MEMORY FORENSIC

COMPUTER FORENSIC

Brief History of Memory Analysis

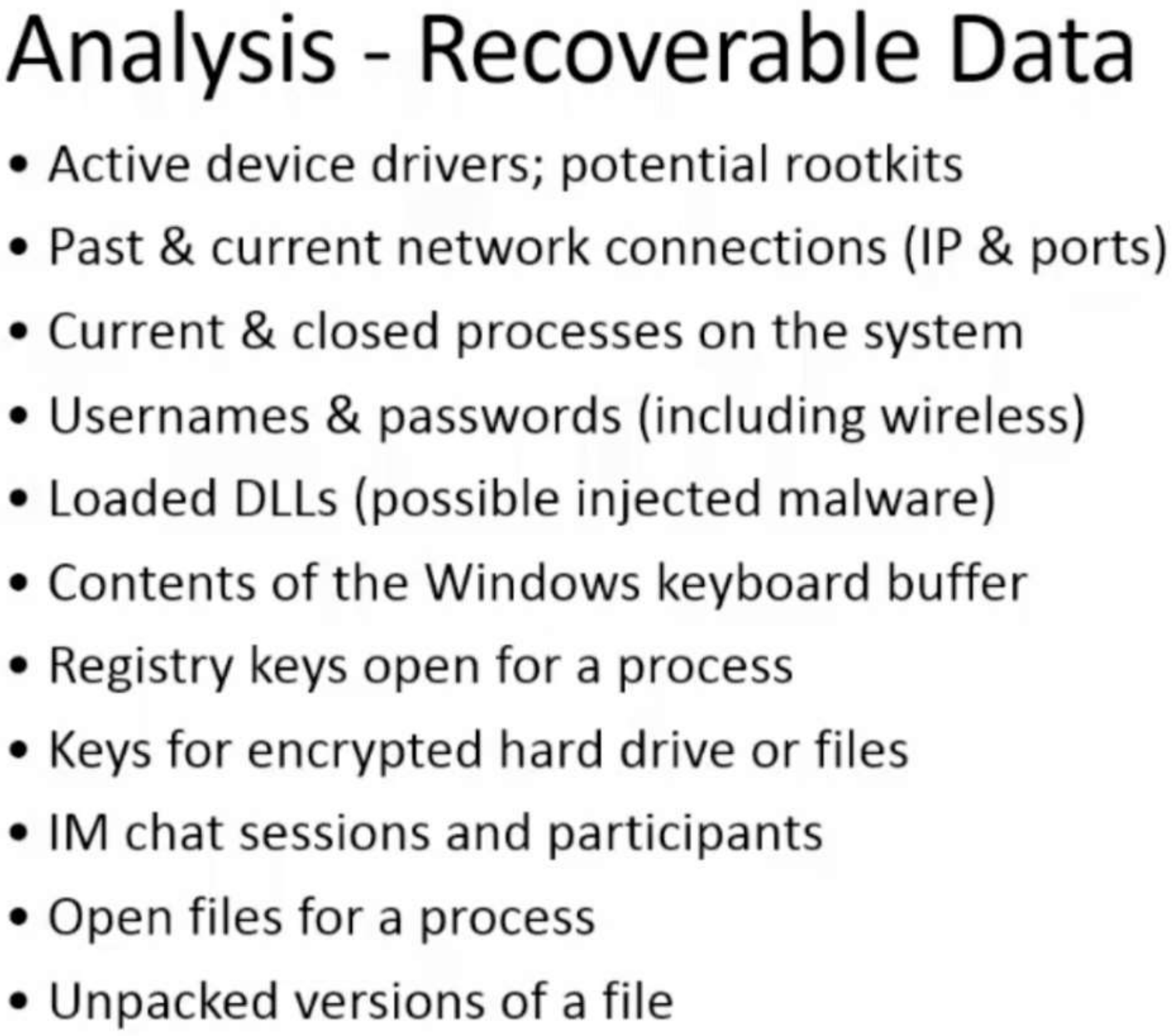
- Memory Analysis is a relatively new field.
  - The idea of memory captures began in the 1990's. The only capability was string searches with no context or understanding.
  - 2005 – DFRWS issued a *Memory Analysis Challenge*
    - Memparser developed by Chris Betz
    - First tool capable of identifying basic memory structures for forensic analysis. (Process lists, DLLs, PIDs)
  - 2007 – Aaron Walters and Nick Petroni release Volatility.
    - Open source tool provided unprecedented understanding of Windows Memory structures (XP only).
  - 2011 – Volatility 2.0 released
    - Offers expanded capabilities, additional plug-ins and works on a number of additional platforms.

What's the Big Deal with Memory Analysis?

- 2003 – 2006 – Rootkits became very popular and powerful.
  - Tipped the scales to the malware author's because their code was very good at hiding from the Windows API and difficult to identify via forensic dead drive analysis.
  - Some malware only existed in memory. When the system was shut down, all trace was gone.
- Memory analysis was a huge leap forward for forensic analysis.
  - Provided the ability to directly examine kernel-level processes regardless of their efforts to hide from Windows or the file system.

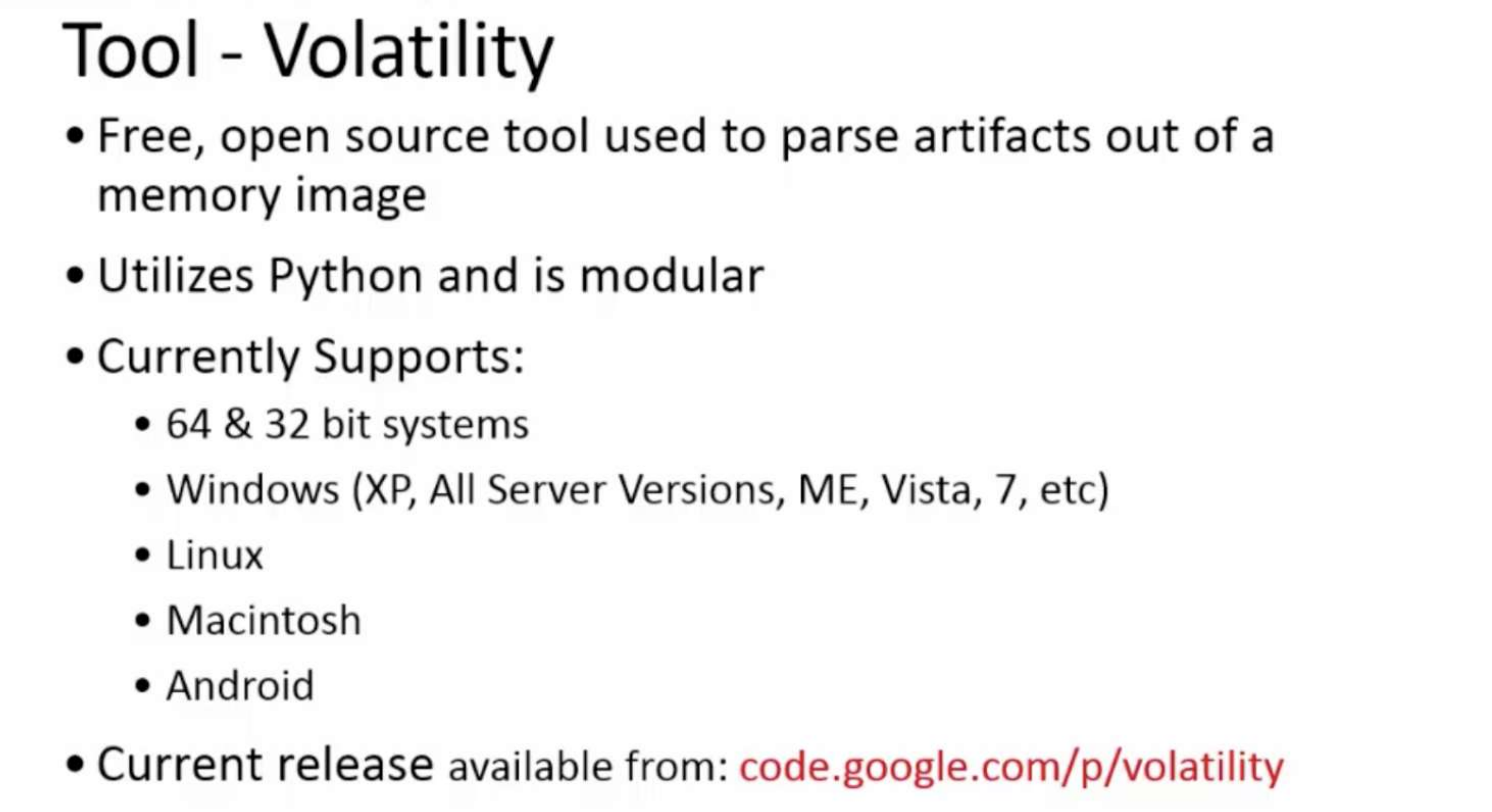
Analyzing Different Memory Formats

- Most of these analysis techniques can be applied to different types of memory files
  - Acquired RAM dump
  - VMware VMEM file
- Files that need to be converted to a raw image format before analysis
  - Hibernation file
  - BSOD – crash dump file
  - Conversion tools include Volatility and Moonsols



Analysis - Recoverable Data

- Active device drivers; potential rootkits
- Past & current network connections (IP & ports)
- Current & closed processes on the system
- Usernames & passwords (including wireless)
- Loaded DLLs (possible injected malware)
- Contents of the Windows keyboard buffer
- Registry keys open for a process
- Keys for encrypted hard drive or files
- IM chat sessions and participants
- Open files for a process
- Unpacked versions of a file



Tool - Volatility

- Free, open source tool used to parse artifacts out of a memory image
- Utilizes Python and is modular
- Currently Supports:
  - 64 & 32 bit systems
  - Windows (XP, All Server Versions, ME, Vista, 7, etc)
  - Linux
  - Macintosh
  - Android
- Current release available from: code.google.com/p/volatility

https://github.com/volatilityfoundation/volatility

Volatility Download Types (v 2.3.1)

- **Volatility-2.3.1standalone.exe** – No dependencies required, functions by itself from any media type.
  - Usage example:

```
volatility-2.3.1.standalone.exe pslist – f “C:\Memorydump\zeus.vmem”
```

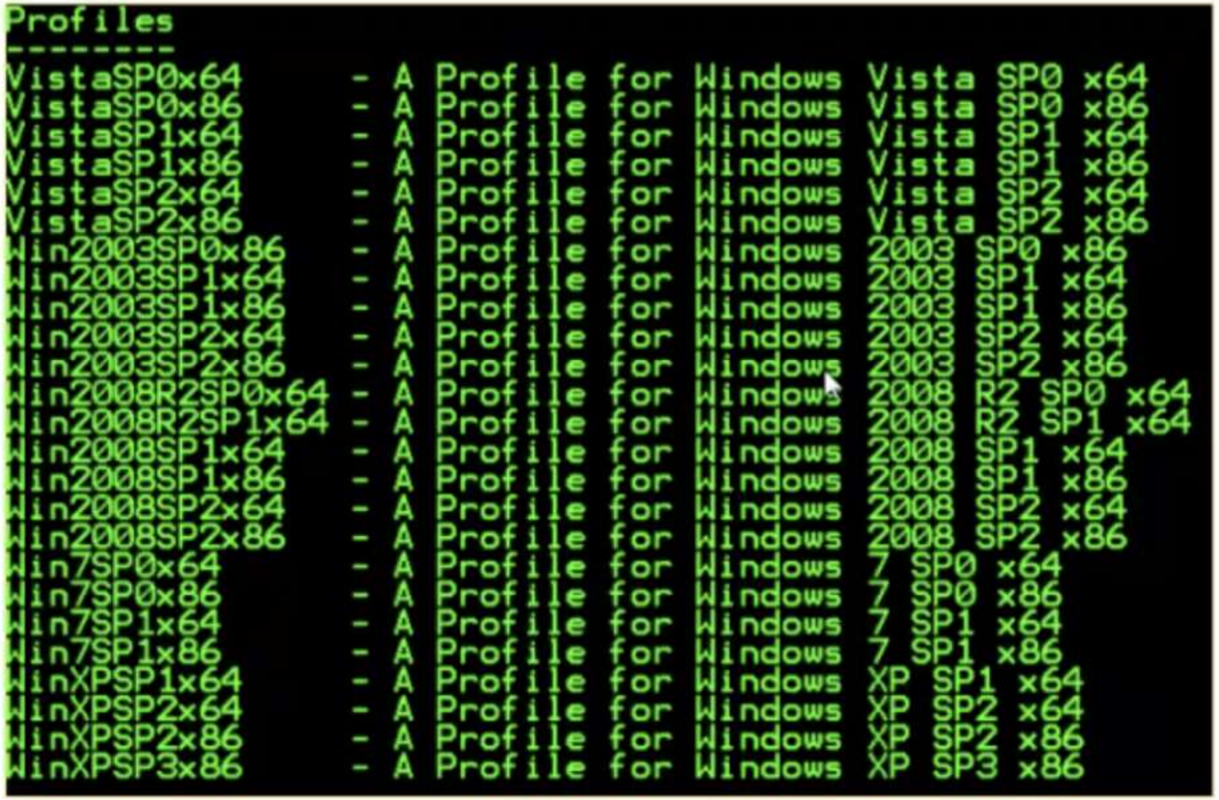
- **Volatility-2.3.1win32.exe** – Installs Volatility Python code. Used for editing and authoring new plug-ins. Requires pre-installed Python.
  - Usage example:

```
python vol.py pslist – f “C:\Memorydump\zeus.vmem”
```

Volatility Profile Commands

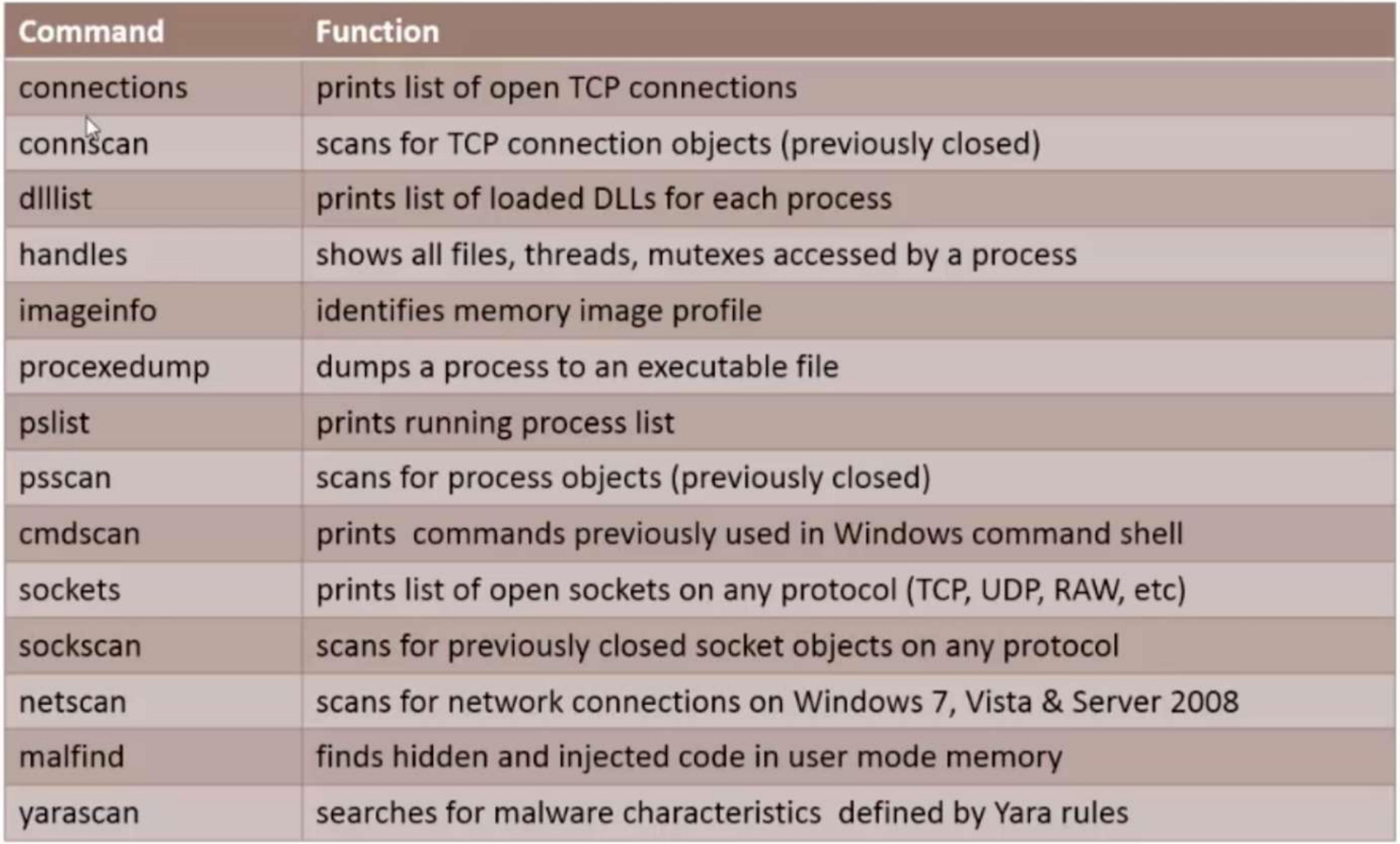
- Windows XP x86 (32 bit) is the default profile. All others require a specific flag.

```
volatility-2.3.standalone.exe pslist – f “C:\Memorydump\zeus.vmem” --profile=Win7SP1x64
```



```
Profiles
--------
VistaSP0x64     - A Profile for Windows Vista SP0 x64
VistaSP0x86     - A Profile for Windows Vista SP0 x86
VistaSP1x64     - A Profile for Windows Vista SP1 x64
VistaSP1x86     - A Profile for Windows Vista SP1 x86
VistaSP2x64     - A Profile for Windows Vista SP2 x64
VistaSP2x86     - A Profile for Windows Vista SP2 x86
Win2003SP0x86   - A Profile for Windows 2003 SP0 x86
Win2003SP1x64   - A Profile for Windows 2003 SP1 x64
Win2003SP1x86   - A Profile for Windows 2003 SP1 x86
Win2003SP2x64   - A Profile for Windows 2003 SP2 x64
Win2003SP2x86   - A Profile for Windows 2003 SP2 x86
Win2008R2SP0x64 - A Profile for Windows 2008 R2 SP0 x64
Win2008R2SP1x64 - A Profile for Windows 2008 R2 SP1 x64
Win2008SP1x64   - A Profile for Windows 2008 SP1 x64
Win2008SP1x86   - A Profile for Windows 2008 SP1 x86
Win2008SP2x64   - A Profile for Windows 2008 SP2 x64
Win2008SP2x86   - A Profile for Windows 2008 SP2 x86
Win7SP0x64      - A Profile for Windows 7 SP0 x64
Win7SP0x86      - A Profile for Windows 7 SP0 x86
Win7SP1x64      - A Profile for Windows 7 SP1 x64
Win7SP1x86      - A Profile for Windows 7 SP1 x86
WinXPSP1x64     - A Profile for Windows XP SP1 x64
WinXPSP2x64     - A Profile for Windows XP SP2 x64
WinXPSP2x86     - A Profile for Windows XP SP2 x86
WinXPSP3x86     - A Profile for Windows XP SP3 x86
```

VOLATILITY COMMAND



Command	Function
connections	prints list of open TCP connections
connscan	scans for TCP connection objects (previously closed)
dlllist	prints list of loaded DLLs for each process
handles	shows all files, threads, mutexes accessed by a process
imageinfo	identifies memory image profile
procexedump	dumps a process to an executable file
pslist	prints running process list
psscan	scans for process objects (previously closed)
cmdscan	prints commands previously used in Windows command shell
sockets	prints list of open sockets on any protocol (TCP, UDP, RAW, etc)
sockscan	scans for previously closed socket objects on any protocol
netscan	scans for network connections on Windows 7, Vista & Server 2008
malfind	finds hidden and injected code in user mode memory
yarascan	searches for malware characteristics defined by Yara rules

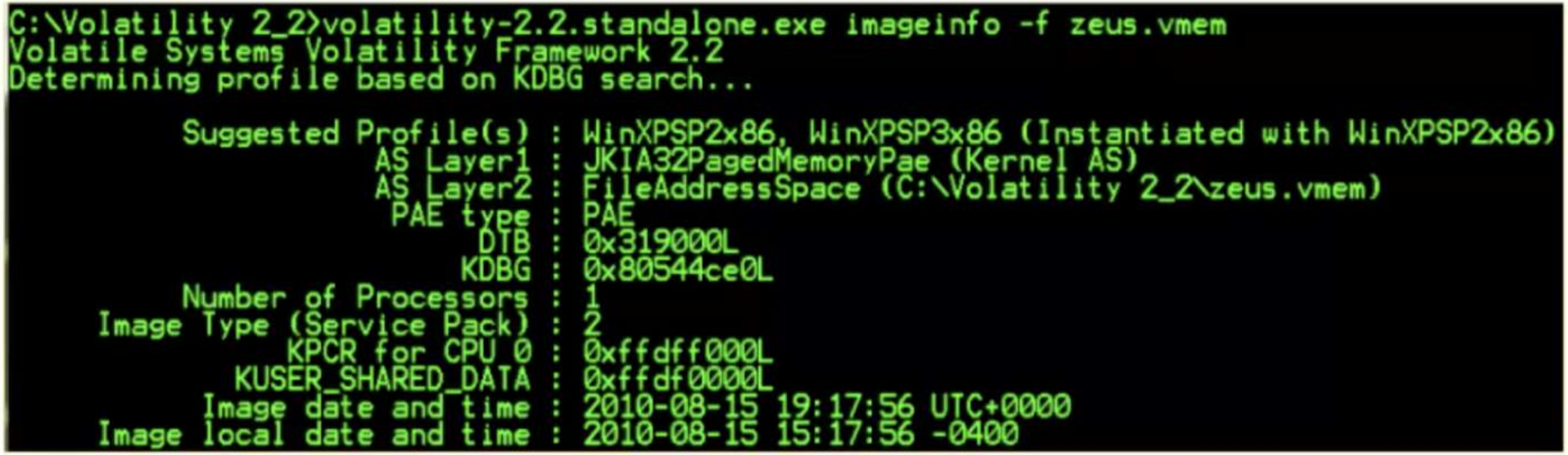
Case Study: Analyzing ZeuS with Volatility

- **ZeuS**
  - Crimeware kit sold in the cyber underground for $700 - $6,000, depending on options
  - Monitors online activity, waits for banking / monetary site logins and records all credentials
  - Exfils credentials back to the attacker and adds victim box to botnet
  - October 2010 – 5 ZeuS authors detained during Operation Trident Breach (an investigation into $70 million in losses)
  - November 2010 – ZeuS merges with SpyEye, a competing banking Trojan with similar capabilities
  - May 2011 – ZeuS Source Code released to general public

Spy Eye

Zeus: Identifying the Profile

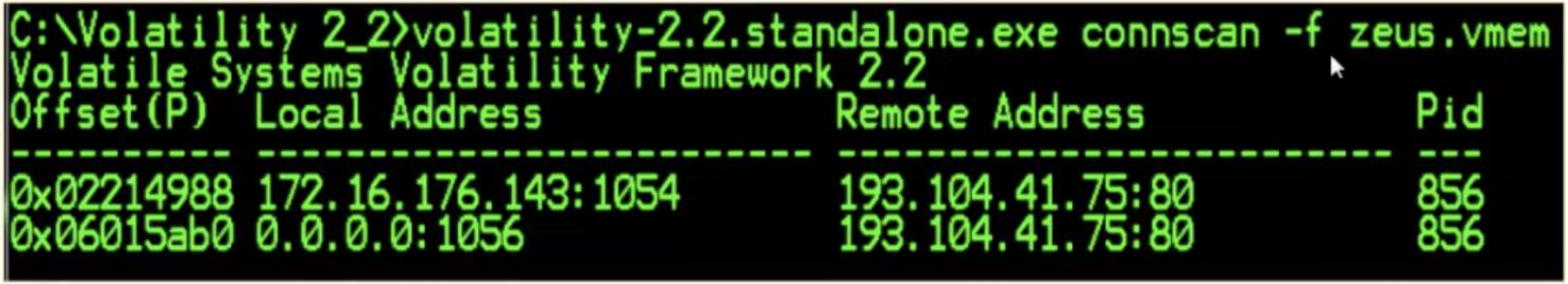
- Imageinfo displays key properties of the memory image:
  - Date and time of image
  - Operating System
  - Service Pack
  - Hardware Architecture (32 bit or 64 bit)
  - Shows required volatility profile



```
C:\Volatility 2_2>volatility-2.2.standalone.exe imageinfo -f zeus.vmem
Volatile Systems Volatility Framework 2.2
Determining profile based on KDBG search...

          Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
                     AS Layer1 : JKIA32PagedMemoryPae (Kernel AS)
                     AS Layer2 : FileAddressSpace (C:\Volatility 2_2\zeus.vmem)
                      PAE type : PAE
                           DTB : 0x319000L
                          KDBG : 0x80544ce0L
          Number of Processors : 1
     Image Type (Service Pack) : 2
                KPCR for CPU 0 : 0xffdff000L
             KUSER_SHARED_DATA : 0xffdf0000L
           Image date and time : 2010-08-15 19:17:56 UTC+0000
     Image local date and time : 2010-08-15 15:17:56 -0400
```

Zeus: Identifying Network Activity

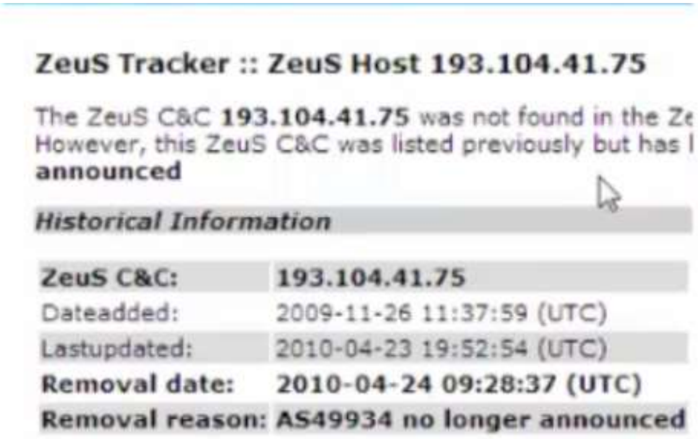
- Connections displays a list of all active TCP network connections
- Connscan searches for previously terminated TCP network connections
- Connscan shows us that the victim system was connected to 193.104.41.75 on port 80 from PID 856



```
C:\Volatility 2_2>volatility-2.2.standalone.exe connscan -f zeus.vmem
Volatile Systems Volatility Framework 2.2
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------- ------------------------- ---
0x02214988 172.16.176.143:1054       193.104.41.75:80          856
0x06015ab0 0.0.0.0:1056              193.104.41.75:80          856
```

Who is 193.104.41.75?

- **ZeusTracker:** Formerly a known Zeus Command and Control
- **Network Whois:** Registered in Ukraine
- **Maxmind Geolocation:** Server physically located in Moldova

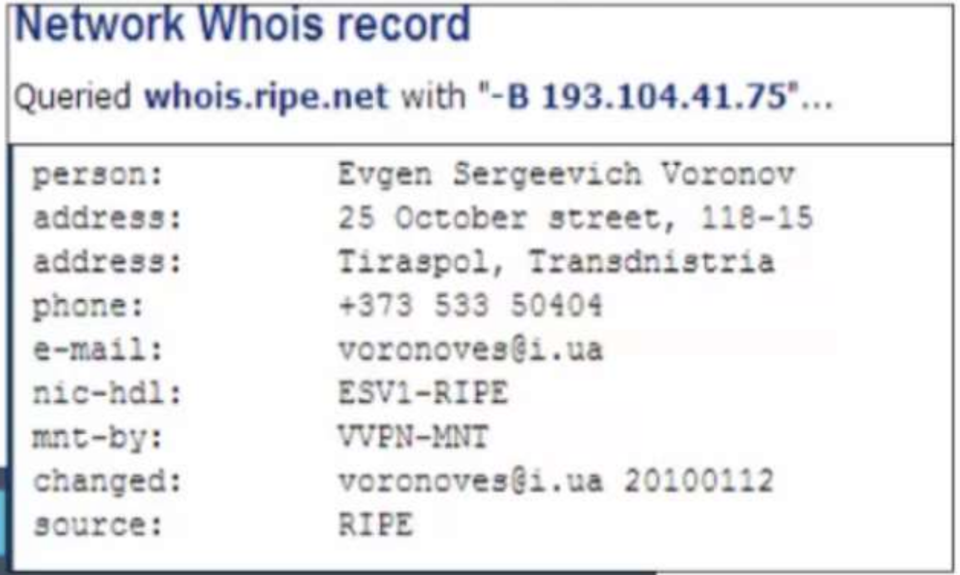


ZeuS Tracker :: ZeuS Host 193.104.41.75

The ZeuS C&C **193.104.41.75** was not found in the Ze
However, this ZeuS C&C was listed previously but has l
announced

Historical Information

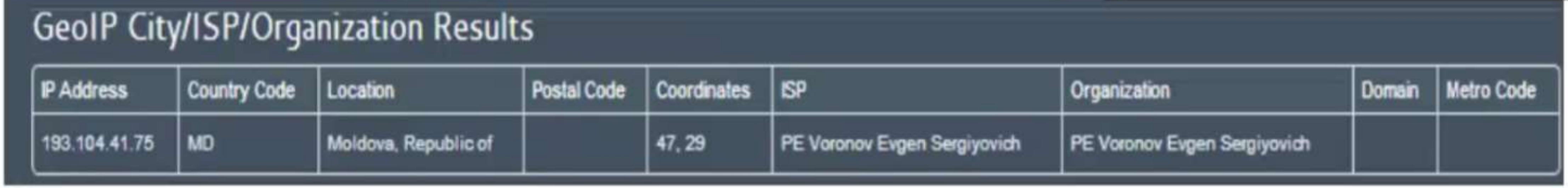
ZeuS C&C:	**193.104.41.75**
Dateadded:	2009-11-26 11:37:59 (UTC)
Lastupdated:	2010-04-23 19:52:54 (UTC)
Removal date:	**2010-04-24 09:28:37 (UTC)**
Removal reason:	**AS49934 no longer announced**



Network Whois record

Queried **whois.ripe.net** with "**-B 193.104.41.75**"...

```
person:         Evgen Sergeevich Voronov
address:        25 October street, 118-15
address:        Tiraspol, Transdnistria
phone:          +373 533 50404
e-mail:         voronoves@i.ua
nic-hdl:        ESV1-RIPE
mnt-by:         VVPN-MNT
changed:        voronoves@i.ua 20100112
source:         RIPE
```



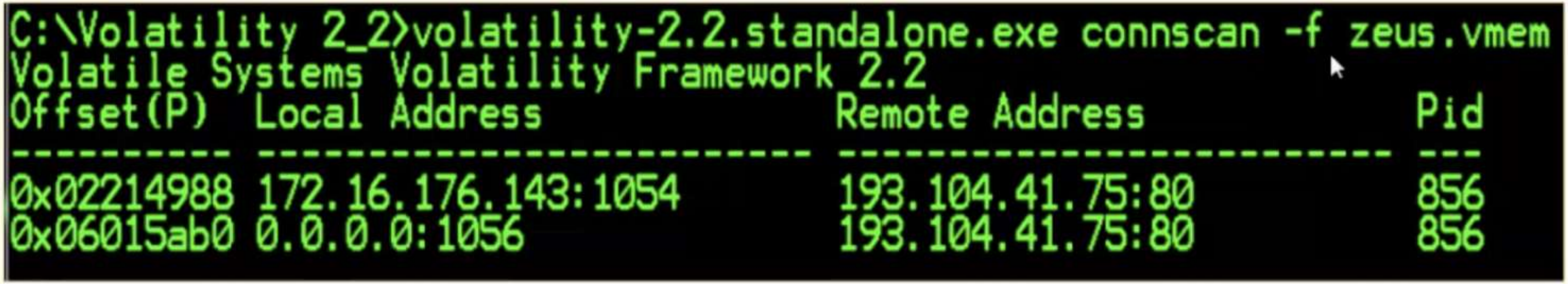
Try our GeoIP demo: 193.104.41.75 GO

GeoIP City/ISP/Organization Results

IP Address	Country Code	Location	Postal Code	Coordinates	ISP	Organization	Domain	Metro Code
193.104.41.75	MD	Moldova, Republic of		47, 29	PE Voronov Evgen Sergiyovich	PE Voronov Evgen Sergiyovich		

Zeus: Identifying Network Activity

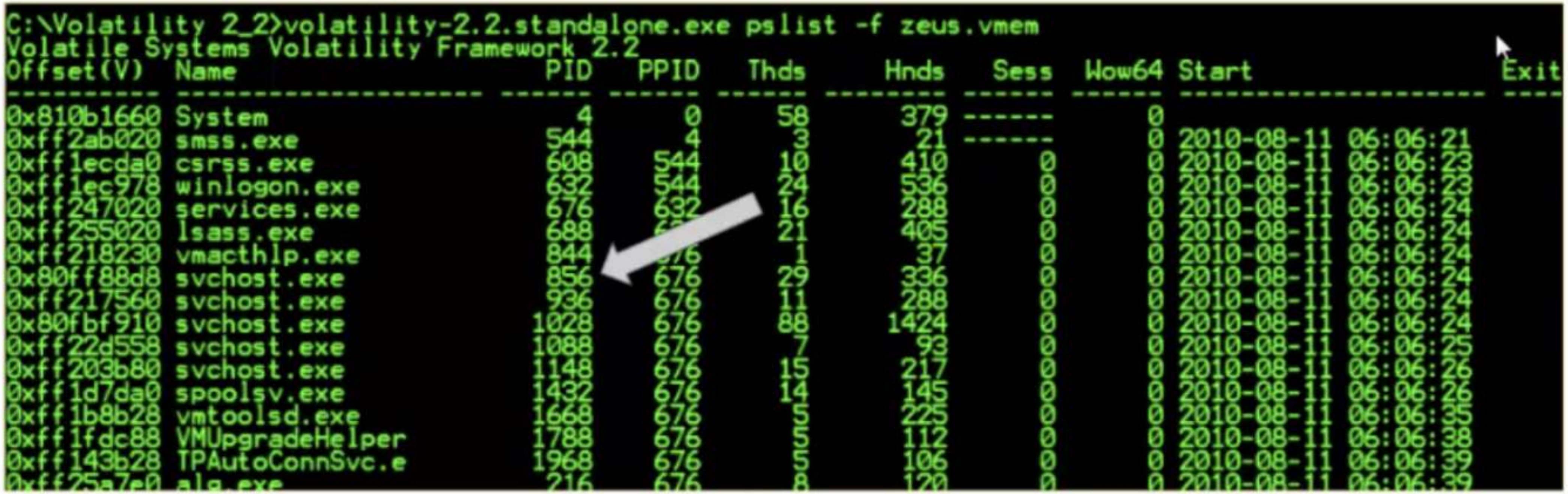
- Connections displays a list of all active TCP network connections
- Connscan searches for previously terminated TCP network connections
- Connscan shows us that the victim system was connected to 193.104.41.75 on port 80 from PID 856



```
C:\Volatility 2_2>volatility-2.2.standalone.exe connscan -f zeus.vmem
Volatile Systems Volatility Framework 2.2
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------- ------------------------- ---
0x02214988 172.16.176.143:1054       193.104.41.75:80          856
0x06015ab0 0.0.0.0:1056              193.104.41.75:80          856
```

Zeus: Identifying Processes

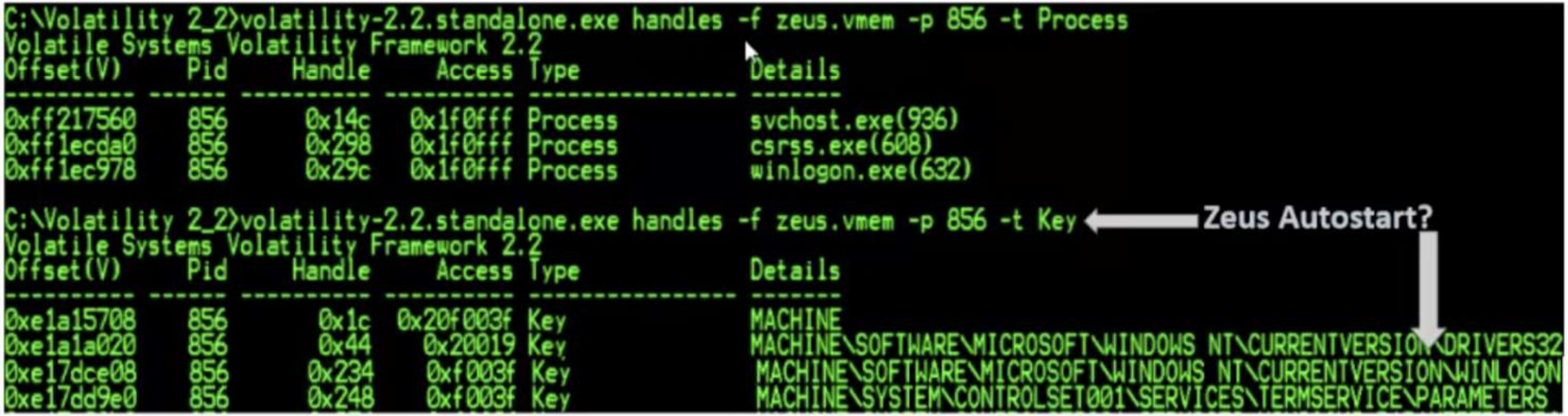
- pslist displays key information about running processes
- psscan shows processes that had previously exited
- pslist shows that PID 856 belongs to an instantiation of svchost.exe and gives its location in memory
- Its parent process was services.exe, which was likely legitimate as it started all other Windows services



```
C:\Volatility 2_2>volatility-2.2.standalone.exe pslist -f zeus.vmem
Volatile Systems Volatility Framework 2.2
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ -------------------- ----
0x810b1660 System                    4      0     58      379 ------      0
0xff2ab020 smss.exe                544      4      3       21 ------      0 2010-08-11 06:06:21
0xff1ecda0 csrss.exe               608    544     10      410      0      0 2010-08-11 06:06:23
0xff1ec978 winlogon.exe            632    544     24      536      0      0 2010-08-11 06:06:23
0xff247020 services.exe            676    632     16      288      0      0 2010-08-11 06:06:24
0xff255020 lsass.exe               688    632     21      405      0      0 2010-08-11 06:06:24
0xff218230 vmacthlp.exe            844    676      1       37      0      0 2010-08-11 06:06:24
0x80ff88d8 svchost.exe             856    676     29      336      0      0 2010-08-11 06:06:24
0xff217560 svchost.exe             936    676     11      288      0      0 2010-08-11 06:06:24
0x80fbf910 svchost.exe            1028    676     88     1424      0      0 2010-08-11 06:06:24
0xff22d558 svchost.exe            1088    676      7       93      0      0 2010-08-11 06:06:25
0xff203b80 svchost.exe            1148    676     15      217      0      0 2010-08-11 06:06:26
0xff1d7da0 spoolsv.exe            1432    676     14      145      0      0 2010-08-11 06:06:26
0xff1b8b28 vmtoolsd.exe           1668    676      5      225      0      0 2010-08-11 06:06:35
0xff1fdc88 VMUpgradeHelper        1788    676      5      112      0      0 2010-08-11 06:06:38
0xff143b28 TPAutoConnSvc.e        1968    676      5      106      0      0 2010-08-11 06:06:39
0xff25a7e0 alg.exe                 216    676      8      120      0      0 2010-08-11 06:06:39
```

Zeus: Identifying Process Activity

- handles: displays all files, registry keys, mutexes, named pipes, events, window stations, threads, and objects opened by a process
  - Note: used the –p flag to specify a process and –t to specify return data
- Svchost showed winlogon.exe and the winlogon registry key as open handles
  - Is this the autostart location?

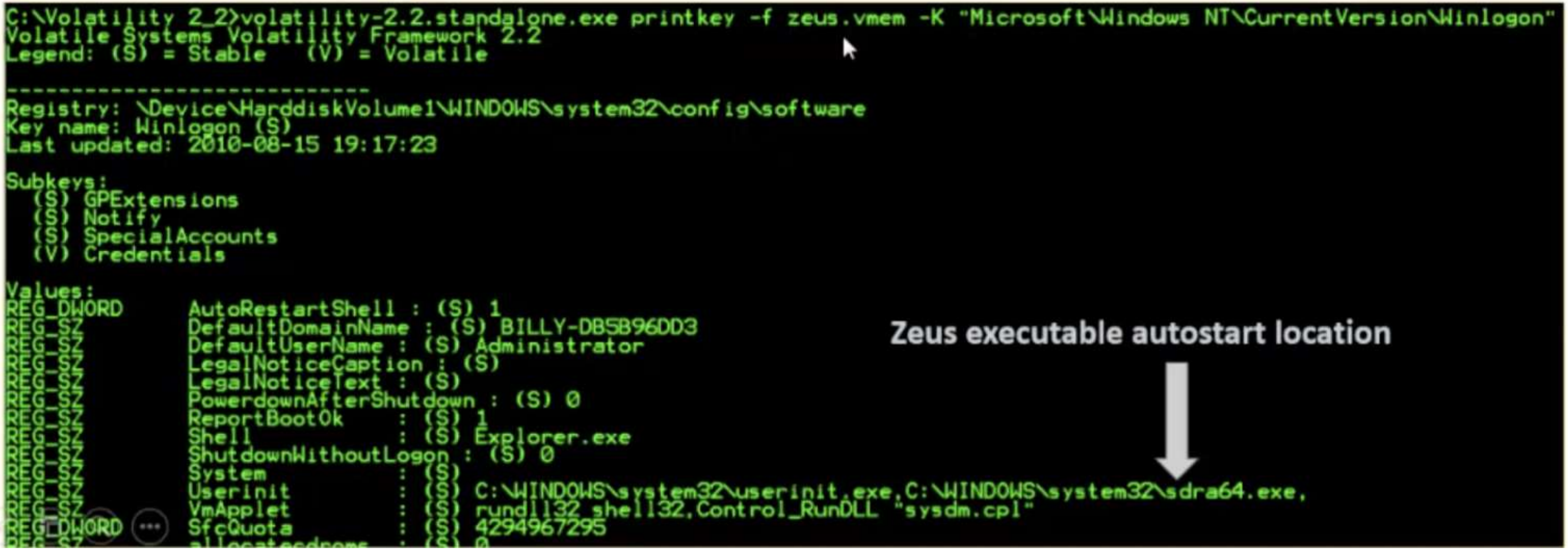


```
C:\Volatility 2_2>volatility-2.2.standalone.exe handles -f zeus.vmem -p 856 -t Process
Volatile Systems Volatility Framework 2.2
Offset(V)     Pid      Handle     Access Type             Details
---------- ------ ---------- ---------- ---------------- -------
0xff217560    856      0x14c   0x1f0fff Process          svchost.exe(936)
0xff1ecda0    856      0x298   0x1f0fff Process          csrss.exe(608)
0xff1ec978    856      0x29c   0x1f0fff Process          winlogon.exe(632)

C:\Volatility 2_2>volatility-2.2.standalone.exe handles -f zeus.vmem -p 856 -t Key
Volatile Systems Volatility Framework 2.2
Offset(V)     Pid      Handle     Access Type             Details
---------- ------ ---------- ---------- ---------------- -------
0xe1a15708    856       0x1c   0x20f003f Key              MACHINE
0xe1a1a020    856       0x44    0x20019 Key              MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION\DRIVERS32
0xe17dce08    856      0x234    0xf003f Key              MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION\WINLOGON
0xe17dd9e0    856      0x248    0xf003f Key              MACHINE\SYSTEM\CONTROLSET001\SERVICES\TERMSERVICE\PARAMETERS
```

Zeus Autostart?

Zeus: Registry Enumeration

- Printkey: displays contents of registry keys running in memory
  - Used the –K command to specify the Winlogon Registry key
- Winlogon key shows the userinit value includes sdra64.exe (known Zeus executable)



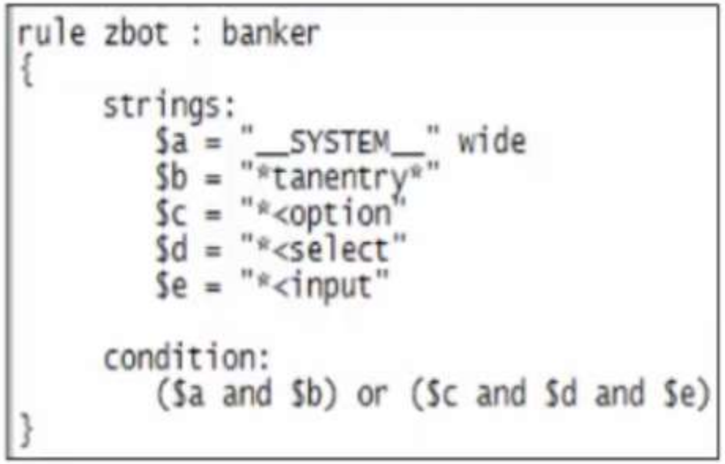
```
C:\Volatility 2_2>volatility-2.2.standalone.exe printkey -f zeus.vmem -K "Microsoft\Windows NT\CurrentVersion\Winlogon"
Volatile Systems Volatility Framework 2.2
Legend: (S) = Stable   (V) = Volatile

----------------------------
Registry: \Device\HarddiskVolume1\WINDOWS\system32\config\software
Key name: Winlogon (S)
Last updated: 2010-08-15 19:17:23

Subkeys:
  (S) GPExtensions
  (S) Notify
  (S) SpecialAccounts
  (V) Credentials

Values:
REG_DWORD     AutoRestartShell : (S) 1
REG_SZ        DefaultDomainName : (S) BILLY-DB5B96DD3
REG_SZ        DefaultUserName : (S) Administrator
REG_SZ        LegalNoticeCaption : (S)
REG_SZ        LegalNoticeText : (S)
REG_SZ        PowerdownAfterShutdown : (S) 0
REG_SZ        ReportBootOk     : (S) 1
REG_SZ        Shell            : (S) Explorer.exe
REG_SZ        ShutdownWithoutLogon : (S) 0
REG_SZ        System           : (S)
REG_SZ        Userinit         : (S) C:\WINDOWS\system32\userinit.exe,C:\WINDOWS\system32\sdra64.exe,
REG_SZ        VmApplet         : (S) rundll32 shell32,Control_RunDLL "sysdm.cpl"
REG_DWORD     SfcQuota         : (S) 4294967295
REG_SZ        allocatedcdroms  : (S) 0
```

Zeus executable autostart location

Yara introduction

- Yara enables malware researchers to identify & classify malware families
    - Estimated 100,000 new malware pieces every day – most are built on the same base code
    - Yara signatures identify base code characteristics and are used to search unknown processes for known malicious properties
    - Yara signature files can use:
        - Text strings (ASCII and Unicode)
        - Hexadecimal strings
        - Regular Expressions
        - Wildcards

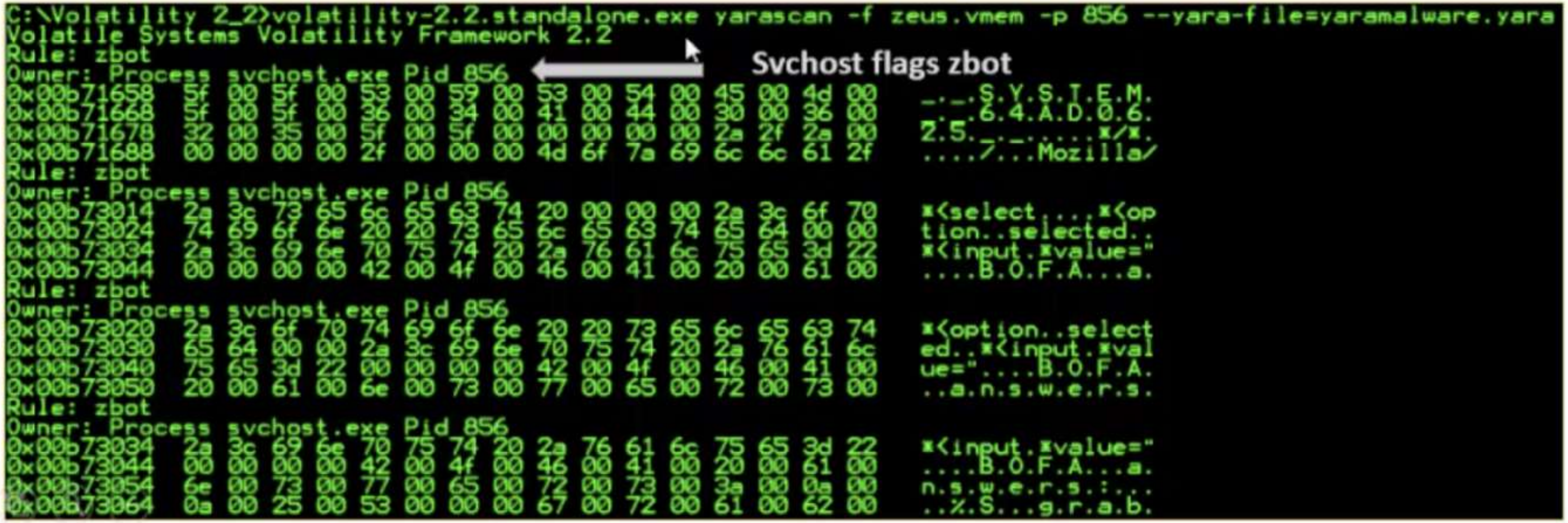
Sample Zeus Yara Rule



```
rule zbot : banker
{
      strings:
         $a = "__SYSTEM__" wide
         $b = "*tanentry*"
         $c = "*<option"
         $d = "*<select"
         $e = "*<input"

      condition:
         ($a and $b) or ($c and $d and $e)
}
```

Zeus: Capability Classification with Yara

- yarascan will scan memory for known malware characteristics
  - Used –p flag to specify svchost
  - Used –yara-file=<pathtofile> flag to specify Yara rules file
- Svchost flags on Yara Zbot rules in multiple locations

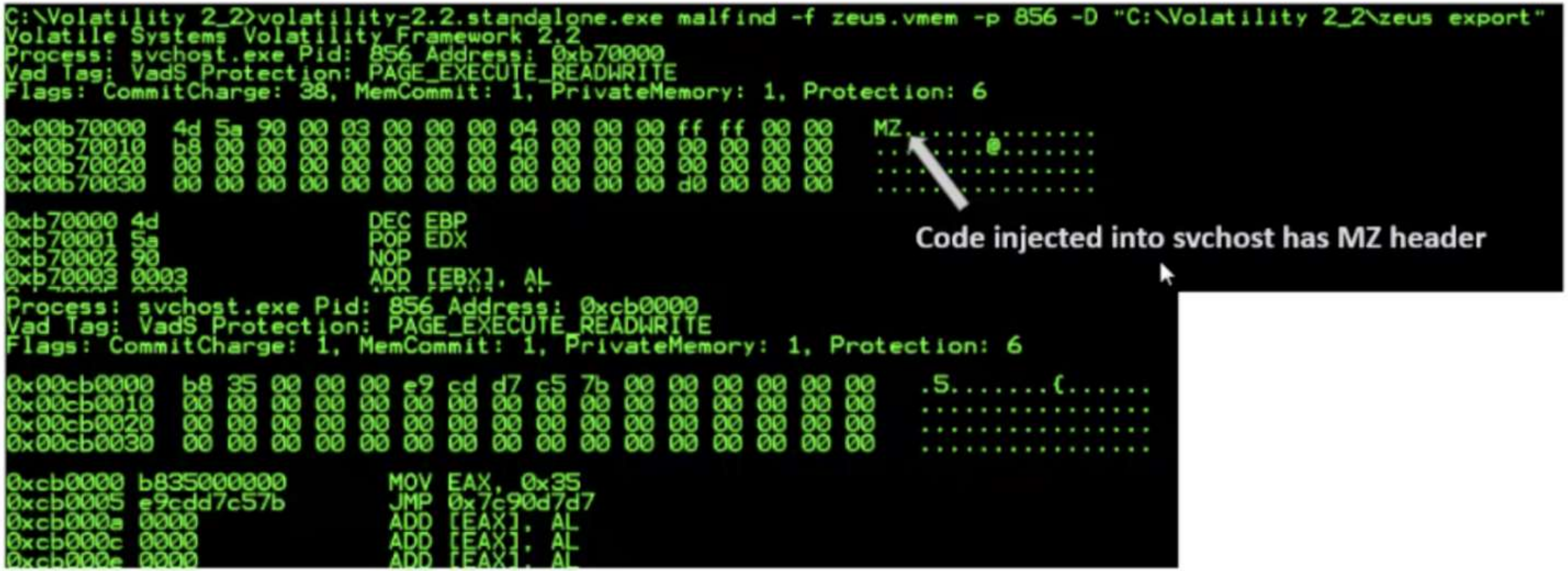


```
C:\Volatility 2_2>volatility-2.2.standalone.exe yarascan -f zeus.vmem -p 856 --yara-file=yaramalware.yara
Volatile Systems Volatility Framework 2.2
Rule: zbot
Owner: Process svchost.exe Pid 856
0x00b71658  5f 00 5f 00 53 00 59 00 53 00 54 00 45 00 4d 00   _._.S.Y.S.T.E.M.
0x00b71668  5f 00 5f 00 36 00 34 00 41 00 44 00 30 00 36 00   _._.6.4.A.D.0.6.
0x00b71678  32 00 35 00 5f 00 5f 00 00 00 00 00 2a 2f 2a 00   2.5._._.....*/*.
0x00b71688  00 00 00 00 2f 00 00 00 4d 6f 7a 69 6c 6c 61 2f   ..../...Mozilla/
Rule: zbot
Owner: Process svchost.exe Pid 856
0x00b73014  2a 3c 73 65 6c 65 63 74 20 00 00 00 2a 3c 6f 70   *<select....*<op
0x00b73024  74 69 6f 6e 20 20 73 65 6c 65 63 74 65 64 00 00   tion..selected..
0x00b73034  2a 3c 69 6e 70 75 74 20 2a 76 61 6c 75 65 3d 22   *<input.*value="
0x00b73044  00 00 00 00 42 00 4f 00 46 00 41 00 20 00 61 00   ....B.O.F.A...a.
Rule: zbot
Owner: Process svchost.exe Pid 856
0x00b73020  2a 3c 6f 70 74 69 6f 6e 20 20 73 65 6c 65 63 74   *<option..select
0x00b73030  65 64 00 00 2a 3c 69 6e 70 75 74 20 2a 76 61 6c   ed..*<input.*val
0x00b73040  75 65 3d 22 00 00 00 00 42 00 4f 00 46 00 41 00   ue="....B.O.F.A.
0x00b73050  20 00 61 00 6e 00 73 00 77 00 65 00 72 00 73 00   ..a.n.s.w.e.r.s.
Rule: zbot
Owner: Process svchost.exe Pid 856
0x00b73034  2a 3c 69 6e 70 75 74 20 2a 76 61 6c 75 65 3d 22   *<input.*value="
0x00b73044  00 00 00 00 42 00 4f 00 46 00 41 00 20 00 61 00   ....B.O.F.A...a.
0x00b73054  6e 00 73 00 77 00 65 00 72 00 73 00 3a 00 0a 00   n.s.w.e.r.s.:...
0x00b73064  0a 00 25 00 53 00 00 00 67 00 72 00 61 00 62 00   ..%.S...g.r.a.b.
```

Svchost flags zbot

Zeus: Identifying Injected Code

- malfind searches for hidden or injected code in user mode memory base
  - Used –p flag to specify process and –D to dump the injected code to the hard drive
- Malfind located two injected code locations in svchost
  - One has an MZ (executable) header – Highly suspicious
  - -D extracts the injected executable to the hard drive for further analysis



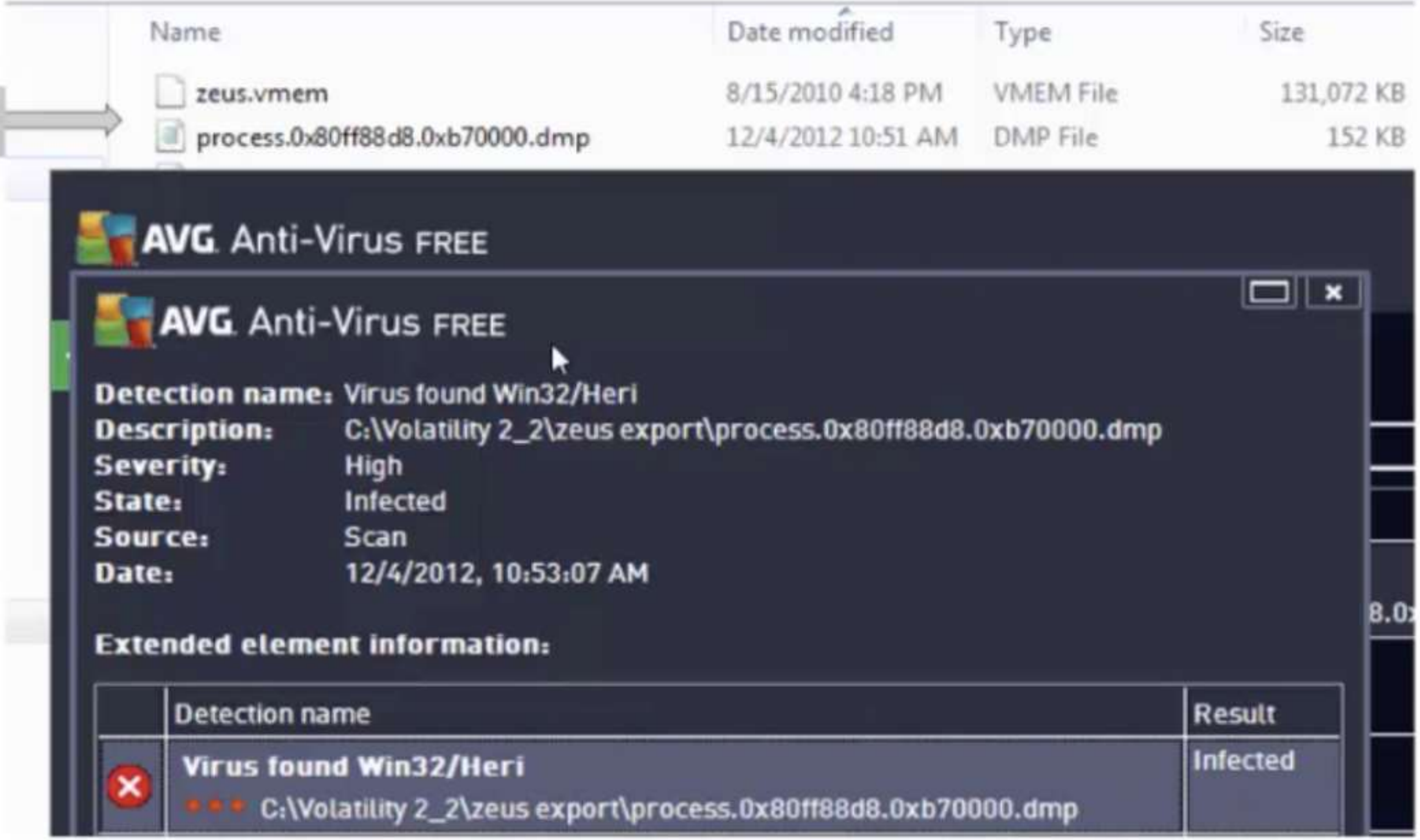
```
C:\Volatility 2_2>volatility-2.2.standalone.exe malfind -f zeus.vmem -p 856 -D "C:\Volatility 2_2\zeus export"
Volatile Systems Volatility Framework 2.2
Process: svchost.exe Pid: 856 Address: 0xb70000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 38, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x00b70000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x00b70010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x00b70020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00b70030  00 00 00 00 00 00 00 00 00 00 00 00 d0 00 00 00   ................

0xb70000 4d               DEC EBP
0xb70001 5a               POP EDX
0xb70002 90               NOP
0xb70003 0003             ADD [EBX], AL
Process: svchost.exe Pid: 856 Address: 0xcb0000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 1, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x00cb0000  b8 35 00 00 00 e9 cd d7 c5 7b 00 00 00 00 00 00   .5.......{......
0x00cb0010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00cb0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00cb0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................

0xcb0000 b835000000       MOV EAX, 0x35
0xcb0005 e9cdd7c57b       JMP 0x7c90d7d7
0xcb000a 0000             ADD [EAX], AL
0xcb000c 0000             ADD [EAX], AL
0xcb000e 0000             ADD [EAX], AL
```

Code injected into svchost has MZ header

Antivirus Scan of Extracted Code

- Used AVG antivirus to scan the code extracted from svchost
  - Returned as Win32/Heri infected file – this is how AVG classified Zeus

Export of the injected code



Name	Date modified	Type	Size
zeus.vmem	8/15/2010 4:18 PM	VMEM File	131,072 KB
process.0x80ff88d8.0xb70000.dmp	12/4/2012 10:51 AM	DMP File	152 KB

Detection name: Virus found Win32/Heri
Description: C:\Volatility 2_2\zeus export\process.0x80ff88d8.0xb70000.dmp
Severity: High
State: Infected
Source: Scan
Date: 12/4/2012, 10:53:07 AM

Extended element information:

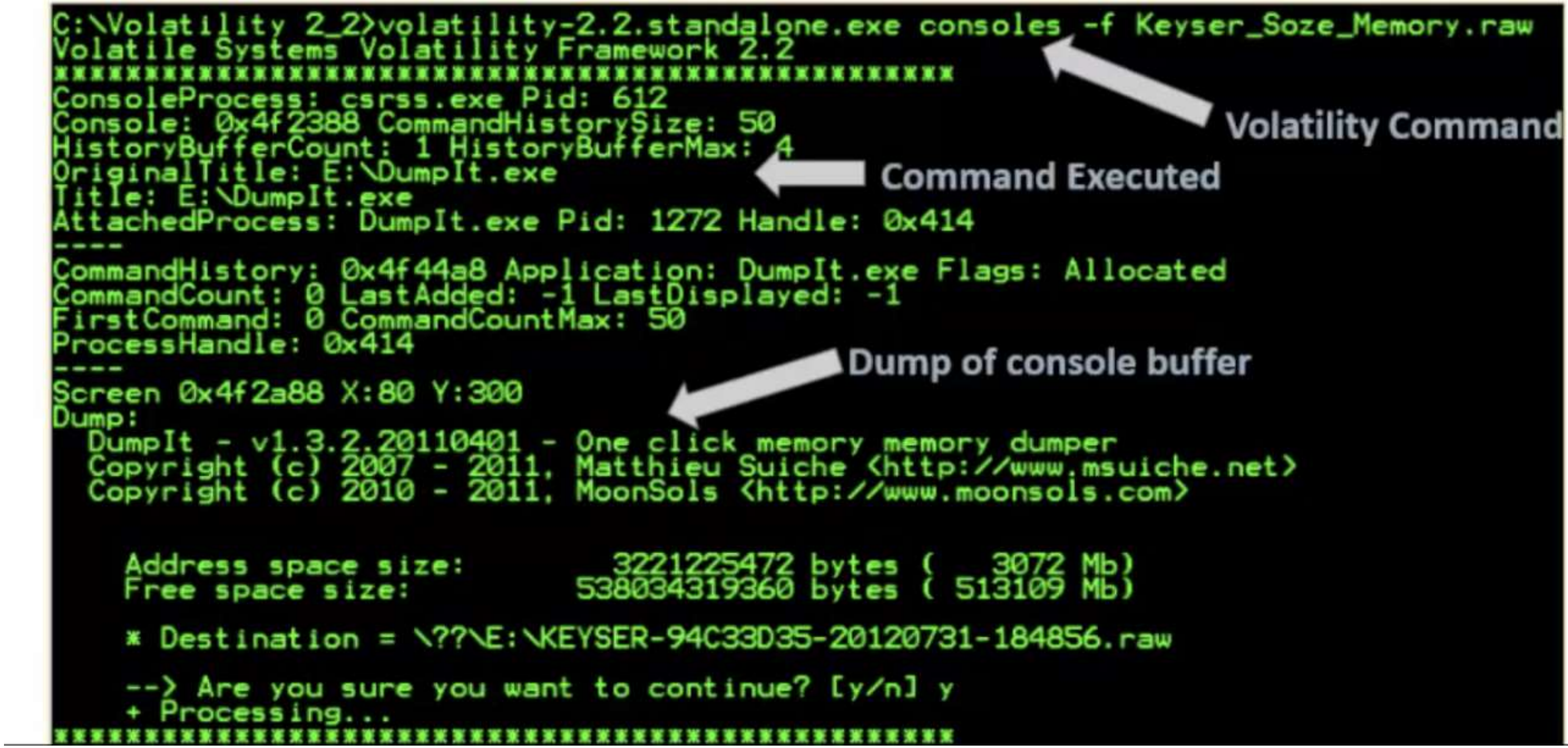
	Detection name	Result
☒	**Virus found Win32/Heri** C:\Volatility 2_2\zeus export\process.0x80ff88d8.0xb70000.dmp	Infected

Review of Memory Analysis

- connscan showed PID 856 had a TCP connection with a known Zeus Command and Control site (193.104.41.75)
- pslist showed that PID 856 was svchost.exe
- handles showed svchost used the winlogon registry key, indicating a potential autostart location
- printkey isolated the winlogon key and showed that userinit was set to autorun sdra64.exe (known Zeus executable name)
- yarascan indicated that svchost may contain Zeus, and may have VM & Debugger identification capabilities
- malfind identified the injected code in svchost.exe and extracted it to the hard drive
- AVG antivirus confirmed that we successfully extracted the malicious Zeus code from svchost.exe in memory

cmdscan & consoles

- Cmdscan provides a history of commands entered into the command shell. This may show specific attacker commands.
- Consoles provides the same but includes the screen buffer. It will show what the attacker actually saw.



```
C:\Volatility 2_2>volatility-2.2.standalone.exe consoles -f Keyser_Soze_Memory.raw
Volatile Systems Volatility Framework 2.2
***********************************************
ConsoleProcess: csrss.exe Pid: 612
Console: 0x4f2388 CommandHistorySize: 50
HistoryBufferCount: 1 HistoryBufferMax: 4
OriginalTitle: E:\DumpIt.exe
Title: E:\DumpIt.exe
AttachedProcess: DumpIt.exe Pid: 1272 Handle: 0x414
----
CommandHistory: 0x4f44a8 Application: DumpIt.exe Flags: Allocated
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x414
----
Screen 0x4f2a88 X:80 Y:300
Dump:
  DumpIt - v1.3.2.20110401 - One click memory memory dumper
  Copyright (c) 2007 - 2011, Matthieu Suiche <http://www.msuiche.net>
  Copyright (c) 2010 - 2011, MoonSols <http://www.moonsols.com>


    Address space size:        3221225472 bytes (   3072 Mb)
    Free space size:         538034319360 bytes ( 513109 Mb)

    * Destination = \??\E:\KEYSER-94C33D35-20120731-184856.raw

    --> Are you sure you want to continue? [y/n] y
    + Processing...
***********************************************
```

Other Useful VolatilityCommands

- dlllist – lists all dynamic link library (dll) files called by specific processes; great for identifying dll injection attacks
- dlldump – extract dll files from a process's memory space
- procexedump – extract a process's disk-mode executable from memory
- procmemdump – extract a process's memory mode executable (including slack space)

Even More Useful Volatility Commands

- imagecopy – convert crashdump, hibernation file, or live firewire session to a raw memory dump capable of analysis.
- userassist – lists contents of the NTUSER.DAT UserAssist registry key, showing programs executed by specific users.
- hashdump – extract domain password hashes from SYSTEM and SAM registry keys.