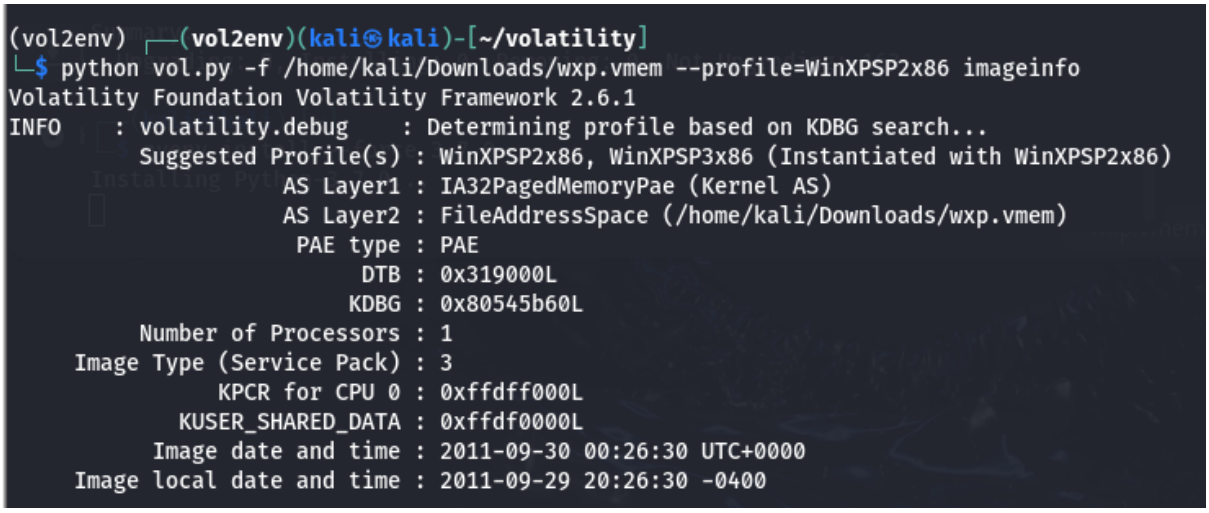
PRACTICA 2: FUNDAMENTOS DE LA SEGURIDAD EN EL SOFTWARE Y EN LOS COMPONENTES

FORENSE: ANÁLISIS DE MEMORIA

Paula Daniela Sanchez Rodriguez | Enrique García Cuadrado

Parte 1. Análisis de memoria general.



En esta práctica, realizaremos un análisis de memoria utilizando Volatility.

En esta primera sección, hemos usado el comando imageinfo para obtener información general de la memoria capturada en la imagen wxp.vmem.

Este análisis inicial nos dará detalles como el perfil de sistema sugerido, la fecha y hora de captura, el tipo de arquitectura y otros datos del entorno del sistema operativo analizado.

a) Número total y listado de procesos (indicando si observa alguno sospechoso) y su relación con otros.

```
(vol2env) ┌──(vol2env)(kali㉿kali)-[~/volatility]
└─$ python vol.py -f /home/kali/Downloads/wxp.vmem --profile=WinXPSP2x86 pslist
Volatility Foundation Volatility Framework 2.6.1
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ------------------------------
0x819cc830 System                    4      0     60      209 ------      0
0x818efda0 smss.exe                384      4      3       19 ------      0 2011-09-26 01:33:32 UTC+0000
0x81616ab8 csrss.exe               612    384     12      473      0      0 2011-09-26 01:33:35 UTC+0000
0x814c9b40 winlogon.exe            636    384     16      498      0      0 2011-09-26 01:33:35 UTC+0000
0x81794d08 services.exe            680    636     15      271      0      0 2011-09-26 01:33:35 UTC+0000
0x814a2cd0 lsass.exe               692    636     24      356      0      0 2011-09-26 01:33:35 UTC+0000
0x815c2630 vmacthlp.exe            852    680      1       25      0      0 2011-09-26 01:33:35 UTC+0000
0x81470020 svchost.exe             868    680     17      199      0      0 2011-09-26 01:33:35 UTC+0000
0x818b5248 svchost.exe             944    680     11      274      0      0 2011-09-26 01:33:36 UTC+0000
0x813a0458 MsMpEng.exe            1040    680     16      322      0      0 2011-09-26 01:33:36 UTC+0000
0x816b7020 svchost.exe            1076    680     87     1477      0      0 2011-09-26 01:33:36 UTC+0000
0x817f7548 svchost.exe            1200    680      6       81      0      0 2011-09-26 01:33:37 UTC+0000
0x8169a1d0 svchost.exe            1336    680     14      172      0      0 2011-09-26 01:33:37 UTC+0000
0x813685e0 spoolsv.exe            1516    680     14      159      0      0 2011-09-26 01:33:39 UTC+0000
0x818f5cd0 explorer.exe           1752   1696     32      680      0      0 2011-09-26 01:33:45 UTC+0000
0x815c9638 svchost.exe            1812    680      4      102      0      0 2011-09-26 01:33:46 UTC+0000
0x8192d7f0 VMwareTray.exe         1876   1752      3       84      0      0 2011-09-26 01:33:46 UTC+0000
0x818f6458 VMwareUser.exe         1888   1752      9      245      0      0 2011-09-26 01:33:47 UTC+0000
0x8164a020 msseces.exe            1900   1752     11      205      0      0 2011-09-26 01:33:47 UTC+0000
0x81717370 ctfmon.exe             1912   1752      3       93      0      0 2011-09-26 01:33:47 UTC+0000
0x813a5b28 svchost.exe            2000    680      6      119      0      0 2011-09-26 01:33:47 UTC+0000
0x81336638 vmtoolsd.exe            200    680      5      234      0      0 2011-09-26 01:33:47 UTC+0000
0x81329b28 VMUpgradeHelper         424    680      5      100      0      0 2011-09-26 01:33:48 UTC+0000
0x812d6020 wscntfy.exe            2028   1076      3       63      0      0 2011-09-26 01:33:55 UTC+0000
0x812c1718 TPAutoConnSvc.e        2068    680      5       99      0      0 2011-09-26 01:33:55 UTC+0000
0x812b03e0 alg.exe                2272    680      7      112      0      0 2011-09-26 01:33:55 UTC+0000
0x81324020 TPAutoConnect.e        3372   2068      3       90      0      0 2011-09-26 01:33:59 UTC+0000
0x814e7b38 msiexec.exe            2396    680      5      127      0      0 2011-09-26 01:34:45 UTC+0000
0x814db608 cmd.exe                3756   1752      3       56      0      0 2011-09-30 00:20:44 UTC+0000
0x812f59a8 cmd.exe                3128    200      0 --------      0      0 2011-09-30 00:26:30 UTC+0000   2011-09-30 00:26:30 UTC+0000

(vol2env) ┌──(vol2env)(kali㉿kali)-[~/volatility]
└─$ python vol.py -f /home/kali/Downloads/wxp.vmem --profile=WinXPSP2x86 pslist | sed '1d;$d' | wc -l
Volatility Foundation Volatility Framework 2.6.1
30
```

Número total de procesos: 30

Listado de procesos:

- **System, smss.exe, csrss.exe, winlogon.exe, services.exe, lsass.exe**: Estos son procesos del sistema normales y necesarios en windows XP. No muestran ningún comportamiento raro.
- **Svchost.exe:** Es común ver muchas instancias de svchost.exe en windows ya que este proceso administra muchos servicios. El 1076 llama la atención porque tiene un número alto de handles o punteros a recursos del sistema 1477 en total, lo cual puede ser normal dependiendo del servicio que esté ejecutando, pero también es un punto para investigar más.
- **MsMpEng.exe:** Este proceso corresponde a las herramientas de Security Essentials que vienen por defecto con el antivirus de microsoft. No debería ser sospechoso.
- **vmacthlp.exe**, **VMwareTray.exe**, **VMwareUser.exe** y **vmtoolsd.exe**: Estos procesos están relacionados con herramientas de VMware. Como la imagen es posible que venga de una máquina virtualizada, estos procesos no son sospechosos.

- **spoolsv.exe** es el servicio de cola de impresión de Windows. En entornos sin impresoras este proceso puede ser sospechoso. Su número de handles está dentro de los límites.
- **explorer.exe** es el proceso de la interfaz de usuario de Windows, por lo que estar ahí es normal y esperado.
- **cmd.exe** puede ser normal, pero es raro ver dos instancias, especialmente con un intervalo de tiempo entre ellas (2011-09-30 00:20:44 y 2011-09-30 00:26:30). Esto podría indicar actividad del usuario o ser señal de un malware que utilizó la línea de comandos.

Posibles procesos sospechosos:

- svchost.exe con PID 1076, debido al alto número de handles.
- cmd.exe (PID 3756 y 3128) por la existencia de múltiples instancias a diferentes horas.
- Es recomendable comprobar también la ruta de MsMpEng.exe para confirmar la autenticidad.

Relación entre procesos

- Explorer.exe tiene varios procesos hijos (VMwareTray.exe, VMwareUser.exe, msseces.exe, ctfmon.exe), lo cual es normal.
- services.exe es el proceso padre de múltiples instancias de svchost.exe, lo cual también es esperado.

b) Número total y listado de conexiones (y procesos asociados a cada conexión).

```
(vol2env) ┌──(vol2env)(kali㉿kali)-[~/volatility]
└─$ python vol.py -f /home/kali/Downloads/wxp.vmem --profile=WinXPSP2x86 connscan
Volatility Foundation Volatility Framework 2.6.1
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------ ------------------------ ---
0x014f6ab0 10.0.0.109:1072          209.190.4.84:443          1752
0x01507380 10.0.0.109:1073          209.190.4.84:443          1752
0x016c2b00 10.0.0.109:1065          184.173.252.227:443       1752
0x017028a0 10.0.0.109:1067          184.173.252.227:443       1752
0x01858cb0 10.0.0.109:1068          209.190.4.84:443          1752
```

- **Número total de conexiones**: Hay un total de 5 conexiones activas.
- **Direcciones locales y remotas**: Las conexiones están establecidas desde la dirección IP local 10.0.0.109 en varios puertos. Las direcciones remotas son la 209.190.4.84 en tres conexiones y la 184.173.252.227 en dos conexiones.

- Las conexiones a direcciones IP externas en el puerto 443 desde explorer.exe podrían ser indicativas de actividad maliciosa, ya que es raro que el proceso explorer.exe maneje conexiones HTTPS directamente.

c) Número de dlls.

```
(vol2env) ┌──(vol2env)(kali㉿kali)-[~/volatility]
└─$ python vol.py -f /home/kali/Downloads/wxp.vmem --profile=WinXPSP2x86 dlllist | grep -i ".dll" | wc -l
Volatility Foundation Volatility Framework 2.6.1
1257
```

Hay un total de 1257 DLLs cargadas en la memoria.

Explicación del comando:

1. Ejecuta dlllist para listar las DLLs cargadas en la memoria.
2. Usa grep -i ".dll" para filtrar solo las líneas que contienen .dll (indicando archivos DLL).
3. Cuenta las líneas resultantes con wc -l, obteniendo el número total de DLLs.

d) Número de ficheros.

```
(vol2env) ┌──(vol2env)(kali㉿kali)-[~/volatility]
└─$ python vol.py -f /home/kali/Downloads/wxp.vmem --profile=WinXPSP2x86 filescan | wc -l
Volatility Foundation Volatility Framework 2.6.1
1123
```

Hay un total de 1123 archivos abiertos en la memoria.

Explicación del comando:

1. Ejecuta filescan para listar los archivos abiertos.
2. Usa wc -l para contar todas las entradas, lo que te da el número de archivos abiertos.

e) Número de variables de entorno.

```
(vol2env) ┌──(vol2env)(kali㉿kali)-[~/volatility]
└─$ python vol.py -f /home/kali/Downloads/wxp.vmem --profile=WinXPSP2x86 envars | wc -l
Volatility Foundation Volatility Framework 2.6.1
605
```

Hay un total de 605 variables de entorno en la memoria.

Explicación del comando:

1. Ejecuta envars para mostrar las variables de entorno por proceso.
2. Usa wc -l para contar las líneas y obtener el número total de variables de entorno.

f) Número de servicios registrados.

```
(vol2env) ┌──(vol2env)(kali㉿kali)-[~/volatility]
└─$ python vol.py -f /home/kali/Downloads/wxp.vmem --profile=WinXPSP2x86 svcscan | grep -i "service" | wc -l
Volatility Foundation Volatility Framework 2.6.1
1030
```

Hay un total de de 1030 servicios registrados en la memoria.

Este comando:

1. Ejecuta svcscan para listar los servicios de Windows.
2. Usa grep -i "service" para filtrar las líneas relacionadas con servicios.
3. Cuenta las líneas para obtener el número total de servicios registrados.

g) Con las verificaciones realizadas anteriormente ¿se podría identificar algún código malicioso?

Hay varios indicadores de actividad potencialmente maliciosa:

- **Conexiones inusuales de explorer.exe** a direcciones IP externas en el puerto 443, lo cual puede indicar una comunicación encubierta.
- **Alto número de handles en svchost.exe (PID 1076)**, lo que podría sugerir la ejecución de un servicio sospechoso o un proceso inyectado.
- **Múltiples instancias de cmd.exe** abiertas, lo que es atípico y podría ser un intento de ejecutar comandos maliciosos en segundo plano.

Estos factores sugieren que **podría haber código malicioso** en la imagen de memoria, posiblemente utilizando explorer.exe y svchost.exe para evadir la detección y comunicarse con servidores externos.

Parte 2. Análisis de malware

En el análisis de la imagen de memoria **cridex.vmem** usando el comando malfind, se han identificado varios posibles indicadores de actividad maliciosa. Aquí están los hallazgos detallados sobre los procesos y áreas de memoria sospechosas:

1. Procesos y áreas de memoria

- **Proceso csrss.exe (PID 584)**:
  - Dirección de memoria: 0x7f6f0000
  - Protección: PAGE_EXECUTE_READWRITE, lo que permite tanto la ejecución como la escritura en esta parte de la memoria, lo cual es raro para un proceso legítimo y podría indicar código inyectado.
  - Las instrucciones en esta parte incluyen operaciones de entrada y salida además de manipulación de registros, lo cual no es típico para csrss.exe y podría indicar un shellcode o payload inyectado.
- **Proceso winlogon.exe (PID 608)**:
  - Se han encontrado varias áreas de memoria con permisos PAGE_EXECUTE_READWRITE en winlogon.exe, en direcciones como 0x13410000, 0xf9e0000, 0x4ee0000, y otras.
  - Estas regiones de memoria contienen patrones repetitivos y operaciones triviales (ADD [EAX], AL), lo cual podría ser un intento de ocultar código malicioso entre operaciones buenas o un loader que aún no ha ejecutado su carga maliciosa.
  - Estas características son comunes en malware que utiliza técnicas de evasión aprovechando procesos críticos del sistema como winlogon.exe para pasar desapercibido.
- **Proceso explorer.exe (PID 1484)**:
  - Dirección de memoria: 0x1460000
  - Se observa una estructura de encabezado "MZ", que indica el inicio de un archivo ejecutable en el formato PE (Portable Executable). La presencia de esta firma en la memoria de explorer.exe podría indicar que un archivo ejecutable malicioso fue inyectado directamente en la memoria de este proceso. Esto es bastante sospechoso, ya que explorer.exe no debería contener ejecutables en su espacio de memoria a menos que haya sido comprometido. Esto sugiere una técnica de inyección en la que se carga un ejecutable directamente en el proceso explorer.exe para ejecutar código malicioso.

- **Proceso reader_sl.exe (PID 1640)**:
    - Dirección de memoria: 0x3d0000
    - También presenta una firma "MZ" de un archivo ejecutable, lo que implica que un ejecutable pudo haber sido cargado en la memoria de este proceso. reader_sl.exe está asociado con Adobe Reader y, al ser una aplicación común, puede ser blanco de ataques para esconder malware.

2. Tipo de malware

- La presencia de segmentos de memoria con permisos de ejecución y escritura (PAGE_EXECUTE_READWRITE) y la existencia de encabezados "MZ" en los procesos críticos (winlogon.exe, explorer.exe, y reader_sl.exe) sugiere una posible infección con un **loader o dropper**. Este tipo de malware suele inyectarse en procesos del sistema para ejecutar código malicioso sin levantar sospechas.
- La técnica utilizada parece ser **inyección de código** en procesos de alto nivel y servicios críticos de Windows, lo cual es típico de malware avanzado que intenta evitar la detección aprovechando procesos confiables.
- Dado que esta imagen se llama "cridex", podría estar relacionada con el troyano bancario **Cridex**, conocido por robar credenciales de banca en línea y propagarse a través de procesos inyectados y técnicas de evasión.

3. Ficheros y recursos infectados

- No se han identificado nombres de archivos específicos en el análisis de malfind. Sin embargo la inyección en explorer.exe, winlogon.exe, y reader_sl.exe sugiere que estos procesos están infectados. La inyección en procesos reales les permite a los atacantes acceder a recursos del sistema y a datos del usuario sin ser detectados.
- Dado que Cridex es un virus conocido por robar información y comunicarse con servidores de comando y control, es probable que también existan conexiones de red activas desde estos procesos.

A partir del análisis del otro archivo **wxp.vmem**, con el comando malfind hemos visto muchos segmentos de memoria en procesos que presentan patrones indicativos de posible actividad sospechosa.

Procesos con actividad sospechosa

1. **csrss.exe (PID 612)**:
    a. **Dirección de memoria**: 0x7f6f0000

    b. **Permisos**: PAGE_EXECUTE_READWRITE, que permite la ejecución y escritura, algo poco común en procesos legítimos del sistema y generalmente un indicador de inyección de código.
    c. **Instrucciones anómalas**: Contiene múltiples instrucciones de manipulación de registros y operaciones triviales (ADD, OUT, OR), que pueden formar parte de un payload o shellcode malicioso.
2. **winlogon.exe (PID 636)**:
    a. **Múltiples áreas de memoria sospechosas**: Entre ellas, las direcciones 0x177a0000, 0xfe50000, 0x24770000, y 0x436f0000.
    b. **Encabezado "MZ"**: Varias de estas áreas de memoria contienen un encabezado "MZ", lo que indica un posible archivo ejecutable PE cargado en memoria. Esto sugiere una técnica de inyección que permite al malware operar desde winlogon.exe, aprovechando su legitimidad y su acceso a servicios críticos del sistema.
    c. **Instrucciones de interrupción (INT 3) y saltos**: Estas podrían estar presentes para desviar el flujo de ejecución y realizar actividades maliciosas o cargadores que se ejecutan en etapas.
3. **explorer.exe (PID 1752)**:
    a. **Direcciones de memoria**: 0x3380000 y 0x36e0000
    b. **Indicador de encabezado PE ("MZ")**: Este encabezado muestra que un archivo ejecutable podría estar cargado en memoria. Inyectar código en explorer.exe es una técnica común para malware que intenta realizar actividades maliciosas sin ser detectado, ya que explorer.exe tiene permisos de usuario y acceso a recursos del sistema.
4. **Otros procesos**:
    a. **VMwareTray.exe (PID 1876)**, **VMwareUser.exe (PID 1888)**, **msseces.exe (PID 1900)**, y **cmd.exe (PID 3756)** también contienen encabezados "MZ" en áreas de memoria con permisos de ejecución y escritura. Esto sugiere que podrían estar comprometidos o siendo utilizados por el malware para ejecutar funciones adicionales o persistir en el sistema.

Tipo de Malware y Análisis

La evidencia sugiere la presencia de un **loader** o **dropper** de malware, con un posible **comportamiento de inyección de código**. Algunos aspectos clave que apoyan esta conclusión son:

- **Inyección de código en procesos críticos**: csrss.exe, winlogon.exe, y explorer.exe son procesos esenciales en Windows y su compromiso es típico de malware que intenta esconderse usando la inyección de procesos. Esto permite

al código malicioso aprovechar los permisos de estos procesos y ejecutar operaciones sin levantar sospechas.

- **Uso de encabezados PE ("MZ") en memoria**: La presencia de encabezados "MZ" en varios procesos indica que archivos ejecutables han sido inyectados directamente en memoria, lo cual es una táctica común en malware sofisticado para evitar ser detectado en el sistema de archivos.
- **Permisos de ejecución y escritura en memoria**: Las secciones de memoria con permisos PAGE_EXECUTE_READWRITE son ideales para que el malware ejecute su código sin restricciones.

Parte 3. Generación de archivo vmem

Para el caso de estudio se proporciona un ejemplo de un volcado de memoria concreto. Es interesante conocer y probar el procedimiento a seguir para obtener nuestro propio volcado para la máquina que estemos analizando.

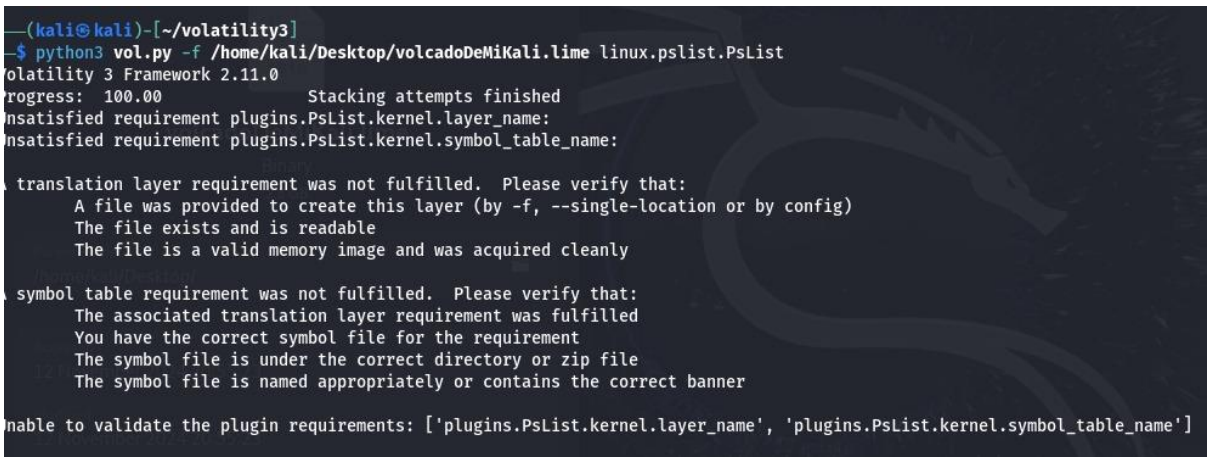
- **Explique el procedimiento que debería seguir para obtener este volcado de memoria de una máquina virtual propia.**

Para crear un volcado de memoria de una máquina virtual en VMware, hay que hacer un snapshot completo de la VM, lo cual generará un archivo .vmem en el directorio de la máquina virtual.

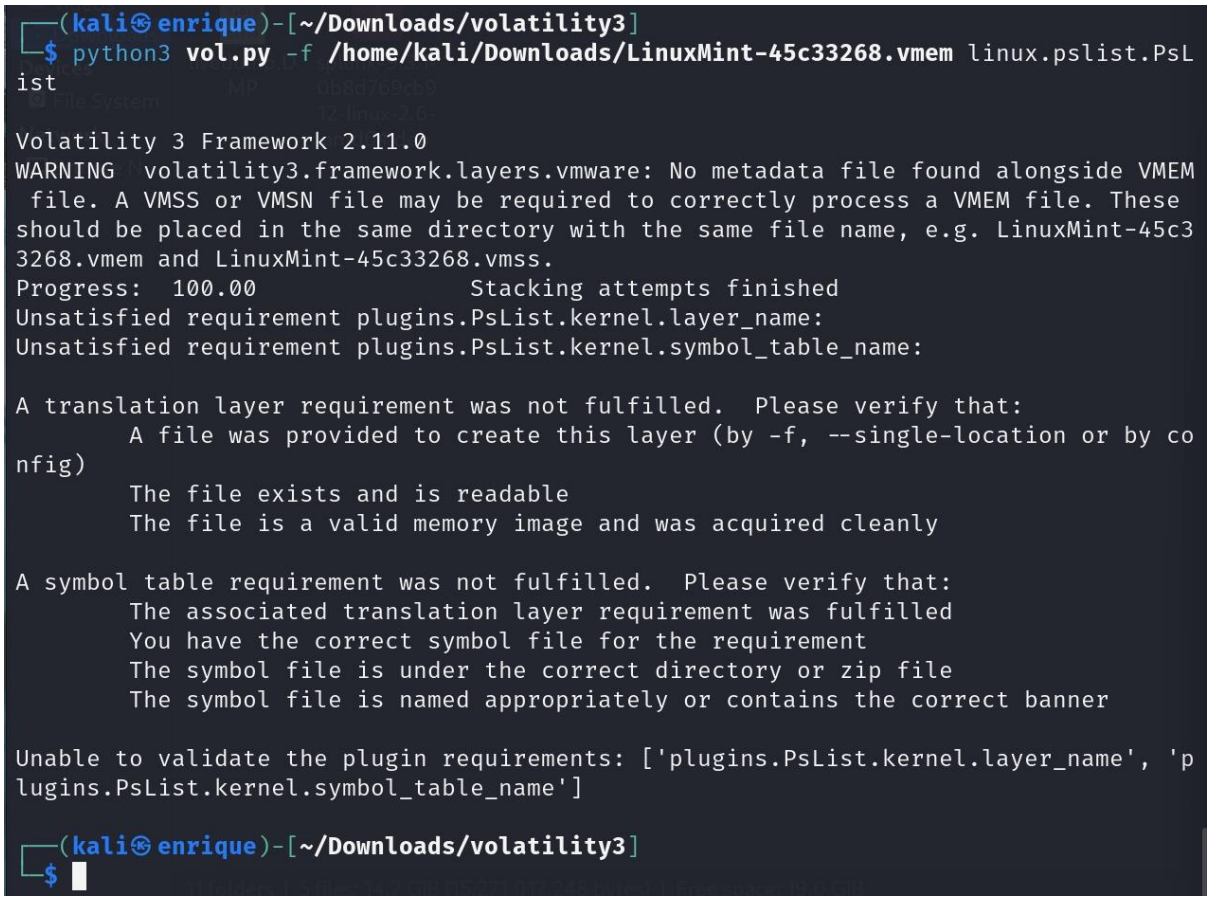
Se puede forzar un volcado de memoria seleccionando la opción "Suspend" para detener la máquina y generar el archivo .vmem en el msimo directorio.

- **Compruebe si puede realizar el análisis posterior de forma análogo a los apartados anteriores.**

En esta parte hemos tenido problemas a la hora de analizar las imágenes creadas por nosotros mismos. En un primer momento hemos intentado analizar la imagen de la memoria de un Windows 11 y después la de un Linux Mint.



En este punto creíamos que podría ser por la extensión .lime debido a que no tuviésemos bien configurado el volatility pero hemos probado con otras extensiones como .vmem iguales que las anteriores imágenes de memoria de la práctica y tampoco nos ha funcionado ni con volatility2 ni con volatility3.



El análisis como se puede ver en la imagen se completa al 100% pero da un error de símbolos que no se ha podido solucionar

- **¿Podría obtener esta información desde una máquina física (total o parcialmente)? Desarrolle la respuesta.**

Esto sí sería posible usando herramientas especializadas como **WinPMem** (para Windows) que permite realizar un volcado de la memoria física de sistemas Windows en tiempo real y en donde sólo es necesario ejecutar el siguiente comando:

winpmem.exe --output C:\ruta\del\volcado.raw.

Este archivo .raw se puede analizar en Volatility exactamente igual que una imagen de máquina virtual.

LiME (Linux Memory Extractor) es una herramienta desarrollada específicamente para sistemas Linux que permite extraer el contenido completo de la memoria RAM y generar un archivo en formato .lime, adecuado para su posterior análisis forense. Este formato es compatible con herramientas como Volatility y es de los más usados en el sector.

Para utilizar LiME, solo es necesario compilar e instalar la herramienta en la máquina Linux de destino y luego ejecutar el siguiente comando:

insmod lime-6.11.2-amd64.ko "path=/home/Kali/Desktop/volcadoDeMiKali.lime format=lime"

- insmod es el comando utilizado para cargar el módulo de LiME en el kernel de Linux.
- lime-6.11.2-amd64.ko es el nombre del archivo de módulo que corresponde a la versión del kernel en el sistema.
- path=especifica la ruta de destino donde se almacenará el volcado de memoria.
- format=lime indica que el archivo de volcado debe guardarse en el formato .lime que se puede analizar con Volatility al igual que con los anteriores formatos.

Ejemplo de uso: