**Analysis:**

1. Memory analysis:

Tools:

1. Volatility framework
2. Yara Rules Repository10

**Procedure:**

1. The volatility framework was used to analyse the memory dump of the system taken by checking the image info:

* *sudo ./vol.py -f /media/sdb1/windows/memory.img imageinfo*

A screenshot of a computer program

Description automatically generated

The profile refers to the operating system whose memory is being analyzed. The kprc indicates the kernel process control block that holds information of the running processes. The Directory Table Base (DTB) indicates the memory space allocation for the operating system. The Kernel Debugger Block (KDBG) contains the kernel information.

To simplify the volatility search to the specified image, use the image info parameters.

1. Verify the running processes within the image dump using the image profile information:

* Sudo ./vol.py -f /media/sdb1/windows/memory.img –kdbg=0x82461820 –dtb=0x1a8000 –kprc=0x8248b000 –profile=Win10x86\_44b89eea pslist.

A screenshot of a computer screen

Description automatically generated

1. The memory dump was scanned for known malware signatures using the Yara Rules Repository10 specifically the CVE\_Rules, Exploit\_Kits and malware directories.

* *find CVE\_Rules/ Exploit\_Kits/ malware/ -name ‘\*.yar’ -exec echo include \”`pwd`/{}\” \: > rules.yar*

A black screen with orange text

Description automatically generated

1. From the specified rules, the memory dump was scanned :

* *sudo /home/enisa/training/tools/vol.py yarascan -f /media/sdb1/Windows/memory.img –profile=Win10x86\_44B89EEA -y rules.yar > results.txt*



1. The scan flagged some known signatures and on counting all the distinct rules found:

* *grep ‘Rule:’ results.txt | sort | unique -c*

A black screen with orange and blue text

Description automatically generated

The SharedStrings, Spyeye\_plugins were likely false positives while the with\_sqlite was too generic upon analysis. The UPX and Xtreme rules however seemed more suspicious.

1. Searching the processes running the UPX and Xtrem rules :

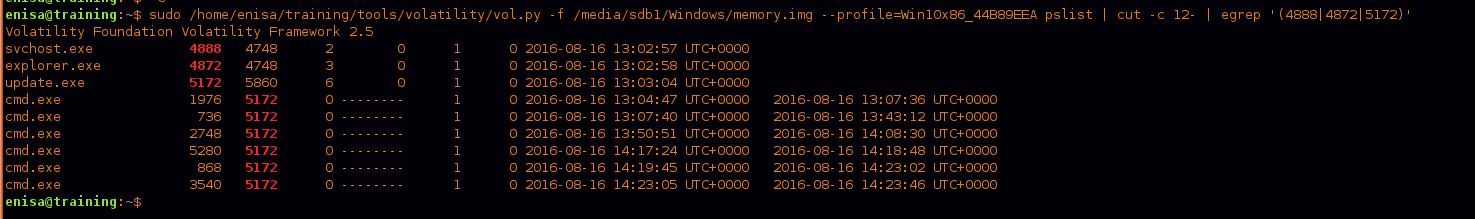
A black screen with orange letters

Description automatically generated

The output indicated the rules detected legitimate processes or malware attempting to mimic legitimate processes.

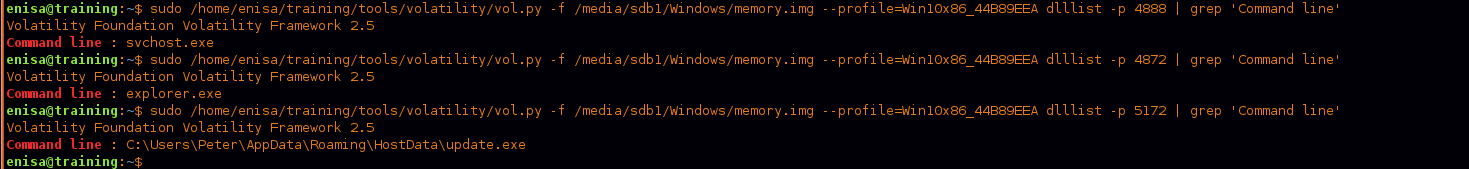
1. The process list for the PIDs of processes containing malicious code from the xtrem and UPX were searched:

* *sudo /home/enisa/training/tools/vol.py yarascan -f /media/sdb1/Windows/memory.img –profile=Win10x86\_44B89EEA pslist | cut -c 12 | egrep ‘(4889|4872|5172)’*



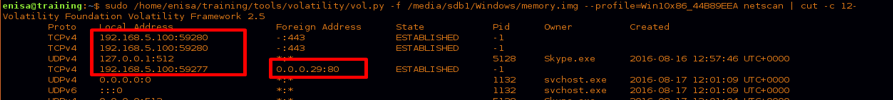
The creation time for each of the processes was determined. From the analysis, the update.exe process might have been used to launch the cmd.exe process .

1. The parent processes svchost.exe, explorer.exe and update.exe were searched and the commandline used to start the process checked:



1. The memory was further analyzed for artefacts to network connections using netscan

* *sudo /home/enisa/training/tools/vol.py yarascan -f /media/sdb1/Windows/memory.img –profile=Win10x86\_44B89EEA netscan | cut -c 12*



This revealed TCP connections to a local host 192.168.5.100.

**Results:**

The memory analysis indicates that the system is running Windows 10 and has several processes, including svchost.exe, update.exe, and explorer.exe, exhibiting suspicious behavior. YARA scans have identified potential malicious activity associated with these processes, suggesting the presence of UPX-compressed or Xtreme malware. The attack is believed to have originated on August 16, 2016, at 13:02:57.