MALWARE ANALYSIS
BY NASHIB

Contents

Tools:

REMnux, WINXP, PEstudio, yara-rules, apateDNS, CFF explorer: referenced at end.

Introduction

This report presents a comprehensive malware analysis. The purpose of this analysis is to deepen understanding and demonstrate practical skills in identifying, dissecting, and mitigating potential threats posed by malicious software. This document encapsulates the methodologies applied in both static and dynamic analysis phases, providing a detailed examination of various malware samples to uncover their behaviours, impacts, and the mechanisms they employ to evade detection.

Scope of the Report

The report is structured into two primary parts, each addressing specific tasks and tools used in the analysis process:

Part A: Focuses on the initial analysis of PDF samples and executable files, using tools like REMnux, PDFid, and PeepPDF to identify characteristics indicative of malware. This section lays the groundwork by discussing the setup of analysis environments, preliminary scans, and detailed examination of file structures and embedded scripts.

Part B: Delves into dynamic analysis, illustrating the malware's behaviour in a controlled environment. It covers network interactions, registry manipulations, and the malware's attempts to communicate with external servers. Tools such as PROCMON, REGSHOT, and FAKENET are employed to monitor and log the malware’s actions in real time.

Analysis Tools and Techniques

Throughout the report, various industry-standard tools are utilized to dissect and analyse the malware samples. These tools include, but are not limited to, YARA for rule-based analysis, VirusTotal for hash checking, and custom Python scripts for deeper insights. Each tool's role and contribution to the analysis are described, showcasing how they integrate to form a comprehensive malware investigation toolkit.

Objectives

The objectives of this analysis are to:

Equip the reader with the knowledge to perform detailed malware analysis.

Demonstrate the application of various cybersecurity tools in real-world scenarios.

Provide a methodological approach to understanding and documenting the behaviour of malicious software.

Conclusion

The findings and methodologies documented in this report are intended to contribute to the broader knowledge base of malware analysis within the cybersecurity community. By detailing the steps and tools used, this report serves as a resource for educators, students, and practitioners alike, aiming to enhance the collective capabilities in combating cybersecurity threats.

PART A

TASK 1 –

CW_PDF_SAMPLE1.PDF

I first began with installation and set-up of my Remnux machine and all the necessary items such as 7z then proceeded to transfer the file from my pc to unzip.

```
remnux@remnux:~/Documents/task1$ 7z x cw_pdf_files.7z -pinfected

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,1 CPU AMD Ryzen 7 3700X 8-Core
 Processor              (870F10),ASM,AES-NI)

Scanning the drive for archives:
1 file, 224078 bytes (219 KiB)

Extracting archive: cw_pdf_files.7z
--
Path = cw_pdf_files.7z
Type = 7z
Physical Size = 224078
Headers Size = 206
Method = LZMA2:384k 7zAES
Solid = +
Blocks = 1

Everything is Ok

Files: 2
Size:       289383
Compressed: 224078
```

Figure 1- Unzipped cw_pdf_files.

I did a preliminary scan using pdfid, to look for keywords etc... and found that '...sample1.pdf' does not contain any obvious signs of being a malicious document.

```
remnux@remnux:~/Documents/task1$ pdfid.py
PDFiD 0.2.8 cw_pdf_sample1.pdf
 PDF Header: %PDF-1.4
 obj                    86
 endobj                 86
 stream                 50
 endstream              50
 xref                    2
 trailer                 2
 startxref               2
 /Page                   3
 /Encrypt                0
 /ObjStm                 0
 /JS                     0
 /JavaScript             0
 /AA                     0
 /OpenAction             0
 /AcroForm               0
 /JBIG2Decode            0
 /RichMedia              0
 /Launch                 0
 /EmbeddedFile           0
 /XFA                    0
 /URI                    0
 /Colors > 2^24          0
```

Figure 2- pdfid sample1.pdf.

There is no detected JS, AA/OpenAction object which are all used for executing actions automatically upon opening the pdf, nor are there any /Launch or /EmbeddedFile detected for executables.

Since the output suggests no suspicious elements, I then used peepdf:

```
remnux@remnux:~/Documents/task1$ peepdf -i cw_pdf_sample1.pdf
Warning: PyV8 is not installed!!

File: cw_pdf_sample1.pdf
MD5: c30c96c9d9e9bf9a454ab0b8fb754b14
SHA1: 05433eccfea1c825b760a415d1ef432eadfb0c74
SHA256: dc3f10f2d8123ea1317c716a028cd2ff96b3d982243ae6564d1ef3c51976a85c
Size: 139996 bytes
Version: 1.4
Binary: True
Linearized: True
Encrypted: False
Updates: 1
Objects: 86
Streams: 50
URIs: 0
Comments: 0
Errors: 0

Version 0:
        Catalog: 26
        Info: 24
        Objects (1): [25]
        Streams (0): []

Version 1:
        Catalog: No
        Info: No
        Objects (85): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 5
1, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86]
        Streams (50): [86, 40, 42, 43, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 66, 67, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 3, 4, 5, 6, 7, 8, 13, 15, 16, 17, 18
, 23]
                Encoded (43): [86, 40, 42, 43, 48, 49, 50, 51, 52, 53, 54, 55, 56, 58, 59, 60, 62, 63, 66, 67, 70, 71, 72, 74, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 3, 5, 7, 8, 13, 15, 16, 17, 18]
```

'..sample1.pdf' does not show any obvious signs of being malicious, the file is not encrypted; no detected URIs to download content; no detected errors in the file structure; the file has been linearised meaning that it is optimised for web viewing.

Further analysis leads me to believe 'cd_pdf_sample1.pdf' is not a malicious document as pdf-parser also fails to return any suspicious elements.

```
remnux@remnux:~/Documents/task1$ pdf-parser.py --search keyword cw_pdf_sample1.pdf
remnux@remnux:~/Documents/task1$ pdf-parser.py --search javascript --search aa --search openaction cw_pdf_sample1.pdf
remnux@remnux:~/Documents/task1$ pdf-parser.py --object 15 cw_pdf_sample1.pdf
obj 15 0
 Type: /XObject
 Referencing:
 Contains stream

  <<
    /Subtype /Image
    /Length 23
    /Filter /FlateDecode
    /ImageMask true
    /BitsPerComponent 1
    /Width 192
    /Height 6
    /Type /XObject
  >>
```

Figure 3- pdf-parser sample1.pdf

CW_PDF_SAMPLE2.PDF

Using YARA to analyse sample2.pdf found multiple matched rules meaning further investigation is required, but one of the more troubling matches is the 'vmdetect' matched rule, where the file is possibly trying to detect virtualised environments.

```
remnux@remnux:~/Documents/task1$ yara -w -msg ~/Documents/task1/yara-rules/rules-master/index.yar cw_pdf_sample2.pdf
vmdetect [] [author="nex",description="Possibly employs anti-virtualization techniques"] cw_pdf_sample2.pdf
0x24139:$vmware_mac_2c: 005056
Big_Numbers1 [] [author="_pusher_",description="Looks for big numbers 32:sized",date="2016-07"] cw_pdf_sample2.pdf
0xbfc:$c0: 5B850F96C4FCDF11B406A67D192E4400
0xda3:$c0: 5C850F96C4FCDF11B406A67D192E4400
0xf52:$c0: 5D850F96C4FCDF11B406A67D192E4400
0x10f9:$c0: 5E850F96C4FCDF11B406A67D192E4400
0x1434:$c0: 1F7F02AB2E03DC11B6D7BD4D5F2A0D7A
0x2257c:$c0: 1F7F02AB2E03DC11B6D7BD4D5F2A0D7A
```

Figure 4- yara index.yar sample2.pdf

Moving onto peepdf found multiple suspicious elements to investigate such as JS and an embedded file.

```
remnux@remnux:~/Documents/task1$ peepdf -if cw_pdf_sample2.pdf
Warning: PyV8 is not installed!!

File: cw_pdf_sample2.pdf
MD5: 15d8b554bc3e87889c3199c4faa82d48
SHA1: 8b6e1fcad823d24c8b38a61d2a10c617ed2a8976
SHA256: de059b7b16f38bb115e3bd14cd29e258c028020c24ced2d0b561bca0769522
Size: 149387 bytes
Version: 1.6
Binary: False
Linearized: True
Encrypted: False
Updates: 0
Objects: 146
Streams: 55
URIs: 0
Comments: 0
Errors: 0

Version 0:
        Catalog: 8
        Info: 6
        Objects (146): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,
50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67
02, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 1
4, 145, 146]
                Errors (1): [144]
        Streams (55): [5, 17, 19, 20, 22, 24, 25, 35, 37, 38, 40, 42,
123, 126, 128, 131, 133, 136, 138, 140, 143, 144]
                Encoded (37): [52, 62, 66, 68, 69, 70, 71, 72, 73, 74,
        Objects with JS code (3): [141, 142, 144]
        Suspicious elements:
                /AcroForm (2): [8, 144]
                /Names (2): [4, 8]
                /XFA (1): [144]
                /AA (2): [11, 48]
                /JS (2): [141, 142]
                /JavaScript (2): [141, 142]
                /EmbeddedFiles: [8]
                /EmbeddedFile: [144]
```

Figure 5- peepdf

Hence using pdf-parser.py cw_pdf_sample2.pdf, I was able to look through the entire document and find obj 142 showing that is a JS function for date formatting. However, also finding items of note with object 144/5/6 which has an embedded file.

```
remnux@remnux:~/Documents/task1$ pdf-parser.py --object 142 -f -w cw_pdf_sample2.pd
obj 142 0
 Type:
 Referencing:
  <<</JS (AFDate_KeystrokeEx\("mm/dd/yyyy"\);) /S /JavaScript >>

  <<
    /JS '(AFDate_KeystrokeEx\\("mm/dd/yyyy"\\);)'
    /S /JavaScript
  >>

  <<</JS (AFDate_KeystrokeEx\("mm/dd/yyyy"\);) /S /JavaScript >>
```

Figure 6 – javascript.

Moving onto the embedded file I was able to find suspicious obfuscated code which may suggest an attempt to perform malicious actions.

```
remnux@remnux:~/Documents/task1$ pdf-parser.py --object 144 --filter --raw cw_pdf_sample2.pdf > embedded_file.bin
remnux@remnux:~/Documents/task1$ file extracted_embedded_file.bin
extracted_embedded_file.bin: PDF document, version 1.6, ASCII text, with very long lines
remnux@remnux:~/Documents/task1$ strings extracted_embedded_file.bin
obj 144 0
 Type: /EmbeddedFile
 Referencing:
 Contains stream
  <<
    /Length 4493
    /Type /EmbeddedFile
    /Filter /FlateDecode
  >>
 b'%PDF-1.6\n%\xe2\xe3\xcf\xd3\n1 0 obj\n<</MediaBox [0 0 1 1] /Type/Page /Contents 3 0 R /Parent 5 0 R>>\nendobj\n5
:xdp="http://ns.adobe.com/xdp/">\n\n\r\n<config><present>\r\n<pdf><interactive>1</interactive>\r\n<version test2=\'a
fa.org/schema/xfa-template/2.5/\'>\r\n<subform name="a1"><asd/>\r\n\t\t<subform name="v236536b346">\r\n\t\t<field qw
<script contentTyp=\'application\'\r\ncontentType=\'application/x-javascript\'>\r\n\r\nif((String+\'\').substr(1,4)=
t;+I}*/DkR%x-W[]mCj^?:LBKQYEUqFM\';\r\nl=\'l\';\r\ne=e()[((2+3)&#63;\'e\'+\'v\':"")+"a"+l];\r\ns=[];\r\na=\'pus\'+\'
;s[a](z);z=c[\'s\'+"ubstr"](3,1);s[a](z);z=c[\'s\'+"ubstr"](4,1);s[a](z);z=c[\'s\'+"ubstr"](5,1);s[a](z);z=c[\'s\'+"
"ubstr"](9,1);s[a](z);z=c[\'s\'+"ubstr"](10,1);s[a](z);z=c[\'s\'+"ubstr"](11,1);s[a](z);z=c[\'s\'+"ubstr"](12,1);s[a
;s[a](z);z=c[\'s\'+"ubstr"](14,1);s[a](z);z=c[\'s\'+"ubstr"](12,1);s[a](z);z=c[\'s\'+"ubstr"](15,1);s[a](z);z=c[\'s\
's\'+"ubstr"](12,1);s[a](z);z=c[\'s\'+"ubstr"](9,1);s[a](z);z=c[\'s\'+"ubstr"](1,1);s[a](z);z=c[\'s\'+"ubstr"](18,1
```

Figure 7- suspicious obfuscation

Conclusion

Samepl1.pdf contains no suspicious elements, is not encrypted and other elements indicate sample1 is not a malicious document.

Sample2.pdf contains several malicious elements such as obfuscation of javascript, this alongside my last check on virus total, makes me believe it is a malicious file:

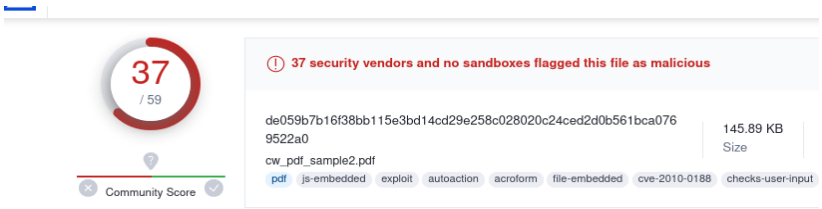


Figure 8- VirusTotal Sample2.pdf

TASK 2:

a) Using 'file unknown.file' I was able to determine it being a 32-bit executable for windows, further backed up by using trid on windows xp to see that it is a windows32 executable.

```
remnux@remnux:~/Documents/task2$ file unknown.file
unknown.file: PE32 executable (console) Intel 80386, for MS Windows, UPX compressed
remnux@remnux:~/Documents/task2$
```

Figure 9- file unknown.file.

```
C:\Documents and Settings\admin\My Documents\Part A\task2>"C:\Documents and Sett
ings\admin\Desktop\Tools\trid\trid.exe" unknown.file

TrID/32 - File Identifier v2.10 - (C) 2003-11 By M.Pontello
Definitions found:  5405
Analyzing...

Collecting data from file: unknown.file
 42.3% (.EXE) UPX compressed Win32 Executable (30569/9/7)
 36.7% (.EXE) Win32 EXE Yoda's Crypter (26569/9/4)
  9.1% (.DLL) Win32 Dynamic Link Library (generic) (6578/25/2)
  6.2% (.EXE) Win32 Executable (generic) (4508/7/1)
  2.7% (.EXE) Generic Win/DOS Executable (2002/3)
```

Figure 10- Trid.exe win32

To execute it for analysis would involve changing the file extension to .exe, however, for the purpose of this task there is no need for further action beside unzipping it from unknown.7z.

b) We know that the sample is packed already due to the previous information from 42.3% being compressed win32 but I investigated it using below as well.

```
C:\Documents and Settings\admin\My Documents\Part A\task2>"C:\Documents and Sett
ings\admin\Desktop\Tools\upx\upx391w\upx.exe" unknown.file -o unknown_unpacked.f
ile
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2013
UPX 3.91w        Markus Oberhumer, Laszlo Molnar & John Reiser   Sep 30th 2013

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
upx: unknown.file: AlreadyPackedException: already packed by UPX

Packed 1 file: 0 ok, 1 error.
```

Figure 11- packed evidence 1

```
remnux@remnux:~/Documents/task2$ upx -t unknown.file
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2020
UPX 3.96         Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

testing unknown.file [OK]
```

Figure 12- packed evidence 2

Other characteristics that make it likely to be a packed malware file include high entropy code scoring 7.8 out of 8.

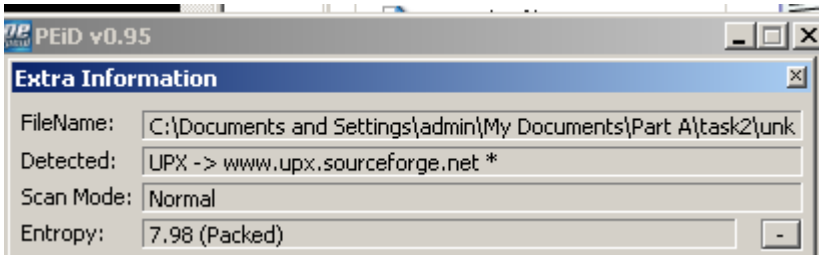


Figure 13- packed evidence 3 - high entropy

In line with the previous findings, there is also a significant difference between virtual size and raw data size within the UPX0 section, further suggesting a packed file.

Figure 14- packed evidence 4 - virtual size vs raw size

Doing further investigation using strings.exe to output into a file shows that there is a lack of real strings, further evidencing a packed file.

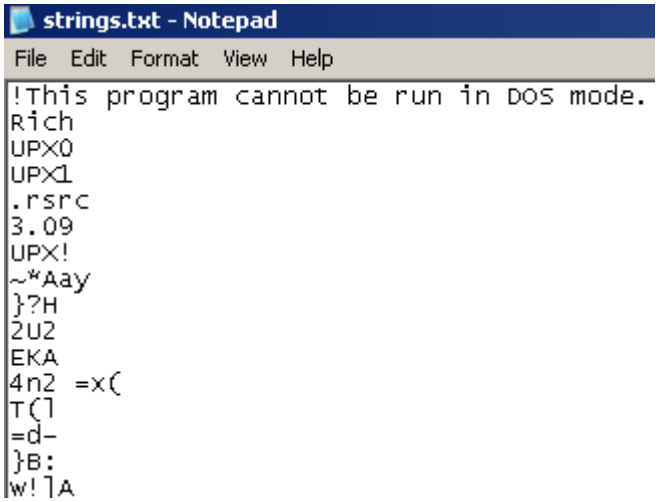


Figure 15- packed evidence 5 - no real strings

Using various tools such as PEviewer and Dependency walker, I found that there are very few imports, which is another characteristic of a packed malware.

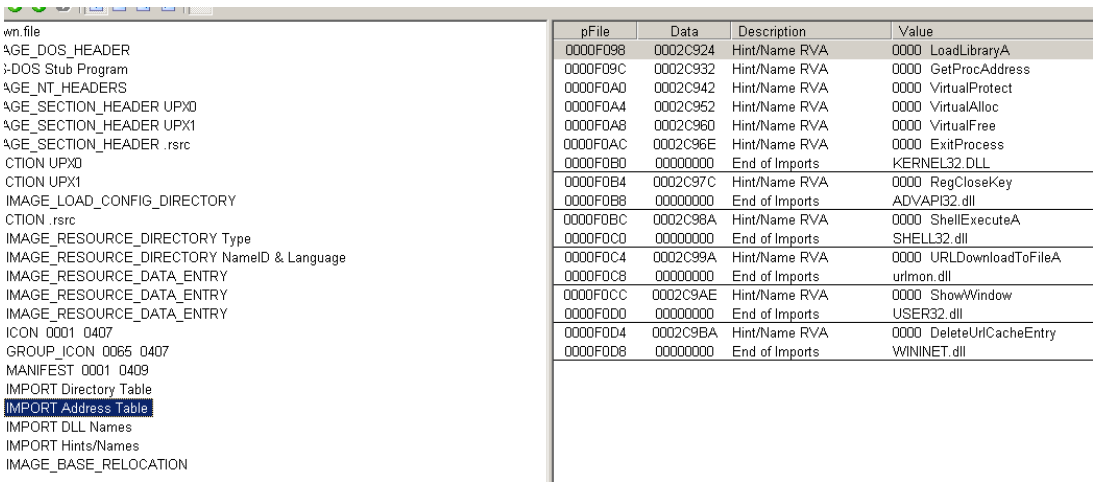


Figure 16- packed evidence 6 - few imports

Finally, there are very few sections that I can view using PEview, another point implying a packed executable.

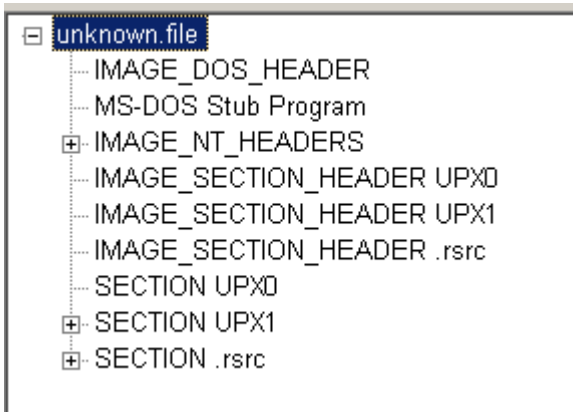


Figure 17- packed evidence 7 – few sections

TASK 3:

Analysis of Imports

I began with unpacking the malware sample to analyse with PEview:

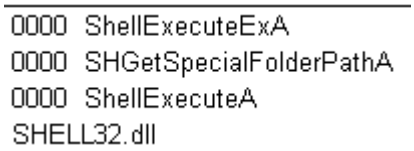


Figure 18- PEview 1.0

I found that the imported functions from shell32.dll are 'SHGetSpecialFolderPathA' and 'ShellExecute…', the first of which indicates a path for a special folder, and the second function performs operations such as opening or launching an application. It can also be used for opening URL in the browser. Together, the imports seem to suggest performance of operations on files or URLs while interacting with folders specified.

Additional investigation into imported functions show 'DeleteUrlCacheEntry' and 'URLDownloadToFileA' functions; the first function makes me believe that the malicious file may be attempting to cover it's tracks by clearing traces of its activities, while the second suggests attempts to download something from the internet, this could be a payload. The two functions together work together in downloading payloads from the internet and then clearing its traces from the URL cache to hide.

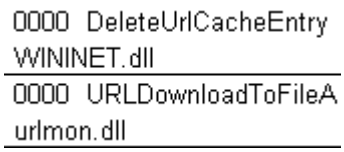


Figure 19 - PEview 1.1

Using PEstudio, we can see ‘DeleteFileA’ function within imports, which further evidence attempts to hide traces, or remove files.

DeleteFileA	✖	-	-	-	kernel32.dll
CreateFileA	✖	-	-	-	kernel32.dll

Figure 20- 'DeleteFileA'

String Analysis

Using ‘string unknown_unpacked.file’, attempts to obfuscate strings were found.

```
abcdefghijklmnopqrstuvwxyz{|}~.
abcdefghijklmnopqrstuvwxyz[\]^_`abcdefghijklmnopqr
abcdefghijklmnopqrstuvwxyz{|}~.....................
abcdefghijklmnopqrstuvwxyz
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqr
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`ABCDEFGHIJKLMNOPQR
ABCDEFGHIJKLMNOPQRSTUVWXYZ{|}~.....................
```

Figure 21- Encoded strings

Registry Manipulation

```
RegSetValueExA
RegEnumKeyExA
RegCloseKey
RegOpenKeyExA
RegQueryValueExA
RegCreateKeyExA
```

Figure 22- registry manipulation.

Further indicators are found in functions that manipulate and interact with Windows Registry, ‘RegSetValueExA’ is usually used for persistence to open whenever OS is started.

VirusTotal Hash Search

Running the hash through VirusTotal indicates it might be a trojan/banking malware aiming to steal sensitive information.

Popular threat label ⓘ trojan.doina/badur **Threat categories** trojan banker **Family labels** doina badur banbra

Figure 23- VirusTotal Hash Search

XML Manifest File

```
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
  <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
    <security>
      <requestedPrivileges>
        <requestedExecutionLevel level="asInvoker" uiAccess="false">
      </requestedPrivileges>
    </security>
  </trustInfo>
</assembly>
```

Figure 24- privilege escalation

A worrying XML manifest, due to previous indicators of compromise, suggests the manifest is attempting to use user privilege and running in the background.

IOCs

Finally using PEstudio, I was able to find numerous indicators and characteristics of malware, including high number of functions exceeding threshold, which may be used for manipulating registries and interacting with other processes. It's also not signed digitally suggesting it is not a legitimate application.

Indicator (24)	Severity
The file queries for files and streams	1
The count (15) of Memory Management Functions has reached the maximum threshold (1) provided	1
The count (5) of Tool Help Functions has reached the maximum threshold (1) provided	1
The count (3) of Error Handling Functions has reached the maximum threshold (1) provided	1
The count (3) of Directory Management Functions has reached the maximum threshold (1) provided	1
The count (11) of Console Functions has reached the maximum threshold (1) provided	1
The count (13) of Dynamic-Link Library Functions has reached the maximum threshold (1) provided	1
The count (41) of Process and Thread Functions has reached the maximum threshold (1) provided	1
The count (5) of SEH Functions has reached the maximum threshold (1) provided	1
The count (25) of File Management Functions has reached the maximum threshold (1) provided	1
The count of blacklisted strings has reached the maximum threshold (30) provided	1
The count of file extensions detected has reached the maximum threshold (5) provided	1
The count of deprecated imported functions has reached the maximum threshold (5) provided	1
The count of imported blacklisted functions has reached the maximum threshold (1) provided	1
The count of Antidebug imported functions has reached the maximum threshold (1) provided	1
The file modifies the registry	2
The file starts child Processes	2
The file queries for visible/invisible window	2
The file references the resources of an executable	2
The count (11) of Registry Functions has reached the maximum threshold (1) provided	2
The file uses Address Space Layout Randomization (ASLR) as Mitigation technique	2
The file checksum is invalid	2
The file has no version information	2
The file is not signed with a Digital Certificate	2

Figure 25 – Indicators

Conclusion

Static analysis of the file found various points of suspicion and indicators of compromise. It interacts with windows registry, uses suspicious functions to access and delete URL/UrlCache, and further absence of digital certificate helps to support this conclusion.

TASK 4

Dynamic Analysis

I began with starting PROCMON, FAKENET then taking first shot of the registry with Regshot.

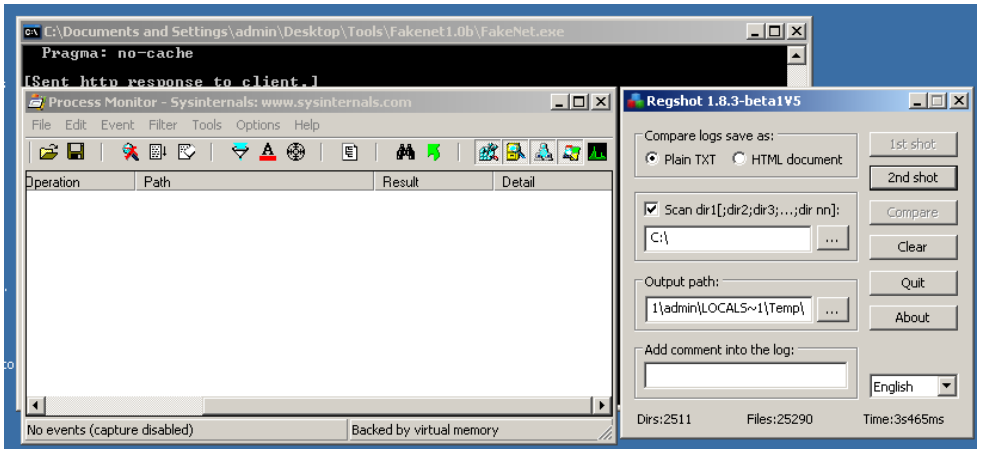


Figure 26-PROCMON, FAKENET, REGSHOT -1st

PROCMON

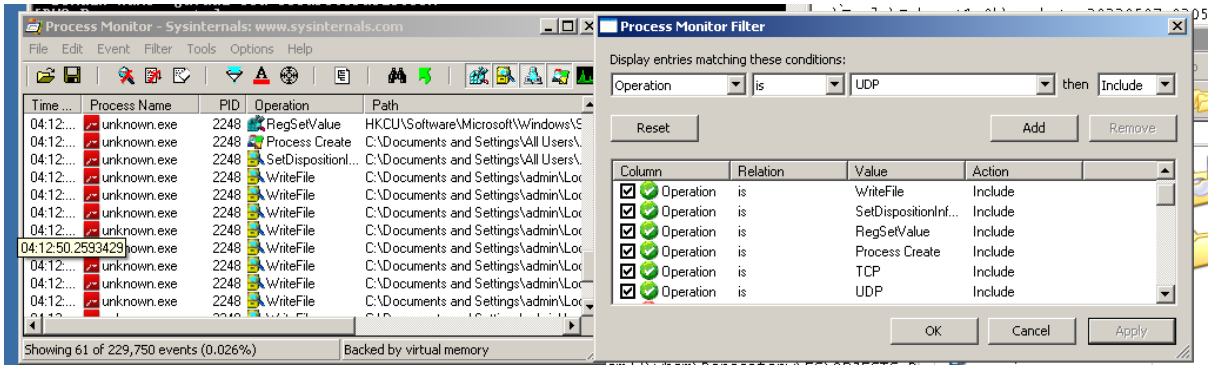


Figure 27- FIltered PROCMON

Within procmon we can see that uknown.exe is modifying MS Cryptographic API – seed values. Possibly for further attacks.

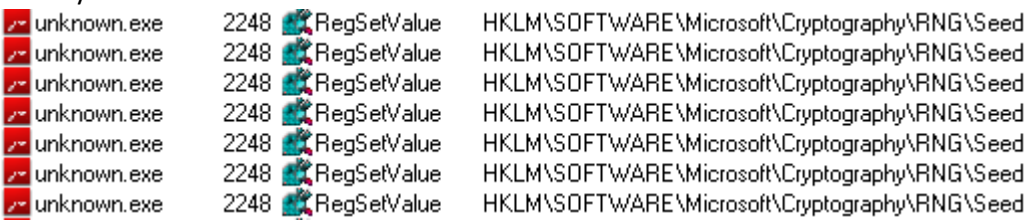


Figure 28- MS CRYPTOGRAPHIC

Here we can see the malware modifying registries related to internet cache. It might be doing this to maintain persistence, hide activities or use the cache directory as the start of a malicious payload.

unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\Cache
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\Directory
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\Paths
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path1\CachePath
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path2\CachePath
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path3\CachePath
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path4\CachePath
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path1\CacheLimit
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path2\CacheLimit
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path3\CacheLimit
unknown.exe	2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Cache\Paths\path4\CacheLimit
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\Cookies
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\History

Figure 29- Cache persistence

Further investigation into the internet settings shows that the malware is modifying registries related to proxy bypass, this could be done as a method of bypassing security measures or avoid detection.

2248	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\Common AppData
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\AppData
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\MigrateProxy
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyEnable
2248	RegSetValue	HKLM\System\CurrentControlSet\Hardware Profiles\0001\Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyEnable
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Connections\SavedLegacySettings
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName
2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet

We can also see below that the malware has successfully downloaded two files and like mentioned previously, is using the temporary internet cache directory as the execution start point. In the end the files are deleted, possibly to maintain persistence and avoid detection or discovery.

unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 0, Leng
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 942, Le
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 2,048,
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 8,192,
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 9,134,
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 16,384
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 17,326
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 24,576
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe	SUCCESS	Offset: 25,518
unknown.exe	2248	WriteFile	C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	Offset: 0, Leng
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{24cd9042-601e-11e4-aa13-806d6172696f}\BaseClass	SUCCESS	Type: REG_S
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{24cd9040-601e-11e4-aa13-806d6172696f}\BaseClass	SUCCESS	Type: REG_S
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\ShellNoRoam\MUICache\C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	Type: REG_S
unknown.exe	2248	Process Create	C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	PID: 2340, Co
unknown.exe	2248	SetDispositionI...	C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	Delete: True
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 0, Leng
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 942, Le
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 2,048,
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 8,192,
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 9,134,
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 16,384
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 17,326
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 24,576
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 25,518
unknown.exe	2248	WriteFile	C:\Documents and Settings\All Users\WindowsXP-KB503313.exe	SUCCESS	Offset: 0, Leng
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\ShellNoRoam\MUICache\C:\Documents and Settings\All Users\WindowsXP-KB503313.exe	SUCCESS	Type: REG_S
unknown.exe	2248	Process Create	C:\Documents and Settings\All Users\WindowsXP-KB503313.exe	SUCCESS	PID: 2368, Co
unknown.exe	2248	SetDispositionI...	C:\Documents and Settings\All Users\WindowsXP-KB503313.exe	SUCCESS	Delete: True

Figure 30 - Two payloads inside temporary internet cache

Furthermore, writing of files within 'All Users' directory could be another attempt to disguise as a legitimate file.

unknown.exe	2248	WriteFile	C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	Offset: 0, Leng
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{24cd9042-601e-11e4-aa13-806d6172696f}\BaseClass	SUCCESS	Type: REG_S
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{24cd9040-601e-11e4-aa13-806d6172696f}\BaseClass	SUCCESS	Type: REG_S
unknown.exe	2248	RegSetValue	HKCU\Software\Microsoft\Windows\ShellNoRoam\MUICache\C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	Type: REG_S
unknown.exe	2248	Process Create	C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	PID: 2340, Co
unknown.exe	2248	SetDispositionI...	C:\Documents and Settings\All Users\WindowsXP-KB503303.exe	SUCCESS	Delete: True
unknown.exe	2248	WriteFile	C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\WindowsXP-KB926139-v2-x86-ENU[1].exe	SUCCESS	Offset: 0, Leng

Figure 31- Procmon 'All Users'

REGSHOT

Using the comparison between REGSHOT 1 and REGSHOT 2, we can see further evidence for persistence such as adding of the files shown in Figure 30.

```
----------------------------------
Files added:8
----------------------------------
C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\OD6FK3AF\windowsXP-KB926139-v2-x86-ENU[1].exe
C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\Content.IE5\U70JKPUX\dotnetfx35setup[1].exe
C:\Documents and Settings\admin\My Documents\Part A\task4\Logfile.CSV
C:\Documents and Settings\admin\Recent\task4.lnk
C:\WINDOWS\Prefetch\REGSHOT.EXE-029EBBAD.pf
C:\WINDOWS\Prefetch\UNKNOWN.EXE-0E62A447.pf
C:\WINDOWS\Prefetch\WINDOWSXP-KB503303.EXE-1EA15746.pf
C:\WINDOWS\Prefetch\WINDOWSXP-KB503313.EXE-2A8A34FC.pf
```

Figure 32- REGSHOT files from figure 30

Attempts to avoid detection can be seen by adding entries into the MuiCache of the Windows registry, which is normally used to store information about application files. This might be done to disguise the payloads as a legitimate application.

```
HKU\S-1-5-21-725345543-746137067-1060284298-1003\Software\Microsoft\windows\ShellNoRoam\MUICache\C:\Documents and Settings\admin\My Documents\Part A\task4\unknown.exe: "unknown"
HKU\S-1-5-21-725345543-746137067-1060284298-1003\Software\Microsoft\windows\ShellNoRoam\MUICache\C:\Documents and Settings\All Users\windowsXP-KB503303.exe: "windowsXP-KB503303"
HKU\S-1-5-21-725345543-746137067-1060284298-1003\Software\Microsoft\windows\ShellNoRoam\MUICache\C:\Documents and Settings\All Users\windowsXP-KB503313.exe: "windowsXP-KB503313"
```

Figure 33- MuiCache Payloads

FAKENET

Network activity can be seen on port 80 where the potentially malicious payloads are being downloaded.

```
C:\Documents and Settings\admin\Desktop\Tools\Fakenet1.0b\FakeNet.exe
[Received new connection on port: 80.]
[New request on port 80.]
  GET /download/0/6/1/061F001C-8752-4600-A198-53214C69B51F/dotnetfx35setup.exe H
TTP/1.1
  Accept: */*
  Accept-Encoding: gzip, deflate
  User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET4.0C;
.NET4.0E; .NET CLR 2.0.50727; .NET CLR 3.0.04506.648; .NET CLR 3.5.21022)
  Host: download.microsoft.com
  Connection: Keep-Alive

[Sent http response to client.]


[Received new connection on port: 80.]
[New request on port 80.]
  GET /download/7/3/4/7345bb7d-0b07-40e8-9480-5b8c55b9c8b7/WindowsXP-KB926139-v2
-x86-ENU.exe HTTP/1.1
  Accept: */*
  Accept-Encoding: gzip, deflate
  User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET4.0C;
.NET4.0E; .NET CLR 2.0.50727; .NET CLR 3.0.04506.648; .NET CLR 3.5.21022)
  Host: download.microsoft.com
  Connection: Keep-Alive
```

Figure 34- PORT 80 GET request.

Previously I mentioned that the malware modified the seed value within the registries, and below we can see the malware interacting with the Microsoft-CryptoApi, possibly to exploit cryptographic functions.

```
[Received new connection on port: 80.]
[New request on port 80.]
  GET /msdownload/update/v3/static/trustedr/en/authrootseq.txt HTTP/1.1
  Accept: */*
  User-Agent: Microsoft-CryptoAPI/5.131.2600.5512
  Host: www.download.windowsupdate.com
  Connection: Keep-Alive
  Cache-Control: no-cache
  Pragma: no-cache

[Sent http response to client.]


[Received new connection on port: 80.]
[New request on port 80.]
  GET /msdownload/update/v3/static/trustedr/en/authrootstl.cab HTTP/1.1
  Accept: */*
  User-Agent: Microsoft-CryptoAPI/5.131.2600.5512
  Host: www.download.windowsupdate.com
  Connection: Keep-Alive
  Cache-Control: no-cache
  Pragma: no-cache

[Sent http response to client.]
[New request on port 443 with SSL.]
```

Figure 35- CrypoAPI

CONCLUSION

Using Procmon, Regshot and fakenet, I was able to dynamically analyse the malware and found that it is modifying various registries including ones related to proxy bypass and internet cache. This alongside the download of the two files inside the the cache and 'All Users' directory hints at the malwares attempts at disguising itself as an legitimate application. Evidence of persistence and avoidance – of detection - have also been found through use of 'MuiCache', deletion of payload files after use and utilisation of internet caches.

TASK 5

NORMAL NETWORK:

```
TCP     54 ams > http [ACK] Seq=1 Ack=1 Win=64240 Len=0
HTTP   395 GET /download/0/6/1/061F001C-8752-4600-A198-53214C69B51F/dotnetfx35setup.exe HTTP/1.1
TCP     60 http > ams [ACK] Seq=1 Ack=342 Win=65535 Len=0
HTTP   303 HTTP/1.1 302 Moved Temporarily
HTTP   409 GET /download/7/3/4/7345bb7d-0b07-40e8-9480-5b8c55b9c8b7/WindowsXP-KB926139-v2-x86-ENU.exe HTTP/1.1
TCP     60 http > ams [ACK] Seq=250 Ack=697 Win=65535 Len=0
HTTP   317 HTTP/1.1 302 Moved Temporarily
```

Figure 36-HTTP GET Request

When running the malware on a normal network, the malware executable sent a HTTP GET request to download additional payloads, and the server responsed with an acknowledgment as part of the three way handshake, however HTTP 303 status code is sent back, indicating that the resource requested has been moved into a different URL, and the server provides the new location, which in

this instance, is a redirection of the same resource over HTTPS instead of HTTP.

```
⊞ Content-Length: 0\r\n
  Location: https://download.microsoft.com/download/7/3/4/7345bb7d-0b07-40e8-9480-5b8c55b9c8b7/WindowsXP-KB926139-v2-x86-ENU.exe\r\n
```

Figure 37-resource redirection.

ISOLATED NETWORK

After setting an isolated network between REMnux and WINXP machine. I was able to analyse the network behaviour using inetsim, apateDNS and wireshark:

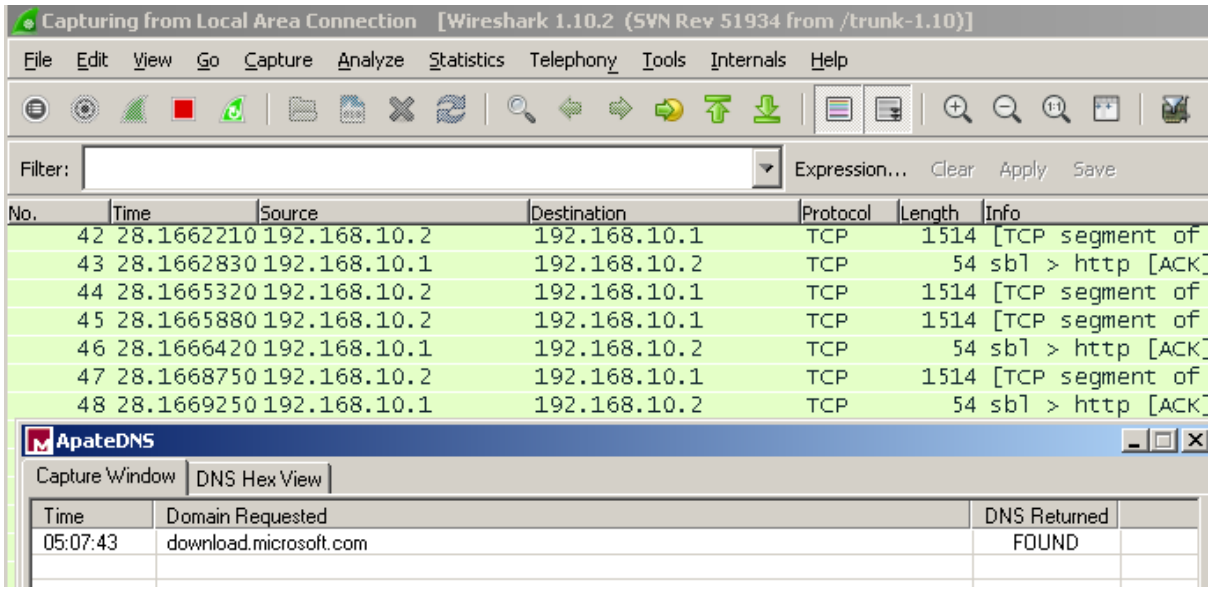


Figure 38- network analysis tools.

Running unknown.exe file under isolated network exhibits various network-based behaviour. Reading the Inetsim logs after running the malware for 20 minutes shows that HTTP connections were attempted, and files were downloaded from external sources.

Malware first attempted to make an HTTP GET request to download a file from "http://download.microsoft.com/download/0/6/1/061F001C-8752-4600-A198-53214C69B51F/dotnetfx35setup.exe", to which Inetsim created a fake file in response called sample_gui.exe.

Once again malware made another HTTP GET request from "http://download.microsoft.com/download/7/3/4/7345bb7d-0b07-40e8-9480-5b8c55b9c8b7/WindowsXP-KB926139-v2-x86-ENU.exe", and again Inetsim created a fake file named 'sample_gui.exe' in response.

```
remnux@remnux:~$ sudo cat /var/log/inetsim/report/report.1528.txt
=== Report for session '1528' ===

Real start date           : 2023-05-09 23:07:13
Simulated start date      : 2023-05-09 23:07:13
Time difference on startup : none

2023-05-09 23:07:45  First simulated date in log file
2023-05-09 23:07:45  HTTP connection, method: GET, URL: http://download.microsoft.com/download/0/6/1/061F001C-8752-4600-A198-53214C69B51F/dotnetfx35setup.exe, file name: /var/lib/inetsim/http/fakefiles/sample_gui.exe
2023-05-09 23:07:47  HTTP connection, method: GET, URL: http://download.microsoft.com/download/7/3/4/7345bb7d-0b07-40e8-9480-5b8c55b9c8b7/WindowsXP-KB926139-v2-x86-ENU.exe, file name: /var/lib/inetsim/http/fakefiles/sample_gui.exe
2023-05-09 23:07:47  Last simulated date in log file

===
remnux@remnux:~$
```

Figure 39- Inetsim logs.

We see futher evidence of this HTTP GET request using wireshark on WINXP machine:

32	28.1527480	192.168.10.1	192.168.10.2	TCP	54 sbl > http [ACK] Seq=1 Ack=1 Win=64240 Len=0
33	28.1546020	192.168.10.1	192.168.10.2	HTTP	409 GET /download/7/3/4/7345bb7d-0b07-40e8-9480-5b8c55b9c8b7/WindowsXP-KB926139-v2-x86-ENU.exe HTTP/1.1
34	28.1559460	192.168.10.2	192.168.10.1	TCP	60 http > sbl [ACK] Seq=1 Ack=356 Win=63885 Len=0

Figure 40-wireshark HTTP GET request.

Periodically the malware also sent TCP bytes in range of 50-60s to DST port: 1043.

No.	Time	Source	Destination	Protocol	Length	Info
79	71.8854800	192.168.10.2	192.168.10.1	SSLv3	61	Alert (Level: Fatal, Description: Handshake Failure)
80	71.8924600	192.168.10.2	192.168.10.1	TCP	60	https > afrog [FIN, ACK] Seq=8 Ack=79 Win=64162 Len=0
81	71.8924960	192.168.10.1	192.168.10.2	TCP	54	afrog > https [ACK] Seq=79 Ack=9 Win=64233 Len=0
82	71.8939980	192.168.10.1	192.168.10.2	TCP	54	afrog > https [FIN, ACK] Seq=79 Ack=9 Win=64233 Len=0
83	71.8950550	192.168.10.2	192.168.10.1	TCP	60	https > afrog [ACK] Seq=9 Ack=80 Win=64162 Len=0
84	71.8954080	192.168.10.1	192.168.10.2	TCP	62	boinc-client > https [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_P
85	71.8964500	192.168.10.2	192.168.10.1	TCP	62	https > boinc-client [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=
86	71.8964740	192.168.10.1	192.168.10.2	TCP	54	boinc-client > https [ACK] Seq=1 Ack=1 Win=64240 Len=0
87	71.8979840	192.168.10.1	192.168.10.2	SSLv2	99	Client Hello
88	71.8990250	192.168.10.2	192.168.10.1	TCP	60	https > boinc-client [ACK] Seq=1 Ack=46 Win=64195 Len=0
89	71.9013370	192.168.10.2	192.168.10.1	TLSv1.2	61	Alert (Level: Fatal, Description: Protocol Version)
90	71.9015740	192.168.10.1	192.168.10.2	TCP	54	boinc-client > https [FIN, ACK] Seq=46 Ack=8 Win=64233 Len=0
91	71.9046080	192.168.10.2	192.168.10.1	TCP	60	https > boinc-client [FIN, ACK] Seq=8 Ack=47 Win=64194 Len=0

```
    Destination: 192.168.10.1 (192.168.10.1)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
⊟ Transmission Control Protocol, Src Port: https (443), Dst Port: boinc-client (1043), Seq: 1, Ack: 46, Len: 0
    Source port: https (443)
    Destination port: boinc-client (1043)
```

Figure 41- TCP.

CONCLUSION:

The malware executable in both instances attempts to download additional payload externally.

PART B

TASK 1

As DLL needs a host process to run it cannot be executed by double-clicking it, meaning the friend's system is unlikely to be infected, therefore, to make sure of this assumption, I ran a dynamic analysis using Regshot, Fakenet and PROCEXPLORER.

System Idle Process	94.00	0 K	16 K	0		
System		0 K	212 K	4		
Interrupts	6.00	0 K	0 K	n/a	Hardware Interrupts and DPCs	
smss.exe	System	164 K	372 K	364	Windows NT Session Mana...	Microsoft Corporation
csrss.exe		1,608 K	3,240 K	512	Client Server Runtime Process	Microsoft Corporation
winlogon.exe		6,324 K	4,280 K	536	Windows NT Logon Applicat...	Microsoft Corporation
services.exe		1,568 K	3,136 K	584	Services and Controller app	Microsoft Corporation
VBoxService.exe		1,276 K	3,484 K	788	VirtualBox Guest Additions S...	Oracle Corporation
svchost.exe		3,052 K	4,660 K	856	Generic Host Process for Wi...	Microsoft Corporation
wmiprvse.exe		2,320 K	4,644 K	256	WMI	Microsoft Corporation
svchost.exe		1,632 K	3,996 K	940	Generic Host Process for Wi...	Microsoft Corporation
svchost.exe		12,380 K	20,268 K	1032	Generic Host Process for Wi...	Microsoft Corporation
wscntfy.exe		448 K	1,776 K	1928	Windows Security Center No...	Microsoft Corporation
wuauclt.exe		6,356 K	6,456 K	1152	Automatic Updates	Microsoft Corporation
svchost.exe		1,188 K	3,316 K	1084	Generic Host Process for Wi...	Microsoft Corporation
svchost.exe		1,652 K	4,284 K	1120	Generic Host Process for Wi...	Microsoft Corporation
spoolsv.exe		3,028 K	4,336 K	1568	Spooler SubSystem App	Microsoft Corporation
jqs.exe		1,920 K	1,384 K	816	Java Quick Starter Service	Oracle Corporation
alg.exe		1,060 K	3,360 K	160	Application Layer Gateway S...	Microsoft Corporation
lsass.exe		3,660 K	5,688 K	596	LSA Shell (Export Version)	Microsoft Corporation
explorer.exe		13,980 K	5,024 K	1516	Windows Explorer	Microsoft Corporation
VBoxTray.exe		1,328 K	3,580 K	1712	VirtualBox Guest Additions Tr...	Oracle Corporation
jusched.exe		628 K	2,084 K	1728	Java(TM) Update Scheduler	Oracle Corporation
ctfmon.exe		788 K	2,800 K	1760	CTF Loader	Microsoft Corporation
msmsgs.exe		1,168 K	1,652 K	1772	Windows Messenger	Microsoft Corporation
regshot.exe		23,336 K	24,932 K	1980	Regshot	Regshot Team
procexp.exe		11,108 K	14,860 K	1988	Sysinternals Process Explorer	Sysinternals - www.sysinter...
Procmon.exe		5,264 K	8,160 K	1796		
FakeNet.exe		7,452 K	9,872 K	1008		
ipconfig.exe		68 K	72 K	984		

Figure 42 - Before double-clicking.

System		0 K	212 K	4		
Interrupts	3.00	0 K	0 K	n/a	Hardware Interrupts and DPCs	
smss.exe		164 K	372 K	364	Windows NT Session Mana...	Microsoft Corporation
csrss.exe	1.00	1,616 K	3,232 K	512	Client Server Runtime Process	Microsoft Corporation
winlogon.exe		6,324 K	4,280 K	536	Windows NT Logon Applicat...	Microsoft Corporation
services.exe		1,568 K	3,136 K	584	Services and Controller app	Microsoft Corporation
VBoxService.exe		1,276 K	3,488 K	788	VirtualBox Guest Additions S...	Oracle Corporation
svchost.exe		2,972 K	4,628 K	856	Generic Host Process for Wi...	Microsoft Corporation
wmiprvse.exe		2,320 K	4,644 K	256	WMI	Microsoft Corporation
svchost.exe		1,632 K	3,996 K	940	Generic Host Process for Wi...	Microsoft Corporation
svchost.exe		12,248 K	19,972 K	1032	Generic Host Process for Wi...	Microsoft Corporation
wscntfy.exe		448 K	1,776 K	1928	Windows Security Center No...	Microsoft Corporation
wuauclt.exe		6,344 K	6,448 K	1152	Automatic Updates	Microsoft Corporation
svchost.exe		1,200 K	3,328 K	1084	Generic Host Process for Wi...	Microsoft Corporation
svchost.exe		1,628 K	4,276 K	1120	Generic Host Process for Wi...	Microsoft Corporation
spoolsv.exe		3,068 K	4,368 K	1568	Spooler SubSystem App	Microsoft Corporation
jqs.exe		1,920 K	1,388 K	816	Java Quick Starter Service	Oracle Corporation
alg.exe		1,048 K	3,352 K	160	Application Layer Gateway S...	Microsoft Corporation
lsass.exe		3,660 K	5,688 K	596	LSA Shell (Export Version)	Microsoft Corporation
explorer.exe		11,476 K	7,640 K	1516	Windows Explorer	Microsoft Corporation
VBoxTray.exe		1,328 K	3,580 K	1712	VirtualBox Guest Additions Tr...	Oracle Corporation
jusched.exe		628 K	2,084 K	1728	Java(TM) Update Scheduler	Oracle Corporation
ctfmon.exe		788 K	2,800 K	1760	CTF Loader	Microsoft Corporation
msmsgs.exe		1,156 K	1,728 K	1772	Windows Messenger	Microsoft Corporation
regshot.exe		23,336 K	24,932 K	1980	Regshot	Regshot Team
procexp.exe		11,112 K	14,916 K	1988	Sysinternals Process Explorer	Sysinternals - www.sysinter...
Procmon.exe		6,272 K	8,888 K	1796		
FakeNet.exe		7,452 K	9,880 K	1008		
ipconfig.exe		68 K	72 K	984		
ResourceHacker.exe		4,772 K	6,600 K	2468	Resource viewer, decompiler...	Angus Johnson

Figure 43- suspicious resourcehacker.exe popup

From the two procexplorer states, there is nothing running to suggest an infection. However, looking more closely into the regshot comparison we can see attempts manipulating system settings and registries.

The ResourceHacker.exe pops up when double-clicking on 'malware.dll' and the regshot shows addition of new keys that add shell menu for the resource hacker, which is suspicious.

```
----------------------------------
Keys added:19
----------------------------------
HKLM\SOFTWARE\Classes\Applications\ResourceHacker.exe
HKLM\SOFTWARE\Classes\Applications\ResourceHacker.exe\shell
HKLM\SOFTWARE\Classes\Applications\ResourceHacker.exe\shell\ResourceHacker
HKLM\SOFTWARE\Classes\Applications\ResourceHacker.exe\shell\ResourceHacker\command
```

Figure 44-added key.

With further investigation, we can see that the file 'malsample.dll.lnk' has been added, possibly creating shortcuts or another link, plus the directory that malware container in – Part B – has also had a link created – 'Part B.lnk'.

```
----------------------------------
Files added:4
----------------------------------
C:\Documents and Settings\admin\Local Settings\History\History.IE5\MSHist012023050720230508\index.dat
C:\Documents and Settings\admin\Recent\malsample.dll.lnk
C:\Documents and Settings\admin\Recent\Part B.lnk
C:\WINDOWS\SoftwareDistribution\DataStore\Logs\tmp.edb
```

Figure 45-Added files.

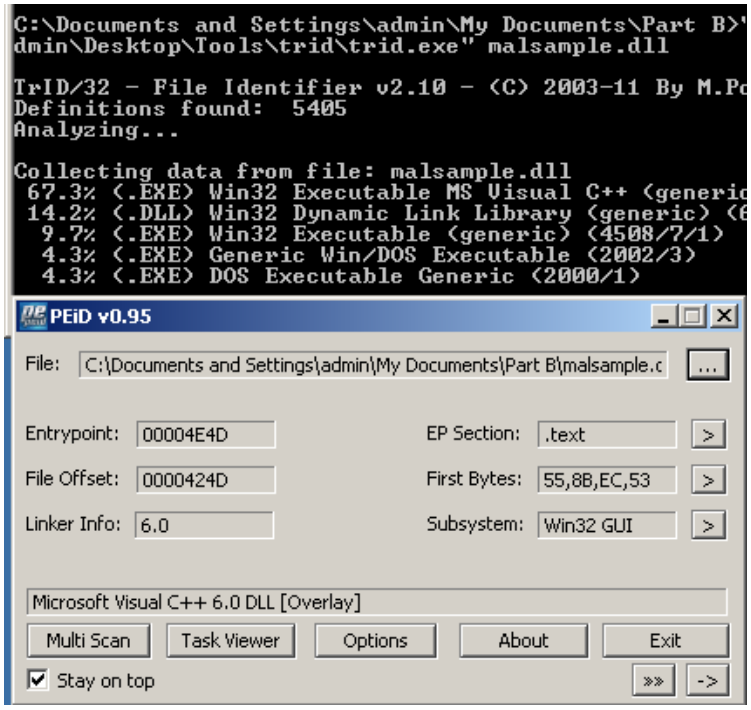
The addition of 'index.dat' which is a file used to store browsing history alongside the folder below adds further suspicion as internet explorer wasn't opened during this period.

```
----------------------------------
Folders added:1
----------------------------------
C:\Documents and Settings\admin\Local Settings\History\History.IE5\MSHist012023050720230508S
```

However, creation of 'index.dat' file is a standard practice and 'malsample.dll.lnk' Is merely a shortcut rather than being inherently malicious. This alongside the Fakenet output after double-clicking on the .dll file makes me believe that the friend's system is not infected, and the registry changes are from the legitimate resourcehacker.exe program.

TASK 2

With the use of trid.exe and PEiD, we can assume that the malware is not packed and is instead a Windows DLL.



VIRUSTOTAL

Scans from VirusTotal confirms that it is malicious and within the 'Ulise' and Gmlh' malware family, it has been detected as a trojan and backdoor. Given the results, we can safely assume it is malicious and a threat to the system and advised to be quarantined or deleted.

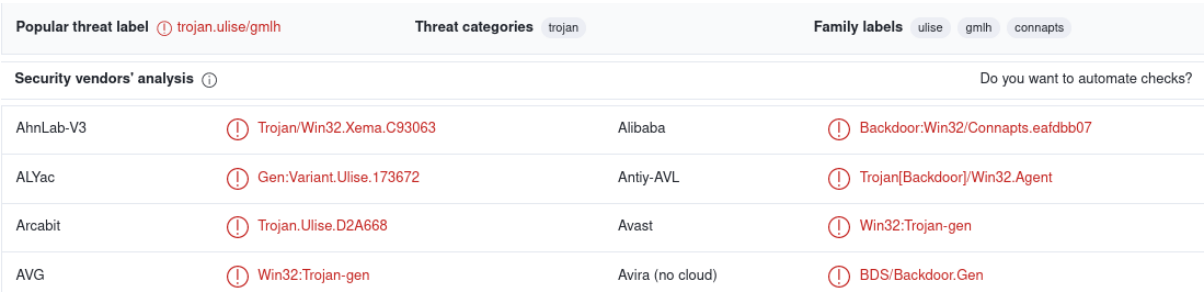


Figure 46- Virus total scans

IMPORTS/EXPORTS

Using CFF Walker, we can see that the malware.dll has export functionality to allow installation, uninstallations, and entry point through 'ServiceMain' might to either start a service or run as a service to feign legitimacy and evade detection.

Ordinal	Function RVA	Name Ordinal	Name RVA	Name
(nFunctions)	Dword	Word	Dword	szAnsi
00000001	00004706	0000	00005969	Install
00000002	00003196	0001	00005978	ServiceMain
00000003	00004B18	0002	00005984	UninstallService
00000004	00004B0B	0003	00005995	installA
00000005	00004C2B	0004	0000599E	uninstallA

Figure 47-CFF Explorer Exports

To further evidence actions through service, we can see the imported functions within 'advapi32.dll' relate to windows services, possibly to install itself as a service for persistence.

```
0147  OpenServiceA
0078  DeleteService
0172  RegOpenKeyExA
017B  RegQueryValueExA
015B  RegCloseKey
0145  OpenSCManagerA
004C  CreateServiceA
0034  CloseServiceHandle
015E  RegCreateKeyA
0186  RegSetValueExA
018E  RegisterServiceCtrlHandlerA
01AE  SetServiceStatus
ADVAPI32.dll
```

Figure 48- PEview advapi32.dll

Through 'kernel32.dll' imported functions, we can see that the malware also has functions for creating and manipulating processes, while also having functions for file manipulation or information gathering e.g… 'GetCurrentDirectoryA' or 'GetModuleFileNameA'

```
0150  GetStartupInfoA
0043  CreatePipe
00F5  GetCurrentDirectoryA
0044  CreateProcessA
0308  lstrlenA
0271  SetLastError
01F5  OutputDebugStringA
001B  CloseHandle
0218  ReadFile
0165  GetTempPathA
0121  GetLongPathNameA
01C2  LoadLibraryA
013E  GetProcAddress
004A  CreateThread
015D  GetSystemTime
02CE  WaitForSingleObject
029F  TerminateThread
0296  Sleep
011A  GetLastError
0124  GetModuleFileNameA
KERNEL32.dll
```

Figure 49- PEview Imports Kernel32.dll

We can see further evidence of file manipulation through ‘msvscrt.dll’ imported functions for string, file manipulation and memory allocation e.g… ‘malloc’ or ‘strcpy’

00AC	_chdir
01C5	_strnicmp
009D	_adjust_fdiv
0291	malloc
010F	_initterm
025E	free
000E	??1type_info@@UAE@XZ
00CA	_except_handler3
0041	_CxxThrowException
01C1	_stricmp
0042	_EH_prolog
0049	__CxxFrameHandler
02B7	strchr
0134	_itoa
02C5	strstr
02BF	strncat
02BE	strlen
02B5	sscanf
023E	atol
000F	??2@YAPAXI@Z
0299	memset
02F1	wcstombs
02C1	strncpy
02B6	strcat
02BA	strcpy
023D	atoi
024C	fclose
024F	fflush
0010	??3@YAXPAX@Z
0266	fwrite
0257	fopen
02C3	strrchr

Figure 50- msvcrt.dll imports.

Using PEview we can also see that the malware internet/network communication functionality. This is evidenced by functions such as ‘InternetConnectA’ in wininet.dll or ‘WSASocketA’ in ws2_32.dll which creates network socket for communication.

0056	InternetCloseHandle
006F	InternetOpenA
005A	InternetConnectA
0045	HttpOpenRequestA
0049	HttpSendRequestA
0047	HttpQueryInfoA
0077	InternetReadFile
WININET.dll	
000B	
003D	WSASocketA
0003	
0004	
000A	
0013	
0012	
0097	
0010	
0016	
0073	
0039	
0074	
0009	
WS2_32.dll	

Figure 51-internet and network communications

STRINGS:

The strings extracted provide further insight into the functionality of the malware. For example, we see functions mentioned previously that deal with network communication and use of windows service to collect and store information.

```
LocalSystem
ErrorControl
DisplayName
Description
Depends INA+, Collects and stores network configuration and location information, and notifies applications when this information changes.
ImagePath
%SystemRoot%\System32\svchost.exe -k
SYSTEM\CurrentControlSet\Services\
CreateService(%s) error %d
Intranet Network Awareness (INA+)
%SystemRoot%\System32\svchost.exe -k netsvcs
OpenSCManager()
You specify service name not in Svchost//netsvcs, must be one of following:
RegQueryValueEx(Svchost\netsvcs)
netsvcs
```

Figure 52- INA+ collects and stores information.

There are also suspicious encoded strings, and evidence of file/system interaction through 'CreateProcess' or command execution 'cmd.exe /c'

```
CreateProcessA
kernel32.dll
.exe
HTTP/1.1
%s %s
1234567890123456
quit
exit
getfile
cmd.exe /c
```

Figure 53- System interaction

```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
```

Figure 54-Encoded strings

SECTIONS:

During investigation into 'section.data' I managed to find a suspicious encoding alongside a URL, a 'serve.html' and a get http request.

```
Y29ubmVjdA==....
.`......practica
lmalwareanalysis
.com............
................
........serve.ht
ml..............
................
................
.........`...`..
.`...`..dW5zdXBw
b3J0....c2xlZXA=
....Y21k....cXVp
dA==....*/*. Win
dows XP 6.11....
tt..CreateProces
sA..kernel32.dll
....wb...exe....
\...GET.HTTP/1.1
```

Figure 55- section.data.

The encoding is in base64 and decodes to connect, this alongside the previous findings, indicates suspicious behaviour on the internet and potentially malicious .html file.

connect

Figure 56- Y29ubmVjdA== decoded.

CONCLUSION:

In Summary, through the static analysis of the malware, we find various functions to be extremely suspicious, indicating a higher likelihood of malicious intent. The malware has functions to manipulate processes and threats, possibly to create a Command-and-control channel, has network communication abilities to possibly move lateral through the network, it also communicates through the internet. The section also contains suspicious encoding and URL with GET http/1.1, to provide further insight into malware behaviour. See below for multiple indicators of compromise using PEstudio:

Indicator (24)	Severity
The count (11) of WinINet Functions has reached the maximum threshold (3) provided	1
The count (5) of Dynamic-Link Library Functions has reached the maximum threshold (1) provided	1
The count (9) of Process and Thread Functions has reached the maximum threshold (1) provided	1
The count (15) of Service Functions has reached the maximum threshold (1) provided	1
The count (5) of File Management Functions has reached the maximum threshold (1) provided	1
The count of blacklisted strings has reached the maximum threshold (30) provided	1
The file contains 1 MIME64 Encoding string(s)	1
The file original name is "Lab03-02.dll"	1
The count of file extensions detected has reached the maximum threshold (5) provided	1
The count of deprecated imported functions has reached the maximum threshold (5) provided	1
The count of imported blacklisted functions has reached the maximum threshold (1) provided	1
The file embeds a file (Type: Unknown, MD5: 93B885ADFE0DA089CDF634904FD59F71)	1
The file modifies the registry	2
The file references the Service Control Manager (SCM)	2
The file starts child Processes	2
The count (9) of Registry Functions has reached the maximum threshold (1) provided	2
The file ignores Data Execution Prevention (DEP) as Mitigation technique	2
The file ignores Address Space Layout Randomization (ASLR) as Mitigation technique	2
The file checksum is invalid	2
The file is resource-less	2
The file has no version information	2
The file ignores Cookies placed on the Stack (GS) as Mitigation technique	2
The file is not signed with a Digital Certificate	2
The file imports 3 decorated Symbols	2

Figure 57- IOCs - e.g checksum invalid, filename changed, embedded file.

TASK 3

Using previous analysis and more look into PEview exports, we can see that the .dll file installs as a service, therefore, we run it with the following command 'rundll32.exe malsample.dll install'.

pFile	Data	Description	Value
00004D28	00004706	Function RVA	0001 Install
00004D2C	00003196	Function RVA	0002 ServiceMain
00004D30	00004B18	Function RVA	0003 UninstallService
00004D34	00004B0B	Function RVA	0004 installA
00004D38	00004C2B	Function RVA	0005 uninstallA

Figure 58 - PEview exports

However, before that I start up fakenet and take the 1st regshot for analysis.

```
Keys added:6
HKLM\SYSTEM\ControlSet001\Services\IPRIP
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Parameters
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Security
HKLM\SYSTEM\CurrentControlSet\Services\IPRIP
HKLM\SYSTEM\CurrentControlSet\Services\IPRIP\Parameters
HKLM\SYSTEM\CurrentControlSet\Services\IPRIP\Security

Values deleted:2
HKLM\SYSTEM\ControlSet001\Services\kmixer\Enum\0: "SW\{b7eafdc0-a680-11d0-96d8-00aa0051e51d}\{9B365890-165F-11D0-A195-0020AFD156E4}"
HKLM\SYSTEM\CurrentControlSet\Services\kmixer\Enum\0: "SW\{b7eafdc0-a680-11d0-96d8-00aa0051e51d}\{9B365890-165F-11D0-A195-0020AFD156E4}"

Values added:20
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Security\Security: 01 00 14 80 90 00 00 00 9C 00 00 00 14 00 00 00 30 00 00 00 02 00 1C 00 01 00 00 00 02 80 14 00 FF 01 0F 00 01 01 00 00 00 00
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Parameters\ServiceDll: "C:\Documents and Settings\admin\My Documents\Part B\malsample.dll"
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Type: 0x00000020
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Start: 0x00000002
HKLM\SYSTEM\ControlSet001\Services\IPRIP\ErrorControl: 0x00000001
HKLM\SYSTEM\ControlSet001\Services\IPRIP\ImagePath: "%SystemRoot%\System32\svchost.exe -k netsvcs"
HKLM\SYSTEM\ControlSet001\Services\IPRIP\DisplayName: "Intranet Network Awareness (INA+)"
HKLM\SYSTEM\ControlSet001\Services\IPRIP\ObjectName: "LocalSystem"
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Description: "Depends INA+, Collects and stores network configuration and location information, and notifies applications when this information changes."
HKLM\SYSTEM\ControlSet001\Services\IPRIP\DependOnService: 'RpcSs'
HKLM\SYSTEM\CurrentControlSet\Services\IPRIP\Security\Security: 01 00 14 80 90 00 00 00 9C 00 00 00 14 00 00 00 30 00 00 00 02 00 1C 00 01 00 00 00 02 80 14 00 FF 01 0F 00 01 01 00 00 00
HKLM\SYSTEM\CurrentControlSet\Services\IPRIP\Parameters\ServiceDll: "C:\Documents and Settings\admin\My Documents\Part B\malsample.dll"
```

Figure 59- regshot comparison.

We can see that a service 'IPRIP' has been installed under the name 'INA+', whilst also important to note is that the imagepath is under 'svchost.exe -k netsvcs'. Using command line, I was able to start the service:

```
C:\Documents and Settings\admin\My Documents\Part B>net start IPRIP
The Intranet Network Awareness (INA+) service is starting.
The Intranet Network Awareness (INA+) service was started successfully.
```

Figure 60- start IPRIP service.

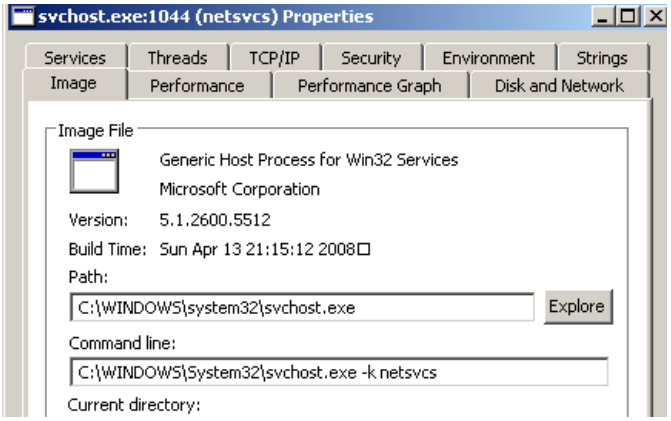


Figure 61-svchost.exe disguise

The malware is clearly trying to disguise itself as a windows service to avoid detection and further persistence is created with display name as 'Intranet Network Awareness (INA+).

The malware also added new values such as ServiceDLL which points to the malicious .dll for when the service is executed.

```
Values added:20
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Security\Security: 01 00 14 80 90 00 00 00 9C 00 00 00 14 00 00 00 30 00 00 00 02 00 1C 00 (
HKLM\SYSTEM\ControlSet001\Services\IPRIP\Parameters\ServiceDll: "C:\Documents and Settings\admin\My Documents\Part B\malsample.dll"
```

Figure 62- ServiceDLL value

Further evidence of registry keys and value modification is given by the system log files:

```
iles [attributes?] modified:2
:\WINDOWS\system32\config\software.LOG
:\WINDOWS\system32\config\system.LOG
```

Figure 63- System logs.

Using the PID found from process explorer, I was able to locate it with filters on process monitor to further evidence malicious activities such as 'ProcessCreate'

..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	Process Create
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
5.0843742	st.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	WriteFile
..	svchost.exe	1044	RegSetValue
..	svchost.exe	1044	RegSetValue
..	svchost.exe	1044	RegSetValue
	svchost.exe	1044	WriteFile

Figure 64- PID filter PROCMON

CONCLUSION:

The analysis shows that malsample.dll masquerades as a windows service under the display name 'INA+' creating persistence and avoiding detection by using image path 'scvhost.exe -k netsvcs'.

The serviceDLL executes the malware upon start of service and modifications of the registries show evidence of malicious activity within the system.

TASK 4

To set up a safe virtual network analysis environment, I would initially choose two virtual machines for Virtual box; XP to act as host for the malware and REMnux for networking analysis. Then I would set the networking setting within my XP VM to host-only adapter, to check for any activities using

Fakenet. After analysing the XP machine alone, I would switch to 'internal networking' settings on both VMs, while setting up the network interface with IP address for both Windows XP machine and REMNUX machine as seen below so that they are placed within an isolated network but able to communicate with each other and analyse the network activity using ApateDNS and Inetsim.

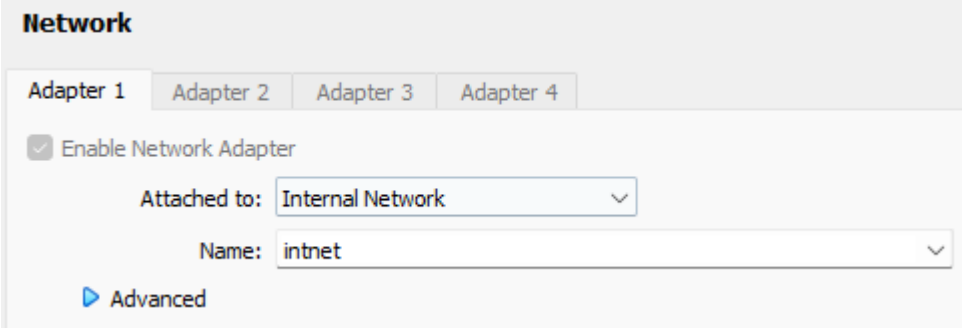


Figure 65-VM Network setting.

```
remnux@remnux:~$ sudo ifconfig enp0s3 192.168.10.2 netmask 255.255.255.0 broadcast 192.168.10.255
```

Figure 66-remnux nterwork interface

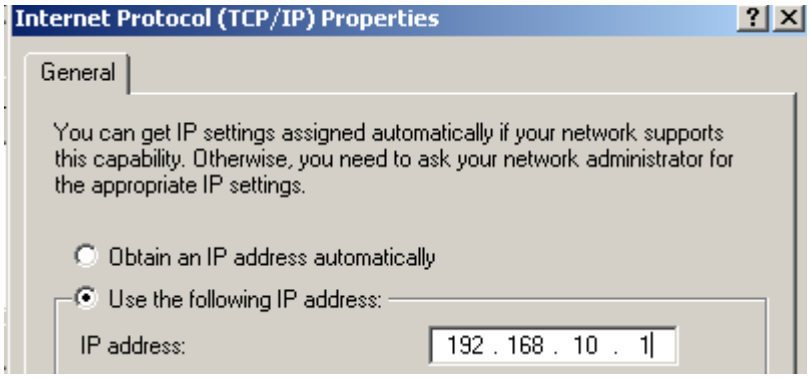


Figure 67-XP network interface.

```
Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.10.1
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

C:\Documents and Settings\admin>ping 192.168.10.2

Pinging 192.168.10.2 with 32 bytes of data:

Reply from 192.168.10.2: bytes=32 time=2ms TTL=64
Reply from 192.168.10.2: bytes=32 time=1ms TTL=64
Reply from 192.168.10.2: bytes=32 time=1ms TTL=64

Ping statistics for 192.168.10.2:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
```

Figure 68-Internal connectivity.

Once this is done and tested, it's time to set up the tools such as INetSIM, Fakenet and wireshark to capture and analyse the traffic. Finally, taking a snapshot before running the malware.

After setting up an isolated environment between windows XP and REMnux, I found that the malware sends a HTTP GET request to an URL retrieve .html file.

```
Hypertext Transfer Protocol
  GET /serve.html HTTP/1.1\r\n
    Accept: */*\r\n
    User-Agent: winxp Windows XP 6.11\r\n
    Host: practicalmalwareanalysis.com\r\n
    \r\n
    [Full request URI: http://practicalmalwareanalysis.com/serve.html]
    [HTTP request 1/1]
    [Response in frame: 15]
```

Figure 69- HTTP GET REQUEST

This request is done using the system name – winxp, possibly using the imported functions to accomplish this.

We can also see using ApateDNS that query is made for 'practicalmalwareanalysis.com'.

ApateDNS

Capture Window | DNS Hex View

Time	Domain Requested	DNS Returned
10:02:07	practicalmalwareanalysis.com	FOUND

Figure 70- DNS query

Service also sends packets of various sizes – within 60 range – periodically to port 80 using TCP.

16	117.779358	127.0.0.1	127.0.0.2	TCP	42
19	270.218555	127.0.0.2	127.0.0.1	TCP	40
20	270.218555	127.0.0.1	127.0.0.2	TCP	40

```
Frame 16: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
  Encapsulation type: Raw IP (7)
  Arrival Time: May  9, 2023 10:28:46.982556000 W. Europe Standard
  [Time shift for this packet: 0.000000000 seconds]
  Epoch Time: 1683620926.982556000 seconds
  [Time delta from previous captured frame: 0.080115000 seconds]
  [Time delta from previous displayed frame: 0.080115000 seconds]
  [Time since reference or first frame: 117.779358000 seconds]
  Frame Number: 16
  Frame Length: 42 bytes (336 bits)
  Capture Length: 42 bytes (336 bits)
  [Frame is marked: False]
  [Frame is ignored: False]
  [Protocols in frame: raw:ip:tcp]
  [Coloring Rule Name: HTTP]
  [Coloring Rule String: http || tcp.port == 80]
```

Figure 71-TCP PORT 80

CONCLUSION

Net activities include TCP packets being sent periodically, DNS query to 'practicalmalwareanalysis.com' and HTTP GET requests to 'practicalmalwareanalysis.com', the request is for a html file called '/serve.html' and user agent is the system name.

TASK 5

EXPORTS

After opening malsample.dll using IDA pro I used the Export subview to find a total of 6 export functions with the addresses:

Install: 10004706
ServiceMain: 10003196
UninstallService: 10004B18
installA: 10004B0B
uninstallA: 10004C2B
DllEntryPoint: 1004E4D

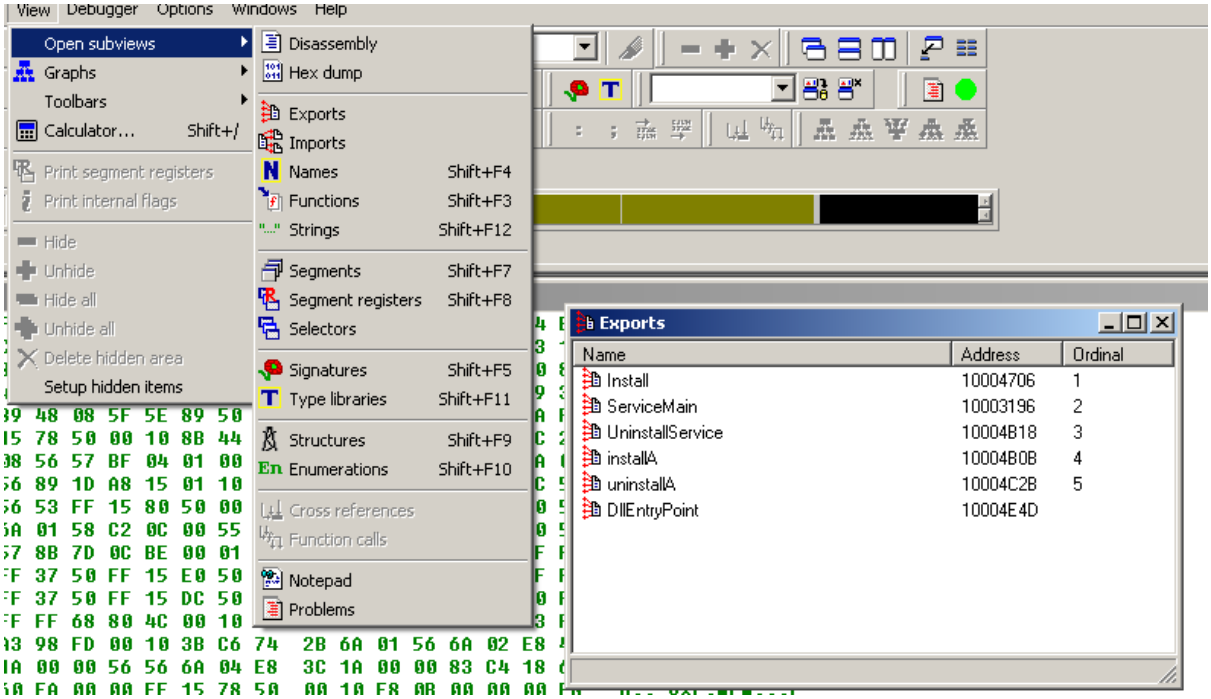


Figure 72- Exports

LoadLibrary API

After opening 'Jump to address' menu, I searched the API LoadLibraryA to find the imported function, then using 'jump to Xrefs to operand', I found that the list of all functions that call kernel32 API LoadLibrary, which is 1.

Figure 73- LoadLibrary API.

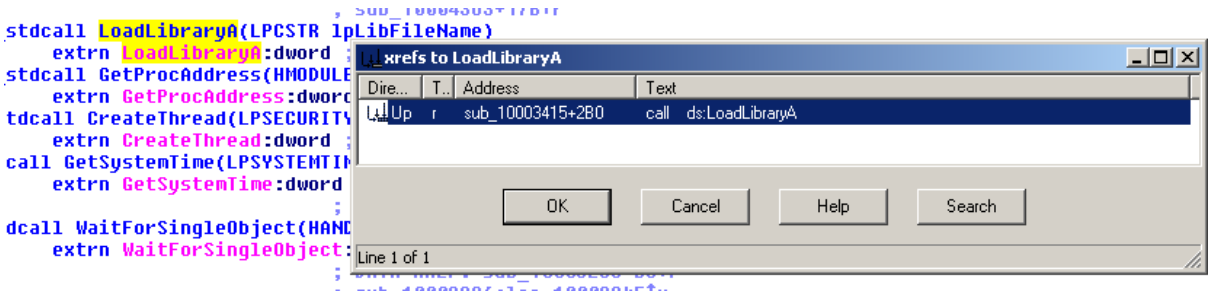


Figure 74- LoadLibrary Calls.

SLEEP CALL

Searching Sleep on the imported function tab and double-clicking on it takes me to the instance of Sleep within the IDS view.

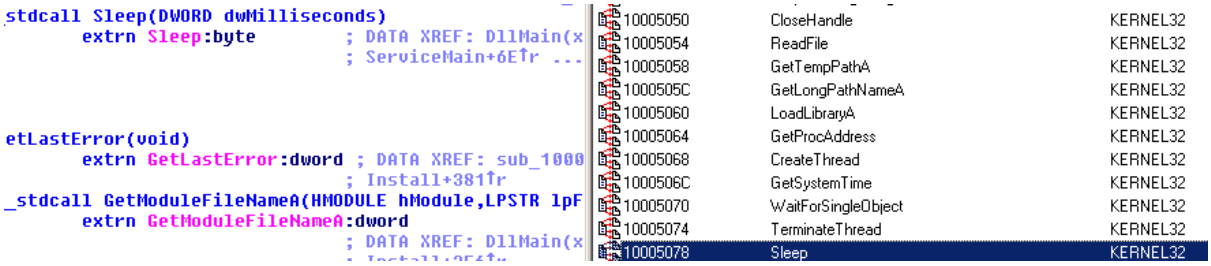


Figure 75- Sleep API.

Then right clicking on 'Sleep' and selecting 'jump to xref to perand..' takes me all instances were 'Sleep' is called, which happens to be a total of 9 times, by 7 different functions.

Figure 76-Sleep().

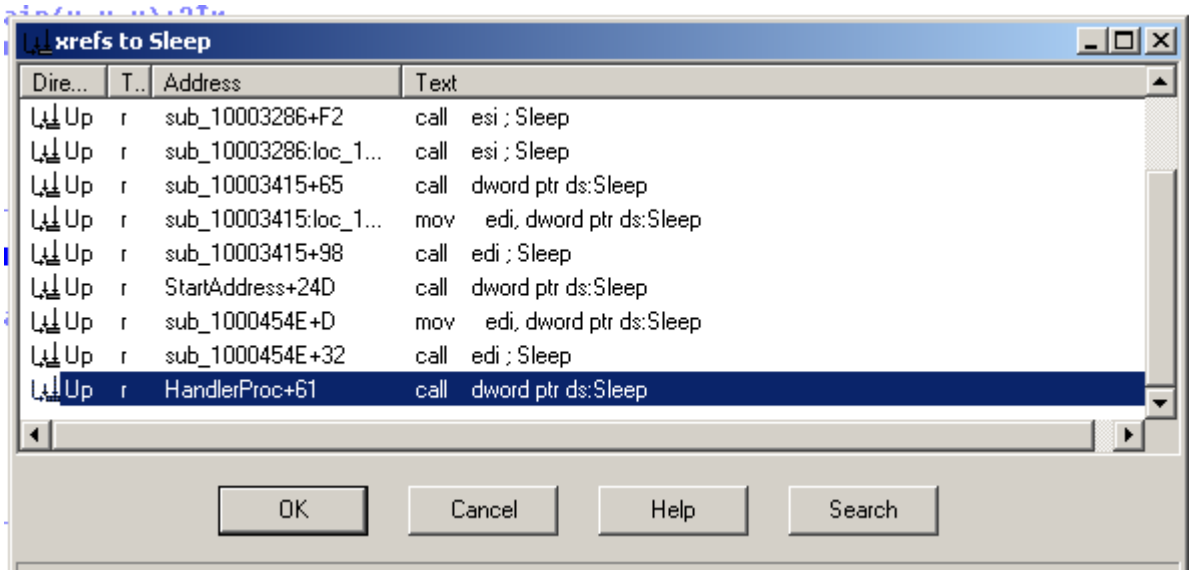


Figure 77- Sleep() Calls.

TASK 6

After opening malsample.dll in IDA pro and selecting the ServiceMain function I was able to open the graph view by using the view graph option at the top:

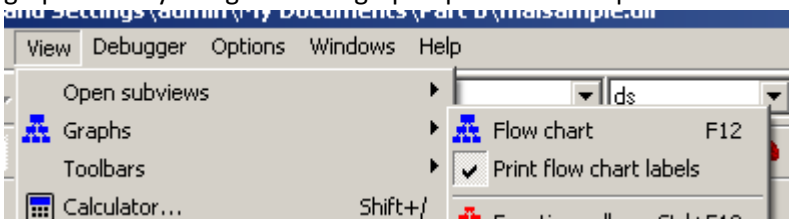
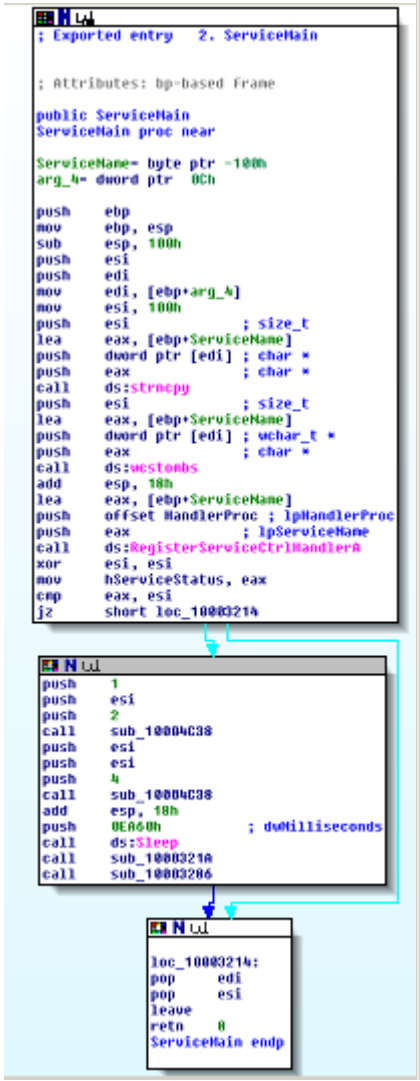


Figure 78-Graph view



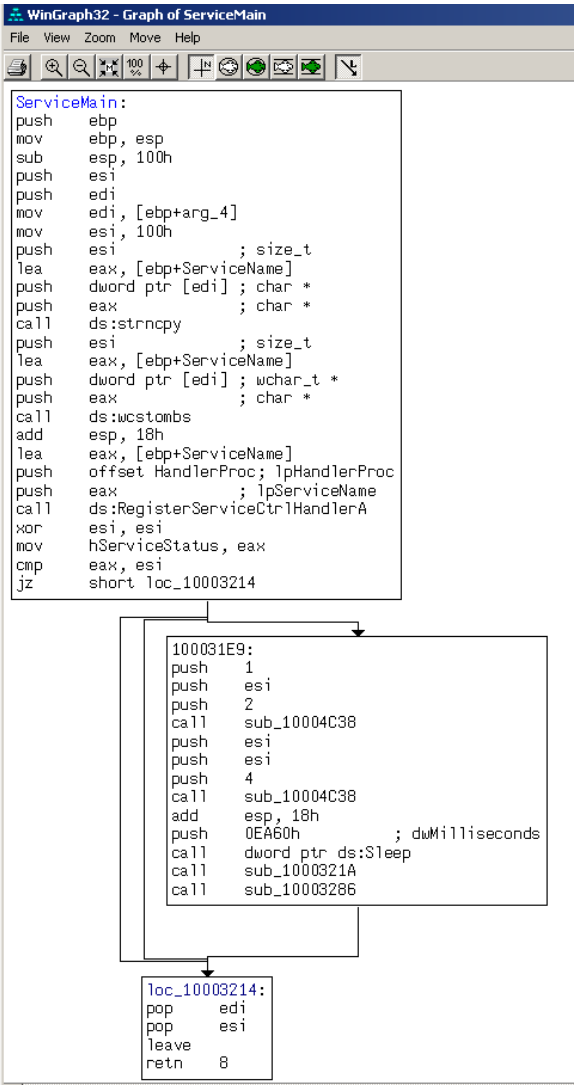


Figure 79-ServiceMain Function. *Figure 80-ServiceMain Graph.*

The code calls API Sleep() after JZ assembly instruction, and the parameter used by Sleep() is '0EA60h', this is defined by the structure of the Sleep function itself where we can see that it takes the 'dwMilliseconds' parameter, which in this case is '0EA60h'.

```
void __stdcall Sleep(DWORD dwMilliseconds)
                extrn Sleep:dword        ; DATA
```

Figure 81-Sleep() Parameters.

```
jz      short loc_10003214

push    1
push    esi
push    2
call    sub_10004C38
push    esi
push    esi
push    4
call    sub_10004C38
add     esp, 18h
push    0EA60h          ; dwMilliseconds
call    ds:Sleep
call    sub_1000321A
call    sub_10003286
```

Figure 82-Sleep() within ServiceMain.

References.

"Explorer Suite." NTCore, ntcore.com/?page_id=388. Accessed 10 May 2023.

Https://docs.remnux.org/ https://docs.remnux.org/. Https://Docs.Remnux.Org/Install-Distro/Get-Virtual-Appliance.

Marculescu, Ana. "ApateDNS (Windows) - Download & Review." Softpedia, 25 June 2013, www.softpedia.com/get/Network-Tools/Misc-Networking-Tools/ApateDNS.shtml#download.

Winitor, www.winitor.com/download. Accessed 10 May 2023.

Yara-Rules. "Yara-Rules/Rules: Repository of Yara Rules." GitHub, github.com/Yara-Rules/rules. Accessed 10 May 2023.

Yerima, Dr Suleiman. WINXP VM, demontfortuniversity-my.sharepoint.com/:u:/g/personal/syerim00_dmu_ac_uk/EWIt_QyDlFVHit4rtBhhWfMBMrjgMnLLDFESN6-mhV3FQA?e=5JHbM2.