Department of Electrical Engineering and Computer Science

Howard University

Washington, DC 20059

EECE 404 Senior Design Research

Fall 2021 - Spring 2022

Memory Forensics using Volatility

By

Patience Jato, Davia McKenzie, Roli Bolorunfe, Obi Oguh

Instructor: Dr. Charles Kim

Date Submitted: April 20, 2022

Summary

As technology and development continues to grow and become more innovative the world is now seeing a shift in our society where the Internet of Things (IOT) and embedded systems has become the norm in our everyday lives. These systems are either computers or utilize the internet. From communication, transportation, entertainment, retail, medical practices etc, the world can see how dependent our society has become on such technology. With that being said, everyone runs into the risk of cyber crimes and activities that can jeopardize operation on large or small scales. Thus, it has increased the need for protection and security. Cybersecurity practices such as Memory Forensics have the capabilities of capturing the memory of compromised devices and performing analysis to identify unusual activities. Our research is a continuation of last year's research. They were able to determine which memory forensic tool provides more efficiency while we delved into the methods of which we will perform Memory Forensics on our system. Developing methods that would be efficient and effective to detect suspicious activity or malware within our RAM.

Problem Statement

Since commonly known attack methods have become increasingly sophisticated, we must determine which memory forensic method would provide the best physical memory coverage against those common attacks in order to support secure operational environments.

The project goals are to research and gain a deeper understanding of Memory Forensics, its operation and importance. This is to gain an understanding of RAM and how the processes communicate with one another to depict the operations or functions within a system. With such information, the team will determine a methodology that will be used to detect suspicious activity, potential malware, within a computer systems' memory.

Design Requirement

To be able to perform research of Memory Forensics there are several financial, software and hardware requirements. This project required a budget of $200 to cover the expense of a hard drive that allowed us to process the data captured from memory, as well as additional software tools. For software requirements, the computers of team members needed to be compatible with our intended Memory Forensic Tool, Volatility 3. This means that, the computers needed to have at least a Windows 10 operating system. The project required the downloading of FTK Imager which was used to capture the RAM, memory of the system. Python version 3 was also needed to run certain commands as well as Git Bash. It was also required for us to have a system processor that was at least 2.5 GHz Dual Core. For hardware requirements, the team needed at least 16 GB RAM to capture the memory and 6 GB to run other applications and software.

Additionally to gain understanding of Memory Forensic the book "The Art of Memory Forensics" provides the team with the tools and information that set the basis of the research. It was also important to understand the industry regulations and standards with the United States, NIST and CFFT. It is also very important to understand the Environmental and Social Responsibility of Memory Forensics and how our data could be impactful.

Solution Design

Using Memory Forensic the team generated two methods that could be used to gather information regarding the functions and process of a captured memory. Using these methods the team was able to test on a clean system and on a "compromised" system that would mimic the behavior or activity of malware. Comparing and contrasting the activities in the process as well as the methods that would be more efficient.

PPID	ImageFileName	Offset(V)	Threads	Handles	SessionId	Wow64	CreateTime	ExitTime	File Output
0	System	0xe184476b4040	209	N/A	FALSE	2022-04-07	9:51:44	N/A	Disabled
4	Registry	E184479E4040	4	N/A	FALSE	2022-04-07	19:51:42	N/A	Disabled
4	smss.exe	E1844993D040	2	N/A	FALSE	2022-04-07	19:51:44	N/A	Disabled
572	csrss.exe	E1844D866140	12	0	FALSE	2022-04-07	19:51:46	N/A	Disabled
572	wininit.exe	E1844E88C080	1	0	FALSE	2022-04-07	19:51:48	N/A	Disabled
816	csrss.exe	E1844E890140	14	1	FALSE	2022-04-07	19:51:48	N/A	Disabled
824	services.exe	E1844E622280	6	0	FALSE	2022-04-07	19:51:48	N/A	Disabled
824	lsass.exe	E1844E90E0C0	10	0	FALSE	2022-04-07	19:51:48	N/A	Disabled
900	svchost.exe	E1844E9802C0	12	0	FALSE	2022-04-07	19:51:48	N/A	Disabled
824	fontdrvhost.ex	E1844E981080	5	0	FALSE	2022-04-07	19:51:48	N/A	Disabled
900	svchost.exe	E1844F019340	11	0	FALSE	2022-04-07	19:51:48	N/A	Disabled
900	svchost.exe	E1844F0412C0	3	0	FALSE	2022-04-07	19:51:48	N/A	Disabled
816	winlogon.exe	E1844F059080	6	1	FALSE	2022-04-07	19:51:48	N/A	Disabled
652	fontdrvhost.ex	E1844F0C21C0	5	1	FALSE	2022-04-07	19:51:48	N/A	Disabled
652	LogonUI.exe	E1844F139240	0	1	FALSE	2022-04-07	19:51:49	19:52:01	Disabled
652	dwm.exe	0xe1844f13b100	14	1	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F140340	2	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F142340	7	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F1340C0	3	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F19F300	4	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F19E080	1	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F20A2C0	6	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	IntelCpHDCPSvc	E1844F208080	3	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F226300	16	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F250340	8	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F255080	6	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F2572C0	3	0	FALSE	2022-04-07	19:51:49	N/A	Disabled
900	svchost.exe	E1844F259340	2	0	FALSE	2022-04-07	19:51:49	N/A	Disabled

Figure 1: Data generated using the PsCommands (PsList)

PID	PPID	ImageFileName	Offset(V)	Threads	Handles	SessionId	Wow64	CreateTime	ExitTime
4	0	System 0xe18447	6b4040 209	-	N/A	FALSE	2022-04-07	19:51:44	N/A
* 420	4	smss.exe	E1844993D040	2	N/A	FALSE	2022-04-07	19:51:44	N/A
* 124	4	Registry	E184479E4040	4	N/A	FALSE	2022-04-07	19:51:42	N/A
* 2380	4	MemCompression	E1844F4B1040	50	N/A	FALSE	2022-04-07	19:51:49	N/A
616	572	csrss.exe	E1844D866140	12	0	FALSE	2022-04-07	219:51:46	N/A
824	572	wininit.exe	E1844E88C080	1	0	FALSE	2022-04-07	19:51:48	N/A
* 664	824	fontdrvhost.ex	E1844E981080	5	0	FALSE	2022-04-07	19:51:48	N/A
* 924	824	lsass.exe	E1844E90E0C0	10	0	FALSE	2022-04-07	19:51:48	N/A
* 900	824	services.exe	E1844E622280	6	0	FALSE	2022-04-07	19:51:48	N/A
** 1540	900	svchost.exe	E1844F20A2C0	6	0	FALSE	2022-04-07	19:51:49	N/A
*** 3844	1540	taskhostw.exe	0xe1844fb20080	7	1	FALSE	2022-04-07	19:51:50	N/A
** 3592	900	svchost.exe	E1844FAE3080	8	0	FALSE	2022-04-07	19:51:49	N/A
** 1548	900	IntelCpHDCPSvc	E1844F208080	3	0	FALSE	2022-04-07	19:51:49	N/A
** 1560	900	svchost.exe	E1844F226300	16	0	FALSE	2022-04-07	19:51:49	N/A
** 3612	900	svchost.exe	E1844FAE2080	5	0	FALSE	2022-04-07	19:51:49	N/A
** 1056	900	svchost.exe	E1844F39C340	7	0	FALSE	2022-04-07	19:51:49	N/A
** 4648	900	svchost.exe	E1844FEBC340	13	0	FALSE	2022-04-07	19:51:50	N/A
** 2604	900	svchost.exe	E1844F69D340	1	0	FALSE	2022-04-07	19:51:49	N/A
** 2092	900	WUDFHost.exe	E1844F882080	9	0	FALSE	2022-04-07	19:51:49	N/A
** 4656	900	svchost.exe	E1844FEBD080	7	0	FALSE	2022-04-07	19:51:50	N/A
** 576	900	svchost.exe	E1844F0412C0	3	0	FALSE	2022-04-07	19:51:48	N/A
** 3140	900	svchost.exe	E1844F8EE2C0	15	0	FALSE	2022-04-07	19:51:49	N/A
** 9796	900	svchost.exe	E18451F8D080	2	0	FALSE	2022-04-07	19:52:09	N/A
** 4168	900	svchost.exe	E1844FD3D280	11	0	FALSE	2022-04-07	19:51:50	N/A
** 1616	900	svchost.exe	E1844F250340	8	0	FALSE	2022-04-07	19:51:49	N/A
** 4180	900	AppleOSSMgr.ex	E1844FD3F280	3	0	FALSE	2022-04-07	19:51:50	N/A
** 2136	900	svchost.exe	E1844F421080	7	0	FALSE	2022-04-07	19:51:49	N/A
** 2652	900	svchost.exe	E1844F6DC2C0	4	0	FALSE	2022-04-07	19:51:49	N/A

Figure 2: Data generated using the PsCommands (PsTree)

PID	PPID	ImageFileName	Offset(V)	Threads	Handles	SessionId	Wow64	CreateTime	ExitTime	File output
4	0	System	0xe184476b4040	209	N/A	FALSE	4/7/2022	19:51:44	N/A	Disabled
124	4	Registry	E184479E4040	4	N/A	FALSE	4/7/2022	19:51:42	N/A	Disabled
420	4	smss.exe	E1844993D040	2	N/A	FALSE	4/7/2022	19:51:44	N/A	Disabled
616	572	csrss.exe	E1844D866140	12	0	FALSE	4/7/2022	19:51:46	N/A	Disabled
900	824	services.exe	E1844E622280	6	0	FALSE	4/7/2022	19:51:48	N/A	Disabled
824	572	wininit.exe	E1844E88C080	1	0	FALSE	4/7/2022	19:51:48	N/A	Disabled
832	816	csrss.exe	E1844E890140	14	1	FALSE	4/7/2022	19:51:48	N/A	Disabled
924	824	lsass.exe	E1844E90E0C0	10	0	FALSE	4/7/2022	19:51:48	N/A	Disabled
464	900	svchost.exe	E1844E9802C0	12	0	FALSE	4/7/2022	19:51:48	N/A	Disabled
664	824	fontdrvhost.ex	E1844E981080	5	0	FALSE	4/7/2022	19:51:48	N/A	Disabled
708	900	svchost.exe	E1844F019340	11	0	FALSE	4/7/2022	19:51:48	N/A	Disabled
576	900	svchost.exe	E1844F0412C0	3	0	FALSE	4/7/2022	19:51:48	N/A	Disabled
652	816	winlogon.exe	E1844F059080	6	1	FALSE	4/7/2022	19:51:48	N/A	Disabled
1060	652	fontdrvhost.ex	E1844F0C21C0	5	1	FALSE	4/7/2022	19:51:48	N/A	Disabled
1192	900	svchost.exe	E1844F1340C0	3	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1160	652	LogonUI.exe	E1844F139240	0	1	FALSE	4/7/2022	19:51:49	19:52:01	Disabled
1168	652	dwm.exe	0xe1844f13b100	14	1	FALSE	4/7/2022	19:51:49	N/A	Disabled
1180	900	svchost.exe	E1844F140340	2	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1188	900	svchost.exe	E1844F142340	7	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1340	900	svchost.exe	E1844F19E080	1	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1332	900	svchost.exe	E1844F19F300	4	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1548	900	IntelCpHDCPSvc	E1844F208080	3	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1540	900	svchost.exe	E1844F20A2C0	6	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1560	900	svchost.exe	E1844F226300	16	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1616	900	svchost.exe	E1844F250340	8	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1664	900	svchost.exe	E1844F255080	6	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1696	900	svchost.exe	E1844F2572C0	3	0	FALSE	4/7/2022	19:51:49	N/A	Disabled
1704	900	svchost.exe	E1844F259340	2	0	FALSE	4/7/2022	19:51:49	N/A	Disabled

Figure 3: Data generated using the PsCommands (PsScan)

Project Implementation Plan

The objective of research is to determine the best method to perform Memory Forensic using volatility that could be used to detect and prevent malicious attacks. To implement the final solutions the team performed sprints that allowed us to form an objective over a course to time. The first sprint, or increment of our project was to generate a memory capture of a clean RAM using the FTK Imager. Clean RAM, referring to the limitations on active applications running on the system during the capture. This required us to install the software and become familiar with its functions that would allow us to capture memory.

Our second sprint was to generate our Memory Forensic tool, Volatility, and our first method that utilized the Ps Commands (PsList, PsTree, PsScan) that would be used to process the system's RAM data. Then the team would determine another method that could be used. By the third sprint our second method, we refer to as Commandline, would be implemented on a system that had active applications and processes. Using this method, we would determine the steps that could be used to identify and process data with the system's memory.

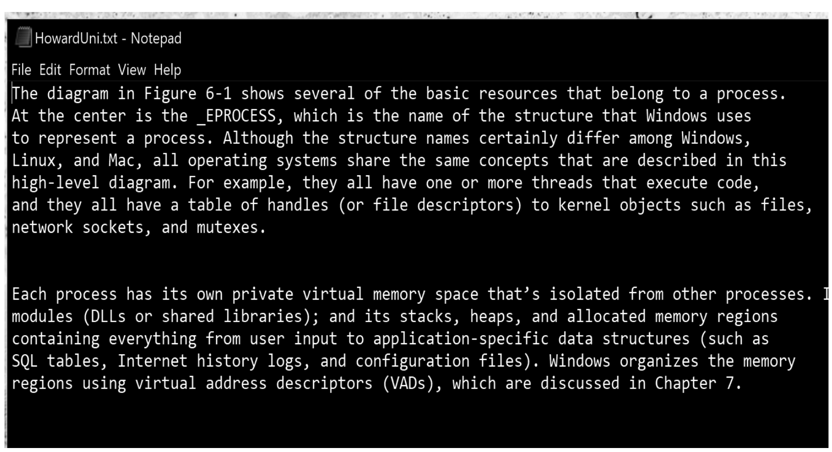
In our fourth, rather last sprint, we would compare the results of using the Ps Commands and Commandline on fairly clean and uncompromised systems. We would then create a system that would be by definition “compromised” by common malware mimicking its behavior and attack methods. As we did with the uncompromised system, we would capture its memory. Using both of the methods to identify suspicious activities that would in turn provide us information on efficiency and effectiveness of PsCommands and Commandline.

<u>Project Implementation Process</u>

➔ **Memory Forensic Software:**

➔ **Text File Implementation:**

For research we wanted to create a text file that would be used as a guide to navigate and identify the memory processes and relationship with a memory dump. We would also use this file to determine when our system has been compromised.


HowardUni.txt - Notepad

File Edit Format View Help

The diagram in Figure 6-1 shows several of the basic resources that belong to a process. At the center is the _EPROCESS, which is the name of the structure that Windows uses to represent a process. Although the structure names certainly differ among Windows, Linux, and Mac, all operating systems share the same concepts that are described in this high-level diagram. For example, they all have one or more threads that execute code, and they all have a table of handles (or file descriptors) to kernel objects such as files, network sockets, and mutexes.

Each process has its own private virtual memory space that’s isolated from other processes. I modules (DLLs or shared libraries); and its stacks, heaps, and allocated memory regions containing everything from user input to application-specific data structures (such as SQL tables, Internet history logs, and configuration files). Windows organizes the memory regions using virtual address descriptors (VADs), which are discussed in Chapter 7.

Figure : HowardUni.txt File

➔ **Memory Capture of Uncompromised System:**

◆ Method 1: Generating PsCommands:

Using the FTK Imager to capture the memory on a clean RAM. We had little to no running programs on our system. Using our Memory Forensic tool, Volatility3, we ran the PsCommands (PsList, PsTree, and PsScan) and analyzed the data. We identified the Notepad.exe process and it was able to provide information on what opened the HowardUni.txt file as we were capturing memory. It was able to confirm what we expected to see in the memory dump. We were also able to see how many processes share the same PID. The processes that shared the same PID allowed us to make sense of the relationship between PID and PPID.

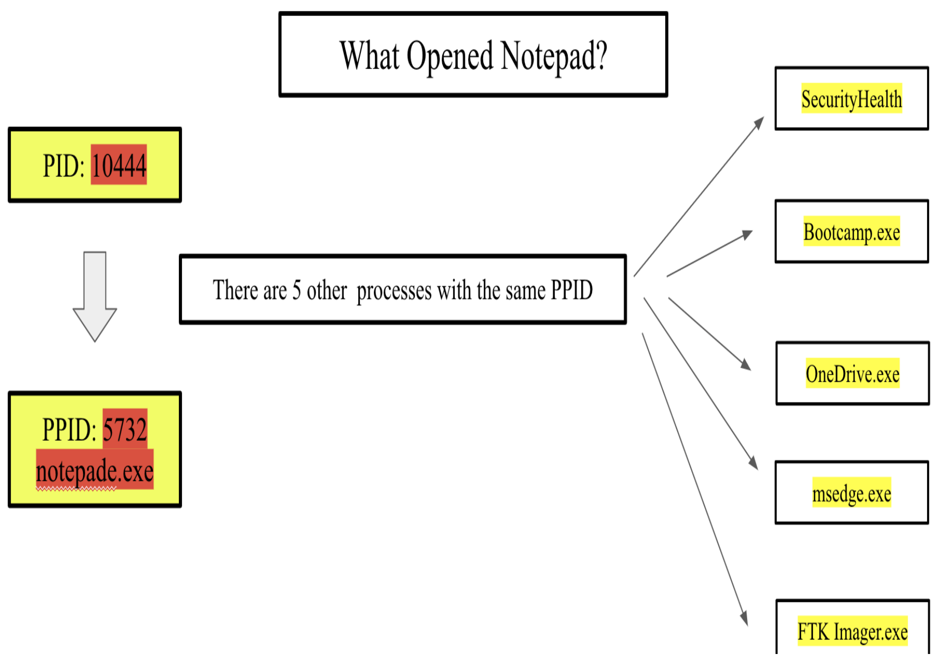


Figure : PID and PPID on a RAM with no running programs

```
C:\Users\patie\Desktop\volatility3\volatility3-develop>python vol.py -f C:\Users\patie\OneDrive\Desktop\memdump.mem windows.pslist --pid 5748
Volatility 3 Framework 2.0.3
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

5748    5600    notepad.exe     0xb20151a020c0  4       -       1       False   2022-03-31 20:33:55.000000      N/A     Disabled
```

Figure 1: Data generated using the PsCommands (PsList)

```
* 1180  728     dwm.exe 0xa50fb7ce6240  28      -       1       False   2022-04-05 02:37:20.000000      N/A
* 5688  728     userinit.exe    0xa50fb8da6340  0       -       1       False   2022-04-05 02:37:22.000000      2022-04-05 02:37:45.000000
** 5732 5688    explorer.exe    0xa50fb8f7f0c0  80      -       1       False   2022-04-05 02:37:22.000000      N/A
*** 9412        5732    SecurityHealth  0xa50fba8c1240  4       -       1       False   2022-04-05 02:37:34.000000      N/A
*** 10444       5732    notepad.exe     0xa50fb9b8b080  7       -       1       False   2022-04-05 02:37:44.000000      N/A
*** 1260        5732    FTK Imager.exe  0xa50fb7dd0080  26      -       1       False   2022-04-05 02:37:53.000000      N/A
*** 9840        5732    msedge.exe      0xa50fba150080  39      -       1       False   2022-04-05 02:37:38.000000      N/A
**** 9856       9840    msedge.exe      0xa50fba152080  9       -       1       False   2022-04-05 02:37:38.000000      N/A
**** 10080      9840    msedge.exe      0xa50fbacc2080  13      -       1       False   2022-04-05 02:37:38.000000      N/A
**** 8960       9840    msedge.exe      0xa50fb8c1d080  17      -       1       False   2022-04-05 02:37:39.000000      N/A
**** 10184      9840    msedge.exe      0xa50fbac4b080  7       -       1       False   2022-04-05 02:37:38.000000      N/A
**** 10092      9840    msedge.exe      0xa50fbad37080  15      -       1       False   2022-04-05 02:37:38.000000      N/A
**** 2604       9840    identity_helpe  0xa50fb8c1e080  10      -       1       False   2022-04-05 02:37:39.000000      N/A
**** 8948       9840    msedge.exe      0xa50fba0af080  16      -       1       False   2022-04-05 02:37:39.000000      N/A
*** 9652        5732    OneDrive.exe    0xa50fba4770c0  39      -       1       False   2022-04-05 02:37:36.000000      N/A
*** 9524        5732    Bootcamp.exe    0xa50fb99cd080  12      -       1       False   2022-04-05 02:37:35.000000      N/A
* 1172  728     LogonUI.exe     0xa50fb745d080  0       -       1       False   2022-04-05 02:37:20.000000      2022-04-05 02:37:38.000000
8660    3524    GoogleCrashHan  0xa50fba589080  5       -       0       True    2022-04-05 02:37:25.000000      N/A
8704    3524    GoogleCrashHan  0xa50fba5910c0  5       -       0       False   2022-04-05 02:37:25.000000      N/A

C:\Users\patie\Desktop\volatility3\volatility3-develop>python vol.py -f C:\Users\patie\OneDrive\Desktop\memdump.mem windows.pstree.PsTree
```

Figure 2: Data generated using the PsCommands (PsTree)

```
C:\Users\patie\Desktop\volatility3\volatility3-develop>python vol.py -f C:\Users\patie\OneDrive\Desktop\memdump.mem windows.psscan --pid 5748
Volatility 3 Framework 2.0.3
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

5748    5600    notepad.exe     0xb20151a020c0  4       -       1       False   2022-03-31 20:33:55.000000      N/A     Disabled
```

Figure 3: Data generated using the PsCommands (PsScan)

- Method 2: Generating Commandline:

Using the FTK Imager to capture the memory of a more active RAM that had active running programs on our system. Using our Memory Forensic tool, Volatility3, we ran the Commandline and analyzed the data that was generated. As seen within Method 1 using the PsCommands, we were able to see similar information regarding the Notepad.exe process. Commandline was also able to provide information on what opened the HowardUni.txt file as we were capturing memory with little investigation. It did not require us to browse through multiple lines of data. As well, confirmed what we expected to see in the memory dump. We were also able to see how many processes share the same PID. In contrast to the cleaner RAM within Method 1, there were more processes in association with the Notepad.exe process.

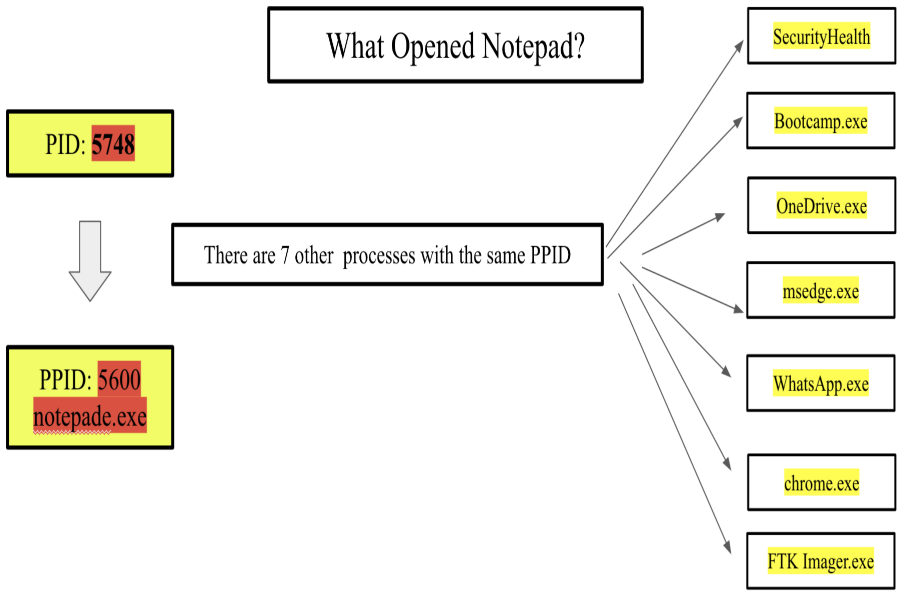
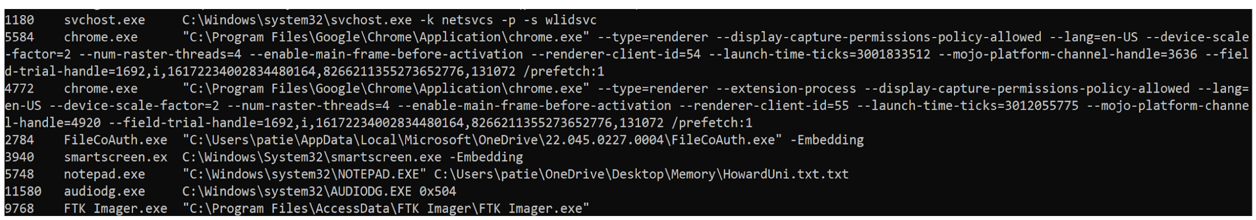


Figure : PID and PPID on a RAM with running programs



```
1180    svchost.exe     C:\Windows\system32\svchost.exe -k netsvcs -p -s wlidsvc
5584    chrome.exe      "C:\Program Files\Google\Chrome\Application\chrome.exe" --type=renderer --display-capture-permissions-policy-allowed --lang=en-US --device-scale
-factor=2 --num-raster-threads=4 --enable-main-frame-before-activation --renderer-client-id=54 --launch-time-ticks=3001833512 --mojo-platform-channel-handle=3636 --fiel
d-trial-handle=1692,i,16172234002834480164,8266211355273652776,131072 /prefetch:1
4772    chrome.exe      "C:\Program Files\Google\Chrome\Application\chrome.exe" --type=renderer --extension-process --display-capture-permissions-policy-allowed --lang=
en-US --device-scale-factor=2 --num-raster-threads=4 --enable-main-frame-before-activation --renderer-client-id=55 --launch-time-ticks=3012055775 --mojo-platform-channe
l-handle=4920 --field-trial-handle=1692,i,16172234002834480164,8266211355273652776,131072 /prefetch:1
2784    FileCoAuth.exe  "C:\Users\patie\AppData\Local\Microsoft\OneDrive\22.045.0227.0004\FileCoAuth.exe" -Embedding
3940    smartscreen.ex  C:\Windows\System32\smartscreen.exe -Embedding
5748    notepad.exe     "C:\Windows\system32\NOTEPAD.EXE" C:\Users\patie\OneDrive\Desktop\Memory\HowardUni.txt.txt
11580   audiodg.exe     C:\Windows\system32\AUDIODG.EXE 0x504
9768    FTK Imager.exe  "C:\Program Files\AccessData\FTK Imager\FTK Imager.exe"
```

Figure : Data generated using CommandLine

➔ Memory Capture of Compromised System:

One of the importance of Memory Forensics revolves around the idea that many malicious attacks start within programs that are then loaded within memory and executed. We were able to determine two methods that could be used with the Volatility 3 tool to generate information and analysis. Using these techniques we need to test them them on a "compromised" device that we know should provide information that could be seen as abnormal in comparison to our uncompromised device. Researching the methods of malicious attacks we were able to create a simple form of a "virus" that we would use to infect the contents of our HowadUni.txt file.

Creating this virus required us to create a python program that would be executed to compromise our system. The “virus”, once the program is executed, will compromise the contents of any .txt file opened in the background or opened after the virus is executed. Will this occured, we captured the memory using FTK Imager. With the memory captured we analyzed the memory dump using the PsCommands and Commandline methods.

Using PsCommands and CommandLine, we were able to generate a model of what an uncompromised system should look like. Using this, we wanted to use these same methods to determine if there would be any abnormalities in the memory’s processes after running our “virus”.

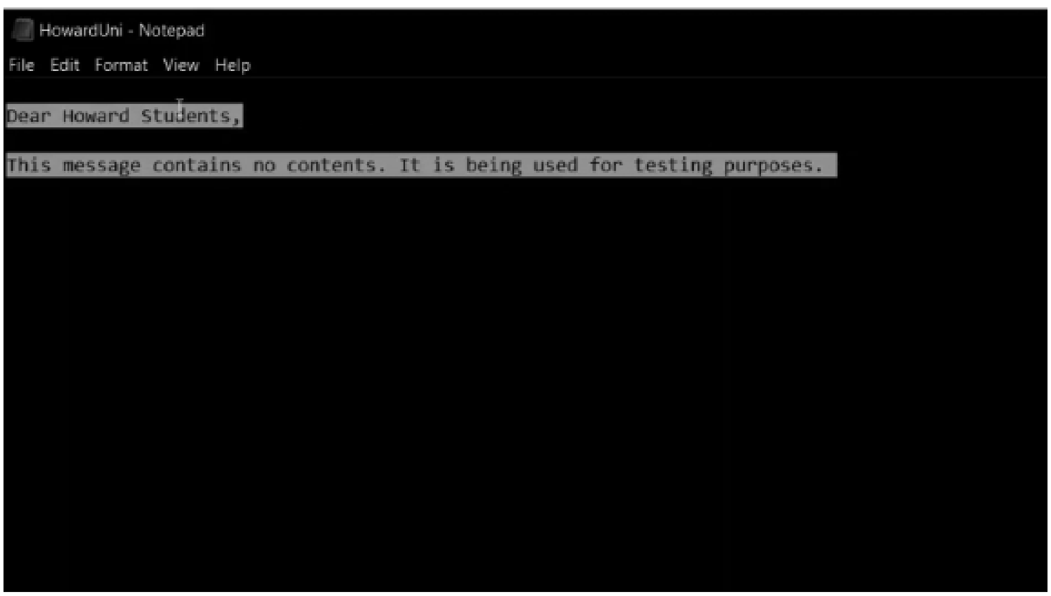


Figure : HowardUni.txt File unaffected by the virus

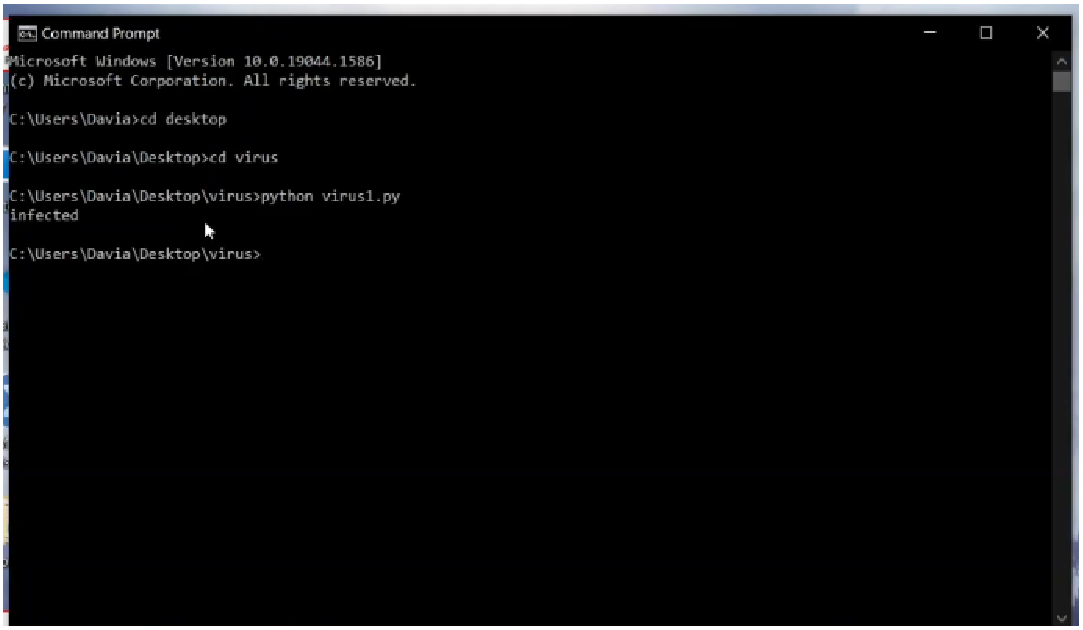


Figure : Running python script that will execute the virus

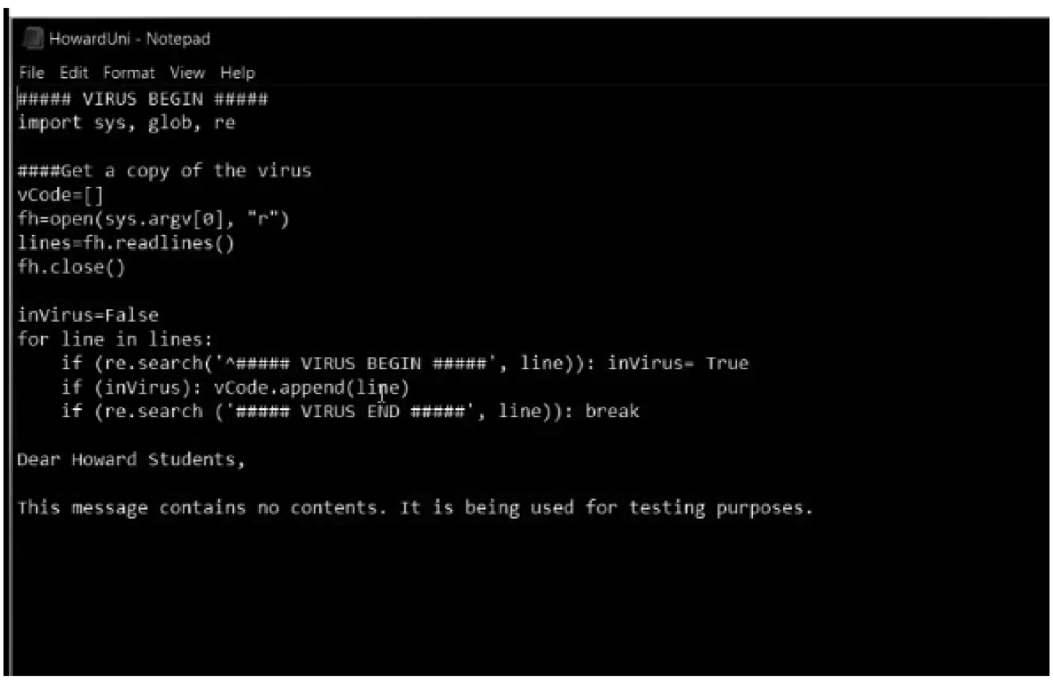


Figure : HowardUni.txt File affected by the virus

```
13220   svchost.exe     C:\Windows\System32\svchost.exe -k NetworkService -p -s DoSvc
13780   SearchApp.exe   "C:\Windows\SystemApps\Microsoft.Windows.Search_cw5n1h2txyewy\SearchApp.exe" -ServerName:CortanaUI.AppX8z9r6jm96hw4bsbneegw0kyxx296wr9t.mca
1948    FTK Imager.exe  "C:\Program Files\AccessData\FTK Imager\FTK Imager.exe"
3988    cmd.exe "C:\Windows\system32\cmd.exe"
3136    conhost.exe     \??\C:\Windows\system32\conhost.exe 0x4
8876    WmiPrvSE.exe    Required memory at 0xef25afa020 is not valid (process exited?)
10112   notepad.exe     "C:\Windows\system32\NOTEPAD.EXE" C:\Users\patie\Desktop\Virus\HowardUni.txt
14260   SearchProtocol  "C:\Windows\system32\SearchProtocolHost.exe" Global\UsGthrFltPipeMssGthrPipe_S-1-5-21-359206446-484028507-2670308062-100121_ Global\UsGthrCtrlFltPipeMssGthrPipe_S-1-5-21-359206446-484028507-2670308062-100121 1 -2147483646 "Software\Microsoft\Windows Search" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT; MS Search 4.0 Robot)" "C:\ProgramData\Microsoft\Search\Data\Temp\usgthrsvc" "DownLevelDaemon"  "1"
13920   SearchProtocol  "C:\Windows\system32\SearchProtocolHost.exe" Global\UsGthrFltPipeMssGthrPipe22_ Global\UsGthrCtrlFltPipeMssGthrPipe22 1 -2147483646 "Software\Microsoft\Windows Search" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT; MS Search 4.0 Robot)" "C:\ProgramData\Microsoft\Search\Data\Temp\usgthrsvc" "DownLevelDaemon"
14212   SearchFilterHo  "C:\Windows\system32\SearchFilterHost.exe" 0 812 816 824 8192 820 792
14284   notepad.exe     Required memory at 0x500ef35020 is not valid (process exited?)
```

Figure : Commandline Data generated from the Compromised system (Virus)

158	**11788**	464	CompPkgSrv.exe	E184501BD080	3	1	FALSE	2022-04-07	20:35:27	N/A	Disabled
159	**8360**	5772	Code.exe	E1844FC82080	31	1	FALSE	2022-04-07	21:02:12	N/A	Disabled
160	**2560**	8360	Code.exe	E1845187B080	7	1	FALSE	2022-04-07	21:02:12	N/A	Disabled
161	**12140**	8360	Code.exe	E184528AE080	14	1	FALSE	2022-04-07	21:02:12	N/A	Disabled
162	**2344**	8360	Code.exe	E184577B5080	14	1	FALSE	2022-04-07	21:02:12	N/A	Disabled
163	**5144**	8360	Code.exe	E184572790C0	20	1	FALSE	2022-04-07	21:02:12	N/A	Disabled
164	**4008**	8360	Code.exe	E18457274080	14	1	FALSE	2022-04-07	21:02:13	N/A	Disabled
165	**5632**	8360	Code.exe	E184563F4080	23	1	FALSE	2022-04-07	21:02:14	N/A	Disabled
166	**8664**	5632	Code.exe	E18456FC2080	12	1	FALSE	2022-04-07	21:02:14	N/A	Disabled
167	**5188**	5632	Code.exe	E184511D4080	14	1	FALSE	2022-04-07	21:02:14	N/A	Disabled
168	[illegible]	464	[illegible]	[illegible]	0	1	FALSE	[illegible]	[illegible]	[illegible]	[illegible]
143	**8844**	900	svchost.exe	E18451989080	7	0	FALSE	2022-04-07	19:53:51	N/A	Disabled
144	**10628**	900	WUDFHost.exe	E184528A0080	6	0	FALSE	2022-04-07	19:59:05	N/A	Disabled
145	**11260**	1272	python.exe	E184507CF080	0	1	FALSE	2022-04-07	19:59:14	19:59:18	Disabled
146	**9864**	464	dllhost.exe	E184527E3080	9	1	FALSE	2022-04-07	19:59:39	N/A	Disabled
147	**9492**	464	FileCoAuth.exe	E184520E1300	8	1	FALSE	2022-04-07	20:00:27	N/A	Disabled

Figure : PsList generated from the Compromised system (Virus)

202	**11260**	1272	python.exe	E184507CF080	0	1	FALSE	2022-04-07	19:59:14	19:59:18
203	**11732**	9788	Code.exe	E18456B40080	0	1	FALSE	2022-04-07	20:10:15	21:02:02
204	**9576**	588	Code.exe	E184573AD080	0	1	FALSE	2022-04-07	20:10:20	21:02:01
205	**10336**	580	Code.exe	E18456ADE080	0	1	FALSE	2022-04-07	20:11:13	20:11:14
206	**508**	8916	Code.exe	E18456A1A080	0	1	FALSE	2022-04-07	20:11:42	20:11:43

Figure : PsTree generated from the Compromised system (Virus)

Conclusions

We were able to generate two methods that could be used to analyze data provided by the memory dump of our systems. Memory dump is a snapshot of a system's memory at a specific instant of time/process. The first method requires us to generate the Ps Commands. Through this we can analyze the processes of the entire system that gives us information of what process has been opened, closed, and even what process opened other processes (PPID and PID relationship). The other method requires the use of a Commandline that instantly provides information of our intended process. Utilizing both methods on a compromised and uncompromised system we were able to determine which method would be more effective and efficient in providing Memory Forensic. Both methods were able to provide information and data result of what occurred with the system memory, however after weighing the pros and cons Commandline was more reactive and quick with providing adequate information that could be used to identify suspicious activities within the memory.

References

Ligh, M. H., Case, A., Levy, J., & Walters, A. (2014). The art of memory forensics: detecting malware and threats in windows, linux, and Mac memory. John Wiley & Sons.

Paul1. (2020, November 10). Malware analysis: Memory forensics with volatility 3. On The Hunt. Retrieved April 19, 2022, from https://newtonpaul.com/malware-analysis-memory-forensics-with-volatility-3/

Pearson, A. (2021, May 10). Volatility 3 CheatSheet. onfvp. Retrieved April 01, 2022, from https://blog.onfvp.com/post/volatility-cheatsheet/

Volatilityfoundation. (n.d.). Command reference · volatilityfoundation/volatility wiki. GitHub. Retrieved March 17, 2022, from https://github.com/volatilityfoundation/volatility/wiki/Command-Reference