

# Sheridan College

<b>Course</b>	<b>INFO43921</b> <b>Malicious Code Design and Defense</b>
Activity Title	<b>Assignment # 1: Basic Static Analysis Techniques</b>
Student Name	Jackson Yuan
Lab performed on (Date):	Jan 31, 2023 11:59 PM

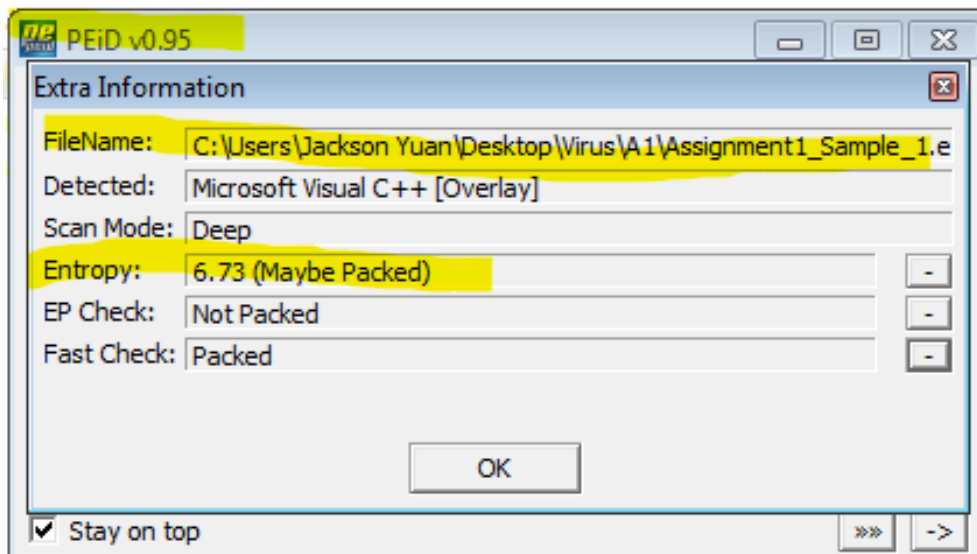
## Objectives

- Perform basic Static Analysis on the given malware samples.
- Use tools discussed during lectures such as but not limited to, PEiD, TriDNet, HashMyFiles, CFF Explorer, BinText, Notepad++, etc.
- Understand malware naming schemes and be able to figure out the malware type, platform it infects, malware family name and group name.
- Be able to document and cite using IEEE and APA
- Be able to detect if a malware has a code signing certificate

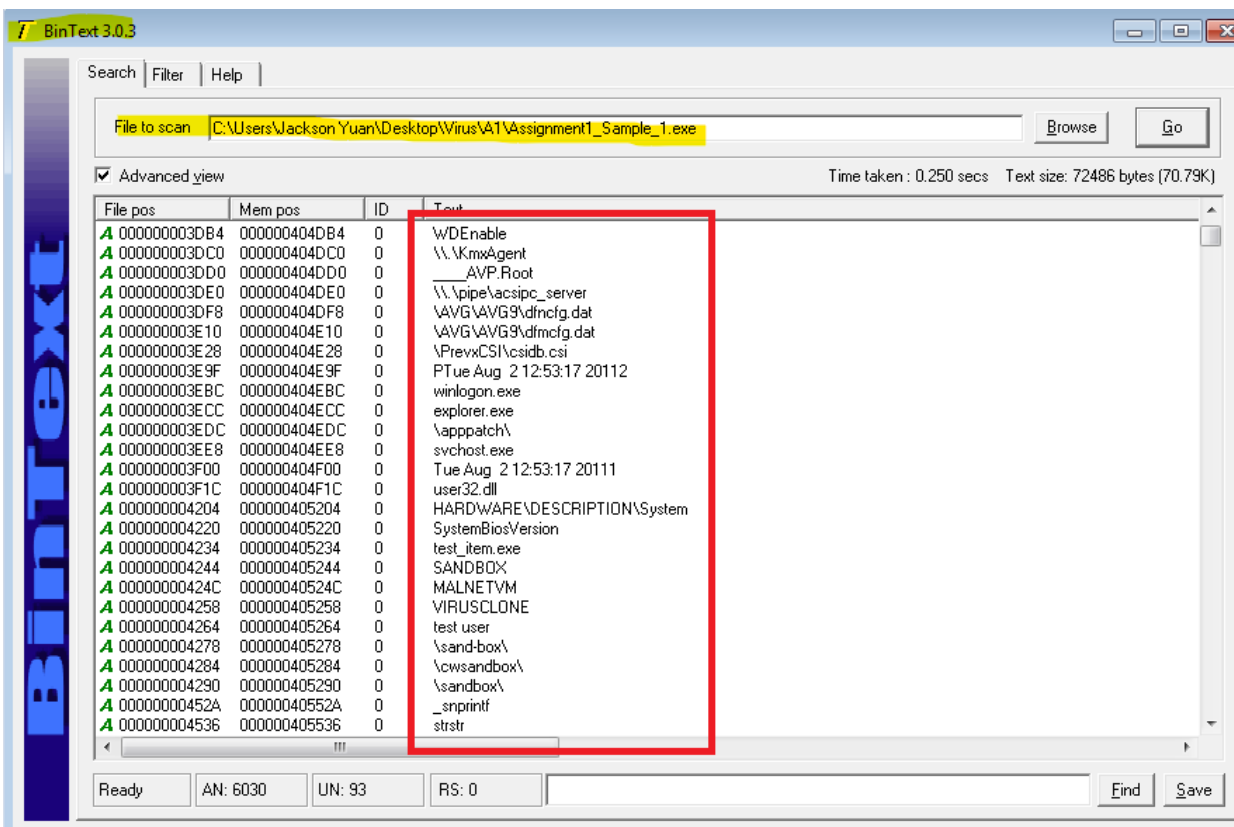
## Output Section for Sample #1

### Output#1: Packed or Unpacked Analysis

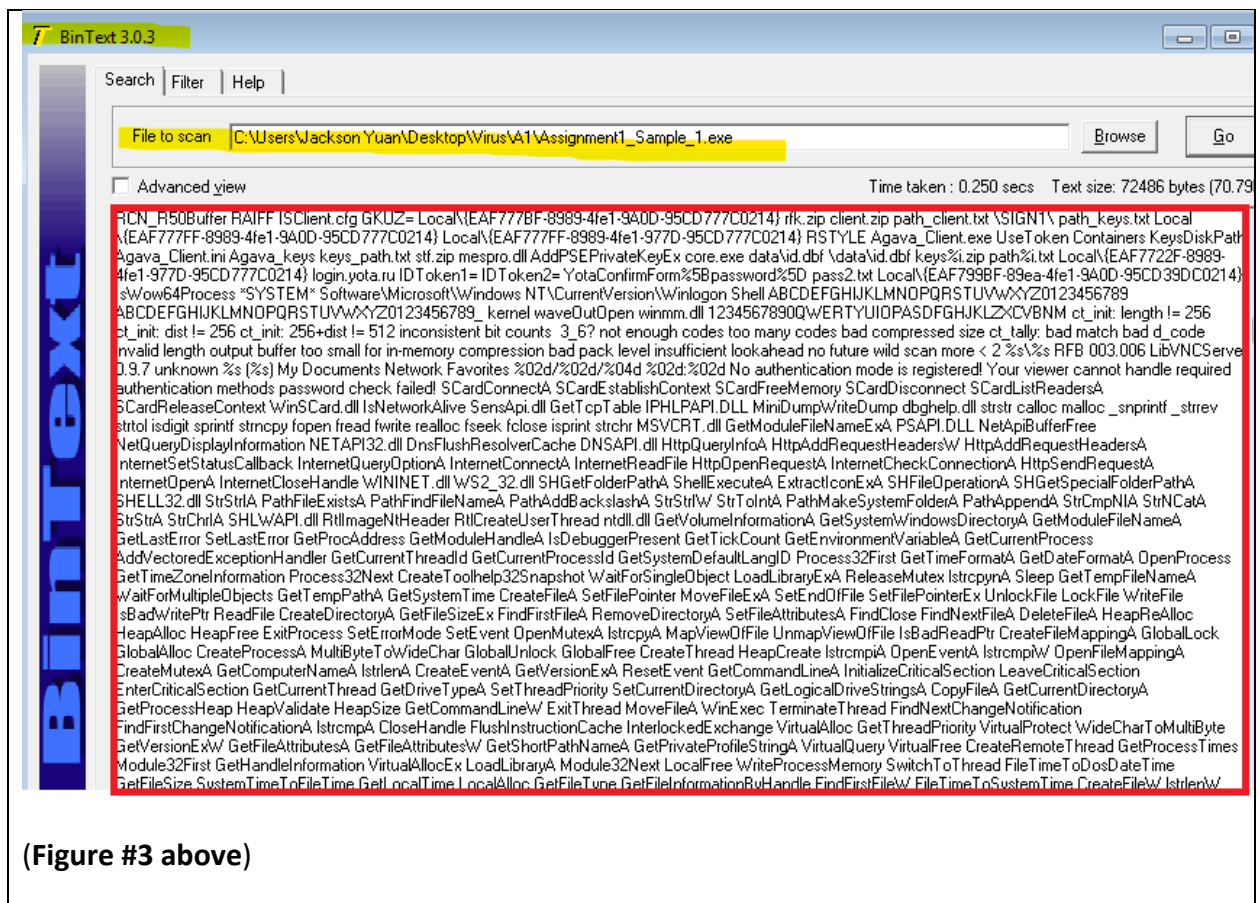
This sample is not packed. To confirm if this hypothesis is true or not, we used tools such as PEiD to check the entropy of the sample. In figure #1, we can see that PEiD gives us an entropy of "6.73" and beside it saying that it's "maybe packed". However, we can use another tool to double check such as BinText. We can do a simple string analysis on it as shown on Figure #2 and Figure #3. Notice that all the text are readable thus confirming that the sample is unpacked.



(Figure #1 above)



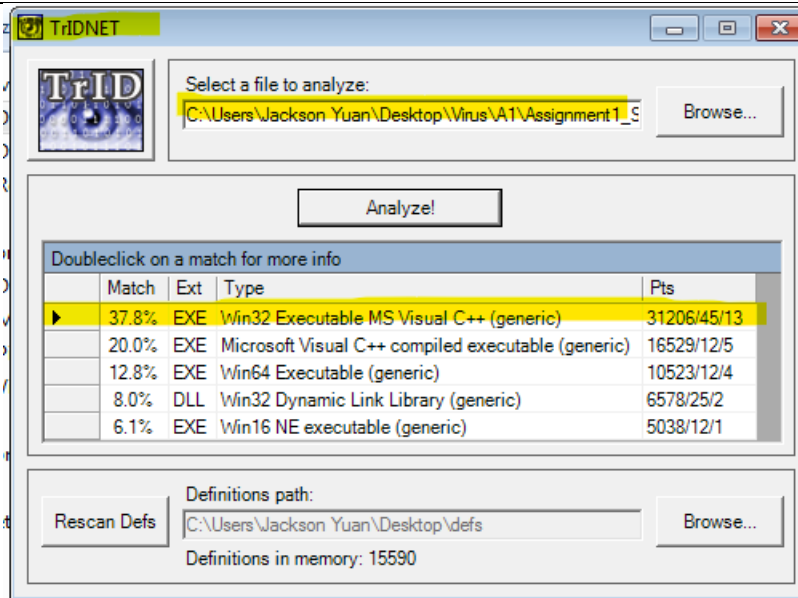
(Figure #2 above)



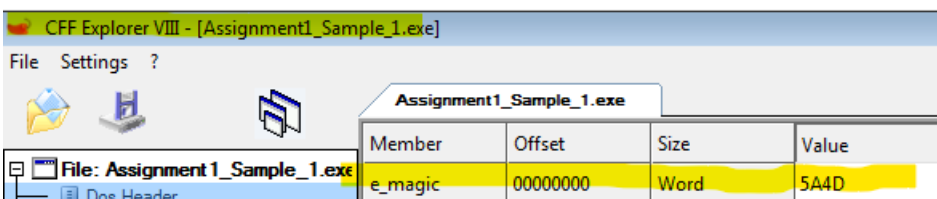
(Figure #3 above)

## Output#2: File Format Identification

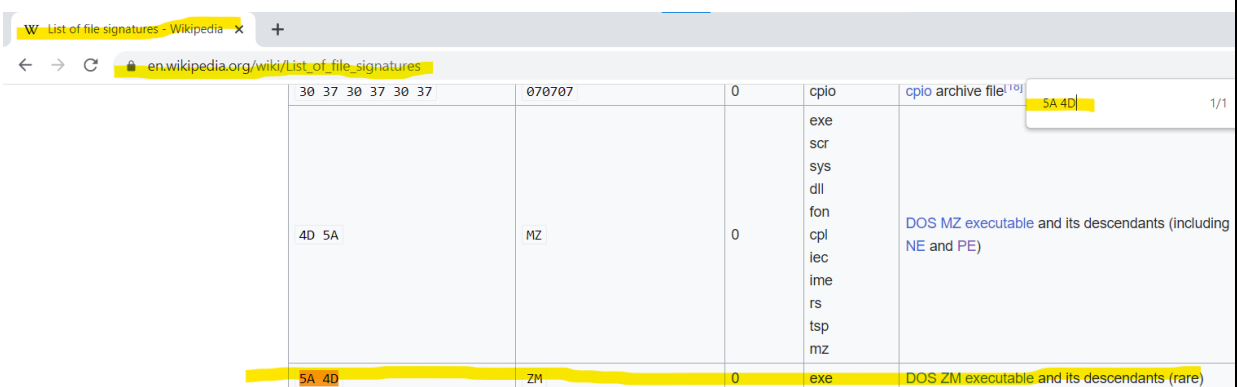
To identify the file format of this sample, we use the tool TRIDNet to check. According to Figure #4, TRIDNet confirms that this sample file is an “.exe” file format and of type Win32 Executable MS Visual C++. We can also check the magic byte of the file and as you can see using CFF exploer in figure #5, we get 5A4D in hex. We then cross reference this on Wikipedia on Figure #6 and we can confirm that it is indeed a “.exe” file format (*List of File Signatures*, 2023).



(Figure #4 above)



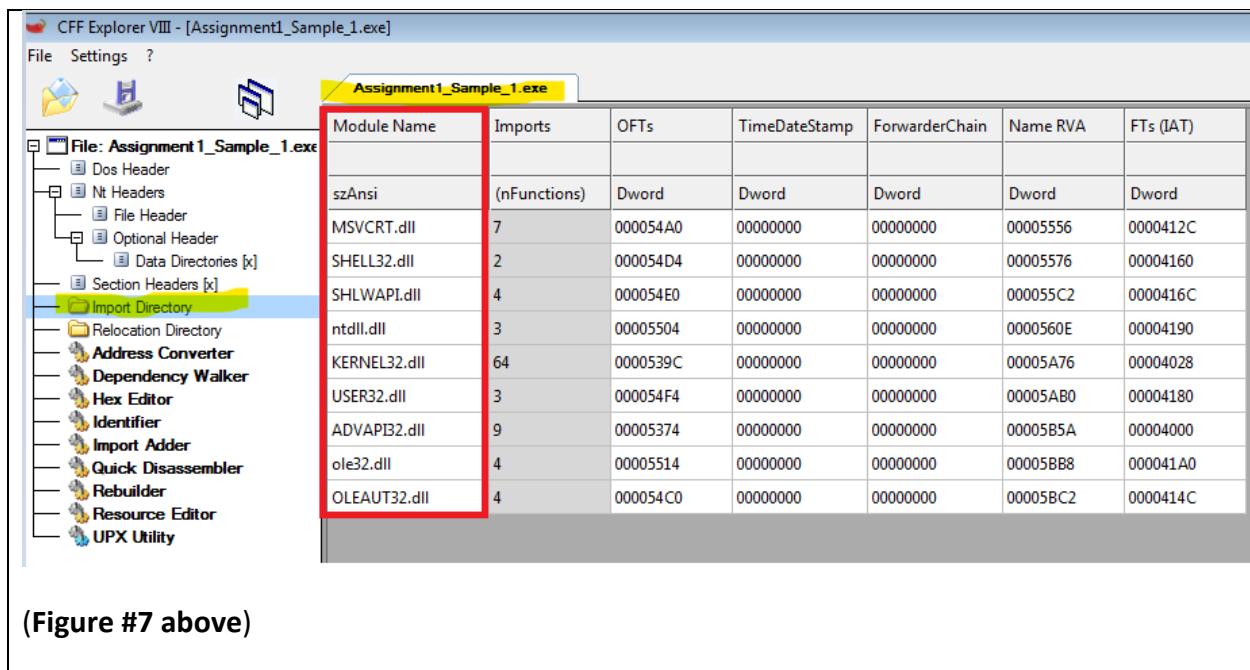
(Figure #5 above)



(Figure #6 above) (List of File Signatures, 2023)

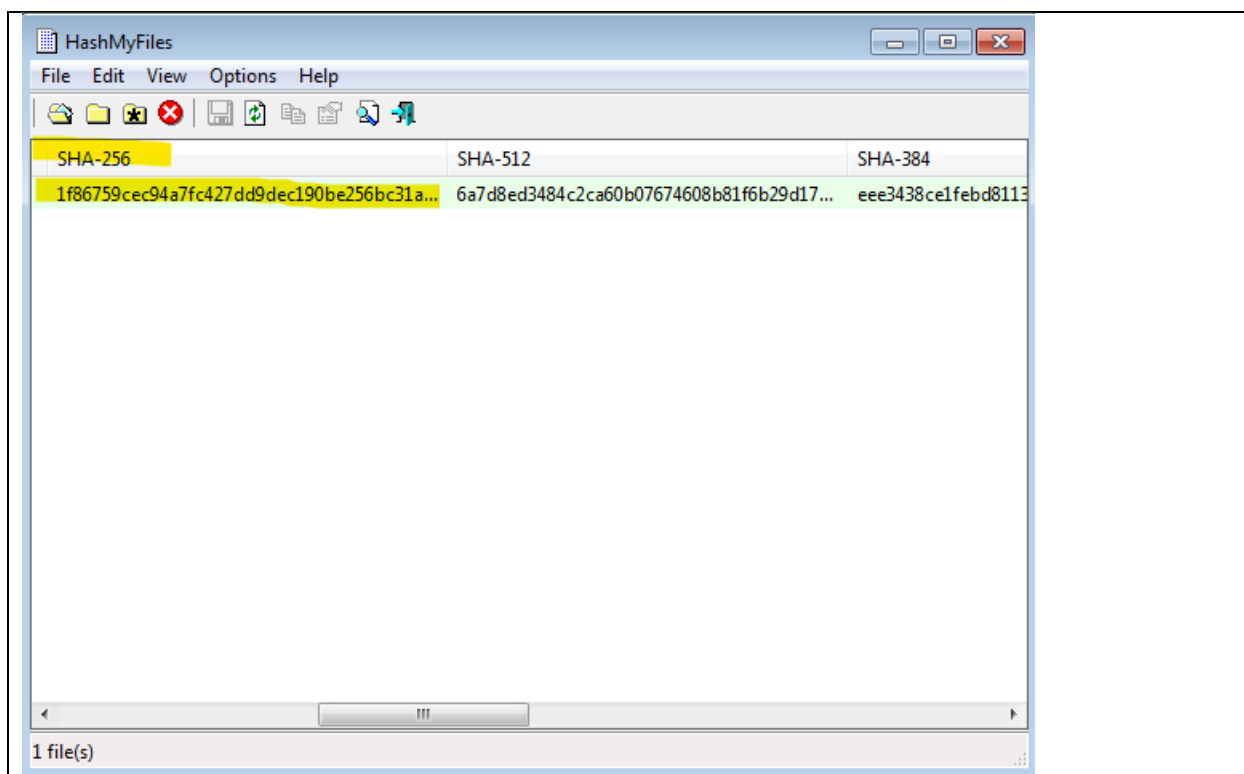
### Output#3: Identifying libraries and packages for file execution

The libraries or packages the sample need to be executed are shown in Figure #7 below.



#### Output#4: Calculating the hash of the file

To calculate the hash of the file, we use the tool “HashMyFiles” as shown in Figure #8. To check when the file was last analysed, we use VirusTotal and search the hash as shown in Figure #9. As you can see, it was last analysed on “2023, 01, 18 03:50:48 UTC” (Assignment1\_Sample\_1, 2019).



(Figure #8 above)

History ⓘ	
Creation Time	2011-08-02 09:26:00 UTC
First Seen In The Wild	2019-06-21 23:28:32 UTC
First Submission	2019-09-11 04:57:21 UTC
Last Submission	2023-01-26 16:51:41 UTC
Last Analysis	2023-01-18 03:50:48 UTC

(Figure #9 above) Assignment1\_Sample\_1, 2019

## Output#5: Identifying suspicious strings

To identify suspicious strings, we use BinText to analysis the sample. Firstly, we can see there are a set of anti virus vendors on Figure #10. I believe the virus is scanning its environment and checking if that anti-virus exists as the malware is armoring itself. Next suspicious string was on Figure #11 “[\\?\globalroot\system32\vmx\\_fb.dll](#)”. Upon google searching, I found that someone posted on the internet regarding this string in which it is not a legitimate DLL


on a windows machine and the users computer became unstable in the post as shown on Figure #12 & Figure #13. Finally, these subset of strings in the form of registry keys looked very suspicious in Figure #14. Upon google searching I came across that it is a remote access related string as shown in Figure #15 & Figure #16.


A	00000004A6F8	000000448AF8	0	avast.com
A	00000004A704	000000448B04	0	kaspersky
A	00000004A710	000000448B10	0	drweb
A	00000004A718	000000448B18	0	eset.com
A	00000004A724	000000448B24	0	antivir
A	00000004A72C	000000448B2C	0	avira
A	00000004A734	000000448B34	0	virusinfo
A	00000004A740	000000448B40	0	z-oleg.com
A	00000004A74C	000000448B4C	0	kltest.org.ru
A	00000004A758	000000448B58	0	trendsecure
A	00000004A768	000000448B68	0	anti-malware
A	00000004A774	000000448B74	0	comodo.com
A	00000004A784	000000448B84	0	

(Figure# 10 above)

A	00000005C290	00000045C290	0	dumpcap.exe
A	00000005C29C	00000045C29C	0	idag.exe
A	00000005C2A8	00000045C2A8	0	vmwaretray.exe
A	00000005C2B8	00000045C2B8	0	\\?\globalroot\systemroot\system32\vmx_fb.dll
A	00000005C2E8	00000045C2E8	0	SystemDrive
A	00000005C2F4	00000045C2F4	0	software\microsoft\windows nt\currentversion\winlogon
A	00000005C330	00000045C330	0	software\microsoft\windows\currentversion\run
A	00000005C360	00000045C360	0	userinit

(Figure #11 above)

 The application or DLL globalroot\systemroot\system32\\*.dll is not a Windows machine. Please check this against your installation diskette  
Started by htimbuck , Oct 10 2009 11:33 PM

 This topic is locked

(Figure #12 above) (Htimbuck, 2009)

Posted 10 October 2009 - 11:33 PM

Hi,

I was browsing ppcgeeks.com and suddenly started getting tons of popups of fake windows security center.

I ran all virus checks , Symantec, McAfee and found that my attempts to perform any activity was preceded by a message "The application or DLL globalroot\systemroot\system32\\*.dll is not a Windows machine. Please check this against your installation diskette".

I have been struggling for the past few hours and the computer has nearly become unusable, I am running a Windows XP machine and the hijackthis transcript is below .  
 Help is very much appreciated.

(Figure #13 above) (Htimbuck, 2009)

File to scan				C:\Users\Jackson Yuan\Desktop\Virus\A1\Assignment1_Sample_1.exe
<input checked="" type="checkbox"/> Advanced view				
File pos	Mem pos	ID	Text	
A 0000000A7B28	0000004A7B28	0	Local\{AAFEE2BF-8989-4fe1-9A0D-95CD39DC0A14}	
A 0000000A7B58	0000004A7B58	0	FAKTURA	
A 0000000A7B60	0000004A7B60	0	sks2xyz.dll	
A 0000000A7B6C	0000004A7B6C	0	vb_pfx_import	
A 0000000A7B9C	0000004A7B9C	0	\*.bk	
A 0000000A7BA4	0000004A7BA4	0	Local\{EAF7eafF-8989-4fe1-9A0D-95CD777C0214}	
A 0000000A7BD4	0000004A7BD4	0	HANDY	
A 0000000A7CD4	0000004A7CD4	0	Local\{EAF799BF-8989-4fe1-9A0D-95CD39DC0214}	
A 0000000A7D04	0000004A7D04	0	IBANK	
A 0000000A7D40	0000004A7D40	0	BEGIN SIGNATURE	
A 0000000A7D50	0000004A7D50	0	END SIGNATURE	
A 0000000A7D60	0000004A7D60	0	secret.key	
A 0000000A7D78	0000004A7D78	0	pubkeys.key	
A 0000000A7D84	0000004A7D84	0	Local\{AAF799BF-8989-4fe1-9A0D-95CD39DC0A14}	
A 0000000A7DB4	0000004A7DB4	0	INIST	
A 0000000A7DBC	0000004A7DBC	0	path1.txt	
A 0000000A7DC8	0000004A7DC8	0	inter.zip	
A 0000000A7DD4	0000004A7DD4	0	interpro.ini	
A 0000000A7DE4	0000004A7DE4	0	DefaultPrivateDir	
A 0000000A7DF8	0000004A7DF8	0	General	
A 0000000A7E00	0000004A7E00	0	Local\{EAF329BF-8989-4fe1-9A0D-95CD39DC0214}	
A 0000000A7E30	0000004A7E30	0	INTER	
A 0000000A7E38	0000004A7E38	0	cbssmain.dll	
A 0000000A7E6C	0000004A7E6C	0	Local\{BE3C9D87-B777-4e47-8B10-69798A04C732}	
A 0000000A7EA0	0000004A7EA0	0	&txtSubId=	
A 0000000A7EAC	0000004A7EAC	0	&txtPin=	
A 0000000A7EB8	0000004A7EB8	0	ebank.laiki.com	
A 0000000A7EC8	0000004A7EC8	0	pass.txt	
A 0000000A7ED4	0000004A7ED4	0	Local\{EAF339BF-8989-4fe1-9A0D-95CD39DC0214}	
A 0000000A7F04	0000004A7F04	0	OFFSHORE	
A 0000000A7F10	0000004A7F10	0	w.qiwi.ru	
A 0000000A7F1C	0000004A7F1C	0	phone=	
A 0000000A7F2C	0000004A7F2C	0	Local\{EAF799BF-8989-4fe1-9A0D-95CD777C0214}	
A 0000000A7F5C	0000004A7F5C	0	FilialRCn.dll	
A 0000000A7F6C	0000004A7F6C	0	RCN_R50BBuffer	
A 0000000A7F7C	0000004A7F7C	0	RAIFF	
A 0000000A7F84	0000004A7F84	0	ISClient.cfg	
A 0000000A7F94	0000004A7F94	0	GKUZ=	
A 0000000A7FA0	0000004A7FA0	0	Local\{EAF777BF-8989-4fe1-9A0D-95CD777C0214}	
A 0000000A7FD0	0000004A7FD0	0	rfk.zip	
A 0000000A7FD8	0000004A7FD8	0	client.zip	
A 0000000A7FE4	0000004A7FE4	0	path_client.txt	
A 0000000A7FF4	0000004A7FF4	0	\SIGN1\	
A 0000000A7FFC	0000004A7FFC	0	path_keys.txt	
A 0000000A800C	0000004A800C	0	Local\{EAF777FF-8989-4fe1-9A0D-95CD777C0214}	
A 0000000A8040	0000004A8040	0	Local\{EAF777FF-8989-4fe1-977D-95CD777C0214}	
A 0000000A8070	0000004A8070	0	RSTYLE	
A 0000000A8078	0000004A8078	0	Agava_Client.exe	
A 0000000A808C	0000004A808C	0	UseToken	
A 0000000A8090	0000004A8090	0	Contains...	

(Figure #14 above)



https://www.hybrid-analysis.com › sample

## sample - Hybrid Analysis

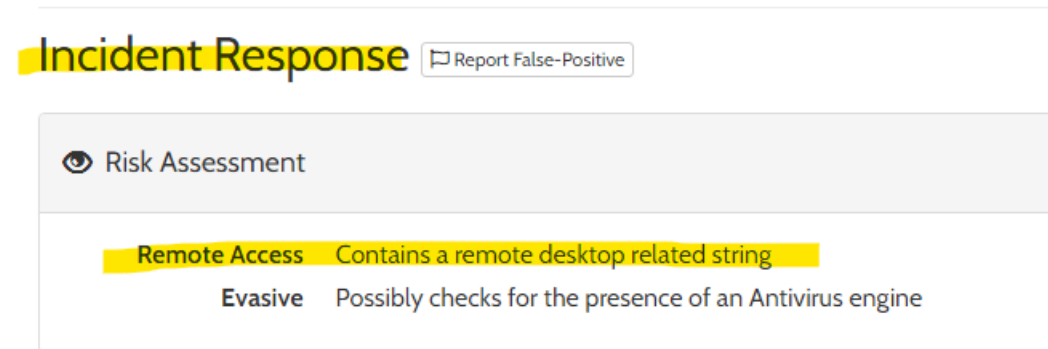
Local{AAFEE2BF-8989-4fe1-9A0D-95CD39DC0A14}. Ansi based on Memory/File Scan (4d58a5a83a6ce10202473afba04ff6b40178d89641900f15b2768410e5b0c944).

https://www.hybrid-analysis.com › sample

## orig\_sample - Hybrid Analysis

Local{AAFEE2BF-8989-4fe1-9A0D-95CD39DC0A14}. Ansi based on Memory/File Scan (606cd815e946531ae4b79258f7b7105d8629e1659ef1eb5aec90654a5e027ffa).

(Figure #15 above)



(Figure #16 above) (String Analysis, 2018)

### Output#6: Identifying the malware family

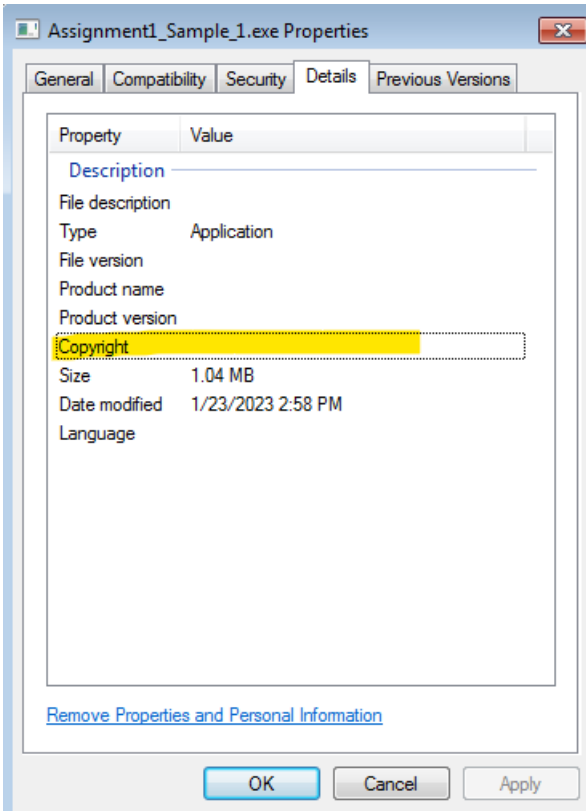
As discussed in class, there does not exist any universal naming convention however, the closes example I can find to the CARO Malware Naming Scheme is in Figure #17, which states that the malware family is identified as “starter” (Assignment1\_Sample\_1, 2019). Additional, information on this sample is that the malware type is a Trojan as described in Figure #17 and infects Windows 32 platform. Finally, the group name for this malware is “ali1000030” as indicated in Figure #17.



(Figure #17 above) (Assignment1\_Sample\_1, 2019)

### Output#7: Identifying code signing certificate

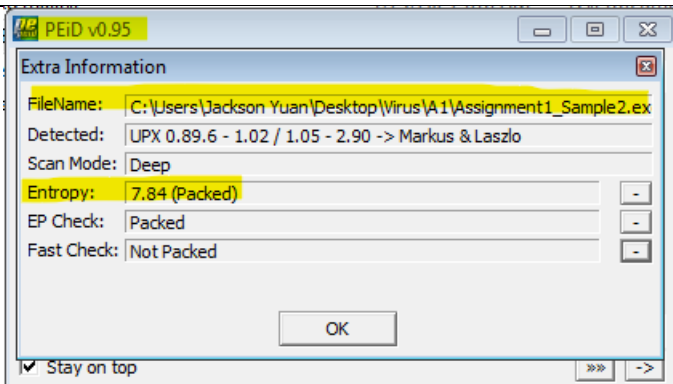
There does not exist any code signing certificate for this sample as shown below



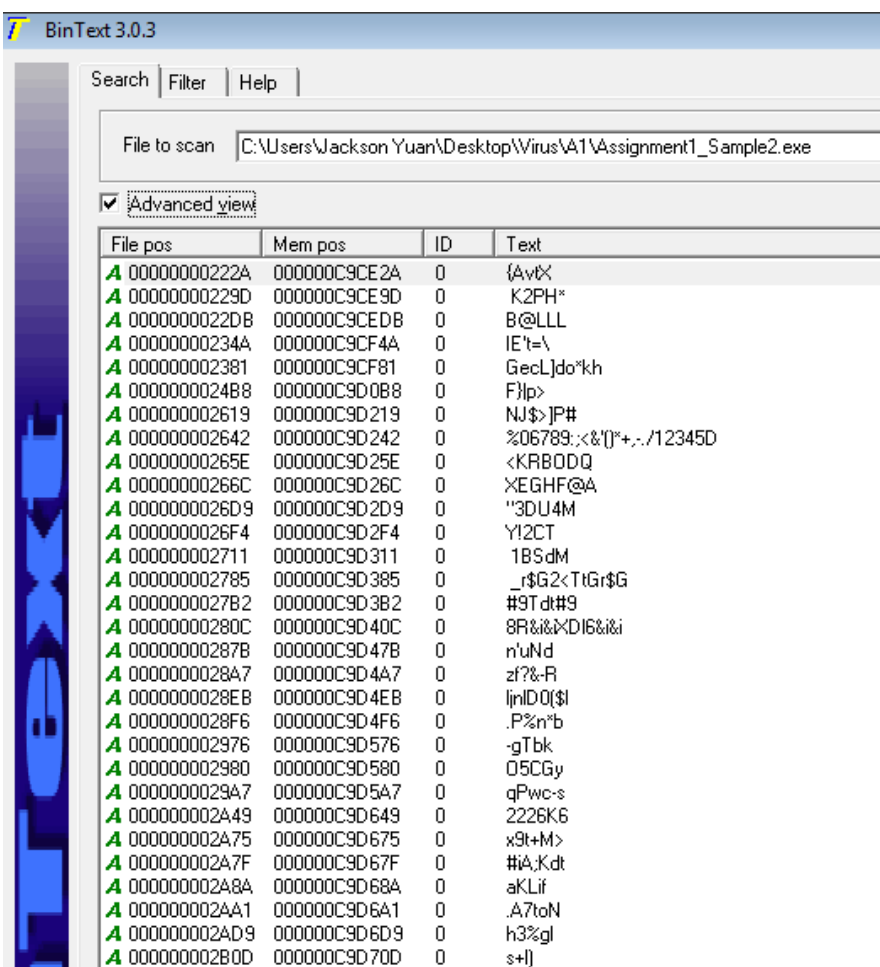
## Output Section for Sample #2

### Output#1: Packed or Unpacked Analysis

We can check if it's packed by calculating the files entropy. The tool I have used to do so is PEiD. Figure #18 demonstrates this as it shows an entropy 7.84 which is packed. To cross check this, we can also check BinText to see how much randomness in the strings shown below in Figure #19.



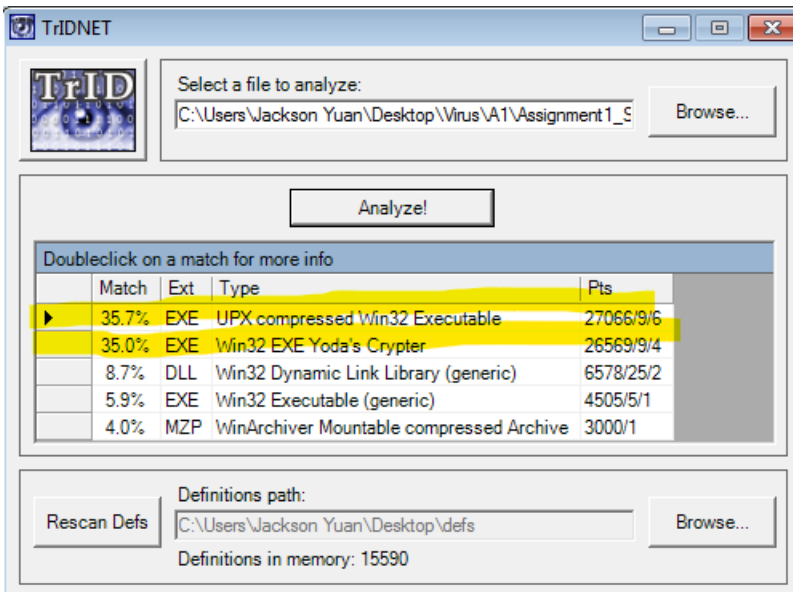
(Figure #18 above)



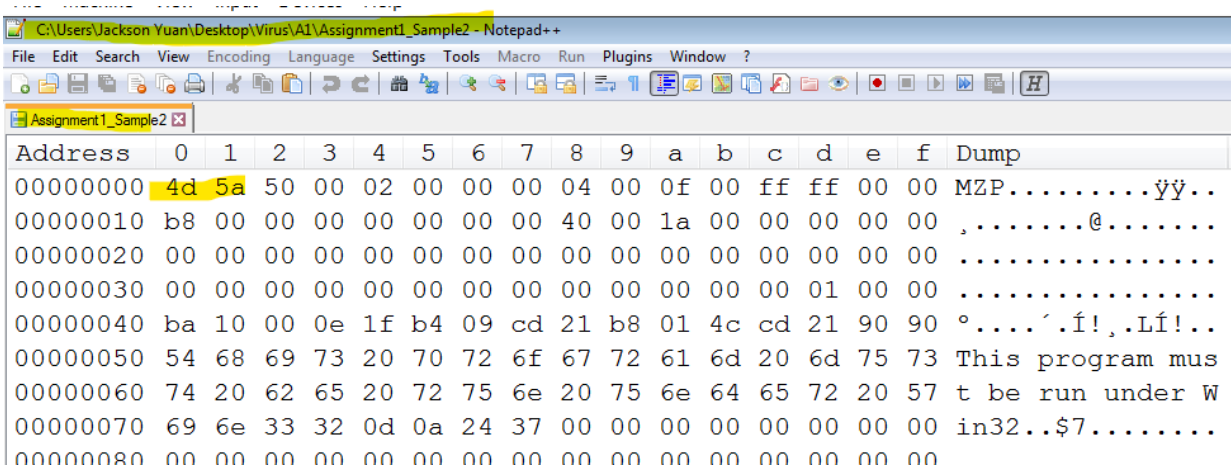
(Figure #19 above)

## Output#2: File Format Identification

The file format for sample #2 is an “.exe” as shown in Figure #20. Furthermore, we can double check with notepad++ on the magic bytes shown in Figure #21; and by cross referencing with [Figure #6](#), we can confirm it’s a “.exe” file format. The packer is “Yoda’s Crypter” shown in Figure #20.



(Figure #20 above)



(Figure #21 above)

### Output#3: Identifying libraries and packages for file execution

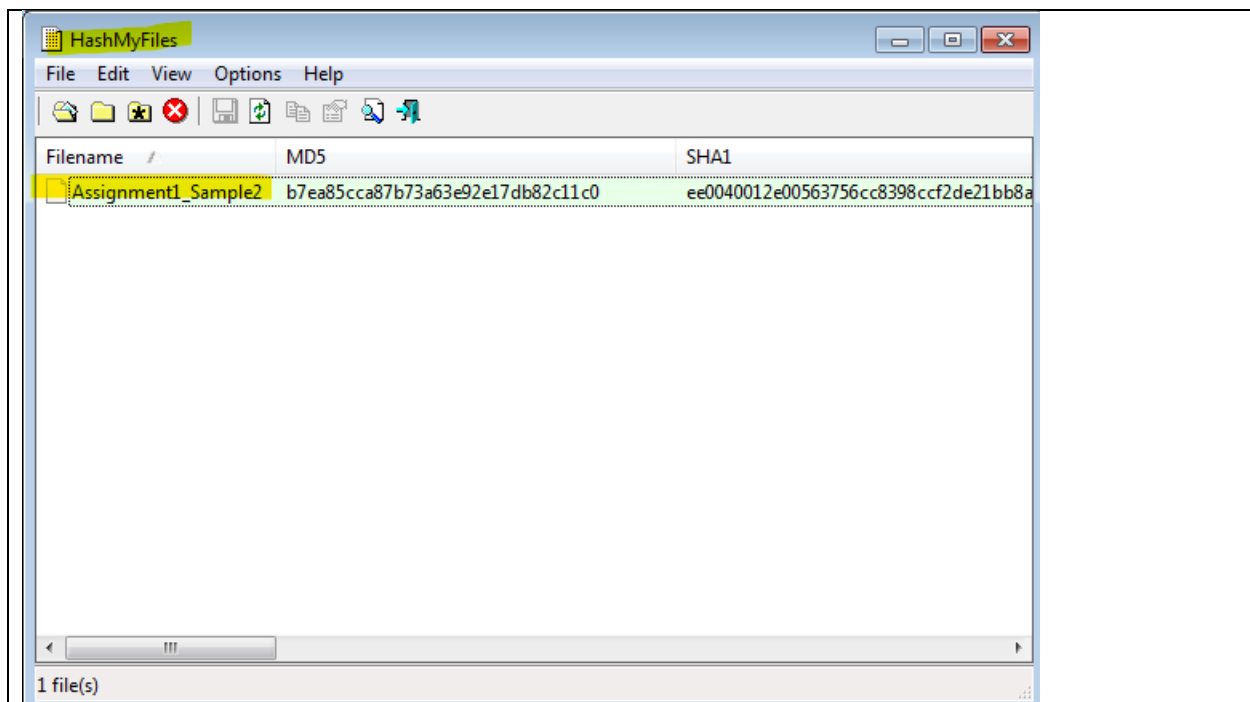
The list of libraires and packages are listed below in Figure #22

Assignment1_Sample2						
Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.DLL	6	00000000	00000000	00000000	000387A8	00038744
advapi32.dll	1	00000000	00000000	00000000	000387B5	00038760
ntdll.dll	1	00000000	00000000	00000000	000387C2	00038768
oleaut32.dll	1	00000000	00000000	00000000	000387CC	00038770
PSAPI.dll	1	00000000	00000000	00000000	000387D9	00038778
shell32.dll	1	00000000	00000000	00000000	000387E3	00038780
shlwapi.dll	1	00000000	00000000	00000000	000387EF	00038788
urlmon.dll	1	00000000	00000000	00000000	000387FB	00038790
user32.dll	1	00000000	00000000	00000000	00038806	00038798
wininet.dll	1	00000000	00000000	00000000	00038811	000387A0

(Figure #22 above)

#### Output#4: Calculating the hash of the file

To calculate the hash, we use the tool “HashMyFiles” as shown in Figure #23. Furthermore, I have used virus total to find when the last time this sample was analyzed and it shows “2023-01-29 22:25:16 UTC” (*Assignment1\_Sample2*, 2016) as shown in Figure #24.



(Figure #23 above)

#### History ⓘ

Creation Time	1992-06-19 22:22:17 UTC
First Seen In The Wild	2016-01-29 14:15:33 UTC
First Submission	2016-01-29 14:23:31 UTC
Last Submission	2023-01-31 15:50:35 UTC
Last Analysis	2023-01-29 22:25:16 UTC

(Figure #24 above) (Assignment1\_Sample2, 2016)

### Output#5: Identifying suspicious strings

One suspicious string state writes “keylog” as shown in Figure #25 and in Figure #26, it shows a very suspicious copyright name and upon googling it, a user confirms its indeed a backdoor trojan in Figure #27 (BC\_Programming, 2012).

```

A 0000000072E8 000000CA1EE8 0 d:K[PH
A 00000000735C 000000CA1F5C 0 Config
A 000000007364 000000CA1F64 0 KWindow
A 00000000737D 000000CA1F7D 0 2Func
A 000000007389 000000CA1F89 0 Keylogg
A 0000000073A1 000000CA1FA1 0 lallA
A 000000007457 000000CA2057 0 .w\p\X.
A 000000007498 000000CA2098 0 R=xk}
A 00000000761B 000000CA221B 0 j380~m
A 0000000076C2 000000CA22C2 0 a:xs2

```

(Figure #25 above)

```

A 000000001047 000000C9BC47 0 tions Copyright
A 000000001056 000000C9BC56 0 (c) 19
A 000000001061 000000C9BC61 0 ,2003 Avenger by NhT

```

(Figure #26 above)

"Portions Copyright (c) 1999,2003 Avenger by NhT"

Well actually, that's the only interesting one I found.

Anyway, it's a backdoor trojan Horse. Virus total on this:

<https://www.virustotal.com/file/330bfc556275c512f4b59a463f38565fb0c8bbf2d117167e49b5d6286a20e76f/analysis/>

(Figure #27 above) (BC\_Programming, 2012)

## Output#6: Identifying the malware family

As discussed in class, there does not exist any universal naming convention however, the closes example I can find to the CARO Malware Naming Scheme is shown in Figure #28, and the malware family of this sample is "Scar" (*Assignment1\_Sample2*, 2016). Additional information on this virus is that the malware type is a Trojan both confirmed in Figure #27 and in Figure #28 and infects Windows 32 platform. Finally, the group name for this malware is "R15220" as indicated in Figure #28.

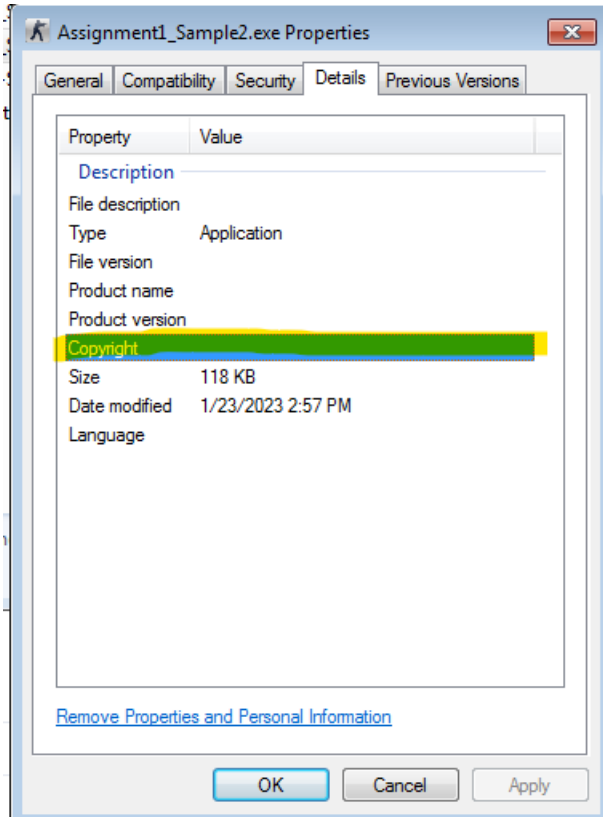
AhnLab-V3

! Trojan/Win32.Scar.R15220

(Figure #28 above) (*Assignment1\_Sample2*, 2016)

## Output#7: Identifying code signing certificate

There does not exist any code signing certificate on this sample as shown below however, the malware author tries to fake a copyright indicated on [Figure # 26](#) during strings analysis.





## References

- Assignment1\_Sample\_1*. (2019, June 21). Virus Total. Retrieved January 30, 2023, from <https://www.virustotal.com/gui/file/1f86759cec94a7fc427dd9dec190be256bc31ab58c9df3e3281f463db68e6ae7/details>
- Assignment1\_Sample2*. (2016, January 29). Virus Total. Retrieved January 31, 2023, from <https://www.virustotal.com/gui/file/3f50009c9460cce36879bedf7f173e939934db829cb640d2dc8fe4a00801971d>
- BC\_Programming. (2012, November 18). *Re: recent poster that posted a malware program*. Minecraft Forum. <https://www.minecraftforum.net/forums/off-topic/computer-science-and-technology/488204-re-recent-poster-that-posted-a-malware-program>
- Htimbuck, [Htimbuck]. (2009, October 10). *The application or DLL globalroot\systemroot\system32\\*.dll is not a Windows machine. Please check this against your installation diskette*. BleepingComputer.com. Retrieved January 30, 2023, from <https://www.bleepingcomputer.com/forums/t/263622/the-application-or-dll-globalrootssystemrootssystem32dll-is-not-a-windows-machine-please-check-this-against-your-installation-diskette/>
- List of file signatures*. (2023, January 21). Wikipedia. Retrieved January 30, 2023, from [https://en.wikipedia.org/wiki/List\\_of\\_file\\_signatures](https://en.wikipedia.org/wiki/List_of_file_signatures)
- String Analysis*. (2018, February 4). Hybrid Analysis. Retrieved January 30, 2023, from <https://www.hybrid-analysis.com/sample/4d58a5a83a6ce10202473afba04ff6b40178d89641900f15b2768410e5b0c944?environmentId=110>