Malware Memory Forensics

Monnappa

www.SecurityXploded.com

Disclaimer

The Content, Demonstration, Source Code and Programs presented here is "AS IS" without any warranty or conditions of any kind. Also the views/ideas/knowledge expressed here are solely of the trainer’s only and nothing to do with the company or the organization in which the trainer is currently working.

However in no circumstances neither the Trainer nor SecurityXploded is responsible for any damage or loss caused due to use or misuse of the information presented here.

Acknowledgement

- Special thanks to **Null** community for their extended support and co-operation.
- Special thanks to **ThoughtWorks** for the beautiful venue.
- Thanks to all the trainers who have devoted their precious time and countless hours to make it happen.

Advanced Malware Analysis Training

This presentation is part of our **Advanced Malware Analysis** Training program. Currently it is delivered only during our local meets for FREE of cost.

For complete details of this course, visit our Security Training page.

Who am I

Monnappa (m0nna)

- Member of SecurityXploded
- Info Security Investigator @ Cisco
- Reverse Engineering, Malware Analysis, Memory Forensics
- GREM, CEH
- Email: monnappa22@gmail.com
- Twitter: @monnappa22
- LinkedIn: http://www.linkedin.com/pub/monnappa-ka-grem-ceh/42/45a/1b8

Contents

- Why Memory Forensics?
- Steps in Memory Forensics
- Volatility Quick Overview
- Volatility help and plugins
- Demo 1
- Demo 2

Why Memory Forensics?

- Finding and extracting forensic artefacts
- Helps in malware analysis
- Determining process, network, registry activities
- Reconstructing original state of the system
- Assists with unpacking, rootkit detection and reverse engineering

Steps in Memory Forensics

- **Memory acquisition - Dumping the memory of a target machine**
  - **tools: Win32dd/Win64dd, Memoryze, DumpIt, FastDump**
  - **In Virtual machine: Suspend the VM and use .vmem file**
- **Memory analysis - Analyzing the memory dump for forensic artefacts**
  - **tools: Volatility, Memoryze**

Volatility Quick Overview

- Advanced memory Forensics Framework written in python
- Installation details:

  - http://code.google.com/p/volatility/wiki/FullInstallation

- Use -h or --help option to get list of command-line switches

  - example: python vol.py –h

- Use -f <filename> and --profile to indicate the memory dump you are analyzing

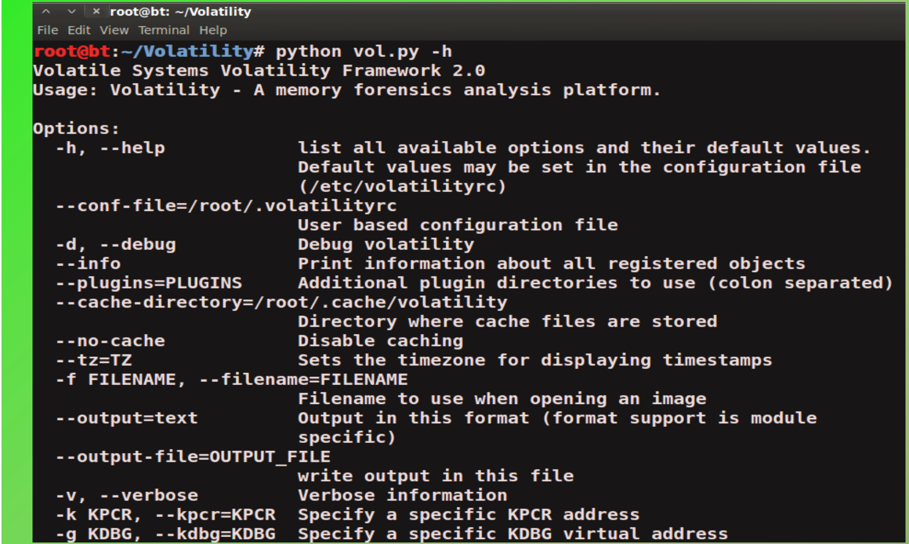
  example: python vol.py -f mem.dmp --profile=WinXPSP3x86

- To know the --profile info use below command:

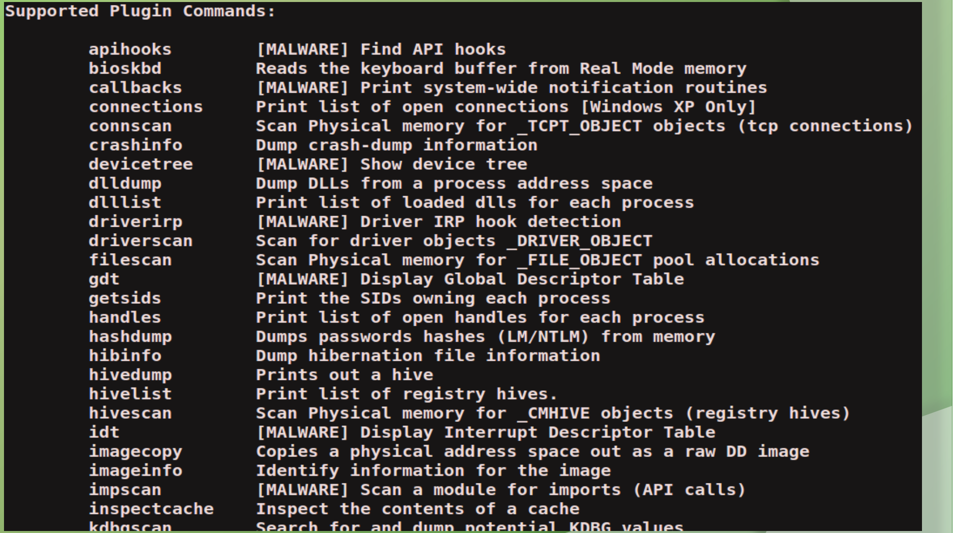
  example: python vol.py -f mem.dmp imageinfo

Volatility help and plugins

-h or –help option displays help and available plug-in commands in volatility.



```
root@bt:~/Volatility# python vol.py -h
Volatile Systems Volatility Framework 2.0
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=/root/.volatilityrc
                        User based configuration file
  -d, --debug           Debug volatility
  --info                Print information about all registered objects
  --plugins=PLUGINS     Additional plugin directories to use (colon separated)
  --cache-directory=/root/.cache/volatility
                        Directory where cache files are stored
  --no-cache            Disable caching
  --tz=TZ               Sets the timezone for displaying timestamps
  -f FILENAME, --filename=FILENAME
                        Filename to use when opening an image
  --output=text         Output in this format (format support is module
                        specific)
  --output-file=OUTPUT_FILE
                        write output in this file
  -v, --verbose         Verbose information
  -k KPCR, --kpcr=KPCR  Specify a specific KPCR address
  -g KDBG, --kdbg=KDBG  Specify a specific KDBG virtual address
```

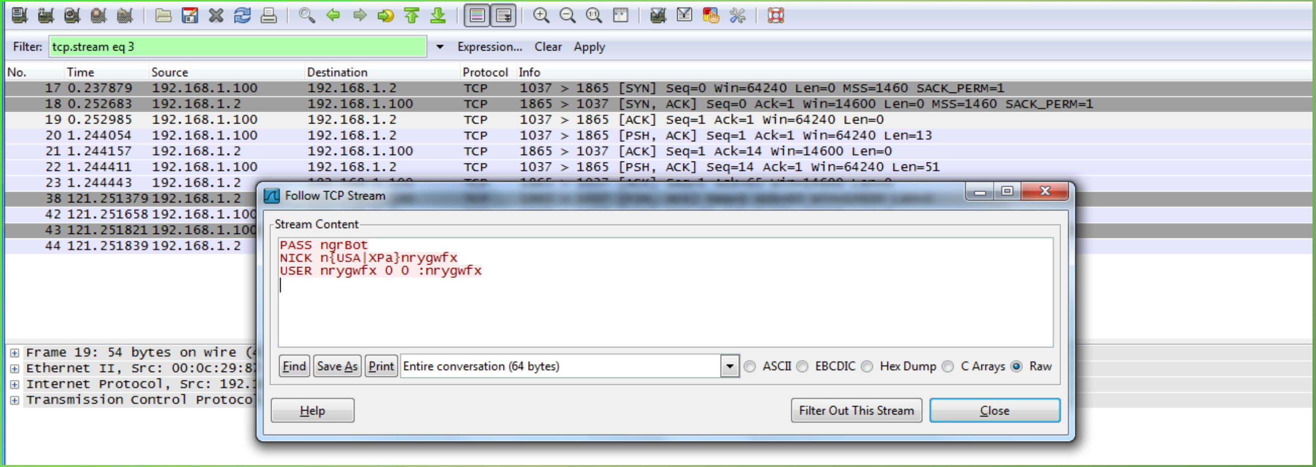


```
Supported Plugin Commands:

        apihooks        [MALWARE] Find API hooks
        bioskbd         Reads the keyboard buffer from Real Mode memory
        callbacks       [MALWARE] Print system-wide notification routines
        connections     Print list of open connections [Windows XP Only]
        connscan        Scan Physical memory for _TCPT_OBJECT objects (tcp connections)
        crashinfo       Dump crash-dump information
        devicetree      [MALWARE] Show device tree
        dlldump         Dump DLLs from a process address space
        dlllist         Print list of loaded dlls for each process
        driverirp       [MALWARE] Driver IRP hook detection
        driverscan      Scan for driver objects _DRIVER_OBJECT
        filescan        Scan Physical memory for _FILE_OBJECT pool allocations
        gdt             [MALWARE] Display Global Descriptor Table
        getsids         Print the SIDs owning each process
        handles         Print list of open handles for each process
        hashdump        Dumps passwords hashes (LM/NTLM) from memory
        hibinfo         Dump hibernation file information
        hivedump        Prints out a hive
        hivelist        Print list of registry hives.
        hivescan        Scan Physical memory for _CMHIVE objects (registry hives)
        idt             [MALWARE] Display Interrupt Descriptor Table
        imagecopy       Copies a physical address space out as a raw DD image
        imageinfo       Identify information for the image
        impscan         [MALWARE] Scan a module for imports (API calls)
        inspectcache    Inspect the contents of a cache
        kdbgscan        Search for and dump potential KDBG values
```

DEMO 1

Demo-Scenario 1

Your security device alerts on a malicious irc connection to ip address 192.168.1.2 on port 1865 from a source ip 192.168.1.100 (shown below). you are asked to investigate and perform memory forensics on the machine 192.168.1.100



Filter: tcp.stream eq 3

```
No.  Time        Source          Destination     Protocol Info
  17 0.237879    192.168.1.100   192.168.1.2     TCP      1037 > 1865 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
  18 0.252683    192.168.1.2     192.168.1.100   TCP      1865 > 1037 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SACK_PERM=1
  19 0.252985    192.168.1.100   192.168.1.2     TCP      1037 > 1865 [ACK] Seq=1 Ack=1 Win=64240 Len=0
  20 1.244054    192.168.1.100   192.168.1.2     TCP      1037 > 1865 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=13
  21 1.244157    192.168.1.2     192.168.1.100   TCP      1865 > 1037 [ACK] Seq=1 Ack=14 Win=14600 Len=0
  22 1.244411    192.168.1.100   192.168.1.2     TCP      1037 > 1865 [PSH, ACK] Seq=14 Ack=1 Win=64240 Len=51
  23 1.244443    192.168.1.2     192.168.1.100   TCP      1865 > 1037 [ACK] Seq=1 Ack=65 Win=14600 Len=0
  38 121.251379  192.168.1.2
  42 121.251658  192.168.1.100
  43 121.251821  192.168.1.100
  44 121.251839  192.168.1.2

⊞ Frame 19: 54 bytes on wire (
⊞ Ethernet II, Src: 00:0c:29:8
⊞ Internet Protocol, Src: 192.1
⊞ Transmission Control Protoco
```

Follow TCP Stream

Stream Content

```
PASS ngrBot
NICK n{USA|XPa}nrygwfx
USER nrygwfx 0 0 :nrygwfx
```

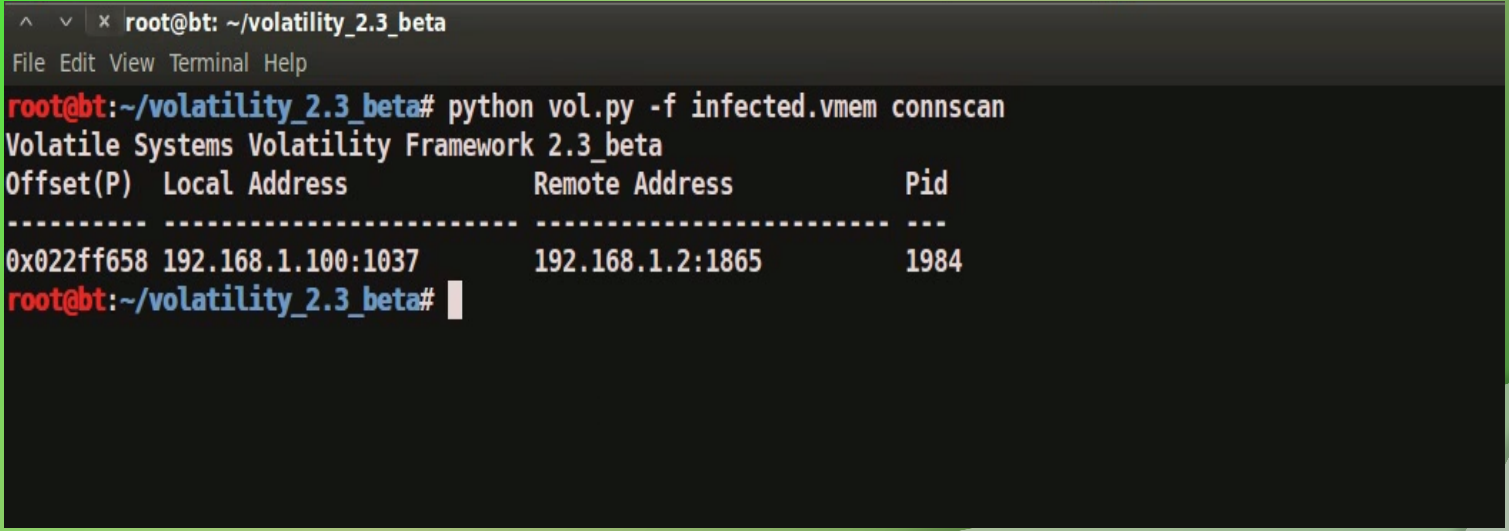
Entire conversation (64 bytes) ○ ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays ● Raw

- To start with, acquire the memory image "infected.dmp" from 192.168.1.100, using memory acquisition tools (like Dumpit or win32dd)

- Analyze the memory dump "infected.dmp"

Step 1 – Start With what you know

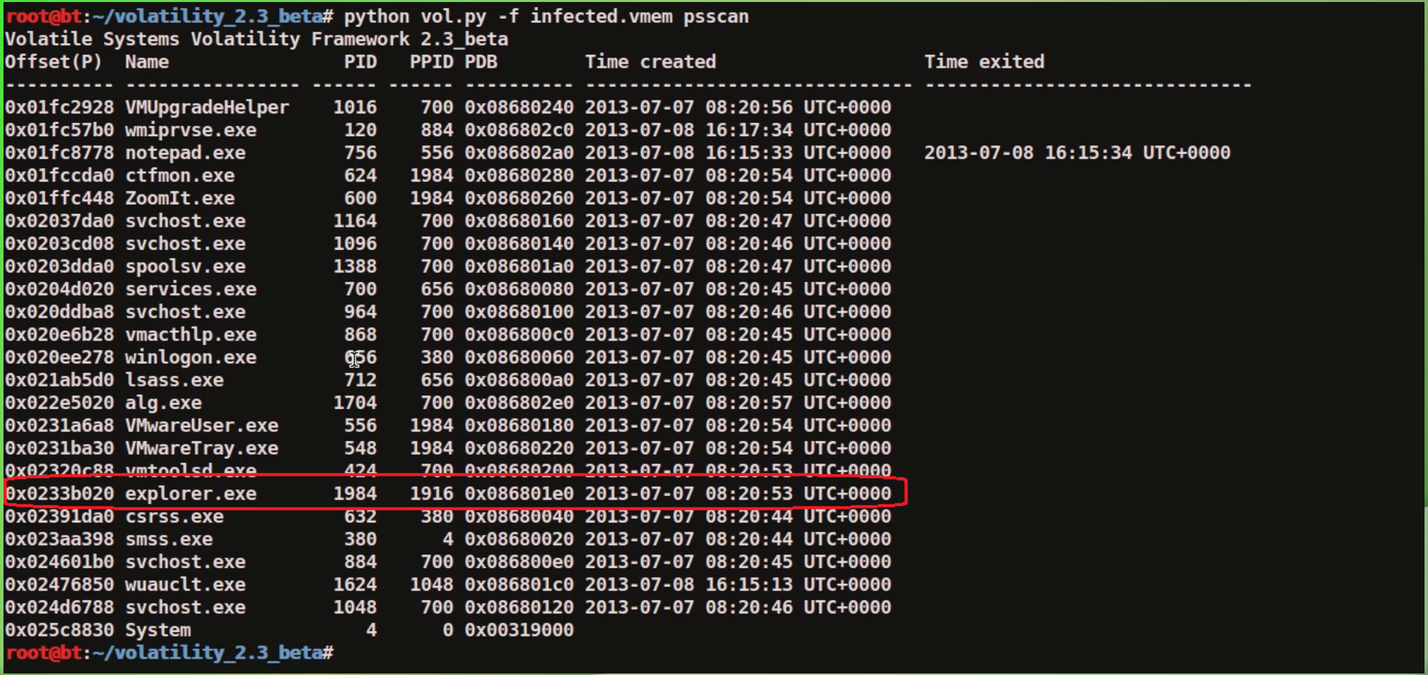
Volatility’s connscan module shows connection to the malicious ip on port 1865 by pid 1984



```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem connscan
Volatile Systems Volatility Framework 2.3_beta
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------- ------------------------- ---
0x022ff658 192.168.1.100:1037        192.168.1.2:1865          1984
root@bt:~/volatility_2.3_beta#
```

Step 2 – Who is Pid 1984?

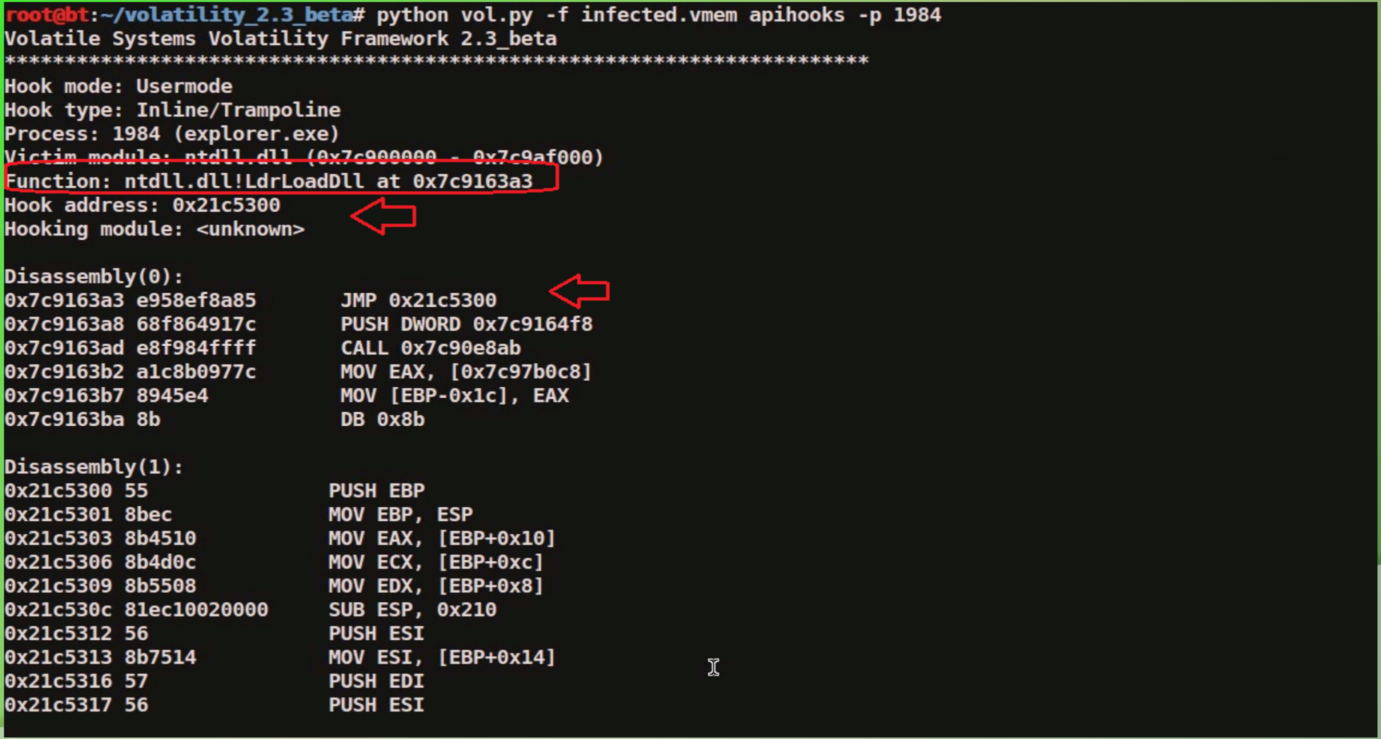
“psscan” shows pid 1984 belongs to explorer.exe



```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem psscan
Volatile Systems Volatility Framework 2.3_beta
Offset(P)  Name                PID   PPID PDB        Time created                   Time exited
---------- ---------------- ------ ------ ---------- ------------------------------ ------------------------------
0x01fc2928 VMUpgradeHelper    1016    700 0x08680240 2013-07-07 08:20:56 UTC+0000
0x01fc57b0 wmiprvse.exe        120    884 0x086802c0 2013-07-08 16:17:34 UTC+0000
0x01fc8778 notepad.exe         756    556 0x086802a0 2013-07-08 16:15:33 UTC+0000   2013-07-08 16:15:34 UTC+0000
0x01fccda0 ctfmon.exe          624   1984 0x08680280 2013-07-07 08:20:54 UTC+0000
0x01ffc448 ZoomIt.exe          600   1984 0x08680260 2013-07-07 08:20:54 UTC+0000
0x02037da0 svchost.exe        1164    700 0x08680160 2013-07-07 08:20:47 UTC+0000
0x0203cd08 svchost.exe        1096    700 0x08680140 2013-07-07 08:20:46 UTC+0000
0x0203dda0 spoolsv.exe        1388    700 0x086801a0 2013-07-07 08:20:47 UTC+0000
0x0204d020 services.exe        700    656 0x08680080 2013-07-07 08:20:45 UTC+0000
0x020ddba8 svchost.exe         964    700 0x08680100 2013-07-07 08:20:46 UTC+0000
0x020e6b28 vmacthlp.exe        868    700 0x086800c0 2013-07-07 08:20:45 UTC+0000
0x020ee278 winlogon.exe        656    380 0x08680060 2013-07-07 08:20:45 UTC+0000
0x021ab5d0 lsass.exe           712    656 0x086800a0 2013-07-07 08:20:45 UTC+0000
0x022e5020 alg.exe            1704    700 0x086802e0 2013-07-07 08:20:57 UTC+0000
0x0231a6a8 VMwareUser.exe      556   1984 0x08680180 2013-07-07 08:20:54 UTC+0000
0x0231ba30 VMwareTray.exe      548   1984 0x08680220 2013-07-07 08:20:54 UTC+0000
0x02320c88 vmtoolsd.exe        424    700 0x08680200 2013-07-07 08:20:53 UTC+0000
0x0233b020 explorer.exe       1984   1916 0x086801e0 2013-07-07 08:20:53 UTC+0000
0x02391da0 csrss.exe           632    380 0x08680040 2013-07-07 08:20:44 UTC+0000
0x023aa398 smss.exe            380      4 0x08680020 2013-07-07 08:20:44 UTC+0000
0x024601b0 svchost.exe         884    700 0x086800e0 2013-07-07 08:20:45 UTC+0000
0x02476850 wuauclt.exe        1624   1048 0x086801c0 2013-07-08 16:15:13 UTC+0000
0x024d6788 svchost.exe        1048    700 0x08680120 2013-07-07 08:20:46 UTC+0000
0x025c8830 System                4      0 0x00319000
root@bt:~/volatility_2.3_beta#
```

Step 3 – apihooks in explorer.exe

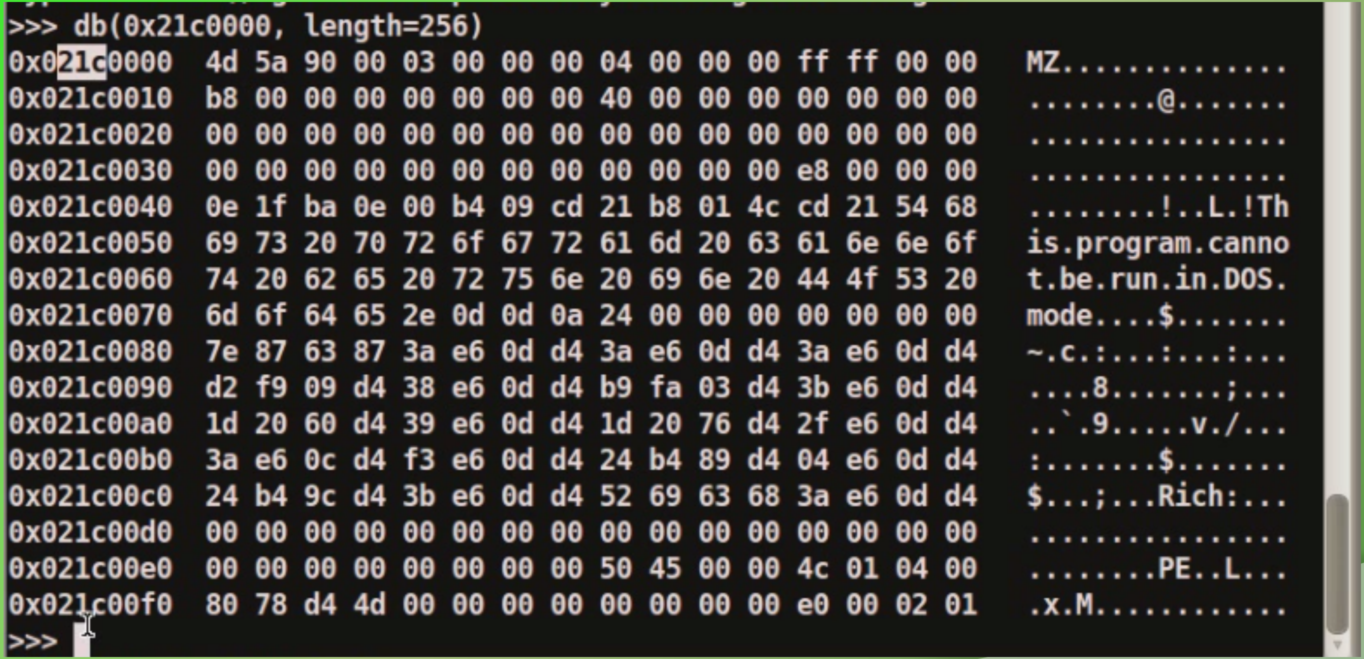
apihooks module show, inline api hooks in explorer.exe (pid 1984) and jump to an unknown location



```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem apihooks -p 1984
Volatile Systems Volatility Framework 2.3_beta
************************************************************************
Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 1984 (explorer.exe)
Victim module: ntdll.dll (0x7c900000 - 0x7c9af000)
Function: ntdll.dll!LdrLoadDll at 0x7c9163a3
Hook address: 0x21c5300
Hooking module: <unknown>

Disassembly(0):
0x7c9163a3 e958ef8a85       JMP 0x21c5300
0x7c9163a8 68f864917c       PUSH DWORD 0x7c9164f8
0x7c9163ad e8f984ffff       CALL 0x7c90e8ab
0x7c9163b2 a1c8b0977c       MOV EAX, [0x7c97b0c8]
0x7c9163b7 8945e4           MOV [EBP-0x1c], EAX
0x7c9163ba 8b               DB 0x8b

Disassembly(1):
0x21c5300 55               PUSH EBP
0x21c5301 8bec             MOV EBP, ESP
0x21c5303 8b4510           MOV EAX, [EBP+0x10]
0x21c5306 8b4d0c           MOV ECX, [EBP+0xc]
0x21c5309 8b5508           MOV EDX, [EBP+0x8]
0x21c530c 81ec10020000     SUB ESP, 0x210
0x21c5312 56               PUSH ESI
0x21c5313 8b7514           MOV ESI, [EBP+0x14]
0x21c5316 57               PUSH EDI
0x21c5317 56               PUSH ESI
```

Step 4 – Embedded exe in explorer.exe

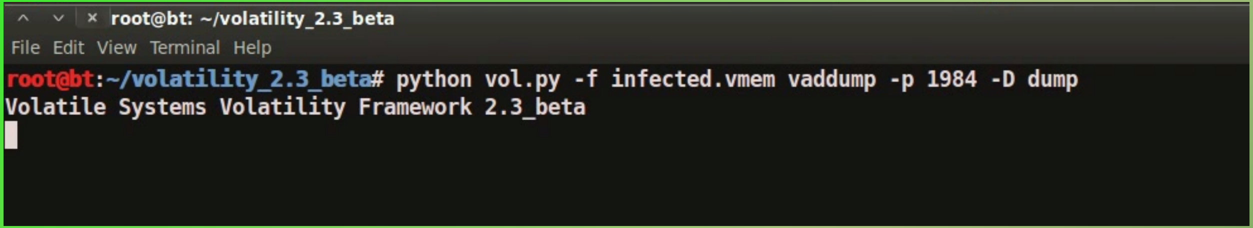
Printing the bytes show the presence of embedded executable in explorer.exe



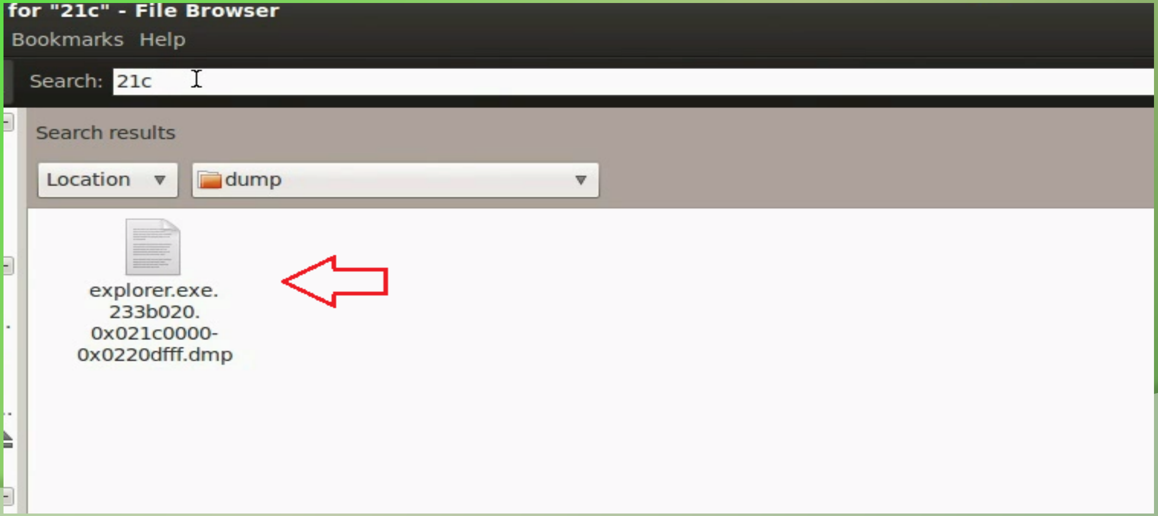
```
>>> db(0x21c0000, length=256)
0x021c0000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x021c0010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x021c0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x021c0030  00 00 00 00 00 00 00 00 00 00 00 00 e8 00 00 00   ................
0x021c0040  0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68   ........!..L.!Th
0x021c0050  69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f   is.program.canno
0x021c0060  74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20   t.be.run.in.DOS.
0x021c0070  6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00   mode....$.......
0x021c0080  7e 87 63 87 3a e6 0d d4 3a e6 0d d4 3a e6 0d d4   ~.c.:...:...:...
0x021c0090  d2 f9 09 d4 38 e6 0d d4 b9 fa 03 d4 3b e6 0d d4   ....8.......;...
0x021c00a0  1d 20 60 d4 39 e6 0d d4 1d 20 76 d4 2f e6 0d d4   ..`.9.....v./...
0x021c00b0  3a e6 0c d4 f3 e6 0d d4 24 b4 89 d4 04 e6 0d d4   :.......$.......
0x021c00c0  24 b4 9c d4 3b e6 0d d4 52 69 63 68 3a e6 0d d4   $...;...Rich:...
0x021c00d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x021c00e0  00 00 00 00 00 00 00 00 50 45 00 00 4c 01 04 00   ........PE..L...
0x021c00f0  80 78 d4 4d 00 00 00 00 00 00 00 00 e0 00 02 01   .x.M............
>>>
```

Step 5 – dumping the embedded exe

vaddump dumps the embedded exe from explorer.exe



```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem vaddump -p 1984 -D dump
Volatile Systems Volatility Framework 2.3_beta
```



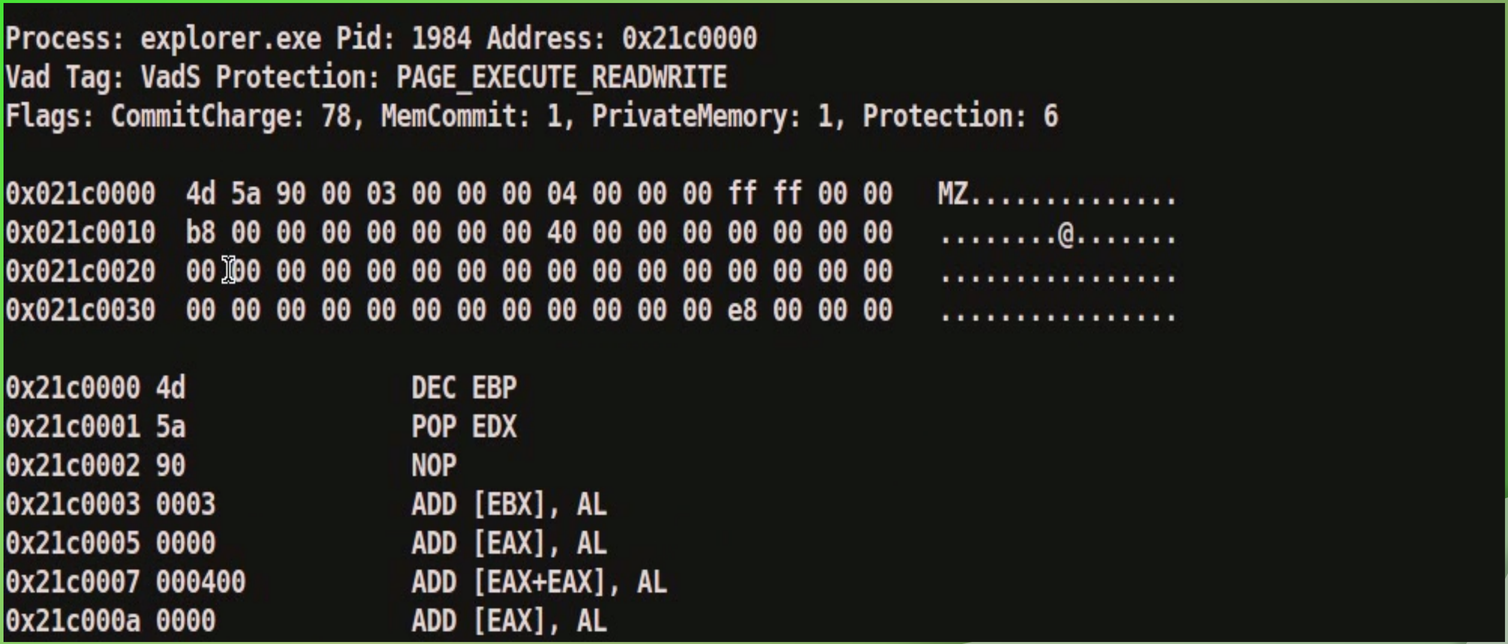
Search: 21c

dump

explorer.exe.233b020.0x021c0000-0x0220dfff.dmp

Step 6 – embedded exe by malfind plugin

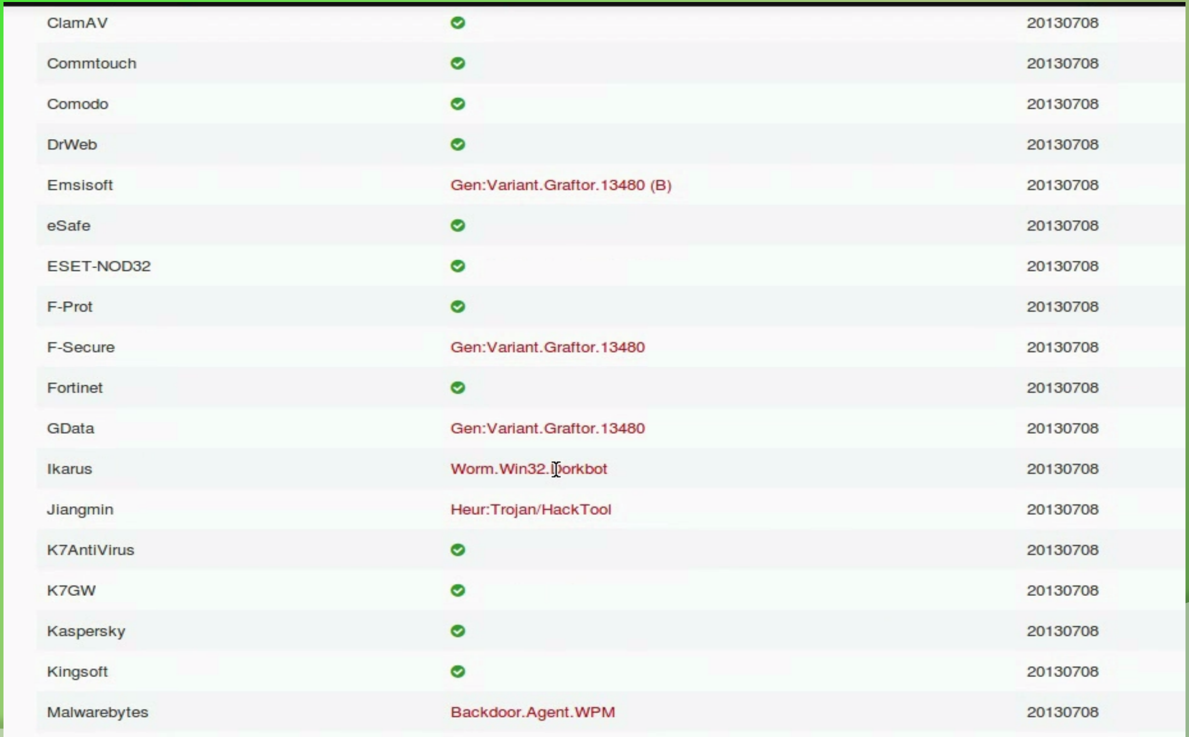
Malfind plugin can also be used to detect embedded exe and dump it as shown below



```
Process: explorer.exe Pid: 1984 Address: 0x21c0000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 78, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x021c0000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x021c0010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x021c0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x021c0030  00 00 00 00 00 00 00 00 00 00 00 00 e8 00 00 00   ................

0x21c0000 4d             DEC EBP
0x21c0001 5a             POP EDX
0x21c0002 90             NOP
0x21c0003 0003           ADD [EBX], AL
0x21c0005 0000           ADD [EAX], AL
0x21c0007 000400         ADD [EAX+EAX], AL
0x21c000a 0000           ADD [EAX], AL
```

Step 7 – VirusTotal submission

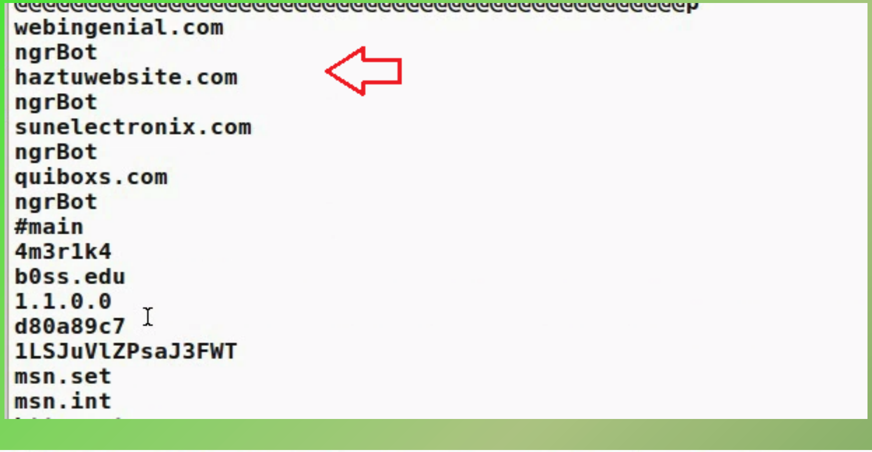
Submission to virustotal, confirms the dumped executable to be malicious



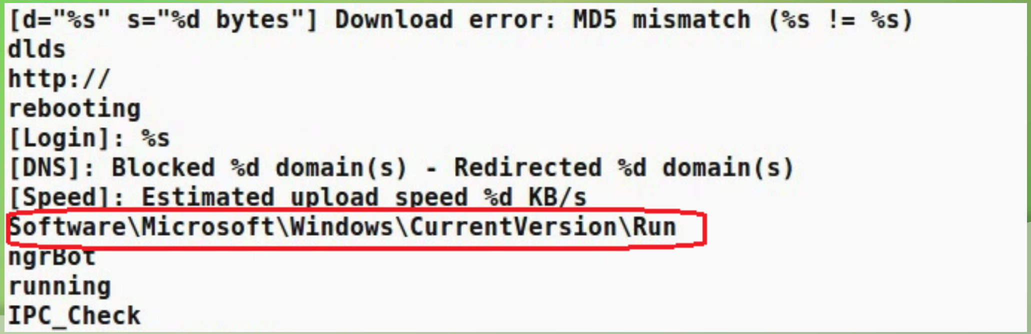
ClamAV	✓	20130708
Commtouch	✓	20130708
Comodo	✓	20130708
DrWeb	✓	20130708
Emsisoft	Gen:Variant.Graftor.13480 (B)	20130708
eSafe	✓	20130708
ESET-NOD32	✓	20130708
F-Prot	✓	20130708
F-Secure	Gen:Variant.Graftor.13480	20130708
Fortinet	✓	20130708
GData	Gen:Variant.Graftor.13480	20130708
Ikarus	Worm.Win32.Dorkbot	20130708
Jiangmin	Heur:Trojan/HackTool	20130708
K7AntiVirus	✓	20130708
K7GW	✓	20130708
Kaspersky	✓	20130708
Kingsoft	✓	20130708
Malwarebytes	Backdoor.Agent.WPM	20130708

Step 8 – getting more information

Strings extracted from the dumped executable, show reference to interesting artifacts (domains and the registry key)



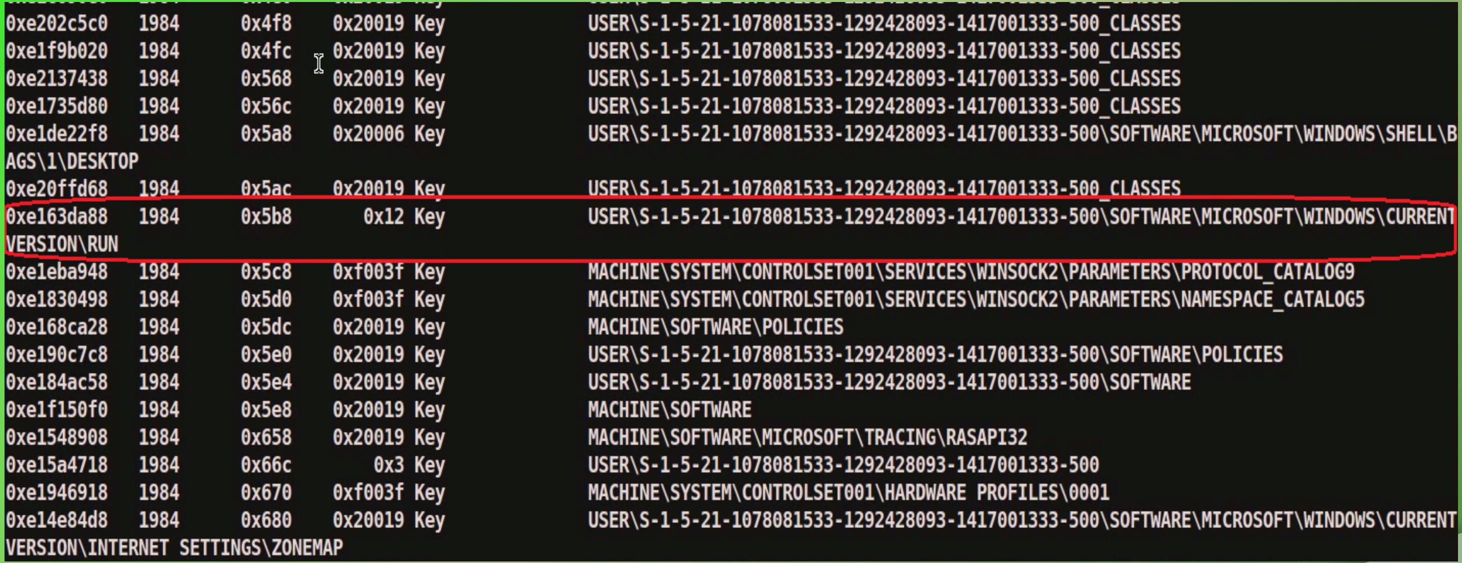
```
webingenial.com
ngrBot
haztuwebsite.com
ngrBot
sunelectronix.com
ngrBot
quiboxs.com
ngrBot
#main
4m3r1k4
b0ss.edu
1.1.0.0
d80a89c7
1LSJuVlZPsaJ3FWT
msn.set
msn.int
```



```
[d="%s" s="%d bytes"] Download error: MD5 mismatch (%s != %s)
dlds
http://
rebooting
[Login]: %s
[DNS]: Blocked %d domain(s) - Redirected %d domain(s)
[Speed]: Estimated upload speed %d KB/s
Software\Microsoft\Windows\CurrentVersion\Run
ngrBot
running
IPC_Check
```

Step 9 – explorer.exe handles

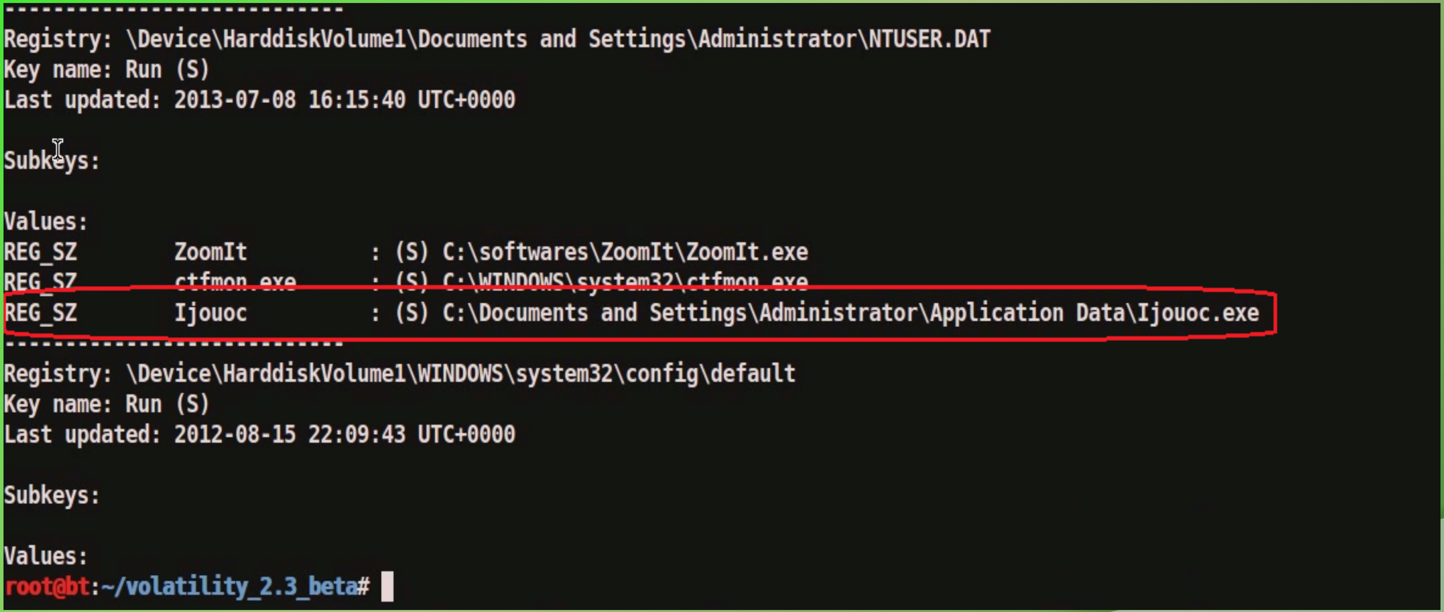
Handles in the explorer.exe (pid 1984) shows the presence of the run registry key



```
0xe202c5c0   1984      0x4f8    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500_CLASSES
0xe1f9b020   1984      0x4fc    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500_CLASSES
0xe2137438   1984      0x568    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500_CLASSES
0xe1735d80   1984      0x56c    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500_CLASSES
0xe1de22f8   1984      0x5a8    0x20006 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500\SOFTWARE\MICROSOFT\WINDOWS\SHELL\B
AGS\1\DESKTOP
0xe20ffd68   1984      0x5ac    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500_CLASSES
0xe163da88   1984      0x5b8       0x12 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500\SOFTWARE\MICROSOFT\WINDOWS\CURRENT
VERSION\RUN
0xe1eba948   1984      0x5c8    0xf003f Key            MACHINE\SYSTEM\CONTROLSET001\SERVICES\WINSOCK2\PARAMETERS\PROTOCOL_CATALOG9
0xe1830498   1984      0x5d0    0xf003f Key            MACHINE\SYSTEM\CONTROLSET001\SERVICES\WINSOCK2\PARAMETERS\NAMESPACE_CATALOG5
0xe168ca28   1984      0x5dc    0x20019 Key            MACHINE\SOFTWARE\POLICIES
0xe190c7c8   1984      0x5e0    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500\SOFTWARE\POLICIES
0xe184ac58   1984      0x5e4    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500\SOFTWARE
0xe1f150f0   1984      0x5e8    0x20019 Key            MACHINE\SOFTWARE
0xe1548908   1984      0x658    0x20019 Key            MACHINE\SOFTWARE\MICROSOFT\TRACING\RASAPI32
0xe15a4718   1984      0x66c        0x3 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500
0xe1946918   1984      0x670    0xf003f Key            MACHINE\SYSTEM\CONTROLSET001\HARDWARE PROFILES\0001
0xe14e84d8   1984      0x680    0x20019 Key            USER\S-1-5-21-1078081533-1292428093-1417001333-500\SOFTWARE\MICROSOFT\WINDOWS\CURRENT
VERSION\INTERNET SETTINGS\ZONEMAP
```

Step 10 – Printing the registry key

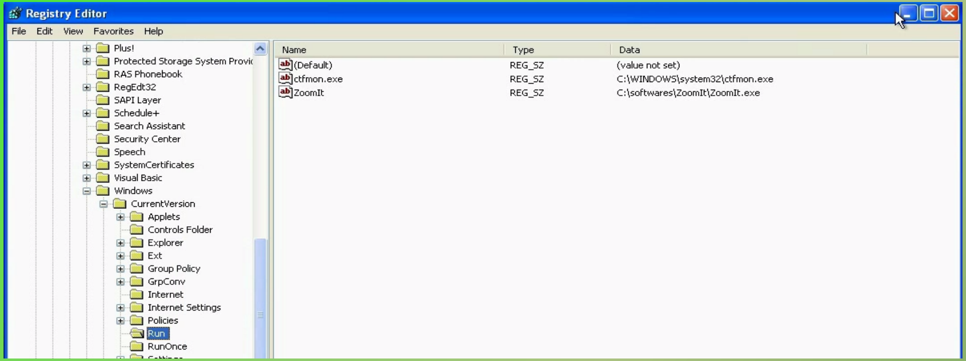
Malware adds values to registry key to survive the reboot



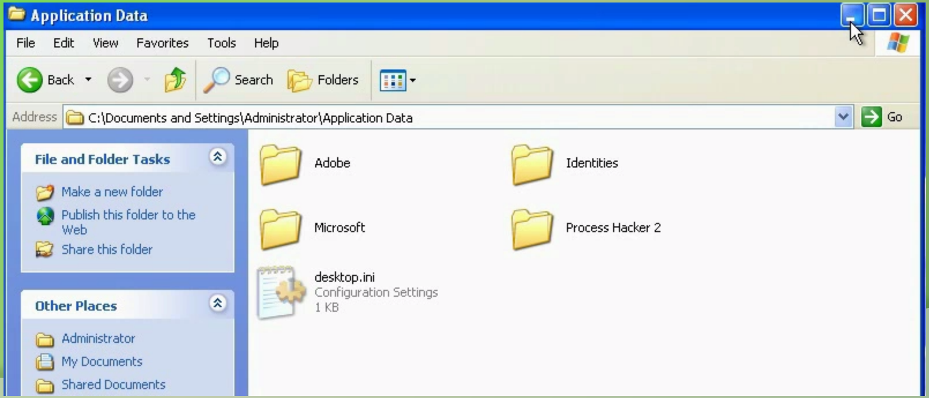
```
----------------------------
Registry: \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT
Key name: Run (S)
Last updated: 2013-07-08 16:15:40 UTC+0000

Subkeys:

Values:
REG_SZ        ZoomIt          : (S) C:\softwares\ZoomIt\ZoomIt.exe
REG_SZ        ctfmon.exe      : (S) C:\WINDOWS\system32\ctfmon.exe
REG_SZ        Ijouoc          : (S) C:\Documents and Settings\Administrator\Application Data\Ijouoc.exe
----------------------------
Registry: \Device\HarddiskVolume1\WINDOWS\system32\config\default
Key name: Run (S)
Last updated: 2012-08-15 22:09:43 UTC+0000

Subkeys:

Values:
root@bt:~/volatility_2.3_beta#
```

Step 11 – examining the infected system

Malware hides the registry value and the malicious file on the infected system



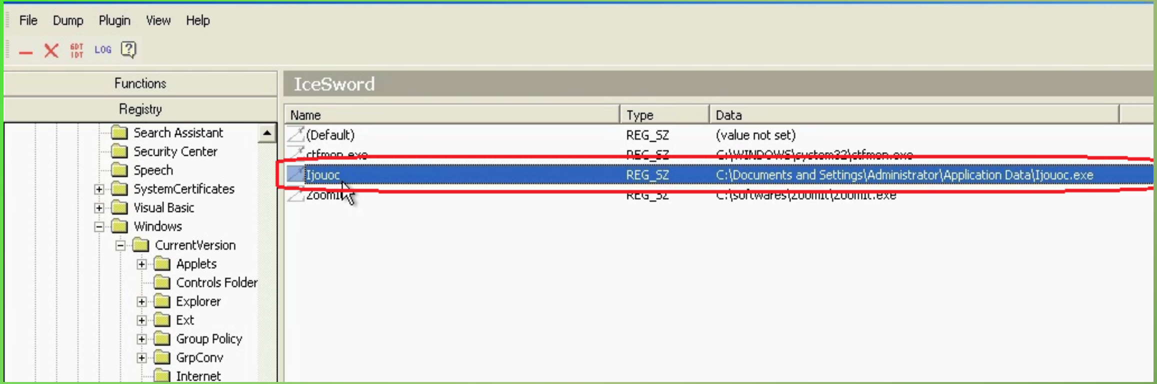
Name	Type	Data
(Default)	REG_SZ	(value not set)
ctfmon.exe	REG_SZ	C:\WINDOWS\system32\ctfmon.exe
ZoomIt	REG_SZ	C:\softwares\ZoomIt\ZoomIt.exe



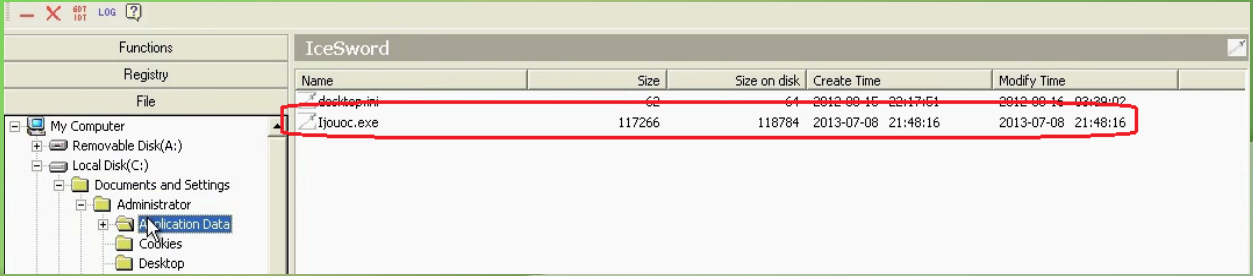
Address: C:\Documents and Settings\Administrator\Application Data

Step 12 – Finding the malware on infected system

Rootkit detection tool detects the hidden file and the registry entry



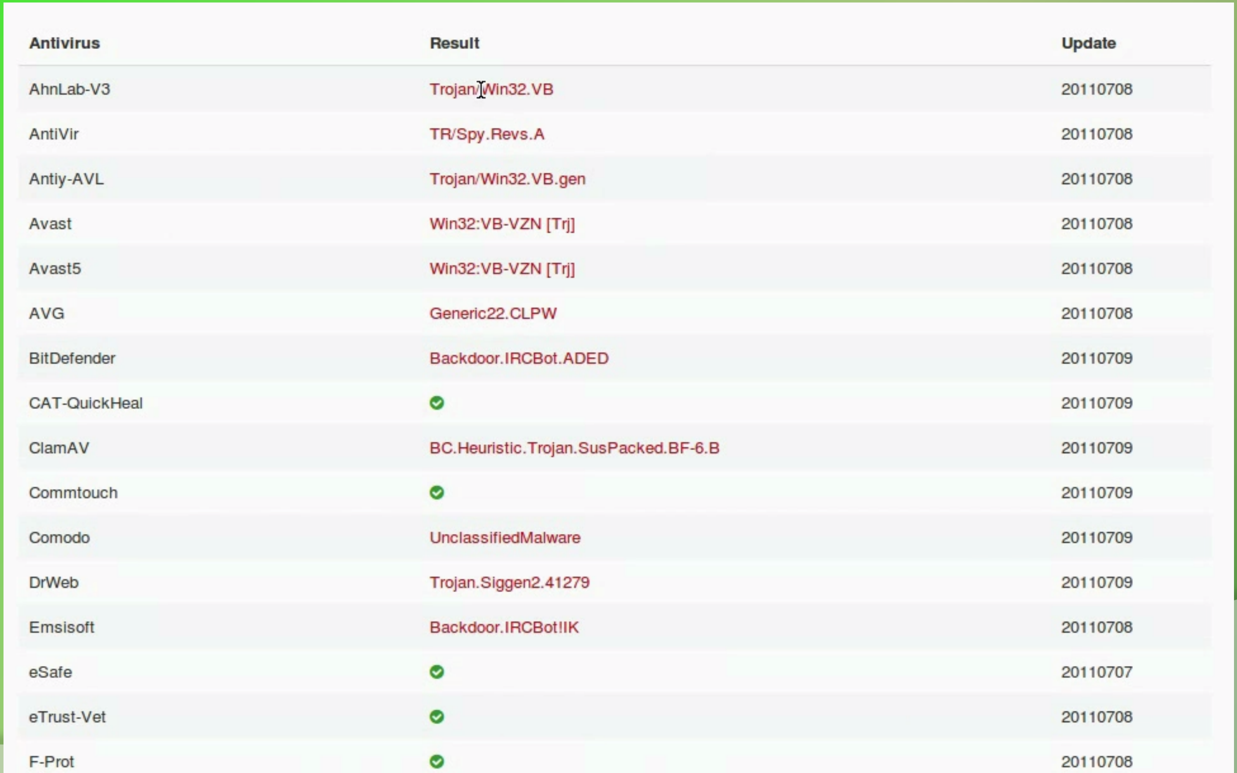
Name	Type	Data
(Default)	REG_SZ	(value not set)
ctfmon.exe	REG_SZ	C:\WINDOWS\system32\ctfmon.exe
Ijouoc	REG_SZ	C:\Documents and Settings\Administrator\Application Data\Ijouoc.exe
ZoomIt	REG_SZ	C:\softwares\zoomit\zoomit.exe



Name	Size	Size on disk	Create Time	Modify Time
desktop.ini	62	64	2012-08-15 22:17:51	2012-08-16 03:39:02
Ijouoc.exe	117266	118784	2013-07-08 21:48:16	2013-07-08 21:48:16

Step 13 – VirusTotal submission

Submitting the malicious file from the infected system to virustotal confirms the file to be malicious

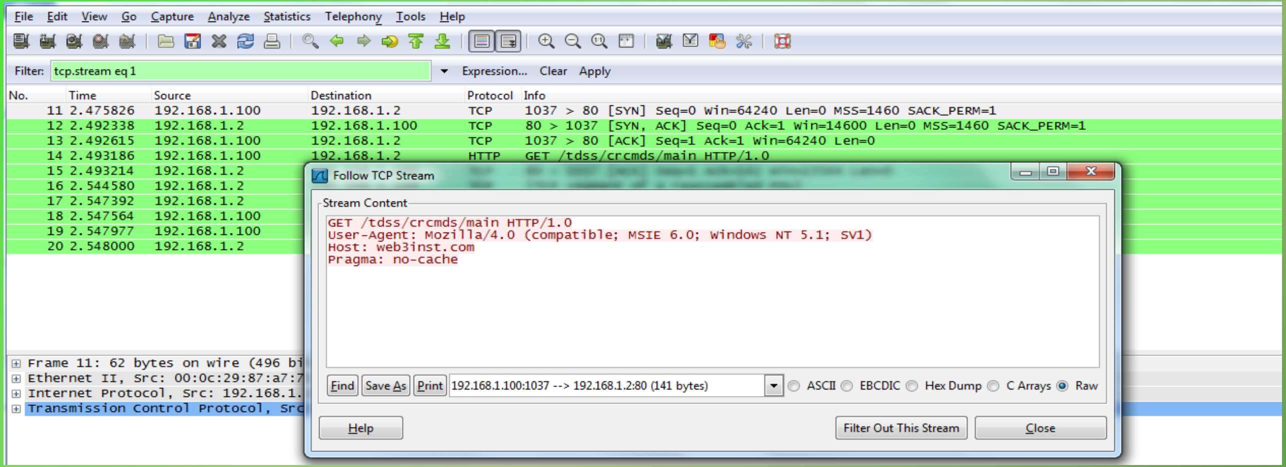


Antivirus	Result	Update
AhnLab-V3	Trojan/Win32.VB	20110708
AntiVir	TR/Spy.Revs.A	20110708
Antiy-AVL	Trojan/Win32.VB.gen	20110708
Avast	Win32:VB-VZN [Trj]	20110708
Avast5	Win32:VB-VZN [Trj]	20110708
AVG	Generic22.CLPW	20110708
BitDefender	Backdoor.IRCBot.ADED	20110709
CAT-QuickHeal	✓	20110709
ClamAV	BC.Heuristic.Trojan.SusPacked.BF-6.B	20110709
Commtouch	✓	20110709
Comodo	UnclassifiedMalware	20110709
DrWeb	Trojan.Siggen2.41279	20110709
Emsisoft	Backdoor.IRCBot!IK	20110708
eSafe	✓	20110707
eTrust-Vet	✓	20110708
F-Prot	✓	20110708

DEMO 2

Demo-Scenario 2

Your security device alerts on malicious http connection to the domain “web3inst.com” which resolves to 192.168.1.2, communication is detected from a source ip 192.168.1.100 (shown below)..you are asked to investigate and perform memory forensics on the machine 192.168.1.100


File Edit View Go Capture Analyze Statistics Telephony Tools Help
Expression... Clear Apply
Follow TCP Stream
Find Save As Print
Help
Filter Out This Stream
Close

Filter: tcp.stream eq 1

```
No.     Time      Source          Destination     Protocol Info
     11 2.475826  192.168.1.100   192.168.1.2     TCP      1037 > 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
     12 2.492338  192.168.1.2     192.168.1.100   TCP      80 > 1037 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SACK_PERM=1
     13 2.492615  192.168.1.100   192.168.1.2     TCP      1037 > 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
     14 2.493186  192.168.1.100   192.168.1.2     HTTP     GET /tdss/crcmds/main HTTP/1.0
     15 2.493214  192.168.1.2
     16 2.544580  192.168.1.2
     17 2.547392  192.168.1.2
     18 2.547564  192.168.1.100
     19 2.547977  192.168.1.100
     20 2.548000  192.168.1.2
```

```
⊞ Frame 11: 62 bytes on wire (496 bi
⊞ Ethernet II, Src: 00:0c:29:87:a7:7
⊞ Internet Protocol, Src: 192.168.1.
⊞ Transmission Control Protocol, Src
```

Stream Content

```
GET /tdss/crcmds/main HTTP/1.0
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: web3inst.com
Pragma: no-cache
```

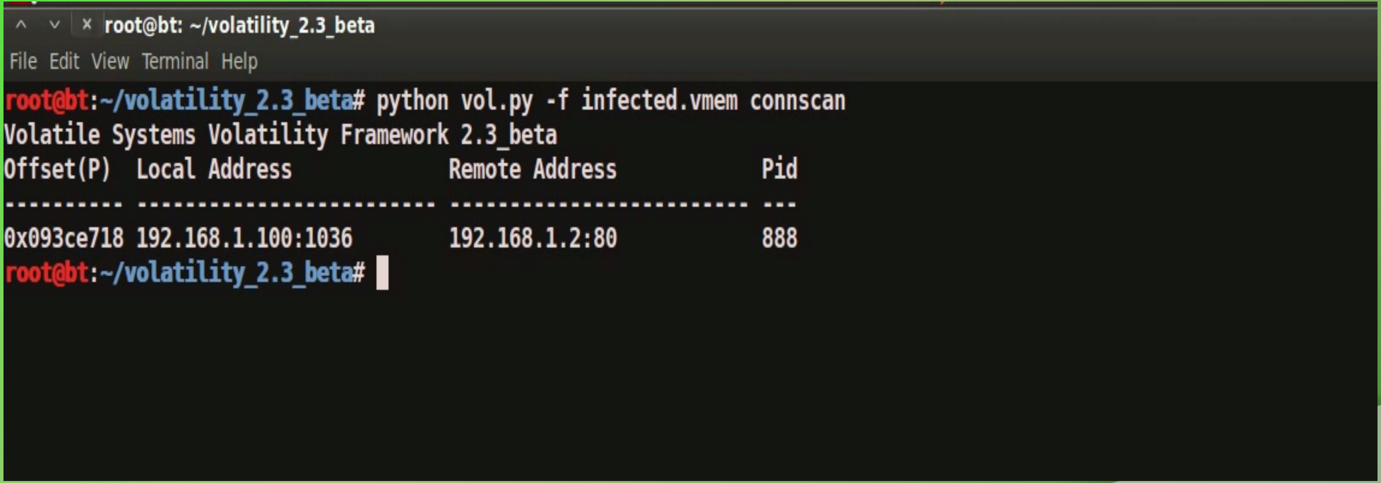
192.168.1.100:1037 --> 192.168.1.2:80 (141 bytes) ○ ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays ● Raw

- To start with, acquire the memory image “infected.dmp” from 192.168.1.100, using memory acquisition tools (like Dumpit or win32dd)

- Analyze the memory dump “infected.dmp”

Step 1 – Network connections

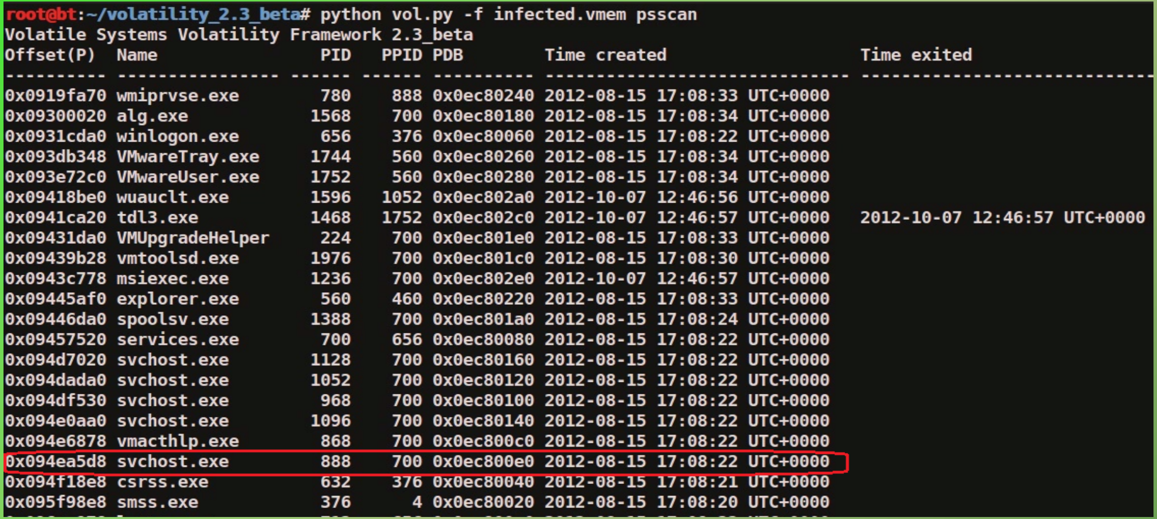
Volatility's connscan module shows connection to the malicious http connection by pid 888



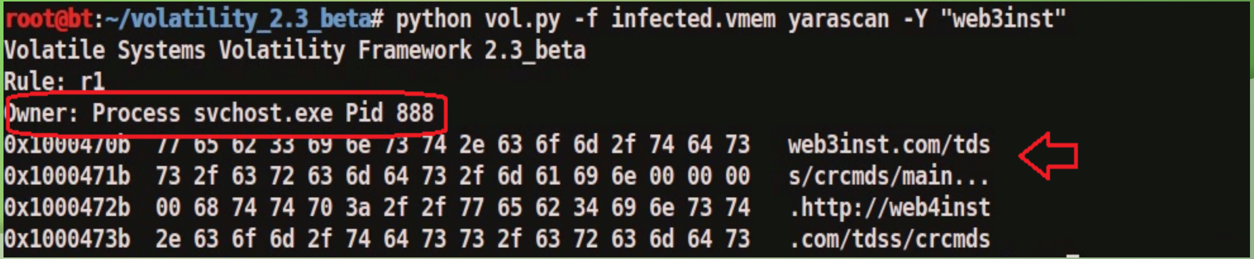
```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem connscan
Volatile Systems Volatility Framework 2.3_beta
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------- ------------------------- ---
0x093ce718 192.168.1.100:1036        192.168.1.2:80            888
root@bt:~/volatility_2.3_beta#
```

Step 2 – process determination and YARA scan

Volatility’s psscan shows pid 888 is associated with svchost.exe and YARA scan shows that malicious domain is found in the address space of pid 888 (svchost.exe)



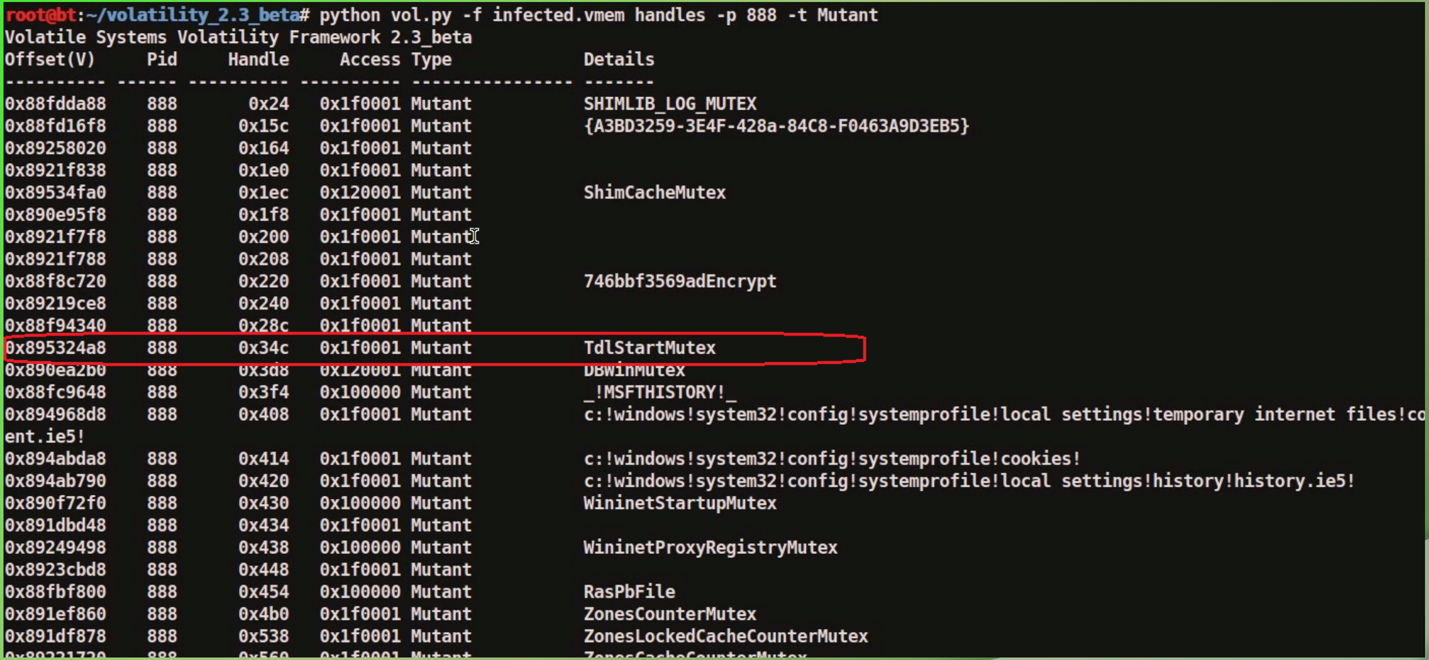
```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem psscan
Volatile Systems Volatility Framework 2.3_beta
Offset(P)  Name                PID   PPID PDB        Time created                   Time exited
---------- ---------------- ------ ------ ---------- ------------------------------ ------------------------------
0x0919fa70 wmiprvse.exe        780    888 0x0ec80240 2012-08-15 17:08:33 UTC+0000
0x09300020 alg.exe            1568    700 0x0ec80180 2012-08-15 17:08:34 UTC+0000
0x0931cda0 winlogon.exe        656    376 0x0ec80060 2012-08-15 17:08:22 UTC+0000
0x093db348 VMwareTray.exe     1744    560 0x0ec80260 2012-08-15 17:08:34 UTC+0000
0x093e72c0 VMwareUser.exe     1752    560 0x0ec80280 2012-08-15 17:08:34 UTC+0000
0x09418be0 wuauclt.exe        1596   1052 0x0ec802a0 2012-10-07 12:46:56 UTC+0000
0x0941ca20 tdl3.exe           1468   1752 0x0ec802c0 2012-10-07 12:46:57 UTC+0000   2012-10-07 12:46:57 UTC+0000
0x09431da0 VMUpgradeHelper     224    700 0x0ec801e0 2012-08-15 17:08:33 UTC+0000
0x09439b28 vmtoolsd.exe       1976    700 0x0ec801c0 2012-08-15 17:08:30 UTC+0000
0x0943c778 msiexec.exe        1236    700 0x0ec802e0 2012-10-07 12:46:57 UTC+0000
0x09445af0 explorer.exe        560    460 0x0ec80220 2012-08-15 17:08:33 UTC+0000
0x09446da0 spoolsv.exe        1388    700 0x0ec801a0 2012-08-15 17:08:24 UTC+0000
0x09457520 services.exe        700    656 0x0ec80080 2012-08-15 17:08:22 UTC+0000
0x094d7020 svchost.exe        1128    700 0x0ec80160 2012-08-15 17:08:22 UTC+0000
0x094dada0 svchost.exe        1052    700 0x0ec80120 2012-08-15 17:08:22 UTC+0000
0x094df530 svchost.exe         968    700 0x0ec80100 2012-08-15 17:08:22 UTC+0000
0x094e0aa0 svchost.exe        1096    700 0x0ec80140 2012-08-15 17:08:22 UTC+0000
0x094e6878 vmacthlp.exe        868    700 0x0ec800c0 2012-08-15 17:08:22 UTC+0000
0x094ea5d8 svchost.exe         888    700 0x0ec800e0 2012-08-15 17:08:22 UTC+0000
0x094f18e8 csrss.exe           632    376 0x0ec80040 2012-08-15 17:08:21 UTC+0000
0x095f98e8 smss.exe            376      4 0x0ec80020 2012-08-15 17:08:20 UTC+0000
```



```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem yarascan -Y "web3inst"
Volatile Systems Volatility Framework 2.3_beta
Rule: r1
Owner: Process svchost.exe Pid 888
0x1000470b  77 65 62 33 69 6e 73 74 2e 63 6f 6d 2f 74 64 73   web3inst.com/tds
0x1000471b  73 2f 63 72 63 6d 64 73 2f 6d 61 69 6e 00 00 00   s/crcmds/main...
0x1000472b  00 68 74 74 70 3a 2f 2f 77 65 62 34 69 6e 73 74   .http://web4inst
0x1000473b  2e 63 6f 6d 2f 74 64 73 73 2f 63 72 63 6d 64 73   .com/tdss/crcmds
```

Step 3 – Suspicious mutex in svchost.exe

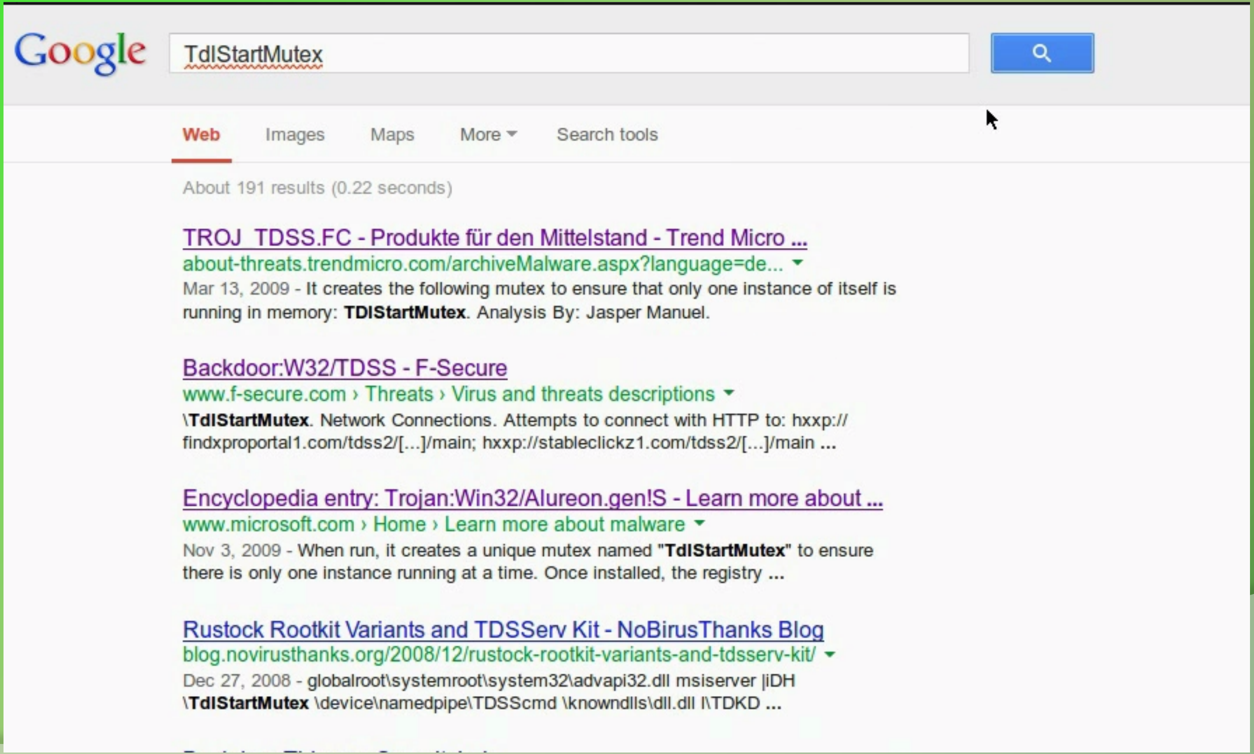
Volatility’s mutantscan shows suspicious mutex



```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem handles -p 888 -t Mutant
Volatile Systems Volatility Framework 2.3_beta
Offset(V)     Pid     Handle     Access Type             Details
---------- ------ ---------- ---------- ---------------- -------
0x88fdda88    888       0x24   0x1f0001 Mutant           SHIMLIB_LOG_MUTEX
0x88fd16f8    888      0x15c   0x1f0001 Mutant           {A3BD3259-3E4F-428a-84C8-F0463A9D3EB5}
0x89258020    888      0x164   0x1f0001 Mutant
0x8921f838    888      0x1e0   0x1f0001 Mutant
0x89534fa0    888      0x1ec   0x120001 Mutant           ShimCacheMutex
0x890e95f8    888      0x1f8   0x1f0001 Mutant
0x8921f7f8    888      0x200   0x1f0001 Mutant
0x8921f788    888      0x208   0x1f0001 Mutant
0x88f8c720    888      0x220   0x1f0001 Mutant           746bbf3569adEncrypt
0x89219ce8    888      0x240   0x1f0001 Mutant
0x88f94340    888      0x28c   0x1f0001 Mutant
0x895324a8    888      0x34c   0x1f0001 Mutant           TdlStartMutex
0x890ea2b0    888      0x3d8   0x120001 Mutant           DBWinMutex
0x88fc9648    888      0x3f4   0x100000 Mutant           _!MSFTHISTORY!_
0x894968d8    888      0x408   0x1f0001 Mutant           c:!windows!system32!config!systemprofile!local settings!temporary internet files!co
ent.ie5!
0x894abda8    888      0x414   0x1f0001 Mutant           c:!windows!system32!config!systemprofile!cookies!
0x894ab790    888      0x420   0x1f0001 Mutant           c:!windows!system32!config!systemprofile!local settings!history!history.ie5!
0x890f72f0    888      0x430   0x100000 Mutant           WininetStartupMutex
0x891dbd48    888      0x434   0x1f0001 Mutant
0x89249498    888      0x438   0x100000 Mutant           WininetProxyRegistryMutex
0x8923cbd8    888      0x448   0x1f0001 Mutant
0x88fbf800    888      0x454   0x100000 Mutant           RasPbFile
0x891ef860    888      0x4b0   0x1f0001 Mutant           ZonesCounterMutex
0x891df878    888      0x538   0x1f0001 Mutant           ZonesLockedCacheCounterMutex
0x89221720    888      0x560   0x1f0001 Mutant           ZonesCacheCounterMutex
```

Step 4 – malicious mutex

Google search shows that this suspicious mutex is associated with TDSS rootkit



About 191 results (0.22 seconds)

TROJ_TDSS.FC - Produkte für den Mittelstand - Trend Micro ...
about-threats.trendmicro.com/archiveMalware.aspx?language=de...
Mar 13, 2009 - It creates the following mutex to ensure that only one instance of itself is running in memory: **TDIStartMutex**. Analysis By: Jasper Manuel.

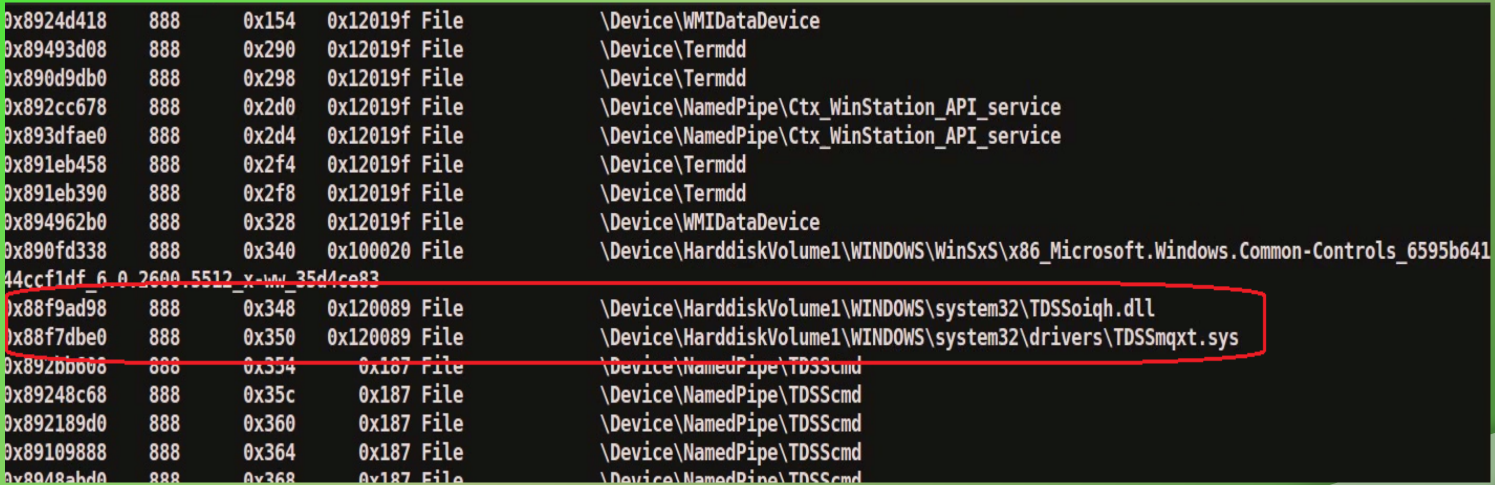
Backdoor:W32/TDSS - F-Secure
www.f-secure.com › Threats › Virus and threats descriptions
TdlStartMutex. Network Connections. Attempts to connect with HTTP to: hxxp://findxproportal1.com/tdss2/[...]/main; hxxp://stableclickz1.com/tdss2/[...]/main ...

Encyclopedia entry: Trojan:Win32/Alureon.gen!S - Learn more about ...
www.microsoft.com › Home › Learn more about malware
Nov 3, 2009 - When run, it creates a unique mutex named "**TdlStartMutex**" to ensure there is only one instance running at a time. Once installed, the registry ...

Rustock Rootkit Variants and TDSServ Kit - NoBirusThanks Blog
blog.novirusthanks.org/2008/12/rustock-rootkit-variants-and-tdsserv-kit/
Dec 27, 2008 - globalroot\systemroot\system32\advapi32.dll msiserver |iDH **TdlStartMutex** \device\namedpipe\TDSScmd \knowndlls\dll.dll I\TDKD ...

Step 5 – File handles

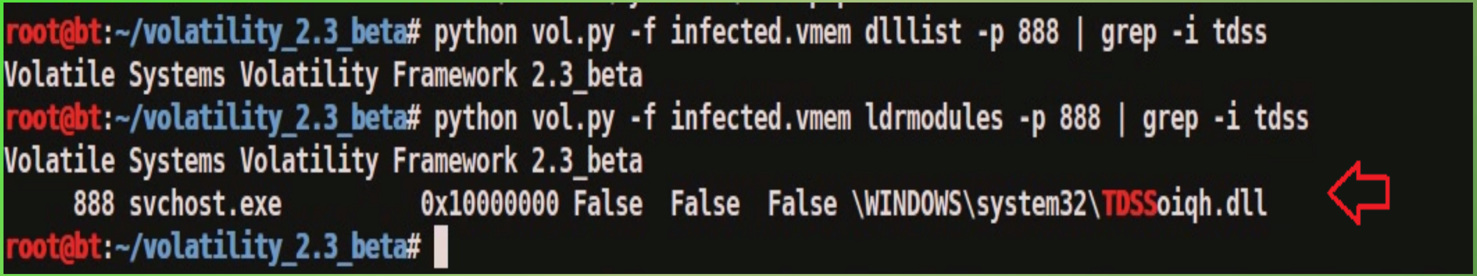
Examining file handles in svchost.exe (pid 888) shows handles to suspicious files (starting with TDSS)



```
0x8924d418    888      0x154   0x12019f File            \Device\WMIDataDevice
0x89493d08    888      0x290   0x12019f File            \Device\Termdd
0x890d9db0    888      0x298   0x12019f File            \Device\Termdd
0x892cc678    888      0x2d0   0x12019f File            \Device\NamedPipe\Ctx_WinStation_API_service
0x893dfae0    888      0x2d4   0x12019f File            \Device\NamedPipe\Ctx_WinStation_API_service
0x891eb458    888      0x2f4   0x12019f File            \Device\Termdd
0x891eb390    888      0x2f8   0x12019f File            \Device\Termdd
0x894962b0    888      0x328   0x12019f File            \Device\WMIDataDevice
0x890fd338    888      0x340   0x100020 File            \Device\HarddiskVolume1\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b641
44ccf1df_6.0.2600.5512_x-ww_35d4ce83
0x88f9ad98    888      0x348   0x120089 File            \Device\HarddiskVolume1\WINDOWS\system32\TDSSoiqh.dll
0x88f7dbe0    888      0x350   0x120089 File            \Device\HarddiskVolume1\WINDOWS\system32\drivers\TDSSmqxt.sys
0x892bb608    888      0x354      0x187 File            \Device\NamedPipe\TDSScmd
0x89248c68    888      0x35c      0x187 File            \Device\NamedPipe\TDSScmd
0x892189d0    888      0x360      0x187 File            \Device\NamedPipe\TDSScmd
0x89109888    888      0x364      0x187 File            \Device\NamedPipe\TDSScmd
0x8948abd0    888      0x368      0x187 File            \Device\NamedPipe\TDSScmd
```

Step 6 – Hidden DLL

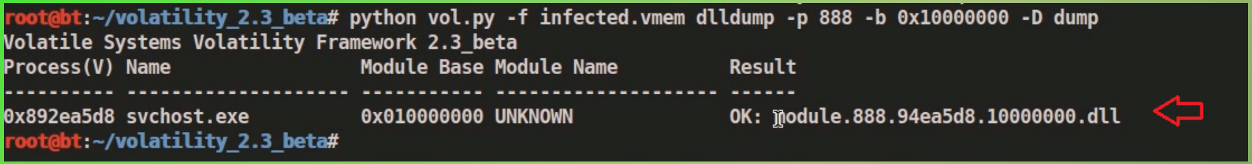
Volatility’s dlllist module couldn’t find the DLL starting with “TDSS” whereas ldrmodules plugin was able to find it. This confirms that the DLL (TDSSoiqh.dll) was hidden, malware hides the DLL by unlinking from the 3 PEB lists



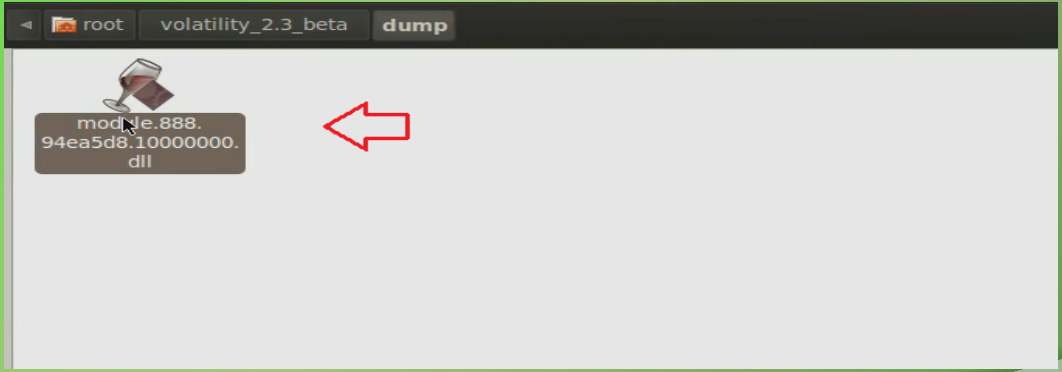
```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem dlllist -p 888 | grep -i tdss
Volatile Systems Volatility Framework 2.3_beta
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem ldrmodules -p 888 | grep -i tdss
Volatile Systems Volatility Framework 2.3_beta
    888 svchost.exe          0x10000000 False  False  False \WINDOWS\system32\TDSSoiqh.dll
root@bt:~/volatility_2.3_beta#
```

Step 7– Dumping the hidden DLL

Volatility’s dlldump module dumps the hidden dll

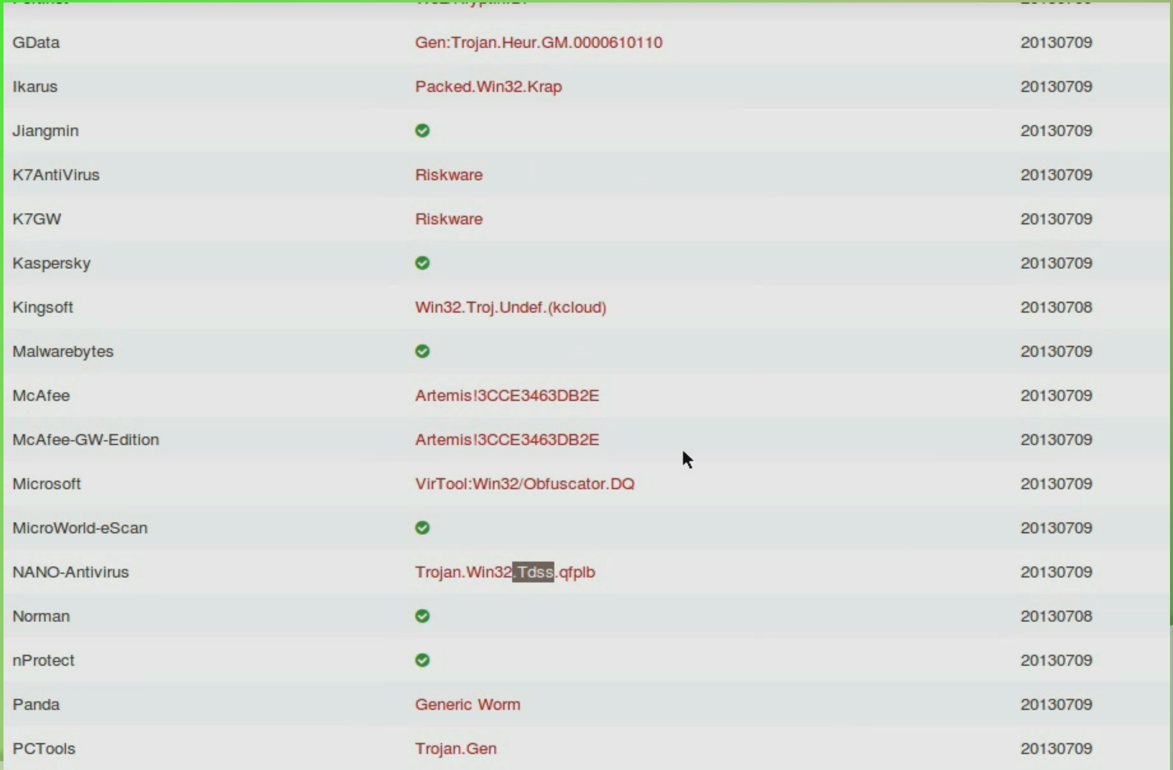


```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem dlldump -p 888 -b 0x10000000 -D dump
Volatile Systems Volatility Framework 2.3_beta
Process(V) Name                 Module Base Module Name          Result
---------- -------------------- ----------- -------------------- ------
0x892ea5d8 svchost.exe          0x010000000 UNKNOWN              OK: module.888.94ea5d8.10000000.dll
root@bt:~/volatility_2.3_beta#
```



Step 8– VirusTotal submission of DLL

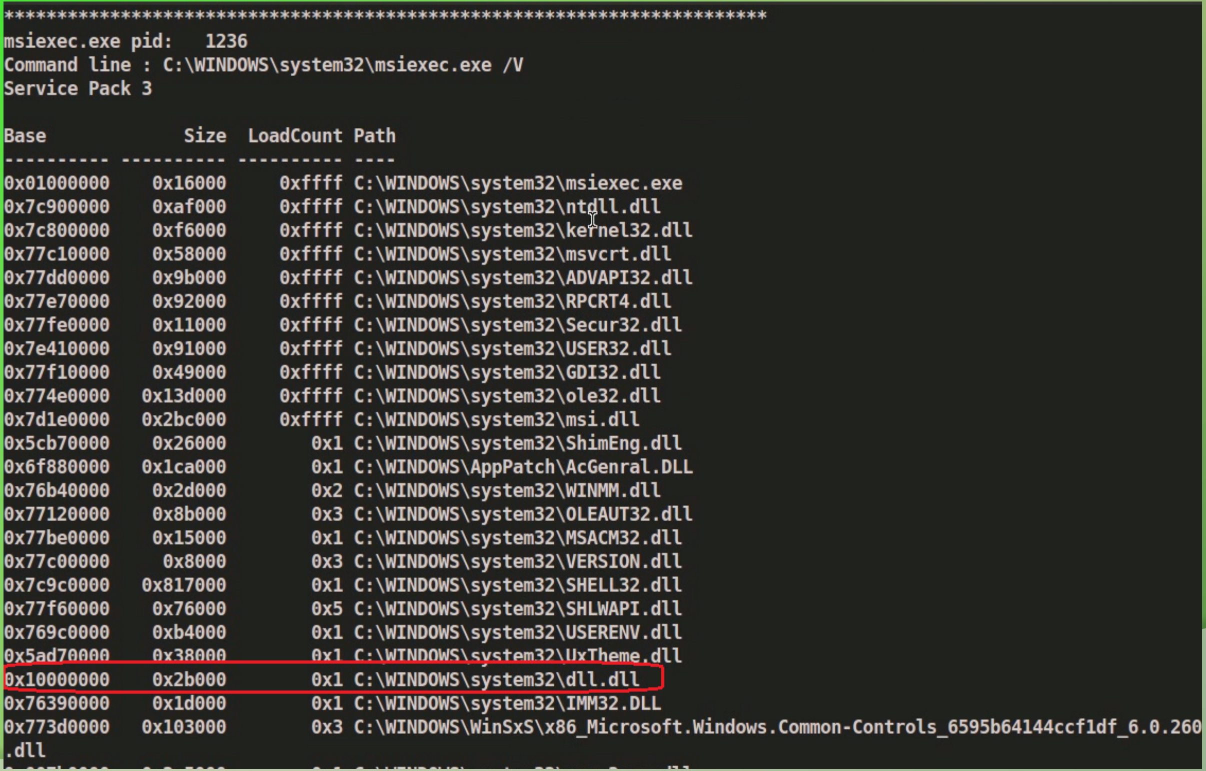
Submitting the dumped dll to VirusTotal confirms that it is malicious



GData	Gen:Trojan.Heur.GM.0000610110	20130709
Ikarus	Packed.Win32.Krap	20130709
Jiangmin	✓	20130709
K7AntiVirus	Riskware	20130709
K7GW	Riskware	20130709
Kaspersky	✓	20130709
Kingsoft	Win32.Troj.Undef.(kcloud)	20130708
Malwarebytes	✓	20130709
McAfee	Artemis!3CCE3463DB2E	20130709
McAfee-GW-Edition	Artemis!3CCE3463DB2E	20130709
Microsoft	VirTool:Win32/Obfuscator.DQ	20130709
MicroWorld-eScan	✓	20130709
NANO-Antivirus	Trojan.Win32.Tdss.qfplb	20130709
Norman	✓	20130708
nProtect	✓	20130709
Panda	Generic Worm	20130709
PCTools	Trojan.Gen	20130709

Step 9 – Suspicious DLL loaded by msiexec

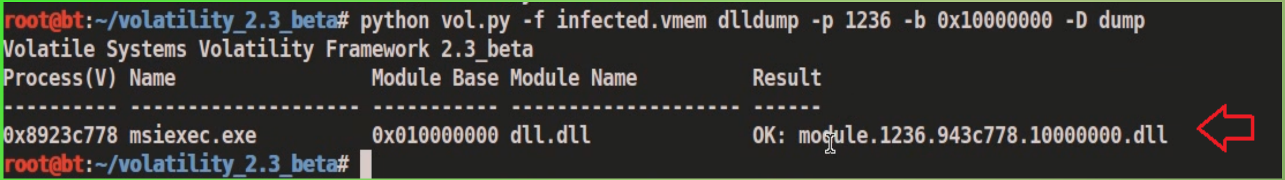
dlllist shows suspicious dll loaded by msiexec.exe



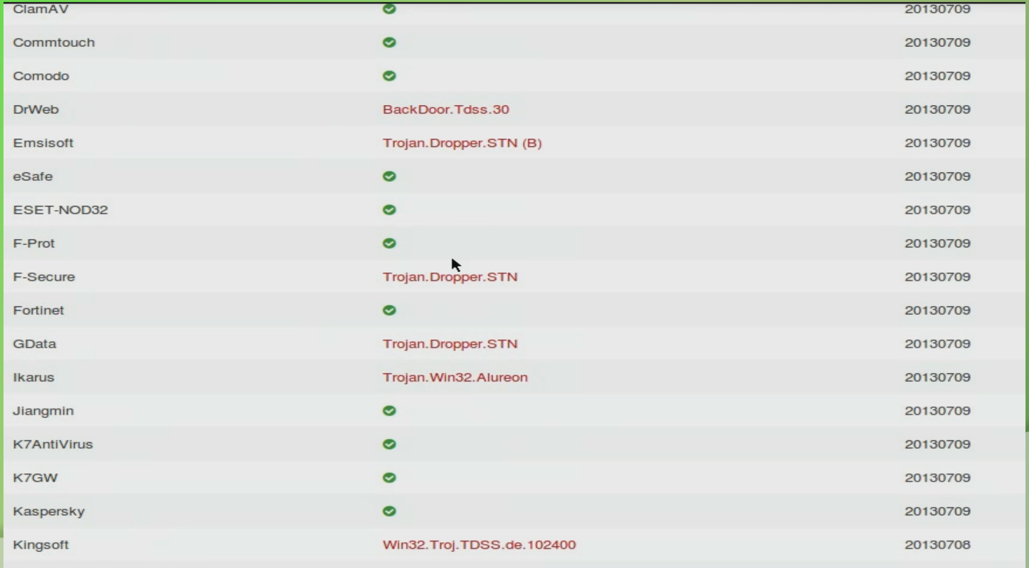
```
************************************************************************
msiexec.exe pid:   1236
Command line : C:\WINDOWS\system32\msiexec.exe /V
Service Pack 3

Base             Size  LoadCount Path
---------- ---------- ---------- ----
0x01000000    0x16000     0xffff C:\WINDOWS\system32\msiexec.exe
0x7c900000    0xaf000     0xffff C:\WINDOWS\system32\ntdll.dll
0x7c800000    0xf6000     0xffff C:\WINDOWS\system32\kernel32.dll
0x77c10000    0x58000     0xffff C:\WINDOWS\system32\msvcrt.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
0x7e410000    0x91000     0xffff C:\WINDOWS\system32\USER32.dll
0x77f10000    0x49000     0xffff C:\WINDOWS\system32\GDI32.dll
0x774e0000   0x13d000     0xffff C:\WINDOWS\system32\ole32.dll
0x7d1e0000   0x2bc000     0xffff C:\WINDOWS\system32\msi.dll
0x5cb70000    0x26000        0x1 C:\WINDOWS\system32\ShimEng.dll
0x6f880000   0x1ca000        0x1 C:\WINDOWS\AppPatch\AcGenral.DLL
0x76b40000    0x2d000        0x2 C:\WINDOWS\system32\WINMM.dll
0x77120000    0x8b000        0x3 C:\WINDOWS\system32\OLEAUT32.dll
0x77be0000    0x15000        0x1 C:\WINDOWS\system32\MSACM32.dll
0x77c00000     0x8000        0x3 C:\WINDOWS\system32\VERSION.dll
0x7c9c0000   0x817000        0x1 C:\WINDOWS\system32\SHELL32.dll
0x77f60000    0x76000        0x5 C:\WINDOWS\system32\SHLWAPI.dll
0x769c0000    0xb4000        0x1 C:\WINDOWS\system32\USERENV.dll
0x5ad70000    0x38000        0x1 C:\WINDOWS\system32\UxTheme.dll
0x10000000    0x2b000        0x1 C:\WINDOWS\system32\dll.dll
0x76390000    0x1d000        0x1 C:\WINDOWS\system32\IMM32.DLL
0x773d0000   0x103000        0x3 C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.260
.dll
```

Step 10– Dumping DLL and VT submission

Dumping the suspicious DLL (dll.dll) and submitting to VirusTotal confirms that this is associated with TDSS rootkit



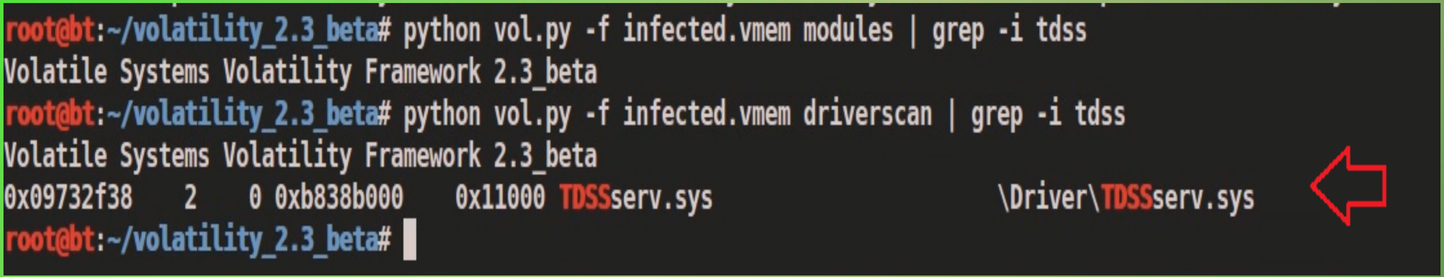
```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem dlldump -p 1236 -b 0x10000000 -D dump
Volatile Systems Volatility Framework 2.3_beta
Process(V) Name                Module Base Module Name          Result
---------- ------------------- ----------- -------------------- ------
0x8923c778 msiexec.exe         0x010000000 dll.dll              OK: module.1236.943c778.10000000.dll
root@bt:~/volatility_2.3_beta#
```



ClamAV	✓	20130709
Commtouch	✓	20130709
Comodo	✓	20130709
DrWeb	BackDoor.Tdss.30	20130709
Emsisoft	Trojan.Dropper.STN (B)	20130709
eSafe	✓	20130709
ESET-NOD32	✓	20130709
F-Prot	✓	20130709
F-Secure	Trojan.Dropper.STN	20130709
Fortinet	✓	20130709
GData	Trojan.Dropper.STN	20130709
Ikarus	Trojan.Win32.Alureon	20130709
Jiangmin	✓	20130709
K7AntiVirus	✓	20130709
K7GW	✓	20130709
Kaspersky	✓	20130709
Kingsoft	Win32.Troj.TDSS.de.102400	20130708

Step 11– Hidden Kernel driver

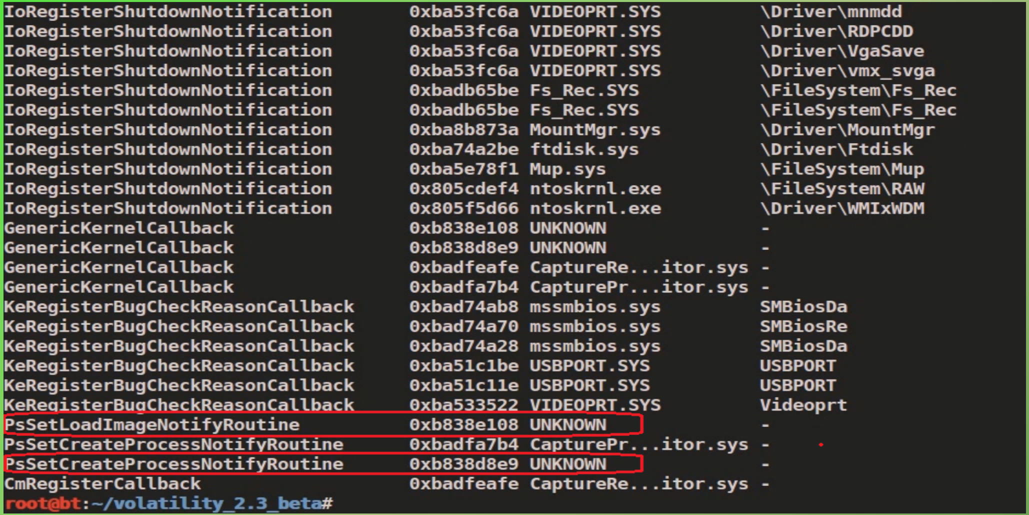
Volatility's modules plugin couldn't find the drivers starting with "TDSS" whereas driverscan plugin was able to find it. This confirms that the kernel driver (TDSSserv.sys) was hidden



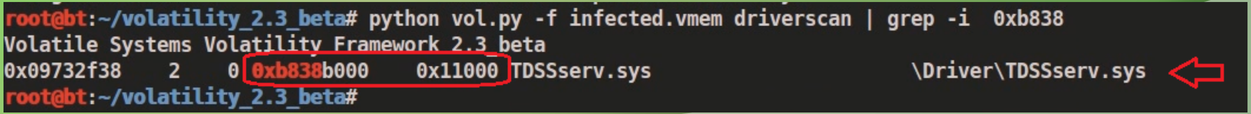
```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem modules | grep -i tdss
Volatile Systems Volatility Framework 2.3_beta
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem driverscan | grep -i tdss
Volatile Systems Volatility Framework 2.3_beta
0x09732f38    2    0 0xb838b000    0x11000 TDSSserv.sys            \Driver\TDSSserv.sys
root@bt:~/volatility_2.3_beta# 
```

Step 12– Kernel Callbacks

Callbacks were set by an unknown driver. The below screenshot shows that this unknown driver falls under the address range of TDSSserv.sys



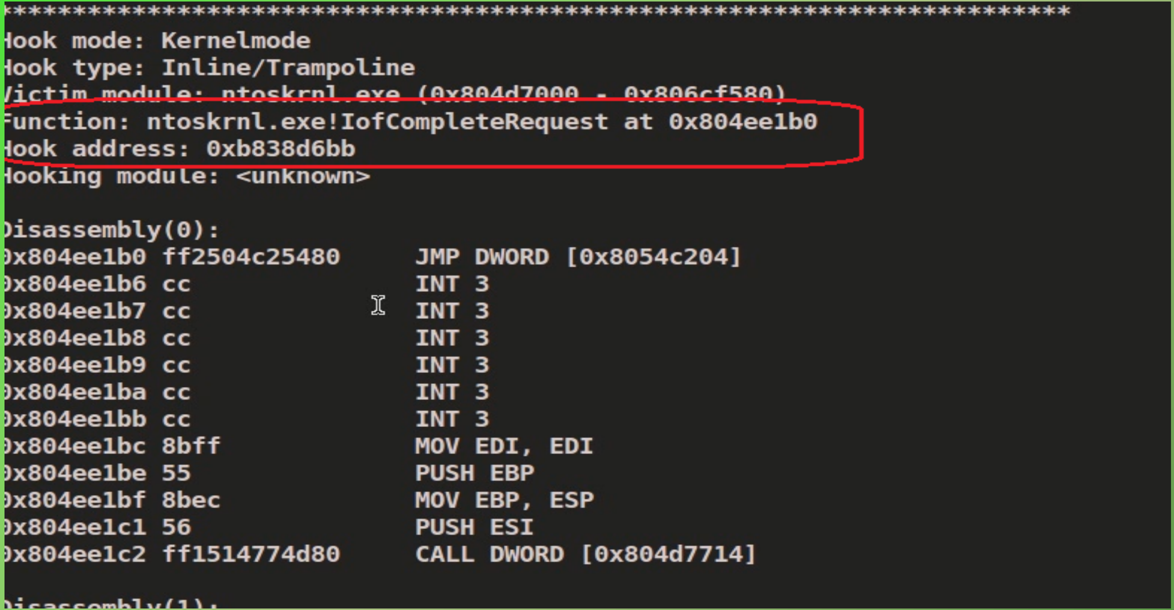
```
IoRegisterShutdownNotification         0xba53fc6a VIDEOPRT.SYS          \Driver\mnmdd
IoRegisterShutdownNotification         0xba53fc6a VIDEOPRT.SYS          \Driver\RDPCDD
IoRegisterShutdownNotification         0xba53fc6a VIDEOPRT.SYS          \Driver\VgaSave
IoRegisterShutdownNotification         0xba53fc6a VIDEOPRT.SYS          \Driver\vmx_svga
IoRegisterShutdownNotification         0xbadb65be Fs_Rec.SYS            \FileSystem\Fs_Rec
IoRegisterShutdownNotification         0xbadb65be Fs_Rec.SYS            \FileSystem\Fs_Rec
IoRegisterShutdownNotification         0xba8b873a MountMgr.sys          \Driver\MountMgr
IoRegisterShutdownNotification         0xba74a2be ftdisk.sys            \Driver\Ftdisk
IoRegisterShutdownNotification         0xba5e78f1 Mup.sys               \FileSystem\Mup
IoRegisterShutdownNotification         0x805cdef4 ntoskrnl.exe          \FileSystem\RAW
IoRegisterShutdownNotification         0x805f5d66 ntoskrnl.exe          \Driver\WMIxWDM
GenericKernelCallback                  0xb838e108 UNKNOWN               -
GenericKernelCallback                  0xb838d8e9 UNKNOWN               -
GenericKernelCallback                  0xbadfeafe CaptureRe...itor.sys -
GenericKernelCallback                  0xbadfa7b4 CapturePr...itor.sys -
KeRegisterBugCheckReasonCallback       0xbad74ab8 mssmbios.sys          SMBiosDa
KeRegisterBugCheckReasonCallback       0xbad74a70 mssmbios.sys          SMBiosRe
KeRegisterBugCheckReasonCallback       0xbad74a28 mssmbios.sys          SMBiosDa
KeRegisterBugCheckReasonCallback       0xba51c1be USBPORT.SYS           USBPORT
KeRegisterBugCheckReasonCallback       0xba51c11e USBPORT.SYS           USBPORT
KeRegisterBugCheckReasonCallback       0xba533522 VIDEOPRT.SYS          Videoprt
PsSetLoadImageNotifyRoutine            0xb838e108 UNKNOWN               -
PsSetCreateProcessNotifyRoutine        0xbadfa7b4 CapturePr...itor.sys -
PsSetCreateProcessNotifyRoutine        0xb838d8e9 UNKNOWN               -
CmRegisterCallback                     0xbadfeafe CaptureRe...itor.sys -
root@bt:~/volatility_2.3_beta#
```



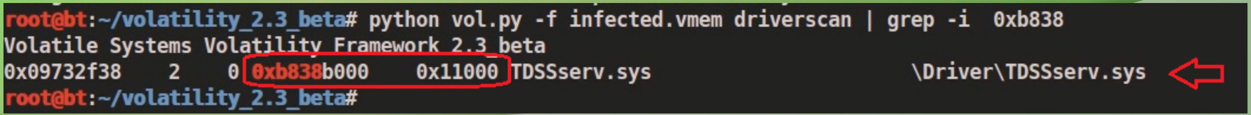
```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem driverscan | grep -i  0xb838
Volatile Systems Volatility Framework 2.3_beta
0x09732f38     2     0 0xb838b000     0x11000 TDSSserv.sys                  \Driver\TDSSserv.sys
root@bt:~/volatility_2.3_beta#
```

Step 13– Kernel API hooks

Malware hooks the Kernel API and the hook address falls under the address range of TDSSserv.sys



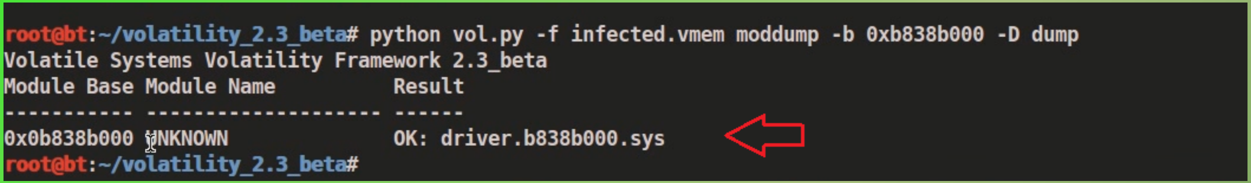
```
************************************************************************
Hook mode: Kernelmode
Hook type: Inline/Trampoline
Victim module: ntoskrnl.exe (0x804d7000 - 0x806cf580)
Function: ntoskrnl.exe!IofCompleteRequest at 0x804ee1b0
Hook address: 0xb838d6bb
Hooking module: <unknown>

Disassembly(0):
0x804ee1b0 ff2504c25480     JMP DWORD [0x8054c204]
0x804ee1b6 cc               INT 3
0x804ee1b7 cc               INT 3
0x804ee1b8 cc               INT 3
0x804ee1b9 cc               INT 3
0x804ee1ba cc               INT 3
0x804ee1bb cc               INT 3
0x804ee1bc 8bff             MOV EDI, EDI
0x804ee1be 55               PUSH EBP
0x804ee1bf 8bec             MOV EBP, ESP
0x804ee1c1 56               PUSH ESI
0x804ee1c2 ff1514774d80     CALL DWORD [0x804d7714]

Disassembly(1):
```



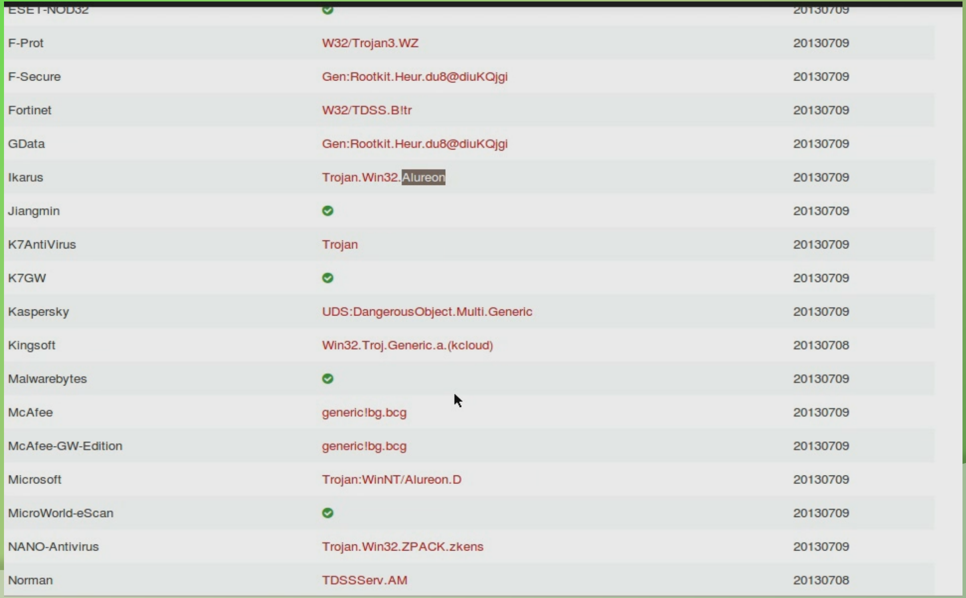
```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem driverscan | grep -i  0xb838
Volatile Systems Volatility Framework 2.3_beta
0x09732f38    2    0 0xb838b000    0x11000 TDSSserv.sys                 \Driver\TDSSserv.sys
root@bt:~/volatility_2.3_beta#
```

Step 14– Dumping the kernel driver

Dumping the kernel driver and submitting it to VirusTotal confirms that it is TDSS (Alureon) rootkit



```
root@bt:~/volatility_2.3_beta# python vol.py -f infected.vmem moddump -b 0xb838b000 -D dump
Volatile Systems Volatility Framework 2.3_beta
Module Base Module Name          Result
----------- -------------------- ------
0x0b838b000 UNKNOWN              OK: driver.b838b000.sys
root@bt:~/volatility_2.3_beta#
```



ESET-NOD32	✓	20130709
F-Prot	W32/Trojan3.WZ	20130709
F-Secure	Gen:Rootkit.Heur.du8@diuKQjgi	20130709
Fortinet	W32/TDSS.B!tr	20130709
GData	Gen:Rootkit.Heur.du8@diuKQjgi	20130709
Ikarus	Trojan.Win32.Alureon	20130709
Jiangmin	✓	20130709
K7AntiVirus	Trojan	20130709
K7GW	✓	20130709
Kaspersky	UDS:DangerousObject.Multi.Generic	20130709
Kingsoft	Win32.Troj.Generic.a.(kcloud)	20130708
Malwarebytes	✓	20130709
McAfee	generic!bg.bcg	20130709
McAfee-GW-Edition	generic!bg.bcg	20130709
Microsoft	Trojan:WinNT/Alureon.D	20130709
MicroWorld-eScan	✓	20130709
NANO-Antivirus	Trojan.Win32.ZPACK.zkens	20130709
Norman	TDSSServ.AM	20130708

Reference

Complete Reference Guide for Advanced Malware Analysis Training

[Include links for all the Demos & Tools]

Thank You !

www.SecurityXploded.com