Práctica 4. Análisis forense de sistemas Linux.

La volatilidad en Linux.

Resumiendo lo estudiado en clase y lo experimentado en la práctica 2 anterior, hay varias formas de tomar una imagen de la memoria de un sistema Linux. Además de realizar un volcado de memoria de la máquina virtual (VMware, VirtualBox, KVM, etc.) o tomar imágenes a través de la interfaz Firewire, hay dos formas de realizar volcados de memoria a un sistema en ejecución a través de herramienta de adquisición en vivo:

1. Copiar /dev/mem directamente con las herramientas del espacio de usuario. El comando típico podría ser el siguiente:

```
dd if = /dev/mem of=test.dump bs=512 conv=noerror
```

Sin embargo, esto solo funciona si la opción del kernel CONFIG_STRICT_DEVMEM no está configurada. Se puede chequear si esta característica está configurada buscando en /boot/config-$(uname -r). Si está configurada, el espacio de usuario lee de /dev/mem o /dev/kmem no puede ir más allá del primer megabyte. Si no está configurado, LiME (ver más abajo) no es necesario y se puede tomar un volcado del kernel de la forma antes mencionada.

El uso de LiME tiene dos otras ventajas: en primer lugar, se puede utilizar en sistemas Android y, en segundo lugar, se puede volcar en un formato compatible con depuradores de kernel estándar.

2. Usando un módulo de kernel especial (LiME o fmem). Se tratan de módulos de kernel cargables (LKM) que permiten la adquisición de memoria volátil desde Linux y dispositivos basados en Linux, como Android. LiME en particular es la primera herramienta que permite realizar capturas de memoria completa en dispositivos Android. También minimiza su interacción entre el usuario y los procesos de espacio del kernel durante la adquisición, lo que le permite producir capturas de memoria que son más sólidas desde el punto de vista forense que las de otras herramientas diseñadas para la adquisición de memoria Linux.

A continuación se ofrece una descripción general de las herramientas utilizadas en este ejercicio:

- **Volatility:** Es un software opensource para la extracción de artefactos digitales de la memoria volátil (RAM). Está desarrollado y respaldado por The Volatility Foundation.
- **Python:** Es un lenguaje de programación interpretado requerido para ejecutar Volatility. Por lo menos se requiere la versión 2.6 (mejor 2.7). Los desarrollos de Volatility actuales no soportan Python 3, por lo que desde 2019 se viene desarrollando Volatility3 desde cero para adaptarlo a Python3. Hay que tener en

cuenta que Volatility necesita varias bibliotecas de Python que normalmente no están instaladas automáticamente al instalar el paquete principal de Python.

- **Distorm3**: Se trata de una biblioteca de desensamblador para arquitecturas x86 / AMD64. La biblioteca convierte un flujo de datos binarios en instrucciones de ensamblador, representadas como estructuras de datos de Python. La biblioteca es necesaria para Volatility complementos apihooks, impscan, callbacks, volshell, linux_volshell y mac_volshell.

- **Yara**: Se trata de una herramienta para clasificar malware, i. mi. crear firmas de malware. Se necesita para la opción “yarascan” de Volatility.

Objetivo:

- Aprender a realizar análisis básicos de volcados de memoria en Linux.

Materiales

- Una distribución Linux cualquiera
- Volatility 2
- Volatility 3

Se pide crear o usar una máquina virtual Linux dónde se realizarán las siguientes tareas:

1. **Descargar el volcado de memoria de una máquina Linux que puedes encontrar aquí.**

2. **Volatility 2**

    a. **Instalar la herramienta Volatillity 2.**

    b. **Preparar Volatility 2 para trabajar con determinados perfiles Linux (linux overlays)**

        - Descarga aquí perfil de Volatility que se ajusta al volcado del apartado (1) y copialo en el lugar adecuado (overlays/linux) para poder utilizarlo con volatility 2.
        - Si sólo dispusiera de volcado de memoria, ¿cómo podría determinar la versión del sistema operativo a la que pertenece? (Modificador banner?)

    c. **Realizar análisis básico de los volcados de memoria en Linux.**

    Utiliza el comando “vol.py” y el volcado de memoria del apartado anterior para describir qué información podemos obtener usando los modificadores siguientes:

    - Análisis de procesos: **linux_pslist, linux_psaux, linux_pstree, linux_cpuinfo**

- Análisis de red: **linux_arp, linux_ifconfig, linux_route_cache, linux_netstat**

- Ficheros y análisis del kernel: **linux_enumerate_files, linux_find_file, linux_recover_filesystem, linux_mount, linux_mount_cache, linux_bash, linux_dmesg**

d. Generar perfiles específicos para Volatility. Sigue los pasos estudiados en clase para generar (module.dwarf y /boot/config/System.map) un perfil específico para Volatility del sistema operativo Linux que estés utilizando en esta práctica.

3. Volatility 3

a. Instalar la herramienta Volatility 3

b. **Preparar Volatility 3 para trabajar con determinados perfiles Linux**

- Descarga aquí perfil de Volatility que se ajusta al volcado del punto (1) y copialo en el directorio SYMBOLS de Volatility 3.

c. **Realizar análisis básico de los volcados de memoria en Linux: banners.Banners, linux.bash.Bash, linux.kmsg.Kmsg, linux.lsmod.Lsmod, linux.lsof.Lsof, linux.malfind.Malfind, linux.mountinfo.MountInfo, linux.proc.Maps, linux.psaux.PsAux, linux.pslist.PsList, linux.psscan.PsScan, linux.pstree.PsTree, linux.sockstat.Sockstat**

d. Generar perfiles específicos para Volatility. Sigue los pasos estudiados en clase para generar (dwarf2json, vmlinux y System.map) un perfil específico para Volatility del sistema operativo Linux que estés utilizando en esta práctica.