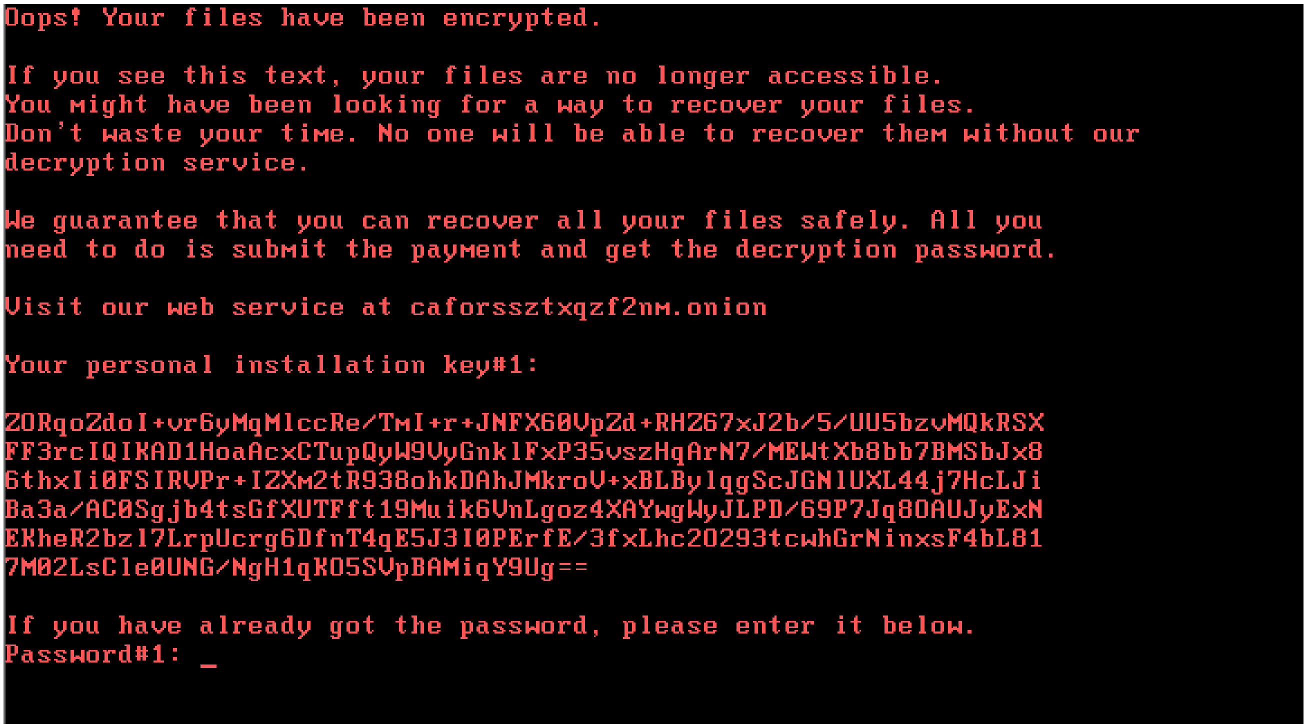
NotPetya Returns as Bad Rabbit

intezer.com/notpetya-returns-bad-rabbit/

October 24, 2017

Written by Jay Rosenberg - 24 October 2017

Large scale cyber attacks seem to be happening once a month these days. Originally discovered by ESET (https://www.welivesecurity.com/2017/10/24/kiev-metro-hit-new-variant-infamous-diskcoder-ransomware/), Ukrainian and Russian organizations have been hit with the latest ransomware attack named Bad Rabbit. At the time of writing this post, the ransomware has believed to have originated from compromised webpages with a fake popup for updating Adobe Flash Player. It has been reported that much of the behavior of Bad Rabbit has been similar to a previous ransomware known as NotPetya.

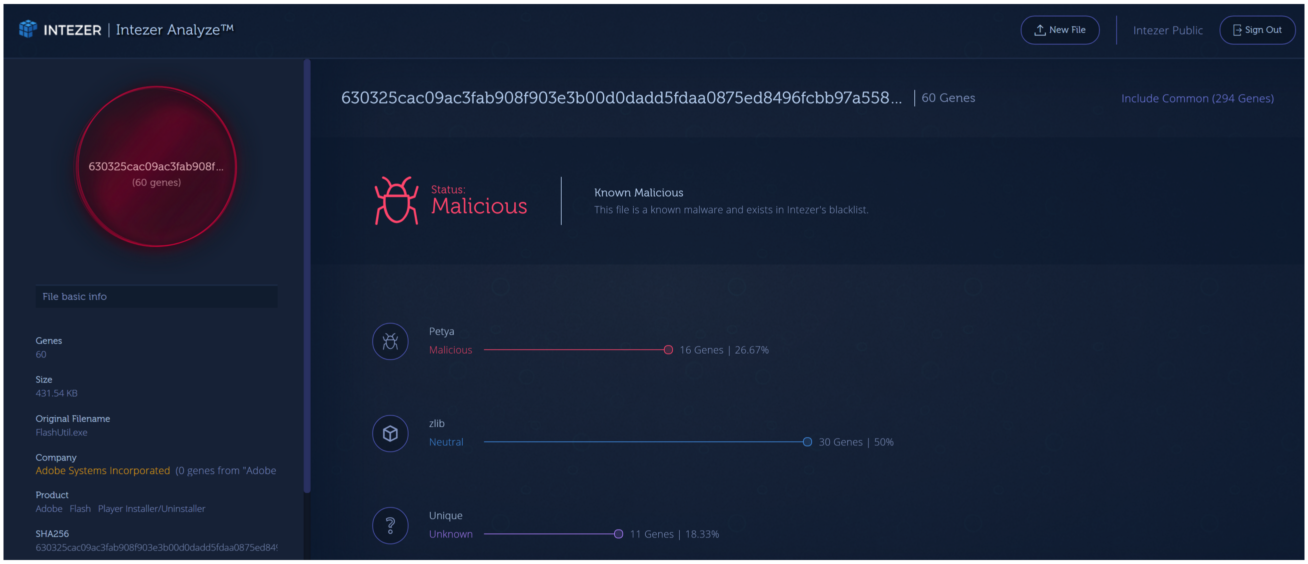


```
Oops! Your files have been encrypted.

If you see this text, your files are no longer accessible.
You might have been looking for a way to recover your files.
Don't waste your time. No one will be able to recover them without our
decryption service.

We guarantee that you can recover all your files safely. All you
need to do is submit the payment and get the decryption password.

Visit our web service at caforssztxqzf2nm.onion

Your personal installation key#1:

ZORqoZdoI+vr6yMqMlccRe/TmI+r+JNFX60VpZd+RHZ67xJ2b/5/UU5bzvMQkRSX
FF3rcIQIKAD1HoaAcxCTupQyW9VyGnklFxP35vszHqArN7/MEWtXb8bb7BMSbJx8
6thxIi0FSIRVPr+IZXm2tR938ohkDAhJMkroV+xBLBylqgScJGNlUXL44j7HcLJi
Ba3a/AC0Sgjb4tsGfXUTFft19Muik6VnLgoz4XAYwgWyJLPD/69P7Jq8OAUJyExN
EKheR2bzl7LrpUcrg6DfnT4qE5J3I0PErfE/3fxLhc2O293tcwhGrNinxsF4bL81
7M02LsCle0UNG/NgH1qKO5SVpBAMiqY9Ug==

If you have already got the password, please enter it below.
Password#1: _
```

(screenshot from ESET report, after ransomware has infected a computer)

Using Intezer Analyze™, we have found code reuse from NotPetya throughout different binaries of Bad Rabbit.

The Bad Rabbit loader, with the original name (install_flash_player.exe) and metadata (Adobe Systems Incorporated as the company and Adobe Flash Player Installer/Uninstaller), was made to look like the Adobe Flash Player installer. You can see in the screenshot below that according to our analysis, the binary did not contain any code from any Adobe product but does contain code from NotPetya. In fact, we find that 27% of the code in the loader has been seen in only NotPetya samples. Find the public report here (https://analyze.intezer.com/#/analyses/6ba279af-8ce2-46c6-8b86-5fa65a5ed42a)



630325cac09ac3fab908f903e3b00d0dadd5fdaa0875ed8496fcbb97a558... | 60 Genes

Include Common (294 Genes)

630325cac09ac3fab908f...
(60 genes)

Status:
Malicious

Known Malicious
This file is a known malware and exists in Intezer's blacklist.

File basic info

Genes
60

Size
431.54 KB

Original Filename
FlashUtil.exe

Company
Adobe Systems Incorporated (0 genes from "Adobe

Product
Adobe Flash Player Installer/Uninstaller

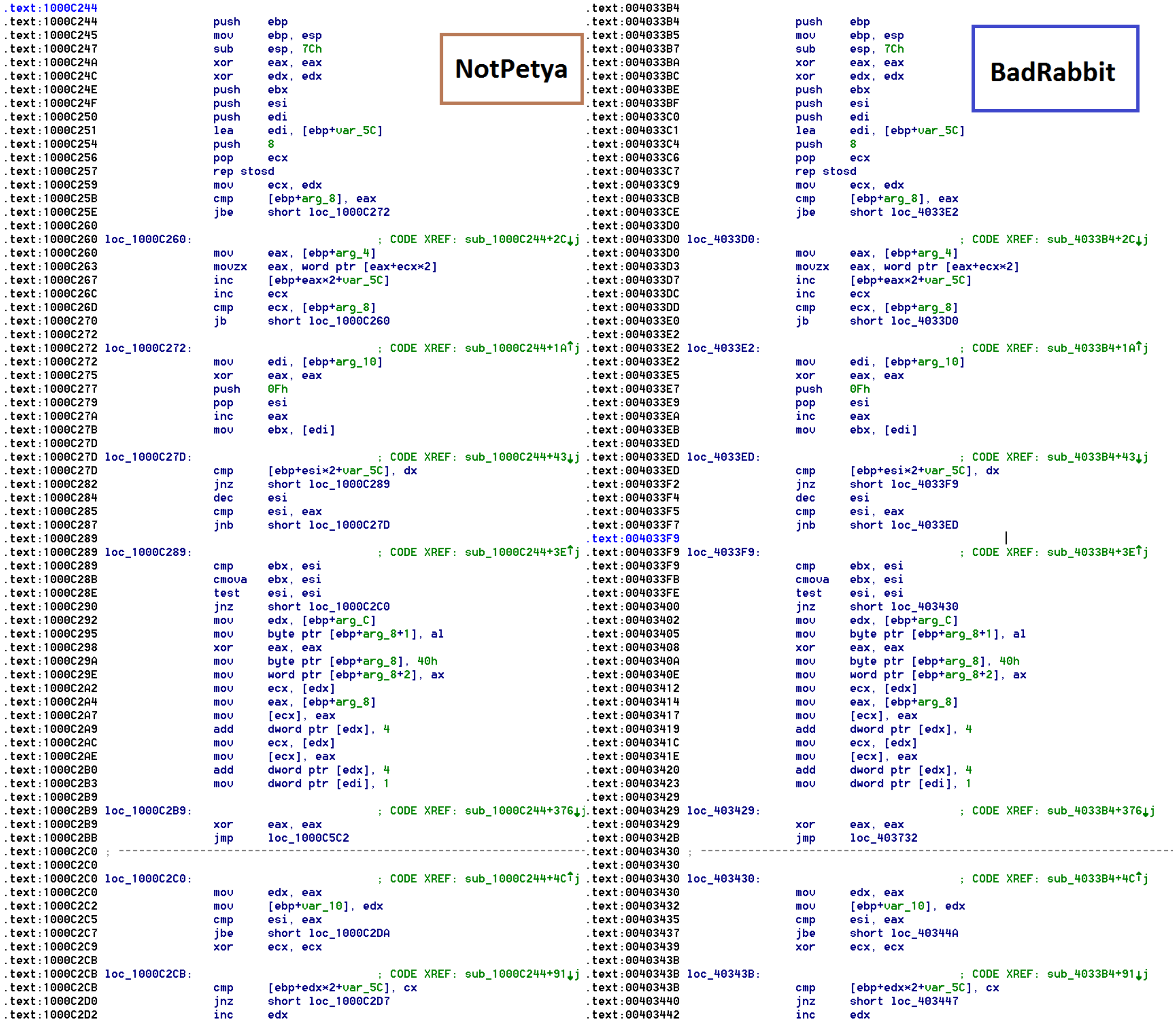
SHA256
630325cac09ac3fab908f903e3b00d0dadd5fdaa0875ed84

Petya
Malicious
16 Genes | 26.67%

zlib
Neutral
30 Genes | 50%

Unique
Unknown
11 Genes | 18.33%

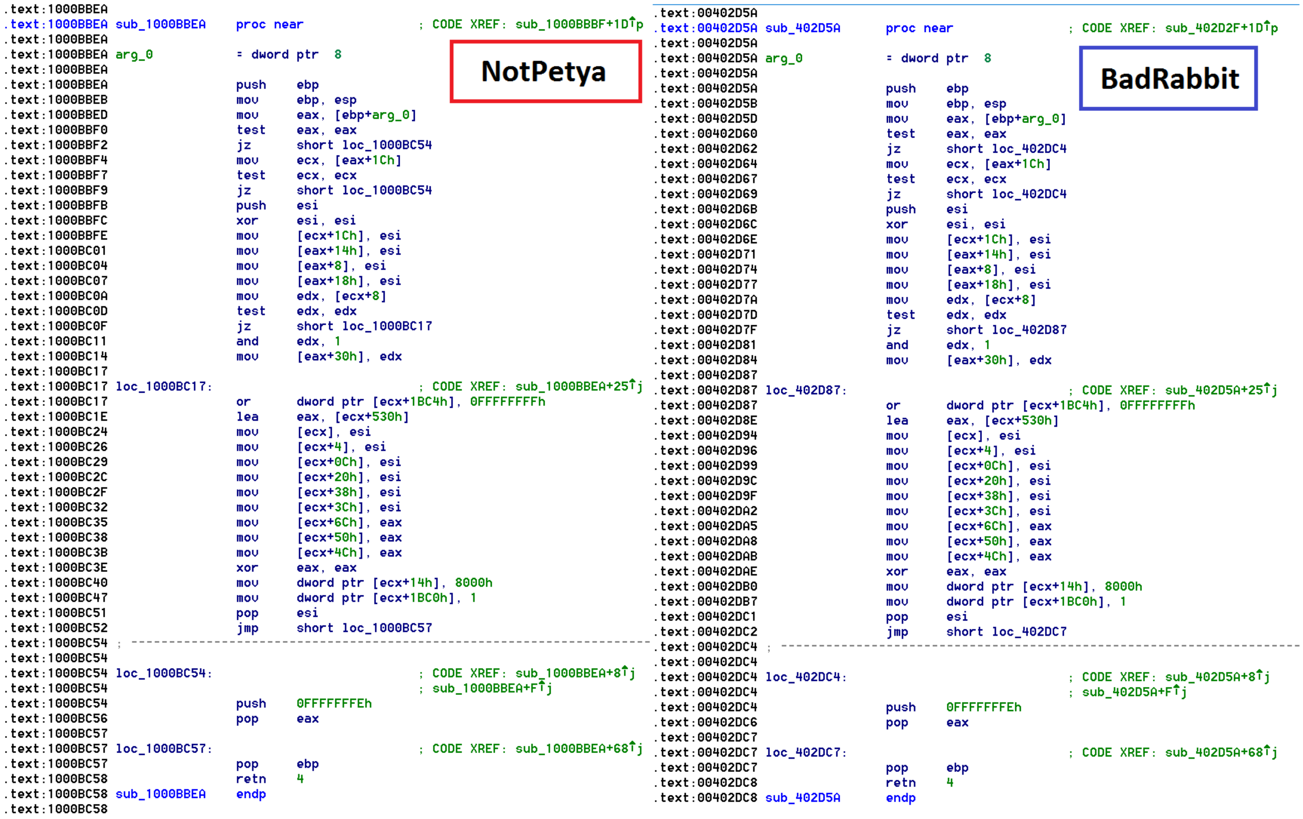
Below is a direct comparison of function (0x1000C244) of NotPetya (027cc450ef5f8c5f653329641ec1fed91f694e0d229928963b30f6b0d7d3a745) and function (0x4033B4) of the Bad Rabbit loader (630325cac09ac3fab908f903e3b00d0dadd5fdaa0875ed8496fcbb97a558d0da).



```
.text:1000C244
.text:1000C244                 push    ebp
.text:1000C245                 mov     ebp, esp
.text:1000C247                 sub     esp, 7Ch
.text:1000C24A                 xor     eax, eax
.text:1000C24C                 xor     edx, edx
.text:1000C24E                 push    ebx
.text:1000C24F                 push    esi
.text:1000C250                 push    edi
.text:1000C251                 lea     edi, [ebp+var_5C]
.text:1000C254                 push    8
.text:1000C256                 pop     ecx
.text:1000C257                 rep stosd
.text:1000C259                 mov     ecx, edx
.text:1000C25B                 cmp     [ebp+arg_8], eax
.text:1000C25E                 jbe     short loc_1000C272
.text:1000C260
.text:1000C260 loc_1000C260:                           ; CODE XREF: sub_1000C244+2C↓j
.text:1000C260                 mov     eax, [ebp+arg_4]
.text:1000C263                 movzx   eax, word ptr [eax+ecx*2]
.text:1000C267                 inc     [ebp+eax*2+var_5C]
.text:1000C26C                 inc     ecx
.text:1000C26D                 cmp     ecx, [ebp+arg_8]
.text:1000C270                 jb      short loc_1000C260
.text:1000C272
.text:1000C272 loc_1000C272:                           ; CODE XREF: sub_1000C244+1A↑j
.text:1000C272                 mov     edi, [ebp+arg_10]
.text:1000C275                 xor     eax, eax
.text:1000C277                 push    0Fh
.text:1000C279                 pop     esi
.text:1000C27A                 inc     eax
.text:1000C27B                 mov     ebx, [edi]
.text:1000C27D
.text:1000C27D loc_1000C27D:                           ; CODE XREF: sub_1000C244+43↓j
.text:1000C27D                 cmp     [ebp+esi*2+var_5C], dx
.text:1000C282                 jnz     short loc_1000C289
.text:1000C284                 dec     esi
.text:1000C285                 cmp     esi, eax
.text:1000C287                 jnb     short loc_1000C27D
.text:1000C289
.text:1000C289 loc_1000C289:                           ; CODE XREF: sub_1000C244+3E↑j
.text:1000C289                 cmp     ebx, esi
.text:1000C28B                 cmova   ebx, esi
.text:1000C28E                 test    esi, esi
.text:1000C290                 jnz     short loc_1000C2C0
.text:1000C292                 mov     edx, [ebp+arg_C]
.text:1000C295                 mov     byte ptr [ebp+arg_8+1], al
.text:1000C298                 xor     eax, eax
.text:1000C29A                 mov     byte ptr [ebp+arg_8], 40h
.text:1000C29E                 mov     word ptr [ebp+arg_8+2], ax
.text:1000C2A2                 mov     ecx, [edx]
.text:1000C2A4                 mov     eax, [ebp+arg_8]
.text:1000C2A7                 mov     [ecx], eax
.text:1000C2A9                 add     dword ptr [edx], 4
.text:1000C2AC                 mov     ecx, [edx]
.text:1000C2AE                 mov     [ecx], eax
.text:1000C2B0                 add     dword ptr [edx], 4
.text:1000C2B3                 mov     dword ptr [edi], 1
.text:1000C2B9
.text:1000C2B9 loc_1000C2B9:                           ; CODE XREF: sub_1000C244+376↓j
.text:1000C2B9                 xor     eax, eax
.text:1000C2BB                 jmp     loc_1000C5C2
.text:1000C2C0 ; ---------------------------------------------------------------------------
.text:1000C2C0
.text:1000C2C0 loc_1000C2C0:                           ; CODE XREF: sub_1000C244+4C↑j
.text:1000C2C0                 mov     edx, eax
.text:1000C2C2                 mov     [ebp+var_10], edx
.text:1000C2C5                 cmp     esi, eax
.text:1000C2C7                 jbe     short loc_1000C2DA
.text:1000C2C9                 xor     ecx, ecx
.text:1000C2CB
.text:1000C2CB loc_1000C2CB:                           ; CODE XREF: sub_1000C244+91↓j
.text:1000C2CB                 cmp     [ebp+edx*2+var_5C], cx
.text:1000C2D0                 jnz     short loc_1000C2D7
.text:1000C2D2                 inc     edx
```

```
.text:004033B4
.text:004033B4                 push    ebp
.text:004033B5                 mov     ebp, esp
.text:004033B7                 sub     esp, 7Ch
.text:004033BA                 xor     eax, eax
.text:004033BC                 xor     edx, edx
.text:004033BE                 push    ebx
.text:004033BF                 push    esi
.text:004033C0                 push    edi
.text:004033C1                 lea     edi, [ebp+var_5C]
.text:004033C4                 push    8
.text:004033C6                 pop     ecx
.text:004033C7                 rep stosd
.text:004033C9                 mov     ecx, edx
.text:004033CB                 cmp     [ebp+arg_8], eax
.text:004033CE                 jbe     short loc_4033E2
.text:004033D0
.text:004033D0 loc_4033D0:                             ; CODE XREF: sub_4033B4+2C↓j
.text:004033D0                 mov     eax, [ebp+arg_4]
.text:004033D3                 movzx   eax, word ptr [eax+ecx*2]
.text:004033D7                 inc     [ebp+eax*2+var_5C]
.text:004033DC                 inc     ecx
.text:004033DD                 cmp     ecx, [ebp+arg_8]
.text:004033E0                 jb      short loc_4033D0
.text:004033E2
.text:004033E2 loc_4033E2:                             ; CODE XREF: sub_4033B4+1A↑j
.text:004033E2                 mov     edi, [ebp+arg_10]
.text:004033E5                 xor     eax, eax
.text:004033E7                 push    0Fh
.text:004033E9                 pop     esi
.text:004033EA                 inc     eax
.text:004033EB                 mov     ebx, [edi]
.text:004033ED
.text:004033ED loc_4033ED:                             ; CODE XREF: sub_4033B4+43↓j
.text:004033ED                 cmp     [ebp+esi*2+var_5C], dx
.text:004033F2                 jnz     short loc_4033F9
.text:004033F4                 dec     esi
.text:004033F5                 cmp     esi, eax
.text:004033F7                 jnb     short loc_4033ED
.text:004033F9
.text:004033F9 loc_4033F9:                             ; CODE XREF: sub_4033B4+3E↑j
.text:004033F9                 cmp     ebx, esi
.text:004033FB                 cmova   ebx, esi
.text:004033FE                 test    esi, esi
.text:00403400                 jnz     short loc_403430
.text:00403402                 mov     edx, [ebp+arg_C]
.text:00403405                 mov     byte ptr [ebp+arg_8+1], al
.text:00403408                 xor     eax, eax
.text:0040340A                 mov     byte ptr [ebp+arg_8], 40h
.text:0040340E                 mov     word ptr [ebp+arg_8+2], ax
.text:00403412                 mov     ecx, [edx]
.text:00403414                 mov     eax, [ebp+arg_8]
.text:00403417                 mov     [ecx], eax
.text:00403419                 add     dword ptr [edx], 4
.text:0040341C                 mov     ecx, [edx]
.text:0040341E                 mov     [ecx], eax
.text:00403420                 add     dword ptr [edx], 4
.text:00403423                 mov     dword ptr [edi], 1
.text:00403429
.text:00403429 loc_403429:                             ; CODE XREF: sub_4033B4+376↓j
.text:00403429                 xor     eax, eax
.text:0040342B                 jmp     loc_403732
.text:00403430 ; ---------------------------------------------------------------------------
.text:00403430
.text:00403430 loc_403430:                             ; CODE XREF: sub_4033B4+4C↑j
.text:00403430                 mov     edx, eax
.text:00403432                 mov     [ebp+var_10], edx
.text:00403435                 cmp     esi, eax
.text:00403437                 jbe     short loc_40344A
.text:00403439                 xor     ecx, ecx
.text:0040343B
.text:0040343B loc_40343B:                             ; CODE XREF: sub_4033B4+91↓j
.text:0040343B                 cmp     [ebp+edx*2+var_5C], cx
.text:00403440                 jnz     short loc_403447
.text:00403442                 inc     edx
```

Another example of code reuse in the loader from a function that seems to initialize some type of struct.



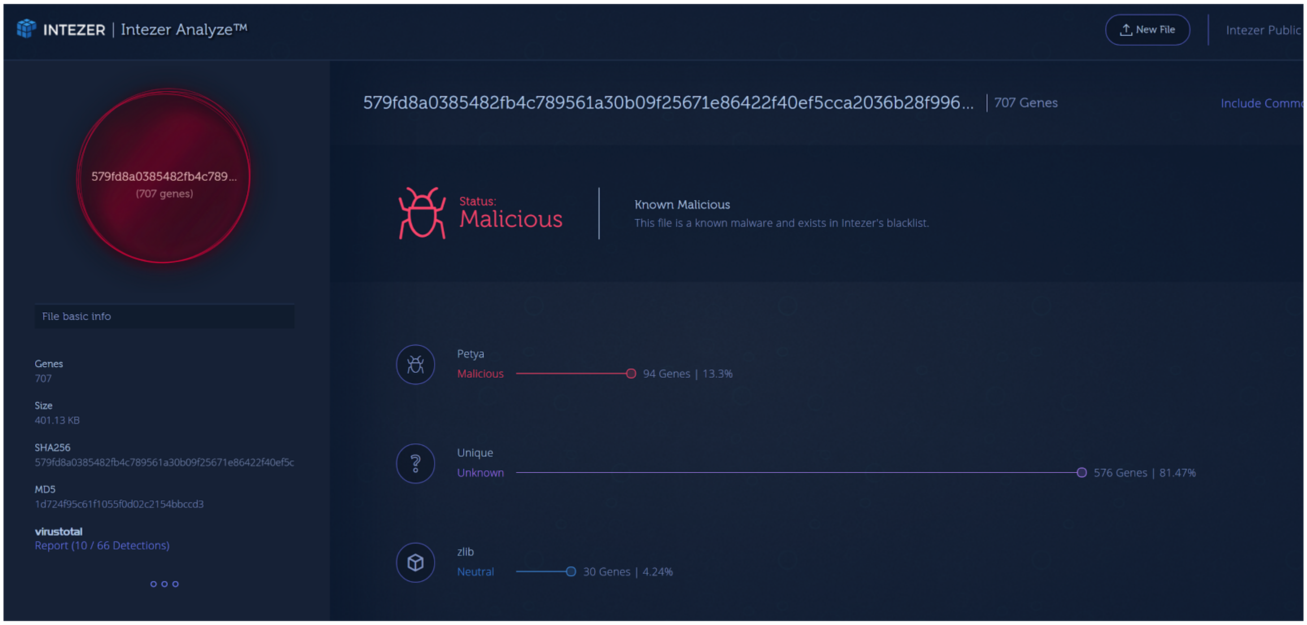
```
.text:1000BBEA
.text:1000BBEA sub_1000BBEA    proc near               ; CODE XREF: sub_1000BBBF+1D↑p
.text:1000BBEA
.text:1000BBEA arg_0           = dword ptr  8
.text:1000BBEA
.text:1000BBEA                 push    ebp
.text:1000BBEB                 mov     ebp, esp
.text:1000BBED                 mov     eax, [ebp+arg_0]
.text:1000BBF0                 test    eax, eax
.text:1000BBF2                 jz      short loc_1000BC54
.text:1000BBF4                 mov     ecx, [eax+1Ch]
.text:1000BBF7                 test    ecx, ecx
.text:1000BBF9                 jz      short loc_1000BC54
.text:1000BBFB                 push    esi
.text:1000BBFC                 xor     esi, esi
.text:1000BBFE                 mov     [ecx+1Ch], esi
.text:1000BC01                 mov     [eax+14h], esi
.text:1000BC04                 mov     [eax+8], esi
.text:1000BC07                 mov     [eax+18h], esi
.text:1000BC0A                 mov     edx, [ecx+8]
.text:1000BC0D                 test    edx, edx
.text:1000BC0F                 jz      short loc_1000BC17
.text:1000BC11                 and     edx, 1
.text:1000BC14                 mov     [eax+30h], edx
.text:1000BC17
.text:1000BC17 loc_1000BC17:                           ; CODE XREF: sub_1000BBEA+25↑j
.text:1000BC17                 or      dword ptr [ecx+1BC4h], 0FFFFFFFFh
.text:1000BC1E                 lea     eax, [ecx+530h]
.text:1000BC24                 mov     [ecx], esi
.text:1000BC26                 mov     [ecx+4], esi
.text:1000BC29                 mov     [ecx+0Ch], esi
.text:1000BC2C                 mov     [ecx+20h], esi
.text:1000BC2F                 mov     [ecx+38h], esi
.text:1000BC32                 mov     [ecx+3Ch], esi
.text:1000BC35                 mov     [ecx+6Ch], eax
.text:1000BC38                 mov     [ecx+50h], eax
.text:1000BC3B                 mov     [ecx+4Ch], eax
.text:1000BC3E                 xor     eax, eax
.text:1000BC40                 mov     dword ptr [ecx+14h], 8000h
.text:1000BC47                 mov     dword ptr [ecx+1BC0h], 1
.text:1000BC51                 pop     esi
.text:1000BC52                 jmp     short loc_1000BC57
.text:1000BC54 ; ---------------------------------------------------------------------------
.text:1000BC54
.text:1000BC54 loc_1000BC54:                           ; CODE XREF: sub_1000BBEA+8↑j
.text:1000BC54                                         ; sub_1000BBEA+F↑j
.text:1000BC54                 push    0FFFFFFFEh
.text:1000BC56                 pop     eax
.text:1000BC57
.text:1000BC57 loc_1000BC57:                           ; CODE XREF: sub_1000BBEA+68↑j
.text:1000BC57                 pop     ebp
.text:1000BC58                 retn    4
.text:1000BC58 sub_1000BBEA    endp
.text:1000BC58
```

```
.text:00402D5A
.text:00402D5A sub_402D5A      proc near               ; CODE XREF: sub_402D2F+1D↑p
.text:00402D5A
.text:00402D5A arg_0           = dword ptr  8
.text:00402D5A
.text:00402D5A                 push    ebp
.text:00402D5B                 mov     ebp, esp
.text:00402D5D                 mov     eax, [ebp+arg_0]
.text:00402D60                 test    eax, eax
.text:00402D62                 jz      short loc_402DC4
.text:00402D64                 mov     ecx, [eax+1Ch]
.text:00402D67                 test    ecx, ecx
.text:00402D69                 jz      short loc_402DC4
.text:00402D6B                 push    esi
.text:00402D6C                 xor     esi, esi
.text:00402D6E                 mov     [ecx+1Ch], esi
.text:00402D71                 mov     [eax+14h], esi
.text:00402D74                 mov     [eax+8], esi
.text:00402D77                 mov     [eax+18h], esi
.text:00402D7A                 mov     edx, [ecx+8]
.text:00402D7D                 test    edx, edx
.text:00402D7F                 jz      short loc_402D87
.text:00402D81                 and     edx, 1
.text:00402D84                 mov     [eax+30h], edx
.text:00402D87
.text:00402D87 loc_402D87:                             ; CODE XREF: sub_402D5A+25↑j
.text:00402D87                 or      dword ptr [ecx+1BC4h], 0FFFFFFFFh
.text:00402D8E                 lea     eax, [ecx+530h]
.text:00402D94                 mov     [ecx], esi
.text:00402D96                 mov     [ecx+4], esi
.text:00402D99                 mov     [ecx+0Ch], esi
.text:00402D9C                 mov     [ecx+20h], esi
.text:00402D9F                 mov     [ecx+38h], esi
.text:00402DA2                 mov     [ecx+3Ch], esi
.text:00402DA5                 mov     [ecx+6Ch], eax
.text:00402DA8                 mov     [ecx+50h], eax
.text:00402DAB                 mov     [ecx+4Ch], eax
.text:00402DAE                 xor     eax, eax
.text:00402DB0                 mov     dword ptr [ecx+14h], 8000h
.text:00402DB7                 mov     dword ptr [ecx+1BC0h], 1
.text:00402DC1                 pop     esi
.text:00402DC2                 jmp     short loc_402DC7
.text:00402DC4 ; ---------------------------------------------------------------------------
.text:00402DC4
.text:00402DC4 loc_402DC4:                             ; CODE XREF: sub_402D5A+8↑j
.text:00402DC4                                         ; sub_402D5A+F↑j
.text:00402DC4                 push    0FFFFFFFEh
.text:00402DC6                 pop     eax
.text:00402DC7
.text:00402DC7 loc_402DC7:                             ; CODE XREF: sub_402D5A+68↑j
.text:00402DC7                 pop     ebp
.text:00402DC8                 retn    4
.text:00402DC8 sub_402D5A      endp
```

> #BadRabbit (#NotPetya v2) unpacked DLL: infpub.dat : https://t.co/Ey5Yffsn74
>
> — hasherezade (@hasherezade) October 24, 2017

The final module that gets loaded and is responsible for encrypting the files on disk (579fd8a0385482fb4c789561a30b09f25671e86422f40ef5cca2036b28f99648) also has a code connection with NotPetya samples. According to our technology, we can see that 13% of the code has been reused. You can find the public report here. (https://analyze.intezer.com/#/analyses/d41e8a98-a106-4b4f-9b7c-fd9e2c80ca7d)

> #badrabbit found to have 13% code reuse of #notpetya #petya
> here's a public report with the unpacked sample: https://t.co/NOIul4yLVT
>
> — Jay Rosenberg (@jaytezer) October 24, 2017



579fd8a0385482fb4c789561a30b09f25671e86422f40ef5cca2036b28f996... | 707 Genes

Status:
Malicious

Known Malicious
This file is a known malware and exists in Intezer's blacklist.

File basic info

Genes
707

Size
401.13 KB

SHA256
579fd8a0385482fb4c789561a30b09f25671e86422f40ef5c

MD5
1d724f95c61f1055f0d02c2154bbccd3

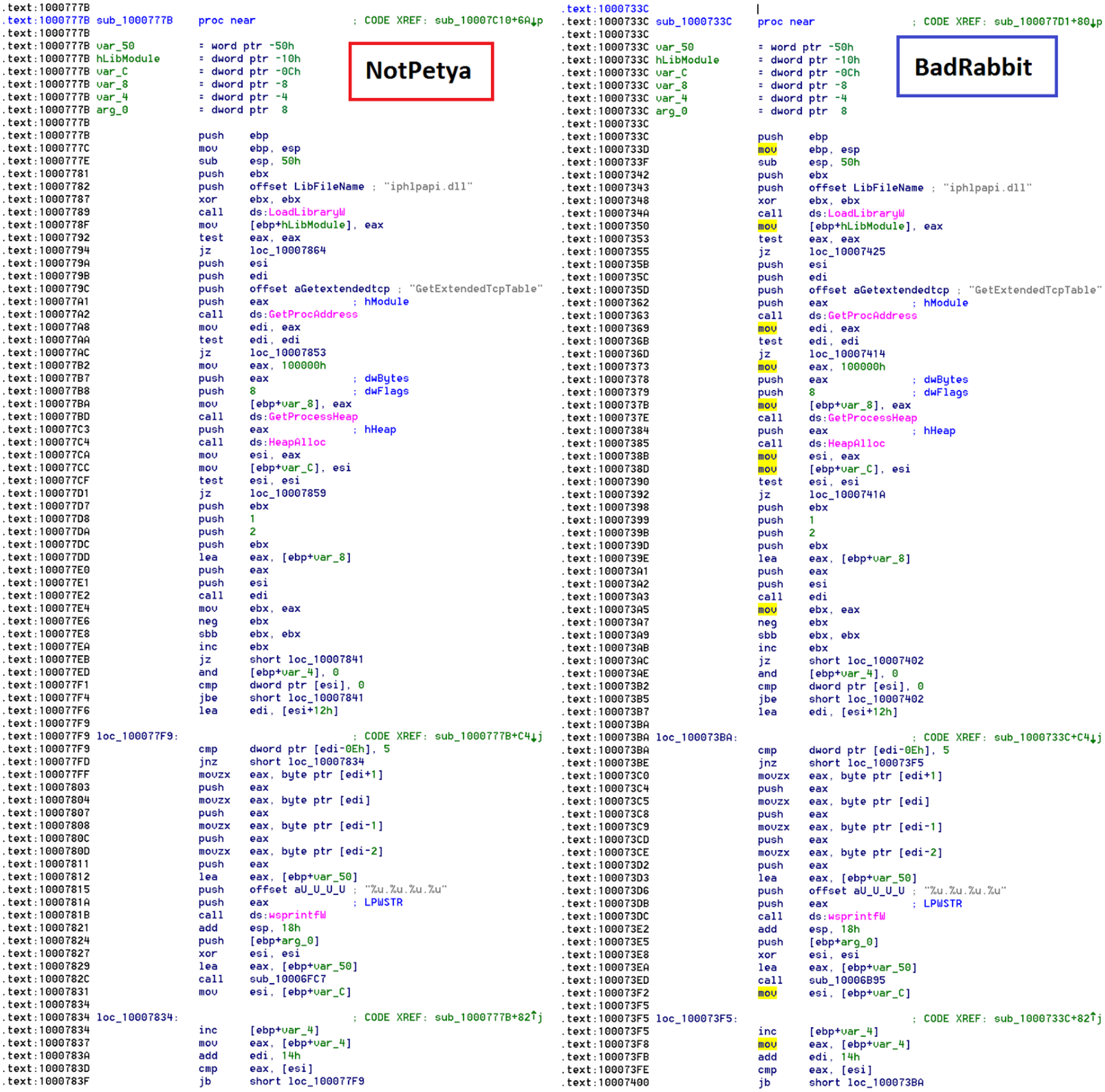
virustotal
Report (10 / 66 Detections)

Petya
Malicious 94 Genes | 13.3%

Unique
Unknown 576 Genes | 81.47%

zlib
Neutral 30 Genes | 4.24%

Below is a screenshot comparing a function (0x1000777B) of NotPetya (027cc450ef5f8c5f653329641ec1fed91f694e0d229928963b30f6b0d7d3a745) and a function (0x1000733C) of the encryptor module of Bad Rabbit (579fd8a0385482fb4c789561a30b09f25671e86422f40ef5cca2036b28f99648).



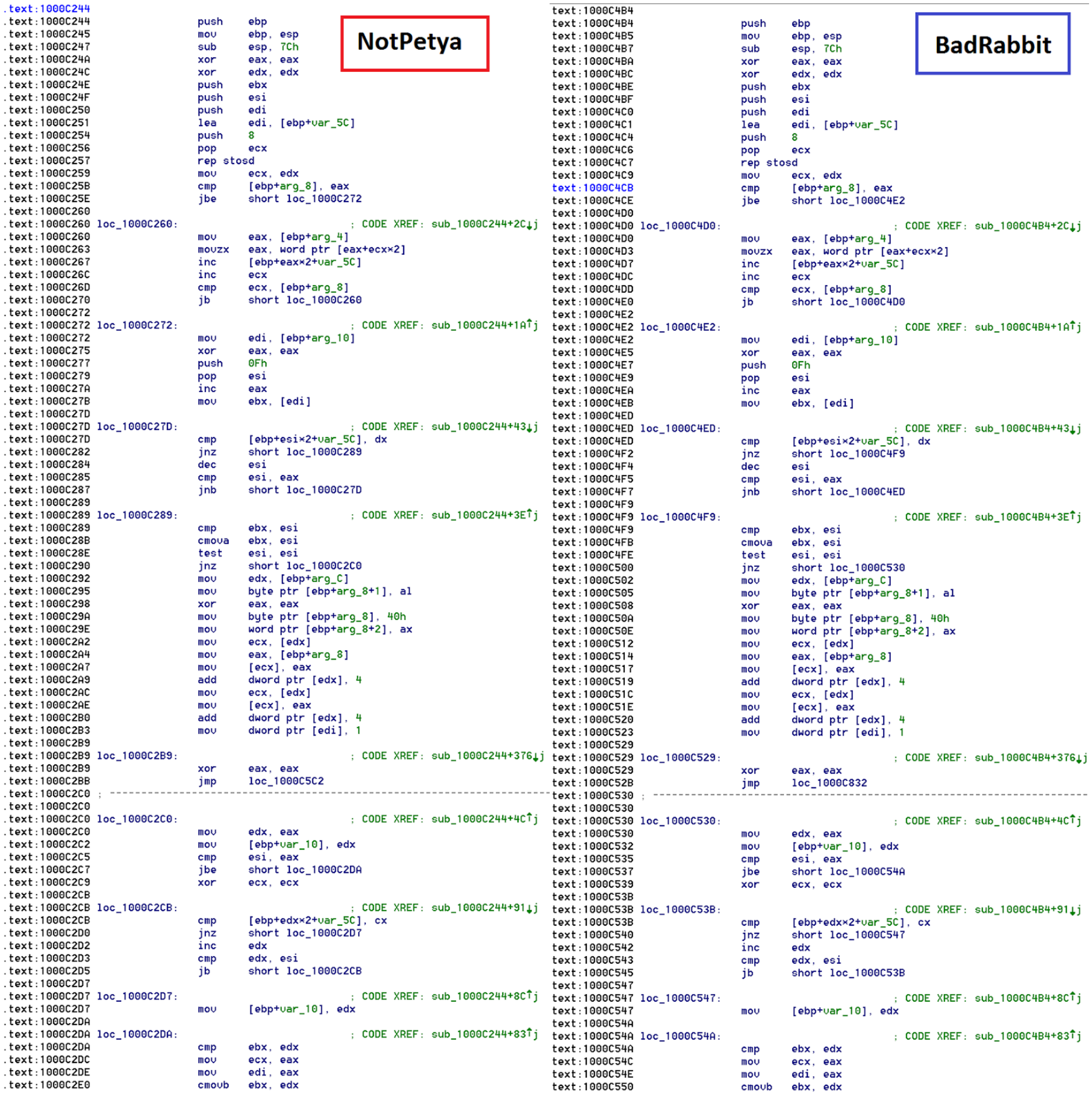
NotPetya

```
.text:1000777B
.text:1000777B sub_1000777B    proc near               ; CODE XREF: sub_10007C10+6A↓p
.text:1000777B
.text:1000777B var_50          = word ptr -50h
.text:1000777B hLibModule      = dword ptr -10h
.text:1000777B var_C           = dword ptr -0Ch
.text:1000777B var_8           = dword ptr -8
.text:1000777B var_4           = dword ptr -4
.text:1000777B arg_0           = dword ptr  8
.text:1000777B
.text:1000777B                 push    ebp
.text:1000777C                 mov     ebp, esp
.text:1000777E                 sub     esp, 50h
.text:10007781                 push    ebx
.text:10007782                 push    offset LibFileName ; "iphlpapi.dll"
.text:10007787                 xor     ebx, ebx
.text:10007789                 call    ds:LoadLibraryW
.text:1000778F                 mov     [ebp+hLibModule], eax
.text:10007792                 test    eax, eax
.text:10007794                 jz      loc_10007864
.text:1000779A                 push    esi
.text:1000779B                 push    edi
.text:1000779C                 push    offset aGetextendedtcp ; "GetExtendedTcpTable"
.text:100077A1                 push    eax             ; hModule
.text:100077A2                 call    ds:GetProcAddress
.text:100077A8                 mov     edi, eax
.text:100077AA                 test    edi, edi
.text:100077AC                 jz      loc_10007853
.text:100077B2                 mov     eax, 100000h
.text:100077B7                 push    eax             ; dwBytes
.text:100077B8                 push    8               ; dwFlags
.text:100077BA                 mov     [ebp+var_8], eax
.text:100077BD                 call    ds:GetProcessHeap
.text:100077C3                 push    eax             ; hHeap
.text:100077C4                 call    ds:HeapAlloc
.text:100077CA                 mov     esi, eax
.text:100077CC                 mov     [ebp+var_C], esi
.text:100077CF                 test    esi, esi
.text:100077D1                 jz      loc_10007859
.text:100077D7                 push    ebx
.text:100077D8                 push    1
.text:100077DA                 push    2
.text:100077DC                 push    ebx
.text:100077DD                 lea     eax, [ebp+var_8]
.text:100077E0                 push    eax
.text:100077E1                 push    esi
.text:100077E2                 call    edi
.text:100077E4                 mov     ebx, eax
.text:100077E6                 neg     ebx
.text:100077E8                 sbb     ebx, ebx
.text:100077EA                 inc     ebx
.text:100077EB                 jz      short loc_10007841
.text:100077ED                 and     [ebp+var_4], 0
.text:100077F1                 cmp     dword ptr [esi], 0
.text:100077F4                 jbe     short loc_10007841
.text:100077F6                 lea     edi, [esi+12h]
.text:100077F9
.text:100077F9 loc_100077F9:                           ; CODE XREF: sub_1000777B+C4↓j
.text:100077F9                 cmp     dword ptr [edi-0Eh], 5
.text:100077FD                 jnz     short loc_10007834
.text:100077FF                 movzx   eax, byte ptr [edi+1]
.text:10007803                 push    eax
.text:10007804                 movzx   eax, byte ptr [edi]
.text:10007807                 push    eax
.text:10007808                 movzx   eax, byte ptr [edi-1]
.text:1000780C                 push    eax
.text:1000780D                 movzx   eax, byte ptr [edi-2]
.text:10007811                 push    eax
.text:10007812                 lea     eax, [ebp+var_50]
.text:10007815                 push    offset aU_U_U_U ; "%u.%u.%u.%u"
.text:1000781A                 push    eax             ; LPWSTR
.text:1000781B                 call    ds:wsprintfW
.text:10007821                 add     esp, 18h
.text:10007824                 push    [ebp+arg_0]
.text:10007827                 xor     esi, esi
.text:10007829                 lea     eax, [ebp+var_50]
.text:1000782C                 call    sub_10006FC7
.text:10007831                 mov     esi, [ebp+var_C]
.text:10007834
.text:10007834 loc_10007834:                           ; CODE XREF: sub_1000777B+82↑j
.text:10007834                 inc     [ebp+var_4]
.text:10007837                 mov     eax, [ebp+var_4]
.text:1000783A                 add     edi, 14h
.text:1000783D                 cmp     eax, [esi]
.text:1000783F                 jb      short loc_100077F9
```

BadRabbit

```
.text:1000733C                 |
.text:1000733C sub_1000733C    proc near               ; CODE XREF: sub_100077D1+80↓p
.text:1000733C
.text:1000733C var_50          = word ptr -50h
.text:1000733C hLibModule      = dword ptr -10h
.text:1000733C var_C           = dword ptr -0Ch
.text:1000733C var_8           = dword ptr -8
.text:1000733C var_4           = dword ptr -4
.text:1000733C arg_0           = dword ptr  8
.text:1000733C
.text:1000733C                 push    ebp
.text:1000733D                 mov     ebp, esp
.text:1000733F                 sub     esp, 50h
.text:10007342                 push    ebx
.text:10007343                 push    offset LibFileName ; "iphlpapi.dll"
.text:10007348                 xor     ebx, ebx
.text:1000734A                 call    ds:LoadLibraryW
.text:10007350                 mov     [ebp+hLibModule], eax
.text:10007353                 test    eax, eax
.text:10007355                 jz      loc_10007425
.text:1000735B                 push    esi
.text:1000735C                 push    edi
.text:1000735D                 push    offset aGetextendedtcp ; "GetExtendedTcpTable"
.text:10007362                 push    eax             ; hModule
.text:10007363                 call    ds:GetProcAddress
.text:10007369                 mov     edi, eax
.text:1000736B                 test    edi, edi
.text:1000736D                 jz      loc_10007414
.text:10007373                 mov     eax, 100000h
.text:10007378                 push    eax             ; dwBytes
.text:10007379                 push    8               ; dwFlags
.text:1000737B                 mov     [ebp+var_8], eax
.text:1000737E                 call    ds:GetProcessHeap
.text:10007384                 push    eax             ; hHeap
.text:10007385                 call    ds:HeapAlloc
.text:1000738B                 mov     esi, eax
.text:1000738D                 mov     [ebp+var_C], esi
.text:10007390                 test    esi, esi
.text:10007392                 jz      loc_1000741A
.text:10007398                 push    ebx
.text:10007399                 push    1
.text:1000739B                 push    2
.text:1000739D                 push    ebx
.text:1000739E                 lea     eax, [ebp+var_8]
.text:100073A1                 push    eax
.text:100073A2                 push    esi
.text:100073A3                 call    edi
.text:100073A5                 mov     ebx, eax
.text:100073A7                 neg     ebx
.text:100073A9                 sbb     ebx, ebx
.text:100073AB                 inc     ebx
.text:100073AC                 jz      short loc_10007402
.text:100073AE                 and     [ebp+var_4], 0
.text:100073B2                 cmp     dword ptr [esi], 0
.text:100073B5                 jbe     short loc_10007402
.text:100073B7                 lea     edi, [esi+12h]
.text:100073BA
.text:100073BA loc_100073BA:                           ; CODE XREF: sub_1000733C+C4↓j
.text:100073BA                 cmp     dword ptr [edi-0Eh], 5
.text:100073BE                 jnz     short loc_100073F5
.text:100073C0                 movzx   eax, byte ptr [edi+1]
.text:100073C4                 push    eax
.text:100073C5                 movzx   eax, byte ptr [edi]
.text:100073C8                 push    eax
.text:100073C9                 movzx   eax, byte ptr [edi-1]
.text:100073CD                 push    eax
.text:100073CE                 movzx   eax, byte ptr [edi-2]
.text:100073D2                 push    eax
.text:100073D3                 lea     eax, [ebp+var_50]
.text:100073D6                 push    offset aU_U_U_U ; "%u.%u.%u.%u"
.text:100073DB                 push    eax             ; LPWSTR
.text:100073DC                 call    ds:wsprintfW
.text:100073E2                 add     esp, 18h
.text:100073E5                 push    [ebp+arg_0]
.text:100073E8                 xor     esi, esi
.text:100073EA                 lea     eax, [ebp+var_50]
.text:100073ED                 call    sub_10006B95
.text:100073F2                 mov     esi, [ebp+var_C]
.text:100073F5
.text:100073F5 loc_100073F5:                           ; CODE XREF: sub_1000733C+82↑j
.text:100073F5                 inc     [ebp+var_4]
.text:100073F8                 mov     eax, [ebp+var_4]
.text:100073FB                 add     edi, 14h
.text:100073FE                 cmp     eax, [esi]
.text:10007400                 jb      short loc_100073BA
```

The next screenshot is of another matching function between the two samples.



```
.text:1000C244
.text:1000C244                 push    ebp
.text:1000C245                 mov     ebp, esp
.text:1000C247                 sub     esp, 7Ch
.text:1000C24A                 xor     eax, eax
.text:1000C24C                 xor     edx, edx
.text:1000C24E                 push    ebx
.text:1000C24F                 push    esi
.text:1000C250                 push    edi
.text:1000C251                 lea     edi, [ebp+var_5C]
.text:1000C254                 push    8
.text:1000C256                 pop     ecx
.text:1000C257                 rep stosd
.text:1000C259                 mov     ecx, edx
.text:1000C25B                 cmp     [ebp+arg_8], eax
.text:1000C25E                 jbe     short loc_1000C272
.text:1000C260
.text:1000C260 loc_1000C260:                           ; CODE XREF: sub_1000C244+2C↓j
.text:1000C260                 mov     eax, [ebp+arg_4]
.text:1000C263                 movzx   eax, word ptr [eax+ecx*2]
.text:1000C267                 inc     [ebp+eax*2+var_5C]
.text:1000C26C                 inc     ecx
.text:1000C26D                 cmp     ecx, [ebp+arg_8]
.text:1000C270                 jb      short loc_1000C260
.text:1000C272
.text:1000C272 loc_1000C272:                           ; CODE XREF: sub_1000C244+1A↑j
.text:1000C272                 mov     edi, [ebp+arg_10]
.text:1000C275                 xor     eax, eax
.text:1000C277                 push    0Fh
.text:1000C279                 pop     esi
.text:1000C27A                 inc     eax
.text:1000C27B                 mov     ebx, [edi]
.text:1000C27D
.text:1000C27D loc_1000C27D:                           ; CODE XREF: sub_1000C244+43↓j
.text:1000C27D                 cmp     [ebp+esi*2+var_5C], dx
.text:1000C282                 jnz     short loc_1000C289
.text:1000C284                 dec     esi
.text:1000C285                 cmp     esi, eax
.text:1000C287                 jnb     short loc_1000C27D
.text:1000C289
.text:1000C289 loc_1000C289:                           ; CODE XREF: sub_1000C244+3E↑j
.text:1000C289                 cmp     ebx, esi
.text:1000C28B                 cmova   ebx, esi
.text:1000C28E                 test    esi, esi
.text:1000C290                 jnz     short loc_1000C2C0
.text:1000C292                 mov     edx, [ebp+arg_C]
.text:1000C295                 mov     byte ptr [ebp+arg_8+1], al
.text:1000C298                 xor     eax, eax
.text:1000C29A                 mov     byte ptr [ebp+arg_8], 40h
.text:1000C29E                 mov     word ptr [ebp+arg_8+2], ax
.text:1000C2A2                 mov     ecx, [edx]
.text:1000C2A4                 mov     eax, [ebp+arg_8]
.text:1000C2A7                 mov     [ecx], eax
.text:1000C2A9                 add     dword ptr [edx], 4
.text:1000C2AC                 mov     ecx, [edx]
.text:1000C2AE                 mov     [ecx], eax
.text:1000C2B0                 add     dword ptr [edx], 4
.text:1000C2B3                 mov     dword ptr [edi], 1
.text:1000C2B9
.text:1000C2B9 loc_1000C2B9:                           ; CODE XREF: sub_1000C244+376↓j
.text:1000C2B9                 xor     eax, eax
.text:1000C2BB                 jmp     loc_1000C5C2
.text:1000C2C0 ; ---------------------------------------------------------------------------
.text:1000C2C0
.text:1000C2C0 loc_1000C2C0:                           ; CODE XREF: sub_1000C244+4C↑j
.text:1000C2C0                 mov     edx, eax
.text:1000C2C2                 mov     [ebp+var_10], edx
.text:1000C2C5                 cmp     esi, eax
.text:1000C2C7                 jbe     short loc_1000C2DA
.text:1000C2C9                 xor     ecx, ecx
.text:1000C2CB
.text:1000C2CB loc_1000C2CB:                           ; CODE XREF: sub_1000C244+91↓j
.text:1000C2CB                 cmp     [ebp+edx*2+var_5C], cx
.text:1000C2D0                 jnz     short loc_1000C2D7
.text:1000C2D2                 inc     edx
.text:1000C2D3                 cmp     edx, esi
.text:1000C2D5                 jb      short loc_1000C2CB
.text:1000C2D7
.text:1000C2D7 loc_1000C2D7:                           ; CODE XREF: sub_1000C244+8C↑j
.text:1000C2D7                 mov     [ebp+var_10], edx
.text:1000C2DA
.text:1000C2DA loc_1000C2DA:                           ; CODE XREF: sub_1000C244+83↑j
.text:1000C2DA                 cmp     ebx, edx
.text:1000C2DC                 mov     ecx, eax
.text:1000C2DE                 mov     edi, eax
.text:1000C2E0                 cmovb   ebx, edx
```

```
text:1000C4B4
text:1000C4B4                 push    ebp
text:1000C4B5                 mov     ebp, esp
text:1000C4B7                 sub     esp, 7Ch
text:1000C4BA                 xor     eax, eax
text:1000C4BC                 xor     edx, edx
text:1000C4BE                 push    ebx
text:1000C4BF                 push    esi
text:1000C4C0                 push    edi
text:1000C4C1                 lea     edi, [ebp+var_5C]
text:1000C4C4                 push    8
text:1000C4C6                 pop     ecx
text:1000C4C7                 rep stosd
text:1000C4C9                 mov     ecx, edx
text:1000C4CB                 cmp     [ebp+arg_8], eax
text:1000C4CE                 jbe     short loc_1000C4E2
text:1000C4D0
text:1000C4D0 loc_1000C4D0:                           ; CODE XREF: sub_1000C4B4+2C↓j
text:1000C4D0                 mov     eax, [ebp+arg_4]
text:1000C4D3                 movzx   eax, word ptr [eax+ecx*2]
text:1000C4D7                 inc     [ebp+eax*2+var_5C]
text:1000C4DC                 inc     ecx
text:1000C4DD                 cmp     ecx, [ebp+arg_8]
text:1000C4E0                 jb      short loc_1000C4D0
text:1000C4E2
text:1000C4E2 loc_1000C4E2:                           ; CODE XREF: sub_1000C4B4+1A↑j
text:1000C4E2                 mov     edi, [ebp+arg_10]
text:1000C4E5                 xor     eax, eax
text:1000C4E7                 push    0Fh
text:1000C4E9                 pop     esi
text:1000C4EA                 inc     eax
text:1000C4EB                 mov     ebx, [edi]
text:1000C4ED
text:1000C4ED loc_1000C4ED:                           ; CODE XREF: sub_1000C4B4+43↓j
text:1000C4ED                 cmp     [ebp+esi*2+var_5C], dx
text:1000C4F2                 jnz     short loc_1000C4F9
text:1000C4F4                 dec     esi
text:1000C4F5                 cmp     esi, eax
text:1000C4F7                 jnb     short loc_1000C4ED
text:1000C4F9
text:1000C4F9 loc_1000C4F9:                           ; CODE XREF: sub_1000C4B4+3E↑j
text:1000C4F9                 cmp     ebx, esi
text:1000C4FB                 cmova   ebx, esi
text:1000C4FE                 test    esi, esi
text:1000C500                 jnz     short loc_1000C530
text:1000C502                 mov     edx, [ebp+arg_C]
text:1000C505                 mov     byte ptr [ebp+arg_8+1], al
text:1000C508                 xor     eax, eax
text:1000C50A                 mov     byte ptr [ebp+arg_8], 40h
text:1000C50E                 mov     word ptr [ebp+arg_8+2], ax
text:1000C512                 mov     ecx, [edx]
text:1000C514                 mov     eax, [ebp+arg_8]
text:1000C517                 mov     [ecx], eax
text:1000C519                 add     dword ptr [edx], 4
text:1000C51C                 mov     ecx, [edx]
text:1000C51E                 mov     [ecx], eax
text:1000C520                 add     dword ptr [edx], 4
text:1000C523                 mov     dword ptr [edi], 1
text:1000C529
text:1000C529 loc_1000C529:                           ; CODE XREF: sub_1000C4B4+376↓j
text:1000C529                 xor     eax, eax
text:1000C52B                 jmp     loc_1000C832
text:1000C530 ; ---------------------------------------------------------------------------
text:1000C530
text:1000C530 loc_1000C530:                           ; CODE XREF: sub_1000C4B4+4C↑j
text:1000C530                 mov     edx, eax
text:1000C532                 mov     [ebp+var_10], edx
text:1000C535                 cmp     esi, eax
text:1000C537                 jbe     short loc_1000C54A
text:1000C539                 xor     ecx, ecx
text:1000C53B
text:1000C53B loc_1000C53B:                           ; CODE XREF: sub_1000C4B4+91↓j
text:1000C53B                 cmp     [ebp+edx*2+var_5C], cx
text:1000C540                 jnz     short loc_1000C547
text:1000C542                 inc     edx
text:1000C543                 cmp     edx, esi
text:1000C545                 jb      short loc_1000C53B
text:1000C547
text:1000C547 loc_1000C547:                           ; CODE XREF: sub_1000C4B4+8C↑j
text:1000C547                 mov     [ebp+var_10], edx
text:1000C54A
text:1000C54A loc_1000C54A:                           ; CODE XREF: sub_1000C4B4+83↑j
text:1000C54A                 cmp     ebx, edx
text:1000C54C                 mov     ecx, eax
text:1000C54E                 mov     edi, eax
text:1000C550                 cmovb   ebx, edx
```

As you can see in this attack, and in many other cases, malware authors constantly reuse their code. By recognizing code reuse, you force malware authors to rewrite code and come up with new techniques to avoid detection. This changes the playing field and makes it far less cost effective for malware authors and cyber crime organizations.

IOCs:

630325cac09ac3fab908f903e3b00d0dadd5fdaa0875ed8496fcbb97a558d0da

8ebc97e05c8e1073bda2efb6f4d00ad7e789260afa2c276f0c72740b838a0a93

579fd8a0385482fb4c789561a30b09f25671e86422f40ef5cca2036b28f99648

Jay Rosenberg