SEGURIDAD EN SISTEMAS OPERATIVOS

4º Grado en Informática

Curso 2016-17

Práctica 3.- Auditoría informática e Informática forense

Sesión 1.- Análisis forense en Linux

Objetivos: En esta práctica, veremos los pasos a seguir para la recogida y análisis de información de un disco, tal como se hace en auditoría informática o el análisis forense: Primero, utilizando exclusivamente herramientas estándares de Linux. En segundo lugar, veremos una herramienta construida exclusivamente para realizar estas funciones.

1.- Elementos básico de auditoría informática y de informática forense

- Determinar la estructura de un disco

Dos herramientas simples nos permiten determinar la estructura de un disco conectado a nuestro sistema. La primera es fdisk que usaremos con la opción -l. Por ejemplo, si en nuestro sistema las particiones reciben el nombre de *sdXX*:

```
$ /usr/sbin/fdisk -l /dev/sd*
Disk /dev/sda: 320.1 GB, 320072933376 bytes
255 heads, 63 sectors/track, 38913 cylinders, 625142448 sectores en total
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Identificador del disco: 0x6025dfe9

Disposit. Inicio    Comienzo       Fin      Bloques  Id  Sistema
/dev/sda1              63     4209029     2104483+  82  Linux swap / Solaris
/dev/sda2   *     4209030    46154744    20972857+  83  Linux
/dev/sda3        46154745   625137344   289491300    f  W95 Ext'd (LBA)
/dev/sda5        46154808    50363774     2104483+  82  Linux swap / Solaris
/dev/sda6        50363838    92309489    20972826   83  Linux
/dev/sda7        92309553   625137344   266413896   83  Linux

...
```

Es conveniente salvar esta información en un archivo para un uso posterior, redirigiendo su salida a un archivo (el nombre es arbitrario):

```
$ /usr/sbin/fdisk -l /dev/sd* > fdisk.disco1
```

- Crear una imagen forense del disco

Es conveniente crear una imagen del disco en estudio para asegurarnos de no afectar al disco original. Si suponemos que deseamos copiar un disquete, podríamos hacerla de la siguiente forma:

```
$ dd if=/dev/fd0 of=/imagen.disco1  bs=512
```

Por cuestiones de seguridad y prácticas, deberíamos cambiar los permisos para ajustarlos a solo lectura:

```
$ chmod 444 imagen.disco1
```

Una vez creada la imagen del disco, podemos restaurarla en otro disco para su posterior análisis, es decir, utilizaremos la copia de seguriad como disco de trabajo. Para ello insertaríamos otro disquete y restauramos la imagen en él:

```
$ dd if=imagen.disco1 of=/dev/fd0 bs=512
```

Debemos observar que `dd` crea un duplicado exacto del dispositivo físico. Esto incluye todos los archivos sueltos y el espacio sin asignar, y no solo la estructura de archivos lógica del mismo. A diferencia de otras herramientas de imagen forenses, dd no rellena la imagen con ningún dato o información propietario, es simplemente una copia de flujo de bits desde el inicio hasta el fin.

- Montando la imagen restaurada

Una vez copiado el disco, podemos montar el disco clonado como copia de trabajo y poder ver su contenido.

```
$ mount -t vfat -ro,noexec /dev/fd0 /mnt/analisis
```

Observar, como hemos montado el disco de solo lectura (`-ro`) y de no ejecución (`-noexec`, para evitar la posible ejecución de binarios desde el punto de montaje), de cara a proteger tanto el disco de nosotros como nuestro sistema del disco. Otra opción útil de `mount` sería, por ejemplo, `noatime`.

Ahora podemos cambiar el directorio de trabajo al punto de montaje y navegar por los contenidos del mismo.

Otra forma de ver los contenidos del disco imagen sin tener que restaurarlo en otro disco es montándolo con la interfaz loop:

```
$ mount -t vfat -o,noexec,loop imagen.disco1 /mnt/analisis
```

No olvidemos desmontar el disco al finalizar.

- Hash de archivos

Un paso importante en cualquier análisis es verificar la integridad de nuestro datos antes y después de que se complete el análisis. Podemos obtener un hash (CRC, MD5 o SHA) de cada archivo de diferentes formas. Nosotros utilizaremos un hash SHA. SHA es un generador de firmas que suministra una huella de 160 bits de un archivo o disco. No es factible para alguien computacionalmente recrear un archivo basado en su hash SHA, lo que significa que igualar la signatura SHA es tener el mismo archivo.

Obtenemos la suma SHA[1] del disco cambiándonos al directorio donde tenemos la imagen del

1 En los ejemplos, hemos utilizado sha1sum por conveniencia pero debemos tener en cuenta que actualmente se desaconseja su uso dado que es "teóricamente" atacable (http://securityaffairs.co/wordpress/40884/hacking/sha-1-

disco y ejecutando

```
$ sha1sum /dev/fd0
```

o

```
$ sha1sum /dev/df0 > SHA.disco1
```

En este último caso, hemos guardado el valor hash en un archivo para posteriores comprobaciones. Para obtener el hash de un disco raw, el disco no debe de estar montado, ya que queremos obtener el hash del disco no del sistema de archivos.

Podemos obtener el hash de cada archivo independientemente utilizando la orden `find` y una opción para ejecutar la orden por cada archivo encontrado. Podemos obtener una lista muy útil de hash SHA para cada archivo del disco, ahora si previamente montado.

```
$ mount -t vfat -o,noexec,loop imagen.disco1 /mnt/analisis
$ cd /mnt/analisis
$ find . -type f -exec sha1sum {}  \;
>/root/evidencias/SHA.listaArchivos
```

Esta última orden, localiza cualquier archivo regular desde el directorio actual y ejecuta para el mismo la orden `sha1sum`, redirigiendo la salida al archivo *SHA.listaArchivos*. La secuencia de escape "`\;`" finaliza la orden `sha1sum`.

Podemos hacer que Linux realice la verificación por nosotros. Para verificar que nada a cambiado en el disquete origina, podemos utilizar la opción -c de sha1sum. Si el disco no ha cambiado, nos devolverá un "ok". Si tenemos el disquete insertado, podemos ejecutar:

```
$ sha1sum -c /root/evidencias/SHA.disco1
```

Si el hash SHA iguala al disquete y al archivo de salida de SHA, la orden retornará "ok" para */dev/fd0*. Podemos hacer lo mismo con la lista de SHA de los archivos. Montamos el disquete en /mnt/analisis, cambiamos a este directorio y ejecutamos:

```
$ sha1sum -c /root/evidencias/SHA.listaArchivos
```

- El análisis

Ya podemos ver el contenido del disco montado, bien la copia restaurada o la imagen montado con loop. Para ello, podemos utilizar nuestro navegador de archivos favorito, o utilizar la línea de órdenes. Esta última tiene la ventaja de que nos permite, mediante redirecciones, hacer copia permanente de nuestro análisis. En lo que sigue, optaremos por la línea de órdenes.

Lo primero será hacer una lista de todos los archivos, por ejemplo

```
$ ls -laiRtu > /root/evidencias/lista.archivos
```

o bien

```
$ find . -type f -print >/root/evidencias/lista.archivos.2
```

También podemos utilizar la orden grep sobre alguna de las lista anteriores para buscar cualquier cadena o extensión que deseemos. Por ejemplo, si estamos buscando imágenes, podemos ejecutar

```
$  grep -i jpg lista.archivos
```

---

collision-attack.html). Podéis usar alguna de las otras órdenes de vuestro sistema: `/usr/bin/sha256sum`, ...

Podemos hacer una lista de archivos por su tipo con la orden file. Por ejemplo:

```
$ find . -type f -exec file {} \; >/root/evidencias/lista.tipos.archivos
```

De forma que podemos buscar las imágenes de la forma:

```
$ cat /root/evidencias/lista.tipos.archivos | grep image
```

En el caso de archivos de texto o datos, podemos ver su contenido con las órdenes `cat`, `more`, o `less`. En muchos casos una mejor alternativa es utilizar la orden `strings` para ello.

- Búsqueda de texto en espacio disperso o no asignado

Hasta ahora hemos estudiado los archivos/directorios del disco, pero que pasa con el espacio disperso o no asignado. Supongamos que el disco en estudio pertenece a un empleado descontento del que sospechamos que utilizado el disco para infectar la red de la empresa con un programa o correo y que posteriormente a borrado el archivo. Necesitamos conocer si el disco contiene aún parte de la información que almacenaba el archivo, para lo cual necesitamos analizar los bloque des disco no en uso que pueden aún contener información útil para nuestro propósito.

Para ello podemos utilizar la orden grep. Lo primero que debemos hacer es crear una lista de palabras clave a utilizar en la búsqueda. Por ejemplo, decidimos buscar las palabras "rescate", "€5000", y "virus", que son las palabras más significativas que aparecían en el correo que se envió a la empresa.

Creamos la lista con editor en el archivo */root/evidencias/listaBusqueda.txt* asegurándonos que cada palabra o frase aparece en una línea y que no hay líneas en blanco en el archivo. Tras lo cual podemos realizar la búsqueda:

```
$ grep -aibf listaBusqueda.txt imagen.disco1 >aciertos.txt
```

Si se producen coincidencias, el archivo *aciertos.txt* contendrá líneas donde se ha producido así como el desplazamiento de bytes de las mismas (opción `-b` de `grep`). Por ejemplo:

```
  $ cat aciertos.txt
  75441: la empresa debe para el rescate
  75500: no contacte con la policia si no quiere que un virus infecte la
red.
```

Podemos utilizar esta información para visualizar con xxd los archivos. Para cada desplazamiento, ejecutaremos:

```
$ xxd -s offset imagen.disco1 | less
```

O bien podemos utilizar un editor hexadecimal con interfaz gráfica, como por ejemplo `Ghex`.

- Algunas cuestiones adicionales

Los ejemplos vistos con anterioridad hacían referencia a un disquete, pero ¿qué ocurre si deseamos manipular un disco duro grande? Al copiar el disco con dd, esta orden copia todos los componentes, lo que suele incluir un sector de arranque, una tabla de particiones y las propias particiones. Si este es el caso, la orden `mount` es incapaz de encontrar el sistema de archivos ya que ésta no reconoce la tabla de particiones. La forma más fácil de soslayar esto (si bien, no la más eficiente en grandes discos) es crear imágenes separadas para cada una de las particiones que

deseamos analizar.

Para un disco simple con una única partición, podemos crear dos imágenes:

```
$ dd if=/dev/sda of=imagen.disco bs=4096   #disco entero
$ dd if=/dev/sda1 of=imagen.disco bs=4096  #primera partición
```

La primera orden contiene una copia de seguridad del disco completo, incluyendo en sector de arranque y la tabla de particiones. La segunda orden obtiene copia de la partición que podremos montar vía el dispositivo loop. Si bien ambas copias contienen el mismo sistema de archivos, sus sha1sum's no coincidirán.

Otra forma de manejar grandes discos (montando la imagen con el dispositivo loop) es indicar a la orden `mount` que salte los primeros 63 sectores de la imagen. Estos sectores contienen entre otra información el MBR, que no es parte de los datos de la partición. Como cada sector es de 512 bytes y hay 63, esto nos da un desplazamiento de 32256 bytes desde el inicio de nuestra imagen de la primera partición. Por lo que ejecutaremos:

```
$ mount -t vfat -ro,noexec,loop,offset=32256 imagen.disco /mnt/analisis
```

Si la orden `dd` nos da algún error debido al tamaño de los archivos podemos leer saltando el error con la opción `conv=error`, que indica a `dd` que ignore el error. Con esta opción, es buena idea incluir la opción `sync`, para la salida de `dd` se rellene donde se encontró el error y asegurarnos que la salida esta sincronizada con el original.

```
$ dd if=/dev/sdax of=imagen.disco conv=noerror,sync
```

Otro problema que podemos encontrar proviene de grep, que podría darnos un error de memoria exhausta si lee más de 200MB si encontrar un carácter "newline". Podemos salvar el problema con la orden:

```
$ tr '[:cntrl:]' '\n' <imagen.disco1 |grep -aibf listaBusqueda.txt > aciertos.txt
```

La orden `tr` lo que hace es traducir todos los caracteres del juego de control ([:cntrl:]) en *newlines* (\n).

Una de las labores de la práctica forense del análisis de discos es "limpiarlos" antes de restaurar una imagen forense del mismo. Esto nos asegura que cualquier dato encontrado en el disco restaurado proviene de la imagen y no de datos residuales del disco donde hacemos la copia. La idea es simple, creamos un disco vacío relleno de ceros.

```
$ dd if=/dev/zero of=/dev/sdax bs=4096
```

Si deseamos comprobar que efectivamente el disco esta "limpio" podemos analizar sectores aleatorios del mismo con el editor hexadecimal para ver que efectivamente nos dice que están a cero.

Ejercicio 1.- Vamos a crear en nuestro *pendrive* un archivo con un supuesto texto de una amenaza y luego vamos a borrarlo. Aplicando las herramientas anteriores vamos a intentar recuperar lo que quede del archivo borrado haciendo una copia del *pendrive* sobre la que trabajar, no directamente sobre el *pendrive*.

2.- Creación de una imagen forense

De entre las diferentes herramientas existentes para la creación de contenedores forenses, vamos a usar la herramienta `guymager` (http://guymager.sourceforge.net/) que se puede instalar desde el repositorio de Ubuntu.

Cuando iniciamos la herramienta, esta nos muestra los dispositivos montados en el sistema, como aparece en la Figura 1.

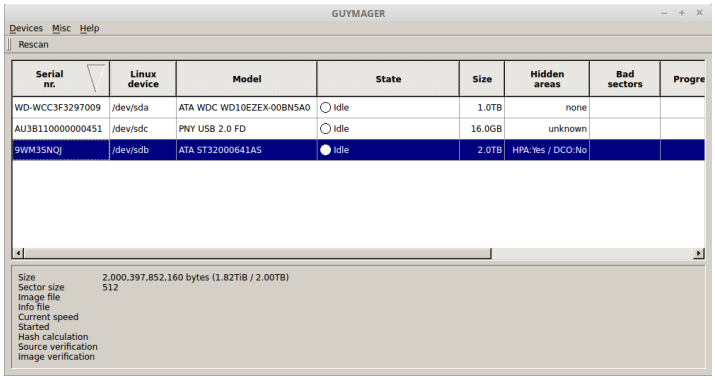

Figura 1.- Ventana principal de `guymager`.

Seleccionamos el dispositivo del que deseamos generar la imagen con el botón derecho del ratón y se nos despliega una ventana como la Figura 2.

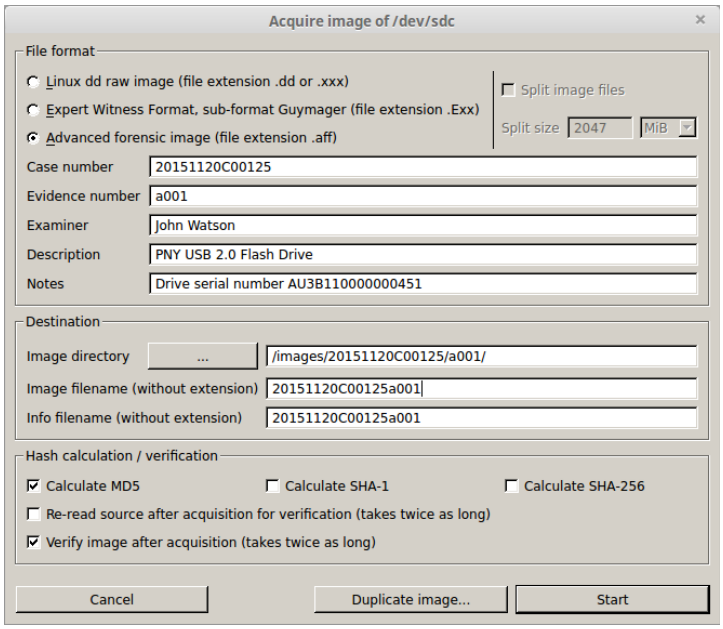

Figura 2.- Ventana de configuración de la imagen.

Por defecto tenemos seleccionado el formato .dd, pero podemos seleccionar los formatos .EWF (.E01) o .AFF[2]. Además nos solicita la siguiente información:

- Número de caso

2 En versiones reciente de la herramienta, el formato AFF ha sido relegado, pero se puede incluir si el archivo de configuración de la herramienta */etc/guymager/local.cfg* incluimos `AffEnabled=true`.

- Número de prueba
- Nombre del examinador
- Descripción del dispositivo del que vamos a realizar la imagen
- Notas del examinador (la herramienta inserta automáticamente el número de serie del dispositivo., si esta disponible)
- Ubicación del archivo imagen

Toda esta información ira en los metadatos de la imagen. Además, la herramienta crea un hash (existen varias posibilidades) del dispositivo del que hemos generado la imagen y de la imagen para mostrar la verificación de la misma.

En un caso real, etiquetaríamos el dispositivo original y su bolsa del almacenamiento antiestática con los mismos metadatos suministrados a la herramienta. Este etiquetado consistente es necesario para documentar las evidencias y mantener la cadena de custodia. Una vez etiquetados el dispositivo y su contenedor pueden entrar en custodia tras realizar la imagen.

Una vez completados todos los campos, podemos Iniciar la adquisición de la imagen. Durante el proceso aparece una ventana similar a la Figura 3. En ella, se puede ver el archivo destino de la imagen (.aff o .awf) y los metadatos (.info). Cuando acabe el proceso, se mostrará "Acabado" en en la entrada correspondiente al dispositivo concreto de la ventana de la Figura 1 de la herramienta. Ya se puede cerrar la herramienta.

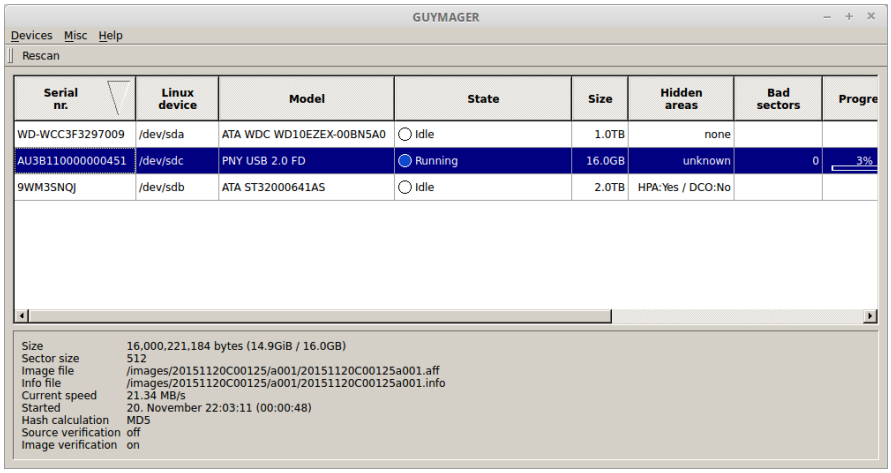

Figura 3.- Ventana del proceso de adquisición de la imagen.

Podemos ver los metadatos, mostrando el contenido del archivo con extensión .info. La igualación de los hash verifica que la integridad forense de la imagen. La herramienta genera un hash del volumen, por ejemplo, del dispositivo */dev/sdc*).

De cara a mantener la cadena de custodia, necesitamos crear un hash del archivo imagen. Este hash no será el mismo que el hash del volumen salvado por la herramienta en los metadatos del archivo .info. Este hash verifica la integridad del proceso de toma de la imagen, pero ahora que tenemos la imagen, podremos hacer más copias de ella. Podemos verificar la integridad del archivo imagen y de las copias de trabajo.

Ejercicio 2.- Realizar una imagen forense del *pendrive* con la herramienta `guymager`.

3.- The Sleuth Kit y Autopsy

En este apartado, vamos a manejar una herramienta open source multiplataforma para el análisis forense de uso común *The Sleuth Kit* (TSK) y su interfaz gráfica *Autopsy*.

TSK es una librería y un conjunto de herramientas entre las que podemos encontrar:

- `ils` – lista los entradas de metadatos, tales como inodos.
- `blkls` – muestra los bloques de datos de un sistema de archivos.
- `fls` – muestra los nombres de archivos asignados y no asignados de un sistema de archivos.
- `fsstat` – muestra información estadística de un sistema de archivos sobre una imagen o medio de almacenaje.
- `ffind`- busca nombres de archivos que apuntan a una entrada de metadatos específica.
- `machine` – crea una línea de tiempo para todos los archivos basada en sus tiempos de última modificación (MAC times).
- `disk_stat` – descubre la existencia de un Área Protegida del Anfitrión (área del HD que no es visible al SO).

En cuanto a Autopsy, contiene funcionalidad para:

- *Análisis de líneas temporales* – visor gráfico avanzado de eventos (tutorial en http://www.sleuthkit.org/autopsy/timeline.php)
- *Filtrado hash* – que permite marcar archivo malos e ignorar los buenos.
- *Búsqueda por palabras clave* – búsqueda indexada para encontrar archivos que mencionan términos relevantes (direcciones de correo, números de teléfono, IPs, URLs, etc.).
- *Artefactos web* – permite extraer el historial, bookmarks y cookies de navegadores web.
- *Recuperación de datos* – recuperar archivos borrados de espacio sin asignar de disco usando PhotoRec (http://www.cgsecurity.org/wiki/PhotoRec).
- *Multimedia* – extraer información EXIF (metadatos) de imágenes y videos.
- *Indicadores de compromiso* – escanea el computador usando STIX(lenguaje estructurado para inteligencia de ciber-amenzas, ver http://stixproject.github.io/).

La herramienta esta diseñada con los siguientes principios:

a) Extensible: el usuario puede añadir nueva funcionalidad creando pluging que puedan analizar todos o parte de las fuentes de datos.

b) Frameworks: ofrece enfoques estándares para la toma y análisis de datos, así como para la generación de informes, de forma que cada desarrollador pueda seguir el mismo patrón de diseño.

c) Fácil de usar: el navegador *Autopsy* ofrece herramientas históricas y de guía de forma que el usuario pueda repetir sus pasos sin excesivas reconfiguraciones.

Para usar la herramienta bien podemos hacerlo desde una distribución de Kali, bien podemos instalarla en nuestra distribución de Linux (o Windows, en este caso podemos descargar la última versión de http://www.sleuthkit.org/autopsy/download.php).

En esta práctica, vamos a optar por instalarlo en una distribución de Linux. Los pasos serían:

1. Descargamos TSK de http://www.sleuthkit.org/sleuthkit/download.php y Autopsy de http://www.sleuthkit.org/autopsy/v2/download.php.
2. Primero, procedemos con *TSK*, para el cual:
   a) Debemos extraer los archivos

b) **Configurarlo, ejecutando** `./configure`

c) **Compilarlo con** `./make` **primero y luego** `./make_istall`

3. **Continuamos con *Autopsy*:**

a) **Extraemos los archivos:**

b) **Los configuramos** `./configure`**, que nos pedirá identificar el nombre del directorio de evidencias, que previamente habremos creado.**

c) **Ya podemos ejecutarlo:** `./autopsy`**. El proceso nos muestra la dirección que debemos usar en el navegador para acceder a la herramienta.**

d) **Abrir en navegador y suministrar la dirección *http://localhost:9999/autopsy*.**

Cuando se abre Autopsy, la primer tarea que se debe llevar a cabo es o crear un nuevo caso o abrir uno existente, donde un caso es la unidad lógica que contendrá todo lo relacionado a la investigación. Por lo tanto, al crear un caso se ingresa información como su nombre y número y la persona que examinará los datos. El siguiente paso consiste en asociar al caso uno o más orígenes de datos, entre los cuales se encuentran los discos físicos conectados a la computadora de análisis, o una imagen forense que se ha adquirido previamente de la computadora a ser investigada.

Una vez creado el caso, el siguiente paso es agregar una imagen, esta será una réplica del disco (disco duro, pendrive, etc.) que deseamos analizar. En nuestro caso, podemos usar el pendrive utilizado en el apartado anterior, o el propio disco duro del sistema o una partición.

Establecida la imagen a analizar, esta se explora para recolectar evidencias para el análisis. En esta fase la herramienta clasifica todas la evidencias por categorías: documentos por tipo, correos, historial del navegador, etc. incluyendo archivos borrados. Los archivos que no son permanentes o que han sido borrados, se marcan con una cruz roja. Teniendo en cuenta que la cantidad de archivos encontrados en la imagen puede ser demasiadamente extensa, Autopsy cuenta con una barra de menús, que tienen la opción de buscar archivos o evidencias, por palabras claves, iniciales, tipo de archivos, metadatos, sectores específicos del disco y otras series de opciones, que permiten optimizar al máximo la búsqueda de evidencia.

Para las evidencias encontradas, podemos consultar todos los metadatos de las mismas. Además, la herramienta nos permite generar informes para cada evidencia encontrada sobre el estado, atributos, características y contenido de la misma. El informe permite visualizar los hash MD5 y SHA1 con el fin de comprobar que la evidencia examinada dese una copia no ha sido

modificada o alterada con respecto a la evidencia original.

Si bien la herramienta es bastante intuitiva, podemos ver documentación sobre su manejo en:

(1) Blog de Highsec, "Análisis Forense -Parte 3 – Autopsy: como analizar un disco duro" en http://highsec.es/2013/09/analisis-forense-parte-iii-autopsy-como-analizar-un-disco-duro/

(2) A. E. Caballero Quezada "Autopsy" versión 1.6 de 2015. Disponible en http://www.reydes.com/archivos/Autopsy3_ReYDeS.pdf.

(3) José Luis Rivas López, "Introducción al análisis forense", editado por la UOC y disponible en http://jlrivas.webs.uvigo.es/downloads/publicaciones/Analisis%20forense%20de%20sistema%20informaticos.pdf, que nos dará una introducción a la materia y herramienta.

Ejercicio 3.- Como en el ejercicio 1 y partiendo de la imagen forense del 2, buscar con la herramienta Autopsy las evidencias de la amenaza realizada.
