

# SEGURIDAD EN SISTEMAS OPERATIVOS

4º Grado en Informática – Complementos de Ing. del Software  
Curso 2019-20

---

**Práctica [3].** Auditoría informática e informática forense.

**Sesión [2].** Análisis forense en Linux (II).

**Autor<sup>1</sup>:** Nazaret Román Guerrero

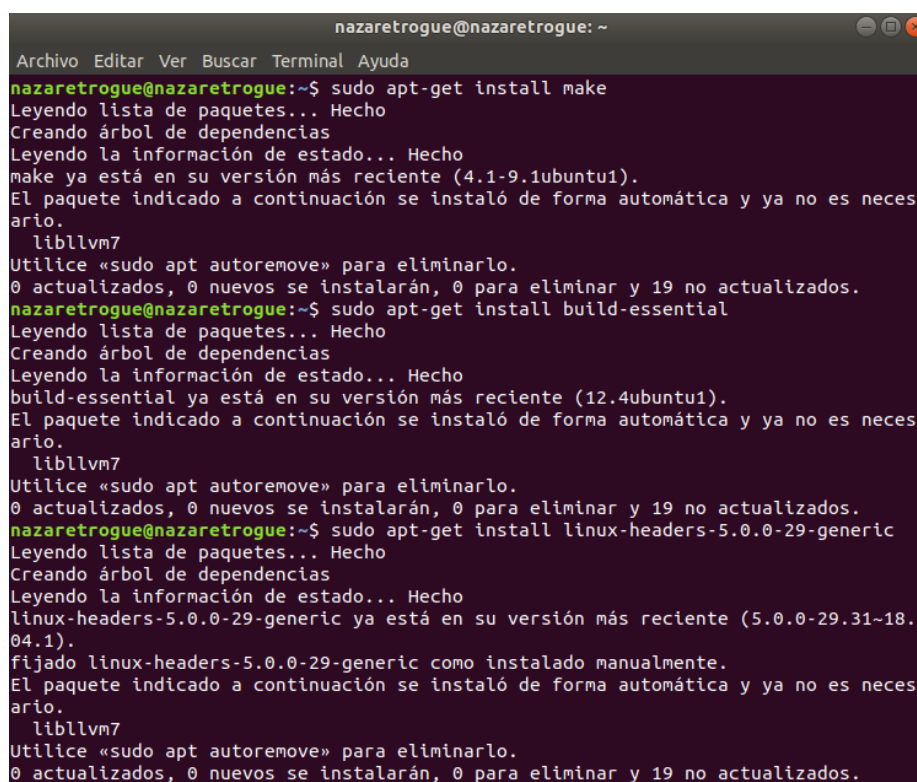
---

## Ejercicio 1.

---

Crear un volcado de memoria en formato .lime de la máquina que estéis utilizando.

Lo primero que debemos hacer es instalar los módulos necesarios en nuestra máquina. Debemos instalar make, build-essentials y linux-headers (para el último es necesario saber qué versión del kernel tiene la máquina por lo que debemos utilizar antes la opción `uname -r` para saberlo). La salida de ejecutar dichos comandos es:



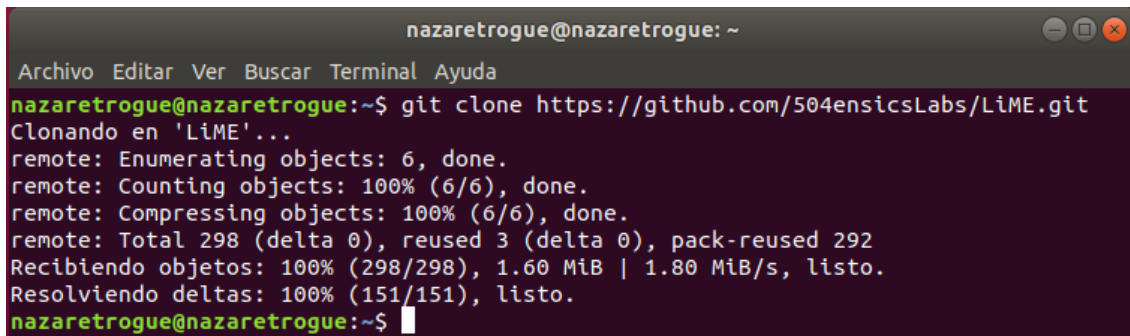
```
nazaretrogue@nazaretrogue: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
nazaretrogue@nazaretrogue:~$ sudo apt-get install make  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
make ya está en su versión más reciente (4.1-9.1ubuntu1).  
El paquete indicado a continuación se instaló de forma automática y ya no es neces  
ario.  
liblvm7  
Utilice «sudo apt autoremove» para eliminarlo.  
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 19 no actualizados.  
nazaretrogue@nazaretrogue:~$ sudo apt-get install build-essential  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
build-essential ya está en su versión más reciente (12.4ubuntu1).  
El paquete indicado a continuación se instaló de forma automática y ya no es neces  
ario.  
liblvm7  
Utilice «sudo apt autoremove» para eliminarlo.  
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 19 no actualizados.  
nazaretrogue@nazaretrogue:~$ sudo apt-get install linux-headers-5.0.0-29-generic  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
linux-headers-5.0.0-29-generic ya está en su versión más reciente (5.0.0-29.31~18.  
04.1).  
fijado linux-headers-5.0.0-29-generic como instalado manualmente.  
El paquete indicado a continuación se instaló de forma automática y ya no es neces  
ario.  
liblvm7  
Utilice «sudo apt autoremove» para eliminarlo.  
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 19 no actualizados.
```

Como podemos ver, todos estaban instalados y en su última versión. Por tanto, procedemos

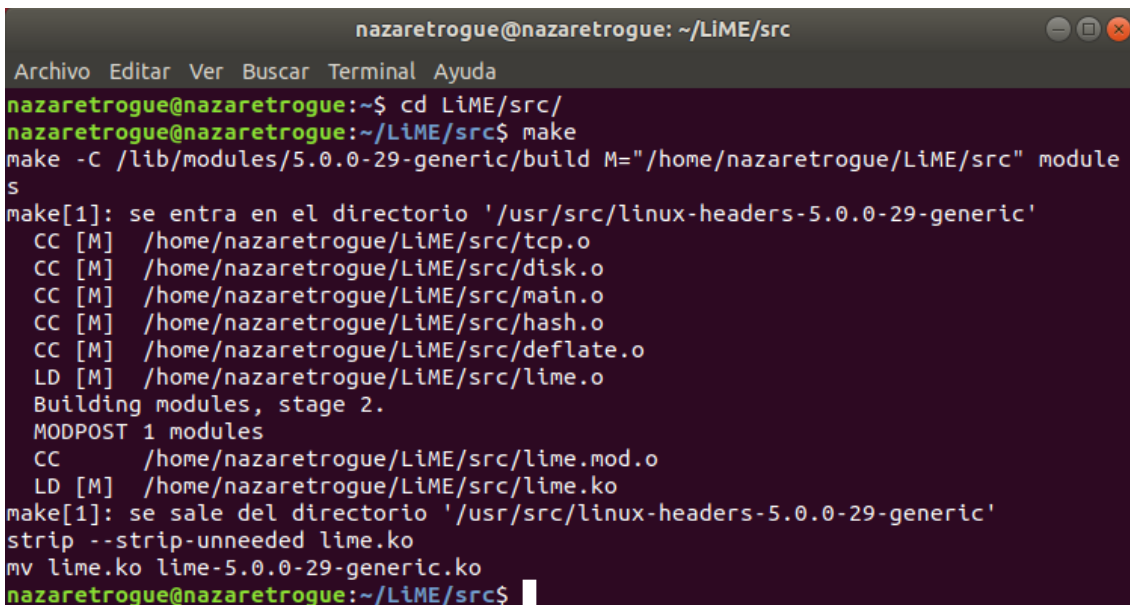
---

1 Como autor declaro que los contenidos del presente documento son originales y elaborados por mi. De no cumplir con este compromiso, soy consciente de que, de acuerdo con la “Normativa de evaluación y de calificaciones de los estudiantes de la Universidad de Granada” esto “conllevará la calificación numérica de cero ... independientemente del resto de calificaciones que el estudiante hubiera obtenido ...”

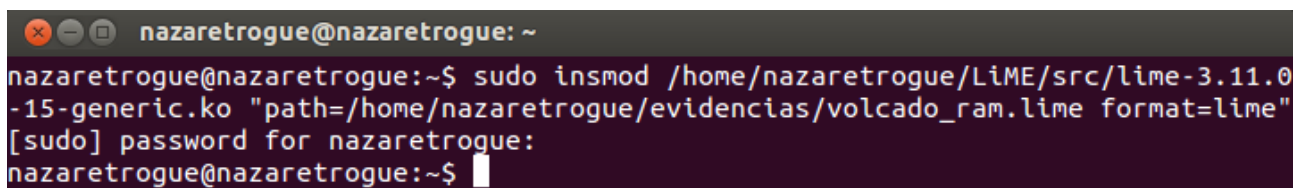
a instalar la herramienta LiME. Para poder clonar el repo necesitamos tener git instalado. Tras instalarlo, clonamos el repo:

A terminal window titled 'nazaretroque@nazaretroque: ~' with a menu bar (Archivo, Editar, Ver, Buscar, Terminal, Ayuda). The command 'git clone https://github.com/504ensicsLabs/LiME.git' is executed. The output shows the cloning progress: 'Clonando en 'LiME'...', 'remote: Enumerating objects: 6, done.', 'remote: Counting objects: 100% (6/6), done.', 'remote: Compressing objects: 100% (6/6), done.', 'remote: Total 298 (delta 0), reused 3 (delta 0), pack-reused 292', 'Recibiendo objetos: 100% (298/298), 1.60 MiB | 1.80 MiB/s, listo.', and 'Resolviendo deltas: 100% (151/151), listo.' The prompt returns to 'nazaretroque@nazaretroque:~\$'.

Una vez clonado, entramos en el directorio de los fuentes y compilamos el programa con make, tal y como se ve en la imagen:

A terminal window titled 'nazaretroque@nazaretroque: ~/LiME/src' with a menu bar (Archivo, Editar, Ver, Buscar, Terminal, Ayuda). The command 'cd LiME/src/' is executed, followed by 'make'. The output shows the compilation process: 'make -C /lib/modules/5.0.0-29-generic/build M="/home/nazaretroque/LiME/src" module s', 'make[1]: se entra en el directorio '/usr/src/linux-headers-5.0.0-29-generic'', a list of compilation commands for various object files (tcp.o, disk.o, main.o, hash.o, deflate.o, lime.o), 'Building modules, stage 2.', 'MODPOST 1 modules', a list of linking commands for lime.mod.o and lime.ko, 'make[1]: se sale del directorio '/usr/src/linux-headers-5.0.0-29-generic'', 'strip --strip-unneeded lime.ko', and 'mv lime.ko lime-5.0.0-29-generic.ko'. The prompt returns to 'nazaretroque@nazaretroque:~/LiME/src\$'.

Una vez compilado el programa, para hacer el volcado de memoria en formato lime, solo tenemos que ejecutar el comando `sudo insmod lime-5.0.0.29-generic.ko "path=/home/nazaretroque/evidencias/volcado_ram.lime format=lime"` que nos generará el volcado, como podemos ver en la imagen:

A terminal window titled 'nazaretroque@nazaretroque: ~' with a menu bar (x, -, □, nazaretroque@nazaretroque: ~). The command 'sudo insmod /home/nazaretroque/LiME/src/lime-3.11.0-15-generic.ko "path=/home/nazaretroque/evidencias/volcado\_ram.lime format=lime"' is executed. The output shows '[sudo] password for nazaretroque:' followed by the prompt returning to 'nazaretroque@nazaretroque:~\$'.

Es una imagen de una RAM de Ubuntu 12.04 con el kernel 3.2.1. Tras un total de 5 horas ejecutándose el volcado sobre una memoria de 512MB, obtenemos el archivo que podremos analizar en el siguiente ejercicio.

## Ejercicio 2.

Instalar volatility para analizar la imagen de la RAM obtenida en el ejercicio 1. Obtener con los plugins correspondientes información sobre:

- a) las conexiones de red activas,
- b) los procesos en ejecución,
- c) las bibliotecas que está usando uno de los programas listados, y
- d) las órdenes ejecutadas desde una consola.

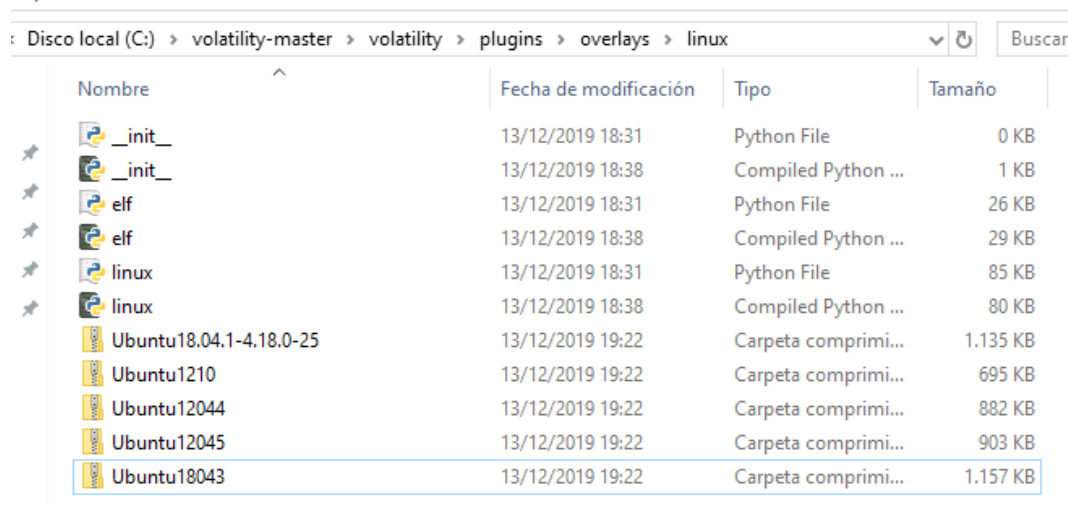
Para poder hacer un análisis forense primero debemos instalar la herramienta. Para ello primero es necesario tener instaladas algunas dependencias, como python, que como podemos ver en la imagen ya estaban instaladas y actualizadas:

Para este ejercicio, voy a utilizar Windows, puesto que es el sistema que más se ha acercado a funcionar. He probado en Ubuntu 18.04 con dos kernels: 5.0.0.29 y 3.2.1; también he probado Ubuntu 12.04 con tres kernels: 3.11.0, 3.5.0 y 3.2.1. He probado en Kali Linux 2019.4, tampoco ha funcionado.

Utilizando Windows 10 he conseguido acercarme un poco más al funcionamiento del análisis, pero tampoco lo he conseguido.

Voy a explicar los pasos que he seguido en Windows.

Lo primero que debemos hacer es instalar Python2.7 con todas sus dependencias. Una vez hecho esto, debemos añadir los perfiles de Linux que es de donde se ha extraído el volcado con LiME:

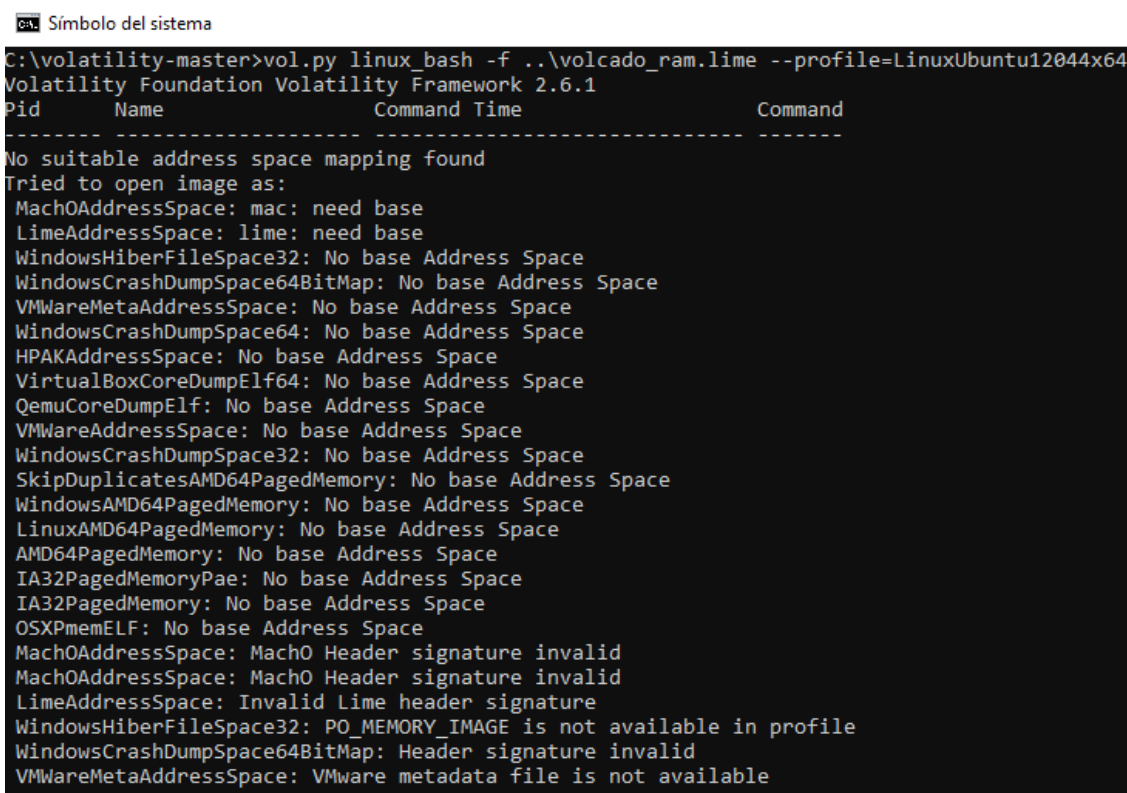


Nombre	Fecha de modificación	Tipo	Tamaño
__init__	13/12/2019 18:31	Python File	0 KB
__init__	13/12/2019 18:38	Compiled Python ...	1 KB
elf	13/12/2019 18:31	Python File	26 KB
elf	13/12/2019 18:38	Compiled Python ...	29 KB
linux	13/12/2019 18:31	Python File	85 KB
linux	13/12/2019 18:38	Compiled Python ...	80 KB
Ubuntu18.04.1-4.18.0-25	13/12/2019 19:22	Carpeta comprimi...	1.135 KB
Ubuntu1210	13/12/2019 19:22	Carpeta comprimi...	695 KB
Ubuntu12044	13/12/2019 19:22	Carpeta comprimi...	882 KB
Ubuntu12045	13/12/2019 19:22	Carpeta comprimi...	903 KB
Ubuntu18043	13/12/2019 19:22	Carpeta comprimi...	1.157 KB

Concretamente yo he añadido los perfiles para los sistemas que he utilizado, y, aunque en la captura siguiente voy a utilizar solo uno de ellos, los he probado todos y con todos da el mismo resultado: no se encuentra un espacio de direcciones que coincida con el del volcado.

- Para listar las conexiones de red activas se utiliza el plugin `linux_netstat`, que muestra los sockets activos en el momento del volcado.
- Para listar los procesos en ejecución se utiliza el plugin `linux_pslist`.
- Para listar las bibliotecas que está utilizando uno de los procesos se utiliza `linux_library_list`.
- Por último, para listar los comandos de la sesión activa de bash que hay en ese momento se utiliza el plugin `linux_bash`.

Todos los comandos dan un resultado similar: primero aparece la cabecera de la tabla que ha extraído durante el volcado, e inmediatamente después indica que no ha podido encontrar un perfil que coincida con el espacio de direcciones. En este documento solo incluyo una imagen con el último comando, el que lista los comandos de la sesión de bash; no obstante, los demás han sido también probados y con el mismo resultado, por lo que incluirlos en esta memoria me parecía inútil puesto que no muestran información diferente a esta:



```

C:\volatility-master>vol.py linux_bash -f ..\volcado_ram.lime --profile=LinuxUbuntu12044x64
Volatility Foundation Volatility Framework 2.6.1
Pid      Name      Command Time      Command
-----
No suitable address space mapping found
Tried to open image as:
MachOAddressSpace: mac: need base
LimeAddressSpace: lime: need base
WindowsHiberFileSpace32: No base Address Space
WindowsCrashDumpSpace64BitMap: No base Address Space
VMWareMetaAddressSpace: No base Address Space
WindowsCrashDumpSpace64: No base Address Space
HPAKAddressSpace: No base Address Space
VirtualBoxCoreDumpElf64: No base Address Space
QemuCoreDumpElf: No base Address Space
VMWareAddressSpace: No base Address Space
WindowsCrashDumpSpace32: No base Address Space
SkipDuplicatesAMD64PagedMemory: No base Address Space
WindowsAMD64PagedMemory: No base Address Space
LinuxAMD64PagedMemory: No base Address Space
AMD64PagedMemory: No base Address Space
IA32PagedMemoryPae: No base Address Space
IA32PagedMemory: No base Address Space
OSXPmemELF: No base Address Space
MachOAddressSpace: MachO Header signature invalid
MachOAddressSpace: MachO Header signature invalid
LimeAddressSpace: Invalid Lime header signature
WindowsHiberFileSpace32: PO_MEMORY_IMAGE is not available in profile
WindowsCrashDumpSpace64BitMap: Header signature invalid
VMWareMetaAddressSpace: VMWare metadata file is not available

```

Como se puede observar, al inicio parece que va a mostrar la lista de comandos con el PID, el momento que se ejecutó... Pero justo después el programa revienta y no es capaz de leer más del volcado.

A pesar de todo, también he decidido incluir el trabajo que he hecho en Ubuntu 18.04, para mostrar que tampoco lo he conseguido.

Lo primero es instalar las dependencias de Python.

```
nazaretroque@nazaretroque: ~
Archivo Editar Ver Buscar Terminal Ayuda
nazaretroque@nazaretroque:~$ sudo apt-get install python python-crypto
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
python ya está en su versión más reciente (2.7.15~rc1-1).
python-crypto ya está en su versión más reciente (2.6.1-8ubuntu2).
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
 liblvm7 linux-headers-4.18.0-15 linux-headers-4.18.0-15-generic
 linux-image-4.18.0-15-generic linux-modules-4.18.0-15-generic
 linux-modules-extra-4.18.0-15-generic
Utilice «sudo apt autoremove» para eliminarlos.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 24 no actualizados.
nazaretroque@nazaretroque:~$
```

Tras esto, debemos descargar el programa desde google. Está en formato .tar.gz, por lo que lo primero es descomprimirlo:

```
nazaretroque@nazaretroque: ~/Descargas
Archivo Editar Ver Buscar Terminal Ayuda
nazaretroque@nazaretroque:~/Descargas$ gunzip volatility-2.3.1.tar.gz
nazaretroque@nazaretroque:~/Descargas$ tar -xvf volatility-2.3.1.tar
volatility-2.3.1/
volatility-2.3.1/README.txt
```

Tras esto, lo movemos al directorio que le corresponde, /opt; y compilamos e instalamos la herramienta en el sistema, como muestran las dos imágenes siguientes (la primera mueve la herramienta a /opt y la compila y la segunda lo instala).

```
nazaretroque@nazaretroque: /opt/volatility-2.3.1
Archivo Editar Ver Buscar Terminal Ayuda
nazaretroque@nazaretroque:~/Descargas$ sudo mv volatility-2.3.1 /opt/
nazaretroque@nazaretroque:~/Descargas$ cd /opt/volatility-2.3.1/
nazaretroque@nazaretroque:/opt/volatility-2.3.1$ make
python setup.py build
running build
```

```
nazaretroque@nazaretroque: /opt/volatility-2.3.1
Archivo Editar Ver Buscar Terminal Ayuda
nazaretroque@nazaretroque:/opt/volatility-2.3.1$ sudo make install
python setup.py install
running install
running build
running build.py
```

Tras esto, descargamos los perfiles de Ubuntu desde github y los añadimos a volatility:



```
nazaretroque@nazaretroque: /opt/volatility-2.3.1
Archivo Editar Ver Buscar Terminal Ayuda
nazaretroque@nazaretroque:/opt/volatility-2.3.1$ ls -l profiles/Linux/
total 24
drwxr-xr-x 4 nazaretroque nazaretroque 4096 dic 11 00:50 CentOS
drwxr-xr-x 4 nazaretroque nazaretroque 4096 dic 11 00:50 Debian
drwxr-xr-x 4 nazaretroque nazaretroque 4096 dic 11 00:50 Fedora
drwxr-xr-x 4 nazaretroque nazaretroque 4096 dic 11 00:50 OpenSUSE
drwxr-xr-x 4 nazaretroque nazaretroque 4096 dic 11 00:50 RedHat
drwxr-xr-x 4 nazaretroque nazaretroque 4096 dic 11 00:50 Ubuntu
nazaretroque@nazaretroque:/opt/volatility-2.3.1$ cp -r profiles/Linux/Ubuntu/ /o
pt/volatility-2.3.1/volatility/plugins/overlays/linux/
nazaretroque@nazaretroque:/opt/volatility-2.3.1$
```

Una vez hecho esto, solo queda ejecutar la prueba. La captura añadida aquí no utiliza la opción `--profile` pero también la he utilizado con los perfiles que he explicado anteriormente con los que he sacado las distintas imágenes. Pero con todos da la misma salida que a continuación:

```
nazaretroque@nazaretroque:/opt/volatility-2.3.1$ vol.py pslist -f /home/nazaretroque/evidencias/volcado_ram.lime
Volatility Foundation Volatility Framework 2.3.1
No suitable address space mapping found
Tried to open image as:
MachOAddressSpace: mac: need base
LimeAddressSpace: lime: need base
WindowsHiberFileSpace32: No base Address Space
WindowsCrashDumpSpace64: No base Address Space
HPAKAddressSpace: No base Address Space
VirtualBoxCoreDumpElf64: No base Address Space
VMWareSnapshotFile: No base Address Space
WindowsCrashDumpSpace32: No base Address Space
AMD64PagedMemory: No base Address Space
IA32PagedMemoryPae: No base Address Space
IA32PagedMemory: No base Address Space
MachOAddressSpace: MachO Header signature invalid
MachOAddressSpace: MachO Header signature invalid
LimeAddressSpace: Invalid Lime header signature
WindowsHiberFileSpace32: No xpress signature found
WindowsCrashDumpSpace64: Header signature invalid
HPAKAddressSpace: Invalid magic found
VirtualBoxCoreDumpElf64: ELF64 Header signature invalid
VMWareSnapshotFile: Invalid VMware signature: 0x0
WindowsCrashDumpSpace32: Header signature invalid
AMD64PagedMemory: Incompatible profile WinXPSP2x86 selected
IA32PagedMemoryPae: No valid DTB found
IA32PagedMemory: No valid DTB found
FileAddressSpace: Must be first Address Space
ArmAddressSpace: No valid DTB found
nazaretroque@nazaretroque:/opt/volatility-2.3.1$ vol.py linux_pslist -f /home/nazaretroque/evidencias/volcado_ram.lime
Volatility Foundation Volatility Framework 2.3.1
ERROR : volatility.commands : This command does not support the profile WinXPSP2x86
nazaretroque@nazaretroque:/opt/volatility-2.3.1$
```

Un hecho curioso es que el segundo comando de la captura anterior utiliza un plugin de linux, que es supuestamente el sistema operativo del que he sacado el volcado; no obstante, volatility indica que ese plugin no es soportado por el volcado que le he dado, puesto que el perfil es de Windows, algo que es imposible porque LiME es una herramienta de Linux (Linux Memory Extractor).