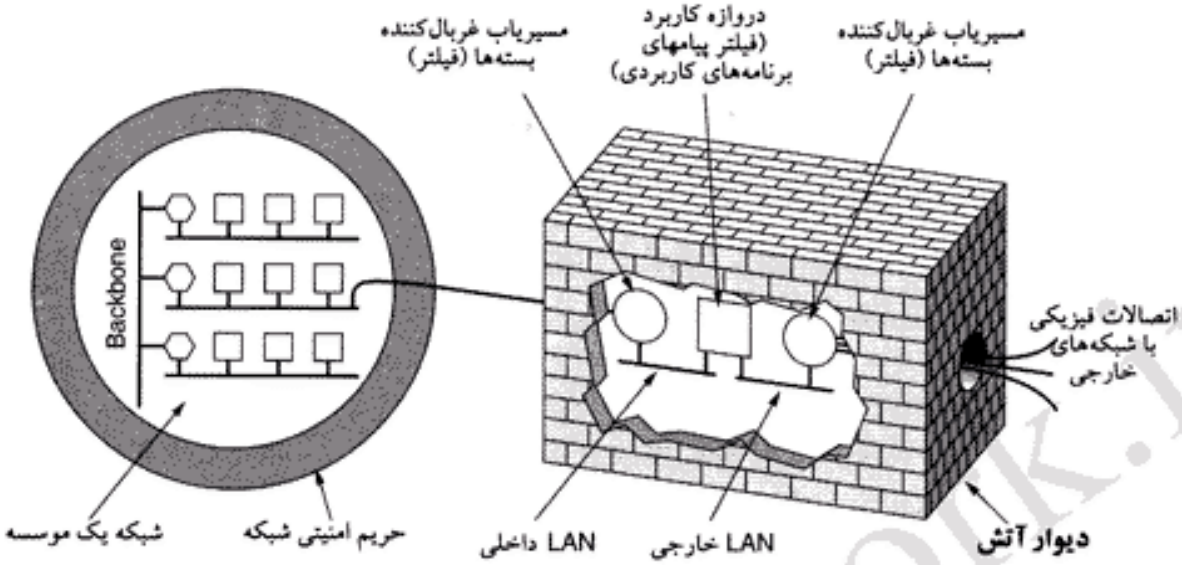
«دیوار آتش» (Firewall) پیاده‌سازی مدرنی از روش قدیمی و قرون وسطایی حصارهای امنیتی است: خندقی عمیق دور تا دور قلعهٔ خود حفر کنید! این الگو همه را مجبور می‌کند تا برای ورود یا خروج از قلعه، از یک پل متحرک و واحد بگذرند و بتوان همه را توسط پلیس حراست بازرسی کرد. در دنیای شبکه‌های کامپیوتری، همین راهکار ممکن خواهد بود: یک شرکت می‌تواند هر تعداد شبکهٔ محلی داشته باشد که به صورت دلخواه به هم متصل شده‌اند، 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 مسیریاب که عمل غربال‌سازی بسته‌ها را انجام می‌دهند. (Packet Filtering) (۲) دروازهٔ برنامه‌های کاربردی (Application Gateway). ساختار ساده‌تری نیز وجود دارد ولیکن حُسن بزرگ این طرح آنست که هر بسته باید از دو مرحلهٔ غربال‌سازی (فیلترینگ) و یک مرحله بازرسی محتوایی توسط دروازه، بگذرد. هیچ مسیر دیگری نیز وجود ندارد. خوانندگانی که فکر می‌کنند فقط یک مرحله بازرسی امنیتی کافی است، به احتمال زیاد اخیراً یک پرواز بین‌المللی با خطوط هوایی نداشته‌اند!

هر غربال کنندهٔ بسته، یک مسیریاب استاندارد با برخی از ویژگیهای بیشتر (درخصوص غربال‌سازی)[1] است. این قابلیت اجازه می‌دهد که تمام بسته‌های ورودی و خروجی بازرسی شوند. بسته‌هایی که بتوانند برخی از معیارها

---

۱. تمام مسیریابهای امروزی قابلیت فیلترینگ را دارند. -م.



شکل ۸-۲۹. یک دیوار آتش شامل دو «فیلترکنندهٔ بسته» و یک «دروازهٔ کاربردی».

و شرایط را احراز کنند بطور طبیعی هدایت می‌شوند و آنهایی که در این بازرسی مردود شوند حذف می‌گردند.

در شکل ۸-۲۹، غربال‌کنندهٔ داخلی (متصل به LAN) بسته‌های خروجی از شبکه را و غربال کنندهٔ بیرونی [متصل به خط ارتباط بیرونی] بسته‌های ورودی به شبکه را بازرسی می‌کنند. بسته‌هایی که بتوانند از اولین مانع عبور کنند، برای بازرسی بیشتر واردِ **«دروازهٔ برنامه‌های کاربردی»** می‌شوند. قرار دادن دو غربال‌کننده این تضمین را می‌دهد که هیچ بسته‌ای نتواند قبل از بررسی‌های ابتدایی و سپس گذر از دروازه، از شبکه خارج یا به آن وارد شود.

غربال کنندهٔ بسته عموماً بنابر جدولی که توسط مسئول شبکه تنظیم می‌شود، در خصوص بسته‌ها تصمیم‌گیری می‌کند. در این جدول آدرس مبداء یا آدرس مقصد ماشینهای مجاز و ماشینهای غیرمجاز و همچنین قواعدی در خصوص شرایط تردّد بسته‌ها درج می‌شود.

در شبکه‌هایی که عموماً با TCP/IP پیکربندی شده‌اند مبداء و مقصد با آدرس IP و شماره پورت مشخص می‌شود. شمارهٔ پورت مشخص می‌کند که چه سرویسی مورد نظر است. به عنوان مثال پورت شمارهٔ 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 می‌باشد. یک شرکت می‌تواند تمام بسته‌ها 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 بسته‌های یوزنت، مانع از آن شود که کارمندانش روز خود را با خواندن اخبار بگذرانند.

مسدود و حذف کردن بسته‌های خروجی از شبکه نیاز به زیرکی بیشتری دارد زیرا اگرچه اکثر سایتها خودشان را مقیّد به شماره‌گذاری استاندارد پورتها کرده‌اند ولی اجباری به انجام این کار نیست.[1] گذشته از آن، در سرویسهای بسیار مهمی نظیر FTP (پروتکل انتقال فایل) شماره پورت به صورت پویا تعیین می‌شود. اگرچه

---

۱. مثلاً اغلب سایتها از پورت ۸۰ صرفاً برای سرویس‌دهندهٔ وب استفاده می‌نمایند ولی برخی از افراد برای ردّ گم کردن شمارهٔ پورت ۸۰ را برای سرویس‌دهنده‌های دیگر (مثل Telnet یا NetCat) انتخاب کرده‌اند. -م.

مسدود ساختن اتصالات TCP (از طریق حذف بسته‌ها) کاری مشکل است، حذف بسته‌های 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 کننده‌های بسته به گونه‌ای پیکربندی شده‌اند که به سادگی ورود و خروج بسته‌های UDP را در دو جهت مسدود و قدغن کنند.

نیمهٔ دوم یک دیوار آتش «**دروازهٔ برنامه‌های کاربردی یا** Application Gateway» است. این قسمت به جای آن که بررسی خود را بر روی مشخصات بسته‌های خام متمرکز کند در سطح لایهٔ کاربرد عمل می‌کند. مثلاً دروازهٔ پست الکترونیکی (Mail Gateway)، دروازه‌ای است که براساس مشخصات هر پیام تصمیم می‌گیرد 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 می‌گیرد. (به عنوان مثال در یک مقر نظامی، وجود کلماتی نظیر «بمب»، «اتمی» در پیام ممکن است به انجام عملیات ویژه بیانجامد.) مؤسسات آزادند که به هر تعداد «دروازهٔ برنامهٔ کاربردی» برای سرویس دهنده‌های خود نصب نمایند ولی بطور کلی تردد نامه‌های الکترونیکی و استفاده از وب جهانی در اغلب سازمانها مجاز شمرده می‌شود. بقیهٔ سرویسها معمولاً مسدود هستند. ترکیب رمزنگاری و غربال‌سازی بسته‌ها می‌تواند 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 گونه‌ای پیکربندی شده باشد که فقط به بسته‌های یک شبکه خاص (مثل شبکهٔ متعلق به یکی از کارخانه‌های شرکت) اجازهٔ تردّد بدهد، یک اخلالگر قادر خواهد بود با قرار دادن آدرسهای غلط (آدرس مبداء جعلی) از حصار این بازرسی عبور نماید. اگر یکی از افراد در داخل شبکه بخواهد یک سند محرمانه را خارج نماید می‌تواند 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 کننده‌ای که حتی درون متن را جستجو می‌کند، بگذرد. هنوز این حقیقت را خاطرنشان نکرده‌ایم که ۷۰ درصد از کل حملات از درون شبکه و قبل از دیوار آتش صورت می‌گیرد که به عنوان مثال توسط کارمندان ناراضی هدایت می‌شود. (Schneier, 2000)

به علاوه، یک ردهٔ کامل از حملات وجود دارند که دیوار آتش هیچ کاری در مواجهه با آنها نمی‌تواند انجام بدهد. ایدهٔ اصلی دیوار آتش آن است که جلوی ورود اخلالگران 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 بسته‌های مجاز در تعداد بسیار زیاد به یک هدف در شبکه انجام می‌دهند تا هدف زیر بار بالا درهم بشکند. به عنوان مثال برای زمین‌گیر کردن یک سایت وب، اخلالگر می‌تواند حجم بسیار زیادی بسته‌های 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 بستهٔ SYN+ACK باز پس می‌فرستد. اگر اخلالگر هیچ واکنشی به این بسته‌های پاسخ نشان ندهد، جدول تخصیص داده شده برای چندین ثانیه و تا زمان انقضای مهلت، در حافظه باقی خواهد ماند. اگر اخلالگر بتواند هزاران بستهٔ تقاضای اتصال (TCP SYN) را ارسال کند، جدول مربوطه پر شده و از آن به بعد برقرای هیچ ارتباط مُجاز نیز ممکن نخواهد بود. حملاتی که در آنها مقصود اخلالگر به جای سرقت اطلاعات از کار انداختن یک هدف در شبکه است اصطلاحاً «**حملهٔ DoS**»[1] نامیده می‌شود. معمولاً بسته‌های تقاضا دارای آدرس مبداء غلط هستند و بدین ترتیب براحتی نمی‌توان اخلالگر را تعقیب کرد.

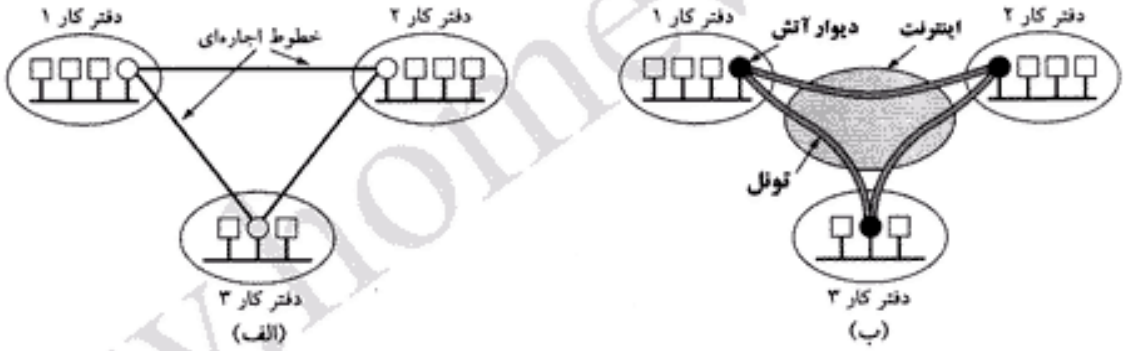
---

۱. حملهٔ اخلال در سرویس‌دهی یا Denial of Service

یک حالت بسیار خطرناکتر آن است که اخلالگر 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 حملهٔ اخلالگر را افزایش می‌دهد بلکه احتمال تعقیب و یافتن او نیز کاهش می‌یابد زیرا بسته‌هایی به هدف گسیل می‌شوند که متعلق به ماشین کاربران عادی و غیرمشکوک هستند. به چنین حمله‌ای اصطلاحاً «**حملهٔ DDoS**»[1] گفته می‌شود؛ مقابله با چنین حمله‌ای واقعاً مشکل است. حتی اگر ماشین تحت حمله بتواند سریعاً درخواستهای غیرمتعارف را از درخواستهای مجاز تشخیص بدهد زمانی طول می‌کشد تا پردازش و حذف شوند و اگر این درخواستها در ثانیه، از حدّی بیشتر شود کلّ زمان CPU صرف پردازش آنها خواهد شد.

۸-۶-۳ شبکه‌های خصوصی مجازی (VPN)

بسیاری از شرکتها داری دفاتر و کارخانه‌هایی هستند که در شهرها و گاهی در چندین کشور پراکنده‌اند. در ایام گذشته، قبل از ایجاد شبکه‌های عمومی داده، برای اتصال شبکه‌های پراکندهٔ متعلق به یک شرکت، رایج‌ترین کار استفاده از خطوط اجاره‌ای (Leased Line) متعلق به شرکتهای مخابرات بود. شبکه‌ای که از کامپیوترهای یک شرکت و خطوط اجاره‌ای تلفن تشکیل شده اصطلاحاً «**شبکهٔ خصوصی**» (Private Network) نامیده می‌شود. مثالی از یک شبکهٔ خصوصی که سه شبکه را به هم متصل ساخته در شکل ۸-۳۰-الف نشان داده شده است.



شکل ۸-۳۰. (الف) یک شبکهٔ خصوصی با خطوط اجاره‌ای. (ب) شبکهٔ خصوصی مجازی.

شبکه‌های خصوصی، بسیار خوب و مطمئن عمل می‌کنند. اگر خطوطِ در اختیار شرکت، تماماً اجاره‌ای باشند، هیچ ترافیکی نمی‌تواند به بیرون از شرکت نشت کند و اخلالگر مجبور است به صورت فیزیکی از خطوط انتقال انشعاب گرفته و بدانها متصل شود که انجام این کار نیز ساده نخواهد بود. مشکل بزرگ شبکه‌های 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 شبکه‌های عمومی داده[2] و بعد از آن اینترنت به صحنه آمد، بسیاری از شرکتها تصمیم گرفتند که انتقال داده‌های خود (و احتمالاً انتقال صوت) را از طریق شبکه‌های عمومی موجود انجام بدهند که هزینهٔ ناچیزی دارد ولیکن در عوض امنیت یک شبکهٔ خصوصی را ندارد.

احساس این نیاز به ابداع VPN (**شبکهٔ خصوصی مجازی**) انجامید که بر روی زیرساخت شبکهٔ عمومی بنا شده است ولی اکثرِ ویژگیهای یک شبکهٔ خصوصی را عرضه می‌کند. این گونه شبکه‌ها از آن جهت «**مجازی**» نامیده شده‌اند که صرفاً یک توّهم (یا بهتر بگوییم یک مدل انتزاعی) هستند؛ دقیقاً مثل «مدار مجازی» که در آن هیچ مدار واقعی در کار نیست یا «حافظهٔ مجازی» که در آن هیچ حافظهٔ واقعی از نوع RAM وجود ندارد.

---

۱. Distributed Denial of Service

۲. Public Data Network

اگرچه VPN را می‌توان بر روی ATM (یا شبکه Frame Relay) پیاده کرد ولیکن عمومیترین روش 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 گونه‌ای که در شکل ۸-۳۰-ب 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 می‌توان ترافیک جاری بین دو طرف ارتباط را از طریق یک SA[1] که احراز هویت و رمزنگاری شده است، ارسال کرد و بدین ترتیب صحت و امنیت داده‌ها تضمین می‌شود و همچنین ایمنی قابل توجهی در مقابل خطر «تحلیل ترافیک» (که در بخش IPsec بدان اشاره شد) بدست می‌آید.

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 مدلهای معمولی مسیریابها نیز همین کار را بخوبی انجام می‌دهند، ولیکن از آنجایی که دیوارهای آتش ذاتاً در محیطهای امن بکار گرفته می‌شوند بنابراین طبیعی است که تونلهایی داشته باشیم که از یک دیوار آتش شروع و به دیگری ختم می‌گردند تا تفکیک کاملی بین اینترنت و شبکهٔ یک شرکت ایجاد نماییم. بدین ترتیب دیوارهای آتش ، VPN و IPsec به همراه ESP (در حالت تونل) ترکیب طبیعی برای چنین محیطهایی است و در عمل به صورت گسترده‌ای از این ترکیب استفاده می‌شود.

پس از آن که SA ایجاد شد، ترافیک می‌تواند جریان پیدا کند. از دیدگاه مسیریابهایی که در درون ساختار شبکهٔ اینترنت واقع هستند بسته‌هایی که از تونل VPN منشاء گرفته‌اند هیچ تفاوتی با بسته‌های معمولی IP ندارند.[2] تنها چیزی که بسته‌های IP حاوی داده‌های معمولی را از بسته‌های IP حاوی بستهٔ IPsec جدا می‌کند سرآیند بستهٔ IPsec است که دقیقاً بعد از سرآیند بستهٔ IP معمولی قرار گرفته است ولی برای مسیریابها این سرآیند اضافی هیچ اهمیتی ندارد و در شرایط عادی تأثیری بر عمل هدایت و مسیریابی نمی‌گذارد چراکه مسیریابها عموماً به این سرآیند اضافی اعتنایی نمی‌کنند.

بزرگترین حسن ایجاد شبکهٔ VPN آن است که بطور کلی از تمام نرم‌افزارهای کاربر مستقل بوده و هیچ تغییری در آنها نیاز نیست و امکانات آن کاملاً نامرئی است: دیوارهای آتش به صورت مستقل SAهای لازم را ایجاد و مدیریت می‌کنند. تنها کسی که از تنظیمات آن اطلاع دارد مسئول شبکه است که طبعاً موظف به پیکربندی و مدیریت دیوارهای آتش می‌باشد. برای بقیهٔ افراد، این شبکه دقیقاً مشابه با شبکه‌های خصوصی با خطوط اجاره‌ای است. برای کسب اطلاعات بیشتر در خصوص VPN مرجع (Brown, 1999; Izzo, 2000) را نگاه کنید.

۸-۶-٤ امنیت شبکه‌های بی‌سیم

از لحاظ منطقی طراحی سیستمی که کاملاً امن باشد بکمک 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 می‌افتد که برخی از ماشینهای شبکه بی‌سیم بوده و از مخابرهٔ رادیویی استفاده کرده باشند که در این صورت در همهٔ جهات پیرامون دیوار آتش، داده‌های در حال تبادل قابل شنود هستند. محدودهٔ کاری شبکهٔ ۸۰۲.۱۱ حدود چند صد متر است، لذا هر کسی که بخواهد جاسوسی یک شرکت را بنماید می‌تواند براحتی خودروی خود را صبح زود در پارکینگ کارمندان پارک کرده و یک کامپیوتر کیفی مجهز به شبکهٔ ۸۰۲.۱۱ را در درون خودرو به گونه‌ای پیکربندی نماید که هر آنچه را می‌شنود 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 بسته‌های IPsec در درون یک بستهٔ IP معمولی جاسازی می‌شوند. -م.

دیدگاه تئوری فرض بر آن است که هیچ فردی به بانک دستبرد نمی‌زند!!

بسیاری از مشکلات امنیتی در شبکه‌های بی‌سیم به سازندگان ایستگاههای ثابت[۱] بر می‌گردد که سعی می‌کنند محصولاتشان هر چه بیشتر ساده و با محیطی دوستانه باشد. عموماً اگر یک کاربر دستگاه جانبی مورد نیاز برای اتصال به شبکهٔ بی‌سیم را خریداری و آن را به برق متصل کند، بلافاصله عملیاتی شده و شروع به کار می‌کند در حالی که هیچگونه امنیتی وجود ندارد و تمام اطلاعات محرمانه در محدودهٔ برد شبکهٔ بی‌سیم فاش خواهد شد. شبکه بی‌سیم رویای جاسوسان الکترونیکی را محقق کرده است یعنی: «دسترسی آزاد به داده‌ها بدون نیاز به انجام هیچ کاری!»

بنابراین باید اشاره کرد که امنیت اطلاعات در شبکه‌های بی‌سیم از اهمیت و حساسیت بیشتری نسبت به شبکه‌های مبتنی بر سیم برخوردار است. در این بخش به روشهایی که سعی در برقراری امنیت در شبکه‌های بی‌سیم دارند نگاهی خواهیم انداخت. اطلاعات بیشتر را می‌توانید در (Nichols and Lekkas, 2002) بیابید.

امنیت شبکه ۸۰۲.۱۱

استاندارد ۸۰۲.۱۱ پروتکلی برای ایجاد امنیت در سطح لایهٔ پیوند داده‌ها به نام WEP[۲] معرفی کرده که هدف از طراحی آن، برقراری امنیت در شبکه‌های بی‌سیم در سطحی معادل با شبکه‌های مبتنی بر سیم بوده است. از آنجایی که شبکه‌های محلی مبتنی بر سیم، به صورت پیش فرض هیچ امنیتی ندارند رسیدن به این هدف بسیار ساده است و قطعاً WEP (به گونه‌ای که خواهیم دید) به این هدف رسیده است!!!!

وقتی گزینهٔ امنیت (WEP) در شبکهٔ بی‌سیم 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 سازندهٔ سخت‌افزار بی‌سیم به صورت کاملاً تصادفی انتخاب و از قبل تنظیم شده باشد. بعداً می‌توان این کلیدها را تعویض کرد. نهایتاً چه ایستگاه ثابت و چه ماشین کاربران (ایستگاههای متحرک) می‌توانند از طریق این کلیدِ 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 می‌تواند برای ماهها یا حتی سالها ثابت باقی بماند.

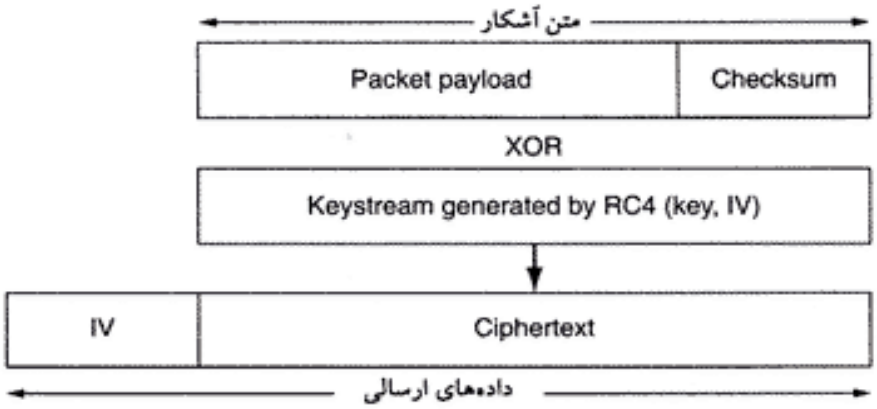
WEP برای رمزنگاری از روش Stream Cipher[۳] و الگوریتم RC4 استفاده می‌کند. RC4 توسط رونالد ری‌وست (از ابداع کنندگان RSA) طراحی شده و الگوریتم آن محرمانه نگاه داشته شده بود تا آن که در سال ۱۹۹۴ این الگوریتم کشف و در اینترنت اعلام شد! قبلاً هم اشاره کرده بودیم که محرمانه نگه داشتن الگوریتم تقریباً غیرممکن 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 داده‌های رمز نشده XOR می‌کند تا متن رمزشده بدست آید.

فیلد دادهٔ هر بستهٔ اطلاعاتی طبق روشی که در شکل ۸-۳۱ نشان داده شده رمز می‌شود. ابتدا از اصل پیام با استفاده از یک چند جمله‌ای CRC-32، یک کد چهار بایتی کشف خطا استخراج و به انتهای داده‌ها ضمیمه می‌شود تا مجموع این دو به الگوریتم رمزنگاری تحویل گردد. سپس این داده‌های رمز نشده با یک Keystream طولانی XOR می‌شود. حاصل این XOR، داده‌های رمز شده خواهد بود. IV[۴] مورد نیاز برای شروع در الگوریتم RC4 به همراه داده‌ها ارسال خواهد شد. وقتی گیرنده بسته‌ای را دریافت می‌کند داده‌های 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 می‌کند. سپس برای بررسی هر گونه دستکاری در داده‌ها، کد کشف خطای آن [یعنی CRC] را آزمایش و بررسی می‌نماید.

اگرچه در نگاه اول این روش بسیار خوب به نظر می‌رسد ولی راهی برای شکستن آن تدوین و پیشنهاد شده است. (Borisov et al., 2001) در ادامه خلاصه‌ای از روش درهم شکستن WEP را ارائه می‌کنیم. اول از همه آنکه در بسیاری از مؤسسات، کلیدی مشترک و یکسان برای همهٔ کاربران تعریف شده که در چنین حالتی یک کاربر می‌تواند ترافیک تمام کاربران دیگر را براحتی بدست آورده و بخواند. این مسئله دقیقاً مشابه با شبکهٔ اترنت است ولی به هیچوجه امن نیست.

حتی اگر هر کاربر کلیدی مجزا داشته باشد باز هم می‌توان به WEP حمله کرد. از آنجایی که کلیدها برای مدت زمانی بسیار طولانی تغییر نمی‌کنند، WEP توصیه کرده که لااقل IV برای هر بسته تغییر کند تا نتوان از طریق حملهٔ Keystream Reuse Attack که شرح آن در بخش ۸-۲-۳ آمد، اطلاعات را رمزگشایی کرد. (تغییر IV فقط توصیه شده ولی اجباری نیست.) متأسفانه در بسیاری از کارتهای واسط شبکهٔ ۸۰۲.۱۱ که برای کامپیوترهای کیفی ارائه شده، وقتی کارت در درون کامپیوتر قرار داده می‌شود، مقدار IV به صفر تنظیم و به ازای ارسال هر بسته یک واحد به IV اضافه می‌شود. از آنجایی که افراد، این کارت را باز و بسته می‌کنند، لذا مقدار IV عموماً کم است. اگر ترودی بتواند تعدادی بسته را که با IV مشابه توسط یک کاربر خاص ارسال شده، جمع‌آوری 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 می‌شود.)

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 جمعاً ۲۴ بیت است و پس از ارسال 2^{24} بسته، از شماره‌های تکراری استفاده خواهد شد. بدتر از آن، اگر IV به صورت تصادفی انتخاب شود طبق روشی که در بخش ۸-۴-۴ در مورد «**حملهٔ روز تولد**» اشاره شد، متوسط بسته‌هایی که باید ارسال شود تا به یک زوج IV مشابه برخورد کنیم حدود ۵۰۰۰ است. ($2^{24/2}=2^{12}=4096$) طبق این استدلال، اگر ترودی برای چند دقیقه به بسته‌های 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 می‌آید. این رشتهٔ بیت را می‌توان به روشهای مختلف مورد حمله قرار داد تا اصل پیامها بازیابی شود. با اندکی کار بیشتر، Keystream متناظر با آن IV بدست خواهد آمد. ترودی می‌تواند کار خود را اندکی ادامه بدهد و یک دیکشنری برای Keystream متناظر با تمام IVها تشکیل دهد. پس از شکسته شدن IV، تمام بسته‌هایی 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 ترودی بتواند یک زوج (*Keystream* و *IV*) 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 مبادلهٔ 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 اجازهٔ چنین کاری را داده است[1]؛ 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 ترودی 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 سرراست است. ما به تمام فهرست حملاتی که آقای Borisov کشف کرده نپرداختیم.

در آگوست سال ۲۰۰۱، یک ماه پس از انتشار مقالهٔ Borisov، یک حملهٔ ویرانگر بر علیه WEP توسط Fluhrer ، تدوین 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 تئوریک این ادعا، Fluhrer و گروهش هیچ تلاشی برای شکستن یک شبکهٔ محلی مبتنی بر ۸۰۲.۱۱ نکرده بود.

در مقابل، وقتی یک دانشجوی کارآموز و دو محقق در آزمایشگاه AT&T از حملهٔ 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 Flahrer 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 شبکهٔ اترنت نیست!!
۲. همین امنیت ناقص بهتر از نبود آنست!
۳. سعی کنید از روشهای امنیتی دیگر بهره بگیرید. (مثلاً امنیت در سطح لایهٔ انتقال)
۴. نسخهٔ 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 شبکهٔ ۸۰۲.۱۱ دارد لذا در این شبکه نمی‌توان با پارک در پارکینگ یک مؤسسه و استراق سمع داده‌ها به آن حمله کرد ولی کماکان امنیت بلوتوث مورد مهمی بشمار می‌رود. به عنوان مثال فرض کنید که کامپیوتر آلیس مجهز به یک صفحه کلید بی‌سیم مبتنی بر بلوتوث است. اگر در این شبکه امنیت وجود نداشته باشد ، ترودی در دفتر مجاور آلیس ، می‌تواند هر آنچه را آلیس تایپ می‌کند (حتی نامه‌های ارسالی او) را بخواند. او همچنین می‌تواند هر آنچه را که آلیس برای چاپ بر روی «چاپگر بلوتوث» می‌فرستد بدست آورد (مثل نامه‌های دریافتی یا گزارشهای محرمانه). خوشبختانه بلوتوث ، ساختار امنیتی دقیقی دارد و تلاش می‌کند تا فعالیتهای اخلالگران دنیا را خنثی کند. در ادامه ویژگیهای اصلی این ساختار را به اختصار بررسی می‌نماییم.

بلوتوث سه حالت امنیتی متفاوت در محدودهٔ «بدون رمزنگاری» تا «رمزنگاری کامل» و «امکان بررسی صحت داده‌ها» دارد. همانند شبکهٔ ۸۰۲.۱۱ هر گاه گزینهٔ امنیت غیرفعال شده باشد (که به صورت پیش فرض این‌گونه است) هیچ امنیتی برای داده‌ها وجود ندارد. بسیاری از کاربران گزینهٔ امنیت را غیرفعال نگه می‌دارند تا وقتی که یک اشکال جدی برایشان اتفاق بیفتد؛ پس از آن امنیت را فعال می‌کنند. در دنیای کشاورزی به این بی‌احتیاطی «بستن در اصطبل پس از فرار اسب» گفته می‌شود!!!

بلوتوث امنیت را در چندین لایه عرضه کرده است: در لایهٔ فیزیکی تعویض مستمر فرکانس (Frequency Hopping) امنیت ناچیزی عرضه می‌کند 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 می‌شود 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 می‌کند. فرض شده که این دو دستگاه دارای یک کلید مشترک سرّی هستند که از قبل درون آنها درج شده است. در برخی از حالات این کلید مشترک به صورت سخت‌افزاری توسط کارخانهٔ سازنده بر روی آن ابزار ذخیره می‌شود. (به عنوان مثال همانند شماره‌ای که به صورت پیش‌فرض بر روی گوشی تلفن همراه وجود دارد.) در برخی دیگر از حالات یکی از دستگاهها دارای کلیدی تعبیه شده بر روی سخت‌افزار است در حالی که کاربر (برای فعال کردن دستگاه) مجبور است این کلید را در قالب عددی دهدهی وارد کند. این کلیدهای مشترک اصطلاحاً **«کلیدهای عبور»** (Passkeys) نامیده می‌شوند.

برای ایجاد یک کانال، ماشین اصلی (Master) و ماشین «پیرو» (Slave) هر یک بررسی می‌کنند که آیا دیگری کلید عبور را می‌داند؟ 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 می‌کنند. سپس یک کلید ۱۲۸ بیتی که برخی از بیتهای آن عمومی و آشکار است برای نشست انتخاب می‌نمایند. 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، اجازهٔ 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[1] که E_0 نام دارد، انجام می‌شود. بررسی صحت اطلاعات نیز به روش SAFER+ انجام می‌گیرد. این دو روش براساس روش معمولی رمزنگاری با کلید متقارن هستند. SAFER+ برای مسابقهٔ AES (که نهایتاً به پیروزی Rijndael انجامید) ارسال شد ولیکن در همان مراحل مقدماتی از دور مسابقه خارج گردید چراکه از بقیهٔ روشهای پیشنهادی کُندتر بود. طراحی شبکهٔ بلوتوث قبل از

---

۱. بخش ۸-۲-۳ را ببینید.

انتخاب برندهٔ AES، پایان یافته بود وگرنه به احتمال زیاد در آن از روش Rijndael استفاده می‌شد.

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 (Stream Key) XOR شده و متن رمز را بدست می‌دهد. متأسفانه E_0 نیز (شبیه به RC4) می‌تواند اشکالات اساسی داشته باشد (Jokobson & Wetzel, 2001). اگرچه هنوز E_0 شکسته نشده است (تا زمان نوشتن این کتاب) ولیکن شباهتهای آن با رمز A5/1 که اِشکال واضح آن ترافیک تلفنهای همراه GSM را در معرض حمله قرار داده است به این نگرانی دامن می‌زند. گاهی این موضوع افراد را سردرگم و متعجب می‌کند 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 می‌کند نه کاربر را، فلذا یک دستگاه دزدیده شدهٔ مبتنی بر بلوتوث می‌تواند به سارق اجازهٔ دسترسی به حساب کاربری و دیگر امکانات صاحب آن دستگاه را بدهد. با این حال، بلوتوث امنیت را در لایه‌های بالاتر نیز پیاده‌سازی کرده است؛ بنابراین حتی اگر امنیت در لایهٔ پیوند داده نابود شود، در لایه‌های بالاتر باقی خواهد ماند؛ بالاخص در برخی از برنامه‌های کاربردی این ویژگی مثبت وجود دارد که گاهی برای آن که کاربر بتواند کاری را انجام بدهد از او کد PIN (شمارهٔ شناسایی شخصی) مطالبه می‌کنند.

امنیت در WAP 2.0

در اکثر بخشها، مجمع توسعه دهندهٔ WAP از غیراستاندارد بودن پشتهٔ پروتکلی WAP 1.0 درس گرفته و به همین دلیل WAP 2.0 در تمام لایه‌ها به طرز گسترده‌ای از پروتکلهای استاندارد استفاده می‌کند. از آنجایی که WAP مبتنی بر IP است در لایهٔ شبکه به طور کامل از IPsec حمایت می‌کند. در لایهٔ انتقال نیز، از یک اتصال TCP به کمک TLS حفاظت می‌شود. (TLS استاندارد IETF است که در همین فصل بدان خواهیم پرداخت.) در لایهٔ بالاتر از روش «احراز هویت HTTP» که در RFC 2617 تشریح شده است، بخوبی حمایت می‌شود. وجود یک کتابخانهٔ سیستمی (Library) در سطح لایهٔ کاربرد به منظور رمزنگاری، امکانات کافی جهت کنترل صحت و غیرقابل انکار بودن پیامها را در اختیار برنامه‌نویسان 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 داده‌ها، غیرقابل انکار بودن و محرمانه ماندن پیامها، از امنیت 802.11 و Bluetooth بهتر و مطمئن‌تر باشد.

۸-۷ پروتکلهای احراز هویت

«احراز هویت» (Authentication) روشی است که براساس آن یک پروسه بررسی می‌کند که آیا شریک او در یک ارتباط (یعنی پروسهٔ طرف مقابل)، همانی است که باید باشد یا یک نفوذگرست که خود را به جای طرف واقعی جا زده است. بررسی هویت واقعی یک پروسهٔ راه دور در شرایطی که با اخلالگران فعال و بدخواه روبرو هستیم فرآیندی بسیار دشوار است و به پروتکلهای پیچیدهٔ مبتنی بر رمزنگاری نیاز دارد. در این بخش برخی از پروتکلهای بی‌شمار احراز هویت را که در شبکه‌های ناامن مورد استفاده قرار می‌گیرد، مطالعه خواهیم کرد.

گاهی مردم اصطلاح «**احراز هویت**» (Authentication) را با «**صدور مجوز**» (Authorization) اشتباه می‌گیرند. «احراز هویت» با این سؤال سر و کار دارد که آیا شما حقیقتاً در حال محاوره و تبادل اطلاعات با یک پروسهٔ خاص هستید. «صدور مجوز» با این مقوله سر و کار دارد که یک پروسه، مجوز انجام چه کارهایی را دارد. به عنوان مثال، یک پروسهٔ مشتری با یک سرویس دهندهٔ فایل ارتباط برقرار کرده و اعلام می‌کند: «من پروسهٔ «اسکات» هستم و می‌خواهم فایل *cookbook.old* را پاک کنم.» از دیدگاه سرویس‌دهندهٔ فایل پاسخ دو سؤال

بایدمشخص شود:

۱. آیا این پروسه حقیقتاً پروسه «اسکات» است؟ (احراز هویت)
۲. آیا «اسکات» اجازهٔ 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 حساس‌تر و کلیدی‌تر است. پس از آن که سرویس دهندهٔ فایل متوجه شد که با چه کسی صحبت می‌کند، 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 پروتکلهای احراز هویت از آن استفاده می‌کنند بدین نحو است که مثلاً: آلیس کارش را با ارسال پیامی به باب یا یک «KDC **مورد اعتماد**»[1] که صادق و امین همه است، شروع می‌کند. چندین پیام دیگر بین طرفین و در دو جهت مبادله می‌شود. ممکن است در حالی که این پیامها در حال مبادله هستند شخص ثالثی مثل ترودی مشغول استراق سمع، دستکاری در پیام یا تکرار پیامها باشد تا بدین نحو آلیس یا باب را فریب بدهد یا در کار آنها اخلال کند.

علیرغم تمام این کارشکنی‌ها، 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 سرّی نشست**» ایجاد می‌کنند تا در محاورهٔ خود از آن [برای رمزنگاری] استفاده نمایند. در عمل و به دلایل سرعت و کارآیی، تمام ترافیک داده‌ها در حین نشست به روش رمزنگاری با کلید متقارن (عموماً AES یا DES) رمز می‌شوند، اگرچه از روش «رمزنگاری کلید عمومی» در پروتکلهای احراز هویت برای ایجاد «کلید نشست» در سطح گسترده‌ای استفاده می‌شود.

دلیل آنکه برای هر «اتصال» جدید یک کلید تصادفی به عنوان کلید نشست انتخاب می‌شود آنست که حجم ترافیکی که مستقیماً توسط کلید سرّی یا کلید عمومی کاربر رمز می‌گردد حداقل بماند و در نتیجه میزان اطلاعات رمز شده (که در تمام آنها از یک کلید استفاده شده) کاهش یابد. همچنین هر گاه پروسه‌ای دچار اشکال شده و درهم بشکند (crash کند)، ممکن است «تصویر حافظهٔ» آن پروسه در اختیار افراد ناباب قرار بگیرد و آنها بتوانند کلید اصلی کاربر را از درون آن استخراج کنند.[2] در این حالت حتی اگر کلیدی فاش شود کلید نشست خواهد بود که آنهم ثابت نیست. تمام کلیدهای ثابت و دائمی پس از آن که نشست برقرار شد از حافظه پروسه‌ها پاک شده و فقط از کلید نشست استفاده می‌شود.

۸-۷-۱ احراز هویت براساس کلید مشترک و سرّی

در اولین پروتکل احراز هویت، فرض می‌کنیم که آلیس و باب قبلاً در مورد یک کلید سرّی به نام K_{AB} با یکدیگر توافق کرده‌اند. 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 سرّی را از طریق شبکهٔ ناامن، برای یکدیگر ارسال نکرده‌اند. این پروتکل بر اصولی استوار است که تمام پروتکلهای دیگر احراز هویت نیز از آن تبعیت می‌کنند: «یکی از طرفین عددی تصادفی برای دیگری ارسال می‌کند و طرف مقابل تبدیل خاصی را بر روی آن اعمال کرده و نتیجه را بر می‌گرداند». چنین پروتکلهایی اصطلاحاً پروتکلهای «**چالش-پاسخ**» (Challenge-Response) نامیده می‌شوند.

---

۱. KDC: مرکز توزیع کلید یا Key Distribution Center

۲. در سیستمهای عاملی مثل یونیکس هرگاه پروسه‌ای دچار اشکال شده و crash کند، کل فضای حافظهٔ در اختیار آن پروسه بر روی دیسک سخت ذخیره می‌شود 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[1] آلیس و باب هستند.

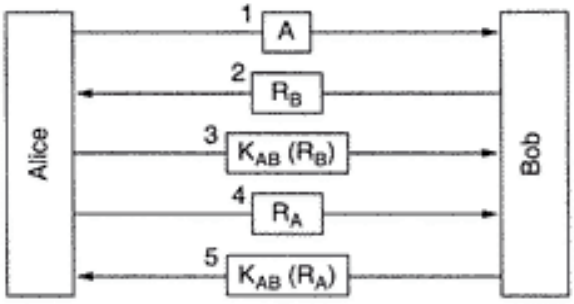
R_i رشته‌های «چالش» (Challenge) هستند که پانویس آنها یعنی i فرستندهٔ آن را مشخص می‌کند.

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 مبادلهٔ پیامها نشان داده شده است. در پیام ۱، آلیس مشخصهٔ شناسایی خود یعنی A را به گونه‌ای برای باب می‌فرستد که او بتواند آن را بفهمد (بصورت آشکار و بدون رمزنگاری). البته باب (فعلاً) 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 رشتهٔ **«چالش»** (Challenge) انتخاب کرده و آنرا در پیام شمارهٔ ۲ بصورت آشکار به آلیس بر می‌گرداند. اعداد تصادفی بکار رفته در پروتکلهای «چالش-پاسخ» مثل این پروتکل، اصطلاحاً *nonce* نامیده می‌شوند. آلیس پیام شمارهٔ ۲ را با کلید مشترک خود رمزنگاری کرده و داده‌های رمز شده یعنی $K_{AB}(R_B)$ را در پیام شمارهٔ 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 رشتهٔ «چالش» را [بدون داشتن کلید] حدس بزند وجود ندارد.

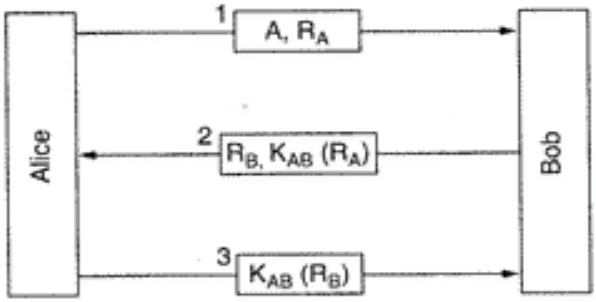
در این جا، باب مطمئن شده که در حال صحبت با آلیس است ولی آلیس از هیچ چیز مطمئن نیست زیرا ممکن است ترودی پیام شماره ۱ را استراق سمع کرده باشد و در پاسخ R_B را برگرداند. اصلاً ممکن است باب، شب قبل فوت کرده باشد!! برای آن که آلیس بداند که در حال صحبت با چه کسی است عدد تصادفی R_A را انتخاب و در پیام شماره 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

۱. Identifier

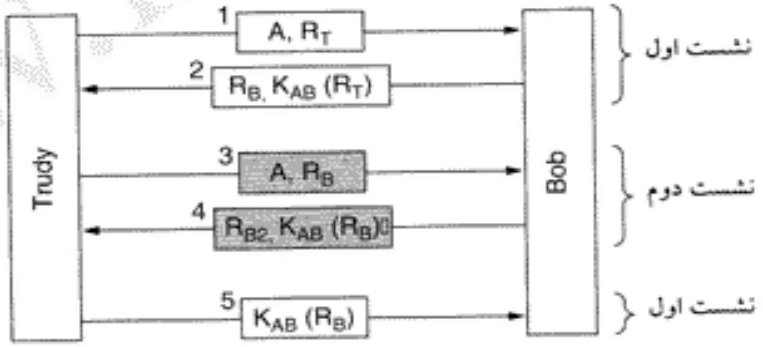


شکل ۸-۳۳. پروتکل دومرحله‌ای احراز هویت با تعداد مراحل کمتر.

مراحل «چالش-پاسخ» توسط باب شود خودش شخصاً این کار را شروع می‌کند.[1] به روش مشابه، باب وقتی به چالش آلیس پاسخ می‌دهد، رشتهٔ چالش خودش یعنی R_B را نیز برای آلیس می‌فرستد. بدین ترتیب کل پروتکل به جای پنج مرحله به سه مرحله کاهش می‌یابد.

آیا پروتکل جدید مزیتی بر پروتکل اصلی دارد؟ به صورت حسّی شاید بگوییم کوتاهتر و سریعتر است. متأسفانه اشتباه است! در شرایط خاص ترودی می‌تواند این پروتکل را با استفاده از تکنیکی نام **«حملهٔ 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 می‌افتد که به عنوان مثال، باب یک بانک باشد و طبعاً باید بتواند ارتباط همزمان چندین ماشین خودپرداز (Teller Machine) را بپذیرد.

«حملهٔ بازتاب» که توسط ترودی انجام می‌شود در شکل ۸-۳۴ نشان داده شده است. او حملهٔ خود را با ارسال R_T و اعلام آن که آلیس است ، شروع می‌کند. باب طبق معمول پاسخ این چالش را به همراه رشتهٔ چالش خود یعنی R_B برمی‌گرداند. حال ترودی گیر می‌افتد. او $K_{AB}(R_B)$ را نمی‌داند. پس چه کاری می‌تواند انجام بدهد؟



شکل ۸-۳۴. حملهٔ بازتاب.

او می‌تواند نشست دومی را با ارسال پیام ۳ شروع نماید و R_B بدست آمده از مرحلهٔ 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 می‌فرستد. در شکل ۸-۳۴، پیامهای نشست دوم را برای تمایز از نشست اول، خاکستری نشان داده‌ایم. حال ترودی اطلاعاتی را که برای نشست اول کم داشت در اختیار دارد لذا می‌تواند 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، رشتهٔ چالش یعنی R_A را برای باب می‌فرستد.

آلیس را تقاضا می‌کند، باب بی‌هیچ پرسشی اطاعت می‌نماید. وقتی ترودی مثلاً از باب می‌خواهد که تمام موجودی او را به یک حساب محرمانه در سوئیس 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 قاعدهٔ 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 ترودی مدرکی در خصوص هویت خود ارائه داده باشد، اطلاعات با ارزشی را از دست می‌دهد.

۲. شروع کننده و پاسخ دهنده 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 پاسخ دهنده را وادار کنید که رشته‌های «چالش» خود را از مجموعه‌های متفاوتی انتخاب نمایند. مثلاً شروع کننده مجبور باشد اعداد زوج را انتخاب کند و پاسخ دهنده 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.

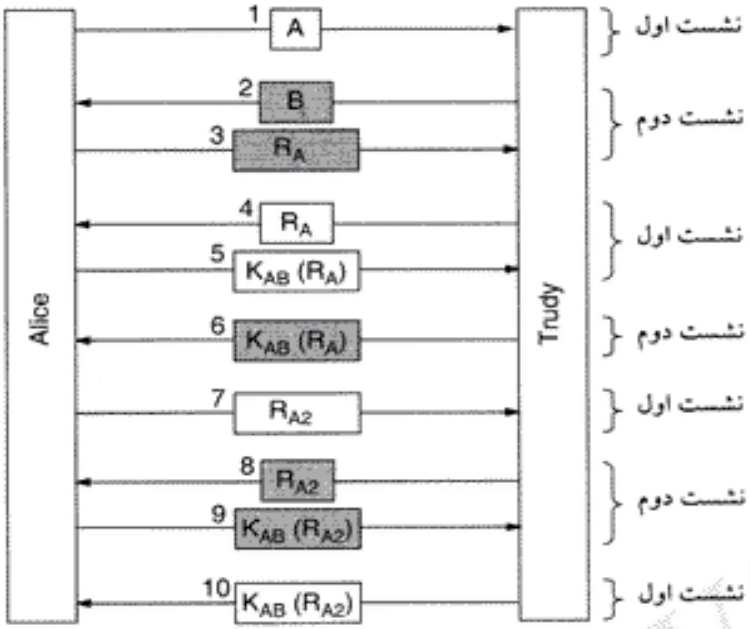
حتی اگر یکی از این چهار قاعده نقض شود، پروتکل به کرّات قابل شکستن خواهد بود. در پروتکل سه مرحله‌ای مثال قبل هر چهار قاعده نقض شده بود و بالطبع نتایج خطرناکی به بار خواهد آورد.

حال اجازه بدهید به شکل ۸-۳۲ بازگردیم و نگاهی دقیقتر به آن بیندازیم. آیا این پروتکل مطمئناً در معرض **«حملهٔ بازتاب»** (Reflection Attack) قرار نمی‌گیرد؟ بستگی دارد! قضیه کمی پیچیده است! ترودی می‌تواند این پروتکل را با استفاده از حملهٔ 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 ترودی چه می‌تواند بکند.

برای آنکه متوجه شوید که حملهٔ ترودی به چه نحو خواهد بود به شکل ۸-۳۵ نگاه کنید. آلیس با اعلام مشخصهٔ شناسایی خود برای باب، کارش را شروع می‌کند. ترودی 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 ترودی گیر افتاده چون نمی‌تواند ثابت کند باب است. ترودی چه کاری می‌تواند انجام بدهد؟

او به نشست اول بر می‌گردد؛ در آنجا نوبت اوست که رشتهٔ **«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 ترودی برای ادامهٔ نشست دوم بدان نیاز داشته است، لذا آنرا از طریق نشست دوم برای آلیس می‌فرستد. در اینجا ترودی در نشست دوم پاسخ موفقیت‌آمیزی را برای آلیس ارسال می‌کند. حال او می‌تواند نشست اول را لغو کرده و باقیماندهٔ مراحل نشست دوم را تکمیل کند و بدین ترتیب یک نشست موفق و مورد تایید با آلیس خواهد داشت. (نشست ۲)

ولیکن ترودی پلیدتر از این حرفهاست و می‌خواهد مردم‌آزاری خود را ادامه بدهد!!! به جای آنکه اعدادی قدیمی و بی‌ارزش را برای تکمیل نشست ۲ ارسال کند، منتظر می‌ماند که آلیس در ادامهٔ نشست اول رشتهٔ چالش خود یعنی R_{A2} را برایش بفرستد. اگرچه ترودی نمی‌داند که در پاسخ به آن چه باید برگرداند ولیکن باز هم از



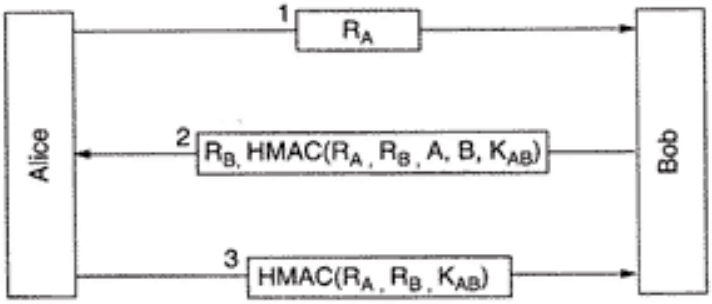
شکل ۸-۳۵. حملهٔ بازتاب بر علیه پروتکل شکل ۸-۳۲.

«حملهٔ بازتاب» بهره می‌گیرد و R_{A2} را در پیام هشتم ارسال می‌نماید. آلیس براحتی آن را رمز کرده و در پیام نهم بر می‌گرداند. ترودی مجدداً به نشست اول بازگشته و عدد بدست آمده از پیام نهم را در پیام ۱۰ برای آلیس پس می‌فرستد. در این لحظه ترودی دو نشست آماده و کامل با آلیس ترتیب داده است.

نتیجهٔ این حمله، با حمله به پروتکل سه‌مرحله‌ای شکل ۸-۳۴ متفاوت است. در اینجا ترودی دو ارتباط تائید شده و کامل با آلیس دارد در حالی که در مثال قبلی او فقط یک ارتباط تائید شده با باب برقرار کرده بود. در اینجا نیز اگر چهار قاعدهٔ اساسی در پروتکلهای احراز هویت که قبلاً عنوان کردیم رعایت می‌شد، چنین حمله‌ای ممکن نبود. تشریح کامل این نوع از حملات و چگونگی خنثی‌سازی آنها در مرجع (Bird et.al, 1993) آمده است. آنها نشان داده‌اند که می‌توان پروتکلهای متفاوتی را ایجاد کرد که صحّت عملکردشان قابل اثبات باشد. ساده‌ترین پروتکل از این‌گونه ، تا حدودی پیچیده است به همین دلیل ما در اینجا ردهٔ متفاوتی از پروتکلهای 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[1] 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 می‌کند. باب با انتخاب R_B برای خود، آن رادر قالب رشتهٔ درهم‌شدهٔ HMAC برای آلیس پس می‌فرستد. HMAC به گونه‌ای سازماندهی شده است که یک ساختمان داده شامل: R_A ارسالی توسط آلیس، R_B متعلق به باب و مشخصه‌های شناسایی هر دو و همچنین کلید مشترک آنهایعنی K_{AB} را در بر می‌گیرد. این ساختمان داده به روشی مثل SHA-1 درهم شده و در HMAC قرار می‌گیرد. وقتی آلیس پیام ۲ را دریافت می‌کند، R_A (که خودش در ابتدا انتخاب کرده بود)، R_B که به صورت آشکار برایش ارسال شده بود ، دو مشخصهٔ شناسایی (مشخصه خودش و باب) و کلید محرمانه را در اختیار دارد و بدین ترتیب می‌تواند بطور مستقل HMAC (یعنی رشتهٔ Hash) را برای این آیتمها محاسبه نماید. اگر HMAC محاسبه شده با

۱. Hashed Message Authentication Code



شکل ۸-۳۶. احراز هویت با استفاده از روش HMAC.

HMAC ارسالی توسط باب معادل باشد، آلیس می‌تواند مطمئن باشد که در حال صحبت با باب است زیرا ترودی K_{AB} را نمی‌داند و بالطبع نمی‌تواند HMAC را به صورت جعلی و دروغین ساخته و آن را از طرف باب ارسال نماید. آلیس نیز در پاسخ ، HMAC مربوط به سه آیتم R_A، R_B، K_{AB} را ارسال می‌کند. (باز هم ترودی نمی‌تواند این پیام را جعل کند، چون کلید K_{AB} را در اختیار ندارد.)

آیا ترودی می‌تواند به طریقی این پروتکل را شکست بدهد؟ خیر؛ چون او نمی‌تواند طبق وقایعی که در شکل ۸-۳۵ اتفاق افتاد، طرفین را وادار کند تا مقادیر دلخواه او را رمزنگاری یا رشتهٔ Hash را محاسبه نماید. هر دوی HMACها شامل مقادیری هستند که در کنترل ترودی نیست.

استفاده از HMAC تنها روش بهره‌گیری از ایدهٔ فوق نیست. اغلب به جای محاسبهٔ HMAC از روشی جایگزین استفاده می‌شود که در آن دنبالهٔ آیتمها به روش «**زنجیره‌سازی بلوکهای رمز**» رمز و ارسال می‌شوند.

۸-۷-۲ ایجاد کلید مشترک: مبادلهٔ کلید به روش «دیفی-هلمن»

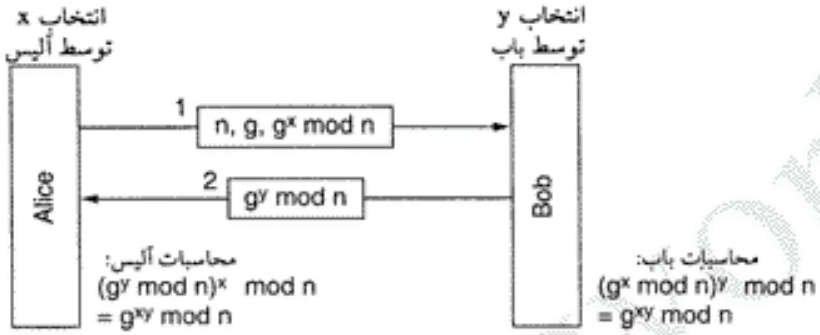
تا اینجا فرض کرده‌ایم که آلیس و باب بر روی یک کلید سرّی توافق کرده‌اند. حال فرض را بر آن بگذارید که آنها چنین کاری نکرده‌اند زیرا مثلاً هیچ مرکز جهانی و مورد وفاق PKI برای امضاء و توزیع گواهینامه‌های دیجیتالی وجود ندارد. در چنین حالتی چگونه این دو نفر می‌توانند یک کلید مشترک ایجاد کنند. یک راه آن است 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 می‌توانند یک ملاقات ترتیب داده و هر کدام، گذرنامه یا گواهینامهٔ رانندگی خود و سه کارت اعتباری معتبر و مهم را همراه آورده و پس از تائید هویت یکدیگر، بر روی یک کلید مشترک توافق نمایند. برای مردم گرفتار، شاید تنظیم قرار ملاقات برای ماهها امکان‌پذیر نباشد. خوشبختانه راه حلّی برای توافق و ایجاد کلید سرّی بین افراد ناآشنا با یکدیگر وجود دارد که آنها می‌توانند در روز روشن و حتی وقتی ترودی قادر به استراق سمع و ضبط تمام پیامهاست، یک کلید سرّی ایجاد کنند، هر چند این موضوع ممکن است اندکی غیرقابل باور به نظر برسد.

پروتکلی که به افراد غریبه و ناآشنا با یکدیگر، اجازه می‌دهد یک کلید مشترک و سرّی ایجاد کنند، پروتکل «**مبادلهٔ کلید دیفی ـ هلمن**»[1] نام دارد و به صورت زیر عمل می‌کند: آلیس و باب می‌باید 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 اوّل است و شرایط خاصی نیز بر روی g اعمال می‌شود. این دو عدد عمومی و غیرسرّی بوده و هر کسی می‌تواند 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 سرّی نزد

۱. Diffie-Hellman key exchange; (Diffie and Hellman, 1976)

خود نگاه می‌دارد. به همین روش باب نیز یک عدد سرّی مثل y برای خود انتخاب می‌کند.

آلیس، طبق شکل ۸-۳۷، پروتکل مبادلهٔ کلید را با ارسال پیامی شامل آیتمهای (n و g و $g^x \bmod n$) آغاز می‌کند. باب نیز در پاسخ، پیامی را ارسال می‌کند که در برگیرندهٔ $g^y \bmod n$ است. حال آلیس، عدد ارسالی توسط باب را در پیمانهٔ n به توان x می‌رساند 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 می‌دهد. طبق روابط حاکم بر نظریهٔ اعداد، هر دو نفر حاصل $g^{xy} \bmod n$ را بدست خواهند آورد. دقّت کنید! آلیس و باب به ناگاه صاحب یک کلید مشترک و سرّی با فرمول $g^{xy} \bmod n$ شده‌اند.

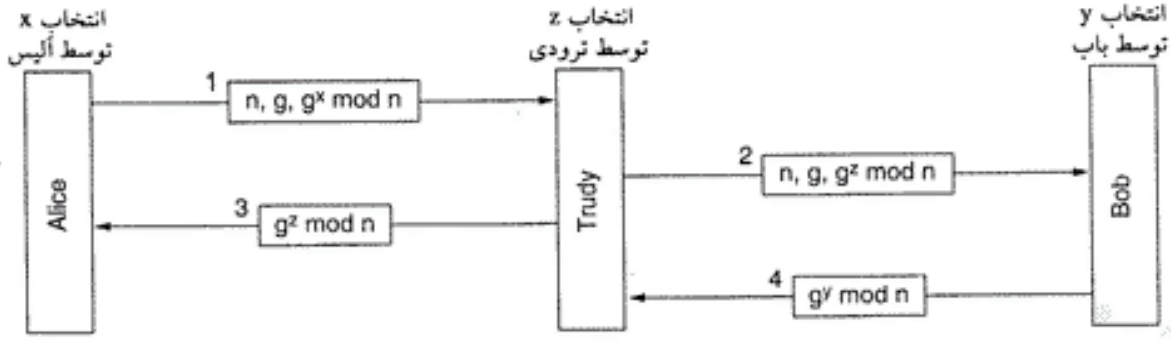


شکل ۸-۳۷. مبادلهٔ کلید بروش دیفی-هلمن.

البته در این میان ترودی هر دو پیام را می‌بیند 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 سرّی را تعیین نماید ولیکن مشکل اینجاست که با داشتن $g^x \bmod n$ نمی‌توان x را پیدا کرد زیرا هیچ الگوریتم عملی برای محاسبهٔ لگاریتم گسسته در پیمانهٔ اعداد اوّل بسیار بزرگ، تاکنون کشف نشده است.

برای آن که مثال بالا را بهتر تفهیم کنیم از اعداد $n=47$ و $g=3$ استفاده می‌نمائیم (این اعداد در عمل بسیار کوچک و غیرقابل استفاده‌اند). آلیس عدد $x=8$ و باب عدد $y=10$ را برای خود انتخاب می‌کند؛ هر دوی این اعداد سرّی نگه داشته می‌شوند. پیام آلیس به باب شامل آیتمهای (۲۸ و ۳ و ۴۷) خواهد بود چراکه حاصل $3^8 \bmod 47$ معادل ۲۸ است. پیام باب به آلیس عدد ۱۷ است. آلیس مقدار $17^8 \bmod 47$ را محاسبه می‌کند که ۴ بدست می‌آید. باب نیز حاصل $28^{10} \bmod 47$ را بدست می‌آورد که باز هم ۴ است. در اینجا باب و آلیس هر دو کلید سرّی و مشترک ۴ را دارند که به صورت مستقل بدست آمده است. برای بدست آوردن کلید سرّی، ترودی مجبور است معادلهٔ $3^x \bmod 47 = 28$ را حل کند که فقط با جستجوی کامل [در محدودهٔ صفر تا n-1] امکان‌پذیر است و اگر n عددی بسیار بزرگ در حدّ چند صد بیت باشد، حل این معادله عملاً ممکن نخواهد بود. حتی اگر از ابرکامپیوترهای موازی استفاده شده باشد، تمام الگوریتمهای شناخته شدهٔ امروزی برای حلّ این معادله، نیاز به وقت بسیار زیادی دارند.

علیرغم زیبائی الگوریتم «دیفی ـ هلمن» ، مشکلی در آن وجود دارد: وقتی باب سه آیتم (۲۸ و ۳ و ۴۷) را دریافت می‌کند از کجا بداند که واقعاً از طرف آلیس پیشنهاد شده است نه از طرف ترودی؟ هیچ راهی برای این تشخیص وجود ندارد! متأسفانه ترودی می‌تواند به گونه‌ای 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 می‌کنند، ترودی نیز z را برای خود بر می‌گزیند. حال آلیس پیام اول را برای باب می‌فرستد ولیکن این پیام در میانهٔ راه توسط ترودی دریافت و متوقف می‌شود. ترودی g و n (که عمومی و غیرسرّی هستند) را به همراه $g^z \bmod n$ (به جای $g^x \bmod n$) برای



شکل ۸-۳۸. حملهٔ «گروه آتش‌نشان» (یا حملهٔ 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.

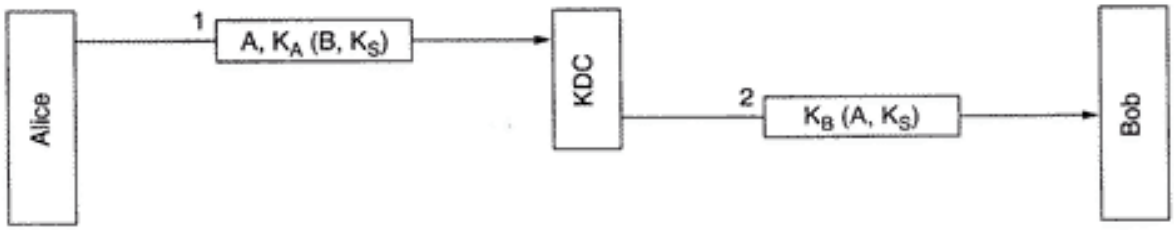
حال هر سه نفر محاسبات پیمانه‌ای (Modular Arithmetic) خود را آغاز می‌کنند. آلیس و ترودی کلید سرّی $g^{xz} \bmod n$ را محاسبه می‌کنند. از طرف دیگر باب و ترودی نیز $g^{yz} \bmod n$ را به عنوان کلید سرّی خود محاسبه می‌نمایند. آلیس در این خیال است که با باب صحبت می‌کند لذا یک کلید نشست با ترودی ایجاد کرده 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 **«حملهٔ گروه آتش‌نشان»** (Bucket Brigade Attack) مشهور است زیرا تقریباً به انتقال سطلهای آب از کامیون به محل آتش‌سوزی شباهت دارد که آتش‌نشانهای داوطلب در یک صف، سطلهای آب را دست به دست هدایت می‌کنند. این حمله همچنین به نام حملهٔ **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 سرّی 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 وفاق 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 واسطهٔ KDC انجام می‌شود. در شکل ۸-۳۹ ساده‌ترین پروتکل احراز هویتِ مبتنی بر KDC نشان داده شده که شامل یک مرکز معتمدِ 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 سرّی و مشترک بین آلیس و KDC که آن را K_A نامیده‌ایم رمز می‌شود. KDC این پیام را رمزگشایی کرده و مشخصهٔ شناسایی باب و کلید نشست را از درون آن استخراج می‌نماید. سپس KDC، پیام جدیدی را می‌سازد و در آن مشخصهٔ 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 بکمک KDC.

مشترک بین باب و KDC انجام می‌شود. وقتی باب این پیام را رمزگشایی کند، متوجه می‌شود که آلیس تمایل به محاوره با او دارد و کلید رمزی را که برای محاوره باید استفاده شود، بدست می‌آورد.

این روش احراز هویت بسیار ساده انجام می‌شود: KDC می‌داند که پیام قاعدتاً 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 می‌داند که پیام ۲ از طرف KDC که مورد اعتماد اوست صادر شده چون هیچکس کلید سرّی او را در اختیار ندارد (مگر KDC).

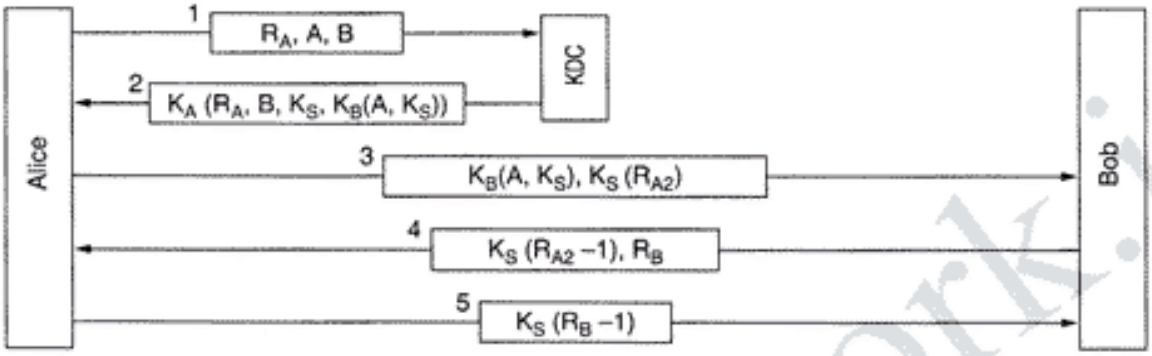
متأسفانه این پروتکل دارای یک اشکال جدّی است. به سناریوی زیر دقت کنید. ترودی به مقداری پول احتیاج دارد لذا به گونه‌ای برنامه‌ریزی می‌کند که خدماتی متعارف برای آلیس انجام بدهد و سپس به استخدام موقت او در می‌آید. پس از انجام کار، ترودی مؤدبانه از آلیس خواهش می‌کند تا حقوقش را به حساب بانکی او واریز نماید. آلیس با بانکدار خود یعنی باب ، یک کلید نشست ایجاد کرده و سپس از باب تقاضا می‌کند که پول درخواستی را از حساب او به حساب ترودی واریز نماید.

در این میان ، ترودی به راه و روش قبلی خود یعنی جاسوسی و پرسه زدن در شبکه برمی‌گردد. 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 می‌کند. ترودی بعداً این دو پیام را بار دیگر برای باب تکرار می‌کند. باب این پیامها را دریافت کرده و با خود می‌اندیشد که آلیس ترودی را برای بار دیگر به استخدام خود در آورده است لذا همان مقدار پول را از حساب بانکی آلیس به حساب ترودی واریز می‌کند. پس از دریافت پنجاهمین پیام، باب از دفترش بیرون می‌آید تا ترودی را پیدا کرده و برای توسعهٔ کار تجارت به او پیشنهاد وام بدهد!!! به این نوع حمله اصطلاحاً **«حملهٔ تکرار»** (Replay Attack) گفته می‌شود.

چندین راه حل برای جلوگیری از حملهٔ تکرار وجود دارد. اولین راه حل آن است که هر پیام دارای **«مُهر زمان»** (Timestamp) باشد. در این صورت هرگاه کسی یک پیام قدیمی دریافت کرد می‌تواند براحتی آنرا حذف کند. یکی از مشکلات این روش آن است که نمی‌توان در یک شبکه، همهٔ ساعتها را بدقت با هم تنظیم کرد لذا باید به گونه‌ای برنامه‌ریزی نمود که مهر زمان در یک بازهٔ مشخص [مثلاً پانزده دقیقه] اعتبار داشته باشد و بدین ترتیب ترودی در این بازهٔ 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 رشتهٔ 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 رشته‌ها باید برای همیشه نگه‌داری شوند مبادا ترودی پیامی مربوط به پنج سال قبل را به صورت تکراری بفرستد. همچنین اگر ماشینی درهم بشکند و فهرست این اعداد یا رشته‌ها از دست برود در معرض «حملهٔ تکرار» قرار خواهد گرفت. می‌توان از ترکیب «مُهر زمان» و «الصاق عدد یا رشتهٔ تصادفی چالش» (nonce) در پیامها استفاده کرد. بدین ترتیب طول مدتی که باید این اعداد و رشته‌ها حفظ شوند کاهش پیدا می‌کند ولیکن پروتکل پیچیده‌تر 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 ـ 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)، مشخصهٔ 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. مشخصهٔ شناسایی باب بدان جهت درون این پیام جاسازی شده که اگر تروری در میانهٔ راه، پیام شمارهٔ یک را دستکاری نموده و مشخصهٔ 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 تروری) 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 تروری 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 تروری 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 «حملهٔ تکرار» (Replay) منتفی است. [در این ساختار، KDC نیز نقش کوتاه ولی بسیار موثری ایفاء می‌کند.]

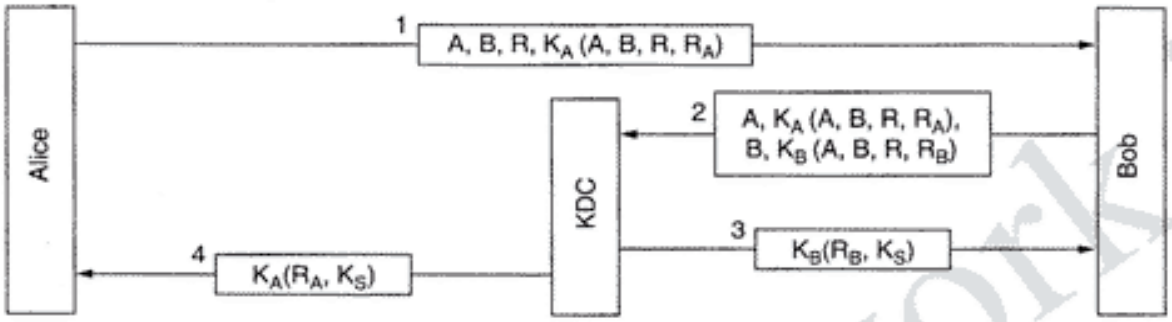
اگرچه این پروتکل بسیار محکم به نظر می‌رسد ولیکن یک ضعف بسیار جزیی دارد. اگر تروری بتواند با برنامه‌ریزی یک کلید نشستِ قدیمی که در گذشته از آن استفاده شده را بنحوی بدست بیاورد[1] می‌تواند به صورت

---

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 مرحلهٔ سوم شروع کرده و نشست جدیدی را با باب شروع کند و او را متقاعد نماید که آلیس است. در اینجا تروِدی می‌تواند بدون هیچ زحمت اضافه‌ای، 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 مجلهٔ 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 ساده‌تر و کوتاه‌تر حل می‌کرد. شکل ۸-۴۱ پروتکل «اتوی-ریس» را با تغییر جزیی نشان می‌دهد.



شکل ۸-۴۱. پروتکل احراز هویت «اتوی-ریس».

در پروتکل اتوی-ریس، آلیس کار خود را با تولید دو عدد تصادفی بزرگ آغاز می‌کند: R که به عنوان یک شناسهٔ مشترک بکار می‌رود و R_A که آلیس از آن به عنوان رشتهٔ «چالش» (Challenge) استفاده می‌نماید. وقتی باب پیام اول را [طبق شکل] دریافت می‌کند، با استفاده از بخش رمزنگاری شدهٔ پیام آلیس، پیام جدیدی را ساخته و برای KDC می‌فرستد. در این پیام، باب آیتمی مشابه با بخش رمزنگاری شدهٔ پیام اول تولید و به آن می‌افزاید.[1] K_A و K_B کلیدهای محرمانهٔ آلیس و باب هستند و آیتمهای A و B و R و R_A یا R_B با آنهارمز می‌شوند.

حال از آنجایی که KDC کلید رمز هر دو نفر را در اختیار دارد، با دریافت این پیام ابتدا بررسی می‌کند که آیا R بدست آمده از دو پیام رمزنگاری شده یکی است یا خیر، زیرا ممکن است به دلیل دستکاری در پیام اول توسط تروِدی، با هم مغایرت داشته باشند. اگر هر دو R با هم یکی بودند KDC متقاعد می‌شود که پیام ارسالی از باب [که در حقیقت تقاضای احراز هویت به حساب می‌آید] 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 می‌فرستد و بار دیگر به همراه R_B با کلید باب رمز کرده و برای باب ارسال می‌دارد. هرکدام از این پیامها (پیام ۳ و ۴) شامل عدد تصادفی تولید شده توسط گیرندهٔ آن پیام است که ثابت می‌کند واقعاً KDC آن را تولید کرده و به صورت جعلی یا تکراری توسط تروِدی ارسال نشده است. در این لحظه آلیس و باب دارای کلید مشترکی هستند که توسط KDC پیشنهاد شده و می‌توانند با اطمینان محاورهٔ خود را آغاز نمایند. در اولین پیامی که بین آنها ردّ و بدل می‌شود آنها می‌توانند 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 می‌شود.

۸-۷-۴ احراز هویت با استفاده از Kerberos

یک پروتکل احراز هویت که در بسیاری از سیستمهای واقعی (مثل ویندوز ۲۰۰۰) بکار گرفته می‌شود، Kerberos است که براساس گونه‌ای از پروتکل «نیدهام ـ شرودر» بنا نهاده شده است. نام Kerberos برگرفته از یک اسطورهٔ

---

۱. یعنی پیام دوم شامل $K_A(A,B,R,R_A)$ و A است که توسط آلیس ارسال شده و در آن دخل و تصرفی نمی‌شود و همچنین شامل $K_B(A,B,R,R_B)$ و B است که خودش آنرا می‌سازد. ـم.

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 اجازهٔ دسترسی مطمئن به منابع شبکه را می‌دهد. مهمترین تفاوت آن با پروتکل «نیدهام ـ 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. نسخهٔ V4 آن کاربرد بسیار گسترده‌ای در صنعت دارد به همین دلیل، آن را توضیح می‌دهیم. سپس چند کلمه‌ای در خصوص نسخهٔ 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 دهندهٔ دیگر سر و کار دارد:

۱. **سرویس دهندهٔ احراز هویت یا** *AS* (Authentication Server): این سرویس دهنده کاربران را در حین ورود به سیستم (Login) بازرسی می‌نماید.

۲. **سرویس دهندهٔ صدور بلیط یا** *TGS* (Ticket Granting Server): این سرویس دهنده بلیطهای تائید هویت صادر می‌کند.

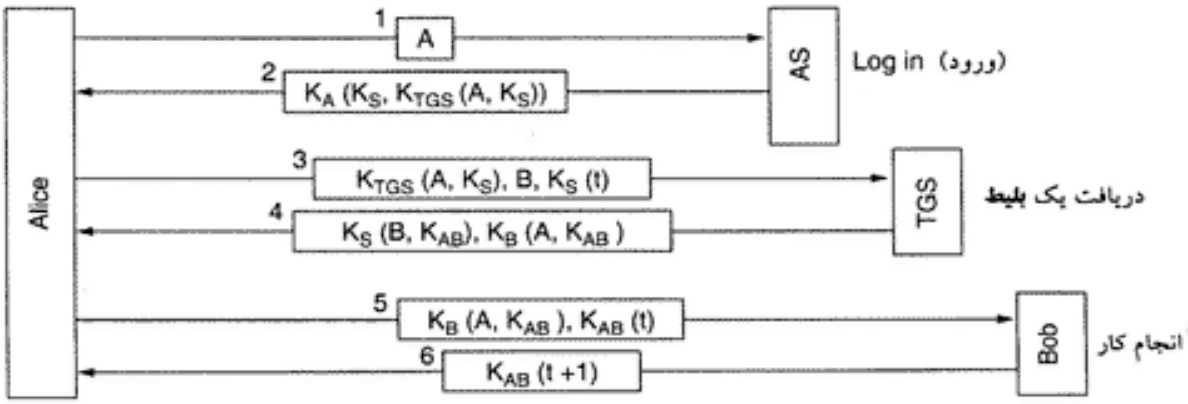
۳. **سرویس دهندهٔ باب** (یا هر سرویس دهنده‌ای که خدماتی را ارائه می‌کند): این سرویس دهنده کاری را که آلیس از او می‌خواهد انجام می‌دهد.

سرویس دهندهٔ AS شبیه به KDC عمل می‌کند که در آن کلیدهای سرّی تمام کاربران شبکه نگهداری می‌شود. کار سرویس دهندهٔ TGS آن است که بلیطهای معتبری را برای کاربران صادر کند تا سرویس‌دهنده‌های 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 دهندهٔ 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 دهندهٔ TGS تحویل داده شود. این دو آیتم درون یک پیام جاسازی شده و با استفاده از کلید سرّی آلیس رمز می‌شود فلذا هیچکس جز آلیس نمی‌تواند آن را رمزگشایی کند. وقتی پیام ۲ دریافت شود، ایستگاه از آلیس کلمهٔ عبور او را سؤال می‌کند. سپس از کلمهٔ 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 کلمهٔ عبور آلیس را از حافظه پاک می‌کند تا بیش از چند میلی ثانیه در حافظهٔ ایستگاه باقی نماند. اگر ترودی سعی کند با نام آلیس وارد شود (Login کند)، کلمهٔ عبوری که درج می‌کند غلط خواهد بود و ایستگاه این موضوع را براحتی کشف خواهد کرد چراکه بدنهٔ 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 دهندهٔ فایل باب، ارتباط برقرار کند. در اینجا ایستگاه پیام ۳ را به سرویس دهندهٔ TGS می‌فرستد و از او می‌خواهد تا بلیطی برای استفاده از سرویس دهندهٔ باب صادر نماید. نکتهٔ اساسی در این تقاضا $K_{TGS}(A,K_S)$ است که با کلید سرّی سرویس‌دهندهٔ TGS رمزنگاری شده و محتوای آن ثابت می‌کند که فرستندهٔ بلیط واقعاً آلیس است. سرویس دهندهٔ 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. نسخهٔ دوم با کلید سرّی باب یعنی K_B رمز می‌شود که هیچکس جز باب نمی‌تواند آن را بخواند.

ترودی می‌تواند پیام ۳ را دریافت و کپی کند و تلاش نماید تا آن را مجدداً بکار بگیرد ولیکن تلاش او با وجود مُهر زمان t که رمزنگاری نیز شده است، بی‌ثمر خواهد ماند. ترودی 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 می‌کند، بی‌اطلاع است. حتی اگر ترودی 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 می‌تواند 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 می‌کند ثابت خواهد کرد که واقعاً با باب محاوره می‌کند نه ترودی. [بدین نحو هویت باب تائید می‌شود.]

پس از این چند مرحله مبادلهٔ پیام (جمعاً ۶ پیام) آلیس می‌تواند در پوشش کلید رمز K_{AB} با باب تبادل اطلاعات داشته باشد. هر گاه او تصمیم بگیرد که با یک سرویس دهندهٔ دیگر محاوره نماید (مثلاً سرویس دهندهٔ کارول) از پیام سوم شروع می‌کند با این تفاوت که به جای مشخصهٔ شناسایی B در این پیام، C را درج می‌کند. TGS سریعاً برای او بلیطی صادر می‌کند که به جای K_B با K_C رمز شده است و آلیس می‌تواند آن را برای کارول بفرستد و کارول نیز آن را به عنوان یک سند معتبر که از طرف آلیس ارسال شده می‌پذیرد.

نکتهٔ ارزشمند در این ساختار آنست که آلیس می‌تواند به تمام سرویس‌دهنده‌های موجود در شبکه به روشی مطمئن دسترسی داشته باشد و به هیچ وجه کلمهٔ عبور او بر روی شبکه منتقل نخواهد شد. در حقیقت کلمهٔ عبور او فقط برای چند میلی ثانیه درون ایستگاهی که او در کنار آن قرار دارد، ذخیره می‌شود. به این نکته دقت کنید که هر سرویس دهنده بعداً احراز هویت خاص خود را انجام می‌دهد یعنی وقتی آلیس بلیط خود را به سرویس‌دهندهٔ باب عرضه می‌کند، تنها کاری که این بلیط انجام می‌دهد آن است که برای باب ثابت کند که او واقعاً آلیس است. حال کارهایی که آلیس مجاز به انجام آن است صرفاً توسط سرویس دهندهٔ باب تعیین می‌شود.

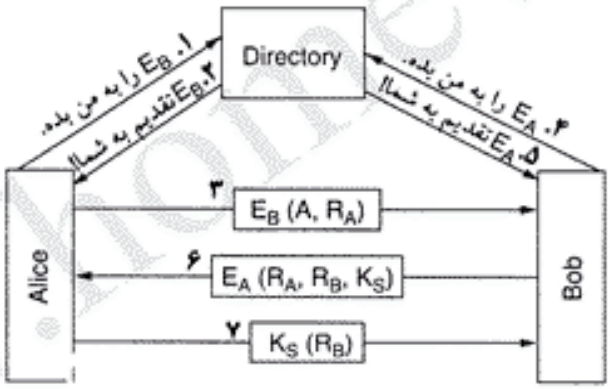
از آنجایی که طراحان Kerberos توقع نداشتند که کل دنیا به یک سرویس دهندهٔ واحد و جهانی احراز هویت اعتماد کنند، لذا این امکان را فراهم کردند که تعدادی ناحیهٔ مستقل وجود داشته باشد و هر ناحیه از یک AS و TGS خاصّ خود استفاده کند. آلیس برای آن که بتواند بلیطی را برای یک سرویس دهنده در ناحیه‌ای دور به دست بیاورد، از سرویس‌دهندهٔ TGS خودش می‌خواهد که بلیطی برایش صادر کند که موردِ پذیرش TGS ناحیهٔ راه دور باشد. اگر TGS ناحیهٔ دور در TGS محلی ثبت شده باشد (به همان روشی که سرویس دهنده‌های محلی ثبت می‌شوند)، TGS محلی به آلیس بلیطی می‌دهد که نزد TGS دیگر معتبر است. او می‌تواند با این بلیط کار دلخواهش را در آن TGS انجام بدهد مثلاً می‌تواند از آن TGS بلیط دسترسی به یک سرویس‌دهندهٔ دیگر در آن ناحیه را دریافت نماید. بهرحال برای آن که طرفین در دو ناحیهٔ مختلف بتوانند کارشان را انجام بدهند باید به TGS

یکدیگر اعتماد داشته باشند.

نسخهٔ V5 از Kerberos، مفصلتر از نسخهٔ V4 است و زحمت و سربار زیادتری 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 زیادتر شده و اجازه داده شده بلیطها تجدید شوند. به علاوه پروتکل V5 ، لااقل در تئوری به DES وابسته نیست (در حالی که V_4 این‌گونه است) و از وجود نواحی مختلف و تفویض صدور کلید به سرویس‌دهنده‌های این نواحی حمایت می‌نماید.

۸-۷-۵ 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[1] (مرکز توزیع کلید عمومی) با ساختار «سرویس‌دهندهٔ 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، بازمی‌گردد. پس از آن که آلیس صَحتِ امضای باب را بررسی و تائید کرد، برای او پیامی می‌فرستد که در آن مشخصهٔ 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 ترودی، ولی به همان روش عمل کرده و از «سرویس‌دهندهٔ 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 ترودی 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 غیرتکراری 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 ترودی می‌تواند این پروتکل را درهم بریزد و در آن رسوخ کند؟ او می‌تواند 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 ترودی می‌دانند] ولی وقتی آلیس در پیام ششم R_A را مشاهده می‌کند متوجه می‌شود که چنین عددی را او نفرستاده و به همین دلیل مراحل کار را ادامه نخواهد داد. ترودی قادر نخواهد بود پیام هفتم را به صورت جعلی تولید کرده و برای باب بفرستد، چون R_B و K_S را نمی‌داند 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 نامه‌های 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 استفاده‌های بعدی ذخیره کنند. برخلاف آنچه که بسیاری از مردم می‌اندیشند، حریم خصوصی عملاً وجود ندارد ولیکن علیرغم این بسیاری از افراد علاقمندند نامه‌هایی را که ارسال می‌کنند 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 نامه‌های 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 غنی‌تر در مورد سیستمهای امن پست الکترونیکی، مرجع (Kaufman et. al. 2002; Schneier, 1995) را ملاحظه نمائید.

۸-۸-۱ PGP (Pretty Good Privacy)

PGP ، به عنوان اولین مثال از سیستم پست الکترونیکی امن، زائیدهٔ تفکر شخصی به نام «زیمرمَن» (Phil Zimmermann,1995) بود. شعار زیمرمن که یکی از طرفداران حفظ حریم خصوصی افراد است، این بود که: «اگر ایجاد حریم خصوصی و حفظ اسرار مردم قانون‌شکنی است آنگاه فقط حریم خصوصی قانون‌شکنان حفظ خواهد شد.» PGP یک بستهٔ نرم‌افزاری کامل برای امنیت نامه‌های الکترونیکی است که در سال ۱۹۹۱ منتشر شد و با یک ساختار بسیار سادهٔ کاربری، تمام امکانات ذیل شامل «**تدوین نامه‌های خصوصی**» (رمزشده)، «**احراز هویت**»، «**امضای دیجیتالی**» و حتی «**فشرده‌سازی اطلاعات**» را به صورت یکجا عرضه کرده است. به علاوه، بستهٔ نرم‌افزاری بهمراه کدهای برنامه، بصورت رایگان از طریق اینترنت در دسترس عموم قرار می‌گیرد. امروزه به دلیل کیفیت بالا، عدم هزینه (قیمت صفر) و وجود نسخه‌های متفاوت بر روی یونیکس، لینوکس، ویندوز و سیستم عامل مکینتاش (Mac OS)، از آن به صورت گسترده‌ای استفاده می‌شود.

PGP داده‌ها (محتویات نامه) را با استفاده از روشی به نام IDEA[1] که مبتنی بر رمزنگاری بلوکی است رمز می‌کند و در آن از کلیدهای ۱۲۸ بیتی و متقارن بهره می‌گیرد. این روش رمزنگاری در سوئیس زمانی ابداع شد که ضعف و اشکالات DES ارزش آن را خدشه‌دار کرده بود ولی هنوز AES اختراع و معرفی نشده بود. IDEA شبیه به DES و AES است: در چندین دور (Round) بیتهای داده را با هم ترکیب می‌کند ولیکن جزئیات توابع ترکیب بیتها، در مقایسه با DES و AES متفاوت است. در PGP، برای مدیریت کلیدها از RSA و برای بررسی صحت داده‌ها 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 زیمرمن

---

۱. International Data Encryption Algorithm

هیچ اقدامی برای آن که افراد را از توزیع PGP بر روی اینترنت منع کند انجام نداده بود و هر کسی در هر نقطهٔ دنیا می‌توانست 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 زیمرمن پنج سال طول کشید ولی نهایتاً پرونده مختومه شد؛ شاید به دو دلیل: اول آن که زیمرمن 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 زیمرمن بدان معنا تلقی می‌شود که یک وب‌سایت که برنامه‌ای قابل دریافت (Downloadable) در خصوص امنیت ارائه می‌دهد در شمول قانون منع فروش تجهیزات استراتژیک مثل تانک، زیردریایی، هواپیماهای نظامی و موشکهای هسته‌ای 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 زیمرمن را به مدت پنج سال گرفتار دادگاه کرد در حالیکه اگر کسی کدهای برنامهٔ PGP را به زبان C در یک کتاب تدوین (و آن را با حروف بزرگ و قابل اسکن چاپ می‌نمود) و سپس آن کتاب را صادر می‌کرد توسط دولت فقط جریمه می‌شد چرا که کتاب در ردهٔ تجهیزات جنگی قرار نمی‌گیرد. به هر حال لااقل برای عموسام قدرت شمشیر از قلم بیشتر است.

یکی دیگر از مشکلات حقوقی PGP، مسئلهٔ نقض مالکیت امتیاز بود. شرکتی که امتیاز RSA را برای خود ثبت کرده بود (شرکت RSA Security Inc.) اقامهٔ دعوی کرد که PGP با استفادهٔ بدون مجوز از الگوریتم RSA، مرتکب نقض مالکیتِ حقوق معنوی آن شده است ولی از نسخهٔ 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.

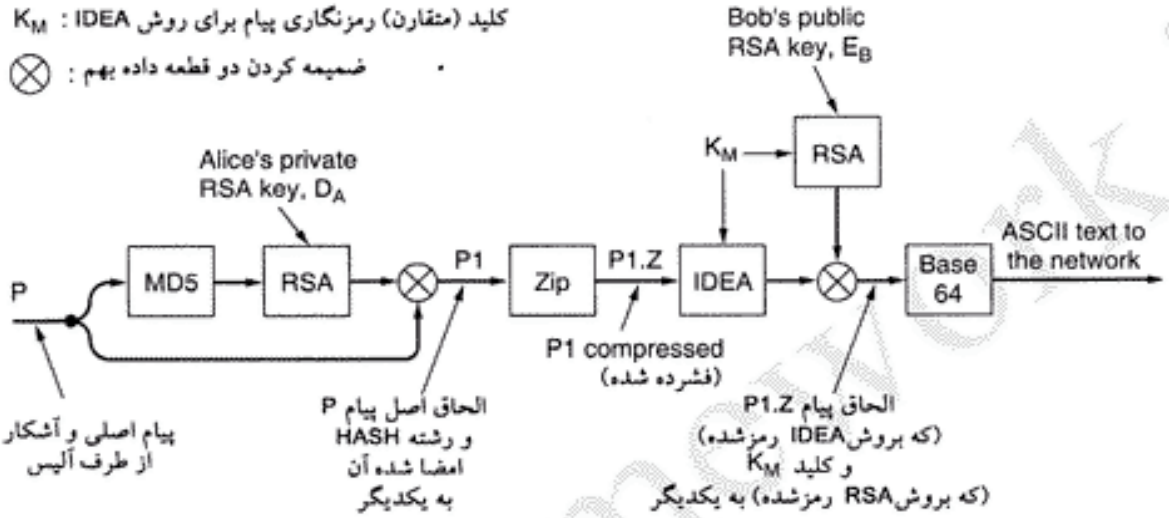
از آنجا که کدهای برنامهٔ PGP آزاد است لذا افراد و گروههای مختلف، نسخه‌های متفاوتی از آن را تولید و عرضه کردند. برخی از این نسخه‌ها با این هدف طراحی شده‌اند که قوانین «منع صدور تجهیزات حساس» را دور بزنند و برخی دیگر بر آن متمرکز بوده که از الگوریتمهای ثبت مالکیت شده استفاده نشود و حتی برخی دیگر خواستند 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 نسخه‌های کاملاً ناسازگار از PGP با نامهای مختلف به جریان بیفتد. توضیح زیر بر روی نسخهٔ کلاسیک PGP که قدیمیترین و ساده‌ترین نسخهٔ آن است متمرکز خواهد بود. نسخهٔ رایج دیگر آن یعنی Open PGP در RFC 2440 تشریح شده است. همچنین نسخهٔ 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 نبوده‌اند. برای افرادی که به حکومت اعتماد ندارند این ویژگی، امتیاز بزرگی محسوب می‌شود.

PGP از فشرده‌سازی متن، سرّی‌سازی آن و امضاهای دیجیتالی حمایت می‌کند و امکانات بسیار جالبی در خصوص مدیریت کلید عرضه کرده است ولی امکانات کمی در خصوص تدوین و ارسال نامهٔ الکترونیکی دارد. PGP را می‌توان یک «**پیش پردازنده**» تلقی کرد که متن اصلی را به عنوان ورودی می‌گیرد و یک متن امضاء شده سرّی را که در قالب base64 کد شده است، باز می‌گرداند. البته خروجی آن بعداً می‌تواند توسط پست الکترونیکی

ارسال شود. برخی از پیاده‌سازیهای PGP در آخرین مرحله می‌توانند «عامل کاربر» [مثلاً نرم‌افزاری مثل Outlook] را جهت ارسال پیام فراخوانی کنند.

برای آن که ببینیم PGP چگونه کار می‌کند، اجازه بدهید مثال شکل ۸-۴۴ را بررسی کنیم. در این شکل آلیس می‌خواهد 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. فعلا فرض می‌کنیم هر کدام، کلید عمومی دیگری را می‌دانند؛ بعداً در خصوص مدیریت کلیدها به اختصار توضیح خواهیم داد.



شکل ۸-۴۴. عملکرد PGP برای ارسال یک پیام.

آلیس پشت کامپیوترش نشسته و کارش را با فراخوانی برنامه PGP آغاز می‌کند. PGP ابتدا به روش MD5 پیام P را درهم ریخته (Hash کرده) و یک رشتهٔ خلاصهٔ پیام (Message Digest) استخراج نموده و سپس آنرا طبق روش RSA با کلید خصوصی خودش D_A رمز می‌کند. بعداً وقتی باب پیام را دریافت کند می‌تواند رشتهٔ Hash را با کلید عمومی آلیس رمزگشایی کرده و صحّت آن را بررسی نماید. حتی اگر شخص ثالثی (مثل تروودی) 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 صحّت و سلامت پیام (Integrity) تضمین می‌شود.

رشتهٔ Hash رمز شده و پیام اصلی به یکدیگر ضمیمه می‌شوند و یک پیام واحد به نام *P1* را ایجاد می‌کنند. سپس *P1* با استفاده از برنامهٔ ZIP که از الگوریتم «لیمپل-زیو» استفاده می‌کند فشرده می‌شود. (Lempel - Ziv, 1977). خروجی این مرحله را *P1.Z* بنامید.

PGP در ادامه، از آلیس یک ورودی تصادفی مطالبه می‌کند. 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 می‌شود. (K_M در ادبیات PGP کلید نشست نامیده می‌شود ولی در واقع کلید نشست، اسم بی‌مسمائی است چراکه در ارسال نامه هیچ نشستی ترتیب داده نمی‌شود). حال پیام *P1.Z* به روش IDEA و در حالت Feedoack Mode[1]، با کلید K_M رمزنگاری می‌شود. مضاف براین، خود K_M نیز با استفاده از کلید عمومی ب می E_B رمز می‌شود. نهایتاً این دو مولفه (یعنی کلید رمز+پیام فشرده و رمزشده) بهم ضمیمه شده و سپس در مبنای base64 که در بخش MIME از

۱. بخش ۸-۲۰-۳ را ببینید.

فصل هفتم تشریح کردیم، کدگذاری و تبدیل به متن ASCII می‌شود. پیام حاصل فقط شامل حروف الفباء، ارقام و سمبولهای '+'، '='، '/' خواهد بود بدین معنا که می‌تواند براحتی در بدنهٔ یک نامهٔ RFC 822 قرار گرفته و بدون هیچ تغییری به مقصد برسد.

وقتی باب پیام را دریافت می‌کند ابتدا عکس عمل کدگذاری base64 را انجام داده و کلید رمز IDEA را با کلید خصوصی RSA خود، استخراج می‌نماید. سپس با استفاده از این کلید، بدنهٔ پیام را رمزگشایی می‌کند تا *P1.Z* را بدست بیاورد. پس از آن که متن از حالت فشرده خارج شد، باب متن اصلی را از رشتهٔ رمزنگاری شدهٔ Hash جدا کرده و این رشته را رمزگشایی می‌کند. اگر رشتهٔ Hash متن اصلی با رشته MD5 ارسالی تطابق داشت او از صحت پیام P مطمئن شده و یقین حاصل می‌کند 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 رشتهٔ ۱۲۸ بیتی MD5 (۲) برای رمزنگاری کلید ۱۲۸ بیتی IDEA. اگرچه الگوریتم RSA بسیار کُند 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، ترودی به تلاش بسیار زیادی، نیازمند خواهد بود. در عوض حجم سنگین عملیات رمزنگاری بدنهٔ پیام، به روش IDEA که بارها از RSA سریعتر است، انجام می‌شود. PGP، امنیت، فشرده‌سازی 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 می‌دهد.

PGP از کلیدهای RSA با چهار اندازهٔ مختلف حمایت می‌کند. انتخاب طول مناسب بر عهدهٔ کاربر گذاشته شده است. این چهار انتخاب عبارتند از:

۱. **کلیدهای عِلّی (۳۸۴ بیت)**: امروزه می‌توان براحتی آن را شکست.

۲. **کلیدهای تجاری (۵۱۲ بیت)**: فقط توسط مؤسسات سه حرفی قابل شکستن است.[۱]

۳. **کلیدهای نظامی (۱۰۲۴ بیتی)**: هیچکس بر روی کره زمین نمی‌تواند آن را بشکند.

۴. **کلیدهای کیهانی (۲۰۴۸ بیتی)**: هیچکس حتی در سیارات دیگر نمی‌تواند آن را بشکند!!

از آنجایی که RSA فقط در دو محاسبهٔ کوچک [یعنی برای رمزنگاری ۲کلید ۱۲۸بیتی] بکار گرفته می‌شود منطقی است که عموم افراد از کلید با طول ۲۰۴۸ بیت استفاده کنند.

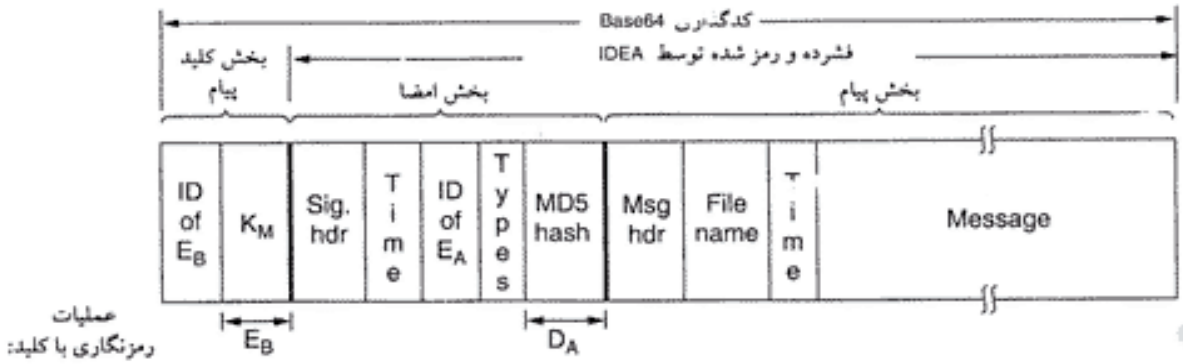
قالب یک پیام کلاسیک PGP در شکل ۸-۴۵ نشان داده شده است. البته از قالبهای بیشمار دیگری نیز استفاده می‌شود. پیام در سه بخش شامل بخش کلید IDEA، بخش امضاء و بخش بدنهٔ پیام سازماندهی شده است. بخش حاوی کلید نه تنها کلید را در بر می‌گیرد بلکه شمارهٔ شناسایی کلید عمومی را نیز شامل می‌شود. بدین ترتیب کاربران مجاز 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 شمارهٔ شناسایی کلید عمومی فرستنده درج می‌شود که توسط این کلید عمومی، رشتهٔ Hash امضاء دیجیتالی از رمز خارج خواهد شد. سپس در ادامه، یک فیلد Type قرار داده شده که نوع الگوریتم بکار رفته را تعیین می‌کند (تا مثلاً اجازه بدهد در صورت ابداع MD6 یا RSA2 از آنها استفاده شود!) سپس در انتهای بخش امضاء، رشتهٔ کد MD5 پیام قرار می‌گیرد.

بخش پیام نیز دارای یک فیلد سرآیند است، سپس نام پیش فرض فایل درج می‌شود تا اگر گیرندهٔ پیام خواست آنرا بر روی دیسک ذخیره کند از این نام استفاده شود. سپس مهر زمانِ ایجادِ پیام، درج شده و نهایتاً پیام اصلی قرار

---

۱. منظور از مؤسسات سه حرفی NSA، FBI، CIA و امثال آنهاست! -م.



شکل ۸-۴۵. قالب پیامهای PGP.

می‌گیرد.

در PGP به مدیریت کلیدها توجه ویژه‌ای شده است چرا که پاشنهٔ آشیل تمام سیستمهای امنیتی محسوب می‌شود! مدیریت کلیدها در PGP بدین نحو انجام می‌گیرد که هر کاربر به صورت محلی از دو ساختمان دادهٔ خاص نگهداری می‌کند: الف) «دسته کلید خصوصی» ب) «دسته کلید عمومی». **«دستهٔ کلید خصوصی»** شامل یک یا چند زوج کلید عمومی و همتای خصوصی آنهاست. دلیل آن که از چندین زوج کلید برای هر کاربر حمایت می‌شود 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 (شمارهٔ شناسایی) متناظر است که براساس آن فرستندهٔ پیام به گیرنده اطلاع می‌دهد که برای رمزگشایی پیام از کدام کلید باید استفاده کند. شناسهٔ کلیدها، ۶۴ بیتِ کم‌ارزش 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 کلمهٔ عبور (با اندازهٔ طولانی) رمز می‌شوند تا از خطر سرقت مصون بمانند.

«دسته کلید عمومی» در برگیرندهٔ کلیدهای عمومی متعلق به کاربران طرف مقابل ارتباط است. برای رمزنگاری کلیدهای هر پیام، به کلید عمومی گیرندهٔ پیام نیز نیاز است. هر درایه (Entry) در دستهٔ کلیدهای عمومی، نه تنها شامل خود کلید است بلکه شناسهٔ ۶۴بیتی 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 برمی‌گیرد. در زیر فلسفهٔ این فیلد مشخص شده است.

مسئله‌ای 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 ترودی بتواند نامه‌های محرمانهٔ 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 می‌کند، ترودی خواهد توانست در میانهٔ راه نامه‌های او را استراق سمع نماید.[۱]

برای پیشگیری از این گونه حملات یا حداقل کاهش تبعات آن، آلیس نیازمند آن است که بداند به چه اندازه می‌تواند به آیتمی که در دستهٔ کلیدهای عمومی، به باب منسوب شده اعتماد کند. فیلد **«میزان اعتماد»** به همین منظور بکار می‌رود. اگر آلیس کلید عمومی باب را بر روی یک فلاپی دیسک شخصاً از باب گرفته باشد، می‌تواند فیلد **«میزان اعتماد»** این کلید را به مقدار بالایی تنظیم نماید. این روش مدیریت کلید که بصورت غیرمتمرکز و تحت نظارت کاربر انجام می‌شود ، 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 دهندهٔ مورداعتماد توزیع کلید،

---

۱. حملهٔ «گروه آتش‌نشان» (bucket brigade) را در بخش ۸-۷-۲ مطالعه نمائید.

بدست می‌آورند. به همین دلیل پس از آن که X.509 استاندارد شد، PGP در کنار مکانیزم سنتی خود از گواهینامهٔ دیجیتالی X.509 نیز حمایت کرد. تمام نسخه‌های فعلی PGP از گواهینامه‌های X.509 پشتیبانی می‌کنند.

۸-۸-۲ PEM (Privacy Enhanced Mail)

برخلاف PGP که از ابتدا یک نمایش تکنفره با هنرنمایی زیمرمن بود، دومین مثال ما، PEM، یک استاندارد رسمی اینترنت است که در اواخر دههٔ 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 نامه‌ها در سیستمهای پست الکترونیکی مبتنی بر RFC 822 هستند؛ ولیکن در روشها و فناوری بکار رفته در آن با PGP تفاوتهایی دارد.

پیامهای ارسالی توسط PEM در ابتدا به قالب استاندارد تبدیل می‌شود به نحوی که تمام نامه‌ها از قواعد و استاندارد مشابهی در خصوص فضاهای خالی (همانند فاصله‌ها، Tab و...) پیروی کنند. سپس با استفاده از MD2 یا MD5 برای نامه یک رشته Hash بدست می‌آید. پس از الحاق رشته Hash به پیام، مجموع این دو با استفاده از روش DES رمزنگاری می‌شود. با در نظر داشتن ضعف کلیدهای ۵۶بیتی در DES، این انتخاب کاملاً مشکوک به نظر می‌رسد. سپس پیام رمز شده به روش base64 کُدگذاری و برای گیرندهٔ آن ارسال می‌شود.

همانند PGP، هر پیام با استفاده از کلید یک‌مرحله‌ای و متفاوت رمز شده و آن کلید نیز در کنار پیام جاسازی و ارسال می‌شود. این کلید با استفاده از روش رمزنگاری RSA یا روش Triple DES (DES سه گانه مبتنی بر مکانیزم EDE) رمز می‌گردد.

مدیریت کلید در PEM نسبت به PGP سازمان یافته‌تر است. کلیدها باگواهینامه‌های X.509 که توسط مراکز صدور گواهی -CA- صادر شده‌اند، تصدیق می‌شوند. مراکز صدور گواهی در یک ساختار سلسله مراتبی که از مرکزی به نام «ریشه» (Root) شروع می‌شود، سازماندهی شده‌اند. حسن این ساختار آنست که ابطال گواهینامه‌ها با انتشار متناوب فهرست CRL[1] توسط «ریشه»، بسادگی میسر می‌باشد.

تنها مشکل PEM آن است که هیچکس تاکنون از آن استفاده نکرده و تقریباً به [illegible]خ 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 می‌ترسند. جدی‌ترین نامزد این مسئولیت، شرکت RSA بود که می‌خواست 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 می‌شد به شرکت RSA پول بپردازند. نهایتاً هیچکس برای پذیرش مسئولیت «ریشه» [در ساختار سلسله مراتبی صدور گواهینامه] پیدا نشد و PEM شکست خورد.

۸-۸-۳ 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»، «سرّی نگاه داشتن پیام» و «غیرقابل انکار بودن پیام» را تضمین کرده است. همچنین این روش کاملاً قابل انعطاف

---

۱. CRL: «فهرست گواهینامه‌های باطل شده»

بوده و از الگوریتمهای رمزنگاری مختلفی پشتیبانی می‌کند. همانگونه که از نام S/MIME مشخص است این روش، استانداردِ MIME را تکمیل و امن کرده است فلذا عجیب نیست که از انواع مختلف پیامها حمایت و حفاظت می‌کند.[۱] در این استاندارد تعدادی سرآیند جدید برای MIME تعریف شده که برای عملیاتی نظیر مشخص کردن امضای دیجیتالی پیام، کاربرد دارند.

IETF قطعاً از تجربهٔ 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 مؤسسهٔ واحد به عنوان ریشه، مجموعه‌ای از مؤسسات مورد اعتمادِ صدور گواهینامه (Trust Anchors) را به خدمت بگیرند. هر گاه در سلسله‌مراتب تائید گواهینامه‌های دیجیتالی، گواهینامه‌ای به یکی از این مؤسسات مورداعتماد کاربر ختم گردد، معتبر فرض می‌شود. S/MIME از پروتکلها و استانداردهایی استفاده می‌کند که تا اینجا همهٔ آنها را بررسی کرده‌ایم و بیش از این، بدانها نخواهیم پرداخت. برای دسترسی به شرح مبسوط آن به RFCهای مربوطه مراجعه نمائید.

۹-۸ امنیت وب

تا اینجا دو زمینهٔ بسیار مهم را که مقولهٔ امنیت در آنها ضروری است، مطالعه کرده‌ایم: امنیت در مخابرهٔ 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 اخلالگران 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 کُد 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. بیائید مثالهایی از آنچه که در گذشته اتفاق افتاده را بررسی کنیم. اولاً صفحهٔ وب اصلی (Home Page) بسیاری از مؤسسات مورد حمله قرار گرفته و با صفحات وب دلخواه اخلالگران (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 اخلالگر یا «Cracker» بنامیم.) سایتهایی که تاکنون درهم شکسته و در آنها اخلال شده شامل سایتهای مشهوری مثل Yahoo ، U.S.Army ، CIA ، NASA و New York Times بوده است. در بسیاری موارد، اخلالگران 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 حملهٔ DoS (حملهٔ نوع اخلال در سرویس‌دهی) از کار افتاده‌اند و اخلالگر 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 اخلالگران بوده‌اند[۱] (حملهٔ نوع DDoS[۲]). 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 اخلالگر سوئدی در سایت وب Microsoft Hotmail رسوخ کرد و با ایجاد یک سایت معادل (Mirror Site) که در آن کاربران نام کاربری و کلمهٔ 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، اخلالگر ۱۹ سالهٔ روسی که ماکسیم نام داشت توانست به یک سایت تجارت الکترونیکی نفوذ کند و شمارهٔ 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 خواستهٔ او اعتنایی نکردند و او نیز با لجاجت شمارهٔ کارتهای اعتباری دیگران را بر روی اینترنت گذاشت و ضرر و زیان زیادی به شماری از قربانیان بی‌گناه تحمیل کرد.

در یک حملهٔ دیگر، یک دانشجوی ۲۳ ساله کالیفرنیایی گزارشی را برای یک آژانس خبری ارسال کرد و به دروغ عنوان نمود که شرکت «امولکس» (Emulex Corporation) می‌خواهد در تراز اقتصادی سه ماههٔ 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 معاملهٔ 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 صفحهٔ 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 بپردازیم. برای اطلاعات بیشتر در خصوص انواع مختلف و مستند مشکلات امنیتی وب به مراجع (Anderson, 2001; Garfinkel with Spafford, 2002; Schneier, 2000) مراجعه نمائید. با جستجو در اینترنت نیز تعداد بی‌شماری از موارد مشابه را خواهید یافت.

۸-۹-۲ 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 صفحهٔ 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 حقه قدیمی خود استفاده کرده باشد. به عنوان مثال ممکن است او در میانهٔ 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 صفحهٔ 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 شمارهٔ کارت اعتباری خود را جهت خرید کالا از «باب» ارسال نماید.

---

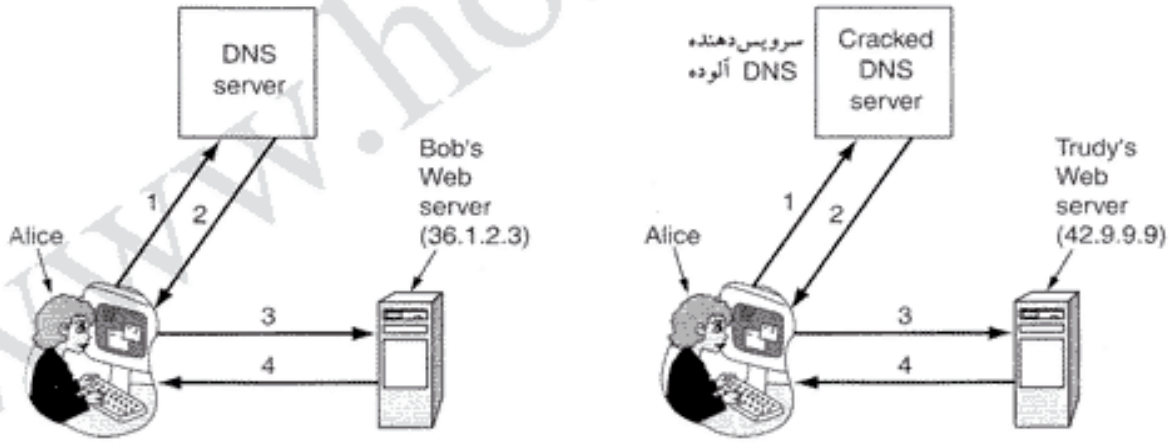
۱. به ماشینهایی که خود قربانی نفوذ بدخواهان شده‌اند و ناخودآگاه در حمله به یک سایت شرکت می‌کنند اصطلاحاً «ماشینهای زامبی» (Zambie) گفته می‌شود. -م.

۲. Distributed Denial of Service

یکی از اشکالات حملهٔ mitm[1] (از دیدگاه اخلالگران) 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 باید از خط تلفن آلیس یا باب انشعاب بگیرد چرا که انشعاب از ستون فقرات شبکه‌ای 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 ساده‌تری را در پیش می‌گیرد:

DNS Spoofing

به عنوان نمونه فرض کنید ترودی قادر به نفوذ در سیستم DNS شده است و توانسته در حافظهٔ 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، حملهٔ 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 می‌کند. (۲) پاسخ آن را دریافت می‌کند. (۳) از باب، صفحهٔ اصلی سایت (Home Page) او را تقاضا می‌کند. (۴) صفحه را دریافت می‌نماید. 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 می‌کند، آدرس ترودی را تحویل خواهد گرفت لذا کل ترافیک داده‌هایی که باید به سمت باب هدایت شود به سوی ترودی می‌رود. حال ترودی می‌تواند حملهٔ mitm را پایه‌ریزی کند بدون آن که نیازی به انشعاب گرفتن از خط تلفن باب یا آلیس داشته باشد. او فقط باید بتواند در سرویس دهندهٔ DNS نفوذ کرده و تنها یک رکورد را تغییر بدهد که بسیار ساده‌تر از انشعاب گرفتن از خطوط ارتباطی است!



۱. لطفا آدرس IP باب را به من بدهید.
۲. 36.1.2.3 (آدرس IP باب)
۳. تقاضای دریافت صفحه اصلی با
۴. دریافت صفحه وب باب
(الف)

۱. لطفا آدرس IP باب را به من بدهید.
۲. 42.9.9.9 (آدرس IP ترودی!)
۳. تقاضای دریافت صفحه وب اصلی
۴. ترودی صفحه وب جعل شده باب را ارسال می‌کند.
(ب)

شکل ۸-۴۶. (الف) وضعیت طبیعی. (ب) حمله‌ای بر اساس دستکاری در داده‌های DNS و تغییر در رکورد مربوط به باب.

---

۱. Man In the Middle

ترودی چگونه می‌تواند DNS را فریب داده و دستکاری کند؟ این کار نسبتاً ساده به نظر می‌رسد! بطور اجمالی، ترودی می‌تواند سرویس دهندهٔ DNS در ISP آلیس را وادار به ارسال تقاضائی جهت ترجمهٔ آدرس باب نماید. متأسفانه چون DNS از UDP بهره می‌گیرد، سرویس دهندهٔ DNS هیچ راهی برای بررسی آن که واقعاً چه کسی پاسخ این تقاضا را داده، ندارد. ترودی می‌تواند از این ویژگی سوءاستفاده کرده و پاسخ مورد نظر DNS را جعل و بدین نحو آدرس IP غلطی را در حافظهٔ نهان سرویس‌دهندهٔ DNS (DNS Cache) تزریق نماید. برای ساده‌تر شدن بحث فرض می‌کنیم که سرویس دهندهٔ DNS در ISP آلیس، هیچ درایه‌ای (Entry) در خصوص آدرس وب‌سایت باب ، *bob.com* ، ندارد. البته اگر چنین درایه‌ای وجود داشته باشد ترودی می‌تواند آنقدر صبر کند تا زمان اعتبار آن منقضی شود و بعداً تلاش خود را از سر بگیرد. (یا از روشهای دیگر استفاده کند.)

ترودی حملهٔ خود را با ارسال تقاضای ترجمهٔ آدرس bob.com به سمت سرویس دهندهٔ DNS در ISP آلیس، آغاز می‌نماید. از آنجایی که هیچ درایه‌ای برای این نام در حافظهٔ DNS وجود ندارد، سرویس‌دهندهٔ DNS از سرویس دهندهٔ سطح بالا یعنی سرویس دهندهٔ com. در خصوص این نام سؤال می‌کند. حال ترودی در نقش سرویس دهندهٔ DNS سطح بالا حمله کرده و یک پاسخ جعلی و دروغین با این مضمون برای DNS می‌فرستد: «**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 حافظهٔ سرویس دهندهٔ DNS درج می‌شود و پاسخ واقعی که بعداً می‌رسد به عنوان پاسخی بیجا و سرزده کنار گذاشته خواهد شد. فریب دادن سرویس دهندهٔ DNS به نحوی که آدرس IP جعلی و دروغین را در حافظهٔ نهان خود درج کند اصطلاحاً «DNS Spoofing» نامیده می‌شود.[1] به حافظهٔ نهان که در آن آدرسهای IP غلط درج شده است اصطلاحاً «**حافظهٔ نهان سمّی**» (Poisoned Cache) گفته می‌شود.

البته انجام این کارها با دشواریهایی روبروست: اول آن که ISP آلیس، بررسی می‌کند تا ببیند آیا بستهٔ پاسخ دریافتی، آدرس IP سرویس دهندهٔ معتبر و واقعی سطح بالا را در خود حمل می‌کند یا خیر؟ از آنجایی که ترودی می‌تواند هر چه که خواست در فیلد آدرس IP بستهٔ پاسخ قرار بدهد لذا قادر است این بررسی را ناکام بگذارد زیرا آدرسهای IP سرویس دهنده‌های سطح بالا شناخته شده هستند و او می‌تواند از این آدرسها استفاده کرده و پاسخ جعلی خود را با این آدرسها بفرستد.

دوم آنکه سرویس دهندهٔ DNS برای تشخیص اینکه کدام پاسخ متعلق به کدام تقاضا بوده است برای هر بستهٔ تقاضا یک شمارهٔ ترتیب در نظر می‌گیرد. برای آن که ISP آلیس فریب داده شود، ترودی مجبور است شمارهٔ ترتیب تقاضای جاری را بداند. ساده‌ترین راه برای فهمیدن شمارهٔ 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 دهندهٔ DNS برای آدرس ثبت شده خودش به نام *dns.trudy-the-intruder.com* ایجاد می‌کند. 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 می‌کند تا از سرویس‌دهندهٔ سطح بالای *com.* در خصوص نام حوزهٔ جدید ترودی سؤال کند.

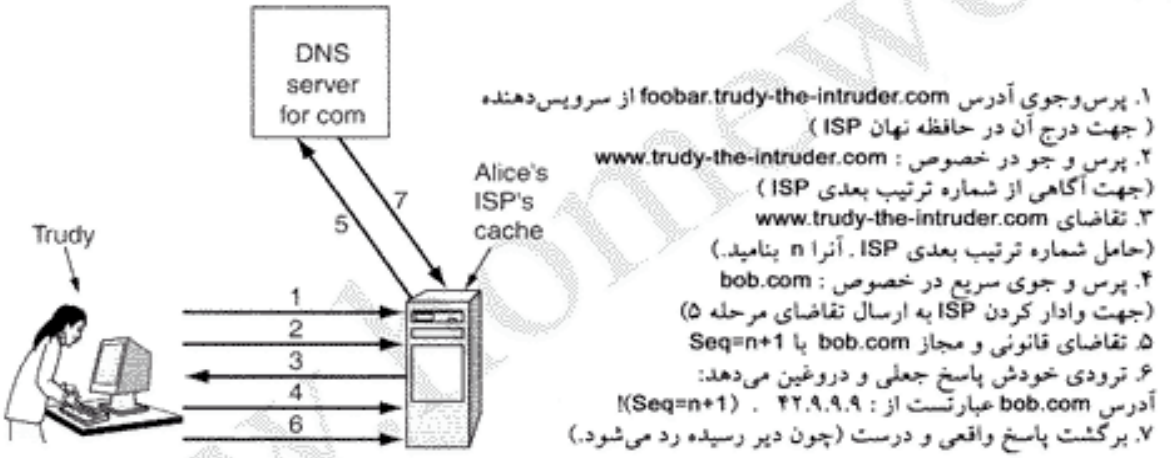
پس از آنکه آدرس *dns.trudy-the-intruder.com* در حافظهٔ نهان DNS آلیس درج شد حملهٔ اصلی می‌تواند شروع شود. ترودی تقاضایی به سمت DNS متعلق به ISP آلیس می‌فرستد و ترجمهٔ نام *www.trudy-the-intruder.com* را خواستار می‌شود. این ISP طبق معمول یک بستهٔ پرسش به سمت سرویس

---

۱. یعنی DNS متعلق به ISP آلیس را وادار به پرسش در خصوص نام bob.com می‌کند و بلافاصله خودش به این پرسش جواب غلط می‌دهد؛ چون DNS از UDP بهره می‌گیرد تشخیص هویت پاسخ‌دهنده میسر نیست. -م.

دهندهٔ ترودی ارسال می‌کند. این بستهٔ تقاضا شمارهٔ ترتیب مورد نیاز ترودی را با خود حمل می‌نماید. ترودی بلافاصله از ISP آلیس در مورد نام حوزهٔ باب (bob.com) سؤال می‌کند. سپس به سرعت به سؤال خودش پاسخ داده و بسته‌های جعلی را که در ظاهر به نظر می‌رسد از سرویس‌دهنده‌های سطح بالا رسیده برای آن ISP می‌فرستد. شمارهٔ ترتیب این بسته‌های جعلی یکی بیشتر از بسته‌ای است که چند لحظه قبل دریافت شده بود. البته برای اطمینان بیشتر ترودی می‌تواند یک بسته پاسخ با شماره ترتیب دو واحد بیشتر یا دهها بسته با شمارهٔ ترتیب متوالی (با شمارهٔ بیشتر) بفرستد. بالاخره یکی از آنها با شمارهٔ ترتیب تقاضای ارسالی منطبق خواهد شد و بقیه حذف خواهند گردید. وقتی پاسخ جعلی توسط DNS آلیس دریافت شد در حافظهٔ نهان ذخیره می‌شود. 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 می‌کند به او گفته می‌شود که از آدرس 42.9.9.9 استفاده کند که همان آدرس ترودی است. بدین ترتیب ترودی موفق شده از اتاق محل زندگی خود، حملهٔ mitm را برعلیه آلیس پی‌ریزی 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 می‌تواند در صورت عدم موفقیت آنها را نیز بیازماید.



شکل ۸-۴۷. چگونگی فریب دادن ISP آلیس توسط ترودی .

Secure DNS

این حملهٔ خاص را می‌توان بدین نحو خنثی کرد که DNS بجای استفاده از شمارهٔ ترتیب شمارشی برای بسته‌های پرسش و پاسخ از شماره‌های کاملاً تصادفی بهره بگیرد ولی به نظر می‌رسد که هر گاه یک رخنهٔ نفوذ مسدود شود، در کنار آن رخنهٔ دیگری در سیستم ایجاد می‌شود. 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 نمی‌آمد؛ مهمترین مورد آن بود که اینترنت در همه حال کار کند. در طول این سالها شرایط حاکم بر محیط اینترنت بشدت تغییر کرد؛ بهمین دلیل در سال ۱۹۹۴، IETF گروهی را مأمور ساخت تا DNS را بطور بنیادی امن نمایند. این پروژه به نام DNSsec شناخته می‌شود و نتیجهٔ آن در RFC 2535 ارائه شده است. متأسفانه از DNSsec در ترکیب شبکه‌ها به صورت گسترده و کامل استفاده نمی‌شود و به همین دلیل هنوز هم بسیاری از سرویس دهنده‌های DNS نسبت به حملهٔ DNS Spoofing آسیب‌پذیر هستند.

DNSsec ، از دیدگاه مفهومی فوق‌العاده ساده است. این سرویس دهنده بر اصول رمزنگاری یا 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 دهندهٔ DNS فرستاده می‌شود قبل از ارسال، با کلید خصوصی «ناحیهٔ مبدأ»[1] امضاء (رمز) شده و بدین ترتیب گیرندهٔ آنها می‌تواند هویت این اطلاعات را بررسی نماید.

DNSsec سه دسته خدمات اساسی زیر را ارائه می‌کند:

۱. اثبات آن که داده‌ها از کجا منشاء گرفته‌اند و به چه کسی متعلقند.
۲. توزیع کلید عمومی
۳. احراز هویت تقاضاها و تعاملاتی که با سرویس‌دهنده صورت می‌گیرد.

اصلی‌ترین سرویس همان مورد اول است که براساس آن بررسی می‌شود 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 حملهٔ نوع تکرار (Replay Attack) و حمله DNS Spoofing را خنثی کرد. دقت کنید که در DNSsec خدمات رمزنگاری داده‌ها ارائه نمی‌شود زیرا تمام اطلاعات DNS، عمومی و غیرمحرمانه است. انتظار می‌رود مراحل جایگزینی DNSsec با DNS معمولی چندین سال طول بکشد لذا سرویس دهندهٔ DNSsec باید این قابلیت اساسی را داشته باشد که با سرویس‌دهنده‌های معمولی و ناامن نیز کار کند و طبعاً 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 RRSets (Resource Record Sets) دسته‌بندی شده‌اند. تمام رکوردهایی که «نام دامنه» ، «نوع» و «کلاس» مشابه دارند در یک مجموعهٔ RRSet قرار می‌گیرند.[2] یک مجموعهٔ RRSet ممکن است شامل چندین رکورد A (رکوردهای آدرس IP) باشد چراکه مثلاً یک سرویس‌دهنده ممکن است دارای یک آدرس IP اولیه و یک آدرس ثانویه باشد. در هر مجموعهٔ RRSet چندین رکورد نوع جدید (که در ادامه بررسی خواهند شد) قرار می‌گیرد. همچنین برای هر مجموعهٔ RRSet یک رشتهٔ Hash (که صحت آن مجموعه را تائید می‌کند) به یکی از روشهای معمول مثل SHA-1 یا MD5 محاسبه می‌شود و نهایتاً با استفاده از کلید خصوصی صاحب آن ناحیه (Zone) به روشی مثل RSA رمزنگاری می‌گردد. کوچکترین واحد اطلاعات که برای مشتری ارسال می‌شود یک مجموعهٔ امضاء شدهٔ RRSet است. پس از دریافت RRSet امضاء شده، مشتری (Client) می‌تواند بررسی کند که آیا واقعاً با کلید خصوصی ناحیهٔ مبداء امضا شده است؟ اگر امضای آن منطبق بود داده‌ها پذیرفته می‌شوند. از آنجایی که هر مجموعهٔ RRSet امضای خودش را به همراه دارد می‌تواند در هر جایی ذخیره (cache) شود، حتی بر روی سرویس دهنده‌های غیرقابل 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 مشخصهٔ دیگر را نگهداری می‌کند. کلید عمومی به صورت آشکار (رمزنشده) ذخیره می‌شود. از گواهینامه‌های X.509 به دلیل حجم بزرگ استفاده نشده است. در رکورد نوع KEY فیلد مشخص کنندهٔ الگوریتم، برای امضاهای مبتنی بر MD5/RSA به عدد ۱ تنظیم می‌شود (انتخاب ارجح) و مقادیر دیگر، ترکیب الگوریتمهای مورد نظر را مشخص می‌کند. فیلد پروتکل می‌تواند استفاده از IPsec یا هر پروتکل امنیتی دیگر را (در صورت وجود) تعیین نماید.

دومین نوع جدید رکوردها، رکورد SIG است. این رکورد، رشتهٔ Hash امضاء شدهٔ یک ناحیه (Zone) را

---

۱. Originating Zone ۲. در خصوص این واژه‌ها به فصل هفتم مراجعه کنید. -م.

مشخص می‌نماید. (رشتهٔ Hash براساس الگوریتمی که در رکورد KEY مشخص شده، امضاء می‌گردد.) این امضاء بر روی تمام رکوردهای یک مجموعهٔ RRSet شامل رکوردهای KEY (به استثنای خودش)، اعمال می‌شود. رکورد SIG ، زمان شروع اعتبار و زمان انقضای امضاء و همچنین نام امضاءکننده و چند آیتم دیگر را نیز تعیین می‌کند.

طراحی DNSsec به گونه‌ای است که می‌توان 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 دادهٔ DNS به صورت دستی (مثلاً توسط CD-ROM) بر روی یک ماشین غیرمتصل به شبکه که کلیدهای خصوصی را نگه می‌دارد منتقل می‌گردد. در آنجا تمام مجموعه‌های RRSet امضاء شده و برای هر مجموعه، یک رکورد SIG (امضاء) تولید و نتیجه توسط CD-ROM به ماشین اصلی برگردانده می‌شود. بدین ترتیب می‌توان کلیدهای خصوصی را بر روی یک CD-ROM ذخیره کرد و CD-ROM به جز در مواقعی که باید مجموعه‌های 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 می‌شود. در این روال، امنیت الکترونیکی به امنیت فیزیکی کاهش می‌یابد و پیچیدگی چندانی ندارد.

این روش که در آن مجموعه‌های RRSet از قبل امضاء می‌شوند، سرعت فرآیند پاسخ به تقاضاها را بشدت افزایش می‌دهد چرا که لازم نیست در حین پاسخ و ارسال هیچگونه عمل رمزنگاری انجام شود.

وقتی ماشین مشتری، یک مجموعهٔ RRSet امضاء شده را دریافت می‌کند ابتدا باید کلید عمومی آن ناحیه را اعمال کرده و رشتهٔ Hash را بدست بیاورد؛ سپس خودش رشتهٔ Hash را مستقلاً محاسبه کرده و این دو مقدار را با هم مقایسه کند. اگر این دو مقدار یکسان بود، داده‌ها معتبر فرض می‌شود. این روال یک سؤال ایجاد می‌کند 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 سرویس‌دهندهٔ معتبر و مورد اعتماد و براساس یک اتصال امن نظیر IPsec اخذ شود.

با این وجود، در عمل فرض شده که ماشینهای مشتری به گونه‌ای پیکربندی شده‌اند که کلیدهای عمومی حوزه‌های سطح بالا را به صورت پیش فرض در اختیار دارند. حال اگر آلیس بخواهد به وب‌سایتِ باب مراجعه کند می‌تواند از DNS بخواهد که مجموعهٔ 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. مجموعهٔ RRSet توسط مسئول حوزهٔ سطح بالای com. امضاء می‌شود لذا آلیس می‌تواند اعتبار آن را بررسی کند زیرا کلید عمومی حوزهٔ com. را همه می‌دانند و نیازی به سؤال نیست. یک مثال از رکوردهایی که می‌تواند در یک مجموعهٔ RRSet وجود داشته باشد در شکل ۸-۴۸ نشان داده شده است.

حال با در اختیار داشتن نسخه‌ای معتبر از کلید عمومی باب، آلیس می‌تواند از سرویس دهندهٔ DNS متعلق به باب، در خصوص آدرس IP معادل با www.bob.com سؤال کند. اگر تروری به نحوی برنامه‌ریزی کند که تعدادی RRSet را به صورت جعلی تشکیل داده و در حافظهٔ نهان یک سرویس دهندهٔ 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 حوزهٔ bob.com. رکورد KEY حاوی کلید عمومی باب است. رکورد SIG حاوی رشتهٔ HASH امضاء شده توسط سرویس‌دهندهٔ 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 چراکه رکورد SIG (امضاء) با مجموعهٔ RRSet مطابقت ندارد و غلط است.

با این وجود DNSsec یک مکانیزم مبتنی بر رمزنگاری برای تطبیق بسته‌های پاسخ متناظر با تقاضاهای ارسالی ارائه کرده است تا از پی‌ریزی حملهٔ DNS Spoofing توسط ترودی (مبتنی بر شکل ۸-۴۷) جلوگیری شود. در این مکانیزم اختیاری (که antispoof نام دارد) به هر بستهٔ پاسخ یک رشتهٔ Hash اضافه می‌شود که این Hash از بستهٔ تقاضا استخراج شده و سپس با کلید خصوصی پاسخ دهنده، امضا (رمز) می‌شود. از آنجایی که ترودی کلید خصوصی سرویس دهندهٔ سطح بالای com. را نمی‌داند لذا قطعاً نخواهد توانست پاسخ ارسالی توسط این سرویس دهنده به ISP آلیس را جعل نماید. البته او می‌تواند بستهٔ جعلی خود را برای آلیس بفرستد ولیکن به دلیل امضای اشتباهی که دارد، حذف خواهد شد.

DNSsec از چندین رکورد دیگر نیز حمایت می‌کند. به عنوان مثال رکورد CERT می‌تواند برای ذخیره و نگهداری گواهینامه‌هایی مثل X.509 مورد استفاده قرار بگیرد. این رکورد بدان دلیل عرضه شده که برخی افراد علاقمندند DNS خود را در ساختار PKI[1] قرار بدهند؛ اگرچه هنوز در عمل چنین کاری انجام نشده است. در اینجا توضیحات خود در خصوص DNSsec را به پایان می‌رسانیم. برای تفصیل بیشتر از RFC 2535 کمک بگیرید.

«اسامی خودگواهی» (Self-Certifying Names)

استفاده از سرویس‌دهندهٔ امن DNS (DNSsec) تنها راه اطمینان کردن به اسامی نیست. یک راهکار کاملاً متفاوت در Secure File System (سیستم مطمئن فایل) بکار گرفته می‌شود. (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 گواهینامه‌های دیجیتالی یا بدون وجود هیچگونه PKI کار کند. در این بخش نشان خواهیم داد که چگونه می‌توان نظریات آنان را در وب اعمال کرد. بر این اساس، در توضیحات این بخش به جای استفاده از اصطلاحات و واژه‌های بکار رفته در مقالهٔ 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 نرم‌افزارهای مرتبط با وب ایجاد شود.

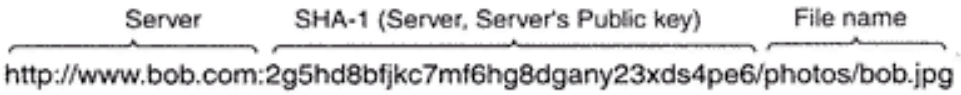
برای شروع فرض می‌کنیم که هر سرویس‌دهندهٔ وب دارای یک جفت کلید عمومی و خصوصی است. دلیل این فرض آنست که در هر URL یک رشتهٔ Hash برای بررسی صحت نام سرویس دهنده و صحت کلید عمومی آن، (به عنوان بخشی از URL) درج می‌شود. به عنوان مثال در شکل ۸-۴۹، URL یک فایل تصویر متعلق به باب را می‌بینیم. این آدرس طبق معمول با گزینهٔ http شروع شده که نام پروتکل[2] را تعیین می‌کند، سپس در ادامه، آدرس DNS سرویس دهندهٔ باب (یعنی *www.bob.com*) درج شده است. بعد از آن یک علامت کالون (:) و سپس ۳۲ کاراکتر به عنوان رشتهٔ Hash اضافه شده است. در انتهای این رشته، باز هم طبق روش معمول، نام و موقعیت فایل می‌آید. به غیر از رشتهٔ Hash، بقیهٔ URL استاندارد و معمولی است. با وجود رشتهٔ Hash، این URL اصطلاحاً «**خودگواهی**» می‌شود؛ (خودش صحت خود را تائید می‌کند).

سؤال بدیهی آن است که: رشته Hash برای چیست؟ پس از الحاق نام نمادین و کلید عمومی یک سرویس‌دهنده به یکدیگر، الگوریتم SHA-1 بر روی آن اعمال می‌شود تا رشتهٔ ۱۶۰ بیتی Hash بدست آید. در

---

۱. Public Key Infrastructure

۲. Protocol یا Schema



شکل ۸-۴۹. یک «URL خودگواهی» شامل رشتهٔ HASH از نام و کلید عمومی سرویس‌دهنده.

الگوی فوق رشته Hash به صورت دنباله‌ای 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 صرفنظر شده است، بنابراین جمعاً ۳۲ حرف و رقم باقی می‌ماند؛ (۲۴ حرف کوچک و ۸ رقم). با این سی و دو کاراکتر می‌توان پنج بیت را کدگذاری کرد. بدین ترتیب با یک رشتهٔ ۳۲ کاراکتری می‌توان جمعاً ۱۶۰ بیت رشته SHA-1 را کُد نمود؛ (۳۲ کاراکتر هر کاراکتر معادل ۵ بیت) البته در واقع، الزامی در استفاده از Hash نیست و می‌توان به جای آن خود کلید عمومی را قرار داد؛ حُسن استفاده از رشتهٔ Hash آنست که طول نام (که رشتهٔ Hash را در بر می‌گیرد) کاهش یابد.

ساده‌ترین روش (ولی در عین حال نامناسب‌ترین روش از لحاظ سهولت) برای مراجعه به فایل تصویر باب، آنست که آلیس رشتهٔ شکل ۸-۴۹ را در مرورگر خود درج نماید. مرورگر پیامی برای سرویس‌دهندهٔ وب متعلق به باب فرستاده و کلید عمومی او را مطالبه می‌کند. وقتی کلید عمومی باب دریافت شد مرورگر، نام سرویس دهنده و کلید عمومی را بهم چسبانده و الگوریتم Hash را بر روی آن اعمال می‌کند. اگر رشتهٔ Hash حاصل‌شده با رشتهٔ Hash سی و دو کاراکتری در URL مطابقت داشت، مرورگر مطمئن خواهد شد که کلید عمومی ارسالی واقعاً متعلق به باب است. حتی اگر ترودی بتواند در میانهٔ راه این تقاضا را دریافت کرده و یک پاسخ جعلی برگرداند نمی‌تواند هیچ کلید عمومی که رشتهٔ Hash آن با رشتهٔ درج شده در URL مطابقت داشته باشد پیدا کند لذا هر گونه دخالت او در جعل پاسخ، به سادگی کشف خواهد شد. پس از دریافت کلید عمومی باب، می‌توان آن را برای استفاده‌های آتی در حافظهٔ 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 می‌سازد 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 می‌فرستد. 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 می‌تواند مطمئن شود که با باب صحبت می‌کند. همچنین در اینجا آلیس و باب دارای یک کلید نشست AES مشترک هستند که می‌توانند برای تقاضاهای GET بعدی و پاسخهای ارسالی از آن استفاده کنند.

پس از آن که آلیس فایل تصویر متعلق به باب (یا هر صفحهٔ وب دیگر) را بدست آورد می‌تواند آدرس URL آن را در دفترچهٔ یادداشت مرورگر خود ذخیره نماید، بدین ترتیب او مجبور نخواهد بود در آینده، URL کامل را تایپ کند. بعلاوه URLهای جاسازی شده در درون صفحات وب نیز می‌توانند از نوع «**خودگواهی**» باشند، بنابراین فقط با یک کلیک می‌توان از آنها به صورت معمولی استفاده کرد؛ البته بشرطی که صفحهٔ وب برگشتی واقعی و دست نخورده باشد. یک روش دیگر برای آن که نیازی به درج URLهای «**خودگواهی**» نباشد آنست که این اسامی را از طریق یک سرویس‌دهندهٔ مورد اعتماد (که دارای گواهینامهٔ X.509 امضا شده توسط CA[1] باشد) و

---

۱. Certificate Authority

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 می‌توان این موتور جستجو را در خصوص یک URL مورد سؤال قرار داد و طبعاً نتیجهٔ کار ، یک صفحهٔ وب امضاء شده خواهد بود که درون آن تمام URLها از نوع «خودگواهی» هستند و می‌توان بر روی آنها کلیک کرد، بدون آن که نیازی به درج رشته‌های طولانی باشد.

حال ببینیم این روش چگونه می‌تواند در مقابل حملهٔ DNS Spoofing که توسط ترودی انجام می‌شود مقاومت نماید: اگر ترودی به نحوی برنامه‌ریزی کند تا حافظهٔ نهان ISP متعلق به آلیس آلوده شود، تقاضای آلیس برای دریافت یک صفحهٔ وب، به جای آن که توسط باب دریافت شود تحویل ترودی خواهد شد. حال طبق پروتکل فوق، مرورگر آلیس در اولین پیام ، از ترودی می‌خواهد که کلید عمومی خود را ارائه بدهد. اگر ترودی کلید عمومی خود را برگرداند، آلیس فوراً حمله را کشف خواهد کرد چراکه رشتهٔ Hash موجود در URL خودگواهی (که مبتنی بر کلید عمومی باب است) ، با رشتهٔ Hash محاسبه شده به روش SHA-1، منطبق نیست. اگر ترودی 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 می‌کند لذا ترودی پس از دریافت این پیامها هیچ راهی برای رمزگشایی و استخراج کلید AES و عدد تصادفی درون پیام نخواهد داشت. در اینجا اگرچه حملهٔ DNS Spoofing سودی برای ترودی (جهت بهره‌برداری از اطلاعات ارسالی) ندارد ولیکن می‌تواند برای حملهٔ «اخلال در سرویس‌دهی» (Denial of Service) مؤثر باشد، زیرا ترودی کاری کرده که آلیس نتواند با باب ارتباط برقرار کند.

۸-۹-۳ SSL: لایهٔ سوکتهای 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 می‌آیند. در بدو زمانی که وب در صحنهٔ اجتماع پدیدار گردید، از آن فقط برای توزیع صفحات وب ایستا[1] استفاده می‌شد. 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 **نت‌اسکیپ** (Netscape) عرضه کنندهٔ مرورگر مهم دنیا، با معرفی یک بستهٔ امنیتی به نام SSL (Secure Socket Layer) به این نیاز پاسخ داد. امروزه از این نرم‌افزار و پروتکل معرفی شدهٔ آن، بطور فزاینده‌ای استفاده می‌شود و حتی مرورگر IE مایکروسافت نیز از آن حمایت می‌کند، لذا بجاست که آنرا با اندکی شرح بیشتر بررسی کنیم.

«SSL یک اتصال مطمئن با مشخصات زیر، بین دو سوکت ایجاد می‌نماید:

۱. امکان مذاکره و توافق پارامترها بین سرویس دهنده و مشتری
۲. احراز هویت سرویس دهنده و مشتری بطور مستقل و مجزا
۳. مخابرهٔ سرّی و رمزنگاری شدهٔ داده‌ها
۴. مراقبت از صحت و سلامت داده‌ها

---

۱. صفحاتی که فقط ارائه دهندهٔ اطلاعات هستند و هیچگونه دریافت اطلاعات ندارند. -م

قبلاً هر یک از موضوعات فوق را به صورت جداگانه بررسی کرده‌ایم و نیازی به شرح مبسوط آنها نیست.

محل قرار گرفتن لایهٔ SSL در پشتهٔ پروتکلی TCP/IP، در شکل ۸-۵۰ نشان داده شده است. در حقیقت SSL یک لایهٔ جدید بین لایهٔ کاربرد و لایهٔ انتقال است که تقاضاهای مرورگر را گرفته و آنها را از طریق لایهٔ TCP برای سرویس دهنده ارسال می‌کند. پس از آن که یک اتصال مطمئن ایجاد شد، مهمترین وظیفهٔ SSL ، انجام عملیات فشرده‌سازی و رمزنگاری اطلاعات (و بالعکس) خواهد بود. وقتی از HTTP بر روی SSL استفاده می‌شود، اصطلاحاً آنرا HTTPS (Secure HTTP) می‌نامند، هرچند هیچ تفاوتی با پروتکل استاندارد HTTP ندارد؛ فقط در لایهٔ زیرین تغییراتی ایجاد شده است. البته برخی اوقات به جای پورت استاندارد ۸۰، HTTPS بر روی پورت جدید ۴۴۳ در دسترس قرار می‌گیرد. از آنجایی که SSL یک لایهٔ مستقل بر روی TCP است لذا استفاده از آن به مرورگرهای وب محدود نشده و دیگر برنامه‌ها نیز می‌توانند از امکانات آن استفاده کنند ولیکن عمومی‌ترین کاربرد آن در وب است.

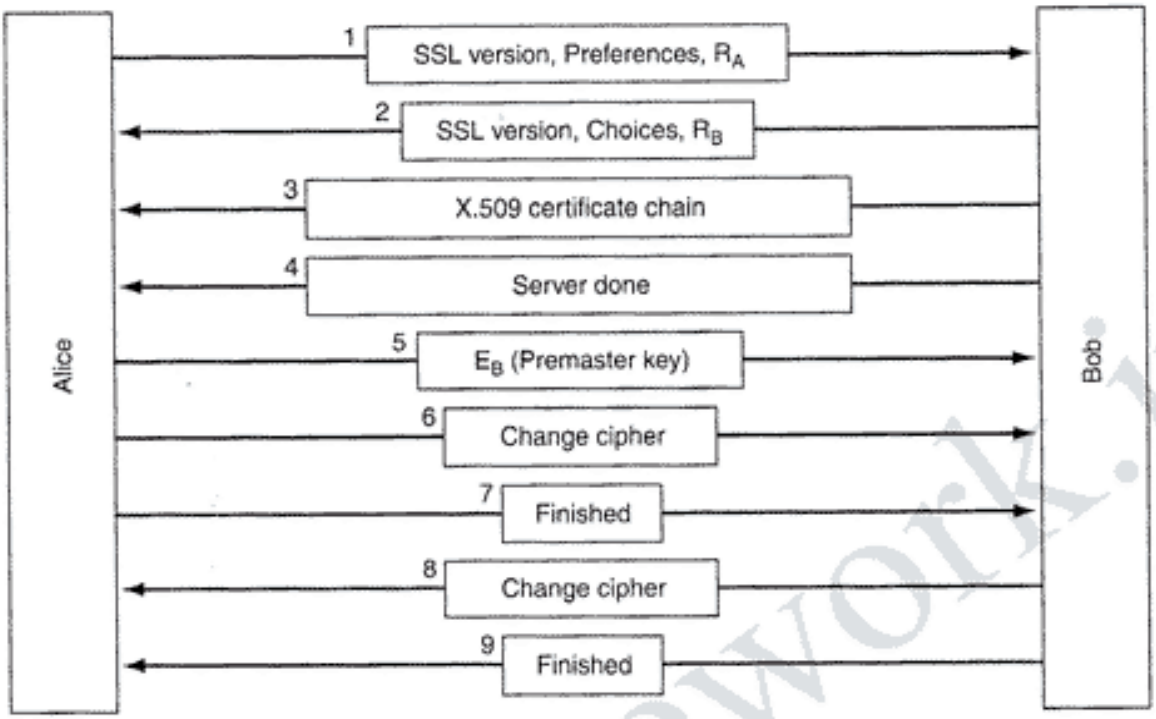
Application (HTTP)
Security (SSL)
Transport (TCP)
Network (IP)
Data link (PPP)
Physical (modem, ADSL, cable TV)

شکل ۸-۵۰. لایه‌ها (و پروتکلها) برای یک کاربر خانگی که از طریق SSL به مرور وب می‌پردازد.

پروتکل SSL از چندین نسخهٔ مختلف، گذر کرده و متکامل شده است. در زیر ما فقط نسخهٔ ۳ آنرا بررسی می‌کنیم چراکه بطور فزآینده‌ای از آن استفاده می‌شود و نسخه‌های قبلی عملاً منسوخ شده‌اند. SSL از چندین الگوریتم و گزینه‌های مختلف حمایت می‌کند. گزینه‌ها شامل استفاده یا عدم استفاده از فشرده‌سازی ، تعیین نوع الگوریتم رمزنگاری و مواردی در خصوص محدودیت صادرات محصولات مبتنی بر رمزنگاری است. آخرین مورد یعنی گزینه‌های 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 بقیهٔ موارد طول کلید به ۴۰ بیت محدود شده است که بسیاری از رمزنگاران این طول کلید را یک شوخی تلقی می‌کنند. شرکت نت‌اسکیپ 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 «**زیرپروتکل**» (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 می‌آید. زیرپروتکل ایجاد اتصال، در شکل ۸-۵۱ نشان داده شده است. وقتی آلیس تقاضای برقراری یک ارتباط امن با باب را می‌دهد، این زیرپروتکل کار خود را با ارسال پیام ۱ شروع می‌کند. این پیام نسخهٔ SSL مورد استفاده توسط آلیس و همچنین گزینه‌های انتخابی او نظیر تمایل به فشرده‌سازی و الگوریتم مورد نظر او جهت رمزنگاری را مشخص می‌کند. 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 گزینه‌های پیشنهادی که آلیس از آنها حمایت می‌کند، گزینه‌هایی را انتخاب کرده و این انتخابها را به همراه عدد تصادفی R_B و نسخهٔ SSL مورد استفاده، برای آلیس می‌فرستد. سپس



شکل ۸-۵۱. نسخهٔ ساده شده از زیرپروتکل برقراری اتصال در SSL.

در پیام سوم باب گواهینامهٔ 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 زنجیرهٔ گواهینامه‌ها را به نحوی برای آلیس بفرستد تا نهایتاً به یکی از این صد مرکز ختم شود.[1] 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 گواهینامهٔ 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 «**شاه‌کلید**» اولیه ۳۸۴بیتی و ارسال آن برای باب به او پاسخ می‌دهد در حالی که این شاه‌کلید 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، خاتمهٔ 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 که از او تقاضا کند با ارائه نام ورود و کلمهٔ عبور، Login نماید. پروتکل لازم برای عملیات Login، خارج از محدوده این کتاب است. پس از تکمیل عملیات Login، انتقال اطلاعات می‌تواند آغاز شود.

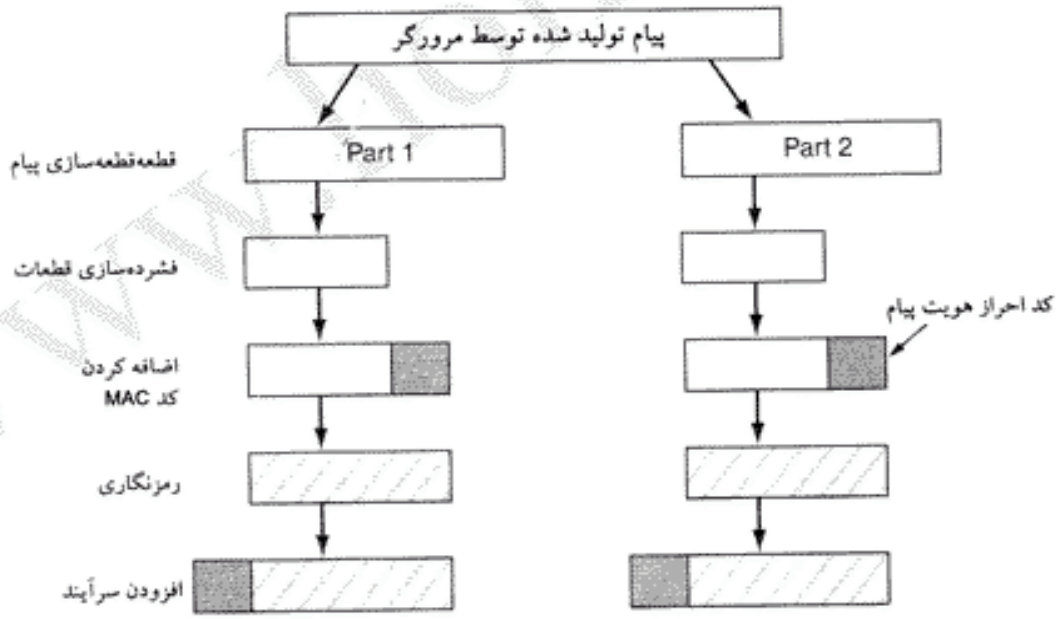
به گونه‌ای که قبلا اشاره شد، SSL از چندین الگوریتم رمزنگاری حمایت می‌کند. در قدرتمندترین روش،

---

۱. برای آشنایی با زنجیرهٔ 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 می‌شود. ترکیب این دو پروتکل نسبتاً کُند عمل می‌کند و اغلب برای عملیات بانکداری یا کاربردهایی که در آنها امنیت بالا حیاتی است مورد استفاده قرار می‌گیرد. برای عملیات معمولی تجارت الکترونیکی، از رمزنگاری RC4 با کلید ۱۲۸ بیتی و روش MD5 برای تائید صحت اطلاعات استفاده می‌شود. RC4 کلید ۱۲۸ بیتی را به عنوان نقطهٔ شروع گرفته و برای استفاده در الگوریتم، آنرا به یک عدد بزرگ بسط می‌دهد و سپس از آن برای تولید Keystream بهره می‌گیرد. این Keystream (به نحوی که در بخش ۸-۲-۳ تشریح شد) با داده‌های ارسالی ، XOR می‌شود تا متن رمز شده (به روش Stream Cipher) بدست بیاید؛ (شکل ۸-۵۲). نسخهٔ صادراتی SSL نیز از کلیدهای ۱۲۸ بیتی RC4 استفاده می‌کند ولیکن ۸۸ بیت از آن عمومی و آشکار است تا بتوان رمز آن را براحتی شکست. (زیرا صدور محصولاتِ 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 می‌شود. پیامهای ارسالی از مرورگر، ابتدا به قطعات ۱۶ کیلوبایتی شکسته می‌شود. اگر گزینهٔ فشرده‌سازی فعال شده باشد، هر قطعه به صورت مستقل فشرده می‌شود. سپس یک کلید سرّی که از ترکیب دو عدد تصادفی (nonce) و شاه‌کلید اولیه بدست می‌آید (موقتاً) به متن فشرده‌شده ضمیمه می‌شود و براساس الگوریتم مورد توافق (معمولاً MD5)، از آن یک رشته Hash استخراج می‌گردد. این رشتهٔ Hash به عنوان MAC[1] (کد تائید صحت پیام) به قطعهٔ فشرده‌شده ضمیمه می‌شود. سپس مجموعهٔ این دو با استفاده از الگوریتم متقارن مورد توافق (معمولاً با XOR کردن داده‌ها با Keystreamهای مبتنی بر الگوریتم RC4)، رمزنگاری می‌شود. نهایتاً به قطعهٔ حاصل یک سرآیند اضافه شده و نتیجه بر روی اتصال TCP ارسال می‌گردد.



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 می‌توان رمز متن را شکست، لذا امنیت SSL وقتی که در آن از RC4 استفاده شده بسیار متزلزل و شکننده است. مرورگرهایی به کاربر اجازه می‌دهند شخصاً الگوریتم مورد

---

۱. Message Authentication Code

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 می‌کند.

اشکال دیگر SSL آن است که طرفین ارتباط ممکن است گواهینامهٔ دیجیتالی نداشته باشند یا حتی در صورت دارا بودن گواهینامه، همیشه صحّت و تطابق کلیدهای مورد استفاده را بررسی نکنند.

در سال ۱۹۹۶ شرکت نت‌اسکیپ، SSL را برای استانداردسازی به IETF ارائه کرد. نتیجهٔ کار TLS (Transprot Layer Security) بود که در RFC 2246 تشریح شده است.

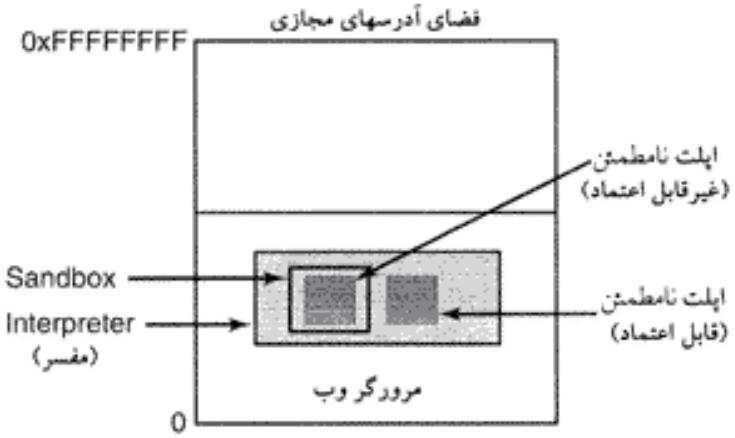
تغییراتی که در SSL ایجاد شد نسبتاً کم بود ولیکن همین تغییرات ناچیز کافی بود تا SSL نسخهٔ ۳ با TLS سازگار نباشد. به عنوان مثال روشی که براساس آن از شاه‌کلید اولیه و اعداد تصادفی (nonce)، کلید نشست استخراج می‌شود تغییر داده شد تا کلید قوی‌تر و محکم‌تر شود. (رمزشکنی آن سخت‌تر شود.) TLS گاهی به عنوان نسخهٔ 3.1 از SSL نیز نامیده می‌شود. اولین پیاده‌سازی 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.

۸-۹-٤ 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 فایلهای HTML ایستا بودند و درون آنها هیچگونه کد اجرایی وجود نداشت. امروزه اغلب صفحات وب، در برگیرندهٔ برنامه‌های کوچک اجرایی شامل اپلت‌های جاوا، کنترلهای Active X و اسکریپتهای 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 اپلتهای جاوا

اپلتهای جاوا، برنامه‌های کوچک جاوا هستند که برای اجرا بر روی یک «ماشین مجازی مبتنی بر پشته» (Stack Oriented) به نام JVM (Java Virtual Machine) ترجمه (کامپایل) می‌شوند. این برنامه‌ها می‌توانند درون یک صفحهٔ وب قرار گرفته و به همراه آن صفحه، بارگذاری و اجرا شوند. به گونه‌ای که در شکل ۸-۵۳ نشان داده شده است پس از بارگذاری صفحهٔ وب، اپلتها جهت اجرا تحویل مفسر JVM (تعبیه شده در درون مرورگر) می‌شوند.



شکل ۸-۵۳. اپلتها می‌توانند توسط مرورگر تفسیر و اجرا شوند.

یک مزیت اجرای کدهای مفسّری (Interpreted Code) در مقایسه با کدهای ترجمه شده (Compiled Code) آن است که هر دستورالعمل قبل از اجرا توسط مفسر بررسی می‌شود. این قابلیت، فرصت بررسی اعتبار آدرس هر دستورالعمل را به مفسر فرمان می‌دهد. به علاوه هرگونه فراخوانی سیستمی (System Call) توسط مفسر دریافت و قبل از اجرا تفسیر می‌شود. چگونگی برخورد با این فراخوانی‌های سیستمی، جوهرهٔ سیاستهای امنیتی را تشکیل می‌دهد. به عنوان مثال هر گاه یک اپلت قابل اعتماد باشد (از دیسک محلی خودِ کامپیوتر دریافت و اجرا شده باشد) هر گونه فراخوانی سیستمی بدون چون و چرا اجرا می‌شود ولیکن اگر اپلت مورد اعتماد نباشد (از طریق اینترنت دریافت شده باشد) باید در یک شرایط کاملاً بسته و حفاظت شده (که Sandbox نامیده می‌شود) اجرا گردد تا بتوان بر رفتار آن، اعمال محدودیت کرد و از هر گونه تلاش برای دسترسی به منابع سیستم جلوگیری شود.

هر گاه یک اپلت تلاش کند تا از منابع سیستمی ماشین بهره بگیرد، فراخوانی سیستمیِ آن اپلت، به منظور تائید بلافاصله تحویل یک «**ناظر امنیت**» (Security Monitor) می‌شود. «ناظر امنیت» براساس سیاستهای امنیتی که به صورت محلی تدوین شده تصمیم می‌گیرد که آیا باید به آن فراخوانی اجازهٔ اجرا بدهد یا آنرا ردّ کند. بدین ترتیب فقط امکان دسترسی به برخی از منابع سیستمی وجود دارد نه همهٔ آنها. متأسفانه حقیقت آنست که این مدل امنیتی بد عمل می‌کند و اشکالات آن همیشه به ناگاه پدیدار می‌شود.

Active X

کنترلهای اکتیواکس، برنامه‌های اجرایی و دودویی برای ماشینهای پنتیوم هستند که می‌توانند درون صفحات وب جاسازی شوند. وقتی مرورگر به یکی از آنها بر می‌خورد، ابتدا بازرسی و آزمایشهایی انجام می‌شود تا ببیند آیا قابل اجرا است؛ در صورتی که بتواند این آزمون را با موفقیت بگذراند اجرا می‌شود. این کنترلها به هیچ‌وجه تفسیر (Interpret) و نظارت نمی‌شوند لذا به اندازهٔ برنامهٔ معمولی کاربران قدرت اجرایی دارند و می‌توانند آسیبهای بالقوه خطرناکی را به سیستم تحمیل کنند.

روشی که شرکت مایکروسافت برای تصمیم‌گیری در خصوص اجرا یا عدم اجرای کنترلهای 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 رشتهٔ Hash از کد اجرایی است که توسط کلید خصوصی طراح آن ، رمزنگاری و امضاء می‌شود. وقتی یک کنترل اکتیواکس ظاهر می‌شود، مرورگر ابتدا امضای آن را بررسی می‌کند تا مطمئن شود که در خلال انتقال دستکاری نشده است. اگر امضای آن صحیح بود مرورگر جدول داخلی خود را بررسی می‌کند تا ببیند آیا پدیدآورندهٔ آن برنامه قابل اعتماد است؟ یا آن که زنجیرهٔ گواهینامه‌های آن، به یک تولیدکننده مورد اعتماد ختم می‌شود یا خیر؟ اگر پدیدآورندهٔ اکتیواکس مورد اعتماد باشد، برنامه اجرا می‌شود، در غیر این صورت اجرا نخواهد شد. سیستمی که مایکروسافت برای بررسی کنترلهای اکتیواکس پیاده کرده به نام Authenticode مشهور است.

مقایسهٔ 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 نمی‌شود. در عوض یک «مفسر زمان اجرا»، این اطمینان را فراهم می‌آورد که اپلت کارهایی را که صاحب ماشین اجازه نداده ، انجام نمی‌دهد. برعکس، در اکتیواکس با اطمینان به امضای کدهای اجرایی، بر عملکرد و رفتار کُد در حال اجرا نظارت نمی‌شود. اگر آن برنامه از منبع قابل اعتماد دریافت شده و در حین گذر از شبکه دستکاری نشده باشد فوراً اجرا می‌شود. هیچ تلاشی نیز برای بررسی اینکه آیا کدهای برنامه مخرب هستند یا نه انجام نمی‌گیرد. اگر برنامه‌نویس اصلی اکتیواکس (که دارای گواهینامهٔ معتبر است و مورداعتماد تلقی می‌شود)، در نظر داشته باشد که کُد او کل دیسک سخت ماشین را نابود کرده و سپس

Flash ROM کامپیوتر را پاک کند تا دیگر بوت نشود، کُد او بی‌چون و چرا اجرا شده و کل کامپیوتر را نابود خواهد کرد. (مگر آن که گزینهٔ Active X در تنظیمات مرورگر غیرفعال شده باشد.)

بسیاری از افراد در خصوص اعتماد به شرکتهای ناشناختهٔ تولید نرم‌افزار نگران و بدبین هستند. یک برنامه‌نویس در سیاتل آمریکا برای آن که این اشکال را نشان بدهد یک شرکت نرم‌افزاری بوجود آورد و یک گواهینامهٔ اعتماد برای خود تهیه کرد؛ انجام این کار ساده است. سپس یک کنترل اکتیواکس نوشت که کامپیوتر را خاموش (shutdown) می‌کرد. 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 می‌شد ماشین را از نو راه‌اندازی نمود. واکنش رسمی به کار او آن بود که گواهینامهٔاش را برای این کنترل اکتیواکس خاص باطل کردند و بدین نحو به این داستان خجالت‌آور پایان داده شد ولیکن هنوز زمینهٔ بروز چنین مشکلاتی برای سوء استفادهٔ برنامه‌نویسان شرور وجود دارد. (Garfinkel with Spafford,2002) از آنجایی که هیچگاه نمی‌توان بر هزاران شرکت تولید نرم‌افزار که ممکن است کدهای متحرک بنویسند نظارت کرد، لذا روش امضای کدهای اجرایی، فاجعه‌ای است که هر لحظه در شرف وقوع خواهد بود.

اسکریپتهای جاوا

اسکریپتهای جاوا ذاتاً دارای هیچ مدل امنیتی رسمی و شناخته شده‌ای نیستند و پیاده‌سازیهای ناامن و نفوذپذیر آنها پیشینه‌ای طولانی دارد. هر تولیدکنندهٔ نرم‌افزار از یک دیدگاه خاص به امنیت می‌پردازد. به بعنوان مثال نسخهٔ دوم از مرورگر نت‌اسکیپ از مدلی شبیه به مدل جاوا استفاده می‌کند (یعنی مبتنی بر نظارت و کنترل فراخوانیها)، در حالی که در نسخهٔ چهارم به سمت مدل امضای کدها حرکت کرده است.

مشکل اساسی آنست که اجازهٔ 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 خانهٔ خود دعوت کرده و سپس تلاش کنید او را به دقت تحت نظر بگیرید مبادا از آشپزخانه به اتاق نشیمن فرار کند. اگر اتفاق غیرمنتظره‌ای بیفتد یا آن که برای لحظاتی غفلت کنید، حوادث ناخوشایندی می‌تواند رخ بدهد. گرایش به کدهای اجرایی و متحرک مثل اسکریپتها از آنجایی ناشی می‌شود که این کدها امکان نمایش گرافیکهای متحرک و تعاملات سریعتر و بهینه‌تری را فراهم می‌کنند و بسیاری از طراحان وب‌سایت وب فکر می‌کنند 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 نمی‌کند. تفاوت بین یک ویروس و کدهای متحرک معمولی آن است که ویروسها به گونه‌ای نوشته می‌شوند که خود را تکثیر کنند. وقتی یک ویروس از طریق صفحات وب ، ضمیمه‌های نامهٔ الکترونیکی یا روشهای دیگر به یک ماشین می‌رسد، کار خود را با آلوده کردن برنامه‌های اجرایی روی دیسک آغاز می‌کند، وقتی یکی از این برنامه‌ها اجرا شود، کنترل اجرا به ویروس منتقل شده و آن ویروس مجدداً تلاش می‌کند به روشهایی مثل ارسال کپی خودش از طریق پست الکترونیکی (به آدرسهایی که قربانی در کامپیوترش یادداشت کرده) ، خود را تکثیر کند. برخی از ویروسها بوت‌سکتور دیسک سخت را آلوده می‌کنند به نحوی که وقتی ماشین بوت می‌شود، ویروس نیز بلافاصله اجرا می‌گردد. ویروسها به یک مشکل عمده برای شبکهٔ اینترنت تبدیل شده‌اند و میلیاردها دلار خسارت به بار می‌آورند. شاید نسل جدید سیستمهای عامل که براساس تکنولوژی **ریزهستهٔ امن** (Secure Microkernel) و تفکیک دقیق و محکم کاربران ، پروسه‌ها و منابع بنا نهاده شده است به کاهش این مشکل کمک کند.

۸ـ۱۰ زمینه‌ها و پی‌آمدهای اجتماعی

اینترنت و تکنولوژی امنیت آن، موضوعی است که در آن موارد اجتماعی، سیاستهای عمومی و تکنولوژی با یکدیگر تلاقی می‌کنند 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 می‌توانیم 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 "Censonrship" (سانسور) یا "Copyright" را در یک موتور جستجو ، تایپ کنید و نتایج جستجو را دنبال نمایید. همچنین می‌توانید به وب‌سایت همین کتاب مراجعه کنید تا در این خصوص، چندین لینک به سایتهای مفید بدست بیاورید.

۸ـ۱۰ـ۱ حریم خصوصی افراد (Privacy)

آیا افراد از حقّ داشتن حریم خصوصی بهره‌مند هستند؟ سؤال بسیار خوبی است! در چهارمین اصلاحیهٔ قانون اساسی در ایالات متحدهٔ آمریکا آمده که حکومت بدون دلیل موجه و قانونی حق جستجوی منازل مردم، نوشته‌ها 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 قاعدهٔ عمومی تبدیل شده است.

در طول چند دههٔ گذشته چیزهایی تغییر کرده‌اند که هم کار حکومت در جاسوسی از شهروندان را ساده‌تر کرده و هم این امکان را برای شهروندان فراهم آورده که جلوی این جاسوسی را بگیرند! در قرن هجدهم، برای آن که دولت بتواند نوشته‌های یک نفر را تفتیش کند، مجبور بود که یک پلیس اسب سوار را به مزرعهٔ 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 ارائهٔ مجوز لازم به سادگی امکان ایجاد انشعاب و استراق سمع را در اختیار می‌گذارند. 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 بستهٔ نرم‌افزاری PGP را دریافت و نصب کند و از کلید رمز مطمئن و مستحکم استفاده نماید می‌تواند اطمینان داشته باشد که هیچکس در این جهان قادر به خواندن نامه‌های 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 روزنامه‌نگاران و رقبای سیاسی بسیار سخت‌تر می‌شود. بدین دلیل برخی از دولتها بکارگیری یا صدور محصولات رمزنگاری را محدود یا حتی ممنوع کرده‌اند. 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 سخت‌افزار رمزنگار به نام Cliper Chip را به عنوان استانداردی برای تمام مخابرات شبکه بسازد. البته عنوان شده بود که حریم خصوصی شهروندان محترم شمرده می‌شود. همچنین اشاره شده بود که این تراشهٔ عرضه شده توسط دولت قادر است تمام ترافیک داده‌ها را با استفاده از روشی به نام **Key escrow** رمزگشایی کند؛ این ساختار اجازه می‌داد 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 داده‌ها خواهد پرداخت. بدیهی است که خشم عموم برانگیخته شد، طرفداران حفظ حریم خصوصی شهروندان آن را شرم‌آور و ننگین دانستند در حالی که مجریان قانون آن را ستایش می‌کردند. 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* در دسترس عموم قرار دارد.

نامه‌پراکن‌های ناشناس (Anonymous Remailer)

PGP، SSL 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 مخابرهٔ 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 بپردازیم: اول آن که مخالفان سیاسی که در سیطرهٔ 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 خلافکاری 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.

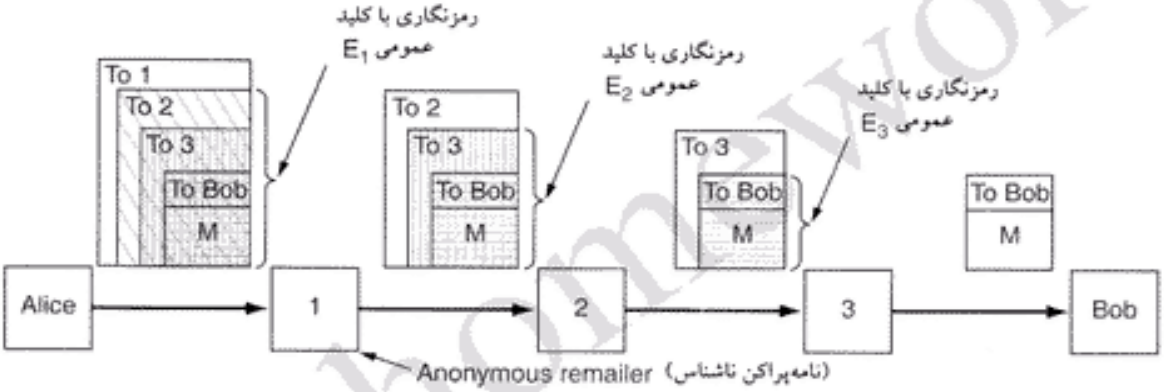
بیائید به یک مثال خاص بپردازیم. در سال ۱۹۹۰ برخی از مخالفین یک اقلیت مذهبی، دیدگاههای خود را از طریق یک سیستم نامه‌پراکن ناشناس (Anonymous Remailer) برای یک گروه خبری در یوزنت ارسال کردند. این سرویس‌دهنده به کاربران اجازه می‌داد تا برای خود اسم مستعار انتخاب کرده و نامه‌های الکترونیکی خود را برای آن 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 سرویس‌دهنده مکاتبه‌ای داشتند مشخص شد. (این اولین باری نبود که یک گروه مذهبی از انتشار اسرار درونی خود بر می‌آشفت: ویلیام تیندال در سال ۱۵۳۹ به دلیل ترجمهٔ انجیل به زبان انگلیسی در آتش خشم سوخت.)

بخش قابل ملاحظه‌ای از جامعهٔ 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 سرویس‌دهندهٔ 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 به‌گونه‌ای 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 گزینهٔ From:) و سپس آن را با کلید عمومی سرویس‌دهندهٔ نامه‌پراکن رمز می‌کند و نهایتاً پیام را برای آن نامه‌پراکن

می‌فرستد. در این سرویس دهنده، سرآیند خارجی RFC 822 حذف و پیام رمزگشایی می‌شود و سپس برای دیگران ارسال می‌گردد. سرویس دهندهٔ نامه‌پراکن، هیچگونه حساب کاربری (Account) تعریف نکرده و هیچ «فایل سوابق» (Log File) ذخیره و نگهداری نمی‌کند و بدین ترتیب حتی اگر بعداً این سرویس‌دهنده توقیف شود هیچ ردّپایی از پیامهایی که از آن گذر کرده‌اند بجا نمی‌ماند.

بسیاری از کاربران که علاقه‌مند و مصّر به ناشناس ماندن هستند به نحوی که در شکل ۸-۵۴ نشان داده شده، پیامهای خود را به صورت زنجیره‌ای از چندین نامه‌پراکن عبور می‌دهند. در این شکل، آلیس تمایل دارد که یک پیام را به صورت کاملاً ناشناس برای باب بفرستد. به همین دلیل از سه سرویس دهندهٔ نامه‌پراکن استفاده می‌کند. او پیام M را تنظیم کرده و سرآیند لازم را که آدرس پست الکترونیکی باب جزو آنست به پیام اضافه می‌کند. سپس کل آن را با کلید عمومی سومین سرویس‌دهندهٔ نامه‌پراکن یعنی E3 رمز می‌کند. (در شکل این پیام به صورت هاشورهای افقی نشان داده شده است.) سپس به پیام سرآیند جدیدی که در برگیرندهٔ آدرس پست الکترونیکی سرویس دهندهٔ سوم است اضافه می‌کند. 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 نامه‌پراکن ناشناس.

سپس پیام جدید را با کلید عمومی سرویس‌دهندهٔ نامه‌پراکن دوم رمز می‌کند. (در شکل این پیام با هاشورهای عمودی نشان داده شده است.) در ادامه سرآیندی حاوی آدرس پست الکترونیکی سرویس‌دهندهٔ دوم به متن حاصل می‌افزاید. این پیام در شکل ۸-۵۴بین سرویس دهندهٔ ۱ و ۲ نشان داده شده است. نهایتاً او کل پیام را با کلید عمومی سرویس دهندهٔ نامه‌پراکن ۱ رمز کرده و آدرس پست الکترونیکی او را به آن اضافه می‌کند. این پیام همانی است که بین آلیس و سرویس دهندهٔ اول در شکل مبادله می‌شود و پیام واقعی که انتقال داده خواهد شد همین پیام است.

وقتی پیام به اولین سرویس‌دهندهٔ نامه پراکن بر می‌خورد، سرآیند بیرونی آن جدا شده و متن درون آن رمزگشایی و برای سرویس دهندهٔ دوم ارسال می‌شود. همین مراحل در دو سرویس‌دهندهٔ بعدی اتفاق می‌افتد.

اگرچه تعقیب ردّ پیامها برای هر کس بی‌نهایت دشوار است ولیکن برای اطمینان بیشتر، بسیاری از این نامه‌پراکنها اقدامات احتیاطی اضافه‌تری را انجام می‌دهند. 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 سرویس‌دهندهٔ نامه‌پراکن می‌رود و بدین ترتیب حمله‌ای که براساس تحلیل ترافیک صورت می‌گیرد خنثی خواهد شد. برای شرح بیشتر در خصوص سیستم پست الکترونیکی ناشناس و مدرن به مراجع (Mazieres and Kaashoek, 1998) مراجعه نمائید.

ناشناس ماندن (Anonimity) 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 سرویس‌دهندهٔ Anonymizer به عنوان **پراکسی** استفاده نماید. از آن پس تمام تقاضاهای HTTP به سوی این سرویس‌دهنده ارسال می‌شود و این سرویس دهنده به نیابت از کاربر صفحات را تحویل گرفته و به او بر می‌گرداند. تمام وب‌سایتهای دنیا، این سرویس‌دهنده را (یعنی Anonymizer را) مبداء اصلی تقاضا می‌بینند نه کاربر اصلی را. تا زمانی که سرویس‌دهندهٔ Anonymizer از نگهداری «فایل سوابق» (Log) اجتناب ورزد هیچکس نمی‌تواند تعیین کند چه کسی تقاضای کدام صفحه را داده است.

۸-۱۰-۲ 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 میلیون صفحهٔ 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 ضدّملی و ضدّاجتماعی

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 همهٔ 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 - تنین 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 گسترهٔ وب جهانی است و نمی‌توان هیچ قانون واحدی را بر آن حاکم کرد. از طرفی امروزه سرویسهایی عرضه شده که اگرچه در مقابل پدیدهٔ سانسور قرار می‌گیرند ولیکن می‌تواند درهم شکنندهٔ 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 سرویس‌دهنده‌ها 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 سرویس‌دهنده‌های 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، عرضهٔ 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.

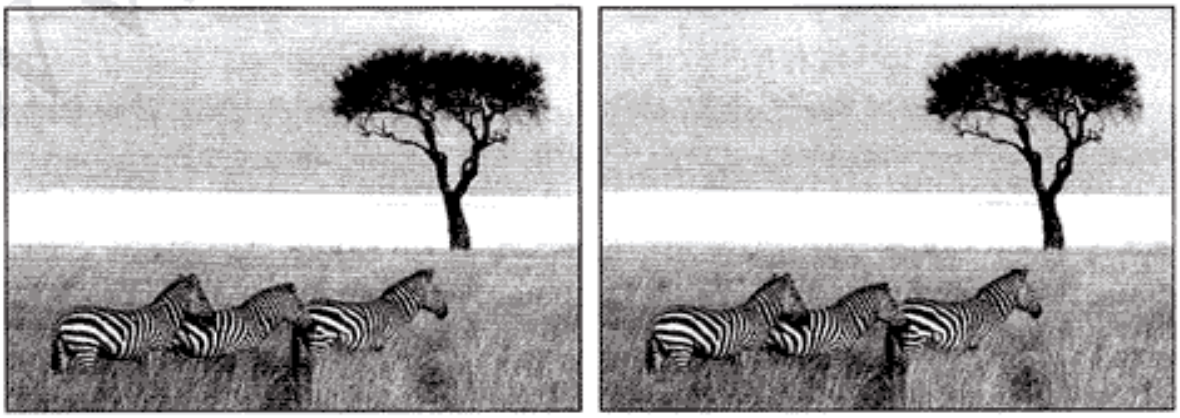
بسیاری از کشورها به صورت فزاینده‌ای در تلاش هستند تا صدور محصولات غیرعینی و غیرفیزیکی را که بیشتر در برگیرندهٔ سایتهای وب، نرم‌افزار، مقالات علمی، نامه‌های الکترونیکی و مشاوره‌های تلفنی هستند، قانونمند و محدود کنند. حتی انگلستان که برای قرنها مدعی آزادی بیان بوده، بطور جدی در حال بررسی قوانین محدودکننده و سرسختانه‌ای 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 سیاستهایی مناقشه برانگیز هستند.

استیگانوگرافی[1] (پوشیده‌نویسی)

در کشورهایی که حجم سانسور بسیار زیاد است مخالفان اغلب سعی می‌کنند برای فرار از آن از تکنولوژی استفاده نمایند. رمزنگاری اجازه می‌دهد که پیامها (حتی اگر قانونی هم نباشند) ارسال شوند ولیکن اگر حکومت، آلیس را شخصی بد و مخرّب بینگارد، افرادی مثل باب که با او مراوده دارند را نیز همفکر او تلقی می‌کند. سرویس‌دهندهٔ «نامه‌پراکن ناشناس» (Anonymous Remailer) می‌تواند به آنها کمک کند ولی اگر چنین سرویس‌دهنده‌هایی 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 راه‌گشا است.

افرادی که می‌خواهند به صورت سرّی با یکدیگر ارتباط داشته باشند اغلب سعی می‌کنند این ارتباط را به هر نحوی پنهان نگاه دارند. علم مخفی کردن پیامها اصطلاحاً «**استیگانوگرافی**» نامیده می‌شود که برگرفته از دو کلمهٔ یونانی به معنای «پوشیده‌نویسی» است. در حقیقت در ابتدا یونانیان باستان خود از این روش استفاده کرده‌اند. «هرودوت» مورخ یونانی از یک ژنرال ارتش یاد کرده که سر پیام‌رسان 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! تکنیکهای 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 اقاقیا دیده می‌شوند. تصویر ۸-۵۵-ب نیز مشابه با عکس قبلی به نظر می‌رسد 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»، «شاه‌لیر»، «مکبث»، «تاجر ونیزی» و «ژولیوس سزار». مجموع این نمایشنامه‌ها جمعاً ۷۰۰کیلوبایت 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!!

۱. Steganography

این مخفی‌سازی چگونه انجام می‌شود؟ ابعاد تصویر رنگی و اصلی ۱۰۲۴x۷۶۸ نقطه (پیکسل) است. هر پیکسل شامل سه عدد هشت بیتی است که هر یک شدت رنگهای قرمز، سبز و آبی را در هر نقطهٔ تصویر مشخص می‌کنند. از ترکیب این سه رنگ (با شدتهای متفاوت) رنگ هر نقطه بدست می‌آید. در روش کدگذاری مخفی از کم‌ارزش‌ترین بیت هر یک از سه مقدار رنگهای RGB به عنوان «کانالهای مخفی» (Covert Channel) استفاده می‌شود. بنابراین هر پیکسل فضایی معادل ۳ بیت برای جاسازی اطلاعات سرّی در اختیار می‌گذارد؛ (یک بیت در مقدار قرمز، یکی در آبی و یکی در سبز). در تصویری به ابعاد فوق مجموعاً ۱۰۲۴x۷۶۸x۳ بیت (معادل ۲۹۴۹۱۲ بایت) از اطلاعات سرّی را می‌توان جاسازی کرد.

متن کامل این پنج نمایشنامه به همراه یک توضیح کوتاه جمعاً حدود ۷۳۴۸۹۱ بایت است. این متن ابتدا با استفاده از الگوریتم استاندارد فشرده‌سازی، به ۲۷۴ کیلوبایت فشرده شده و سپس نتیجه ، با استفاده از الگوریتم IDEA رمزنگاری و در کم ارزش‌ترین بیت از مقادیر رنگها ذخیره شده است. به گونه‌ای که مشاهده می‌شود (یا به عبارت بهتر به گونه‌ای که مشاهده نمی‌شود!!) وجود این اطلاعات کاملاً غیرقابل رویت است. حتی در تصویر بزرگ شده و تمام رنگی این عکس باز هم چیزی قابل رویت نیست. چشم نمی‌تواند براحتی تفاوت بین رنگهای ۲۱ بیتی یا ۲۴ بیتی را تشخیص بدهد.

البته مشاهدهٔ دو تصویر فوق به صورت سیاه و سفید و با دقت پایین در این کتاب، نمی‌تواند در مورد قدرت این تکنیک قضاوت کند. برای آن که عملکرد استیگانوگرافی را بهتر احساس کنید، مؤلف یک نمونهٔ نمایشی از استیگانوگرافی، شامل تصویر تمام رنگی با دقت بالا از شکل ۸-۵۵-ب را به همراه پنج نمایشنامهٔ جاسازی شده، عرضه کرده است. این نمونه نمایشی شامل یک ابزار برای جاسازی و استخراج متن در تصویر است که می‌توانید آن را در وب‌سایت این کتاب بدست بیاورید.

برای استفاده از استیگانوگرافی به منظور محاورهٔ مخفیانه، معاندین می‌توانند یک وب‌سایت ایجاد کنند که سرشار از تصاویر مُجاز 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. فایلهای صوتی نیز بخوبی کارآیی دارند. فایلهای ویدیویی دارای پهنای باند بسیار عظیمی برای پنهان‌سازی اطلاعات هستند. حتی ترکیب چیده شدن عناصر (Layout) و ترتیب برچسبها در فایل HTML (HTML Tags) نیز می‌تواند حامل اطلاعات باشد!

استیگانوگرافی تنها برای حمل اطلاعات مخفی نیست و کاربردهای دیگری هم دارد. یکی از کاربردهای عمومی آن می‌تواند این باشد که صاحب حقوقی یک عکس پیامهائی سرّی در درون یک تصویر جاسازی کند. هر گاه چنین تصویری دزدیده شده و در یک وب‌سایت قرار داده شود، مالک قانونی آن می‌تواند این پیام محرمانه و سرّی را برای اثبات مالکیت آن، به دادگاه عرضه کند. به این تکنیک اصطلاحاً نشانه‌گذاری (Watermarking) گفته می‌شود 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 پدیدهٔ سانسور موضوعاتی هستند که سیاستهای عمومی و تکنولوژی را رو در رو قرار داده‌اند. مورد سوّم **مالکیت معنوی آثار** است. مالکیت معنوی (Copyright) بدین معناست که امتیاز

بهره‌برداری از عواید یک اثر برای مدتی برای آفرینندگان آن (شامل نویسندگان، هنرمندان، آهنگسازان، نوازندگان، عکاسان، سینماگران) محفوظ بماند که این دورهٔ زمانی می‌تواند ۵۰ تا ۷۵ سال پس از عمر صاحب اثر باشد. پس از آن که طول دورهٔ این امتیاز که به صورت قانونی ثبت می‌شود به سر آمد، آن اثر همگانی تلقی شده و همه می‌توانند به دلخواه از آن بهره ببرند یا آن را بفروشند. به عنوان مثال پروژهٔ گوتنبرگ یا آثار شکسپیر امروزه همگانی تلقی می‌شوند و به رایگان در وب در دسترس هستند. در سال ۱۹۹۸ کنگرهٔ 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 دهندهٔ خدمات مبادلهٔ رایگان موزیک) به ۵۰ میلیون نفر رسید. از آنجایی که Napster هیچگونه عمل کپی فایلهای موزیک را انجام نمی‌داد دادگاه به طرح این اتهام پرداخت که در اختیار گذاشتن یک بانک اطلاعاتی از اینکه چه کسی چه موزیکهایی را در اختیار دارد معاونت در جرم تلقی می‌شود و Napster بدین ترتیب به وقوع جرم کمک کرده است. اگرچه هیچکس مخالف قانون مالکیت معنوی آثار نیست (هرچند برخی ادعا می‌کنند طول دوره و جزئیات سختگیرانهٔ آن زیاد است) ولیکن دور جدیدی از به اشتراک‌گذاری موزیک، یک مناقشه عظیم را دامن زده است.

به عنوان مثال شبکه‌ای نقطه‌به‌نقطه را در نظر بگیرید که در آن افراد فایلهای قانونی خود را (شامل موزیکهای همگانی، ویدیوهای شخصی، اعلانهای مذهبی که در آن اسراری وجود ندارد) و شاید چند فایل که حق مالکیت آن انحصاری است را به اشتراک گذاشته‌اند. فرض کنید افراد به صورت دائم از طریق یک خط ADSL یا کابل، به شبکه وصل شده‌اند. هر ماشین یک فهرست از آنچه بر روی دیسک سخت او موجود است و همچنین فهرستی از بقیهٔ مشترکین را در اختیار دارد. هر کسی که به دنبال یک مورد خاص می‌گردد یکی از اعضای این فهرست را انتخاب کرده و بررسی می‌کند که آیا او این آیتم را در اختیار دارد یا خیر؟ اگر نداشت او می‌تواند در فهرستِ یکایک اعضا بررسی کند. پس از پیدا شدن آیتم مورد نظر، متقاضی می‌تواند آن را کپی نماید.

اگر آنچه که به اشتراک گذاشته می‌شود تحت حمایت قانون مالکیت معنوی باشد، آنهایی که چنین آیتمهایی را برداشت می‌کنند مرتکب قانون‌شکنی شده‌اند. (هر چند مشخص نیست وقتی که انتقال چنین آیتمهایی به صورت بین‌المللی انجام می‌شود، چه قانونی قابل‌اعمال و استناد خواهد بود.) با کسی که چنین زمینه‌ای 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 کرده‌اید، 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 گسترده‌ای پیرامون مالکیت معنوی آثار در گرفته است و نزاع شدیدی بین هالیوود و شرکتهای کامپیوتری بوجود آمده است. هالیوود می‌خواهد تا قوانین حفاظت از آثار، سختگیرانه و سنگین‌تر شود و شرکتهای کامپیوتری نیز نمی‌خواهند مأمور حفاظت از منافع هالیوود باشند.

در اکتبر ۱۹۹۸، کنگرهٔ آمریکا قانونی به نام DMCA (Copyright Act Digital Millennium) را از تصویب گذراند که براساس آن هر گونه فریبکاری و حیله‌پردازی برای نقض مالکیت آثار در پوششهای به ظاهر قانونی یا انتشار چنین راههایی برای فرار از قانون، جرم محسوب می‌شود. چنین قانونی در اروپای متحد نیز از تصویب گذشت. در حالی که بسیاری از افراد آگاه نیستند که در شرق دور نسخه‌برداری از آثار مجاز شمرده می‌شود! و خوشبینانه بدان می‌اندیشند که قانون DMCA می‌تواند بین حقوق مالکین و پدیدآورندگان 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 می‌کردند سیستمی غیرقابل‌نفوذ 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 می‌کند.) یک گروه از محققین امنیت سیستم از چندین دانشگاه به سرپرستی پروفسور ادوارد فِلتن از دانشگاه پرینستون بدین چالش علمی وارد شده و این سیستم را درهم شکستند. سپس مقاله‌ای در خصوص یافته‌های خود نوشته و آن را برای کنفرانس امنیت USENIX ارسال کردند. قبل از آن که موعد ارائه این مقاله فرا برسد، فِلتن نامه‌ای 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 می‌دیدند: 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 مقالهٔ فوق‌الذکر منتشر گردید.

مورد دیگری که به بحث ما مرتبط است بسط **«نظریهٔ استفادهٔ جوانمردانه از آثار»** (Fair Use Doctorine) است که توسط قانونگذاران قوهٔ قضاییه در بسیاری از کشورها وضع شده است. این نظریه بیان می‌کند که خریداران یک اثر که حقوق معنوی آن متعلق به دیگران است، اجازه دارند طبق ضوابط خاصی از آن کار نسخه‌برداری 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 نسخهٔ اصلی یک اثر چیزی از دست ندهند، از آن چندین نسخهٔ کپی تهیه کنند. برای بررسی آن که آیا از یک اثر، استفادهٔ جوانمردانه می‌شود یا نه، باید معیارهای زیر ارزیابی شود: (۱) آیا استفاده از آن اهداف تجاری دارد. (۲) چند درصد از کل آن نسخه‌برداری می‌شود. (۳) نسخه‌برداری 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 می‌داند، استفادهٔ 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 اینتل و مایکروسافت به نام TCPA[1] ارائه شد. نظریه آن بود که یک تراشهٔ CPU و سیستم عامل آن، بدقت بر رفتار و عملکرد کاربران نظارت داشته باشد (مثل اجرای یک موزیک یا نرم‌افزار که به صورت غیرقانونی کپی شده است) و جلوی اعمال غیرمجاز کاربر را بگیرد. این سیستم حتی به مالکان نرم‌افزارها یا دیگر کالاهای الکترونیکی اجازه می‌دهد 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)، اخلالگران و دیگر موارد امنیتی که مردم با آن دست به گریبانند، تمام تلاش خود را مصروف اجرای قانون حمایت از پدیدآورندگان آثار کرده‌اند!!

کوتاه سخن آن که قانونگذاران و وکلا در سالهای آتی نیز درگیر مسئله توازن بین حقوق مصرف‌کننده و حقوق

---

۱. Trusted Computing Platform Alliance

پدیدآورندگان آثار خواهند بود. دنیای الکترونیکی امروز بی‌شباهت به صحنهٔ جنگ نیست: گروهی به جان گروه دیگر می‌افتند، یکدیگر را لگدمال می‌کنند، کارشان به دادگاه می‌کشد و خوشبختانه سرانجام کار، اکثراً مصالحه می‌کنند 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 کِرکهُف بنا شده‌اند یعنی الگوریتم بکار رفته در آنها آشکار و عمومی و فقط کلید رمز سرّی است. بسیاری از الگوریتمهای رمزنگاری از تبدیلهای پیچیدهٔ ریاضی شامل عملیات جانشینی و جایگشتی برای تبدیل متن به رمز بهره گرفته‌اند. ولیکن هر گاه رمزنگاری کوانتومی بتواند در محیط عمل وارد شود، روش One-Time Pad یک سیستم رمزنگاری واقعاً غیرقابل شکست عرضه خواهد کرد.

الگوریتمهای رمزنگاری را می‌توان 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 می‌کنند 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 می‌توان در حالتهای Electronic Code Book ، Cipher Block ، Chaining Mode ، Stream Cipher Mode ، Counter Mode و نظایر آن بهره گرفت.

الگوریتمهای رمزنگاری با کلید عمومی این ویژگی را دارند که کلیدهای رمزنگاری و رمزگشایی متفاوت از یکدیگر هستند و نمی‌توان با داشتن کلید رمزنگاری، کلید رمزگشایی را استخراج کرد. این ویژگی اجازه می‌دهد تا بتوان کلیدهای عمومی را منتشر کرد. اصلی‌ترین الگوریتم کلید عمومی، RSA است که قدرت خود را از آنجایی بدست آورده که تجزیهٔ 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 (رشتهٔ خلاصه و درهم شدهٔ پیام) استخراج شده و سپس این رشته به جای متن اصلی رمزنگاری می‌شود.

مدیریت کلیدهای عمومی با استفاده ازگواهینامه‌های دیجیتالی که در آن هویت شخص و کلید عمومی او درج شده، قابل انجام است.گواهینامه‌های دیجیتالی توسط مراکز معتمد مردم یا اشخاص حقیقی تائید می‌شوند. «ریشه» (یعنی Root یا عالیترین مرکز گواهی امضاء) باید پیشاپیش مشخص باشد ولیکن اغلب مرورگرها گواهینامهٔ دیجیتالی بسیاری از مراکز اصلی گواهی امضاء را به صورت درونی در اختیار دارند.

ابزارهای رمزنگاری می‌توانند برای امن کردن ترافیک جاری بر روی شبکه به کار گرفته شوند. IPsec در سطح لایهٔ شبکه عمل می‌کند و بسته‌هایی را که از یک ماشین به ماشین دیگر روانه می‌شوند، رمزنگاری می‌کند. دیوارهای آتش می‌توانند بر ورود و خروج اطلاعات یک سازمان نظارت کنند که این کار اغلب براساس نوع پروتکل (شمارهٔ پروتکل لایهٔ انتقال) و شمارهٔ پورت انجام می‌گیرد. شبکه‌های VPN می‌توانند شبکه‌های قدیمی که مبتنی بر خطوط اجاره‌ای و انحصاری بودند را با سطح امنیت مورد نظر شبیه‌سازی کنند. نهایتاً شبکه‌های بی‌سیم نیاز به امنیت خوب و کافی دارند در حالی که شبکهٔ 802.11 WEP چنین امنیتی را فراهم نکرده است؛ اگرچه 802.11i خواهد توانست این ویژگی را بهبود بخشد.

وقتی دو طرف با یکدیگر نشستی را ترتیب می‌دهند، در ابتدا مجبور هستند یکدیگر را تائید هویت کنند و یک

کلید نشست ایجاد نمایند. در این خصوص، پروتکلهای احراز هویت متفاوتی وجود دارد، شامل: روش مبتنی بر یک شخص ثالث و معتمد، روش دیفی ـ هلمن، روش Kerberos و روش رمزنگاری کلید عمومی.

امنیت نامه‌های الکترونیکی را می‌توان 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 می‌شوند. کلید رمز IDEA، بکمک کلید عمومی گیرندهٔ پیام، رمز می‌شود و به همراه پیام ارسال می‌گردد. بعلاوه برای بررسی صحت پیامها، از درون آن یک رشتهٔ Hash استخراج شده و امضاء می‌شود.

امنیت وب نیز یکی از عناوین مهم در امنیت شبکه است که با مقولهٔ «نامگذاری امن» آغاز می‌شود. DNSsec روشی است که نامها خودشان صحت خود را گواهی کنند و بدین ترتیب از حملهٔ DNSspoofing پیشگیری می‌شود. بسیاری از وب‌سایتهای تجارت الکترونیکی برای برقراری نشستهای امن و احراز هویت شده بین مشتری و سرویس دهنده، از SSL بهره می‌گیرند. روشهای 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 سیاستهای 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 تک‌حرفی (Monoalphabetic) زیر را بشکنید. متن اصلی فقط از حروف الفباء تشکیل شده و یک قطعهٔ 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 sidzkf zhx zyy ur om zid rzk
hu foiia mztx kfd ezindhkdi kfda kfzhgdx ftb boef rui kfzk

۲. رمز قطعه کد زیر را که به صورت جایگشت ستونی رمزنگاری شده، بشکنید. این متن از یک کتاب معمولی رشتهٔ کامپیوتر انتخاب شده و طبعاً کلمهٔ computer محتمل‌ترین کلمهٔ درون آن است. متن اصلی فقط از حروف الفباء انگلیسی تشکیل شده و فاصلهٔ خالی درون آن نیست. فقط برای سادگی در خواندن، متن رمز شده به صورت دسته‌های پنج حرفی نشان داده شده است (فواصل خالی را در حین محاسبات خود حذف نمائید.)

aauan cvlre rurnn dltme aeepb ytust iceat npmcy iicgo gorch srsoc
nntii imiha oofpa gsivt tpsit lbolr otoex

۳. یک رشته بیت One-Time Pad ۷۷بیتی برای متن رمز شدهٔ شکل ۸-۴ پیدا کنید تا متن "Donald Duck" را تولید نماید.

٤. رمزنگاری کوانتومی به یک تفنگ فوتونی نیاز دارد تا بتواند در صورت نیاز، یک تک فوتون حامل بیت ۱ تولید نماید. محاسبه نمائید 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 سانتی‌متر در نانو ثانیه (20Cm/nsec) در نظر بگیرید.

۵. وقتی از رمزنگاری کوانتومی استفاده می‌شود اگر تروریستی بتواند فوتونهای نوری را دریافت و بازتولید نماید برخی از بیتها را اشتباه دریافت خواهد کرد و همچنین خطاهایی را در رشته بت One-Time Pad متعلق به باب بوجود خواهد آورد. بطور متوسط چه نسبتی از رشته بیت باب خراب خواهد شد؟

۶. یکی از اصول پایهٔ رمزنگاری بیان می‌کند که تمام پیامها باید افزونگی داشته باشد. ولیکن از طرفی با این موضوع آشنا شدیم که وجود افزونگی به اخلالگر 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 رشتهٔ 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-Boxها و S-boxها 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-Boxها و سپس تمام S-boxها 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، فاصلهٔ خالی، کاما، نقطهٔ 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 رمزشکن 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 ۱۰۲۴ پردازنده داشته باشند، برای شکستن رمز AES باید کارآیی 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 ماشینی 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 رمزشده 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 (d1,n1) دقیقاً معادل با کلید

عمومی RSA از یک کاربر دیگر به نام فرانسیس با کلید (e2,n2) است. به عبارت دیگر d1=e2 و n1=n2 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 می‌اندازد؟

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 می‌شود و او چگونه می‌تواند از بروز این اشکال پیشگیری نماید.

۱۸. در شکل ۸-۲۰ می‌بینیم که چگونه آلیس پیامی امضاء شده را برای باب می‌فرستد. اگر ترودی متن P را کلاً عوض کند باب متوجه خواهد شد. به نظر شما اگر ترودی هم P و هم امضای آن را عوض کند چه اتفاقی می‌افتد؟

۱۹. امضاهای دیجیتالی دارای یک ضعف بالقوه هستند که از تنبلی کاربران ناشی می‌شود. در معاملات تجارت الکترونیکی پیش‌نویس یک قرارداد تهیه شده و از کاربر خواسته می‌شود تا رشتهٔ SHA-1 Hash متناظر با آن قرارداد را با کلید خصوصی خود امضاء نماید. اگر کاربر بررسی نکند که آیا حقیقتاً رشتهٔ Hash که آنرا امضاء می‌کند متناظر با قرارداد مورد نظر اوست ممکن است سهواً Hash یک قرارداد دیگر را امضاء کند. فرض کنید مافیا سعی می‌کند 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 می‌خواهند که شمارهٔ کارت اعتباری خود را وارد نمایند. سپس قراردادی را برای مشتری می‌فرستند تا آنرا امضاء کند، با این نیّت که اکثر کاربران بدون بررسی آنکه آیا Hash مربوط به قرارداد متناظر و متعلق به پیام هست یا خیر، آن را امضاء می‌کنند. نشان بدهید که چگونه مافیا می‌تواند 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 دانش‌آموز دارد. احتمال آن که حداقل دو دانش‌آموز 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 می‌کند ضمن دیکته کردن پیامهای آتی خود از طریق یک ماشین خاص، منشی خود را نیز عوض کند. سپس مرلین به گونه‌ای برنامه‌ریزی می‌کند 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 می‌تواند از «حملهٔ روز تولد» (Birthday Attack) برای جعل پیامها بهره بگیرد و اگر می‌تواند چگونه؟ (راهنمایی: این کار ممکن است).

۲۲. به تلاش ناموفق آلیس در دریافت کلید عمومی باب در شکل ۸-۲۳ دقت نمائید. فرض کنید باب و آلیس از قبل بر روی یک کلید سرّی و مشترک توافق کرده‌اند 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 می‌خواهد با باب به کمک کلید عمومی، تبادل اطلاعات داشته باشد. او یک اتصال با کسی که فکر می‌کند باب است برقرار می‌نماید و از طرف مقابل می‌خواهد که کلید عمومی خود را بفرستد؛ طرف مقابل نیز کلید عمومی خود را به صورت آشکار و به همراه گواهینامهٔ X.509 خود که توسط مرکز عالی CA امضاء

شده ، ارسال می‌کند. آلیس نیز یک کلید عمومی که توسط مرکز CA امضاء شده در اختیار دارد. چند مراحلی 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 سرویس‌دهندهٔ عمومی است.)

۲۴. فرض کنید که یک سیستم از PKI مبتنی بر ساختار «سلسله‌مراتب مراکز CA» بهره می‌گیرد. آلیس می‌خواهد که با باب ارتباط برقرار کند و به همین دلیل پس از ایجاد ارتباط، گواهینامهٔ دیجیتالی باب را که توسط یک CA با نام X امضاء شده دریافت می‌کند. فرض کنید او هیچ چیزی در مورد مرکز X نمی‌داند. آلیس باید چه مراحلی برای بررسی هویت کسی که با او صحبت می‌کند، پشت سر بگذارد.

۲۵. آیا در یک ماشین که در پشت جعبهٔ NAT (NAT BOX) قرار گرفته می‌توان از IPsec (با استفاده از AH و در **حالت انتقال**) بهره گرفت؟

۲۶. یک مزیت استفاده از HMAC به جای RSA ، برای امضای رشته‌های 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 می‌کنید این بررسیها احتمال موفقیت دارد؟

۲۸. قالب بستهٔ WEP در شکل ۸-۳۱ نشان داده شده است. فرض کنید که کُد کشف خطای ۳۲بیتی در این بسته، با XOR کردن کلمات ۳۲بیتی 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 می‌خواهد 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 گونه‌ای تغییر دهید تا در مقابل حملهٔ بازتاب (Reflection Attack) نفوذناپذیر شود. تشریح کنید که این تغییر شما به چه نحو کار می‌کند.

۳۱. برای ایجاد یک کلید سرّی بین آلیس و باب از روش «مبادلهٔ کلید دیفی ـ هلمن» استفاده شده است. آلیس آیتمهای (۱۹۱ و ۳ و ۷۱۹) را برای باب می‌فرستد. باب با (۵۴۳) پاسخ می‌دهد. عدد سرّی و محرمانهٔ آلیس یعنی x معادل ۱۶ است. کلید سرّی و مشترک چیست؟

۳۲. اگر آلیس و باب هیچگاه یکدیگر را ملاقات نکرده و هیچ گواهینامهٔ دیجیتالی یا کلید سرّی در اختیار نداشته باشند، می‌توانند توسط الگوریتم دیفی ـ هلمن یک کلید مشترک و سرّی ایجاد نمایند. تشریح کنید که در این الگوریتم چرا مقابله با حملهٔ نوع man-in-the-middle بسیار دشوار است؟

۳۳. در پروتکل شکل ۸-۳۹ چرا مشخصهٔ A به صورت آشکار (رمز نشده) بهمراه کلید رمزشدهٔ نشست ارسال می‌شود؟

۳۴. در پروتکل شکل ۸-۳۹ اشاره کردیم که اگر هر قطعهٔ متن پیام با ۳۲بیت صفر شروع شود یک تهدید امنیتی به وجود خواهد آمد. فرض کنید که هر پیام با یک شمارهٔ تصادفی که به ازای هر کاربر تولید می‌شود و یک کلید سرّی که فقط برای کاربر و KDC مشخص است، شروع شود. آیا این ساختار مشکل حمله از طریق «متون شناخته شده» (Known Plaintext) را حل می‌کند؟ چرا؟

۳۵. در پروتکل «نیدهام ـ شرودر»، آلیس دو رشتهٔ چالش R_A و R_{A2} تولید می‌کند. این کار بی‌مورد 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 ۷، 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 بدهید.

۳۸. ترمینالهای فروش که از کارتهای اعتباری مغناطیسی و کدهای PIN استفاده می‌کنند یک اشکال جدی دارند: یک فروشندهٔ 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 آتی خود سوءاستفاده کند. نسل آیندهٔ ترمینالهای فروش از کارتهایی استفاده می‌کنند که بر روی آنها یک CPU کامل، صفحهٔ کلید و یک نمایشگر کوچک قرار گرفته است. پروتکلی برای این سیستم ابداع کنید تا هیچ فروشندهٔ 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 شکّ دیگران برانگیخته شود را کاهش خواهد داد. به نظر شما این مرحلهٔ 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 شمارنده‌ای که IDهای متوالی تولید می‌کند، از ID تصادفی استفاده شود. جنبه‌های امنیتی این روش را تشریح کنید.

۴۳. در پروتکل انتقال دادهٔ SSL، دو عدد (nonce) و یک شاه‌کلید اولیه بکار رفته است. این اعداد (nonce) چه مقادیری و چه کاربردی دارند؟

۴۴. تصویر شکل ۸-۵۵-ب در برگیرندهٔ متن آسکی 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 همهٔ پیامهایش با یک اسم مستعار مشابه می‌رسیدند. با فرض آن که سرویس‌دهندهٔ نامه‌پراکن (Remailer) به درستی کار کرده باشد، تروری نمی‌توانسته خودش را به جای آلیس جا بزند. پس از آن که این سرویس‌دهندهٔ نامه‌پراکن (Remailer) از کار افتاد، آلیس به یک سرویس‌دهندهٔ Cypherpunk Remailer تغییر سرویس‌دهنده داد و با این سرویس‌دهنده ارسالهای خود به گروه خبری را از سر گرفت. راهی ابداع کنید تا تروری 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 صفحه‌ای تهیه کنید.

۴۷. در اینترنت بدنبال چند مورد قضایی در خصوص «مالکیت حقوق معنوی آثار» (Copyright) جستجو کرده و خلاصهٔ یافته‌های خود را در یک صفحه، گزارش نمائید.

۴۸. برنامه‌ای 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 نمائید. برنامهٔ شما باید همانند یک فیلتر عمل کرده و متن اصلی را از طریق ورودی استاندارد دریافت نماید و نتیجهٔ 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 نقطهٔ شروع مولد عدد تصادفی (Seed) استفاده می‌شود.

۴۹. پروسیجری بنویسید که رشتهٔ SHA-1 Hash متناظر با یک بلوک داده را محاسبه نماید. این پروسیجر باید دو پارامتر داشته باشد: یک اشاره‌گر به بافر ورودی و یک اشاره‌گر 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.