Agent Tesla Malware Analysis

cyber-forensics.blog/2024/05/06/formbook-analysis/

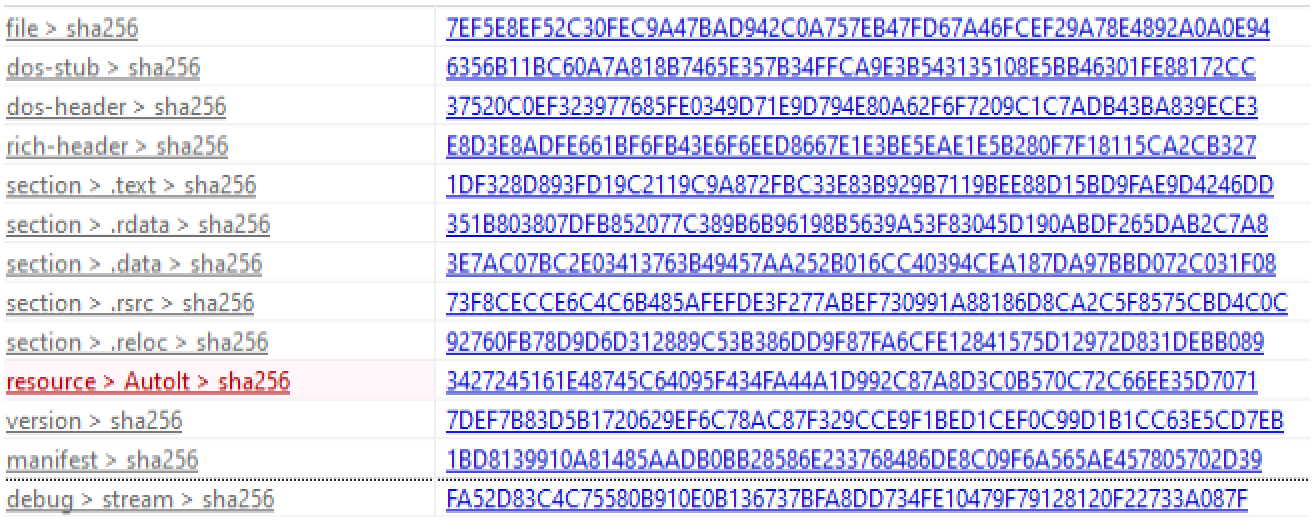
May 6, 2024

Hi, welcome to my first public analysis for malware. I spent most of the weekend analyzing the Agent Tesla malware. From reading other malware analysis of Agent Tesla it seems that it's been in the wild for a little over 5 years and is mainly spread through phishing campaigns. It is also a well known commercial malware that is being sold as a service. My analysis is broken into 2 parts, static and dynamic analysis. If you find any errors or just want to get into contact, my email is in the about section any recommendations or suggestions are always welcome! Anyways without further ado, here is the analysis.

Download:
https://bazaar.abuse.ch/download/7ef5e8ef52c30fec9a47bad942c0a757eb47fd67a46fcef29a78e4892a0a0e94/

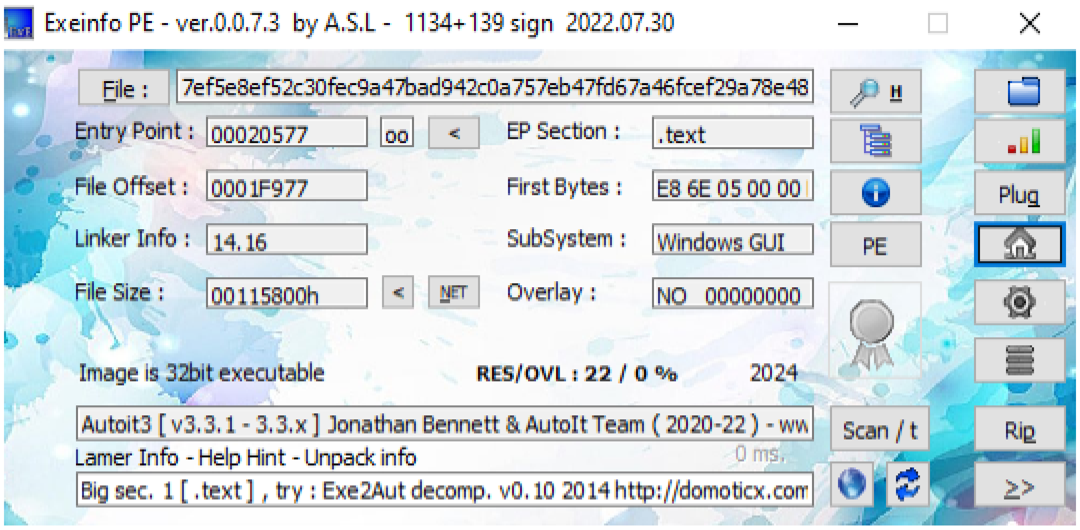
Static analysis:

Hashes:



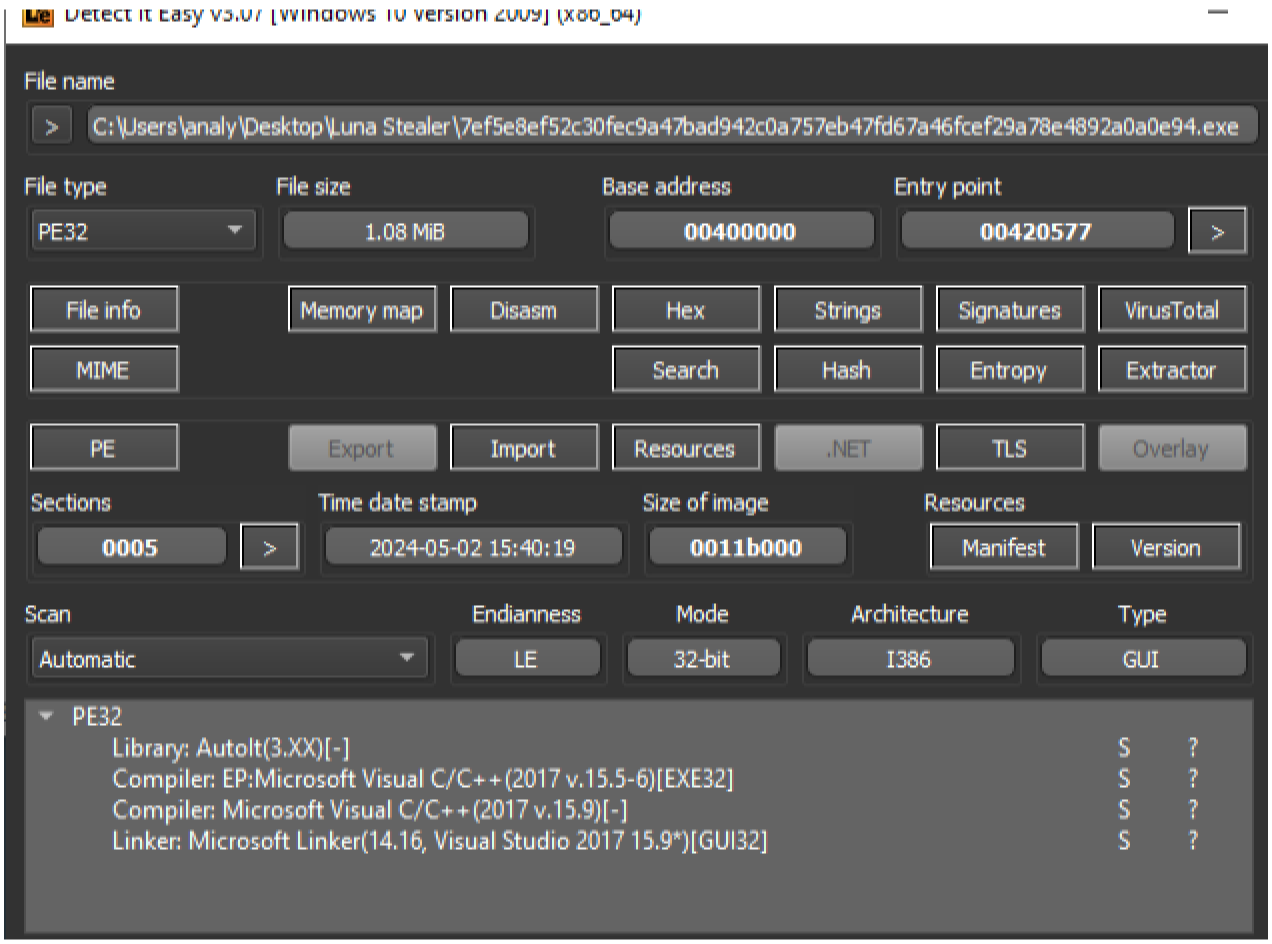
file > sha256	7EF5E8EF52C30FEC9A47BAD942C0A757EB47FD67A46FCEF29A78E4892A0A0E94
dos-stub > sha256	6356B11BC60A7A818B7465E357B34FFCA9E3B543135108E5BB46301FE88172CC
dos-header > sha256	37520C0EF323977685FE0349D71E9D794E80A62F6F7209C1C7ADB43BA839ECE3
rich-header > sha256	E8D3E8ADFE661BF6FB43E6F6EED8667E1E3BE5EAE1E5B280F7F18115CA2CB327
section > .text > sha256	1DF328D893FD19C2119C9A872FBC33E83B929B7119BEE88D15BD9FAE9D4246DD
section > .rdata > sha256	351B803807DFB852077C389B6B96198B5639A53F83045D190ABDF265DAB2C7A8
section > .data > sha256	3E7AC07BC2E03413763B49457AA252B016CC40394CEA187DA97BBD072C031F08
section > .rsrc > sha256	73F8CECCE6C4C6B485AFEFDE3F277ABEF730991A88186D8CA2C5F8575CBD4C0C
section > .reloc > sha256	92760FB78D9D6D312889C53B386DD9F87FA6CFE12841575D12972D831DEBB089
resource > AutoIt > sha256	3427245161E48745C64095F434FA44A1D992C87A8D3C0B570C72C66EE35D7071
version > sha256	7DEF7B83D5B1720629EF6C78AC87F329CCE9F1BED1CEF0C99D1B1CC63E5CD7EB
manifest > sha256	1BD8139910A81485AADB0BB28586E233768486DE8C09F6A565AE457805702D39
debug > stream > sha256	FA52D83C4C75580B910E0B136737BFA8DD734FE10479F79128120F22733A087F

Starting off, best thing to look at is the PE file information, using "Detect It Easy" and "Exeinfo PE" I was able to gain a good understanding of what exactly was contained within the executable.


Exeinfo PE - ver.0.0.7.3 by A.S.L - 1134+139 sign 2022.07.30

File : 7ef5e8ef52c30fec9a47bad942c0a757eb47fd67a46fcef29a78e48
Entry Point : 00020577 oo < EP Section : .text
File Offset : 0001F977 First Bytes : E8 6E 05 00 00
Linker Info : 14.16 SubSystem : Windows GUI
File Size : 00115800h < NET Overlay : NO 00000000
Image is 32bit executable RES/OVL : 22 / 0 % 2024
Autoit3 [v3.3.1 - 3.3.x] Jonathan Bennett & AutoIt Team (2020-22) - ww
Lamer Info - Help Hint - Unpack info 0 ms.
Big sec. 1 [.text] , try : Exe2Aut decomp. v0.10 2014 http://domoticx.com

We can see that this file is obviously embedded with a Autoit3 file, now I am not to familiar with writing Autoit code but from what I understand it is a BASIC-like scripting language or automating windows tasks. To ensure that Exeinfo is giving the correct information I also ran it through DIE(Detect It Easy).


Detect It Easy v3.07 [Windows 10 Version 2009] (x86_64)

File name
C:\Users\analy\Desktop\Luna Stealer\7ef5e8ef52c30fec9a47bad942c0a757eb47fd67a46fcef29a78e4892a0a0e94.exe

File type: PE32
File size: 1.08 MiB
Base address: 00400000
Entry point: 00420577

Sections: 0005
Time date stamp: 2024-05-02 15:40:19
Size of image: 0011b000

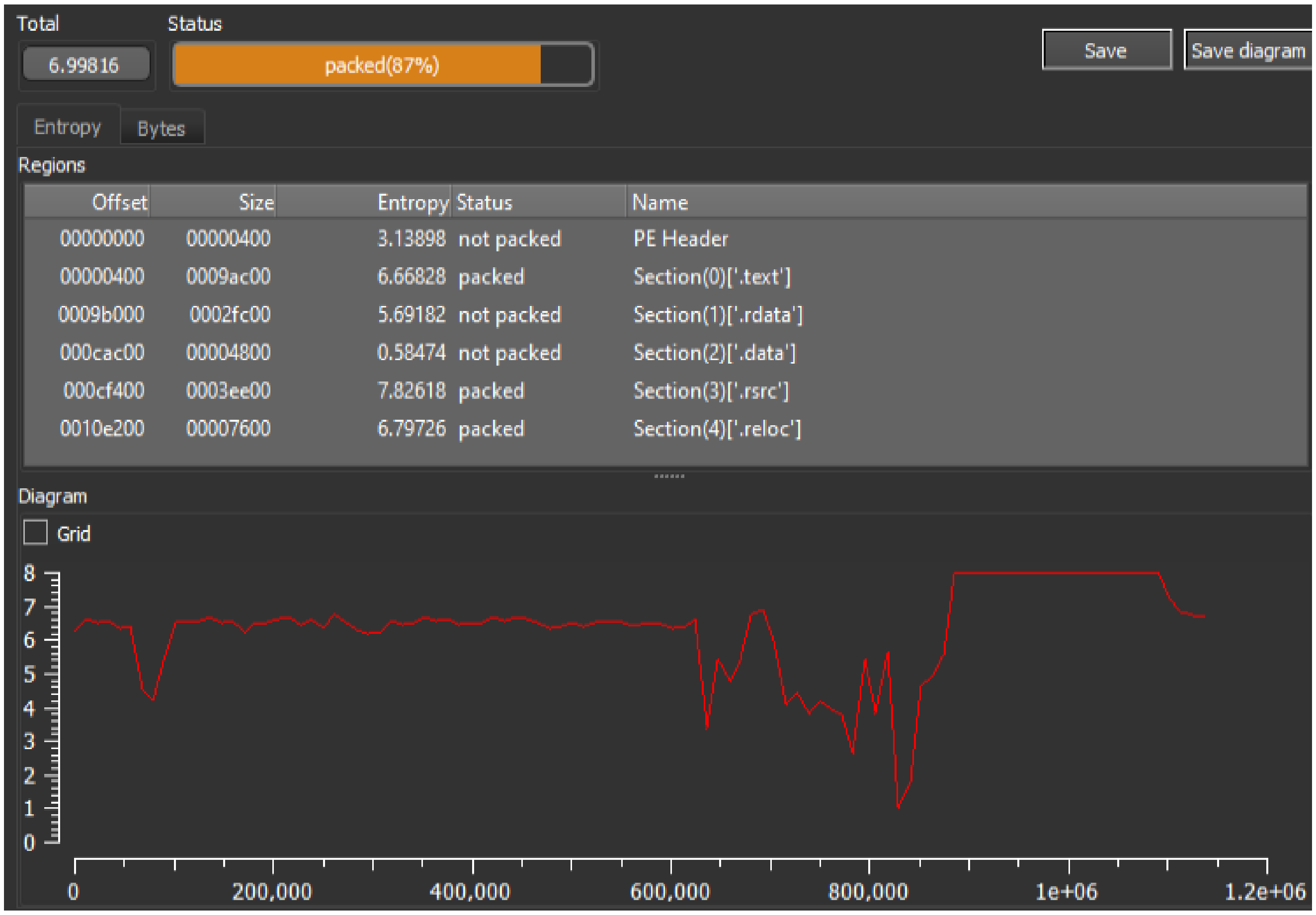
Scan: Automatic
Endianness: LE
Mode: 32-bit
Architecture: I386
Type: GUI

```
PE32
    Library: AutoIt(3.XX)[-]                                       S   ?
    Compiler: EP:Microsoft Visual C/C++(2017 v.15.5-6)[EXE32]      S   ?
    Compiler: Microsoft Visual C/C++(2017 v.15.9)[-]               S   ?
    Linker: Microsoft Linker(14.16, Visual Studio 2017 15.9*)[GUI32]  S   ?
```

With another confirmation that it contains a Autoit script along with using the C++ compiler and Microsoft linker, we can assume that this executable is a loader of some sorts that prepares memory and loads the Autoit script into memory in order to be executed. We also gain some value information such as the compiler time stamp, though this cannot be completely trusted we can assume that it was compiled recently. Also please ignore the “Luna Stealer” file name, it was a previous analysis and my file organization is terrible.

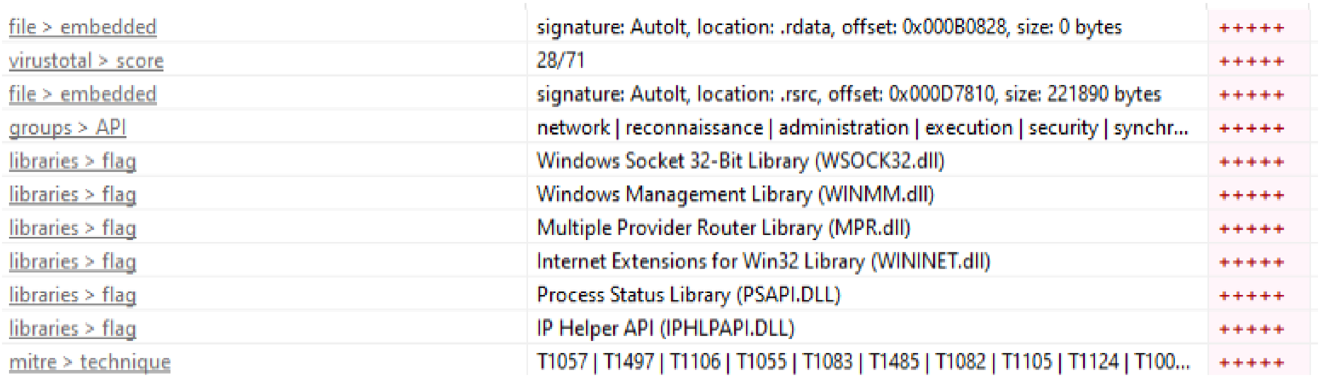
Time Date Stamp: 2024-05-02

After looking at the entropy of the file we can also see that mostly the .rsrc and .reloc of the PE sections are mainly packed, going off this and previous malware analysis we can assume that the Autoit script is contained within the .rsrc section and .reloc section of the PE file.



Offset	Size	Entropy	Status	Name
00000000	00000400	3.13898	not packed	PE Header
00000400	0009ac00	6.66828	packed	Section(0)['.text']
0009b000	0002fc00	5.69182	not packed	Section(1)['.rdata']
000cac00	00004800	0.58474	not packed	Section(2)['.data']
000cf400	0003ee00	7.82618	packed	Section(3)['.rsrc']
0010e200	00007600	6.79726	packed	Section(4)['.reloc']

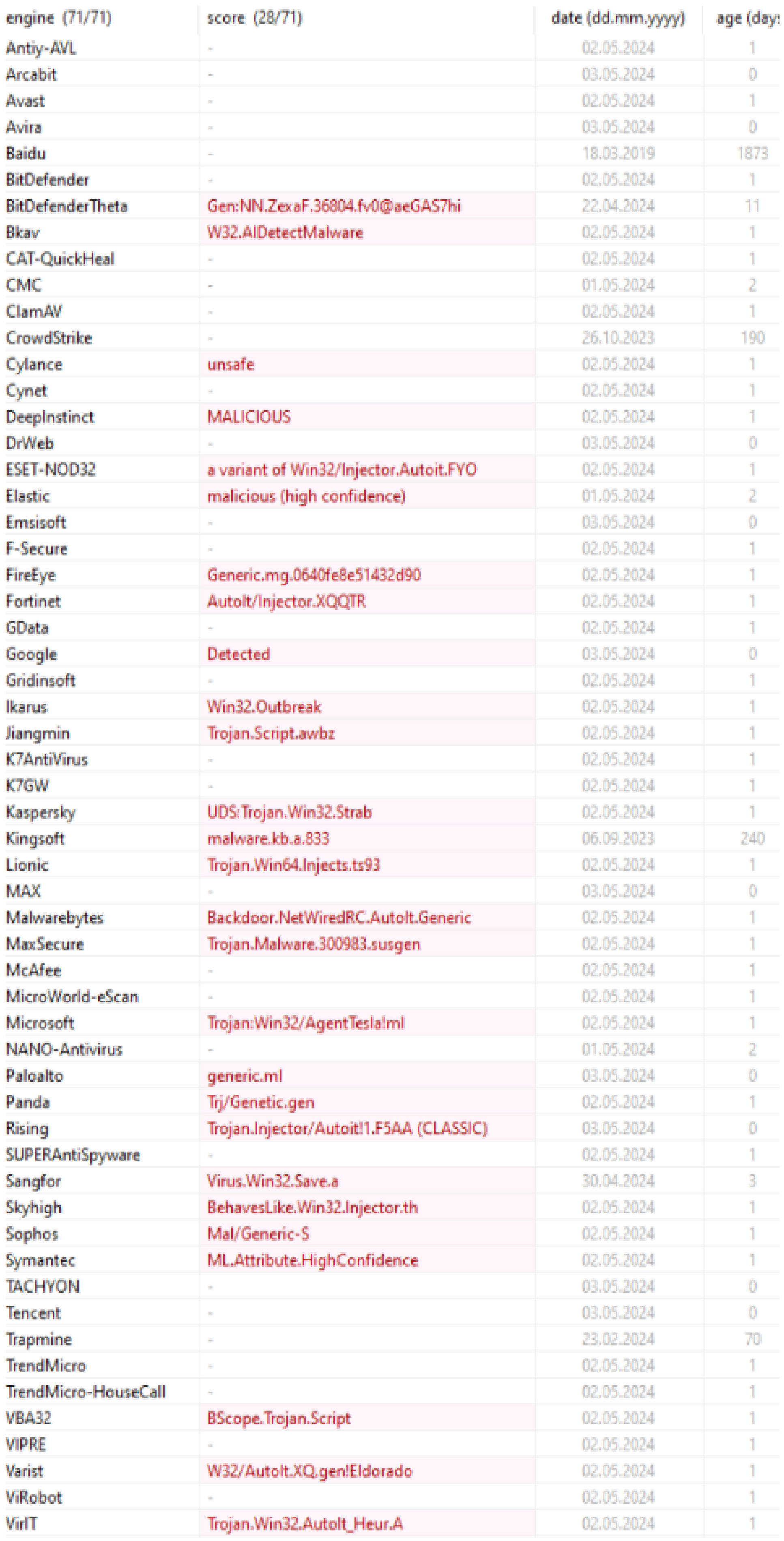
Using more one tool to get a better high level view of the PE file, I decided to throw it into PE Studio to gain a better understanding of if my assumptions were correct.



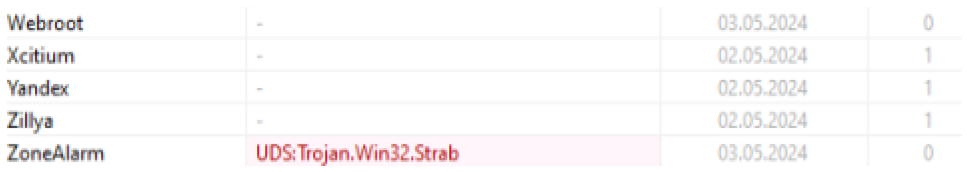
file > embedded	signature: AutoIt, location: .rdata, offset: 0x000B0828, size: 0 bytes	+++++
virustotal > score	28/71	+++++
file > embedded	signature: AutoIt, location: .rsrc, offset: 0x000D7810, size: 221890 bytes	+++++
groups > API	network | reconnaissance | administration | execution | security | synchr...	+++++
libraries > flag	Windows Socket 32-Bit Library (WSOCK32.dll)	+++++
libraries > flag	Windows Management Library (WINMM.dll)	+++++
libraries > flag	Multiple Provider Router Library (MPR.dll)	+++++
libraries > flag	Internet Extensions for Win32 Library (WININET.dll)	+++++
libraries > flag	Process Status Library (PSAPI.DLL)	+++++
libraries > flag	IP Helper API (IPHLPAPI.DLL)	+++++
mitre > technique	T1057 | T1497 | T1106 | T1055 | T1083 | T1485 | T1082 | T1105 | T1124 | T100...	+++++

As you can see so far everything seems to be correct, the Autoit file is embedded within the resources section of the executable.

Running a quick VirusTotal scan also shows that it has been detected by 28/71 anti-viruses.



engine (71/71)	score (28/71)	date (dd.mm.yyyy)	age (day
Antiy-AVL	-	02.05.2024	1
Arcabit	-	03.05.2024	0
Avast	-	02.05.2024	1
Avira	-	03.05.2024	0
Baidu	-	18.03.2019	1873
BitDefender	-	02.05.2024	1
BitDefenderTheta	Gen:NN.ZexaF.36804.fv0@aeGAS7hi	22.04.2024	11
Bkav	W32.AIDetectMalware	02.05.2024	1
CAT-QuickHeal	-	02.05.2024	1
CMC	-	01.05.2024	2
ClamAV	-	02.05.2024	1
CrowdStrike	-	26.10.2023	190
Cylance	unsafe	02.05.2024	1
Cynet	-	02.05.2024	1
DeepInstinct	MALICIOUS	02.05.2024	1
DrWeb	-	03.05.2024	0
ESET-NOD32	a variant of Win32/Injector.Autoit.FYO	02.05.2024	1
Elastic	malicious (high confidence)	01.05.2024	2
Emsisoft	-	03.05.2024	0
F-Secure	-	02.05.2024	1
FireEye	Generic.mg.0640fe8e51432d90	02.05.2024	1
Fortinet	AutoIt/Injector.XQQTR	02.05.2024	1
GData	-	02.05.2024	1
Google	Detected	03.05.2024	0
Gridinsoft	-	02.05.2024	1
Ikarus	Win32.Outbreak	02.05.2024	1
Jiangmin	Trojan.Script.awbz	02.05.2024	1
K7AntiVirus	-	02.05.2024	1
K7GW	-	02.05.2024	1
Kaspersky	UDS:Trojan.Win32.Strab	02.05.2024	1
Kingsoft	malware.kb.a.833	06.09.2023	240
Lionic	Trojan.Win64.Injects.ts93	02.05.2024	1
MAX	-	03.05.2024	0
Malwarebytes	Backdoor.NetWiredRC.AutoIt.Generic	02.05.2024	1
MaxSecure	Trojan.Malware.300983.susgen	02.05.2024	1
McAfee	-	02.05.2024	1
MicroWorld-eScan	-	02.05.2024	1
Microsoft	Trojan:Win32/AgentTesla!ml	02.05.2024	1
NANO-Antivirus	-	01.05.2024	2
Paloalto	generic.ml	03.05.2024	0
Panda	Trj/Genetic.gen	02.05.2024	1
Rising	Trojan.Injector/Autoit!1.F5AA (CLASSIC)	03.05.2024	0
SUPERAntiSpyware	-	02.05.2024	1
Sangfor	Virus.Win32.Save.a	30.04.2024	3
Skyhigh	BehavesLike.Win32.Injector.th	02.05.2024	1
Sophos	Mal/Generic-S	02.05.2024	1
Symantec	ML.Attribute.HighConfidence	02.05.2024	1
TACHYON	-	03.05.2024	0
Tencent	-	03.05.2024	0
Trapmine	-	23.02.2024	70
TrendMicro	-	02.05.2024	1
TrendMicro-HouseCall	-	02.05.2024	1
VBA32	BScope.Trojan.Script	02.05.2024	1
VIPRE	-	02.05.2024	1
Varist	W32/AutoIt.XQ.gen!Eldorado	02.05.2024	1
ViRobot	-	02.05.2024	1
VirIT	Trojan.Win32.AutoIt_Heur.A	02.05.2024	1



Webroot	-	03.05.2024	0
Xcitium	-	02.05.2024	1
Yandex	-	02.05.2024	1
Zillya	-	02.05.2024	1
ZoneAlarm	UDS:Trojan.Win32.Strab	03.05.2024	0

Next step was to look at the functions imported within the executable(loader), luckily since the file wasn't completely packed I didn't need to spend much time trying to unpack and deobfuscate it.

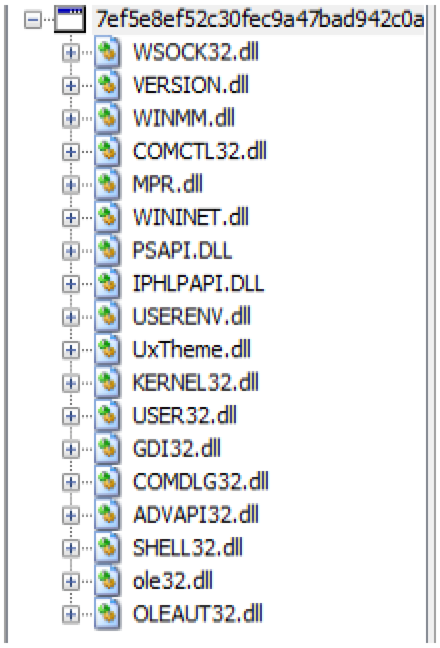
Some interesting function imports include: registry manipulation, access token manipulation(presumably for privilege escalation), process creation/discovery and manipulation, API execution, file and directory manipulation, anti debugger functions. Here is a list of some of them that seem to be important to the executable:

- DeleteFileW,x,0x000CA0BE,0x000CA0BE,214 (0x00D6),file,T1485 | Data Destruction,implicit,-,KERNEL32.dll
- InitializeSecurityDescriptor,-,0x000CB344,0x000CB344,375 (0x0177),security,T1134 | Access Token Manipulation,implicit,-,ADVAPI32.dll
- InitializeAcl,-,0x000CB364,0x000CB364,374 (0x0176),security,T1134 | Access Token Manipulation,implicit,-,ADVAPI32.dll
- AdjustTokenPrivileges,x,0x000CB374,0x000CB374,31 (0x001F),security,T1134 | Access Token Manipulation,implicit,-,ADVAPI32.dll
- OpenThreadToken,x,0x000CB38C,0x000CB38C,508 (0x01FC),security,T1134 | Access Token Manipulation,implicit,-,ADVAPI32.dll
- RegSetValueExW,x,0x000CB52A,0x000CB52A,638 (0x027E),registry,T1112 | Modify Registry,implicit,-,ADVAPI32.dll
- RegCreateKeyExW,x,0x000CB518,0x000CB518,569 (0x0239),registry,T1112 | Modify Registry,implicit,-,ADVAPI32.dll
- CreateProcessW,x,0x000CA5A4,0x000CA5A4,168 (0x00A8),execution,T1106 | Execution through API,implicit,-,KERNEL32.dll
- ShellExecuteW,x,0x000CB58C,0x000CB58C,290 (0x0122),execution,T1106 | Execution through API,implicit,-,SHELL32.dll
- LoadLibraryW,-,0x000CA1CE,0x000CA1CE,831 (0x033F),dynamic-library,T1106 | Execution through API,implicit,-,KERNEL32.dll
- MoveFileW,x,0x000CA0EA,0x000CA0EA,867 (0x0363),file,T1105 | Remote File Copy,implicit,-,KERNEL32.dll
- GetSystemDirectoryW,-,0x000CA51A,0x000CA51A,624 (0x0270),reconnaissance,T1083 | File and Directory Discovery,implicit,-,KERNEL32.dll
- GetWindowsDirectoryW,-,0x000CA55E,0x000CA55E,687 (0x02AF),reconnaissance,T1083 | File and Directory Discovery,implicit,-,KERNEL32.dll
- FindFirstFileW,x,0x000CA078,0x000CA078,313 (0x0139),file,T1083 | File and Directory Discovery,implicit,-,KERNEL32.dll

- IsDebuggerPresent,-,0x000C9D4E,0x000C9D4E,768 (0x0300),reconnaissance,T1082 | System Information Discovery,implicit,-,KERNEL32.dll
- VirtualAllocEx,x,0x000C9F66,0x000C9F66,1258 (0x04EA),memory,T1055 | Process Injection,implicit,-,KERNEL32.dll
- WriteProcessMemory,x,0x000C9F78,0x000C9F78,1326 (0x052E),memory,T1055 | Process Injection,implicit,-,KERNEL32.dll
- ReadProcessMemory,x,0x000C9F8E,0x000C9F8E,963 (0x03C3),memory,T1055 | Process Injection,implicit,-,KERNEL32.dll
- VirtualFree,-,0x000CA2A6,0x000CA2A6,1260 (0x04EC),memory,T1055 | Process Injection,implicit,-,KERNEL32.dll
- VirtualAlloc,x,0x000CA5DA,0x000CA5DA,1257 (0x04E9),memory,T1055 | Process Injection,implicit,-,KERNEL32.dll
- OpenProcess,x,0x000C9F58,0x000C9F58,896 (0x0380),execution,T1055 | Process Injection,implicit,-,KERNEL32.dll

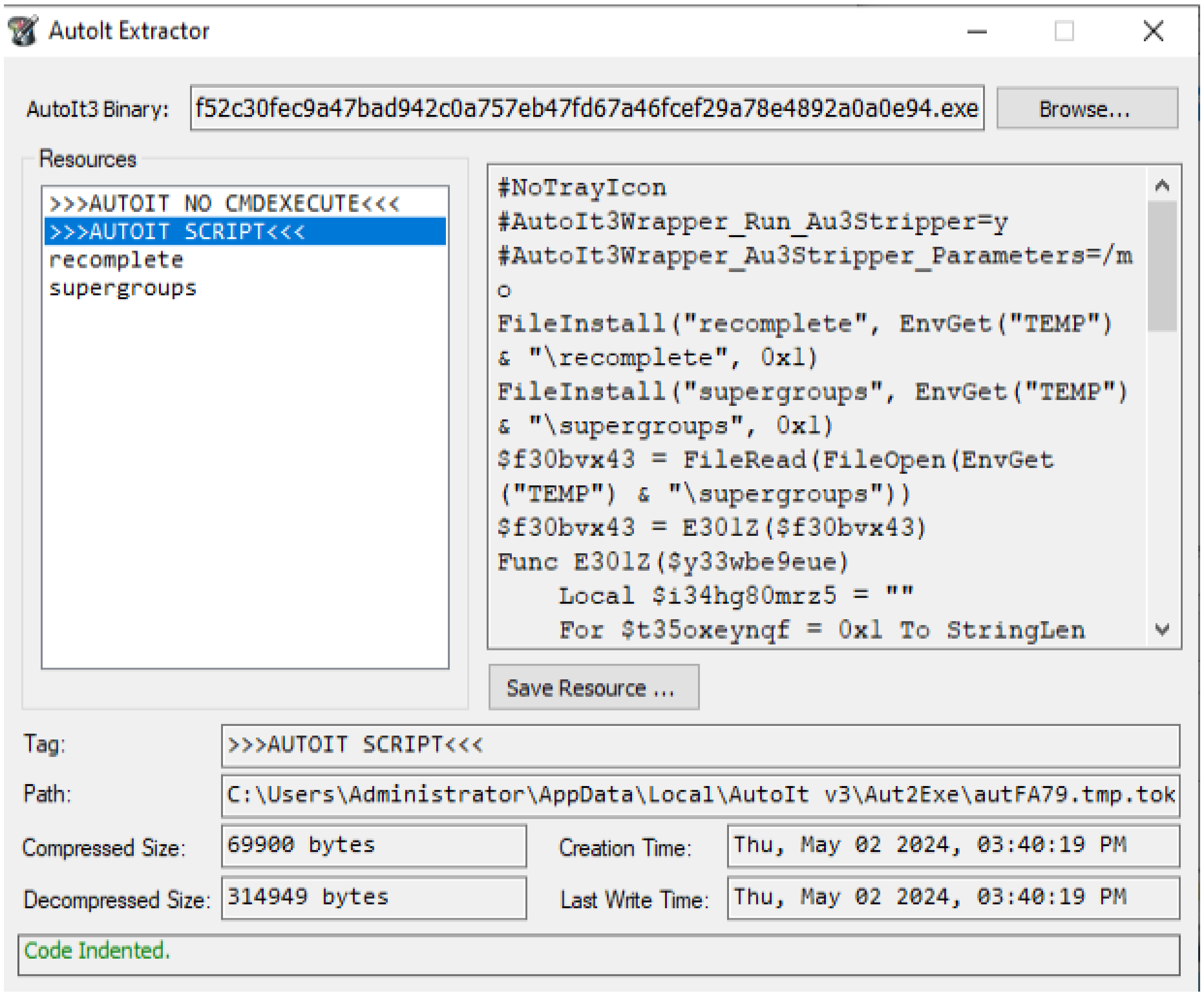
We can see that this is definitely some form of loader just basing it off the functions, we can also assume that this program prepare the embedded file for execution and does some form of process hollowing and injection before it executes. Given this, the executable we might be looking for will probably not contain the same name as the executable.

Here is a list of the DLL's the executable has statically linked, we can see that it uses WSOCK32 and WININET for some form of communication. As well as containing KERNAL32 and USER32 for process and file manipulation, along with using AVDVAPI32 for presumably some form of privilege escalation.



```
7ef5e8ef52c30fec9a47bad942c0a
    WSOCK32.dll
    VERSION.dll
    WINMM.dll
    COMCTL32.dll
    MPR.dll
    WININET.dll
    PSAPI.DLL
    IPHLPAPI.DLL
    USERENV.dll
    UxTheme.dll
    KERNEL32.dll
    USER32.dll
    GDI32.dll
    COMDLG32.dll
    ADVAPI32.dll
    SHELL32.dll
    ole32.dll
    OLEAUT32.dll
```

My next steps were to try to extract the Autoit script, now I've used tools such as Exe2Aut before but sadly that came up with: "Either it's not an Autoit-Executable or it's protected". Luckily instead of having to attempt to dig through memory for the Autoit script I came across a great tool called "Autoit-Extractor", I am in no way vouching for this tool as being safe so definitely make sure you run it within a contained environment. Anyways here the link and a picture of what it came up with:



AutoIt3 Binary: f52c30fec9a47bad942c0a757eb47fd67a46fcef29a78e4892a0a0e94.exe

```
>>>AUTOIT NO CMDEXECUTE<<<
>>>AUTOIT SCRIPT<<<
recomplete
supergroups
```

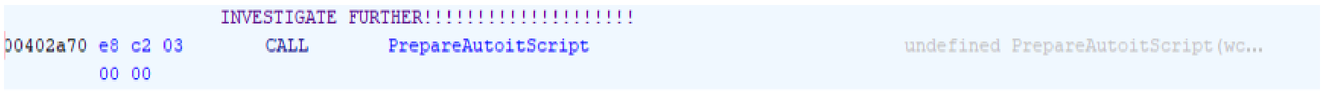
```
#NoTrayIcon
#AutoIt3Wrapper_Run_Au3Stripper=y
#AutoIt3Wrapper_Au3Stripper_Parameters=/m
o
FileInstall("recomplete", EnvGet("TEMP")
& "\recomplete", 0x1)
FileInstall("supergroups", EnvGet("TEMP")
& "\supergroups", 0x1)
$f30bvx43 = FileRead(FileOpen(EnvGet
("TEMP") & "\supergroups"))
$f30bvx43 = E301Z($f30bvx43)
Func E301Z($y33wbe9eue)
    Local $i34hg80mrz5 = ""
    For $t35oxeynqf = 0x1 To StringLen
```

Tag: >>>AUTOIT SCRIPT<<<
Path: C:\Users\Administrator\AppData\Local\AutoIt v3\Aut2Exe\autFA79.tmp.tok
Compressed Size: 69900 bytes Creation Time: Thu, May 02 2024, 03:40:19 PM
Decompressed Size: 314949 bytes Last Write Time: Thu, May 02 2024, 03:40:19 PM
Code Indented.

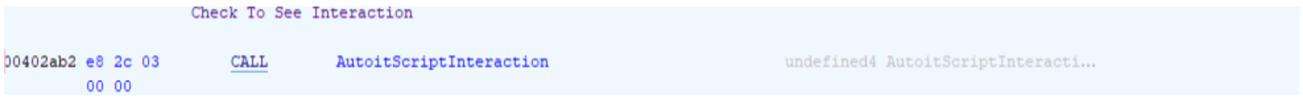
Link: https://github.com/digitalsleuth/autoit-extractor

The Autoit script was highly obfuscated and I was waaay to lazy to go through it and attempt piece together its complete functionality, instead I opted to just run the executable and see what it mainly does.

After using Ghidra to do some more advanced static analysis we can see that our presumptions about this loader is definitely correct as it includes functions to prepare the Autoit script and interact with it.



```
                   INVESTIGATE FURTHER!!!!!!!!!!!!!!!!!!!!
00402a70 e8 c2 03      CALL       PrepareAutoitScript                       undefined PrepareAutoitScript(wc...
         00 00
```

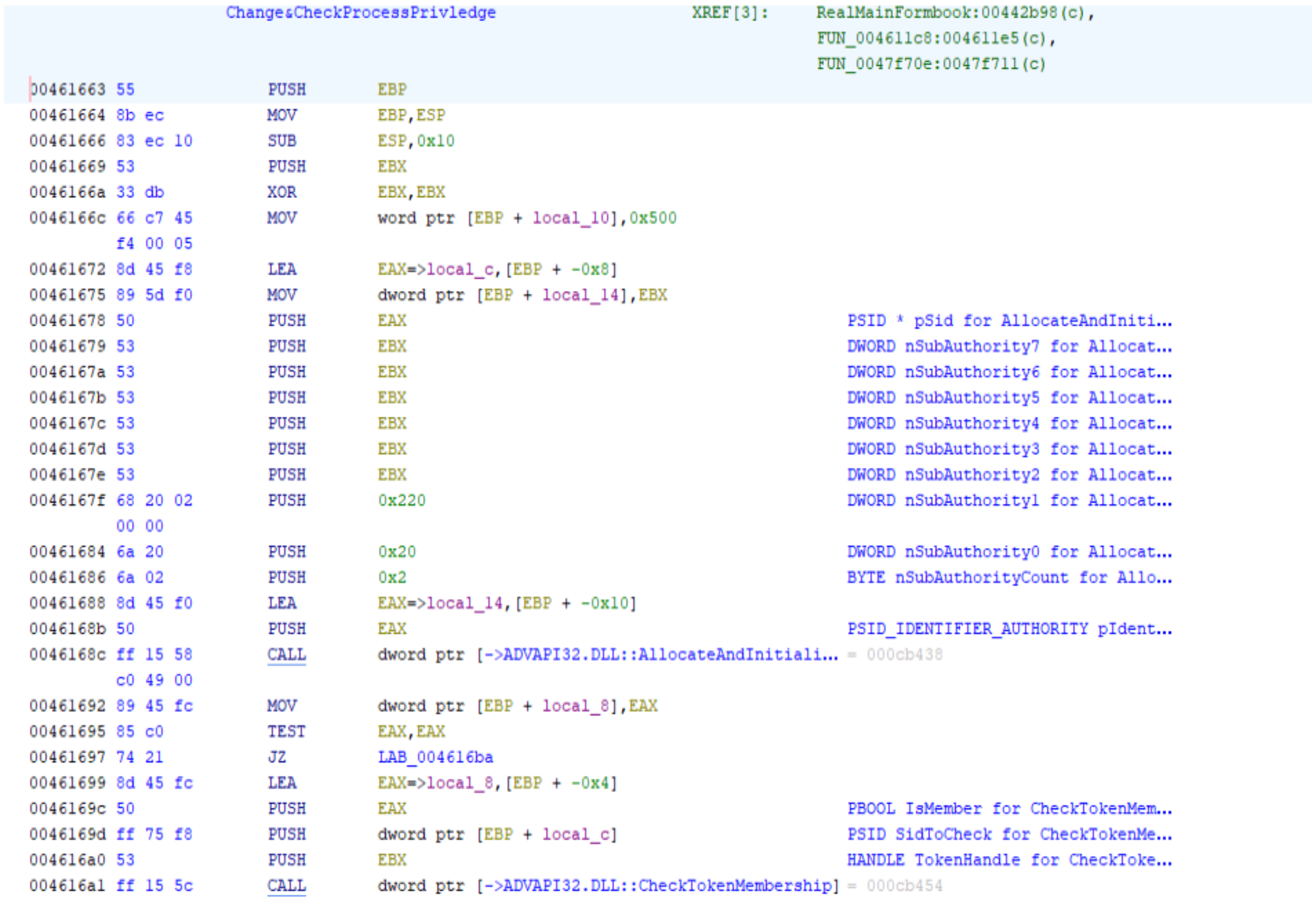


```
                    Check To See Interaction

00402ab2 e8 2c 03     CALL       AutoitScriptInteraction                    undefined4 AutoitScriptInteracti…
         00 00
```

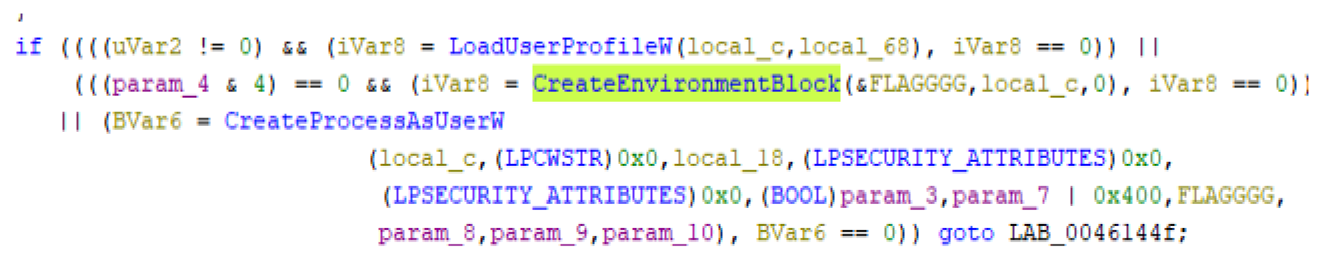
There also was privilege escalation contained within the binary, as you can see below. The function first checks to see if it has administrator rights, if not it'll proceed to attempt to escalate its privileges by creating a suspended process hallowing it out and then writing it's malicious code to that suspended state.

Privilege Check:



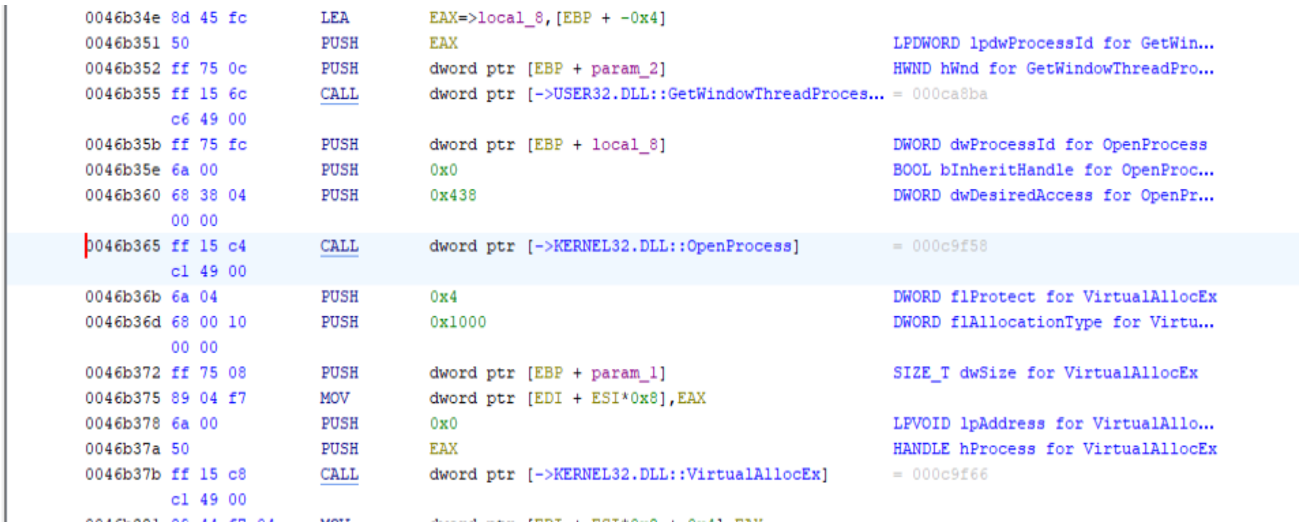
```
                       Change&CheckProcessPrivledge                 XREF[3]:     RealMainFormbook:00442b98(c),
                                                                                 FUN_004611c8:004611e5(c),
                                                                                 FUN_0047f70e:0047f711(c)
00461663 55            PUSH       EBP
00461664 8b ec         MOV        EBP,ESP
00461666 83 ec 10      SUB        ESP,0x10
00461669 53            PUSH       EBX
0046166a 33 db         XOR        EBX,EBX
0046166c 66 c7 45      MOV        word ptr [EBP + local_10],0x500
         f4 00 05
00461672 8d 45 f8      LEA        EAX=>local_c,[EBP + -0x8]
00461675 89 5d f0      MOV        dword ptr [EBP + local_14],EBX
00461678 50            PUSH       EAX                                      PSID * pSid for AllocateAndIniti…
00461679 53            PUSH       EBX                                      DWORD nSubAuthority7 for Allocat…
0046167a 53            PUSH       EBX                                      DWORD nSubAuthority6 for Allocat…
0046167b 53            PUSH       EBX                                      DWORD nSubAuthority5 for Allocat…
0046167c 53            PUSH       EBX                                      DWORD nSubAuthority4 for Allocat…
0046167d 53            PUSH       EBX                                      DWORD nSubAuthority3 for Allocat…
0046167e 53            PUSH       EBX                                      DWORD nSubAuthority2 for Allocat…
0046167f 68 20 02      PUSH       0x220                                    DWORD nSubAuthority1 for Allocat…
         00 00
00461684 6a 20         PUSH       0x20                                     DWORD nSubAuthority0 for Allocat…
00461686 6a 02         PUSH       0x2                                      BYTE nSubAuthorityCount for Allo…
00461688 8d 45 f0      LEA        EAX=>local_14,[EBP + -0x10]
0046168b 50            PUSH       EAX                                      PSID_IDENTIFIER_AUTHORITY pIdent…
0046168c ff 15 58      CALL       dword ptr [->ADVAPI32.DLL::AllocateAndInitiali… = 000cb438
         c0 49 00
00461692 89 45 fc      MOV        dword ptr [EBP + local_8],EAX
00461695 85 c0         TEST       EAX,EAX
00461697 74 21         JZ         LAB_004616ba
00461699 8d 45 fc      LEA        EAX=>local_8,[EBP + -0x4]
0046169c 50            PUSH       EAX                                      PBOOL IsMember for CheckTokenMem…
0046169d ff 75 f8      PUSH       dword ptr [EBP + local_c]                PSID SidToCheck for CheckTokenMe…
004616a0 53            PUSH       EBX                                      HANDLE TokenHandle for CheckToke…
004616a1 ff 15 5c      CALL       dword ptr [->ADVAPI32.DLL::CheckTokenMembership] = 000cb454
```

Process Creation:

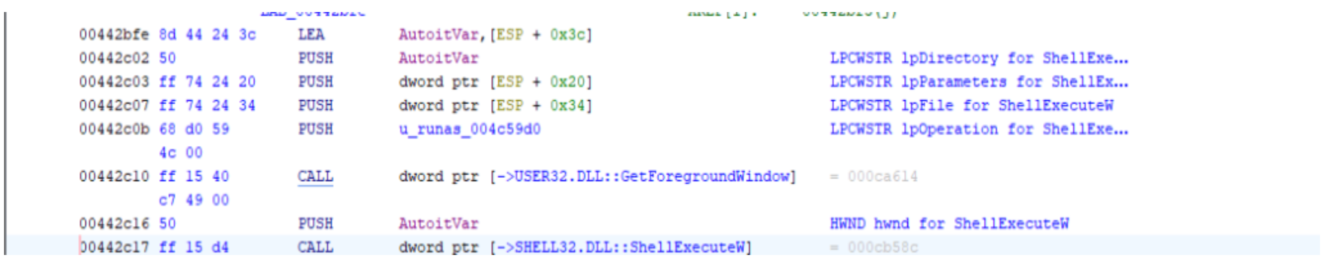


```
  if ((((uVar2 != 0) && (iVar8 = LoadUserProfileW(local_c,local_68), iVar8 == 0)) ||
      (((param_4 & 4) == 0 && (iVar8 = CreateEnvironmentBlock(&FLAGGGG,local_c,0), iVar8 == 0))
     || (BVar6 = CreateProcessAsUserW
                           (local_c,(LPCWSTR)0x0,local_18,(LPSECURITY_ATTRIBUTES)0x0,
                            (LPSECURITY_ATTRIBUTES)0x0,(BOOL)param_3,param_7 | 0x400,FLAGGGG,
                           param_8,param_9,param_10), BVar6 == 0)) goto LAB_0046144f;
```

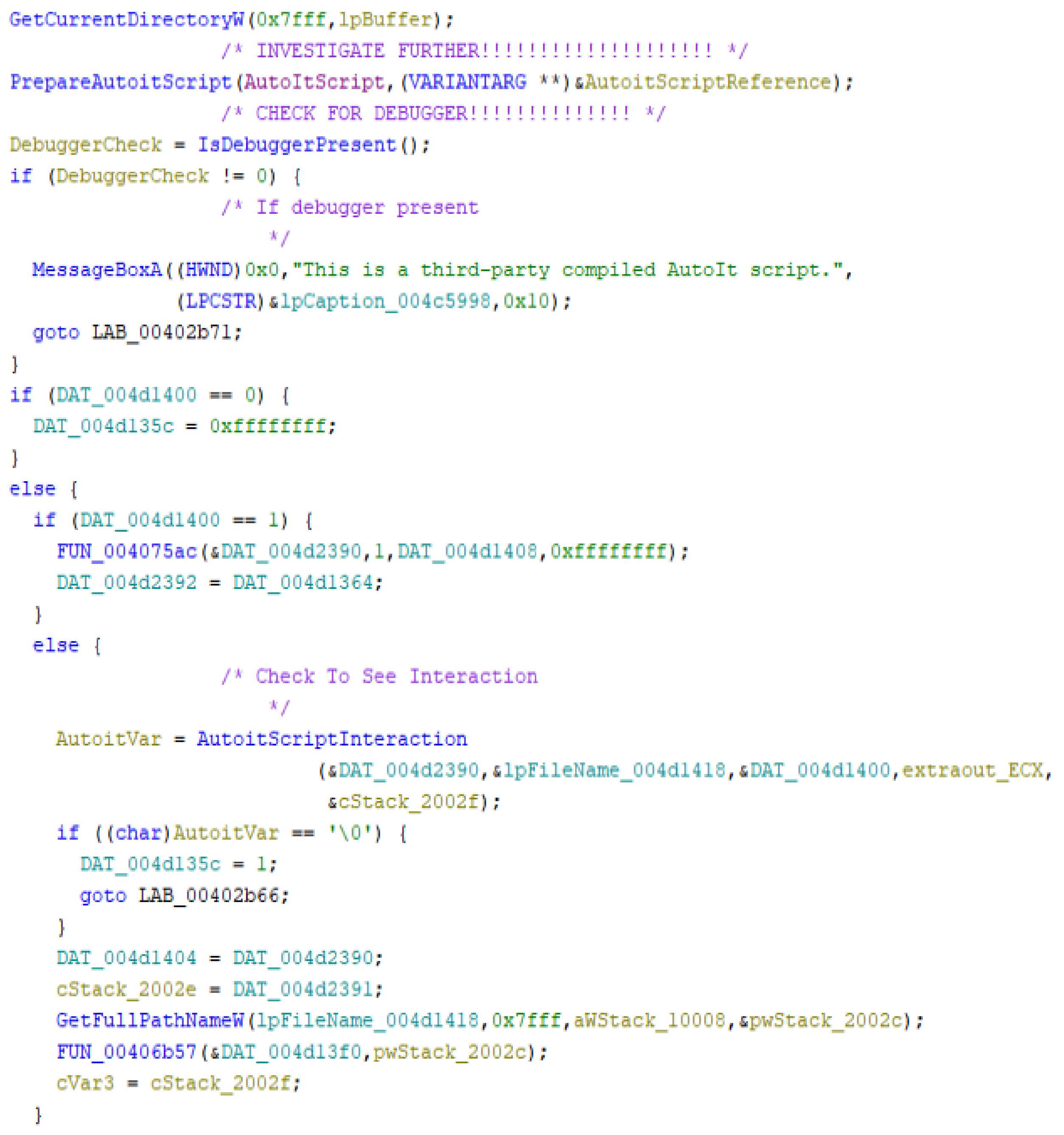
Process Injection:

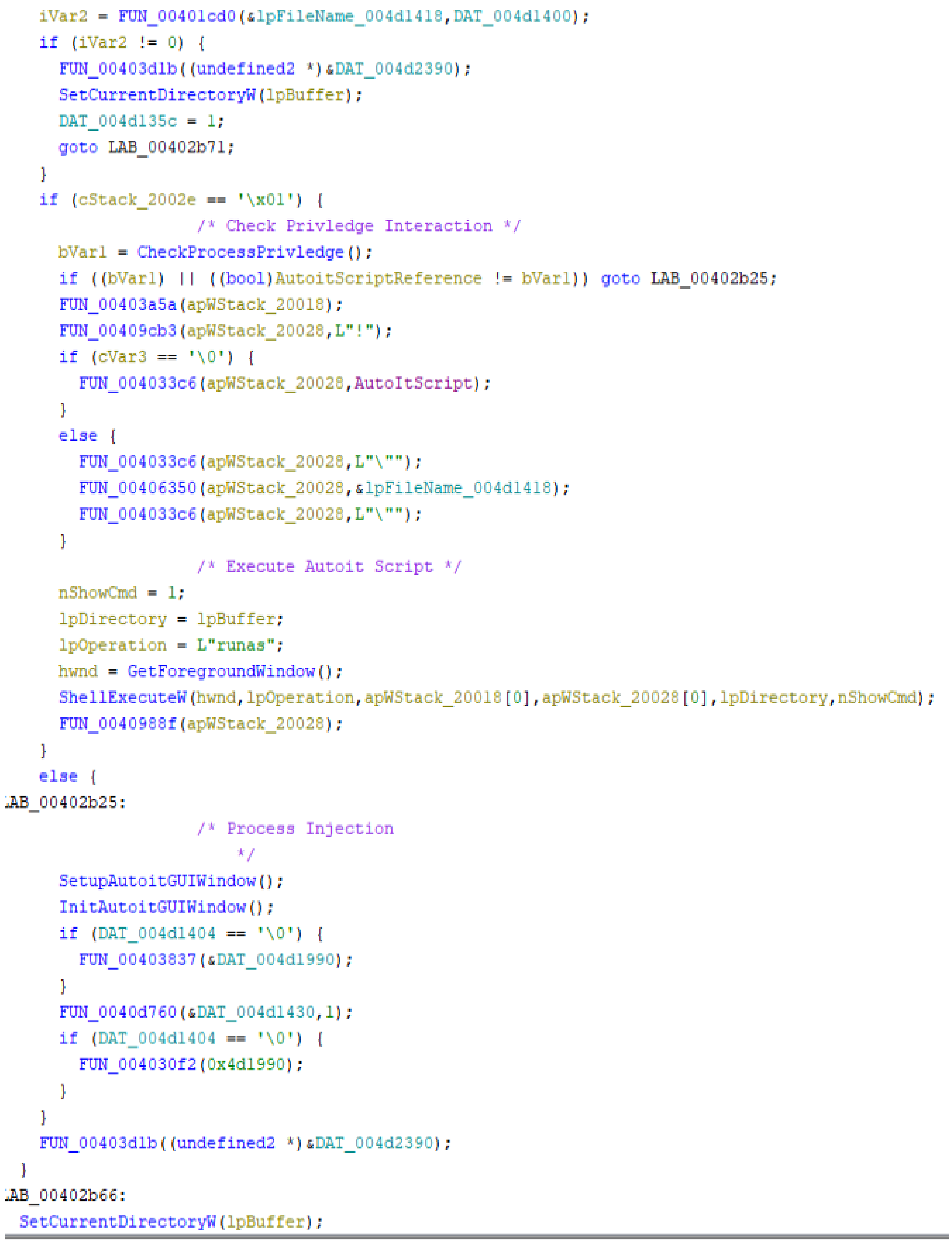

```
0046b34e 8d 45 fc      LEA     EAX=>local_8,[EBP + -0x4]
0046b351 50            PUSH    EAX                                             LPDWORD lpdwProcessId for GetWin...
0046b352 ff 75 0c      PUSH    dword ptr [EBP + param_2]                       HWND hWnd for GetWindowThreadPro...
0046b355 ff 15 6c      CALL    dword ptr [->USER32.DLL::GetWindowThreadProces... = 000ca8ba
         c6 49 00
0046b35b ff 75 fc      PUSH    dword ptr [EBP + local_8]                       DWORD dwProcessId for OpenProcess
0046b35e 6a 00         PUSH    0x0                                             BOOL bInheritHandle for OpenProc...
0046b360 68 38 04      PUSH    0x438                                           DWORD dwDesiredAccess for OpenPr...
         00 00
0046b365 ff 15 c4      CALL    dword ptr [->KERNEL32.DLL::OpenProcess]         = 000c9f58
         c1 49 00
0046b36b 6a 04         PUSH    0x4                                             DWORD flProtect for VirtualAllocEx
0046b36d 68 00 10      PUSH    0x1000                                          DWORD flAllocationType for Virtu...
         00 00
0046b372 ff 75 08      PUSH    dword ptr [EBP + param_1]                       SIZE_T dwSize for VirtualAllocEx
0046b375 89 04 f7      MOV     dword ptr [EDI + ESI*0x8],EAX
0046b378 6a 00         PUSH    0x0                                             LPVOID lpAddress for VirtualAllo...
0046b37a 50            PUSH    EAX                                             HANDLE hProcess for VirtualAllocEx
0046b37b ff 15 c8      CALL    dword ptr [->KERNEL32.DLL::VirtualAllocEx]      = 000c9f66
         c1 49 00
```

And finally the execution:


```
00442bfe 8d 44 24 3c   LEA     AutoitVar,[ESP + 0x3c]
00442c02 50            PUSH    AutoitVar                                       LPCWSTR lpDirectory for ShellExe...
00442c03 ff 74 24 20   PUSH    dword ptr [ESP + 0x20]                          LPCWSTR lpParameters for ShellEx...
00442c07 ff 74 24 34   PUSH    dword ptr [ESP + 0x34]                          LPCWSTR lpFile for ShellExecuteW
00442c0b 68 d0 59      PUSH    u_runas_004c59d0                                LPCWSTR lpOperation for ShellExe...
         4c 00
00442c10 ff 15 40      CALL    dword ptr [->USER32.DLL::GetForegroundWindow]   = 000ca614
         c7 49 00
00442c16 50            PUSH    AutoitVar                                       HWND hwnd for ShellExecuteW
00442c17 ff 15 d4      CALL    dword ptr [->SHELL32.DLL::ShellExecuteW]        = 000cb58c
```

Here is a breakdown of the main function within the malicious executable. We can clearly see that it checks for debuggers and privileges, along with preparing the Autoit script and injecting into a process in order to run the malicious code.


```
GetCurrentDirectoryW(0x7fff,lpBuffer);
                  /* INVESTIGATE FURTHER!!!!!!!!!!!!!!!!!!!! */
PrepareAutoitScript(AutoItScript,(VARIANTARG **)&AutoitScriptReference);
                  /* CHECK FOR DEBUGGER!!!!!!!!!!!!!! */
DebuggerCheck = IsDebuggerPresent();
if (DebuggerCheck != 0) {
                  /* If debugger present
                      */
  MessageBoxA((HWND)0x0,"This is a third-party compiled AutoIt script.",
              (LPCSTR)&lpCaption_004c5998,0x10);
  goto LAB_00402b71;
}
if (DAT_004d1400 == 0) {
  DAT_004d135c = 0xffffffff;
}
else {
  if (DAT_004d1400 == 1) {
    FUN_004075ac(&DAT_004d2390,1,DAT_004d1408,0xffffffff);
    DAT_004d2392 = DAT_004d1364;
  }
  else {
                  /* Check To See Interaction
                      */
    AutoitVar = AutoitScriptInteraction
                          (&DAT_004d2390,&lpFileName_004d1418,&DAT_004d1400,extraout_ECX,
                           &cStack_2002f);
    if ((char)AutoitVar == '\0') {
      DAT_004d135c = 1;
      goto LAB_00402b66;
    }
    DAT_004d1404 = DAT_004d2390;
    cStack_2002e = DAT_004d2391;
    GetFullPathNameW(lpFileName_004d1418,0x7fff,aWStack_10008,&pwStack_2002c);
    FUN_00406b57(&DAT_004d13f0,pwStack_2002c);
    cVar3 = cStack_2002f;
  }
```

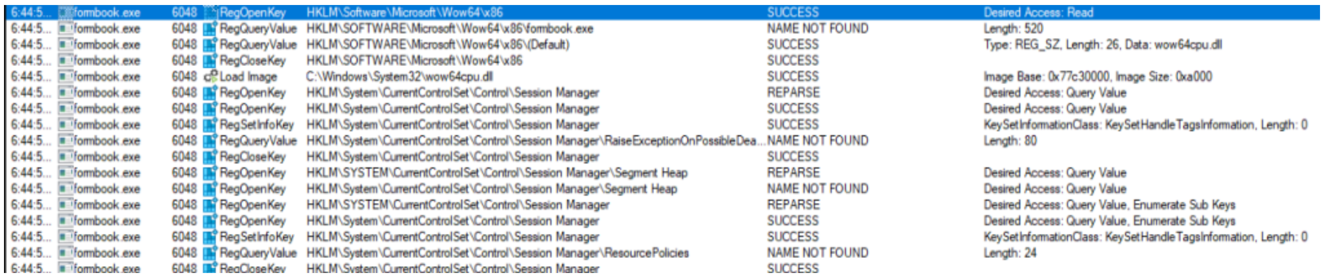


```
    iVar2 = FUN_00401cd0(&lpFileName_004d1418,DAT_004d1400);
    if (iVar2 != 0) {
      FUN_00403d1b((undefined2 *)&DAT_004d2390);
      SetCurrentDirectoryW(lpBuffer);
      DAT_004d135c = 1;
      goto LAB_00402b71;
    }
    if (cStack_2002e == '\x01') {
                    /* Check Privledge Interaction */
      bVar1 = CheckProcessPrivledge();
      if ((bVar1) || ((bool)AutoitScriptReference != bVar1)) goto LAB_00402b25;
      FUN_00403a5a(apWStack_20018);
      FUN_00409cb3(apWStack_20028,L"!");
      if (cVar3 == '\0') {
        FUN_004033c6(apWStack_20028,AutoItScript);
      }
      else {
        FUN_004033c6(apWStack_20028,L"\"");
        FUN_00406350(apWStack_20028,&lpFileName_004d1418);
        FUN_004033c6(apWStack_20028,L"\"");
      }
                    /* Execute Autoit Script */
      nShowCmd = 1;
      lpDirectory = lpBuffer;
      lpOperation = L"runas";
      hwnd = GetForegroundWindow();
      ShellExecuteW(hwnd,lpOperation,apWStack_20018[0],apWStack_20028[0],lpDirectory,nShowCmd);
      FUN_0040988f(apWStack_20028);
    }
    else {
AB_00402b25:
                    /* Process Injection
                        */
      SetupAutoitGUIWindow();
      InitAutoitGUIWindow();
      if (DAT_004d1404 == '\0') {
        FUN_00403837(&DAT_004d1990);
      }
      FUN_0040d760(&DAT_004d1430,1);
      if (DAT_004d1404 == '\0') {
        FUN_004030f2(0x4d1990);
      }
    }
    FUN_00403d1b((undefined2 *)&DAT_004d2390);
  }
AB_00402b66:
  SetCurrentDirectoryW(lpBuffer);
```

Now, since I am quite lazy and I find dynamic analysis to be a lot easier and more honestly more fun I decided to skip straight to the dynamic analysis portion instead of attempting to completely statically analyze the executable.

Dynamic analysis:

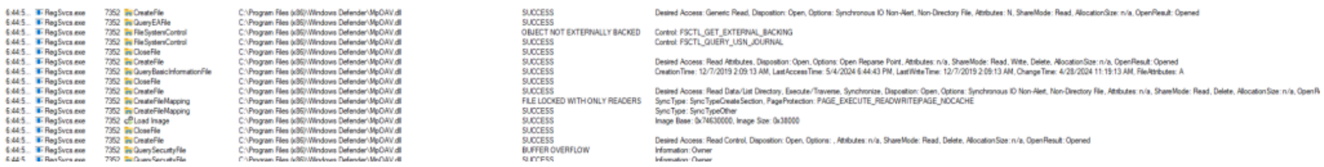
On first glace after running the executable I noticed within ProcessHacker it creates a process called “RegSvcs.exe” and injects its malicious code into there then terminates the parent process. The loader mainly seems to be checking for system information in order for the malicious code to work. It does things like check the internet settings using registry keys, queries system information and passes this to the real malicious process known as “RegSvcs.exe”. Agent Tesla seems to attempt to hide itself as a .NET Services installation tool called “Regsvcs.exe” I am assuming it does this in order to attempt to bypass detection and blend in with the other processes. Since it is an installation tool I am assuming that it does this in order to better query file information and make changes to the system without detection. Using Procmon we can analyze how both the executables function within runtime.



```
6:44:5...  formbook.exe  6048  RegOpenKey     HKLM\Software\Microsoft\Wow64\x86                                               SUCCESS          Desired Access: Read
6:44:5...  formbook.exe  6048  RegQueryValue  HKLM\SOFTWARE\Microsoft\Wow64\x86\formbook.exe                                  NAME NOT FOUND   Length: 520
6:44:5...  formbook.exe  6048  RegQueryValue  HKLM\SOFTWARE\Microsoft\Wow64\x86\(Default)                                     SUCCESS          Type: REG_SZ, Length: 26, Data: wow64cpu.dll
6:44:5...  formbook.exe  6048  RegCloseKey    HKLM\SOFTWARE\Microsoft\Wow64\x86                                               SUCCESS
6:44:5...  formbook.exe  6048  Load Image     C:\Windows\System32\wow64cpu.dll                                                SUCCESS          Image Base: 0x77c30000, Image Size: 0xa000
6:44:5...  formbook.exe  6048  RegOpenKey     HKLM\System\CurrentControlSet\Control\Session Manager                           REPARSE          Desired Access: Query Value
6:44:5...  formbook.exe  6048  RegOpenKey     HKLM\System\CurrentControlSet\Control\Session Manager                           SUCCESS          Desired Access: Query Value
6:44:5...  formbook.exe  6048  RegSetInfoKey  HKLM\System\CurrentControlSet\Control\Session Manager                           SUCCESS          KeySetInformationClass: KeySetHandleTagsInformation, Length: 0
6:44:5...  formbook.exe  6048  RegQueryValue  HKLM\System\CurrentControlSet\Control\Session Manager\RaiseExceptionOnPossibleDea...NAME NOT FOUND   Length: 80
6:44:5...  formbook.exe  6048  RegCloseKey    HKLM\System\CurrentControlSet\Control\Session Manager                           SUCCESS
6:44:5...  formbook.exe  6048  RegOpenKey     HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Segment Heap              REPARSE          Desired Access: Query Value
6:44:5...  formbook.exe  6048  RegOpenKey     HKLM\System\CurrentControlSet\Control\Session Manager\Segment Heap              NAME NOT FOUND   Desired Access: Query Value
6:44:5...  formbook.exe  6048  RegOpenKey     HKLM\SYSTEM\CurrentControlSet\Control\Session Manager                           REPARSE          Desired Access: Query Value, Enumerate Sub Keys
6:44:5...  formbook.exe  6048  RegOpenKey     HKLM\System\CurrentControlSet\Control\Session Manager                           SUCCESS          Desired Access: Query Value, Enumerate Sub Keys
6:44:5...  formbook.exe  6048  RegSetInfoKey  HKLM\System\CurrentControlSet\Control\Session Manager                           SUCCESS          KeySetInformationClass: KeySetHandleTagsInformation, Length: 0
6:44:5...  formbook.exe  6048  RegQueryValue  HKLM\System\CurrentControlSet\Control\Session Manager\ResourcePolicies          NAME NOT FOUND   Length: 24
6:44:5...  formbook.exe  6048  RegCloseKey    HKLM\System\CurrentControlSet\Control\Session Manager                           SUCCESS
```

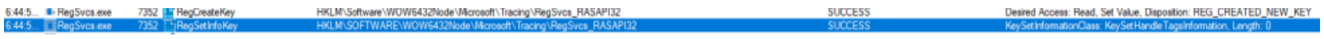
After the Agent Tesla executable does it thing collecting system information it is then terminated and spawns the Regsvcs process in order to actually perform the malicious code.

Regsvcs does thing such as attempting to disable Windows Defender by messing with the MpOAV.dll commonly used by that application.

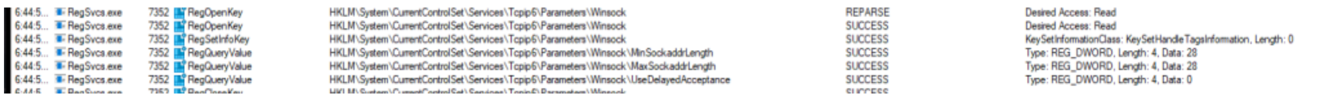


```
6:44:5...  RegSvcs.exe  7352  CreateFile                 C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: N, ShareMode: Read, AllocationSize: n/a, OpenResult: Opened
6:44:5...  RegSvcs.exe  7352  QueryEAFile                C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS
6:44:5...  RegSvcs.exe  7352  FileSystemControl          C:\Program Files (x86)\Windows Defender\MpOAV.dll  OBJECT NOT EXTERNALLY BACKED  Control: FSCTL_GET_EXTERNAL_BACKING
6:44:5...  RegSvcs.exe  7352  FileSystemControl          C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       Control: FSCTL_QUERY_USN_JOURNAL
6:44:5...  RegSvcs.exe  7352  CloseFile                  C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS
6:44:5...  RegSvcs.exe  7352  CreateFile                 C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a, OpenResult: Opened
6:44:5...  RegSvcs.exe  7352  QueryBasicInformationFile  C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       CreationTime: 12/7/2019 2:09:13 AM, LastAccessTime: 5/4/2024 6:44:43 PM, LastWriteTime: 12/7/2019 2:09:13 AM, ChangeTime: 4/28/2024 11:19:13 AM, FileAttributes: A
6:44:5...  RegSvcs.exe  7352  CloseFile                  C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS
6:44:5...  RegSvcs.exe  7352  CreateFile                 C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       Desired Access: Read Data/List Directory, Execute/Traverse, Synchronize, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: n/a, ShareMode: Read, Delete, AllocationSize: n/a, OpenR
6:44:5...  RegSvcs.exe  7352  CreateFileMapping          C:\Program Files (x86)\Windows Defender\MpOAV.dll  FILE LOCKED WITH ONLY READERS  SyncType: SyncTypeCreateSection, PageProtection: PAGE_EXECUTE_READWRITE|PAGE_NOCACHE
6:44:5...  RegSvcs.exe  7352  CreateFileMapping          C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       SyncType: SyncTypeOther
6:44:5...  RegSvcs.exe  7352  Load Image                 C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       Image Base: 0x74630000, Image Size: 0x38000
6:44:5...  RegSvcs.exe  7352  CloseFile                  C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS
6:44:5...  RegSvcs.exe  7352  CreateFile                 C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       Desired Access: Read Control, Disposition: Open, Options: , Attributes: n/a, ShareMode: Read, Delete, AllocationSize: n/a, OpenResult: Opened
6:44:5...  RegSvcs.exe  7352  QuerySecurityFile          C:\Program Files (x86)\Windows Defender\MpOAV.dll  BUFFER OVERFLOW               Information: Owner
6:44:5...  RegSvcs.exe  7352  QuerySecurityFile          C:\Program Files (x86)\Windows Defender\MpOAV.dll  SUCCESS                       Information: Owner
```

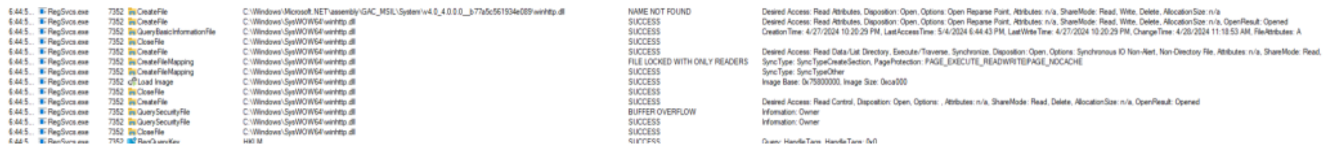
It also sets registry information for RASAPI32, I am assuming this is some sort of backdoor to the system as this is commonly used with remote connections to the system. Along with changing values for Winsock registry and using the winhttp.dll file we can definitely assume that this executable attempt to put a backdoor within the system. We can also notice that it looks for many VNC clients I am assuming this is either to steal network information of other computer within the infected network or it attempts to use these program in order to gain remote access.

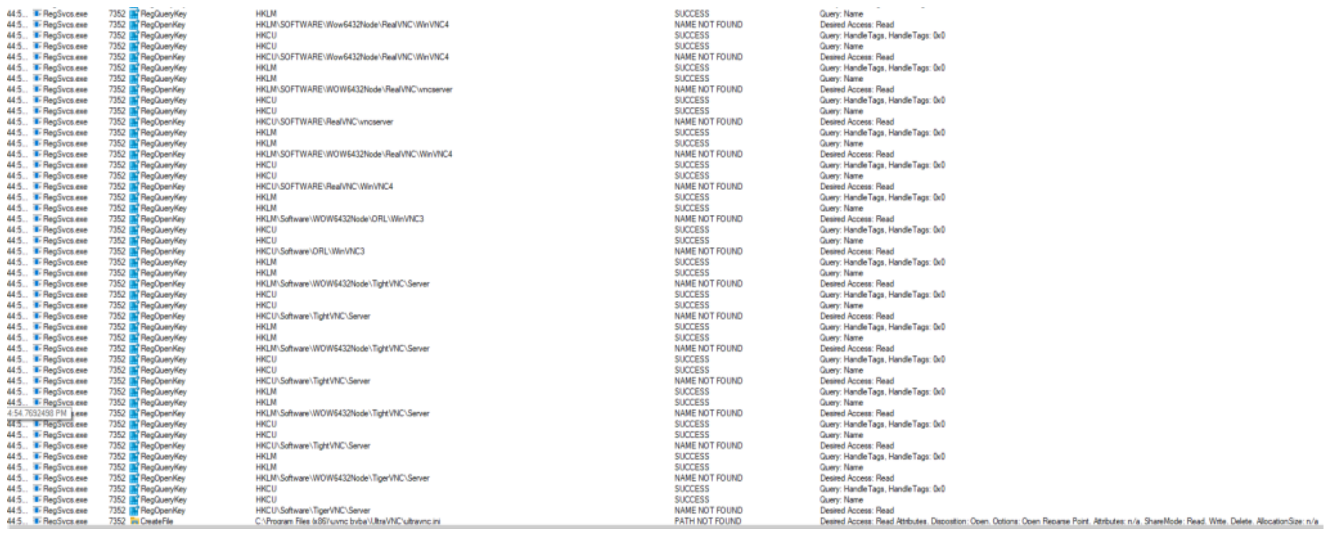


```
6:44:5...  RegSvcs.exe  7352  RegCreateKey   HKLM\Software\WOW6432Node\Microsoft\Tracing\RegSvcs_RASAPI32  SUCCESS  Desired Access: Read, Set Value, Disposition: REG_CREATED_NEW_KEY
6:44:5...  RegSvcs.exe  7352  RegSetInfoKey  HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\RegSvcs_RASAPI32  SUCCESS  KeySetInformationClass: KeySetHandleTagsInformation, Length: 0
```



```
6:44:5...  RegSvcs.exe  7352  RegOpenKey     HKLM\System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock                          REPARSE  Desired Access: Read
6:44:5...  RegSvcs.exe  7352  RegOpenKey     HKLM\System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock                          SUCCESS  Desired Access: Read
6:44:5...  RegSvcs.exe  7352  RegSetInfoKey  HKLM\System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock                          SUCCESS  KeySetInformationClass: KeySetHandleTagsInformation, Length: 0
6:44:5...  RegSvcs.exe  7352  RegQueryValue  HKLM\System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock\MinSockaddrLength        SUCCESS  Type: REG_DWORD, Length: 4, Data: 28
6:44:5...  RegSvcs.exe  7352  RegQueryValue  HKLM\System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock\MaxSockaddrLength        SUCCESS  Type: REG_DWORD, Length: 4, Data: 28
6:44:5...  RegSvcs.exe  7352  RegQueryValue  HKLM\System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock\UseDelayedAcceptance     SUCCESS  Type: REG_DWORD, Length: 4, Data: 0
6:44:5...  RegSvcs.exe  7352  RegCloseKey    HKLM\System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock                          SUCCESS
```

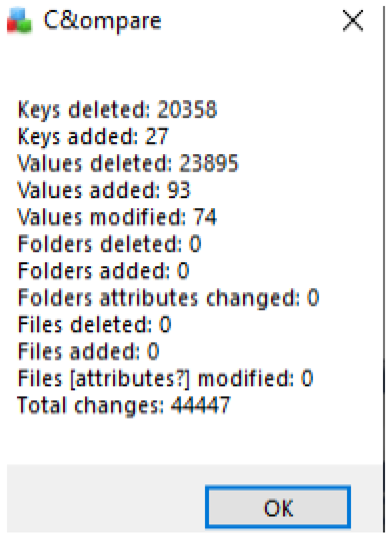

```
6:44:5... RegSvcs.exe  7352 CreateFile                C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System\v4.0_4.0.0.0__b77a5c561934e089\winhttp.dll  NAME NOT FOUND               Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a
6:44:5... RegSvcs.exe  7352 CreateFile                C:\Windows\SysWOW64\winhttp.dll  SUCCESS                      Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a, OpenResult: Opened
6:44:5... RegSvcs.exe  7352 QueryBasicInformationFile C:\Windows\SysWOW64\winhttp.dll  SUCCESS                      CreationTime: 4/27/2024 10:20:29 PM, LastAccessTime: 5/4/2024 6:44:43 PM, LastWriteTime: 4/27/2024 10:20:29 PM, ChangeTime: 4/28/2024 11:18:53 AM, FileAttributes: A
6:44:5... RegSvcs.exe  7352 CloseFile                 C:\Windows\SysWOW64\winhttp.dll  SUCCESS
6:44:5... RegSvcs.exe  7352 CreateFile                C:\Windows\SysWOW64\winhttp.dll  SUCCESS                      Desired Access: Read Data/List Directory, Execute/Traverse, Synchronize, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: n/a, ShareMode: Read,
6:44:5... RegSvcs.exe  7352 CreateFileMapping         C:\Windows\SysWOW64\winhttp.dll  FILE LOCKED WITH ONLY READERS SyncType: SyncTypeCreateSection, PageProtection: PAGE_EXECUTE_READWRITE|PAGE_NOCACHE
6:44:5... RegSvcs.exe  7352 CreateFileMapping         C:\Windows\SysWOW64\winhttp.dll  SUCCESS                      SyncType: SyncTypeOther
6:44:5... RegSvcs.exe  7352 Load Image                C:\Windows\SysWOW64\winhttp.dll  SUCCESS                      Image Base: 0x75800000, Image Size: 0xca000
6:44:5... RegSvcs.exe  7352 CloseFile                 C:\Windows\SysWOW64\winhttp.dll  SUCCESS
6:44:5... RegSvcs.exe  7352 CreateFile                C:\Windows\SysWOW64\winhttp.dll  SUCCESS                      Desired Access: Read Control, Disposition: Open, Options: , Attributes: n/a, ShareMode: Read, Delete, AllocationSize: n/a, OpenResult: Opened
6:44:5... RegSvcs.exe  7352 QuerySecurityFile         C:\Windows\SysWOW64\winhttp.dll  BUFFER OVERFLOW              Information: Owner
6:44:5... RegSvcs.exe  7352 QuerySecurityFile         C:\Windows\SysWOW64\winhttp.dll  SUCCESS                      Information: Owner
6:44:5... RegSvcs.exe  7352 CloseFile                 C:\Windows\SysWOW64\winhttp.dll  SUCCESS
6:44:5... RegSvcs.exe  7352 RegQueryKey               HKLM                             SUCCESS                      Query: HandleTags, HandleTags: 0x0
```


```
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKLM\SOFTWARE\Wow6432Node\RealVNC\WinVNC4         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\SOFTWARE\Wow6432Node\RealVNC\WinVNC4         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKLM\SOFTWARE\WOW6432Node\RealVNC\vncserver       NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\SOFTWARE\RealVNC\vncserver                   NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKLM\SOFTWARE\WOW6432Node\RealVNC\WinVNC4         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\SOFTWARE\RealVNC\WinVNC4                     NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKLM\Software\WOW6432Node\ORL\WinVNC3             NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\Software\ORL\WinVNC3                         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKLM\Software\WOW6432Node\TightVNC\Server         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\Software\TightVNC\Server                     NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKLM\Software\WOW6432Node\TightVNC\Server         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\Software\TightVNC\Server                     NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
4:54.7692498 PM .exe 7352 RegOpenKey   HKLM\Software\WOW6432Node\TightVNC\Server         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\Software\TightVNC\Server                     NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKLM                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKLM\Software\WOW6432Node\TigerVNC\Server         NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: HandleTags, HandleTags: 0x0
44:5... RegSvcs.exe  7352 RegQueryKey  HKCU                                              SUCCESS         Query: Name
44:5... RegSvcs.exe  7352 RegOpenKey   HKCU\Software\TigerVNC\Server                     NAME NOT FOUND  Desired Access: Read
44:5... RegSvcs.exe  7352 CreateFile   C:\Program Files (x86)\uvnc bvba\UltraVNC\ultravnc.ini  PATH NOT FOUND  Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a
```

Some other notable things to come out of procmon were the malicious executable use of cryptography features within the sytstem. It uses things such as bcrypt.dll in hash passwords within the file or prevent traffic analysis. along with querying for a bunch of the systems passwords. It checks browsers passwords, such as firefox, google chrome, edge, brave and etc and creates user data for them. It also checks for Microsoft credentials, Office, Outlook, FTP, and VNC profiles, and of course Discord.

Going based off this information so far, I can tell that Agent Tesla seems to be more of information stealer.

I also used Regshot in order to gain a better understanding of what registry keys it manipulated. We can see that the program modifies and deletes a ton of registry keys and adds 27 of its own.


```
C&ompare

Keys deleted: 20358
Keys added: 27
Values deleted: 23895
Values added: 93
Values modified: 74
Folders deleted: 0
Folders added: 0
Folders attributes changed: 0
Files deleted: 0
Files added: 0
Files [attributes?] modified: 0
Total changes: 44447
```

Using Regshot it deletes tons of driver configuration keys, most of them being within this category:

- HKLM\DRIVERS\DriverDatabase\DeviceIds\
- HKLM\DRIVERS\DriverDatabase\DeviceIds\
- HKLM\DRIVERS\DriverDatabase\DriverPackages\
- HKLM\DRIVERS\DriverDatabase\DriverFiles\
- HKLM\DRIVERS\DriverDatabase\DriverInfFiles\ Configurations\BthMini.NT\Services\BTHPORT\Parameters\
- HKLM\DRIVERS\DriverDatabase\DriverPackages\

It also adds 27 of it's own keys most likely related to remote connection, persistence, and some wireshark keys, which I am not to sure about that I am assuming this is why I received nothing in wireshark when I was trying to analyze it. Here are some of the most notable ones:

- HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\RegSvcs_RASAPI32
- HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\RegSvcs_RASMANC
- Creates these keys for remote acces
- HKU\.DEFAULT\Software\Microsoft\Windows\CurrentVersion\Explorer\BitBucket
- HKU\.DEFAULT\Software\Microsoft\Windows\CurrentVersion\Explorer\BitBucket\Volume
- HKU\.DEFAULT\Software\Microsoft\Windows\CurrentVersion\Explorer\BitBucket\Volume\
- Seems to manipulate the Recycling bin?
- HKU\S-1-5-21-769274696-41944572-4139179709-1001\SOFTWARE\Wireshark
- HKU\S-1-5-21-769274696-41944572-4139179709-1001\SOFTWARE\Wireshark\WinSparkle Settings
- Seems to mess with the wireshark settings

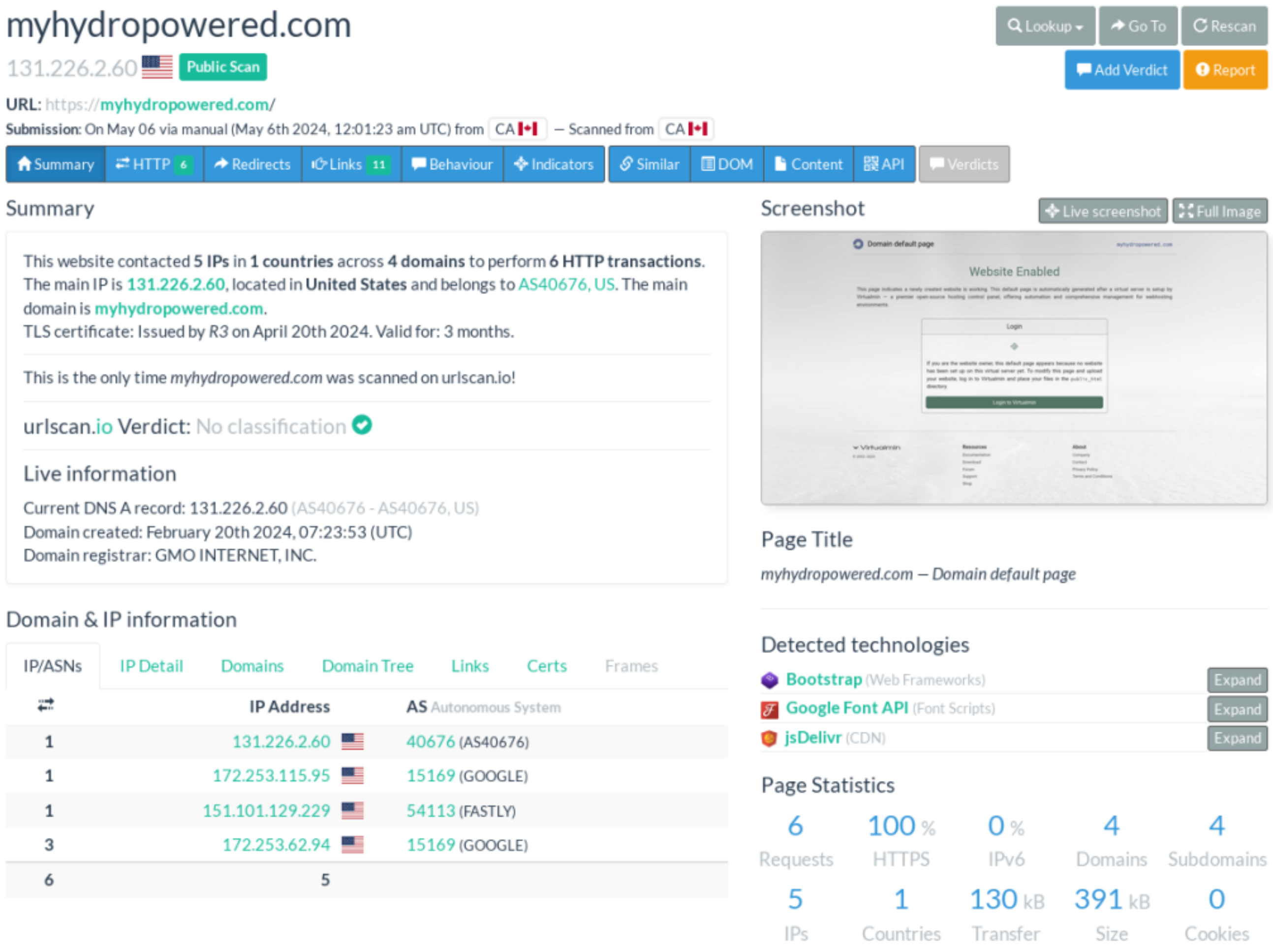
The last and final step was to search through the running proccesses strings. This was mostly filled with computer system and file system information that it gathered but 3 lines stood out more than anything.

0x32a2b9c (46): mail.myhydropowered.com

0x32a2bd8 (52): asksiri@myhydropowered.com

0x32a2c48 (60): superreport@myhydropowered.com

Now I could've gone further and attempted to reverse the running process but I am lazy it's the weekend and I am tired haha. So with that being that this is all an assumption but I am assuming it attempts to steal computer system and file information and then use SMTP or POP3 as it did contain those strings as well in order to send the passwords to the bad actors. After a quick google search of the website we can see that the domain is still up and running, it contains a virtualmin login screen which is primarily used for web hosting or in this case email accounts. I am assuming that after Agent Tesla executes it attempts to send the stolen information to these emails. Running a urlscan on this website provides a nice screen shot of the infultrators domain. We can see that this domain is hosted in the US, and the bad actors use this website in some way with the Agent Tesla malware.



myhydropowered.com

131.226.2.60 Public Scan

Lookup · Go To · Rescan · Add Verdict · Report

URL: https://myhydropowered.com/

Submission: On May 06 via manual (May 6th 2024, 12:01:23 am UTC) from CA — Scanned from CA

Summary · HTTP 6 · Redirects · Links 11 · Behaviour · Indicators · Similar · DOM · Content · API · Verdicts

Summary

This website contacted **5 IPs** in **1 countries** across **4 domains** to perform **6 HTTP transactions**. The main IP is 131.226.2.60, located in **United States** and belongs to AS40676, US. The main domain is myhydropowered.com.
TLS certificate: Issued by R3 on April 20th 2024. Valid for: 3 months.

This is the only time *myhydropowered.com* was scanned on urlscan.io!

urlscan.io Verdict: No classification

Live information

Current DNS A record: 131.226.2.60 (AS40676 - AS40676, US)
Domain created: February 20th 2024, 07:23:53 (UTC)
Domain registrar: GMO INTERNET, INC.

Domain & IP information

IP/ASNs · IP Detail · Domains · Domain Tree · Links · Certs · Frames

⇄	IP Address	AS Autonomous System
1	131.226.2.60	40676 (AS40676)
1	172.253.115.95	15169 (GOOGLE)
1	151.101.129.229	54113 (FASTLY)
3	172.253.62.94	15169 (GOOGLE)
6	5	

Screenshot

Live screenshot · Full Image

Domain default page — myhydropowered.com

Website Enabled

This page indicates a newly created website is working. This default page is automatically generated after a virtual server is setup by Virtualmin — a premier open-source hosting control panel, offering automation and comprehensive management for webhosting environments.

Login

If you are the website owner, this default page appears because no website has been set up on this virtual server yet. To modify this page and upload your website, log in to Virtualmin and place your files in the public_html directory.

Login to Virtualmin

Virtualmin · Resources: Documentation, Download, Forum, Support, Shop · About: Company, Contact, Privacy Policy, Terms and Conditions

Page Title

myhydropowered.com — Domain default page

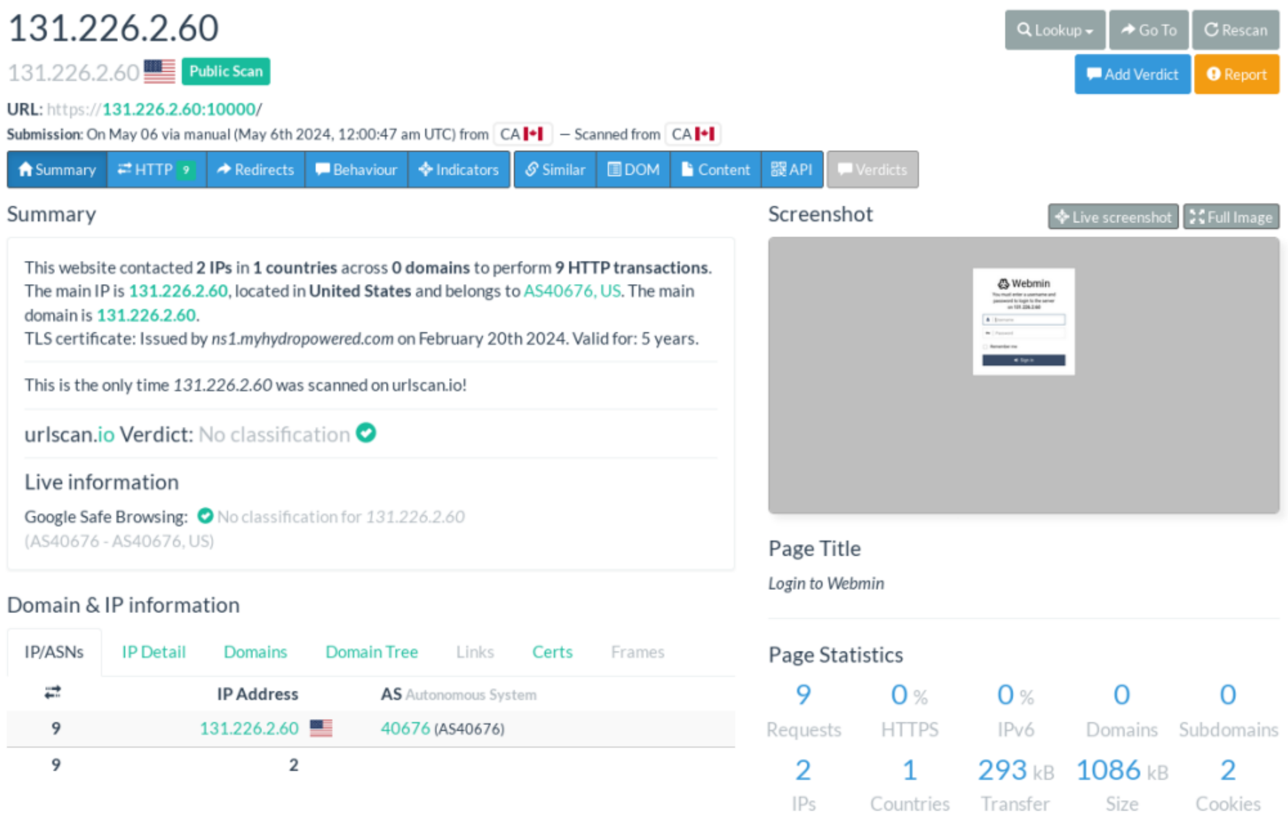
Detected technologies

Bootstrap (Web Frameworks)	Expand
Google Font API (Font Scripts)	Expand
jsDelivr (CDN)	Expand

Page Statistics

6	100 %	0 %	4	4
Requests	HTTPS	IPv6	Domains	Subdomains
5	1	130 kB	391 kB	0
IPs	Countries	Transfer	Size	Cookies

After running another urlscan on the login page we are redirected to a Webmin login portal. I am quite familiar with Webmin as I have used it many times in the past. I am assuming that their using this to host the emails that they send their stolen information to. We can see that this website was only created only a few months ago and based off the compiler time stamp we can tell that this virus is quite new to the ecosystem.



131.226.2.60

131.226.2.60 Public Scan

URL: https://131.226.2.60:10000/

Submission: On May 06 via manual (May 6th 2024, 12:00:47 am UTC) from CA — Scanned from CA

Summary

This website contacted **2 IPs** in **1 countries** across **0 domains** to perform **9 HTTP transactions**. The main IP is 131.226.2.60, located in **United States** and belongs to AS40676, US. The main domain is 131.226.2.60.
TLS certificate: Issued by *ns1.myhydropowered.com* on February 20th 2024. Valid for: 5 years.

This is the only time *131.226.2.60* was scanned on urlscan.io!

urlscan.io Verdict: No classification

Live information

Google Safe Browsing: No classification for *131.226.2.60*
(AS40676 - AS40676, US)

Screenshot

Page Title

Login to Webmin

Domain & IP information

IP/ASNs | IP Detail | Domains | Domain Tree | Links | Certs | Frames

⇄	IP Address	AS Autonomous System
9	131.226.2.60	40676 (AS40676)
9	2	

Page Statistics

9	0 %	0 %	0	0
Requests	HTTPS	IPv6	Domains	Subdomains
2	1	293 kB	1086 kB	2
IPs	Countries	Transfer	Size	Cookies

I went ahead and sent the domain registar an email of the suspected abuse so hopefully they are able to deal with it and take down the website, now I know this won't completely stop them but atleast if my assumptions are correct this will annoy them a little and that makes me a little bit happier knowning I hopefully ruined their day a little haha.

Anyways thank you to anyone who took the time to read this, if you have any questions or suggestions or spot any errors please let me know by sending me an email(which is located within the About section of the website). I am always looking for suggestions, corrections, and to learn so I am open to hearing any of your ideas. Also please know that I tried to use the words "assumption and assuming" as much as possble as that is all this is. Especially within the world of reverse engeering and malware analysis I can never be 100% certain. Hopefully some of my assumptions are correct, anyways thanks for reading.