Memory Forensics: Using Volatility Framework

October 29, 2020 By Raj Chandel

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 a scene of the crime.

Table of Contents

- **Memory Acquisition**
- **Importance of Memory Acquisition**
- **Dump Format Supported**
- **Memory Analysis Plugins**
- **Imageinfo**
- **Kdbgscan**
- **Processes**
- **DLLs**
- **Handles**
- **Netscan**
- **Hivelist**
- **Timeliner**
- **Hashdump**
- **Lsadump**
- **Modscan**
- **Filescan**
- **Svcscan**
- **History**
- **Dumpregistry**
- **Moddump**
- **Procdump**
- **Memdump**
- **notepad**

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 a capture, the following artifacts 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.

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

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 analyzing the memory dump in Kali Linux where Volatility comes pre-installed.

Dump Format Supported

- Raw format
- Hibernation File
- VM snapshot
- Microsoft crash dump

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
```

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
```

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

```
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
```

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. 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

**************************************************
Instantiating KDBG using: /root/ram.mem WinXPSP2x86 (5.1.0 32bit)
Offset (P)                    : 0x2bfc0a0
KDBG owner tag check          : True
Profile suggestion (KDBGHeader): Win2008R2SP1x64_24000
PsActiveProcessHead           : 0x2c32b90
PsLoadedModuleList            : 0x2c50e90
KernelBase                    : 0xfffff80002a0b000
```

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. 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 analyzed in the memory dump. There are a few plugins that can be used to list down the processes

Pslist

To identify the presence of any rogue processes and to view any high-level running processes, one can use

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

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.

```
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. It can not detect hidden or unlinked processes.

```
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
0x000000013e42db30 msdtc.exe           2000    512 0x000000008fd96000 2020-10-01 16:24:50 UTC+0000
0x000000013e4aa200 dwm.exe             2568    840 0x000000008a6bb000 2020-10-01 16:25:32 UTC+0000
0x000000013e525b30 VSSVC.exe           2060    512 0x000000008ccbf000 2020-10-01 16:24:54 UTC+0000
0x000000013e5d0b30 taskhost.exe        2268    512 0x000000008a364000 2020-10-01 16:25:25 UTC+0000
0x000000013e5f9b30 sdclt.exe           2176    512 0x00000001290b6000 2020-10-01 16:27:43 UTC+0000
0x000000013e62bb30 svchost.exe         2752    512 0x000000006b96e000 2020-10-01 16:26:48 UTC+0000
0x000000013e661b30 VGAuthService.      1308    512 0x0000000095d5f000 2020-10-01 16:24:44 UTC+0000
0x000000013e698b30 vmtoolsd.exe        1368    512 0x0000000094a93000 2020-10-01 16:24:46 UTC+0000
0x000000013e7717c0 dllhost.exe         1748    512 0x0000000092967000 2020-10-01 16:24:48 UTC+0000
0x000000013e78e9e0 dllhost.exe         1920    512 0x00000000916ef000 2020-10-01 16:24:50 UTC+0000
0x000000013e830b30 svchost.exe          800    512 0x000000009cc2c000 2020-10-01 16:24:40 UTC+0000
0x000000013e83b060 putty.exe           1936   2592 0x0000000061d5b000 2020-10-01 16:27:00 UTC+0000
0x000000013e848890 svchost.exe          840    512 0x000000009cd3f000 2020-10-01 16:24:40 UTC+0000
0x000000013e853b30 svchost.exe          868    512 0x000000009ca47000 2020-10-01 16:24:40 UTC+0000
0x000000013e87bb30 audiodg.exe          944    800 0x000000009c231000 2020-10-01 16:24:40 UTC+0000
0x000000013e88cb30 explorer.exe        2592   2560 0x00000000885d6000 2020-10-01 16:25:32 UTC+0000
0x000000013e8a9b30 svchost.exe          128    512 0x000000009b4d3000 2020-10-01 16:24:41 UTC+0000
0x000000013e8ce060 svchost.exe          400    512 0x000000009aed9000 2020-10-01 16:24:41 UTC+0000
```

Pstree

In this plugin, the pslist 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
.. 0xfffffa8032430b30:svchost.exe                     800    512     23    448 2020-10
... 0xfffffa803247bb30:audiodg.exe                    944    800      6    130 2020-10
.. 0xfffffa803278e9e0:dllhost.exe                    1920    512     17    213 2020-10
.. 0xfffffa8032b7ab30:SearchIndexer.                 2896    512     15    629 2020-10
... 0xfffffa8032bd57c0:SearchFilterHo                2992   2896      4     86 2020-10
... 0xfffffa8032bc4710:SearchProtocol                2972   2896      8    233 2020-10
.. 0xfffffa8030f75060:svchost.exe                    2764    512      6     73 2020-10
.. 0xfffffa8032448890:svchost.exe                     840    512     20    433 2020-10
... 0xfffffa80328aa200:dwm.exe                       2568    840      6    137 2020-10
.. 0xfffffa8032925b30:VSSVC.exe                      2060    512      6    121 2020-10
.. 0xfffffa803262bb30:svchost.exe                    2752    512     14    342 2020-10
.. 0xfffffa80327717c0:dllhost.exe                    1748    512     22    213 2020-10
.. 0xfffffa8032698b30:vmtoolsd.exe                   1368    512     13    274 2020-10
.. 0xfffffa80329d0b30:taskhost.exe                   2268    512      9    167 2020-10
.. 0xfffffa80323cc240:taskhost.exe                   1776    512      5 66 ... 3 2020-10
.. 0xfffffa8032453b30:svchost.exe                     868    512     49   1114 2020-10
.. 0xfffffa80323a9b30:svchost.exe                     620    512     12    376 2020-10
```


DLLs

DLLlist

```
volatility -f ram.mem --profile=Win7SP1x64 dlllist -p 1116,788
```

DLLs stand for Dynamic-link library automatically that is added to this list when a process according to calls Load Library and they aren't removed until. To display the DLLs for any particular process instead of all processes.

```
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
0x0000000077370000          0x11f000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007fefd4d0000            0x6b000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007fefea40000            0xdb000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007fefeec0000            0x9f000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007feff580000            0x1f000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007fefeb20000           0x12d000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x0000000077270000            0xfa000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007fefe7b0000            0x67000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007feff570000             0xe000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007feff6d0000            0xc9000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007feff4d0000            0x71000             0xffff 2020-10-01 16:26:51 UTC+0000   C:\Windows
0x000007fefd850000           0xd88000             0xffff 2020-10-01 16:26:51 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/ramdump/
```

```
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
```

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\DESCRIPTIO
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
```

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 gets its -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:~#
```

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
0x13e6fbef0        TCPv6    :::445                         :::0                 LISTENING
0x13e6feef0        TCPv4    0.0.0.0:49155                  0.0.0.0:0            LISTENING
0x13e6feef0        TCPv6    :::49155                       :::0                 LISTENING
0x13e70f670        TCPv4    0.0.0.0:3389                   0.0.0.0:0            LISTENING
0x13e7728f0        TCPv4    0.0.0.0:49156                  0.0.0.0:0            LISTENING
0x13e7c3a60        TCPv4    0.0.0.0:49156                  0.0.0.0:0            LISTENING
0x13e7c3a60        TCPv6    :::49156                       :::0                 LISTENING
0x13e7e8320        TCPv4    0.0.0.0:3389                   0.0.0.0:0            LISTENING
0x13e7e8320        TCPv6    :::3389                        :::0                 LISTENING
0x13e805430        TCPv4    0.0.0.0:49152                  0.0.0.0:0            LISTENING
0x13e805430        TCPv6    :::49152                       :::0                 LISTENING
0x13e805ef0        TCPv4    0.0.0.0:49152                  0.0.0.0:0            LISTENING
0x13e844880        TCPv4    0.0.0.0:49153                  0.0.0.0:0            LISTENING
```

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
```

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: Compute
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
```

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:::
```

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   ,...............
0x00000010  01 00 00 00 6d 0a d5 a6 c8 ab aa fc b5 40 02 f7   ....m........@..
0x00000020  29 b2 5f 3f 6f 98 d7 da 6a 69 16 26 3c 49 8f 76   )._?o...ji.&<I.v
0x00000030  71 84 e5 f6 1e 34 fb ef 3b c2 71 00 00 00 00 00   q....4..;.q.....
```

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. To

```
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
0x000000013e6385a0 tdtcp.sys            0xfffff88005bab000             0xb000 \SystemRoot
0x000000013e70e770 vmhgfs.sys           0xfffff88005b51000            0x2c000 \SystemRoot
0x000000013e7d33d0 tssecsrv.sys         0xfffff88005bb6000             0xf000 \SystemRoot
0x000000013e8c5510 rspndr.sys           0xfffff8800545d000            0x18000 \SystemRoot
0x000000013e8c71b0 lltdio.sys           0xfffff88001823000            0x15000 \SystemRoot
0x000000013e91e770 HTTP.sys             0xfffff88005475000            0xc9000 \SystemRoot
0x000000013e975e10 bowser.sys           0xfffff8800553e000            0x1e000 \SystemRoot
0x000000013e9783c0 mpsdrv.sys           0xfffff8800555c000            0x18000 \SystemRoot
0x000000013e97b510 mrxsmb.sys           0xfffff88005574000            0x2d000 \SystemRoot
0x000000013e981300 mrxsmb10.sys         0xfffff880055a1000            0x4d000 \SystemRoot
0x000000013e9a88e0 mrxsmb20.sys         0xfffff88005400000            0x24000 \SystemRoot
0x000000013e9afcf0 srv.sys              0xfffff88005ab8000            0x99000 \SystemRoot
0x000000013e9c24c0 vmmemctl.sys         0xfffff88005424000             0xa000 \SystemRoot
```

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
0x000000013e022070      4      0 R--r-d \Device\HarddiskVolume1\Windows\System32\QAGENT.DLL
0x000000013e023070     17      1 R--r-d \Device\HarddiskVolume1\Windows\System32\en-US\bthprops.
0x000000013e023f20     11      0 R--r-d \Device\HarddiskVolume1\Windows\System32\FXSRESM.dll
0x000000013e025900      1      1 R--rw- \Device\HarddiskVolume1\Windows\winsxs\amd64_microsoft.
0x000000013e025c70      7      0 R--r-d \Device\HarddiskVolume1\Windows\System32\FXSST.dll
0x000000013e025dc0      7      0 R--r-d \Device\HarddiskVolume1\Windows\System32\FXSAPI.dll
0x000000013e027070     16      0 R--r-d \Device\HarddiskVolume1\Windows\System32\en-US\gameux.d
0x000000013e02add0      1      1 R--rw- \Device\HarddiskVolume1\Windows\winsxs\amd64_microsoft.
0x000000013e02af20      1      1 R--rw- \Device\HarddiskVolume1\Windows\winsxs\amd64_microsoft.
0x000000013e02bc20      1      1 -W-rw- \Device\HarddiskVolume1\Users\raj\AppData\Local\Temp\FX
0x000000013e0344a0      1      1 R--r-d \Device\HarddiskVolume1\Windows\System32\en-US\gameux.d
0x000000013e035e20      4      0 RW-rwd \Device\HarddiskVolume1\$Directory
0x000000013e037430      1      1 RW-rw- \Device\HarddiskVolume1\Users\raj\AppData\Local\Microso
```

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: 0×cc8500
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 LocalServiceNetworkRestricte

Offset: 0×cc8410
Order: 69
Start: SERVICE_SYSTEM_START
Process ID: -
Service Name: DfsC
Display Name: DFS Namespace Client Driver
Service Type: SERVICE_FILE_SYSTEM_DRIVER
Service State: SERVICE_RUNNING
Binary Path: \FileSystem\DfsC

Offset: 0×cc8320
Order: 68
Start: SERVICE_DEMAND_START
Process ID: -
Service Name: defragsvc
Display Name: Disk Defragmenter
Service Type: SERVICE_WIN32_OWN_PROCESS
Service State: SERVICE_STOPPED
Binary Path: -
```

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=Win7SP1×64 cmdscan
Volatility Foundation Volatility Framework 2.6
**************************************************
CommandProcess: conhost.exe Pid: 2840
CommandHistory: 0×1e8ce0 Application: RamCapture64.exe Flags: Allocated
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0×64
Cmd #15 @ 0×180158:
Cmd #16 @ 0×1e7e50:
root@kali:~#
```

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
**************************************************
Process: 2592 explorer.exe
Cache type "URL " at 0x2955300
Record length: 0x100
Location: Visited: raj@file:///E:/raj.txt
Last modified: 2020-09-17 17:43:58 UTC+0000
Last accessed: 2020-09-17 17:43:58 UTC+0000
File Offset: 0x100, Data Offset: 0x0, Data Length: 0x88
**************************************************
Process: 2592 explorer.exe
Cache type "URL " at 0x2955400
Record length: 0x100
Location: Visited: raj@file:///C:/Users/raj/Desktop/memdump.mem
Last modified: 2020-09-26 11:48:11 UTC+0000
Last accessed: 2020-09-26 11:48:11 UTC+0000
File Offset: 0x100, Data Offset: 0x0, Data Length: 0xa0
**************************************************
Process: 2592 explorer.exe
Cache type "URL " at 0x2965000
```

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

*****************************************************
*****************************************************
Writing out registry: registry.0xfffff8a00058c010.SOFTWARE.reg

Physical layer returned None for index 23f2000, filling with NULL
Physical layer returned None for index 23f3000, filling with NULL
Physical layer returned None for index 23f4000, filling with NULL
*****************************************************
*****************************************************
Writing out registry: registry.0xfffff8a00000f010.no_name.reg

*****************************************************
*****************************************************
Writing out registry: registry.0xfffff8a000e4d010.NTUSERDAT.reg
```

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
0xfffff880011be000 pacer.sys            OK: driver.fffff880011be000.sys
0xfffff8800186c000 tcpip.sys            OK: driver.fffff8800186c000.sys
0xfffff88000c00000 rasl2tp.sys          OK: driver.fffff88000c00000.sys
0xfffff88001000000 storport.sys         OK: driver.fffff88001000000.sys
0xfffff96000600000 cdd.dll              OK: driver.fffff96000600000.sys
0xfffff880015e1000 Msfs.SYS             OK: driver.fffff880015e1000.sys
0xfffff88004263000 vm3dmp.sys           OK: driver.fffff88004263000.sys
0xfffff88001823000 lltdio.sys           OK: driver.fffff88001823000.sys
0xfffff880041e5000 intelppm.sys         OK: driver.fffff880041e5000.sys
0xfffff88005bc5000 RDPWD.SYS            OK: driver.fffff88005bc5000.sys
0xfffff880015c6000 pcw.sys              OK: driver.fffff880015c6000.sys
0xfffff960000e0000 win32k.sys           OK: driver.fffff960000e0000.sys
0xfffff88000fa7000 partmgr.sys          OK: driver.fffff88000fa7000.sys
0xfffff88001108000 intelide.sys         OK: driver.fffff88001108000.sys
0xfffff88005a00000 spsys.sys            OK: driver.fffff88005a00000.sys
```

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
0xfffffa8032369600 0x00000000ffa30000 lsm.exe              OK: executable.528.exe
0xfffffa80323a9b30 0x00000000ff020000 svchost.exe          OK: executable.620.exe
0xfffffa80323e7b30 0x00000000ff020000 svchost.exe          OK: executable.704.exe
0xfffffa8032430b30 0x00000000ff020000 svchost.exe          OK: executable.800.exe
0xfffffa8032448890 0x00000000ff020000 svchost.exe          OK: executable.840.exe
0xfffffa8032453b30 0x00000000ff020000 svchost.exe          OK: executable.868.exe
0xfffffa803247bb30 0x00000000ffc70000 audiodg.exe          OK: executable.944.exe
```

Memdump

The memdump plugin is used to dump the memory-resident pages of a process into a separate file. You can also look up 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
```

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=WinXPSP2x86 notepad
```

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

Process: 1804
```