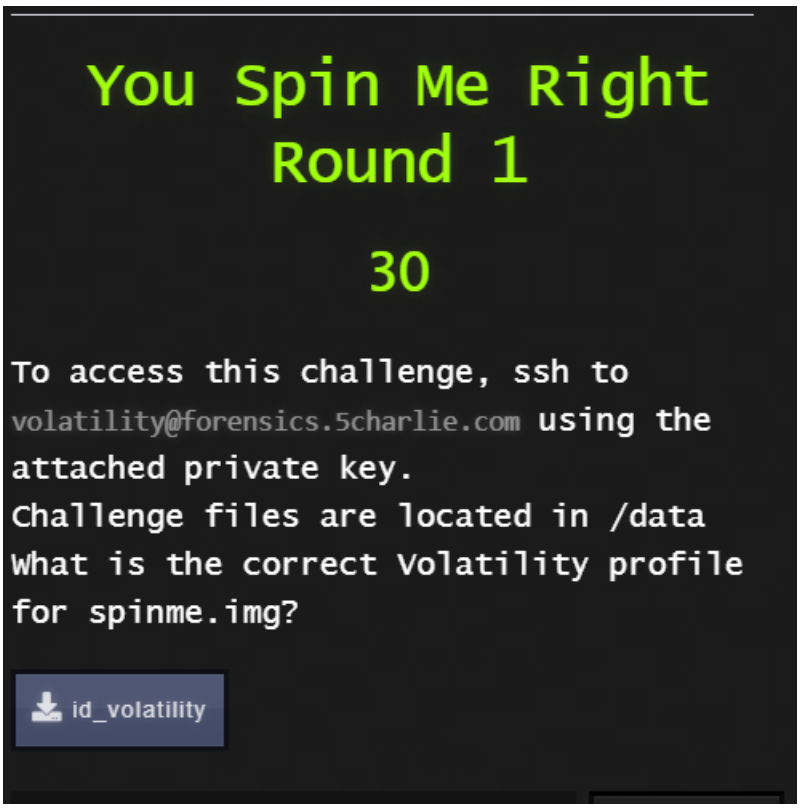
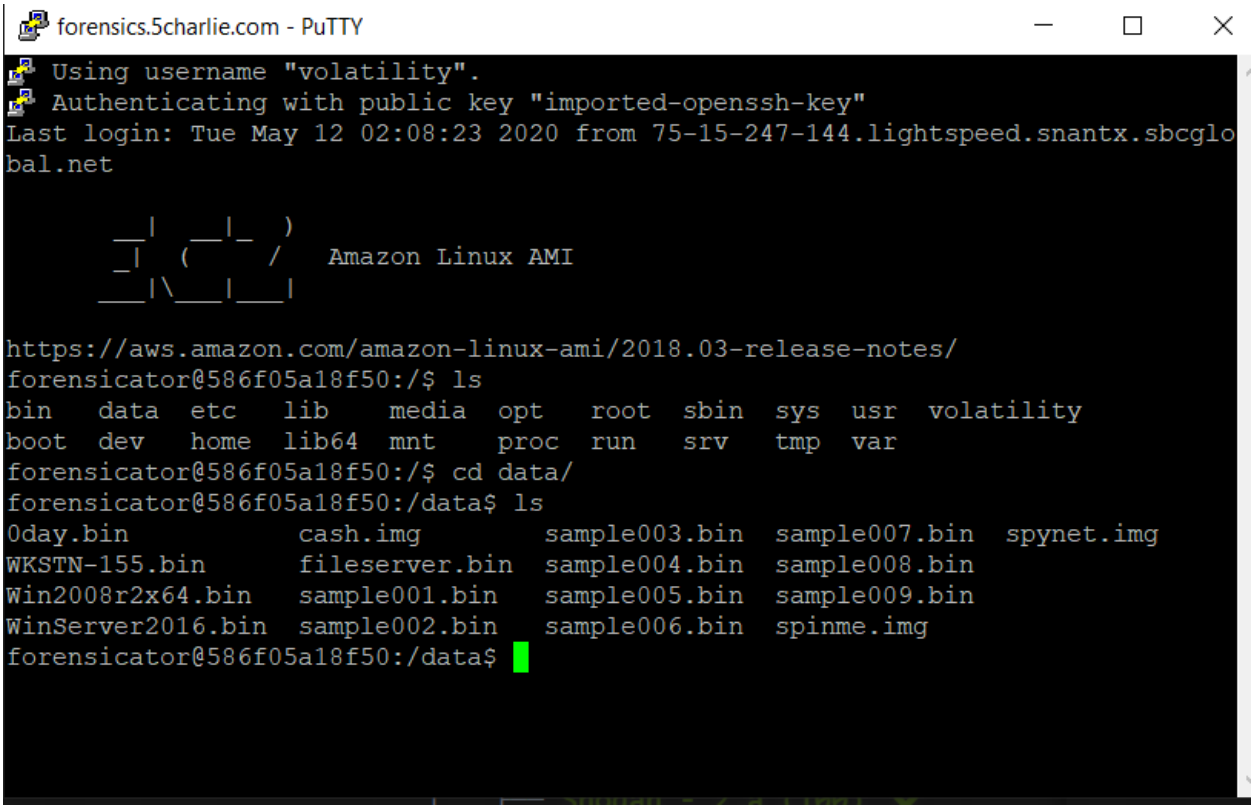
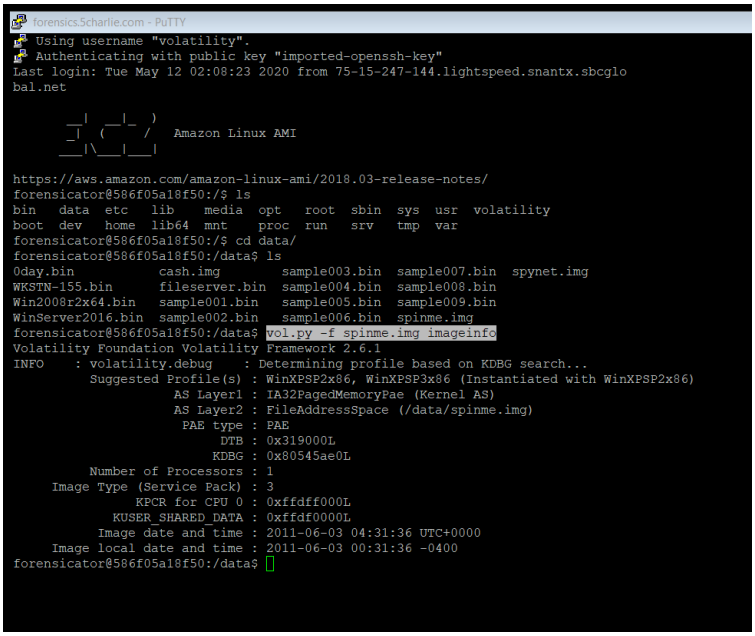


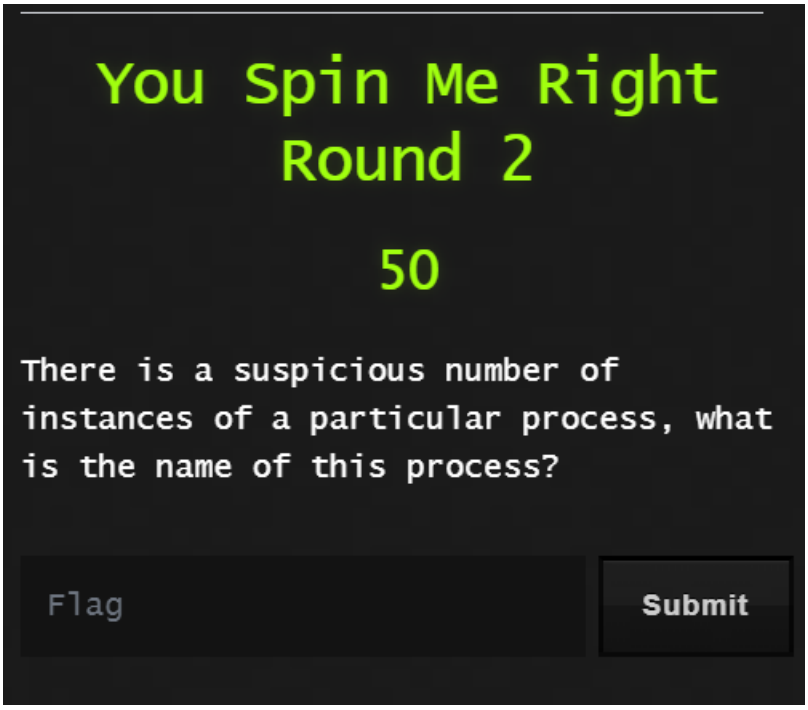
Access the vm through ssh and move into the /data folder



Run vol.py -f spinme.img imageinfo to get the information



Flag: WinXPSP3x86



Run the following command: vol.py -f spinme.img --profile=WinXPSP3x86 pstree

```
forensicator@586f05a18f50:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 pstree
Volatility Foundation Volatility Framework 2.6.1
Name                                                  Pid   PPid   Thds   Hnds Time
-------------------------------------------------- ------ ------ ------ ------ ----
 0x823c8830:System                                      4      0     59    403 1970-01-01 00:00:00 UTC+0000
. 0x820df020:smss.exe                                 376      4      3     19 2010-10-29 17:08:53 UTC+0000
.. 0x821a2da0:csrss.exe                               600    376     11    395 2010-10-29 17:08:54 UTC+0000
.. 0x81da5650:winlogon.exe                            624    376     19    570 2010-10-29 17:08:54 UTC+0000
... 0x82073020:services.exe                           668    624     21    431 2010-10-29 17:08:54 UTC+0000
.... 0x81fe52d0:vmtoolsd.exe                         1664    668      5    284 2010-10-29 17:09:05 UTC+0000
..... 0x81c0cda0:cmd.exe                              968   1664      0 ------ 2011-06-03 04:31:35 UTC+0000
...... 0x81f14938:ipconfig.exe                        304    968      0 ------ 2011-06-03 04:31:35 UTC+0000
.... 0x822843e8:svchost.exe                          1032    668     61   1169 2010-10-29 17:08:55 UTC+0000
..... 0x822b9a10:wuauclt.exe                          976   1032      3    133 2010-10-29 17:12:03 UTC+0000
..... 0x820ecc10:wscntfy.exe                         2040   1032      1     28 2010-10-29 17:11:49 UTC+0000
.... 0x81e61da0:svchost.exe                           940    668     13    312 2010-10-29 17:08:55 UTC+0000
.... 0x81db8da0:svchost.exe                           856    668     17    193 2010-10-29 17:08:55 UTC+0000
..... 0x81fa5390:wmiprvse.exe                        1872    856      5    134 2011-06-03 04:25:58 UTC+0000
.... 0x821a0568:VMUpgradeHelper                      1816    668      3     96 2010-10-29 17:09:08 UTC+0000
.... 0x81fee8b0:spoolsv.exe                          1412    668     10    118 2010-10-29 17:08:56 UTC+0000
.... 0x81ff7020:svchost.exe                          1200    668     14    197 2010-10-29 17:08:55 UTC+0000
.... 0x81c47c00:lsass.exe                            1928    668      4     65 2011-06-03 04:26:55 UTC+0000
.... 0x81e18b28:svchost.exe                          1080    668      5     80 2010-10-29 17:08:55 UTC+0000
.... 0x8205ada0:alg.exe                               188    668      6    107 2010-10-29 17:09:09 UTC+0000
.... 0x823315d8:vmacthlp.exe                          844    668      1     25 2010-10-29 17:08:55 UTC+0000
.... 0x81e0eda0:jqs.exe                              1580    668      5    148 2010-10-29 17:09:05 UTC+0000
.... 0x81c498c8:lsass.exe                             868    668      2     23 2011-06-03 04:26:55 UTC+0000
.... 0x82279998:imapi.exe                             756    668      4    116 2010-10-29 17:11:54 UTC+0000
... 0x81e70020:lsass.exe                              680    624     19    342 2010-10-29 17:08:54 UTC+0000
 0x820ec7e8:explorer.exe                             1196   1728     16    582 2010-10-29 17:11:49 UTC+0000
. 0x81c543a0:Procmon.exe                              660   1196     13    189 2011-06-03 04:25:56 UTC+0000
. 0x81e86978:TSVNCache.exe                            324   1196      7     54 2010-10-29 17:11:49 UTC+0000
. 0x81e6b660:VMwareUser.exe                          1356   1196      9    251 2010-10-29 17:11:50 UTC+0000
. 0x8210d478:jusched.exe                             1712   1196      1     26 2010-10-29 17:11:50 UTC+0000
. 0x81fc5da0:VMwareTray.exe                          1912   1196      1     50 2010-10-29 17:11:50 UTC+0000
forensicator@586f05a18f50:/data$
```

There are 3 lsass.exe, at most there should only be 1

Flag: lsass.exe

You Spin Me Right Round 3

50

How many DLL files are loaded into or referenced by the legitimate version of the process you found in You Spin Me Right Round 2?

Run the dlllist on pid 638 (legitimate version that spawned from the correct parent process)

```
0x5d090000    0x9a000        0x1                                C:\WINDOWS\system32\comctl32.dll
forensicator@f3273919d790:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 dlllist -p 1928 | wc -l
Volatility Foundation Volatility Framework 2.6.1
35
forensicator@f3273919d790:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 dlllist -p 680 | wc -l
Volatility Foundation Volatility Framework 2.6.1
64
forensicator@f3273919d790:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 dlllist -p 680
Volatility Foundation Volatility Framework 2.6.1
************************************************************************
lsass.exe pid:    680
Command line : C:\WINDOWS\system32\lsass.exe
Service Pack 3

Base             Size  LoadCount LoadTime                       Path
---------- ---------- ---------- ------------------------------ ----
0x01000000     0x6000     0xffff                                C:\WINDOWS\system32\lsass.exe
0x7c900000    0xaf000     0xffff                                C:\WINDOWS\system32\ntdll.dll
0x7c800000    0xf6000     0xffff                                C:\WINDOWS\system32\kernel32.dll
0x77dd0000    0x9b000     0xffff                                C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000    0x92000     0xffff                                C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000    0x11000     0xffff                                C:\WINDOWS\system32\Secur32.dll
0x75730000    0xb5000     0xffff                                C:\WINDOWS\system32\LSASRV.dll
0x71b20000    0x12000     0xffff                                C:\WINDOWS\system32\MPR.dll
0x7e410000    0x91000     0xffff                                C:\WINDOWS\system32\USER32.dll
0x77f10000    0x49000     0xffff                                C:\WINDOWS\system32\GDI32.dll
0x77b20000    0x12000     0xffff                                C:\WINDOWS\system32\MSASN1.dll
0x77c10000    0x58000     0xffff                                C:\WINDOWS\system32\msvcrt.dll
```

Wc comes in with 64, but we need to chop off the lines at the top or grep for 0x in load count column) to get the correct answer

Flag: 57

You Spin Me Right Round 4

50

The malicious processes were spawned by services.exe, if they were legitimate, what process name would they have been spawned by on a Windows XP system?

The legitimate process spawned with a ppid of 624, or winlogon.exe

```
forensicator@f3273919d790:/data$ vol.py -f spinme.img --profile=W
Volatility Foundation Volatility Framework 2.6.1
Name                                                 Pid   PPid
---------------------------------------------------- ------ ------
 0x823c8830:System                                      4      0
. 0x820df020:smss.exe                                 376      4
.. 0x821a2da0:csrss.exe                               600    376
.. 0x81da5650:winlogon.exe                            624    376
... 0x82073020:services.exe                           668    624
.... 0x81fe52d0:vmtoolsd.exe                         1664    668
..... 0x81c0cda0:cmd.exe                              968   1664
...... 0x81f14938:ipconfig.exe                        304    968
.... 0x822843e8:svchost.exe                          1032    668
..... 0x822b9a10:wuauclt.exe                          976   1032
..... 0x820ecc10:wscntfy.exe                         2040   1032
.... 0x81e61da0:svchost.exe                           940    668
.... 0x81db8da0:svchost.exe                           856    668
..... 0x81fa5390:wmiprvse.exe                        1872    856
.... 0x821a0568:VMUpgradeHelper                      1816    668
.... 0x81fee8b0:spoolsv.exe                          1412    668
.... 0x81ff7020:svchost.exe                          1200    668
.... 0x81c47c00:lsass.exe                            1928    668
.... 0x81e18b28:svchost.exe                          1080    668
.... 0x8205ada0:alg.exe                               188    668
.... 0x823315d8:vmacthlp.exe                          844    668
.... 0x81e0eda0:jqs.exe                              1580    668
.... 0x81c498c8:lsass.exe                             868    668
.... 0x82279998:imapi.exe                             756    668
... 0x81e70020:lsass.exe                              680    624
 0x820ec7e8:explorer.exe                             1196   1728
. 0x81c543a0:Procmon.exe                              660   1196
. 0x81e86978:TSVNCache.exe                            324   1196
. 0x81e6b660:VMwareUser.exe                          1356   1196
. 0x8210d478:jusched.exe                             1712   1196
. 0x81fc5da0:VMwareTray.exe                          1912   1196
```

Flag: winlogon.exe

Run the following command: vol.py -f spinme.img --profile=WinXPSP3x86 connections

vol.py -f spinme.img --profile=WinXPSP3x86 connscan

```
forensicator@586f05a18f50:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 connections
Volatility Foundation Volatility Framework 2.6.1
Offset(V)  Local Address             Remote Address            Pid
---------- ------------------------ ------------------------ ---
forensicator@586f05a18f50:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 netscan
Volatility Foundation Volatility Framework 2.6.1
ERROR   : volatility.debug    : This command does not support the profile WinXPSP3x86
forensicator@586f05a18f50:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 connscan
Volatility Foundation Volatility Framework 2.6.1
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------ ------------------------ ---
forensicator@586f05a18f50:/data$
```

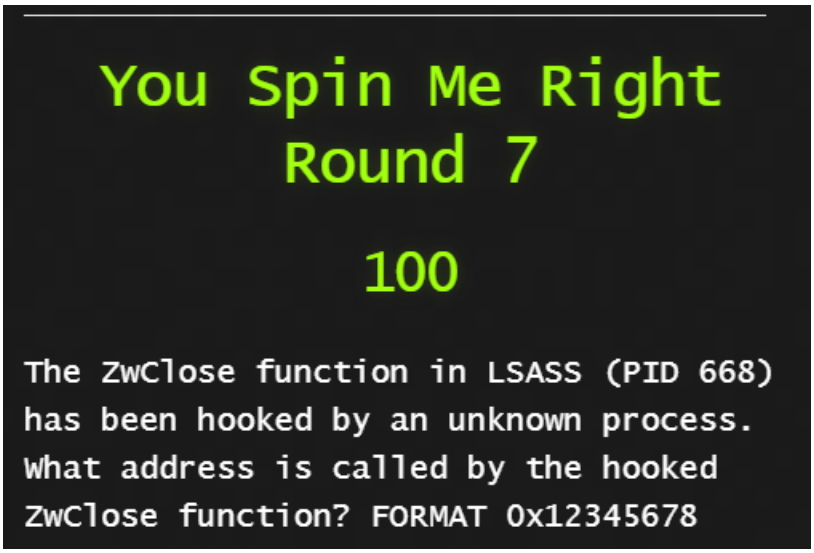
Both came back with 0 network connections

Flag: 0

We can use ldrmodules to show all the tables that load dlls.

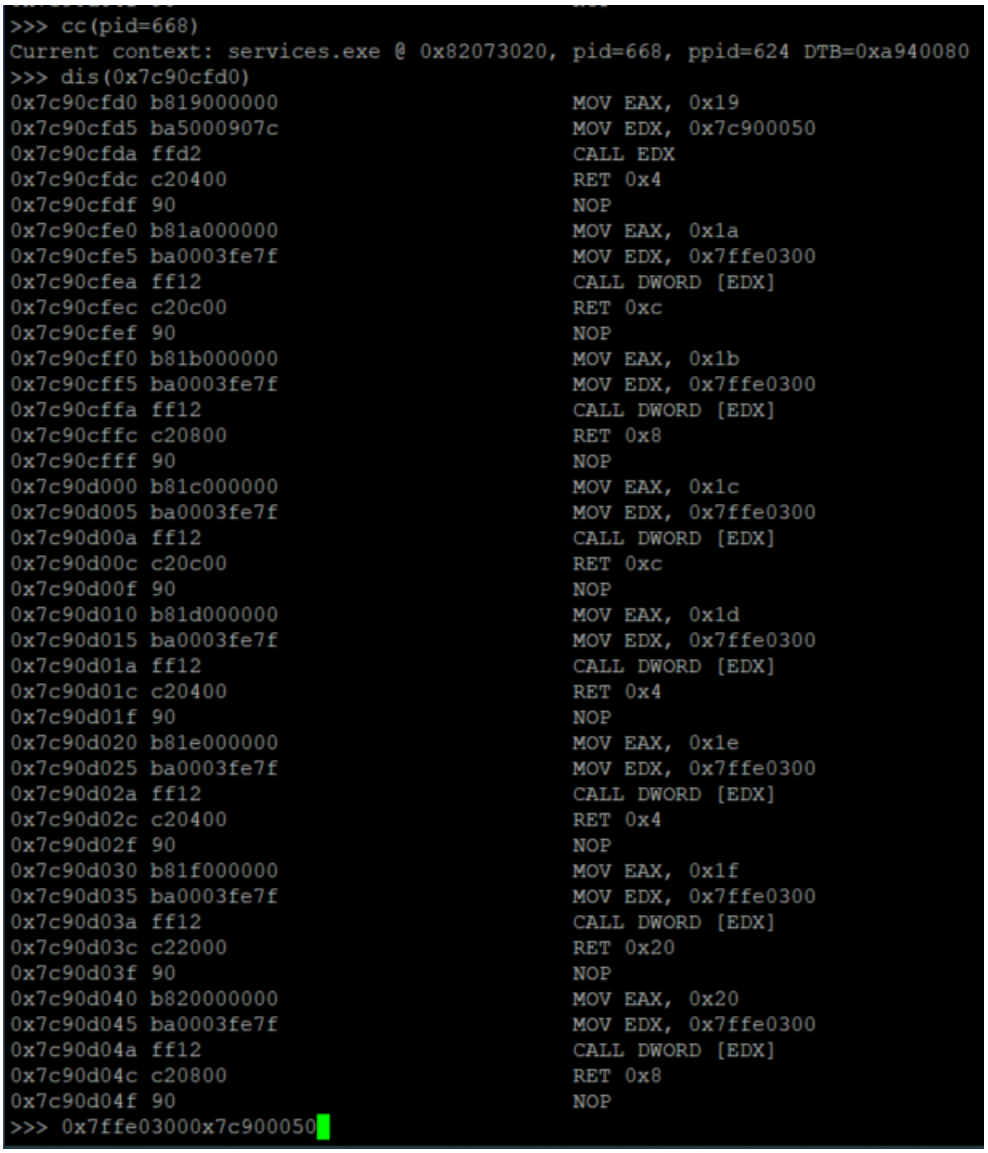
```
forensicator@f3273919d790:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 ldrmodules -p 1928
Volatility Foundation Volatility Framework 2.6.1
Pid      Process            Base       InLoad InInit InMem MappedPath
-------- ------------------ ---------- ------ ------ ----- ----------
    1928 lsass.exe          0x00080000 False  False  False
    1928 lsass.exe          0x7c900000 True   True   True  \WINDOWS\system32\ntdll.dll
    1928 lsass.exe          0x773d0000 True   True   True  \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83\comctl32.dll
    1928 lsass.exe          0x77f60000 True   True   True  \WINDOWS\system32\shlwapi.dll
    1928 lsass.exe          0x771b0000 True   True   True  \WINDOWS\system32\wininet.dll
    1928 lsass.exe          0x77a80000 True   True   True  \WINDOWS\system32\crypt32.dll
    1928 lsass.exe          0x77fe0000 True   True   True  \WINDOWS\system32\secur32.dll
    1928 lsass.exe          0x77c00000 True   True   True  \WINDOWS\system32\version.dll
    1928 lsass.exe          0x01000000 True   False  True
    1928 lsass.exe          0x5b860000 True   True   True  \WINDOWS\system32\netapi32.dll
    1928 lsass.exe          0x77e70000 True   True   True  \WINDOWS\system32\rpcrt4.dll
    1928 lsass.exe          0x71ab0000 True   True   True  \WINDOWS\system32\ws2_32.dll
    1928 lsass.exe          0x71ad0000 True   True   True  \WINDOWS\system32\wsock32.dll
    1928 lsass.exe          0x774e0000 True   True   True  \WINDOWS\system32\ole32.dll
    1928 lsass.exe          0x7e410000 True   True   True  \WINDOWS\system32\user32.dll
    1928 lsass.exe          0x77f10000 True   True   True  \WINDOWS\system32\gdi32.dll
    1928 lsass.exe          0x77120000 True   True   True  \WINDOWS\system32\oleaut32.dll
    1928 lsass.exe          0x76d60000 True   True   True  \WINDOWS\system32\iphlpapi.dll
    1928 lsass.exe          0x769c0000 True   True   True  \WINDOWS\system32\userenv.dll
    1928 lsass.exe          0x7c800000 True   True   True  \WINDOWS\system32\kernel32.dll
    1928 lsass.exe          0x76bf0000 True   True   True  \WINDOWS\system32\psapi.dll
    1928 lsass.exe          0x77c10000 True   True   True  \WINDOWS\system32\msvcrt.dll
    1928 lsass.exe          0x77dd0000 True   True   True  \WINDOWS\system32\advapi32.dll
    1928 lsass.exe          0x7c9c0000 True   True   True  \WINDOWS\system32\shell32.dll
    1928 lsass.exe          0x00870000 True   True   True
    1928 lsass.exe          0x76f20000 True   True   True  \WINDOWS\system32\dnsapi.dll
    1928 lsass.exe          0x5d090000 True   True   True  \WINDOWS\system32\comctl32.dll
    1928 lsass.exe          0x71aa0000 True   True   True  \WINDOWS\system32\ws2help.dll
    1928 lsass.exe          0x77b20000 True   True   True  \WINDOWS\system32\msasn1.dll
```

Flag: 29



Open the volshell and switch over to the pid 668.

```
forensicator@f3273919d790:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 volshell
```

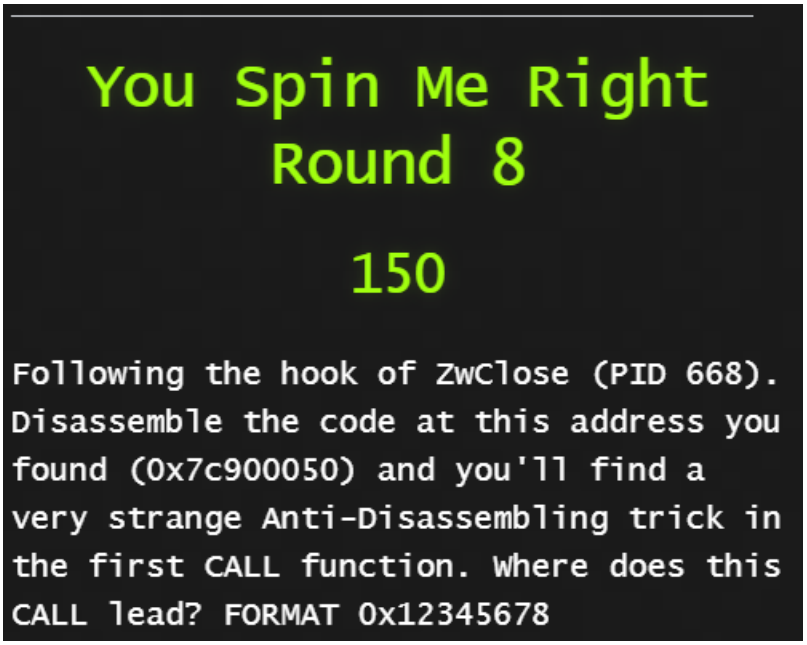


The hint points us to the location of the ZwClose location inside this process, so all we need to do is call that location in a dissembled view.

dis(0x7c90cfd0)

2nd line down gives us our answer

Flag: 0x7c900050



First you need to break into the shell and switch into the context of a process that has been hooked. Then navigate to the hooked API. I'll start with ZwClose which is at 0x7C900050.

```
>>> dis(0x7c900050)
0x7c900050 b203                                MOV DL, 0x3
0x7c900052 eb08                                JMP 0x7c90005c
0x7c900054 b204                                MOV DL, 0x4
0x7c900056 eb04                                JMP 0x7c90005c
0x7c900058 b205                                MOV DL, 0x5
0x7c90005a eb00                                JMP 0x7c90005c
0x7c90005c 52                                  PUSH EDX
0x7c90005d e804000000                          CALL 0x7c900066
0x7c900062 f20094005aff2269                    ADD [EAX+EAX+0x6922ff5a], DL
0x7c90006a 6e                                  OUTS DX, BYTE [ESI]
```

We follow the JMP to 0x7c900066

```
>>> dis(0x7c900066)
0x7c900066 5a                                  POP EDX
0x7c900067 ff22                                JMP DWORD [EDX]
0x7c900069 696e20444f5320                      IMUL EBP, [ESI+0x20], 0x20534f44
0x7c900070 6d                                  INS DWORD [ES:EDI], DX
0x7c900071 6f                                  OUTS DX, DWORD [ESI]
0x7c900072 64652e0d0d0a2400                    OR EAX, 0x240a0d
0x7c90007a 0000                                ADD [EAX], AL
0x7c90007c 0000                                ADD [EAX], AL
0x7c90007e 0000                                ADD [EAX], AL
0x7c900080 084063                              OR [EAX+0x63], AL
0x7c900083 fe4c210d                            DEC BYTE [ECX+0xd]
0x7c900087 ad                                  LODSD
0x7c900088 4c                                  DEC ESP
0x7c900089 210dad4c210d                        AND [0xd214cad], ECX
```

when the CALL at 0x7c90005d is executed, its return address (0x7c900062) is pushed onto the stack. The POP EDX instruction at 0x7c900066 then removes that value from the stack and places it in EDX. At 0x7c900067, EDX is dereferenced and called. So, the pointer being dereferenced is 0x7c900062. The 4 bytes f2009400 @ 0x7c900062. Given endianness, this is actually 0x009400F2 - our official next hop in following the rootkit

Flag: 0x009400F2

You Spin Me Right Round 9

150

From IOC scanning, we've found that this malware is using the MUTEX "{E41362C3-F75C-4ec2-AF49-3CB6BCA591CA}" which reporting has linked to a known malware. When was this mutex likely created?
FORMAT YYYY-MM-DD HH:MM:SS

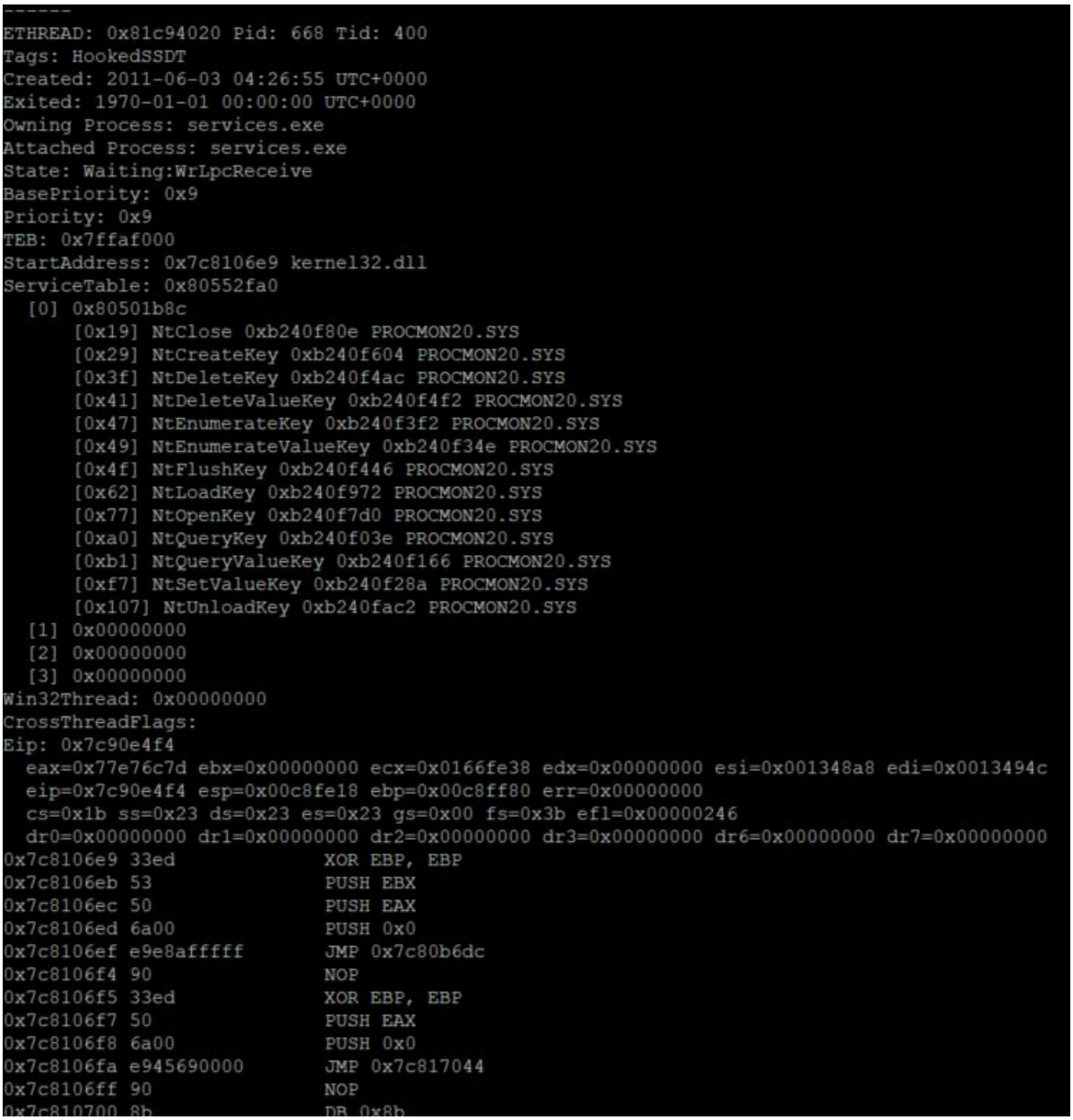
First, we should look for the mutant with handles of the pid 668

vol.py -f spinme.img --profile=WinXPSP3x86 handles | grep -A 10 -B 10 -i e4136

this will take us to any lines that have the mutant listed and the 10 lines before and after.

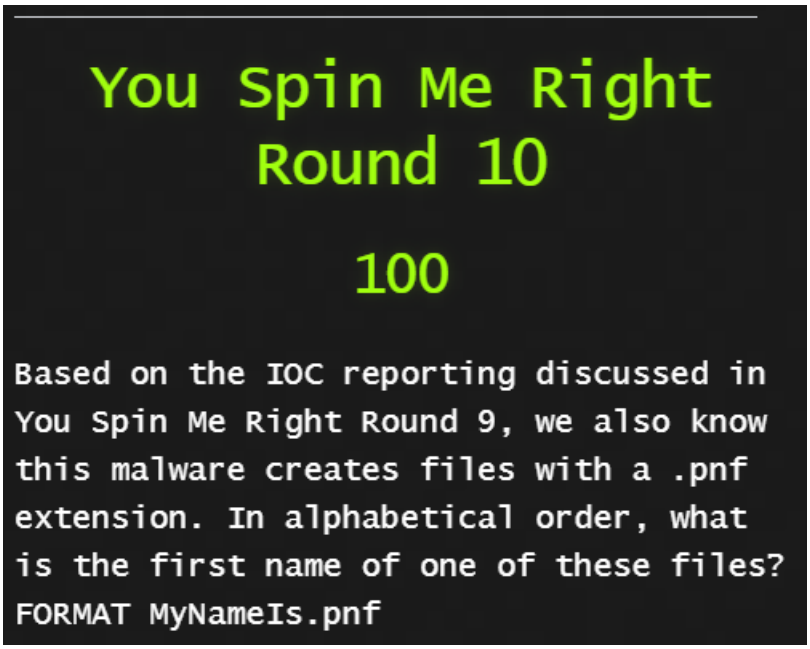
```
vol.py -f spinme.img --profile=WinXPSP3x86 handles | grep -A 10 -B 10 -i e4136
forensicator@f3273919d790:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 handles | grep -A 10 -B 10 -i e4136
Volatility Foundation Volatility Framework 2.6.1
0x81e6c220    668      0x5bc   0x1f0003 Event
0x820853a8    668      0x5c0   0x100001 File            \Device\HarddiskVolume1\WINDOWS\system32\config\systemprofile\Application Data\Microsoft\SystemCertificates\My
0x81c6d180    668      0x5c4   0x1f03ff Thread          TID 476 PID 668
0x81c3ef10    668      0x5c8   0x1f0001 Mutant
0x81bf95a0    668      0x5cc   0x1f0001 Mutant
0x81fc3cd8    668      0x5d0   0x1f0001 Mutant          {5EC171BB-F130-4a19-B782-B6E655E091B2}
0x81c94020    668      0x5d4   0x1f03ff Thread          TID 400 PID 668
0x81e89390    668      0x5d8   0x1f0003 Semaphore       shell.{210A4BA0-3AEA-1069-A2D9-08002B30309D}
0x81f12f60    668      0x5dc   0x1f0003 Event
0x81f402b8    668      0x5e0   0x1f0003 Event           {CAA6BD26-6C7B-4af0-95E2-53DE46FDDF26}
0x821b75f8    668      0x5e4   0x1f0001 Mutant          {E41362C3-F75C-4ec2-AF49-3CB6BCA591CA}
0xe29fa850    668      0x5e8    0xf0007 Section         {4A9A9FA4-5292-4607-B3CB-EE6A87A008A3}
0x81f703a8    668      0x5ec   0x1f0001 Mutant
0x81be6390    668      0x5f0   0x1f0003 Event           WkssvcShutdownEvent2
0x81be6390    668      0x5f4   0x1f0003 Event           WkssvcShutdownEvent2
0x81f492b0    668      0x5f8   0x1f03ff Thread          TID 1552 PID 668
0x82279d38    668      0x5fc   0x1f0001 Mutant
0x82109ee8    668      0x600   0x1f0001 Mutant
0x81e09c10    668      0x604   0x1f0001 Mutant
0x81f34650    668      0x608   0x1f0003 Event
0x81eae8d8    668      0x60c   0x1f0001 Mutant
```

Right after the mutant we can see that there is TID 400 PID 668 or thread 400. We will use threads on pid 668 and we can follow this to thread id 400 to find the hooked ssdt and the created date.



```
------
ETHREAD: 0x81c94020 Pid: 668 Tid: 400
Tags: HookedSSDT
Created: 2011-06-03 04:26:55 UTC+0000
Exited: 1970-01-01 00:00:00 UTC+0000
Owning Process: services.exe
Attached Process: services.exe
State: Waiting:WrLpcReceive
BasePriority: 0x9
Priority: 0x9
TEB: 0x7ffaf000
StartAddress: 0x7c8106e9 kernel32.dll
ServiceTable: 0x80552fa0
  [0] 0x80501b8c
      [0x19] NtClose 0xb240f80e PROCMON20.SYS
      [0x29] NtCreateKey 0xb240f604 PROCMON20.SYS
      [0x3f] NtDeleteKey 0xb240f4ac PROCMON20.SYS
      [0x41] NtDeleteValueKey 0xb240f4f2 PROCMON20.SYS
      [0x47] NtEnumerateKey 0xb240f3f2 PROCMON20.SYS
      [0x49] NtEnumerateValueKey 0xb240f34e PROCMON20.SYS
      [0x4f] NtFlushKey 0xb240f446 PROCMON20.SYS
      [0x62] NtLoadKey 0xb240f972 PROCMON20.SYS
      [0x77] NtOpenKey 0xb240f7d0 PROCMON20.SYS
      [0xa0] NtQueryKey 0xb240f03e PROCMON20.SYS
      [0xb1] NtQueryValueKey 0xb240f166 PROCMON20.SYS
      [0xf7] NtSetValueKey 0xb240f28a PROCMON20.SYS
      [0x107] NtUnloadKey 0xb240fac2 PROCMON20.SYS
  [1] 0x00000000
  [2] 0x00000000
  [3] 0x00000000
Win32Thread: 0x00000000
CrossThreadFlags:
Eip: 0x7c90e4f4
  eax=0x77e76c7d ebx=0x00000000 ecx=0x0166fe38 edx=0x00000000 esi=0x001348a8 edi=0x0013494c
  eip=0x7c90e4f4 esp=0x00c8fe18 ebp=0x00c8ff80 err=0x00000000
  cs=0x1b ss=0x23 ds=0x23 es=0x23 gs=0x00 fs=0x3b efl=0x00000246
  dr0=0x00000000 dr1=0x00000000 dr2=0x00000000 dr3=0x00000000 dr6=0x00000000 dr7=0x00000000
0x7c8106e9 33ed            XOR EBP, EBP
0x7c8106eb 53              PUSH EBX
0x7c8106ec 50              PUSH EAX
0x7c8106ed 6a00            PUSH 0x0
0x7c8106ef e9e8afffff      JMP 0x7c80b6dc
0x7c8106f4 90              NOP
0x7c8106f5 33ed            XOR EBP, EBP
0x7c8106f7 50              PUSH EAX
0x7c8106f8 6a00            PUSH 0x0
0x7c8106fa e945690000      JMP 0x7c817044
0x7c8106ff 90              NOP
0x7c810700 8b              DB 0x8b
```

Flag: 2011-06-03 04:26:55



We need to look at the mftparse plugin and if we follow the timeline to question 9, then we are looking for a file that was created around the 2011-06-03 04:24 timeframe.

First, we run the plugin and save the results to a file to make it less time consuming to search the output.

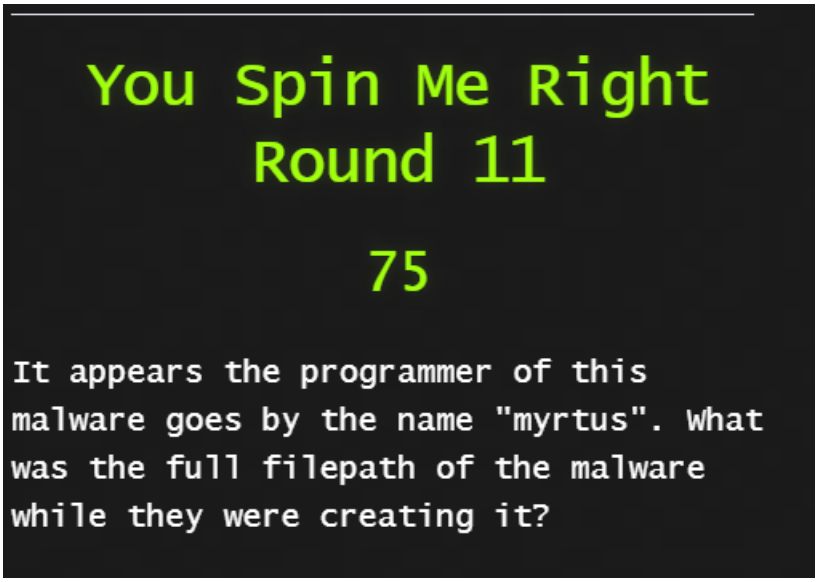
vol.py -f spinme.img --profile=WinXPSP3x86 mftparser > /tmp/mft

next we run this command on that output file: cat /tmp/mft | grep -i pnf | grep 2011-06-03

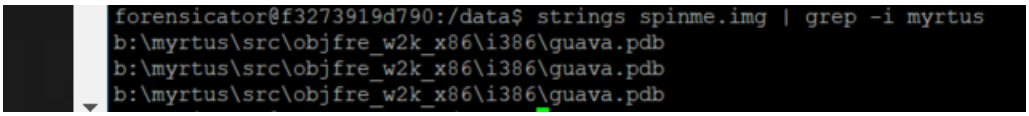
we are selecting only pnf files on the date of 2011-06-03 and we get 4 results back with our flag.

```
forensicator@56397305cc98:/data$ cat /tmp/mft | grep -i pnf | grep 2011-06-03
2011-06-03 04:26:47 UTC+0000 2011-06-03 04:26:47 UTC+0000   2011-06-03 04:26:47 UTC+0000   2011-06-03 04:26:47 UTC+0000   WINDOWS\inf\mdmcpq3.PNF
2011-06-03 04:26:55 UTC+0000 2011-06-03 04:26:55 UTC+0000   2011-06-03 04:26:55 UTC+0000   2011-06-03 04:26:55 UTC+0000   WINDOWS\inf\oem6C.PNF
2011-06-03 04:26:47 UTC+0000 2011-06-03 04:26:47 UTC+0000   2011-06-03 04:26:47 UTC+0000   2011-06-03 04:26:47 UTC+0000   WINDOWS\inf\oem7A.PNF
2011-06-03 04:26:47 UTC+0000 2011-06-03 04:26:47 UTC+0000   2011-06-03 04:26:47 UTC+0000   2011-06-03 04:26:47 UTC+0000   WINDOWS\inf\mdmeric3.PNF
```

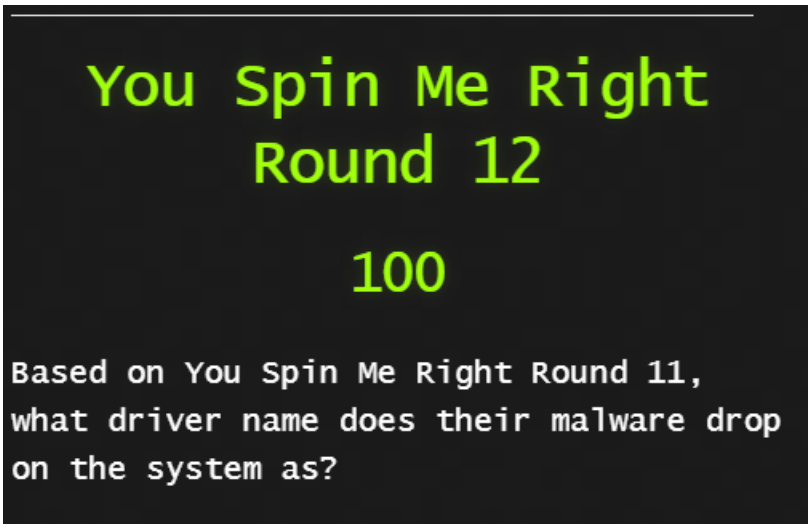
Flag: mdmcpq3.PNF



For this one we can grep the strings of the file for myrtus and should have no issues finding the file:



Flag: b:\myrtus\src\objfre_w2k_x86\i386\guava.pdb



For this we can jump in to modules and callbacks to see what information looks interesting

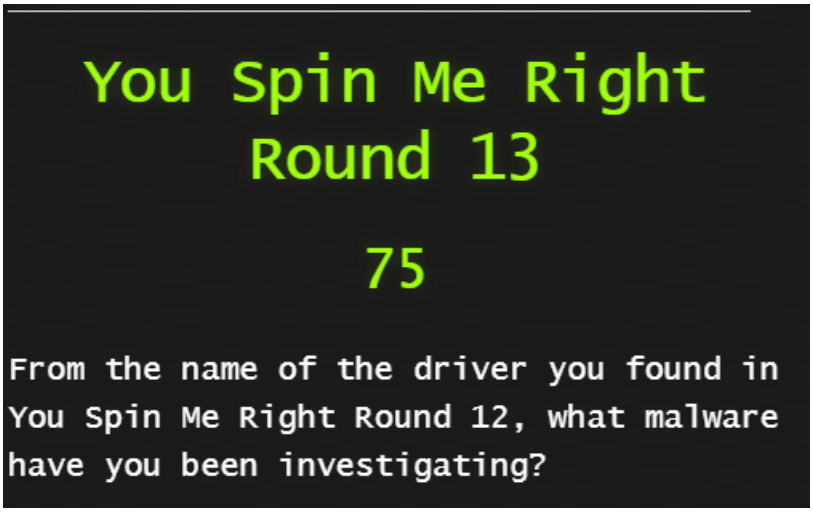
For Kernel callbacks, or a mechanism that provides a general way for drivers to request and provide notification when certain conditions are satisfied. We see a handful of drivers that have no details including some of the PROCMON20.SYS that I saw when looking at number 11. From here I also see a couple that start with mrx and what to look into those because there are 2 different drivers there.

```
IoRegisterShutdownNotification      0xf8bb05be Fs_Rec.SYS          \FileSystem\Fs_Rec
IoRegisterShutdownNotification      0xf853c2be ftdisk.sys          \Driver\Ftdisk
IoRegisterShutdownNotification      0x805cdef4 ntoskrnl.exe        \FileSystem\RAW
IoRegisterShutdownNotification      0xf83d98f1 Mup.sys             \FileSystem\Mup
IoRegisterShutdownNotification      0x805f5d66 ntoskrnl.exe        \Driver\WMIxWDM
IoRegisterFsRegistrationChange      0xf84be876 sr.sys              -
GenericKernelCallback               0xf87ad194 vmci.sys            -
IoRegisterFsRegistrationChange      0xb21d89ec mrxnet.sys          -
GenericKernelCallback               0xb240ce4c PROCMON20.SYS       -
GenericKernelCallback               0x805f81a6 ntoskrnl.exe        -
GenericKernelCallback               0xb240cc9a PROCMON20.SYS       -
GenericKernelCallback               0xf895ad06 mrxcls.sys          -
GenericKernelCallback               0xb240cb94 PROCMON20.SYS       -
GenericKernelCallback               0xb240cb94 PROCMON20.SYS       -
IoRegisterFsRegistrationChange      0xf84d54b8 fltMgr.sys          -
PsSetLoadImageNotifyRoutine         0xb240ce4c PROCMON20.SYS       -
PsSetLoadImageNotifyRoutine         0x805f81a6 ntoskrnl.exe        -
PsSetLoadImageNotifyRoutine         0xf895ad06 mrxcls.sys          -
PsSetCreateThreadNotifyRoutine      0xb240cc9a PROCMON20.SYS       -
PsSetCreateProcessNotifyRoutine     0xf87ad194 vmci.sys            -
PsSetCreateProcessNotifyRoutine     0xb240cb94 PROCMON20.SYS       -
KeBugCheckCallbackListHead          0xf83e65ef NDIS.sys            Ndis miniport
```

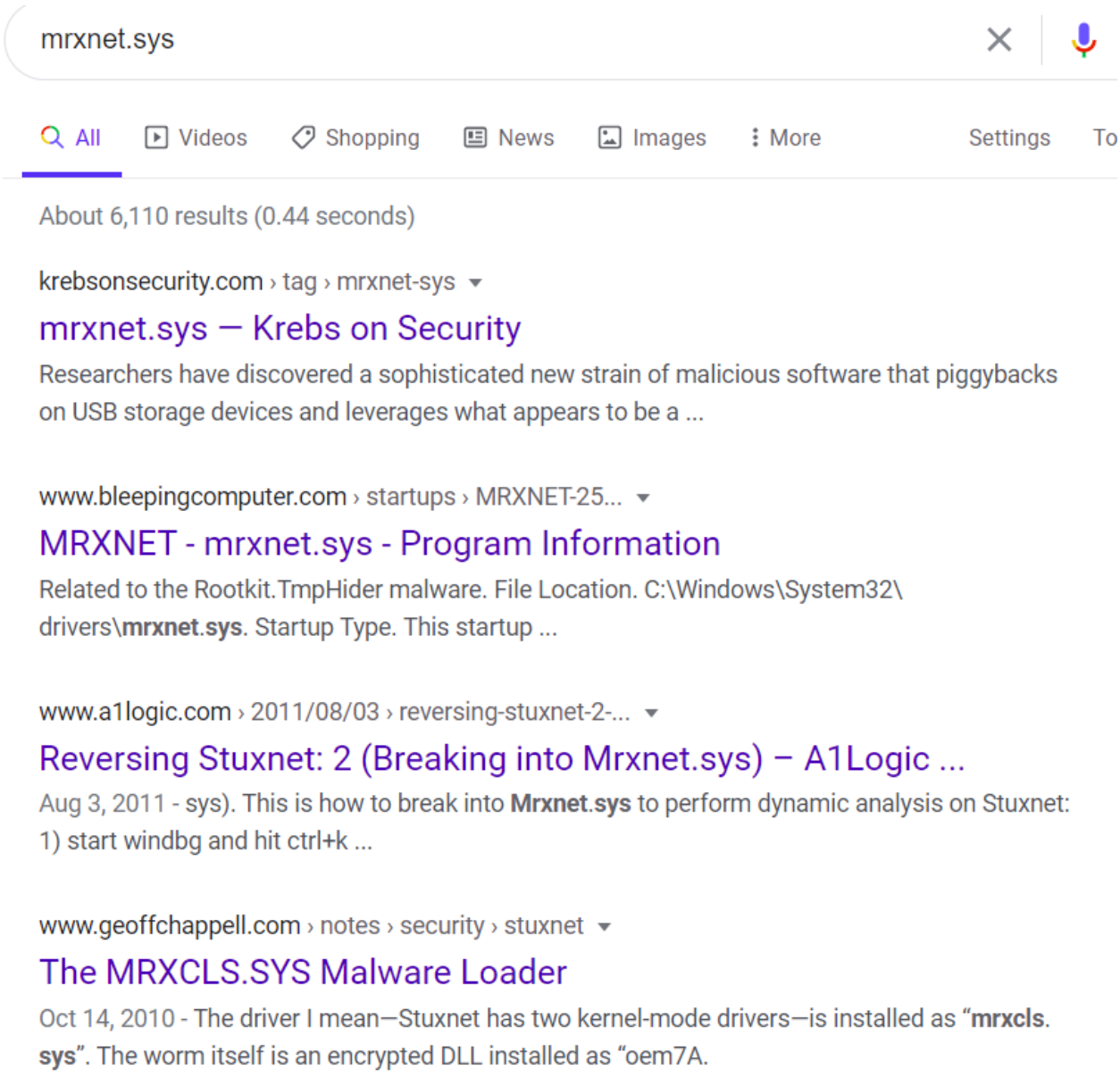
vol.py -f spinme.img --profile=WinXPSP3x86 modules | grep mrx

```
forensicator@a96f7664d3f8:/data$ vol.py -f spinme.img --profile=WinXPSP3x86 modules | grep mrx
Volatility Foundation Volatility Framework 2.6.1
0x820a31a0 mrxsmb.sys          0xb2c49000    0x70000 \SystemRoot\system32\DRIVERS\mrxsmb.sys
0x82306ad0 mrxdav.sys          0xb285e000    0x2d000 \SystemRoot\system32\DRIVERS\mrxdav.sys
0x81f8cb60 mrxcls.sys          0xf895a000     0x5000 \??\C:\WINDOWS\system32\Drivers\mrxcls.sys
0x81c2a530 mrxnet.sys          0xb21d8000     0x3000 \??\C:\WINDOWS\system32\Drivers\mrxnet.sys
```

Flag: mrxnet.sys



Quick Google search on this returns the following.



Flag: Stuxnet