

Practical malware analysis

MALWARE.UNKNOWN.EXE.MALZ

DIRK F.

Table of contents

LIST OF FIGURES.....	2
EXECUTIVE SUMMARY	3
STATIC ANALYSIS	4
DYNAMIC ANALYSIS	7
YARA RULES	12
FINDINGS SUMMARY	12
OBSERVED INDICATORS OF COMPROMISE	13

List of figures

Figure 1 - File-type and -architecture identification.....	4
Figure 2 - PE-header analysis result from pedump.....	4
Figure 3 - Abbreviated floss output for the specimen.....	4
Figure 4 - Imports by the specimen as identified by peframe.....	5
Figure 5 - DetectItEasy analysis result.....	5
Figure 6 - De-obfuscation of strings contained in specimen.....	6
Figure 7 - Process Hacker output when running the specimen.....	7
Figure 8 - Process Monitor tree-view output.....	7
Figure 9 - Process Monitor tree for specimen with "cosmo.jpeg" present.....	8
Figure 10 - Specimen beaconing to hey[.]youup[.]local (Wireshark).....	8
Figure 11 - GET request details for the "beacon" traffic (Wireshark).....	9
Figure 12 - ProcessMonitor with DNS traffic and access to cosmo.jpeg.....	9
Figure 13 - fakends output for specimen running without access to the beacon-URL.....	10
Figure 14 - Wireshark capture of subdomain DNS resolution.....	10
Figure 15 – base64 decoded subdomain.....	10
Figure 16 - Modified "cosmo.jpeg" for verification.....	10
Figure 17 - Wireshark traffic running the specimen on updated "cosmo.jpeg".....	11
Figure 18 - base64 decoded subdomain request strings.....	11
Figure 19 - Indicators of compromise found during analysis.....	13

Executive summary

Sample name	Malware.unknown.exe.malz
SHA1 Hash	ec0d565afe635c2c7863b2a05df8a49c58b703a3
MD5 Hash	812a7c7eb9d7a4332b9e166aa09284d7
VirusTotal detection rate	4 detections (generic / CobaltStrike)
Source	GitHub Link
Operating system and architecture	Windows, 32Bit
Language Compiler	Nim Language (not packed)
Analysis date	13.02.2022
Analysis author	Dirk F.

The specimen examined for this report, was obtained from a GitHub address in .7z format. At the time of report creation, it has a very low detection rate on VirusTotal with only four generic detections (MalwareBytes flags it as CobaltStrike).

Static- and dynamic analysis, lead to the conclusion that the specimen's purpose is to exfiltrate data via DNS traffic. The malware tries to connect to a hard-coded domain and only proceeds with the exfiltration if the domain **can't be reached** ("kill-switch").

The only network traffic observed is to the kill-switch domain and to DNS for data exfiltration. No additional activities (downloading additional stages of malware, persistence mechanisms, process injection/spawning) has been observed during analysis.

Three Indicators of Compromise (one host based, two network based) have been observed that can help identify the specimen.

The next chapters will substantiate the findings above via static- and dynamic analysis.

Static analysis

File type and target operating system analysis confirm, the specimen targets Microsoft Windows operating systems and has been compiled for 32bit versions of Windows:

```
file
remnux@remnux:~/malware/PMAT$ file Malware.unknown.exe.malz
Malware.unknown.exe.malz: PE32 executable (GUI) Intel 80386 (stripped to external PDB), for MS
Windows
```

Figure 1 - File-type and -architecture identification

Looking at the PE header, no indications of packer usage are apparent

pedump										
=== SECTIONS ===										
NAME	RVA	VSZ	RAW_SZ	RAW_PTR	nREL	REL_PTR	nLINE	LINE_PTR	FLAGS	
.text	1000	19b14	19c00	400	0	0	0	0	60500060	R-X
.data	1b000	a0	200	1a000	0	0	0	0	c0600040	RW-
.rdata	1c000	33f0	3400	1a200	0	0	0	0	40600040	R--
.bss	20000	bff4	0	0	0	0	0	0	c0600080	RW-
.idata	2c000	7f4	800	1d600	0	0	0	0	c0300040	RW-
.CRT	2d000	34	200	1de00	0	0	0	0	c0300040	RW-
.tls	2e000	8	200	1e000	0	0	0	0	c0300040	RW-

Figure 2 - PE-header analysis result from pedump

String output (abbreviated for clarity) of *floss* shows several indicators of HTTP protocol usage from the nim language and a potential host based indicator “Desktop\cosmo.jpeg”. There are no strings that indicate capabilities to launch new processes or to inject into running processes.

The last three strings look like they’re obfuscated with repeating characters (w, x and B):

```
floss
[...]
@Can't load inet_ntop proc from Ws2_32.dll
@kernel32
@kernel32
@Ws2_32.dll
@Ws2_32.dll
@Bcrypt.dll
[...]
@Desktop\cosmo.jpeg
@200 OK
@Authorization
@Host
@httpclient.nim(1144, 15) `false`
@Transfer-Encoding
@Content-Type
@Content-Length
@httpclient.nim(1082, 13) `not url.contains({'\r', '\n'})` url shouldn't contain any newline
characters
@Nim httpclient/1.6.2
@hwtwtwpw:w/w/whwewyw.wywowuwupw.wlwowcwawlw
@axuxtzhx.xnxsx.xlxoxcxaxlx
@.BcBoBsBmBoBsBfBuBrBbBoBoBtBsBeBmBpBoBrBiBuBmB.BlBoBcBaBlB
[...]
```

Figure 3 - Abbreviated floss output for the specimen

PRACTICAL MALWARE ANALYSIS

The specimen only imports three DLLs – none of which hints at any network connectivity. Having identified this to be nim executable, nim's dynamic DLL loading capabilities need to be considered. The *floss* output also hints a string “@Ws2_32.dll” as a possible candidate.

peframe	

Import function	

KERNEL32.dll	23
msvcrt.dll	49
USER32.dll	1

Figure 4 - Imports by the specimen as identified by peframe

The string “@Bcrypt.dll” hints to encryption potentially being used by the specimen.

DetectItEasy confirms the nim language compiler assumptions:

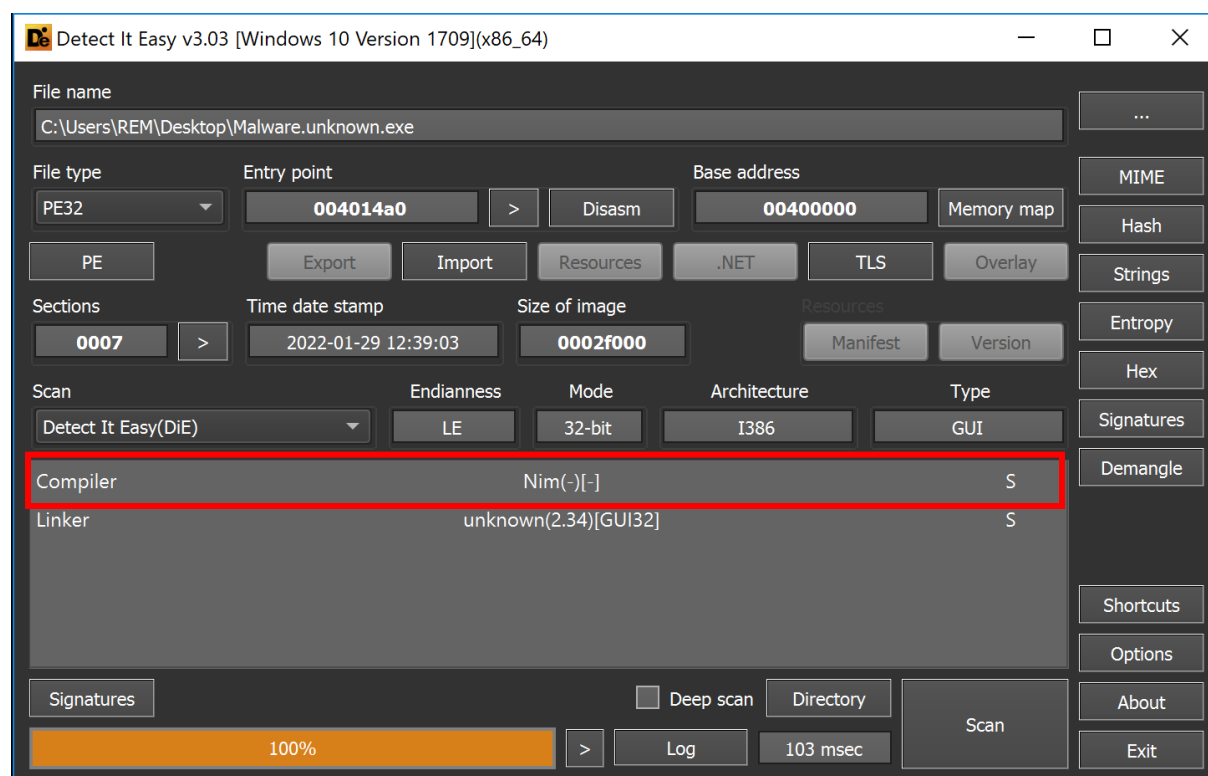


Figure 5 - DetectItEasy analysis result

Simple de-obfuscation of the last three strings from *floss* shows three hostnames:

- [http://hey\[.\]youup\[.\]local](http://hey[.]youup[.]local)
- Auth[.]ns[.]local
- [.]cosmofurbootsemporium[.]local (note the leading ".")

```
// Obfuscated strings (FLOSS output)
obfuscated_strings = ["hwtwtwpw:w/w/whwewyw.wywowuwuwpw.wlwowcwawlw",
                     "axuxtzhx.xnxsx.xlxoxcxaxlx",
                     ".BcBoBsBmBoBsBfBuBrBbBoBoBtBsBeBmBpBoBrBiBuBmB.BlBoBcBaBlB"]

// Obfuscation characters to remove
obfuscation_characters = ["w", "x", "B"]

// Removing obfuscation characters
[obfuscated_strings[i].replace(obfuscation_characters[i], "") for i in range(len(obfuscated_strings))]

// Clear-text urls
['http://hey.youup.local', 'auth.ns.local', '.cosmofurbootsemporium.local']
```

Figure 6 - De-obfuscation of strings contained in specimen

Conclusion:

Basic static analysis points to this being an unpacked nim 32bit Windows binary.

Initial working hypothesis is that the binary has no functionality to spawn or inject processes but potentially has networking capabilities (via runtime DLL loading).

The de-obfuscated strings point to HTTP server ([http://hey\[.\]youup\[.\]local](http://hey[.]youup[.]local)) being involved – the “auth[.]ns[.]local” hostname might be an indicator for DNS traffic to be of interest. This is further substantiated by the usage of the root-domain “[.]cosmofurbootsemporium[.]local”, which lacks the subdomain needed to complete the full hostname.

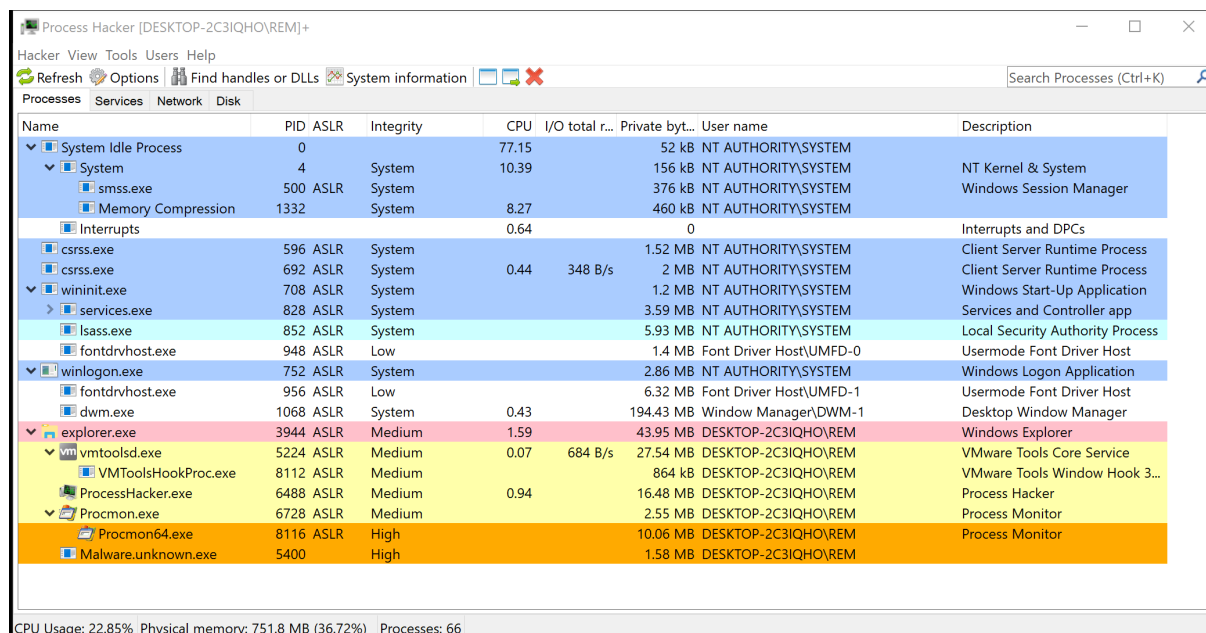
A filename pointing to a jpeg image (“cosmo.jpeg”) on the user’s desktop has been identified as a possible host-based indicator. At this point, it’s unclear, if that jpeg will be uploaded / downloaded or serves any other purpose.

To verify the hypothesis, and obtain more insights into the specimen’s behavior, basic dynamic analysis will be performed next.

PRACTICAL MALWARE ANALYSIS

Dynamic analysis

Running the specimen with administrative rights on a fresh snapshot of *Flare VM* shows the specimen running for about 15 seconds, without spawning any client processes. After 15 seconds it simply quits. The *FlareVM* has no internet access.



Process Hacker [DESKTOP-2C3IQHO\REM]+

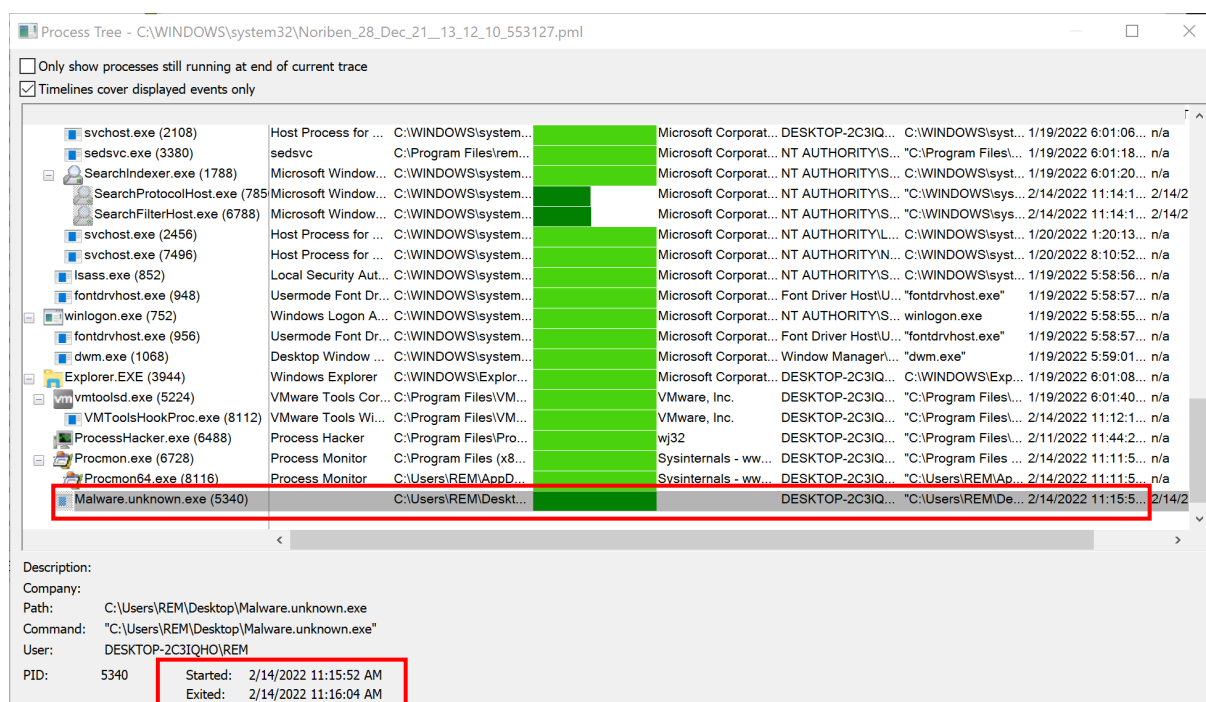
Refresh Options Find handles or DLLs System information Search Processes (Ctrl+K)

Name	PID	ASLR	Integrity	CPU	I/O total r...	Private byt...	User name	Description
System Idle Process	0			77.15		52 kB	NT AUTHORITY\SYSTEM	
System	4		System	10.39		156 kB	NT AUTHORITY\SYSTEM	NT Kernel & System
smss.exe	500	ASLR	System			376 kB	NT AUTHORITY\SYSTEM	Windows Session Manager
Memory Compression	1332		System	8.27		460 kB	NT AUTHORITY\SYSTEM	
Interrupts				0.64		0		Interrupts and DPCs
csrss.exe	596	ASLR	System			1.52 MB	NT AUTHORITY\SYSTEM	Client Server Runtime Process
csrss.exe	692	ASLR	System	0.44	348 B/s	2 MB	NT AUTHORITY\SYSTEM	Client Server Runtime Process
wininit.exe	708	ASLR	System			1.2 MB	NT AUTHORITY\SYSTEM	Windows Start-Up Application
services.exe	828	ASLR	System			3.59 MB	NT AUTHORITY\SYSTEM	Services and Controller app
lsass.exe	852	ASLR	System			5.93 MB	NT AUTHORITY\SYSTEM	Local Security Authority Process
fontdrvhost.exe	948	ASLR	Low			1.4 MB	Font Driver Host\UMFD-0	Usermode Font Driver Host
winlogon.exe	752	ASLR	System			2.86 MB	NT AUTHORITY\SYSTEM	Windows Logon Application
fontdrvhost.exe	956	ASLR	Low			6.32 MB	Font Driver Host\UMFD-1	Usermode Font Driver Host
dwm.exe	1068	ASLR	System	0.43		194.43 MB	Window Manager\DWDM-1	Desktop Window Manager
explorer.exe	3944	ASLR	Medium	1.59		43.95 MB	DESKTOP-2C3IQHO\REM	Windows Explorer
vmtoolsd.exe	5224	ASLR	Medium	0.07	684 B/s	27.54 MB	DESKTOP-2C3IQHO\REM	VMware Tools Core Service
VMToolsHookProc.exe	8112	ASLR	Medium			864 kB	DESKTOP-2C3IQHO\REM	VMware Tools Window Hook 3...
Process Hacker.exe	6488	ASLR	Medium	0.94		16.48 MB	DESKTOP-2C3IQHO\REM	Process Hacker
Procmon.exe	6728	ASLR	Medium			2.55 MB	DESKTOP-2C3IQHO\REM	Process Monitor
Procmon64.exe	8116	ASLR	High			10.06 MB	DESKTOP-2C3IQHO\REM	Process Monitor
Malware.unknown.exe	5400		High			1.58 MB	DESKTOP-2C3IQHO\REM	

CPU Usage: 22.85% Physical memory: 751.8 MB (36.72%) Processes: 66

Figure 7 - Process Hacker output when running the specimen

The process-tree view of *Process Monitor* confirms the 15 seconds runtime and the lack of any child-processes.



Process Tree - C:\WINDOWS\system32\Noriben_28_Dec_21_13_12_10_553127.pml

☐ Only show processes still running at end of current trace
☒ Timelines cover displayed events only

Process	Path	Company	Description	Started	Exited
svchost.exe (2108)	C:\WINDOWS\system32\svchost.exe	Microsoft Corporation	Host Process for ...	1/19/2022 6:01:06...	n/a
sedsvchost.exe (3380)	C:\Program Files\rem...	Microsoft Corporation	sedsvchost.exe	1/19/2022 6:01:18...	n/a
SearchIndexer.exe (1788)	C:\WINDOWS\system32\SearchIndexer.exe	Microsoft Corporation	Microsoft Window...	1/19/2022 6:01:20...	n/a
SearchProtocolHost.exe (785)	C:\WINDOWS\system32\SearchProtocolHost.exe	Microsoft Corporation	Microsoft Window...	2/14/2022 11:14:1...	2/14/2
SearchFilterHost.exe (6788)	C:\WINDOWS\system32\SearchFilterHost.exe	Microsoft Corporation	Microsoft Window...	2/14/2022 11:14:1...	2/14/2
svchost.exe (2456)	C:\WINDOWS\system32\svchost.exe	Microsoft Corporation	Host Process for ...	1/20/2022 1:20:13...	n/a
svchost.exe (7496)	C:\WINDOWS\system32\svchost.exe	Microsoft Corporation	Host Process for ...	1/20/2022 8:10:52...	n/a
lsass.exe (852)	C:\WINDOWS\system32\lsass.exe	Microsoft Corporation	Local Security Aut...	1/19/2022 5:58:56...	n/a
fontdrvhost.exe (948)	C:\WINDOWS\system32\fontdrvhost.exe	Microsoft Corporation	Usermode Font Dr...	1/19/2022 5:58:57...	n/a
winlogon.exe (752)	C:\WINDOWS\system32\winlogon.exe	Microsoft Corporation	Windows Logon A...	1/19/2022 5:58:55...	n/a
fontdrvhost.exe (956)	C:\WINDOWS\system32\fontdrvhost.exe	Microsoft Corporation	Usermode Font Dr...	1/19/2022 5:58:57...	n/a
dwm.exe (1068)	C:\WINDOWS\system32\dwm.exe	Microsoft Corporation	Desktop Window ...	1/19/2022 5:59:01...	n/a
Explorer.EXE (3944)	C:\WINDOWS\Explorer.exe	Microsoft Corporation	Windows Explorer	1/19/2022 6:01:08...	n/a
vmtoolsd.exe (5224)	C:\Program Files\VMware, Inc.\vmtoolsd.exe	VMware, Inc.	VMware Tools Cor...	1/19/2022 6:01:40...	n/a
VMToolsHookProc.exe (8112)	C:\Program Files\VMware, Inc.\VMToolsHookProc.exe	VMware, Inc.	VMware Tools Wi...	2/14/2022 11:12:1...	n/a
Process Hacker.exe (6488)	C:\Program Files\Process Hacker\Process Hacker.exe	wj32	Process Hacker	2/11/2022 11:44:2...	n/a
Procmon.exe (6728)	C:\Program Files\Sysinternals - ww... \Procmon.exe	Sysinternals - ww...	Process Monitor	2/14/2022 11:11:5...	n/a
Procmon64.exe (8116)	C:\Users\REM\AppData\Local\Temp\Procmon64.exe	Sysinternals - ww...	Process Monitor	2/14/2022 11:11:5...	n/a
Malware.unknown.exe (5340)	C:\Users\REM\Desktop\Malware.unknown.exe		Malware.unknown.exe	2/14/2022 11:15:5...	2/14/2

Description:
Company:
Path: C:\Users\REM\Desktop\Malware.unknown.exe
Command: "C:\Users\REM\Desktop\Malware.unknown.exe"
User: DESKTOP-2C3IQHO\REM
PID: 5340
Started: 2/14/2022 11:15:52 AM
Exited: 2/14/2022 11:16:04 AM

Figure 8 - Process Monitor tree-view output

PRACTICAL MALWARE ANALYSIS

So far, neither the “Desktop\cosmo.jpeg”, nor the two hostnames (http://hey[.]youup[.]local, [.]cosmosfurbootsemporium[.]local) have been accounted for.

Step 1 (FlareVM without network access):

- Add a file named “cosmo.jpeg” to the desktop
- Re-run the specimen with administrative rights (still no internet access)
- Fille the cosmo.jpeg file with a series of “a” characters for easier recognition in network traffic, etc.

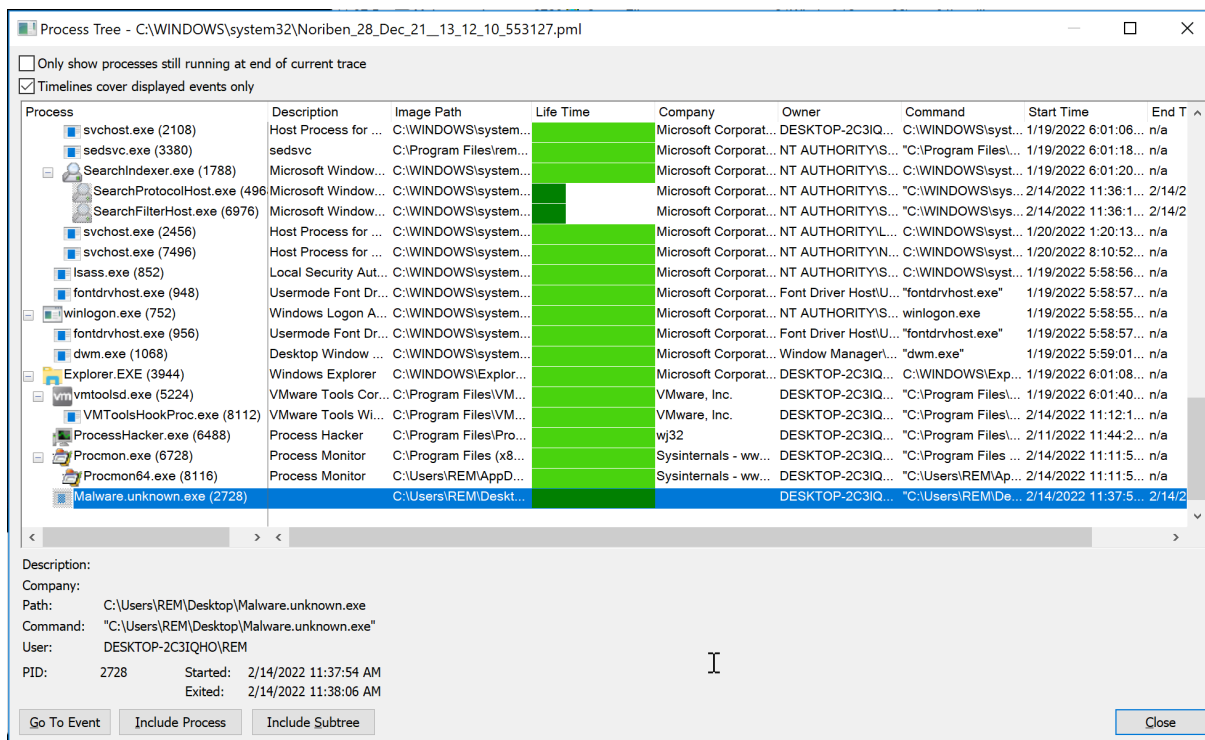


Figure 9 - Process Monitor tree for specimen with “cosmo.jpeg” present

- Runtime reduced to about 12 seconds, then the specimen exits
- No child processes spawned
- Contents of “cosmo.jpeg” unchanged after running the specimen again

Step 2 (FlareVM connected to REMnux VM for simulated internet access – HTTP, DNS):

- Enabling *inetsim* (including DNS) on REMnux to simulate network connectivity

No.	Time	Source	Destination	Protocol	Length	Info
12	2022-02-14 12:00:20	305077672 192.168.23.128	192.168.23.129	TCP	62	53049 → 80 [ACK] Seq=1 Ack=1 Win=525568 Len=0
13	2022-02-14 12:00:20	305148856 192.168.23.128	192.168.23.129	HTTP	155	GET / HTTP/1.1
14	2022-02-14 12:00:20	305153324 192.168.23.128	192.168.23.129	TCP	56	80 → 53049 [ACK] Seq=1 Ack=100 Win=64256 Len=0
15	2022-02-14 12:00:20	314832491 192.168.23.128	192.168.23.129	TCP	206	80 → 53049 [PSH, ACK] Seq=1 Ack=100 Win=64256 Len=150 [TCP segment of a reassembled PDU]
16	2022-02-14 12:00:20	316355152 192.168.23.128	192.168.23.129	HTTP	314	HTTP/1.1 200 OK (text/html)
17	2022-02-14 12:00:20	316701499 192.168.23.128	192.168.23.129	TCP	62	53049 → 80 [ACK] Seq=100 Ack=410 Win=525056 Len=0
18	2022-02-14 12:00:23	335217925 192.168.23.128	192.168.23.129	TCP	68	53050 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
19	2022-02-14 12:00:23	335241847 192.168.23.128	192.168.23.129	TCP	68	80 → 53050 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
20	2022-02-14 12:00:23	335644724 192.168.23.128	192.168.23.129	TCP	62	53050 → 80 [ACK] Seq=1 Ack=1 Win=525568 Len=0
21	2022-02-14 12:00:23	335644809 192.168.23.128	192.168.23.129	HTTP	155	GET / HTTP/1.1
22	2022-02-14 12:00:23	335673509 192.168.23.128	192.168.23.129	TCP	56	80 → 53050 [ACK] Seq=1 Ack=100 Win=64256 Len=0
23	2022-02-14 12:00:23	345316323 192.168.23.128	192.168.23.129	TCP	206	80 → 53050 [PSH, ACK] Seq=1 Ack=100 Win=64256 Len=150 [TCP segment of a reassembled PDU]
24	2022-02-14 12:00:23	346965050 192.168.23.128	192.168.23.129	HTTP	314	HTTP/1.1 200 OK (text/html)
25	2022-02-14 12:00:23	347319570 192.168.23.128	192.168.23.129	TCP	62	53050 → 80 [ACK] Seq=100 Ack=410 Win=525056 Len=0
26	2022-02-14 12:00:26	351511812 192.168.23.128	192.168.23.129	TCP	68	53051 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
27	2022-02-14 12:00:26	351539244 192.168.23.128	192.168.23.129	TCP	68	80 → 53051 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
28	2022-02-14 12:00:26	351874891 192.168.23.128	192.168.23.129	TCP	62	53051 → 80 [ACK] Seq=1 Ack=1 Win=525568 Len=0
29	2022-02-14 12:00:26	352051969 192.168.23.128	192.168.23.129	HTTP	155	GET / HTTP/1.1
30	2022-02-14 12:00:26	352060343 192.168.23.128	192.168.23.129	TCP	56	80 → 53051 [ACK] Seq=1 Ack=100 Win=64256 Len=0
31	2022-02-14 12:00:26	362158273 192.168.23.128	192.168.23.129	TCP	206	80 → 53051 [PSH, ACK] Seq=1 Ack=100 Win=64256 Len=150 [TCP segment of a reassembled PDU]
32	2022-02-14 12:00:26	364484585 192.168.23.128	192.168.23.129	HTTP	314	HTTP/1.1 200 OK (text/html)
33	2022-02-14 12:00:26	364885845 192.168.23.128	192.168.23.129	TCP	62	53051 → 80 [ACK] Seq=100 Ack=410 Win=525056 Len=0

Figure 10 - Specimen beaconing to hey[.]youup[.]local (Wireshark)

PRACTICAL MALWARE ANALYSIS

- The network traffic consists of the same number of bytes transmitted every three seconds – the request header doesn't contain any data other than a GET request to hey[.]youup[.]local:

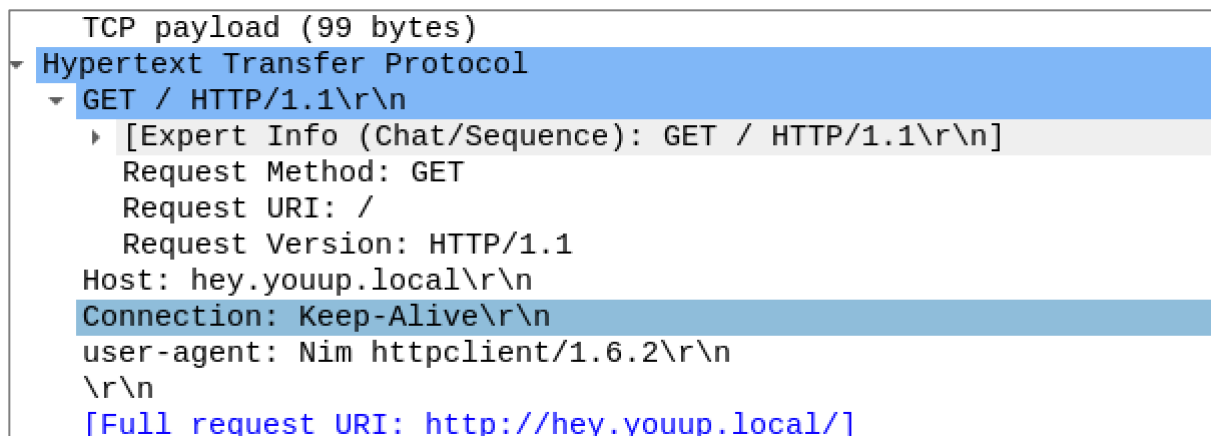


Figure 11 - GET request details for the "beacon" traffic (Wireshark)

- The specimen continues to run until terminated via *ProcessHacker*
- "cosmo.jpeg" remains unchanged while the specimen runs
- No DNS traffic to auth[.]ns[.]local has been observed
- No child processes have been spawned

Step 3 (Remove HTTP connectivity, continue with DNS only):

- Restart *Inetsim* on *REMnux* without HTTP/HTTPS responses configured
- Run *fakedns* on *REMnux* to monitor DNS traffic
- Flush DNS resolver cache on *FlareVM*
- The specimen now creates traffic on port 53 – which indicates DNS traffic

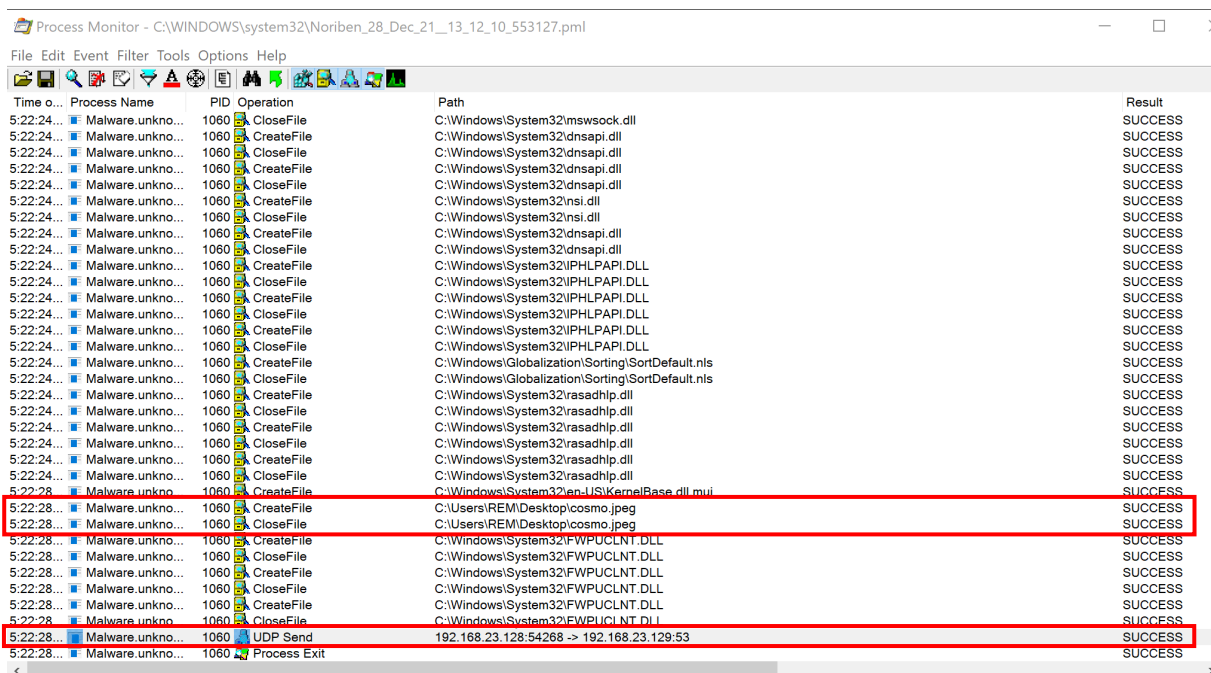


Figure 12 - ProcessMonitor with DNS traffic and access to cosmo.jpeg

PRACTICAL MALWARE ANALYSIS

- The specimen exits after a few seconds without spawning any sub-processes
- Observing DNS traffic captured by *fakedns* shows a DNS lookup for a subdomain of “[.]cosmosfurbootsemporium[.]local”:

```
remnux@remnux:~/malware/PMAT$ sudo fakedns
fakedns[INFO]: dom.query. 60 IN A 192.168.23.129
fakedns[INFO]: Response: win1710.ipv6.microsoft.com -> 192.168.23.129
fakedns[INFO]: Response: hey.youup.local -> 192.168.23.129
fakedns[INFO]: Response: YWFhYWZhYWZhYWZhYWZhYWE=.cosmosfurbootsemporium.local -> 192.168.23.129
```

Figure 13 - *fakedns* output for specimen running without access to the beacon-URL

- Wireshark shows the unsuccessful attempt to connect to hey[.]youup[.]local (1)
- Wireshark also shows the DNS traffic to resolve a domain named “YWFhYWZhYWZhYWZhYWZhYWE=[.]cosmosfurbootsemporium[.]local” (2)

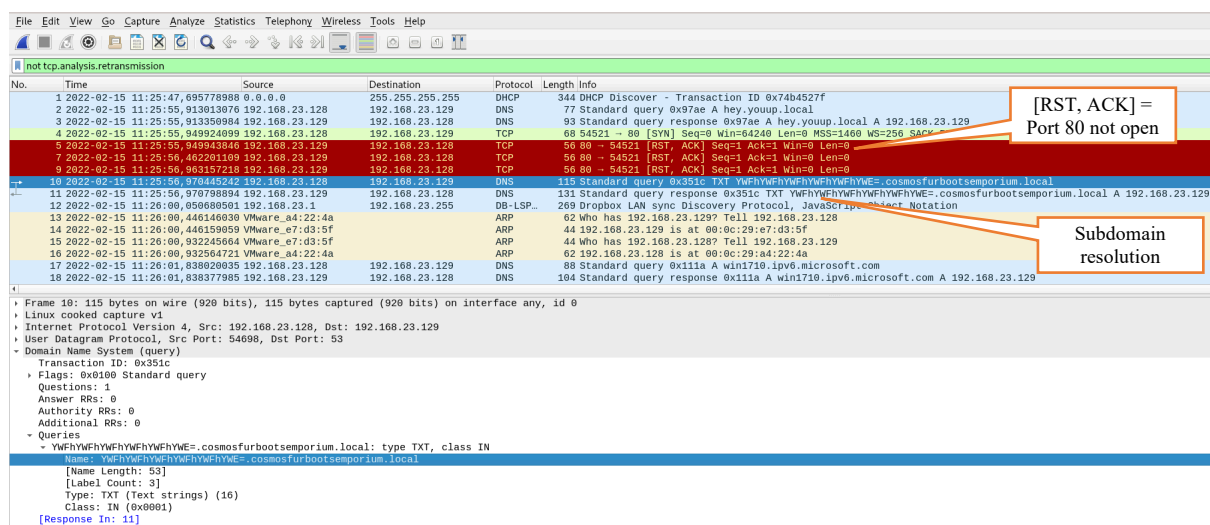


Figure 14 - Wireshark capture of subdomain DNS resolution

- The name of the subdomains is base64 encoded and decodes to “aaaaaaaaaaaa” – which is the content of our fictitious “cosmo.jpeg” file

```
base64 -d
remnux@remnux:~/malware/PMAT$ echo "YWFhYWZhYWZhYWZhYWZhYWE=" | base64 -d
aaaaaaaaaaaaaaaaaaaa
```

Figure 15 – base64 decoded subdomain

- The contents of the “cosmo.jpeg” file remain unchanged
- To assess if the base64 encoded “a...a” string actually is the content of the “cosmo.jpeg”, an alternative version of “cosmo.jpeg” will be used for a re-run of the specimen:

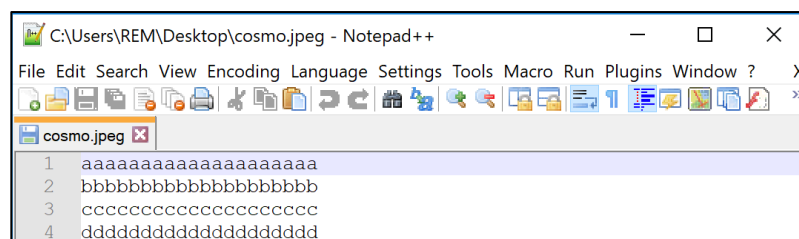


Figure 16 - Modified “cosmo.jpeg” for verification

- [illegible]

Figure 17 - Wireshark traffic running the specimen on updated "cosmo.jpeg"

- ```
base64 -d
```
- ```
remnux@remnux:~/malware/PMAT$ echo  
"YWFhYWVhYWFhYWFhYWFhYWFhYWENCmJiYmJiYmJiYmJiYmJiYmJiDQpjY2NjY2NjY2NjY2NjY2NjY2NjYw0KZGRkZGRkZGRkZGRkZGRkZGRkZGQ=" | base64 -d  
aaaaaaaaaaaaaaaaaaaaaa  
bbbbbbbbbbbbbbbbbbbbb  
ccccccccccccccccccccc  
Figure 18 – base64 decoded subdomain request strings
```

Observed behavior during dynamic analysis:

- The specimen exfiltrates data from a file (with hard-coded filename) via DNS traffic
- Exfiltration happens via resolution of subdomains of the `"[.]cosmofurbootsemporium[.]local"`
- The data is base64 encoded for transmission and broken up into a separate chunk after 62 characters
- No additional processes have been observed in *ProcessMonitor* or *ProcessHacker*
- The specimen checks for a successful TCP connection to [http://hey\[.\]youup\[.\]local](http://hey[.]youup[.]local) and **if it can connect on port 80, does not exfiltrate data**

YARA Rules

Based on the findings outlined above, the following Yara rule can be used to detect the specimen:

```
rule malware_unknown {
  meta:
    description = "Detects the Malware.Unknown.exe provided as part of the PMAT course"
    md5 = "812a7c7eb9d7a4332b9e166aa09284d7"
    sha1 = "ec0d565afe635c2c7863b2a05df8a49c58b703a3"
    filename = "Malware.unknown.exe.malz"
    author = "Dirk F."

    Block = true
    Log = true
    Quarantine = false

  strings:
    $PE_Magic_Byte = "MZ"

    $malware_user_agent = "httpclient/1.6.2"
    $malware_exfil_file = "Desktop\\cosmo.jpeg"
    $malware_kill_switch_url = "hwtwtwpw:w/w/whwewyw.wywowuwupw.wlwowcwawlw"
    $malware_exfil_domain = ".BcBoBsBmBoBsBfBuBrBbBoBoBtBsBeBmBpBoBrBiBuBmB.BlBoBcBaBlB"

  condition:
    $PE_Magic_Byte at 0 and
    all of ($malware*)
}
```

Findings summary

- The specimen is coded in the nim language (32 bit)
- The malware sample performs DNS exfiltration
- The malware exfiltrates a hard-coded picture, named "cosmo.jpeg" from the user's desktop
- The specimen contacts a webserver at http://hey[.]youup[.]local
- If a connection to the webserver URL is successful, the malware **does not execute**, effectively making this a WannaCry-like kill switch
- If the kill switch URL isn't reachable, the malware exfiltrates the content of "cosmo.jpeg" in base64 encoded chunks by trying to resolve the base64 encoded text as subdomain of "[.]cosmofurbootsemporium[.]local"
- To recreate the original file content of "cosmo.jpeg", the DNS server listening on auth[.]ns[.]local would have to concatenate all subdomain-requests for "[.]cosmofurbootsemporium[.]local" and base64-decode the final text into an image file

Observed indicators of compromise

Type of ICO	Indicator	Description
Host-based	Desktop\cosmo.jpeg	File being exfiltrated via DNS
Network-based	http://hey[.]youup[.]local	“Kill-Switch” domain
Network-based	auth[.]ns[.]local	DNS Server for subdomain resolution
Network-based	[.]cosmofurbootsemporium[.]local	Domain-name used for data exfiltration

Figure 19 - Indicators of compromise found during analysis