

Locked Out: An analysis of the Lockbit 2.0 Ransomware

3/12/23

Connor Fuchs

cfuchs@ufl.edu

Practical 2 - CAP4136

Executive Summary

Lockbit 2.0 is a malicious piece of Windows x86 GUI ransomware that infiltrates a user's system and then a network of systems, and encrypts and steals data. It first checks a user's drives, copies data in all directories, places ransom notes in these directories, encrypts all files with a .lockbit extension, and changes the desktop background to a large ransom note.

Lockbit 2.0 appears to exploit ADDs (Active Directory Domain Services) with LDAP (Lightweight Directory Access Protocol), as well as Group Policy security settings to escalate privilege and gain access to users data over a network. Notable imports to accomplish this include CreateWellKnownSid, ADsOpenObject, FreeADsMem, GetADComputer, and the LDAP Root DSE.

Lockbit 2.0 maintains persistence and administrator access by modifying and adding registry keys, notably a new key with a custom Security Identifier (SID). It continuously modifies security settings via an LSA Shell command, getting the current user and iterating through each user, updating Security Policy settings. The malware encrypts files via some form of XOR encryption, shown in the dynamic analysis section. Monitoring network processes via InetSim, Fakedns, Apatedns, and Wireshark proved to be questionable at best. There is a flurry of network activity upon launching the malware, and Lockbit 2.0 seems to make use of mailslots as a means of interprocess communication and data harvesting. Given the nature of the encryptions performed and presence of .onion websites, finding destinations and sources proved difficult.

There were some significant host and network based indicators that were utilized in creating a rule for recognizing the malware in the future. The command that the malware uses to delete part of itself, `/C ping 127.0.0.7 -n 3 > Nul & fsutil file setZeroData offset=0 length=524288 \"%s\" & Del /f /q \"%s\"`, as well as a provided URL found in strings, a reference to "RESTORE-MY-FILES.txt", and the first bytes of the header (`uint16(0) == 0x5a4d`) all successfully identify the existence of the sample on a system. A conditional which AND's all of these conditions together results in no false positives detected.

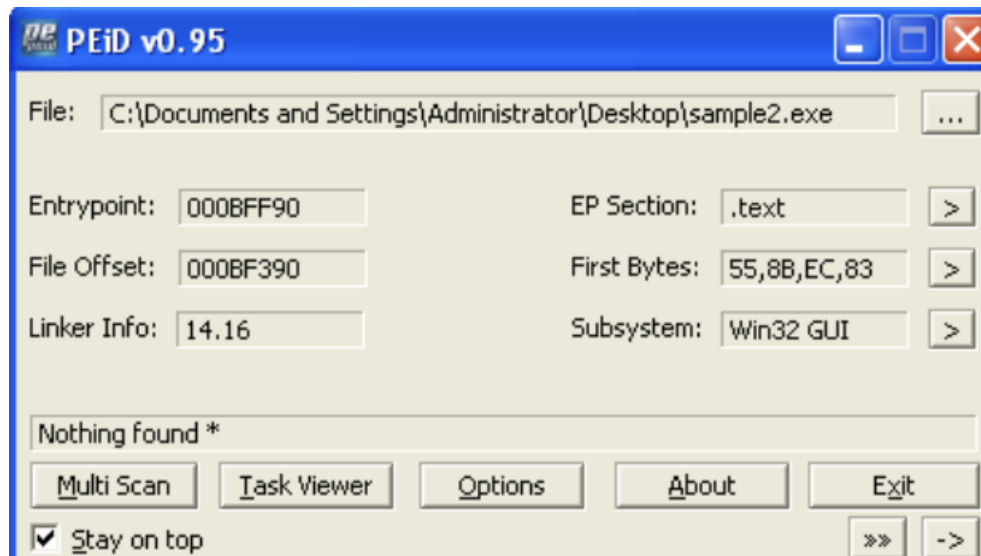
Static Analysis

The compiler stamp reads that the program was compiled Monday, July 26 at 7:34, 2021. It appears as though the program uses Windows GUI. A stub program is invoked when a file is run in MS-DOS, and the resulting message explains. The dos stub for sample2.exe in pestudio has the message: "This program cannot be run in DOS mode!" Additionally, PEiD lists the subsystem utilized as Windows GUI. PEview also lists the subsystem being utilized as IMAGE_SUBSYSTEM_WINDOWS_GUI.

One indicator that points towards the program being packed is the entropy. An entropy level of 6.682 is getting to be relatively high, but it is not definite, and indicates that the program has an amount of randomness.

Another indicator of packedness is a large difference between virtual size and raw size of the program. If the raw size is small and virtual memory assignment large, the program has likely been packed. In this program's case, the virtual size is 1015933 bytes, and the raw size is 981504. Each of the sections retain a similar ratio, with the .data section having the most likelihood of having packed elements with an entropy of 6.79, a raw size of 63488 bytes, and a virtual size of 98372 bytes. Still, there are not very solid indicators of the program being packed.

A tool that can be utilized to detect different packer algorithms used is PEiD. When running PEiD on sample2.exe, the following results:



No packing algorithm usage is detected in the sample.

| pFile | Data | Description | Value |
|----------|-------------|-------------------------|--------------------------------|
| 00000208 | 2E 64 61 74 | Name | .data |
| 0000020C | 61 00 00 00 | | |
| 00000210 | 00018044 | Virtual Size | |
| 00000214 | 000E1000 | RVA | |
| 00000218 | 0000F800 | Size of Raw Data | |
| 0000021C | 000E0400 | Pointer to Raw Data | |
| 00000220 | 00000000 | Pointer to Relocations | |
| 00000224 | 00000000 | Pointer to Line Numbers | |
| 00000228 | 0000 | Number of Relocations | |
| 0000022A | 0000 | Number of Line Numbers | |
| 0000022C | C0000040 | Characteristics | |
| | 00000040 | | IMAGE_SCN_CNT_INITIALIZED_DATA |
| | 40000000 | | IMAGE_SCN_MEM_READ |
| | 80000000 | | IMAGE_SCN_MEM_WRITE |

The program has 3 main sections: .text, .data, and .idata. The section .text likely is what contains most of the executable code. The .text section often is where execution will begin, and contains pointers to an import table. The .data section is where global program data is. The .idata section stores import function information, like “CheckTokenMembership”, “CreateWellKnownSid”, “APVAPI32.dll”, and more.

This program obviously has many suspicious strings, with these listed being some of the more notable:

URL’S:

- <https://tox.chat/download.html>
- <http://lockbitapt6vx57t3eeqjofwgcglnutr3a35nygvokja5uuccip4ykyd.onion>
- <https://bigblog.at>

Note the presence of an .onion urls. Only accessible via a TOR compatible browser, utilizing highly encrypted network traffic for anonymity.

Processes

Here we find some of the notable imports found in the .idata section:

- SHChangeNotify
 - Turns off notifications to the system of the shell modifying the system.
- CheckTokenMembership
 - Checks whether a SID (security identifier) is enabled in an access token.
- CreateWellKnownSID
 - Creates a SID for a predefined alias
- CoSetProxyBlanket
 - Sets authentication information used to make calls on a proxy.
- FreeADsMem

- Free memory previously allocated.
- ADsOpenObject
 - Binds to an ADSI object (Active Directory Service Interface) using an explicitly defined username and password.

Filenames / Strings

- LockBit_Ransomware.hta (HTML application file)
- RESTORE-MY-FILES.TXT
- "Found volume %s on %s"
- "Volume %s mounted to %s"
- GPT.INI (initialization for GPT style partitioning)
- LDAP://rootDSE (this is the top of a directory information tree, and is a special entry that contains access to all directories)
- \BaseNamedObjects

Within Software\Policies\Microsoft\Windows Defender:

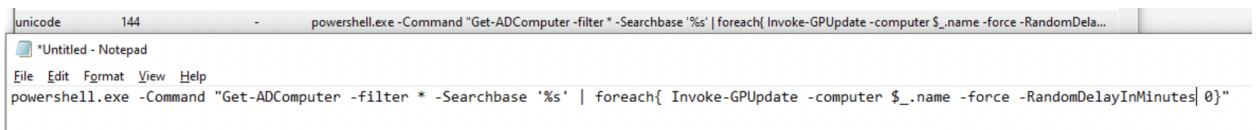
Here we find the malware tampering with settings within Windows Defender.

- ;DisableAntiSpyware
- ;DisableRealtimeMonitoring
- ;SubmitSamplesConsent
- ;Notification_Suppress

Within C:\Windows\system32

- Mshta.exe (used to execute HTA or HTML files, can bypass security)
- Taskkill.exe (used to kill tasks)

Within Group Policy Settings:



- This shell command is essentially grabbing the current user via -Get ADComputer, and forcing an update to Group Policy security settings by setting the randomized update delay on -GPUUpdate to 0. The new security settings will be constantly updated, and therefore enforced. Group Policy will affect all computers on a network.

Dynamic Analysis

Execution

```
datetime:2023/3/17 01:41:42 , 2023/3/17 01:43:47
Computer: MALWARE-2665B12 , MALWARE-2665B12
Username: Administrator , Administrator

-----
Keys added:1
-----
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\FB5EDCE4E87552

-----
Values added:6
-----
HKLM\SYSTEM\ControlSet001\Control\Session Manager\PendingFileRenameOperations: 5C 3F 3F 5C 43 3A 5C 44 6F 63 75 6D 65 6E 74 73 20 61 6E
HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\PendingFileRenameOperations: 5C 3F 3F 5C 43 3A 5C 44 6F 63 75 6D 65 6E 74 73 20 6
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\ShellNoRoam\MUICache\C:\Documents and Settings\Administrator
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\FB5EDCE4E87552\Private: 91 0D B8 04 A6 FB D9 E8 4F E8 77 3A F4 72 51 25 EB A6
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\FB5EDCE4E87552\Public: 72 57 32 1F CF 8E DA 10 F2 6B 46 3E 2D DB BE E3 66 5C B

-----
Values modified:5
-----
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 66 A8 85 5B 8E A8 12 5F 29 B7 2F 2C 11 E4 57 0D 4B AE 41 92 DB 79 FA 0A 5E BB 88 DB 26 A
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed: A5 06 79 F0 BD 62 03 2B C6 1D 63 68 F6 B8 43 B5 B8 98 BB 79 9C 3E 3C CB 95 71 10 17 20 C
HKU\S-1-5-21-117609710-1078145449-725345543-500\Control Panel\Desktop\Wallpaper: "C:\WINDOWS\web\wallpaper\Bliss.bmp"
HKU\S-1-5-21-117609710-1078145449-725345543-500\Control Panel\Desktop\Wallpaper: "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\3.tmp.bmp"
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-
HKU\S-1-5-21-117609710-1078145449-725345543-500\SessionInformation\ProgramCount: 0x00000002
HKU\S-1-5-21-117609710-1078145449-725345543-500\SessionInformation\ProgramCount: 0x00000001

-----
Files added:8139
-----
C:\Documents and Settings\Administrator\Cookies\administrator@downloads.sourceforge[2].txt.lockbit
C:\Documents and Settings\Administrator\Cookies\Restore-My-Files.txt
C:\Documents and Settings\Administrator\Desktop\dynamic Analysis\debugger_x86\1394\1394dbg.cat.lockbit
C:\Documents and Settings\Administrator\Desktop\dynamic Analysis\debugger_x86\1394\1394dbg.inf.lockbit
C:\Documents and Settings\Administrator\Desktop\dynamic Analysis\debugger_x86\1394\Restore-My-Files.txt
C:\Documents and Settings\Administrator\Desktop\dynamic Analysis\debugger_x86\sdk\help\dbghelp.chm.lockbit
C:\Documents and Settings\Administrator\Desktop\dynamic Analysis\debugger_x86\sdk\help\Restore-My-Files.txt
C:\Documents and Settings\Administrator\Desktop\dynamic Analysis\debugger_x86\sdk\inc\dbgeng.h.lockbit

-----
Files [attributes?] modified:6
-----
C:\Documents and Settings\Administrator\ntuser.dat.LOG
C:\WINDOWS\Prefetch\CMD.EXE-087B4001.pf
C:\WINDOWS\Prefetch\PING.EXE-31216D26.pf
C:\WINDOWS\system32\config\software.LOG
C:\WINDOWS\system32\config\system
C:\WINDOWS\system32\config\system.LOG

-----
Total changes:15805
```

After taking a snapshot of the registry via RegShot, running the malware, and then taking a second snapshot, it becomes even more clear what the malware is doing. It adds one registry key, which appears to follow the format of a new SID (security identifier). It added 6 values, two of which relate to file rename operations, likely associated with adding the .lockbit extension to files. All of these added values are added under the unique SID key in the “Keys added” section.

5 values are recorded to be modified in the “Values modified” section. Two cryptography keys are modified, likely related to the encryption algorithm associated with the .lockbit extension. The desktop wallpaper is modified. File explorer is also

modified 4 times through UserAssist settings. This key contains settings and data of programs launched through explorer.

Within the “Files added” and “Files deleted” sections, we can see the obvious “Restore-My-Files.txt” which appears on the desktop. Additionally, many if not all of the system's files have been deleted and replaced with a .lockbit extension, and are unable to be opened. A “Restore-My-Files.txt” file is added to most directories affected. Several file attributes have been modified as well, including a modification to ntuser.dat, which contains a user’s profile settings.

In ProcMon, we can see the immediate effects of the Group Policy modification that was uncovered during static analysis. Security features of Windows have been disabled because they are updated continuously to be turned off.

| Time of Day | Process Name | PID | Operation | Path | Result | Detail |
|--------------------|--------------|-----|---------------|-----------------------------------|------------------|----------------------|
| 9:24:15.9250854 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy | SUCCESS | Desired Access: R... |
| 9:24:15.9251055 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: R... |
| 9:24:15.9251231 PM | lsass.exe | 708 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\D... | BUFFER OVERFL... | Length: 12 |
| 9:24:15.9251558 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 9:24:15.9251673 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: R... |
| 9:24:15.9251829 PM | lsass.exe | 708 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\D... | SUCCESS | Type: REG_NONE... |
| 9:24:15.9252215 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 9:24:15.9254402 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy | SUCCESS | |
| 9:24:15.9256576 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy | SUCCESS | Desired Access: R... |
| 9:24:15.9256746 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: R... |
| 9:24:15.9256900 PM | lsass.exe | 708 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\D... | BUFFER OVERFL... | Length: 12 |
| 9:24:15.9257210 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 9:24:15.9257408 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: R... |
| 9:24:15.9257556 PM | lsass.exe | 708 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\D... | SUCCESS | Type: REG_NONE... |
| 9:24:15.9257863 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 9:24:15.9259699 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy | SUCCESS | |
| 9:24:15.9265012 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy | SUCCESS | Desired Access: R... |
| 9:24:15.9265186 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: R... |
| 9:24:15.9265336 PM | lsass.exe | 708 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\D... | BUFFER OVERFL... | Length: 12 |
| 9:24:15.9265644 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 9:24:15.9265750 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: R... |
| 9:24:15.9265901 PM | lsass.exe | 708 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\D... | SUCCESS | Type: REG_NONE... |
| 9:24:15.9266205 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 9:24:15.9268030 PM | lsass.exe | 708 | RegCloseKey | HKLM\SECURITY\Policy | SUCCESS | |
| 9:24:15.9270041 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy | SUCCESS | Desired Access: R... |
| 9:24:15.9270223 PM | lsass.exe | 708 | RegOpenKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: R... |
| 9:24:15.9270376 PM | lsass.exe | 708 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\D... | BUFFER OVERFL... | Length: 12 |

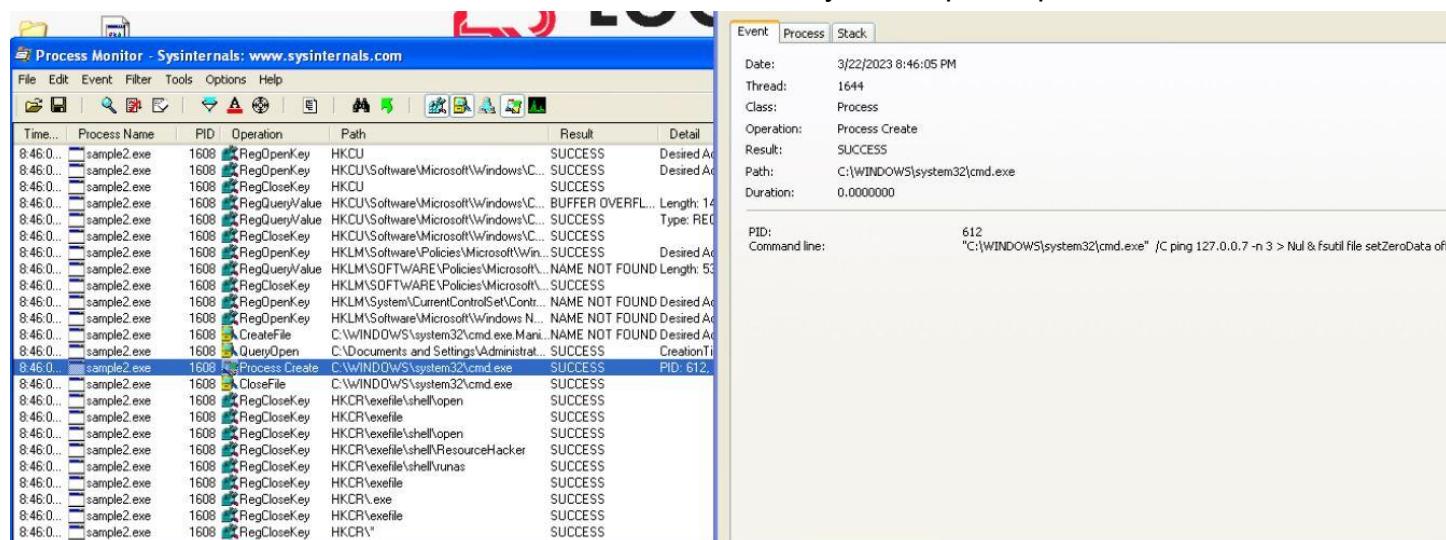
The LSA Shell command shown in the static analysis section is being called over and over again. The malware is highly persistent, and does not allow the user to attempt to shut down lockbit processes. Restarting the system proves ineffective.

| Process Tree | | | | | |
|---|-----------------------|----------------------|-----------|-----------------------|--|
| <input type="checkbox"/> Only show processes still running at end of current trace <input checked="" type="checkbox"/> Timelines cover displayed events only | | | | | |
| Process | Description | Image Path | Life Time | Company | Command |
| Idle (0) | Idle | | | | |
| System (4) | System | | | | |
| smss.exe (376) | Windows NT Ses... | C:\WINDOWS\S... | | Microsoft Corporat... | \SystemRoot\System32\smss.exe |
| csrss.exe (592) | Client Server Runt... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\csrss.exe ObjectDirectory=\Windows SharedSecti |
| winlogon.exe (616) | Windows NT Log... | C:\WINDOWS\sy... | | Microsoft Corporat... | winlogon.exe |
| services.exe (636) | Services and Cont... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\services.exe |
| vmacthlp.exe (864) | VMware Activatio... | C:\Program Files\... | | VMware, Inc. | "C:\Program Files\VMware\VMware Tools\vmacthlp.exe" |
| svchost.exe (880) | Generic Host Proc... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\svchost.exe -k DcomLaunch |
| wmiprvse.exe (59) | WMI | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\wbem\wmiprvse.exe |
| svchost.exe (960) | Generic Host Proc... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\svchost.exe -k rpcss |
| svchost.exe (1044) | Generic Host Proc... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\svchost.exe -k netsvcs |
| wscntfy.exe (124) | Windows Security... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\wscntfy.exe |
| wuauclt.exe (236) | Automatic Updates | C:\WINDOWS\sy... | | Microsoft Corporat... | "C:\WINDOWS\system32\wuauclt.exe" |
| svchost.exe (1104) | Generic Host Proc... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\svchost.exe -k NetworkService |
| svchost.exe (1172) | Generic Host Proc... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\svchost.exe -k LocalService |
| spoolsv.exe (1388) | Spooler SubSyste... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\spoolsv.exe |
| svchost.exe (1872) | Generic Host Proc... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\svchost.exe -k bthsvcs |
| VGAuthService.exe | VMware Guest Au... | C:\Program Files\... | | VMware, Inc. | "C:\Program Files\VMware\VMware Tools\VMware VGAuthService |
| vmtoolsd.exe (384) | VMware Tools Cor... | C:\Program Files\... | | VMware, Inc. | "C:\Program Files\VMware\VMware Tools\vmtoolsd.exe" |
| lsass.exe (1504) | Authentication L... | C:\WINDOWS\sy... | | Microsoft Corporat... | C:\WINDOWS\system32\lsass.exe |

Here we see a list of processes in use through ProcMon. ProcMon is killed in the Windows 10 / 7 VM upon launching the malware.

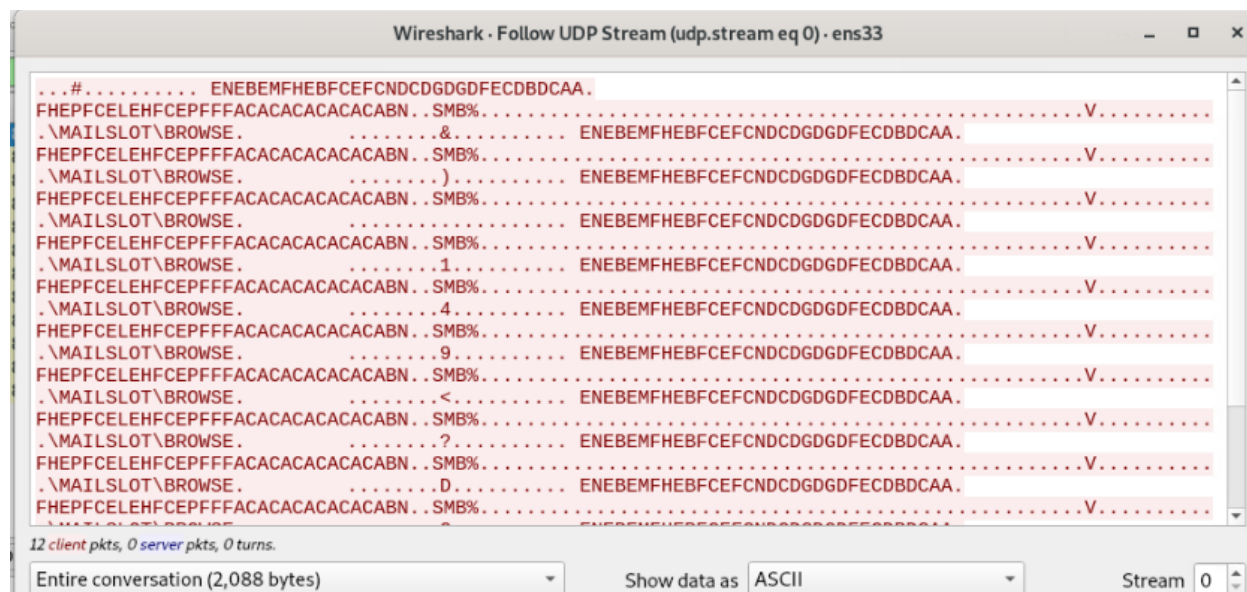
Applying some filters, we can see that sample2.exe is querying and mapping files, as well as applying changes to gpedit.dll, the group policy security settings editor. Here we find the malware starting an interesting process - a command line argument is passed in. It appears to be a sort of cleanup with a ping delay. Note the new user that it has

created - MALWARE-2665B12\Administrator. This certainly will help with persistence.

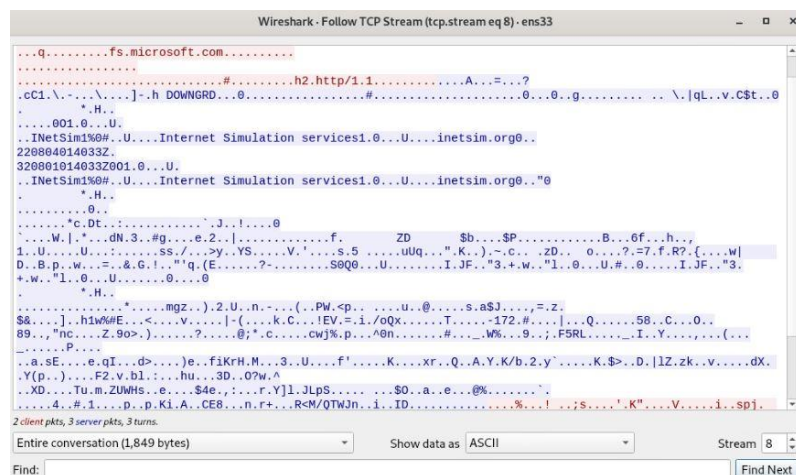


Running the malware on the different versions of windows did produce some different results. In all cases, the malware changes the desktop background, and encrypts most files with .lockbit. On windows XP, files remain on the desktop, and it is possible to monitor the processes with ProcMon. Windows 7 and 10 clear the desktop of files, and files seem to be encrypted more consistently, and programs are killed via Taskkill.exe.

Network

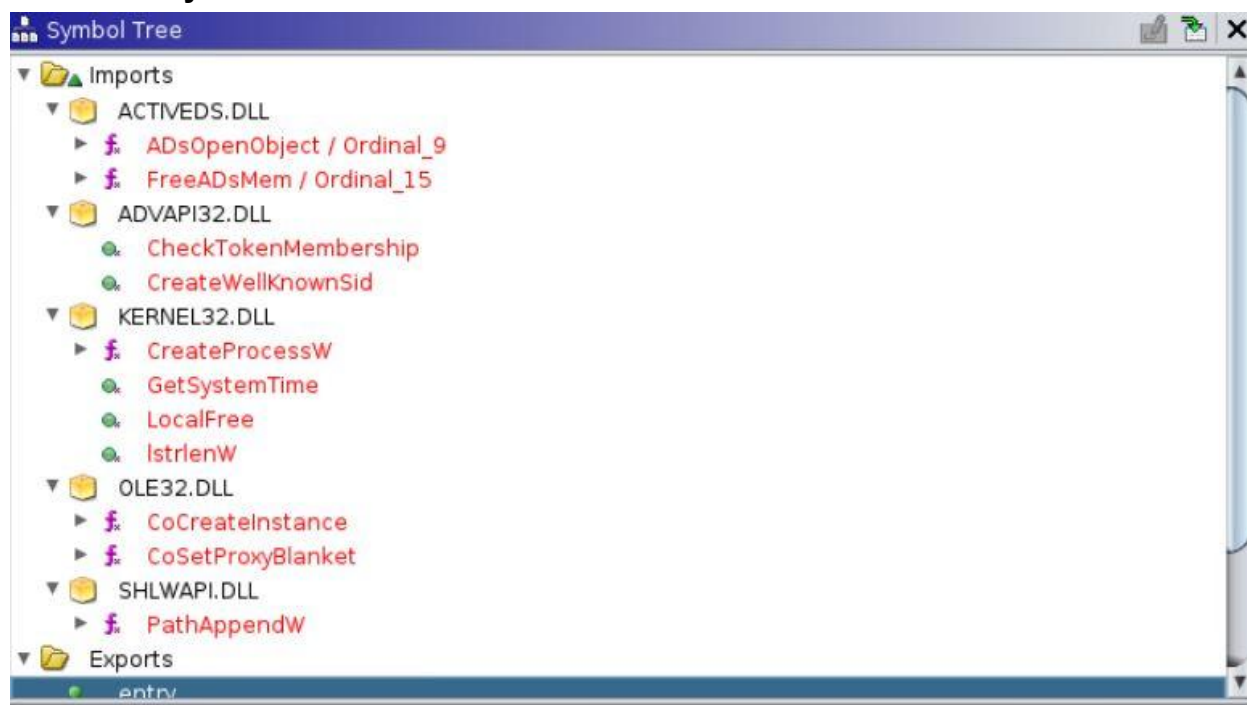


On Wireshark, the traffic involved with the malware seems to be very encrypted. Here the browser client is requesting the local master browser server for backup browser servers. It appears to be using mailslots as a method of interprocess communication. The owner of the mailslot will be able to retrieve whatever has been stored there. This occurs several times while the malware is running. However, Wireshark and fakedns analysis has not proven particularly fruitful because of the encryptions and obfuscation that the malware performs.



Traffic is very encrypted.

Disassembly



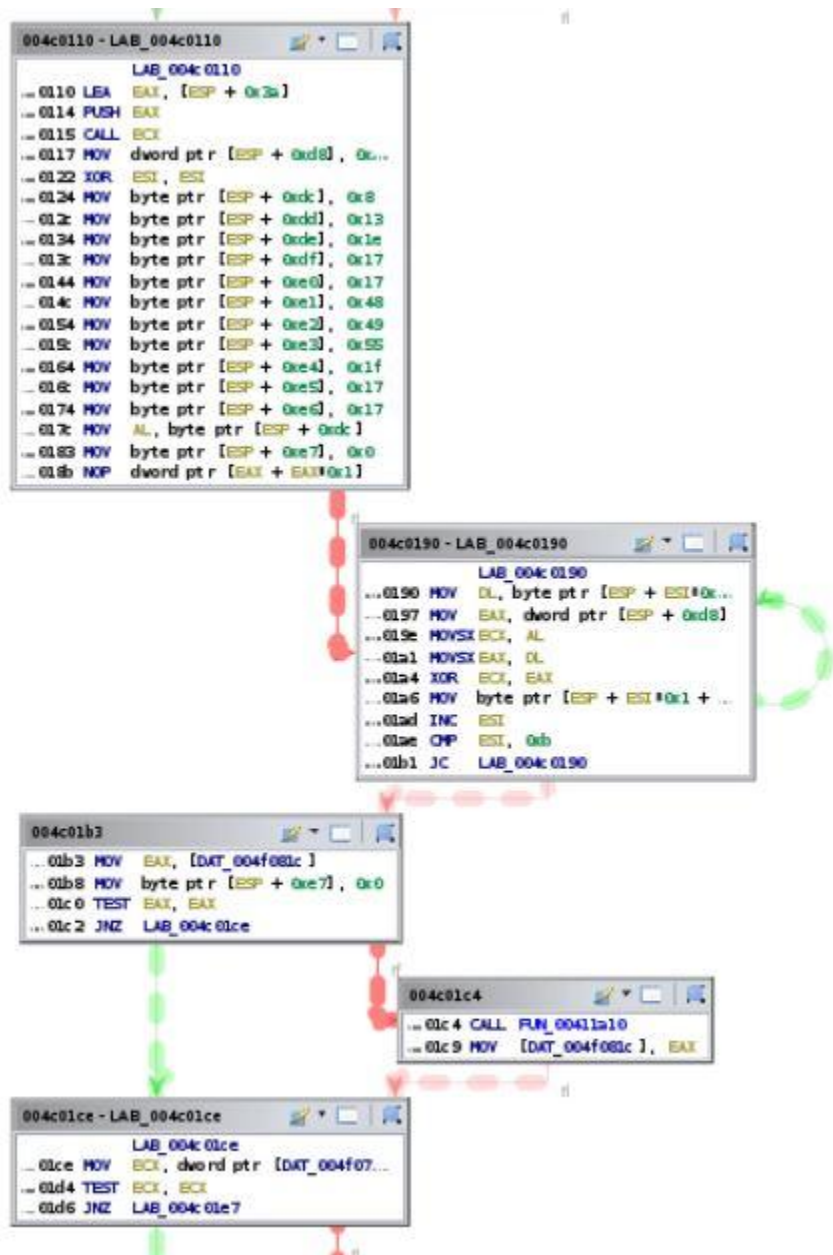
Upon loading the malware into Ghidra, we can see several DLL imports and their utilized functions.

Taking a look at the strings in Ghidra, there was one of interest that had not been caught during static analysis:

| | | | |
|----------|---------------------------------|---------------------------------|---------|
| 004034cc | SHChangeNotify | "SHChangeNotify" | ds |
| 004034e8 | /C ping 127.0.0.7 -n 3 > Nul... | u" /C ping 127.0.0.7 -n 3 > ... | unicode |
| 004035c8 | {%02X%02X%02X%02X-%02... | u" {%02X%02X%02X%02X-%... | unicode |
| 00403668 | SOFTWARE\\%02X%02X%02X... | u"SOFTWARE\\%02X%02X%0... | unicode |

"/C ping 127.0.0.7 -n 3 > Nul & fsutil file setZeroData offset=0 length=524288 "%s" & Del /f /q "%s"

This string seems to indicate some sort of deletion operation within the malware. This may prove useful in creating an effective Yara rule. The ping command suggests a delay, and the first 524288 bytes are set to 0.



The malware's function graph seems to indicate that the malware copies and moves data via EDI, ESI, ESP, ECX, EAX, AL and offsets. There is a clear pattern in the function graph: Move data onto ESP register, XOR operations (possibly for encryption), increment ESI, repeat.

As far as the encryption utilized by the .lockbit extension goes, this function seems to be the process:

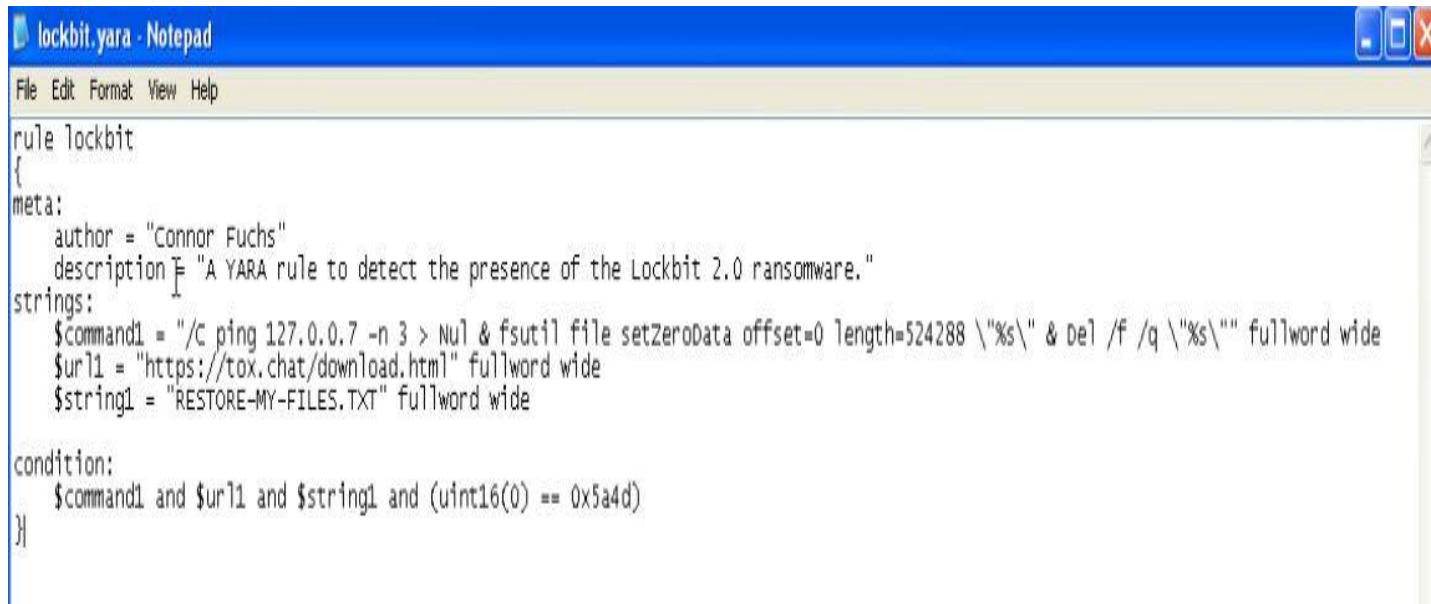
```
Decompile: FUN_00411a10 - (sample2.exe)

14 piVar2 = **(int **)(*(int *)((int)ProcessEnvironmentBlock + 0xc) + 0xc);
15 piVar7 = piVar2;
16 do {
17     pcVar3 = (char *)piVar7[0xc];
18     uVar6 = 0;
19     uVar5 = (uint)(*(ushort *) (piVar7 + 0xb) >> 1);
20     local_8 = 0x811c9dc5;
21     if (pcVar3 + uVar5 * 2 < pcVar3) {
22         uVar5 = 0;
23     }
24     if (uVar5 != 0) {
25         do {
26             cVar1 = *pcVar3;
27             pcVar3 = pcVar3 + 2;
28             uVar4 = (int)cVar1 | 0x20;
29             if (0x19 < (byte)(cVar1 + 0xbfU)) {
30                 uVar4 = (int)cVar1;
31             }
32             uVar6 = uVar6 + 1;
33             local_8 = (uVar4 ^ local_8) * 0x1000193;
34         } while (uVar6 != uVar5);
35         if (local_8 == 0xa3e6f6c3) {
36             return piVar7[6];
37         }
38     }
39     piVar7 = (int *)*piVar7;
40     if ((piVar7 == piVar2) || (piVar7[6] == 0)) {
41         return 0;
42     }
43 } while( true );
44 }
```

The layout of the algorithm and presence of | and ^ operators suggest that an XOR cipher could be utilized, although this is not a definite assertion. Note the use of the Process Environment Block. Certain members of the block are copied as well as modified.

Indicators of Compromise

There are several specific indicators of compromise within the malware to create an effective Yara rule. Utilizing the string `/C ping 127.0.0.7 -n 3 > Nul & fsutil file setZeroData offset=0 length=524288 "%s" & Del /f /q "%s"` uncovered during analysis with Ghidra proved to be the most simple and effective rule for not creating false positives, along with some of the URL's and strings found during static analysis for extra specificity. Additionally, the inclusion of the first header value acquired from HxD adds another layer of specificity.



```
lockbit.yara - Notepad
File Edit Format View Help

rule lockbit
{
  meta:
    author = "Connor Fuchs"
    description = "A YARA rule to detect the presence of the Lockbit 2.0 ransomware."
  strings:
    $command1 = "/C ping 127.0.0.7 -n 3 > Nul & fsutil file setZeroData offset=0 length=524288 \"%s\" & Del /f /q \"%s\"" fullword wide
    $url1 = "https://tox.chat/download.html" fullword wide
    $string1 = "RESTORE-MY-FILES.TXT" fullword wide
  condition:
    $command1 and $url1 and $string1 and (uint16(0) == 0x5a4d)
}
```

Here is the YARA rule, and the results when recursively searching the “Binary Collection” directory containing many different malware samples, to ensure no false positives exist.

```

C:\Documents and Settings\Administrator\Desktop>
C:\Documents and Settings\Administrator\Desktop>
C:\Documents and Settings\Administrator\Desktop>yara32 -s -r lockbit.yara "C:\Do
cuments and Settings\Administrator\Desktop\Practical Malware Analysis Labs\Binary
Collection"
lockbit C:\Documents and Settings\Administrator\Desktop\Practical Malware Analys
is Labs\BinaryCollection\sample2.exe
0x28ea:$command1: ^\x00C\x00 \x00p\x00i\x00n\x00g\x00 \x001\x002\x007\x00.\x000\
\x00.\x000\x00.\x007\x00 \x00-\x00n\x00 \x003\x00 \x00>\x00 \x00N\x00u\x001\x00 \
\x00&\x00 \x00f\x00s\x00u\x00t\x00i\x001\x00 \x00f\x00i\x001\x00e\x00 \x00s\x00e\
\x00t\x00Z\x00e\x00r\x00o\x00D\x00a\x00t\x00a\x00 \x00o\x00f\x00f\x00s\x00e\x00t\
\x00=\x000\x00 \x001\x00e\x00n\x00g\x00t\x00h\x00=\x005\x002\x004\x002\x008\x008\
\x00 \x00"\x00%\x00s\x00"\x00 \x00&\x00 \x00D\x00e\x001\x00 \x00/\x00f\x00 \x00/\
\x00g\x00 \x00"\x00%\x00s\x00"\x00
0x37bc:$url1: h\x00t\x00t\x00p\x00s\x00:\x00/\x00/\x00t\x00o\x00x\x00.\x00c\x00h
\x00a\x00t\x00/\x00d\x00o\x00w\x00n\x001\x00o\x00a\x00d\x00.\x00h\x00t\x00m\x001
\x00
0x32c4:$string1: R\x00E\x00S\x00T\x00O\x00R\x00E\x00-\x00M\x00Y\x00-\x00F\x00I\x
00L\x00E\x00S\x00.\x00T\x00X\x00T\x00
C:\Documents and Settings\Administrator\Desktop>yara32 -r lockbit.yara "C:\Docum
ents and Settings\Administrator\Desktop\Practical Malware Analysis Labs\BinaryCo
llection"
lockbit C:\Documents and Settings\Administrator\Desktop\Practical Malware Analys
is Labs\BinaryCollection\sample2.exe
C:\Documents and Settings\Administrator\Desktop>

```

The YARA rule has been run twice here: one time to show the found strings with -s, and one time showing the directory of the detected sample2.exe.