Apuntes de Seguridad I

Objetivos

La seguridad informática abarca áreas que van desde la protección de los dispositivos físicos (hardware), hasta la protección de información que se encuentra físicamente en los ordenadores o la que viaja a través de la red, por eso son muy diversos los tipos de amenazas contra los que debemos proteger nuestro sistema. Desde amenazas físicas, como los cortes eléctricos, hasta errores no intencionados de los usuarios, pasando por los virus informáticos (amenazas programadas) o el robo, destrucción o modificación de la información.

Observando meticulosamente las amenazas, así como las vulnerabilidades de los sistemas de información parece evidente que existen una serie de patrones básicos (objetivos), que cualquier sistema de seguridad informática debe asegurar sobre la información que se encarga de proteger. La intensificación de estos parámetros básicos va a depender del campo donde vayan a ser empleados y se agrupan fundamentalmente bajo los términos de confidencialidad, integridad, disponibilidad y autenticidad.

Confidencialidad

La confidencialidad es el servicio de seguridad, o condición, que asegura que la información no pueda estar disponible o ser descubierta por personas o entidades no autorizadas. Así pues, cuanto más complicado sea acceder al contenido de la información, mayor será el grado de confidencialidad del sistema en concreto. Fundamentalmente existen dos formas básicas de garantizar la confidencialidad en un sistema:

- Restringiendo el acceso a la información, ya sea por software o por seguridad física, tanto en sistemas cerrados como en sistemas abiertos.
- Mediante la criptografía. Este método nos va a garantizar confidencialidad porque el hecho de cifrar la información implica que individuos no autorizados (no poseedores de la clave para descifrar la información) puedan acceder al contenido de la misma.

Disponibilidad

La disponibilidad es la condición de seguridad que permite que la información esté en el lugar, momento y forma requeridos por el usuario autorizado. Para ello el sistema, tanto hardware como software, debe ser capaz de atender las peticiones que recibe con la máxima inmediatez posible y, en caso de fallo, recuperarse rápidamente.

Lo opuesto a disponibilidad se denomina “denegación de servicio” (denial of service) y conlleva que los usuarios no puedan obtener del sistema la información deseada, bien por falta o saturación de los recursos o bien por una caída del sistema. La falta de disponibilidad puede deberse a un ataque al sistema pero, en general, se debe al mal diseño del mismo.

Integridad

La integridad es el servicio de seguridad que garantiza que la información sólo pueda ser modificada por quien está autorizado y de manera controlada. No se trata ahora de asegurar la privacidad de la información, sino de poder asegurar que la información de la que disponemos es realmente la que

debe ser y que ésta no ha sido modificada por usuarios no autorizados. Para cumplir este objetivo también tendrá que tenerse en cuenta las modificaciones no intencionadas que puedan ser realizadas por usuarios autorizados, ya sea por error o por desconocimiento.

Para garantizar la integridad disponemos de dos herramientas fundamentales:

- Copia de seguridad. Consiste en almacenar aquella información importante para el usuario, tanto archivos como configuraciones, en medios de almacenamiento tecnológicamente disponibles.
- RAID. Es un conjunto de unidades de disco que aparecen lógicamente como si fueran uno solo. Así los datos, distribuidos en bandas, se dividen entre dos o más unidades. Esta técnica incrementa el rendimiento y proporciona una redundancia que protege contra el fallo de uno de los discos de la formación. Existen varios niveles RAID, desde el nivel 0, en el que los datos se dispersan en varias unidades pero no hay redundancia, lo que supone un gran rendimiento pero seguridad nula. Pasando por el nivel 1 o mirroring (espejo) en el que los datos se escriben duplicados en distintas unidades, de manera que no se incrementa el rendimiento pero sí la seguridad, siendo uno de los más utilizados. Hasta niveles superiores que son una combinación de los conceptos anteriores y buscan aumentar la seguridad y el rendimiento simultáneamente.

Autenticidad

La autenticidad es la condición de seguridad que consiste en poder tener certeza del origen de la información. Una información no será segura si no conocemos con certeza de donde proviene. Por tanto la seguridad informática, deberá establecer una serie de métodos que nos permitan garantizar que la información enviada o recibida sea realmente de quien debe ser. En la mayoría de los casos la autenticidad se considera una parte de la integridad, por ello una de las formas más extendidas de asegurar el origen de la información es el uso de huellas digitales que se añaden a los mensajes en el origen. Este método nos permite asegurar que un usuario es quien dice ser y que lo enviado es igual a lo recibido, ya que la firma es recalculada en el destino. Por último indicar que en el concepto de autenticidad juega un papel importante lo que es conocido como el no repudio. Esta propiedad permite asegurar que cualquier entidad que envía o recibe información, no puede alegar ante terceros que no la envió o la recibió. Esta forma de garantizar la autenticidad de la información es difícil de lograr ya que depende de las personas y no de unos mecanismos infalibles que nos resuelvan el problema en concreto.

Análisis de riesgos

El Análisis de riesgos es un método formal que permite investigar los riesgos a los que se encuentra expuesto el sistema, así como su probabilidad de ocurrencia y el impacto de los mismos con el objetivo de controlarlos tomando las medidas oportunas. Para ello se debe calcular el valor del sistema, identificar sus vulnerabilidades, las posibles amenazas y verificar que se cumple la ecuación de equilibrio.

Valor del sistema

Para calcular el impacto que pueden tener ciertos riesgos sobre nuestro sistema, el primer paso es evaluar el valor del mismo. Este valor puede desglosarse en dos partes fundamentales:

- Valor intrínseco. Es la parte más sencilla de valorar, ya que en la mayoría de los casos podemos establecer un valor para cada uno de los recursos incluidos en el sistema informático. Aquí debemos tener en cuenta el valor del hardware, del software, de la información personal almacenada y del esfuerzo y material invertido para obtener los datos.
- Costes derivados. Son bastante más difíciles de enumerar y cuantificar que los anteriores, ya que dependen de las consecuencias de la materialización de los riesgos y pueden ser muy distintos según el tipo de sistema con el que tratemos. Entre ellos podemos considerar el valor de sustituir el hardware, el valor de sustituir el software, el valor de los resultados perdidos y el coste de regenerar la información personal.

Vulnerabilidades. Clasificación

Las vulnerabilidades del sistema son aquellos puntos en los que el sistema es débil y, por tanto, es susceptible de ser dañado o atacado. Algunos tipos de vulnerabilidad de un sistema son los siguientes:

- Vulnerabilidades naturales o físicas a las que está expuesto un determinado sistema (climatología…)
- Vulnerabilidades de hardware y software (saturación de recursos, mal funcionamiento del S.O…)
- Vulnerabilidades de comunicación. Se trata de la vulnerabilidad más amenazada. Cuanto mayor sea el número de individuos que puedan atacar nuestro sistema, mayores son las probabilidades de saturación o violación de los datos de los que disponemos.
- Vulnerabilidades humanas. Referentes al administrador y a los usuarios del sistema.

Amenazas. Clasificación

Las amenazas son posibles peligros a los que queda expuesto el sistema y están claramente en función de las vulnerabilidades del mismo. Cuanto mayor sea una vulnerabilidad de nuestro sistema más probable será el éxito de un ataque (amenaza). Las amenazas se clasifican según el efecto causado en el sistema en:

- Intercepción. Se dan cuando un determinado individuo o programa consigue el acceso a una información a la que no está autorizado, produciéndose en consecuencia una violación del principio de la confidencialidad. Las intercepciones son las amenazas más difíciles de detectar ya que, en la mayoría de casos, no se suele realizar una modificación de los datos que pertenecen al sistema. Un ejemplo es la escucha de una línea de datos.
- Modificación. No solo se accede a la información no autorizada sino que se modifica la información sin autorización. Esta amenaza rompe con el principio de la integridad que indica que todo sistema de información seguro debe garantizar que la información que contiene no sea susceptible de modificaciones ajenas. Son modificaciones cambiar el contenido de una base de datos o cambiar líneas de código en un programa.
- Interrupción. Interrumpir mediante algún método el funcionamiento del sistema. Esta amenaza, que puede ser intencionada o accidental, violaría el principio de disponibilidad. En el caso de

que el usuario, simultáneamente al ataque, realizara una petición de un dato o de un recurso no se le podría facilitar. Un ejemplo sería saturar la memoria del sistema operativo.

- Generación. Cambiar la composición de los datos del sistema añadiendo elementos. Esta amenaza rompe con el principio de integridad al igual que el de modificación. Como ejemplo de generación nos encontramos a los virus que añaden fragmentos de código a los programas para llevar a cabo su labor.

Desde el punto de vista del origen de las amenazas, estas se clasifican en:

- Amenazas naturales o físicas. Son las que ponen en peligro los componentes físicos del sistema. En ellas podemos distinguir por un lado los desastres naturales, como las inundaciones, rayos o terremotos, y las condiciones medioambientales, tales como la temperatura, humedad, presencia de polvo…

- Amenazas involuntarias. Son aquellas relacionadas con el uso descuidado del sistema por falta de entrenamiento, de concienciación sobre la seguridad o por desconocimiento del sistema. Entre las más comunes están anotar el password en un sitio visible o borrar parte de la información sin querer.

- Amenazas intencionadas. Son aquellas procedentes de personas que pretenden acceder al sistema para borrar, modificar o robar la información (Hackers).

Ecuación de equilibrio

La ecuación de equilibrio permite estudiar si realmente nos interesa realizar una implantación de un sistema de seguridad informática realizando un análisis de costes. En este análisis van a influir los siguientes costes o valores fundamentales:

- Valor del sistema informático (recursos e información a proteger).

- Coste del ataque, es decir, los medios necesarios para romper las medidas de seguridad establecidas en nuestro sistema.

- Coste de las medidas de seguridad.

De esta forma, todo sistema de seguridad correctamente diseñado, deberá tener la siguiente disposición con respecto al valor de las variables presentadas:

Coste del ataque > Valor del sistema informático > Coste de las medidas de seguridad

Así pues, si el coste del ataque es mayor que el valor de lo que deseamos proteger, tendremos un punto a nuestro favor, ya que este hecho reduce la probabilidad de que se produzca un ataque con estas condiciones. Por otra parte, que el valor del sistema informático sea mayor que el coste de implantación del sistema de seguridad significa que no debe costar más proteger la información que la información protegida. Si esto ocurriese, nos resultaría más conveniente no proteger nuestro sistema y volver a obtener la información en caso de pérdida.

Gestión de riesgos

La gestión de riesgos consiste en tomar las medidas oportunas y cumplirlas para garantizar la seguridad del sistema. Consta de dos partes, la construcción de una política de seguridad y las

auditorías que verifican el cumplimiento de la política de seguridad.

Política de seguridad

La política de seguridad consiste en realizar un análisis profundo del sistema que va a ser protegido y del que se desea que en todo momento se cumplan los objetivos de la seguridad. Para ello tendremos que conocer de qué disponemos, realizar una distribución de las responsabilidades de cada uno de los usuarios, así como implantar medidas que en mayor o menor grado (en función del análisis de riesgos realizado) nos permitan garantizar la seguridad de la información.

Contramedidas. Clasificación

Las medidas que nos permitirán garantizar la seguridad de nuestro sistema frente a las amenazas son las contramedidas. Éstas se establecen bajo un sistema de capas organizadas jerárquicamente, de forma que unas contramedidas condicionan a otras. Se agrupan fundamentalmente en:

- Contramedidas físicas. Aplican mecanismos para impedir el acceso directo o físico no autorizado al sistema. También protegen al sistema de desastres naturales o condiciones medioambientales adversas. Se trata fundamentalmente de establecer un perímetro de seguridad en nuestro sistema. Un ejemplo de esta seguridad física sería poner directamente un guardia de seguridad delante de nuestro sistema.
- Contramedidas lógicas. Incluye las medidas de acceso a los recursos y a la información y al uso correcto de los mismos, así como a la distribución de las responsabilidades entre los usuarios. Se refiere más a la protección de la información almacenada. Un ejemplo de este tipo de contramedidas sería el establecimiento de comunicaciones con elementos criptográficos (certificados, firmas digitales…) o la realización de copias de seguridad.
- Contramedidas administrativas. Gestión correcta de la política de seguridad del sistema. Entre las medidas a tomar destacan la creación de documentación para informar a los individuos sobre la política de seguridad del sistema, así como establecer un plan que permita formar a los usuarios para evitar amenazas de tipo involuntario.
- Contramedidas legales. Se refiere más a la aplicación de medidas legales para disuadir al posible atacante o para aplicarle algún tipo de castigo a posteriori. Si los castigos por el acceso a elementos confidenciales es castigado de forma drástica el índice de ataques se reduce en un alto porcentaje. Se trata de disuadir al atacante.

Auditoría

La auditoría es una parte fundamental de la gestión de riesgos. De hecho es el elemento más importante de la seguridad, ya que es a través de ella como se garantiza el cumplimiento de la política de seguridad establecida. Las auditorías de seguridad de los sistemas informáticos permiten conocer en el momento de su realización cuál es la situación exacta de sus activos de información (información a resguardar) en cuanto a protección, control y medidas de seguridad. Para llevar a cabo una auditoría informática se utilizan diversas herramientas y técnicas:

- Cuestionarios impresos que se envían a personas concretas que el auditor cree adecuadas.
- Entrevistas en las que el auditor recoge más información, y mejor matizada, que la proporcionada por medios propios puramente técnicos o por las respuestas escritas a cuestionarios.
- Checklist. Listas de preguntas sistematizadas y clasificadas por materias.

- Trazas. Herramientas software que permiten verificar que tanto los programas del sistema como de usuario realizan las funciones previstas.

Criterios de seguridad

Son un conjunto de aspectos que deben ser tenidos en cuenta a la hora de diseñar un determinado sistema de seguridad informática. Estos aspectos se representan de forma esquemática en los principios de menor privilegio, de la oscuridad, de la defensa en profundidad, de la existencia de seguridad en caso de fallo y de la participación universal.

Principio del menor privilegio

Establece que cualquier objeto o sujeto debe tener los privilegios de uso y de acceso que le son necesarios. Este privilegio se ve claramente en sistemas UNIX. Su política de seguridad establece una serie de permisos sobre los ficheros (información) que según sea el grado de necesidad de uso del usuario se pueden ampliar o no. En el caso del administrador (*root*) dispone de todos los permisos sobre los ficheros.

Principio de la oscuridad

La ocultación de los posibles defectos y vulnerabilidades del sistema no garantiza seguridad. Esta información sigue estando ahí, únicamente que no puede ser accedida por usuarios no autorizados.

Principio de la defensa en profundidad

Establecer el mayor número de mecanismos de seguridad posibles. De este modo cualquier atacante tendrá que superar varias barreras para acceder a nuestro sistema.

Principio del eslabón más débil

El máximo grado de seguridad de un sistema es aquel que tiene su eslabón más débil. Cuando se diseña una política de seguridad o se establecen los mecanismos necesarios para ponerla en práctica, se deben contemplar todas las vulnerabilidades y amenazas. No basta con establecer unos mecanismos muy fuertes y complejos en algún punto en concreto, sino que hay que proteger todos los posibles puntos de ataque.

Principio de la existencia de seguridad en caso de fallo

En caso de que cualquier mecanismo de seguridad falle el sistema debe seguir en un estado seguro.

Principio de la participación universal

En caso de detectar algún fallo en el sistema el usuario deberá comunicárselo rápidamente al administrador para que lo subsane inmediatamente. Así se puede lograr dar una mayor robustez al sistema de seguridad.

Criptografía

La criptografía es la ciencia que nos va asegurar, en principio, la confidencialidad de la información estableciendo una serie de patrones a partir de los cuales la información no es accesible a individuos no autorizados ajenos a ésta. Por tanto, el secreto de esta ciencia consiste en que la información no desaparece, sí su significado. Entre los patrones más importantes, se encuentra el establecimiento de claves privadas que son las que le dan significado. Si la clave se pierde o es expuesta a individuos ajenos, el método criptográfico ya no resulta eficiente.

Criptoanálisis

Es el estudio de sistemas criptográficos (criptosistema) con el objetivo de encontrar sus debilidades y, así, poder obtener el sentido de una información cifrada, sin acceso a la información secreta requerida para obtener este sentido normalmente. Típicamente, esto se traduce en conseguir la clave secreta. Para lo cual se puede desde sabotear los canales de comunicación, hasta estudiar altos contenidos de texto cifrado para obtener información sobre estos (estadíticas…).

Criptosistema

Un criptosistema es una quíntupla formada por cinco conjuntos (M,C,K,E,D), donde:

M → Conjunto de los mensajes en claro, sin cifrar.

C → Representa el conjunto de todos los posibles mensajes cifrados.

K → Representa el conjunto de claves que se pueden utilizar en el sistema.

E → Conjunto de las transformaciones de cifrado o familia de funciones que se aplica a cada elemento de M para obtener C. Existe un transformación diferente para cada valor de la clave k.

D → Conjunto de las transformaciones de descifrado o familia de funciones que se aplica a los elementos de C para obtener los mensajes iniciales.

Así pues cualquier critpsosistema debe cumplir que al aplicar las transformaciones de cifrado con una clave k sobre un mensaje M debemos obtener C de tal forma que contenga la mínima información posible sobre M y, al aplicar el conjunto de las transformaciones de descifrado sobre el mensaje C con la misma clave k, podamos obtener M sin ningún tipo de problema. Si un criptosistema no cumple estas características no será válido para lograr los objetivos deseados.

Por otra parte, si ciframos el mensaje con una clave k y luego lo desciframos con la misma clave k y no obtenemos lo que teníamos al principio, además de fallar, el sistema no cumple con los principios básicos de la seguridad informática (integridad de la información).

Criptosistemas simétricos o de clave privada

En estos sistemas se utiliza una llave k que sirve tanto para cifrar como para descifrar y cuya longitud es fundamental. Elegir mal la longitud de la clave da lugar a problemas, ya que si la longitud de la clave es corta el método no es seguro y si la longitud de la clave es larga resulta incómodo. Otro problema presente en este tipo de criptosistemas es que ambas partes deben conocer la clave k, por lo que habrá que establecer una forma de transmisión de datos que nos asegure mantener la privacidad de la clave que constituye el secreto del sistema. Como ventaja, indicar que el coste computacional no es excesivamente elevado.

Criptosistemas asimétricos o de clave pública

En estos sistemas se emplean dos claves distintas, una para cifrar y la otra para descifrar, lo que evita tener que transmitir la clave privada de forma segura, como ocurría en los criptosistemas simétricos o de clave privada, asegurándose la confidencialidad. La clave de cifrado es pública, conocida por todo el mundo, la de descifrado privada y por tanto "idealmente" sólo conocida por su propietario. Ambas constituyen un par de claves (K,P), siendo K la clave privada y P la pública. La principal desventaja de los criptosistemas de clave privada es el coste computacional de los métodos empleados para realizar las transformaciones de cifrado y de descifrado, frente a los criptosistemas de clave pública. Además, un individuo que disponga de la clave pública (que puede ser cualquiera) dispone de cantidades infinitas de texto cifrado y, por tanto, de información de los mensajes cifrados sobre los mensajes iniciales.

Cifrado elemental

Dentro de este grupo de algoritmos de cifrado se encuentran aquellos algoritmos que se habían utilizado históricamente, antes de la aparición de sistemas computacionales. Su importancia radica en que la combinación de múltiples de ellos pueden dar lugar a métodos criptográficos altamente eficientes y difíciles de criptoanalizar, por ejemplo el DES que es una combinación de métodos de difusión y confusión. En la actualidad no disponen de credibilidad por sí solos, ya que se pueden criptoanalizar con facilidad por medio de métodos estadísticos y de la teoría de la información. Los métodos criptográficos elementales se clasifican en:

- Algoritmos de sustitución alfabética. En este tipo de métodos los elementos del texto en claro se sustituyen por otros símbolos que pertenecen a otro alfabeto.
- Algoritmos de cifrado por transposición de símbolos. Se agrupan aquí todos los algoritmos y transformaciones sobre el mensaje inicial en los que se atiende a la posición y no al contenido del mensaje en claro.

César

Consiste en sustituir los símbolos del mensaje inicial por símbolos de otro alfabeto. Este alfabeto es una función del alfabeto en el que está escrito el mensaje inicial. Esta función consiste en desplazar los símbolos del alfabeto i posiciones a la derecha. Así pues, al símbolo A le corresponde el símbolo del alfabeto resultante de desplazar A i posiciones a la derecha, al B el símbolo del alfabeto resultante de desplazar B i posiciones a la derecha y así sucesivamente.

Indicar que el algoritmo de César tenía el valor constante i = 3 y que el secreto del algoritmo no se basa únicamente en la clave, sino también en el propio algoritmo y en el hecho de estar usando un algoritmo.

Basándose en el método explicado se puede realizar la generalización del algoritmo de César. Suponemos que n es el número de símbolos del alfabeto inicial, i el desplazamiento y M la posición del símbolo del mensaje inicial en el alfabeto, así pues calcularemos la posición del símbolo cifrado en el alfabeto de sustitución con la expresión:

$C = (M + i) \bmod n$

De este modo, si consideramos el alfabeto Ascii que dispone de n = 256, con un desplazamiento de 5, y consideramos que el mensaje inicial es "H", el resultado será *posición del símbolo cifrado = (posición("H")+ 3) mod 256* , es decir, c = 93, que se corresponde con el símbolo "M", es decir el símbolo del alfabeto inicial resultante de desplazar 5 posiciones a la derecha el símbolo del mensaje inicial.

Vigenère

Se trata de un algoritmo de sustitución polialfabética, en este caso el cifrado de un determinado símbolo depende no solo de la posición que ocupa en un determinado alfabeto sino también de la posición que ocupa dentro del texto nativo que se procede a cifrar. La clave no está formada por un único desplazamiento, sino por d desplazamientos $\{i_0, i_1, i_2, i_3 \dots i_{d-1}\}$.Así pues para cifrar un texto se agrupa el mensaje inicial en bloques de d símbolos, aplicándole a cada uno de los símbolos de cada uno de los bloques desde j = 0 hasta j = d-1 la siguiente expresión:

$C= (M_j + i_j) \bmod n$

donde M_j es la posición que ocupa el símbolo j-ésimo del bloque de d caracteres en el alfabeto j-ésimo.

El algoritmo de Vigenère no es más que una aplicación reiterada del algoritmo de césar con alfabetos y claves variables.

Transposición

Dentro de este grupo de algoritmos encontramos:

- Algoritmo de transposición para bloques de n caracteres. Este algoritmo se fundamenta, al igual que todos los métodos de transposición, en cambiar el orden de los mensajes del texto inicial, siendo el fruto de esta variación del orden el mensaje cifrado. Así pues es necesario definir la longitud de los bloques y la permutación de los n elementos que forman los mismos. Esta permutación va a indicar de qué forma se va a variar el orden de los símbolos del mensaje inicial. De tal forma que el contenido de la posición i-ésima indica, que en el mensaje cifrado la posición i-esima debe ir en la posición que indica el contenido de la permutación en dicha posición. Así pues, si por ejemplo nos encontramos con el mensaje “Hola” y con la permutación {1, 3, 2, 4} el mensaje cifrado sería “Hloa”. Para descifrar algo cifrado con este método, se aplica la permutación de n elementos en orden inverso.
- Algoritmo de transposición por columnas. En este método se dispone el texto por filas de una determinada longitud, rellenándose el final de la última fila con un carácter cualquiera, por ejemplo el espacio. El texto cifrado se obtiene leyendo la matriz resultante por columnas. La clave de descifrado es simplemente el número de columnas utilizado.

Criptosistemas de llave privada

El cifrador ideal

El cifrador ideal es el que proporciona como mensaje cifrado el resultado de realizar una operación XOR (operación bit a bit) entre el mensaje en claro y la clave. De esta forma, es necesario que la clave tenga una longitud mayor o igual que el mensaje a cifrar. El descifrado del mensaje consiste en realizar de nuevo una XOR del mensaje cifrado con la misma clave, ya que:

$A \oplus K = B$

$B \oplus K = A$

Lo que convierte en ideal a este método es que resulta imposible averiguar el mensaje en claro si no se dispone de la clave, ya que cualquier mensaje cifrado de cierta longitud puede descifrarse como cualquier otro mensaje de la misma longitud dependiendo de la clave utilizada, que si se mantiene correctamente en secreto no se dispone de ella.

Cifradores de bloque

Son aquellos que operan sobre grupos de bits de longitud fija (bloques) aplicándoles una transformación invariante. Al cifrar toman un bloque del texto en claro como entrada que, combinándolo con la clave, produce un bloque de igual tamaño de texto cifrado. El descifrado es similar: se ingresan bloques de texto cifrado y se producen bloques de texto en claro.

Redes de Feistel

Es un método de cifrado en bloque con una estructura particular que debe su nombre a Horst Feistel. Un gran número de algoritmos de cifrado por bloques lo utilizan, siendo el más conocido el algoritmo Data Encryption Standard (DES). Las redes de Feistel presentan la ventaja de ser reversibles por lo que las operaciones de cifrado y descifrado son idénticas, requiriendo únicamente invertir el orden de las subclaves utilizadas.

Su funcionamiento consiste en una iteración inicial en la que se divide la longitud del bloque mínimo de cifrado, también conocido como n bits, en dos partes que se suelen nombrar L y R. De este modo, L almacena o consta de los n/2 bits de mayor peso y R de los n/2 bits de menor peso. Así pues la red de Feistel, se define como un cifrado iterativo en el que la salida de cada una de las rondas o iteraciones se usa como entrada para la siguiente según la siguiente relación:

si $i < n$:

$L_i = R_{i-1}$

$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$

$L_n = L_{n-1} \oplus f(R_{n-1}, K_n)$

$R_n = R_{n-1}$

Esta representación nos muestra que en la iteración i la entrada de datos depende de la salida de la iteración anterior. La entrada está constituida por el contenido de los registros L_i y R_i. El contenido de L_i pasa a ser el contenido del registro R_{i-1}, de la fase anterior, y la entrada del registro R_i , es una función no solo de L_{i-1}, sino también de una transformación conocida como función “f”. Esta función, que depende de cual es el algoritmo de cifrado utilizado, recibe un bloque de de n/2 bits y una clave K_i (la clave correspondiente a la ronda i) y nos devuelve otro bloque de n/2 bits transformado. La salida de esta función f, se opera a través del operando XOR con el contenido del registro L_{i-1}. Indicar que el secreto de las redes de Feistel, gracias a las propiedades de XOR, es que a la hora de descifrar no hace falta rediseñar otra disposición de los registros, sino que consiste en realizar las transformaciones desde la iteración n hasta la 0 de la siguiente forma:

$R_{j-1} = L_j$

$L_{j-1} = R_j \oplus f(L_j, K_j)$

DES

El DES (Data Encription Standard) es el algoritmo más popular basado en redes de Feistel, ya que ha sido reconocido como el algoritmo estándar por las agencias americanas y dispone de una gran fortaleza. El origen del DES se debe a una petición realizada en 1973 por el NBS (National Bureau of Standards) a distintos fabricantes para someter criptosistemas que pudieran servir como base a un estándar de cifrado de textos reservados no clasificados.

Se basa en el algoritmo desarrollado por IBM, conocido con el sobrenombre de LUCIFER, que utilizaba claves de 128 bits de longitud y que dotaba de una gran seguridad. Así pues del intento de creación de un estándar nació el DES. Tras ser analizado el algoritmo LUCIFER por expertos de la NSA (National Security Agency), redujeron las claves de 128 bits que usaba este a 56 bits naciendo así el nuevo criptosistema de llave privada. El funcionamiento del DES, atendiendo a los parámetros que intervienen en la red de Feistel (número de bits, función f y número de iteraciones), consiste en codificar bloques de n=64 bits empleando claves de una longitud de 56 bits. Como es una red de Feistel de 16 rondas, dispondrá de 16 claves para las transformaciones de cifrado más dos permutaciones, una que se aplica al principio sobre el bloque inicial de 64 bits, y otra que se aplica al contenido de los registros L16 y R16 resultantes de las 16 rondas de la red de Feistel. Para calcular la clave de cada ronda se parte de una clave inicial de 64 bits, a la que se aplica una permutación inicial que reduce la clave a 56 bits, y posteriormente se separa en dos registros de 28 bits cada uno, aplicando desplazamientos de bits a cada una de las dos mitades. Sobre cada uno de los registros desplazados se realiza una elección permutada, que nos permite obtener una clave de 48 bits para cada una de las 16 elecciones permutadas.

Por último queda describir la función f. Recordemos que la función f recibe un dato de 32 bits y una clave y devuelve un dato transformado de 32 bits. La función f, dispone de dos permutaciones, una inicial y otra final. La permutación inicial, además de cambiar el orden de los caracteres, convierte el dato de 32 bits en uno de 48 y la permutación final recibe un dato de 32 bits procedente de sucesivas transformaciones y nos devuelve el valor final de 32 bits transformado. El resultado de la permutación inicial se opera XOR con la clave utilizada en la ronda i y este resultado es pasado a través de 8 cajas de sustitución multialfabética que, a partir de los 48 bits, nos devuelven los 32 bits de entrada de la permutación final.

Cabe indicar que, debido a que la longitud de las claves DES es de únicamente 56 bits, se demostró que es posible realizar una ataque por fuerza bruta al sistema y obtener la clave en un tiempo razonable. Para solucionar este problema se emplea, lo que es conocido como DES3 que consiste en una combinación de cifrados y descifrados con 3 claves DES de la siguiente forma:

$E_{k3}(D_{k2}(E_{k1}(m)))$

Esto se puede realizar gracias a que el conjunto de claves del criptosistema DES no cumple con una estructura de grupo, es decir, que dadas dos claves no existe una clave equivalente a otras claves. Esto nos permite aumentar la longitud de las claves del DES, lo que hace disminuir las posibilidades de que un ataque por fuerza bruta tenga éxito.

Modos

Los modos son las diferentes formas que existen de encadenar los bloques obtenidos en un cifrador por bloques. Algunos de ellos son: ECB, CBC y CFB .

ECB

ECB es el modo más sencillo e intuitivo para cifrar mensajes y datos de texto en claro por bloques con una longitud mayor que la unidad mínima de cifrado. Llamamos Mi a cada uno de los bloques de n bits en los que se puede dividir el texto en claro. Así pues, consiste en dividir el texto en claro en bloques de n bits (Mi) aplicando las transformaciones de cifrado correspondientes a cada uno de estos bloques cifrados, obteniendo así el mensaje cifrado (C) como la concatenación del resultado de cada una de estas transformaciones (Ci). Indicar que cada uno de los bloques de texto en claro se codifica con la misma clave.

Si hablamos en el caso del DES, cifrar el texto en claro con una clave de 56 bits K, según el modo ECB, consiste en dividir el texto en bloques de 64 bits, aplicando a cada uno de los bloques de texto en claro con la clave K las transformaciones de cifrado pertinentes.

La principal ventaja de este método es que funciona muy bien en canales en los que puede ser probable que se dé un fallo en la codificación de algunos de los bloques, de tal forma que si se consuma el error no es necesario cifrar todo el texto otra vez, sino que basta con cifrar el bloque de mensaje inicial en el que se ha producido el error de cifrado. Por otra parte, presenta varios inconvenientes, entre los que se encuentran el hecho de que pueden eliminarse porciones del texto cifrado sin que se note, esto es, puede ocurrir que si se conocen las características y posición de cierta información en el mensaje inicial, ésta puede eliminarse del texto cifrado sin impedir un correcto descifrado del mismo. Otro inconveniente importante es que si el mismo bloque de texto en claro aparece más de una vez en el mensaje, éste siempre produce el mismo texto cifrado, por lo que no se debe usar en caso de querer cifrar información repetitiva.

CBC

CBC es un modo en el que el cifrado de un bloque del mensaje (Mi) depende no sólo de Mi, sino del cifrado de los (i-1) bloques anteriores. Para ello el CBC, divide el trabajo a realizar en fases o iteraciones. En cada una de estas fases, realiza una operación XOR del bloque que queremos cifrar (Mi) con el resultado de las transformaciones de cifrado con la clave k de la fase anterior. Existe por tanto una retroalimentación de datos en cada una de las fases, siendo necesario para una determinada fase, la ejecución de las fases anteriores. Como en la primera fase (cifrado de M1), no existen resultados de la ejecución de la fase anterior, se utiliza un vector de inicialización o un registro que almacene un valor inicial de n bits (longitud de los bloques).

En cuanto a las ventajas del CBC nos encontramos con que el uso de esta filosofía de retroalimentación hace imposible sustituir un bloque de texto cifrado como una posible medida de criptoanálisis y, además, elimina el problema que tiene el modo ECB con la información repetitiva. Una posible desventaja, si el contenido de VI es fijo, es que mensajes iguales se codifican de la misma forma, y lo que es más grave, mensajes iguales hasta ciertos puntos, se codifican de las misma forma hasta alcanzar dichos puntos. Una forma de solucionar este problema consiste en almacenar en VI un dato de n bits aleatorio. Así pues, se logra que comparando dos mensajes cifrados que guardan ciertos parecidos (misma cabecera por ejemplo) , sea imposible obtener información relacionada con el criptosistema.

Por último, indicar que el CBC permite cambiar la filosofía de los algoritmos de cifrado por bloques a algoritmos de cifrado por flujo o stream.

CFB

El CFB es un modo de operación que permite obtener unidades de mensaje cifrado inferiores a n (longitud de un bloque). Suponiendo que p es la longitud en bits de la nueva unidad de cifrado (p debe ser divisor de n y por tanto menor). En este modo se mantiene un registro de n bits donde se van almacenando por la derecha el resultado de operaciones de cifrado para la iteración siguiente.
Se cifran bloques sucesivos de n bits de dicho registro. El byte más significativo del resultado se

combina (XOR) con el siguiente bloque de p bits del mensaje inicial para dar lugar al bloque de cifrado de p bits a transmitir. Además este último byte se reintroduce en el registro provocando un desplazamiento de su contenido. Al iniciar el cifrado no existen resultados de una ejecución anterior por lo que es necesario un vector de inicialización, pero como ocurría en el CBC, es necesario que el vector contenga un dato aleatorio para evitar que partes iguales de mensajes se codifiquen igual.

El CFB es muy útil si se necesita el resultado o fracciones del mensaje cifrado al mismo tiempo en el que se están cifrando las mismas. Este método es muy utilizado por ejemplo para cifrar caracteres según un stream o flujo de datos de forma que, a la vez que se escribe el texto original, se obtiene de forma simultánea el mensaje cifrado. Por otra parte, al necesitarse un cifrado por cada byte, no por cada bloque, resulta computacionalmente muy lento (es n veces más lento que el CBC). Por último, indicar el CFB permite cambiar la filosofía de los algoritmos de cifrado por bloques a algoritmos de cifrado por flujo o stream.

Cifradores de flujo

Los cifradores de flujo son algoritmos de cifrado que pueden realizar el cifrado incrementalmente, convirtiendo el texto en claro en texto cifrado bit a bit. Esto se logra construyendo un generador de flujo de clave, es decir, una secuencia de bits de tamaño arbitrario que puede emplearse para oscurecer los contenidos de un flujo de datos combinando el flujo de clave con el flujo de datos mediante la función XOR.

RC4

El RC4 es el cifrador de flujo más popular y se utiliza en protocolos como el SSL (para proteger el tráfico de Internet) y el WEP (para proteger redes Wireless). Es muy simple y rápido, pero presenta algunas debilidades que hacen que no se recomiende su uso en los nuevos sistemas, a pesar de que algunos sistemas basados en RC4 son lo suficientemente seguros para un uso común.

Derivación de llaves desde contraseñas

Es el procedimiento por el cual se obtiene una llave a partir de una contraseña mediante la aplicación de una función. Para evitar los ataques por fuerza bruta, se realizan varias derivaciones, actualmente se recomiendan entre 1000 y 2000. La obtención de la llave usando este procedimiento es relativamente costosa, pero es esta característica la que consigue disuadir a los posibles atacantes.

Generación de números aleatorios

Los números aleatorios son sucesiones de números o bits seleccionados al azar de forma uniforme, es decir, todo número o bit tiene la misma probabilidad de ser escogido. Podemos hablar indiferentemente de números o de bits aleatorios porque la agrupación de bits da lugar a números y seleccionando determinados bits de los números obtenemos sucesiones de bits.

Resultan de gran importancia en la criptografía porque son muy útiles a la hora de generar las claves para el cifrado de mensajes gracias a la impredecibilidad de los términos de la sucesión. Pero su generación, a partir de los datos recogidos del movimiento del ratón o de la tarjeta de red por ejemplo, resulta muy lenta. Además, la transmisión y almacenaje de números aleatorios no es práctica por lo que se suelen usar sucesiones de números pseudoaleatorios.

Los números pseudoaleatorios son números generados por medio de una función (determinista, no aleatoria), que se aplica iterativamente a partir de un valor inicial, y que aparentan ser aleatorios.

Las principales desventajas de las sucesiones de números pseudoaleatorios son que a partir de un mismo valor inicial se genera la misma sucesión y que, en general, la sucesión es periódica. Esto se soluciona escogiendo generadores con periodos largos. En cualquier caso, estos "defectos" pueden resultar útiles para conseguir que la sucesión parezca aleatoria y a la vez conocer los números que se repiten periódicamente y cuál es la longitud del periodo y, también, en caso de querer repetir un experimento en las mismas condiciones.

A pesar de las desventajas mencionadas, se utilizan generadores de números pseudoaleatorios en vez de los de números aleatorios porque sólo se requiere almacenar y transmitir la semilla que genera la sucesión y, además, generan las sucesión más rápidamente.

Criptosistema de llave pública RSA

El algoritmo de cifrado RSA es el criptosistema de clave pública más extendido. Su nombre proviene de sus creadores Rivest, Shamir y Adleman, quienes lo desarrollaron en el MIT en 1978. Como cualquier criptosistema de clave pública, dispone de una clave pública P y de una clave privada K, existiendo una relación entre ambas de forma que, aunque sea conocida esta relación, no se pueda obtener K conociendo P.

Fundamentos

Números primos

Un número es primo si sólo se puede dividir por 1 o por sí mismo. Además, no existe ninguna fórmula que genere los números primos de manera que, para conseguirlo, es necesario ir probando con todos los números para obtener su factorización y así comprobar su primalidad.

Por otra parte, los algoritmos existentes para factorizar un número en sus factores primos se basan en prueba y error, lo que hace que factorizar un número cuyos factores son primos muy grandes sea computacionalmente inabordable.

Sin embargo, se puede saber si un número es primo mediante el test probabilístico de Rabin-Miller. Si el número pasa el test, la probabilidad de no ser primo es de 0.25. Si lo pasa dos veces, es de 0.25*0.25 y así sucesivamente. De esta forma, si se aplica el test muchas veces y el número pasa el test, la probabilidad de que no sea primo es prácticamente cero.

Primos fuertes

Un número primo es fuerte si cumple las siguientes condiciones:

- n-1 tiene algún factor grande
- (n-1)/2 también es primo

Aritmética modular

La aritmética modular permite, trabajando con números grandes, estar siempre dentro del mismo conjunto de números y se basa en que, dada una división entera, entre a y n, se cumple por definición que a = k*n + b.

Decimos que a y b son congruentes ("iguales") módulo n si para todo k, a = b + k*n, y lo expresamos como a = b (mod n). Además, si a y n no comparten factores primos (son primos entre sí), existe la inversa de a módulo n, es decir, existe un a^{-1}, tal que $a*a^{-1}$ = 1 (mod n). Obsérvese que ello significa

que $a*a^{-1} = 1 + k*n$ para todo k.

La inversa de un número se puede calcular mediante una fórmula algebraica, pero resulta inviable porque necesita de la descomposición en números primos. Con este fin se utiliza el algoritmo extendido de Euclides que dice que si n es primo, se cumple que $a^{n-1}=1 \pmod n$ y si $n = p*q$, y a no es divisible ni por p ni por q, se cumple que $a^{(p-1)*(q-1)} = 1 \pmod n$.

Generación de llaves

Para generar las llaves del RSA seguimos los siguientes pasos:

- Generamos dos números grandes p y q primos y, generalmente, fuertes. Para ello generamos dos números aleatorios, nos aseguramos de su tamaño poniendo su bit más significativo a 1 y de que sean impares, poniendo su bit menos significativo a 1, y aplicamos el test de primalidad.
- Sea $n = p*q$, de forma que $n > 2^{512}$ para evitar problemas de seguridad, ya que podría ser susceptible de ser factorizable por fuerza bruta, y $n < 2^{4096}$ para evitar posibles problemas por falta de soporte. En la práctica se usa $2^{1024} <= n <= 2^{2048}$.
- Elegimos un número e primo pequeño, normalmente 65537. Al ser primo, existe su inversa módulo cualquier número, en particular, calculamos d, tal que $d*e = 1 \pmod{(p-1)*(q-1)}$ con el algoritmo extendido de Euclides, es decir que $d*e = 1 + k*(p-1)*(q-1)$ para todo k.

El par (e, n) es la llave pública RSA, siendo e y n públicas. El par (d, n) es la llave privada RSA, siendo d secreta y n pública. Aunque n es pública no se pueden averiguar a partir de ella p y q, por lo que son secretas y tampoco se puede obtener d, la inversa de e, porque para ello necesitamos conocer p y q.

Cifrado y autenticidad

El cifrado de información consiste en realizar la siguiente operación:

$c = m^e \pmod n$

Y el descifrado consiste en realizar la siguiente operación:

$m = c^d \pmod n$

Ello se debe a que $c^d = (m^e)^d = m^{e*d} = m^{1 + k*(p-1)*(q-1)} = m*(m^{(p-1)*(q-1)})^k = m*1^k = m \pmod n$. Ambas operaciones muestran que, la relación entre e y d, hace posible que todo lo cifrado por la clave pública, pueda ser descifrado por la clave privada d, asegurándose así la confidencialidad de la información transmitida. Además, se pueden intercambiar las llaves de forma que se cifre con la llave privada d y se descifre con la pública e, garantizándose de este modo la autenticidad de la información transmitida:

$c = m^d \pmod n$

$m = c^e \pmod n = (m^d)^e \pmod n$

Indicar que tanto las operaciones de cifrado como las de descifrado, debido a la necesidad de la realización de operaciones de exponenciación, implican una elevada actividad computacional. Esta es

la razón por la que en los algoritmos de cifrado asimétricos el cifrado y descifrado se realiza de forma muy poco eficiente. En consecuencia, si se quiere cifrar un mensaje de longitud mayor que n es necesario trocearlo en bloques de tamaño n, pero esto resulta excesivamente lento.

Funciones hash

Las funciones hash (o funciones de resumen) son un serie de transformaciones matemáticas (suma, resta, xor, xnor...) que permiten reducir una información determinada. Resultan muy útiles en criptografía por los principios en los que se sustentan y porque permiten realizar las operaciones de cifrado y descifrado mucho más rápido que aplicando sucesivas transformaciones con un algoritmo de clave pública. Estos principios son:

- La entrada de una función hash es de un tamaño cualquiera. La salida es siempre del mismo tamaño.
- Dado un mensaje y su correspondiente resumen, cualquier cambio que se realice en el mensaje original, por mínimo que sea, produce cambios importantes en el resumen de dicho mensaje.
- Dado el resumen de una determinada información no es posible obtener la información que lo generó.
- Dado el resumen de una determinada información, ha de ser altamente complicado obtener una determinada información que obtenga el mismo resumen. Es decir, la probabilidad de colisión debe ser muy baja. Esto se consigue en la práctica haciendo que la cardinalidad del conjunto de resúmenes sea muy alta.

Una función que cumpla estas condiciones también se denomina de huella digital, ya que dado un mensaje nos proporciona su huella. Algunas de las funciones de resumen o de huella digital, junto con el tamaño del mensaje que generan son:

Función hash	Tamaño del mensaje
md2	128
md4	128
md5	128
sha1	160
mha2	256 / 512

Indicar que el mínimo razonable de la salida de una función hash debe ser de 100 bits. En caso contrario, fallarían algunos de los pilares básicos, sobre los que se apoya el funcionamiento de las funciones de resumen.

Cabe destacar la importancia de estas funciones por sus múltiples aplicaciones. Se utilizan en redes P2P para comprobar la validez de las partes descargadas y también en seguridad para garantizar la autenticidad de los mensajes.

Usos de la criptografía de llave pública

La criptografía de llave pública permite tanto firmar los mensajes como cifrarlos, pero por sí sola presenta algunos problemas. Las operaciones de cifrado y descifrado de los algoritmos de llave pública permiten dotar de confidencialidad al mensaje, ya que el emisor lo cifra con la llave pública del receptor y, así, sólo este puede descifrarlo. Pero estas operaciones resultan bastante costosas computacionalmente, por este motivo se combina el algoritmo de llave pública con uno de llave

privada.

Por otra parte, si el emisor cifra el mensaje con su llave privada, el receptor podrá verificar la autenticidad del mensaje si posee la llave pública del emisor. Pero si no la tiene, además de no poder comprobar la autenticidad, tampoco podrá leer el mensaje, con lo que se dará un problema de disponibilidad. Para evitar este problema se combina el algoritmo de llave pública con la firma electrónica.

Cifrado de mensajes

Los algoritmos de llave pública permiten establecer una comunicación confidencial entre dos individuos Ay B. Basta para ello con que A cifre el mensaje con la llave pública de B (P_b), de forma que, si el algoritmo funciona correctamente, solo el que posea la clave privada asociada a P_b, es decir, K_b, podrá descifrar el mensaje correctamente. Así pues, B para recuperar la información inicial deberá realizar las transformaciones de descifrado con su clave privada sobre el mensaje cifrado.

El inconveniente de este método es que las operaciones de cifrado y descifrado de los algoritmos de llave pública implican numerosas operaciones y, por tanto, un elevado tiempo de proceso. Esto hace que sea necesario el diseño de otro método que nos permita enviar la información cifrada sin la implicación de tantas operaciones innecesarias. Por ello en la práctica, para dar confidencialidad a un determinado canal de comunicación, se usa la combinación de los algoritmos de llave pública con los de llave privada. Dicha combinación consistirá en cifrar la información o el mensaje inicial, con un algoritmo simétrico de clave S, y enviar junto con dicha transformación la clave S cifrada con la clave pública de quien recibe el mensaje.

Así pues, A genera aleatoriamente una clave privada S, llamada clave de sesión, y cifra m con dicha clave. Posteriormente cifra, con el algoritmo asimétrico, S con la clave Pb y envía ambas informaciones en el mensaje. B recibe el mensaje y debe realizar las siguientes transformaciones sobre la información que recibe:

$S = D_{kb}(E_{pb}(s))$ y $M = D_s(C)$

Así pues, como se observa en dichas transformaciones, lo único que se cifra y se descifra con el algoritmo asimétrico es la clave de sesión S. Como dicha clave es una clave privada de un algoritmo simétrico, tendrá una longitud mucho más reducida de lo que podría tener el mensaje original, por lo que se reducen en alto grado las operaciones computacionales sobre la información.

También se da el caso de que, si el emisor desea enviar un mismo mensaje a varios receptores, este método nos permite reducir al máximo la información enviada y duplicar únicamente lo que sea estrictamente necesario. Para enviar un mensaje a varios receptores, A deberá mandar únicamente la clave de sesión cifrada con cada una de las claves públicas de cada uno de los receptores. Si lo que desea es guardarse el mensaje se incluirá a él mismo dentro de los receptores.

Firma electrónica

La firma electrónica es el resultado de cifrar con la clave privada del emisor el resultado obtenido al aplicar una función hash a un mensaje m. Así, se puede garantizar la autenticidad de una determinada información sin necesidad de tener que realizar las transformaciones de cifrado que conllevan los algoritmos de clave pública sobre el mensaje.

El emisor, A, transforma el resumen de m con K_a y lo envía junto con m. De esta forma, se reduce el coste computacional, ya que transformar el resultado de una función hash supone un menor trabajo

que extender dicha tarea a todo el mensaje. El resultado obtenido, $F = E_{ka}(H(m))$, es la firma electrónica. Así pues, el receptor, B, deberá en primer lugar obtener de la firma electrónica el resultado de la función de resumen calculado por A (lo llamaremos resumen del mensaje de salida) realizando $H(mi) = D_{pa}(F)$. Una vez obtenido el resultado del resumen del mensaje de salida, B deberá calcular el resumen del mensaje de llegada H(mf) (que podría no ser el mismo) y posteriormente comparar H(mi) y H(mf). Si son iguales ambos resultados, se garantiza que realmente ha sido A quien ha enviado el mensaje, lo que supone autenticidad. Además garantiza que el mensaje enviado por A es el que llega a B, lo que supone integridad. Esto es así por el principio de las funciones hash que indica que una modificación de un mensaje, por pequeña que sea, produce un cambio muy notable en el resultado de su resumen.

Como posible ataque podría plantearse no sólo modificar m sino también la firma poniendo dentro de la misma el resumen del mensaje nuevo pero, gracias a las características del modelo diseñado, esto no se puede realizar ya que, al no disponer de K_a, no podríamos volver a construir la firma electrónica.

Para garantizar autenticidad y confidencialidad al mensaje, se deben realizar dos operaciones firmar y cifrar. Se nos presentan así dos alternativas:

1. $E_{pb}(E_{ka}(m))$ primero firmar y posteriormente cifrar.
2. $E_{ka}(E_{pb}(m))$ primero cifrar y después firmar.

Ambos métodos son equivalentes, sin embargo, resulta más seguro el método 1, ya que nos aporta menos información que el método 2. Del método 2 se podría extraer, que se trata de un mensaje firmado por A obteniendo H(mi) con la clave pública de A y comprobando que coincide con el mensaje de llegada, con lo que se perdería confidencialidad.

Confianza

El problema

Los criptosistemas de llave pública aseguran la autenticidad del mensaje si el receptor conoce la llave pública del emisor correspondiente a la llave privada que ha usado para firmar el mensaje. Pero el problema es cómo asegurar que la llave pública del emisor es realmente suya y no de otro individuo. Para ello se establecen dos modelos de confianza diferentes, el horizontal y el vertical.

Horizontal

Modelo de confianza que consiste en que se dé la llave en mano, en una comunicación directa o en una comunicación entre personas conocidas (amigos).

A - B - C

Para asegurarse A de que la llave de C es válida, B firma con su llave privada la llave pública de C, entonces se la envía a A que acepta que la llave es válida porque ya confiaba en B. Se establecen varias etapas de confianza en las que todos los interesados están al mismo nivel.

El principal problema de este modelo es que requiere que el usuario sepa en todo momento lo que está haciendo porque la confianza depende de él mismo, por este motivo es un buen modelo para profesionales.

Siguiendo este modelo se ha usado el programa PGP (actualmente GPG) que permite proteger la

información distribuida a través de Internet mediante el uso de criptografía de llave pública así como facilitar la autenticación de documentos gracias a firmas digitales.

Vertical

Modelo de confianza que se basa en TTP (Third Trusted Party), en español tercero de confianza. En este caso este tercero está en un nivel superior (notario) y confiamos en todos aquellos que nos garantiza. En el caso de la criptografía el tercero de confianza es una Autoridad Certificadora.

Modelo x509

Es el modelo de ISO que define toda la utilización de criptografía de llave pública para la confianza vertical en entornos abiertos. Pero resulta insuficiente, por lo que se complementa con los documentos PKCS (Public Key Cryptographic Standard) y con los documentos RFC, que explican cómo funciona Internet y complementan al modelo x509.

Certificado

Un certificado es un objeto informático que asocia una llave pública con unos determinados datos personales. En el modelo x509 es emitido por la Autoridad Certificadora y consta de los siguientes campos:

- DN subject : campo donde se especifican los datos personales del poseedor del certificado. Se trata de un campo importante, ya que nos va a permitir diferenciar entre dos certificados de distintas CA's con un mismo número de serie. Está formado por los siguientes subcampos:
  - CN (Common name) : campo que almacena el nombre. No tiene por qué ser el nombre completo del individuo, puede ser también su alias.
  - O (Organization): nombre de la empresa.
  - OU (Organization Unit): nombre del departamento de gestión donde trabaja.
  - C (Country): campo donde se especifican las siglas del país de procedencia.
  - L (Locality): indica el pueblo (localidad) de donde procede.
  - ST (State or province): indica la comunidad o estado de donde procede.
  - Email: su valor corresponde al Email del poseedor.

Indicar con respecto al DN, que puede darse el caso de que algunos de los campos estén vacíos. También puede darse el caso que otros de sus subcampos sean opcionales. Además, al tratarse de un modelo anglosajón, no se ha tenido en cuenta un campo para el DNI que es lo que identifica a una persona en España. Por este motivo se puede incluir en el CN junto al nombre completo o como una extensión.

- DN issuer: está formado por los mismos campos que el DN subject , a diferencia de que el valor de todos sus campos se refieren a la Autoridad Certificadora que se ha encargado de emitir el certificado.
- Serial Number: campo que especifica el número de serie del certificado. El valor de este campo lo emite la Autoridad Certificadora cuando valida el certificado, es una función del número de certificados expedidos antes de la solicitud de nuestro certificado. El número de serie, va a

depender de la CA, de tal forma que dos certificados emitidos por una misma CA, no pueden tener el mismo número de serie. Este campo, es importante ya que nos va a permitir diferenciar nuestro certificado entre otros certificados emitidos por una misma CA.
- Fecha de expedición: día en el que la CA emite el certificado.
- Fecha de caducidad: día en el que el certificado deja de ser válido.
- Clave pública: contiene la clave pública del poseedor del certificado.
- Referencias a los algoritmos empleados: indica a que algoritmo de clave pública pertenece la clave pública, así como cual ha sido el algoritmo o función de resumen con el que se ha firmado.
- Firma de la CA: se genera por la CA cuando emite el certificado. La información a la cual se le aplica el resumen suele ser una combinación de algunos de los campos anteriores. Sirve para dar autenticidad a los datos que almacena el certificado. Además, si un individuo, modificara alguno de los datos del certificado, se podría detectar rápidamente esa alteración de la integridad de los datos, comprobando la validez de la firma de la CA.

ASN.1, DER, PEM

Los certificados, como objetos informáticos, precisan de la definición de su estructura y sintaxis para lo que se utiliza el ASN.1. Éste es el lenguaje de especificación de las estructuras de datos de un certificado en formato binario y permite indicar cómo especificar los contenidos.

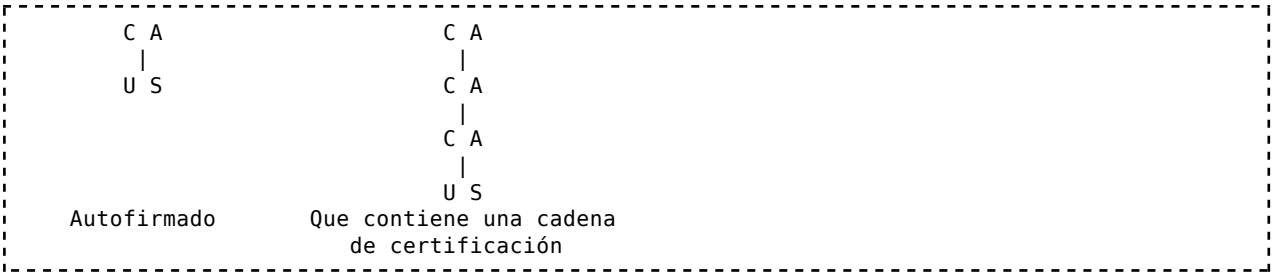
Por otra parte, las reglas de codificación definen cómo se traduce a una forma binaria la información del certificado. Estas reglas de codificación binaria son las DER (Distinguished Encoding Rules) que se basan en las Basic Encoding Rules (BER) más generales. Además, en aquellas transmisiones que no aceptan transmisiones binarias se utiliza el PEM (Privacy Enhanced Mail). Éste es una forma de representar el DER, traduciendo la forma binaria a una forma ASCII utilizando la codificación Base64. Hay que indicar que esta versión codificada se sitúa entre líneas delimitadoras de principio y fin de certificado.

Autoridad certificadora

Una Autoridad certificadora (CA) es una entidad de confianza, responsable de emitir y revocar los certificados digitales, utilizados en la firma electrónica.

Las Autoridades certificadoras emiten certificados para servidores, personas y software. Para comprobar la validez de los mismos sólo hace falta la llave pública de la CA, que va dentro del propio certificado.

Los certificados de una CA los firma otra CA o la propia CA, estableciéndose así dos tipos de certificados:



El certificado autofirmado es aquel certificado que cumple la particularidad de que la clave privada con la que se ha firmado en el campo CAsign , se corresponde con su homóloga pública, correspondiente

al campo que almacena la clave pública del mismo certificado. Diremos que aquellas entidades que dispongan de certificados autofirmados son autoridades certificadoras raíz.

Por otra parte las CA's pueden ser públicas o privadas. Los certificados raíz de las CA's públicas pueden o no estar instalados en los navegadores pero son reconocidos como entidades confiables, frecuentemente en función de la normativa del país en el que operan. Las CA's públicas emiten los certificados para la población en general y además firman CA's de otras organizaciones.

Tramitación de certificados

La solicitud de un certificado se realiza a través de unos objetos criptográficos que reciben el nombre de objetos pkcs10 (certificate request). Para generar este objeto, para posteriormente ser enviado a la CA correspondiente se debe, en primer lugar, decidir entre una tecnología u otra en función de nuestras necesidades. Decidir, el algoritmo de clave pública a utilizar, la longitud de las claves de dicho algoritmo, el algoritmo de firma… Una vez decidida la tecnología a emplear, se procederá a generar las claves privada y pública. En el proceso de generación de claves, existen dos alternativas, en primer lugar que sea el solicitante del certificado, quien se encargue de generar dichas claves o que se encargue la CA. El estándar x509 define que debe ser el solicitante quien deba generar dichas claves. Así pues, el solicitante genera un par de llaves K y P, se guarda K e incluye en el objeto pkcs10 sus datos personales junto con la clave pública y todo lo anterior firmado con su clave privada. Cabe destacar la importancia de la inclusión de la firma en la petición de certificado, ya que esto permite probar la validez de K y P y, además, asegurar que los datos que llegan a la CA coinciden con los que han sido enviados. De esta forma, la CA, comprueba la validez de la firma con P (que se ha incluido en el objeto), y solo en caso de ser válido procede al proceso de certificación.

El proceso de certificación consiste en analizar cada uno de los campos del DN y, según cual sea la política de certificación y las fuentes de las que dispone la CA, validar unos campos u otros. La CA, al tratarse de una entidad gestora, dispondrá de unas fuentes, en las que podrá observar si realmente los campos incluidos en la petición de certificado son correctos o no lo son. En función de cual sea el nivel de restricción del certificado y el campo validado, la CA dispone de distintos niveles de certificación CA1, CA2…..CAn. Así pues, en la caracterización de los certificados no va a influir únicamente el DN y el número de serie, sino las restricciones de acceso a la información y los campos validados por la CA. Existe un documento legal llamado CPS (certificate practic statement) que especifica el nivel de certificación de cada uno de los tipos de certificado emitidos por la CA. Así pues, resulta bastante importante que, a la hora de pedir un determinado certificado, el solicitante tenga presente cual es la política de certificación de la CA.

Una vez que la CA, ha validado los campos correspondientes, se encarga de enviar o publicar el certificado del solicitante. Normalmente lo enviado por la CA se corresponde con la información necesaria para constituir el certificado, además de la información anterior firmada por la CA (con la clave privada de la misma Kca) imprescindible para constituir el valor del campo firma del certificado.

El proceso anterior se ha descrito suponiendo que es el solicitante quien genera el par de llaves K y P, pero puede ocurrir que sea la CA la interesada en generar el par de llaves. Para solucionar y facilitar esta tarea el estándar x509 define unos objetos criptográficos denominados pkcs12, que cifran una determinada información. La información a cifrar, es el objeto x509 del solicitante y su clave privada. Así pues, el proceso de solicitud consiste, por parte del solicitante, únicamente en enviar el campo DN. La CA genera la clave privada y el certificado y lo incluye cifrado en el objeto pkcs12, enviándole esa información al solicitante. Como dicha información va cifrada, por la descripción del pkcs12, la CA también deberá encargarse de transmitir la clave que protege al certificado y a la clave privada. Esto genera el problema de cómo enviar de forma segura la clave que protege al pkcs12. Además, presenta el inconveniente de que la CA, al disponer de la clave privada, puede acceder a la información que otros usuarios cifran con nuestra clave pública.

Por otra parte, cabe mencionar OpenSSL como un robusto paquete de herramientas de

administración y librerías relacionadas con la criptografía, que resulta de gran ayuda a la hora de manejar todos los objetos relacionados con los certificados. En UNIX podemos utilizarlo medinate el comando openssl para distintos fines siguiendo la siguiente sintaxis:

openssl req, que realiza las operaciones:

- Tratamiento de objetos pkcs10, por ejemplo: req -text -in fichero_con_objeto_pkcs10
- Generación de nuevos objetos pkcs10 (+ clave privada): req -new
- Generación de certificados autofirmados (+ clave privada): req -new -x509

openssl x509, que permite:

- Tratar certificados.
- Generar un objeto pkcs10 con los datos de un certificado x509 y con una clave.
- Tomar un objeto pkcs10 y una clave privada y generar un certificado x509 autofirmado.

openssl pkcs12, que permite:

- Generar objetos pkcs12 con una clave privada, el certificado que le corresponde y todos los certificados que se quieran añadir: pkcs12 -export -inkey fichero_con_la_clave_privada -in fichero_con_el_cert
- Mostrar el contenido de objetos pkcs12.

CRL

El CRL es un documento expedido periódicamente por la CA en el que se muestra una lista con los números de serie de los certificados revocados hasta la fecha de consulta. Un certificado revocado es aquel cuya fecha de fin de validez ha sido adelantada, bien porque el propietario ha perdido la clave privada y no tiene acceso a ella, bien porque aún poseyéndola ha sido expuesta a individuos ajenos a la clave privada.

El CRL presenta dos problemas:

- El tamaño. El CRL aumenta cada vez más y más rápido. Como solución se presentan los CRL incrementales, en los que se añaden los nuevos certificados revocados a los ya existentes.
- La periodicidad. El CRL se debe emitir periódicamente lo que puede suponer un problema, ya que desde que se solicita la revocación de un certificado hasta que se emite el CRL hay un tiempo en que el certificado sigue siendo válido. Esto se resuelve hoy en día emitiendo el CRL cada vez que se revoca un certificado, pero esto supone un problema de propagación. Como solución se propone un protocolo OCSP, que permite consultar on-line a la CA si un determinado certificado está o no revocado. En consecuencia será necesario que el certificado contenga en las extensiones la URL que permite consultar el OCSP.

Dispositivos criptográficos

La infraestructura de llave pública (PKI) es el conjunto formado por la Autoridad Certificadora, los certificados emitidos por la misma y los usuarios a los que certifican. Su principal problema es que generalmente los usuarios no tienen conocimientos de informática, por lo que es necesario indicarles como instalarlos. Por otra parte, un certificado y una llave privada no se pueden memorizar, lo que implica la necesidad de poder transportarlos.

Estos aspectos dan lugar a problemas, ya que el usuario final ve el certificado y la llave privada como algo que se instala en una base de datos en el disco duro del ordenador y que no se puede usar fuera del mismo. En Windows, por ejemplo, se copia el contenido del objeto pkcs12 con un asistente, lo que le facilita la tarea al usuario, pero al eliminar el certificado la llave privada no se borra. Esto puede provocar problemas de seguridad ya que, si el usuario instala su certificado en otro ordenador, al eliminarlo su clave privada permanecerá en él. De esto se desprende que, para que el pkcs12 fuera un buen sistema, las aplicaciones deberían consultar su contenido y no copiarlo. Como solución se usan dispositivos criptográficos que son dispositivos específicos para almacenar esta información. Pero su uso presenta dos problemas:

- Es necesaria la existencia de un lector para el dispositivo en el ordenador al que lo queremos conectar.
- En caso de exista el lector para el dispositivo, es necesario que esté instalado el software para dicho lector.

Si se utiliza una memoria USB para el transporte del certificado, los problemas anteriores están resueltos, ya que existe el lector y el driver del dispositivo ya está instalado, pero todavía es necesario el software del modelo x509. Como solución se utiliza el pkcs11 que especifica una interfaz de programación para su uso con dispositivos criptográficos de cualquier tipo. Esta interfaz tiene un enfoque basado en objetos que permite que las aplicaciones realicen operaciones criptográficas sin conocer los detalles de la tecnología de los dispositivos. Su principal desventaja es que cada dispositivo requiere su pkcs11, lo que implica que cada aplicación necesite cargar los pkcs11 de los dispositivos que quiere usar. Este problema se resuelve en Windows con la Cryptoapi que es una interfaz que permite gestionar los dispositivos criptográficos globalmente.

Como ejemplos de dispositivos criptográficos cabe mencionar el DNI electrónico y el Clauer. El DNI electrónico tiene un chip que genera automáticamente el par de llaves e impide que la llave privada se pueda copiar, ya que no existe ningún comando para este fin. De esta forma sólo existe una copia de la llave privada que permanece siempre en el dispositivo. Además, el chip usado proporciona una protección física, ya que el intento de abrasión implica un deterioro del silicio. Por otra parte, el Clauer es una memoria, lo que supone la necesidad de extraer de ella la llave privada para su uso. Como ventaja presenta una gran comodidad al poder utilizarlo en prácticamente cualquier hardware. Su principal desventaja es que existen varias copias de la llave privada, lo que supone un mayor riesgo.

Seguridad en redes

Las redes informáticas suponen la posibilidad de compartir una enorme cantidad de recursos hardware y software tanto a nivel local como mundial ya que, con la conexión a Internet, cualquiera puede acceder a ordenadores de todo el mundo para desarrollar y ejecutar sus programas o para cualquier otra finalidad así como para acceder a información de todo tipo.

En contraposición se han incrementado los problemas de seguridad así como el número de atacantes, la privacidad de los datos sensibles se hace más difícil de garantizar, los controles de seguridad se vuelven más complejos y difíciles de implementar…

Como solución para intentar garantizar la seguridad se utiliza la criptografía en la mayoría de niveles de las redes de ordenadores. Éstas se asientan sobre el TCP/IP (Transmisión Control Protocol / Internet Protocol). Dicho protocolo se compone de los siguientes niveles:

- Nivel 1 o nivel de red. Nivel de acceso a la red que define como se deben transmitir los datos a través de los dispositivos físicos (ethernet,…)
- Nivel 2 o nivel Internet. Nivel que define los paquetes que compondrán la transmisión y que se encarga del encaminamiento de dichos paquetes. En este nivel se usa el protocolo IP.

- Nivel 3 o nivel de transporte. Nivel que proporciona los servicios de transmisión extremo a extremo garantizando una serie de características en la transmisión. En este nivel se utilizan fundamentalmente los protocolos TCP y UDP.
- Nivel 4 o nivel de aplicación. Nivel donde se encuentran las distintas aplicaciones y procesos que utilizan la red, tales como el telnet, ftp o el correo electrónico.

La información se empaqueta en paquetes (compuestos por una cabecera y unos datos) que en el emisor se transfieren desde las capas superiores a las inferiores, posteriormente circulan a través de la red física y en el receptor se transfieren de niveles inferiores a superiores.

De esta forma, se pueden establecer dos tipos de cifrado, dependiendo del nivel en el que este se produzca, de nominados cifrado de enlace y cifrado extremo a extremo.

Cifrado de enlace

En este esquema el cifrado se realiza en el nivel 1 de la jerarquía TCP/IP y se cifra tanto la información del mensaje incluida en cada paquete, como las cabeceras añadidas por todos los niveles superiores.

Tiene la ventaja de que toda la información de cada paquete está cifrada. Sin embargo presenta el inconveniente de que los paquetes únicamente van cifrados mientras se encuentran en el nivel de red, posteriormente, cuando llegan a niveles superiores de la jerarquía la información está en claro, con lo que cualquiera podría hacerse con ella. También tiene el inconveniente de que para poder llevarse a cabo con éxito es necesario que los dispositivos físicos (tarjetas ethernet, módems,...) sean capaces de cifrar/descifrar la información antes de enviarla.

A este tipo de cifrado pertenece el IPSEC (Internet Protocol Security).

Cifrado extremo a extremo

En este esquema el cifrado se realiza a nivel de aplicación, es decir, es la propia aplicación la que cifra los datos que posteriormente se enviarán a través de la red.

Tiene como ventaja que el usuario puede cifrar sólo parte de la información que transmite y puede hacerlo usando el método de cifrado que él elija. Además, los datos se encuentran cifrados desde el origen al destino de la transmisión, de manera que la red no necesita disponer de ninguna característica específica de cifrado.

Presenta el inconveniente de que se transmite parte de la información en claro y que el emisor y el receptor deben ponerse de acuerdo para realizar el mismo tipo de cifrado e intercambiar la clave o claves correspondientes previamente.

A este tipo de cifrado pertenece el SSL.

Autenticación

La autenticación es el proceso por el cual se comprueba la identidad de alguien o algo, para ver si es lo que dice ser. Para ello se utilizan credenciales (usuario y contraseña o con criptografía de llave pública) o pruebas de identidad (uso de retos). Este proceso es indispensable en las redes, ya que permite garantizar la seguridad a la hora de acceder a los recursos que ofrecen los servidores.

Usuario y contraseña

Este modelo de autenticación es el más usado, probablemente porque en Internet se persigue más conseguir el anonimato que la autenticidad. Cuando se usa este sistema no se está comprobando la autenticidad, se está dando acceso a una cuenta que cumple ciertos requisitos. Si realmente se quiere autenticar a la persona es necesario que haya un proceso administrativo previo en el que se verifique esta identificación.

Este modelo presenta el inconveniente de que se comparte un secreto (la contraseña) entre el cliente y el servidor, lo que puede dar lugar a problemas de seguridad si un tercero averigua esta información porque podrá usar las credenciales del cliente.

Uso de retos

Para evitar los problemas que tiene la autenticación mediante usuario y contraseña en páginas web no seguras se utilizan los retos. Este modelo consiste en que el servidor envía un reto aleatorio al cliente y éste lo cifra con su contraseña. El servidor utiliza la contraseña que ha almacenado del cliente para descifrar el reto. De esta forma se puede autenticar al cliente y, además, se evita el robo de las credenciales en claro, puesto que sólo se envían al servidor la primera vez que se conecta el cliente. Cabe destacar que el servidor almacena la contraseña en claro o algo derivado de ella, por ejemplo el hash.

Otra forma de aplicar este sistema, con el mismo fin, es que el cliente envíe el reto cifrado y que el usuario lo devuelva descifrado.

Un sistema muy usado basado en el uso de retos es el CHAP (Challenge Handshake Authentication Protocol). Este sistema verifica periódicamente la identidad del cliente enviándole el reto de forma que, si se produce un ataque en el que se pincha la línea y se echa al cliente, el atacante sólo puede suplantarle durante un pequeño periodo de tiempo.

Con criptografía de llave pública

Otra forma de realizar la autenticación es utilizando las características de la criptografía de llave pública. En este sistema no se comparte ningún secreto entre cliente y servidor ya que la información intercambiada son los respectivos certificados, que son públicos. En general es un sistema utilizado para la autenticación del servidor, ya que la mayoría de clientes no disponen de certificado y, en caso de disponer de él, hay que tener en cuenta los problemas de movilidad que presenta.

El protocolo SSL está basado en este tipo de autenticación.

Protocolo SSL

El SSL es un estándar de cifrado de extremo a extremo que se emplea para enviar información confidencial a través de la red. Su uso está más extendido para la aplicaciones web, pero se puede emplear en cualquier aplicación en el que se establezcan sesiones y transferencias inmediatas entre un emisor y un receptor (telnet, ftp …). El SSL fue desarrollado por la empresa Netscape, pero existen otros estándares, como el TLS, que son libres y realizan la misma función que el SSL. El SSL, es un estándar pensado en primer lugar para la protección de los clientes en un modelo cliente-servidor, aunque de forma opcional también puede establecer fases que permitan autenticar al cliente. Sus funciones fundamentales van a consistir en garantizar la confidencialidad de la información enviada entre cliente y servidor, la autenticidad del servidor y, opcionalmente, la autenticidad del cliente.

Así pues, en la conexión entre un cliente y un servidor, según el estándar SSL, se establece según las siguientes fases:

- El cliente y el servidor se ponen en contacto, para establecer convenios acerca de los

algoritmos a utilizar para la transmisión de información (algoritmo de firma , algoritmo de clave privada, algoritmo de clave pública…). Normalmente ambos tenderán a elegir la tecnología más eficiente y la más segura. En el caso de que no puedan ponerse de acuerdo, la conexión finaliza.

- En la segunda fase, se produce la autenticación del certificado del servidor. Para ello el servidor envía al cliente su certificado y el cliente comprueba la firma del certificado. Para ello le hará falta el certificado de la autoridad certificadora, para disponer de la clave pública de la misma y, así, poder comprobar la firma.
- En la fase anterior se ha comprobado que el certificado del servidor es correcto, pero no que el servidor se encuentre al otro lado de la comunicación. Esto se realiza en la tercera fase y consiste en que el cliente le envía al servidor un reto. Como después del reto se va a establecer el envío de información cifrada, para aprovechar dicha circunstancia, el reto consistirá en enviar la clave de sesión s cifrada con la clave pública del servidor. Si el servidor es realmente quien dice ser, podrá descifrar con su llave privada la llave de sesión y así podrá descifrar toda la información que el cliente le envíe con s. El envío de s como reto evita tener que añadir una fase en la que el servidor le indique al cliente que ha podido descifrar el reto. Si el servidor puede descifrar la información que le llega del cliente, se garantiza al cliente que al otro lado de la comunicación se encuentra realmente el servidor.
- La última fase, que es opcional, consiste en realizar la autenticación del cliente. Para ello el cliente envía su certificado al servidor, éste lo valida y con la clave pública del emisor el servidor envía un reto al cliente. Posteriormente el cliente enviará al servidor la información descifrada para que éste pueda comprobar la autenticidad del cliente. Indicar que, a diferencia de la fase anterior, todas las comunicaciones que se establecen entre cliente y servidor son confidenciales.

Cabe destacar que el SSL no garantiza siempre la autenticidad del servidor ya que, tras la fase de autenticación del mismo, el cliente puede decidir seguir adelante con la comunicación aunque haya fallado la autenticación del servidor. Esto provocara que la fase de cifrado sea dudosa porque también lo ha sido la fase de autenticación.

SMIME

El SMIME es un estándar para criptografía de clave pública y firmado de correo electrónico encapsulado en MIME, lo que permite dotar al correo electrónico de la confidencialidad y autenticidad que no tiene. Por este motivo es necesario conocer las partes fundamentales de un correo electrónico y el estándar MIME para entender el funcionamiento de SMIME.

El formato base de un correo electrónico especifica que este debe estar formado por una cabecera, un cuerpo y una línea en blanco que separa ambas partes. La cabecera es una parte con fines de control de la información que se envía. Se compone de una serie de líneas y se separa del cuerpo por una línea en blanco. Cada una de sus líneas tiene la siguiente estructura:

Campo:Valor ; otra cosa=valor

El problema de esta especificación es que, en los mensajes sencillos, el cuerpo está formado por texto simple que contiene información ASCII dentro del rango (32…126). El problema por tanto que se plantea es cómo enviar a través del correo información binaria, no imprimible por pantalla (rango 0…31), o caracteres de ASCII extendido como “¿”, o “ñ”. Como es deseable enviar cualquier tipo de información a través del correo electrónico se utiliza el estándar MIME, que proporciona una serie de formatos de codificación que permiten transformar información binaria en información ASCII, de forma

que sea susceptible de ser enviada.

Así pues, un mensaje MIME añade a la cabecera del mensaje el campo MIME-Version, en la actualidad 1.0., además del campo Content-Type, cuyo valor indica el tipo de información que se va a enviar (tipo MIME).Cada tipo MIME consta de dos partes, una que indica el tipo de medio del que se trata (video, imagen, audio, etc) y otra que indica de qué clase de ese medio se trata. Por ejemplo, el valor adecuado para un campo Content-Type sería Image/gif.

La primera parte especifica por tanto el medio, que en este caso es Image, y la segunda parte el tipo de medio, en este caso formato gif. Los tipos MIME se especifican en una lista que contiene los más extendidos. En el caso de que la lista de tipos reconocidos no contenga el que se desea enviar, se añade “x” delante del valor de la primera parte. En caso de no especificarse el tipo MIME, el contenido del cuerpo se interpretará como text/plane.

Si la información a enviar debe convertirse en caracteres legibles, debe codificarse. La codificación empleada se especifica en el campo Content-Transfer-Encoding. Existen 4 formas fundamentales de codificar la información, ASCII 7 bits, ASCII 8 bits, base 64 y quoted-printable. Las dos primeras formas de codificación se presuponen, base64 se utiliza para información binaria y la última de las formas de codificación para la impresión de texto.

Así pues, un correo básico MIME sólo nos permite enviar un único objeto MIME de forma que, si se desea enviar más de un objeto, es necesario utilizar una serie de tipos reservados MIME que permiten incluir en un correo electrónico más de un objeto MIME. Nos interesan Multipart/mixed, para el envío de varios tipos mime, y Multipart/signed para el envío de mensajes firmados.

De esta forma si se desea enviar más de un objeto, se especifica en el campo Content-Type, que se trata de un objeto Multipart/mixed. Además, es necesario establecer un criterio que permita separar unos objetos de otros, este criterio se especifica en el campo boundary, en la misma línea que la especificación del tipo mixto. De este modo, si se quiere indicar que ya ha terminado o que empieza un nuevo objeto MIME, se utiliza --boundary. Indicar que entre la especificación del tipo MIME y su contenido se emplea un espacio en blanco. Para indicar la finalización completa del tipo MIME se emplea --boundary--.

Así pues, esta forma de definir la inclusión de más de un objeto MIME en un mensaje, permite definir una recursividad entre multipart/mixed, de tal forma que algunos de los objetos MIME que se encuentran en un multipart/mixed sean otro multipart/mixed. En este caso habrá un boundary para cada uno de los multipart/mixed.

El SMIME es una ampliación del MIME que permite mantener toda la funcionalidad del MIME, además de poder enviar mensajes seguros (cifrados y firmados). Para realizar dicha labor dispone de unos objetos criptográficos denominados pkcs7 que permiten almacenar información relativa a la seguridad. Existen principalmente cuatro modos distintos que se diferencian por el tipo de información que contienen. Estos modos son:

- Data: el objeto criptográfico contiene únicamente datos.
- Signed data: el objeto contiene la firma del emisor del mensaje.
- Enveloped data: contiene datos cifrados.
- Signed and enveloped data: contiene datos cifrados y firmados.

Como dichos objetos criptográficos están definidos en formato DER (información binaria), a la hora de ser enviados serán especificados como objetos MIME y serán codificados en base64.

Correo firmado

Para el envío de mensajes firmados se emplea el tipo reservado MIME multipart/signed que permite

enviar dos objetos MIME, el primer objeto es el propio mensaje y el segundo el objeto pkcs7, en modo signed data. Como el receptor del mensaje debe comprobar la firma, para observar si realmente viene de A y para saber que el canal de comunicación no ha sido interceptado, se envía en el mensaje el certificado del emisor, de forma que el receptor del mensaje pueda disponer de la clave pública del emisor y así poder comprobar la firma. En la especificación del tipo MIME también deben hacerse referencias acerca del algoritmo empleado para firmar.

Correo cifrado

Un correo cifrado consiste únicamente en enviar el objeto pkcs7 en modo enveloped data. Por tanto en la cabecera del mensaje es necesario incluir un tipo MIME en el campo Content-Type y debe ser codificado en base64.

Con respecto a los mensajes cifrados, indicar que los agentes de correo no muestran las partes MIME que componen el mensaje habitualmente, sino que descifran los datos que contiene el pkcs7 y construyen un nuevo mensaje con los datos descifrados y con el resto de la cabecera del mensaje inicial, que no se refiere al cifrado de la información. Esto evita que el usuario de SMIME tenga que descifrar toda la información, escondiendo así la complejidad de los métodos criptográficos empleados.

Para enviar un correo cifrado a varia personas es necesario hacer tantas copias del correo como destinatarios a los que vaya dirigido, ya que el correo va cifrado para el destinatario. Pero esto resulta muy costoso, así que como solución se cifra el correo con una llave de sesión que se cifra con la llave pública de cada uno de los destinatarios, de forma que el receptor debe buscar entre toda la información recibida cuál es la llave de sesión que ha sido cifrada con su llave pública. Para realizar esta tarea eficientemente se envía el identificador del receptor junto a cada llave de sesión cifrada. Este identificador será el certificado de cada receptor, lo que hará que el emisor necesite tener el certificado de cada receptor para poder enviarle el correo cifrado. Hay que indicar que no se envía el certificado completo como identificador, sino el nombre de la CA (campo DN issuer) y el número de serie del certificado, ya que esto es suficiente para identificar el receptor porque una CA no repite nunca dos números de serie.

Firewall

En las redes de ordenadores existen multitud de amenazas de las que es necesario protegerse. Para ello, además de la criptografía, existen otros mecanismos como son los firewall o cortafuegos. Un firewall es un mecanismo que controla el tráfico entre un sistema (de confianza) y una red externa (de menor confianza, como puede ser Internet). Para ello, el firewall filtra los paquetes que pasan a través de él y actúa de acuerdo a la política de seguridad general establecida:

- Política abierta: se permite el acceso a todos los paquetes y se deniega el acceso a ciertos paquetes con ciertas características.
- Política cerrada: se deniega el acceso a todos los paquetes y se aceptan paquetes con ciertas características.

La primera de las políticas, se emplea en redes locales con muchos usuarios y en las que no es necesario tener un control muy estricto sobre la información que entra y sale tanto a la red local como al propio cortafuegos. La segunda, se emplea en redes locales con un menor número de usuarios y en las que es necesario tener un control muy estricto sobre los paquetes que entran y salen a la red local o al cortafuegos.

Para realizar el filtrado de paquetes, el cortafuegos analiza los paquetes TCP/IP que le llegan y, en función de cuales sean sus características (a que puerto van dirigidos a nivel TCP y origen y destino a

nivel Ip), permitirá el paso al sistema de dichos paquetes o les denegará el acceso. En caso de que el análisis sea negativo podrá realizar dos acciones:

- Drop: no se comunica nada el emisor del paquete, se desprecia como si el sistema no estuviera operativo.
- Reject: Se devuelve una respuesta al emisor del paquete indicándole que no tiene permiso para introducirse en el sistema.

Existen fundamentalmente tres tipos de filtrado, dependiendo de las características de la máquina que se encargue de gestionar el control de los paquetes:

- Forwarding: tipo de filtrado básico en el que se controla el tráfico entre dos redes. Se analizan los paquetes que entran o salen desde o hacia el interior de una red local.
- Input: modo de filtrado que se emplea cuando el cortafuegos, además de controlar los paquetes, ofrece una serie de servicios en la red. En ese caso, también interesará tener un control de los paquetes para proteger a los usuarios y al sistema que implementa. Así pues, las reglas del tipo de filtrado Input controlan el acceso de paquetes TCP/IP al interior del cortafuegos o al sistema.
- Output: de forma análoga, también interesa tener un control de los paquetes que salen desde el cortafuegos-servidor a Internet. Las reglas del tipo Output, permiten gestionar dicho control.

Hay que indicar que, la definición de reglas de filtrado en caso de una política cerrada, resulta complicada porque una mala definición puede provocar que no llegue ni salga información del sistema.

Como herramienta estándar altamente versátil para la configuración de firewalls disponemos de una aplicación llamada iptables, que se ejecuta en entornos Unix y que permite trabajar con los tres tipos fundamentales de filtrado de paquetes. Para usarla sólo hay que abrir una consola e introducir comandos en el intérprete de comandos correspondiente, mediante los cuales se pueden establecer reglas acerca de lo que se desea hacer con los paquetes que intentan penetrar en la red interior-firewall o salir de la misma hacia el exterior. Estas reglas se almacenan en una tabla que se consulta de principio a fin buscando alguna regla que cumpla el paquete, de forma que se ejecuta la acción asociada a ella. Algunas de las opciones fundamentales son:

iptables –A regla_de_filtrado: permite añadir una nueva regla de filtrado al final de la tabla.

iptables –I regla_de_filtrado: se añade una nueva regla de filtrado al principio de la tabla.

iptables -P tipo_de_filtrado : permite establecer una política, con el tipo de filtrado especificado en el comando. Para especificar si se trata de una política de aceptación o de rechazo por defecto, se utiliza la palabra ACCEPT o DROP respectivamente.

iptables -p protocolo: permite establecer sobre que identificador de puerto se van a aplicar las reglas, especificadas en la línea que lo contiene.

iptables -s dirección/es_ip: para especificar las ips origen sobre las que se van aplicar las reglas correspondientes.

iptables -d dirección/es_ip: para especificar las ips destino sobre las que se van a aplicar las reglas correspondientes.

iptables –m módulo: permite cargar módulos en el núcleo y habilitar las correspondientes opciones en iptables. Algunos destacables son el módulo recent, que cuenta los accesos por ip y mantiene una tabla (que se puede consultar en /proc/net/xpt-recent/) que permite ver cada ip cuantas veces accede y especificar reglas para cada una de ellas, y el módulo limit, que permite limitar el número de

peticiones que entran por unidad de tiempo.

Proxy

El proxy es un elemento software, que trabaja a nivel de aplicación, que permite establecer una seguridad muy alta en una red. El proxy establece una especie de puente entre los equipos que están totalmente aislados de Internet e Internet, de forma que intercede entre los ordenadores ante ciertos servicios provocando que los equipos sólo necesiten estar conectados al proxy. Los proxies se pueden usar literalmente o de forma transparente. Si se utilizan de forma transparente es el firewall el que se encarga de hacerlo ya que, cuando intercepta paquetes con destino al puerto 80, los redirige al proxy. Además, el proxy presenta otras utilidades. Permite loggear las peticiones web que se realizan, de forma que se puede monitorizar la actividad de la red. También puede incluir un antimalware, lo que permite la detección de virus, troyanos... Y puede tener una cache, lo que permite reutilizar la misma página hasta que se cumple la caducidad de la misma.

Problemas de seguridad en la red

La mayor parte de problemas de seguridad en la red se deben al web, aunque estos no suelen ser muy graves. Esto es debido a que, inicialmente, las páginas web eran informativas pero con el tiempo se han multiplicado las páginas que son aplicaciones que se ejecutan en el servidor. Por todo esto, a la hora de implementar una aplicación web se debe prestar especial atención a:

- la programación. La existencia de fallos en el software permite abrir puertas en el sistema a los atacantes, que se encargarán de realizar una intrusión a la información. El ataque típico a las aplicaciones es el buffer overflow, que consiste en añadir código dentro del bloque de activación de una subrutina fuente que se ejecuta con cierta frecuencia. El éxito de la introducción de este código consiste en que la dirección de retorno de la subrutina en la pila, se corresponde con el inicio del código que se va a encargar de realizar la intrusión. Cabe indicar que este es un problema que se da en los lenguajes compilados y no en los interpretados porque en estos últimos la reserva de memoria se realiza en el acto. Como solución al buffer overflow se puede:
  - hacer que las páginas de la pila sean no ejecutables.
  - utilizar la técnica de randomización de la pila. De esta forma se descuadra la pila de una llamada a otra para que el atacante no pueda localizar las direcciones usadas.
- la interacción entre los servidores. Se debe diseñar un sistema de autenticación cuidadosamente.
- la autorización. Muchos de los problemas son debidos a la falta de comprobación de autorización, ya que esto es algo que se debe comprobar en cada una de las etapas en las que el cliente se conecta al servidor.

Virus y amenazas programadas

No existe un acuerdo general acerca de lo que es un virus, aunque se suele entender como una determinada cantidad de código cuya misión es sobrevivir, intentado infectar y reproducirse. Puede tener un efecto maligno o no, pero normalmente lo tiene en forma de daños colaterales. Se encuentra caracterizado como una amenaza programada, entendiendo como tal cualquier tipo de amenaza sobre la información de un determinado sistema que tiene carácter automático. Así pues, el concepto de amenaza programada está altamente relacionado con el concepto de programa, a diferencia de que el objetivo de ésta es difundirse y realizar tareas no-lícitas desde el punto de vista legal-moral.

Además de los virus, existen otras amenazas programadas como:

- Caballo de Troya: es un programa que aparentemente realiza una función útil para quién lo ejecuta, pero que en realidad (o además) realiza una función que el usuario desconoce, generalmente dañina. La forma más fácil de descubrir caballos de Troya (aparte de sufrir sus efectos una vez activado) es compara los ficheros bajo sospecha con una copia de los originales, copia que evidentemente se ha de haber efectuado antes de poner el sistema en funcionamiento y debe haber sido guardada en un lugar seguro, para evitar así que el atacante modifique también la versión de nuestro backup. Algunos troyanos conocidos son por ejemplo, las versiones 78 y 79 del SCAN de McAfee o la versión 3.0 del pkzip.
- Gusanos: son programas capaces de viajar por sí mismos a través de redes de computadores para realizar cualquier actividad una vez alcanzada una máquina; aunque esta actividad no tiene porque entrañar peligro, los gusanos pueden instalar en el sistema alcanzado un virus, atacar a este sistema como haría un intruso o, simplemente, consumir excesivas cantidades de ancho de banda en la red afectada. Uno de los más conocidos es el que invadió Internet en 1988.
- Spyware: es un programa que se instala furtivamente en un ordenador para recopilar información sobre las actividades realizadas en éste. La función más común que tienen estos programas es la de recopilar información sobre el usuario y distribuirlo a empresas publicitarias u otras organizaciones interesadas, pero también se han empleado en organismos oficiales para recopilar información contra sospechosos de delitos, como en el caso de la piratería de software.

Origen

El nacimiento de los virus, como amenazas programadas, está íntimamente ligado a la difusión de MS-DOS como sistema operativo estándar para realizar la administración de los ordenadores personales. El problema fundamental MS-DOS, es su propia filosofía de funcionamiento: un sistema monousuario y monotarea. Al ser monosuario y monotarea el propio sistema no protege lo que otros usuarios puedan hacer sobre la información que administra y, lo que es más grave, no puede controlar lo que los procesos de un determinado usuario pueden hacer con informaciones ligadas a otros usuarios. De ahí es donde va a nacer la motivación por parte de los programadores de computadores para realizar los virus. Al contrario que en MS-DOS, los virus en UNIX están restringidos al campo y a las restricciones de acceso de los usuarios al estar claramente diferenciado lo que los procesos de un determinado usuario pueden hacer con informaciones asociadas a otros usuarios. En MS-DOS los virus se clasifican en:

- Virus en el sector de arranque: Comúnmente denominados virus de “boot”. Su acción se fundamenta en que el sector de arranque de los disquetes y la tabla de particiones del disco duro contienen un pequeño programa que se carga al encender el ordenador. Los virus de “boot” copian el sector de original en algún otro lado y después se copian ellos en su lugar. Dado que hay poco espacio suelen ocupar también algún que otro sector del disco. Para evitar que los sectores donde se copia el arranque original o alguna parte del virus sean sobreescritos, suelen marcarlos como defectuosos en la FAT. Una vez activos en memoria, van infectando a todos los discos a los que accede, controlando la E/S sustituyendo en las zonas del sistema operativo reservadas al tratamiento de los periféricos, código que infecte el sector de arranque para cada uno de los discos a los que se accede. Este tipo de virus tiene como característica su fácil infección, sin embargo no resulta tan fácil es su propagación, ya que tomando ciertas medidas se puede evitar.

- Virus de programa: estos virus se diferencian de los anteriores porque estos infectan los programas. Su modo de actuar es añadiendo código al programa que infectan modificando el contador de programa para que pase a ejecutar la primera instrucción del virus. Una vez esté ejecutado el virus devuelven el control al programa que el usuario quiere ejecutar, evitando así ser detectados en tiempo de ejecución. Una copia del programa infectado supone una forma de propagación para este tipo de virus.

Después de la aparición de Windows 95 como sistema operativo más difundido, los virus residentes en memoria desaparecen debido a la gestión de memoria que hace el sistema operativo (MS-DOS utiliza memoria real y a partir de Windows 95 la memoria es paginada), pero no supone el final de los virus. Con la gran aceptación de las herramientas ofimáticas de Microsoft aparece otra generación de virus informáticos: los virus de macro. Estos virus suelen ir adjuntos a documentos Word o bases de datos como Access. Se basan en la posibilidad que ofrecen muchos programas de ejecutar otros programas (denominados macros) al cargar ficheros. Esto se debe a que existen programas que vienen con lenguajes de programación incluidos, los cuales están diseñados para “ayudar” a los usuarios a automatizar ciertas tareas relacionadas con el funcionamiento de la propia aplicación. Un ejemplo de este tipo de aplicaciones son las aplicaciones del Office de Microsoft. El concepto de virus de macro funciona porque estos lenguajes de programación proporcionan accesos a memoria y a los discos duros. De esa forma un virus de macro es simplemente una macro para uno de estos programas. La mayoría de estos virus son “Auto-Open” es decir, que se ejecutan causando el daño correspondiente cuando un documento o plantilla que contiene el virus se abre utilizando la aplicación correspondiente.

Como virus de macro más comunes nos encontramos al Melissa y el famoso I love you. Ambos se trasmiten por correo electrónico. Este último se hizo famoso por colapsar diversas redes informáticas de empresas. La macro que contenía el virus era un script en Visual Basic Script, que se ejecutaba al abrir el mensaje de correo electrónico que lo adjuntaba con Microsoft Outlook. El virus provocaba una sobrecarga de las redes informáticas al enviarse a todas aquellas direcciones que tuviese el usuario en la agenda de Outlook y borraba ficheros de cierto tipo. Además, creaba dos registros en el registro principal de Windows, provocando su ejecución automática al arrancar dicho sistema operativo. El Melissa, sin embargo, era una macro de Word y tenía que ser abierto por esa aplicación.

Contramedidas: Antivirus

Hay que tener en cuenta que es imposible estar prevenido contra cualquier posibilidad de ser atacado por un virus. En cualquier caso una serie de medidas pueden ser tomadas:

- Elegir uno o varios “scanners” o antivirus.
- Acostumbrarse a pasar el antivirus a todo programa que recibamos.
- Crear un disco de emergencia.
- Utilizar un programa residente de protección.
- Utilizar comprobadores de integridad.
- Conviene desactivar toda posibilidad de ejecución de programas vía e-mail y usar herramientas seguras para gestión de correo, ya que un virus es un programa que debe ejecutarse para estar activo y por tanto un virus anexo a un mensaje de correo no hace nada hasta que se ejecuta.

Los antivirus son programas que buscan secuencias de bytes características de los virus. Algunos de los más conocido son Fprot, McAfee Viruscan, Panda Antivirus…, en el caso de que se utilicen varios es necesario volver a arrancar la máquina entre uno y otro para evitar falsos positivos. Suelen comprobar que no han sido modificados y que no hay virus en memoria antes de ejecutarse. Normalmente suelen dividirse en dos partes:

- El motor de búsqueda. Proporciona desde la interfaz con el usuario hasta el sistema interno de búsqueda.
- La base de datos que contiene las descripciones de los virus, nombres, características, patrones o secuencias de bytes etc…

El funcionamiento se logra con la coordinación de estas dos partes. El motor de búsqueda es el encargado de buscar en los discos y memoria aquellos patrones o secuencias identificados en la base de datos. Es necesario tener en cuenta que por cada actualización del motor de búsqueda, aparecen numerosas versiones de la base de datos de virus. Diariamente aparecen nuevos virus por lo que las bases de datos deben ser actualizadas de la forma más rápida posible. Igual de importante es la fase de detección como la fase de eliminación, de tal forma que el antivirus eficiente es aquel que incorpora ambos factores en la medida posible.

Ingeniería social

La ingeniería social es la práctica de obtener información confidencial a través de la manipulación de usuarios legítimos. El principio en el que sustenta es que en cualquier sistema “los usuarios son el eslabón débil”. La forma típica de obtener información en la ingeniería social consiste en engañar al usuario porque resulta efectivo y rápido. Un buen ejemplo es el phising, en el que el estafador se hace pasar por una persona o empresa de confianza en una aparente comunicación oficial electrónica, por lo común un correo electrónico, pidiendo la comunicación de información confidencial (como puede ser una contraseña o información detallada sobre tarjetas de crédito u otra información bancaria). Como solución al phising existen varias técnicas diferentes para combatirlo:

- la legislación
- la creación de tecnologías específicas que tienen como objetivo evitarlo. Los lectores de correo pueden detectarlo comparando la URL del correo electrónico con la real.
- en las empresas, el entrenamiento de los empleados de modo que puedan reconocer posibles ataques phishing.

Hay que indicar que el impacto de los virus junto con la ingeniería social es difícil de medir, pero que existe un alto porcentaje de quipos afectados por alguno de ellos produciéndose, en consecuencia, pérdidas económicas considerables.

Seguridad en Sistemas Operativos

Un sistema operativo es un programa cuya misión es controlar los recursos de la máquina y al resto de programas, de forma que haga un reparto eficiente y equilibrado de los primeros y proporcione seguridad a los segundos frente a otros programas y frente al propio sistema operativo. Para ello se basará tanto en el hardware como en los modelos de seguridad existentes.

Hardware y seguridad

El mejor o peor funcionamiento de un sistema operativo depende en gran parte del hardware. Si tenemos un sistema operativo monousuario y monotarea, como es el caso de MS-DOS, podrá funcionar prácticamente en cualquier máquina. En MS-DOS sólo hay un usuario (administrador) y los programas se ejecutan uno tras otro. Además trabaja suponiendo la arquitectura de PC original, en la que la memoria es lineal y el procesador utiliza las direcciones de la 0 a la 640 y la memoria y la memoria de vídeo se encuentran a partir de la dirección 640. El procesador 8088 no tiene ningún tipo de control, sólo ejecuta instrucciones, lo que provoca que el usuario pueda acceder a cualquier parte de la memoria, incluida la del procesador.

Por este motivo, es necesario que el procesador tenga dos modos de ejecución, usuario y núcleo. De esta forma se garantiza que los programas de usuario, que se ejecutan en modo no privilegiado, no puedan acceder a memoria del núcleo. Este control se debe establecer mediante hardware, ya que el paso a modo privilegiado se produce al llegar una interrupción.

Modelos de seguridad

La caracterización de los sistemas operativos es fundamental a la hora de establecer las políticas de seguridad y de autenticación de los usuarios, ya que estas últimas dependen del tipo de sistema operativo. La clasificación de sistemas operativos es la siguiente:

- Sistemas operativos monosuario y monotarea. Son sistemas operativos en los que no es necesario realizar una autenticación de los usuarios porque existe un único usuario. Así, cualquier usuario que emplee la máquina en concreto goza de todos los permisos acerca de la información que se encuentra almacenada, por lo tanto el sistema es altamente inseguro ya que rompe con uno de los principios básicos de la seguridad en computadores, el principio del menor privilegio. Un ejemplo de este tipo de sistema operativo es MS-DOS de Microsoft.
- Sistemas operativos multiusuario. A diferencia de los sistemas anteriores, el sistema se debe encargar de autenticar a los usuarios, es decir, establecer métodos que nos permitan saber si las personas que desean acceder a una determinada información son realmente quienes dicen ser. Además de tener en cuenta la autenticación de los usuarios, una vez que un determinado usuario está haciendo uso de los recursos del sistema, deben tener en cuenta cuales son los permisos sobre la información que maneja. Esto se lleva a cabo en la constitución del sistema de ficheros, que puede consistir en asignar a cada fichero permisos que controlen y restrinjan lo que otros usuarios pueden hacer con dichos ficheros. Un caso concreto de los sistemas multiusuario, son los multitarea, es decir, aquellos sistemas en los que pueden ser ejecutados varios procesos a la vez a su vez lanzados por usuarios distintos. En este caso, es necesario controlar lo que los procesos de un determinado usuario pueden hacer sobre la información que manejan procesos que fueron lanzados por otros usuarios. Normalmente, esta protección se lleva a cabo realizando una gestión correcta de la memoria, marcando las zonas de memoria correspondientes a unos procesos y a otros. Fundamentalmente hay dos modelos de seguridad:
  - Vallado. La CPU marca el fin y el inicio de la zona de memoria reservada para la ejecución de un determinado proceso de un determinado usuario. Así pues, si un proceso, accede a una zona de memoria no comprendida entre el valor sus marcas, la CPU se encarga de generar una excepción que será tratada por el sistema operativo ejecutando código que se encuentra en la zona reservada para el núcleo. La implementación de estas marcas se lleva a cabo por hardware mediante el uso de dos registros del procesador (registros de vallado), cuyo contenido cambia según el proceso que se esté ejecutando.
  - Memoria virtual. El sistema operativo realiza una gestión de la memoria central como si estuviera dividida en páginas. Así pues, cada proceso tiene asociado un conjunto de páginas. Con la memoria virtual las direcciones referenciadas por el proceso se corresponden con la memoria asignada al mismo (desde la dirección 0 hasta el límite), no a la memoria física completa. De esta forma se realiza un control por hardware que hace que el proceso sólo pueda acceder a las páginas que se le han asignado.

Seguridad en UNIX

UNIX es un sistema operativo multiusuario que incorpora multitarea y que fue desarrollado originalmente por Ken Thompson y Dennis Ritchie en los laboratorios de AT&T Bell. Tiene diversas variantes y se considera potente, más transportable e independiente de equipos concretos que otros sistemas operativos porque está escrito en lenguaje C. Además es un sistema operativo muy seguro que se basa en la utilización de contraseñas, permisos y ciertos bits para controlar el acceso a la información.

Contraseñas

La gestión de la autenticación de los usuarios en Unix se realiza mediante la comprobación de un usuario y una password. Estos van a ser los parámetros que van a identificar a un usuario ante el sistema operativo y le van a permitir saber que el individuo correspondiente es realmente quien dice ser. Para la gestión de las password es fundamental conocer la estructura de los ficheros que Unix emplea para la autenticación de los usuarios, estos son /etc/passwd y /etc/shadow.

El fichero /etc/passwd contiene información acerca de los usuarios locales o remotos de una determinada máquina. Se trata de un fichero ASCII (que puede ser editado por cualquier tipo de editor de texto), compuesto por líneas cada una de las cuales presentan el siguiente formato:

```
us : pass : uid : gid : cuenta : dir : shell
```

- us: contiene el login del usuario que se describe en la línea correspondiente.
- pass: contiene la password cifrada o en su lugar una x o cualquier otro carácter que simbolice la presencia de la password cifrada en el fichero /etc/shadow.
- uid: contiene el user id, se trata de un valor numérico que identifica al usuario correspondiente.
- gid: contiene el group id, se trata de un valor numérico que identifica el grupo primario del usuario.
- cuenta: contiene una descripción de la cuenta correspondiente.
- dir: contiene la ubicación en el sistema de directorios (la ruta), del directorio asociado al usuario correspondiente.
- shell: contiene la ubicación (la ruta) del shell de inicio.

Como puede observarse en el contenido de cada uno de los campos, el fichero /etc/passwd sirve para controlar los usuarios del sistema y para dar información a los procesos acerca de los usuarios que los ejecutan. Indicar que se emplea para identificar al usuario tanto su login como su Id. Esto se emplea por dos razones, la primera para que sea más cómodo para el usuario acordarse de algo significativo y no de un número carente de significado y, por otra parte, para aportar información a los procesos indicándoles a que login está asociado el correspondiente uid. Así pues, si ejecutamos el comando ls -l que nos lista los ficheros y directorios, observamos que en los poseedores de los ficheros, tanto en grupo como en usuarios particulares, aparece el nombre del login y no del uid. Esto puede ser llevado a cabo gracias al contenido de los campos asociados a cada uno de los usuarios en el fichero /etc/passwd.

Con respecto a la estructura de este fichero y su gestión cabe indicar que, dos líneas que posean el mismo uid, serán considerados por el sistema operativo como el mismo usuario aunque posean distinto login. Esto constituye una vulnerabilidad que va a ser explotada al máximo por los atacantes del sistema operativo. En concreto la vulnerabilidad, radica en el uid 0 que se corresponde con el del administrador del sistema. Así pues, podrá existir un usuario del sistema con un login y una password distinta a la de root con el uid a 0, de forma que éste podría gozar de los mismos permisos que root.

Por otra parte, el fichero /etc/shadow contiene información confidencial acerca de los usuarios del sistema. La información confidencial que almacena son las passwords que emplea el usuario correspondiente para autenticarse en el sistema. Las claves de acceso empleadas por el usuario para acceder a la máquina se encuentran cifradas utilizando un criptosistema irreversible basado en las funciones estándar de la librería <crypt.h>.

Las contraseñas se guardaban, en un principio, cifradas con DES, de forma que se tomaban como clave los ocho primeros caracteres de la contraseña elegida por el usuario para cifrar un bloque de texto en claro de 64 bits puestos a cero. Este sistema tenía como inconveniente que un atacante podía fabricar una base de datos con todas las posibles contraseñas cifradas indexadas (índice_contraseña_cifrada : valor_original_contraseña).

Para evitar este problema se realiza una permutación durante el proceso de cifrado elegida de forma automática y aleatoria para cada usuario, basada en un campo formado por un número de 12 bits llamado “salt”. Así, se almacenan en /etc/shadow los ocho bytes de la contraseña concatenados con el salt. Hay que indicar que el salt se almacena en claro, así que su única misión es aumentar el tamaño de un posible diccionario con el que indexar las contraseñas. Con el tiempo se ha decidido usar más algoritmos de cifrado para permitir contraseñas de mayor tamaño y usar algoritmos más robustos que el DES, así los más usados son el md5 y el blowfish. Cada uno de los algoritmos usados se identifica con un número que aparece al principio del campo en el que se almacena la contraseña (idsalt$algoritmo_cifrado).

La estructura del fichero se encuentra dividida en líneas cada una de las cuales se encuentra asociada a cada uno de los usuarios del sistema. El formato de cada línea es el siguiente:

```
us : passcifr : umod : dc : minc : maxc : da : di : ti : res
```

- us: login del usuario que se describe en la línea correspondiente.
- passcifr: campo que contiene la password del usuario correspondiente cifrada. Está formado por el identificador del algoritmo de cifrado, el salt y la password cifrada.
- umod: contiene la última modificación del password.
- dc: Días transcurridos del último cambio de clave desde el día 1/1/70
- minc: Días transcurridos antes de que la clave se pueda modificar.
- maxc: Días transcurridos antes de que la clave tenga que ser modificada.
- da: Días de aviso al usuario antes de que expire la clave.
- di: Días que se desactiva la cuenta tras expirar la clave.
- ti: Días de duración de la cuenta desde el 1/1/70.
- res: campo reservado.

Así pues, cuando un usuario se conecta y aporta información al sistema (login y password), el sistema operativo cifrará la password introducida y comprobará, consultando el fichero /etc/shadow si el contenido del mismo coincide o no. En caso de que coincida la información permitirá el acceso al usuario, en caso contrario lo denegará. Cabe indicar que el éxito de este sistema de autenticación radica en las siguientes dos propiedades:

- El fichero /etc/passwd debe ser accesible por todos los usuarios/procesos del sistema en solo lectura. Esto parece obvio, ya que si estuvieran activos los permisos de escritura cualquier usuario podría crear nuevos usuarios o modificar datos acerca de los ya existente.
- El fichero /etc/shadow, debe ser accesible únicamente por root y, por tanto, debe tener los permisos de escritura, lectura y ejecución para cualquier tipo de usuario o proceso del sistema.

Permisos

Una de las formas más habituales para proteger la información de los usuarios en un sistema operativo, consiste en asignar a cada uno de los ficheros que almacena la información cadenas de bits que especifican quién puede acceder al fichero y de qué forma puede hacerlo. Este es el mecanismo usado en los sistemas operativos UNIX. Los permisos se dividen en tres ternas en función de a qué usuarios afectan (la primera afecta al propietario, la segunda al grupo del propietario del fichero y la tercera al resto de usuarios); cada una de ellas indica la existencia o la ausencia de permiso para leer, escribir o ejecutar el fichero o directorio:

Permiso	Fichero	Directorio
r	permite la lectura del contenido	permite listar el contenido
w	permite la escritura y/o modificación del contenido	permite crear y borrar ficheros del el directorio (independientemente de los permisos del fichero)
x	permite su ejecución	permite que el directorio forme parte del camino de un proceso

Además de la información que aportan los permisos anteriores, también se especifica información que permite distinguir los ficheros de los directorios. Si el primer bit es una “d” significa que se trata de un directorio, en caso de que este bit no esté presente se trata de un fichero.

El propietario y el grupo de un fichero se pueden modificar con las órdenes chown y chgrp respectivamente; ambas reciben como parámetros al menos el nombre de usuario o grupo (los nombres válidos de usuario son los que poseen una entrada /etc/passwd mientras que los grupos válidos se leen de /etc/group) al que vamos a otorgar la posesión del fichero, así como el nombre del archivo a modificar.

Para modificar los permisos de un archivo se utiliza la orden chmod. Este comando generalmente recibe como parámetro el permiso en octal que queremos asignar a cierto fichero, así como el nombre del mismo.

Es fundamental que el usuario conozca el significado de los permisos, ya que constituyen una contramedida muy efectiva contra amenazas de tipo alteración de la información. Por este motivo una buena política de seguridad del sistema informático, deberá informar a sus usuarios sobre el significado de los permisos y de las herramientas de que dispone para modificarlos.

En Linux los permisos se almacenan como un mapa de bits, pero también existe la posibilidad de trabajar con ACL si el núcleo lo soporta y se dispone de las herramientas pertinentes para su manejo.

ACL

Otra forma de almacenar aquello que pueden hacer los usuarios sobre los recursos del sistema consiste en construir una tabla en la que hay una entrada por cada usuario y recurso con aquellas acciones que puede realizar el primero sobre el segundo. Esta tabla se almacena en la práctica:

- Por filas. De forma que cada usuario tiene una serie de permisos sobre una lista de recursos. Si un recurso no está en la lista se asumen los permisos por defecto. Los sistemas que utilizan este esquema se llaman de capabilities (capacidades) y son muy seguros, pero muy complejos y difíciles de mantener.
- Por columnas. Dado un recurso se establecen unos permisos para los usuarios del sistema. Cuando se crea un recurso se le asigna una lista de permisos. Esta lista de permisos es conocida como ACL e indica para cada recurso quien está autorizado a accederlo y en qué condiciones. Es importante que exista una forma de expresar “el resto”, ya que de lo contrario resultaría tedioso tener que indicar las capacidades de cada uno de los usuarios para un recurso. En Linux se puede establecer la ACL de un fichero mediante el comando setfacl.

Bit suid y sticky

Los permisos de los archivos en Unix se corresponden con un número en octal que varían entre 000 y 777; sin embargo, existen unos permisos especiales que hacen variar ese número entre 0000 y 7777; se trata de los bits de permanencia (1000), SGID (2000) y SUID (4000).

El bit SUID o setuid se activa sobre un fichero añadiéndole 4000 a la representación octal de los permisos del archivo y otorgándole además permiso de ejecución al propietario del mismo; al hacer esto, en lugar de la x en la primera terna de los permisos, aparecerá una s o S sino hemos otorgado el permiso de ejecución correspondiente.

El bit SUID activado sobre un fichero indica que todo aquél que ejecute el archivo va a tener durante la ejecución los mismos privilegios que quién lo creó. Para ello se utilizan el euid (effective user identifier), que identifica al usuario bajo el cual se comporta el proceso, y el ruid (real user identifier), que identifica al usuario que lanzó la ejecución. A continuación se describe el valor de cada uno de estos campos al ejecutarse un programa normal y un programa seguid:

	Programa Normal	**Programa Setuid**
euid	usuario	propietario del fichero
ruid	usuario	usuario

Esto también es aplicable al bit setgid pero, en este caso, a nivel de grupos del fichero en lugar de propietario.

Los bits de setuid y setgid dan a Unix una gran flexibilidad, pero constituyen al mismo tiempo la mayor fuente de ataques internos al sistema. Cualquier sistema Unix tiene un cierto número de ejecutables setuidados y/o setgiados. Cada uno de ellos se ejecuta con los privilegios de quien lo creó (usualmente el root) lo que directamente implica que cualquier usuario tiene la capacidad en modo privilegiado si es el administrador quien creó los ejecutables. Evidentemente, estas tareas han de estar controladas de una forma exhaustiva, ya que si una de ellas se comporta de forma anormal puede causar daños irreparables al sistema.

Es por esto que conviene estar atentos a los nuevos ficheros de estas características que se localicen en la máquina. Demasiadas aplicaciones de Unix se instalan por defecto con ejecutables setuidados cuando realmente no es necesario por lo que, a la hora de instalar nuevo software o actualizar el existente, hemos de acordarnos de resetear el bit de los ficheros que no lo necesiten.

Este tipo de ficheros a pesar de presentar ciertos riesgos son estrictamente necesarios en Unix. Un ejemplo clásico es el fichero /bin/passwd. Este fichero, entre otras, tiene la función de modificar el fichero de claves (/etc/shadow). Como no resulta una solución apropiada dar permisos para todos los usuarios al fichero de contraseñas, lo que se hace es activar el bit setuid.

Por otra parte, el sticky bit o bit de permanencia se activa sumándole 1000 a la representación octal de los permisos de un determinado archivo y otorgándole además permiso de ejecución; si hacemos esto, en lugar de una x en la terna correspondiente al resto de usuarios aparece una t (si no le hemos dado permiso de ejecución al archivo aparecerá una T).

Si el bit de permanencia de un fichero está activado le estamos indicando al sistema operativo que se trata de un archivo muy utilizado, por lo que es conveniente que permanezca en memoria principal el mayor tiempo posible; esta opción se utilizaba en sistemas antiguos que disponían de muy poca RAM.

Lo que sí que sigue vigente es el efecto del sticky bit activado sobre un directorio. En este caso se indica al sistema operativo que, aunque los permisos normales digan que cualquier usuario puede crear y eliminar ficheros, sólo el propietario de cierto archivo y el administrador pueden borrar un archivo guardado en un directorio con esas características. Este bit, que sólo tiene efecto cuando es

activado por el administrador, se utiliza principalmente en directorios del sistema de ficheros en los que interesa que todos puedan escribir pero que no todos puedan borrar los datos escritos como /tmp.

SELinux

SELinux (Security Enhaced Linux) es un Proyecto de la Agencia de Seguridad Nacional de los Estados Unidos que, por medio de parches que modifican el kernel del sistema operativo GNU/Linux, fortalece los mecanismos de control de acceso y fuerza la ejecución de los procesos dentro de un entorno con los mínimos privilegios necesarios.

Cuenta con una arquitectura de seguridad integrada en el kernel 2.6.x, usando los módulos de seguridad GNU/Linux, conocidos como Linux Security Modules (LSM) y se integra muy bien con las políticas de seguridad de cualquier distribución GNU/Linux.

La configuración de SELinux resulta compleja, pero una vez realizada supone una protección importante para el sistema.

Copias de seguridad

Las copias de seguridad son el mecanismo más cómodo y más utilizado para restaurar el sistema en caso de que se produzca una pérdida de datos. Esta pérdida de información se puede producir tanto por un fallo del soporte en el que se almacena la información como por una intrusión a nuestro sistema o un uso incorrecto malintencionado por parte del administrador o de algún usuario que suponga una amenaza para la información.

Existen varios problemas asociados a los backups. Como son por ejemplo la verificación de las copias realizadas, la política de etiquetado, la ubicación de las copias de seguridad y la definición del contenido de los backups.

Políticas

La forma más elemental de realizar una copia de seguridad consiste simplemente en volcar los archivos a salvaguardar a un dispositivo de backup. Esta forma de realizar backups volcando en el dispositivo de copia los archivos o directorios deseados se denomina copia de seguridad completa.

Las copias completas presentan graves inconvenientes; uno de ellos es la dificultad para restaurar ficheros si utilizamos múltiples dispositivos de copia de seguridad. Otro inconveniente, aún más importante, es la cantidad de recursos que consumen. Para solucionar este problema se introduce el concepto de backup incremental o progresivo que consiste en copiar solamente los archivos que han cambiado desde la realización de otra copia. Aplicando esta técnica se ahorra tiempo en la realización de los backups, pero a cambio, es más complicado la tarea de recuperación de ficheros.

De esta forma, parece lógico que la estrategia a seguir sea una combinación de las mencionadas estrategias. El medio de almacenamiento también es importante a la hora de diseñar una política de copias de seguridad correcta. Si se trata de dispositivos baratos, como los CD-ROMs no suele haber muchos problemas, ya que para cada volcado (sea del tipo que sea) se utiliza uno que no se suele volver a utilizar a no ser que se necesite recuperar los datos. No obstante, algo muy diferente son los medios de almacenamiento más caros, generalmente las cintas magnéticas, donde lo habitual es reutilizar unidades sobreescribiendo las copias de seguridad más antiguas con otras más actualizadas.

Dispositivos

Respecto al dispositivo de almacenamiento de las copias de seguridad, hay que tener en cuenta que actualmente existe una gran variedad, no obstante todos han de cumplir una norma básica: el medio elegido ha de ser estándar, de esta forma se evita cualquier problema que puede acontecer referente al hardware en la recuperación.

- Discos flexibles. Es un medio bastante barato y portable entre diferente sistemas operativos. En cambio, su fiabilidad es muy baja y su capacidad de almacenamiento reducida.
- Discos duros. Este tipo de dispositivos tiene una capacidad considerablemente mayor. A veces, incluso, puede resultar interesante hacer una copia idéntica del disco duro instalado en el sistema.
- Cintas magnéticas. Han sido desde antaño el medio de backup por excelencia aunque en la actualidad está en desuso a pesar de su alta fiabilidad y su relativa velocidad de trabajo, la capacidad es bastante limitada.
- CD-ROMs. Son el medio más usado actualmente, ya que los requisitos de hardware son baratos y además utiliza dispositivos de bajo coste y con una capacidad de almacenamiento suficiente para muchos sistemas
- Memorias USB. Es un medio muy utilizado para realizar copias de seguridad por su bajo coste y gran capacidad de almacenamiento, pero hay que tener en cuenta que es muy mala opción por su alta tasa de fallos.

Software

Existen gran variedad de herramientas para realizar backups, pero la mayoría de ellas suelen presentar un grave problema a la hora de recuperar archivos, ya que en muchos casos se trata de software propietario, por lo que si queremos restaurar total o parcialmente archivos almacenados con este tipo de programas necesitamos el propio programa para hacerlo.

Es por este motivo, por el que muchos administradores optan por realizar las copias de seguridad utilizando herramientas estándar, como un shellscript. Algunos de los comandos más utilizados son:

- tar. Es una herramienta de fácil manejo que permite volcar ficheros o directorios completos en un único fichero. Su principal desventaja es que, bajo ciertas condiciones, si falla una porción del medio se puede perder toda la copia de seguridad; además, tar no es capaz de realizar por sí mismo más que copias de seguridad completas, por lo que hace falta un poco de programación para realizar copias progresivas o diferenciales. Su sintaxis es: tar –cf archivotar archivoa directorio archivob…
- cpio. Permite copiar archivos a o desde un contenedor cpio, que no es más que un fichero que almacena otros archivos e información sobre ellos (permisos, nombres, propietario …). Este contenedor puede ser un disco, otro archivo o una cinta, mientras que los ficheros a copias pueden ser archivos normales, pero también dispositivos o sistemas de ficheros completos. La sintaxis de esta orden es bastante más confusa que la de tar debido a la interpretación de lo que cpio entiende por dentro y fuera: copiar fuera es generar un contenedor en salida estándar, mientras que copiar dentro es extraer archivos de la entrada estándar.
- rsync. Es una forma alternativa de hacer copias de seguridad. Tiene las propiedades de recuperación total y parcial y permite mantener fotos, lo que impide realizar copias incrementales. rsync tiene la opción de backup, de forma que si se le pide hacer una copia de un fichero ya existente, crea uno nuevo con otro nombre en el mismo directorio o lo almacena en otro. Esto resulta de gran utilidad si se quieren recuperar versiones anteriores de la

información.

Ubicación

Otro aspecto importante es el lugar donde las copias de seguridad se almacenan. Es conveniente almacenar las copias de seguridad, si es posible, en el lugar más alejado del sistema informático. En ocasiones por comodidad del responsable de realizar las copias de seguridad, éstas se guardan cerca del sistema informático, lo que supone un gran error y una gran ventaja para el atacante ya que, en caso de que una amenaza se haga efectiva y se dañen los datos, también se dañarían los datos de la propia copia, y , además, es el lugar más evidente donde un atacante registrará para la obtención de más información acerca del sistema. La ubicación de las copias se clasifica en:

- Locales. Para almacenar las copias de seguridad utilizamos uno de los múltiples dispositivos periféricos de que el sistema informático dispone localmente.
- Remotas. Las copias de seguridad sobre la información, se almacenan en los dispositivos de almacenamiento de un sistema remoto. En este caso, la seguridad de nuestra política de copias está supeditada a las medidas de seguridad que el sistema remoto emplee para proteger la información correspondiente. Sin embargo, supone una ventaja sobre el anterior método de implementación de copias de seguridad, ya que la probabilidad de que los dos dispositivos de almacenamiento (los locales y remotos) fallen simultáneamente es muy pequeña. A pesar de esta ventaja, se puede producir una violación del principio básico de confidencialidad usando este método, ya que si no somos los administradores de la máquina remota la información quedará expuesta a individuos ajenos a ella. Para evitar este problema basta con realizar una copia de los datos cifrados.

Otras consideraciones

A la hora de realizar una copia de seguridad se debe almacenar aquella información realmente importante, ya que de lo contrario se estará ocupando espacio en los soportes de copia con información irrelevante. La información que se debe almacenar en la copia de seguridad será aquella que se haya valorado como importante tras el análisis de riesgos.

También resulta de gran importancia almacenar la configuración del sistema, ya que esto permitirá la recuperación del mismo con la configuración establecida rápidamente.

start.txt · Last modified: 2010/07/09 18:17 by sandra