

Objective-See

objective-see.com/blog/blog_0x1D.html

HandBrake Hacked!

› osx/proton (re)appears

5/06/2017

love these blog posts? support my tools & writing on [patreon!](#) Mahalo :)



Want to play along? I've shared the malware, which can be downloaded [here](#) (password: infect3d). Please don't infect yourself!

Background

Today started like pretty much every other day in Hawaii, surf & beach ;)



However, around noon, Eric Holtam ([@eholtam](#)) tweeted the following:



Eric Holtam
[@eholtam](#)

[Follow](#)

Heads up, Handbrake hacked to install malware.
Check your distros

Mahalo to [@mikeymikey](#) to notifying me about this tweet!

Handbrake is an 'open-source video transcoder.' Heading over to their website ([handbrake.fr](#)), we can see they've added a 'Security Alert' for Mac Users:



HandBrake

The open source video transcoder

HandBrake is a tool for converting video from nearly any format to a selection of modern, widely supported codecs.

Reasons you'll love HandBrake:

- Convert video from nearly any format
- Free and Open Source
- Multi-Platform (Windows, Mac and Linux)

Download HandBrake 1.0.7

For Mac OS X 10.7 or later

[\(Other Platforms\)](#)

It's free!



SECURITY ALERT: [Mac Users Please Read](#)

Following the url, links to a rather dire security alert, [Mirror Download Server Compromised](#):

SECURITY WARNING

Anyone who has downloaded HandBrake on Mac between [02/May/2017 14:30 UTC] and [06/May/2017 11:00 UTC] needs to verify the SHA1 / 256 sum of the file before running it.

Anyone who has installed HandBrake for Mac needs to verify their system is not infected with a Trojan. You have 50/50 chance if you've downloaded HandBrake during this period.

Yikes!

The security alert also provided a hash of the disk image (0935a43ca90c6c419a49e4f8f1d75e68cd70b274) that was trojaned by the hackers.

Hopping over to VirusTotal, we can see that while this .dmg was submitted for analysis, no anti-virus engines are currently flagging it. No surprise there :|



SHA256:	013623e5e50449bbdf6943549d8224a122aa6c42bd3300a1bd2b743b01ae6793
File name:	HandBrake-1.0.7-2.dmg
Detection ratio:	0 / 55
Analysis date:	2017-05-06 20:12:15 UTC (5 minutes ago)

Analysis

Once we have a copy of the infected .dmg (I've shared it [here](#) password: infect3d), analysis can commence. Since it's the weekend - I'm going to take the 'lazy' (efficient?) route and basically just run the infected application and see what happens :)

In order to facilitate malware analysis I wrote a simple user-mode 'process monitor' library that allows us to easily track what application is doing - in terms of spawning other processes, etc:

```
ProcessMonitor* procMon = nil;
```

```
//block
```

```
ProcessCallbackBlock block = ^(Process* newProcess)
```

```
{
```

```
    NSLog(@"new process: %@", newProcess);
```

```
};
```

```
//start monitoring!  
procMon = [[ProcessMonitor alloc] init];  
[procMon start:block];  
  
//run loop  
[[NSRunLoop currentRunLoop] run];
```

Running this code and executing the infected Handbrake application, we can see exactly what's going on:

```
[new process]  
pid=1368  
binary=/Volumes/HandBrake/HandBrake.app/Contents/MacOS/HandBrake  
signatureStatus = "-67062 (unsigned)"
```

```
[new process]  
pid=1370  
binary=path=/bin/sh  
args: "-c", "pgrep -x activity_agent && echo Queue.hbqueue"
```

```
[new process]  
pid=1371  
binary=/usr/bin/unzip  
args: "-P",  
"qzyuzacCELFEYiJ52mhjEC7HYI4eUPAR1EEf63oQ5iTkuNIhzRk2JUKF4IXTRdiQ",  
"/Volumes/HandBrake/HandBrake.app/Contents/Resources/HBPlayerHUDMainController.nib",  
"-d", "/tmp"
```

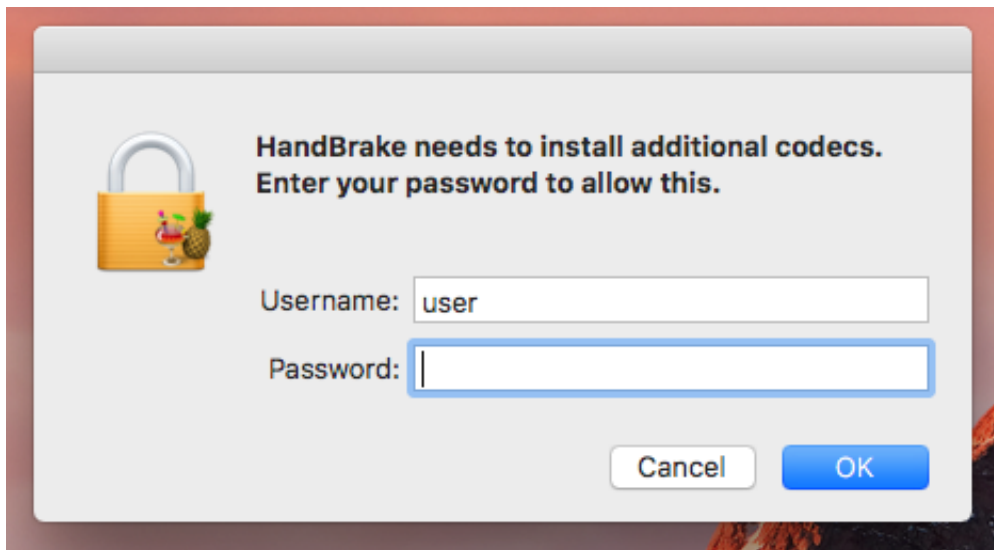
```
[new process]  
pid=1372  
binary=/usr/bin/open  
args: "/tmp/HandBrake.app"
```

So yah, when run, the infected Handbrake application:

1. unzips Contents/Resources/HBPlayerHUDMainController.nib to /tmp/HandBrake.app.
This 'nib' is a password protected zip file who's password is:
qzyuzacCELFEYiJ52mhjEC7HYI4eUPAR1EEf63oQ5iTkuNIhzRk2JUKF4IXTRdiQ
2. launches (opens) /tmp/HandBrake.app

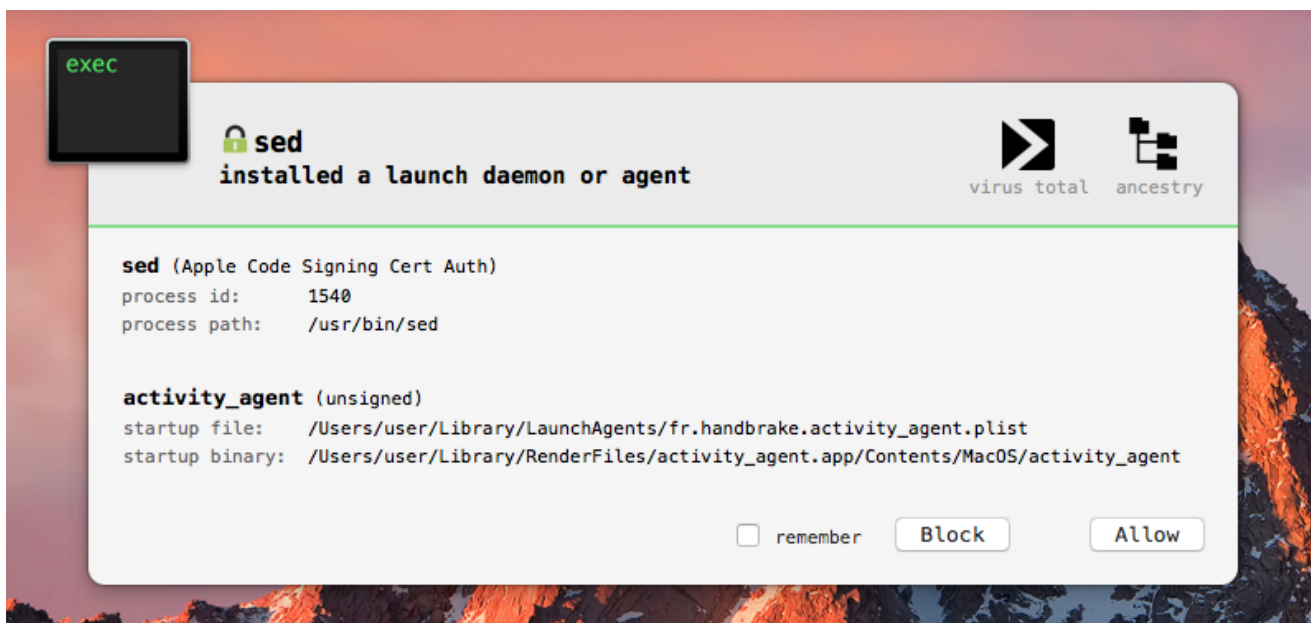
Once the /tmp/HandBrake.app is launched, it displays a (fake) authentication popup - which

is how the malware attempts to elevate its privileges:



If the user is tricked into providing a user name and password the malware will install itself (/tmp/HandBrake.app) persistently as: 'activity_agent.app'.

Thankful, BlockBlock can alert us of this fact:



Dumping the Launch Agent plist file (fr.handbrake.activity_agent.plist), we can see the malware has been set to automatically start each time the user logs in:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>KeepAlive</key>
```


```
<true/>
...
<key>ProgramArguments</key>
<array>
  <string>/Users/user/Library/RenderFiles/activity_agent.app/
    Contents/MacOS/activity_agent</string>
</array>
<key>RunAtLoad</key>
<true/>

</dict>
</plist>
```

According to the HandBrake advisory, the malware's persistent component, activity_agent.app is a 'a new variant of OSX.PROTON'

Proton (variant 'A') was discussed earlier this year by the media (for example, see: ["Hackers Selling Undetectable Proton Malware for macOS in 40 BTC"](#))

Though Apple released an XProtect signature for it, the sample was never publicly shared. However, the malware author was kind enough to describe ('advertise') its capabilities:



Vendor

Joined: Jan 15, 2017

Messages: 12

Likes Received: 3

Greetings everyone.

Newest and only macOS RAT in market!

PROTON

PROTON is a professional FUD surveillance & control solution, with which you can do almost everything with target's Mac.

- Execute any bash command under root
- Monitor keystrokes (we even have a tariff allowing to log passwords)
- Get notified each time your client enters something
- Upload files to remote machine
- Download files from remote machine
- Connect directly via SSH/VNC to remote machine
- Get screenshots/webcamshots
- Satisfy gatekeeper by choosing signed bundle
- Develop your own panel/program bundle with our extensive API
- Get updates on the air

With the HandBrake hack, finally now we have a variant for analysis :). Initial triage confirms, yes this a variant of OSX/Proton ('B'), although some of the features found in the 'A' variant, (such as the ability to take screenshots) are not present.

Again, unsurprisingly this new variant of OSX/Proton is also currently undetected by any anti-virus engines on VirusTotal:



SHA256:	bec7bfc5375dd1c4bac23121c8d83b80f484cd53261f0d3f9f3f64177e4b7caf
File name:	activity_agent
Detection ratio:	0 / 56
Analysis date:	2017-05-06 11:20:09 UTC (10 hours, 40 minutes ago)

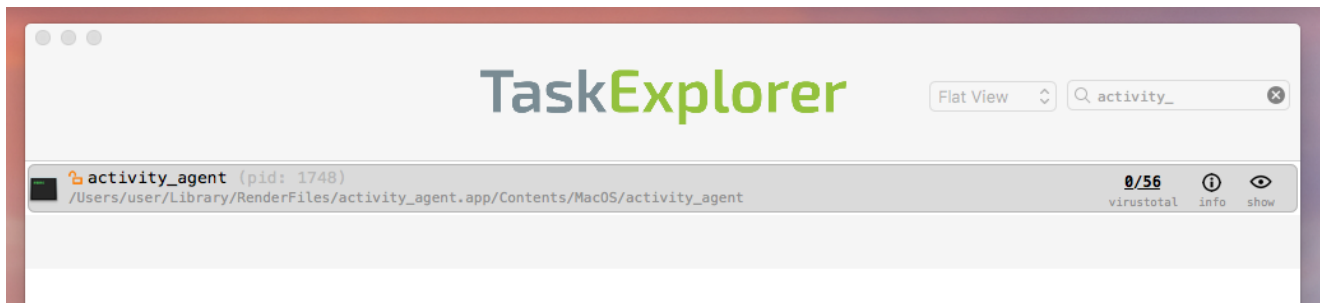
Interested in the technical details of OSX/Proton? I wrote it in a new blog, "[OSX/Proton.B \(a brief analysis, at 6 miles up\)](#)".

Conclusions

As with KeRanger and Keydnep, hackers targeted an official distribution website of legitimate macOS software. With access to HandBrake's mirror, they trojaned the legitimate application, meaning any user who downloaded the application would inadvertently infect themselves!

Luckily the trojaned disk image was only online for a few days. However as is often (always!?) the case, no anti-virus products flagged the malware :(So if you recently download HandBrake, unless you were running something like BlockBlock you'd likely have been infected.

To check if you're infected, look for the following:



- a process named 'activity_agent'
- an application name 'activity_agent.app' in ~/Library/RenderFiles/
- a plist file: '~/Library/LaunchAgents/fr.handbrake.activity_agent.plist'

Apple has now also pushed out an XProtect signature, meaning that all new infections should be thwarted. Hooray!

```
$ cat /System/Library/CoreServices/XProtect.bundle/Contents/Resources/XProtect.yara
```

```
private rule Macho
{
  meta:
  description = "private rule to match Mach-O binaries"
  condition:
  uint32(0) == 0xfeedface or uint32(0) == 0xcefaedfe or uint32(0) == 0xfeedfacf
  or uint32(0) == 0xcffaedfe or uint32(0) == 0xcafebabe or uint32(0) == 0xbebafeca
}
```

```

rule XProtect_OSX_Proton_B
{
  meta:
    description = "OSX.Proton.B"

  condition:
Macho and filesize < 800000 and hash.sha1(0, filesize) ==
"a8ea82ee767091098b0e275a80d25d3bc79e0cea"
}

```

Ok, so kudos to Apple for the quick turn around on signatures....but the signature is:

- just a SHA-1 hash
- matches on the trojaned Handbrake binary
(HandBrake.app/Contents/MacOS/HandBrake on the .dmg)

```

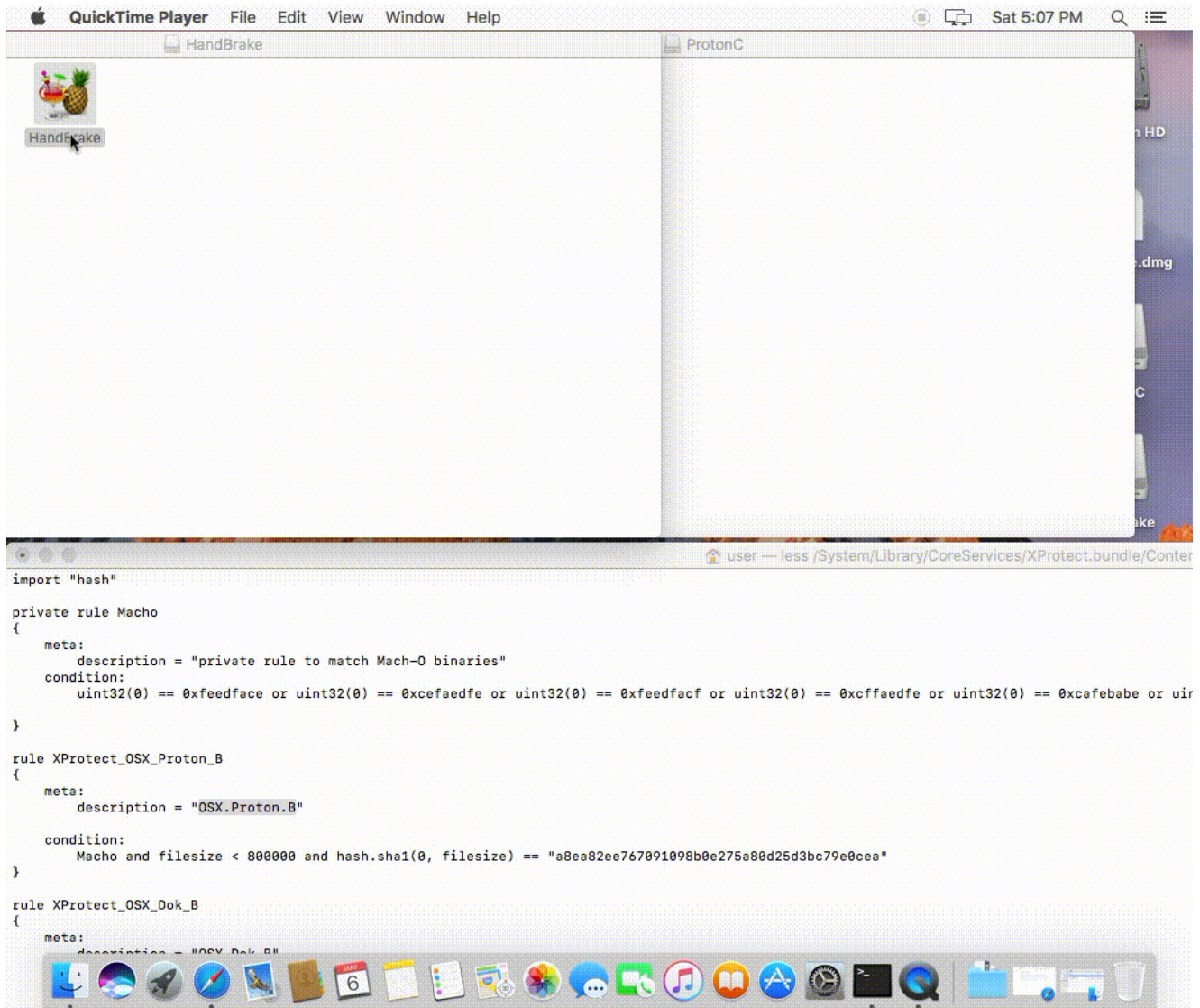
$ shasum -a 1 /Volumes/HandBrake/HandBrake.app/Contents/MacOS/HandBrake

a8ea82ee767091098b0e275a80d25d3bc79e0cea
/Volumes/HandBrake/HandBrake.app/Contents/MacOS/HandBrake

```

This means if the malware authors used any other infection vector, or even just recompiled the trojaned binary, this signature would no longer flag the malware :/ Do people still say 'yolo'?

And even if you don't have source code, you can just flip a single bit in the binary to thwart the signature. To test this, I changed the final byte in HandBrake.app/Contents/MacOS/HandBrake from an 0x00 to 0xFF. While this doesn't impact functionality of the binary, it changes its SHA-1 hash, meaning the malware is no longer blocked by XProtect. The following, first shows XProtect 'blocking' OSX/Proton.B. However, a second modified version is allowed to run, as the signature no longer matches:



© 2017 objective-see llc