# Malware Static Analysis Report

This report shows my steps to perform static analysis on game.exe malware
Every step I took is documented using screenshots

## Before Unpacking

Frist I get the hash of game .exe

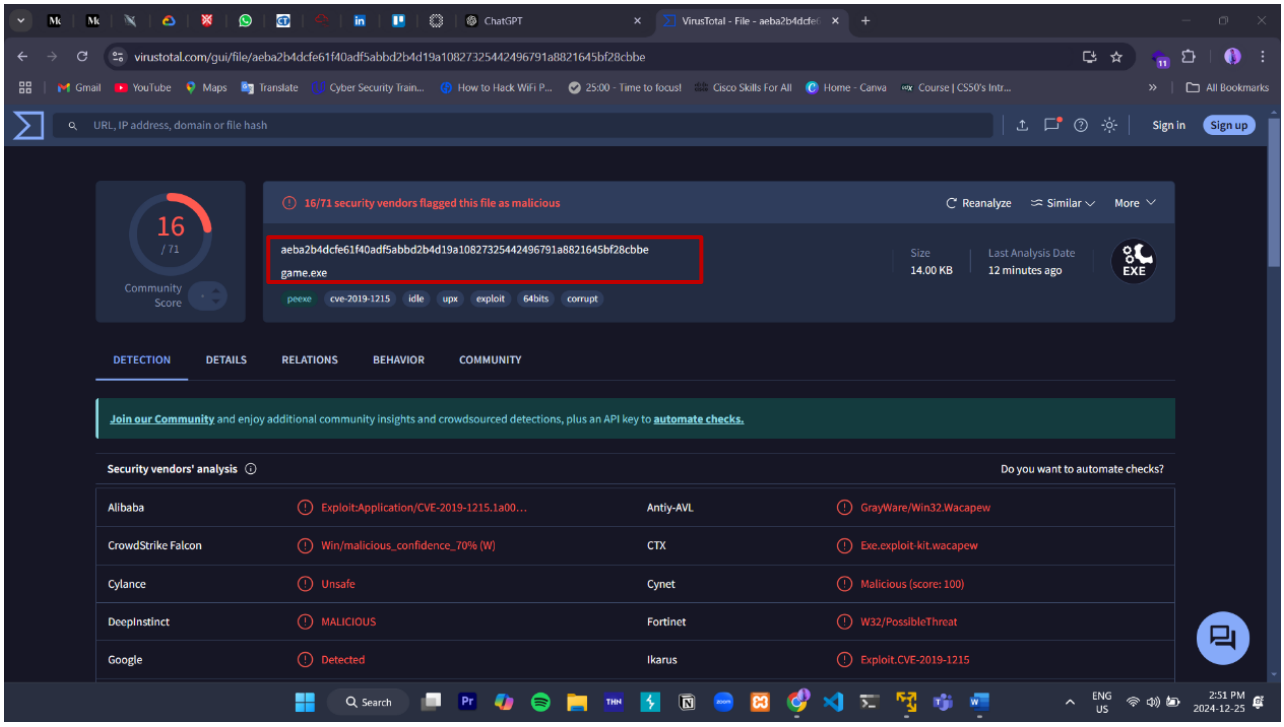
certutil -hashfile Game.exe SHA256

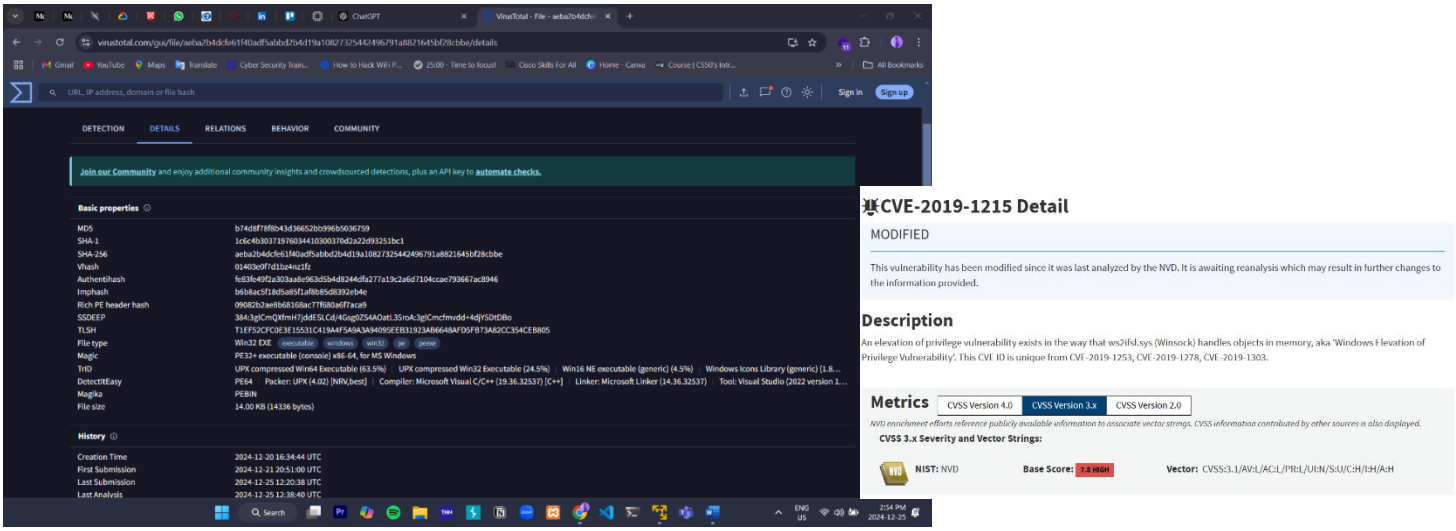
```
C:\Users\IEUser>certutil -hashfile C:\Users\IEUser\Desktop\Game.exe SHA256
SHA256 hash of C:\Users\IEUser\Desktop\Game.exe:
aeba2b4dcfe61f40adf5abbd2b4d19a10827325442496791a8821645bf28cbbe
CertUtil: -hashfile command completed successfully.
```

certutil -hashfile Game.exe MD5

```
C:\Users\IEUser>certutil -hashfile C:\Users\IEUser\Desktop\Game.exe MD5
MD5 hash of C:\Users\IEUser\Desktop\Game.exe:
b74d8f78f8b43d36652bb996b5036759
CertUtil: -hashfile command completed successfully.
```

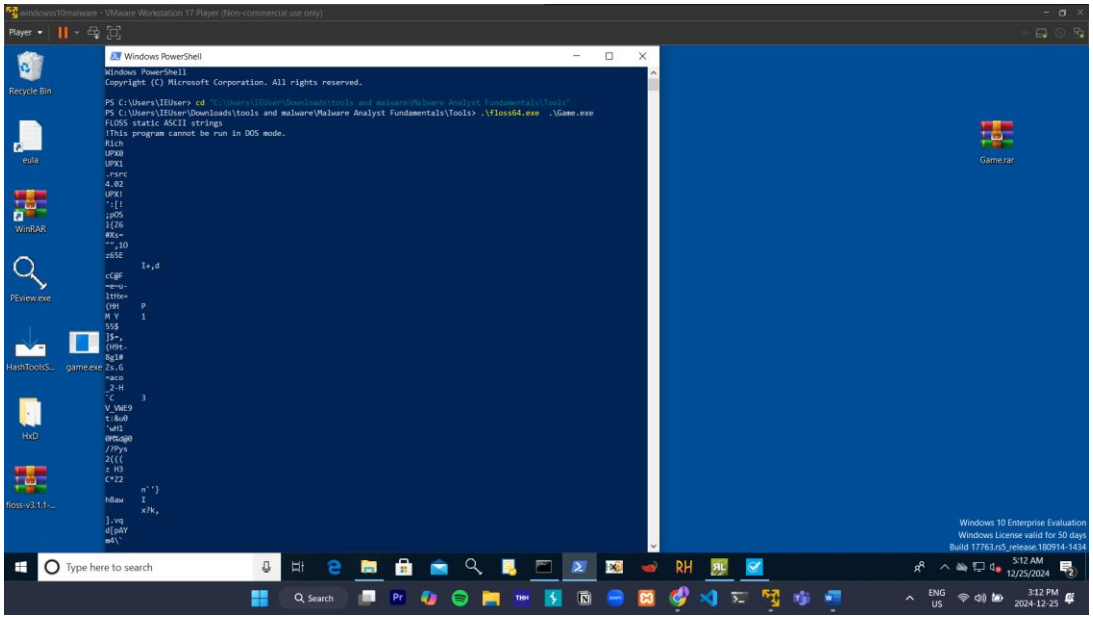
then I searched using the md5 hash on virus total






# Floss

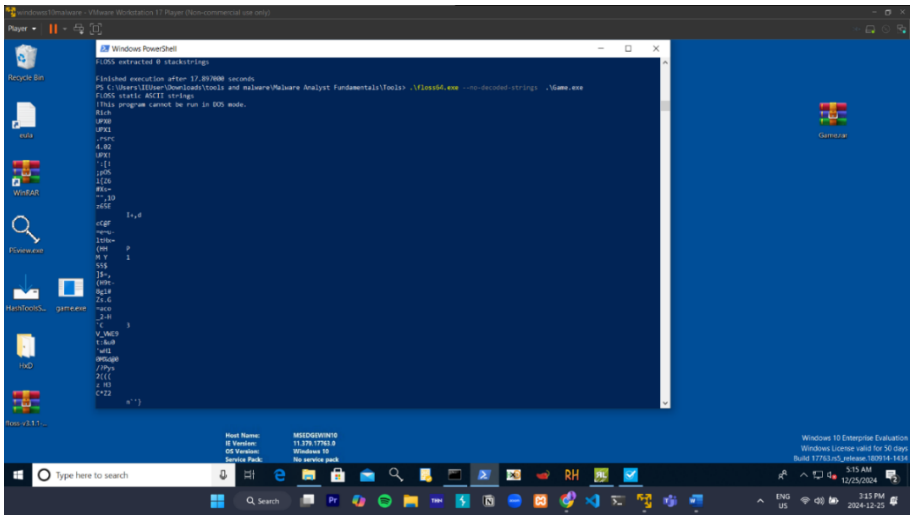
.\floss64.exe  .\Game.exe



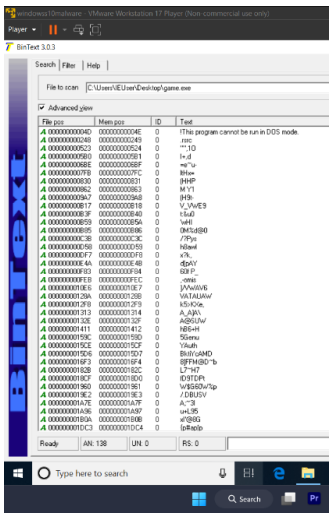

.\floss64.exe --no-decoded-strings .\Game.exe

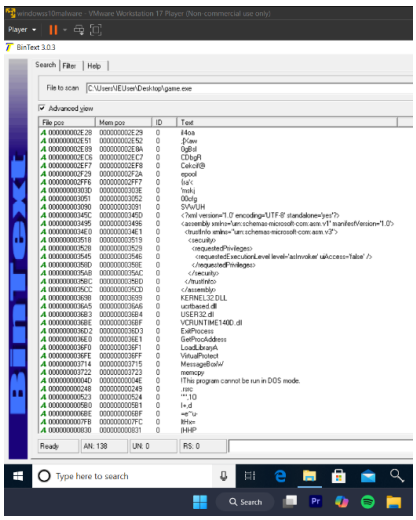
**We know malware is Packed with UPX**

The presence of strings like UPX0, UPX1, and UPX! indicates the binary might be packed using the UPX packer. Packers are often used to compress executables, but they can also be employed to obscure malware code.

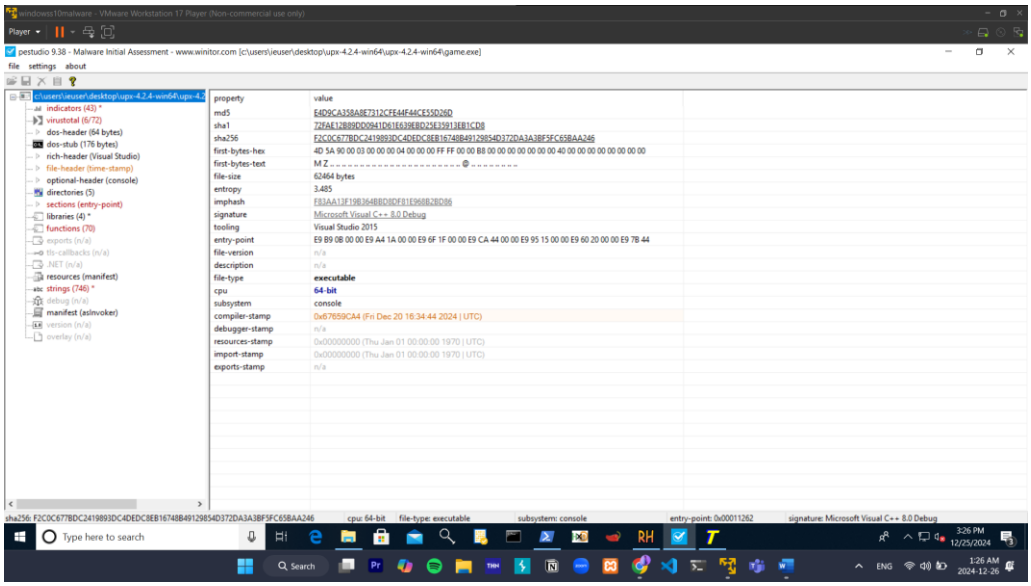


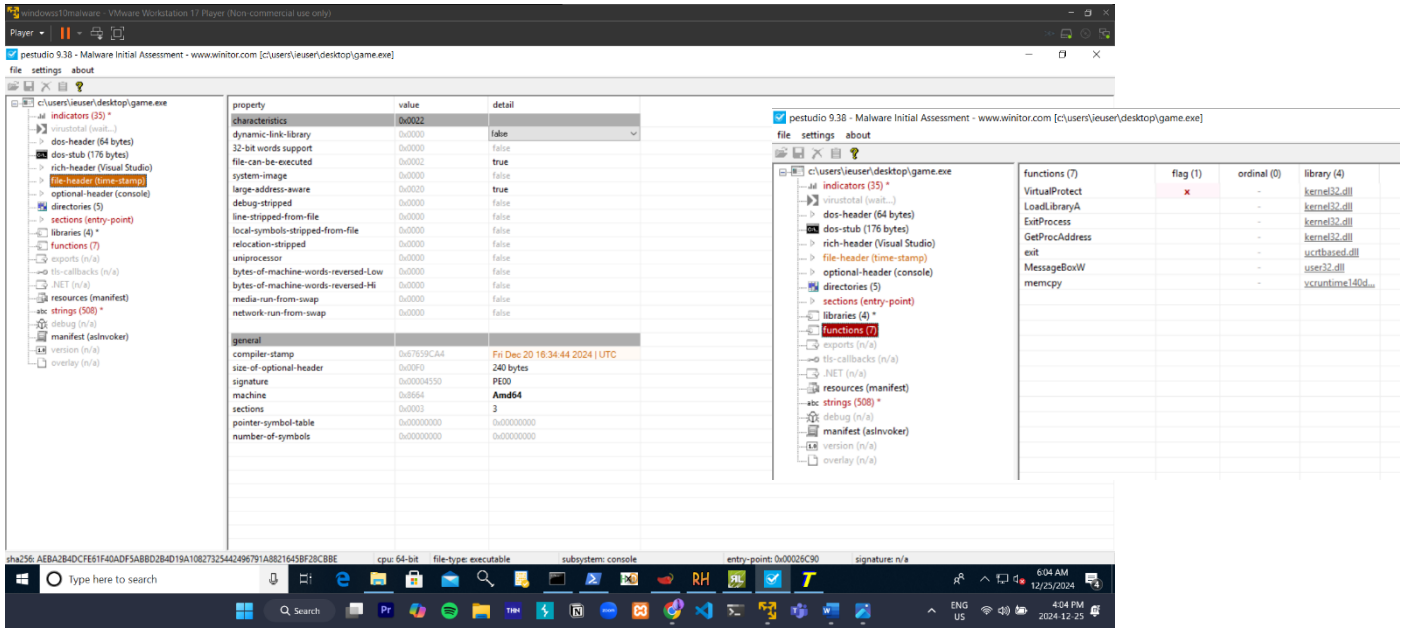
# Bintext

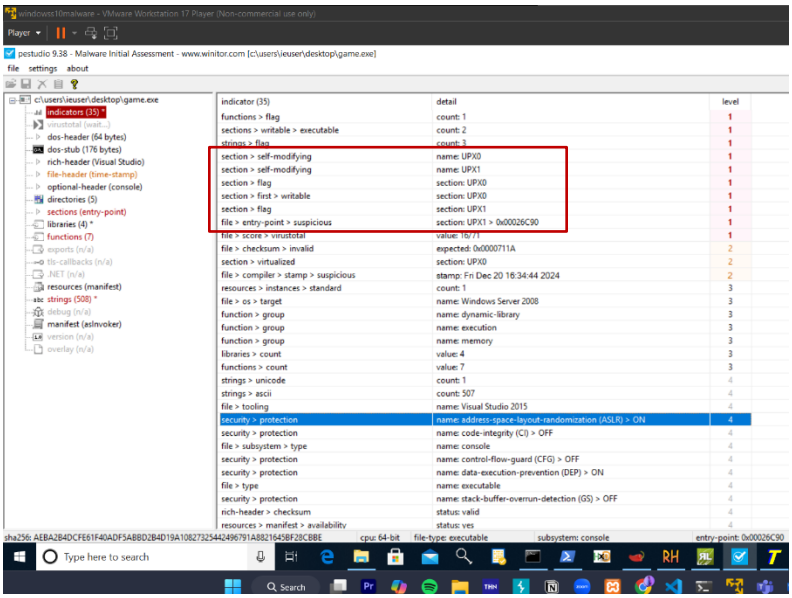


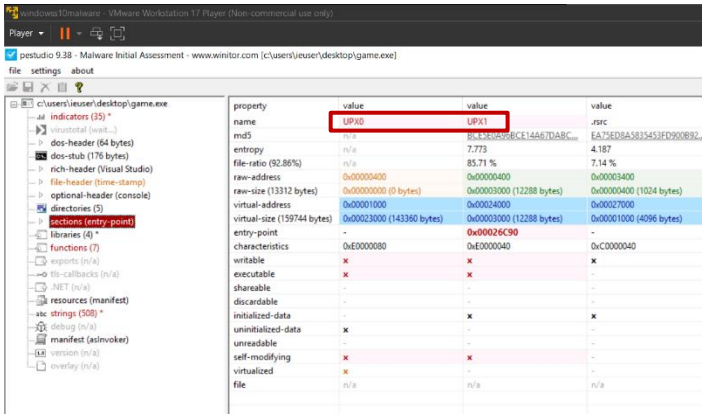


# Pestudio



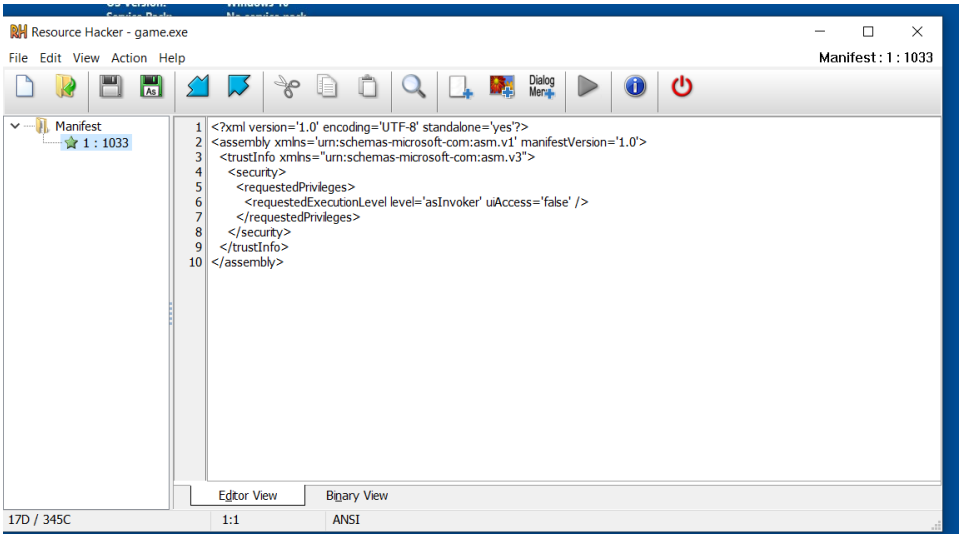



# Resource Hacker

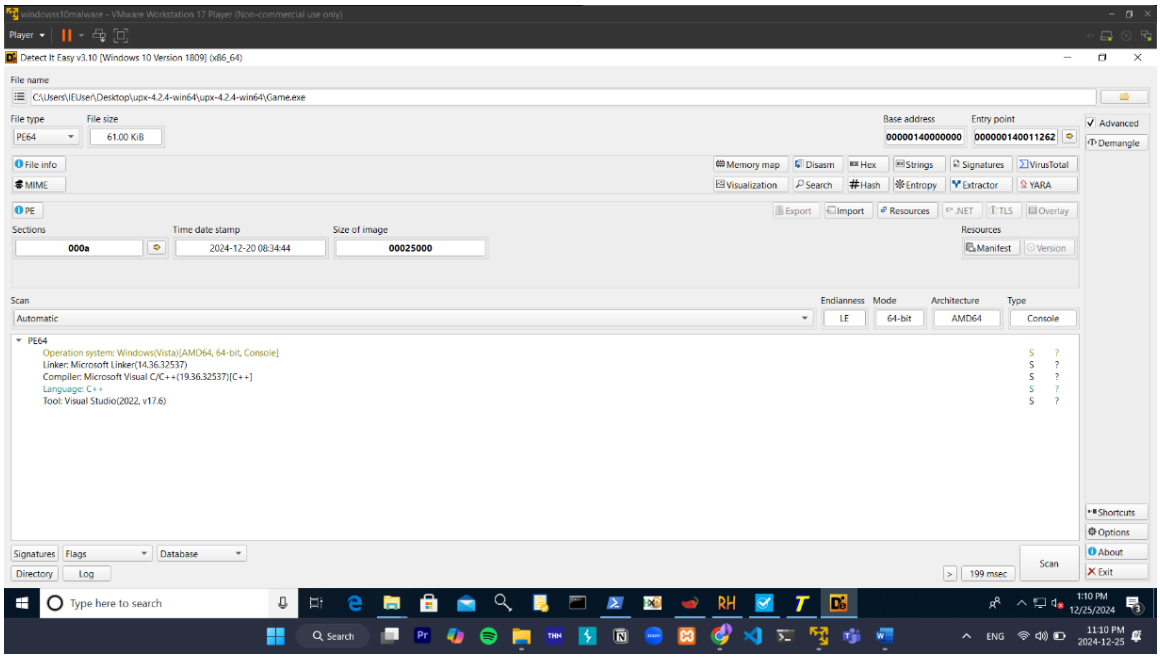
showing resource section



# After Unpacking

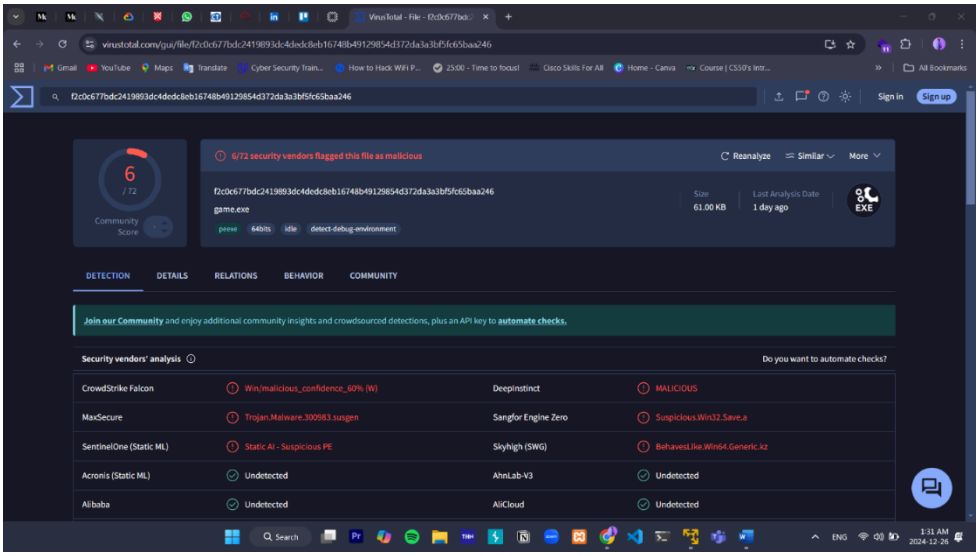
```
PS C:\Users\IEUser\Desktop\upx-4.2.4-win64\upx-4.2.4-win64> .\upx.exe -d Game.exe
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2024
UPX 4.2.4       Markus Oberhumer, Laszlo Molnar & John Reiser    May 9th 2024

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
     62464 <-     14336   22.95%    win64/pe     Game.exe

Unpacked 1 file.
PS C:\Users\IEUser\Desktop\upx-4.2.4-win64\upx-4.2.4-win64>
```

Making sure of unpacking

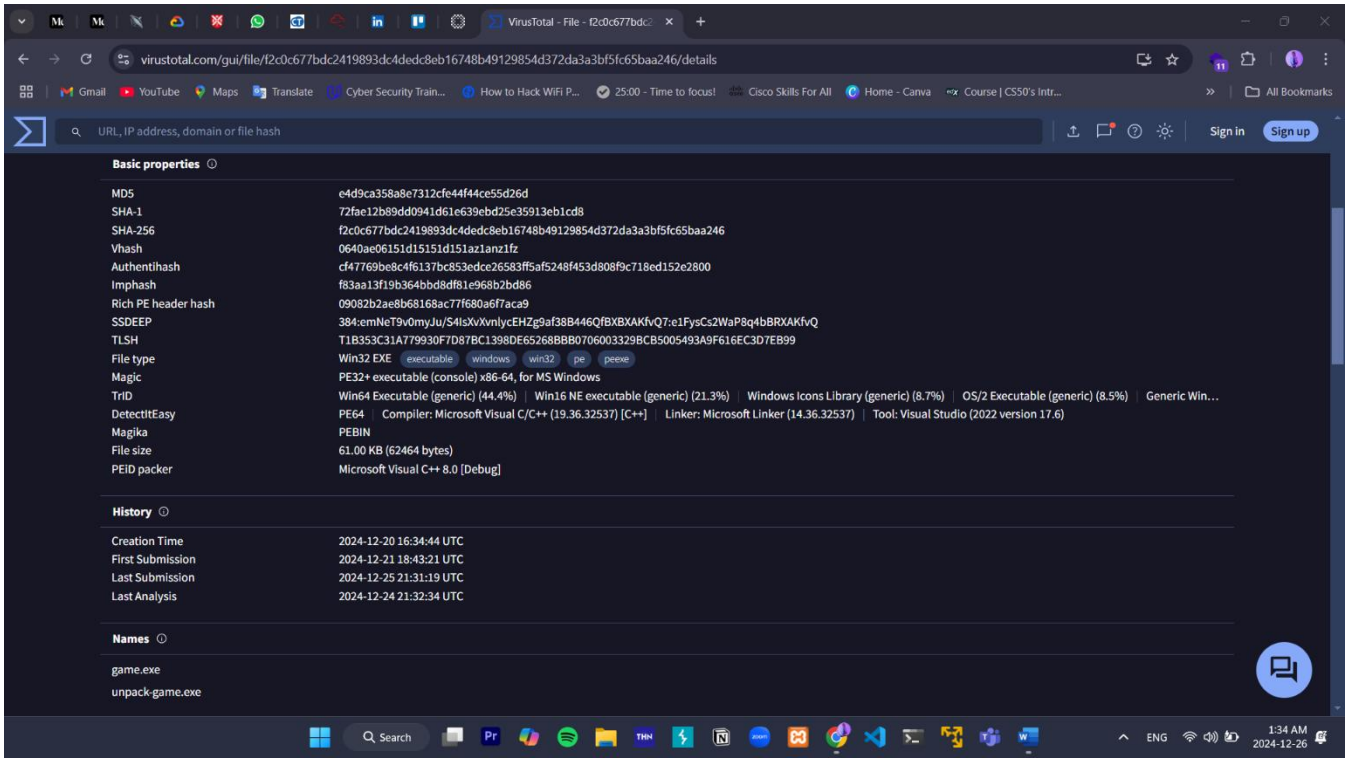



Then doing again all what I done before

```
PS C:\Users\IEUser\Desktop\upx-4.2.4-win64\upx-4.2.4-win64> certutil -hashfile Game.exe SHA256
SHA256 hash of Game.exe:
f2c0c677bdc2419893dc4dedc8eb16748b49129854d372da3a3bf5fc65baa246
CertUtil: -hashfile command completed successfully.
PS C:\Users\IEUser\Desktop\upx-4.2.4-win64\upx-4.2.4-win64> certutil -hashfile Game.exe MD5
MD5 hash of Game.exe:
e4d9ca358a8e7312cfe44f44ce55d26d
CertUtil: -hashfile command completed successfully.
```




The detection "Win/malicious_confidence_60% (W)" by CrowdStrike Falcon indicates that a file has been flagged as potentially malicious, with a 60% confidence level. This suggests that the file exhibits behaviors or characteristics commonly associated with malware. Trojan.Malware. 300983.susgen" is a detection name used by MaxSecure antivirus to identify files it considers potentially harmful
The term "susgen" likely stands for "suspicious generic," indicating that the antivirus has identified certain behaviors or characteristics in a file that are commonly associated with malware.



VirusTotal - File - f2c0c677bdc2

virustotal.com/gui/file/f2c0c677bdc2419893dc4dedc8eb16748b49129854d372da3a3bf5fc65baa246/details

URL, IP address, domain or file hash

Sign in | Sign up

**Basic properties**

| | |
|---|---|
| MD5 | e4d9ca358a8e7312cfe44f44ce55d26d |
| SHA-1 | 72fae12b89dd0941d61e639ebd25e35913eb1cd8 |
| SHA-256 | f2c0c677bdc2419893dc4dedc8eb16748b49129854d372da3a3bf5fc65baa246 |
| Vhash | 0640ae06151d15151d151az1anz1fz |
| Authentihash | cf47769be8c4f6137bc853edce26583ff5af5248f453d808f9c718ed152e2800 |
| Imphash | f83aa13f19b364bbd8df81e968b2bd86 |
| Rich PE header hash | 09082b2ae8b68168ac77f680a6f7aca9 |
| SSDEEP | 384:emNeT9v0myJu/S4IsXvXvnlycEHZg9af38B446QfBXBXAKfvQ7:e1FysCs2WaP8q4bBRXAKfvQ |
| TLSH | T1B353C31A779930F7D87BC1398DE65268BBB0706003329BCB5005493A9F616EC3D7EB99 |
| File type | Win32 EXE executable windows win32 pe peexe |
| Magic | PE32+ executable (console) x86-64, for MS Windows |
| TrID | Win64 Executable (generic) (44.4%) \| Win16 NE executable (generic) (21.3%) \| Windows Icons Library (generic) (8.7%) \| OS/2 Executable (generic) (8.5%) \| Generic Win… |
| DetectItEasy | PE64 \| Compiler: Microsoft Visual C/C++ (19.36.32537) [C++] \| Linker: Microsoft Linker (14.36.32537) \| Tool: Visual Studio (2022 version 17.6) |
| Magika | PEBIN |
| File size | 61.00 KB (62464 bytes) |
| PEiD packer | Microsoft Visual C++ 8.0 [Debug] |

**History**

| | |
|---|---|
| Creation Time | 2024-12-20 16:34:44 UTC |
| First Submission | 2024-12-21 18:43:21 UTC |
| Last Submission | 2024-12-25 21:31:19 UTC |
| Last Analysis | 2024-12-24 21:32:34 UTC |

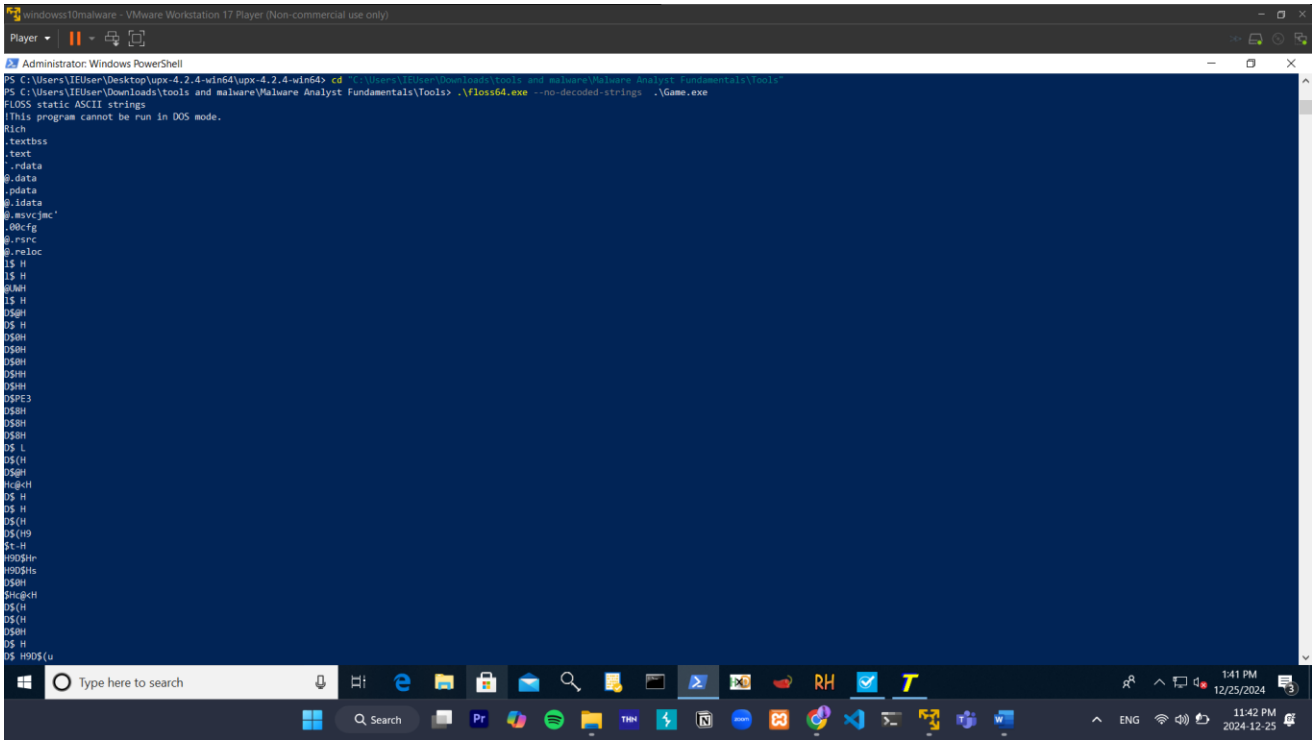
**Names**

game.exe

unpack-game.exe

# Floss

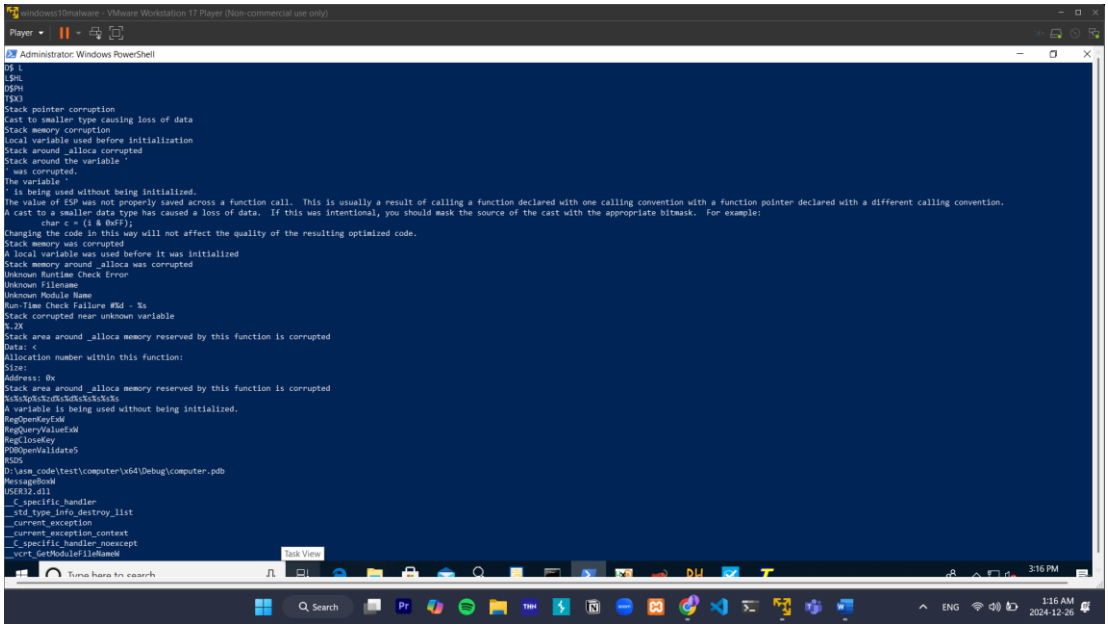
UPX does not appear again it means it is unpacked

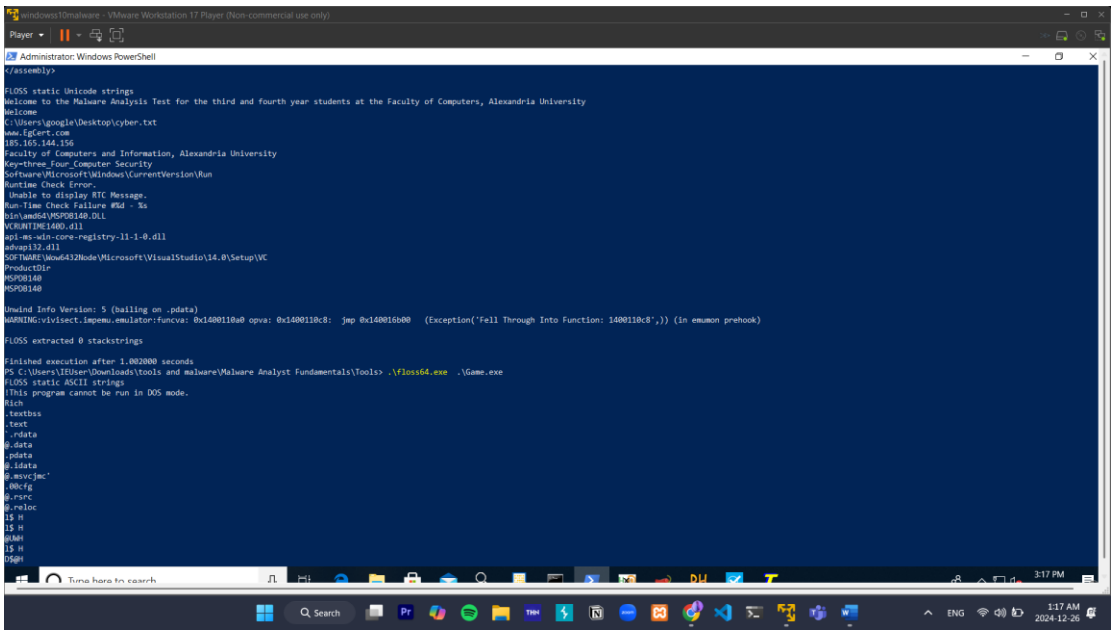


windowss10malware - VMware Workstation 17 Player (Non-commercial use only)

Administrator: Windows PowerShell

```
PS C:\Users\IEUser\Desktop\upx-4.2.4-win64\upx-4.2.4-win64> cd "C:\Users\IEUser\Downloads\tools and malware\Malware Analyst Fundamentals\Tools"
PS C:\Users\IEUser\Downloads\tools and malware\Malware Analyst Fundamentals\Tools> .\floss64.exe --no-decoded-strings  .\Game.exe
FLOSS static ASCII strings
!This program cannot be run in DOS mode.
Rich
.textbss
.text
`.rdata
@.data
.pdata
@.idata
@.msvcjmc'
.00cfg
@.rsrc
@.reloc
1$ H
1$ H
@AWH
1$ H
D$@H
D$ H
D$0H
D$0H
D$0H
D$HH
D$HH
D$PE3
D$8H
D$8H
D$8H
D$ L
D$(H
D$@H
Hc@<H
D$ H
D$ H
D$(H
D$(H9
$t-H
H9D$Hr
H9D$Hs
D$0H
$Hc@<H
D$(H
D$(H
D$0H
D$ H
D$ H9D$(u
```



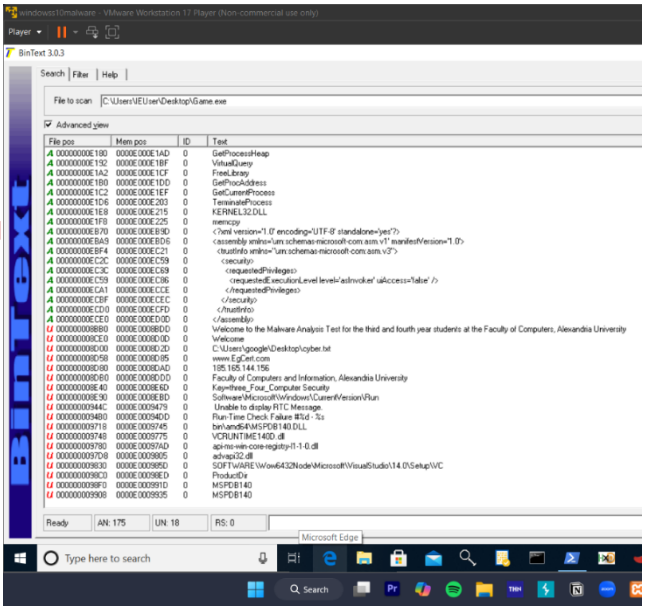

```
FLOSS static Unicode strings
Welcome to the Malware Analysis Test for the third and fourth year students at the Faculty of Computers, Alexandria University
Welcome
C:\Users\google\Desktop\cyber.txt
www.EgCert.com
185.165.144.156
Faculty of Computers and Information, Alexandria University
Key=three_Four_Computer Security
Software\Microsoft\Windows\CurrentVersion\Run
Runtime Check Error.
 Unable to display RTC Message.
Run-Time Check Failure #%d - %s
bin\amd64\MSPDB140.DLL
VCRUNTIME140D.dll
api-ms-win-core-registry-l1-1-0.dll
advapi32.dll
SOFTWARE\Wow6432Node\Microsoft\VisualStudio\14.0\Setup\VC
ProductDir
MSPDB140
MSPDB140

Unwind Info Version: 5 (bailing on .pdata)
WARNING:vivisect.impemu.emulator:funcva: 0x1400110a0 opva: 0x1400110c8:  jmp 0x140016b00   (Exception('Fell Through Into Function: 1400110c8',)) (in emumon prehook)

FLOSS decoded 0 strings

FLOSS extracted 0 stackstrings

Finished execution after 1.327000 seconds
```

# Bintext

When analyzing using bintext certain string patterns and symbols provided critical insights into the behavior and origin of the malware. Strings like "This program cannot be run in DOS mode" or function names (e.g., MessageBoxW, USER32.dll) can indicate a PE (Portable Executable) file typical of Windows malware

Also Strings such as "Stack memory corruption" or "Local variable used before initialization" suggest potential vulnerabilities or behaviors the malware might exploit.

| File pos | Mem pos | ID | Text |
|---|---|---|---|
| A 000000000436B | 0000E0004398 | 0 | L$0H3 |
| A 000000004733 | 0000E0004760 | 0 | D$H9D$ s" |
| A 0000000047E8 | 0000E0004815 | 0 | D$HE3 |
| A 000000008F30 | 0000E0008F5D | 0 | Stack pointer corruption |
| A 000000008F50 | 0000E0008F7D | 0 | Cast to smaller type causing loss of data |
| A 000000008F88 | 0000E0008FB5 | 0 | Stack memory corruption |
| A 000000008FA8 | 0000E0008FD5 | 0 | Local variable used before initialization |
| A 000000008FE0 | 0000E000900D | 0 | Stack around _alloca corrupted |
| A 000000009050 | 0000E000907D | 0 | Stack around the variable ' |
| A 000000009070 | 0000E000909D | 0 | ' was corrupted. |
| A 000000009088 | 0000E00090B5 | 0 | The variable ' |
| A 000000009098 | 0000E00090C5 | 0 | ' is being used without being initialized. |
| A 0000000090F0 | 0000E000911D | 0 | The value of ESP was not properly saved across a function call. This is usually a result of calling a |
| A 000000009200 | 0000E000922D | 0 | A cast to a smaller data type has caused a loss of data. If this was intentional, you should mask the |
| A 0000000092A9 | 0000E00092D6 | 0 | char c = (i & 0xFF); |
| A 0000000092C0 | 0000E00092ED | 0 | Changing the code in this way will not affect the quality of the resulting optimized code. |
| A 000000009358 | 0000E0009385 | 0 | Stack memory was corrupted |
| A 000000009380 | 0000E00093AD | 0 | A local variable was used before it was initialized |
| A 0000000093C0 | 0000E00093ED | 0 | Stack memory around _alloca was corrupted |
| A 0000000093F8 | 0000E0009425 | 0 | Unknown Runtime Check Error |
| A 000000009500 | 0000E000952D | 0 | Unknown Filename |
| A 000000009518 | 0000E0009545 | 0 | Unknown Module Name |
| A 000000009530 | 0000E000955D | 0 | Run-Time Check Failure #%d - %s |
| A 000000009558 | 0000E0009585 | 0 | Stack corrupted near unknown variable |
| A 000000009588 | 0000E00095B5 | 0 | %.2X |
| A 000000009590 | 0000E00095BD | 0 | Stack area around _alloca memory reserved by this function is corrupted |
| A 0000000095F1 | 0000E000961E | 0 | Data: < |
| A 000000009601 | 0000E000962E | 0 | Allocation number within this function: |
| A 000000009639 | 0000E0009666 | 0 | Size: |
| A 000000009649 | 0000E0009676 | 0 | Address: 0x |
| A 000000009660 | 0000E000968D | 0 | Stack area around _alloca memory reserved by this function is corrupted |
| A 0000000096B8 | 0000E00096E5 | 0 | %s%s%p%s%zd%s%d%s%s%s%s |
| A 0000000096D8 | 0000E0009705 | 0 | A variable is being used without being initialized. |
| A 0000000097F8 | 0000E0009825 | 0 | RegOpenKeyExW |
| A 000000009808 | 0000E0009835 | 0 | RegQueryValueExW |
| A 000000009820 | 0000E000984D | 0 | RegCloseKey |
| A 000000009920 | 0000E000994D | 0 | PDBOpenValidate5 |
| A 00000000A034 | 0000E000A061 | 0 | D:\asm_code\test\computer\x64\Debug\computer.pdb |
| A 00000000DC4A | 0000E000DC77 | 0 | MessageBoxW |
| A 00000000DC56 | 0000E000DC83 | 0 | USER32.dll |
| A 00000000DC64 | 0000E000DC91 | 0 | __C_specific_handler |
| A 00000000DC7C | 0000E000DCA9 | 0 | __std_type_info_destroy_list |
| A 00000000DC9C | 0000E000DCC9 | 0 | __current_exception |
| A 00000000DCB2 | 0000E000DCDF | 0 | __current_exception_context |
| A 00000000DCD0 | 0000E000DCFD | 0 | __C_specific_handler_noexcept |
| A 00000000DCF0 | 0000E000DD1D | 0 | __vcrt_GetModuleFileNameW |




File to scan C:\Users\IEUser\Desktop\Game.exe

Advanced view

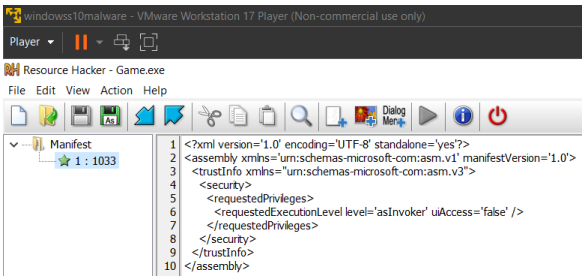
| File pos | Mem pos | ID | Text |
|---|---|---|---|
| A 00000000004D | 00000000004E | 0 | !This program cannot be run in DOS mode. |
| A 0000000001F8 | 0000E0000225 | 0 | .textbss |
| A 000000000220 | 0000E000024D | 0 | .text |

File paths like D:\asm_code\test\computer\x64\Debug\computer.pdb might reveal debug or developer information that could indicate the source of the binaryPDB (Program Database) paths can provide insight into the malware author's environment, potentially exposing developer tools or environments.
Malicious code frequently interacts with low-level system components through uncommon or suspicious API calls. Functions such as RegOpenKeyExW and RegQueryValueExW are indicative of malware engaging in registry manipulation, a common method used for persistence or configuration modification on infected systems.
function names such as __C_specific_handler, __std_type_info_destroy_list, and __current_exception are related to exception handling. Malware often manipulates these to bypass standard error detection.
References to libraries like USER32.dll indicate interaction with the Windows GUI, often used by malware to display deceptive messages or prompts designed to mislead the user into taking unsafe actions.
Unreadable strings like D$H9D$s" or L$@H3 may indicate attempts at obfuscation, which is a common tactic used to avoid detection also strings like "The value of ESP was not properly saved across a function call" can indicate exploitation techniques, such as stack manipulation or buffer overflow.

# Resource hacker

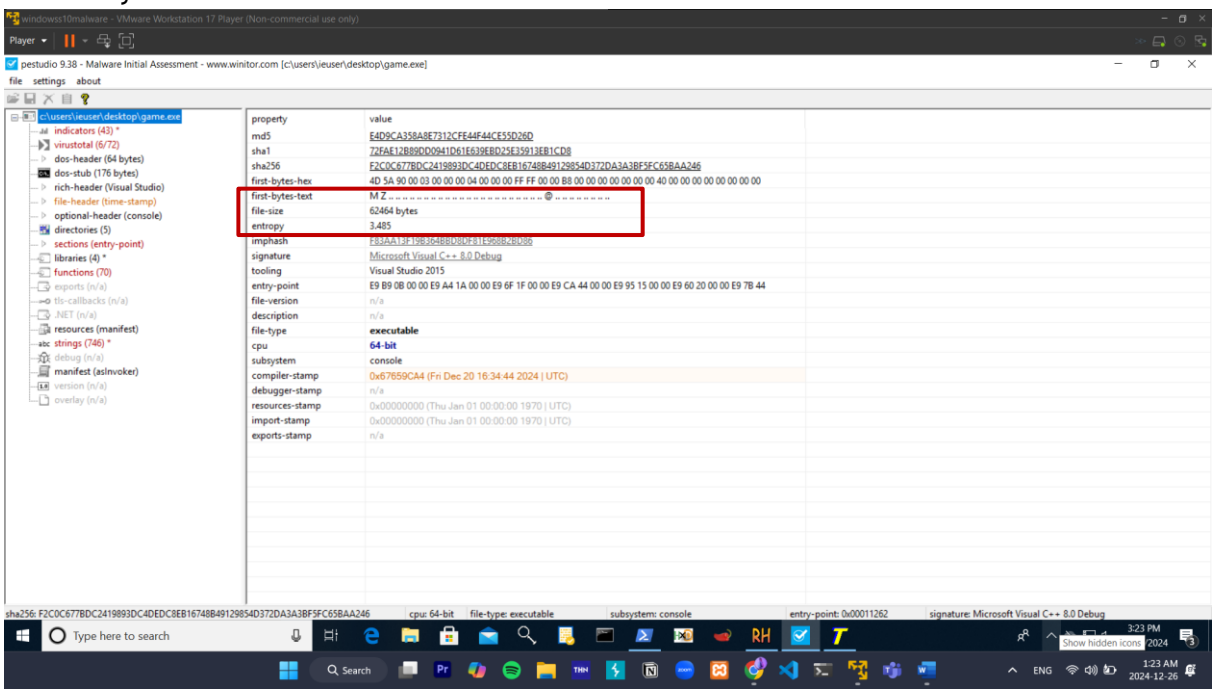
The XML snippet is part of the **manifest file** of Game.exe This file defines metadata about the application, such as its required privileges and compatibility settings

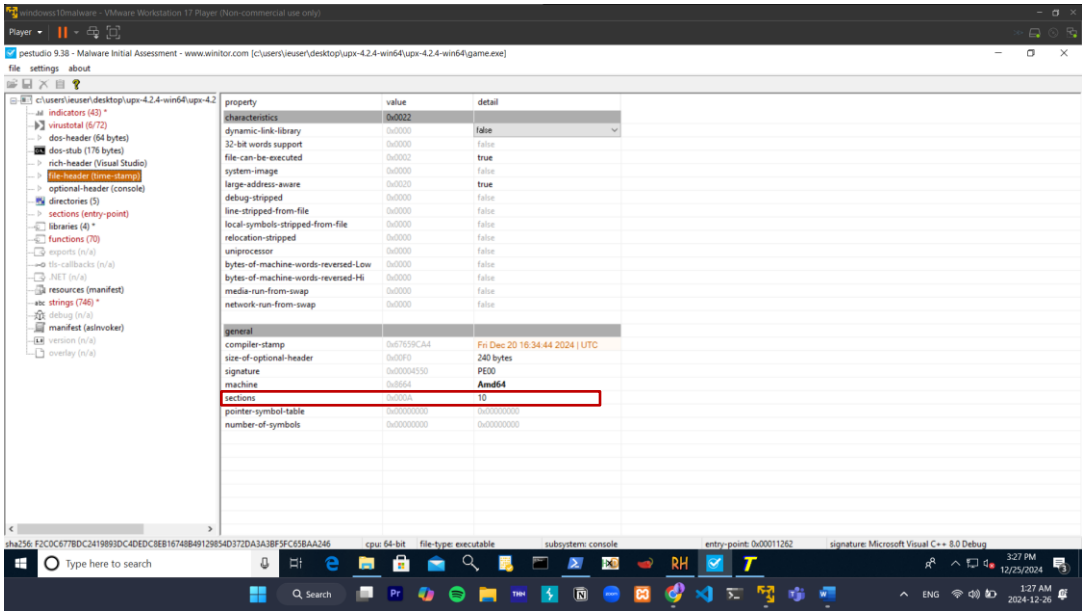
The manifest is clean here, but malware might embed additional data or configurations in other resource sections.

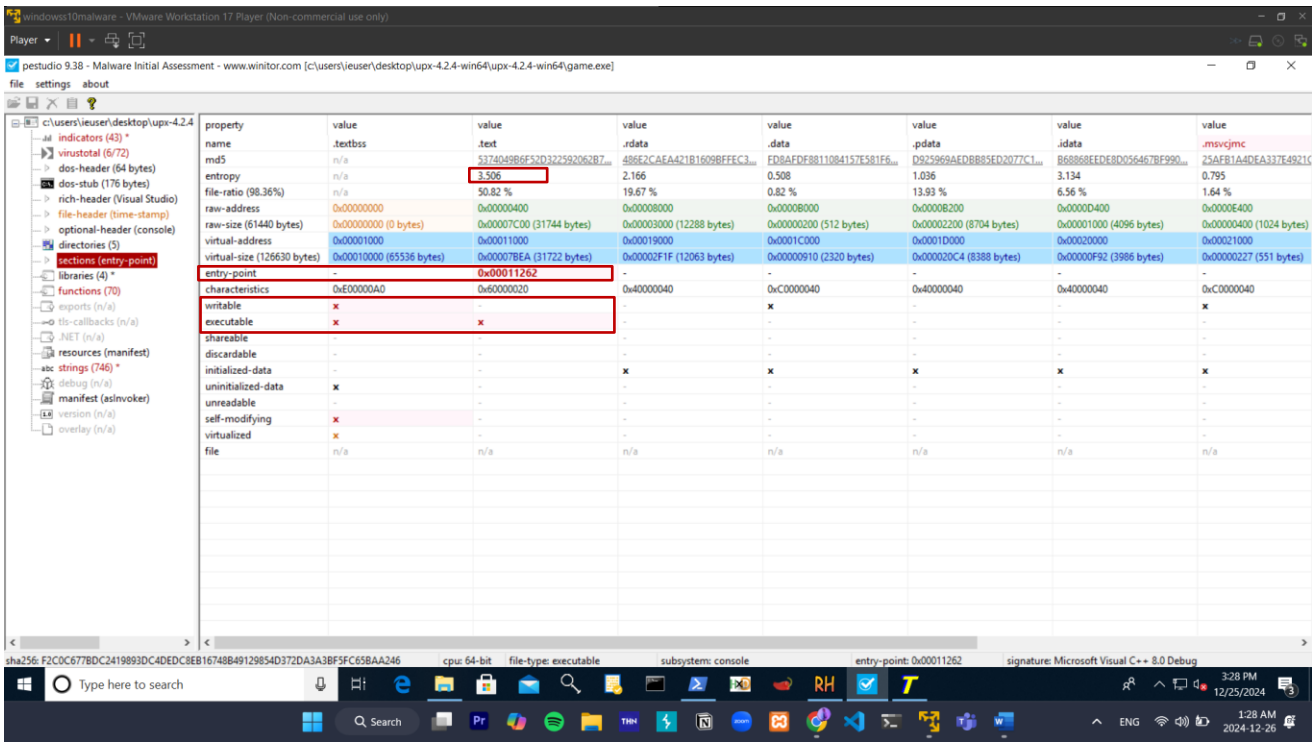



# Pestudio

It gives me SHA256 and MD5 Hashes and The compiler timestamp can help determine when the file was created also that the file type is executable file with a 64-bit architecture, which suggests it targets modern systems.

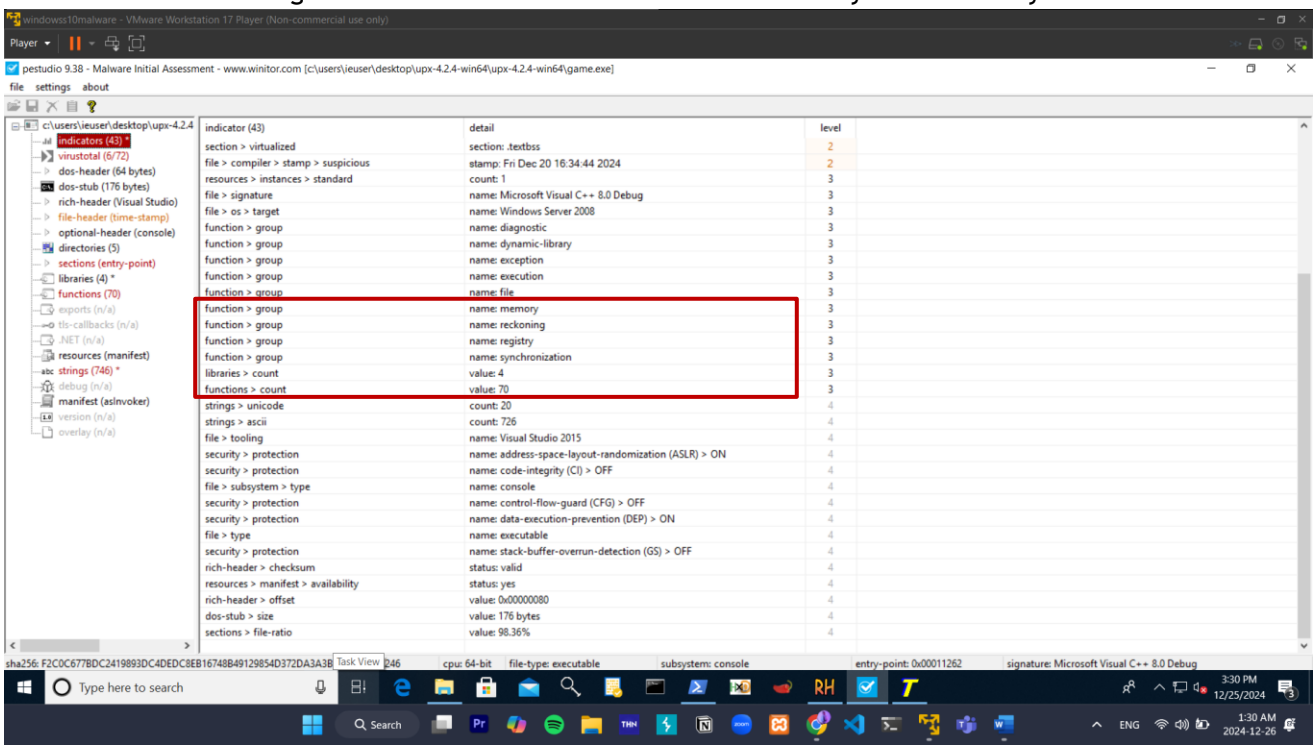


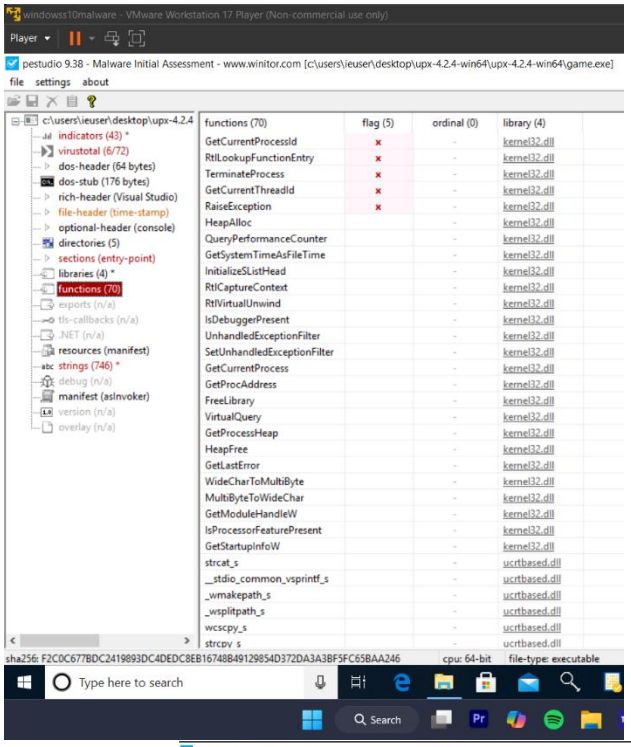


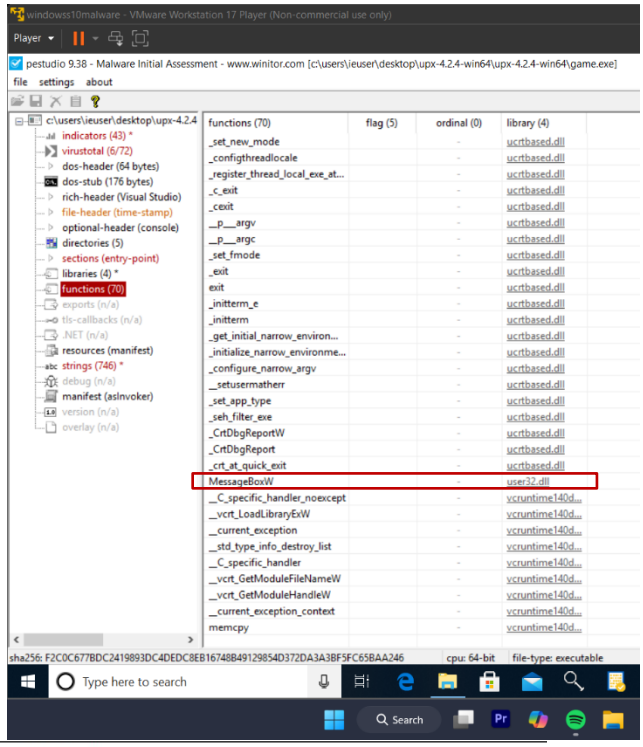

| property | value | value | value | value | value | value | value |
|---|---|---|---|---|---|---|---|
| name | .textbss | .text | .rdata | .data | .pdata | .idata | .msvcjmc |
| md5 | n/a | 5374049B6F52D32592062B7... | 486E2CAEA421B1609BFFEC3... | FD8AFDF8811084157E581F6... | D925969AEDBB85ED2077C1... | B68868EEDE8D056467BF990... | 25AFB1A4DEA337E4921C |
| entropy | n/a | 3.506 | 2.166 | 0.508 | 1.036 | 3.134 | 0.795 |
| file-ratio (98.36%) | n/a | 50.82 % | 19.67 % | 0.82 % | 13.93 % | 6.56 % | 1.64 % |
| raw-address | 0x00000000 | 0x00000400 | 0x00008000 | 0x0000B000 | 0x0000B200 | 0x0000D400 | 0x0000E400 |
| raw-size (61440 bytes) | 0x00000000 (0 bytes) | 0x00007C00 (31744 bytes) | 0x00003000 (12288 bytes) | 0x00000200 (512 bytes) | 0x00002200 (8704 bytes) | 0x00001000 (4096 bytes) | 0x00000400 (1024 bytes) |
| virtual-address | 0x00001000 | 0x00011000 | 0x00019000 | 0x0001C000 | 0x0001D000 | 0x00020000 | 0x00021000 |
| virtual-size (126630 bytes) | 0x00010000 (65536 bytes) | 0x00007BEA (31722 bytes) | 0x00002F1F (12063 bytes) | 0x00000910 (2320 bytes) | 0x000020C4 (8388 bytes) | 0x00000F92 (3986 bytes) | 0x00000227 (551 bytes) |
| entry-point | - | 0x00011262 | - | - | - | - | - |
| characteristics | 0xE00000A0 | 0x60000020 | 0x40000040 | 0xC0000040 | 0x40000040 | 0x40000040 | 0xC0000040 |
| writable | x | - | - | x | - | - | x |
| executable | x | x | - | - | - | - | - |
| shareable | - | - | - | - | - | - | - |
| discardable | - | - | - | - | - | - | - |
| initialized-data | - | - | x | x | x | x | x |
| uninitialized-data | x | - | - | - | - | - | - |
| unreadable | - | - | - | - | - | - | - |
| self-modifying | x | - | - | - | - | - | - |
| virtualized | x | - | - | - | - | - | - |
| file | n/a | n/a | n/a | n/a | n/a | n/a | n/a |

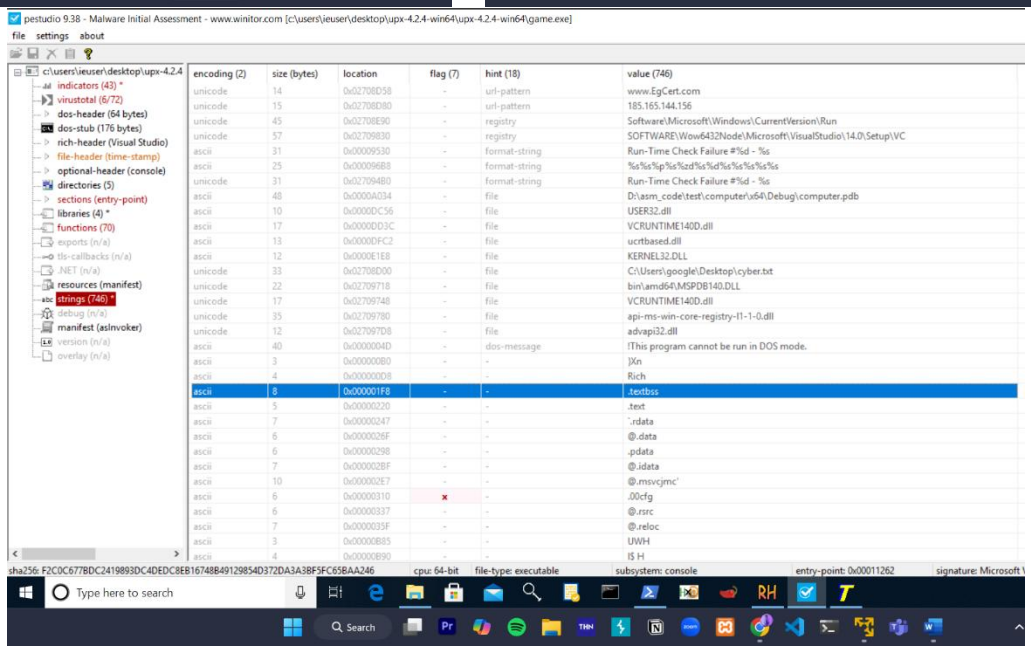
High entropy in .text (3.506) compared to entropy in other sections
The .text section shows a large size (31,744 bytes virtual vs. 65,536 raw). This discrepancy might indicate packing, as the raw size is reduced due to compression Sections flagged as executable and writable are concerning because malware can execute code directly from memory




| indicator (43) | detail | level |
|---|---|---|
| section > virtualized | section: .textbss | 2 |
| file > compiler > stamp > suspicious | stamp: Fri Dec 20 16:34:44 2024 | 2 |
| resources > instances > standard | count: 1 | 3 |
| file > signature | name: Microsoft Visual C++ 8.0 Debug | 3 |
| file > os > target | name: Windows Server 2008 | 3 |
| function > group | name: diagnostic | 3 |
| function > group | name: dynamic-library | 3 |
| function > group | name: exception | 3 |
| function > group | name: execution | 3 |
| function > group | name: file | 3 |
| function > group | name: memory | 3 |
| function > group | name: reckoning | 3 |
| function > group | name: registry | 3 |
| function > group | name: synchronization | 3 |
| libraries > count | value: 4 | 3 |
| functions > count | value: 70 | 3 |
| strings > unicode | count: 20 | 4 |
| strings > ascii | count: 726 | 4 |
| file > tooling | name: Visual Studio 2015 | 4 |
| security > protection | name: address-space-layout-randomization (ASLR) > ON | 4 |
| security > protection | name: code-integrity (CI) > OFF | 4 |
| file > subsystem > type | name: console | 4 |
| security > protection | name: control-flow-guard (CFG) > OFF | 4 |
| security > protection | name: data-execution-prevention (DEP) > ON | 4 |
| file > type | name: executable | 4 |
| security > protection | name: stack-buffer-overrun-detection (GS) > OFF | 4 |
| rich-header > checksum | status: valid | 4 |
| resources > manifest > availability | status: yes | 4 |
| rich-header > offset | value: 0x00000080 | 4 |
| dos-stub > size | value: 176 bytes | 4 |
| sections > file-ratio | value: 98.36% | 4 |




# Conclusion

The static examination of game.exe validates it as a harmful executable, recognized via hash checking on VirusTotal. The existence of UPX packer strings emphasizes its application of packing to conceal code and avoid being detected. Prior to unpacking the strings disclosed minimal details but unpacking uncovered references such as MessageBoxW, USER32.dll, and file paths indicating interaction with the Windows GUI and the potential exploitation of system vulnerabilities. PEStudio identified high entropy in sections may detect encryption or obfuscation, with suspicious API usage pointing to possible registry alterations or memory exploitation. the manifest is clear also additional examination is necessary to dig more to find harmful data.