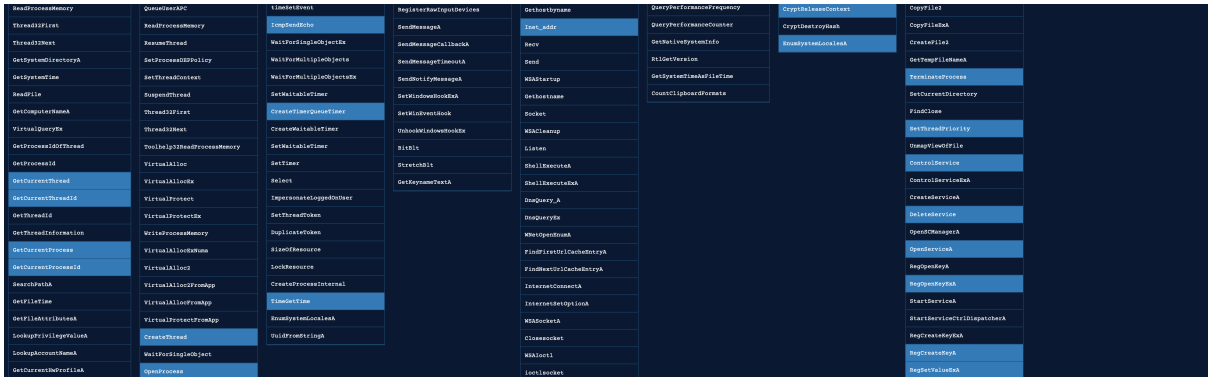


Ransomware//MedusaLocker

Sept. 27th, 2022. Written and Analyzed by Av4x

Twitter:

twitter.com/av4xor

Email:

av4x@av4x.su

Executive Summary:

SHA256 Hash	54b8ca90cd5c6b8053a612d2e8d99bf05f427b36e7fccc0f63427e1f386db186
Language	C
Architecture	x86/32 bits
File type	Executable (.exe)
OS	Windows

Medusa Locker is a strain of ransomware written in C which affects Windows systems. Medusa behaves similarly to most other ransomware families already in the wild and seeks to quickly infect a system, enumerate it, and then encrypt and lock down files to demand a ransom payment from the victim.

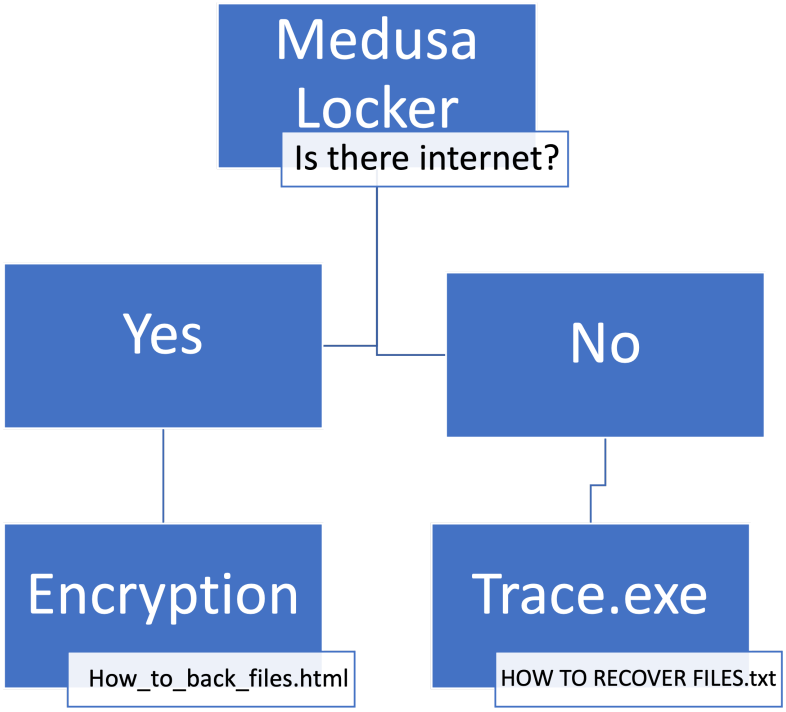
Medusa Locker is spread primarily via social engineering tactics where an attacker convinces an unsuspecting user to download and execute a file on their system. Unlike some strains of ransomware, Medusa quickly makes its presence known on the system due to it's method of infection. Once executed, Medusa causes Windows to prompt the user with a popup stating that the system needs to be rebooted in order to turn off User Access Control followed by several more popups stating that there are new external drives on the system. Meanwhile, Medusa Locker is deleting any backups saved on the host computer, creating necessary processes for persistence and to encrypt the system once it restarts, and enumerating the local network looking for more hosts to infect.

Once the host computer has been rebooted, all files will have been encrypted and have the *.netlock* extension (this can differ, Medusa chooses from a list of different extensions to use or generates them). There will be two files on the desktop called "how_to_back_files.html" which contain instructions on how to pay the ransom necessary to unlock files.

High-Level Technical Summary

Medusa Locker acts similar to other Ransomware with its ability to quickly encrypt a file system along with a ransom note and instructions for payment being left on the infected host. Medusa does however differ due to its ability to also infect external drives and devices that may be connected to a computer. On top of this, Medusa also enumerates the local network searching for other hosts to possibly infect meaning essentially that it also has worm capabilities.

Medusa Locker also has a list of processes that it will kill either to deactivate anti-virus or ensure that there is no data loss and that all files can be encrypted. Encryption is done via AES+RSA.



Strings:

“Fail to schedule the chore!”

“broken promise”

“future already retrieved”

“promise already satisfied”

“RAgui.exe,supervise.exe,Culture.exe,RTVscan.exe,GDscan.exe,ZhuDongFangYu.exe”

“.deadfiles,.netlock,.marlock1,.lockdata1,.locklock,.far18k,.”

“Your personal ID”

“<!— !!! dont changing this !!! —>”

“/!\ YOUR COMPANY NETWORK HAS BEEN PENETRATED /!\”

“All your important files have been encrypted!”

“ithelp08@decorous.cyou”

“ithelp08@wholeness.business”

“minkernel\crts\ucrt\inc\corecrt_internal_strtox.h”

Libraries:

crypt32.dll	mpr.dll
netapi32.dll	iphlpapi.dll
ws2_32.dll	rstrtmgr.dll

API Calls:

Initial Detonation:

Initial detonation started showing signs of infection immediately. The first sign is a Windows popup stating that to turn off User Access Control the system needs to be rebooted, which is followed by AutoPlay displaying 3 external drives being loaded onto the system as seen in Figure E.1.

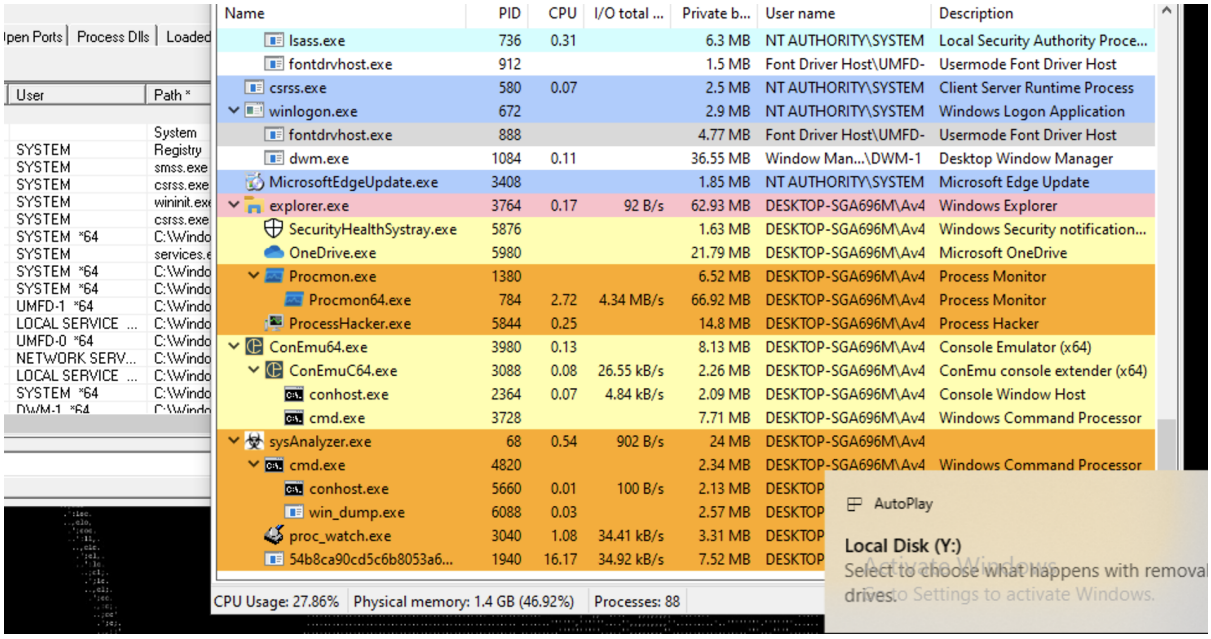


Figure E.1

I ran this binary via sysAnalyzer which worked to list down the API's called, directorys modified/created and other useful information. In Figure E.2 you can see part of the dump from the process.

The binary quickly went to enumerate the entire file system but it did this by also running specific commands as it went from directory from directory (seen in figures E.2-E.5).

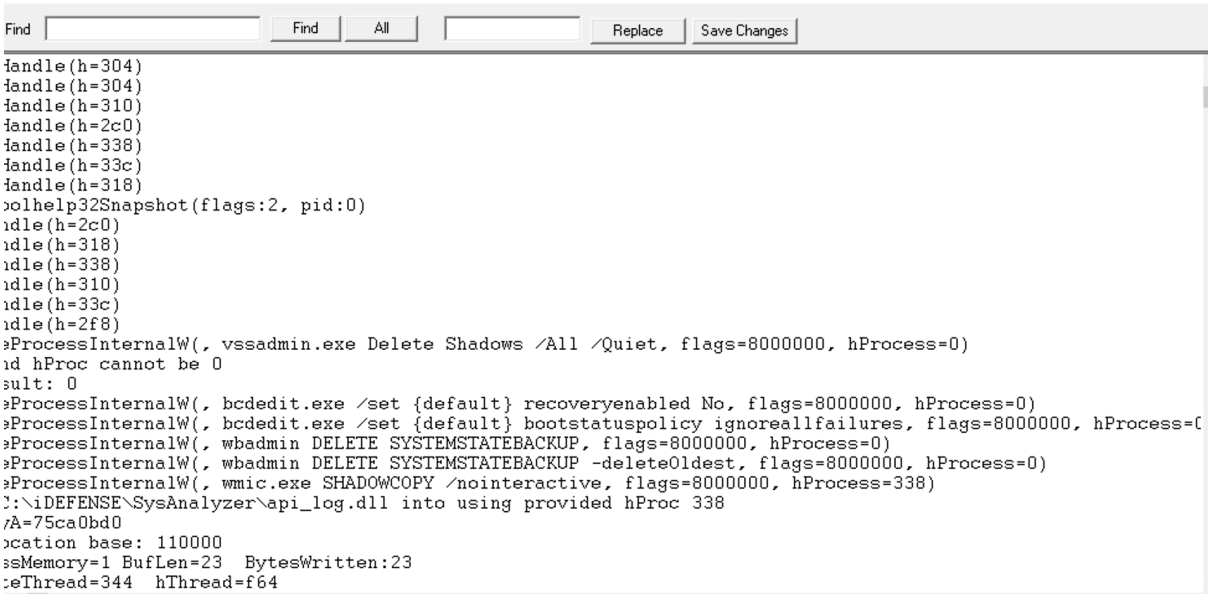


Figure E.2

The binary used vssadmin.exe and bcdedit.exe to delete all shadow backup files as well as ensuring that there is no system recovery methods available.

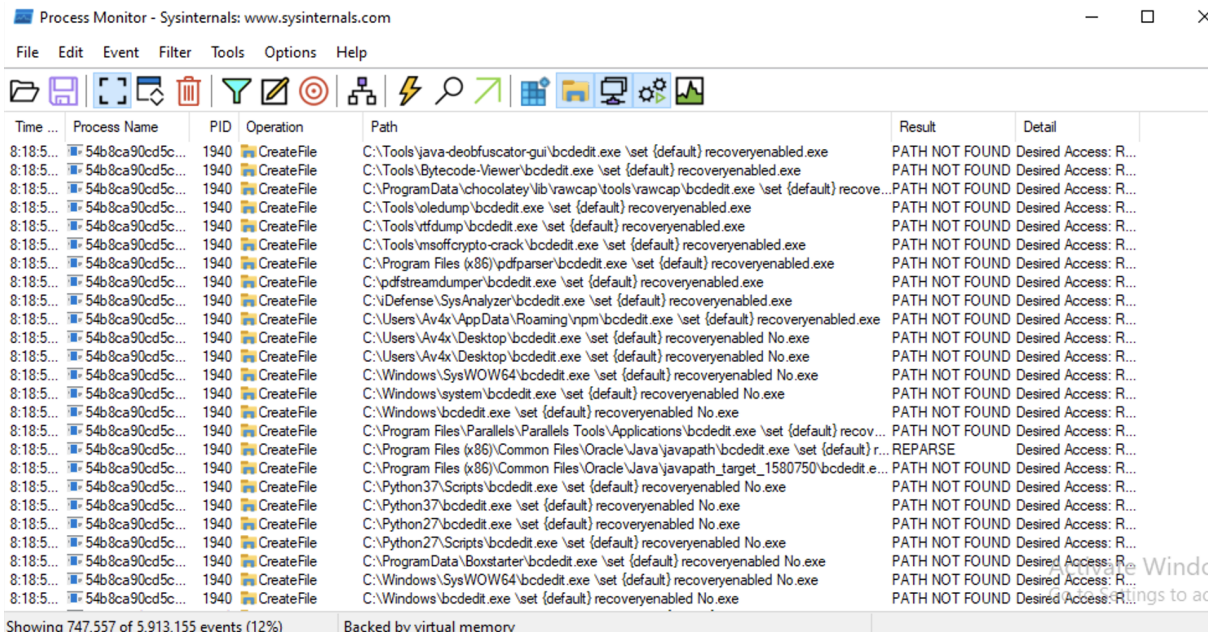


Figure E.3

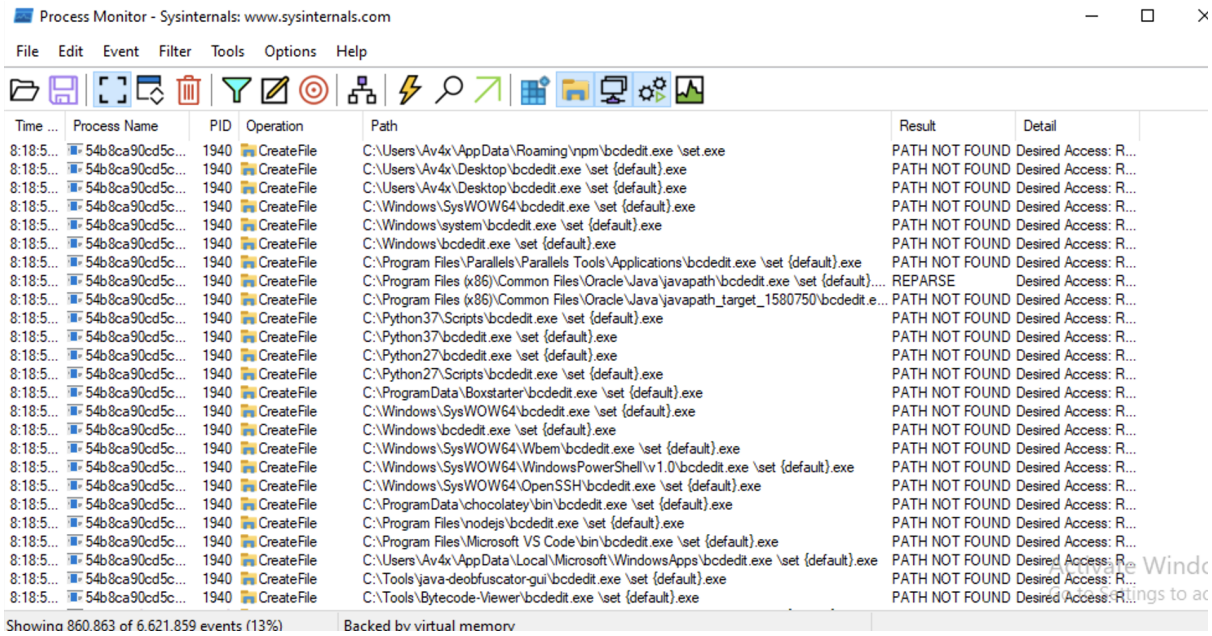


Figure E.4

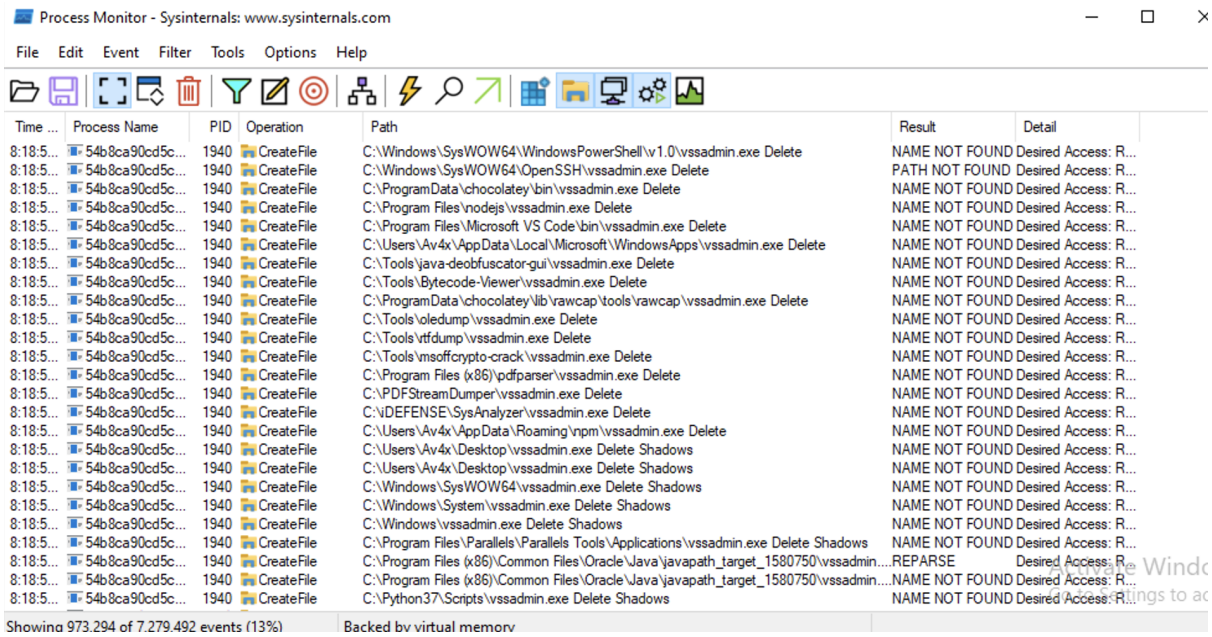


Figure E.5

Displayed in Figure E.6 are the external drives that were installed onto the system:

- Y:
- Z:

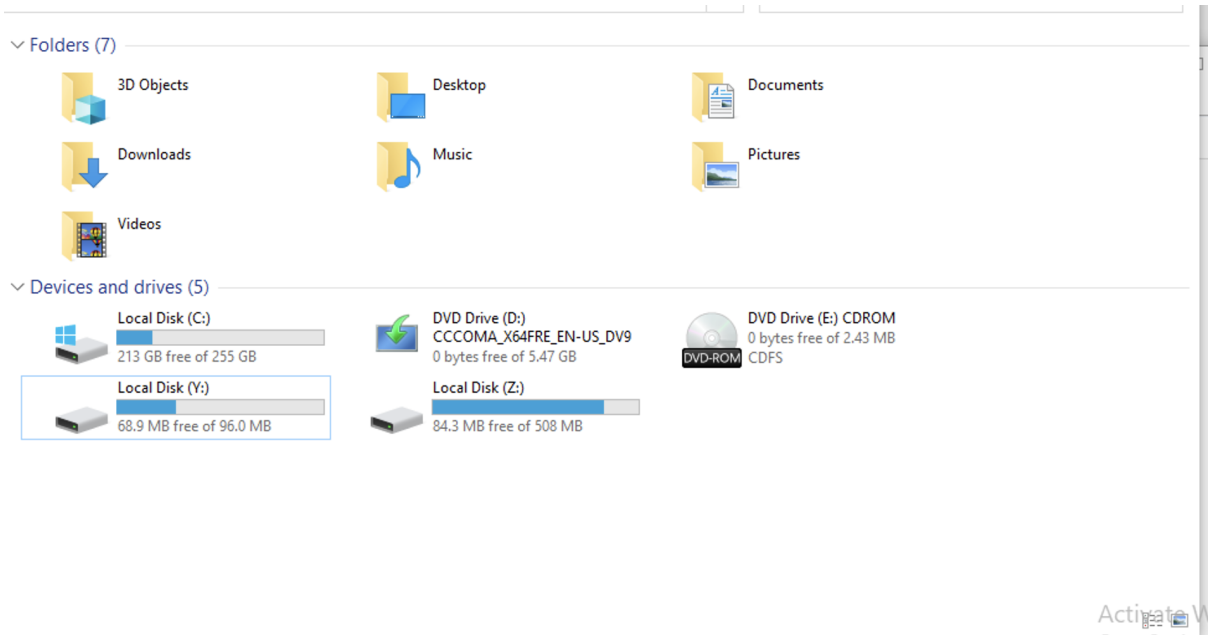
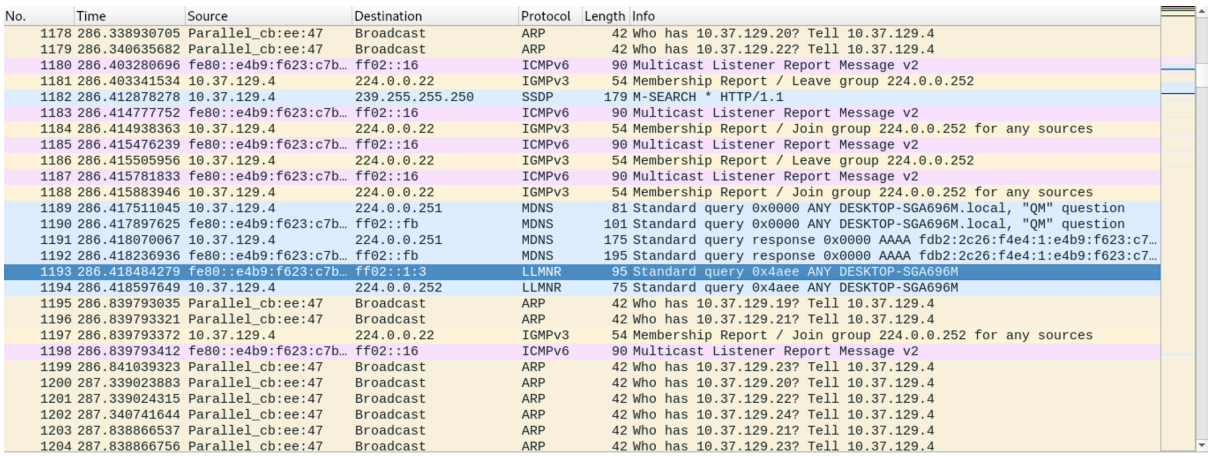


Figure E.6

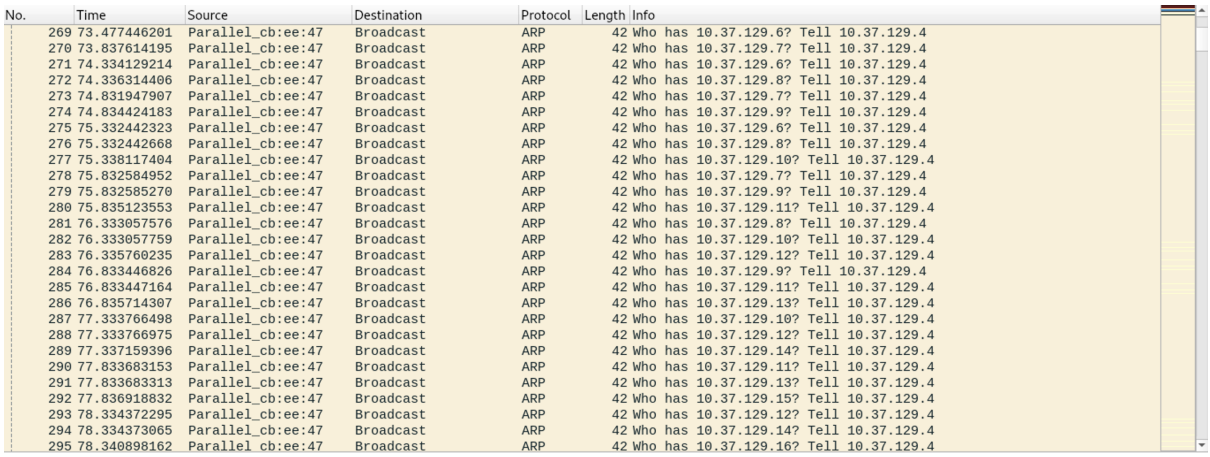
While the binary was running through the system and making sure to delete backups it's also enumerating the local network and attempting to locate any other local hosts to infect. This means that this ransomware also has worm capabilities to spread itself. (Figure E.7-E.8)

This seems to be done via LLMNR which may indicate an attempt to spread itself via LLMNR poisoning.



No.	Time	Source	Destination	Protocol	Length	Info
1178	286.338930705	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.20? Tell 10.37.129.4
1179	286.340635682	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.22? Tell 10.37.129.4
1180	286.403280696	fe80::e4b9:f623:c7b…	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
1181	286.403341534	10.37.129.4	224.0.0.22	IGMPv3	54	Membership Report / Leave group 224.0.0.252
1182	286.412878278	10.37.129.4	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1
1183	286.414777752	fe80::e4b9:f623:c7b…	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
1184	286.414938363	10.37.129.4	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.252 for any sources
1185	286.415476239	fe80::e4b9:f623:c7b…	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
1186	286.415505956	10.37.129.4	224.0.0.22	IGMPv3	54	Membership Report / Leave group 224.0.0.252
1187	286.415781833	fe80::e4b9:f623:c7b…	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
1188	286.415883946	10.37.129.4	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.252 for any sources
1189	286.417511045	10.37.129.4	224.0.0.251	MDNS	81	Standard query 0x0000 ANY DESKTOP-SGA696M.local, "QM" question
1190	286.417897625	fe80::e4b9:f623:c7b…	ff02::fb	MDNS	101	Standard query 0x0000 ANY DESKTOP-SGA696M.local, "QM" question
1191	286.418070067	10.37.129.4	224.0.0.251	MDNS	175	Standard query response 0x0000 AAAA fdb2:2c26:f4e4:1:e4b9:f623:c7…
1192	286.418236936	fe80::e4b9:f623:c7b…	ff02::fb	MDNS	195	Standard query response 0x0000 AAAA fdb2:2c26:f4e4:1:e4b9:f623:c7…
1193	286.418484279	fe80::e4b9:f623:c7b…	ff02::1:3	LLMNR	95	Standard query 0x4aee ANY DESKTOP-SGA696M
1194	286.418597649	10.37.129.4	224.0.0.252	LLMNR	75	Standard query 0x4aee ANY DESKTOP-SGA696M
1195	286.839793035	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.19? Tell 10.37.129.4
1196	286.839793321	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.21? Tell 10.37.129.4
1197	286.839793372	10.37.129.4	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.252 for any sources
1198	286.839793412	fe80::e4b9:f623:c7b…	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
1199	286.841039323	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.23? Tell 10.37.129.4
1200	287.339023883	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.20? Tell 10.37.129.4
1201	287.339024315	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.22? Tell 10.37.129.4
1202	287.340741644	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.24? Tell 10.37.129.4
1203	287.838866537	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.21? Tell 10.37.129.4
1204	287.838866756	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.23? Tell 10.37.129.4

Figure E.7



No.	Time	Source	Destination	Protocol	Length	Info
269	73.477446201	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.6? Tell 10.37.129.4
270	73.837614195	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.7? Tell 10.37.129.4
271	74.334129214	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.6? Tell 10.37.129.4
272	74.336314406	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.8? Tell 10.37.129.4
273	74.831947907	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.7? Tell 10.37.129.4
274	74.834424183	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.9? Tell 10.37.129.4
275	75.332442323	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.6? Tell 10.37.129.4
276	75.332442668	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.8? Tell 10.37.129.4
277	75.338117404	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.10? Tell 10.37.129.4
278	75.832584952	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.7? Tell 10.37.129.4
279	75.832585270	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.9? Tell 10.37.129.4
280	75.835123553	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.11? Tell 10.37.129.4
281	76.333057576	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.8? Tell 10.37.129.4
282	76.333057759	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.10? Tell 10.37.129.4
283	76.335760235	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.12? Tell 10.37.129.4
284	76.833446826	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.9? Tell 10.37.129.4
285	76.833447164	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.11? Tell 10.37.129.4
286	76.835714307	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.13? Tell 10.37.129.4
287	77.333766498	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.10? Tell 10.37.129.4
288	77.333766975	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.12? Tell 10.37.129.4
289	77.337159396	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.14? Tell 10.37.129.4
290	77.833683153	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.11? Tell 10.37.129.4
291	77.833683313	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.13? Tell 10.37.129.4
292	77.836918832	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.15? Tell 10.37.129.4
293	78.334372295	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.12? Tell 10.37.129.4
294	78.334373065	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.14? Tell 10.37.129.4
295	78.340898162	Parallel_cb:ee:47	Broadcast	ARP	42	Who has 10.37.129.16? Tell 10.37.129.4

Figure E.8

During this entire process the system becomes extremely difficult to use due to how much it slows down. One of the programs I had been running crashed or was shut down.

The infectious file later forces a reboot of the system and once it's booted back up, all of the files have been encrypted and have a .netlock extension. There are two files generated on the desktop detailing how to decrypt your system and send payment, shown below.

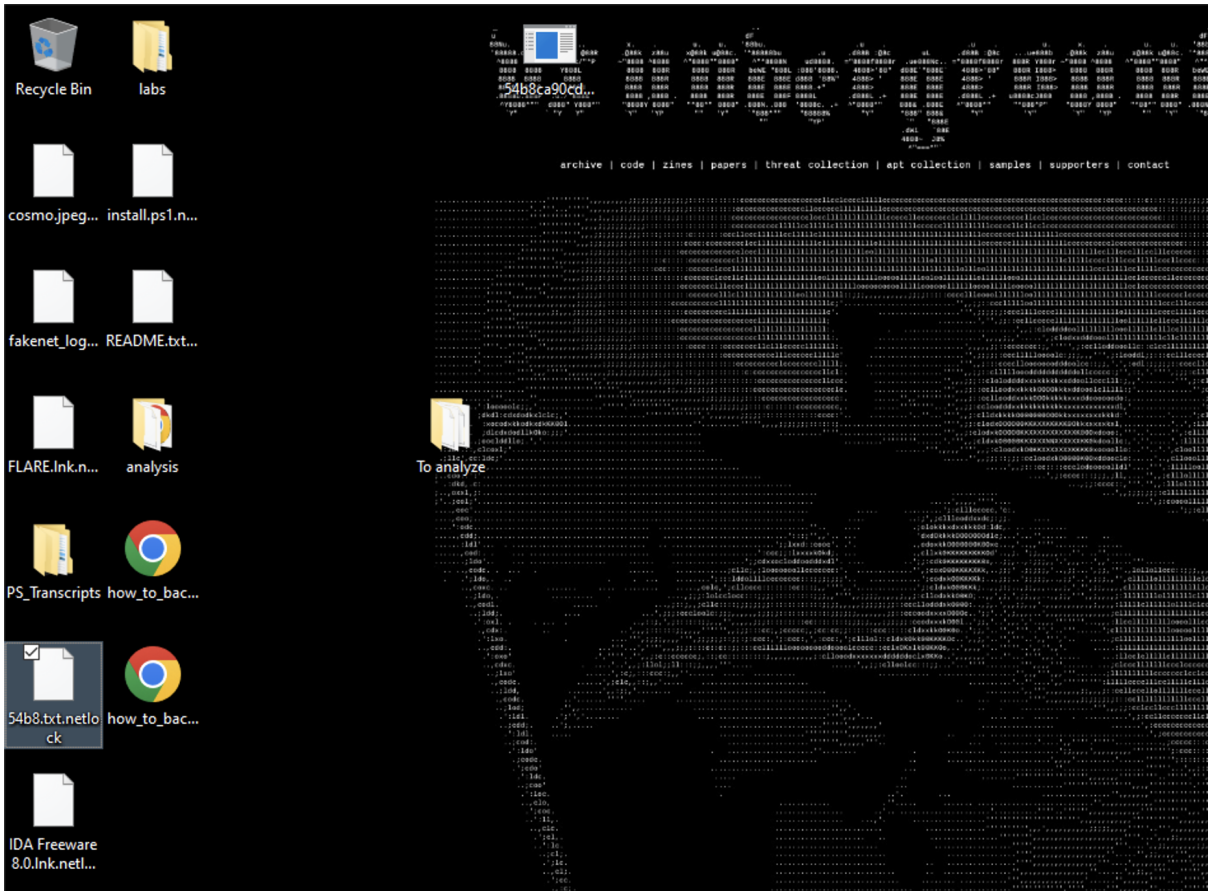

Recycle Bin
labs
cosmo.jpeg...
install.ps1.n...
fakenet_log...
README.txt...
FLARE.lnk.n...
analysis
PS_Transcripts
how_to_bac...
54b8.txt.netlock
how_to_bac...
IDA Freeware 8.0.lnk.netl...
54b8ca90cd...
archive | code | zines | papers | threat collection | apt collection | samples | supporters | contact
To analyze

Figure E.9

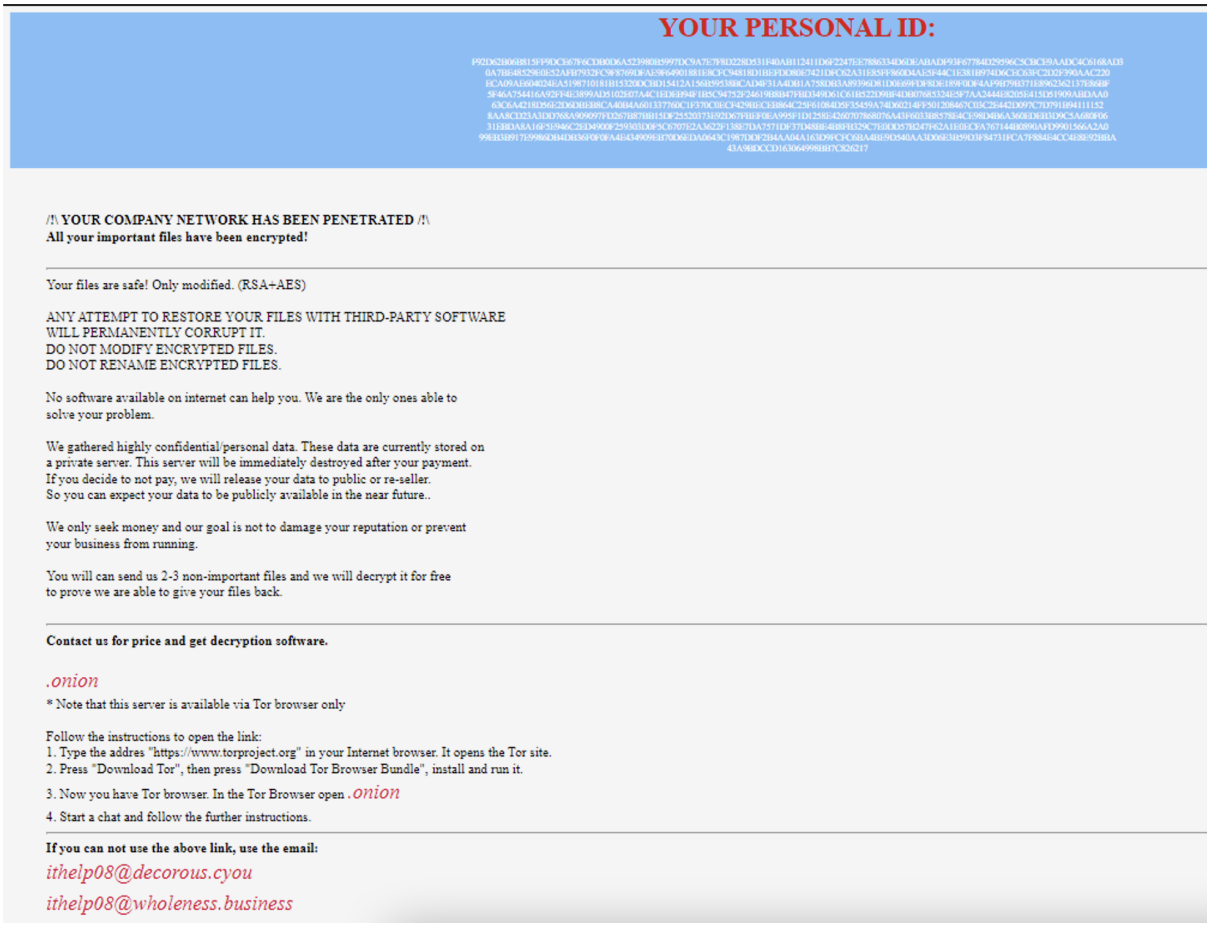


Figure E.10

Secondary Detonation:

With inetsim turned off, and thus the system completely without internet the binary will still encrypt the entire system but works a bit different. Instead of the .netlock file extension, files are now locked with .dfjfkdf (seemingly randomly generated) and a file called HOW TO RECOVER FILES.txt (E.12) takes the place of the previous ransom note seen before:

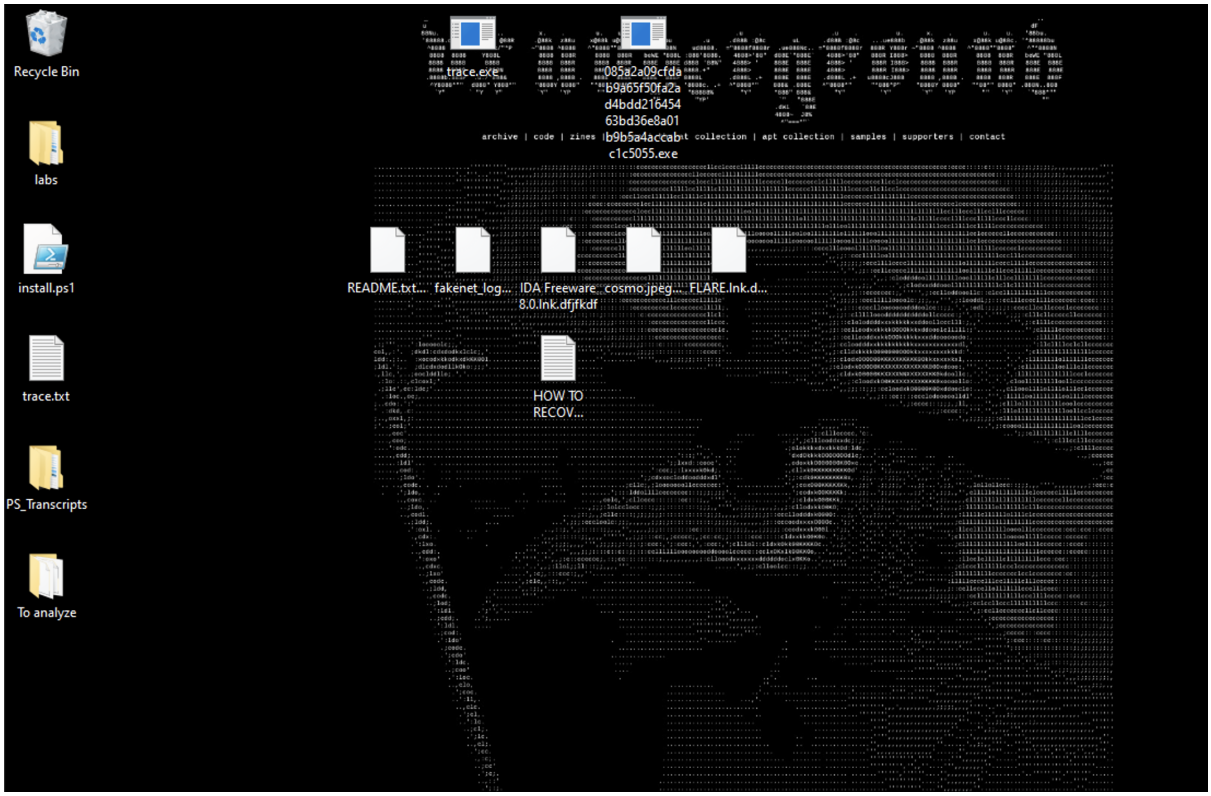


Figure E.11

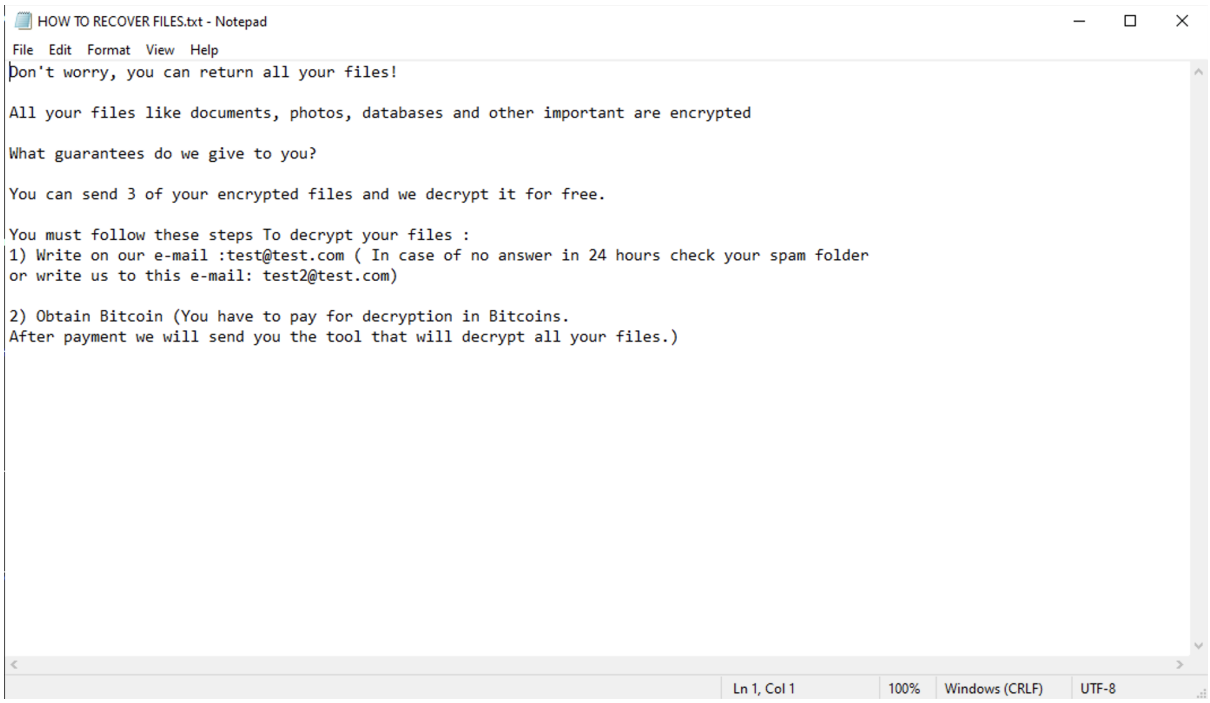


Figure E.12

Without internet, another new file called trace.exe takes the place of the original. My disassembler states its written in CIL (disassembler also states original file name was 333.exe).

This file contains some interesting strings, most of which seem to be custom variables such as “forbiddenCountry”:

userName	userDir	encryptionAesRsa
checkSpread	checkSleep	checkAdminPrivilage

sleepOutOfTempFolder	forbiddenCountry	rsaKey
AES_Encrypt	RSA_Encrypt	RandomStringForExtension
copyResistForAdmin	SetWallaper	CultureInfo/get_Culture
op_Equality	op_Inequality	<lookForDirectories>b__b
get_FriendlyName	ApartmentState	soon.exe

Note: The file soon.exe appears to install itself on drives that it hasn't yet infected. Trace.exe tried to create this file on the E:\ and D:\ drives but failed due to denied access

Advanced Static Analysis:

Medusa doesn't seem to be encrypted or obfuscated much. When analyzing the binary with cutter, the strings are readily available and easy to analyze.

Shown in figure E.13 is a hardcoded string that all variants of Medusa share as a mutex identifier.

```
0x00405ca5      lea     eax, [var_140h]
0x00405cab      push    eax             ; int32_t arg_8h
0x00405cac      call    fcn.004261d1
0x00405cb1      add     esp, 8
0x00405cb4      lea     ecx, [var_140h]
0x00405cba      call    fcn.00405540
0x00405cbf      lea     ecx, [var_148h]
0x00405cc5      call    fcn.00405540
0x00405cca      push    str.LOCKER__Is_running ; 0x48268c
0x00405ccf      lea     ecx, [var_f7h]
0x00405cd5      call    fcn.00401100
0x00405cda      mov     ecx, eax
0x00405cdc      call    fcn.004017b0
0x00405ce1      push    str.8761ABBD_7F85_42EE_B272_A76179687C63 ; 0x4826b8 ; int32_t arg_8h
0x00405ce6      lea     ecx, [var_160h]
0x00405cec      call    fcn.00407cd0
0x00405cf1      lea     ecx, [var_160h]
0x00405cf7      push    ecx             ; int32_t arg_8h
0x00405cf8      call    fcn.00405630
0x00405cfd      add     esp, 4
0x00405d00      mov     byte [var_d5h], al
0x00405d06      lea     ecx, [var_160h]
0x00405d0c      call    fcn.00407b40
0x00405d11      movzx   edx, byte [var_d5h]
0x00405d18      test    edx, edx
0x00405d1a      je      0x405d3a
0x00405d1c      push    str.LOCKER__Is_already_running ; 0x482708
0x00405d21      lea     ecx, [var_f8h]
0x00405d27      call    fcn.00401100
0x00405d2c      mov     ecx, eax
0x00405d2e      call    fcn.004017b0
0x00405d33      xor     eax, eax
0x00405d35      jmp     0x4064f0
0x00405d3a      lea     ecx, [var_6h]
```

Figure E.13. String: 8761ABBD-7F85-42EE-B272-A76179687C63

Medusa then checks whether it has permissions as admin or user and if it doesn't have admin level permissions it will use a User Access Control bypass method. This also explains why a popup regarding User Access Control being disabled appears immediately after detonation.

```
0x00405d5d      test    eax, eax
0x00405d5f      je      0x405d6d
0x00405d61      mov     dword [var_10ch], str.LOCKER__Priv:_ADMIN ; 0x482744
0x00405d6b      jmp     0x405d77
0x00405d6d      mov     dword [var_10ch], str.LOCKER__Priv:_USER ; 0x482770
0x00405d77      mov     ecx, dword [var_10ch]
0x00405d7d      mov     dword [var_130h], ecx
0x00405d83      lea     edx, [var_130h]
0x00405d89      push    edx
0x00405d8a      lea     ecx, [var_f9h]
0x00405d90      call    fcn.00401100
0x00405d95      mov     ecx, eax
0x00405d97      call    fcn.004017b0
0x00405d9c      call    fcn.00405680
0x00405da1      lea     ecx, [var_74h]
0x00405da4      call    fcn.00415480
0x00405da9      push    str.LOCKER__Init_cryptor ; 0x48279c
0x00405dae      lea     ecx, [var_fah]
0x00405db4      call    fcn.00401100
0x00405db9      mov     ecx, eax
0x00405dbb      call    fcn.004017b0
0x00405dc0      mov     ecx, 0x4a3ac0
0x00405dc5      call    fcn.00401100
0x00405dca      push    eax             ; uint32_t arg_8h
0x00405dcb      call    fcn.00401650
0x00405dd0      add     esp, 4
0x00405dd3      push    eax             ; int32_t arg_8h
0x00405dd4      lea     ecx, [var_74h]
0x00405dd7      call    fcn.00415500
0x00405ddc      movzx   eax, al
0x00405ddf      test    eax, eax
0x00405de1      jne     0x405e24
0x00405de3      push    str.LOCKER__Init_cryptor_is_failed ; 0x482808
0x00405de8      lea     ecx, [var_fbh]
```

Figure E.14

More can be read about UAC Bypass Methods here.

Medusa Locker quickly adds itself to autorun, starts up svhost and begins scanning the host for backup drives or hidden devices before shutting down services and killing processes. This also explains why in the initial detonation one of my applications died.

```
0x00405ef5      mov     eax, dword [var_138h]
0x00405efb      jmp     0x4064f0
0x00405f00      lea     ecx, [var_ah]
0x00405f03      call    fcn.00401100
0x00405f08      push    str.LOCKER__Add_to_autorun ; 0x48285c
0x00405f0d      lea     ecx, [var_efh]
0x00405f13      call    fcn.00401100
0x00405f18      mov     ecx, eax
0x00405f1a      call    fcn.004017b0
0x00405f1f      push    str.svhost ; 0x4828e4 ; int32_t arg_8h
0x00405f24      lea     ecx, [var_190h]
0x00405f2a      call    fcn.00407cd0
0x00405f2f      mov     ecx, 0x4a3ac0
0x00405f34      call    fcn.00412320
0x00405f39      push    eax             ; uint32_t arg_ch
0x00405f3a      lea     eax, [var_190h]
0x00405f40      push    eax             ; int32_t arg_8h
0x00405f41      lea     ecx, [var_ah]
0x00405f44      call    fcn.0041f730
0x00405f49      lea     ecx, [var_190h]
0x00405f4f      call    fcn.00407b40
0x00405f54      push    str.LOCKER__Scan_hidden_devices ; 0x4828f4
0x00405f59      lea     ecx, [var_f0h]
0x00405f5f      call    fcn.00401100
0x00405f64      mov     ecx, eax
0x00405f66      call    fcn.004017b0
0x00405f6b      lea     ecx, [var_9h]
0x00405f6e      call    fcn.00401100
0x00405f73      lea     ecx, [var_9h]
0x00405f76      call    fcn.0041c020
0x00405f7b      push    str.LOCKER__Stop_and_delete_services ; 0x482930
0x00405f80      lea     ecx, [var_f1h]
0x00405f86      call    fcn.00401100
0x00405f8b      mov     ecx, eax
```

Figure E.15

This ransomware also appears to search for files related to VMware to see whether or not the host is on a VM. I ran this analysis running Parallels and however and I couldn't find that it checked for that.

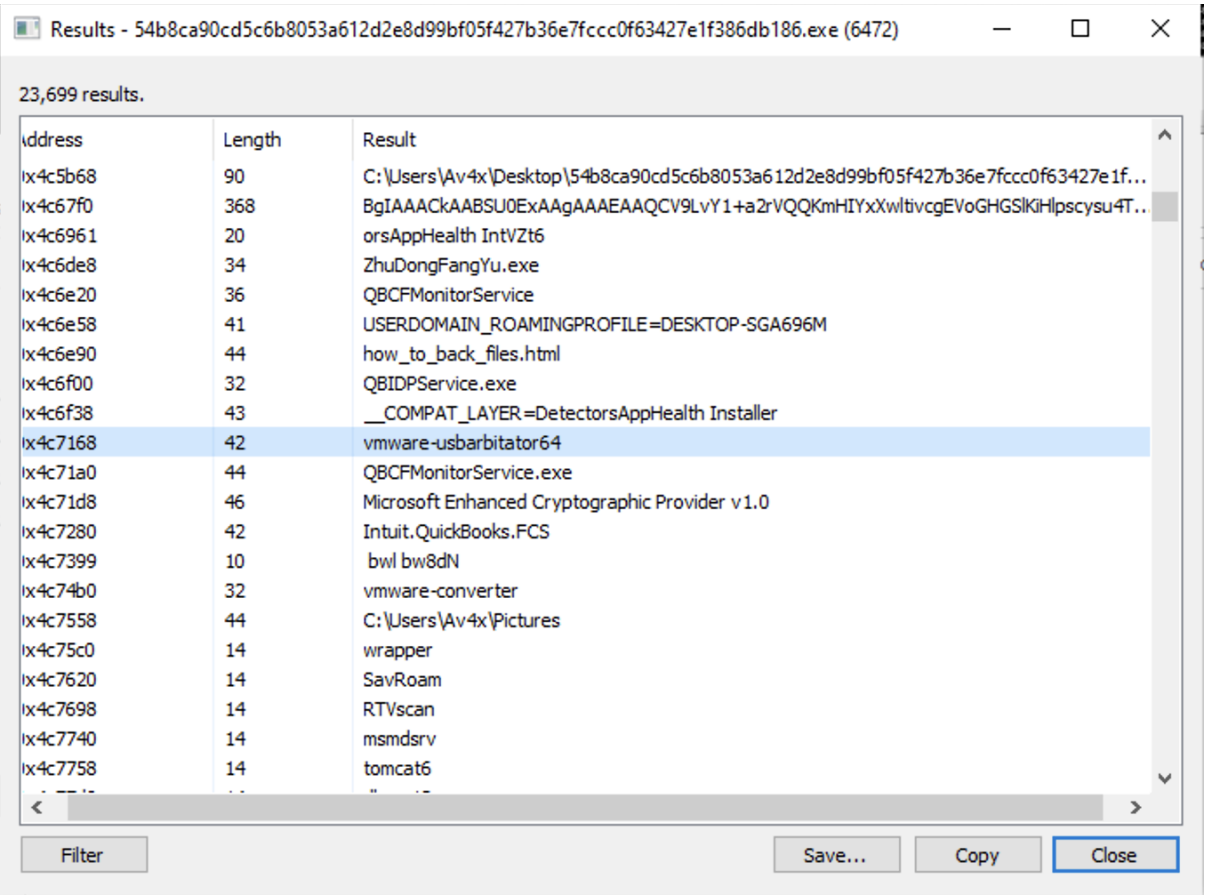


Figure E.16

Indicators of Compromise:

Network Indicators:

Medusa Locker did not reach out to any remote hosts or attempt connections to a C2 or create a backdoor. The only network indicators of compromised that were displayed during my analysis was the enumerating of the local network via ARP requests (Figure E.7 & E.8). This attempt at infecting other hosts on the network was very noisy and would probably be picked up by a SIEM or SoC.

Host-Based Indicators:

Most of the host-based indicators that come along with Medusa aren't present until the system has been infected. The only available IoC's available pre-detonation are the presence of suspicious .exe files which can be detected via YARA rules (See Appendix A). The YARA rules for this ransomware will check the strings unique to Medusa and will help in identifying any suspicious files.

However, once the binary has been exected Medusa's IoC's include encryption of files with .netlock (or other) extensions unique to it, high CPU usage, UAC popup, external drives :Y and :Z, processes being killed, a unrequested system reboot, and the presence of the previously mentioned ransom notes on the desktop.

Appendix A - YARA Rules:

```
rule Yara_MedusaLocker {

    meta:
        last_updated = "2022-08-28"
```

```
        author = "Av4x"
        description = "A simple YARA rule for detecting MedusaLocker"
    strings:
        $string1 = ".netlock" ascii
        $string2 = "{8761ABBD-7F85-42EE-B272-A76179687C63}"
        $string3 = "/!\ YOUR COMPANY NETWORK HAS BEEN PENETRATED /!\" ascii
        $string4 = "[LOCKER]"
        $string5 = "ithelp08@decorous.cyou"
        $string6 = "vssadmin.exe DELETE"
        $string7 = "bcdedit.exe \set (default).exe"
}
```

.dll,.sys,.ini,.rdp,.encrypted,.exe,.network6,.datalock17,.datalock18,.datalock19,.datalock20,.LOCK1,.lockhyp,.LOCK1,.LOCK2,.l

.sql,.mdf

wrapper,DefWatch,ccEvtMgr,ccSetMgr,SavRoam,sqlservr,sqlagent,sqladhlp,Culserver,RTVscan,sqlbrowser,SQLADHLP,QBIDP
usbarbitator64,vmware-converter,dbsrv12,dbeng8
wxServer.exe,wxServerView,sqlservr.exe,sqlmangr.exe,RAgui.exe,supervise.exe,Culture.exe,RTVscan.exe,Defwatch.exe,sqlbr

how_to_back_files.html