«به نام خالق آرامش»

نام کتاب: اصول امنیت شبکه های کامپیوتری (بخش اول)

نام نویسنده: ویلیام استالینگز

نام مترجم: مسعود موحد

تعداد صفحات: ۲۴۰ صفحه

تاریخ انتشار: ــــــــــــــــ





اصول امنیت شبکه‌های کامپیوتری

کاربردها و استانداردها

ویرایش سوّم (۲۰۰۷ میلادی)

اثر: William Stallings

ترجمهٔ: مسعود موحّد
مدرّس
مرکز آموزش شرکت مخابرات ایران
و
دانشکدهٔ علمی_کاربردی پست و مخابرات

در بارهٔ نویسنده :

William Stallings دارای لیسانس مهندسی برق از دانشگاه Notre Dame و درجهٔ Ph.D. از دانشگاه M.I.T. کشور امریکاست. خلاقیت‌های او اثرات منحصربفردی در فهم شبکه‌های کامپیوتری و معماری کامپیوتر داشته است. وی مؤلف ۱۷ کتاب در زمینه‌های علوم کامپیوتر است که هرکدام بارها تجدید چاپ شده‌اند. در ۲۰ سال گذشته وی مدیر چندین مؤسسهٔ فنی و تحقیقاتی بوده و در حال حاضر یک مشاور مستقل است که مشتریان او کارخانجات سازندهٔ کامپیوتر و تجهیزات شبکه، تولیدکنندگان نرم‌افزار و مؤسسات تحقیقاتی برجسته دولت امریکا می‌باشند. نامبرده هفت بار برندهٔ جایزهٔ بهترین کتاب‌های دانشگاهی علوم مهندسی کامپیوتر بوده است.

Stallings سایت مرجع دانشجویان رشتهٔ کامپیوتر با آدرس WilliamStallings.com/StudentSupport/html را خلق کرده است که اسناد و لینک‌های بسیار متنوعی در زمینه‌های مختلف علوم کامپیوتری برای افراد آماتور و حرفه‌ای فراهم می‌آورد. او عضو هیئت تحریریهٔ مجلهٔ *Cryptologia* است که آخرین دستاوردهای علم رمزشناسی را عرضه می‌نماید. دربین کتاب‌های اوکتاب *Data and Computer Communications*، که اخیراً ویرایش هشتم آن بچاپ رسیده است، بعنوان کتاب استاندارد این رشته در سراسر دنیا شناخته شده است.

در بارهٔ مترجم :

مسعود موّحد دارای فوق لیسانس مهندسی برق و الکترونیک از دانشکدهٔ فنی دانشگاه تهران و درجهٔ D.Eng. در رشتهٔ مخابرات از دانشگاه George Washington کشور امریکاست. وی از سال ۱۳۵۸ به خدمت شرکت مخابرات ایران درآمده و تا کنون ضمن تصدی مشاغل مختلف مدیریت آموزشی، به تدریس در زمینه‌های الکترونیک و مخابرات مشغول بوده است. نامبرده همچنین بمدت چهارسال رئیس مرکز آموزش و مجری طرح‌های آموزشی شرکت مخابرات ایران بوده و بطور همزمان سرپرستی دانشکدهٔ پست و مخابرات وابسته به وزارت ارتباطات و فناوری اطلاعات را بعهده داشته است.

نامبرده در حال حاضر در زمینهٔ مخابره و انتقال داده‌ها، شبکه‌های کامپیوتری و علی‌الخصوص مسائل امنیتی شبکه‌ها و اینترنت به تدریس و تحقیق اشتغال دارد. وی کتاب حاضر را بارها در دوره‌های ضمن خدمت کارشناسان شرکت‌های وابسته به وزارت ارتباطات تدریس نموده است.

بنام آفرینندهٔ اندیشه‌های ژرف

اگرچه ترجمهٔ این کتاب را از مدت‌ها قبل آماده کرده بودم ولی انتشار آن را به دو دلیل به تعویق انداختم. اول اینکه چندین بار آن را تدریس کنم تا ضمن کسب تجربه، با استنباط نویسندهٔ محترم در مورد مطالب کتاب بیشترآشنا شوم و دوم اینکه ویرایش سوم این کتاب که برای سال ۲۰۰۷ میلادی آماده شده است به بازار آید تا آخرین تغییراتی که مؤلف محترم در متن کتاب اعمال نموده‌اند را نیز در متن ترجمه شده وارد نمایم. اکنون که این دو امر حاصل شده است، خداوند را شاکرم که بمن توفیق داد تا گامی بسیار کوچک در وادی بسیار بزرگ امنیت شبکه بردارم. از کلیهٔ دانشجویانی که در کلاس درس با راهنمائی‌های ارزندهٔ خود مشوق من در این امر بوده‌اند سپاسگزارم و از همهٔ خوانندگانی که متن کتاب را ملاحظه می‌فرمایند تقاضا دارم تا بر من منت گذاشته و با ارسال انتقادات و پیشنهادات خود به آدرس پست الکترونیک movahed@ictfaculty.ir من را در فعالیت‌های آینده‌ام راهنمائی فرمایند.

در خاتمه لازم است از جناب آقای دکتر غلامعلی حسنی صدر رئیس محترم مرکز آموزش شرکت مخابرات ایران و دانشکدهٔ علمی-کاربردی پست و مخابرات که دستور چاپ این کتاب را صادر فرموده‌اند و همچنین از سرکار خانم معصومهٔ گرامی‌زاده کارشناس محترم مرکز آموزش مخابرات ایران که در تهیهٔ شکل‌ها و جداول کتاب زحمات زیادی را متقبل شده‌اند صمیمانه سپاسگزاری نمایم.

مسعود موحّد
زمستان ۱۳۸۵

فهرست مطالب

پیش گفتار

در این عصر اتصال الکترونیکی جهان، عصر ویروس‌ها و هَکرها، عصر استراق سمع الکترونیک و عصر فریب الکترونیک، واقعاً زمانی قابل تصور نیست که در آن امنیت مطرح نباشد. دو رَوَند متفاوت گرد هم آمده‌اند تا موضوع این کتاب را از اهمیت حیاتی برخوردار سازند. اوّل، رشد انفجارآمیز سیستم‌های کامپیوتری و اتصال آنها از طریق شبکه‌ها، وابستگی هم سازمان‌ها و هم افراد به اطلاعات ذخیره‌شده و مبادله شده با استفاده از این سیستم‌ها را افزایش داده است. این امر بنوبهٔ خود هوشیاری کاربران نسبت به لزوم محافظت دیتا و منابع اطلاعاتی از افشا، تضمین موثق بودن داده‌ها و پیام‌ها، و محافظت سیستم‌ها از حملات مبتنی بر شبکه را ارتقاء بخشیده است. دوّم، مقوله‌های مرتبط با رمزنگاری و امنیت شبکه کمال بیشتری یافته و منجر به ایجاد کاربردهای عملی‌تر وآماده‌تری برای اعمال امنیت شبکه شده‌اند.

اهداف

هدف این کتاب، فراهم‌آوردن یک بررسی عملی از کاربردها و استانداردهای امنیت شبکه است. در مورد کاربردها، تأکید برکاربردهائی خواهد بود که بطور فزاینده‌ای در شبکه‌های سازمان‌ها و اینترنت مورد استفاده قرار گرفته‌اند. در مورد استانداردها نیز تأکید بر روی استانداردهای اینترنتی است که در سطح گسترده‌ای مورد استفاده هستند.

مخاطبین مورد نظر

این کتاب هم برای مخاطبین دانشگاهی و هم برای مخاطبین حرفه‌ای تدارک دیده شده است. بعنوان یک کتاب دانشگاهی، برای یک درس سه واحدی امنیت شبکه دورهٔ لیسانس برای دانشجویان رشتهٔ علوم کامپیوتر، مهندسی کامپیوتر و مهندسی برق مناسب است. این کتاب همچنین می‌تواند یک مرجع قابل مطالعه برای افراد علاقه‌مند باشد.

طراحی کتاب

کتاب در سه قسمت سازمان یافته است:

قسمت اول- رمزنگاری: مرور فشرده‌ای بر الگوریتم‌ها و پروتکل‌های زیرساخت امنیت شبکه دارد که شامل رمزنگاری، توابع درهم‌ساز، امضاءهای دیجیتال و مبادلهٔ کلید است.

قسمت دوم-کاربردهای امنیت شبکه: ابزارهای مهم امنیت شبکه، شامل Kerberos، گواهی‌نامه‌های X.509v3، PGP، S/MIME، امنیت IP، SSL/TLS، SET و SNMPv3 بررسی می‌شوند.

قسمت سوم- امنیت سیستم: به مقوله‌های امنیت در سطح سیستم می‌پردازد که شامل تهدیدها و روش‌های مقابله با آنها در برابر تهاجم و ویروس‌ها، و همچنین استفاده از دیوارهای آتش و سیستم‌های معتمد است.

علاوه بر آن،کتاب شامل یک واژه‌نامهٔ مفصل، یک لیست از علائم اختصاری کثیرالاستعمال و یک فهرست مراجع است. هر فصل شامل تعدادی سؤال، مسأله، لیستی از واژه‌های کلیدی و منابع پیشنهادی برای مطالعهٔ بیشتر است.

خلاصهٔ مفصل‌تری از مطالب طرح‌شده در فصول هر قسمت، در ابتدای آن قسمت ذکر شده است.

مطالب پشتیبان تدریس

برای پشتیبانی مدرسین، مطالب زیر فراهم آمده است:

- **حل المسائل**: پاسخ به تمام سؤالات مرورکنندهٔ بحث و مسائل انتهای هر فصل.
- **اسلایدهای PowerPoint**: مجموعه‌ای از اسلایدها برای تمام فصول، که به امر تدریس کمک می‌کند.
- **فایل‌های PDF**: تمام شکل‌ها و جداول کتاب.
- **لیست پروژه‌ها**: تکالیفی بصورت پروژه، در تمام مقوله‌هائی که در زیر ذکر شده است.

مدرسین می‌توانند برای دسترسی به این مطالب با نمایندهٔ Pearson Education یا Prentice Hall تماس حاصل کنند.

علاوه بر این، پایگاه وِب کتاب مدرسین را با موارد زیر مجهز می‌کند:

- لینک‌هائی به پایگاه‌های وِب درس‌های دیگر، که در تدریس آنها از این کتاب استفاده می‌شود.
- اطلاعات عضویت برای یک لیست پستی اینترنتی برای مدرسین.

سرویس اینترنتی برای اساتید و دانشجویان

برای این کتاب یک صفحهٔ وِب وجود دارد که برای استفادهٔ دانشجویان و اساتید طراحی شده است. این صفحه شامل لینک‌هائی به سایت‌های مرتبط، شکل‌ها و جداول کتاب با فرمت PDF و اطلاعات ثبت نام برای لیست پستی اینترنتی کتاب است. آدرس صفحهٔ وِب **WilliamStallings.com/NetSec2e.html** است. یک لیست پستی اینترنتی هم فراهم شده است تا مدرسینی که این کتاب را تدریس می‌کنند بتوانند اطلاعات، پیشنهادها و سؤالات خود را با یکدیگر و با نویسنده مبادله نمایند. غلط های احتمالی موجود در کتاب نیز در یک لیست غلط‌نامه وارد شده‌اند. علاوه برآن، سایت مرجع دانشجویان علوم کامپیوتر با آدرس **WilliamStallings.com/StudentSupport.html** نیز اسناد، اطلاعات و لینک‌های مفیدی را برای دانشجویان علوم کامپیوتر و افراد حرفه‌ای به همراه دارد.

پروژه‌های مربوط به تدریس امنیت شبکه

برای بسیاری از مدرسین، بخش مهمی از یک درس رمزنگاری و یا امنیت شبکه، یک پروژه یا مجموعه‌ای از پروژه‌هائی است که بتوسط آنها دانشجویان پشتوانهٔ تجربی بهتری را از درس بدست آورند. این کتاب با فراهم آوردن مؤلفهٔ پروژه، نوعی پشتیبانی غیر موازی از درس را بوجود آورده است. جزوهٔ راهنمای استاد نه تنها شامل راهنمائی در مورد تخصیص و ساخت پروژه است، بلکه شامل یک مجموعه از پروژه‌های پیشنهادی است که بازهٔ وسیعی از موضوعات متن را پوشش می‌دهد:

- **پروژه‌های تحقیقاتی:** یک سری از موضوعاتی که به دانشجو آموزش می‌دهد تا چگونه دربارهٔ یک موضوع در اینترنت تحقیق کرده و گزارش تهیه کند.
- **پروژه‌های برنامه‌نویسی:** یک سری از پروژه‌های برنامه‌نویسی که محدودهٔ وسیعی از موضوعات را پوشش داده و می‌توانند به هر زبان مناسبی و روی هر کامپیوتری نوشته شوند.
- **تمرین‌های آزمایشگاهی:** یک سری پروژه‌هائی که شامل برنامه‌نویسی و تجربه آموزی در بارهٔ مفاهیم کتاب است.
- **تکالیف نوشتنی:** یک مجموعه از تکالیف کتبی برای هر فصل.
- **تکالیف مطالعاتی/ گزارش‌دهی:** یک لیست از مقالات موجود در مقوله‌های ذیربط، یکی برای هر فصل، که می‌تواند به دانشجو محول شده تا از روی آن گزارش تهیه نماید.

مطالب جدید در ویرایش سوم

در ظرف سه سالی که از چاپ دوم این کتاب می‌گذرد، مقولهٔ امنیت، نوآوری‌ها و توسعه‌های جدیدی را بخود دیده است. در این چاپ جدید، من کوشیده‌ام تا در ضمن این که ساختار کلی کتاب را تغییر ندهم، تغییرات جدید را نیز درآن منعکس نمایم. برای شروع این بازنگری، چاپ دوم بتوسط اساتید متعددی که این کتاب را تدریس نموده‌اند مرور گردید. علاوه برآن عده‌ای از افراد شاغل در این حرفه نیز فصول این کتاب را مطالعه نمودند. نتیجهٔ کار این شد که در برخی موارد، انشای مطلب برای فهم بهتر تصحیح گردید و شکل‌ها تغییر یافت. همچنین تعدادقابل توجه‌ای مسأله، به فصول اضافه گردید.

علاوه بر این اصلاحات که در جهت فهم بهتر مطلب انجام شد، تغییرات قابل توجهی نیز در برخی سرفصل‌ها ایجاد گردید. عمدهٔ آنها بشرح زیر است:

- **رمزهای دنباله‌ای**: رمزهای دنباله‌ای در تعدادی از پروتکل‌های امنیت شبکه و کاربردها بکار می‌روند. چاپ سوم، این مقوله را پوشش داده و پراستفاده‌ترین الگوریتم این حوزه که RC4 است را توصیف کرده است.
- **زیرساخت کلید عمومی (PKI)**: این مقولهٔ مهم در ویرایش جدید مورد بحث قرار گرفته است.
- **حملات انکار سرویس توزیع شده (DDoS)**: حملات DDoS در سال‌های اخیر توجه زیادی را بخود جلب کرده‌اند.
- **معیارهای مشترک برای ارزیابی امنیت تکنولوژی اطلاعات**: معیارهای مشترک، به یک چهارچوب بین‌المللی برای تشریح نیازهای امنیتی و ارزیابی محصولات و پیاده‌سازی‌ها مبدل شده‌اند.

علاوه براین، بسیاری از سایر مطالب کتاب بازنگری و بروزرسانی شده‌اند.

ارتباط این کتاب با کتاب رمزنگاری و امنیت شبکه، ویرایش چهارم

این کتاب از کتاب *Cryptography and Network Security, Fourth Edition* (CNS4e) اقتباس شده است. CNS4e پوشش جامعی از مبحث رمزنگاری داشته که شامل تحلیل مفصلی از الگوریتم‌ها و پشتوانهٔ ریاضی آنهاست که خود قریب به ۴۰۰ صفحه از کتاب را دربر می‌گیرد.

Network Security Essentials: Applications and Standards, Third Edition (NSE3e) درعوض مرور مختصری بر مباحث فوق در فصول ۲ و ۳ دارد. مابقی NSE3e شامل کلیهٔ مطالب باقیمانده CNS4e است. NSE3e همچنین امنیت SNMP را پوشش می‌دهد که در CNS4e پوشش داده نشده است. بنابراین NSE3e برای تدریس در کالج‌ها و برای خوانندگان حرفه‌ای که علاقهٔ آنها بیشتر به کاربردهای امنیت شبکه بوده و نیاز و توجه کمتری به عمیق شدن در تئوری‌ها و اصول رمزنگاری دارند، طراحی شده است.

فصل ۱

مقدمه

زمه‌های **امنیت اطلاعات** در درون یک سازمان، در طی دهه‌های اخیر دو تغییر عمده یافته است. قبل از استفادهٔ گسترده از تجهیزات پردازش داده‌ها، امنیت اطلاعاتی که از نظر یک سازمان ارزشمند تلقی می‌شد عمدتاً از طـــریق روش‌های فیزیکی و مدیریتی فراهم می‌گردید. مثالی از روش فیزیکی، استفاده از کمدها و فایل‌های مستحکم با قفل های رمزی برای حفاظت از اسناد مهم است. مثالی از روش مدیریتی، جمع‌آوری اطلاعات گزینشی در هنگام استخدام پرسنل است.

با ورود رایانه، نیاز به لوازم خودکار برای حفاظت از فایل‌ها و سایر اطلاعات ذخیره‌شده در رایانه آشکار گردید. این موضوع علی‌الخصوص در مورد یک سیستم به اشتراک گذاشته شده همانند یک سیستم اشتراک زمانی جدی‌تر بوده و حتی در مورد سیستم‌هائی که از طریق شبکهٔ تلفنی، شبکهٔ دیتا، و یا اینترنت قابل دست‌یابی هستند حیاتی‌تر است. نام کلی مجموعهٔ لوازمی که برای حفاظت داده‌ها و خنثی کردن نیات بداندیشان طراحی شده است، **امنیت رایانه** است.

تغییر عمدهٔ دیگر که امنیت را تحت تأثیر قرار داده است، ورود سیستم‌های توزیع شده و استفاده از تسهیلات شبکه‌ای و ارتباطی برای حمل داده‌ها بین پایانه و رایانه، و بین رایانه و رایانه است. معیارهای امنیت شبکه برای حفاظت از داده‌ها در هنگام انتقال آن‌ها ضروری است. در واقع اصطلاح **امنیت شبکه** تا حدودی گمراه‌کننده است زیرا تمام مشاغل، دولت و سازمان‌های آموزشی، تجهیزات پردازش دیتای خود را با مجموعه‌ای از شبکه‌های متعامل بهم وصل کرده‌اند. چنین مجموعه‌ای را اغلب اینترنت نامند و در رابطه با آن، اصطلاح **امنیت اینترنت** و یا امنیت بین شبکه‌ای بکار می‌رود.

بین این دو نوع امنیت، مرزبندی روشنی وجود ندارد. مثلاً یکی از معروف‌ترین انواع حملات بر روی سیستم‌های اطلاعاتی، ویروس رایانه‌ای است. یک ویروس ممکن است از طریق یک دیسکت و یا یک دیسک نوری بصورت فیزیکی وارد شده و متعاقباً روی رایانه‌ای بارگذاری شود. ویروس‌ها همچنین ممکن است از طریق اینترنت وارد شوند. در هر یک از دو مورد، همین‌که ویروس روی یک سیستم رایانه مستقر گردید، لوازم امنیت داخلی رایانه برای تشخیص آن و نجات رایانه از شرّ آن، مورد نیاز خواهد بود.

این کتاب بر روی امنیت اینترنت متمرکز شده است که شامل معیارهائی برای شناسائی، جلوگیری، تشخیص و اصلاح تخلفات امنیتی که اطلاعات را تحت تأثیر قرار می‌دهند، می‌باشد. این مقولهٔ گسترده‌ای است که موارد وسیعی را در بردارد. برای این‌که احساسی از مسائلی که دراین کتاب مورد بحث قرار می‌گیرد داشته باشید، به مثال‌های زیر از مخاطرات امنیتی توجه کنید:

۱- کاربرA یک فایل را برای کاربرB می‌فرستد. فایل شامل اطلاعات حساسی (مثل اطلاعات مالی) است که بایستی از دسترس بیگانه دور باشد. کاربر C که مجاز به خواندن فایل نمی‌باشد، قادر به پائیدن انتقال اطلاعات بوده و یک نسخه از فایل را در هنگام ارسال به دست می‌آورد.

۲- یک مدیر شبکه D، پیامی را برای رایانهٔ E که تحت مدیریت اوست می‌فرستد. پیام به رایانهٔ E فرمان می‌دهد که فایل افراد مجاز را به‌روزکرده و نام چندین کاربر جدیدکه می‌توانند به سیستم دست یابند را در آن وارد کند. کاربر F پیام را دزدیده، محتویات آن را با اضافه کردن و یا حذف کردن نام‌های دلخواه خود تغییر داده و سپس آن را برای E می‌فرستد. E پیام را با تصور این‌که از سوی مدیر D ارسال شده پذیرفته و فایل افراد مجاز را بر اساس آن به‌روز درمی‌آورد.

۳- بجای دزدیدن یک پیام، کاربر F، پیام مورد نظر خویش با ورودی‌های دلخواه خود را ساخته و آن را طوری برای E می‌فرستد که E خیال می‌کند از جانب مدیرD صادر شده است و بنابراین فایل افراد مجاز را بر اساس آن به‌روز در می‌آورد.

۴- کارمندی بدون اخطار قبلی اخراج می‌شود. مدیر امور اداری پیامی به سیستم سِرور می‌فرستد تا حساب او را از اعتبار خارج نماید. وقتی این عمل انجام می‌شود، سِرور بایستی تذکاریه‌ای را برای فایل کارمند ارسال کرده و انجام عمل را تأئید کند. کارمند قادر به استراق سمع پیام بوده و ارسال آن را آنقدر به تأخیر می‌اندازد تا خود بتواند آخرین دست‌یابی به سِرور را پیدا کرده و اطلاعات حساس را استخراج کند. پس از آن پیام ارسال شده، عمل صورت پذیرفته و تأئید آن داده می‌شود. عمل این کارمند ممکن است برای مدت قابل ملاحظه‌ای کشف نشود.

۵- یک پیام ممکن است از طرف یک مشتری برای خرید سهام به کارگزار او فرستاده شود. متعاقباً ممکن است قیمت سهام پائین آمده و مشتری ارسال چنین پیامی را انکار کند.

اگرچه این لیست بهیچ وجه تمام تهدیدهای امنیتی را پوشش نمی‌دهد، ولی نمایش‌گر محدودهٔ وسیع امنیت شبکه است.

امنیت بین شبکه‌ای، هم جذاب و هم پیچیده است. برخی دلایل آن به قرار زیراند:

۱- امنیتی که ارتباطات و شبکه‌ها درگیرآنند آنچنان که در ابتدا برای یک تازه‌کار جلوه می‌کند، ساده نیست. اهداف این امنیت ممکن است خیلی ساده بیان شوند و در واقع نیازهای اساسی سیستم‌های امنیتی اغلب با یک کلمه بیان می‌شوند: محرمانگی، اعتبارسنجی، عدم انکار، صحت. اما مکانیسم‌هائی که بایستی برای حصول این نیازها استفاده شوند اغلب بسیار پیچیده بوده و فهم آنها ممکن است استدلالات زیرکانه‌ای را ایجاب کند.

۲- در طراحی یک مکانیسم و یا الگوریتم امنیتی بخصوص، همیشه بایستی حملات مؤثر بر علیه آن ویژگی امنیتی را در نظر داشت. در بسیاری موارد، حملات موفقیت‌آمیز با نگاهی کاملاً متفاوت به مسأله طراحی می‌شوند و از نقاط ضعف غیرقابل انتظار مکانیسم استفاده می‌کنند.

۳- بعلت نکتهٔ بالا، روش‌های مورد استفاده برای فراهم نمودن سرویس‌های بخصوص، اغلب ساده به ذهن نمی‌آیند. اغلب بیان یک نیاز امنیتی نمی‌تواند زنجیرهٔ پیچیده‌ای از عملیاتی که طراحی شده‌اند را به راحتی توجیه کند. تنها زمانی این معیارها معنی پیدا می‌کنند که شگردهای ضدامنیتی آنها مطالعه شوند.

۴- وقتی مکانیسم‌های متفاوت امنیتی طراحی شدند، لازم است تصمیم گرفته شود که در کجا باید از آنها استفاده کرد. این امر هم در مورد محل فیزیکی آنها (این که مکانیسم‌های امنیتی در کدام نقطهٔ شبکه مورد نیازند) و هم در مفهوم منطقی آنها (این که مکانیسم‌های امنیتی درکدام لایه و یا لایه‌های یک معماری، مثل TCP/IP، باید گنجانده شوند)، صحیح می‌باشد.

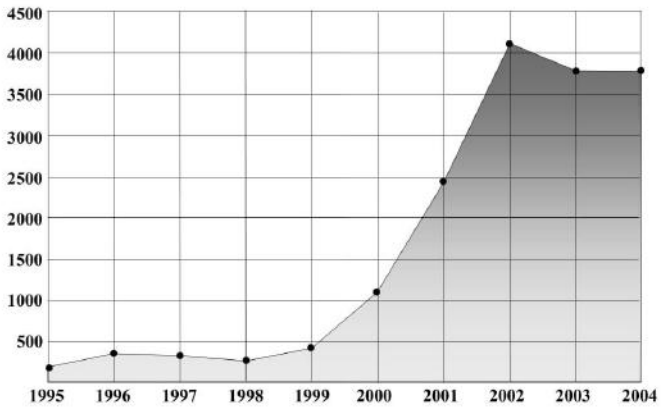
۵- مکانیسم‌های امنیتی معمولاً شامل بیش از یک الگوریتم و یا یک پروتکل هستند. آنها معمولاً اشخاص را مقید می‌کنند تا برخی اطلاعات سرّی را در اختیار داشته باشند (مثلاً یک کلید رمزنگاری) که این خود سؤالاتی در زمینهٔ تولید، توزیع و حفاظت از این اطلاعات سرّی را مطرح می‌سازد. همچنین اتکاء به رفتار برخی پروتکل‌های ارتباطی ممکن است وظیفهٔ طراحی مکانیسم‌های امنیتی را با مشکل مواجه سازد. بعنوان مثال، اگر عملکرد صحیح یک مکانیسم امنیتی نیاز به محدود کردن زمان انتقال یک پیام بین فرستنده و گیرنده را داشته باشد، آنگاه هر پروتکل یا شبکه‌ای که تأخیر زمانی متغیر و یا غیرقابل انتظاری در ارسال پیام را ایجاد نماید، ممکن است این معیار امنیتی را بی‌معنی سازد.

بنابراین، نکات بسیاری را بایستی در نظر داشت. این فصل یک نگاه کلّی به مسأله داشته و ساختار مطالب بقیهٔ کتاب را سازمان می‌دهد. موضوع را با یک بحث کلی در مورد سرویس‌های امنیت شبکه و انواع حملات امنیتی که این سرویس‌ها بایستی به آنها پاسخ دهند، شروع می‌کنیم. آنگاه یک مدل عمومی که سرویس‌ها و مکانیسم‌های امنیتی بایستی از منظر آن دیده شوند را ارائه می‌دهیم.

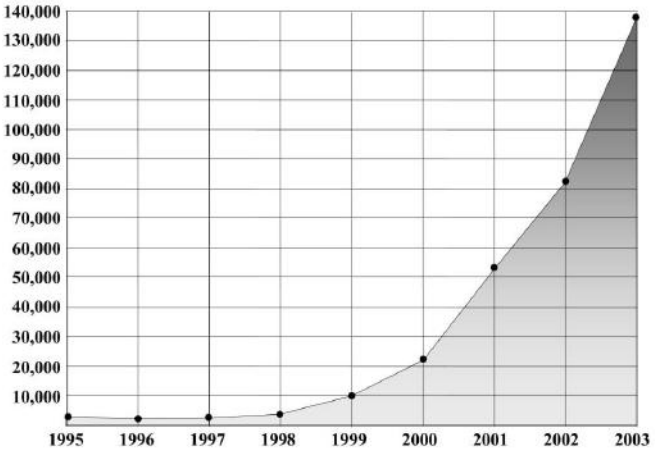
۱-۱ رَوَند امنیت

در سال ۱۹۹۴ میلادی، گروه معماری اینترنت (IAB) گزارشی با عنوان «امنیت در معماری اینترنت» را منتشر نمود (RFC 1636). این گزارش بیان‌کنندهٔ توافق جمعی بر این مطلب بود که اینترنت به امنیت بیشتر و بهتری نیاز دارد. در ضمن، زمینه‌های کلیدی مکانیسم‌های امنیتی نیز در این گزارش مشخص شده بود. در این گزارش به مقوله‌هائی چون نیاز به مصون کردن زیرساخت شبکه از پایش‌های غیرمجاز، کنترل ترافیک شبکه و امن کردن ترافیک کاربرانتهائی- به-کاربر انتهائی با استفاده از مکانیسم‌های رمزنگاری و اعتبارسنجی اشاره شده بود.

این نگرانی‌ها کاملاً بجا هستند. برای تأئید این امر به گزارش رَوَند امنیت مرکز هماهنگی تیم پاسخگوئی به فوریت‌های رایانه‌ای(CERT/CC) توجه کنید. شکل ۱-۱الف رَوَند رشد آسیب‌پذیری‌های مرتبط- با- اینترنت گزارش شده به CERT در طی یک دورهٔ ده‌ساله را نشان می‌دهد. اینها شامل ضعف‌های امنیتی موجود در سیستم‌های عامل رایانه‌ها (مثل Windows و Linux) و همچنین آسیب‌پذیری‌های موجود در مسیریاب‌های اینترنت و سایر تجهیزات شبکه‌اند. شکل ۱-۱ب تعداد مشکلات مرتبط- با- امنیت گزارش شده به CERT را نشان می‌دهد. اینها شامل حملات انکار سرویس، جعل IP که در آن مهاجمین بسته‌هائی با آدرس IP جعلی خلق کرده و کاربردهائی را که اعتبارسنجی مبتنی بر IP دارند را فریب می‌دهند، و فرم‌های متنوعی از استراق‌سمع و بوکشیدن بسته‌ها که در آن حمله‌کنندگان اطلاعات انتقال‌یافته از قبیل اطلاعات مربوط به اتصال به سیستم و محتوای پایگاه‌های داده را می‌خوانند، هستند.



شکل ۱-۱الف آسیب‌پذیری‌های گزارش شده بتوسط CERT



شکل ۱-۱ ب حوادث امنیتی گزارش شده بتوسط CERT

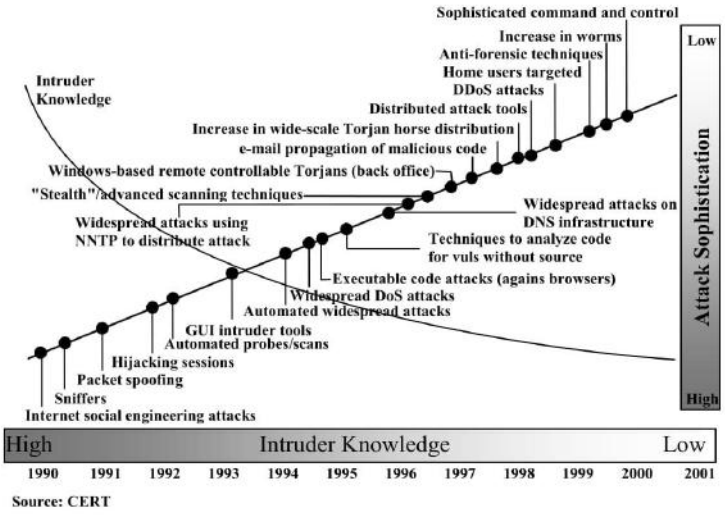
در طول زمان، حملات به اینترنت و سیستم‌های متصل به اینترنت پیچیده‌تر شده در حالی‌که مهارت و معلومات لازم برای انجام حمله کاهش یافته است (شکل ۲-۱). در ضمن، حملات فرم خودکارتری به خود گرفته و می‌توانند صدمات بیشتری را وارد نمایند.

این افزایش در تعداد حملات، با استفاده بیشتر از اینترنت و افزایش پیچیدگی پروتکل‌ها، کاربردها و خود اینترنت مقارن بوده‌اند. زیرساخت‌های حیاتی، بطور روزافزونی برای عملیات خود به اینترنت متکی‌اند. کاربران منفرد نیز به امنیت اینترنت، پست الکترونیک، وِب و کاربردهای مبتنی بر وِب بیش از پیش اتکا دارند. در نتیجه یک محدودهٔ وسیع از تکنولوژی‌ها و ابزارها، برای مقابله با این تهدیدهای فزاینده مورد نیازند. در سطح ابتدائی، الگوریتم‌های رمزنگاری با هدف محرمانگی و اعتبارسنجی اهمیت زیادتری دارند. همچنین طراحان نیازمند تمرکز بر پروتکل‌های مبتنی- بر- اینترنت و آسیب‌پذیری‌های سیستم‌های عامل و کاربردها می‌باشند. این کتاب تمام این زمینه‌های تکنیکی را بررسی می‌نماید.

۱-۲ معماری امنیت OSI

برای تعیین نیازهای امنیتی یک سازمان، و برای ارزیابی و انتخاب خط‌مشی‌ها و محصولات امنیتی مختلف، مدیر مسئول امنیت نیازمند یک روش سیستماتیک برای تشخیص نیازهای امنیتی و مشخص کردن روش‌های تأمین این نیازهاست. این امرخود بقدر کافی در یک محیط متمرکز پردازش داده‌ها پیچیده بوده و در صورت استفاده از شبکه‌های LAN و WAN پیچیدگی آن چندین برابر می‌شود.

توصیه‌نامهٔ X.800 سازمان ITU-T با نام، *معماری امنیت برای OSI*، چنین روش سیستماتیکی را تعریف می‌کند. معماری امنیت OSI برای سازماندهی وظیفهٔ ایجاد امنیت برای مدیران، مفید است. علاوه برآن چون این معماری بصورت یک استاندارد بین‌المللی طراحی شده است، سازندگان رایانه‌ها و تجهیزات ارتباطی، خصوصیات امنیتی محصولات خود را بر اساس تعاریف، سرویس‌ها و مکانیسم‌های این معماری فراهم نموده‌اند.



شکل ۲-۱ رَوَند پیچیدگی حملات و آگاهی‌های مهاجم

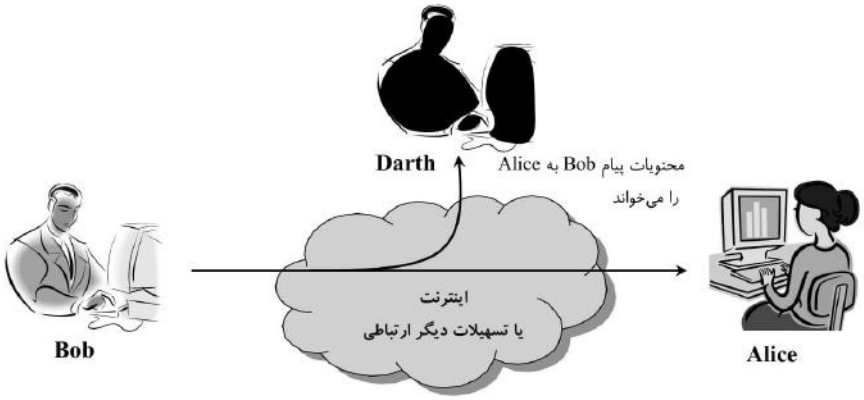
برای مقاصد ما، معماری امنیت OSI یک دید کلی، اگرچه مبهم، از مباحثی که در این کتاب به آنها پرداخته شده است را فراهم می‌سازد. معماری امنیت OSI بر حملات امنیتی، مکانیسم‌ها و سرویس‌ها تمرکز دارد. این عناوین را بطور خلاصه چنین می‌توان تعریف کرد:

- **حملهٔ امنیتی**: هر عملی که امنیت اطلاعات متعلق به یک سازمان را به مخاطره اندازد.
- **مکانیسم امنیتی**: سازوکاری که برای تشخیص، جلوگیری و یا بخودآمدن از یک حملهٔ امنیتی بکار رود.
- **سرویس امنیتی**: سرویسی که امنیت سیستم‌های پردازش اطلاعات و انتقال اطلاعات در یک سازمان را ارتقاء بخشد. هدف این سرویس‌ها مقابله با حملات امنیتی بوده و از یک یا چند مکانیسم امنیتی برای فراهم آوردن سرویس استفاده می‌کنند.

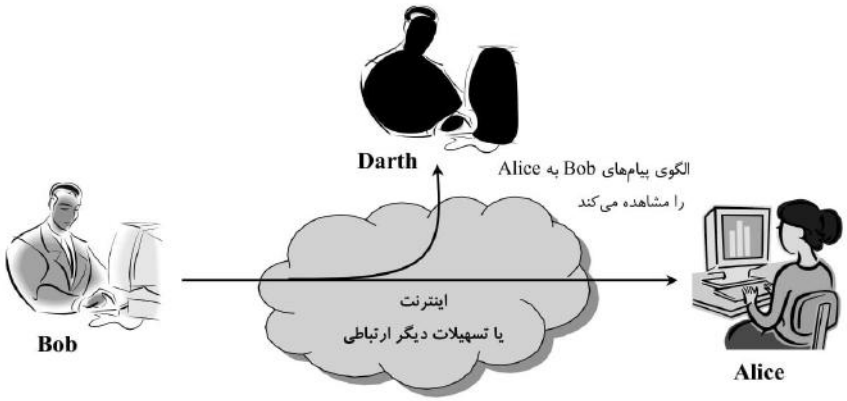
در ادبیات امنیتی، واژه‌های تهدید و حمله مکرراً بکار رفته و تقریباً دارای یک معنی هستند. جدول ۱-۱ تعاریفی که از *واژه‌نامهٔ امنیت اینترنت* RFC 2828 اقتباس شده است را نشان می‌دهد.

جدول ۱-۱ تهدیدها و حمله‌ها (RFC 2828)

تهدید
استعداد بالقوهٔ نقض امنیت در صورت وجود شرایط، قابلیت، عمل، یا اتفاقی که امنیت را مورد مخاطره قرار دهد. یعنی یک تهدید یک خطر احتمالی است که ممکن است از یک نقطهٔ آسیب‌پذیر امنیتی سوءاستفاده نماید.
حمله
هجومی بر امنیت سیستم است که از یک تهدید هوشمند سرچشمه می‌گیرد. یعنی عملی هوشمندانه است که تلاشی زیرکانه برای حمله به سرویس‌های امنیتی و نقض سیاست‌های امنیتی سیستم دارد.



(الف) افشای محتویات پیام



(ب) تحلیل ترافیک

شکل ۳-۱ حملات غیرفعال

۳-۱ حملات امنیتی

یک روش مناسب برای دسته‌بندی حملات امنیتی که هم در X.800 و هم در RFC 2828 استفاده شده است، تقسیم این حملات به دو دستهٔ *حملات غیرفعال* و *حملات فعال* می‌باشد. یک حملهٔ غیرفعال تلاش دارد تا اطلاعات سیستم را به دست آورده و یا از آن استفاده کند ولی روی منابع سیستم تأثیر نمی‌گذارد. یک حملهٔ فعال سعی دارد تا منابع سیستم را تغییر داده و یا بر عملیات آن تأثیر بگذارد.

حملات غیرفعال

حملات غیرفعال دارای ماهیت استراق سمع و یا شنود اطلاعات انتقال یافته است. هدف دشمن در این نوع حمله، دست‌یابی به اطلاعات است. دو نوع حملهٔ غیرفعال، یکی افشای محتویات پیام و دیگری تحلیل ترافیک است.

افشای محتویات پیام را می‌توان بسهولت درک کرد (شکل ۳-۱الف). یک مکالمهٔ تلفنی، یک پیام پست الکترونیک، و یک فایل انتقال یافته ممکن است شامل اطلاعات حساس و یا محرمانه باشند. علاقه‌مندیم که از دست‌یابی دشمن به این اطلاعات جلوگیری نمائیم.

نوع دیگر حملهٔ غیرفعال، **تحلیل ترافیک** است (شکل ۳-۱ب). فرض کنید با توسل به روشی محتویات پیام و یا سایر اطلاعات ترافیکی را طوری تغییر داده‌ایم که دشمنان، حتی اگر پیام را سرقت کنند، نتوانند اطلاعات آن را استخراج نمایند. تکنیک معمول برای این کار رمزنگاری است. ولی حتی اگر حفاظت رمزنگاری را نیز در جای خود داشته باشیم، یک دشمن بازهم ممکن است بتواند الگوی این پیام‌ها را کشف کند. دشمن می‌تواند محل و هویت طرفین ارتباط را تعیین کرده و از تعداد و طول پیام‌هائی که بین آنها ردوبدل می‌شود، آگاه شود. این اطلاعات ممکن است در حدس ماهیت ارتباطی که در حال انجام است مفید باشد.

تشخیص حملات غیرفعال بسیار مشکل است زیرا تأثیری روی خود داده‌ها نمی‌گذارند. معمولاً ترافیک پیام با روند عادی ارسال و دریافت شده و نه فرستنده و نه گیرنده از اینکه طرف سومی پیام را خوانده و یا الگوی ترافیک را ملاحظه کرده است مطلع نمی‌شوند. با وجود این معقول است که از موفقیت چنین حملاتی، معمولاً با رمزنگاری، جلوگیری کرد. بنابراین برای مقابله با حملات غیرفعال تأکید بر پیش‌گیری، بجای تشخیص، است.

حملات فعال

حملات فعال شامل ایجاد تغییرات در جریان دیتا و یا خلق جریان جدیدی از داده‌هاست و می‌توان آنها را به چهار دسته تقسیم کرد: نقاب‌دار، بازخوانی، تغییر پیام و انکار سرویس.

یک حملهٔ **نقاب‌دار** وقتی صورت می‌پذیرد که شخصی یا واحدی وانمود کند که شخص یا واحد دیگری است (شکل۴-۱الف). یک حملهٔ نقاب‌دار معمولاً با حملهٔ فعال دیگری همراه است. بعنوان مثال، دنباله‌های اعتبارسنجی می‌توانند دزدیده شده و پس از این‌که یک عمل اعتبارسنجی معتبر به پایان رسید، بازخوانی شوند و بدین ترتیب به یک واحد مجاز که دارای سطح دست‌یابی پائین‌تری است اجازه دهد تا با جعل هویت واحد دیگری که دارای سطح دست‌یابی بالاتری است، امتیازات بیشتری کسب کند.

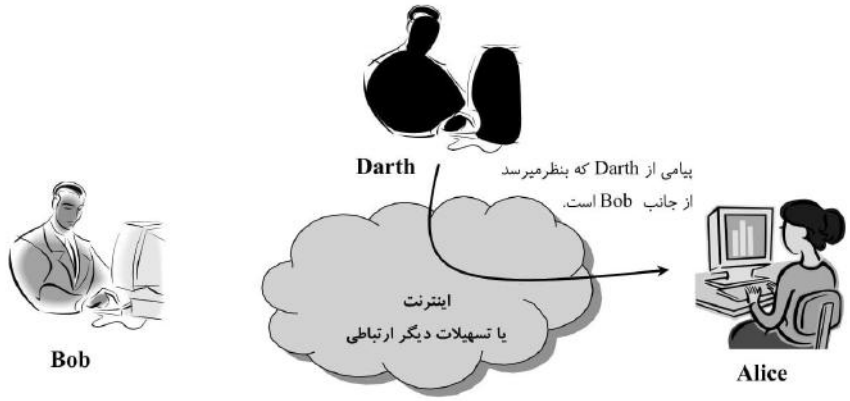
حملهٔ **بازخوانی** شامل دزدیدن غیرفعال واحدهای دیتا و ارسال مجدد آنها با تأخیر، برای ایجاد یک اثر مخرب است (شکل ۴-۱ب).

تغییر پیام بسادگی دارای این معنی است که بخشی از یک پیام قانونی تغییر داده شود، یا این‌که پیام‌ها تأخیر یافته یا نظم آنها برهم زده شود تا نهایتاً باعث اثری غیرمجاز گردند (شکل۴-۱ج). مثلاً پیام " به آقای حمید حمیدی اجازه دهید تا فایل حساب‌های محرمانه را مشاهده کند" به پیام "به آقای مجید مجیدی اجازه دهید تا فایل حساب‌های محرمانه را مشاهده کند" تغییر می‌یابد.

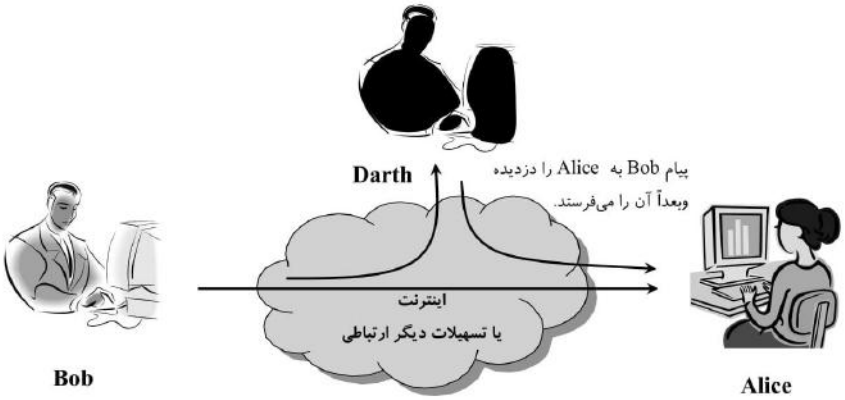
انکار سرویس مانع کارکرد نرمال تجهیزات شده و یا از مدیریت تسهیلات ارتباطی جلوگیری می‌نماید (شکل ۴-۱د). این حمله ممکن است هدف معینی را نشانه بگیرد. مثلاً واحدی ممکن است تمام پیام‌هائی را که برای یک مقصد بخصوص

ارسال می‌شوندحذف کند(مثل بازرسی امنیتی). صورت دیگری از انکار سرویس، ایجاد اختلال در تمام شبکه است که این کار یا با ایجاد خرابی در شبکه و یا با ارسال پیام‌های بسیار زیاد به شبکه بمنظور ایجاد اختلال در عملکرد آن صورت می‌پذیرد.

حملات فعال دارای مشخصاتی خلاف حملات غیرفعال هستند. در حالی‌که تشخیص حملات غیرفعال مشکل است ولی روش‌هائی برای جلوگیری از موفقیت آنها موجود می‌باشد. برعکس، جلوگیری از حملات فعال کاری بس دشوار است زیرا نیاز به محافظت فیزیکی تمام تسهیلات و مسیرهای ارتباطی در تمام زمان‌ها دارد. بجای این کار، هدف تشخیص این حملات و رفع مشکلات و یا تأخیرهائی است که این حملات ممکن است در شبکه ایجاد نمایند. چون تشخیص، خود دارای اثر بازدارندگی است، این کار ممکن است به جلوگیری از حملات نیز کمک کند.

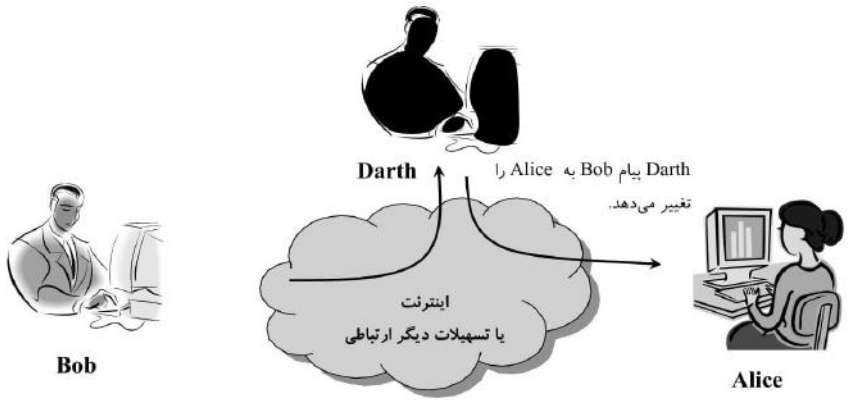


(الف) نقاب‌دار



(ب) بازخوانی

شکل ۴-۱ حملات فعال



(ج) تغییر پیام

(د) انکار سرویس

شکل ۴-۱ حملات فعال (ادامه شکل قبل)

۴-۱ سرویس‌های امنیتی

X.800 یک سرویس امنیتی را بعنوان سرویسی تعریف می‌کند که بتوسط یک لایهٔ پروتکلی سیستم‌های باز ارتباطی فراهم شده و امنیت کافی برای سیستم‌ها و یا انتقال داده‌ها را فراهم می‌سازد. شاید RFC 2828 تعریف روشن‌تری را ارائه کند که چنین است: سرویس امنیتی یک سرویس ارتباطی، و یا پردازشی است که بتوسط یک سیستم ایجاد شده تا نوع تعریف شده‌ای از حفاظت را برای منابع سیستم بوجود آورد. سرویس‌های امنیتی، خط‌مشی‌های امنیتی را از طریق مکانیسم‌های امنیتی پیاده‌سازی می‌کنند.

X.800 این سرویس‌ها را به پنج گروه و چهارده سرویس مشخص تقسیم می‌کند (جدول ۲-۱). هریک از این گروه‌ها را بنوبت بررسی می‌کنیم.

اعتبارسنجی

سرویس اعتبارسنجی مسئول اطمینان یافتن از این است که یک ارتباط معتبر است. در مورد یک پیام تنها، مانند یک سیگنال هشداردهنده یا آلارم، وظیفهٔ سرویس اعتبارسنجی این است که به گیرندگان پیام اطمینان دهد که این سیگنال واقعاً از منبعی که ادعا دارد سرچشمه گرفته است. در مورد یک تعامل دائمی، همانند اتصال یک پایانه به یک رایانه، موضوع دو جنبه دارد. اول این که در هنگام برقراری ارتباط، سرویس اعتبارسنجی به طرفین ارتباط اطمینان دهد که طرف مقابل معتبر بوده و هریک از طرفین واقعاً همانی هستند که ادعا می‌کنند. دوم این که سرویس اعتبارسنجی بایستی تضمین کند که اتصال بین دو کاربر در اشغال فرد ثالثی که بتواند خود را بجای هریک از طرفین جازده و ارسال و دریافت غیرمُجازی را ایجاد نماید، درنیامده است.

دو سرویس اعتبارسنجی مشخص در استاندارد تعریف شده‌اند:

- **اعتبارسنجی واحد نظیر**: برای تأئید هویت یک واحد نظیر(peer) در یک مجتمع رایانه‌ای بکار می‌رود. دو واحد را نظیر هم خوانند اگر آنها در دو سیستم مختلف در پروتکل یکسانی پیاده‌سازی شوند. مثلاً مدول‌های TCP در دو سیستم ارتباطی، نظیر هم هستند. استفاده از این سرویس در هنگام برقراری ارتباط و یا در خلال انتقال داده‌هاست. این سرویس تلاش می‌کند تا این اطمینان را فراهم سازد که یک واحد خود را بجای واحد دیگر جانزده و یا یک ارتباط قدیمی را بازخوانی نکرده باشد.
- **اعتبارسنجی منبع دیتا**: برای تأئید هویت منبع یک واحد دیتا بکار می‌رود. حفاظتی در برابر تکرار و یا تغییر داده‌ها ایجاد نمی‌کند. این نوع سرویس از کاربردهائی همانند پست الکترونیک که در آنها هیچ تعاملی بین واحدهای مرتبط وجود ندارد، حمایت می‌کند.

کنترل دست‌یابی

در مقولهٔ امنیت شبکه، کنترل دست‌یابی به مفهوم قابلیت محدودکردن و کنترل دست‌یابی به سیستم‌های میزبان و کاربردها از طریق پیوند ارتباطی است. برای حصول این امر، هر واحد که تمایل به دست‌یابی به سیستم یا کاربردی را دارد بایستی اول شناسائی و یا اعتبارسنجی گردد تا حق دست‌یابی مختص خودش به او داده شود.

محرمانگی داده‌ها

محرمانگی عبارت از حفاظت اطلاعات انتقال‌یافته در برابر حملات غیرفعال است. در رابطه با محتویات یک انتقال دیتا چندین سطح حفاظت را می‌توان تعریف کرد. وسیع‌ترین سرویس، تمام دیتای انتقال‌یافته بین دو کاربر در طول زمان را محافظت می‌کند. مثلاً وقتی یک اتصال TCP بین دو سیستم برقرار می‌شود، این حفاظت وسیع از برملا شدن هرگونه داده کاربر روی اتصال TCP جلوگیری می‌کند. شکل ضعیف‌تر سرویس این است که حفاظت فقط از یک پیام و یا حتی بخش‌های مشخصی از یک پیام صورت پذیرد. این سرویس پالایش شده کمتر از سرویس وسیع مفید بوده و حتی ممکن است به پیچیدگی و هزینهٔ بیشتری منجر شود.

جدول ۲-۱ سرویس‌های امنیتی(X.800)

اعتبارسنجی (AUTHENTICATION)

اطمینان از اینکه واحد ارتباطی همان است که ادعا می‌کند.

اعتبارسنجی واحد نظیر(Peer Entity Authentication)
در رابطه با یک ارتباط منطقی تعریف شده تا نسبت به هویت واحدهای مرتبط ایجاد اطمینان نماید.

اعتبارسنجی منبع دیتا (Data-Origin Authentication)
در یک انتقال غیراتصالی، این اطمینان را ایجاد می‌کند که منبع دیتای دریافت شده همان است که ادعا می‌کند.

کنترل دست‌یابی (ACCESS CONTROL)

ممانعت از استفادهٔ غیرمجاز از یک منبع (یعنی این سرویس کنترل می‌کند که چه کسی می‌تواند به یک منبع دست یافته، تحت چه شرایطی این دست‌یابی می‌تواند انجام شود، و آنهائی که دست‌یابی پیدا کنند مجاز به انجام چه کارهائی هستند).

محرمانگی داده‌ها (DATA CONFIDENTIALITY)

حفاظت از داده‌ها در برابر افشای غیرمجاز

محرمانگی اتصالی (Connection Confidentiality)
حفاظت از تمام دادهٔ کاربر در طول اتصال

محرمانگی غیراتصالی (Connectionless Confidentiality)
حفاظت از تمام دادهٔ کاربر در یک بلوک منفرد

محرمانگی میدان انتخابی (Selective-Field Confidentiality)
محرمانگی میدان‌های انتخاب شده دادهٔ کاربر در تمام یک اتصال و یا در یک بلوک دیتا

محرمانگی جریان ترافیک (Traffic-Flow Confidentiality)
حفاظت از اطلاعاتی که ممکن است از مشاهدهٔ جریان ترافیک داده‌ها بدست آید.

صحت داده‌ها (DATA INTEGRITY)

اطمینان از اینکه دادهٔ دریافت شده دقیقاً همانی است که بتوسط یک واحد معتبر ارسال شده است (یعنی تغییر نیافته، اضافه نشده، حذف نشده و بازخوانی نشده است).

صحت اتصالی با بازیابی (Connection Integrity with Recovery)
صحت کل دادهٔ کاربر روی یک اتصال را تأئید کرده و هرگونه تغییر، حذف، اضافه، و یا بازخوانی داده‌ها در یک دنبالهٔ کامل دیتا را تشخیص می‌دهد. بازیابی داده نیز مورد نظر است.

صحت اتصالی بدون بازیابی(Connection Integrity without Recovery)
همانند مورد بالاست، ولی تنها تشخیص و نه بازیابی مورد توجه است.

صحت اتصالی میدان‌های انتخابی
(Selective-Field Connection Integrity)
صحت میدان‌های انتخاب‌شده‌ای از دادهٔ کاربر در یک بلوک را بعهده داشته و تعیین می‌کند که آیا میدان‌های انتخاب‌شده تغییریافته، حذف یا اضافه شده و یا بازخوانی شده‌اند.

صحت غیراتصالی (Connectionless Integrity)
صحت یک بلوک دیتای منفرد را تعیین نموده و ممکن است قدرت تشخیص تغییر داده را داشته باشد. علاوه برآن توان محدودی از تشخیص بازخوانی را نیز می‌تواند تأمین کند.

صحت غیراتصالی میدان‌های انتخابی
(Selective-Field Connectionless Integrity)
صحت میدان‌های انتخاب شده‌ای در یک بلوک دیتای غیراتصالی را تعیین می‌کند. مشخص می‌کند که آیا میدان‌های مورد نظر تغییر یافته‌اند.

عدم انکار (NONREPUDIATION)

در برابر انکار یکی از طرفین ارتباط نسبت به انکار تمام و یا بخشی از ارتباط، ایجاد حفاظت می‌کند.

عدم انکار، مبدأ (Nonrepudiation, Origin)
اثبات اینکه پیام بتوسط طرف اصلی ارسال شده است.

عدم انکار، مقصد (Nonrepudiation, Destination)
اثبات اینکه پیام بتوسط طرف اصلی دریافت شده است.

جنبهٔ دیگر محرمانگی، حفاظت جریان ترافیک در برابر تجزیه و تحلیل دشمن است. لازمهٔ این کار این است که یک مهاجم نتواند منبع، مقصد، تواتر، طول و یا سایر مشخصه‌های ترافیکی یک تسهیلات ارتباطی را مشاهده نماید.

صحت داده‌ها

همانند محرمانگی، کنترل صحت و یا اصالت داده‌ها می‌تواند به جریان دائمی پیام‌ها، به یک پیام منفرد و یا به میدان‌های انتخاب شده از یک پیام اعمال گردد. بازهم مفیدترین و سرراست‌ترین روش، حفاظت از کل جریان داده‌هاست.

یک سرویس صحت اتصال‌گرا، سرویسی که با جریان پیوستهٔ پیام‌ها سروکار دارد، بایستی این اطمینان را ایجاد کندکه پیام‌ها همانطور که ارسال می‌شوند دریافت گردند، بدون اینکه مجدداً تکرار شده، چیزی به آنها اضافه شده، تغییر یافته، نظم آنها بهم خورده و یا بازخوانی شده باشند. تخریب داده‌ها نیز تحت همین سرویس قرار دارد. بنابراین سرویس صحت با گرایش اتصالی هم تغییر داده‌ها و هم انکار سرویس را مورد خطاب قرار می‌دهد. از سوی دیگر یک سرویس صحت غیراتصالی آن است که تنها با پیام‌های منفرد، بدون توجه به محدودهٔ وسیع آنها، سروکار داشته و فقط در برابر تغییر محتویات پیام حفاظت ایجادکند.

می‌توان بین سرویس‌های با بازیابی و بدون بازیابی تمایز قائل شد. چون سرویس صحت مربوط به حملات فعال است، جنبهٔ تشخیص آنها و نه جنبهٔ جلوگیری از آنها دارای اهمیت است. اگر در صحت دیتا خللی مشاهده گردد، سرویس مربوط ممکن است تنها این خلل را گزارش نماید و لازم باشد تا بخش دیگری از نرم‌افزار و یا نوعی دخالت انسانی مشکل را حل کند. از سوی دیگر مکانیسم‌هائی نیز وجود دارند که علاوه بر تشخیص عدم صحت به حل مشکل نیز کمک می‌کنند. قراردادن مکانیسم‌های بازیابی خودکار معمولاً انتخاب‌های برجاذبه‌تری هستند.

عدم انکار

عدم انکار، چه فرستنده و چه گیرنده را از انکار یک پیام ارسال شده مانع می‌شود. بنابراین وقتی پیامی ارسال می‌شود، گیرنده پیام می‌تواند اثبات کند که حتماً همان فرستندهٔ ذکرشده، پیام را ارسال کرده است. بطریق مشابه وقتی پیامی دریافت می‌گردد، فرستندهٔ پیام می‌تواند اثبات کند که حتماً همان گیرندهٔ ذکرشده، پیام را دریافت کرده است.

سرویس قابلیت دسترسی

هم X.800 و هم RFC 2828 قابلیت دسترسی (availability) را خاصیت یک سیستم و یا یک منبع می‌دانند که در صورت تقاضا از سوی یک واحد مجاز و بر اساس مشخصه‌های عملکرد، سیستم و منابع آن آمادهٔ سرویس‌دهی باشند (یعنی یک سیستم وقتی قابل دسترس است که هروقت کاربران بخواهند، بتواند بر اساس طراحی خود به آنان ارائهٔ سرویس نماید). حملات مختلفی می‌توانند باعث کم شدن قابلیت دسترسی شوند. از بعضی از این حمله‌ها می‌توان با چاره‌جوئی‌های خودکار مثل اعتبارسنجی و رمزنگاری جلوگیری کرد در حالیکه بازیابی از برخی دیگر در یک سیستم گسترده، نیاز به نوعی دخالت فیزیکی دارد.

X.800 قابلیت دسترسی را بعنوان خاصیتی مرتبط با سرویس‌های مختلف امنیتی می‌شناسد. با وجود این منطقی است که به دنبال یک سرویس مخصوص قابلیت دسترسی نیز باشیم. این سرویس، نگرانی‌های امنیتی مرتبط با حملات انکار سرویس را مورد توجه قرار می‌دهد. این سرویس اتکاء به مدیریت منظم و کنترل منابع سیستم داشته، و بنابراین وابسته به سرویس کنترل دست‌یابی و سایر سرویس‌های امنیتی است.

۱-۵ مکانیسم‌های امنیتی

جدول ۳-۱ لیست مکانیسم‌های امنیتی تعریف شده در X.800 را نشان می‌دهد. همانطور که می‌توان دید، مکانیسم‌ها به دودسته، یکی آنهائی که در یک لایهٔ پروتکلی خاص قرار دارند و دیگری آنهائی که مختص لایهٔ خاص و یا سرویس امنیتی خاصی نیستند تقسیم می‌شوند. این مکانیسم‌ها در محل مناسب خود در این کتاب مورد بحث قرار خواهند گرفت و بنابراین فعلاً وارد جزئیات آنها نمی‌شویم. فقط در مورد تعریف قابلیت رمزنگاری، به نکته‌ای اشاره می‌کنیم. X.800 بین مکانیسم‌های رمزنگاری برگشت‌پذیر و مکانیسم‌های رمزنگاری برگشت‌ناپذیر تفاوت قائل است. یک مکانیسم رمزنگاری برگشت‌پذیر بسادگی یک الگوریتم رمزنگاری است که اجازه می‌دهد تا داده‌ها به رمز درآمده و متعاقباً از رمز خارج شوند. مکانیسم‌های رمزنگاری برگشت‌ناپذیر شامل الگوریتم‌های درهم‌سازی و کُدهای اعتبارسنجی پیام بوده که در کاربردهای امضاء دیجیتال و اعتبارسنجی پیام بکار می‌روند.

جدول ۴-۱ که بر اساس X.800 بنا شده است رابطهٔ بین سرویس‌های امنیتی و مکانیسم‌های امنیتی را نشان می‌دهد.

جدول ۳-۱ مکانیسم‌های امنیتی (X.800)

مکانیسم‌های امنیتی مخصوص	مکانیسم‌های امنیتی فراگیر
ممکن است در لایه پروتکلی مناسب قرار داده شود تا بعضی از سرویس‌های امنیتی OSI را فراهم نماید. **رمزنگاری (Encipherment)** استفاده از الگوریتم‌های ریاضی، تا داده‌ها را به شکل غیرقابل فهمی درآورد. تبدیل دیتا و بازیابی آیندهٔ آن بستگی به الگوریتم و احتمالاً یک یا چند کلید رمز دارد. **امضاء دیجیتال (Digital Signature)** دیتای وصل شده به/ یا تبدیل رمزنگاری شدهٔ یک واحد دیتا که به یک دریافت‌کنندهٔ واحد دیتا اجازه می‌دهد تا منبع دیتا و صحت دیتا را اثبات کرده و از تقلب جلوگیری نماید. **کنترل دست‌یابی (Access Control)** مکانیسم‌های متنوعی که حق دست‌یابی به منابع را قانون‌مند می‌سازند. **صحت دیتا (Data Integrity)** مکانیسم‌های متنوعی که از آنها برای اطمینان از صحت یک واحد دیتا و یا دنباله‌ای از واحدهای دیتا استفاده می‌شود. **مبادلهٔ اعتبارسنجی (Authentication Exchange)** مکانیسمی با هدف اطمینان یافتن از هویت یک واحد از طریق مبادلهٔ اطلاعات. **لایه‌لا کردن ترافیک (Traffic Padding)** واردکردن بیت‌ها در شکاف‌های دیتا به منظور خنثی کردن تلاش‌های تحلیل ترافیک.	**کنترل مسیریابی (Routing Control)** انتخاب مسیرهای فیزیکی امن برای بعضی داده‌ها را امکان‌پذیر کرده و اجازه می‌دهد تا در صورت بروز یک تهدید امنیتی، مسیر تعویض شود. **ثبت سند (Notarization)** استفاده از یک طرف ثالث معتمد برای اطمینان یافتن از خصوصیات یک مبادلهٔ دیتا. مکانیسم‌هائی که به سرویس امنیتی و یا یک لایهٔ پروتکلی خاص OSI وابسته نیستند. **عملکرد مطمئن (Trusted Functionality)** اینکه دیتا موافق با شرایط خاصی صحیح باشد (مثلاً بر اساس یک خط‌مشی امنیتی) **برچسب امنیتی (Security Label)** نشانه‌ای که به یک منبع (که ممکن است یک واحد دیتا باشد) وصل می‌گردد تا مشخصه‌های امنیتی آن منبع را نشان دهد. **تشخیص وقایع (Event Detection)** تشخیص پیشامدهای مرتبط با امنیت. **ردپای ممیزی امنیتی (Security Audit Trail)** دیتای جمع‌آوری‌شده که بطرز مؤثری برای تسهیل یک ممیزی امنیتی بکار رود. مروری مستقل بر سوابق سیستم و فعالیت‌های آن است. **بازیابی امنیتی (Security Recovery)** مربوط به درخواست از مکانیسم‌ها، همانند رتق و فتق وقایع و عملیات مدیریتی بوده که به بازیابی منتهی شود.

جدول ۴-۱ رابطهٔ بین سرویس‌های امنیتی و مکانیسم‌های امنیتی

سرویس	رمزنگاری	امضاء دیجیتال	کنترل دست‌یابی	صحت داده‌ها	مبادلهٔ اعتبارسنجی	لایه‌لائی ترافیک	کنترل مسیریابی	ثبت‌سند
اعتبارسنجی واحد نظیر	بلی	بلی			بلی			
اعتبارسنجی منبع دیتا	بلی	بلی						
کنترل دست‌یابی			بلی					
محرمانگی	بلی						بلی	
محرمانگی جریان ترافیک	بلی					بلی	بلی	
صحت داده‌ها	بلی	بلی		بلی				
عدم انکار		بلی		بلی				بلی
قابلیت دسترسی				بلی	بلی			

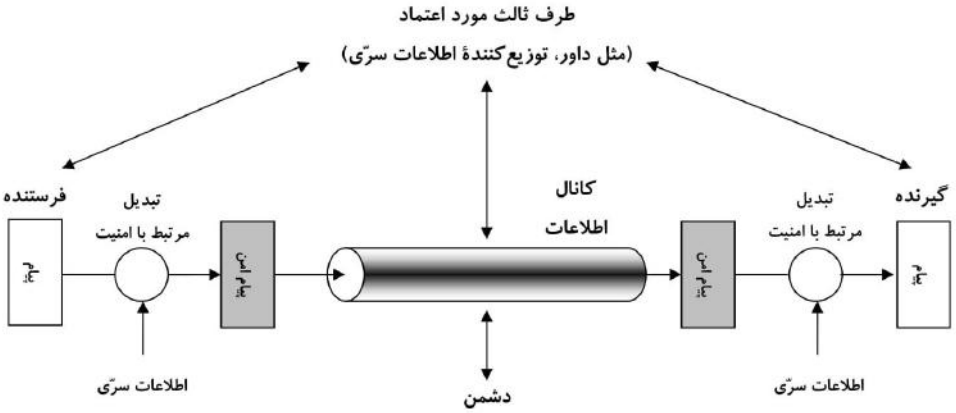
۱-۶ یک مدل برای امنیت شبکه

یک مدل برای بیشتر مطالبی که مورد بحث قرار خواهد گرفت، در حالت بسیار کلی، در شکل ۵-۱ نشان داده شده است. قرار است یک پیام، از یک طرف مکالمه به طرف دیگر، در عرض نوعی اینترنت ارسال گردد. دو طرف مکالمه که *رؤسای* (*principals*) این ارتباط هستند بایستی با یکدیگر همکاری نموده تا این انتقال صورت پذیرد. با تعریف یک مسیر ارتباطی از درون شبکه‌ها، که مبدأ را به مقصد متصل می‌کند، و همکاری در استفاده از پروتکل‌های ارتباطی (مثل TCP/IP)، یک کانال اطلاعاتی منطقی بین دو رئیس ارتباط برقرار می‌شود.

جنبه‌های امنیتی وقتی وارد قضیه می‌گردند که لازم باشد تا انتقال اطلاعات را از یک دشمن که ممکن است تهدیدی برای محرمانگی، اعتبار و غیره ایجاد کند، محافظت کرد. تمام تکنیک‌هائی که امنیت را فراهم می‌سازند دارای دو مؤلفه هستند:

- یک تبدیل مرتبط با امنیت، روی پیامی که قرار است ارسال شود صورت می‌پذیرد. مثال‌های این مورد شامل رمزنگاری پیام است که طوری پیام را درهم می‌ریزد تا قابل خواندن بتوسط دشمن نباشد و همچنین اضافه کردن یک کُد مبتنی بر محتویات پیام که می‌تواند برای تأئید هویت ارسال‌کننده بکار رود.
- نوعی اطلاعات سرّی که بین دو رئیس ارتباط مشترک است و امید می‌رود تا برای دشمن ناشناخته باشد. مثالی در این زمینه یک کلید رمزنگاری است که به‌همراه تبدیل برای درهم ریختن پیام قبل از ارسال، و اصلاح پیام پس از دریافت بکار می‌رود.

یک طرف ثالث قابل اعتماد نیز ممکن است برای انجام انتقال امن اطلاعات مورد نیاز باشد. برای مثال طرف ثالث ممکن است مسئولیت توزیع اطلاعات سرّی به دو رئیس و پنهان نگاه داشتن آن از دید دشمن را بعهده داشته باشد. یا طرف ثالث ممکن است برای داوری اختلافاتی که ممکن است بین دو رئیس در مورد اعتبار یک پیام انتقال یافته رخ دهد مورد نیاز باشد.



شکل ۵-۱ مدل امنیت شبکه

مدل کلّی نشان می‌دهد که در طراحی یک سرویس امنیتی خاص، چهار وظیفهٔ اصلی وجود دارد:

۱- یک الگوریتم برای انجام تبدیل مرتبط با امنیت پیام بایستی طراحی شود. الگوریتم باید چنان باشد که یک دشمن نتواند هدف آن را شکست دهد.
۲- اطلاعات سرّی لازم بتوسط الگوریتم تولید شود.
۳- روش‌هائی برای توزیع و به اشتراک گذاشتن اطلاعات سرّی تعیین گردد.
۴- پروتکلی تعیین شود که دو رئیس از آن استفاده کرده و با بهره‌برداری از الگوریتم امنیتی و اطلاعات سرّی، سرویس امنیتی خاصی را برای طرفین فراهم آورد.

قسمت دوم این کتاب روی مکانیسم‌های امنیتی و سرویس‌هائی که در مدل شکل ۵-۱ قرار دارند متمرکز است. ولی مقوله‌های دیگری نیز در این کتاب مورد بحث قرارگرفته که مرتبط با امنیت بوده ولی دقیقاً در این مدل نمی‌گنجند. یک نمونه از این موارد در شکل ۶-۱ نشان داده شده است که نگرانی‌های مربوط به حفاظت یک سیستم اطلاعاتی از دست‌یابی ناخواسته را نشان می‌دهد. بیشتر خوانندگان با نگرانی‌های ناشی از تلاش هَکرها برای نفوذ به سیستم‌های یک شبکه آشنائی دارند. یک هَکر می‌تواند فردی باشد که بدون نیت سوء، از شکستن دیوارهای امنیتی و ورود غیرمجاز به یک سیستم رایانه‌ای لذت ببرد. یک مهاجم ممکن است یک کارمند ناراضی بوده که بخواهد به سیستم صدمه زده و یا مجرمی باشد که بخواهد از تسهیلات رایانه‌ای برای سوء استفاده مالی استفاده کند (مثل بدست آوردن شمارهٔ کارت‌های اعتباری یا انجام نقل و انتقال غیرقانونی پول).

نوع دیگری از دست‌یابی نامطلوب، قراردادن نوعی منطق در سیستم رایانه‌ای برای بهره‌گیری از نقاط آسیب‌پذیر سیستم بوده که می‌تواند برنامه‌های کاربردی و همچنین برنامه‌های سیستمی مانند ویرایش‌گرها و کامپایلرها را تحت تأثیر قرار دهد. برنامه‌های موذی می‌توانند دو نوع تهدید را بوجود آورند:

- **تهدیدهای دست‌یابی به اطلاعات** که داده‌ها را به نمایندگی از طرف کاربرهائی که نبایستی دست‌یابی به آنها داشته باشند دزدیده و یا تغییر می‌دهند.
- **تهدیدهای سرویس** که از نواقص سرویس‌ها در رایانه‌ها سوء استفاده کرده و مانع استفادهٔ کاربران قانونی از آنها می‌شوند.

شکل ۶-۱ مدل امنیتی دست‌یابی به شبکه

ویروس‌ها و کِرم‌ها دو مثال از حملات نرم‌افزاری می‌باشند. چنین حملاتی می‌تواند از طریق یک دیسکت علیه رایانه صورت پذیرفته و شامل منطقی نامطلوب باشد که در پوشش یک نرم‌افزار مفید پنهان شده است. این حمله‌ها همچنین می‌توانند از طریق شبکه علیه سیستم انجام شوند که مورد اخیر بیشتر در مقولهٔ امنیت شبکه قرار دارد.

مکانیسم‌های امنیتی که بایستی با دست‌یابی‌های ناخواسته مقابله نمایند به دو دستهٔ بزرگ تقسیم می‌شوند (شکل۶-۱). دستهٔ اول همانند یک دروازه‌بان عمل می‌کنند. اینها شامل روش‌های ورود به سیستم بر اساس استفاده از کلمهٔ عبور و روش‌های بازرسی می‌باشند که برای تشخیص و حذف کِرم‌ها، ویروس‌ها و سایر حملات مشابه بکار می‌روند. ولی اگر یک کاربر ناخواسته و یا نرم‌افزار بداندیش توانست به سیستم دست یابد آنگاه خط دوم دفاعی که شامل موارد متنوع کنترل‌های داخلی اعم از پائیدن فعالیت‌ها و تحلیل اطلاعات ذخیره‌شده می‌باشند وارد عمل شده و تلاش خواهند کرد تا حضور مهاجمین ناخواسته را تشخیص دهند. این مطالب در قسمت سوم این کتاب مورد بحث قرار می‌گیرند

۱-۷ استانداردهای اینترنت و انجمن اینترنت

بسیاری از پروتکل‌هائی که مجموعهٔ پروتکلی TCP/IP را می‌سازند، استاندارد شده و یا در شرف استانداردشدن هستند. با موافقت جهانی، سازمانی بنام انجمن اینترنت (Internet Society) مسئول ایجاد و انتشار این استانداردهاست. انجمن اینترنت یک سازمان حرفه‌ای است که بر نیروهای وسیعی که درگیر کارهای اینترنتی و استانداردسازی هستند، نظارت می‌کند.

این بخش توصیف مختصری از روش‌هائی که برای ایجاد استاندارد مجموعهٔ پروتکلی TCP/IP بکار می‌رود را فراهم می‌سازد.

سازمان‌های اینترنت و انتشارات RFC

انجمن اینترنت کمیتهٔ هماهنگ‌کنندهٔ طراحی، مهندسی، و مدیریت اینترنت است. محدودهٔ پوششی آن، عملیات خود اینترنت و استانداردکردن پروتکل‌هائی است که بتوسط سیستم‌های انتهائی بکار می‌روند. سه سازمان تحت مدیریت انجمن اینترنت، مسئول واقعی کار استانداردها و انتشارات می‌باشند:

- **گروه معماری اینترنت Internet Architecture Board (IAB)**: مسئول تعریف معماری کلی اینترنت بوده و راهنمائی و جهت فعالیت IETF را فراهم می‌آورد.
- **نیروی مهندسی اینترنت Internet Engineering Task Force (IETF)**: بازوی توسعه و مهندسی اینترنت.
- **گروه راهبری مهندسی اینترنت Internet Engineering Steering Group (IESG)**: مسئول مدیریت فنی فعالیت‌های IETF و پیاده‌سازی استانداردهای اینترنت.

گروه‌های کاری که بتوسط IETF بسیج می‌شوند، توسعه واقعی استانداردهای جدید و پروتکل‌های اینترنت را بعهده دارند. عضویت در یک گروه، کاری داوطلبانه است و هر گروه علاقه‌مند می‌تواند در آن شرکت نماید. در جریان تهیهٔ یک مشخصه، یک گروه کاری یک پیش‌نویس از اسناد موجود بعنوان یک پیش‌نویس اینترنت تهیه کرده و آن را در فهرست "Internet Draft" بطور مستقیم روی خط قرار می‌دهد. این سند ممکن است تا شش ماه در محل ذکرشده قرار داشته تا گروه‌های علاقه‌مند بتوانند آن را مطالعه کرده و نظرات خود را ابراز دارند. در خلال این مدت، IESG ممکن است انتشار این سند را بعنوان یکRFC (Request for Comment) تصویب کند. اگر این پیش‌نویس ظرف یک دورهٔ شش ماهه، فرم

RFC بخود نگرفت از فهرست خارج خواهد شد. گروه کاری در پی آن ممکن است یک نسخهٔ اصلاح و دستکاری شدهٔ آن را انتشار دهد.

IETF مسئول انتشار RFCها با تصویب IESG است. RFCها یادداشت‌های کاری کمیتهٔ توسعه و مهندسی اینترنت است. یک سند در این سری ممکن است هر موضوعی در رابطه با ارتباطات رایانه‌ای بوده و می‌تواند هرچیزی از گزارش یک ملاقات تا مشخصات یک استاندارد باشد.

کار IETF به هشت شعبه تقسیم شده که هرشعبه دارای یک مدیر بوده و خود شامل گروه‌های کاری بسیار است. جدول ۵-۱ شعبات IETF و وظایف آنها را نشان می‌دهد.

رَوَند استانداردسازی

تصمیم‌گیری راجع به اینکه کدام RFC یک استاندارد اینترنت شود، بتوسط IESG و بر اساس توصیهٔ IETF صورت می‌پذیرد. برای اینکه یک مشخصه بصورت استاندارد درآید، بایستی دارای شرایط زیر باشد:

- پایدار بوده و خوب درک شده باشد.
- از نظر تکنیکی، رقیب تکنیک‌های دیگر باشد.
- دارای صور پیاده‌سازی متعدد، مستقل و متعامل با تجارب عملیاتی قابل توجه باشد.
- از حمایت عمومی چشم‌گیری برخوردار باشد.
- بطور قابل ملاحظه‌ای در بخشی و یا تمام بخش‌های اینترنت مفید باشد.

جدول ۵-۱ عرصه‌های IETF

عرصهٔ IETF	موضوع	نمونهٔ گروه‌های کاری
عمومی	روش‌های کاری IETF	Policy Framework Process for Organization of Internet Standards
کاربردها	کاربردهای اینترنت	Web- related protocols(HTTP) EDI- Internet integration LDAP
اینترنت	زیرساخت اینترنت	IPv6 PPP extensions
عملیات و مدیریت	استانداردها و تعاریف مرتبط با عملیات شبکه	SNMPv3 Remote Network Monitoring
مسیریابی	پروتکل‌ها و مدیریت اطلاعات مسیریابی	Multicast routing OSPF QoS routing
امنیت	پروتکل‌های امنیتی و فن‌آوری‌ها	Kerberos IPsec X.509 S/MIME TLS
حمل و نقل	پروتکل‌های لایه حمل‌ونقل	Differentiated services IP telephony NFS RSVP
سرویس‌های کاربران	روش‌های ارتقاء کیفیت اطلاعات برای کاربران اینترنت	Responsible Use of the Internet User Services FYI documents

اختلاف کلیدی بین این شرایط و آنهائی که از طرف ITU بصورت استاندارد بین‌المللی مطرح می‌شوند این است که در اینجا، تأکید بر تجارب عملیاتی است.

سمت چپ شکل ۷-۱ قدم‌هائی را نشان می‌دهد که *مسیر استاندارد* نامیده می‌شوند و یک مشخصه بایستی آنها را پیموده تا بصورت یک استاندارد درآید. این تحول در RFC 2026 تعریف شده است. این قدم‌ها شامل میزان فزاینده‌ای از بازرسی‌های دقیق و آزمایش‌های متنوع است. در هر قدم، IETF بایستی توصیه‌هائی برای رشد پروتکل را فراهم آورد و IESG بایستی آن را تصویب کند. عمل وقتی آغاز می‌شود که IESG نسخهٔ منتشرشدهٔ پیش‌نویس سند را بعنوان یک RFC پیشنهادشده برای استاندارد بپذیرد.

خانه‌های سفید در شکل ۷-۱ نمایش حالات موقت بوده که بایستی برای حداقل زمان ممکن اشغال شوند. با وجود این یک سند بایستی حداقل شش ماه بصورت یک استاندارد پیشنهادشده، و حداقل چهارماه بصورت یک پیش‌نویس استاندارد، در حالت انتظار قرار داشته باشد تا زمان کافی برای تجدیدنظر و پیشنهادها موجود باشد. خانه‌های خاکستری نمایش‌گر حالات طولانی بوده که ممکن است سال‌ها طول بکشند.

برای اینکه یک مشخصه به حالت پیش‌نویس استاندارد ارتقاء یابد، حداقل بایستی دو پیاده‌سازی مستقل و متعامل از آن با تجربیات عملی کسب شده وجود داشته باشد.

پس از اینکه مشخصه بصورت قابل توجهی پیاده‌سازی شده و تجربیات عملی از آن بدست آمده باشد، آنگاه آن مشخصه ممکن است به سطح یک استاندارد اینترنت ارتقاء یابد. در این مرحله، به مشخصه یک شمارهٔ STD و همچنین یک شمارهٔ RFC داده می‌شود.

بالاخره زمانی که یک پروتکل دیگر به درد نخورد، حالت تاریخی به آن نسبت داده می‌شود.

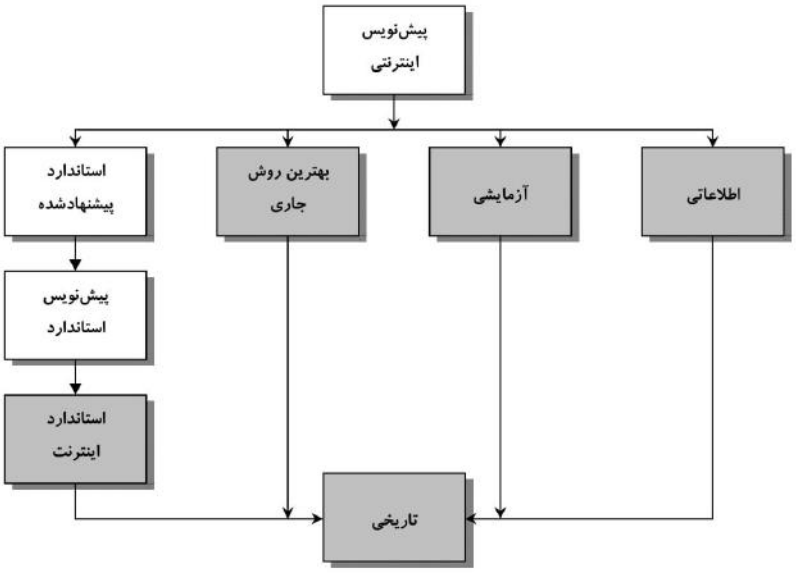
دسته‌بندی استانداردهای اینترنتی

تمام استانداردهای اینترنت در یکی از دو گروه زیر قرار دارند:

- **مشخصات فنی (Technical specification (TS)**: یک TS، یک پروتکل، سرویس، روش، قرارداد و یا فرمت را تعریف می‌کند. بیشتر استانداردها از نوع TS هستند.
- **گزارش قابلیت عملیاتی (Applicability statement (AS)**: یک AS مشخص می‌سازد که چگونه و تحت چه شرایطی یکی و یا بیش از یکی از TSها می‌توانند برای حمایت از یک قابلیت اینترنتی بکار گرفته شوند. یک AS، یک یا چند TS که مرتبط با یک قابلیت بوده را معرفی کرده و ممکن است مقادیر و یا محدودهٔ پارامترهای مخصوصی را که مرتبط با یک TS و یا زیرمجموعهٔ عملیاتی یک TS باشند را مشخص نماید.

سایر انواع RFC

RFCهای بسیاری وجود دارند که جهت آنها این نیست که بصورت یک استاندارد اینترنت درآیند. بعضی RFCها نتیجهٔ فعالیت های موشکافانهٔ انجمن‌ها در مورد بیان یک اصل و یا نتایج این که بهترین راه حل برای انجام برخی عملیات یا وظیفهٔ یک IETF کدام است را استاندارد می‌کنند. این RFCها را بهترین روش جاری Best Current Practice (BCP) می‌نامند. تصویب BCPها تقریباً همان مسیر تصویب استانداردهای پیشنهاد شده را طی می‌کند. برخلاف اسنادی که روی خط استاندارد قرار دارند، برای BCPها یک دورهٔ سه مرحله‌ای وجود ندارد. یک BCP، حالت پیش‌نویس اینترنت تا BCP تصویب شده را در یک قدم طی می‌کند.



شکل ۷-۱ رَوَند انتشار یک RFC اینترنت

یک پروتکل یا مشخصهٔ دیگری که آمادهٔ استاندارد شدن تشخیص داده نمی‌شود، ممکن است بصورت یک RFC آزمایشی منتشر گردد. پس از کار بیشتری، مشخصه ممکن است مجدداً ارائه گردد. معمولاً اگر مشخصه پایدار بوده، اهداف طراحی مشخصی را برآورده کرده، درک خوبی از آن حاصل شده، بازنگری‌های قابل توجهی در آن بوجود آمده و بنظر برسد که ارزش قابل ملاحظه‌ای دارد، آنگاه آن RFC بصورت یک استاندارد پیشنهاد شده درمی‌آید.

نهایتاً یک مشخصهٔ اطلاعاتی (Informational Specification) برای اطلاع انجمن منتشر می‌شود.

۱-۸ ساختار این کتاب

این فصل وظیفهٔ معرفی تمام کتاب را بعهده دارد. بقیهٔ کتاب در سه قسمت سازمان داده شده است:

- **قسمت اول**: مرور مختصری بر الگوریتم‌های رمزنگاری و پروتکل‌های زیرساخت کاربردهای امنیت شبکه دارد که شامل رمزنگاری، توابع درهم‌ساز، امضاءهای دیجیتال و مبادلهٔ کلید هستند.
- **قسمت دوم**: استفاده از الگوریتم‌های رمزنگاری و پروتکل‌های امنیتی، برای فراهم‌آوردن امنیت در شبکه‌ها و اینترنت را بررسی می‌کند. عناوینی همچون اعتبارسنجی کاربر، امنیتe-mail، امنیت IP و امنیت WEB در این فصل گنجانده شده‌اند.

- **قسمت سوم**: مربوط به تسهیلات امنیتی طراحی شده برای حفاظت سیستم‌های رایانه‌ای، در برابر تهدیدهای امنیتی مانند مهاجمین، ویروس‌ها و کِرم‌هاست. دراین قسمت به تکنولوژی دیوارآتش نیز پرداخته می‌شود.

بسیاری از الگوریتم‌های رمزنگاری، پروتکل‌ها و کاربردهای امنیت شبکه که در این کتاب مورد توصیف قرار گرفته‌اند بصورت استاندارد درآمده‌اند. مهم‌ترین آنها، استانداردهای اینترنت‌که در RFCهای اینترنت تعریف شده، و استانداردهای فدرال پردازش اطلاعات (FIPS) که به توسط سازمان ملی استانداردها و تکنولوژی آمریکا(NIST) منتشر می‌شوند. می باشند.

۱-۹ منابع مطالعاتی

[PFLE02] امنیت رایانه و امنیت شبکه را بخوبی معرفی کرده است. دو بررسی فوق‌العادهٔ دیگر را می‌توان در [PIEP03] و [BISH05] جستجو کرد. [BISH03] تقریباً همان مطالب [BISH05] را با جزئیات ریاضی بیشتر و قوی‌تری پوشش داده است. [SCHN00] یک منبع خواندنی ارزنده برای هرکسی است که در زمینهٔ امنیت رایانه و امنیت شبکه فعالیت می‌کند. این کتاب محدودیت‌های تکنولوژی و علی‌الخصوص رمزنگاری در فراهم‌آوردن امنیت را بررسی کرده و نیاز توجه به سخت‌افزار، پیاده‌سازی‌های نرم‌افزاری، شبکه‌ها و مردمی که در ایجاد امنیت و اخلال در امنیت مشارکت دارند را گوشزد می‌کند.

BISH03 Bishop, M. *Computer Security: Art and Science*. Boston : Addison-Wesley, 2003.
BISH05 Bishop, M. *Introduction to Computer Security*. Boston : Addison-Weseley, 2005.
PFLE02 Pfleeger, C. *Security in Computing*. Upper Saddle River, NJ: Prentice Hall, 2002.
PIEP03 Pieprzyk, J.; Hardjono, T.; and Seberry, J. *Fundamentals of Computer Security*. New York: Springer-Verlag, 2003.
SCHN00 Schneier, B. *Secrets and Lies: Digital Security in a Networked World*. New York: Wiley 2000.

۱-۱۰ منابع اینترنت و وِب

منابع متعددی در اینترنت پشتیبان این کتاب بوده و به فرد کمک می‌کنند تا خود را با پیشرفت‌های این حوزه هم‌گام سازد.

وِب‌سایت‌های این کتاب

یک صفحهٔ وِب بطور اختصاصی برای این کتاب در آدرس زیر تهیه شده است:

WilliamStallings.com/NetSec/NetSec3e.html

سایت شامل مطالب زیر است:

- **سایت‌های مفید در وِب**: لینک‌هائی به سایت‌های دیگر مرتبط با مطلب، بر اساس فصول کتاب، فراهم شده که شامل سایت‌های این بخش و فصول دیگر است.
- **غلط‌نامه**: یک لیست از غلط‌های کتاب در اینجا نگهداری و مرتباً به روز می‌شود. لطفاً به هر اشتباهی برخورد می‌کنید، آن را برای نگارندهٔ کتاب e-mail کنید.
- **شکل‌ها**: تمام شکل‌های این کتاب با فرمت PDF.
- **جدول‌ها**: تمام جدول‌های این کتاب با فرمت PDF.
- **اسلایدها**: مجموعه‌ای از اسلایدها بصورت Power Point برای هرفصل.
- **لیست پستی اینترنت**: سایت شامل اطلاعات لازم برای ثبت نام در لیست پستی این کتاب است.
- **دوره‌های امنیت شبکه**: لینک‌هائی به دوره‌هائی که بر اساس این کتاب تدریس می‌شود وجود دارد که می‌تواند برای سایر مدرسین جهت معماری درس مفید باشد.

اینجانب همچنین سایت Computer Science Student Resource Site را در آدرس ذیل نگاه داشته‌ام

WilliamStalling.com/StudentSupport.html

هدف این سایت فراهم‌آوردن اسناد، اطلاعات، و لینک‌هائی برای دانشجویان کامپیوتر و افراد حرفه‌ای است. لینک‌ها و اسناد در چهار گروه طبقه‌بندی شده‌اند:

- **ریاضی**: شامل قسمت یادآوری ریاضیات، تئوری مقدماتی صف، بحث مقدماتی سیستم‌های اعداد، و لینک‌های متعددی به سایر سایت‌های ریاضی است.
- **چگونه**: هدایت و راهنمائی برای حل تکالیف، نوشتن گزارش‌های فنی و آماده‌سازی برای ارائهٔ مطلب.
- **منابع تحقیق**: لینک‌هائی به مجموعهٔ مقالات، گزارشات فنی و فهرست‌هاست.
- **متفرقه**: موارد متنوع دیگری از اسناد و لینک‌ها.

سایر وِب‌سایت‌ها

سایت‌های متعددی وجود دارند که در مورد عناوین مطرح شده در این کتاب ارائهٔ اطلاعات می‌نمایند. در فصول آینده، درهرفصل سایت‌های مرتبط با مطلب را در بخش *منابع مطالعاتی* معرفی خواهیم کرد. نظر به این‌که آدرس سایت‌ها مکرراً تغییر می‌کنند، آنها را در این کتاب نیاورده‌ایم. برای تمام سایت‌هائی که در این کتاب لیست شده است، لینک مرتبطی را می‌توان در سایت این کتاب پیدا کرد. سایر لینک‌هائی که در این کتاب ذکر نشده است در طول زمان به سایت اضافه خواهند شد.

سایت‌های زیر در رابطه با رمزنگاری و امنیت شبکه قابل توجه‌اند:

- **COAST**: مجموعه کاملی از لینک‌های مرتبط با رمزنگاری و امنیت شبکه.
- **IETF Security Area**: مطالب مربوط به تلاش‌های استانداردسازی امنیت اینترنت.
- **Computer and Network Security Reference Index**: یک مرجع خوب از سازندگان و محصولات تجاری، سؤالاتی که مکرراً پرسیده می‌شوند(FAQ)، آرشیو گروه‌های خبری، مقاله‌ها و لیست سایر سایت‌ها.
- **The Cryptography FAQ**: سؤالات مفصل و ارزشمند در تمام زمینه‌های رمزنگاری به همراه پاسخ آنها.
- **Tom Dunigan's Security Page**: یک لیست فوق‌العاده از سایت‌های مرجع در مورد رمزنگاری و امنیت شبکه.
- **IEEE Technical Committee on Security and Privacy**: کپی خبرنامه‌ها، و اطلاعات و فعالیتهای مرتبط با IEEE.
- **Computer Security Resource Center**: مربوط به سازمان ملی استانداردها و تکنولوژی آمریکا (NIST) که شامل اطلاعات وسیعی در مورد تهدیدهای امنیتی، فن‌آوری و استانداردهاست.
- **Security Focus**: اطلاعات متنوعی در بارهٔ امنیت، با تأکید خاصی بر محصولات فروشندگان و نیازهای کاربران انتهائی.
- **SANS Institute**: مشابه Security Focus است. مجموعهٔ وسیعی از مقالات را دربر دارد.
- **Data Protection Resource Directory**: مجموعهٔ گوناگونی از لینک‌ها.

گروه‌های خبری USENET

تعدادی از گروه‌های خبری USENET، به بعضی زمینه‌های امنیت شبکه یا رمزنگاری اختصاص دارند. تقریباً همانند تمام گروه‌های خبری USENET، مطالب سازمان یافته نبوده و پراکنده‌اند ولی ممکن است با جستجو در لابلای آنها نکات مفیدی را بدست آورد. مرتبط‌ترین آنها بقرار زیراند:

- **sci.cryp.research**: بهترین گروهی است که میتوان دنبال کرد. عمدتاً عناوین مقالاتی را معرفی می‌کند که به جنبه‌های فنی رمزنگاری ارتباط دارند.
- **sci.crypt**: مباحث عام رمزنگاری و عناوین مرتبط با آن.
- **sci.crypt.random-numbers**: بحث در مورد تصادفی بودن توان رمزنگاری.
- **alt.security**: یک بحث کلی از عناوین رمزنگاری.
- **comp.security.misc**: مباحث عام امنیت رایانه.
- **comp.security.firewalls**: بحث در مورد محصولات دیوارهای آتش و فن‌آوری آنها.
- **comp.security.announce**: اخبار، اطلاعیه‌های CERT.
- **comp.risks**: بحث در مورد خطراتی که از جانب رایانه‌ها و کاربران متوجه جامعه است.
- **comp.virus**: بحث ساده شده‌ای در مورد ویروس‌های رایانه‌ای.

۱-۱۱ واژه‌های کلیدی، سؤالات مرورکنندهٔ بحث و مسائل

واژه‌های کلیدی

access control	کنترل دست‌یابی	intruder	مهاجم
active threat	تهدید فعال	masquerader	نقاب‌دار
authentication	اعتبارسنجی	nonrepudiation	عدم انکار
authenticity	اعتبار	OSI security architecture	معماری امنیت OSI
availability	قابلیت دسترسی	passive threat	تهدید غیرفعال
data confidentiality	محرمانگی داده‌ها	replay	بازخوانی
data integrity	صحت داده‌ها	security attacks	حملات امنیتی
denial of service	انکار سرویس	security mechanisms	مکانیسم‌های امنیتی
encryption	رمزنگاری	security services	سرویس‌های امنیتی
integrity	صحت، یکپارچگی	traffic analysis	تحلیل ترافیک

سؤالات مرورکنندهٔ بحث

۱-۱ معماری امنیت OSI چیست؟

۱-۲ تفاوت بین حملات امنیتی فعال و غیرفعال چیست؟

۱-۳ حملات امنیتی فعال و غیرفعال را دسته‌بندی کرده و بطور مختصر تعریف کنید.

۱-۴ سرویس‌های امنیتی را طبقه‌بندی کرده و بطور مختصر تعریف کنید.

۱-۵ مکانیسم‌های امنیتی را طبقه‌بندی کرده و بطور مختصر تعریف کنید.

مسائل

۱-۱ جدولی مشابه با جدول ۱-۴ ترسیم کنید که رابطهٔ بین سرویس‌های امنیتی و حملات را نشان دهد.

۱-۲ جدولی مشابه با جدول ۱-۴ ترسیم کنید که رابطهٔ بین مکانیسم‌های امنیتی و حملات را نشان دهد.

قسمت اول

رمزنگاری

تا این زمان، مهم‌ترین وسیلهٔ خودکار مورد استفاده در امنیت شبکه و امنیت اطلاعات، رمزنگاری است. دو شکل از رمزنگاری مرسوم است: رمزنگاری رسمی یا متقارن و رمزنگاری کلید- عمومی یا نامتقارن. قسمت اول مروری کلی بر اصول اساسی رمزنگاری متقارن و رمزنگاری کلید- عمومی داشته، نگاهی به الگوریتم‌های پراستفادهٔ آنها انداخته و کاربردهای اساسی این دو برخورد را مورد بحث قرار می‌دهد.

فصل ۲ رمزنگاری متقارن و محرمانگی پیام

فصل۲ روی رمزنگاری متقارن تمرکز کرده و تأکیدی بر پراستفاده ترین تکنیک رمزنگاری، یعنی استاندارد رمزنگاری دیتا (DES)، و الگوریتم‌های متعاقب آن مثل 3DES و استاندارد رمزنگاری پیشرفته (AES) دارد. صرف‌نظر از سؤالات مربوط به ساختار یک الگوریتم رمزنگاری متقارن، تعدادی از مسائل طراحی، مرتبط با استفاده از رمزنگاری متقارن به منظور ایجاد محرمانگی هستند. این فصل شامل بحثی در مورد رمزنگاری پیام‌های طولانی، رمزنگاری سر- به- سر در مقابل رمزنگاری پیوند و تکنیک‌های توزیع کلید است.

فصل ۳ رمزنگاری کلید– عمومی و اعتبارسنجی پیام

یکی از مسائلی که در زمینهٔ معیارهای امنیتی به اندازهٔ محرمانگی اهمیت دارد، اعتبارسنجی است. اعتبارسنجی پیام این اطمینان را ایجاد می‌کند که یک پیام از یک منبع قانونی سرچشمه گرفته است. علاوه برآن اعتبارسنجی می‌تواند شامل حفاظت پیام در برابر دستکاری، تأخیر، بازخوانی و یا تغییر نظم نیز باشد. فصل۳ با تحلیلی در مورد لازمه‌های اعتبارسنجی شروع شده و آنگاه نگاهی به روش‌های اعتبارسنجی می‌اندازد. یک عنصر کلیدی روش اعتبارسنجی، استفاده از اعتبارسنج است که معمولاً یا کُد اعتبارسنجی پیام (MAC) بوده و یا یک تابع درهم‌ساز (hash) است. ملاحظات طراحی برای الگوریتم‌های هر دو نوع مورد بررسی قرار گرفته و چندین مثال مشخص تحلیل شده‌اند.

بعد از رمزنگاری متقارن، نوع مطرح دیگر رمزنگاری، رمزنگاری کلید- عمومی است که انقلابی در دنیای امنیت اطلاعات بوجود آورده است. دنبالهٔ فصل سوّم رمزنگاری کلید- عمومی را معرفی می‌کند. الگوریتم RSA مفصلاً مورد بحث قرار گرفته و مسألهٔ مدیریت کلید مجدداً مورد توجه قرار می‌گیرد. این فصل همچنین تکنیک پرکاربرد توزیع کلید Diffie-Hellman را پوشش می‌دهد. علاوه بر این، این فصل امضاء دیجیتال را تعریف کرده و کاربردهای آن را بررسی می‌نماید.

فصل ۲

رمزنگاری متقارن و محرمانگی پیام

رمزنگاری متقارن که از آن با عناوین رمزنگاری رسمی، رمزنگاری کلید- سرّی، و یا رمزنگاری با یک- کلید نیز یاد می‌شود، تنها نوع رمزنگاری مورد استفاده قبل از معرفی رمزنگاری کلید- عمومی دراواخر دههٔ ۱۹۷۰بود. این رمزنگاری در زمان حال نیز پراستفاده‌ترین روش، از بین دو نوع رمزنگاری معمول می‌باشد.

این فصل را با نگاهی به یک مدل عمومی رمزنگاری متقارن شروع می‌کنیم. این امر ما را قادر میسازد تا با محیطی که الگوریتم‌ها درآن مورد استفاده واقع می‌شوند آشنا شویم. سپس به سه الگوریتم مهم رمزنگاری نظر می‌افکنیم که عبارت از DES، 3DES و AES می‌باشند. بعد از آن رمزنگاری دنباله‌ای متقارن را معرفی خواهیم کرد و با رمز دنباله‌ای RC4 که موارد استفادهٔ گسترده‌ای دارد آشنا خواهیم شد. در پایان کاربردهای این الگوریتم‌ها برای ایجاد محرمانگی را بررسی می‌کنیم.

۲-۱ اصول رمزنگاری متقارن

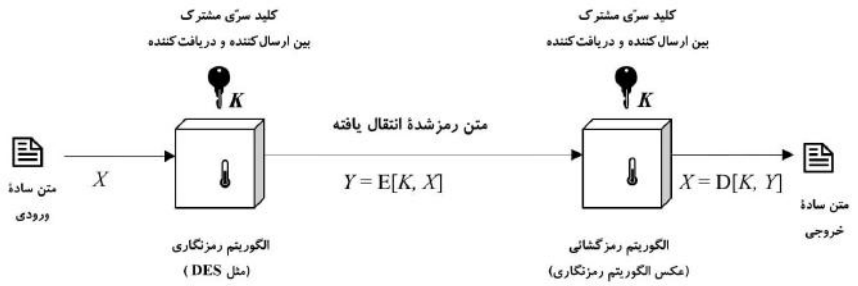
یک طرح رمزنگاری متقارن دارای پنج جزء است (شکل ۲-۱):

- **متن ساده**: این پیام ورودی و یا داده‌هائی است که بعنوان ورودی وارد الگوریتم می‌شود.
- **الگوریتم رمزنگاری**: الگوریتم رمزنگاری، جایگزینی‌ها و تبدیلات مختلفی را روی متن ساده انجام می‌دهد.
- **کلید سرّی**: کلید سرّی نیز یکی از ورودی‌های الگوریتم است. جایگزینی‌ها و تبدیلات انجام‌شده بتوسط الگوریتم، وابسته به این کلید است.
- **متن رمزشده**: این پیام درهم‌ریخته‌شده‌ای است که بعنوان خروجی تولید می‌شود. این متن وابسته به متن ساده و کلید سرّی است. برای یک پیام داده‌شده، دو کلید مختلف دو متن رمزشدهٔ مختلف تولید خواهند کرد.
- **الگوریتم رمزگشائی**: این معمولاً همان الگوریتم رمزنگاری است که بطور معکوس اجرا می‌شود. این الگوریتم متن رمزشده و همان کلید سرّی را گرفته و متن ساده اولیه را تولید می‌کند.

برای استفادهٔ امن از رمزنگاری متقارن دو چیز مورد نیاز است:

۱- به یک الگوریتم رمزنگاری قوی احتیاج داریم. حداقل مایلیم که الگوریتم چنان باشد که دشمنی که الگوریتم را می‌شناسد و به یک یا چند متن رمزشده دسترسی دارد، قادر نباشد تا متن رمزشده را رمزگشائی کرده و یا کلید رمز را کشف کند. این نیاز معمولاً بصورت محکم‌تری چنین بیان می‌گردد: دشمن بایستی قادر نباشد تا متن رمزشده را رمزگشائی کرده و یا کلید رمز را کشف کند، حتی اگر او چند متن رمزشده بهمراه متون سادهٔ نظیر آنها را در اختیار داشته باشد.

۲- فرستنده و گیرنده بایستی کپی‌های کلید سرّی را به روش امنی بدست آورده باشند و آنها را امن نگاه دارند. اگر کسی بتواند کلید را کشف کرده و الگوریتم را نیز بداند، تمام ارتباطاتی که از این کلید استفاده می‌کنند قابل شنود خواهند بود.



شکل ۱-۲ مدل ساده‌شدهٔ رمزنگاری متقارن

مهم است توجه کنیم که امنیت رمزنگاری متقارن بستگی به سرّی بودن کلید، و نه سرّی بودن الگوریتم دارد. یعنی فرض می‌شود که رمزگشائی یک پیام بر مبنای دانستن متن رمزشده *بعلاوهٔ* دانستن الگوریتم رمزنگاری/ رمزگشائی کاری غیرعملی است. بعبارت دیگر، لازم نیست که ما الگوریتم را مخفی نگاه داریم، بلکه فقط کافی است که کلید را مخفی داشته باشیم.

این خصیصهٔ رمزنگاری متقارن همان چیزی است که آن را برای استفادهٔ گسترده مقبول می‌سازد. این واقعیت که الگوریتم لازم نیست تا مخفی بماند، بدین معنی است که سازندگان می‌توانند تراشه‌های ارزان قیمتی که الگوریتم‌های رمزنگاری را عملیاتی می‌سازند تولید نمایند و چنین نیز شده است. این تراشه‌ها در سطح وسیعی در دسترس بوده و در تعدادی محصولات نیز بکار گرفته شده‌اند. در استفاده از رمزنگاری متقارن مسألهٔ اصلی امنیت، حفظ سرّی بودن کلید است.

رمزنگاری

سیستم‌های رمزنگاری معمولاً از سه بُعد مستقل دسته‌بندی می‌شوند:

۱- **نوع عملیات بکار گرفته شده برای تبدیل متن ساده به متن رمزشده:** تمام الگوریتم‌های رمزنگاری بر مبنای دو اصل عمومی قرار دارند: جایگزینی، که در آن هر عنصر متن ساده (بیت، حرف، گروهی از بیت‌ها یا حروف) با عنصر دیگری جایگزین شده، و جابجائی که در آن عناصر متن ساده جای خود را عوض می‌کنند. مهم‌ترین و اصلی‌ترین الزام این است که هیچ اطلاعاتی گم نشود (یعنی تمام عملیات برگشت‌پذیر باشند). بیشتر سیستم‌ها که از آنها با عنوان سیستم‌های ترکیبی یاد می‌شود، شامل چندین مرحلهٔ جایگزینی و جابجائی هستند.

۲- **تعداد کلیدهای استفاده شده:** اگر هم فرستنده و هم گیرنده از یک کلید استفاده کنند، سیستم رمزنگاری را متقارن، تک-کلیدی، کلید- سرّی و یا رسمی گویند. اگر فرستنده و گیرنده هرکدام از یک کلید متفاوت استفاده کنند، سیستم رمزنگاری را نامتقارن، دو- کلیدی و یا کلید- عمومی نامند.

۳- **نحوهٔ پردازش متن سادهٔ پیام:** یک *رمز قالبی*(*block cipher*)، ورودی را بصورت یک بلوک در هر زمان مورد پردازش قرار داده و یک بلوک خروجی برای هر بلوک ورودی تولید می‌کند. یک *رمز دنباله‌ای* (*stream cipher*) عناصر ورودی را بصورت پیوسته پردازش کرده و همینطور که جلو می‌رود، عناصر خروجی نیز بطور پیوسته از آن خارج می‌گردند.

شکستن رمز یا کشف رمز

کوشش برای کشف کردن متن سادهٔ پیام و یا کلید را *شکستن رمز* گویند. استراتژی بکارگرفته‌شده در این مورد، بستگی به طبیعت روش رمزنگاری و اطلاعات قابل دسترسی مسئول کشف رمز دارد.

جدول ۱-۲ انواع مختلف حملات کشف رمز، بر مبنای میزان اطلاعاتی که در اختیار کشف‌کنندهٔ رمز قرار دارد را نشان می‌دهد. مشکل‌ترین مورد زمانی است که *فقط متن رمزشده* در دسترس باشد. در بعضی موارد نه تنها الگوریتم رمزنگاری شناخته شده است، بلکه عموماً می‌توانیم فرض کنیم که دشمن از الگوریتم استفاده شده برای رمزنگاری مطلع است. یکی از حملاتی که تحت این شرایط ممکن است رخ دهد، جستجوی جامع(brute-force) امتحان کردن همهٔ کلیدهای ممکن است. اگر فضای کلید خیلی وسیع باشد، این امر غیرعملی خواهد شد. بنابراین دشمن بایستی به تجزیه و تحلیل خود متن رمزشده متکی بوده و تست‌های آماری مختلفی را روی آن انجام دهد. برای استفاده از این روش، دشمن بایستی ایده‌هائی نسبت به نوع متن ساده پیام که پنهان شده است داشته باشد. یعنی مثلاً بداند که پیام، یک متن انگلیسی، یک متن فرانسه، یک فایل EXE، یک برنامهٔ Java، یک سند مالی و غیره است.

دفاع در برابر حملهٔ فقط- متن رمزشده ساده‌ترین نوع دفاع است زیرا دشمن کمترین اطلاعات را داراست. در بسیاری موارد، شکنندهٔ رمز اطلاعات بیشتری دارد. او ممکن است قادر باشد تا یک یا چند پیام ساده به همراه فرم رمزنگاری‌شدهٔ آنها را بدست آورد. کشف‌کنندهٔ رمز ممکن است بداند که الگوهای مخصوصی در پیام وجود دارد. مثلاً فایلی که با فرمت Postscript کُد می‌شود همیشه با الگوی خاصی شروع می‌شود و یا ممکن است یک عنوان استانداردشده، همیشه همراه یک پیام الکترونیکی انتقال‌دهندهٔ پول وجود داشته باشد. همهٔ اینها مثال‌هائی از *متن سادهٔ معلوم*اند. با چنین معلوماتی، تحلیل‌گر رمز ممکن است بتواند کلید را، بر مبنای آگاهی از الگوی متن سادهٔ رمزنشده، بدست آورد.

جدول ۱-۲ انواع حملات به پیام‌های رمزنگاری شده

نوع حمله	آنچه برای کشف کنندهٔ رمز معلوم است
فقط- متن رمزشده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود
متن سادهٔ معلوم	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • یک یا چندجفت متن ساده- متن رمزشده بتوسط کلید سرّی
متن سادهٔ انتخاب شده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • پیام سادهٔ انتخاب شده بتوسط کشف‌رمزکننده، بهمراه متن رمزشدهٔ نظیر آن بتوسط کلید سرّی
متن رمزشدهٔ انتخاب شده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • متن رمزشدهٔ انتخاب‌شده بتوسط شکنندهٔ رمز، بهمراه متن رمزگشائی شده نظیر آن با استفاده از کلید سرّی
متن انتخاب شده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • پیام سادهٔ انتخاب‌شده بتوسط کشف‌رمزکننده، بهمراه متن رمزشدهٔ نظیرآن با استفاده از کلید سرّی • متن رمزشدهٔ انتخاب‌شده بتوسط کشف‌رمزکننده، بهمراه متن رمزگشائی شده نظیر آن با استفاده از کلید سرّی

موردی که خیلی مرتبط با حملهٔ متن سادهٔ معلوم است، چیزی است که می‌توان از آن با نام حملهٔ کلمهٔ محتمل یاد کرد. اگر دشمن روی کشف رمز یک متن عمومی کار کند، او ممکن است اطلاعات کمی نسبت به محتویات پیام داشته باشد. ولی اگر دشمن بدنبال اطلاعات خیلی تخصصی باشد، آنگاه ممکن است بخشی از پیام را بشناسد. بعنوان مثال اگر یک سند اطلاعات مالی منتقل میشود، دشمن ممکن است از محل برخی کلمات کلیدی در عنوان فایل با خبر باشد. مثال دیگر اینکه کُد اولیهٔ یک برنامهٔ تهیه‌شده در یک سازمان ممکن است شامل یک جملهٔ مربوط به نام سازمان در یک محل مشخص و استاندارد باشد.

اگر تحلیل‌گر بطریقی قادر باشد تا سیستم منبع را وادارد تا پیام انتخاب‌شده‌ای بتوسط تحلیل‌گر را رمزنگاری کند، آنگاه یک حملهٔ *متن سادهٔ انتخاب‌شده* محتمل است. در حالت کلی، اگر تحلیل‌گر بتواند پیام‌هائی را جهت رمزنگاری انتخاب کند، او ممکن است زیرکانه از پیام‌هائی استفاده کند که انتظار می‌رود تا ساختار کلید را آشکار سازند.

جدول ۱-۲ دو نوع حملهٔ دیگر را نیز ذکرکرده است: متن رمزشدهٔ انتخاب‌شده و متن انتخاب‌شده. این حمله‌ها کمتر بعنوان تکنیک‌های کشف رمز بکار می‌روند، ولی با این وجود راه‌های گشوده‌ای برای حمله‌اند.

تنها الگوریتم‌های نسبتاً ضعیف در برابر حملهٔ فقط- متن رمزشده شکست می‌خورند. معمولاً یک الگوریتم رمزنگاری طوری طراحی می‌شود که در مقابل حملهٔ متن سادهٔ معلوم نیز مقاومت کند.

یک روش رمزنگاری در صورتی از **نظر محاسباتی امن** است که متن رمزشده بتوسط آن روش، یک و یا هردو شرط زیر را داشته باشد:

- هزینهٔ شکستن رمز، از ارزش اطلاعات رمزشده تجاوز کند.
- زمان لازم برای شکستن رمز، از عمر مفید اطلاعات تجاوز کند.

متأسفانه بسیار سخت است تا میزان کوشش لازم برای کشف متن رمزشده را تخمین زد. ولی با فرض اینکه هیچ ضعف ذاتی ریاضی در الگوریتم وجود نداشته باشد، آنگاه با تصور یک حملهٔ همه جانبه میتوان تخمین معقولی نسبت به هزینه‌ها و زمان کشف رمز بدست آورد.

روش حملهٔ همه جانبه، شامل امتحان کردن همهٔ کلیدهای ممکن است تا یک ترجمهٔ قابل فهم از متن رمزنگاری شده به متن ساده به دست آید. بطور متوسط، برای موفقیت بایستی نصف کلیدهای ممکن را امتحان کرد. جدول ۲-۲ زمان صرف شده برای کلیدهائی با اندازه‌های مختلف را نشان می‌دهد. در الگوریتم DES از یک کلید ۵۶- بیتی استفاده می‌شود. برای اندازهٔ هر کلید، نتایج با فرض اینکه یک میکروثانیه برای هر رمزگشائی ساده صرف خواهد شد، نشان داده شده است. یک میکروثانیه برای هر رمزگشائی، اندازهٔ معقولی برای ماشین‌های امروزی است. با استفاده از تعداد زیادی میکروپروسسور با سازماندهی موازی، ممکن است نرخ پردازش را به چندین برابر افزایش داد. ستون آخر در جدول ۲-۲ نتایج را برای سیستمی که بتواند یک میلیون کلید در هر میکروثانیه را آزمایش کند، نشان می‌دهد. همانطور که مشاهده می‌کنید، در چنین سطح عملکردی، DES دیگر نمی‌تواند از نظر محاسباتی امن فرض شود.

ساختار رمز Feistel

بیشتر الگوریتم‌های رمزنگاری متقارن قالبی، از جمله DES، دارای ساختاری هستند که برای اولین بار بتوسط Horst Feistel از شرکت IBM در سال ۱۹۷۳ [FEIS73] توصیف گردید و در شکل ۲-۲ نشان داده شده است. ورودی‌های الگوریتم رمزنگاری، یک بلوک از متن ساده با طول $2w$ بیت و یک کلید K است. بلوک متن ساده به دو نیمهٔ L_0 و R_0 تقسیم میشود. دو نیمهٔ دیتا، n دُور پردازش را پشت سر گذاشته و آنگاه با یکدیگر ترکیب شده تا بلوک متن رمزشده را

جدول ۲-۲ زمان متوسط لازم برای امتحان همهٔ کلیدهای رمز

اندازهٔ کلید (bits)	تعداد کلیدهای ممکن	زمان لازم کشف رمز با یک رمزگشائی در هر میکروثانیه	زمان لازم کشف رمز با یک میلیون رمزگشائی در هر میکروثانیه
۳۲	$۲^{۳۲} = ۴٫۳×۱۰^{۹}$	$۲^{۳۱}$ میکروثانیه= ۳۵٫۸ دقیقه	۲٫۱۵ میلی ثانیه
۵۶	$۲^{۵۶} = ۷٫۲×۱۰^{۱۶}$	$۲^{۵۵}$ میکروثانیه= ۱۱۴۲ سال	۱۰٫۰۱ ساعت
۱۲۸	$۲^{۱۲۸} = ۳٫۴×۱۰^{۳۸}$	$۲^{۱۲۷}$ میکروثانیه=$۱۰^{۲۴}$ × ۵٫۴ سال	$۱۰^{۱۸}$ × ۵٫۴ سال
۱۶۸	$۲^{۱۶۸} = ۳٫۷×۱۰^{۵۰}$	$۲^{۱۶۷}$ میکروثانیه=$۱۰^{۳۶}$ ×۵٫۹ سال	$۱۰^{۳۰}$ × ۵٫۹ سال
۲۶ کاراکتر (جایگشت)	$۲۶! = ۴×۱۰^{۲۶}$	$۲×۱۰^{۲۶}$ میکروثانیه=$۱۰^{۱۲}$ × ۶٫۴ سال	$۱۰^{۶}$ × ۶٫۴ سال

ایجاد نمایند. هر دُور i دارای ورودی‌های L_{i-1} و R_{i-1} که خروجی دور ماقبل بوده، و همچنین یک زیرکلید K_i که از کلید K مشتق شده است می‌باشد. عموماً زیرکلیدهای K_i با یکدیگر و همچنین با K فرق داشته و بتوسط یک الگوریتم تولید زیرکلید خلق می‌شوند.

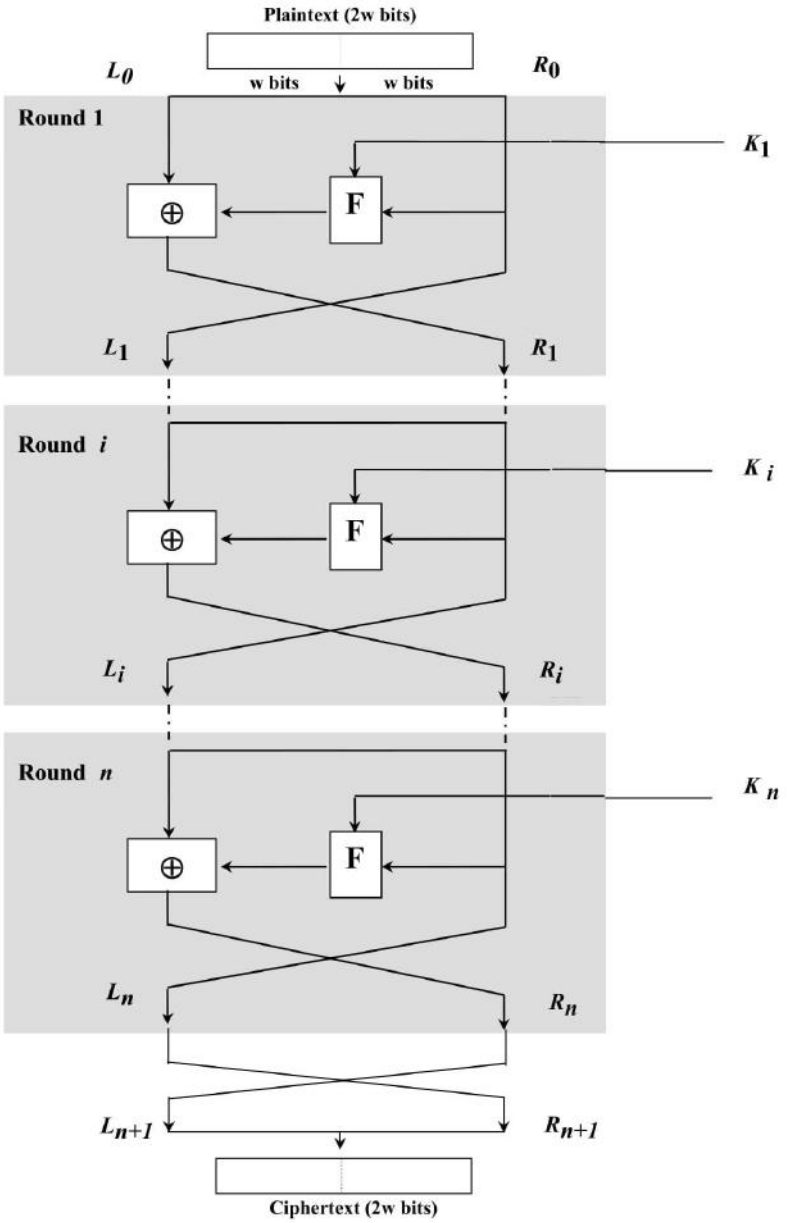
همهٔ دُورهای رمزنگاری دارای ساختار یکسانی هستند. یک جایگزینی روی نیمهٔ چپ دیتا انجام می‌شود. این امر با اعمال یک *تابع دُور*(*round function*) F به نیمهٔ راست دیتا و سپس XOR کردن خروجی این تابع با نیمهٔ چپ دیتا حاصل می‌گردد. در هر دُور، تابع دُور دارای ساختار عمومی یکسانی است که بتوسط زیرکلید دُور K_i پارامترهای آن تغییر می‌یابد. پس از این جایگزینی، یک جایگشت صورت می‌پذیرد که شامل تعویض محل دو نیمهٔ دیتاست.

ساختار Feistel یک مثال خاص از ساختار عمومی‌تری است که بتوسط تمام رمزهای قالبی متقارن مورد استفاده قرار می‌گیرد. در حالت کلی، یک رمز قالبی متقارن شامل تعدادی از دُورهای متوالی است که در هر دُور، عملیات جایگزینی و جابجائی با وابستگی به اندازهٔ کلید سرّی دُور صورت می‌پذیرد. تحقق واقعی یک رمز قالبی متقارن، بستگی به انتخاب پارامترهای زیر و موارد طراحی دارد:

- **اندازهٔ بلوک**: هرچقدر اندازهٔ بلوک‌ها بزرگتر باشد (با فرض ثابت بودن سایر پارامترها)، امنیت بیشتر ولی سرعت رمزنگاری/ رمزگشائی کمتر است. مصالحهٔ مناسب در این مورد، انتخاب بلوکی با طول ۱۲۸ بیت بوده که در طراحی رمز قالبی، تقریباً انتخابی همگانی است.
- **اندازهٔ کلید**: اندازهٔ بزرگتر کلید بمنزلهٔ امنیت بیشتر است، ولی ممکن است سرعت رمزنگاری/ رمزگشائی را کاهش دهد. معمول‌ترین کلیدها در الگوریتم‌های مدرن، دارای طول ۱۲۸ بیت هستند.
- **تعداد دُورها**: جوهرهٔ یک رمز قالبی متقارن در این است که تنها یک دُور رمزنگاری، امنیت مناسبی را ایجاد نمی‌کند و بنابراین دُورهای بیشتری از رمزنگاری برای افزایش امنیت مورد نیاز است. اندازهٔ معمول در این مورد، ۱۶ دُور است.
- **الگوریتم تولید زیرکلید**: پیچیدگی بیشتر در این الگوریتم، بایستی باعث افزایش پیچیدگی در شکستن رمز گردد.
- **تابع دُور**: باز هم پیچیدگی بیشتر، معمولاً بمعنای مقاومت بیشتر در مقابل کشف رمز است.

دو مورد دیگر را نیز در طراحی یک رمز قالبی متقارن بایستی درنظر گرفت:

نرم‌افزار رمزنگاری/ رمزگشائی سریع: در بسیاری موارد، رمزنگاری در دل کاربردها و یا توابع اجرائی طوری قرار گرفته است که خارج از حیطهٔ اجرای سخت‌افزاری الگوریتم است. بنابراین سرعت اجرای الگوریتم یکی از نکته‌های قابل تأمل است.



شکل۲-۲ شبکهٔ کلاسیک Feistel

- **سهولت تحلیل**: اگرچه علاقه‌مندیم که برای جلوگیری از شکستن رمز، هرچه ممکن است الگوریتم را پیچیده‌تر کنیم ولی ایجاد سهولت در تحلیل الگوریتم محسنات زیادی دارد. یعنی اگر الگوریتم بتواند بطور مختصر و روشن بیان شود، تحلیل آن برای آسیب‌پذیری‌های مربوط به شکستن رمز هم ساده‌تر بوده و بنابراین سطح اطمینان به قدرت آن را می‌توان افزایش داد. بعنوان مثال، DES دارای عملکرد تحلیلی ساده‌ای نیست.

رمزگشائی با یک رمز قالبی متقارن نیز ضرورتاً مثل همان رمزنگاری آن است. قاعده چنین است: متن رمزشده را بعنوان ورودی الگوریتم بکار برده ولی زیرکلیدهای K_i را با نظم معکوس استفاده می‌کنیم. یعنی K_n در دور اول، K_{n-1} در دور دوم، و بهمین ترتیب تا K_1 که در دور آخر مورد استفاده قرار می‌گیرد. این خاصیت جذابی است زیرا لازم نیست تا از دو الگوریتم مختلف، یکی برای رمزنگاری و یکی برای رمزگشائی استفاده شود.

۲-۲ الگوریتم‌های رمزنگاری قالبی متقارن

معمول‌ترین الگوریتم‌های بکارگرفته شده برای رمزنگاری متقارن، رمزهای قالبی هستند. یک رمز قالبی، متن سادهٔ ورودی را در قالب بلوک‌هائی با اندازهٔ ثابت پردازش کرده و یک بلوک متن رمزشده با همان اندازه را، برای هر بلوک متن ساده تولید می‌کند. در این بخش سهٔ نوع از مهم‌ترین رمزهای قالبی متقارن را مورد بررسی قرار می‌دهیم: استاندارد رمزنگاری دیتا (DES)، DES سه‌گانه (3DES) و استاندارد رمزنگاری پیشرفته (AES).

استاندارد رمزنگاری دیتا Data Encryption Standard (DES)

پراستفاده‌ترین روش رمزنگاری، بر مبنای استاندارد رمزنگاری دیتا (DES) قرار دارد که درسال ۱۹۷۷ بتوسط دفتر ملی استانداردها در آمریکا که امروز مؤسسهٔ ملی استانداردها و تکنولوژی(NIST) خوانده می‌شود، تحت عنوان استاندارد فدرال پردازش اطلاعات ۴۶ (FIP PUB46) پذیرفته شد. از خود الگوریتم، با نام الگوریتم رمزنگاری دیتا (DEA) یاد می‌شود.

توصیف الگوریتم

متن ساده دارای طول ۶۴ بیت بوده و طول کلید۵۶ بیت است. متون سادهٔ طویل‌تر در بلوک‌های ۶۴- بیتی مورد پردازش قرار می‌گیرند. ساختار DES تقریباً همان ساختار شبکهٔ Feistel با کمی تغییرات است که در شکل ۲-۲ نشان داده شده است. ۱۶ دُور پردازش وجود دارد. از کلید اولیهٔ ۵۴- بیتی، شانزده زیرکلید تولید می‌شود که هرکدام در یک دُور پردازش مورد استفاده قرار می‌گیرند.

نحوهٔ رمزگشائی با DES ضرورتاً شبیه نحوهٔ رمزنگاری با آن است. قاعده چنین است: متن رمزشده را بعنوان ورودی الگوریتم DES بکار برده ولی از زیرکلیدها با نظم معکوس استفاده کنید. یعنی در اولین تکرار کلید K_{16}، در دومین تکرار کلید K_{15}، و بهمین نحو جلورفته و در شانزدهمین و آخرین تکرار کلید K_1 را بکار برید.

توانائی DES

نگرانی نسبت به توانائی DES در دو مقولهٔ جدا قرار دارد: نگرانی در مورد خود الگوریتم و نگرانی در مورد استفاده از یک کلید ۵۴- بیتی. اولین نگرانی، در رابطه با امکان شکستن رمز با استفاده از بکارگیری مشخصه‌های الگوریتم DES است.

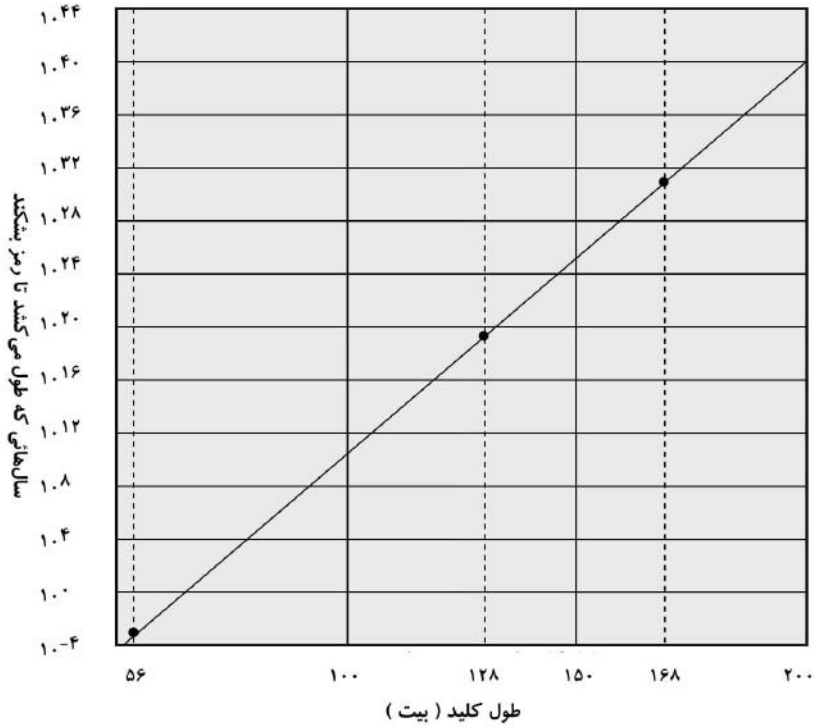
در طول سالیان گذشته، تلاش‌های بسیاری برای کشف و سوءاستفاده از نقاط ضعف الگوریتم DES انجام شده و بهمین مناسبت DES الگوریتمی است که بیش از همه مورد مطالعه قرارگرفته است. با وجود تلاش‌های فراوان، تا کنون کسی نتوانسته است که یک ضعف حیاتی در DES پیدا کند.

نگرانی جدی‌تر مربوط به طول کلید است. با کلیدی با طول ۵۶ بیت، تعداد $۲^{۵۶}$ کلید ممکن وجود دارد که تقریباً $۷/۲ \times ۱۰^{۱۶}$ کلید است. بنابراین در صورت ظاهر، یک حملهٔ همه جانبه غیرعملی خواهد بود. با فرض اینکه بطور متوسط نصف فضای کلید را بایستی برای یافتن آن جستجو کرد، اگر قرار باشد یک ماشین تنها که در هر میکروثانیه یک فقره رمزگشایی DES را انجام میدهد بکارگرفته شود، بیش از هزارسال طول خواهد کشید تا رمز شکسته شود (جدول ۲-۲ را ملاحظه کنید).

اما فرض یک رمزنگاری در هر میکروثانیه بیش از حد، محافظه‌کارانه است. بالاخره و بطور قطعی در ماه جولای سال ۱۹۹۸ اثبات گردید که DES ناامن است. دلیل امر این بود که Electronic Frontier Foundation (EFF) اعلام کرد که رمز یک DES را با استفاده از یک ماشین مخصوص "DES cracker" که با هزینهٔ کمتر از ۲۵۰,۰۰۰ دلار ساخته شده است، شکسته است. این حمله کمتر از سه روز طول کشیده بود. EFF توصیف مفصلی از ماشین مزبور را منتشر نموده است تا دیگران نیز بتوانند رمزشکن خود را بسازند [EFF98] و البته با توجه به اینکه با افزایش سرعت، قیمت سخت‌افزارها پائین می‌آید، DES بطور ضمنی بی‌ارزش خواهد شد.

مهم است توجه شود که در حملهٔ جستجوی کلید، نکات مهم‌تری نیز سوای جستجوی همهٔ کلیدها وجود دارد. بغیر از موردی که واقعاً یک متن ساده در دسترس باشد، یک تحلیل‌گر بایستی یک متن ساده را بعنوان متن ساده تشخیص دهد. اگر پیام صرفاً یک متن ساده به زبان انگلیسی باشد در این صورت نتیجه به سهولت استخراج خواهد شد، اگرچه وظیفهٔ شناخت زبان انگیسی را باید خودکار نمود. اگر متن پیام قبل از رمزنگاری فشرده شده باشد، شناخت آن دشوارتر خواهد شد و اگر پیام نمونهٔ عام‌تری از دیتا، همانند یک فایل عددی بوده و فشرده‌سازی هم شده باشد، مشکل تحلیل خودکار آن بازهم پیچیده‌تر خواهد گردید. بنابراین برای فراهم‌آوردن روش همه جانبه، مقداری دانش در مورد متن ساده مورد نیاز بوده و همچنین لازم است تا وسیله‌ای برای تمیزدادن متن ساده از یک متن بی‌معنی بصورت خودکار وجود داشته باشد. روش EFF این مقوله را مورد توجه قرار داده و همچنین تکنیک‌های خودکاری را که در برخی زمینه‌ها مؤثرند، معرفی می‌کند.

یک نکتهٔ نهائی: اگر تنها فرم ممکن حمله نسبت به یک الگوریتم رمزنگاری، حملهٔ همه جانبه باشد، آنگاه روش مقابله با آن کاملاً روشن بوده و آن استفاده از کلیدهای طولانی‌تراست. برای این که ایده‌ای نسبت به اندازهٔ کلید مورد نیاز پیدا کنیم، اجازه دهید تا از رمزشکن EFF برای تخمین امر استفاده نمائیم. EFF cracker یک نمونهٔ منحصربفرد بوده و ما می‌توانیم فرض کنیم که با تکنولوژی امروز، ساخت یک ماشین سریع‌تر مقرون بصرفه‌تر است. اگر فرض کنیم که یک رمزشکن بتواند در هر میکروثانیه یک میلیون رمزگشائی انجام دهد، که نرخی است که در جدول ۲-۲ از آن استفاده شده است، آنگاه تقریباً ۱۰ ساعت طول خواهد کشید تا یک رمز DES شکسته شود. سرعت این رمزشکنی تقریباً ۷ برابر بیشتر از نتیجهٔ EFF است. با استفاده از این نرخ، شکل ۳-۲ نشان می‌دهد که چقدر طول خواهد کشید تا الگوریتمی با فرم DES را برحسب تابعی از اندازهٔ کلید شکست. بعنوان مثال برای یک کلید ۱۲۸- بیتی، که در الگوریتم‌های فعلی مرسوم است، بیش از $۱۰^{۱۸}$ سال طول خواهد کشید تا بتوان رمزی را، با استفاده از رمزشکن EFF شکست. حتی اگر بتوان سرعت رمزشکن را با فاکتور یک تریلیون ($۱۰^{۱۲}$) افزایش داد، بازهم یک میلیون سال طول خواهد کشید تا رمز شکسته شود. بنابراین یک کلید ۱۲۸- بیتی برای استفاده در الگوریتمی که در مقابل حملهٔ همه جانبه شکست‌ناپذیر باشد، یک انتخاب تضمین شده است.



شکل ۳-۲ زمان شکستن یک رمز (با فرض $۱۰^۶$ رمزگشائی در هر میکروثانیه)

Triple DES

Triple DES (3DES)، اولین بار در سال ۱۹۸۵ برای استفاده درکاربردهای مالی، با نام X9.17 در استانداردهای ANSI ثبت گردید. 3DES با انتشار FIPS PUB 46-3 در سال ۱۹۹۹ بعنوان بخشی از استاندارد رمزنگاری دیتا (DES) بکار گرفته شد.

3DES از سه کلید و سه بار اجرای الگوریتم DES استفاده می‌کند. تابع از یک دنبالهٔ رمزنگاری- رمزگشائی- رمزنگاری (EDE) تبعیت می‌کند(شکل ۴-۲الف):

$$C = \mathrm{E}(K_3, \mathrm{D}(K_2, \mathrm{E}(K_1, P)))$$

که در آن

C = متن رمزشده (ciphertext)

P = متن ساده (plaintext)

$\mathrm{E}[K, X]$ = رمزنگاری X با استفاده از کلید K

$\mathrm{D}[K, Y]$ = رمزگشائی Y با استفاده از کلید K

رمزگشائی بسادگی همان عملیات قبل است که ترتیب کلیدها در آن عوض شده است(شکل ۴-۲ب):

$$P = \mathrm{D}(K_1, \mathrm{E}(K_2, \mathrm{D}(K_3, C)))$$

ازنظر رمزنگاری، هیچ ویژگی خاصی در استفاده از رمزگشائی مرحلهٔ دوم رمزنگاری 3DES وجود ندارد. تنها حسن آن این است که به کاربران 3DES اجازه می‌دهد تا داده‌هائی را که بتوسط فرم قدیمی DES رمزنگاری شده بودند، رمزگشائی نمایند:

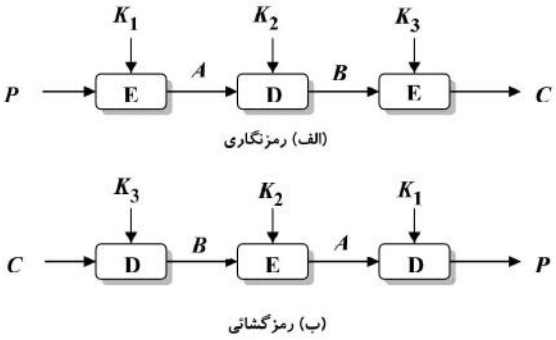
$$C = \mathrm{E}(K_1, \mathrm{D}(K_1, \mathrm{E}(K_1, P))) = \mathrm{E}[K, P]$$

با سه کلید متمایز، 3DES دارای کلیدی با طول مؤثر ۱۶۸ بیت است. FIPS 46-3 همچنین استفاده از دو کلید، $K_1 = K_3$، را اجازه میدهد که در این مورد طول کلید ۱۱۲ بیت خواهد بود. FIPS 46-3 شامل سه دستورالعمل زیر برای 3DES است:

- 3DES الگوریتم رمزنگاری متقارن منتخب و تأئیدشده بتوسط FIPS است.
- DES اولیه که از یک کلید ۵۶- بیتی استفاده می‌کند تنها در استاندارد سیستم‌هائی که وارث آن هستیم مجاز می‌باشد. ساختارهای جدید بایستی از 3DES حمایت نمایند.
- توصیه میشود که سازمان‌های دولتی، سیستم‌های موروثی DES را با 3DES تعویض نمایند.
- پیش‌بینی میشود که 3DES و AES در کنار هم، بعنوان الگوریتم‌های پذیرفته شدهٔ FIPS، همزیستی داشته و در طول زمان بتدریج 3DES حذف و AES استاندارد غالب شود.

بسهولت می‌توان دریافت که 3DES یک الگوریتم نیرومند است، چون الگوریتم رمزنگاری بستر آن DEA (Data Encryption Algorithm) است، 3DES می‌تواند در مقابل تلاش‌های مربوط به کشف رمز، همان ادعاهای DEA را داشته باشد. علاوه برآن با یک کلید ۱۶۸- بیتی، حملهٔ همه جانبه به آن عملاً غیرممکن است.

بالاخره و در نهایت قرار است AES جایگزین 3DES شود، ولی این تحول سالها طول خواهد کشید. NIST پیش‌بینی میکند که 3DES برای آینده‌ای قابل پیش‌بینی، الگوریتم موفقی باشد.



شکل ۴-۲ Triple DES

استاندارد رمزنگاری پیشرفته Advanced Encryption Standard (AES)

3DES دارای دو جاذبه است که استفادهٔ گسترده از آن در چندسال آینده را تضمین می‌کند. اولاً با طول کلید ۱۶۸- بیتی خود، بر آسیب‌پذیری‌های ناشی از حملهٔ همه جانبه در DEA غلبه می‌کند. ثانیاً الگوریتم رمزنگاری 3DES همان DEA است. این الگوریتم بیش از هر الگوریتم رمزنگاری دیگر در طول زمان مورد رسیدگی دقیق قرار گرفته و هیچ نوع آسیب‌پذیری، بجز مورد حملهٔ همه جانبه، در آن مشاهده نشده است. در نتیجه در مورد مقاومت 3DES در مقابل کشف رمز اعتماد زیادی وجود دارد. از این‌رو اگر فقط مسألهٔ امنیت مورد توجه بود، 3DES الگوریتم انتخابی مناسبی برای رمزنگاری در طول دهه‌های آینده باقی می‌ماند.

مشکل اصلی 3DES این است که این الگوریتم از نظر نرم‌افزاری لَخت است. DEA اولیه برای تجهیزات سخت‌افزاری سال‌های میانهٔ ۱۹۷۰ طراحی شده بود و کُد نرم‌افزاری بهره‌وری را تولید نمی‌کند. 3DES که سه برابر DES عملیات اجرائی دارد، حتماً کندتر خواهد بود. مشکل دوم این است که DEA و 3DES هر دو از یک بلوک دیتا با اندازهٔ ۶۴- بیت استفاده می‌کنند. به دلایلی که هم مربوط به بهره‌وری و هم مربوط به مسائل امنیتی می‌شود، استفاده از بلوکی با اندازهٔ بزرگتر مطلوب‌تر است.

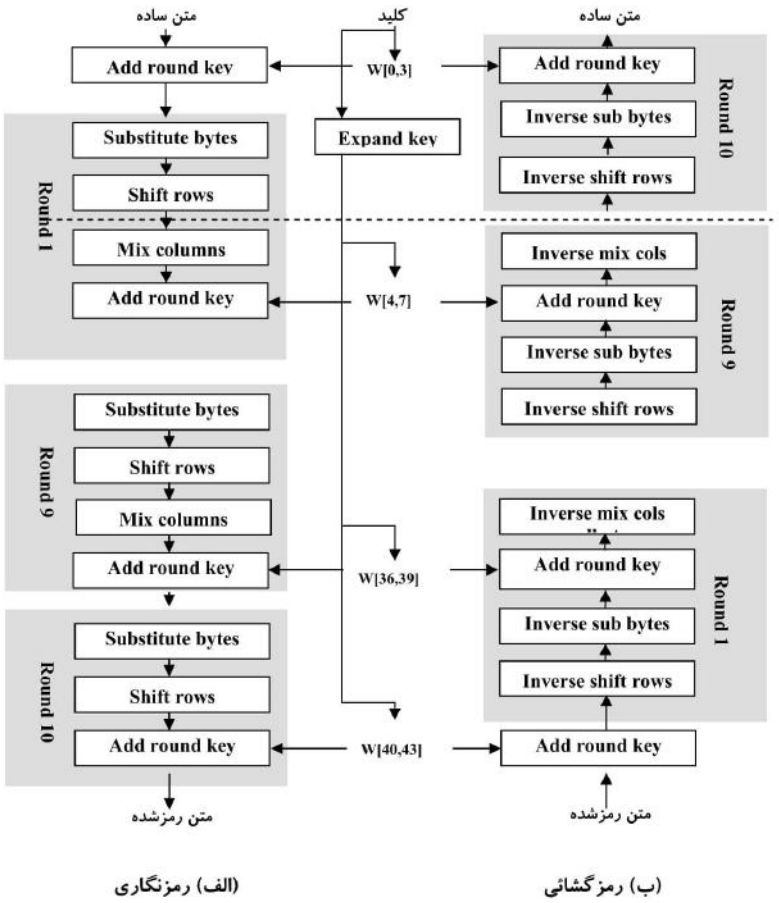
بعلت این مشکلات، 3DES در درازمدت کاندیدای معقولی نیست. برای جانشینی آن با انتخاب بهتری، NIST در سال ۱۹۹۷، فراخوانی برای طراحی یک استاندارد رمزنگاری پیشرفته (AES) منتشر کرد که بایستی دارای توان امنیتی برابر و یا بهتر از 3DES و بهره‌وری قابل ملاحظه‌ای می‌بود. علاوه بر بیان نیازهای کلی، NIST مشخص نمود که AES بایستی یک رمز قالبی متقارن با طول بلوک ۱۲۸- بیت بوده و از کلیدهائی با طول ۱۲۸، ۱۹۲، و ۲۵۶ بیت پشتیبانی نماید. نکات مورد ارزیابی شامل امنیت، بهره‌وری محاسباتی، نیازهای مربوط به حافظه، تناسب سخت‌افزار و نرم‌افزار و قابلیت انعطاف اعلام گردید.

در اولین دور ارزیابی، ۱۵ الگوریتم از بین پیشنهادها انتخاب شدند. در دور بعدی، ۵ الگوریتم پذیرفته شدند. بالاخره NIST ارزیابی خود را به پایان رسانده و یک استاندارد نهائی (FIPS PUB 197) را در نوامبر سال ۲۰۰۱ منتشرنمود. Rijndael بعنوان الگوریتم انتخابی AES پذیرفته شد. دو پژوهشگری که Rijndael را تهیه و ارائه کردند هر دو رمزنگارانی از بلژیک به نام‌های Dr. Joan Daemen و Dr. Vincent Rijmen بودند.

بررسی الگوریتم

AES از یک بلوک دیتا با طول ۱۲۸ بیت و یک کلید که میتواند ۱۲۸، ۱۹۲، و یا ۲۵۶ بیت باشد استفاده می‌کند. در این بررسی طول کلید را ۱۲۸ بیت فرض می‌کنیم که احتمالاً یکی از پراستفاده‌ترین آنها خواهد بود.

شکل ۵-۲ ساختار کلی AES را نشان می‌دهد. ورودی الگوریتم‌های رمزنگاری و رمزگشائی یک بلوک منفرد ۱۲۸- بیتی است. در FIPS PUB 197، این بلوک بصورت یک ماتریس مربعی از بایت‌ها تعریف شده است. این بلوک در رشتهٔ **state** کپی شده که در هر مرحلهٔ رمزنگاری یا رمزگشائی تعدیل می‌شود. بعد از آخرین مرحله، **state** در ماتریس خروجی کپی می‌شود. بطریق مشابه، کلید ۱۲۸- بیتی بصورت یک ماتریس مربعی از بایت‌ها تعریف می‌شود. این کلید سپس بر اساس برنامهٔ کلید(key schedule) گسترش می‌یابد. هرکلمه شامل ۴ بایت بوده و کل برنامه کلید، ۴۴ کلمه برای یک کلید ۱۲۸- بیتی را تولید می‌کند. نظم بایت‌ها در یک ماتریس، ستونی است. بنابراین برای مثال چهار بایت اول ورودی متن ساده ۱۲۸- بیتی به رمزنگار، اولین ستون ماتریس **ورودی**، چهار بایت دوم ستون دوم و غیره را تشکیل می‌دهد. بطریق مشابه، اولین ۴ بایت کلید گسترش یافته که یک کلمه را تشکیل می‌دهد، اولین ستون ماتریس **w** را می‌سازد.



شکل ۵-۲ رمزنگاری و رمزگشائی AES

موارد ذیل، نکاتی در مورد AES را روشن می‌سازد:

۱- یکی ازخصوصیات قابل توجه این ساختار این است که یک ساختار Feistel نیست. بخاطر آورید که در ساختار کلاسیک Feistel، یک نیمه از بلوک دیتا برای تغییر نیمهٔ دیگر بکار میرفت و آنگاه دو نیمه جای خود را عوض می‌کردند. AES از ساختارFeistel استفاده نکرده بلکه در هر دُور، کل بلوک دیتا را بصورت موازی بکارگرفته و جایگزینی و جابجائی را در آن انجام می‌دهد.

۲- کلیدی که در ورودی فراهم میشود، بصورت یک رشتهٔ ۴۴- تائی از کلمات ۳۲ بیتی **w**[i] گسترش می‌یابد. در هر دُور ۴ کلمهٔ مجزا (۱۲۸ بیت) بعنوان کلید دُور مورد استفاده قرار می‌گیرد.

۳- از چهار عمل مختلف که یکی از آنها جابجائی و سه‌تای دیگر جایگزینی است استفاده می‌شود:

- **بایت‌ها جابجا شوند:** از یک جدول که S-box نامیده می‌شود استفاده کرده تا بایت به بایت بلوک را جابجا کند.
- **سطرها شیفت داده شوند:** یک جابجائی ساده که ردیف به ردیف انجام می‌شود.
- **ستون‌ها مخلوط شوند:** یک جابجائی که هر بایت یک ستون را بصورت تابعی از تمام بایت‌های همان ستون تغییر می‌دهد.
- **کلید دُور اضافه شود:** یک XOR ساده که بیت‌های بلوک فعلی را با بخشی از کلید گسترش یافته XOR نماید.

۴- ساختار کاملاً ساده است. هم برای رمزنگاری و هم برای رمزگشائی، رمز با یک مرحله اضافه کردن کلید دُور (Add Round Key) شروع شده و بدنبال آن با نه دُور دیگر که هرکدام شامل چهار مرحله است ادامه یافته و در انتها با سه مرحله در دُور دهم خاتمه می‌یابد. شکل ۶-۲ سازمان یک دُور رمزنگاری کامل را نشان می‌دهد.

۵- تنها مرحلهٔ Add Round Key از کلید استفاده می‌کند. بهمین دلیل رمز با مرحلهٔ Add Round Key شروع و خاتمه می‌یابد. هر مرحلهٔ دیگر بدون نیاز به کلید قابل برگشت بوده و بنابراین چیزی به امنیت اضافه نمی‌کند.

۶- مرحلهٔ Add Round Key به‌تنهائی نیرومند نیست. سه مرحلهٔ دیگر بیت‌ها را مخلوط کرده ولی خود امنیتی را ایجاد نمی‌نمایند، زیرا از کلید استفاده نمی‌کنند. میتوان رمز را بصورت رمزنگاری XOR (Add Round Key) یک بلوک و پس از آن درهم ریختن بلوک (سه مرحلهٔ دیگر) و بدنبال آن رمزنگاری XOR و غیره در نظر گرفت. این روش هم بهره‌ور و هم بغایت امن است.

۷- هر مرحله به آسانی برگشت‌پذیر است. برای مراحل جابجائی بایت، شیفت ردیف، و مخلوط کردن ستون، یک تابع معکوس در الگوریتم رمزگشائی بکار رفته است. برای مرحلهٔ Add Round Key، عمل عکس با XOR کردن همان کلید دور به بلوک حاصل می‌گردد زیرا $A \oplus A \oplus B = B$ است.

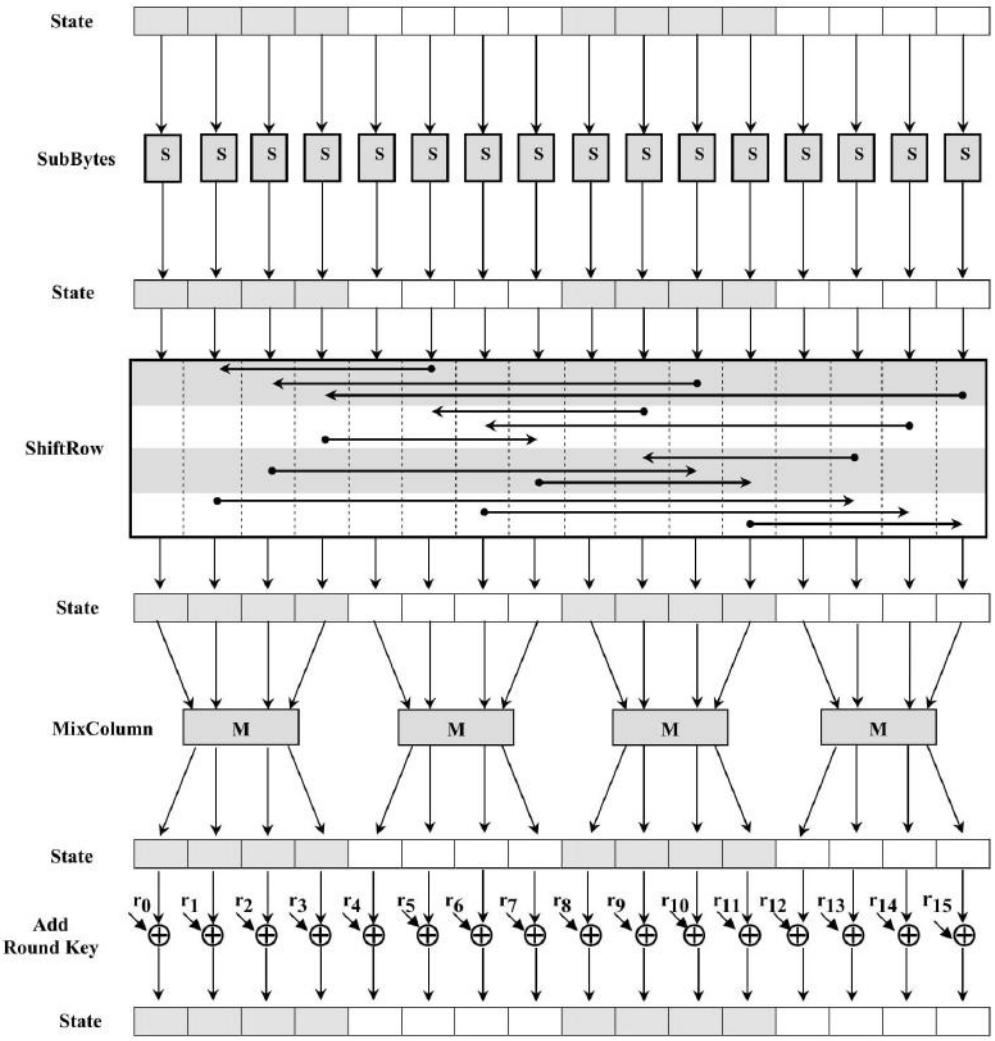
۸- همانند اکثر رمزهای قالبی، الگوریتم رمزگشائی از کلید گسترش‌یافته با نظم معکوس استفاده می‌کند. با وجود این الگوریتم رمزگشائی شبیه الگوریتم رمزنگاری نیست. این نتیجهٔ ساختار خاص AES است.

۹- وقتی روشن شد که هر چهار مرحله بازگشت‌پذیر هستند. آنگاه تأئید اینکه رمزگشائی متن ساده را احیاء خواهد کرد، آسان خواهد بود. شکل ۵-۲ رمزنگاری و رمزگشائی را در دو ستون کنارهم با جهت‌های مختلف نشان داده است. در هر سطح افقی (مثل خط‌چین‌ها در شکل)، state برای هم رمزنگاری و هم رمزگشائی یکی است.

۱۰ - دور نهائی چه در عمل رمزنگاری و چه در عمل رمزگشائی فقط دارای سه مرحله است. بازهم این نتیجهٔ ساختار خاص AES است و لازم است چنین باشد تا رمز بازگشت‌پذیر باشد.

۲-۳ رمزهای دنباله‌ای و RC4

یک *رمز قالبی* در هر زمان یک بلوک از عناصر ورودی را پردازش نموده و یک بلوک خروجی برای آن بلوک ورودی تولید می‌کند. یک *رمز دنباله‌ای*، عناصر ورودی را بطور پیوسته پردازش کرده و همینطور که جلو می‌رود عنصر به عنصر متن رمزشده را تولید می‌کند. اگرچه رمزهای قالبی بسیار متداول‌ترند، ولی در برخی کاربردها یک رمز دنباله‌ای گزینه‌ای مناسب‌تر است. مثال‌هائی از این کاربردها را در بخش‌های بعدی معرفی خواهیم کرد. در این بخش به متداول‌ترین رمز دنباله‌ای متقارن یعنی RC4 نگاهی می‌اندازیم. ابتدا مروری بر ساختار رمزهای دنباله‌ای داشته و سپس RC4 را بررسی خواهیم کرد.



شکل ۶-۲ یک دَور عملیاتی AES

ساختار رمزهای دنباله‌ای

یک رمز دنباله‌ای معمولاً متن ساده را بصورت بایت- به- بایت رمزنگاری می‌نماید. البته می‌توان رمزهای دنباله‌ای دیگری خلق کرد که داده‌ها را بصورت بیت- به- بیت و یا در واحدهای بزرگ‌تر از بایت در هر زمان رمزنگاری نماید. شکل ۷-۲ نمایش‌دهندهٔ ساختار یک رمز دنباله‌ای است. در این ساختار یک کلید، ورودی یک تولیدکنندهٔ شبه‌تصادفی بیت‌ها بوده که یک

دنبالهٔ ۸- بیتی که ظاهراً تصادفی به نظر می‌رسد را تولید می‌کند. خروجی تولیدکنندهٔ شبه تصادفی، که یک **دنبالهٔ کلید (keystream)**، نامیده می‌شود با دنبالهٔ متن سادهٔ ورودی بصورت یک بایت در هر زمان و بصورت عمل XOR روی بیت‌ها ترکیب می‌شود. برای مثال اگر بایت تولیدشده بتوسط مولد 01101100 و بایت متن ساده 11001100 باشد، آنگاه بایت متن رمزشدهٔ حاصل چنین است:

متن ساده	11001100 ⊕
دنبالهٔ کلید	01101100
متن رمزشده	10100000

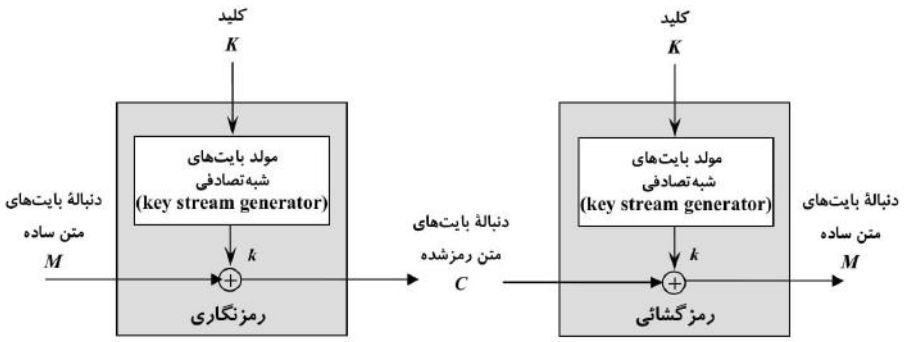
رمزگشائی نیاز به استفاده از همان ردیف شبه‌تصادفی را دارد.

متن رمزشده	10100000 ⊕
دنبالهٔ کلید	01101100
متن ساده	11001100

[KUMA97] ملاحظات مهم زیر در طراحی یک رمز دنباله‌ای را ذکر کرده است:

۱- دنبالهٔ رمز بایستی دارای دورهٔ تناوب بزرگی باشد. یک تولیدکنندهٔ اعداد شبه‌تصادفی از تابعی استفاده می‌کند که یک دنبالهٔ یقینی از بیت‌ها را تولید کرده که نهایتاً بعداز مدتی تکرار می‌شوند. هرچقدر دورهٔ تناوب این تکرار طویل‌تر باشد، عمل شکستن رمز سخت‌تر خواهد بود.

۲- دنبالهٔ کلید بایستی با تقریب بسیار خوب، خواص یک دنبالهٔ عدد تصادفی واقعی را داشته باشد. بعنوان مثال تقریباً بایستی تعداد 1ها و 0ها در این دنباله برابر باشند. اگر دنباله کلید بصورت یک ردیف از بایت‌ها مورد استفاده قرار گیرد، آنگاه تمام ۲۵۶ حالت ممکن بایستی تقریباً بصورت مساوی مورد استفاده قرار گیرند. هر چقدر دنبالهٔ کلید تصادفی‌تر بنظر آید، متن رمزشده تصادفی‌تر بوده و شکستن رمز سخت‌تر خواهد بود.



شکل ۷-۲ دیاگرام رمز دنباله‌ای

۳- با توجه به شکل ۷-۲ می‌توان دریافت که خروجی یک مولد اعداد شبه‌تصادفی به اندازهٔ کلید ورودی وابسته است. برای جلوگیری از حملات همه جانبه، کلید بایستی به اندازهٔ کافی بزرگ باشد. همان ملاحظاتی که در مورد رمزهای قالبی وجود داشت در اینجا نیز صادق‌اند. بنابراین با تکنولوژی کنونی، کلیدی با طول حداقل ۱۲۸ بیت مناسب بنظر می‌رسد.

با یک مولد اعداد شبه‌تصادفی با طرح مناسب، یک رمز دنباله‌ای می‌تواند بهمان اندازهٔ یک رمز قالبی، با همان طول کلید، امن باشد. مزیت اصلی یک رمز دنباله‌ای این است که رمزهای دنباله‌ای تقریباً همیشه سریع‌تر بوده و نسبت به رمزهای قالبی از حجم برنامهٔ کمتری استفاده می‌کنند. رمز RC4 که در این بخش تشریح شده است تنها می‌تواند با چند خط برنامهٔ کامپیوتری پیاده‌سازی شود. جدول ۳-۲ که از اطلاعات [RESC01] اقتباس شده است، زمان اجرای RC4 را با سه رمز قالبی معروف مقایسه کرده است. حسن یک رمز قالبی در این است که شما می‌توانید از کلید رمز بارها استفاده کنید. در رمز دنباله‌ای اگر دو متن ساده با یک کلید یکسان رمزنگاری شوند، آنگاه شکستن رمز غالباً بسیار آسان خواهد بود [DAWS96]. اگر دو دنبالهٔ متن رمزشده با هم XOR شوند، نتیجه با XOR دو متن سادهٔ نظیر آنها یکسان خواهدبود. حال اگر متن ساده، دنبالهٔ کوتاهی همانند شمارهٔ کارت‌های اعتباری و یا ردیف‌های دیگری با خواص شناخته شده باشند، عمل شکستن رمز ممکن است موفقیت‌آمیز باشد.

برای کاربردهائی همانند کانال‌های مخابرهٔ داده‌ها و یا مرور لینک‌های وِب که نیاز به رمزنگاری/ رمزگشائی دنباله‌های دیتا دارند، یک رمز دنباله‌ای می‌تواند گزینهٔ بهتری باشد. برای کاربردهائی همچون انتقال فایل، پست الکترونیک و پایگاه داده که با بلوک‌های دیتا سروکار دارند، رمزهای قالبی می‌توانند مناسب‌تر باشند. با وجود این هر دو نوع رمز تقریباً در هر کاربردی قابل استفاده‌اند.

الگوریتم RC4

RC4 یک رمز دنباله‌ای است که در سال ۱۹۸۷ میلادی بتوسط Ron Rivest برای کمپانی RSA Security طراحی گردید. RC4 یک رمز دنباله‌ای با طول کلید متغیر بوده و عملیات آن روی بایت‌ها انجام می‌شود. الگوریتم بر مبنای استفاده از یک جایگشت تصادفی بنا نهاده شده است. تحلیل این رمز نشان می‌دهد که دورهٔ تناوب رمز با احتمال قریب به یقین بزرگتر از 10^{100} است [ROBS95a]. برای تولید هر بایت خروجی بین ۸ تا ۱۶ عمل لازم است و انتظار می‌رود که رمزنگاری در نرم‌افزار به سرعت انجام شود. RC4 در استانداردSSL/TLS (Secure Socket Layer/Transport Layer Security) که برای ارتباط بین مرورگرهای وِب و سِرورها تعریف شده است، بکار می‌رود. این رمز همچنین در پروتکل WEP (Wired Equivalent Privacy) و پروتکل جدیدتر WPA (WiFi Protected Access) که بخشی از استانداردهای IEEE 802.11 مربوط به LAN بی‌سیم هستند، مورد استفاده است. RC4 از نظر تجاری مدتها از سوی کمپانی RSA Security پنهان نگاه داشته شده بود. در سپتامبر ۱۹۹۴ این الگوریتم بصورت ناشناس در لیست پستی Cypherpunks قرار گرفت و لو رفت.

الگوریتم RC4 بصورت قابل توجهی ساده بوده و تشریح آن کاملاً آسان است. یک کلید با طول متغیر ۱ تا ۲۵۶ بایت (۸ تا ۲,۰۴۸ بیت) برای آغازیدن یک بردار حالت ۲۵۶- بایتی S با مؤلفه‌های S[0]، S[1]..... S[255] مورد استفاده قرار می‌گیرد. در همهٔ حالات، S شامل جایگشت همهٔ اعداد ۸- بیتی از صفر تا ۲۵۵ است. برای رمزنگاری و رمزگشائی، یک بایت k (شکل ۷-۲ را ببینید) از میان ۲۵۵ مؤلفهٔ S بصورت سیستماتیک انتخاب می‌شود. همینطور که هر مقدار k تولید می‌شود، مؤلفه‌های S یک‌بار دیگر جایگشت می‌یابند.

جدول ۳-۲ مقایسهٔ سرعت پـــردازش رمزهای متقارن روی یک پـــردازش‌گر Pentium II

نوع رمز	طول کلید	سرعت (Mbps)
DES	۵۶	۹
3DES	۱۶۸	۳
RC2	متغیر	۰/۹
RC4	متغیر	۴۵

آغازیدن S

برای شروع، مقادیر صفر تا ۲۵۵ بصورت صعودی در مؤلفه‌های S قرار داده می‌شـود، یعنـی S[0] = 0، S[1] = 1 و S[255] = 255. همچنین یک بردار موقت T خلق می‌شود. اگر طول کلید K برابر ۲۵۶ بایت باشد، آنگـاه K بـه T منتقـل می‌شود. در غیر اینصورت برای کلیدی با طول keylen بایت، اولین مؤلفه‌های T از K کپی شـده و سـپس K هـر چنـدبار لازم باشد تکرار شده تا T پر شود. این عملیات ابتدائی را می‌توان چنین خلاصه کرد:

```
/* Initialization */
for I = 0 to 255 do
S[i] = i ;
T[i] = K[i mod keylen] ;
```

سپس از T برای جایگشت آغازین S استفاده می‌شود. این امر با S[0] شروع شده و تا S[255] ادامه می‌یابد. هر S[i] با بایت دیگری در S بر اساس روشی که بتوسط T[i] دیکته می‌شود تعویض می‌شود:

```
/* Initial Permutation of S */
J = 0 ;
for I = 0 to 255 do
j = ( j + S[i] + T[i] ) mod 256 ;
Swap (S[i] , S[j]) ;
```

چون تنها عمل روی S یک تعویض محل بایت‌هاست، تنها اثر این امر ایجاد یک جایگشت است. S همچنان شامل تمام اعداد بین صفر تا ۲۵۵ خواهد بود.

تولید دنباله

همین‌که بردار S با مقادیر اولیه پر شد، دیگر از کلید ورودی استفاده نخواهد شد. تولید دنباله شامل عبور از S[0] تا S[255] بوده و هر مقدار S[i] با بایت دیگری در S، برحسب قانونی که بتوسط وضع فعلی S دیکته می‌شود، جایگزین می‌گردد. بعد از اینکه به S[255] رسیدیم، پردازش با شروع مجدد از S[0] ادامه می‌یابد:

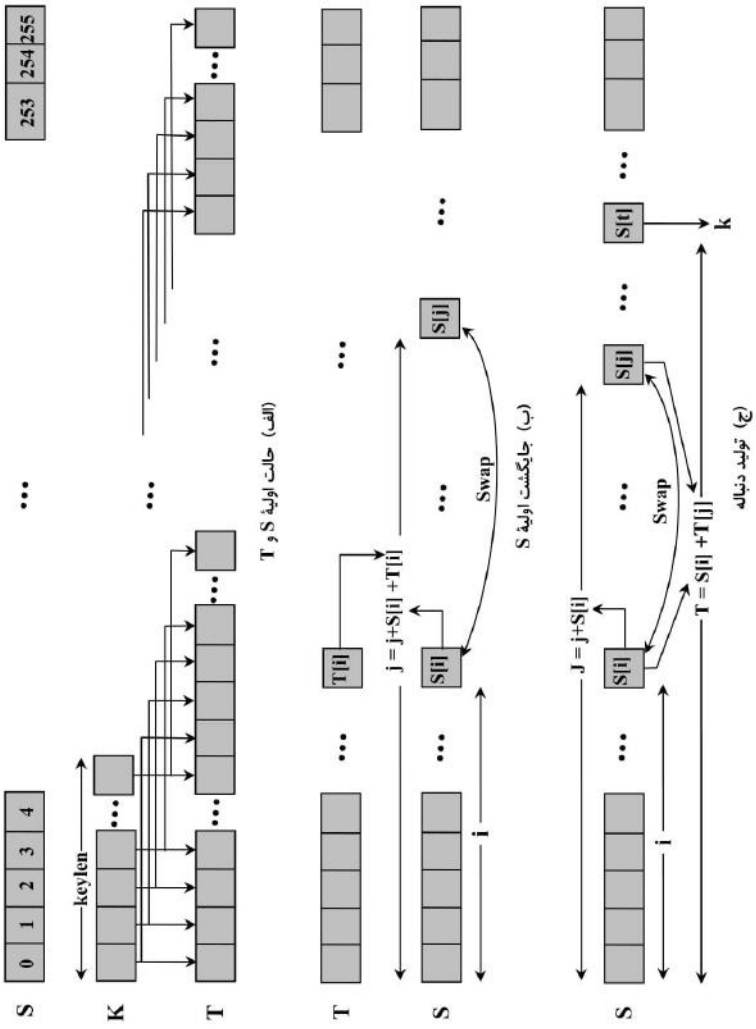
```
/* Stream Generation */
i , j = 0 ;
While (true)
I = (i+1) mod 256 ;
j = (j + S[i]) mod 256
Swap (S[i] , S[j]) ;
T = ( S[i] + S[j] ) mod 256 ;
K = S[t] ;
```

برای رمزنگاری، اندازهٔ k را با بایت بعدی متن ساده XOR می‌کنیم. برای رمزگشائی، اندازهٔ k را با بایت بعدی متن رمزشده XOR می‌کنیم.

شکل ۸-۲ منطق RC4 را نشان می‌دهد.

توانائی RC4

مقاله‌های متعددی نوشته شده‌اند که روش حمله به RC4 را تحلیل کرده‌اند (مثلاً [KNUD98]، [MIST98]، [FLUH00] و [MANT01]). هیچکدام از این روش‌ها برای حمله به RC4 با کلیدی که دارای طول منطقی همچون ۱۲۸ بیت باشد، عملی نیستند. یک مورد جدی‌تر در [FLUH01] مطرح گردید. نویسندگان مقاله نشان دادند که پروتکل WEP که برای فراهم نمودن محرمانگی در شبکه‌های LAN بی‌سیم از پروتکل 802.11 استفاده می‌کند، در برابر حملهٔ بخصوصی آسیب‌پذیر است. در واقع مشکل به RC4 ربطی نداشته بلکه به روشی که کلیدها برای استفاده در ورودی RC4 تولید می‌شوند، مرتبط است. این مشکل بخصوص در سایر کاربردهائی که از RC4 استفاده می‌کنند ظاهر نشده و در WEP نیز با تغییر روش تولید کلیدها، مشکل رفع خواهد شد. این مسأله، مشکل طراحی یک سیستم امن، که هم از تابع رمزنگاری و هم از پروتکل‌هائی که این توابع را بکار می‌گیرند استفاده می‌کند، را خاطرنشان می‌سازد.



شکل ۸-۲

۲-۴ مُودهای عملیاتی رمزهای قالبی

یک رمز قالبی متقارن، داده‌ها را بصورت یک بلوک در هر زمان پردازش می‌کند. در DES و 3DES طول بلوک ۶۴ بیت است. برای متون ساده با طول بیشتر، لازم است تا متن به بلوک‌های ۶۴- بیتی تقسیم شود (اگر لازم باشد، آخرین بلوک با بیت‌های اضافی کامل می‌شود). ساده‌ترین راه برای این کار چیزی است که آن را مُود کتاب کُد الکترونیکی electronic codebook (ECB) گویند که در آن در هر لحظه، ۶۴ بیت از متن ساده تحت پردازش قرار گرفته و همهٔ بلوک‌های متن با کلید واحدی پردازش می‌شوند. اصطلاح *کتاب کُد* (*codebook*) از این جهت بکار گرفته شده است که برای یک کلید واحد، یک متن رمزشدهٔ یکتا برای هربلوک ۶۴- بیتی دیتا حاصل می‌شود. بنابراین میتوان یک کتاب کُد عظیمی را تصور کرد که در آن برای هر بلوک ۶۴- بیتی ممکن از متن ساده، یک متن رمزشدهٔ نظیر آن وجود داشته باشد.

در ECB، اگر همان بلوک ۶۴- بیتی متن ساده بیش از یکبار در پیام ظاهر شود، همیشه همان متن رمزشدهٔ دفعهٔ اول حاصل خواهد شد. بهمین دلیل برای پیام‌های طولانی، مُود ECB ممکن است امن نباشد. اگر پیام بشدت ساختاریافته باشد، یک شکنندهٔ رمز ممکن است بتواند از این نظم سوءاستفاده کند. بعنوان مثال اگر معلوم باشد که پیام همیشه با میدان‌های از قبل تعریف شدهٔ معینی شروع می‌شود، آنگاه شکنندهٔ رمز ممکن است تعدادی زوج متن ساده- متن رمزشده را در اختیار داشته و روی آنها کار کند. اگر پیام دارای عناصر تکرارشده‌ای باشد که پریود تکرار آنها مضربی از ۶۴ بیت باشد، آنگاه این عناصر می‌توانند بتوسط تحلیل‌گر شناخته شوند. این موارد ممکن است به تحلیل رمز کمک کرده و یا ممکن است فرصتی برای جایگزینی و یا تغییر سازمان بلوک بدست دهند.

برای غلبه بر کمبودهای امنیتی ECB، علاقه‌مند به تکنیکی هستیم که با استفاده از آن، همان بلوک متن ساده در صورت تکرار، بلوک‌های رمزشدهٔ متفاوتی را ایجاد کند. در این قسمت به دو روش مختلف که در FIPS PUB 81 تعریف شده است، نگاهی می‌اندازیم.

مُود زنجیره‌ای رمز قالبی (Cipher Block Chaining Mode)

در مُود زنجیره‌ای رمز قالبی(CBC) ، (شکل ۹-۲)، ورودی الگوریتم رمزنگاری از XOR بلوک متن سادهٔ فعلی با بلوک متن رمزشدهٔ قبلی بدست می‌آید و از کلید واحدی نیز برای همهٔ مراحل استفاده شده است. اثر این امر این است که پردازش ردیف بلوک‌های متن ساده را بهم زنجیر کرده‌ایم. در نتیجه ورودی تابع رمزنگاری برای هر بلوک متن ساده، رابطهٔ ثابتی با خود بلوک متن ساده ندارد. به همین دلیل قالب‌های تکرار شده ۶۴- بیتی در خروجی ظاهر نخواهند شد.

برای رمزگشائی، هر بلوک رمزشده از الگوریتم رمزگشائی عبور می‌کند. نتیجهٔ این عمل با بلوک متن رمزشدهٔ قبلی XOR شده تا بلوک متن ساده بدست آید. برای اطمینان از صحت این روش، می‌توان نوشت:

$$C_i = \mathrm{E}\left([K, [C_{i-1} \oplus P_i]\right)$$

که درآن $\mathrm{E}[K, X]$ رمزنگاری متن سادهٔ X با استفاده از کلید K بوده و $\oplus$ عمل XOR را نشان میدهد. آنگاه

$$\mathrm{D}[K, C_i] = \mathrm{D}(K, \mathrm{E}(K, [C_{i-1} \oplus P_i]))$$
$$\mathrm{D}[K, C_i] = C_{i-1} \oplus P_i$$
$$C_{i-1} \oplus \mathrm{D}(\mathrm{K}, C_i) = C_{i-1} \oplus C_{i-1} \oplus P_i = P_i$$

که شکل ۹-۲ب را تأئید می‌کند.

برای تولید اولین بلوک متن رمزشده، یک بردار آغازگر Initialization Vector (IV) با اولین بلوک متن ساده XOR می‌شود. در زمان رمزگشائی، IV با خروجی الگوریتم رمزگشائی XOR شده تا اولین بلوک متن ساده بدست آید.

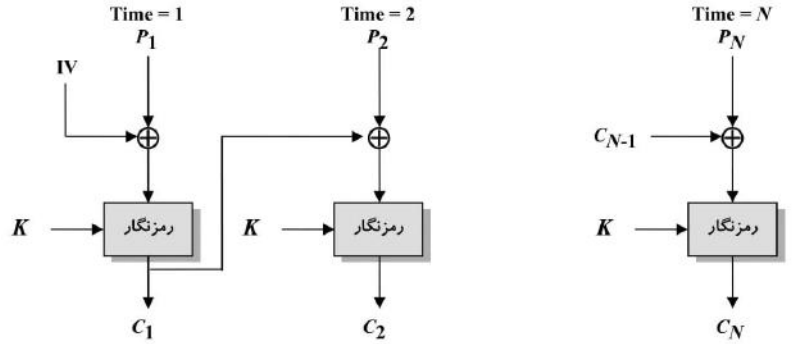
IV بایستی هم برای فرستنده و هم برای گیرنده شناخته شده باشد. برای ایجاد ماکزیمم امنیت، از IV نیز همانند کلید بایستی محافظت نمود. یکی از دلایل حفاظت از IV این است: اگر یک دشمن بتواند گیرنده را به استفاده از مقدار دیگری برای IV وادار کند، آنگاه دشمن خواهد توانست تا بیت‌های انتخاب‌شده در اولین بلوک متن ساده را معکوس نماید. برای روشن شدن مطلب فرض کنید:

$$C_1 = \mathrm{E}\,(K, [\mathrm{IV} \oplus P_1])$$

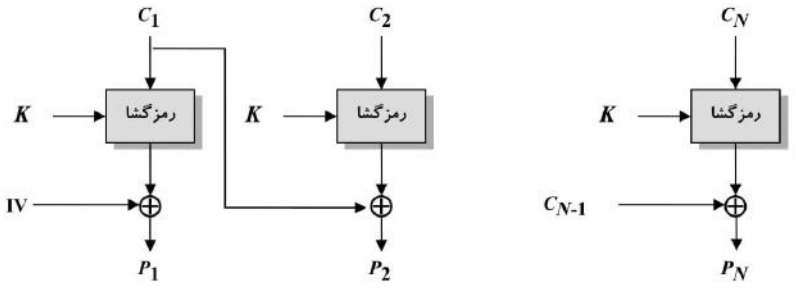
$$P_1 = \mathrm{IV} \oplus \mathrm{D}\,(K, C_1)$$

با استفاده از نمایش $X[j]$ بعنوان j امین بیت مقدار ۶۴- بیتی X ، آنگاه

$$P_1[j] = \mathrm{IV}[j] \oplus \mathrm{D}\,(K, C_1)[j]$$



(الف) رمزنگاری



(ب) رمزگشائی

شکل ۹-۲ مُود زنجیره‌ای رمز قالبی (CBC)

سپس با استفاده از خواص عمل XOR می‌توان گفت

$$P_1[j]' = \mathrm{IV}[j]' \oplus \mathrm{D}(K, C_1)[j]$$

که علامت پریم(') نمایش یک بیت نفی شده است. این بدین معنی است که اگر دشمن با تخمین بتواند بیت‌های IV را عوض کند، بیت‌های نظیر اندازهٔ دریافت شده P_1 می‌توانند تغییر یابند.

CBC کاربرد گسترده‌ای در مسائل امنیتی دارد که بعداً به آن اشاره خواهد شد.

مُود فیدبک رمز (Cipher Feedback Mode)

این امکان وجود دارد که با استفاده از مُود فیدبک رمز (CFB)، هر رمز قالبی را بصورت یک رمز دنباله‌ای درآورد. در یک رمز دنباله‌ای نیازی نیست که با اضافه کردن بیت‌ها به پیام، تا حد مضربی از بلوک‌ها، آن را کامل کرد. همچنین این رمز میتواند در حالت بلادرنگ کار کند. بنابراین اگر دنباله‌ای از کاراکترها قرار است ارسال شوند، هر کاراکتر میتواند با استفاده از یک رمز دنباله‌ای با گرایش کاراکتری، بلافاصله رمزنگاری و ارسال گردد.

یکی از خصوصیات مطلوب یک رمز دنباله‌ای این است که متن رمزشده دارای همان طول متن ساده است. بنابراین اگر کاراکترهای ۸- بیتی ارسال می‌گردند، هر کاراکتر بایستی با ۸ بیت رمزنگاری شود. اگر بیش از ۸ بیت مورد استفاده قرار گیرد، ظرفیت انتقال تلف خواهد شد.

شکل ۱۰-۲ روش CFB را به تصویر کشیده است. در این شکل فرض شده است که واحد انتقال s بیت است که اندازهٔ معمول آن ۸ میباشد. همانند CBC، واحدهای متن ساده بهم زنجیر شده‌اند بطوری‌که متن رمزشده هر واحد متن ساده، تابعی از تمام متون سادهٔ قبلی است.

در ابتدا رمزنگاری را در نظر بگیرید. ورودی تابع رمزنگاری یک شیفت رجیستر ۶۴- بیتی است که در ابتدا با یک بردار اولیه(IV) پر می‌شود. چپ‌ترین (با اهمیت‌ترین) s بیت خروجی تابع رمزنگاری با اولین واحد متن سادهٔ P_1 بصورت XOR درآمده تا اولین واحد متن رمزشدهٔ C_1 را که متعاقباً ارسال خواهد شد تشکیل دهد. علاوه برآن، محتویات شیفت رجیستر باندازهٔ s بیت به چپ شیفت داده شده و C_1 در راست‌ترین (کم اهمیت‌ترین) s بیت شیفت رجیستر جای می‌گیرد. این امر تا وقتی که تمام واحدهای متن ساده رمزنگاری شوند، ادامه می‌یابد.

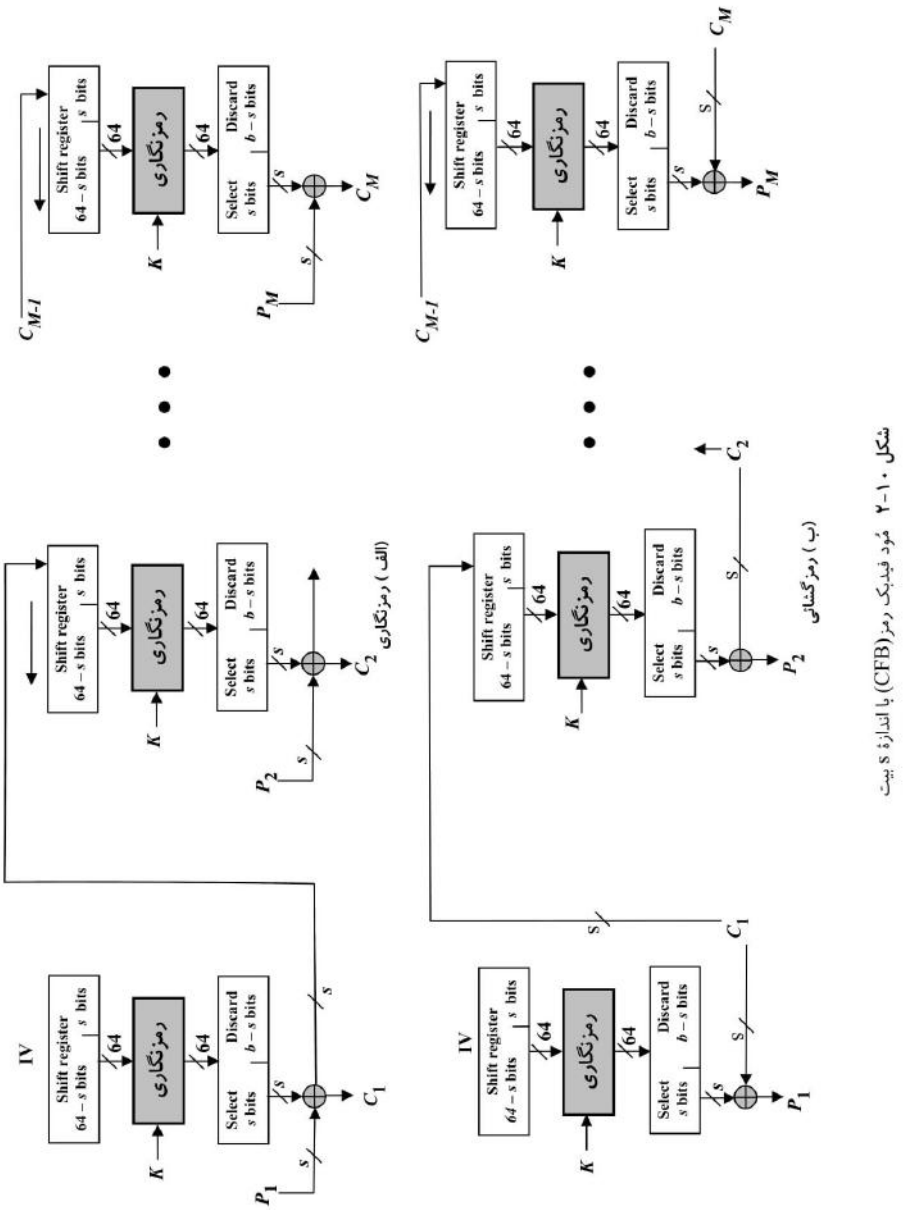
برای رمزگشائی، همین روش مورد استفاده قرار می‌گیرد بجز اینکه متن رمزشده دریافت‌شده با خروجی تابع رمزنگاری XOR شده تا متن ساده را تولید کند. توجه شود که این تابع *رمزنگاری* است که مورد استفاده قرار میگیرد نه تابع رمزگشائی. این مسأله بسادگی قابل توضیح است . فرض کنید $\mathrm{S_S}(X)$ بعنوان با اهمیت‌ترین s بیت X تعریف شود. آنگاه

$$C_1 = P_1 \oplus \mathrm{S_s}[\mathrm{E}(K, \mathrm{IV})]$$

بنابراین

$$P_1 = C_1 \oplus \mathrm{S_s}[\mathrm{E}(\mathrm{K}, \mathrm{IV})]$$

همین استدلال برای قدم‌های بعدی این پردازش نیز صادق است.



شکل ۱۰-۲ مُود فیدبک رمز(CFB) با اندازهٔ s بیت

۲-۵ محل استقرار تجهیزات رمزنگاری

قوی‌ترین و معمول‌ترین روش برای مقابله با حملات امنیتی به شبکه، رمزنگاری است. در استفاده از رمزنگاری لازم است تصمیم بگیریم که چه چیزی را رمزنگاری کرده و لوازم مربوط به رمزنگاری را در کجا قرار دهیم. در این مورد دو انتخاب اصلی وجود دارد: رمزنگاری پیوند(link encryption)، و رمزنگاری سر- به- سر (end-to-end encryption) که استفاده از آنها در عرض یک شبکهٔ سوئیچ بسته‌ای در شکل ۱۱-۲ نشان داده شده است.

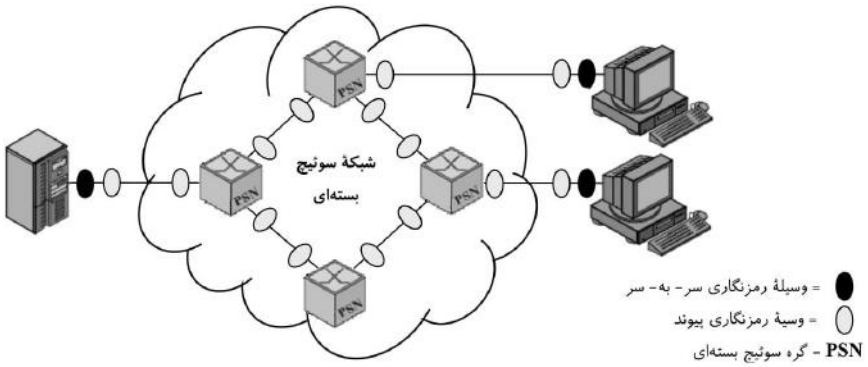
در رمزنگاری پیوند، هر پیوند مخابراتی آسیب‌پذیر، در هر یک از دو انتها با یک وسیلهٔ رمزنگاری تجهیز می‌شود. بنابراین کل ترافیک روی تمام پیوندهای مخابراتی امن خواهند شد. اگرچه در یک شبکهٔ وسیع، این روش نیاز به تعداد زیادی تجهیزات رمزنگاری دارد، ولی در عین حال سطح بالائی از امنیت را ایجاد خواهد کرد. یکی از معایب این روش این است که پیام هر بار که وارد یک سوئیچ بسته‌ای میشود، بایستی رمزگشائی گردد. علت این امر این است که سوئیچ بایستی آدرس موجود در سرآیند بسته (شمارهٔ مدار مجازی) را خوانده تا بتواند آن را مسیریابی نماید. بهمین دلیل پیام از نظر امنیتی در محل سوئیچ آسیب‌پذیر خواهد بود. اگر این شبکه، یک شبکهٔ سوئیچ بسته‌ای همگانی باشد، کاربر کنترلی بر امنیت گره‌ها نخواهد داشت.

در رمزنگاری سر- به- سر، رمزنگاری در دو سیستم انتهائی صورت می‌پذیرد. میزبان یا پایانه منبع، داده‌ها را به رمز در می‌آورد. آنگاه داده‌ها، با فرم رمزنگاری شده و بدون تغییر در عرض شبکه به پایانه و یا میزبان مقصد عبور می‌کند. مقصد با کلیدی که همانند کلید منبع است، پیام را رمزگشائی می‌کند. بنظر میرسد که این روش، انتقال پیام در مقابل حملاتی که به پیوندها و یا سوئیچ‌ها می‌شود را تضمین می‌نماید. با وجود این هنوز یک نقطهٔ سُست باقی است.

حالت زیر را در نظر بگیرید. یک میزبان به یک شبکهٔ سوئیچ بسته‌ای X.25 وصل شده، یک مدار مجازی با میزبان دیگری را برقرار کرده و آماده است تا دیتا را با استفاده از رمزنگاری سر- به- سر برای میزبان دیگر بفرستد. دیتا روی چنین شبکه‌ای بصورت بسته‌هائی منتقل می‌گردد که شامل یک سرآیند و بخش داده‌هاست. میزبان کدام بخش را باید رمزنگاری نماید؟ فرض کنید که میزبان کل بسته که شامل سرآیند نیز هست را رمزنگاری کند. این امر عملی نخواهد بود، زیرا فراموش نکنید که تنها میزبان انتهائی قادر به رمزگشائی است. دراین حالت، گرهٔ سوئیچ بسته‌ای که یک بستهٔ رمزنگاری شده را دریافت می‌کند قادر به خواندن سرآیند آن نبوده و بنابراین نخواهد توانست تا آن را مسیریابی نماید. پس چنین بنظر میرسد که میزبان مبدأ فقط می‌تواند بخش داده‌های بسته دیتا را رمزنگاری کرده و بایستی بخش سرآیند را آزاد گذاشته تا شبکه بتواند با خواندن آن بسته را به مسیر صحیح هدایت کند.

بنابراین در رمزنگاری سر- به- سر، داده‌های کاربر امن می‌مانند. اما چون سرآیند بسته‌ها بصورت ساده منتقل می‌گردند، اُلگوی ترافیک امن نخواهد بود. برای ایجاد امنیت بیشتر، هم رمزنگاری پیوند و هم رمزنگاری سر- به- سر مورد نیازند که این مطلب در شکل ۱۱-۲ نشان داده شده است.

بطور خلاصه، هر وقت از هر دو فرم رمزنگاری استفاده می‌شود، میزبان مبدأ در ابتدا بخش داده‌های کاربر در یک بستهٔ دیتا را با استفاده از یک کلید رمزنگاری سر- به- سر رمزنگاری می‌کند و سپس تمام بسته با استفاده از یک کلید رمزنگاری پیوند به رمز درمی‌آید. همینطور که بسته دیتا در عرض شبکه عبور می‌نماید، هر سوئیچ، بسته را با استفاده از یک کلید رمزنگاری پیوند رمزگشائی کرده تا سرآیند آن را خوانده و سپس مجدداً تمام بسته را برای ارسال روی پیوند بعدی مسیر رمزنگاری می‌نماید. بدین ترتیب تمام یک بستهٔ دیتا، مگر در زمانی که بسته در حافظهٔ یک سوئیچ بسته‌ای قرار داشته، امن است که فقط در آن زمان سرآیند بسته بصورت رمزنشده قابل مشاهده خواهد بود.



شکل ۱۱-۲ رمزنگاری در عرض یک شبکهٔ سوئیچ بسته‌ای

۲-۶ توزیع کلید

برای اینکه رمزنگاری متقارن عملی گردد، طرفین ارتباط بایستی دارای کلید رمز واحدی بوده و این کلید بایستی از دست‌یابی دیگران محافظت گردد. علاوه بر این معمولاً لازم است تا مکرراً کلید را تعویض کرده تا احتمال فاش شدن داده‌ها، در صورت دست‌یابی یک دشمن به کلید، را به حداقل برسانیم. بنابراین قدرت یک سیستم رمزنگاری مرتبط با روش توزیع کلید است. اصطلاح «توزیع کلید» به روش تحویل کلید به دو طرفی اشاره می‌کندکه تمایل به مبادلهٔ دیتا دارند، بدون اینکه دیگران بتوانند کلید را مشاهده نمایند. توزیع کلید را میتوان به چند صورت انجام داد. برای دو طرف A و B:

۱- کلید میتواند بتوسط A انتخاب شده و بصورت فیزیکی به B تحویل گردد.

۲- شخص ثالثی میتواند کلید را انتخاب کرده و بصورت فیزیکی آن را به A و B تحویل دهد.

۳- اگر A و B قبلاً و اخیراً از کلیدی استفاده می‌کرده‌اند، یکی از طرفین میتواند کلید جدید را انتخاب کرده و آن را بصورت رمزنگاری شده با استفاده از کلید قدیم، به طرف دیگر تحویل دهد.

۴- اگر A و B هرکدام یک ارتباط رمزنگاری شده با شخص ثالث C دارند، C می‌تواند یک کلید را از طریق پیوندهای رمزنگاری شده با A و B به آنها تحویل دهد.

روش‌های ۱ و ۲ نیاز به تحویل دستی یک کلید دارند. در رمزنگاری پیوند، این یک نیاز معقول است زیرا هر دستگاه رمزنگاری پیوند تنها می‌خواهد داده‌ها را با شریک خود در طرف دیگر پیوند مبادله نماید. ولی در رمزنگاری سر- به- سر تحویل دستی غیرمعقول است. در یک سیستم توزیع شده، هر میزبان و یا پایانه ممکن است نیاز داشته باشد تا در مبادلهٔ کلید با میزبانان بسیار دیگری مشارکت کند. بنابراین هر دستگاه نیاز به تعدادی کلید داشته که بایستی در طول زمان بصورت پویائی تولید شوند. این مشکل علی‌الخصوص در یک سیستم توزیع شدهٔ پهناور حادتر است.

روش۳ امکانی است که هم برای رمزنگاری پیوند و هم برای رمزنگاری سر- به- سر وجود دارد ولی اگر یک حمله‌کننده یکبار موفق به دست‌یابی به کلیدی گردد، آنگاه همهٔ کلیدهای بعدی نیز لو خواهند رفت. حتی اگر در مورد کلیدهای رمزنگاری پیوند، تعویض‌های مکرری صورت پذیرد، این کار بایستی بطور دستی انجام شود. در مورد تهیهٔ کلیدها برای رمزنگاری سر- به- سر، روش ۴ دارای ارجحیت بیشتری است.

شکل۱۲-۲ روشی را نشان میدهد که گزینه ۴ برای رمزنگاری سر- به- سر را عملی نموده است. در این شکل از رمزنگاری پیوند صرفنظر شده است. این مورد را میتوان بر حسب نیاز اضافه کرد و یا نکرد. در این روش دو نوع کلید تعریف شده است:

- **کلید اجلاس**: وقتی دو سیستم انتهائی (میزبان، پایانه و غیره) تمایل به ارتباط دارند، آنها یک اتصال منطقی (مثل مدار مجازی) را ایجاد می‌کنند. در خلال تداوم اتصال منطقی، تمام داده‌های کاربر با یک کلید اجلاس یکبارمصرف رمزنگاری می‌شود. در خاتمهٔ گفتگو، کلید اجلاس معدوم می‌گردد.
- **کلید دائم**: یک کلید دائم کلیدی است که برای توزیع کلیدهای اجلاس بین واحدها بکار میرود.

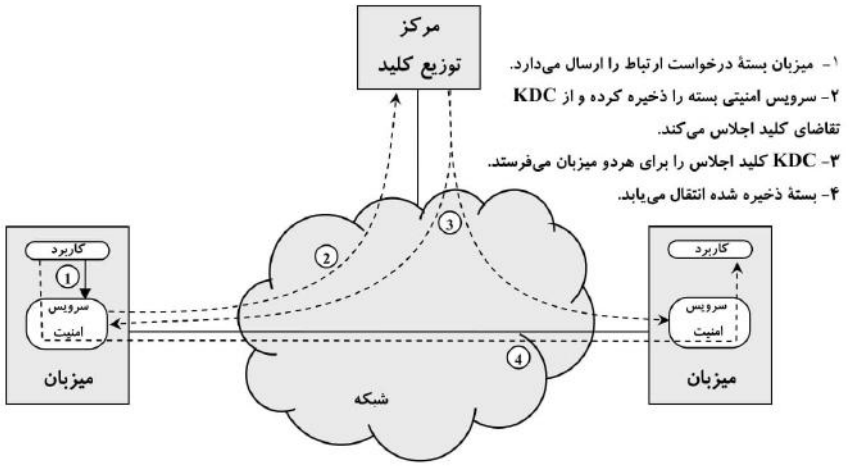
پیکربندی شکل ۱۲-۲ شامل عناصر زیر است:

- **مرکز توزیع کلید(KDC)**: مرکز توزیع کلید تعیین می‌کند که کدام سیستم‌ها مجاز به ارتباط با یکدیگرند. وقتی به دو سیستم اجازه داده شد تا با هم ارتباط یابند، مرکز توزیع کلید یک کلید اجلاس یکبارمصرف را برای این ارتباط فراهم می‌آورد.
- **مدول سرویس امنیتی (SSM)**: این مدول که ممکن است شامل عملکرد در یک لایه پروتکلی باشد، رمزنگاری سر- به- سر را انجام داده و کلید اجلاس را از جانب کاربران بدست می‌آورد.

قدم‌هائی که برای برقراری ارتباط برداشته می‌شود در شکل ۱۲-۲ نشان داده شده است. وقتی یک میزبان می‌خواهد تا ارتباطی با میزبان دیگر برقرار سازد، یک بستهٔ درخواست ارتباط را ارسال میکند (قدم اول). SSM آن بسته را ذخیره کرده و از KDC اجازه میخواهد تا ارتباط را برقرار کند (قدم دوم). ارتباط بین SSM و KDC با یک کلید اصلی که تنها در اختیار SSM و KDC است رمزنگاری می‌شود. اگر KDC تقاضای اتصال را بپذیرد، یک کلید اجلاس تهیه کرده و آن را با استفاده از یک کلید یکتای دائم برای هر SSM، به دو SSM ذیربط تحویل می‌دهد (قدم سوم). اکنون SSM متقاضی ارتباط می‌تواند بستهٔ درخواست ارتباط را رها کرده و یک اتصال بین دو سیستم انتهائی برقرار می‌شود (قدم چهارم). تمام داده‌های کاربر که بین دو سیستم انتهائی رد و بدل می‌شوند بتوسط SSMهای ذیربط و با استفاده از کلید اجلاس یکبار مصرف رمزنگاری می‌شوند.

روش توزیع اتوماتیک کلید، انعطاف‌پذیری و پویائی لازم برای ارتباط تعدادی پایانه با تعدادی میزبان و همچنین ارتباط میزبان‌ها برای مبادلهٔ داده‌ها با یکدیگر را فراهم می‌آورد.

روش دیگری برای توزیع کلید، استفاده از رمزنگاری کلید- عمومی است که در فصل سوم مورد بحث قرار گرفته است.



شکل ۱۲-۲ توزیع اتوماتیک کلید برای پروتکل با گرایش اتصالی

۲-۷ منابع مطالعاتی

عناوین این فصل با جزئیات بیشتری در [STAL06a] پوشش داده شده است. در زمینهٔ الگوریتم‌های رمزنگاری، [SCHN96] یک مرجع کامل است که تقریباً تمام الگوریتم‌ها و پروتکل‌های رمزنگاری که تا زمان نشر این کتاب منتشر شده‌اند در آن توصیف شده است. یکی دیگر از بررسی‌های دقیق و ارزشمند [MENE97] است. یک نگرش عمیق‌تر بهمراه بحث‌های مفصل ریاضی در [STIN06] آمده است.

MENE97 Menezes, A.; van Oorshoot, P.; and Vanstone, S. *Handbook of Applied Cryptography.* Boca Raton, FL: CRC Press, 1997.
SCHN96 Schneier, B. *Applied Cryptography*. New York: Wiley, 1996.
STAL06a Stallings, W. *Cryptography and Network Security: Principles and Practice, Fourth Edition.* Upper Saddle River, NJ: Prentice Hall, 2006.
STIN06 Stinson, D. *Cryptography: Theory and Practice*. Boca Raton, FL: CRC Press, 2006.

وِب سایت‌های مفید

- **AES home page**: صفحهٔ NIST در مورد AES، شامل استاندارد و یک سری اسناد مرتبط دیگر است.
- **AES Lounge**: شامل یک فهرست مفصل از اسناد و مقاله‌ها در مورد AES با قابلیت کپی برداشتن الکترونیک از آنهاست.
- **Block Cipher Modes of Operation**: صفحهٔ NIST با اطلاعات کاملی در مورد مُودهای مورد تأئید NIST.

۲-۸ واژه‌های کلیدی، سؤالات مرورکنندهٔ بحث و مسائل

واژه‌های کلیدی

رمزنگاری	encryption	استاندارد رمزنگاری پیشرفته	Advanced Encyption Standard (AES)
رمزنگاری سر- به- سر	end-to-end encryption	رمز قالبی	block cipher
رمز Feistel	Feistel cipher	حملهٔ همه جانبه	brute-force attack
توزیع کلید	key distribution	مُود زنجیره‌ای رمز قالبی	cipher block chaining (CBC) mode
رمزنگاری پیوند	link encryption	مود فیدبک رمز	cipher feedback (CFB) mode
متن ساده	plaintext	متن رمزشده	ciphertext
کلید اجلاس	session key	شکستن رمز-کشف رمز	cryptoanalysis
رمز دنباله‌ای	stream cipher	رمزنگاری	cryptography
زیرکلید	subkey	استاندارد رمزنگاری دیتا	Data Encryption Standard (DES)
رمزنگاری متقارن	symmetric encryption	رمزگشائی	decryption
DES سه‌گانه	triple DES (3DES)	مُود کتاب کُد الکترونیک	electronic codebook (ECB) mode

سوالات مرورکننده بحث

۲-۱ اجزاء ضروری یک رمز متقارن کدامند؟

۲-۲ دو عمل اساسی که در الگوریتم‌های رمزنگاری از آنها استفاده می‌شود، کدامند؟

۲-۳ برای اینکه دو نفر از طریق یک رمز متقارن با هم ارتباط یابند، چند کلید لازم است؟

۲-۴ اختلاف بین یک رمز قالبی با یک رمز دنباله‌ای چیست؟

۲-۵ دو روش معمول حمله به رمز کدامند؟

۲-۶ چرا بعضی از مُودهای عملیاتی رمز قالبی تنها از رمزنگاری استفاده کرده در حالی که برخی دیگر هم از رمزنگاری و هم از رمزگشائی استفاده می‌کنند؟

۲-۷ رمزنگاری سه‌گانه چیست؟

۲-۸ چرا بخش میانی 3DES رمزگشائی است و رمزنگاری نیست؟

۲-۹ اختلاف بین رمزنگاری پیوند و رمزنگاری سر- به- سر در چیست؟

۲-۱۰ راه‌های ممکن برای توزیع یک کلید سرّی بین دو واحد مرتبط را نام ببرید.

۲-۱۱ اختلاف یک کلید اصلی با یک کلید اجلاس در چیست؟

۲-۱۲ یک مرکز توزیع کلید چیست؟

مسائل

۲-۱ نشان دهید که رمزگشائی Feistel عکس رمزنگاری Feistel است.

۲-۲ کدام اندازهٔ کلید RC4، بردار حالت S در مرحلهٔ آغازیدن را تغییر نخواهد داد؟ یعنی بعد از جایگشت اولیهٔ S، مؤلفه‌های S برابر مقادیر صفر تا ۲۵۵ بصورت صعودی خواهند بود.

۲-۳ RC4 دارای یک حالت داخلی سرّی است که جایگشت تمام مقادیر ممکن بردار S و دو اندیس i و j است.

الف- با استفاده از یک روش سرراست برای ذخیره نمودن حالت داخلی، چه تعداد بیت مورد استفاده قرار می‌گیرد؟

ب - فرض کنید که به این مسأله با این دید نگاه کنیم که چه مقدار اطلاعات با این حالت مرتبط است. در این صورت لازم است تعیین کنیم که چند حالت مختلف وجود دارد و آنگاه لگاریتم این عدد در مبنای ۲ را حساب کرده تا دریابیم که چند بیت اطلاعات نشان‌دهندهٔ این حالت است. با این روش چند بیت لازم است تا حالت را نشان دهد.

۲-۴ در مُود ECB، اگر خطائی در یک بلوک متن رمزشدهٔ ارسال شده بوجود آید، تنها بلوک متن سادهٔ نظیرآن تحت تأثیر واقع می‌شود. در حالی‌که در مُود CBC، این خطا منتشر می‌شود. بعنوان مثال، یک خطا در C_1 ارسال شده (شکل ۹-۲) بطور آشکار P_1 و P_2 را خراب میکند.

الف - آیا بلوک دیگری بجز P_2 تحت تأثیر خطا قرار می‌گیرد؟

ب - فرض کنید که یک خطا در نسخهٔ ابتدائی P_1 وجود دارد. این خطا در چند بلوک متن رمزشده منتشر می‌شود ؟ اثر این امر در گیرنده چه خواهد بود؟

۲-۵ CBC-Pad یک مود عملیاتی رمز قالبی است که در رمز قالبی RC5 بکار می‌رود ولی می‌تواند در هر رمز قالبی دیگر نیز از آن استفاده شود. CBC-Pad می‌تواند به هر متن ساده با هر طولی اعمال گردد. طول متن رمز شده نظیر، حداکثر به اندازهٔ یک بلوک از طول متن ساده بیشتر خواهد بود. برای اینکه طول متن ساده مضربی از طول بلوک گردد، بیت‌های لائی به متن اضافه می‌گردد. فرض می‌شود که متن سادهٔ اولیه مضرب صحیحی از بایت‌هاست. به انتهای این متن ساده از 1 تا bb بایت اضافه می‌شود که bb معادل اندازهٔ بلوک برحسب بایت است. بایت‌های لائی همه یکسان بوده و این مقدار برابر تعداد بایت‌ها لائی است. بعنوان مثال اگر ۸ بایت لائی وجود داشته باشد، هر بایت دارای اندازهٔ 00001000 است. چرا از عدم استفاده از لائی اجتناب می‌شود؟ یعنی اگر طول متن سادهٔ اولیه مضرب صحیحی از اندازهٔ بلوک باشد، چرا بازهم از لائی استفاده می‌شود؟

۲-۶ استفاده از بیت‌های لائی همیشه مناسب نیست. مثلاً ممکن است علاقه‌مند باشیم تا دیتای رمزشده را در همان حافظهٔ موقت که دیتای متن ساده را نگهداری می‌نماید ذخیره کنیم. در این حالت طول دیتای رمزشده بایستی با طول دیتای اولیه برابر باشد. مُود عملیاتی مخصوص این کار مُود CTS (Ciphertext Stealing Mode) نام دارد. شکل ۱۳-۲ الف پیاده‌سازی این مُود را نشان می‌دهد.

الف- طرز عمل این مود را تشریح کنید.

ب- توضیح دهید که C_{n-1} و C_n چگونه رمزگشائی می‌شوند.

۲-۷ شکل ۱۳-۲ب روش دیگری برای تولید یک متن رمزشده با طولی مساوی متن ساده، در حالتی که متن ساده مضرب صحیحی از طول بلوک نیست، را نشان می‌دهد.

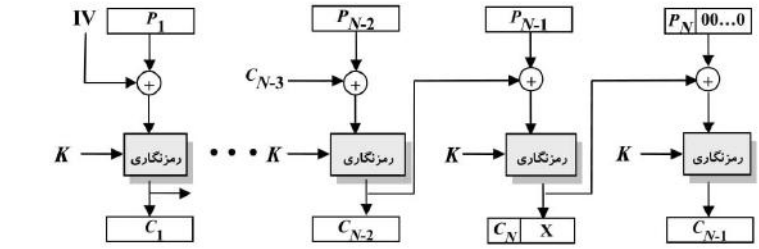
الف- الگوریتم را تشریح کنید.

ب- توضیح دهید که چرا CTS به روشی که در شکل ۱۳-۲ب نشان داده شده است، ارجح است.

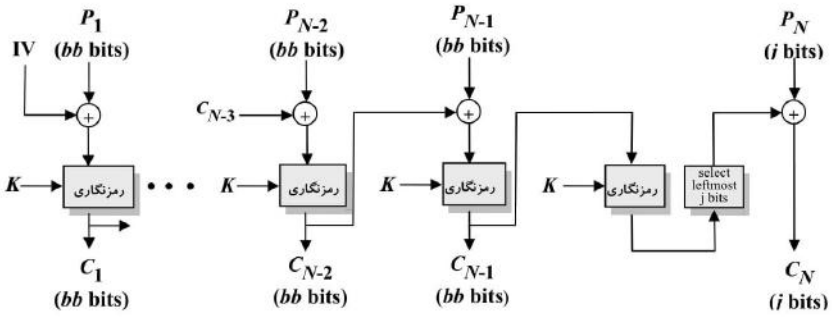
۲-۸ اگر یک بیت خطا در انتقال یک کاراکتر ۸- بیتی رمزشده در مُود CBF اتفاق افتد، این خطا تا چه مسافتی منتشر می‌شود؟

۲-۹ روش‌های توزیع کلید که با استفاده از یک مرکز کنترل دست‌یابی و/ یا مرکز توزیع کلید انجام می‌شوند دارای نقاط آسیب‌پذیر مرکزی‌اند. در مورد امنیت ضمنی چنین تمرکزی بحث نمائید.

۲-۱۰ فرض کنید که فردی راه زیر را برای تأئید اینکه هر دوی شما صاحب یک کلید واحد سرّی می‌باشید پیشنهاد می‌کند. یک دنبالهٔ تصادفی از بیت‌ها با همان طول کلید تولید کرده، آن را با کلید XOR کرده و نتیجه را روی کانال بفرستید. شریک شما، بلوک ورودی را با کلید XOR نموده و آن را پس می‌فرستد. شما بیت‌های دریافتی را کنترل کرده و اگر آنها برابر دنبالهٔ تصادفی اولیه شما بودند تأئید می‌کنید که کلید شریک شما همان کلید شماست و با این روش هیچیک از شما خود کلید را انتقال نداده‌اید. آیا در این روش عیبی وجود دارد؟



(الف) مُود CTS



(ب) روش دیگر

شکل ۱۳-۲ مُودهای رمز قالبی برای متون ساده‌ای که مضرب صحیحی از طول بلوک نیستند

فصل ۳

رمزنگاری کلید- عمومی و اعتبارسنجی پیام

لاوه بر محرمانگی پیام، اعتبارسنجی پیام(message authentication) نیز یک وظیفهٔ مهم امنیتی است. این فصل سه جنبه از اعتبارسنجی پیام را مورد بحث قرار می‌دهد. ابتدا به استفاده از کُدهای اعتبارسنجی پیام و توابع درهم‌ساز (hash functions) برای فراهم آوردن اعتبارسنجی پیام نگاه می‌کنیم. سپس به اصول رمزنگاری کلید- عمومی و دو الگوریتم مخصوص آن نظرمی‌افکنیم. این الگوریتم‌ها در مبادلهٔ کلیدهای رمزنگاری سنّتی مفید هستند. پس از آن به استفاده از رمزنگاری کلید- عمومی برای تولید امضاء دیجیتال توجه می‌کنیم که نوع ارتقاء یافتهٔ اعتبارسنجی پیام است. بالاخره مجدداً نگاهی به مقولهٔ مدیریت کلید می‌اندازیم.

۳-۱ نحوهٔ برخورد با اعتبارسنجی پیام

رمزنگاری، در برابر حملات امنیتی غیرفعال (استراق سمع) حفاظت ایجاد می‌کند. نیاز متفاوت دیگر این است که در برابر حملات فعال (جعل اسناد و داده‌ها)، ایجاد حفاظت نمائیم. حفاظت در مقابل چنین حملهٔ‌هائی را اعتبارسنجی پیام گویند.

یک پیام، فایل، سند، و یا مجموعهٔ دیگری از داده‌ها را وقتی معتبر خوانند که دست اول بوده و از یک منبع قانونی منشأ گرفته باشد. اعتبارسنجی پیام روشی است که به طرفین درگیر در ارتباط اجازه می‌دهد تا معتبر بودن پیام را تأئید نمایند. دو جنبهٔ مهم امر، یکی تحقیق در مورد دست نخورده بودن پیام و دومی معتبر بودن خود منبع است. همچنین ممکن است علاقه‌مند باشیم تا بهنگام بودن پیام (اینکه عمداً به تأخیر نیفتاده و بازخوانی نشده باشد) و یا نظم آن نسبت به سایر پیام‌هائی که بین دوطرف ردوبدل می‌شود را تحقیق کنیم.

اعتبارسنجی با استفاده از رمزنگاری متقارن

اعتبارسنجی را میتوان بسادگی با استفاده از رمزنگاری متقارن انجام داد. اگر فرض کنیم که تنها فرستنده و گیرنده یک کلید مشترک در اختیار دارند (که بایستی چنین باشد)، آنگاه تنها فرستندهٔ واقعی است که قادر به رمزنگاری موفقیت‌آمیز پیام برای طرف مقابل است. علاوه برآن اگر پیام شامل یک کُد تشخیص خطا و یک شمارهٔ ردیف باشد، گیرنده مطمئن خواهد بود که تغییری در پیام رخ نداده و نظم پیام نیز صحیح می‌باشد. همچنین اگر پیام دارای یک برچسب زمانی باشد، گیرنده مطمئن خواهد شد که پیام بیش از حد معقولی که معمولاً شبکه در آن تأخیر ایجاد میکند دیر دریافت نشده است.

اعتبارسنجی پیام بدون رمزنگاری پیام

در این بخش، چند روش اعتبارسنجی پیام که متکی به رمزنگاری نیستند را بررسی می‌کنیم. در تمام این روش‌ها برای انتقال، یک دنبالهٔ (tag) اعتبارسنجی به پیام وصل میشود. خود پیام به رمز درنیامده و می‌تواند مستقل از عمل اعتبارسنجی در مقصد خوانده شود.

نظر به اینکه روش‌های مورد بحث در این بخش پیام را به رمز درنمی‌آورند، محرمانگی پیام فراهم نمی‌گردد. با توجه به اینکه رمزنگاری متقارن اعتبارسنجی را فراهم می‌آورد و با توجه به اینکه با حضور محصولات آمادهٔ موجود، رمزنگاری متقارن بطور گسترده‌ای مورد استفاده قرار می‌گیرد، چرا به سهولت از چنین روشی که هم محرمانگی و هم اعتبارسنجی را ایجاد میکند استفاده نمی‌کنیم؟ [DAVI89] در سه حالت، اعتبارسنجی به دور از محرمانگی را ارجح می‌شمارد:

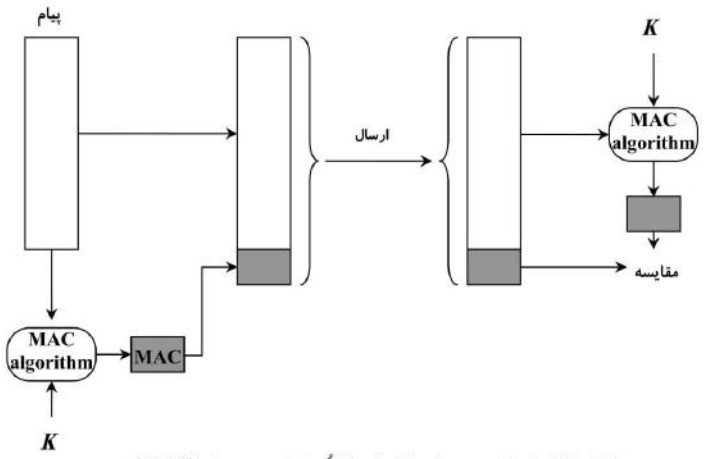
۱- کاربردهائی وجود دارند که در آنها یک پیام به مقاصد متعددی ارسال می‌شود. مثلاً اخطار به کاربران شبکه در مورد اینکه شبکه فعلاً قطع است، و یا آلارم یک مرکز کنترل از این جمله‌اند. در این حالت داشتن تنها یک مقصد مسئول اعتبارسنجی ارزان‌تر و قابل اعتمادتر است. بنابراین پیام بایستی بصورت یک متن ساده بهمراه یک دنبالهٔ اعتبارسنجی پیام پخش شود. اگر نتیجهٔ اعتبارسنجی منفی باشد سیستم مسئول اعتبارسنجی، سیستم‌های دیگر مقصد را بتوسط یک هشدار عمومی خبردار خواهد کرد.
۲- سناریوی ممکن دیگر، تبادل داده‌ها بین دوطرفی است که یکی از آنها بار سنگینی داشته و وقت لازم برای رمزگشائی همهٔ پیام‌های ورودی را ندارد. اعتبارسنجی بر اساس انتخاب نمونه صورت پذیرفته و پیام‌ها بطور تصادفی برای کنترل اعتبار برگزیده می‌شوند.
۳- اعتبارسنجی یک برنامهٔ کامپیوتری با متن ساده، سرویس پرجاذبه‌ای است. برنامهٔ کامپیوتر میتواند بدون اینکه هر بار رمزگشائی شود، اجرا شود که خود صرفه‌جوئی بزرگی در منابع پردازش‌گر است. در عین حال اگر یک دنبالهٔ اعتبارسنجی پیام به برنامه منتقل گردد، میتوان آن را در هر زمان لازم برای اطمینان از اصالت پیام کنترل نمود.

بنابراین برای برآورده نمودن نیازهای امنیتی، جائی برای استفادهٔ توأم از اعتبارسنجی و رمزنگاری وجود دارد.

کُد اعتبارسنجی پیام (MAC)

یکی از روش‌های اعتبارسنجی شامل استفاده از یک کلید سرّی برای تولید یک بلوک کوچک دیتا، بنام کُد اعتبارسنجی پیام (Message Authentication Code) است که به پیام وصل می‌شود. در این روش فرض براین است که دو طرف ارتباط، مثلاً A و B، یک کلید سرّی مشترک مثل K_{AB} را در اختیار دارند. وقتی A دارای پیامی به مقصد B است، کُد اعتبارسنجی پیام را بصورت تابعی از پیام و کلید، محاسبه می‌کند: $\mathrm{MAC}_M = \mathrm{F}(K_{AB}, M)$. پیام باضافهٔ کُد برای گیرندهٔ مورد نظر ارسال می‌شود. گیرنده با استفاده از همان کلید، همان محاسبات را روی پیام ورودی انجام داده و یک کُد اعتبارسنجی جدید بدست می‌آورد. کُد دریافت شده با کُد محاسبه شده مقایسه می‌شود (شکل ۱- ۳). اگر فرض کنیم که تنها گیرنده و فرستنده از کلید سرّی باخبرند و اگر کُد دریافت شده با کُد محاسبه شده تطبیق داشته باشد، آنگاه

۱-گیرنده مطمئن می‌شود که پیام تغییر نیافته است. اگر دشمنی پیام را تغییر داده ولی کُد را تغییر نداده باشد، آنگاه کُد محاسبه شدهٔ گیرنده با کُد دریافت شده فرق خواهد داشت. چون فرض شده است که دشمن از کلید سرّی بی‌خبر است، او نمی‌تواند کُد را طوری تغییر دهد که با تغییرات پیام همخوانی داشته باشد.
۲-گیرنده مطمئن می‌شود که پیام از سوی فرستندهٔ قانونی ارسال شده است. چون کس دیگری از کلید سرّی مطلع نیست، هیچ‌کس نمی‌تواند یک پیام جعلی با کُد صحیح را تهیه نماید.
۳- اگر پیام دارای یک شمارهٔ ردیف باشد (مثل آنچه در X.25 ، HDLC و TCP مورد استفاده است)، آنگاه گیرنده می‌تواند از نظم پیام مطمئن شود زیرا یک دشمن نمی‌تواند شمارهٔ ردیف را بصورت موفقیت‌آمیزی تغییر دهد.



شکل ۱-۳ اعتبارسنجی پیام با استفاده از کُد اعتبارسنجی پیام (MAC)

برای تولید کُد اعتبارسنجی، می‌توان از الگوریتم‌های متعددی استفاده کرد. NIST در مشخصهٔ FIPS PUB 113، استفاده از DES را توصیه می‌کند. DES برای تولید یک نسخهٔ رمزنگاری شده از پیام بکاررفته، و آخرین بیت‌های متن رمزشده بعنوان کُد مورد استفاده قرار می‌گیرد. یک کُد ۱۶ یا ۳۲- بیتی، کُدی مرسوم است.

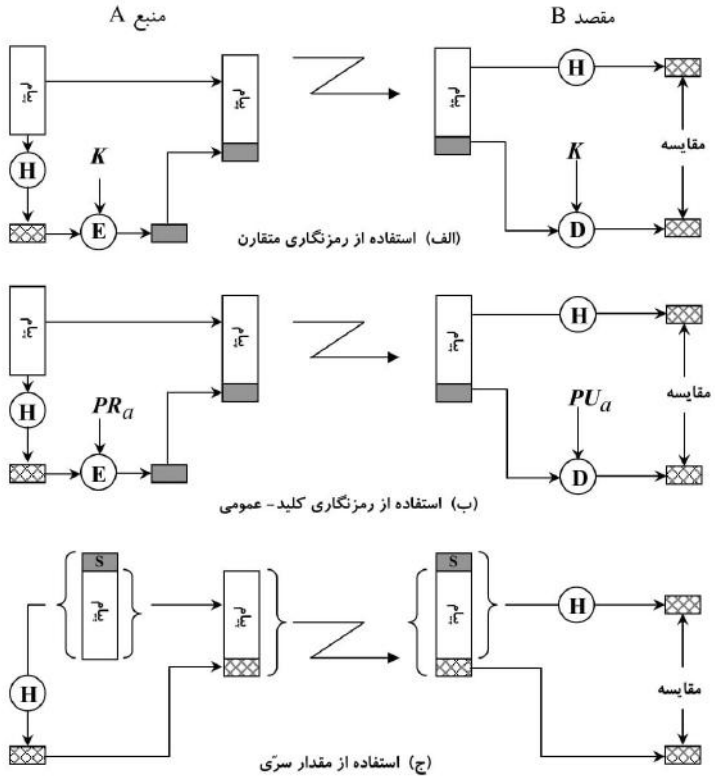
روشی که هم اکنون ذکر شد، شبیه رمزنگاری است. یک اختلاف آن این است که برخلاف آنچه در رمزگشائی لازم است، الگوریتم اعتبارسنجی لازم نیست برگشت‌پذیر باشد. چنین برمی‌آید که بدلیل خواص ریاضی تابع اعتبارسنجی، این تابع در مقابل شکستن رمز آسیب‌پذیری کمتری نسبت به رمزنگاری دارد.

تابع درهم‌ساز یک- طرفه

رقیب دیگری برای کُد اعتبارسنجی پیام، تابع درهم‌ساز یک- طرفه است. همانندکُد اعتبارسنجی پیام، یک تابع درهم‌ساز یک پیام M با طول متغیر بعنوان ورودی را گرفته و یک چکیدهٔ (digest) پیام $H(M)$ با طول ثابت را خارج می‌کند. برخلاف MAC، یک تابع درهم‌ساز یک کلید سرّی را بعنوان ورودی نمی‌خواهد. برای اعتبارسنجی پیام، چکیدهٔ پیام بصورتی به همراه پیام ارسال می‌شود که چکیده معتبر بماند.

شکل ۲-۳ سه روش که پیام می‌تواند بتوسط آنها اعتبارسنجی شود را نشان می‌دهد. چکیدهٔ پیام می‌تواند با استفاده از رمزنگاری متقارن (بخش الف) به رمز درآید. اگر فرض شود که تنها فرستنده و گیرنده کلید سرّی را در اشتراک دارند، آنگاه اعتبارسنجی محرز است. پیام همچنین می‌تواند با استفاده از رمزنگاری کلید- عمومی به رمز درآید (بخش ب) که این مقوله در بخش ۵-۳ توضیح داده شده است. روش کلید- عمومی دارای دو مزیت است: اول اینکه علاوه بر اعتبارسنجی پیام، یک امضاء دیجیتال نیز تولید می‌شود و دوم اینکه نیازی به توزیع کلید بین طرفین ارتباط نیست.

مزیت این دو روش نسبت به روش‌هائی که تمام پیام را رمزنگاری می‌کنند این است که محاسبات کمتری مورد نیاز است. با وجود این علاقهٔ زیادی نسبت به خلق تکنیک‌هائی وجود دارد که بطور کلی از رمزنگاری صرفنظر شود. دلایل متعددی برای این علاقه در [TSUD92] ذکر شده است:



شکل ۲-۳ اعتبارسنجی پیام با استفاده از یک تابع درهم‌ساز یک- طرفه

- نرم‌افزار رمزنگاری تا حد زیادی کند است. اگرچه میزان داده‌هائی که برای هر پیام باید رمزنگاری شود کم است، ولی ممکن است یک جریان پیوسته از پیام‌ها وارد سیستم شده و از آن خارج گردد.
- هزینه‌های سخت‌افزاری رمزنگاری قابل چشم‌پوشی نیست. DES روی تراشه‌های ارزان قیمتی پیاده‌سازی شده است ولی اگر قرار باشد که همهٔ گره‌های یک شبکه قابلیت رمزنگاری را داشته باشند، هزینه بالا خواهد رفت.
- سخت‌افزار رمزنگاری وقتی بهینه‌تر می‌شود که اندازهٔ بلوک‌های دیتا بزرگ‌تر شود. برای بلوک‌های کوچک دیتا، بخش قابل توجهی از زمان برای سربارهای شروع/ فراخوانی صرف خواهد شد.
- یک الگوریتم رمزنگاری ممکن است به ثبت رسیده و استفاده از آن بدون اجازه امکان‌ناپذیر باشد

شکل ۲-۳ج تکنیکی را نشان می‌دهد که از یک تابع درهم‌ساز استفاده شده ولی از رمزنگاری بمنظور اعتبارسنجی پیام استفاده نمی‌کند. این روش فرض می‌کند که دوطرف ارتباط، مثل A و B، یک مقدار سری S_{AB} را در اشتراک دارند. وقتی A

می‌خواهد پیامی را برای B بفرستد، او تابع درهم‌ساز را روی اتصال پیام و مقدار سرّی اعمال می‌کند $MD_M = H(S_{AB} \| M)$ (|| علامت این است که مقادیر دو سمت این علامت پشت سرهم قرار می‌گیرند). A آنگاه $[M \| MD_M]$ را برای B می‌فرستد. چون B، S_{AB} را در اختیار دارد، او می‌تواند $H(S_{AB} \| M)$ را مجدداً محاسبه کرده و MD_M را تأئید کند. چون خود مقدار سری S_{AB} ارسال نمی‌شود، برای یک دشمن میسر نخواهد بود که پیام را دستکاری نماید. همچنین تا زمانی که مقدار سرّی محفوظ باشد، برای دشمن میسر نخواهد بود که یک پیام تقلبی ایجاد کند.

نوع تعدیل شده‌ای از تکنیک سوم، بنام HMAC برای امنیت IP انتخاب شده است (فصل ششم). این روش همچنین برای SNMPv3 (فصل هشتم) نیز تعیین گردیده است.

۳-۲ توابع درهم‌ساز امن و HMAC

تابع درهم‌ساز امن یک- طرفه یا تابع hash امن، نه تنها در اعتبارسنجی پیام بلکه در امضاءهای دیجیتال نیز حائز اهمیت است. این بخش را با بررسی خواص مورد نیاز یک تابع درهم‌ساز امن شروع می‌کنیم. سپس یکی از مهم‌ترین توابع درهم‌ساز یعنی SHA را مورد بررسی قرار می‌دهیم. در پایان HAMC را معرفی خواهیم کرد.

لازمه‌های تابع درهم‌ساز

هدف یک تابع درهم‌ساز این است که یک «اثرانگشت» از یک فایل، یک پیام، و یا بلوک دیگری از دیتا بوجود آورد. برای این که یک تابع درهم‌ساز H به درد اعتبارسنجی پیام بخورد بایستی دارای خواص زیر باشد:

۱- H بتواند به یک بلوک داده با هر اندازه‌ای اعمال گردد.
۲- H یک خروجی با طول ثابت ایجاد کند.
۳- محاسبهٔ $H(x)$ برای هر x نسبتاً ساده بوده و بتوان آن را بصورت سخت‌افزاری و نرم‌افزاری اجرا نمود.
۴- برای هر مقدار h، پیداکردن x بطوری که $H(x) = h$ باشد از نظر محاسباتی مقدور نباشد.
۵- برای هر بلوک x، پیداکردن یک مقدار $y \neq x$ بطوری‌که $H(y) = H(x)$ باشد، از نظر محاسباتی مقدور نباشد. این مورد را گاهی **مقاومت ضعیف در برابر تصادم (weak collision resistance)** نامند.
۶- پیدا کردن زوجی مانند (x,y) بطوری که $H(x) = H(y)$ باشد از نظر محاسباتی مقدور نباشد.این مورد را گاهی **مقاومت قوی در برابر تصادم (strong collision resistance)** نامند.

سه خاصیت اول مربوط به نیازهای کاربردهای عملی یک تابع درهم‌ساز برای اعتبارسنجی پیام است. خاصیت چهارم یک خاصیت «یک- طرفه» است: این ساده است تا کُد مربوط به یک پیام را تولید کرد ولی عملاً محال است که با داشتن کُد، پیام را بدست آورد. این خاصیت در صورتی که تکنیک اعتبارسنجی شامل استفاده از یک مقدار سرّی باشد (شکل۲-۳ج)، مهم است. اگرچه خود مقدار سرّی ارسال نمی‌شود ولی اگر تابع درهم‌ساز یک- طرفه نباشد، یک دشمن بسهولت می‌تواند مقدار سرّی را کشف کند: اگر دشمن بتواند به یک انتقال گوش فراداده و یا آن را مشاهده نماید، او پیام M و کُد درهم شدهٔ $MD_M = H(S_{AB} \| M)$ را به چنگ می‌آورد. دشمن آنگاه تابع درهم‌ساز را معکوس کرده تا $S_{AB} \| M = H^{-1}(MD_M)$ را بدست آورد. حال چون دشمن هم M و هم $S_{AB} \| M$ را در اختیار دارد، بسادگی می‌تواند S_{AB} را استخراج نماید.

خاصیت پنجم تضمین می‌کند که امکان‌پذیر نیست تا پیام دیگری با همان اندازهٔ hash پیام اصلی بدست آورد. این امر از تقلب در زمانی که یک کُد hash رمزشده بکار می‌رود جلوگیری می‌کند (شکل‌های ۲-۳الف و ب). اگر این خاصیت جاری نباشد، یک حمله‌کننده قادر به انجام عملیات زیر خواهد بود: اول، یک پیام باضافهٔ کُد hash رمزنگاری شدهٔ آن را ملاحظه یا استراق سمع کند. دوم، یک کُد hash رمزنشده از پیام را تهیه کند. سوم، یک پیام دیگر با همان کُد hash تهیه نماید.

یک تابع hash که پنج خاصیت اول لیست قبل را داشته باشد، تابع hash ضعیف می‌خوانند. اگر علاوه برآن خاصیت ششم نیز ارضاء گردد، آنگاه تابع hash را قوی می‌نامند. خاصیت ششم از یک دسته حملات پیچیده که حملهٔ روز تولد خوانده می‌شوند، جلوگیری می‌نماید. جزئیات این حمله فراتر از افق دید این کتاب است. این حمله توانائی یک تابع درهم‌ساز m- بیتی را از 2^m به $2^{m/2}$ کاهش می‌دهد. [YUVA79] و یا [STAL06a] را ملاحظه نمائید.

علاوه بر فراهم نمودن اعتبارسنجی، یک چکیدهٔ پیام صحت پیام را نیز تضمین می‌کند. عمل آن همانند دنبالهٔ کنترل فریم (FCS) است: اگر هر بیتی از پیام در حین انتقال بطور تصادفی تغییر نماید، چکیدهٔ پیام عوض خواهد شد.

توابع درهم‌ساز ساده

تمام توابع درهم‌ساز بر اساس اصول عمومی زیر کار می‌کنند. ورودی (پیام، فایل، و غیره) بصورت دنباله‌ای از بلوک‌های n- بیتی تلقی می‌گردد. ورودی بصورت یک بلوک در هر زمان و به روش تکرار مورد پردازش قرار گرفته تا یک تابع درهم‌سازی شدهٔ n- بیتی را تولید کند.

یکی از ساده‌ترین توابع درهم‌سازی، XOR کردن بیت- به- بیت هر بلوک است. این را می‌توان بصورت زیر نشان داد:

$$C_i = b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{im}$$

که در آن

C_i - امین بیت کُد hash $1 \le i \le n$

m - تعداد بلوک‌های n- بیتی در ورودی

b_{ij} = i امین بیت در j امین بلوک

$\oplus$ - عمل XOR

شکل ۳-۳ این عملیات را نشان می‌دهد. هر بیت کُد hash بسادگی یک بیت توازن (parity) بوده که تست افزونگی عمودی (VRC) نیز خوانده می‌شود. این عمل برای داده‌های تصادفی بعنوان یک تست صحت دیتا کاملاً مؤثر است. هریک از مقادیر n- بیتی کُد درهم‌سازی شده بطور مساوی محتمل‌اند. بنابراین احتمال اینکه یک خطا در دیتا باعث تغییرنکردن کُد hash آن شود، 2^{-n} است. اگر فرمت دیتا قابل پیش‌بینی‌تر باشد، تابع کمتر مؤثر خواهد بود. مثلاً در بیشتر فایل‌های متنی نرمال، بیت پرارزش هراُکتت همیشه صفر است. بنابراین اگر از یک مقدار ۱۲۸- بیتی hash استفاده شود، بجای اینکه تأثیر آن $۲^{-۱۲۸}$ باشد، تابع hash چنین دیتائی دارای تأثیری برابر $۲^{-۱۱۲}$ خواهد بود.

راه ساده‌ای برای بهبود کار این است که پس از پردازش هربلوک، یک گردش و یا شیفت حلقوی باندازهٔ یک بیت روی مقدار hash انجام دهیم. روش را می‌توان بصورت زیر خلاصه کرد:

۱- در ابتدا مقدار n- بیتی hash را برابر صفر قرار دهید.

۲- هر بلوک n- بیتی دیتا را پس از آن متوالیاً به روش زیر پردازش نمائید:

الف- مقدار جاری hash را یک بیت به سمت چپ بچرخانید.

ب- بلوک را با مقدار hash XOR نمائید.

اثر این امر «تصادفی»تر کردن و غلبه بر هر نوع نظمی است که در ورودی وجود دارد.

اگرچه روش دوم معیار خوبی برای سنجش صحت دیتا را فراهم می‌آورد، ولی وقتی همانند شکل‌های ۲-۳الف و ب یک کُد hash رمزنگاری شده بهمراه یک پیام متن ساده بکار می‌رود، از نظر امنیت داده‌ها بی‌ارزش است. برای یک پیام داده شده، کار آسانی است تا پیام جدیدی که همان کُد hash را تولید می‌کند خلق کرد: بسادگی پیام مورد نظر جدید را تهیه کرده و آنگاه یک بلوک n- بیتی که باعث شود تا پیام جدید باضافهٔ بلوک، کُد hash مطلوب را ایجاد نماید به آن اضافه کنید.

اگرچه یک XOR ساده و یا یک XOR چرخش یافته(RXOR) در حالتی که فقط کُد hash رمزنگاری می‌شود کافی نیست، ولی ممکن است هنوز تصور کنید که در صورتی‌که هم پیام و هم کُد hash دنبال آن رمزنگاری گردند، چنین تابع ساده‌ای می‌تواند مفید واقع شود. اما بایستی دقت کرد. تکنیکی که در ابتدا بتوسط دایرهٔ ملی استانداردهای آمریکا پیشنهاد گردید از یک XOR ساده که به بلوک‌های ۶۴- بیتی پیام اعمال می‌شد و سپس از مُود زنجیره‌ای رمز قالبی (CBC) استفاده می‌کرد، تشکیل شده بود. روش را می‌توان چنین توصیف کرد: با داشتن پیامی که شامل دنباله‌ای از بلوک‌های ۶۴- بیتی $X_1, X_2, \ldots, X_n$ است، کُد hash آن را بصورت XOR بلوک- به- بلوک تمام بلوک‌ها تعریف کرده و کُد hash را بعنوان آخرین بلوک به پیام وصل کنید:

$$C = X_{N+1} = X_1 \oplus X_2 \oplus \ldots \oplus X_N$$

سپس تمام پیام بعلاوهٔ کُد hash را با استفاده از مُود CBC رمزنگاری کرده تا پیام رمزنگاری شده $Y_1, Y_2, \ldots, Y_{N+1}$ بدست آید. [JUEN85] به چندین راه که در آنها متن رمزشدهٔ این پیام می‌تواند طوری دستکاری شده که بتوسط کُد hash قابل آشکارسازی نباشد، اشاره می‌کند. بعنوان مثال، برحسب تعریف CBC (شکل ۹-۲) داریم

$$X_1 = IV \oplus D(K, Y_1)$$
$$X_i = Y_{i-1} \oplus D(K, Y_i)$$
$$X_{N+1} = Y_N \oplus D(K, Y_{N+1})$$

	بیت 1	بیت 2	• • •	بیت n
بلوک 1	b_{11}	b_{21}		b_{n1}
بلوک 2	b_{12}	b_{22}		b_{n2}
	•	•	•	•
	•	•	•	•
	•	•	•	•
بلوک m	b_{1m}	b_{2m}		b_{nm}
کُد Hash	C_1	C_2		C_n

شکل ۳-۳ تابع hash ساده با استفاده از XOR بیت- به- بیت

اما X_{N+1} کُد hash است:

$$X_{N+1} = X_1 \oplus X_2 \oplus \ldots \oplus X_N$$
$$= [IV \oplus D(K, Y_1)] \oplus [Y_1 \oplus D(K, Y_2)] \oplus \ldots \oplus [Y_{N-1} \oplus D(K, Y_N)]$$

چون عبارات جملهٔ قبل را می‌توان با هر نظمی XOR نمود، نتیجه می‌گیریم که اگر بلوک‌های متن رمزشده تحت هر جایگشتی قرار گیرند، کُد hash تغییر نخواهد کرد.

تابع درهم‌ساز امن SHA-1

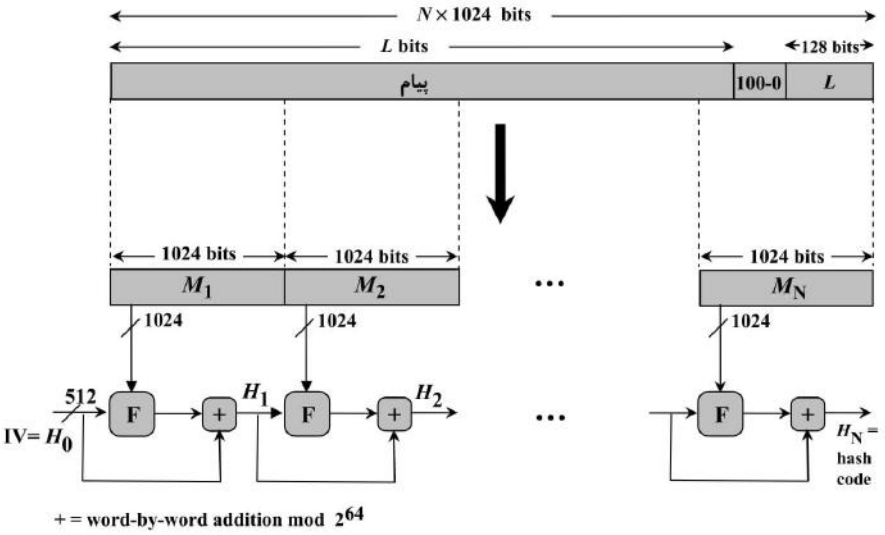
الگوریتم درهم‌ساز امن Secure Hash Algorithm (SHA) بتوسط انستیتوی ملی استانداردها و تکنولوژی آمریکا (NIST) تولید و بعنوان یک استاندارد فدرال پردازش اطلاعات (FIPS 180) در سال ۱۹۹۳ منتشر گردید. نسخهٔ بازنگری شدهٔ آن بعنوان FIPS 180-1 در سال ۱۹۹۵ منتشر شد و معمولاً از آن با نام SHA-1 یاد می‌شود. SHA-1 همچنین در RFC 3174 تعریف شده است که عموماً شامل همان تعاریف FIPS 180-1 بوده ولی یک کُد به زبان C نیز به آن اضافه شده است.

SHA-1 یک اندازهٔ hash با طول ۱۶۰ بیت ایجاد می‌کند. در سال ۲۰۰۲، NIST یک نسخهٔ جدید از استاندارد، FIPS 180-2 را تولید کرد که سه نسخهٔ جدید از SHA با طول‌های ۲۵۶، ۳۸۴ و ۵۱۲ بیت برای اندازهٔ hash را تعریف کرده است. این توابع SHA-256، SHA-384 و SHA-512 خوانده می‌شوند (جدول ۱-۳). این نسخه‌های جدید همان ساختار زیربنائی را داشته و از همان حساب پیمانه‌ای و عملیات منطقی باینری SHA-1 استفاده می‌کنند. در سال ۲۰۰۵، NIST قصد خود نسبت به کنارگذاشتن SHA-1 و اتکاء بر سایر نسخه‌های SHA تا سال ۲۰۱۰ را اعلام کرد. کمی بعد از آن، یک تیم تحقیقاتی حمله‌ای را توصیف کرد که درآن با استفاده از 2^{69} عملیات می‌توان دو پیام را پیدا کرد که دارای اندازهٔ hash یکسان باشند. این تعداد عملیات بسیار کمتر از 2^{80} عملیاتی است که تصور می‌شد برای پیداکردن یک تصادم در SHA-1 لازم است [WANG05]. این نتیجه نشان می‌دهد که گذر به سایر نسخ SHA بایستی سریع‌تر صورت پذیرد.

در این بخش توصیفی از SHA-512 را ارائه می‌دهیم. سایر نسخه‌ها کاملاً مشابه این نسخه‌اند. الگوریتم بعنوان ورودی یک پیام با ماکزیمم طول کمتر از 2^{128} را گرفته و یک چکیدهٔ پیام ۵۱۲- بیتی را تولید می‌کند. ورودی در بلوک‌های ۱۰۲۴- بیتی مورد پردازش قرار می‌گیرد. شکل ۴-۳ پردازش کامل یک پیام برای تولید یک چکیده را نشان می‌دهد. پردازش شامل قدم‌های زیر است:

جدول ۱-۳ مقایسهٔ پارامترهای SHA

	SHA-1	SHA-256	SHA-384	SHA-512
اندازهٔ چکیدهٔ پیام	160	256	384	512
اندازهٔ پیام	$<2^{64}$	$<2^{64}$	$<2^{128}$	$<2^{128}$
اندازهٔ بلوک	512	512	1024	1024
اندازهٔ کلمه	32	32	64	64
تعداد قدم‌ها	80	64	80	80
امنیت	80	128	192	256



شکل۴-۳ تولید چکیدهٔ پیام با استفاده SHA-1

- **قدم اول: بیت‌های لائی (padding) را به پیام وصل کنید.** بیت‌های لائی طوری به بیت‌های پیام اضافه می‌شوند که طول آن 1024 modulo 896 گردد. بیت‌های لائی همیشه به پیام اضافه می‌گردند حتی اگر پیام خود دارای طول مطلوب باشد. بنابراین تعداد بیت‌های لائی بین ۱ تا ۱,۰۲۴ خواهد بود. لائی شامل یک بیت 1 و بدنبال آن تعداد لازم بیت‌های 0 است.
- **قدم دوم: طول پیام را وصل کنید.** یک بلوک ۱۲۸- بیتی به انتهای پیام وصل می‌شود. این بلوک بصورت یک عدد صحیح ۱۲۸- بیتی بدون علامت (با ارزش‌ترین بایت در ابتدا واقع می‌شود) بوده و شامل طول پیام اولیه (قبل از اضافه شدن) است.

اجرای دو قدم اول، پیامی را بدست می‌دهد که طول آن مضرب صحیحی از ۱,۰۲۴ بیت است. در شکل ۴-۳ پیام توسعه‌یافته بصورت یک دنباله از بلوک‌های ۱,۰۲۴ بیتی $M_1, M_2, ..., M_N$ نشان داده شده بطوری‌که طول کلی پیام توسعه‌یافته $N \times 1024$ بیت است.

- **قدم سوم: حافظهٔ hash را با مقادیر اولیه پرکنید.** یک حافظهٔ موقت ۵۱۲- بیتی برای نگهداری مقادیر میانی و انتهائی تابع hash بکار می‌رود. این حافظهٔ موقت می‌تواند بصورت ۸ رجیستر ۶۴- بیتی (a, b, c, d, e, f, g, h) نشان داده شود. این حافظه‌های موقت در ابتدا با اعداد صحیح ۶۴- بیتی زیر (بصورت هکزادسیمال) پر می‌شوند:

a = 6A09E667F3BCC908	e = 510E527FADE682D1
b = BB67AE8584CAA73B	f = 9B05688C2B3E6C1F
c = 3C6EF372FE94F82B	g = 1F83D9ABFB41BD6B
d = A54FF53A5F1D36F1	h = 5BE0CDI9137E2179

این مقادیر با فرم big-endian ذخیره می‌شوند که در آن با اهمیت‌ترین بایت یک کلمه، در محل پائین‌ترین (چپ‌ترین) آدرس بایت قرارمی‌گیرد. این کلمات با انتخاب۶۴ بیت اول بخش‌های اعشاری جذر اولین هشت عدد اول بدست آمده‌اند.

- **قدم چهارم: پیام را در بلوک‌های ۱۰۲۴- بیتی (۱۲۸-کلمه‌ای) پردازش کنید.** قلب الگوریتم مدولی است که شامل ۸۰ دُور پردازش است. این مدول درشکل ۴-۳ با F نشان داده شده است. منطق عمل درشکل ۵-۳ رسم شده است.

هر دُور بعنوان ورودی، یک بلوک ۵۱۲- بیتی abcdefgh حافظه موقت را گرفته و محتویات حافظهٔ موقت را به روز درمی‌آورد. بعنوان ورودی اولین دُور، حافظهٔ موقت دارای اندازهٔ hash میانی H_{i-1} است هر دور t از یک اندازهٔ ۶۴- بیتی W_t استفاده می‌کند که از بلوک ۱۰۲۴- بیتی در حال پردازش (M_i) مشتق شده است. هر دُور همچنین از یک ثابت جمع شونده K_t استفاده کرده که درآن $0 \le t \le 79$ نمایش دهندهٔ یکی از ۸۰ دُور است. این کلمات اولین ۶۴ بیت بخش‌های اعشاری ریشهٔ سوم اولین هشت عدد اول می‌باشند ثابت‌ها یک الگوی تصادفی ۶۴- بیتی را ایجاد می‌کنند که قاعدتاً هرگونه نظم در دیتای ورودی را از بین خواهد برد.

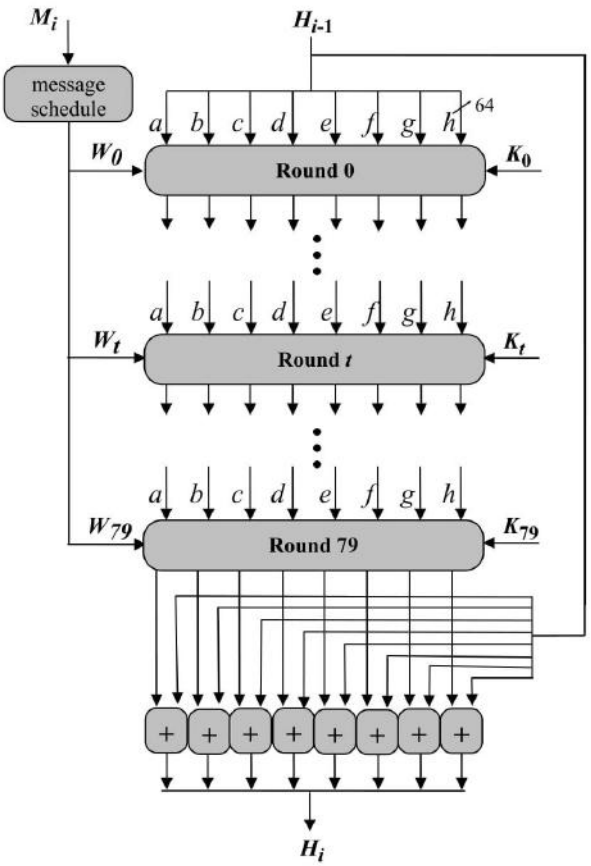
خروجی هشتادمین دُور به ورودی اولین دُور (H_{i-1}) اضافه شده تا H_i را تولید کند. جمع برای هریک از ۸ کلمهٔ موجود در حافظهٔ موقت با هر کلمهٔ نظیر در H_{i-1} جداگانه و بصورت modulo 2^{64} انجام می‌شود.

- **قدم پنجم: خروجی.** بعدازاینکه تمام N بلوک ۱۰۲۴- بیتی پردازش گردید، خروجی N امین مرحله چکیدهٔ ۵۱۲- بیتی پیام را تشکیل می‌دهد.

الگوریتم SHA-512 دارای این خاصیت است که هر بیت کُد hash تابعی از تمام بیت‌های ورودی است. تکرار پیچیدهٔ تابع اصلی F نتایجی را تولید می‌کند که بخوبی مخلوط شده‌اند. یعنی غیرمحتمل است که دو پیامی که بصورت تصادفی انتخاب شده‌اند، حتی اگر نظم مشابه‌ای از خود نشان دهند، دارای همان کُد hash یکسان باشند. مگر ضعف پنهان شده‌ای در SHA-512 وجود داشته باشد که تا کنون هویدا نشده باشد والا مسألهٔ رسیدن به دو پیامی که دارای چکیدهٔ یکسانی باشند نیاز به عملیاتی با حجم 2^{256} دارد. به همین ترتیب مورد پیداکردن پیامی با یک چکیدهٔ داده شده، نیاز به 2^{512} عملیات دارد.

سایر توابع درهم‌ساز امن

همانگونه که در مورد رمزهای قالبی متقارن نیز چنین بود، طراحان توابع درهم‌ساز امن تمایلی نداشته‌اند که از یک ساختار به اثبات رسیده عدول نمایند. DES بر مبنای رمز Feistel بناشده است. همهٔ رمزهای قالبی پس از آن نیز تلویحاً از طراحی Feistel تبعیت کرده‌اند، زیرا این طراحی را می‌توان طوری تغییر داد که در برابر تهدیدهای کشف رمزی که جدیداً کشف شده نیز مقاوم گردد. اگر بجای آن از یک طراحی کاملاً نو برای یک رمز قالبی متقارن استفاده می‌شد، این نگرانی وجود داشت که ساختار جدید راه‌های حملهٔ جدیدی که تا کنون نسبت به آن فکر نشده بود را بگشاید. بدلیل مشابه، بیشتر توابع درهم‌ساز جدید و مهم نیز از همان ساختار اساسی شکل ۴-۳ که تابع درهم‌ساز تکرارشونده خوانده شده و در ابتدا بتوسط Merkle[MERK79,MERK89] پیشنهاد شده است تبعیت می‌کنند. محرک این ساختار تکرارشونده از مشاهدات Merkle[MERK89] و Damgard[DAMG89] سرچشمه می‌گیرد که اگر تابع مربوط به یک بلوک منفرد، که **تابع فشرده‌سازی** خوانده می‌شود، در مقابل تصادم مقاوم باشد، نتیجهٔ تابع درهم‌شده و تکرارشدهٔ آن هم، همان خاصیت را خواهد داشت. بنابراین ساختار را می‌توان برای تولید یک تابع درهم‌ساز امن، برای عمل روی پیامی با هر طول بکار برد. مسألهٔ طراحی یک تابع درهم‌ساز امن به مسألهٔ طراحی یک تابع فشرده‌ساز مقاوم در مقابل تصادم که روی ورودی‌هائی با طول ثابت کار می‌کنند، برمی‌گردد. ثابت شده است که این برخورد، روش مطمئنی است و طرح‌های جدیدتر تنها ساختار را پالایش کرده و به طول کُد افزوده‌اند.



شکل ۵-۳ پردازش SHA-512 یک بلوک ۱۰۲۴- بیتی

در این بخش نگاهی به دو تابع درهم‌ساز امن دیگر، که علاوه بر SHA پذیرش تجاری یافته‌اند، می‌اندازیم.

الگوریتم چکیدهٔ پیام MD5

الگوریتم چکیدهٔ پیام MD5 (RFC 1321) بتوسط Ron Rivest طراحی گردید. تا چندین سال قبل که هنوز نگرانی‌های مربوط به حملهٔ همه جانبه و کشف رمز جدی نبود، MD5 پراستفاده‌ترین الگوریتم درهم‌سازی امن بود. الگوریتم بعنوان ورودی، یک پیام با طول اختیاری را پذیرفته و بعنوان خروجی یک چکیدهٔ ۱۲۸- بیتی از پیام را تولید می‌کند. ورودی بصورت بلوک‌های ۵۱۲- بیتی مورد پردازش قرار می‌گیرد.

همینطور که سرعت پردازش‌گرها افزایش یافته است، امنیت یک کُد ۱۲۸- بیتی نیز زیر سؤال رفته است. میتوان نشان دادکه پیچیدگی رسیدن به دو پیام متفاوت که دارای یک چکیده باشند در مرز 2^{64} عملیات قرار داشته در حالی‌که پیچیدگی پیداکردن یک پیام با یک چکیدهٔ مورد نظر در حد 2^{128} عملیات است. رقم قبلی برای امنیت خیلی کوچک است. علاوه برآن تعدادی عملیات شکستن رمز صورت پذیرفته است که آسیب‌پذیری MD5 در مقابل کشف رمز را نشان می‌دهد [BERS92,BOER93,DOBB96].

Whirlpool

Whirlpool [BARR03,STAL06b] بتوسط Vincent Rijmen اهل بلژیک و سهیم در اختراع الگوریتم Rijndael (الگوریتمی که بعنوان استاندارد پیشرفتهٔ رمزنگاری AES پذیرفته شد) و Paulo Barreto که یک رمزنگار برزیلی است طراحی شده است. Whirlpool یکی از تنها دو تابع درهم‌سازی است که مورد تأئید NESSIE (New European Schemes for Signatures, Integrity, and Encryption) می‌باشد [PREN02]. پروژهٔ NESSIE یک تلاش اتحادیهٔ اروپا برای فراهم کردن یک مجموعه از توابع نیرومند و متنوع رمزنگاری است که شامل رمزهای قالبی، رمزهای دنباله‌ای، رمزهای متقارن، توابع درهم‌ساز و کُدهای اعتبارسنجی پیام می‌باشد.

Whirlpool بر مبنای استفاده از یک رمزقالبی برای تابع فشرده‌سازی قرار دارد. Whirpool از یک رمز قالبی استفاده می‌کند که بطور اخص برای استفاده در تابع hash طراحی شده است و غیرمحتمل است که هرگز بعنوان یک تابع رمزنگاری تنها بکار رود. علت این امر این است که طراحان می‌خواسته‌اند از یک رمز قالبی با امنیت و بهره‌وری AES ولی با یک طول hash که امنیت بالقوه‌ای برابر SHA-512 را فراهم نماید، استفاده کنند. نتیجهٔ این کار تولید رمز قالبی W است که ساختاری مشابه AES داشته و از همان توابع ابتدائی استفاده می‌کند ولی اندازهٔ بلوک و اندازهٔ کلید آن ۵۱۲ بیت است.

الگوریتم بعنوان ورودی یک پیام با ماکزیمم طول کمتر از 2^{256} بیت را قبول کرده و یک چکیدهٔ پیام ۵۱۲- بیتی را تولید می‌کند. ورودی در بلوک‌های ۵۱۲- بیتی پردازش می‌گردد

HMAC

در سال‌های اخیر تمایل فزاینده‌ای به تولید یک MAC از یک کُد hash رمزی همانند SHA-1 وجود داشته است. محرک‌های این علاقه چنین بوده‌اند:

- توابع رمزی hash معمولاً سریع‌تر از الگوریتم‌های رمزنگاری متقارن مثل DES اجرا می‌شوند.
- کُدهای کتابخانه‌ای توابع hash بصورت گسترده‌ای در دسترس‌اند.

یک تابع درهم‌ساز مثل SHA-1 برای استفاده بصورت MAC طراحی نشده است و بنابراین نمی‌تواند مستقیماً برای این منظور بکار رود، زیرا متکی به یک کلید سرّی نیست. پیشنهادهای متعددی برای بکارگرفتن یک کلید سرّی در یک الگوریتم درهم‌سازی موجود داده شده است. روشی که بیشترین پذیرش را داشته است HMAC [BELL96a,BELL96b] است. HMAC تحت عنوان RFC 2104 منتشرشده و بعنوان یک انتخاب قطعی برای امنیت IP پذیرفته شده است. از HMAC همچنین در پروتکل‌های اینترنت دیگری چون امنیت لایهٔ حمل و نقل (TLS که بزودی جایگزین SSL خواهد شد) و اسناد امن الکترونیک (SET) استفاده می‌شود.

اهداف طراحی HMAC

RFC 2104 اهداف طراحی زیر برای HMAC را لیست کرده است:

- از توابع hash موجود بدون تغییرات استفاده کند، علی‌الخصوص توابعی که از نظر نرم‌افزاری خوب عمل کرده و برای آنها برنامهٔ آزاد و فراهم وجود دارد.
- اجازه دهد تا در صورتی که توابع hash امن‌تری پیدا شده و یا مورد نیاز واقع شوند، آن توابع بتوانند جایگزین تابع hash موجود در آن گردند.
- بتواند عملکرد اولیهٔ تابع hash را حفظ کرده، بدون اینکه کیفیت آن تنزل قابل ملاحظه‌ای یابد.
- بتواند کلیدها را به روش ساده‌ای مورد استفاده قرار دهد.
- تحلیل رمزنگاری قابل‌فهمی از قدرت مکانیسم اعتبارسنجی، بر اساس فرضیات معقول نسبت به تابع hash ارائه دهد.

اولین دو هدف در مورد پذیرش HMAC دارای اهمیت است. HMAC با تابع درهم‌ساز بصورت یک «جعبهٔ سیاه» برخورد می‌کند. این امر دو حسن دارد. اول اینکه یک فرم اجرائی از یک تابع hash می‌تواند بصورت یک مدول اجرائی در HMAC بکار رود. در این صورت بخش عمدهٔ کُد برنامهٔ HMAC از قبل آماده بوده و می‌تواند بدون دست خوردن مورد استفاده قرار گیرد. ثانیاً اگر روزی بخواهیم تا یک تابع hash در یک اجرای HMAC را عوض کنیم، تنها کار لازم این است که مدول تابع hash جاری را برداشته و مدول جدید را جایگزین آن نمائیم. این کار را می‌توان در صورتی که تابع hash سریع‌تری مورد نیاز باشد انجام داد. مهم‌تر اینکه اگر امنیت تابع hash موجود در HMAC به مخاطره افتد، امنیت HMAC را بسادگی می‌توان با جایگزین کردن تابع hash آن با یک تابع hash امن‌تر جبران نمود.

آخرین هدف طراحی در لیست قبل در واقع مزیت عمدهٔ HMAC بر سایر روش‌های پیشنهادشده مبتنی بر درهم‌سازی بوده است. HMAC در صورتی امن خواهد بود که تابع hash درونی آن دارای توان‌های رمزنگاری معقولی باشد. بعداً در این بخش به این نکته برمی‌گردیم، ولی فعلاً ساختار HMAC را بررسی می‌کنیم.

الگوریتم HMAC

شکل ۶-۳ عملیات HMAC را نشان می‌دهد. واژه‌های زیر را تعریف می‌کنیم:

H = تابع hash لحاظ شده در HAMC (مثل SHA-1)

M = پیام ورودی به HMAC (شامل بیت‌های لائی مشخص شده در تابع hash)

Y_i = بلوک i ام M $0 \le i \le (L\text{-}1)$

L = تعداد بلوک‌ها در M

b = تعداد بیت‌ها در یک بلوک

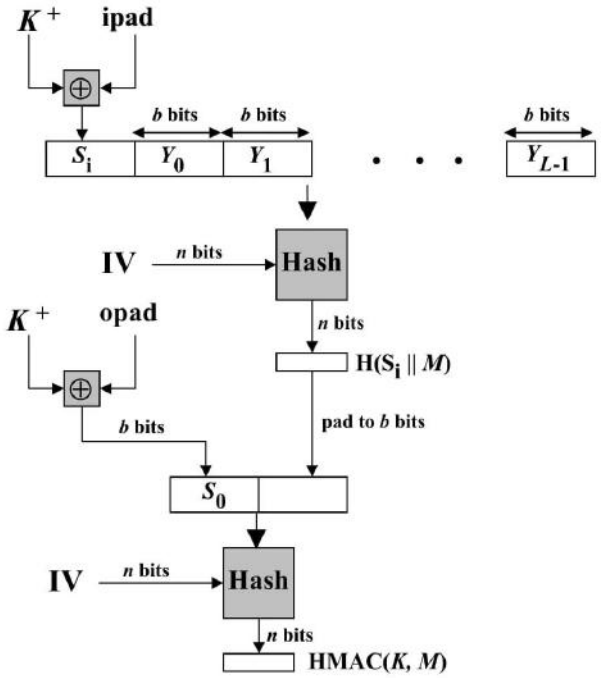
n = طول کُد hash تولیدشده بتوسط تابع درهم‌ساز

K =کلید سرّی. اگر طول کلید از b بزرگتر است، کلید وارد تابع hash شده تا یک کلید n- بیتی تولید شود. طول توصیه شده برای کلید بزرگتر و یا مساوی n است.

K^+ = K که بتوسط قراردادن صفرهائی در سمت چپ آن طویل‌تر شده تا طول آن به b بیت برسد.

ipad = 00110110 (معادل 36 هکزادسیمال) که $b/8$ بار تکرار شده است.

opad = 01011100 (معادل 5C هکزادسیمال) که $b/8$ بار تکرار شده است.



شکل ۶-۳ ساختار HMAC

با این تعاریف HMAC را می‌توان بصورت زیر نشان داد:

$$\text{HMAC}\ (K, M) = \text{H}[(K^{+} \oplus \text{opad}) \,||\, \text{H}[(K^{+} \oplus \text{ipad}) \,||\, \text{M}]]$$

که با کلمات چنین می‌شود:

۱- صفرهائی را به سمت چپ K اضافه کنید تا یک دنبالهٔ b- بیتی K^{+} حاصل شود (مثلاً اگر K دارای طول ۱۶۰ بیت بوده و $b = ۵۱۲$ باشد، آنگاه K با ۴۴ بایت 0 کامل می‌شود).

۲- K^{+} را بیت- به- بیت با ipad بصورت XOR درآورده تا بلوک b- بیتی S_i حاصل شود.

۳- M را به S_i وصل (جمع رشته‌ای) کنید.

۴- H را به دنبالهٔ تولیدشده در قدم سوم اعمال نمائید.

۵- K^{+} را با opad، XOR نموده تا بلوک b- بیتی S_0 حاصل شود.

۶- نتیجهٔ hash قدم چهارم را به S_0 وصل (جمع رشته‌ای) کنید.

۷- H را به دنبالهٔ تولیدشده در قدم ششم اعمال کرده و نتیجه را خارج سازید.

توجه شود که XOR با ipad باعث عوض شدن نیمی از بیت‌های K می‌شود. بطریق مشابه، XOR کردن با opad باعث تعویض نیمی از بیت‌های K در موقعیت‌های متفاوت دیگری می‌شود. در واقع با عبوردادن S_i و S_o از الگوریتم درهم‌سازی، بطور شبه‌تصادفی دو کلید از K تولید کرده‌ایم.

برای پیام‌های طولانی، زمان اجرای HMAC بایستی تقریباً برابر زمان اجرای تابع hash درون آن باشد. HMAC سه اجرای تابع hash درون خود را در بردارد (برای S_i و S_o و بلوکی که از hash درونی حاصل می‌شود).

۳-۳ اصول رمزنگاری کلید- عمومی

در ردیف اهمیت رمزنگاری متقارن، رمزنگاری کلید- عمومی قرار دارد که در اعتبارسنجی پیام و توزیع کلید مورد استفاده است. در این بخش ابتدا نگاهی به مفهوم رمزنگاری کلید- عمومی انداخته و سپس نگاهی ابتدائی به مقولهٔ توزیع کلید می‌اندازیم. بخش ۳-۴ دوقلم از مهم‌ترین الگوریتم‌های کلید عمومی یعنی RSA و Diffie-Hellman را بررسی می‌کند. در بخش ۳-۵ امضاءهای دیجیتال را معرفی می‌کنیم.

ساختار رمزنگاری کلید- عمومی

رمزنگاری کلید- عمومی که برای اولین بار بتوسط Diffie وHellman درسال۱۹۷۶درمعرض عموم قرار گرفت[DIFF76]. اولین جهش انقلابی واقعی در رمزنگاری در طول هزاران سال است. از یک دید، الگوریتم‌های کلید- عمومی بجای اینکه شامل عملیات ساده‌ای بر روی بیت‌ها باشند، مبتنی بر توابع ریاضی‌اند. مهم‌تر این‌که رمزنگاری کلید- عمومی بر خلاف رمزنگاری سنّتی که تنها از یک کلید استفاده می‌کند شامل استفاده از دو کلید مجزا بوده و نامتقارن است. استفاده از دو کلید نتایج فوق‌العاده‌ای در زمینهٔ محرمانگی، توزیع کلید و اعتبارسنجی به بارمی‌آورد.

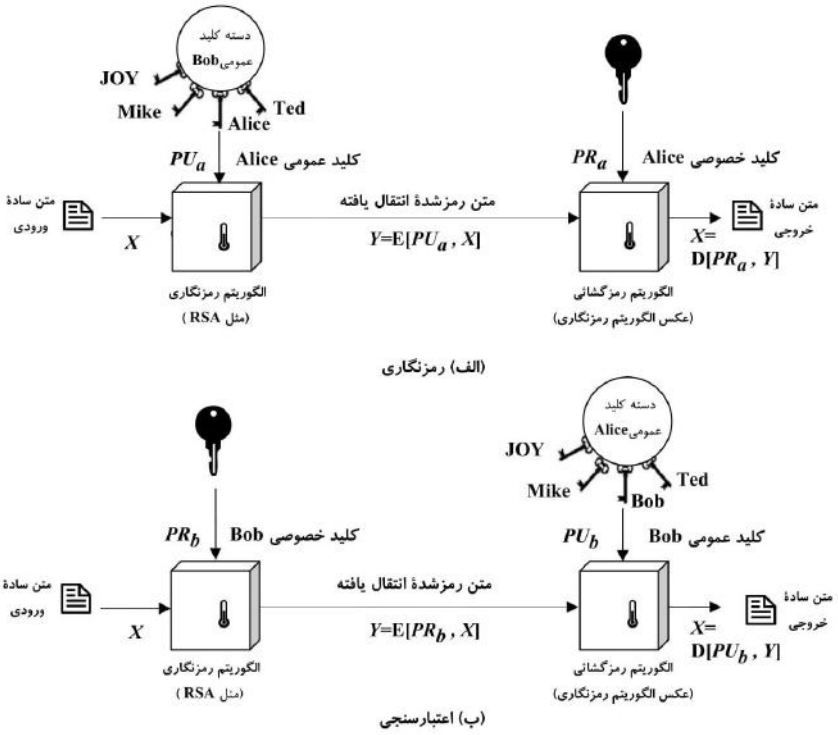
قبل از این‌که جلوتر برویم، بایستی در ابتدا به چندین اشتباه معمول در مورد رمزنگاری کلید- عمومی اشاره کنیم. یکی این‌که گفته می‌شود رمزنگاری کلید- عمومی در مقابل شکستن رمز امن‌تر از رمزنگاری سنّتی است. در واقع امنیت هر روش رمزنگاری بستگی به: (۱) طول کلید و (۲) کار محاسباتی لازم برای شکستن رمز دارد. از نظر اصولی هیچ چیز در مورد رمزنگاری سنّتی و یا رمزنگاری کلید- عمومی وجود ندارد که یکی را نسبت به دیگری در مقابل شکستن رمز ارجحیت دهد. اشتباه دوم این است که رمزنگاری کلید- عمومی را یک تکنیک عام تصور می‌کنند که رمزنگاری سنّتی را از میدان بدر کرده است. برعکس، بعلت سربارهٔ محاسباتی روش‌های رمزنگاری کلید- عمومی موجود، هیچ پیش‌بینی احتمالی در مورد زمان واگذاری رمزنگاری سنّتی نمی‌توان کرد. بالاخره این احساس وجود دارد که در استفاده از رمزنگاری کلید- عمومی، توزیع کلید در مقایسه با عملیات پیچیده‌ای که در مراکز توزیع کلید رمزنگاری سنّتی وجود دارد، کار ساده‌ای است. واقعیت این است که در این رمزنگاری جدید هم نوعی پروتکل که معمولاً شامل یک عامل مرکزی است مورد نیاز بوده و روش‌های امر نه ساده‌تر و نه بهره‌ورتر از آنهائی هستند که برای رمزنگاری سنّتی بکار می‌روند.

یک روش رمزنگاری کلید- عمومی دارای شش جزء است (شکل ۳-۷الف):

- **متن ساده**: این پیام خوانا و یا دیتائی است که به ورودی الگوریتم داده می‌شود.
- **الگوریتم رمزنگاری**: الگوریتم رمزنگاری، تبدیل‌های مختلفی را روی دیتا انجام می‌دهد.

- **کلید عمومی و کلید خصوصی**: یک جفت کلید است که طوری انتخاب می‌شوند که اگر از یکی برای رمزنگاری استفاده شود، از دیگری برای رمزگشائی استفاده خواهد شد. تبدیل‌های واقعی انجام‌شده بتوسط الگوریتم رمزنگاری، وابسته به کلید عمومی و یا کلید خصوصی است که در ورودی از آن استفاده شده است.
- **متن رمزشده**: این پیام درهم‌ریخته‌ای است که بعنوان خروجی تولید می‌شود. این پیام بستگی به متن ساده و کلید دارد. برای یک پیام واحد، دو کلید مختلف دو متن رمزشدهٔ مختلف ایجاد خواهند کرد.
- **الگوریتم رمزگشائی**: این الگوریتم متن رمزشده و کلید ذیربط را پذیرفته و متن سادهٔ اولیه را تولید می‌کند.

همانطور که از نام آن برمی‌آید، از بین دو کلید، کلید عمومی برای همه شناخته‌شده بوده و می‌توانند آن را بکار گیرند در حالی‌که کلید خصوصی تنها برای صاحب آن شناخته شده است. یک الگوریتم رمزنگاری کلید- عمومی با مصرف عام، متکی بر یک کلید برای رمزنگاری و یک کلید متفاوت ولی مرتبط برای رمزگشائی است.



شکل ۷-۳ رمزنگاری کلید- عمومی

قدم‌های اصلی اجرای روش بقرار زیراند:

۱- هر کاربر یک زوج کلید که برای رمزنگاری و رمزگشائی پیام‌ها بکار خواهد رفت را تولید می‌کند.
۲- هر کاربر یکی از دو کلید را در یک مخزن عمومی و یا فایل قابل دسترس دیگر قرار می‌دهد. این همان کلید عمومی است. کلید نظیر آن مخفی نگاه داشته خواهد شد. همان‌طور که شکل ۷-۳الف نشان می‌دهد، هر کاربر مجموعه‌ای از کلیدهای عمومی که از دیگران گرفته است را در اختیار دارد.
۳- اگر Bob بخواهد یک پیام خصوصی برای Alice بفرستد، Bob پیام را با استفاده از کلید عمومی Alice رمزنگاری خواهد کرد.
۴- وقتی Alice پیام را دریافت نمود، او پیام را با استفاده از کلید خصوصی خود رمزگشائی می‌کند. هیچ گیرندهٔ دیگری نمی‌تواند پیام را رمزگشائی کند زیرا تنها Alice از کلید خصوصی Alice مطلع است

در چنین روشی، تمام شرکت‌کنندگان به کلیدهای عمومی دسترسی دارند و کلیدهای خصوصی بطور محلی و بتوسط هر شرکت‌کننده تولید می‌شود و بنابراین لازم نیست تا توزیع شوند. تا زمانی که یک کاربر از کلید خصوصی خود محافظت می‌نماید، ارتباطات ورودی امن خواهند بود. در هرزمان، یک کاربر می‌تواند کلید خصوصی خود را عوض کرده و کلید عمومی مرتبط با آن را برای جایگزینی کلید قبلی به اطلاع عموم برساند.

کلید استفاده‌شده در رمزنگاری متقارن را معمولاً **کلید سرّی** می‌گویند. دو کلیدی که در رمزنگاری کلید- عمومی بکار می‌رود را **کلید عمومی** و **کلید خصوصی** می‌نامند. کلید خصوصی بدون استثناء سرّی نگاه داشته می‌شود ولی از این جهت آن را کلید خصوصی، و نه کلید سرّی، می‌گویند که با رمزنگاری سنّتی اشتباه نشود.

کاربردهائی برای سیستم‌های رمزنگاری کلید- عمومی

قبل از این‌که جلوتر برویم، لازم است یکی از جنبه‌های سیستم‌های رمزنگاری کلید- عمومی که ممکن است باعث گمراهی شود را روشن نمائیم. سیستم‌های کلید- عمومی با استفاده از یک نوع الگوریتم رمزنگاری با دو کلید که یکی از آنها خصوصی نگاه داشته شده و دیگری در دسترس عموم است، مشخص می‌گردد. بر حسب اینکه نوع کاربرد چیست، فرستنده یا از کلید خصوصی خود و یا از کلید عمومی گیرنده و یا از هردو کلید برای انجام نوعی عمل رمزنگاری استفاده می‌کند. در مفهوم وسیع، موارد استفاده از سیستم‌های رمزنگاری کلید- عمومی را می‌توانیم به سه دسته تقسیم کنیم:

- **رمزنگاری/ رمزگشائی**: فرستنده یک پیام را با کلید عمومی گیرنده به رمز درمی‌آورد.
- **امضاء دیجیتال**: فرستنده یک پیام را با کلید خصوصی خود «امضاء» می‌کند. امضاء با اعمال یک الگوریتم رمزنگاری به پیام و یا به بلوک کوچکی از دیتا که تابعی از پیام است انجام می‌شود.
- **مبادلهٔ کلید**: دوطرف برای مبادلهٔ یک کلید اجلاس همکاری می‌کنند. چندین روش مختلف که شامل کلید یا کلیدهای خصوصی یکی از طرفین و یا هردوی آنهاست وجود دارد.

بعضی از الگوریتم‌ها برای هرسه کاربرد مناسب‌اند در حالی‌که برخی دیگر تنها برای یک یا دو کاربرد به درد می‌خورند. جدول ۲-۳ کاربردهائی که مورد حمایت الگوریتم‌های مورد بحث در این فصل یعنی RSA و Diffie Hellman هستند را نشان می‌دهد. جدول همچنین شامل استاندارد امضاء دیجیتال (DSS) و رمزنگاری خَم بیضوی نیز هست که بعداً در همین فصل به آنها اشاره خواهد شد.

جدول ۲-۳ کاربردهای سیستم‌های رمزنگاری کلید- عمومی

الگوریتم	رمزنگاری / رمزگشائی	امضاء دیجیتال	مبادلهٔ کلید
RSA	بلی	بلی	بلی
Diffie-Hellman	خیر	خیر	بلی
DSS	خیر	بلی	خیر
Elliptic Curve	بلی	بلی	بلی

لازمه‌های رمزنگاری کلید- عمومی

سیستم رمزنگاری نشان‌داده‌شده در شکل ۷-۳ وابسته به یک الگوریتم رمزنگاری است که بر مبنای دو کلید مرتبط بنا نهاده شده است. Diffie و Hellman چنین سیستمی را مفروض دانسته و بدون این‌که نشان دهند که واقعاً چنین الگوریتمی وجود دارد، شرایطی که چنین الگوریتم‌هائی باید داشته باشند را چنین بیان کرده‌اند[DIFF76]:

۱- از نظر محاسباتی برای طرف B ساده است تا یک زوج کلید (کلید عمومی PU_b و کلید خصوصی PR_b) تولید کند.

۲- از نظر محاسباتی برای فرستندهٔ A ساده است تا با دانستن کلید عمومی و پیامی که باید رمزنگاری شود(M)، متن رمزشدهٔ نظیر را تولیدکند.

$$C = \mathrm{E}\,(PU_b, M)$$

۳- از نظر محاسباتی برای گیرندهٔ B ساده است تا متن رمزشدهٔ نتیجه را با استفاده از کلید خصوصی رمزگشائی نماید:

$$M = \mathrm{D}\,(PR_b, C) = \mathrm{D}\,[PR_b, \mathrm{E}\,(PU_b, M)]$$

۴- از نظر محاسباتی برای یک دشمن میسر نیست که با داشتن کلید عمومی PU_b، کلید خصوصی PR_b را بدست آورد.

۵- از نظر محاسباتی برای یک دشمن میسر نیست که با داشتن کلید عمومی PU_b و یک متن رمزشده C، پیام اولیهٔ M را استخراج نماید.

می‌توانیم شرط مشخصی را که اگرچه مفید است ولی برای کاربردهای کلید عمومی لازم نیست اضافه نمائیم:

۶- هریک از دو کلید مربوط می‌تواند برای رمزنگاری بکار رود و از کلید دیگر برای رمزگشائی استفاده خواهد شد.

$$M = \mathrm{D}\,[PU_b, \mathrm{E}\,(PR_b, M)] = \mathrm{D}\,[PR_b, \mathrm{E}\,(PU_b, M)]$$

۳-۴ الگوریتم‌های رمزنگاری کلید- عمومی

دو مورد از پراستفاده‌ترین الگوریتم‌های کلید- عمومی، RSA و Diffie- Hellman هستند. ما در این بخش هر دوی آنها را بررسی نموده و دو الگوریتم دیگر را نیز بطور مختصر معرفی خواهیم کرد.

الگوریتم رمزنگاری کلید- عمومی RSA

یکی از اولین روش‌های کلید- عمومی است که در سال ۱۹۷۷ بتوسط Adi Shamir,Ron Rivest و Len Adleman در دانشگاه MIT شکل گرفت و برای اولین بار در ۱۹۷۸ منتشر گردید[RIVE78]. روش RSA از آن زمان بعنوان پذیرفته‌شده‌ترین و پرکاربردترین روش رمزنگاری کلید- عمومی در اوج قرارداشته است. RSA یک رمز قالبی است که درآن متن ساده و متن رمزشده اعداد صحیحی بین 0 تا n-1 برای مقداری از n می‌باشند.

برای بلوک متن سادهٔ M و بلوک متن رمزشدهٔ C، رمزنگاری و رمزگشائی بشکل زیر است:

$$C = M^e \bmod n$$
$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

هم گیرنده و هم فرستنده بایستی از مقادیر n و e مطلع بوده ولی فقط گیرنده اندازهٔ d را می‌داند. این یک الگوریتم رمزنگاری کلید- عمومی با کلید عمومی $PU = \{e,n\}$ و کلید خصوصی $PR = \{d,n\}$ است. برای این‌که این الگوریتم برای رمزنگاری کلید- عمومی رضایت‌بخش باشد، نیازهای زیر بایستی برآورده شوند:

۱- بایستی ممکن باشد مقدار e، d و n را طوری پیدا نمود که برای جمیع مقادیر M ،$M< n$ باشد. $M^{ed} = M \bmod n$

۲ - محاسبهٔ M^e و C^d برای جمیع مقادیر $M < n$ نسبتاً آسان باشد.

۳ - با داشتن e و n، تعیین d مقدور نباشد.

دو نیاز اول به سهولت برآورده می‌شوند. نیاز سوم برای مقادیر بزرگ e و n قابل حصول است.

شکل ۸-۳ الگوریتم RSA را خلاصه کرده است. در ابتدا دو عدد اول p و q انتخاب و حاصلضرب آنها n محاسبه می‌گردد که پیمانهٔ (module) رمزنگاری و رمزگشائی است. سپس به کمیت $\phi(n)$ نیاز داریم که تابع Euler نامیده شده و تعداد اعداد صحیح مثبت کوچک‌تر از n و اول نسبت به n است. آنگاه یک عدد صحیح e طوری انتخاب می‌گردد که نسبت به $\phi(n)$ اول باشد [یعنی بزرگترین مقسوم‌علیه مشترک e و $\phi(n)$ مساوی ۱ باشد]. بالاخره d بعنوان معکوس ضربی n و بصورت modulo $\phi(n)$ محاسبه می‌شود. می‌توان نشان داد که d و e دارای خواص مطلوب هستند.

فرض کنید که کاربر A کلید عمومی خود را معـرفی کرده و کاربر B می‌خواهد پیـام M را برای A بفرستد. B مقدار $C = M^e \pmod n$ را محاسبه کرده و C را ارسال می‌کند. کاربر A پس از دریافت مـتن رمزشـده، $M = C^d \bmod n$ را محاسبه و آن را از رمز درمی‌آورد.

مثالی از [SING99] در شکل ۹-۳ نشان داده شده است. برای این مثال کلیدها بصورت زیر تولید شده‌اند:

۱- دو عدد اول $p = 17$ و $q = 11$ را انتخاب کنید.

۲- $n = pq = 17\times11 = 187$ را محاسبه نمائید.

۳- $\phi(n) = (p-1)(q-1) = 16\times10 = 160$ را محاسبه نمائید.

۴- e را طوری انتخاب کنید که e نسبت به $\phi(n) = 160$ اول بوده و کمتر از آن هم باشد مثلاً $e = 7$.

۵- d را طوری انتخاب کنید که $de \bmod 160 = 1$ باشد. اندازهٔ صحیح $d = 23$ است زیرا $23\times7 = 1\times160+1$.

تولید کلید

p و q را انتخاب کنید.	p و q هردو اول‌اند و $p \neq q$
$n = p \times q$ را محاسبه نمائید.	
$\phi(n) = (p-1)(q-1)$ را بدست آورید.	
عدد صحیح e را انتخاب کنید.	بزرگترین مقسوم‌علیه مشترک $\phi(n)$ و e برابر ۱ است.
d را حساب کنید.	$de \bmod \phi(n) = 1$
کلید عمومی	$PU = \{e, n\}$
کلید خصوصی	$PR = \{d, n\}$

رمزنگاری

متن ساده :	$M < n$
متن رمزشده :	$C = M^e \bmod n$

رمزگشائی

متن رمزشده :	C
متن ساده :	$M = C^d \bmod n$

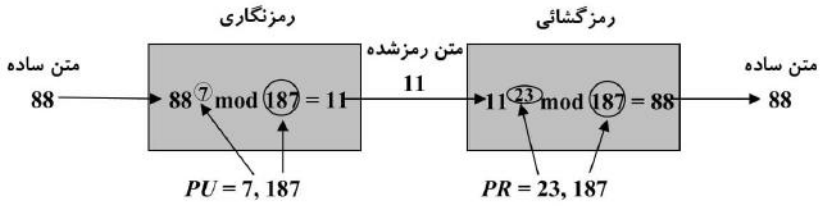
شکل ۸-۳ الگوریتم RSA

کلیدهای بدست‌آمده برابر کلید عمومی $PU = \{7,187\}$ و $PR = \{23,187\}$ هستند. مثال، استفاده از این کلیدها را برای یک ورودی متن ساده $M = 88$ نشان می‌دهد. برای رمزنگاری لازم است $C = 88^7 \bmod 187$ را حساب کنیم. با بکارگیری خواص محاسبات پیمانه‌ای، محاسبه چنین انجام می‌شود:

$$88^7 \bmod 187 = [(88^4 \bmod 187) \times (88^2 \bmod 187) \times (88^1 \bmod 187)] \bmod 187$$
$$88^1 \bmod 187 = 88$$
$$88^2 \bmod 187 = 7744 \bmod 187 = 77$$
$$88^4 \bmod 187 = 59{,}969{,}536 \bmod 187 = 132$$
$$88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187 = 894{,}432 \bmod 187 = 11$$

برای رمزگشائی $M = 11^{23} \bmod 187$ را محاسبه می‌کنیم.

$$11^{23} \bmod 187 = [(11^1 \bmod 187) \times (11^2 \bmod 187) \times (11^4 \bmod 187) \times (11^8 \bmod 187) \times (11^8 \bmod 187)] \bmod 187$$
$$11^1 \bmod 187 = 11$$
$$11^2 \bmod 187 = 121$$
$$11^4 \bmod 187 = 14{,}641 \bmod 187 = 55$$
$$11^8 \bmod 187 = 214{,}358{,}881 \bmod 187 = 33$$
$$11^{23} \bmod 187 = (11 \times 121 \times 55 \times 33 \times 33) \bmod 187 = 79{,}720{,}254 \bmod 187 = 88$$



شکل ۹-۳ مثال الگوریتم RSA

دو روش برای شکست دادن الگوریتم RSA متصوّر است. اولی جستجوی کامل است: تمام کلیدهای ممکن را امتحان کنید. بنابراین هرچقدر تعداد بیت‌های e و d زیادتر باشد، الگوریتم امن‌تر خواهد بود. از طرفی چون هم در تولید کلید و هم در رمزنگاری/ رمزگشائی پای محاسبات پیچیده‌ای در میان است، هرچقدر اندازهٔ کلید بزرگتر باشد سیستم کندتر عمل خواهد کرد.

بیشتر بحث‌های مربوط به شکستن رمز RSA روی فاکتورگیری n به دو عدد اول متمرکز شده است. برای یک n بزرگ با فاکتورهای اول بزرگ، فاکتورگیری یک عمل مشکل است که البته در حال حاضر به سختی قبل نیست. جلوهٔ ویژهٔ این امر در سال ۱۹۷۷ اتفاق افتاد که سه مخترع RSA، خوانندگان مجلهٔ *Scientific American* را تشویق کردند تا رمزی را که آنها در ستون «بازی‌های ریاضی» Martin Gardner [GARD77] چاپ کرده بودند کشف کنند. آنها برای کشف رمز و استخراج متن سادهٔ پیام، جایزه‌ای برابر ۱۰۰ دلار تعیین کرده بودند و پیش‌بینی نموده بودند که این کار زودتر از یک میلیون میلیون سال (10^{12} سال) عملی نمی‌باشد. در آوریل ۱۹۹۴ یک گروه که روی اینترنت کار می‌کردند و ۱,۶۰۰ کامپیوتر را بکارگرفته بودند، پس از هشت ماه کار مدعی این جایزه شدند[LEUT94]. این مبارزه از یک کلید عمومی با اندازهٔ ۱۲۹ رقم دهدهی (طول n) و یا حدوداً ۴۲۸ بیت استفاده کرده بود. این نتیجه بهیچ‌وجه RSA را بی اعتبار نمی‌کند بلکه معنی آن این است که بایستی از کلیدهای طولانی‌تر استفاده کرد. امروزه یک کلید با طول ۱,۰۲۴ بیت (تقریباً ۳۰۰ رقم دهدهی)، یک کلید مستحکم برای تقریباً تمام کاربردها بحساب می‌آید.

مبادلهٔ کلید Diffie-Hellman

نطفهٔ الگوریتم کلید- عمومی در یک مقاله بتوسط Diffie وHellman شکل گرفت که رمزنگاری کلید- عمومی را تعریف کرده است[DIFF76] و معمولاً از آن بعنوان مبادلهٔ کلید Diffie- Hellman یاد می‌شود. تعدادی از محصولات تجاری از این تکنیک مبادلهٔ کلید استفاده می‌کنند.

هدف الگوریتم این است که دو کاربر را قادر سازد که یک کلید سرّی را بصورت امن مبادله نمایند تا بعداً از این کلید برای رمزنگاری آتی پیام‌ها استفاده شود. خود الگوریتم منحصر به مبادلهٔ کلیدهاست.

مؤثربودن الگوریتم Diffie-Hellman متکی به پیچیدگی محاسبه لُگاریتم‌های گسسته است. لُگاریتم گسسته را می‌توان مختصراً چنین تعریف کرد: ابتدا یک ریشهٔ اولیه (primitive) یک عدد اول p را بعنوان عددی که توان‌های آن همهٔ اعداد صحیح 1 تا p-1 را تولید می‌کند تعریف می‌کنیم. یعنی اگر α یک ریشهٔ اولیهٔ عدد اول p باشد، آنگاه اعداد $\alpha \bmod p$، $\alpha^2 \bmod p$ و .... $\alpha^{p-1} \bmod p$ کاملاً متمایز بوده و شامل اعداد صحیح 1 تا p-1 با جایگشت‌هائی می‌باشند.

برای هر عدد صحیح b کوچک‌تر از p و یک ریشهٔ اولیهٔ α از عدداول p ، می‌توان یک نمای یکتای i را طوری پیدا کرد که

$$b = \alpha^i \bmod p \qquad 0 \le i \le (p\text{-}1)$$

نمای i را لُگاریتم گسستهٔ یا اندیس b برای پایهٔ α به پیمانهٔ p نامند. این مقدار را به شکل $\mathrm{dlog}_{\alpha,p}(b)$ نشان می‌دهیم.

الگوریتم

با این زمینه می‌توانیم مبادلهٔ کلید Diffie-Hellman، که در شکل ۱۰-۳ خلاصه شده است، را تعریف کنیم. در این روش دو عدد شناخته شده وجود دارد: یک عدد اول q و یک عدد صحیح α که ریشهٔ اولیهٔ q است. فرض کنید که کاربران A و B بخواهند تا کلیدی را مبادله نمایند. کاربر A یک عدد صحیح $X_A < q$ را بصورت تصادفی انتخاب کرده و $Y_A = \alpha^{X_A} \bmod q$ را حساب می‌کند. بطریق مشابه، کاربر B بصورت مستقل یک عدد صحیح تصادفی $X_B < q$ را انتخاب کرده و $Y_B = \alpha^{X_B} \bmod q$ را حساب می‌کند. هرطرف مقدار X را سرّی نگاه داشته و مقدار Y را بطور آشکار در اختیار طرف مقابل می‌گذارد. کاربر A کلید را بصورت $K = (Y_B)^{X_A} \bmod q$ محاسبه نموده و کاربر B نیز کلید را بصورت $K = (Y_A)^{X_B} \bmod q$ محاسبه می‌کند. این دو محاسبه نتایج یکسانی را تولید می‌کنند.

$$\begin{aligned}
K &= (Y_B)^{X_A} \bmod q \\
&= (\alpha^{X_B} \bmod q)^{X_A} \bmod q \\
&= (\alpha^{X_B})^{X_A} \bmod q \\
&= \alpha^{X_B X_A} \bmod q \\
&= (\alpha^{X_A})^{X_B} \bmod q \\
&= (\alpha^{X_A} \bmod q)^{X_B} \bmod q \\
&= (Y_A)^{X_B} \bmod q
\end{aligned}$$

نتیجه این است که دو طرف یک کلید سرّی را مبادله کرده‌اند. علاوه براین چون X_A و X_B سرّی هستند، یک دشمن فقط می‌تواند با Y_A ،q ،α و Y_B کارکند و مجبور است برای تعیین کلید، یک لُگاریتم گسسته را حساب کند. برای مثال برای حمله به کلید سرّی کاربر B، دشمن بایستی $X_B = \mathrm{dlog}_{\alpha,q}(Y_B)$ را حساب کند. دشمن آنگاه خواهد توانست کلید K را بهمان نحوی که کاربر B آن را محاسبه می‌کند بدست آورد.

مقادیر قابل دسترس عمومی

عدد اول q

عدد α $\alpha < q$ بوده و α ریشهٔ سادهٔ q است.

تولید کلید کاربر A

مقدار خصوصی X_A را انتخاب کنید. $X_A < q$

مقدار عمومی Y_A را حساب کنید. $Y_A = \alpha^{X_A} \bmod q$

تولید کلید کاربر B

مقدار خصوصی X_B را انتخاب کنید. $X_B < q$.

مقدار عمومی Y_B را حساب کنید. $Y_B = \alpha^{X_B} \bmod q$

تولید کلید سرّی به توسط کاربر A

$K = (Y_B)^{X_A} \bmod q$

تولید کلید سرّی به توسط کاربر B

$K = (Y_A)^{X_B} \bmod q$

شکل ۱۰-۳ الگوریتم مبادلهٔ کلید Diffie-Hellman

امنیت مبادلهٔ کلید Diffie- Hellman در این حقیقت نهفته است که در حالی‌که محاسبهٔ نماهائی با مدول یک عدد اول نسبتاً ساده است، محاسبهٔ لُگاریتم گسسته کاری بس مشکل است. برای اعداد اول بزرگ، این کار غیرعملی است.

برای مثال، مبادلهٔ کلید بر اساس استفاده از عدد اول $q = 353$ و یک ریشهٔ اولیهٔ 353 که در این مورد $\alpha = 3$ است قرار دارد. A و B کلیدهای سرّی $X_A = 97$ و $X_B = 233$ را به‌ترتیب انتخاب نموده و هرکدام کلید عمومی خود را محاسبه می‌کنند:

A چنین حساب می‌کند $Y_A = 3^{97} \bmod 353 = 40$

B چنین حساب می‌کند $Y_B = 3^{233} \bmod 353 = 248$

پس از این که آنها کلیدهای عمومی خود را مبادله کردند، هریک خواهد توانست تا کلید سرّی مشترک را حساب کند:

A چنین حساب می‌کند $K = (Y_B)^{X_A} \bmod 353 = 248^{97} \bmod 353 = 160$

B چنین حساب می‌کند $K = (Y_A)^{X_B} \bmod 353 = 40^{233} \bmod 353 = 160$

فرض می‌کنیم که یک حمله کننده اطلاعات زیر را در اختیار داشته باشد:

$$q = 353 \; ; \; \alpha = 3 \; ; \; Y_A = 40 \; ; \; Y_B = 248$$

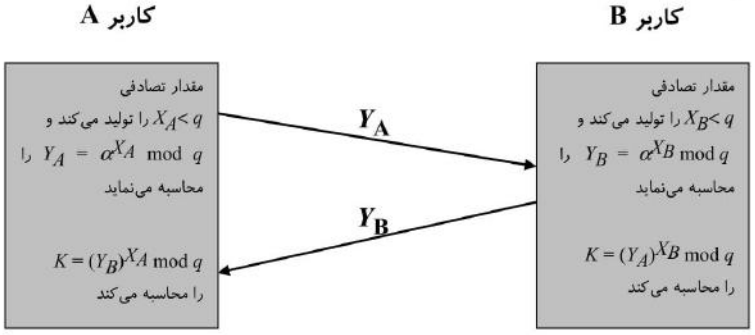
در این مثال ساده، با یک حملهٔ همه جانبه ممکن است که کلید سرّی 160 را بدست آورد. علی‌الخصوص حمله‌کنندهٔ E می‌تواند کلید مشترک را با حل معادلهٔ $3^a \mod 353 = 40$ و یا معادلهٔ $3^b \mod 353 = 248$ پیدا کند. روش جستجوی همه جانبه این است که تمام توان‌های 3 به پیمانهٔ 353 (modulo 353) را محاسبه کنیم تا نتیجه 40 و یا 248 شود. نتیجهٔ مطلوب وقتی است که به 97 برسیم زیرا $3^{97} \mod 353 = 40$ است.

با اعداد بزرگ‌تر، حل مسأله مقدور نخواهد بود.

پروتکل‌های مبادلهٔ کلید

شکل ۱۱-۳ یک پروتکل ساده که از محاسبات Diffie-Hellman استفاده می‌کند را نشان می‌دهد. فرض کنید که کاربر A می‌خواهد ارتباطی را با کاربر B برقرار کرده و از یک کلید سرّی برای رمزنگاری پیام‌ها در این ارتباط استفاده نماید.کاربر A می‌تواند یک کلید سرّی یکبارمصرف X_A را تولید کرده، Y_A را حساب نموده و آن را برای کاربر B بفرستد. کاربر B می‌تواند در پاسخ یک کلید سرّی X_B تولید نموده، Y_B را محاسبه کرده و Y_B را برای کاربر A بفرستد. هر دو کاربر حالا می‌توانند کلید را محاسبه نمایند. مقادیر q و α بایستی قبلاً معلوم باشند. در حالت دیگر کاربر A می‌تواند مقادیر q و α را انتخاب کرده و آنها را در پیام اول بگنجاند.

بعنوان استفادهٔ دیگری از الگوریتم Diffie-Hellman فرض کنید که گروهی از کاربران (مثلاً تمام کاربران یک شبکهٔ LAN) هرکدام یک مقدار سرّی بادوام X_A را تولید کرده و اندازهٔ عمومی Y_A را محاسبه نمایند. این مقادیر عمومی به همراه مقادیر عمومی q و α در یک فهرست مرکزی نگهداری می‌شوند. در هر زمان کاربر B می‌تواند به مقدار آشکار کاربر A دسترسی یافته، یک کلید سرّی را حساب کرده و از آن برای ارسال یک پیام رمزی برای کاربر A استفاده نماید. اگر فهرست مرکزی قابل اطمینان باشد، آنگاه این نوع ارتباط هم محرمانگی و هم تا حدودی اعتبارسنجی را فراهم می‌کند. چون فقط A و B می‌توانند کلید را تعیین نمایند، هیچ کاربر دیگری نمی‌تواند پیام را بخواند (محرمانگی). دریافت‌کنندهٔ A می‌داند که تنها کاربرB می‌توانسته است پیامی با این کلید را خلق کرده باشد (اعتبارسنجی). اما این تکنیک در برابر حملات بازخوانی (replay) امن نیست.



شکل ۱۱-۳ مبادلهٔ کلید Diffie-Hellman

حملهٔ Man-in-the-Middle

پروتکل نشان داده شده در شکل ۱۱-۳ در برابر حملهٔ man-in-the-middle ناامن است. فرض کنید که Alice و Bob می‌خواهند کلیدهائی را مبادله کنند و Darth دشمن فرضی است. حمله چنین جلو می‌رود:

۱- Darth با تولید دو کلید خصوصی X_{D1} و X_{D2} و سپس محاسبهٔ کلیدهای عمومی Y_{D1} و Y_{D2} خود را برای حمله آماده می‌کند.
۲- Alice اندازهٔ Y_A را برای Bob ارسال می‌کند.
۳- Darth اندازهٔ Y_A را دزدیده و Y_{D1} را برای Bob می‌فرستد. Darth همچنین $K2 = (Y_A)^{X_{D2}} \bmod q$ را محاسبه می‌کند.
۴- Bob اندازهٔ Y_{D1} را دریافت کرده و $K1 = (Y_{D1})^{X_B} \bmod q$ را محاسبه می‌کند.
۵- Bob اندازهٔ Y_B را برای Alice می‌فرستد.
۶- Darth اندازهٔ Y_B را دزدیده و Y_{D2} را برای Alice ارسال می‌کند. Darth همچنین $K1 = (Y_B)^{X_{D1}} \bmod q$ را محاسبه می‌کند.
۷- Alice اندازهٔ Y_{D2} را دریافت کرده و $K2 = (Y_{D2})^{X_A} \bmod q$ را محاسبه می‌کند.

در این نقطه، Bob و Alice تصور می‌کنند که یک کلید سرّی را در اشتراک دارند درحالی که واقعیت این است که Bob و Darth کلید K1 را در اشتراک داشته و Alice و Darth نیز کلید K2 را در اشتراک دارند. تمام ارتباطات آتی بین Bob و Alice به طریق زیر لو خواهند رفت:

۱- Alice یک پیام رمزنگاری شدهٔ $E(K2, M)$: M را ارسال می‌کند.
۲- Darth پیام رمزنگاری شده را دزدیده و آن را رمزگشائی کرده تا M را بدست آورد.
۳- Darth مقدار $E(K1, M)$ یا $E(M')$ را برای Bob ارسال می‌کند که درآن M' هر پیام دلخواهی است.

در بند ۲، Darth تنها بسادگی می‌خواهد روی ارتباطات شنود داشته باشد. در بند ۳، Darth می‌خواهد که پیام‌های ارسالی برای Bob را تغییر دهد.

پروتکل مبادلهٔ کلید به چنین حمله‌ای آسیب‌پذیر است زیرا هویت طرفین ارتباط در آن احراز نمی‌گردد. این آسیب‌پذیری را می‌توان با استفاده از امضاء دیجیتال و گواهی‌نامه‌های کلید- عمومی که بعداً در این فصل و فصل ۴ در مورد آنها بحث خواهد شد،از بین برد.

سایر الگوریتم‌های رمزنگاری کلید- عمومی

دو الگوریتم کلید- عمومی دیگر نیز پذیرش تجاری یافته‌اند: DSS و رمزنگاری خَم بیضوی.

استاندارد امضاء دیجیتال Digital Signiture Standard (DSS)

انستیتوی ملی استانداردها و تکنولوژی امریکا (NIST)، استاندارد فدرال پردازش اطلاعات FIP PUB 186 را با نام استاندارد امضاء دیجیتال (DSS) منتشر کرده است. DSS از SHA-1 استفاده کرده و یک تکنیک جدید امضاء دیجیتال بنام الگوریتم امضاء دیجیتال (DSA) را معرفی می‌کند. DSS ابتدا در سال ۱۹۹۱ پیشنهاد گردید و در سال ۱۹۹۳ در پاسخ به

نظراتی که در مورد امنیت روش مطرح گردیده بود مورد بازنگری قرار گرفت. تغییر کوچکی هم در سال ۱۹۹۶ در آن بوجود آمد. DSS از الگوریتمی استفاده می‌کند که تنها برای فراهم‌آوردن تابع امضاء دیجیتال طراحی شده است. این روش نمی‌تواند برای رمزنگاری یا مبادلهٔ کلید بکار رود.

رمزنگاری خَم بیضوی Elliptic-Curve Cryptography (ECC)

اکثر محصولات و استانداردهائی که از رمزنگاری کلید- عمومی و امضاءهای دیجیتال استفاده می‌کنند، RSA را بکار می‌برند. اندازهٔ طول بیت برای RSA امن در طول سال‌های اخیر افزایش یافته و این امر بار پردازش سنگین‌تری را روی کاربردهائی که از RSA استفاده می‌کنند اعمال کرده است. این معضل دارای جلوه‌های متعددی است که علی‌الخصوص در سایت‌های تجارت الکترونیک که اسناد مالی زیادی باید بصورت امن مبادله شوند، نمود بیشتری دارد. اخیراً یک سیستم رقیب، RSA را به مبارزه طلبیده است: رمزنگاری خَم بیضوی (ECC). هم‌اکنون ECC در تلاش‌های استانداردسازی که شامل IEEE P1363 است ویژگی‌های خود را نشان داده است.

جاذبهٔ اصلی ECC در مقایسه با RSA این است که بنظر می‌رسد تکنیک جدید همان امنیت را برای اندازهٔ بیت بسیار کمتری بوجود می‌آورد و در نتیجه سربارهٔ پردازش کم می‌شود. از طرف دیگر اگرچه تئوری ECC مدتهاست که مطرح بوده است، تنها در سال‌های اخیر است که محصولات مرتبط با آن به بازار آمده و علاقهٔ زیادی برای نفوذ در این الگوریتم و کشف نقاط ضعف آن ظاهر شده است. بنابراین سطح اطمینان به ECC هنوز به اندازهٔ RSA بالا نیست.

توضیح مبانی ECC مشکل‌تر از RSA و Diffie-Hellman بوده و توصیف کامل ریاضی آن فراتر از حیطهٔ این کتاب است. مبنای روش، استفاده از یک ساختار ریاضی، بنام خَم بیضوی است.

۵-۳ امضاءهای دیجیتال

از رمزنگاری کلید- عمومی می‌توان بصورت دیگری همانند شکل ۷-۳ب استفاده کرد. فرض کنید که Bob بخواهد تا پیامی را برای Alice بفرستد و اگرچه مهم نیست که پیام سرّی بماند ولی اصرار دارد که Alice مطمئن شود که پیام واقعاً از طرف اوست. در این مورد Bob از کلید خصوصی خود برای رمزنگاری پیام استفاده می‌کند. وقتی Alice متن رمزشده را دریافت می‌دارد، او متوجه می‌شود که می‌تواند پیام را با کلید عمومی Bob رمزگشائی کند و بدین ترتیب اثبات می‌شود که پیام بتوسط Bob رمزنگاری شده است. هیچ شخص دیگری کلید خصوصی Bob را ندارد و بنابراین شخص دیگری نمی‌توانسته است متن رمزشده‌ای را خلق کند که با کلید عمومی Bob باز شود. بنابراین کل پیام رمزشده بصورت یک **امضاء دیجیتال** عمل می‌کند. علاوه برآن غیرممکن است که بتوان پیام را بدون دسترسی به کلید خصوصی Bob تغییر داد و بنابراین اعتبار پیام چه ازنظر منبع ارسال و چه از نظر اصالت تأئید می‌گردد.

در روش قبل، تمام پیام رمزنگاری می‌شودکه اگرچه هم نویسنده و هم محتوای پیام تأئید می‌گردد ولی به حجم حافظهٔ زیادی نیاز دارد. هر سند را بایستی بصورت متن ساده نگهداری نمود تا در صورت نیاز به آن مراجعه کرد. یک کپی از متن رمزشده را نیز بایستی حفظ کرد تا در صورت وجود تناقض بتوان، مبدأ و محتوا را با اصل تطبیق داد. راه بهره‌ورتری برای کسب همین نتایج این است که بلوک کوچکی از بیت‌ها که تابعی از پیام است را رمزنگاری کرد. چنین بلوکی که اعتبارسنج خوانده می‌شود بایستی دارای این خاصیت باشد که امکان نداشته باشد که بتوانا سند را تغییر داد ولی اعتبارسنج تغییر نکند. اگر اعتبارسنج با کلید خصوصی فرستنده رمزنگاری شده باشد، بعنوان یک امضاء که مبدأ، محتوا، و نظم را تأئید می‌کند عمل

خواهد کرد. یک کُد hash امن همانند SHA-1 می‌تواند برای این مقصود بکار رود. شکل ۲-۳ب این سناریو را نشان می‌دهد.

مهم است تأکید کنیم که عمل رمزنگاری که هم‌اکنون تشریح گردید، محرمانگی را فراهم نمی‌آورد. یعنی پیامی که ارسال می‌شود را نمی‌توان تغییر داد ولی می‌توان آن را استراق سمع کرد. این امر در مورد امضائی که مبتنی بر بخشی از پیام است روشن است، زیرا بقیهٔ پیام بصورت متن ساده ارسال می‌گردد. ولی حتی در صورت رمزنگاری کامل پیام، بازهم هیچ حفاظتی در برابر محرمانگی وجود ندارد زیرا هر ناظری می‌تواند پیام را با استفاده از کلید عمومی فرستنده رمزگشائی کند.

۳-۶ مدیریت کلید

یکی از نقش‌های عمدهٔ رمزنگاری کلید- عمومی، موضوع توزیع کلید است. در واقع استفاده از رمزنگاری کلید- عمومی برای این مقصود دارای دو جنبه است:

- توزیع کلیدهای عمومی
- استفاده از رمزنگاری کلید- عمومی برای توزیع کلیدهای سرّی

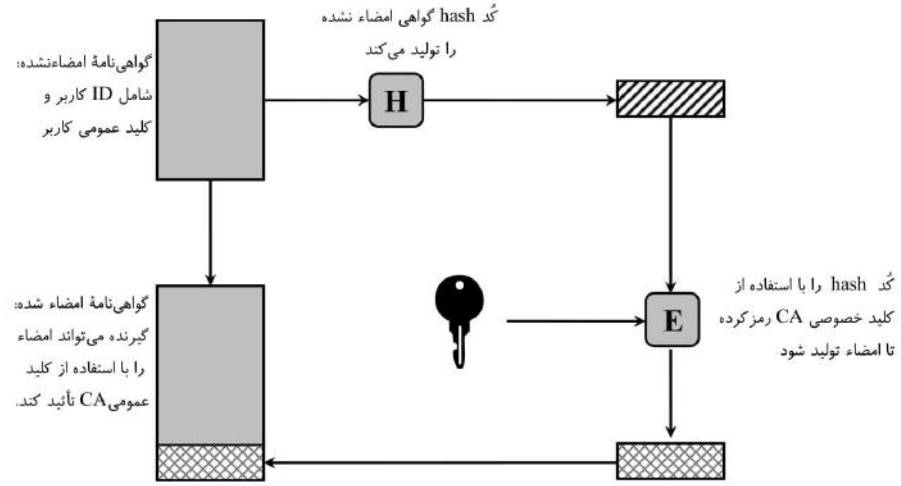
هریک از این دو مقوله را بترتیب بررسی می‌کنیم.

گواهی‌نامه‌های کلید- عمومی

همان‌طور که از اسم آن برمی‌آید، نکتهٔ رمزنگاری کلید- عمومی این است که کلید عمومی، عمومی است. بنابراین با الگوریتم پذیرفته‌شده‌ای همانند RSA، هر شرکت‌کننده می‌تواند کلید عمومی خود را برای هر شرکت‌کنندهٔ دیگری فرستاده و یا آن را برای اطلاع جمعی ارسال نماید. اگرچه این روش مناسب است ولی دارای یک ضعف عمده است. هر شخص دیگری نیز می‌تواند چنین کاری را انجام دهد. یعنی کاربر دیگری می‌تواند تظاهر نماید که کاربر A بوده و یک کلید عمومی را برای کاربر دیگر و یا جمع کاربران ارسال نماید. تا زمانی که کاربر A این تقلب را کشف کرده و به سایر شرکت‌کنندگان اطلاع دهد، فرد متقلب قادر خواهد بود تا تمام پیام‌های رمزنگاری شده به مقصد A را خوانده و از کلیدهای تقلبی برای اعتبارسنجی استفاده کند.

راه حل این مشکل، استفاده از گواهی‌نامهٔ کلید- عمومی است. یک گواهی‌نامه شامل یک کلید عمومی بعلاوه شناسهٔ کاربر(User ID) صاحب کلید است که مجموعهٔ آن بتوسط یک طرف ثالث مورد اعتماد امضاء شده باشد. معمولاً طرف ثالث یک مسئول صدور گواهی‌نامه(Certificate Authority (CA است که همانند یک واحد دولتی و یا یک مؤسسهٔ تجاری، مورد اعتماد جمعیت کاربران است. یک کاربر می‌تواند کلید عمومی خود را با روش امنی به CA ارائه نموده و یک گواهی‌نامه دریافت دارد. کاربر آنگاه می‌تواند این گواهی‌نامه را انتشار دهد. هر کسی که بـــه کلید عمـــومی کاربـــر نیاز دارد می‌تواند این گواهی‌نامه را گرفته و اعتبار آن را تأئید کند. شکل ۱۲-۳ این شیوه را نشان می‌دهد.

روشی که برای صدور گواهی‌نامه‌های کلید- عمومی پذیرش جهانی یافته است، استاندارد X.509 نام دارد. از گواهی‌نامه‌های X.509 در بیشتر کاربردهای امنیت شبکه مثل امنیت IP، لایهٔ سوکت امن (SSL)، اسناد امن الکترونیک (SET)، و S/MIME استفاده می‌شود که همهٔ اینها در فصول بعدی کتاب مورد بحث قرار گرفته است. X.509 بصورت کامل در فصل ۴ بررسی شده است.



شکل ۱۲-۳ استفاده از گواهی‌نامهٔ کلید- عمومی

توزیع کلیدهای سرّی از طریق کلید عمومی

در رمزنگاری سنّتی، نیاز اصلی طرفین برای ارتباط امن این است که یک کلید سرّی را در اشتراک داشته باشند. فرض کنید که Bob بخواهد عملی در رابطه با ارسال پیام انجام دهدکه او را قادر سازد تا با هرشخص دیگری که دسترسی به اینترنت، و یا شبکهٔ دیگری که در اشتراک آن دو است، دارد e-mail امن ردوبدل نماید. فرض کنید که Bob بخواهد تا این عمل را با استفاده از رمزنگاری سنّتی انجام دهد. در رمزنگاری سنّتی، Bob و طرف مقابلش مثلاً Alice بایستی روشی را پیدا کنندکه بتوانند یک کلید سرّی را بدون این‌که کسی متوجه شود ردوبدل نمایند. آنها چگونه باید این کار را انجام دهند؟ اگر Alice در اطاق مجاور Bob باشد، Bob می‌تواند کلید را تهیه کرده و آن را روی یک تکه کاغذ نوشته و یا روی یک دیسکت ذخیره نموده و شخصاً به Alice بدهد. اما اگر Alice در سوی دیگر دنیا باشد، چه باید کرد؟ او می‌تواند این کلید را با استفاده از رمزنگاری متقارن به رمز درآورد و آن را از طریق e-mail برای Alice بفرستد، اما این بدین معنی است که Bob و Alice بایستی برای رمزنگاری این کلید سرّی جدید، یک کلید سرّی مشترک داشته باشند. علاوه برآن Bob و هر کس دیگری که از این بستهٔ نرم‌افزاری e-mail جدید استفاده می‌کند برای ارتباط با هر شخص دیگر دچار همین مشکل‌اند. هرجفت مرتبط، بایستی یک کلید سرّی یکتا در اشتراک داشته باشند.

یک راه حل، مبادلهٔ کلید Diffie-Hellman است. در واقع از این روش استفادهٔ زیادی می‌شود. ولی در این روش یک نقطهٔ ضعف وجود دارد و آن این است که در ساده‌ترین حالت خود Diffie-Hellman هیچ نوع اعتبارسنجی برای دو طرف ارتباط ایجاد نمی‌کند.

راه حل قوی دیگر استفاده از گواهی‌نامه‌های کلید- عمومی است. وقتی Bob می‌خواهد تا با Alice ارتباط برقرار کند می‌تواند چنین کند:

۱- پیامی را آماده نماید.
۲- آن پیام را با یک کلید اجلاس یکبارمصرف بصورت متقارن رمزنگاری نماید.
۳- کلید اجلاس را با استفاده از کلید عمومی Alice و از طریق رمزنگاری کلید- عمومی به رمز درآورد.
۴- کلید اجلاس رمزشده را به پیام وصل کرده و آن را برای Alice بفرستد.

تنها Alice قادر به رمزگشائی کلید اجلاس و بنابراین استخراج پیام اولیه است. اگر Bob کلید عمومی Alice را از طریق گواهی‌نامهٔ کلید- عمومی Alice بدست آورد، آنگاه Bob مطمئن خواهد بود که آن کلید، یک کلید معتبر است.

۳-۷ منابع مطالعاتی

بررسی کامل توابع درهم‌ساز و کُدهای اعتبارسنجی پیام را می‌توان در [STIN06] و [MENE97] پیدا کرد.

آنچه که بعنوان منابع مطالعاتی در فصل ۲ معرفی گردید، هم رمزنگاری سنّتی و هم رمزنگاری کلید- عمومی را پوشش می‌دهند. [DIFF88] بصورت مفصّل چندین تلاش انجام شده برای بکارگیری الگوریتم‌های رمزنگاری دو- کلیدی و تکامل تدریجی تعدادی از پروتکل‌های مبتنی بر آنها را بررسی کرده است. [CORM01] خلاصهٔ خواندنی و مفیدی از تمام الگوریتم‌های مرتبط با تأئید، محاسبه و شکستن رمز RSA را فراهم نموده است.

CORM01 Cormen, T.;Leiserson, C.; Rivest, R.; and Stein, C. *Introduction to Algorithms*. Cambridge, MA: MIT Press, 2001.
DIFF88 Diffie, W. "The First Ten Years of Public- Key Cryptography." *Proceedings of the IEEE*, May 1988. Reprinted in [SIMM92].
MENE97 Menezes, A.; Oorshcot, P.; and Vanstone, S. *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1997.
SIMM92 Simmons, G., ed. *Contemporary Cryptology: The Science of Information Integrity*. Piscataway, NJ: IEEE Press, 1992.
STIN06 Stinson, D. *Cryptography: Theory and Practice*. Boca Raton, FL: CRC Press, 2006.

وِب سایت‌های مفید

- NIST Secure Hashing Page: استانداردهای SHA FIPS و اسناد مرتبط.
- Whirlpool: اطلاعات زیادی در مورد Whirlpool.
- RSA Laboratories: مجموعهٔ مفصلی از مطالب فنی در مورد RSA و سایر عناوین رمزنگاری.

۳-۸ واژه‌های کلیدی، سؤالات مرورکنندهٔ بحث و مسائل

واژه‌های کلیدی

کلید خصوصی	private key	مبادلهٔ کلید DH	Diffie-Hellman key exchange
کلید عمومی	public key	امضاء دیجیتال	digital signature
گواهی‌نامهٔ کلید- عمومی	public-key certificate	استاندارد امضاء دیجیتال	Digital Signature Standard(DSS)
رمزنگاری کلید- عمومی	public-key encryption	رمزنگاری خَم بیضوی	elliptic-curve cryptography(ECC)
نوعی الگوریتم درهم‌سازی	RIPEMD-160	نوعی الگوریتم درهم‌سازی	HMAC
مشهورترین الگوریتم رمزنگاری کلید- عمومی	RSA	مبادلهٔ کلید	key exchange
کلید سرّی	secret key	نوعی الگوریتم درهم‌سازی	MD5
تابع درهم‌ساز آمن	secure hash function	اعتبارسنجی پیام	message authentication
نوعی الگوریتم درهم‌سازی	SHA-1	کُد اعتبارسنجی پیام	message authentication code (MAC)
مقاومت قوی در برابر تصادم	strong collision resistance	چکیدهٔ پیام	message digest
مقاومت ضعیف در برابر تصادم	weak collision resistance	تابع درهم‌ساز یک- طرفه	one-way hash function

سوالات مرورکننده بحث

۳-۱ سه روش برخورد با اعتبارسنجی پیام را نام ببرید.

۳-۲ کُد اعتبارسنجی پیام چیست؟

۳-۳ سه روشی که در شکل ۳-۲ نشان داده شده است را بطور مختصر تشریح کنید.

۳-۴ یک تابع درهم‌ساز چه خواصی داشته باشد تا برای اعتبارسنجی پیام مفید باشد؟

۳-۵ در مقولهٔ یک تابع hash، یک تابع فشرده‌ساز چیست؟

۳-۶ اجزاء اصلی یک سیستم رمزنگاری کلید- عمومی کدامند؟

۳-۷ سه مورد استفاده از یک سیستم رمزنگاری کلید- عمومی را نام برده و توضیح دهید.

۳-۸ اختلاف بین یک کلید خصوصی با یک کلید سرّی در چیست؟

۳-۹ امضاء دیجیتال چیست؟

۳-۱۰ یک گواهی‌نامهٔ کلید- عمومی چیست؟

۳-۱۱ چگونه می‌توان از رمزنگاری کلید- عمومی برای توزیع کلید استفاده کرد؟

مسائل

۳-۱ یکی از پراستفاده‌ترین MACها که الگوریتم اعتبارسنجی دیتا (Data Authentication Algorithm) خوانده می‌شود مبتنی بر DES است. این الگوریتم هم از انتشارات FIPS بوده (FIPS PUB 113)، و هم استاندارد ANSI است (X9.17). می‌توان چنین تعریف کرد که الگوریتم از مُود عملیاتی زنجیره‌ای رمز قالبی(CBC) با بردار اولیهٔ صفر استفاده می‌کند (شکل ۹-۲). دیتائی که قرار است اعتبارسنجی گردد (مثل پیام، رکورد، فایل یا برنامه) به بلوک‌های ۶۴- بیتی مجاور هم P_1، P_2، و P_N، تقسیم می‌شود. اگر لازم باشد بلوک نهائی با صفرهائی در سمت راست پر شده تا ۶۴ بیت آن کامل شود. MAC یا شامل بلوک رمزشدهٔ کامل C_N و یا M بیت سمت چپی بلوک است ($16 \le M \le 64$). نشان دهید که همین نتیجه را می‌توان با استفاده از مُود فیدبک رمز (CFB) نیز به دست آورد.

۳-۲ یک تابع ۳۲- بیتی hash را بصورت جمع رشته‌ای دو تابع ۱۶- بیتی XOR و RXOR که در بخش ۲-۳ بعنوان « دو تابع سادهٔ hash » تعریف شده است، در نظر بگیرید.

الف- آیا این جمع کنترلی(checksum)، تمام خطاهائی که بعلت تغییر تعداد فردی از بیت‌ها حاصل می‌شود را تشخیص می‌دهد؟ توضیح دهید.

ب- آیا این جمع کنترلی، تمام خطاهائی که بعلت تغییر تعداد زوجی از بیت‌ها حاصل می‌شود را تشخیص می‌دهد؟ اگر اینطور نیست، الگوی خطاهائی که باعث شکست این جمع کنترلی می‌شود را مشخص کنید.

ج- نسبت به مؤثربودن این تابع درهم‌ساز برای استفاده بعنوان یک تابع hash اعتبارسنجی نظر دهید.

۳-۳ فرض کنید که $H(m)$ یک تابع درهم‌ساز مقاوم در برابر تصادم بوده که یک پیام با طول هرچند بیت را به به یک اندازهٔ hash با طول n- بیت نگاشت می‌کند.آیا این درست است که برای تمام پیام‌های x و x' که $x \neq x'$ است، $H(x) \neq H(x')$ است؟ پاسخ خود را تشریح کنید.

۳-۴ **الف**- تابع درهم‌ساز زیر را در نظر بگیرید. پیام‌ها بصورت ردیفی از اعداد دهدهی هستند، $M = (a_1, a_2, \ldots, a_t)$. اندازهٔ hash بصورت $\left(\sum_{i=1}^{t} a_i\right) \bmod n$ برای یک مقدار n که از قبل تعریف شده است محاسبه می‌شود. آیا این تابع درهم‌ساز هیچیک از لازمه‌های یک تابع درهم‌ساز که در بخش ۲-۳ لیست شده است را ارضاء می‌کند؟ پاسخ خود را توضیح دهید.

ب- قسمت (الف) را برای تابع $h = \left(\sum_{i=1}^{t} (a_i)^2\right) \bmod n$ تکرار کنید.

ج- تابع hash قسمت (ب) را برای $M = (189,632,900,722,349)$ و $n = 989$ محاسبه کنید.

۳-۵ این مسأله یک تابع درهم‌ساز شبیه به SHA را معرفی می‌کند که بجای عمل روی دیتای باینری بر روی حروف عمل می‌کند. این تابع بنام *toy tetragraph hash* (tth) نامیده می‌شود. اگر پیامی که شامل ردیفی از حروف است را داشته باشیم، tth یک اندازهٔ hash که شامل ۴ حرف است را تولید می‌کند. در ابتدا tth پیام را با صرفنظرکردن جاهای خالی بین کلمات، علائم و حروف بزرگ، بصورت بلوک‌های ۱۶- حرفی درمی‌آورد. اگر طول پیام بر ۱۶ قابل قسمت نباشد، لائی null به اندازهٔ لازم به انتهای آن اضافه می‌شود. یک دنباله چهارتائی عددی که با اندازهٔ (0,0,0,0) آغاز می‌شود پیوسته نگهداری می‌گردد. این دنباله برای پردازش اولین بلوک، در ورودی یک تابع فشرده‌ساز قرار می‌گیرد. تابع فشرده‌ساز شامل دو مرحله است. **مرحلهٔ ۱** : بلوک بعدی متن را گرفته و آن را بصورت یک بلوک ۴ × ۴ ردیفی درآورده و به عدد تبدیل کنید (A = 0, B = 1 و غیره). مثلاً برای بلوک ABCDEFGHIJKLMNOP خواهیم داشت:

A	B	C	D
E	F	G	H
I	J	K	L
M	N	O	P

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

آنگاه هر ستون را بصورت mod 26 جمع کرده و نتیجه را با دنبالهٔ عددی چهارتائی باز هم بصورت mod 26 جمع کنید. در این مثال دنبالهٔ چهارتائی جدید بدست آمده برابر (24,2,6,10) می‌شود. **مرحلهٔ ۲** : از ماتریس مرحلهٔ ۱ استفاده کرده و اولین ردیف را ۱ خانه به چپ، دومین ردیف را ۲ خانه به چپ ، سومین ردیف را ۳ خانه به چپ چرخانده و ترتیب ردیف آخر را معکوس کنید. در مورد مثال ما:

B	C	D	A
G	H	E	F
L	I	J	K
P	O	N	M

1	2	3	0
6	7	4	5
11	8	9	10
15	14	13	12

حال، هر ستون را بصورت mod 26 جمع کرده و نتیجه را نیز بهمین صورت به دنبالهٔ چهارتائی مرحلهٔ قبل اضافه کنید. دنبالهٔ چهارتائی جدید (5,7,9,11) است. این دنباله حالا برای پردازش بلوک بعدی متن در ورودی تابع فشرده‌سازی مرحلهٔ ۱ قرار خواهد گرفت. پس از اینکه آخرین بلوک پیام پردازش گردید، دنبالهٔ چهارتائی نهائی را به حروف تبدیل کنید. برای مثال اگر پیام ABCDEFGHIJKLMNOP بوده است، اندازهٔ hash مقدار FHJL خواهد شد.

الف- شکل‌هائی قابل مقایسه با شکل‌های ۳-۴ و ۳-۵ رسم کنید تا منطق کلی tth و منطق تابع فشرده‌سازی را نشان دهد.

ب- تابع hash پیام ۴۸- حرفی "I leave twenty million dollars to my friendly cousin Bill" را محاسبه کنید.

ج- برای اینکه ضعف tth آشکار شود، یک بلوک ۴۸- حرفی دیگر که همان اندازهٔ hash قسمت (ب) را ایجاد کند پیدا کنید. *راهنمائی:* از حرف A زیاد استفاده کنید.

۳-۶ این امکان وجود دارد که از یک تابع درهم‌ساز برای ساخت یک رمز قالبی با ساختاری مشابه DES استفاده کرد. با توجه به این‌که یک تابع درهم‌ساز، یک- طرفه بوده ولی یک رمز قالبی بایستی برگشت‌پذیر باشد (برای رمزگشائی)، چگونه این امر ممکن است؟

۳-۷ قبل از کشف روش‌های رمزنگاری کلید- عمومی، مثل RSA، اثبات شده بود که رمزنگاری کلید- عمومی در تئوری می‌تواند وجود داشته باشد. توابع $f_1(x_1) = z_1$، $f_2(x_2,y_2) = z_2$ و $f_3(x_3,y_3) = z_3$ که در آنها تمام مقادیر اعداد صحیح و $1 \le x_i, y_i, z_i \le N$ می‌باشند را در نظر بگیرید. تابع f_1 را می‌توان با بردار M1 با طول N که درآن k امین عنصر اندازهٔ $f_1(k)$ است، نشان داد. بطریق مشابه، f_2 و f_3 را می‌توان با ماتریس‌های M2 و M3 که ماتریس‌های $N \times N$ هستند، نمایش داد. هدف این است که عمل رمزنگاری/ رمزگشائی را با مراجعه به جداولی که این جداول دارای اندازهٔ بسیار بزرگ N هستند نشان داد. این جداول بطور غیرعملی بسیار بزرگ بوده ولی از نظر تئوری می‌توانند ساخته شوند. روش چنین است: M1 را با جایگشت تصادفی تمام اعداد صحیح بین 1 و N بسازید. یعنی هر عدد صحیح فقط تنها یکبار در M1 وارد شود. M2 را چنان بسازید که هر ردیف شامل جایگشت تصادفی N عدد صحیح قبلی باشد. بالاخره M3 را چنان بسازید که شرط زیر را ارضاء کند:

برای تمام مقادیر k و p با $1 \le k, p \le N$ داشته باشیم: $f3(f2(f1(k),p),k) = p$

عبارت بالا با کلمات چنین بیان می‌شود:

۱- M1 یک ورودی k را گرفته و خروجی x را تولید می‌کند.

۲- M2 ورودی‌های x و p را گرفته و خروجی z را تولید می‌کند.

۳- M3 ورودی‌های z و k را گرفته و خروجی p را تولید می‌کند.

وقتی سه جدول ساخته شدند، انتشار می‌یابند.

الف- بایستی روشن باشد که می‌توان M3، بطوری که شرایط قبل را ارضاء کند، تشکیل داد. بعنوان مثال M3 را در حالت سادهٔ زیر بسازید:

M1 =

5
4
2
3
1

M2 =

5	2	3	4	1
4	2	5	1	3
1	3	2	4	5
3	1	4	2	5
2	5	3	4	1

M3 =

قرارداد: عنصر i ام M1 متناظر با $k = i$ است. ردیف i ام M2 متناظر با $x = i$ است و ستون j ام M2 متناظر با $p = j$ است. ردیف i ام M3 متناظر با، $z = i$ و ستون j ام M3 متناظر با $k = j$ است.

ب- استفاده از این مجموعهٔ جداول برای انجام عمل رمزنگاری و رمزگشائی بین دو کاربر را توضیح دهید.

ج- استدلال نمائید که این یک روش امن است.

۳-۸ با استفاده از الگوریتم RSA، رمزنگاری و رمزگشائی، همانند شکل ۹-۳، برای مقادیر زیر را انجام دهید:

الف- $p = 3;\quad q = 11, e = 7;\quad M = 5$

ب- $p = 5;\quad q = 11, e = 3;\quad M = 9$

ج- $p = 7;\quad q = 11, e = 17;\quad M = 8$

د- $p = 11;\quad q = 13, e = 11;\quad M = 7$

هـ- $p = 17;\quad q = 31, e = 7;\quad M = 2$

راهنمائی: رمزگشائی آنچنان که تصور می‌شود سخت نیست. کمی زیرکی بکار برید.

۳-۹ در یک سیستم کلید- عمومی که از RSA استفاده می‌کند، متن رمزشده $C = 10$ را که برای کاربری با کلید عمومی $n = 35$ و $e = 5$ ارسال شده است استراق سمع می‌کنید. متن سادهٔ M چیست؟

۳-۱۰ در یک سیستم RSA، کلید عمومی یک کاربر $n = 3599$ و $e = 31$ است. کلید خصوصی این کاربر چیست؟

۳-۱۱ فرض کنید که یک سری بلوک‌هائی در دسترس‌اند که با الگوریتم RSA کُد شده‌اند ولی کلید خصوصی را در اختیار نداریم. فرض کنید $n = pq$ و e کلید عمومی است. همچنین فرض کنید که شخصی بما اطلاع می‌دهد که یکی از بلوک‌های متن ساده دارای فاکتور مشترکی با n است. آیا این امر بهرترتیب کمکی به ما می‌کند؟

۳-۱۲ نشان دهید که چگونه RSA می‌تواند با ماتریس‌های M2،M1، و M3 مسألهٔ ۷-۳ نشان داده شود.

۳-۱۳ روش زیر را در نظر بگیرید:

۱- عدد فرد E را انتخاب کنید.

۲- دو عدد اول P و Q را طوری انتخاب کنید که $(P\text{-}1)(Q\text{-}1)$-1 بطور مساوی قابل تقسیم به E باشد.

۳- P و Q را ضرب کنید تا N بدست آید.

۴- $D = [(P\text{-}1)(Q\text{-}1)(\text{E-}1) + 1] / E$ را حساب کنید.

آیا این روش معادل RSA است؟ پاسخ خود را توجیه کنید.

۳-۱۴ استفاده از RSA با یک کلید شناخته شده را برای ساخت یک تابع درهم‌ساز یک- طرفه در نظر بگیرید. آنگاه یک پیام که شامل دنباله‌ای از بلوک‌هاست را بصورت زیر پردازش کنید: بلوک اول را رمزنگاری نمائید. نتیجه را با بلوک دوم XOR نموده و مجدداً رمزنگاری کنید و بهمین ترتیب ادامه دهید. با حل مسألهٔ زیر نشان دهید که این روش امن نیست: اگر یک پیام شامل دو بلوک B1 و B2 بوده و کُد hash آن چنین باشد

$$\text{RSAH (B1,B2)} = \text{RSA (RSA (B1)} \oplus \text{B2)}$$

اگر بلوک اختیاری C1 داده شده باشد، C2 را چنان اختیار کنید که RSAH(C1, C2) = RSAH(B1, B2). بنابراین تابع درهم‌ساز شرط مقاومت ضعیف در برابر تصادم را ارضاء نمی‌کند.

۳-۱۵ فرض کنید که Bob از سیستم رمزنگاری RSA با یک مدول بسیار بزرگ n استفاده می‌کند که فاکتور کردن آن در زمان معقول قابل تصور نیست. فرض کنید که Alice پیامی را برای Bob می‌فرستد که در آن هر یک از حروف الفباء با یک عدد بین صفر و ۲۵ نمایش داده شده و سپس هر عدد بطور مجزا با الگوریتم RSA با e بزرگ و n بزرگ رمزنگاری شده است. آیا این روش امن است؟ اگر جواب منفی است، بهره‌ورترین روش حمله بر ضد این نوع رمزنگاری چیست؟

۳-۱۶ یک روش Diffie-Hellman با یک عدد اول $q = 11$ و ریشهٔ اولیهٔ آن $\alpha = 2$ را درنظر بگیرید.

الف- اگر کاربر A دارای کلید عمومی $Y_A = 9$ باشد، کلید خصوصی X_A مربوط به این کاربر چیست؟

ب- اگر کاربر B دارای کلید عمومی $Y_B = 3$ باشد، کلید مشترک سرّی K چیست؟

قسمت دوم

کاربردهای امنیت شبکه

در قسمت اول، رمزهای مختلف را بررسی کرده و به استفاده از آنها برای محرمانگی، اعتبارسنجی، مبادلهٔ کلید و وظایف مرتبط با این کاربردها اشاره نمودیم. قسمت دوم ابزارهای مهم امنیت شبکه و کاربردهائی که از این ابزارها استفاده می‌کنند را مورد بررسی قرار می‌دهد. این ابزارها را می‌توان در یک شبکهٔ منفرد، اینترانت یک سازمان، و یا اینترنت بکاربرد.

فصل ۴ کاربردهای اعتبارسنجی

فصل ۴ به بررسی دو عنوان از مهم‌ترین مشخصه‌های اعتبارسنجی زمان حاضراختصاص دارد. Kerberos یک پروتکل اعتبارسنجی مبتنی بر رمزنگاری متقارن بوده که حمایت و کاربرد گسترده‌ای در سیستم‌های متنوع دارد. X.509 یک الگوریتم اعتبارسنجی را تعیین کرده و یک تسهیلات گواهی‌نامه‌ای را فراهم می‌سازد. این تسهیلات کاربران را قادر می‌سازد تا گواهی‌نامه‌های کلید- عمومی را طوری به دست آوردند که یک جمعیت از کاربران به اعتبار کلیدهای عمومی اعتماد داشته باشند. این تسهیلات زیربنای برخی از کاربردهاست.

فصل ۵ امنیت پست الکترونیک

پست الکترونیک پراستفاده‌ترین کاربرد توزیع شده بوده و تمایل فزاینده‌ای در مورد فراهم آوردن سرویس‌های اعتبارسنجی و محرمانگی بعنوان بخشی از تسهیلات پست الکترونیک بوجود آمده است. فصل ۵ به دو روش که احتمالاً در بخش امنیت پست الکترونیک حاکمیت خواهند یافت نگاه می‌کند. Pretty Good Privacy(PGP) یک روش پراستفاده است که متکی بر هیچ مقام مسئول و یا سازمانی نیست. در نتیجه این روش همان قدر که برای پیکربندی شبکه‌هائی که بتوسط سازمان‌ها اداره می‌شوند کارآئی دارد، در مورد مصارف فردی نیز دارای استفاده است. (Secure/Mutipurpose Internet Mail Extensions) S/MIME صرفاً بعنوان یک استاندارد اینترنت طراحی شده است.

فصل ۶ امنیت IP

پروتکل اینترنت (IP) عنصر مرکزی اینترنت و اینترانت‌های خصوصی است. در نتیجه امنیت سطح IP برای طراحی هر روش امنیتی مبتنی بر عملیات بین‌شبکه‌ای مهم است. فصل ۶ نگاهی به روش امنیت IP انداخته که به منظور کار با IP جاری و IP نسل بعد که IPv6 خوانده می‌شود طراحی شده است.

فصل ۷ امنیت WEB

رشد انفجارگونهٔ استفاده از تارجهان‌گستر برای تجارت الکترونیک و انتشار همه جانبهٔ اطلاعات باعث شده است تا نیاز مبرمی برای استقرار یک امنیت قوی مبتنی بر وِب بوجود آید. فصل ۷ این مورد جدید امنیتی و مهم را مورد بررسی قرار داده و به دو استاندارد کلیدی یعنی لایهٔ سوکت امن (SSL) و اسناد الکترونیکی امن (SET) نظر می‌کند.

فصل ۸ امنیت مدیریت شبکه

با استفادهٔ روزافزون از سیستم‌های مدیریت شبکه برای کنترل شبکه‌های متنوع، نیاز فزاینده‌ای به استقرار قابلیت‌های امنیت در این سیستم‌ها بوجود آمده است. فصل ۸ بر پراستفاده‌ترین روش مدیریت شبکه، یعنی پروتکل سادهٔ مدیریت شبکه (SNMP) متمرکز است. نسخهٔ اول SNMP تنها یک تسهیلات اعتبارسنجی ابتدائی مبتنی بر کلمهٔ عبور دارد. SNMPv2 قابلیت‌های فراوان‌تری را بوجود آورده و SNMPv3 یک تسهیلات امنیتی فراگیر برای محرمانگی و اعتبارسنجی ایجاد می‌کند که می‌تواند به همراه با SNMPv1 و SNMPv2 بکار رود.

فصل ۴

کاربردهای اعتبارسنجی

ین فصل برخی از عملیات اعتبارسنجی که برای پشتیبانی از اعتبارسنجی سطح کاربرد و امضاء دیجیتال طراحی شده است را بررسی می‌کند.

بحث را با نگاهی به یکی از ابتدائی‌ترین سرویس‌ها، که پراستفاده‌ترین آنها نیز بوده است و Kerberos خوانده می‌شود، آغاز می‌کنیم. سپس سرویس اعتبارسنجی فهرست راهنمای X.509 را مورد مطالعه قرار می‌دهیم. این استاندارد بعنوان بخشی از سرویس فهرست راهنما که این استاندارد حامی آن است دارای اهمیت بوده ولی مهم‌تر این‌که بعنوان خشت اصلی مورد استفاده در سایر استانداردها، همانند S/MIME، که در فصل ۵ از آن یاد خواهد شد نیز دارای کاربرد است. در پایان مفهوم زیرساخت کلید- عمومی (PKI) را بررسی می‌کنیم.

۴-۱ KERBEROS

Kerberos یک سرویس اعتبارسنجی است که بعنوان بخشی از پروژهٔ آتنه (Athena) در دانشگاه MIT طراحی شده است. مشکلی را که Kerberos مورد توجه قرار می‌دهد چنین است: یک محیط گستردهٔ باز را در نظر بگیرید که در آن کاربرانی که در ایستگاه‌های مختلف کاری حضور دارند، علاقه‌مند به دست‌یابی به سرویس‌های مختلفی که روی سِرورهای متعدد کل شبکه قرار دارند می‌باشند. ما تمایل داریم که سِرورها بتوانند دست‌یابی‌های مربوط به کاربران معتبر را بمیزان دلخواه محدود نموده، و همچنین قادر باشند اعتبار کاربر متقاضی سرویس را بسنجند. در چنین محیطی، نمی‌توان به یک ایستگاه کاری از نظر شناسایی صحیح کاربران خودش برای دست‌یابی به سرویس‌های شبکه اعتماد کرد. علی‌الخصوص سه تهدید زیر همیشه وجود دارند:

- یک کاربر ممکن است به یک ایستگاه کاری بخصوص دسترسی یافته و چنین وانمود کند که کاربر دیگری است که از آن ایستگاه تماس گرفته است.
- یک کاربر ممکن است آدرس شبکهٔ یک ایستگاه کاری را طوری تغییر دهد که باعث شود تقاضاهایی که از این ایستگاه ارسال می‌شوند، بر حسب ظاهر مربوط به ایستگاه دیگری تلقی گردند.
- یک کاربر ممکن است با عمل شنود روی یک خط، حمله‌ای از نوع بازخوانی (replay) انجام داده، وارد سِرور شده و یا عملیات را مختل سازد.

در هریک از این موارد، یک کاربر غیرمعتبر ممکن است به سرویس‌ها و داده‌هایی دست یابد که مجاز به دست‌یابی به آنها نیست. بجای قراردادن پروتکل‌های اعتبارسنجی دشوار در هر سِرور، Kerberos یک سِرور اعتبارسنجی متمرکز که وظیفهٔ آن معرفی کاربران به سِرورها، و سِرورها به کاربران است را فراهم می‌سازد. برخلاف اغلب روش‌های اعتبارسنجی معرفی شده در این کتاب، Kerberos منحصراً بر رمزنگاری متقارن متکی بوده و از رمزنگاری کلید- عمومی استفاده نمی‌کند.

دو نسخه از Kerberos دارای استفادهٔ وسیع‌اند. نسخهٔ ۴ [MILL88,STEI88] هنوز بطور گسترده‌ای مورد استفاده است. نسخهٔ ۵ [KOHL94] بعضی از کمبودهای امنیتی نسخهٔ ۴ را جبران کرده و بعنوان یک استاندارد اینترنت (RFC 1510) پیشنهاد شده است.

این بخش را با بحث مختصری در زمینهٔ انگیزش‌های مربوط به روش Kerberos آغاز می‌کنیم. آنگاه نظر به پیچیدگی Kerberos، موضوع را با بررسی پروتکل اعتبارسنجی بکار رفته درنسخهٔ ۴ شروع می‌کنیم. این موضوع ما را قادر می‌سازد تا جوهر استراتژی Kerberos، بدون نیاز به دانستن جزئیات لازم مربوط به حملات امنیتی هوشمندانه را ملاحظه نمائیم. در نهایت، نسخهٔ ۵ را بررسی خواهیم کرد.

انگیزش

اگر مجموعه‌ای از کاربران دارای رایانه‌های شخصی مخصوص به خود که به هیچ شبکه‌ای متصل نیستند وجود داشته باشند، آنگاه منابع و فایل‌های یک کاربر را می‌توان از طریق حفاظت فیزیکی رایانهٔ او حفاظت کرد. اما وقتی این کاربران از طریق یک سیستم مرکزی با اشتراک زمانی بهم متصل می‌شوند، سیستم عامل اشتراک زمانی مسئول حفاظت مجموعه خواهد بود. سیستم عامل می‌تواند خط‌مشی‌های مربوط به کنترل دست‌یابی را، بر مبنای هویت کاربر، اعمال کرده و روش‌هائی را برای شناسائی کاربران و احراز هویت آنها در زمان اتصال به سیستم به اجرا بگذارد.

امروزه هیچکدام از این سناریوها معمول نمی‌باشند. معمول‌تر این‌است که یک معماری توزیع شده که شامل ایستگاه‌های کاری تخصیص‌یافته برای کاربران (کلاینت‌ها) و سِرورهای توزیع‌شده و یا متمرکز است، وجود داشته باشد. در چنین محیطی، سه برخورد متفاوت با مسألهٔ امنیت را می‌توان تصور کرد:

۱- اتکاء به ایستگاه‌های کاری کلاینت برای شناسائی کاربر یا کاربرانی که میخواهند به آن وصل شوند و اتکاء به هر سِرور برای اجرای خط‌مشی‌های امنیتی بر مبنای شناسائی کاربر (ID).
۲- الزام سیستم‌های کلاینت به معرفی خود در هنگام اتصال به سِرور و اتکاء به سیستم کلاینت برای شناسائی کاربری که می‌خواهد به آن وصل شود.
۳- الزام کاربر به اثبات هویت خود برای هر سرویس درخواستی و همچنین الزام سِرورها به اثبات هویت خود برای کلاینت‌ها.

در یک محیط کوچک بسته که در آن تمام سیستم‌ها متعلق به یک سازمان منفرد بوده و بتوسط همان سازمان اداره می‌شوند، استراتژی‌های اول و یا شاید دوم کفایت می‌کنند. ولی در یک محیط بازتر که در آن از اتصالات شبکه‌ای برای ارتباط ماشین‌ها با یکدیگر استفاده می‌شود، روش سوم برای حفاظت اطلاعات کاربر و منابع مستقر در سِرورها مناسب‌تر است. روش سوم همانست که Kerberos از آن پشتیبانی می‌کند. Kerberos بر مبنای یک معماری کلاینت/ سِرور عمل کرده و از یک یا چند سِرور Kerberos برای فراهم‌آوردن سرویس اعتبارسنجی، یا تشخیص هویت، استفاده می‌نماید.

اولین گزارش منتشرشده در موردKerberos [STEI88]، الزامات زیر را برای Kerberos بیان نموده است:

- **امن**: یک عامل شنود در شبکه نبایستی بتواند اطلاعات لازم برای جعل هویت یک کاربر را بدست آورد. به عبارت کلی‌تر، Kerberos بایستی آنقدر مستحکم باشد که یک دشمن قوی او را ضعیف نشمارد.
- **قابل اعتماد**: برای تمام سرویس‌هائی که برای کنترل دست‌یابی به Kerberos متکی هستند، عدم دسترسی به سرویس Kerberos به مفهوم عدم دسترسی به همه آنهاست. بنابراین Kerberos باید دارای قابلیت اعتماد بالا بوده و بایستی از یک معماری توزیع‌شده استفاده کند تا در صورت وجود مشکل، یک سیستم بتواند پشتیبان سیستم دیگر گردد.
- **شفاف**: در حالت ایده‌ال، کاربر نبایستی بجز وارد کردن کلمهٔ عبور متوجه شود که عملیات اعتبارسنجی صورت می‌پذیرد.
- **مقیاس‌پذیر**: سیستم بایستی قادر به حمایت از تعداد زیادی کلاینت و سِرور باشد. این نیاز، یک معماری توزیع‌شده و پودمانی را پیشنهاد می‌کند.

برای برآورده نمودن این نیازها، شِگرد Kerberos همان استفاده از یک سرویس اعتبارسنجی قابل اعتماد شخص ثالث است که بر مبنای پروتکلی که بتوسط Needham و Schroeder [NEED78] پیشنهاد شده است، قرار دارد. این سرویس از آن‌جهت بایستی قابل اعتماد باشد که کلاینت‌ها و سِرورها برای اعتبارسنجی و تشخیص هویت یکدیگر به میانداری Kerberos تکیه می‌کنند. با فرض اینکه پروتکل Kerberos خوب طراحی شده باشد، آنگاه سرویس اعتبارسنجی در صورتی امن است که خود سِرور Kerberos امن باشد.

نسخهٔ چهارم Kerberos

نسخهٔ چهارم Kerberos از DES که پروتکل نسبتاً پیچیده‌ای است استفاده کرده تا سرویس اعتبارسنجی را فراهم نماید. با نگاهی کلی به پروتکل، فهم نیاز به آن همه جزئیاتی که در آن منظور شده است کار آسانی نیست. بنابراین ما با استفاده از استراتژی بکار گرفته شده بتوسط Bill Bryant در پروژه آتنه[BRYA88]، سعی می‌کنیم تا ابتدا با نگاهی به چند دیالوگ فرضی، پروتکل کامل را بنا نمائیم. هر دیالوگ جدید، برای غلبه کردن بر نقاط آسیب‌پذیر امنیتی دیالوگ قبلی، پیچیدگی‌های جدیدی را در پروتکل ایجاد می‌کند.

پس از بررسی پروتکل، به سایر جنبه‌های نسخهٔ ۴ نیز نگاهی می‌اندازیم.

یک دیالوگ سادهٔ اعتبارسنجی

در محیط حفاظت‌نشدهٔ یک شبکه، هر کلاینت میتواند برای دریافت سرویس به هر سِروری مراجعه نماید. در این حالت ریسک امنیتی آشکار، جعل هویت است. یک دشمن می‌تواند خود را بجای کلاینت دیگری جازده و امتیازات غیرقانونی از سِرورها کسب نماید. برای مقابله با این تهدید، سِرورها بایستی بتوانند هویت کلاینت‌هائی که درخواست سرویس دارند را تأیید نمایند. هر سِرور را می‌توان مجبور کرد تا این وظیفه را برای هربار مبادلهٔ کلاینت/ سِرور انجام دهد ولی در یک محیط باز، این درخواست بار سنگینی را به دوش هر سِرور قرار می‌دهد.

راه دیگر این‌است که از یک سِرور اعتبارسنج Authentication Server (AS) استفاده کرد که کلمات عبور تمام کاربران را دانسته و آنها را در یک پایگاه متمرکز داده ذخیره نماید. علاوه براین، AS با هر سِرور دیگر یک کلید سرّی یکتا را به اشتراک می‌گذارد. این کلیدها بصورت فیزیکی و یا بصورت امن دیگری توزیع شده‌اند. به دیالوگ فرضی زیر توجه کنید:

(۱) C→AS: $ID_C \| P_C \| ID_V$

(۲) AS→C: $Ticket$

(۳) C→V: $ID_C \| Ticket$

$Ticket = \mathrm{E}\,(K_V\,,\,[\,ID_C \| AD_C \| ID_V\,]\,)$

که در آن

C	=	کلاینت
AS	=	سِرور اعتبارسنج
V	=	سِرور
ID_C	=	شناسهٔ کاربر روی C
ID_V	=	شناسهٔ V
P_C	=	کلمهٔ عبور کاربر روی C
AD_C	=	آدرس شبکه C
K_V	=	کلید سرّی رمزنگاری مشترک بین AS و V
$\|$	=	جمع رشته‌ای

در این سناریو،کاربر به یک ایستگاه کاری متصل شده و درخواست دسترسی به سِرور V را می‌نماید. مدول کلاینت در ایستگاه کاری، از کاربر درخواست کلمهٔ عبور نموده و سپس پیامی را به AS میفرستدکه شامل ID کاربر، ID سِرور و کلمهٔعبور کاربر است. AS در پایگاه دادهٔ خود جستجو کرده تا ببیند آیا کاربر کلمهٔعبور صحیح برای ID خود را عرضه کرده است و آیا دست‌یابی این کاربر به سِرور V مجاز می‌باشد. اگر هر دو جواب مثبت باشد، AS کاربر را به عنوان یک کاربر معتبر شناخته و حال بایستی سِرور را متقاعد سازد که این کاربر معتبر است. برای این کار، AS بلیتی (ticket) را آماده می‌سازد که شامل ID کاربر، آدرس شبکه و ID سِرور است. این بلیت بتوسط کلید رمزی که بین AS و سِرور مشترک است، رمزنگاری می‌شود. سپس این بلیت برای C بازگردانده می‌شود. چون بلیت رمزنگاری شده است، نه می‌تواند بتوسط C و نه بتوسط یک دشمن تغییر یابد.

با این بلیت، حالا C می‌تواند برای سرویس به V مراجعه کند. C پیامی را برای V می‌فرستد که شامل ID خود C و بلیت است.V بلیت را رمزگشائی کرده و تأئید می‌نماید که ID کاربر که در بلیت وجود دارد مشابه ID رمزنشده موجود در پیام است. اگر این دو با هم تطبیق نمایند، سِرور فرض را بر این می‌گذارد که کاربر دارای هویت معتبر بوده و سرویس درخواستی را در اختیار او قرار می‌دهد.

هریک از مؤلفه‌های پیام شمارهٔ (۳) دارای اهمیت ویژه‌ای است. بلیت برای جلوگیری از تغییر و یا جعل، به رمز در می‌آید. ID سِرور (ID_V) در بلیت جای میگیرد تا سِرور بتواند تأئید کند که رمزگشائی بلیت صحیح انجام شده است. ID_C در بلیت جای دارد تا نشان دهد که این بلیت بخاطر C صادر شده است. بالاخره AD_C بمنظور مقابله با تهدید زیر مورد

استفاده قرار گرفته است. یک دشمن ممکن است بلیت ارسال شده بهمراه پیام (۲) را تصرف کرده، از نام ID_C استفاده کرده و یک پیام بشکل (۳) را از ایستگاه کاری دیگری ارسال کند. در این صورت سِرور یک بلیت معتبر که با ID کاربر تطبیق دارد را دریافت کرده و دست‌یابی را به کاربر، ولی روی ایستگاه کاری دیگر، اِعطا می‌کند. برای جلوگیری از این حمله، AS آدرس شبکه‌ای را که تقاضای اولیه از آن صادر شده بود در بلیت قرار می‌دهد. حال بلیت تنها وقتی معتبر است که از همان ایستگاه کاری ارسال شود که در بدو امر تقاضای بلیت کرده بود.

یک دیالوگ اعتبارسنجی امن‌تر

اگرچه سناریوی قبل تعدادی از مشکلات اعتبارسنجی در یک محیط شبکه‌ای باز را حل می‌کند ولی بازهم مشکلاتی باقی‌است که دوتای آنها علی‌الخصوص قابل توجه است. اولاً علاقه‌مندیم که تعداد دفعاتی که یک کاربر مجبور به وارد نمودن کلمهٔ عبور خود است را به حداقل برسانیم. فرض کنید که از هر بلیت صادر شده تنها یکبار بتوان استفاده کرد. اگر کاربرC در صبح یک روز کاری به ایستگاه متصل شده و بخواهد نامه‌های خود را در یک سِرور پستی مشاهده نماید، C بایستی کلمهٔ عبور خود را عرضه کرده تا یک بلیت برای سِرور پستی به او داده شود. اگر C بخواهد در طول روز چندین مرتبه نامه‌های خود را کنترل کند، برای هربار تلاش نیاز به عرضه نمودن مجدد کلمهٔ عبور دارد. می‌توان وضعیت را بهبود بخشید اگر بتوان کاری کرد که یک بلیت برای دفعات دیگر نیز قابل استفاده باشد. برای یک بار اتصال و گفتگو، ایستگاه کاری می‌تواند بلیت سِرور پستی را پس از دریافت ذخیره کرده و آن را برای دسترسی به سِرور پستی به دفعات از سوی کاربر مورد استفاده قرار دهد.

ولی در تحت این شرایط، بازهم کاربر برای درخواست هر سرویس جدید نیاز به یک بلیت جدید خواهد داشت. اگر کاربر بخواهد به یک سِرور چاپگر، یک سِرور پستی و یک سِرور فایل دسترسی یابد، برای اولین دسترسی به هریک از اینها نیاز به یک بلیت جدید داشته که در نتیجه برای کسب هر بلیت بایستی کلمهٔ عبور خود را ارائه دهد.

مشکل دوم این‌است که در سناریوی قبل، کلمهٔ عبور به صورت یک متن ساده و رمزنشده انتقال می‌یافت [پیام(۱)]. یک استراق‌سمع‌کننده می‌تواند کلمه عبور را گرفته و از هر سرویسی که قربانی، مجاز به دسترسی به آن بوده است استفاده نماید.

برای حل این مشکلات اضافی، روشی برای جلوگیری از انتقال کلمه عبور بصورت متن ساده، و همچنین یک سِرور جدید بنام سِرور اعطاکنندهٔ بلیت Ticket-Granting Server (TGS) را معرفی می‌کنیم. سناریوی جدید که بازهم فرضی است بقرار زیر است:

یک‌بار برای هر اتصال کاربر به سیستم

(۱) C→AS: $ID_c \| ID_{tgs}$

(۲) S→C: $E(K_c, Ticket_{tgs})$

یک‌باربرای تقاضای هریک از انواع سرویس

(۳) C→TGS: $ID_c \| ID_v \| Ticket_{tgs}$

(۴) TGS→C: $Ticket_v$

یک‌بار برای هر اجلاس استفاده از سرویس

$$(۵)\quad C \rightarrow V: \qquad ID_c \,||\, Ticket_v$$

$$Ticket_{tgs} = E\,(K_{tgs}\,,\,[\,ID_c \,||\, AD_c \,||\, ID_{tgs} \,||\, TS_1 \,||\, Lifetime_1\,])$$

$$Ticket_v \;\; = E\,(K_v\,,\,[\,ID_c \,||\, AD_c \,||\, ID_v \,||\, TS_2 \,||\, Lifetime_2\,])$$

سرویس جدید، TGS، بلیت‌هائی را برای کاربرانی که اعتبار آنها بتوسط AS تأئید شده است صادر می‌کند. بنابراین کاربر ابتدا از AS تقاضای یک بلیت اعطاکنندهٔ بلیت ($Ticket_{tgs}$) می‌نماید. این بلیت بتوسط مدول کلاینت در ایستگاه کاری کاربر ذخیره می‌شود. هربار که کاربر نیازمند دست‌یابی به سرویس جدیدی است، کلاینت آن را به TGS ارائه داده و با استفاده از بلیت ذخیره شده هویت خود را به اثبات می‌رساند. آنگاه TGS یک بلیت برای آن سرویس خاص صادر می‌کند. کلاینت هر بلیت اعطاکنندهٔ سرویس را ذخیره کرده و از آن برای ارائهٔ اعتبار کاربر خود به سِرور، در هر بار تقاضا برای سرویس خاصی، استفاده می‌کند. اجازه دهید به جزئیات امر نگاهی بیندازیم:

۱- کلاینت به نمایندگی کاربر، درخواست یک بلیت اعطاکنندهٔ بلیت را کرده و برای این امرID خود و TGS ID را برای AS می‌فرستد که بیانگر درخواست استفاده از سرویس TGS است.

۲- AS با ارسال یک بلیت،که با کلید K_c رمزنگاری شده است پاسخ می‌دهد. این کلید از کلمهٔ عبور کاربر که قبلاً در AS ذخیره شده است تهیه می‌شود. وقتی این پاسخ وارد کلاینت می‌شود، کلاینت با ارسال پیامی از کاربر تقاضای کلمهٔ عبور کرده و سپس با استفاده از آن کلید را تولید کرده و برای رمزگشائی پیام ورودی تلاش می‌کند. اگر کلمهٔ عبور صحیح باشد، بلیت بطور موفقیت‌آمیزی بازگشائی می‌شود.

نظر باینکه قاعدتاً فقط کاربر اصلی بایستی کلمهٔ عبور را بداند، تنها کاربر اصلی میتواند بلیت را بازیابی کند. بنابراین ما از کلمهٔ عبور برای کسب امتیازات از Kerberos استفاده کرده بدون اینکه نیاز باشد تا کلمهٔ عبور را بصورت متن ساده و رمز نشده ارسال کنیم. بلیت، خود شامل ID و آدرس شبکهٔ کاربر و همچنین ID سِرور TGS است. این بخش نظیر سناریوی اول است و هدف این‌است که کاربر بتواند با استفاده از این بلیت، بلیت‌های اعطاکنندهٔ سرویس متعددی را درخواست کند. بنابراین بلیت اعطاکنندهٔ بلیت بایستی قابل استفادهٔ مکرر باشد. از سوی دیگر مایل نیستیم که یک دشمن بتواند بلیت را دزدیده و از آن استفاده کند. سناریوی زیر را در نظر بگیرید: یک دشمن بلیت را دزدیده و منتظر می‌ماند تا کاربر از ایستگاه کاری خود جدا شود. سپس این دشمن یا به ایستگاه کاری کاربر دسترسی فیزیکی یافته و یا ایستگاه کاری خود را با همان آدرس شبکهٔ ایستگاه کاری قربانی پیکربندی می‌کند. دشمن قادر خواهد بود تا مجدداً از بلیت استفاده کرده و TGS را فریب دهد. برای مقابله با این امر، بلیت شامل یک برچسب زمانی(timestamp) است که دارای تاریخ و لحظهٔ صدور و یک طول عمر که مشخص‌کننده محدودهٔ زمانی معتبر آن است، می‌باشد (مثلاً ۸ ساعت). بنابراین کلاینت حالا یک بلیت قابل استفاده مکرر داشته و لازم نیست تا برای هر سرویس جدید از کاربر تقاضای کلمهٔ عبور نماید. بالاخره توجه کنید که بلیت اعطاکنندهٔ بلیت با یک کلید رمز سرّی که تنها برای AS و TGS شناخته شده است رمزنگاری می‌شود. این امر تغییر بلیت را ناممکن می‌سازد. بلیت مجدداً با یک کلید که از کلمهٔ عبور کاربر مشتق شده است، رمزنگاری می‌شود. این موضوع این اطمینان را ایجاد میکند که بلیت تنها بتوسط کاربر معتبری که هویت صحیح خود را اظهار میکند می‌تواند استخراج شود.

حال که کلاینت یک بلیت اعطاکنندهٔ بلیت دارد، دسترسی به هر سِروری با استفاده از قدم‌های ۳ و ۴ امکان‌پذیر است.

۳- کلاینت به نمایندگی کاربر یک بلیت اعطاکنندهٔ سرویس را تقاضا می‌کند. برای این مقصود، کلاینت پیامی که شامل ID کاربر، ID سرویس مورد نیاز و بلیت اعطاکنندهٔ بلیت است را برای TGS می‌فرستد.

۴- TGS بلیت ورودی را با کلیدی(K_{tgs}) که تنها بین AS و TGS به اشتراک گذاشته شده است رمزگشائی کرده و موفقیت رمزگشائی را با کشف ID خود تأئید می‌نماید. همچنین کنترل می‌کند که طول عمر بلیت منقضی نشده باشد. سپس ID کاربر و آدرس شبکه را با اطلاعات ورودی تطبیق کرده تا اعتبار کاربر را بسنجد. اگر کاربر مجاز به دسترسی به V باشد، TGS یک بلیت برای دسترسی به سرویس تقاضاشده صادر می‌نماید.

بلیت اعطاکنندهٔ سرویس دارای همان ساختار بلیت اعطاکنندهٔ بلیت است. در واقع چون TGS یک سِرور است، طبیعتاً انتظار میرود که همان عناصری که برای معرفی یک کلاینت به TGS لازم‌اند برای معرفی کلاینت به سِرور کاربردها نیز مورد نیاز باشند. بازهم بلیت شامل برچسب زمانی و طول عمر است. اگر کاربر درخواست دسترسی به همان سرویس در زمانی دیگر را داشته باشد، کلاینت میتواند بسادگی از بلیت اعطاکنندهٔ سرویس قبلی استفاده کرده و مجدداً برای اخذ کلمهٔ عبور مزاحم کاربر نشود. توجه کنید که بلیت با کلید سری K_v رمزنگاری شده که این کلید فقط برای TGS و سِرور شناخته شده بوده و بنابراین دخل تصرف در آن ممکن نیست.

بالاخره با یک بلیت اعطاکنندهٔ سرویس، کلاینت می‌تواند از طریق قدم ۵ به سرویس مورد نظر دست یابد.

۵- کلاینت به نمایندگی کاربر تقاضای دسترسی به سرویسی را می‌نماید. برای این منظور کلاینت پیامی را به سرور منتقل می‌کند که شامل ID کاربر و بلیت اعطاکنندهٔ سرویس است. سِرور با استفاده از محتویات بلیت، اعتبار آن را می‌سنجد.

این سناریوی جدید دو نیاز ذکر شده یعنی فقط یکبار درخواست کلمهٔ عبور در هر مرتبه اتصال کاربر به شبکه، و همچنین محافظت از کلمهٔ عبور کاربر را برآورده می‌سازد.

دیالوگ اعتبارسنجی نسخهٔ ۴

اگرچه سناریوی ذکرشده در بالا در مقایسه با سناریوی اول، امنیت را بهبود می‌بخشد ولی هنوز دو مشکل باقی است. جوهر مشکل اول، طول عمری است که در بلیت اعطاکنندهٔ بلیت گنجانده شده است. اگر طول عمر خیلی کوتاه باشد (مثلاً چند دقیقه)، آنگاه به دفعات از کاربر تقاضای ارائهٔ کلمهٔ عبور خواهد شد. اگر طول عمر زیاد باشد(مثلاً ساعت‌ها)، آنگاه یک دشمن فرضی فرصت زیادتری برای بازخوانی خواهد داشت. یک دشمن ممکن است روی شبکه استراق‌سمع کرده و یک کپی از بلیت اعطاکنندهٔ بلیت را بدزدد و سپس آنقدر صبر کند تا کاربر قانونی از سیستم خارج شود. در این صورت دشمن میتواند آدرس شبکهٔ کاربر قانونی را جعل کرده و پیام مرحلهٔ (۳) را برای TGS ارسال کند. این امر به دشمن اجازه خواهد داد تا دسترسی نامحدودی به منابع و فایل‌های کاربر قانونی پیدا نماید.

بهمین ترتیب اگر دشمن یک بلیت اعطاکنندهٔ سرویس را دزدیده و از آن قبل از پایان مهلت استفاده کند، می‌تواند به سرویس‌های نظیر آن دست یابد.

بنابراین نیاز دیگری چهره می‌نماید. یک سرویس شبکه (TGS یا یک سرویس کاربردی) بایستی بتواند اثبات کند که شخصی که از بلیت استفاده می‌کند همان شخصی است که بلیت برای او صادر شده است.

مسألهٔ دوم این است که ممکن است نیاز باشد تا سِرورها نیز اعتبار خود را برای کاربران به اثبات برسانند. بدون اثبات چنین اعتباری، یک دشمن ممکن است پیکربندی را طوری مورد خرابکاری قرار دهد که پیام‌های بمقصد سِرور به محل دیگری بروند. در اینصورت سِرور قلابی بجای سِرور اصلی نشسته، اطلاعات کاربر را دریافت نموده و مانع از دادن سرویس صحیح به او می‌شود.

این مشکلات را بنوبت بررسی کرده و به جدول ۱-۴ که پروتکل Kerberos واقعی را نشان میدهد ارجاع می‌دهیم.

در وهلهٔ اول به مسألهٔ ربوده شدن بلیت اعطاکنندهٔ بلیت و نیاز به اثبات اینکه عرضه‌کنندهٔ بلیت همان کلاینتی است که بلیت برای او صادر شده است می‌پردازیم. تهدیدی که در اینجا وجود دارد این‌است که دشمن بلیت را دزدیده و قبل از انقضای مهلت از آن استفاده نماید. برای غلبه بر این مشکل فرض می‌کنیم AS را مجبور سازیم تا هم کلاینت و هم TGS را با نوعی اطلاعات سرّی به‌نحو امنی تجهیز نماید. آنگاه کلاینت می‌تواند هویت خود را با آشکارنمودن همان اطلاعات سرّی، بازهم به‌نحو امنی، به اثبات برساند. یک روش مؤثر برای انجام این امر استفاده از یک کلید رمزنگاری امن است که در Kerberos کلید اجلاس(Session key) خوانده می‌شود.

جدول ۱-۴الف روش توزیع کلید اجلاس را نشان می‌دهد. همانند قبل، کلاینت با ارسال یک پیام به AS درخواست دسترسی به TGS را می‌نماید. AS با یک پیام،که بتوسط یک کلید که از کلمهٔ عبور کاربر مشتق شده (K_C) رمزنگاری شده است و شامل بلیت است پاسخ می‌دهد. پیام رمزنگاری شده همچنین شامل یک کپی از کلید اجلاس $K_{c,tgs}$ است که در آن اندیس‌ها نشان می‌دهند که این کلید اجلاس C و TGS است. چون کلید اجلاس در درون پیامی است که با K_C رمزنگاری

جدول ۱-۴ خلاصه‌ای از مبادلهٔ پیام‌ها در Kerberos Version 4

(الف) مبادلهٔ سرویس اعتبارسنجی: برای کسب بلیت اعطاکنندهٔ بلیت	
(۱)	$\mathbf{C \rightarrow AS:}\quad ID_c \,\|\|\, ID_{tgs} \,\|\|\, TS_1$
(۲)	$\mathbf{AS \rightarrow C:}\quad E(K_c, [K_{c,tgs} \,\|\|\, ID_{tgs} \,\|\|\, TS_2 \,\|\|\, Lifetime_2 \,\|\|\, Ticket_{tgs}])$ $Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \,\|\|\, ID_c \,\|\|\, AD_c \,\|\|\, ID_{tgs} \,\|\|\, TS_2 \,\|\|\, Lifetime_2])$
(ب) مبادلهٔ سرویس اعطاء- بلیت: برای کسب بلیت اعطاکنندهٔ سرویس	
(۳)	$\mathbf{C \rightarrow TGS:}\quad ID_v \,\|\|\, Ticket_{tgs} \,\|\|\, Authenticator_c$
(۴)	$\mathbf{TGS \rightarrow C:}\quad E(K_{c,tgs}, [K_{c,v} \,\|\|\, ID_v \,\|\|\, TS_4 \,\|\|\, Ticket_v])$ $Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \,\|\|\, ID_c \,\|\|\, AD_c \,\|\|\, ID_{tgs} \,\|\|\, TS_2 \,\|\|\, Lifetime_2])$ $Ticket_v = E(K_v, [K_{c,v} \,\|\|\, ID_c \,\|\|\, AD_c \,\|\|\, ID_v \,\|\|\, TS_4 \,\|\|\, Lifetime_4])$ $Authenticator_c = E(K_{c,tgs}, [ID_c \,\|\|\, AD_c \,\|\|\, TS_3])$
(ج) مبادلهٔ اعتبارسنجی کلاینت/ سِرور: برای کسب سرویس	
(۵)	$\mathbf{C \rightarrow V:}\quad Ticket_v \,\|\|\, Authenticator_c$
(۶)	$\mathbf{V \rightarrow C:}\quad E(K_{c,v}, [TS_5 + 1])$ (برای اعتبارسنجی متقابل) $Ticket_v = E(K_v, [K_{c,v} \,\|\|\, ID_c \,\|\|\, AD_c \,\|\|\, ID_v \,\|\|\, TS_4 \,\|\|\, Lifetime_4])$ $Authenticator_c = E(K_{c,v}, [ID_c \,\|\|\, AD_c \,\|\|\, TS_5])$

شده است، تنها کلاینت کاربر می‌تواند آن را بخواند. همین کلید اجلاس در بلیت نیز قرار دارد که تنها می‌تواند بتوسط TGS خوانده شود. بنابراین کلید اجلاس بصورت امن هم به C و هم به TGS تحویل شده است.

توجه کنید که چند بخش اطلاعات اضافی به اولین فاز دیالوگ اضافه شده است. پیام (۱) شامل یک برچسب زمانی بوده تا AS بداند که پیام دارای محدودیت زمانی است. پیام (۲) شامل چندین عنصر بلیت به فرمی است که قابل دسترس برای C باشد. باین ترتیب C قادر است تأئید کند که این بلیت برای TGS بوده و از زمان انقضای آن آگاهی می‌یابد.

با مسلح شدن به بلیت و کلید اجلاس، C آماده است که به TGS نزدیک شود. همانند قبل، C پیامی را که شامل بلیت باضافهٔ ID سرویس درخواستی [پیام (۳) در جدول ۱-۴ب] است برای TGS می‌فرستد. علاوه برآن، C یک اعتبارسنج که شامل ID و آدرس کاربرC و یک برچسب زمانی است را ارسال می‌کند. برخلاف بلیت، که دوباره قابل استفاده است، اعتبارسنج فقط یک‌بار قابل استفاده بوده و طول عمر کوتاهی دارد. TGS میتواند با کلیدی که با AS به اشتراک دارد، بلیت را رمزگشائی نماید. این بلیت نشان میدهد که کاربر C با کلید اجلاس $K_{c,tgs}$ تجهیزشده است. در واقع بلیت میگوید، «هرکه از $K_{c,tgs}$ استفاده می‌کند بایستی C باشد». TGS از کلید اجلاس برای رمزگشائی اعتبارسنج استفاده می‌کند. بدنبال آن TGS می‌تواند اسم و آدرس استخراج شده از اعتبارسنج را با همین موارد در بلیت و آدرس شبکه‌ای که پیام از آن وارد شده است مقایسه نماید. اگر همهٔ اینها با هم تطبیق داشته باشند، آنگاه TGS مطمئن می‌شود که ارسال‌کنندهٔ این بلیت واقعاً همان صاحب بلیت است. در واقع تأئیدکننده می‌گوید «در زمان TS_3، من بدین‌وسیله از $K_{c,tgs}$ استفاده می‌کنم.» توجه شود که بلیت هویت کسی را اثبات نمی‌کند بلکه روش امنی برای توزیع کلیدهاست. این اعتبارسنج است که هویت کلاینت را به اثبات می‌رساند. چون از اعتبارسنج تنها یک‌بار می‌توان استفاده کرد و دارای طول عمر کوتاهی نیز هست، امکان اینکه یک دشمن هم بلیت و هم اعتبارسنج را دزدیده و در آینده از آن استفاده کند از بین می‌رود.

پاسخ TGS در پیام (۴) از فرم پیام (۲) تبعیت می‌کند. پیام بتوسط کلید اجلاس که مشترک بین TGS و C است رمزنگاری شده و شامل یک کلید اجلاس که بین C و سِرورV مشترک است، ID سِرورV، و برچسب زمانی بلیت می‌باشد. خود بلیت شامل همان کلید اجلاس است.

C اکنون دارای یک بلیت اعطاکنندهٔ بلیت برایV است که می‌تواند بارها مورد استفاده قرار گیرد. وقتی C این بلیت را، همانند آنچه در پیام (۵) نشان داده شده است عرضه می‌دارد، یک اعتبارسنج را نیز با آن می‌فرستد. سِرور میتواند بلیت را رمزگشائی کرده، کلید اجلاس را استخراج نموده و اعتبارسنج را نیز از رمز درآورد.

اگر اعتبارسنجی متقابل مورد نیاز باشد، سِرور میتواند همانند پیام (۶) در جدول ۱-۴ پاسخ دهد. سِرور اندازهٔ برچسب زمانی در اعتبارسنج را به اندازهٔ یک واحد اضافه کرده و پس از رمزنگاری با کلید اجلاس آن را برمی‌گرداند. C می‌تواند این پیام را از رمز درآورده و برچسب زمانی افزایش یافته را استخراج نماید. نظر به اینکه پیام بتوسط کلید اجلاس رمز شده بود، C مطمئن است که این تنها می‌توانسته بتوسط V خلق شود. محتویات پیام به C اطمینان می‌دهدکه این بازخوانی یک پاسخ قدیمی نیست.

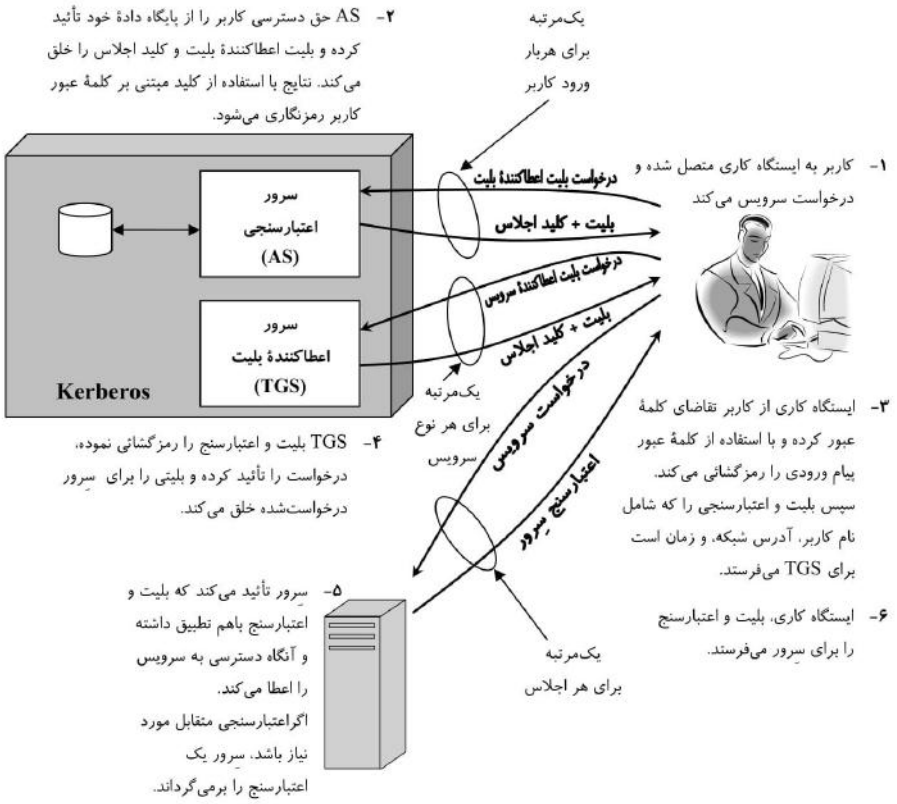
بالاخره در پایان این مرحله، کلاینت و سِرور یک کلید سرّی را به اشتراک می‌گذارند. از این کلید می‌توان برای رمزنگاری پیامهای آتی بین این دو، و یا برای مبادلهٔ یک کلید اجلاس تصادفی جدید برای این مقصود استفاده کرد.

جدول ۲-۴ وجود هریک از عناصر پروتکل Kerberos را توجیه کرده و شکل ۱-۴ یک نمای ساده از عملیات را نشان میدهد.

جدول ۲-۴ دلایل منطقی وجود مولفه‌های Kerberos Version 4

(الف) مبادلهٔ سرویس اعتبارسنجی	
پیام (۱)	کلاینت، بلیت اعطاکنندهٔ بلیت درخواست می‌کند
ID_C	به AS می‌گوید که هویت کاربری که از این کلاینت تماس می‌گیرد چیست
ID_{tgs}	به AS می‌گوید که کاربر تقاضای دست‌یابی به TGS را دارد
TS_1	AS را قادر می‌سازد تا تأئید کند که ساعت کلاینت با ساعت AS هم‌آهنگ است
پیام (۲)	AS بلیت اعطاکنندهٔ بلیت را تهیه کرده و برای کلاینت برمی‌گرداند
K_c	رمزنگاری مبتنی بر کلمهٔ عبور کاربر است، AS و کلاینت را قادر می‌سازد تا کلمهٔ عبور را تأئید نمایند و همچنین محتویات پیام (۲) را محافظت می‌نماید
$K_{c,tgs}$	کپی کلید اجلاس که به توسط کلاینت قابل دسترسی است، به توسط AS خلق شده است تا بدون این‌که نیاز به یک کلید دائمی باشد مبادلهٔ امن بین کلاینت و TGS را امکان‌پذیر نماید
ID_{tgs}	تائید می‌کند که این بلیت برای TGS است
TS_2	کلاینت را از زمان صدور این بلیت آگاه می‌سازد
$Lifetime_2$	کلاینت را از طول عمر این بلیت آگاه می‌سازد
$Ticket_{tgs}$	بلیتی است که از سوی کلاینت برای دسترسی به TGS مورد استفاده قرار می‌گیرد
(ب) مبادلهٔ سرویس اعطاء بلیت	
پیام (۳)	کلاینت، بلیت اعطاکنندهٔ سرویس درخواست می‌کند
ID_V	به TGS می‌گوید که کاربر تقاضای دسترسی به سِرور V را دارد
$Ticket_{tgs}$	به TGS اطمینان می‌دهد که این کاربر به توسط AS اعتبارسنجی شده است
$Authenticator_C$	بتوسط کلاینت تولید شده تا بلیت را معتبر نماید
پیام (۴)	TGS بلیت اعطاکنندهٔ سرویس را تهیه کرده و برای کلاینت برمی‌گرداند
$K_{c,tgs}$	کلیدی که فقط بین C و TGS مشترک است و محتویات پیام (۴) را محافظت می‌کند
$K_{c,v}$	کپی کلید اجلاس قابل دسترسی بتوسط کلاینت، بتوسط TGS خلق می‌گردد تا بدون نیاز به یک کلید دائم مشترک بین کلاینت و سِرور، مبادلهٔ امن بین آنها را امکان‌پذیر نماید
ID_V	تأئید می‌کند که این بلیت برای سِرور V است
TS_4	کلاینت را از زمان صدور این بلیت آگاه می‌سازد
$Ticket_V$	بلیتی است که از سوی کلاینت برای دسترسی به سِرور V مورد استفاده قرار می‌گیرد
$Ticket_{tgs}$	قابل استفادهٔ مکرر است تا کاربر مجبور نباشد تا هربار کلمهٔعبور خود را وارد کند
K_{tgs}	بلیت با کلیدی که فقط برای AS و TGS شناخته شده است رمزنگاری می‌شود تا از تحریف آن جلوگیری شود
$K_{c,tgs}$	کلید اجلاس قابل دسترسی بتوسط TGS، برای رمزگشائی اعتبارسنج و بنابراین اعتبارسنجی بلیت بکار می‌رود
ID_C	نمایش‌گر صاحب واقعی این بلیت است
AD_C	از استفاده از بلیت بتوسط یک ایستگاه کاری، بجز ایستگاهی که بدواً تقاضای بلیت کرده بود جلوگیری می‌کند

ID_{tgs}	به سرور اطمینان می‌دهد که بلیت را بطور صحیح رمزگشائی کرده است
TS_2	TGS را از زمان صدور این بلیت آگاه می‌سازد
$Lifetime_2$	از بازخوانی پس از انقضای بلیت جلوگیری می‌کند
$Authenticator_c$	به TGS اطمینان می‌دهد که عرضه‌کنندهٔ بلیت همان کلاینتی است که بلیت برای او صادر شده است. دارای طول عمر کوتاهی است تا از بازخوانی جلوگیری شود
$K_{c,tgs}$	اعتبارسنج با کلیدی که تنها برای کلاینت و TGS شناخته شده است رمزنگاری می‌شود تا از تحریف جلوگیری شود
ID_C	بایستی با ID بلیت تطبیق داشته باشد تا بلیت معتبر شناخته شود
AD_C	بایستی با آدرس بلیت تطبیق داشته باشد تا بلیت معتبر شناخته شود
TS_3	TGS را از زمان تولید این اعتبارسنج آگاه می‌سازد
	(ج) مبادلهٔ اعتبارسنجی کلاینت/ سرور
پیام (۵)	کلاینت درخواست سرویس می‌کند
$Ticket_v$	به سرور اطمینان می‌دهد که این کاربر بتوسط AS اعتبارسنجی شده است
$Authenticator_c$	بتوسط کلاینت خلق شده تا بلیت را معتبر سازد
پیام (۶)	اعتبارسنجی اختیاری سرور برای کلاینت
$K_{c,v}$	به C اطمینان می‌دهد که این پیام از سوی V است
$TS_5 + 1$	به C اطمینان می‌دهد که این بازخوانی یک پاسخ قدیمی نیست
$Ticket_v$	قابل استفادهٔ مجدد بوده تا لازم نباشد که کاربر برای هربار دست‌یابی به یک سرور معین تقاضای یک بلیت جدید کند
K_v	بلیت بتوسط کلیدی که فقط برای TGS و سرور شناخته شده است رمزنگاری شده تا از تحریف جلوگیری شود
$K_{c,v}$	کپی کلید اجلاس قابل دست‌یابی بتوسط کلاینت. برای رمزگشائی اعتبارسنج و بنابراین تأئید اعتبار بلیت استفاده می‌شود
ID_C	صاحب اصلی این بلیت را نشان می‌دهد
AD_C	از استفاده از بلیت یک ایستگاه کاری بجز آن که بدواً تقاضای بلیت کرده بود جلوگیری می‌کند
ID_V	به سرور اطمینان می‌دهد که که بلیت را بطرز صحیح رمزگشائی کرده است
TS_4	سرور را از زمان صدور این بلیت آگاه می‌سازد
$Lifetime_4$	از بازخوانی پس از انقضای بلیت جلوگیری می‌کند
$Authenticator_c$	به سرور اطمینان می‌دهد که عرضه‌کنندهٔ بلیت همان کلاینتی است که بلیت برای او صادر شده است. دارای طول عمر کوتاهی است تا از بازخوانی جلوگیری شود
$K_{c,v}$	اعتبارسنج با کلیدی که تنها برای کلاینت و سرور شناخته شده است رمزنگاری می‌شود تا از تحریف جلوگیری شود
ID_C	بایستی با ID بلیت تطبیق داشته باشد تا بلیت معتبر شناخته شود
AD_C	بایستی با آدرس بلیت تطبیق داشته باشد تا بلیت معتبر شناخته شود
TS_5	سرور را از زمان تولید این اعتبارسنج آگاه می‌سازد



شکل ۱-۴ مروری بر Kerberos

قلمرو Kerberos و Kerberosهای متعدد

محیط خدماتی کامل یک سرور Kerberos که شامل یک سرور Kerberos، تعدادی کلاینت و تعدادی سرورهای کاربردی است به موارد زیر نیاز دارد:

۱- سرور Kerberos بایستی ID کاربران(UID) و کلمهٔ‌عبور درهم‌سازی‌شدهٔ همهٔ کاربران حوزه را در پایگاه دادهٔ خود داشته باشد. تمام کاربران بایستی در نزد Kerberos ثبت‌نام شده باشند.

۲- سرور Kerberos بایستی با هر سرور دیگر یک کلید سرّی مشترک داشته باشد. تمام سرورها بایستی در نزد سرور Kerberos ثبت‌نام شده باشند.

چنین محیطی را یک **قلمرو(Kerberos realm)** خوانند. مفهوم یک *قلمرو* را می‌توان چنین تشریح کرد: یک قلمرو Kerberos یک مجموعه از گره‌های مدیریت شده است که همگی پایگاه دادهٔ Kerberos را در اشتراک دارند. پایگاه دادهٔ Kerberos در سیستم کامپیوتری اصلی Kerberos قرار دارد که نوعاً بایستی در یک اطاق با امنیت فیزیکی خوب قرار داشته باشد. یک نسخهٔ فقط قابل خواندن از این پایگاه داده نیز قاعدتاً می‌تواند روی کامپیوترهای دیگر سیستم نصب شود. دست‌یابی و یا تغییر محتوای پایگاه دادهٔ Kerberos نیاز به کلمهٔ عبور اصلی Kerberos دارد. مفهوم دیگری که با این مسأله مرتبط است، وجود یک **رئیس (Kerberos principal)** است که سرویس و یا کاربری است که برای سیستم Kerberos شناخته شده است. هر رئیس Kerberos با نام ریاست خود شناخته می‌شود. نام‌های ریاست دارای سه جزء نام یک سرویس و یا یک کاربر، نام یک مورد، و نام یک قلمرو می‌باشند.

شبکه‌های متشکل از کلاینت‌ها و سِرورها در سازمان‌های مدیریتی مختلف، معمولاً قلمروهای متفاوتی را تشکیل می‌دهند. این که کاربران و سِرورهای یک حوزهٔ مدیریتی، در سِرور Kerberos حوزهٔ مدیریتی دیگری ثبت‌نام شده باشند نه عملی است و نه معمولاً با خط‌مشی‌های اداری منطبق است. از سوی دیگر، ممکن است کاربران یک قلمرو نیاز به دست‌یابی به سِرورهای قلمرو دیگری داشته و یا اینکه بعضی سِرورهای یک قلمرو تمایل داشته باشند تا به کاربران معتبر قلمرو دیگر ارائهٔ سرویس نمایند.

Kerberos مکانیسمی را برای حمایت از این اعتبارسنجی بین قلمروها ایجاد نموده است. برای اینکه دو قلمرو از اعتبارسنجی بین قلمروها حمایت کنند، نیاز سومی به پروتکل اضافه می‌شود:

۳- سِرور Kerberos هر قلمرو بایستی با سِرور Kerberos قلمرو دیگر یک کلید سرّی را به اشتراک بگذارد. دو سِرور Kerberos بایستی در یکدیگر ثبت‌نام شده باشند.

این روش نیازمند این‌است که سِرور Kerberos یک قلمرو به سِرور Kerberos قلمرو دیگر، نسبت به سنجش اعتبار کاربران خود اعتماد داشته باشد. علاوه بر آن سِرورهای قلمرو دوم نیز بایستی تمایل به اعتماد به سِرور Kerberos قلمرو اول داشته باشند.

با استقرار این قواعد در جای خود، مکانیسم عمل را می‌توان بصورت زیر تشریح کرد (شکل ۲-۴): کاربری که مایل به اخذ سرویس از سِروری در قلمرو دیگر است، نیاز به یک بلیت برای آن سِرور دارد. کلاینت کاربر، همان روش‌های ذکرشده برای دسترسی به TGS محلی را دنبال کرده و سپس یک بلیت اعطاکنندهٔ بلیت برای TGS دور (TGS قلمرو دیگر) تقاضا می‌کند. بدنبال آن کلاینت می‌تواند از TGS دور درخواست یک بلیت اعطاکنندهٔ سرویس جهت استفاده از سِرور مورد نیاز در قلمرو او را بنماید.

جزئیات مبادلاتی که در شکل ۲-۴ نشان داده شده است بشرح زیر است (با جدول ۱-۴ مقایسه کنید):

(۱) C→AS: $ID_c \| TS_1 \| ID_{tgs}$

(۲) S→C: $E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$

(۳) C→TGS: $ID_{tgsrem} \| Ticket_{tgs} \| Authenticator_c$

(۴) TGS→ C: $E(K_{c,tgs}, [K_{c,tgsrem} \| ID_{tgsrem} \| TS_4 \| Ticket_{tgsrem}])$

(۵) $C \rightarrow TGS_{rem}$: $ID_{vrem} \| Ticket_{tgsrem} \| Authenticator_c$

(۶) $TGS_{rem} \rightarrow C$: $E(K_{c,tgsrem}, [K_{c,vrem} \| ID_{vrem} \| TS_6 \| Ticket_{vrem}])$

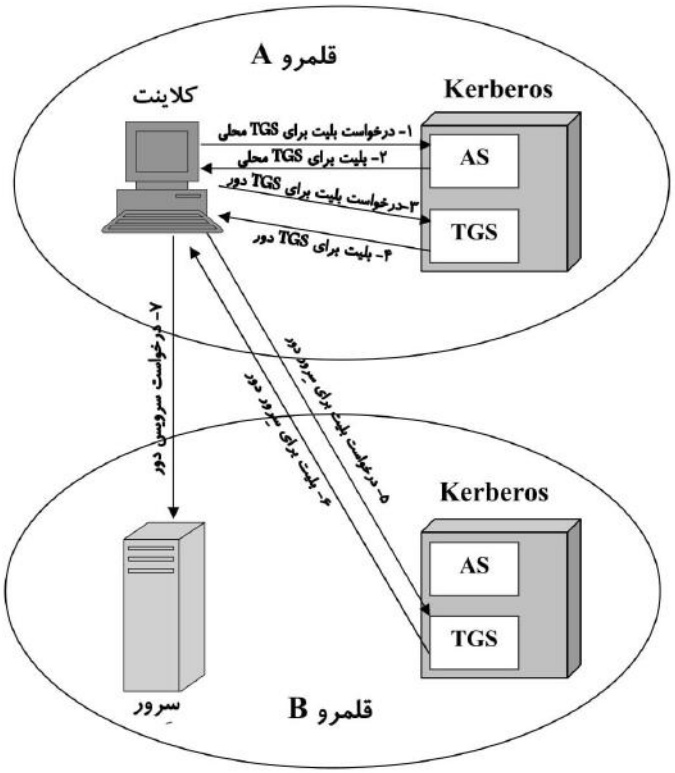
(۷) $C \rightarrow V_{rem}$: $Ticket_{vrem} \| Authenticator_c$

بلیت ارائه شده به سِرور دور (V_{rem}) نشان‌دهندهٔ قلمروی است که درآن ابتداءً کاربر اعتبارسنجی شده بود. سِرور در پاسخ دادن به این درخواست مخیّر است.

یکی از مشکلاتی که در روش بالا خودنمائی میکند این است که در صورت افزایش زیاد قلمروها، این روش به‌نحو مناسبی مقیاس‌پذیر نیست. اگر N قلمرو وجود داشته باشد، آنگاه بایستی $N(N\text{-}1)/2$ مبادلهٔ امن کلیدها صورت پذیرد تا قلمرو هرKerberos بتواند با قلمروهای بقیهٔ Kerberosها تعامل نماید.

نسخهٔ پنجم Kerberos

نسخهٔ پنجم Kerberos در RFC 1510 تعریف شده و مزیت‌هائی نسبت به نسخهٔ چهارم آن دارد[KOHL94]. برای شروع، نگاهی به تفاوت‌های نسخهٔ پنجم نسبت به نسخهٔ چهارم نموده و سپس پروتکل نسخهٔ ۵ را بررسی می‌کنیم.



شکل ۲-۴ درخواست سرویس از قلمرو دیگر

تفاوت‌های بین نسخهٔ ۴ و نسخهٔ ۵

نسخهٔ پنجم برای رفع محدودیت‌های نسخهٔ چهارم در دو زمینه طراحی شده است: نواقص محیطی و کمبودهای تکنیکی. اجازه دهید تا بطور مختصر بهبودهای ایجادشده در هر زمینه را خلاصه کنیم.

نسخهٔ چهارم Kerberos برای استفاده در محیط پروژهٔ Athena طراحی شده بود و بنابراین نیاز در زمینهٔ اهداف کلی را نادیده گرفته است. این امر باعث ایجاد **نقائص محیطی** زیر شده است:

۱- **وابستگی به سیستم رمزنگاری**: نسخهٔ ۴ نیاز به استفاده از DES دارد. محدودیت‌های صادراتی DES و همچنین تردید در مورد توانائی‌های آن، از موارد نگران‌کننده است. در نسخهٔ ۵، متن رمزشده با یک شناسهٔ نوع رمزنگاری مجهزشده و بنابراین می‌توان از هر نوع تکنیک رمزنگاری استفاده کرد. کلیدهای رمزنگاری دارای یک دنبالهٔ مشخص‌کنندهٔ نوع و اندازه بوده و این امر اجازه میدهد تا از یک کلید در الگوریتم‌های مختلف استفاده کرده و همچنین تغییرات متفاوتی را روی یک الگوریتم داده شده انجام داد.

۲- **وابستگی به پروتکل اینترنت**: نسخهٔ ۴ نیاز به استفاده از آدرس‌های پروتکل اینترنت(IP) دارد. سایر انواع آدرس، مثل آدرس شبکهٔ ISO، را نمی‌توان بکار گرفت. در نسخهٔ ۵، آدرس‌های شبکه با دنبالهٔ نوع و طول مجهز بوده و اجازه می‌دهد تا از هر نوع آدرسی استفاده کرد.

۳- **نظم بایت‌های پیام**: در نسخهٔ ۴، ارسال‌کنندهٔ یک پیام نظم بایت‌ها را از جانب خود انتخاب کرده و با الصاق یک دنباله به پیام مشخص می‌سازد که آیا بایت دارای کمترین اهمیت در پائین‌ترین آدرس قرار دارد و یا بایت دارای بیشترین اهمیت پائین‌ترین آدرس را اشغال کرده است. این روش عملی است ولی از قراردادهای جاافتاده تبعیت نمی‌کند. در نسخهٔ ۵ تمام ساختارهای پیام با استفاده از Abstract Syntax Notation One (ASN.1) و Basic Encoding Rule(BER) تعریف شده و بنابراین نظم بایت‌ها بطور غیرقابل ابهامی مشخص می‌گردند.

۴- **طول عمر بلیت**: اندازه‌های طول عمر در نسخهٔ ۴ بصورت یک کمیت ۸- بیتی در واحدهای ۵ دقیقه‌ای کُد شده‌اند. بنابراین ماکزیمم طول عمری که می‌تواند تعریف شود برابر ۱,۲۸۰= ۵ ×$۲^{۸}$ دقیقه و یا چیزی بالاتر از ۲۱ ساعت است. این مقدار برای برخی کاربردها ممکن است کافی نباشد (مثل یک شبیه‌سازی طولانی که در طول اجرا نیاز به پشتیبانی مستمر Kerberos دارد). در نسخهٔ ۵، بلیت‌ها دارای یک زمان شروع و یک زمان خاتمه صریح بوده و بنابراین یک بلیت می‌تواند هر طول عمری را داشته باشد.

۵- **جلوراندن اعتبارسنجی**: نسخهٔ ۴ اجازه نمی‌دهد که اعتبار صادر شده برای یک کلاینت به میزبان دیگری اهدا شده و کلاینت دیگری از آن استفاده نماید. این قابلیت به یک کلاینت اجازه خواهد داد تا به یک سِرور دست یافته و از آن سِرور بخواهد که از طرف آن کلاینت به سِرور دیگری دست یابد. برای مثال، یک کلاینت تقاضائی را برای یک سِرور چاپگر می‌فرستد که با استفاده از اعتبار کلاینت به فایل کلاینت در یک سِرور فایل دسترسی یابد. نسخهٔ ۵ این قابلیت را فراهم آورده است.

۶- **اعتبارسنجی بین قلمروها**: در نسخهٔ ۴، عملیات بین N قلمرو نیاز به حدود N^2 ارتباط Kerberos به Kerberos داشته که قبلاً در مورد آن بحث شد. نسخهٔ ۵ برابر آنچه بزودی تشریح خواهد شد نیاز به روابط کمتری دارد.

جدا از این محدودیت‌های محیطی، در خود پروتکل نسخهٔ ۴ یک سری **نواقص تکنیکی** وجود دارد. بیشتر این نواقص در[BELL90] ذکر شده و نسخهٔ ۵ برای رفع آنها تلاش کرده است. نواقص به قرار زیراند:

۱- **رمزنگاری دوبل**: در جدول۱-۴ توجه کنید [پیام‌های(۲)و(۴)] که بلیت‌های صادرشده برای کلاینت‌ها دوبار رمزنگاری می‌شوند که بار اول با کلید سرّی سِرور هدف و بار دوم با کلید سرّی آشنا برای کلاینت این‌کار صورت می‌پذیرد. رمزنگاری بار دوم مورد نیاز نبوده و از نظر محاسباتی وقت‌گیر است.

۲- **رمزنگاریPCBC**: رمـزنگاری در نسخهٔ ۴ از یک مُـود غیراستانـــدارد DES به نام (PCBC) propagating cipher block chaining استفاده می‌کند. نشان داده شده است که این مُود نسبت به یک حمله که شامل تعویض بلوک‌های رمزشده است، آسیب‌پذیراست[KOHL89]. PCBC قرار بود تا بعنوان بخشی از عملیات رمزنگاری، صحت داده‌ها را نیز کنترل نماید. نسخهٔ ۵ یک مکانیسم کنترل صحت صریح ایجاد کرده که اجازه میدهد تا از مُود CBC استاندارد برای رمزنگاری استفاده شود. علی‌الخصوص، یک جمع کنترلی یا کُد hash قبل از رمزنگاری با استفاده از CBC به پیام وصل می‌گردد.

۳- **کلیدهای اجلاس**: هر بلیت شامل یک کلید اجلاس است که از طرف کاربر برای رمزکردن اعتبارسنج ارسال‌شدهٔ متناظر با آن بلیت بکار می‌رود. علاوه برآن، کلید اجلاس می‌تواند متعاقباً بتوسط کلاینت و سِرور برای محافظت پیام‌هائی که در خلال اجلاس ردوبدل می‌شوند بکار رود. ولی چون یک بلیت ممکن است مکرراً برای دست‌یابی به سرویس یک سِرور خاص مورد استفاده قرار گیرد، این خطر وجود دارد که یک دشمن پیام‌های مربوط به یک اجلاس کهنه را برای کلاینت یا سِرور بازخوانی کند. در نسخهٔ ۵ این امکان وجود دارد که یک کلاینت با سِرور در مورد یک کلید زیراجلاس به این توافق برسند که از این کلید فقط برای یک بار اتصال استفاده شود. دسترسی جدید از سوی کلاینت، نیاز به استفاده از یک کلید زیراجلاس جدید دارد.

۴- **حملات کلمهٔ عبور**: هر دو نسخه در مقابل حملهٔ کلمهٔ عبور آسیب‌پذیرند. پیام AS به کلاینت شامل مطالبی است که با یک کلید که مشتق از کلمهٔ عبور کلاینت است رمزنگاری شده است. یک دشمن ممکن است این پیام را دزدیده و با بکارگیری کلمات عبور مختلف، آن را رمزگشائی نماید. اگر نتیجهٔ یکی از این رمزگشائی‌ها موفقیت‌آمیز باشد، آنگاه دشمن کلمهٔعبور کلاینت را کشف کرده و ممکن است بعداً آن را برای کسب امتیازات اعتبارسنجی از Kerberos بکار برد. این همان نوع حملهٔ کلمهٔ عبوری است که در فصل ۹ در مورد آن بحث شده و همان پاتک‌های ذکرشده در مورد آن قابل اجراست. نسخهٔ ۵، مکانیسم جدیدی بنام پیش‌اعتبارسنجی بکاربرده که حملات کلمهٔعبور را دشوارتر نموده ولی کاملاً از آنها جلوگیری نمی‌نماید.

دیالوگ اعتبارسنجی نسخهٔ ۵

جدول ۳-۴ دیالوگ اصلی نسخهٔ ۵ را خلاصه کرده است. این دیالوگ می‌تواند به بهترین نحو با مقایسه با نسخهٔ ۴ تشریح گردد (جدول ۱-۴).

در ابتدا **مبادلهٔ سرویس اعتبارسنجی** را درنظر بگیرید. پیام(۱) درخواست کلاینت برای یک بلیت اعطاکنندهٔ بلیت است. همانند قبل این درخواست شامل ID کاربر و TGS است. اقلام جدید زیر به آن اضافه شده‌اند:

جدول ۳-۴ خلاصهٔ مبادلهٔ پیام‌ها در Kerberos Version 5

(الف) مبادلهٔ سرویس اعتبارسنجی: برای کسب بلیت اعطاکنندهٔ بلیت	
(۱)	$\mathbf{C \rightarrow AS:}\quad \text{Options} \| ID_C \| Realm_c \| ID_{tgs} \| Times \| Nonce_1$
(۲)	$\mathbf{AS \rightarrow C:}\quad Realm_c \| ID_C \| Ticket_{tgs} \| E(K_c, [K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}])$ $Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$
(ب) مبادلهٔ سرویس اعطاء بلیت: برای کسب بلیت اعطاکنندهٔ سرویس	
(۳)	$\mathbf{C \rightarrow TGS:}\quad \text{Options} \| ID_v \| Times \| Nonce_2 \| Ticket_{tgs} \| Authenticator_c$
(۴)	$\mathbf{TGS \rightarrow C:}\quad Realm_c \| ID_C \| Ticket_v \| E(K_{c,tgs}, [K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_v])$ $Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$ $Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times])$ $Authenticator_c = E(K_{c,tgs}, [ID_C \| Realm_c \| TS_1])$
(ج) مبادلهٔ اعتبارسنجی کلاینت/ سرور: برای کسب سرویس	
(۵)	$\mathbf{C \rightarrow V:}\quad \text{Options} \| Ticket_v \| Authenticator_c$
(۶)	$\mathbf{V \rightarrow C:}\quad E(K_{c,v}, [TS_2 \| Subkey \| Seq\#])$ $Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times])$ $Authenticator_c = E(K_{c,v}, [ID_C \| Realm_c \| TS_2 \| Subkey \| Seq\#])$

- **قلمرو(Realm)**: قلمرو کاربر را نشان می‌دهد.
- **موارد اختیاری(Options)**: افراشتن پرچم‌های معینی در بلیت بازگشتی را درخواست می‌نماید.
- **زمان‌ها(Times)**: تنظیم زمان‌های زیر در بلیت، با تقاضای کلاینت، را ممکن می‌سازد:
  - از: زمان آغاز برای بلیت تقاضاشده
  - تا: زمان انقضاء برای بلیت تقاضاشده
  - تمدید: زمان تمدید برای بلیت تقاضاشده
- **حال فعلی(nonce)**: یک مقدار تصادفی که بایستی در پیام(۲) تکرار گردد تا مطمئن شویم که پاسخ تازه بوده و فرم قدیمی بازخوانی شده بتوسط دشمن نمی‌باشد.

پیام (۲) یک بلیت اعطاکنندهٔ بلیت را برگردانده، اطلاعات مربوط به کلاینت را بیان نموده و یک بلوک رمزنگاری‌شده با استفاده از کلید رمزی که از کلمهٔ عبور کاربر مشتق شده است را تهیه می‌کند. این بلوک شامل کلید اجلاسی بوده که بایستی بین کلاینت و TGS مورد استفاده قرار گیرد. زمان‌هائی که در پیام (۱) مشخص شده، nonce از پیام (۱) و اطلاعات شناسائی TGS است. خود بلیت شامل کلید اجلاس، اطلاعات شناسائی کلاینت، مقادیر زمانی تقاضاشده و پرچم‌هائی است که وضعیت این بلیت و اختیارات تقاضاشده را نشان می‌دهند. این پرچم‌ها قابلیت‌های جدید قابل ملاحظه‌ای را به نسخهٔ ۵ اضافه می‌کنند. فعلاً بحث در مورد این پرچم‌ها را به تعویق انداخته و روی ساختار کلی پروتکل نسخهٔ ۵ تمرکز می‌کنیم.

حال اجازه دهید تا **مبادلهٔ سرویس اعطاکردن بلیت** در نسخه‌های ۴ و ۵ را با هم مقایسه کنیم. همانطور که دیده می‌شود، پیام (۳) برای هر دو نسخه شامل یک اعتبارسنج، یک بلیت و نام سرویس درخواستی می‌باشد. علاوه بر آن نسخهٔ ۵ شامل زمان‌های تقاضاشده و موارد اختیاری بلیت و یک nonce است که همهٔ آنها شبیه موارد پیام (۱) اند. اعتبارسنج خود ضرورتاً همانی است که در نسخهٔ (۴) بکار رفته است.

پیام (۴) دارای همان ساختار پیام (۲) بوده و یک بلیت را بهمراه اطلاعاتی که مورد نیاز کلاینت است برمی‌گرداند. اطلاعات با همان کلید اجلاسی که حالا بین کلاینت و TGS مشترک است، رمزنگاری می‌گردد.

بالاخره، برای **مبادلهٔ اعتبارسنجی کلاینت/سِرور**، چند مشخصهٔ جدید در نسخهٔ ۵ گنجانده شده است. در پیام (۵)، کلاینت بعنوان یک درخواست اختیاری، ممکن است تقاضای اعتبارسنجی متقابل نماید. برای این منظور، اعتبارسنج دارای چند میدان جدید است:

- **زیرکلید (subkey)**:کلاینت حق دارد تا یک کلید رمزنگاری برای حفاظت این اجلاس کاربردی خاص درخواست کند. اگر این میدان حذف شود، کلید اجلاس بلیت($K_{C,V}$) مورد استفاده قرار خواهد گرفت.
- **شمارهٔ ردیف (Sequence number)**: یک میدان اختیاری تعیین‌کنندهٔ شمارهٔ ردیف آغازین برای استفادهٔ سِرور در شماره‌گذاری پیام‌هایی است که در این اجلاس برای کلاینت ارسال می‌شود. پیام‌ها از این‌جهت ممکن است شماره‌گذاری شوند تا حمله‌ای از نوع بازخوانی را کشف نمایند.

اگر اعتبارسنجی متقابل مورد نیاز باشد، سِرور با پیام (۶) پاسخ می‌دهد. این پیام شامل برچسب زمانی اعتبارسنج است. توجه کنید که در نسخهٔ ۴، برچسب زمانی یک واحد افزایش می‌یافت. این امر در نسخهٔ ۵ مورد نیاز نبوده زیرا فرمت پیام بنحوی است که برای یک دشمن امکان‌پذیر نخواهد بود که پیام (۶) را بدون اطلاع از کلیدهای رمزنگاری مناسب، خلق نماید. میدان زیرکلید، اگر وجود داشته باشد، میدان زیرکلید در پیام ۵ اگر وجود داشته باشد را ملغی می‌سازد. میدان اختیاری شماره ردیف، اولین شماره‌ای که بایستی بتوسط کلاینت مورد استفاده قرار گیرد را تعیین می‌کند.

پرچم‌های بلیت (Ticket Flags)

میدان‌های پرچم قرارداده شده در نسخهٔ ۵، قابلیت‌های عملکرد را نسبت به آنچه که در نسخهٔ ۴ است توسعه بخشیده است. جدول ۴-۴ پرچم‌هائی که ممکن است در یک بلیت وجود داشته باشند را نشان داده است.

پرچم INITIAL نشان می‌دهد که این بلیت بتوسط AS و نه TGS صادر شده است. وقتی یک کلاینت از TGS درخواست یک بلیت اعطاکنندهٔ- سرویس می‌نماید، یک بلیت اعطاکنندهٔ بلیت را که از AS دریافت کرده است عرضه می‌دارد. در نسخهٔ ۴، این تنها راه اخذ یک بلیت اعطاکنندهٔ- سرویس بود. نسخهٔ ۵ این قابلیت جدید را فراهم می‌سازد که کلاینت بتواند یک بلیت اعطاکنندهٔ- سرویس را مستقیماً از AS دریافت نماید. فایدهٔ این امر چنین است: یک سِرور، مثل یک سِرور تعویض‌کنندهٔ کلمهٔ عبور، ممکن است علاقه‌مند باشد تا بداند که کلمهٔ عبور کلاینت جدیداً تست شده است.

پرچم PRE-AUTHENT، اگر افراشته باشد، نمایشگر این مطلب است که وقتی AS تقاضای اولیه را دریافت کرده است (پیام (۱))، اعتبار کلاینت را قبل از صدور یک بلیت سنجیده است. شکل دقیق این پیش‌اعتبارسنجی، تعیین‌نشده باقی مانده است. برای مثال، در نسخهٔ ۵ ساخت MIT، پیش‌اعتبارسنج برچسب زمانی را رمزنگاری کرده است که در حالت پیش‌فرض فعال خواهد بود. وقتی کاربری می‌خواهد تا یک بلیت دریافت نماید، بایستی برای AS یک بلوک پیش اعتبارسنج

جدول ۴-۴ پرچم‌های Kerberos Version 5

INITIAL	این بلیت با استفاده از پروتکل AS صادر شده است و بر اساس بلیت اعطاکنندهٔ بلیت صادر نشده است.
PRE-AUTHENT	در هنگام اعتبارسنجی اولیه، کلاینت قبل از صدور بلیت بتوسط KDC اعتبارسنجی شده است.
HW-AUTHENT	پروتکلی که برای اعتبارسنجی اولیه بکارگرفته شده است نیاز به استفاده از سخت‌افزاری داشته است که انتظار می‌رود منحصراً در مالکیت کلاینت نام برده شده، بوده باشد.
RENEWABLE	به TGS می‌گوید که این بلیت می‌تواند برای کسب یک بلیت جایگزین که در تاریخ دیرتری منقضی می‌گردد بکار رود.
MAY-POSTDATE	به TGS می‌گوید که یک بلیت آتی می‌تواند بر اساس این بلیت اعطاکنندهٔ بلیت صادر شود.
POSTDATED	نشان می‌دهد که این بلیت تمدید شده است. سرور انتهائی می‌تواند میدان زمان اعتبارسنجی را کنترل کرده تا از زمان اعتبارسنجی اولیه خبردار گردد.
INVALID	این کلید دارای اعتبار نبوده و بایستی قبل از استفاده بتوسط KDC اعتبار آن تأئید شود.
PROXIABLE	به TGS می‌گوید که یک بلیت اعطاکنندهٔ سرویس جدید با یک آدرس شبکهٔ متفاوت می‌تواند بر اساس بلیت عرضه‌شده صادر شود.
PROXY	نشان می‌دهد که این بلیت یک پروکسی است.
FORWARDABLE	به TGS می‌گوید که یک بلیت اعطاکنندهٔ بلیت جدید با یک آدرس شبکهٔ متفاوت می‌تواند بر اساس این بلیت اعطاکنندهٔ بلیت صادر شود.
FORWARDED	نشان می‌دهد که این بلیت یا به جلورانده شده است و یا بر اساس یک اعتبارسنجی که شامل یک بلیت اعطاکنندهٔ بلیت به جلورانده بوده است صادر شده است.

ارسال کند که شامل یک مغشوش‌کنندهٔ تصادفی، شمارهٔ نسخه و یک برچسب زمانی بوده که با کلید مبتنی بر کلمهٔ عبور کاربر رمزنگاری شده باشد. AS بلوک را رمزگشائی کرده و یک بلیت اعطاکنندهٔ بلیت را پس نمی‌فرستد مگر اینکه برچسب زمانی موجود در بلوک پیش‌اعتبارسنج، در محدودهٔ مجاز زمانی باشد (محدوده‌ای که انحراف پالس ساعت و تأخیرهای شبکه را منظورکرده باشد). امکان دیگر استفاده از کارت هوشمندی است که مرتباً کلمات عبور متغیری را که در پیام‌های پیش-اعتبارسنجی شده وجود دارد تولید می‌کند. کلمات عبور تولیدشده بتوسط کارت می‌توانند مبتنی بر کلمهٔ عبور کاربر باشند ولی بتوسط کارت طوری تغییر یابند که در عمل کلمات عبور متفاوتی بکار رود. این امر از حمله‌ای که بخواهد بر اساس حدس‌زدن کلمات عبور ساده صورت پذیرد جلوگیری می‌نماید. اگر یک کارت هوشمند و یا دستگاه مشابهی بکارگرفته شود، این موضوع بتوسط پرچم PRE-AUTHENT نشان داده خواهد شد.

وقتی یک بلیت دارای طول عمرزیادی است، خطر سرقت و استفادهٔ غیرمجاز و طولانی از آن بتوسط دشمن زیاد است. اگر از طول عمر کمتری برای کاهش این تهدید استفاده شود، آنگاه افزایش سرباره بدلیل درخواست مکرر بلیت‌های جدید اجتناب‌ناپذیر است. در مورد یک بلیت اعطاکنندهٔ بلیت، کلاینت یا بایستی کلید سرّی کاربر را ذخیره نموده، که این کار دارای ریسک بالایی است، و یا بایستی مکرراً از کاربر درخواست کلمهٔ عبور نماید. یک روش بینابینی در این مورد استفاده از بلیت‌های قابل بروزرسانی است. یک بلیت دارای یک پرچم افراشتهٔ RENEWABLE شامل دو زمان انقضاء است: یکی برای این بلیت خاص و دیگری که آخرین زمان مجاز مربوط به انقضاء را نشان می‌دهد. یک کلاینت می‌تواند بلیت را به TGS

عرضه کرده و یک زمان انقضای جدید درخواست کند. اگر زمان جدید در محدودهٔ آخرین اندازهٔ مجاز قرار داشته باشد، TGS می‌تواند یک بلیت جدید با یک زمان اجلاس جدید و یک زمان انقضای مشخص صادر نماید. مزیت این مکانیسم این‌است که TGS ممکن است در صورتی که گزارشی از سرقت بلیت داشته باشد از صدور بلیت جدید امتناع ورزد.

یک کلاینت ممکن است تقاضا کند که AS یک بلیت اعطاکنندهٔ بلیت با پرچم افراشتهٔ MAY-POSTDATE برای او صادر نماید. کلاینت آنگاه می‌تواند این بلیت را برای درخواست یک بلیت که دارای پرچم‌های افراشتهٔ POSTDATED و INVALID هستند مورد استفاده قرار دهد. متعاقب آن، کلاینت ممکن است بلیت منقضی‌شده را برای تأئید ارائه کند. این روش ممکن است برای اجرای عملیات گروهی، روی یک سِرور که متناوباً نیاز به بلیت دارد مفید باشد. کلاینت می‌تواند برای این اجلاس تعدادی بلیت با زمان‌های متنوع را یکباره بدست آورد. تمام این بلیت‌ها بجز اولی در ابتدا فاقد اعتبارند. وقتی عملیات در زمان بجائی می‌رسد که به بلیت جدیدی نیاز است، کلاینت می‌تواند بلیت مناسب امر را دارای اعتبار نماید. با بکارگرفتن این روش، کلاینت مجبور نیست تا مکرراً بلیت اعطاکنندهٔ بلیت خود را بکار گرفته تا یک بلیت اعطاکنندهٔ سرویس بدست آورد.

در نسخهٔ ۵، این امکان وجود دارد که یک سِرور از جانب کلاینت بصورت یک پروکسی عمل نماید که اثر آن استفاده از اعتبار و امتیازات کلاینت برای درخواست سرویس از سِرور دیگر است. اگر یک کلاینت بخواهد از این مکانیسم استفاده نماید، درخواست یک بلیت اعطاکنندهٔ بلیت با پرچم افراشتهٔ PROXIABLE می‌نماید. وقتی این بلیت به TGS عرضه می‌شود، TGS مجاز به صدور یک بلیت اعطاکنندهٔ بلیت با آدرس شبکهٔ دیگری خواهد بود. این بلیت آخر، دارای پرچم PROXY افراشته خواهد بود. کاربری که چنین بلیتی را دریافت میکند، ممکن است آن را پذیرفته و یا نیاز به اعتبارسنجی اضافی برای فراهم‌آوردن یک ردّپای ممیزی داشته باشد.

مقولهٔ پروکسی یک مورد محدود از روش عام‌تر و قوی‌تر به‌جلوراندن است. اگر پرچم FORWARDABLE در یک بلیت افراشته باشد، آنگاه یک TGS می‌تواند برای متقاضی، یک بلیت اعطاکنندهٔ بلیت با یک آدرس شبکهٔ متفاوت صادر کند که پرچم FORWARDED آن افراشته باشد. این بلیت می‌تواند به یک TGS دور عرضه شود. این توانائی به کلاینت اجازه می‌دهد تا به یک سِرور در قلمرو دیگر دست یابد، بدون اینکه نیاز باشد که هر Kerberos یک کلید سرّی با هر Kerberos دیگر در سایر قلمروها را به اشتراک بگذارد. بعنوان مثال، قلمروها می‌توانند دارای ساختار درختی باشند. در اینصورت یک کلاینت می‌تواند یک شاخه درخت را تا یک گرهٔ مشترک بالا رفته و سپس برای دسترسی به قلمرو هدف از شاخهٔ دیگر پائین آید. هرقدم این راه‌پیمائی شامل به‌جلوراندن یک بلیت اعطاکنندهٔ بلیت به TGS بعدی مسیر است.

۴-۲ سرویس اعتبارسنجی X.509

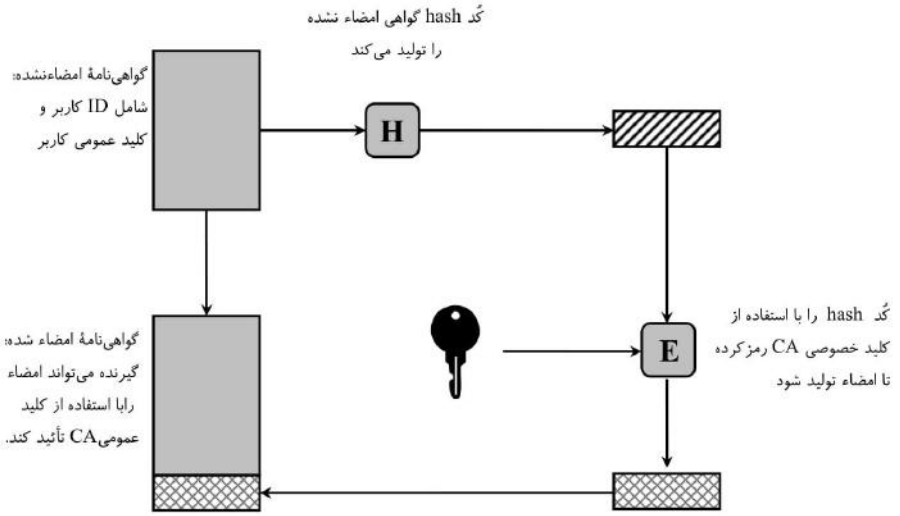
توصیه‌نامهٔ X.509 که مربوط به ITU-T است، بخشی از سری توصیه‌نامه‌های X.500 است که یک سرویس فهرست راهنما را تعریف می‌کند. فهرست راهنما در واقع یک سِرور و یا مجموعه‌ای توزیع‌شده از سِرورهاست که یک پایگاه اطلاعاتی داده در موردکاربران را نگهداری می‌کند. اطلاعات شامل یک نگاشت از نام کاربر به آدرس شبکه و همچنین سایر صفات و اطلاعات مربوط به کاربر است.

X.509 یک محدودهٔ کاری برای فراهم‌آوردن سرویس‌های اعتبارسنجی بتوسط فهرست راهنمای X.500 برای کاربران خود فراهم می‌سازد. فهرست راهنما ممکن است همانند یک صندوقچه نگهداری گواهی‌نامه‌های کلید- عمومی عمل نماید. هر گواهی شامل کلید عمومی یک کاربر بوده و بتوسط کلید خصوصی یک مسئول قابل اعتماد صدور گواهی‌نامه امضاء می‌گردد. علاوه‌براین، X.509 پروتکل‌های اعتبارسنجی دیگری که مبتنی بر استفاده از گواهی‌نامه‌های کلید- عمومی هستند را تعریف می‌نماید.

X.509 یک استاندارد مهم است زیرا ساختار گواهی‌نامه و پروتکل‌های اعتبارسنجی تعریف‌شده در X.509 در مقوله‌های متعددی مورد استفاده قرار می‌گیرند. مثلاً فرمت گواهی X.509 در S/MIME (فصل پنجم)، امنیتIP (فصل ششم)، و SSL/TLS و SET (فصل هفتم) بکار می‌روند.

X.509 بدواً در سال ۱۹۸۸ منتشر گردید. متعاقباً تغییراتی در این استاندارد داده شد تا برخی نگرانی‌های امنیتی ذکرشده در [IANS90] و [MITC90] مورد توجه قرار گیرند. توصیه‌نامهٔ اصلاح‌شده در سال ۱۹۹۳ مستند گردید. نسخهٔ سوم آن در سال ۱۹۹۵ انتشار یافت و در سال ۲۰۰۰ مورد بازنگری قرار گرفت.

X.509 بر مبنای استفاده از رمزنگاری کلید- عمومی و امضاءهای دیجیتال قرار گرفته است. استاندارد، استفاده از یک الگوریتم خاص را اجباری نمی‌سازد ولی RSA را توصیه می‌کند. فرض شده است که تکنیک امضاء دیجیتال نیاز به استفاده از یک تابع درهم‌ساز دارد. بازهم استاندارد الگوریتم درهم‌سازی خاصی را پیشنهاد نمی‌کند. توصیه‌نامهٔ ۱۹۸۸ شامل توصیف یک الگوریتم درهم‌سازی توصیه‌شده بود که بعداً مشخص گردید که ناامن بوده و بنابراین از توصیه‌نامهٔ سال ۱۹۹۳ حذف گردید. شکل ۳-۴ نحوهٔ تولید یک گواهی‌نامهٔ کلید- عمومی را نشان می‌دهد.



شکل ۳-۴ استفاده از گواهی‌نامهٔ کلید- عمومی

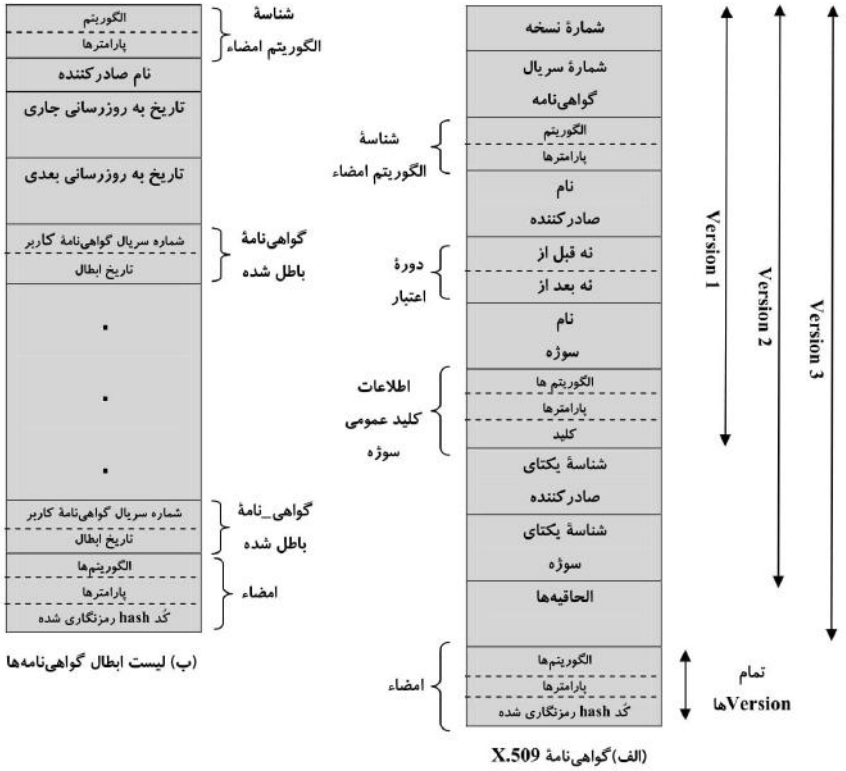
گواهی‌نامه‌ها (Certificates)

قلب روش X.509، گواهی‌نامهٔ کلید- عمومی مرتبط با هر کاربر است. فرض براین است که گواهی‌نامه‌های کاربران بتوسط یک مسئول قابل اعتماد صدور گواهی‌نامه Certification Authority (CA) صادر شده و بتوسط CA یا کاربر در فهرست راهنما درج شده است. سِرور مخصوص فهرست راهنما، خود مسئول تولید کلیدهای عمومی و یا عملیات صدور گواهی نبوده و صرفاً محل قابل دسترسی ساده‌ای است که گواهی‌نامه‌های درخواستی کاربران را در اختیار آنها می‌گذارد.

شکل ۴-۴الف فرمت عمومی یک گواهی‌نامه را نشان میدهد که شامل عناصر زیر است:

- **شمارهٔ نسخه**: بین نسخه‌های مختلف انتشاریافتهٔ فرمت گواهی‌نامه فرق می‌گذارد. پیش‌فرض آن Version 1 است. اگر شناسهٔ یکتای صادرکننده و یا شناسهٔ یکتای سوژه در فرمت حضور داشته باشند، اندازهٔ این میدان بایستی Version 2 باشد. اگر یکی یا بیشتر از الحاقیه‌ها موجود باشند، نسخه بایستی Version 3 باشد.
- **شمارهٔ سریال**: یک عدد صحیح و یکتا در نزد صادرکنندهٔ CA است که بدون هیچگونه ابهامی فقط مرتبط با این گواهی‌نامه است.
- **شناسهٔ الگوریتم امضاء**: الگوریتم و پارامترهای مربوطی است که برای امضاء این گواهی‌نامه بکارگرفته شده است. چون این اطلاعات در میدان امضاء واقع در انتهای این گواهی دوباره تکرار می‌شود، این میدان استفادهٔ کمی دارد.
- **نام صادرکننده**: نام CA صادرکننده و امضاءکنندهٔ این گواهی‌نامه در فرمت X.500 .
- **دورهٔ اعتبار**: شامل دو تاریخ است: تاریخ اول و تاریخ آخری که این گواهی‌نامه بین این دو تاریخ اعتبار دارد.
- **نام سوژه**: نام کاربری که این گواهی‌نامه مربوط به اوست. یعنی این گواهی، کلید عمومی سوژه‌ای را که کلید خصوصی مرتبط را در اختیار دارد تأئید می‌نماید.
- **اطلاعات کلید عمومی سوژه**: کلید عمومی سوژه بعلاوهٔ یک شناسهٔ مرتبط با الگوریتمی که این کلید برای آن بکار خواهد رفت، به همراه پارامترهای مربوطه.
- **شناسهٔ یکتای صادرکننده**: یک میدان از دنبالهٔ بیت‌ها که برای شناسائی یکتای CA صادرکننده بکار میرود، در صورتی‌که نام X.500 برای واحدهای مختلف مکرراً استفاده شده باشد.
- **شناسهٔ یکتای سوژه**: یک میدان از دنبالهٔ بیت‌ها که برای شناسائی یکتای سوژه بکار میرود، در صورتی‌که نام X.500 برای واحدهای مختلف مکرراً استفاده شده باشد.
- **الحاقیه‌ها**: مجموعه‌ای از یک یا چند میدان الحاق شده. الحاقیه‌ها در نسخهٔ ۳ به توصیه‌نامه اضافه شده و بعداً در همین بخش توصیف خواهند شد.
- **امضاء**: همهٔ میدان‌های دیگر گواهی‌نامه را پوشش می‌دهد. این شامل کُد hash سایر میدان‌ها بوده که بتوسط کلید خصوصی CA رمزنگاری می‌شود. این میدان شامل شناسهٔ الگوریتم امضاء است.

میدان‌های یکتای شناسه‌ها در نسخهٔ ۲ اضافه شد تا مشکلات احتمالی مربوط به استفادهٔ مجدد از سوژه و یا نام‌های صادرکنندگان گواهی‌نامه در طول زمان را حل کند. این میدان‌ها بندرت مورد استفاده قرار می‌گیرند.



شکل ۴-۴ فرمت‌های X.509

استاندارد از فرم زیر برای تعریف یک گواهی استفاده می‌کند:

$$CA\langle\langle A\rangle\rangle = CA\{V, SN, AI, CA, T_A, A, Ap\}$$

که درآن

$Y\langle\langle X\rangle\rangle$ = گواهی‌نامهٔ کاربر X که بتوسط مسئول صدور گواهی Y صادر شده است.

$Y\{I\}$ = امضاء I بتوسط Y. این شامل I و کُد hash رمزنگاری شدهٔ آن است که به آن وصل شده است.

CA گواهی‌نامه را با کلید خصوصی خود امضاء می‌کند. اگر کلید عمومی مرتبط با آن برای یک کاربر شناخته شده باشد، آنگاه کاربر میتواند تأئید کند که گواهی امضاءشده بتوسط CA معتبر است. این یک روش مرسوم برای امضاء دیجیتال است که در شکل ۲-۳ب نشان داده شده است.

اخذ گواهی‌نامه یک کاربر

گواهی‌نامه‌هائیکه بتوسط CA برای کاربران صادر می‌شود، دارای مشخصات زیر است:

- هرکاربری که به کلید عمومی CA دسترسی داشته باشد می‌تواند کلید عمومی گواهی‌شدهٔ کاربر را تأئید نماید.
- هیچکس بغیر از مسئول صدور گواهی نمی‌تواند بدون اینکه مچش گیرافتد، گواهی‌نامه را دستکاری نماید.

نظر به اینکه گواهی‌نامه‌ها غیرقابل جعل‌اند، می‌توان آنها را در یک فهرست راهنما قرار داد بدون اینکه نیازی به حفاظت از آنها باشد.

اگر همهٔ کاربران، مشترکین فقط یک CA باشند، آنگاه اعتماد مشترکی نسبت به آن CA در همهٔ آنها وجود دارد. تمام گواهی‌های کاربران را می‌توان در فهرست راهنمائی قرار داد که بتوسط همهٔ آنها قابل دسترس باشد. علاوه بر آن یک کاربر می‌تواندگواهی‌نامهٔ خود را مستقیماً به کاربر دیگری منتقل نماید. در هریک از دو مورد، همین‌که B گواهی A را در اختیار خود گرفت، B اطمینان دارد که پیام‌هائی که او با کلید عمومی A رمزنگاری می‌کند نسبت به شنود امن بوده و پیام‌هائی که با کلید خصوصی A امضاءشده باشند غیرقابل جعل‌اند.

اگر تعداد کاربران خیلی زیاد باشد، عملی نخواهد بود که تمام آنها مشترک یک CA خاص باشند. چون این CA است که گواهی‌ها را امضاء می‌کند، هرکاربر مشترک CA بایستی یک کپی از کلید عمومی CA را در اختیار داشته باشد تا بتوسط آن بتواند گواهی‌ها را تأئید نماید. این کلید عمومی بایستی بصورت کاملاً امنی(از نظر اصالت و اعتبار) در اختیار هر کاربر قرار گیرد تا کاربر نسبت به گواهی‌های صادره اطمینان داشته باشد. بنابراین با تعداد کاربران زیاد، ممکن است راه حل عملی‌تر این باشد که تعدادی CA وجود داشته باشند که هریک از آنها کلید عمومی خود را بطور امنی در اختیار بخشی از کاربران قرار دهند.

حال فرض کنید که A یک گواهی‌نامه از مسئول صدور گواهی X_1 اخذ نموده و B نیز یک گواهی‌نامه از CA خود یعنی X_2 اخذ کرده باشد. اگرA بطور امنی از کلید عمومی X_2 خبر نداشته باشد، آنگاه گواهی‌نامهٔ B که بتوسط X_2 صادرشده است برای A بی‌ارزش خواهد بود. ولی اگر دو CA کلیدهای عمومی خود را بطور امنی مبادله کرده باشند، آنگاه روش زیر A را قادر خواهد ساخت تا کلید عمومی B را بدست آورد.

۱- A از فهرست راهنما، گواهی‌نامهٔ X_2 امضاء شده بتوسط X_1 را می‌گیرد. چون A بطور امنی از کلید عمومی X_1 باخبر است، A میتواند کلید عمومی X_2 را از گواهی‌نامهٔ او بدست آورده و آن را بتوسط امضاء X_1 که روی گواهی‌نامه است تأئید نماید.

۲- A سپس به فهرست راهنما برگشته و گواهی‌نامه B را که بتوسط X_2 امضاء شده است دریافت می‌کند. حال چون A یک کپی مطمئن از کلید عمومی X_2 را در اختیار دارد، A میتواند امضاء را تأئید کرده و بطور امنی کلیدعمومیB را استخراج نماید.

A برای بدست آوردن کلید عمومی B از زنجیره‌ای از گواهی‌نامه‌ها استفاده کرده است. برحسب علائم X.509 این زنجیره چنین بیان می‌شود:

$$X_1<<X_2>>X_2<<B>>$$

بهمین روش B میتواند کلید عمومی A را با زنجیرهٔ معکوس بدست آورد.

$$X_2<<X_1>>X_1<<A>>$$

این روش لازم نیست که فقط محدود به دو گواهی‌نامه باشد. یک مسیر طولانی از CAها را می‌توان برای ایجاد یک زنجیر در نظر گرفت. یک زنجیر با N عنصر چنین بیان می‌گردد:

$$X_1<<X_2>>X_2<<X_3>> \cdots X_N<<B>>$$

در این مورد، هرزوج از CAها در زنجیرهٔ (X_i , X_{i+1}) بایستی برای یکدیگر گواهی‌نامه صادر کرده باشند.

تمام این گواهی‌نامه‌های CAها بتوسط CAهای دیگر لازم است که در فهرست راهنما وجود داشته باشند و کاربر لازم است بداندکه اینها چگونه با هم مرتبط‌اند تا بتواند مسیر را تا یافتن کلید عمومی B دنبال نماید. X.509 پیشنهاد می‌کند که CAها در یک ساختار سلسله‌مراتبی طوری سازمان‌دهی شوند که ناوبری بین آنها ساده باشد.

شکل ۵-۴ که از X.509 گرفته شده است، مثالی از چنین ساختاری است. دایره‌های بهم وصل شده، ارتباط سلسله‌مراتبی CAها را نشان می‌دهد و مربع‌های مربوطه نمایشگر گواهی‌نامه‌های نگهداری شده در فهرست راهنمای هر CAاند. اقلام فهرست راهنما در هر CA شامل دو نوع گواهی‌اند:

- **گواهی‌های مستقیم:** گواهی‌های X که بتوسط سایر CAها صادر شده‌اند.
- **گواهی‌های معکوس:** گواهی‌های تولیدشده بتوسط X که گواهی‌نامهٔ سایر CAها هستند.

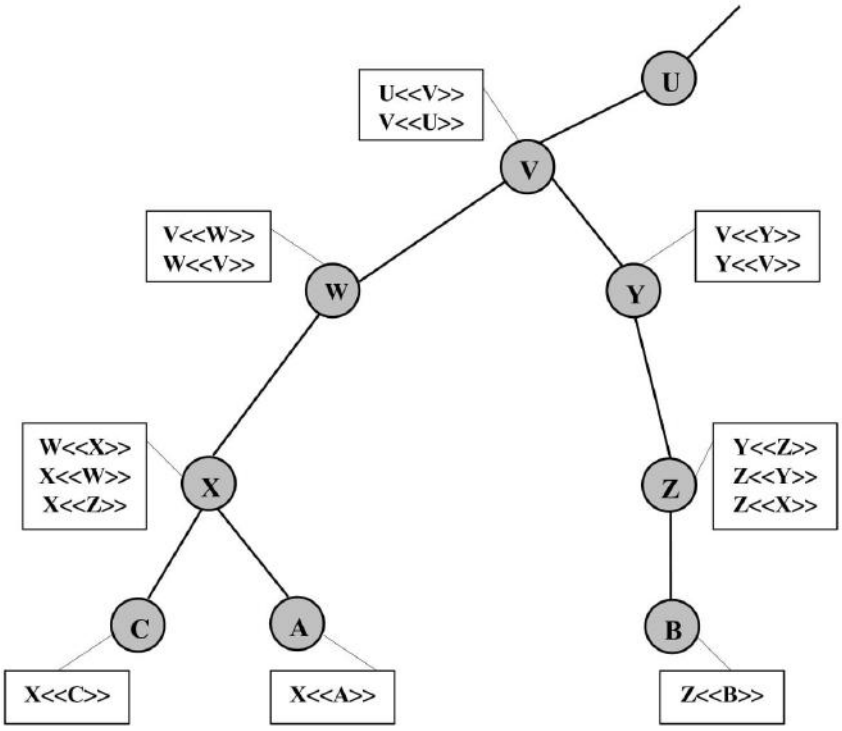
در این مثال، کاربرA میتواند گواهی‌های زیر را از فهرست راهنما بدست آورده و مسیر را تا B دنبال کند:

$$X<<W>>W<<V>>V<<Y>>Y<<Z>>Z<<B>>$$

وقتی A این گواهی‌ها را بدست آورد، می‌تواند مسیر را بترتیب دنبال کرده تا یک کپی مورد اطمینان از کلید عمومی B را بدست آورد. با استفاده از این کلید عمومی، A می‌تواند پیام‌های رمزنگاری‌شدهٔ خود را برای B بفرستد. اگر A تمایل داشته باشد تا پیام‌های رمزنگاری‌شده‌ای را از طرف B دریافت کند و یا پیام‌هائی را که برای B ارسال میشود امضاء نماید، آنگاه B نیاز به کلید عمومی A خواهد داشت که میتواند آن را از مسیر زیر بدست آورد:

$$Z<<Y>>Y<<V>>V<<W>>W<<X>>X<<A>>$$

B می‌تواند این مجموعهٔ گواهی‌ها را از فهرست راهنما استخراج کرده و یا A می‌تواند آنها را بصورت بخشی از پیام اولیه برای B ارسال دارد.



شکل ۵-۴ سلسله مراتب X.509 ؛ یک مثال فرضی

ابطال گواهی‌نامه‌ها

از شکل ۴-۴ بخاطر آورید که هر گواهی‌نامه همانند یک کارت اعتباری دارای یک دورهٔ اعتبار است. معمولاً بایستی قبل از انقضای دورهٔ اعتبار، گواهی‌نامهٔ جدیدی صادر شود. علاوه براین ممکن است در مواردی علاقه‌مند باشیم تا به دلایل زیر یک گواهی‌نامه را قبل از انقضای مهلت آن باطل کنیم:

۱- احتمال داده شود که کلید خصوصی کاربر لو رفته است.
۲- کاربر، دیگر بتوسط این CA تأئید نمی‌شود.
۳- احتمال داده شود که گواهی‌نامهٔ CA لو رفته است.

هر CA بایستی لیستی که شامل تمام گواهی‌های باطل شده ولی منقضی نشده صادره بتوسط آن CA، اعم از گواهی‌های صادرشده برای کاربران، یا سایر CAها را نگهداری نماید. این لیست‌ها همچنین بایستی در فهرست راهنما اعلان شود.

هر لیست گواهی‌های باطل شده certificate revocation list (CRL)،که در فهرست راهنما اعلان می‌گردد بتوسط صادرکننده امضاءشده و شامل (شکل ۴-۴ب) نام صادرکننده، تاریخی که لیست تولید شده است، تاریخی که CRL بعدی قرار است منتشر شود و یک ورودی برای هر گواهی باطل شده است. هر ورودی شامل شمارهٔ سریال یک گواهی‌نامه و تاریخ ابطال آن گواهی‌نامه است. چون شمارهٔ سریال‌های مربوط به یک CA یکتا هستند، شمارهٔ سریال برای شناسائی یک گواهی‌نامه کافی است.

وقتی یک کاربر یک گواهی‌نامه در یک پیام را دریافت کرد، او بایستی تحقیق کند که گواهی‌نامه باطل نشده باشد. کاربر می‌تواند هربار که یک گواهی‌نامه دریافت می‌کند، فهرست راهنما را کنترل نماید. بـــرای جلوگیری از تأخیر زمـــانی (و احتمالاً هزینه) مرتبط با جستجوی فهرست راهنما، محتمل است که کاربر یک حافظهٔ محلی را برای نگهداری گواهی‌نامه‌ها و لیست گواهی‌نامه‌های باطل‌شده در اختیار داشته باشد.

رَویه‌های اعتبارسنجی

X.509 همچنین شامل سه رَویهٔ مختلف اعتبارسنجی است که برای استفاده در کاربردهای متنوعی طراحی شده‌اند. تمام این رَویه‌ها از امضاءهای کلید- عمومی استفاده می‌کنند. فرض براین است که دو طرف کلید عمومی یکدیگر را می‌دانند که این امر یا با کسب گواهی‌نامه‌های یکدیگر از فهرست راهنما و یا بدلیل اینکه گواهی‌نامه در پیام اولیه هرطرف وجود داشته است امکان‌پذیر خواهد بود.
شکل ۶-۴ این سه رَویه را نشان میدهد.

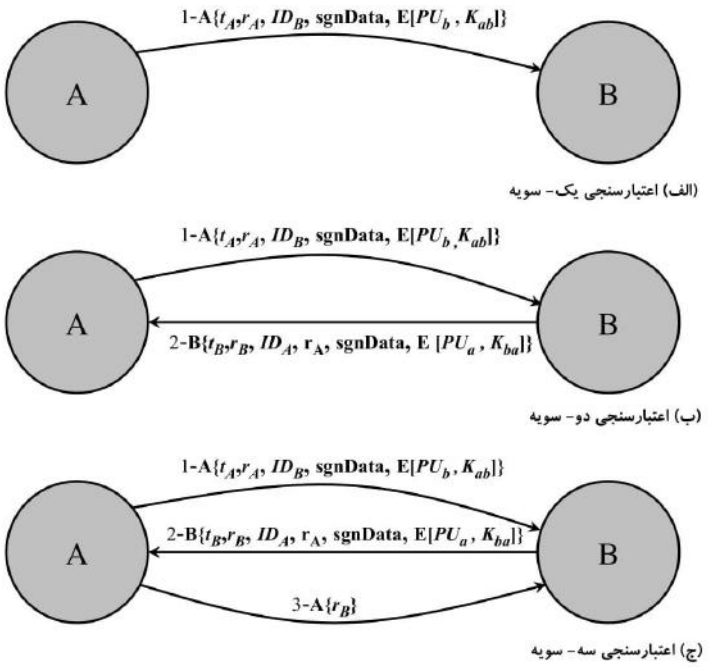
اعتبارسنجی یک- سویه

اعتبارسنجی یک- سویه شامل یک بار انتقال اطلاعات از سوی کاربر(A) برای کاربر(B) است و موارد زیر را روشن می‌سازد:

۱- هویت A و پیامی که بتوسط A تولیدشده است.
۲- اینکه پیام به مقصدB است.
۳- صحت و دست اول بودن پیام (اینکه چندبار ارسال نشده باشد).

توجه کنید که در این روش تنها هویت واحد شروع‌کنندهٔ ارتباط، و نه واحد پاسخ‌دهنده به ارتباط، تأئید می‌شود.

پیام حداقل دارای یک برچسب زمانی t_A، یک nonce (r_A) و هویت B بوده و بتوسط کلید خصوصی A امضاء می‌شود. برچسب زمانی شامل یک بخش اختیاری زمان شروع و زمان خاتمه است. این امر از تأخیر در تسلیم پیام جلوگیری می‌کند. nonce می‌تواند برای تشخیص حملات بازخوانی بکار رود. اندازهٔ nonce بایستی در طول زمان اعتبار پیام، یکتا باشد. بنابراینB می‌تواند nonce را تا زمان انقضای آن ذخیره کرده و هر پیام جدید با همان nonce را نپذیرد.



شکل ۶-۴ رَویهٔ های قدرتمند اعتبارسنجی X.509

برای اعتبارسنجی صِرف، پیام بسادگی برای ارائهٔ امتیازات به B بکار می‌رود. پیام همچنین ممکن است شامل اطلاعاتی باشد که بایستی عرضه شود. این اطلاعات، SgnData، در محدودهٔ امضاء بوده و اعتبار و اصالت آن را تضمین می‌نماید. پیام همچنین ممکن است برای رساندن یک کلید اجلاس به B بکار رود و با کلید عمومی B رمزشده باشد.

اعتبارسنجی دو- سویه

علاوه بر سه موردی که در بالا ذکرشد، اعتبارسنجی دو- سویه مقوله‌های اضافی زیر را نیز مورد توجه قرار می‌دهد:

۴- هویت B و اینکه پاسخ پیام از سوی B، واقعاً بتوسط خود B تولیدشده است.

۵- اینکه پیام به مقصد A ارسال شده است.

۶- صحت و دست اول بودن پیام

بنابراین اعتبارسنجی دو- سویه به هردو طرف درگیر اجازه می‌دهد تا هویت طرف مقابل را تأئید نمایند.

پیام پاسخ، شامل nonce از طرف A برای اعتباربخشیدن به پاسخ است. پیام همچنین شامل یک برچسب زمانی و nonce تولیدشده بتوسط B است. همانند قبل، پیام ممکن است شامل اطلاعات اضافی امضاءشده و یک کلید اجلاس باشد که بتوسط کلید عمومی A رمزنگاری شده باشد.

اعتبارسنجی سه- سویه

در اعتبارسنجی سه- سویه، یک پیام نهائی از A به B نیز وجود دارد که شامل یک کپی امضاءشدهٔ nonce (r_B) است. هدف این طراحی این است که برچسب‌های زمانی نیازی به کنترل نداشته باشند. چون هر دو nonce بتوسط طرف مقابل برگشت داده میشوند، هرطرف میتواند nonce برگشتی را کنترل کرده و حملات احتمالی بازخوانی را تشخیص دهد. این تکنیک وقتی پالس‌های ساعت همزمان در دسترس نباشند مورد لزوم است.

نسخهٔ سوم X.509

فرمت نسخهٔ دوم X.509 تمام اطلاعاتی که طراحی‌ها و تجارب پیاده‌سازی اخیر نیاز آن را ثابت کرده است، منتقل نمی‌کند. [FORD95] الزامات زیرکه در نسخهٔ دوم برآورده نشده است، را بیان کرده است:

۱- میدان سوژه برای معرفی هویت یک صاحب کلید به یک استفاده‌کننده از کلید عمومی کافی نیست. نام‌های X.509 ممکن است نسبتاً کوتاه بوده و فاقد جزئیات هویتی لازم مورد نیاز کاربر باشند.

۲- میدان سوژه همچنین برای برخی کاربردها که معمولاً واحدها را با آدرس e-mail، یک URL و یا بطریق دیگری در اینترنت شناسائی می‌کنند، ناکافی است.

۳- این نیاز وجود داردکه اطلاعات مربوط به خط‌مشی امنیتی نشان داده شود. این امر یک کاربرد امنیتی یا عمل امنیتی همچون IPSec را قادر می‌سازد تا یک گواهی X.509 را به یک خط‌مشی خاص ربط دهد.

۴- این نیاز وجود دارد تا صدماتی که ممکن است از یک CA معیوب و یا بداندیش ناشی شود را با ایجاد محدودیت‌هائی در کاربرد یک گواهی‌نامهٔ خاص کم کرد.

۵- این که بتوان کلیدهای مختلف مجزا که بتوسط صاحب آن در زمان‌های مختلف مورد استفاده قرار گرفته است را شناسائی نمود، امری مهم است. این خصیصه، مدیریت طول عمر کلید را حمایت کرده و علی‌الخصوص این توانائی که بتوان جفت کلیدها را برای کاربران و CAها در فواصل زمانی منظم و یا تحت شرایط استثنائی به روز درآورد را بوجود می‌آورد.

بجای اینکه مرتباً میدان‌های جدیدی به فرمت ثابت اضافه نمایند، تولیدکنندگان استانداردها احساس کرده‌اند که روش انعطاف‌پذیرتری مورد نیاز است. بنابراین نسخهٔ ۳ شامل تعدادی الحاقیهٔ اختیاری است که ممکن است به فرمت نسخهٔ ۲ اضافه کرد. هریک از این الحاقیه‌ها شامل یک شناسهٔ الحاقیه، یک نمایشگر اهمیت الحاقیه و یک اندازه الحاقیه است. نمایشگر اهمیت الحاقیه بیانگر این مسأله است که آیا می‌توان با خاطری آسوده از یک الحاقیه صرفنظر کرد؟ اگر نمایشگر دارای اندازهٔ true باشد و واحد اجرا آن را نشناسد، گواهی‌نامه غیرقابل قبول تلقی خواهد شد.

الحاقیه‌های گواهی‌نامه‌ها در سه گروه اصلی قرار می‌گیرند: اطلاعات کلید و خط‌مشی، مشخصات سوژه و صادرکنندهٔ گواهی، و محدودیت‌های روند گواهی کردن.

اطلاعات کلید و خط‌مشی

این الحاقیه‌ها، اطلاعات اضافی مربوط به کلید سوژه و کلید صادرکنندهٔ گواهی‌نامه را حمل کرده و بعلاوه نمایش‌دهندهٔ خط‌مشی گواهی‌نامه می‌باشند. خط‌مشی یک گواهی‌نامه شامل مجموعه‌ای از قوانین تبیین شده بوده که نمایش‌دهندهٔ قابلیت کاربرد یک گواهی‌نامه در یک جمعیت خاص و یا دسته‌ای از کاربردها با نیازهای امنیتی مشترک می‌باشند. بعنوان مثال یک نوع خط‌مشی ممکن است قابل اعمال به اعتبارسنجی اسناد مالی EDI (Electronic Data Interchange) برای تجارت اقلام در محدودهٔ قیمت مشخصی باشد.

این ناحیه شامل اقلام زیر است:

- **شناسهٔ کلید CA**: هویت کلید عمومی که بایستی برای تأئید امضاء در این گواهی‌نامه و یا CRL بکار رود را تعیین می‌کند. باعث می‌شود که بتوان بین کلیدهای متفاوت یک CA فرق گذاشت. یکی از موارد استعمال این میدان به‌روز درآوردن زوج کلید CA است.
- **شناسهٔ کلید سوژه**: هویت کلید عمومی که گواهی‌نامهٔ آن در شرف صادرشدن است را تعیین می‌کند. برای به روز رساندن کلید سوژه مفید است. همچنین یک سوژه ممکن است جفت کلیدهای متعدد و نظیرآن گواهی‌های مختلف برای اهداف متفاوت (مثل امضاء دیجیتال و موافقت در مورد کلید رمزنگاری) داشته باشد.
- **موارد استعمال کلید**: محدودیت‌های اعمال شده، نظیر اهدافی که برای آن و سیاست‌هائی که تحت آن کلید عمومی گواهی‌شده می‌تواند مورد استفاده قرار گیرد را نشان می‌دهد و ممکن است نشان‌دهندهٔ یک یا چند مورد زیر باشد: امضاء دیجیتال، عدم انکار، رمزنگاری دیتا، توافق نسبت به کلید، تأئید امضاء CA در گواهی‌نامه‌ها، تأئید امضاء CA در CRLها.
- **دورهٔ استفاده از کلید خصوصی**: مدت زمان قابل استفاده از کلید خصوصی نظیر یک کلید عمومی را تعیین می‌کند. معمولاً کلید خصوصی در طول دوره‌های مختلفی از دورهٔ اعتبار کلید عمومی استفاده می‌شود. مثلاً در رابطه با کلیدهای امضاء دیجیتال، زمان قابل استفاده از کلید خصوصی معمولاً کوتاهتر از دورهٔ تأئید کلید عمومی است.
- **خط‌مشی‌های گواهی‌نامه**: از گواهی‌نامه‌ها ممکن است در محیط‌هائی استفاده شود که خط‌مشی‌های مختلفی حاکم است. این الحاقیه، خط‌مشی‌هائی که گواهی‌نامه آنها را حمایت می‌کند را لیست می‌کند.
- **نگاشت خط‌مشی‌ها**: فقط در گواهی‌نامه‌هائی که برای یک CA از سوی CA دیگر صادر می‌شود کاربرد دارد. این الحاقیه اجازه می‌دهد تا یک CA نشان دهد که یک یا چند مورد از خط‌مشی‌های آن می‌تواند معادل خط‌مشی دیگر در حوزهٔ CA سوژه باشد..

مشخصات سوژه و صادرکنندهٔ گواهی‌نامه

این الحاقیه‌ها از نام‌های مختلف با فرمت‌های مختلف برای سوژهٔ یک گواهی‌نامه یا صادرکنندهٔ یک گواهی‌نامه حمایت کرده و می‌توانند اطلاعات بیشتری در مورد سوژهٔ گواهی‌نامه را انتقال داده تا استفاده‌کننده از گواهی‌نامه اعتماد بیشتری نسبت به یک شخص یا واحد خاص پیدا کند. نمونه‌هائی از این اطلاعات، آدرس پستی، مسئولیت سازمانی فرد و یا تصویر اوست.

میدان‌های الحاقی در این مورد چنین‌اند:

- **نام‌های دیگر سوژه:** شامل یک یا چندنام جایگزین است که می‌توانند فرم‌های مختلفی داشته باشند. این میدان برای حمایت از کاربردهای مختلفی مثل پست الکترونیک، EDI، و IPSec که ممکن است فرم نام‌هایشان متفاوت باشند اهمیت دارد.
- **نام‌های دیگر صادرکننده:** شامل یک یا چند نام جایگزین است که می‌توانند فرم‌های مختلفی داشته باشند.
- **مشخصات سوژه در فهرست راهنما:** اندازه‌های مطلوب مشخصه‌های فهرست راهنمای X.500 که مربوط به سوژهٔ این گواهی‌نامه است را نشان می‌دهد.

محدودیت‌های رَوَند گواهی کردن

این الحاقیه‌ها اجازه می‌دهند تا انواع قیود در گواهی‌نامه‌های صادرشده از طرف یک CA برای CA دیگر درج گردد. این محدودیت‌ها ممکن است نوع گواهی‌هائی را که می‌تواند از طرف CA سوژه صادر گردد و یا آنچه را که متعاقباً در زنجیرهٔ گواهی‌ها رخ میدهد مقید سازد.

میدان‌های الحاقی در این مورد چنین‌اند:

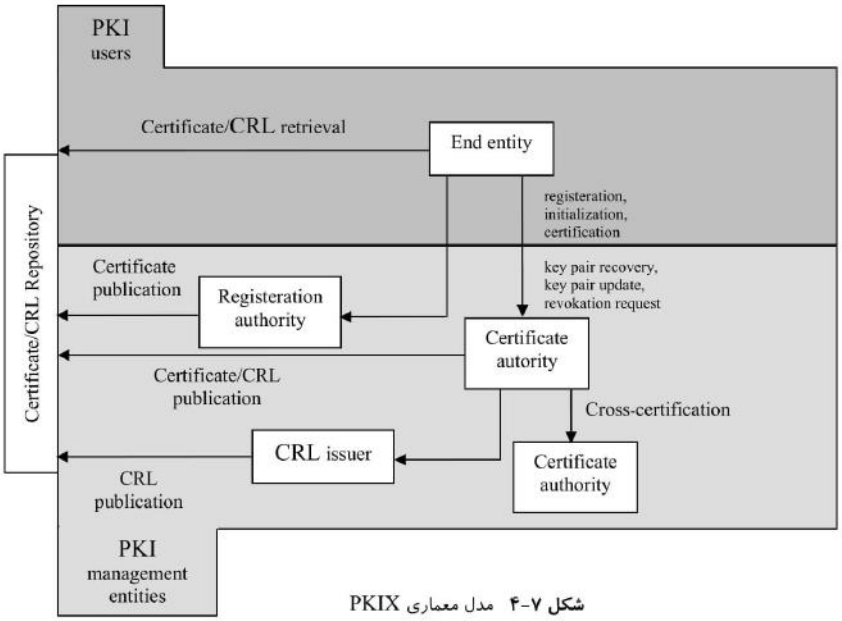
- **قیود اصلی:** نشان می‌دهد که آیا سوژه می‌تواند بصورت یک CA عمل کند. اگر چنین است، ممکن است محدودیتی نسبت به طول مسیر گواهی کردن تعیین گردد.
- **قیود نام‌گذاری:** نشان‌دهندهٔ فضای نام است که در آن نام تمام سوژه‌ها، در گواهی‌های آتی یک مسیر گواهی کردن بایستی جای گیرد.
- **قیود خط‌مشی‌ها:** مشخص‌کنندهٔ قیودی است که ممکن است نیاز به تعیین صریح خط‌مشی‌ها داشته و یا مانع نگاشت خط‌مشی در مابقی مسیر گواهی کردن شوند.

۴-۳ زیرساخت کلید- عمومی (PKI)

RFC 2822 (*فرهنگ لغات امنیتی اینترنت*)، زیرساخت کلید- عمومی (Public-Key Infrastructure) را بصورت مجموعه‌ای از سخت‌افزارها، نرم‌افزارها، آدم‌ها، سیاست‌ها و رَویه‌های لازم برای خلق، مدیریت، نگهداری، توزیع و ابطال گواهی‌نامه‌های دیجیتال بر مبنای رمزنگاری غیرمتقارن تعریف می‌کند. هدف اصلی برپائی یک PKI این است که یک روش امن، سهل و بهره‌ور برای بدست آوردن کلیدهای عمومی ایجاد گردد. گروه کاری زیرساخت کلید- عمومی X.509 در IETF، نیروی محرک ایجاد یک مدل رسمی (و عمومی) مبتنی بر گواهی‌نامهٔ X.509 بوده است تا ساختاری مبتنی بر گواهی‌نامه در اینترنت را ایجاد نماید.

شکل ۷-۴ رابطهٔ درونی عناصر کلیدی مدل PKIX را نشان داده است. این عناصر به قرار زیراند:

- **واحد انتهائی(End entity)**: یک اصطلاح رسمی برای مشخص کردن کاربران انتهائی، دستگاه‌ها (مثل سِرور و مسیریاب) و یا هر موجودیت دیگری که بتواند در مقولهٔ یک گواهی‌نامهٔ کلید- عمومی یافت شود.
- **مسئول گواهی‌نامه‌ها (CA)**: صادرکنندهٔ گواهی‌نامه‌ها و (معمولاً) لیست گواهی‌نامه‌های ابطال شده (CRL). این مقام همچنین ممکن است وظایف مدیریتی دیگری نیز انجام دهد، اگرچه اغلب این وظایف به یک یا چند مسئول ثبت‌نام (Registeration Authority) تفویض می‌شود.



شکل ۷-۴ مدل معماری PKIX

- **مسئول ثبت‌نام (RA)**: یک مؤلفهٔ اختیاری که می‌تواند مسئولیت بخشی از وظایف مدیریتی CA را بعهده گیرد. RA اغلب مسئول پردازش ثبت‌نام واحدهای انتهائی بوده اما می‌تواند در محدوده‌های دیگری نیز فعالیت نماید.
- **صادرکنندهٔ (CRL)**: یک مؤلفهٔ اختیاری که می‌تواند از جانب CA برای انتشار CRLها مأمور شود.
- **مخزن (Repository)**: یک اصطلاح عام که به هر نوع روش کاری برای ذخیره نمودن گواهی‌نامه‌ها و CRLها اشاره دارد، بطوری که آنها بتوانند از طریق واحدهای انتهائی درخواست گردند.

وظایف مدیریتی PKIX

PKIX شماری از وظایف مدیریتی که حتماً لازم است تا بتوسط پروتکل‌های مدیریتی حمایت شوند را مشخص می‌سازد. اینها در شکل ۷-۴ نشان داده شده‌اند و شامل اقلام زیراند:

- **ثبت‌نام (Registeration)**: این عملی است که در آن ابتداءً یک کاربر خود را به CA می‌شناساند (مستقیماً و یا از طریق یک RA). این عمل بایستی قبل از اینکه CA یک گواهی‌نامه و یا گواهی‌نامه‌هائی را برای آن کاربر صادر کند، انجام شود. ثبت‌نام آغاز مرحلهٔ عضوشدن در یک PKI است. ثبت‌نام معمولاً شامل یک سری رَویه‌های برخط (on-line) یا برون خط(off-line) برای احراز هویت متقابل است. معمولاً برای واحد انتهائی یک یا چند کلید سرّی مشترک که در اعتبارسنجی‌های آتی بکار خواهد رفت صادر می‌شود.

- **آغازیدن(Initialization)**: قبل از اینکه یک سیستم کلاینت بتواند بصورت امن عمل نماید، لازم است تا اقلام کلید،که رابطهٔ مناسبی با کلیدهای ذخیره شده در نواحی دیگر زیرساخت دارند، در آن نصب گردند. بعنوان مثال لازم است که کلاینت را با کلید عمومی و سایر اطلاعات مورد نیاز CA مورد اعتماد تجهیز کرد تا بتواند در تأئید مسیر گواهی‌ها از آنها استفاده کند.

- **صدورگواهی‌نامه (Certification)**: این مرحله‌ای است که در آن CA یک گواهی‌نامه برای کلید عمومی یک کاربر صادر نموده و این گواهی‌نامه را به سیستم کلاینت کاربر برگردانده و/ یا آن را در یک مخزن نگاه می‌دارد.

- **بازیابی جفت کلید (Key pair recovery)**: جفت کلیدها می‌توانند برای حمایت از خلق و یا تأئید امضاء دیجیتال و رمزنگاری/رمزگشائی بکار روند. وقتی یک جفت کلید برای رمزنگاری/رمزگشائی بکار می‌رود مهم است، تا برای زمانی که دیگر دسترسی به اقلام کلید میسر نیست، مکانیسمی برای بازیابی کلیدهای رمزگشائی لازم ایجاد کرد. در غیر اینصورت امکان نخواهد داشت که داده‌های رمزنگاری شده را بازیابی نمود. عدم امکان دسترسی به کلید رمزگشائی می‌تواند بعلت فراموش کردن کلمات عبور/PINها، خراب شدن دیسک‌ها، صدمه یافتن ژتون‌های سخت‌افزاری و غیره رخ دهد. بازیابی جفت کلید به واحدهای انتهائی اجازه می‌دهد که جفت کلیدهای رمزنگاری/رمزگشائی خود را از یک تسهیلات مسئول پشتیبانی کلید بازیابی نمایند (معمولاً CA صادرکنندهٔ گواهی‌نامهٔ واحد انتهائی مسئول این کار خواهد بود).

- **به‌روزرسانی جفت کلید (Key pair update)**: تمام جفت کلیدها لازم است تا بطور منظم به‌روزرسانی شوند (یعنی با یک جفت کلید جدید تعویض گردند) و گواهی‌نامه‌های جدیدی صادر گردد. به‌روزرسانی وقتی لازم است که طول عمر گواهی‌نامه تمام شده و یا گواهی‌نامه باطل شود.

- **درخواست ابطال (Revokation request)**: یک فرد مسئول به یک CA اطلاع می‌دهد که به دلیل شرایط غیرنرمال بوجود آمده ابطال یک گواهی‌نامه ضروری است. دلایل ابطال می‌تواند لورفتن کلید خصوصی، تغییر در روش پذیرش و یا تغییر نام باشد.

- **صدور گواهی‌نامه‌های تقاطعی (Cross certification)**: دو CA اطلاعاتی را مبادله می‌کنند که برای صدور یک گواهی‌نامهٔ تقاطعی بکار می‌رود. یک گواهی‌نامهٔ تقاطعی، گواهی‌نامه‌ای است که از سوی یک CA برای CA دیگر صادر می‌شود و شامل یک کلید امضاء CA است که برای صدور گواهی‌نامه بکار می‌رود.

پروتکل‌های مدیریتی PKIX

گروه کاری PKIX دو پروتکل مدیریتی انتخابی متفاوت بین دو واحد PKIX تعریف کرده است که وظایف مدیریتی لیست شده در بخش قبل را بعهده دارند. RFC 2510 پروتکل‌های مدیریت گواهی‌نامه certificate management protocol (CMP) را تعریف می‌کند. در CMP، هریک از وظایف مدیریتی بطور صریح بتوسط مبادله‌های پروتکلی مشخص تعریف شده است. CMP طوری طراحی شده است تا یک پروتکل انعطاف‌پذیر بوده و بتواند نیازهای مدل‌های متنوع فنی، عملیاتی و تجاری را برآورده سازد. RFC 2797 پیام‌های مدیریتی گواهی‌نامه‌ها روی CMS (CMC) را تعریف می‌کند که CMS به RFC 2797 که ساختار پیام‌های رمزی است، اشاره دارد. CMC بر مبنای کارهای ابتدائی‌تر ساخته شده و بمنظور اعمال به پیاده‌سازی‌های موجود بکار می‌رود. در CMC اگرچه تمام عملیات PKIX مورد حمایت‌اند ولی تمام وظایف به مبادلات پروتکلی مشخص نگاشت نمی‌شوند.

۴-۴ منابع مطالعاتی

یک راه بدون دردسر برای درک مفاهیم Kerberos رجوع به [BRYA88] است. یکی از بهترین مراجع Kerberos [KOHL94] می‌باشد. Kerberos [TUNG99] را از نقطهٔ نظر یک کاربر توصیف کرده است.

[PERL99] مدل‌های مختلف اعتماد که می‌توانند در PKI مورد استفاده قرار گیرند را بررسی می‌کند. [GUTM02] مشکلات استفاده از PKI را مورد توجه قرار داده و پیشنهاداتی برای ایجاد یک PKI مؤثر را ارائه می‌دهد.

BRYA88 Bryant,W. *Designing an Authentication System: A Dialogue in Four Scenes.* Project Athena document,February 1988. Available at **http://web.mit.edu/kerberos/www/dialogue.html**.

GUTM02 Gutmann, P."PKI: It`s Not Dead, Just Resting." *Computer*, August 2002.

KOHL94 Kohl,J.; Neuman, B.; and Ts'o,T." The Evolution of the Kerberos Authentication Service." in Brazier, F.,and Johansen,D. *Distributed Open Systems*. Los Alamitos,CA:IEEE Computer Society Press,1994. Available at **http://web.mit.edu/kerberos/www/papers.html**.

PERL99 Perlman, R. "An Overview of PKI Trust Models." *IEEE Network*, November/December 1999.

TUNG99 Tung,B. *Kerberos: A network Authentication System.* Reading, MA: Addison-Wesley,1999.

وب سایت‌های مفید

- **MIT Kerberos Site**: اطلاعاتی در بارهٔ Kerberos که شامل FAQ، مقاله‌ها و اسناد و اشاره به سایت‌های محصولات تجاری است.

- **USC/ISI Kerberos Page**؛ یک منبع خوب دیگر از موارد مربوط به Kerberos.
- **Kerberos Working Group**؛ استانداردهای در حال توسعه مبتنی بر Kerberos گروه IETF.
- **Public-Key Infrastructure Working Group**؛ استانداردهای در حال توسعهٔ مبتنی بر X.509v3 گروه IETF.
- **Verisign**؛ یک فروشندهٔ تجاری پیشتاز محصولات مرتبط با X.509 . مقالات و سایر مطالب ارزنده در این سایت.
- **NIST PKI Program**؛ یک منبع اطلاعاتی خوب.

۴-۵ واژه‌های کلیدی، سؤالات مرورکنندهٔ بحث و مسائل

واژه‌های کلیدی

گواهی‌نامهٔ کلید- عمومی	public – key certificate	اعتبارسنجی	authentication
قلمرو	realm	سِرور اعتبارسنجی	authentication server (AS)
شمارهٔ ردیف	sequence number	یک پروتکل اعتبارسنجی معروف	Kerberos
زیرکلید	subkey	قلمرو Kerberos	Kerberos realm
بلیت	ticket	طول عمر	lifetime
سِرور اعطاکنندهٔ بلیت	ticket–granting server (TGS)	حال فعلی	nonce
گواهی‌نامهٔ X.509	X.509 certificate.	مُود زنجیره‌ای رمز قالبی انتشاریابنده	propagating cipher block chaining (PCBC) mode

سوالات مرورکننده بحث

۴-۱ Kerberos برای پاسخ‌گوئی به چه مشکلی طراحی گردید؟

۴-۲ سه تهدیدی که مرتبط با اعتبارسنجی کاربر روی یک شبکه و یا اینترنت است، کدامند؟

۴-۳ سه روش که می‌تواند اعتبارسنجی کاربر در یک محیط توزیع شده را امنیت بخشد، کدامند؟

۴-۴ چهار نیازی که برای Kerberos تعریف شده است، کدامند؟

۴-۵ کدام اقلام، یک محیط full- service Kerberos را تشکیل می‌دهند؟

۴-۶ در بستر Kerberos، یک قلمرو چیست؟

۴-۷ تفاوت‌های عمدهٔ نسخه‌های چهارم و پنجم Kerberos کدامند؟

۴-۸ هدف استاندارد X.509 چیست؟

۴-۹ یک زنجیره گواهی‌ها چیست؟

۴-۱۰ چگونه یک گواهی‌نامهٔ X.509 باطل می‌شود؟

مسائل

۴-۱ نشان دهید که در مُود PCBC، یک خطای تصادفی در یک بلوک متن رمزنگاری شده به تمام بلوک‌های بعدی متن ساده گسترش می‌یابد(شکل ۹-۴).

۴-۲ فرض کنید که در مُود PCBC، بلوک‌های C_i و C_{i+1} در هنگام انتقال باهم عوض می‌شوند. نشان دهید که این امر فقط بلوک‌های رمزگشائی‌شده P_i و P_{i+1} را تحت تأثیر قرار داده و به بلوک‌های دیگر کاری ندارد.

۴-۳ روش اعتبارسنجی اولیهٔ X.509 نشان داده شده در شکل ۶-۴ج دارای یک نقص امنیتی است. جوهر پروتکل چنین است:

$$\begin{aligned} &A \rightarrow B: && A\{t_A, r_A, ID_B\} \\ &B \rightarrow A: && B\{t_B, r_B, ID_A, r_A\} \\ &A \rightarrow B: && A\{r_B\} \end{aligned}$$

متن X.509 چنین بیان می‌کند که برچسب‌های زمانی t_A و t_B برای اعتبارسنجی سه- سویه اختیاری هستند. اما مثال زیر را در نظر بگیرید: فرض کنید که A و B پروتکل قبل را در موردی قبلاً بکار گرفته و دشمن C سه پیام قبلی را شنود کرده است. علاوه برآن فرض کنید که از برچسب‌ها استفاده نشده و همهٔ آنها مساوی 0 قرارداده شده‌اند. بالاخره فرض کنید که C علاقه‌مند است تا نزد B خود را بجای A جا بزند. C در ابتدا اولین پیام دزدیده شده را برای B می‌فرستد:

$$C \rightarrow B: \quad A\{0, r_A, ID_B\}$$

B در حالی که فکر می‌کند با A صحبت می‌کند، جواب C را چنین می‌دهد:

$$B \rightarrow C: \quad B\{0, r'_B, ID_A, r_A\}$$

C در همین حال بنحوی A را وامیدارد تا به احراز هویت C بپردازد. در نتیجه A پیام زیر را برای C می‌فرستد:

$$A \rightarrow C: \quad A\{0, r'_A, ID_C\}$$

C به A پاسخ می‌دهد و از همان nonce که توسط B برای C تهیه شده بود استفاده می‌کند.

$$C \rightarrow A: \quad C\{0, r'_B, A, r'_A\}$$

A چنین پاسخ می‌دهد:

$$A \rightarrow C: \quad A\{r'_B\}$$

این همان چیزی است که C لازم دارد تا B را متقاعد سازد که دارد با A صحبت می‌کند. و بنابراین C اکنون پیام ورودی را برای B پس می‌فرستد:

$$C \rightarrow B: \quad A\{r'_B\}$$

بنابراین B باور خواهد کرد که دارد با A صحبت می‌کند در حالی که در واقع دارد با C صحبت می‌کند. یک راه حل ساده برای این مشکل پیدا کنید که نیازی به استفاده از برچسب‌های زمانی نداشته باشد.

۴-۴ X.509 در نسخهٔ سال ۱۹۹۸، خواصی که باید کلیدهای RSA داشته تا امن باشند، بر اساس معلومات فعلی در مورد سخت بودن به فاکتور درآوردن اعداد بزرگ، را درج می‌کند. این بحث با قرار دادن یک محدودیت در نمای عمومی و مدول n چنین پایان می‌پذیرد:

بایستی اطمینان یافت که $e > \log_2(n)$ است تا بتوان از حملات با گرفتن ریشهٔ e ام $\bmod n$ به منظور افشاکردن متن ساده جلوگیری کرد.

اگرچه این قید صحیح است ولی دلیل ارائه شده برای نیاز به آن ناصحیح است. در این استدلال چه مشکلی نهفته است و استدلال صحیح چیست؟

ضمیمهٔ ۴- الف تکنیک‌های رمزنگاری KERBEROS

Kerberos شامل یک کتابخانهٔ رمزنگاری است که عملیات متنوعی را که به رمزنگاری مربوط می‌شود پشتیبانی می‌کند. اینها در مشخصه‌های نسخهٔ پنجم Kerberos گنجانده شده بودند و معمولاً در پیاده‌سازی‌های تجاری یافت می‌شوند. در فوریهٔ سال ۲۰۰۵، RFCهای 3961 و 3962 انتشار یافتند که موارد اختیاری تکنیک‌های رمزنگاری را توسعه می‌دهند. در این ضمیمه، تکنیک‌های RFC 1510 را شرح می‌دهیم.

تبدیل کلمهٔ عبور- به- کلید

در Kerberos، کلمات عبور محدود به کاراکترهائی هستندکه می‌توانند با فرمت ۷- بیتی ASCII نمایش داده شوند. این کلمه عبور، که دارای طول دلخواهی است، به یک کلید رمزنگاری تبدیل شده که در پایگاه دادهٔ Kerberos ذخیره می‌شود. شکل ۸-۴ روش کار را نشان می‌دهد.

ابتدا دنبالهٔ کاراکترها، s، بصورت یک دنبالهٔ بیت‌ها، b، در می‌آید بطوری که اولین کاراکتر در اولین ۷ بیت، دومین کاراکتر در دومین ۷ بیت و . . . جای داده می‌شوند. این امر را می‌توان چنین نشان داد

$$b[0] = \text{bit } 0 \text{ of } s[0]$$

$$b[6] = \text{bit } 6 \text{ of } s[0]$$
$$b[7] = \text{bit } 0 \text{ of } s[1]$$

$$b[7i + m] = \text{bit } m \text{ of } s[i] \quad 0 \le m \le 6$$

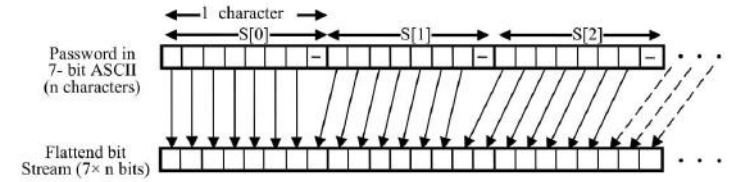
سپس دنبالهٔ بیت‌ها بصورت بادبزنی به یک دنبالهٔ ۵۶- بیتی تبدیل می‌شود. بعنوان مثال اگر دنبالهٔ بیت‌ها دارای طول ۵۹ باشد آنگاه

$$b[55] = b[55] \oplus b[56]$$
$$b[54] = b[54] \oplus b[57]$$
$$b[53] = b[53] \oplus b[58]$$

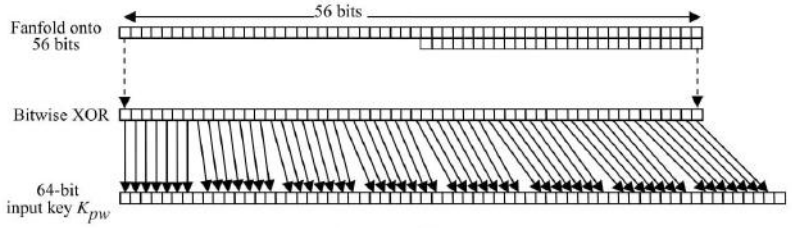
این یک کلید ۵۶- بیتی DES را ایجاد می‌کند. برای اینکه با فرمت مورد انتظار کلید ۶۴- بیتی همخوان باشد، با این دنباله بصورت یک ردیف از بلوک‌های ۷- بیتی رفتار شده که به بلوک‌های ۸ - بیتی نگاشت می‌شود تا کلید ورودی K_{pw} را ایجاد کند.

بالاخره کلمه عبور اولیه با مُود زنجیره‌ای رمز قالبی(CBC) الگوریتم DES با کلید K_{pw} رمزنگاری می‌شود. آخرین بلوک ۶۴- بیتی که از این روش حاصل می‌شود و به نام جمع کنترلی CBC خوانده می‌شود، کلید خروجی مرتبط با این کلمهٔ عبور است.

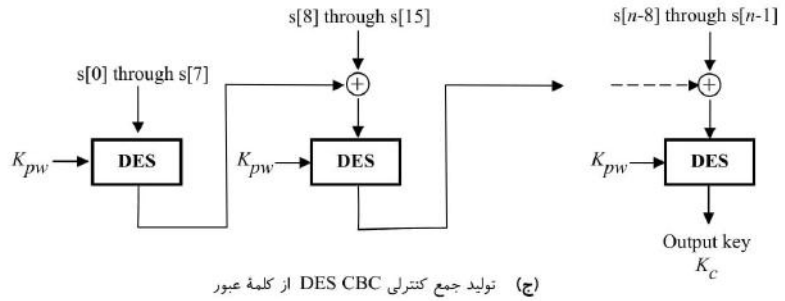
کل الگوریتم را می‌توان تابع درهم‌سازی دانست که یک کلمهٔ عبور دلخواه را به یک کُد hash ۶۴- بیتی تبدیل می‌کند.



(الف) تبدیل کلمهٔ عبور به دنبالهٔ بیت‌ها



(ب) تبدیل دنبالهٔ بیت‌ها به کلید ورودی



(ج) تولید جمع کنترلی DES CBC از کلمهٔ عبور

شکل ۸-۴ تولید کلید رمزنگاری از کلمهٔ عبور

مُود زنجیره‌ای رمزقالبی انتشاریابنده (Propagating Cipher Block Chaining Mode)

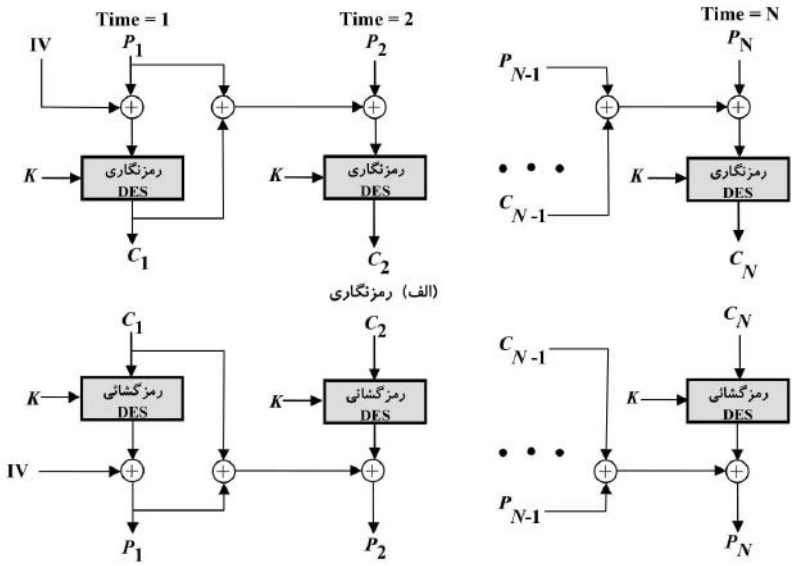
از فصل ۲ بخاطر آورید که در مُود CBC الگوریتم DES، ورودی الگوریتم در هر مرحله شامل XOR بلوک جاری متن ساده با بلوک رمزشده مرحلهٔ قبل بود که برای هر بلوک نیز از همان یک کلید استفاده می‌شد (شکل ۹-۲). مزیت این مُود نسبت به مُود کتاب لغت الکترونیکی (ECB) که در آن هر بلوک بصورت مستقل رمزنگاری می‌شود این است: در CBC اگر یک بلوک متن ساده در جای دیگری تکرار شود بلوک‌های رمزنگاری شدهٔ متفاوتی تولید خواهد شد.

CBC دارای این خاصیت است که اگر در انتقال بلوک رمزشدهٔ C_I خطائی رخ دهد این خطا به بلوک‌های رمزگشائی شدهٔ P_I و P_{I+1} گسترش می‌یابد.

نسخهٔ چهارم Kerberos فرم پیچیده‌تری از CBC را که CBC انتشاریابنده (PCBC) نامیده می‌شود مورد استفاده قرار می‌دهد. این مُود دارای این خاصیت است که هر خطا در یکی از بلوک‌های متن رمزشده به همهٔ بلوک‌های رمزگشائی شدهٔ بعدی گسترش یافته و آنها را بی‌ارزش می‌سازد. بنابراین رمزنگاری و صحت داده‌ها در یک عمل حاصل می‌گردند (برای یک حالت استثناء به مسألهٔ ۲-۴ نگاه کنید).

PCBC در شکل ۹-۴ نشان داده شده است. در این روش ورودی الگوریتم رمزنگاری، XOR بلوک جاری متن ساده، بلوک قبلی متن رمزشده، و بلوک قبلی متن ساده است:

$$C_n = \mathrm{E}\,(K, [C_{n-1} \oplus P_{n-1} \oplus P_n])$$



شکل ۹-۴ مُود زنجیره‌ای رمز قالبی انتشاریابنده (PCBC)

در موقع رمزگشائی، هر بلوک متن رمزشده از الگوریتم رمزگشائی عبور می‌کند. سپس خروجی با بلوک متن رمزشدهٔ قبلی و بلوک متن سادهٔ قبلی XOR می‌شود. با رابطهٔ زیر می‌توان نشان داد که این روش صحیح عمل می‌کند:

$$D(K, C_n) = D(K, E(K, [C_{n-1} \oplus P_{n-1} \oplus P_n]))$$

$$D(K, C_n) = C_{n-1} \oplus P_{n-1} \oplus P_n$$

$$C_{n-1} \oplus P_{n-1} \oplus D[K, C_n] = P_n$$

فصل ۵

امنیت پست الکترونیک

در بین تمام محیط‌های توزیع شده، پست الکترونیک تقریباً پراستفاده‌ترین کاربرد مبتنی بر شبکه است. این سرویس همچنین تنها کاربرد توزیع‌شده‌ای است که در تمام معماری‌ها و سیستم‌های عامل کامپیوتری بطور وسیعی مورد استفاده قرار می‌گیرد. کاربران اینترنت انتظار دارند که بتوانند به همهٔ کسان دیگری که به اینترنت متصل‌اند، صرف‌نظر از سیستم عامل و یا پروتکل ارتباطی مورد استفاده، نامه ارسال نمایند.

با وابستگی روزافزون به پست الکترونیک برای هر مقصود قابل تصور، نیـــاز به سرویس‌های اعتبـــارسنجی و محـــرمانگی مرتباً بیشتر می‌شود. در این زمینه دو روش بطور گسترده‌ای مورد استقبال قرار گرفته و از آنها استفاده می‌شود: Pretty Good Privacy (PGP) و S/MIME. هر دوی آنها در این فصل مورد بررسی قرار می‌گیرند.

۵-۱ PRETTY GOOD PRIVACY

PGP یک پدیدهٔ فوق‌العاده است. PGP با تلاش‌های نسبتاً انفرادی یک نفر بنام Phil Zimmermann، سرویسی است که محرمانگی و اعتبارسنجی را برای پست الکترونیک و همچنین کاربردهای ذخیره‌سازی فایل فراهم می‌آورد. Zimmermann کارهای زیر را انجام داده است:

۱- بهترین الگوریتم‌های موجود رمزنگاری را به عنوان پایه‌های این بنا انتخاب نمود.
۲- این الگوریتم‌ها را طوری در یک کاربرد عام تلفیق کرد که مستقل از سیستم عامل و پردازش‌گر بوده و بر مبنای مجموعهٔ کوچکی از فرامین سهل قرار دارد.
۳- بستهٔ نرم‌افزاری ایجاد شده و اسناد مربوطه که شامل کُد منبع برنامه نیز می‌باشد را از طریق اینترنت، تابلوهای اعلانات و شبکه‌های تجاری همانند AOL (American On Line) بطور مجانی در اختیار کاربران قرار داد.
۴- با یک شرکت (Viacrypt که امروز Network Associates خوانده می‌شود) قراردادی بست که یک نسخهٔ تجاری کاملاً سازگار و ارزان قیمت از PGP را تهیه نماید.

PGP بطور انفجارآمیزی رشدکرده و امروز در سطح گسترده‌ای از آن استفاده می‌شود. برخی از این دلایل این رشد چنین‌اند:

۱- نسخه‌های متعددی از آن که روی کامپیوترهای مختلفی با سیستم عامل‌های متنوع همانندWindows، UNIX، Macintosh و بسیاری دیگر کار می‌کنند بصورت جهانی و مجانی در دسترس است. علاوه براین، نسخهٔ تجاری آن کاربرانی را که به دنبال محصولی با خدمات پشتیبانی بعدی هستند راضی نموده است.
۲- بر مبنای الگوریتم‌هائی قرار دارد که بارها و بارها در آنها تجدید نظر شده و بسیار امن تلقی می‌شوند. علی‌الخصوص بستهٔ نرم‌افزاری شامل RSA، DSS و Diffie-Hellman در حوزهٔ رمزنگاری کلید- عمومی، CAST-128، IDEA و 3DES در حوزهٔ رمزنگاری متقارن و SHA-1 برای درهم‌سازی می‌باشد.

۳- دارای کاربردهای بسیار متنوعی است که از سازمان‌هائی که علاقه‌مند به انتخاب و اجرای یک روش استاندارد برای رمزکردن فایل‌ها و پیام‌ها می‌باشند شروع شده و به اشخاص حقیقی که علاقه‌مند به ارتباط امن روی اینترنت و سایر شبکه‌ها در سطح جهان می‌باشند ختم می‌گردد.

۴- نه بتوسط یک دولت و یا یک سازمان استانداردسازی تولید شده است و نه بتوسط چنین کسانی کنترل می‌شود. برای کسانی که ذاتاً اعتمادی به «تشکیلات» ندارند، این خاصیت PGP پرجاذبه است.

۵- PGP اگرچه امروز روی خط استانداردهای اینترنت قرار گرفته است(RFC 3156)، ولی با وجود این هنوز دارای فضای معطر ضدتشکیلاتی خود است.

بحث را با نگاهی کلی به عملیات PGP آغاز می‌کنیم. در قسمت بعد چگونگی خلق کلیدهای رمزنگاری و ذخیره کردن آنها را بررسی می‌نمائیم. سپس مقولهٔ بسیار مهم مدیریت کلید- عمومی را مورد توجه قرار می‌دهیم.

علائم اختصاری

بسیاری از علائم اختصاری بکار رفته در این فصل را قبلاً نیز مورد استفاده قرار داده‌ایم ولی تعدادی از آنها جدید می‌باشند. شاید بهتر باشد که این علائم اختصاری را در ابتدا خلاصه کنیم. نشانه‌های زیر بکار گرفته شده‌اند:

K_S = کلید اجلاس که در روش رمزنگاری متقارن از آن استفاده می‌شود.

PR_a = کلید خصوصی کاربر A که در روش رمزنگاری کلید- عمومی از آن استفاده می‌شود.

PU_a = کلید عمومی کاربر A که در روش رمزنگاری کلید- عمومی از آن استفاده می‌شود.

EP = رمزنگاری کلید- عمومی

DP = رمزگشائی کلید- عمومی

EC = رمزنگاری متقارن

DC = رمزگشائی متقارن

H = تابع درهم‌ساز (hash)

|| = جمع رشته‌ای

Z = فشرده‌سازی با الگوریتم ZIP

R64 = تبدیل به فرمت radix-64 ASCII

اسناد PGP اغلب از اصطلاح *کلید سرّی* (*secret key*) برای اشاره به کلیدی که در یک روش رمزنگاری کلید- عمومی در کنار کلید عمومی قرار دارد استفاده می‌کنند. همانطور که قبلاً اشاره کردیم، این عمل ممکن است باعث اشتباه شدن این کلید با کلید سرّی مورد استفاده در رمزنگاری متقارن شود. بنابراین ما بجای آن از اصطلاح *کلید خصوصی* (*private key*) استفاده می‌کنیم.

توصیف عملیاتی

عملیات واقعی PGP، صرف نظر از مدیریت کلیدها، شامل پنج سرویس است: اعتبارسنجی، محرمانگی، فشرده‌سازی، سازگاری e-mail و قطعه قطعه کردن دیتا (جدول ۱-۵). هر یک از این پنج سرویس را به نوبت بررسی می‌کنیم.

اعتبارسنجی

شکل ۱-۵الف سرویس امضاء دیجیتال که بتوسط PGP فراهم می‌شود را نشان می‌دهد. این همان امضاء دیجیتال مورد بحث در فصل ۳ و نشان داده شده در شکل ۲-۳ب است. روند کار چنین است:

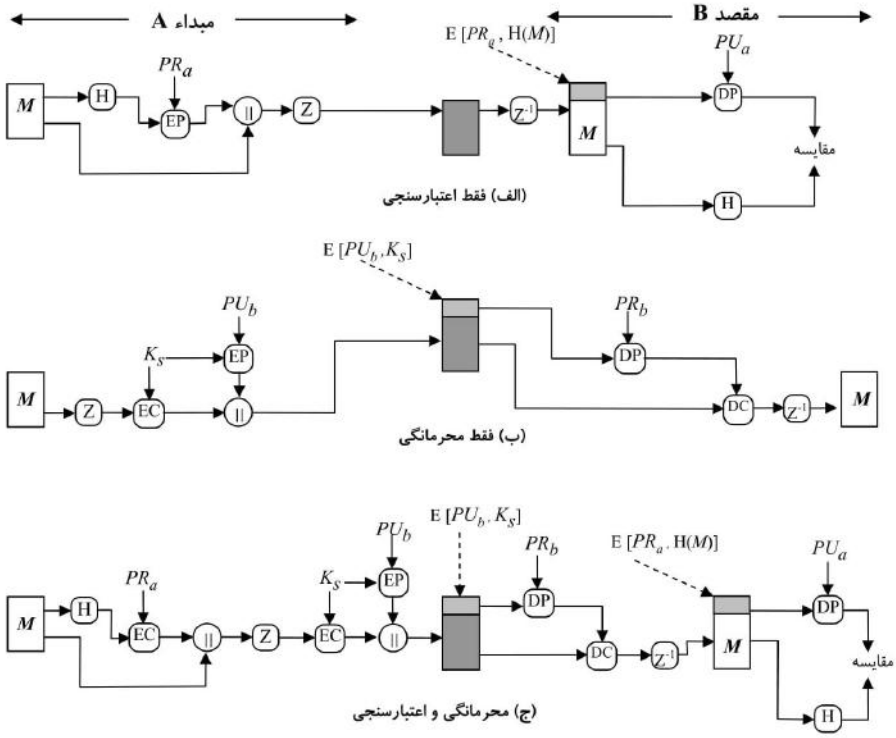
۱- فرستنده پیامی را تهیه می‌کند.
۲- با استفاده از SHA-1 یک کُد hash ۱۶۰- بیتی از پیام ایجاد می‌شود.
۳- کُد hash از طریق RSA رمزنگاری می‌شود که در آن از کلید خصوصی فرستنده استفاده شده است. نتیجه به ابتدای پیام وصل (جمع رشته‌ای) می‌شود.
۴- گیرنده از RSA و کلید عمومی فرستنده استفاده کرده تا کُد hash را رمزگشائی نموده و استخراج کند.
۵- گیرنده یک کُد hash جدید برای پیام تولید کرده و آن را با کُد hash رمزگشائی شده مقایسه می‌کند. اگر دو کُد با هم یکسان باشند، پیام پذیرفته شده و معتبر تلقی می‌گردد.

ترکیب SHA-1 و RSA یک روش مؤثر برای امضاء دیجیتال را فراهم می‌آورد. نظر به قدرت RSA، دریافت‌کننده مطمئن است که تنها صاحب یک کلید خصوصی متناظر، قادر به تولید امضاء بوده است. نظر به قدرت SHA-1، دریافت‌کننده مطمئن است که کس دیگری نمی‌توانسته است پیام جدیدی که کُد hash آن با پیام اصلی و بنابراین با امضاء پیام اصلی یکسان باشد را تولید کند.

در انتخاب دیگر، استفاده از DSS/SHA-1 برای تولید امضاءها مجاز می‌باشد.

جدول ۱- ۵ خلاصهٔ سرویس‌های PGP

عمل	الگوریتم مورد استفاده	توصیف عملیاتی
امضاء دیجیتال	DSS/SHA یا RSA/SHA	با استفاده از SHA-1 یک کُد hash از پیام ساخته می‌شود. این چکیدهٔ پیام با استفاده از DSS یا RSA و بتوسط کلید خصوصی فرستنده رمزنگاری شده و همراه پیام قرار می‌گیرد.
رمزنگاری پیام	CAST یا IDEA یا DES سه کلیدی با Diffie-Hellman یا RSA	یک پیام با استفاده از CAST-128 یا IDEA یا 3DES و بتوسط یک کلید اجلاس یکبار- مصرف تولیدشده در فرستنده رمزنگاری می‌شود. کلید اجلاس با استفاده از Diffie-Hellman یا RSA و بتوسط کلید عمومی گیرنده رمزنگاری می‌شود و همراه پیام قرار می‌گیرد.
فشرده سازی	ZIP	یک پیام ممکن است با استفاده از ZIP، برای ذخیره‌سازی یا انتقال، فشرده گردد.
سازگاری e-mail	تبدیل Radix-64	به منظور ایجاد شفافیت برای کاربردهای e-mail، یک پیام رمزنگاری شده ممکن است با استفاده از تبدیل Radix-64 به یک دنبالهٔ ASCII تبدیل شود.
قطعه قطعه کردن	—	برای حفظ محدودیت‌های طول ماکزیمم پیام، PGP قطعه‌قطعه کردن و دوباره سرهم کردن دیتا را انجام می‌دهد.



شکل ۱-۵ عملیات رمزنگاری PGP

اگرچه امضاءها معمولاً به پیام و یا فایلی که امضاء آنها را تأئید می‌کند وصل‌اند، ولی این امر همیشه صادق نیست. امضاءهای مجزا از پیام نیز پشتیبانی می‌شوند. یک امضاء غیر متصل به پیام نیز می‌تواند بطور جدا از پیام خود در جائی ذخیره شده و انتقال یابد. این امر در زمینه‌هائی مفید واقع می‌شود. یک کاربر ممکن است علاقه‌مند باشد که برای تمام پیام‌های ارسال شده و یا دریافت شده یک کارنامهٔ امضاء جداگانه داشته باشد. یک امضاء غیرمتصل به یک برنامهٔ اجرائی می‌تواند آلودگی‌های ویروسی بعدی را کشف کند. بالاخره امضاءهای غیرمتصل می‌توانند در جائی که بیش از یک طرف امضاءکننده وجود دارد (مثل یک قرارداد قانونی)، مورد استفاده واقع شوند. امضاء هر فرد مستقل بوده و بنابراین تنها سند اصلی را تأئید می‌کند. در غیر این صورت امضاءها بایستی تودرتو باشند و معنی آن این است که امضاءکنندهٔ دوم، تأئیدکنندهٔ هم سند اصلی و هم امضاء امضاءکنندهٔ اول است و همین ترتیب ادامه می‌یابد.

محرمانگی

سرویس اصلی دیگری که بتوسط PGP فراهم می‌آید محرمانگی است که بتوسط رمزنگاری پیام‌هائی که بایستی ارسال شوند و یا بایستی بصورت فایل‌های محلی ذخیره گردند انجام می‌شود. در هر دو مورد، می‌توان از الگوریتم رمزنگاری متقارن

CAST-128 استفاده نمود. راه حل دیگر استفاده از IDEA و یا 3DES است. مُود فیدبک رمز (CFB) ۶۴- بیتی بکار می‌رود.

مثل همیشه، بایستی مشکل توزیع کلید را درنظر داشته باشیم. در PGP، هر کلید متقارن تنها یک‌بار مورد استفاده قرار می‌گیرد. یعنی برای هر پیام، یک کلید جدید بصورت یک عدد تصادفی ۱۲۸- بیتی ایجاد می‌شود. بنابراین اگرچه در سندها این کلید را بنام کلید اجلاس می‌شناسند، ولی در واقع این یک کلید یکبار مصرف است. چون این کلید تنها یک‌بار مورد استفاده قرار می‌گیرد، به پیام متصل شده و همراه آن ارسال می‌گردد. برای حفاظت از این کلید، آن را با کلید عمومی گیرنده رمزنگاری می‌کنیم. شکل ۱-۵ب رَوند عملیات را نشان می‌دهد که می‌توان آن را چنین توصیف نمود:

۱- فرستنده، پیام خود و همچنین یک عدد ۱۲۸- بیتی تصادفی که قرار است بعنوان کلید اجلاس، تنها برای این پیام، بکار رود را تولید می‌کند.
۲- پیام با استفاده از CAST-128 (یا IDEA یا 3DES) رمزنگاری می‌شود.
۳- کلید اجلاس با استفاده از کلید عمومی دریافت‌کننده پیام و RSA رمزنگاری شده و به پیام الصاق می‌گردد.
۴- گیرنده از RSA و کلید خصوصی برای رمزگشائی و بازیابی کلید اجلاس استفاده می‌کند.
۵- کلید اجلاس برای رمزگشائی بکار می‌رود.

بجای RSA برای رمزنگاری کلید، PGP حق انتخاب دیگری بنام *Diffie-Hellman* را فراهم نموده است. همانطور که در فصل ۳ توضیح داده شد، Diffie-Hellman یک الگوریتم مبادلهٔ کلید است. در حقیقت PGP از نوعی Diffie-Hellman که یک نوع رمزنگاری/رمزگشائی بنام ElGamal را فراهم می‌سازد استفاده می‌کند.

چند نکته در این مورد قابل توجه است. اولاً برای کاهش زمان رمزنگاری، ترکیبی از رمزنگاری متقارن و کلید- عمومی بجای استفادهٔ مستقیم از RSA یا ElGamal بکار می‌رود: CAST-128 و سایر الگوریتم‌های متقارن بطور چشمگیرتری سریع‌تر از RSA یا ElGamal هستند. ثانیاً استفاده از الگوریتم‌های کلید- عمومی، مشکل توزیع کلید اجلاس را حل می‌کنند زیرا تنها گیرنده قادر به بازیابی کلید اجلاسی است که به پیام مرتبط شده است. توجه کنید که در اینجا نیازی به یک پروتکل مبادلهٔ کلید اجلاس، از نوعی که در فصل ۳ مورد بحث قرار گرفت، نیست زیرا یک اجلاس جاری را دوباره آغاز نمی‌کنیم. در اینجا هر پیام با کلید مخصوص به خود، یک پیشامد مستقل است که فقط یکبار واقع می‌شود. علاوه براین، با ماهیت store-and-forward پست الکترونیک، استفاده از دستداد(handshaking) برای اطمینان یافتن از اینکه هر دو سمت دارای کلید اجلاس یکسان هستند عملی نمی‌باشد. بالاخره استفاده از کلیدهای متقارن یکبارمصرف روش محکم رمزنگاری متقارن را محکم‌تر می‌کند. با هر کلید، تنها بخش کوچکی از متن ساده رمزشده و هیچ رابطه‌ای بین کلیدها وجود ندارد. بنابراین تا مرزی که الگوریتم کلید عمومی امن است، کل روش امن خواهد بود. تا زمان حاضر، PGP محدوده‌ای از کلیدها بین ۷۶۸ تا ۳,۰۷۲ بیت را برای کاربر فراهم نموده است (کلید DSS برای امضاءهای دیجیتال محدود به ۱,۰۲۴ بیت است).

محرمانگی و اعتبارسنجی

همانطور که شکل ۱-۵ج نشان می‌دهد، هر دو سرویس را می‌توان برای یک پیام واحد بکار برد. ابتدا یک امضاء برای متن سادهٔ پیام تولید شده و به پیام الصاق می‌گردد. آنگاه متن ساده پیام بعلاوهٔ امضاء با استفاده از CAST-128 (یا IDEA یا 3DES) رمزنگاری شده و کلید اجلاس نیز با استفاده از RSA (یا ElGamal) رمز می‌گردد. این دنبالهٔ وقایع به نوع برعکس آن یعنی رمزنگاری پیام و آنگاه تولید یک امضاء برای پیام رمزشده ارجحیت دارد. معمولاً مناسب‌تر است که یک امضاء را به

همراه فرم سادهٔ پیام ذخیره کرد. علاوه بر این برای تأئید شخص ثالث، اگر عمل امضاء در ابتدا صورت پذیرد، شخص ثالث لازم نیست در هنگام تأئید امضاء، نگران کلید رمز متقارن باشد.

خلاصه اینکه وقتی هر دو سرویس مورد استفاده قرار می‌گیرند، فرستنده ابتدا پیام را با کلید خصوصی خود امضاء کرده، سپس آن را با یک کلید اجلاس رمزنگاری نموده و سپس کلید اجلاس را با کلید عمومی گیرنده به رمز درمی‌آورد.

فشرده‌سازی

بصورت پیش‌فرض، PGP پیام را پس از امضاء و قبل از رمزنگاری فشرده می‌نماید. حسن این امر صرفه‌جوئی در فضا هم برای انتقال e-mail و هم برای ذخیره‌سازی فایل است.

نحوهٔ جاسازی الگوریتم فشرده‌سازی، که در شکل ۱-۵ بصورت Z برای فشرده‌سازی و بصورت Z^{-1} برای عکس آن نشان داده شده است، امری مهم است:

۱- به دو دلیل، امضاء قبل از فشرده‌سازی انجام می‌شود:

الف- اصلح است که یک پیام را قبل از فشرده‌سازی امضاء کرد تا بتوان تنها پیام فشرده نشده به همراه امضاء را برای تأئیدهای آتی ذخیره کرد. اگر یک پیام فشرده شده امضاء شده باشد، آنگاه لازم است یا یک نسخهٔ فشرده شدهٔ پیام را ذخیره کرد و یا هروقت لازم باشد برای تأئید، پیام را از حالت فشرده خارج نمود.

ب- حتی اگر راضی باشیم که برای تأئید یک پیام آن را از حالت فشردگی درآوریم، الگوریتم فشرده‌سازی PGP مشکلی را ایجاد می‌کند. الگوریتم یک الگوریتم قطعی نیست و بکارگیری آن با مصالحه‌ای که بین سرعت اجرا و نسبت فشردگی صورت می‌پذیرد، نسخه‌های فشرده شدهٔ مختلفی از پیام را درست می‌کند. با این وجود، این الگوریتم‌های فشرده‌سازی متنوع در بین خود تراکنش داشته زیرا هر نسخهٔ الگوریتم قادر است خروجی هر نسخهٔ دیگر را بطور صحیح بازکند. اِعمال تابع درهم سازی و امضاء بعد از فشرده‌سازی، تمام پیاده‌سازی‌های PGP را به استفاده از یک الگوریتم فشرده‌سازی محدود می‌نماید.

۲- رمزنگاری پیام بعد از فشرده سازی انجام می‌شود تا امنیت رمزنگاری قدرتمندتر گردد. نظر به اینکه پیام فشرده شده دارای افزونگی کمتری نسبت به متن سادهٔ اصلی آن است، کشف رمزآن مشکل‌تر خواهد بود.

الگوریتم فشرده‌سازی مورد استفاده ZIP است که در ضمیمهٔ ۵- الف توصیف گردیده است.

سازگاری E-mail

وقتی PGP مورد استفاده قرار می‌گیرد، حداقل بخشی از بلوکی که باید انتقال یابد رمزنگاری می‌شود. اگر فقط سرویس امضاء بکار گرفته شود، آنگاه چکیدهٔ پیام (digest) رمزنگاری می‌شود (با کلید خصوصی فرستنده). اگر سرویس محرمانگی بکار گرفته شود، آنگاه پیام باضافهٔ امضاء (اگر وجود داشته باشد) رمزنگاری می‌شود (با یک کلید متقارن یکبارمصرف). بنابراین بخشی و یا تمام بلوک نتیجه شده شامل دنباله‌هائی از اُکتت‌های ۸- بیتی اختیاری خواهند بود. اما بسیاری از سیستم‌های پست الکترونیک تنها استفاده از بلوک‌هائی که شامل کُد ASCII باشند را مجاز می‌شمارند. برای همکاری در رفع این محدودیت، PGP سرویسی را فراهم آورده است که دنبالهٔ باینری ۸- بیتی خام را به یک دنبالهٔ قابل چاپ از کاراکترهای ASCII تبدیل می‌کند.

روشی که برای این مقصود بکار می‌رود، تبدیل radix-64 است. هر گروه سه اُکتتی از داده‌های باینری به چهار کاراکتر ASCII تبدیل می‌شوند. این فرمت همچنین یک CRC (Cyclic Redundancy Check) برای تشخیص خطاهای انتقال را به دیتا وصل می‌کند. توصیف این تبدیل در ضمیمهٔ ۵- ب بیان شده است.

استفاده از radix-64، طول یک پیام را بمیزان ۳۳٪ افزایش می‌دهد. خوشبختانه کلید اجلاس و بخش امضاء پیام نسبتاً کوتاه بوده و متن پیام، فشرده شده است. در واقع فشرده‌سازی این بخش بایستی آنقدر باشد که بتواند بر گسترش پیام در تبدیل به radix-64 فایق آید. بعنوان مثال [HELD96] از نسبت فشرده‌سازی متوسطی به میزان ۲ با استفاده از ZIP خبر می‌دهد. اگر از طول نسبتاً کوتاه امضاء و مؤلفه‌های مربوط به کلید صرف‌نظر شود، نتیجهٔ معمول فشرده‌سازی و گسترش یک فایل با طول X برابر X × ۰/۶۶۵ = X × ۰/۵ × ۱/۳۳ خواهد بود. بنابراین رویهم رفته هنوز یک فشردگی بمیزان ۰/۳۳ در پیام ایجاد می‌گردد.

یکی از جنبه‌های قابل توجه الگوریتم radix-64 این است که کورکورانه و بدون توجه به محتوا، دنبالهٔ ورودی را به فرمت radix-64 تبدیل می‌کند، حتی اگر خود دنبالهٔ ورودی متن ASCII باشد. بنابراین اگر پیامی امضاء شده ولی رمزنگاری نشده باشد و تبدیل به همهٔ بلوک اعمال شود، خروجی برای یک ناظر اتفاقی قابل خواندن نخواهد بود و بنابراین خود سطحی از محرمانگی را ایجاد می‌کند. بطور اختیاری، PGP را می‌توان طوری پیکربندی کرد که فرمت radix-64 را تنها به بخش امضاء پیام‌های متنی سادهٔ امضاء شده اعمال نماید. این امر گیرندهٔ انسانی را قادر می‌سازد تا بدون استفاده از PGP پیام را بخواند. در این‌حالت نیز بایستی از PGP برای تأئید امضاء کمک گرفت.

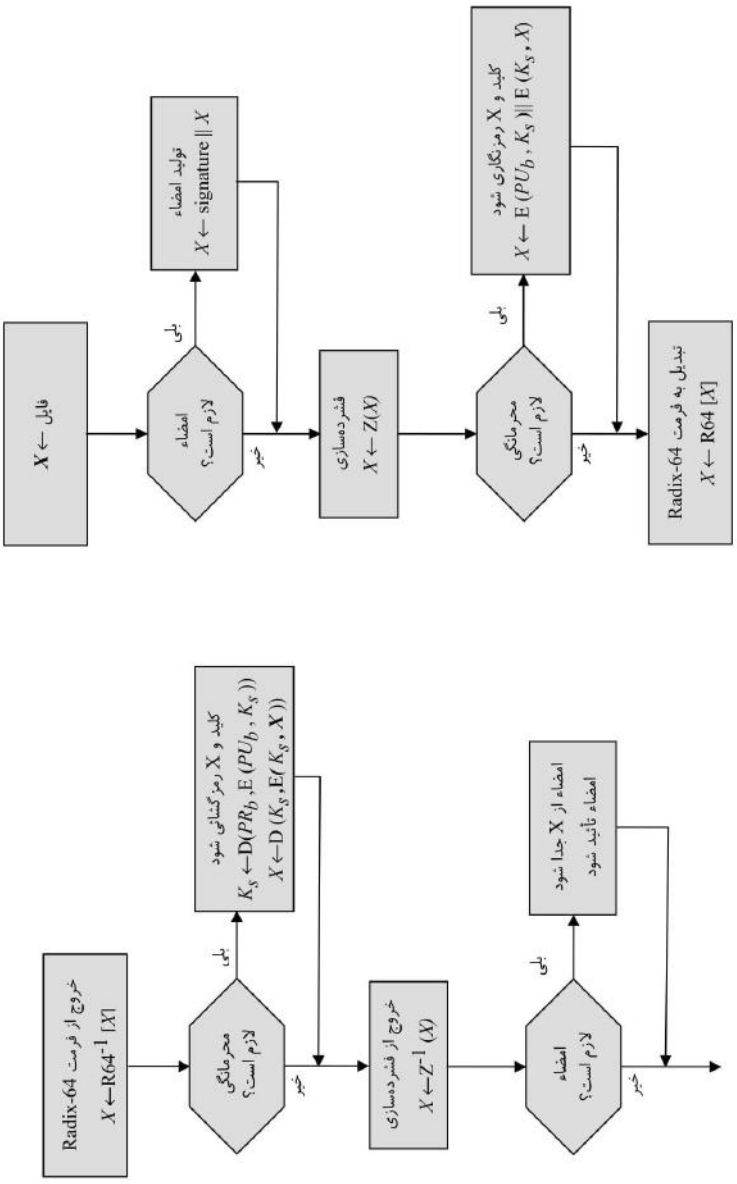
شکل ۲-۵ رابطهٔ بین چهار سرویسی که تا کنون مورد بحث واقع شد را نشان می‌دهد. در هنگام ارسال، اگر لازم باشد، با استفاده از کُد hash متن ساده و فشرده نشده پیام، یک امضاء تولید می‌شود. سپس متن سادهٔ پیام و امضاء (درصورت حضور) فشرده می‌شود. سپس اگر محرمانه‌سازی مورد نیاز باشد، بلوک (صورت فشرده شدهٔ متن ساده و یا صورت فشرده شدهٔ امضاء بعلاوهٔ متن ساده) رمزنگاری شده و در ابتدای آن کلید رمزنگاری متقارن رمزشده بتوسط کلید عمومی، قرار می‌گیرد. بالاخره کل بلوک به فرمت radix-64 در می‌آید.

در زمان دریافت، بلوک ورودی ابتدا از فرمت radix-64 بصورت باینری درمی‌آید. آنگاه اگر پیام رمزنگاری شده باشد، گیرنده کلید اجلاس را بازیابی نموده و پیام را رمزگشائی می‌کند. نتیجهٔ حاصل را باید از حالت فشردگی خارج کرد. اگر پیام امضاء شده باشد، گیرنده کُد hash انتقال یافته را استخراج کرده و آن را با کُد hash ناشی از محاسبات خود مقایسه می‌نماید.

قطعه‌قطعه کردن و دوباره سرهم کردن دیتا

تسهیلات e-mail اغلب از نظر ماکزیمم طول پیام قابل ارسال دارای محدودیت‌اند. بعنوان مثال بسیاری از تسهیلات قابل دسترس از طریق اینترنت، حداکثر طول پیام را ۵۰,۰۰۰ اُکتت منظور کرده‌اند. هر پیامی که طویل‌تر از این مقدار باشد بایستی به قطعات کوچک‌تری تقسیم شده و هر قطعه بطور جداگانه ارسال شود.

برای غلبه بر این محدودیت، PGP پیامی را که بیش از حد طویل است بطور خودکار به قطعاتی که برای ارسال از طریق e-mail باندازهٔ کافی کوچک باشد تقسیم می‌کند. این قطعه‌قطعه کردن پس از همهٔ پردازش‌ها و از جمله تبدیل radix-64 انجام می‌شود. بنابراین عنصر کلید اجلاس و عنصر امضاء تنها یکبار و آنهم در ابتدای اولین قطعه حضور خواهند داشت. در سمت گیرنده، PGP بایستی تمام سرآیندهای e-mail را جدا نموده و قبل از انجام عملیات نشان داده شده در شکل۲-۵ب، تمام بلوک اولیه را سرهم نماید.



شکل ۲ - ۵ ارسال و دریافت پیام‌های PGP

کلیدهای رمزنگاری و دسته‌کلیدها

PGP از چهار نوع کلید استفاده می‌کند: کلیدهای اجلاس که متقارن و یکبار مصرف‌اند، کلیدهای عمومی، کلیدهای خصوصی و کلیدهای متقارن مبتنی بر عبارت عبور (متعاقباً توضیح داده خواهد شد). در رابطه با این کلیدها، سه نیاز جداگانه را می‌توان تشخیص داد:

۱- برای تولید کلیدهای اجلاس غیرقابل پیش‌بینی، روشی مورد نیاز است.

۲- علاقه‌مندیم به یک کاربر اجازه دهیم تا چندین زوج کلید عمومی/کلید خصوصی را در اختیار داشته باشد. یک دلیل آن این است که شاید کاربر علاقه‌مند باشد تا هرچندگاه یکبار کلید خود را عوض کند. وقتی این اتفاق می‌افتد، هر پیامی که در مسیر وجود دارد بتوسط یک کلید خارج از رده ایجاد می‌شود. علاوه براین گیرندگان نیز تا زمانی که کلید جدید به آنها نرسد، تنها کلید قدیم را می‌شناسند. علاوه بر نیاز به تعویض کلید در طول زمان، یک کاربر ممکن است علاقه‌مند باشد تا در هر لحظه زوج کلیدهای متعددی را در اختیار داشته باشد تا با گروه‌های مختلف ارتباط داشته و یا بخواهد با تقسیم رمزنگاری بین کلیدهای مختلف، امنیت را ارتقاء بخشد. نتیجهٔ نهائی بحث این است که بالاخره یک ارتباط یک- به- یک بین کاربران و کلیدهای عمومی آنان وجود ندارد. بنابراین برای شناسائی کلیدهای مختلف، روشی مورد نیاز است.

۳- هر واحد PGP بایستی یک فایل که محتوی زوج‌های کلیدعمومی/خصوصی خود او است و فایل دیگری که محتوی کلیدهای عمومی طرف‌های مقابل است را نگهداری کند.

هر یک از این نیازها را به ترتیب بررسی می‌کنیم.

تولید کلید اجلاس

هر کلید اجلاس مربوط به فقط یک پیام بوده و تنها برای رمزنگاری و رمزگشائی آن پیام بکار می‌رود. بخاطر آورید که رمزنگاری/رمزگشائی پیام بتوسط یک الگوریتم رمزنگاری متقارن انجام می‌شود. CAST-128 و IDEA از کلیدهای ۱۲۸- بیتی استفاده کرده و 3DES از یک کلید ۱۶۸- بیتی استفاده می‌کند. برای این بحث، CAST-128 را در نظر می‌گیریم:

اعداد ۱۲۸- بیتی تصادفی با استفاده از خود CAST-128 تولید می‌شوند. ورودی تولیدکنندهٔ عدد تصادفی شامل یک کلید ۱۲۸- بیتی و دو بلوک ۶۴- بیتی است که دو بلوک اخیر متن ساده‌ای تلقی می‌شود که بایستی رمزنگاری گردد. با استفاده از مُود فیدبک رمز(CFB)، رمزنگار CAST-128 دو بلوک متن رمزشدهٔ ۶۴- بیتی تولید می‌کند که بهم متصل شده تا کلید اجلاس ۱۲۸- بیتی را درست کنند. الگوریتمی که برای این منظور بکار می‌رود، همان است که در ANSI X12.17 ذکر شده است.

«متن ساده» ورودی به تولیدکنندهٔ اعداد تصادفی که شامل دو بلوک ۶۴- بیتی است، خود از یک دنبالهٔ ۱۲۸- بیتی از اعداد تصادفی مشتق می‌شود. این اعداد بر مبنای حرکات کلید کاربر بوجود می‌آیند. هم از زمان استفاده از کلیدها و هم از خود کلیدهای بکار گرفته شده، برای تولید دنبالهٔ تصادفی استفاده می‌شود. بنابراین اگر کاربر کلیدهای صفحهٔ کلید را بطور اختیاری و با سرعت نرمال خود بکار بندد، یک ورودی «تصادفی» معقول تولید خواهد شد. این ورودی تصادفی همچنین با کلید اجلاس قبلی تولیدشده بتوسط CAST-128 ترکیب شده تا کلید ورودی به مولّد را ایجاد نماید. با توجه به خاصیت

مخلوط‌کنندگی مؤثر الگوریتم CAST-128، این عمل ردیفی از کلیدهای اجلاس که بطور چشمگیری غیرقابل پیش‌بینی هستند را بوجود خواهد آورد.

ضمیمهٔ ۵- ج، تکنیک تولید متغیرهای تصادفی در PGP را مورد بحث قرار داده است.

شناسه‌های کلیدها

همانطور که بحث شد، یک پیام رمزنگاری شده با یک فرم رمزشده از کلید اجلاس مورد استفاده همراهی می‌شود. خود کلید اجلاس بتوسط کلید عمومی گیرنده رمزنگاری می‌شود. بنابراین تنها گیرنده قادر به استخراج کلید اجلاس و در نتیجه استخراج پیام خواهد بود. اگر هر کاربر فقط از یک زوج کلید عمومی/خصوصی استفاده می‌کرد، آنگاه گیرنده بطور خودکار می‌دانست که باید از چه کلیدی برای رمزگشائی کلید اجلاس استفاده نماید که همان کلید خصوصی یکتای خود گیرنده است. ولی قبلاً بیان کردیم که لازم است هر کاربر دارای زوج کلیدهای عمومی/خصوصی متعددی باشد.

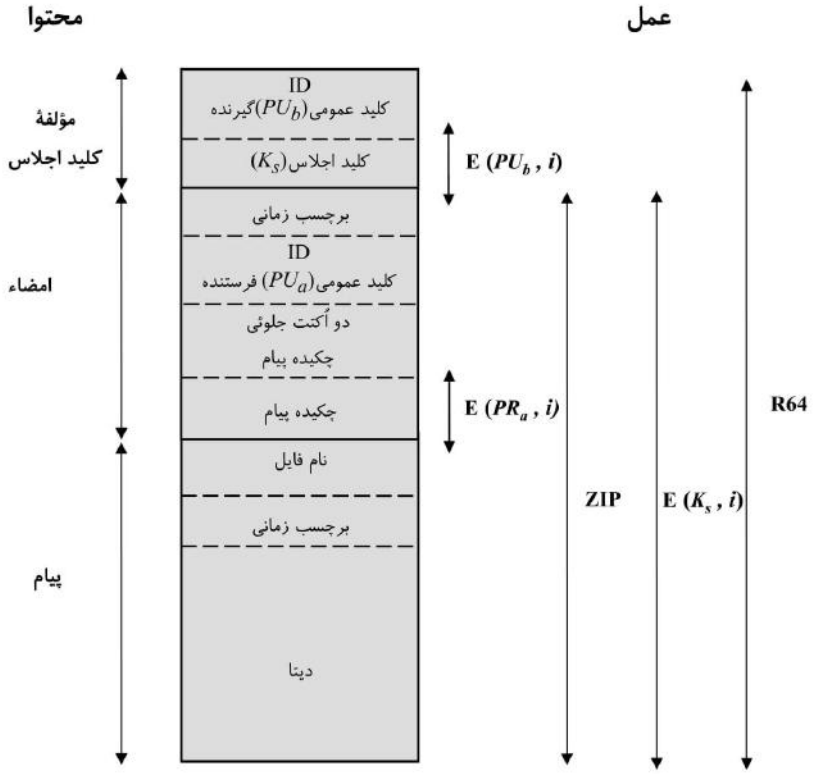
سؤال این است که گیرندهٔ پیام چگونه بداند که از کدام یک از کلیدهای عمومی او برای رمزنگاری کلید اجلاس استفاده شده است؟ یک راه حل ساده این است که کلید عمومی را بهمراه پیام ارسال کرد. دریافت‌کننده در این‌صورت می‌تواند تحقیق کند که این یکی از کلیدهای عمومی او بوده و به مراحل بعد برود. این روش قابل اجرا بوده ولی بی‌جهت فضای انتقال را اشغال می‌کند. یک کلید عمومی RSA ممکن است از صدها رقم اعشاری تشکیل شده باشد. راه حل دیگر این است که با هر کلید عمومی، شناسه‌ای را مرتبط کرد که حداقل در رابطه با یک کاربر یکتا باشد. یعنی ترکیب ID کاربر و ID کلید کافی باشد تا کلید را بصورت یکتا استخراج نمود. در این صورت تنها لازم است که ID مربوط به کلید که خیلی کوتاهتر از خود کلید است ارسال شود. اما این راه حل خود یک مشکل مدیریت و ایجاد سرباره را ایجاد می‌کند: IDهای مربوط به کلیدها بایستی تعیین شده و ذخیره شوند بطوری که هم فرستنده و هم گیرنده بتوانند از روی ID یک کلید به خود کلید عمومی دست یابند. این امر نیز پردردسر بنظر می‌رسد.

راه حلی که بتوسط PGP اتخاذ گردیده است این است که به هر کلید عمومی یک ID کلید تخصیص داده شود که، با احتمال بسیار زیاد، در محدودهٔ ID یک کاربر یکتا باشد. ID کلید مرتبط با هر کلید عمومی کم اهمیت‌ترین ۶۴- بیت آن کلید است. یعنی ID یک کلیدعمومی PU_a برابر $PU_a \bmod 2^{64}$ می‌باشد. این طول برای اینکه احتمال جعل یک ID کلید بسیار کم باشد، طولی معقول است.

یک ID کلید نیز برای امضاء دیجیتال PGP مورد نیاز است. چون یک ارسال‌کننده ممکن است یکی از چند کلید خصوصی را برای رمزنگاری چکیدهٔ پیام بکار برد، دریافت‌کننده بایستی بداند که از کدام کلید عمومی استفاده نماید. بهمین جهت مؤلفهٔ امضاء دیجیتال یک پیام شامل یک ID کلید ۶۴- بیتی مربوط به کلید عمومی مورد نیاز است. وقتی پیام دریافت شد، دریافت‌کننده ابتدا به دنبال تأئید اینکه ID کلید، مربوط به یکی از کلیدهای عمومی شناخته شدهٔ ارسال‌کننده بوده پرداخته، و سپس به دنبال تأئید امضاء می‌رود.

حال که مفهوم ID کلید را معرفی کردیم، می‌توانیم نگاه دقیق‌تری به فرمت یک پیام انتقال یافته که در شکل ۳-۵ نشان داده شده است بیندازیم. یک پیام شامل سه مؤلفه است: مؤلفهٔ پیام، مؤلفهٔ امضاء(اختیاری) و مؤلفهٔ کلید اجلاس (اختیاری).

مؤلفهٔ پیام شامل داده‌های اصلی که بایستی ذخیره و یا ارسال شود بوده و یک نام فایل و یک برچسب زمانی که زمان خلق پیام را تعیین می‌کند نیز با آن همراه است.



علائم اختصاری:

E (PUb , i) = رمزنگاری با کلید عمومی کاربر b

E (PR_a , i) = رمزنگاری با کلید خصوصی کاربر a

E (K_s , i) = رمزنگاری با کلید اجلاس

ZIP = تابع فشرده سازی ZIP

R64 = تابع تبدیل Radix-64

شکل ۳- ۵ فرم عمومی پیام PGP (از A به B)

مولفهٔ امضاء شامل اقلام زیر است:

- **برچسب زمانی:** زمانی را نشان می‌دهد که پیام در آن لحظه امضاء شده است.
- **چکیدهٔ پیام:** چکیدهٔ ۱۶۰- بیتی SHA-1 پیام، که بتوسط کلید خصوصی امضاء ارسال‌کننده رمزنگاری شده است، را نشان می‌دهد. چکیده روی برچسب زمانی مؤلفهٔ امضاء که با بخش دیتای مؤلفهٔ پیام جمع رشته‌ای شده است، محاسبه می‌شود. قراردادن برچسب زمانی مؤلفهٔ امضاء در چکیدهٔ پیام، محافظت در مقابل حمله‌های از نوع بازخوانی را تضمین می‌کند. قرار ندادن نام فایل و برچسب زمانی مؤلفهٔ پیام، این اطمینان را ایجاد می‌کند که امضاءهای

جداشده عیناً شبیه همان امضاءهای اضافه شده به اول پیام است. امضاءهای جداشده در فایلی جداگانه محاسبه می‌شوند که هیچ‌یک از میدان‌های سرآیند پیام را دارا نیستند.

- **دو اُکتت جلوئی چکیدهٔ پیام**: دریافت‌کننده را قادر می‌سازد تا بتواند تعیین کند که آیا کلید عمومی صحیح برای رمزگشائی چکیدهٔ پیام برای اعتبارسنجی بکار رفته است. این عمل با مقایسهٔ کپی متن سادهٔ اولین دو اُکتت، با اولین دو اُکتت رمزگشائی شدهٔ چکیدهٔ پیام انجام می‌شود. این اُکتت‌ها همچنین بعنوان یک FCS (Frame Check Sequence) ۱۶- بیتی برای پیام بکار می‌روند.
- **ID کلید مربوط به کلید عمومی فرستنده**: آن کلید عمومی را نشان می‌دهد که بایستی برای رمزگشائی پیام بکار رود، و بنابراین کلید خصوصی استفاده شده برای رمزنگاری پیام را نیز مشخص می‌کند.

مؤلفهٔ پیام و مؤلفهٔ اختیاری امضاء را می‌توان با استفاده از ZIP فشرده نموده و با یک کلید اجلاس نیز رمزنگاری کرد.

مؤلفهٔ کلید اجلاس شامل کلید اجلاس و شناسهٔ کلید عمومی دریافت‌کننده است که بتوسط فرستنده برای رمزنگاری کلید اجلاس از آن استفاده شده است.

تمام بلوک معمولاً بتوسط کُدینگ radix-64 کُد می‌شود.

دسته‌کلیدها

دیدیم که IDهای کلیدها در عملیات PGP نقش اساسی داشته و دو ID مربوط به دو کلید در هر پیام PGP قرار می‌گیرند تا هم محرمانگی و هم اعتبارسنجی را فراهم‌آورند. این کلیدها لازم است تا بصورت سیستماتیک سازمان‌دهی و ذخیره گردند تا بصورت بهره‌ور و مؤثر بتوسط طرف‌های درگیر مورد استفاده قرار گیرند. روشی که در PGP از آن استفاده می‌شود این است که یک زوج پایگاه داده در هر گره ایجاد شود که یکی از این پایگاه‌ها جفت کلید عمومی/خصوصی متعلق به آن گره را ذخیره کرده و پایگاه دیگر کلیدهای عمومی سایر کاربران شناخته شده برای این گره را نگهداری نماید. این ساختارها را بترتیب دسته‌کلید- خصوصی و دسته‌کلید- عمومی نامند.

شکل ۴-۵ ساختار عمومی یک **دسته‌کلید- خصوصی** را نشان می‌دهد. دسته‌کلید را می‌توان بصورت جدولی در نظر گرفت که در آن هر ردیف نمایشگر یکی از جفت کلیدهای عمومی/خصوصی متعلق به آن کاربر است. هر ردیف شامل مؤلفه‌های زیر است:

- **برچسب زمانی**: تاریخ و زمانی که این جفت کلید تولید شده است.
- **ID کلید**: کم ارزش‌ترین ۶۴- بیت کلید عمومی آن مؤلفه.
- **کلید عمومی**: بخش کلید- عمومی این جفت.
- **کلید خصوصی**: بخش کلید- خصوصی این جفت. این میدان رمزنگاری شده است.
- **ID کاربر**: معمولاً این بخش آدرس e-mail کاربر است (مثل movahed730@yahoo.com). ولی کاربر می‌تواند برای هر جفت کلید، نام متفاوتی را انتخاب نماید (مثل MOV، mov، Mmovahed و غیره) و یا همان ID کاربر را بیش از یکبار تکرار کند.

دسته‌کلید- خصوصی را می‌توان برحسب ID کاربر و یا ID کلید تنظیم کرد. بعداً اهمیت این دو تنظیم را مشاهده خواهیم کرد.

دسته‌کلید- خصوصی

Timestamp	Key ID	Public Key	Encrypted Private Key	User ID
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•
T_i	$PU_i \bmod 2^{64}$	PU_i	$E(H(P_i), PR_i)$	User i
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•

دسته‌کلید- عمومی

Timestamp	Key ID	Public Key	Owner Trust	User ID	Key Legitimacy	Signature(s)	Signiture Trust(s)
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
T_i	$PU_i \bmod 2^{64}$	PU_i	trust_flag $_i$	User i	trust_flag $_i$		
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•

شکل ۴-۵ ساختار عمومی دسته‌کلیدهای- عمومی و خصوصی

اگرچه هدف این است که دسته‌کلید- خصوصی تنها روی ماشین کاربر که این جفت کلیدها را تولید کرده است و صاحب آن است ذخیره شود و تنها در دسترس همان کاربر قرار گیرد، ولی منطقی است که اندازهٔ کلیدهای خصوصی را تا حد ممکن مخفی نگاه داشته و حفاظت کرد. بهمین جهت خود کلید خصوصی در دسته‌کلید ذخیره نمی‌شود. بلکه این کلید با استفاده از CAST-128 (یا IDEA یا 3DES) رمزنگاری می‌گردد. روش عمل چنین است:

۱- کاربر یک عبارت عبور (passphrase) که قرار است برای رمزنگاری کلیدهای خصوصی بکار رود را انتخاب می‌کند.
۲- وقتی سیستم یک جفت کلید عمومی/خصوصی جدید با استفاده از RSA تولید می‌کند، از کاربر نسبت به عبارت عبور سؤال می‌نماید. با استفاده از SHA-1 یک کُد ۱۶۰- بیتی از این عبارت عبور درست شده و خود عبارت عبور معدوم می‌شود.
۳- سیستم، کلید خصوصی را با استفاده از CAST-128 و استفاده از ۱۲۸- بیت کُد hash بعنوان کلید، رمزنگاری می‌نماید. کُد hash متعاقباً معدوم شده و کلید خصوصی رمزشده در دسته‌کلید- خصوصی ذخیره می‌گردد.

متعاقباً وقتی یک کاربر دسته‌کلید- خصوصی را برمی‌دارد تا یک کلید خصوصی را انتخاب کند، او بایستی عبارت عبور را ارائه نماید. PGP کلید خصوصی رمزنگاری شده را بازیابی نموده، کُد hash نظیر عبارت عبور را تولید کرده و کلید خصوصی رمزنگاری شده را با استفاده از CAST-128 و کُد hash رمزگشائی می‌نماید.

این یک روش بسیار ساده و مؤثر است. همانند هر سیستم مبتنی بر کلمهٔ عبور، امنیت این سیستم وابسته به امنیت کلمهٔ عبور است. برای جلوگیری از وسوسهٔ نوشتن کلمهٔ عبور بر روی کاغذ، کاربر از یک عبارت عبور استفاده می‌کند که بسادگی قابل حدس نبوده ولی بسادگی قابل بخاطر سپردن است.

شکل ۴-۵ همچنین ساختار کلی **دسته‌کلید- عمومی** را نشان می‌دهد. این پایگاه داده برای ذخیره کردن کلیدهای عمومی سایر کاربران که نزد این کاربر شناخته شده است بکار می‌رود. فعلاً اجازه دهید بعضی از میدان‌های موجود نشان داده شده در جدول را فراموش کرده و به توصیف میدان‌های زیر بپردازیم:

- **برچسب زمانی**: تاریخ/ زمان خلق این مورد را نشان می‌دهد.
- **ID کلید**: کم اهمیت‌ترین ۶۴- بیت کلید عمومی این مورد.
- **کلید عمومی**: کلید عمومی این مورد.
- **ID کاربر**: صاحب این کلید را مشخص می‌کند. ممکن است چندین ID کاربر مرتبط با یک کلید عمومی منفرد باشند.

دسته‌کلید- عمومی را می‌توان یا بر حسب ID کاربر و یا بر حسب ID کلید رده‌بندی نمود. نیاز به هردو روش را متعاقباً مشاهده خواهیم کرد.

حال در موقعیتی هستیم که نشان دهیم چگونه این دسته‌کلیدها در ارسال و دریافت پیام بکار می‌روند. بمنظور سهولت، از فشرده‌سازی و تبدیل radix-64 در این بحث صرفنظر می‌کنیم. ابتدا ارسال پیام (شکل ۵-۵) را در نظر گرفته و فرض کنید که پیام بایستی هم امضاء شده و هم رمزنگاری شود. PGP ارسال‌کنندهٔ پیام قدم‌های زیر را برمی‌دارد:

۱- امضاء پیام

الف- PGP کلید خصوصی ارسال‌کنندهٔ پیام را از دسته‌کلید- خصوصی او، با استفاده از اندیس your_userid، استخراج می‌کند. اگر your_userid در فرمان وجود نداشته باشد، اولین کلید خصوصی دسته‌کلید انتخاب می‌شود.

ب- PGP عبارت عبور کاربر را از او سؤال کرده تا کلید خصوصی رمزنشده را بازیابی کند.

ج- مؤلفهٔ امضاء پیام ساخته می‌شود.

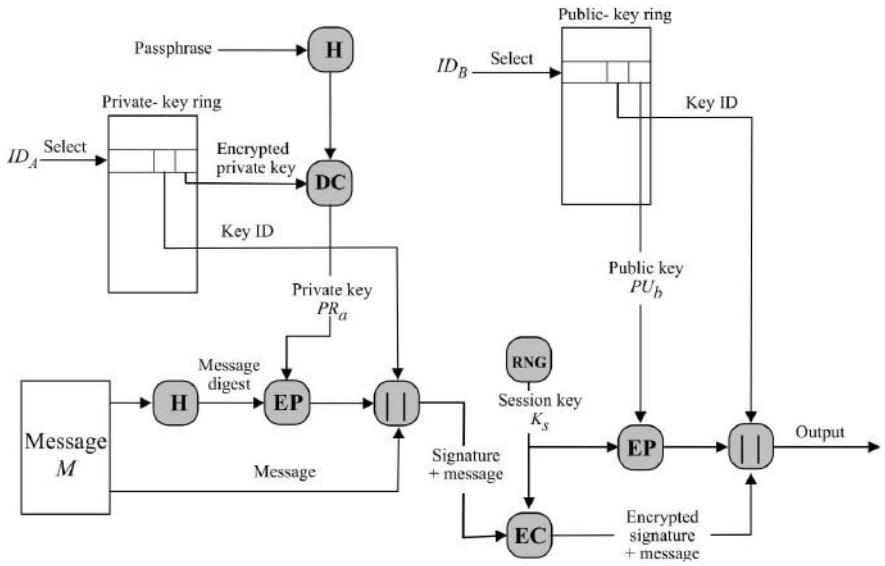
۲- رمزنگاری پیام

الف- PGP یک کلید اجلاس تولید کرده و پیام را رمزنگاری می‌کند.

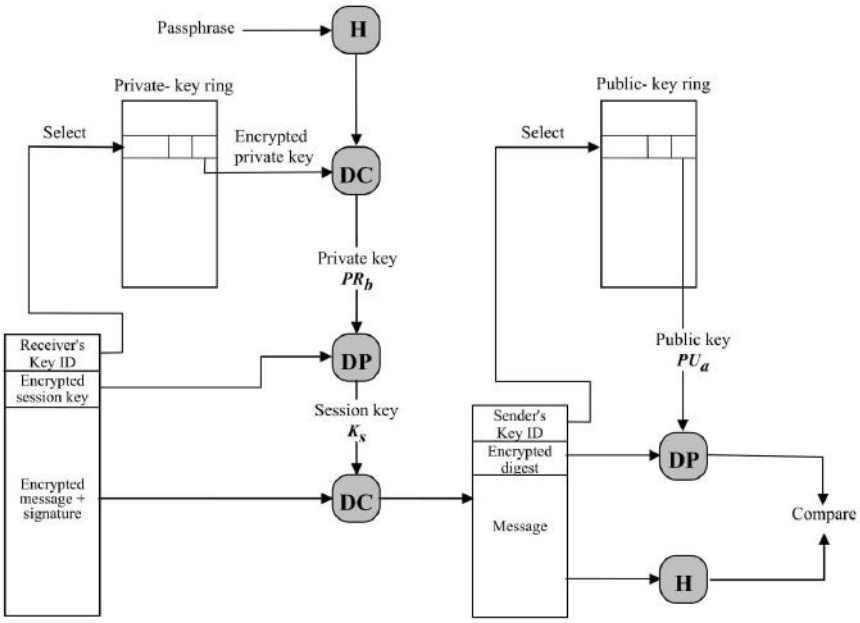
ب- PGP کلید عمومی دریافت‌کنندهٔ پیام را، با استفاده از اندیس her_userid، از دسته‌کلید- عمومی کاربر استخراج می‌کند.

ج- مؤلفهٔ کلید اجلاس پیام ساخته می‌شود.

PGP واحد دریافت‌کنندهٔ پیام قدم‌های زیر را برمی‌دارد (شکل ۶-۵):



شکل ۵-۵ تولید پیام PGP (از کاربر A به کاربر B . بدون فشرده‌سازی و تبدیل Radix-64)



شکل ۶-۵ دریافت پیامPGP (از کاربر A به کاربر B . بدون فشرده‌سازی و تبدیل Radix-64)

۱- رمزگشائی پیام

الف- PGP کلید خصوصی دریافت‌کننده را، با استفاده از میدان ID کلید در مؤلفهٔ امضاء کلید در پیام بعنوان اندیس، از دسته‌کلید- خصوصی استخراج می‌کند.

ب- PGP از کاربر عبارت عبور را سؤال کرده تا کلید خصوصی رمزنگاری نشده را استخراج نماید.

ج- PGP آنگاه کلید اجلاس را بازیابی نموده و پیام را رمزگشائی می‌نماید.

۲- اعتبارسنجی پیام

الف- PGP کلید عمومی ارسال‌کننده را، با استفاده از میدان ID کلید در مؤلفهٔ امضاء کلید در پیام بعنوان اندیس، از دسته‌کلید- عمومی کاربر استخراج می‌کند.

ب- PGP چکیدهٔ پیام انتقال یافته را بازیابی می‌کند.

ج- PGP چکیدهٔ پیام برای پیام دریافت شده را محاسبه کرده و آن را با چکیدهٔ پیام انتقال یافته مقایسه و اعتبار آن را تأئید می‌نماید.

مدیریت کلید- عمومی

همانطور که از بحث‌های انجام شده دیده می‌شود، PGP شامل یک سری عملیات و فرمت‌های هوشیار، بهره‌ور و درهم بافته‌ایست که یک سرویس مؤثر محرمانگی و اعتبارسنجی را ایجاد می‌کند. برای اینکه این سیستم کامل باشد، یک عمل نهائی دیگر نیز بایستی مورد توجه قرار گیرد که آن مدیریت کلید- عمومی است. اسناد PGP اهمیت این مقوله را چنین بیان می‌کند:

> حفاظت کلیدهای عمومی از دست کسانی که می‌خواهند آنها را به چنگ آورند، مشکل‌ترین وظیفه درکاربردهای عملی کلید-عمومی است. این مورد « پاشنهٔ آشیل » رمزنگاری کلید-عمومی بوده و پیچیدگی‌های نرم‌افزاری زیادی با حل این معضل عجین می‌شود.

PGP ساختاری را برای حل این مسأله ایجاد کرده است که انتخاب‌های متعددی را نیز شامل می‌شود. چون PGP بمنظور استفاده در محیط‌های مختلف رسمی و غیررسمی طراحی شده است، هیچ روش مدیریتی سفت و سختی برای مدیریت کلید- عمومی، همانند آنچه که در S/MIME که بعداً در همین فصل به آن خواهیم پرداخت وجود دارد، در نظر نگرفته است.

روش‌های مدیریت کلید- عمومی

اصل مشکل این است: کاربر A بایستی با استفاده از PGP، یک دسته‌کلید- عمومی درست کندکه کلیدهای عمومی سایر کاربران که با او ارتباط دارند در آن وجود داشته باشد. فرض کنید که دسته‌کلید A شامل یک کلید عمومی مربوط به B است که در حقیقت صاحب آن کلید، C است. این امر برای مثال وقتی اتفاق می‌افتد که A کلید را از یک سیستم تابلوی اعلانات (BBS) متعلق به B که کلید عمومی را در آن جا داده بوده است بردارد، ولی کلید بتوسط C تعویض شده باشد. نتیجهٔ امر این است که اکنون دو تهدید وجود دارد. اول اینکه C می‌تواند پیام‌هائی را برای A ارسال کرده و امضاء B را جعل نماید، بطوری که A تصور کند که پیام از طرف B آمده است. دوم اینکه هر پیام رمزشده از A به B می‌تواند بتوسط C خوانده شود.

برای به حداقل رساندن خطر وجود کلیدهای عمومی جعلی در دسته‌کلید یک کاربر، روش‌های متعددی تجربه شده است. فرض کنید A بخواهد یک کلید عمومی قابل اعتماد برای B بدست آورد. برخی از روش‌های پیشنهادی چنین‌اند:

۱- کلید را بصورت فیزیکی از B بگیرد. B می‌تواند کلید عمومی (PU_b) خود را روی یک دیسکت ذخیره کرده و آن را به A بدهد. A می‌تواند بعداً از روی دیسکت کلید را وارد سیستم خود نماید. این روش بسیار امن بوده ولی دارای محدودیت‌های عملی واضحی است.

۲- بتوسط تلفن کلید را تأئید نماید. اگر A بتواند از پشت تلفن B را بشناسد، می‌تواند به B زنگ زده و از وی بخواهد تا کلید را در فرمت radix-64 برای او دیکته نماید. راه حل عملی‌تر اینکه B می‌تواند کلید خود را بتوسط e-mail برای A ارسال کند. A می‌تواند با استفاده از PGP یک چکیدهٔ ۱۶۰- بیتی SHA-1 از کلید تهیه کرده و آن را با فرمت هگزادسیمال نشان دهد که این را «اثرانگشت» کلید گویند. A می‌تواند بعداً به B زنگ زده و از او بخواهد تا اثر انگشت را از پشت تلفن برای او دیکته کند. اگر دو اثر انگشت نزد A و B با هم تطبیق داشته باشند، کلید تأئید می‌شود.

۳- کلید عمومی B را از یک نفر که مورد اعتماد طرفین است، مثل D، دریافت کند. برای این منظور، معرف D یک گواهی‌نامهٔ امضاء شده را فراهم می‌آورد. گواهی‌نامه شامل کلید عمومی B، زمان تولید کلید و مدت اعتبار کلید خواهد بود. D یک چکیده SHA-1 از این گواهی‌نامه تهیه کرده، آن را با کلید خصوصی خود رمزنگاری نموده و امضاء را به گواهی الصاق می‌کند. چون فقط D می‌توانسته است امضاء را تولیدکند، هیچ کس دیگر نمی‌تواند یک کلید عمومی جعلی تهیه کرده و وانمود کند که بتوسط D امضاء شده است. گواهی‌نامهٔ امضاء شده می‌تواند بتوسط B یا D مستقیماً ارسال شده و یا در یک تابلوی اعلانات (BBS) نصب گردد.

۴- کلید عمومی B را از یک مقام مسئول صدور مجوز مورد اعتماد دریافت کند. باز هم یک گواهی‌نامه برای کلید عمومی تولید شده و بتوسط مقام مسئول امضاء می‌شود. A می‌تواند به مقام مسئول دسترسی یافته و با استفاده از ID خود یک گواهی‌نامهٔ امضاء شده را دریافت نماید.

برای موارد ۳ و ۴، A بایستی یک کپی از کلید عمومی معرف را داشته و اطمینان یابد که این کلید معتبر است. بالاخره این بر عهدهٔ A است تا سطحی از اعتماد را نسبت به کسی که بعنوان معرف عمل می‌کند، داشته باشد.

استفاده از اعتماد (Trust)

اگرچه PGP هیچ گونه دستورالعملی برای ایجاد مسئول تأئید و یا برقراری اعتماد ندارد، ولی خود روش‌های مناسبی برای استفاده از اعتماد، ارتباط دادن اعتماد با کلیدهای عمومی و بکارگیری اعتماد را فراهم آورده است.

ساختار اصلی چنین است: هر فقرهٔ موجود در دسته‌کلید- عمومی، همانطور که در بخش قبل توضیح داده شد، یک گواهی‌نامهٔ کلید- عمومی است. به همراه هر فقره گواهی‌نامهٔ کلید- عمومی یک **میدان مشروعیت کلید (key legitimacy field)** وجود دارد که نشان می‌دهد PGP تا چه حدّی نسبت به تعلق این کلید به کاربر مربوطه اعتماد دارد. هر چقدر سطح اعتماد بالاتر باشد، ارتباط کلید با ID کاربر مستحکم‌تر است. این میدان بتوسط PGP محاسبه می‌گردد. همچنین در ارتباط با هر فقره گواهی‌نامهٔ کلید- عمومی، هیچ و یا چند امضاء وجود دارد که صاحب دسته‌کلید آنها را به منظور تأئید این گواهی‌نامه جمع‌آوری کرده است. بنوبهٔ خود در ارتباط با هر امضاء یک **میدان اعتماد به امضاء (signature trust field)** وجود دارد که درجهٔ اعتماد کاربر PGP نسبت به امضاءکننده، برای تأئید کلیدهای عمومی را

نشان می‌دهد. میدان مشروعیت کلید از مجموعهٔ میدان‌های اعتماد به امضاء برای هر گواهی‌نامه مشتق شده است. بالاخره هر فقره گواهی‌نامهٔ کلید- عمومی، یک کلید عمومی مرتبط با یک دارندهٔ مشخص کلید را تعریف کرده و برای آن یک **میدان اعتماد به صاحب کلید (owner trust field)** در نظر گرفته شده است که نشان می‌دهد تا چه حد این کلید عمومی برای امضاء سایر گواهی‌نامه‌های کلید- عمومی مورد اعتماد است. این سطح اعتماد بتوسط کاربر تخصیص داده می‌شود. میدان‌های اعتماد به امضاء را می‌توان کپی‌های ذخیره شده میدان اعتماد به صاحب کلید فقره‌های دیگر دسته‌کلید دانست.

سه میدانی که در بخش قبل به آنها اشاره شد، هریک در ساختاری که به آن بایت پرچم اعتماد (trust flag byte) گویند قرار دارند. محتوای پرچم اعتماد برای هریک از این سه مورد ذکرشده در جدول ۲-۵ نشان داده شده است. فرض کنید که ما با دسته‌کلید- عمومی کاربر A سروکار داریم. عملیات اعتمادسازی را می‌توان چنین توصیف کرد:

۱- وقتی A یک کلید عمومی جدید را در دسته‌کلید- عمومی وارد می‌کند، PGP بایستی اندازه‌ای را به پرچم اعتماد مربوط به صاحب این کلید عمومی تخصیص دهد. اگر صاحب این کلید A است و بنابراین این کلید عمومی در دسته کلید- خصوصی او نیز قرار می‌گیرد، آنگاه یک اندازهٔ *اعتماد کامل* بصورت اتوماتیک به میدان اعتماد اختصاص می‌یابد. در غیر اینصورت PGP از A نسبت به سطح اعتمادی که بایستی به این صاحب کلید تخصیص یابد سؤال می‌کند و A بایستی مقدار مورد نظر خود را وارد کند. کاربر A می‌تواند مشخص نماید که این صاحب کلید ناشناخته، غیرقابل اعتماد، تا حدودی قابل اعتماد و یا کاملاً مورد اعتماد است.

۲- وقتی یک کلید عمومی جدید وارد می‌شود، یک یا چند امضاء ممکن است به آن متصل باشد. امضاءهای دیگری نیز ممکن است در آینده به آن اضافه شوند. وقتی یک امضاء برای یک فقره گواهی‌نامهٔ کلید- عمومی وارد می‌شود، PGP در دسته‌کلید- عمومی جستجو کرده تا ببیند که آیا امضاءکننده در بین صاحبان کلیدهای عمومی شناخته شده هست یا خیر. اگر جواب مثبت باشد، اندازهٔ OWNERTRUST برای این دارندهٔ کلید به میدان SIGTRUST برای این امضاء تخصیص می‌یابد. اگرجواب منفی باشد، مقدار *کاربر ناشناخته* به آن تخصیص می‌یابد.

۳- اندازهٔ میدان مشروعیت کلید بر مبنای میدان‌های اعتماد به امضاء موجود در یک فقره محاسبه می‌گردد. اگر حداقل یک امضاء دارای اندازهٔ *اعتماد کامل* باشد، آنگاه مشروعیت کلید اندازهٔ کامل می‌گیرد. در غیر اینصورت PGP یک جمع تراز داده شده از مقادیر اعتماد را محاسبه خواهد کرد. یک وزن $1/X$ به امضاءهائی که همیشه مورد اعتمادند و یک وزن $1/Y$ به امضاءهائی که معمولاً قابل اعتمادند داده می‌شود. X و Y پارامترهائی هستند که بتوسط کاربر پیکربندی می‌شوند. وقتی جمع ترازهای داده شده معرف‌های یک ترکیب کلید/UserID به ۱ برسد، این پیوند قابل اعتماد تلقی شده و مشروعیت کلید کامل فرض می‌شود. بنابراین در غیاب اعتماد کامل، حداقل X امضاء که همیشه مورد اعتماد بوده و یا Y امضاء که معمولاً قابل اعتمادند و یا ترکیبی از آنها مورد نیاز خواهد بود.

هرچندگاه یکبار، PGP دسته‌کلید- عمومی را مورد پردازش قرار داده تا اقلام آن را با هم سازگار نماید. در واقع این یک پردازش از بالا به پائین است. برای هر میدان OWNERTRUST، PGP دسته‌کلید را برای تمام امضاءهای تأئیدشده بتوسط صاحب آن جستجو کرده و میدان SIGTRUST را بروزرسانی نموده تا معادل میدان OWNERTRUST گردد. این پردازش ابتدا از کلیدهائی شروع می‌شود که برای آنها اعتماد کامل وجود دارد. آنگاه تمام میدان‌های KEYLEGIT بر اساس امضاءهای جداشده محاسبه می‌گردد.

جدول ۲-۵ محتویات بایت پرچم اعتماد (Trust Flag Byte)

(الف) اعتماد تخصیص داده‌شده به صاحب کلیدعمومی (بعد از میدان کلید ظاهر شده و بتوسط کاربر تعریف می‌شود)	(ب) اعتماد تخصیص داده‌شده به زوج USER ID/ Public Key (بعد از میدان User ID ظاهر شده و بتوسط PGP محاسبه می‌شود)	(ج) اعتماد تخصیص داده‌شده به امضاء (بعد از میدان امضاء ظاهرشده وکپی ذخیره‌شده OWNERTRUST برای این امضاءکننده است)
میدان OWNERTRUST - اعتماد تعیین نشده - کاربر ناشناخته - معمولاً برای امضاء کلیدهای دیگر مورد اعتماد نیست - معمولاً برای امضاء کلیدهای دیگر مورد اعتماد است - همیشه برای امضاء کلیدهای دیگر مورداعتماد است - این کلید در دسته کلید خصوصی وجود دارد (اعتماد کامل) بیت BUCKSTOP - این بیت در صورتی set است که این کلید در دسته کلید خصوصی ظاهر شده باشد	میدان KEYLEGIT - اعتماد نامشخص و یا تعیین نشده - مالکیت کلید مورد اعتماد نیست - اعتماد نسبی به مالکیت کلید - اعتماد کامل به مالکیت کلید بیت WARNONLY - این بیت در صورتی set است که کاربر بخواهد در صورت استفاده از یک کلید رمزنگاری که کاملاً معتبر نیست تنها به او هشدار داده شود	میدان SIGTRUST - اعتماد تعیین نشده - کاربر ناشناخته - معمولاً برای امضاء کلیدهای دیگر مورد اعتماد نیست - معمولاً برای امضاء کلیدهای دیگر مورد اعتماد است - همیشه برای امضاء کلیدهای دیگر مورد اعتماد است - این کلید در دسته کلید خصوصی وجود دارد (اعتماد کامل) بیت CONTIG - این بیت در صورتی set است که امضاء به یک مسیربیوسته مورد اعتماد، که نهایتاً به صاحب دسته کلید کاملاً معتمد باز می‌گردد، مربوط شود

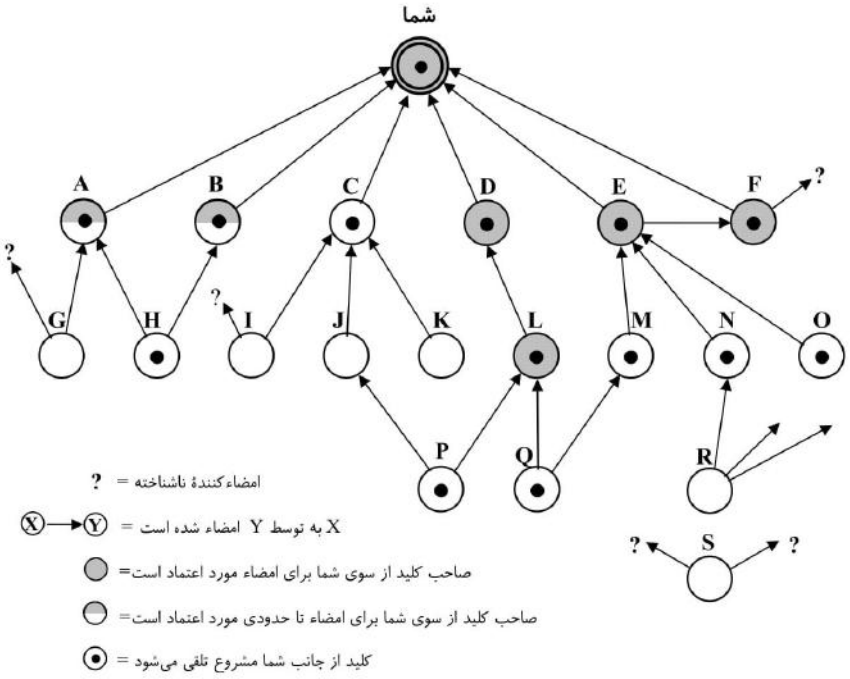
شکل ۷-۵ مثالی از نحوهٔ ارتباط اعتماد به امضاء، به مشروعیت کلید را نشان می‌دهد. در این شکل ساختار یک دسته کلید- عمومی نشان داده شده است. کاربر تعدادی از کلیدهای عمومی را جمع کرده است که برخی از آنها مستقیماً از صاحبان آنها و بعضی دیگر از شخص ثالثی که سِرور کلید است اخذ شده است.

گره‌ای که با عنوان "**شما**" نشان داده شده است به فقره‌ای در دسته‌کلید- عمومی اشاره می‌کند که نظیر این کاربر است. این کلید مشروع بوده و اندازهٔ OWNERTRUST آن اعتماد کامل است. هر گرهٔ دیگری در دسته‌کلید دارای یک اندازهٔ OWNERTRUST تعریف نشده بوده مگر اینکه اندازهٔ دیگری از طرف کاربر برای آن تعیین شده باشد. در این مثال، کاربر مشخص کرده است که همیشه به کاربران D و E و F و L برای امضاء سایر کلیدها اعتماد دارد. این کاربر به کاربران A و B برای امضاء سایر کلیدها، تا حدودی اعتماد دارد.

بنابراین میزان سایه‌دار بودن هر گره در شکل ۷-۵ نمایش دهندهٔ سطح اعتماد تخصیص داده شده بتوسط این کاربر به آن گره است. ساختار درختی نشان می‌دهد که کدام کلیدها بتوسط کدام کاربران امضاء شده اند. اگر یک کلید بتوسط کاربری امضاء شده است که کلیدش در دسته‌کلید وجود دارد، یک پیکان کلید امضاءشده را به امضاءکننده متصل کرده است. اگر کلید بتوسط کاربری امضاء شده است که کلید خود او در دسته‌کلید نیست، یک پیکان کلید امضاءشده را به یک علامت سؤال متصل کرده است که مفهوم آن این است که هویت امضاءکننده برای کاربر ناشناس است.

نکات چندی در شکل ۷-۵ نمایش داده شده است :

۱- توجه کنید که تمام کلیدهائی که صاحبان آنها کاملاً و یا بطور نسبی مورد اعتماد این کاربر بوده‌اند، بجز گرهٔ L، بتوسط این کاربر امضاء شده‌اند. همانطور که حضور گرهٔ L نشان می‌دهد، چنین امضائی از طرف کاربر همیشه ضروری نیست، اما در عمل، بیشتر کاربران محتمل است که اکثر کلیدهای کاربران مورد اعتماد خود را امضاء کنند. بعنوان مثال اگرچه کلید E قبلاً از طرف معرف مورد اعتماد F امضاء شده است، کاربر به انتخاب خود ترجیح داده است که خود هم کلید E را مستقیماً امضاء کند.



شکل ۷-۵ مثالی از مدل اعتماد PGP

۲- فرض می‌کنیم که دو امضاء نسبتاً مورد اعتماد برای تأئید یک کلید کافی باشد. در این صورت کلید کاربر H بتوسط PGP مشروع تلقی می‌گردد زیرا بتوسط A و B که هردوی آنها نسبتاً مورد اعتماد هستند، امضاء شده است.

۳- یک کلید ممکن است مشروع تلقی گردد زیرا بتوسط یک امضاءکننده کاملاً مورد اعتماد و یا دو امضاءکنندهٔ نسبتاً قابل اعتماد امضاء شده است ولی صاحب آن ممکن است برای امضاء سایر کلیدها معتمد فرض نشود. برای مثال، کلید N مشروع است زیرا بتوسط E امضاءشده و این کاربر به E اعتماد دارد، ولی N برای امضاء کلیدهای دیگر مورد اعتماد نیست زیرا این کاربر به N اندازهٔ اعتمادی را تخصیص نداده است. بنابراین اگرچه کلید R بتوسط N امضاءشده است، ولی PGP کلید R را مشروع نمی‌داند. این وضعیت کاملاً معقول است. اگر شما می‌خواهید یک پیام خصوصی برای فردی بفرستید، لازم نیست که به آن فرد از همه نظر اعتماد داشته باشید بلکه تنها کافی است مطمئن باشید که کلید عمومی صحیح آن فرد در اختیار شماست.

۴- شکل ۷-۵ همچنین مثالی از یک گرهٔ "یتیم" S با دو امضاء ناشناخته را نشان می‌دهد. چنین کلیدی ممکن است از یک سرور کلید دریافت شده باشد. PGP نمی‌تواند صرفاً به دلیل اینکه این کلید از یک سرور معروف دریافت شده است آن را مشروع تلقی کند. کاربر بایستی یا با امضاءکردن آن و یا با اظهار تمایل به اینکه یکی از امضاءکنندگان کلید را کاملاً معتمد می‌داند، مشروعیت کلید را به PGP اعلام دارد.

یک نکتهٔ نهائی: قبلاً خاطرنشان گردید که IDهای کاربران متعددی ممکن است با یک کلید عمومی منفرد و یا با یک دسته‌کلید- عمومی مرتبط باشند. این امر بدین خاطر است که یک فرد ممکن است از نام‌های مختلفی استفاده کرده و یا از طریق امضاء تحت نام‌های مختلف، مثلاً آدرس‌های e-mail متفاوتی را برای خودش معرفی نموده باشد. بنابراین می‌توانیم کلید عمومی را همانند ریشهٔ یک درخت بدانیم. یک کلید عمومی دارای تعدادی IDهای مرتبط با آن است که در زیر هر ID نیز تعدادی امضاء قرار دارد. پیوند یک ID کاربر به یک کلید وابسته به امضاءهای مرتبط با آن ID و کلید است، در حالی که سطح اعتماد به آن کلید (برای استفاده در مورد امضاءکردن کلیدهای دیگر) تابعی از تمام امضاءهای وابسته به آن است.

ابطال کلیدهای عمومی

یک کاربر ممکن است بخواهد کلید عمومی خود را باطل کند. این امر یا به دلیل لورفتن کلید و یا به این دلیل است که کاربر کلید را برای مدتی طولانی استفاده کرده و می‌خواهد آن را تعویض کند. توجه کنید که لازمهٔ لورفتن این است که دشمن به نحوی یک کپی از کلید خصوصی رمزنگاری نشده شما را بدست آورده باشد، یا این که دشمن هم کلید خصوصی را از دسته- کلید خصوصی شما بدست آورده و هم عبارت عبور شما را کشف کرده باشد.

قانون ابطال یک کلید- عمومی این است که صاحب آن بایستی یک گواهی‌نامهٔ ابطال که بتوسط صاحب کلید امضاء شده باشد، را تهیه کند. این گواهی‌نامه همان فرم یک گواهی‌نامهٔ امضاء نرمال را داشته اما شامل نشانگری است که نشان می‌دهد که هدف از این گواهی‌نامه لغو استفاده از یک کلید عمومی است. توجه شود که کلید خصوصی نظیر این کلید عمومی بایستی برای امضاء گواهی‌نامه‌ای که یک کلید عمومی را باطل می‌کند بکار رود. صاحب کلید سپس بایستی این گواهی‌نامه را هرچه سریع‌تر و در سطح هرچه وسیع‌تر انتشار دهد تا افراد مرتبط با او متعاقباً دسته‌کلیدهای- عمومی خود را به روز درآورند.

توجه شود که دشمنی که کلید خصوصی یک کاربر را دزدیده است، نیز می‌تواند چنین گواهی‌نامه‌ای را صادر کند. ولی چون این امر هم دشمن و هم صاحب قانونی کلید را از استفاده از کلید محروم می‌سازد، تهدید ایجادشده بسیار کمتر از استفاده بداندیشانه از یک کلید خصوصی دزدیده شده است.

۵-۲ S/MIME

S/MIME (Secure/Multipurpose Internt Mail Extension) که مبتنی بر تکنولوژی برخاسته از شرکت RSA Data Security است، یک شکوفائی امنیتی در MIME که فرمت استاندارد پست الکترونیک در اینترنت است به وجود می‌آورد. اگرچه هم PGP و هم S/MIME هر دو در خط استانداردهای IETF قرار دارند ولی بنظر می‌رسد که نهایتاً S/MIME بصورت استاندارد صنعت برای مصارف تجاری و سازمانی باقی خواهد ماند در حالی که PGP انتخاب غالب کاربران شخصی پست الکترونیک خواهد بود. S/MIME در تعدادی از اسناد تعریف شده است که مهم‌ترین آنها RFCهای 3369، 3370 و 3850 می‌باشند.

برای فهم S/MIME ابتدا لازم است تا از فرمت زیرساخت پست الکترونیکی که S/MIME از آن استفاده می‌کند، یعنی MIME اطلاعاتی داشته باشیم. اما برای درک اهمیت MIME لازم است تا به عقب برگشته و از فرم استاندارد سنّتی پست الکترونیک یعنی RFC 822 که هنوز دارای کاربرد عام است اطلاع حاصل نمائیم. بنابراین در این بخش ابتدا به معرفی این دو استاندارد قدیمی‌تر پرداخته و سپس S/MIME را مورد بحث قرار خواهیم داد.

RFC 822

RFC 822 فرمتی را برای ارسال پیام‌های متنی از طریق پست الکترونیک تعریف می‌کند. این فرمت، استاندارد ارسال پیام‌های متنی مبتنی بر اینترنت بوده و بصورت گسترده‌ای از آن استفاده می‌شود. در بستر RFC 822، چنین تصور می‌شود که هر پیام دارای یک پاکت و یک محتوا است. پاکت شامل همهٔ آن اطلاعاتی است که برای انتقال و تحویل پیام لازم است. محتوا مطلبی است که باید به گیرنده تحویل شود. استاندارد RFC 822 فقط به محتوا مربوط می‌شود. با وجود این، استاندارد محتوا شامل مجموعه‌ای از میدان‌های سرآیند است که ممکن است بتوسط سیستم پستی برای تولید پاکت بکار رود. هدف استاندارد تسهیل شناخت چنین اطلاعاتی بتوسط برنامه‌هاست.

ساختار کلی یک پیام که با RFC 822 همخوانی داشته باشد، بسیار ساده است. یک پیام شامل چند خط سرآیند (*عنوان*) بوده که به دنبال آن متن نامحدودی (*بدنه*) قرار دارد. سرآیند بتوسط یک خط خالی از بدنه جدا می‌شود. به بیان دیگر، یک پیام یک متن ASCII است و تمام خطوط آن تا اولین خط خالی، سرآیندی است که بتوسط عامل کاربر سیستم پستی مورد استفاده قرار می‌گیرد.

یک خط سرآیند معمولاً شامل یک کلمهٔ کلیدی بوده که پس از آن علامت : قرار گرفته و پس ازآن، آرگومان آن کلمهٔ کلیدی نوشته می‌شود. فرمت اجازه می‌دهد که یک خط طولانی به چندین خط کوتاه‌تر شکسته شود. پرکاربردترین کلمات کلیدی *FROM*، *TO*، *SUBJECT* و *DATE* می‌باشند. مثالی از یک پیام در زیر نشان داده شده است:

```
Date: Tue, 16 Jan 1998 10:37:17 (EST)
From: "William Stallings" <ws@shore.net>
Subject: The Syntax in RFC 822
To: Smith@Other-host.com
Cc: Jones@Yet-Another-host.com

Hello. This section begins the actual message body, which is delimited
from the message heading by a blank line.
```

میدان دیگری که معمولاً در سرآیندهای RFC 822 پیدا می‌شود، *Message-ID* است. این میدان شامل یک شناسهٔ یکتا در رابطه با پیام است.

الحاقیه‌های چند منظورهٔ پست الکترونیک (MIME)

MIME توسعه‌ای در چهارچوب RFC 822 ایجاد می‌کند که هدف آن رفع بعضی مشکلات و محدودیت‌های استفاده از SMTP (Simple Mail Transfer Protocol) و یا بعضی پروتکل‌های انتقال پیام دیگر و RFC 822 برای پست الکترونیک می‌باشد. [MURH98] محدودیت‌های زیر برای پروتکل SMTP/822 را ذکر کرده است:

۱- SMTP نمی‌تواند فایل‌های اجرائی یا سایر اشیاء باینری را انتقال دهد. روش‌های مختلفی برای تبدیل فایل‌های باینری به صورت متن وجود دارد که می‌تواند مورد استفادهٔ سیستم‌های پستیSMTP قرار گیرد (مثل روش مرسوم UNIX UUencode/UUdecode). ولی هیچ‌یک از اینها استاندارد نبوده و حتی استاندارد غالب هم نمی‌باشند.

۲- SMTP نمی‌تواند داده‌های متنی شامل کاراکترهای زبان‌های ملّی را انتقال دهد زیرا اینها بتوسط کُدهای ۸- بیتی با مقادیر دهدهی ۱۲۸ به بالا نمایش داده می‌شوند و SMTP محدود به کُد ۷- بیتی ASCII است.

۳- سرورهای SMTP ممکن است پیام‌های پستی طویل‌تر از اندازهٔ معینی را نپذیرند.
۴- دروازه‌های SMTP که مترجم بین کُد ASCII و کُد EBCDIC هستند از یک مجموعه قوانین نگاشت یکسان پیروی نکرده و مشکلات ترجمه ایجاد می‌کنند.
۵- دروازه‌های SMTP به شبکه‌های پست الکترونیک X.400 نمی‌توانند از پس داده‌های غیرمتنی موجود در پیام‌های X.400 برآیند.
۶- بعضی از پیاده‌سازی‌های SMTP کاملاً به استانداردهای SMTP که در RFC 821 تعریف شده است وفادار نیستند. مشکلات معمول چنین‌اند:
- حذف، اضافه و یا بنظم درآوردن بازگشت به اول خط و خط خالی.
- قطع کردن و یا جمع کردن خطوطی که طویل‌تر از ۷۶ کاراکتر هستند.
- حذف فضای سفید در انتهای پیام (کارکترهای tab و space).
- پر کردن بین خطوط یک پیام بصورتی که همه دارای طول یکسان باشند.
- تبدیل کاراکترهای tab به چندین کاراکتر space.

MIME قصد دارد تا این مشکلات را طوری حل کند که با پیاده‌سازی‌های موجود RFC 822 سازگار باشد. مشخصه‌ها در RFCهای 2045 تا 2049 درج شده‌اند.

مروری بر MIME

مشخصه‌های MIME شامل عناصر زیر است:

۱- پنج میدان جدید برای پیام تعریف شده است که می‌توانند در سرآیند RFC 822 جای گیرند. این میدان‌ها شامل اطلاعاتی در مورد بدنه پیام است.
۲- تعدادی فرمت برای محتوا تعریف شده است که صورت ظاهر e-mailهائی که پست الکترونیک چندرسانه‌ای را پشتیبانی می‌کنند استاندارد می‌نماید.
۳- کُدینگ‌هائی برای انتقال تعریف شده‌اند که تبدیل هر نوع فرمت محتوای پیام به فرمی که در برابر تغییر به توسط سیستم پستی محافظت شده است را فراهم می‌سازد.

در این قسمت پنج میدان سرآیند پیام را معرفی می‌کنیم. سپس به فرمت‌های محتوای پیام و کُدینگ‌های انتقال می‌پردازیم.
پنج میدان سرآیند که در MIME تعریف شده است به قرار زیراند:

- **شماره نسخهٔ MIME**: اندازهٔ پارامتر این میدان بایستی 1.0 باشد. این میدان نشان می‌دهد که پیام از RFC 2045 و RFC 2046 تبعیت می‌نماید.
- **نوع محتوا**: داده‌ای که در بدنهٔ پیام قرار دارد را با جزئیات کافی توصیف می‌کند تا کاربر دریافت‌کننده بتواند عامل و یا مکانیسم مناسبی را برای نمایش این داده بکار گیرد و در غیراینصورت با روش مناسبی با دیتا برخورد نماید.
- **روش کُدینگ انتقال محتوا**: نوع تبدیل بکار گرفته شده برای نمایش بدنه پیام بصورتی که برای انتقال پستی قابل قبول باشد را مشخص می‌کند.
- **کُد شناسائی محتوا**: برای معرفی اقلام MIME در زمینه‌های چندگانه بصورت یکتا استفاده می‌شود.

- **توصیف محتوا**، یک توصیف متنی از شیئی که همراه بدنهٔ پیام است. این وقتی مفید است که این شیء قابل خواندن نباشد (مثل داده‌های صوتی).

یک و یا همهٔ میدان‌ها ممکن است در یک سرآیند نرمال RFC 822 ظاهر شوند. یک پیاده‌سازی مبتنی بر این پروتکل بایستی میدان‌های شماره نسخهٔ MIME، نوع محتوا و روش کُدینگ محتوا را پشتیبانی کرده ولی میدان‌های شناسائی محتوا و توصیف محتوا اختیاری بوده و ممکن است در سیستم گیرنده مورد توجه قرار نگیرند.

انواع محتویات MIME

بخش قالب مشخصه‌های MIME مربوط به تعریف اقلام متنوعی برای محتوای پیام است. این امر منعکس‌کنندهٔ نیاز فراهم‌آوردن روش‌های استاندارد نمایش اطلاعات، در یک محیط چند رسانه‌ای است.

جدول ۳-۵ انواع محتوا در RFC 2046 را نشان می‌دهد. ۷ نوع عمده برای محتوا و جمعاً ۱۵ زیرمجموعهٔ محتوائی در این جدول نشان داده شده است. بطور کلی یک محتوا مبین شکل عمومی دیتا بوده و یک زیرمحتوا، فرمت خاص آن محتوا را تعریف می‌کند.

جدول ۳-۵ انواع محتویات MIME

نوع (Type)	زیرنوع (Subtype)	توصیف
Text	Plain	متن فرمت نشده. ممکن است ASCII و یا ISO 8859 باشد.
	Enriched	انعطاف‌پذیری بیشتری را در فرمت فراهم می‌آورد.
Multipart	Mixed	بخش‌های مختلف مستقل از هم بوده ولی بایستی با هم منتقل شوند. آنها بایستی با همان نظمی که در پیام پستی قرار دارند به گیرنده عرضه گردند
	Parallel	فرق آن با زیرنوع Mixed در این است که نظمی برای تحویل بخش‌های مختلف به گیرنده تعریف نشده است.
	Alternative	بخش‌های مختلف، نسخه‌های متفاوت یک نوع اطلاعات می‌باشند. آنها بر حسب نزدیکی بیشتر با فرم اولیه به نظم درآمده‌اند و سیستم پستی گیرنده بایستی «بهترین» نسخه را به کاربر عرضه نماید.
	Digest	شبیه Mixed است با این تفاوت که پیش‌فرض type/subtype هر بخش message/rfc822 است.
Message	rfc822	بدنهٔ پیام خود یک پیام کپسولی شده بر اساس RFC 822 است.
	Partial	برای قطعه قطعه کردن یک واحد طولانی پستی بکار می‌رود بطوری که برای دریافت‌کننده مرئی نباشد.
	External-body	شامل یک نشانگری است که به یک عنصر در جای دیگر اشاره می‌کند.
Image	jpeg	تصویر دارای فرمت JPEG و کُدینگ JFIF است.
	gif	تصویر دارای فرمت GIF است.
Video	mpeg	فرمت MPEG
Audio	Basic	کُدینگ ۸- بیتی تک کانالهٔ ISDN با استفاده از قانون μ و نرخ 8 khz
Application	PostScript	Adobe Postscript
	Octet-stream	داده‌های باینری عمومی شامل بایت‌های ۸- بیتی

اگر بدنهٔ پیام از **نوع متن (text type)** باشد، بجز پشتیبانی از مجموعهٔ کاراکترهای مشخص شده هیچ نرم‌افزار مخصوص دیگری مورد نیاز نیست. زیر مجموعهٔ متن، یکی متن ساده (*plain*) است، که صرفاً دنباله‌ای از کاراکترهای ASCII و یا کاراکترهای مخصوص ISO 8859 است. زیرمجموعهٔ دیگر متن غنی شده (*enriched*) است که قابلیت بیشتری را در فرمت دیتا می‌پذیرد.

نوع چندبخشی (multipart type) نشان می‌دهد که بدنهٔ پیام شامل بخش‌های متعدد و مستقل است. مبدان سرآیند نوع محتوا شامل یک پارامتر بنام مرز (boundry) است که فاصلهٔ بین بخش‌های بدنه را تعریف می‌کند. هر مرز از اول یک خط جدید شروع شده و شامل دو خط فاصله (hyphen) و به دنبال آن یک اندازهٔ مرز است. مرز نهائی که شامل انتهای آخرین بخش است نیز دارای یک پسوند با دو خط فاصله است. درون هر بخش ممکن است یک سرآیند اختیاری معمولی MIME وجود داشته باشد.

در زیر مثال ساده‌ای از یک پیام چندبخشی نشان داده شده است که شامل دو بخش بوده و هر بخش نیز شامل یک متن ساده است (اقتباس از RFC 2046).

```
From:  Nathaniel Borenstein <nsb@bellcore.com>
To:  Ned Freed  <ned@innosoft.com>
Subject:  Sample message
MIME-Version:  1.0
Content-type:  multipart/mixed;  boundary ="simple boundary"

This is the preamble. It is to be ignored, though it is a handy place for
 mail composers to include an explanatory note to non-MIME conformant
readers.--simple boundary

This is implicitly typed plain ASCII text. It does NOT end with a
linebreak.--simple boundary
Content-type:  text/plain;  charset=us-ascii

This is explicitly typed plain ASCII text. It DOES end with a linebreak.

--simple boundary--
This is the epilogue. It is also to be ignored.
```

نوع چندبخشی خود دارای چهار زیـــرنـوع (subtype) است که تمـام آنهـا دارای یـک انشـای کلی هستند. **multipart/mixed subtype** وقتی مورد استفاده قرار می‌گیرد که چندین بخش مستقل در پیام وجود داشته باشد که بایستی به ترتیب مشخصی به هم گره بخورند. در **multipart/parallel subtype** نظم بخش‌های مختلف دارای اهمیت نمی‌باشند. اگر سیستم دریافت مناسب باشد، بخش‌های مختلف پیام می‌توانند بطور موازی نمایش داده شوند. برای مثال یک بخش تصویری و یا متنی می‌تواند با توضیحات صوتی همراه باشد که در حالی که تصویر و یا متن در حال نمایش است بخش صوتی نیز به همراه آن اجرا گردد.

برای **multipart/alternative subtype** بخش‌های مختلف پیام، نمایش‌های مختلفی از یک نوع اطلاعات هستند. مثال زیر نمونه‌ای از آن است:

```
From: Nathaniel Borenstein <nsb@ bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Formatted text mail
MIME-Version: 1.0
Content Type: multipart/alternative; boundary=boundary42
--boundary42

Content-Type: text/plain; charset=us-ascii

… plain text version of message goes here …

--boundary42
Content-Type: text/enriched

… RFC 1896 text/enriched version of some message goes here…
--boundary42--
```

در این زیرنوع، بخش‌های بدنهٔ پیام بر حسب رجحان بر یکدیگر مرتب می‌شوند. برای مثال بالا اگر سیستم گیرنده قادر به نمایش پیام با فرمت متن غنی شده باشد این عمل انجام می‌شود و در غیر اینصورت متن ساده بکار خواهد رفت.

multipart/digest subtype وقتی بکار می‌رود که هریک از بخش‌های بدنه، بصورت یک پیام RFC 822 با سرآیندهای آن تعبیر شود. این زیرنوع ما را قادر به ساخت پیامی خواهد نمود که بخش‌های مختلف آن پیام‌های انفرادی هستند. بعنوان مثال میاندار یک گروه ممکن است پیام‌های e-mail افراد گروه را جمع‌آوری کرده، این پیام‌ها را بسته‌بندی نموده و آنها را بصورت یک پیام کپسولی شدهٔ MIME بفرستد.

نوع پیام(message type) تـعدادی قـابلیت‌های مهم درMIME را فراهم می‌سازد. **message/rfc822 subtype** نشان می‌دهد که خود بدنهٔ پیام یک پیام کامل، شامل سرآیند و بدنه، است. صرفنظر از نام این زیرنوع، پیام کپسولی شده ممکن است نه تنها یک پیام سادهٔ RFC 822 بلکه هر نوع پیام دیگرMIME باشد.

message/partial subtype قطعه‌قطعه کردن یک پیام طولانی به بخش‌های مختلف را امکان‌پذیر می‌کندکه بایستی در مقصد دوباره بهم بپیوندند. برای این زیرنوع، سه پارامتردر میدان Content-Type:Message/Partial field مشخص شده است: یک *id* که برای تمام قطعات مشترک است، یک *شماره ردیف* که برای هر قطعه متفاوت است و تعداد *کل* قطعات.

message/external-body subtype نشان می‌دهد که دیتای واقعی که بایستی از طریق پیام تحویل گردد، در بدنهٔ پیام نیست. بجای آن بدنه شامل اطلاعات لازم برای دسترسی به داده است. همانند دیگر انواع پیام، message/external-body subtype دارای یک سرآیند خارجی و یک پیام کپسولی شده با سرآیند خود آن است. تنها میدان لازم در سرآیند خارجی، میدان نوع محتوا است که این پیام را بعنوان یک زیرنوع message/external-body معرفی می‌کند. سرآیند داخلی، سرآیند پیام برای پیام کپسولی شده است. میدان نوع محتوا در سرآیند خارجی بایستی شامل یک پارامتر نوع دسترسی باشد که نمایشگر روش دسترسی مثل FTP (file transfer protocol) است.

نوع کاربرد(application type) به سایر انواع دیتا اشاره می‌کند که نوعاً یا دیتای باینری ترجمه نشده و یا اطلاعاتی است که بایستی بتوسط یک برنامهٔ کاربردی مبتنی بر پست الکترونیک پردازش شود.

کُدینگ‌های انتقال MIME

یکی از مؤلفه‌های مهم دیگر در مشخصه‌های MIME، علاوه بر تعیین نوع محتوا، تعریف کُدینگ انتقال برای بدنهٔ پیام است. هدف این امر تحویل قابل اعتماد پست الکترونیک در محدودهٔ وسیعی از محیط‌های گوناگون است.

استاندارد MIME دو روش برای کُدکردن دیتا را تعریف کرده است. میدان کُدینگ انتقال محتوا در واقع می‌تواند برابر آنچه در جدول ۴-۵ لیست شده است، شش مقدار را بپذیرد. اما سه تا از این مقادیر (7bit، 8bit و binary) نمایشگر این واقعیت‌اند که هیچ کُدینگی انجام نشده است و فقط اطلاعاتی در مورد نوع دیتا را فراهم می‌سازند. برای انتقال SMTP استفاده از 7bit امن است. فرم‌های 8bit و binary ممکن است در بسترهای حمل پستی دیگر قابل استفاده باشند. اندازهٔ دیگر کُدینگ انتقال محتوا، مقدار x-token است که نشان می‌دهد روش کُدینگ دیگری بکار گرفته شده و بایستی برای آن نامی ارائه گردد. این روش ممکن است مخصوص یک سازندهٔ خاص و یا کاربرد خاص باشد. دو روشی که در این مورد تعریف شده‌اند یکی quoted-printable و دیگری base64 است. این دو روش برای این تعریف شده‌اند که یک حق انتخاب بین یک تکنیک انتقال که ضرورتاً قابل خواندن بتوسط انسان است و دیگری که برای همهٔ انواع دیتا مطمئن بوده و نسبتاً مختصر است، وجود داشته باشد.

کُدینگ انتقال quoted-printable وقتی مفید است که دیتا عمدتاً شامل اُکتت‌هائی باشد که نظیر کاراکترهای قابل چاپ ASCII اند. در واقع این روش کاراکترهای ناامن را با فرم هگزادسیمال کُد آنها نمایش داده و خطوط خالی قابل برگشت (نَرم) را برای محدودکردن خطوط پیام به ۷۶ کاراکتر معرفی می‌کند.

کُدینگ انتقال base64 که کُدینگ radix-64 نیز خوانده می‌شود، روشی معمول برای کُدکردن هر نوع دیتای باینری به نحوی است که در برابر پردازش برنامه‌های حمل پستی آسیب‌ناپذیر باشد. از این روش در PGP هم استفاده می‌شود و در ضمیمهٔ ۵ ب این فصل توصیف شده است.

یک مثال چندبخشی

شکل ۸-۵ که از RFC 2045 گرفته شده است، طرح یک پیام چندبخشی مرکب را نشان می‌دهد. پیام دارای پنج بخش است که بایستی بطور سریال نمایش داده شوند: دو متن ساده در مقدمهٔ، یک پیام چندبخشی جاسازی شده در داخل آن، یک بخش متن غنی‌شده و یک پیام متنی کپسولی شده که با کاراکترهای غیر ASCII بیان شده است. پیام چندبخشی جاسازی شده دارای دو قسمت است که بایستی نشان داده شوند، یک تصویر و یک قطعهٔ صوتی.

جدول ۴-۵ کُدینگ‌های انتقال MIME

7bit	تمام دیتا با خطوط کوتاهی از کاراکترهای ASCII نشان داده می‌شود.
8bit	خطوط کوتاه‌اند ولی ممکن است شامل کاراکترهائی غیر از ASCII باشند (اُکتت‌هائی با مجموعهٔ بیت‌هائی از درجهٔ بالا)
binary	نه تنها کاراکترهای غیر ASCII می‌توانند وجود داشته باشند بلکه خطوط پیام الزاماً برای انتقال از طریق پروتکل SMTP به اندازهٔ کافی کوتاه نیستند.
quoted-printable	داده‌ها را به ترتیبی کُد می‌کند که اگر داده‌های کُد شده بیشتر شامل متون ASCII باشند، فرم کُدشده تا حد زیادی قابل شناسائی بتوسط انسان خواهد بود.
base64	دیتا را با نگاشتی از بلوک‌های ۶- بیتی ورودی به بلوک‌های ۸- بیتی خروجی طوری کُد می‌کند که همهٔ آنها بتوسط کاراکترهای ASCII قابل چاپ‌اند.
x-token	یک نوع کُدینگ غیراستاندارد است.

```
MIME-Version: 1.0
From: Nathaniel Borenstein nsb@bellcore.com
To: Ned Freed ned@innosoft.com
Subject: A multipart example
Content-Type: multipart/mixed;
Boundry=unique-boundry-1

This is the preamble area of a multipart message. Mail readers that understand multipart
format should ignore this preamble. If you are reading this text you might want to consider
changing to a mail reader that understands how to properly display multipart messages.

--unique-boundry-1

  …Some text appears here…
[Note that the preceding blank line means no header fields were given and this is text with
charset US ASCII.
It could have been done wiyh explicit typing as in the next part.]

--unique-boundry-1
Content-type: text/plain;charset=US-ASCII

This could have been part of the previous part, but illustrates explicit versus implicit typing
of body parts.

--unique-boundry-1
Content-Type: multipart/parallel: boundry=unique-boundry-2

--unique-boundry-2
Content-Type: audio/basic
Content-Transfer-Encoding: base64

…base64-encoded 8000 Hz single-channel mu-law-format audio data goes here….

--unique-boundry-2
Content-Type: image/jpeg
Content-Transfer-Encoding: base64

…base64-encoded image data goes here…

--unique-boundry-2--

--unique-boundry-1
Content-type: text/enriched

This is <bold><italic>richtext.</italic></bold><smaller>as defined in RFC  1896</smaller>

Isn`t it<bigger><bigger>cool?</bigger></bigger>

--unique-boundry-1
Content-Type:message/rfc822

From: (mailbox in US-ASCII)
To: (address in US-ASCII)
Subject: (subject in US-ASCII)
Content-Type: Text/plain; charset=ISO-8859-1
Content-Transfer-Encoding: Quoted-printable

…Additional text in ISO-8859-1 goes here…

--unique-boundry-1--
```

شکل ۸-۵ مثالی از ساختار پیام MIME

جدول ۵-۵ فرم‌های بومی و قانونی

فرم بومی (Native Form)	بدنه پیامی که باید ارسال شود با فرمت بومی سیستم ارسال‌کننده خلق می‌گردد. از مجموعهٔ کاراکترهای بومی استفاده شده و در جای مناسب از قوانین محلی پایان خط استفاده می‌شود. بدنه ممکن است یک فایل متنی مبتنی بر UNIX، یک تصویر مبتنی بر Sun، یک فایل با اندیس VMS، یک دیتای صوتی مبتنی بر سیستم که در حافظه ذخیره شده و یا هرچیز دیگری در رابطه با مدل محلی برای نمایش نوعی اطلاعات باشد. اصولاً دیتا به فرم «بومی» که مرتبط با نوع تعیین شده بتوسط نوع رسانه است خلق می‌گردد.
فرم قانونی (Canonical Form)	تمام بدنه پیام، شامل اطلاعات «خارج از باند» مثل طول رکوردها و احتمالاً اطلاعات مربوط به صفات فایل‌ها، به فرم قانونی تبدیل می‌شود. نوع رسانهٔ مخصوص بدنه و مشخصات مربوطه، فرم قانونی بکارگرفته شده را تعیین می‌کنند. تبدیل به فرم قانونی مناسب ممکن است شامل تبدیل مجموعهٔ کاراکترها، تبدیل داده‌های صوتی، فشرده‌سازی و یا سایر عملیات مختص به رسانه‌های مختلف باشند. اگر تبدیل مجموعهٔ کاراکترها مورد نظر باشد بایستی دقت کرد که دیکتهٔ لغات مورد توجه قرار گیرد زیرا ممکن است در تبدیل مجموعه‌ها به یکدیگر تناقضاتی در فرم نمایش آنها بوجود آید.

فرم قانونی

یکی از مفاهیم مهم MIME و S/MIME فرم قانونی(canonical) است. فرم قانونی یک فرمت است، که در تناسب با نوع محتوا، برای استفاده بین سیستم‌ها استاندارد شده است. این در تضاد با یک فرمت بومی است که ممکن است برای یک سیستم خاص دیگر عجیب جلوه نماید. جدول ۵-۵ که از RFC 2049 اقتباس شده است بایستی به درک مطلب کمک کند.

عملکرد S/MIME

از نظر عملکرد کلّی، S/MIME خیلی شبیه PGP است. هردوی آنها قابلیت امضاء و/ یا رمزنگاری پیام‌ها را فراهم می‌آورند. در این قسمت بطور مختصر توانمندی S/MIME را بیان می‌کنیم. سپس با بررسی فرمت‌های پیام و آماده‌سازی پیام به جزئیات این توانمندی می‌پردازیم.

عملیات

S/MIME عملیات زیر را ممکن می‌سازد:

- **Enveloped data**: این شامل محتوای رمزنگاری شده از هر نوع، و کلیدهای رمز محتوای رمزنگاری شده برای یک یا چند گیرنده است.
- **Signed data**: یک امضاء دیجیتال با محاسبهٔ چکیدهٔ پیام از محتوائی که باید امضاء شود و سپس رمزنگاری آن با کلید خصوصی امضاءکننده ایجاد می‌گردد. سپس محتوا باضافهٔ امضاء دیجیتال آن با استفاده از کُدینگ base64 کُد می‌شود. یک پیام signed data تنها بتوسط گیرنده‌ای قابل رؤیت است که قابلیت S/MIME را داشته باشد.

- **Clear-signed data**: همانند signed data، یک امضاء دیجیتال از محتوا تولید می‌شود ولی در این مورد فقط امضاء دیجیتال با استفاده از کُدینگ base64 کُد می‌شود. در نتیجه گیرنده‌هائی که به S/MIME مجهز نیستند نیز می‌توانند محتوای پیام را مشاهده نمایند ولی نمی‌توانند امضاء را تصدیق کنند.
- **Signed and enveloped data**: واحدهای signed-only و encrypted-only می‌توانند تودرتو باشند بطوری که دیتای رمزنگاری شده بتواند امضاء شده و دیتای امضاءشده بتواند رمزنگاری شود.

الگوریتم‌های رمزنگاری

جدول ۶-۵ الگوریتم‌های رمزنگاری بکار رفته در S/MIME را خلاصه کرده است. S/MIME از واژه‌های زیر که از RFC 2119 گرفته شده است استفاده کرده تا سطح نیاز را مشخص نماید:

- **بایستی (MUST)**: یک نیاز قطعی مشخصه است. یک اجراء باید شامل این ویژگی یا این تابع باشد تا با استاندارد تطبیق کند.
- **شایسته است (SHOULD)**: ممکن است در شرایط خاصی دلایل متقنی برای ملحوظ نداشتن این ویژگی یا این تابع وجود داشته باشد، ولی توصیه می‌شود که یک اجراء شامل این ویژگی یا تابع باشد.

S/MIME سه الگوریتم کلید- عمومی را بکار می‌گیرد. استاندارد امضاء دیجیتال (DSS) که در فصل ۳ از آن یاد شد، الگوریتم انتخاب شده برای امضاء دیجیتال است. S/MIME از Diffie-Hellman بعنوان الگوریتم منتخب برای رمزنگاری کلیدهای اجلاس استفاده می‌کند. در حقیقت، S/MIME از یک نوع تغییر یافتهٔ Diffie-Hellman بنام ElGamal که رمزنگاری/ رمزگشائی را فراهم می‌آورد استفاده می‌کند. در انتخاب دیگر، RSA که از آن نیز در فصل ۳ یاد گردید می‌تواند هم برای امضاءها و هم برای رمزنگاری کلید اجلاس بکار رود. این‌ها همان الگوریتم‌هائی هستند که در PGP بکار می‌روند و سطح بالائی از امنیت را فراهم می‌سازند. برای تابع hash که برای خلق امضاء دیجیتال بکار گرفته می‌شود، مشخصه، تابع ۱۶۰- بیتی SHA-1 را تعیین نموده است ولی توصیه می‌کند که گیرنده تابع ۱۲۸- بیتی MD5 را نیز به منظور سازگاری با نسخه‌های قدیمی‌تر S/MIME پشتیبانی نماید. همانطور که در فصل ۳ خاطرنشان گردید، نگرانی‌های قابل بحثی در مورد امنیت MD5 وجود دارد و بنابراین SHA-1 قطعاً انتخاب بهتری است.

برای رمزنگاری پیام، DES سه‌گانه (3DES) سه کلیدی توصیه شده است ولی اجراهای منطبق بایستی RC2 ۴۰- بیتی را پشتیبانی نمایند. مورد اخیر یک الگوریتم رمزنگاری ضعیف ولی منطبق با قوانین کنترل صادرات آمریکا است.

مشخصه‌های S/MIME شامل بحثی در مورد نحوهٔ تصمیم‌گیری نسبت به انتخاب الگوریتم رمزنگاری محتوای پیام است. در واقع یک عامل ارسال‌کنندهٔ پیام بایستی نسبت به دو مورد تصمیم‌گیری نماید. اول اینکه آیا عامل دریافت‌کننده قادر به رمزگشائی یک الگوریتم رمزنگاری هست یا نه. دوم اینکه اگر عامل دریافت‌کننده تنها قادر به پذیرش محتویات رمزنگاری ضعیف است، آیا این امر برای عامل ارسال قابل پذیرش است یا خیر. برای حمایت از این روند تصمیم‌گیری، یک عامل ارسال‌کننده می‌تواند قابلیت‌های رمزگشائی خود را بر حسب یک لیست ترجیحی برای هر پیامی که ارسال می‌شود اعلام دارد. عامل دریافت‌کننده ممکن است این اطلاعات را برای استفاده‌های آتی ذخیره کند.

قواعد زیر، برحسب ترتیب، بایستی بتوسط یک عامل ارسال رعایت شوند:

۱- اگر عامل فرستنده دارای لیستی از قابلیت‌های رمزگشائی گیرندهٔ مورد نظر بصورت ترجیحی است، او شایسته است که نخستین مورد (بالاترین اولویت) لیست،که قادر به استفاده از آن است، را انتخاب کند.

جدول ۶-۵ الگوریتم‌های رمزنگاری استفاده شده در S/MIME

عمل	نیازها
یک چکیدهٔ پیام خلق می‌گردد تا بعداً در تولید یک امضاء دیجیتال از آن استفاده شود.	**بایستی** SHA-1 را پشتیبانی کند. عامل‌های گیرنده **شایسته است** MD5 را برای سازگاری با نسخه‌های قدیمی‌تر پشتیبانی کنند.
چکیدهٔ پیام رمزنگاری می‌شود تا امضاء دیجیتال تولید شود.	عامل‌های فرستنده و گیرنده **بایستی** DSS را پشتیبانی کنند.. عامل‌های فرستنده **شایسته است** رمزنگاری RSA را پشتیبانی کنند. عامل‌های گیرنده **شایسته است** تأئید امضاءهای RSA با کلیدهائی از طول ۵۱۲ تا ۱,۰۲۴ بیت را پشتیبانی کنند.
کلید اجلاس رمزنگاری می‌گردد تا به همراه پیام ارسال شود.	عامل‌های فرستنده و گیرنده **بایستی** رمزنگاری RSA با طول کلیدهائی از ۵۱۲ تا ۱,۰۲۴ بیت را پشتیبانی کنند. عامل‌های فرستنده و گیرنده **شایسته است** Diffie-Hellman را پشتیبانی کنند.
پیام بتوسط کلید اجلاس یکبار- مصرف رمزنگاری می‌شود.	عامل‌های فرستنده و گیرنده **بایستی** رمزنگاری با 3DES را پشتیبانی کنند. عامل‌های فرستنده **شایسته است** رمزنگاری با AES را پشتیبانی کنند. عامل‌های فرستنده **شایسته است** رمزنگاری با RC2/40 را پشتیبانی کنند.
یک کُد اعتبارسنجی پیام خلق می‌شود.	عامل‌های فرستنده **بایستی** HMAC با SHA-1 را پشتیبانی کنند. عامل‌های گیرنده **شایسته است** HMAC با SHA-1 را پشتیبانی کنند.

۲- اگر عامل فرستنده چنین لیستی از یک گیرندهٔ مورد نظر را در اختیار ندارد ولی قبلاً یکی دو پیام از گیرنده دریافت کرده است، آنگاه شایسته است که در پیام خروجی از همان الگوریتم رمزنگاری استفاده کندکه در آخرین پیام امضاء و رمزنگاری شده از همان گیرنده، دریافت کرده است.

۳- اگر عامل فرستنده هیچ اطلاعاتی در مورد قابلیت‌های رمزگشائی گیرندهٔ مورد نظر نداشته ولی آمادگی این ریسک را دارد که حتی به قیمت غیرقابل رمزگشائی شدن پیام، پیام را ارسال کند شایسته است که از 3DES استفاده نماید.

۴- اگر عامل فرستنده هیچ اطلاعاتی در مورد قابلیت‌های رمزگشائی گیرندهٔ مورد نظر نداشته و نمی‌خواهد این ریسک را پذیرا شود که گیرنده نتواند پیام او را بخواند، بایستی از RC2/40 استفاده کند.

اگر قرار باشد که یک پیام به گیرنده‌های متعددی ارسال گردد و یک الگوریتم رمزنگاری مشترک نتواند برای همهٔ آنها انتخاب شود، آنگاه عامل فرستنده نیاز به ارسال دو پیام دارد. در چنین صورتی به این مهم بایستی توجه گردد که امنیت پیام بتوسط انتقال کپی با امنیت پائین‌تر آسیب‌پذیر خواهد شد.

جدول ۷-۵ انواع محتوای S/MIME

نوع	زیرنوع	پارامتر S/MIME	توصیف
Multipart	Signed		یک پیام امضاء شدهٔ صریح در دو بخش: یک بخش پیام و یک بخش امضاء.
Application	pkcs7-mime	signedData	یک موجودیت امضاء شدهٔ S/MIME.
	pkcs7-mime	envelopedData	یک موجودیت رمزنگاری شدهٔ S/MIME.
	pkcs7-mime	degenerate signedData	یک موجودیت که فقط شامل یک گواهی‌نامهٔ کلید- عمومی است.
	pkcs7-mime	compressedData	یک موجودیت فشرده‌سازی شدهٔ S/MIME.
	pkcs7-signature	signedData	نوع محتوای امضاء زیرنوع یک پیام multipart/signed است.

پیام‌های S/MIME

S/MIME از تعدادی محتوای جدید MIME استفاده می‌کند که در جدول ۷-۵ نشان داده شده است. تمام کاربردهای جدید از PKCS استفاده می‌کنند. PKCS به مجموعه‌ای از مشخصه‌های رمزنگاری کلید- عمومی اشاره می‌کند که بتوسط لابراتوارهای RSA نشر شده و در اختیار پروژه S/MIME گذاشته شده است.

در اینجا ابتدا نگاهی به رَوند عمومی آماده‌سازی پیام S/MIME انداخته و سپس محتویات جدید را بررسی می‌کنیم.

ایمن‌سازی یک واحد MIME

S/MIME یک واحد MIME را با امضاء، رمزنگاری و یا هردوی آنها ایمن می‌سازد. یک واحد MIME ممکن است تمام یک پیام (بجز سرآیندهای RFC 822) بوده و یا اگر نوع محتوا از نوع چندبخشی باشد، یک واحد MIME یک یا چند زیربخش از پیام است. یک واحد MIME بر اساس قواعد نرمال آماده‌سازی پیام MIME تهیه می‌شود. سپس واحد MIME بعلاوهٔ بعضی داده‌های مرتبط با امنیت، مثل شناسه‌های الگوریتم‌ها و گواهی‌نامه‌ها، بتوسط S/MIME مورد پردازش قرار گرفته تا آنچه بنام عنصر PKCS است تهیه شود. سپس یک عنصر PKCS بعنوان محتوای پیام در نظر گرفته شده و در MIME لفافه‌بندی می‌شود (بتوسط سرآیندهای مناسب MIME). رَوند عملیات وقتی به عناصر مشخص پرداخته و مثال‌هائی را عرضه کنیم، روشن خواهد شد.

در تمام موارد، پیامی که قرار است ارسال شود به فرم قانونی تبدیل می‌شود. علی‌الخصوص برای یک نوع و زیرنوع داده شده، فرم قانونی مناسب برای پیام انتخاب می‌گردد. برای یک پیام چندبخشی، فرم قانونی مناسب برای هر زیربخش رعایت می‌گردد.

استفاده از کُدینگ انتقال نیاز به توجه ویژه دارد. در بیشتر موارد، نتیجهٔ اعمال الگوریتم‌های امنیتی، تهیهٔ یک عنصر است که بخشی و یا همهٔ آن بصورت دیتای باینری نمایش داده شده است. این عنصر سپس در یک پیام MIME بیرونی لفافه‌بندی شده و سپس کُدینگ انتقال که معمولاً base64 است به آن اعمال می‌گردد. اما در مورد یک پیام چندبخشی امضاءشده که جزئیات آن به زودی توصیف خواهد شد، محتوای پیام در یکی از زیربخش‌ها بتوسط پروسهٔ امنیتی دست نخورده باقی خواهد ماند. بغیر از وقتی که محتوا base64 است، کُدینگ انتقال بایستی از base64 و یا quoted-printable استفاده کند تا خطر تغییر محتوا که امضاء به آن اعمال شده است وجود نداشته باشد.

حال به هریک از انواع محتوای S/MIME نگاهی می‌اندازیم.

EnvelopedData

یک زیرنوع application/pkcs7- mime برای پردازش یکی از چهار دسته S/MIME مورد استفاده قرار می‌گیرد که هریک آنها دارای یک پارامتر smime-type یکتاست. در تمام موارد، عنصر نتیجه شده که یک *object* خوانده می‌شود بصورت فرمی که بنام Basic Encoding Rules (BER) خوانده شده و در توصیه نامهٔ ITU-T X.209 تعریف شده است درمی‌آید. فرمت BER شامل دنباله‌ای از اُکتت‌هاست و بنابراین دیتای باینری است. چنین عنصری بایستی از طریق base64 در پیام بیرونی MIME کُدبندی شود. ابتدا به envelopedData نگاه می‌کنیم.

مراحل آماده‌سازی یک واحد envelopedData در MIME چنین است:

۱- یک کلید اجلاس شبه تصادفی برای یک الگوریتم رمزنگاری متقارن (RC2/40 یا 3DES) تولید شود.

۲- برای هر گیرنده، کلید اجلاس با کلید عمومی RSA گیرنده رمزنگاری شود.

۳- برای هر گیرنده بلوکی با نام RecipientInfo که شامل شناسهٔ گواهی‌نامهٔ کلید- عمومی گیرنده (این گواهی‌نامهٔ X.509 است که بعداً آن را در همین بخش تعریف خواهیم کرد)، یک شناسه برای الگوریتم رمزنگاری استفاده شده برای رمزنگاری کلید اجلاس و خود کلید اجلاس است تهیه شود.

۴- محتوای پیام با کلید اجلاس رمزنگاری شود.

بلوک‌های RecipientInfo که به دنبال آن محتوای رمزنگاری شده قرار داده شده است، envelopedData را تشکیل می‌دهند. این اطلاعات سپس بتوسط base64 کُد می‌شود. نمونه‌ای از این پیام چنین است (سرآیندهای RFC 822 نشان داده نشده‌اند):

```
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
        Name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

Rfvbnj756tbBghyHhHUujhJhjH77n8HHGT9HG4VqpfyF467GhIGfHfYT6
7n8HHGghyHhHUujhJh4VqpfyF467GhIGfHfYGTrfvbnjT6jH7756tbB9H
F8HHGTrfvhjhjH776tbB9HG4VQbnj567GhIGfHfYT6ghyHhHUujpfyF4
0GhIGfHfQbnj756YT64V
```

برای بازیابی پیام رمزنگاری شده، گیرنده ابتدا کُد base64 را باز می‌کند. سپس کلید خصوصی گیرنده برای استخراج کلید اجلاس بکار می‌رود. بالاخره محتوای پیام با استفاده از کلید اجلاس رمزگشائی می‌گردد.

SignedData

signedData smime-type در واقع می‌تواند بتوسط یک و یا چند امضاءکننده بکار رود. به منظور سهولت، توصیف خود را به مورد یک امضاء دیجیتال منفرد محدود می‌کنیم. مراحل آماده‌سازی یک واحد signedData در MIME چنین است:

۱- یک الگوریتم برای چکیدهٔ پیام انتخاب شود (SHA یا MD5).
۲- اندازهٔ چکیدهٔ پیام و یا تابع hash محتوا که باید امضاء شود تهیه گردد.
۳- چکیدهٔ پیام با کلید خصوصی امضاءکننده، رمزنگاری شود.
۴- یک بلوک بعنوان SignerInfo که شامل گواهی‌نامهٔ کلید- عمومی امضاءکننده، شناسه‌ای برای الگوریتم چکیدهٔ پیام، شناسه‌ای برای الگوریتم استفاده شده برای رمزنگاری چکیدهٔ پیام و نهایتاً چکیدهٔ رمزنگاری شدهٔ پیام است، تهیه گردد.

واحد signedData شامل یک سری بلوک‌هائی است که شامل شناسهٔ الگوریتم چکیدهٔ پیام، پیامی که بایستی امضاء شود و SignerInfo است. واحد signedData همچنین می‌تواند شامل یک سری گواهی‌نامهٔ کلید- عمومی باشد که بتواند سلسله مراتب مسئول گواهی‌نامه (CA) مرتبط با امضاءکننده را نشان دهد. این اطلاعات سپس با کُد base64 کُدبندی می‌شود. یک نمونه پیام (بجز سرآیندهای RFC 822) چنین است:

```
Content-Type: application/pkcs7-mime; smime-type=signed-data;
        Name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

567GhIGfHfYT6ghyHhHUujpfyF4f8HHGTrfvhJhjH776tbB9HG4VQbnj7
77n8HHGT9HG4VqpfyF467GhIGfHfYT6rfvbnj756tbBghyHhHUujhJhjH
HuujhJh4VqpfyF467GhIGfHfYGTrfvbnjT6jH7756tbB9H7n8HHGghyHh
6YT64V0GhIGfHfQbnj75
```

برای بازیابی پیام امضاءشده و تأئید امضاء، گیرنده ابتدا کُدینگ base64 را باز می‌کند. آنگاه کلید عمومی امضاءکننده برای رمزگشائی چکیدهٔ پیام مورد استفاده قرار می‌گیرد. گیرنده بطور مستقل چکیدهٔ پیام را محاسبه کرده و به منظور تأئید امضاء آن را با چکیدهٔ رمزگشائی شدهٔ پیام مقایسه می‌کند.

Clear Signing

Clear signing با استفاده از نوع محتوای چندبخشی با یک زیرنوع امضاءشده بدست می‌آید. همانطور که ذکر شد، عمل امضاء شامل رمزکردن پیامی که باید امضاء شود نیست و بنابراین پیام بصورت "clear" ارسال می‌شود. بنابراین گیرنده‌هائی که قابلیت MIME را داشته ولی فاقد قابلیت‌های S/MIME هستند قادر به خواندن پیام ورودی خواهند بود.

یک پیام multipart/signed دارای دو قسمت است. قسمت اول می‌تواند هریک از انواع MIME بوده باشد ولی بایستی طوری تنظیم شود که در خلال انتقال بین فرستنده و گیرنده تغییر نکند. این بدین معنی است که اگر قسمت اول بصورت 7bit نیست، لازم است که با استفاده از base64 و quoted-printable کُدبندی شود. آنگاه این قسمت به همان صورت signedData پردازش می‌شود، اما در این مورد عنصری با فرمت signedData خلق می‌شود که محتوای پیام آن خالی است. این عنصر یک امضاء جدا از پیام است. سپس کُدینگ انتقال با استفاده از base64 روی آن اعمال شده تا قسمت دوم پیام multipart/signed را درست کند. قسمت دوم دارای نوع MIME از نوع application و زیرنوع pkcs7-signature است. نمونه‌ای از این پیام چنین است:

```
Content- Type:  multipart/signed;
     Protocol="application/pkcs7-signature";
     Micalg=shal; boundary=boundary42

--boundary42
Content-Type:  text/plain

This is a clear-signed message.

--boundary42
Content-Type: application/pkcs-signature; name=smime.p7s
Content-Transfer-Encoding:  base64
Content-Disposition:  attachment; filename=smime.p7s

ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB0HG4VqpfyF467GhIGfhfYT6
4VqpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
N8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756
--boundary42--
```

پارامتر پروتکل نشان می‌دهد که این یک واحد two-part clear-signed است. پارامتر micalg نشان‌دهندهٔ نوع چکیدهٔ پیام است. گیرنده می‌تواند با چکیده گرفتن از قسمت اول و مقایسهٔ آن با چکیدهٔ استخراج شده از امضاء در قسمت دوم، امضاء را تأئید نماید.

تقاضای ثبت‌نام

نوعاً یک کاربرد و یا یک کاربر برای تهیهٔ یک گواهی‌نامهٔ کلید- عمومی به یک مسئول صدور گواهی‌نامهٔ (CA) متوسل می‌شود. application/pkcs10 واحد S/MIME برای انتقال درخواست گواهی‌نامه بکار می‌رود. درخواست گواهی‌نامه شامل بلوک certificationRequestInfo بعلاوهٔ یک شناسه الگوریتم رمزنگاری کلید- عمومی بعلاوهٔ امضاء بلوک certificationRequestInfo است که با استفاده از کلید خصوصی فرستنده امضاء شده است. بلوک certificationRequestInfo شامل یک نام (نام واحدی که کلید عمومی او بایستی تأئید گردد) و دنباله‌ای از بیت‌هاست که نمایشگر کلید عمومی کاربر است.

پیام Certificate-Only

یک پیام که فقط شامل گواهی‌نامه‌ها و یا لیست ابطال گواهی‌نامه‌ها (CRL) است می‌تواند در پاسخ به یک تقاضای ثبت‌نام ارسال گردد. پیام یک application/pkcs7-mime type/subtype با یک پارامتر smime-type ابطال است. مراحل اینجا همانند مراحل خلق یک پیام signedData بوده بجز اینکه در اینجا محتوای پیام وجود نداشته و میدان signerInfo خالی است.

پردازش گواهی‌نامه‌های S/MIME

S/MIME از گواهی‌نامه‌های کلید- عمومی که منطبق با نسخهٔ سوم X.509 هستند (به فصل ۴ مراجعه شود) استفاده می‌کند. روش مدیریت-کلید که S/MIME از آن استفاده می‌کند تا حدودی مخلوطی از روش سلسله مراتبی X.509 و وب‌های معتمد PGP است. همانند مدل PGP، مدیران و یا کاربران S/MIME بایستی هر کلاینت را با لیستی از کلیدهای مورد اعتماد و زمان انقضای کلیدها پیکربندی نمایند. یعنی مسئولیت نگهداری گواهی‌نامه‌های لازم برای تأئید امضاءهای ورودی و رمزنگاری پیام‌های خروجی یک مسئولیت محلّی است. علاوه برآن گواهی‌نامه‌ها بتوسط مسئولین صدور گواهی، امضاء می‌شوند.

نقش عامل کاربر

یک کاربر S/MIME بایستی چندین عمل مدیریتی در زمینهٔ مدیریت-کلید انجام دهد:

- **تولید کلید**: کاربر یک برنامهٔ مدیریتی مرتبط (مثل کسی که مدیریت یک شبکهٔ LAN را داراست)، بایستی قادر به تولید جفت کلیدهای Diffie-Hellman و DSS بوده و شایسته است که بتواند جفت کلیدهای RSA را نیز خلق کند. هر جفت کلید بایستی از یک منبع خوب با ورودی تصادفی غیریقینی اخذ شده و به طریق امنی ذخیره گردد. یک عامل کاربر شایسته است جفت کلیدهای RSA را با طولی بین ۷۶۸ تا ۱,۰۲۴ بیت خلق کرده و نبایستی کلیدی با طول کمتر از ۵۱۲ بیت خلق کند.
- **ثبت نام**: کلید عمومی یک کاربر بایستی به منظور اخذ یک گواهی‌نامهٔ کلید- عمومی X.509 در نزد یک مسئول صدور گواهی‌نامه (CA) به ثبت برسد.
- **ذخیره‌سازی و بازیابی گواهی‌نامه‌ها**: یک کاربر، نیازمند دسترسی به یک لیست محلی از گواهی‌نامه‌هاست تا بتواند امضاءهای ورودی را تأئید کرده و پیام‌های خروجی را رمزنگاری نماید. چنین لیستی بایستی یا بتوسط کاربر، و یا بتوسط یک واحد مدیریت محلّی به نیابت تعدادی از کاربران، نگهداری گردد.

گواهی‌نامه‌های VeriSign

سازمان‌های مختلفی وجود دارند که مسئولیت صدور گواهی‌نامه‌های دیجیتال (CA) را تقبل می‌کنند. بعنوان مثال، Nortel یک بنگاه تجاری CA را فراهم نموده و می‌تواند حمایت از S/MIME در درون یک سازمان را عهـده‌دار گردد. CAهائی که مبتنی بر اینترنت هستند نیز وجود داشته که VeriSign، GTE و U.S.Portal Service از آن جمله‌اند. در بین اینها سرویس VeriSign CA بیشترین کاربرد را داشته که توصیف مختصری از آن را در اینجا می‌آوریم.

VeriSign یک سیستم CA را فراهم آورده است که هدف آن سازگاری با S/MIME و تعداد متنوع دیگری از کاربردهاست. VeriSign گواهی‌نامه‌های X.509 را با نام تجاری VeriSign Digital ID صادر می‌کند. در اوایل سال ۱۹۹۸ میلادی بیش از ۳۵,۰۰۰ وب سایت تجاری از VeriSign Digital ID استفاده کرده و بیش از یک میلیون Digital ID برای کاربران مرورگرهای Netscape و Microsoft صادر شده بود.

اطلاعاتی که در یک Digital ID قرار دارد وابسته به نوع Digital ID و موارد استفادهٔ آن دارد. یک Digital ID حداقل شامل اقلام زیر است:

- کلید عمومی صاحب این Digital ID
- نام صاحب Digital ID و یا نام مستعار او
- تاریخ انقضاء Digital ID
- شمارهٔ سریال Digital ID
- نام مسئول صدور گواهی که این Digital ID را صادر کرده است.
- امضاء دیجیتال مسئول صدور گواهی‌نامه که Digital ID را صادر کرده است.

Digital IDها همچنین می‌توانند شامل اطلاعات دیگری باشند که کاربر آنها را عرضه کرده است مثل:

- آدرس
- آدرس e-mail
- اطلاعات عمومی ثبت ID (مثل کشور، کُد محلّی، سن و جنسیت)

VeriSign برابر جدول ۸-۵ سه سطح و یا کلاس امنیتی برای گواهی‌نامه‌های کلید- عمومی فراهم می‌آورد. یک کاربر می‌تواند بصورت برخط از سایت VeriSign و یا سایت‌های مرتبط با آن یک گواهی‌نامه درخواست کند. گواهی‌های Class 1 و Class 2 بصورت برخط پردازش شده و معمولاً ظرف چندثانیه به تأئید می‌رسند. بطور خلاصه رَویه‌های زیر بکار گرفته می‌شود:

- برای Class 1 Digital ID، VeriSign آدرس e-mail کاربر را با ارسال یک PIN و یک فرم برداشت اطلاعات Digital ID به آدرس e-mail او که در درخواست وجود دارد، تأئید می‌کند.
- برای Class 2 Digital ID، علاوه بر انجام عملیات مرتبط با Class 1، یک مقایسهٔ اتوماتیک بین اطلاعات ارائه‌شده در فرم درخواست، با پایگاه داده مشتریان نیز بعمل می‌آورد. در نهایت، تأئیدیه به آدرس پستی مشخص شده ارسال گردیده و به کاربر اطلاع داده می‌شود که یک Digital ID بنام او صادر شده است.
- برای Class 3 Digital ID، VeriSign نیاز به اطمینان سطح بالاتری از هویت درخواست‌کننده دارد. یک فرد متقاضی بایستی هویت خود را از طریق ارائهٔ مدارک ثبت شده‌ای در جای دیگر و یا با مراجعهٔ حضوری به اثبات برساند.

سرویس‌های امنیتی افزوده

تا زمان کتابت این کتاب، سه سرویس امنیتی افزوده در پیش‌نویس‌های اینترنت پیشنهاد شده‌اند. جزئیات این سرویس‌ها ممکن است تغییر کرده و سرویس‌های دیگری نیز به آنها اضافه شوند. این سه سرویس بقرار زیراند:

جدول ۸-۵ انواع گواهی‌نامه‌های کلید- عمومی VeriSign

	نحوهٔ احراز هویت	نحوهٔحفاظت از کلید- خصوصی IA	محافظت از کلید خصوصی متقاضی گواهی و مشترک	کاربردهای قابل انجام و موردنظر کاربر
Class1	جستجوی بدون ابهام و خودکار نام و آدرس e-mail	PCA : سخت‌افزار قابل اعتماد. CA : نرم‌افزار قابل اعتماد یا سخت افزار قابل اعتماد.	نرم‌افزار رمزنگاری (محافظت شده با PIN) توصیه میشود ولی لازم نیست.	مرور صفحات وِب و برخی استفاده‌ها از e-mail
Class2	همانند Class1 باضافهٔ کنترل اتوماتیک اطلاعات عضویت و کنترل اتوماتیک آدرس	PCA و CA : سخت افزار قابل اعتماد.	نرم‌افزار رمزنگاری (محافظت شده با PIN) لازم است.	e-mail شخصی و متعلق به سازمان، اشتراک برخط، تعویض کلمهٔ عبور و تأئید نرم‌افزار
Class3	همانند Class1 باضافهٔ حضور فردی با مدارک معتبر احراز هویت بعلاوهٔ کنترل خودکار Class2 ID برای افراد، و سوابق اداری برای سازمان‌ها	PCA و CA : سخت افزار قابل اعتماد.	نرم‌افزار رمزنگاری (محافظت شده با PIN) لازم است. ژتون سخت‌افزاری تـوصیه می‌شود ولی لازم نیست.	بانکداری الکترونیک، دست‌یابی به پایگاه داده، عملیات بانکی شخصی، سرویس‌های برخط ، پذیرش عضویت، سِرور تجارت الکترونیک، تأئید نرم افزار، اعتبارسنجی LRAAها و رمزنگاری مستحکم برای سِرورهای خاص

IA = Issuing Authority
CA = Certification Authority
PCA = VeriSign Public Primary Certification Authority
PIN = Personal Identification Number
LRAA = Local Registeration Authority Administrator

- **رسیدهای امضاءشده**: یک رسید امضاءشده ممکن است در یک عنصر VeriSign مورد درخواست قرار گیرد. برگرداندن یک رسید امضاءشده برای فرستندهٔ پیام، تحویل پیام را به اثبات رسانده و به ارسال‌کننده اجازه می‌دهد تا به شخص ثالثی اثبات کند که گیرنده، پیام را دریافت کرده است. در واقع گیرنده تمام پیام اولیه بعلاوهٔ امضاء اولیه (امضاءفرستنده) را امضاء کرده و امضاء جدید را به پیام وصل می‌نماید تا یک پیام S/MIME جدید تولید شود.
- **برچسب‌های امنیتی**: یک برچسب امنیتی ممکن است بهمراه مشخصات اعتبارسنجی شدهٔ یک عنصر SignedData ارسال گردد. یک برچسب امنیتی مجموعه‌ای از اطلاعات امنیتی مربوط به حساسیت محتوا است که بتوسط کپسولی کردن S/MIME فراهم آمده است. برچسب‌ها ممکن است برای کنترل دست‌یابی بکار رفته، و نشان دهند که چه کاربرانی می‌توانند به یک عنصر دست یابند. مورد استفادهٔ دیگر آنها تعیین اولویت‌ها (سرّی، محرمانه، محدود و غیره) و یا تعیین نقش فرد می‌باشند که بیانگر نوع آدم‌هائی است که می‌توانند اطلاعات را رؤیت کنند (مثل تیم پزشکی یک بیمار، بخش تعرفه‌های پزشکی و غیره).

- **لیست‌های پستی امن**: وقتی یک کاربر پیامی را برای گیرندگان متعددی می‌فرستد، برای هر گیرنده میزانی پردازش بایستی روی پیام انجام شودکه شامل استفاده از کلید عمومی هر یک از گیرندگان است. کاربر می‌تواند با استفاده از سرویس‌های S/MIME Mail List Agent (MLA) از این وظیفه رها شود. یک MLA می‌تواند یک پیام ورودی تنها را گرفته، رمزنگاری مختص گیرنده برای هر گیرنده را انجام داده و سپس پیام را به جلو راند. ارسال‌کنندهٔ اولیهٔ پیام تنها لازم است پیام را به MLA بفرستد که در این صورت رمزنگاری با کلید عمومی MLA انجام می‌شود.

۵-۳ منابع مطالعاتی

وِب سایت‌های مفید

- **PGP Home Page**: وِب سایت PGP مربوط به .PGP Corp فروشندهٔ پیشتاز محصولات PGP.
- **International PGP Home Page**: برای ارتقاء جهانی استفاده از PGP طراحی شده است. شامل اسناد و لینک‌های مرتبط است.
- **MIT Distribution Site for PGP**: توزیع‌کنندهٔ پیشتاز PGP رایگان. شامل FAQ و سایر اطلاعات بوده و لینک‌هائی نیز به سایت‌های مرتبط دارد.
- **PGP Charter**: آخرین RFCها و پیش‌نویس‌های اینترنت برای Open Specification PGP.
- **S/MIME Charter**: آخرین RFCها و پیش‌نویس‌های اینترنت در مورد S/MIME.

۵-۴ واژه‌های کلیدی، سؤالات مرورکنندهٔ بحث و مسائل

واژه‌های کلیدی

نوعی الگوریتم برای تبدیل داده‌های باینری	radix-64	امضاء جداشده	detached signature
کلید اجلاس	session key	پست الکترونیک	electronic mail
یک ساختار امنیتی برای پست الکترونیک	S/MIME	الحاقیه‌های چندمنظورهٔ پست الکترونیک	Multipurpose Internet Mail Extensions (MIME)
اعتماد	trust	یک ساختار امنیتی برای پست الکترونیک	Pretty Good Privacy (PGP)
یک نوع الگوریتم فشرده‌سازی	ZIP		

سؤالات مرورکننده بحث

۵-۱ پنج سرویس عمده‌ای که بتوسط PGP فراهم می‌آیند کدامند؟

۵-۲ فایدهٔ یک امضاء جدا شده چیست؟

۵-۳ چرا PGP یک امضاء را قبل از فشرده‌سازی تولید می‌کند؟

۵-۴ تبدیل R64 چیست؟

۵-۵ چرا تبدیل R64 برای یک کاربرد پست الکترونیک مفید است؟

۵-۶ چرا عمل قطعه‌قطعه کردن و دوباره سرهم کردن دیتا در PGP مورد نیاز است؟

۵-۷ چگونه PGP از مفهوم trust استفاده می‌کند؟

۵-۸ RFC 822 چیست؟

۵-۹ MIME چیست؟

۵-۱۰ S/MIME چیست؟

مسائل

۵-۱ PGP از مُود فیدبک رمز (CFB) الگوریتم CAST-128 استفاده می‌کند در حالی که اغلب کاربردهای رمزنگاری متقارن (بغیر از رمزنگاری کلید) از مُود زنجیره‌ای رمز قالبی (CBC) استفاده می‌کنند. داریم

$$\text{CBC:} \quad C_i = \mathrm{E}(K, [C_{i-1} \oplus P_i]); \qquad P_i = C_{i-1} \oplus \mathrm{D}(K, C_i)$$
$$\text{CFB:} \quad C_i = P_i \oplus \mathrm{E}(K, C_{i-1}); \qquad P_i = C_i \oplus \mathrm{E}(K, C_{i-1})$$

بنظر می‌رسد که هر دو روش امنیت یکسانی را فراهم می‌سازند. دلیلی ارائه کنید که چرا PGP از مُود CFB استفاده می‌کند.

۵-۲ در روش PGP، تعداد مورد انتظار کلیدهای اجلاس تولید شده، قبل از تکرار یک کلید اجلاس قبلاً خلق شده، چقدر است؟

۵-۳ در PGP، احتمال اینکه کاربری با N کلید عمومی، حداقل یک ID کلید تکراری داشته باشد چقدر است؟

۵-۴ اولین ۱۶ بیت چکیدهٔ پیام در یک امضاء PGP بصورت clear تفسیر می‌گردد.

الف- این امر تا چه حد امنیت الگوریتم hash را زیر سؤال می‌برد؟

ب- این امر واقعاً تا چه حد مقصود را که همانا کمک به درک این مطلب است که آیا کلید صحیح RSA برای رمزگشائی چکیده بکار رفته است، برآورده می‌نماید؟

۵-۵ در شکل ۵-۴ هر قلم در دسته‌کلید- عمومی شامل یک میدان trust است که میزان اعتماد مرتبط با صاحب این کلید- عمومی را نشان می‌دهد. چرا این کافی نیست؟ یعنی اگر این صاحب کلید مورد اعتماد است و این همان کلید عمومی اوست، چرا این اعتماد برای PGP کافی نیست تا این کلید عمومی را بکار برد.

۵-۶ تبدیل radix-64 را بعنوان نوعی رمزنگاری در نظر بگیرید. در این صورت کلیدی وجود ندارد. اما فرض کنید که یک دشمن تنها می‌داند که نوعی الگوریتم جایگذاری برای رمزکردن متن انگلیسی بکار رفته است. این الگوریتم در برابر شکستن رمز تا چه حد امن است؟

۵-۷ Phil Zimmermann، IDEA، 3DES سه کلیدی و CAST-128 را بعنوان الگوریتم‌های رمزنگاری متقارن برای PGP برگزید. دلایلی ذکر کنید که چرا الگوریتم‌های رمزنگاری متقارن زیر که در این کتاب مورد بحث قرار گرفته‌اند برای PGP مناسب و یا نامناسب‌اند: DES، 3DES دو کلیدی و AES.

ضمیمهٔ ۵- الف فشرده‌سازی دیتا با استفاده از ZIP

PGP از یک بستهٔ نرم‌افزاری مخصوص فشرده‌سازی بنام ZIP استفاده می‌کندکه بتوسط Jean-lup Gailly، Mark Adler و Richard Wales نوشته شده است. ZIP یک نرم‌افزار رایگان بوده که به زبان C نوشته شده است و بعنوان یک برنامهٔ سودمند روی UNIX و بعضی سیستم‌های دیگر اجرا می‌شود. ZIP از نظر عملیاتی معادل PKZIP است که یک اشتراک‌افزار پراستفاده در سیستم‌های Windows بوده و به توسط PKWARE,Inc. تهیه شده است. الگوریتم zip شاید معمول‌ترین تکنیک فشرده‌سازی در سیستم عامل‌های متفاوت بوده و نسخه‌های رایگان و اشتراکی آن برای Macintosh و سایر سیستم‌ها ازجمله Windows و UNIX موجود است.

Zip و الگوریتم‌های مشابه آن از تحقیقات Jacob Ziv و Abraham Lempel سرچشمه می‌گیرند. در سال ۱۹۷۷ میلادی، آنها روشی را که بر پایهٔ یک حافظهٔ موقت از نوع پنجرهٔ لغزان قرار داشته و آخرین متن پردازش شده را نگاه می‌داشت، توصیف نمودند[ZIV77]. از این الگوریتم معمولاً با نام LZ77 یاد می‌شود. نسخه‌ای از این الگوریتم در روش فشرده‌سازی zip بکار گرفته شده است (PKZIP، gzip، zipit و غیره).

LZ77 و انواع دیگر آن از این واقعیت استفاده می‌کنند که کلمات و جملات یک متن (صور تصویری در مورد GIF) دارای تکرارهای احتمالی هستند. وقتی تکرار واقع می‌شود، ردیف تکرارشده را می‌توان با یک کُد کوتاه جایگزین نمود. برنامهٔ فشرده‌سازی به دنبال چنین تکرارهائی گشته و کُدهائی را برای جایگزینی دنباله‌های تکرارشده تولید می‌نماید. در طول زمان از کُدها برای پیداکردن دنباله‌های جدید استفاده می‌شود. الگوریتم بایستی بنحوی تعریف شود که برنامهٔ بازکننده قادر به کُدگشائی و بازیابی متن اصلی داده‌ها باشد.

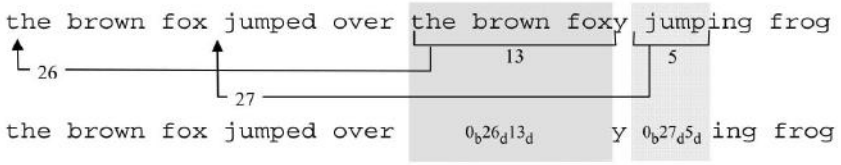
قبل از مطالعهٔ جزئیات LZ77 اجازه دهید تا به یک مثال ساده بپردازیم. جملهٔ بی‌معنی زیر را

the brown fox jumped over the brown foxy jumping frog

که دارای طول ۵۳ اُکتت = ۴۲۴ بیت است را در نظر بگیرید (این مثال از[WEIS93] اقتباس شده است). الگوریتم این متن را، از چپ به راست پردازش می‌کند. در ابتدا هر کاراکتر بصورت یک پترن ۹- بیتی که شامل یک بیت 1 و به دنبال آن نمایش ۸- بیتی کُد ASCII آن کاراکتر است در می‌آید. همین‌طور که پردازش ادامه می‌یابد، الگوریتم به دنبال دنباله‌های تکراری می‌گردد. وقتی به یک تکرار برخورد می‌کند، الگوریتم به اسکن خود ادامه داده تا تکرار خاتمه یابد. بعبارت دیگر هربار تکراری واقع می‌شود، الگوریتم هر تعداد کاراکتر را که ممکن است جایگزین می‌کند. اولین دنبالهٔ تکراری در جملهٔ بالا، **the brown fox** است. این دنباله بتوسط یک نشانگر به دنبالهٔ قبلی و همچنین طول دنباله جایگزین می‌شود. در این مورد، دنبالهٔ قبلی **the brown fox** در ۲۶ کاراکتر قبل واقع شده و طول دنبالهٔ تکرارشده ۱۳ کاراکتر است. برای این مثال، دو راه حل برای کُدینگ تصور کنید: یک نشانگر ۸- بیتی و یک طول ۴- بیتی، یا یک نشانگر ۱۲- بیتی و یک طول ۶- بیتی. یک سرآیند۲- بیتی نشان می‌دهد که کدام روش انتخاب شده است، 00 نمایش‌دهندهٔ روش اول و 01 نمایش‌دهندهٔ روش دوم است. بنابراین دومین وقوع **the brown fox** بصورت $<00_b><26_d><13_d>$ و یا 1101 00011010 00 کُد می‌شود.

بخش‌های باقیماندهٔ پیام فشرده شده، حرف y، دنبالهٔ $<0_b><27_d><5_d>$ که جایگزین دنبالهٔ شامل کاراکتر space و به دنبال آن **jump** می‌شود و دنبالهٔ کاراکترهای **ing frog** است می‌گردد.

شکل ۹-۵ نگاشت فشرده‌سازی را نشان می‌دهد. پیام فشرده شده شامل ۳۵ کاراکتر ۹- بیتی و دو کُد است که عملاً ۳۴۳= ۱۴ × ۲ + ۹ × ۳۵ بیت می‌شود. اگر این پیام فشرده شده را با پیام فشرده نشدهٔ اصلی که شامل ۴۲۴ بیت است مقایسه کنیم، نسبت فشرده‌سازی برابر ۱٫۲۴ بدست می‌آید.



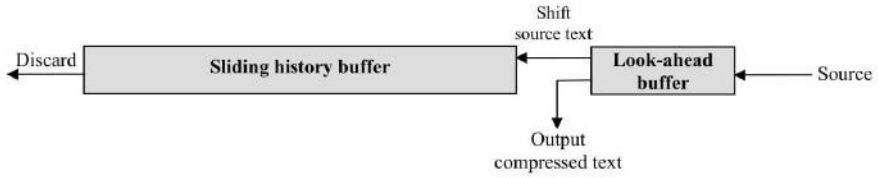
شکل ۹-۵ مثالی از روش LZ77

الگوریتم فشرده‌سازی

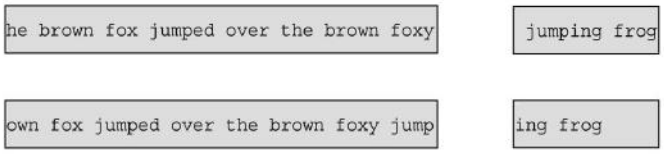
الگوریتم فشرده‌سازی LZ77 و انواع متنوع دیگر آن از دو حافظهٔ موقت استفاده می‌کنند. یک **حافظهٔ موقت sliding history** شامل آخرین N کاراکتر منبع که مورد پردازش قرار گرفته‌اند بوده و یک **حافظهٔ موقت look-ahead** که شامل L کاراکتر بعدی است که بایستی پردازش شوند (شکل ۱۰-۵الف). الگوریتم تلاش می‌کند تا دو یا چند کاراکتر از شروع حافظهٔ موقت look-ahead را با یک دنباله در حافظهٔ موقت sliding history تطبیق دهد. اگر چنین تطبیقی یافت نشود، اولین کاراکتر در حافظهٔ look-ahead بصورت یک کاراکتر ۹- بیتی خارج شده و به درون پنجرهٔ لغزان شیفت داده می‌شود و ازآن طرف قدیمی‌ترین کاراکتر درون پنجرهٔ لغزان نیز بیرون رانده می‌شود. اگر تطبیقی یافت شود، الگوریتم به اسکن کردن ادامه داده تا طویل‌ترین تطبیق را پیدا کند. آنگاه دنبالهٔ تطبیق یافته بصورت یک میدان سه‌تائی (نمایشگر، نشانگر، طول) خارج می‌شود. برای یک دنبالهٔ K تائی، قدیمی‌ترین K کاراکتر موجود در پنجرهٔ لغزان بیرون رانده شده و K کاراکتر دنبالهٔ کُدشده به داخل پنجره رانده می‌شوند.

شکل ۱۰-۵ب این عملیات را بر روی دنبالهٔ مثال ما نشان می‌دهد. در این نمایش یک پنجرهٔ لغزان ۳۹-کاراکتری و یک حافظهٔ موقت look-ahead با ۱۳ کاراکتر در نظر گرفته شده‌اند. بخش بالای شکل، اولین ۴۰ کاراکتر پردازش شده و نسخهٔ فشرده نشدهٔ اخیرترین ۳۹ کاراکتر در داخل پنجرهٔ لغزان است. بقیهٔ کاراکترهای منبع در پنجرهٔ look-ahead قرار دارند. الگوریتم فشرده‌سازی تطبیق بعدی را تعیین نموده، ۵ کاراکتر را از حافظهٔ موقت look-ahead به داخل پنجرهٔ لغزان رانده و کُد خروجی این دنباله را تعیین می‌کند. وضعیت حافظهٔ موقت پس از این عملیات در قسمت پائین شکل نشان داده شده است.

در حالیکه LZ77 مفید بوده و سعی در تطبیق خود با ماهیت داده‌های ورودی دارد، ولی دارای نقاط ضعفی نیز هست. الگوریتم برای جستجو و تطبیق در متن قبلی از یک پنجرهٔ محدود استفاده می‌کند. برای یک بلوک خیلی طولانی از متن، که قابل مقایسه با اندازهٔ پنجره باشد، خیلی از تطبیق‌های مؤثر حذف می‌شوند. اندازهٔ پنجره را می‌توان افزایش داد ولی این امر شامل دو پنالتی خواهد بود: (۱) زمان پردازش الگوریتم افزایش می‌یابد زیرا الگوریتم بایستی برای هر مکان پنجرهٔ لغزان یک مقایسهٔ دنباله‌ای با حافظهٔ موقت look-ahead انجام دهد و (۲) میدان< نشانگر > بایستی وسیع‌تر بوده تا پرش‌های بزرگتری را امکان‌پذیر سازد.



(الف) ساختار عمومی



(ب) مثال

شکل ۱۰-۵ روش LZ77

الگوریتم معکوس فشرده‌سازی

از فشردگی خارج کردن یک متن فشرده شده بتوسط LZ77 کار ساده‌ای‌است. الگوریتم این عمل بایستی آخرین N کاراکتر خروجی بازشده را ذخیره نماید. وقتی به یک دنبالهٔ کُدشده برخورد می‌شود، الگوریتم بازکننده از میدان‌های< نشانگر> و < طول> استفاده کرده و کُد را با دنبالهٔ واقعی متن جایگزین می‌نماید.

ضمیمهٔ ۵- ب تبدیل Radix-64

هم PGP و هم S/MIME از یک روش کُدینگ که تبدیل radix-64 خوانده می‌شود استفاده می‌کنند. این تکنیک هر ورودی باینری دلخواه را به خروجی‌های قابل چاپ تبدیل می‌کند. فرم کُدکردن دارای خصوصیات مرتبط زیر است:

۱- برد تابع یک مجموعه از کاراکترهاست که بطور جهانی در هر سایتی قابل نمایش است و نه یک کُد باینری خاص از این مجموعهٔ کاراکتری. بنابراین خود کاراکترها می‌توانند بتوسط یک سیستم خاص به هر فرمی‌که لازم است کُد شوند. بعنوان مثال، کاراکتر "E" در یک سیستم مبتنی بر کُد ASCII بصورت 45 هکزادسیمال و در یک سیستم مبتنی بر کُد EBCDIC بصورت C5 هکزادسیمال نشان داده می‌شود:

۲- مجموعهٔ کاراکتری شامل ۶۵ کاراکتر قابل چاپ است که یکی از آنها برای لائی (padding) بکار می‌رود. با ۶۴-۲۶ کاراکتر موجود، هر کاراکتر میتواند برای نمایش ۶ بیت ورودی بکار رود.

۳- هیچ کاراکتر کنترلی در مجموعه وجود ندارد. بنابراین یک پیام کُدشده بصورت radix-64 می‌تواند در یک سیستم پستی که دنبالهٔ دیتا را بمنظور یافتن کاراکترهای کنترلی اسکن می‌کند به جلو رانده شود.

۴- کاراکتر خط فاصله ("-") بکار نمی‌رود. این کاراکتر در RFC 822 دارای استفادهٔ خاص بوده و بنابراین بایستی در اینجا از آن پرهیز شود.

جدول ۹-۵ کُدینگ Radix-64

6-bit value	Character encoding	6-bit value	Character encoding	6-bit value	Character encoding	6-bit value	Character encoding
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/
						(pad)	=

جدول ۹-۵ نگاشت مقادیر ۶- بیتی ورودی به کاراکترها را نشان می‌دهد. مجموعهٔ کاراکترها شامل کاراکترهای حرفی و عددی باضافهٔ "+" و "/" است. کاراکتر "=" بعنوان کاراکتر padding مورد استفاده قرار می‌گیرد.

شکل ۱۱-۵ روش سادهٔ نگاشت را نشان می‌دهد. ورودی باینری بصورت بلوک‌های ۳- اُکتتی یا ۲۴- بیتی پردازش می‌شوند. هر گروه ۶- بیتی در بلوک ۲۴- بیتی به یک کاراکتر نگاشت می‌شود. در شکل کاراکترها بصورت مقادیر ۸ - بیتی کُد شده‌اند. در موارد معمول، یک ورودی ۲۴ - بیتی بصورت یک خروجی ۳۲ - بیتی توسعه می‌یابد.

برای مثال، دنبالهٔ متن خام ۲۴- بیتی 00100011 01011100 10010001 که در فرم هگزادسیمال می‌تواند بصورت 235C91 نشان داده شود را در نظر بگیرید. این ورودی را بصورت بلوک‌های ۶- بیتی مرتب می‌کنیم:

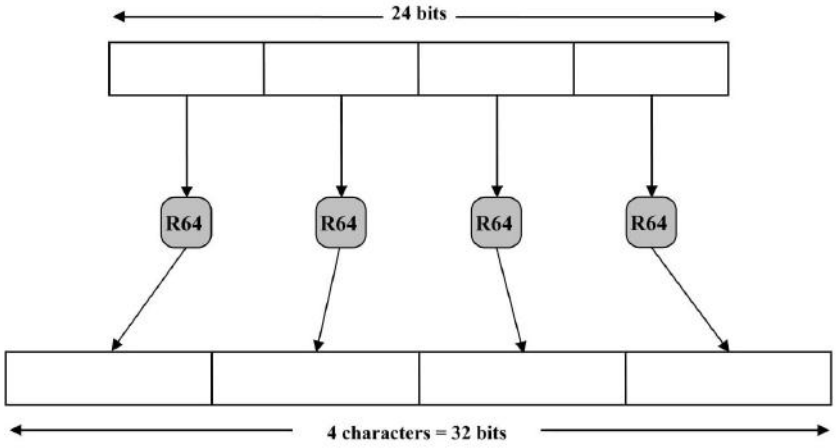
001000 110101 110010 010001

اندازه‌های دهدهی نظیر ۶- بیتی‌ها ۸، ۵۳، ۵۰ و ۱۷ هستند. با مراجعه به جدول ۹-۵ کُد radix-64 این دنباله I1yR خواهد بود. اگر این کاراکترها به فرمت ASCII با بیت parity صفر نگاشت شوند، خواهیم داشت

01001001 0011001 01111001 01010010

در هگزادسیمال نمایش این دنباله بصورت 49317952 خواهد بود. بطور خلاصه،

دادهٔ ورودی	
نمایش باینری	00100011 01011100 10010001
نمایش هگزادسیمال	235C91
کُدینگ Radix-64 دادهٔ ورودی	
نمایش علائم	I1yR
کُد ASCII(8 bit,zero parity)	01001001 00110001 01111001 01010010
نمایش هگزادسیمال	49317952



شکل ۱۱-۵ کُدینگ قابل چاپ داده‌های باینری به فرمی Radix-64

ضمیمهٔ ۵- ج تولید اعداد تصادفی در PGP

PGP از یک روش پیچیده و قدرتمند برای تولید اعداد تصادفی و اعداد شبه‌تصادفی، برای منظورهای متفاوت، استفاده می‌کند. PGP اعداد تصادفی را از نوع و زمان حرکت کلیدها بتوسط کاربر، و اعداد شبه تصادفی را با استفاده از یک الگوریتم که مبتنی بر روشی در ANSI X9.17 است تولید می‌کند. PGP از این اعداد برای مقاصد زیر استفاده می‌کند:

- اعداد تصادفی واقعی:
  - برای تولید جفت کلیدهای RSA
  - بعنوان بذر اولیه در تولید اعداد شبه‌تصادفی
  - برای تولید ورودی دیگری در خلال تولید اعداد شبه‌تصادفی
- اعداد شبه‌تصادفی:
  - برای تولیدکلیدهای اجلاس
  - برای تولید بردارهای شروع (IV) همراه با کلید اجلاس در مُود رمزنگاری CFB

اعداد تصادفی واقعی

PGP یک حافظهٔ موقت ۲۵۶- بایتی از بیت‌های تصادفی را نگاه می‌دارد. هربار که PGP انتظارحرکت کلیدی را دارد، زمان شروع انتظار را بصورت یک فرمت ۳۲- بیتی ثبت می‌کند. وقتی حرکت کلید دریافت می‌شود، زمان حرکت کلید و نوع کلید فشرده‌شده با یک اندازهٔ ۸- بیتی ثبت می‌گردد. اطلاعات زمان و حرکت کلید برای تولید یک کلید رمز بکار گرفته شده که این کلید بنوبهٔ خود برای رمزکردن اندازهٔ جاری حافظهٔ موقت بیت- تصادفی بکار می‌رود.

اعداد شبه‌تصادفی

تولید عدد شبه‌تصادفی از یک بذر ۲۴- اُکتتی استفاده کرده و یک کلید اجلاس ۱۶- اُکتتی ، یک بردار شروع ۸- اُکتتی و یک بذر جدید برای استفاده در دور بعدی تولید عدد تصادفی را تولید می‌کند. الگوریتم مورد استفاده مبتنی بر الگوریتم X9.17 بوده که برای رمزنگاری بجای DES از CAST-128 استفاده می‌کند. الگوریتم از ساختمان دادهٔ زیر استفاده می‌کند:

۱- ورودی

- randseed.bin (۲۴ اُکتت): اگر این فایل خالی باشد، با ۲۴ اُکتت تصادفی واقعی پر می‌شود.
- message: کلید اجلاس و IV که از آنها برای رمزنگاری یک پیام استفاده می‌شود خود تابعی از آن پیام هستند. این امر به تصادفی‌تر شدن کلید و IV کمک می‌کند و اگر یک دشمن قبلاً متن سادهٔ پیام را پیدا کرده باشد ظاهراً نیازی به کلید اجلاس یکبار- مصرف نیست.

۲- خروجی

- K (۲۴ اُکتت): اولین ۱۶ اُکتت،K[0…15]، شامل یک کلید اجلاس و آخرین ۸ اُکتت،K[16…23]، شامل یک IV است.
- randseed.bin (۲۴ اُکتت): یک اندازهٔ جدید بذر در این فایل قرار می‌گیرد.

۳- ساختمان دادهٔ داخلی

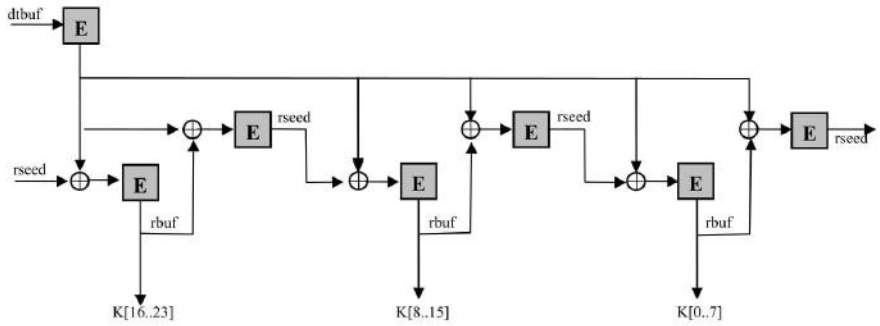
- dtbuf (۸ اُکتت): ۴ اُکتت اول، dtbuf[0…3]، در ابتدا با اندازه‌های جاری تاریخ/زمان پر می‌شوند. این حافظهٔ موقت، معادل متغیر DT در الگوریتم X12.17 است.
- rkey (۱۶ اُکتت): کلید رمزنگاری CAST-128 که در تمام مراحل الگوریتم از آن استفاده می‌شود.
- rseed (۸ اُکتت): معادل متغیر V_i در X12.17 .
- rbuf (۸ اُکتت): یک عدد شبه‌تصادفی که بتوسط الگوریتم تولید می‌شود. این حافظهٔ موقت معادل متغیر R_i در X12.17 است.
- K' (۲۴ اُکتت): حافظهٔ موقت برای اندازهٔ جدید randseed.bin.

الگوریتم شامل نه قدم G1 تا G9 است. قدم‌های اول و آخر قدم‌های ابهام‌زائی به منظور کاهش ارزش یک فایل randseed.bin در صورت کشف دشمن است. قدم‌های باقیمانده ضرورتاً معادل سه بار تکرار الگوریتم X12.17 بوده و در شکل ۱۲-۵ نشان داده شده است. بطور خلاصه:

G1. [Prewash previous seed]

الف- randseed.bin را در K[0…23] کپی کنید.

ب- hash پیام را حساب کنید (اگر پیام امضاء شده باشد این مقدار قبلاً محاسبه شده است. در غیر این‌صورت اولین 4K اُکتت پیام بکار خواهد رفت). از نتیجه به عنوان یک کلید استفاده کنید، از یک IV صفر استفاده کرده و K را در مُود CFB رمزنگاری نمائید. نتیجه را در K ذخیره نمائید.



شکل ۱۲-۵ تولید کلید اجلاس و IV در PGP (قدم‌های G2 تا G8)

[Set initial seed] . G2

الف- dtbuf[0…3] را با ۳۲- بیت زمان محلی تنظیم کنید. dtbuf[4…7] را تماماً صفر بگذارید. rkey←K[0…15] و rseed←K[16…23].

ب- dtbuf ۶۴- بیتی را با استفاده از rkey ۱۲۸- بیتی در مُود ECB رمزنگاری کرده و نتیجه را در dtbuf ذخیره کنید.

[Prepare to generate random octets] . G3 rcount← 0 و k← 23. قدم‌های G4-G7 بصورت یک حلقه ۲۴ بار (k=23…0) اجرا می‌شوند که هر بار برای یک اُکتت تصادفی تولید شده و قرارداده شده در K است. متغیر rcount تعداد اُکتت‌های تصادفی استفاده نشده در rbuf را نشان می‌دهد. برای تولید ۲۴ اُکتت سه بار از 8 تا 0 رو به پائین شمارش می‌شوند.

[Bytes available?] . G4 اگر rcount = 0 به G5 و در غیر اینصورت به G7 بروید. قدم‌های G5 و G6 یک مرتبه الگوریتم X12.17 را برای تولید یک گروه هشت‌تائی از اُکتت‌های تصادفی اجرا می‌کنند.

[Generate new random octets] . G5

الف- rseed ← rseed ⊕ dtbuf

ب - rseed ← E (rkey , rseed) در مُود ECB.

[Generate next seed] . G6

الف- rseed ← rbuf ⊕ dtbuf

ب- rseed ← E (rkey , rseed) در مُود ECB

ج- rcount ← 8 قرار داده شود.

[Transfer one byte at a time from rbuf to K] . G7

الف- rcount ← rcount-1 قرار داده شود.

ب- یک بایت تصادفی واقعی b تولید کرده و K[k] ← rbuf[rcount] ⊕ b

[Done?] If k=0 goto G9 else set k← k-1 and goto G4 . G8

[Postwash seed and return result] . G9

الف- ۲۴ بایت دیگر بتوسط روش G4-G7 تولید کنید باستثنای اینکه در G7 عمل XOR بایت تصادفی را انجام ندهید.

ب- K' را با کلید K[0…15] و IV را با کلید K[16…23] در مُود CFB رمزنگاری نمائید. نتیجه را در randseed.bin ذخیره کنید.

ج- K را برگردانید.

قاعدتاً نبایستی بتوان کلید اجلاس را از ۲۴ اُکتت جدید تولیدشده در قدم G9 الف تعیین نمود. با وجود این برای اطمینان از اینکه فایل randseed.bin ذخیره‌شده هیچگونه اطلاعاتی در مورد آخرین کلید اجلاس به دست نمی‌دهد، ۲۴ اُکتت جدید، رمزنگاری شده و نتیجهٔ عمل بعنوان بذر جدید ذخیره می‌شود.

این الگوریتم پیچیده قاعدتاً اعداد شبه‌تصادفی قدرتمندی را فراهم می‌آورد.

فصل ۶

امنیت IP

معیت اینترنت، مکانیسم‌های امنیتی متفاوتی را برای کاربردهای مختلف و مختص به آنها طراحی نموده است که شامل پست الکترونیک (PGP ,S/MIME)، کلاینت/ سرور (Kerberos)، دست‌یابی به وب (Secure Sockets Layer) و غیره است. با وجود این، کاربران در رابطه با امنیت دارای نگرانی‌های هستند که مربوط به لایه‌های پروتکلی است. بعنوان مثال یک بنگاه تجاری بزرگ می‌تواند یک شبکهٔ TCP/IP خصوصی امن را با جلوگیری از ارتباط با سایت‌های غیرمطمئن، رمزنگاری بسته‌هائی که از سازمان خارج می‌شوند و اعتبارسنجی بسته‌های دیتائی که به سازمان وارد می‌شوند بوجود آورد. با پیاده‌سازی امنیت در سطح IP، یک سازمان می‌تواند شبکه‌ای امن، نه تنها برای کاربردهائی که مکانیسم امنیتی دارند، بلکه برای بسیاری از کاربردهائی که از امنیت بی‌بهره اند بوجود آورد.

امنیت سطح IP سه محدودهٔ عملیاتی را در بر می‌گیرد: اعتبارسنجی، محرمانگی و مدیریت کلید. مکانیسم اعتبارسنجی این اطمینان را ایجاد می‌کند که یک بستهٔ دریافت شده در واقع بتوسط همان واحدی که در سرآیند بسته مشخص شده ارسال گردیده است. بعلاوه این مکانیسم اطمینان می‌دهد که بسته در مسیر ترانزیت بین فرستنده و گیرنده تغییر نکرده است. سرویس محرمانگی، گره‌های مرتبط را قادر می‌سازد پیام‌ها را رمزنگاری کرده تا از استراق سمع اشخاص ثالث محفوظ بمانند. تسهیلات مدیریت کلید، مرتبط با مبادلهٔ امن کلیدهاست.

این فصل را با مروری بر امنیت IP (IPSec) و معرفی معماری IPSec آغاز می‌کنیم. آنگاه به هریک از سه سطح عملیاتی نگاهی مفصل می‌اندازیم. ضمیمهٔ این فصل، مروری بر پروتکل‌های اینترنت است.

۶-۱ مروری بر امنیت IP

در سال ۱۹۹۴ میلادی، گروه معماری اینترنت (IAB) گزارشی را با عنوان « امنیت در معماری اینترنت» ارائه نمودند (RFC1636). گزارش بیانگر این اتفاق نظر بود که اینترنت نیاز به امنیت بیشتر و بهتری دارد. رئوس کلیدی این نیازها نیز در این گزارش ذکر شده بود. در بین اینها، نیاز به امن ماندن زیرساخت شبکه از پایش‌های غیرمجاز، نیاز به کنترل ترافیک شبکه و نیاز به امن نگاه داشتن ترافیک بین یک کاربر انتهائی و کاربر انتهائی دیگر با استفاده از سازوکارهای اعتبارسنجی و رمزنگاری وجود داشت.

این نگرانی‌ها کاملاً بجا هستند. در تأئید آن، گزارش سالیانهٔ ۲۰۰۱ تیم پاسخگوئی به فوریت‌های کامپیوتری (CERT) قریب به ۵۲،۰۰۰ پیشامد امنیتی را لیست نموده است. جدی‌ترین حملات، IP Spoofing بوده که در آن مهاجمین بسته‌هائی را با آدرس‌های IP جعلی خلق کرده و کاربردهائی که از اعتبارسنجی مبتنی بر IP استفاده می‌نمایند را مورد سوء استفاده قرار داده بودند. همچنین فرم‌های متفاوت استراق سمع و بوکشیدن بسته‌ها که درآن مهاجمین، اطلاعات ارسال شده شامل اطلاعات logon و محتویات پایگاه‌های داده، را خوانده بودند مشاهده می‌شد.

در پاسخ به این مقوله‌ها، IAB اعتبارسنجی و رمزنگاری را بعنوان مشخصه‌های امنیتی لازم در نسل بعد IP که بنام IPv6 نامیده شده بود جا داد. خوشبختانه این قابلیت‌های امنیتی طوری طراحی شده بودند که بتوانند هم در نسخهٔ فعلی IPv4 و هم در نسخهٔ آتی IPv6 قابل استفاده باشند. این بدین معنی است که فروشندگان محصولات کامپیوتری می‌توانند این مشخصه‌ها را در محصولات خود عرضه نمایند که هم اکنون بسیاری از آنها قابلیت‌های IPSec را در این محصولات گنجانده‌اند.

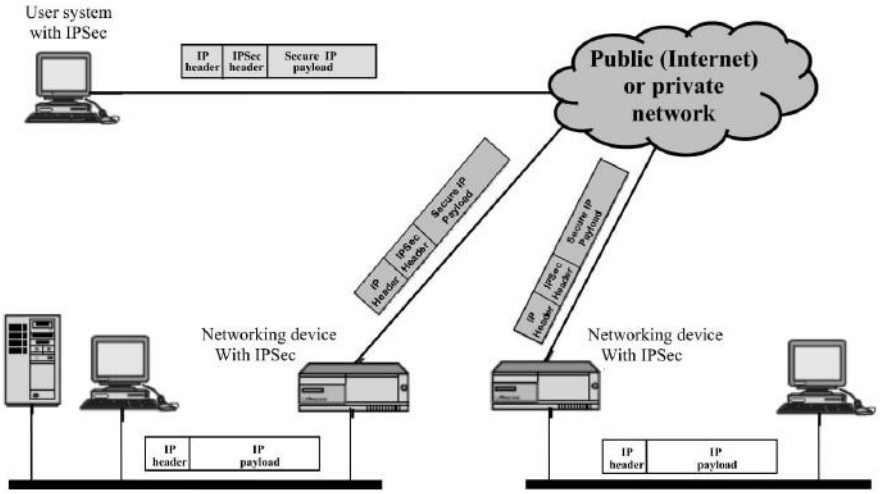
کاربردهای IPSec

IPSec قابلیت ارتباطات امن در عرض یک شبکهٔ LAN، در عرض شبکه‌های خصوصی و عمومی WAN و در عرض اینترنت را فراهم می‌آورد. مثال‌هائی از استفاده از IPSec بقرار زیر است:

- **اتصال امن شاخه‌های اداری از طریق اینترنت**: یک کمپانی می‌تواند یک شبکهٔ خصوصی مجازی امن را روی اینترنت و یا روی یک WAN عمومی بنا نماید. این امر باعث می‌شود تا این کسب وکار عمدتاً متکی به اینترنت بوده و نیاز آن به شبکه‌های خصوصی کمتر شود. در نتیجه هم هزینه و هم سربارهٔ مدیریت شبکه کاهش می‌یابد.
- **دست‌یابی امن به دوردست از طریق اینترنت**: یک کاربر انتهائی که سیستم او به پروتکل‌های امنیتی IP مجهز است می‌تواند به یک فراهم‌آورندهٔ سرویس اینترنتی(ISP) تلفن زده و به شبکهٔ یک کمپانی دسترسی یابد. این امر هزینهٔ تردد اداری کارمندان را کاهش خواهد داد.
- **برقراری ارتباط اینترانتی و اکسترانتی با شرکاء**: IPSec می‌تواند برای ایمن‌سازی ارتباطات با سایر سازمان‌ها بکار رود که اطمینان از اعتبارسنجی صحیح و محرمانگی از خواص آن بوده و مکانیسم مبادلهٔ کلید نیز در آن فراهم است.
- **ارتقاء امنیت تجارت الکترونیک**: با وجود اینکه تعدادی از کاربردهای مربوط به تجارت الکترونیک و وِب، در پروتکل‌های امنیتی فراهم شده در محصولات موجود می‌باشند، ولی استفاده از IPSec این امنیت را ارتقاء می‌بخشد.

مشخصهٔ اصلی IPSec که آن را قادر می‌سازد تا از این کاربردهای متنوع حمایت نماید این است که می‌تواند تمام ترافیک در سطح IP را رمزنگاری و/ یا اعتبارسنجی کند. بنابراین تمام کاربردهای توزیع شده که شامل اتصال از دور، کلاینت/ سِرور، e-mail، انتقال فایل، دسترسی به وِب و غیره‌اند می‌توانند امن باشند.

شکل ۱-۶ یک سناریوی معمول استفاده از IPSec را نشان می‌دهد. یک سازمان می‌تواند LAN های متفاوتی را در موقعیت‌های جغرافیائی مختلف داشته باشد. برای هر LAN، ترافیک غیرامن IP در نظر گرفته شده است ولی برای ترافیک خارج از شبکه‌ها که از طریق نوعی WAN خصوصی یا عمومی صورت می‌پذیرد از پروتکل‌های IPSec استفاده می‌شود. این پروتکل‌ها در تجهیزات شبکه همچون یک مسیریاب و یا یک دیوار آتش، که یکLAN را به دنیای خارج پیوند می‌دهند،کار می‌کنند. تجهیزات شبکه‌ای IPSec معمولاً تمام ترافیک داخل‌شونده به WAN را فشرده سازی و رمزنگاری نموده و تمام ترافیک خروجی از WAN و ورودی به LAN را از فشردگی درآورده و رمزگشائی می‌نماید. تمام این عملیات برای ایستگاه‌های کاری و سِرورهای LAN نامرئی هستند. ارسال اطلاعات بصورت امن همچنین برای کاربران منفردی که از طریق تماس تلفنی وارد WAN می‌شوند نیز ممکن است. این ایستگاه‌های کاری بایستی پروتکل‌های IPSec را برای فراهم آوردن امنیت در درون خود پیاده‌سازی کنند.



شکل ۱-۶ یک سناریو برای امنیت IP

مزایای IPSec

[MARK97] مزایای زیر را برای IPSec ذکر می‌کند:

- وقتی IPSec در یک دیوار آتش یا مسیریاب بکار گرفته می‌شود، یک امنیت محکم برای تمام ترافیکی که از محدودهٔ این دو دستگاه عبور می‌کند فراهم می‌آورد. ترافیک داخل سازمان یا یک گروه کاری، از بکارگیری سربارهٔ مرتبط با پردازش‌های امنیتی آزادند.
- اگر تمام ترافیک خارج از محدوده بایستی از IP استفاده کنند و دیوار آتش تنها راه ورودی اینترنت به سازمان باشد، IPSec در این دیوار آتش در برابر نادیده گرفته شدن و میان بُر زدن دیتا مقاوم است.
- IPSec در زیرِ لایهٔ حمل‌ونقل (UDP,TCP) قرار گرفته و بنابراین برای کاربردها نامرئی است. وقتی IPSec در یک مسیریاب یا دیوار آتش بکار گرفته می‌شود، نیازی به تعویض نرم‌افزارهای کاربران و یا سِرورها نیست. حتی اگر IPSec در سیستم انتهائی هم بکار گرفته شود نرم‌افزارهای لایه‌های بالاتر که شامل کاربردها هم هستند تحت تأثیر واقع نمی‌شوند.
- IPSec می‌تواند برای کاربران انتهائی نامرئی باشد. نیازی نیست که کاربران را نسبت به مکانیسم‌های امنیتی آموزش داد و مثلاً لازم نیست خلق اقلام کلید برای هر کاربر و یا ابطال اقلام کلید در هنگام ترک سازمان را به آنها آموخت.

- IPSec می‌تواند در صورت لزوم امنیت را برای تک‌تک کاربران فراهم آورد. این مورد برای کار در خارج از محل سازمان و همچنین برای ایجاد یک زیرشبکهٔ مجازی در درون سازمان برای کاربردهای حساس مناسب است.

کاربردهای مسیریابی

علاوه بر حمایت از کاربران انتهائی و محافظت از سیستم‌ها و شبکه‌ها، IPSec می‌تواند نقشی حیاتی در معماری مسیریابی مورد نیاز عملیات بین‌شبکه‌ای داشته باشد. [HUIT98] مثال‌های زیر استفاده از IPSec را لیست کرده است. IPSec اطمینان می‌دهد که:

- اعلان حضور یک مسیریاب (یک مسیریاب حضور خود را اعلان می‌کند)، از یک مسیریاب معتبر آمده است.
- اعلان حضور یک مسیریاب به همسایگان (یک مسیریاب به دنبال برقراری و یا نگهداری یک رابطهٔ همسایگی با مسیریاب دیگر است)، از یک مسیریاب معتبر آمده است.
- یک پیام تغییر مسیر از همان مسیریابی آمده است که بستهٔ اولیه دیتا برای او ارسال شده بود.
- بروزرسانی یک مسیریاب، جعلی نیست.

بدون چنین معیارهای امنیتی، یک دشمن می‌تواند ارتباطات را مختل کرده و یا مسیر ترافیک را عوض کند. پروتکل‌های مسیریابی، مانند OSPF، بایستی در بالای اتحادهای امنیتی تعریف شده بین مسیریاب‌ها بتوسط IPSec کار کنند.

۶-۲ معماری امنیت IP

مشخصه‌های IPSec بسیار پیچیده شده‌اند. برای اینکه درکی ازکلّ معماری IPSec حاصل شود، به اسنادی که IPSec را تعریف می‌کنند نگاهی می‌اندازیم. آنگاه سرویس‌های IPSec را تعریف کرده و مفهوم اتحاد امنیتی (SA) را معرفی می‌کنیم.

اسناد IPSec

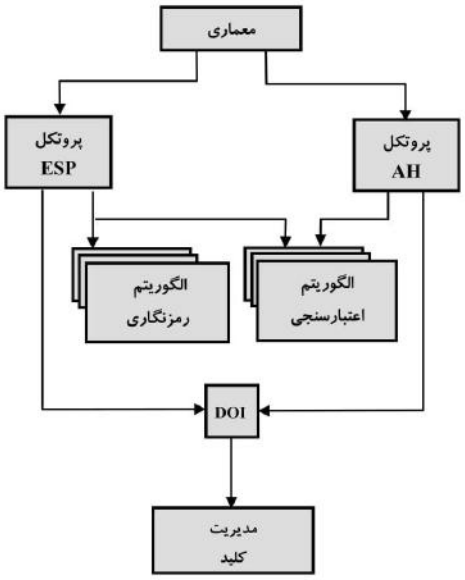
مشخصه‌های IPSec شامل اسناد متعددی است. مهم‌ترین آنها که در نوامبر ۱۹۹۸ میلادی منتشر شد، RFCهای 2401، 2402، 2406 و 2408 است:

- RFC 2401: مروری بر یک معماری امنیتی
- RFC 2402: توصیف یک الحاقیهٔ اعتبارسنجی بسته دیتا به IPv4 و IPv6
- RFC 2406: توصیف یک الحاقیهٔ رمزنگاری بستهٔ دیتا به IPv4 و IPv6
- RFC 2408: تعیین قابلیت‌های مدیریت کلید

تبعیت از این مشخصه‌ها برای IPv6 اجباری و برای IPv4 اختیاری است. در هردو مورد، مشخصه‌های امنیتی بصورت سرآیندهای الحاقی که بعد از سرآیند IP قرار می‌گیرند پیاده‌سازی می‌شوند. سرآیند الحاقی مربوط به اعتبارسنجی بنام سرآیند اعتبارسنجی (AH) و سرآیند الحاقی مربوط به رمزنگاری بنام سرآیندکپسولی‌کردن محمولهٔ امنیتی (ESP) نامیده می‌شود.

علاوه بر این چهار RFC، تعداد دیگری از پیش‌نویس‌ها از سوی IP Security Protocol Working Group که بتوسط IETF تأسیس شده است منتشر گردیده است. اسناد به هفت گروه، مطابق شکل ۲-۶ (RFC 2401) تقسیم شده‌اند:

- **معماری**: مفاهیم کلی، نیازهای امنیتی، تعاریف و مکانیسم‌هائی که تکنولوژی IPSec را تعریف می‌کنند، می‌پوشاند.
- **کپسولی کردن محمولهٔ امنیتی (ESP)**: فرمت بسته و مقوله‌های عمومی مرتبط با استفاده از ESP برای رمزنگاری بسته و اختیاراً اعتبارسنجی را توصیف می‌کند.
- **سرآیند اعتبارسنجی (AH)**: فرمت بسته و مقوله‌های عمومی مرتبط با استفاده از AH برای اعتبارسنجی بسته را توصیف می‌کند.
- **الگوریتم رمزنگاری**: مجموعه‌ای از اسناد که توصیف می‌نمایند چگونه الگوریتم‌های مختلف رمزنگاری برای ESP بکار می‌روند.
- **الگوریتم اعتبارسنجی**: مجموعه‌ای از اسناد که توصیف می‌نمایند چگونه الگوریتم‌های مختلف اعتبارسنجی برای AH و ESP بکار می‌روند.
- **مدیریت کلید**: اسنادی که روش‌های مدیریت کلید را توصیف می‌کنند.
- **محدودهٔ تعبیر (DOI)**: شامل اندازه‌های لازم برای سایر اسناد جهت مرتبط نمودن آنها به یکدیگر است. این شامل شناسه‌های رمزنگاری‌های معتبر و الگوریتم‌های اعتبارسنجی و همچنین پارامترهای اختیاری همچون طول عمر یک کلید است.



شکل ۲-۶ مروری بر اسناد IPSec

سرویس‌های IPSec

IPSec سرویس‌های امنیتی در سطح IP را بنحوی فراهم می‌آورد که سیستم قادر است تا پروتکل‌های امنیتی لازم را انتخاب کرده، الگوریتم(های) لازم برای سرویس(ها) را تعیین نموده و کلیدهای رمزنگاری لازم برای سرویس‌های درخواست شده را در محل مناسب قرار دهد. دو پروتکل برای ایجاد امنیت بکار می‌رود: یک پروتکل اعتبارسنجی که بتوسط سرآیند پروتکل یعنی Authentication Header (AH) شناسائی شده و یک پروتکل مخلوط رمزنگاری/ اعتبارسنجی که بتوسط فرمت بستهٔ آن پروتکل Encapsulating Security Payload (ESP) شناسائی می‌گردد. سرویس‌ها بقرار زیراند:

- کنترل دست‌یابی
- صحت دیتا در حالت غیراتصالی
- اعتبارسنجی مبدأ دیتا
- رد بسته‌های بازخوانی شده
- محرمانگی (رمزنگاری)
- محرمانگی محدود جریان ترافیک

جدول ۱-۶ نشان می‌دهد که کدام سرویس‌ها بتوسط پروتکل‌های AH و ESP ایجاد می‌شوند. برای ESP دو حالت وجود دارد: حضور و یا عدم حضور اعتبارسنجی بصورت اختیاری. AH و ESP هر دو محمل‌هائی برای کنترل دست‌یابی‌اند که مبتنی بر توزیع کلیدهای رمزنگاری و مدیریت جریان‌های ترافیک مرتبط با این پروتکل‌های امنیتی می‌باشند.

اتحادهای امنیتی (Security Associations)

یک مفهوم کلیدی که در هردو مکانیسم اعتبارسنجی و محرمانگی IP ظاهر می‌شود، یک اتحاد امنیتی (SA) است. یک اتحاد یک رابطهٔ یک- طرفه بین یک فرستنده و یک گیرنده است که سرویس‌های امنیتی را برای ترافیک حمل شده روی آن فراهم می‌کند. اگر یک رابطهٔ نظیر نیز مورد نیاز باشد آنگاه برای مبادلهٔ امن دوطرفه، دو اتحاد امنیتی لازم است. سرویس‌های امنیتی فقط برای استفاده از AH یا ESP، و نه هردوی آنها، به یک SA داده می‌شود.

جدول ۱-۶ سرویس‌های IPSec

	AH	ESP (فقط رمزنگاری)	ESP (رمزنگاری بعلاوه اعتبارسنجی)
کنترل دست‌یابی	✓	✓	✓
صحت دیتا در حالت غیراتصالی	✓		✓
اعتبارسنجی منبع دیتا	✓		✓
رد بسته‌های بازخوانی شده	✓	✓	✓
محرمانگی دیتا		✓	✓
محرمانگی محدود جریان ترافیک		✓	

یک اتحاد امنیتی بطور یکتا با سه پارامتر تعیین می‌گردد:

- **شاخص پارامترهای امنیتی (SPI)**: دنباله‌ای از بیت‌ها که به این SA اختصاص داده شده و فقط اهمیت محلی دارد. SPI در سرآیندهای AH و ESP حمل شده تا سیستم گیرنده را قادر سازد تا یک SA که تحت آن یک بستهٔ دریافتی مورد پردازش قرار می‌گیرد را انتخاب کند.
- **آدرس IP مقصد**: در حال حاضر تنها آدرس‌های unicast مجاز است. این آدرس نقطهٔ انتهائی مقصد SA است که ممکن است یک سیستم انتهائی و یا یک سیستم شبکه مثل یک دیوار آتش و یا یک مسیریاب باشد.
- **شناسهٔ پروتکل امنیتی**: نمایشگر این است که آیا اتحاد، یک اتحاد AH و یا یک اتحاد ESP است.

بنابراین در هر بستهٔ IP، اتحاد امنیتی بطور یکتا بتوسط Destination Address در سرآیند IPv4 یا IPv6 و SPI در سرآیند الحاقی (AH یا ESP) مشخص می‌گردد.

پارامترهای SA

در هر پیاده‌سازی IPSec، یک پایگاه داده اتحاد امنیتی (Security Association Database) وجود دارد که پارامترهای مرتبط با هر SA را تعریف می‌کند. یک اتحاد امنیتی معمولاً با پارامترهای زیر تعریف می‌شود:

- **کنتر شمارهٔ ردیف**: یک اندازهٔ ۳۲- بیتی که برای تولید میدان Sequence Number سرآیندهای AH یا ESP بکار می‌رود و در بخش ۳-۶ تعریف شده است (برای همهٔ پیاده‌سازی‌ها لازم است).
- **سرریز کنتر شمارهٔ ردیف**: یک پرچم که نشان‌دهندهٔ این است که آیا سرریز کنتر شمارهٔ ردیف‌ها بایستی یک پیشامد قابل ممیزی را تولید کرده و از انتقال بیشتر بسته‌ها در این SA جلوگیری نماید (برای همهٔ پیاده‌سازی‌ها لازم است).
- **پنجرهٔ ضد- بازخوانی**: برای تعیین اینکه آیا یک بستهٔ AH یا ESP یک بازخوانی است یا نه، که در بخش ۳-۶ توضیح داده شده است (برای همهٔ پیاده‌سازی‌ها لازم است).
- **اطلاعات AH**: الگوریتم اعتبارسنجی، کلیدها، طول عمر کلیدها و پارامترهای مرتبطی که با AH بکار می‌رود (مورد نیاز پیاده‌سازی‌های AH).
- **اطلاعات ESP**: الگوریتم‌های رمزنگاری و اعتبارسنجی، کلیدها، مقادیر اولیه، طول عمر کلیدها و پارامترهای مرتبطی که با ESP بکار می‌رود (مورد نیاز پیاده‌سازی‌های ESP).
- **طول عمر این اتحاد امنیتی**: یک طول زمانی یا شمارش بایت که بعد از آن، این SA بایستی با یک SA جدید (و SPI جدید) تعویض شده و یا خاتمه یابد بعلاوهٔ نمایشگری برای اینکه نشان دهد کدامیک از این دو عمل بایستی واقع شود (برای همهٔ پیاده‌سازی‌ها لازم است).
- **مُود پروتکل IPSec**: Tunnel، transport و یا wildcard (برای همهٔ پیاده‌سازی‌ها لازم است). این مُودها بعداً در همین بخش توضیح داده می‌شوند.

- **MTU مسیر**: ماکزیمم واحد انتقال مشاهده شده در مسیر (اندازهٔ ماکزیمم بسته‌ای که می‌تواند بدون قطعه- قطعه شدن انتقال یابد) و متغیرهای نمایش طول عمر (برای همهٔ پیاده‌سازی‌ها لازم است).

مکانیسم مدیریت کلید که برای توزیع کلیدها بکار می‌رود با مکانیسم‌های اعتبارسنجی و محرمانگی تنها از طریق SPI مرتبط است. بنابراین اعتبارسنجی و محرمانگی مستقل از هرنوع مکانیسم مدیریت کلید خاصی تعریف شده‌اند.

انتخاب‌کننده‌های SA

IPSec انعطاف‌پذیری قابل ملاحظه‌ای را، در انتخاب اینکه کدام سرویس‌های IPSec به ترافیک IP اعمال شوند، برای کاربر ایجاد می‌کند. همانطور که بعداً خواهیم دید، SAها می‌توانند به روش‌های مختلف ترکیب شده و پیکربندی مناسب کاربر را ایجاد کنند. علاوه برآن IPSec درجهٔ بالائی از تشخیص برای تمایز بین ترافیکی که IPSec به آن اعمال شده با ترافیکی که می‌تواند IPSec را دور بزند ایجاد نموده که مورد اول ترافیک IP را به SAهای بخصوص پیوند می‌دهد.

ابزاری که ترافیک IP را به SAهای مشخص (یا نبود SA در مورد ترافیکی که می‌تواند IPSec را دور بزند) مرتبط می‌سازد، پایگاه دادهٔ خط‌مشی امنیتی Security Policy Database (SPD) است. در ساده‌ترین فرم خود، یک SPD شامل اقلامی است که هریک از آنها یک زیرمجموعه از ترافیک IP را تعیین کرده و به یک SA برای آن ترافیک اشاره می‌کند. در محیط‌های پیچیده‌تر، ممکن است اقلام متعددی وجود داشته باشند که بالقوه مرتبط با یک SA منفرد یا SAهای متعدد نظیر یک SPD منفرد باشند. خواننده در صورت نیاز بایستی به اسناد IPSec مراجعه نماید.

هر SPD بتوسط یک مجموعه از اندازهٔ میدان‌های پروتکل IP و لایهٔ بالاتر بنام *انتخاب‌کننده‌ها (selectors)* تعریف می‌شود. در واقع این انتخاب کننده‌ها، برای فیلترکردن ترافیک خروجی بمنظور نگاشت آنها به یک SA بخصوص استفاده می‌شوند. پردازش داده‌های خارج شونده، از مراحل عمومی زیر برای هر بستهٔ IP تبعیت می‌کند:

۱- اندازهٔ میدان‌های مرتبط در بسته (میدان‌های selector) را با SPD مقایسه کرده تا یک تطبیق پیدا شود که به هیچ و یا چند SA اشاره نماید.

۲- اگر SA برای این بسته موجود است آن را تعیین کرده و SPI مرتبط با آن را استخراج کند.

۳- پردازش لازم IPSec را انجام دهد (یعنی پردازش ESP یا AH).

انتخاب‌کننده‌های زیر تعیین‌کنندهٔ یک قلم موجود در SPD هستند:

- **آدرس IP مقصد**: این می‌تواند یک آدرس IP منفرد، محدوده‌ای از آدرس‌ها و یا یک آدرس عام (mask) باشد. دوتای آخر از این جهت مورد نیازند که بیش از یک سیستم مقصد با SA یکسان را حمایت کنند (مثل پشت یک دیوار آتش).

- **آدرس IP منبع**: این می‌تواند یک آدرس IP منفرد، محدوده‌ای از آدرس‌ها و یا یک آدرس عام باشد. دوتای آخر از این جهت مورد نیازند که بیش از یک سیستم منبع با SA یکسان را حمایت کنند (مثل پشت یک دیوار آتش).
- **شمارهٔ شناسائی کاربر**: یک شناسهٔ کاربر(UserID) در سیستم عامل. این یک میدان در سرآیندIP و یا سرآیندهای لایهٔ بالاتر نیست بلکه فقط در صورتی وجود دارد که IPSec روی همان سیستم عامل که کاربر به آن وصل است کار کند.
- **سطح امنیتی دیتا**: برای سیستم‌هائی که امنیت جریان اطلاعات را فراهم می‌کنند بکار می‌رود (مثلاً سرّی یا طبقه‌بندی نشده).
- **پروتکل لایهٔ حمل‌ونقل**: از پروتکل IPv4 و یا میدان IPv6 Next Header بدست می‌آید. این ممکن است شمارهٔ یک پروتکل منفرد، لیستی از شمارهٔ پروتکل‌ها و یا محدوده‌ای از شمارهٔ پروتکل‌ها باشد.
- **پورت‌های منبع و مقصد**: اینها ممکن است اندازهٔ یک پورت منفرد TCP یا UDP، لیستی از پورت‌های مختلف و یا یک پورت عام باشند.

مُودهای حمل‌ونقل و تونل

هم AH و هم ESP دو مُود استفاده دارند: مُود حمل‌ونقل (transport) و مُود تونل(tunnel). عملکرد این دو مُود به بهترین نحو پس از توصیف AH و ESP روشن خواهد شد که در بخش‌های ۳-۶ و ۴-۶ به آن خواهیم پرداخت. فعلاً مروری کوتاه بر آنها ارائه می‌دهیم.

مُود حمل‌ونقل

مُود حمل‌ونقل حفاظت را عمدتاً برای پروتکل‌های لایهٔ بالاتر فراهم می‌آورد. یعنی حفاظت مُود حمل‌ونقل به محمولهٔ بستهٔ IP اعمال می‌شود. مثال‌هائی از این دست یک سگمنت TCP یا UDP و یا یک بستهٔ ICMP است که همهٔ آنها مستقیماً در بالای IP کار می‌کنند. معمولاً مُود حمل‌ونقل برای ارتباطات سر- به- سر بین دو میزبان بکار می‌رود (مثلاً یک کلاینت و یک سِرور و یا دو ایستگاه کاری). وقتی یک میزبان، AH یا ESP را روی IPv4 اجرا می‌کند، محموله همان دیتائی است که بطور نرمال بعد از سرآیند IP قرار می‌گیرد. برای IPv6، محموله دیتائی است که معمولاً بعد از سرآیند IP و هر سرآیند الحاقی موجود دیگر قرار دارد بجز حالت استثنائی سرآیند option که ممکن است در بخش حفاظت شده قرارگیرد.

ESP در مُود حمل‌ونقل، محمولهٔ IP و نه سرآیند IP را، رمزنگاری و بطور اختیاری اعتبارسنجی می‌نماید. AH در مُود حمل‌ونقل محمولهٔ IP و بخش‌های انتخاب شده‌ای از سرآیند IP را رمزنگاری می‌کند.

مُود تونل

مُود تونل حفاظت را برای تمام بستهٔ IP ایجاد می‌کند. برای این منظور پس از اینکه میدان‌های AH یا ESP به بستهٔ IP اضافه شدند، تمام بسته باضافهٔ میدان‌های امنیتی بصورت محمولهٔ یک بستهٔ IP جدید «بیرونی‌تر» با سرآیند IP جدید

درخواهند آمد. تمام بستهٔ اولیه یا درونی از درون یک «تونل» از یک نقطه شبکهٔ IP به نقطهٔ دیگر حرکت کرده و هیچ مسیریابی در مسیر آن قادر نیست سرآیند IP درونی را بررسی کند. چون بستهٔ اولیه کپسولی شده است، بستهٔ جدیدتر و بزرگتر ممکن است دارای آدرس‌های مبدأ و مقصد کاملاً متفاوت باشند که این خود به امنیت می‌افزاید. مُود تونل وقتی استفاده می‌شود که یک یا هردو انتهای SA یک دروازهٔ امنیتی همچون دیوار آتش یا مسیریابی باشد که IPSec را بکار می‌گیرد. در مُود تونل، تعدادی از میزبانان روی شبکه و پشت دیوار آتش می‌توانند بدون پیاده‌سازی IPSec ، ارتباطات امن داشته باشند. بسته‌های حفاظت نشده که از طرف چنین میزبان‌هائی تولید می‌شوند از درون شبکه‌های خارجی بتوسط SAهای مُود تونل که بتوسط نرم‌افزار IPSec در دیوار آتش و یا مسیریاب‌های امن در مرزهای شبکه فراهم گشته‌اند، تونل می‌شوند.

در اینجا مثالی از اینکه مُود تونل IPSec چطور کار می‌کند، ارائه می‌دهیم. میزبان A روی یک شبکه، یک بستهٔ IP با آدرس مقصد میزبان B روی شبکهٔ دیگری را تولید می‌کند. این بسته از میزبان مبدأ به یک دیوار آتش یا مسیریاب امن در مرز شبکهٔ A می‌رود. دیوار آتش تمام بسته‌های خروجی را فیلتر کرده تا نیاز به پردازش IPSec را تعیین کند. اگر این بسته از A به B نیاز به IPSec داشته باشد، دیوار آتش پردازش IPSec را انجام داده و بسته را با یک سرآیند IP بیرونی کپسولی می‌نماید. آدرس IP منبع این بسته IP بیرونی، این دیوار آتش بوده و آدرس مقصد ممکن است دیوار آتشی باشد که مرز شبکهٔ محلی B را تشکیل می‌دهد. حالا این بسته به سمت دیوار آتش B مسیریابی می‌شود و مسیریاب‌های وسط راه فقط سرآیند IP بیرونی را وارسی می‌کنند. در دیوار آتش B، سرآیند IP بیرونی کنده می‌شود و بسته درونی به B تحویل می‌گردد.

ESP در مُود تونل، تمام بستهٔ IP درونی که شامل سرآیند IP درونی نیز می‌شود را رمزنگاری و بطور اختیاری اعتبارسنجی می‌نماید. AH در مُود تونل تمام بستهٔ IP درونی و بخش‌های انتخاب‌شده‌ای از سرآیند IP بیرونی را اعتبارسنجی می‌نماید.

جدول ۲-۶ عملکرد مُودهای حمل‌ونقل و تونل را خلاصه کرده است.

۶-۳ سرآیند اعتبارسنجی (Authentication Header)

سرآیند اعتبارسنجی وظیفهٔ اطمینان از صحت دیتا و اعتبارسنجی بسته‌های IP را بعهده دارد. خاصیت مربوط به صحت دیتا این اطمینان را ایجاد می‌کند که دخل تصرف تشخیص داده نشده در محتویات بسته‌های در حال ترانزیت غیرممکن است. خاصیت اعتبارسنجی، یک سیستم یا یک دستگاه متصل به شبکه را قادر می‌سازد تا هویت یک کاربر و یا یک کاربرد را بررسی کرده و ترافیک را بر اساس آن فیلتر کند. اعتبارسنجی همچنین از حملات تقلیدآدرس(spoofing) که امروزه در اینترنت مشاهده می‌شود جلوگیری می‌کند. AH همچنین در برابر حملات بازخوانی (replay) ایجاد مصونیت می‌نماید.

اعتبارسنجی با استفاده از کُد اعتبارسنجی پیام (MAC) صورت می‌پذیرد که همانطور که قبلاً در مورد آن بحث شده است نیاز به اشتراک گذاردن یک کلید سّری بین طرفین ارتباط دارد.

سرآیند اعتبارسنجی شامل میدان‌های زیر است (شکل ۳-۶) :

- **Next Header (8 bits)**، نوع سرآیندی که بلافاصله پس از این سرآیند قرار می‌گیرد را مشخص می‌کند.

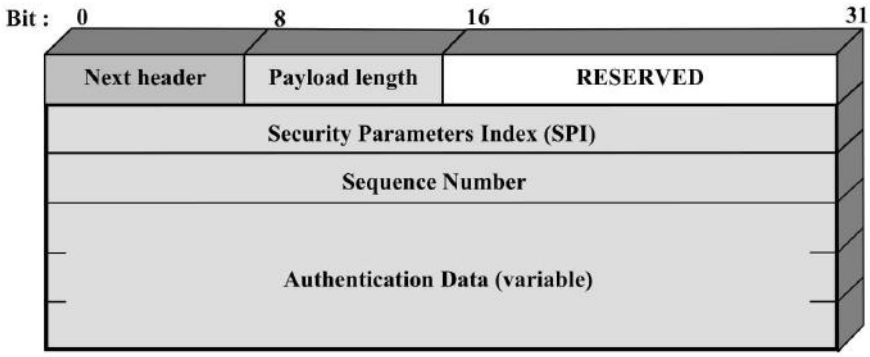
جدول ۲-۶ عملکرد مُود حمل‌ونقل و مُود تونل

	Transport Mode SA	Tunnel Mode SA
AH	محمولهٔ IP و بخش‌های انتخاب شده‌ای از سرآیند IP و سرآیندهای الحاقی IPv6 را اعتبارسنجی می‌نماید.	تمام بستهٔ IP درونی (سرآیند درونی بعلاوهٔ محمولهٔ IP) باضافهٔ قسمت‌های انتخاب‌شده‌ای از سرآیند IP بیرونی و سرآیندهای الحاقی IPv6 بیرونی را اعتبارسنجی می‌نماید.
ESP	محمولهٔ IP و هر سرآیند الحاقی IPv6 که بعد از سرآیند ESP قرار دارد را رمزنگاری می‌کند.	بستهٔ IP درونی را رمزنگاری می‌کند.
ESP با اعتبارسنجی	محمولهٔ IP و هر سرآیند الحاقی IPv6 که بعد از سرآیند ESP قرار دارد را رمزنگاری می‌کند. محمولهٔ IP و نه سرآیند IP را اعتبارسنجی می‌کند.	بستهٔ IP درونی را رمزنگاری می‌کند. بستهٔ IP درونی را اعتبارسنجی می‌کند.

- **Payload Length (8 bits)**: طول سرآیند اعتبارسنجی برحسب کلمات ۳۲- بیتی منهای ۲. بعنوان مثال، طول پیش فرض میدان اعتبارسنجی دیتا ۹۶ بیت و یا ۳ کلمهٔ ۳۲- بیتی است. با یک سرآیند ثابت سه کلمه‌ای، شش کلمه در سرآیند وجود خواهد داشت و اندازهٔ میدان Payload Length برابر ۴ خواهد بود.
- **Reserved (16 bits)**: برای مصارف آینده رزرو شده است.
- **Security Parameters Index (32 bits)**: یک اتحاد امنیتی را مشخص می‌کند.
- **Sequence Number (32 bits)**: یک شمارنده که اندازهٔ آن بطور یکنواخت زیاد می‌شود و بعداً مورد بحث قرار خواهد گرفت.
- **Authentication Data (variable)**: یک میدان با طول متغیر (که بایستی مضرب صحیحی از کلمات ۳۲- بیتی باشد) که شامل Integrity Check Value (ICV) ،یا MAC، برای این بسته است و بعداً در مورد آن صحبت خواهیم کرد.

سرویس ضد- بازخوانی (Anti-Replay Sevice)

یک حملهٔ بازخوانی چنین است که در آن حمله‌کننده یک نسخه از بستهٔ اعتبارسنجی شده را به دست آورده و بعداً آن را برای مقصد مورد نظر ارسال می‌دارد. دریافت مجدد بسته‌های اعتبارسنجی شده در مقصد، ممکن است سرویس را بنحوی مختل کرده و یا نتایج نامطلوب دیگری به دنبال داشته باشد. میدان Sequence Number برای مقابله با چنین حملاتی طراحی شده است. ابتدا نحوهٔ تولید شماره ردیف بتوسط فرستنده را مورد بحث قرار داده و سپس خواهیم دید که این میدان چگونه بتوسط گیرنده مورد پردازش قرار می‌گیرد.



شکل ۳-۶ سرآیند اعتبارسنجی (AH) در IPSec

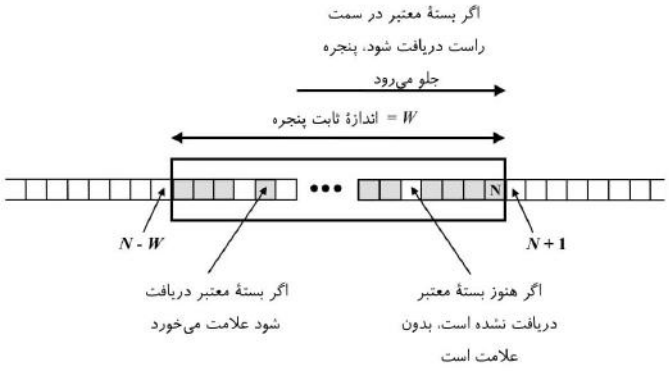
زمانی که یک SA جدید برپا می‌شود، **فرستنده** کنتر شمارهٔ ردیف را روی 0 تنظیم می‌کند. هربار که یک بسته روی این SA ارسال می‌گردد، فرستنده کنتر را یک واحد افزایش داده و اندازهٔ آن را در میدان Sequence Number قرار می‌دهد. بنابراین اولین اندازه‌ای که استفاده می‌شود 1 است. اگر سرویس ضد- بازخوانی فعال باشد (پیش فرض)، فرستنده نبایستی اجازه دهد تا شمارهٔ ردیف پس از عبور از $۲^{۳۲}-۱$ به صفر برگردد، در غیر اینصورت بسته‌های متعددی با شماره ردیف یکسان وجود خواهند داشت. اگر مرز $۲^{۳۲}-۱$ فرا رسد، فرستنده بایستی این SA را خاتمه داده و SA جدیدی با یک کلید جدید را با گیرنده تشکیل دهد.

چون IP یک سرویس غیراتصالی و غیرقابل اعتماد است، پروتکل تضمینی در برابر تحویل منظم بسته‌ها و همچنین تحویل تمام بسته‌ها ندارد. بنابراین اسناد اعتبارسنجی IPSec چنین دیکته می‌کند که **گیرنده بایستی** پنجره‌ای با اندازهٔ W را ایجاد نماید که پیش فرض آن $W = ۶۴$ است. لبهٔ سمت راست پنجره، بالاترین شماره ردیف N، مربوط به آخرین بستهٔ معتبر دریافت شده، را نشان می‌دهد. برای هر بسته‌ای با شمارهٔ ردیفی در محدودهٔ $N-W+1$ تا N که بطور صحیح دریافت شده است (یعنی اعتبار آن سنجیده شده است)، شیار نظیر آن در پنجره علامت می‌خورد (شکل ۴-۶). پس از دریافت یک بسته، یک پردازش بشکل زیر روی آن انجام می‌شود:

۱- اگر بستهٔ دریافت شده در داخل پنجره قرار داشته و جدید باشد، اندازهٔ MAC کنترل می‌شود. اگر بسته معتبر باشد، شیار نظیر آن در پنجره علامت‌گذاری می‌شود.

۲- اگر بستهٔ دریافت شده در سمت راست پنجره قرار داشته و جدید باشد، اندازهٔ MAC کنترل می‌شود. اگر بسته معتبر باشد، پنجره جلو می‌رود بنحوی که این شماره ردیف لبهٔ سمت راست پنجره را تشکیل دهد و شیار نظیرآن در پنجره علامت‌گذاری می‌شود.

۳- اگر بستهٔ دریافت شده در سمت چپ پنجره واقع باشد و یا اعتبارسنجی با شکست مواجه شود، بسته نابود شده و این یک پیشامد قابل ثبت و ممیزی است.



شکل ۴-۶ مکانیسم ضد- بازخوانی

اندازهٔ کنترل صحت (Integrity Check Value)

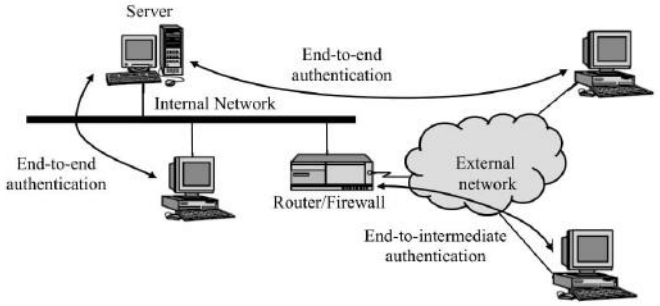
میدان Authentication Data اندازه‌ای را نگاه می‌دارد که به آن اندازهٔ کنترل صحت Integrity Check Value (ICV) گویند. ICV یک کُد اعتبارسنجی پیام و یا فرم مقطّعی از این کُد است که بتوسط الگوریتم MAC تولید می‌شود. مشخصه‌های فعلی، پیاده‌سازی و حمایت از دو الگوریتم زیر را دیکته می‌کنند:

- HMAC-MD5-96
- HMAC-SHA-1-96

هردوی اینها از الگوریتم HMAC استفاده می‌کنند که اولی کُد درهم‌ساز MD5 و دومی کُد درهم‌ساز SHA-1 را بکار می‌برد (تمام این الگوریتم‌ها در فصل ۳ مورد بحث قرار گرفته‌اند). در هردو حالت، اندازهٔ کامل HMAC محاسبه شده ولی بعداً بریده شده و تنها ۹۶ بیت اول آن مورد استفاده قرار می‌گیرد که طول پیش فرض میدان Authentication Data است.

MAC روی اقلام زیر محاسبه می‌شود:

- میدان‌های سرآیند IP که یا در هنگام ترانزیت تغییر نکرده‌اند(immutable) و یا اندازهٔ آنها در هنگام ورود به نقطهٔ انتهائی برای AH SA قابل پیش‌بینی است. میدان‌هائی که ممکن است در حال ترانزیت تغییر کرده و یا اندازهٔ آنها در هنگام ورود غیرقابل پیش‌بینی است بمنظور محاسبه در مبدأ و مقصد صفر منظور می‌شوند.
- سرآیند AH به غیر از میدان Authentication Data. میدان Authentication Data برای محاسبه در مبدأ و مقصد صفر منظور می‌شود.
- تمام دیتای پروتکل لایهٔ بالاتر که در هنگام ترانزیت تغییرناپذیر فرض شده‌اند(مثلاً یک سگمنت TCP و یا یک بستهٔ IP درونی در مُود تونل).



شکل ۵-۶ اعتبارسنجی End-to-End در برابر اعتبارسنجی End-to-Intermediate

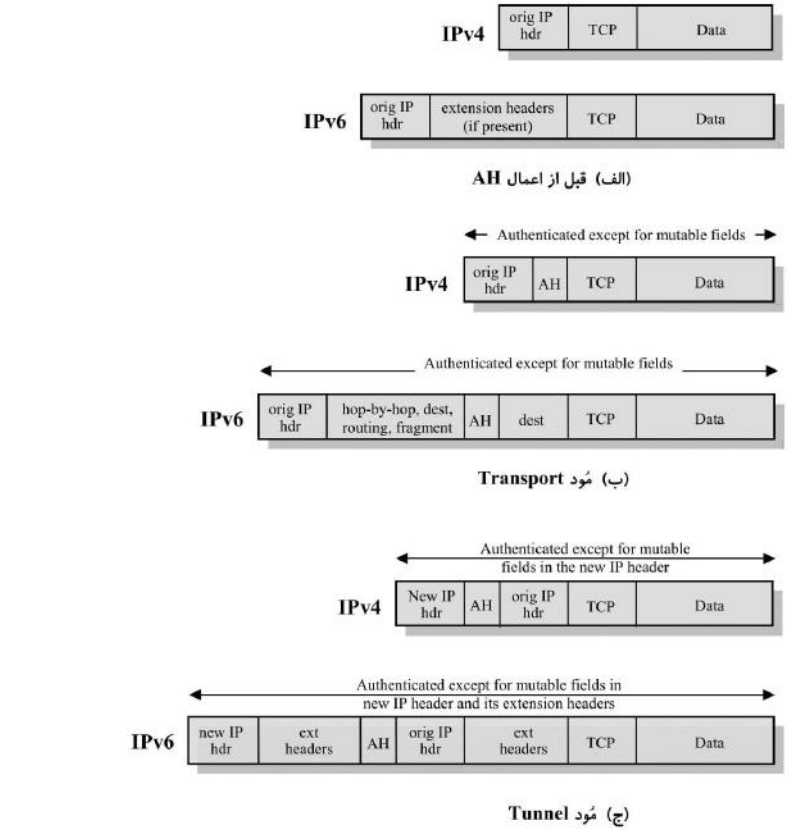
برای IPv4، مثال‌هائی از میدان‌های تغییرناپذیر Internet Header Length و Source Address هستند. مثالی از یک میدان تغییرپذیر ولی قابل پیش‌بینی Destination Address است (با مسیریابی منبع). مثال‌هائی از میدان‌های تغییرپذیر که قبل از محاسبات ICV صفر هستند، میدان‌های Time to Live و Header Checksum هستند. توجه کنید که هر دو میدان آدرس منبع و مقصد حفاظت شده‌اند بطوری که از جعل آدرس جلوگیری می‌شود.

برای IPv6، مثال‌هائی در سرآیند اصلی، Version (تغییرناپذیر)، Destination Address (تغییرپذیر ولی قابل پیش‌بینی) و Flow Label (تغییرپذیر و برای محاسبات برابر صفر) می‌باشند.

مُودهای حمل‌ونقل و تونل

شکل ۵-۶ دو حالت که در آنها سرویس اعتبارسنجی IPSec می‌تواند مورد استفاده قرار گیرد را نشان می‌دهد. در یک حالت، اعتبارسنجی مستقیماً بین یک سِرور و ایستگاه کاری کلاینت انجام می‌شود که ایستگاه کاری می‌تواند روی همان شبکهٔ سِرور یا روی یک شبکهٔ خارجی باشد. تا زمانی که ایستگاه کاری و سِرور یک کلید سّری حفاظت شده را در اشتراک دارند، عمل اعتبارسنجی امن است. این مورد از یک SA در مُود حمل‌ونقل استفاده می‌کند. در حالت دیگر، یک ایستگاه کاری دور هویت خود را برای دیوار آتش همان سیستم مشخص می‌نماید، یا برای اینکه به تمام شبکهٔ داخلی دسترسی یابد و یا بدلیل اینکه سِرور درخواست شده اعتبارسنجی را حمایت نمی‌کند. این مورد از یک SA در مُود تونل استفاده می‌کند.

در این قسمت به قلمرو اعتبارسنجی ایجادشده بتوسط AH و محل قرارگرفتن سرآیند اعتبارسنجی برای این دو مُود نگاهی می‌اندازیم. ملاحظات برای IPv4 و IPv6 قدری متفاوت‌اند. شکل ۶-۶الف بسته‌های معمول IPv4 و IPv6 را نشان می‌دهد. در این شکل محمولهٔ IP یک سِگمنت TCP است. این محموله می‌تواند یک واحد دیتا برای هر پروتکل دیگری مانند UDP و یا ICMP باشد که مستقیماً از IP استفاده می‌کند.



شکل ۶-۶ افق دید اعتبارسنجی AH

برای **AH مُود حمل‌ونقل** که از IPv4 استفاده می‌کند، AH بعد از سرآیند معمول IP و قبل از محمولهٔ IP (مثلاً یک سگمنت TCP)، قرار می‌گیرد که این امر در بخش بالائی شکل ۶-۶ب نشان داده شده است. اعتبارسنجی، تمام بسته بجز میدان‌های تغییرپذیر در سرآیند IPv4 که برای محاسبات MAC مساوی صفر قرار داده می‌شوند، را در بر می‌گیرد.

در مقولهٔ IPv6، AH بعنوان یک محمولهٔ سر- به- سر تلقی می‌گردد، یعنی نه بتوسط مسیریاب‌های میانی مورد بازبینی قرار گرفته و نه پردازشی روی آن صورت می‌پذیرد. بنابراین AH بعد از سرآیند اصلی IPv6 و سرآیندهای الحاقی hop-by-hop ، routing و fragment قرار می‌گیرد. سرآیندهای الحاقی اختیاری مربوط به مقصد می‌توانند قبل و یا بعد از سرآیند AH قرار گیرند که بستگی به منطق مورد استفاده دارد. بازهم اعتبارسنجی تمام بسته، بجز میدان‌های تغییرپذیر که برای محاسبات MAC برابر صفر قرار می‌گیرند، را پوشش می‌دهد.

برای **AH مُود تونل**، تمام بستهٔ IP اولیه اعتبارسنجی می‌شود و AH بین سرآیند اولیهٔ IP و سرآیند جدید بیرونی IP (شکل ۶-۶ج) وارد می‌گردد. سرآیند IP درونی، آدرس‌های مبدأ و مقصد نهائی را مشخص می‌کند در حالی که یک سرآیند IP بیرونی می‌تواند شامل آدرس‌های IP متفاوت باشد(مثلاً آدرس یک دیوار آتش و یا دروازه‌های امنیتی دیگر).

در مُود تونل تمام بستهٔ IP درونی، که شامل کل سرآیند IP درونی نیز هست، بتوسط AH محافظت می‌شود. سرآیند IP بیرونی (و در مورد IPv6 سرآیندهای الحاقی IP بیرونی)، بجز میدان‌های تغییرپذیر و غیرقابل پیش‌بینی، نیز در محدودهٔ حفاظتی قرار دارند.

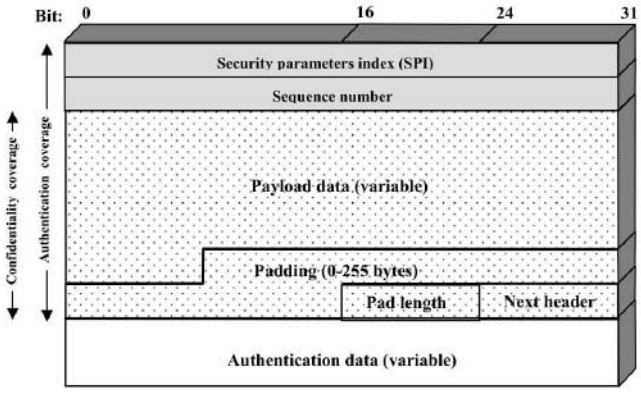
۶-۴ کپسولی‌کردن محمولهٔ امنیتی (Encapsulating Security Payload)

کپسولی‌کردن محمولهٔ امنیتی (ESP)، یک سرویس محرمانگی را فراهم می‌آورد که شامل محرمانگی محتویات پیام و محرمانگی محدود جریان ترافیک است. بصورت اختیاری، ESP می‌تواند یک سرویس اعتبارسنجی را نیز فراهم آورد.

فرمت ESP

شکل ۷-۶ فرمت بستهٔ ESP را نشان می‌دهد. این بسته شامل میدان‌های زیر است:

- **Security Parameters Index (32 bits)**: یک اتحاد امنیتی را مشخص می‌کند.
- **Sequence Number (32 bits)**: اندازهٔ یک شمارنده است که بطور یکنواخت اضافه می‌شود. این امر برای محافظت در برابر حملات بازخوانی، همانطور که در AH بحث شد، مورد استفاده قرار می‌گیرد.
- **Payload Data (variable)**: این یک سِگمنت سطح حمل‌ونقل و یا یک بستهٔ IP مُود تونل است که با رمزنگاری محافظت گردیده است.



شکل ۷-۶ فرمت ESP در IPSec

- **Padding (0-255 bytes)**: هدف این میدان بعداً تعریف خواهد شد.
- **Pad Length (8 bits)**: نمایش‌دهندهٔ تعداد بایت‌های لایی است که درست قبل از این میدان قرار دارند.
- **Next Header (8 bits)**: مشخص‌کنندهٔ نوع داده‌های موجود در میدان Payload Data است که بتوسط اولین سرآیندآن محموله تعیین می‌گردد (بعنوان مثال یک سرآیند الحاقی در IPv6 و یا یک پروتکل لایهٔ بالاتر شبیه TCP).
- **Authentication Data (variable)**: یک میدان با طول متغیر (بایستی مضربی از کلمات ۳۲- بیتی باشد) که شامل Integrity Check Value است که روی بستهٔ ESP منهای میدان Authentication Data محاسبه شده است.

الگوریتم‌های رمزنگاری و اعتبارسنجی

میدان‌های Payload Data، Padding، Pad Length و Next Header بتوسط سرویس ESP رمزنگاری می‌شوند. اگر الگوریتم بکار گرفته شده برای رمزنگاری محموله نیاز به داده‌هائی برای همزمانی رمزنگاری، نظیر بردار شروع (IV) داشته باشد آنگاه این داده‌ها ممکن است بطور صریح در شروع میدان Payload Data حمل شوند. اگر IV داشته باشیم، معمولاً رمزنگاری نمی‌شود، اگرچه اغلب به آن بعنوان بخشی از متن رمزشده نگاه می‌شود.

مشخصه‌های جاری چنین دیکته می‌کنند که یک پیاده‌سازی سازگار بایستی DES در مُود CBC را حمایت کند. تعدادی از الگوریتم‌های دیگر نیز در اسناد DOI دارای شناسه‌های معین بوده و بنابراین می‌توانند برای رمزنگاری مورد استفاده قرار گیرند. اینها شامل اقلام زیراند

- Three-key triple DES
- RC5
- IDEA
- Three-key triple IDEA
- CAST
- Blowfish

همانند AH، ESP استفاده از یک MAC با طول پیش فرض ۹۶ بیت را حمایت می‌نماید. همچنین همانند AH، مشخصهٔ جاری دیکته می‌کند که پیاده‌سازی سازگار بایستی HMAC-MD5-96 و HMAC-SHA-1-96 را حمایت نماید.

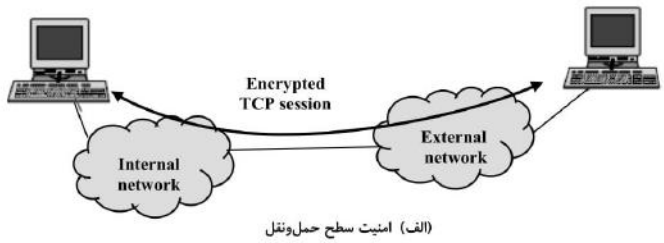
لائی (Padding)

میدان Padding دارای چند هدف است:

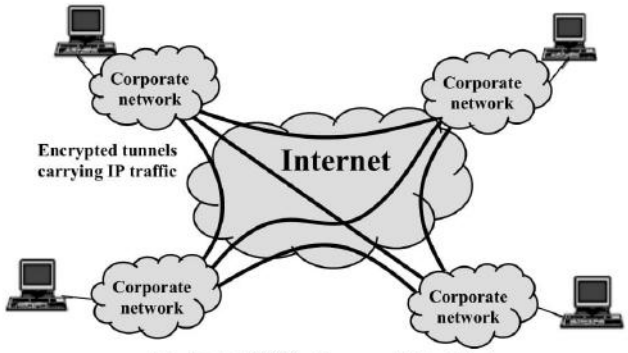
- اگر در یک الگوریتم رمزنگاری لازم باشد که متن ساده مضرب صحیحی از تعدادی بایت باشد (مثلاً مضربی از طول یک بلوک در رمز قالبی)، از میدان Padding برای توسعهٔ متن ساده (شامل میدان‌های Payload Data، Padding، Pad Length و Next Header) به طول مطلوب استفاده می‌شود.
- فرمت ESP نیاز دارد تا میدان‌های Pad Length و Next Header در سمت راست یک کلمهٔ ۳۲- بیتی قرار گیرند. بهمین ترتیب متن رمزشده بایستی مضربی از ۳۲ بیت باشد. میدان Padding برای اطمینان از این امر بکار می‌رود.
- Padding اضافه‌تری ممکن است برای ایجاد محرمانگی جریان ترافیک بکار رود تا طول واقعی محموله را پنهان سازد.

مُودهای حمل‌ونقل و تونل

شکل ۸-۶ دو روش که در آنها سرویس IPSec ESP را می‌توان بکار برد، نشان می‌دهد. در قسمت بالای شکل، رمزنگاری (و بطور اختیاری اعتبارسنجی) بین دو میزبان که مستقیماً بهم وصل‌اند فراهم شده است. شکل ۸-۶ ب نشان می‌دهد که چگونه



(الف) امنیت سطح حمل‌ونقل



(ب) یک شبکهٔ خصوصی مجازی (VPN) از طریق مُود تونل

شکل ۸-۶ رمزنگاری مُود حمل‌ونقل در برابر رمزنگاری مُود تونل

عملیات مُود تونل می‌تواند برای برقراری یک شبکهٔ خصوصی مجازی (VPN) بکار رود. در این مثال، یک سازمان دارای چهار شبکهٔ خصوصی است که در عرض اینترنت بهم متصل‌اند. میزبان‌های روی شبکه‌های داخلی از اینترنت برای انتقال داده‌ها استفاده کرده ولی با سایر میزبان‌های روی اینترنت تعاملی ندارند. با خاتمه دادن به تونل‌ها در دروازه‌های امنیتی هر شبکهٔ داخلی، پیکربندی به میزبانان اجازه می‌دهد که از پیاده‌سازی قابلیت‌های امنیتی اجتناب نمایند. تکنیک قبلی از مُود حمل‌ونقل SA و تکنیک اخیر از مُود تونل SA استفاده می‌کند.

در این قسمت به افق دید ESP برای دو مُود توجه می‌کنیم. ملاحظات برای IPv4 و IPv6 قدری متفاوت‌اند. همانند بحث مربوط به افق دید AH، فرمت بسته‌ها در شکل ۶-۶الف را بعنوان نقطهٔ شروع بکار می‌بریم.

ESP مُود حمل‌ونقل

ESP مُود حمل‌ونقل برای رمزنگاری و اختیاراً اعتبارسنجی داده‌هائی که بتوسط IP حمل می‌شوند (مثلاً یک سِگمنت TCP)، همانند شکل ۹-۶الف، بکار می‌رود. برای این مُود و با استفاده از IPv4، سرآیند ESP در داخل بستهٔ IP درست قبل از سرآیند لایهٔ حمل‌ونقل (مثل TCP، UDP، ICMP) قرار گرفته و تەآیند ESP (میدان‌های Padding، Pad Length و Next Header) بعد از بستهٔ IP قرار می‌گیرند. اگر اعتبارسنجی نیز مورد انتخاب قرار گیرد،میدان ESP Authentication Data نیز پس از تەآیند ESP خواهد آمد. تمام سِگمنت سطح حمل‌ونقل بعلاوهٔ تەآیند ESP رمزنگاری می‌شود. اعتبارسنجی، تمام متن رمزشده بعلاوهٔ سرآیند ESP را شامل می‌شود.

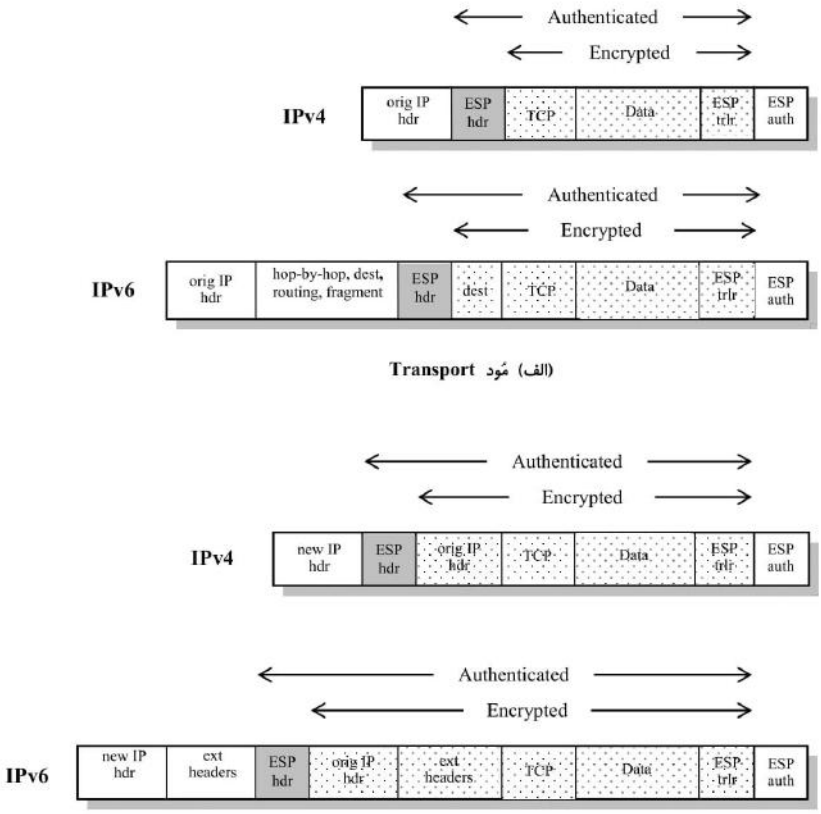
در مقولهٔ IPv6 به ESP بصورت یک محمولهٔ سر- به- سر نگاه می‌شود، یعنی بتوسط مسیریاب‌های میانی مورد پردازش قرار نمی‌گیرد. بنابراین سرآیند ESP بعد از سرآیند اصلی IPv6 base header و سرآیندهای hop-to-hop، routing و fragment extension قرار می‌گیرد. سرآیند الحاقی destination options می‌تواند قبل و یا بعداز سرآیند ESP قرار گیرد که بستگی به منطق مورد استفاده خواهد داشت. برای IPv6، رمزنگاری تمام سِگمنت لایهٔ حمل‌ونقل بعلاوهٔ تەآیند ESP و سرآیند الحاقی مقصد، اگر پس از سرآیند ESP قرار گیرد، را می‌پوشاند. بازهم اعتبارسنجی، متن رمزشده بعلاوهٔ سرآیند ESP را پوشش می‌دهد.

عملیات مُود حمل‌ونقل را می‌توان بصورت زیر خلاصه نمود:

۱- در مبدأ، بلوک دیتا که شامل تەآیند ESP بعلاوهٔ تمام سِگمنت لایهٔ حمل‌ونقل است رمزنگاری شده و متن سادهٔ این بلوک با متن رمزنگاری آن تعویض شده تا انتقال یابد. اعتبارسنجی در صورت انتخاب به آن اضافه می‌گردد.

۲- بستهٔ دیتا آنگاه به سمت مقصد مسیریابی می‌گردد. هر مسیریاب میانی لازم است تا سرآیند IP بعلاوهٔ سرآیندهای الحاقی IP بصورت متن ساده را بررسی و پردازش نماید، ولی نیازی نیست تا متن رمزشده را وارسی کند.

۳- گرهٔ مقصد، سرآیند IP باضافهٔ سرآیندهای الحاقی IP را بصورت متن ساده بررسی می‌کند. آنگاه بر اساس SPI در سرآیند ESP، بقیهٔ بسته را برای دست‌یابی به سِگمنت لایهٔ حمل‌ونقل رمزگشائی می‌نماید.



(الف) مُود **Transport**

(ب) مُود **Tunnel**

شکل ۹-۶ افق دید رمزنگاری و اعتبارسنجی ESP

عملیات مُود حمل‌ونقل، برای هر کاربردی که آن را بکار می‌برد محرمانگی را فراهم می‌سازد و بنابراین از ایجاد محرمانگی در تک‌تک کاربردها اجتناب خواهد شد. این مُود عملیات بصورت معقولی بهره‌ور بوده زیرا مقدار نسبتاً کمی به طول بسته IP اضافه می‌نماید. یکی از نقاط ضعف این مُود این است که می‌توان روی بسته‌های انتقال یافته، تحلیل ترافیک انجام داد.

مُود تونل ESP

از ESP در مُود تونل برای رمزنگاری کل بستهٔ IP استفاده می‌شود (شکل ۹-۶ب). برای این مُود، سرآیند ESP در ابتدای بسته قرار گرفته و آنگاه بسته بعلاوهٔ تەآیند ESP رمزنگاری می‌شوند. این متد می‌تواند برای مقابله با تحلیل ترافیک بکار رود.

چون سرآیند IP شامل آدرس مقصد و احتمالاً دستورات مسیریابی منبع و اطلاعات اختیاری hop-to-hop است، ممکن نیست که بتوان بصورت آسان بستهٔ IP رمزنگاری شده که در ابتدای آن سرآیند ESP قرار دارد را منتقل کرد. مسیریاب‌های بین راه قادر نخواهند بود تا چنین بسته‌ای را پردازش نمایند. بنابراین لازم است که تمام بلوک (سرآیند ESP بعلاوهٔ متن رمزشده بعلاوهٔ Authentication Data، اگر موجود باشد) را با یک سرآیند IP جدید که حاوی اطلاعات کافی برای مسیریابی، ولی نه برای تحلیل ترافیک، باشد کپسولی کرد.

در حالی که مُود حمل‌ونقل برای محافظت اتصالات بین میزبان‌هائی که ESP را حمایت می‌کنند مناسب است، مُود تونل برای استفاده در پیکربندی‌هائی که شامل یک دیوار آتش و یا نوعی دروازهٔ امنیتی دیگر که یک شبکهٔ مورد اعتماد را از شبکه‌های خارجی محافظت می‌کند، مناسب می‌باشد. در این مورد آخر رمزنگاری تنها بین یک میزبان خارجی با دروازهٔ امنیتی و یا بین دو دروازهٔ امنیتی صورت می‌پذیرد. این امر میزبان‌های روی شبکهٔ داخلی را از رَنج رمزنگاری رها ساخته و کار توزیع کلید را با کاهش تعداد کلیدهای مورد نیاز آسان می‌کند. علاوه برآن با تحلیل ترافیک مبتنی بر مقصد نهائی مقابله می‌کند.

موردی را در نظر بگیرید که درآن یک میزبان خارجی می‌خواهد با یک میزبان روی یک شبکهٔ داخلی که بتوسط دیوار آتش از آن محافظت می‌شود و در آن ESP بین میزبان خارجی و دیوار آتش برقرار است، ارتباط پیدا کند. برای انتقال یک سِگمنت لایهٔ حمل‌ونقل از میزبان خارجی به میزبان داخلی قدم‌های زیر بایستی برداشته شود:

۱- مبدأ یک بستهٔ IP درونی با آدرس مقصد میزبان شبکهٔ داخلی را درست می‌کند. این بستهٔ با سرآیند ESP تجهیز شده و آنگاه بسته و ته‌آیند ESP رمزنگاری شده و Authentication Data ممکن است به آن اضافه گردد. بلوک منتجه با یک سرآیند IP جدید (برای IPv6 سرآیند اصلی بعلاوهٔ سرآیندهای الحاقی نظیر routing و hop-to-hop) که آدرس مقصد آن دیوار آتش است کپسولی می‌گردد. این بستهٔ IP بیرونی را شکل می‌دهد.

۲- بستهٔ بیرونی به سمت دیوار آتش مقصد مسیریابی می‌گردد. هر مسیریاب بین راه لازم است که سرآیند IP بیرونی بعلاوهٔ سرآیندهای الحاقی دیگر را وارسی و پردازش نموده ولی نیازی نیست که متن رمزشده را بازدید کند.

۳- دیوار آتش مقصد، سرآیند IP بیرونی باضافهٔ هر سرآیند الحاقی دیگر را بررسی و پردازش می‌کند. آنگاه بر اساس SPI موجود در سرآیند ESP، گرهٔ مقصد بقیهٔ بسته را رمزگشائی کرده تا به متن ساده بستهٔ IP درونی دست‌یابد. این بسته آنگاه در شبکهٔ داخلی انتقال می‌یابد.

۴- بستهٔ درونی از یک یا چند مسیریاب در شبکهٔ داخلی عبور کرده تا به میزبان مقصد برسد.

۶-۵ ترکیب اتحادهای امنیتی

یک SA منفرد می‌تواند یکی از پروتکل‌های AH و یا ESP و نه هردو را اجرا کند. گاهی اوقات یک جریان ترافیک بخصوص، نیازمند هر دو سرویس AH و ESP است. علاوه بر آن یک جریان ترافیک بخصوص ممکن است نیازمند سرویس‌های IPSec بین میزبان‌ها و برای همان جریان، سرویس‌های مجزا بین دروازه‌های امنیتی مثل دیوارهای آتش باشد. در تمام این موارد، SAهای متعددی بایستی برای همان جریان ترافیک بکار گرفته شود تا سرویس‌های IPSec مطلوب را ایجاد نماید. اصطلاح security association bundle به ردیفی از SAها اشاره می‌کند که ترافیک بایستی از درون آنها عبور کرده تا

مجموعهٔ مطلوبی از سرویس‌های IPSec برای آن فراهم شود. SAهای موجود در یک دسته می‌توانند در نقاط انتهائی مختلف و یا همه در یک نقطه خاتمه یابند.

اتحادهای امنیتی می‌توانند به دو صورت با هم دسته‌بندی شوند:

- **مجاورت مُودهای حمل‌ونقل**: به اعمال بیش از یک پروتکل امنیتی به یک بستهٔ IP، بدون استفاده از تونل اشاره می‌کند. این روش ترکیب AH و ESP، فقط ترکیب در یک سطح را مجاز می‌شمارد. لانه‌سازی کردن(nesting) بیشتر سودی ندارد زیرا پردازش در یک مورد IPSec و آنهم در مقصد انتهائی صورت می‌پذیرد.
- **تونل‌های تودرتو**: به اعمال لایه‌های متعدد پروتکل‌های امنیتی که از طریق IPSec اعمال می‌شوند اشاره دارد. این روش سطوح متعدد لانه‌سازی را مجاز دانسته زیرا هر تونل می‌تواند در سایت‌های متفاوت IPSec در طول مسیر، ایجاد شده و یا خاتمه یابد.

این دو روش می‌توانند با هم ترکیب شوند. مثالی در این مورد عبور یک SA حمل‌ونقل بین دو میزبان از درون SA تونل بین دروازه‌های امنیتی، در بخشی از مسیر است.

یک مطلب جالب توجه در هنگام ملاحظهٔ دسته‌های SA، ترتیب قرارگرفتن رمزنگاری و اعتبارسنجی بین یک زوج گرهٔ انتهائی و روش‌های انجام آن است. این مطلب را در دنبالهٔ این بحث مطالعه می‌کنیم. آنگاه به ترکیب‌هائی از SA که شامل حداقل یک تونل هستند اشاره می‌کنیم.

اعتبارسنجی بعلاوهٔ محرمانگی

رمزنگاری و اعتبارسنجی را می‌توان با هم ترکیب کرد تا یک بستهٔ IP را با محرمانگی و اعتبارسنجی بین میزبان‌ها انتقال داد. به چندروش ممکن نگاهی می‌اندازیم.

ESP با قابلیت اعتبارسنجی

این روش در شکل ۹-۶ نشان داده شده است. در این روش، کاربر ابتدا ESP را به دیتائی که باید محافظت شود اعمال کرده و آنگاه میدان Authentication Data را به آن اضافه می‌کند. در واقع دو حالت امکان‌پذیر است:

- **ESP مُود حمل‌ونقل**: اعتبارسنجی و رمزنگاری به محمولهٔ IP که به میزبان تحویل داده می‌شود اعمال شده ولی سرآیند IP محافظت نمی‌شود.
- **ESP مُود تونل**: اعتبارسنجی به تمام بستهٔ IP که به یک آدرس مقصد IP بیرونی (مثلاً دیوار آتش) تحویل می‌گردد اعمال شده و اعتبارسنجی در مقصد صورت می‌پذیرد. تمام بستهٔ IP درونی بتوسط مکانیسم سرّی کردن برای تحویل به مقصد IP درونی محافظت می‌شود.

برای هر دو مورد، اعتبارسنجی بجای اینکه به متن ساده اعمال شود به متن رمزشده اعمال می‌گردد.

مجاورت دو مُود حمل‌ونقل

روش دیگری برای اعمال اعتبارسنجی پس از رمزنگاری، استفاده از دو SA حمل‌ونقل است که درونی آن ESP SA و بیرونی آن AH SA باشد. در این مورد ESP بدون اعتبارسنجی خواهد بود. چون SA درونی یک SA حمل‌ونقل است، رمزنگاری به محمولهٔ IP اعمال می‌شود. بستهٔ منتجه شامل یک سرآیند IP (و احتمالاً ملحقات سرآیند IPv6) و بدنبال آن یک ESP خواهد بود. AH سپس در مُود حمل‌ونقل بکارگرفته شده بطوری که اعتبارسنجی ESP بعلاوهٔ سرآیند IP اولیه (و ملحقات) بغیر از میدان‌های تغییرپذیر را می‌پوشاند. مزیت این روش نسبت به استفاده ساده از یک ESP SA منفرد با اعتبارسنجی اختیاری ESP این است که اعتبارسنجی، میدان‌های بیشتری که شامل آدرس‌های IP مبدأ و مقصد است را می‌پوشاند. عیب آن وجود سربارهٔ دو SA در مقایسه با یک SA است.

مجاورت مُود حمل‌ونقل با مُود تونل

استفاده از اعتبارسنجی قبل از رمزنگاری می‌تواند به دلایل متعددی ارجح باشد. اول اینکه چون دیتای اعتبارسنجی بتوسط رمزنگاری محافظت می‌شود، غیرممکن است که کسی بدون اینکه لو رود بتواند پیام را گرفته و اطلاعات اعتبارسنجی آن را تغییر دهد. ثانیاً ممکن است لازم باشد که اطلاعات اعتبارسنجی همراه پیام را برای مصارف آتی در مقصد ذخیره کرد. این امر در صورتی که اطلاعات اعتبارسنجی به پیام رمزنگاری نشده اعمال گردد ساده‌تر خواهد بود، در غیراینصورت پیام بایستی دوباره رمزنگاری شود تا اطلاعات مربوط به اعتبارسنجی را بتوان تأئید نمود.

یکی از روش‌های اعمال اعتبارسنجی قبل از رمزنگاری بین دو میزبان این است که از یک دسته که شامل یک AH transport SA درونی و یک ESP tunnel SA بیرونی است استفاده کرد. در این مورد اعتبارسنجی به محمولهٔ IP باضافهٔ سرآیند IP (وملحقات)، بجز میدان‌های تغییرپذیر، اعمال خواهد شد. بستهٔ IP نتیجه شده آنگاه در مُود تونل بتوسط ESP پردازش خواهد شد که نتیجهٔ آن این است که تمام بستهٔ درونی اعتبارسنجی شده، رمزنگاری شده و یک سرآیند IP بیرونی جدید (وملحقات) به آن اضافه می‌گردد.

ترکیب‌های اصلی اتحادهای امنیتی

اسناد معماری IPSec چهار مثال از ترکیب SAها که بایستی بتوسط میزبان‌های منطبق با IPSec (مثل ایستگاه‌های کاری، سِرورها) و یا دروازه‌های امنیتی (مثل دیوار آتش، مسیریاب) مورد حمایت قرار گیرند را ذکر کرده است. این ترکیب‌ها در شکل ۱۰-۶ نشان داده شده‌اند. قسمت پائین هر مورد در شکل نمایش دهندهٔ اتصال فیزیکی عناصر است. قسمت فوقانی نمایشگر اتصال منطقی از طریق یک یا چند SA تودرتو است. هر SA می‌تواند یا AH و یا ESP باشد. برای SAهای میزبان- به- میزبان، مُود می‌تواند حمل‌ونقل و یا تونل باشد. در غیر اینصورت مُود حتماً تونل است.

در **مورد اول**، کل امنیت بتوسط سیستم‌های انتهائی که از IPSec استفاده می‌کنند فراهم شده است. برای هر دو سیستم انتهائی که از طریق SA باهم ارتباط برقرار می‌کنند، بایستی کلیدهای سرّی مناسب به اشتراک گذاشته شوند. موارد زیر ترکیب‌های ممکن را نشان می‌دهد:

الف- AH در مُود حمل‌ونقل

ب - ESP در مُود حمل‌ونقل

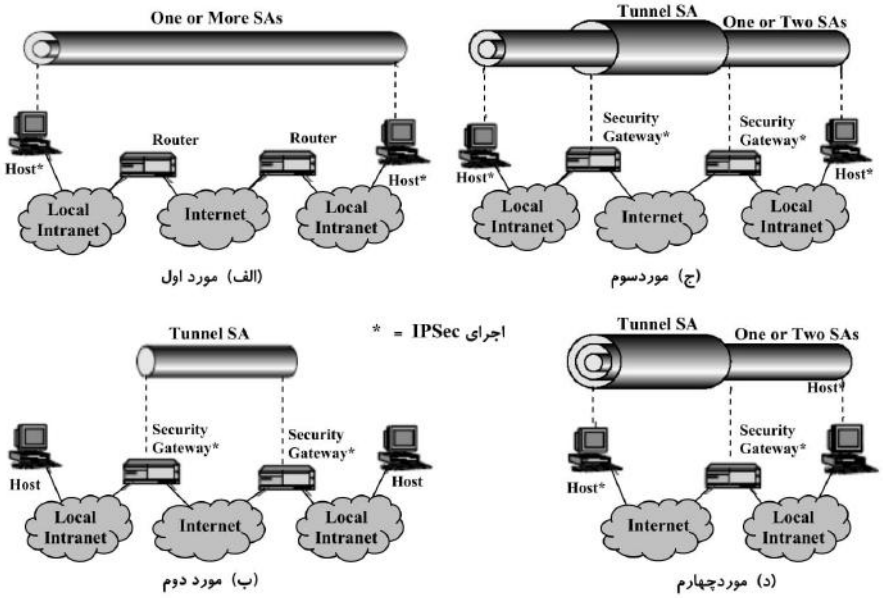
ج- AH به دنبال ESP در مُود حمل‌ونقل (یک ESP SA در درون یک AH SA).

د- هریک از موارد الف، ب، یا ج در داخل یک AH یا ESP در مُود تونل.

قبلاً در مورد انواع ترکیب‌های ذکرشده که می‌تواند برای اعتبارسنجی، رمزنگاری، اعتبارسنجی قبل از رمزنگاری و اعتبارسنجی بعد از رمزنگاری بکار گرفته شود صحبت کرده‌ایم.

برای **مورد دوم**، امنیت فقط بین دروازه‌ها (مسیریاب‌ها، دیوارهای آتش و غیره) فراهم شده و هیچ میزبانی IPSec را بکار نمی‌گیرد. این مثال، استفاده از یک شبکهٔ خصوصی مجازی را روشن می‌کند. سند معماری امنیتی تعیین می‌کند که تنها یک تونل منفرد SA برای این حالت مورد نیاز است. تونل می‌تواند AH، ESP و یا ESP با اعتبارسنجی را حمایت نماید. چون سرویس‌های IPSec به تمام بستهٔ درونی اعمال می‌شود، تونل‌های تودرتو مورد نیاز نیستند.

مورد سوم، روی مورد دوم و با اضافه کردن امنیت سر- به- سر ساخته شده است. همان ترکیب‌های بحث شده در موارد ۱ و ۲ در اینجا نیز مجاز هستند. تونل دروازه- به- دروازه، اعتبارسنجی، محرمانگی و یا هردوی آنها را بین سیستم‌های انتهائی ایجاد می‌کند. وقتی تونل دروازه- به- دروازه، ESP است تا حدی محرمانگی ترافیک را نیز ایجاد می‌کند. هر یک از میزبان‌ها خود می‌توانند سرویس‌های IPSec اضافی را نیز بوسیلهٔ SAهای سر- به- سر برای کاربردهای مختلف و یا کاربرهای مختلف بکار گیرند.



شکل ۱۰-۶ ترکیب‌های اصلی اتحادهای امنیتی

مورد چهارم، از یک میزبان راه دورکه از اینترنت برای دست‌یابی به یک دیوار آتش یک سازمان و سپس به یک سِرور و یا ایستگاه کاری پشت آن دیوار آتش استفاده می‌کند حمایت می‌کند. تنها مُود تونل بین میزبان دور و دیوار آتش مورد نیاز است. همانند مورد اول یک یا چند SA می‌تواند بین میزبان دور و میزبان محلی مورد استفاده قرارگیرد.

۶-۶ مدیریت کلید

بخش مدیریت کلید IPSec، تعیین و توزیع کلیدهای سرّی را بعهده دارد. یک مورد معمول ارتباط بین دو کاربرد، نیاز به چهار کلید دارد. یک جفت کلید ارسال و دریافت برای AH و یک جفت کلید ارسال و دریافت برای ESP. معماری اسناد IPSec به حمایت از دو نوع مدیریت کلید حکم می‌دهد:

- **دستی**: مدیر سیستم، هر سیستم را با کلیدهای خودش و کلیدهای سیستم‌های ارتباطی دیگر بصورت دستی پیکربندی می‌نماید.این مورد برای محیط‌های کوچک و نسبتاً استاتیک کارآئی دارد.
- **خودکار**: یک سیستم خودکار، خلق کلید برای SAها بر اساس تقاضا را برعهده داشته و استفاده از کلیدها در یک سیستم توزیع شده گسترده با پیکربندی در حال تکامل را تسهیل می‌نماید.

پروتکل مدیریت خودکار کلید IPSec را ISAKMP/Oakley می‌نامند و شامل عناصر زیر است:

- **پروتکل تعیین کلید Oakley**: Oakley یک پروتکل مبادلهٔ کلید است که مبتنی بر الگوریتم Diffie-Hellman بوده اما امنیت بیشتری را فراهم می‌آورد. Oakley از اینجهت عام است که فرمت خاصی را دیکته نمی‌کند.
- **پروتکل اتحاد امنیتی و مدیریت کلید اینترنت (ISAKMP)**: ISAKMP چهارچوبی را برای مدیریت کلید در اینترنت فراهم آورده و حمایت‌های جانبی همانند نوع فرمت‌ها بمنظور توافق بر روی جنبه‌های امنیتی را ایجاد می‌کند.

ISAKMP فی‌ذاته الگوریتم مبادلهٔ کلید خاصی را تعیین نمی‌کند بلکه ISAKMP شامل یک مجموعه از انواع پیام‌هاست که استفاده از الگوریتم‌های مبادلهٔ کلید متنوعی را ممکن می‌سازد. Oakley الگوریتم مبادلهٔ کلید خاصی است که برای استفاده از نسخهٔ اولیهٔ ISAKMP اجباری بود.

ابتدا مروری بر Oakley داشته و آنگاه به ISAKMP نگاهی می‌اندازیم.

پروتکل تعیین کلید Oakley

Oakley یک فرم پالایش شده از الگوریتم مبادلهٔ کلید Diffie-Hellman است. بیاد آورید که Diffie-Hellman شامل تعامل‌های زیر بین کاربر A و B بود. از قبل روی دو پارامتر q که یک عدد اول بزرگ و α که یک ریشهٔ اولیهٔ q است توافق می‌شود. A یک عدد صحیح تصادفی X_A را بعنوان کلید خصوصی خود انتخاب می‌کند و کلید عمومی خود یعنی $Y_A = \alpha^{X_A} \bmod q$ را برای B می‌فرستد. بهمین ترتیب B یک عدد صحیح تصادفی X_B را بعنوان کلید خصوصی خود

انتخاب کرده و کلید عمومی خود یعنی $Y_B = \alpha^{X_B} \bmod q$ را برای A ارسال می‌کند. هریک از دو طرف اکنون می‌توانند کلید سرّی اجلاس را بصورت زیر محاسبه نمایند:

$$K = (Y_B)^{X_A} \bmod q = (Y_A)^{X_B} \bmod q = \alpha^{X_A X_B} \bmod q$$

الگوریتم Diffie-Hellman دو مشخصهٔ جالب دارد:

- کلیدهای سرّی فقط وقتی مورد نیازاند خلق می‌شوند. هیچ نیازی نیست تا کلیدهای سرّی را برای مدتی طولانی ذخیره کرد و بدین ترتیب آنها را در مقابل آسیب‌پذیری‌های اضافی قرار داد.
- مبادله کلید نیاز به هیچ زیرساخت از قبل موجودی، بجز توافق روی پارامترهای q و α ندارد.

با وجود این ضعف‌هائی در روش Diffie-Hellman موجود است که در [HUIT98] به آنها اشاره شده است:

- هیچ اطلاعاتی در مورد هویت طرفین به دست نمی‌دهد.
- در معرض حملهٔ man-in-the-middle قرار دارد که در آن طرف سوم C خود را در هنگام مکالمه با A بجای B، و در هنگام مکالمه با B بجای A، جا می‌زند. هردو طرف A و B برای خلق کلید سرّی با C به توافق می‌رسند که در این صورت C می‌تواند به ترافیک گوش کرده و آن را عبور دهد. حملهٔ man-in-the-middle چنین جلو می‌رود:

۱- B کلید عمومی خود Y_B را در پیامی به مقصد A می‌فرستد.

۲- دشمن (E) این پیام را می‌گیرد. E کلید عمومی B را نگاه داشته و یک پیام به مقصد A ارسال کرده که ID کاربر B را داشته ولی کلید عمومی Y_E را حمل می‌کند. این پیام بنحوی ارسال می‌شود که بنظر می‌رسد از طرف سیستم میزبان B ارسال شده است. A پیام E را گرفته و کلید عمومی E که ID کاربر B را دارد نگاه می‌دارد. بهمین ترتیب، E یک پیام را با کلید عمومی E برای B فرستاده و چنین وانمود می‌کند که از A آمده است.

۳- B یک کلید سرّی K_1 بر اساس کلید خصوصی B و Y_E را محاسبه می‌کند. A یک کلید سرّی K_2 که بر اساس کلید خصوصی A و Y_B قرار دارد را محاسبه می‌کند. E کلید K_1 را با استفاده از کلید خصوصی X_E و Y_B، و کلید K_2 را با استفاده از X_E و Y_A محاسبه می‌نماید.

۴- از این به بعد E قادر است تا پیام‌های A به B و پیام‌های B به A را گرفته و رمزنگاری آنها را درطول مسیر تغییردهد. در این صورت نه A و نه B متوجه نمی‌شوندکه آنها با E ارتباط دارند و نه با یکدیگر.

- از نظر محاسباتی حجیم است. در نتیجه در برابر یک حملهٔ clogging که در آن دشمن تقاضای کلیدهای بسیاری را ارسال می‌نماید آسیب‌پذیر است. منابع قربانی بجای انجام کار واقعی درگیر انجام محاسبات بی‌حاصل نمائی و پیمانه‌ای می‌گردند.

Oakley برای بکارگیری مزایای Diffie-Hellman و در عین حال مقابله با ضعف‌های آن طراحی شده است.

خصوصیات Oakley

الگوریتم Oakley با پنج خاصیت مهم مشخص می‌گردد:

۱- از مکانیسمی بنام cookies برای مقابله با حملات clogging استفاده می‌کند.
۲- دو طرف را قادر می‌سازد تا برای ایجاد یک *group* به توافق برسند که این در اصل تعیین پارامترهای اصلی مبادلهٔ کلید Diffie-Hellman است.
۳- برای اطمینان از مقابله با حملات بازخوانی، از nonce استفاده می‌کند.
۴- مبادلهٔ کلیدهای عمومی Diffie-Hellman را ممکن می‌سازد.
۵- مبادلهٔ کلید Diffie-Hellman را برای مقابله با حملات man-in-the-middle اعتبارسنجی می‌نماید.

Diffie-Hellman را قبلاً مورد بحث قرار داده‌ایم. اجازه دهید تا بقیهٔ این عناصر را به نوبت بررسی کنیم. اول، مسألهٔ حملات clogging را در نظر می‌گیریم. در این حمله یک دشمن آدرس منبع یک کاربر قانونی را تقلیدکرده و یک کلید عمومی Diffie-Hellman را برای قربانی می‌فرستد. قربانی عملیات نمائی و پیمانه‌ای را انجام داده تا کلید سرّی را محاسبه کند. پیام‌های پشت سرهم و تکراری از این دست می‌توانند سیستم قربانی را با کارهای بی‌حاصل کُند کنند. **مبادلهٔ cookie** هریک از دو سمت را ملزم می‌سازد تا یک عدد تصادفی، یا همان cookie، را در پیام اولیه ارسال نمایند که طرف دیگر آن را تأئید کند. این تأئید بایستی در اولین پیام مبادلهٔ کلید Diffie-Hellman تکرار شود. اگر آدرس منبع جعل گردد، دشمن هیچ جوابی را دریافت نمی‌دارد. بنابراین یک دشمن تنها می‌تواند یک کاربر را به تولید تأئیدیه مشغول سازد و نه اینکه او را به محاسبات Diffie-Hellman مشغول نماید.

ISAKMP به ملاحظهٔ سه مطلب در تولید cookie حکم می‌دهد:

۱- cookie بایستی وابسته به طرف‌های مشخص باشد. این امر یک حمله‌کننده را از دریافت یک cookie با استفاده از یک آدرس IP حقیقی و پورت UDP، و سپس استفاده از آن به منظور فروبردن قربانی در باطلاق تقاضاهای مکرر از آدرس‌های IP و یا پورت‌های بصورت تصادفی انتخاب شده باز می‌دارد.
۲- نبایستی برای هیچکس بجز واحد صادرکننده cookie امکان داشته باشد که بتواند cookieای درست کندکه بتوسط همان واحد پذیرفته شود. برای تحقق این امر، واحد صادرکنندهٔ cookie بایستی از اطلاعات سرّی محلی در تولید و تأئید آتی یک cookie استفاده کند. بایستی ممکن نباشد که این اطلاعات سرّی را از هیچ cookie خاص استخراج کرد. نکته نهفته در این الزام این است که واحد صادرکننده لازم نیست تا کپی cookieهایش را ذخیره کند، که در این صورت در برابر کشف آسیب‌پذیرتر خواهند بود، بلکه باید بتواند در هر زمان که لازم است cookie ورودی را تأئید نماید.
۳- روش‌های تولید و تأئید cookie بایستی سریع باشند تا با حملاتی که هدف آنها تخریب منابع پردازشی و سرگرم کردن بی‌حاصل آنهاست مقابله شود.

روش توصیه شده برای تولید cookie این است که از یک تابع درهم‌ساز سریع (مثل MD5) روی آدرس‌های IP منبع و مقصد، پورت‌های UDP منبع و مقصد، و یک اندازهٔ سرّی تولید شده در محل استفاده گردد.

Oakley استفاده از **گروه**های مختلف برای مبادلهٔ کلید Diffie-Hellman را حمایت می‌کند. هر گروه شامل تعریف دو پارامتر عمومی و هویت الگوریتم مورد استفاده است. مشخصه‌های فعلی شامل گروه‌های زیر می‌باشند:

- به توان رساندن پیمانه‌ای با یک پیمانهٔ ۷۶۸- بیتی

$$q = 2^{768} - 2^{704} - 1 + 2^{64} \times (\lfloor 2^{638} \times \pi \rfloor + 149686)$$
$$\alpha = 2$$

- به توان رساندن پیمانه‌ای با یک پیمانهٔ ۱۰۲۴- بیتی

$$q = 2^{1024} - 2^{960} - 1 + 2^{64} \times (\lfloor 2^{894} \times \pi \rfloor + 129093)$$
$$\alpha = 2$$

- به توان رساندن پیمانه‌ای با یک پیمانهٔ ۱۵۳۶- بیتی
  - پارامترها بایستی تعیین شوند
- گروه خَم بیضوی روی 2^{155}
  - موّلد (هکزادسیمال): $X = 7B$ و $Y = 1C8$
  - پارامترهای خم بیضوی (هکزادسیمال): $A = 0$ و $Y = 7338F$
- گروه خم بیضوی روی 2^{185}
  - موّلد (هکزادسیمال): $X = 18$ و $Y = D$
  - پارامترهای خم بیضوی (هکزادسیمال): $A = 0$ و $Y = 1EE9$

سه گروه اول الگوریتم‌های کلاسیک Diffie-Hellman هستند که از بتوان رساندن پیمانه‌ای استفاده می‌کنند. دو گروه آخر از خم بیضوی مشابه با Diffie-Hellman استفاده می‌کنند که قبلاً در مورد این روش صحبت شده است. Oakley از **nonce**ها برای اطمینان از مقابله در برابر حملات بازخوانی استفاده می‌کند. هر nonce یک عدد شبه تصادفی تولیدشده در محل است. nonceها در پاسخ‌ها ظاهر شده و در خلال بخش‌های معینی از عملیات مبادله برای امن ماندن رمزنگاری می‌شوند.

سه روش **اعتبارسنجی** متفاوت می‌تواند به همراه Oakley بکار گرفته شود:

- **امضاء دیجیتال**: مبادله با امضاء یک hash که در هر دو سمت قابل حصول باشد اعتبارسنجی می‌گردد. هر طرف hash را با کلید خصوصی خود رمزنگاری می‌کند. اندازهٔ hash روی پارامترهای مهم همانند ID کاربر و nonceها محاسبه می‌گردد.
- **رمزنگاری کلید- عمومی**: مبادله بتوسط پارامترهای رمزنگاری همچون ID ها و nonce ها و با استفاده از کلید خصوصی فرستنده اعتبارسنجی می‌شود.
- **رمزنگاری کلید- متقارن**: یک کلید که بتوسط یک مکانیسم خارج از محدوده تهیه شده است می‌تواند از طریق رمزنگاری متقارن پارامترها، برای اعتبارسنجی مبادله بکار رود.