# Critical - Memory dump scenario via TryHackMe

### Volatility on Windows

# Introduction

Tools used

- Volatility

"Hattori" reported strange behavior on his computer and realized that some PDF files had been encrypted, including a critical document to the company named *important_document.pdf*. It's suspected that some credentials were stolen, so the DFIR team captured some evidence.

## Memory Forensics

- Memory forensics is a subset of computer forensics that analyzes volatile memory, usually on a compromised machine; in Windows OS, it corresponds to the Random Access Memory (RAM), and its content is flushed with every reboot or shutdown, making it one of the usual initial task to perform during an incident.
- The process differs from disk forensics analysis since it not only provides information about what resides on the target computer but also provides information about the processes or applications that were running at a particular time and detailed information on the execution flow on a system that may not be present in regular storage units or application logs.
- This memory analysis can help with an immediate snapshot of an application's or a timestamp of an attacker's actions.

## Two main phases: Memory Acquisition & Memory Analysis

- We'll copy the live memory to a file, commonly referred to as a dump, to perform the analysis without risking losing the data from an inadvertent reboot on the compromised system and have proof of the analysis inc as needed
- During the memory analysis phase, we'll analyze the obtained memory dump of the forensic data

### Questions

1. **What type of memory is analyzed during a forensic memory task?**

RAM

2. **In which phase will you create a memory dump of the target system?**

Memory Acquisition

# Critical - Memory dump scenario via TryHackMe

### Volatility on Windows

## Environment & Setup

A memory dump named memdump.mem will be present at the home address at /home/analyst
Use Volatility 3, a popular choice among analysts to inspect a memory during an incident.

```
user@machine$ vol -h
usage: volatility [-h] [-c CONFIG] [--parallelism [{processes,threads,off}]] [-e EXTEND] [-p PLUGIN_DIRS] [-s SYMBOL_DIRS]
                  [-v] [-l LOG] [-o OUTPUT_DIR] [-q] [-r RENDERER] [-f FILE] [--write-config] [--save-config SAVE_CONFIG]
                  [--clear-cache] [--cache-path CACHE_PATH] [--offline] [--single-location SINGLE_LOCATION]
                  [--stackers [STACKERS [STACKERS ...]]]
                  [--single-swap-locations [SINGLE_SWAP_LOCATIONS [SINGLE_SWAP_LOCATIONS ...]]]
```

The command vol will execute Volatility3 in the terminal.
The -h switch can display the help menu. windows keyword as an argument to search for Windows plugins with the --help switch

```
user@machine$ vol windows --help
usage: volatility [-h] [-c CONFIG] [--parallelism [{processes,threads,off}]] [-e EXTEND] [-p PLUGIN_DIRS] [-s SYMBOL_DIRS] [-
v] [-l LOG] [-o OUTPUT_DIR] [-q] [-r RENDERER] [-f FILE] [--write-config] [--save-config SAVE_CONFIG] [--clear-cache] [--
cache-path CACHE_PATH] [--offline]
                  [--single-location SINGLE_LOCATION] [--stackers [STACKERS [STACKERS ...]]] [--single-swap-locations
[SINGLE_SWAP_LOCATIONS [SINGLE_SWAP_LOCATIONS ...]]]
                  plugin ...
volatility: error: argument plugin: plugin windows matches multiple plugins (windows.bigpools.BigPools,
windows.cachedump.Cachedump, windows.callbacks.Callbacks, windows.cmdline.CmdLine, windows.crashinfo.Crashinfo,
windows.devicetree.DeviceTree, windows.dlllist.DllList, windows.driverirp.DriverIrp, windows.drivermodule.DriverModule,
windows.driverscan.DriverScan, windows.dumpfiles.DumpFiles, windows.envars.Envars, windows.filescan.FileScan,
windows.getservicesids.GetServiceSIDs, windows.getsids.GetSIDs, windows.handles.Handles, windows.hashdump.Hashdump,
windows.info.Info, windows.joblinks.JobLinks, windows.ldrmodules.LdrModules, windows.lsadump.Lsadump,
windows.malfind.Malfind, windows.mbrscan.MBRScan, windows.memmap.Memmap, windows.mftscan.ADS, windows.mftscan.MFTScan,
windows.modscan.ModScan, windows.modules.Modules, windows.mutantscan.MutantScan, windows.netscan.NetScan,
windows.netstat.NetStat, windows.poolscanner.PoolScanner, windows.privileges.Privs, windows.pslist.PsList,
windows.psscan.PsScan, windows.pstree.PsTree, windows.registry.certificates.Certificates, windows.registry.hivelist.HiveList,
windows.registry.hivescan.HiveScan, windows.registry.printkey.PrintKey, windows.registry.userassist.UserAssist,
windows.sessions.Sessions, windows.skeleton_key_check.Skeleton_Key_Check, windows.ssdt.SSDT, windows.statistics.Statistics,
windows.strings.Strings, windows.svcscan.SvcScan, windows.symlinkscan.SymlinkScan, windows.vadinfo.VadInfo,
windows.vadwalk.VadWalk, windows.vadyarascan.VadYaraScan, windows.verinfo.VerInfo, windows.virtmap.VirtMap)
```

### Questions

1. **Which plugin can help us to get information about the OS running on the target machine?**

Windows.info

2. **Which tool referenced above can help us take a memory dump on a Linux OS?**

LIME

3. **Which command will display the help menu using Volatility on the target machine?**

vol -h

# Critical - Memory dump scenario via TryHackMe

### Volatility on Windows

## Gathering Target Intel

### Obtaining Information

Get information about the target using the `-f` switch to indicate the file to analyze, in this case, `memdump.mem` followed by the plugin `windows.info` used to gather general information

```
user@machine$ vol -f memdump.mem windows.info
Volatility 3 Framework 2.5.2
Progress:  100.00PDB scanning finished
VariableValue

Kernel Base  0xf9066161c000
DTB   0x1ac000
Symbolsfile:    ///home/analyst/volatility3-2.5.2/volatility3/symbols/windows/ntkrnlmp.pdb/4DBE144182FF4156845CD3BD8B654E56-1.json.xz
Is64Bit   False
IsPAE   False
layer_name   0 WindowsIntel32e
memory_layer   1 FileLayer
KdVersionBlock   0xf8066222a400
Major/Minor   15.19041
MachineType   34404
KeNumberProcessors   2
SystemTime   2024-02-24 22:52:52
NtSystemRoot   C:\Windows
NtProductType   NtProductWinNt
NtMajorVersion   7
NtMinorVersion   0
PE MajorOperatingSystemVersion   10
PE MinorOperatingSystemVersion   0
PE Machine   34404
PE TimeDateStamp   Sat Jan 13 03:45:32 2085
```

This shows information to identify the machine we are working on, such as architecture, number of processors, and version. All this can help us correlate information and data with other analyses performed on separate hardware of the compromised machine or the network itself while still having proof this is the machine that was compromised

Navigate to the `working` directory in the VM and execute the command `vol -f memdump.mem windows.info`

#### Questions

**Is the architecture of the machine x64 (64bit) Y/N?**

Yes

**What is the Version of the Windows OS**

10

**What is the base address of the kernel?**

0xf8066161b000

# Critical - Memory dump scenario via TryHackMe

### Volatility on Windows

## Hunting for Suspicious Activity

### Starting the Search

Use the plugin `windows.netstat` to see if there's an interesting or unusual connection navigate to the `/home/analyst` directory and execute the command.
`vol -f memdump.mem windows.netstat`

```
analyst@ip-10-10-219-197:~$ vol -f memdump.mem windows.netstat

Volatility 3 Framework 2.5.2
Progress:  100.00               PDB scanning finished
Offset  Proto   LocalAddr       LocalPort       ForeignAddr     ForeignPort     State   PID     Owner   Created

0xe50ed9170ac0  TCPv4   192.168.182.139 49723   192.16.49.85    80      CLOSE_WAIT      4780    SearchApp.exe   2024-02-24 22:48:49.000000
0xe50ed8a4ca20  TCPv4   192.168.182.139 49814   52.142.223.178  443     SYN_SENT        368     svchost.exe     2024-02-24 22:52:43.000000
0xe50ed9275a20  TCPv4   192.168.182.139 3389    192.168.182.150 49253   ESTABLISHED     744     svchost.exe     2024-02-24 22:47:52.000000
0xe50ed9df3a20  TCPv4   192.168.182.139 49745   13.107.213.254  443     CLOSE_WAIT      4780    SearchApp.exe   2024-02-24 22:50:42.000000
0xe50ed8c52a20  TCPv4   192.168.182.139 49719   23.222.237.202  443     CLOSE_WAIT      4780    SearchApp.exe   2024-02-24 22:48:47.000000
0xe50ed9427a20  TCPv4   192.168.182.139 49694   20.7.1.246      443     ESTABLISHED     368     svchost.exe     2024-02-24 22:47:54.000000
0xe50ed83ea4d0  TCPv4   192.168.182.139 49743   23.222.237.203  443     CLOSE_WAIT      4780    SearchApp.exe   2024-02-24 22:50:39.000000
0xe50edac57a20  TCPv4   192.168.182.139 49712   152.199.55.200  443     CLOSE_WAIT      4780    SearchApp.exe   2024-02-24 22:48:06.000000
0xe50ed9508a20  TCPv4   192.168.182.139 49744   23.222.237.203  443     CLOSE_WAIT      4780    SearchApp.exe   2024-02-24 22:50:39.000000
0xe50ed6390cb0  TCPv4   0.0.0.0 135     0.0.0.0 0       LISTENING       896     svchost.exe     2024-02-24 22:47:36.000000
0xe50ed6390cb0  TCPv6   ::      135     ::      0       LISTENING       896     svchost.exe     2024-02-24 22:47:36.000000
0xe50ed6391910  TCPv4   0.0.0.0 135     0.0.0.0 0       LISTENING       896     svchost.exe     2024-02-24 22:47:36.000000
0xe50ed6391bd0  TCPv4   192.168.182.139 139     0.0.0.0 0       LISTENING       4       System  2024-02-24 22:47:36.000000
0xe50ed818d310  TCPv4   0.0.0.0 445     0.0.0.0 0       LISTENING       4       System  2024-02-24 22:47:37.000000
0xe50ed818d310  TCPv6   ::      445     ::      0       LISTENING       4       System  2024-02-24 22:47:37.000000
0xe50ed36a59f0  TCPv4   0.0.0.0 3389    0.0.0.0 0       LISTENING       744     svchost.exe     2024-02-24 22:47:36.000000
0xe50ed36a59f0  TCPv6   ::      3389    ::      0       LISTENING       744     svchost.exe     2024-02-24 22:47:36.000000
0xe50ed36a5730  TCPv4   0.0.0.0 3389    0.0.0.0 0       LISTENING       744     svchost.exe     2024-02-24 22:47:36.000000
```

Observe a connection established on port `3389` from the IP `192.168.182.139` with timestamp `2024-02-24 22:47:52.00` ; this could indicate an attacker's initial access.
A volatility plugin to use is `windows.pstree`, which will display a tree of the process running on the OS `vol -f memdump.mem windows.pstree`

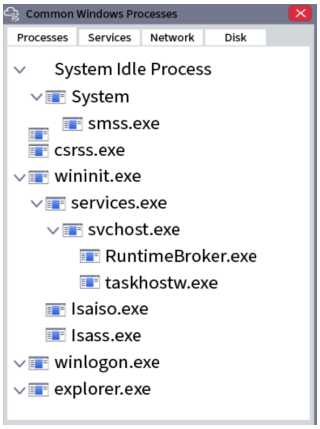
```
analyst@ip-10-10-219-197:~$ vol -f memdump.mem windows.pstree
Volatility 3 Framework 2.5.2
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime

4       0       System  0xe50ed3687040  150     -       N/A     False   2024-02-24 22:47:35.000000      N/A
* 312   4       smss.exe        0xe50ed68b0040  2       -       N/A     False   2024-02-24 22:47:35.000000      N/A
* 1600  4       MemCompression  0xe50ed379e280  50      -       N/A     False   2024-02-24 22:47:36.000000      N/A
* 92    4       Registry        0xe50ed36ed080  4       -       N/A     False   2024-02-24 22:47:31.000000      N/A
424     400     csrss.exe       0xe50ed67d7140  9       -       0       False   2024-02-24 22:47:35.000000      N/A
500     400     wininit.exe     0xe50ed7366080  2       -       0       False   2024-02-24 22:47:35.000000      N/A
* 664   500     lsass.exe       0xe50ed7360080  8       -       0       False   2024-02-24 22:47:35.000000      N/A
* 776   500     fontdrvhost.ex  0xe50ed7c69140  6       -       0       False   2024-02-24 22:47:35.000000      N/A
* 636   500     services.exe    0xe50ed73d3080  6       -       0       False   2024-02-24 22:47:35.000000      N/A
** 896  636     svchost.exe     0xe50ed7d112c0  9       -       0       False   2024-02-24 22:47:36.000000      N/A
** 1924 636     svchost.exe     0xe50ed73ab2c0  5       -       0       False   2024-02-24 22:47:36.000000      N/A
** 3464 636     svchost.exe     0xe50ed88e3080  7       -       1       False   2024-02-24 22:47:39.000000      N/A
** 7312 636     SecurityHealth  0xe50ed9af1280  10      -       0       False   2024-02-24 22:47:56.000000      N/A
** 2964 636     dllhost.exe     0xe50ed858d280  14      -       0       False   2024-02-24 22:47:37.000000      N/A
** 3348 636     svchost.exe     0xe50ed8b722c0  6       -       0       False   2024-02-24 22:47:39.000000      N/A
** 7060 636     WUDFHost.exe    0xe50ed9ad41c0  9       -       0       False   2024-02-24 22:47:53.000000      N/A
** 792  636     svchost.exe     0xe50ed7c85240  13      -       0       False   2024-02-24 22:47:35.000000      N/A
```

# Critical - Memory dump scenario via TryHackMe

## Volatility on Windows

The command provides information on processes hierarchically running on the system, indicating the process and their respective parent process. Services.exe is the parent process of dllhost.exe




One of the most common ways is to check the name of the process; threat actors commonly use names to try to disguise the execution. One of the ways to do this is to check that this process is not usually present

Observe a process with the truncated name critical_updat


This process does not look like it is part of the system, and observing in detail, it's the parent process of updater.exe, which is also not listed as a process part of the Windows OS.

### Questions

**Using the plugin "windows.netscan" can you identify the IP address that establish a connection on port 80?**

192.168.182.128

**Using the plugin "windows.netscan," can you identify the program (owner) used to access through port 80?**

msedge.exe

# Critical - Memory dump scenario via TryHackMe

## Volatility on Windows

**Analyzing the process present on the dump, what is the PID of the child process of critical_updat?**

1612

**What is the time stamp time for the process with the truncated name 'critical_updat'?**

2024-02-24 22:51:50.000000

## Finding Interesting Data

Investigate the process `critical_updat` which has a child process called `updater`
Start by looking at where on the disk it was saved; for that, use the plugin `windows.filescan` to examine the files accessed that are stored in the memory dump

Access the data in a better way, we will use the `>` character in bash to redirect the output to a file, in this case, `filescan_out`
`vol -f memdump.mem windows.filescan > filescan_out`

```
analyst@ip-10-10-68-174:~$ cat filescan_out |grep updater
0xe50ed736e8a0  \Users\user01\Documents\updater.exe     216
0xe50ed846fc60  \Program Files (x86)\Microsoft\EdgeUpdate\1.3.185.17\msedgeupdateres_en.dll 216
0xe50ed8482d10  \Program Files (x86)\Microsoft\EdgeUpdate\1.3.185.17\msedgeupdateres_en.dll 216
analyst@ip-10-10-68-174:~$
```

The files have been stored in the Directory `\Users\user01\Documents\updater.exe` or `C:\Users\user01\Documents\updater.exe`
Use the plugin `windows.mftscan.MFTScan`, whose output is also quite big, so we will redirect the output to the file `mftscan_out`

```
analyst@ip-10-10-68-174:~$ cat mftscan_out | grep updater
* 0xd389c63ce528        FILE    111417  2       File    Archive FILE_NAME       2024-02-24 22
:51:50.000000       2024-02-24 22:51:50.000000      2024-02-24 22:51:50.000000      2024-02-2
4 22:51:50.000000       updater[1].exe
analyst@ip-10-10-68-174:~$
```

`vol -f memdump.mem windows.mftscan.MFTScan > mftscan_out`
Use the grep command again to parse the file for the appearance of `updater.exe` `cat mftscan_out | grep updater`

### Getting the Goods

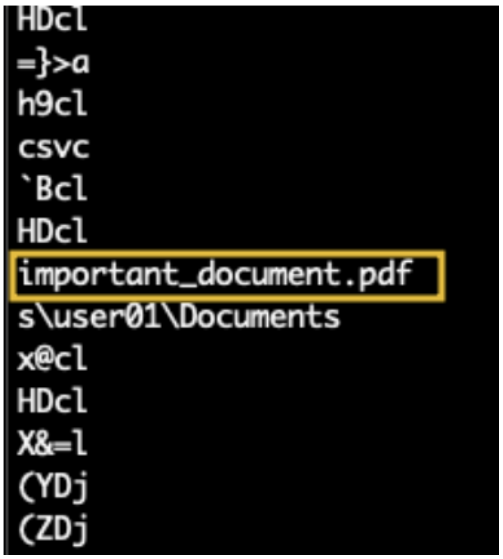
Dump the memory region corresponding to `updater.exe` , and examine it

```
PROCESSOR_IDENTIFIER=AMD64 Family 25 Model 97 Stepping 2, AuthenticAMD
hZG_
tN}frL
tN}frL
CommonProgramFiles(x86)=C:\Program Files (x86)\Common Files
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
h8H$
DriverData=C:\Windows\System32\Drivers\DriverData
8[G_
USERDOMAIN_ROAMINGPROFILE=DESKTOP-3NMNM0H
C:\Users\user01\Documents\updater.exe
WB0_
SB<_
_B8_
http://key.critical-update.com/encKEY.txt
LOCALAPPDATA=C:\Users\user01\AppData\Local
CommonProgramFiles=C:\Program Files\Common Files
ProgramFiles(x86)=C:\Program Files (x86)
```

# Critical - Memory dump scenario via TryHackMe

## Volatility on Windows

Use the plugin windows.memmap and specify the output dir with the -o switch
Use the same directory denoted by the character " . "and the option --dump followed by the option --pid and the PID of the process, which in the case of updater.exe is
vol -f memdump.mem -o . windows.memmap --dump --pid 1612 *to have a file with an extension .dmp in our working directory* strings pid.1612.dmp |less



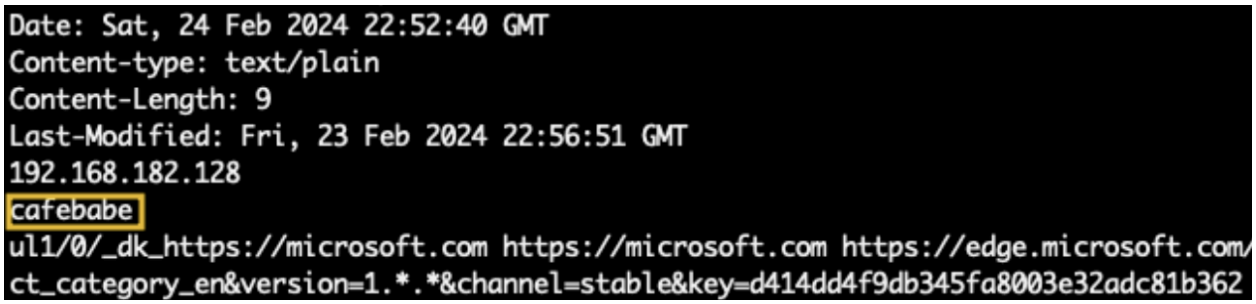

Look for key patterns like HTTP or key or any pattern that can lead us quickly to an artefact. Another way to scroll the terminal is by using the strings command piped to less to navigate through the output as the command output (cont.)

Immediately identified a possible key and a domain from a URL that the process may have accessed. Also, by scrolling down, we found more indications that this is a malicious process since we can find the important_document.pdf filename indicating an interaction with the file

The process updater.exe accessed the document important_document.pdf and accessed a "key" at some point in the URL http://key.critical-update.com/encKEY.txt

Use the command grep to look for the HTTP request that may be stored in memory, we can do it using -B and -A to look for 10 lines above and below our match to see if we can spot something else. strings pid.1612.dmp |grep -B 10 -A 10 "http://key.critical-update.com/encKEY.txt"


Observe at the end of the HTTP request the content of the file encKey.txt, and on the same request, we can observe data with the value cafebabe. This could be the key to encrypting the PDF used by the attacker that was not downloaded to disk.



# Critical - Memory dump scenario via TryHackMe

### Volatility on Windows

Questions

1. **Analyzing the "windows.filescan" output, what is the full path and name for critical_updat?**

C:\Users\user01\Documents\critical_update.exe

2. **Analyzing the "windows.mftscan.MFTScan" what is the Timestamp for the created date of important_document.pdf?**

2024-02-24 20:39:42.000000

3. **Analyzing the updater.exe memory output, can you observe the HTTP request and determine the server used by the attacker?**

SimpleHTTP/0.6 Python/3.10.4

## Conclusion

Learned how to gather information about the machine the dump belongs to, search for connections, enumerate and investigate processes, and examine the content for malicious patterns in a memory dump using tools such as Volatility3.