

# Ejercicio Análisis de Memoria, Volatility

— — —

Javier Alvarez

2025

(Me gusta Volatility)

```
-(javi@kali)-[~/volatility]
$ python2.7 vol.py -h
Volatility Foundation Volatility Framework 2.6.1
Usage: Volatility - A memory forensics analysis platform.

Options:
-h, --help                list all available options and their default values.
                           Default values may be set in the configuration file
                           (/etc/volatilityrc)
--conf-file=/home/javi/.volatilityrc
                           User based configuration file
-d, --debug               Debug volatility
--plugins=PLUGINS         Additional plugin directories to use (colon separated)
--info                    Print information about all registered objects
--cache-directory=/home/javi/.cache/volatility
                           Directory where cache files are stored
--cache                   Use caching
--tz=TZ                   Sets the (Olson) timezone for displaying timestamps
                           using pytz (if installed) or tzset
-f FILENAME, --filename=FILENAME
                           Filename to use when opening an image
--profile=WinXPSP2x86     Name of the profile to load (use --info to see a list
                           of supported profiles)
-l LOCATION, --location=LOCATION
                           A URN location from which to load an address space
-w, --write               Enable write support
--dtb=DTB                 DTB Address
--shift=SHIFT             Mac KASLR shift address
--output=text             Output in this format (support is module specific, see
                           the Module Output Options below)
--output-file=OUTPUT_FILE
                           Write output in this file
-v, --verbose             Verbose information
--physical_shift=PHYSICAL_SHIFT
                           Linux kernel physical shift address
--brl2rcf=brl2rcf=brl2rcf
                           brl2rcf=brl2rcf
-A' --leprose             Leprose information
                           Mlife onibnc in rps tife
--onibnc-tife=ONIBNC-TIFE
                           rps woqng onibnc obfious pefom)
--onibnc=rcf             onibnc in rps forwsc (snbbolc is woqng zbecific' zec
--brl2rcf=brl2rcf       W9c KV2GK brl2rcf qdqlezz
--qrp=DIB                DIB Vdqlezz
-M' --mlife               Eusprc mlife snbbolc
                           V-owk locatou lton eusprc to load an address space
```

# El perfil recomendado para el análisis

---

Output del comando **imageinfo**, que sirve para identificar el perfil correcto del sistema operativo en la imagen de memoria (**WinXPSP2x86**). Otros datos útiles, **PAE** type PAE, **Image date and time** 10 de octubre de 2011, **AS Layer** capas de abstracción, **KDBG** dirección del bloque de depuración del kernel.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem imageinfo
Volatility Foundation Volatility Framework 2.6.1
INFO      : volatility.debug      : Determining profile based on KDBG search...
           Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
           AS Layer1            : IA32PagedMemoryPae (Kernel AS)
           AS Layer2            : FileAddressSpace (/home/javi/volatility/windows.vmem)
           PAE type             : PAE
           DTB                  : 0x319000L
           KDBG                 : 0x80544ce0L
           Number of Processors : 1
           Image Type (Service Pack) : 2
           KPCR for CPU 0       : 0xffdff000L
           KUSER_SHARED_DATA     : 0xffdf0000L
           Image date and time   : 2011-10-10 17:06:54 UTC+0000
           Image local date and time : 2011-10-10 13:06:54 -0400
```



# Detalle de los perfiles recomendado para el análisis

— — —

Perfiles sugeridos: **WinXPSP2x86**, **WinXPSP3x86**. Perfil instanciado **WinXPSP2x86**. PAE **activado**. Número de procesadores **1**. Fecha de la imagen **2011-10-10 17:06:54 UTC**. Direccion del KDBG: **0x80544ce0**. Tipo de imagen **Windows XP Service Pack 2**, arquitectura de **32 bits**.

```
(javi@kali)-[~/volatility]
└─$ python2.7 vol.py -f windows.vmem imageinfo
Volatility Foundation Volatility Framework 2.6.1
INFO      : volatility.debug      : Determining profile based on KDBG search...
           Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
           AS Layer1            : IA32PagedMemoryPae (Kernel AS)
           AS Layer2            : FileAddressSpace (/home/javi/volatility/windows.vmem)
           PAE type             : PAE
           DTB                  : 0x319000L
           KDBG                 : 0x80544ce0L
           Number of Processors : 1
           Image Type (Service Pack) : 2
           KPCR for CPU 0       : 0xffdff000L
           KUSER_SHARED_DATA    : 0xffdf0000L
           Image date and time  : 2011-10-10 17:06:54 UTC+0000
           Image local date and time : 2011-10-10 13:06:54 -0400
```

# Listado de procesos pslist

Se ejecuto el comando Pslist para obtener el listado de procesos activos. En este listado se identificaron varios procesos sospechosos.  cmd.exe PID544 padre explorer.exe PID1956, el explorador no suele lanzar la consola por si solo.  svchost.exe PID964 con Handles 1058, consumo excesivo de recursos, puede indicar que esta cargando servicios maliciosos.

```
(javi@kali) - [~/volatility]
$ python2.7 vol.py -f windows.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	55	162		0		
0x81945020	smss.exe	536	4	3	21		0	2011-10-10 17:03:56 UTC+0000	
0x816c6020	csrss.exe	608	536	11	355	0	0	2011-10-10 17:03:58 UTC+0000	
0x813a9020	winlogon.exe	632	536	24	533	0	0	2011-10-10 17:03:58 UTC+0000	
0x816da020	services.exe	676	632	16	261	0	0	2011-10-10 17:03:58 UTC+0000	
0x813c4020	lsass.exe	688	632	23	336	0	0	2011-10-10 17:03:58 UTC+0000	
0x81772ca8	vmacthlp.exe	832	676	1	24	0	0	2011-10-10 17:03:59 UTC+0000	
0x8167e9d0	svchost.exe	848	676	20	194	0	0	2011-10-10 17:03:59 UTC+0000	
0x817757f0	svchost.exe	916	676	9	217	0	0	2011-10-10 17:03:59 UTC+0000	
0x816c6da0	svchost.exe	964	676	63	1058	0	0	2011-10-10 17:03:59 UTC+0000	
0x815daca8	svchost.exe	1020	676	5	58	0	0	2011-10-10 17:03:59 UTC+0000	
0x813aeda0	svchost.exe	1148	676	12	187	0	0	2011-10-10 17:04:00 UTC+0000	
0x817937e0	spoolsv.exe	1260	676	13	140	0	0	2011-10-10 17:04:00 UTC+0000	
0x81754990	VMwareService.e	1444	676	3	145	0	0	2011-10-10 17:04:00 UTC+0000	
0x8136c5a0	alg.exe	1616	676	7	99	0	0	2011-10-10 17:04:01 UTC+0000	
0x815c4da0	wsentfy.exe	1920	964	1	27	0	0	2011-10-10 17:04:39 UTC+0000	
0x813bcd0	explorer.exe	1956	1884	18	322	0	0	2011-10-10 17:04:39 UTC+0000	
0x816d63d0	VMwareTray.exe	184	1956	1	28	0	0	2011-10-10 17:04:41 UTC+0000	
0x8180b478	VMwareUser.exe	192	1956	6	83	0	0	2011-10-10 17:04:41 UTC+0000	
0x818233c8	reader_sl.exe	228	1956	2	26	0	0	2011-10-10 17:04:41 UTC+0000	
0x815e7be0	wuauclt.exe	400	964	8	173	0	0	2011-10-10 17:04:46 UTC+0000	
0x817a34b0	cmd.exe	544	1956	1	30	0	0	2011-10-10 17:06:42 UTC+0000	



# La jerarquía de los procesos pstree

explorer.exe PID: 1956

cmd.exe PID: 544 PPID: 1956

Esto quiere decir que explorer.exe lanzó cmd.exe, lo cual es sospechoso, porque el explorador normalmente no lanza la consola por sí solo.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 pstree
Volatility Foundation Volatility Framework 2.6.1
```


Name	Pid	PPid	Thds	Hnds	Time
0x819cc830:System	4	0	55	162	1970-01-01 00:00:00 UTC+0000
. 0x81945020:smss.exe	536	4	3	21	2011-10-10 17:03:56 UTC+0000
.. 0x816c6020:csrss.exe	608	536	11	355	2011-10-10 17:03:58 UTC+0000
.. 0x813a9020:winlogon.exe	632	536	24	533	2011-10-10 17:03:58 UTC+0000
... 0x816da020:services.exe	676	632	16	261	2011-10-10 17:03:58 UTC+0000
.... 0x817757f0:svchost.exe	916	676	9	217	2011-10-10 17:03:59 UTC+0000
.... 0x81772ca8:vmacthlp.exe	832	676	1	24	2011-10-10 17:03:59 UTC+0000
.... 0x816c6da0:svchost.exe	964	676	63	1058	2011-10-10 17:03:59 UTC+0000
..... 0x815c4da0:wscntfy.exe	1920	964	1	27	2011-10-10 17:04:39 UTC+0000
..... 0x815e7be0:wuauclt.exe	400	964	8	173	2011-10-10 17:04:46 UTC+0000
..... 0x8167e9d0:svchost.exe	848	676	20	194	2011-10-10 17:03:59 UTC+0000
.... 0x81754990:VMwareService.e	1444	676	3	145	2011-10-10 17:04:00 UTC+0000
.... 0x8136c5a0:alg.exe	1616	676	7	99	2011-10-10 17:04:01 UTC+0000
.... 0x813aeda0:svchost.exe	1148	676	12	187	2011-10-10 17:04:00 UTC+0000
.... 0x817937e0:spoolsv.exe	1260	676	13	140	2011-10-10 17:04:00 UTC+0000
.... 0x815daca8:svchost.exe	1020	676	5	58	2011-10-10 17:03:59 UTC+0000
... 0x813c4020:lsass.exe	688	632	23	336	2011-10-10 17:03:58 UTC+0000
0x813bcd0:explorer.exe	1956	1884	18	322	2011-10-10 17:04:39 UTC+0000
. 0x8180b478:VMwareUser.exe	192	1956	6	83	2011-10-10 17:04:41 UTC+0000
. 0x817a34b0:cmd.exe	544	1956	1	30	2011-10-10 17:06:42 UTC+0000
. 0x816d63d0:VMwareTray.exe	184	1956	1	28	2011-10-10 17:04:41 UTC+0000
. 0x818233c8:reader_sl.exe	228	1956	2	26	2011-10-10 17:04:41 UTC+0000

# Posibles procesos ocultos psxview

Este plugin compara y te muestra si un proceso aparece en pslist y no en psscan por ejemplo. Si pslist es false pero otros métodos es true es que está oculto. No aparece en el listado normal pero si en la memoria. Hay tres servicios ocultos System, smss.exe y csrss. ● **System** oculto en pspcid, session, deskthrd **debería ser visible pasa algo aquí.** ● **smss.exe** Session Manager Subsystem es uno de los primeros procesos en arrancar Windows, **si esta oculto puede haber sido modificado.** ● csrss.exe solo tiene un false pero habría que revisarlo.

```
(Javi@kali) [~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 psxview
Volatility Foundation Volatility Framework 2.6.1
```

Offset(P)	Name	PID	pslist	psscan	thrdproc	pspcid	csrss	session	deskthrd	ExitTime
0x015a9020	winlogon.exe	632	True	True	True	True	True	True	True	
0x018da020	services.exe	676	True	True	True	True	True	True	True	
0x0156c5a0	alg.exe	1616	True	True	True	True	True	True	True	
0x018d63d0	VMwareTray.exe	184	True	True	True	True	True	True	True	
0x019757f0	svchost.exe	916	True	True	True	True	True	True	True	
0x015c4020	lsass.exe	688	True	True	True	True	True	True	True	
0x01972ca8	vmacthlp.exe	832	True	True	True	True	True	True	True	
0x019a34b0	cmd.exe	544	True	True	True	True	True	True	True	
0x0187e9d0	svchost.exe	848	True	True	True	True	True	True	True	
0x017daca8	svchost.exe	1020	True	True	True	True	True	True	True	
0x01954990	VMwareService.e	1444	True	True	True	True	True	True	True	
0x018c6da0	svchost.exe	964	True	True	True	True	True	True	True	
0x01a233c8	reader_sl.exe	228	True	True	True	True	True	True	True	
0x017e7be0	wuauclt.exe	400	True	True	True	True	True	True	True	
0x019937e0	spoolsv.exe	1260	True	True	True	True	True	True	True	
0x015bcdad	explorer.exe	1956	True	True	True	True	True	True	True	
0x017c4da0	wscntfy.exe	1920	True	True	True	True	True	True	True	
0x01a0b478	VMwareUser.exe	192	True	True	True	True	True	True	True	
0x015aeda0	svchost.exe	1148	True	True	True	True	True	True	True	
0x01bcc830	System	4	True	True	True	True	False	False	False	
0x01b45020	smss.exe	536	True	True	True	True	False	False	False	
0x018c6020	csrss.exe	608	True	True	True	True	False	True	True	



# Procesos su path y comandos dlllist, cmdscan y consoles

El proceso cmd.exe PID 544, lanzado por explorer.exe PID 1956, tenía como ruta C:\WINDOWS\system32\cmd.exe. A través de los plugins cmdscan y consoles se identificaron comandos ejecutados. Sacamos los dll del hijo no del padre, ahí me había equivocado. Aunque se haga a través del padre el malicioso es el hijo.

```
[javi@kali] (~/.volatility)
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 dlllist -p 544
Volatility Foundation Volatility Framework 2.6.1
*****
cmd.exe pid: 544
Command line : "C:\WINDOWS\system32\cmd.exe"
Service Pack 2
```

Base	Size	LoadCount	LoadTime	Path
0x4ad00000	0x61000	0xffff		C:\WINDOWS\system32\cmd.exe
0x7c900000	0xb0000	0xffff		C:\WINDOWS\system32\ntdll.dll
0x7c800000	0xf4000	0xffff		C:\WINDOWS\system32\kernel32.dll
0x77c10000	0x58000	0xffff		C:\WINDOWS\system32\msvcrt.dll
0x77d40000	0x90000	0xffff		C:\WINDOWS\system32\USER32.dll
0x77f10000	0x46000	0xffff		C:\WINDOWS\system32\GDI32.dll
0x5cb70000	0x26000	0x1		C:\WINDOWS\system32\ShimEng.dll
0x6f880000	0x1ca000	0x1		C:\WINDOWS\AppPatch\AcGenral.DLL
0x77dd0000	0x9b000	0x17		C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000	0x91000	0xb		C:\WINDOWS\system32\RPCRT4.dll
0x76b40000	0x2d000	0x2		C:\WINDOWS\system32\WINMM.dll
0x774e0000	0x13c000	0x2		C:\WINDOWS\system32\ole32.dll
0x77120000	0x8c000	0x1		C:\WINDOWS\system32\OLEAUT32.dll
0x77be0000	0x15000	0x1		C:\WINDOWS\system32\MSACM32.dll
0x77c00000	0x8000	0x3		C:\WINDOWS\system32\VERSION.dll
0x7c9c0000	0x814000	0x1		C:\WINDOWS\system32\SHELL32.dll
0x77f60000	0x76000	0x3		C:\WINDOWS\system32\SHLWAPI.dll
0x769c0000	0xb3000	0x1		C:\WINDOWS\system32\USERENV.dll
0x5ad70000	0x38000	0x1		C:\WINDOWS\system32\UxTheme.dll
0x10000000	0x59000	0x1		C:\WINDOWS\system32\mf42u1.dll
0x71ab0000	0x17000	0x2		C:\WINDOWS\system32\WS2_32.dll
0x71aa0000	0x8000	0x1		C:\WINDOWS\system32\WS2HELP.dll
0x71f60000	0x8000	0x1		C:\WINDOWS\system32\snmpapi.dll
0x773d0000	0x102000	0x1		C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600
80_x-ww_a84f1ff9\comctl32.dll				
0x5d090000	0x97000	0x1		C:\WINDOWS\system32\comctl32.dll
0x77b40000	0x22000	0x1		C:\WINDOWS\system32\Apphelp.dll

# Procesos su path y comandos **dlllist**, **cmdscan** y **consoles**

**dlllist** no muestra comandos ejecutados solo librerías entre las que aparecía mfc42ul.dll sospechosa. Hacemos ahora **cmdscan** para saber que hizo cmd.exe. Alguien está comprobando si un servicio llamado **malware** está funcionando, primero lo escribe mal.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 cmdscan
Volatility Foundation Volatility Framework 2.6.1
*****
CommandProcess: csrss.exe Pid: 608
CommandHistory: 0x11132d8 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 2 LastAdded: 1 LastDisplayed: 1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x4c4
Cmd #0 @ 0x4e1eb8: sc query malwar
Cmd #1 @ 0x11135e8: sc query malware
```



# Últimos comandos consolas

El comando `consoles` nos da la solución. A la segunda lo escribe bien.

Resultado: hay un servicio llamado `malware` corriendo como driver de kernel, lo cual es muy grave. Type 1 Kernel Driver, tiene acceso al núcleo del sistema, se oculta. State 4 Running está activo en ese momento. Ignores Shutdown no se detiene aunque se apague el sistema, persistencia.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 consoles
Volatility Foundation Volatility Framework 2.6.1
*****
ConsoleProcess: csrss.exe Pid: 608
Console: 0x4e2370 CommandHistorySize: 50
HistoryBufferCount: 2 HistoryBufferMax: 4
OriginalTitle: %SystemRoot%\system32\cmd.exe
Title: C:\WINDOWS\system32\cmd.exe
AttachedProcess: cmd.exe Pid: 544 Handle: 0x4c4
---
CommandHistory: 0x1113498 Application: sc.exe Flags:
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x0
---
CommandHistory: 0x11132d8 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 2 LastAdded: 1 LastDisplayed: 1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x4c4
Cmd #0 at 0x4e1eb8: sc query malwar
Cmd #1 at 0x11135e8: sc query malware
---
Screen 0x4e2a70 X:80 Y:300
Dump:
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>sc query malware
[SC] EnumQueryServicesStatus:OpenService FAILED 1060:

The specified service does not exist as an installed service.
```

The specified service does not exist as an installed service.

C:\Documents and Settings\Administrator>sc query malware

```
SERVICE_NAME: malware
        TYPE               : 1  KERNEL_DRIVER
        STATE                : 4  RUNNING
                           (STOPPABLE,NOT_PAUSABLE,IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE       : 0  (0x0)
        SERVICE_EXIT_CODE   : 0  (0x0)
        CHECKPOINT           : 0x0
        WAIT_HINT            : 0x0
```

C:\Documents and Settings\Administrator>

```
(javi@kali)-[~/volatility]
```

# Variables de entorno del sistema envvars

El comando **envvars** nos revela las variables de entorno del proceso cmd.exe (PID 544). Username Administrator, User Domian Generalle, Logon Server \\Generalle. Es decir el proceso lo ejecuto el usuario Administrador, en una sesion local Console no remota.

```
400 wuauclt.exe 0x00010000 windir C:\WINDOWS
544 cmd.exe 0x00010000 ALLUSERSPROFILE C:\Documents and Settings\All Users
544 cmd.exe 0x00010000 APPDATA C:\Documents and Settings\Administrator\Application Data
544 cmd.exe 0x00010000 CLIENTNAME Console
544 cmd.exe 0x00010000 CommonProgramFiles C:\Program Files\Common Files
544 cmd.exe 0x00010000 COMPUTERNAME GENERALLEE
544 cmd.exe 0x00010000 ComSpec C:\WINDOWS\system32\cmd.exe
544 cmd.exe 0x00010000 FP_NO_HOST_CHECK NO
544 cmd.exe 0x00010000 HOMEDRIVE C:
544 cmd.exe 0x00010000 HOMEPATH \Documents and Settings\Administrator
544 cmd.exe 0x00010000 J2D_D3D false
544 cmd.exe 0x00010000 LOGONSERVER \\GENERALLEE
544 cmd.exe 0x00010000 NUMBER_OF_PROCESSORS 1
544 cmd.exe 0x00010000 OS Windows_NT
544 cmd.exe 0x00010000 Path C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem
544 cmd.exe 0x00010000 PATHEXT .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
544 cmd.exe 0x00010000 PROCESSOR_ARCHITECTURE x86
544 cmd.exe 0x00010000 PROCESSOR_IDENTIFIER x86 Family 6 Model 42 Stepping 7, GenuineIntel
544 cmd.exe 0x00010000 PROCESSOR_LEVEL 6
544 cmd.exe 0x00010000 PROCESSOR_REVISION 2a07
544 cmd.exe 0x00010000 ProgramFiles C:\Program Files
544 cmd.exe 0x00010000 PROMPT $P$G
544 cmd.exe 0x00010000 SESSIONNAME Console
544 cmd.exe 0x00010000 SystemDrive C:
544 cmd.exe 0x00010000 SystemRoot C:\WINDOWS
544 cmd.exe 0x00010000 TEMP C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp
544 cmd.exe 0x00010000 TMP C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp
544 cmd.exe 0x00010000 USERDOMAIN GENERALLEE
544 cmd.exe 0x00010000 USERNAME Administrator
544 cmd.exe 0x00010000 USERPROFILE C:\Documents and Settings\Administrator
544 cmd.exe 0x00010000 windir C:\WINDOWS
```

# Las conexiones del Host **connections**

El plugin **connections** no mostró conexiones activas en el momento del volcado. Puede ser por limitaciones del sistema, que las conexiones fueron cerradas justo antes, o que el malware haya usado técnicas de evasión. netscan no funcionó.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 netscan
Volatility Foundation Volatility Framework 2.6.1
ERROR : volatility.debug : This command does not support the profile WinXPSP2x86
```

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 connections
Volatility Foundation Volatility Framework 2.6.1
```

Offset(V)	Local Address	Remote Address	Pid
-----------	---------------	----------------	-----

```
(javi@kali)-[~/volatility]
$
```

# Los sockets del host sockscan

El plugin **sockscan** reveló varios sockets abiertos en el sistema. El Proceso `explorer.exe` PID 1956 (padre del proceso `cmd.exe` PID 544) tenía un socket TCP en el puerto 1026, lo que indica que podía recibir conexiones.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 sockscan
Volatility Foundation Volatility Framework 2.6.1
```

Offset(P)	PID	Port	Proto	Protocol	Address	Create Time
0x01796a78	688	500	17	UDP	0.0.0.0	2011-10-10 17:04:00 UTC+0000
0x018118d8	4	445	17	UDP	0.0.0.0	2011-10-10 17:03:55 UTC+0000
0x0186a008	964	1029	17	UDP	127.0.0.1	2011-10-10 17:04:42 UTC+0000
0x01887e98	1616	1025	6	TCP	127.0.0.1	2011-10-10 17:04:01 UTC+0000
0x0194fe98	1148	1900	17	UDP	127.0.0.1	2011-10-10 17:04:41 UTC+0000
0x019517e8	964	123	17	UDP	127.0.0.1	2011-10-10 17:04:00 UTC+0000
0x01953008	688	4500	17	UDP	0.0.0.0	2011-10-10 17:04:00 UTC+0000
0x01953b20	688	0	255	Reserved	0.0.0.0	2011-10-10 17:04:00 UTC+0000
0x0197e3c0	1956	1026	6	TCP	0.0.0.0	2011-10-10 17:04:39 UTC+0000
0x01a328d8	916	135	6	TCP	0.0.0.0	2011-10-10 17:03:59 UTC+0000
0x01addc08	4	445	6	TCP	0.0.0.0	2011-10-10 17:03:55 UTC+0000



# Volcado del proceso sospechoso procdump

Con el comando procdump obtenemos un archivo que contiene la memoria del proceso cmd.exe PID 544 para poder analizarlo con strings, pestudio, yara, capa, virustotal etc.

```
(javi@kali)-[~/volatility]
```

```
$ mkdir dump
```

```
(javi@kali)-[~/volatility]
```

```
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 procdump -p 544 -D ./dump/
```

```
Volatility Foundation Volatility Framework 2.6.1
```

Process(V)	ImageBase	Name	Result
------------	-----------	------	--------

0x817a34b0	0x4ad00000	cmd.exe	OK: executable.544.exe
------------	------------	---------	------------------------

# Extracción de cadenas strings

Se extrajeron las cadenas del volcado mediante `strings`. El archivo resultante se analizó y se encontraron comandos como `malloc` `calloc` `memcpy` de C, de la librería `msvcrt.dll`. También se encontraron funciones como `ImpersonateLoggedOnUser`, `RevertToSelf`, `GetFileSecurityW` etc. Suplantación de identidad, modificación de registro, ejecución de comandos, cargas de dll como `ADVAPI32.dll`, `SHELL32.dll` y `MPR.dll`, sugieren que el proceso tenía capacidades para manipular el entorno. `GetObjectInformationW` intentando manipular sesiones. `VirtualQuery` y `RaiseException` detectan entornos de análisis y evaden sandboxes.

```
(javi@kali)-[~/volatility]
$ strings ./dump/executable.544.exe > strings_cmd.txt
(javi@kali)-[~/volatility]
```

```
QRPh
ADVAPI32.dll
SHELL32.dll
MPR.dll
SaferRecordEventLogEntry
ImpersonateLoggedOnUser
SaferCloseLevel
SaferComputeTokenFromLevel
SaferIdentifyLevel
RevertToSelf
RegQueryValueW
RegEnumKeyW
RegDeleteKeyW
RegSetValueW
RegCloseKey
RegQueryValueExW
RegOpenKeyW
RegSetValueExW
RegCreateKeyExW
CreateProcessAsUserW
RegOpenKeyExW
FreeSid
LookupAccountSidW
GetSecurityDescriptorOwner
GetFileSecurityW
ShellExecuteExW
SHChangeNotify
WNetCancelConnection2W
WNetGetConnectionW
WNetAddConnection2W
msvcrt.dll
KERNEL32.dll
USER32.dll
```

# Análisis con Yara, me creo una regla propia

Creamos un archivo .yar con una regla YARA personalizada para detectar actividad maliciosa. Una de las cadenas definidas en la regla, como `sc query malware` y `CreateProcessAsUserW` está presente en el binario analizado.

```
Archivo Acciones Editar Vista Ayuda
GNU nano 8.4
rule cmd_malware_behavior
{
    meta:
        author = "Javi"
        description = "Detecta actividad sospechosa en cmd.exe"
        date = "2025-09-17"
    strings:
        $a = "sc query malware"
        $b = "CreateProcessAsUserW"
        $c = "ImpersonateLoggedOnUser"
        $d = "ShellExecuteExW"
        $e = "RegSetValueExW"
    condition:
        any of them
}
```

```
(javi@kali)-[~/volatility]
$ yara yara_rules.yar ./dump/executable.544.exe
cmd_malware_behavior ./dump/executable.544.exe
```

```
(javi@kali)-[~/volatility]
```

# Identificadores de sesión sessions

El análisis mediante el plugin sessions reveló que el proceso cmd.exe fue ejecutado dentro de la sesión principal del sistema iD 0, junto con procesos legítimos como explorer.exe que además es su padre. Es decir la actividad maliciosa se originó desde el entorno del usuario legítimo Administrador y no desde una sesión remota. Alguien desde dentro. ID 0, 20 procesos.

```
(javi@kali) - [~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 sessions
Volatility Foundation Volatility Framework 2.6.1
*****
Session(V): f9eb6000 ID: 0 Processes: 20
PagedPoolStart: bc000000 PagedPoolEnd bc3fffff
Process: 608 csrss.exe 2011-10-10 17:03:58 UTC+0000
Process: 632 winlogon.exe 2011-10-10 17:03:58 UTC+0000
Process: 676 services.exe 2011-10-10 17:03:58 UTC+0000
Process: 688 lsass.exe 2011-10-10 17:03:58 UTC+0000
Process: 832 vmacthlp.exe 2011-10-10 17:03:59 UTC+0000
Process: 848 svchost.exe 2011-10-10 17:03:59 UTC+0000
Process: 916 svchost.exe 2011-10-10 17:03:59 UTC+0000
Process: 964 svchost.exe 2011-10-10 17:03:59 UTC+0000
Process: 1020 svchost.exe 2011-10-10 17:03:59 UTC+0000
Process: 1148 svchost.exe 2011-10-10 17:04:00 UTC+0000
Process: 1260 spoolsv.exe 2011-10-10 17:04:00 UTC+0000
Process: 1444 VMwareService.e 2011-10-10 17:04:00 UTC+0000
Process: 1616 alg.exe 2011-10-10 17:04:01 UTC+0000
Process: 1920 wscntfy.exe 2011-10-10 17:04:39 UTC+0000
Process: 1956 explorer.exe 2011-10-10 17:04:39 UTC+0000
Process: 184 VMwareTray.exe 2011-10-10 17:04:41 UTC+0000
Process: 192 VMwareUser.exe 2011-10-10 17:04:41 UTC+0000
Process: 228 reader_sl.exe 2011-10-10 17:04:41 UTC+0000
Process: 400 wuaucflt.exe 2011-10-10 17:04:46 UTC+0000
Process: 544 cmd.exe 2011-10-10 17:06:42 UTC+0000
Image: 0x817921c8, Address bf800000, Name: win32k.sys
Image: 0x813cd0e0, Address bf9c1000, Name: dxg.sys
Image: 0x8148c5c8, Address bf9d3000, Name: vmx_fb.dll
Image: 0xbf7f009c, Address c07cf5b0, Name:
```



# Los privilegios con los que se ejecuta privs

El análisis de privilegios del proceso cmd.exe mediante el plugin privs reveló que contaba con múltiples privilegios elevados. SeDebugPrivilege, SeimpersonatePrivilege y SeLoadDriverPrivilege. Estos permiten inspeccionar otros procesos, suplantar identidades y cargar drivers en el kernel. Capacidad para realizar acciones maliciosas con alto impacto en el sistema.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 privs -p 544
Volatility Foundation Volatility Framework 2.6.1
```

Pid	Process	Value	Privilege	Attributes	Description
544	cmd.exe	23	SeChangeNotifyPrivilege	Present,Enabled,Default	Receive notifications of changes to files or directories
544	cmd.exe	8	SeSecurityPrivilege	Present	Manage auditing and security log
544	cmd.exe	17	SeBackupPrivilege	Present	Backup files and directories
544	cmd.exe	18	SeRestorePrivilege	Present	Restore files and directories
544	cmd.exe	12	SeSystemTimePrivilege	Present	Change the system time
544	cmd.exe	19	SeShutdownPrivilege	Present	Shut down the system
544	cmd.exe	24	SeRemoteShutdownPrivilege	Present	Force shutdown from a remote system
544	cmd.exe	9	SeTakeOwnershipPrivilege	Present	Take ownership of files/objects
544	cmd.exe	20	SeDebugPrivilege	Present	Debug programs
544	cmd.exe	22	SeSystemEnvironmentPrivilege	Present	Edit firmware environment values
544	cmd.exe	11	SeSystemProfilePrivilege	Present	Profile system performance
544	cmd.exe	13	SeProfileSingleProcessPrivilege	Present	Profile a single process
544	cmd.exe	14	SeIncreaseBasePriorityPrivilege	Present	Increase scheduling priority
544	cmd.exe	10	SeLoadDriverPrivilege	Present,Enabled	Load and unload device drivers
544	cmd.exe	15	SeCreatePagefilePrivilege	Present	Create a pagefile
544	cmd.exe	5	SeIncreaseQuotaPrivilege	Present	Increase quotas
544	cmd.exe	25	SeUndockPrivilege	Present,Enabled	Remove computer from docking station
544	cmd.exe	28	SeManageVolumePrivilege	Present	Manage the files on a volume
544	cmd.exe	29	SeImpersonatePrivilege	Present,Enabled,Default	Impersonate a client after authentication
544	cmd.exe	30	SeCreateGlobalPrivilege	Present,Enabled,Default	Create global objects

# Acceso que tiene el proceso sospechoso handles

El análisis de acceso del proceso mediante el plugin handles reveló que el proceso tenía abiertos múltiples recursos del sistema. Incluyendo archivos de perfil de usuario, claves de registro y objetos de sincronización. Destaca el acceso a claves como DRIVER32, que se podría usar para cargar componentes maliciosos. Y a objetos como ShimCacheMutex que sugiere manipulación de la caché.

```
python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 handles -p 544
```

Volatility Foundation Volatility Framework 2.6.1

Offset(V)	Pid	Handle	Access	Type	Details
0xe1000080	544	0x4	0xf0003	KeyedEvent	CritSecOutOfMemoryEvent
0xe13c7410	544	0x8	0x3	Directory	KnownDlls
0x816e8db8	544	0xc	0x100020	File	\Device\HarddiskVolume1\Documents and Settings\Administrator
0xe1555270	544	0x10	0xf001f	Section	
0xe16b2c38	544	0x14	0xf000f	Directory	Windows
0xe1c72248	544	0x18	0x21f0001	Port	
0x815b6160	544	0x1c	0x21f0003	Event	
0x81882080	544	0x20	0x1f0003	Event	
0x815c7138	544	0x24	0xf037f	WindowStation	WinSta0
0xe1580bf0	544	0x28	0x2000f	Directory	BaseNamedObjects
0x815c7138	544	0x2c	0xf037f	WindowStation	WinSta0
0x816799a0	544	0x30	0xf01ff	Desktop	Default
0xe1c822d0	544	0x34	0x20f003f	Key	MACHINE
0x81633f58	544	0x38	0x1f0003	Event	
0x813bcba0	544	0x3c	0x100003	Semaphore	
0x81804d80	544	0x40	0x100003	Semaphore	
0xe17f0660	544	0x44	0x20019	Key	MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION\DRIVERS32
0x815f6af8	544	0x48	0x100001	File	\Device\KsecDD
0x817ebe58	544	0x4c	0x1f0003	Event	
0x817ebe88	544	0x50	0x1f0003	Event	
0xe1c9be48	544	0x54	0x20019	Key	MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION\DRIVERS32
0x818dfe78	544	0x58	0x1f0003	Semaphore	shell.{A48F1A32-A340-11D1-BC68-00A0C90312E1}
0xe18a5d80	544	0x5c	0x20f003f	Key	USER\S-1-5-21-839522115-73586283-214712571-500
0x818a6798	544	0x60	0x100020	File	\Device\HarddiskVolume1\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls
.2180_x-ww_a84f1fff9					
0x81605890	544	0x64	0x1f0003	Event	userenv: User Profile setup event
0xe1ca0f98	544	0x68	0x20019	Key	MACHINE\SYSTEM\CONTROLSET001\CONTROL\NLS\LOCALE
0xe1cb8b80	544	0x6c	0x20019	Key	MACHINE\SYSTEM\CONTROLSET001\CONTROL\NLS\LOCALE\ALTERNATE SORTS
0xe1bb78d8	544	0x70	0x20019	Key	MACHINE\SYSTEM\CONTROLSET001\CONTROL\NLS\LANGUAGE_GROUPS
0x8192ad58	544	0x7c	0x120001	Mutant	ShimCacheMutex
0xe17a6198	544	0x80	0x2	Section	ShimSharedMemory

# Listado de servicios y sospechoso svcscan

El plugin svcscan reveló la presencia de un servicio llamado malware, display name malware2. Configurado como service kernel driver y running. Se inicia automáticamente al arrancar el sistema, service system start, y tiene la ruta \Driver\malware. Es un driver malicioso diseñado para establecer persistencia, evadir detección y manipular el sistema a bajo nivel.

```
Offset: 0x6f53b0
Order: 96
Start: SERVICE_AUTO_START
Process ID: 1148
Service Name: LmHosts
Display Name: TCP/IP NetBIOS Helper
Service Type: SERVICE_WIN32_SHARE_PROCESS
Service State: SERVICE_RUNNING
Binary Path: C:\WINDOWS\system32\svchost.exe -k LocalService

Offset: 0x6f5440
Order: 97
Start: SERVICE_SYSTEM_START
Process ID: -
Service Name: malware
Display Name: malware2
Service Type: SERVICE_KERNEL_DRIVER
Service State: SERVICE_RUNNING
Binary Path: \Driver\malware

Offset: 0x6f54d0
Order: 98
Start: SERVICE_DISABLED
Process ID: -
Service Name: Messenger
Display Name: Messenger
Service Type: SERVICE_WIN32_SHARE_PROCESS
Service State: SERVICE_STOPPED
Binary Path: -

Offset: 0x6f5560
Order: 99
Start: SERVICE_SYSTEM_START
Process ID: -
Service Name: mnmd
Display Name: mnmd
Service Type: SERVICE_KERNEL_DRIVER
Service State: SERVICE_RUNNING
```

# Analisis de modulos modules y modscan

El análisis de módulos con modules y modscan no revelo ningún componente llamado malware, a pesar de que el servicio fue identificado como service kernel driver. Esto puede ser porque pudo haberse usado técnicas de evasión o carga manual o renombramiento. El único sospechoso es winsys32.sys que no es un driver legítimo.

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 modules
Volatility Foundation Volatility Framework 2.6.1
```

Offset(V)	Name	Base	Size	File
0x819fc3a0	ntoskrnl.exe	0x804d7000	0x1f6280	\WINDOWS\system32\ntkrnlpa.exe
0x819fc338	hal.dll	0x806ce000	0x20380	\WINDOWS\system32\hal.dll

```
(javi@kali)-[~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 modscan
Volatility Foundation Volatility Framework 2.6.1
```

Offset(P)	Name	Base	Size	File
0x00000000015d0918	win32k.sys	0xbf800000	0x1c1000	\SystemRoot\System32\win32k.sys
0x0000000001680630	netbios.sys	0xf9b4c000	0x9000	\SystemRoot\system32\DRIVERS\netbios.sys
0x000000000168d0d0	rdpdr.sys	0xf94a7000	0x31000	\SystemRoot\system32\DRIVERS\rdpdr.sys
0x000000000168d230	mssmbios.sys	0xf9e6c000	0x4000	\SystemRoot\system32\DRIVERS\mssmbios.sys

0x0000000001920b58	srv.sys	0xf0c5e000	0x53000	\SystemRoot\system32\DRIVERS\srv.sys
0x00000000019220e0	dump_scsiport.sys	0xf9595000	0x4000	\SystemRoot\System32\Drivers\dump_diskdump.sys
0x0000000001925c68	mouclass.sys	0xf9c64000	0x6000	\SystemRoot\system32\DRIVERS\mouclass.sys
0x0000000001925df8	vmxnet.sys	0xf9c7c000	0x8000	\SystemRoot\system32\DRIVERS\vmxnet.sys
0x000000000197f228	winsys32.sys	0xf9eb4000	0x2000	\\?\C:\WINDOWS\system32\drivers\winsys32.sys
0x0000000001983008	HTTP.sys	0xf07f3000	0x41000	\SystemRoot\System32\Drivers\HTTP.sys
0x0000000001996b38	vmmemctl.sys	0xf9ec8000	0x2000	\\?\C:\Program Files\VMware\VMware Tools\Drivers\memctl\vmemctl.sys
0x00000000019980d8	usbccgp.sys	0xf9cd4000	0x8000	\SystemRoot\system32\DRIVERS\usbccgp.sys
0x00000000019a5ba8	rasacd.sys	0xf9689000	0x3000	\SystemRoot\system32\DRIVERS\rasacd.sys
0x00000000019a8668	Fs_Rec.SYS	0xf9eaa000	0x2000	\SystemRoot\System32\Drivers\Fs_Rec.SYS



# Conclusión

## Evidencias recolectadas

**Proceso Cmd.exe (PID 544).** Ejecutado desde la sesión de usuario administrador. Privilegios elevados **SedebugPrivilege**, **SeimpersonatePrivilege**, etc. Acceso a claves de registro sensibles y objetos del sistema.

**Servicio** malicioso detectado. Malware display name **malware2**. Tipo service kernel driver. Estado running. Ruta **\Driver\malware**.

**Modulo** oculto en memoria. Winsys32.sys detectado con modscan. No aparece en modules, posible evasión.

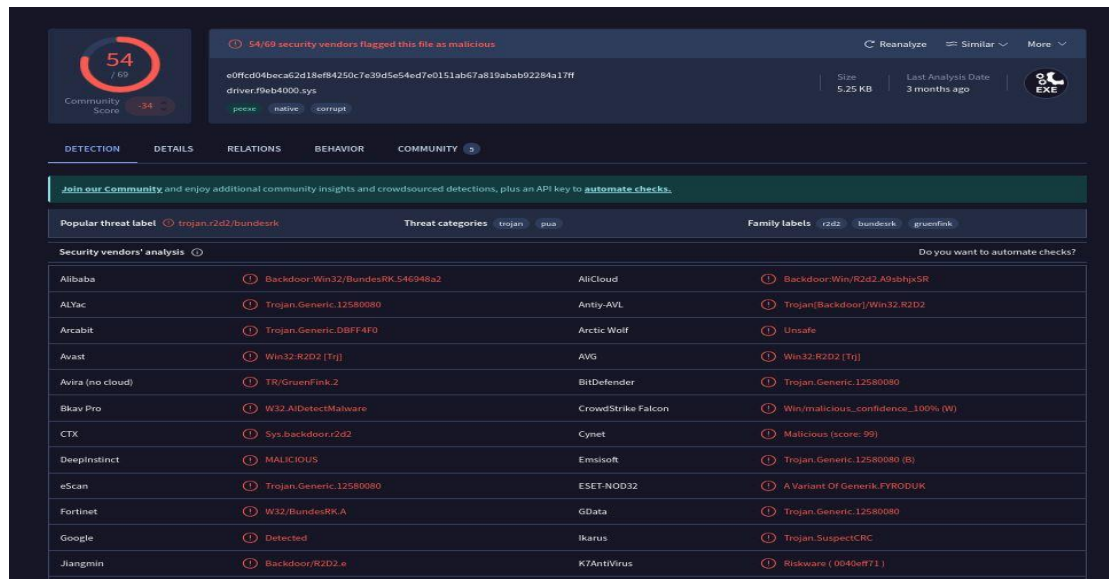
**Volcado** profundo. Binario extraído con procdump. Cadenas sospechosas **CretaeProcessAsUser**, **ShellExecuteExw**, etc. Regla **YARA** personalizada positiva **cmd\_malware\_behavior**.

**Conclusión** final. Sistema comprometido por un **malware en el núcleo**. La combinación de privilegios elevados, servicio malicioso en el kernel y un módulo oculto confirma una **infección avanzada con capacidad para persistir y manipular a bajo nivel**.

# Conclusión

Me descargo el binario winsys y lo paso a Virustotal para  
sacar de dudas y ..... boom

```
(javi@kali) - [~/volatility]
$ python2.7 vol.py -f windows.vmem --profile=WinXPSP2x86 moddump -r winsys32.sys -D ./dump/
Volatility Foundation Volatility Framework 2.6.1
Module Base Module Name Result
0x0f9eb4000 winsys32.sys OK: driver.f9eb4000.sys
```



The screenshot shows the VirusTotal analysis interface for a file named 'driver.f9eb4000.sys'. The file is 5.25 KB and was last analyzed 3 months ago. It has a community score of 54/69, with 54 security vendors flagging it as malicious. The file is identified as a Trojan.Generic.12580080. The interface includes tabs for Detection, Details, Relations, Behavior, and Community. A table lists security vendors and their analysis results, showing various detections such as Backdoor.Win32/BundesRK, Trojan.Generic, and Win32/R2D2.

Security vendors' analysis	Do you want to automate checks?
Alibaba	Backdoor.Win32/BundesRK.546948a2
ALYac	Trojan.Generic.12580080
Arcabit	Trojan.Generic.DBFF4F0
Avast	Win32:R2D2 [Trj]
Avira (no cloud)	TR/GreenLink.2
Bkav Pro	W32/AIDetect/Malware
CTX	Sys.backdoor.r2d2
DeepInSight	MALICIOUS
eScan	Trojan.Generic.12580080
Fortinet	W32/BundesRKA
Google	Detected
Jiangmin	Backdoor/R2D2.e