MBSD Blog

Analyzing “Ragnar Locker” ransomware that threats a company by its name

Takashi Yoshikawa

Senior Malware Analyst, Cyber Intelligence Group

Mitsu Bussan Secure Directions, Inc.

Kei Sugawara

Senior Malware Analyst, Cyber Intelligence Group

Mitsu Bussan Secure Directions, Inc.

November 2020

Table of Contents

1. Introduction

In November 2020, CAPCOM's cyber-attack news widely spread out primarily in foreign media. According to the news, the cyber-attack involved ransomware called Ragnar Locker. After the incident happened, Ragnar Locker attack group actually released a criminal statement on November 9, 2020.

Based on the publicly available data, we investigated and found a suspected specimen on VirusTotal.

This article describes the results of the analysis of the relevant specimen.

Please note that after we discovered the specimen on VirusTotal, we contacted a person who supplied information to publications such as Bleeping Computer. The person has confirmed the specimen is identical to the one available on the VirusTotal. At the same time however, we didn't verify that the specimen was actually used in CAPCOM cyber-attack. Therefore, the content of this article only describes the result of my analysis of the same sample as Ragnar Locker mentioned in the publication.

Hash value: 6f559fd57304197443b71d8bf553cce3c9de8d53

2. Specimen

The relevant sample was submitted from Japan to VirusTotal on November 4, 2020, and only one sample was submitted at the time of this analysis.

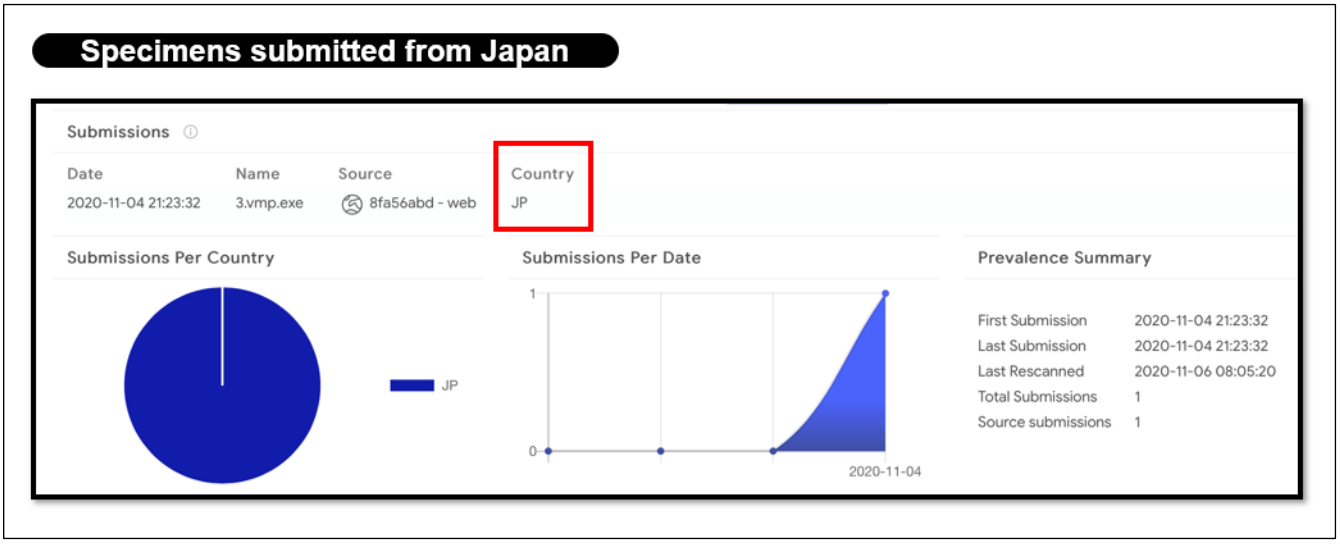

Fig. 1 - Ragnar Locker specimen uploaded to VirusTotal from Japan

The original file name of the file is uploaded with the name "3.vmp.exe", and the file icon does not exist as shown in the following figure.

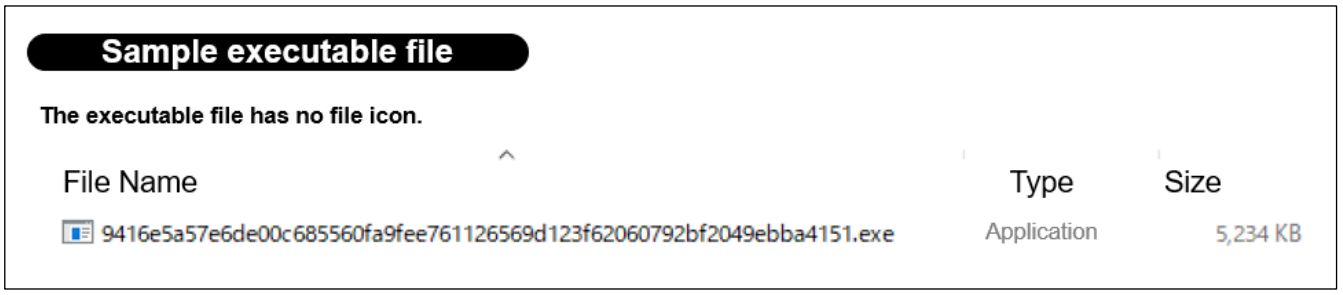

Fig. 2 - Executable file of Ragnar Locker does not have an icon

Ragnar Locker executable file for this specimen has a valid digital signature, as shown in the following figure, and the time stamp of the digital signature is November 1, 2020. The official announcement of CAPCOM stated that the incident occurred from November 2. If this specimen was actually used for the corresponding cyber-attack, then it may have been signed immediately before the attack.

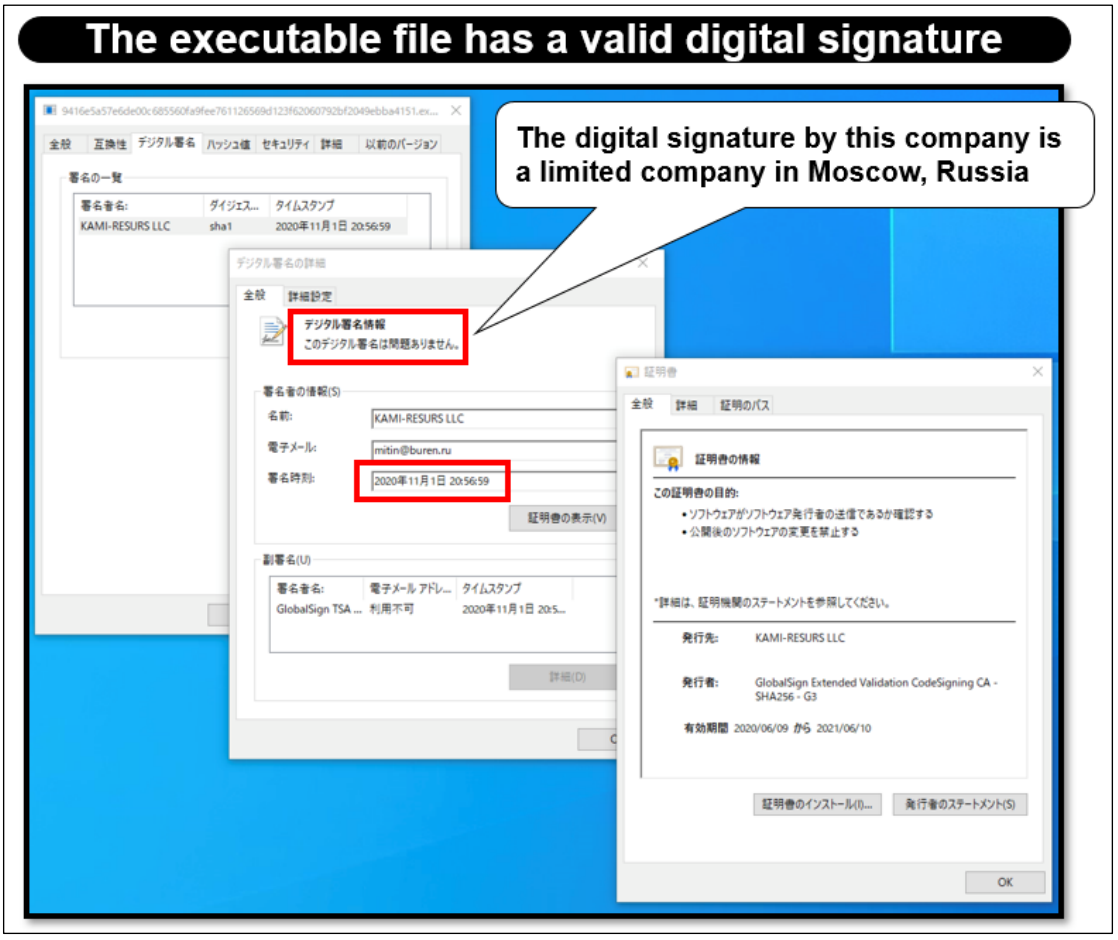

Fig. 3 - A valid digital signature found in the Ragnar Locker executable file

In addition, Time Date Stamp in the PE header, which indicates the compilation date and time of the executable file, is October 20, 2020. This indicates, same as digital signature, the creation of the ransomware itself may have created just before the attack.

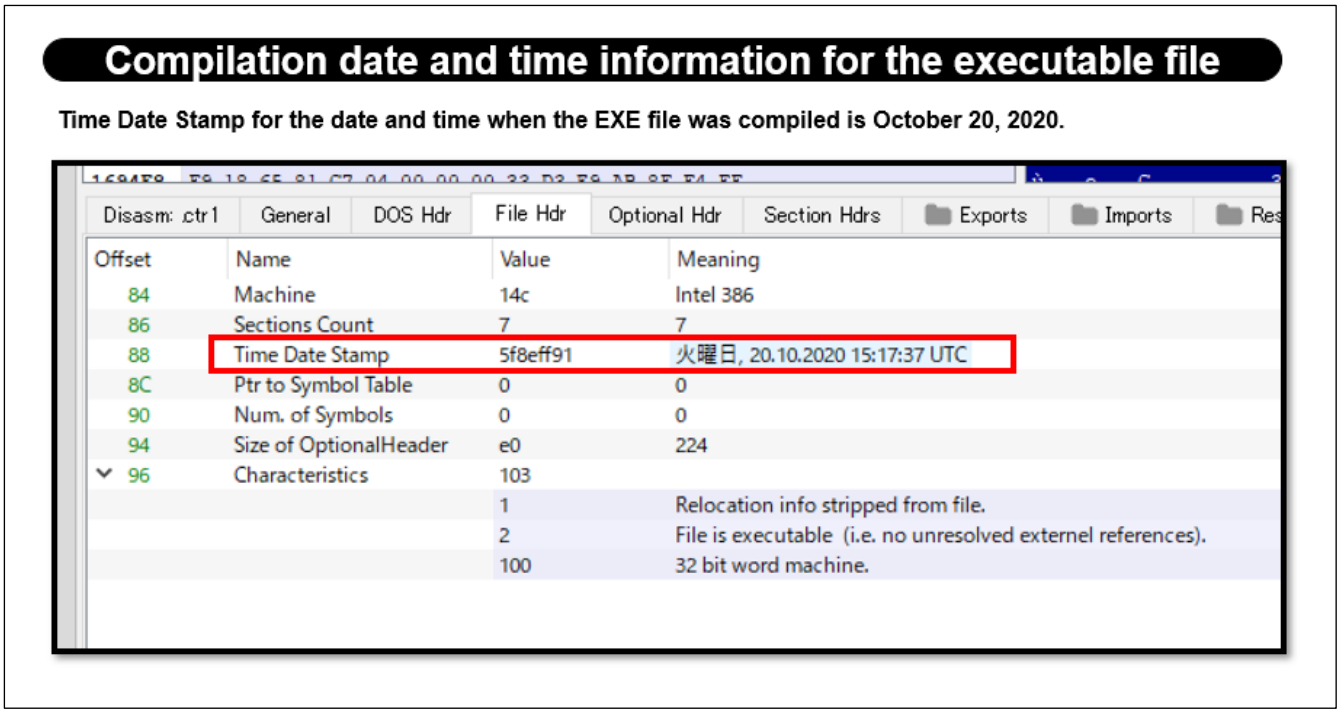

Fig. 4 - Ragnar Locker executable compilation date and time

3. A number of anti-analysis features

Ragnar Locker EXE-file has many analysis prevention features, so it is not easy to analyze.

First, it is packaged by VMProtect, a very robust commercial protector, as follows:

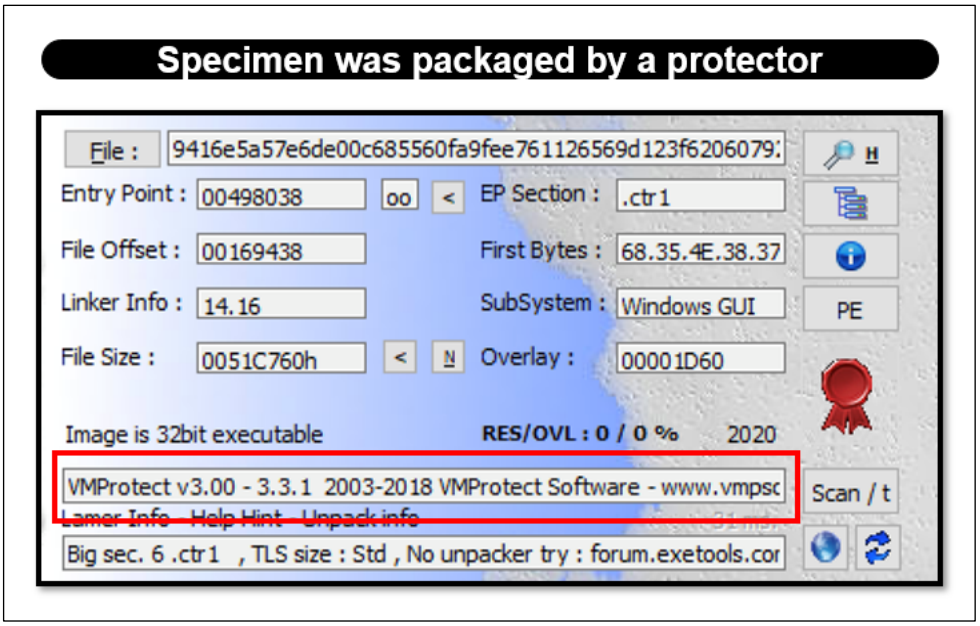

Fig. 5 Analyzed by a surface analysis tool

If you simply try to analyze using a debugger, the packer shows an error message as shown below, and Ragnar Locker executable file is forcibly terminated, and the analysis cannot continue.

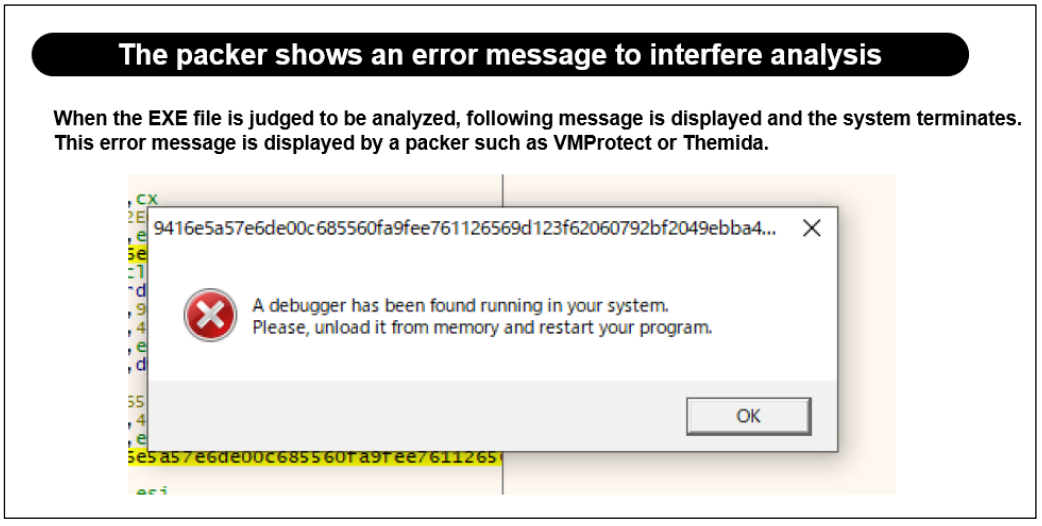

Fig. 6 - Debugger detection message that interfere analysis

In addition, even after the main code of Ragnar Locker is successfully unpacked onto memory, a mechanism that interferes API hook is implemented.

Specifically, the first 5 bytes of the DbgUiRemoteBreakin function of the ntdll.dll (which is a system DLL) is written in order to jump to its own original hook function, so that it is not able to be attached by the debugger (see below figure).

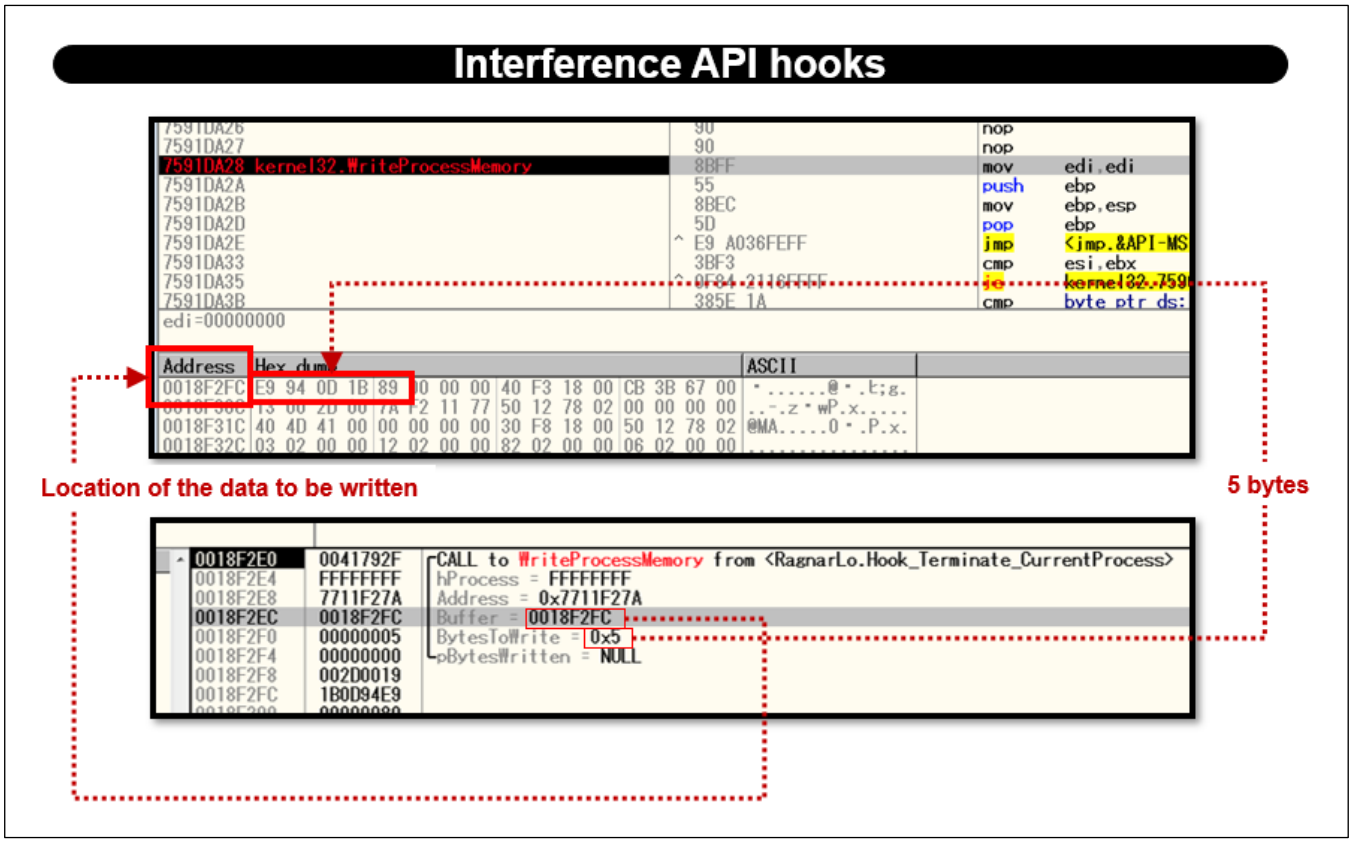

Fig. 7 - Ragnar Locker writes the first 5 bytes of the system DLL

The following illustration shows before-and-after comparison of DbgUiRemoteBreakin function in in ntdll.dll. Before Ragnar Locker writes the first 5 bytes, it shows "6A 08 68 50 BB", which have been tampered to "E9 94 0D 1B 89". This is a jump instruction with destination address to Ragnar Locker.

The reason why Ragnar Locker tampers the first 5 bytes of the DbgUiRemoteBreakin function is that Ragnar Locker does not want to be debugged by DbgUiRemoteBreakin function. Usually debug function attaches to the object process and the DbgUiRemoteBreakin function is called. However, if the first 5 bytes of the DbgUiRemoteBreakin function is tampered, then it will automatically jump to Ragnar Locker's function.

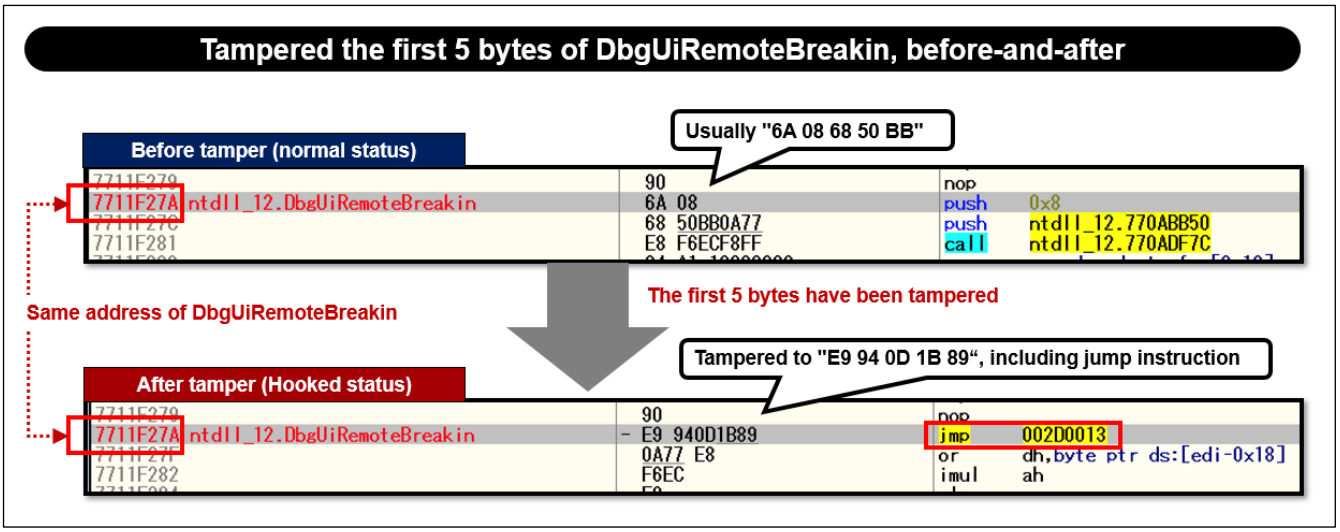

Fig. 8 - Comparing the first 5 bytes of the APIs

The jump destination address, tampered by the hook, instructs jump to another address again and again, then it finally jumps to an address that instructs force termination of its own process (see the figure below). This means that when the analyst attaches the debugger, the DbgUiRemoteBreakin function is called, and Ragnar Locker terminates, resulting that the analyst cannot continue analysis.

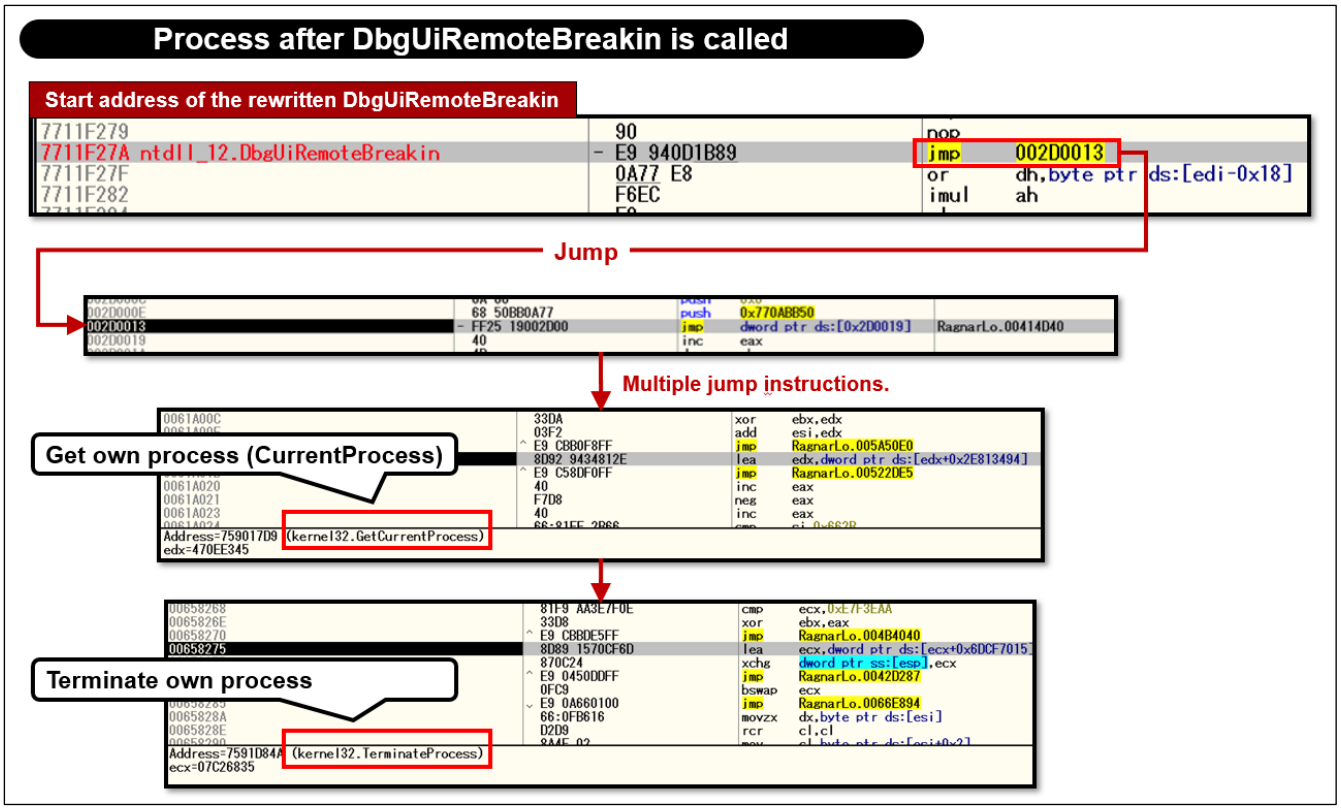

Fig. 9 - Process when DbgUiRemoteBreakin is called

Maze ransomware also focused on DbgUiRemoteBreakin function, but it only changes the first byte to the “return” instruction, so that DbgUiRemoteBreakin immediately returns to the main program, meaning that the DbgUiRemoteBreakin function eventually do nothing. On the other hand, Ragnar Locker tampers the first 5 bytes and ultimately force terminate own process, so I think Ragnar Locker technique is more complicated than Maze ransomware.

Likewise, NtProtectVirtualMemory function is hooked and it jumps to their own code.

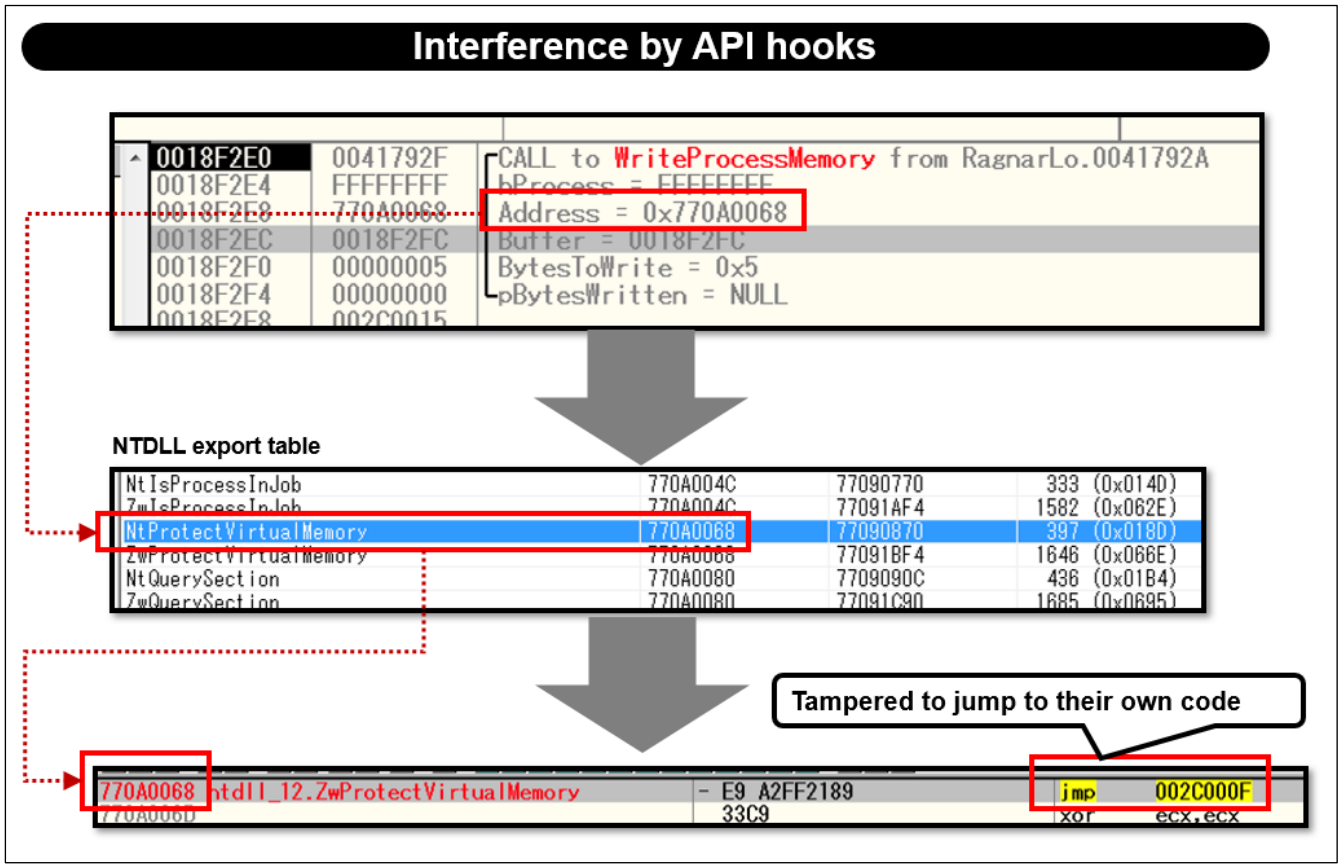

Fig. 10 - NtProtectVirualMemory is also tampered

4. Checking language of infected PC

Ragnar Locker does not infect PCs that is identified Russian and other specific languages. Ragnar Locker obtains language information of the infected PC by using GetLocaleInfo, and if the result matches to one of the pre-defined languages, then it finally terminates its own process.

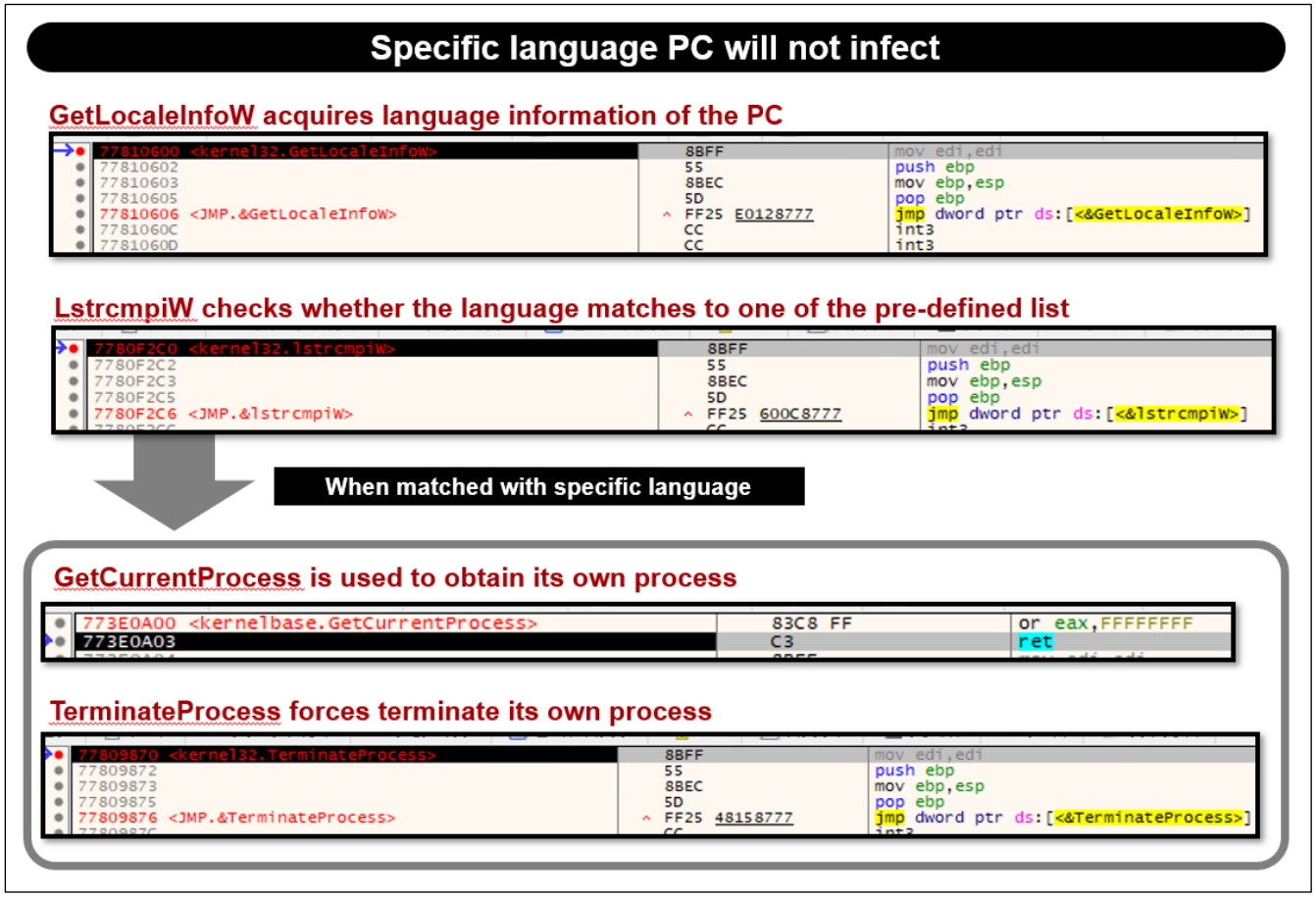

Fig. 11 - Process to check infected PC's language

Below is the list of languages that will NOT infect Ragnar Locker.

● Azerbaijani	● Tajik
● Armenian	● Russian
● Belorussian	● Turkmen
● Kazakh	● Uzbek
● Kyrgyz	● Ukrainian
● Moldavian	● Georgian

5. Interfering with system recovery

Ragnar Locker will then take action that interferes with the recovery of the system.

First, disables “System Restore” by using Windows regular process, wmic, and runs a command to delete volume shadow copy.

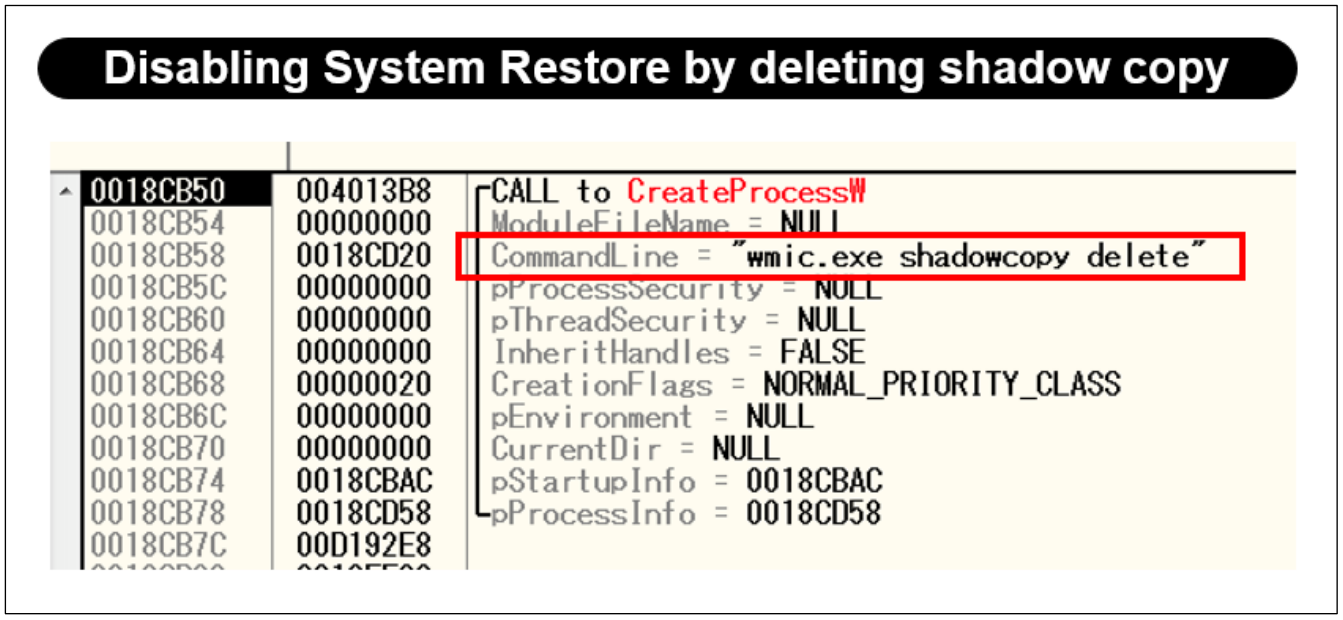

Fig. 12 - Deleting shadow copy

Then disables the ability to automatic repair when Windows fails startup, by passing the "recoveryenabled No" parameter to Bcdedit, which is a Windows' regular process.

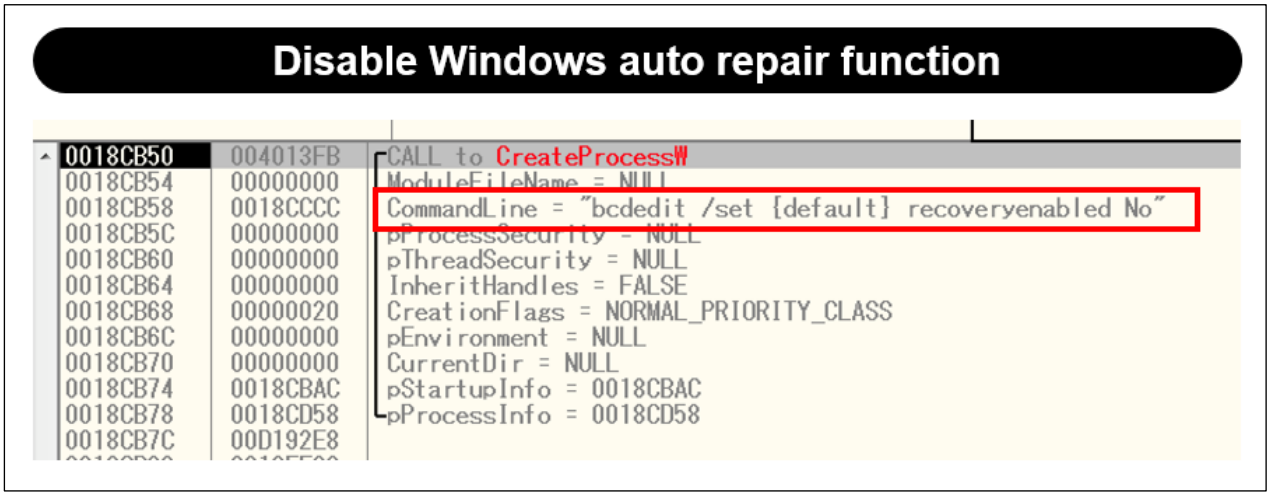

Fig. 13 - Disabling Windows’ automatic repair function (1)

Similarly, Bcdedit (Windows' regular process) to configure a normal startup without repairing Windows startup failures by passing a "bootstatuspolicy IgnoreAllFailures" parameter.

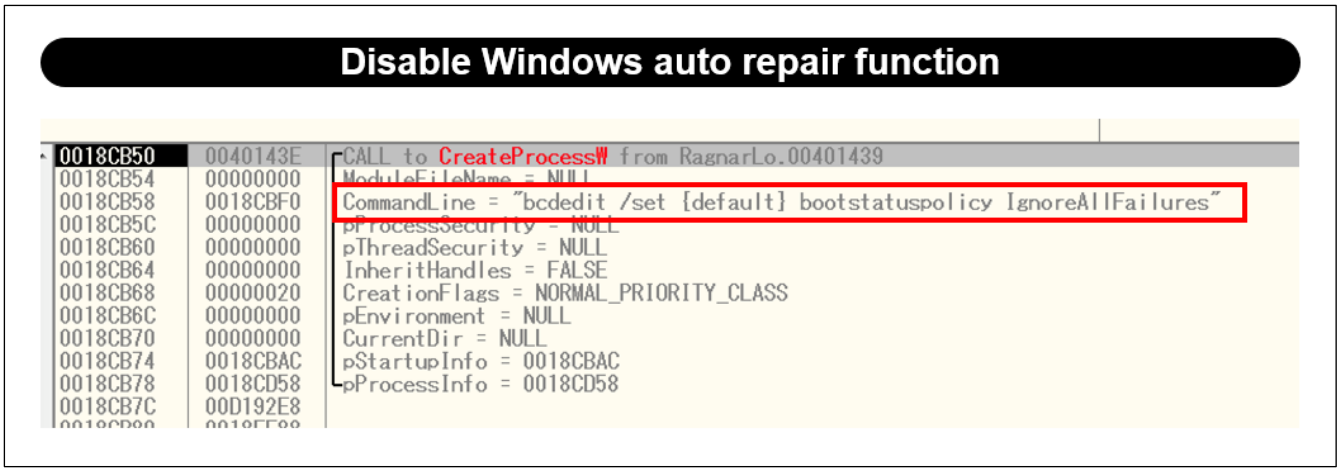

Fig. 14 - Disabling Windows' automatic repair function (2)

In addition, by passing the "advancedoptions false" parameter to Bcdedit, the "Advanced Settings of Startup Options" menu is set to be inaccessible from the F8 key at Windows startup.

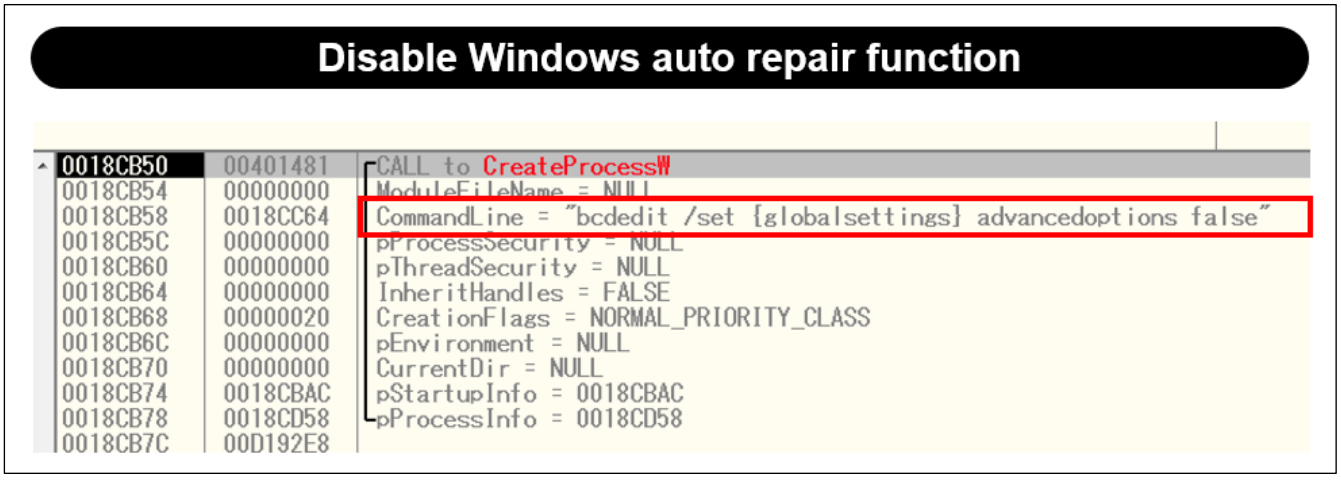

Fig. 15 - Disabling Windows' automatic repair function (3)

As mentioned above, there are some activities that prevent the PC from being repaired when Windows start up. However, it seems Ragnar Locker does not destroy files that requires Windows start up. We guess that the attacker wanted to keep current Windows environment, rather than encrypt start up files and user recovers to unknows environment.

6. Stopping processes and services

Ragnar Locker then checks active processes to see if particular process is running. After Ragnar Locker acquires the list of processes running on the PC, it compares with the list that is hard-coded in the Ragnar Locker. As shown in the figure below, the list includes databases, document software, and mail software that need to be terminated, in order to encrypt files.

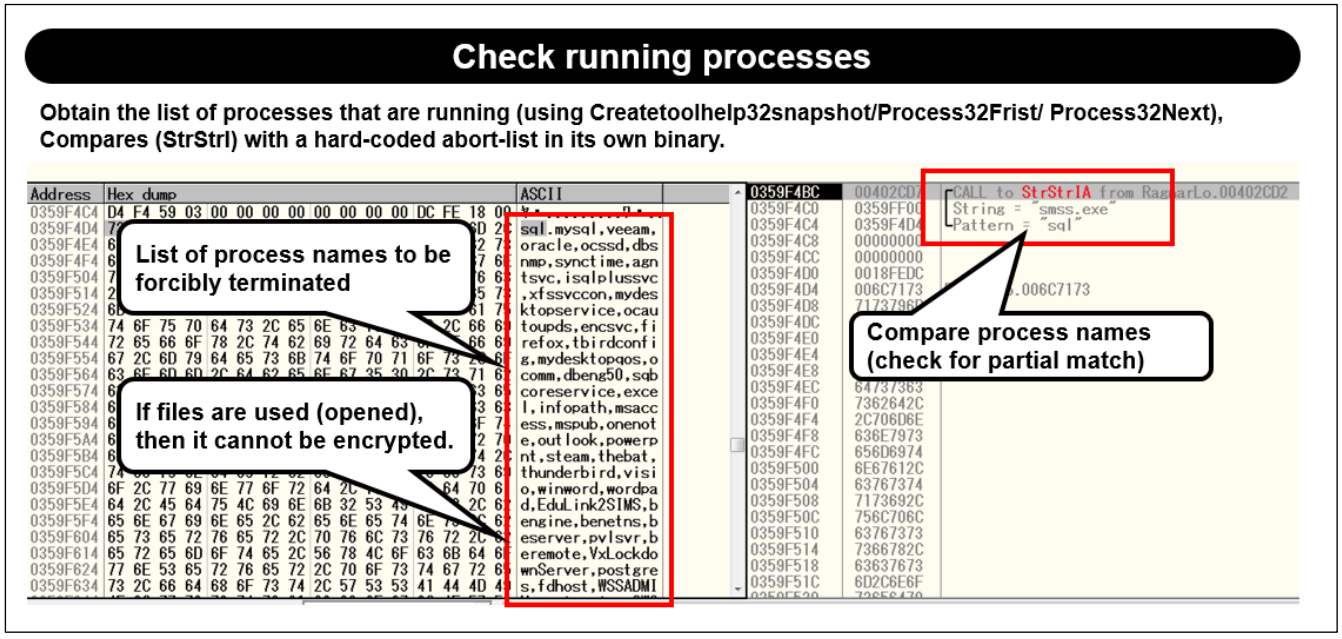

Fig. 16 - Checking if a specific process is running

If it is determined that a matching process is running, the corresponding process is terminated forcibly (see the figure below). If a process keeps running, then the corresponding data files are locked by the process and Ragnar Locker will NOT able to encrypt those data files, so they need to be terminated beforehand.

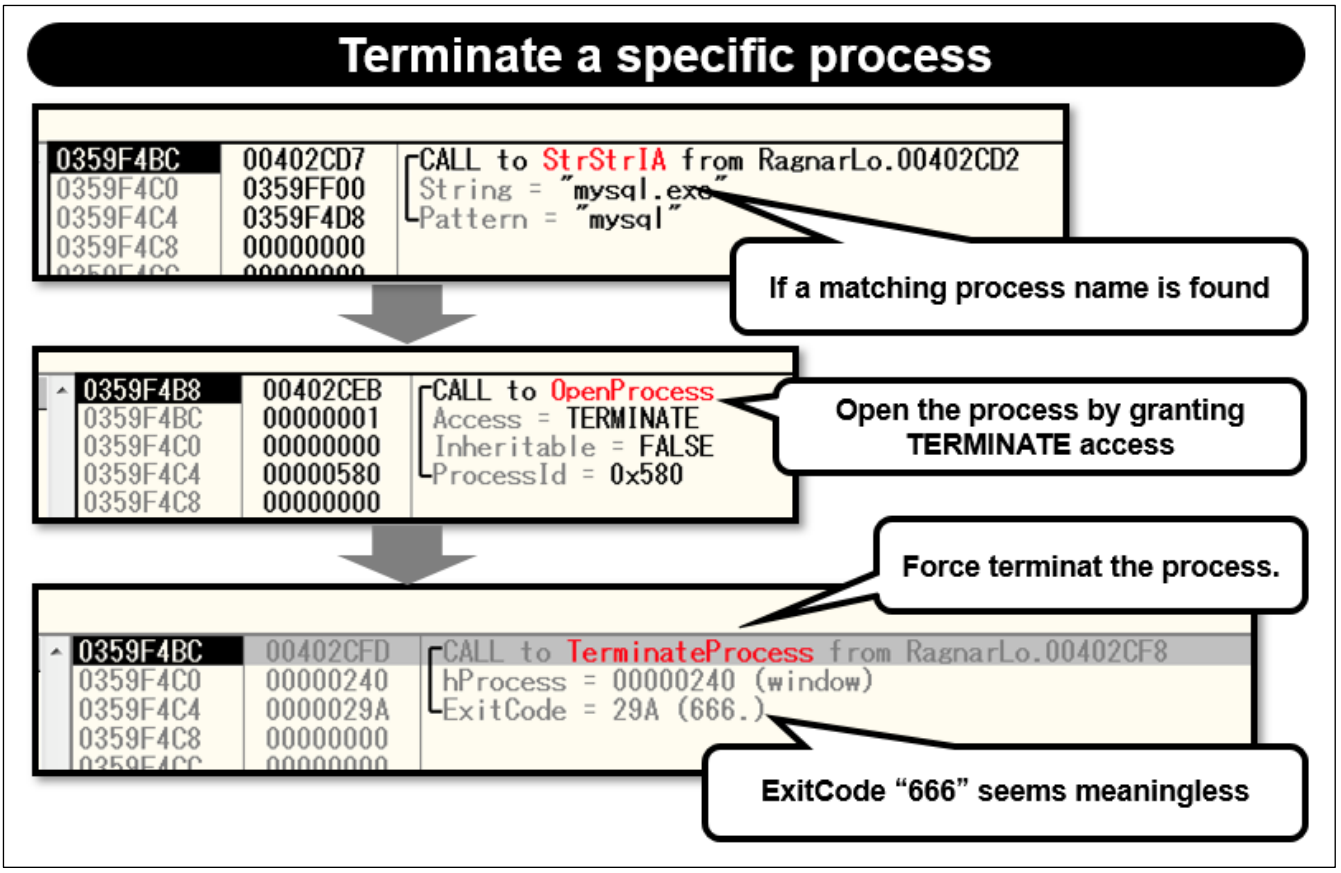

Fig.17 - A process is forcibly terminated if it is running

Following is the list of processes that Ragnar Locker terminates:

sql	msaccess	wsstracing
mysql	mspub	OWSTIMER
veeam	onenote	dfssvc.exe
oracle	outlook	dfsrs.exe
ocssd	powerpnt	swc_service.exe
dbsnmp	steam	sophos
synctime	thebat	SAVAdminService
agntsvc	thunderbird	SavService.exe
isqlplussvc	visio	Hyper-v

xfssvccon	winword	TeamViewer
mydesktopservice	wordpad	Teamviwer
ocautoupds	EduLink2SIMS	Vmware
encsvc	bengine	vss
firefox	benetns	memtas
tbirdconfig	beserver	mepocs
mydesktopqos	pvlsvr	backup
ocomm	beremote	pulseway
dbeng50	VxLockdownServer	logme
sqbcoreservice	postgres	logmein
excel	fdhost	connectwise
infopath	WSSADMIN	splashtop
		wuauserv

It also forcibly stops specific services. If it is determined that specific services are running, the corresponding services are requested to stop (see the figure below). ControlService function is used and SERVICE_CONTROL_STOP is passed to request that the service to stop working.

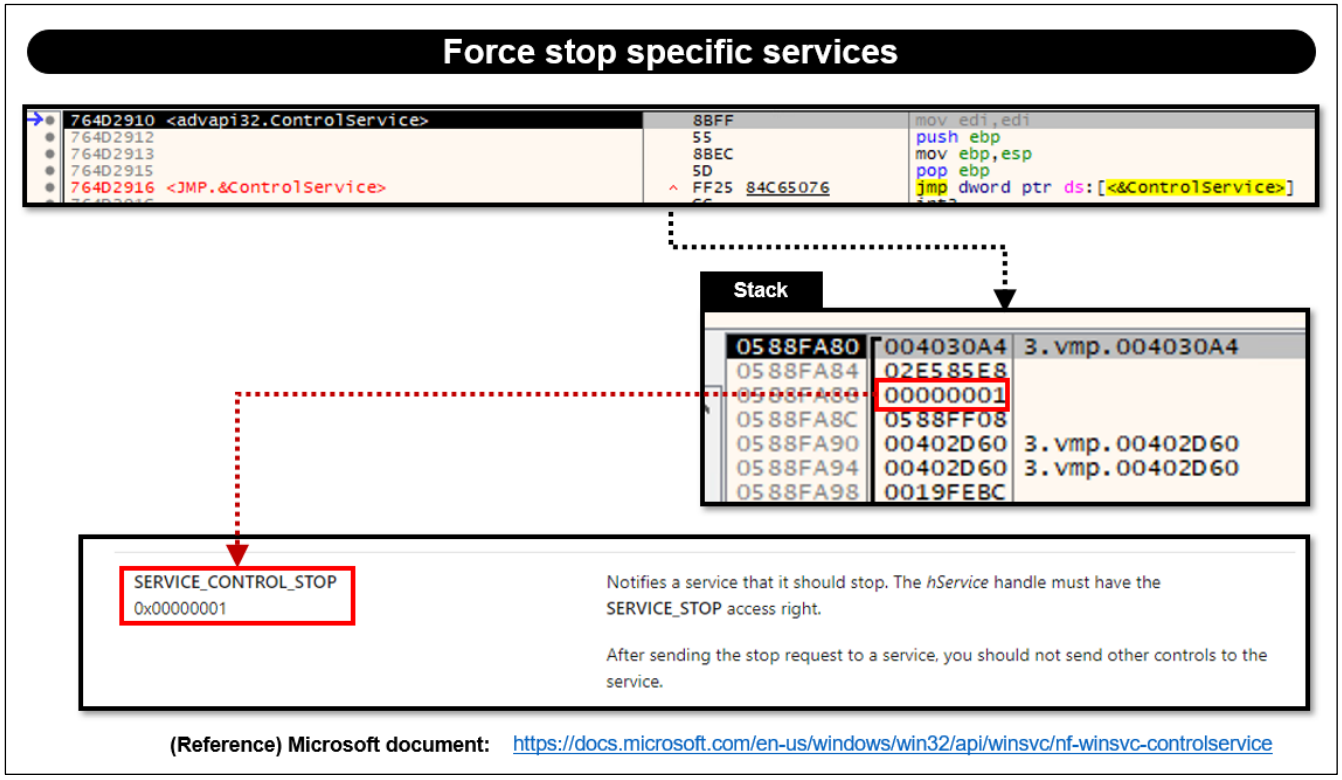

Fig. 18 - Force stop when specific services are running

The actual list of services to stop is as follows:

vss	pulseway
sql	logme
memtas	logmein
mepocs	connectwise
sophos	splashtop
veeam	wuauserv
backup	

7. Encryption of files

Next, Ragnar Locker enters into the file encryption process, but one of the distinctive behaviors of Ragnar Locker is the following drive-mapping behavior:

Instead of simply enumerating the drives like other ransomware, Ragnar Locker forces to map volumes that are not mapped as local drives before encrypting them. This means that hidden volumes that are not mapped as drives are forced to the mapping state and encrypted. Following shows how drive mapping is performed.

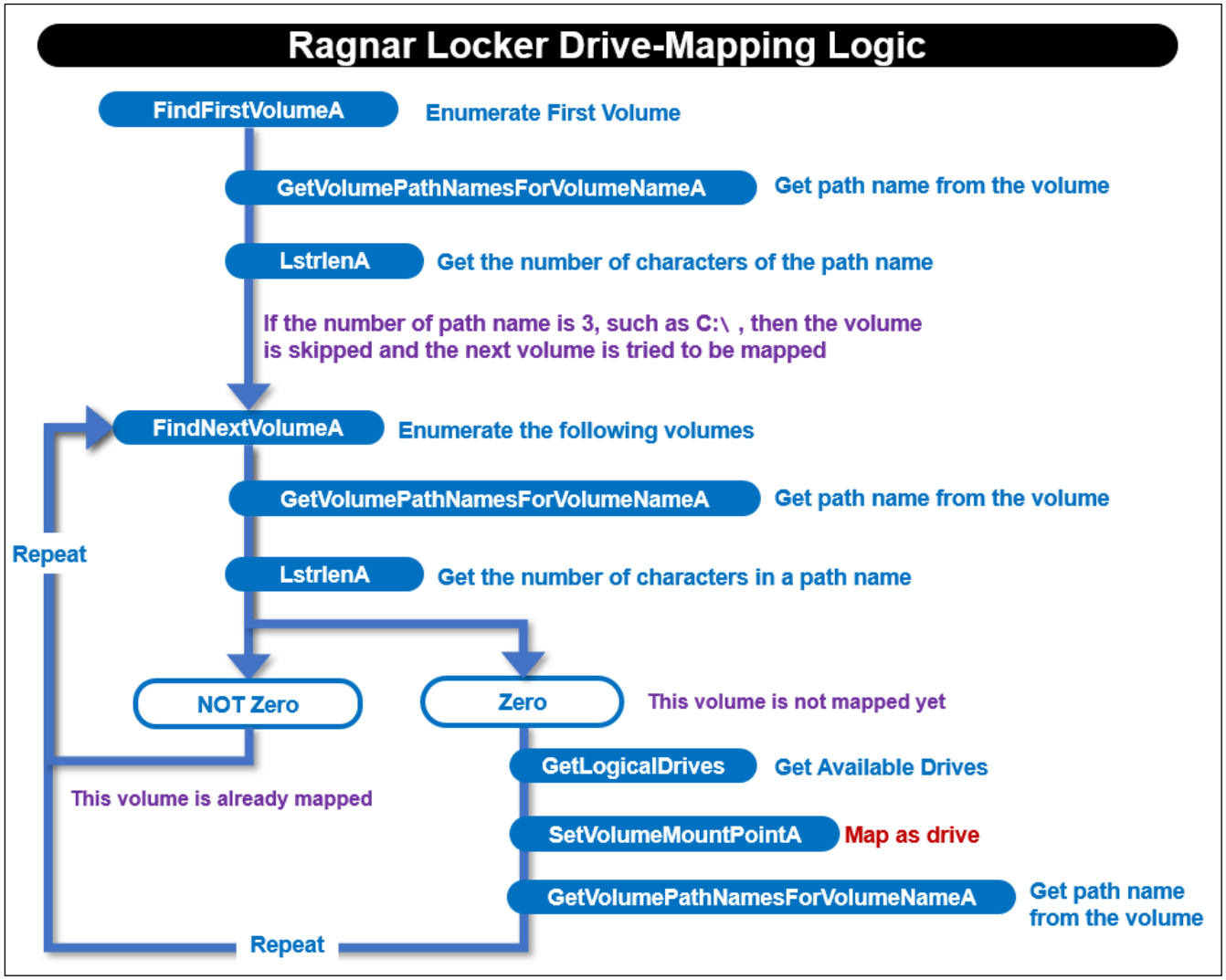

Fig. 19 - Drive-mapping logic of Ragnar Locker

Ragnar Locker encrypts all files that were found on the drives including newly mapped drives. However, following specific file names, extensions, and files in folders are excluded from the encryption.

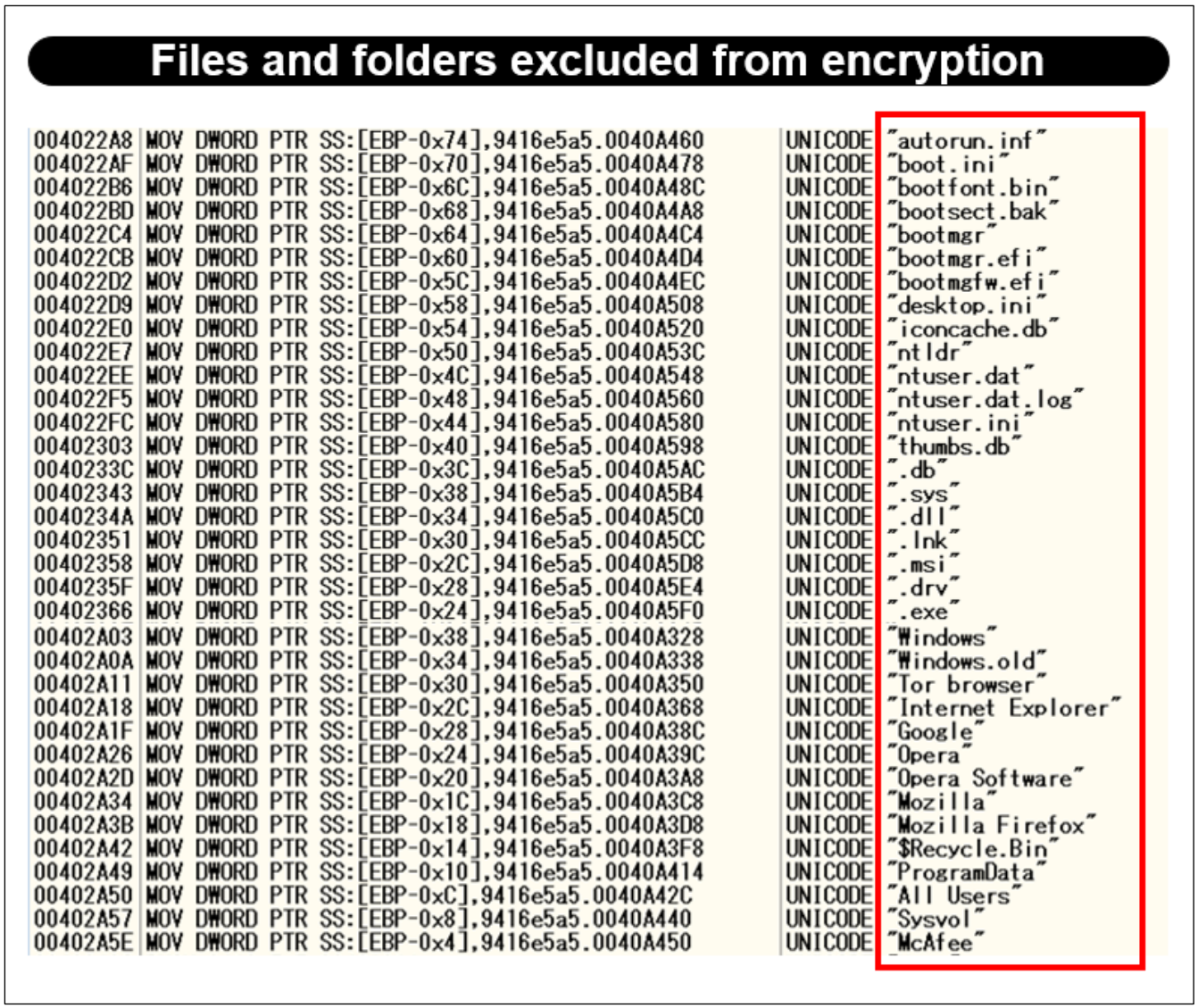

Fig. 20 - List of Ragnar Locker excluded from encryption

Ragnar Locker file encryption process is shown below. As you can see, it uses ReadFile and WriteFile, which is the same as another ransomware.

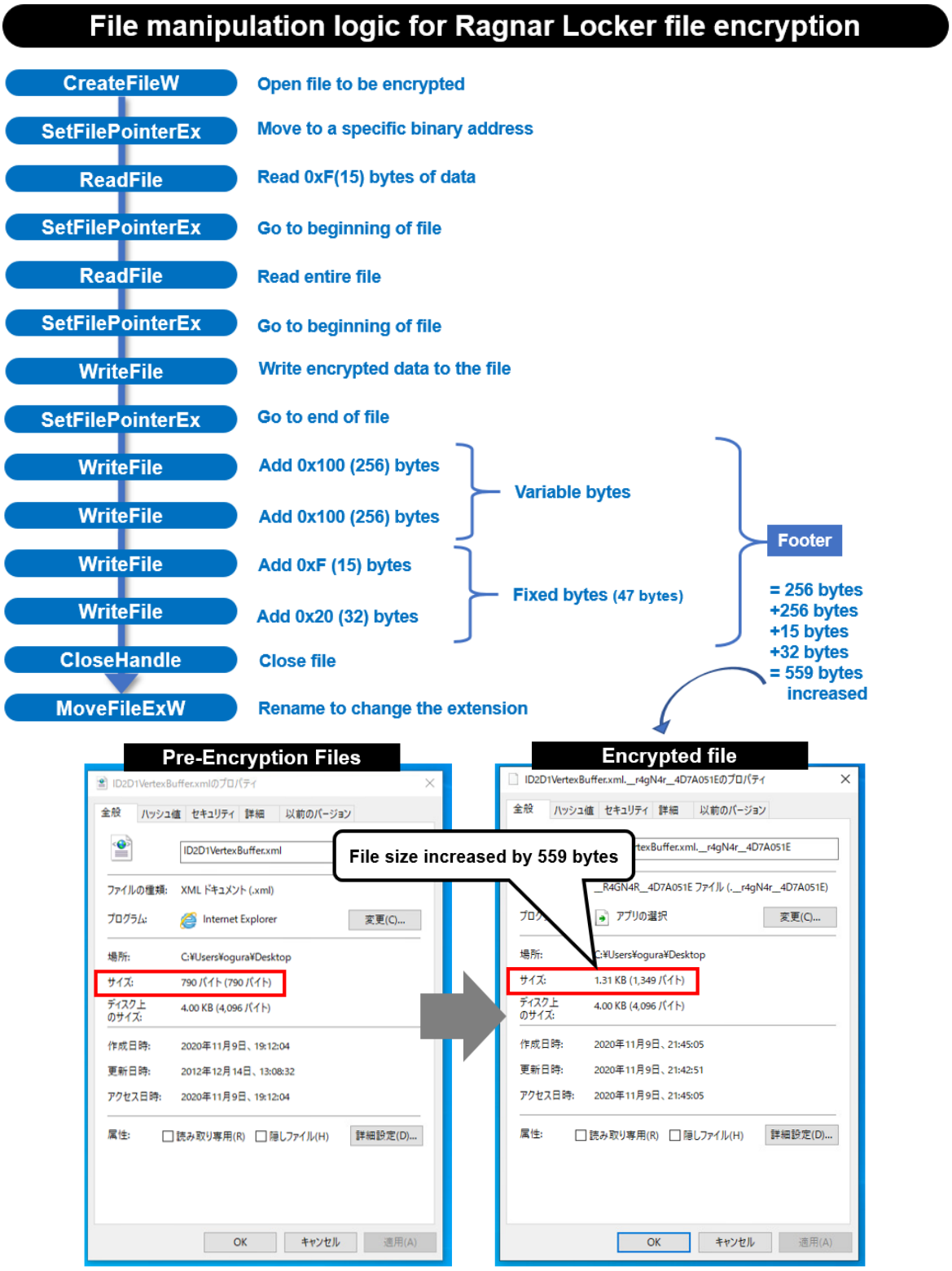

Fig. 21 - File manipulation logic during encryption

The last 47 bytes of the data is shown below. Since all files encrypted by Ragnar Locker have the same footer at the end of the 47 bytes, checking this area allows you to determine if they are Ragnar Locker encrypted files.

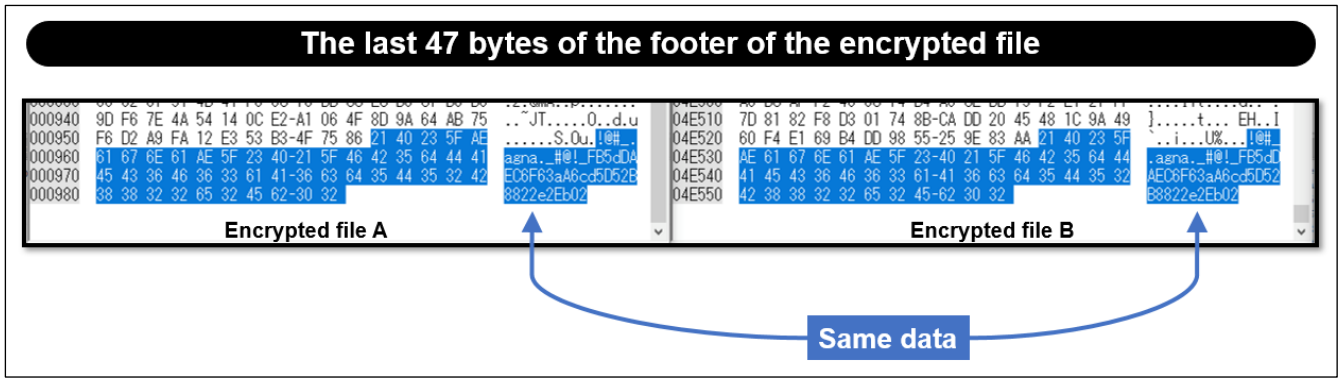

Fig. 22 - Common Footer, 47 bytes of an encrypted file

Encrypted file name is changed by moving the file using MoveFileEx as follows:

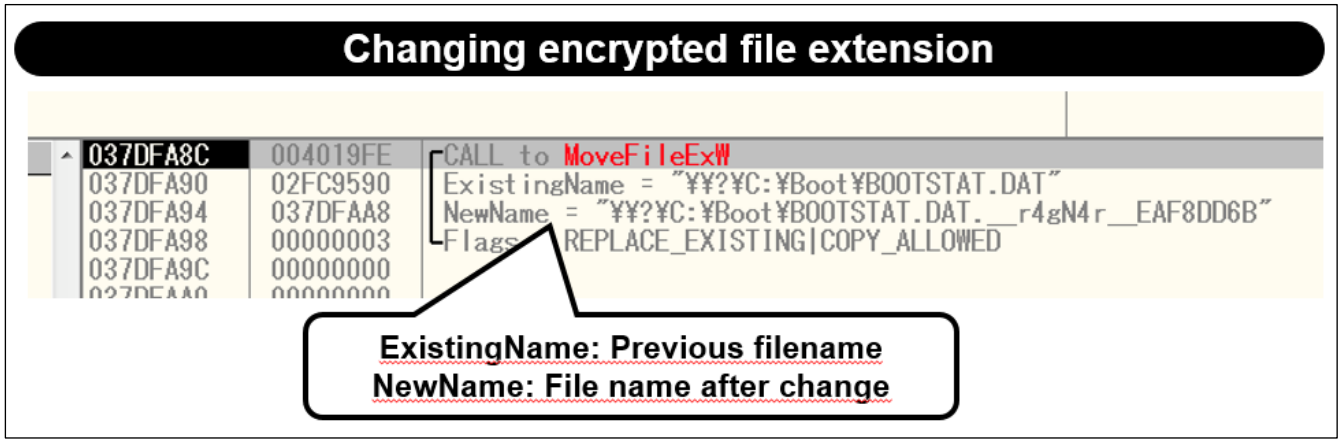

Fig. 23 - Changing encrypted file extension

This gives the encrypted file the following extensions:

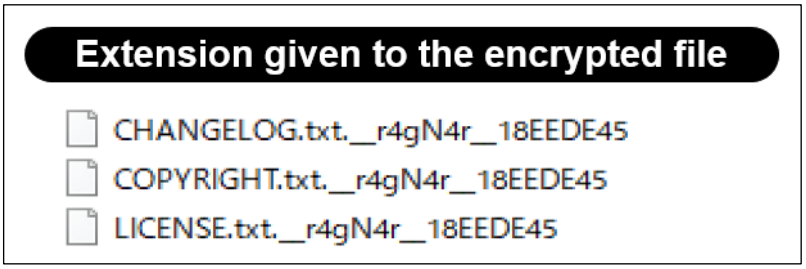

Fig. 24 - Extensions given to the encrypted files

The following RSA public key is used as part of the encryption process:

Public key used for encryption

```
'-----BEGIN PUBLIC KEY-----',0Ah
                    ; DATA XREF: .data:off_40C018↑o
'MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA1BCMqlGD1Mlq/Qtya4hO'
'+5N1BvJSo6/9ubWM7/S8rofNFGsj/ZGTovnLRDXKn7A2qwIEtPTpM8tih09HuZT8'
'wsu/nenL1rjvOPuMtDGoguNc8vpbeQEUDYneKUEdLq1fke6rD0Eq545eUDpgUQ7m'
'Ts5vlAK5a/ySGCe1r2k2hHl0kYdEb7Rg5vGP1uC5tRLPPE/MOFm/d+xs1Sed5FWc'
'BJSVIXiYlzdifBDc58nBDFXARmJX2xeqohJwoSW3j0asRKYuuznyQW/zTzyAh3GR'
'ZUpnGBVc33UO6AgysBmj3nMGdbBTG12mBfMxXjEotrjSMjV0g97i9Q9hI/TZdM9O'
'uQIDAQAB',0Ah
'-----END PUBLIC KEY-----',0Ah,0
```

Fig. 25 - Public key used for encryption

8. Creating and displaying threatening letter

Ragnar Locker creates a threatening letter in all folders as a text file. The file names always start from “!!!_READ_ME” (see illustration below).

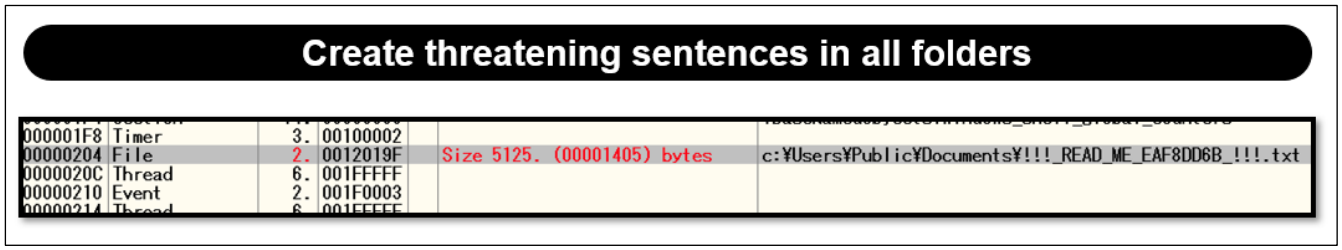

Fig. 26 - Creating a text file with threats in all folders

Some sentences are hard-coded in Ragnar Locker binary file to be used for the threatening letter.

Fig. 27 - Threatening letter strings visible in memory

Ragnar Locker writes threatening letters from memory to text files by using WriteFile (see illustration below).

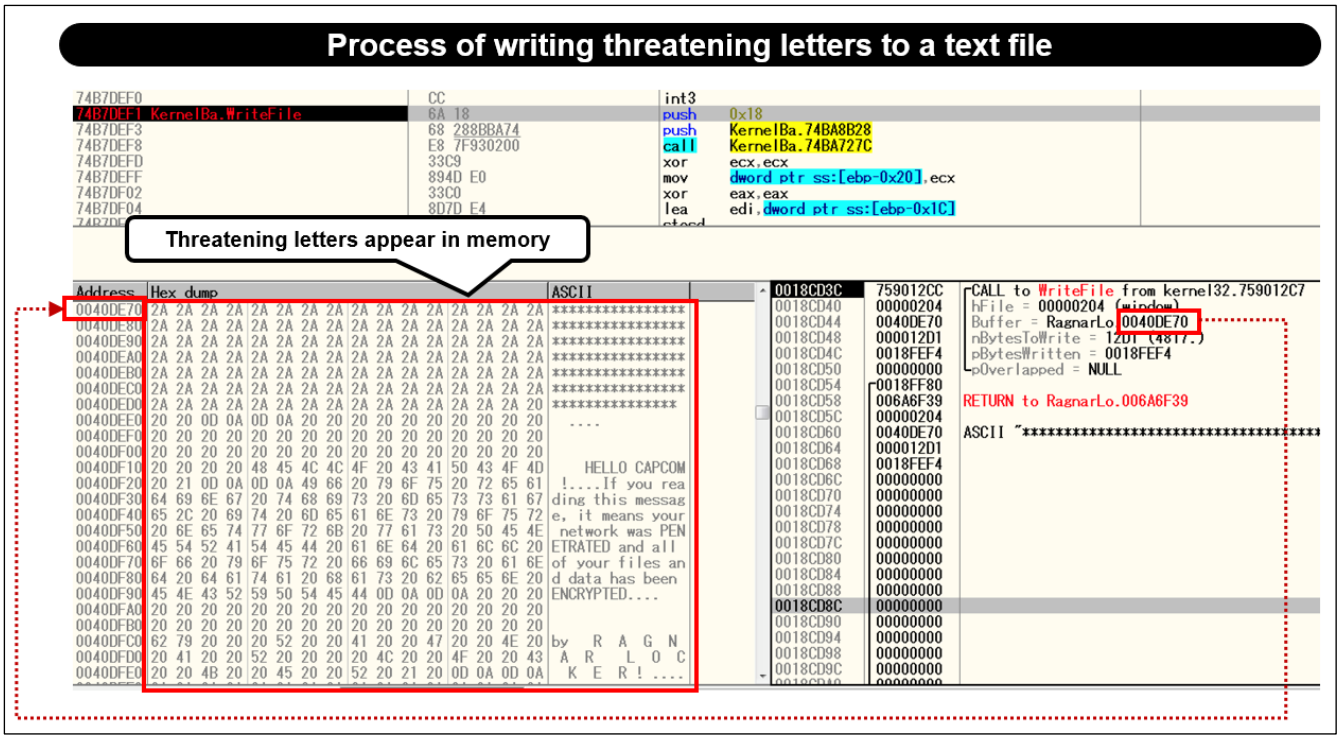

Fig. 28 - Writing Threatening letters from memory to text files

Threatening letter consists from 2 parts: fixed sentences and dynamically generated sentences, each written in two sessions. The following is the second write session that writes encrypted sentences encoded with a Base64 that was dynamically generated.

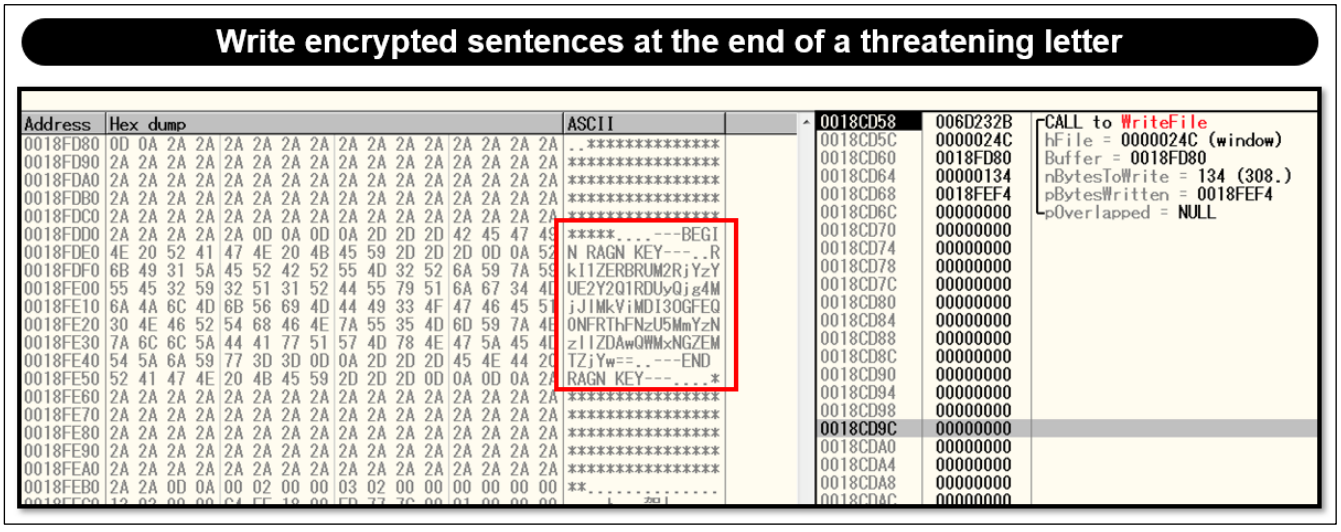

Fig. 29 - Writing encrypted sentences at the end of a threatening letter

Note that Ragnar Locker provides temporary and private leak page that is restricted in access separately from the leak sites to be disclosed to the public, and private leak page is presented only to the target (if the target did not pay the ransom money, the stolen information will be released to the public leak sites). The URL and password of the private leak page is only presented to the affected PC, and those URL and password is also hard-coded in Ragnar Locker binaries (see figure below).

Fig. 30 - Temporary leak page URL and password

When all encryption is completed, Ragnar Locker uses log-in user's privilege and start notepad.exe, and then displays the threatening letter created in the Document folder, by using CreateProcessAsUserW function in KernelBase.dll.

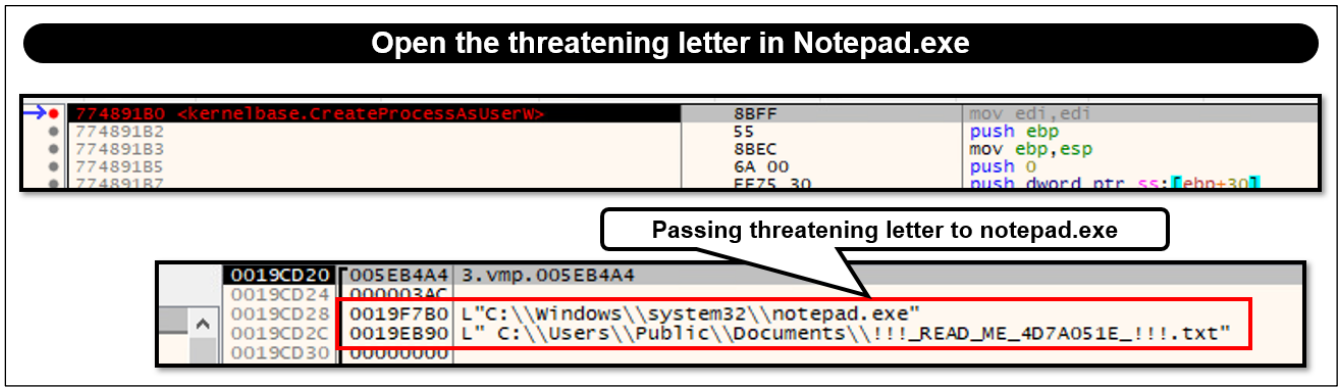

Fig. 31 - Opening the threatening letter in notepad.exe

The following shows the threatening letter on the infected PC. As you can see from the text beginning with Hello <company name>, the attacker clearly targeted the company and sent the ransomware binaries individually to the target company

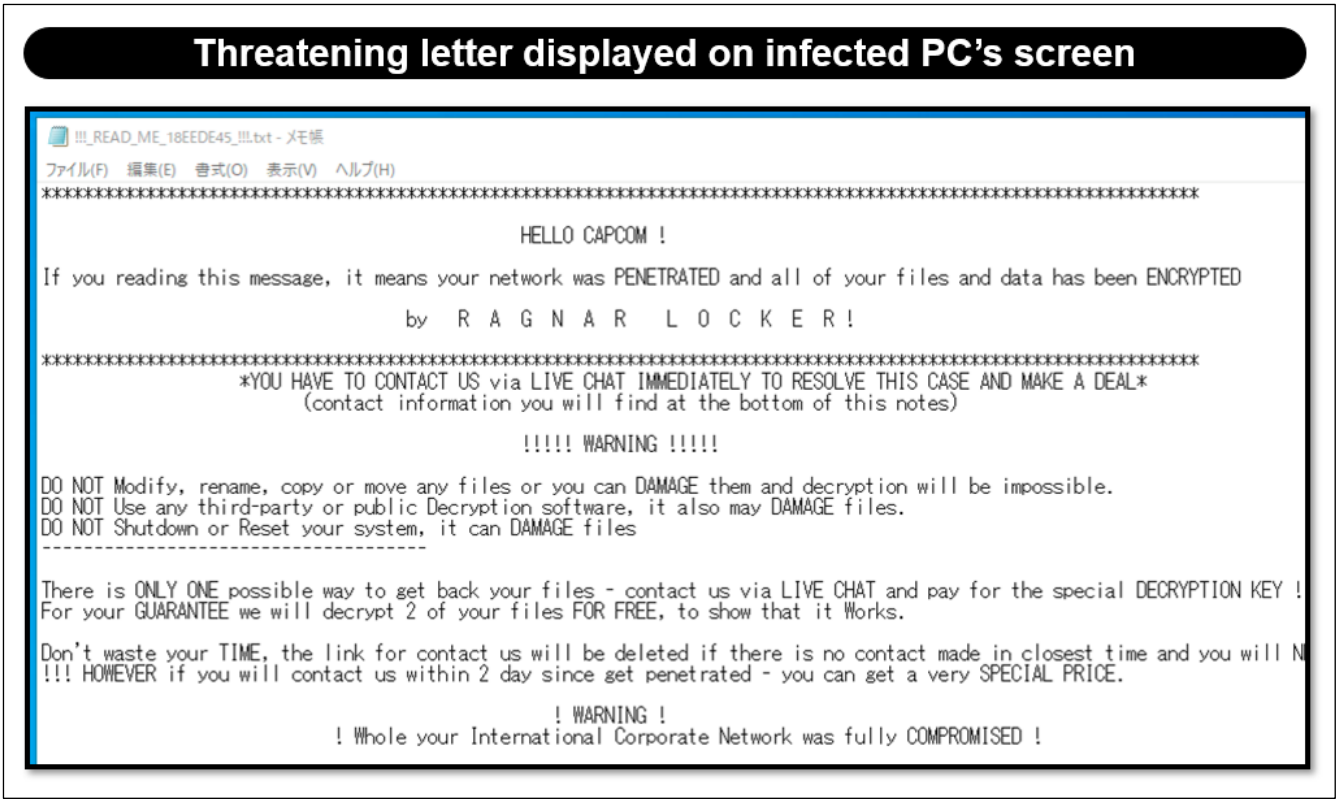

Fig. 32 - Threatening letter displayed after all encryption is completed

As shown above in Fig. 32, the threatening letter begins with the word "HELLO CAPCOM", the attacker customized the Ragnar Locker before they send. In addition to this specimen, we confirmed other Ragnar Locker also customized individually to each target company, in the format HELLO <company name>.

9. Supplemental information - Checking behavior of execution arguments

This article analyzed Ragnar Locker without adding arguments when executed. However, Ragnar Locker executable file has been designed to recognize the following executable arguments.

- Backup
- List
- Force
- Vm
- Vmback
- Share_network

In addition, when these execution arguments are given, there are minor differences such as the behavior of only process kill without file encryption and threatening statement creation, and the same behavior as when no execution argument is given. However, the major portion of behavior is the same as described above.

Other Ragnar Lockers discovered so far have been distributed together with the VM image at the time of deployment after intrusion, and attack methods that exploit the virtual environment have also been confirmed. Arguments such as -vm and -vmback have been confirmed in those methods. However, in this analysis, the attack method of the invasion route is unknown because I focused specimen analysis, so it is not clear how it was actually executed.

10. Ragnar Locker leak site

Like other targeted ransomware, Ragnar Locker has the following public leak sites, and the leak sites contain a large number of targeted companies.

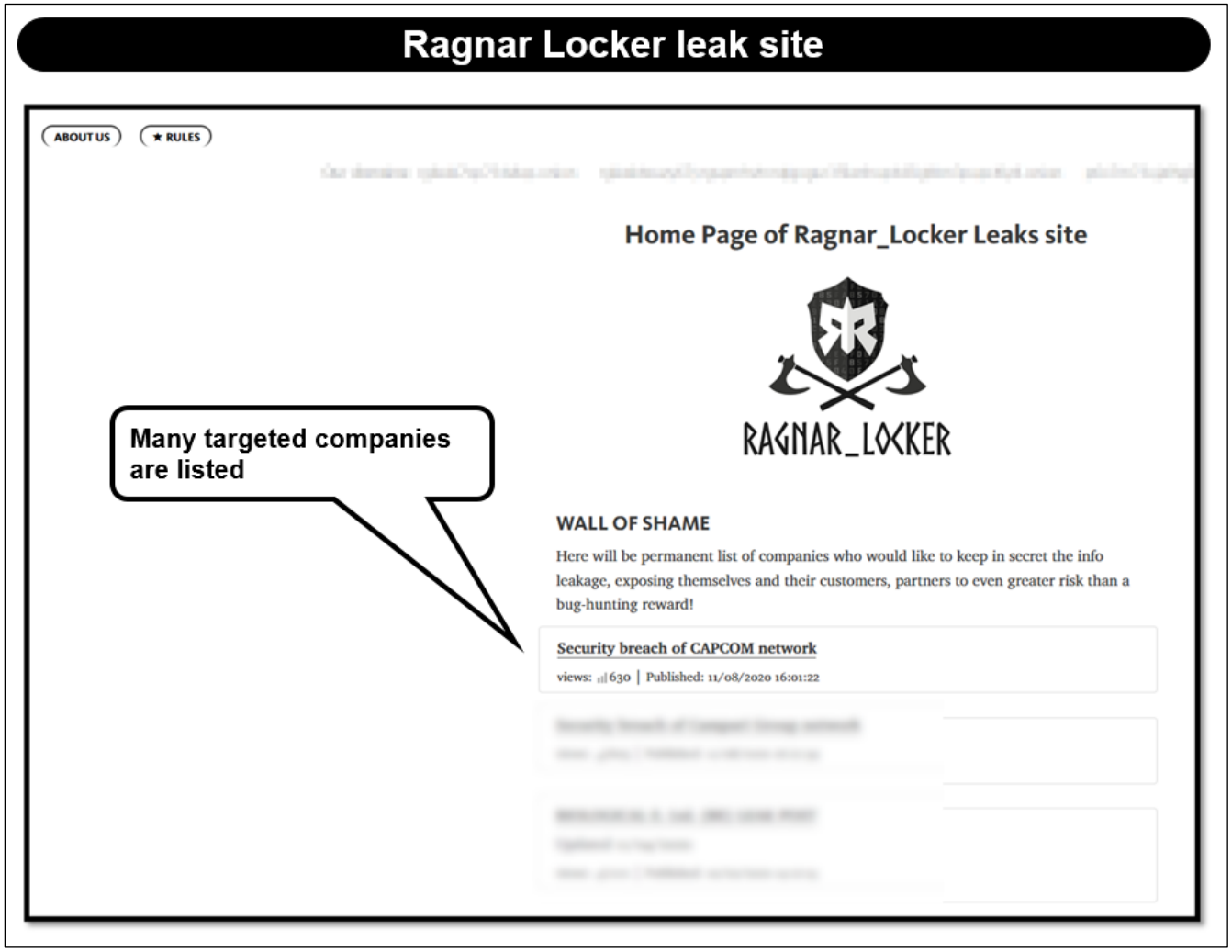

Fig. 33 - Ragnar Locker leakage site

Large amount of data that were stolen by the Ragnar Locker are available to download.

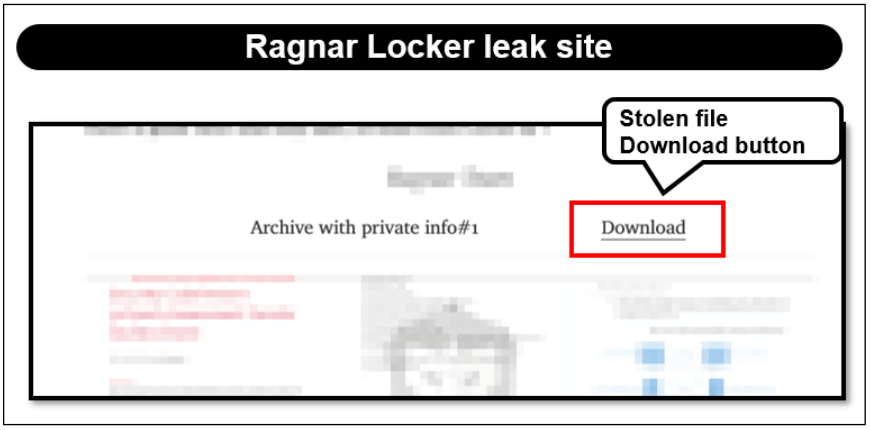

Fig. 34 - Stolen data from the targeted companies can be downloaded

The leak site contains an attack statement on CAPCOM, but at the time of this analysis, it is only a warning message, and no data has become public on the leak site. However, a screenshot is presented on the leak site as an evidence.

*** Updated on Nov. 11, 2020: A portion of the stolen data was confirmed disclosed on the site.**

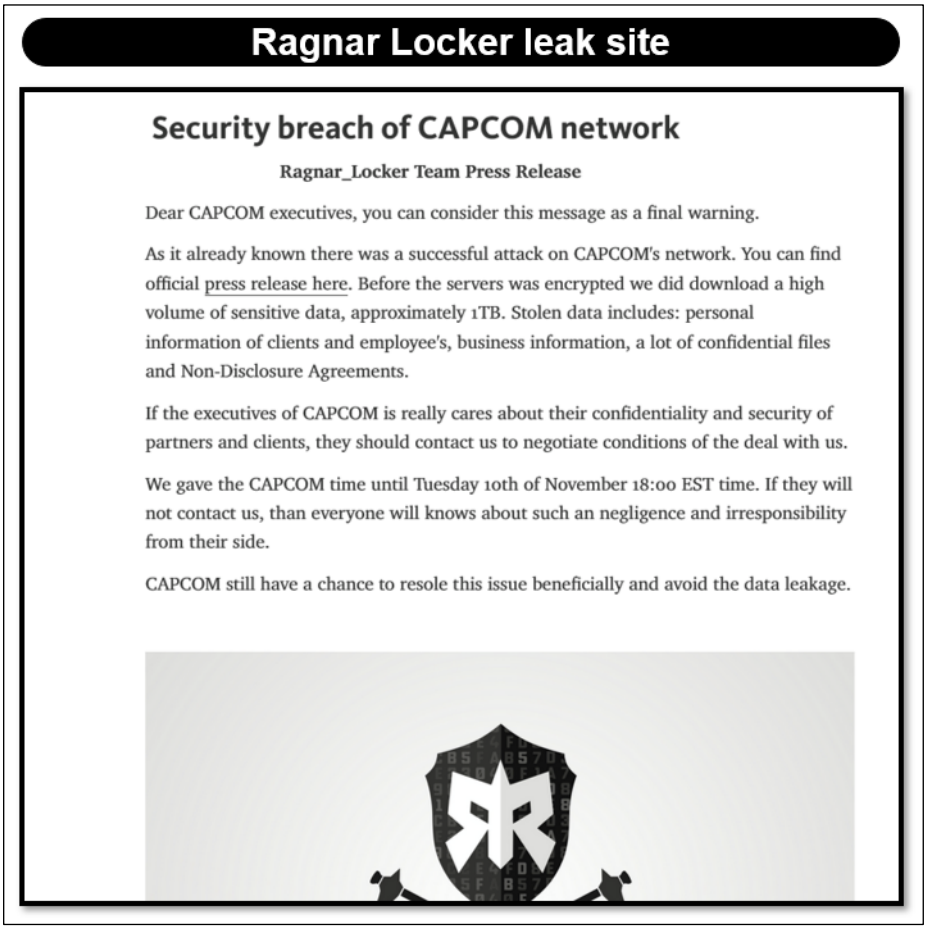

Fig. 35 - CAPCOM related message on Ragnar Locker leak site

11. Summary

As mentioned in the analysis article of SNAKE (EKANS) ransomware, there is a trend that targeted ransomware in recent years is sent after narrowing down the target to the attack target and customize it in advance. Therefore, even with the same type of ransomware, there is a high possibility that the behavior will be different depending on the attacked target. Therefore, it is going to be difficult to clearly and uniformly describe the behavior of the specimen. Detailed analysis of each specimen is essential to understand the behavior of the specimen itself.

12. About us

MBSD (Mitsui Bussan Secure Directions, Inc.) is the Japanese leading security company in managed security services, vulnerability assessment and testing, GRC (Governance, Risk, Compliance) consulting, incident response and handling, digital forensics, and secure programming training services. The MBSD services are provided by its personnel including the leading security experts in the field of secure programming, application security, penetration testing and threat analysis who have in-depth knowledge and understanding of attackers' methodologies. MBSD is working for the Internet infrastructure companies, cyber commerce and media giants, financial institutes, global enterprise, and government agencies in Japan to support their strategies against rapidly increasing threats from cyber space.

Company Contact Information:

Mitsui Bussan Secure Directions, Inc.

Yusen Suitengumae Building 6F, 1-14-8, Nihonbashi Ningyo Cho, Chuo-ku, Tokyo, 103-0013, Japan

+81 3 5649 1961 | http://www.mbsd.jp/