Financial Malware Analysis- Secrets of ZeuS

Rushita Dave[1],Anisetti Anjaneyulu[2]

[1]Forensics Analyst eSF Labs Ltd.

Abstract- *Malware becomes one of the internet's major security threats these days. Malware is the most widespread threat towards IT now. Because of the huge quantity of new malware samples, researchers rely on dynamic malware analysis. Malware analysis is an important part of understanding the objectives of the malware and how to defend against this threat. To be able to defend against the threat imposed by malware we need to understand both how and why the malware exists.*

I. INTRODUCTION

Malware is a general term for a piece of software inserted into an information system to cause harm to the system. Malware can gain remote access to system, record and send data to a third party without the user's permission or knowledge. Malware are commonly described as viruses, worms, trojan horses, backdoors, keystroke loggers, rootkits or spyware. Software may contain vulnerabilities, in its structure caused by imperfect coding. Once this type of vulnerabilities is revealed, malware can be developed to exploit them for malicious purposes.

Zeus refers to an entire family of trojans and their respective botnets. Zeus installs a rootkit component to stay hidden on infected systems. To bypass firewalls and to remain active on infected systems, Zeus inserts itself in the address space of other running processes typically Windows Explorer.

Features of Zeus: Capture credentials over HTTP and HTTPS, Steals HTTP and flash cookies, encrypted configuration file, modifies local host files, unique bot identification string.

II. MEMORY FORENSICS FOR MALWARE ANALYSIS

RAM contains critical information about the runtime state of the system while the system is active. By capturing an entire RAM and analyzing it on a separate computer, it is possible to reconstruct the state of the original system, including what applications were running, which files those applications were accessing, which network connections were active, and many other artifacts. For these reasons, memory forensics is extremely important to incident response. Memory forensics helps with unpacking, rootkit detection, and reverse engineering.

A. Tools to capture memory

ManTech Memory DD

Mandiant Memoryze

FTK Imager

DumpIt

Virtual Machine Files (vmem):

Virtual machines provide a useful environment for dynamic analysis of malware. Acquire RAM from guest machines by just suspending or pausing the VM, and guest's RAM will be written to a file on the host's disk. Then grab the ".vmem" file for analysis. The default location of the vmem file for VMware is %My Documents%\My Virtual Machines\<VM Name>*.vmem

B. Tool to analyze the memory

Volatility

The Volatility Framework is an open collection of tools for the extraction of digital artifacts from volatile memory samples. Whether your memory dump is in raw format, a Microsoft crash dump, hibernation file, or virtual machine snapshot, Volatility is able to work with it. It is a very good tool used for malware analysis as malware runs in the memory.

III. MALWARE ANALYSIS USING VOLATILITY

A. Image information (imageinfo)

python vol.py -f Zeus_Final.vmem imageinfo

Set a profile to tell Volatility what operating system the dump came from. The "imageinfo" output suggests profile that should be passed as the parameter to --profile=PROFILE

B. Scan for the ZeuS Malware (zbotscan)

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem zbotscan

This will show the version of the ZeuS - ZEUS 2.1.0.1, malware infected process with pid - 1512, registry key - HKCU\SOFTWARE\Microsoft\Irpao and executable - ixmo.exe

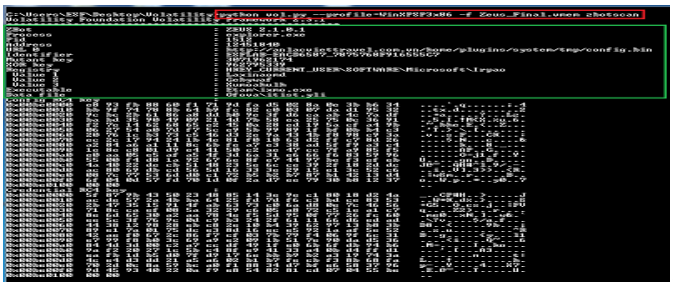

Fig 1. zbot command to check for the infected process

C. Checking the process list (psscan)

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem psscan

This is used for analysis of malware and rootkit activities. It scans for inactive, hidden and unlinked processes by a malware or rootkit.

Check for the process that was shown in the zbotscan command.

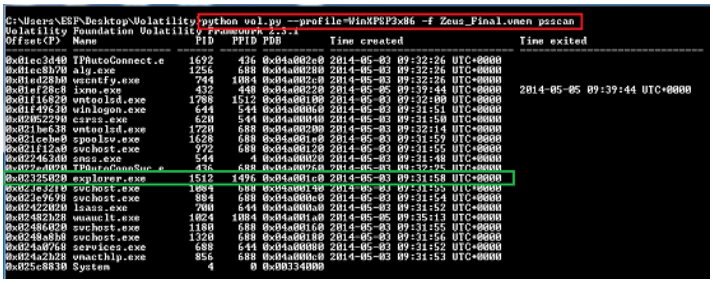

Fig.2 List of the process running in the system at the time the dump was taken

Here it shows the detail about the "explorer.exe" which was found before.

D. Handles of the Process (handles)

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem handles -p 1512 -t Process

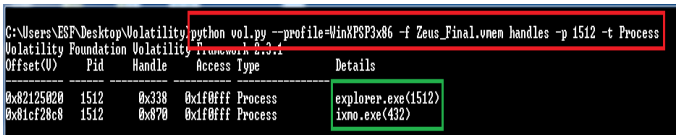

Fig.3 Process handles of the "explorer.exe"

Here it shows that explorer.exe is communicating with 2 processes. One is itself explorer.exe and other is ixmo.exe which is executable shown before.

E. Check for the hooks (apihooks)

It is used to find API hooks in user mode or kernel mode. This finds IAT, EAT, Inline style hooks. . For Inline hooks, it detects CALLs and JMPs to direct and indirect locations.

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem handles

It show the hook type - Inline hook, Process - explorer.exe, victim module - ntdll.dll, kernel32.dll, user32.dll.

F. Check for the modules (ldrmodules)

It is also possible for malware to hide a DLL by simply overwriting the path. Tools that only look for unlinked entries may miss the fact that malware overwrite C:\malicious.dll to show C:\windows\system32\kernel32.dll.

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem ldrmodules

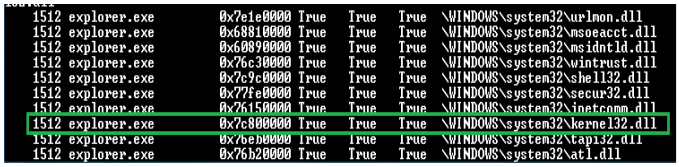

Fig.4 modules of explorer.exe

It shows the base address of process that can be further use for analyzing the process for finding the injected code in it.

G. Volatility Shell (volshell)

Volshell is the teminal of the volatiltiy itself.

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem volshell

cc(pid=1512)

db(offset)

offset - found from the result of the ldrmodules.

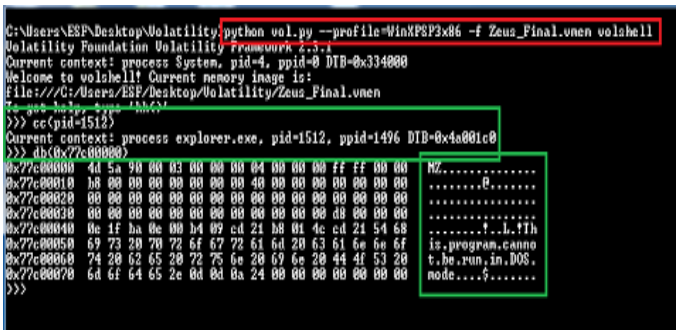

Fig.5 volatilitiy shell

H. Finding malicious code (malfind)

The malfind is used to find hidden or injected code/DLLs in user mode memory. It is also used to locate sequence of bytes, regular expressions, ANSI strings, or Unicode strings in user mode or kernel memory. The purpose of malfind is to locate DLLs that standard methods/tools do not see.

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem malfind

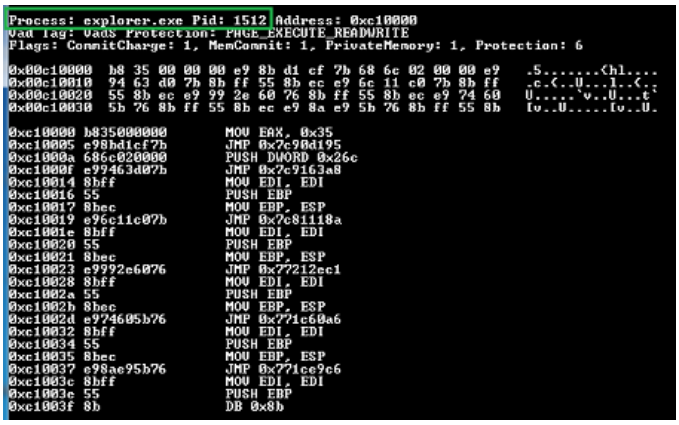

Fig.6 malfind command for hidden or injected code

detect the presence of Zeus. The first memory segment was detected because its executable and has meaning that there is memory mapped file already occupying the space.

I. Yara File Scan (yarascan)

Yara file is a set of rules to identify the behavior of the malware. It contains the list of the strings that is needed to be searched in the memory image for analyzing the behavior of the malware.

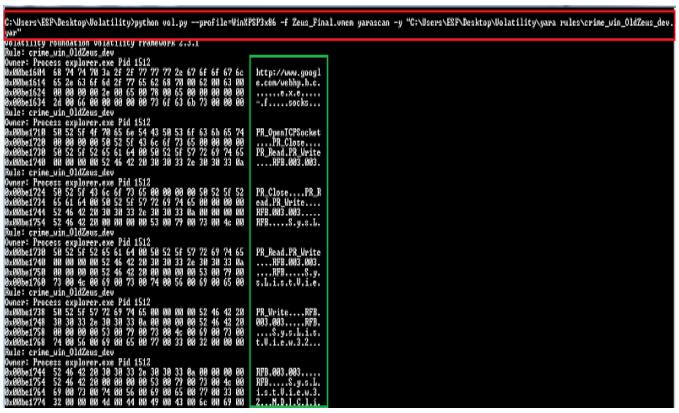

Fig.7 volatilitiy shell

J. Printing registry key (printkey)

This command is display the subkeys, values, data, and data types contained within a specified registry key, use the printkey command.

a. Check for the registry key which was found in zbotscan and zeusscan2.

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem printkey -K "Software\Microsoft\Irpao"

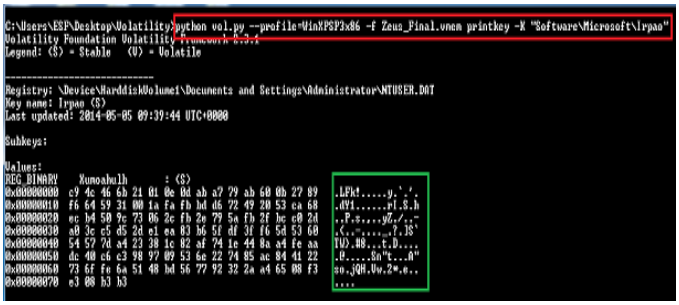

Fig.8 volatilitiy shell

b. "ControlSet001\Services\SharedAccess\Parameters\Firewall Policy\StandardProfile"

It is used to detect if the windows firewall is enabled or disabled.

K. Checking event logs (evtlogs)

python vol.py --profile=WinXPSP2x86 -f Zeus_2_Final.vmem evtlogs -D C:\Users\ESF\Desktop\Dump

The "sysevent.txt" file is the important file to be analyzed from the four files that is acquired by this command.

It shows that a socket operation was attempted to unreachable host. It means that some process of the system has tried to connect to the remote server. It also shows that the computer browser is stopped. It means that at some point of time the computer browser was stared.

VI. CONCLUSION

Volatility tool gives the detailed result of the malware behavior. It shows the infected process among the all running processes in the system, the malware information, the infected executable. It also shows the string of the malware. It states that the malware tries to hide itself behind the legitimate process of the system and infect the system.

REFERENCES

[1] Malware Analyst's cookbook and DVD:Tools and Techniques for Fighting Malicious Code Paperback by Michael Ligh (Author), Steven Adair (Author), Blake Hartstein (Author),Matthew Richard (Author)
[2] Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software Paperback by Michael Sikorski (Author), Andrew Honig (Author)
[3] Mandiant Practical Malware Analysis by Kris Kendall
[4] https://code.google.com/p/volatility/wiki/CommandReference#Image_Identification
[5] https://code.google.com/p/volatility/wiki/LinuxMemoryForensics
[6] http://msdn.microsoft.com/en/US
[7] Symantec : ZeuS: King of the bots by Nicolas Falliere and Eric Chien
[8] www.secarma.co.uk/blog/2013/04/03/how-to-installing-volatility-on-windows-7-64-bit/
[9] SANS CheetSheet v2.3
[10] Reversing and Malware Analysis Training [2012]
[11] Malware Analysis: An Introduction, SANS Institute InfoSec Reading Room, by Author: Dennis Distler