2017

Actividad 5: Informe sobre cifrado simétrico y funciones hash

SEGURIDAD INFORMÁTICA

ORQUIDEA SEIJAS

TABLA DE CONTENIDO

I. CIFRADO SIMÉTRICO BÁSICO.

Para iniciar la sesión, se ha realizado un ejercicio para observar la diferencia entre cifrado y codificación. En primer lugar, se ha elegido una palabra como clave, de forma arbitraria, para cifrar un documento. Esta clave se ha cifrado en base64 utilizando el siguiente comando:

```
openssl enc -base64 -in file.txt -out filecoded.txt
```

Donde *file.txt* es el archivo que contiene la clave, en este caso *holi*, y *filecoded.txt* contiene la clave codificada en base64. Para comprobar que la codificación era correcta, se ha utilizado el mismo comando con la opción de decodificar:

```
openssl enc -base64 -d -in filecoded.txt -out filedecoded.txt
```

Se obtuvo la misma clave introducida al principio, por lo que se pudo concluir que la codificación había sido exitosa. A continuación, se ha utilizado la palabra escogida para cifrar un documento utilizando DES en modo CBC utilizando el comando:

```
openssl enc -des-cbc -in file.txt -out fileencrypted.txt
```

Donde *file.txt* es el fichero que contiene el texto a encriptar y *fileencrypted.txt* es el fichero que contiende el texto obtenido una vez se realiza la encriptación. Puesto que este archivo iba a ser reenviado a un compañero, se consideró necesario comprobar que la encriptación había sido exitosa y se decidió desencriptarlo:

```
openssl enc -des-cbc -d -in fileencrypted.txt -out filedecrypted.txt
```

A continuación, se recibieron dos ficheros por parte de un compañero: uno contenía la clave codificada y otro contenía el texto cifrado. Se utilizaron los mismos comandos comentados anteriormente para la descodificación de la clave, que es *pizza*, y para la desencriptación del texto, que es el siguiente:

> *El lobo (Canis lupus) es una especie de mamífero placentario del orden de los carnívoros. El perro doméstico (Canis lupus familiaris) se considera miembro de la misma especie según distintos indicios, la secuencia del ADN y otros estudios genéticos. Los lobos fueron antaño abundantes y se distribuían por Norteamérica, Eurasia y el Oriente Medio. Actualmente, por una serie de razones relacionadas con el hombre, incluyendo el muy extendido hábito de la caza, los lobos habitan únicamente en una muy limitada porción del que antes fue su territorio.*

a. DIFERENCIA ENTRE CIFRADO Y CODIFICACIÓN

La diferencia más significativa entre codificación y cifrado es que la transformación aplicada al elemento que se desea proteger:

Si se codifica un texto, simplemente se está realizando un intercambio, más o menos complejo, de un carácter por otro. El resultado de la codificación tiene una equivalencia directa, carácter a carácter, con el texto original, gracias a una serie de normas preestablecidas para alterar la semántica del mensaje.

Por otra parte, el cifrado de un texto, se realiza a través de un algoritmo que puede modificar la extensión y composición original. Solo es posible recuperar el texto cifrado a través de un algoritmo

equivalente. El cifrado suele usar una clave para transformar la estructura, de forma tal, que, si un tercero interviene, no pueda acceder a la información. Este cifrado puede ser simétrico, siendo la clave igual a ambos sentidos de la comunicación, o asimétrico, donde se utiliza un sistema de clave pública y clave privada.

II. CIFRADO/DESCIFRADO

Con el fin de comprender mejor el funcionamiento del cifrado y descifrado, se utilizó el algoritmo DES (*Data Encryption Standard*) en modo CBC (*Cipher Block Chaining Mode*) para cifrar un texto pequeño y uno más grande. Se comparó el valor del vector de inicialización para ambos textos y se pudo comprobar que los valores eran diferentes. Puesto que el cifrado en modo CBC supone que el cifrado de cada bloque dependa del contenido del bloque anterior y no solo de su propio contenido. Este vector se utiliza para, como bien indica su nombre, iniciar el cifrado ya que en el primer bloque no hay un bloque previo para utilizar.

Al probar con el cifrado DES en modo ECB (*Code Book Mode*) se pudo comprobar que, tal y como se esperaba dado su comportamiento, el vector inicial no forma parte de los elementos que se utilizan para el cifrado del vector. Este vector no forma parte del encriptado en este modo, puesto que los bloques se cifran individualmente, dependiendo únicamente de sí mismos. Esto provoca una debilidad, ya que las repeticiones en el texto plano, cifradas con la misma clave, que sí que es común a todos los bloques, originan el mismo texto cifrado. Es decir, que en caso de que se intentara producir un ataque, se proveería al atacante con demasiada información.

a. COMPARACIÓN DEL CIFRADO UTILIZANDO DIFERENTES ALGORITMOS SIMÉTRICOS.

Algoritmo	Longitud de clave	Tamaño de bloque	Tamaño de fichero cifrado	Vector de inicialización
DES	64 bits (56+8 de comprobación)	64 bits	6568 bytes	iv =D3A5D33CD1425C30
3DES	168 bits	64 bits	6568 bytes	iv =8BC3364BCB50829D
AES (128)	128 bits	128 bits	6576 bytes	iv =F6A7A9391A5CBCC851CC027B0070D3FC
AES (192)	192 bits	128 bits	6576 bytes	iv=D694A171547CDC0829C549E9CDF9486A
AES (256)	256 bits	128 bits	6576 bytes	iv =448D5631B42FC1ABDBD0BB55167AFE0C

Para la obtención de la tabla presentada se han utilizado los siguientes comandos:

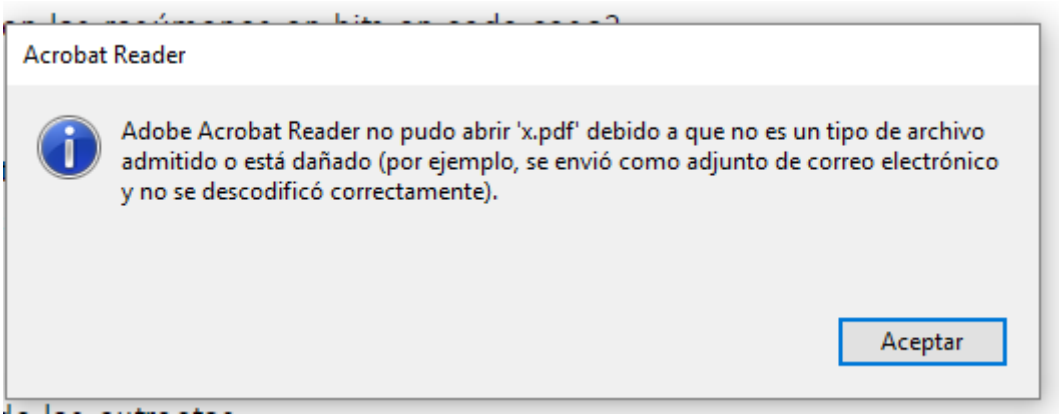
- `openssl enc -p -des-ede3-cbc -in a.txt -out b.txt`
- `openssl enc -p -des-cbc -in a.txt -out b.txt`
- `openssl enc -p -aes-128-cbc -in a.txt -out b.txt`
- `openssl enc -p -aes-192-cbc -in a.txt -out b.txt`
- `openssl enc -p -aes-256-cbc -in a.txt -out b.txt`

b. Aplicación de un algoritmo de cifrado simétrico a un fichero PDF.

Tal y como se explicó en el apartado I de este informe, los algorimos de cifrado transforman la estructura de un documento. Esto provoca que se modifique a tal nivel el archivo, que, de ser un tipo de fichero complejo, deje de seguir los patrones que permiten que se pueda identificar como tal. Por ejemplo, en este caso se probó a encriptar un fichero pdf con el siguiente comando:

```
openssl enc -p -aes-256-cbc -in Actividad\ 5.pdf -out x.pdf
```

A continuación, se intentó abrir el fichero con el lector utilizado habitualmente y saltó este mensaje de error, conforme no se podía abrir el *pdf* ya que se creía corrupto:



Al desencriptar con el siguiente comando se pudo recuperar apropiadamente el fichero y abrirlo en el fichero indicado como *out*:

```
openssl enc -d -aes-256-cbc -in x.pdf -out f.pdf
```

c. Rellenado en los algoritmos de cifrado simétrico

Puesto que algunos algoritmos requieren bloques de bits de cierto tamaño, es necesario añadir un cierto *relleno* o *padding* para conseguir que los bits de un archivo a cifrar coincidan con lo que se desea. Así, el tamaño del mensaje es un múltiplo de los bloques que se utilizan para la encriptación y esta se puede efectuar apropiadamente.

III. Extracto o resumen digital

a. Resumen digital de un documento

A continuación, se obtuvo el extracto digital, o resumen, de un documento de texto plano. Este resumen se obtuvo a través del algoritmo MD5 (*Message Digest 5*) y del algoritmo SHA-1(*Secure Hash Algorithm 1*) con los siguientes comandos:

```
openssl dgst -md5 file.txt
openssl dgst -sha1 file.txt
```

Por un lado, el resumen del algoritmo MD5 (*35899082e51edf667f14477ac000cbba*) es de 128 bits, lo que puede llegar a ser una vulnerabilidad ya que, como se demostró en 2004, es posible realizar un ataque de fuerza bruta en un tiempo computacional razonable. Por otro lado, el resumen del algoritmo de SHA-1 (*e7505beb754bed863e3885f73e3bb6866bdd7f8c*) ocupa 160 bits, lo que, en

comparativa, lo hace más robusto. Sin embargo, se encontraron debilidades criptográficas en este algoritmo y la mayor parte de los estándares criptográficos ya no lo aceptan desde 2010.

A continuación, se modificó un único carácter en el texto y se obtuvieron los resúmenes de nuevo. Ambos resúmenes variaron, ya que se trata de un archivo diferentes y, por lo tanto, el resumen asociado debe serio. Este resumen cambió radicalmente, ya que se supone que debe tener una distribución estadística lo suficientemente grande como para no ser ni parecido al resumen original. Por otra parte, ya que el resumen debe ser siempre de la misma longitud, esta no cambió en ninguno de los dos casos.

b. USOS POSIBLES PARA UN EXTRACTO DIGITAL

Un extracto o resumen digital se puede utilizar:

- Para firmas digitales, de forma que se hace que la firma sea dependiente de todo el mensaje a firmar, en lugar de bloque a bloque (como se hace en los algoritmos de firma por bloques). Además, se mejora la velocidad de firma, ya que los valores hash suelen ser mucho más cortos, y además, las funciones hash destruyen cualquier estructura que pueda ser vulnerable al criptoanálisis.
- Para suma de verificación: para proteger frente a errores de almacenamiento o transmisión. En este caso se utilizan funciones hash con ciertas propiedades de forma que puedan ser utilizadas para verificar hasta cierto punto el dato.
- Para realizar una prueba de la integridad de los contenidos: se proporciona un valor resumen del contenido, de forma que este valor se obtenga al realizar la función resumen sobre el contenido distribuido. A este proceso se le suele llamar *checksum criptográfico.*
- Para realizar autenticación de entidades: el cliente, que se quiere autenticar, genera un mensaje y calcula su valor resumen. Estos datos se envían al servidor, que verifica. En este momento, el servidor comprueba la autenticidad del mensaje con el valor resumen. Así, verifica que el cliente tiene la clave y lo autentifica.
- Almacenamiento seguro de contraseñas: basta con almacenar el resumen digital obtenido tras la aplicación de uno de los algoritmos de obtención de cifrado. Para comprobar si una contraseña es correcta, bastará con comparar los resúmenes.

c. OBTENCIÓN DE EXTRACTO DIGITAL SHA-1 PARA ARCHIVOS MÁS COMPLEJOS.

Con el fin de comparar la longitud de los extractos digitales para SHA-1 en otros casos, seguidamente se obtuvieron para texto plano de mayor tamaño que el anterior, que tan solo era un texto de una línea, y una imagen.

Para el texto plano se obtuvo esta clave: *deb6004ea83a49b43e47124acda417ce1266bc75* y para el fichero asociado a una imagen, se obtuvo: *da08f5fbc1e187bac44986f2e198e17c2ec3baa5*. Tal y como se explicó previamente, la longitud tiene que ser la misma siempre, debido al algoritmo utilizado para calcular el extracto, y en este caso se comprobó que seguía siendo así.

d. USO DE LOS ALGORITMOS SHA-256 Y SHA-512 PARA OBTENCIÓN DE EXTRACTOS.

El algoritmo SHA-1 no es el único *Secure Hash Algorithm*, es por ello por lo que se decidió estudiar el extracto obtenido con otros dos algoritmos de la misma familia. Estos algoritmos son SHA-256, que tiene un extracto de longitud de 256 bits, y SHA-512, que tiene un extracto de longitud de

512 bits. Se han utilizado los algoritmos en todos los archivos de los que se obtuvo un extracto previamente, con los siguientes comandos:

```
openssl dgst -sha512 a.txt
openssl dgst -sha256 a.txt
```

Sustituyendo "*a.txt*" por el archivo del que se quería obtener el extracto digital. Estos extractos son mucho más seguros que los vistos previamente, ya que no se pueden atacar por fuerza bruta en un tiempo computacional razonable.

IV. CIFRADO SIMÉTRICO EN HTTPS

Para finalizar esta sesión, se decidió estudiar el uso del cifrado simétrico en el protocolo HTTPS (*Hyper Text Transfer Protocol Secure*). Se han buscado una serie de ejemplos tales como el Campus virtual, el servidor de correo de la universidad, *Gmail*, *PayPal*, *ABANCA* y *fotoefectos*. Se han utilizado dos navegadores para estas pruebas: *Google Chrome* y *Mozilla Firefox*.

- *Campus virtual:* en este caso, el algoritmo de cifrado simétrico que se utiliza en ambos navegadores es AES 128.
- *Servidor de correo de la Universidad de Santiago de Compostela:* en este caso, para *Google Chrome* se utiliza el algoritmo de cifrado simétrico AES 128 y para *Mozilla Firefox* AES 256.
- *Gmail:* en este caso, el algoritmo de cifrado simétrico que se utiliza en ambos navegadores es AES 128.
- *PayPal:* en este caso, el algoritmo de cifrado simétrico que se utiliza en ambos navegadores es AES 128.
- *ABANCA:* en este caso, el algoritmo de cifrado simétrico que se utiliza en ambos navegadores es AES 256 con HMAC – SHA1. *Google Chrome* advierte que se trata de un cifrado obsoleto.
- *fotoefectos:* en este caso, el algoritmo de cifrado simétrico que se utiliza en ambos navegadores es AES 256.

a. SELECCIÓN DEL ALGORITMO DE CIFRADO

Finalmente, fue necesario estudiar cómo se selecciona el algoritmo de cifrado simétrico a utilizar en una conexión particular. Esta selección se realiza en el momento en el que se establece la conexión SSL. Las metas de esta conexión son: certificar que el cliente está hablando con el servidor apropiado, que ambos involucrados han acordado un algoritmo de cifrado, que se utilizará para intercambiar información, y, finalmente, que ambos involucrados hayan acordado cuántas claves se necesitan para el algoritmo.

Esta fase de conexión se divide en tres fases:

1. **Inicio:** en esta fase, el cliente envía al servidor un mensaje para abrir la conexión. Este mensaje contiene toda la información necesaria para que el servidor se pueda conectar al cliente vía SSL, incluidos varios tipos de cifrado. El servidor responde con otro mensaje de saludo, que contiene información necesaria para el cliente, incluyendo qué cifrado y qué versión de SSL se va a utilizar.

2. **Intercambio de certificado:** una vez el contacto se ha establecido, el servidor tiene que demostrar su identidad utilizando su certificado SSL. El servidor puede pedir el certificado del cliente, pero no es necesario.
3. **Intercambio de clave:** para comunicarse, ambos involucrados tienen que estar de acuerdo en la clave simétrica, que se acuerda a través del uso de encriptado asimétrico y de las claves privada y pública del servidor.

REFERENCIAS

- *¿Codificación o cifrado? Aclaraciones que marcan la diferencia*. C. Gutiérrez Amaya. 7/12/2016. https://www.welivesecurity.com/la-es/2016/12/07/codificacion-o-cifrado-diferencia/ [Última visita: 19/10/2017]
- *Enc.* OpenSSL. https://wiki.openssl.org/index.php/Enc [Última visita: 19/10/2017]
- *Security in Computing*. *Apartado 2.3: Criptography*. *Apartado 6.6: Cryptography in Network security.* 5º edición. C.P. Pfleeger, et al. ISBN: 9780134085043.
- Secure Hash Algorithms. Wikipedia. 2017. https://en.wikipedia.org/wiki/Secure_Hash_Algorithms [Última visita: 20/10/2017]
- MD5. Wikipedia. 2017. https://es.wikipedia.org/wiki/MD5#Seguridad [Última visita: 20/10/2017]
- *Criptografía Simétrica (II): Algoritmos de cifrado simétrico.* 2017. P. Cariñena. Campus Virtual de la Universidad de Santiago de Compostela. [Última visita: 20/10/2017]
- *Funciones hash criptográficas.* 2017. P. Cariñena. Campus Virtual de la Universidad de Santiago de Compostela. [Última visita: 20/10/2017]
- *How does HTTPS actually work?* 2014. R. Heaton. https://robertheaton.com/2014/03/27/how-does-https-actually-work/ [Última visita: 21/10/2017]
- Función Hash. Wikipedia. 2017. https://es.wikipedia.org/wiki/Función_hash#Aplicaciones [Última visita: 20/10/2017]