CS 5332 Assignment 4

Detection of the rootkit Stuxnet using Volatility along with ClamAV

The methodology is to investigate whether a known malware that uses rootkit to effect the system and/or hide itself, is in the system or not in an automated fashion so that we do not have to manually investigate almost every outputs of the used plugins. To do that memory sample from a virtual machine that is infected with the rootkit Stuxnet is analyzed. I mostly followed the tutorial from volatility labs guide:

https://volatility-labs.blogspot.com/2016/08/automating-detection-of-known-malware.html

About the rootkit Stuxnet:

Stuxnet is designed to infect Windows systems especially SCADA systems. This rootkit is capable of stealing code, uploading its own code, hiding itself and it "is the first publicly known rootkit that is able to hide injected code located on a programmable logic controller, PLC." Stuxnet uses hooking so that when the software is used to view code blocks on the PLC, the blocks that are injected by the rootkit is undetectable.

Source: https://www.symantec.com/connect/blogs/stuxnet-introduces-first-known-rootkit-scada-devices

Analysis:

The analysis is done on the operating system Kali Linux.

The first plugin of Volatility that is used is dlldump. The command is:

```
$ volatility -f stuxnet.vmem --profile=WinXPSP2x86 dlldump –memory -D stuxout/
```

This command extracts the dll files and main application executables to the output file stuxout. Gathering the output on a file makes the analysis easier since the following investigations will be done on the output of one another.

```
root@kali:~# volatility -f stuxnet.vmem --profile=WinXPSP2x86 dlldump –memory -D stuxout/
Volatility Foundation Volatility Framework 2.6
Process(V) Name                Module Base Module Name          Result
---------- -------------------- ----------- -------------------- ------
0x820df020 smss.exe             0x048580000 smss.exe             OK: module.376.22df020.485
0000.dll
0x820df020 smss.exe             0x07c900000 ntdll.dll            OK: module.376.22df020.7c9
0000.dll
0x821a2da0 csrss.exe            0x04a680000 csrss.exe            OK: module.600.23a2da0.4a6
0000.dll
0x821a2da0 csrss.exe            0x07c900000                      Error: DllBase is paged
0x821a2da0 csrss.exe            0x075b40000 CSRSRV.dll           OK: module.600.23a2da0.75b
0000.dll
0x821a2da0 csrss.exe            0x077f10000 GDI32.dll            Error: DllBase is paged
0x821a2da0 csrss.exe            0x07e720000 sxs.dll              Error: DllBase is paged
0x821a2da0 csrss.exe            0x077e70000 RPCRT4.dll           Error: DllBase is paged
0x821a2da0 csrss.exe            0x077dd0000 ADVAPI32.dll         Error: DllBase is paged
0x821a2da0 csrss.exe            0x077fe0000 Secur32.dll          Error: DllBase is paged
0x821a2da0 csrss.exe            0x075b50000 basesrv.dll          Error: DllBase is paged
0x821a2da0 csrss.exe            0x07c800000 KERNEL32.dll         Error: DllBase is paged
```

As seen below the stuxout file is generated and the input of the file is shown above.

```
a0000.dll
0x81c47c00 lsass.exe            0x05d090000 comctl32.dll         OK: module.1928.1e47c00.5d0
90000.dll
0x81c47c00 lsass.exe            0x077b20000 MSASN1.dll           OK: module.1928.1e47c00.77b
20000.dll
root@kali:~# ls
Desktop    Downloads  Music     Public        stuxout    Videos
Documents  evilgrade  Pictures  stuxnet.vmem  Templates
root@kali:~#
```

Second plugin of volatility that is used is malfind. This plugin is used for detecting malicious executables especially in dll files. The command that is used is:

```
$ volatility vol.py -f stuxnet.vmem --profile=WinXPSP2x86 malfind  -D stuxout/
```

```
Documents  evilgrade  Pictures  stuxnet.vmem  Templates
root@kali:~# volatility -f stuxnet.vmem --profile=WinXPSP2x86 malfind -D stuxout/
Volatility Foundation Volatility Framework 2.6
Process: csrss.exe Pid: 600 Address: 0x7f6f0000
Vad Tag: Vad  Protection: PAGE_EXECUTE_READWRITE
Flags: Protection: 6

0x7f6f0000  c8 00 00 00 1f 01 00 00 ff ee ff ee 08 70 00 00   .............p..
0x7f6f0010  08 00 00 00 00 fe 00 00 00 00 10 00 00 20 00 00   .................
0x7f6f0020  00 02 00 00 00 20 00 00 8d 01 00 00 ff ef fd 7f   .................
0x7f6f0030  03 00 08 06 00 00 00 00 00 00 00 00 00 00 00 00   .................

0x7f6f0000 c8000000          ENTER 0x0, 0x0
0x7f6f0004 1f                POP DS
0x7f6f0005 0100              ADD [EAX], EAX
0x7f6f0007 00ff              ADD BH, BH
0x7f6f0009 ee                OUT DX, AL
0x7f6f000a ff                DB 0xff
```

```
Process: lsass.exe Pid: 868 Address: 0x1000000
Vad Tag: Vad  Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 2, Protection: 6

0x01000000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x01000010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x01000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x01000030  00 00 00 00 00 00 00 00 00 00 00 00 d0 00 00 00   ................

0x01000000 4d                DEC EBP
0x01000001 5a                POP EDX
0x01000002 90                NOP
0x01000003 0003              ADD [EBX], AL
0x01000005 0000              ADD [EAX], AL
0x01000007 000400            ADD [EAX+EAX], AL
0x0100000a 0000              ADD [EAX], AL
0x0100000c ff                DB 0xff
0x0100000d ff00              INC DWORD [EAX]
0x0100000f 00b800000000      ADD [EAX+0x0], BH
0x01000015 0000              ADD [EAX], AL
0x01000017 004000            ADD [EAX+0x0], AL
0x0100001a 0000              ADD [EAX], AL
0x0100001c 0000              ADD [EAX], AL
0x0100001e 0000              ADD [EAX], AL
0x01000020 0000              ADD [EAX], AL
```

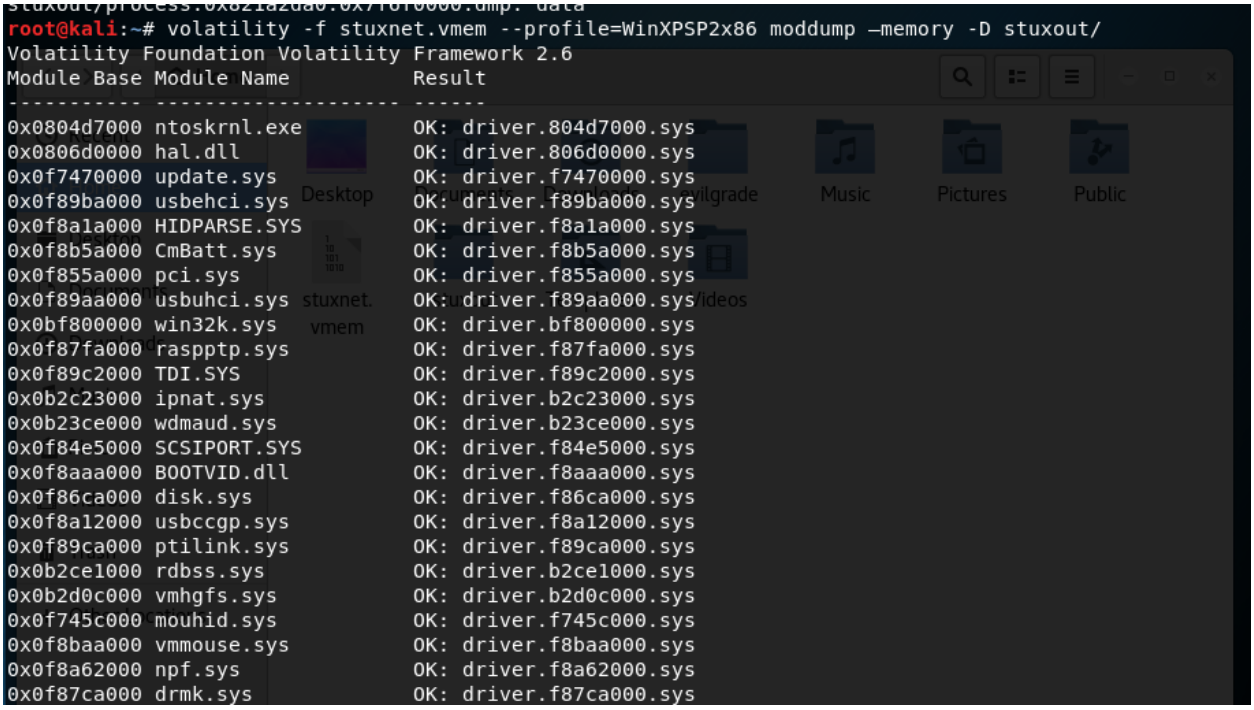
The output of the malfind plugin as seen above is the suspicious memory regions and their addresses. By running the command file on the extracted files, we can detect the probable malicious file.

```
$ file stuxout/*
```

```
root@kali:~# file stuxout/*
stuxout/module.1032.24843e8.1000000.dll:   PE32 executable (GUI) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.3fde0000.dll:  PE32 executable (DLL) (console) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.478c0000.dll:  PE32 executable (DLL) (console) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.4c0a0000.dll:  PE32 executable (DLL) (console) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.4d4f0000.dll:  PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.50000000.dll:  PE32 executable (DLL) (console) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.50040000.dll:  PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.50640000.dll:  PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.59490000.dll:  PE32 executable (DLL) (console) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.597f0000.dll:  PE32 executable (DLL) (console) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.5ad70000.dll:  PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
stuxout/module.1032.24843e8.5b860000.dll:  PE32 executable (DLL) (console) Intel 80386, for MS Windows
```

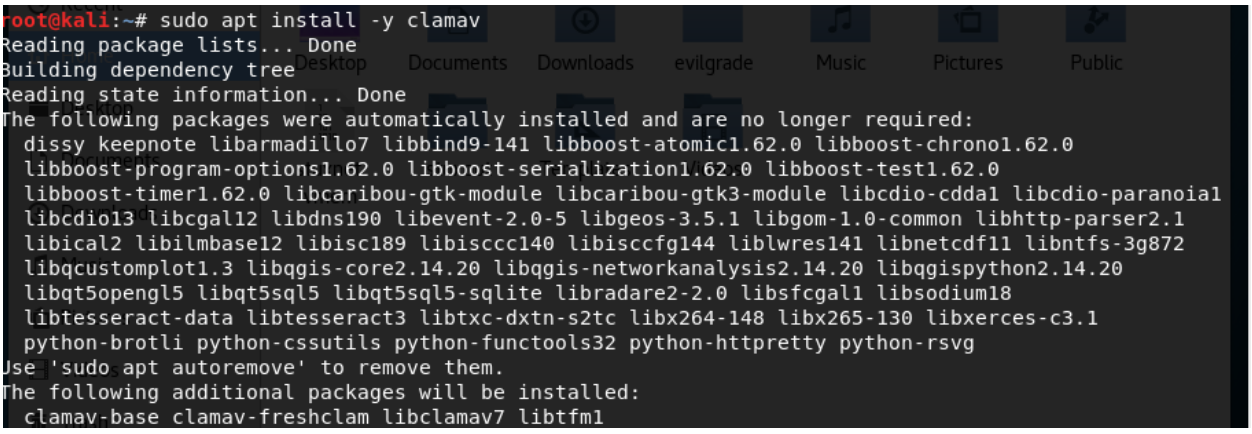
The next plugin is moddump. The moddump extracts all the executable drivers which includes drivers from the kernel memory. By gathering the dll dumps and the kernel dumps to a file the analysis is made easier for the detection with ClamScan. The command is as follows:

```
$ volatility vol.py -f stuxnet.vmem --profile=WinXPSP2x86 moddump –memory -D stuxout/
```



Now that we extracted all the executables that we need for the rootkit detection analysis on a single file we can install the ClamAV which is an open source anti-virus engine that has most of the malwares on their database.

```
$ sudo apt install -y clamav
```



```
$ sudo apt-get install clamav
```

```
$ sudo freshclam
```

```
root@kali:~# sudo apt-get install clamav
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  dissy keepnote libarmadillo7 libbind9-141 libboost-atomic1.62.0 libboost-chrono1.62.0
  libboost-program-options1.62.0 libboost-serialization1.62.0 libboost-test1.62.0
  libboost-timer1.62.0 libcaribou-gtk-module libcaribou-gtk3-module libcdio-cdda1 libcdio-paranoia1
  libcdio13 libcgal12 libdns190 libevent-2.0-5 libgeos-3.5.1 libgom-1.0-common libhttp-parser2.1
  libical2 libilmbase12 libisc189 libisccc140 libisccfg144 liblwres141 libnetcdf11 libntfs-3g872
  libqcustomplot1.3 libqgis-core2.14.20 libqgis-networkanalysis2.14.20 libqgispython2.14.20
  libqt5opengl5 libqt5sql5 libqt5sql5-sqlite libradare2-2.0 libsfcgal1 libsodium18
  libtesseract-data libtesseract3 libtxc-dxtn-s2tc libx264-148 libx265-130 libxerces-c3.1
  python-brotli python-cssutils python-functools32 python-httpretty python-rsvg
```

```
$ clamscan stuxout/ | grep -v ": OK$"
```

```
root@kali:~# clamscan stuxout/ | grep -v ": OK$"

stuxout/module.940.2061da0.d00000.dll: Win.Trojan.Agent-229176 FOUND
stuxout/process.0x81c498c8.0x80000.dmp: Win.Worm.Stuxnet-49 FOUND
stuxout/process.0x81c47c00.0x1000000.dmp: Win.Trojan.5873027-1 FOUND
stuxout/module.1928.1e47c00.870000.dll: Win.Trojan.Agent-229176 FOUND
stuxout/process.0x81e61da0.0xb70000.dmp: Win.Worm.Stuxnet-49 FOUND
stuxout/process.0x81c498c8.0x1000000.dmp: Win.Trojan.5873027-1 FOUND
stuxout/module.1928.1e47c00.1000000.dll: Win.Trojan.Duqu-10 FOUND
stuxout/driver.f895a000.sys: Win.Trojan.Rootkit-8720 FOUND
stuxout/process.0x81c47c00.0x80000.dmp: Win.Worm.Stuxnet-49 FOUND
stuxout/module.868.1e498c8.1000000.dll: Win.Trojan.Duqu-10 FOUND
stuxout/process.0x81c47c00.0x6f0000.dmp: Win.Worm.Stuxnet-49 FOUND
stuxout/module.668.2273020.13f0000.dll: Win.Trojan.Agent-229176 FOUND

----------- SCAN SUMMARY -----------
Known viruses: 4566249
Engine version: 0.100.0
Scanned directories: 1
Scanned files: 1262
Infected files: 12
Data scanned: 580.57 MB
Data read: 611.08 MB (ratio 0.95:1)
Time: 180.631 sec (3 m 0 s)
root@kali:~#
```

The output of the Clamscan shows that the malware is detected. It shows that the system is infected and the name of the rootkit Stuxnet is also shown. This is an easy and fast way to detect if a known rootkit is in the system or not through a memory sample.