UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ACATLÁN

UNAM CAMPUS ACATLÁN

LA CRIPTOGRAFÍA COMO MECANISMO DE SEGURIDAD UTILIZADO PARA EL INTERCAMBIO DE INFORMACIÓN CONFIDENCIAL EN INTERNET

T E S I S

QUE PARA OBTENER EL TÍTULO DE

LICENCIADO EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN

P R E S E N T A :

JORGE CLAUDIO MERCENARIO RAMÍREZ

ASESOR: FIS. MAT. JORGE LUIS SUÁREZ MADARIAGA

NOVIEMBRE, 2005

A MIS PADRES

Por su gran amor, consejos, dedicación e impulso para ayudarme a alcanzar mis metas.

A MI HERMANA Y MI SOBRINO

Por su cariño y por ser una parte fundamental en mi vida.

A MIS AMIGOS

Abraham, Alethia, Delia y Graciela, por sus consejos y su apoyo incondicional, así como todos los momentos gratos e inolvidables.

AGRADECIMIENTOS:

Fís. Mat. Jorge Luis Suárez Madariaga, por fungir como mi asesor y brindarme su valioso apoyo, sin el cual no hubiera sido posible culminar este trabajo.

Agradezco a Manuel Aceves Mercenario su interés y apoyo que me brindó para la realización de esta tesis.

INDICE

INTRODUCCIÓN

El objetivo general de este trabajo es dar a conocer las principales técnicas criptográficas así como los protocolos que las utilizan. Asimismo, determinar y destacar las ventajas y desventajas que la Criptografía ofrece en materia de seguridad dentro del intercambio de información confidencial en Internet.

La creciente expansión de las redes de comunicaciones ha hecho necesario la adopción y desarrollo de herramientas de seguridad que protejan tanto los datos transmitidos como el acceso a los elementos de la red de los posibles ataques que puedan sufrir. Las redes de computadoras son particularmente delicadas, debido a ciertas características en su estructura, tal como la ausencia de conexión física directa entre el emisor y el receptor, lo cual no asegura la inmediata transmisión entre ambos, ni garantiza que ésta llegara a producirse, ya que los datos son vulnerables a ataques por parte de terceros no autorizados durante el trayecto entre una computadora y otra. Para que el emisor tenga certeza de que los datos transmitidos alcanzan su destino, se requiere de mensajes de confirmación de reparto o lectura enviados por el receptor.

Los ataques contra un sistema de comunicaciones pueden agruparse en cuatro categorías:

1. Interrupción: ocurre cuando algún elemento del sistema o infraestructura es puesto fuera de servicio.
2. Interceptación: se presenta cuando un usuario no autorizado accede a cierta información o a cualquier elemento de la red.
3. Modificación: ocurre cuando un usuario no autorizado altera el contenido de un mensaje o elemento de la red tras haber accedido al mismo.
4. Fabricación: se presenta cuando un usuario no autorizado inserta información en el sistema, falsificando su identidad,

El ataque de interceptación se considera "pasivo", ya que el intruso no ejerce acción directa sobre los datos que viajan a través de la red o el sistema, mientras que los otros tres tipos de ataques se clasifican como ataques activos debido a que el intruso ejecuta acciones orientadas a obtener, modificar e inhabilitar los datos y/o sistemas dentro de una red.

Para poder detectar y prevenir estos ataques, las redes deben incorporar mecanismos que proporcionen servicios de seguridad. Estos servicios tienen objetivos específicos que pueden resumirse en los siguientes, y que responden a los requerimientos de seguridad en el intercambio de información en Internet:

- **Confidencialidad**: asegura que sólo el usuario o usuarios autorizados puedan acceder al contenido de la información guardada en un sistema informático. En algunos casos, no sólo hay que proteger el contenido de los mensajes (confidencialidad del mensaje), sino también las identidades del emisor y del receptor (confidencialidad del tráfico de mensajes).

- **Integridad**: impide que personas no autorizadas puedan modificar la información transmitida o almacenada. El mensaje debe llegar a su destino sin haber sufrido alteración alguna en su contenido o en el orden de la recepción de sus unidades si se compone de varios bloques.

- **Disponibilidad**: el sistema no debe permitir que usuarios no autorizados dejen fuera de funcionamiento elementos de la red e impidan de este modo las comunicaciones y, por consiguiente, el acceso a la información en el momento en que se requiera.

- **Autentificación**: asegura que el origen de un mensaje, así como el emisor y receptor del mismo, se encuentren plenamente identificados.

- **No-repudio**: propiedad que consiste en dejar constatado si un usuario envía o recibe algún mensaje. De esta manera, ni el emisor del mensaje ni el receptor del mismo pueden negar que se haya efectuado la transmisión.

- **Control de acceso**: asegura que sólo los usuarios autorizados debidamente identificados puedan obtener permiso de acceso a los recursos del sistema.

Si se tuvieran que enumerar las principales características de Internet, se podría observar, en primer lugar, que se trata de una red totalmente abierta y pública, sin ningún tipo de jerarquía establecida, ni de autoridad central. El número de usuarios aumenta día a día de forma exponencial, así como el tipo de operaciones realizadas. Por lo tanto, se puede decir que Internet no es más que una red general formada por la interconexión de una multitud de redes.

Con estas características, es fácil imaginar la inseguridad de los datos transmitidos y almacenados en los equipos de cómputo interconectados. En las distintas capas del modelo de Internet, el cual está basado en el modelo teórico de referencia OSI, se pueden establecer diferentes controles de seguridad atendiendo al tipo de los datos. Por un lado, en el nivel IP o de red hay que establecer un modelo de seguridad que controle los paquetes que circulan por la red y que proporcione servicios de seguridad. Por otro lado, es necesario atender la seguridad en las aplicaciones, es decir, la seguridad de los datos con los que el usuario trabaja de manera más directa. También es necesario atender la seguridad global del sistema terminal y su entorno local.

Las características que proporcionan a Internet su dinamismo, también lo convierten en un ámbito inseguro, tal como las calles de una gran ciudad. Las personas usualmente no caminan descuidadamente en una ciudad de gran tamaño y vibrante de actividad. Del mismo modo, una persona cuidadosa hace uso de Internet con precaución. La información de negocios que viaja a través de Internet es susceptible de ser falsificada, modificada o interceptada por individuos no autorizados. Por otro lado, la tecnología y estilo de los cuales surgió Internet es diferente a los modelos de comunicación tradicionales, por lo que las técnicas tradicionales de seguridad no son aplicables en este terreno.

El correo electrónico es, probablemente, la aplicación distribuida de mayor utilización. Los mensajes de correo electrónico que viajan a través de una red están totalmente expuestos al examen y manipulación de cualquier persona, sea o no el usuario al que van dirigidos. En general, los sistemas de seguridad para mensajería electrónica siguen un esquema híbrido, en el que se utilizan conjuntamente algoritmos de cifrado de llave simétrica o privada y algoritmos de llave pública o asimétrica. Algunas de las ventajas que se pueden obtener mediante este esquema son la velocidad de los sistemas simétricos junto con la seguridad adicional en la gestión de llaves, la cual se deriva de los sistemas asimétricos.

La Criptografía ha surgido como la mejor y más difundida alternativa para proteger los datos en Internet de manera efectiva. Las técnicas modernas de encriptación provienen de la evolución de las técnicas de codificación de siglos pasados, desarrolladas y aumentadas por medio de un conocimiento más profundo de las matemáticas modernas. En la actualidad, se han desarrollado productos y tecnologías criptográficas particularmente para aplicaciones que se emplean en Internet.

El capítulo 1 describe los componentes de un sistema de comunicaciones en Internet, en el cual pueden estar involucrados dos o más equipos de cómputo, pudiendo ser éstos clientes o servidores. Se indican los posibles riesgos en materia de seguridad en el intercambio de información desde un punto de vista tanto del cliente como del servidor. Se describen los ataques más usuales por parte de personas no autorizadas que pudieran intervenir una conexión remota entre equipos de cómputo, así como las consecuencias de los riesgos de seguridad para el usuario final. También se mencionan las medidas de seguridad y operativas que pueden adoptarse para poder obtener un grado razonable de seguridad en el intercambio de información confidencial en Internet.

En el capítulo 2 se describe los antecedentes y desarrollo histórico de la Criptografía. Se definen conceptos básicos de la Criptografía, tales como criptografía y criptoanálisis, cifrado, llave, entre otros. Asimismo, se analizan los componentes de un sistema criptográfico así como los algoritmos y parámetros de seguridad que se toman en cuenta para evaluar su funcionamiento y

efectividad. Se analiza la forma en que trabajan los principales algoritmos de llave pública y privada utilizados en la Criptografía, así como sus principales ventajas y desventajas. Se destacan las propiedades de un buen algoritmo criptográfico, así como las fallas que se pueden presentar en un sistema criptográfico. También, se describen los riesgos y ataques a los que son susceptibles los algoritmos criptográficos.

En el capítulo 3 se analizan los elementos que conforman la Infraestructura de Llave Pública, comúnmente conocida como PKI, por sus siglas en inglés. En el mismo capítulo se muestra cómo la Criptografía de Llave Pública permite que cada mensaje enviado por cualquier usuario que desee confirmar su identidad de manera segura contenga una firma digital, análoga a la firma habitual en la correspondencia ordinaria. El hecho de poder firmar digitalmente un mensaje hace posible que el destinatario se asegure de que el mensaje que recibe es enviado por quien dice ser el remitente. Por otro lado, una firma digital proporciona una mayor garantía que una firma manual, puesto que asegura, como se verá más a detalle posteriormente, que ninguna parte del mensaje que se ha recibido ha sido modificada.
Las firmas digitales deben ser fáciles de hacer, fáciles de verificar y difíciles de falsificar. En muchas ocasiones, con el fin de evitar un ataque contra las firmas por sustitución, donde un usuario no autorizado hace uso de un mensaje antiguo o fragmentos del mismo, se incluye un número de secuencia del mensaje o un sello temporal.

Por último, en el capítulo 4 se describen y analizan los principales protocolos de seguridad empleados en las conexiones seguras a través de Internet y transacciones electrónicas. Se muestra la forma en que dichos protocolos se integran dentro de una conexión de datos a través de Internet para proporcionar los servicios de seguridad mencionados con anterioridad. Asimismo, se describen algunas de las tecnologías emergentes diseñadas específicamente para el comercio electrónico, las cuales permiten la transmisión segura de información confidencial a través de Internet.

El presente trabajo va dirigido a todos aquellos profesionales y/o empresas que consideran necesario preservar la seguridad de sus activos de información, los cuales se pueden traducir en ganancias económicas, o en el aseguramiento de la continuidad operativa de la organización.

HIPOTESIS

Mediante el conocimiento sobre el funcionamiento de los distintos algoritmos propios de la Criptografía, así como del surgimiento y evolución de las tecnologías que hacen uso de los mismos, será posible definir y ampliar servicios de seguridad en una infraestructura de cómputo, de tal manera que proporcione confidencialidad e integridad a información sensitiva transmitida a través de Internet.

CAPÍTULO 1

LOS PROBLEMAS DE SEGURIDAD EN INTERNET

1.1 La seguridad en Internet y la Criptografía

De manera muy elemental, una conexión en Internet se compone de tres partes (Fig. 1.1):

1. Cliente.
2. Servidor.
3. La conexión entre ambos.

Fig. 1.1: Componentes de una conexión en Internet

El cliente puede ser cualquier computadora que hace uso de una aplicación a la que generalmente se le conoce como navegador, por medio del cual se comunica con otro equipo o programa al que se le denomina *servidor*. En general, un servidor es un equipo que proporciona recursos compartidos a los usuarios de una red. La conexión entre ambas partes puede ser por medio de una LAN (red de área local), delimitada a un edificio o área específica y relativamente limitada, o por una WAN (red de área extensa) que permite conectar equipos separados por una gran distancia geográfica, y a la que se pueden conectar mediante un dispositivo específico de hardware instalado en cada sistema, como por ejemplo, un módem, o mediante cable de conexión directa, que permite prescindir del módem en las comunicaciones asincrónicas entre dos equipos próximos; o bien de manera remota sin necesidad de cableado.

Al estar conectado a Internet, el usuario, mediante el navegador instalado en el cliente desde donde se comunica, se conecta a un servidor remoto del que desee obtener alguna información y el cliente solicita una respuesta. El servidor envía de vuelta una respuesta, que se traduce en un documento que el navegador despliega. Aunque este procedimiento a primera vista parece muy simple, dentro de las múltiples ocasiones que se realiza al estar trabajando en línea, existe una serie de premisas básicas generalmente asumidas, pero que si no son tomadas en cuenta o revisadas con detenimiento, la seguridad de los sistemas y la información que se transmite entre ellos estaría seriamente comprometida. A continuación se mencionan dichas premisas:

Desde el punto de vista del usuario

- El servidor remoto es mantenido y operado por la organización que aparenta ser su propietaria.
- Los documentos que el servidor remoto devuelve están libres de virus.
- El servidor remoto no almacenará o distribuirá información que el usuario considere privada, por ejemplo sus hábitos de navegación por Internet.

Desde el punto de vista del administrador del servidor remoto

- El usuario que accede a su servidor mediante el cliente no tratará de acceder al sistema remoto sin autorización o alterará el contenido del sitio Web[1] que reside en dicho servidor.
- El usuario no tratará de obtener documentos del servidor a los cuales no tiene privilegios de acceso.
- El usuario no tratará de bloquear o dañar el servidor, haciéndolo inservible o inaccesible para otros.
- En caso de que se le requiera al usuario una contraseña o cualquier otro tipo de autenticación, se asume que el usuario es quien dice ser.

Desde ambos puntos de vista

- La conexión en red está libre de intrusos espiando sobre la línea de comunicaciones.
- La información enviada entre el navegador y el servidor es transmitida íntegramente, sin alteraciones hechas por terceros.

El propósito principal de la seguridad en Internet es garantizar que estas premisas se mantengan válidas.

Las comunicaciones serían relativamente seguras si el equipo de cómputo estuviera físicamente protegido. Esto se podría lograr mediante el confinamiento del equipo, tal como teclados, CPUs (procesadores), servidores y cableado en áreas restringidas a personas externas por medio de candados o alarmas. Sin embargo, este confinamiento estricto dentro de una oficina no tendría ningún efecto en el caso de que cualquier persona perteneciente a la compañía deseara enviar algún documento o información a través de un equipo conectado a la red en una locación externa. Esto ocurre generalmente cuando se trata de un equipo conectado a Internet.

Las técnicas criptográficas proporcionan un balance adecuado entre el alcance de las comunicaciones y la seguridad en las mismas. Estas técnicas transforman y dan un nuevo formato a los mensajes transmitidos a través de la red, con el objeto de protegerlos contra la revelación y/o alteración de los

[1]Se le denomina comúnmente sitio Web a cualquier ubicación remota en Internet, la cual puede contener documentos que pueden ser visualizados mediante un navegador.

mismos por personas ajenas. Estas técnicas han sido utilizadas por miles de años, aunque las utilizadas actualmente reflejan un alto grado de sofisticación matemática.

Una de las técnicas que hace uso de la Criptografía son las *firmas digitales*. Estas permiten proteger de alteraciones a los mensajes o documentos transmitidos y además verificar de una manera más confiable la identidad del autor de los mismos.

Los riesgos de seguridad pueden variar dependiendo del extremo de la conexión del que se trate, ya sea el del cliente o el del servidor. En ambos casos se encuentran riesgos en común. Por ejemplo, la alteración de documentos efectuada por intrusos afecta a todos, por lo que la encriptación de los mismos es de común interés. Sin embargo, ciertos intereses de cada uno difieren. Por ejemplo, si el administrador de un sitio Web cambia el uso de un archivo CGI (el cual se ejecuta en el servidor) a un applet de Java (el cual se ejecuta en el cliente), los riesgos de seguridad que pudiera traer dicho archivo o documento se transfieren al usuario final. Es por esto que es importante considerar los problemas de seguridad que afectan a cada extremo de la conexión por separado, así como a los que se enfrentan ambos en común y así lograr un balance satisfactorio para ambas partes.

1.2 Seguridad desde el punto de vista del cliente

A continuación se mencionan algunos problemas de seguridad que pueden afectar específicamente la privacidad y seguridad del equipo de un usuario particular (o varios) conectado a la red. Estos riesgos se han ido acrecentando debido a que los navegadores para Internet se han vuelto más sofisticados.

1.2.1 Contenido Activo (Active Content)

En Internet existen actualmente miles de sitios de los cuales se pueden obtener infinidad de documentos y archivos de distintos tipos. Muchos sitios contienen no sólo documentos pasivos, sino que también dentro de esos documentos puede haber "contenido activo", consistente en programas (scripts) CGI, Java, VBscript, etc. ejecutados en la PC del usuario una vez que el navegador ha bajado el documento del servidor. Por otro lado, documentos anteriormente considerados "pasivos", es decir, que no contenían aplicaciones o código adicional en su interior, como por ejemplo los documentos de un procesador de textos, actualmente son susceptibles también a contener virus que se ejecutan a través del contenido activo.

El contenido activo abarca una colección heterogénea de tecnologías que hacen a las páginas Web más interesantes e interactivas. Los applets de Java y los controles ActiveX son los ejemplos más conocidos, pero los plug-

ins (características adicionales) de los navegadores, aplicaciones auxiliares, código ejecutado de Javascript o VBscript son también parte del contenido activo. El código del contenido activo escrito de manera correcta mejora el aspecto de las páginas Web con animaciones, juegos interactivos y aplicaciones como navegadores de bases de datos. En cambio, el código del contenido activo que contenga fallas, puede tener ciertos huecos de seguridad que comprometen la privacidad del usuario o la integridad de la información almacenada en su computadora. En este sentido, el contenido activo escrito con malos propósitos puede dañar el sistema del usuario o lograr acceso no autorizado a una red de área local.

Sin embargo, el contenido activo sigue siendo más un problema potencial que algo real o frecuente. Existe una variedad de applets defectuosas, pero existen pocos ataques de importancia registrados. El principal problema en este sentido siguen siendo las applets "molestas", es decir, páginas con contenido activo cuyo propósito es lograr que el navegador se paralice o se interrumpa.

1.2.2 Invasión a la privacidad

Un riesgo mayor para el usuario final es la pérdida de la privacidad. Aún cuando el navegar por Internet puede parecer una actividad completamente anónima, la realidad es muy diferente. Cada vez que un usuario accede a una página en un sitio remoto, deja algún rastro a su paso. Esto podría ser solamente la dirección de Internet (IP) de su equipo, o podría ser información más personal. No se puede saber con certeza el tipo de uso que cada sitio remoto haga de esa información.

Los sitios Web pueden obtener información de un usuario de diferentes maneras. La más básica es por medio de un log (registro) del servidor, el cual almacena la fecha y hora de la conexión, la dirección IP de la máquina del usuario, la identidad del documento que el usuario solicitó y la dirección de la página inmediata anterior que el usuario estaba visualizando. Hay mayor información disponible para los proveedores de servicios de Internet, cuyos servidores mantienen un registro de todos los sitios visitados por los clientes adscritos al proveedor de servicios.

Otra forma por la que los sitios Web pueden recolectar información de los usuarios es por medio de lo que se denominan *cookies*, los cuales son pequeños identificadores que los sitios Web utilizan para identificar navegadores específicos. Generalmente consisten en un archivo de texto que se almacena en el disco duro local del cliente y contienen algún dato determinado por el servidor que sirve como etiqueta para identificar al navegador en visitas posteriores a la misma página. Fueron originalmente ideadas para ampliar las capacidades de navegación al permitir a los usuarios personalizar la visualización de las páginas Web, explorar bases de datos remotas, navegar a través de mapas de imágenes complejos y realizar otras actividades que requieren continuidad a través de cada sesión de

navegación. Sin embargo, el uso de las *cookies* ha sido objeto de abuso por parte de agencias publicitarias para identificar consumidores, es decir, éstas podrían obtener información detallada acerca de los hábitos de navegación de los usuarios.

De cualquier forma, la mayor amenaza a la privacidad no es la información interceptada en formas ocultas, sino la información que los usuarios voluntariamente proporcionan, por ejemplo, en encuestas de usuarios, grupo de noticias, mensajes de correo electrónico y compras por pedidos electrónicos. Una vez que la información está fuera de las manos del usuario, no hay ningún control sobre cómo es utilizada, y sobre cómo los sitios Web de manera rutinaria la intercambian, agregan y correlacionan con información recolectada por otros sitios. En suma, la misma integración global que ha hecho de la Internet un medio tan completo y conveniente, le ha dado el potencial para convertirse en una vasta red de cámaras espías.

El manejo de la información personal es un tema ampliamente debatido. La necesidad de maximizar la privacidad del usuario se contrapone en un nivel fundamental con la necesidad de los negocios y empresas de minimizar el fraude. La primera necesidad tiene como meta maximizar el anonimato del usuario. La segunda necesidad requiere que los usuarios sean potente e inequívocamente identificados. Es por esto que debe de haber de alguna manera un compromiso equitativo entre ambas partes.

1.3 Seguridad desde el punto de vista del servidor

Algunos riesgos de seguridad afectan principalmente al servidor Web remoto. Dentro de éstos se encuentran las intrusiones, sabotaje y ataques de bloqueo que incapacitan al servidor.

1.3.1 Intrusión y sabotaje

Sin lugar a dudas, uno de los eventos más atemorizantes para los administradores de los servidores es la posibilidad de que sus sitios sean penetrados ilegalmente y modificados por vándalos. En varias ocasiones, el contenido de un sitio es reemplazado con parodias desfavorables o con mensajes de corte político. Las consecuencias pueden ser apenas bochornosas para la organización representada por el sitio, o puede haber efectos más grandes o duraderos. Por ejemplo, una organización que se ha construido cierta reputación en materia de seguridad, tal como alguna rama del gobierno o alguna institución financiera, puede sufrir un daño perdurable a su reputación después de un evento de sabotaje cibernético. En contraste con algunos de los riesgos descritos anteriormente, el sabotaje es un evento que dista de ser raro. Cientos de sitios han sido atacados y modificados,

incluyendo organizaciones altamente sofisticadas como la CIA de Estados Unidos, la Casa Blanca y la NASA.

Los llamados "vándalos cibernéticos" pueden sabotear un sitio al descubrir e infiltrarse por huecos de seguridad derivados de un sistema operativo mal configurado, un servidor Web mal configurado o software con fallas en su ejecución. Los servidores Web, cuyas características rápidamente escalables tienden a rebasar los sistemas de control de calidad de los proveedores, tienen una pobre calificación dentro del ámbito de la seguridad. Muchos servidores Web antiguos tienen fallas que pueden significar grandes huecos de seguridad, dentro de los cuales se encuentran varios servidores comerciales que fueron llamados seguros por su habilidad para poder encriptar comunicaciones. Sin embargo, aun en la generación actual de servidores pueden existir huecos de seguridad aun desconocidos, es decir, situaciones no previstas.

Otra amplia fuente de problemas de seguridad son los scripts CGI[2] y módulos de código del servidor. Estos fragmentos de código extienden la habilidad del servidor de crear páginas Web dinámicas, crear interfases a buscadores y bases de datos, y para generar respuestas a los usuarios cuando proporcionan alguna información. Los *scripts* CGI son relativamente fáciles de programar, fáciles de instalar y de implantar. Los sitios populares hacen un amplio uso de ellos, y algunos tienen cientos de scripts personalizados instalados. Sin embargo esta ventaja a su vez significa su más grande debilidad. Muchos de estos scripts son muy sencillos de escribir y son creados por programadores con poca o nula experiencia en programación en redes y sin reparar en asuntos relacionados a la seguridad. Por lo tanto, la lista de huecos de seguridad en los scripts CGI es amplia y creciente.

Un servidor Web no es más fuerte que su vínculo más débil, es decir, no basta con que un solo componente de un sistema sea seguro. El sistema operativo debe ser seguro, el sistema de archivos también debe ser seguro, así como el software del servidor y cada uno de los scripts y módulos del servidor deben ser seguros. Una falla en cualquiera de estos componentes hace que todas las demás medidas de seguridad sean insignificantes.

Aunque algunos vándalos sólo entran a un servidor Web con el propósito de modificar o parodiar su contenido, otros pueden tener planes más serios, tales como utilizar el servidor como una base para realizar otros ataques desde ahí a las bases de datos de la organización, los servidores de archivos y otros sistemas críticos. Muy a menudo, un servidor Web provee de un portal de entrada a los intrusos, tal como si fuera una puerta amplia dentro de los muros de un castillo.

Para defenderse de este tipo de ataques, es necesario hacer el servidor lo más seguro posible. Sin embargo, el problema fundamental con

[2] CGI son las siglas para Common Gateway Interface, los cuales son archivos que contienen un segmento de código (script) que se ejecuta en el servidor, generalmente en lenguaje Perl o ASP.

los sitios Web es que éstos son sistemas complejos y constantemente cambiantes, los cuales están bajo frecuente presión de crecimiento. Durante este proceso, incluso el mejor administrador puede cometer algún error de programación o configuración que pudiera abrir algún hueco de seguridad.

Cuando se usan apropiadamente, los *firewalls* pueden proteger sistemas de importancia crítica de cualquier amenaza que entre a través del servidor Web. Un firewall es una combinación de hardware y software que proporciona un sistema de seguridad, generalmente para impedir el acceso no autorizado desde el exterior a una red interna o intranet. Un firewall impide la comunicación entre equipos de red y externos mediante el enrutamiento de la comunicación a través de un servidor proxy que se encuentra fuera de la red. El servidor proxy determina si es seguro permitir que pase un archivo con destino a la red. Los firewalls también se denominan servidores de seguridad, puertas de enlace o *gateways* de límite de seguridad. El software del firewall o servidor de seguridad se utiliza para establecer restricciones acerca de qué información se comunica desde una red doméstica o de oficina pequeña a Internet, y viceversa. Por esto, los firewalls también pueden fortalecer efectivamente las defensas contra ataques al servidor. Si se utiliza inadecuadamente, un firewall puede volver a un sitio Web inservible o peor aún, un servidor mal configurado puede crear un hueco de seguridad en el mismo firewall, volviéndolo inútil.

1.3.2 Ataques de bloqueo

En el caso de que un vándalo no pueda acceder y sabotear el sitio Web, puede hacerlo quedar mal haciendo que se bloquee, congele o que procese las peticiones de respuesta de otros servidores remotos muy lentamente, con lo que se volvería inutilizable. Esto puede resultar desastroso para un sitio que requiere de disponibilidad las 24 horas del día.

Los ataques de bloqueo pueden estar dirigidos al sistema operativo, al software del servidor, a los scripts CGI, Javascripts, VBscripts, etc. o a los servicios del sitio Web. Aun cuando no existe una fórmula infalible para evitar los ataques de este tipo, se puede disminuir su impacto poniendo límites a los recursos usados por el servidor y otros programas, así como restringiendo áreas con vulnerabilidades conocidas en el sistema operativo u otro software utilizado.

1.4 Problemas de seguridad en común entre el cliente y el servidor.

Cualquier cosa que afecte la seguridad de la conexión en red entre el cliente y el servidor tiene la misma importancia para ambos. Entre estos riesgos se encuentra la interrupción o bloqueo del servicio, pero también se encuentran principalmente el espionaje y el fraude por Internet.

1.4.1 Espionaje

Para alguien que utiliza el Internet, una conexión entre una computadora A y una computadora B parece tan simple y directa como un trozo de cable que conecta a dos máquinas, pero la realidad es otra. A menos que los equipos estén conectados a la misma red de área local, los datos que fluyen de una máquina a otra pasan por varias localidades antes de llegar a su destino final (Fig. 1.2).

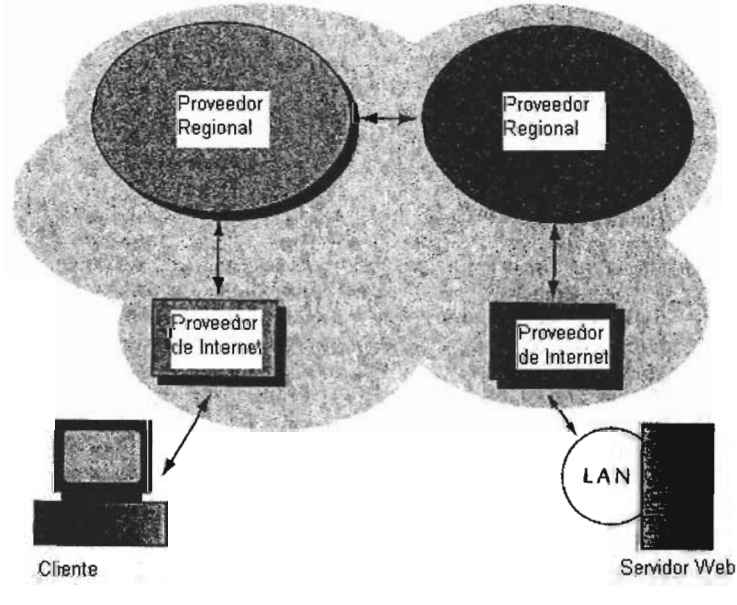


Fig. 1.2: Esquema general de transmisión de datos en Internet

Por ejemplo, un mensaje del navegador del cliente puede viajar primero a través de la línea telefónica al enrutador[3] del proveedor de servicios de Internet (ISP). De ahí, es conducido a través de una línea rentada hacia el proveedor de servicios (RSP). Luego, es transferido a través de una línea de alta velocidad a otro RSP en alguna parte del mundo. Posteriormente, el mensaje viaja al ISP del sitio Web, hacia la red de área local del sitio, y finalmente al equipo servidor del sitio Web.

En cualquier punto de este largo camino, el mensaje puede ser interceptado por individuos sin escrúpulos. Algunos programas pequeños llamados "detectores de paquetes", pueden ser puestos para espiar dentro del tráfico de la red, con el objeto de buscar patrones interesantes en los datos (contraseñas o números de tarjetas de crédito, por ejemplo) y después reportar dicha información al interesado. Para interceptar la comunicación entre un navegador y un servidor Web, un detector de paquetes puede ser instalado en cualquier punto a lo largo del camino entre ambos.

Para poder instalar un detector de paquetes, un individuo sin escrúpulos debe primero acceder a una de las computadoras a lo largo del camino, la cual puede ser una máquina de alguno de los ISP's, alguno de los RSP's, una computadora conectada a la red de área local de la organización que controla el sitio Web, o el propio servidor Web. Debido a que los ISP's de menor tamaño son generalmente más vulnerables que los grandes proveedores regionales, son los blancos más comunes para este tipo de ataques. La red de área local de la organización que controla el sitio Web también es un blanco atractivo para los intrusos.

Una vez que el detector de paquetes es instalado, alguien puede estar al tanto de la conversación entre el navegador y el servidor. Entre las cosas que pueden ser interceptadas, son las direcciones URL que el navegador solicita, el documento que el servidor regresa, las contraseñas proporcionadas por el usuario para acceder áreas restringidas del sitio y los contenidos de cualquier formulario que el usuario envíe. Un individuo malicioso y técnicamente hábil puede ir más allá y alterar los contenidos de la información en camino. Por ejemplo, un criminal cibernético podría alterar el contenido de los mensajes mandados por un usuario al sitio de su banco, para modificar una transferencia electrónica de fondos de tal manera que el dinero fuera transferido a una cuenta diferente a la del usuario.

Algunos "kits" de estos detectores de paquetes se encuentran disponibles en Internet y son los primeros programas que se instalan cuando alguien se introduce en un sistema. Es poco común que un administrador de red se encuentre con algún detector que ha estado corriendo inadvertidamente en un sistema por días o semanas. Sólo hasta el momento en que ocurre algún daño a la infraestructura o activos de información, el administrador se percata de que su sistema pudo haber sido penetrado ilegalmente desde tiempo atrás sin tener la menor idea sobre el asunto. En

[3] Hardware que contribuye a que las redes de área local y de área extensa (LAN y WAN) dispongan de las capacidades de interacción y conexión, y puedan vincular LAN con topologías de red diferentes.

ambientes corporativos, los detectores de paquetes son algo que empleados entrometidos podrían instalar en sus estaciones de trabajo para estar al tanto del tráfico en su red de área local para poder, por ejemplo, observar las sesiones de navegación en la red de otros empleados.

La nueva generación de servicio de Internet, que incluye al cable módem o al Internet inalámbrico, también aumenta el riesgo de espionaje. En un sistema típico de servicio de módem de cable o en redes inalámbricas, todos los usuarios dentro de una misma colonia o región cubierta por un proveedor inalámbrico o un mismo cable de distribución para toda el área comparten paquetes de datos. Debido a que la red inalámbrica y el cable de distribución de área se pueden expandir a lo largo de varias calles, existe el riesgo de que cualquiera o varios vecinos pudieran estar al pendiente del tráfico de información de los demás.

La Criptografía es la principal defensa contra el espionaje cibernético. Al encriptar todas las comunicaciones entre un navegador y un servidor Web, la información se encuentra a salvo de terceros. Esto constituye la primera y más ampliamente aceptada interpretación de la seguridad en la red.

A pesar de los riesgos evidentes de los detectores de paquetes, el problema continúa siendo en buena parte teórico. Aunque algunos vendedores de software podrían tratar de hacer creer al comprador lo contrario, la escasa ocurrencia de incidentes mayores no se debe sólo a la expansión en el uso de software criptográfico. Aun cuando los navegadores con encriptación incluida son comunes, la mayoría de las sesiones se realizan con la encriptación desactivada. Puede ser también que existan muchos casos no reportados, o tal vez existen medios más fáciles para robar la información.

1.4.2 Fraude

Relacionado al problema del espionaje se encuentra el del fraude por Internet. Para ilustrar esta situación se puede mencionar un ejemplo: Cuando un usuario se conecta a una página en Internet que se ve exactamente como la de una tienda reconocida y coloca una orden, existe la posibilidad de que se tratase de un sitio "pirata" que ha copiado las páginas del sitio legítimo y ha establecido un sitio muy similar. Por citar otro ejemplo, cuando un banco implementa servicios de transferencia de fondos en línea, no tiene forma de averiguar con certeza si el usuario que transfiere cierta cantidad de una cuenta a otra está autorizado a hacerlo. En el Internet no existe el frente a frente con las personas, no se puede solicitar alguna identificación con fotografía ni otras referencias personales. Por lo tanto, para realizar negocios en Internet, debe de haber un modo confiable para autentificar a las personas y a las organizaciones.

Convenientemente, las mismas técnicas que son utilizadas para encriptar las comunicaciones también pueden ser utilizadas para crear "firmas

digitales" no falsificables con el propósito de autentificar a los usuarios. Este sistema funciona básicamente asignando certificados virtuales a diversos participantes en Internet. Los certificados son *vouchers* electrónicos expedidos por compañías cuyo trabajo consiste en garantizar la identidad de personas y organizaciones. De esta forma, los sitios Web se autentifican a sí mismos con los certificados de sitio, mientras que los usuarios individuales se pueden autentificar con certificados personales. Existen tipos especiales de certificados para cada tipo de usuario, por ejemplo, para fabricantes de software, compañías de tarjetas de crédito y bancos, entre otros.

Aun cuando la encriptación y autentificación generalmente van de la mano, es posible autentificar sin encriptar y encriptar sin autentificar. La combinación que finalmente se hace de ambas acciones permite tener una conexión segura en Internet.

1.5 Metas a alcanzar y costos para obtener un nivel de seguridad aceptable.

El nivel de riesgo y el nivel de seguridad son casi siempre cuestión de una decisión de negocios. Por esto, es necesario revisar cuidadosamente los riesgos potenciales. Por ejemplo, si el sitio realiza transacciones comerciales, es necesario ver la magnitud de los riesgos que se pudieran derivar de transacciones fraudulentas. Para poder tomar una decisión adecuada, es prudente analizar cuál sería la mayor pérdida posible que pudiera surgir, a partir de la cual se podría sospechar o detectar algún fraude. Generalmente, todas las tiendas al mayoreo y menudeo aceptan tener pérdidas habituales como consecuencia de hacer negocios, por lo que las medidas de seguridad que cada una adopte deben tratar de establecer un balance entre el costo de las mismas y las pérdidas que pudieran derivarse al no establecer medida alguna.

Es poco probable tener productos perfectamente integrados que satisfagan todos y cada uno de los requerimientos de seguridad listados a continuación. Esto es debido a que la seguridad implica casi siempre un intercambio o sacrificio de algún tipo, ya que sería muy difícil cumplir con todos los objetivos de un sistema operativo y al mismo tiempo cumplir con todos los requerimientos para que el sistema de seguridad pueda funcionar aceptablemente en un frente más amplio, si se considera por ejemplo, su implantación en amplias redes de cómputo.

A continuación se enumerará una lista de parámetros que se toman en cuenta para alcanzar la meta de lograr la seguridad en las comunicaciones al aplicar la criptografía o cualquier otra medida de seguridad en redes. La revisión de los siguientes parámetros asume que existe la necesidad de transmisión de información confidencial de negocios o transacciones comerciales valiosas, la cual se realizará mediante el uso de protocolos de Internet. La meta básica es proteger las comunicaciones con un nivel

razonable de seguridad de que los intrusos no podrán leer o modificar los mensajes. Sin embargo, los siguientes parámetros pueden implicar desventajas en otras áreas o sacrificar funcionalidad o costos, especialmente si se desean aplicar todos a la vez, por lo que se deben de tomar en cuenta primeramente aquellos de los que se sospecha exista alguna amenaza directa relacionada con los mismos. Por ejemplo, posiblemente no sea necesario utilizar la criptografía para distribuir información libremente a través de Internet. Sin embargo, existen amenazas más factibles para vendedores que distribuyen información a cambio de algún pago. Por consiguiente, es necesario revisar la aplicación que se va a utilizar en Internet y decidir si alguien se beneficiaría al manipularla de alguna forma. Si tal conducta constituye una amenaza a los objetivos del usuario o la organización, entonces es necesario recurrir a las medidas de seguridad que puedan proteger en mayor medida sus intereses.

Dentro de los siguientes parámetros de seguridad se destacan los costos u obstáculos o desventajas que surgen de la necesidad de mantener a las organizaciones comunicadas, y que se contraponen muchas veces a las medidas de seguridad requeridas.

- **Autentificación poderosa de los mensajes**

Se adopta esta medida cuando la organización necesita asociar mensajes con individuos particulares o agentes autorizados, y requiere hacerlo de una manera muy confiable. También, cuando la organización puede enfrentar pérdidas serias o daño si mensajes importantes son alterados o imitados fraudulentamente.

- **Alta confidencialidad**

La confidencialidad es importante para la mayoría de las organizaciones, pero para algunas es realmente crucial. En tales casos, la fuga de un solo mensaje puede comprometer seriamente los objetivos de la organización y ocasionar un daño del que sería muy difícil recuperarse. La alta confidencialidad es muy costosa de adquirir.

- **Comunicación fácil entre varios servidores**

Cada servidor en una organización necesita comunicarse con una creciente comunidad de servidores distintos. Esto implica generalmente que cada servidor esté conectado por lo menos a una red de área local (LAN). Este es un caso muy común hoy en día, por lo que lograr una comunicación no sólo fácil y efectiva, sino también segura entre los servidores representa un reto para las organizaciones.

- **Acceso al Internet en general**

Las funciones que algunos usuarios desempeñan requieren rutinariamente de consultar información de la red mundial (mejor conocida como WWW, o World Wide Web en inglés) y de intercambio de mensajes de correo electrónico con otros usuarios en general de Internet. Esto constituye un obstáculo importante. El Internet proporciona un vasto caudal de información y oportunidades de comunicación incomparables, pero también trae consigo amplia amenaza internacional directamente a cualquier equipo conectado.

- **Venta de productos a usuarios externos de la red a través de Internet.**

Los clientes actuales y potenciales necesitan acceder al servidor, navegar a través del sitio para encontrar los productos deseados y cerrar de manera segura una operación de compra-venta. Tanto el comprador como el vendedor desean identificar a la otra parte dentro de la transacción y proteger la misma de interferencias por parte de terceros. Sin embargo, no hay forma de saber de antemano si los visitantes al sitio son clientes nuevos o potenciales, o si se trata de alguien que desee atacar el sitio.

- **Economía de costos de mantenimiento y facilidad de uso**

Este es el principal obstáculo para muchas organizaciones. Las soluciones de seguridad muy sofisticadas y de alto costo son inconvenientes para muchas organizaciones. Sin embargo, varias de ellas estarían dispuestas a aceptar gastos más altos a cambio de mejor seguridad.

Cada una de estas metas puede ser benéfica para un negocio y algunas para otro. Sin embargo, es imposible lograr todos estos objetivos simultáneamente mediante una sola aplicación. Si es necesario alcanzar distintitas metas que originan algún conflicto entre sí, puede ser necesario apartar distintas redes para lograr cada una. Dentro del análisis de objetivos de seguridad de una organización es recomendable enfocarse primero a las necesidades operacionales inmediatas. La seguridad puede ser muy costosa, por lo que es preferible lidiar con los riesgos inmediatos primero y las necesidades operacionales.

CAPÍTULO 2

FUNDAMENTOS DE LA CRIPTOGRAFÍA

2.1 Criptología

Se entiende por Criptología el estudio y práctica de los sistemas de cifrado destinados a ocultar el contenido de mensajes enviados entre dos partes: emisor y receptor. La Criptología es la suma de dos ramas principales: Criptografía y Criptoanálisis.

La Criptografía es la parte de la Criptología que estudia cómo cifrar efectivamente los mensajes. La palabra criptografía proviene del griego *kryptos*, que significa esconder y *gráphein*, escribir, es decir, escritura escondida. La criptografía ha sido usada a través de los años para mandar mensajes confidenciales con el propósito de que sólo las personas autorizadas pudieran entender el mensaje. Actualmente tiene el significado de ciencia de la comunicación segura y su objetivo es que dos partes puedan intercambiar información sin que una tercera parte no autorizada, a pesar de que capte los datos, sea capaz de descifrar la información. La criptografía actúa mediante **criptosistemas**, un sistema que permite cifrar los mensajes de tal forma que una persona no autorizada no pueda descifrar el mensaje.

El Criptoanálisis es la parte de la Criptología que estudia cómo descifrar los mensajes cifrados.

2.1.1 Antecedentes de la Criptografía

Tradicionalmente, la Criptografía ha sido asociada con soldados, diplomáticos y espías. De hecho, las técnicas criptográficas han sido usadas por siglos para proteger mensajes importantes de negocios o comerciales. Con la evolución de las comunicaciones por medio de las computadoras, se han desarrollado fuertes técnicas criptográficas para propósitos comerciales así como para proteger los mensajes entre gobiernos de países. Inicialmente, estas técnicas sólo eran usadas por instituciones que tenían muchos riesgos y estaban dispuestas a invertir considerablemente en su protección.

Se puede decir que la criptografía es tan antigua como la civilización. Las cuestiones militares, religiosas o comerciales a través de los siglos han sido los principales impulsores del uso de escrituras secretas; por ejemplo, en el antiguo Egipto, el pueblo utilizaba la lengua demótica, mientras que los sacerdotes usaban la escritura hierática (jeroglífica) incomprensible para el resto.

Los antiguos babilonios también utilizaron métodos criptográficos en su escritura cuneiforme. El primer caso claro de uso de métodos criptográficos se dio durante la guerra entre Atenas y Esparta, el cifrado se basaba en la

alteración del mensaje original mediante la inclusión de símbolos innecesarios que desaparecían al enrollar la lista en un rodillo llamado *escítala*, por lo que el mensaje quedaba claro cuando se enrollaba la tira de papel alrededor del rodillo (escítala) de longitud y grosor adecuados.

Carlomagno sustituía ya las letras por símbolos extraños. En la época de los romanos se utilizó el cifrado de César que consistía en cambiar cada letra por la que ocupaba tres lugares más adelante en el abecedario.

En la Edad Media San Bernardino evitaba la regularidad de los signos (con lo que el criptoanálisis por el método de las frecuencias no era efectivo) sustituyendo letras por varios signos distintos, así tenía un símbolo para cada consonante, usaba tres signos distintos para cada una de las vocales y utilizaba signos sin ningún valor.

El libro más antiguo del que se tiene constancia y que trata sobre criptografía es el Liber Zifrorum escrito por Cicco Simoneta en el siglo XIV. En el siglo XV destaca León Battista Alberti, quien es considerado por muchos el padre de la Criptología; crea la primera máquina de criptografiar que consiste en dos discos concéntricos que giran independientes consiguiendo con cada giro un alfabeto de transposición.

En el siglo XVI, Girolamo Cardano utilizó el método de la tarjeta con agujeros perforados, que se debía colocar sobre un texto para poder leer el mensaje cifrado; en ese mismo siglo Felipe II utilizó una complicada clave que el francés Viete logró descifrar. En ese mismo siglo, Blaise de Vigénere publica Traicté des Chiffres donde recoge los distintos métodos utilizados en su época, el método Vigenére es un método clásico de cifrado por sustitución que utiliza una clave.

Carlos I de Inglaterra usó en el siglo XVII códigos de sustitución silábica. Napoleón, en sus campañas militares y en los escritos diplomáticos, usó los llamados métodos Richelieu y Rossignol y para evitar la regularidad de los símbolos asignaba números a grupos de una o más letras.

En el siglo XIX se utiliza ampliamente el método de transposición, consistente en la reordenación según distintos criterios de los símbolos del mensaje. Kerchoffs escribe el libro La Criptografía Militar, en el que da las reglas que debe cumplir un buen sistema criptográfico.

En la Primera Guerra Mundial los alemanes usaron el sistema denominado ADFGX en el que a cada combinación de dos letras del grupo ADFGX se le hace corresponder una letra del alfabeto y a la que posteriormente se le hacía una transposición en bloques de longitud 20.

El mayor desarrollo de la criptografía se dio en el periodo de entreguerras por la necesidad de establecer comunicaciones militares y diplomáticas seguras. En 1940 se construyó la máquina Hagelin C-48 consistente en seis volantes unidos por el eje y con distinto número de dientes. En la Segunda Guerra Mundial se construyó por parte alemana la

máquina Enigma, que se basaba en un perfeccionamiento del cilindro de Jefferson, pero la máquina británica Colossus consiguió descifrar los mensajes cifrados con Enigma. Los americanos construyeron la máquina Magic utilizada para descifrar el código púrpura japonés; los norteamericanos a su vez usaron a los indios navajos con su difícil lenguaje para la transmisión de mensajes.

Con el desarrollo de la informática en la segunda mitad del siglo XX y con el uso cada vez más extendido de las redes informáticas y del almacenamiento masivo de información se ha dado paso a un gran salto en el estudio de sistemas criptográficos.

En 1975 Diffie y Hellman establecieron las bases teóricas de los algoritmos de llave pública, hasta entonces no se concebía un sistema de cifrado que no fuese de llave secreta. En la actualidad se usan distintos métodos criptográficos, el AES (de llave privada), método RSA (de llave pública), entre otros.

2.1.2 Criptografía

La Criptografía es una colección de técnicas que transforman datos de forma tal que sean difíciles de revertir a su forma original por otras personas. Al proceso de convertir un mensaje que se encuentra en texto plano o normal en una forma de texto codificado o también llamado texto cifrado se le denomina *encriptación.* En muchos países la encriptación está sujeta a las leyes y regulaciones del gobierno.

La criptografía se usa en general para:

- Proteger los datos que viajan a través de las redes, de la interceptación y manipulación no autorizada.
- Proteger la información almacenada en las computadoras de la visualización y alteración indebidas.

Adicionalmente, generalmente se espera de la criptografía que se encargue de las siguientes tareas:

- **Integridad**: Debe ser posible para el usuario que recibe un mensaje verificar y evitar las alteraciones accidentales o intencionales de datos en tránsito.

- **Autentificación**: Debe ser posible para el usuario que recibe un mensaje el poder verificar la autenticidad del origen de una transacción o documento, de tal forma que un intruso no pueda hacerse pasar por un usuario legítimo al interceptar un mensaje.

- **No Repudiación**: El remitente no puede negar haber sido el autor del mensaje posteriormente a su envío.

2.1.3 Criptoanálisis

Es el proceso inverso al realizado por la Criptografía. Consiste en la reconversión de mensajes encriptados nuevamente a texto plano. El criptoanálisis exitoso puede recuperar ya sea el mensaje de texto plano o la llave utilizada para desencriptarlo. También puede llevar a encontrar vulnerabilidades en los sistemas criptográficos que lleven a los resultados mencionados. A un intento de criptoanálisis se le considera un ataque.

Un supuesto fundamental en el criptoanálisis, mencionado primeramente por el holandés A. Kerchoffs en el siglo XIX es que el secreto debe residir fundamentalmente en la llave. Este supuesto se enuncia suponiendo que el criptoanalista conoce el funcionamiento e implementación del algoritmo criptográfico. La mayor parte del conocimiento sobre la Criptografía ha sido mantenido por los gobiernos, principalmente de Estados Unidos, por lo que organizaciones privadas y matemáticos han desarrollado sus propias técnicas criptográficas así como de criptoanálisis. Conforme se hacían más modernas y más eficientes, las técnicas de comunicaciones tendían a sacrificar velocidad por seguridad y no siempre proveían de suficiente seguridad.

La fortaleza básica de la criptografía moderna reside en un largo conjunto de trabajo matemático el cual contiene ciertos cálculos que son inevitablemente más difíciles de ser reversibles. Por ejemplo, si dos números se han multiplicado para producir un tercero (por ejemplo 225389) es muy difícil comenzar con ese número y determinar cuáles dos números primos lo produjeron. Esto reduce el riesgo de debilidades en el diseño del algoritmo que permitieran "atajos" sistemáticos para romperlo.

El resultado matemático básico buscado al analizar un código moderno es la evidencia de que no existe ningún atajo para determinar el texto plano[1] o la llave asociada con algún texto cifrado. El resultado de eso implica que un ataque criptoanalítico generalmente debe usar un ataque denominado "fuerza bruta", el cual intenta utilizar todas las llaves posibles en busca de la correcta. Este tipo de acercamiento es práctico si existe un número reducido de llaves y los criptoanalistas tienen una computadora rápida. Nunca podrá existir una defensa contra este tipo de ataque, por lo que los diseños modernos de los algoritmos criptográficos buscan que los ataques de fuerza bruta tomen el mayor tiempo posible para descifrar el código. Por lo tanto, un código matemático es más seguro si tiene una llave más larga.

El criptoanálisis se ha basado principalmente en la larga y pacienzuda aplicación del ensayo y error para poder romper algún código. Con el avance tecnológico, el proceso se ha visto agilizado. Por esto actualmente se puede descifrar un texto cifrado que haya sido encriptado con una llave corta, utilizando una estación de trabajo. Por ejemplo, muchas aplicaciones comerciales como hojas de cálculo o procesadores de palabras proveen

[1] Texto sin formato o datos que no están cifrados. A veces se llama también texto no cifrado.

protección por contraseña encriptada para sus documentos. Debido a que muchos usuarios tienden a olvidar estas contraseñas, algunos desarrolladores de software han producido independientemente programas de utilerías para recuperar contraseñas, frecuentemente por ataque de fuerza bruta.

Por otro lado, existen cuatro tipos principales de ataques criptoanalíticos:

1.- **Ataque de texto cifrado único**: El criptoanalista tiene en texto cifrado de varios mensajes, los cuales han sido todos encriptados usando el mismo algoritmo de encriptación. El trabajo del criptoanalista consiste entonces en recuperar el texto plano de cuantos mensajes sea posible, o bien de deducir la llave o llaves usadas para encriptar los mensajes, para así poder desencriptar otros mensajes encriptados con las mismas llaves. Esta forma de ataque se puede describir matemáticamente de la siguiente forma:

Dado $C_1 = E_k(P_1), C_2 = E_k(P_2), \ldots C_i = E_k(P_i)$
Deducir ya sea $P_1, P_2 \ldots P_i; k$; o un algoritmo para inferir P_{i+1} de $C_{i+1} = E_k(P_{i+1})$

2.- **Ataque de texto plano conocido**: El criptoanalista tiene acceso no sólo al texto cifrado de varios mensajes sino al texto plano de esos mensajes. Su trabajo consiste en deducir la llave (o llaves) usadas para encriptar los mensajes o un algoritmo para desencriptar cualesquiera otros mensajes encriptados con la misma llave o llaves. Esta forma de ataque se puede describir matemáticamente de la siguiente forma:

Dado $P_1, C_1 = E_k(P_1), P_2, C_2 = E_k(P_2) \ldots P_i, C_i = E_k(P_i)$
Deducir ya sea k o un algoritmo para inferir P_{i+1} de $C_{i+1} = E_k(P_{i+1})$

3.- **Ataque de texto plano escogido**: El criptoanalista no sólo tiene acceso al texto cifrado y el texto plano asociado para varios mensajes, sino que además escoge el texto plano que es encriptado. Este ataque es más poderoso que un ataque de texto plano conocido porque el criptoanalista puede escoger bloques específicos de texto plano para encriptar, de tal forma que sean algunos que produzcan más información sobre la llave. Su trabajo es deducir la llave o llaves usadas para encriptar los mensajes o un algoritmo para desencriptar cualesquiera nuevos mensajes encriptados con la misma llave o llaves. Esta forma de ataque se puede describir matemáticamente de la siguiente forma:

Dado $P_1, C_1 = E_k(P_1), P_2, C_2 = E_k(P_2) \ldots P_i, C_i = E_k(P_i)$
Donde el criptoanalista puede escoger entre $P_1, P_2, \ldots P_i$
Deducir ya sea k o un algoritmo para inferir P_{i+1} de $C_{i+1} = E_k(P_{i+1})$

4.- **Ataque de texto plano escogido adaptable**: Este es un caso especial de un ataque de texto plano escogido. Esto no sólo permite que el criptoanalista escoja el texto plano que sea encriptado, sino que también puede modificar

su elección basado en los resultados de la encriptación previa. En un ataque de texto plano escogido, el criptoanalista puede probablemente ser capaz de escoger un bloque largo de texto plano a ser encriptado; en cambio, en un ataque de texto plano escogido adaptable se puede escoger un bloque más pequeño de texto plano y después escoger otro basado en los resultados del primero, y así sucesivamente.

Los ataques de texto plano conocido y de texto plano escogido son más comunes de lo que se pudiera pensar. No es insólito el hecho de que algún criptoanalista haya obtenido un mensaje de texto plano que ha sido encriptado o sobornar a alguien para que encripte un mensaje escogido. A veces el soborno no es necesario, ya que se le puede entregar un mensaje a un intermediario, por ejemplo, y después se encripta para enviarlo al destinatario. Muchos mensajes tienen principios y finales parecidos y pueden ser conocidos por el criptoanalista. El código fuente de programas es especialmente vulnerable por la aparición frecuente de palabras clave como por ejemplo *#define, struct, else,return*. El código ejecutable encriptado tiene el mismo tipo de problemas: funciones, estructuras recursivas, etcétera. Como antecedente histórico, los ataques de texto plano conocido (e incluso de texto plano escogido) fueron empleados exitosamente en contra de los alemanes y los japoneses durante la Segunda Guerra Mundial.

2.1.4 El ataque de "Fuerza Bruta"

El ataque denominado *Fuerza Bruta* opera tratando con todos los posibles valores para una llave hasta encontrar la correcta para poder descifrar los mensajes encriptados. Una vez logrado esto, el atacante puede leer el mensaje encriptado con esa llave, así como cualquier otro mensaje encriptado con la misma. La principal defensa contra el ataque de fuerza bruta es producir una lista larga de llaves posibles. Conforme la lista va creciendo, así como la longitud de la llave, mayor es la cantidad de trabajo que toma encontrar la llave correcta.

Tipo de llave	No. de bits	Número de llaves	Tiempo de prueba*	No. de pruebas paralelas	Tiempo promedio de búsqueda
Candado de maleta de 3 dígitos	10	1000	2 seg.	1	17 min.
PIN de tarjeta bancaria**	14	10,000	60 seg.	1	3.5 días
Contraseña de texto corta	28	81,450,625	50 µseg	1	34 min.
Llave de Netscape (versión exportación)	40	1,099,511,627,766	50 µseg	1	10 meses
Llave de Netscape (versión exportación)	40	1,099,511,627,766	50 µseg	50	6 días
Contraseña de texto larga	56	6,634,204,312,890,620	50 µseg	1	5,274 años
Llave DES	56	72,057,594,037,927,900	50 µseg	1	57,280 años
Llave DES	56	72,057,594,037,927,900	.02 µseg	57,600	3.5 horas
* Un microsegundo (µseg) equivale a una millonésima de segundo (0.000001 seg) ** PIN.- Número de identificación personal					

Tabla 2.1 Tiempos de rompimiento de llaves por fuerza bruta

La tabla 2.1 ilustra el tiempo que pueden tardar en funcionar los ataques de fuerza bruta. La primera columna indica el tipo de llave y su tamaño en bits en la segunda columna. De ahí se calcula el número de llaves posibles, elevando 2 a la potencia correspondiente al número de bits indicado en la segunda columna, cuyo resultado aparece en la tercera columna. La cuarta columna indica el tiempo para probar una llave. El primer ejemplo es una llave de candado de maleta, la cual generalmente es un candado con tres ruedas metálicas marcadas con dígitos del 0 al 9. El candado se abre cuando la combinación correcta es indicada. El tiempo que se toma para probar una llave corresponde al tiempo necesario para cambiar la combinación a una nueva y ver si funciona. La quinta columna indica el número de pruebas paralelas que se llevan a cabo dentro del sistema, es decir, indica cuántas llaves se están probando a la vez en cada prueba. La última columna indica en promedio el tiempo que tardaría encontrar la llave correcta usando el ataque de fuerza bruta. Retomando el ejemplo mencionado, aun cuando el candado de una maleta tiene miles de combinaciones posibles, el tiempo promedio para encontrar la correcta es menos de una hora y media. Una persona afortunada podría encontrarla en menor tiempo, pero ninguna búsqueda tomaría el doble del tiempo promedio.

El ejemplo de la segunda columna es un número de identificación personal (PIN) en una tarjeta bancaria típica. El tiempo para probar un PIN de tarjeta refleja el tiempo necesario para insertar la tarjeta en el cajero automático, ingresar el PIN y esperar a que la máquina rechace la tarjeta si el PIN es incorrecto. Debido a que existen más combinaciones posibles, el atacante del sistema debe seguir intentando en promedio, por 3.5 días más. Sin embargo, durante este tiempo un número excesivo de intentos podría bloquear la tarjeta o llamar la atención de las autoridades del banco.

El resto de la tabla muestra búsquedas basadas en computadoras. Las contraseñas en sistemas tradicionales Unix son vulnerables a ataques de fuerza bruta debido a que el archivo de contraseñas puede ser leído por todos los usuarios, por lo que la confidencialidad de las contraseñas es preservada por medio del encriptamiento. Para muchos atacantes o hackers[2], se ha convertido en un popular pasatiempo el capturar el archivo de contraseñas de Unix de algún sitio en particular y tratar de adivinar la contraseña. Como se indicó en la tabla 2.1, los atacantes al sistema pueden recuperar contraseñas cortas más fácilmente que las largas, las cuales generalmente resisten los ataques.

Los ejemplos del navegador de Internet Netscape y el algoritmo DES ilustran el valor del trabajo en equipo en el ataque criptoanalítitco de fuerza bruta. La quinta columna de la tabla 2.1 indica el número de llaves que han sido probadas simultáneamente en diferentes ataques. Los ejemplos del navegador Netscape (cuarta y quinta filas de la tabla) se refieren a la encriptación usada en acceso seguros a la red mundial de Internet (World Wide Web) por el navegador Netscape. El software de encriptación del navegador usa una razonablemente larga llave, pero la mayoría de las copias del software sólo manejan llaves secretas de 40 bits de largo (este fue un compromiso para adquirir permiso de exportación del software). La tabla 2.1 también muestra que una estación de trabajo simple requiere en promedio 10 meses de trabajo para recuperar los 40 bits. La quinta fila de la tabla ilustra qué pasaría si 50 estaciones de trabajo fueran todas aplicadas al problema simultáneamente. En este caso, el tiempo promedio es reducido a ligeramente más de seis días.

La posibilidad de descifrar una llave de 40 bits de Netscape por medio del ataque de fuerza bruta fue demostrada por un par de estudiantes franceses en el verano de 1995. Comenzaron con una transacción de Internet de muestra que alguien más había encriptado con el navegador Netscape de 40 bits de encriptación. Para obtener agilidad en el cómputo, utilizaron una oficina llena de estaciones de trabajo, las cuales fueron dejadas trabajando durante las vacaciones de verano en Francia. Para poder romper el código de esta muestra transcurrió poco más de una semana, lo cual resultó ser un poco más que el promedio de tiempo utilizado con 50 estaciones de trabajo.

[2] Entusiasta de la informática. Un hacker es una persona que siempre está deseando aprender y superar nuevos retos, entre los que se pueden encontrar el acceder a un cierto sistema teóricamente cerrado. Pero esto no quiere decir que se haga con malicia, sino por el propio reto en sí. Cuando se trata de alguien con intenciones maliciosas se suele emplear la palabra "cracker". (http://www.proclave.com/esp/cursos/glosario.htm)

Este resultado contrasta considerablemente con el tiempo promedio que tomaría lograrlo con una sola estación de trabajo, el cual es de 10 meses. Sin embargo, cincuenta estaciones de trabajo no serían difíciles de adquirir por un precio razonable por alguna pequeña compañía o particulares. Un atacante especialmente acaudalado e impaciente podría adquirir más o más veloces estaciones de trabajo y reducir el tiempo de ataque considerablemente.

El algoritmo de encriptación DES ha sido utilizado por la industria bancaria, incluso por el sistema de la reserva federal de Estados Unidos desde su introducción en 1975. La tabla 2.1 muestra la buena resistencia de una llave de 56 bits al ataque de fuerza bruta en estaciones de trabajo típicas. Seguramente las máquinas serían un montón de chatarra y polvo para el final de los 57,000 años en promedio que toma el descifrar un código cifrado utilizando DES. Por otro lado, el uso de hardware optimizado y combinado con un cómputo paralelo masivo podría cambiar el panorama.

Por ejemplo, la última fila de la tabla 2.1 es un ejemplo teórico de una propuesta de Michael Wiener de Bell-Northern Research. Wiener trabajó sobre el diseño de un dispositivo basado en la tecnología disponible que ejecutara búsquedas tipo fuerza bruta para descifrar llaves DES. Wiener estimó que la versión mínima de la máquina costaría 100,000 dólares y podría descifrar una llave en 35 horas. Por lo tanto, con un millón de dólares se podría producir una máquina proporcionalmente más robusta que rompiera la encriptación del DES en tres horas y media. Dado el acelerado y constante avance tecnológico, el costo y la velocidad del rompimiento de códigos por medio de fuerza bruta tenderá a mejorar con el paso del tiempo.

Algunas fuentes han cuestionado el hecho de que la máquina propuesta por Weiner representa un proyecto realista, especialmente en el aspecto de costo de componentes y escalabilidad. Sin embargo, la cada vez más acelerada historia de la tecnología demuestra que la propuesta de Weiner es precisamente el tipo de artefacto que es más factible conforme la tecnología va madurando. Los costos para dispositivos digitales han caído en un porcentaje significativo cada año por décadas, y el desempeño ha mejorado similarmente en proporción. El diseño propuesto de un millón de dólares de Weiner probablemente continuará como un dispositivo caro de propósito específico que captura el interés de agencias de inteligencia de gobiernos o a las grandes organizaciones. Aunque los atacantes solitarios aun tienen pocas posibilidades de lograr su desempeño, el dispositivo se puede utilizar como punto de comparación para el poder criptoanalítico del ataque de fuerza bruta. La máquina de Weiner puede probar casi 3 trillones de llaves por segundo.

Algoritmo	No. de bits	No. de llaves	Tiempo de búsqueda con un dispositivo de Weiner
DES	56	10^{16}	3.5 hrs.
SKIPJACK	80	10^{24}	6,655 años
Triple DES con 2 llaves	112	10^{33}	3×10^{13} años
IDEA, RC4	128	10^{38}	2×10^{18} años

Tabla 2.2

La tabla 2.2 ilustra cómo las llaves de algoritmos criptográficos modernos resisten un dispositivo como la máquina propuesta por Weiner. Esto nos puede dar una idea del poder con el que aun cuentan las llaves de estos algoritmos para preservar la confidencialidad e integridad de la información.

2.2 Niveles de encriptación

La encriptación puede completarse en tres niveles diferentes de comunicación de datos:

- Nivel de Enlace

En el nivel de enlace, los datos son encriptados inmediatamente antes de ser transmitidos al receptor. En la mayoría de los casos, este proceso es realizado a través del uso de módems que residen en la computadora emisora y la receptora. Estos módems contienen dispositivos de seguridad que realizan el proceso de encriptación. La encriptación al nivel de enlace ocurre como un proceso separado del programa de aplicación. Los datos son encriptados solamente durante el proceso de transmisión. Por lo tanto, en el sistema del emisor los datos están desprotegidos desde el momento en que el programa los genera hasta antes de ser transmitidos.

- Nivel de Sesión

La encriptación en el nivel de sesión ocurre dentro del proceso de soporte de comunicaciones entre un servidor y un cliente o entre dos equipos cualquiera. El programa de aplicación aun no está involucrado en el proceso de encriptación. El proceso de soporte de comunicaciones determina cuáles aplicaciones requieren de la encriptación. Solamente los datos en sí son encriptados, y el enlace no está involucrado en el proceso de encriptación. Aun cuando los datos pueden ser protegidos durante la transmisión entre dispositivos de hardware, no existe protección en el nivel de aplicación.

- Nivel de Aplicación

La encriptación al nivel de aplicación es más segura que al nivel de enlace o al nivel de sesión. En este nivel, el proceso de la encriptación es un

paso a realizar dentro de la misma aplicación. Conforme se van creando datos por la aplicación, éstos son encriptados y después redireccionados al proceso de comunicación. Aun cuando se requiere de mayor coordinación entre los participantes del intercambio, la encriptación al nivel de aplicación es la más utilizada y definida también en el estándar ANSI X12.58.

En cualquier caso, la necesidad del negocio a la que la encriptación atienda, determinará cuál de estos niveles es elegido.

2.3 Componentes de un sistema criptográfico

Un sistema criptográfico moderno tiene varios elementos esenciales que determinan cómo funciona.

- Algoritmo: Primeramente se encuentra el algoritmo criptográfico, el cual especifica la transformación matemática que es realizada a los datos para encriptarla o desencriptarla. Algunos algoritmos son para realizar cifrados continuos, los cuales encriptan un flujo de datos un bit a la vez. Sin embargo, los algoritmos más conocidos son para cifrados de bloque, los cuales transforman los datos en bloques de tamaño fijo, un bloque a la vez. Cuando los cifrados de bloque son aplicados al flujo de datos, el *modo de cifrado* define cómo el algoritmo es aplicado bloque a bloque al flujo de datos.

Un algoritmo criptográfico es un procedimiento que toma los datos en texto plano y los transforma en texto cifrado en un modo reversible. Un buen algoritmo produce texto cifrado que produce el menor número de pistas posible acerca de la llave o el texto plano que lo produjo. El término *algoritmo* es un término matemático generalmente asociado con las ciencias de la computación. En este caso indica que la transformación entre texto plano y texto cifrado puede ser descrita en términos puramente simbólicos implicando la transformación de conjuntos de símbolos matemáticos. En la práctica, se trabaja con los algoritmos criptográficos en los circuitos digitales o en paquetes de software. Si un dispositivo en particular, paquete o producto asegura implementar fielmente un algoritmo en particular, entonces las debilidades en el algoritmo también están presentes en el dispositivo. Las fortalezas del algoritmo deben estar presentes en el dispositivo, asumiendo que no han sido debilitadas por alguna otra característica del dispositivo. A menudo, aunque no siempre, los dispositivos que implementan el mismo algoritmo pueden desencriptar mutuamente el texto cifrado, asumiendo que las llaves correspondientes son utilizadas por ambos.

- Llaves de Encriptación: Son una pieza de información que se usa dentro de un algoritmo de encriptación para hacer de la encriptación o desencriptación un proceso único. Al igual que las contraseñas, el usuario necesita usar la llave correcta para tener acceso o descifrar un mensaje. La llave equivocada descifrará el mensaje en forma ilegible.

- Longitud de la llave: Una longitud predeterminada para la llave. Cuanto más larga sea la llave, más difícil será verse descubierta en un ataque de fuerza bruta, donde se utilizan todas las posibles combinaciones de llaves.

Los sistemas de encriptación efectivos dependen del secreto y de la dificultad de descifrar una llave, de la existencia de puertas traseras por las que un archivo encriptado pueda ser desencriptado sin conocerse la llave, de la capacidad de desencriptar todo un mensaje de texto cifrado si uno conoce la forma en que se desencripta una parte de éste (ataque de texto conocido, mencionado anteriormente) y de las propiedades del texto plano conocido por un perpetrador.

La mayoría de las transacciones encriptadas que viajan a través de Internet usan una combinación de llaves privadas, llaves públicas, llaves secretas, funciones Hash (valores fijos derivados matemáticamente a partir de un mensaje de texto) y certificados digitales, para lograr la confidencialidad, la integridad del mensaje y el no repudio por el remitente o por el destinatario. Este proceso híbrido de encriptación de llave pública/privada permite que los datos sean almacenados y transportados con poca exposición así como los datos corporativos de una compañía están seguros a medida que se desplazan a través de Internet o de otras redes. La combinación de llaves privadas y públicas se relaciona con dos tipos de sistemas criptográficos: los sistemas criptográficos de llave privada y los de llave pública.

2.4 Administración y distribución de llaves

El factor más importante en los procesos de encriptación y autentificación es la llave. Para su manejo dentro de transacciones electrónicas existe el estándar ANSI X9.1, Estándar de Manejo de Llaves para Instituciones Financieras.

2.4.1 Tipos de llaves

Existen generalmente tres clases de llaves: llaves maestras, llaves de encriptación de llaves y llaves de datos.

- Llave maestra

La llave maestra o llave de almacenamiento, como se le llama en ocasiones, es de 128 bits de largo y es utilizada para encriptar otras llaves almacenadas. La llave maestra reside en un dispositivo de hardware resistente a daños que contiene el algoritmo DES o AES.

- Llave de encriptación de llaves

La llave de encriptación de llaves, conocida abreviadamente como KEK (siglas en inglés de *Key Encryption Key*), es de 128 bits de largo (o 64

bits en implantaciones más antiguas) es utilizada por una de las partes involucradas en una transmisión de datos a otra parte, la cual tiene una llave maestra diferente.

- Llave de datos

Las llaves de datos son de 64 bits de largo y son definidas para encriptar, desencriptar y autentificar mensajes. Una vez que las partes han establecido una relación por medio de una KEK, pueden utilizar ésta para intercambiar con seguridad KEKs adicionales y llaves de datos entre sí.

Con una llave de 8 bits o una de 64 bits, los primeros siete bits de cada byte forman parte de la llave. El octavo bit es usado para ajuste de paridad del byte. Si una parte desea intercambiar datos encriptados o autentificados con otra parte, ambas deben usar la misma llave secreta. La llave en común permite poner en marcha el criptosistema. La confidencialidad de la llave de encriptación es muy importante, ya que todo el proceso de encriptación y autentificación depende de esto. Las llaves son almacenadas en formato ilegible, pero el nombre de la llave es legible.

Con una llave de datos, el receptor puede verificar si cualquier dato en cuestión viene de alguien que tiene la misma llave. El receptor realiza esto al obtener la llave apropiada, desencriptándola y autentificando los datos. Si se utiliza una llave de datos incorrecta para la desencriptación, los datos no se desencriptarán correctamente en texto plano. Si se utiliza una llave de datos incorrecta para autentificar los datos, el código de autentificación de mensaje que se genera al enviar los datos no corresponderá con el que se genera al recibirlos. Por esto, la llave de datos es lo primero que un intruso trataría de atacar para vencer el proceso de encriptación. Si el intruso conoce la llave de datos, puede desencriptar datos sin necesidad de autoridad especial o crear un mensaje que aparente provenir de una fuente confiable.

El estándar ANSI X12.58 recomienda que una sola llave de datos sea utilizada para un solo propósito, ya sea encriptación o autentificación, pero no ambos a la vez. Entonces, un intruso tendría que descifrar dos llaves de datos para poder leer y alterar un mensaje específico.

2.4.2 Vigencia de llaves

Como precaución en el manejo de llaves, una llave de datos debe tener un uso limitado establecido previamente. La misma llave debe ser usada solo por un número específico de veces o por un periodo específico de tiempo. Algunas aplicaciones usan llaves de datos solamente una vez por sesión y luego se desechan. A mayor frecuencia de cambio de llaves, menores probabilidades hay de que una llave en particular pueda ser descifrada.

Las llaves con una vigencia limitada tienen lo que se denomina "criptoperiodo". Cada llave, al ser generada, tiene una fecha de vigencia. La

llave no puede ser usada antes de esta fecha y debe ser usada dentro de un tiempo específico después de dicha fecha. Con los criptoperiodos, las llaves pueden, por ejemplo, ser creadas para un día específico y ser usadas sólo ese día. Si un mensaje que ha sido encriptado y autentificado ha sido enviado, el software de seguridad del recipiente debe asegurarse de que cada llave esté dentro de su periodo de vigencia.

2.4.3 Administración de llaves

Los sistemas de llaves pueden tener una arquitectura de dos o de tres capas. En una arquitectura de dos capas, solamente una llave de encriptación de datos (KEK) es distribuida antes de que una llave de datos sea distribuida. Por ejemplo, si A desea enviar datos encriptados y autentificados a B, para establecer una relación criptográfica cada parte debe intercambiar una llave de encriptación de datos (KEK). Partes diferentes de la KEK son comunicadas en texto plano en tiempos diferentes a través de rutas diferentes. Después, cada quien ensambla las partes para crear una KEK completa. En este punto, ambas personas están listas para intercambiar llaves de datos. Conforme se vaya necesitando, A debe crear cada llave de datos, encriptarla utilizando la KEK y transmitirla a B, quien desencripta cada llave de datos con la KEK.
En una arquitectura de tres capas, existe una KEK inicial, la cual es distribuida manualmente, y las múltiples y subsecuentes llaves son distribuidas automáticamente. La distribución automática de KEKs provee de una mayor eficiencia en el manejo de llaves. Si la confidencialidad de una llave de datos en particular es cuestionable, no es necesario cambiar todas las llaves de datos. En lugar de eso, solamente las llaves de datos que comparten una misma KEK con la llave de datos en cuestión tendrían que ser reemplazadas.

El proceso de crear, distribuir, proteger y desactivar llaves se le denomina administración de llaves y generalmente es realizada por administradores de sistema en cada uno de los extremos de la comunicación. Las tareas de administración de llaves deben ser manejadas por dos personas en cada uno de los extremos. De esta forma, se previene que un solo administrador conozca los valores de las llaves y de que las crea o desactive accidentalmente.

Existen dos métodos para crear llaves de encriptación de datos (KEK). En sistemas más antiguos, se crean realizando un proceso de división. Por ejemplo, una llave de encriptación de datos de 64 bits está compuesta de dos partes, cada una de 32 bits de longitud. Un administrador de sistema crea una mitad del valor de la llave y la ingresa en el sistema. El otro administrador crea e ingresa la otra mitad. Cada uno conserva el valor de su parte en secreto. Debido a que cada administrador conoce la mitad del valor de cada llave, es difícil para cualquiera de los dos conocer el valor completo de la llave. Sin embargo, este método no es infalible, ya que entonces un administrador de sistema tendría que averiguar solamente 32 bits para determinar el valor real de la llave.

Un método más seguro para crear llaves de encriptación de datos restringe el conocimiento de cada administrador aun más. Cada administrador crea e ingresa un componente de 64 bits, en lugar de 32 bits. En lugar de simplemente combinar ambos valores, el sistema somete a estos dos componentes a un proceso de OR exclusivo para crear una llave de 64 bits. Aun cuando cada administrador conoce uno de los dos componentes de la llave, es mucho más difícil poder llegar a la versión original de la llave. Este mismo procedimiento se puede aplicar a la creación de llaves de 128 bits. La administración de llaves debe seguir los estándares marcados por ANSI X9.17, referente a la administración de llaves de instituciones financieras.

2.4.4 Distribución de llaves

Para los criptosistemas de llave privada, una vez que la llave ha sido creada, debe ser distribuida ya sea manualmente o automáticamente. La parte emisora generalmente es responsable de esto.

La distribución manual involucra distribuir las llaves por algún tipo de medio, generalmente un diskette o una tarjeta de seguridad. Las llaves son transportadas físicamente por correo o mensajería al destinatario. Cuando el destinatario notifica de la llegada de la llave, ambas partes acuerdan cargar sus respectivas llaves en sus sistemas simultáneamente. Aunque el método de distribución manual requiere de mayor labor por parte de los administradores de sistemas, es el método más común de distribución.

La distribución automática de llaves implica el intercambio de una llave de encriptación de llaves inicial manualmente por correo o mensajería pero el intercambio de las llaves subsecuentes se realiza electrónicamente. Después de que la llave inicial ha sido cargada por ambas partes, el emisor genera KEKs y llaves de datos subsecuentes, y les da formato con un conjunto de transacciones llamado Mensaje de Servicio Criptográfico (MSC). Las llaves son transmitidas dentro del MSC al destinatario. Para poder mandar y recibir MSCs, ambas partes deben tener software compatible y deben estar de acuerdo en el formato de los mensajes. El MSC debe seguir los estándares ANSI X9.17, referente a la administración de llaves de instituciones financieras y el estándar ANSI X9.17 referente al conjunto de transacciones de mensajes de servicios criptográficos.

Existen otros dos métodos de distribución de llaves. En primer lugar, un centro de distribución de llaves puede generar y distribuir llaves para las partes. Alternativamente, un centro de traducción de llaves puede cifrar y descifrar llaves para las partes. En cualquiera de los dos casos, las partes no comparten una KEK en común. En lugar de eso, cada parte comparte una KEK con el centro.

Por otro lado, los sistemas de llave pública no sufren los mismos problemas de distribución de llaves ya que éstas se generan localmente y se genera un par de llaves públicas adicional. Por esto la criptografía de llave

pública se utiliza ampliamente como un método de distribución de llaves alternativo al método manual. Por ejemplo, se puede encriptar una llave privada usando un sistema de llave pública para su distribución segura a través de la red). Sin embargo, normalmente no es recomendable usar criptosistemas de llave pública para realizar toda la labor criptográfica en una red, ya que el desempeño de estos algoritmos generalmente es menor al de los algoritmos de llave privada. Por esta razón, los sistemas de llave privada tienden a ser favorecidos para la encriptación del tráfico de una red.

2.4.5 Criterios de elección del tamaño de llave

La elección del tamaño de llave estará determinada por lo que esté disponible en los productos que se tengan al alcance. El rango más probable puede ser entre 40 y 128 bits, aunque también hay de mayor tamaño, como algunas implementaciones de 168 bits, como el triple DES. La decisión dependerá generalmente de si los productos adecuados tienen el tamaño de llave suficiente para proteger la información.

La siguiente es una lista de preguntas asociadas con la sensibilidad de los datos. Cada respuesta afirmativa a estas preguntas indicará que una llave mayor será preferida sobre una más corta.

1. ¿Se utilizará una sola llave para proteger un número grande de mensajes diferentes y transacciones?

   Por ejemplo, algunos sistemas generarán y usarán una llave diferente para cada sesión o transacción mientras que otros reutilizarán la misma llave por un periodo de tiempo. Cuanto más sea utilizada una llave, más larga debe ser.

2. ¿Puede un atacante causar daños serios al recuperar y usar una llave criptográfica?

   En el peor de los casos, las vidas de otras personas pueden estar en peligro. Si este es el caso, la llave debe ser muy grande y el resto del sistema debe ser de de muy alta calidad.

3. ¿La información en un mensaje es tratada como estrictamente confidencial por los dueños de la misma, sujeto a abuso costoso o raramente publicado excepto en algunos mensajes?
   Por ejemplo, un banco podría asignar un número de cuenta confidencial a un cuentahabiente que sea difícil de cambiar. Otro ejemplo puede ser las contraseñas reutilizables para acceder a bases de datos u otros recursos importantes. Los números de tarjetas de crédito no son un ejemplo de esto, ya que son liberados a cualquier comerciante que acepte la tarjeta para pagar una compra.

4. ¿Previene la llave de la manipulación de activos de información valiosos?

A mayor valor de los activos protegidos por una llave, ésta debe ser de mayor tamaño. Por ejemplo, si la llave protege una venta con tarjeta de crédito que pudiera tener un valor muy alto, la llave debe ser más larga.

Si comenzamos con una llave de 40 bits, podemos sentirnos seguros siempre y cuando la respuesta a todas estas preguntas sea negativa. Cada pregunta a la que se responda afirmativamente indica que se debe utilizar una llave más grande. Conforme van apareciendo respuestas afirmativas, una llave aun mas larga debe utilizarse, en algunos casos hasta los 112 bits del triple DES por ejemplo, o la llave de 128 bits del algoritmo IDEA. La selección del tamaño de llave también se ve afectada por la solución criptográfica utilizada, esto es, la forma en que se utiliza para proteger el tráfico de una red y cómo se integra dentro de varios protocolos.

2.5 Propiedades de un buen algoritmo criptográfico

- No dependencia de la confidencialidad del algoritmo

Aun cuando en algunos casos se aumenta el trabajo del atacante al mantener un algoritmo lo más secreto posible, el mantenerlo de esta forma puede representar un arma de doble filo. Esto es debido a que si la forma de trabajar de un algoritmo permanece secreta, no podemos determinar si tiene alguna falla fácil de explotar. Por ejemplo, el algoritmo "secreto" o "propietario" de un producto puede tener un cifrado simple que genere un texto cifrado que aparente ser bueno, pero que podría ser atacado por alguien competente.

Los buenos algoritmos criptográficos dependen exclusivamente de las llaves para proteger la información. El revelar la forma de trabajar de un algoritmo no aumenta significativamente la probabilidad de éxito de un atacante. Actualmente, los algoritmos más utilizados son los más atacados y rotos, por ejemplo el DES.

- Explícitamente diseñado para la encriptación

El algoritmo debe ser diseñado en primer lugar para resistir el criptoanálisis. Esto no siempre es verdadero en los algoritmos utilizados para la encriptación. Por ejemplo, algunos productos usan generadores de números aleatorios simples para producir un cifrado simple tipo Vernam. Las nociones simples de aleatoriedad estadística no garantizan fortaleza contra el criptoanálisis. Ocasionalmente, un producto o protocolo utilizará un algoritmo "hash" como el MD5 con una llave secreta para producir un cifrado Vernam. Esto no es para lo que el algoritmo MD5 fue diseñado, y sin un análisis serio no hay ninguna razón para asumir que tal cifrado resistirá el criptoanálisis.

- Disponible para análisis

Idealmente, el algoritmo ha sido publicado y sujeto a escrutinio por la comunidad pública criptográfica. A mayor análisis del algoritmo por parte de los matemáticas y criptoanalistas, mayor probabilidad habrá de que encuentren sus debilidades. Por ejemplo, el algoritmo DES ha pasado la prueba del tiempo y es probable que siga siendo utilizado en varios años por venir de una forma o de otra. Resulta menos satisfactorio cuando el desarrollador ofrece el algoritmo para análisis bajo varios acuerdos de confidencialidad y no divulgación, ya que menos expertos lo pueden revisar bajo las condiciones mencionadas, y el tiempo y las circunstancias tienden a limitar la profundidad de la revisión. Muchas veces los expertos son traídos a escena como parte de un truco publicitario, por lo que los recursos disponibles para la revisión muchas veces son dictados por las apariencias. Este tipo de análisis rara vez continúa más allá del punto en el que un reporte menos superficial pueda ser producido.

Es importante cuestionar si reconocidos criptoanalistas han publicado estudios referentes a la fortaleza del algoritmo. Idealmente, reconocidos expertos deberían estar discutiendo abiertamente el algoritmo y publicando análisis en publicaciones profesionales que aseguren que el trabajo está siendo revisado por otros expertos. Esto casi nunca ocurre excepto en los casos en que el algoritmo ha sido publicado.

En otros casos, el algoritmo podría haber sido revisado por expertos reconocidos en criptoanálisis que hayan publicado opiniones referentes a su fortaleza frente a vulnerabilidades reconocidas en códigos similares. Desafortunadamente, este acercamiento sufre de trucos o estrategias de publicidad como en el caso anterior. Siempre es importante juzgar a los expertos y su nivel de experiencia.

- No hay debilidades prácticas

El análisis realizado debe mostrar que no hay debilidades serias en el algoritmo que puedan ser explotadas fácilmente por un atacante. Los algoritmos personalizados incluidos en el software comercial tienden a tener serias debilidades. Si un paquete comercial asegura encriptar datos y no usa un algoritmo reconocido, no se debe asumir que proteja contra algún atacante motivado.

2.6 Criptografía de Llave Privada

Este tipo de criptografía emplea la misma llave para encriptar y desencriptar el mensaje o los datos de control. También se le conoce como *criptografía simétrica*. La simetría se refiere a que las partes tienen la misma llave tanto para cifrar como para descifrar.

La criptografía de llave privada ha sido la más usada en toda la historia. Ha podido ser implementada en diferentes dispositivos, manuales mecánicos, eléctricos, hasta los algoritmos actuales que son programables en cualquier computadora. La idea general es aplicar diferentes funciones al mensaje que se quiere cifrar de tal modo que sólo conociendo una llave pueda aplicarse de forma inversa para poder así descifrar. Aunque no existe un tipo de diseño estándar, quizá el más popular es el de Fiestel, que consiste esencialmente en aplicar un número finito de interacciones de cierta forma, dando finalmente como resultado el mensaje cifrado. Este es el caso del sistema criptográfico simétrico más conocido, DES.

Los algoritmos simétricos han tenido una muy buena aceptación gracias a la tecnología de los ordenadores, que permiten hacer computaciones elevadas sea cual sea la longitud de bits elegida. Se caracterizan por ser altamente eficientes (con relación al tamaño de su llave) y robustos. Sin embargo, la tendencia es de utilizarlos poco o para cuestiones que no necesiten un alto grado de protección. Los productos actuales combinan técnicas de llave privada y pública para poder explotar los beneficios de cada una.

La desventaja de estos algoritmos puede ser el hecho de que dos personas dispondrían de una llave, que deberían comunicarse por un medio imposible de interceptar (canal seguro), como por ejemplo pasarla en mano. Sin embargo, no siempre es fácil disponer de ese tipo de canal. Por otra parte, seguiría necesitándose una llave para cada par de personas, por lo que se necesitaría un gran número de llaves cuando hay un número mayor de usuarios.

2.6.1 Tipos de cifrado

Los algoritmos simétricos encriptan bloques de texto aplicando cifrados de bloques. El tamaño de los bloques puede ser constante o variable según el tipo de algoritmo.

A continuación se describen los tipos más usuales de cifrado encontrados en algoritmos criptográficos. El cifrado es el proceso por medio del cual el texto plano pasa a ser codificado de tal forma que se produzca texto ininteligible (texto cifrado) para poder ser transmitido a través del canal de comunicaciones. Algunos algoritmos realizan cifrados de flujo, los cuales encriptan un flujo de datos un bit a la vez. Sin embargo, los algoritmos más conocidos realizan cifrado por bloques, el cual transforma los datos en bloques de determinado tamaño, un bloque a la vez. Cuando el cifrado por bloques es aplicado al lujo de datos, el *modo de cifrado* define cómo el algoritmo es aplicado bloque por bloque al flujo de datos.

2.6.1.1 Cifrado continuo o de flujo

Los algoritmos de cifrado continuo son diseñados para aceptar una llave criptográfica y un fragmento continuo de texto plano para producir un fragmento equivalente en texto cifrado (fig. 2.1). El cifrado continuo digital clásico es el cifrado Vernam, desarrollado a principios del siglo XX para encriptar tráfico telegráfico. El cifrado Vernam original utilizaba una llave criptográfica guardada en un rollo muy largo de cinta. Cada bit del texto plano en el flujo de texto tomaba el siguiente bit de la cinta que contenía la llave, y el texto cifrado era producido sumando el bit del texto plano y el de la llave, descartando el bit de acarreo. Esta operación de suma es la misma que la operación lógica XOR (OR exclusivo). Esta suma produce un solo bit de texto cifrado, y la operación se realiza sucesivamente hasta el final del mensaje.

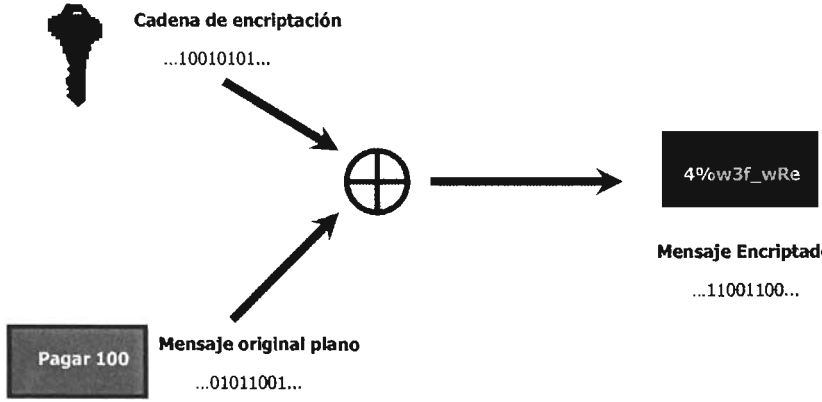


Fig. 2.1 Cifrado continuo o de flujo

En la práctica, los cifrados Vernam pueden ser fáciles de romper. Un atacante puede matemáticamente combinar mensajes de texto cifrado en pares para obtener por factorización la llave y producir un mensaje que sea más fácil de descifrar. Si el atacante descifra el mensaje extrayendo el texto plano del texto cifrado, el texto plano se puede usar para deducir la llave. Una vez hecho esto, el atacante puede descifrar todos los demás mensajes protegidos por esa llave.
Este pequeño ejemplo ilustra la importancia de los detalles en las debilidades del código al construir sistemas criptográficos efectivos. No se pueden evitar los errores a menos que se revisen las debilidades de técnicas previas y actuales. El conocimiento de técnicas para rompimiento de códigos provee una guía para identificar las características de un código fuerte.

El cifrado Vernam continúa siendo un cifrado importante, aun con sus debilidades. Se vuelve débil cuando se utilizan llaves repetitivas, pero puede ser muy efectivo cuando la llave varía constantemente. Esta técnica es utilizada con frecuencia en los cifrados continuos modernos. Estos consisten de dos partes: un procedimiento para generar una secuencia de bits que las

personas ajenas no puedan adivinar y un cifrado continuo Vernam usando esa secuencia como llave. Se puede tomar la llave aleatoria generada a su extremo lógico y usar una nueva llave generada aleatoriamente para cada bit de cada mensaje que se envíe. Esto produce un tipo especial de cifrado denominado *cifrado de una sola vez*. Un resultado importante de este cifrado es que resulta imposible extraer el texto plano de un texto cifrado por este método si la llave es realmente aleatoria y nunca se reutiliza.

Actualmente, pocos cifrados continuos son utilizados en sistemas comerciales. El más conocido es RC4 (Rivest Cipher #4), desarrollado por Ron Rivest de la compañía RSA Data Security. RC4 sigue siendo ampliamente utilizado en parte porque el gobierno de los Estados Unidos permite que sea exportado si el software asociado usa una llave aceptablemente corta.

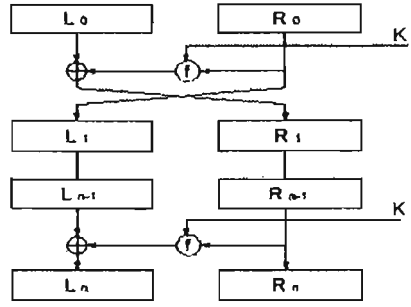
2.6.1.2 Cifrado por bloques

El cifrado por bloques está diseñado para tomar bloques de datos de un tamaño en particular para ser encriptados con una llave de tamaño particular y producir un bloque de texto cifrado de cierto tamaño. Los cifrados por bloques actuales generan un bloque de texto cifrado del mismo tamaño que el texto plano. Muchos de los cifrados por bloques tienen en común que dividen un bloque de longitud n en dos mitades, L y R. Se define entonces un cifrado de producto iterativo en el que la salida de cada ronda se usa como entrada para la siguiente según la relación:

$$\left.\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus f(R_{i-1}, K_i) \end{aligned}\right\} \quad \text{si } i < n.$$

$$\begin{aligned} L_n &= L_{n-1} \oplus f(R_{n-1}, K_n) \\ R_n &= R_{n-1} \end{aligned}$$

Este tipo de estructura se denomina Red de Feistel, y es empleada en multitud de algoritmos, como DES, Lucifer, FEAL, CAST, Blowfish, etcétera. Tiene la interesante propiedad de que para invertir la función de cifrado basta con aplicar el mismo algoritmo, pero con las K_i en orden inverso. Además, esto ocurre independientemente de cómo sea la función f.



La tabla 2.3 ilustra los algoritmos más conocidos que realizan el cifrado por bloques, así como sus propiedades de llaves. El algoritmo DES es el más conocido de los que realizan cifrado por bloques. Ha sido un estándar nacional en Estados Unidos y muy ampliamente utilizado a nivel mundial. Este hecho le ha dado un interés especial a muchos usuarios, pues demuestra que gobiernos de potencias mundiales lo utilizan para proteger datos confidenciales y comerciales.

Algoritmo de cifrado por bloques de llave privada	Tamaño del bloque de datos (bits)	Tamaño de la llave (bits)	Utilizado actualmente
DES	64	56	Sí (está siendo reemplazado por AES de 128 bits)
IDEA (International Data Encryption Algorithm)	64	128	Sí
MMB (Modular Multiplication Block Cipher)	128	128	No
CA-1.1 (Cifrado de autómata celular)	384	1088	No
SKIPJACK	64	80	Sí

Tabla 2.3 Algoritmos simétricos de cifrado por bloques

Los cifrados por bloques son analizados y probados por su habilidad de encriptar bloques de datos del mismo tamaño que se haya determinado. Un cifrado razonable debería generar un texto cifrado que tenga la menor cantidad de propiedades discernibles. Un análisis estadístico del texto cifrado generado por el algoritmo de cifrado por bloques debe encontrar que los bits de datos individuales así como los patrones de bits aparezcan completamente aleatorios. Los patrones no aleatorios son lo primero que un intruso busca, ya que pueden proveer la vía de entrada para romper un código.

Los patrones en el texto cifrado se convierten en un problema cuando se aplican cifrados por bloque a los flujos de datos. Si se encripta el mismo bloque dos veces con la misma llave, se obtiene el mismo texto cifrado. La figura 2.2 muestra un ejemplo de esto. La operación encripta bloques de cuatro caracteres cada uno. Al encriptar el mensaje "UNO POR UNO", la palabra "UNO" aparece dos veces, cayendo ambas veces en un extremo del bloque encriptado. Esto produce el texto cifrado "4%3f" dos veces, por lo que se produce un patrón repetitivo en el texto cifrado.

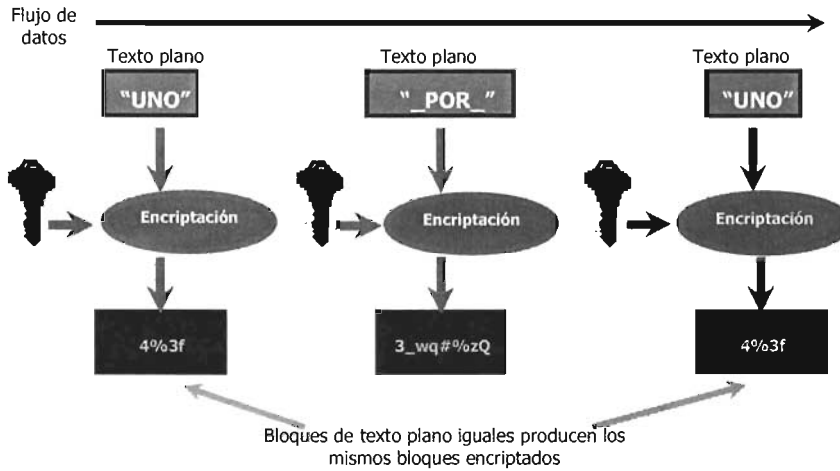


Fig. 2.2 Cifrado por bloques

Cantidades razonables de texto plano en comunicaciones en general producen secuencias repetidas como la del ejemplo mencionado. Si encriptamos un flujo de datos bloque por bloque, patrones en el texto plano producirán patrones significativos estadísticamente en el texto cifrado. Esto puede significar la entrada necesaria para que un atacante descifre el código.

Otro posible problema peor en las aplicaciones comerciales es que un atacante con algún conocimiento de los contenidos de un mensaje puede sustituir bloques encriptados de un mensaje en los de otro mensaje. Por ejemplo, un atacante podría cortar y pegar campos entre los mensajes de depósito bancario y retiro para producir un mensaje benéfico para sí mismo, sin haber logrado descifrar las llaves.

El término "modo de cifrado" se refiere a un conjunto de técnicas usadas para aplicar un cifrado por bloques a un flujo de datos. Varios modos han sido desarrollados para ocultar los bloques repetidos y así mejorar la seguridad de los cifrados por bloque. Cada modo define un método de combinar el texto plano, la llave y el texto cifrado encriptado de una manera especial para generar el flujo de texto cifrado que se transmite al recipiente. Los estándares publicados para el DES incluyen especificaciones para utilizarse con los modos estándar de cifrado. En teoría podrían existir incontables maneras de combinar las entradas y salidas de un cifrado. En la práctica, y de acuerdo al estándar de encriptación ANSI X9.23, existen cuatro modos básicos de cifrado por bloques son usados:

1. ECB (Electronic Code Book, o libro de código electrónico)
2. CBC (Cipher Block Chaining, o encadenamiento de cifrado por bloques)
3. CFB (Cipher Feedback, o retroalimentación de cifrado)
4. OFB (Output Feedback, o retroalimentación de la salida)

A continuación se describen cada uno de estos modos:

- ECB (Electronic Code Book, o libro de código electrónico): ECB es el caso ilustrado en la figura 2.2. El cifrado es simplemente aplicado al texto plano bloque por bloque. Es el modo más eficiente. Puede ser acelerado utilizando hardware paralelo y a diferencia de otros modos, no requiere un dato extra como semilla para un ciclo de retroalimentación. Sin embargo, ECB tiene problemas de seguridad que limitan su utilización. Los patrones en el texto plano pueden producir patrones en el texto cifrado. También es fácil modificar un mensaje cifrado añadiendo, removiendo o cambiando bloques encriptados.

- CBC (Cipher Block Chaining, o encadenamiento de cifrado por bloques)

El modo CBC esconde patrones en el texto plano combinando sistemáticamente cada bloque de texto plano con un bloque de texto cifrado antes de encriptarlo. Los dos bloques son combinados bit a bit usando una operación de OR exclusivo. En lugar de encriptar directamente los datos, el cifrado por bloques de este modo encripta el texto plano después de que ha sido combinado con el texto cifrado de apariencia aleatoria (fig 2.3).

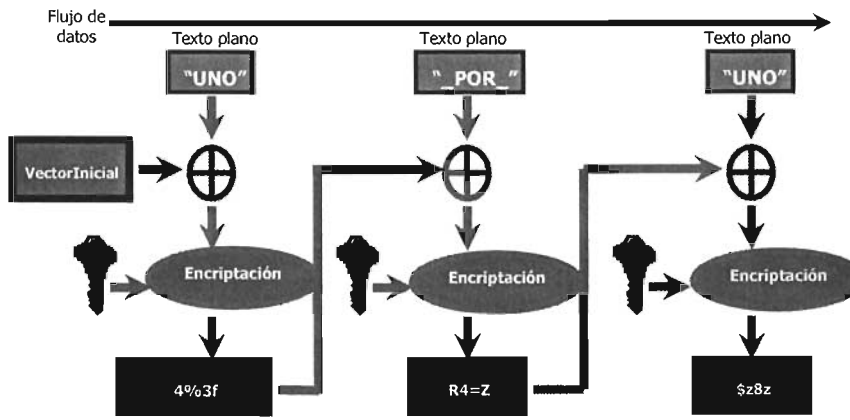


Fig. 2.3 Modo de cifrado CBC

Para poder garantizar que siempre haya texto cifrado de apariencia aleatoria para aplicar al texto plano, el proceso se inicia con un bloque de bits aleatorio llamada *vector de inicialización* (mejor conocido como IV, o "Initialization Vector"). Dos mensajes no producirán nunca el mismo texto cifrado, aun si los dos bloques de texto plano son idénticos, siempre que el IV sea diferente para cada mensaje. La mayoría de las implementaciones de CBC añaden el IV al principio del mensaje en texto plano.

Una desventaja de CBC en aplicaciones de red es que los mensajes encriptados pueden ser sólo el doble del tamaño de los del mismo mensaje en EBC. Un bloque es añadido para transmitir el IV al recipiente. La

desencriptación apropiada depende del IV para empezar el proceso de retroalimentación. Algunas aplicaciones sólo transmiten el IV en un paquete inicial y dependen del recipiente para mantener un control del último bloque desencriptado. Esto puede afectar la eficiencia de la desencriptación ya que impide al recipiente desencriptar inmediatamente paquetes que lleguen fuera de orden.

CBC es el modo más utilizado porque tiene propiedades de seguridad razonablemente buenas. Los patrones en el texto plano son ocultos como se pretendía. Además, la interrelación entre bloques de datos causada por el encadenamiento puede dificultar el modificar mensajes de una manera útil. Sin embargo, existen riesgos de ataques de tipo "copiar y pegar" contra mensajes de formato libre como el correo electrónico. Por ejemplo, un intruso puede tomar una secuencia de bloques de un mensaje diferente encriptado con la misma llave e insertarlos en otro mensaje. El proceso de CBC generaría un bloque de texto basura donde los datos fueron insertados y el resto del mensaje se desencriptaría normalmente. Un mensaje también se desencriptaría correctamente si el intruso lo trunca al principio o al final. Esto significa un riesgo para la integridad de los datos, ya que un intruso podría modificar una transacción a su favor si trunca un mensaje o si inserta basura en el mismo para evitar alguna transacción, por mencionar unos ejemplos.

- CFB (Cipher Feedback, o retroalimentación de cifrado)

El modo CFB es similar al CBC en el sentido de que "alimenta" de nuevo el bloque de texto cifrado a través del cifrado por bloques. Sin embargo, el CFB es diferente porque el cifrado por bloques es usado para generar una constantemente cambiante llave que encripta el texto plano con un cifrado Vernam. En otras palabras, los bloques de texto cifrado son sumados por medio de un OR exclusivo con bloques sucesivos de datos generados por el cifrado para así generar el texto cifrado (fig. 2.4). Este modo también es llamado Cifrado de Llave Automática (CTAK).

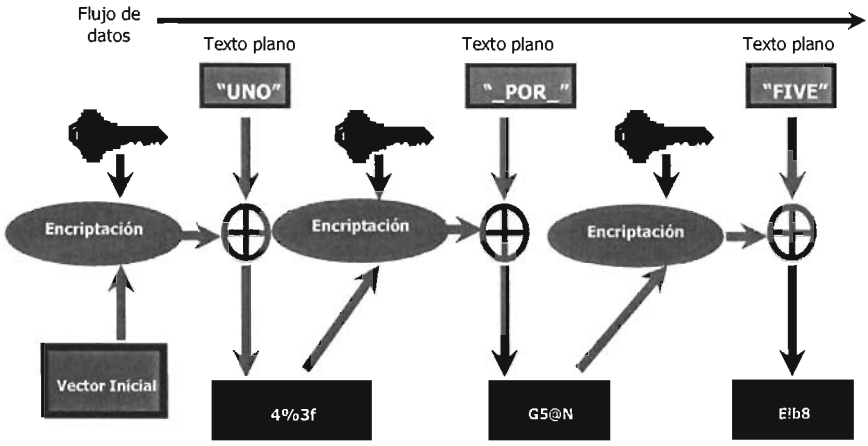


Fig. 2.4 Modo de cifrado CFB

Un beneficio en particular del CFB es que no está limitado al tamaño de bloque del texto cifrado, ya que el modo puede ser adaptado para trabajar con bloques más pequeños hasta bits individuales. Un problema con CFB es que debe operar con bloques totalmente llenos. Esto hace de CFB más eficiente en términos de tamaños de mensaje y manejo de tráfico encriptado. Al igual que CBC, los mensajes de CFB requieren de un valor de un vector de inicialización.

El texto cifrado resultante tiene propiedades de seguridad comparables a las de CBC. Los intrusos pueden tratar de copiar y pegar nuevos mensajes a otros encriptados con la misma llave. Esos mensajes producirán una cantidad pequeña de basura en el texto plano en el punto en el que el copiado y pegado tuvo lugar, y se desencriptarán correctamente después de eso.

.

- OFB (Output Feedback, o retroalimentación de la salida)

El modo OFB es similar a CFB pero un poco más simple. Este modo utiliza el cifrado por bloques por sí solo para generar las llaves Vernam (fig. 2.5). A diferencia de los otros dos modos, el flujo llave (producido mediante la encriptación de la llave con el IV) no depende del flujo de datos. Ni el texto plano ni el texto cifrado son alimentados de vuelta al proceso de encriptación. En este modo, el cifrado por bloques sólo es utilizado para generar las llaves. A este modo también se le conoce como "modo de autollave".

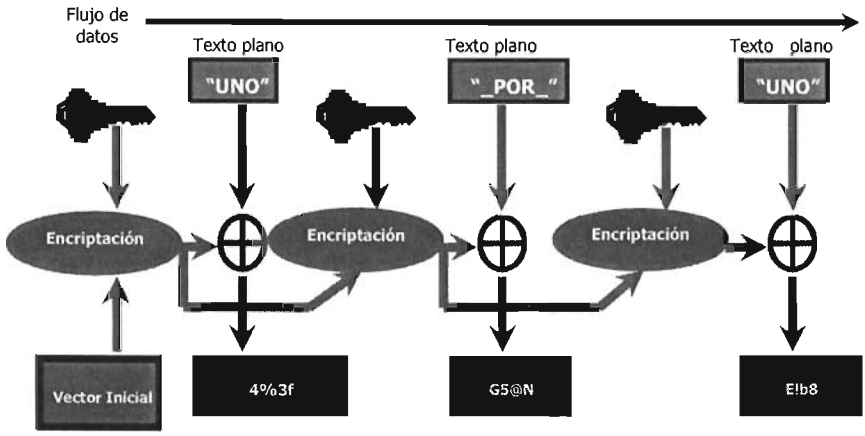


Fig. 2.5 Modo de cifrado OFB

Al igual que CFB, la longitud del texto plano no debe necesariamente de caber exactamente dentro de los límites del bloque. Cada mensaje requiere de un IV, por lo que el texto cifrado debe incluir un bloque extra de datos que contenga el IV.

Mientras que el texto cifrado del OFB tiene mejores propiedades de seguridad que ECB, tiene defectos en comparación con otros modos. En particular, existe una relación directa entre bits en el texto plano y bits en el texto cifrado, por lo que un mensaje que contenga algún fragmento de texto plano conocido es vulnerable a alteraciones. Un ataque de tipo sobreescritura como éste puede pasar desapercibido y nunca deja basura en el texto plano desencriptado.

Aun cuando es posible construir otros modos más sofisticados para propósitos especiales, la mejor elección será siempre uno de estos cuatro modos de cifrado. Los defectos mencionados para algunos modos no son siempre relevantes para todas las aplicaciones. Las propiedades operacionales y de manejo de errores podrían hacer de modos atractivos una mala elección para algunas aplicaciones.

Si se utiliza un cifrado por bloques, su resistencia a los ataques depende en parte de si se está utilizando el modo correcto con el mismo. Desafortunadamente, todos estos modos producen texto cifrado que puede ser manipulado de una u otra forma. Si la autenticidad del mensaje es crucial, entonces el sistema criptográfico debe aplicar pro separado la protección de la integridad. A continuación se mencionan algunas aplicaciones de comunicaciones de datos y los modos sugeridos para soportarlas.

- Protocolos de comunicaciones arbitrarios con datos arbitrarios: CBC, CFB.

Si por ejemplo, se aplica un cifrado por bloques a paquetes a través de

Internet que contengan datos arbitrarios los modos CBC y CFB son las mejores elecciones. El texto plano repetido sería oculto efectivamente. El constante cambio en las llaves de encriptación en estos modos vence los ataques cripotanalíticos aun si el atacante es capaz de generar un texto plano escogido. Existe el riesgo de ataques de "copiar y pegar", pero probablemente dejarían basura residual en los mensajes desencriptados.

- Protocolos con protección criptográfica de integridad: CBC, CFB, OFB

Si el protocolo incorpora protección criptográfica de integridad, entonces OFB podría ser agregado a la lista de modos de cifrado aceptables. Los mensajes protegidos por OFB son especialmente vulnerables a ser modificados por un atacante que conoces algunos de los contenidos de los mensajes de texto plano, además de que a diferencia de los ataques de copiar y pegar en otros modos, las modificaciones no dejan rastro alguno.

- Pequeñas cantidades de datos realmente aleatorios: ECB

Si la información que está siendo encriptada es realmente aleatoria, entonces el modo ECB puede ser utilizado seguramente. Por ejemplo, ECB puede ser utilizado para encriptar material generado aleatoriamente que esté siendo enviado a través de la red. ECB nunca debe ser utilizado para encriptar información que pudiera haber sido proporcionada por un atacante, ya que esto podría apoyar un ataque criptoanalítico de texto conocido. Los otros tres modos podrían ser utilizados con seguridad para esta situación, pero no son recomendados en primera instancia por razones de eficiencia.

2.6.2 Algoritmos utilizados en la Criptografía de Llave Privada

2.6.2.1 DES (Data Encryption Standard)

Es el sistema criptográfico de llave privada más común. El DES es un algoritmo diseñado por IBM, publicado por la Oficina Nacional de Estándares (NBS) de Estados Unidos, y utilizado habitualmente desde los años 70. Es un método de cifrado altamente resistente frente a ataques criptoanalíticos diferenciales. La vulnerabilidad citada con mayor frecuencia es su tamaño de llave de 56 bits, que lo hace vulnerable a ataques de fuerza bruta. Un ataque "patrocinado" por la empresa de seguridad RSA Data Security Inc. contra un mensaje con cifrado DES requirió el uso de cientos de ordenadores durante 140 días. Sin embargo, hay diseños de máquinas que, con un costo de un millón de dólares, podrían descifrar mensajes DES en poco tiempo, como se mencionó anteriormente.

DES toma como entrada un bloque de 64 bits del mensaje y este se somete a 16 interacciones. DES usa bloques de 64 bits, en la práctica el bloque de la llave tiene 56 bits, ya que hay 8 bits de control (corrección de errores). Cualquier número de 56 bits puede usarse como llave y hay 2^{56}

llaves posibles en el espectro de llaves. Cuando damos un bloque de texto plano de 64 bits, DES opera de la siguiente forma:

- Realiza una transposición inicial, agrupando los 64 bits siguiendo un patrón de modo que el bloque resultante es dividido en dos mitades de 32 bits.
- Repite el proceso, teniendo en cuenta el resultado anterior, 16 veces basado en la llave por medio de un proceso de sustitución, transposición y operaciones matemáticas XOR (OR exclusivo).
- Después de completar las 16 rondas, las dos mitades resultantes son unidas nuevamente, y una transposición final, que es la inversa de la transposición inicial, completa la operación de codificación.

En la actualidad no se ha podido romper el sistema DES desde la perspectiva de poder deducir la llave simétrica a partir de la información interceptada. Sin embargo, DES ya no es considerada una solución criptográfica fuerte ya que todo su espectro de llaves puede ser violado por la técnica de "fuerza bruta", con la cual se prueban todas las llaves posibles, usando grandes sistemas de computación en período breve de tiempo. Por ejemplo, en 1998 un grupo de interés privado construyó una computadora de propósito especial para romper el DES, que costó menos de $250,000 dólares con la cual se pudo romper en Enero de 1999. Esa computadora rompió en 56 horas una llave de 56 bits de DES buscando el 24.8 por ciento del espectro de llaves, u 88 billones de llaves por Segundo. De igual modo, una llave DES también ha sido rota usando los esfuerzos combinados de un gran número de computadoras haciendo procesamiento paralelo. Lo anterior quiere decir que es posible obtener la llave del sistema DES en un tiempo relativamente corto, por lo que lo hace inseguro para propósitos de alta seguridad. La opción que se ha tomado para poder suplantar a DES ha sido usar lo que se conoce como cifrado múltiple, es decir aplicar varias veces el mismo algoritmo para fortalecer la longitud de la llave. Esto ha tomado la forma de un nuevo sistema de cifrado que se conoce actualmente como **Triple DES** o **3DES**.

2.6.2.2 3DES o Triple DES

Es una técnica mediante la cual se aplica tres veces el algoritmo DES al bloque de texto plano, potencialmente utilizando una llave de 56 bits diferente en cada vez. Las formas más típicas usan un par de llaves DES de 56 bits, K1 y K2, siendo K1 utilizada para una primera encriptación, K2 para realizar una desencriptación, la cual al ser realizada con una llave diferente a K1 nos dará como resultado un texto diferente al texto plano original, siendo enseguida K1 reutilizada para la operación final de encriptación, produciendo así una longitud total de llave de 112 bits. Algunas aplicaciones llegan a utilizar tres llaves diferentes, produciendo así un tamaño de llave total de 168 bits, el cual es considerablemente grande para un algoritmo de llave privada. Esto sugiere que si se intenta atacar por fuerza bruta, una llave de este

tamaño ofrecería suficiente protección contra este tipo de ataques como para que el algoritmo se volviera obsoleto.

El funcionamiento de 3DES consiste en aplicar 3 veces DES de la siguiente manera: la primera vez se usa una clave **K1** junto con el bloque **B0**, se realiza la encriptación DES, obteniendo el bloque **B1**. La segunda vez se toma a **B1** con la clave **K2**, diferente a **K1** de forma inversa, realizando la desencriptación y la tercera vez a **B2** con una clave **K3** diferente a **K1** y **K2**, se realiza nuevamente la encriptación, es decir, aplica de la iteración 1 a la 16 a **B0** con la clave **K1**, después aplica de la 16 a la 1, a **B1** con la clave **K2**, finalmente aplica una vez mas de la 1 a la 16 a **B2** usando la clave **K3**, obteniendo finalmente a **B3**. (fig. 2.6)

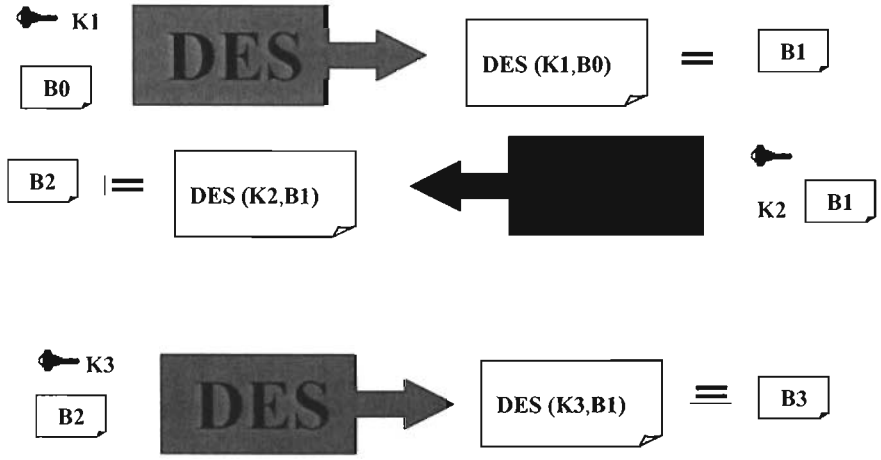


Fig. 2.6 Proceso de cifrado de 3DES

Este sistema 3DES usa entonces una clave de 168 bits, aunque se ha podido mostrar que los ataques actualmente pueden romper a 3DES con una complejidad de 2^{112}, es decir efectuar al menos 2^{112} operaciones para obtener la clave por fuerza bruta.

2.6.2.3 IDEA (International Data Encryption Algorithm)

IDEA, o Algoritmo de Encriptación de Datos Internacional, es un algoritmo simétrico que realiza cifrado por bloques, surgido en 1990. Fue desarrollado por Xuejia Lai y James Massey en el Instituto Federal Suizo de Tecnología. Trabaja con bloques de texto de 64 bits, operando siempre con números de 16 bits usando operaciones como XOR y suma y multiplicación de enteros. IDEA es más eficiente de implementar en software que el DES o el 3DES y su llave de 128 bits lo hace más atractivo que el DES convencional. IDEA puede ser usado con los modos usuales de cifrado por

bloques y es probablemente el mejor algoritmo de bloques existente. Se han realizado numerosos intentos de aplicar técnicas criptoanalíticas a este algoritmo, sin embargo sus desarrolladores han demostrado que resiste el criptoanálisis mejor que DES. Otros análisis han detectado que formas más simples de IDEA son vulnerables a ciertos ataques, pero que son eliminadas cuando se aplica el algoritmo completamente.

Este algoritmo es de libre difusión y no está sometido a ningún tipo de restricciones o permisos internacionales, por lo que se ha difundido ampliamente, utilizándose en sistemas como UNIX y en programas de cifrado de correo como PGP.

2.6.2.4 RC4 (Rivest Cipher #4)

RC4 es un algoritmo de cifrado por flujo desarrollado por Ron Rivest y comercializado por RSA Data Security. Originalmente el código fue protegido bajo secreto comercial, pero posteriormente su código fuente ha sido publicado en grupos de discusión. Consiste básicamente en realizar una serie de operaciones XOR (OR exclusivo) y transposiciones a un arreglo auxiliar de 16x16, cuyos valores son posteriormente sometidos a operaciones XOR con los valores de la llave y el mensaje, para producir así el texto cifrado. Este algoritmo forma una parte vital del sistema de cifrado en capas SSL, ampliamente utilizado en navegadores de Internet tales como Netscape Navigator y Microsoft Internet Explorer. Por desgracia, la versión exportable de RC4 tiene una llave de solamente 40 bits, lo que lo hace altamente vulnerable a de ataques fuerza bruta. La versión EEUU, con llave de 128 bits, es inmune a ataques criptoanalíticos.

2.6.2.5 RC5 (Rivest Cipher #5)

El sistema criptográfico simétrico RC5 es el sucesor de RC4, frente al que presenta numerosas mejoras. RC4 consiste en hacer un XOR al mensaje con un arreglo que se supone aleatorio y que se desprende de la llave, mientras que RC5 usa otra operación llamada dependencia de datos, para obtener así el mensaje cifrado. Este algoritmo fue también desarrollado por RSA Data Security Inc., que es actualmente una de las empresas más importantes en el campo de los sistemas de cifrado y protección de datos.

RC5 permite diferentes longitudes de llave (aunque está prohibida su exportación fuera de EEUU con longitudes superiores a 56 bits), y funciona como un generador de números aleatorios que se suman al texto mediante una operación de tipo OR-Exclusiva. Es además ampliamente configurable, permitiendo fijar diferentes longitudes de llave, número de iteraciones y tamaño de los bloques a cifrar, por lo que le permite adaptarse a cualquier aplicación. Por ejemplo, este algoritmo también es usado por Nestcape para implementar su sistema de seguridad en comunicaciones SSL (Secure Socket Layer) como un sustituto del RC4.

En cuanto a su seguridad, al igual que el RC4, el RC5 con longitudes de llave de 56 bits no es lo suficientemente seguro, a diferencia de la versión EEUU como en el caso de RC4, que tiene una mayor longitud de llave. Ambos han sido ya rotos en diversos concursos publicados en línea por diferentes equipos alrededor del mundo. En el caso de RC5 se han descifrado mensajes encriptados con llaves de hasta 64 bits, aunque la propia compañía RSA impulsa nuevos desafíos cada vez con una mayor longitud de llave.

2.6.2.6 SKIPJACK

Es un algoritmo de cifrado por bloques desarrollado por la NSA (National Security Agency). Encripta bloques de 64 bits utilizando una llave de 80 bits. Se encuentra implementado en chips como los utilizados en la tarjeta criptográfica *Fortezza*, la cual es una tarjeta de PC que contiene un procesador criptográfico y cuenta con almacenamiento de llaves.

El algoritmo SKIPJACK es clasificado, lo cual limita dramáticamente el conocimiento de sus propiedades criptográficas esenciales. Sin embargo, de acuerdo a reportes de fuentes no verificables, el algoritmo es tan resistente como los mejores algoritmos comerciales, y que su funcionamiento se mantiene en secreto para proteger las técnicas de diseño de la NSA. La NSA promueve el SKIPJACK para proteger comunicaciones militares en el Sistema de Mensajes de Defensa (DMS) y para proteger información clasificada en algunos casos. Esto refleja una medida de confianza que es consistente con los reportes acerca de este algoritmo.

2.6.2.7 AES (Advanced Encryption Algorithm)

El NIST (National Institute of Standards Technology) de Estados Unidos convocó a un concurso para poder tener un sistema simétrico que sea seguro y pueda usarse al menos en los próximos 20 años como estándar. Las principales características que se pidieron de AES son que al menos fuera tan seguro y rápido como TDES, es decir, que al menos evitara los ataques conocidos, además de que pudiera ser implementado en una gran parte de aplicaciones.

A este concurso se presentaron numerosos autores, y tras un largo proceso de selección fue seleccionado como futuro estándar el denominado Rijndael, creado por los belgas Vincent Rijmen y Joan Daemen.

Rijndael es un cifrador de bloque que opera con bloques y llaves de longitudes variables, que pueden ser especificadas independientemente a 128, 192 ó 256 bits. En comparación, DES cifra bloques de 64 bits con una clave de 56 bits solamente. El triple DES utiliza normalmente cifras con bloques de 64 bits con una clave de 112 bits. El esquema de llave del AES consiste en dos operaciones, expansión de llave y selección de llave de vuelta de cifrado, y el proceso de cifrado consta de tres pasos: una adición

inicial de la llave de vuelta, i-1 vueltas de cifrado y una vuelta final.

El principio de funcionamiento del AES está descrito en la figura 2.7. En primer lugar, se añade bit a bit al mensaje la llave secreta K_0 . Después, como para todos los algoritmos de cifrado por bloques, se itera una función F, parametrizada por subclaves que se obtienen de la clave maestra mediante un algoritmo de expansión de llaves. En el caso de AES, se itera 10 veces la función F.

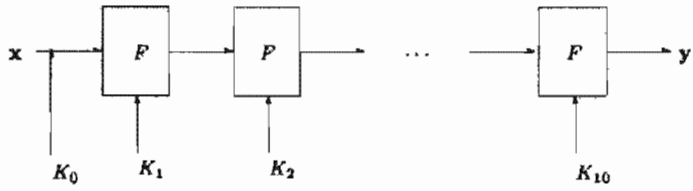


Fig. 2.7 Iteraciones del AES

La figura 2.8 describe cómo se itera la función F para cifrar. Se toma como entrada bloques de 128 bits repartidos en 16 octetos. Antes de nada, se aplica a cada octeto la misma permutación S. Seguidamente, se aplica a los 16 octetos una segunda permutación P. Entonces al resultado obtenido se le añade bit a bit la subclave de 128 bits obtenida por el algoritmo de expansión de llave.

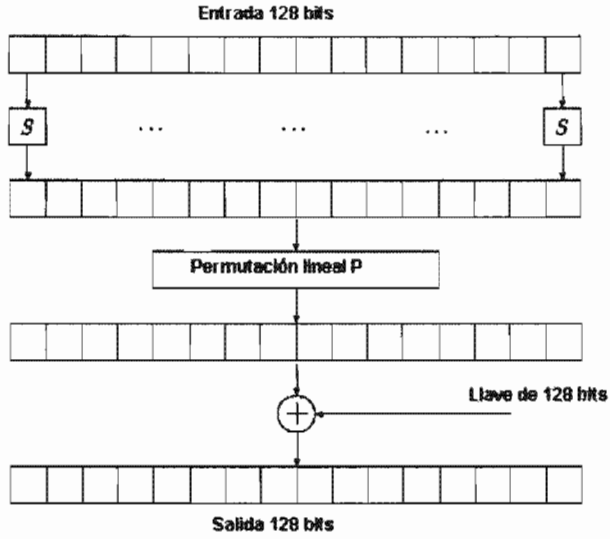


Fig. 2.8 Función F

El algoritmo de expansión de llave permite calcular la llave K_i de la i-ésima iteración en función de la subllave K_{i-1} de la (i-1)-ésima iteración,

siendo K_0 la llave secreta. Esto se describe en la figura 2.9. Los 16 octetos de la clave K_{i-1} se toman de 4 en 4. Los últimos 4 octetos se permutan usando una permutación S, que es la misma que se utiliza para permutar los bits de cada octeto de la función. Después se suma al primer octeto del nuevo conjunto el elemento α^i. Este elemento es un octeto que depende del número i de la iteración considerada. Por último, para obtener K_i, se añade bit a bit los 4 octetos obtenidos a los 4 primeros octetos de K_{i-1}, después el resultado obtenido es agregado a los 4 octetos siguientes y así sucesivamente.

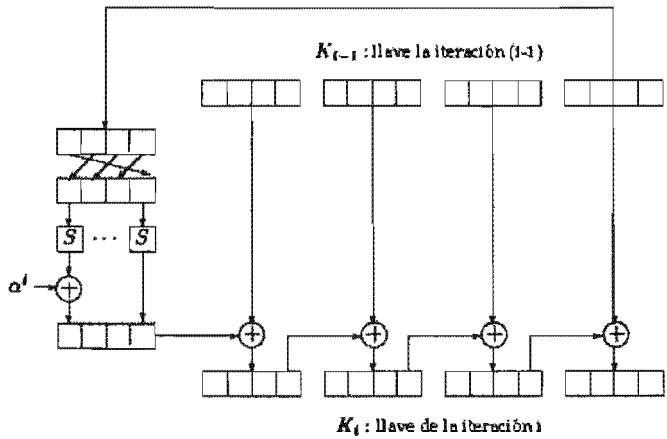


Fig. 2.9 Cálculo de la llave K_i

Técnicamente y por simplicidad, un octeto se puede considerar como un elemento de un conjunto con 256 elementos llamado cuerpo finito, sobre el que existen todo tipo de operaciones simples, entre otras las operaciones de adición, producto e inversión. La permutación S anteriormente mencionada corresponde a la inversión sobre este conjunto. La permutación P está especificada como una operación muy simple, implementándose fácilmente. El elemento α^i corresponde a elevar a la potencia i un elemento del cuerpo. Estas consideraciones permiten implementar AES de forma muy eficaz.

Como el AES no comprende más que operaciones muy simples sobre los octetos se tienen dos ventajas enormes:

- La implementación por software de AES es extremadamente rápida. Por ejemplo, una implementación en C++ sobre un Pentium a 200MHz permite cifrar a 70Mbits/s.
- Es un algoritmo resistente al criptoanálisis tanto lineal como diferencial y uno de los más seguros en la actualidad.

2.6.2.8 Otros algoritmos de llave privada

- Blowfish: Creado por Bruce Schneier, autor del libro "Applied Criptography". Utiliza llaves de hasta 448 bits y, hasta el momento, ha resistido con éxito todos los ataques. Por ello y por su estructura se le considera uno de los algoritmos más seguros, a pesar de lo cual no se utiliza masivamente. Tal vez se deba a su relativa juventud. Su autor no ha patentado el método para que pueda ser empleado sin limitaciones.

- CAST (Carlisle Adams y Stafford Tavares): Tiene estructura similar a la de Blowfish. Parece ser un buen algoritmo, aunque tampoco lleva el tiempo suficiente como para haber sido atacado hasta la saciedad. De momento, sus posibilidades son buenas. Se conocen ataques criptoanalíticos contra la versión de clave 64 bits, aunque distan mucho de ser eficaces (requieren 2^17 textos sin cifrar y 2^48 cálculos diferentes). No se conocen ataques contra la versión de 128 bits. Ha sido patentado por Entrust Technologies, quienes permiten el uso libre de este algoritmo.

- SAFER: Algoritmo diseñado por Robert Massey (uno de los creadores de IDEA). Tiene claves de hasta 128 bits y, a pesar de algunas debilidades en la primera versión y de ciertos ataques, parece un algoritmo seguro. Este programa fue desarrollado para la empresa Cylink, que algunos ligan a la no muy querida Agencia de Seguridad Nacional norteamericana (NSA); por ello, hay quien no se fía.

2.6.3 Ventajas y desventajas de la Criptografía de Llave Privada

Hay dos ventajas principales a favor de los criptosistemas de llave privada como DES o AES. La primera es que el usuario tiene que recordar o saber únicamente una llave tanto para la encriptación como para la desencriptación. La segunda es que los criptosistemas de llave privada son generalmente menos complicados y por lo tanto usan menos energía de procesamiento que las técnicas de llave pública. Esto hace que los sistemas criptográficos de llave privada sean idealmente adecuados para la encriptación de datos en bloque. La principal desventaja de este método es cómo lograr que las llaves lleguen a manos de aquellos con quienes uno quiere intercambiar información, en particular en los entornos de comercio electrónico, donde los clientes son entidades desconocidas en las que no se confía.

En este sentido, los espacios de llave criptográfica privada son susceptibles de ser descifrados. La ley de Moore establece que el poder de cómputo (es decir, el número de transistores por pulgada cuadrada en los circuitos integrados) se duplica aproximadamente cada 18 meses. Por lo tanto, los costos asociados con el diseño de computadoras para fines especiales como montar un ataque de fuerza bruta a un espacio de llaves

decrecerán en un factor de 10 aproximadamente cada cinco años más o menos. Debido a estos adelantos, DES está siendo reemplazado por AES, como se mencionó anteriormente. Bajo la tecnología de la actualidad, no es factible que llaves de tamaños entre 128 y 256 bits sean rotas incluso con una inversión de cientos millones de dólares y probablemente podrán proteger a los activos de información por muchos años.

2.7 Criptografía de Llave Pública

Los sistemas criptográficos de llave pública, desarrollados para la distribución de llaves, resuelven el problema de hacer que llaves simétricas lleguen a las manos de dos personas que no se conocen entre sí, pero que quieren intercambiar información en una forma segura. Sobre la base de un proceso de encriptación asimétrica o de llave pública, dos llaves funcionan juntas como un par. Una llave se usa para encriptar datos, y la otra se usa para desencriptarlos. Cualquiera de las llaves se puede usar para encriptar o para desencriptar, pero una vez que una llave se ha usado para encriptar datos, sólo su pareja puede ser usada para desencriptarlos.

Se dice que las llaves son asimétricas porque están relacionadas inversamente entre sí. Sobre una base de una factorización integral matemática, la idea es generar un solo producto de dos números primos, donde no es posible factorizar el número y recuperar los dos factores. Este proceso de factorización integral constituye la base para la criptografía de llave pública (es decir, función fácil de computar en una dirección, pero muy difícil o impráctica en la otra). El sistema involucra aritmética modular, exponenciación y números primos grandes de miles de bits de longitud. Como las llaves son números grandes (por ejemplo, de 1024 bits), se usan tanto para mensajes cortos como para encriptar llaves simétricas de DES o para crear firmas digitales.

2.7.1 Algoritmos utilizados en la Criptografía de Llave Pública

2.7.1.1 Diffie-Hellman

Este algoritmo de encriptación de Whitfield Diffie y Martin Hellman supuso una verdadera revolución en el campo de la criptografía, ya que fue el punto de partida para los sistemas asimétricos, basados en dos llaves diferentes, la pública y la privada. Fue desarrollado en 1976.

Su importancia se debe sobre todo al hecho de ser el inicio de los sistemas asimétricos, ya que en la práctica sólo es válido para el intercambio de claves simétricas, y con esta funcionalidad es muy usado en los diferentes sistemas seguros implementados en Internet, como SSL (Secure Socket

Layer) y VPN (Virtual Private Network).

Matemáticamente se basa en las potencias de los números y en la función mod (módulo discreto de una división). Uniendo estos dos conceptos se define la potencia discreta de un número como $Y = X_a \bmod q$. Si bien el cálculo de potencias discretas es fácil, la obtención de su función inversa, el logaritmo discreto, no tiene una solución analítica para números grandes.

Para implementar el sistema se realizan los siguientes pasos:

Se busca un número primo muy grande, q.
Se obtiene el número ß, raíz primitiva de q, es decir, que cumple que ß mod q, ß2 mod q,...., ßq-1 mod q son números diferentes.
ß y *q* son las claves públicas.

Para generar una llave simétrica compartida entre dos usuarios, A y B, ambos parten de un generador de números pseudoaleatorios, que suministra un número de este tipo diferente a cada uno, X_a y X_b. Estos son las llaves privadas de A y B. Con estos números y las llaves públicas ß y *q* que ambos conocen, cada uno genera un número intermedio, Y_a e Y_b, mediante las fórmulas:

$$Y_a = ß^{Xa} \bmod q$$

$$Y_b = ß^{Xb} \bmod q$$

Estos números son intercambiados entre ambos, y luego cada uno opera con el que recibe del otro, obteniendo en el proceso el mismo número ambos:

$$K = Y_b{}^{Xa} \bmod q$$

$$K = Y_a{}^{Xb} \bmod q$$

Este número K es la clave simétrica que a partir de ese momento ambos comparten, y que pueden usar para establecer una comunicación cifrada mediante cualquiera de los sistemas simétricos. Se puede representar de manera elemental la relación entre estos números en la figura 2.8.

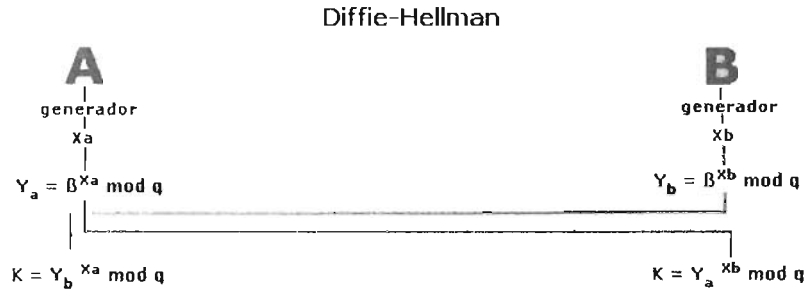


Fig. 2.10 Generación de llaves en el algoritmo Diffie-Hellman

Con este esquema, si se desea compartir una clave privada con otro usuario cualquiera, basta con acceder a su Y_u y enviarle la nuestra. Para facilitar este proceso se suelen publicar las Y_u de todos los usuarios interesados en un directorio de acceso común. El punto crucial del algoritmo es que ambos usuarios obtendrán el mismo resultado numérico y nadie más podrá obtener el mismo resultado a partir de la información pública. No existen formas de encontrar los exponentes secretos X_a y X_b, lo cual es igual al problema de calcular logaritmos discretos en un campo finito, que no tienen forma de calcularse en estos casos. El resultado de K se puede utilizar entonces con cualquier algoritmo de llave privada.

En la práctica, el algoritmo Diffie-Hellman es utilizado fundamentalmente para el intercambio y protección de llaves. Por ejemplo, los protocolos de intercambio de llaves propuestos para el IPSEC usan este algoritmo.

Una variante del algoritmo Diffie-Hellman, *ElGamal*, provee de funciones similares a las que provee el algoritmo RSA: encriptación y obtención de firmas digitales. Sin embargo, la encriptación tiene la característica peculiar de generar el doble de texto cifrado que el texto plano. El Estándar de Firmas Digitales (DSS) establecido por el NIST utiliza una variante de ElGamal.

2.7.1.2 RSA

El algoritmo de clave pública RSA fue creado en 1977 por Rivest, Shamir y Adelman, y es el sistema criptográfico asimétrico más conocido y usado. Los desarrolladores se basaron en el artículo de Diffie-Hellman sobre sistemas de llave pública, crearon su algoritmo y fundaron la empresa RSA Data Security Inc., que es actualmente una de las más prestigiosas en el entorno de la protección de datos. Básicamente la seguridad del RSA se fundamenta en que es relativamente fácil multiplicar dos números primos grandes y casi imposible factorizar cantidades que son producto de dos números primos largos.

La técnica de RSA consiste en producir llaves públicas que estén relacionadas con llaves privadas específicas. Cada usuario obtiene un par de claves, una pública y otra privada. La llave pública de cada persona será difundida de tal manera que puede ser conocida por cualquiera. Un usuario A que desee enviar un mensaje a un usuario B utilizará la llave pública de B para cifrar el mensaje que desea enviarle además de su propia llave secreta (la de A). La llave secreta se mantiene en secreto por el propietario de la misma. Con esta llave se podrán descifrar los mensajes que lleguen. Así con el ejemplo anterior, B utilizaría la llave pública de A y su propia llave secreta para descifrar el mensaje enviado por A.

Con el esquema anterior, se puede observar que únicamente las llaves públicas serán difundidas y podrán ser conocidas por cualquiera. Las llaves privadas no son transmitidas o compartidas. La base de este criptosistema está en que no se puede obtener la llave privada del usuario receptor a partir de su llave pública.

RSA también permite al remitente encriptar un mensaje con su llave privada para que cualquiera que tenga su llave pública pueda desencriptarlo. Esto naturalmente no provee de confidencialidad, pero sí autentificación por medio de las firmas digitales, con las cuales se puede demostrar que quien realizó el cifrado y envío de un mensaje es quien dice ser.

El sistema RSA crea sus llaves de la siguiente manera:

1. Se buscan dos números primos lo suficientemente grandes: **p** y **q** (de entre 100 y 300 dígitos).
2. Se obtienen los números **n** = p * q y $\varphi(n)$ = (p-1) * (q-1).
3. Se busca un número **e** que sea relativamente primo a $\varphi(n)$, es decir, que no tenga factores comunes con $\varphi(n)$ aparte de 1.

4. Se calcula **d**, de modo que e*d mod ($\varphi(n)$) = 1.

Este enunciado implica que el resto de dividir e*d / $\varphi(n)$ debe ser igual a 1. Para que esto sea posible, el producto resultante de e*d debe ser un número suficientemente cercano a $\varphi(n)$ como para que al dividir ambos su resto sea 1. Por ejemplo, si $\varphi(n)$ = 160 y e=7, un número cercano sería 161, el cual si se divide entre 160 quedará como resto 1.

Esto se puede representar como e*d = $\varphi(n)$ + 1, por lo que si despejamos *e*, podemos obtener el valor de *d* como d = ((p-1) * (q-1)+1) / e. Si utilizamos los números de ejemplo, resulta que d = 23, por lo que si sustituimos los valores e y d queda entonces: 7*23 mod (160) = 1 por lo que se cumple la condición **e*d mod ($\varphi(n)$) = 1.**

Los valores de **e** y **d** son los exponentes público y privado, respectivamente. La llave pública será entonces el par (n,e) y la llave privada el par (n,d). Los números p, q y (p-1) * (q-1) se desechan para evitar que alguien más pueda obtener la llave privada a través de los mismos.

El cálculo de estas llaves se realiza en secreto en la máquina en la que se va a guardar la clave privada, y una vez generada ésta, conviene protegerla mediante un algoritmo criptográfico simétrico.

RSA hace uso de una propiedad descubierta por Euler, la cual dice que si *M* es un número cualquiera relativamente primo a *n* entonces:

$(M^e)^d = M \pmod n$ y $(M^d)^e = M \pmod n$

Si alguien quiere mandar un mensaje *m* a algún usuario, obtiene su llave pública (n,e), y podrá mandar el mensaje cifrado *c*, que solo podrá descifrar el usuario correspondiente. El mensaje *m* es convertido a su valor numérico. Un buen método para hacerlo puede ser a través de los valores binarios de cada carácter (en ASCII) convirtiendo, posteriormente, el valor binario final a decimal. Así, por ejemplo, las letras XYZ son los valores ASCII 88 (1011000), 89 (1011001) y 90 (1011010). Se realiza esta codificación por bloques y cada bloque se somete a la siguiente operación (donde e es constante y público):

$$c = m^e \bmod n$$

Cuando la información cifrada llega a su destino el receptor procede a descifrar el mensaje con la siguiente fórmula:

$$m = c^d \bmod n$$

El resultado *m* será entonces el valor ASCII correspondiente, con lo cual se podrán revertir los bloques a su forma alfabética y conjuntarlos para recuperar el mensaje original.

Estas formulas son inversas y por lo tanto dan el resultado deseado, (n,e) son llave pública, la llave privada es la pareja (n,d). La relación que existe entre d y e es que uno es el inverso multiplicativo del otro módulo $\lambda(n)$ donde $\lambda(n)$ es el mínimo común múltiplo de p-1 y q-1

En cuanto a las longitudes de llaves, el sistema RSA permite longitudes variables, siendo aconsejable actualmente el uso de claves de no menos de 1024 bits. El último récord[3] conocido sobre factorización de números enteros producto de dos primos de la misma longitud es de 155 (512 bits) dígitos alcanzado en Julio de 1999, aunque se necesitaron más de 5 meses y casi 300 procesadores trabajando juntos para hacerlo, usando 160 estaciones SGI y Sun de 175-400 MHz, 8 procesadores Origin de 250 MHz de SGI, 120 Pentium II de 300-450 MHz, y 4 Compaq Alphas de 500 MHz, un

[3] RSA Data Security, http://www.rsasecurity.com/rsalabs/challenges/factoring/rsa155.html

total de aproximadamente 8000 Mips[4]-años.

RSA basa su seguridad en ser una función computacionalmente segura, ya que si bien realizar la exponenciación modular es fácil, su operación inversa, la extracción de raíces de módulo (p-1) * (q-1) no es factible a menos que se conozca la factorización de d, clave privada del sistema. La seguridad dependerá del tamaño del módulo *n*. Cuanto más grande sea, mejor será en cuanto a seguridad. Como contra tenemos que la velocidad será bastante pobre cuanto mayor sea este número.

RSA es el más conocido y usado de los sistemas de llave pública, y también el más rápido de ellos. Presenta todas las ventajas de los sistemas asimétricos, incluyendo la firma digital, aunque resulta más útil a la hora de implementar la confidencialidad el uso de sistemas simétricos, por ser más rápidos. Se suele usar también en los sistemas mixtos para encriptar y enviar la clave simétrica que se usará posteriormente en la comunicación cifrada.

Aun cuando RSA puede ser vulnerable al ataque de fuerza bruta al igual que los demás algoritmos, su fundamento matemático también lo hace vulnerable a otros ataques. Estos incluyen ataques contra la confidencialidad de los mensajes y contra técnicas de generación de llaves públicas. Se han identificado ataques contra datos encriptados y firmas digitales, sin embargo, se pueden evitar mediante un uso cuidadoso de RSA.

Existen dos formas de aplicación del algoritmo RSA. La utilización de cada una depende del propósito para el cual se desee utilizar el algoritmo. Así, existen dos formas comunes y se llaman el esquema de firma y el esquema de cifrado. El esquema de firma se detallará en el siguiente capítulo. El esquema de cifrado se usa principalmente para cifrar llaves de sistemas simétricos (claves de 128 bits aproximadamente) y utiliza el procedimiento descrito con anterioridad. A continuación se describe el esquema de cifrado:

1.- Se toma el mensaje **M** (por ejemplo una clave simétrica de 128 bits), como en la practica actual es recomendable usar arreglos de longitud de 1024 bits, se complementan esos 128 bits con una serie de técnicas para obtener un arreglo de 1024 bits, después se aplica un proceso de codificación para que la computadora entienda al mensaje como un número entero *m*.

2.- Se le aplica la fórmula de cifrado de **RSA** al entero *m*.

3.- Se envía el número entero *c*.

4.- Al recibir este número se aplica la formula de descifrado al entero *c* para obtener el entero *m*.

5.- Se decodifica *m* para obtener el mensaje **M**

[4] MIPS.- Millones de instrucciones por segundo

Ejemplo:

- Generación de parámetros

$p = 3$, $q = 5$ (se eligen dos números primos como clave privada)
$n = 15$ (se calcula el producto, que es la llave pública)
$\varphi(n)=(3-1)(5-1)=8$
Sea $e = 3$, entonces $d = 3$, ya que $e*d = 3*3 = 9 \bmod 8 = 1$ (como este caso solo es para mostrar el funcionamiento no importa que d sea igual a e, sin embargo en la práctica e es pequeño y d es muy grande)
Sea el mensaje codificado en forma numérica $m = 2$

- Proceso de cifrado

El mensaje cifrado es $c = m^e \bmod n$, es decir, $c = 2^3 \bmod 15$, o sea $c = 8$

- Proceso de descifrado

Para descifrar el mensaje $m = 8^3 \bmod 15$, es decir, $m = 512 \bmod 15$, así $m=2$ (ya que $512/15=2 \bmod 15 = m$)

Por lo tanto es correcto el funcionamiento.

El método que actualmente es usado para aplicaciones en el esquema de cifrado es el OAEP, este resiste a los ataques que actualmente se conocen y el estándar más conocido sobre RSA es el PKCS#1 v.2 de la compañía RSA Data Security.

- Vulnerabilidades de RSA

Existen varias vulnerabilidades del algoritmo RSA, las cuales consisten en diferentes tipos de ataque, pero que pueden ser prevenidos tomando precauciones como las que se detallan a continuación.

  - Ataques propiciados por un tamaño de llave privada reducido

    Existe un atajo matemático para recuperar una llave privada si ésta es relativamente pequeña y la llave pública correspondiente es relativamente grande. Debido esto, las aplicaciones que las emplean típicamente usan una llave pública pequeña (generalmente 3 o 65,537) por lo que nunca resulta una llave privada que sea vulnerable a este ataque.

  - Ataque por tamaño de texto plano pequeño

    Si el valor de la llave pública es 3 y la longitud del mensaje es menos de un tercio de la longitud del módulo n, entonces existen atajos para tratar de recuperar el mensaje. Si el mensaje es más pequeño que la raíz cúbica de n, entonces el mensaje puede ser recuperado simplemente obteniendo la raíz cúbica de n. El formato de cifrado

PKCS previene que esto suceda forzando el bloque de mensaje a cifrar a ser siempre más grande que *n*.

- Ataque de exponente bajo de encriptación

Si un mensaje en particular o una serie de mensajes similares matemáticamente son encriptados con el mismo valor bajo para *e*, por ejemplo 3, entonces existe la posibilidad de un ataque para recuperar el texto plano. Sin embargo, este ataque se invalida si todos los mensajes contienen texto aleatorio entre palabras de tal forma que dos mensajes no sean similares. El cifrado PKCS previene esto poniendo texto aleatorio en todos los mensajes encriptados con RSA.

- Ataque de texto cifrado escogido

Consiste en que el atacante, para poder descifrar un mensaje encriptado con la llave pública del destinatario y del cual desconoce la llave privada, intercepta el mensaje y envía un mensaje aparte relacionado matemáticamente con el enviado. El destinatario al desencriptar este mensaje, encuentra que produce basura como resultado, pero el atacante, si recibe esta aparente basura de vuelta, puede combinarla con el mensaje original interceptado para producir el mensaje en texto plano. Este ataque es mucho menos frecuente ya que además requiere del uso de ingeniería social por parte del atacante, para lograr que el destinatario desencripte el mensaje relacionado y reenvíe el resultado al atacante.

De estos ataques potenciales se puede concluir que RSA no es simplemente un algoritmo que provee consistentemente de seguridad. Debe ser usado muy cuidadosamente de tal forma que ninguna de las vulnerabilidades sea explotada por intrusos. Cualquier implementación debe seguir los estándares establecidos por el PKCS y aplicar las operaciones criptográficas en una manera controlada. Si el sistema es muy flexible en cuanto a encriptación y desencriptación, un intruso puede tomar ventaja de esto.

2.7.1.3 ElGamal

Algoritmo desarrollado por Taher ElGamal. Utiliza llaves de tamaño variable entre 512 y 1024 bits. Provee de funciones similares a las de RSA, como la generación de firmas digitales. Sin embargo, la encriptación tiene la característica peculiar de generar el doble de texto cifrado a comparación del texto plano. El Estándar de Firmas Digitales (DSS) establecido por el NIST de Estados Unidos utiliza una variante del algoritmo de firmas digitales de ElGamal.

A diferencia de RSA, su uso ha sido limitado debido a una disputa de patentes con los desarrolladores del algoritmo Diffie-Hellman. Sin embargo,

posterior a la expiración de la patente del Diffie-Hellman, han aparecido algunos sistemas criptográficos basados en ElGamal.

2.7.2 Ventajas y desventajas de la Criptografía de Llave Pública

Una de las ventajas de la criptografía de llave pública se refiere a la posibilidad de simplificar la administración de llaves, este último uno de los grandes problemas, en el ambiente de redes, de la criptografía de llave privada. Se requiere sólo una cantidad mínima de información compartida entre usuarios y permite a un usuario arbitrariamente mandar una llave a otro usuario específico, esto sin la necesidad de que los usuarios tengan que intercambiar llaves manualmente. Lo único que el remitente necesita es la llave pública del destinatario y de esta forma el remitente puede transmitir de manera segura la llave privada. De esta forma, la técnica RSA hace uso de las mejores características de la criptografía de llave pública y privada.

La criptografía de llave pública permite enviar un mensaje encriptado a un recipiente específico, pero es vulnerable a ataques si se usa de manera general para encriptar mucha información, ya que las llaves son relativamente pequeñas en comparación con el volumen de información confidencial que se desee transmitir. En contraste, la criptografía de llave privada es más rápida y segura, pero su seguridad radica en qué tanto las llaves privadas sean secretas.

La criptografía de llave pública presenta otra ventaja sobre la criptografía de llave privada o secreta. Si *n* personas desean comunicarse mediante un criptosistema de llave secreta, cada una de ellas debe disponer de una llave diferente para cada persona del grupo. Por tanto hace falta poder generar en total n(n-1) llaves. Teniendo en cuenta que *n* puede ser del orden de varios millares, será necesario generar archivos de varios millones de llaves. Además, añadir un miembro al grupo no es sencillo, ya que habrá que generar otras n llaves para que el nuevo miembro pueda comunicarse con los demás integrantes del grupo, y después distribuir las nuevas llaves a todo el grupo. Por el contrario, en el caso de un criptosistema asimétrico, se guardan las *n* llaves públicas de los miembros del grupo en un directorio. Para añadir un miembro, basta con que ponga su llave pública en el directorio, lo cual es más manejable en redes de distintos tamaños. Esto ha dado lugar a la elaboración de arquitecturas que incluyen certificados de llave pública y servidores de directorio que los contienen.

En los párrafos anteriores, se ha visto que la criptografía de llave pública aporta solución a muchos más problemas que la criptografía de llave secreta. Sin embargo, los criptosistemas de llave pública presentan vulnerabilidades como las mencionadas anteriormente para el algoritmo RSA, además de otras desventajas:

Por una parte, una razón práctica. La mayoría de los sistemas de cifrado de clave pública son muy lentos. Por ejemplo, RSA es varios centenares de veces más lento que el AES implementándose en software y

es completamente imposible implementarlo en hardware. Actualmente donde la velocidad de la transmisión de la información constituye un punto crucial, el algoritmo de cifrado no puede ser el factor limitante.

El proceso de generación de llaves puede ser también una fuente de vulnerabilidad. Esto es debido a que las llaves se generan a partir de dos números primos muy grandes (p y q), los cuales generalmente se producen por medio de un generador de números pseudoaleatorios, los cuales parten sus cálculos de un valor semilla, el cual debe ser verdaderamente aleatorio. Típicamente, p y q son generados iniciando con dos valores completamente aleatorios que se someten a un procedimiento de generación de primos. Este procedimiento toma un valor binario aleatorio y calcula un número primo del mismo. Esto generalmente involucra generar un número primo candidato y después se aplican una serie de pruebas estadísticas para asegurar que el número es realmente primo. El software de generación de llaves de RSA generalmente debe ceder entre la eficiencia en la generación de llaves y la seguridad. El software que genera llaves más grandes para proteger información sensitiva tendrá que aplicar más pruebas para asegurar que cada número sea primo. A mayor cantidad de pruebas, más existe la probabilidad de que los números sean realmente primos. Si la aplicación tiene requerimientos de seguridad menores, se necesitarán menos pruebas y tiempo de procesamiento. Por otro lado, la posibilidad de crear una tabla de posibles números primos posibles para el rango de llaves de 512 bits y realizar una búsqueda de diccionario para así atacar las llaves privadas queda descartada, pues existiría un estimado de 10^{151} números primos posibles, y una búsqueda de diccionario excede las capacidades físicas y de procesamiento de una búsqueda exhaustiva.

El tamaño de las llaves necesario en criptografía de llave pública para tener un nivel de seguridad satisfactoria es mayor que el tamaño de llaves en criptografía de llave secreta. En realidad la noción y la importancia del tamaño de llave para lograr la seguridad no tienen sentido más que en el caso de la llave secreta. De hecho, esos sistemas se basan en la hipótesis de que los únicos posibles ataques sean los de fuerza bruta. Por ejemplo, en el caso de una clave de 128 bits, el tamaño del espacio a enumerar es 2^{128}.

En el caso de criptosistemas de llave pública, el tamaño de clave no tiene sentido más que cuando se considera el mismo sistema. La única medida válida para evaluar un criptosistema de llave pública es la complejidad del mejor ataque conocido. La mayor diferencia es que jamás se está al abrigo de ataques teóricos. Por ejemplo, el sistema RSA de 512 bits es mucho menos seguro que un AES de 128 bits. Esto es debido a que para obtener la llave pública, un intruso sólo debe de probar los valores potenciales para p y q contra el módulo n. En contraste, para encontrar la llave privada, el intruso necesita recolectar cantidades adecuadas de texto cifrado junto con una noción de lo que pudiera tener el texto plano. Además, existirá la incertidumbre de saber si al realizar una operación de desencriptación, ésta haya sido correcta, a menos que se pueda anticipar si el producto es texto codificado en ASCII, en binario, o en algún otro formato conocido. Si se está utilizando criptografía de llave pública y privada en

conjunto, es importante escoger un tamaño de llave en ambas que sea comparablemente fuerte. La llave pública debe ser al menos igual de difícil de romper que la llave privada y para algunas aplicaciones la llave pública debe ser más fuerte. La tabla 2.4 presenta una lista de llaves privadas junto con el tamaño equiparable de llaves públicas que podría presentar a un criptoanalista un reto difícil. Los tamaños de llave están basados en modelos para estimar las cantidades de esfuerzo criptoanalítico publicadas por el Consejo Nacional de Investigación de Estados Unidos.

Tamaño de Llaves Secretas		Tamaño de Llaves Públicas	
# de Bits	Algoritmos	# de Bits	Aplicaciones
56	DES	256	
70		384	PGP antiguo, tamaño mínimo
80	SKIPJACK	512	DSS corto, PGP "low grade"
96		768	PGP "high grade"
112	3DES con 2 llaves	1,024	DSS largo, PGP "military grade"
128	RC4	1,440	
150		2,047	PGP "alien grade"
168	3DES con 3 llaves	2,880	
192	Energía solar anual	3,000	Energía solar anual

Tabla 2.4 Llaves privadas y llaves públicas de fortaleza comparable

La llave pública es un blanco más valioso que una llave pública porque generalmente es usada como llave para encriptar llaves privadas. Si los intrusos atacan la llave pública, pueden interceptar y descifrar el tráfico subsecuente de mensajes si logran descifrar los mensajes de intercambio de llaves privadas que los antecedieron. En cambio, si los intrusos descifran una llave privada, solamente pueden tener los mensajes encriptados con esa llave. En muchos la llave será desechada al terminar la transmisión, por lo que los intrusos no podrán utilizarla para descifrar o alterar más tráfico en la red. Aunque la llave pública no encripte tanta información como la privada, su efectividad es crucial.

Sin embargo, aun a pesar de las vulnerabilidades de algoritmos de llave pública como el RSA, su fortaleza permanece por mucho, casi intacta cuando se utilizan números suficientemente grandes para calcular las llaves. Aun con las computadoras más veloces, la factorización de números primos largos es muy difícil y puede tomar billones de años para factorizar. Por ejemplo, un número de 200 dígitos requeriría 665 bits de almacenamiento. Para factorizar un número de 665 bits usando los algoritmos más veloces, se requerirían alrededor de 1.2×10^{23} operaciones. Asumiendo que se cuenta con una velocidad de procesamiento de 10^{10} operaciones por segundo, esta factorización requeriría 1.2×10^{13} segundos o 380,267 años de tiempo de procesamiento de CPU. Si se duplica el tamaño del número (400 dígitos) resulta en aproximadamente 8.6×10^{15} años para factorizar el número. Es por esta razón que la única forma en que se ha podido factorizar el número n y obtener las llaves para descifrar un mensaje ha sido mediante el esfuerzo coordinado de varios procesadores alrededor del mundo y durante varios

meses, generalmente necesitando del patrocinio de alguna organización para su realización, por lo que se vuelve improbable que un atacante o grupo reducido de ellos pudiera lograr su cometido.

Por lo anterior, se puede concluir que el uso de RSA como algoritmo de encriptación de llave pública cuenta con la fortaleza suficiente para continuar por varios años.

CAPÍTULO 3

LAS FIRMAS DIGITALES

3.1 La Infraestructura de Llave Pública (PKI)

Uno de los mayores problemas relacionados con el acceso a redes de cómputo es el de la confianza. Un identificador de usuario y una contraseña son el vehículo para controlar el acceso pero no proveen de medios para verificar que el usuario es quien dice ser. Esta es una de las razones por las que los hackers logran entrar a las redes, ya que usan un identificador de usuario y contraseña legítimos para conectarse.

Las firmas y certificados digitales, así como las llaves públicas y privadas, son un método utilizado por personal de seguridad informática para proporcionar a la red un nivel adicional para verificar los identificadores de usuario. La base de confianza es mayor debido a que se asume que la persona que presenta el certificado digital y las llaves de encriptación tiene mayores probabilidades de ser el usuario legítimo propietario del identificador y la contraseña. Existe un nivel adicional de confianza en este caso debido a que existen mayores obstáculos para que un usuario sea capaz de obtener estas credenciales digitales. Si estas credenciales se procesan apropiadamente, uno o más usuarios pueden verificar la identidad del propietario. Si la identidad es puesta en tela de juicio, se puede regresar al inicio del proceso para verificar quién expidió los certificados y cuál fue el proceso de identificación utilizado previo a la expedición del certificado digital.

La Infraestructura de Llave Pública, mejor conocida como PKI, por sus siglas en inglés ("Public Key Infrastructure") hace uso de certificados digitales y llaves públicas encriptadas para autentificar a los usuarios, encriptar archivos y directorios propiedad del usuario y en el proceso asegura la integridad de los datos y la no repudiación de los usuarios, esto es, un usuario que envía un documento firmado digitalmente no puede negar el haberlo enviado él mismo posteriormente.

La Infraestructura de Llave Pública hace uso de la encriptación para crear certificados digitales, los cuales son generados, distribuidos y mantenidos por una infraestructura de servidores y software. La infraestructura no debe ser necesariamente complicada, pero sí debe de operar eficientemente. Por otro lado, existen diversos proveedores de software que ofrecen soluciones completas de PKI; sin embargo, no existe una solución de PKI que satisfaga todas las necesidades de seguridad y autentificación.

Una parte fundamental de la Infraestructura de Llave Pública son las firmas y certificados digitales. La infraestructura es construida para crear, organizar, almacenar, distribuir y mantener llaves públicas y privadas utilizadas dentro de los certificados digitales.

A continuación se mencionan los principales elementos que constituyen la Infraestructura de Llave Pública:

- **Autoridad Certificadora (AC):** La Autoridad Certificadora es una entidad independiente cuya operación está basada en un sistema de cómputo que requiere de la interacción humana para emitir certificados digitales. Inicialmente, un usuario realiza una petición para obtener un certificado digital desde su computadora, la cual interactúa con el sistema de la Autoridad-Certificadora. Para poder emitir un certificado, la AC solicita al usuario una serie de documentos de identidad para verificar al usuario, así como un pago por la emisión y vigencia del certificado. Una vez realizado lo anterior, el sistema de la AC emite y envía un certificado digital al usuario. Las Autoridades Certificadoras más conocidas son las compañías Verisign, CyberTrust y RSA.

- **Certificados digitales:** Los certificados digitales están compuestos de dos datos esenciales: la llave pública y la llave privada. Ambas son emitidas por la AC y la misma conserva una copia de ellas para propósitos de verificación de los certificados en caso de que un tercero requiera verificar la autenticidad de los mismos para un usuario determinado. La copia de ambas llaves también reside en el equipo del usuario.

- **Computadoras de escritorio y servidores:** Ambos equipos usualmente necesitan contar con una aplicación o cliente de PKI para que el usuario sea capaz de encriptar y desencriptar datos.

- **LDAP o directorios X.500:** LDAP (Lightweight Directory Access Protocol) y X.500 son protocolos de acceso a servidores de directorio, los cuales contienen bases de datos que recolectan y distribuyen llaves internamente. Frecuentemente, se cuenta con uno para uso interno y otro o varios para usuarios remotos o socios de negocios en localidades remotas.

- **Autoridad Registradora:** Versión interna de la Autoridad Certificadora. Si una organización de mayor tamaño tiene muchos oficinas y/o divisiones, con frecuencia dejaran que la Autoridad Registradora procese las peticiones para certificados digitales, verifique identidades y envíe esta información a la AC para poder expedir los certificados digitales.

3.2 Propósito de las firmas y certificados digitales

Cuando firmamos manualmente un documento perseguimos dos objetivos:

1) Hacemos una rúbrica que sólo nosotros sabemos hacer, para dejar constancia de que ese documento lo subscribimos realmente nosotros.

2) Ponemos la firma justo al final del documento para evitar que alguien pueda introducir texto adicional, es decir, para garantizar la integridad del texto.

La comunicación electrónica permite un sistema de firma mucho más segura. Las firmas digitales consisten en una función resumen del texto codificada con nuestra clave privada. Este sistema garantiza a la vez nuestra identidad y que el texto no ha sido modificado ni en una sola coma.

Una firma digital es una firma electrónica que puede ser utilizada para autentificar la identidad del emisor de un mensaje, el firmante del documento o el dueño de una tarjeta de crédito. También puede ser utilizada para asegurar que el contenido original de un mensaje o documento enviado no ha sido cambiado. Una firma digital es usualmente generada desde un certificado digital usando una tecnología de llave pública y privada. Es importante notar que una firma digital no sólo es una imagen grabada de una firma, un error relativamente común. Los beneficios adicionales de una firma digital son que es fácil de transportar, no puede ser fácilmente repudiada, no puede ser imitada por otra persona y puede ser automáticamente sellada a tiempo. Una firma digital puede ser utilizada con cualquier tipo de mensaje, ya sea codificado o no, de forma que el receptor puede estar seguro de la identidad del emisor del mensaje así como de que el mensaje llegó intacto. Un certificado digital también contiene la firma digital de la autoridad emisora del certificado de forma que cualquiera puede verificar que el cerificado es real.

Por otro lado, un certificado digital es un "pasaporte" electrónico el cual establece sus credenciales cuando se está haciendo negocios u otras transacciones en el Web. Un certificado es emitido por una Autoridad Certificadora (AC) y contiene su nombre, un número serial, fecha de expiración, una copia de la llave pública del dueño del certificado (utilizada para encriptar y desencriptar tanto mensajes como firmas digitales), y la firma digital de la autoridad emisora del certificado (Verisign, por ejemplo) de forma que el receptor pueda verificar que el certificado es real. De igual forma que un pasaporte sirve para

dar identidad a quien la porta en ciertos casos, el certificado digital da identidad a una clave pública y se comporta como una persona en el espacio cibernético. Con el certificado digital se permite garantizar que el autor del mensaje es quien realmente dice ser. El siguiente problema es imposibilitar que nadie pueda manipular el contenido del mensaje, para lo cual se hace uso de la firma digital. Cuando un usuario A genera un mensaje para un usuario B, lo encripta junto con su certificado. Opcionalmente puede protegerlo con la clave pública del usuario B. Esto es lo que se llama "firmar digitalmente" o construir lo que se denomina un "sobre electrónico".

Es probable que en un futuro cercano las organizaciones e individuos tengan varios certificados digitales (ID) para un rango de actividades las cuales requieran validar sus identidades. Por ejemplo, una persona dentro de un departamento de gobierno puede utilizar una de éstas para acceder información confidencial de un Intranet mientras tenga un ID separado para comprar en el web. Un gobierno, o un departamento, pueden ser una autoridad emisora de certificados en la jerarquía.

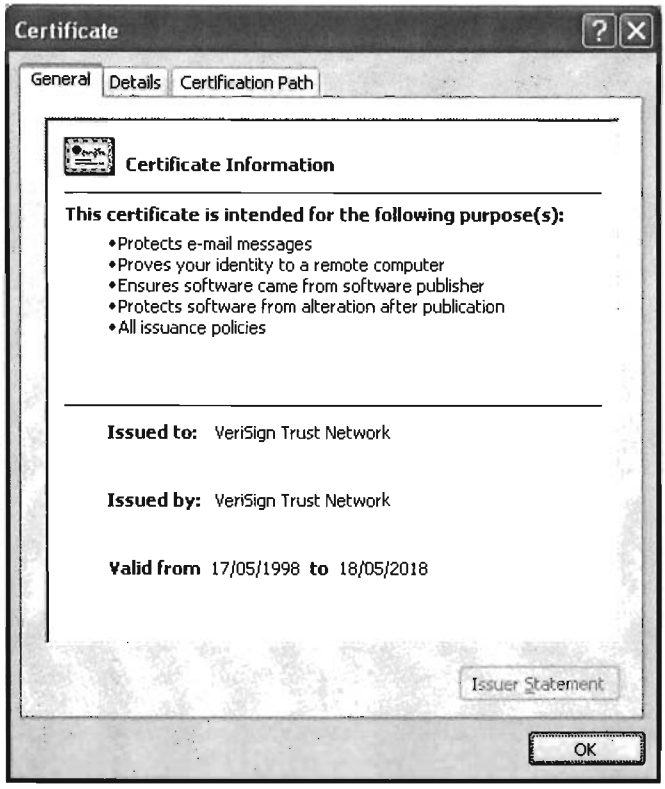
El nacimiento de las firmas y certificados digitales fue a raíz de resolver el problema de administrar las claves públicas y que la identidad del dueño no pueda ser falsificada. La idea es que una tercera entidad intervenga en la administración de las claves públicas y asegure que las claves públicas tengan asociado un usuario claramente identificado.

Las tres partes más importantes de un certificado digital son:

1) Una clave pública
2) La identidad del implicado: nombre y datos generales,
3) La firma privada de una tercera entidad llamada autoridad certificadora que todos reconocen como tal y que valida la asociación de la clave pública en cuestión con la persona que dice ser.

En la actualidad casi todas las aplicaciones de comercio electrónico y transacciones seguras requieren un certificado digital. Se ha propagado tanto su uso que se tiene ya un formato estándar de certificado digital, conocido como ANSI X509 v. 3. Algunos de los datos más importantes de este formato son los siguientes:

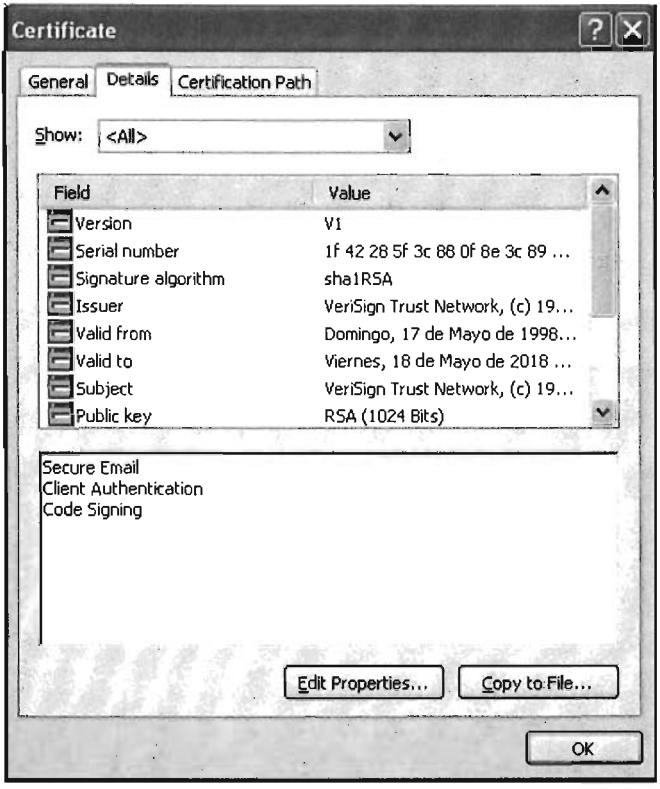
En una aplicación o navegador que es compatible con el uso de certificados digitales (en este caso, Internet Explorer) un certificado digital se puede ver de manera similar a las siguientes pantallas:



La pantalla anterior sólo muestra la información general del certificado. Sus elementos, en detalle, son los siguientes:

Versión: 1,2 o 3
Número de Serie
Identificador del Algoritmo usado en la firma: SHA-1, RSA, DSA, etc.
Emisor del Certificado: (en este caso, VeriSign)
Identificador del Algoritmo usado en la firma: RSA, DSA o CE
Periodo de Validez
Sujeto (en este caso, es el propio VeriSign Trust Network ya que es una Autoridad Certificadora raíz)
Información de la llave pública del sujeto: algoritmo empleado y longitud
Firma o "huella" de la Autoridad Certificadora (conocida como Thumbprint o Fingerprint) generada por algoritmos tales como SHA-1 o MD5
Nombre que identifica al certificado
Usos posibles de la llave pública (Email seguro, Autentificación de clientes, firmas digitales, etc.)

Dichos elementos pueden visualizarse también en el navegador dentro de la pestaña de detalles del certificado:



Un certificado digital entonces se reduce a un archivo de uno o dos kilobytes de tamaño, que autentica a un usuario de la red.

3.3 Técnicas de protección de la integridad de los mensajes

La encriptación por sí sola no protege la integridad de los mensajes, aun si las llaves no han sido descifradas. Por otro lado, las firmas digitales por sí solas no proveen de autentificación confiable, pues no impiden que un intruso alterase un mensaje y después incluyera la firma original junto con el mismo. Es necesario utilizar técnicas especialmente desarrolladas para proteger la integridad de los mensajes junto con la encriptación y las firmas digitales. La técnica más común es las funciones hash, las cuales se basan en el cálculo de sumas de comprobación criptográficas.

3.3.1 Suma de comprobación criptográfica

Una técnica para detectar cambios en un mensaje protegido es la **suma de comprobación criptográfica**. A diferencia de la encriptación, esta técnica no oculta el contenido de un mensaje. En lugar de eso, realiza cálculos para proteger el mensaje contra cambios. Una suma de comprobación criptográfica efectiva no permitirá a un atacante producir o modificar un mensaje protegido.

Una **suma de comprobación** es un procedimiento computacional cuyo resultado depende del contenido de algún bloque arbitrario de datos. Idealmente, cualquier cambio en el contenido de los datos producirá una suma de comprobación diferente. Si los datos y la suma son enviados a través de la red en el mismo mensaje, el destinatario puede verificar que los datos llegaron intactos repitiendo el cálculo de la suma de comprobación y comparando el resultado con la suma contenida en el mensaje. Una forma sencilla de calcular una suma de comprobación de un mensaje consiste en sumar los valores ASCII correspondientes de cada carácter del mensaje. Por ejemplo, el mensaje "$109" produce una suma de comprobación igual a 190 (sumando los valores ASCII equivalentes de cada carácter, 36 + 49 + 48 + 57 = 190).

El problema de la seguridad en redes consiste en que pueden existir individuos motivados a modificar mensajes sistemáticamente para obtener algún beneficio. Por ejemplo, un intruso podría interceptar un mensaje, modificarlo, recalcular la suma de comprobación y reemplazarla en el mensaje con la nueva. El destinatario no podría detectar la diferencia, ya que la nueva suma validaría el nuevo contenido del mensaje modificado.

La suma de comprobación criptográfica vence este tipo de falsificaciones usando una llave criptográfica al calcular el valor de la suma de comprobación. Por ejemplo, se podría usar el algoritmo DES con una llave secreta para encriptar una suma de comprobación de 64 bits de un mensaje. El equipo receptor del mensaje tomaría el resultado de la suma de comprobación de los datos recibidos, lo encriptaría usando la misma llave privada y compararía el resultado con la suma de comprobación encriptada en el mensaje. Este proceso se ilustra en la figura 3.1.

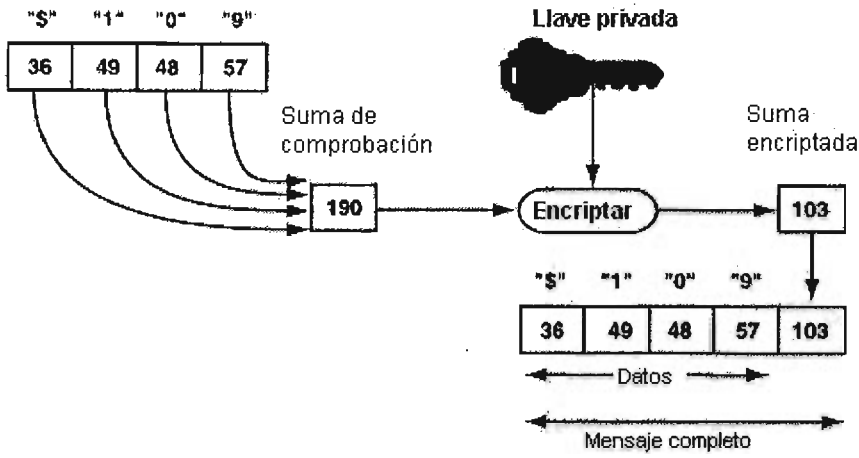


Fig. 3.1: Suma de comprobación criptográfica

El emisor calcula la suma de comprobación de los datos y la encripta junto con los mismos usando una llave privada. El receptor del mensaje procede a desencriptar el mensaje y puede verificar la autenticidad del mismo recalculando la suma de comprobación criptográfica usando la misma llave. La suma de comprobación criptográfica debe coincidir con el valor de la suma recibido junto con el mensaje. Los intrusos no pueden calcular sumas de comprobación criptográficas válidas sin la llave.

Para prevenir a los atacantes de falsificar o modificar mensajes, la suma de comprobación criptográfica debe resistir dos ataques en general. En primer lugar, el atacante no debe de poder construir una suma de comprobación criptográfica válida para un mensaje alterado. El cálculo debe estar basado en información secreta de algún tipo que el atacante no pueda tener y que debe ser usada para producir la suma de comprobación de una manera que el atacante no pueda duplicar. En segundo lugar, el atacante no debe poder tomar un mensaje existente con una suma de comprobación criptográfica legal y ser capaz de construir un mensaje diferente que tuviera la misma suma de comprobación criptográfica. En este caso, un atacante no necesita conocer la llave privada para sustituir el contenido legítimo del mensaje. Lo único que el atacante necesita saber es la forma en que se calcula la suma de comprobación y con esto puede calcular su valor. Si encuentra un mensaje diferente y más conveniente a sus intereses que produzca la misma suma, lo podría sustituir por el mensaje original. Por ejemplo, un mensaje cuyo contenido es "$109" produce la suma de comprobación 190. Este es el mismo resultado de la suma de comprobación para el mensaje "$910" (sumando los valores ASCII de cada carácter, 36 + 57 + 49 + 48 = 190). Debido a que los valores de la suma de comprobación coinciden, los valores de la suma de comprobación criptográfica también deben de coincidir. El receptor del mensaje no sería capaz de

determinar si hubo cambios en el contenido del mensaje. Debido a esto, existen técnicas tales como las funciones hash.

3.3.2 Funciones hash o de resumen de una sola dirección

Una técnica ampliamente utilizada para preservar la autenticidad e integridad de los mensajes es la función **hash**, o función de resumen. Una función hash es también ampliamente usada para la firma digital, ya que los documentos a firmar son en general demasiado grandes. Es similar a la encriptación en el sentido de que codifica un mensaje mediante un algoritmo, pero a diferencia de ésta, no existe un procedimiento inverso a la función hash con el que se pudiera regresar al mensaje original partiendo del resultado de la función, por lo que generalmente se le denomina función hash *de una sola dirección*. La función hash les asocia una cadena de longitud 160 bits que los hace más manejables para el propósito de firma digital. De alguna forma, lo que se hace es tomar el mensaje, partirlo en pedazos de longitud constante y combinar de una manera determinada pedazo por pedazo hasta obtener un solo mensaje de longitud fija. Las funciones hash son similares a las sumas de comprobación, pero están diseñadas explícitamente de tal forma que un atacante no pueda construir un mensaje falsificado que produzca una suma igual a la legítima. Al igual que las sumas de comprobación, las funciones hash toman un mensaje de longitud arbitraria y calculan un valor de determinado tamaño, llamado el valor *hash* o valor resumen. El propósito de este valor es obtener un "resumen" (hash) representativo de un mensaje particular, que solamente pueda generarse mediante la llave secreta del firmante y mediante el mensaje, y que no sea de longitud excesiva. Esto es con el objeto de evitar que los sistemas puedan generar firmas iguales para mensajes distintos. A diferencia de las sumas de comprobación, las funciones hash tratan de generar una "huella digital" de los datos. Las funciones están diseñadas de tal forma que sea lo más difícil posible construir otra secuencia de datos que genere el mismo resultado.

Por lo anterior, las funciones hash se pueden representar de la siguiente manera. Sea M el mensaje a cifrar, H la función hash y h el valor resumen del mensaje, entonces:

H(M) = h, donde la función H debe cumplir las siguientes características:

- Para un mensaje M dado, debe ser fácil obtener h. Para un valor h dado, debe ser muy difícil calcular M tal que H(M) = h.
- Dado un mensaje M1 dado, debe ser muy difícil encontrar un mensaje M2 de tal manera que H(M1) = H(M2)

Las funciones hash son más sensibles a cambios menores en el texto de entrada, por lo que proveen de mayor protección a longitudes mayores de datos. Por ejemplo, los vendedores de software frecuentemente publican parches en

Internet para corregir alguna falla en su producto de tal forma que los clientes puedan bajarlos directamente. Los vendedores publican valores hash, usualmente calculados por medio del algoritmo MD5, de tal forma que los recipientes puedan verificar la integridad del software después de bajarlo. El cambiar un solo bit cambiaría el resultado del hash. Si el valor hash del archivo bajado coincide con el publicado, entonces el receptor puede estar seguro de que el archivo no ha sido modificado desde que el vendedor lo creó.

El valor hash publicado protege de cambios accidentales e intencionales a los programas. Los cambios accidentales ocurren ocasionalmente, particularmente cuando datos importantes son copiados de un sitio de Internet a otro alternativo. Por otro lado, existen parches y software alterado en diversos sitios de Internet. Las sumas de comprobación publicadas permiten proteger a los consumidores dándoles una verificación de que el software que recibieron es auténtico y está intacto. Sin embargo, las comunicaciones entre dos sitos deben de tomar medidas adicionales. Si solamente se incluye el valor hash junto con el mensaje, un intruso puede modificar mensajes o producir nuevos y tratar de hacerlos pasar como auténticos incluyendo el valor hash calculado de los contenidos modificados. Sin embargo, la falsificación es bloqueada si la función hash incorpora una llave secreta.

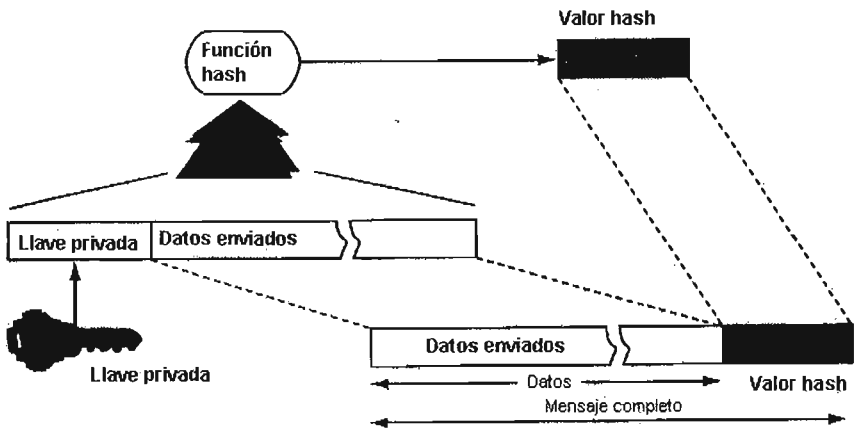


Fig. 3.2: Funciones hash

La figura 3.2 ilustra la generación del valor hash encriptado con una llave privada. Los datos a enviar son combinados con la llave y el valor hash es calculado de esta combinación. El resultado del hash es entonces enviado junto con el mensaje. A diferencia de la suma de comprobación, el valor hash que incorpora la llave privada bloquea efectivamente los ataques de falsificación. Esto es debido a que los intrusos no pueden generar el valor correcto de la suma de comprobación para el mensaje a menos que tengan la llave que se

esté utilizando. El valor correcto de la suma ahora depende de dos elementos, el mensaje y la llave que se utiliza. Por otro lado, los intrusos no pueden construir un mensaje modificado similar que produjera el mismo valor hash. Por lo tanto, las probabilidades de construir un mensaje que resultara benéfico para un intruso y que resultara en el mismo valor hash calculado a partir de la combinación del mensaje original y una llave privada, son muy remotas.

Aun cuando las funciones hash proveen de cierto grado de autentificación, no son lo mismo que las firmas digitales. Las funciones hash generalmente usan una llave secreta que es compartida entre el emisor y el receptor. Esto significa que ambos pueden generar mensajes auténticos. Las firmas digitales están generalmente basadas en tecnología de llave pública, la cual provee de llaves privadas únicas para cada individuo. Las funciones hash asocian los datos protegidos con un grupo de individuos que comparten la llave, mientras que las firmas digitales pueden asociar los datos protegidos con un individuo o dispositivo en particular.

Algunos algoritmos de funciones hash que incorporan llaves privadas se mencionan a continuación:

- **MD2** (Message Digest 2). Se diseñó para ordenadores con procesador de 8 bits, y hoy apenas se utiliza. Se conocen ataques a versiones parciales de MD2.

- **MD4** (Message Digest 4). Fue desarrollado por Ron Rivest, de RSA Data Security. Su diseño es la base de otros algoritmos hash, aunque se le considera inseguro. Un ataque desarrollado por Hans Dobbertin permite generar colisiones (mensajes aleatorios con los mismos valores hash) en cuestión de minutos para cualquier PC. Por ese motivo, está en desuso.

- **MD5** (Message Digest 5). También fue creado por Ron Rivest. Hasta hace poco se consideraba un algoritmo seguro, hasta tal punto que es el utilizado en el programa de cifrado PGP para firmar mensajes de correo electrónico con claves de tipo RSA; también se utiliza como autentificador de mensajes en el protocolo SSL. Sin embargo, en 1996 el mismo Hans Dobbertin que rompió MD4 consiguió demostrar que la función de compresión del algoritmo MD5 es insegura, consiguiendo colisiones en ese caso. Sus ataques son parciales y no pueden extenderse a la totalidad del algoritmo MD5. Asimismo cabe destacar que estos ataques comprometían la propiedad de no-colisión. Esto significa que se puede obtener un valor hash idéntico para dos mensajes obtenidos al azar, pero queda por demostrar que se pueda encontrar un mensaje con igual valor hash a *otro mensaje en concreto*. Con todo, resulta inquietante que un algoritmo hash considerado seguro tenga tales vulnerabilidades potenciales. A menos que se le refuerce, su uso irá decreciendo en el futuro.

- **RIPEMD-160** (RIPE MessageDigest). Fue desarrollado por un grupo europeo encabezado por Hans Dobbertin, Antoon Bosselaers y Bart Preneel, dentro del proyecto RIPE (RACE Integrity Primitives Evaluation). Su versión primera (RIPEMD) causaba colisiones del mismo tipo de las de MD4 y MD5. Posteriormente se reforzó y se llamó RIPEMD-160. Al contrario que la mayoría de los algoritmos (que dan valores hash de 128 bits), RIPEMD-160 crea un hash de 160 bits. Existe una versión de 128 bits, y se planean versiones con hash de 256 y 320 bits. Es uno de los algoritmos hash más rápidos, no está patentado y su código fuente es de libre acceso. Por el momento se le considera seguro.

- **SHA-1** (Secure Hash Algorithm). Fue desarrollado como parte del Estándar de Hash Seguro (Secure Hash Standard, SHS) y el Estándar de Cifrado Digital (Digital Signature Standard, DSS) por la Agencia de Seguridad Nacional norteamericana (NSA). A pesar de su origen, parece un hash seguro y sin fisuras, al menos por ahora. La primera versión, conocida como SHA, fue mejorada como protección ante un tipo de ataque que nunca fue revelado. El documento FIPS (Federal Information Processing Standard) que oficialmente lo describe afirma que los principios subyacentes al SHA-1 son similares a los del MD4 de Rivest. Su implementación puede estar cubierta por patentes en EEUU y el extranjero. A falta de ataques ulteriores, se le puede considerar seguro. Es el algoritmo de firmado utilizado por el programa PGP en sus nuevas claves DH/DSS (que significa: cifrado mediante clave Diffie-Hellman y firmado mediante función hash de acuerdo al Digital Signature Standard).

Los valores hash efectivos también pueden ser construidos utilizando un buen cifrado por bloques tal como DES en un modo de cifrado apropiado tal como CBC o CFB. La tabla 3.1 muestra los algoritmos más utilizados de los mencionados anteriormente, junto con el tamaño de llave que emplean y el tamaño del valor hash que producen.

Algoritmo Hash	Tamaño de llave (bits)	Tamaño de hash (bits)
SHA-1	128 (variable)	160
MD5	128 (variable)	128
ANSI X9.9 con DES modo CBC	56	64

Tabla 3.1

3.3.3 Requerimientos técnicos para las funciones hash

A continuación se enlistan los requerimientos técnicos para obtener funciones hash con un nivel adecuado de seguridad:

- **Las funciones deben ser de un solo sentido.**
  Un atacante no debería de poder determinar los datos originales siguiendo un proceso inverso al cálculo de la suma criptográfica, partiendo del resultado de la misma.

- **El cambio de un bit debe alterar el resultado de la suma criptográfica.**
  Un atacante no debería de poder cambiar un dato original sin que la suma criptográfica resultante se vea afectada.

- **Las transposiciones de bits deben de alterar el valor de la suma criptográfica.**
  Al igual que en el punto anterior, el más mínimo cambio en los datos originales debe de alterar el resultado de la suma criptográfica.

- **Los cambios en la longitud del mensaje deben de cambiar el valor de la suma criptográfica.**
  Esto es particularmente importante para valores hash calculados a partir de datos en los cuales la llave privada viene incluida. El uso de llaves no provee de protección si se obtiene el mismo valor hash si éstas son suprimidas para realizar el cálculo de la suma.

- **Las funciones deben de resistir colisiones.**
  Esto se encuentra implícito hasta cierto punto en los requerimientos anteriores. El requerimiento esencial es, sin embargo, que un atacante no sea capaz de construir un mensaje sistemáticamente con el mismo valor hash de un mensaje ya existente, excepto por medio del ataque de fuerza bruta.

3.3.4 Generación de números pseudo aleatorios

Un sistema criptográfico práctico y seguro requiere tener llaves que sean difíciles de adivinar. No debe existir ninguna forma por la cual un usuario ajeno pudiera predecir las llaves que se están utilizando, ni que pudiera duplicar aproximadamente el proceso de generación de llaves, por lo que se requiere tener generadores que produzcan llaves que no pudieran ser adivinadas incluso si los atacantes conocieran la forma en la que el generador trabaja. Para lograr esto, se necesita generar números que sean prácticamente imposibles de predecir. Las computadoras por sí solas son fuentes pobres de números impredecibles, ya que éstas son esencialmente máquinas deterministicas que están diseñadas para ser predecibles. Por esto, diversos

matemáticos han desarrollado diversos procedimientos, llamados Generadores de Números Pseudo Aleatorios, los cuales generan secuencias de números difíciles de predecir. Sin embargo, éstos números son resultados consistentes provenientes de procedimientos matemáticos consistentes, por lo que para que se pueda lograr una aleatoriedad más auténtica, se debe de tomar un número inicial como semilla para iniciar el proceso. En aplicaciones criptográficas, la semilla debe ser difícil de determinar también, ya que no basta con que ésta cambie un par de veces. Si un atacante lograra dar con un valor aproximado al de la semilla, entonces podría probablemente adivinar qué llaves fueron generadas usando el mismo procedimiento.
Un ejemplo de esto sucedió con el navegador de Internet Netscape Navigator, cuya seguridad dependía en versiones previas de una llave de sesión aleatoria utilizada para transmitir datos a través del protocolo SSL. En 1995, un grupo de estudiantes logró descifrar el proceso de generación de números aleatorios, el cual era aceptable, pero el proceso de ingreso de la semilla no lo era. El navegador tomaba muestras de números provenientes de cantidades variables en el sistema, incluyendo del reloj del sistema y con estos elementos construía la semilla aleatoria. Los estudiantes descubrieron mediante la observación y estudio exhaustivo del proceso, que ciertos valores de semilla aproximados a los que producía el navegador, podían ser generados. Esto les permitió escribir un programa que fuera capaz de adivinar rápidamente la llave que Netscape usaría en una conexión encriptada por medio de SSL. El programa calculaba los valores posibles de semilla, con lo cual se reducía considerablemente el número de llaves que se debían probar. Debido a esto, el proceso de generación de semillas tuvo que ser mejorado en versiones posteriores del Netscape Navigator.

Debido a lo expuesto anteriormente, existe la necesidad de contar con una técnica aleatoria para generar valores semilla aleatorios. En la práctica, existen 3 métodos computacionales a partir de los cuales se pueden generar datos auténticamente aleatorios:

1. Hardware de monitoreo que genere datos aleatorios.
2. Recolección de datos originados de la interacción con el usuario.
3. Recolección de datos difícilmente predecibles provenientes de la misma computadora.

La generación de valores aleatorios por medio de hardware de monitoreo es el mejor pero más costoso método de los tres. Este tipo de generador es regularmente un circuito electrónico que es sensible a algún evento físico aleatorio, tal como el ruido provocado por algún diodo o rayos cósmicos, etc. y convierte estos eventos en una secuencia impredecible de bits. Este tipo de circuitos generalmente producen los suficientes bits aleatorios como para que un generador de números pseudo aleatorios sea innecesario. Sin embargo, debido a su grado de sofisticación y rareza en el mercado los convierte en algo más costoso.

La interacción con el usuario es una buena fuente de datos aleatorios, aunque podría ser inconveniente en algunos casos. Este método consiste en recolectar datos provenientes de algún evento realizado por el usuario. Por ejemplo, el paquete de seguridad para correo electrónico PGP registra pulsaciones de teclas y mide el tiempo entre pulsaciones para producir un valor de semilla aleatorio. De esta forma, PGP recolecta varios cientos de bits aleatorios en menos de un minuto de tecleo. Otro método de interacción un poco menos eficiente podría ser el de requerir al usuario ingresar un valor aleatorio directamente.

Un valor aleatorio también puede ser construido a partir de diversas cantidades variables provenientes del propio sistema. El número aleatorio se construye por medio de la obtención de una muestra de cantidades, extrayendo los bits de orden inferior y concatenando los bits los bits en un número aproximadamente largo. Las cantidades tomadas como muestra podrían ser el reloj del sistema, el número de archivos en el sistema de archivos, el número de bloques de memoria libres o usados, la cantidad de memoria en uso, etc. Sólo los bits de orden inferior se utilizan, ya que la cantidad se vuelve predecible conforme se utilizan bits de mayor orden. La aleatoriedad también depende de seleccionar valores que sean independientes unos de otros. Por ejemplo, las versiones recientes del Netscape Navigator utilizan cientos de valores variables internos del sistema para construir una semilla aleatoria utilizada en el protocolo SSL.

En suma, los generadores de números pseudo aleatorios en un sistema criptográfico deben ser fuertes, entendiéndose por esto, impredecibles. Esto requiere de dos propiedades importantes:

1. No deben existir relaciones matemáticas obvias entre números en la secuencia, tal como múltiplos comunes, ordenamientos o patrones de valores.
2. Una llave cualquiera podría ser generada como parte de una variedad de secuencias diferentes de llaves.

El segundo requerimiento ilustra una propiedad particularmente importante. El problema con los generadores de números pseudo aleatorios consiste en que producen secuencias de números y al ser procedimientos matemáticos, tienen la tendencia a producir las mismas secuencias una y otra vez. Debido a que un intruso pudiera obtener una de las llaves generadas, existe el riesgo de que pudiera ser capaz de reconstruir otras llaves recientemente generadas a partir del mismo procedimiento con el que se generó la llave interceptada.

Los algoritmos criptográficos generalmente juegan un papel importante en la construcción de secuencias aleatorias de números. Durante varios años, los bancos han generado secuencias de valores aleatorios utilizando el algoritmo

DES. Esta técnica produce llaves que son difíciles de predecir y repetir. Esto es debido a que son generadas utilizando una llave secreta aleatoria como semilla. Es poco probable que las llaves se repitieran ya que las llaves secretas que se utilizan inicialmente son generadas al encriptar una cantidad cambiante tal como lo es el reloj del sistema. En ese caso, si un intruso lograr recuperar alguna llave generada, ésta no le puede proporcionar información sobre la forma en que se han generado las demás. Una alternativa de utilización del algoritmo DES para generar valores semilla aleatorios es el modo "autollave", en el cual se utiliza el valor pseudo aleatorio generado por la primera iteración como semilla para generar el siguiente valor, generando así llaves que sean prácticamente imposibles de adivinar aun si se conoce la forma de trabajar del algoritmo empleado.

3.4 Funcionamiento de las firmas digitales

Las firmas digitales son el mecanismo más novedoso proporcionado por la tecnología criptográfica moderna. Son un mecanismo que no sólo autentifica la fuente de donde provienen los mensajes, sino que protegen los datos de cambios que de otra forma no podrían ser detectados. Las firmas digitales asocian los datos con el dueño de una llave privada en particular. Si se puede verificar la firma con la llave pública correspondiente, podemos tener la certeza de que los datos fueron firmados con la llave privada del remitente, con lo que se comprueba que es la persona quien dice ser. Por lo tanto, las firmas digitales son una técnica ampliamente aceptada dentro de comercio electrónico, ya que proveen de credenciales electrónicas que son difíciles de falsificar a los participantes de una transacción.

La Criptografía de llave pública se utiliza para producir una firma digital, generalmente por medio del algoritmo RSA. Por ejemplo, un emisor puede firmar digitalmente un mensaje encriptándolo con su llave privada. Cualquier persona que obtenga su llave pública podrá desencriptarlo, pero el mensaje solo se desencriptará correctamente si la llave privada correspondiente es la correcta, por lo que esto significa que la llave privada con la que fue encriptado el mensaje debe provenir necesariamente de la persona que la posee, que es quien firma el documento y de ninguna otra persona.

Las firmas digitales usan una llave privada para producir una suma de comprobación. Las sumas de comprobación criptográficas basadas en técnicas convencionales de llave privada sólo pueden ser verificadas por personas a las que les es confiada la llave privada, además de que esta técnica no permite determinar cual de las personas que poseen la llave privada fue la que produjo la suma de comprobación. En cambio, las firmas digitales están ligadas a una llave privada en particular con una llave pública correspondiente. Así, cualquier persona que obtenga la llave pública puede validar la suma de comprobación, al

estar ligada a una y solo una llave privada. Entonces, se puede asumir con seguridad que solamente el dueño de la llave privada pudo haber producido la firma digital correspondiente.

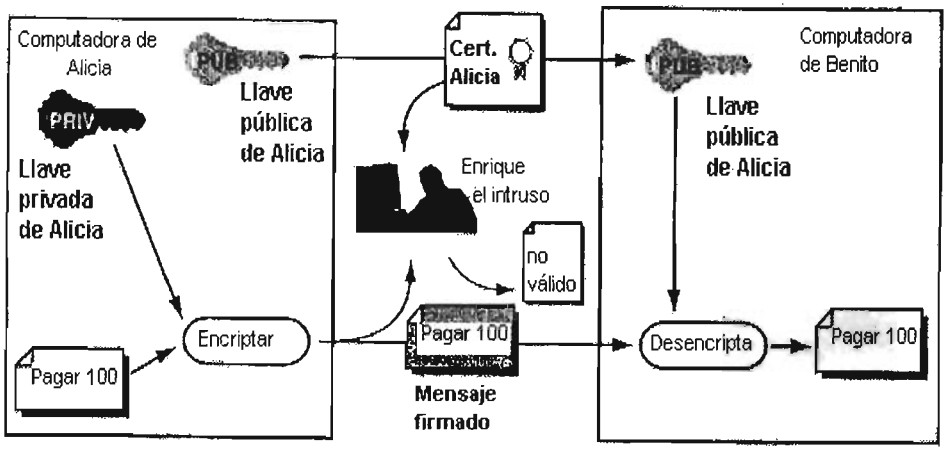


Fig. 3.3: Funcionamiento de las firmas digitales

La figura 3.3 ilustra el funcionamiento de las firmas digitales. Para fines del ejemplo, se le llamará a la persona emisora del mensaje "Alicia", a la persona receptora "Benito" y al intruso "Enrique". Entonces, Alicia escribe un mensaje, el cual dice "Pagar $100" para Benito y lo "firma", esto es, en el caso más simple, encriptar el mensaje con su llave privada. Después de esto, transmite su mensaje hacia Benito. El obtiene la llave pública de Alicia y puede verificar que el mensaje efectivamente proviene de Alicia desencriptándolo con su llave pública. Al hacer esto, él debe de obtener como resultado un mensaje de texto plano legible, si el mensaje fue realmente encriptado por Alicia. Enrique, el intruso, no puede construir directamente un mensaje falsificado y enviárselo a Benito. Si intenta enviar un mensaje de texto plano que diga "Pagar $100", éste será reducido a algo sin sentido cuando Benito aplique la llave pública de Alicia para desencriptarlo, ya que no fue encriptado con la llave privada de Alicia. Por otro lado, si Enrique intenta enviar el mensaje encriptándolo con la llave pública de Alicia, también producirá algo sin sentido cuando Benito lo desencripte, ya que él estará usando la misma llave pública para desencriptarlo, y como se mencionó en el capítulo anterior, una misma llave no puede ser utilizada tanto para encriptar como para desencriptar. Por lo tanto, Enrique no puede producir una buena falsificación de un mensaje sin la llave privada de Alicia. Sin embargo, la seguridad no está garantizada totalmente. Existen formas en las que un atacante hábil puede vencer el mecanismo. El riesgo latente consiste en que Enrique pudiera construir un mensaje de tal forma que al aplicarle la llave pública de Alicia, produzca un mensaje razonablemente legible, con lo cual se podría pensar que Alicia firmó el mensaje, pero que contiene algo que ella nunca

intentó decir. Para evitar esto, se calcula el valor hash del mensaje a firmar y después se encripta, como se muestra en la figura 3.4. Esta técnica también remueve varias vulnerabilidades matemáticas asociadas con los cálculos de firmas digitales, las cuales se mencionarán posteriormente.

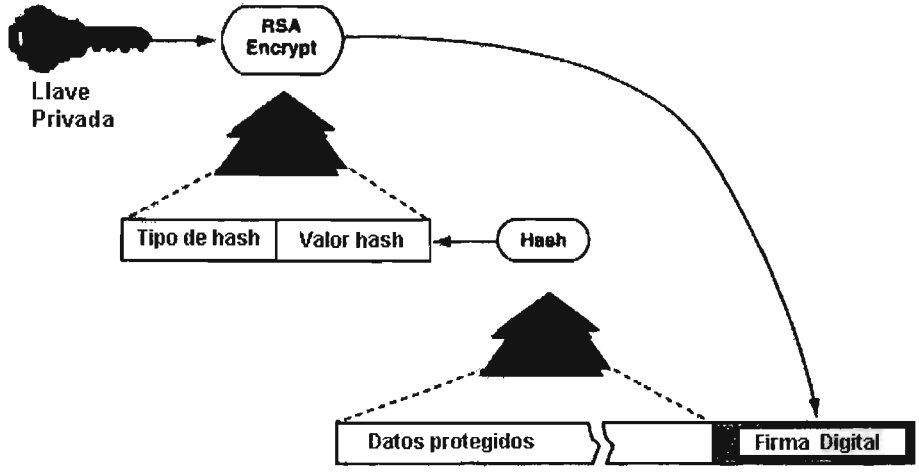


Fig. 3.4: Protección de datos por medio de la firma digital

Como lo muestra la figura anterior, para proteger los datos por medio de una firma digital, se calcula la función hash del mensaje y después se encripta el hash con la llave privada del autor. Se incluye también el tipo de hash en la firma digital por razones de seguridad y compatibilidad. La figura 3.5 muestra cómo se valida la firma digital una vez recibido el mensaje.

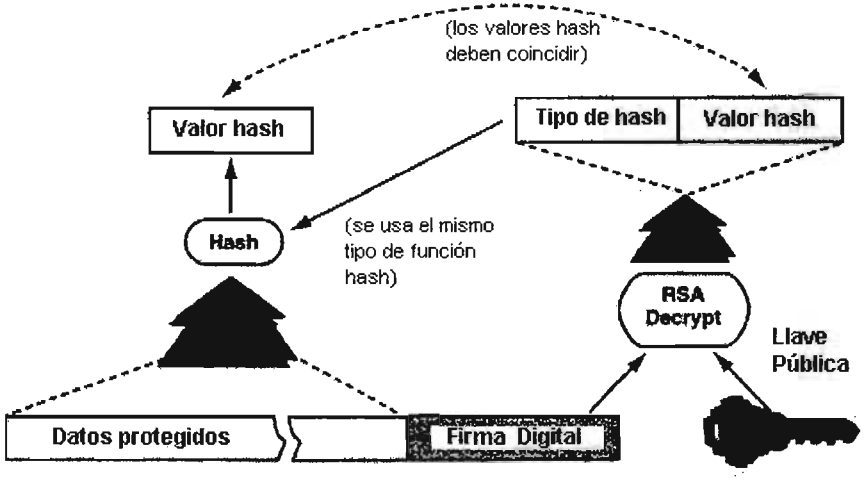


Fig. 3.5: Validación de los datos firmados digitalmente

El destinatario valida los datos al verificar el valor hash encriptado. Para esto, desencripta la firma digital con la llave pública del autor, obteniendo el valor y el tipo del hash. Después, la función hash es aplicada a los datos recibidos. El resultado es entonces comparado con el que fue protegido por la firma digital.

3.5 El Estándar de Firmas Digitales (DSS)

El algoritmo RSA es el más común en los sistemas comerciales y es utilizado también para la generación de firmas digitales. Sin embargo, existe también el Algoritmo de Firmas Digitales (también conocido como DSA, por su nombre en inglés, *Digital Signature Algorithm*), el cual sirve para firmar datos a los cuales se les ha calculado su valor hash mediante el algoritmo SHA. Ambos algoritmos fueron desarrollados por la NSA. Por otro lado, el NIST[1] estableció el Estándar de Firmas Digitales (*Digital Signature Standard)* como el Estándar de Procesamiento de Información Federal (FIPS) a partir de los algoritmos DSA y SHA.

El algoritmo DSA es una variante del algoritmo de llave pública ElGamal, el cual es a su vez una variante del algoritmo Diffie-Hellman, mencionado anteriormente. El proceso del DSS se muestra en la figura 3.6. Primero, se calcula el valor hash del mensaje utilizando el algoritmo SHA, el cual genera un resultado de 160 bits de longitud. Después, el firmante debe generar un número aleatorio que debe mantenerse secreto. El valor hash firmado es generado al aplicar el algoritmo DSA al hash junto con el número aleatorio y la llave privada del firmante. El número aleatorio es borrado y el valor calculado de la firma es incluido en el mensaje.

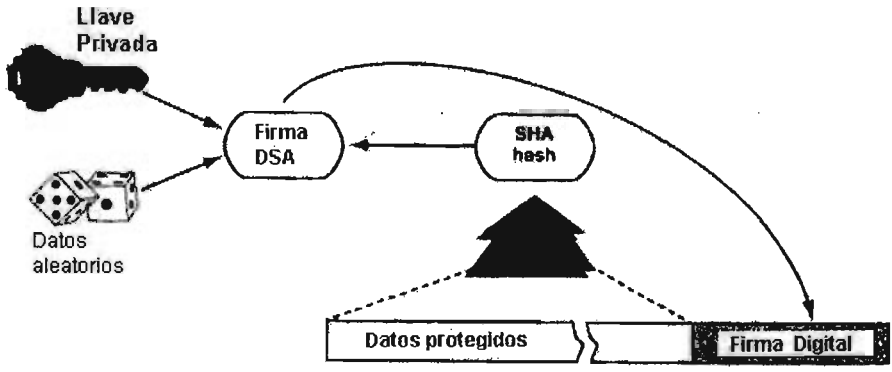


Fig. 3.6: Firma de datos por medio de DSS

[1] National Institute of Standards Technology, o Instituto Nacional de Estándares de Tecnología, de Estados Unidos.

Una debilidad práctica en el DSS es en el valor aleatorio secreto requerido para cada firma. La llave privada con la que se firma el mensaje es vulnerable a ataques si el valor aleatorio es usado incorrectamente. Si un intruso puede adivinar el valor aleatorio usado para calcular la firma digital, pueden utilizarlo para extraer la llave privada implicada en el cálculo de la firma digital. Otro problema con el valor aleatorio secreto es que cada firma digital debe ser calculada con un número aleatorio diferente. Si dos o más mensajes son firmados con el mismo número aleatorio, los atacantes pueden usar esa información para extraer la llave privada de nuevo. Una vez que la llave privada se logra extraer, los intrusos pueden falsificar mensajes con esa firma digital. Los riesgos asociados con el número aleatorio son causados por propiedades matemáticas de los cálculos del DSS. Algunos analistas sospechan otras debilidades, dependiendo de ciertas implementaciones. Por ejemplo, la llave pública del DSS contiene números que pudieran ser constantes como una conveniencia de implementación (los números "p" y "q", por ejemplo) por lo que existe la preocupación de que esto pudiera simplificar el criptoanálisis de firmas digitales.

Otra área de preocupación es la generación de llaves. Al igual que el algoritmo RSA, el DSS debe generar números primos largos para obtener llaves privadas fuertes. Algunos números primos pueden ser más fáciles de factorizar que otros. Sin embargo, la definición publicada del DSS describe un procedimiento para generar números primos suficientemente fuertes y en donde se rechazan los posibles números primos débiles si éstos resultan ser generados como posibles llaves.

Las observaciones anteriores producen la siguiente lista de requerimientos que se deben de cumplir al implementar el DSS. Estos se enumeran de acuerdo a su prioridad, de mayor a menor:

- **Usar un generador de números aleatorios adecuado.**
  Los números primos deben ser esencialmente imposibles de predecir. Para esto se deben seguir las recomendaciones mencionadas anteriormente en la sección 3.2.4.

- **Utilizar valores de módulo variables.**
  Los valores P y Q de llave pública no deben ser constantes durante la implementación. Se deben hacer una parte variable dentro del material de llave pública.

- **Utilizar una implementación confiable.**
  Para esto, asegurarse de que el software del DSS a implementar tenga un nivel de confiabilidad comprobable.

Aún cuando siempre es importante tratar con vendedores confiables y respetables, en el caso del algoritmo DSS convierte esto en algo particularmente

importante. Una falla dentro de su funcionamiento puede permitir a un desarrollador de DSS incluir pequeñas cantidades de información dentro de cada firma digital. A este mecanismo se le llama *canal subliminal* debido a que puede traer consigo información oculta mientras su funcionamiento continúa normalmente. Un desarrollador sin escrúpulos podría usar este canal para transmitir información acerca del usuario del software del DSS. En el peor de los casos, el canal podría enviar fragmentos de las llaves privadas del usuario. No existen casos sobresalientes de este tipo de subversión ocurriendo en un producto auténtico; sin embargo es un riesgo del cual se debe estar consciente.

3.6 Requerimientos técnicos para las firmas digitales

Para el adecuado funcionamiento de las firmas digitales, existe una serie de requerimientos técnicos. Los mismos se enlistan a continuación en orden de mayor a menor importancia:

- **Seguir las reglas de seguridad para el algoritmo.** Para RSA, la implementación debe cumplir con los requerimientos definidos en los Estándares de Criptografía de Llave Pública (PKCS, siglas en inglés). Para DSS, la implementación debe cumplir con los requerimientos especificados en la sección 3.3.

- **Firmar el resultado de una función hash adecuada.** No sólo se deben de firmar los datos por sí solos, también se debe de firmar el resultado de la función hash calculada de los datos. Las elecciones razonables para calcular los valores hash incluyen los algoritmos MD5 y SHA. Las funciones hash deben de cumplir con los requerimientos descritos en la sección 3.2.3.

- **Utilizar una llave pública suficientemente larga.** Como se ha mencionado con anterioridad, las llaves públicas deben ser significativamente más largas que las llaves privadas para dificultar los ataques de fuerza bruta, tal como se menciona en la tabla 2.4, capítulo 2.

- **Utilizar una llave pública separada para las firmas digitales.** En teoría, se puede utilizar la misma llave pública para la encriptación y las firmas digitales, especialmente si se usa RSA. Sin embargo, debe ser posible utilizar llaves separadas para cada operación. Las restricciones legales podrían limitar el tamaño de la llave utilizada para la encriptación, pero no se aplica una restricción similar para las llaves utilizadas en las firmas digitales. Si es posible utilizar llaves separadas, entonces las firmas pueden ser difíciles de falsificar aun si la encriptación es vulnerable a los ataques de fuerza bruta.

- **Evitar firmar los mensajes no cambiados de otras personas.** Se debe de procurar que el mensaje produzca un valor hash diferente a cualquiera que pudiera ser adivinado por otra persona. Para lograr esto, cualquier cambio trivial será suficiente. Un cambio trivial puede consistir en un saludo u otro comentario sin importancia. Esto previene un ataque de "cumpleaños". Aun cuando un ataque de este tipo es poco probable, esto se considera una buena práctica a seguir.

3.7 Ataques y Vulnerabilidades de las Firmas Digitales

A continuación se mencionan los ataques conocidos contra las firmas digitales y las llaves que se utilizan durante su proceso de generación. Sin embargo, todos los ataques pueden ser prevenidos mediante el uso cuidadoso de las firmas digitales en el software de aplicación.

3.7.1 Ataque de números "lisos"

Un *número liso* es el producto de dos números primos razonablemente pequeños. Si un intruso tiene una colección de mensajes firmados por la persona emisora y las firmas están compuestas de números primos pequeños o productos de los mismos, puede utilizar estos mensajes para construir una especie de "alfabeto" de los valores de las firmas digitales del emisor. De esta forma, el intruso puede construir un mensaje firmado que utilice un valor existente o un producto o potencia de valores en ese "alfabeto".

Este ataque se vence fácilmente combinando el valor firmado con un valor aleatorio diferente de cero. Este procedimiento es requerido por el estándar PKCS para las firmas digitales.

3.7.2 Ataque de raíz cúbica

Como se mencionó con anterioridad, si el valor de la llave pública es 3, entonces el valor de una firma digital generalmente es una raíz cúbica. Si al valor hash se le añaden ceros a la derecha y después se genera la firma digital, un atacante puede generar un texto diferente para firmar, obtener el valor hash, obtener la raíz cúbica del valor y llenar con dígitos aleatorios a la derecha para completar a la siguiente raíz cúbica entera. Con este procedimiento podría falsificar una firma digital.

Este ataque es contrarrestado firmando el resultado del valor hash junto con un indicador de la función hash que fue utilizada (MD5, SHA-1, etc.). Además se deben de añadir valores aleatorios a la izquierda de una manera

controlada. Posteriormente el valor entero es firmado digitalmente. Esta técnica es recomendada en las especificaciones del estándar PKCS de firmas digitales.

3.7.3 Ataque de valores coincidentes

Así como la encriptación debe ser diseñada para resistir ataques de fuerza bruta, una función hash debe resistir intentos de construir diferentes versiones de un mensaje que produzcan el mismo valor hash. Al hecho de calcular dos valores hash iguales para dos mensajes diferentes se le llama *colisión*. Una función hash que genere valores hash de 64 bits requeriría 2^{64} mensajes para que una colisión ocurriera. Si se requieren 50 μseg. Para generar valores hash, entonces tomaría 12 millones de años en promedio para encontrar un valor hash de 64 bits igual. Sin embargo, se puede reducir el tiempo de búsqueda. Esto se logra mediante la generación de un número largo de mensajes para localizar dos que tengan el mismo valor hash. Al aumentar el tamaño de los números, la búsqueda requeriría solamente la mitad del número de bits, por lo que tomaría por ejemplo, 2^{32} mensajes en promedio para encontrar dos textos que produzcan el mismo valor hash de 64 bits. Si el tiempo de generación de valores hash es de 50 μseg, entonces la búsqueda se tomaría menos de medio día en promedio para atacar una función hash de 64 bits.

Para lograr aplicar este ataque de manera exitosa, una persona podría alterar un mensaje firmado digitalmente por el emisor a su conveniencia, de tal forma que si se variaran las palabras o se intercambiaran por sinónimos, se generarían billones de variaciones (por lo menos 2^{32}) de cada mensaje. Para cada uno se calcularían los valores hash, de tal forma que se buscaría un valor hash coincidente con el del mensaje original. Una vez hallado el mensaje adecuado, el receptor reemplaza el mensaje original por el que encontró con un valor hash coincidente y cuyo contenido pudiera convenir a sus intereses. Este mensaje podría ser por ejemplo, un acuerdo firmado digitalmente de algún pago u oferta. Al momento de concretarse la transacción, si al emisor se le requiere comprobar que su mensaje fue firmado por él, la firma sería válida debido a que los valores hash coincidirían. Este tipo de ataque implicaría cierto costo considerable en términos de tiempo, equipo y esfuerzo. Sin embargo, existen transacciones de grandes volúmenes que valdría la pena proteger ante la posibilidad de un ataque de este tipo.

La medida básica contra este tipo de ataques es utilizar un tamaño de valor hash que sea por lo menos el doble del tamaño de un posible ataque de fuerza bruta. Por ejemplo, algoritmo MD5 genera un valor hash de 128 bits mientras que el algoritmo SHA produce un valor de 160 bits. Otra técnica consiste en no mandar un mensaje que pudiera tener un valor hash predecible. Para esto, se puede hacer un pequeño cambio al mensaje antes de firmarlo, como por ejemplo añadir un número aleatorio de espacios al mensaje, con lo cual se afectaría el

valor hash. De esta manera sería aun más difícil encontrar un valor hash que coincidiera con el del mensaje original.

3.8 Los certificados de llave pública

A diferencia de las llaves privadas, las llaves públicas pueden proteger información aun si se convierten en conocimiento público. Aun cuando esto las hace más sencillas de distribuir que las llaves privadas, existen algunos problemas para su distribución. No podemos confiar en las capacidades de la Criptografía de Llave Pública si no podemos estar seguros de quién realmente es dueño de la llave privada correspondiente. El objetivo esencial es enviar tanto la llave pública como el nombre correcto de su dueño. En la práctica, los sistemas usan una combinación de dos mecanismos para distribuir las llaves públicas:

1. **Certificados**: La mayoría de las llaves se distribuyen en bloques de datos especiales llamados *certificados*. Un certificado siempre contiene 3 porciones de información: un nombre, una llave pública y una firma digital compuesta de los dos elementos anteriores. El propósito del certificado es mostrar que una llave pública en particular proviene de un individuo específico.

2. **Distribución manual segura**: El resto de las llaves públicas se distribuyen mediante técnicas manuales seguras tales como tarjetas criptográficas, envío tradicional por mensajería con papel y sobres especiales o partición de llaves y envío de los fragmentos por separado. Generalmente, las únicas llaves públicas que se distribuyen manualmente son aquellas que se usan para verificar las firmas en los certificados. Sin embargo, éstas se pueden obtener directamente de la autoridad certificadora sin que sea indispensable la distribución manual.

Un certificado de llave pública es una estructura de datos que identifica al dueño de una llave en particular. El certificado es un bloque de datos digitalmente firmados que contiene una llave pública y el nombre del dueño de la llave. El certificado declara que una entidad con un nombre en particular es dueña de una llave pública en particular. La firma en el certificado verifica que la llave y el nombre vayan juntos. Sin la firma de verificación, un atacante podría sustituir una llave pública por otra y hacerse pasar por una persona u otra durante un intercambio de datos utilizando llaves públicas.

La firma de un certificado digital es producida por una tercera entidad llamada *Autoridad Certificadora*. El software criptográfico que se emplee para el intercambio de datos debe tener una copia de la llave pública de la autoridad

certificadora para poder verificar la firma digital de un certificado. La verificación se realiza de acuerdo a la siguiente figura:

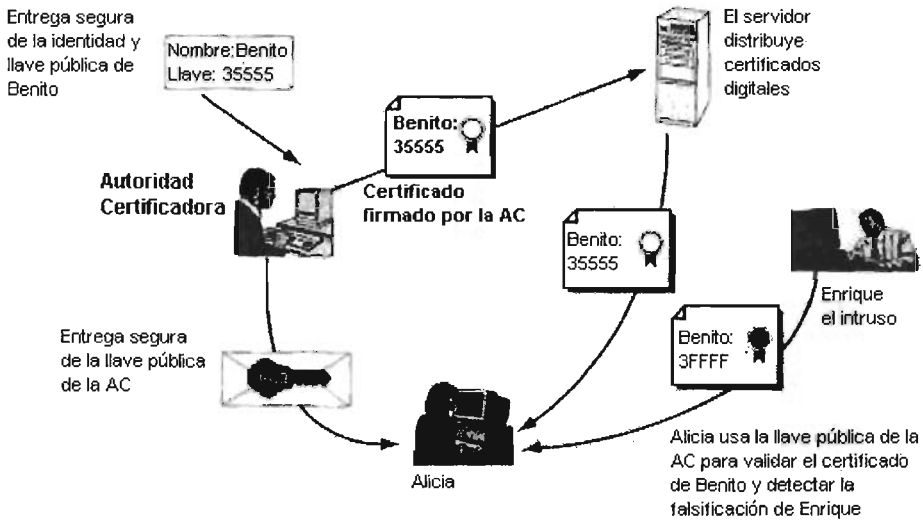


Fig. 3.7: Los certificados se firman con la llave privada y validados con la llave pública de la Autoridad Certificadora

Un valor hash es calculado sobre el nombre y el valor de la llave pública. La firma digital es desencriptada usando la llave pública de la autoridad certificadora, lo cual produce un segundo valor hash. Los dos valores hash deben de coincidir si el certificado es válido.

Un software configurado apropiadamente rechazará certificados que no hayan sido emitidos por una autoridad certificadora que no esté definida dentro del mismo como una autoridad reconocible. El software también rechazará todos los certificados que pretenden ser emitidos por una autoridad aceptable pero que pueden sospecharse como falsificaciones. La prueba para detectar la falsificación consiste en verificar la firma digital del certificado usando la llave pública de la autoridad certificadora.

Los certificados pueden usar cualquier mecanismo de firmas digitales, aunque la mayoría de los sistemas de software comercial usan el algoritmo RSA. Por ejemplo, los certificados tradicionalmente usados por los servidores de Netscape basados en el protocolo SSL contienen una firma digital generada por medio del algoritmo RSA. Todas las copias del navegador de Netscape contienen la llave pública de una autoridad certificadora específica en Verisign, Inc. La versión más reciente del navegador permite a los usuarios ingresar llaves públicas adicionales para otras autoridades que utilicen otros algoritmos. Protocolos para el envío seguro de correo electrónico tales como PEM (Privacy Enhanced Mail) y PGP (Pretty Good Privacy) utilizan el algoritmo RSA para

firmar sus certificados, mientras que la tarjeta "Fortezza", utilizada mayormente en el ámbito militar en Estados Unidos, emplea el Estándar de Firmas Digitales.

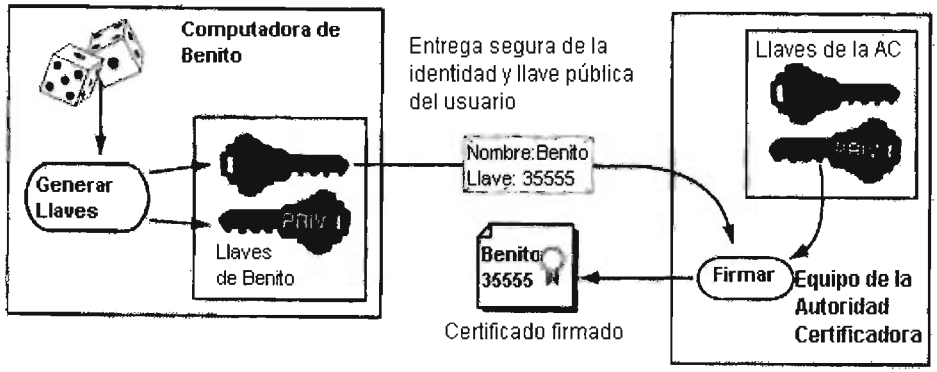


Fig. 3.8: Generación de certificados digitales

La figura 3.8 ilustra la construcción de un certificado de llave pública. El usuario genera su llave privada y pública mediante un algoritmo tal como RSA. Mantiene la llave privada secreta y la pública la envía a una autoridad certificadora de manera segura por ejemplo, mediante una VPN (Red Privada Virtual) o por algún método de distribución manual seguro como los mencionados con anterioridad. La autoridad certificadora verifica la identidad del usuario contactándolo y verificando sus datos y a partir de ese momento genera un par de llaves (pública y privada) con las cuales firma un certificado que sirve para comprobar la identidad del usuario cada vez que éste transmita datos encriptados a otros usuarios.

3.8.1 Generación de pares de llaves públicas

Existen dos métodos de generación de pares de llaves públicas para la generación de certificados digitales. Algunos sistemas las generan en el equipo del propietario de la llave mientras que otras se generan desde la autoridad certificadora. La figura 3.8 ilustra el primer caso. El usuario genera un par de llaves, pública y privada, retiene la llave privada y envía la llave pública a la autoridad certificadora para producir el certificado. En el segundo caso, la autoridad certificadora genera el par de llaves, produce el certificado firmado y lo entrega al usuario interesado y el cual debe recibir el par de llaves.

Existen varias ventajas cuando las llaves son generadas por el usuario dueño de las mismas. Así por ejemplo, una de las principales es que la llave privada no necesita ser copiada o respaldada, ya que ésta no tiene necesidad de residir en otra computadora que no sea la del dueño. El realizar copias de la misma simplemente incrementa el riesgo de que la llave pudiera ser transmitida

y compartida por usuarios no autorizados que pudieran realizar falsificaciones con la misma. Por otro lado, empero, la problemática que este proceso envuelve es que el dueño de las llaves debe enviarlas a la autoridad certificadora

Existe un número de implicaciones prácticas y de seguridad cuando la autoridad certificadora debe generar todas las llaves. Este método centralizado impone procedimientos más simples a los usuarios. Por ejemplo, cuando el usuario debe generar sus propias llaves, el proceso requiere de tres pasos en lugar de uno (generar las llaves, enviarlas a la autoridad certificadora y recibir el certificado de la misma). Si la autoridad certificadora genera las llaves, el usuario simplemente tendría que esperar la entrega de las llaves y el certificado firmado. Las copias de respaldo de lo anterior que realiza la autoridad certificadora le permiten recuperar datos que el usuario hubiese encriptado con anterioridad. Esto podría ser útil para proteger al usuario de pérdidas de llaves o información. Asimismo, podrían existir beneficios adicionales de seguridad si la autoridad certificadora utiliza hardware especial para generar números aleatorios más difíciles de predecir y validar más efectivamente los factores primos de las llaves públicas. Sin embargo, una posible problemática de este método reside en que las llaves privadas del usuario son copiadas y susceptibles de mayor movilidad. Cada vez que se copia y/o envía una llave, se incrementa el riesgo de una divulgación indebida de la misma, a diferencia de cuando el usuario genera las llaves, pues en ese caso la llave privada no tiene razón para abandonar el equipo. Mientras más copias existieran de las llaves privadas, se incrementaría el riesgo de que personas no autorizadas intentaran obtenerlas para usos indebidos.

3.8.2 Revocación de certificados

En el evento de que una llave privada fuese revelada de manera accidental o se necesitara cambiarse por cualquier razón, se genera un problema complicado. El certificado de la llave pública correspondiente seguiría permaneciendo válido y no habría forma de acceder a cada computadora que almacenara una copia de dicho certificado. Aun cuando existen varios mecanismos que han sido diseñados para tratar este problema, pocos son efectivos y no han sido desplegados ampliamente.

La solución típica en los sistemas actuales de llave pública es la "Lista de Revocación de Certificados", la cual contiene una lista de número seriales de certificados que han perdido su validez. La Autoridad Certificadora recolecta reportes de certificados que ya no son válidos. Si el sistema tiene más de una Autoridad Certificadora, las autoridades comparten sus listas para producir una lista global de certificados inválidos. Esto es similar a las listas de tarjetas de crédito rechazadas que los comerciantes de manera electrónica verifican al procesar una compra.

Cuando alguien intenta validar un certificado, la Autoridad Certificadora consulta una Lista de Revocación de Certificados, la cual se encuentra por lo regular almacenada en un servidor LDAP, el cual es un servidor que provee de servicios de directorio. La actualización del mismo, sin embargo, es una tarea laboriosa que pudiera traer consigo problemas. Para simplificar esta tarea, al crear un certificado digital existe un campo en el cual se puede definir una fecha de expiración. Cuando se alcanza la fecha indicada, el certificado se vuelve automáticamente inutilizable para nuevas transacciones. Posteriormente, es necesario emitir un nuevo certificado para reemplazar el expirado. Sin embargo, esta tarea es más simple que cancelar un certificado y actualizar la Lista de Revocación de Certificados cada vez que esto se realizara. Por lo tanto, el definir fechas de expiración para certificados digitales debería ser un procedimiento estándar cuando se trabaja con un gran número de los mismos.

3.8.3 Estación de trabajo de la Autoridad Certificadora

La Estación de trabajo de la Autoridad Certificadora, o *Certification Authority Workstation* (**CAW**) como se le conoce comúnmente en inglés, es un sistema especial y dedicado que firma certificados de llave pública de la forma descrita en la figura 3.8.

La estación de trabajo de la autoridad certificadora (CAW) contiene las llaves privadas de la autoridad certificadora así como el hardware y software requerido para almacenar, firmar, distribuir y revocar certificados. Por lo tanto, la estación de trabajo debe de contar con controles de seguridad física para prevenir el acceso no autorizado, de tal forma que sólo pudiera acceder a la misma el personal a cargo de su operación, así como contar con controles de seguridad lógica para reducir el riesgo de un mal uso del equipo, tal como realizar falsificaciones de certificados.
La CAW debe ser tratada como un dispositivo especial y no como un paquete de software dentro de una computadora de usos múltiples. Existen muchos riesgos en el proceso de certificación que pudieran existir si se compartiera el equipo con otras actividades o si se encontrara conectado a una red. Asimismo, si se introdujera software adicional en el equipo, se podrían generar errores de software que podrían interferir con el proceso de generación de certificados y dañar material criptográfico ya existente. Eventualmente, la presencia de software innecesario en el equipo podría servir como una entrada al equipo, por medio de la cual se pudieran recibir ataques que pudieran volver al equipo inutilizable, por lo que la estación de trabajo no debería ser conectada a una red de uso común.

La llave privada que reside en la CAW constituye un fragmento crítico de información dentro del sistema de manejo de llaves de una compañía. La integridad de los certificados reside enteramente en la confidencialidad de la llave privada. Aun cuando tal llave reside en la CAW, la misma pertenece a la

organización que las deposita en el equipo de la autoridad certificadora. Por lo tanto, si dicha autoridad certificadora dejara de operar, seguiría siendo necesario utilizar la misma llave. De lo contrario, sería necesario actualizar la llave pública de la autoridad certificadora en todo el software distribuido a lo largo de la organización y fuera de ella, y luego volver a emitir todos los certificados bajo la nueva llave. Sin embargo, esta solución no es práctica.
Debido a que la llave privada pudiera ser almacenada en la CAW como cualquier otro tipo de dato, la solución radica en configurar el equipo de tal forma que el personal de la autoridad certificadora no tenga acceso directo a la misma. El grado de restricción de acceso que se aplique depende de la magnitud del daño que se pudiera tener en caso de que los certificados fuesen falsificados. Por lo tanto, una solución práctica consiste en aplicar permisos de archivo y posiblemente alguna encriptación secundaria a la llave privada de tal forma que un operador del equipo no pudiera tener acceso a la misma. Idealmente, la llave debería permanecer encriptada cuando no se encontrara en uso. De esta forma, el operador podría firmar certificados sin necesidad de otorgar o contar con acceso directo a las llaves. Sin un acceso directo a las llaves, se dificulta la posibilidad de copiar las llaves privadas para hacer un uso indebido de las mismas.

El método más efectivo para restringir el acceso a las llaves privadas que se utilizan en los certificados digitales es el uso de componentes de hardware que contienen las llaves y algoritmos criptográficos. Estos componentes encapsulan los datos en un paquete a prueba de intrusos. Un ejemplo de esto es la tarjeta *Fortezza*, desarrollada por el gobierno de Estados Unidos, la cual es utilizada en aplicaciones en conjunto con el protocolo SSL. La tarjeta *Fortezza* trae consigo sus propios algoritmos criptográficos y llaves en una tarjeta para PC. Estas tarjetas proveen de protección fuerte para las llaves criptográficas y para la integridad de los algoritmos criptográficos. Los intrusos no podrían extraer llaves de las mismas aun si se encontraran en su modo normal de operación. En el último de los casos, un atacante podría intentar forzar una tarjeta activa para aplicar algún algoritmo sobre ciertos datos; sin embargo, el daño potencial estaría limitado al tiempo disponible para poder efectuar operaciones criptográficas separadas.
Existen versiones comerciales de la tarjeta *Fortezza*, las cuales implementan los algoritmos criptográficos estándar, tales como AES, RC4, IDEA y son producidas por proveedores como Nortel, Spyrus, Hewlett-Packard, entre otros.
Las llaves almacenadas en la CAW también pueden ser almacenadas en este tipo de dispositivos y éstos a su vez pueden ser almacenados en una ubicación física segura. Adicionalmente a la seguridad física en el almacenamiento de los dispositivos, el operador debe siempre ingresar una contraseña para activar el módulo. Esta contraseña permite utilizar las llaves junto con los algoritmos criptográficos contenidos en el dispositivo sin revelar las llaves. Un uso similar de contraseñas en implementaciones de software convencionales no protegería a las llaves de ser reveladas a un atacante persistente.

La ventaja adicional que ofrece un dispositivo de hardware tal como la tarjeta *Fortezza*, utilizado para el almacenamiento de llaves y algoritmos, consiste en que la Autoridad Certificadora elegida para emitir certificados tendrá acceso físico al dispositivo así como la contraseña para activarlo y poder generar firmas digitales durante el tiempo que dure el contrato acordado. En caso de que terminara el contrato de servicio con la Autoridad Certificadora o ésta dejara de existir, no tendrá más acceso al o los dispositivos de almacenamiento, por lo que el personal no podrá continuar firmando certificados digitales. Asimismo, el dispositivo permite el cambio de contraseña de acceso como medida de seguridad adicional. La única forma en que el personal de la Autoridad Certificadora pudiese continuar haciendo uso de las llaves sería sustrayendo el dispositivo. Sin embargo, al encontrase éste bajo la custodia de una Autoridad Certificadora, se reduciría considerablemente el riesgo de un robo, debido a que la tarjeta o dispositivo de hardware servirían como una evidencia física de la existencia de las llaves criptográficas, con lo que se facilitaría el rastreo de los responsables y la realización de las acciones legales pertinentes.

3.9 Distribución de Certificados

Los certificados digitales nos proporcionan un método seguro para distribuir llaves públicas a través de medios electrónicos. El siguiente problema consiste en entregar los certificados a los equipos que necesitan autentificar a determinados recipientes. Existen dos técnicas principales para distribuir certificados digitales: distribución transparente y distribución interactiva.

3.9.1 Distribución transparente

Se realiza por medio de servidores de directorio y protocolos de intercambio de llaves. El protocolo más utilizado actualmente para acceder a dichos servidores de directorio es el Lightweight Directory Access Protocol (LDAP). Por otro lado, los protocolos utilizados para la encriptación en redes proveen de facilidades para entregar certificados de llave pública mediante el intercambio de llaves. Esto se puede observar en protocolos tales como IPSEC, SKIP, ISKAMP y Photuris. También es una parte importante del protocolo SSL y su variante, PCT.

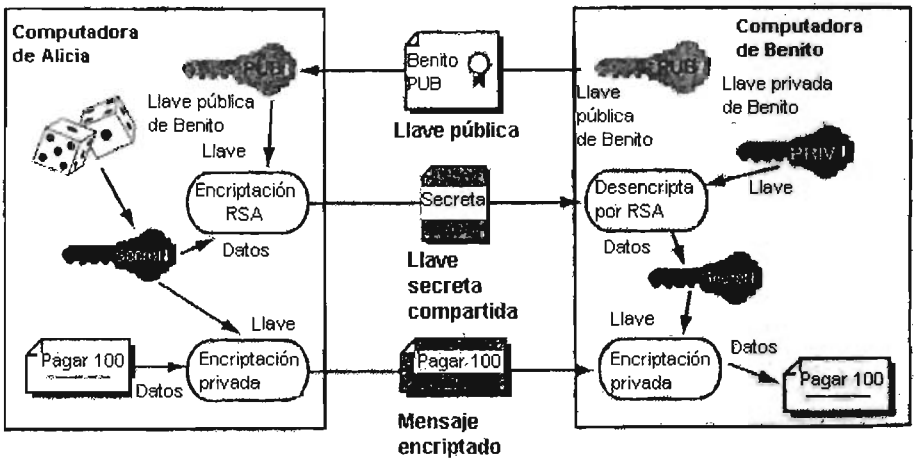


Fig. 3.9: Intercambio de llaves privadas por medio de RSA

Este proceso se puede observar en la figura 3.9 Si un usuario A inicia un intercambio de llaves privadas, el usuario B provee de su certificado de llave pública. El usuario A valida este certificado usando la llave pública de la Autoridad Certificadora (AC) y la llave pública del usuario B para proteger el mensaje de intercambio de llaves. Si el usuario B desea autentificar al usuario A, puede requerirle su certificado de llave pública. Entonces, el usuario B valida el certificado usando la llave pública de la AC e incorpora la llave pública del usuario A en el proceso de intercambio de llaves. El protocolo de intercambio de llaves IPSEC maneja los certificados de una manera similar. Asimismo, se utilizan certificados para autentificar los valores de llave pública generados a través del algoritmo Diffie-Hellman u otros.

3.9.2 Distribución interactiva

Aun cuando la distribución transparente de certificados es usualmente la mejor técnica, muchos sistemas se apoyan en otros mecanismos para distribuir certificados interactivamente. Cuando los equipos de los usuarios necesitan certificados, realizan una búsqueda electrónica para encontrar el certificado que necesitan. Existen diferentes sistemas de obtención de dichos certificados. Estos sistemas proveen de un mecanismo en forma de directorio para almacenar una colección de certificados que el dueño del equipo podría necesitar. De esta forma, también se puede incrementar la eficiencia de la verificación de certificados. El sistema puede verificar la firma en el certificado una vez que ha sido salvado y usarlo enseguida. Sin embargo, este método puede generar algunos problemas, si el certificado almacenado se vuelve inválido.

Existen tres mecanismos típicos empleados para recolectar certificados interactivamente:

- E-mail

  El correo electrónico (e-mail) distribuye los certificados de dos maneras diferentes. Primeramente, muchos sistemas para e-mail con soporte de encriptación proveen de una forma de incluir un certificado con el mensaje que se envía. Esto simplifica labores en caso de que el recipiente necesite enviar una respuesta, ya que el certificado se encuentra disponible inmediatamente. Sin embargo, esto puede aumentar el tamaño de los mensajes innecesariamente. En la práctica, los emisores suelen incluir un enlace hacia donde se puede localizar el certificado.
  Un segundo método consiste en utilizar un servidor de certificados que acepte peticiones provenientes de e-mail. Diversos sistemas cuentan con configuraciones para atender peticiones de certificados mediante el protocolo PGP (Pretty Good Privacy) usado en aplicaciones de correo electrónico. Cuando un usuario requiere de un certificado en particular éste envía su petición a un servidor PGP de llaves localizado en alguna región del mundo. El mismo método puede ser aplicado a directorios basados en el protocolo X.500.

- Sitios Web

  Conforme los sitios Web se vuelven más y más comunes, también lo son las páginas personales. Los usuarios pueden publicar sus certificados directamente en su página personal o fácilmente ligados a la misma. De esta forma, se puede enlazar directamente a los mismos proporcionando la URL.

- Peticiones de finger

  Muchos sitios de Internet soportan peticiones "finger", las cuales devuelven información de identificación acerca del dueño de una dirección de e-mail en Internet. Típicamente, una petición "finger" devuelve el nombre del dueño y su información en un formato de texto ASCII. Los usuarios de e-mail que incorporan PGP también tienen la posibilidad de incluir su certificado de tal forma que otros usuarios puedan ver su información fácilmente.

Independientemente de la forma en que los certificados son recibidos, todos están expuestos a un nivel equivalente de riesgo de un ataque. Por esto, se debe de verificar la firma digital contenida en un certificado contra una llave pública en la que se pueda confiar (por ejemplo, a través de la Autoridad Certificadora) antes de utilizarla. De otra forma, un atacante podría explotar

vulnerabilidades de los diferentes mecanismos de entrega para proveer con un certificado falsificado.

Las vulnerabilidades aparecen durante la travesía de los certificados por Internet, así como en los servidores de llaves. Los certificados en tránsito podrían ser sustituidos por un atacante sofisticado, o éstos podrían ser completamente fabricados y transmitidos por un host (sistema remoto) invadido. La mejor práctica a seguir en estos casos para evitar información fraudulenta es verificar que la fuente o autoridad emisora de certificados es razonablemente confiable y comparar la información recibida con datos que se puedan verificar directamente con la entidad certificadora en caso de que existan dudas sobre el origen de un certificado en particular.

3.10 Métodos de certificación digital

3.10.1 Autoridad Certificadora Centralizada

El método más simple de certificación digital se logra mediante la Autoridad Certificadora Centralizada. El método se basa en la existencia de una sola AC con un conjunto único de llaves para firmar todos los certificados válidos en el sistema. Todo el software de llave pública debe de contener una copia válida de la llave pública de la AC para validar los certificados. Cualquier otro certificado firmado por una AC diferente sería rechazado. Este método tiene varias ventajas de seguridad, pero carece de la flexibilidad que muchos desarrolladores de aplicaciones requieren. Por lo tanto, muchos proveedores de software proveen de mecanismos para instalar llaves públicas de múltiples autoridades.

El método de certificación digital mediante una Autoridad Certificadora Centralizada provee de ciertas ventajas, especialmente para organizaciones preocupadas por la seguridad de sus activos de información. Proporciona a las organizaciones un control total sobre el proceso de certificación debido a lo siguiente:

1. Los certificados son aceptados o rechazados únicamente sobre la base de que éstos fueron firmados efectivamente firmados por la Autoridad Certificadora de la organización. Terceras personas no pueden crear certificados aceptables sin antes pasar por las medidas de seguridad de la organización.

2. El punto centralizado de certificación facilita el cumplimiento de condiciones específicas confiables para la emisión de certificados, debido a que es difícil mantener las políticas uniformes a través de múltiples autoridades, especialmente si algunas de éstas se encuentran en otras organizaciones.

3. La organización puede utilizar los certificados como credenciales para otorgar acceso a recursos restringidos, tales como accesos remotos al sitio mediante dial-in sobre comunicaciones encriptadas (VPN), ya que la organización controla por completo las condiciones en las que los certificados son emitidos.

4. La organización es la única responsable por la seguridad física de las llaves privadas utilizadas en la certificación. Por lo tanto, no son vulnerables a huecos de seguridad o descuidos por parte de otras organizaciones involucradas.

La desventaja principal de este método es que puede producir "cuellos de botella" en el sistema. Esto es debido a que cada operación de certificación debe de realizarse a través de una sola estación de trabajo y cada una debe esperar su turno. Estas actividades son costosas y podrían no contar con la supervisión adecuada, especialmente debido a que se requiere de un entrenamiento especial y un alto grado de confiabilidad. Asimismo, se podrían producir problemas de índole política o legal si la Autoridad pertenece a una organización y emite certificados a otras. Como ejemplo, esta situación se presentó en las primeras versiones del software de seguridad Web SSL de Netscape, al ser este proveedor el primero en emplear la certificación digital en navegadores Web.
A continuación se describe más a detalle la autentificación utilizada por Netscape, debido a que este método ha sido el más utilizado dentro de los navegadores Web.

3.10.2 Autentificación por servidor de Netscape

Las versiones iniciales del navegador Web Netscape Navigator validaban los certificados de los servidores Web contra la llave de una autoridad central. Inicialmente, todos los certificados de servidores que empleaban el protocolo SSL fueron emitidos por una AC comercial operada por Verisign, Inc. La llave pública de la autoridad se encuentra integrada en toda copia del navegador de Netscape. Cuando el navegador inicia una sesión SSL con un servidor, éste envía su certificado de llave pública. El navegador valida la firma digital del certificado usando la llave pública de la AC, tal como Verisign. Este método ha brindado simplicidad operativa y una interpretación más clara de los certificados que son enviados y recibidos. Esta simplicidad provino de la confianza en una sola llave pública, debido a que los usuarios en un principio no necesitaban instalar llaves criptográficas, ya que el software SSL del navegador establecía automáticamente una llave de sesión y validaba el certificado del servidor. Las actividades criptográficas estaban basadas enteramente en la llave de Verisign y otros datos incluidos en todas las copias del navegador. Los servidores no tenían ninguna elección en el asunto: para interactuar con los navegadores debían obtener un certificado de la AC Verisign.

La interpretación más claro de los certificados digitales provino de la Autoridad Certificadora Centralizada. Los certificados fueron creados con la intención de identificar de manera confiable servidores y equipos conectados en Internet como soporte para las transacciones comerciales electrónicas. Los desarrolladores de software tomaron en cuenta las necesidades de usuarios en general para comunicarse, especialmente si desearan realizar una compra electrónica. Los certificados fueron diseñados de tal forma que mostraran a la compañía propietaria del sitio Web en cuestión así como para identificar el nombre del host del sitio de Internet. Verisign estableció un conjunto de reglas para emitir certificados que requirieran documentación legal que acreditara al poseedor de una llave pública en particular como la entidad legítima poseedora del certificado. Los usuarios podrían de esta forma comprobar que los nombres en los certificados coincidirían con los de los hosts en Internet y que sólo serían emitidos a la organización que realmente fuese dueña de esos hosts.

Sin embargo, este rígido método también ha tenido desventajas. Esto es debido a que no funcionó de manera óptima para organizaciones interesadas en utilizar el navegador Netscape y el protocolo SSL para propósitos privados. Comenzaron a surgir otras organizaciones que operaban sus propios sistemas de certificación y no deseaban soportarse en Verisign para obtener y proporcionar el serivcio. Asimismo, estudiantes e investigadores que conforman una parte importante de la comunidad en Internet no tuvieron la posibilidad de experimentar eficientemente con mecanismos de certificados digitales debido a que requerían necesariamente de certificados emitidos por un servidor de Verisign.

Esto dio lugar al soporte hacia múltiples Autoridades Certificadoras. Por ejemplo, a partir de Netscape Navigator 3.0, el navegador comenzó a permitir la instalación y uso de llaves públicas provenientes de otras AC's, así como soporte para AC's jerárquicas, como se describe más adelante.

3.10.3 Manejo de múltiples Autoridades Certificadoras

Debido a que una sola Autoridad Certificadora, si bien simplifica el software de certificación digital, también lo vuelve inflexible. Eso por esto que una alternativa para versiones más recientes de navegadores, tales como Netscape Navigator e Internet Explorer, es la instalación de llaves públicas adicionales para Autoridades Certificadoras alternas. Por lo tanto, una vez instaladas en el navegador, éste validará los certificados firmados por cualquiera de las autoridades que se encuentren en su base de datos de llaves públicas. Esto provee de un balance útil de flexibilidad, conveniencia y riesgo. La flexibilidad y conveniencia resultan atractivas principalmente para los desarrolladores que experimenten con las capacidades ofrecidas por las técnicas de Criptografía de Llave Pública.

El riesgo en este caso proviene de la incertidumbre producida por una base de datos variable de llaves públicas aceptables, ya que una llave falsificada podría ser instalada junto con las legítimas o un atacante podría reemplazar la llave instalada perteneciente a la autoridad central (en caso de que se usara este método) con una llave falsificada. El ataque podría ser hasta cierto punto reconocible ya que impediría que los certificados legítimos trabajaran correctamente. Sin embargo, el riesgo cambia cuando se cuenta con una lista de llaves para verificar. Si un atacante inserta una llave falsa en la lista, las llaves legítimas continuarían funcionando. La llave falsa simplemente significaría que los certificados firmados por la autoridad ficticia también serían aceptados como legítimos.

Otro riesgo consiste en que un certificado legítimo podría ser malinterpretado. Por ejemplo, si un certificado de un servidor Web fue emitido por una autoridad principal, el equipo está configurado para contener una copia de la llave pública. Sin embargo, el mismo podría estar configurado para contener llaves públicas de otras entidades certificadoras. Por lo tanto, un atacante podría intentar obtener de alguna de esas entidades un certificado similar al que fue emitido por la autoridad principal, haciéndose pasar por la organización dueña del servidor Web. De esta forma, el intruso podría utilizar el certificado obtenido para llevar un ataque de tipo "Hombre En Medio" en el cual, cuando un usuario del sitio Web inicia una conexión y sesión de autentificación el intruso interrumpe la transmisión del certificado para insertar su propio certificado. Debido a que el certificado que el intruso obtuvo fue firmado previamente por una de las autoridades registradas en la base de datos del servidor Web, el navegador del usuario aceptaría este certificado como válido.

Los riesgos antes descritos se presentaron como vulnerabilidades propias de versiones tempranas de los navegadores de Netscape, así como las versiones 4.0 y 5.0 del navegador Internet Explorer, sin embargo, los proveedores han publicado parches para corregir estas debilidades, al requerir que coincidan los nombres de los servidores Web en las URL's y los que se encuentran registrados en el certificado usado durante la sesión. Este parche es suficiente para prevenir un ataque de este tipo, sin embargo este riesgo demuestra que cualquier software que utilice certificación mediante llaves públicas que acepte varias autoridades debe de tener una interpretación y validación segura de distintos certificados.

3.10.4 Autoridad de Certificación Jerárquica

Debido al inconveniente que para muchas empresas y actividades representaba el utilizar una autoridad centralizada debido a que se formaban "cuellos de botella", se desarrolló el método de la certificación jerárquica, el cual pretende combinar las características de una certificación centralizada con un arreglo más distribuido. Las técnicas PEM y DMS (desarrollado para uso en el

ejército de Estados Unidos) así como el sistema de autentificación de usuarios en versiones más recientes de los navegadores Netscape e Internet Explorer utilizan el método jerárquico.

Los sistemas jerárquicos comienzan con una autoridad raíz con una llave pública que usualmente es distribuida a todos los clientes participantes. A diferencia de los sistemas centralizados, la emisión de certificados es delegada a varias autoridades certificadoras. La autoridad raíz sólo firma certificados para algunas autoridades. Algunas otras autoridades solamente podrán firmar para autoridades de menor nivel, y así sucesivamente. En la base del sistema se encuentran las autoridades que firman certificados para entidades individuales de usuarios.

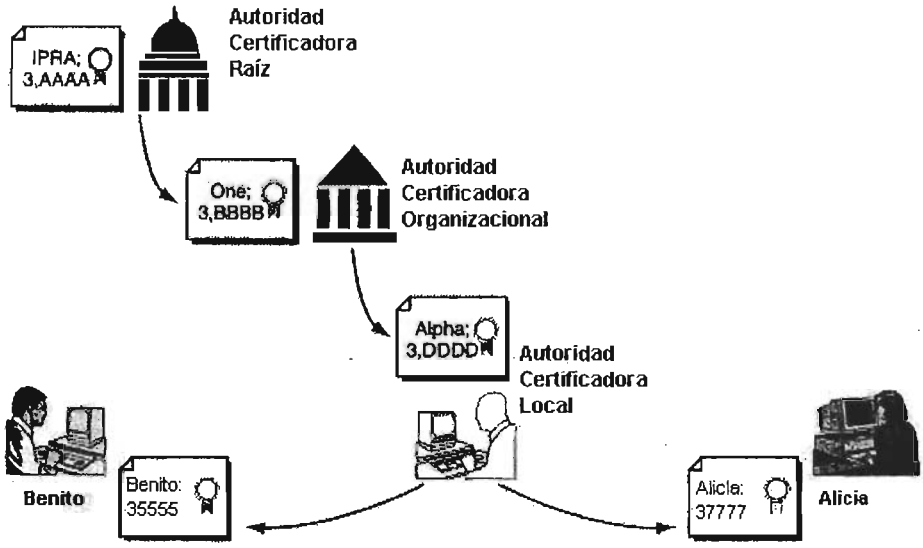


Fig. 3.10: Esquema de Autoridad de Certificación Jerárquica

La serie de certificados que conducen de vuelta a la llave pública raíz deben ser recuperados y verificados para poder validar el certificado de un usuario. Debido a que el certificado de un usuario es firmado por una AC local, se utiliza la llave pública del certificado emitido por la misma para validarlo. Para validar el certificado de la autoridad local es necesario usar la llave pública de la autoridad organizacional (llamada también, "Autoridad de Creación de Políticas") que lo firmó. Este proceso se repite sucesivamente hasta llegar a la llave raíz que fue instalada en el software. Esta llave podría ser la llave raíz de la jerarquía entera o una llave subordinada, dependiendo de cómo esté configurada la infraestructura jerárquica.

Las Autoridades de Creación de Políticas (ACP), que se encuentran en segundo nivel dentro de la infraestructura jerárquica, establecen reglas que deben ser acatadas por un grupo particular de AC's a las cuales firman certificados digitales. Por ejemplo, puede existir una autoridad raíz de "alto aseguramiento" que emita certificados a otras autoridades que requieran reforzar un alto estándar de seguridad y autenticidad. Estas autoridades de alto aseguramiento podrían requerir realizar investigaciones de campo antes de emitir un certificado a los usuarios y establecer requerimientos particularmente rigurosos a las AC's. En estas circunstancias, una ACP podría ser utilizada para verificar el cumplimiento de estos requerimientos, o también podría existir una ACP por separado de mediano o bajo aseguramiento, que pudiera emitir por sí misma certificados en circunstancias menos rigurosas. Finalmente, los certificados emitidos por grupos experimentales de desarrollo y pruebas de sistemas o por organizaciones pequeñas, se ubicarían en la categoría de bajo aseguramiento, al ser la flexibilidad operativa normalmente más importante que el aseguramiento del certificado.

Las ACP's emiten certificados a localidades geográficas u organizaciones tales como entidades gubernamentales locales. Estas a su vez operan AC's para usuarios bajo su jurisdicción. Las organizaciones menores usarían su llave pública para firmar los certificados de los usuarios mientras que las organizaciones mayores la usarían para firmar AC's de menor nivel. Algunas organizaciones podrían tener dos o más certificados provenientes de ACP's si necesitaran operar Autoridades Certificadoras bajo distintas políticas.

La arquitectura de certificación jerárquica ha sido escasamente utilizada en la práctica, debido a que originalmente fue establecida a través de la compañía RSA Data Security como autoridad raíz, lo cual fue visto por muchos usuarios como un intento de establecer a un solo proveedor como autoridad raíz, monopolizando de cierto modo la estructura de certificación jerárquica. Sin embargo, los principios de esta arquitectura de certificación han servido como base para la arquitectura utilizada en las tarjetas Fortezza. El sistema incluye ACP's, autoridades organizacionales y una serie de AC's locales para emitir certificados a usuarios individuales.

La arquitectura de certificación jerárquica ha dado lugar a la evolución de diferentes sistemas jerárquicos. En muchos de ellos, la autoridad raíz y las ACP's han desaparecido. En su lugar, un proveedor que ofrece llaves públicas para proporcionar servicios de certificación define su propia autoridad raíz e instala la llave pública correspondiente en el software. De esta forma, se tiene un "bosque" de AC's en lugar de un solo árbol jerárquico.

Lo anterior resulta práctico debido a que los diferentes árboles de certificación pueden ser desarrollados por diferentes razones. Por ejemplo, la autoridad Verisign continúa firmando certificados para el navegador de Netscape, sin embargo ahora existen otros proveedores, tales como Nortel, GTE

y AT&T, los cuales tienen sus llaves públicas instaladas en el navegador. Por otro lado, existe otro tipo de jerarquía desarrollada especialmente para soportar el protocolo SET, el cual es utilizado para proteger transacciones con tarjetas de crédito.

3.10.5 "Red de Confianza" de PGP

El sistema de certificación de PGP (Pretty Good Privacy) presenta características que usualmente se ajustan más a requerimientos de certificación a nivel individual más que a nivel de transacciones comerciales: autoridad personal, flexibilidad y pérdidas personales si surgieran fallas. La criptografía empleada en PGP fue diseñada para permitir a individuos autentificarse entre ellos para lograr comunicaciones razonablemente confiables. Esta solución generalmente resulta más apropiada para un grupo de usuarios particulares o pequeñas empresas.

La característica fundamental de la certificación mediante PGP es la no distinción entre un usuario individual y una Autoridad Certificadora. Por lo tanto, cualquier usuario podría firmar las llaves de otro, actuando como una AC al hacerlo. La convención entre usuarios de PGP es que los certificados sólo son firmados si el firmante está razonablemente seguro de que el certificado es legítimo. Los usuarios juzgan la legitimidad del certificado dependiendo del grado de confianza que tengan acerca de la firma y de si proviene de alguien en quien confíen.

Existen dos criterios esenciales para confiar en un certificado basados en la firma en particular de un usuario: Se debe contar con una copia legítima de la llave pública del usuario y se debe tener un grado razonable de confianza de que el usuario en cuestión sólo acostumbra firmar certificados legítimos. Ambos criterios son importantes. Si se omite uno de ellos, se podrían producir falsificaciones. Por ejemplo, se tienen 4 usuarios, A, B, C y D. Los certificados de los usuarios A y B han sido firmados por C, por lo que A y B confían en los certificados de C, pero también pueden confiar mutuamente en sus certificados. El usuario B ha firmado un certificado al usuario D, sin embargo éste no podría confiar plenamente en los certificados de A y C, pues D no tiene una referencia directa de ellos.
Esta situación podría proporcionar una entrada a los intrusos. Si uno de ellos elaborara una colección de certificados falsificados de los usuarios A, B y C antes mencionados, de tal forma que se pudiera establecer una relación de confianza entre todos ellos, y los presentara al usuario D, éste podría entonces confiar no sólo en los certificados provenientes del usuario B, sino de su falsificación correspondiente, así como las de los otros usuarios. Sin embargo, esto podría prevenirse si el usuario D verifica de manera independiente un certificado de uno de los otros usuarios para detectar discrepancias.

Otro problema que normalmente surge en las redes de confianza de PGP es la "confianza transitiva". Un ejemplo de esto sería si, por ejemplo, A firmara el certificado de B, y éste a su vez firmara el de C. Otros usuarios que confiaran en A podrían confiar también en la legitimidad de los certificados de B ya que éstos fueron firmados por A. Esto no necesariamente implica que los certificados que B firme son necesariamente confiables. Si se deseara establecer una relación estricta de confianza, los demás usuarios no podrían confiar en la legitimidad de los certificados de C, ya que éstos fueron firmados por B y podrían no contar con referencias directas de este usuario para poder determinar que las firmas emitidas por él son del todo confiables. Sin embargo, en la práctica muchos usuarios depositarían confianza transitiva, esto es, todos los certificados de B son confiables puesto que fueron firmados por A. Por ende, todas las firmas digitales generadas por B hacia los certificados de C u otros usuarios también son confiables. Esta práctica es aceptable para un uso casual de PGP. Sin embargo, si se realiza con demasiada frecuencia, se incrementaría el riesgo de aceptar un certificado falsificado.

Algunos usuarios de PGP consideran la firma de certificados digitales como algo de seria importancia, especialmente si están utilizando este método para aplicaciones importantes. Los usuarios precavidos sólo confiarían en certificados para los cuales pudieran asegurar su autenticidad. Para lograr esto, generalmente se recurre a dos métodos:

1. Contacto social: En esta situación, los usuarios de PGP se encuentran físicamente e intercambian certificados por medio de un disco u otro medio de almacenamiento. Si los dos usuarios se conocen personalmente y se tienen confianza mutua, podrían detectar impostores con mayor facilidad. Este es el método más efectivo para el intercambio de llaves.

2. "Fingerprinting" o huella digital: Muchos usuarios de PGP optan por compartir una "huella digital" de su llave pública, la cual consiste en un valor hash de 128 bits calculado por medio del algoritmo MD5. Esto resulta mucho más compacto y fácil de manejar que la llave entera, la cual contiene varios cientos de bits. El valor hash podría ser incluido en los mensajes enviados por su propietario, en su página Web personal o incluso en sus tarjetas de presentación. Esto resulta práctico si se desea evitar una relación de confianza transitiva. Por ejemplo, si un usuario no confía plenamente en las firmas digitales del usuario B, podría validar el certificado de C al calcular la huella digital del mismo y compararla contra la que el usuario C ha publicado en su página personal o algún otro medio con anterioridad. El hash generado por el algoritmo MD5 proporcionado un grado razonable de seguridad en este tipo de aplicación de tal forma que muchos usuarios optan por confiar en certificados cuyas huellas digitales sean coincidentes.

Por otro lado, los usuarios menos cautelosos podrían aceptar y firmar certificados recibidos en mensajes de correo electrónico, intercambiando confiabilidad por conveniencia. Esta situación probablemente no tendría mayor impacto en situaciones en las que los usuarios se desenvuelven en un ambiente de menor riesgo o el impacto de una posible pérdida o difusión de información confidencial es bajo. Por ejemplo, el falsificar un mensaje de correo electrónico proveniente de un usuario o sitio aleatorio en Internet representaría un reducido beneficio práctico para un intruso.

3.11 Desventajas comunes de la Infraestructura de Llave Pública (PKI)

La mayoría de las desventajas asociadas con el uso de certificados digitales y soluciones de PKI surgen del hecho de que no existe una implementación estándar, por lo que se generan problemas de compatibilidad. Otra de las desventajas asociadas consiste en que regularmente este tipo de sistemas no presentan una interfaz amigable al usuario final.

A continuación se enumeran los problemas más comunes asociados con sistemas de PKI:

- No todas las aplicaciones aceptan los mismos certificados digitales.
- No todas las aplicaciones han sido creadas para aceptar ningún tipo de certificado digital.
- Los certificados digitales pueden ser falsificados.
- Los usuarios generalmente desconocen la forma de distinguir un certificado confiable de uno falsificado.
- La interfaz de usuario para certificados digitales es pobre.
- Si la seguridad de una Autoridad Certificadora en particular es pobre, se dificultaría confiar en la autenticidad de los certificados digitales.
- Podría dificultarse la asociación de un certificado con un individuo en particular, especialmente si el usuario desconoce la huella digital del mismo o no la tiene a la mano.
- Cuando un certificado digital expira, no todos los servidores de llaves son actualizados con esta información.
- Si el usuario pierde la contraseña de su certificado digital o dispositivo de almacenamiento de llaves, resulta muy difícil o imposible usar las llaves de nuevo o removerlas de los servidores de llaves.
- Si el usuario pierde sus llaves debido a una falla en el disco duro de su equipo o pierde el dispositivo de almacenamiento, no siempre se pueden recuperar esas llaves por algún otro método.
- Implementar y mantener un sistema de PKI requiere de intensa labor y personal altamente capacitado en el tema.

Si bien existen diversos proveedores que ofrecen soluciones completas de PKI, éstos no garantizan el funcionamiento de su solución al interactuar con las soluciones de otros proveedores. Por ejemplo, si una compañía contara con la solución de PKI del vendedor A y su mayor cliente utiliza la solución del vendedor B, no hay garantía de que los sistemas usados por ambas compañías puedan intercambiar información de manera completamente segura.

Sin embargo, muchos de los problemas mencionados anteriormente también surgen de sistemas de PKI configurados inapropiadamente. Cuando un sistema de PKI tiene una configuración 100% correcta, la mayoría de los problemas son resueltos. Sin embargo, debido a que esta configuración debe hacer de forma manual y personalizada, se requiere de profesionales para realizarla. Posteriormente, se requiere de capacitación apropiada a los usuarios para de esta forma reducir aun más algunos de los problemas asociados a los sistemas de PKI.

CAPÍTULO 4

PROTOCOLOS DE SEGURIDAD PARA TRANSACCIONES ELECTRÓNICAS E INTERCAMBIO DE INFORMACIÓN

4.1 S-HTTP

El protocolo S-HTTP fue diseñado por Enterprise Integration Technologies para CommerceNet, un consorcio dedicado a la promoción del uso del comercio electrónico a través de Internet. En realidad, se trata de una extensión de seguridad para el protocolo http que actúa a nivel de aplicación, protegiendo de forma independiente cada transacción realizada. Proporciona servicios de autentificación, confidencialidad e integridad de los datos.

Los algoritmos soportados por este protocolo son los siguientes:

- Funciones hash: MD2, MD5, SHA-1
- Algoritmos simétricos: DES con cifrado por bloques (CBC), 3DES con CBC, IDEA, RC2, RC4.
- Algoritmos asimétricos: RSA, DSS.

La gestión de llaves puede realizarse de varias formas:

- A través de RSA, según los formatos de PEM o el estándar PKCS #7.
- De forma manual, previo al establecimiento de la comunicación.
- Utilizando los "tickets" de Kerberos.

La forma de indicar que se utiliza S-HTTP en una sesión de un navegador de Internet es iniciando la línea de la URL de la forma *shttp://*.

4.2 Secure Sockets Layer (SSL)

Regularmente, las conexiones que se realizan dentro de Internet entre cliente y servidor sin emplear la encriptación, son relativamente anónimas. El empleo de *cookies*[1] permitiría a un servidor mantener un registro de las visitas realizadas por una computadora en particular. Sin embargo, la misma no

[1] Pequeño archivo de texto que un sitio web coloca en el disco duro de una computadora que lo visita. Al mismo tiempo, recoge información sobre el usuario. Agiliza la navegación en el sitio. Su uso es controvertido, porque pone en riesgo la privacidad de los usuarios. (http://www.sitiosargentina.com.ar/Help/diccionario%20tecnico.htm#c)

almacena información personal del usuario de forma directa. Únicamente se guarda información propia del navegador, fecha y hora de las peticiones y algunos otros datos. Debido a que en estos casos no se realiza la transferencia de información personal de un usuario entre una computadora y un servidor Web, se podría pensar que sería difícil para un tercero obtener esa información. Sin embargo, existe la posibilidad de que se produzcan engaños y actividades ilegales en Internet que pudiesen provocar que un usuario proporcionara información personal inadvertidamente a personas no autorizadas. Estos engaños podrían presentarse en la forma de sitios Web que emulen a otros sitios genuinos, especialmente aquellos que se dediquen al comercio electrónico y que a simple vista pudieran parecer genuinos también ya que son réplicas de sitios auténticos. Usualmente estos sitios engañosos intentan obtener del usuario su identificador (ID) y su contraseña. Esta situación se ha convertido en un problema crítico en Internet en los últimos años, de tal modo que muchas de las grandes compañías que emplean el comercio electrónico han creado departamentos especializados para resolver casos relativos a fraude.

Asimismo, existen otros tipos de fraude, tales como la intervención de conexiones en red en las cuales en determinado momento de la transmisión de datos un intruso podría sustraer información, lo cual es conocido como *ataque de hombre en medio*. Este tipo de ataques son más difíciles de detectar debido a que un usuario común de Internet no podría detectar alguna actividad sospechosa a simple vista. Por otro lado, la persona que interviene la conexión en estos casos espera obtener información personal importante para el usuario, tal como su ID y contraseña. Este tipo de ataque normalmente es posible en las conexiones no encriptadas.

El protocolo SSL (Secure Sockets Layer) fue creado con el propósito de mitigar los riesgos de que se produzcan los ataques mencionados anteriormente, mediante el uso e intercambio de llaves de encriptación durante las conexiones en red. Las primeras implementaciones de SSL estaban limitadas a llaves de encriptación de longitud de 40 bits, sin embargo la mayoría de ellas actualmente son capaces de manejar llaves de encriptación de 128 bits.

Existe una serie de transacciones y acuerdos que se establecen entre una computadora y un servidor Web durante una conexión que emplea el protocolo SSL, tal como se muestra en la siguiente figura:

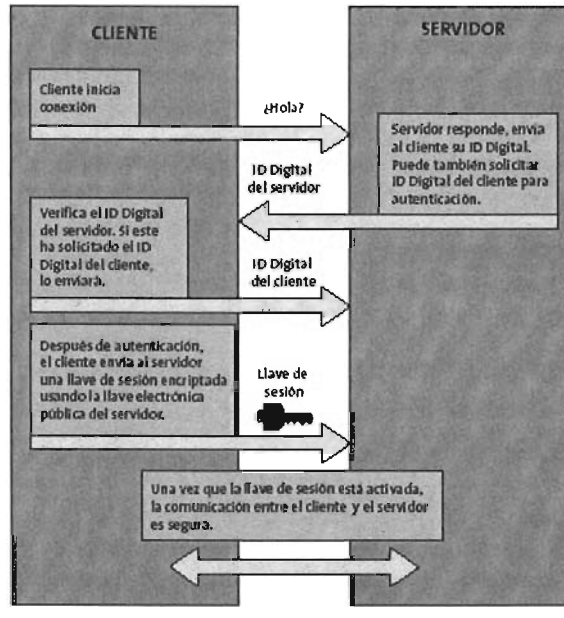


Fig. 4.1: Proceso SSL

1. La computadora inicia la conexión al enviar una petición de conexión segura a un sitio que ofrezca este servicio.

2. El servidor Web responde a la petición enviando su certificado digital con su respectiva llave pública.

3. Cuando la computadora recibe esta información, el navegador revisa la lista de certificados que tiene instalados e información relativa a las autoridades certificadoras. Posteriormente, compara el certificado digital que recibió del servidor y lo compara con la lista antes mencionada para determinar si el certificado es confiable o no.

4. Si el navegador determina que el certificado digital es confiable, calcula una llave de sesión y la envía al servidor Web. Esta llave es encriptada junto con la llave pública del servidor Web.

5. Después de que el servidor Web recibe la llave de sesión, se inicia la encriptación de las comunicaciones entre ambos equipos. Esto se tornará visible en el navegador en el momento en que la dirección Web (URL) que aparece en el mismo se convierta de *http://* a *https://*. Asimismo, dependiendo del navegador que se emplee, regularmente se podrá apreciar en la barra de estado del

navegador un ícono pequeño que represente una llave o un candado, con lo cual indica que la conexión que se está realizando es segura.

No todas las compañías que emiten certificados digitales se encontrarán enlistadas en la sección de certificados propia del navegador. Esto es debido a las relaciones de negocios establecidas entre los proveedores de navegadores de Internet y algunas autoridades certificadoras, por lo que los certificados incluidos en la lista del navegador no deben ser necesariamente más confiables que los que no se encontraran en la misma. Sin embargo, debido a las relaciones de negocios establecidas entre algunos proveedores de navegadores y compañías certificadoras, existe una mayor posibilidad de obtener soporte en caso de que surgieran problemas con los certificados.

Existen distintos propósitos descritos en la lista de certificados de cada navegador. Por lo regular, aquellos que se emplean dentro del proceso de validación de certificados intercambiados en una conexión mediante SSL, son aquellos conocidos como "Certificados Raíz", los cuales son empleados para la autentificación de servidores. La fecha de expiración de un certificado raíz es de suma importancia, ya que una vez expirado no sería posible depositar confianza en los certificados provenientes de un sitio Web en particular.

Aún cuando el protocolo SSL es empleado para proteger y encriptar las transacciones realizadas en Internet, existe otro protocolo que se utiliza como complemento, denominado TLS (Transport Layer Secùrity, o Seguridad en la Capa de Transporte), el cual se emplea en la autentificación entre navegadores y servidores Web para negociar el algoritmo de encriptación a utilizar y las llaves a emplear antes de iniciar la transmisión de datos. El protocolo TLS hace uso de algoritmos de la Criptografía de Llave Pública, tales como RSA o DSS para autentificar equipos y negociar llaves privadas, y algoritmos de la Criptografía de Llave Privada, tales como DES o RC4 para encriptar los datos, tales como números de tarjetas de crédito, previo a la transmisión de los datos a través de la red. Cualquier transmisión de mensajes incluye un código de autentificación de mensaje (MAC) creado por medio de una función hash tal como MD5 o SHA para prevenir la falsificación de información.

El protocolo TLS se compone de los siguientes elementos, los cuales funcionan como complemento del protocolo SSL:

- Protocolo de registro: Asegura que la conexión entre un equipo y el servidor Web sea privada mediante el uso de llaves simétricas.
- Protocolo de Saludo o "Handshake": Negocia el algoritmo de encriptación a utilizarse antes de que se continúe la comunicación entre ambos equipos.

Además de los elementos mencionados anteriormente, TLS también constituye un complemento ventajoso para SSL, debido a que es un estándar abierto, gobernado por la IETF[2].

La infraestructura fundamental requerida para implementar un sitio Web que haga uso del protocolo SSL para transacciones electrónicas seguras, consiste de una red local de cómputo con una conexión a Internet de alta velocidad, así como un nombre de dominio registrado y una dirección IP estática. Los elementos básicos que componen a la red antes mencionada deben ser los siguientes:

- ✓ **Servidor:** El cual se empleará como servidor Web para hospedar la página del negocio. El servidor podría ser incluso un equipo de escritorio si no se espera tener mucho tráfico en el mismo.

- ✓ **Servidor Redundante:** Este servidor tiene como propósito funcionar como respaldo en caso de emergencia. Por ejemplo, si el sitio Web sufriera ataques por parte de hackers, fallas en disco u otras eventualidades. El equipo debe ser configurado de tal forma que sea fácil cambiar la operatividad de uno a otro con la capacidad suficiente para que pueda mantener la carga de tráfico durante el tiempo que el servidor principal permanezca fuera de servicio.

- ✓ **Firewall:** Es un dispositivo ya sea de hardware y/o software, desarrollado de forma casera usando herramientas gratuitas o adquirido de un proveedor. Se emplea para proteger una red interna o local (LAN). Sin embargo, el servidor Web no debe encontrarse detrás del firewall, pues se alentaría el tráfico hacia el mismo considerablemente. El dispositivo de firewall se emplearía para controlar el tráfico entre el servidor Web y la red local a la que pertenece el resto de los equipos de la organización. Asimismo, el sistema operativo del servidor debe tener las actualizaciones de seguridad debidamente instaladas. Por otro la computadora que hospede el software del firewall debe ser empleada únicamente para dicho fin y ningún otro.

- ✓ **Servidor de llaves/certificados digitales:** Se emplea para generar las llaves y certificados digitales que requiere el protocolo SSL para operar. Este servidor debe encontrarse en la red interna, detrás del firewall. Si bien muchos servidores Web son capaces de llevar a cabo las funciones de generación e intercambio de llaves, un servidor separado resulta conveniente cuando se espera un mayor tráfico desde y hacia el servidor Web. Asimismo, el contar con un servidor de llaves separado detrás de un firewall, proporciona de una capa adicional de seguridad debido a que

[2] Internet Engineering Task Force.- Organización no gubernamental dedicada a crear estándares de desarrollo para soluciones en Internet.

dificulta a un atacante introducirse al sistema y obtener las llaves de encriptación.

- ✓ **Servidor de bases de datos:** Se emplea para almacenar los datos del servidor Web: imágenes, scripts de Java, inventarios, precios y órdenes, información de clientes y de pagos, etc. Fundamentalmente, este servidor funciona como aquél que lleva a cabo todas las operaciones de oficina que normalmente se llevarían en una tienda real. Esta información nunca debería de estar en el mismo servidor que el servidor Web, ya que es sensitiva para el negocio, por lo que el servidor de bases de datos debe encontrarse detrás del firewall y protegido por controles de acceso lógicos y medidas adicionales para reforzar al máximo la seguridad.

- ✓ **Servidor o dispositivo de respaldo**: El servidor u otros dispositivos de respaldo, tales como cintas magnéticas o discos externos, podrían emplearse para respaldar los datos contenidos en las bases de datos, por lo que deberían ser actualizados con frecuencia para poder tener la información más actualizada en caso de que se requiera realizar una restauración de datos ante alguna contingencia.

- ✓ **Tarjetas criptográficas de aceleración**: Este es un elemento opcional y que puede volverse necesario únicamente cuando se reciban más de 300 a 500 peticiones de conexión remotas por segundo en el servidor Web. Debido a que los protocolos SSL y TLS realizan funciones criptográficas, esto implica que gran parte de la potencia de procesamiento de los equipos sea empleado en generar llaves criptográficas. Este procesamiento podría disminuir significativamente la velocidad de respuesta del servidor hacia el usuario cuando éste intente obtener una conexión segura. Al instalar un acelerador criptográfico en el servidor Web, dicho dispositivo absorbería la carga de procesamiento del servidor, por lo que el sitio Web podría responder más rápidamente.

Los elementos mencionados anteriormente constituyen sólo la infraestructura mínima requerida para que un sitio de comercio electrónico funcione. Sin embargo, si se tiene planeado contar con un sitio más complejo y con varias combinaciones de servicios y opciones, se requerirá mayor cantidad de equipos y personal especializado encargado de su operación y supervisión. Por otro lado, si no se espera tener mucho tráfico en el sitio o no se cuenta con un departamento con recursos humanos especializados, existen soluciones comerciales de SSL y proveedores de servidores de páginas Web que cuentan con la infraestructura necesaria para efectuar transacciones seguras por medio de SSL, mediante el pago de una cuota mensual por sus servicios.

4.3 XML (eXtensible Markup Language)

XML es un lenguaje posterior a HTML, empleado también para el desarrollo de páginas Web. XML añade una capa más de seguridad a las páginas mediante el uso de nuevas etiquetas que definen no sólo el contenido a encriptar dentro de la página sino que también pueden manejar distintos niveles de encriptación dentro de la misma página o documento. Para lograr este propósito, XML es empleado en conjunto con otro lenguaje, llamado SSML (Security Sheet Markup Language) el cual consiste en una serie de políticas de seguridad que deben seguir las etiquetas de XML. Mediante el uso de las etiquetas <secure> y </secure>, XML indica al navegador encriptar todo lo que sea colocado entre esas dos etiquetas. Asimismo, los elementos encriptados no se limitan a texto, ya que también las imágenes y archivos de sonido son encriptados automáticamente. La llave de encriptación es enviada de manera separada de la página Web, de tal forma que el contenido cifrado y la llave nunca aparezcan en el mismo documento. El algoritmo de encriptación es decidido por una regla creada mediante SSML y comienza a funcionar en cuanto la página se carga por completo.

SSML es empleado para definir las políticas de seguridad y las reglas de implementación, empleando diferentes niveles de encriptación dentro de un mismo documento. Por ejemplo, si una página contiene un formulario en el cual el usuario debe ingresar su nombre, dirección, número de tarjeta de crédito y otra información, se podría encriptar el nombre y dirección empleando un algoritmo y el número de tarjeta de crédito empleando un algoritmo diferente. La seguridad se duplica en este caso debido a que si en determinado momento alguna llave fuese descubierta, sólo una porción de la información sería revelada. Un intruso no podría deducir el otro algoritmo o la llave empleada para encriptar el resto de la página a partir de la primera llave, por lo que no podría explotar de manera efectiva la información que pudiese obtener.

XML es un estándar ya definido; sin embargo, su uso aún no es muy generalizado. Esto es debido a que aún existen algunas versiones más antiguas de navegadores que no proveen de la funcionalidad del código XML. Asimismo, existen problemas de compatibilidad con HTML, ya que una página Web no puede estar codificada en ambos lenguajes de manera simultánea. Por lo tanto, si se desea utilizar XML para desarrollar páginas Web más seguras, se debe primero codificar la página de tal forma que primero detecte si el navegador puede o no aceptar XML, y en caso de que no lo acepte, se pueda desplegar una página en HTML en su lugar.

XML constituye un avance significativo en el fortalecimiento de la seguridad y capacidades de encriptación de las páginas Web comunes y especialmente de aquellas diseñadas para el comercio electrónico.

4.4 Redes Privadas Virtuales (VPN)

El tráfico de datos entre cualquier equipo de cómputo conectado a Internet y un servidor, en cualquier momento, es susceptible a ser interceptado, saboteado, capturado y almacenado en un archivo por terceros. Un método para "ocultar" las conexiones de forma que éstas no puedan ser detectadas ni interceptadas por intrusos, es mediante la utilización de una Red Privada Virtual (VPN, por sus siglas en inglés).

Una VPN divide los datos a transmitir en fragmentos llamados "paquetes" y después los encripta con el objeto de que sólo los destinatarios designados puedan leerlos. Cada paquete contiene encabezados de información acerca del tamaño y tipo de datos, e información para verificar la integridad de los mismos. Esto constituye una medida de seguridad para garantizar que los datos no han sido cambiados durante su transmisión. Una vez que los datos han sido divididos y encriptados, la VPN establece un canal para enviar los datos. Este canal es llamado "túnel IP", lo cual es la característica principal de una VPN. Aún cuando los datos se envían a través de una red pública como la Internet, este canal no puede ser detectado por personas no autorizadas que estuvieran intentando intervenir en las comunicaciones por medio de "sniffers", los cuales son programas que interceptan paquetes de datos que viajan a través de la red, ya que las VPN's pueden ocultar la presencia de conexiones de red. Lo anterior es posible mediante el uso de software especial, protocolos y comandos que se generan por el equipo empleado para establecer la VPN. Actualmente, la mayoría de los dispositivos de comunicaciones tales como switches o ruteadores son capaces de crear una VPN. Asimismo, los firewalls más recientes son capaces de distinguir el tráfico que entra y sale de una red local para determinar si se trata de tráfico encriptado por VPN o sin encriptación.

Generalmente, una VPN se utiliza para 3 distintos tipos de seguridad:

- **Resguardar información confidencial de una compañía.**

Para lograr esto, se requiere de una intranet, la cual es una infraestructura común de red a través de diversas ubicaciones físicas conectadas a un centro de procesamiento de datos. Con este tipo de VPN, se pueden segregar redes departamentales sin necesidad de instalar cableado adicional o infraestructura nueva. Esto es efectivo cuando se asignan permisos de acceso a información sensitiva basándose en las distintas áreas y entornos de red separados, como podrían ser entorno de desarrollo y entorno de producción. La VPN es transparente a los usuarios y regularmente no requiere de software especial instalado en las estaciones de trabajo para que los usuarios puedan acceder a la VPN.

- **Compartir información con clientes y proveedores**

Para lograr este objetivo, sería necesario configurar una red externa. En este caso, la VPN utiliza el Internet como el principal medio de transmisión de datos y se usa generalmente cuando una compañía tiene un gran número de usuarios y oficinas en distintas ubicaciones dispersas. Este tipo de VPN permite a clientes, proveedores y sucursales acceder a recursos corporativos a través de varias arquitecturas de red. La gran masa de usuarios de Internet no es capaz de detectar la VPN, ni son capaces de firmarse en ella inadvertidamente, debido a los requisitos de autentificación de la VPN. Para reforzar la seguridad aún más, la compañía también podría implementar IPSEC para asegurar la máxima compatibilidad y seguridad.

- **Crear túneles ocultos**

Los túneles ocultos tienen por objeto permitir a los usuarios remotos a firmarse de manera segura en la red corporativa desde salas de juntas, instalaciones del cliente u otras locaciones. En estos casos, el equipo de cómputo portátil necesita tener instalado software de cliente de VPN para iniciar la conexión y autentificar al usuario. Nuevamente, el tráfico fluye a través de Internet hacia la red corporativa, pero los usuarios generales de Internet no pueden ver el tráfico.

En la mayoría de los casos, el proceso de encriptación tiende a incrementar la carga de trabajo en los dispositivos de red y puede afectar al desempeño en general de los mismos. Sin embargo, la mayor parte de la encriptación realizada por medio de VPN es procesada rápidamente y la mayoría de los usuarios no llega a apreciar una disminución considerable en la velocidad de su conexión. La mayoría de los problemas de desempeño son resultado de conexiones de Internet inconsistentes más que consecuencia del proceso de encriptación, a menos que se haga uso de terminales virtuales o aplicaciones que corran desde un servidor central en lugar de una estación de trabajo.

Como sucede regularmente con otro tipo de soluciones informáticas, existen diferentes técnicas y estándares de encriptación por medio de VPN, de las cuales han sobresalido algunos protocolos que se han logrado colocar como estándares dentro de las soluciones de VPN, como los que a continuación se mencionan.

4.4.1 PPP y PPTP

El Protocolo de Tunel de Punto a Punto (PPTP, ó *Point to Point Tunnelling Protocol*, por sus siglas en inglés) tuvo como antecesor inmediato al

Protocolo de Punto a Punto (PPP). Este protocolo aún sigue utilizándose en algunos lugares, principalmente para permitir a usuarios que utilizan modem poder conectarse a Internet; sin embargo, no es empleado en VPN's.

El protocolo PPTP fue desarrollado por Microsoft y U.S. Robotics, usando PPP como base. La porción de autentificación de PPTP utiliza el protocolo de autentificación de Microsoft, *Challenge Handshake Authentication Protocol* (CHAP) versión 2 en adelante. PPTP no encripta el tráfico por default debido a que su función principal es crear un túnel virtual en el cual se pueda transportar datos y autentificar usuarios. Sin embargo, también puede ser configurado para encriptar el túnel así como para utilizar el algoritmo RC4. La configuración por default para RC4 es de 40 bits, lo cual podría no proporcionar suficiente seguridad en las llaves; sin embargo se puede configurar a 128 bits de encriptación. Debido a que RC4 es un algoritmo simétrico, tanto el usuario como el dispositivo de VPN intercambian una misma llave privada. La llave es desechada tan pronto como la sesión finalice.

Las VPN's que emplean PPTP son utilizadas normalmente en conexiones internas de cliente al switch o dispositivo de enlace principal de la red o en conexiones entre redes distintas. Normalmente se emplea óptimamente en redes con clientes cuyo sistema operativo sea Windows. Si se cuenta con distintos sistemas operativos dentro de la red, resulta más conveniente alguno de los otros protocolos de VPN que son conocidos como estándares.

4.4.2 L2TP

El protocolo *Layer 2 Tunneling Protocol* (L2TP) es una combinación de dos protocolos, el *Layer 2 Forwarding* (L2F), el cual fue desarrollado por Cisco y el PPTP. Los desarrolladores de ambos protocolos colaboraron conjuntamente, obteniendo como resultado un producto que contuviera las mejores características de ambos: el protocolo L2TP.

L2TP tiene la capacidad de autentificar usuarios, crear un túnel virtual y encriptar y comprimir los datos que viajan a través de dicho túnel. Sin embargo, cabe destacar que el túnel virtual no está encriptado por sí mismo. En el caso de que se estén utilizando productos de Microsoft para usar junto con el protocolo L2TP, el protocolo IPSEC debe ser incorporado para asegurar la encriptación de los datos que viajan a través del túnel.

Una de las principales ventajas de L2TP es que puede ser utilizado en redes en las que se tiene más de un protocolo de red implementado. Por lo tanto, este protocolo puede ser implementado en redes IP, IPX, NetBEUI, etc.

4.4.3 IPSEC

El protocolo IPSEC es una mejora al protocolo TCP/IP que provee de características de seguridad que TCP/IP carece, debido a que este protocolo no fue diseñado con un concepto de seguridad en mente, por lo que posee de una gran variedad de vulnerabilidades de seguridad. Fue desarrollado para el nuevo estándar IPv6 y después fue portado a IPv4.

IPSEC requiere del uso de llaves o certificados digitales para poder operar. En el caso de que se utilicen certificados digitales, es necesario implementar una Infraestructura de Llave Pública (PKI) para realizar la creación e intercambio de certificados digitales. Por otro lado, si se emplean llaves, éstas deben ser simétricas (privadas) por lo que se debe de implementar un mecanismo adicional para asegurar su intercambio. Cuando se inicia una sesión de red empleando IPSEC, ambos equipos de cómputo se autentifican mutuamente y el usuario de la computadora de la que se origina la transmisión de datos ingresa la información relativa a la llave de encriptación. Una vez que esta información ha sido transmitida correctamente, los datos que se transfieran entre ambos sistemas estarán encriptados.

Sólo los sistemas compatibles con el protocolo IPSEC pueden emplearlo. Adicionalmente, todos los sistemas que se comuniquen entre sí empleando este protocolo deben emplear una llave de encriptación común o certificados digitales, así como tener configuradas políticas de seguridad similares dentro del protocolo.

La combinación de IPSEC con los protocolos de túneles virtuales de VPN provee la mejor solución de seguridad puesto que se consigue que la transmisión de datos sea encriptada y al mismo tiempo oculta del resto del tráfico de la red o Internet por medio del túnel.

Dentro de IPSEC se distinguen los siguientes componentes:

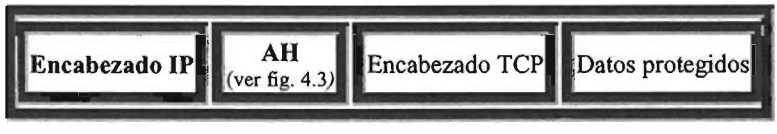
- Dos protocolos de seguridad: IP AH (*Authentication Header* o Encabezado de Autentificación) e IP ESP (Encapsulating Security Payload, o Carga de Seguridad Encapsulada) que proporcionan mecanismos de seguridad para proteger tráfico IP.
- Un protocolo de gestión de llaves Internet Key Exchange (IKE) que permite a dos nodos negociar las llaves y todos los parámetros necesarios para establecer una conexión AH o ESP.

Existen dos modalidades independientes empleadas por IPSEC para proteger los datos: **"modo de transporte"** y **"modo de túnel"**. El modo de transporte hace uso del AH y se emplea típicamente cuando las comunicaciones se realizan entre servidores que emplean IPSEC. El modo de túnel hace uso del ESP y es más frecuentemente utilizado cuando se tienen computadoras de

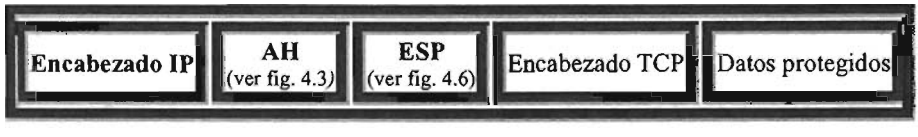
escritorio (PC's) que se comunican con servidores IPSEC. El modo de túnel encripta el encabezado y el resto de cada paquete de información, mientras que el modo de transporte no encripta el encabezado. Ambos modos hacen uso de un valor numérico llamado SPI (*Security Parameter Index,* o Índice de Parámetro de Seguridad). Cada vez que un cliente receptor procesa un encabezado, utiliza el SPI para identificar las llaves criptográficas.

En el modo **transporte**, el contenido transportado dentro del datagrama AH o ESP son datos de la capa de transporte (por ejemplo, datos TCP o UDP). Por tanto, la cabecera IPSEC se inserta inmediatamente a continuación de la cabecera IP y antes de los datos de los niveles superiores que se desean proteger. En otras palabras, un AH añadido al paquete cubrirá el resumen criptográfico del encabezado TCP y algunos campos de la cabecera IP extremo a extremo, y un encabezado ESP cubrirá el cifrado del encabezado TCP y los datos, pero no la cabecera IP extremo a extremo. El modo transporte tiene la ventaja de que asegura la comunicación extremo a extremo, pero requiere que ambos extremos entiendan el protocolo IPSEC.

Un ejemplo de un paquete AH en modo transporte es:



Debido a que un encabezado ESP no puede autenticar el encabezado IP exterior, es útil combinar un encabezado AH y un ESP para obtener lo siguiente:



En el modo **túnel**, el contenido del datagrama AH o ESP es un datagrama IP completo, incluido el encabezado IP original. Así, se toma un datagrama IP al cual se añade inicialmente un encabezado AH y/o ESP, posteriormente se añade un nuevo encabezado IP que es el que se utiliza para encaminar los paquetes a través de la red. El modo túnel se usa normalmente cuando el destino final de los datos no coincide con el dispositivo que realiza las funciones IPSEC.

Un ejemplo de un paquete AH en modo túnel es:

El modo túnel es empleado principalmente en el establecimiento de VPN's. Asimismo, también es empleado por los gateways IPSEC, con objeto de identificar la red que protegen bajo una misma dirección IP y centralizar de este modo el procesado del tráfico IPSEC en un equipo. El modo túnel también es útil, cuando se utiliza junto con ESP, para ocultar la identidad de los nodos que se están comunicando.

IPSEC puede ser implementado ya sea en un servidor o bien en un equipo dedicado, tal como un ruteador o un *firewall,* el cual al realizar estas funciones se denomina *gateway*[3] IPSEC. La figura 4.2 muestra los dos modos de funcionamiento del protocolo IPSEC, donde:

1. En la figura 4.2.1 se representan dos equipos que entienden IPSEC y que se comunican de forma segura. Esta comunicación se realiza en modo transporte, por tanto la información que se protege es únicamente el protocolo TCP o UDP, así como los datos de aplicación.

2. En la figura 4.2.2 se muestran dos equipos que utilizan dos *gateways* IPSEC para conectarse y, por lo tanto, emplean una implementación en modo túnel. Se puede ver que la comunicación se realiza a través de una red de datos pública, entre una PC situado en una red local con otra PC situado en una red local remota, de modo que entre los *gateways* IPSEC se establece un túnel a través del cual viajan protegidas las comunicaciones entre ambas redes locales. Sin embargo ambos PC's envían y reciben el tráfico en claro (no cifrados), como si estuviesen situados en la misma red local. Este esquema tiene la ventaja de que los nodos situados en redes separadas pueden comunicarse de forma segura y transparente, concentrándose, al mismo tiempo, las funciones de seguridad en un único punto, facilitando así las labores de administración.

[3] Dispositivo de comunicación entre dos o más redes locales (LANs) y remotas, usualmente capaz de convertir distintos protocolos, actuando de traductor para permitir la comunicación. Como término genérico, es utilizado para denominar a todo instrumento capaz de convertir o transformar datos que circulan entre dos medios o tecnologías.
(http://www.sitiosespana.com/paginas/diccionario_informatica/g.htm)

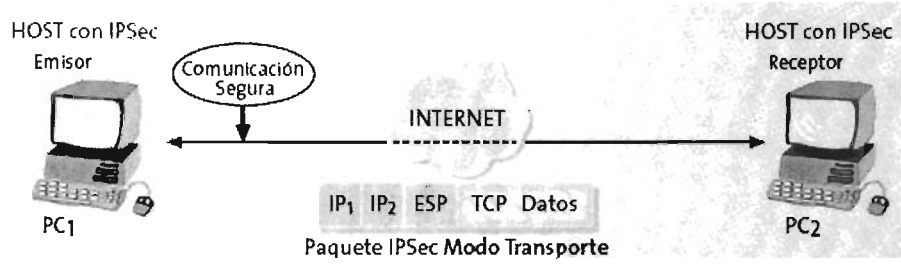


Fig. 4.2.1: Modo Transporte

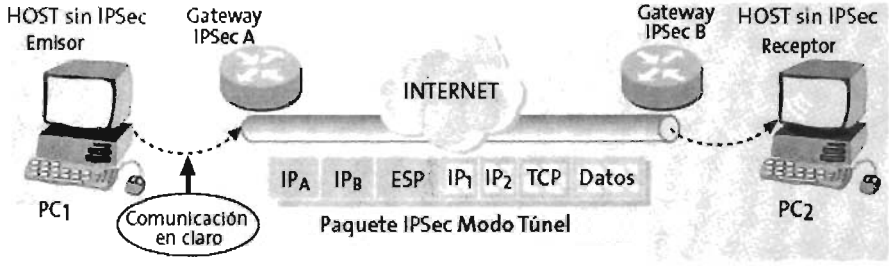


Fig. 4.2.2: Modo Túnel

Para poder establecer comunicación, cada par de clientes y/o servidores que emplean IPSEC deben establecer una **Asociación de Seguridad** mutua. Ésta se define como un canal de comunicación unidireccional que conecta dos nodos, a través del cual fluyen los paquetes o datagramas protegidos mediante mecanismos criptográficos acordados previamente. Al identificar únicamente un canal unidireccional, una conexión IPSEC se compone de dos AS's, una por cada sentido de la comunicación. La asociación de seguridad establece la forma en que se establece la protección dentro de IPSEC, esto es, el tipo de protección a aplicar, el algoritmo de encriptación y las llaves que van a ser utilizadas. Es necesario que ambos nodos estén de acuerdo tanto en los algoritmos criptográficos a emplear como en los parámetros de control. Esta operación puede realizarse mediante una configuración manual, o mediante algún protocolo de control que se encargue de la negociación automática de los parámetros necesarios, tal como el protocolo IKE, el cual se detallará más adelante. La asociación de seguridad que aplica a un determinado paquete de información es determinada por la dirección IP de destino y el SPI en el encabezado del paquete. El software que emplee IPSEC deberá mantener la siguiente información para cada SPI:

1. Especificación de los métodos de encriptación a ser utilizados por el SPI.

Los formatos del AH y ESP son muy generales y deben ser adecuados a los métodos a utilizar de encriptación y cálculo de sumas de comprobación.

2. Las llaves a ser empleadas por los algoritmos criptográficos al procesar el SPI.

3. Los clientes y/o servidores asociados dentro de las comunicaciones de datos.

Esta información permite despejar ambigüedades en asociaciones de seguridad en caso de que dos o más equipos asignen una SPI idéntica.

Cuando un equipo receptor procesa el primer encabezado IPSEC de un paquete, el SPI es utilizado para identificar la Asociación de Seguridad apropiada. Enseguida, aplica el algoritmo criptográfico definido en la asociación utilizando la llave privada acordada. Debido a que es probable que exista un encabezado AH y un encabezado ESP por separado, el proceso se repite en el siguiente encabezado IPSEC, utilizando el SPI para localizar el material criptográfico adecuado. Si el SPI no existe o el paquete es inválido una vez procesado, se descarta sin emitir ninguna alerta al equipo remitente.

Cada Asociación de Seguridad viene definida por un único flujo unidireccional de datos, y por regla general, desde un único punto hasta otro, cubriendo el tráfico que se distingue por algún "selector único". Todo el flujo de tráfico sobre una única Asociación de Seguridad se trata del mismo modo. Algo del tráfico puede estar sujeto a varias Asociaciones de Seguridad, cada uno de las cuales aplica algún tipo de transformación criptográfica. Un grupo de Asociaciones de Seguridad se conoce como un "Haz" (más comúnmente conocido en inglés, como "SA Bundle"). Los paquetes entrantes se pueden asignar a una Asociación de Seguridad particular por medio de los tres campos de definición:

- Dirección IP de destino
- Índice del Parámetro de Seguridad (SPI)
- Protocolo de seguridad

SPI se puede considerar como un "cookie" manejado por el receptor de la Asociación de Seguridad cuando se negocian los parámetros de la conexión. El protocolo de seguridad debe ser AH o ESP. Debido a que la dirección IP del receptor es parte del trío, ésta es un valor único garantizado. Se puede encontrar desde el encabezado IP exterior y el primer encabezado de seguridad (que contiene el SPI y el protocolo de seguridad).

4.4.3.1 Encabezado de Autentificación (AH)

El AH provee de información que permite verificar la integridad de los paquetes, de tal forma que se pueda detectar si la información contenida en los mismos ha sido alterada o falsificada durante su viaje a través de redes inseguras. El AH contiene una suma de comprobación criptográfica para asegurar que no ocurra lo anterior. Si el contenido del paquete fue efectivamente alterado, la suma de comprobación no coincidiría con la calculada inicialmente por el emisor. La suma de comprobación criptográfica incorpora información relativa a la llave secreta de tal forma que un atacante no podría calcular una suma alterna que coincidiera con la original.

El protocolo AH calcula la suma de comprobación criptográfica basada en la llave privada, el contenido del paquete y las partes inmutables del encabezado IP (como son las direcciones IP). Tras esto, añade el encabezado AH al paquete. El encabezado AH se muestra en la figura 4.3:

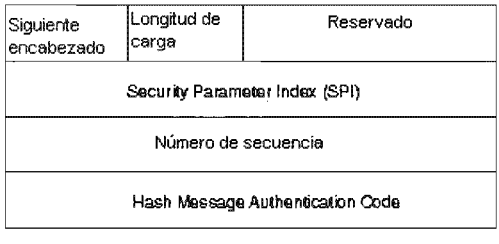


Fig. 4.3: Encabezado AH

El encabezado AH mide 24 bytes. El primer byte es el campo *Siguiente cabecera*. Este campo especifica el protocolo de la siguiente cabecera. En modo túnel se encapsula un paquete IP completo, por lo que el valor de este campo es 4. Al encapsular un paquete TCP en modo transporte, el valor correspondiente es 6. El siguiente byte especifica la longitud del contenido del paquete. Este campo está seguido de dos bytes reservados. Los siguientes 4 bytes especifican el Índice de Parámetro de Seguridad (SPI). El SPI especifica la asociación de seguridad (SA) a emplear para el desencapsulado del paquete. El Número de Secuencia de 32 bit protege frente a ataques por repetición, los cuales consisten en retransmitir una determinada cantidad de información interceptada y alterada con la intención de hacerla pasar como legítima. Finalmente, los últimos 96 bit almacenan el Código Hash de Resumen para la Autentificación de Mensaje (HMAC), es decir el resultado de la suma de comprobación criptográfica. Este HMAC protege la integridad de los paquetes ya que sólo los miembros de la comunicación que conozcan la llave privada pueden crear y comprobar HMAC's.

Como el protocolo AH protege al encabezado IP incluyendo las partes inmutables de la cabecera IP, tales como las direcciones IP, el protocolo AH no permite NAT. NAT (*Network Address Translation* - Traducción de direcciones de

red, también conocido como Enmascaramiento de direcciones) reemplaza una dirección IP del encabezado IP (normalmente la IP de origen) por una dirección IP diferente. Tras el intercambio, la HMAC ya no es válida. La extensión a IPSEC NAT-transversal implementa métodos que evitan esta restricción.

A continuación se muestra el detalle del contenido de cada uno de los encabezados en un datagrama AH:

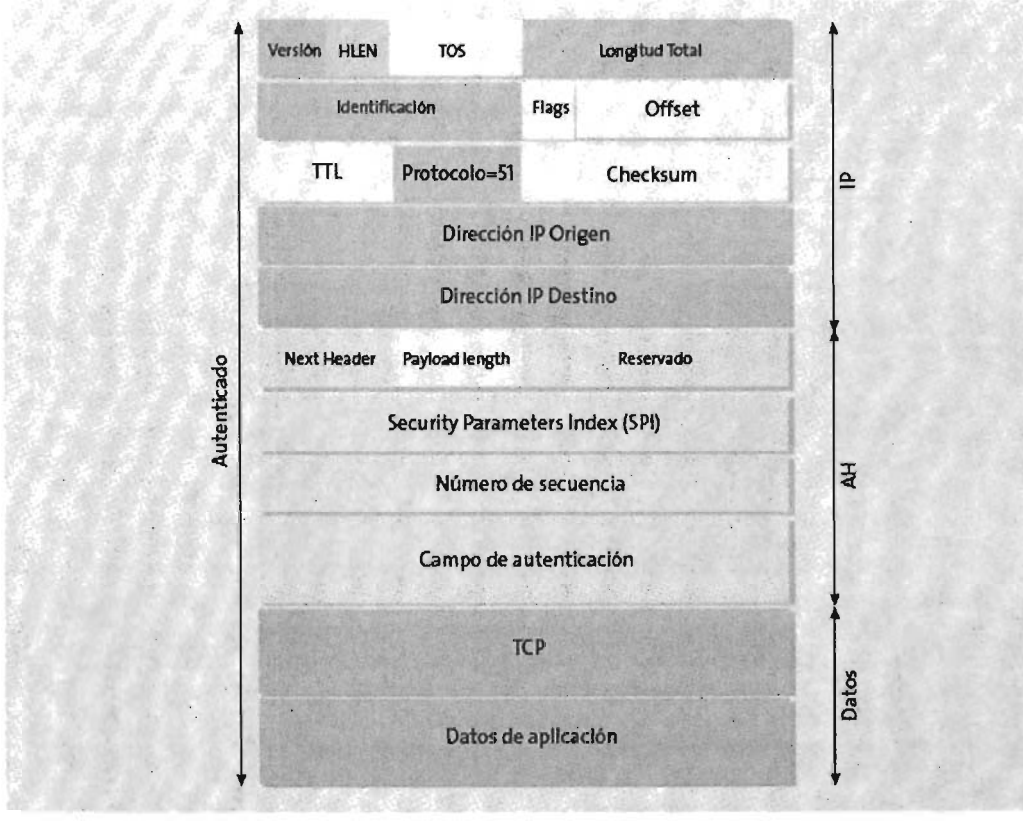


Fig. 4.4: Datagrama empleado por AH

Todas las implementaciones de IPSEC en modo transporte requieren el manejo de un AH que utilice el algoritmo hash MD5 con una llave privada, para producir un valor hash de 128 bits. El protocolo normalmente no especifica el tamaño de la llave, siempre y cuando los equipos que establecen comunicación acuerdan previamente el tamaño de la llave y el ordenamiento de los bits; sin embargo, usualmente se emplea un tamaño de llave de 128 bits. La llave

privada es anexada a los datos protegidos dos veces, una al principio de los datos y otra al final. De esta forma los datos son más resistentes a un ataque en lugar de que la llave fuese anexada una sola vez.

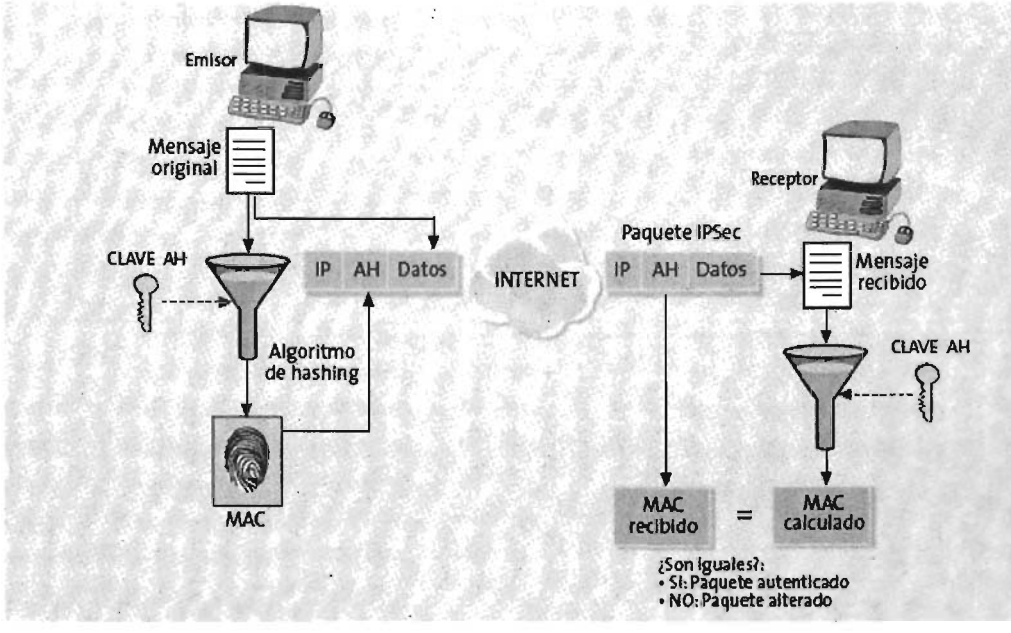


Fig. 4.5: Funcionamiento de AH

En la Figura 4.5 se muestra el modo en que funciona el protocolo AH. Por medio del algoritmo MD5, el emisor calcula la suma de comprobación (HMAC) del mensaje original sobre el encabezado IP combinado con los encabezados posteriores al AH. Al incluir el encabezado IP en este cálculo, el AH puede detectar cambios en la información del direccionamiento del paquete o intentos de eludir al propio AH. Primeramente, cada valor se define como aquel que el recipiente debería de ver al obtener el paquete. Todos aquellos campos que hubieran de cambiar en tránsito son definidos en 0. Posteriormente este encabezado IP es combinado con los encabezados posteriores al AH para calcular la suma de comprobación criptográfica, la cual se copia en uno de los campos de la cabecera AH. El paquete así construido se envía a través de la red, repitiéndose el procedimiento en el extremo receptor para obtener el cálculo de la suma y comparándola con el recibido en el paquete. Todos aquellos campos con valores impredecibles al momento de la recepción se definen en 0. La llave asociada con el SPI es usada para calcular la suma. Si los resultados no coincidieran con el valor recibido en el AH, el paquete es desechado. Si son

iguales, el receptor tiene la seguridad de que el paquete IP no ha sido modificado en tránsito y que procede efectivamente del origen esperado.

El formato del AH por sí mismo no define el tamaño o la naturaleza de la suma criptográfica. A la suma de comprobación le es asignado un formato de 32 bits. La asociación de seguridad debe ser utilizada para especificar el tipo de suma, qué tan larga es y la forma en que será calculada. El SPI también indica cuál llave privada deberá ser empleada en el cálculo de la suma.

Si analizamos con detalle el protocolo AH, podemos concluir que su seguridad reside en que el cálculo de la suma de comprobación criptográfica o HMAC es imposible sin conocer la llave privada, y que dicha llave (en la Figura 4.5, clave AH) sólo la conocen el emisor y el receptor.

4.4.3.2 Carga de Seguridad Encapsulada (ESP)

El objetivo principal del protocolo ESP (Encapsulating Security Payload) es proporcionar confidencialidad, para ello especifica el modo de cifrar los datos que se desean enviar y cómo este contenido cifrado se incluye en un datagrama IP. Adicionalmente, puede ofrecer los servicios de integridad y autentificación del origen de los datos incorporando un mecanismo similar al de AH. El protocolo ESP puede asegurar la integridad del paquete empleando una HMAC y la confidencialidad empleando cifrado. El encabezado ESP encripta el contenido de datos en el paquete a enviar de tal forma que no pueda ser extraído durante el trayecto a través de redes inseguras. El formato del ESP varía de acuerdo con el tipo y el modo de encriptación que se emplee. En todos los casos, la llave asociada con la encriptación es seleccionada mediante el SPI.

Todas las implementaciones de IPSEC en modo túnel requieren soportar el protocolo ESP utilizando al menos algún algoritmo de simétrico o de llave privada, tal como el algoritmo DES o AES en modo CBC (Cipher Block Chaining, o encadenamiento de cifrado por bloques, descrito en el capítulo 2). Una asociación de seguridad que utilice este método requiere en su implementación más elemental al menos una llave privada de 56 bits asociada con el algoritmo DES. Sin embargo, con el surgimiento del AES como algoritmo sustituto, usualmente se emplea una llave de al menos 128 bits de longitud.

El ESP se define como un nuevo encabezado insertado en el paquete. El procesamiento por medio de ESP también incluye la transformación de los datos en forma ilegible (encriptada). El encabezado ESP contiene el SPI para determinar la asociación de seguridad del equipo receptor. El formato de los datos "encapsulados" depende enteramente de los algoritmos de encriptación utilizados por la asociación de seguridad. El formato general de un encabezado ESP se muestra en la figura 4.6:

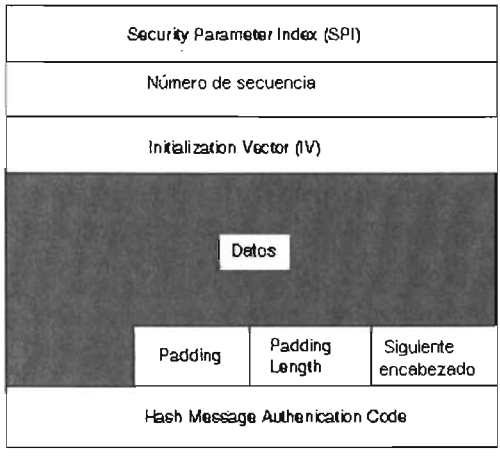


Fig. 4.6: Encabezado ESP

Los primeros 32 bits de la cabecera ESP especifican el SPI. Este SPI especifica qué Asociación de Seguridad emplear para desencapsular el paquete ESP. Los siguientes 32 bits almacenan el Número de Secuencia. Este número de secuencia se emplea para protegerse de ataques por repetición de mensajes. Los siguientes 32 a 64 bits especifican el Vector de Inicialización (IV - Initialization Vector) que se emplea para el proceso de cifrado. Los algoritmos de cifrado simétrico pueden ser vulnerables a ataques por análisis de frecuencias si no se emplean IV's. El IV asegura que dos cargas de datos idénticas generen dos cargas cifradas diferentes.
IPSEC emplea cifradores de bloque para el proceso de cifrado. Por ello, puede ser necesario rellenar la carga del paquete si la longitud de la carga no es un múltiplo de la longitud del paquete. En ese caso se añade la longitud del relleno (pad length). Este campo tiene una función adicional: es posible añadir caracteres de relleno al campo de datos para ocultar así su longitud real y, por tanto, las características del tráfico. Un atacante suficientemente hábil podría deducir cierta información a partir del análisis de ciertos parámetros de las comunicaciones, aunque estén cifradas, tales como el retardo entre paquetes y su longitud. La función de relleno está pensada para dificultar este tipo de ataques, los cuales se conocen como ataques por análisis de frecuencias.
Tras la longitud del relleno se coloca el campo de 2 bytes "Siguiente encabezado", que especifica el siguiente encabezado. Por último, se añaden los 96 bit del HMAC para asegurar la integridad del paquete. Este HMAC sólo tiene en cuenta la carga del paquete; la cabecera IP no se incluye dentro de su proceso de cálculo.

Dado que ESP proporciona más funciones que AH, el formato del encabezado es más complejo; este formato consta de una cabecera y una cola que rodean los datos transportados. Dichos datos pueden ser cualquier protocolo IP (por ejemplo, TCP, UDP o ICMP, o incluso un paquete IP completo).

En la figura 4.7 se muestra la estructura de un paquete empleado por el protocolo ESP, en la que se observa cómo el contenido o carga útil viaja cifrado.

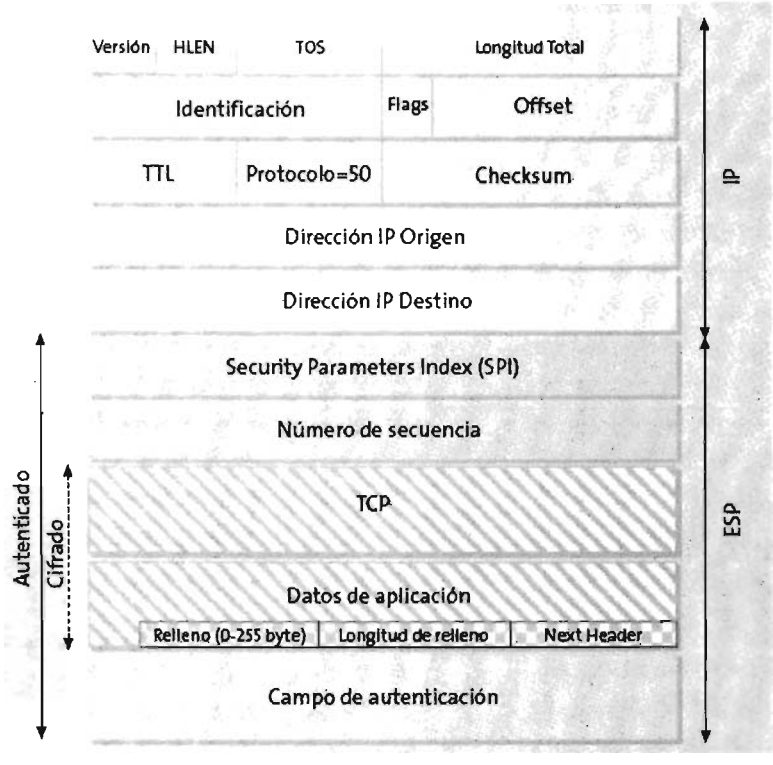


Fig. 4.7: Datagrama empleado por ESP

En circunstancias normales, el ESP se encontrará incluido dentro del AH. El encabezado ESP se genera y añade al paquete tras cifrarlo y calcular su HMAC. El equipo de cómputo que genera el paquete de información encriptará los datos usando el procedimiento y llave elegidos en la asociación de seguridad y colocará el SPI en el encabezado ESP. Durante el tránsito hasta su destino, si el paquete es interceptado por un tercero, sólo obtendrá un conjunto de bits ininteligibles. En el destino, el receptor aplica de nuevo el algoritmo de cifrado con la misma clave, recuperando los datos originales. Al recibirse el paquete, el AH será procesado primero, si está presente. La autentificación es efectuada sobre el contenido encriptado del paquete. Si los datos encriptados han sido alterados en tránsito, el procesamiento del AH desechará el paquete. De lo contrario, el equipo receptor extraerá la llave y el procedimiento criptográfico asociado con el ESP y desencriptará los datos.

Si el ESP no estuviese protegido por un AH durante su recorrido, existen dos métodos relativamente confiables para indicar si los datos encriptados han sido modificados en tránsito. El primero es por medio del proceso de cifrado CBC en el cual se verifica que la longitud de cada bloque encriptado corresponda a 32 o 64 bits dependiendo del tipo de implementación que se esté utilizando. Por otro lado, el proceso de cifrado CBC puede verificar si el valor del siguiente encabezado tiene un valor válido.

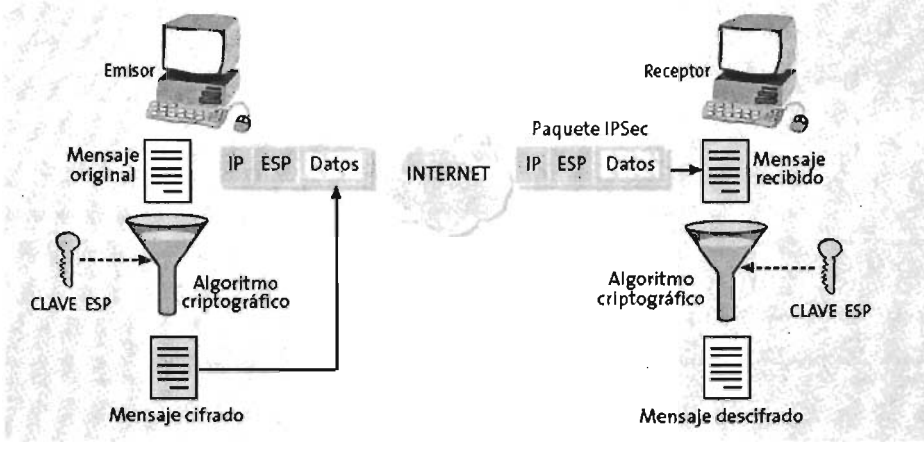


Fig. 4.8: Funcionamiento de ESP

En la figura 4.8 se representa cómo el protocolo ESP permite enviar datos de forma confidencial. El emisor toma el mensaje original, lo cifra, utilizando una llave determinada, y lo incluye en un paquete IP a continuación del encabezado ESP. Durante el tránsito hasta su destino, si el paquete es interceptado por un tercero sólo obtendrá un conjunto de bits ininteligibles. En el destino, el receptor aplica de nuevo el algoritmo de cifrado con la misma llave, recuperando los datos originales. Evidentemente, la seguridad de este protocolo reside en la robustez del algoritmo de cifrado, es decir, que un atacante no puede descifrar los datos sin conocer la llave, así como en que la llave ESP únicamente la conocen el emisor y el receptor.

La distribución de claves de forma segura es, por consiguiente, un requisito esencial para el funcionamiento de ESP y también de AH, como hemos visto anteriormente. Asimismo, es fundamental que el emisor y el receptor estén de acuerdo tanto en el algoritmo de cifrado o de hash y como en el resto de parámetros comunes que utilizan, por medio del uso de las asociaciones de seguridad. La labor de puesta en contacto y negociación de llaves es realizada por el protocolo de control denominado IKE (*Internet Key Exchange*, o Intercambio de Llaves de Internet), el cual se mencionó con anterioridad.

4.4.3.3 Administración e intercambio de llaves en IPSEC

Un requerimiento esencial para IPSEC o cualquier otro protocolo de seguridad es contar con un método para definir la relación entre entidades particulares autorizadas, sus llaves de encriptación y códigos de identificación dentro de los mensajes que se intercambian. Esto ha dado surgimiento a las asociaciones de seguridad en IPSEC. Como se mencionó anteriormente, el SPI es empleado para seleccionar la asociación de seguridad correcta para un encabezado IPSEC en particular. El siguiente paso es contar con un método para establecer las asociaciones de seguridad y la asignación de SPI's a las mismas. A partir de esta necesidad, han surgido varios protocolos que han sido propuestos como estándar pero pocos de ellos se han puesto en práctica.

Actualmente, el protocolo IKE se emplea como un estándar, definido por el IETF para realizar tanto esta función de gestión automática de claves como el establecimiento de las asociaciones de seguridad correspondientes. IKE es un protocolo híbrido que ha resultado de la integración de dos protocolos complementarios: ISAKMP (*Internet Security Association and Key Management Protocol*) y Oakley. ISAKMP define de forma genérica el protocolo de comunicación y la sintaxis de los mensajes que se utilizan en IKE, mientras que Oakley especifica la lógica de cómo se realiza de forma segura el intercambio de una clave entre dos partes que no se conocen previamente.

En la figura 4.9 se muestra de manera esquemática el funcionamiento del protocolo IKE y el modo en que se obtiene una clave de sesión, que es la que se utiliza para proteger las conexiones ESP o AH:

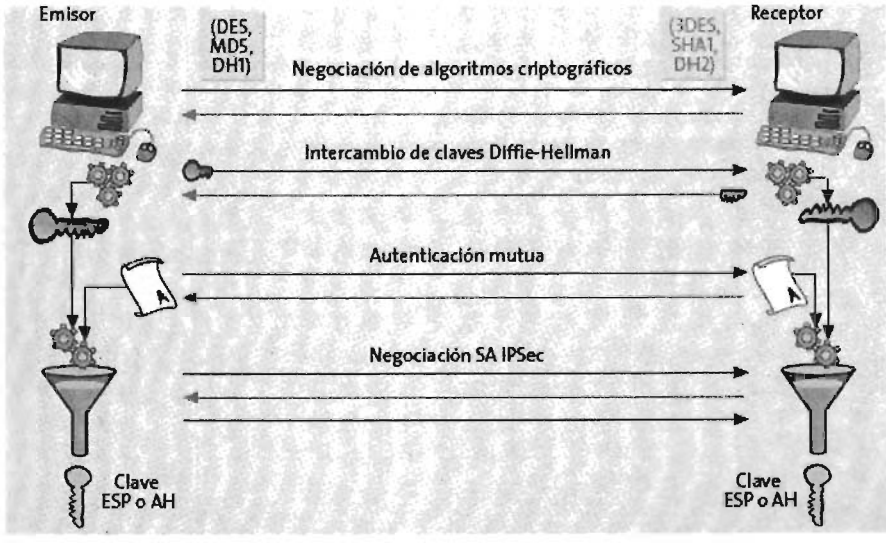


Fig. 4.9: Funcionamiento de IKE

El objetivo principal de IKE consiste en establecer una conexión cifrada y autenticada entre dos entidades, a través de la cual se negocian los parámetros necesarios para establecer una asociación de seguridad IPSEC. Dicha negociación se lleva a cabo en dos fases:

1. La fase común a cualquier aplicación, en la que ambos nodos establecen un canal seguro y autentificado. Dicho canal seguro se consigue mediante el uso de un algoritmo de cifrado simétrico y un algoritmo HMAC. Las llaves necesarias se derivan de una llave maestra que se obtiene mediante un algoritmo de intercambio de claves Diffie-Hellman. Este procedimiento no garantiza la identidad de los nodos, para ello es necesario un paso adicional de autentificación. Existen varios métodos de autentificación, los dos más comunes se describen a continuación:

- El primer método de autentificación se basa en el conocimiento de un secreto compartido que, como su propio nombre indica, es una cadena de caracteres que únicamente conocen los dos extremos que quieren establecer una comunicación IPSEC. Mediante el uso de funciones hash cada extremo demuestra al otro que conoce el secreto sin revelar su valor; así los dos se autentifican mutuamente. Para no debilitar la seguridad de este mecanismo de autentificación, debe configurarse un secreto distinto para cada par de nodos, por lo que el número de secretos crece muy rápidamente cuando aumenta el número de nodos. Por esta razón en entornos en los que se desea interconectar muchos nodos IPSEC la gestión de llaves se torna muy complicada. En este caso no resulta óptimo el uso de autentificación mediante secreto compartido, sino la autentificación basada en certificados digitales.

- En los estándares IPSEC está previsto el uso de un método de autentificación que se basa en utilizar certificados digitales X509v3. El uso de certificados permite distribuir de forma segura la llave pública de cada nodo, de modo que éste puede probar su identidad mediante la posesión de la llave privada y ciertas operaciones de criptografía pública. La utilización de certificados requiere de la aparición de un elemento más en la arquitectura IPSEC, la PKI (Infraestructura de Clave Pública), cuya integración se tratará con detalle más adelante.

2. En la segunda fase el canal seguro IKE es usado para negociar los parámetros de seguridad específicos asociados a un protocolo determinado, en este caso IPSEC. Durante esta fase se negocian las características de la conexión ESP o AH y todos los parámetros necesarios. El equipo que ha iniciado la comunicación ofrecerá todas las posibles opciones que tenga configuradas en su política de seguridad y con la prioridad que se hayan configurado. El sistema receptor aceptará la primera que coincida con los parámetros de seguridad que tenga definidos. Asimismo, ambos nodos se informan del tráfico que van a intercambiarse a través de dicha conexión.

4.4.3.4 Integración del protocolo IPSEC con la Infraestructura de Llave Pública (PKI)

El uso de una PKI aparece en IPSEC como respuesta a la necesidad de un procedimiento para autentificar de forma fiable a un conjunto de nodos que desean comunicarse mediante IPSEC, siendo dicho conjunto de nodos muy numeroso. La existencia de una PKI ofrece otras ventajas, ya que se centraliza el alta y baja de los usuarios, además se posibilita la introducción de tarjetas inteligentes para soportar los certificados, lo cual es muy interesante para la aplicación de IPSEC en un entorno de trabajadores remotos o usuarios móviles.

Como se vio en el capítulo anterior, bajo el nombre de PKI se engloban todos los elementos y procedimientos administrativos que permiten emitir, revocar y, eventualmente, renovar los certificados digitales para una comunidad de usuarios. En el caso de IPSEC los sujetos de los certificados son los nodos IPSEC, mientras que la función de los certificados es proporcionar un medio fiable para autentificar la identidad de los dispositivos IPSEC. Cada uno de los dispositivos IPSEC dispondrá de un certificado digital que contendrá su llave pública y la información suficiente para identificar de forma unívoca al dispositivo (tal como su nombre DNS, su dirección IP o su número de serie). Esta asociación entre llave pública e identidad está avalada por la firma de la Autoridad de Certificación integrada en la PKI, que da validez al certificado. Se supone que todos los dispositivos IPSEC reconocerán como válida la misma AC, para lo cual deberán disponer de una copia del certificado de la propia AC.

Los protocolos para la interacción de los dispositivos IPSEC con una PKI no están especificados en ninguno de los protocolos de IPSEC. Todos los fabricantes utilizan X.509v3 como formato común de los certificados, así como los estándares de la serie PKCS para la solicitud y descarga de certificados. Sin embargo, el protocolo de comunicaciones, mediante el cual los dispositivos IPSEC dialogan con la PKI, no está totalmente estandarizado. Esto hace que existan varias alternativas según el fabricante de que se trate.

En general los nodos IPSEC necesitan realizar ciertas operaciones básicas con una PKI: acceder al certificado de la AC, solicitar y descargar un certificado, así como comprobar la validez de un certificado recibido. En la actualidad, la mayoría de los nodos IPSEC realizan la validación de los certificados mediante consultas de la Lista de Certificados Revocados (LCR), que se almacena en el directorio de la PKI. Para ello, cada uno de los nodos mantendrá una copia de la LCR, que actualizará periódicamente mediante una consulta por medio del protocolo LDAP al directorio de la PKI. Típicamente, los periodos de actualización de la LCR serán del orden de horas, de modo que existirá cierto retardo desde que la PKI revoca un certificado hasta que todos los dispositivos tengan constancia de dicha revocación.

Para la solicitud y descarga de certificados existe un protocolo denominado SCEP (*Simple Certificate Enrollment Protocol*), que se ha convertido en un estándar de facto en las operaciones de registro y descarga de certificados para aplicaciones IPSEC. SCEP es un protocolo desarrollado originalmente por Cisco y Verisign, que se basa en el intercambio de mensajes PKCS, mediante protocolo HTTP, para automatizar los procesos de solicitud y descarga de certificados.

En la Figura 4.10 se representan los flujos de comunicación entre una PKI y un nodo IPSEC. Inicialmente, cada uno de los nodos genera un par de llaves (pública y privada) y envía una petición de certificado a la AC, en la que incluye información de su identidad y su llave pública. Al mismo tiempo, el nodo descarga el certificado raíz de la AC; a continuación, la AC genera un certificado para el dispositivo IPSEC y éste lo recibe. A partir de ese momento el nodo IPSEC podrá usar su certificado en una negociación IKE para autenticarse frente a otros dispositivos. Periódicamente los dispositivos IPSEC accederán al directorio de la PKI para actualizar la CRL.

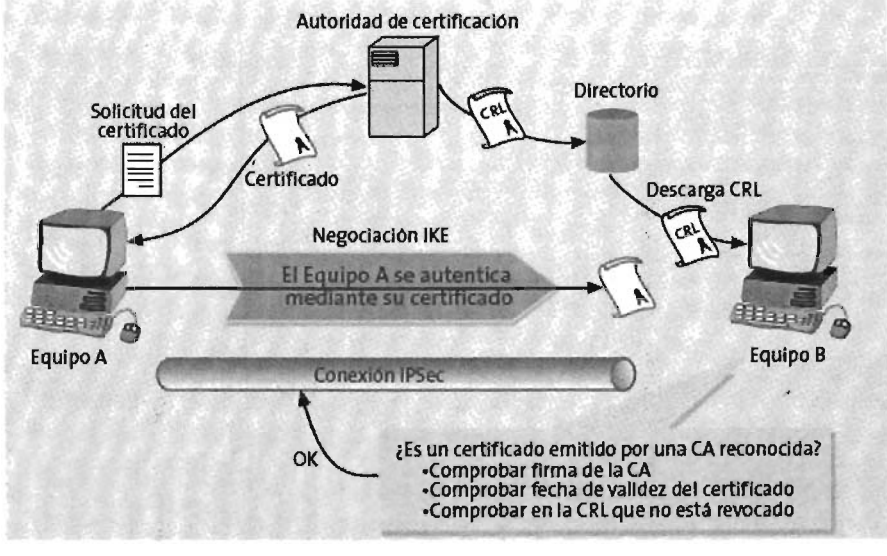


Fig. 4.10: Integración de PKI con IPSEC

4.4.3.5 Características de seguridad ofrecidas por IPSEC

Un sistema de seguridad que emplee la Criptografía como mecanismo de seguridad principal se considera razonablemente seguro cuando proporciona un

nivel adecuado de integridad, confidencialidad y no repudio del origen de los datos transmitidos.

A continuación se mencionan las características de seguridad que pueden encontrarse dentro del protocolo IPSEC, las cuales cumplen con lo anterior y proporcionan algunas ventajas adicionales.

- **Integridad y autentificación del origen de los datos**

El protocolo AH es el más adecuado si no se requiere cifrado. La opción de autentificación del protocolo ESP ofrece una funcionalidad similar, aunque esta protección, a diferencia de AH, no incluye al encabezado IP. Como se mencionó anteriormente, esta opción es de gran importancia para aquellas aplicaciones en las cuales es importante garantizar la invariabilidad del contenido de los paquetes IP.

- **Confidencialidad**

El servicio de confidencialidad se obtiene mediante la función de cifrado incluida en el protocolo ESP. En este caso es recomendable activar la opción de autentificación, ya que si no se garantiza la integridad de los datos el cifrado es inútil. Esto es debido a que aunque los datos no pudiesen ser interpretados por nadie en tránsito, éstos podrían ser alterados haciendo llegar al receptor del mensaje tráfico sin sentido que sería aceptado como tráfico válido.

Además de ofrecer el cifrado del tráfico, el protocolo ESP también tiene herramientas para ocultar el tipo de comunicación que se está realizando; para ello permite introducir caracteres de relleno en el contenido de los datos del paquete, de modo que se oculta la verdadera longitud del mismo. Ésta es una protección útil contra las técnicas de análisis de tráfico, que permiten a un atacante deducir información útil a partir del estudio de las características del tráfico cifrado. El análisis de tráfico es un riesgo que debe considerarse seriamente, ya que, por mencionar un ejemplo, recientemente se ha documentado la viabilidad para deducir información a partir del tráfico cifrado de una conexión SSH. Es previsible que este tipo de ataques se vuelvan más habituales y sofisticados en el futuro, conforme se generalice el cifrado de las comunicaciones.

- **Detección de repeticiones**

La autentificación protege contra la suplantación de la identidad IP; sin embargo, un atacante todavía podría capturar paquetes válidos y reenviarlos al destino. Para evitar este ataque, tanto ESP como AH incorporan un procedimiento para detectar paquetes repetidos. Dicho procedimiento está basado en un número de secuencia incluido en la cabecera ESP o AH, el emisor

incrementa dicho número por cada datagrama que envía y el receptor lo comprueba, de forma que los paquetes repetidos serán ignorados.
Esta secuencia no podrá ser modificada por el atacante, debido a que se encuentra protegida por medio de la opción de integridad para cualquiera de los dos protocolos (AH y ESP) y cualquier modificación en este número provocaría un error en la comprobación de la integridad del paquete.

- **Control de acceso: autentificación y autorización**

Dado que el uso de ESP y AH requiere el conocimiento de claves, y dichas claves son distribuidas de modo seguro mediante una sesión IKE en la que ambos nodos se autentican mutuamente, existe la garantía de que sólo los equipos deseados participan en la comunicación. Es conveniente aclarar que una autentificación válida no implica un acceso total a los recursos, ya que IPSEC proporciona también funciones de autorización. Durante la negociación IKE se especifica el flujo de tráfico IP que circulará a través de la conexión IPSEC. Esta especificación es similar a un filtro de paquetes, considerándose el protocolo, las direcciones IP de los puertos origen y destino, el byte "TOS" (ver fig. 4.4 y fig. 4.7) y otros campos. Por ejemplo, puede utilizarse IPSEC para permitir el acceso desde una sucursal a la red local del centro corporativo, pero impidiendo el paso de tráfico hacia máquinas especialmente protegidas.

- **No repudio**

El servicio de no repudio, el cual asegura que el remitente es realmente quien dice ser, es técnicamente posible en IPSEC, si se usa IKE con autentificación mediante certificados digitales. En este caso, el procedimiento de autentificación se basa en la firma digital de un mensaje que contiene, entre otros datos, la identidad del participante. Dicha firma, gracias al vínculo entre la llave pública y la identidad que garantiza el certificado digital, es una prueba inequívoca de que se ha establecido una conexión IPSEC con un equipo determinado, de modo que éste no podrá negarlo. En la práctica, sin embargo, esta prueba es más compleja, ya que requeriría almacenar los mensajes de negociación IKE y, además, no está definido un procedimiento para referenciar este evento a una fecha concreta.

4.5 Otras tecnologías que hacen uso de la Criptografía

Las redes de cómputo hoy en día proporcionan una gran variedad de servicios a infinidad de usuarios, y necesitan una arquitectura cuidadosamente diseñada para dar seguridad a las operaciones que en ellas se realizan. La arquitectura de seguridad crea un entorno de trabajo completo, dentro del cual se realizan distintos servicios de seguridad. Debe dar respuesta adecuada a cada una de las aplicaciones que requieren seguridad, y cambiar su esquema

dependiendo de la acción solicitada (por ej. generar e intercambiar llaves, autorizar o identificar a un individuo, etc.) y según el tipo de aplicación que requiera el servicio, ya que, a manera de ejemplo, no es lo mismo autentificar usuarios dentro de una aplicación de correo electrónico, aplicación en la que las demoras de tiempo no tienen demasiada importancia, que realizar la autentificación en un servidor que esté siendo consultado frecuentemente.

Además de dar respuesta a los requisitos de seguridad mencionados en el apartado anterior, hay otras características importantes a la hora de diseñar un sistema seguro. Resulta importante considerar una arquitectura de servidores distribuidos, de tal forma que cuando un elemento o grupo de elementos del sistema se queden inoperantes, no se afecte a la totalidad de la red y la deje fuera de servicio, especialmente si los sistemas han de dar servicio a un gran número de clientes, y la ampliación de este número no deba producir ningún problema. También se debe procurar que las operaciones para la autentificación sean prácticamente transparentes para el usuario, en donde sólo se le requiera introducir un nombre de usuario y contraseña en una consola o por algún otro medio (certificado digital, tarjeta magnética, tarjeta inteligente, controles de acceso biométrico, tales como la huella dactilar, reconocimiento del iris, etc.). Todo esto conduce a la idea de utilizar un sistema de autentificación de usuarios basado en una arquitectura distribuida como el ejemplo que se menciona a continuación.

4.5.1 Autentificación segura por medio de Kerberos

Consiste en un servicio de autentificación basado en una arquitectura cliente/servidor distribuida, diseñado por el Instituto Tecnológico de Massachusetts (MIT) y que genera y emplea "tickets" para autentificar usuarios. Se han realizado varias versiones, siendo la versión 4 la más difundida, aunque con algunas deficiencias de funcionamiento que corrige la versión 5.

En la figura 4.11 se muestra, a grandes rasgos, el proceso de autentificación por medio de Kerberos. El algoritmo de cifrado simétrico utilizado en este caso es el DES.

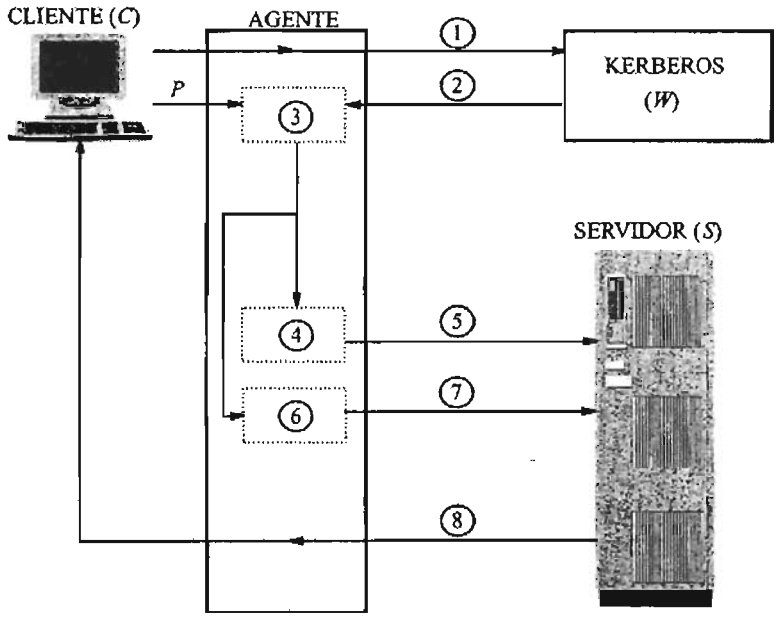


Fig. 4.11: Sistema de autentificación Kerberos

Un cliente (C) desea acceder a un sistema protegido, un servidor (S), y para ello solicita permiso a Kerberos (W) a través de un agente (paso 1). La petición contiene la identidad de C, la identidad de S y la hora de la petición. Si Kerberos aprueba la petición envía a C varios parámetros, cifrados con DES bajo una clave simétrica compartida por C y por W (paso 2). Estos parámetros son la clave de sesión que va a ser utilizada por C y S, la identidad de S, una estampa de tiempo (*timestamp*), el eco de la hora de la petición y el ticket solicitado, cuya información está cifrada por una llave simétrica (privada) compartida sólo por W y S de forma que sólo S puede entenderlo.

En el paso 3, C descifra la información recibida de W con la llave privada compartida por ambos y que ha sido calculada mediante una función unidireccional a partir de una contraseña P, de C; tras este paso, se desecha dicha llave.

Una vez descifrada la información, se comprueba si son correctas la identidad de S y la hora. Si es así, se valida el ticket (paso 4) que está cifrado con la llave simétrica compartida por S y W, y se envía a S (paso 5). Para que S tenga certeza de que el ticket es válido, se cifra con la llave de sesión que está incluida en el propio ticket (paso 6), y se envía a S (paso 7). En S se descifran el ticket y su versión cifrada para autenticar a C, se comprueba su validez por los

timestamps y se comparan ambas copias descifradas del ticket. Si coinciden, S envía a C el permiso de acceso (paso 8).

4.5.2 Correo electrónico seguro por medio de Secure-MIME (S/MIME)

El protocolo S/MIME es una especificación para envío de mensajes y correo electrónico seguro que comenzó a desarrollarse en 1995 por varios fabricantes de software con la finalidad de proporcionar seguridad a mensajes de correo electrónico con formato MIME. S/MIME proporciona servicios de confidencialidad, integridad, no repudio del origen de los datos y autentificación.

S/MIME combina algoritmos criptográficos seguros con los estándares de correo electrónico más difundidos, y ha sido pensado para conseguir la máxima interoperabilidad, de forma que dos paquetes de software diferentes que implementen S/MIME no tengan ningún problema para establecer una comunicación segura. En la actualidad, existe un grupo de trabajo, el S/MIME Working Group, que mantiene una gran actividad para conseguir tanto la normalización completa de S/MIME como para lograr que éste alcance una gran difusión. Desde hace tiempo circula la especificación Internet dada por La IETF ha especificado un estándar en Internet para S/MIME, del cual destacan dos documentos:

- **S/MIME Message Specification**. Describe un protocolo para añadir servicios de cifrado y firma criptográfica a datos de tipo MIME. Se especifica cómo crear una parte del cuerpo MIME que ha sido tratada criptográficamente (según el PKCS #7[4]), cómo pedir una certificación (según el PKCS #10) y los tipos MIME que han de transportar esas partes cifradas y esas peticiones. También aparecen descritos los requerimientos y las recomendaciones necesarios para el funcionamiento de los agentes de usuario.

- **S/MIME Certificate Handling**. Hace una descripción de los mecanismos S/MIME utilizados para crear y validar las llaves por medio de certificados.

S/MIME puede ser utilizado por aplicaciones de correo electrónico tradicionales para añadir servicios de seguridad criptográfica a los mensajes que envían, aunque su uso no está restringido al correo, sino que puede utilizarse en cualquier protocolo que transporte datos con formato MIME, como HTTP.

[4] PKCS es un conjunto de normas para la implantación de Criptografía de llave pública redactado por la empresa RSA Data Security Inc. en cooperación con Apple, Microsoft, DEC, Lotus, Sun y MIT.

S/MIME contempla la utilización de algoritmos criptográficos simétricos para proporcionar la confidencialidad y algoritmos de llave pública que permiten realizar el intercambio de las mismas y las firmas digitales.

El algoritmo cifrador simétrico por omisión es el RC2 con clave de 40 bits, aunque también es posible utilizar DES con el método de cifrado CBC, Triple-DES o RC5. Si se quiere utilizar encriptación más fuerte, es conveniente el uso de los algoritmos Triple-DES y RC5 (empleando bloques de 64 bits), con 64 ó 128 bits de llave. Como mejor práctica, se realizan las firmas digitales con el algoritmo SHA-1, aunque los receptores deberían ser capaces de entender tanto los mensajes firmados con SHA-1 como aquellos firmados con MD5 (utilizado por las versiones más antiguas de S/MIME).

El algoritmo de llave pública por omisión es RSA (según el PKCS #1) con llaves desde 512 bits hasta 1,024 bits. El algoritmo utilizado para realizar el intercambio de claves es el de Diffie-Hellman.

Cuando no se sepa el tipo de Criptografía simétrica utilizada por el receptor, es preferible que el agente de usuario (aplicación de mensajería) del emisor cifre con RC2 de 40 bits si la llave pública del receptor es de 512 bits. Si la llave pública es mayor de 512 bits, entonces se recomienda cifrar utilizando el Triple-DES.

El formato de los mensajes S/MIME es una combinación de los cuerpos de mensaje dados por MIME y de los objetos especificados por los estándares PKCS #7 y PKCS #10. Así, los datos que han de protegerse tienen un formato MIME canónico al que se le añaden otros datos, como los certificados e identificadores de algoritmos que siguen las pautas marcadas por los PKCS y forman un objeto PKCS. Finalmente, a este objeto PKCS se le aplica un formato MIME. Concretamente, en el PKCS #7 se define un formato de mensajes extensible y flexible para representar los resultados de datos que han sufrido transformaciones criptográficas. En el PKCS #10 está redactada la sintaxis de los mensajes para peticiones de certificación.

Los mensajes S/MIME utilizan tres campos:

- Campo de datos (DataContentType). Activados por los agentes para indicar los servicios de seguridad aplicados en el mensaje.

- Campo de datos firmados (SignedDataContentType). Los agentes emisores han de utilizar este campo para situar la firma digital del mensaje o, en su defecto, llevar el certificado.

- Campo de datos protegidos (EnvelopedDataContentType). Se utiliza para dar protección a los mensajes.

Los mensajes S/MIME tienen un único formato para los datos protegidos, y varios formatos tanto para los datos que sólo van firmados como para los que van cifrados y firmados. Ello es debido a la necesidad de adaptar distintos entornos, especialmente cuando se trata de mensajes firmados.

S/MIME puede garantizar la confidencialidad del tráfico de datos utilizando direcciones de correo anónimas, que deshacen la dirección original. Además, para evitar que un posible intruso pudiera identificar la identidad del emisor de un mensaje a través de su firma digital, calcula primeramente dicha firma, la añade al mensaje original y cifra todo el bloque. S/MIME utiliza certificados digitales basados en la norma X.509[5].

Si se comparan las propuestas de S/MIME con los sistemas PEM y PGP, mencionados en el capítulo anterior, podemos observar claramente las ventajas que aporta éste. El sistema de certificados utilizados por PGP, basado en la "red de confianza", es útil para grupos pequeños, pero cuando el número de usuarios es elevado, presenta serias dificultades. S/MIME tiene la misma facilidad a la hora de manejar los certificados, con independencia del número de usuarios. Por otra parte, PEM utiliza un formato de mensajes basado en mensajes de texto de 7 bits. En contraste, S/MIME está diseñado para trabajar con archivos adjuntos tanto binarios como de texto.

4.6 Protocolos y aplicaciones orientadas al comercio electrónico

En la mayor parte de estos protocolos y aplicaciones resulta prioritario identificar perfectamente al emisor y al receptor de los datos antes que la protección de los mismos. Por lo tanto, se deben de cuidar mucho más los servicios de autentificación que los de confidencialidad. Esto, unido a que usualmente los mensajes enviados son cortos, hace que en este tipo de aplicaciones tenga mayor interés la utilización de algoritmos de llave pública y en algunos casos, algoritmos de cifrado por bloques.

El paso de la Criptografía desde el ámbito militar hacia ámbitos civiles llegó en gran medida por la necesidad de establecer comunicaciones seguras para las aplicaciones bancarias. Algunas de estas aplicaciones seguras se han consolidado desde tiempo atrás. Esto ocasiona que hagan uso de algoritmos y protocolos de seguridad aparentemente desfasados, aunque mantienen plenamente su vigencia. La mayor parte de las aplicaciones bancarias y financieras seguras se desarrollaron para una infraestructura de líneas alquiladas de comunicaciones, tipo X.25[6]. Muchas de ellas sufrieron

[5] Estándar de mayor difusión para la definición de certificados digitales, recomendado por la ITU (Internacional Telecommunication Union)

[6] Protocolo de comunicaciones estándar de amplia difusión, aprobado por la ITU en 1976. Define las tres capas inferiores del modelo de referencia OSI.

modificaciones y adaptaciones para ser utilizadas sobre otro tipo de infraestructuras, tales como X.400. En la actualidad, la tendencia clara es la utilización de Internet como soporte de comunicaciones. Esto hace que algunas aplicaciones estén ya introduciendo modificaciones para adaptarse e integrarse con el formato MIME, actualmente el principal estándar de Internet.

A continuación se analizarán algunas de las aplicaciones desarrolladas tiempo atrás para la realización de transacciones bancarias seguras, y que sirvieron de antecedente para las actuales tendencias en el comercio electrónico, el cual continúa en pleno desarrollo, debido a su creciente utilización a través de Internet.

4.6.1 ISO 8730

Es una de las normas más antiguas y extendidas en aspectos de seguridad para transferencias interbancarias. La norma ISO 8730, basada en la norma ANSI X9.9, utiliza algoritmos de llave privada, tales como DES, para autentificar las transferencias. Los mecanismos de distribución de llaves están regulados por la norma ISO 8732.

Según ISO 8730, varios de los campos que integran una transferencia interbancaria deben estar incluidos en el mensaje de autentificación. Entre ellos están:

- MAC (*Message Authentication Code*): código de autentificación del mensaje, el cual consta de 8 dígitos hexadecimales.
- DMC: fecha en la que se calculó el MAC.
- IDA: identificador para que el receptor utilice la clave de autentificación.
- MID: identificador de mensaje. Consiste en un número generado por el emisor a partir de la DMC e IDA como protección contra la duplicación o pérdida del mensaje.
- Elementos específicos en el texto del mensaje, tales como valor de la transacción, entidades participantes, beneficiarios, etc.

Los distintos campos tienen sus propios delimitadores, los cuales marcan el inicio y el final de cada uno de ellos.

En la norma ISO 8730 existen varias opciones para tratar los datos formateados a la hora de calcular el valor del código de autentificación del mensaje (MAC, por sus siglas en inglés). Entre estas opciones se encuentra la posibilidad de considerar los datos como texto ASCII o como un archivo binario. Si el mensaje se trata como texto, puede dejarse como está o editarlo, suprimiéndole los caracteres superfluos que hacen más lenta la transmisión y le restan transparencia. Asimismo, en el caso de que se traten los datos como

texto, se puede seleccionar la posibilidad de calcular el MAC a partir del mensaje completo o decidir qué campos se incluyen.

4.6.2 SWIFT

SWIFT (*Society for Worldwide Interbank Financial Telecommunication*, o Sociedad para la Telecomunicación Financiera Interbancaria Mundial) ofrece un servicio para transferencia de pagos con varios mecanismos de seguridad, tales como el cifrado de líneas interurbanas, la protección de acceso a la red mediante empleo de códigos y la posibilidad de cifrado en las conexiones usuario-red alquiladas. Los usuarios de SWIFT suelen utilizar productos específicos de esa entidad, los cuales son añadidos a sus terminales para permitir la identificación del operador.

Las comunicaciones se realizan sobre una red de paquetes conmutados tipo X.25 y las transacciones financieras pueden tener varios protocolos, formatos y otras medidas de seguridad, tales como la autentificación. SWIFT ha mejorado la seguridad básica de la red añadiendo algunos elementos de seguridad de usuario, tales como el intercambio seguro de llaves y el control de acceso con tarjetas inteligentes.

Para realizar operaciones de mayor volumen, tales como pago de nómina, pensiones, información de gestión de riesgos, protección del estado de cuenta, etc., SWIFT ha creado la Transferencia de Archivos Interbancaria (IFT9 que opera sobre infraestructura X.400. Para la transferencia de archivos se utiliza el protocolo p-IFT dentro de los mensajes X.400, el cual incluye un encabezado de seguridad basado en un certificado tipo X.509. Los servicios de seguridad ofrecidos por este protocolo son: integridad del contenido del mensaje, autentificación y confidencialidad. Se pueden usar varios algoritmos de llave pública o privada, para lo cual el emisor genera un certificado conocido como "token" que contiene un identificador del algoritmo seleccionado. Las funciones de seguridad son de extremo a extremo y la elección de los algoritmos o la gestión de las llaves no necesariamente dependen de SWIFT.

4.6.3 Aplicación de Seguridad EDI (Intercambio Electrónico de Datos)

La aplicación conocida como EDI, la cual significa por sus siglas en inglés *Electronic Data Interchange*, o Intercambio Electrónico de Datos, tiene servicios de seguridad definidos y normalizados. Su utilización en transacciones comerciales así lo requiere, debido a las cuestiones relacionadas con servicios de autentificación.

El ITU (*Internacional Telecommunication Union*) ha definido dos normas: las recomendaciones F.435 y X.435, que permiten utilizar EDI en el sistema de mensajería X.400. La recomendación F.435 define los servicios de mensajes EDI y la X.435los sistemas de mensajes EDI; es decir, cómo operan los servicios escritos en la norma F.435.

Los mensajes EDI (conocidos como EDIM) constan de un encabezado y de un cuerpo que, juntos, forman el campo de contenido de un mensaje X.400. En el encabezado aparecen varios campos relacionados con la identificación y el tratamiento que ha de tener el mensaje, pero sólo algunos de estos campos están relacionados con aspectos de seguridad. En el cuerpo del mensaje pueden aparecer intercambios del tipo EDIFACT (definidos por la norma ISO 9735) o del tipo ANSI X12.

Una notificación EDI (EDIN) consiste en un mensaje enlazado al mensaje original, a través de referencias cruzadas, con el que tiene una serie de campos comunes seguidos de campos que indican el tipo de notificación. Las notificaciones EDI permiten confirmar la recepción del mensaje en el nivel de aplicación y no sólo su reparto. En la figura 4.12 se puede ver el proceso de transferencia de mensajes y notificaciones EDI.

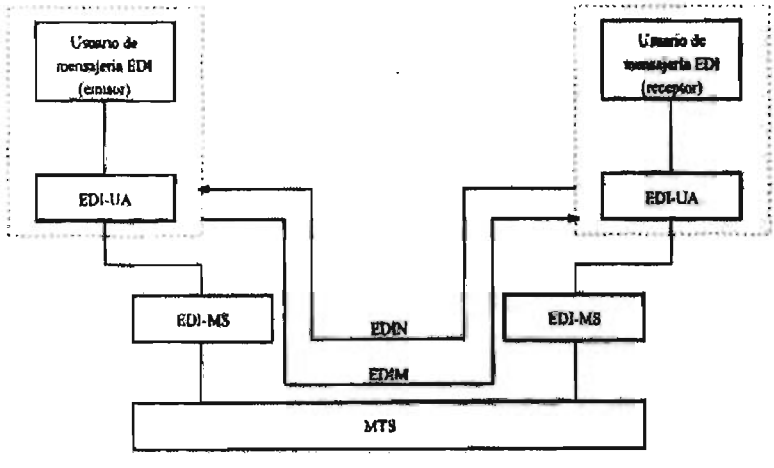


Fig. 4.12: Intercambio de mensajes y notificaciones en EDI

F.435 y X.435 introducen elementos de seguridad adicionales a los ya presentes en X.400, fundamentalmente para garantizar la autenticidad y evitar el repudio tanto de los mensajes como de las notificaciones. La persona que origina el mensaje EDI puede solicitar el servicio de no-repudio de la notificación

EDI activando los bits adecuados en el campo de petición de notificación que hay en el encabezado del mensaje.

En MIME se han incluido tres tipos de contenido diferentes para reconocer los formatos utilizados por los mensajes EDI:

- Un tipo MIME para incluir un intercambio EDI con formato ANSI X. 12.
- Un tipo MIME para incluir un intercambio EDI con formato EDIFACT.
- Un tipo MIME para incluir un intercambio EDI con formato diferente al ANSI X. 12 o al EDIFACT y que debe ser acordado previamente por los usuarios emisor y receptor.

Cuando se realiza el transporte de mensajes EDI sobre Internet se pueden utilizar los mecanismos de seguridad ofrecidos por S/MIME como servicios añadidos para la seguridad global de la estructura del mensaje.

EDIFACT

Aunque la infraestructura X.400 se utiliza para proteger un intercambio de datos EDI completo (mensaje más notificación) no es la única infraestructura que puede ser utilizada. Los intercambios están compuestos por mensajes y grupos funcionales, los cuales son grupos de mensajes similares, todos con un mismo destino, pero no necesariamente todos requieren protección. La infraestructura X.435 no tiene capacidad para realizar una seguridad selectiva, por lo que, para proteger los intercambios, y sus subunidades, se ha pensado en construir la protección dentro de la misma estructura EDI utilizando unos indicadores. Para regular este tipo de operaciones han aparecido varias normas, entre las que destacan la serie ANSI X12 y, particularmente, EDIFACT.

La estructura de un mensaje EDIFACT se muestra en la figura 4.13. Ese puede observar que cada parte del mensaje tiene unos delimitadores de cabecera (UNA, UNB, UNG, UNH) y de final (UNZ, UNE, UNT). Los delimitadores de la cabecera identifican y especifican el intercambio y el grupo funcional o mensaje, mientras que los delimitadores de final se utilizan para realizar chequeos.

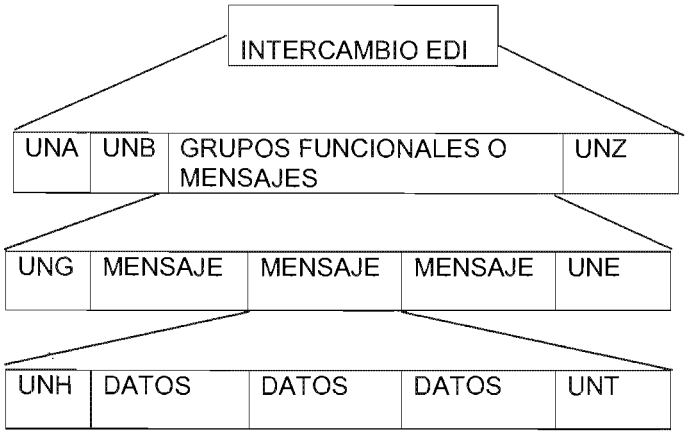


Fig. 4.13

El programa TEDIS, de la Comisión Europea, ha elaborado una propuesta sobre los servicios de seguridad de EDIFACT, principalmente para controlar la autenticidad en los mensajes y en los niveles de intercambio basada en la Criptografía de Llave Pública. Asimismo, se encuentran en desarrollo productos EDI tanto para X.25 como para X.400, y el formato EDIFACT en los mensajes irá reemplazando paulatinamente al formato original diseñado por SWIFT.

4.6.4 Sistemas de pago electrónico

El comercio electrónico, en general, y el que se realiza sobre Internet, en particular, dependen de la disponibilidad y del amplio despliegue de sistemas de pago electrónico. El núcleo de un sistema de pago electrónico está constituido por uno o varios protocolos de pago. Estos protocolos pueden estar implantados en navegadores WWW sobre HTTP, en clientes de correo sobre SMTP o en otros programas que utilicen algunos protocolos de aplicación específica.

Existen varios sistemas de pago electrónico, los cuales se han basado en protocolos desarrollados y consolidados hace tiempo; sin embargo, continuamente surgen nuevas tecnologías de pago electrónico y mejoras a las ya existentes. Aunque en la actualidad estos sistemas funcionan de forma separada, la tendencia es que puedan coexistir varios de ellos con perfecta interoperabilidad, de tal forma que los sistemas que se describen a continuación marcan la pauta de las tendencias actuales.

4.6.4.1 Dinero electrónico

La finalidad del dinero electrónico (*e-cash*) es proporcionar un sistema electrónico que sea equivalente al dinero físico que estamos acostumbrados a manejar. Los elementos que integran este sistema son los siguientes:

- Un banco suministrador del servicio, el cual proporciona el dinero electrónico.
- Una persona (cliente) que va a gastar el dinero proporcionado por el banco.
- Una persona (vendedor) que va a recibir el dinero del cliente.
- Un banco que recibe el dinero ingresado por el vendedor.

La transacción electrónica consta de tres operaciones distintas e independientes:

- El cliente hace una solicitud de transferencia desde su cuenta al banco que le proporciona el servicio de e-cash. Este banco envía al cliente la cantidad solicitada, y el cliente lo almacena bien en su disco duro o en una tarjeta inteligente u otro dispositivo de almacenamiento. De esta forma, el cliente obtiene el dinero electrónico.
- Una vez que el dinero está en poder del cliente, éste puede realizar sus compras. El cliente transferirá a un vendedor la cantidad correspondiente por un producto adquirido.
- El vendedor, tras haber obtenido el dinero electrónico pagado por un cliente, tendrá que transferirlo al banco suministrador y éste hará llegar el dinero a la cuenta del vendedor.

Hay algunas propiedades generales que debe satisfacer el dinero electrónico. En primer lugar, debe ser totalmente independiente de la plataforma utilizada o el lugar donde esté ubicada. El dinero electrónico debe mantener la propiedad de anonimato característica del dinero normal, que no proporciona información útil para conocer a los propietarios anteriores. Así, el dinero electrónico debería poder pasarse de una persona a otra sin que quede constancia o "traza" de quién ha sido previamente su dueño. Sin embargo, deben proporcionarse los medios para que cada cliente no pueda reutilizar una cantidad de dinero ya gastada. El almacenamiento del dinero electrónico en el disco duro o en la tarjeta inteligente debe ser seguro.

La propiedad de anonimato ha sido y es muy controvertida, ya que podría ser utilizada por falsificadores. Por ello, los sistemas que se desarrollan son *casi* anónimos, de forma que sólo se puede conocer la identidad del cliente bajo ciertas condiciones (por ejemplo, por mandato judicial).

En 1982, David Chaum desarrolló un mecanismo de firma "ciega" por medio de algoritmo RSA para dar anonimato al dinero electrónico, y fundó la

empresa DigiCash para comercializar este esquema bajo el nombre de "Ecash". En 1995, un banco estadounidense fue el primero en ofrecer este servicio.

En el esquema Ecash, las monedas digitales se almacenan localmente en el sistema del cliente. Estas monedas digitales se pueden cifrar con una llave secreta, derivada de una contraseña, o bien se pueden utilizar tarjetas inteligentes. Antes de aceptar una moneda, el vendedor tiene que examinar su autenticidad y su integridad con la correspondiente llave pública del banco suministrador. Posteriormente, debe verificar en línea con este banco si la moneda ha sido utilizada previamente o no. Una moneda sólo puede ser utilizada una vez; para lograr este propósito, el banco anota los números de serie para poder identificar y rechazar una moneda que se intente utilizar de nuevo. La idea de anonimato en Ecash ha sido estudiada a fondo en un proyecto llamado *Conditional Accesss For Europe*, conocido por las siglas CAFE. El resultado de la investigación realizada en este proyecto ha sido que no siempre puede realizarse la comprobación en línea de las monedas, y que el aprovechamiento de los sistemas es mejor si esta comprobación se realiza fuera de línea.

Para evitar el doble uso de la misma moneda en las comprobaciones fuera de línea, se han desarrollado algunas soluciones. Una de ellas es la utilización de algoritmos matemáticos complejos que permitan asegurar la identidad del propietario, de modo que una parte de sus características de identificación se transmitan en el proceso de pago. Esta información no revela por sí sola la identidad de su propietario, sino que tiene que combinarse con otra parte de la identificación. Otra de las soluciones es la utilización de los monederos electrónicos, que necesitan de un hardware adicional para almacenar las monedas gastadas. Otras implementaciones de dinero electrónico son "NetCash", que ofrece un anonimato débil, y "CyberCash", que carece de él.

4.6.4.2 Cheques electrónicos

Los elementos de un sistema para cheques electrónicos son los siguientes:

- Un cliente y su banco.
- Un vendedor y su banco.
- Una cámara de compensación que procese los cheques entre diferentes bancos.

En este caso, las transacciones se realizan como se indica a continuación:

1. El cliente realiza alguna compra y envía el cheque electrónico correspondiente al vendedor, que lo validará con la adecuada autorización de su banco. Si el cheque es válido, el vendedor completa la transacción con el cliente.

2. El vendedor envía el cheque a su banco para depositarlo.

3. El banco del vendedor envía el cheque a la cámara de compensación para cobrarlo. La cámara acude al banco del cliente para transferir el dinero al banco del vendedor, el cual actualiza su cuenta. Por otro lado, el banco del cliente también debe actualizar su cuenta.

Desde el punto de vista técnico, el cheque electrónico es muy simple. Puede ser un documento firmado digitalmente con la llave privada del cliente y que el vendedor comprobará con la llave pública de aquél, según el protocolo habitual de firmas digitales con llave asimétrica.

El cheque digital introduce importantes ventajas sobre el de papel, especialmente en cuanto a tiempo de procesado se refiere. Como ejemplos de sistemas de cheques electrónicos se puede mencionar al sistema de cheques conocido como "PayNow", de la empresa CyberCash. Por otro lado, el sistema conocido como "NetCheque" utiliza un esquema Kerberos para realizar la autentificación.

4.6.4.3 Pagos con tarjeta de crédito

Existen varios protocolos de seguridad para realizar pagos con tarjeta de crédito en compras electrónicas, tales como *iKP*, desarrollado por IBM, y el protocolo SET (*Secure Electronic Transaction*, o Transacción Electrónica Segura), el cual es un desarrollo conjunto de Visa y MasterCard. Este último ha tenido mayor difusión que otros protocolos similares y ha servido de base para sistemas más sofisticados de pagos con tarjeta de crédito.

La primera versión de SET fue liberada en febrero de 1996, y la segunda, en junio del mismo año. Esta última consta de tres partes: Libro 1, donde se describe el tipo de comercio; Libro 2, que es una guía para los programadores, y Libro 3, que contiene la definición del protocolo formal.

A continuación se enumeran los elementos que integran un entorno SET, junto con el nombre con el que son más usualmente conocidos en inglés dentro de los sistemas de pago electrónico:

- Centro emisor de tarjetas de usuario (*Issuer*): Es la institución financiera que emite las tarjetas.

- Usuario de la tarjeta (*Cardholder*): Es el dueño de una tarjeta bancaria autorizada por un centro emisor.
- Comerciante (*Merchant*): Es el vendedor que acepta los pagos electrónicos.
- Entidad financiera de servicio a los comerciantes (*Adquirer*): Institución financiera que da soporte a los comerciantes para proporcionarles un servicio de transacciones con tarjetas bancarias.
- Pasarela de pago (*Payment gateway*): Sistema que proporciona servicios de comercio en línea (*online*) a los vendedores.
- Autoridades de certificación (*Certification Authorities*, o CA): Entidades independientes que certifican las llaves públicas de los elementos integrantes del sistema.

Al ejecutar una transacción de pago los pasos que se dan son los siguientes:

1. Cuando un usuario decide realizar una compra, envía una instrucción de pago en línea al vendedor.
2. El comerciante se comunica en línea con su entidad financiera a través de la pasarela de pago, para que ésta autorice y capture la transacción. Normalmente, lo que envía el comerciante es la instrucción de pago recibida del cliente.
3. La entidad financiera del comerciante captura la información y puede solicitar una transacción al centro emisor de la tarjeta.
4. Finalmente, si todo está en orden, una cadena de confirmaciones (del centro emisor a la entidad financiera, de esta al comerciante y del comerciante al cliente) permite al dueño de la tarjeta realizar la compra.

En el entorno SET, el uso de llaves públicas permite las siguientes funciones:

- ✓ Cifrado de instrucciones de pago para proteger el número de la tarjeta.
- ✓ Autentificación de los tarjetahabientes ante los comerciantes y las entidades financieras para proteger contra el robo de tarjetas. Este servicio es opcional.
- ✓ Autentificación del comerciante ante el cliente y la entidad financiera como protección contra suplantadores de identidad.
- ✓ Autentificación de las entidades financieras ante los propietarios de tarjetas y ante los comerciantes, para evitar que un suplantador pudiera obtener datos sensibles contenidos en las instrucciones de pago.
- ✓ Integridad de los datos transmitidos.

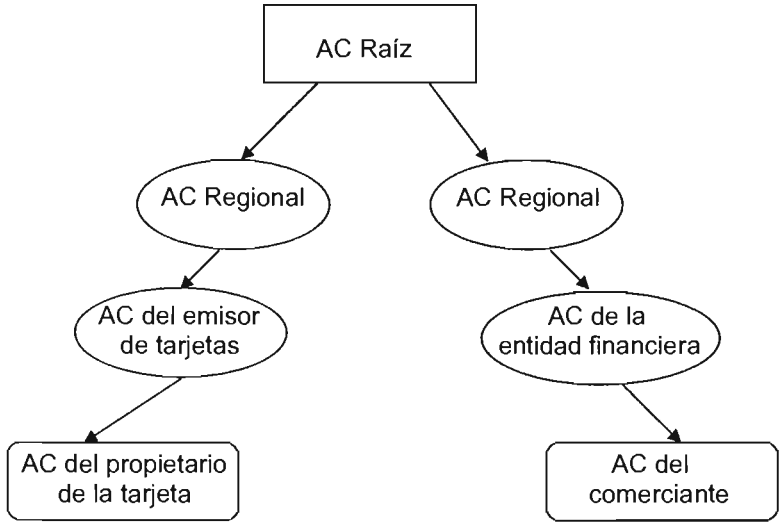


Fig. 4.14

La PKI empleada en sistemas de pago con tarjeta de crédito utiliza una arquitectura jerárquica que se muestra en la figura 4.14. Existe una autoridad suprema (Autoridad Certificadora raíz) que certificará a las AC's de los emisores de tarjetas. La AC de cada emisor de tarjetas podrá considerar distintos entornos geopolíticos y certificará a la AC correspondiente de cada uno de estos entornos, que será la AC que certificará, a su vez, a otras AC's de propietarios de tarjetas que, finalmente, certificarán a estos usuarios.

4.6.4.4 Micropagos

Un factor muy importante en la evaluación de los sistemas de pago es el costo adicional que suponen los servicios bancarios para las distintas transacciones. Partiendo de esta idea, han comenzado a surgir sistemas llamados de "micropago", diseñados para manejar muchas operaciones de montos pequeños, que en la mayor parte de los casos se busca agrupar para realizar una única transacción.

Algunas de las implantaciones más conocidas son:

- Millicent, de DEC.
- PayWord y MicroMint, diseñados por Rivest y Shamir.
- CyberCoin, de CyberCash.

CONCLUSIONES

Internet se ha convertido en los últimos tiempos en un atractivo escaparate donde se puede, además de consultar una gran variedad de información e interactuar de diversas maneras, realizar compras o transacciones bancarias desde la comodidad del hogar o lugar de trabajo. Al igual que una tienda convencional, brinda a los vendedores la posibilidad de montar su negocio y hacer llegar la publicidad de sus productos a un número incalculable de clientes potenciales por un precio muy reducido. A los compradores, les ofrece la facilidad de adquirir los productos sin desplazarse a otros lugares, pudiendo examinar y comparar sus características y precios.

La seguridad en el intercambio de información confidencial en Internet, así como las transacciones propias del comercio electrónico, dependen de la disponibilidad y amplio despliegue de sistemas que puedan proveer de un grado razonable de seguridad en los datos transmitidos así como protección de la identidad de las partes involucradas.

Como se ha podido observar a lo largo de este trabajo, los algoritmos propios de la Criptografía han permitido el surgimiento de varias soluciones tecnológicas que ofrecen distintos tipos de protección de datos y niveles de seguridad. El funcionamiento de todas ellas parte del uso de algoritmos de llave pública o de llave privada, o una combinación de ambos. Después de haber realizado un análisis de ambos tipos de algoritmos, se han podido destacar distintas ventajas y desventajas, tanto de seguridad como operativas.

Uno de los problemas más importantes que presenta la utilización de algoritmos simétricos o de llave privada en las redes de comunicaciones, es la distribución de las llaves. A pesar de los inconvenientes de este tipo de cifrado, los algoritmos simétricos de cifrado en bloque y cifrado en flujo son los únicos que permiten alcanzar altas velocidades al ser implantados en una aplicación de cómputo o en un circuito electrónico para usarlos en las redes de comunicaciones.

Los sistemas criptográficos de llave pública o asimétricos resuelven de una forma cómoda algunos de los inconvenientes de los sistemas simétricos, principalmente por medio de la autentificación y las firmas digitales (a partir de la llave pública del emisor) y, aunque proporcionan confidencialidad (a partir de la llave pública del receptor), regularmente se requiere emplear una mayor cantidad de recursos del sistema de cómputo y la red, por lo que esto repercute en una baja velocidad en el envío de mensajes cifrados de gran longitud. Por lo tanto, la implantación más adecuada de los algoritmos criptográficos resulta ser una combinación de ambos tipos de cifrado. De esta forma, los algoritmos de llave pública resultan útiles para el envío de llaves privadas de forma segura a cada uno de los nodos de la red, en los cuales se pueden emplear algoritmos

simétricos para cifrar los datos de tal manera que éstos puedan comenzar a enviar la información encriptada a través de redes inseguras.

A continuación se presenta, a manera de resumen, una comparación de los servicios y características que ofrecen cada uno de los dos tipos de Criptografía.

Criptografía de Llave Privada	Criptografía de Llave Pública
• Confidencialidad • Cierto grado de autentificación • Difícil intercambio de llaves • No emplea firma digital • Alta velocidad	• Confidencialidad • Autentificación total • Fácil intercambio de llaves • Firma digital • Baja velocidad

Si bien el propósito del presente trabajo ha sido describir y analizar las principales técnicas criptográficas y su aplicación en Internet, con lo que permiten lograr una transmisión segura de información confidencial, resulta importante haber considerado los ataques y vulnerabilidades más comunes de los algoritmos y protocolos criptográficos. Por otro lado, la fortaleza matemática que presentan los algoritmos no garantiza la seguridad de los sistemas criptográficos, ya que los errores operativos y de configuración de los sistemas operativos de cómputo, ya sea por desconocimiento de la tecnología u omisión, a menudo representan la principal vulnerabilidad de las tecnologías de encriptación. Debido a esto, aún las tecnologías y productos criptográficos más robustos y sofisticados pueden resultar altamente vulnerables debido a los descuidos.

Debido a que no existe una solución tecnológica única para cubrir al 100% y de manera óptima los requerimientos de seguridad en las comunicaciones a través de redes inseguras tales como Internet y que se ajuste a todo tipo de usuarios y organizaciones, se analizaron distintas técnicas y aplicaciones criptográficas. Por medio de un conocimiento más amplio de las mismas, así como de sus ventajas y desventajas, será posible contar con un criterio más amplio para poder entender, evaluar y validar los distintos productos que se encuentran hoy en día en el mercado y de este modo, poder seleccionar aquel que satisfaga de la mejor manera los requerimientos particulares o de negocios.

El uso efectivo de los sistemas criptográficos y la elección de los mismos requiere de un claro entendimiento de los objetivos de seguridad de la organización o usuarios que deseen explotar las amplias posibilidades que ofrece la Criptografía en materia de seguridad y el grado de dependencia hacia sus sistemas de cómputo y las comunicaciones con otros sistemas remotos.

BIBLIOGRAFÍA

Smith, Richard E. "Internet Cryptography". Ed. Addison Wesley Longman. Estados Unidos, 1997.

Stein, Lincoln D. "Web Security". Ed. Addison-Wesley. Estados Unidos, 1998.

Garfinkel, Simson y Spafford, Gene. "Web Security and Commerce". O'Reilly & Associates. Estados Unidos, 1997.

Fúster Sabater, Amparo, et. al. "Técnicas Criptográficas de Protección de Datos". Ed. Grupo Alfaomega Editor. México, 2001.

Rodríguez, Amador. "Protección del la Información". Ed. Paraninfo. España, 1986.

Rodríguez, Luis Angel. "Seguridad de la Información en Sistemas de Cómputo". Ventura Ediciones. México, 1995.

Caballero, Pino. "Seguridad Informática". Alfaomega Grupo Editor. México, 1997.

Fahn, Paul. "RSA Cryptography Today FAQ". RSA Laboratories, 1993. (http://www-ipg.umds.ac.uk/d.hill/FAQs/cryptography-faq/rsa/part1/faq.html)

Kent, S. "IP Encapsulating Security Payload (ESP)". BBN Technologies, 2005. (http://www.ietf.org/internet-drafts/draft-ietf-ipsec-esp-v3-10.txt)

Pérez Iglesias, Santiago. "Análisis del Protocolo IPSec: el estándar de seguridad en IP". Telefónica Investigación y Desarrollo, 2001. (http://www.tid.es/presencia/publicaciones/comsid/esp/23/04.pdf)

"Cómo funciona IPSEC". http://www.ipsec-howto.org/spanish/x161.html

"AES: Advanced Encryption Standard". http://www.nist.gov/aes/

"Algoritmo RSA". http://www.rsasecurity.com