Shape

Description automatically generated

Submitted by

Name: Ankit Kumar Singh

Roll no: 53.

Reg no: 11910439

Course code: INT301

CA-3

INTRODUCTION

Objective

Investigate the system run time state of a device (RAM), extract the information present in RAM with the help of volatility Framework.

Description

Volatility is a free memory forensic tool; it is mainly used by malware and SOC analysts within a blue team (people who are on the defensive side in cyber security field). Volatility and its plugins are written in Python. Volatility framework extracts digital artifacts from volatile memory (RAM) samples. We use FTK imager tool for dumping the ram and investigate the dumped (.mem) file, we will be investigating in Windows operating system hence the extension is mem.

Scope

This project is based on analysing the dumped RAM memory through various techniques and finding out information regarding the system like processes, build version, architecture etc, and detecting malwares and rootkits which may be hidden in processes.

System Description

OS: Windows 10

Processor: AMD Ryzen 9 5900HX

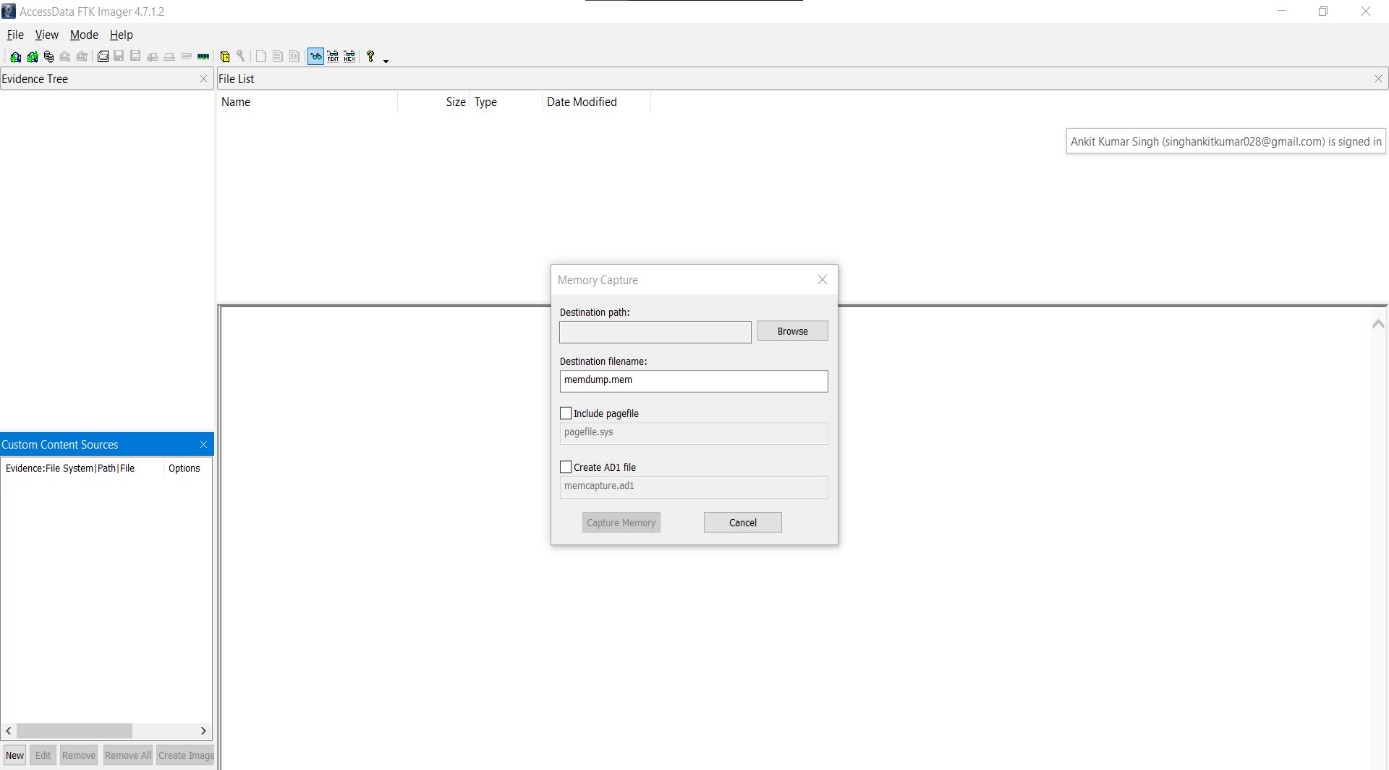
RAM: 16 GB

System type: 64-bit operating system

Analysis report

**Step 1:**

We need to install python, volatility3 framework and FTK imager. After installing all of them open FTK Imager and go to files and select capture memory.



*Download links*

Python: [https://www.python.org/downloads/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbFEyLUdTWWRRQkN5SWZjczRRRXJZN3Q2SzRMUXxBQ3Jtc0tsRVZTMElUSWVtOV9OaE5FNm02Y1JyNG1qa2VYTnFLcjVNdTI2S0prZ01CNnBlU3JOYnBvQmIwRXNPeXdhUGRGcGVzV2FzLXRUcXY4YUdDX2VDZ3FMS2lsOEQwSmN6UG9xSjJoV01ubHJVcWk2RWliWQ&q=https%3A%2F%2Fwww.python.org%2Fdownloads%2F&v=-bMde2glwnE)

Volatility framework: [https://www.volatilityfoundation.org/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbE0wR0VwSFdXSnV2b3ZlQ0hWRDRNelBULUdid3xBQ3Jtc0trT0VkX01LaDBQR1hIUm5odks4NWFYVEtsZV9YaDdSODRtSzFxUXV4X1BHM0VXTHdBU3l1dWhlOFZLVS1xU2pMSTcwZmd5RUViR21GMTQyaXAtWU9sSzNMUmZVLVd6OWxDc0p3TnlzTUtpOHlSdFoxbw&q=https%3A%2F%2Fwww.volatilityfoundation.org%2F&v=-bMde2glwnE)

FTK Imager: [https://www.exterro.com/ftk-imager](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbE9kYzQ5cHVNaWhhVFFJcnBjSEpJUzhQeVhPZ3xBQ3Jtc0trYURLOUhJYnliX3YySldMeWlBWDJYVFdiSUFGQzNNWURlNGRzbXo0ZWhqN096ems1NDg4S041dkRiXzVIVnBUZlQyLXBxSjNJRWRzZnNWWDRMUlRiMWFUNTdsN3l0ZXVaSDlfYnphT0hPYkY5ZHlMdw&q=https%3A%2F%2Fwww.exterro.com%2Fftk-imager&v=-bMde2glwnE)

**Step 2**

Then we create a memory dump of RAM, by clicking on capture memory and browsing it to your favourable location. In my case here it was around 16.4 gb.

Graphical user interface, application

Description automatically generated

**Step 3**

Now we need to open our command prompt in the same location where we stored our captured memory, I have created a folder named “dump\_analysis” where the memdump.mem (memory dump of RAM) is stored.

Graphical user interface

Description automatically generated with medium confidence

**Step 4**

Now after opening command prompt, we need to type: “python vol.py” to check if it’s installed or not, for that we need to be in the same directory where you have installed volatility. We can view help by typing: “python vol.py -h” command.

**Step 5**

Since I am using volatility on windows 10, I need to know the information of the memory dump. So, we will use the “windows.info” plugin for gaining information like, processor and architecture version of the memory.

Syntax: python vol.py -f “<filename with path>” windows.info

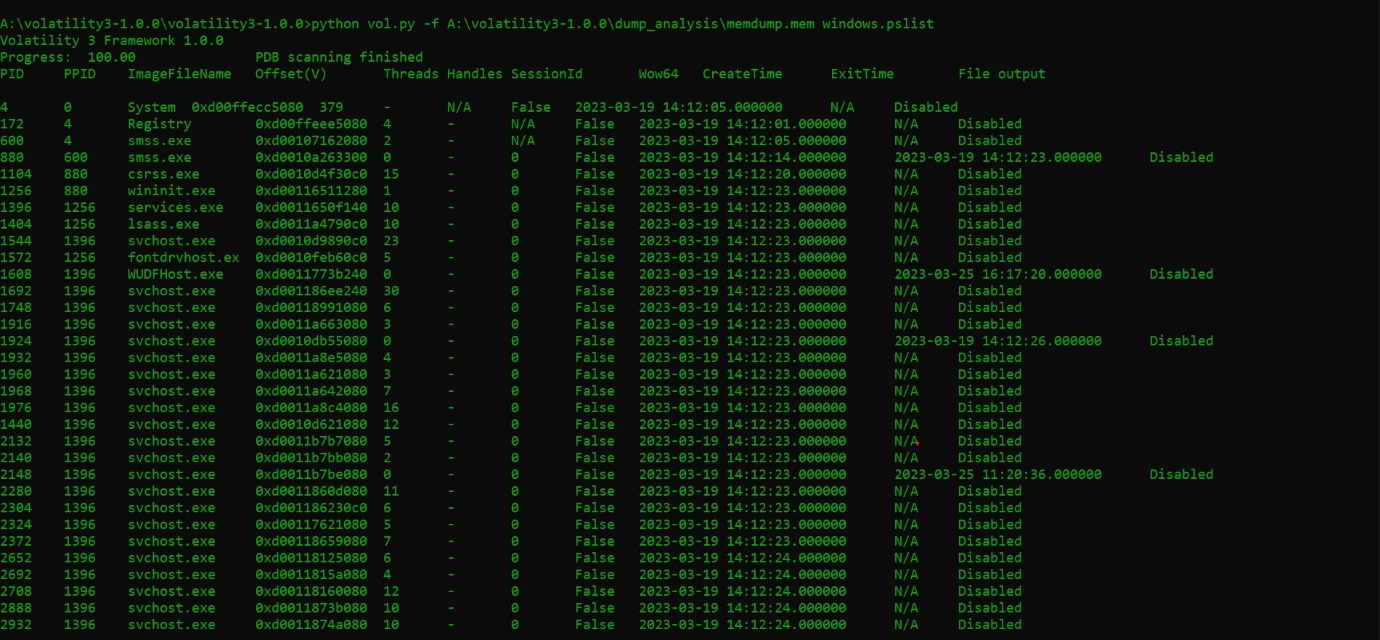
Text

Description automatically generated

**Step 6**

Now we are going to list processes using the plugin “pslist”, this plugin will get the list of processes from doubly linked list that keeps track of processes in memory. It’s like viewing task manager.

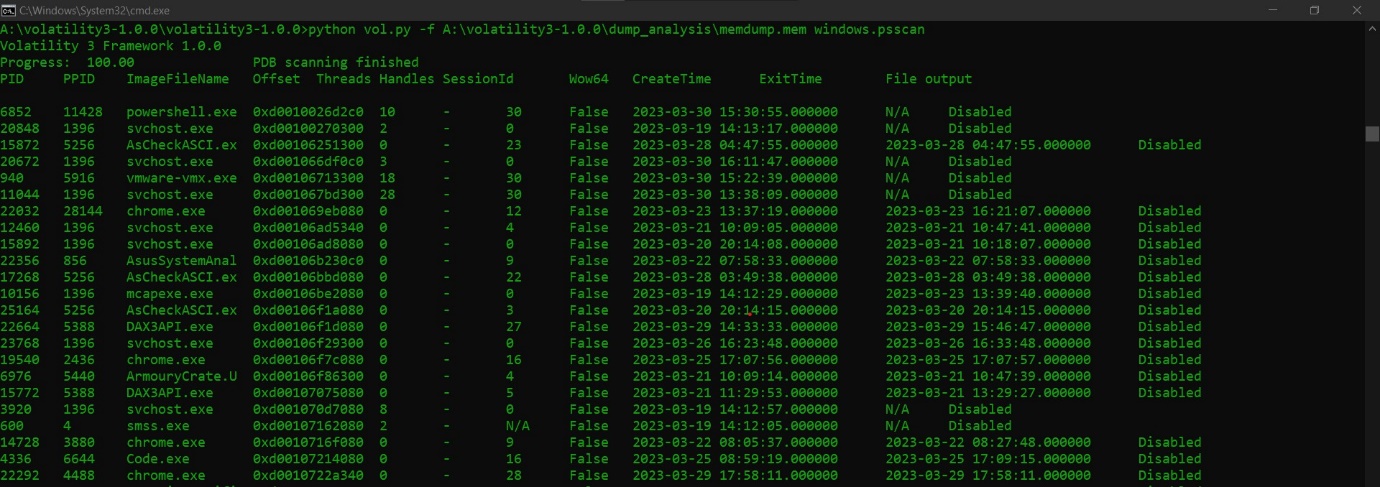
Syntax: python vol.py -f “<filename with path>” windows.pslist



**Step 7**

What we did above was simply listing processes, but sometimes we need to find malicious activity to prevent damage that a malware can do, they hide themselves by unlinking themselves from the list and we won’t be able to see their process. So, to counter this evasion technique use “psscan” this technique can help locate processes by finding the data structure that match \_EPROCESS. But this listing technique can also result in false positives.

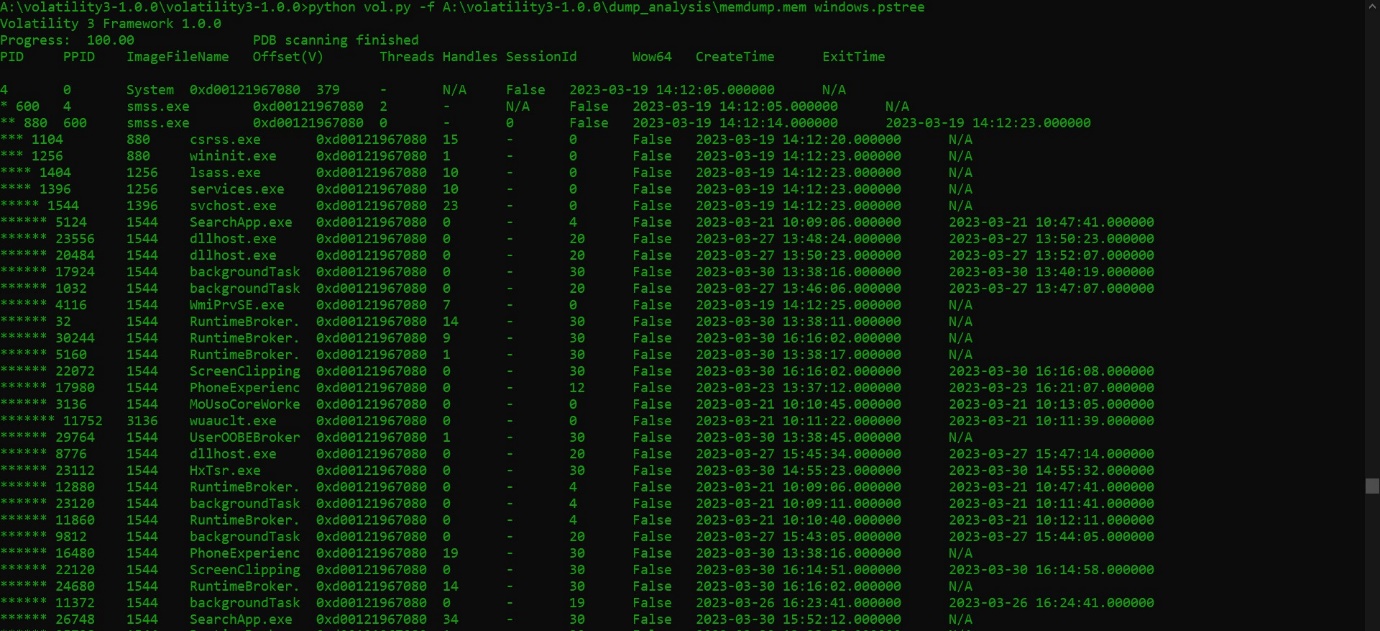
Syntax: python vol.py -f “<filename with path>” windows.psscan



**Step 8**

Sometimes it is useful if we know the full story of processes and what may have occurred during extraction, for this purpose we will be using “pstree” plugin it will list all processes based on their parent process ID.

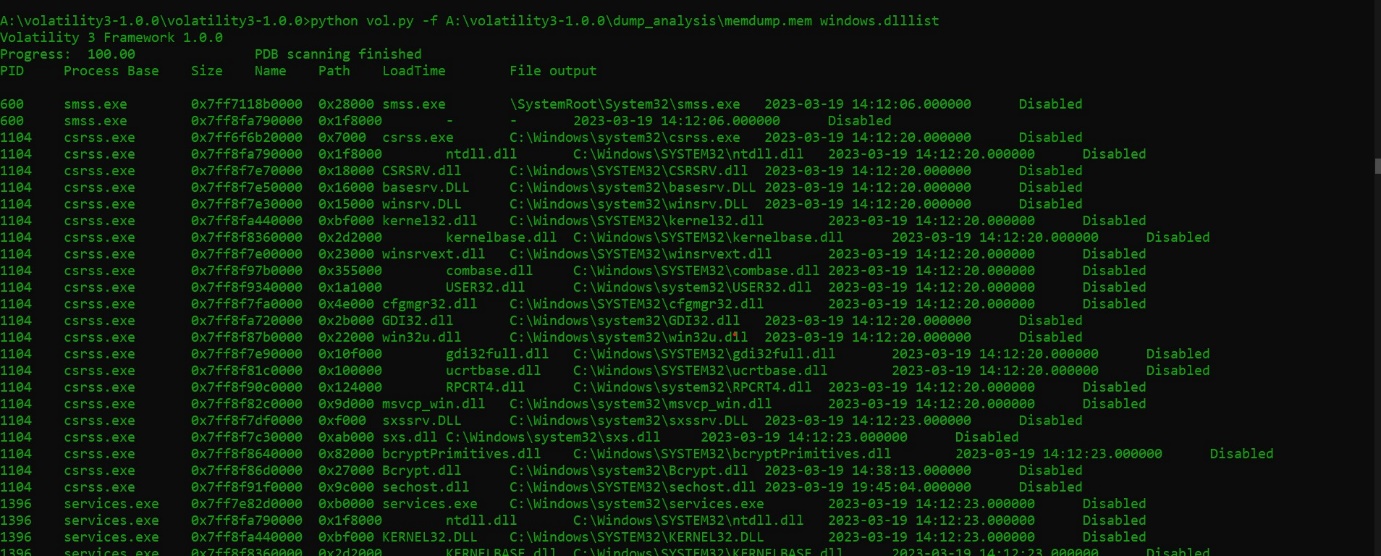
Syntax: python vol.py -f “<filename with path>” windows.pstree



**Step 9**

Now we will be using the plugin “dlllist” it will list all DLLs associated with processes at the time of extraction. This can be especially helpful once you have done further analysis and can filter output to a specific DLL that might be an indicator for a specific type of malware that might be present in the system.

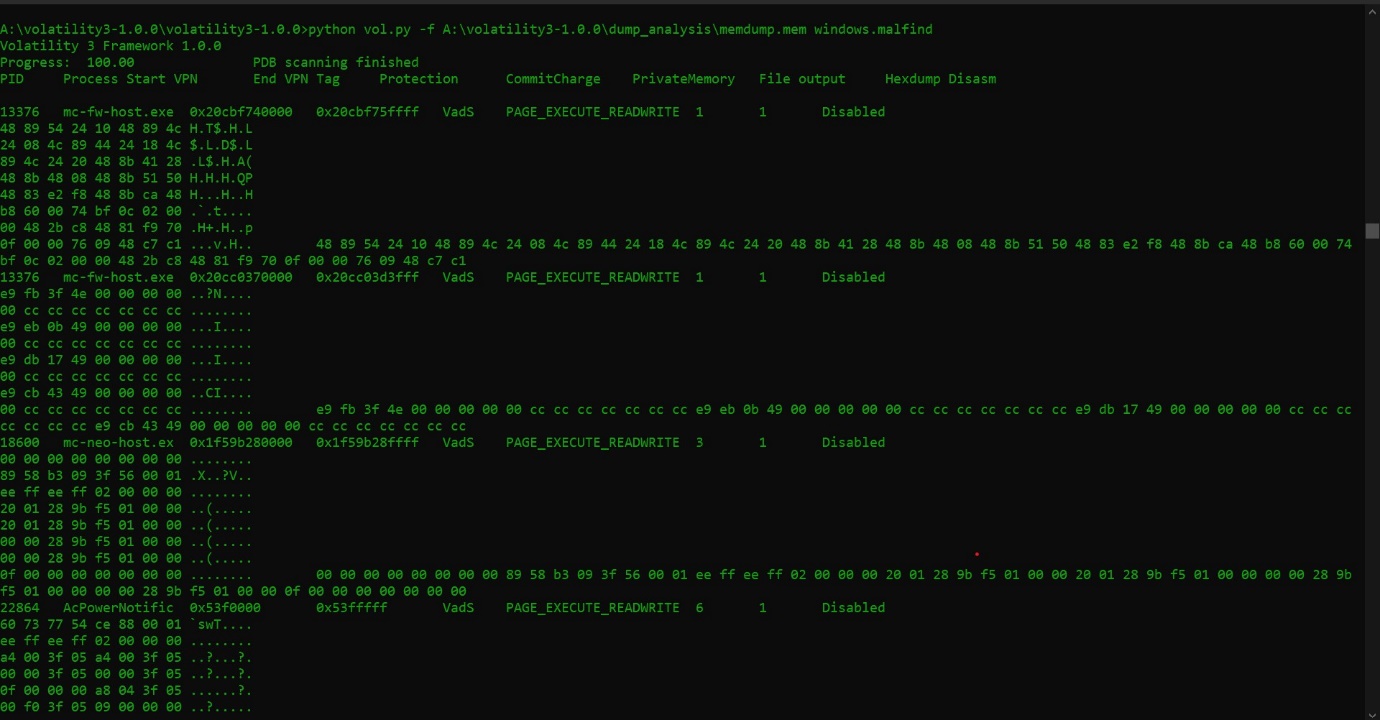
Syntax: python vol.py -f “<filename with path>” windows.dlllist



**Step 10**

There are many plugins that are used for hunting and detecting malwares in a system using volatility one of them being “malfind”. This is useful when hunting for code injection, it identifies injected processes and their PID’s along with their offset address and a Hex, ascii and disassembly view of the infected area.

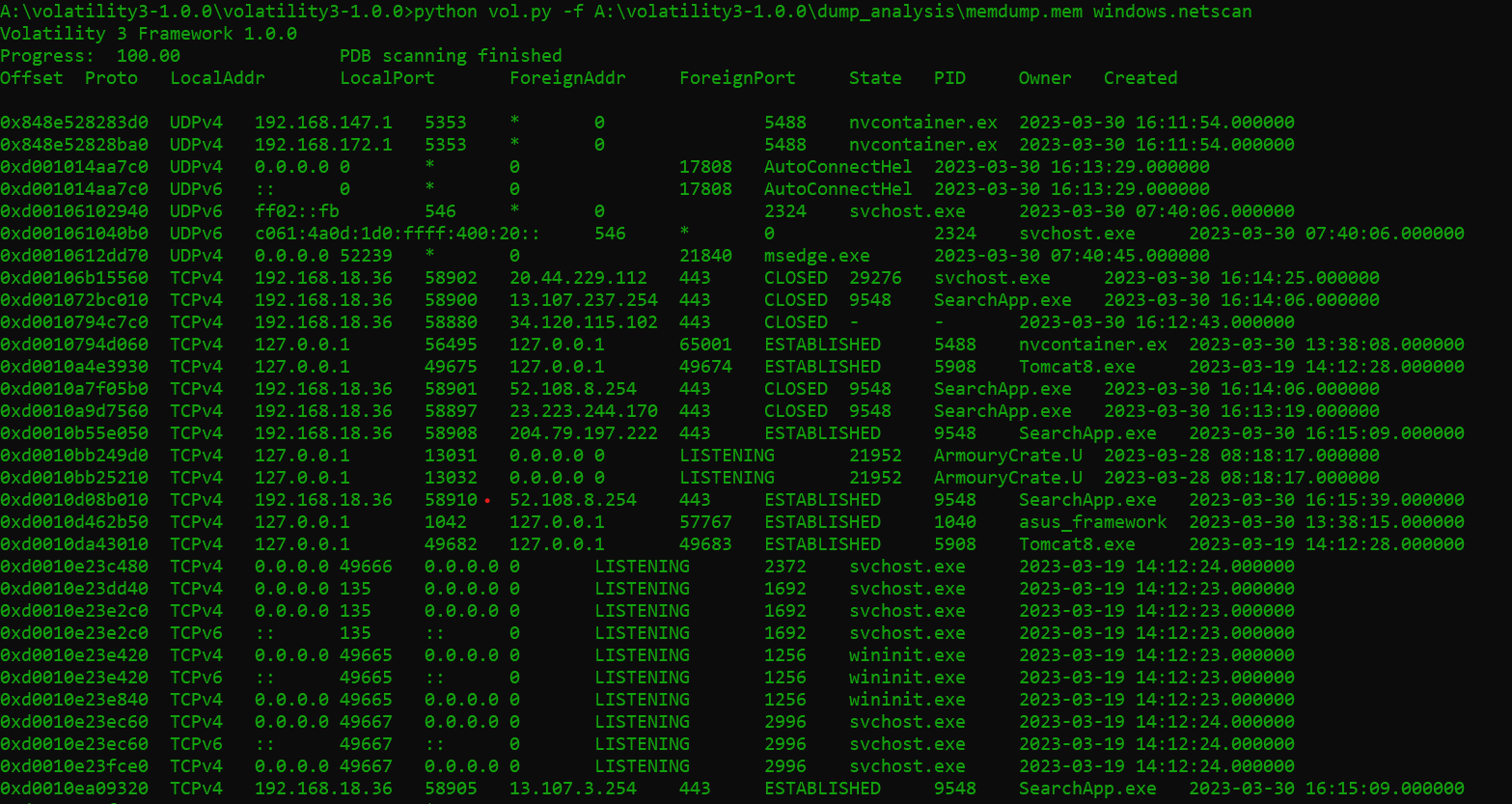
Syntax: python vol.py -f “<filename with path>” windows.malfind



**Step 11:**

The final plugin we will be using is netscan, which will help us identify network connections. When we captured the RAM dump at the same moment the network connections will also get stored in it, this will be helpful in identifying malicious destination IP, source port, destination port, and network activities related to the processes.

Syntax: python vol.py -f <filename> windows.netscan



Reference

1. <https://www.varonis.com/blog/how-to-use-volatility>
2. <https://infosecwriteups.com/forensics-memory-analysis-with-volatility-6f2b9e859765>
3. https://github.com/volatilityfoundation/volatility