

Practical Reversing III – Malware Memory Forensics

Monnappa (m0nna)



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 & Garage4Hackers** community for their extended support and cooperation.
- Thanks to all the trainers who have devoted their precious time and countless hours to make it happen.

Reversing & Malware Analysis Training

This presentation is part of our **Reverse Engineering & Malware Analysis** Training program. Currently it is delivered only during our local meet 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

Contents

- ④ Why Memory Forensics?
- ④ Steps in Memory Forensics
- ④ Volatility Quick Overview
- ④ Volatility help and plugins
- ④ Demo

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 artifacts**
 - 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
File Edit View Terminal Help
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

<http://youtu.be/YcVusDjnBxw>

Demo-Scenario

Your security device alerts, show malicious http connection to ip address 208.91.197.54 from a source ip 192.168.1.100 on 8th june 2012 at around 13:30hrs...you are asked to investigate and do memory forensics on that machine 192.168.1.100

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

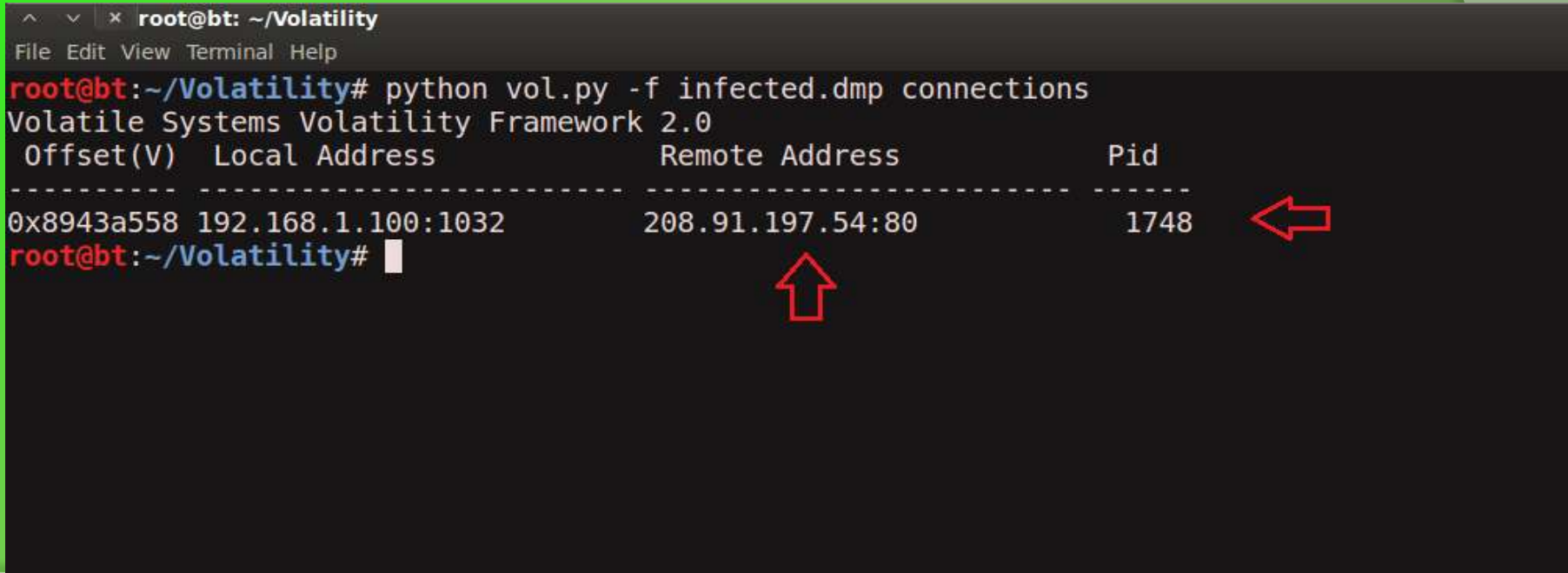
command: win32dd.exe /f infected.dmp

- Analyze the memory dump “infected.dmp”

Step 1 – Start With what you know

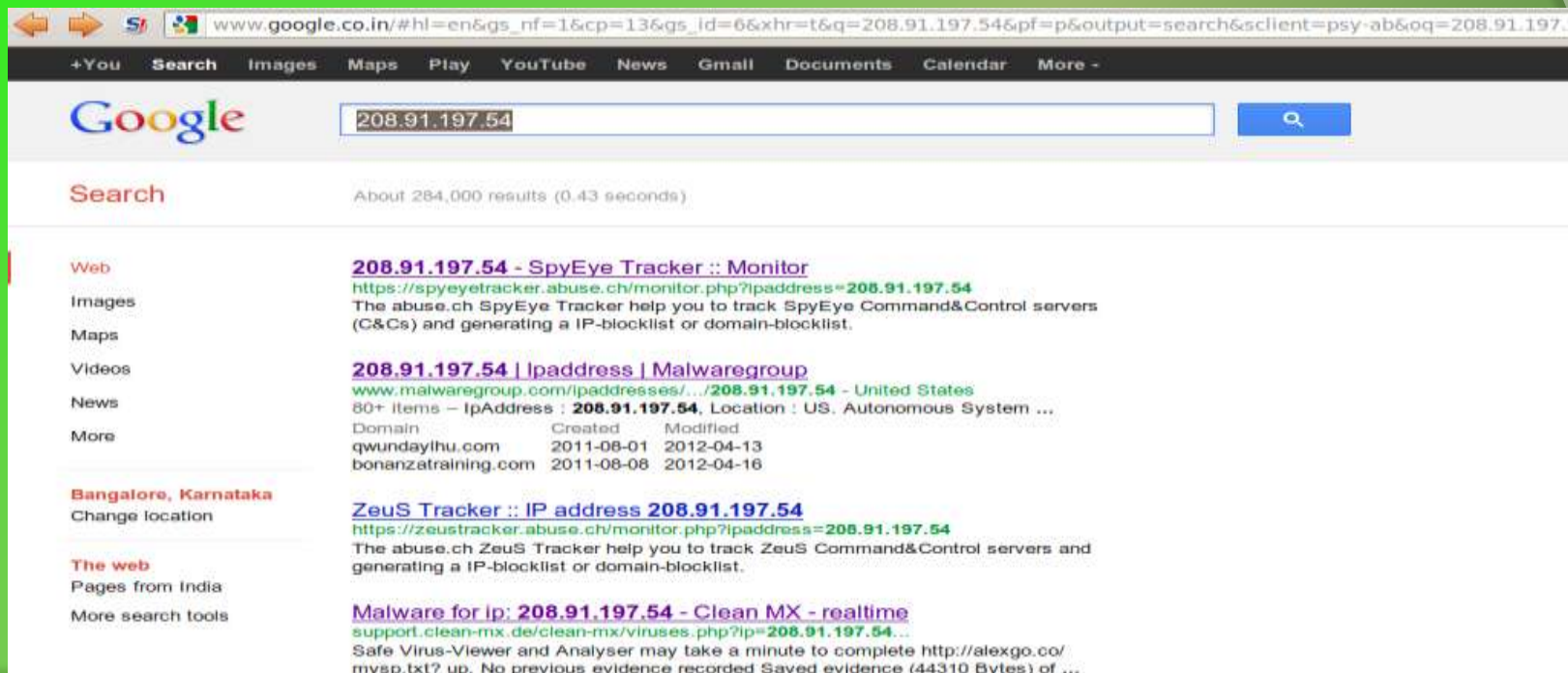
Volatility's connections module shows connection to the malicious ip by pid 1748

```
root@bt: ~/Volatility
File Edit View Terminal Help
root@bt:~/Volatility# python vol.py -f infected.dmp connections
Volatile Systems Volatility Framework 2.0
Offset(V)  Local Address          Remote Address          Pid
-----
0x8943a558 192.168.1.100:1032     208.91.197.54:80       1748
root@bt:~/Volatility#
```



Step 2 – Info about 208.91.197.54

Google search shows 208.91.197.54 associated with malware, probably “spyeye”, we need to confirm that yet.



www.google.co.in/#hl=en&gs_nf=1&cp=13&gs_id=66&xhr=t&q=208.91.197.54&pf=p&output=search&scient=psy-ab&oq=208.91.197.54

+You Search Images Maps Play YouTube News Gmail Documents Calendar More -

Google

Search About 284,000 results (0.43 seconds)

Web [208.91.197.54 - SpyEye Tracker :: Monitor](https://spyeyetracker.abuse.ch/monitor.php?ipaddress=208.91.197.54)
<https://spyeyetracker.abuse.ch/monitor.php?ipaddress=208.91.197.54>
The abuse.ch SpyEye Tracker help you to track SpyEye Command&Control servers (C&Cs) and generating a IP-blocklist or domain-blocklist.

Images

Maps

Videos

News [208.91.197.54 | Ipaddress | Malwaregroup](http://www.malwaregroup.com/ipaddresses/.../208.91.197.54)
www.malwaregroup.com/ipaddresses/.../208.91.197.54 - United States
80+ items - IpAddress : **208.91.197.54**, Location : US, Autonomous System ...

More

Domain	Created	Modified
qwundayihu.com	2011-08-01	2012-04-13
bonanzatraining.com	2011-08-08	2012-04-16

Bangalore, Karnataka
Change location

The web
Pages from India
More search tools

ZeuS Tracker :: IP address 208.91.197.54
<https://zeustracker.abuse.ch/monitor.php?ipaddress=208.91.197.54>
The abuse.ch ZeuS Tracker help you to track ZeuS Command&Control servers and generating a IP-blocklist or domain-blocklist.

Malware for ip: 208.91.197.54 - Clean MX - realtime
[support.clean-mx.de/clean-mx/viruses.php?ip=208.91.197.54...](http://support.clean-mx.de/clean-mx/viruses.php?ip=208.91.197.54)
Safe Virus-Viewer and Analyser may take a minute to complete <http://alexgo.co/mysp.txt>? up, No previous evidence recorded Saved evidence (44310 Bytes) of ...

Step 3 – Who is Pid 1748?

“psscan” shows pid 1748 belongs to explorer.exe, also two process created during same time reported by security device (i.e june 8th 2012)

```
root@bt: ~/Volatility
File Edit View Terminal Help
root@bt:~/Volatility# python vol.py -f infected.dmp psscan
Volatile Systems Volatility Framework 2.0
Offset      Name                PID    PPID    PDB          Time created          Time exited
-----
0x0932b020  B6232F3A9F9.exe    1672   1748   0x0f9c02a0   2012-06-08 13:27:55   2012-06-08 13:27:56
0x09339020  wmiprvse.exe       584    880    0x0f9c0260   2012-02-26 12:07:19
0x0934c4a8  VMUpgradeHelper    428    700    0x0f9c0240   2012-02-26 12:07:19
0x09350740  vmtoolsd.exe       216    700    0x0f9c0220   2012-02-26 12:07:19
0x0935a360  explorer.exe       1748   1712   0x0f9c01c0   2012-02-26 12:07:17
0x093662b8  svchost.exe        964    700    0x0f9c0100   2012-02-26 12:07:11
0x094c6da0  svchost.exe        880    700    0x0f9c00e0   2012-02-26 12:07:11
0x095ffa58  ctfmon.exe         1900   1748   0x0f9c0200   2012-02-26 12:07:18
0x0964c020  erm.exe            1648   1888   0x0f9c0280   2012-06-08 13:27:53   2012-06-08 13:27:57
0x09656020  VMwareUser.exe    1888   1748   0x0f9c01e0   2012-02-26 12:07:18
0x09665630  winlogon.exe       656    376    0x0f9c0060   2012-02-26 12:07:11
0x097166a8  VMwareTray.exe    1880   1748   0x0f9c0180   2012-02-26 12:07:18
0x0971ea38  svchost.exe        1092   700    0x0f9c0140   2012-02-26 12:07:11
0x09732da0  csrss.exe          632    376    0x0f9c0040   2012-02-26 12:07:10
0x097aebf0  services.exe       700    656    0x0f9c0080   2012-02-26 12:07:11
0x09811020  lsass.exe          712    656    0x0f9c00a0   2012-02-26 12:07:11
0x09821020  smss.exe           376    4      0x0f9c0020   2012-02-26 12:07:10
0x0984c8e0  svchost.exe        1124   700    0x0f9c0160   2012-02-26 12:07:11
0x0984e170  svchost.exe        1048   700    0x0f9c0120   2012-02-26 12:07:11
0x098523b0  vmacthlp.exe       868    700    0x0f9c00c0   2012-02-26 12:07:11
0x0992b830  System              4      0      0x00319000
root@bt:~/Volatility#
```

Step 4 – Process handles of explorer.exe

Explorer.exe opens a handle to the B6232F3A9F9.exe, indicating explorer.exe created that process, which might be malicious...focusing on explorer.exe for now.

```
root@bt: ~/Volatility
File Edit View Terminal Help
root@bt:~/Volatility# python vol.py -f infected.dmp handles -p 1748 -t Process
Volatile Systems Volatility Framework 2.0
Offset(V)      Pid      Type      Details
0x8915a348     1748     Process   explorer.exe(1748)
0x8912b008     1748     Process   B6232F3A9F9.exe(1672)
0x8912b008     1748     Process   B6232F3A9F9.exe(1672)
root@bt:~/Volatility#
```


Step 5 – apihooks in explorer.exe

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

```
root@bt: ~/Volatility
File Edit View Terminal Help
root@bt:~/Volatility# python vol.py -f infected.dmp apihooks -p 1748
Volatile Systems Volatility Framework 2.0
Name                               Type      Target                                     Value
explorer.exe[1748]                  inline   user32.dll!TranslateMessage[0x7e418bf6] 0x7e418bf6 JMP 0xbb6bddc (UNKNOWN)
explorer.exe[1748]                  inline   crypt32.dll!PFXImportCertStore[0x77aeff8f] 0x77aeff8f JMP 0xbb70462 (UNKNOWN)
explorer.exe[1748]                  inline   wininet.dll!HttpSendRequestA[0x7806cd40] 0x7806cd40 JMP 0xbb82a3e (UNKNOWN)
explorer.exe[1748]                  inline   wininet.dll!HttpSendRequestW[0x78080825] 0x78080825 JMP 0xbb82b9c (UNKNOWN)
explorer.exe[1748]                  inline   wininet.dll!InternetCloseHandle[0x7805da59] 0x7805da59 JMP 0xbb7dc40 (UNKNOWN)
explorer.exe[1748]                  inline   wininet.dll!InternetWriteFile[0x78073645] 0x78073645 JMP 0xbb82cfa (UNKNOWN)
explorer.exe[1748]                  inline   advapi32.dll!CryptEncrypt[0x77dee340]    0x77dee340 JMP 0xbb7c597 (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!NtEnumerateValueKey[0x7c90d2d0] 0x7c90d2d0 JMP 0xbb6a7f0 (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!NtQueryDirectoryFile[0x7c90d750] 0x7c90d750 JMP 0xbb74885 (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!NtResumeThread[0x7c90db20]     0x7c90db20 JMP 0xbb861f8 (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!NtSetInformationFile[0x7c90dc40] 0x7c90dc40 JMP 0xbb6a53a (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!NtVdmControl[0x7c90df00]       0x7c90df00 JMP 0xbb7493b (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!ZwEnumerateValueKey[0x7c90d2d0] 0x7c90d2d0 JMP 0xbb6a7f0 (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!ZwQueryDirectoryFile[0x7c90d750] 0x7c90d750 JMP 0xbb74885 (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!ZwResumeThread[0x7c90db20]     0x7c90db20 JMP 0xbb861f8 (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!ZwSetInformationFile[0x7c90dc40] 0x7c90dc40 JMP 0xbb6a53a (UNKNOWN)
explorer.exe[1748]                  inline   ntdll.dll!ZwVdmControl[0x7c90df00]       0x7c90df00 JMP 0xbb7493b (UNKNOWN)
explorer.exe[1748]                  inline   ws2_32.dll!send[0x71ab4c27]             0x71ab4c27 JMP 0xbb7d3a6 (UNKNOWN)
Finished after 17.2333590984 seconds
root@bt:~/Volatility#
```



Step 6 – exploring the hooks

Disassembled hooked function (TranslateMessage), shows a short jump and then a long jump to malware location

```
File Edit View Terminal Help
root@bt:~/Volatility# python vol.py -f infected.dmp volshell
Volatile Systems Volatility Framework 2.0
Current context: process System, pid=4, ppid=0 DTB=0x319000
Welcome to volshell! Current memory image is:
file:///root/Volatility/infected.dmp
To get help, type 'hh()'
>>> hh()
ps() : Print a process listing.
cc(offset=None, pid=None, name=None) : Change current shell context.
dd(address, length=128, space=None) : Print dwords at address.
db(address, length=128, width=16, space=None) : Print bytes as canonical hexdump.
hh(cmd=None) : Get help on a command.
dt(objct, address=None) : Describe an object or show type info.
list_entry(head, objname, offset=-1, fieldname=None, forward=True) : Traverse a _LIST_ENTRY.
dis(address, length=128, space=None) : Disassemble code at a given address.

For help on a specific command, type 'hh(<command>)'
>>> cc(pid=1748)
Current context: process explorer.exe, pid=1748, ppid=1712 DTB=0xf9c01c0
>>> dis(0x7e418bf6, length=32)
0x7e418bf6 eb01 JMP 0x7e418bf9
0x7e418bf8 c3 RET
0x7e418bf9 e9de31758d JMP 0xbb6bddc
0x7e418bfe 086681 OR [ESI-0x7f], AH
0x7e418c01 7e08 JLE 0x7e418c0b
0x7e418c03 e500 IN EAX, 0x0
0x7e418c05 0f84667e0200 JZ 0x7e440a71
0x7e418c0b 6a00 PUSH 0x0
```

Step 7 – Embedded exe in explorer.exe

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

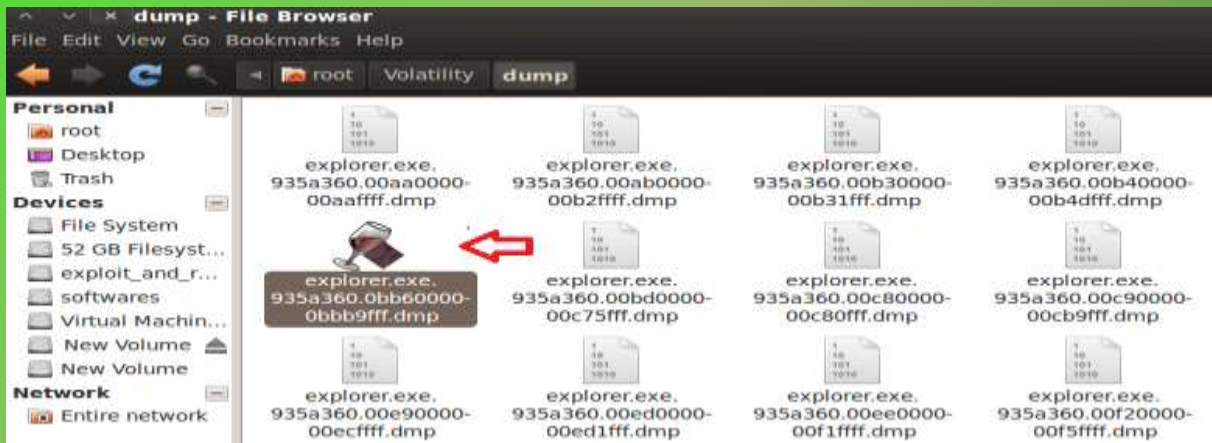
```
>>> db(0x0bb60000, length=256)
0bb60000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00  MZ.....
0bb60010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  .....@.....
0bb60020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb60030  00 00 00 00 00 00 00 00 00 00 00 00 00 e0 00 00  .....
0bb60040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb60050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb60060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb60070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb60080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb60090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb600a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb600b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb600c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb600d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0bb600e0  50 45 00 00 4c 01 02 00 92 60 ed 4d 00 00 00 00  PE..L....\..M....
0bb600f0  00 00 00 00 e0 00 02 01 0b 01 0a 00 00 a2 04 00  .....

>>> █
```

Step 8 – dumping the embedded exe

vaddump dumps the embedded exe from explorer.exe

```
root@bt: ~/Volatility
File Edit View Terminal Help
root@bt:~/Volatility# python vol.py -f infected.dmp vaddump -p 1748 -D dump/
Volatile Systems Volatility Framework 2.0
Pid: 1748
*****
root@bt:~/Volatility#
```



Step 9 – virustotal submission

Submission to virustotal, confirms the dumped executable as component of “spyeye”

Detection ratio: **8 / 39**

Analysis date: 2012-06-08 19:56:31 UTC (2 minutes ago)

[More details](#)

Antivirus	Result	Update
AhnLab-V3	Packed/Win32.Morphine	20120608
AntiVir	TR/Dropper.Gen	20120608
Antiy-AVL	-	20120608
Avast	Win32:Spyeye-XY [Trj]	20120608
BitDefender	-	20120608
ByteHero	-	20120606
CAT-QuickHeal	-	20120608
ClamAV	-	20120608
Commltouch	-	20120608
Comodo	-	20120608
Emsisoft	Trojan.Win32.Spyeye!IK	20120608
eSafe	-	20120607
F-Prot	-	20120608
F-Secure	-	20120608
Fortinet	-	20120608
GData	Win32:Spyeye-XY	20120608
Ikarus	Trojan.Win32.Spyeye	20120608

Step 10 – Can we get more info?

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

```
root@bt: ~/Volatility/test
File Edit View Terminal Help
root@bt:~/Volatility/test# strings explorer.exe.935a360.0bb60000-0bbb9fff.dmp > ascii strings.txt
```

```
Filesystem Browser
Connection: close
Connection:
Content-Length:
Content-Length:
Content-Encoding:
Content-Encoding: deflate
Content-Encoding: gzip
Transfer-Encoding:
chunked
Content-Length: %u|
HTTP/
User-Agent:
Accept-Encoding:
Keep-Alive:
Connection: keep-alive
Proxy-Connection: keep-alive
SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\RUN
*.
*.2x
cookies-nontor.xml
cookies.txt
sessionstore.js
sessionstore.bak
```

```
Filesystem Browser
f98u
_^[t
8&0l
C:\WINDOWS\system32\WININET.dll
C:\Recycle.Bin\A705B3960358085
C:\WINDOWS\system32\ntdll.dll
C:\WINDOWS\system32\ntdll.dll
C:\WINDOWS\system32\USER32.dll
C:\WINDOWS\system32\ntdll.dll
C:\WINDOWS\system32\CRYPT32.dll
C:\Recycle.Bin\B6232F3A9F9.exe
C:\Recycle.Bin\
B6232F3A9F9.exe
A705B3960358085
s1PSg1LF.exe
C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\
```

Step 11 – Printing the registry key

Malware creates registry key to survive the reboot

```
root@bt:~/Volatility# python vol.py -f infected.dmp printkey -K "SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\RUN"
Volatile Systems Volatility Framework 2.0
Legend: (S) = Stable (V) = Volatile

-----
Registry: \Device\HarddiskVolume1\Documents and Settings\LocalService\NTUSER.DAT
Key name: Run (S)
Last updated: 2011-10-31 15:07:20

Subkeys:

Values:
-----
Registry: \Device\HarddiskVolume1\WINDOWS\system32\config\default
Key name: Run (S)
Last updated: 2011-10-31 20:28:57

Subkeys:

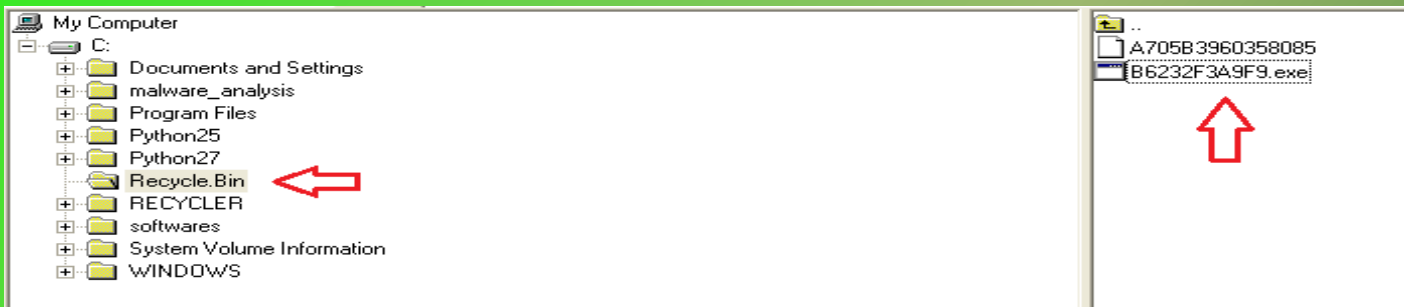
Values:
-----
Registry: \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT
Key name: Run (S)
Last updated: 2012-06-08 13:27:56

Subkeys:

Values:
REG_SZ      ctfmon.exe      : (S) C:\WINDOWS\system32\ctfmon.exe
REG_SZ      4Y3Y0C3A1F7XZHZWACQUD : (S) C:\Recycle.Bin\B6232F3A9F9.exe ←
```

Step 12 – Finding the malicious exe on infected machine

Finding malicious sample from infected host and virustotal submission confirms spyeye infection



Detection ratio: 37 / 42
Analysis date: 2012-06-08 20:38:03 UTC (0 minutes ago)

Antivirus	Result	Update
AhnLab-V3	Spyware.Win32.SpyEye	20120608
Avast	Win32-Spyeye.N.2112	20120608
Avira-AL	Trojan.Win32.SpyEye.gen	20120608
Avast	Win32-Spyeye-XY [Trj]	20120608
AVG	PWS/Generis.CAM	20120608
BitDefender	Trojan.Agent.ASGZ	20120608
BytePass	-	20120608
CAT-QuickHeal	TrojanSpy.SpyEyes.ikd	20120608
ClamAV	PWS.Win32.Packer.Ant-4	20120608

Kaspersky	Trojan-Spy.Win32.SpyEyes.ikd
McAfee	PWS-Spyeye.q
McAfee-GW-Edition	Heuristic.BehavesLike.Win32.ModifiedUPX.C
Microsoft	Trojan:Win32/EyeStye.N
NOD32	a variant of Win32/Spy.SpyEye.CA
Norman	W32/Suspicious_Gen2.QPFLH
nProtect	Trojan/W32.Agent.320512.CW
Panda	-
PCTools	Trojan.Spyeye

Reference

- [Complete Reference Guide for Reversing & Malware Analysis Training](#)

Thank You !



www.SecurityXploded.com