1. INTRODUCCIÓN A LA CRIPTOGRÁFIA

La utilización del cifrado, con el objetivo de impedir que otros puedan entender la información contenida en algún mensaje, tiene sus orígenes en la antigüedad. Hasta la llegada de las computadoras, el principal impedimento para la utilización de la criptografía era la limitada capacidad humana de realizar las transformaciones necesarias en el tiempo requerido y además dar la posibilidad de cambiar frecuentemente el código utilizado.

La siguiente figura muestra el esquema de cifrado válido desde Julio César hasta nuestros días.

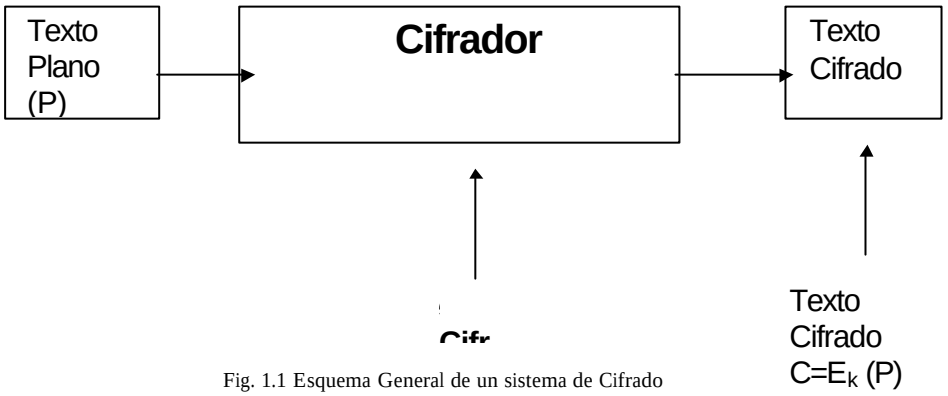


Fig. 1.1 Esquema General de un sistema de Cifrado

De forma análoga podemos revertir el proceso para obtener el proceso de descifrado. Si llamamos D_k a la función de descifrado podemos escribir:

$$D_k(E_k(P))=P$$

De la notación anterior se desprende que E y D son funciones matemáticas de dos parámetros: la clave de cifrado / descifrado **K** y el mensaje **P**. Por regla general el protocolo de trabajo señalado por E es público. Sobre este particular es larga la lista de algoritmos que han intentado mantenerse secretos y han sido "sorpresivamente" descritos a través de Internet. Por tal razón es consenso general entre los diseñadores que la fortaleza de los algoritmos no debe descansar en el secreto de su estructura sino en la complejidad de esta y la longitud de la clave.

La verdadera llave para poder obtener P desde el texto cifrado es la clave K. La longitud de K es sumamente importante, una clave por ejemplo de 6 bits significan 2^6 , 64 posibles valores,, muy

pocos incluso para Julio Cesar. En cambio una clave de 128 bits significan 3.402823669209e+38 posibilidades, muchas incluso quizás, para las computadoras de nuestros hijos.

Mientras mayor sea la clave mayor será el factor de trabajo necesario para descifrar el texto cifrado. El secreto de un buen algoritmo de codificación está en tener un algoritmo robusto y una clave suficientemente grande.

1.1 Técnicas Básicas de Cifrado

Existen 2 técnicas básicas de cifrado que se utilizan de alguna forma en todos los algoritmos de cifrado que existen:

- Cifrado por Sustitución
- Cifrado por Transposición

El cifrado por sustitución consiste en remplazar un símbolo por otro según determinada clave. Un ejemplo sencillo se obtiene si pensamos en el alfabeto, la clave podría ser el número que ha de desplazarse el carácter para obtener el correspondiente en el texto cifrado.

Podemos caracterizar el mecanismo de cifrado como:

$C = E_k(p) = (p+k) \bmod 26$

Mientras que el algoritmo de descifrado puede ser visto como:

$P = D_k(C) = (c-k) \bmod 26$

Este método de cifrado donde el alfabeto de origen y destino s on los mismos se llama sustitución monoalfabética.

Los cifrados por sustitución monoalfabética tienen varias deficiencias que los hacen inservibles por si solos hoy en día:

1. Los algoritmos de cifrado y descifrado son muy sencillos y sumamente conocidos hoy en día
2. La cantidad de claves elegibles es muy pequeña.
3. El lenguaje del texto plano influye estadísticamente en la estructura del texto cifrado. Por ejemplo en castellano la vocal más común es la *e* por lo tanto el símbolo que identifique a esta letra en el alfabeto de salida será el más común en el texto cifrado. Un muy buen dato para un criptoanalista.

Una forma de aumentar la fortaleza de los algoritmos de cifrado por sustitución es utilizar más de un alfabeto de salida. Todos estos algoritmos se basan en las siguientes reglas:

1. Existe un número n ($0 < n < ?$) de reglas de sustitución monoalfabéticas

2. Una clave determina cual de las n posibles reglas es escogida para una transformación.h

El más conocido de estos algoritmos es el Vigenére. En este esquema el conjunto de posibles reglas consisten en los 26 posibles esquemas monoalfabéticos vistos anteriormente cada uno identificado por una clave de 0 a 25. El esquema de cifrado es sencillo, dado un texto plano p de m símbolos cada uno de ellos será cifrado por m de los posibles n esquemas monoalfabéticos. La clave aquí no define cual es el desplazamiento dentro del alfabeto de entrada, sino cual de los n esquemas será utilizado.

En los cifrados por transposición la idea es cambiar (de nuevo según una clave) el orden en que aparecen los símbolos del mensaje original en el mensaje cifrado. Al igual que los cifrados por sustitución, los cifrados por transposición por si solos son vulnerables.

M	E	G	A	S	I	T	O
5	*2*	*3*	*1*	*7*	*4*	*8*	*6*
P	A	C	A	E	D	E	R
O	V	E	R	N	E	C	H
R	O	R	C	T	L	T	O
F	R	R	U	A	R	O	Y

Fig. 1.2 Cifrado por Transposición.

Clave: MEGASITO TEXTO PLANO: Por favor cerrar cuenta del rector hoy

TEXTO CIFRADO: **arcuavorcerrdelrporfrhventaecto**

1.2. Relleno de una vez

La clave más efectiva, si hablamos de su estructura y no de su longitud, es aquella que sea tan larga como el texto a cifrar, de forma que podamos asociar un termino de la clave con uno del texto plano y no posea relación estadística ninguna con este. Un método que utiliza esta técnica fue propuesto por un ingeniero de AT & T llamado Gilbert Vernam y mejorado por un oficial del Army Signal Corp. llamado Joseph Mauborgne. El método sigue una regla de transformación sumamente sencilla, para el bit i de un texto plano cualquiera :

$C_i = p_i \oplus k_i$

Donde $\oplus$ es la operación OR EXCLUSIVO

Para descifrar la operación es:
$P_i = C_i \oplus k_i$

Es decir el texto cifrado C se va obteniendo bit a bit mediante el XOR de un bit de texto plano y un bit de la clave.

En el esquema original de Vernam la clave K tenía tamaño fijo, de forma que si m era la longitud del mensaje y n la longitud de la clave, al momento de cifrar se generaba una clave de m / n K bits. Esto facilita el manejo de la clave pero sin dudas debilita notablemente el algoritmo. Mauborgne propone generar una clave de forma aleatoria y de longitud m. Obtenido de esta última forma, el texto cifrado es inviolable.

La dificultad práctica se debe a la propia estructura de la clave ya que hay generar, mantener y proteger una clave, cuya longitud es tan grande como el propio texto. Esta dificultad ha hecho al algoritmo poco práctico. Sin embargo para condiciones en las cuales el tamaño del texto a cifrar es pequeño, conocido y fijo resulta una alternativa eficiente, sencilla y fuerte. El autor utiliza este algoritmo reiteradamente en su trabajo como administrador de sistemas.

MODELOS ASIMÉTRICOS DE CIFRADO

Como hemos mencionado la principal deficiencia de los modelos simétricos de cifrado es el mantenimiento de la clave, que es la misma para cifrar y descifrar. Para resolver esta dificultad se utilizan los modelos asimétricos.

Los algoritmos de clave asimétrica, también conocidos como de clave pública, utilizan diferentes claves para cifrar y descifrar el mensaje. En este tipo de cifrado cada usuario posee dos claves: una privada que sólo él conoce y una publica que puede ser conocida por todos.

Los modelos asimétricos siguen varias normas bien establecidas:

- Teniendo una clave es imposible obtener la otra. Es decir podemos difundir libremente la clave pública sin peligro de que a través de esta se llegue a conocer la privada.

- Si un mensaje es cifrado utilizando una de las claves solamente puede ser descifrado con su pareja. Es decir si un mensaje es cifrado utilizando la clave publica de un usuario X, sólo puede ser descifrada por el usuario X, ya que nadie más conoce su clave privada.

- Si un mensaje puede ser descifrado satisfactoriamente con una clave es condición inequívoca de que fue cifrado con su pareja.

Veamos un ejemplo. Supongamos que el usuario Juan desea enviar un mensaje cifrado a María. Juan conoce 2 claves : su clave privada y la clave publica de María. Para cifrar el mensaje, Juan utilizará la clave pública de María. Ahora María podrá descifrar el mensaje utilizando su clave privada. Tanto Juan como María pueden estar tranquilos de que sólo ellos conocen el contenido del mensaje, ya que este fue cifrado con una clave que únicamente permitía que fuese descifrado por María, debido a que sólo ella conoce su clave privada.

Si María desea enviar la respuesta del mensaje a José todo lo que tiene que hacer es invertir el mecanismo, es decir, cifrar el mensaje de respuesta con la clave pública de José, quien lo descifrará utilizando su clave privada.

Pero no es solo la confidencialidad del mensaje lo que nos interesa. Existen otros elementos que desearíamos resolver:

- ¿Cómo sabe María que fue realmente Juan quien envió el mensaje?

- ¿Cómo sabe María que el mensaje no fue cambiado en el camino?

Volveremos sobre estos temas más tarde.

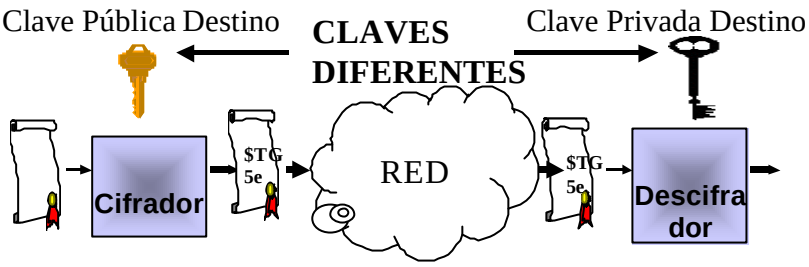


Fig 4.1 Cifrado con claves asimétricas

3.2 RSA

El algoritmo de clave asimétrica más utilizado actualmente fue publicado en 1977 y lleva por nombre las iniciales de sus tres creadores Ronal Rivets, Adi Shamir y Leonard Adleman, RSA

La fortaleza de RSA está basada en la dificultad de factorizar dos números suficientemente grandes.

Veamos cómo funciona RSA :

1) Escojamos dos números primos suficientemente grandes a los que llamaremos *p* y *q*. Al producto de ambos números lo llamaremos *n*.

2) Elijamos un número *e* tal que sea relativamente primo a *(p-1) * (q-1)* , o sea a ? (n)

3) Encontremos ahora un número d tal que *(e*d) -1* sea divisible entre *(p-1) * (q-1)*

4) Se puede probar utilizando los teoremas de Fermat y Euler, que para cualquier número m menor que n,

$$m^{e*d} \bmod n = m \bmod n = m$$

Para nuestros fines m sería el mensaje a cifrar. Quedaría entonces :

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

Donde c es el mensaje cifrado.

C puede ser descifrado utilizando *d* de la siguiente forma :

$c^{d} \bmod n = m^{e*d} \bmod n = m \bmod n = m$

Así que tenemos un mecanismo para codificar y descodificar en el que:

e y **d** conforman la clave pública

d y **n** conforman la clave privada

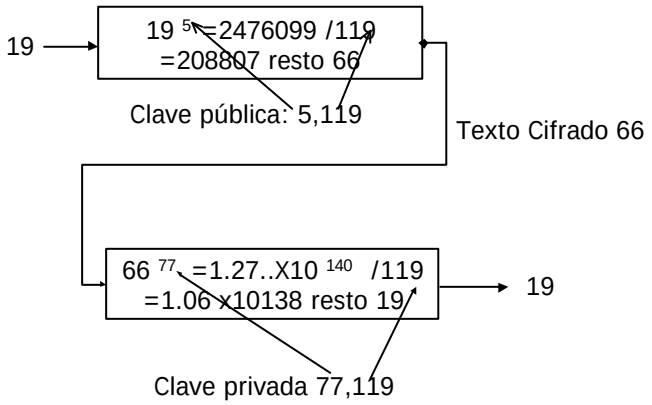


Fig 4.2 Ejemplo de funcionamiento de RSA

La fortaleza de RSA radica en la dificultad de factorizar números grandes. Para mostrar esto calculemos el tiempo necesario, utilizando las mejores técnicas disponibles hoy en día, para factorizar un número de 200 dígitos decimales. Para representar este valor se necesitan al alrededor de 666 dígitos binarios. Los algoritmos más eficientes requieren aproximadamente 1.2×10^{23} operaciones para factorizar un número de este tamaño.

Asumamos que tenemos una máquina capaz de realizar 10^{10} operaciones por segundo, sólo de factorización. Para realizar 1.2×10^{23} operaciones requiere de 1.2×10^{23} segundos, es decir 380267 años.

Si escogemos un número no de 200 cifras sino de 400 cifras, el tiempo requerido es de 8.6×10^{15} años. ¡La edad estimada del universo cabe 430 000 veces en este número!

4.3 Otra forma de ver el mecanismo

El mecanismo de clave pública implementado como hemos explicado hasta ahora permite asegurar la comunicación de la interferencia de intrusos Sin embargo, todavía nos queda un problema por resolver : ¿ Cómo sabemos que el remitente de un mensaje es quien dice ser y no ha sido suplantado?

Como hemos visto hasta ahora, cualquiera pudiese realizar esta tarea pues todo lo que necesita para enviar un mensaje cifrado es obtener una clave que todos conocen, es pública.

¿Y si invertimos en mecanismo? Es decir, utilizamos la clave privada del remitente para cifrar los mensajes y en consecuencia en el receptor desciframos el mensaje utilizando la clave pública del remitente.

El algoritmo así implementado tiene poca utilidad si lo que se desea es proteger la confidencialidad de un mensaje, por cuanto si un mensaje es cifrado con una clave privada puede ser descodificado con una clave que todos conocen: la clave pública. Sin embargo, este segundo modelo no es tan inservible como aparenta. Si un mensaje es cifrado de esta forma, el receptor no podrá estar seguro de que la información no ha sido leída por un intruso, pero, si él logra descifrar correctamente el mensaje utilizando la clave pública del supuesto remitente puede confiar en que este es realmente quien dice ser, ya que sólo él conoce su propia clave privada. El punto clave está en la palabra “correctamente”, sobre este particular volveremos más adelante.

Un mensaje así es como el sello de un documento, nos da garantía de quien lo ha emitido, pero no de su contenido.

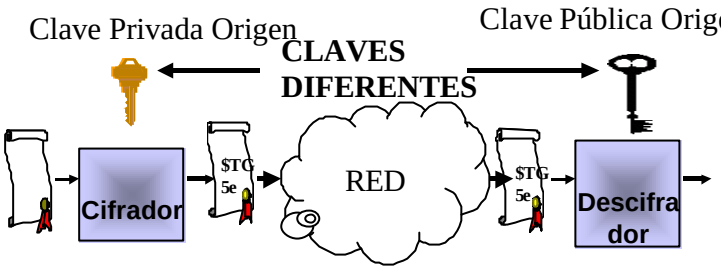


Fig 4.3 Cifrado con claves asimétricas, para autenticar origen del mensaje

Característica	Primer Esquema	Segundo Esquema
Claves involucradas		
Cifrado	Clave Pública del destino	Clave Privada del Origen
Descifrado	Clave Privada del Destino	Clave Pública del Origen
Confidencialidad	Completa	Nula
Autenticación del Origen	Nula	Completa
Garantía de que el mensaje no ha sido adulterado	Media	Nula
Costo Computacional	Alto	Alto

Tabla 4.1 Comparación entre esquemas de cifrado asimétrico

4.4 Firmas Digitales

Volvamos al tema de descifrar correctamente el mensaje que ha sido cifrado con el segundo mecanismo. El segundo esquema basa su efectividad para definir si el origen es quien pretende ser en dos elementos:

- La clave privada de un usuario solo debe ser conocida por *él*.. Sobre este particular volveremos más adelante.

- Debemos tener un mecanismo para definir si un mensaje no ha sido cambiado desde que el emisor lo transmitió.

Claramente podríamos trasmitir el mensaje dos veces, cada una cifrado con un esquema, y luego comparar los resultados luego de descifrarlos ambos con las claves correspondientes. No hay que pensar mucho para darse cuenta de los inconvenientes de este esquema:

- Altísimo costo computacional

- Una vez que enviemos en mensaje cifrado con el segundo esquema un intruso no necesitará del primer mensaje para conocer su contenido.

Necesitamos entonces un mecanismo diferente para obtener los resultados deseados.

Una función *hash* , resumen o compendio toma una cadena de entrada, usualmente de longitud no definida, y produce un número de salida que es función de la entrada. Las funciones *hash* producen una única salida para la misma entrada, es decir son determinísticas.

Existen tres propiedades adicionales que hacen especialmente útiles las funciones *hash:*

- El algoritmo es de una sola vía, es decir, es muy difícil obtener el mensaje de entrada teniendo el número de salida de la función.

- Un pequeño cambio en el mensaje de entrada (incluso de un sólo bit) produce un número *hash* totalmente diferente.

- Un bit de la salida es producto de las características de varios bits de la entrada, por lo tanto se hace difícil encontrar patrones estadísticos del texto original .

Imaginemos ahora que calculamos el *hash* de un documento y luego lo ciframos, utilizando un algoritmo de clave asimétrica con nuestra clave privada. Ahora trasmitimos el mensaje original y el *hash* cifrado. Quien reciba nuestro mensaje si descifra el *hash* , calcula el *hash* del mensaje recibido y obtiene el mismo número puede obtener buenas noticias:

- El mensaje original no ha sido cambiado, de lo contrario el *hash* calculado por él no puede ser igual al incluido en el mensaje original.

- El origen del mensaje es correcto pues de lo contrario el *hash* descifrado no puede ser igual al calculado por él.

La ventaja de cifrar sólo el *hash* y no todo el documento salta a la vista: eficiencia. Es mucho más sencillo cifrar un simple número y no todo el documento. Producir el *hash* a partir del mensaje de entrada requiere muchos menos recursos computacionales y por tanto menos tiempo de cómputo.

Si además se desea que nadie pueda leer el mensaje todo lo que habría que hacer es cifrar el mensaje con el primer algoritmo y luego trasmitirlo con su *hash*.

Un mensaje al que se ha agregado el *hash* cifrado se dice que está *Digitalmente Firmado*. Las firmas digitales no sirven solo para autenticar mensajes pueden ser utilizados para probar la autenticidad de cualquier documento electrónico. Por ejemplo es importante para un usuario tener una prueba de que el archivo que ha obtenido por ftp no ha sido modificado ni su origen es apócrifo.

Una firma digital debe permitir además de verificar el autor del documento, tener refencia de la hora y fecha en que fue estampada en el mismo. Adicionalmente la firma digital debe ser verificable por terceros.

4.3.1 MD5

El algoritmo de *hash* más utilizado en estos momentos es el MD5. Este algoritmo fue desarrollado por Ronald Rivest en 1995 y está basado en dos algoritmos anteriores MD2 y MD4.

Todos estos protocolos producen un número de 128 bits a partir de un texto de cualquier longitud.

MD4 fue desarrollado para mejorar el rendimiento de MD2, sin embargo, varios problemas fueron detectados y en 1996 fueron publicados elementos que hacen hoy en día inservible el algoritmo. MD5 sustituyó a MD4 y aunque no tiene el rendimiento de su antecesor, hasta el momento no han sido publicados elementos que comprometan su integridad y funcionamiento.

MD5 comienza rellenando el mensaje a una longitud congruente en módulo 448 mod 512.Es decir la longitud del mensaje es 64 bits menos que un entero múltiplo de 512. El relleno consiste en un bit en 1 seguido por cuentos bits en 0 sean necesarios. La longitud original del mensaje es almacenada en los últimos 64 bits del relleno.

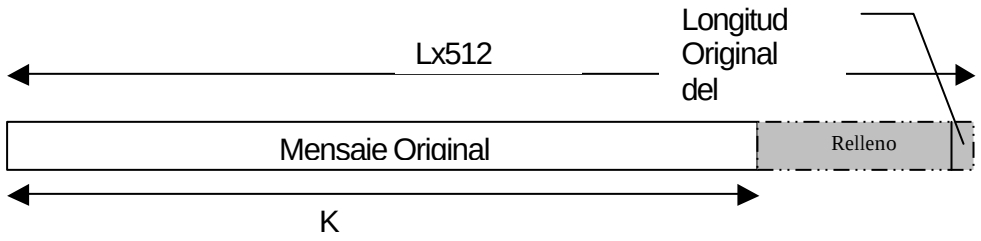


Fig 4.4 Relleno del mensaje original hasta múltiplo de 512 en MD5

Adicionalmente se inicializa, con un valor fijo, un buffer de 128 bits. Este buffer puede verse como 4 registros de 32 bits (A,B,C,D) y son inicializados con los siguientes valores hexadecimales:

A=67452301 B=EFCDAB89

C=98BADCFE

D=10325476

Durante varias rondas de procesamiento el algoritmo toma bloques de 512 bits de la entrada y los mezcla con los 128 bits del buffer. Este proceso es repetido hasta que todos los bloques de entrada han sido consumidos. El valor resultante en el buffer es el *hash* del mensaje.

MD5 no es el único algoritmo de *hash* conocido. Existe otra función llamada Secure *Hash* Algorithm, (SHA), desarrollado por NSA.. A diferencia de MD5, SHA genera números *hash* de 160 bits.

4.3.2 El Ataque de Cumpleaños

Aparentemente las funciones *hash* son suficientemente seguras. Por ejemplo si un *hash* cifrado C es trasmitido con el correspondiente mensaje no cifrado M, un intruso tendría que encontrar un mensaje M' tal que H(M') = H(M). Si lo obtiene podría sustituir M por M' y engañar al receptor. Como promedio el atacante debe probar 2^m posibles mensajes, siendo m la longitud de la función *hash.*

Sin embargo una segunda aproximación es posible, basada en la paradoja del cumpleaños, un tema clásico de probabilidades. Esta segunda estrategia es como sigue:

- El origen A, está preparado para firmar un mensaje obteniendo el *hash* del mismo y cifrándolo con su clave privada.

- El atacante genera $2^{m/2}$ posibles variaciones del mensaje que conllevan al mismo significado. Por ejemplo se puede sustituir un simple espacio por la cadena "espacio-borrado hacia atrás- espacio"

- El atacante prepara igual cantidad de mensajes fraudulentos .

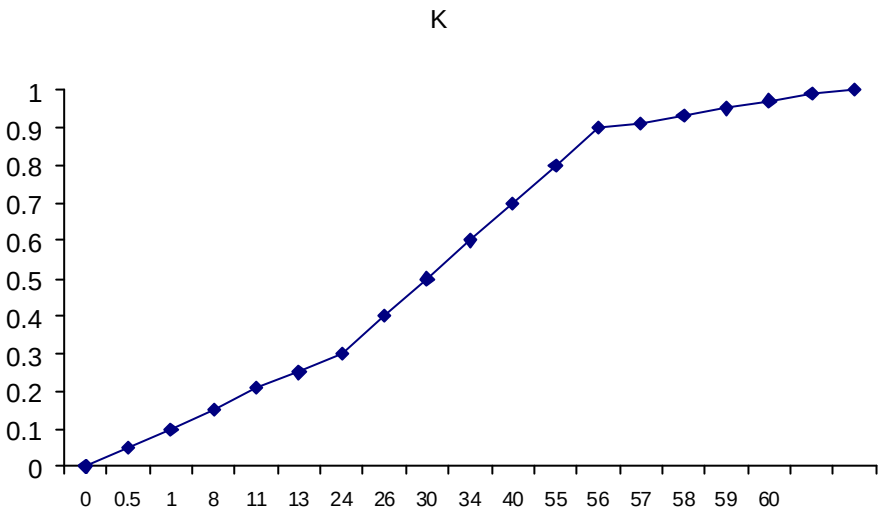
- Los dos conjuntos de mensajes son comparados para encontrar una pareja que produzca el mismo *hash*. La probabilidad de que esto ocurra, por la paradoja del cumpleaños, es de 0.5.

- Si no se encuentra coincidencia mas mensajes son generados.

- Si se encuentra coincidencia el oponente ha obtenido un *hash* válido para un mensaje falso.

La paradoja del cumpleaños

¿Cuál es el mínimo valor de K tal que la probabilidad de que en un grupo de K personas al menos 2 cumplan años el mismo día sea mayor de 0.5?



4.5 Algoritmo De Deffie-Hellman para el Intercambio de seguro de claves

Hasta ahora hemos basado nuestras conclusiones sobre la seguridad de los algoritmos de clave asimétrica en el supuesto de que la clave privada nunca es conocida por nadie más que su propio dueño.

Una alternativa sencilla para obtener esto es que cada usuario genere por separado pero como veremos más adelante este mecanismo trae consigo más problemas que los que resuelve. La alternativa más ampliamente aceptada es que existan instituciones que se dediquen a la tarea de generar y brindar a los usuarios las claves que necesiten, tanto las suyas propias como las de otros (las públicas claro).

Bajo este esquema es necesario un mecanismo para asegurar que una vez que las claves son generadas estas no puedan ser interceptadas mientras viajan a su dueño.

El algoritmo de Diffie-Helman propone un método sencillo y efectivo para el intercambio seguro de claves. Su efectividad depende de la dificultad de computar logaritmos discretos.

Antes de continuar definamos algunos elementos necesarios:

- La raíz , a, de un número primo p como uno de los siguientes valores:

$$a \bmod p,\ a^2 \bmod p,\ \ldots..,\ a^{p-1} \bmod p$$

Con este elemento claro comencemos a definir el algoritmo. Existen dos números públicos: q un número primo cualquiera y ? entero raíz primitiva de q. Supongamos que los usuarios A y B desean intercambiar claves.

1) El usuario A selecciona aleatoriamente un entero $X_a < q$
2) El usuario A computa $Y_a = ?^{Xa} \bmod q$
3) El usuario B selecciona aleatoriamente un entero $X_b < q$
4) El usuario B computa $Y_b = ?^{Xb} \bmod q$
5) Cada lado hace público Y y mantiene en secreto X.
6) El usuario A computa la clave como $K=(Y_b)^{Xa}$, mientras el lado B lo hace $K=(Y_a)^{Xb}$. Ambos cálculos conducen a idénticos resultados:

$$K=(Y_b)^{Xa} \bmod q$$
$$=(?^{xb} \bmod q)^{Xa} \bmod q$$
$$=(?^{xb})^{Xa} \bmod q$$
$$=?^{XaXb} \bmod q$$
$$=(?^{xa})^{Xb} \bmod q$$
$$=(?^{Xa} \bmod q)^{Xb} \bmod q$$
$$=(Y_a)^{Xb} \bmod q$$

Como vemos, ambos lados pueden llegar a la clave generada por el otro. Como X_a y X_b son secretos el intruso tendría resolver de forma inversa la siguiente ecuación:

$Y_b = ?^{Xb} \bmod q$ suponiendo un ataque a la clave privada de B

Lo cual es una tarea sumamente difícil si se trata de números suficientemente grandes.

4.5 Certificados Digitales y Autoridades de Certificación

Como hemos mencionado en la sección anterior el esquema más ampliamente aceptado de distribución de claves involucra a entidades que se encarguen de distribuir las mismas, certificar la autenticidad y vigilar por su integridad y validez. Estas entidades se conocen como Autoridades de Certificación.

Una autoridad de certificación emite Certificados Digitales que contienen:

- La clave pública del usuario
- Fecha de emisión
- Usuario de Destino
- Período de Validez

Los certificados son entregados cifrados con la clave privada de la autoridad de certificación, por lo tanto pueden ser descifrados por cualquiera y adicionalmente puede ser verificada su autenticidad. Cada certificado es entregado por la autoridad de certificación a su dueño junto a su correspondiente clave privada, siguiendo los algoritmos vistos anteriormente.