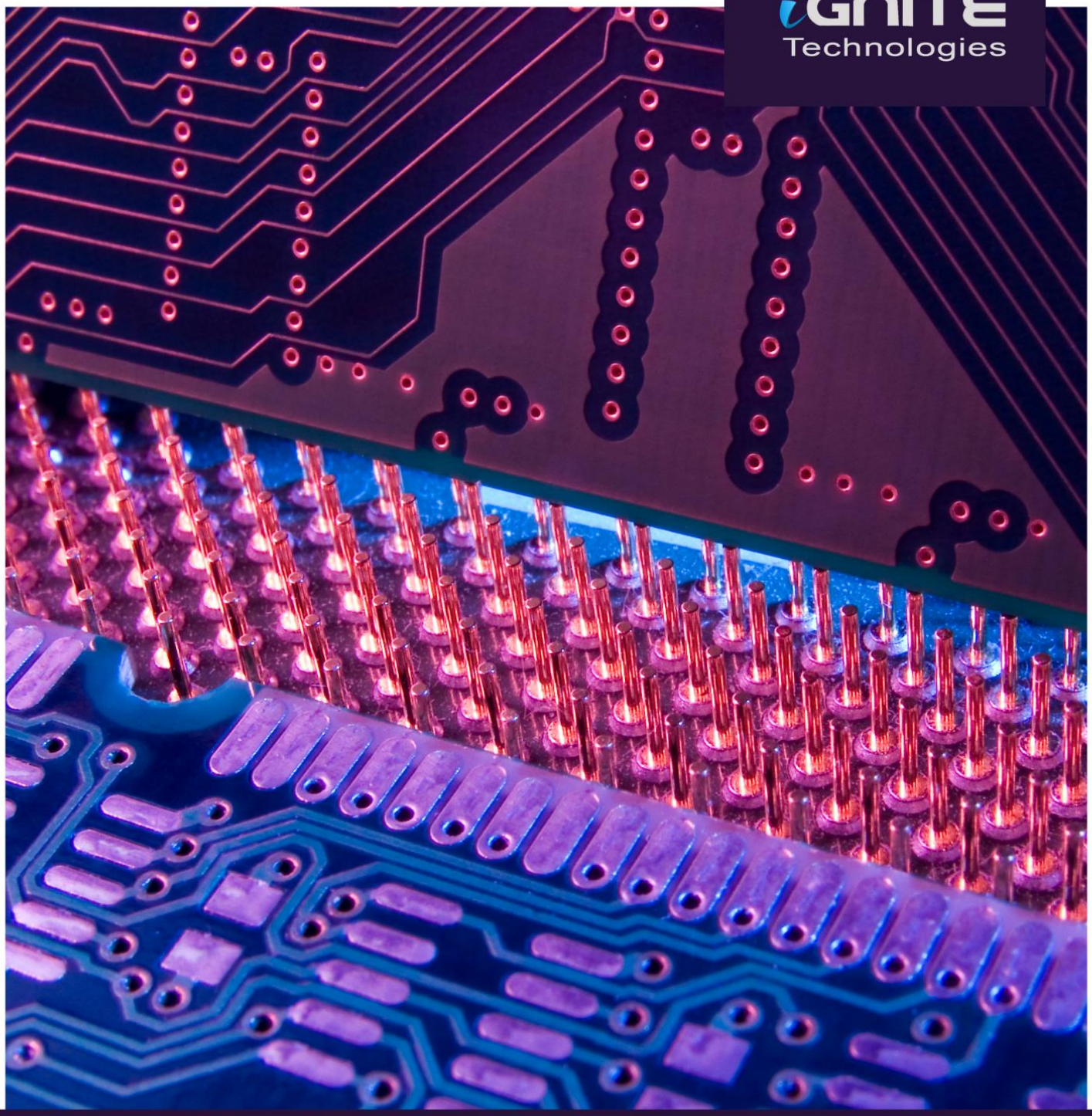
iGNITE Technologies



MEMORY FORENSICS

VOLATILITY

FRAMEWORK & WORKBENCH

WWW.HACKINGARTICLES.IN

Table of Contents

Abstract

Cyber Criminals and **attackers** have become so creative in their crime type that they have started finding methods to hide data in the **volatile memory** of the systems. Today, in this article we are going to have a greater understanding of live **memory acquisition** and its **forensic analysis**. Live Memory acquisition is a method that is used to collect data when the system is found in an active state at the scene of the crime.

Memory forensics is a division of digital forensics that generally emphasizes extracting **artefacts** from the volatile memory of a system that was compromised. This domain is speedily spreading in cybercrime investigations. The main reason for this is that certain artefacts are extracted from system memory only and cannot be found anywhere else.

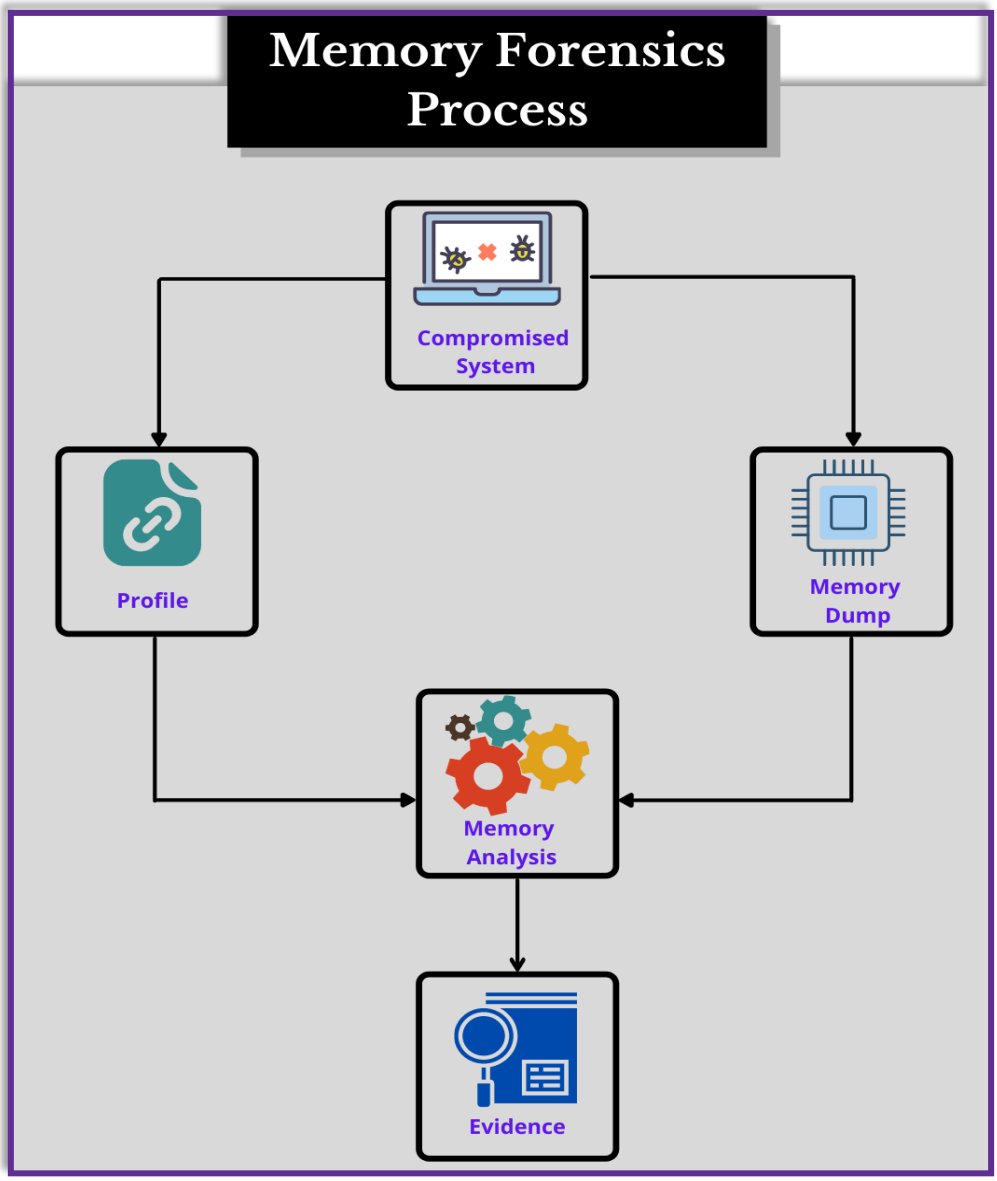
Analysing memory after **capturing the ram** is extremely important when it comes to collecting information on ports that were in use, the number of processes running, and the path of certain executables on the system while carrying out the investigation. The **Volatility Framework** is one such memory analysis tool that works on **command-line** on **Windows** and **Linux** systems.

Volatility Workbench is a **GUI version** of one of the same tool Volatility for analysing the artefacts from a memory dump. It is available free of cost, open-source, and runs on the Windows Operating system.

Introduction

Memory Forensics

Memory Forensics is a budding field in Digital Forensics Investigation which involves recovering, extracting and analysing evidence such as images, documents, or chat histories etc from the structured volatile memory into non-volatile devices like Hard-drives or USB drives.



NOTE: *We have taken a memory dump of a Windows7 system using the Belkasoft RAM Capturer, which can be downloaded from here.*

Memory Acquisition

- It is the method of capturing and dumping the contents of a volatile content into a non-volatile storage device to preserve it for further investigation.
- A ram analysis can only be successfully conducted when the acquisition has been performed accurately without corrupting the image of the volatile memory.
- In this phase, the investigator has to be careful about his decisions to collect the volatile data as it won't exist after the system undergoes a reboot.
- The volatile memory can also be prone to alteration of any sort due to the continuous processes running in the background.
- Any external move made on the suspect system may impact the device's ram adversely.

Importance of Memory Acquisition

When a volatile memory is captured, the following artefacts can be discovered which can be useful to the investigation:

- On-going processes and recently terminated processes
- Files mapped in the memory (.exe, .txt, shared files, etc.)
- Any open TCP/UDP ports or any active connections
- Caches (clipboard data, SAM databases, edited files, passwords, web addresses, commands)
- Presence of hidden data, malware, etc.

Memory Analysis

Once the dump is available, we will begin with the forensic analysis of the memory using the Volatility Memory Forensics Framework which can be downloaded from **here**. The volatility framework support analysis of **memory dump** from all the versions and services of Windows from **XP** to **Windows 10**. It also supports **Server 2003** to **Server 2016**. In this article, we will be analysing the memory dump in Kali Linux where Volatility comes pre-installed.

NOTE: **Dump Format Supported-** Raw format, Hibernation File, VM snapshot, Microsoft crash dump

Volatility Framework

Volatility Framework processes RAM dumps in various formats which can be used to process crash dumps, hibernation files and, page files that may be found on dumps of storage drives. RAM dumps from virtual machines or hypervisors can also be processed.

Data found using Volatility Framework

A huge amount of data can be availed on analysing volatile memory. It includes data like processes, information on open files, registry handles, information on the network and open ports, passwords and cryptographic keys, hidden data, worms and rootkits etc.

Switch on your Kali Linux Machines, and to get a basic list of all the available options, plugins, and flags to use in the analysis, you can type:

```
volatility -h
```

1. Imageinfo

When a Memory dump is taken, it is extremely important to know the information about the operating system that was in use. Volatility will try to read the image and suggest the related profiles for the given memory dump. The image info plugin displays the date and time of the sample that was collected, the number of CPUs present, etc. To obtain the details of the ram, you can type;

```
volatility -f ram.mem imageinfo
```

```
root@kali:~# volatility -f ram.mem imageinfo
Volatility Foundation Volatility Framework 2.6
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_24000,
                     AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace (/root/ram.mem)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf80002bfc0a0L
          Number of Processors : 4
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff80002bfdd00L
                KPCR for CPU 1 : 0xfffff880009f1000L
                KPCR for CPU 2 : 0xfffff8800316a000L
                KPCR for CPU 3 : 0xfffff880031e1000L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2020-10-01 16:27:05 UTC+0000
     Image local date and time : 2020-10-01 21:57:05 +0530
```

A profile is a categorization of specific operating systems, versions and their hardware architecture, A profile generally includes metadata information, system call information, etc. You may notice multiple profiles would be suggested to you.

2. Kdbgscan

This plugin finds and analyses the profiles based on the Kernel debugger data block. The Kdbgscan thus provides the correct profile related to the raw image. It is extremely important to get the right profile for memory analysis. To supply the correct profile for the memory analysis, type

```
volatility -f ram.mem kdbgscan
```

```
root@kali:~# volatility -f ram.mem kdbgscan
Volatility Foundation Volatility Framework 2.6
**************************************************
Instantiating KDBG using: /root/ram.mem WinXPSP2x86 (5.1.0 32bit)
Offset (P)                    : 0x2bfc0a0
KDBG owner tag check          : True
Profile suggestion (KDBGHeader): Win7SP1x64
PsActiveProcessHead           : 0x2c32b90
PsLoadedModuleList            : 0x2c50e90
KernelBase                    : 0xfffff80002a0b000

**************************************************
Instantiating KDBG using: /root/ram.mem WinXPSP2x86 (5.1.0 32bit)
Offset (P)                    : 0x2bfc0a0
KDBG owner tag check          : True
Profile suggestion (KDBGHeader): Win7SP0x64
PsActiveProcessHead           : 0x2c32b90
PsLoadedModuleList            : 0x2c50e90
KernelBase                    : 0xfffff80002a0b000

**************************************************
Instantiating KDBG using: /root/ram.mem WinXPSP2x86 (5.1.0 32bit)
Offset (P)                    : 0x2bfc0a0
KDBG owner tag check          : True
Profile suggestion (KDBGHeader): Win2008R2SP1x64
PsActiveProcessHead           : 0x2c32b90
PsLoadedModuleList            : 0x2c50e90
KernelBase                    : 0xfffff80002a0b000
```

3. Processes

When a system is in an active state it is normal for it to have multiple processes running in the background and can be found in the volatile memory. It consists of executable program code, imported libraries, allocated memory, execution threads. The presence of any hidden process can also be parsed out of a memory dump. The recently terminated processes before the reboot can also be recorded and analysed in the memory dump. There are a few plugins that can be used to list the processes to carry out forensic Investigation.

PSlist

To identify the presence of any rogue processes and to view any high-level running processes.
On executing this command, the list of processes running is displayed, their respective process ID assigned to them and the parent process ID is also displayed along. The details about the threads, sessions, handles are also mentioned. The timestamp according to the start of the process is also displayed. This helps to identify whether an unknown process is running or was running at an unusual time. It will not give information about processes that were hidden by removing themselves from the process list or the ones that were terminated before

```
volatility -f ram.mem --profile=Win7SP1x64 pslist -P
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 pslist -P
Volatility Foundation Volatility Framework 2.6
Offset(P)          Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start
------------------ ------------------------ ----- ------ ------ -------- ------ ------ ------------------------------
0x000000013fece890 System                    4      0    103      542 ------      0 2020-10-01 16:24:31 UTC+0000
0x000000013f4a02f0 smss.exe                268      4      2       32 ------      0 2020-10-01 16:24:31 UTC+0000
0x000000013ed04060 csrss.exe               352    344      9      504      0      0 2020-10-01 16:24:35 UTC+0000
0x000000013ead82f0 csrss.exe               408    400     10      279      1      0 2020-10-01 16:24:36 UTC+0000
0x000000013ead2a90 wininit.exe             416    344      3       78      0      0 2020-10-01 16:24:36 UTC+0000
0x000000013eb12060 winlogon.exe            464    400      4      115      1      0 2020-10-01 16:24:36 UTC+0000
0x000000013eb32780 services.exe            512    416     11      229      0      0 2020-10-01 16:24:37 UTC+0000
0x000000013eb68450 lsass.exe               520    416      8      595      0      0 2020-10-01 16:24:38 UTC+0000
0x000000013eb69600 lsm.exe                 528    416     12      203      0      0 2020-10-01 16:24:38 UTC+0000
0x000000013eba9b30 svchost.exe             620    512     12      376      0      0 2020-10-01 16:24:39 UTC+0000
0x000000013ebe7b30 svchost.exe             704    512      7      289      0      0 2020-10-01 16:24:39 UTC+0000
0x000000013e830b30 svchost.exe             800    512     23      448      0      0 2020-10-01 16:24:40 UTC+0000
0x000000013e848890 svchost.exe             840    512     20      433      0      0 2020-10-01 16:24:40 UTC+0000
0x000000013e853b30 svchost.exe             868    512     49     1114      0      0 2020-10-01 16:24:40 UTC+0000
0x000000013e87bb30 audiodg.exe             944    800      6      130      0      0 2020-10-01 16:24:40 UTC+0000
0x000000013e8a9b30 svchost.exe             128    512     12      550      0      0 2020-10-01 16:24:41 UTC+0000
0x000000013e8ce060 svchost.exe             400    512     25      634      0      0 2020-10-01 16:24:41 UTC+0000
0x000000013e9331b0 spoolsv.exe            1040    512     14      289      0      0 2020-10-01 16:24:43 UTC+0000
0x000000013e94a060 svchost.exe            1084    512     20      340      0      0 2020-10-01 16:24:43 UTC+0000
0x000000013e661b30 VGAuthService.         1308    512      5      100      0      0 2020-10-01 16:24:44 UTC+0000
0x000000013e698b30 vmtoolsd.exe           1368    512     13      274      0      0 2020-10-01 16:24:46 UTC+0000
0x000000013ff2f4f0 svchost.exe            1600    512      8       97      0      0 2020-10-01 16:24:48 UTC+0000
0x000000013e7717c0 dllhost.exe            1748    512     22      213      0      0 2020-10-01 16:24:48 UTC+0000
0x000000013e78e9e0 dllhost.exe            1920    512     17      213      0      0 2020-10-01 16:24:50 UTC+0000
0x000000013e42db30 msdtc.exe              2000    512     16      158      0      0 2020-10-01 16:24:50 UTC+0000
0x000000013fe79b30 WmiPrvSE.exe           1840    620     12      202      0      0 2020-10-01 16:24:54 UTC+0000
0x000000013e525b30 VSSVC.exe              2060    512      6      121      0      0 2020-10-01 16:24:54 UTC+0000
0x000000013faba920 WmiPrvSE.exe           2124    620     13      310      0      0 2020-10-01 16:25:08 UTC+0000
0x000000013e5d0b30 taskhost.exe           2268    512      9      167      1      0 2020-10-01 16:25:25 UTC+0000
0x000000013e21c9e0 sppsvc.exe             2396    512      4      157      0      0 2020-10-01 16:25:26 UTC+0000
0x000000013e4aa200 dwm.exe                2568    840      6      137      1      0 2020-10-01 16:25:32 UTC+0000
0x000000013e88cb30 explorer.exe           2592   2560     44      990      1      0 2020-10-01 16:25:32 UTC+0000
0x000000013e2f9060 vm3dservice.ex         2684   2592      3       45      1      0 2020-10-01 16:25:34 UTC+0000
0x000000013e268b30 vmtoolsd.exe           2696   2592      9      222      1      0 2020-10-01 16:25:34 UTC+0000
0x000000013e37ab30 SearchIndexer.         2896    512     15      629      0      0 2020-10-01 16:25:40 UTC+0000
0x000000013e3c4710 SearchProtocol         2972   2896      8      233      1      0 2020-10-01 16:25:41 UTC+0000
0x000000013e3d57c0 SearchFilterHo         2992   2896      4       86      0      0 2020-10-01 16:25:41 UTC+0000
0x000000013e0a36c0 RamCapture64.e         2836   2592      4       74      1      0 2020-10-01 16:25:54 UTC+0000
0x000000013e0d8460 conhost.exe            2840    408      3       51      1      0 2020-10-01 16:25:54 UTC+0000
0x000000013e360700 notepad.exe             788   2592      3       82      1      0 2020-10-01 16:26:04 UTC+0000
0x000000013ff75060 svchost.exe            2764    512      6       73      0      0 2020-10-01 16:26:48 UTC+0000
0x000000013e62bb30 svchost.exe            2752    512     14      342      0      0 2020-10-01 16:26:48 UTC+0000
0x000000013ecc8630 iexplore.exe           1116   2592     18      421      1      0 2020-10-01 16:26:51 UTC+0000
0x000000013ed13900 iexplore.exe           2412   1116     18      366      1      0 2020-10-01 16:26:55 UTC+0000
0x000000013e83b060 putty.exe              1936   2592      2       88      1      1 2020-10-01 16:27:00 UTC+0000
0x000000013ffde060 WmiApSrv.exe           2164    512      7      121      0      0 2020-10-01 16:27:11 UTC+0000
0x000000013e5f9b30 sdclt.exe              2176    512      1       18 ------      0 2020-10-01 16:27:43 UTC+0000
0x000000013e271b30 wsqmcons.exe           3020    512      1      257 ------      0 2020-10-01 16:27:43 UTC+0000
0x000000013ebcc240 taskhost.exe           1776    512      5  6684773 ------      0 2020-10-01 16:27:43 UTC+0000
```

PSscan

This plugin can be used to give a detailed list of processes found in the memory dump. On executing this command, the list of processes running is displayed, their respective process ID assigned to them and the parent process ID is also displayed along. The details about the threads, sessions, handles are also mentioned. The timestamp according to the start of the process is also displayed. This helps to identify whether an unknown process is running or was running at an unusual time

```
volatility -f ram.mem --profile=Win7SP1x64 psscan
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 psscan
Volatility Foundation Volatility Framework 2.6
Offset(P)          Name                 PID   PPID PDB                Time created
------------------ -------------------- ----- ----- ------------------ ------------------------------
0x000000013e0a36c0 RamCapture64.e       2836  2592 0x0000000071ec4000 2020-10-01 16:25:54 UTC+0000
0x000000013e0d8460 conhost.exe          2840   408 0x0000000071a49000 2020-10-01 16:25:54 UTC+0000
0x000000013e21c9e0 sppsvc.exe           2396   512 0x00000000893d4000 2020-10-01 16:25:26 UTC+0000
0x000000013e268b30 vmtoolsd.exe         2696  2592 0x000000007ffab000 2020-10-01 16:25:34 UTC+0000
0x000000013e271b30 wsqmcons.exe         3020   512 0x00000001297c4000 2020-10-01 16:27:43 UTC+0000
0x000000013e2f9060 vm3dservice.ex       2684  2592 0x00000000804a6000 2020-10-01 16:25:34 UTC+0000
0x000000013e360700 notepad.exe           788  2592 0x0000000072e8e000 2020-10-01 16:26:04 UTC+0000
0x000000013e37ab30 SearchIndexer.       2896   512 0x000000007d35d000 2020-10-01 16:25:40 UTC+0000
0x000000013e3c4710 SearchProtocol       2972  2896 0x0000000086f7b000 2020-10-01 16:25:41 UTC+0000
0x000000013e3d57c0 SearchFilterHo       2992  2896 0x000000007be61000 2020-10-01 16:25:41 UTC+0000
```

PStree

In this plugin, the process list is represented with a child-parent relationship and shows any unknown or abnormal processes. The child process is represented by indention and periods.

```
volatility -f ram.mem --profile=Win7SP1x64 pstree
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 pstree
Volatility Foundation Volatility Framework 2.6

Name                                                  Pid   PPid   Thds   Hnds Time
----------------------------------------------------- ----- ------ ------ ------ ----
 0xfffffa80322d2a90:wininit.exe                       416    344      3     78 2020-10
. 0xfffffa8032332780:services.exe                     512    416     11    229 2020-10
.. 0xfffffa80324a9b30:svchost.exe                     128    512     12    550 2020-10
.. 0xfffffa80325331b0:spoolsv.exe                    1040    512     14    289 2020-10
.. 0xfffffa80323e7b30:svchost.exe                     704    512      7    289 2020-10
.. 0xfffffa803282db30:msdtc.exe                      2000    512     16    158 2020-10
.. 0xfffffa8032661b30:VGAuthService.                 1308    512      5    100 2020-10
.. 0xfffffa803254a060:svchost.exe                    1084    512     20    340 2020-10
.. 0xfffffa8030f2f4f0:svchost.exe                    1600    512      8     97 2020-10
```

4. DLL

It is extremely important to know which DLLs (Dynamic Linked Libraries) are imported into the process while analysing the memory dump. A DLL can contain malicious executable code that may have a benign process to introduce malicious activity. Therefore, examining the various processes for the presence of malicious DLLs or similar code injections is crucial for analysis. Volatility has various types of plugins for this analysis.

DLLList

Various tools only have the potential to detect the DLLs which are used by a process by consulting the first of the three DLL lists stored in the PEB, which tracks the order in which each DLL is loaded. As a result, malware will sometimes modify that list to hide the presence of a DLL. Volatility has a plugin that also parses this same list, which can be run with the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 dlllist -p 116,788
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 dlllist -p 1116,788
Volatility Foundation Volatility Framework 2.6
**************************************************************************
notepad.exe pid:    788
Command line : "C:\Windows\system32\NOTEPAD.EXE" C:\Users\raj\Desktop\New Text Document.txt
Service Pack 1

Base                             Size          LoadCount LoadTime                       Path
------------------ ------------------ ------------------ ------------------------------ ----
0x00000000ffa60000            0x35000             0xffff 1970-01-01 00:00:00 UTC+0000   C:\Windows
0x0000000077490000            0x1a9000            0xffff 1970-01-01 00:00:00 UTC+0000   C:\Windows
0x0000000077370000            0x11f000            0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefd4d0000            0x6b000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefea40000            0xdb000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefeec0000            0x9f000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007feff580000            0x1f000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefeb20000            0x12d000            0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefe7b0000            0x67000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x0000000077270000            0xfa000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007feff570000             0xe000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007feff6d0000            0xc9000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefe820000            0x97000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007feff4d0000            0x71000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefbbf0000            0x1f4000            0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefd850000            0xd88000            0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fef8050000            0x71000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefecb0000            0x203000            0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007feff190000            0xd7000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefc540000             0xc000             0xffff 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefe5e0000            0x2e000                0x4 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefe690000            0x109000               0x2 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefd2d0000             0xf000                0x1 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefb970000            0x56000                0x3 2020-10-01 16:26:04 UTC+0000   C:\Windows
0x000007fefb080000            0x18000                0x1 2020-10-01 16:26:04 UTC+0000   C:\Windows
**************************************************************************
iexplore.exe pid:   1116
Command line : "C:\Program Files\Internet Explorer\iexplore.exe"
Service Pack 1

Base                             Size          LoadCount LoadTime                       Path
------------------ ------------------ ------------------ ------------------------------ ----
0x0000000000210000            0xac000             0xffff 1970-01-01 00:00:00 UTC+0000   C:\Program
0x0000000077490000            0x1a9000            0xffff 1970-01-01 00:00:00 UTC+0000   C:\Windows
```

DLLDump

This plugin is used to dump the DLLs from the memory space of the processes into another location to analyze it. To take a dump of the DLLs you can type,

```
volatility -f ram.mem --profile=Win7SP1x64 dlldump -dump-dir
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 dlldump --dump-dir /root/ramdump/
Volatility Foundation Volatility Framework 2.6
Process(V)         Name                 Module Base        Module Name          Result
------------------ -------------------- ------------------ -------------------- ------
0xfffffa80318a02f0 smss.exe             0x0000000047850000 smss.exe             OK: module.268.13f4a
0xfffffa80318a02f0 smss.exe             0x0000000077490000 ntdll.dll            OK: module.268.13f4a
0xfffffa8032104060 csrss.exe            0x000000004a520000 csrss.exe            OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x0000000077490000 ntdll.dll            OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefd440000 basesrv.DLL          OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007feff6d0000 USP10.dll            OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x0000000077270000 USER32.dll           OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefd460000 CSRSRV.dll           OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x0000000077370000 kernel32.dll         OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefeb20000 RPCRT4.dll           OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefd2d0000 CRYPTBASE.dll        OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefe7b0000 GDI32.dll            OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefeec0000 msvcrt.dll           OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007feff570000 LPK.dll              OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefd4d0000 KERNELBASE.dll       OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefd2e0000 sxs.dll              OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefd3f0000 sxssrv.DLL           OK: module.352.13ed0
0xfffffa8032104060 csrss.exe            0x000007fefd400000 winsrv.DLL           OK: module.352.13ed0
0xfffffa80322d82f0 csrss.exe            0x000000004a520000 csrss.exe            OK: module.408.13ead
```

5. Handles

This plugin is used to display the open handles that are present in a process. This plugin applies to files, registry keys, events, desktops, threads, and all other types of objects. To see the handles, present in the dump, you can type,

```
volatility -f ram.mem --profile=Win7SP1x64 handles
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 handles
Volatility Foundation Volatility Framework 2.6
Offset(V)          Pid    Handle             Access Type            Details
------------------ ------ ------------------ ------ ---------------- -------
0xfffffa8030ece890      4                0x4 0x1fffff Process         System(4)
0xfffff8a0000711f0      4                0x8  0x2001f Key            MACHINE\CONTROLSET
0xfffff8a000008060      4                0xc  0xf000f Directory       GLOBAL??
0xfffff8a00001aca0      4               0x10      0x0 Key
0xfffff8a00008ed30      4               0x14  0x2001f Key            MACHINE\CONTROLSET
0xfffff8a000072fa0      4               0x18  0xf003f Key            MACHINE\CONTROLSET
0xfffff8a00008ee20      4               0x1c  0x2001f Key            MACHINE\SETUP
0xfffffa8030efea40      4               0x20 0x1f0001 ALPC Port      PowerMonitorPort
0xfffffa8030f0a070      4               0x24 0x1f0001 ALPC Port      PowerPort
0xfffff8a000072ba0      4               0x28  0x20019 Key            MACHINE\DESCRIPTI
0xfffffa8030ff57e0      4               0x2c 0x1fffff Thread         TID 172 PID 4
0xfffff8a00008fa90      4               0x30  0xf003f Key            MACHINE\CONTROLSET
0xfffff8a00008be80      4               0x34  0xf003f Key            MACHINE\CONTROLSET
0xfffff8a000057fa0      4               0x38  0xf003f Key            MACHINE\CONTROLSET
```

6. Getsids

This plugin is used to view the SIDs stands for Security Identifiers that are associated with a process. This plugin can help in identifying processes that have maliciously escalated privileges and which processes belong to specific users. To get detail on a particular process id, you can type:

```
volatility -f ram.mem --profile=Win7SP1x64 getsids -p 464
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 getsids -p 464
Volatility Foundation Volatility Framework 2.6
winlogon.exe (464): S-1-5-18 (Local System)
winlogon.exe (464): S-1-5-32-544 (Administrators)
winlogon.exe (464): S-1-1-0 (Everyone)
winlogon.exe (464): S-1-5-11 (Authenticated Users)
winlogon.exe (464): S-1-16-16384 (System Mandatory Level)
root@kali:~#
```

7. Netscan

This plugin helps in finding network-related artifacts present in the memory dump. It makes use of pool tag scanning. This plugin finds all the TCP endpoints, TCP listeners, UDP endpoints, and UDP listeners. It provides details about the local and remote IP and also about the local and remote port. To get details on the network artifacts, you can type:

```
volatility -f ram.mem --profile=Win7SP1x64 netscan
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 netscan
Volatility Foundation Volatility Framework 2.6
Offset(P)          Proto    Local Address                  Foreign Address      State
0x13e0de9e0        UDPv4    127.0.0.1:65024                *:*
0x13e8dcce0        UDPv4    0.0.0.0:0                      *:*
0x13e8dcce0        UDPv6    :::0                           *:*
0x13e8e4ad0        UDPv4    0.0.0.0:5355                   *:*
0x13e9c2d60        UDPv4    0.0.0.0:4500                   *:*
0x13e9c2d60        UDPv6    :::4500                        *:*
0x13e9d9270        UDPv4    0.0.0.0:4500                   *:*
0x13e9d9930        UDPv4    0.0.0.0:500                    *:*
0x13e9de010        UDPv4    0.0.0.0:500                    *:*
0x13e9de010        UDPv6    :::500                         *:*
0x13e9de500        UDPv4    0.0.0.0:0                      *:*
0x13e9de500        UDPv6    :::0                           *:*
0x13e9deb10        UDPv4    0.0.0.0:0                      *:*
0x13eaed860        UDPv4    192.168.2.11:138               *:*
0x13eb35920        UDPv4    192.168.2.11:137               *:*
0x13e6fb790        TCPv4    0.0.0.0:49155                  0.0.0.0:0            LISTENING
0x13e6fbef0        TCPv4    0.0.0.0:445                    0.0.0.0:0            LISTENING
```

8. Hivelist

This plugin can be used to locate the virtual addresses present in the registry hives in memory, and their entire paths to hive on the disk. To obtain the details on the hivelist from the memory dump, you can type:

```
volatility -f ram.mem --profile=Win7SP1x64 hivelist
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 hivelist
Volatility Foundation Volatility Framework 2.6
Virtual            Physical           Name
------------------ ------------------ ----
0xfffff8a00000f010 0x00000000a97f2010 [no name]
0xfffff8a000024010 0x00000000a987d010 \REGISTRY\MACHINE\SYSTEM
0xfffff8a000057010 0x00000000a95b0010 \REGISTRY\MACHINE\HARDWARE
0xfffff8a00058a010 0x00000000a8270010 \SystemRoot\System32\Config\SECURITY
0xfffff8a00058c010 0x00000000a83f2010 \SystemRoot\System32\Config\SOFTWARE
0xfffff8a00058f010 0x000000009d700010 \SystemRoot\System32\Config\DEFAULT
0xfffff8a0005ff010 0x00000000a8182010 \SystemRoot\System32\Config\SAM
0xfffff8a000e4d010 0x000000009d4e5010 \??\C:\Windows\ServiceProfiles\NetworkService\N
0xfffff8a000eef010 0x000000009d536010 \??\C:\Windows\ServiceProfiles\LocalService\NTU
0xfffff8a0015d7010 0x000000008a545010 \??\C:\Users\raj\ntuser.dat
0xfffff8a0015e5010 0x000000008aa5c010 \??\C:\Users\raj\AppData\Local\Microsoft\Window
0xfffff8a0021c8010 0x00000000610c4010 \??\C:\System Volume Information\Syscache.hve
0xfffff8a00307c010 0x00000000a58f7010 \Device\HarddiskVolume1\Boot\BCD
```

9. Timeliner

This plugin usually creates a timeline from the various artifacts found in the memory dump. To locate the artifacts according to the timeline, you can use the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 timeliner
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 timeliner
Volatility Foundation Volatility Framework 2.6
2020-10-01 16:27:05 UTC+0000|[LIVE RESPONSE]| (System time)|
2020-10-01 16:26:04 UTC+0000|[IEHISTORY]| explorer.exe→Visited: raj@file:///C:/Users/raj/De
2020-09-26 11:42:11 UTC+0000|[IEHISTORY]| explorer.exe→Visited: raj@file:///C:/Users/raj/De
2020-09-17 17:43:58 UTC+0000|[IEHISTORY]| explorer.exe→Visited: raj@file:///E:/raj.txt| PID
2020-09-26 11:48:11 UTC+0000|[IEHISTORY]| explorer.exe→Visited: raj@file:///C:/Users/raj/De
2020-10-01 21:56:04 UTC+0000|[IEHISTORY]| explorer.exe->:2020100120201002: raj@file:///C:/Us
2020-10-01 21:56:04 UTC+0000|[IEHISTORY]| explorer.exe->:2020100120201002: raj@:Host: Comput
2020-10-01 16:26:04 UTC+0000|[IEHISTORY]| iexplore.exe→Visited: raj@file:///C:/Users/raj/De
2020-09-26 11:42:11 UTC+0000|[IEHISTORY]| iexplore.exe→Visited: raj@file:///C:/Users/raj/De
2020-09-17 17:43:58 UTC+0000|[IEHISTORY]| iexplore.exe→Visited: raj@file:///E:/raj.txt| PID
2020-09-26 11:48:11 UTC+0000|[IEHISTORY]| iexplore.exe→Visited: raj@file:///C:/Users/raj/De
```

10. HashDump

This plugin can be used to extract and decrypt cached domain credentials stored in the registry which can be availed from the memory dump. The hashes that are availed from the memory dump can be cracked using John the Ripper, Hashcat, etc. To gather the hashdump, you can use the command:

```
volatility -f ram.mem --profile=Win7SP1x64 hashdump
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 hashdump
Volatility Foundation Volatility Framework 2.6
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
raj:1000:aad3b435b51404eeaad3b435b51404ee:3dbde697d71690a769204beb12283678:::
ignite:1001:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

11. Lsadump

This plugin is used to dump LSA secrets from the registry in the memory dump. This plugin gives out information like the default password, the RDP public key, etc. To perform a lsadump, you can type the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 lsadump
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 lsadump
Volatility Foundation Volatility Framework 2.6
DefaultPassword
0x00000000  08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00000010  31 00 32 00 33 00 34 00 00 00 00 00 00 00 00 00   1.2.3.4.........

DPAPI_SYSTEM
0x00000000  2c 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ,...............
```

12. Modscan

This plugin is used to locate kernel memory and its related objects. It can pick up all the previously unloaded drivers and also those drivers that have been hidden or have been unlinked by rootkits in the system.

```
volatility -f ram.mem --profile=Win7SP1x64 modscan
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 modscan
Volatility Foundation Volatility Framework 2.6
Offset(P)          Name                 Base                             Size File
------------------ -------------------- ------------------ ------------------ ----
0x0000000002fa45e1                      0x894c304b8b48ffad         0x8435e800
0x000000005bdeb5e1                      0x894c304b8b48ffad         0x8435e800
0x000000013e230c00 spsys.sys            0xfffff88005a00000            0x71000 \SystemRoot
0x000000013e2be010 RamCaptur ... er64.SYS 0xfffff88005a71000             0x7000 \??\C:\User
0x000000013e611350 secdrv.SYS           0xfffff88005927000             0xb000 \SystemRoot
0x000000013e6171b0 srvnet.sys           0xfffff88005932000            0x31000 \SystemRoot
0x000000013e629520 rdpdr.sys            0xfffff88005b7d000            0x2e000 \SystemRoot
0x000000013e634480 srv2.sys             0xfffff88005975000            0x6b000 \SystemRoot
```

13. FileScan

This plugin is used to find FILE_OBJECTs present in the physical memory by using pool tag scanning. It can find open files even if there is a hidden rootkit present in the files. To make use of this plugin, you can type the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 filescan
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 filescan
Volatility Foundation Volatility Framework 2.6
Offset(P)          #Ptr   #Hnd Access Name
------------------ ------ ------ ------ ----
0x000000013e000910      1      1 RW-rw- \Device\HarddiskVolume1\Users\raj\AppData\Local\Microso
0x000000013e00c4a0      2      1 ------ \Device\NamedPipe\MsFteWds
0x000000013e00c740      2      0 R--r-d \Device\HarddiskVolume1\Windows\System32\rasdlg.dll
0x000000013e00c9f0      1      0 RW-rwd \Device\HarddiskVolume1\$PrepareToShrinkFileSize
0x000000013e01baf0     12      0 R--r-d \Device\HarddiskVolume1\Windows\System32\wlanutil.dll
0x000000013e01d6e0      1      1 R--rw- \Device\HarddiskVolume1\Windows\winsxs\amd64_microsoft.
0x000000013e020560     14      0 R--r-- \Device\HarddiskVolume1\wkssvc
0x000000013e020920      4      0 R--r-d \Device\HarddiskVolume1\Windows\System32\WWanAPI.dll
0x000000013e021a70     18      1 RW-r-- \Device\HarddiskVolume1\Windows\System32\winevt\Logs\Mi
0x000000013e021dd0     12      0 R--r-d \Device\HarddiskVolume1\Windows\System32\wwapi.dll
0x000000013e021f20      5      0 R--r-d \Device\HarddiskVolume1\Windows\System32\bthprops.cpl
```

14. Svcscan

This plugin is used to see the services are registered on your memory image, use the svcscan command. The output shows the process ID of each service the service name, service name, display name, service type, service state, and also shows the binary path for the registered service – which will be a .exe for user mode services and a driver name for services that run from kernel mode. To find the details on the services

```
volatility -f ram.mem --profile=Win7SP1x64 svcscan
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 svcscan
```

```
Offset: 0xcc8500
Order: 70
Start: SERVICE_AUTO_START
Process ID: 800
Service Name: Dhcp
Display Name: DHCP Client
Service Type: SERVICE_WIN32_SHARE_PROCESS
Service State: SERVICE_RUNNING
Binary Path: C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestri
```

15. Cmdscan

This plugin searches the memory dump of XP/2003/Vista/2008 and Windows 7 for commands that the attacker might have entered through a command prompt (cmd.exe). It is one of the most powerful commands that one can use to gain visibility into an attacker's actions on a victim system. To conduct a cmdscan, you can make use of the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 cmdscan
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 cmdscan
Volatility Foundation Volatility Framework 2.6
**************************************************
CommandProcess: conhost.exe Pid: 2840
CommandHistory: 0x1e8ce0 Application: RamCapture64.exe Flags: Allocated
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x64
Cmd #15 @ 0x180158:
Cmd #16 @ 0x1e7e50:
root@kali:~#
```

16. Iehistory

This plugin recovers the fragments of Internet Explorer history by finding index.dat cache file. To find iehistory files, you can type the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 iehistory
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 iehistory
Volatility Foundation Volatility Framework 2.6
**************************************************
Process: 2592 explorer.exe
Cache type "URL " at 0x2955100
Record length: 0x100
Location: Visited: raj@file:///C:/Users/raj/Desktop/New%20Text%20Document.txt
Last modified: 2020-10-01 16:26:04 UTC+0000
Last accessed: 2020-10-01 16:26:04 UTC+0000
File Offset: 0x100, Data Offset: 0x0, Data Length: 0xac
**************************************************
Process: 2592 explorer.exe
Cache type "URL " at 0x2955200
Record length: 0x100
Location: Visited: raj@file:///C:/Users/raj/Desktop/Confidential.txt
Last modified: 2020-09-26 11:42:11 UTC+0000
Last accessed: 2020-09-26 11:42:11 UTC+0000
File Offset: 0x100, Data Offset: 0x0, Data Length: 0xa4
```

17. Dumpregistry

This plugin allows one to dump a registry hive into a disk location. To dump the registry hive, you use the following command.

```
volatility -f ram.mem --profile=Win7SP1x64 dumpregistry --dump-dir /root/ramdump/
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 dumpregistry --dump-dir /root/ramdump/
Volatility Foundation Volatility Framework 2.6
**************************************************
Writing out registry: registry.0xfffff8a000024010.SYSTEM.reg

**************************************************
**************************************************
Writing out registry: registry.0xfffff8a0015d7010.ntuserdat.reg

**************************************************
**************************************************
Writing out registry: registry.0xfffff8a000eef010.NTUSERDAT.reg

**************************************************
**************************************************
Writing out registry: registry.0xfffff8a00058f010.DEFAULT.reg

Physical layer returned None for index 23000, filling with NULL
**************************************************
**************************************************
Writing out registry: registry.0xfffff8a00058a010.SECURITY.reg

**************************************************
**************************************************
Writing out registry: registry.0xfffff8a0005ff010.SAM.reg
```

18. Moddump

This plugin is used to extract a kernel driver to a file, you can do this by using the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 moddump --dump-dir /root/ramdump/
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 moddump --dump-dir /root/ramdump/
Volatility Foundation Volatility Framework 2.6
Module Base        Module Name          Result
------------------ -------------------- ------
0xfffff80002a0b000 ntoskrnl.exe         OK: driver.fffff80002a0b000.sys
0xfffff80002ff5000 hal.dll              OK: driver.fffff80002ff5000.sys
0xfffff880017d9000 VIDEOPRT.SYS         OK: driver.fffff880017d9000.sys
0xfffff88004a8e000 ksthunk.sys          OK: driver.fffff88004a8e000.sys
0xfffff88004000000 dfsc.sys             OK: driver.fffff88004000000.sys
0xfffff88001aba000 vmstorfl.sys         OK: driver.fffff88001aba000.sys
0xfffff88004570000 rdpbus.sys           OK: driver.fffff88004570000.sys
0xfffff8800169b000 rdpencdd.sys         OK: driver.fffff8800169b000.sys
0xfffff88004adc000 usbccgp.sys          OK: driver.fffff88004adc000.sys
0xfffff88000eee000 WDFLDR.SYS           OK: driver.fffff88000eee000.sys
0xfffff88000f5d000 msisadrv.sys         OK: driver.fffff88000f5d000.sys
```

19. Procdump

This plugin is used to dump the executable processes in a single location, If there is malware present it will intentionally forge size fields in the PE header for the memory dumping tool to fail. To collect the dump on processes, you can type:

```
volatility -f ram.mem --profile=Win7SP1x64 procdump --dump-dir /root/ramdump/
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 procdump --dump-dir /root/ramdump/
Volatility Foundation Volatility Framework 2.6
Process(V)         ImageBase          Name                 Result
------------------ ------------------ -------------------- ------
0xfffffa8030ece890 ------------------ System               Error: PEB at 0x0 is unavailable (possib
0xfffffa80318a02f0 0x0000000047850000 smss.exe             OK: executable.268.exe
0xfffffa8032104060 0x000000004a520000 csrss.exe            OK: executable.352.exe
0xfffffa80322d82f0 0x000000004a520000 csrss.exe            OK: executable.408.exe
0xfffffa80322d2a90 0x00000000ffbc0000 wininit.exe          OK: executable.416.exe
0xfffffa8032312060 0x00000000ffbe0000 winlogon.exe         OK: executable.464.exe
0xfffffa8032332780 0x00000000ff4e0000 services.exe         OK: executable.512.exe
0xfffffa8032368450 0x00000000ff310000 lsass.exe            OK: executable.520.exe
```

20. Memdump

The memdump plugin is used to dump the memory-resident pages of a process into a separate file. You can also lookup a particular process using -p and provide it with a directory path -D to generate the output. To take a dump on memory-resident pages, you can use the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 memdump --dump-dir /root/ramdump/
```

```
root@kali:~# volatility -f ram.mem --profile=Win7SP1x64 memdump --dump-dir /root/ramdump/
Volatility Foundation Volatility Framework 2.6
************************************************************************
Writing System [     4] to 4.dmp
************************************************************************
Writing smss.exe [   268] to 268.dmp
************************************************************************
Writing csrss.exe [   352] to 352.dmp
```

21. Notepad

Notepad files are usually highly looked up files in the ram dump. To find the contents present in the notepad file, you can use the following command:

```
volatility -f ram.mem --profile=Win7SP1x64 notepad
```

```
root@kali:~# volatility -f ram.mem --profile=WinXPSP2x86 notepad
Volatility Foundation Volatility Framework 2.6
Process: 628
Text:
Thcgpune

Process: 1804
```

PassMark Volatility Workbench

Volatility Workbench is a GUI version of one of the most popular tool Volatility for analyzing the artifacts from a memory dump. It is available free of cost, open-source, and runs on the Windows Operating system. You can download it from **Here**.

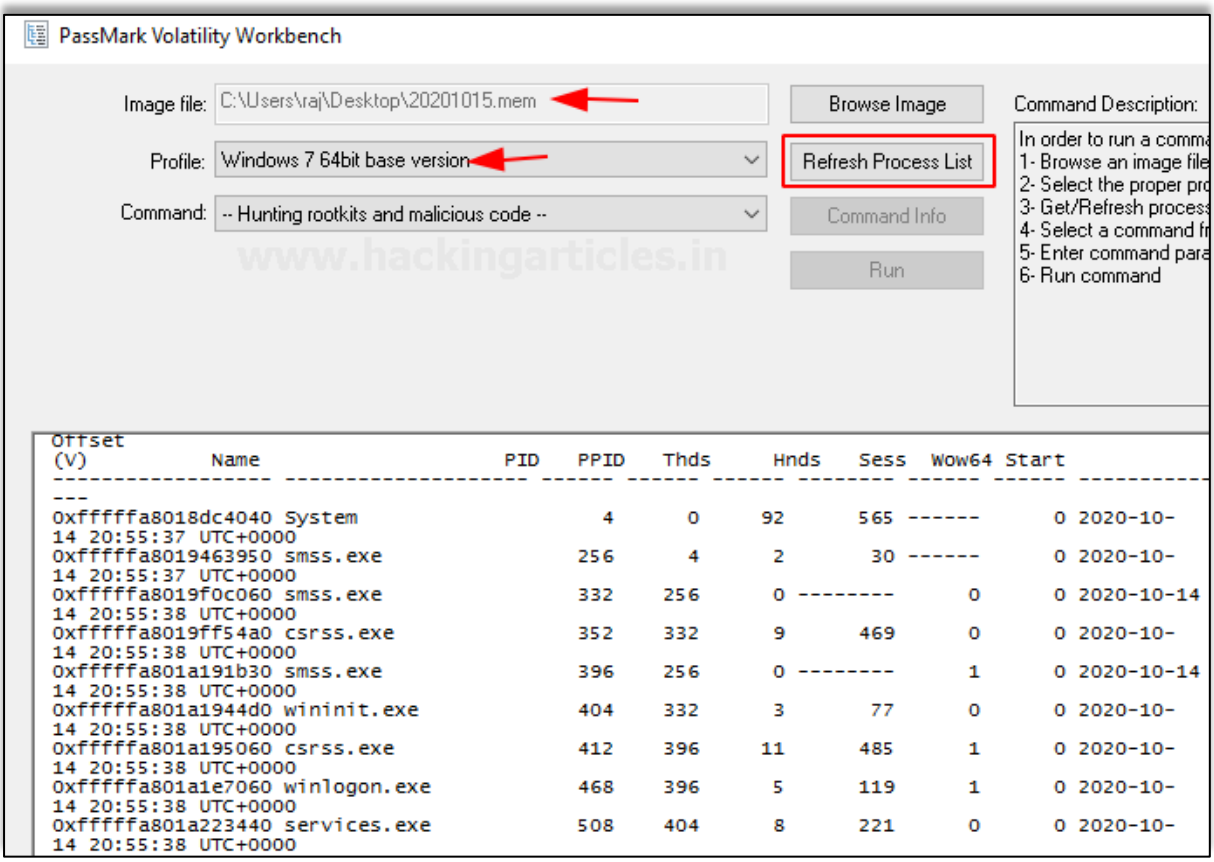
Features of Volatility Workbench

1. A forensic investigator does not have to worry about remembering the parameters of the command line.
2. It has made it easier to store dump information to a file on disk.
3. There is a drop-down list that contains the commands and its brief description.
4. It records the time stamp of the commands that were previously executed.

Download the tool and run it. Now choose the dump file that you have previously created and select the profile of the image that was created which could be used in place of imageinfo command. Now click on Refresh Process List and you can run all the commands.

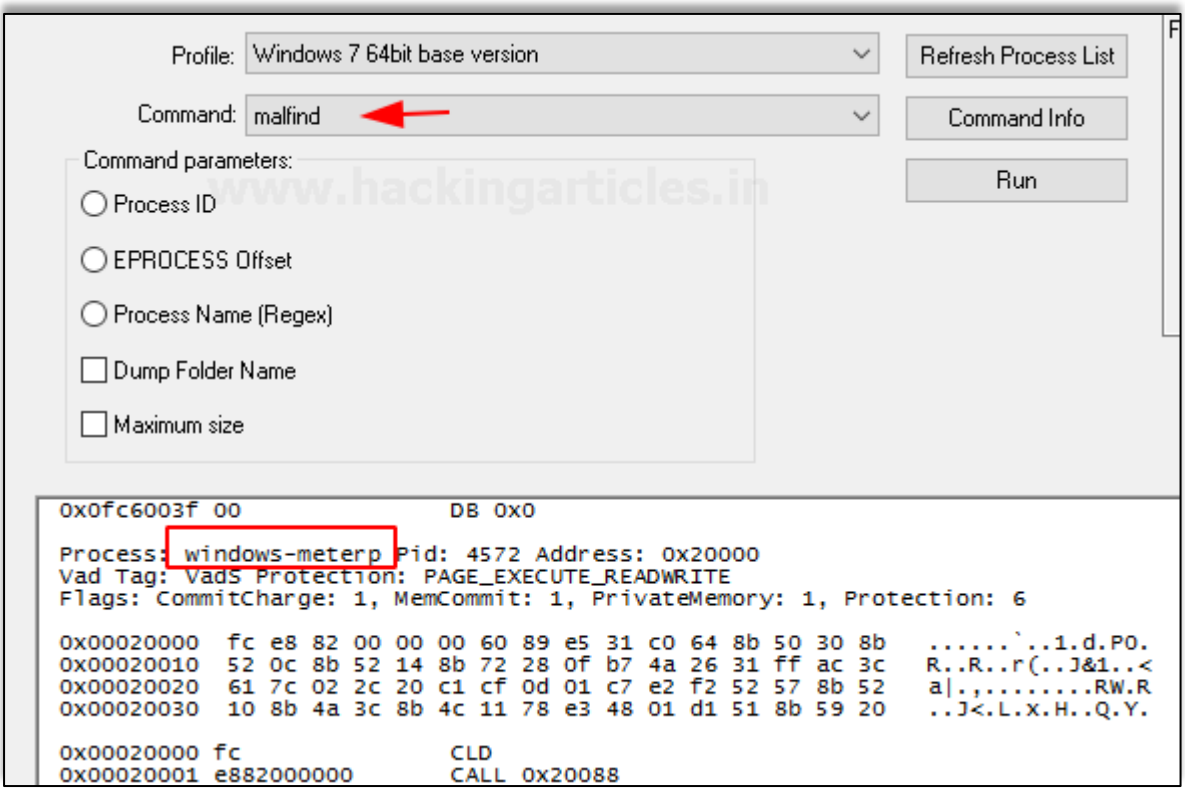
1. Hunting rootkits and malicious code

It tends to run a scan on the memory dump and looks around for the presence of a rootkit or a malicious code that would not be easily seen in the system but could be running in the background.



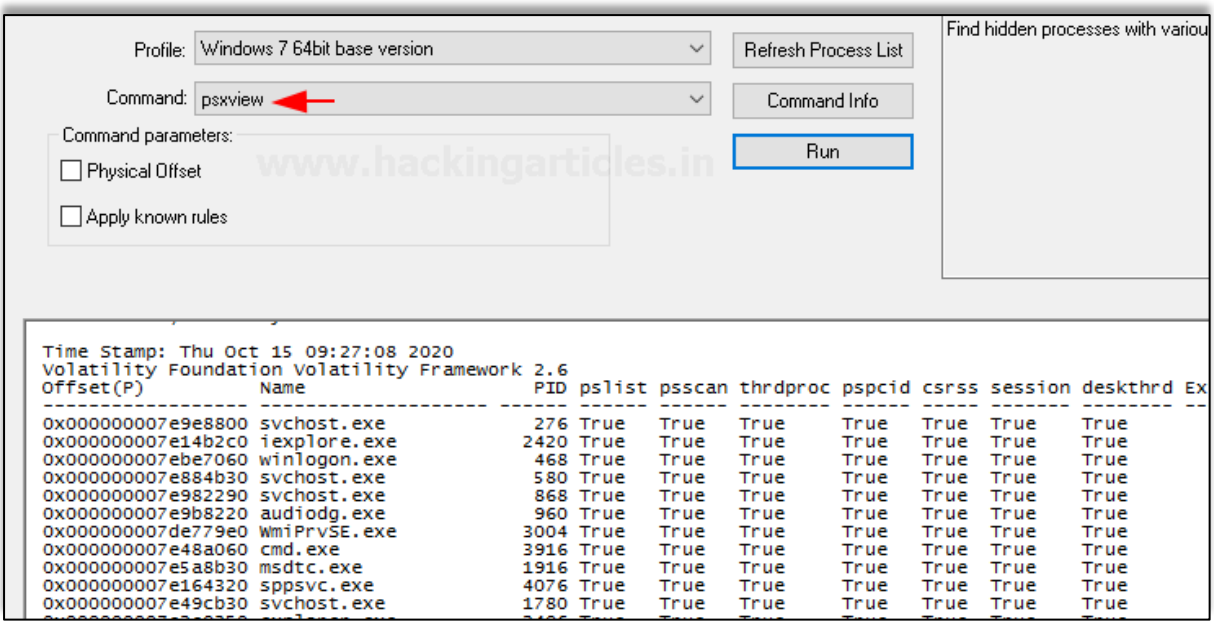
2. Malfind

It is a command which helps in finding a hidden code or a code that has been injected into the user's memory. It doesn't generally detect the presence of a DLL in a process but instead locates them.



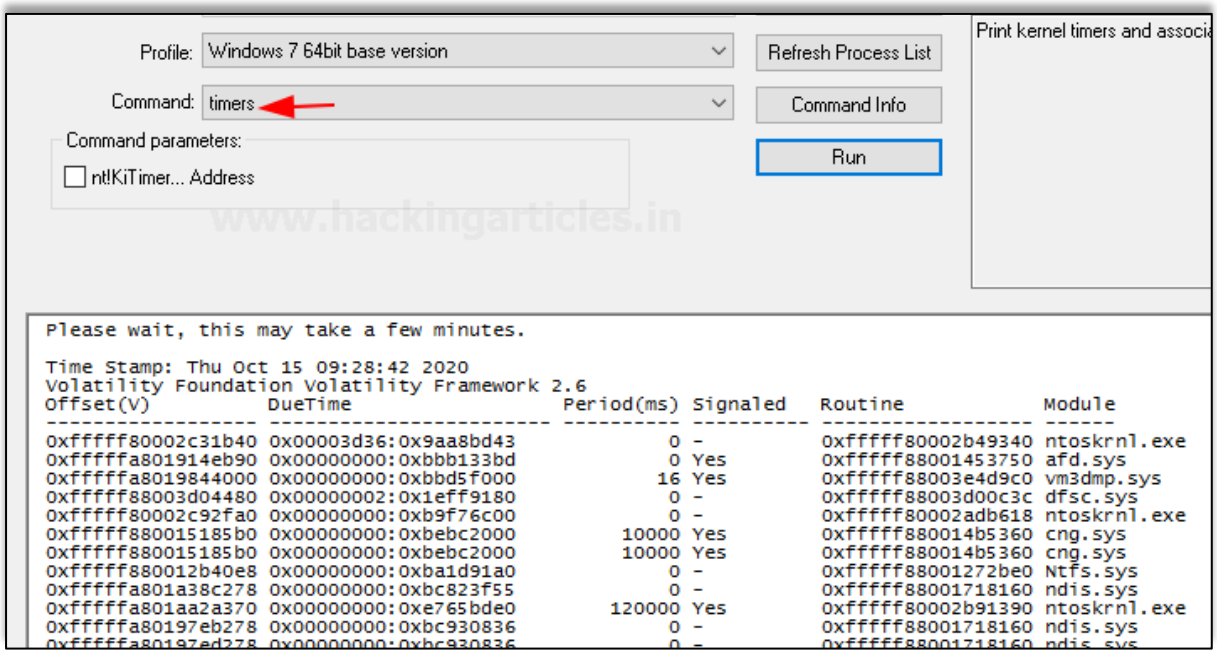
3. psxview

This command usually helps in discovering any hidden processes in the plugin present in the memory dump.



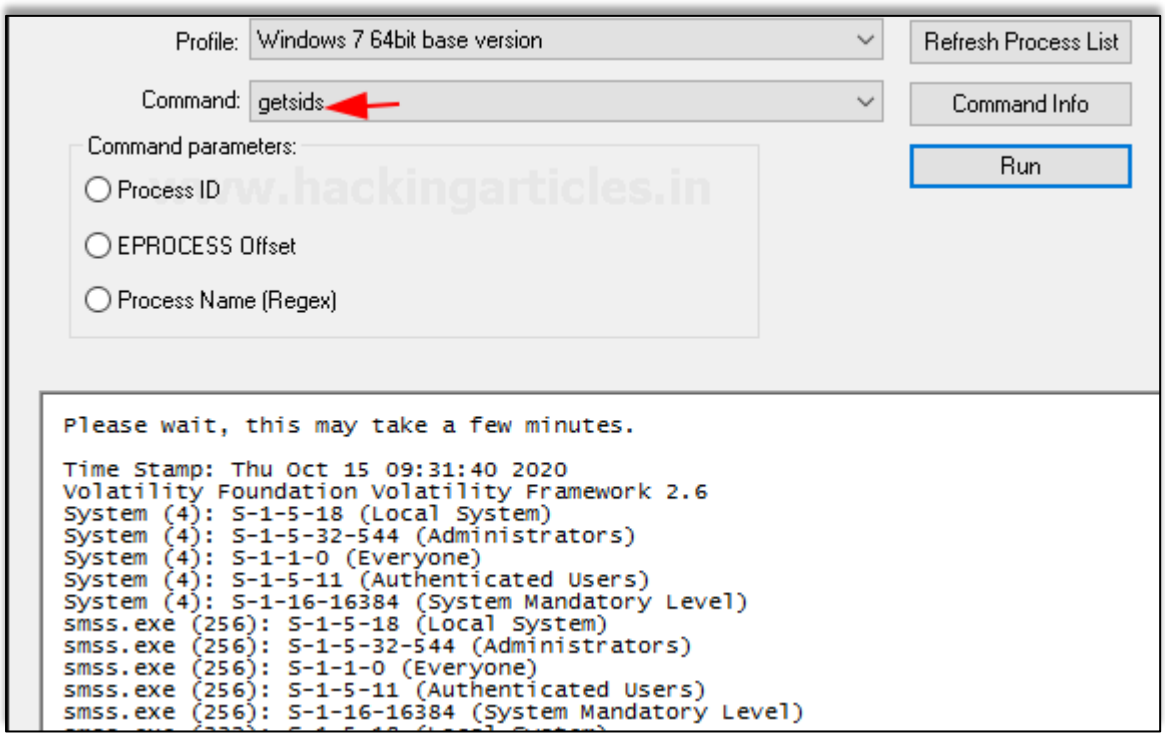
4. Timers

It displays the timer of the kernel and all the associated timers present in the memory dump of the system.



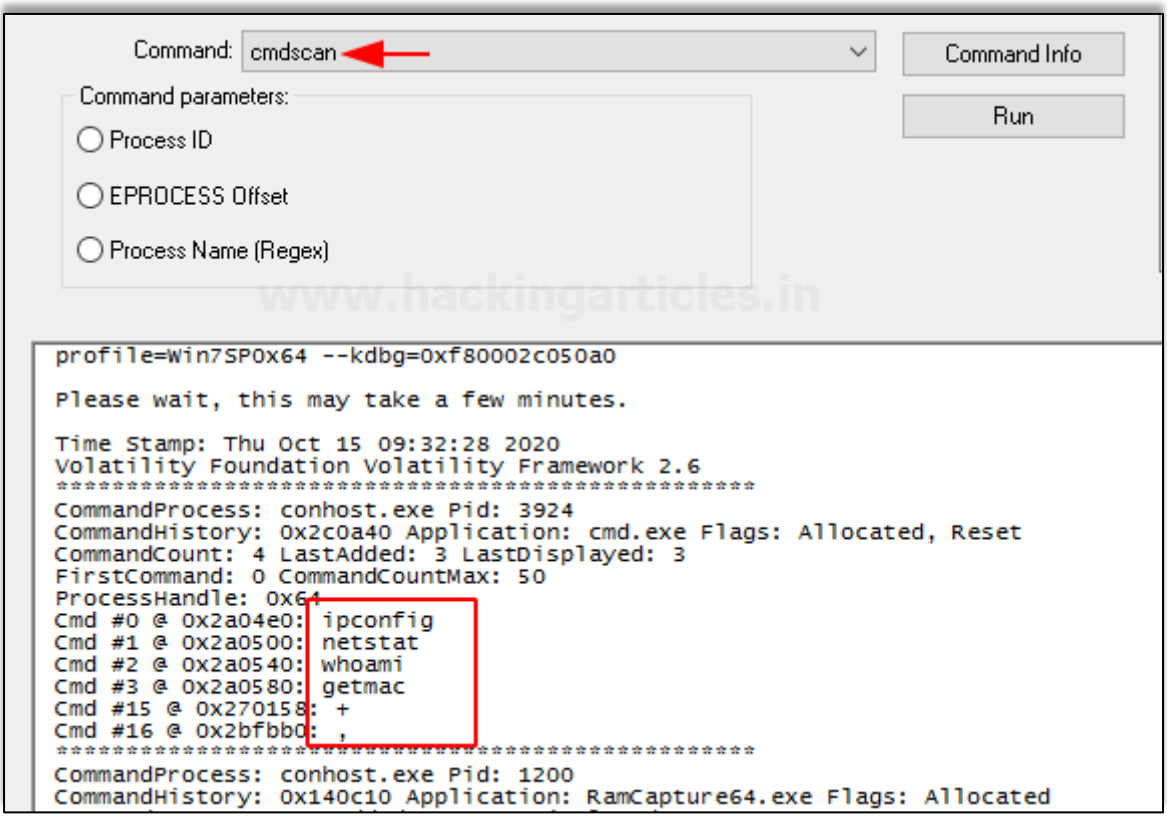
5. Getsids

This command can be used to view the Security Identifiers that are associated with a particular process. With the help of this command, you can identify if any malicious process has taken any privilege escalation.



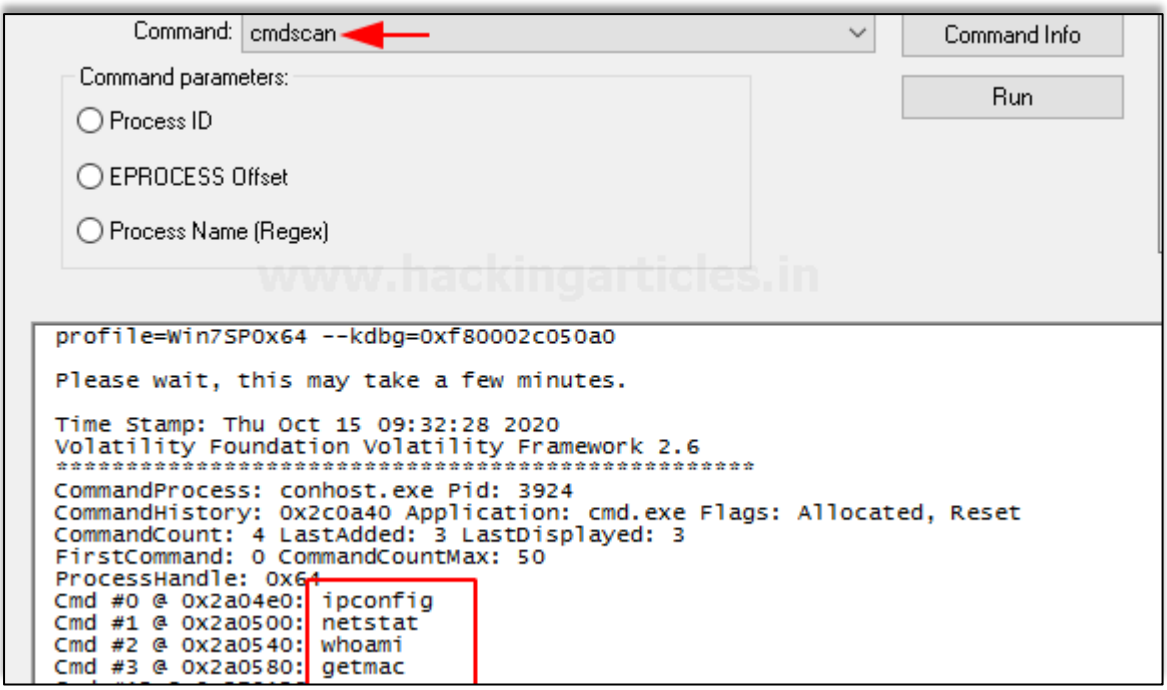
6. Cmdscan

This plugin helps in searching the memory dump for the command the user must have used the cmd.exe application. This command is highly used if the attacker's command activity is to be traced.



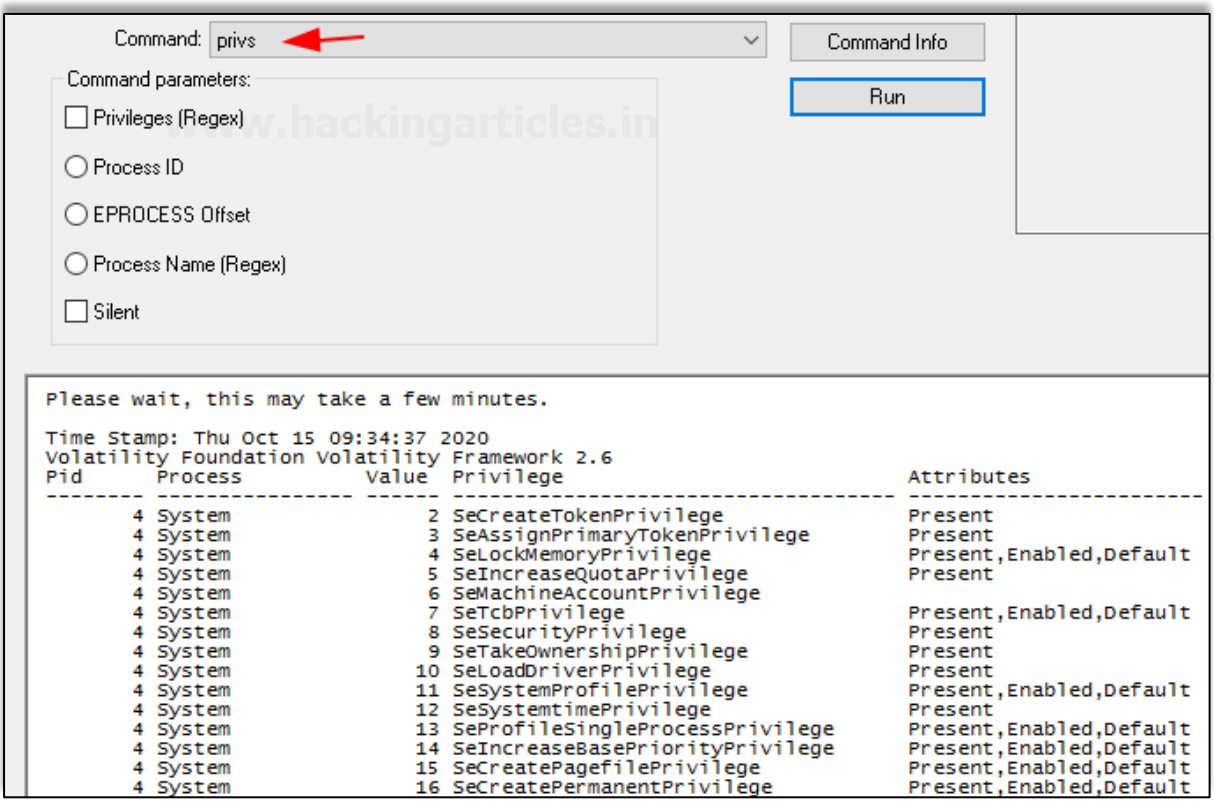
7. Consoles

This command is similar to cmdscan and helps to find if the attacker had typed anything in cmd or had executed anything via the backdoor.



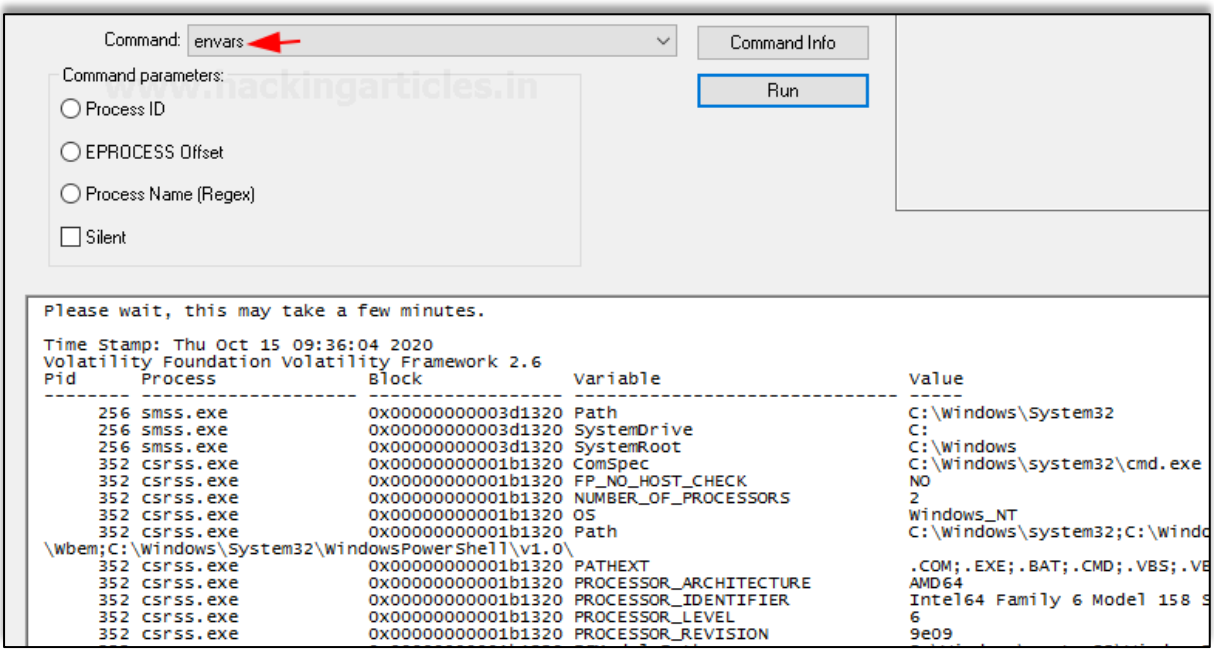
8. Privs

This command displays the privileges assigned to the processes that are enabled or not enabled by default.



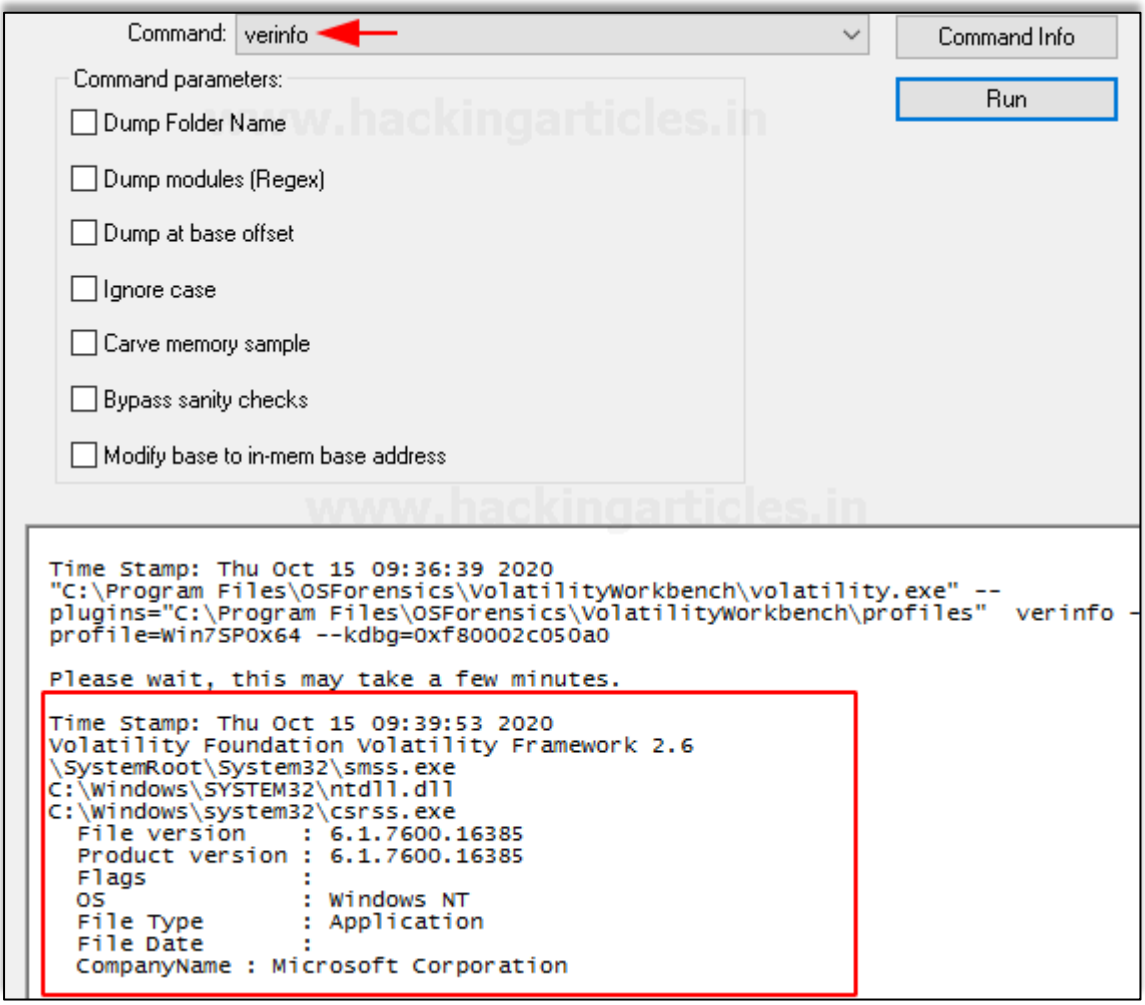
9. Envars

This command displays all the variables in the process, its environment along with its current directory.



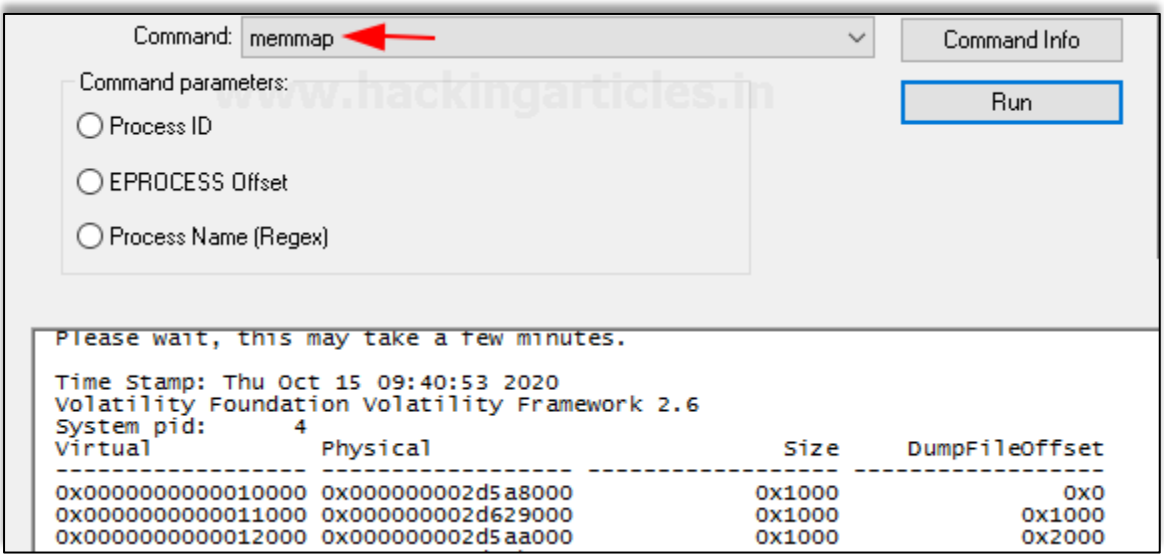
10. Verinfo

This command displays the version information that is present in the PE files. It helps identify any binaries and also correlates with other files.



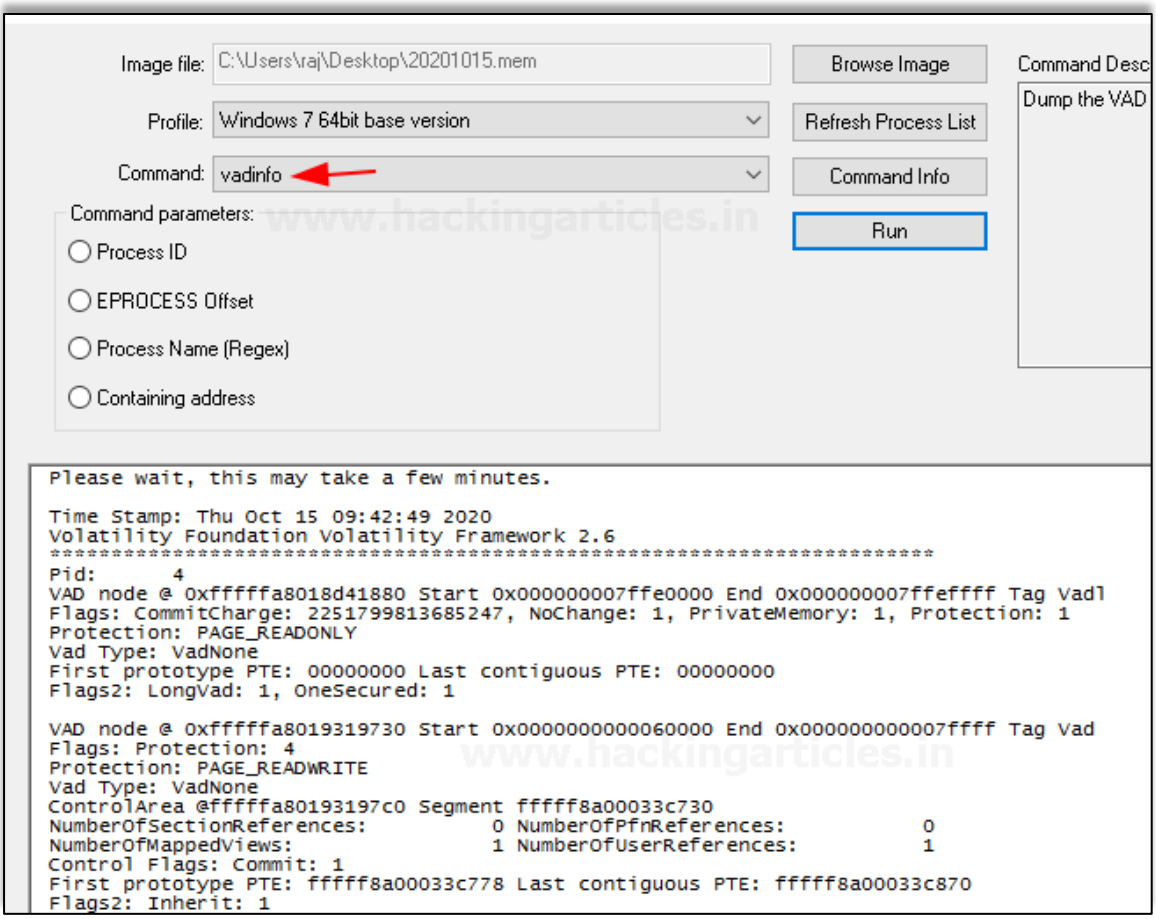
11. Memmap

This command shows the exact pages that are present on the page of a specific process. It also shows the virtual address of the page and the size of its page.



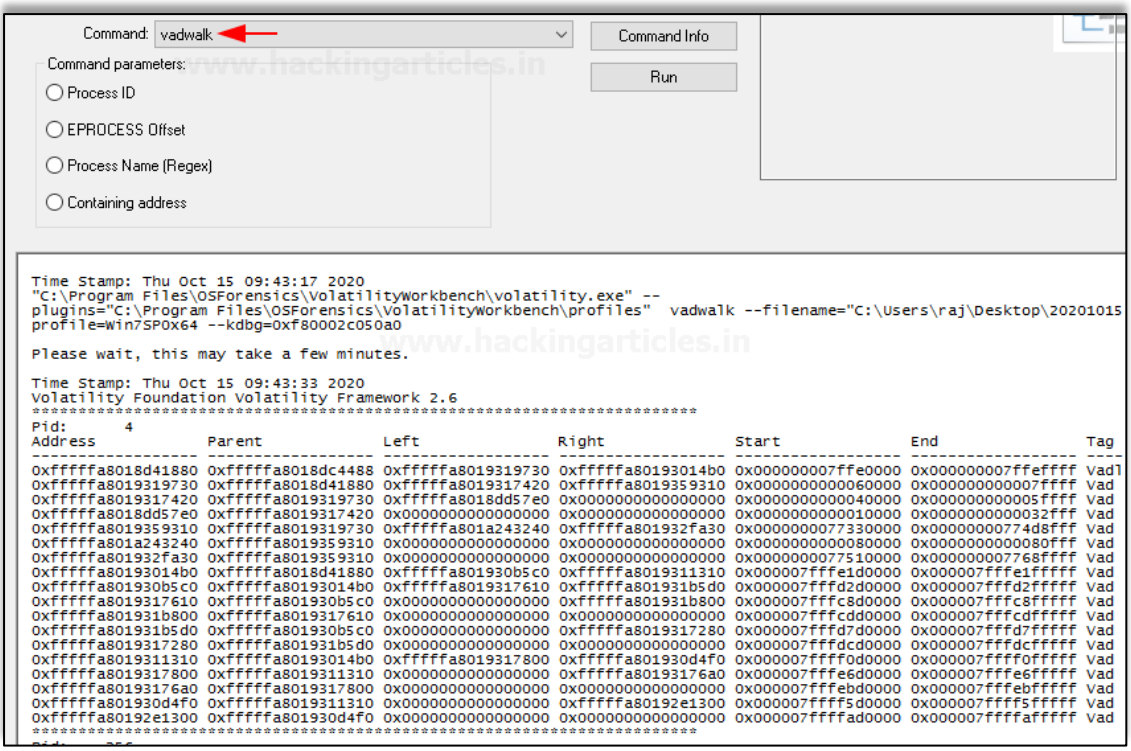
12. Vadinfo

This command usually displays information about a particular process's VAD nodes. It displays the VAD Flags control flags, VAD tags.



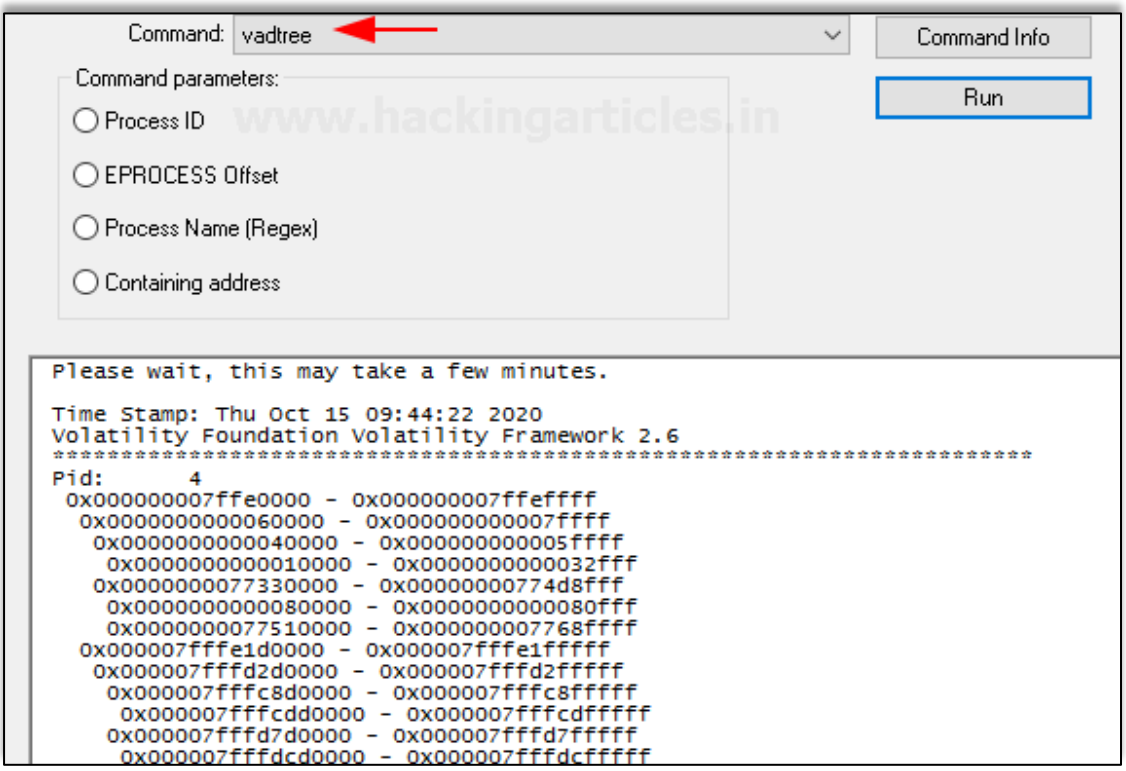
13. Vadwalk

It is a command that is used to display all the VAD nodes in a tabular form.



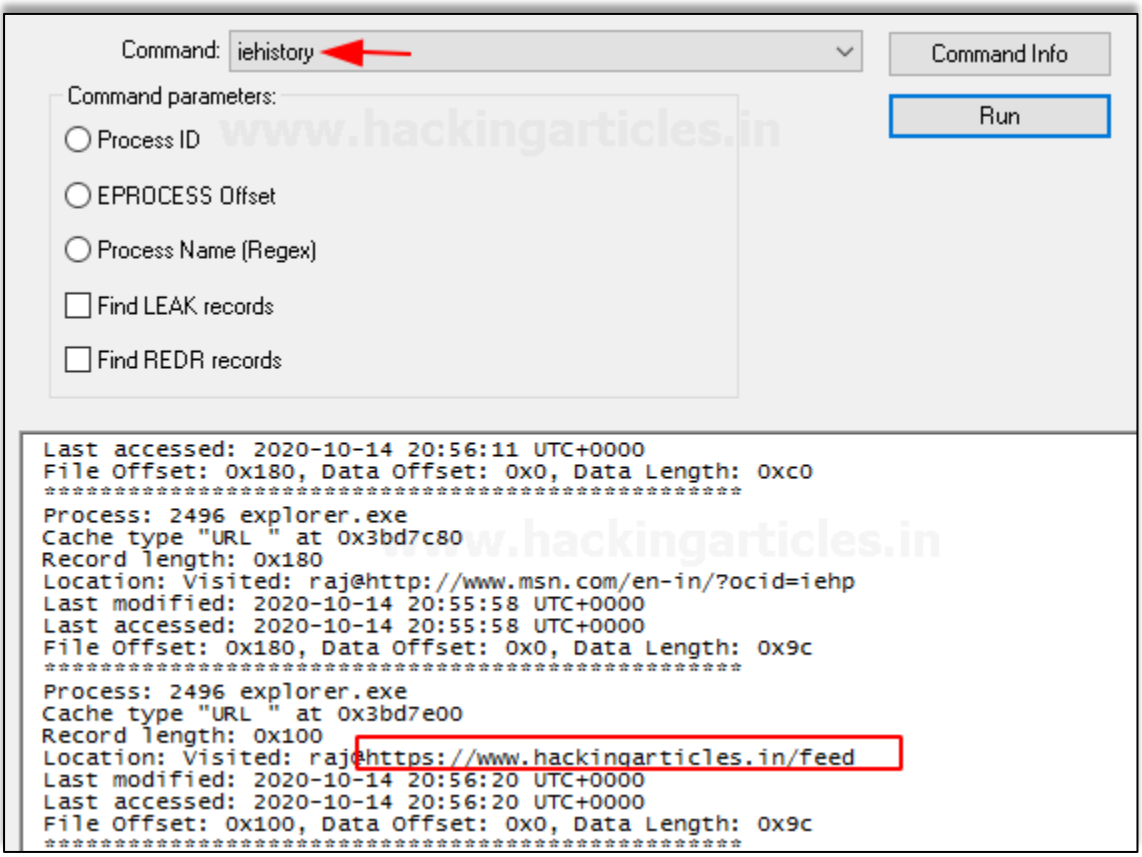
14. Vadtree

This process displays the VAD nodes in a tree form.



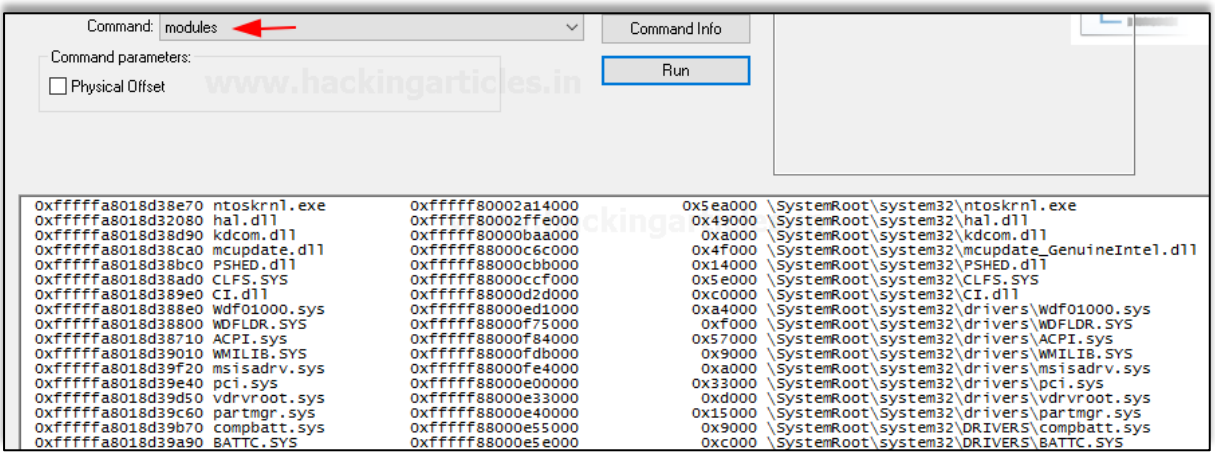
15. iehistory

This Plugin helps in recovering the fragments of the Internet explore history index.dat named cache files. It displays FTP and HTTP links that were accessed, links that were redirected, any deleted entries.



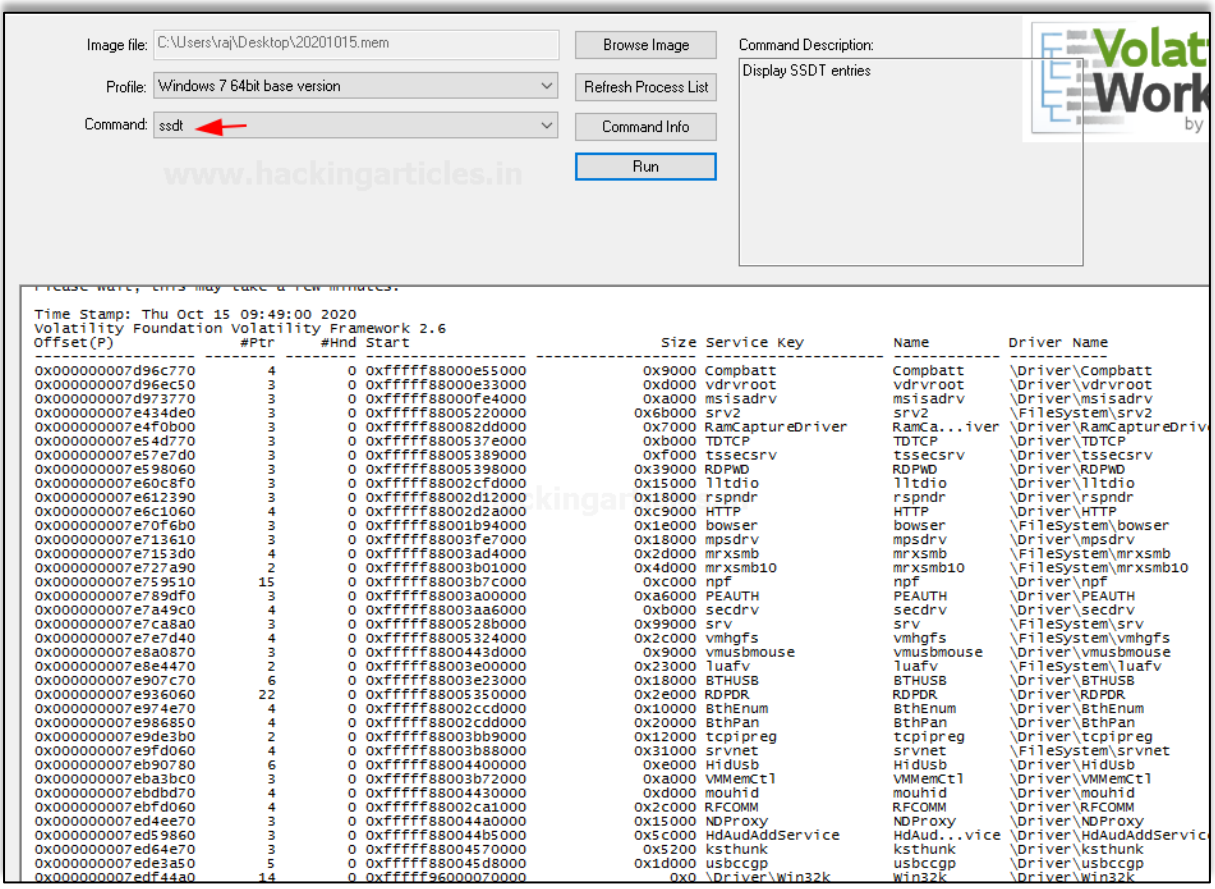
16. Modules

This command is used to list the kernel drivers that are present in the system.



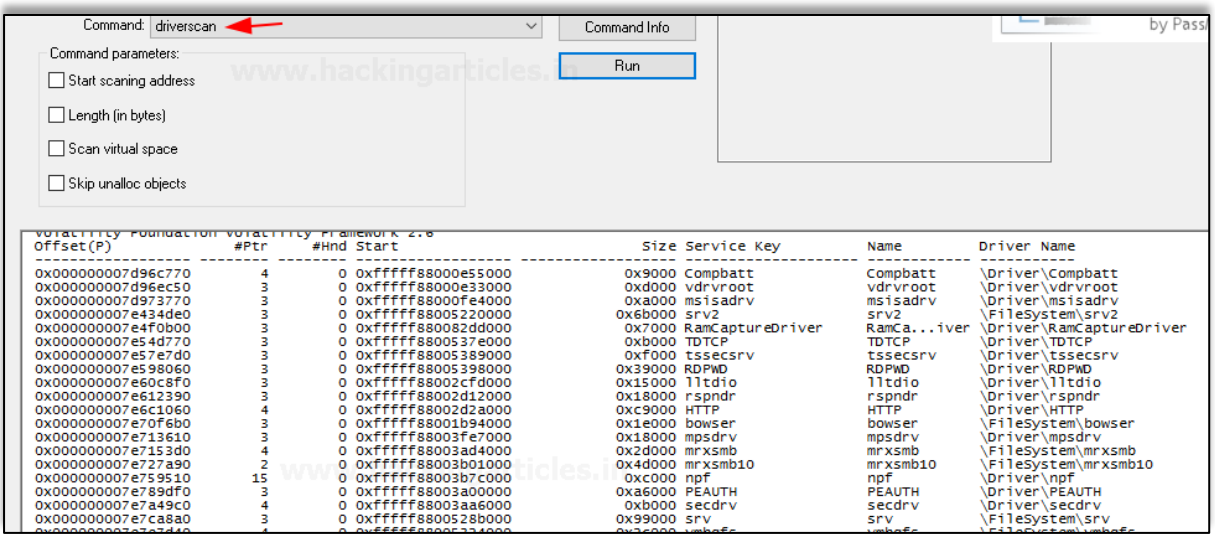
17. SSDT

This command is used to list the functions present in the original and GUI SSDTs. It displays the index, the name of the function, and the owner of the driver of each entry in the SSDT.



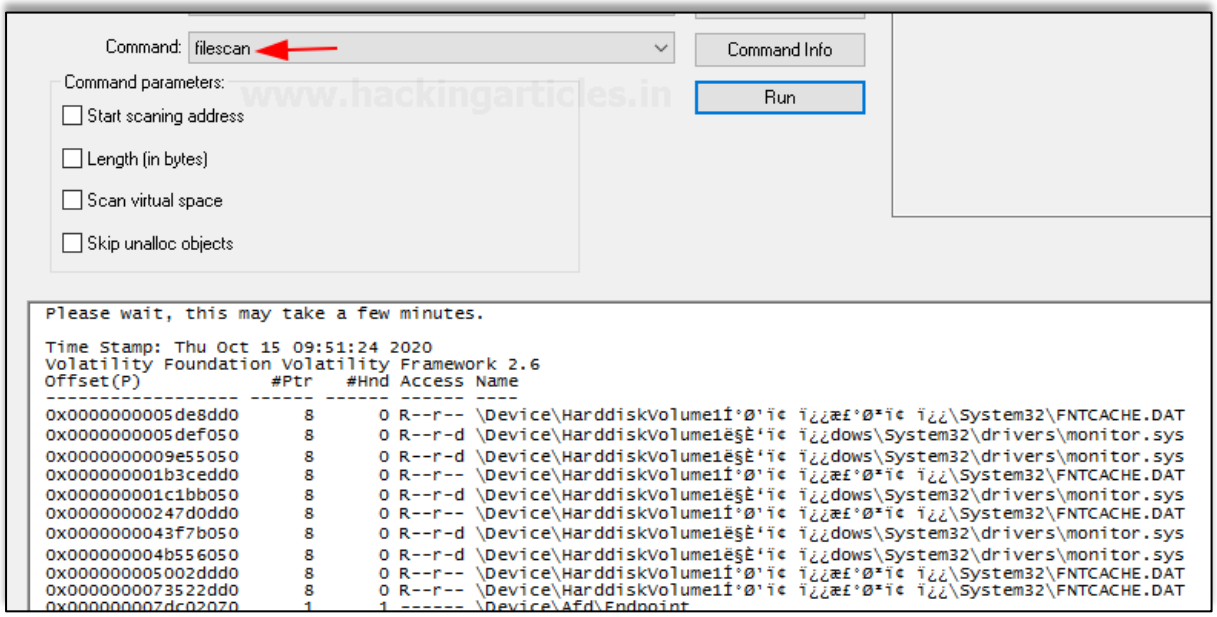
18. Driverscan

This command can be used to find the DRIVER_OBJECT present in the physical memory by making use of a pool tag scan.



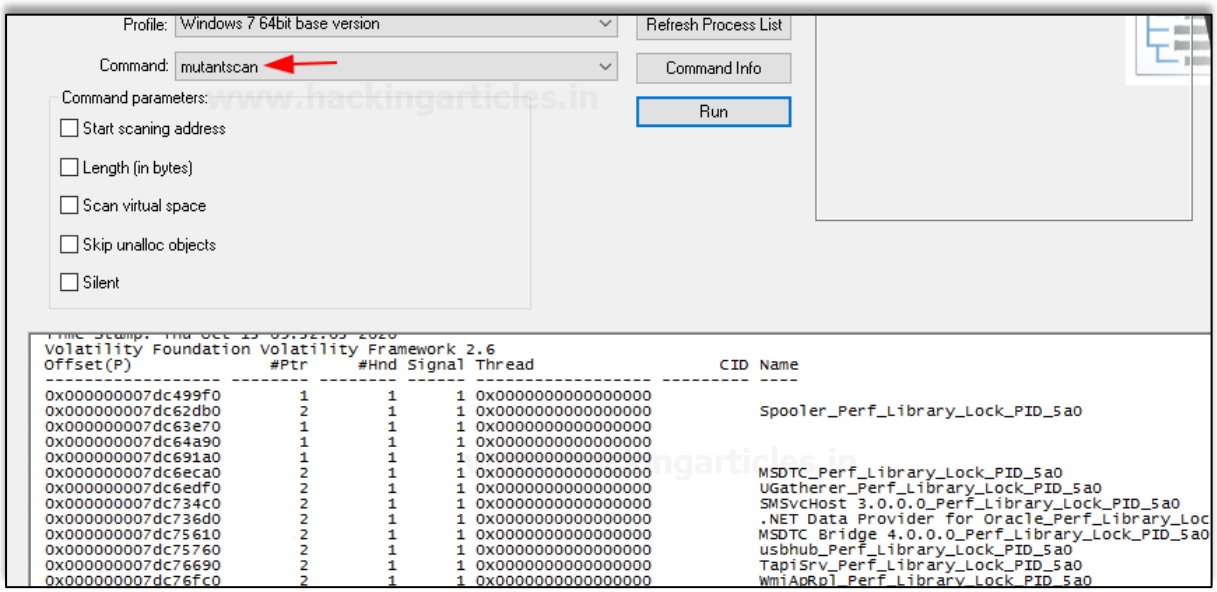
19. File Scan

This command can be used to find File_object that is present in the physical memory by making use of a pool tag scan. This command will help in finding open files in the system dump even if they are hidden with the help of rootkit.



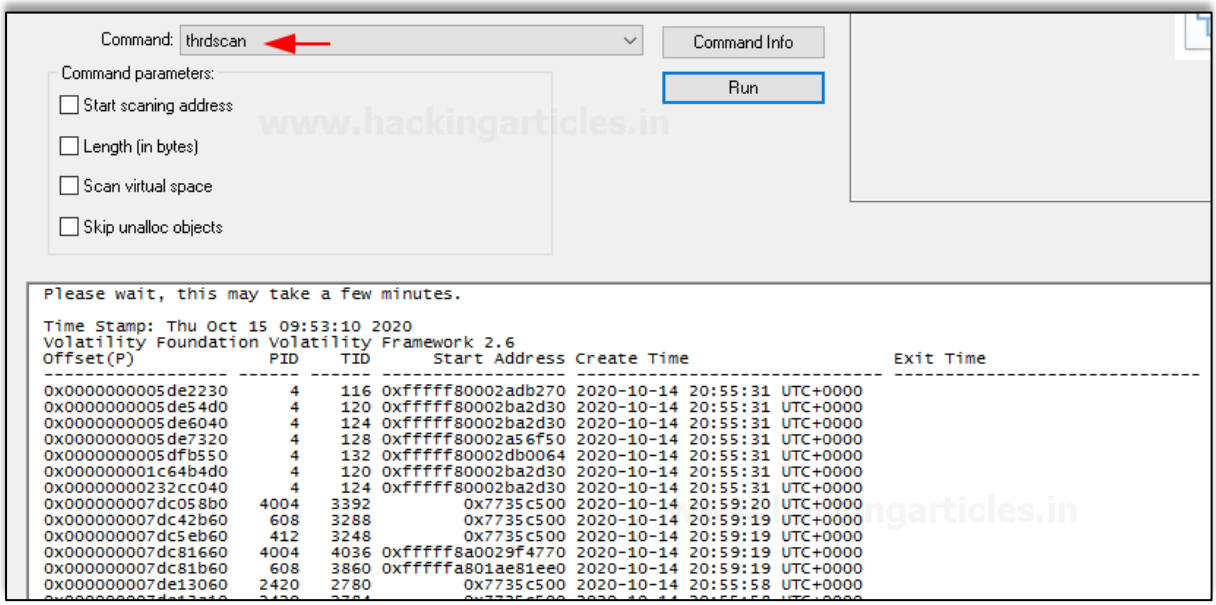
20. Mutant scan

This command is used to scan the physical memory of mutant objects by making use of pool tag scanning.



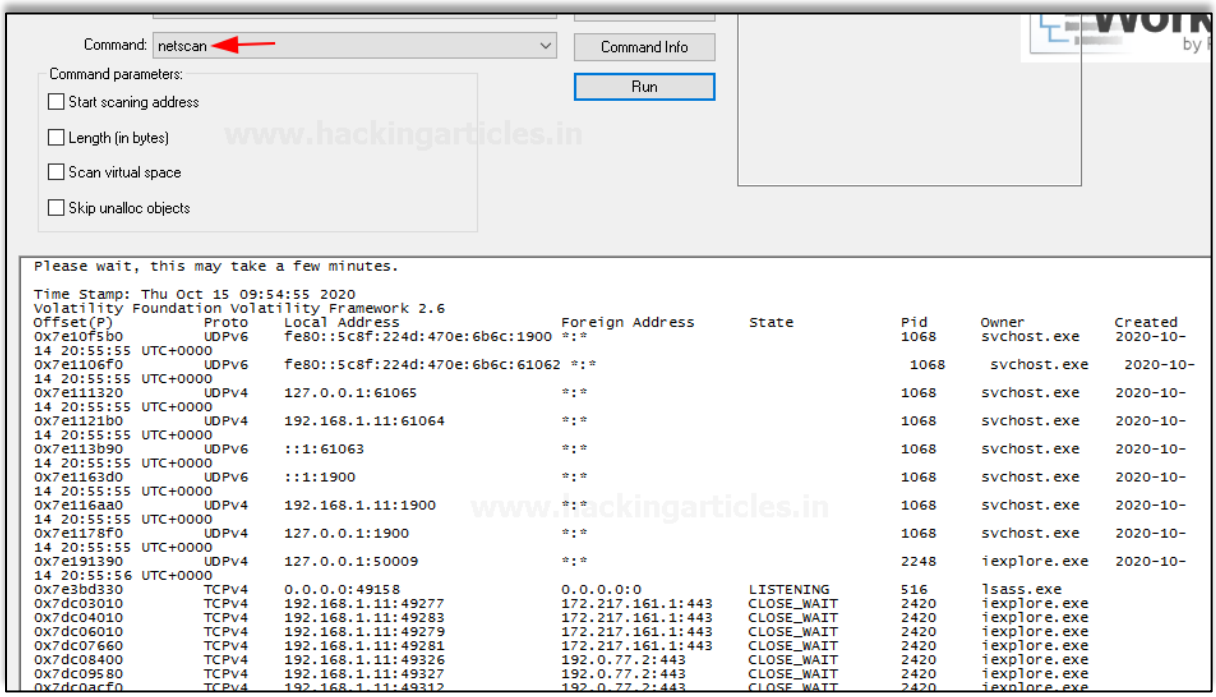
21. Thrdscan

This command is used to find the thread objects that are present in the physical memory with the help of a pool tag scan. It contains certain fields that can identify its parent processes which can help in finding hidden processes.



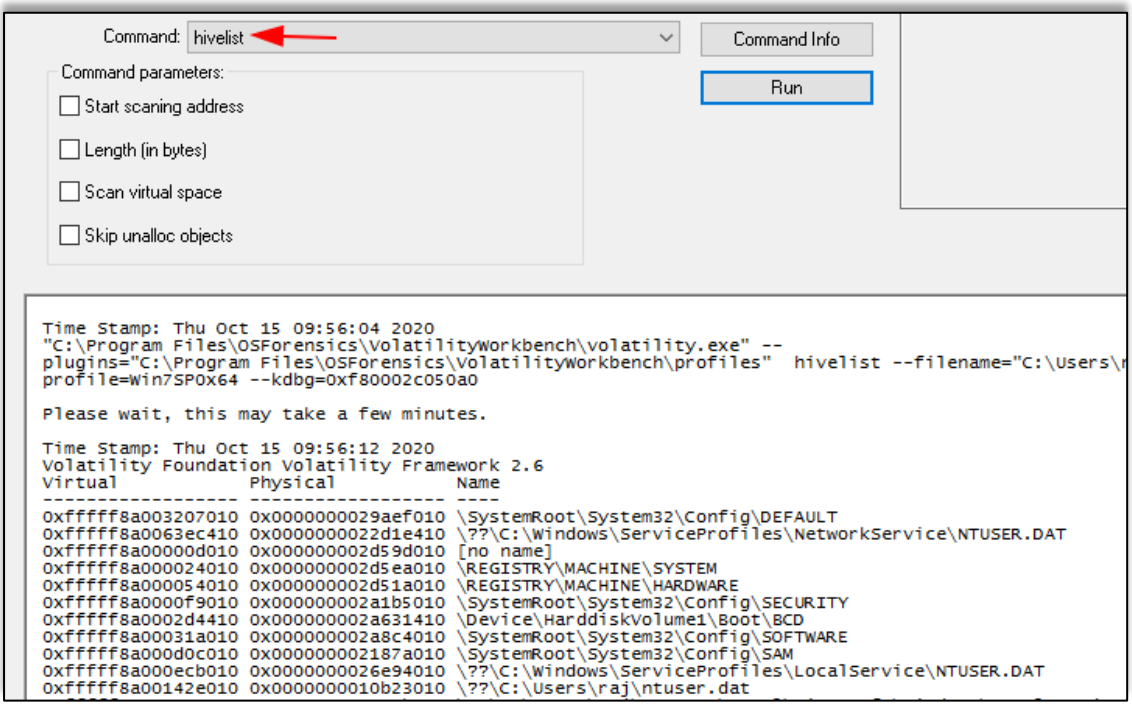
22. Netscan

This plugin helps in finding network-related artefacts present in the memory dump. It makes use of pool tag scanning. This plugin finds all the TCP endpoints, TCP listeners, UDP endpoints, and UDP listeners. It provides details about the local and remote IP and also about the local and remote port



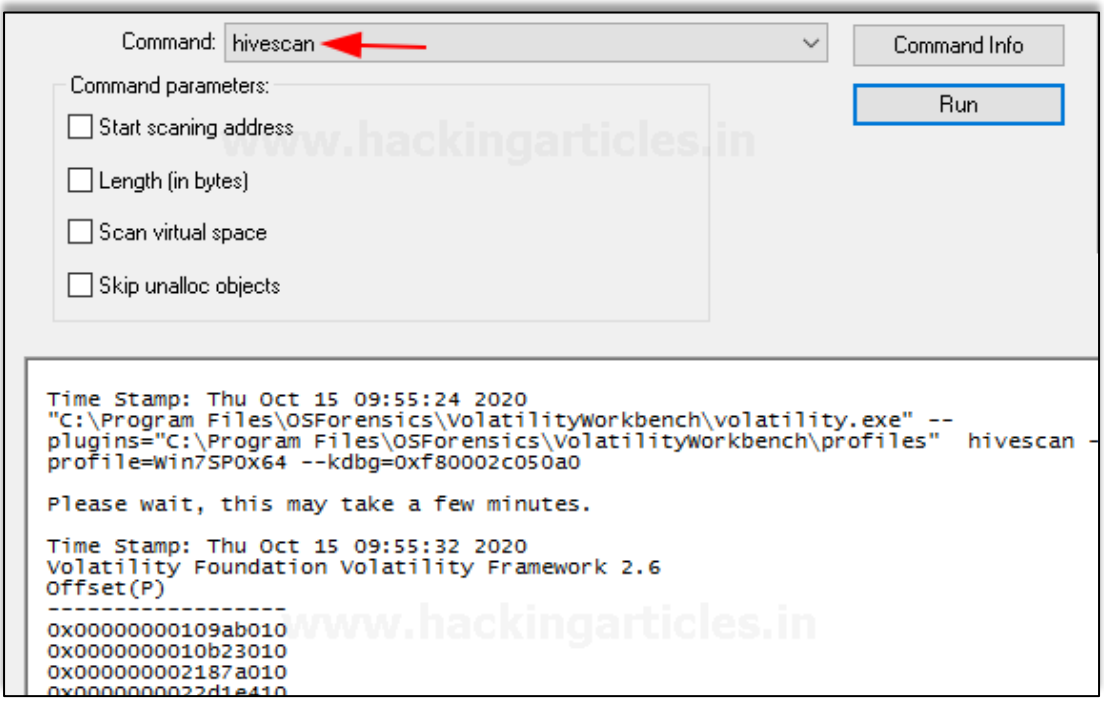
23. Hivelist

This command can be used to locate the virtual addresses present in the registry hives in memory, and their entire paths to hive on the disk.



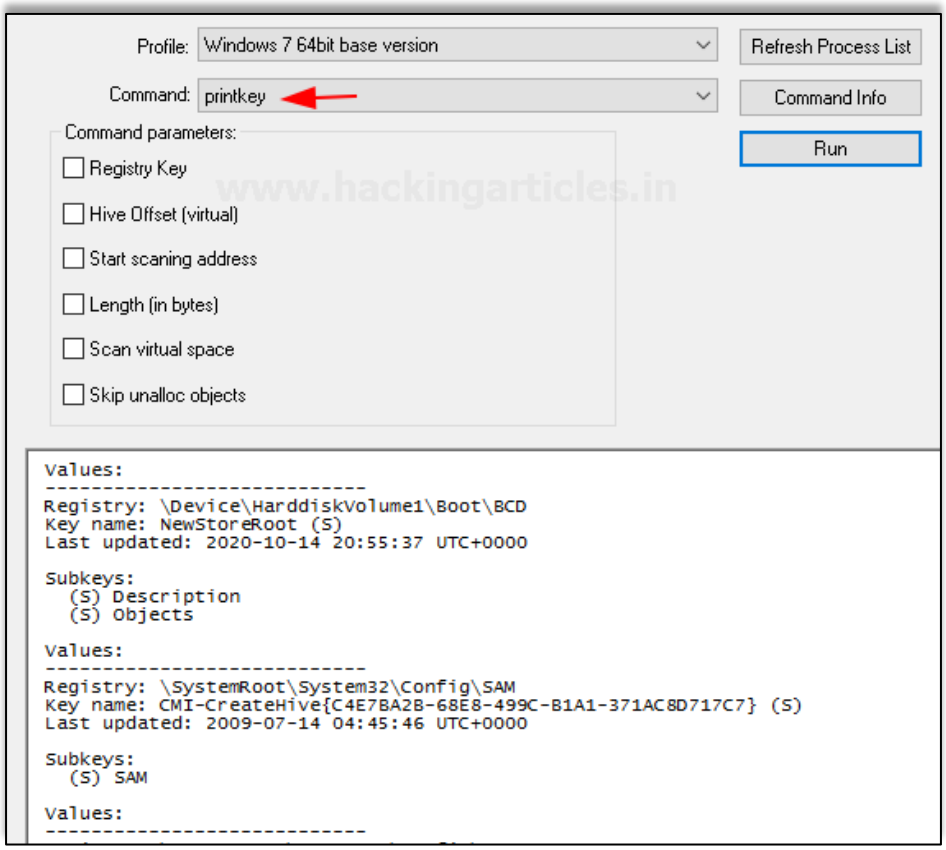
24. Hivescan

This command is used to find the physical address of the registry hives that are present in the memory. It is there to support the hivelist.



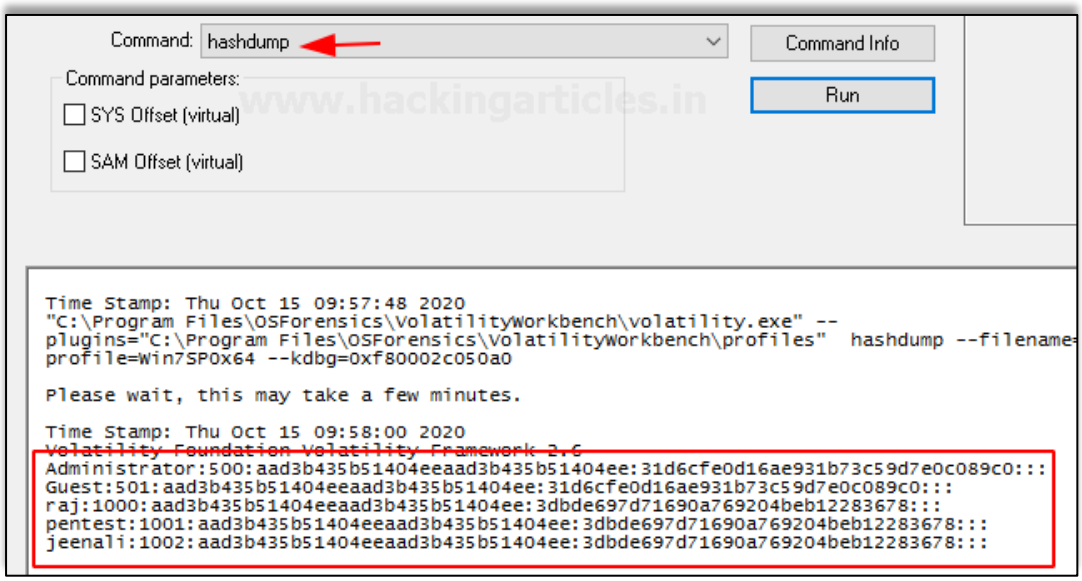
25. Printkey

This command is used to display the values, data, subkeys, and data types that are present in a specified registry.



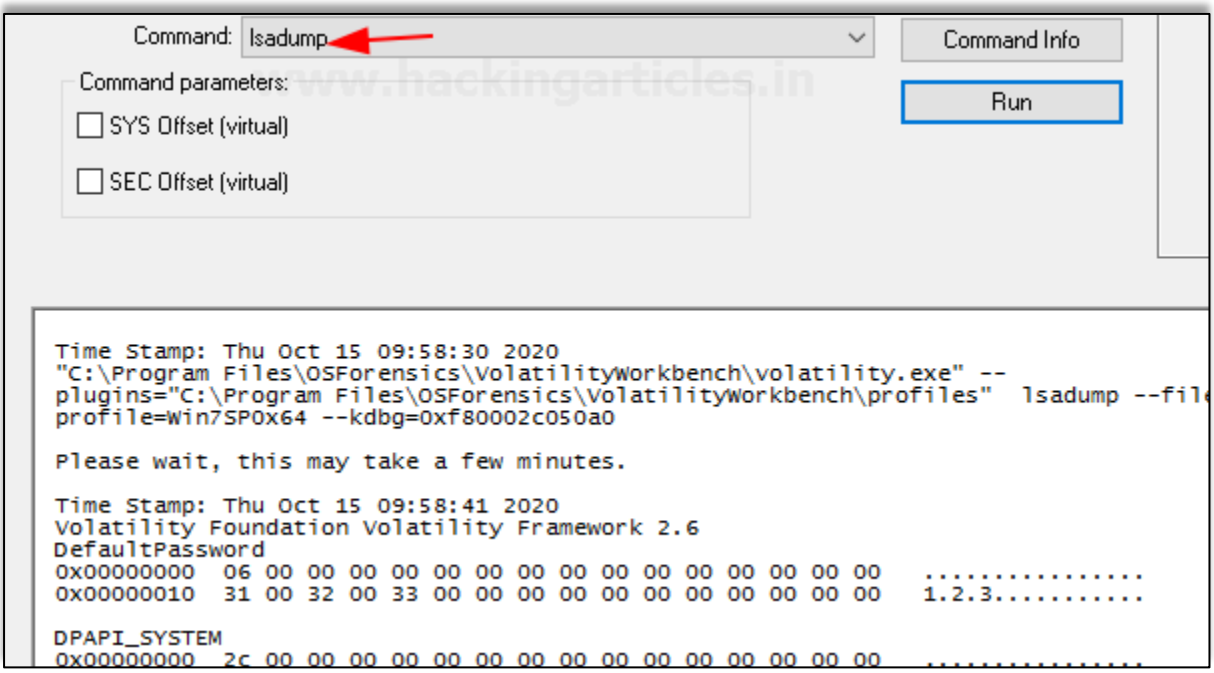
26. Hashdump

This command can be used to extract and decrypt cached domain credentials stored in the registry which can be availed from the memory dump. The hashes that are availed from the memory dump can be cracked using John the Ripper, Hashcat, etc



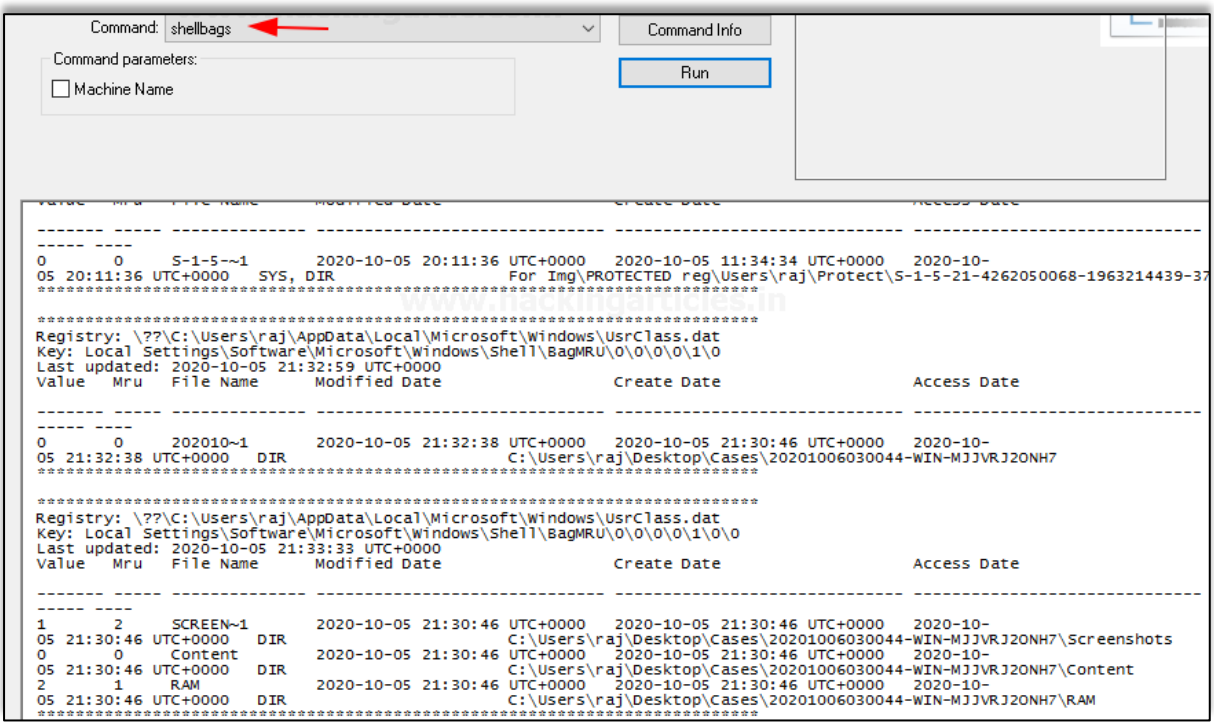
27. Lsadump

This command is used to dump LSA secrets from the registry in the memory dump. This plugin gives out information like the default password, the RDP public key, etc.



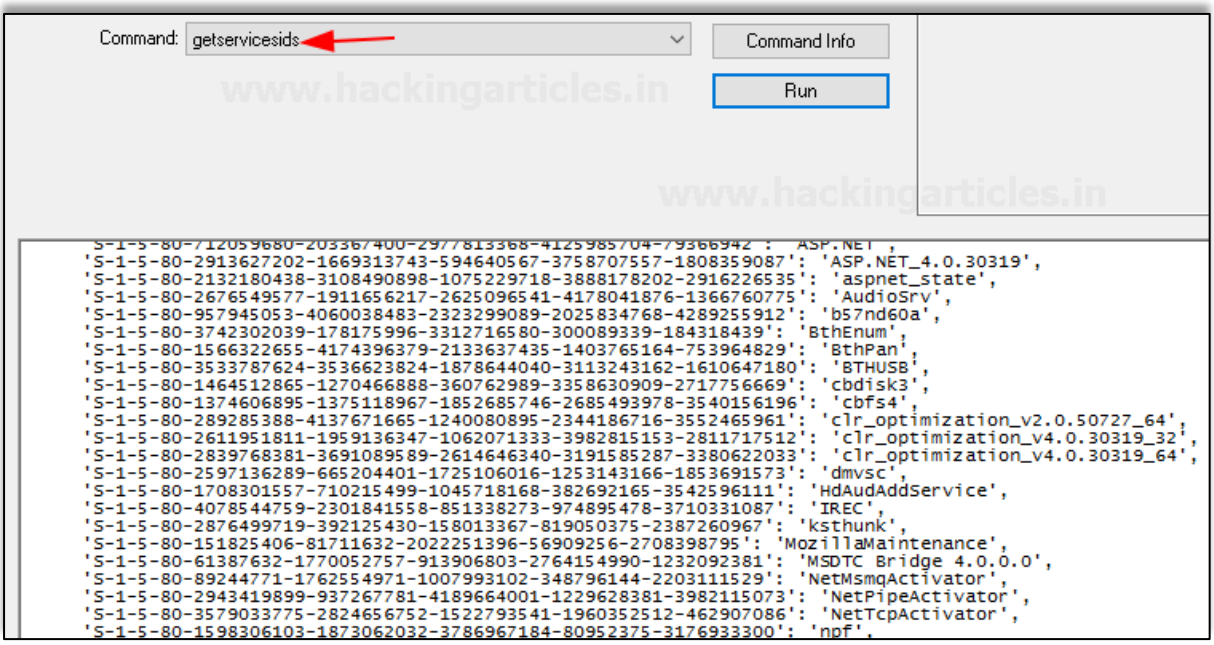
28. Shellbags

This command usually parses and prints the shellbag information that is obtained from the registry.



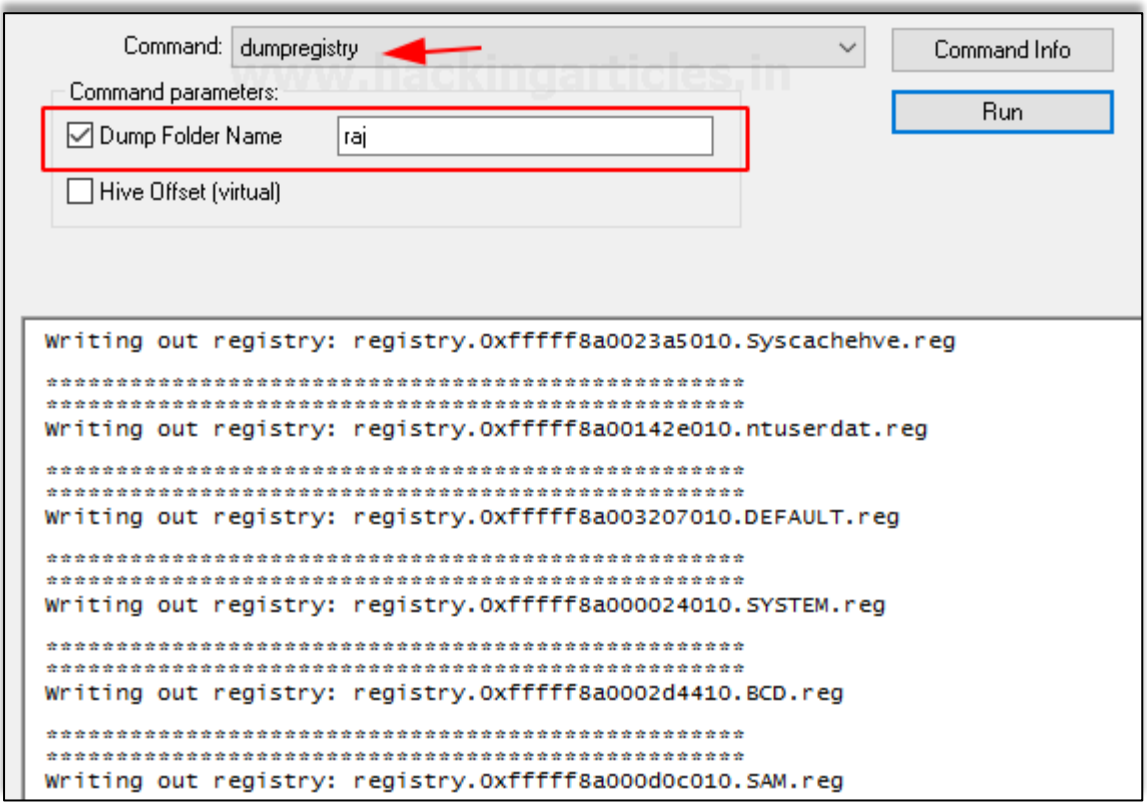
29. Getservicesids

This command does the work of calculating the SIDz for the services that are present on the machine. The name of the services has been taken from the registry.



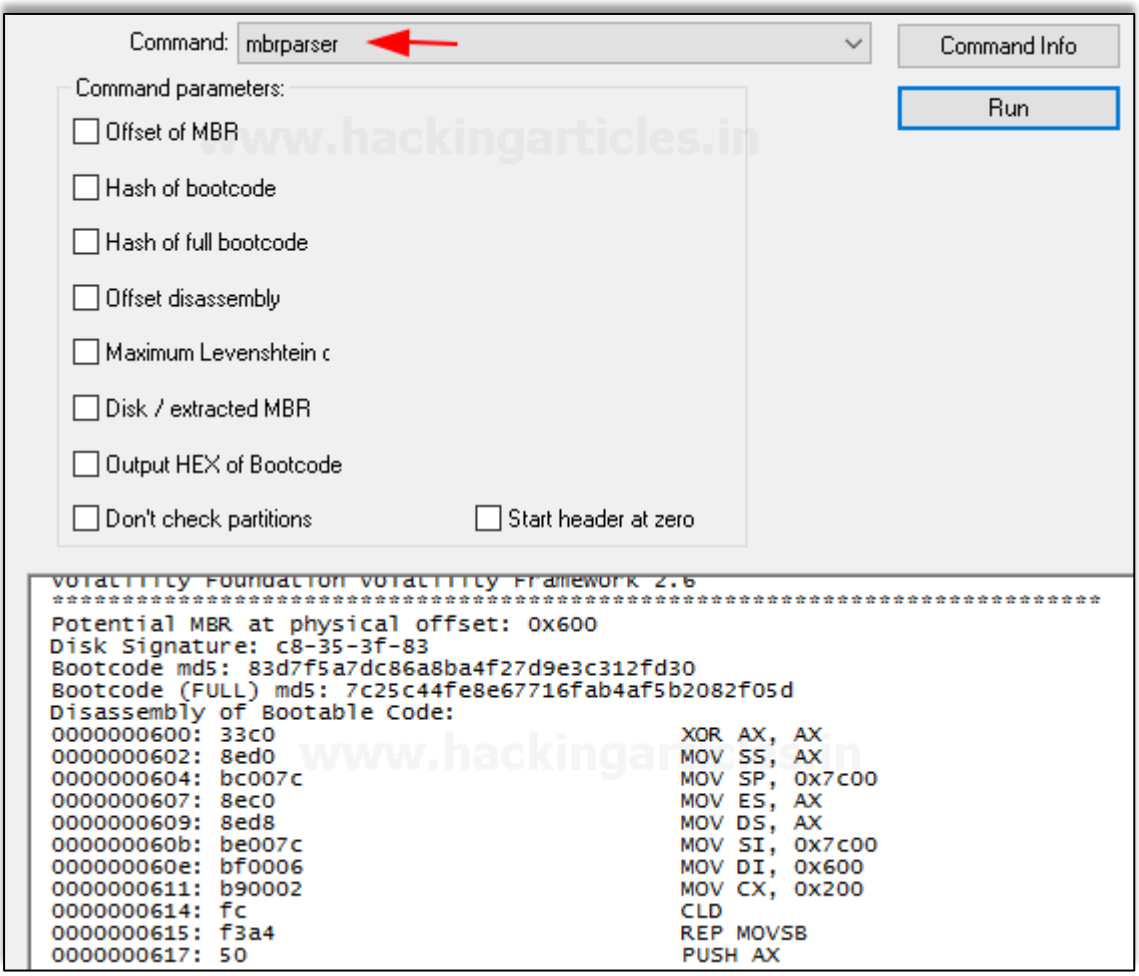
30. Dumpregistry

This plugin allows one to dump a registry hive into a disk location.



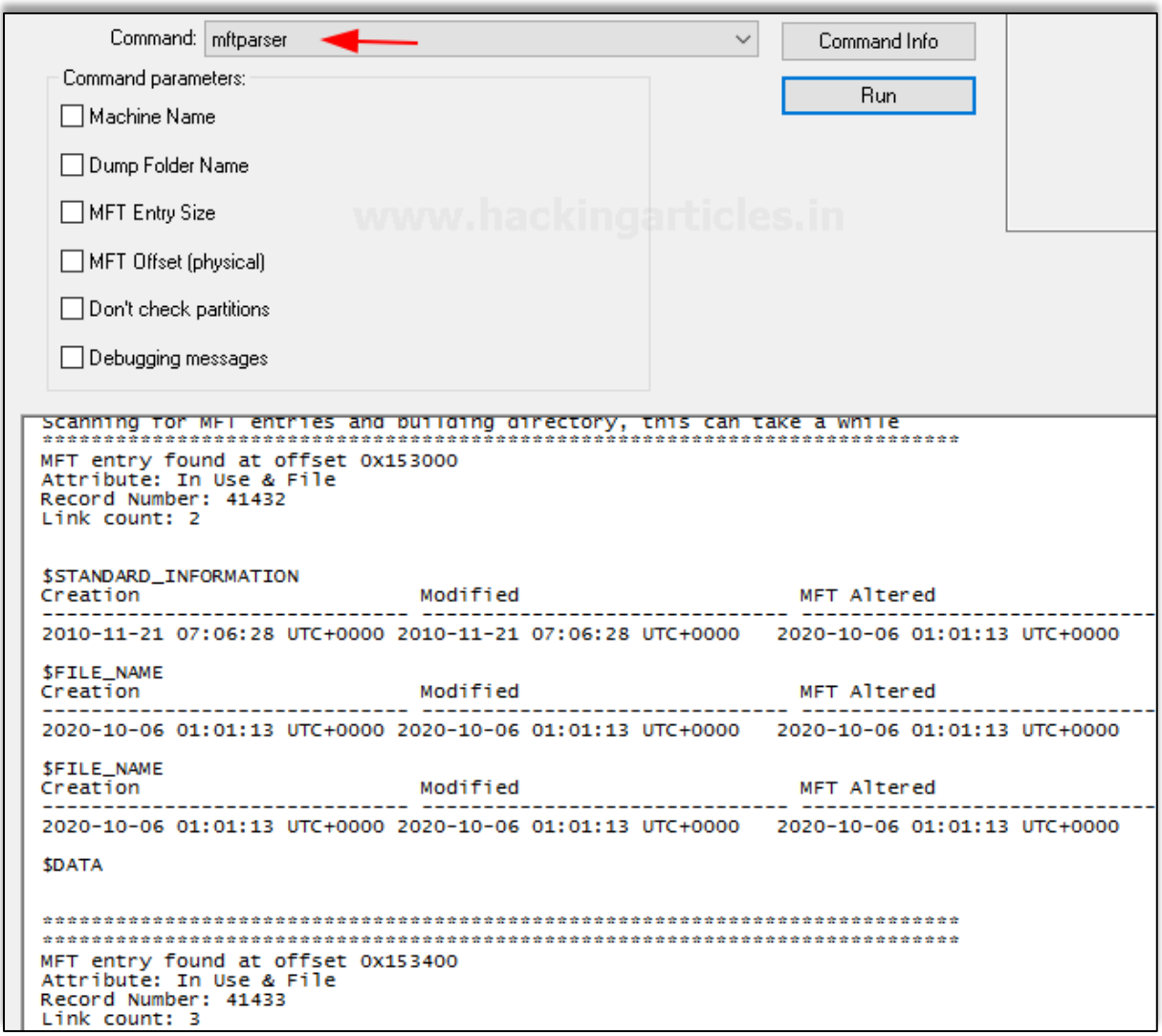
31. Mbrparser

This command scans and parses potential MBR from the memory dump. There are various ways to find MBR and the way of filtering it.



32. Mftparser

This command is used to scan the MFT entries in the memory dump and prints out the information for certain types of file attributes.



References

- https://www.hackingarticles.in/memory-forensics-using-volatility-framework/
- https://www.hackingarticles.in/memory-forensics-using-volatility-workbench/

iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials

ADVANCED

- Burp Suite Pro
- Web Services-API
- Pro Infrastructure VAPT
- Computer Forensics
- Android Pentest
- Advanced Metasploit
- CTF

EXPERT

- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux