Forense 3

Informe Forense

Análisis de Evidencias en Memoria y Disco
Por: Marcos Fernández Sequeiros
Fecha: 27/02/2025

Índice

Caso 1: Análisis de la imagen "caso1Volatility.dmp"

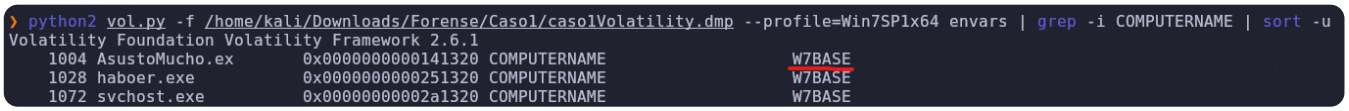
Caso 2: Análisis de la imagen "Windows11.dmp"

Caso 1: Análisis de la imagen "caso1Volatility.dmp"

1. Determinación del nombre del equipo

Procedimiento:

- **Herramienta utilizada:**
  - Volatility (v2 en Python 2).
- **Comando ejecutado:**
  - Se usó el plugin `envars` para extraer las variables de entorno de los procesos, filtrando la salida para localizar la variable `COMPUTERNAME` y ordenando los resultados:
  - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 envars | grep -i COMPUTERNAME | sort -u`
- **Observación:**
  La salida muestra la línea:
  `COMPUTERNAME=W7BASE`
  lo que indica de manera directa el nombre asignado al equipo.
- **Captura ilustrativa:**



Análisis:

- La variable `COMPUTERNAME` es un identificador confiable en sistemas Windows, ya que se encuentra almacenada en la memoria y se asocia de forma única a cada equipo y proceso en ejecución.
- La coincidencia exacta "COMPUTERNAME=W7BASE" confirma que el sistema analizado corresponde al equipo identificado como **W7BASE**, lo cual es crucial cuando se trabaja con múltiples imágenes o se cotejan evidencias de diferentes fuentes.

Conclusión:

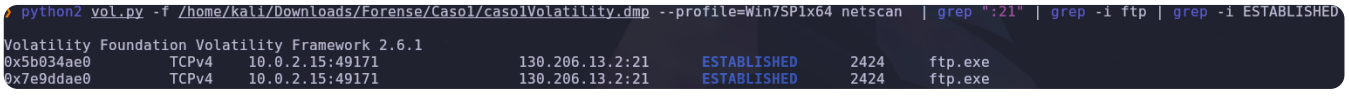
Se concluye que el equipo bajo análisis se denomina **W7BASE**. Este dato resulta fundamental para relacionar la imagen forense con la máquina real, en escenarios donde intervengan múltiples dispositivos.

---

2. Conexión FTP con organismo público español

Procedimiento:

- **Detección de conexiones activas:**
  - Se empleó el plugin `netscan` de Volatility para identificar conexiones de red. Al filtrar específicamente por el puerto 21 (usado por FTP) y el estado "ESTABLISHED", se ejecutó:
    - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 netscan | grep ":21" | grep -i ftp | grep -i ESTABLISHED`
  - **Resultado obtenido:**
    La salida muestra dos líneas idénticas:
    0x5b034ae0 TCPv4 10.0.2.15:49171 130.206.13.2:21 ESTABLISHED 2424 ftp.exe
    0x7e9ddae0 TCPv4 10.0.2.15:49171 130.206.13.2:21 ESTABLISHED 2424 ftp.exe
- **Captura de pantalla:**



- **Identificación de la IP remota:**
  La dirección IP **130.206.13.2** fue consultada mediante herramientas WHOIS/DNS, determinándose que corresponde a un **organismo público español**.

Análisis:

- El hecho de que el proceso **ftp.exe** (PID 2424) establezca una conexión en el puerto 21 a la dirección 130.206.13.2, y que dicha conexión esté en estado **ESTABLISHED**, indica una sesión FTP activa.
- La vinculación con una IP institucional respalda la hipótesis de que el usuario mantuviera comunicación con un organismo público.

Conclusión:

- **Proceso implicado:** ftp.exe (PID 2424)
- **Dirección remota:** 130.206.13.2
- **Estado de la conexión:** ESTABLISHED

Se confirma que el usuario tenía establecida una sesión FTP con un organismo público español, lo que puede tener implicaciones en la cadena de custodia y en la vinculación del sistema con entidades oficiales.

---

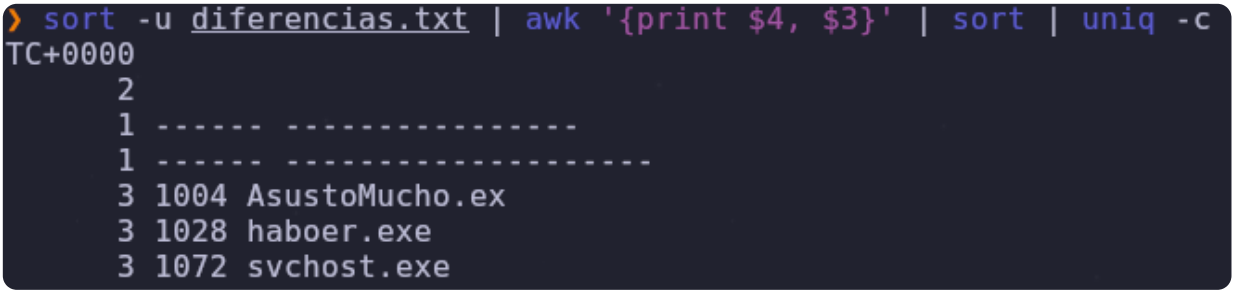
3. Detección de proceso malicioso vinculado a actividad FTP

Procedimiento:

1. **Obtención de listados de procesos:**
   Se ejecutaron dos comandos para extraer la lista de procesos:
   - **pslist:**
     - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 pslist > /home/kali/Downloads/Forense/Caso1/pslist.txt`
   - **psscan:**
     - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 psscan > /home/kali/Downloads/Forense/Caso1/psscan.txt`

2. **Comparación de listados:**
   - Se generó un diff para identificar procesos presentes únicamente en `psscan` (indicativo de ocultamiento):
   - `diff pslist.txt psscan.txt > diferencias.txt`
   - **Capturas de pantalla:**

3. **Filtrado de diferencias:**
   - Con el fin de aislar los PIDs que aparecen únicamente en `psscan`, se ejecutó:
     - `sort -u diferencias.txt | awk '{print $4}' | sort | uniq -c | sort -u`
   - En la captura siguiente se observa la identificación del primer ejecutable sospechoso:
     -



4. **Identificación y análisis de DLLs:**
   - Se determinó que el proceso sospechoso es **AsustoMucho.exe** con **PID 1004**. Para confirmar su naturaleza, se examinó la lista de DLLs asociadas:
     - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 dlllist -p 1004`
   - **Observación importante:**
     La ruta del ejecutable es:
     `C:\ProgramData{f1da652f-fe14-d64d-f1da-a652ffe194bf}\AsustoMucho.exe`
     Se aprecia un **GUID** poco usual, utilizado frecuentemente por malware para disfrazar su actividad.
   - Además, el binario se lanza con el argumento:
     - "C:\ProgramData{f1da652f-fe14-d64d-f1da-a652ffe194bf}\AsustoMucho.exe" --startup=1
   1. - lo que indica un mecanismo de **persistencia** en el arranque.
      - **Captura ilustrativa:**

```
Path
----
C:\ProgramData\{f1da652f-fe14-d64d-f1da-a652ffe194bf}\AsustoMucho.exe
C:\Windows\SYSTEM32\ntdll.dll
C:\Windows\SYSTEM32\wow64.dll
C:\Windows\SYSTEM32\wow64win.dll
C:\Windows\SYSTEM32\wow64cpu.dll
C:\ProgramData\{f1da652f-fe14-d64d-f1da-a652ffe194bf}\AsustoMucho.exe
```

Análisis:

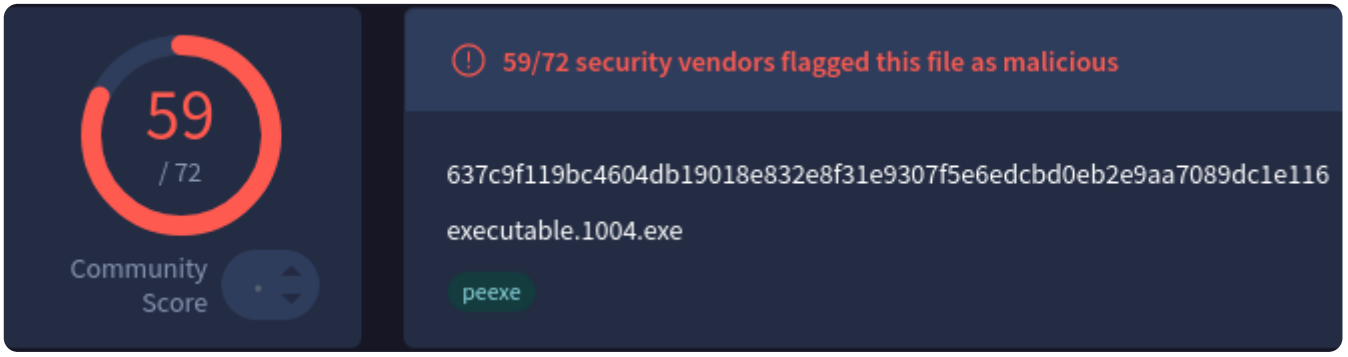
- La presencia exclusiva de **AsustoMucho.exe** en la salida de `psscan` (y su ausencia en `pslist`) es un claro indicador de técnicas de ocultación.
- La ruta con GUID y el uso del parámetro `--startup=1` sugieren que el malware intenta establecer persistencia en el sistema.
- La carga de librerías asociadas a la emulación de procesos de 32 bits en sistemas de 64 bits (como `wow64.dll`, `wow64win.dll` y `wow64cpu.dll`) refuerza la hipótesis de evasión.

Conclusión:

Se identifica que el proceso **AsustoMucho.exe** (PID 1004) es malicioso. Los indicadores clave son:

- Exclusividad en `psscan` (ocultamiento deliberado).
- Ubicación sospechosa en una carpeta con GUID.
- Argumento `--startup=1`, que garantiza su ejecución en cada arranque.
- Carga de librerías de compatibilidad (posible intento de evasión).

Se complementa el análisis con un escaneo en VirusTotal que confirma la naturaleza maliciosa del binario.



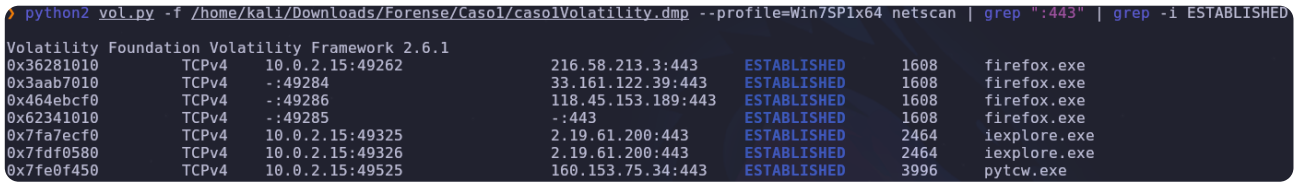
4. Detección de proceso infectado con conexión HTTPS activa

Procedimiento:

1. **Identificación de conexiones HTTPS:**
   - Se filtró la salida del plugin `netscan` para conexiones en el puerto 443 en estado ESTABLISHED:

- python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 netscan | grep ":443" | grep -i ESTABLISHED

- **Captura de pantalla:**



2. **Análisis del proceso asociado:**
   - Se detectó que el proceso **pytcw.exe** tiene una conexión HTTPS establecida con la IP **160.153.75.34**.
   - **Indicadores de sospecha:**
     - El ejecutable **pytcw.exe** no corresponde a un navegador o aplicación conocida (como `firefox.exe` o `iexplore.exe`).
     - Se ejecuta desde una ubicación inusual: `"C:\Users\wadmin\AppData\Local\Temp\pytcw.exe"`, lo cual es típico en casos de malware.
     - La carga de librerías como `wow64.dll` y `schannel.dll` sugiere que, pese a estar en un entorno de 64 bits, se trata de un binario de 32 bits que gestiona comunicaciones seguras, posiblemente para exfiltrar datos sin levantar sospechas.
   - Se profundizó en el análisis ejecutando:
     - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 dlllist -p 3996` (donde 3996 es el PID asociado a **pytcw.exe**).

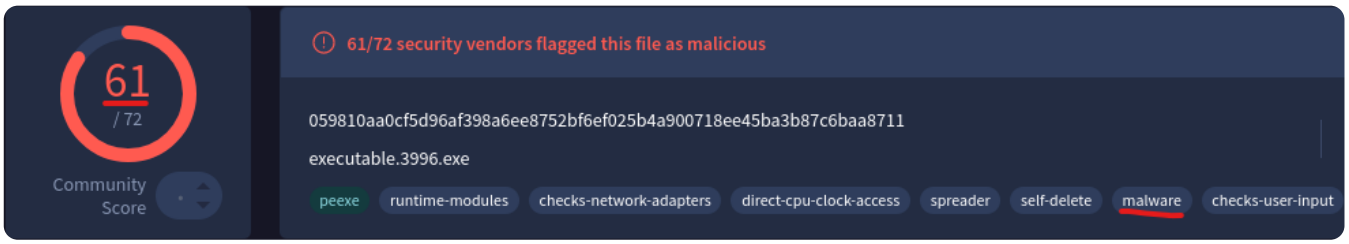
Análisis:

- La combinación de una conexión HTTPS activa con una IP desconocida (**160.153.75.34**) y el hecho de que el proceso se encuentre en la carpeta TEMP, es altamente sugestiva de actividad maliciosa.
- La ubicación inusual y la utilización de librerías propias para cifrado (schannel.dll) indican que el proceso podría estar exfiltrando información de manera encubierta.

Conclusión:

El proceso **pytcw.exe** es considerado malicioso.

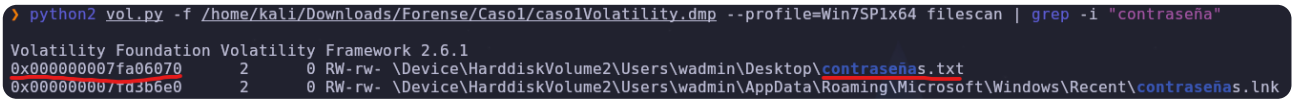
- **Conexión establecida:** HTTPS a la dirección IP **160.153.75.34**.
- **Justificación:** Ubicación en la carpeta TEMP, ejecución de un binario no reconocido, uso de librerías para evasión y cifrado, y confirmación a través de análisis en VirusTotal.



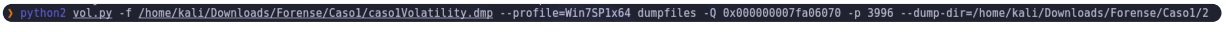
5. Recuperación de la contraseña de un fichero comprimido escrita en el Bloc de Notas

Procedimiento:

1. **Listado de archivos relevantes en la memoria:**
   - Se ejecutó el siguiente comando para buscar en la imagen de memoria cualquier archivo que contenga la palabra "contraseña". Esto permitió identificar de manera precisa el archivo de interés que se encuentra en el escritorio del usuario:
     - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 filescan | grep -i "contraseña"`
   - **Resultado obtenido:**
     - 0x000000007fa06070 2 0 RW-rw- \Device\HarddiskVolume2\Users\wadmin\Desktop\contraseñas.txt
     - - La captura de pantalla respalda la ubicación y existencia del archivo:



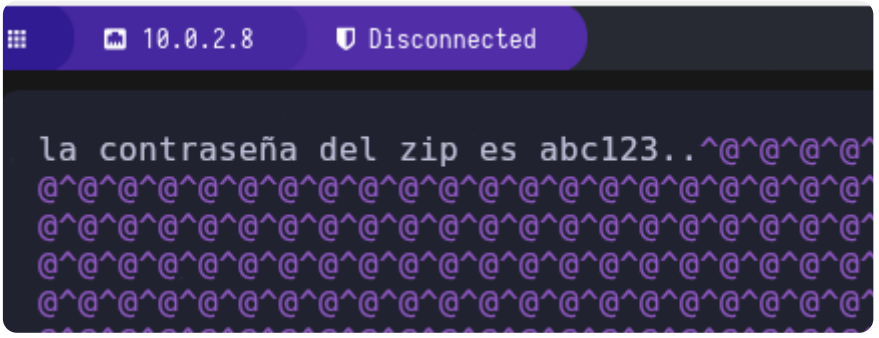
- **Extracción del archivo identificado:**
  - Conociendo la dirección de memoria (0x000000007fa06070) y el proceso relacionado (PID 3996), se procedió a extraer el archivo utilizando el siguiente comando:
    - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 dumpfiles -Q 0x000000007fa06070 -p 3996 --dump-dir=/home/kali/Downloads/Forense/Caso1/2`
  - Durante el proceso se generó el archivo extraído, el cual se nombró de forma automática como:
    - file.None.0xfffffa8001cbc560.dat
  - La extracción fue verificada con la siguiente captura de pantalla:



y se confirmó la generación del archivo mediante:

```
❯ ls
file.None.0xfffffa8001cbc560.dat
```

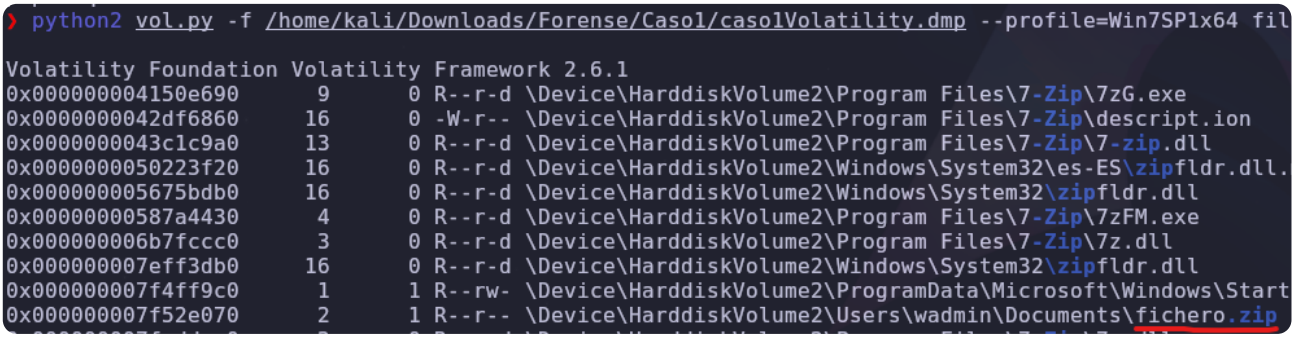
- **Recuperación y análisis del contenido:**
  - Al abrir el archivo extraído se observó que contenía la contraseña necesaria para acceder al fichero comprimido. El contenido textual revelado en el archivo fue:
  - abc123..
  - La captura de pantalla a continuación muestra la visualización del contenido, confirmando la recuperación exitosa de la contraseña:



6. Extracción del contenido de un fichero ZIP en memoria RAM: ¿Qué animal se encuentra dentro?

Procedimiento:

1. **Búsqueda de referencias a archivos ZIP:**
   - Se utilizó el plugin `filescan` para localizar en la memoria referencias a ficheros ZIP:
     - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 filescan | grep -i ".zip"`
   - **Captura de pantalla:**

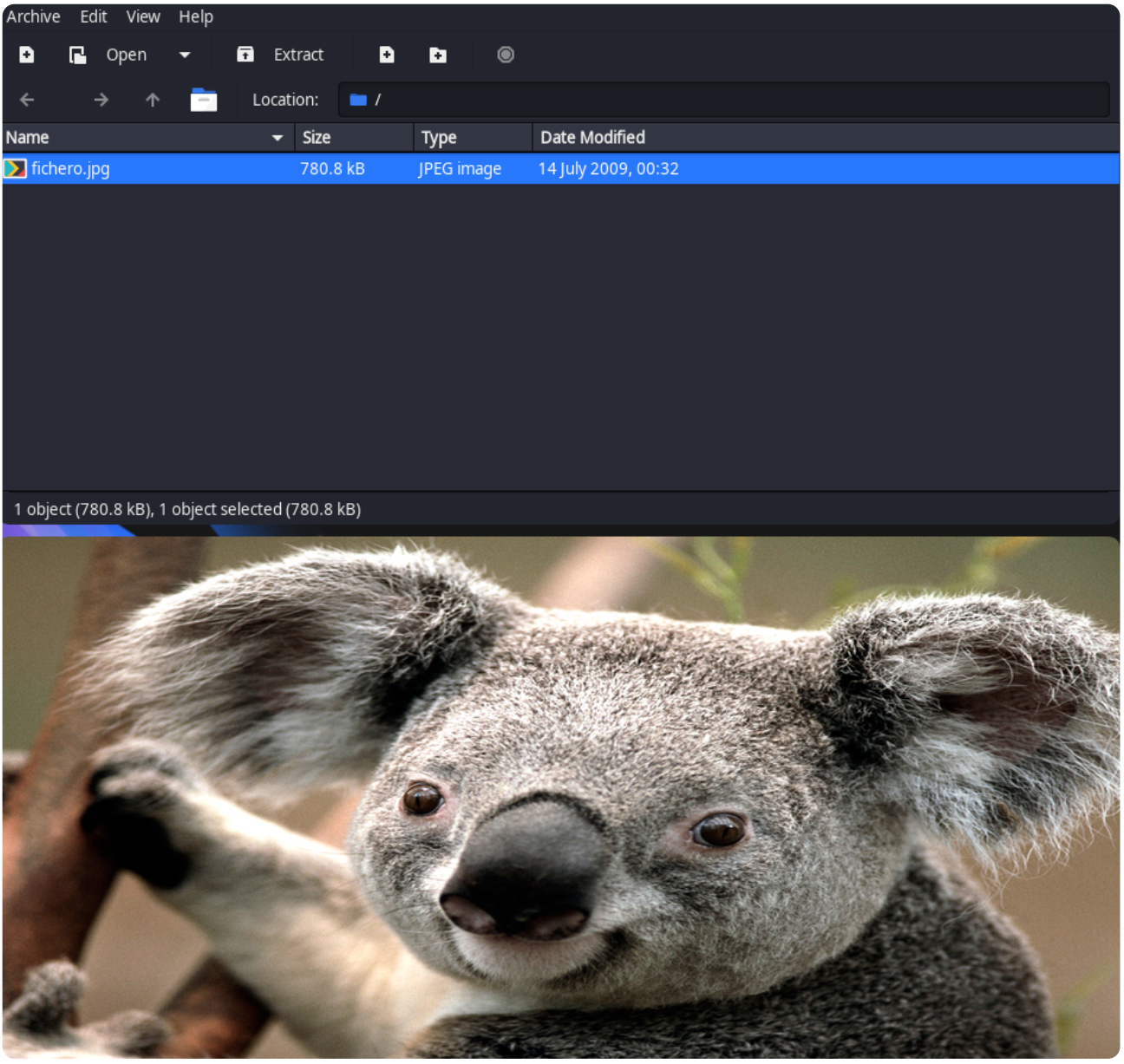


2. **Extracción del fichero ZIP:**
   - Una vez identificada la dirección de memoria (por ejemplo, `0x000000007f52e070`), se procedió a extraer el archivo con:
     - `python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 dumpfiles -Q 0x000000007f52e070 -D /home/kali/Downloads/Forense/Caso1/`
   - 1. **Captura del proceso de extracción:**

```
python2 vol.py -f /home/kali/Downloads/Forense/Caso1/caso1Volatility.dmp --profile=Win7SP1x64 dumpfiles -Q 0x000000007f52e070 -D /home/kali/Downloads/Forense/Caso1/
Volatility Foundation Volatility Framework 2.6.1
```

3. **Análisis del contenido:**
   Al explorar el contenido del ZIP mediante el File Manager, se encontró una imagen que muestra un **koala**.

- **Captura de la imagen interna:**



Análisis:

- La presencia de un fichero ZIP en la memoria RAM indica la posible utilización de archivos comprimidos para ocultar o transportar datos.
- La imagen encontrada dentro del ZIP, que ilustra un koala, añade un elemento distintivo y comprobable al análisis.

Conclusión:

Dentro del fichero ZIP accesible en la memoria RAM se encontró la imagen de un **koala**.

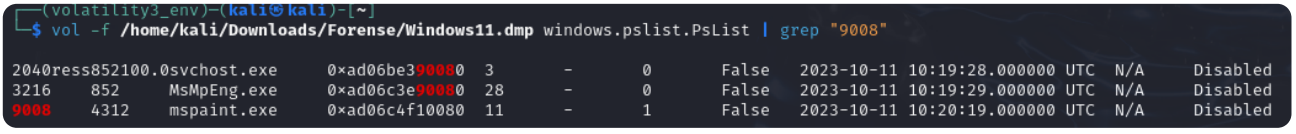
Caso 2: Análisis de la imagen "Windows11.dmp"

1. Proceso de Microsoft Paint: PID y proceso padre

Procedimiento:

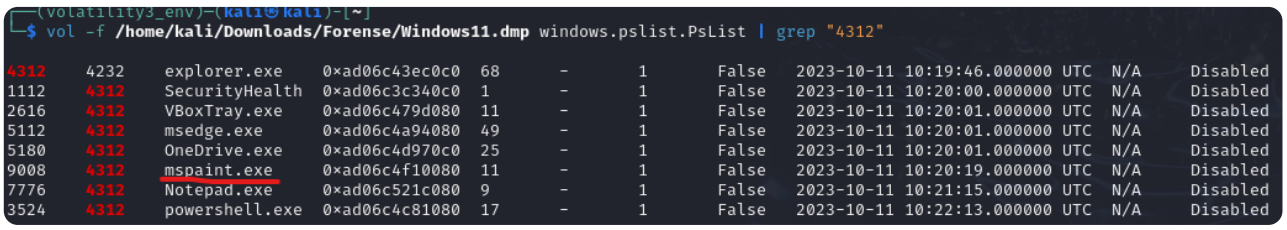
1. **Identificación del proceso Microsoft Paint:**
   - Se utilizó el comando:
     - `vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.pslist.PsList | grep -i "mspaint"`
   - **Captura:**

   - Se identificó que el proceso **mspaint.exe** tiene:
     - **PID:** `9008`
2. **Determinación del proceso padre (PPID):**
   - Para conocer el PPID se filtró nuevamente con:
     - `vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.pslist.PsList | grep "9008"`
   - **Captura:**



   - Se determinó que el **PPID** es `4312`.

- **Verificación del proceso padre:**
  - Se verificó el proceso con:
    - `vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.pslist.PsList | grep "4312"`
  - **Captura:**



Análisis:

- La trazabilidad de procesos es esencial para reconstruir la jerarquía y determinar la legitimidad de las ejecuciones.
- La correlación entre el PID de **mspaint.exe** y su proceso padre permite determinar la cadena de invocación en el sistema.

Conclusión:

- **PID de mspaint.exe:** `9008`
- **PPID (proceso padre):** `4312`

2. Recuperación de la contraseña del usuario "andres"

Procedimiento:

2. Recuperación de la contraseña del usuario "andres"

Procedimiento:

1. **Extracción de hashes del sistema:**
   - Se ejecutó el comando siguiente para extraer los hashes de los usuarios registrados en el sistema a partir de la imagen forense:
     - `vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.hashdump`
   - La salida fue la siguiente:
     -

```
┌──(volatility3_env)─(kali㉿kali)-[~]
└─$ vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.hashdump

Volatility 3 Framework 2.11.0
Progress:  100.00               PDB scanning finished
User    rid     lmhash  nthash

Administrador   500     aad3b435b51404eeaad3b435b51404ee        31d6cfe0d16ae931b73c59d7e0c089c0
Invitado        501     aad3b435b51404eeaad3b435b51404ee        31d6cfe0d16ae931b73c59d7e0c089c0
DefaultAccount  503     aad3b435b51404eeaad3b435b51404ee        31d6cfe0d16ae931b73c59d7e0c089c0
WDAGUtilityAccount      504     aad3b435b51404eeaad3b435b51404ee        de9ed8fa3dc3dc489a568bc472203840
andres  1001    aad3b435b51404eeaad3b435b51404ee        3ec585243c919f4217175e1918e07780
```

2. **Descifrado del hash:**
   - Se copió el hash obtenido (en este caso, el `nthash` para el usuario "andres": `3ec585243c919f4217175e1918e07780`) y se introdujo en el servicio online CrackStation.

     El proceso de descifrado reveló que la contraseña correspondiente es: abc123

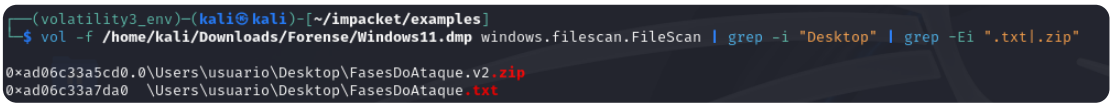
Hash	Type	Result
3ec585243c919f4217175e1918e07780	NTLM	abc123.

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

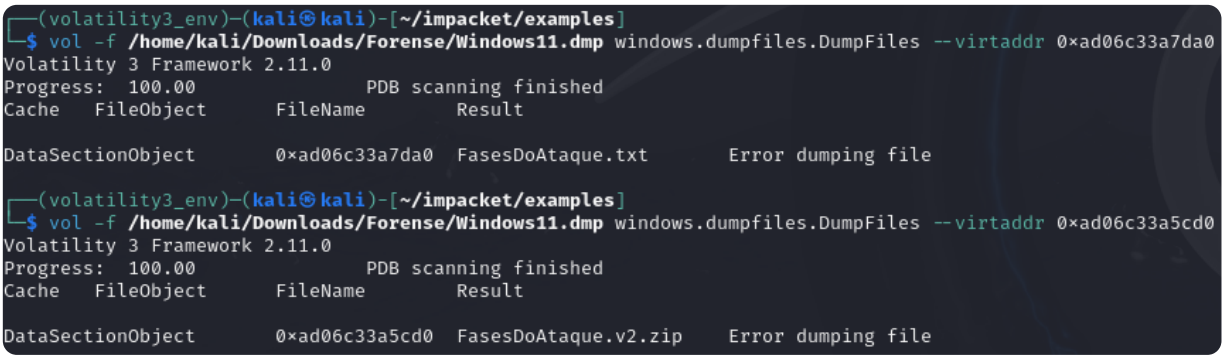
3. Identificación del barrio del cómplice en el fichero de instrucciones

Procedimiento:

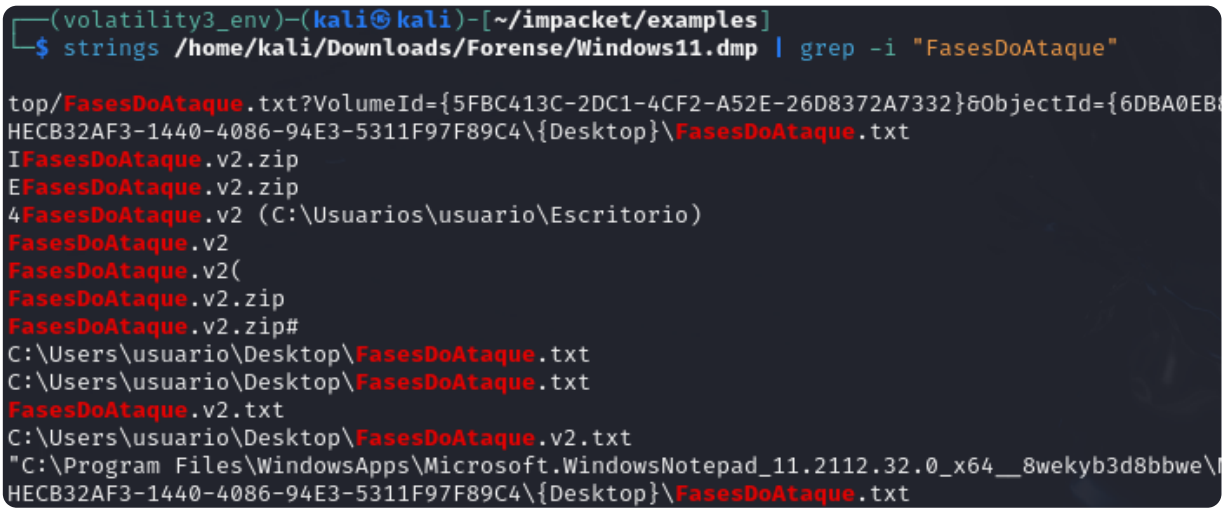
1. **Búsqueda en el Escritorio:**
   - Se realizó un escaneo en la imagen de memoria para localizar el fichero de texto ubicado en el Escritorio del usuario, específicamente buscando "FaseDosAtaque.txt":
     - `vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.filescan.FileScan | grep -i "Desktop" | grep -Ei ".txt|.zip"`
     - **Captura:**



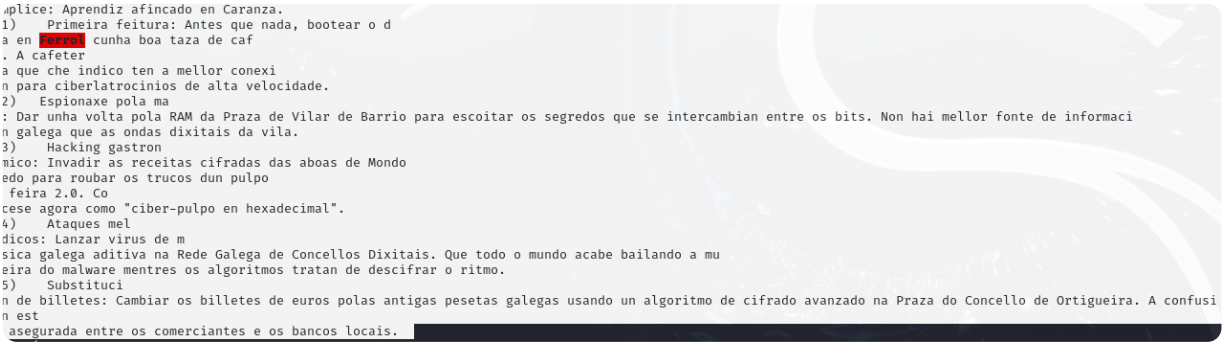
- **Intento de recuperación del fichero:**
  - Se intentó recuperar el archivo y el ZIP contenedor usando:
    - `vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.dumpfiles.DumpFiles --virtaddr 0xad06c33a7da0`
    - `vol -f /home/kali/Downloads/Forense/Windows11.dmp windows.dumpfiles.DumpFiles --virtaddr 0xad06c33a5cd0`
    - **Captura:**



- **Extracción directa desde la memoria:**
  - Ante la corrupción de los ficheros, se optó por buscar referencias textuales en la memoria:
    - strings /home/kali/Downloads/Forense/Windows11.dmp | grep -i "FasesDoAtaque"
    - **Captura:**



- **Lectura ampliada para contexto:**
  - Se empleó el comando:
    - strings /home/kali/Downloads/Forense/Windows11.dmp | grep -i -A50 -B50 "Ferrol"1.
    - **Capturas:**



Análisis:

- El análisis textual permitió identificar menciones específicas al barrio **Ferrol**.
- La frase **"Aprendiz afincado en Caranza"** fue extraída, lo que indica que el cómplice se encuentra en el barrio **Caranza** de Ferrol.

Conclusión:

El archivo de instrucciones revela que el cómplice del organizador es originario del barrio **Caranza** en Ferrol.

4. Descifrado del fichero ZIP cifrado con la última versión del plan

Procedimiento:

1. **Volcado del fichero ZIP desde la imagen forense:**
   - Se realizó un volcado directo del archivo ZIP desde la imagen de memoria utilizando la dirección virtual correspondiente. El comando empleado fue:
     - vol -f /home/kali/Downloads/Forense/Windows11.dmp -o . dumpfiles --virtaddr 0xad06c33a5cd0
     -

```
┌──(volatility3_env)─(kali㉿kali)-[~]
└─$ vol -f /home/kali/Downloads/Forense/Windows11.dmp -o . dumpfiles --virtaddr 0xad06c33a5cd0

Volatility 3 Framework 2.11.0
Progress:  100.00               PDB scanning finished
Cache   FileObject      FileName        Result

DataSectionObject       0xad06c33a5cd0  FasesDoAtaque.v2.zip    Error dumping file
```

   - Esto generó un fichero con nombre complejo:
     - file.0xad06c33a5cd0.0xad06c30884e0.DataSectionObject.FasesDoAtaque.v2.zip.dat
   - Posteriormente, se renombró el archivo para simplificar su uso y establecer la extensión correcta, mediante:
     - mv file.0xad06c33a5cd0.0xad06c30884e0.DataSectionObject.FasesDoAtaque.v2.zip.dat Ataquev2.zip
   - La captura respalda este paso:

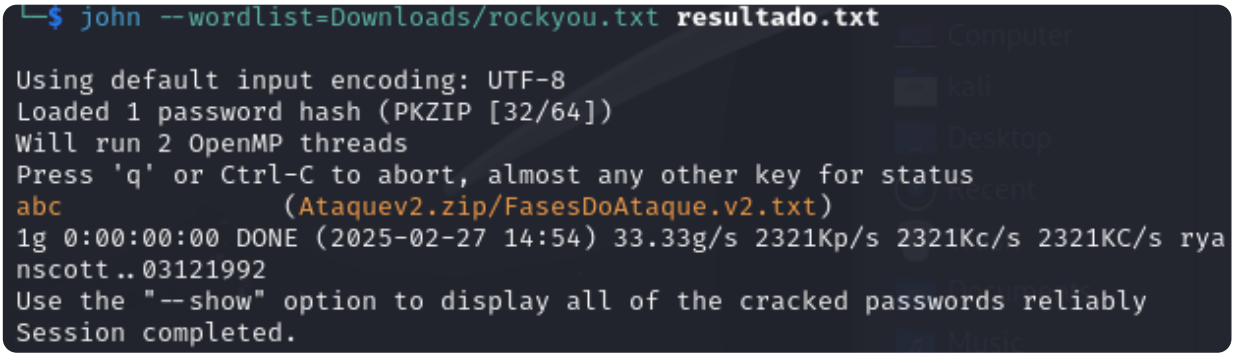
```
┌──(kali㉿kali)-[~]
└─$ mv file.0xad06c33a5cd0.0xad06c30884e0.DataSectionObject.FasesDoAtaque.v2.
zip.dat  Ataquev2.zip
```

2. **Extracción del hash del archivo ZIP con zip2john:**
   - Para proceder con el descifrado, se utilizó la herramienta `zip2john` que permite extraer el hash del fichero comprimido. Se ejecutó el siguiente comando:
     - zip2john Ataquev2.zip > resultado.txt
   - El hash resultante se almacenó en el archivo `resultado.txt`, como se evidencia en la siguiente captura:

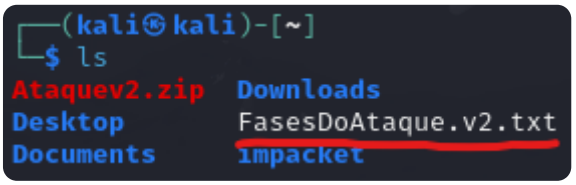
```
└─$ zip2john Ataquev2.zip > resultado.txt

Created directory: /home/kali/.john
ver 2.0 Ataquev2.zip/FasesDoAtaque.v2.txt PKZIP Encr: cmplen=733, decmplen=12
83, crc=8C317172 ts=5595 cs=8c31 type=8
```

3. **Descifrado de la contraseña con John the Ripper:**
   - Con el hash extraído, se procedió a utilizar John the Ripper para recuperar la contraseña. Se empleó el diccionario `rockyou.txt` con el comando:
     - john --wordlist=Downloads/rockyou.txt resultado.txt
   - Tras el procesamiento, se obtuvo la contraseña del ZIP, la cual resultó ser: abc

- La captura confirma este resultado:



- **Descompresión y verificación del contenido del ZIP:**
  - Con la contraseña obtenida, se procedió a descomprimir el archivo para verificar su integridad y contenido. Se utilizó el comando:
  - unzip Ataquev2.zip
  - La operación de descompresión se completó con éxito, lo cual se comprueba en la siguiente captura:



Finalmente, se revisó el contenido extraído, constatando que el fichero contiene la última

versión del plan: