1. Sample-14-2.mem is the memory dump of the execution of Sample-11-7-ssdt- rootkit.exe. It is a rootkit that hooks the SSDT. You can download the Sample-14-2.mem in https://drive.google.com/file/d/1QR0WXcsrokWpjiFTrSaIS97TabDpGRB-

Q1-1- Use Volatility to locate and dump kernel modules (hint: use modscan, and moddump commands) 2 points

A screen shot of a computer program

Description automatically generated

^Locating Kernel Modules

Command: python .\vol.py -f C:\Users\kevin\Downloads\Sample-14-2-mem\Sample-14-2.mem --profile Win7SP1x86\_23418 modscan

The profile was obtained using the command: “python .\vol.py -f C:\Users\kevin\Downloads\Sample-14-2-mem\Sample-14-2.mem imageinfo”.

The plugin modscan scans the loaded memory for kernel modules, locating them.

A computer screen shot of a blue screen

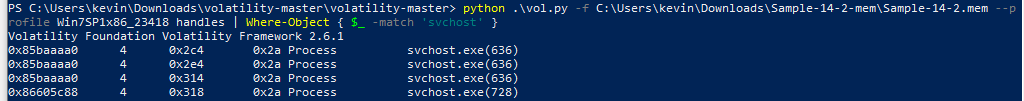
Description automatically generated

^Dumping Kernel Modules

Command: python .\vol.py -f C:\Users\kevin\Downloads\Sample-14-2-mem\Sample-14-2.mem --profile Win7SP1x86\_23418 moddump -D C:\Users\kevin\Downloads\volatility-master\volatility-master\dumpArea

The moddump plugin is used to dump kernel modules into a storage area, which is why this plugin was used.

Q1-2- What is the name of modules you ﬁnd suspicious? What is the reason for detecting it as suspicious one? What is the modules base address? 2 points



^Locating possible suspicious modules that executes malicious Portable Executables through svchost modules. In this case, svchost is chosen initially for analysis into possible malware background activity as Svchost.exe stands for "service host," and malware authors have been known to attach malicious files/payloads towards the service in order to evade detection.

Command: “python .\vol.py -f C:\Users\kevin\Downloads\Sample-14-2-mem\Sample-14-2.mem --profile Win7SP1x86\_23418 handles | Where-Object { $\_ -match 'svchost' }”

A screen shot of a computer program

Description automatically generated

^Suspicious Rootkit like activities have been noticed during the listing of handles, showcasing a possible hook into the SSDT. More notably, in PID 892.

A blue screen with white text

Description automatically generated

^Suspicious Text has been found. “d3b1bbc7-c020-4056-9ded-7c6f40b5a2fc”

Command: “python .\vol.py -f C:\Users\kevin\Downloads\Sample-14-2-mem\Sample-14-2.mem --profile Win7SP1x86\_23418 handles -p 892 -t Mutant”

Through future investigations, the text showcased malicious activities in regard to an information stealer, in the Trojan Malware family as per Open Source Intelligence.

A screenshot of a computer

Description automatically generated

^Source ([alienvault](https://otx.alienvault.com/indicator/file/4d2719868251d27b80b746161fcb2eb78e5ce1927b10c4da5f782ccc51b619e5))

This means that the module detected is determined to indeed be suspicious, due to how it holds evidence of Trojan Malware activities.

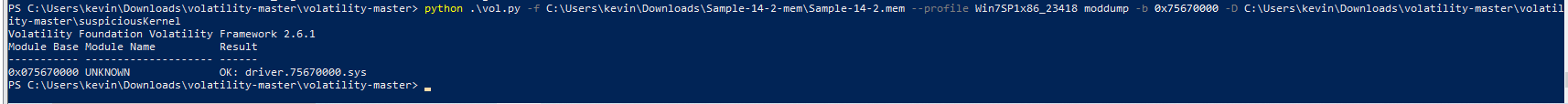
Q1-3- Dump the contents of suspicious kernel module by specifying the base address of the kernel module. 2 points

A computer screen shot of a blue screen

Description automatically generated

^Locating the suspicious Base Address from the PID which contains the string relating to the Trojan Malware.

Command: python .\vol.py -f C:\Users\kevin\Downloads\Sample-14-2-mem\Sample-14-2.mem --profile Win7SP1x86\_23418 ldrmodules -p 892



^Dumping the suspicious kernel module by specifying the Base Address,

Command: python .\vol.py -f C:\Users\kevin\Downloads\Sample-14-2-mem\Sample-14-2.mem --profile Win7SP1x86\_23418 moddump -b 0x75670000 -D C:\Users\kevin\Downloads\volatility-master\volatility-master\suspiciousKernel

Q1-4- Use bintext to see the strings on the modules dump memory. What string you ﬁnd suspicious and why? 2 points

A screenshot of a computer

Description automatically generated

^Suspicious String found

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

(<https://labs.vipre.com/trickbots-tricks/>)

I find the Strings suspicious due to how it matches the kernel API activities of the Trojan Malware Trickbot. This makes sense given the strange mutants found within the module previously containing the same strings that correspond to Trojan Malware families as well.

1. APIMiner:

Q2-1-Execute sample-10-4 by using APIminer. Take snapshot of the successful execution. 2.5 points

A screenshot of a computer

Description automatically generated

^Successful Execution of APIMiner.exe

Q2-2-Read the dump ﬁle and ﬁnd the APIs showing code injection. Take snapshot of the found APIs in the dump ﬁle. 2 points

A screenshot of a computer screen

Description automatically generated

^APIs found showcasing code injection.

Some of the APIs showcasing this includes



^Manipulation of the same obtained file address, injecting the code.

As well as

A screen shot of a computer

Description automatically generated

^Allocating memory and manipulation of the file contents.