Apéndice

Índice

1. *Ficheros aportados*

Ficheros aportados en la documentación del TFM involucrados en este documento:

- RAM-pslist.txt
- RAM-psxview.txt
- RAM-psscan.txt
- RAM-pstree.txt
- RAM-netscan.txt
- RAM-cscriptEvents[dlllist].txt
- RAM-fichero&regKEY.txt
- RAM-modules&modscan.txt
- HDD-recuperarAjedrez.txt
- HDD-recuperarCscript(exe&exeMUI).txt

Tanto el análisis de la memoria RAM como el análisis y exportación del fichero que resulta ser sospechoso para el análisis, se realizan exclusivamente desde una máquina virtual de Kali Linux como resultado añadido a la investigación del TFM desde la máquina de SIFT (pues no soportaba la versión del perfil de Windows [Win10x64_16299] para el análisis, el cual era requerido por ser el de la máquina atacada).

El análisis de la memoria RAM y de los procesos activos en ella, se llevan a cabo mediante el framework ***Volatility 2.6*** (https://github.com/volatilityfoundation/volatility).

El análisis del disco duro y la recuperación del archivo, se hace mediante el toolkit forense de "***The Sleuth Kit***" (https://www.sleuthkit.org/).

Ambos vienen ya instalados de serie en Kali Linux, solo se debe actualizar los repositorios y el sistema para tener la versión más actualizada.

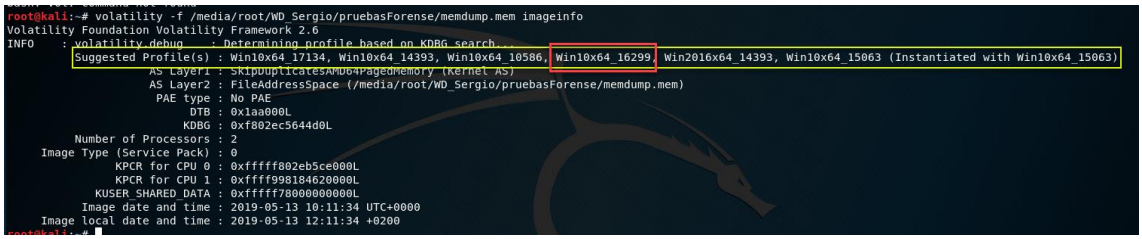
2. *RAM*

2.1. imageinfo

Retorna el perfil sugerido para el parámetro "***--profile=<perfil>***", el cual usarán los plugins automáticamente cuando lo necesiten.

Para lanzar la aplicación, seguir la ruta de arriba o abrir una terminal, y usar el comando:

- "***Volatility -f <ruta/memdump.mem> imageinfo***"
  - ***-f***: indica de que fichero se hará el análisis de la memoria
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.



RAM 1 (volatility - imageinfo)

➔ ***Ver la imagen más detalladamente:*** (ver)

➔ El retorno del comando propone 6 posibles perfiles para usar:

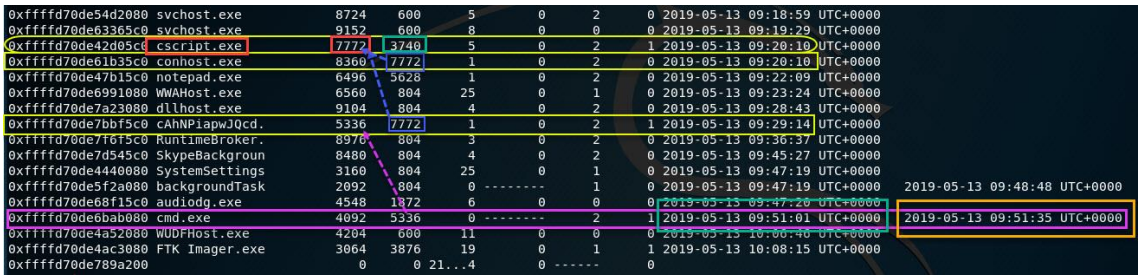
- *Win10x64_17134*
- *Win10x64_14393*
- *Win10x64_10586*
- *Win10x64_16299 (**el que se usará**)*
- *Win2016x64_14393*
- *Win10x64_15063 (el que recomienda)*

2.2. pslist

Se buscan los procesos activos en el momento del incidente de seguridad mediante el plugin ***pslist*** para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> pslist --profile=<perfil>***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***pslist:*** invoca al plugin para ver la lista de procesos del sistema.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.



RAM 2 (volatility - pslist)

➔ El fichero conteniendo el comando ejecutado y todo el contenido retornado por el plugin ***pslist.*** (RAM-pslist.txt).

➔ Del contenido del fichero se sacan los resultados más relevantes, se analiza ***cscript.exe***, perteneciente al ***PID=7772***, con PPID=3740, y ejecutado a las 09:20:10; del cual cuelgan otros 2 procesos:

- ***conhost.exe***: Con un ***PID=8360*** > PPID=7772, y ejecutado a las 09:20:10
- ***cAhNPiapwJQcd.exe***: Con ***PID=5336*** > PPID=7772 y ejecutado a las 09:29:14. Del proceso cuelga otro:
  - ***cmd.exe:*** Con ***PID=4092*** > PPID=5336, ejecutado a las 09:51:01, y finalizado a las 09:51:35. Esto ha dado acceso a una terminal de comandos.

2.3. psxview

Se corre el plugin de ***psxview*** para comprobar si estos procesos están en ***pslist*** y en ***psscan***, comprobando si era cierto el resultado obtenido en el uso de ***pslist***, y antes de proceder al análisis con ***psscan***.

Se buscan los procesos activos en el momento del incidente de seguridad mediante el plugin ***psxview*** para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> --profile=<perfil> psxview***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.
  - ***psxview:*** invoca al plugin para ver la lista de procesos del sistema.

Offset(P)	Name	PID	pslist	psscan	thrdproc	pspcid	csrss	session	deskthrd	ExitTime
0x000000003bed45c0	VBoxTray.exe	5412	True	True	True	False	True	True	False	
0x000000000df32080	fontdrvhost.ex	720	True	True	True	False	True	True	False	
0x000000010de435c0	RuntimeBroker.	7308	True	True	True	False	True	True	False	
0x000000000dc245c0	svchost.exe	804	True	True	True	False	True	True	True	
0x000000000250665c0	svchost.exe	2344	True	True	True	False	True	True	True	
0x00000000343d4080	taskhostw.exe	3820	True	True	True	False	True	True	False	
0x00000000b8c4b5c0	cAhNPiapwJQcd.	5336	True	True	True	False	True	True	False	
0x00000001010bf5c0	explorer.exe	5628	True	True	True	False	True	True	False	
0x00000001002d05c0	cscript.exe	7772	True	True	True	False	True	True	False	
0x00000000b1580080	dllhost.exe	9104	True	True	True	False	True	True	False	
0x000000001ea695c0	SearchIndexer.	5084	True	True	True	False	True	True	True	
0x000000005dc85080	SystemSettings	3160	True	True	True	False	True	True	False	
0x0000000107fb35c0	conhost.exe	8360	True	True	True	False	True	True	False	
0x000000011cad4040	System	4	True	True	True	False	False	False	False	
0x0000000107ee5080	smss.exe	464	True	True	False	False	False	True	False	2019-05-13 09:12:46 UTC+0000
0x00000000111ff080	cmd.exe	4092	True	True	False	False	False	True	False	2019-05-13 09:51:35 UTC+0000
0x00000001010715c0	userinit.exe	4604	True	True	False	False	False	True	False	2019-05-13 09:18:58 UTC+0000
0x00000000006470f080	backgroundTask	7564	False	False	False	False	False	True	False	2019-05-13 10:11:55 UTC+0000

RAM 3 (volatility - psxview)

➔ En la captura se han recortado resultados para poder mostrar todos los procesos remarcados en el paso anterior en 1 sola foto, (para ver todo lo retornado acceder al fichero).

- ***Azul:*** cscript.exe → PID=7772
  - ***Rojo:*** *conhost.exe → PID=8360; cAhNPiapwJQcd.exe → PID=5336*
    - ***Verde:*** cmd.exe → PID=4092

➔ El fichero conteniendo el comando ejecutado y todo el contenido retornado por el plugin ***psxview*** → (ver RAM-psxview.txt).

2.4. psscan

Se corre el plugin de ***psscan*** para comprobar si estos procesos están.

Se buscan los procesos mediante el plugin ***psscan*** para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> psscan --profile=<perfil>***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***psscan:*** invoca al plugin para ver la lista de procesos del sistema.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.

```
root@kali:~# volatility -f /media/root/WD_Sergio/pruebasForense/memdump.mem psscan --profile=Win10x64_16299
Volatility Foundation Volatility Framework 2.6
Offset(P)          Name                PID   PPID PDB               Time created                   Time exited
------------------ ---------------- ------ ------ ------------------ ------------------------------ ------------------------------
0x0000d70de3ed4040 System                4      0 0x00000000001aa000 2019-05-13 09:12:28 UTC+0000
0x0000d70de3f15080 svchost.exe        1588    600 0x000000000151c7000 2019-05-13 09:12:47 UTC+0000
0x0000d70de3f25080 svchost.exe        1484    600 0x0000000014988000 2019-05-13 09:12:47 UTC+0000
0x0000d70de3f2c080 VBoxService.ex     1564    600 0x0000000014e91000 2019-05-13 09:12:47 UTC+0000
0x0000d70de3f3a080 svchost.exe        1540    600 0x000000000ca8f000 2019-05-13 09:12:47 UTC+0000
0x0000d70de42ad5c0 dllhost.exe        4776    804 0x0000000101d3f000 2019-05-13 09:18:23 UTC+0000
0x0000d70de42d05c0 cscript.exe        7772   3740 0x0000000066ee4000 2019-05-13 09:20:10 UTC+0000
0x0000d70de42dc080 svchost.exe        5228    600 0x000000010ea6f000 2019-05-13 09:15:35 UTC+0000
0x0000d70de4440080 SystemSettings     3160    804 0x000000000569b7000 2019-05-13 09:47:19 UTC+0000
0x0000d70de609c5c0 ShellExperienc     1624    804 0x0000000005107f000 2019-05-13 09:18:38 UTC+0000
0x0000d70de60e5080 smss.exe            464    304 0x000000000c1fe000 2019-05-13 09:12:46 UTC+0000   2019-05-13 09:12:46 UTC+0000
0x0000d70de61b35c0 conhost.exe        8360   7772 0x0000000007154e000 2019-05-13 09:20:10 UTC+0000
0x0000d70de6b755c0 svchost.exe        1084    600 0x0000000011200000 2019-05-13 09:12:47 UTC+0000
0x0000d70de6b91040 MemCompression     1728      4 0x000000015a34000 2019-05-13 09:12:48 UTC+0000
0x0000d70de6bab080 cmd.exe            4092   5336 0x0000000083267000 2019-05-13 09:51:01 UTC+0000   2019-05-13 09:51:35 UTC+0000
0x0000d70de6bb55c0 svchost.exe        1160    600 0x0000000001134f000 2019-05-13 09:12:47 UTC+0000
0x0000d70de6bb95c0 svchost.exe        1208    600 0x0000000011a65000 2019-05-13 09:12:47 UTC+0000
0x0000d70de7a23080 dllhost.exe        9104    804 0x00000000bf552000 2019-05-13 09:28:43 UTC+0000
0x0000d70de7bbf5c0 cAhNPiapwJQcd.     5336   7772 0x000000005e905000 2019-05-13 09:29:14 UTC+0000
0x0000d70de7d545c0 SkypeBackgroun     8480    804 0x0000000059fd9000 2019-05-13 09:45:27 UTC+0000
0x0000d70de7f6f5c0 RuntimeBroker.     8976    804 0x00000000ba785000 2019-05-13 09:36:37 UTC+0000
0x0000fb80000d4040 System                4      0 0x00000000001aa000 2019-05-13 09:12:28 UTC+0000
0x0000fb8000115080 svchost.exe        1588    600 0x000000000151c7000 2019-05-13 09:12:47 UTC+0000
```

RAM 4 (volatility - psscan)

➔ En la captura se han recortado resultados para poder mostrar todos los procesos remarcados en el paso anterior en 1 sola foto, (para ver todo lo retornado acceder al fichero).

- ***Azul:*** cscript.exe → PID=7772
  - ***Rojo:*** *conhost.exe → PID=8360; cAhNPiapwJQcd.exe → PID=5336*
    - ***Verde:*** cmd.exe → PID=4092
    -

➔ El fichero conteniendo el comando ejecutado y todo el contenido retornado por el plugin ***psscan.*** → (RAM-psscan.txt).

2.5. pstree

Se corre el plugin de ***pstree*** para pintar la lista de procesos que cuelgan de cada uno de los procesos involucrados.

Con ello, se obtendrá una visión más gráfica de las relaciones encontradas y expuestas en los pasos previos.

Se buscan los procesos mediante el plugin ***pstree*** para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> pstree --profile=<perfil>***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***pstree:*** invoca al plugin para ver la lista de procesos del sistema.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.

```
Name                                                 Pid   PPid   Thds   Hnds Time
---------------------------------------------------- ------ ------ ------ ------ ----
WARNING : volatility.debug    : PID 4 PPID 0 has already been seen
 0xffffd70de42d05c0:cscript.exe                      7772   3740      5      0 2019-05-13 09:20:10 UTC+0000
. 0xffffd70de61b35c0:conhost.exe                     8360   7772      1      0 2019-05-13 09:20:10 UTC+0000
. 0xffffd70de7bbf5c0:cAhNPiapwJQcd.                  5336   7772      1      0 2019-05-13 09:29:14 UTC+0000
.. 0xffffd70de6bab080:cmd.exe                        4092   5336      0 ------ 2019-05-13 09:51:01 UTC+0000
 0xffffd70de76f9080:GoogleCrashHan                   5264   3620      3      0 2019-05-13 09:13:22 UTC+0000
 0xffffd70de7704380:GoogleCrashHan                   5288   3620      3      0 2019-05-13 09:13:23 UTC+0000
```

RAM 5 (volatility - pstree)

➔ En la captura inferior se han recortado los demás resultados para mostrar sólo los más importantes, los pertenecientes a los procesos involucrados en la investigación.

➔ El fichero conteniendo el comando ejecutado y todo el contenido retornado por el plugin ***pslist.*** → (RAM-pstree.txt).

2.6. netscan

Se corre el plugin de ***netscan*** en busca de conexiones que sean generadas por parte de procesos que se están investigando (PID=3740,7772,5336,8360,4092) para "***- -profile=Win10x64_16299***". Este plugin es similar a la herramienta netstat en Windows, pero no admite parámetros.

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> --profile=<perfil> netscan***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.
  - ***netscan:*** invoca al plugin para ver la lista de procesos del sistema.

```
root@kali:~# volatility -f /media/root/WD_Sergio/pruebasForense/memdump.mem --profile=Win10x64_16299 netscan
Volatility Foundation Volatility Framework 2.6

Offset(P)          Proto    Local Address                  Foreign Address      State            Pid      Owner          Created
**********************************************************************************************************************************
0xd70de3e4dc20     UDPv4    -:1900                         *:*                                   3836     svchost.exe    2019-05-13 10:13:25 UTC+0000
0xd70de42f1ec0     UDPv4    0.0.0.0:0                      *:*                                   5212     svchost.exe    2019-05-13 09:15:35 UTC+0000
0xd70de4325ec0     UDPv4    0.0.0.0:0                      *:*                                   1408     svchost.exe    2019-05-13 10:13:30 UTC+0000
0xd70de4325ec0     UDPv6    :::0                           *:*                                   1408     svchost.exe    2019-05-13 10:13:30 UTC+0000
0xd70de4328410     UDPv4    0.0.0.0:500                    *:*                                   5228     svchost.exe    2019-05-13 09:15:35 UTC+0000
0xd70de432b010     UDPv4    0.0.0.0:0                      *:*                                   5228     svchost.exe    2019-05-13 09:15:35 UTC+0000
0xd70de432b010     UDPv6    :::0                           *:*                                   5228     svchost.exe    2019-05-13 09:15:35 UTC+0000
0xd70de432b410     UDPv4    0.0.0.0:0                      *:*                                   5212     svchost.exe    2019-05-13 09:15:35 UTC+0000
0xd70de432b410     UDPv6    :::0                           *:*                                   5212     svchost.exe    2019-05-13 09:15:35 UTC+0000
0xd70de42fbba0     TCPv4    0.0.0.0:49688                  0.0.0.0:0            LISTENING        5212     svchost.exe    2019-05-13 09:15:35 UTC+0000
0xd70de428acc0     TCPv4    10.0.27.8:49700                40.67.254.36:443     CLOSED           2492     svchost.exe
0xd70de4491a90     TCPv4    10.0.27.8:50183                10.0.27.6:37723      ESTABLISHED      5336     cAhNPiapwJQcd.
0xd70de455f730     TCPv4    10.0.27.8:49990                93.184.220.29:80     CLOSE_WAIT       6560     WWAHost.exe
0xd70de4658cc0     TCPv4    10.0.27.8:50333                40.77.226.250:443    CLOSED           2464     svchost.exe
0xd70de46ecb50     TCPv4    10.0.27.8:49994                23.10.74.192:443     CLOSE_WAIT       6560     WWAHost.exe
0xd70de4a01ac0     UDPv4    0.0.0.0:0                      *:*                                   5228     svchost.exe    2019-05-13 09:15:35 UTC+0000
```

RAM 6 (volatility - netscan)

➔ De la captura, se encuentra una conexión con el ***PID=5336 establecida***, a la dirección ***10.0.27.6:37723*** mediante ***TCPv4***, para el que el propietario es ***cAhNPiapwJQcd.exe***

➔ El fichero conteniendo el comando ejecutado y todo el contenido retornado por el plugin ***netscan.*** → (RAM-netscan.txt).

→Analizados los procesos y las conexiones activas en el momento del volcado de la memoria RAM en el fichero memdump.mem que se está analizando. Antes de analizar las dll que carga cada proceso, los ficheros a los que tiene acceso, y los registros de clave de cada uno de ellos, se pueden trazar los siguientes eventos:

- Arranca un ***PID=3740 no traceable***, el cual es el padre del PID=7772.
  - cscript.exe se ejecuta con un PID=7772. De él, se ejecutan:
    - conhost.exe con un PID=8360
    - cAhNPiapwJQcd.exe con un PID=5336; propietario de una conexión establecida por TCPv4 a la dirección 10.0.27.6:37723; y del cual se ejecuta:
      - cmd.exe con un PID=4092 con PPID=5336. Hace pensar que se ejecutan comando remotamente en el equipo infectado.

2.7. dlllist

Se corre el plugin de ***dlllist*** para comprobar la lista de dll que carga cada uno de los procesos involucrados.

Cuando en los resultados obtenidos del comando se visualice en el campo "***LoadCount=0xffff***" (remarcado en morado claro en las imágenes), significa que la ***DLL*** estaba ***presente*** en la ***Import Address Table*** (*IAT*) del ejecutable, por lo que fueron cargadas desde el inicio.

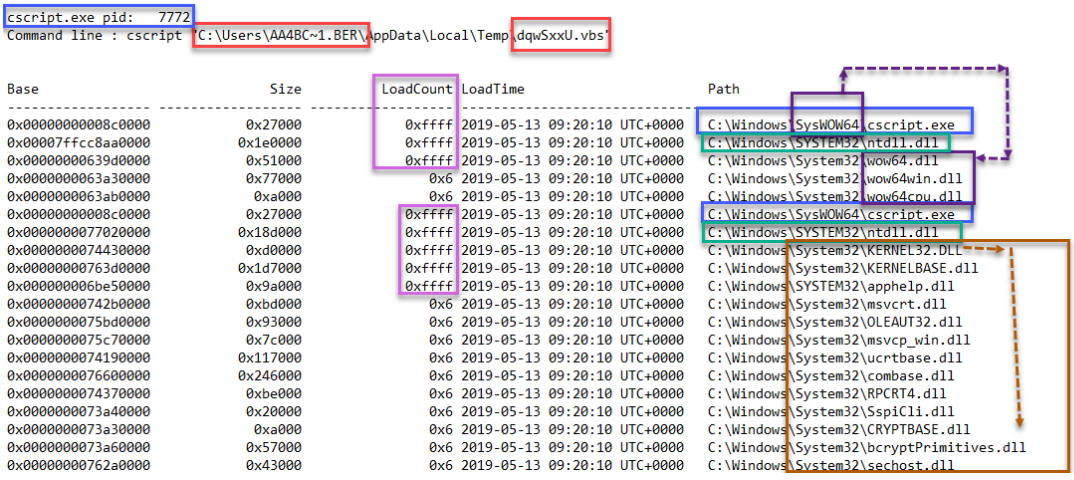
Es normal que procesos importen librerías de otros procesos, pero no es normal, como ejemplo, que un proceso tipo *svchost.exe* cargue funciones de red exportadas por ***wininet.dll***, ***dnsapi.dll***, o ***mswsock.dll (****WS2_32.dll, wsock.dll o wsock32.dll****)*; entre otros.

Por ello, se realiza la búsqueda con este plugin, para comprobar si el proceso cargó alguna DLL que no debiese.

Se comprobará para la lista de procesos, de forma individual, que se están investigando (PID=3740,7772,5336,8360,4092) para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> --profile=<perfil> dlllist -p <pid>***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria.
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.
  - ***dlllist:*** invoca al plugin para ver la lista de procesos del sistema.
  - ***-p:*** indica que se pasará un PID como parámetro.
  - ***<pid>:*** número del proceso que se comprueban las DLL que usa.



```
cscript.exe pid:   7772
Command line : cscript "C:\Users\AA4BC~1.BER\AppData\Local\Temp\dqwSxxU.vbs"

Base                             Size         LoadCount LoadTime                      Path
-------------------- ---------- ---------- ------------------------------ ----
0x00000000008c0000         0x27000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\SysWOW64\cscript.exe
0x00007ffcc8aa0000          0x1e0000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\SYSTEM32\ntdll.dll
0x00000000639d0000         0x51000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\wow64.dll
0x0000000063a30000         0x77000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\wow64win.dll
0x0000000063ab0000          0xa000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\wow64cpu.dll
0x00000000008c0000         0x27000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\SysWOW64\cscript.exe
0x0000000077020000         0x18d000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\SYSTEM32\ntdll.dll
0x0000000074430000         0xd0000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\KERNEL32.DLL
0x000000000763d0000         0x1d7000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\KERNELBASE.dll
0x000000006be50000          0x9a000          0xffff 2019-05-13 09:20:10 UTC+0000   C:\Windows\SYSTEM32\apphelp.dll
0x000000000742b0000         0xbd000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\msvcrt.dll
0x0000000075bd0000         0x93000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\OLEAUT32.dll
0x0000000075c70000          0x7c000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\msvcp_win.dll
0x0000000074190000         0x117000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\ucrtbase.dll
0x0000000076600000         0x246000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\combase.dll
0x0000000074370000          0xbe000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\RPCRT4.dll
0x0000000073a40000          0x20000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\SspiCli.dll
0x0000000073a30000           0xa000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\CRYPTBASE.dll
0x0000000073a60000          0x57000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\bcryptPrimitives.dll
0x000000000762a0000          0x43000             0x6 2019-05-13 09:20:10 UTC+0000   C:\Windows\System32\sechost.dll
```

RAM 7 (volatility – dlllist – PID=7772)

➔ ver imagen dlllist7772

➔ El fichero conteniendo el comando ejecutado con cada proceso relevante en este hilo de investigación y todo el contenido retornado por el plugin ***dlllist.***

- (RAM-cscriptEvents[dlllist].txt).

➔ En la imagen superior, llama la atención que el programa al que pertenece el ***PID=7772*** con el nombre ***cscript.exe***, se lanza por línea de comandos desde la carpeta de usuario, desde el usuario "***AA4BC~1.BER***", dando a entender que *está ofuscado el nombre del usuario desde el que se lanza* (en el TFM aparece 1 usuario con el nombre '***a.bernal***', pudiendo referirse a este), en una ubicación temporal, y con un nombre distinto ***cscript.exe***, siendo este ***dqwSxxU.vbs*** (síntoma de ofuscación).

- Es relevante que tenga extensión del lenguaje Visual Basic, pues suele ser usado para el desarrollo de malware.

➔ Que la primera acción sea ejecutar ***cscript.exe*** desde la ruta '***C:\Windows\SysWOW64***' es indicativo del comienzo de una más que posible acción maliciosa:

- ***SysWOW64*** es una carpeta en sistemas Windows de 64-bits en la cual se contienen las librerías de 32-bits.
  - El nombre WOW64 hace referencia a que se permite ejecutar software de 32-bits en un sistema de 64-bits, ya que redirige el acceso a los archivos para asegurar que los programas funcionan correctamente. (https://www.softzone.es/2018/02/16/system32-syswow64-diferencias-windows/).
- ***WOW64*** se implementa usando 3 DLL:
  - ***Wow64.dll*** : Interfaz principal al núcleo NT que realiza la traducción entre las llamadas de 32-bits y 64-bits, incluyendo las manipulaciones del puntero y la pila.
    - *Posible causa de la no detección del PID=3740, el cual es padre del PID=7772 que se está analizando.*
  - ***Wow64win.dll*** : Proporciona los puntos de entrada apropiados para las aplicaciones de 32-bits.
  - ***Wow64cpu.dll*** : Cambia el modo del procesador entre 32-bits y 64-bits.
- ***Registro y sistema de ficheros:*** Cuando se ejecuta una aplicación de 32-bits, WOW64 redirige las peticiones de DLL al directorio "***%SystemRoot%\sysWOW64***"; contenedor de las bibliotecas y ejecutables heredados.
  - ***Importante!!!*** : Algunos lenguajes como Visual Basic (muy usado junto a C# en el desarrollo de malware), puede realizar peticiones a system32 que responde syswow64; pero si actualiza un archivo, puede ser copiado en system32, provocando acciones ilógicas.

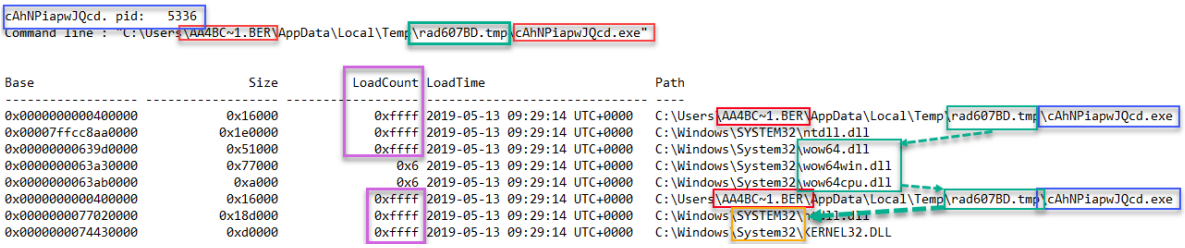
➔ Con esta información, tras la captura anterior, llama la atención que un proceso llamado ***cscript.exe*** con PID=7772, se ejecute desde el usuario "***AA4BC~1.BER***" con un ***path temporal*** y el fichero ***dqwSxxU.vbs;*** que se ejecute una llamada a "***vbscritp.dll***" desde "System32", pero antes, realiza las siguientes acciones:

- Primero se ejecuta desde la IAT el fichero ***wow64.dll*** (el cual puede ser causante de que el *PID=3740* (*PPID de PID=7772*) no retorne información), el cual ***hace llamadas a wow64win.dll y wow64cpu.dll***
- Retorna a la ***IAT*** y llama a ***ntdll.dll*** (exporta Windows Native API [https://es.wikipedia.org/wiki/Anexo:Bibliotecas_DLL_de_Windows]), la cual recibe llamadas como API Nativa de:
  - ***KERNEL32.dll***: expone a las aplicaciones la mayoría de las APIs base de Win32, como la administración de memoria, operaciones de

entrada/salida, creación de procesos e hilos, y funciones de sincronización.

- ***KERNELBASE.dll***: Contiene un conjunto de procedimientos y funciones de los drivers, por lo que es esencial para la función de forma normal del sistema. (https://www.drivereasy.com/knowledge/fix-kernelbase-dll-crash-issue/).
- ***msvcrt.dll***: Primera llamada fuera de la IAT. Biblioteca para el compilador Visual C++ (MSVC), poseedora de programas compilados en curso con la mayoría de las funciones de la biblioteca estándar de C; las cuales incluyen la manipulación de cadenas de texto, asignación de memoria, llamadas de entrada y salida al estilo de C, etc.
- ***oleaut32.dll:*** OLE Automation permite que las aplicaciones manejen archivos e información creada por otras aplicaciones, habilitando el proceso.
- ***msvcp_win.dll***: Biblioteca para C++
- ***ucrtbase.***dll: Biblioteca asociada a Microsoft Visual C++. (https://www.file.net/process/ucrtbase.dll.html)
- ***librerías de representación gráfica:***
  - ***GDI32.dll:*** Funciones primitivas de dibujo para la salida de video, siendo llamada para aprovechar las funciones de bajo nivel, salida de texto, administración de fuentes, etc.
  - ***Gdi32full.dll***
  - ***USER32.dll***: Crea y manipula los elementos estándares de la interfaz de usuario de Windows, permitiendo a los programas crear GUI (para referirse a interfaz gráfica a partir de ahora) que coinciden con la apariencia visual de Windows. Esto le permite completar operaciones como recibir mensajes de ventana (como eventos de ratón o notificaciones del sistema operativo), mostrar texto en la ventana, etc.
- ***ADVAPI32.dll***: Provee de llamadas y funciones seguras para el manejo de registro.

La siguiente captura muestra la "***command line***" desde la que se lanza ***dlllist*** con ***PID=5336*** para "***- -profile=Win10x64_16299***".



RAM 8 (volatility – dlllist – PID=5336)

➔ ver imagen dlllist5336

➔ El fichero conteniendo el comando ejecutado con cada proceso relevante en este hilo de investigación y todo el contenido retornado por el plugin ***dlllist.***

- (RAM-cscriptEvents[dlllist].txt).

➔ En la imagen superior, llama la atención que el programa al que pertenece el ***PID=5336*** con el nombre ***cAhNPiapwJQcd.exe***, se lanza por línea de comandos desde la carpeta de usuario, desde el usuario "***AA4BC~1.BER***", dando a entender que *está ofuscado el nombre del usuario desde el que se lanza* (en el TFM aparece 1 usuario con el nombre '***a.bernal***', pudiendo referirse a este), en una ubicación temporal (seguramente en el entorno montado por el proceso anteriormente analizado), y con un nombre distinto ***cAhNPiapwJQcd.exe***.

- Sigue los pasos explicados más arriba para ejecutar el proceso (*como si estuviese conectándose al entrono montado por el PID=7772*), pero no llama la atención que ***se encuentren 3 llamadas a funciones para crear un socket*** y abrir una comunicación (***WS2_32.dll, WSOCK32.dll, y mswsock.dll***) pues es el propietario de la comunicación establecida por TCPv4 a 10.0.27.6:37723.

```
0x0000000076360000          0x66000              0x6 2019-05-13 09:29:14 UTC+0000   C:\Windows\System32\WS2_32.dll
0x0000000072ed0000           0x8000              0x6 2019-05-13 09:29:14 UTC+0000   C:\Windows\SYSTEM32\WSOCK32.dll
0x000000006fc50000          0x55000              0x6 2019-05-13 09:29:14 UTC+0000   C:\Windows\system32\mswsock.dll
```

RAM 9 (volatility – dlllist – PID=5336 – ws2_32.dll, wsock32.dll, mswsock.dll)

- ***NETAPI32.dll:*** Provee funcionalidades para las peticiones y la administración de interfaces de red.

```
0x0000000073900000          0x13000              0x6 2019-05-13 09:29:20 UTC+0000   C:\Windows\SYSTEM32\NETAPI32.dll
```

RAM 10 (volatility – dlllist – PID=5336 – NETAPI32.dll)

- Levanta funciones de cliente DHCP (para IPv4 e IPv6)y de comunicación a nivel de Shell:
  - ***Dhcpcsvc6.dll y dhcpsvc.dll***
  - ***shell32.dll***: controla ciertas funciones API de la Shell de Windows. Es capaz de registrar las entradas por el teclado y el ratón. (https://www.file.net/process/shell32.dll.html)
  - ***shcore.dll***: https://windows10dll.nirsoft.net/shcore_dll.html

```
0x000000006dd00000          0x13000              0x6 2019-05-13 09:29:20 UTC+0000   C:\Windows\SYSTEM32\dhcpcsvc6.DLL
0x0000000073300000          0x14000              0x6 2019-05-13 09:29:20 UTC+0000   C:\Windows\SYSTEM32\dhcpcsvc.DLL
0x000000000746b0000         0x1333000             0x6 2019-05-13 09:33:01 UTC+0000   C:\Windows\System32\shell32.DLL
0x0000000075e20000          0x88000              0x6 2019-05-13 09:33:01 UTC+0000   C:\Windows\System32\shcore.dll
```

RAM 11(volatility – dlllist – PID=5336 – dhcpscv6.dll, dhcpscv.dll, shell32.dll, shcore.dll)

- Finaliza con una llamada a cmd.exe, la cual ejecuta el proceso con PID=4092.

```
0x0000000066d10000          0x11000              0x6 2019-05-13 10:13:06 UTC+0000   \??\C:\Windows\SYSTEM32\cmd.exe
```

RAM 12 (volatility – dlllist – PID=5336 – cmd.exe)

➔ El fichero conteniendo el comando ejecutado con cada proceso relevante en este hilo de investigación y todo el contenido retornado por el plugin ***dlllist.***

- (RAM-cscriptEvents[dlllist].txt).

2.8. Ficheros y claves de registro

Otra parte fundamental del análisis de la memoria RAM, es ver que ficheros y que registros toca el proceso, así, como el tipo de acceso que tiene para dicho fichero o registro.
Esta tarea la lleva a cabo el plugin ***handles***, el cual, mediante el parámetro "-t <tipo>" mas el PID, es capaz de encontrar los ficheros o claves de registro que toca el proceso para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> --profile=<perfil> handles <-t|-k> <file|key> -p <pid>***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria.
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.
  - ***handles:*** invoca al plugin para ver la lista de procesos del sistema.
  - ***<-t|-k>:*** indica que se pasará un tipo como parámetro. T=fichero – K=clave de registro.
  - ***< file|key >:*** Tipo del parámetro anterior.
  - ***-p:*** indica que se pasará un PID como parámetro.
  - ***<pid>:*** número del proceso que se comprueban las DLL que usa.

Primero se muestran los ficheros a los que tienen acceso PID=5336 y PID=7772 en las imágenes inferiores respectivamente.

```
Volatility Foundation Volatility Framework 2.6
Offset(V)          Pid             Handle              Access Type             Details
------------------ ------ ------------------ ------------------ ---------------- -------
0xffffd70de4815ef0   5336               0x3c           0x100020 File            \Device\HarddiskVolume2\Windows
0xffffd70de46ca860   5336               0x7c           0x100020 File            \Device\HarddiskVolume2\Users\a.bernal\Desktop
0xffffd70de4799a70   5336               0xe4           0x100001 File            \Device\CNG
0xffffd70de4eadca0   5336              0x11c           0x16019f File            \Device\Afd
0xffffd70de637d870   5336              0x1dc           0x100003 File            \Device\KsecDD
0xffffd70de7b327c0   5336              0x234           0x120089 File            \Device\DeviceApi\CMApi
0xffffd70de7c41ca0   5336              0x2bc           0x100080 File            \Device\Nsi
0xffffd70de47b8ef0   5336              0x36c           0x120089 File            \Device\NamedPipe\
```

RAM 13 (volatility – handles -t file – PID=5336)

➔ ver imagen handlesFilesPID5336

```
  Pid             Handle              Access Type             Details
----- ------------------ ------------------ ---------------- -------
 7772               0x3c           0x100020 File            \Device\HarddiskVolume2\Windows
 7772               0x7c           0x100020 File            \Device\HarddiskVolume2\Users\a.bernal\Desktop
 7772               0x80           0x12019f File            \Device\ConDrv\Connect
 7772               0x84           0x12019f File            \Device\ConDrv\Reference
 7772               0xe8           0x100001 File            \Device\CNG
 7772              0x16c           0x120089 File            \Device\HarddiskVolume2\Windows\SysWOW64\es-ES\cscript.exe.mui
 7772              0x1b4           0x120089 File            \Device\HarddiskVolume2\Windows\SysWOW64\cscript.exe
 7772              0x27c           0x100003 File            \Device\KsecDD
 7772              0x2c0           0x120089 File            \Device\HarddiskVolume2\Windows\SysWOW64\wshom.ocx
 7772              0x2c8           0x120089 File            \Device\DeviceApi\CMApi
 7772              0x39c           0x120089 File            \Device\HarddiskVolume2\Windows\SysWOW64\es-ES\KernelBase.dll.mui
 7772              0x3d0           0x120089 File            \Device\HarddiskVolume2\Program Files (x86)\Common Files\system\ado\msado15.dll
 7772              0x4a8           0x120089 File            \Device\HarddiskVolume2\Windows\SysWOW64\es-ES\propsys.dll.mui
 7772              0x5b0           0x120089 File            \Device\HarddiskVolume2\Windows\SysWOW64\scrrun.dll
```

RAM 14 (volatility – handles -t file – PID=7772)

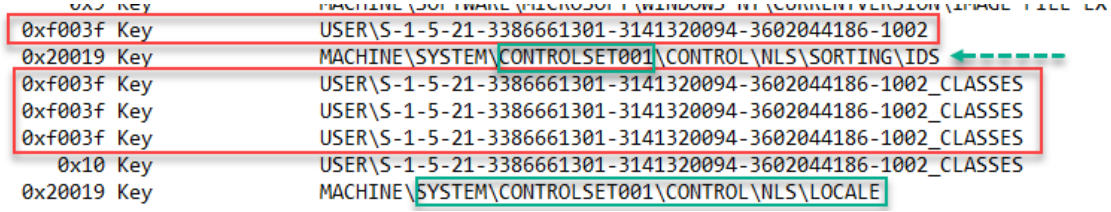
➔ ver imagen handlesFilesPID7772
➔ ver fichero completo: (RAM-fichero&regKEY.txt).

- ➔ Llama la atención en ambas capturas, que el resultado obtenido en la obtención de las DLL pertenecientes a ambos procesos que resultaba lanzar desde la command line del usuario "***AA4BC~1.BER***" (presuntamente ofuscado), se corresponde con el usuario '***a.bernal***' (*remarcado en rojo*), y que en ambas, entra al escritorio del usuario para ejecutar algo.
    - Se entrará en '***C:\Users\a.bernal\desktop'*** buscando un ***.exe*** durante el siguiente apartado mientras se analice el disco duro con "***The Sleuth Kit***".
- ➔ El ***PID=7772*** (PPID del PID=5336) ejecuta desde esa ruta, habilita los drivers para una conexión, ejecuta ***cscript.exe*** para ***montar el entorno API***, el cual *termina de prepararse con la ejecución* de ***srrun.dll*** (*contiene las librerías para las operaciones de escritura, de la lectura y escritura, y de ficheros*), dando paso a la ejecución del PID=5336
    - Entra a '***C:\Users\a.bernal\desktop'*** para ejecutar el proceso y se conecta al entorno API que ha montado el PID=7772 (previsiblemente en Visual Basic según la información obtenida de las DLL a las que llama).

Momento de pasar a buscar las ***claves de registro*** que toca cada proceso. Se muestran 3 capturas, las 2 primeras pertenecen al PID=7772 debido a que toca muchas claves de registro, pero se muestran las más relevantes; y la última, perteneciente al PID=5336.

Se buscan claves que tengan el valor '***0x20019***' (*KEY_EXECUTE = KEY_READ*), o el valor '***0xF003F***' (*KEY_ALL_ACCESS*).

- Información de los tipos de acceso para el registro de Windows según el código: https://docs.microsoft.com/en-us/windows/desktop/sysinfo/registry-key-security-and-access-rights



RAM 15 (volatility – handles -k key – PID=7772)

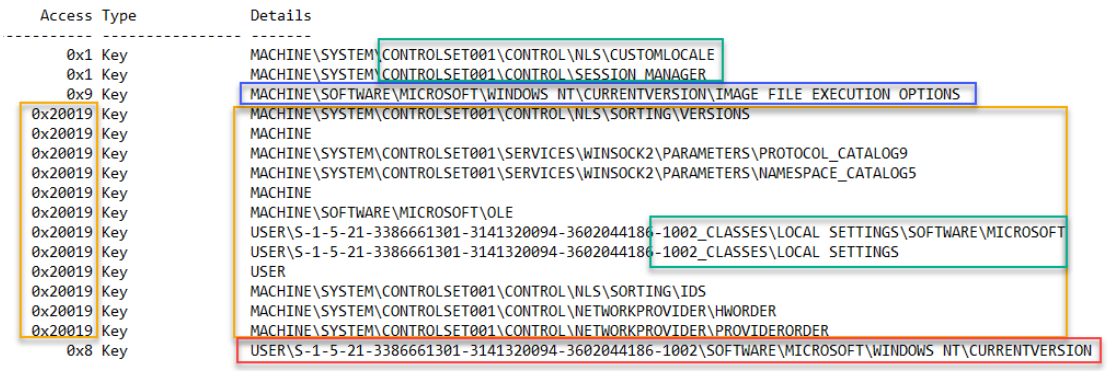
- ➔ En esta primera captura se muestra remarcado en rojo las claves que tienen el acceso '***0xF003F***', resultando curioso que sean las únicas del retorno, y todas para el usuario con SID=*-1002.
    - Remarcado en verde aparece el campo '***CONTROLSET001***', el cual parece ser la creación de un entorno para el IDS, el cual, de alguna forma, engañe a la detección haciendo pasar todo aquel evento acontecido en el entorno '***CONTROLSET001***' como un comportamiento habitual de forma local.

RAM 16 (volatility – handles -k key – PID=7772)

➔ ver imagen handlesKEY7772

➔ En la captura superior se muestran los registros con acceso '**0x20019**':

- ***En verde:*** aparecen todos los registros relacionados con el usuario SID=*-1002
- ***En rosa:*** registros claves que no tienen que ver con el entorno.
- ***En rojo***: Todos los registros de ***WOW6432NODE***, y más en concreto, los que parecen ser los contenedores de la sesión remota para el entrono ***'CONTROLSET001'***, almacenados en las ***'*\PROPERTYBAG'***.
- ***En morado***: la clave de registro ***0x10*** (*DEB_TRACE_LOCKS*), pertenece a la comunicación con Kerberos, y bloquea el debug de la traza, para un '***SYNCROOTMANAGER***'.
- ***En azul***: un indicativo de ***posible caso de persistencia***. Se lanza después de que la ruta de registro ***'\SOFTWARE\WOW6432NODE\MICROSOFT*'*** cargue los registros necesarios para entrar el posible entorno montado en *32-bits mediante el enlazado de las DLL referenciadas en* ***WOW64*** para que el usuario con ***SID=*-1002*** *tenga acceso gracias al registro*. De ahí nace el PID=5336. **'\SOFTWARE\WOW6432NODE\MICROSOFT\WINDOWS NT\CURRENTVERSION**'.
  - La persistencia se buscará en el análisis del kernel.



```
Access Type            Details
------ --------------- -------
   0x1 Key             MACHINE\SYSTEM\CONTROLSET001\CONTROL\NLS\CUSTOMLOCALE
   0x1 Key             MACHINE\SYSTEM\CONTROLSET001\CONTROL\SESSION MANAGER
   0x9 Key             MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION\IMAGE FILE EXECUTION OPTIONS
0x20019 Key            MACHINE\SYSTEM\CONTROLSET001\CONTROL\NLS\SORTING\VERSIONS
0x20019 Key            MACHINE
0x20019 Key            MACHINE\SYSTEM\CONTROLSET001\SERVICES\WINSOCK2\PARAMETERS\PROTOCOL_CATALOG9
0x20019 Key            MACHINE\SYSTEM\CONTROLSET001\SERVICES\WINSOCK2\PARAMETERS\NAMESPACE_CATALOG5
0x20019 Key            MACHINE
0x20019 Key            MACHINE\SOFTWARE\MICROSOFT\OLE
0x20019 Key            USER\S-1-5-21-3386661301-3141320094-3602044186-1002_CLASSES\LOCAL SETTINGS\SOFTWARE\MICROSOFT
0x20019 Key            USER\S-1-5-21-3386661301-3141320094-3602044186-1002_CLASSES\LOCAL SETTINGS
0x20019 Key            USER
0x20019 Key            MACHINE\SYSTEM\CONTROLSET001\CONTROL\NLS\SORTING\IDS
0x20019 Key            MACHINE\SYSTEM\CONTROLSET001\CONTROL\NETWORKPROVIDER\HWORDER
0x20019 Key            MACHINE\SYSTEM\CONTROLSET001\CONTROL\NETWORKPROVIDER\PROVIDERORDER
   0x8 Key             USER\S-1-5-21-3386661301-3141320094-3602044186-1002\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION
```

RAM 17 (volatility – handles -k key – PID=5336)

- ➔ En esta primera captura se muestra remarcado en amarillo las claves que tienen el acceso '***0x20019***':
  - Las 2 primeras entradas dan a entender que se ejecuta sobre ***'CONTROLSET001',*** posible contenedor del entorno para la sesión remota desde '***CUSTOMLOCALE***' y '***SESSION MANAGER***'.
  - La línea remarcada en azul parece ser el enlace al contenedor del entorno mediante '***IMAGE FILE EXECUTION OPTIONS***'.
  - El bloque remarcado en amarillo pertenece a todos los registros que tienen acceso '***0x20019'.***
    - Destacan los 2 registros remarcados en verde, pues parecen ser los registros que parecen ejecutar la autenticación en el posible contenedor.
  - La última línea remarcada en rojo, hace referencia a la ejecución de la posible persistencia.

- ➔ El fichero conteniendo el comando ejecutado con cada proceso relevante en este hilo de investigación y todo el contenido retornado por el plugin ***handles.***
  - (RAM-fichero&regKEY.txt).

2.9. Módulos del kernel

Antes de comenzar a explicar los plugins de kernel que se ven involucrados, es necesario proceder con la explicación de que es el Kernel:

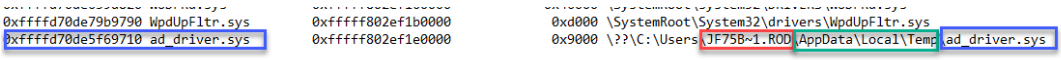
- ***Kernel***: Es la ***parte central de un sistema operativo***, encargado de *realizar toda la comunicación segura entre el software y el hardware* del equipo, *gestionando estos recursos mediante servicios de llamadas al sistema*. También se conoce como la parte que se ejecuta en modo privilegiado, conocido como modo núcleo.
  - Tiene como función básica ***garantizar la carga y la ejecución de los procesos, las entradas/salidas, y proponer una interfaz entre el espacio núcleo o los programas del espacio del usuario***. (Lo que hace la DLL wow64.dll antes de llamar a wow64win.dll, proporcionando la interfaz con el núcleo NT [justo como el último registro de la captura anterior]).

Para esta tarea se pueden usar 2 plugins en volatility (***modules*** y ***modscan***), usándose ambos en este documento para corroborar la evidencia de persistencia en caso de ser encontrada. La persistencia se detecta como un driver cargado en el sistema, pues es un servicio que se ejecuta al arrancar el sistema.

Esta tarea la lleva a cabo el plugin ***modules,*** el cual, se usará para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> --profile=<perfil> modules***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria.
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.
  - ***modules:*** invoca al plugin para ver la lista de procesos del sistema.



RAM 18 (volatility – modules)

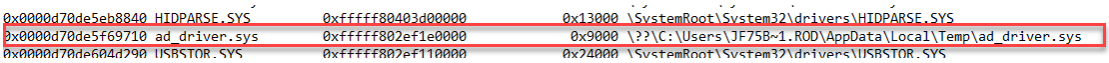
➔ En los resultados obtenidos de la captura superior, se puede deducir que ***se ha encontrado una persistencia*** en la última línea retornada. Llama la atención que no sea capaz de reconocer el inicio de la ruta desde la que se ejecuta el servicio de llamadas al sistema con el nombre "***ad_driver.sys***" (*remarcado en azul*); pero más aún, que sea desde una carpeta temporal (*remarcado en verde*) del usuario '***JF75B~1.ROD***' (*remarcado en rojo*), el cual, parece estar ofuscado.

- '***JF75B~1.ROD***' puede ser '***j.rodriguez'*** por la semejanza el nombre con el del usuario implicado en el documento del TFM.

Momento de pasar a comprobar los resultados obtenidos con el plugin ***modscan,*** el cual, se usará para "***- -profile=Win10x64_16299***".

Para ello, usar el comando:

- " ***volatility -f <ruta/memdump.mem> --profile=<perfil> modscan***"
  - ***volatility:*** invoca al framework.
  - ***-f***: indica de que fichero se hará el análisis de la memoria.
  - ***<ruta/memdump.mem>:*** Ruta hasta el volcado de memoria.
  - ***--profile=<perfil>***: Perfil para el cual se comprueban los procesos.
  - ***modscan:*** invoca al plugin para ver la lista de procesos del sistema.

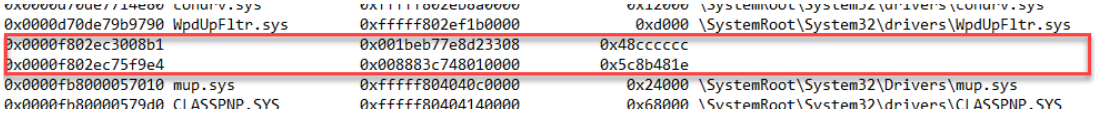


RAM 19 (volatility – modscan – ad_driver.sys)

➔ En los resultados obtenidos de la captura superior, se puede comprobar que ***se ha encontrado una persistencia*** en la línea mostrada del retorno.

➔ Llama la atención, al igual que en el caso de ***modules***, que no sea capaz de reconocer el inicio de la ruta desde la que se ejecuta el servicio de llamadas al sistema con el nombre "***ad_driver.sys***".

➔ Más abajo en el mismo retorno, llama la atención que hay 2 módulos del kernel que no tienen path tal y como se muestra en la imagen inferior.



RAM 20 (volatility – modscan – curiosidad)

➔ Debido a que se ha encontrado un servicio de llamadas al sistema que ejecuta la persistencia para el incidente de seguridad, ir a la ruta desde la que se ejecuta:

- *'**C.\Users\j.rodriguez\AppData\Local\Temp\ad_driver.sys**'*
  - Eliminar el fichero.

➔ El fichero conteniendo el comando ejecutado con cada proceso relevante en este hilo de investigación y todo el contenido retornado por los plugins ***modules y modscan.***

- (RAM-modules&modscan.txt).

3. Disco Duro

Llamo al apartado que se refiere al análisis de la imagen del sistema "***Disco Duro***", pues ***la imagen***, ***se ha obtenido del disco duro*** del equipo del evento.

En este momento, no se pierde el tiempo en ir dando tumbos por los directorios del sistema viendo que hay, pues no se está en el como tal, si no que ***se accede*** *al contenido de la imagen* ***mediante*** *el uso de la toolkit forense "****The Sleuth Kit****"*, el cual se compone de 2 versiones:

- Una versión con ***interfaz gráfica*** (*Autopsy*).
- Una versión por ***terminal de comandos*** (*la que se usará*), compuesta de varias herramientas:
  - ***mmls***: encargada de mostrar las particiones del disco duro en la imagen, la cual retorna el punto de montaje y sector de inicio de la partición, el sector de fin de la misma, y una breve descripción de la misma.
  - ***fls***: encargada de "moverse" por los 'directorios' de la partición a la que se hace referencia mediante el sector de inicio, y accediendo al archivo/directorio deseado mediante su ***inodo***:
    - ***inodo***: Sistema de identificación mediante un número entero único de un fichero en el sistema de archivos.
  - ***icat***: encargada de exportar los ficheros deseados (*pues son evidencias para el posterior análisis obtención de la firma en Virus Total*) desde la partición a la que se hace referencia mediante el sector de inicio, y accediendo al archivo/directorio deseado mediante su ***inodo.***

Llegados a este punto, es momento de buscar y exportar, para su posterior análisis, los siguientes ficheros:

- ***cAhNPiapwJQcd.exe → c:\Users\a.bernal\Desktop*** (una de las rutas desde la que se ejecuta y/o tiene acceso el proceso 5336.
- ***cscript.exe → c:\Windows\SysWOW64\cscript.exe*** (ejecutado desde esa ruta y perteneciente al PID=7772).
- ***cscript.exe.mui → c:\Windows\SysWOW64\es-ES\cscript.exe.mui*** (ejecutado desde esa ruta y perteneciente al PID=7772).

3.1 Recuperar cAhNPiapwJQcd.exe

Como se explica más arriba, lo primero de todo es conocer el sector desde el que se monta la partición '***C:***', pues en ella, se encuentran los ficheros que se desea extraer para su posterior análisis.

La herramienta encargada (dentro de la toolkit forense '***The Sleuth Kit***') es ***mmls.***

Para ello, se usa el comando:

- "***mmls -i afflib <ruta/imagen.raw.001>***"
  - ***mmls***: invoca a la herramienta.
  - ***-i:*** se indica que se pasará el parámetro de la librería que monte la imagen.
  - ***afflib:*** librería usada para montar la imagen.
  - ***<ruta/imagen.raw.001>:*** ruta de la imagen que se monta.

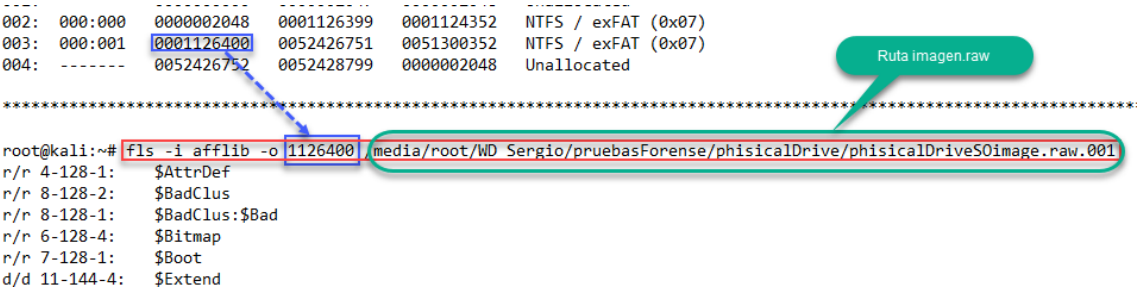
```
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

      Slot      Start        End          Length       Description
000:  Meta      0000000000   0000000000   0000000001   Primary Table (#0)
001:  -------   0000000000   0000002047   0000002048   Unallocated
002:  000:000   0000002048   0001126399   0001124352   NTFS / exFAT (0x07)
003:  000:001   0001126400   0052426751   0051300352   NTFS / exFAT (0x07)
004:  -------   0052426752   0052428799   0000002048   Unallocated
```

HDD 1 (mmls – mostrando sectori inicio C:\)

➔ Se obtiene que la partición ***C:\ tiene su punto de montaje*** en el sector ***0001126400*** del disco duro que se analiza desde la imagen.

Conociendo el sector de inicio () de la partición C:\, toca usar la herramienta fls para moverse por los directorios. La sintaxis del comando es la siguiente:

- "***fls -i afflib -o <sectorInicio> <ruta/imagen.raw.001> <inodo>***"
  - ***fls***: invoca a la herramienta.
  - ***-i:*** se indica que se pasará el parámetro de la librería que monte la imagen.
  - ***afflib:*** librería usada para montar la imagen.
  - ***-o***: se indica que se pasará el parámetro punto de montaje.
  - ***<sectorInicio>:*** sector en el que se monta la partición deseada**.**
  - ***<ruta/imagen.raw.001>:*** ruta de la imagen que se monta.
  - ***<inodo>:*** inodo del fichero al que se desea acceder**.**



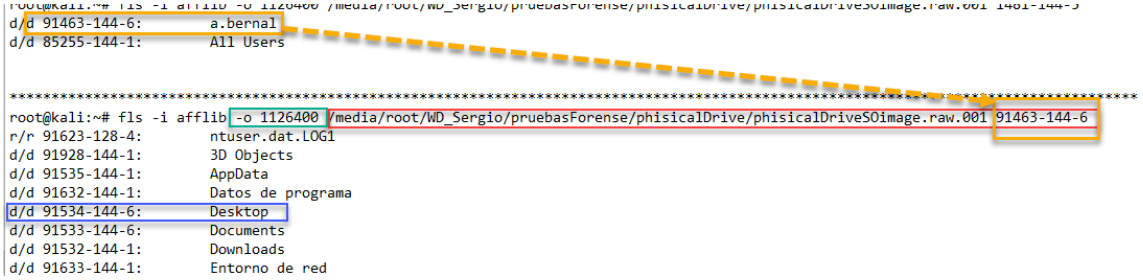
HDD 2 (fls – Acceder a C:\)

➔ Ver imagen fls-1.jpg

➔ En la captura superior se accede a la unidad ***C:*** desde el sector ***1126400***.

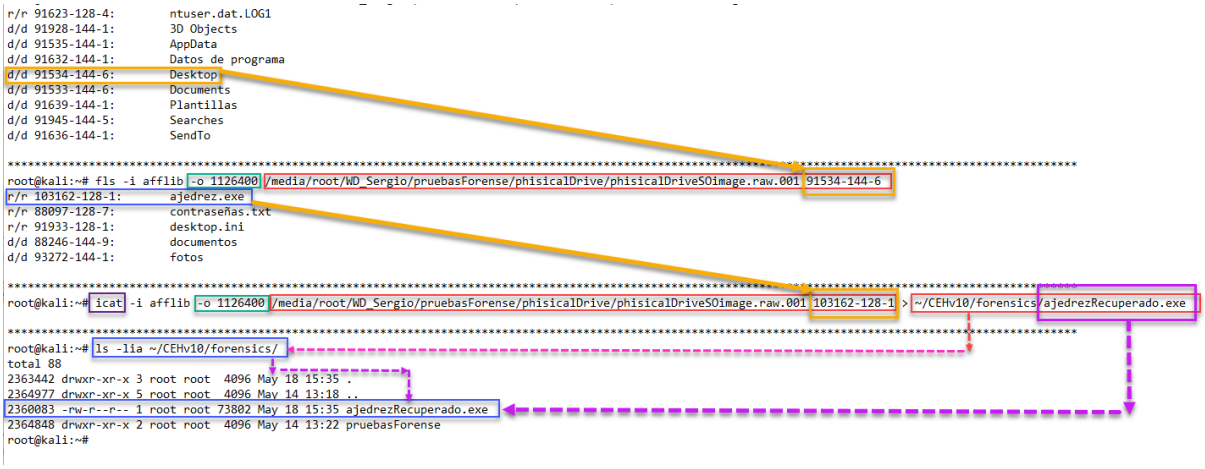
HDD 3 (fls – Acceder a C:\Users)

➔ En la captura superior se accede a la unidad ***C:\Users*** desde el sector ***1126400*** y el inodo ***1481-144-5***.



HDD 4 (fls – Acceder a C:\Users\a.bernal)

➔ En la captura superior se accede a la unidad ***C:\Users\a.bernal*** desde el sector ***1126400*** y el inodo ***91463-144-6***.



HDD 5 (fls –C:\Users\a.bernal\Desktop – icat exportando ajedrez.exe)

➔ En la captura superior se accede a la unidad ***C:\Users\a.bernal\Desktop*** desde el sector ***1126400*** y el inodo ***91534-144-6***.

- Se ha encontrado un único fichero ejecutable en el directorio (.exe), lo cual indica, que es el fichero que se ejecuta desde esta ruta.
  - Llama la atención que dicho fichero no tenga el nombre ***cAhNPiapwJQcd.exe,*** por lo que se deduce, que ***el fichero real es ajedrez.exe y se ofusca con el nombre cAhNPiapwJQcd.exe.***
- Se ha realizado la exportación del fichero mediante el uso de la herramienta icat. Para ello, la sintaxis del comando usado es:
  - "***icat -i afflib -o <sectorInicio> <ruta/imagen.raw.001> <inodo> > <ruta/ficheroExportado>***"
    - ***icat***: invoca a la herramienta.
    - ***-i:*** se indica que se pasará el parámetro de la librería que monte la imagen.
    - ***afflib:*** librería usada para montar la imagen.
    - ***-o***: se indica que se pasará el parámetro punto de montaje.
    - ***<sectorInicio>:*** sector en el que se monta la partición deseada***.***
    - ***<ruta/imagen.raw.001>:*** ruta de la imagen que se monta.
    - ***<inodo>:*** inodo del fichero que se desea exportar***.***
    - > : indica que se va a pasar una ruta donde exportar.
    - ***<ruta/ficheroExportado>***: ruta y nombre del fichero donde se guardará y nombrará una vez se exporte.
- Mediante el uso del comando "ls -lia <ruta>" se ha comprobado y mostrado como se ha exportado el fichero con éxito.

➔ Mostrar imagen recuperarAjedrez.jpg

➔ El fichero contiene el comando ejecutado, con todo el contenido retornado por las herramientas de "***The Sleuth Kit***" (mmls, fls, icat) usadas para exportar el fichero ***ajedrezRecuperado.exe.***

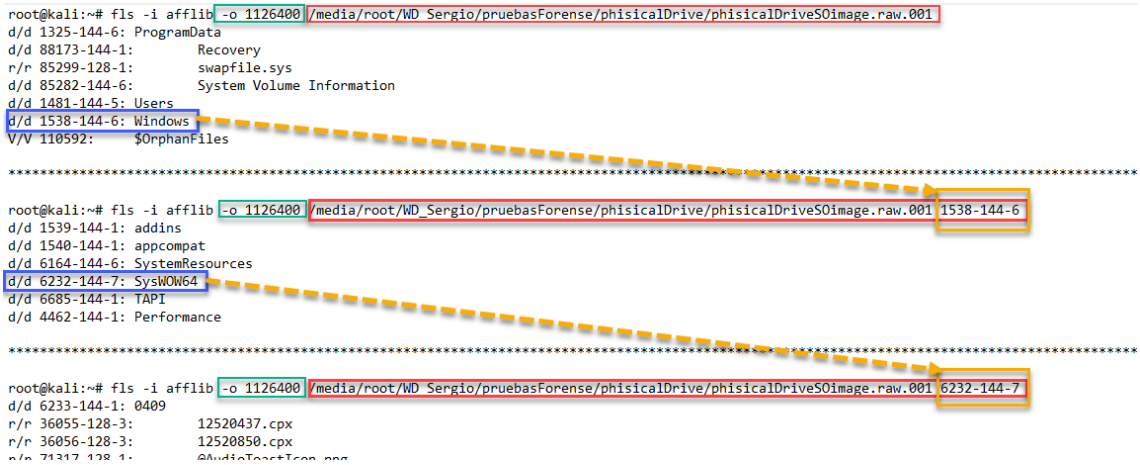
- (HDD-recuperarAjedrez.txt).

3.2 Recuperar cscript.exe y cscript.exe.mui

Momento de exportar para el análisis los otros 2 ficheros implicados. Como ya se explicó más arriba en el documento la sintaxis del comando para fls y el movimiento por los directorios, se ha capturado y resumido en 1 foto (HDD-recuperarCscript(exe&exeMUI).txt todo el contenido retornado por los comandos) en la navegación hasta la ruta ***C:\Windows\SysWOW64***, la cual es compartida por ambos ficheros.

Se accede a dicha ruta mediante:

- ***1126400:*** Sector de inicio del punto de montaje de ***C:***
- ***1538-144-6:*** inodo perteneciente a ***C:\Windows***
- ***6232-144-7:*** inodo perteneciente a ***C:\Windows\SysWOW64***

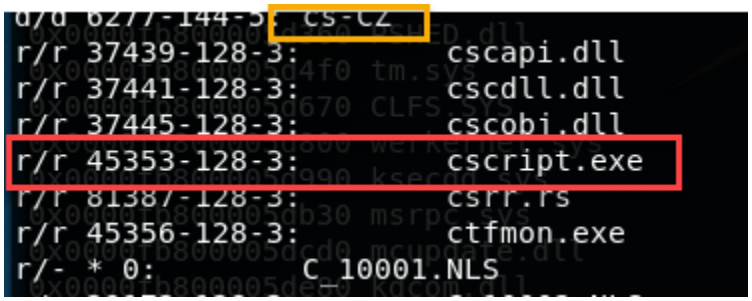


HDD 6 (fls – C:\Windows\SysWOW64)

➔ Ver imagen fls-2-syswow64.jpg

Dentro de la ruta ***C:\Windows\SysWOW64\,*** es momento de buscar cual es el inodo perteneciente al fichero ***cscript.exe***; debido a que el retorno es demasiado largo, se saca una captura cuando se busca para proceder más tarde a la exportación del fichero.

➔ En la captura inferior se muestra el que el inodo perteneciente a ***cscript.exe*** es ***45353-128-3***, y se ha encontrado debajo del directorio de "***cs-CZ***"



HDD 7 (fls – C:\Windows\SysWOW64\cscript.exe con inodo)

Dentro de la ruta ***C:\Windows\SysWOW64\,*** es momento de buscar cual es el inodo perteneciente al directorio ***\es-ES*** (*en el cual se encuentra cscript.exe.mui*); debido a que el retorno es demasiado largo, se saca una captura del inodo perteneciente al directorio.

➔ En la captura inferior se muestra que el inodo 6302-144-6 es el perteneciente a la ruta C:\Windows\SysWOW64\es-ES\. Acceder a ella mediante la herramienta fls tal y como se muestra en el documento HDD-recuperarCscript(exe&exeMUI).txt

```
d/d 6301-144-1:  es
d/d 6302-144-6:  es-ES
d/d 6310-144-5:  es-MX
r/r 38118-128-3:         es.dll
```

HDD 8 (fls – C:\Windows\SysWOW64\es-Es\)

Dentro de la ruta ***C:\Windows\SysWOW64\es-Es\,*** es momento de buscar cual es el inodo perteneciente al fichero ***cscript.exe.mui***; debido a que el retorno es demasiado largo, se saca una captura cuando se busca para proceder más tarde a la exportación del fichero.

➔ En la captura inferior se muestra el que el inodo perteneciente a ***cscript.exe.mui*** es ***28287-128-4.***

```
r/r 66667-128-4:         cscobj.dll.mui
r/r 28287-128-4:         cscript.exe.mui
r/r 66670-128-4:         CSRR.rs.mui
```

HDD 9 (fls – C:\Windows\SysWOW64\es-ES\cscript.exe.mui con inodo)

Por último, se usa la herramienta icat para exportar los dos ficheros deseados (cscript.exe y cscript.exe.mui) mediante su inode:

- ***45353-128-3:*** inode de ***cscript.exe***
- ***28287-128-4:*** inode de ***cscript.exe.mui***

➔ En la captura inferior se muestra como se ejecuta el comando de la herramienta icat para ambos ficheros, y mediante el uso de "ls -lia <ruta>" se muestra que ambos ficheros se han exportado de forma satisfactoria.

```
root@kali:~# icat -i afflib -o 1126400 /media/root/WD_Sergio/pruebasForense/phisicalDrive/phisicalDriveSOimage.raw.001 45353-128-3 > ~/CEHv10/forensics/cscriptRecuperado.exe
root@kali:~# icat -i afflib -o 1126400 /media/root/WD_Sergio/pruebasForense/phisicalDrive/phisicalDriveSOimage.raw.001 28287-128-4 > ~/CEHv10/forensics/cscriptRecuperadoMUI.exe.mui
**********

root@kali:~# ls -lia ~/CEHv10/forensics/
total 244
2363442 drwxr-xr-x 3 root root   4096 May 18 16:08 .
2364977 drwxr-xr-x 5 root root   4096 May 14 13:18 ..
2360083 -rw-r--r-- 1 root root  73802 May 18 15:35 ajedrezRecuperado.exe
2365218 -rw-r--r-- 1 root root 142848 May 18 16:06 cscriptRecuperado.exe
2365219 -rw-r--r-- 1 root root  13824 May 18 16:07 cscriptRecuperadoMUI.exe.mui
2364848 drwxr-xr-x 2 root root   4096 May 14 13:22 pruebasForense
root@kali:~#
```

HDD 10 (icat – exportacion y muestra de cscript.exe y cscript.exe.mui)

➔ Ver imagen icat-cscript.jpg

➔ El fichero contiene el comando ejecutado, con todo el contenido retornado por las herramientas de "The Sleuth Kit" (mmls, fls, icat) usadas para exportar el fichero ***cscriptRecuperadp.exe y cscriptRecuperado.exe.mui.***

- (HDD-recuperarCscript(exe&exeMUI).txt).

[Momento de pasar al análisis de los ficheros exportados en Virus Total.]

4. Virus Total

La forma de comprobar si un código es malicioso o no que tiene Virus Total, es mediante las firmas de los ficheros.

Las firmas, son un hash que identifica unívocamente al programa malicioso, o a parte del código que compone el programa. Puede no ser detectado en caso de ser un código demasiado sofisticado, pues usará la ofuscación para evadir los sistemas defensivos.

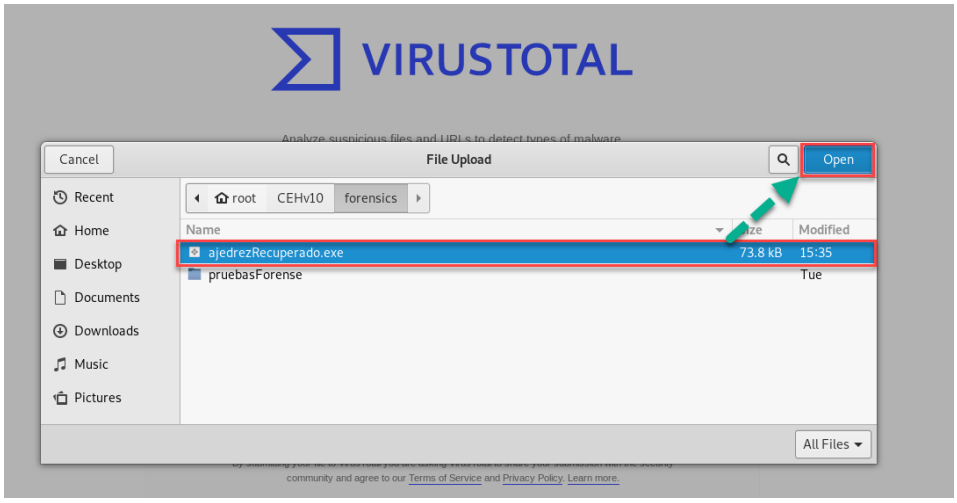
Para acceder a virus total, abrir el navegador y poner en Google "Virus Total", o seguir el enlace:

- https://www.virustotal.com/gui/home/upload

Seleccionar el botón "***Choose file***" para abrir un explorador y buscar el fichero que se desea subir para el análisis.

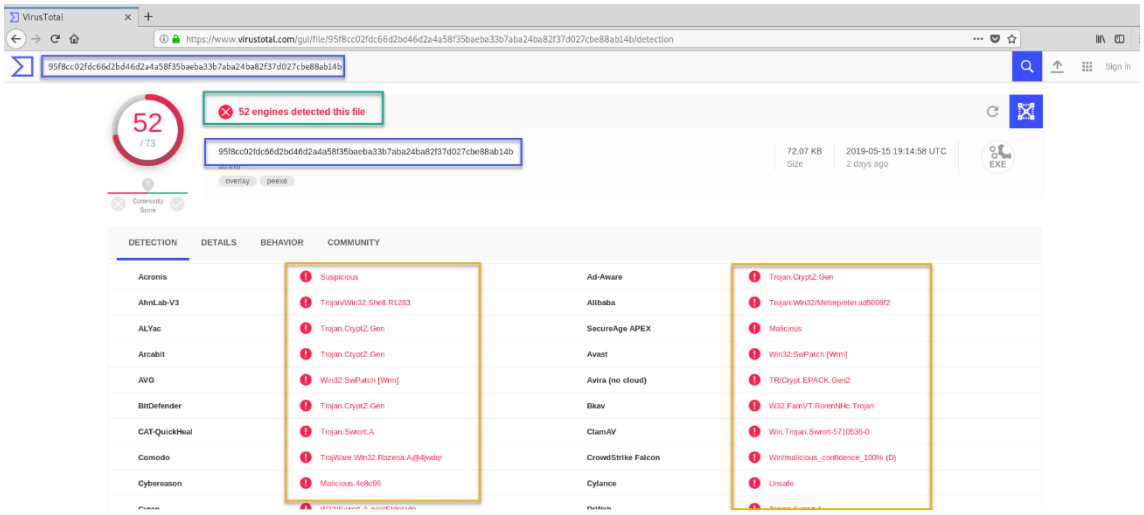
4.1. Subida a virus total

Navegar por el explorador hasta encontrar el fichero ***ajedrezRecuperado.exe*** en la ruta donde esté almacenado tras la exportación (***/home/root/CHEv10/forensics/ajedrezRecuperado.exe***) y dar en el botón "***Open***".



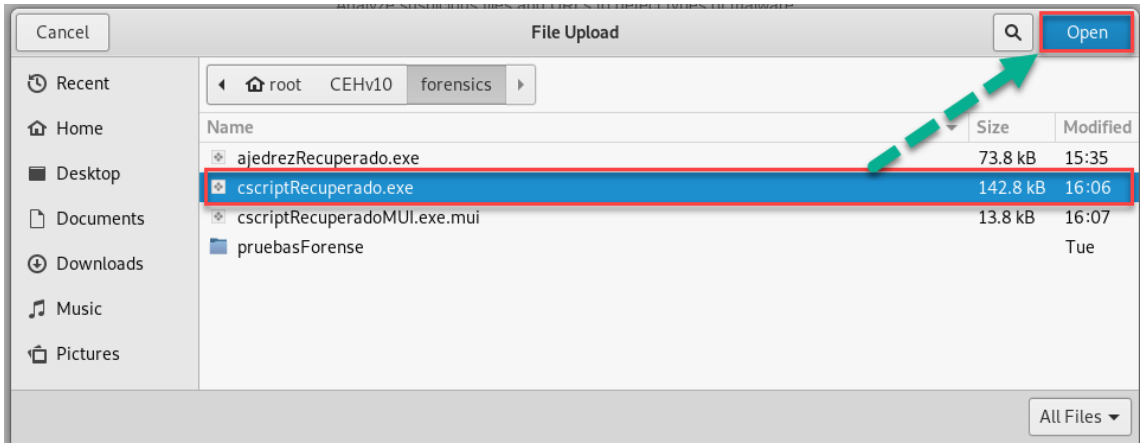
Virus Total 1 (Subir fichero sospechoso – Ajedrez.exe)

Finalizado el análisis, retorna un resultado el cual era de esperar (por las pruebas encontradas pues era correcto el camino seguido en la investigación); pero a la par, alarma que tantas marcas lo detecten como algo malicioso, y que se colase con tanta facilidad.



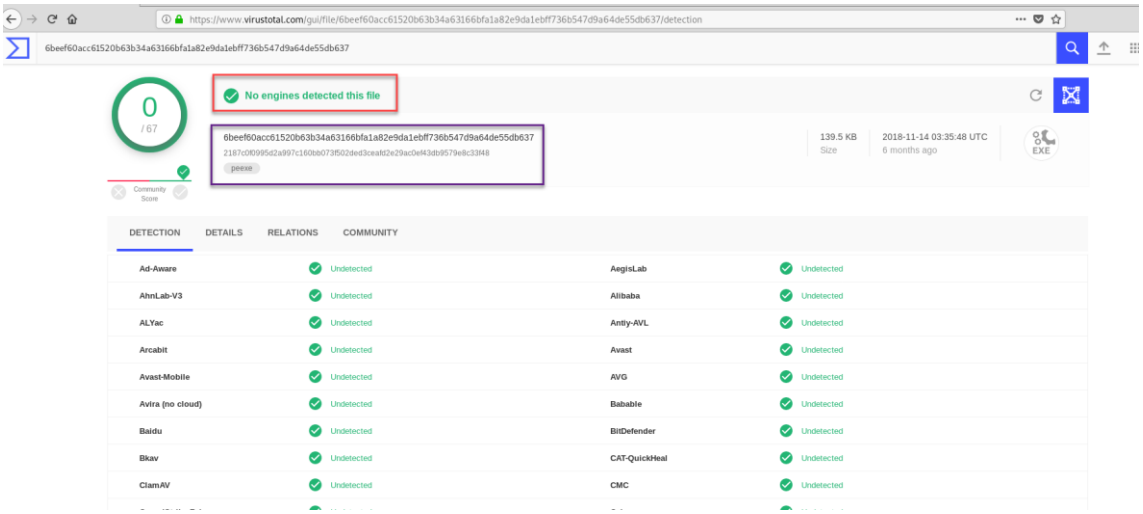
Virus Total 2 (Resultados del análisis – Ajedrez.exe)

Navegar por el explorador hasta encontrar el fichero ***cscriptRecuperado.exe*** en la ruta donde esté almacenado tras la exportación (***/home/root/CHEv10/forensics/cscriptRecuperado.exe***) y dar en el botón "***Open***".



Virus Total 3 (Subir fichero sospechoso – cscript.exe)

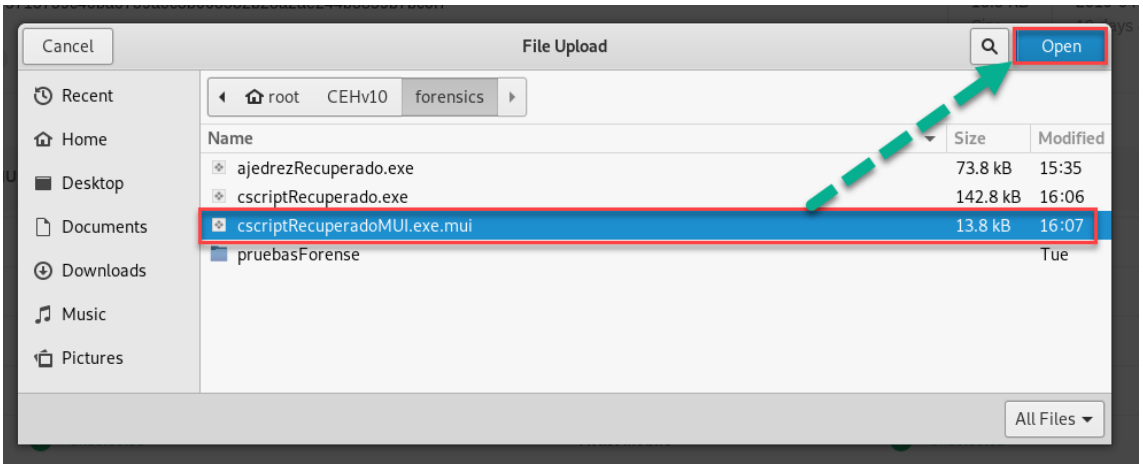
Finalizado el análisis, retorna un resultado el cual ***no era de esperar*** (por las pruebas encontradas pues era correcto el camino seguido en la investigación).



Virus Total 4 (Resultados del análisis – cscript.exe)

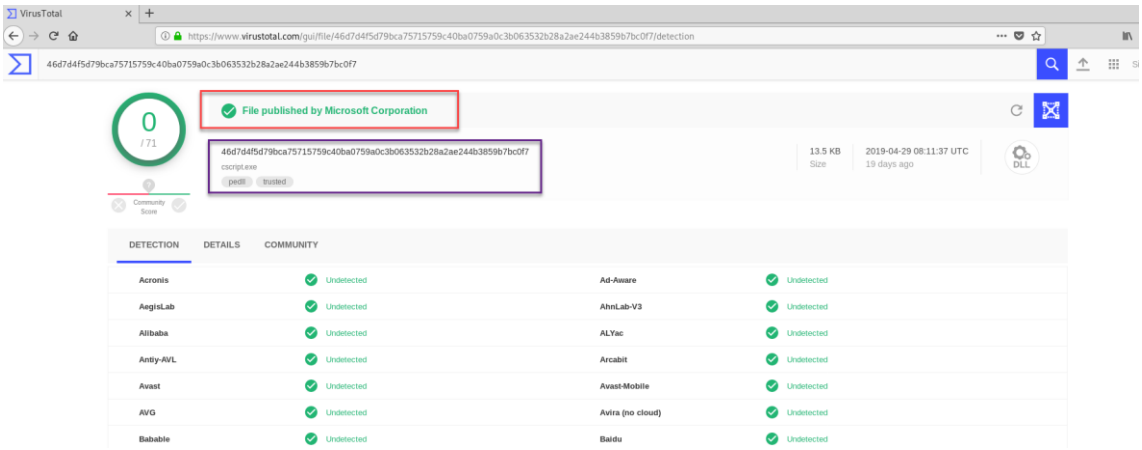
Navegar por el explorador hasta encontrar el fichero ***cscriptRecuperadoMUI.exe*** en la ruta donde esté almacenado tras la exportación:

- (***/home/root/CHEv10/forensics/cscriptRecuperadoMUI.exe***) y dar en el botón "***Open***".



Virus Total 5 (Subir fichero sospechoso – cscript.exe.mui)

Finalizado el análisis, retorna un resultado el cual ***no era de esperar*** (por las pruebas encontradas pues era correcto el camino seguido en la investigación).



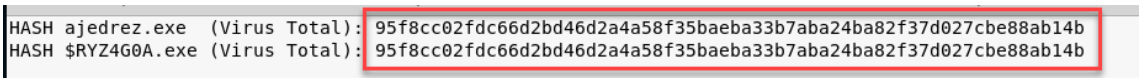
Virus Total 6 (Resultados del análisis – cscript.exe.mui)

- ➔ La primera conclusión del análisis es que fue exitosa la detección del fichero ***Ajedrez.exe***, ya que una vez recuperado y subido al análisis, retorna el resultado de ser un troyano (similar resultado al obtenido por ***$RYZ4G0A.exe*** en el documento del TFM), así como de tener un tamaño parecido si no similar (*72.02KB*), el cual se comprueba mediante el HASH obtenido en el siguiente paso.
- ➔ Es de alarmar que ***al analizar los archivos que lanzaron el ejecutable malicioso*** perteneciente al incidente, ***no de resultados de código malicioso*** pese a ser reconocido como ***peexe*** o ***pedll***.
    - ***De no ejecutarse*** estos archivos, ***no se realizaría la llamada*** a ***cAhNPiapwJQcd.exe*** con un ***PID=5336***; propietario de una ***conexión establecida*** por ***TCPv4*** a la dirección ***10.0.27.6:37723***.

4.2. Comprobación del HASH

Demostrar que el HASH extraído de ajedrez.exe es el mismo que el del TFM.

Para esta comprobación se generó un fichero en el que se pusieron ambos hashes, y el cual se hace entrega para comprobar que el contenido es el similar al mostrado en la siguiente imagen.



Virus Total 7 (Firma del malware en VT)

➔ Fichero con la comprobación del HASH: HASHajedrezVT.txt

➔ El tener el mismo HASH significa que tienen el mismo contenido, con nombres distintos, pero el contenido del fichero es exactamente igual, lo que indica:
- Ajedrez.exe es el inicio de todo el incidente.
- ***$RYZ4G0A.exe*** fue *eliminado del escritorio* de ***'j.rodriguez'*** (conclusión del TFM) pero ***su nombre original debía ser Ajedrez.exe***

5. Índice de imágenes

5.1. RAM

5.2. Disco Duro

5.3. Virus Total