A Deep Dive into Cactus Ransomware

Prepared by: Vlad Pasca, Senior Malware & Threat Analyst

SecurityScorecard.com
info@securityscorecard.com

Tower 49
12 E 49th Street
Suite 15-001
New York, NY 10017
1.800.682.1707

Table of contents

Executive summary

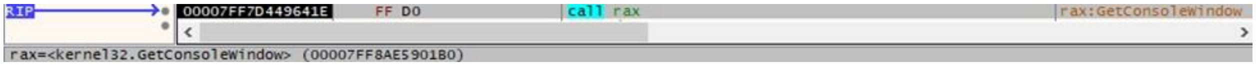
Cactus ransomware was discovered in March 2023. The malware creates a mutex called "b4kr-xr7h-qcps-omu3cAcTuS" to ensure that only one copy is running at a time. Persistence is achieved by creating a scheduled task named "Updates Check Task". The ransomware requires an AES key to decrypt the encrypted public RSA key stored in the binary.

The files are encrypted using the AES algorithm (OpenSSL library), with the key being encrypted using the public RSA key. The extension of the encrypted files is changed to "cts0" or "cts1".

Analysis and findings

SHA256: 78C16DE9FC07F1D0375A093903F86583A4E32037A7DA8AA2F90ECB15C4862C17

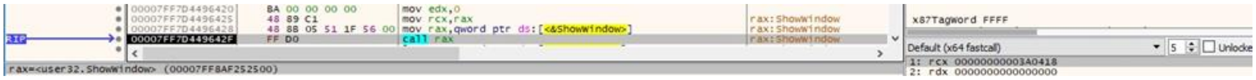
The ransomware is packed with UPX. It retrieves the window handle used by the console:



```
RIP  00007FF7D449641E   FF D0    call rax                rax:GetConsoleWindow
rax=<kernel32.GetConsoleWindow> (00007FF8AE5901B0)
```

Figure 1

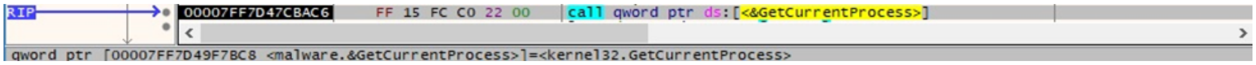
The process hides the window by calling the ShowWindow API (0x0 = **SW_HIDE**):



```
    00007FF7D4496420   BA 00 00 00 00         mov edx,0
    00007FF7D4496425   48 89 C1               mov rcx,rax                        rax:ShowWindow
    00007FF7D4496428   48 8B 05 51 1F 56 00   mov rax,qword ptr ds:[<&ShowWindow>] rax:ShowWindow
RIP 00007FF7D449642F   FF D0                  call rax                           rax:ShowWindow
rax=<user32.ShowWindow> (00007FF8AF252500)

x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 00000000003A0418
2: rdx 0000000000000000
```

Figure 2

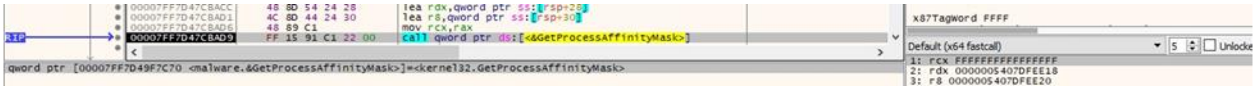
It obtains a pseudo handle for the current process using GetCurrentProcess:



```
RIP  00007FF7D47CBAC6   FF 15 FC C0 22 00   call qword ptr ds:[<&GetCurrentProcess>]
qword ptr [00007FF7D49F7BC8 <malware.&GetCurrentProcess>]=<kernel32.GetCurrentProcess>
```

Figure 3

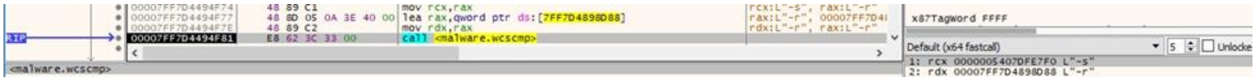
The GetProcessAffinityMask function is utilized to extract the process affinity mask and the system affinity mask for the system:



```
    00007FF7D47CBACC   48 8D 54 24 28      lea rdx,qword ptr ss:[rsp+28]
    00007FF7D47CBAD1   4C 8D 44 24 30      lea r8,qword ptr ss:[rsp+30]
    00007FF7D47CBAD6   48 89 C1            mov rcx,rax
RIP 00007FF7D47CBAD9   FF 15 91 C1 22 00   call qword ptr ds:[<&GetProcessAffinityMask>]
qword ptr [00007FF7D49F7C70 <malware.&GetProcessAffinityMask>]=<kernel32.GetProcessAffinityMask>

x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx FFFFFFFFFFFFFFFF
2: rdx 0000005407DFEE18
3: r8 0000005407DFEE20
```

Figure 4

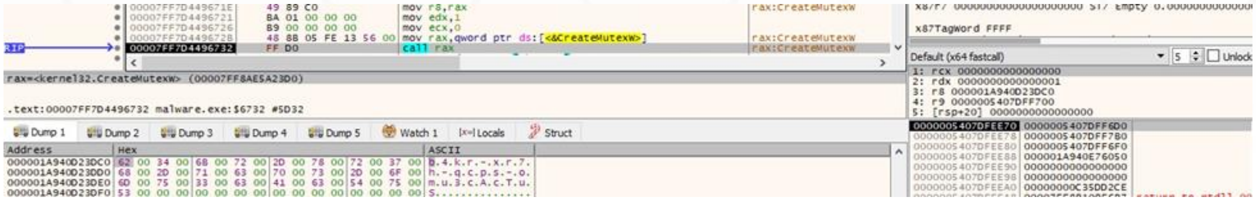
The malware can run with at least one of the following parameters: "-s", "-r", "-i", "-l", "-e", "-c", "-t", "-d", and "-f". We'll describe the purpose of every parameter in the upcoming paragraphs.



```
    00007FF7D4494F74   48 89 C1               mov rcx,rax                          rcx:L"-s", rax:L"-r"
    00007FF7D4494F77   48 8D 05 0A 3E 40 00   lea rax,qword ptr ds:[7FF7D4898D88]   rax:L"-r", 00007FF7D4
    00007FF7D4494F7E   48 89 C2               mov rdx,rax                          rdx:L"-r", rax:L"-r"
RIP 00007FF7D4494F81   E8 62 3C 33 00         call <malware.wcscmp>
<malware.wcscmp>

x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 0000005407DFE7F0 L"-s"
2: rdx 00007FF7D4898D88 L"-r"
```

Figure 5

The binary creates a mutex called “b4kr-xr7h-qcps-omu3cAcTuS” to ensure that only one copy of the executable is running at a time (see Figure 6).



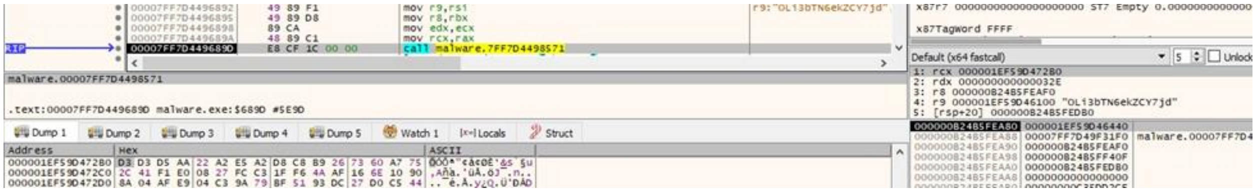
```
00007FF7D449671E   49 89 C0               mov r8,rax                                  rax:CreateMutexW
00007FF7D4496721   BA 01 00 00 00         mov edx,1
00007FF7D4496726   B9 00 00 00 00         mov ecx,0
00007FF7D4496728   48 8B 05 FE 13 56 00   mov rax,qword ptr ds:[<&CreateMutexW>]      rax:CreateMutexW
00007FF7D4496732   FF D0                  call rax                                    rax:CreateMutexW

rax=<kernel32.CreateMutexW> (00007FF8AE5A23D0)

.text:00007FF7D4496732 malware.exe:$6732 #5D32

Address           Hex                                              ASCII
000001A940D23DC0  62 00 34 00 6B 00 72 00 2D 00 78 00 72 00 37 00  b.4.k.r.-.x.r.7.
000001A940D23DD0  68 00 2D 00 71 00 63 00 70 00 73 00 2D 00 6F 00  h.-.q.c.p.s.-.o.
000001A940D23DE0  6D 00 75 00 33 00 63 00 41 00 63 00 54 00 75 00  m.u.3.c.A.c.T.u.
000001A940D23DF0  53 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  S...............
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000000000

x87TagWord FFFF

Default (x64 fastcall)   5   Unlock
1: rcx 0000000000000000
2: rdx 0000000000000001
3: r8 000001A940D23DC0
4: r9 00000005407DFF700
5: [rsp+20] 0000000000000000

00000005407DFEE70  00000005407DFF6D0
00000005407DFEE78  00000005407DFF7B0
00000005407DFEE80  00000005407DFF6F0
00000005407DFEE88  000001A940E76050
00000005407DFEE90  0000000000000000
00000005407DFEE98  0000000000000000
00000005407DFEEA0  00000000C35DD2CE
```

Figure 6

Running with the -s parameter

The public RSA key is stored in an encrypted form. The AES key used to decrypt the RSA key is parsed from the “C:\ProgramData\ntuser.dat” file, which should have been created earlier. The initialization vector is hard-coded “OLi3bTN6ekZCY7jd”:

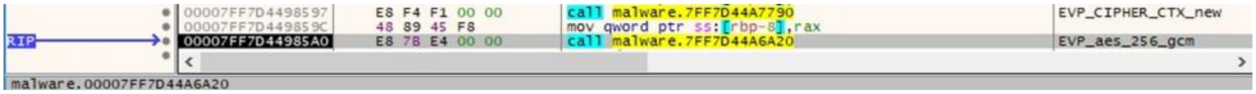


```
00007FF7D4496892   49 89 F1               mov r9,rsi                     r9:"OLi3bTN6ekZCY7jd"
00007FF7D4496895   49 89 D8               mov r8,rbx
00007FF7D4496898   89 CA                  mov edx,ecx
00007FF7D449689A   48 89 C1               mov rcx,rax
00007FF7D449689D   E8 CF 1C 00 00         call malware.7FF7D4498571

malware.00007FF7D4498571

.text:00007FF7D449689D malware.exe:$689D #5E9D

Address           Hex                                              ASCII
000001EF59D472B0  D3 D3 D5 AA 22 A2 E5 A2 D8 C8 B9 26 73 60 A7 75  ÓÓÕª"¢å¢ØÈ¹&s`§u
000001EF59D472C0  2C 41 F1 E0 08 27 FC C3 1F F6 4A AF 16 6E 10 90  ,Añà.'üÃ.öJ¯.n..
000001EF59D472D0  8A 04 AF E9 04 C3 9A 79 BF 51 93 DC 27 D0 C5 44  ..¯é.Ã.y¿Q.Ü'ÐÅD
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000000000

x87TagWord FFFF

Default (x64 fastcall)   5   Unlock
1: rcx 000001EF59D472B0
2: rdx 000000000000032E
3: r8 0000000B24B5FEAF0
4: r9 000001EF59D46100 "OLi3bTN6ekZCY7jd"
5: [rsp+20] 0000000B24B5FEDB0

0000000B24B5FEA80  000001EF59D46440
0000000B24B5FEA88  00007FF7D49F31F0  malware.00007FF7D4
0000000B24B5FEA90  0000000B24B5FEAF0
0000000B24B5FEA98  0000000B24B5FF40F
0000000B24B5FEAA0  0000000B24B5FEDB0
0000000B24B5FEAA8  0000000000000000
```

Figure 7

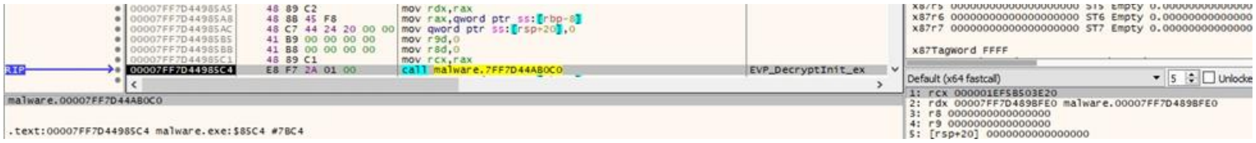
The public key is decrypted using AES256 Galois Counter Mode (GCM):



```
00007FF7D4498597   E8 F4 F1 00 00   call malware.7FF7D44A7790          EVP_CIPHER_CTX_new
00007FF7D449859C   48 89 45 F8      mov qword ptr ss:[rbp-8],rax
00007FF7D44985A0   E8 7B E4 00 00   call malware.7FF7D44A6A20          EVP_aes_256_gcm

malware.00007FF7D44A6A20
```

Figure 8

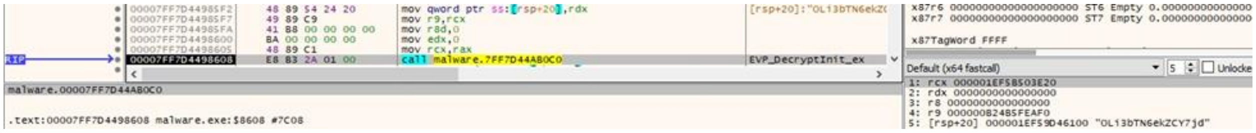
OpenSSL’s EVP_DecryptInit_ex function is used to start decrypting the information, as highlighted below.



```
00007FF7D44985A5   48 89 C2               mov rdx,rax
00007FF7D44985A8   48 8B 45 F8            mov rax,qword ptr ss:[rbp-8]
00007FF7D44985AC   48 C7 44 24 20 00 00   mov qword ptr ss:[rsp+20],0
00007FF7D44985B5   41 B9 00 00 00 00      mov r9d,0
00007FF7D44985BB   41 B8 00 00 00 00      mov r8d,0
00007FF7D44985C1   48 89 C1               mov rcx,rax
00007FF7D44985C4   E8 F7 2A 01 00         call malware.7FF7D44AB0C0      EVP_DecryptInit_ex

malware.00007FF7D44AB0C0

.text:00007FF7D44985C4 malware.exe:$85C4 #7BC4
```

```
x87r5 00000000000000000000 ST5 Empty 0.00000000000000
x87r6 00000000000000000000 ST6 Empty 0.00000000000000
x87r7 00000000000000000000 ST7 Empty 0.00000000000000

x87TagWord FFFF

Default (x64 fastcall)   5   Unlocke
1: rcx 000001EF5B503E20
2: rdx 00007FF7D489BFE0 malware.00007FF7D489BFE0
3: r8 0000000000000000
4: r9 0000000000000000
5: [rsp+20] 0000000000000000
```

Figure 9

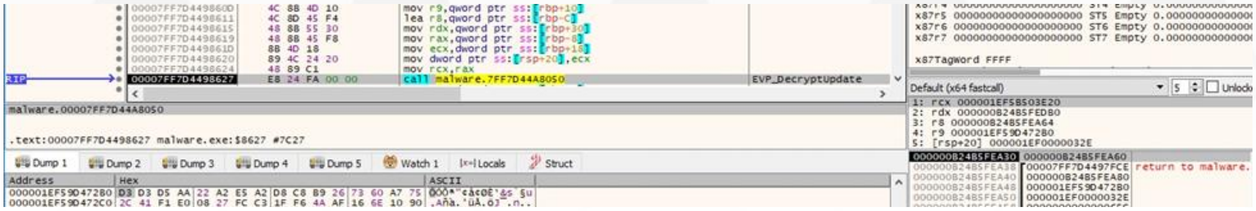


```
00007FF7D44985F2   48 89 54 24 20         mov qword ptr ss:[rsp+20],rdx  [rsp+20]:"OLi3bTN6ekZ(
00007FF7D44985F7   49 89 C9               mov r9,rcx
00007FF7D44985FA   41 B8 00 00 00 00      mov r8d,0
00007FF7D4498600   BA 00 00 00 00         mov edx,0
00007FF7D4498605   48 89 C1               mov rcx,rax
00007FF7D4498608   E8 B3 2A 01 00         call malware.7FF7D44AB0C0      EVP_DecryptInit_ex

malware.00007FF7D44AB0C0

.text:00007FF7D4498608 malware.exe:$8608 #7C08
```

```
x87r6 00000000000000000000 ST6 Empty 0.00000000000000
x87r7 00000000000000000000 ST7 Empty 0.00000000000000

x87TagWord FFFF

Default (x64 fastcall)   5   Unlocke
1: rcx 000001EF5B503E20
2: rdx 0000000000000000
3: r8 0000000000000000
4: r9 0000000B24B5FEAF0
5: [rsp+20] 000001EF59D46100 "OLi3bTN6ekZCY7jd"
```

Figure 10

Finally, the RSA key is decrypted by calling the EVP_DecryptUpdate method (Figure 11).

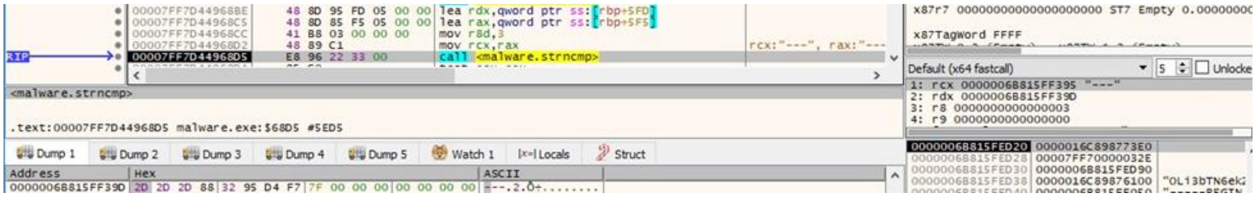


```
00007FF7D449860D  4C 8B 4D 10        mov r9,qword ptr ss:[rbp+10]
00007FF7D4498611  4C 8D 45 F4        lea r8,qword ptr ss:[rbp-C]
00007FF7D4498615  48 8B 55 30        mov rdx,qword ptr ss:[rbp+30]
00007FF7D4498619  48 8B 45 F8        mov rax,qword ptr ss:[rbp-8]
00007FF7D449861D  8B 4D 18           mov ecx,dword ptr ss:[rbp+18]
00007FF7D4498620  89 4C 24 20        mov dword ptr ss:[rsp+20],ecx
00007FF7D4498624  48 89 C1           mov rcx,rax
00007FF7D4498627  E8 24 FA 00 00     call malware.7FF7D44A8050          EVP_DecryptUpdate

malware.00007FF7D44A8050

.text:00007FF7D4498627 malware.exe:$8627 #7C27

Address           Hex                                              ASCII
000001EF59D472B0  D3 D3 D5 AA 22 A2 E5 A2 D8 C8 B9 26 73 60 A7 75  ÓÓÕª"¢å¢ØÈ¹&s`§u
000001EF59D472C0  2C 41 F1 E0 08 27 FC C3 1F F6 4A AF 16 6E 10 90  ,Añà.'üÃ.öJ¯.n..
```

```
x87r5 00000000000000000000 ST5 Empty 0.000000000000000
x87r6 00000000000000000000 ST6 Empty 0.000000000000000
x87r7 00000000000000000000 ST7 Empty 0.000000000000000

x87TagWord FFFF

Default (x64 fastcall)   5   □ Unlock
1: rcx 000001EF5B503E20
2: rdx 000000B24B5FEDB0
3: r8 000000B24B5FEA64
4: r9 000001EF59D472B0
5: [rsp+20] 000001EF0000032E

000000B24B5FEA30  000000B24B5FEA60
000000B24B5FEA38  00007FF7D4497FCE  return to malware.
000000B24B5FEA40  000000B24B5FEA80
000000B24B5FEA48  000001EF59D472B0
000000B24B5FEA50  000001EF0000032E
```

Figure 11

The ransomware checks if the decryption was successful by verifying the first 3 characters:

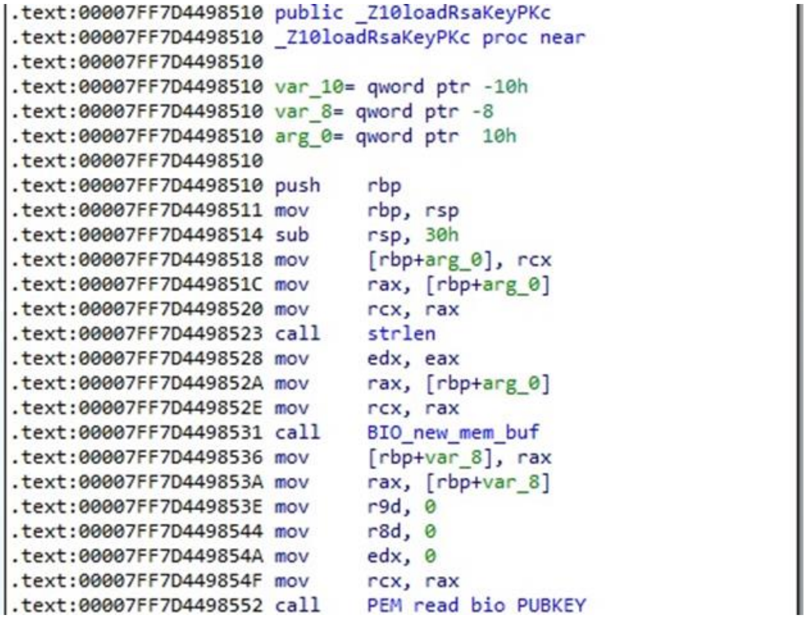


```
00007FF7D44968BE  48 8D 95 FD 05 00 00  lea rdx,qword ptr ss:[rbp+5FD]
00007FF7D44968C5  48 8D 85 F5 05 00 00  lea rax,qword ptr ss:[rbp+5F5]
00007FF7D44968CC  41 B8 03 00 00 00     mov r8d,3
00007FF7D44968D2  48 89 C1              mov rcx,rax                       rcx:"---", rax:"---
00007FF7D44968D5  E8 96 22 33 00        call <malware.strncmp>

<malware.strncmp>

.text:00007FF7D44968D5 malware.exe:$68D5 #5ED5

Address           Hex                                              ASCII
0000006B815FF39D  2D 2D 2D 88 32 95 D4 F7 7F 00 00 00 00 00 00 00  ---.2.Ô÷........
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000

x87TagWord FFFF

Default (x64 fastcall)   5   □ Unlocke
1: rcx 0000006B815FF395 "---"
2: rdx 0000006B815FF39D
3: r8 0000000000000003
4: r9 0000000000000000

0000006B815FED20  0000016C898773E0
0000006B815FED28  00007FF70000032E
0000006B815FED30  0000006B815FED90
0000006B815FED38  0000016C89876100  "OLi3bTN6ek
```

Figure 12

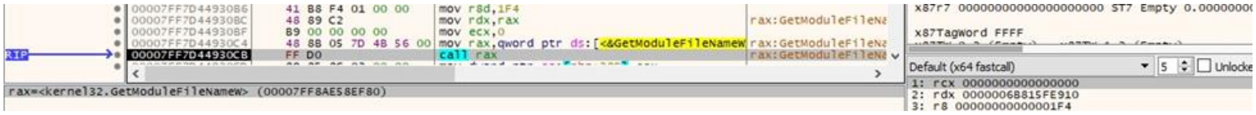
The malicious process loads the public RSA key using the PEM_read_bio_PUBKEY function, as shown in the figure below.



```
.text:00007FF7D4498510 public _Z10loadRsaKeyPKc
.text:00007FF7D4498510 _Z10loadRsaKeyPKc proc near
.text:00007FF7D4498510
.text:00007FF7D4498510 var_10= qword ptr -10h
.text:00007FF7D4498510 var_8= qword ptr -8
.text:00007FF7D4498510 arg_0= qword ptr  10h
.text:00007FF7D4498510
.text:00007FF7D4498510 push    rbp
.text:00007FF7D4498511 mov     rbp, rsp
.text:00007FF7D4498514 sub     rsp, 30h
.text:00007FF7D4498518 mov     [rbp+arg_0], rcx
.text:00007FF7D449851C mov     rax, [rbp+arg_0]
.text:00007FF7D4498520 mov     rcx, rax
.text:00007FF7D4498523 call    strlen
.text:00007FF7D4498528 mov     edx, eax
.text:00007FF7D449852A mov     rax, [rbp+arg_0]
.text:00007FF7D449852E mov     rcx, rax
.text:00007FF7D4498531 call    BIO_new_mem_buf
.text:00007FF7D4498536 mov     [rbp+var_8], rax
.text:00007FF7D449853A mov     rax, [rbp+var_8]
.text:00007FF7D449853E mov     r9d, 0
.text:00007FF7D4498544 mov     r8d, 0
.text:00007FF7D449854A mov     edx, 0
.text:00007FF7D449854F mov     rcx, rax
.text:00007FF7D4498552 call    PEM read bio PUBKEY
```

Figure 13

GetModuleFileNameW is utilized to extract the path of the executable file (see Figure 14).

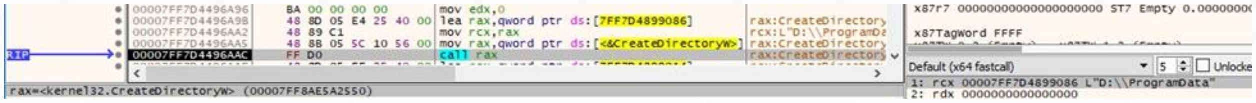


```
00007FF7D44930B6  41 B8 F4 01 00 00     mov r8d,1F4
00007FF7D44930BC  48 89 C2              mov rdx,rax                                 rax:GetModuleFileNa
00007FF7D44930BF  B9 00 00 00 00        mov ecx,0
00007FF7D44930C4  48 8B 05 7D 4B 56 00  mov rax,qword ptr ds:[<&GetModuleFileNameW  rax:GetModuleFileNa
00007FF7D44930CB  FF D0                 call rax                                    rax:GetModuleFileNa

rax=<kernel32.GetModuleFileNameW> (00007FF8AE58EF80)
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000

x87TagWord FFFF

Default (x64 fastcall)   5   □ Unlocke
1: rcx 0000000000000000
2: rdx 0000006B815FE910
3: r8 00000000000001F4
```

Figure 14

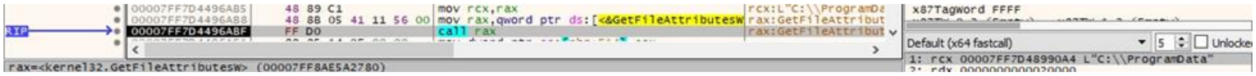
The binary is looking for the “D:\ProgramData” directory via a function call to CreateDirectoryW:



```
00007FF7D4496A96   BA 00 00 00 00        mov edx,0
00007FF7D4496A9B   48 8D 05 E4 25 40 00  lea rax,qword ptr ds:[7FF7D4899086]         rax:CreateDirectory
00007FF7D4496AA2   48 89 C1              mov rcx,rax                                  rcx:L"D:\\ProgramDa
00007FF7D4496AA5   48 8B 05 5C 10 56 00  mov rax,qword ptr ds:[<&CreateDirectoryW>]  rax:CreateDirectory
00007FF7D4496AAC   FF D0                 call rax                                     rax:CreateDirectory
rax=<kernel32.CreateDirectoryW> (00007FF8AE5A2550)

x87r7 00000000000000000000 ST7 Empty 0.0000000
x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 00007FF7D4899086 L"D:\\ProgramData"
2: rdx 0000000000000000
```

Figure 15

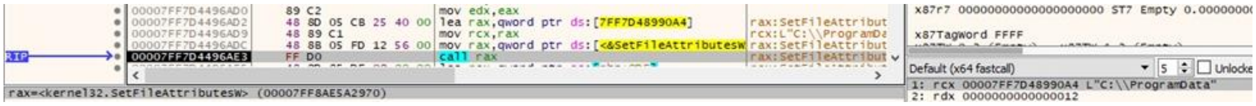
It retrieves file system attributes for the ProgramData folder:



```
00007FF7D4496AB5   48 89 C1              mov rcx,rax                                   rcx:L"C:\\ProgramDa
00007FF7D4496AB8   48 8B 05 41 11 56 00  mov rax,qword ptr ds:[<&GetFileAttributesW   rax:GetFileAttribut
00007FF7D4496ABF   FF D0                 call rax                                      rax:GetFileAttribut
rax=<kernel32.GetFileAttributesW> (00007FF8AE5A2780)

x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 00007FF7D48990A4 L"C:\\ProgramData"
2: rdx 0000000000020000
```

Figure 16

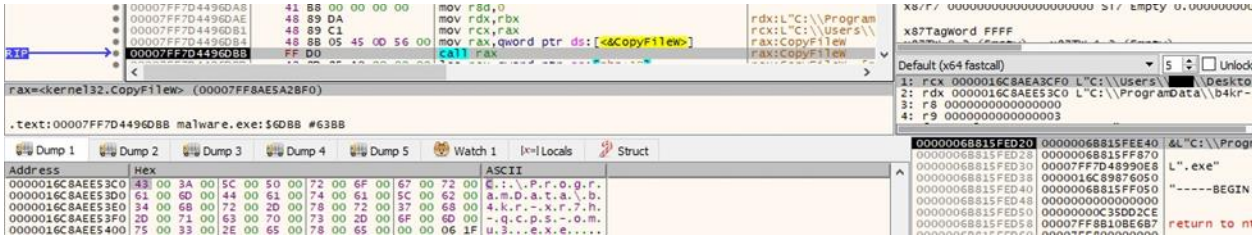
The above folder is hidden using the SetFileAttributesW API (0x12 = **FILE_ATTRIBUTE_DIRECTORY | FILE_ATTRIBUTE_HIDDEN**):



```
00007FF7D4496AD0   89 C2                 mov edx,eax
00007FF7D4496AD2   48 8D 05 CB 25 40 00  lea rax,qword ptr ds:[7FF7D48990A4]          rax:SetFileAttribut
00007FF7D4496AD9   48 89 C1              mov rcx,rax                                   rcx:L"C:\\ProgramDa
00007FF7D4496ADC   48 8B 05 FD 12 56 00  mov rax,qword ptr ds:[<&SetFileAttributesW   rax:SetFileAttribut
00007FF7D4496AE3   FF D0                 call rax                                      rax:SetFileAttribut
rax=<kernel32.SetFileAttributesW> (00007FF8AE5A2970)

x87r7 00000000000000000000 ST7 Empty 0.0000000
x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 00007FF7D48990A4 L"C:\\ProgramData"
2: rdx 0000000000000012
```

Figure 17

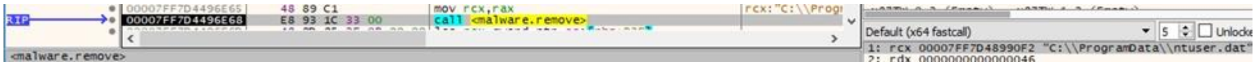
The executable is copied into the ProgramData folder as “C:\ProgramData\b4kr-xr7h-qcps-omu3.exe”:



```
00007FF7D4496DA8   41 B8 00 00 00 00     mov r8d,0
00007FF7D4496DAE   48 89 DA              mov rdx,rbx                            rdx:L"C:\\Program
00007FF7D4496DB1   48 89 C1              mov rcx,rax                            rcx:L"C:\\Users\\
00007FF7D4496DB4   48 8B 05 45 0D 56 00  mov rax,qword ptr ds:[<&CopyFileW>]   rax:CopyFileW
00007FF7D4496DBB   FF D0                 call rax                               rax:CopyFileW
rax=<kernel32.CopyFileW> (00007FF8AE5A28F0)

.text:00007FF7D4496DBB malware.exe:$6DBB #63BB

x87TagWord FFFF
Default (x64 fastcall)   5   Unlock
1: rcx 0000016C8AEA3CF0 L"C:\\Users\\[redacted]\\Deskto
2: rdx 0000016C8AEE53C0 L"C:\\ProgramData\\b4kr-
3: r8 0000000000000000
4: r9 0000000000000003

Address           Hex                                              ASCII
0000016C8AEE53C0  43 00 3A 00 5C 00 50 00 72 00 6F 00 67 00 72 00  C.:.\.P.r.o.g.r.
0000016C8AEE53D0  61 00 6D 00 44 00 61 00 74 00 61 00 5C 00 62 00  a.m.D.a.t.a.\.b.
0000016C8AEE53E0  34 00 6B 00 72 00 2D 00 78 00 72 00 37 00 68 00  4.k.r.-.x.r.7.h.
0000016C8AEE53F0  2D 00 71 00 63 00 70 00 73 00 2D 00 6F 00 6D 00  -.q.c.p.s.-.o.m.
0000016C8AEE5400  75 00 33 00 2E 00 65 00 78 00 65 00 00 00 06 1F  u.3...e.x.e.....

0000006B815FED20  0000006B815FEE40  &L"C:\\Progr
0000006B815FED28  0000006B815FF870
0000006B815FED30  00007FF7D48990E8  L".exe"
0000006B815FED38  0000016C89876050
0000006B815FED40  0000006B815FF050  "-----BEGIN
0000006B815FED48  0000000000000000
0000006B815FED50  00000000C35DD2CE
0000006B815FED58  00007FF8B10BE6B7  return to nt
```

Figure 18

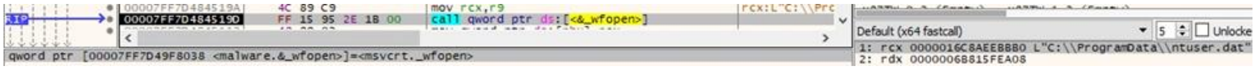
The malicious binary deletes the “ntuser.dat” file found in the ProgramData directory if it exists:



```
00007FF7D4496E65   48 89 C1              mov rcx,rax                rcx:"C:\\Prog
00007FF7D4496E68   E8 93 1C 33 00        call <malware.remove>
<malware.remove>

Default (x64 fastcall)   5   Unlocke
1: rcx 00007FF7D48990F2 "C:\\ProgramData\\ntuser.dat"
2: rdx 0000000000000046
```

Figure 19

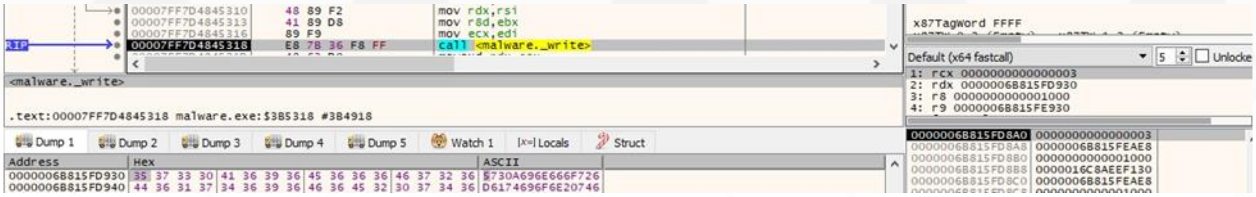
It creates the above file that will be populated:



```
00007FF7D484519A   4C 89 C9              mov rcx,r9                          rcx:L"C:\\Prc
00007FF7D484519D   FF 15 95 2E 1B 00     call qword ptr ds:[<&_wfopen>]
qword ptr [00007FF7D49F8038 <malware.&_wfopen>]=<msvcrt._wfopen>

Default (x64 fastcall)   5   Unlocke
1: rcx 0000016C8AEEBBB0 L"C:\\ProgramData\\ntuser.dat"
2: rdx 0000006B815FEA08
```

Figure 20

Cactus ransomware writes 2 junk strings, the executable path converted to hex, and the AES key passed in the “-i” parameter to the file:



```
00007FF7D4845310   48 89 F2          mov rdx,rsi
00007FF7D4845313   41 89 D8          mov r8d,ebx
00007FF7D4845316   89 F9             mov ecx,edi
RIP  00007FF7D4845318   E8 78 36 F8 FF    call <malware._write>

<malware._write>

.text:00007FF7D4845318 malware.exe:$385318 #384918
```

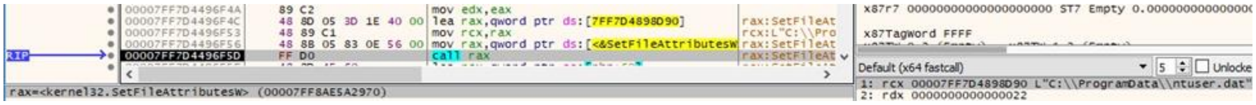
```
x87TagWord FFFF

Default (x64 fastcall)   5   Unlocke
1: rcx 0000000000000003
2: rdx 0000006B815FD930
3: r8 0000000000001000
4: r9 0000006B815FE930
```

```
Address           Hex                                               ASCII
0000006B815FD930  35 37 33 30 41 36 39 36 45 36 36 36 46 37 32 36  5730A696E666F726
0000006B815FD940  44 36 31 37 34 36 39 36 46 36 45 32 30 37 34 36  D6174696F6E20746
```

```
0000006B815FD8A0  0000000000000003
0000006B815FD8A8  0000006B815FEAE8
0000006B815FD8B0  0000000000001000
0000006B815FD8B8  0000016C8AEEF130
0000006B815FD8C0  0000006B815FEAE8
```

Figure 21

The “ntuser.dat” file is hidden via a function call to SetFileAttributesW (Figure 22).

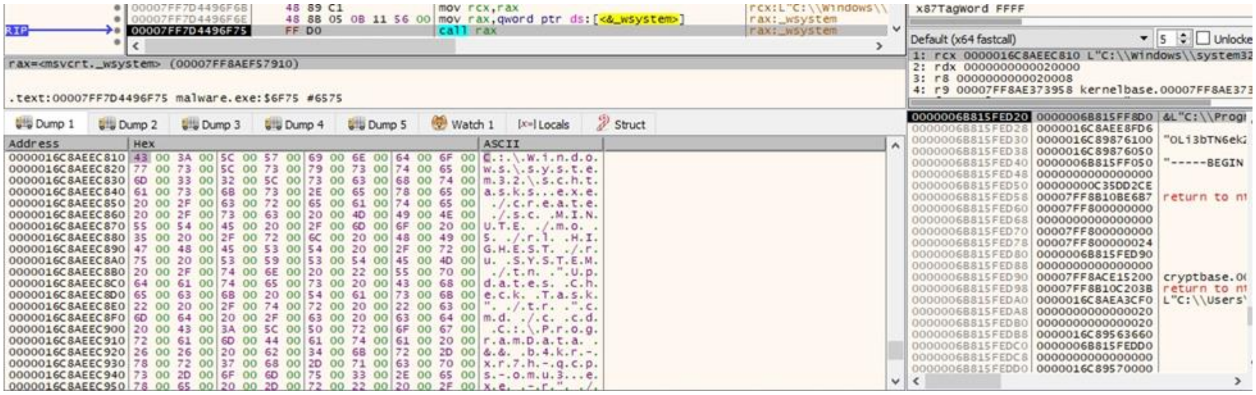


```
00007FF7D4496F4A   89 C2                  mov edx,eax
00007FF7D4496F4C   48 8D 05 3D 1E 40 00   lea rax,qword ptr ds:[7FF7D4898D90]          rax:SetFileAt
00007FF7D4496F53   48 89 C1               mov rcx,rax                                  rcx:L"C:\\Pro
00007FF7D4496F56   48 8B 05 83 0E 56 00   mov rax,qword ptr ds:[<&SetFileAttributesW   rax:SetFileAt
RIP  00007FF7D4496F5D   FF D0             call rax                                     rax:SetFileAt

rax=<kernel32.SetFileAttributesW> (00007FF8AE5A2970)
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000000000

x87TagWord FFFF

Default (x64 fastcall)   5   Unlocke
1: rcx 00007FF7D4898D90 L"C:\\ProgramData\\ntuser.dat"
2: rdx 0000000000000022
```

Figure 22

The ransomware establishes persistence by creating the “Updates Check Task” scheduled task, which runs the malware with the “-r” parameter:



```
00007FF7D4496F6B   48 89 C1               mov rcx,rax                        rcx:L"C:\\Windows\\
00007FF7D4496F6E   48 8B 05 0B 11 56 00   mov rax,qword ptr ds:[<&_wsystem>]  rax:_wsystem
RIP  00007FF7D4496F75   FF D0             call rax                           rax:_wsystem

rax=<msvcrt._wsystem> (00007FF8AEF57910)

.text:00007FF7D4496F75 malware.exe:$6F75 #6575
```

```
x87TagWord FFFF

Default (x64 fastcall)   5   Unlocke
1: rcx 0000016C8AEEC810 L"C:\\Windows\\system32
2: rdx 0000000000020000
3: r8 0000000000020008
4: r9 00007FF8AE373958 kernelbase.00007FF8AE373
```

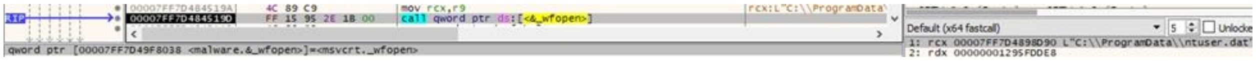
```
Address           Hex                                               ASCII
0000016C8AEEC810  43 00 3A 00 5C 00 57 00 69 00 6E 00 64 00 6F 00  C.:.\.W.i.n.d.o.
0000016C8AEEC820  77 00 73 00 5C 00 73 00 79 00 73 00 74 00 65 00  w.s.\.s.y.s.t.e.
0000016C8AEEC830  6D 00 33 00 32 00 5C 00 73 00 63 00 68 00 74 00  m.3.2.\.s.c.h.t.
0000016C8AEEC840  61 00 73 00 6B 00 73 00 2E 00 65 00 78 00 65 00  a.s.k.s...e.x.e.
0000016C8AEEC850  20 00 2F 00 63 00 72 00 65 00 61 00 74 00 65 00   ./.c.r.e.a.t.e.
0000016C8AEEC860  20 00 2F 00 73 00 63 00 20 00 4D 00 49 00 4E 00   ./.s.c. .M.I.N.
0000016C8AEEC870  55 00 54 00 45 00 20 00 2F 00 6D 00 6F 00 20 00  U.T.E. ./.m.o. .
0000016C8AEEC880  35 00 20 00 2F 00 72 00 6C 00 20 00 48 00 49 00  5. ./.r.l. .H.I.
0000016C8AEEC890  47 00 48 00 45 00 53 00 54 00 20 00 2F 00 72 00  G.H.E.S.T. ./.r.
0000016C8AEEC8A0  75 00 20 00 53 00 59 00 53 00 54 00 45 00 4D 00  u. .S.Y.S.T.E.M.
0000016C8AEEC8B0  20 00 2F 00 74 00 6E 00 20 00 22 00 55 00 70 00   ./.t.n. .".U.p.
0000016C8AEEC8C0  64 00 61 00 74 00 65 00 73 00 20 00 43 00 68 00  d.a.t.e.s. .C.h.
0000016C8AEEC8D0  65 00 63 00 6B 00 20 00 54 00 61 00 73 00 6B 00  e.c.k. .T.a.s.k.
0000016C8AEEC8E0  22 00 20 00 2F 00 74 00 72 00 20 00 22 00 63 00  ". ./.t.r. .".c.
0000016C8AEEC8F0  6D 00 64 00 20 00 2F 00 63 00 20 00 63 00 64 00  m.d. ./.c. .c.d.
0000016C8AEEC900  20 00 43 00 3A 00 5C 00 50 00 72 00 6F 00 67 00   .C.:.\.P.r.o.g.
0000016C8AEEC910  72 00 61 00 6D 00 44 00 61 00 74 00 61 00 20 00  r.a.m.D.a.t.a. .
0000016C8AEEC920  26 00 26 00 20 00 62 00 34 00 6B 00 72 00 2D 00  &.&. .b.4.k.r.-.
0000016C8AEEC930  78 00 72 00 37 00 68 00 2D 00 71 00 63 00 70 00  x.r.7.h.-.q.c.p.
0000016C8AEEC940  73 00 2D 00 6F 00 6D 00 75 00 33 00 2E 00 65 00  s.-.o.m.u.3...e.
0000016C8AEEC950  78 00 65 00 20 00 2D 00 72 00 22 00 20 00 2F 00  x.e. .-.r.". ./.
```

```
0000006B815FED20  0000006B815FF8D0  &L"C:\\Progr
0000006B815FED28  0000016C8AEE8FD6
0000006B815FED30  0000016C89876100  "OLi3bTN6ek
0000006B815FED38  0000016C89876050
0000006B815FED40  0000006B815FF050  "-----BEGIN
0000006B815FED48  0000000000000000
0000006B815FED50  00000000C35DD2CE
0000006B815FED58  00007FF8B10BE6B7  return to nt
0000006B815FED60  00007FF800000000
0000006B815FED68  0000000000000000
0000006B815FED70  00007FF800000000
0000006B815FED78  00007FF800000024
0000006B815FED80  0000006B815FED90
0000006B815FED88  0000000000000000
0000006B815FED90  00007FF8ACE15200  cryptbase.00
0000006B815FED98  00007FF8B10C203B  return to nt
0000006B815FEDA0  0000016C8AEA3CF0  L"C:\\Users\
0000006B815FEDA8  0000000000000020
0000006B815FEDB0  0000000000000020
0000006B815FEDB8  0000016C89563660
0000006B815FEDC0  0000006B815FEDD0
0000006B815FEDC8  0000000000000000
0000006B815FEDD0  0000016C89570000
```

Figure 23

Running with the -r parameter

The wfopen method is utilized to open the file created earlier, as highlighted in the figure below.



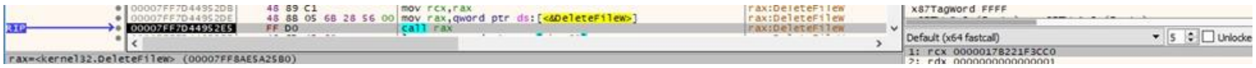
```
00007FF7D484519A   4C 89 C9               mov rcx,r9                          rcx:L"C:\\ProgramData\
RIP  00007FF7D484519D   FF 15 95 2E 18 00  call qword ptr ds:[<&_wfopen>]

qword ptr [00007FF7D49F8038 <malware.&_wfopen>]=<msvcrt._wfopen>
```

```
Default (x64 fastcall)   5   Unlocke
1: rcx 00007FF7D4898D90 L"C:\\ProgramData\\ntuser.dat"
2: rdx 00000001295FDDE8
```

Figure 24

The process extracts the AES key from the file. It’s important to mention that we don’t have the threat actor’s key and performed some modifications that allow the analysis to continue.

The “ntuser.dat” file is deleted afterwards:

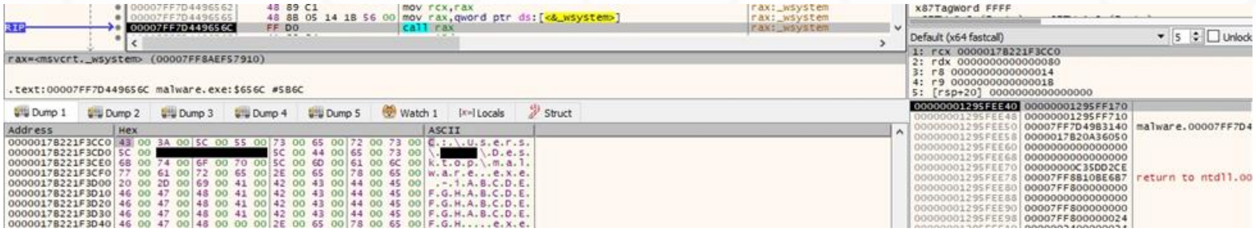


```
00007FF7D44952DB   48 89 C1               mov rcx,rax                        rax:DeleteFileW
00007FF7D44952DE   48 8B 05 6B 28 56 00   mov rax,qword ptr ds:[<&DeleteFileW>]  rax:DeleteFileW
RIP  00007FF7D44952E5   FF D0             call rax                           rax:DeleteFileW

rax=<kernel32.DeleteFileW> (00007FF8AE5A25B0)
```

```
x87TagWord FFFF

Default (x64 fastcall)   5   Unlocke
1: rcx 0000017B221F3CC0
2: rdx 0000000000000001
```

Figure 25

The binary spawns the initial executable with the “-i” parameter, including the AES key that was set to a specific string:



```
00007FF7D4496562 | 48 89 C1             | mov rcx,rax                              | rax:_wsystem
00007FF7D4496565 | 48 8B 05 14 1B 56 00 | mov rax,qword ptr ds:[<&_wsystem>]       | rax:_wsystem
00007FF7D449656C | FF D0                | call rax                                 | rax:_wsystem

rax=<msvcrt._wsystem> (00007FF8AEF57910)

.text:00007FF7D449656C malware.exe:$656C #5B6C
```

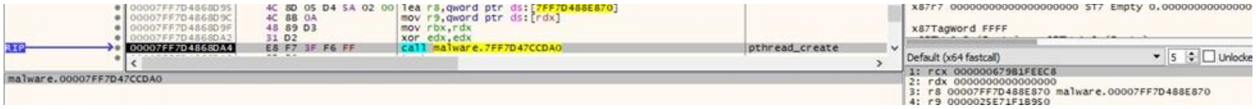
```
Address          | Hex                                             | ASCII
0000017B221F3CC0 | 43 00 3A 00 5C 00 55 00 73 00 65 00 72 00 73 00 | C.:.\.U.s.e.r.s.
0000017B221F3CD0 | 5C 00 [redacted] 5C 00 44 00 65 00 73 00 | \.[redacted]\.D.e.s.
0000017B221F3CE0 | 6B 00 74 00 6F 00 70 00 5C 00 6D 00 61 00 6C 00 | k.t.o.p.\.m.a.l.
0000017B221F3CF0 | 77 00 61 00 72 00 65 00 2E 00 65 00 78 00 65 00 | w.a.r.e...e.x.e.
0000017B221F3D00 | 20 00 2D 00 69 00 41 00 42 00 43 00 44 00 45 00 |  .-.i.A.B.C.D.E.
0000017B221F3D10 | 46 00 47 00 48 00 41 00 42 00 43 00 44 00 45 00 | F.G.H.A.B.C.D.E.
0000017B221F3D20 | 46 00 47 00 48 00 41 00 42 00 43 00 44 00 45 00 | F.G.H.A.B.C.D.E.
0000017B221F3D30 | 46 00 47 00 48 00 41 00 42 00 43 00 44 00 45 00 | F.G.H.A.B.C.D.E.
0000017B221F3D40 | 46 00 47 00 48 00 00 00 2E 00 65 00 78 00 65 00 | F.G.H.....e.x.e.
```

```
x87TagWord FFFF
Default (x64 fastcall)  5  Unlock
1: rcx 0000017B221F3CC0
2: rdx 0000000000000080
3: r8 0000000000000014
4: r9 0000000000000018
5: [rsp+20] 0000000000000000

000000001295FE40  000000001295FF170
000000001295FE48  000000001295FF710
000000001295FE50  00007FF7D49B3140  malware.00007FF7D4
000000001295FE58  0000017B20A36050
000000001295FE60  0000000000000000
000000001295FE68  0000000000000000
000000001295FE70  000000000C35DD2CE
000000001295FE78  00007FF8B10BE6B7  return to ntdll.00
000000001295FE80  00007FF800000000
000000001295FE88  0000000000000000
000000001295FE90  00007FF800000000
000000001295FE98  00007FF800000024
```

Figure 26

Running with the -i parameter

The executable creates a new thread that runs the searchFilesThreadControl function:

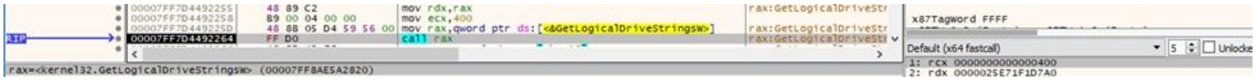


```
00007FF7D4868D95 | 4C 8D 05 D4 5A 02 00 | lea r8,qword ptr ds:[7FF7D488E870]  |
00007FF7D4868D9C | 4C 8B 0A             | mov r9,qword ptr ds:[rdx]           |
00007FF7D4868D9F | 48 89 D3             | mov rbx,rdx                         |
00007FF7D4868DA2 | 31 D2                | xor edx,edx                         |
00007FF7D4868DA4 | E8 F7 3F F6 FF       | call malware.7FF7D47CCDA0           | pthread_create

malware.00007FF7D47CCDA0
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000000000
x87TagWord FFFF
Default (x64 fastcall)  5  Unlocke
1: rcx 00000067981FEEC8
2: rdx 0000000000000000
3: r8 00007FF7D488E870 malware.00007FF7D488E870
4: r9 0000025E71F18950
```

Figure 27

It retrieves the valid drives on the system using the GetLogicalDriveStringsW API (see Figure 28).

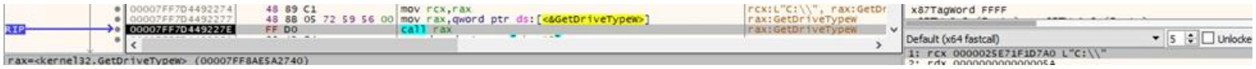


```
00007FF7D4492255 | 48 89 C2             | mov rdx,rax                                   | rax:GetLogicalDriveStr
00007FF7D4492258 | B9 00 04 00 00       | mov ecx,400                                   |
00007FF7D449225D | 48 8B 05 D4 59 56 00 | mov rax,qword ptr ds:[<&GetLogicalDriveStringsW>] | rax:GetLogicalDriveStr
00007FF7D4492264 | FF D0                | call rax                                      | rax:GetLogicalDriveStr

rax=<kernel32.GetLogicalDriveStringsW> (00007FF8AE5A2820)
```

```
x87TagWord FFFF
Default (x64 fastcall)  5  Unlocke
1: rcx 0000000000000400
2: rdx 0000025E71F1D7A0
```

Figure 28

The malware obtains the type of the drive by calling the GetDriveTypeW function:

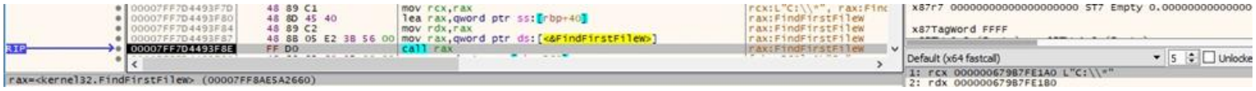


```
00007FF7D4492274 | 48 89 C1             | mov rcx,rax                           | rcx:L"C:\\", rax:GetDr
00007FF7D4492277 | 48 8B 05 72 59 56 00 | mov rax,qword ptr ds:[<&GetDriveTypeW>] | rax:GetDriveTypeW
00007FF7D449227E | FF D0                | call rax                              | rax:GetDriveTypeW

rax=<kernel32.GetDriveTypeW> (00007FF8AE5A2740)
```

```
x87TagWord FFFF
Default (x64 fastcall)  5  Unlocke
1: rcx 0000025E71F1D7A0 L"C:\\"
2: rdx 000000000000005A
```

Figure 29

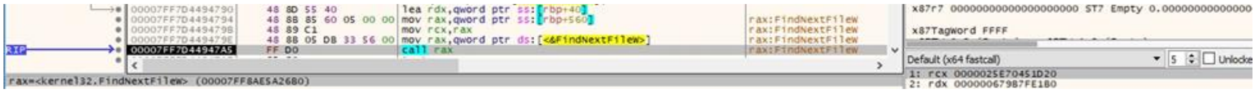
The files are enumerated using the FindFirstFileW and FindNextFileW APIs:



```
00007FF7D4493F7D | 48 89 C1             | mov rcx,rax                              | rcx:L"C:\\*", rax:Finc
00007FF7D4493F80 | 48 8D 45 40          | lea rax,qword ptr ss:[rbp+40]            | rax:FindFirstFileW
00007FF7D4493F84 | 48 89 C2             | mov rdx,rax                              | rax:FindFirstFileW
00007FF7D4493F87 | 48 8B 05 E2 3B 56 00 | mov rax,qword ptr ds:[<&FindFirstFileW>] | rax:FindFirstFileW
00007FF7D4493F8E | FF D0                | call rax                                 | rax:FindFirstFileW

rax=<kernel32.FindFirstFileW> (00007FF8AE5A2660)
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000000000
x87TagWord FFFF
Default (x64 fastcall)  5  Unlocke
1: rcx 00000067987FE1A0 L"C:\\*"
2: rdx 00000067987FE1B0
```

Figure 30



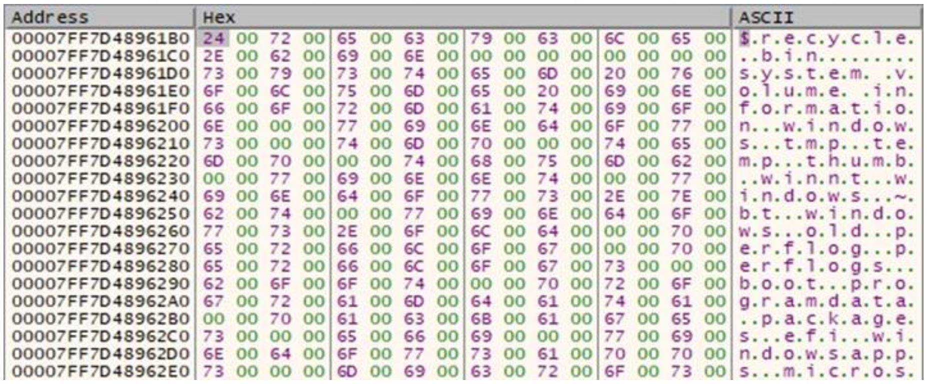
```
00007FF7D4494790 | 48 8D 55 40          | lea rdx,qword ptr ss:[rbp+40]           |
00007FF7D4494794 | 48 8B 85 60 05 00 00 | mov rax,qword ptr ss:[rbp+560]          | rax:FindNextFileW
00007FF7D449479B | 48 89 C1             | mov rcx,rax                             | rax:FindNextFileW
00007FF7D449479E | 48 8B 05 DB 33 56 00 | mov rax,qword ptr ds:[<&FindNextFileW>] | rax:FindNextFileW
00007FF7D44947A5 | FF D0                | call rax                                | rax:FindNextFileW

rax=<kernel32.FindNextFileW> (00007FF8AE5A2680)
```

```
x87r7 00000000000000000000 ST7 Empty 0.00000000000000
x87TagWord FFFF
Default (x64 fastcall)  5  Unlocke
1: rcx 0000025E70451D20
2: rdx 00000067987FE1B0
```

Figure 31

The following directories will not be encrypted:

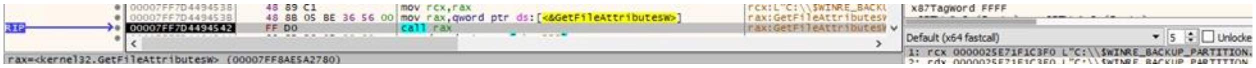
- "$recycle.bin" "system volume information" "windows" "tmp" "temp" "thumb" "winnt" "windows.~bt" "windows.old" "perflog" "perflogs" "boot" "programdata" "packages" "efi" "windowsapps" "microsoft" "windows defender" "microsoft shared" "internet explorer" "tor browser" "ctslck"



```
Address           Hex                                               ASCII
00007FF7D48961B0  24 00 72 00 65 00 63 00 79 00 63 00 6C 00 65 00  $.r.e.c.y.c.l.e.
00007FF7D48961C0  2E 00 62 00 69 00 6E 00 00 00 00 00 00 00 00 00  ..b.i.n.........
00007FF7D48961D0  73 00 79 00 73 00 74 00 65 00 6D 00 20 00 76 00  s.y.s.t.e.m. .v.
00007FF7D48961E0  6F 00 6C 00 75 00 6D 00 65 00 20 00 69 00 6E 00  o.l.u.m.e. .i.n.
00007FF7D48961F0  66 00 6F 00 72 00 6D 00 61 00 74 00 69 00 6F 00  f.o.r.m.a.t.i.o.
00007FF7D4896200  6E 00 00 00 77 00 69 00 6E 00 64 00 6F 00 77 00  n...w.i.n.d.o.w.
00007FF7D4896210  73 00 00 00 74 00 6D 00 70 00 00 00 74 00 65 00  s...t.m.p...t.e.
00007FF7D4896220  6D 00 70 00 00 00 74 00 68 00 75 00 6D 00 62 00  m.p...t.h.u.m.b.
00007FF7D4896230  00 00 77 00 69 00 6E 00 6E 00 74 00 00 00 77 00  ..w.i.n.n.t...w.
00007FF7D4896240  69 00 6E 00 64 00 6F 00 77 00 73 00 2E 00 7E 00  i.n.d.o.w.s...~.
00007FF7D4896250  62 00 74 00 00 00 77 00 69 00 6E 00 64 00 6F 00  b.t...w.i.n.d.o.
00007FF7D4896260  77 00 73 00 2E 00 6F 00 6C 00 64 00 00 00 70 00  w.s...o.l.d...p.
00007FF7D4896270  65 00 72 00 66 00 6C 00 6F 00 67 00 00 00 70 00  e.r.f.l.o.g...p.
00007FF7D4896280  65 00 72 00 66 00 6C 00 6F 00 67 00 73 00 00 00  e.r.f.l.o.g.s...
00007FF7D4896290  62 00 6F 00 6F 00 74 00 00 00 70 00 72 00 6F 00  b.o.o.t...p.r.o.
00007FF7D48962A0  67 00 72 00 61 00 6D 00 64 00 61 00 74 00 61 00  g.r.a.m.d.a.t.a.
00007FF7D48962B0  00 00 70 00 61 00 63 00 6B 00 61 00 67 00 65 00  ..p.a.c.k.a.g.e.
00007FF7D48962C0  73 00 00 00 65 00 66 00 69 00 00 00 77 00 69 00  s...e.f.i...w.i.
00007FF7D48962D0  6E 00 64 00 6F 00 77 00 73 00 61 00 70 00 70 00  n.d.o.w.s.a.p.p.
00007FF7D48962E0  73 00 00 00 6D 00 69 00 63 00 72 00 6F 00 73 00  s...m.i.c.r.o.s.
```

Figure 32

GetFileAttributesW is used to extract file system attributes for a target file, as shown below:

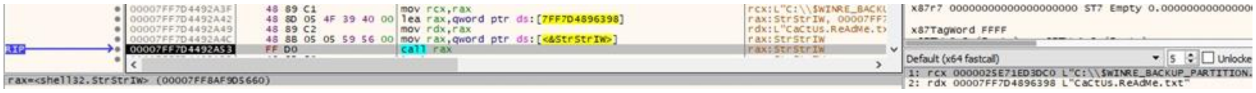


```
    00007FF7D4494538  48 89 C1              mov rcx,rax                                   rcx:L"C:\\$WINRE_BACKU
    00007FF7D449453B  48 8B 05 BE 36 56 00  mov rax,qword ptr ds:[<&GetFileAttributesW>]  rax:GetFileAttributesW
RIP 00007FF7D4494542  FF D0                 call rax                                      rax:GetFileAttributesW
rax=<kernel32.GetFileAttributesW> (00007FF8AE5A2780)

x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 0000025E71F1C3F0 L"C:\\$WINRE_BACKUP_PARTITION.
2: rdx 0000025E71F1C3F0 L"C:\\$WINRE_BACKUP_PARTITION
```

Figure 33

Cactus ransomware doesn’t encrypt the “CaCtUs.ReAdMe.txt” ransom note and the following files:

- "desktop.ini" "update.log" "ntuser.dat”



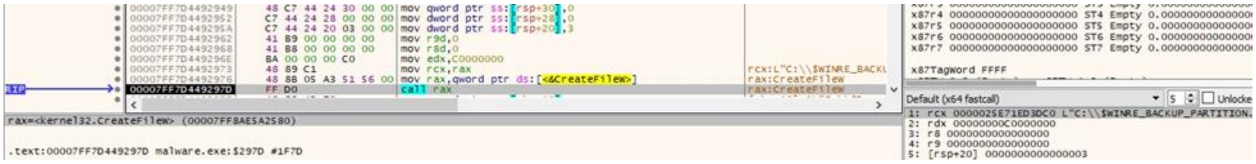
```
    00007FF7D4492A3F  48 89 C1              mov rcx,rax                            rcx:L"C:\\$WINRE_BACKU
    00007FF7D4492A42  48 8D 05 4F 39 40 00  lea rax,qword ptr ds:[7FF7D4896398]    rax:StrStrIW, 00007FF7
    00007FF7D4492A49  48 89 C2              mov rdx,rax                            rdx:L"CaCtUs.ReAdMe.tx
    00007FF7D4492A4C  48 8B 05 05 59 56 00  mov rax,qword ptr ds:[<&StrStrIW>]     rax:StrStrIW
RIP 00007FF7D4492A53  FF D0                 call rax                               rax:StrStrIW
rax=<shell32.StrStrIW> (00007FF8AF9D5660)

x87r7 00000000000000000000 ST7 Empty 0.000000000000000
x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 0000025E71ED3DC0 L"C:\\$WINRE_BACKUP_PARTITION.
2: rdx 00007FF7D4896398 L"CaCtUs.ReAdMe.txt"
```

Figure 34

The following file extensions will be avoided:

- "exe" "dll" "lnk" "sys" "msi" "bat" "cts0" "cts1"

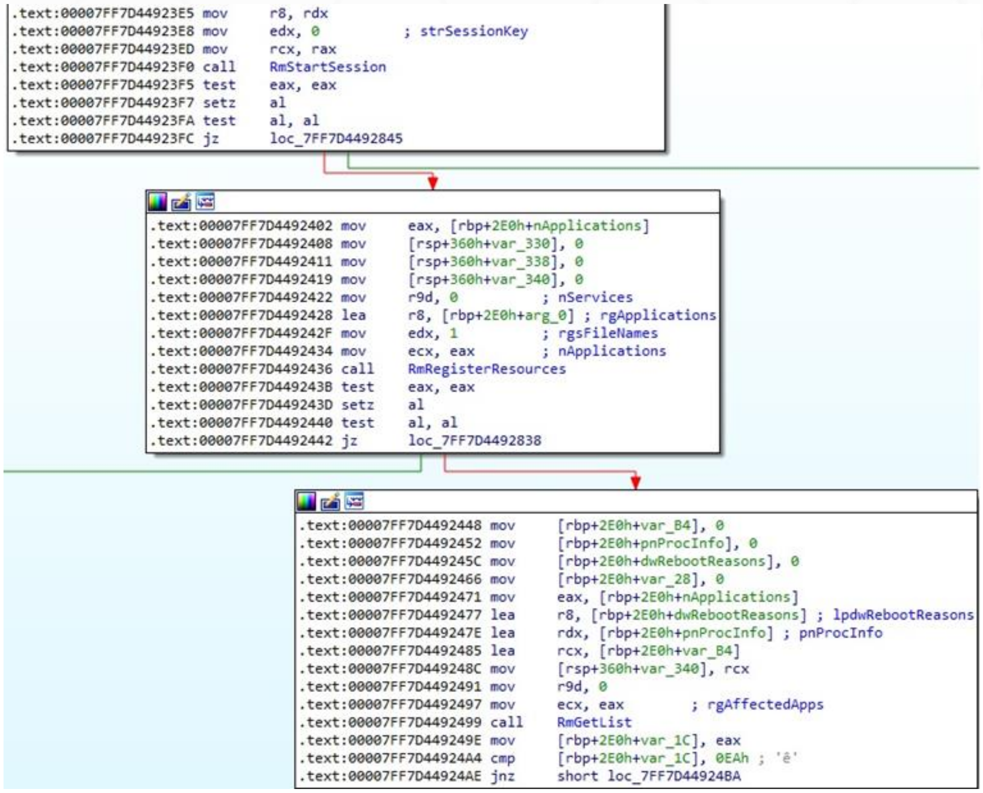
The ransomware opens the target file using CreateFileW (0xC0000000 = **GENERIC_READ** | **GENERIC_WRITE**, 0x3 = **OPEN_EXISTING**):



```
    00007FF7D4492949  48 C7 44 24 30 00 00  mov qword ptr ss:[rsp+30],0
    00007FF7D4492952  C7 44 24 28 00 00 00  mov dword ptr ss:[rsp+28],0
    00007FF7D449295A  C7 44 24 20 03 00 00  mov dword ptr ss:[rsp+20],3
    00007FF7D4492962  41 B9 00 00 00 00     mov r9d,0
    00007FF7D4492968  41 B8 00 00 00 00     mov r8d,0
    00007FF7D449296E  BA 00 00 00 C0        mov edx,C0000000
    00007FF7D4492973  48 89 C1              mov rcx,rax                              rcx:L"C:\\$WINRE_BACKU
    00007FF7D4492976  48 8B 05 A3 51 56 00  mov rax,qword ptr ds:[<&CreateFileW>]    rax:CreateFileW
RIP 00007FF7D449297D  FF D0                 call rax                                 rax:CreateFileW
rax=<kernel32.CreateFileW> (00007FF8AE5A2580)

.text:00007FF7D449297D malware.exe:$297D #1F7D

x87r4 00000000000000000000 ST4 Empty 0.000000000000000
x87r5 00000000000000000000 ST5 Empty 0.000000000000000
x87r6 00000000000000000000 ST6 Empty 0.000000000000000
x87r7 00000000000000000000 ST7 Empty 0.000000000000000
x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 0000025E71ED3DC0 L"C:\\$WINRE_BACKUP_PARTITION.
2: rdx 00000000C0000000
3: r8 0000000000000000
4: r9 0000000000000000
5: [rsp+20] 0000000000000003
```

Figure 35

The binary uses Restart Manager APIs to determine if the target files are blocked by other processes (Figure 36).



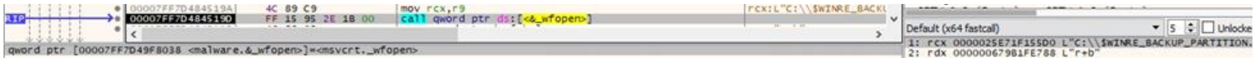
```
.text:00007FF7D44923E5 mov     r8, rdx
.text:00007FF7D44923E8 mov     edx, 0          ; strSessionKey
.text:00007FF7D44923ED mov     rcx, rax
.text:00007FF7D44923F0 call    RmStartSession
.text:00007FF7D44923F5 test    eax, eax
.text:00007FF7D44923F7 setz    al
.text:00007FF7D44923FA test    al, al
.text:00007FF7D44923FC jz      loc_7FF7D4492845
```

```
.text:00007FF7D4492402 mov     eax, [rbp+2E0h+nApplications]
.text:00007FF7D4492408 mov     [rsp+360h+var_330], 0
.text:00007FF7D4492411 mov     [rsp+360h+var_338], 0
.text:00007FF7D4492419 mov     [rsp+360h+var_340], 0
.text:00007FF7D4492422 mov     r9d, 0          ; nServices
.text:00007FF7D4492428 lea     r8, [rbp+2E0h+arg_0] ; rgApplications
.text:00007FF7D449242F mov     edx, 1          ; rgsFileNames
.text:00007FF7D4492434 mov     ecx, eax        ; nApplications
.text:00007FF7D4492436 call    RmRegisterResources
.text:00007FF7D449243B test    eax, eax
.text:00007FF7D449243D setz    al
.text:00007FF7D4492440 test    al, al
.text:00007FF7D4492442 jz      loc_7FF7D4492838
```

```
.text:00007FF7D4492448 mov     [rbp+2E0h+var_B4], 0
.text:00007FF7D4492452 mov     [rbp+2E0h+pnProcInfo], 0
.text:00007FF7D449245C mov     [rbp+2E0h+dwRebootReasons], 0
.text:00007FF7D4492466 mov     [rbp+2E0h+var_28], 0
.text:00007FF7D4492471 mov     eax, [rbp+2E0h+nApplications]
.text:00007FF7D4492477 lea     r8, [rbp+2E0h+dwRebootReasons] ; lpdwRebootReasons
.text:00007FF7D449247E lea     rdx, [rbp+2E0h+pnProcInfo] ; pnProcInfo
.text:00007FF7D4492485 lea     rcx, [rbp+2E0h+var_B4]
.text:00007FF7D449248C mov     [rsp+360h+var_340], rcx
.text:00007FF7D4492491 mov     r9d, 0
.text:00007FF7D4492497 mov     ecx, eax        ; rgAffectedApps
.text:00007FF7D4492499 call    RmGetList
.text:00007FF7D449249E mov     [rbp+2E0h+var_1C], eax
.text:00007FF7D44924A4 cmp     [rbp+2E0h+var_1C], 0EAh ; 'ê'
.text:00007FF7D44924AE jnz     short loc_7FF7D44924BA
```

Figure 36

The wfopen function is utilized to open the file:

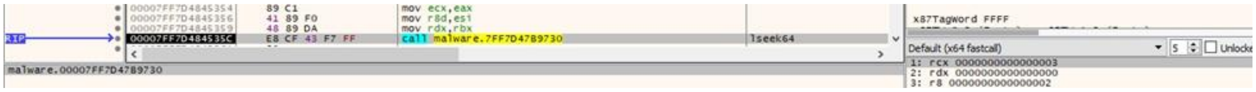


```
     00007FF7D484519A   4C 89 C9            mov rcx,r9                           rcx:L"C:\\$WINRE_BACK
RIP  00007FF7D484519D   FF 15 95 2E 18 00   call qword ptr ds:[<&_wfopen>]
qword ptr [00007FF7D49F8038 <malware.&_wfopen>]=<msvcrt._wfopen>
```

```
Default (x64 fastcall)   5   Unlocke
1: rcx 0000025E71F155D0 L"C:\\$WINRE_BACKUP_PARTITION.
2: rdx 000000679B1FE788 L"r+b"
```

Figure 37

The malicious process moves the file pointer to the end of file using lseek64 (0x2 = **SEEK_END**):

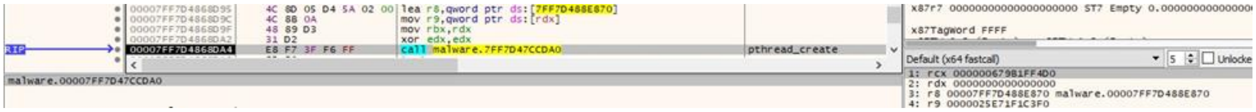


```
     00007FF7D4845354   89 C1               mov ecx,eax
     00007FF7D4845356   41 89 F0            mov r8d,esi
     00007FF7D4845359   48 89 DA            mov rdx,rbx
RIP  00007FF7D484535C   E8 CF 43 F7 FF      call malware.7FF7D47B9730            lseek64
malware.00007FF7D47B9730
```

```
x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 0000000000000003
2: rdx 0000000000000000
3: r8 0000000000000002
```

Figure 38

It creates a new thread that handles the file's encryption (see Figure 39).



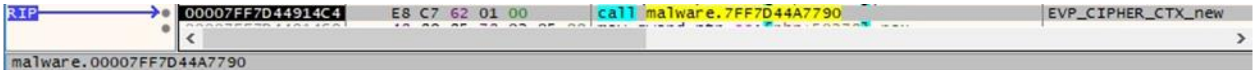
```
     00007FF7D4868D95   4C 8D 05 D4 5A 02 00   lea r8,qword ptr ds:[7FF7D488E870]
     00007FF7D4868D9C   4C 8B 0A               mov r9,qword ptr ds:[rdx]
     00007FF7D4868D9F   48 89 D3               mov rbx,rdx
     00007FF7D4868DA2   31 D2                  xor edx,edx
RIP  00007FF7D4868DA4   E8 F7 3F F6 FF         call malware.7FF7D47CCDA0         pthread_create
malware.00007FF7D47CCDA0
```

```
x87r7 00000000000000000000 ST7 Empty 0.000000000000000
x87TagWord FFFF
Default (x64 fastcall)   5   Unlocke
1: rcx 000000679B1FF4D0
2: rdx 0000000000000000
3: r8 00007FF7D488E870 malware.00007FF7D488E870
4: r9 0000025E71F1C3F0
```

Figure 39

The file's size is compared with 8074034 bytes (approximately 7.7MB). If the size is greater than 7.7MB, then cryptPartFile is called; otherwise, the malware calls the cryptFullFile function. Basically, a large file is partially encrypted (50%, but the percentage can be modified) by Cactus ransomware.

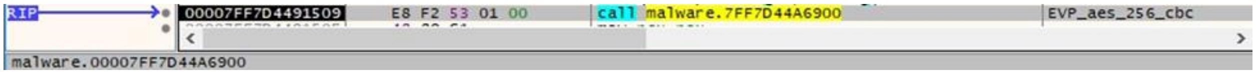
The ransomware allocates and obtains a cipher context using OpenSSL's EVP_CIPHER_CTX_new:



```
RIP  00007FF7D44914C4   E8 C7 62 01 00   call malware.7FF7D44A7790   EVP_CIPHER_CTX_new
malware.00007FF7D44A7790
```

Figure 40

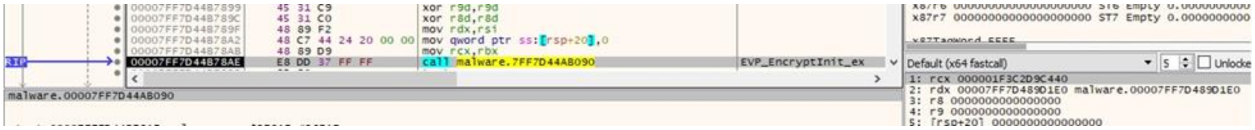
The algorithm used to encrypt the files is AES256 in CBC mode, as highlighted in the figure below.



```
RIP  00007FF7D4491509   E8 F2 53 01 00   call malware.7FF7D44A6900   EVP_aes_256_cbc
malware.00007FF7D44A6900
```

Figure 41

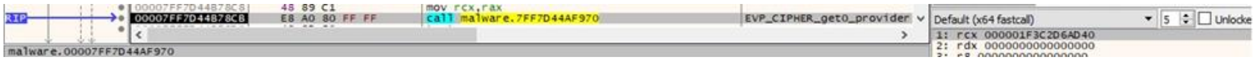
The process sets up the cipher context for encryption using the EVP_EncryptInit_ex method (Figure 42).



```
     00007FF7D4487899   45 31 C9               xor r9d,r9d
     00007FF7D448789C   45 31 C0               xor r8d,r8d
     00007FF7D448789F   48 89 F2               mov rdx,rsi
     00007FF7D44878A2   48 C7 44 24 20 00 00   mov qword ptr ss:[rsp+20],0
     00007FF7D44878AB   48 89 D9               mov rcx,rbx
RIP  00007FF7D44878AE   E8 DD 37 FF FF         call malware.7FF7D44AB090   EVP_EncryptInit_ex
malware.00007FF7D44AB090

x87r6 00000000000000000000 ST6 Empty 0.0000000000
x87r7 00000000000000000000 ST7 Empty 0.0000000000
Default (x64 fastcall)   5   Unlocke
1: rcx 000001F3C2D9C440
2: rdx 00007FF7D489D1E0 malware.00007FF7D489D1E0
3: r8 0000000000000000
4: r9 0000000000000000
5: [rsp+20] 0000000000000000
```

Figure 42

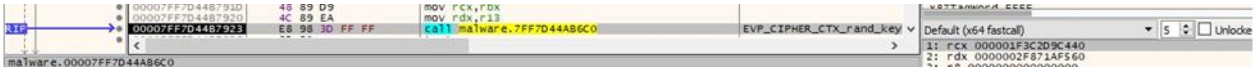
EVP_CIPHER_get0_provider is utilized to obtain an OSSL_PROVIDER pointer to the provider:



```
     00007FF7D44878C8   48 89 C1         mov rcx,rax
RIP  00007FF7D44878CB   E8 A0 80 FF FF   call malware.7FF7D44AF970   EVP_CIPHER_get0_provider
malware.00007FF7D44AF970

Default (x64 fastcall)   5   Unlocke
1: rcx 000001F3C2D6AD40
2: rdx 0000000000000000
```

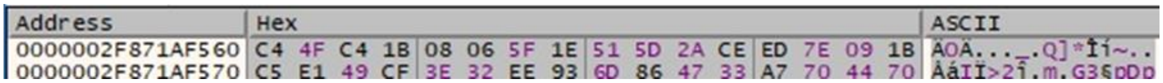
Figure 43

The ransomware generates a random 32-byte key using EVP_CIPHER_CTX_rand_key:



```
     00007FF7D448791D   48 89 D9         mov rcx,rbx
     00007FF7D4487920   4C 89 EA         mov rdx,r13
RIP  00007FF7D4487923   E8 98 3D FF FF   call malware.7FF7D44AB6C0   EVP_CIPHER_CTX_rand_key
malware.00007FF7D44AB6C0

Default (x64 fastcall)   5   Unlocke
1: rcx 000001F3C2D9C440
2: rdx 0000002F871AF560
```

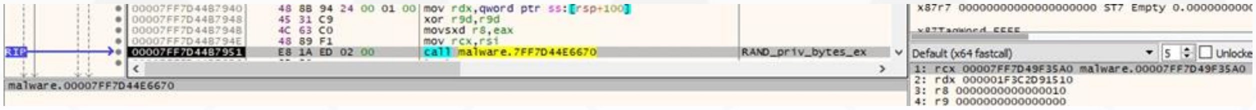
Figure 44



```
Address            Hex                                                ASCII
0000002F871AF560   C4 4F C4 1B 08 06 5F 1E 51 5D 2A CE ED 7E 09 1B   ÄOÄ..._.Q]*Îí~..
0000002F871AF570   C5 E1 49 CF 3E 32 EE 93 6D 86 47 33 A7 70 44 70   ÅáIÏ>2î.m.G3§pDp
```

Figure 45

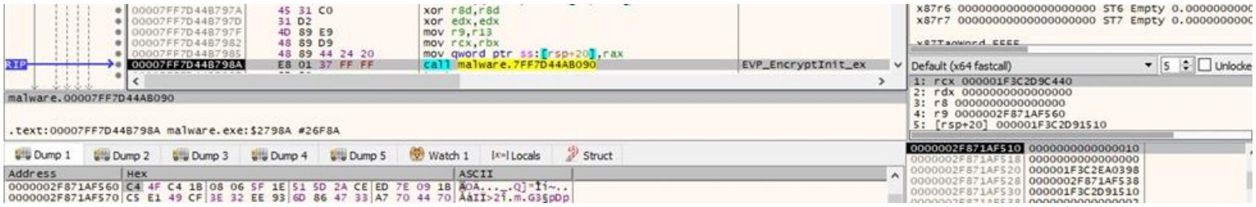
The 16-byte IV is generated by calling the RAND_priv_bytes_ex function:



```
00007FF7D4487940 | 48 8B 94 24 00 01 00 | mov rdx,qword ptr ss:[rsp+100]
00007FF7D4487948 | 45 31 C9             | xor r9d,r9d
00007FF7D448794B | 4C 63 C0             | movsxd r8,eax
00007FF7D448794E | 48 89 F1             | mov rcx,rsi
00007FF7D4487951 | E8 1A ED 02 00       | call malware.7FF7D44E6670        RAND_priv_bytes_ex
malware.00007FF7D44E6670

x87r7 00000000000000000000 ST7 Empty 0.000000000
Default (x64 fastcall)  5  Unlocke
1: rcx 00007FF7D49F35A0 malware.00007FF7D49F35A0
2: rdx 000001F3C2D91510
3: r8 0000000000000010
4: r9 0000000000000000
```

Figure 46

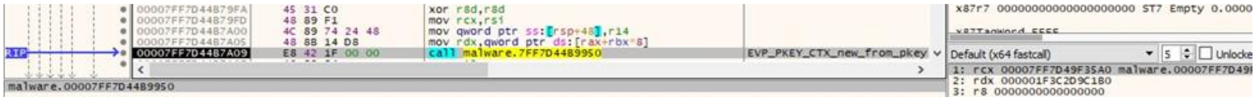
Using the key and IV previously generated, the binary calls the EVP_EncryptInit_ex method again:



```
00007FF7D448797A | 45 31 C0             | xor r8d,r8d
00007FF7D448797D | 31 D2                | xor edx,edx
00007FF7D448797F | 4D 89 E9             | mov r9,r13
00007FF7D4487982 | 48 89 D9             | mov rcx,rbx
00007FF7D4487985 | 48 89 44 24 20       | mov qword ptr ss:[rsp+20],rax
00007FF7D448798A | E8 01 37 FF FF       | call malware.7FF7D44AB090        EVP_EncryptInit_ex
malware.00007FF7D44AB090

.text:00007FF7D448798A malware.exe:$2798A #26F8A

Address          | Hex                                             | ASCII
0000002F871AF560 | C4 4F C4 1B 08 06 5F 1E 51 5D 2A CE ED 7E 09 18 | ÄOA...._.Q]*Îî~..
0000002F871AF570 | C5 E1 49 CF 3E 32 EE 93 6D 86 47 33 A7 70 44 70 | ÅáIÏ>2î.m.G3§pDp

x87r6 00000000000000000000 ST6 Empty 0.000000000
x87r7 00000000000000000000 ST7 Empty 0.000000000
Default (x64 fastcall)  5  Unlocke
1: rcx 000001F3C2D9C440
2: rdx 0000000000000000
3: r8 0000000000000000
4: r9 0000002F871AF560
5: [rsp+20] 000001F3C2D91510

0000002F871AF510  0000000000000010
0000002F871AF518  0000000000000000
0000002F871AF520  000001F3C2EA0398
0000002F871AF528  0000002F871AF538
0000002F871AF530  000001F3C2D91510
```

Figure 47

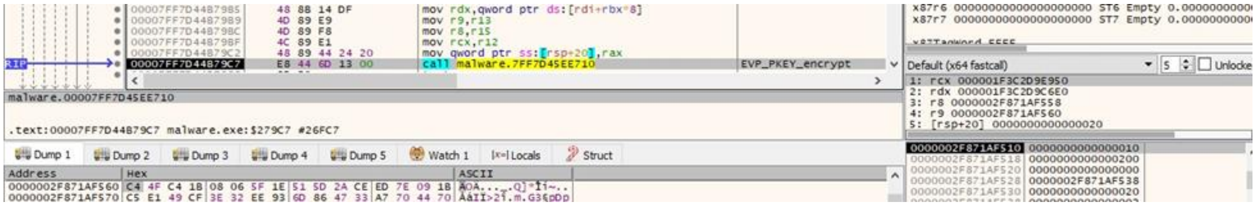
The executable allocates the public RSA key algorithm context via a call to EVP_PKEY_CTX_new_from_pkey, as highlighted in the figure below.



```
00007FF7D44879FA | 45 31 C0             | xor r8d,r8d
00007FF7D44879FD | 48 89 F1             | mov rcx,rsi
00007FF7D4487A00 | 4C 89 74 24 48       | mov qword ptr ss:[rsp+48],r14
00007FF7D4487A05 | 48 8B 14 D8          | mov rdx,qword ptr ds:[rax+rbx*8]
00007FF7D4487A09 | E8 42 1F 00 00       | call malware.7FF7D44B9950        EVP_PKEY_CTX_new_from_pkey
malware.00007FF7D44B9950

x87r7 00000000000000000000 ST7 Empty 0.0000
Default (x64 fastcall)  5  Unlocke
1: rcx 00007FF7D49F35A0 malware.00007FF7D49
2: rdx 000001F3C2D9C1B0
3: r8 0000000000000000
```

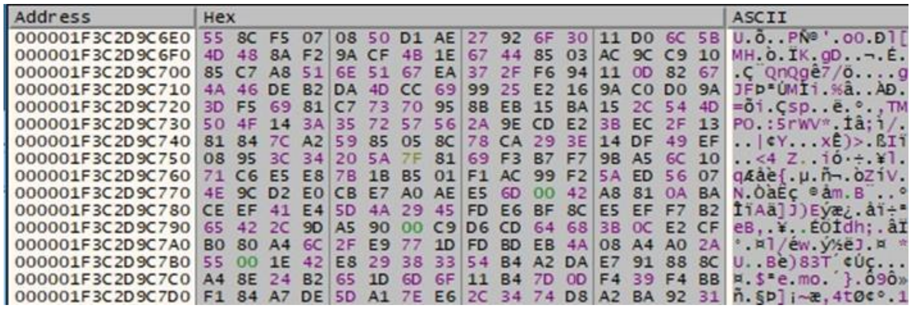
Figure 48

The AES256 key is encrypted using the public key:



```
00007FF7D4487985 | 48 8B 14 DF          | mov rdx,qword ptr ds:[rdi+rbx*8]
00007FF7D4487989 | 4D 89 E9             | mov r9,r13
00007FF7D448798C | 4D 89 F8             | mov r8,r15
00007FF7D448798F | 4C 89 E1             | mov rcx,r12
00007FF7D44879C2 | 48 89 44 24 20       | mov qword ptr ss:[rsp+20],rax
00007FF7D44879C7 | E8 44 6D 13 00       | call malware.7FF7D45EE710        EVP_PKEY_encrypt
malware.00007FF7D45EE710

.text:00007FF7D44879C7 malware.exe:$279C7 #26FC7

Address          | Hex                                             | ASCII
0000002F871AF560 | C4 4F C4 1B 08 06 5F 1E 51 5D 2A CE ED 7E 09 18 | ÄOA...._.Q]*Îî~..
0000002F871AF570 | C5 E1 49 CF 3E 32 EE 93 6D 86 47 33 A7 70 44 70 | ÅáIÏ>2î.m.G3§pDp

x87r6 00000000000000000000 ST6 Empty 0.000000000
x87r7 00000000000000000000 ST7 Empty 0.000000000
Default (x64 fastcall)  5  Unlocke
1: rcx 000001F3C2D9E950
2: rdx 000001F3C2D9C6E0
3: r8 0000002F871AF558
4: r9 0000002F871AF560
5: [rsp+20] 0000000000000020

0000002F871AF510  0000000000000010
0000002F871AF518  0000000000000200
0000002F871AF520  0000000000000000
0000002F871AF528  0000002F871AF538
0000002F871AF530  0000000000000020
```

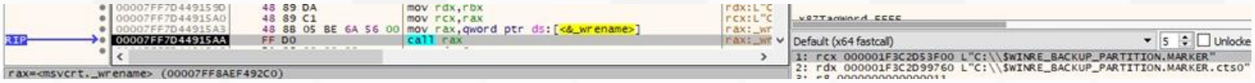
Figure 49



```
Address          | Hex                                             | ASCII
000001F3C2D9C6E0 | 55 8C F5 07 08 50 D1 AE 27 92 6F 30 11 D0 6C 5B | U.õ..PÑ®'.o0.Ðl[
000001F3C2D9C6F0 | 4D 48 8A F2 9A CF 4B 1E 67 44 85 03 AC 9C C9 10 | MH.ò.ÏK.gD..¬.É.
000001F3C2D9C700 | 85 C7 A8 51 6E 51 67 EA 37 2F F6 94 11 0D 82 67 | .Ç¨QnQgê7/ö....g
000001F3C2D9C710 | 4A 46 DE B2 DA 4D CC 69 99 25 E2 16 9A C0 D0 9A | JFÞ²ÚMÌi.%â..ÀÐ.
000001F3C2D9C720 | 3D F5 69 81 C7 73 70 95 8B EB 15 BA 15 2C 54 4D | =õi.Çsp..ë.º.,TM
000001F3C2D9C730 | 50 4F 14 3A 35 72 57 56 2A 9E CD E2 3B EC 2F 13 | PO.:5rWV*.Íâ;ì/.
000001F3C2D9C740 | 81 84 7C A2 59 85 05 8C 78 CA 29 3E 14 DF 49 EF | ..|¢Y...xÊ)>.ßIï
000001F3C2D9C750 | 08 95 3C 34 20 5A 7F 81 69 F3 B7 F7 9B A5 6C 10 | ..<4 Z..ió·÷.¥l.
000001F3C2D9C760 | 71 C6 E5 E8 7B 1B B5 01 F1 AC 99 F2 5A ED 56 07 | qÆåè{.µ.ñ¬.òZíV.
000001F3C2D9C770 | 4E 9C D2 E0 CB E7 A0 AE E5 6D 00 42 A8 81 0A BA | N.ÒàËç ®åm.B¨..º
000001F3C2D9C780 | CE EF 41 E4 5D 4A 29 45 FD E6 BF 8C E5 EF F7 B2 | ÎïAä]J)Eýæ¿.åï÷²
000001F3C2D9C790 | 65 42 2C 9D A5 90 00 C9 D6 CD 64 68 3B 0C E2 CF | eB,.¥..ÉÖÍdh;.âÏ
000001F3C2D9C7A0 | B0 80 A4 6C 2F E9 77 1D FD BD EB 4A 08 A4 A0 2A | °.¤l/éw.ý½ëJ.¤ *
000001F3C2D9C7B0 | 55 00 1E 42 E8 29 38 33 54 B4 A2 DA E7 91 88 8C | U..Bè)83T´¢Úç...
000001F3C2D9C7C0 | A4 8E 24 B2 65 1D 6D 6F 11 B4 7D 0D F4 39 F4 BB | ¤.$²e.mo.´}.ô9ô»
000001F3C2D9C7D0 | F1 84 A7 DE 5D A1 7E E6 2C 34 74 D8 A2 BA 92 31 | ñ.§Þ]¡~æ,4tØ¢º.1
```

Figure 50

The encrypted file's extension is changed to "cts0" or "cts1":

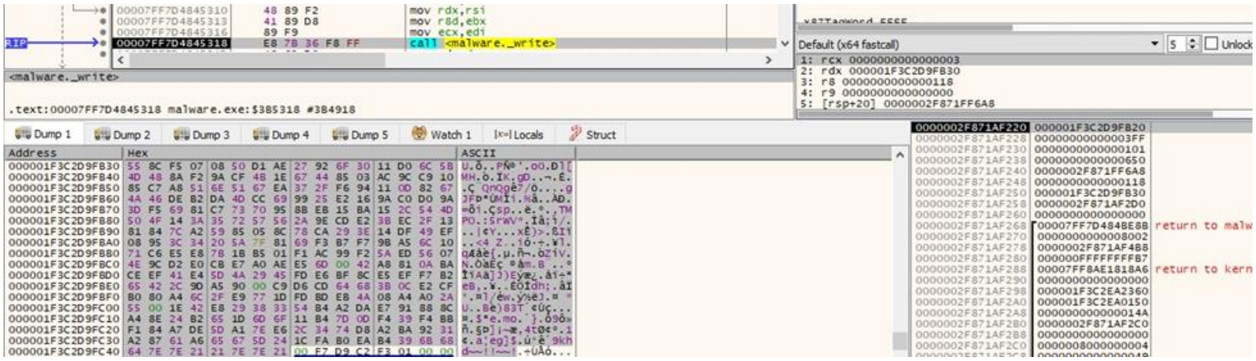


```
00007FF7D449159D   48 89 DA              mov rdx,rbx
00007FF7D44915A0   48 89 C1              mov rcx,rax
00007FF7D44915A3   48 8B 05 BE 6A 56 00  mov rax,qword ptr ds:[<&_wrename>]
00007FF7D44915AA   FF D0                 call rax
rax=<msvcrt._wrename> (00007FF8AEF492C0)
```

```
Default (x64 fastcall)
1: rcx 000001F3C2D53F00 L"C:\\$WINRE_BACKUP_PARTITION.MARKER"
2: rdx 000001F3C2D99760 L"C:\\$WINRE_BACKUP_PARTITION.MARKER.ctso"
```

Figure 51

The ransomware appends the following information to the encrypted file: encrypted AES256 key, non-encrypted IV, 0x64 (encryption percentage), and "~~!!~~!".



```
00007FF7D4845310   48 89 F2         mov rdx,rsi
00007FF7D4845313   41 89 D8         mov r8d,ebx
00007FF7D4845316   89 F9            mov ecx,edi
00007FF7D4845318   E8 7B 36 F8 FF   call <malware._write>
<malware._write>

.text:00007FF7D4845318 malware.exe:$385318 #384918
```

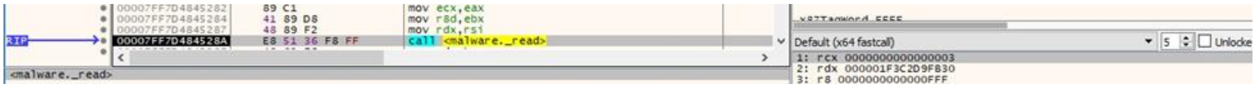
```
Default (x64 fastcall)
1: rcx 0000000000000003
2: rdx 000001F3C2D9FB30
3: r8 0000000000000118
4: r9 0000000000000000
5: [rsp+20] 0000002F871FF6A8
```

```
Address          Hex                                              ASCII
000001F3C2D9FB30 55 8C F5 07 08 50 D1 AE 27 92 6F 30 11 D0 6C 58  U.õ..PÑ®'.o0.Ðl[
000001F3C2D9FB40 4D 48 8A F2 9A CF 48 1E 67 44 85 03 AC 9C C9 10  MH.ò.ÏK.gD..¬.É.
000001F3C2D9FB50 85 C7 A8 51 6E 51 67 EA 37 2F F6 94 11 0D 82 67  .Ç QnQgê7/ö....g
000001F3C2D9FB60 4A 46 DE 82 DA 4D CC 69 99 25 E2 16 9A C0 D0 9A  JFÞ.ÚMÌi.%â..ÀÐ.
000001F3C2D9FB70 3D F5 69 81 C7 73 70 95 8B EB 15 BA 15 2C 54 4D  =õi.Çsp..ë.º.,TM
000001F3C2D9FB80 50 4F 14 3A 35 72 57 56 2A 9E CD E2 3B EC 2F 13  PO.:5rWV*.Íâ;ì/.
000001F3C2D9FB90 81 84 7C A2 59 85 05 8C 78 CA 29 3E 14 DF 49 EF  ..|¢Y...xÊ)>.ßIï
000001F3C2D9FBA0 08 95 3C 34 20 5A 7F 81 69 F3 B7 F7 9B A5 6C 10  ..<4 Z..ió·÷.¥l.
000001F3C2D9FBB0 71 C6 E5 E8 7B 1B B5 01 F1 AC 99 F2 5A ED 56 07  qÆåè{.µ.ñ¬.òZíV.
000001F3C2D9FBC0 4E 9C D2 E0 CB E7 A0 AE E5 6D 00 42 A8 81 0A BA  N.ÒàËç ®åm.B¨..º
000001F3C2D9FBD0 CE EF 41 E4 5D 4A 29 45 FD E6 BF 8C E5 EF F7 B2  ÎïAä]J)Eýæ¿.åï÷²
000001F3C2D9FBE0 65 42 2C 9D A5 90 00 C9 D6 CD 64 68 3B 0C E2 CF  eB,.¥..ÉÖÍdh;.âÏ
000001F3C2D9FBF0 B0 80 A4 6C 2F E9 77 1D FD BD E8 4A 08 A4 A0 2A  °.¤l/éw.ý½èJ.¤ *
000001F3C2D9FC00 55 00 1E 42 E8 29 38 33 54 B4 A2 DA E7 91 88 8C  U..Bè)83T´¢Úç...
000001F3C2D9FC10 A4 8E 24 B2 65 1D 6D 6F 11 B4 7D 0D F4 39 F4 BB  ¤.$²e.mo.´}.ô9ô»
000001F3C2D9FC20 F1 84 A7 DE 5D A1 7E E6 2C 34 74 D8 A2 BA 92 31  ñ.§Þ]¡~æ,4tØ¢º.1
000001F3C2D9FC30 A2 87 61 A6 65 67 5D 24 1C FA B0 EA B4 39 68 68  ¢.a¦eg]$.ú°ê´9hh
000001F3C2D9FC40 64 7E 7E 21 21 7E 7E 21 00 F7 D9 C2 F3 01 00 00  d~~!!~~!.÷ÙÂó...
```

```
0000002F871AF220 000001F3C2D9FB20
0000002F871AF228 00000000000003FF
0000002F871AF230 0000000000000101
0000002F871AF238 0000000000000650
0000002F871AF240 0000002F871FF6A8
0000002F871AF248 0000000000000118
0000002F871AF250 000001F3C2D9FB30
0000002F871AF258 0000002F871AF2D0
0000002F871AF260 0000000000000000
0000002F871AF268 00007FF7D484BE8B return to malw
0000002F871AF270 0000000000008002
0000002F871AF278 0000002F871AF4B8
0000002F871AF280 000000FFFFFFFFB7
0000002F871AF288 00007FF8AE1818A6 return to kern
0000002F871AF290 0000000000000000
0000002F871AF298 000001F3C2EA2360
0000002F871AF2A0 000001F3C2EA0150
0000002F871AF2A8 000000000000014A
0000002F871AF2B0 0000002F871AF2C0
0000002F871AF2B8 0000000000000000
0000002F871AF2C0 0000008000000004
```

Figure 52

It reads the content that will be encrypted using the _read function:

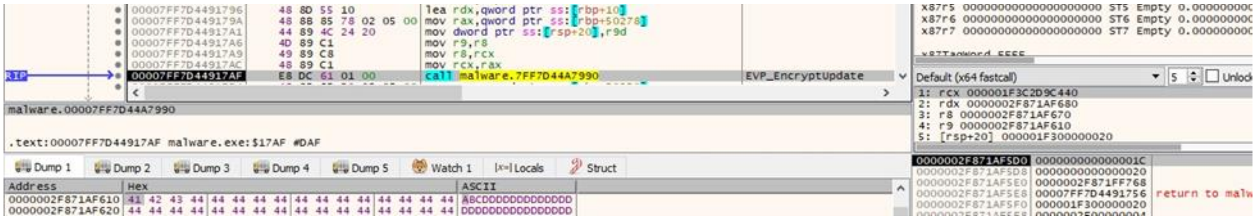


```
00007FF7D4845282   89 C1            mov ecx,eax
00007FF7D4845284   41 89 D8         mov r8d,ebx
00007FF7D4845287   48 89 F2         mov rdx,rsi
00007FF7D484528A   E8 51 36 F8 FF   call <malware._read>
<malware._read>
```

```
Default (x64 fastcall)
1: rcx 0000000000000003
2: rdx 000001F3C2D9FB30
3: r8 0000000000000FFF
```

Figure 53

EVP_EncryptUpdate is used to encrypt data:



```
00007FF7D4491796   48 8D 55 10            lea rdx,qword ptr ss:[rbp+10]
00007FF7D449179A   48 8B 85 78 02 05 00   mov rax,qword ptr ss:[rbp+50278]
00007FF7D44917A1   44 89 4C 24 20         mov dword ptr ss:[rsp+20],r9d
00007FF7D44917A6   4D 89 C1               mov r9,r8
00007FF7D44917A9   49 89 C8               mov r8,rcx
00007FF7D44917AC   48 89 C1               mov rcx,rax
00007FF7D44917AF   E8 DC 61 01 00         call malware.7FF7D44A7990     EVP_EncryptUpdate
malware.00007FF7D44A7990

.text:00007FF7D44917AF malware.exe:$17AF #DAF
```

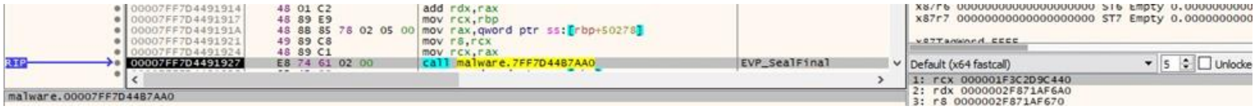
```
x87r5 00000000000000000000 ST5 Empty 0.000000000
x87r6 00000000000000000000 ST6 Empty 0.000000000
x87r7 00000000000000000000 ST7 Empty 0.000000000

Default (x64 fastcall)
1: rcx 000001F3C2D9C440
2: rdx 0000002F871AF680
3: r8 0000002F871AF670
4: r9 0000002F871AF610
5: [rsp+20] 000001F300000020
```

```
Address          Hex                                              ASCII
0000002F871AF610 41 42 43 44 44 44 44 44 44 44 44 44 44 44 44 44  ABCDDDDDDDDDDDDD
0000002F871AF620 44 44 44 44 44 44 44 44 44 44 44 44 44 44 44 44  DDDDDDDDDDDDDDDD
```

```
0000002F871AF5D0 000000000000001C
0000002F871AF5D8 0000000000000020
0000002F871AF5E0 0000002F871FF768
0000002F871AF5E8 00007FF7D4491756 return to malw
0000002F871AF5F0 000001F300000020
```

Figure 54

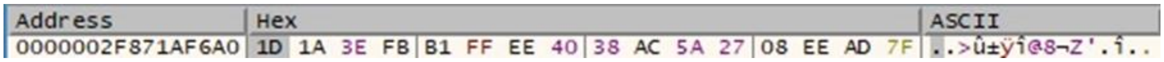
Finally, the malware calls the EVP_SealFinal method:



```
00007FF7D4491914   48 01 C2               add rdx,rax
00007FF7D4491917   48 89 E9               mov rcx,rbp
00007FF7D449191A   48 8B 85 78 02 05 00   mov rax,qword ptr ss:[rbp+50278]
00007FF7D4491921   49 89 C8               mov r8,rcx
00007FF7D4491924   48 89 C1               mov rcx,rax
00007FF7D4491927   E8 74 61 02 00         call malware.7FF7D44B7AA0     EVP_SealFinal
malware.00007FF7D44B7AA0
```

```
x87r7 00000000000000000000 ST7 Empty 0.0000000000

Default (x64 fastcall)
1: rcx 000001F3C2D9C440
2: rdx 0000002F871AF6A0
3: r8 0000002F871AF670
```

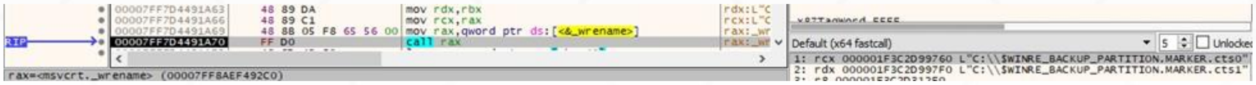
Figure 55



```
Address          Hex                                              ASCII
0000002F871AF6A0 1D 1A 3E FB B1 FF EE 40 38 AC 5A 27 08 EE AD 7F  ..>û±ÿî@8¬Z'.î..
```

Figure 56

The extension is changed again to the other remaining value:

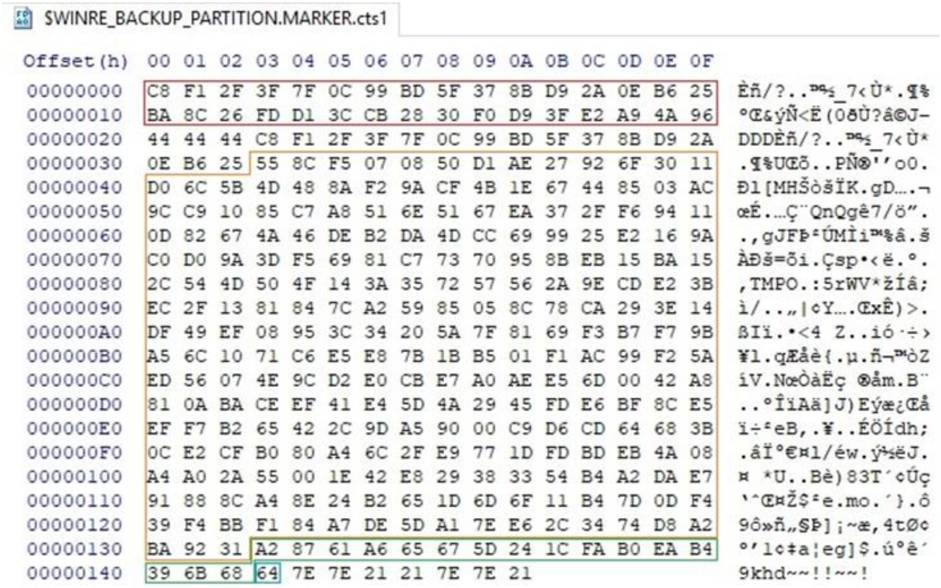


```
        00007FF7D4491A63   48 89 DA              mov rdx,rbx                                  rdx:L"C
        00007FF7D4491A66   48 89 C1              mov rcx,rax                                  rcx:L"C
        00007FF7D4491A69   48 8B 05 F8 65 56 00  mov rax,qword ptr ds:[<&_wrename>]           rax:_wr
RIP --> 00007FF7D4491A70   FF D0                 call rax                                     rax:_wr
rax=<msvcrt._wrename> (00007FF8AEF492C0)
```

```
Default (x64 fastcall)    5    ☐ Unlocked
1: rcx 000001F3C2D99760 L"C:\\$WINRE_BACKUP_PARTITION.MARKER.cts0"
2: rdx 000001F3C2D997F0 L"C:\\$WINRE_BACKUP_PARTITION.MARKER.cts1"
```

Figure 57

The structure of an encrypted file can be seen in Figure 58.



```
$WINRE_BACKUP_PARTITION.MARKER.cts1

Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  C8 F1 2F 3F 7F 0C 99 BD 5F 37 8B D9 2A 0E B6 25  Èñ/?..™½_7‹Ù*.¶%
00000010  BA 8C 26 FD D1 3C CB 28 30 F0 D9 3F E2 A9 4A 96  °Œ&ýÑ<Ë(0ðÙ?â©J–
00000020  44 44 44 C8 F1 2F 3F 7F 0C 99 BD 5F 37 8B D9 2A  DDDÈñ/?..™½_7‹Ù*
00000030  0E B6 25 55 8C F5 07 08 50 D1 AE 27 92 6F 30 11  .¶%UŒõ..PÑ®'’o0.
00000040  D0 6C 5B 4D 48 8A F2 9A CF 4B 1E 67 44 85 03 AC  Ðl[MHŠòšÏK.gD….¬
00000050  9C C9 10 85 C7 A8 51 6E 51 67 EA 37 2F F6 94 11  œÉ.…Ç¨QnQgê7/ö”.
00000060  0D 82 67 4A 46 DE B2 DA 4D CC 69 99 25 E2 16 9A  .‚gJFÞ²ÚMÌi™%â.š
00000070  C0 D0 9A 3D F5 69 81 C7 73 70 95 8B EB 15 BA 15  ÀÐš=õi.Çsp•‹ë.°.
00000080  2C 54 4D 50 4F 14 3A 35 72 57 56 2A 9E CD E2 3B  ,TMPO.:5rWV*žÍâ;
00000090  EC 2F 13 81 84 7C A2 59 85 05 8C 78 CA 29 3E 14  ì/..„|¢Y….ŒxÊ)>.
000000A0  DF 49 EF 08 95 3C 34 20 5A 7F 81 69 F3 B7 F7 9B  ßIï.•<4 Z..ió·÷›
000000B0  A5 6C 10 71 C6 E5 E8 7B 1B B5 01 F1 AC 99 F2 5A  ¥l.qÆåè{.µ.ñ¬™òZ
000000C0  ED 56 07 4E 9C D2 E0 CB E7 A0 AE E5 6D 00 42 A8  íV.NœÒàËç ®åm.B¨
000000D0  81 0A BA CE EF 41 E4 5D 4A 29 45 FD E6 BF 8C E5  ..°ÎïAä]J)Eýæ¿Œå
000000E0  EF F7 B2 65 42 2C 9D A5 90 00 C9 D6 CD 64 68 3B  ï÷²eB,.¥..ÉÖÍdh;
000000F0  0C E2 CF B0 80 A4 6C 2F E9 77 1D FD BD EB 4A 08  .âÏ°€¤l/éw.ý½ëJ.
00000100  A4 A0 2A 55 00 1E 42 E8 29 38 33 54 B4 A2 DA E7  ¤ *U..Bè)83T´¢Úç
00000110  91 88 8C A4 8E 24 B2 65 1D 6D 6F 11 B4 7D 0D F4  ‘ˆŒ¤Ž$²e.mo.´}.ô
00000120  39 F4 BB F1 84 A7 DE 5D A1 7E E6 2C 34 74 D8 A2  9ô»ñ„§Þ]¡~æ,4tØ¢
00000130  BA 92 31 A2 87 61 A6 65 67 5D 24 1C FA B0 EA B4  °’1¢‡a¦eg]$.ú°ê´
00000140  39 6B 68 64 7E 7E 21 21 7E 7E 21                 9khd~~!!~~!
```

Figure 58

Running with the -l parameter

In this case, the ransomware sets the needLogger variable to 1 and creates a log file called “update.log” in the ProgramData directory.

Running with the -e parameter

This is the extra logging feature of the ransomware, which adds even more steps to the same log file.

Running with the -c parameter

This parameter is used to change the encryption percentage (sizeCoverGlobal variable) when partially encrypting the files.

Running with the -t parameter

The number of threads available for encryption can be changed in the maxThreads variable.

Running with the -d parameter

The ransomware only encrypts a specific directory.

Running with the -f parameter

In this case, a single file is encrypted by the malware.

Indicators of Compromise

SHA256

78C16DE9FC07F1D0375A093903F86583A4E32037A7DA8AA2F90ECB15C4862C17

Cactus Ransom Note

CaCtUs.ReAdMe.txt

Mutex

b4kr-xr7h-qcps-omu3cAcTuS

Files created

C:\ProgramData\ntuser.dat

C:\ProgramData\b4kr-xr7h-qcps-omu3.exe

C:\ProgramData\update.log

Scheduled task

Updates Check Task