SEGURIDAD EN SISTEMAS OPERATIVOS

4º Grado en Informática

Curso 2018-19

Práctica 1.- Encriptación/desencriptación en Linux

Sesión 6.- Crifrado de sistemas de archivos. Esteganografía y estegoanálisis.

1.- Sistema de archivos encriptados

En la práctica anterior veíamos herramientas para encriptar/desencriptar un archivo. En esta Sesión veremos como cifrar un sistema de archivos completo, de forma que cuando se monte y se de la clave correspondiente, podamos leer/escribir en el archivos y se desencripten/encripten al vuelo. Por razones de extensión del guion vamos a abordar el caso concreto del cifrado del sistema de archivos de un pendrive. Por tanto, para la realización de la práctica sería deseable disponer de un pendrive en desuso no necesariamente de gran capacidad.

Vamos a estudiar una herramienta común en Ubuntu, **`Cryptsetup`**, que permite que los disco extraíbles encriptados en éste sistema se puedan leer en Windows, algo que es muy útil en el caso soportes extraíbles como *pendrives*.

1º) Instalar `Cryptsetup`

```
% apt-get install cryptsetup
```

2º) Localizamos la partición a encriptar y el dispositivo

```
% fdisk -l
```

nos devolverá algo similar a

```
Dispositivo Inicio    Comienzo      Fin      Bloques  Id  Sistema
/dev/sdf1               1           488     3919828+   b  W95 FAT32
```

En el ejemplo anterior es un pendrive USB de 4Gb formateado en FAT32. Aquí nos interesa la dirección física que Ubuntu a asignado a la partición en nuestro caso /dev/sdf1. Además, si la unidad estuviese montada habría que desmontarla

```
% unmont /punto_de_montaje
```

3º) Realizar el cifrado y la contraseña en el dispositivo

Para realizar el cifrado completo de la partición *sdf1* abrimos un terminal y escribimos:

```
% cryptsetup -c aes -h sha256 -y -s 256 luksFormat /dev/sdf1
WARNING!
========
This will overwrite data on /dev/sdf1 irrevocably.
Are you sure? (Type uppercase yes):
...
```

Comprobamos que nos advierte que vamos a sobrescribir el dispositivo y que se perderán los datos que había en el. Después de aceptar y que el programa escriba los datos necesarios en la partición se nos pedirá una contraseña y la confirmación de ella.

4º) Abrir el dispositivo

Necesitamos que la aplicación cryptsetup abra el dispositivo y lo monte, escribimos

```
% cryptsetup luksOpen /dev/sdf1 test
Enter passphrase for /dev/sdb1:
Key slot 0 unlocked.
```

Ya tenemos acceso al dispositivo como un dispositivo normal, pero nosotros actuaremos como un dispositivo mapeado que no es real, para acabar la configuración.

5º) Mapear dispositivo

Crearemos el sitema de ficheros:

```
% mkfs /dev/mapper/test
```

Tras mostrar bastan informació, nos informa que el sistema se revisara automáticamente a los 34 montajes o 180 días.

Bueno ya tenemos nuestra USB mapeada en la carpeta *test* ahora nos queda montar el dispositivo para acceder a el.

6º) Montar el dispositivo

Para montar el USB, ya no iremos a su dirección física real, */dev/sdf1*, sino a la unidad mapeada anteriormente, */dev/mapper/test*, para lo cual escribimos en el terminal:

```
% mount /dev/mapper/test /mnt/prueba
```

Nota: Tiene que existir la carpeta "prueba" dentro de mnt y con permisos 777

7º) Comprobación

Comprobamos que la unidad se ha montado correctamente y copiamos algunos archivos dentro de ella (necesitamos permisos de root).

8º) Cerrar el dispositivo

Para retirar el USB del pc, debemos escribir en el terminal:

```
% cryptsetup luksClose /dev/mapper/test

Device /dev/mapper/test is busy.
```

Ahora, cuando quitemos y pinchemos el USB de nuevo, nos pedirá la clave para montar el dispositivo.

Para un dispositivo fijo, si deseamos montar automáticamente la unidad cifrada podemos modificar el */etc/fstab* para realizar el montaje automáticamente:

```
/dev/mapper/test /mnt/prueba xfs defaults 0 0
```

Ejercicio 1.- Utilizar `cryptsetup` para crear una partición encriptada en un pedrive. Escribir un archivo en él. Desmontarlo y extraerlo. ¿Qué ocurre cuando volvemos a conectarlo?

1.1.- Otros sistema de archivos encriptados

En el artículo n.º 25 de Linux Magazin "Plataformas de cifrado en Linux. Ficheros 100% seguros" disponible en http://www.libelium.com/uploads/2012/12/linux-magazine-encryption.pdf, podéis encontrar las descripción de varios sistemas de archivos encriptados en Linux, como utilizarlos y sus pros y contras.

En sistemas Windows desde la versión 7, esta disponible la herramienta **`AppLocker`** (https://msdn.microsoft.com/es-es/library/ee424367(v=ws.11).aspx).

2.- Esteganografía

Adicionalmente, he incluido en esta práctica una técnica para ocultar información, mensajes u objetos, dentro de otros (los portadores) denominada *esteganográfia*. No debemos confundir esteganografía con la criptografía, si bien ambas tratan de proteger información y, por tanto, son complementarias (razón por la cual la he incluido), son distintas:

- La estaganografía oculta la información de modo que no sea advertido el hecho de su existencia o envío.
- La criptografía cifra la información para que no sea comprensible, si bien sabe que exite, al personal ajeno a la clave.

Las dos técnicas más comunes utilizadas en esteganografía son la *LSB* (*Least Significant Byte*) y la *inyección*. Con LSB, se aprovecha el hecho de que algunos bytes de un archivo tienen poca relevancia o son innecesarios, por lo que pueden ser reemplazados con bytes de un archivo que deseamos ocultar sin dañar o alterar al portador. Este técnica funciona bien cuando los archivos portadores son imágenes de alta resolución donde el aumento del archivo pasa desapercibido.

La técnica de inyección simplemente dispersa los bytes ocultos dentro del archivo original. También existen técnicas más sofisticadas, como la *esteganográfia entre blogs*, donde las piezas del mensaje secreto se dispersan de forma distribuida a lo largo de un grupo de blogs en forma de comentarios. La clave para descifrar el mensaje es la selección de los bogs utilizados. Otra técnica reciente utiliza redes VoiP para ocultar conversaciones. Algunos artículos interesantes son:

- *A Steganography Scheme in P2P Network* ("Un esquema esteganográfico en redes Peer-to-Peer", 2008), donde a través de un profundo análisis los investigadores encontraron que los

archivos torrent son ideales para esconder datos.

- *Hidden Communication in P2P Networks: Steganographic Handshake and Broadcast* ("Comunicación oculta en redes P2P: sincronización y difusión esteganográfica"), donde investigan cómo un grupo de nodos en una red P2P pueden hallarse y comunicarse sin provocar sospechas entre los censores o monitores de la red.
- *Graffiti Networks: A Subversive, Internet-Scale File Sharing Model* ("Redes Graffiti: Un modelo subversivo, a escala de Internet, de distribución de archivos", 2011), donde proponen un esquema de almacenamiento P2P, inspirado en técnicas esteganográficas, que protege a los usuarios de ser implicados en actividades criminales.
- Reversing the Steganography Myth in Terrorist Operations: The Asymmetrical Threat of Simple Intellig, es un artículo publicado por SANS que nos introduce en diferentes formas de ocultar información de forma sencilla y virtualmente indetectable e intraceable.

Entre las herramientas que podemos utilizar, encontramos:

- Steghide: Libre, portable, rápido, soporta cifrado y compresión. Es uno de los más usados, y está disponible en los repositorios de software de un montón de distribuciones. Trabaja con JPEG, BMP, WAV y AU.
- JPHide y JPSeek: Trabajan con JPEG, y prometen ser programas muy silenciosos, ya que la modificación del contenedor es mínima.
- Gifshuffle: Como su nombre indica, utiliza ficheros GIF (animados también). Una herramienta muy portable, libre, rápida, y que soporta cifrado / compresión.
- SNOW: Del mismo creador que Gifshuffle, utiliza espacios y tabulaciones al final de las líneas de un fichero de texto ASCII. También es altamente portable, libre, rápido y soporta cifrado / compresión.
- wbStego: Tiene varias versiones, y pretende tener una interfaz más amigable. Utiliza formatos BMP, texto ASCII o ANSI, HTML y PDF.
- MP3Stego: Utiliza ficheros MP3 como contenedores, admite compresión y cifrado.
- Stelin y StegoSense: Herramientas de esteganografía sintáctica en textos.
- Fuse::PDF: módulo de Perl que podemos usar para ocultar documentos personales dentro de ficheros PDF.

También, podemos utilizar un sistema de archivos esteganográfico como StegFS (Steganographic File System) (http://www.cl.cam.ac.uk/~mgk25/ih99-stegfs.pdf).

2.1 Esteganografía elemental

Veamos una forma elemental de realizar esteganografía que no requiere herramientas especiales y que podemos hacer con `cat` y `zip`. Supongamos que deseo ocultar un archivo denominado *agenda.txt* en un archivo denominado *foto.jpg*. Los pasos a seguir son:

1º Comprimimos el archivo el archivo a ocultar, por ejemplo, con `rar`. La razón de este paso es que podemos añadir una contraseña en la compresión.

2º Concatenamos el archivo imagen con el que deseamos ocultar: `cat foto.jpg agenda.rar > foto_modificada.jpg`.

3º Podemos borrar el archivo *agenta.txt* original.

Ahora podemos ver el archivo *foto_modificada.jpg* con un visor de imágenes y no observaremos nada.

Para recuperar el archivo original debemos:

- Renombrar *foto_modificada.jpg* a *foto_modificada.rar*.

- Abriremos *foto_modificada.rar* con el programa para descomprimir, y tendremos los dos archivos (también podríamos haber abierto *foto_modificada.jpg* directamente con el descompresor).

2.2 Esteganográfia con StegHide

Vamos a probar la técnica con el paquete **`Steghide`** que podemos descargar de http://lino.ubuntuupdates.org/package/core/raring/universe/base/steghide. Una vez descargado e instalado, vamos a utilizarlo suponiendo que tenemos un archivo denominado *ocultar.txt* de texto que deseamos ocultar en una imagen que reside en el archivo *imagen.jpg* (podríamos utilizar archivos .bmp, -wav, .au).

Ocultamos el archivo con la orden[1]:

```
$ steghide embed -cf imagen.jpg -ef texto.txt
```

Nos pedirá una palabra o frase de clave que habrá que confirmarla. Una vez finalizado este paso podemos borrar el archivo de texto original (que ahora esta comprimido y ocultado en la imagen).

Para extraer el archivo solo debemos ejecutar:

```
$ steghide extract -sf imagen.jpg
```

dando la clave previamente establecida.

Ejercicio 2.- Utilizar la herramienta *Steghide* para ocultar un mensaje dentro de una imagen, tal como acabamos de ver. Comparar los archivos portador antes y después de usar la técnica para ver las diferencias.

3.- Estegoanálisis

El *estegoanálisis* es el proceso de detección de mensajes ocultos mediante una esteganografía. El

1 Encontraremos un manual en español de Steghide en la dirección http://steghide.sourceforge.net/documentation/manpage_es.php.

estegoanálisis manual analiza de manera manual las diferencias entre la imagen original y la estaganográfica para ver las diferencias y localizar datos. El problema es que necesitamos una imagen original y que si bien podemos ver que hay un mensaje es casi imposible saber el contenido.

Otra alternativa es el *estegoanálisis estadístico* que consiste en el cotejo de la frecuencia de distribución de colores de una imagen esteganográfica. Dado que es una técnica lenta se debe emplear software especializado. Estos programas suelen buscar las pautas para ocultar los mensajes que utilizan los programas habituales de esteganografía, lo que los hace muy eficaces cuando se trata de mensajes ocultos con programas estándares. Los mensajes ocultados manualmente son casi imposibles de encontrar.

El uso de técnicas de incrustación de datos basadas en la sustitución LSB, introduce anomalías estadísticas en la imagen que pueden ser detectadas. La inserción de un mensaje sustituyendo el LSB no es una operación simétrica. Cuando se inserta un cero los valores pares siempre conservan su valor, mientras que los valores impares decrementan en uno. Cuando se inserta un uno los valores impares siempre conservan su valor, mientras que los valores pares incrementan en uno. Esto puede ser detectado mediante un histograma de la intensidad de los píxeles, es decir, un gráfico de barras donde cada barra represente la cantidad de píxeles de cada valor. Este histograma será muy diferente en una imagen sin alterar que en una imagen donde se ha escondido un mensaje, dado que la operación asimétrica comentada anteriormente hará que cada barra par siempre ceda píxeles a su barra posterior, y que cada barra impar siempre ceda píxeles a su barra anterior. En consecuencia, las barras tenderán a agruparse en parejas consecutivas que tenderán a tener la misma altura, lo que permitirá detectar la inserción.

Por este motivo las técnicas de sustitución LSB son remplazadas por lo que se conoce como *LSB Matching*. Esta técnica modifica el valor LSB sumando o restando uno, en lugar de sustituir el LSB. El resultado es el mismo (en cuanto a la modificación del LSB) pero esta operación tiene acarreo, lo que hace que la operación sea simétrica y en consecuencia invulnerable al ataque del histograma, comentado en el apartado anterior. Para detectar sistemas de LSB *matching* es necesario recurrir a técnicas más complejas, que creen modelos estadísticos de la relación entre un píxel y sus vecinos. De esta manera una imagen puede ser analizada para ver si el modelo estadístico que sigue es el de una imagen original o el de una imagen alterada por la inserción de un mensaje.

La única herramienta disponible para Ubuntu es VSL (*Visual Steganalysis Laboratory*[2]) esta escrita en java. En el archivo README.TXT que se descarga podemos ver como podemos ejecutarla con java.

No hay un manual para esta herramienta, pero su uso es sencillo. En el video-tutorial que podéis encontrar en https://www.youtube.com/watch?v=__NIlOIyxYI se explicar su manejo.

Ejercicio 3.- Utilizar VSL para analizar la imagen esteganográfica generada en el ejercicio anterior para detectar información oculta.

---

2 http://vsl.sourceforge.net/ o http://sourceforge.net/projects/vsl/?source=typ_redirect