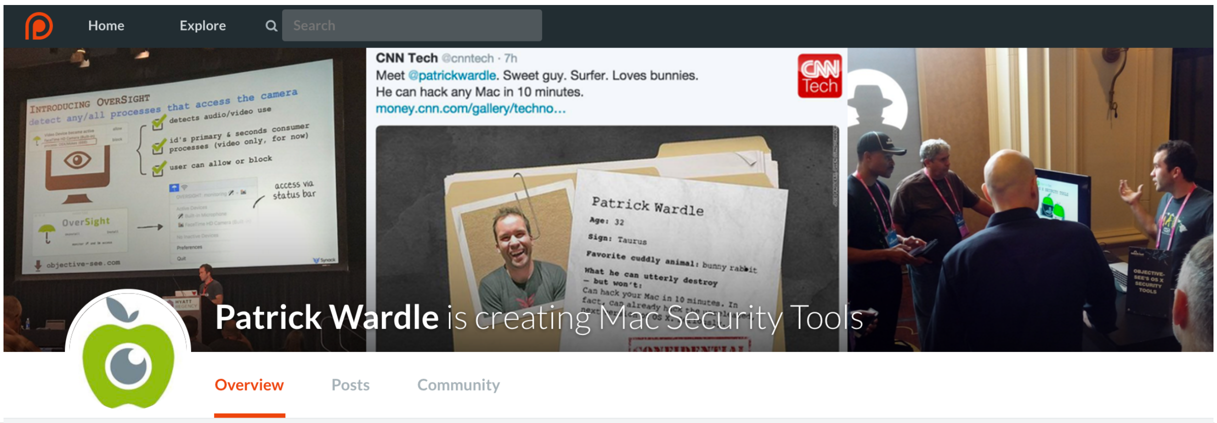
Apple Approved Malware
malicious code ...now notarized!? #2020
by: Patrick Wardle / August 30, 2020

Love these blog posts and/or want to support my research and tools? You can support them via my **Patreon** page!


Home
Explore
Search
CNN Tech @cnntech · 7h
Meet @patrickwardle. Sweet guy. Surfer. Loves bunnies. He can hack any Mac in 10 minutes.
money.cnn.com/gallery/techno...
Patrick Wardle is creating Mac Security Tools
Overview
Posts
Community

```
📝 👾 Want to play along?

I've added the **samples** ('OSX.Shlayer') to our malware
collection (password: infect3d)

…please don't infect yourself!
```

Background

It wasn't too long ago, that Apple's website stated:

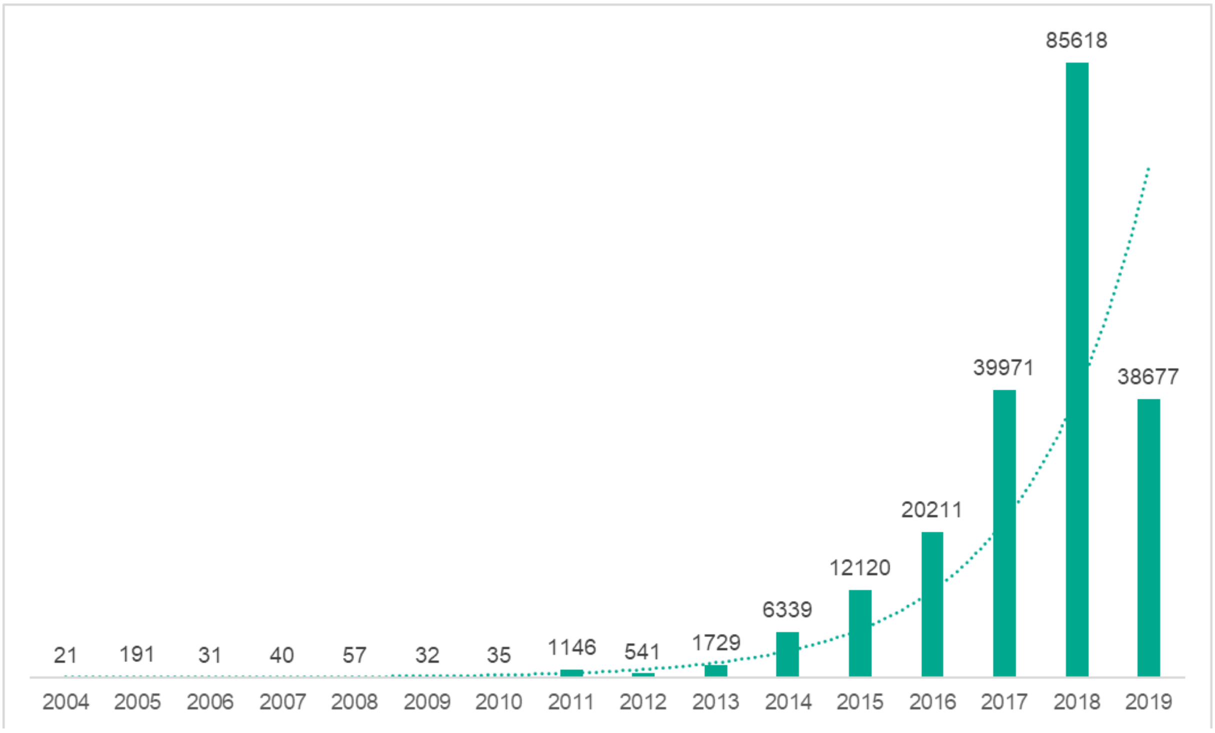
"[Macs] doesn't get PC viruses. A Mac isn't susceptible to the thousands of viruses plaguing Windows-based computers. That's thanks to built-in defenses in Mac OS X that keep you safe without any work on your part" -Apple.com

The "truth" of this nuanced statement lies in the fact that due to inherent cross-platform incompatibilities (not Apple's "defenses"): a native Windows virus cannot directly execute on macOS.

However even this claim is rather subjective as was highlighted in 2019 by a Windows adware specimen **targeting macOS users.** The adware was packaged with a cross-platform framework (Mono) that allowed Windows binaries (.exes) to "natively" run on macOS!

And, even back in 2012, thanks to Java, cross-platform malware could be found **targeting both Windows and macOS.**

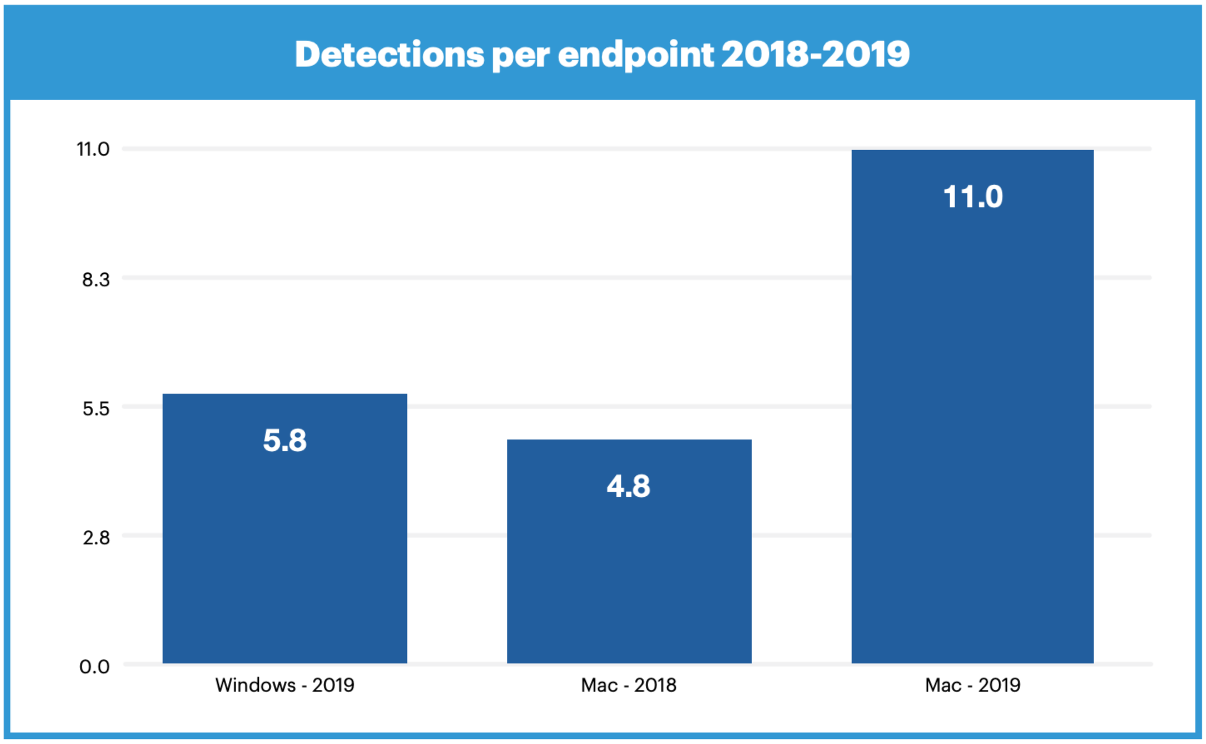
Today, malicious code targeting macOS, unfortunately, is far too common. Kaspersky, in a 2019 report titled "**Threats to macOS users**", noted a sharp uptick in threats targeting Macs:



"Threats to macOS users" (Kaspersky)

…while Malwarebytes stated in their "**2020 State of Malware Report**":

"And for the first time ever, Macs outpaced Windows PCs in number of threats detected per endpoint." -Malwarebytes



Threats per endpoint, Macs vs. Windows (Malwarebytes)

It is important to note these statistics include both adware (and potentially unwanted programs). And the reality is, if a Mac user is infected with malicious code, more than likely, it not going to be some advanced nationstate backdoor, but rather adware (or malware that installs various adware). Unfortunately, such adware and adware campaigns are rather prolific, and their prevalence is only increasing as Mac become ever more popular:

"The vast majority of threats for macOS in 2019 were in the AdWare category." -Kaspersky

It is also important not to underestimate the potential impact of adware, upon its victims. The noted security researcher, **Thomas Reed** articulates this well in a **recent writeup**:

"However, adware and PUPs can actually be far more invasive and dangerous on the Mac than "real" malware. They can intercept and decrypt all network traffic, create hidden users with static passwords, make insecure changes to system settings, and generally dig their roots deep into the system so that it is incredibly challenging to eradicate completely." -Thomas Reed

`For more on the topic of adware and its impact, see:`

- **`"Mac adware is more sophisticated and dangerous than traditional Mac malware"`**
- **`"The line between adware and malware has become increasingly blurred, researcher finds"`**

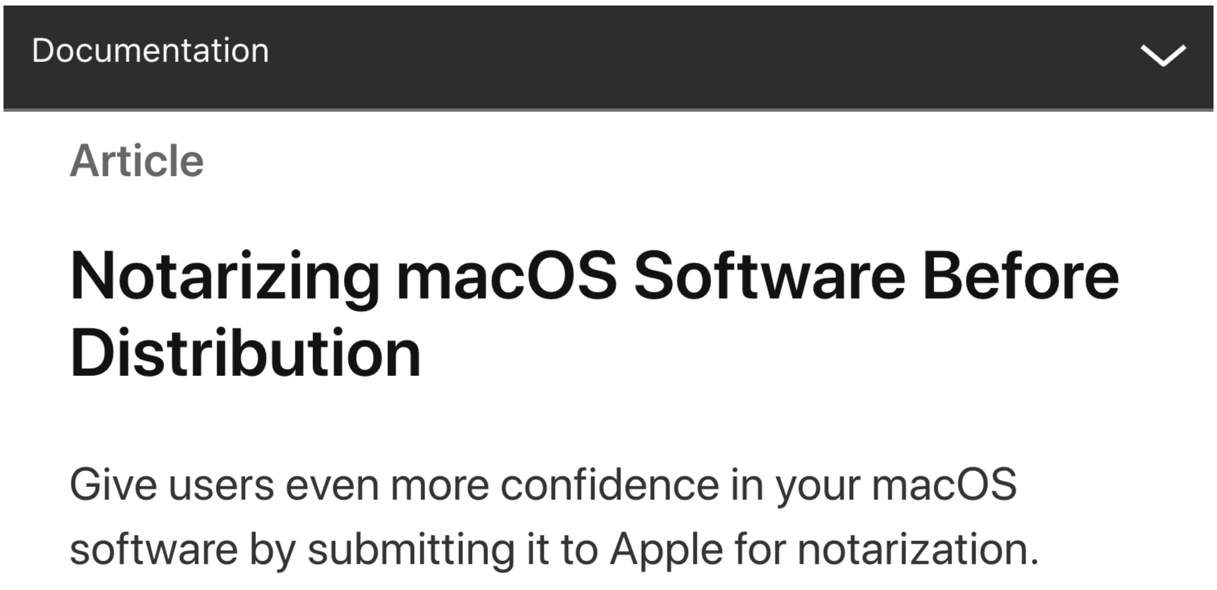
Clearly to in order to protect both users and the perception of security, Apple had to take steps to address the ever rising tide of malicious code targeting macOS.

Their (most recent) answer? Code Notarization Requirements.

Notarization

The main goal of notarization is to allow Apple to "*scan …software for malicious content*" before it is distributed to macOS users. The idea was that malicious code would of course, not be notarized and thus the majority to attacks targeting macOS users (adware campaigns, etc) would be thwarted.

Apple introduced notarization requirements in macOS 10.15 (Catalina), detailing the topic in an apply named document, "**Notarizing macOS Software**":



Documentation

Article

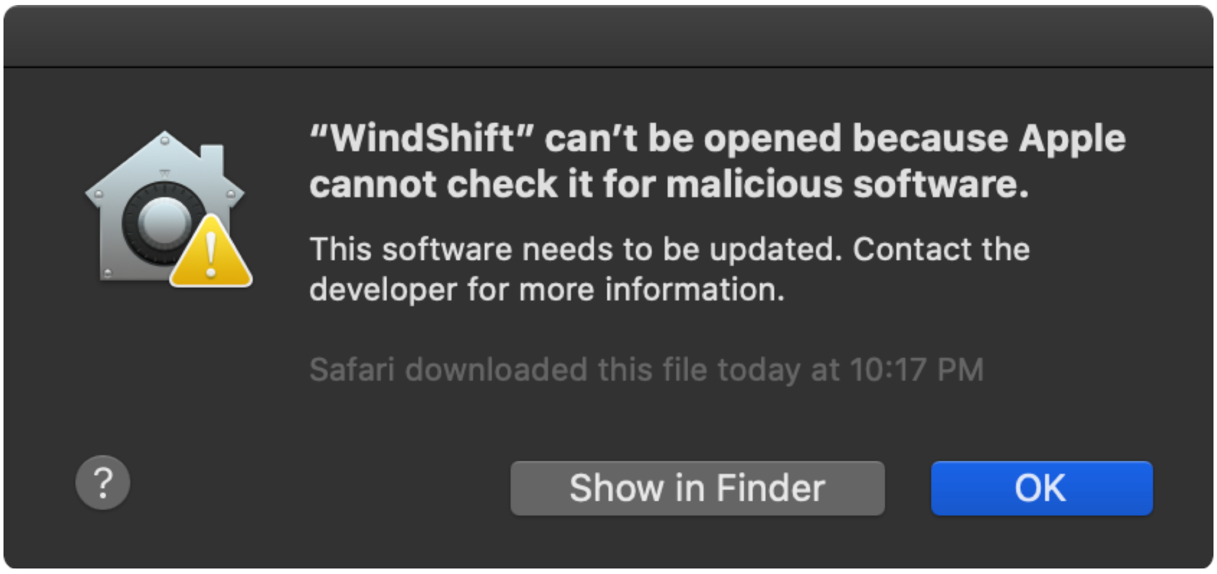
Notarizing macOS Software Before Distribution

Give users even more confidence in your macOS software by submitting it to Apple for notarization.

In short, developers must now submit their applications to Apple before distribution to macOS users. This ensures that Apple can inspect (and approve) all software before it is allowed to run on (recent versions of) macOS:

"Notarization gives users more confidence that the Developer ID-signed software you distribute has been checked by Apple for malicious components." -Apple.com

If software has *not* been notarized, it will be blocked by macOS (with no option to run it, via the alert prompt):



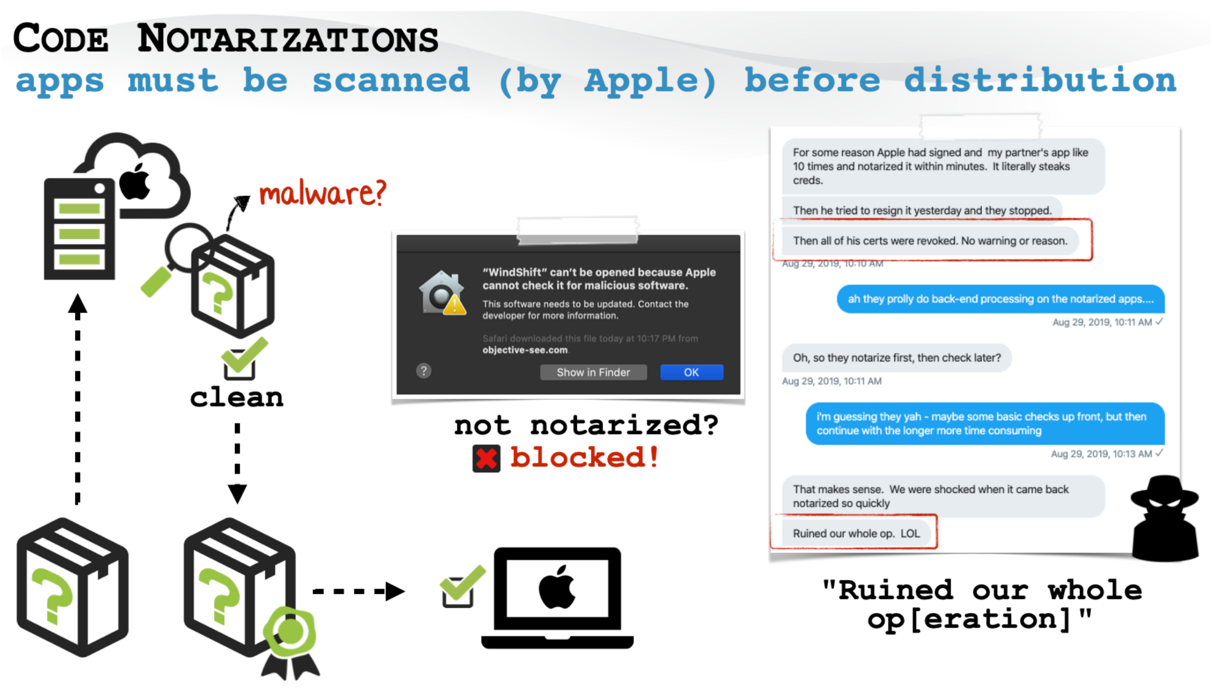
"WindShift" can't be opened because Apple cannot check it for malicious software.

This software needs to be updated. Contact the developer for more information.

Safari downloaded this file today at 10:17 PM

unnotarized software: blocked!

The following provides a conceptual overview of the notarization process, and its impact to both malware and hacking operations:


Code Notarizations
apps must be scanned (by Apple) before distribution
malware?
clean
not notarized?
blocked!
"Ruined our whole op[eration]"
Show in Finder
OK

"WindShift" can't be opened because Apple cannot check it for malicious software.

This software needs to be updated. Contact the developer for more information.

Safari downloaded this file today at 10:17 PM from **objective-see.com**.

For some reason Apple had signed and my partner's app like 10 times and notarized it within minutes. It literally steaks creds.

Then he tried to resign it yesterday and they stopped.

Then all of his certs were revoked. No warning or reason.

Aug 29, 2019, 10:10 AM

ah they prolly do back-end processing on the notarized apps....

Aug 29, 2019, 10:11 AM ✓

Oh, so they notarize first, then check later?

Aug 29, 2019, 10:11 AM

i'm guessing they yah - maybe some basic checks up front, but then continue with the longer more time consuming

Aug 29, 2019, 10:13 AM ✓

That makes sense. We were shocked when it came back notarized so quickly

Ruined our whole op. LOL

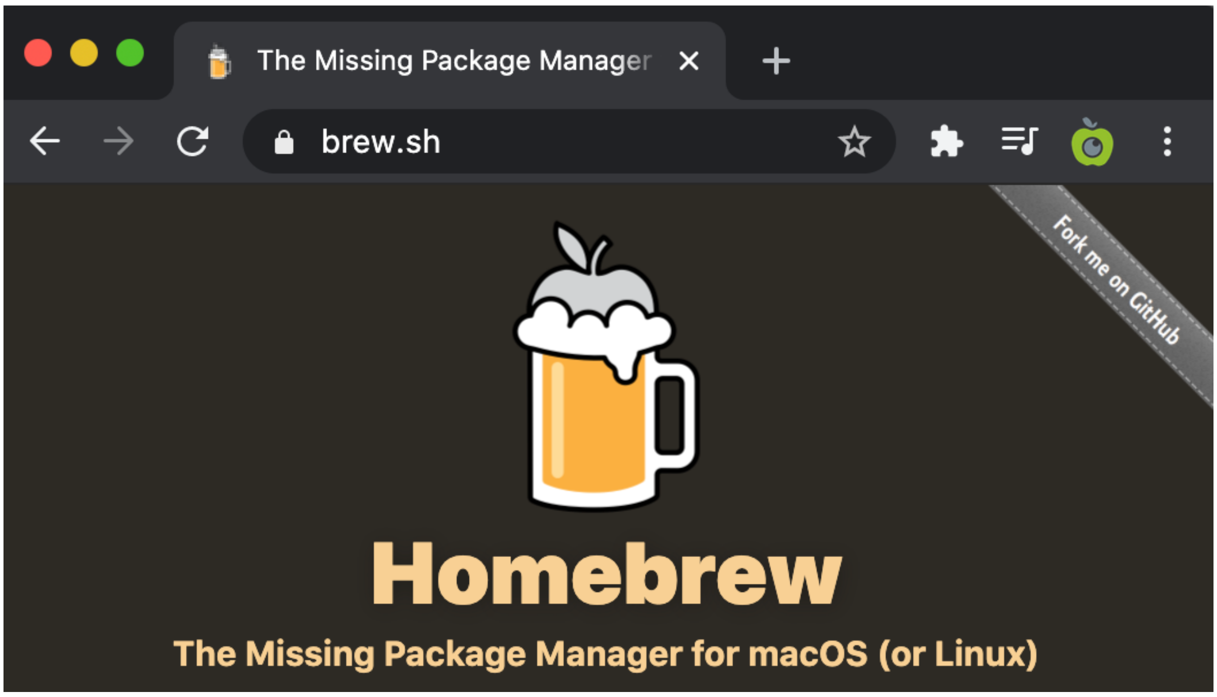
With the goal of stymieing the influx of malicious code targeting macOS, notarization seemed like a promising idea.
…sadly, not all promises are kept.

`For more on the topic of notarization see:`

- **`"How notarization works"`**
- **`"App Notarization: A Quick Primer"`**

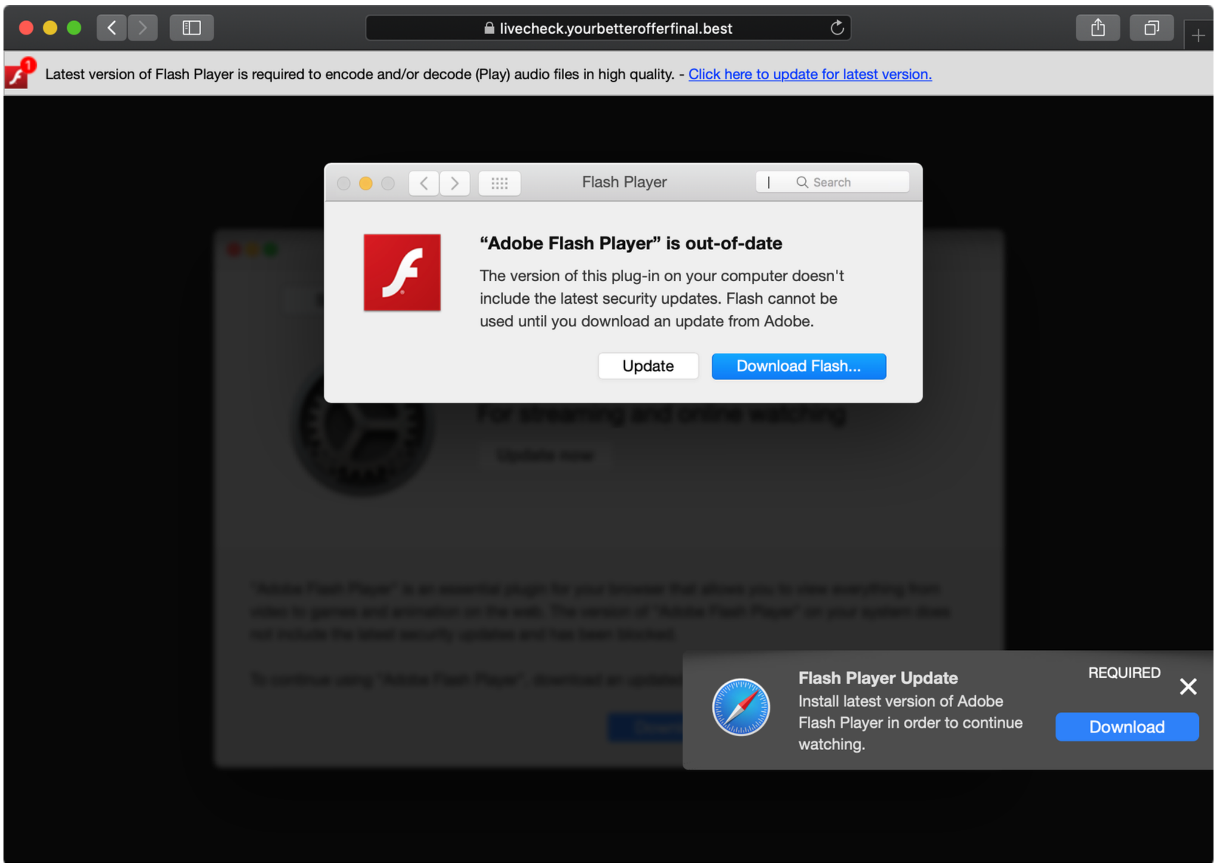
A Notarized Adware Campaign

Many developers are familiar with the Homebrew, hosted at **brew.sh**:


The Missing Package Manager
brew.sh
Fork me on GitHub
Homebrew
The Missing Package Manager for macOS (or Linux)

the (legitimate) Homebrew website

On Friday, twitter user **Peter Dantini** (@PokeCaptain) noticed that the website `homebrew.sh` (not to be confused with the legitimate Homebrew website **brew.sh**), was hosting an active adware campaign. If a user inadvertently visited `homebrew.sh`, after various redirects an update for “Adobe Flash Player” would be aggressively recommended:



Latest version of Flash Player is required to encode and/or decode (Play) audio files in high quality. - Click here to update for latest version.

“Adobe Flash Player” is out-of-date

The version of this plug-in on your computer doesn't include the latest security updates. Flash cannot be used until you download an update from Adobe.

Flash Player Update

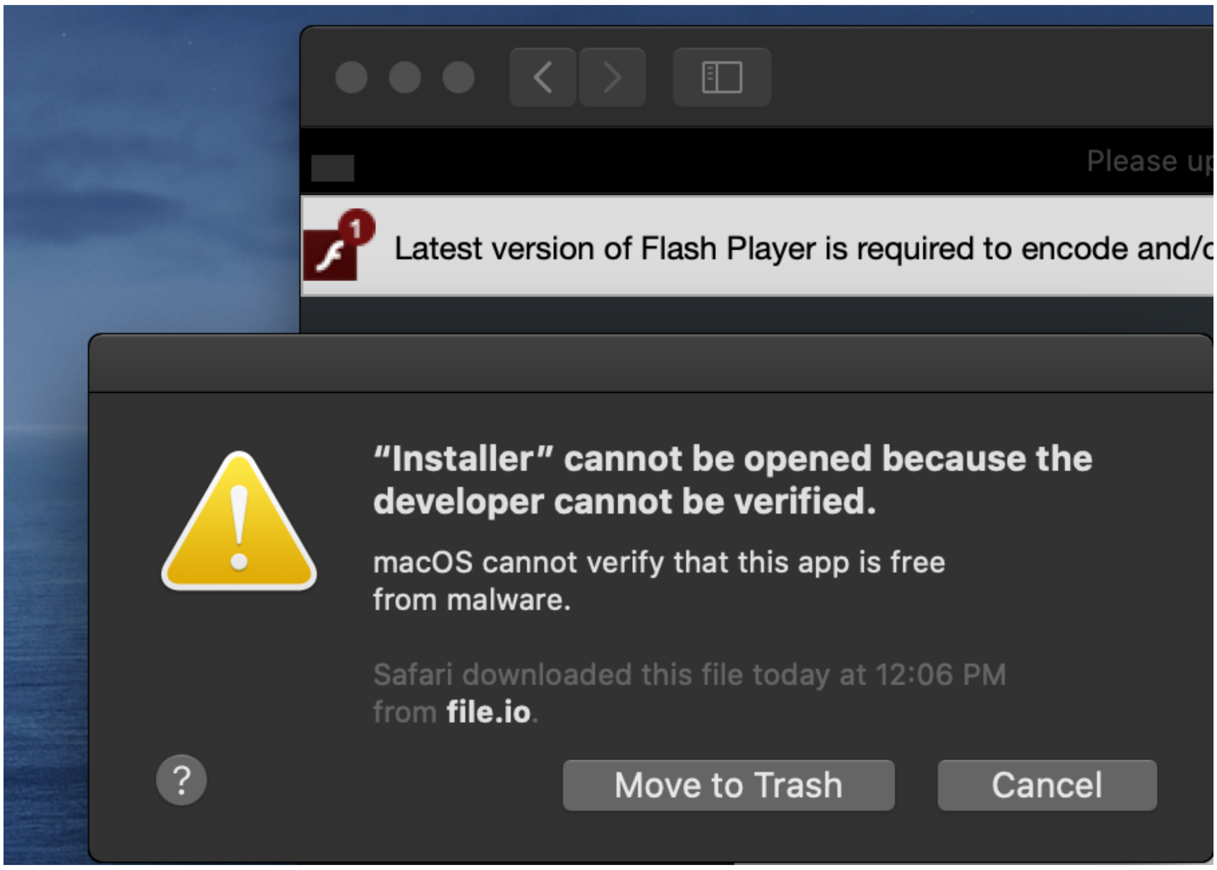
Install latest version of Adobe Flash Player in order to continue watching.

(fake) Abode Flash Player update(s)

```
Kudos to Peter for uncovering this adware campaign, and
sharing the details with me!

                        Mahalo Peter! 🙏🏽
```

Normally such campaigns are rather prosaic and utilize *unnotarized* code. As such, they are normally stopped short in their tracks by Apple's new notarization requirements. For example here, we have a similar adware campaign leveraging *unnotarized* payloads …that, as expected macOS will block:


Please up

Latest version of Flash Player is required to encode and/c

"Installer" cannot be opened because the developer cannot be verified.

macOS cannot verify that this app is free from malware.

Safari downloaded this file today at 12:06 PM from **file.io**.

unnotarized adware? blocked!

```
Note that in the above alert, the only options are "Move to
Trash" and "Cancel".

There is no option to allow the user to run the unnotarized
software.
```

Interestingly, Peter noticed the campaign originating from `homebrew.sh`, leveraged adware payloads were actually fully notarized! 😱

We can confirm the payloads are indeed notarized via the `spctl` command (note the "`source=Notarized Developer ID`"):

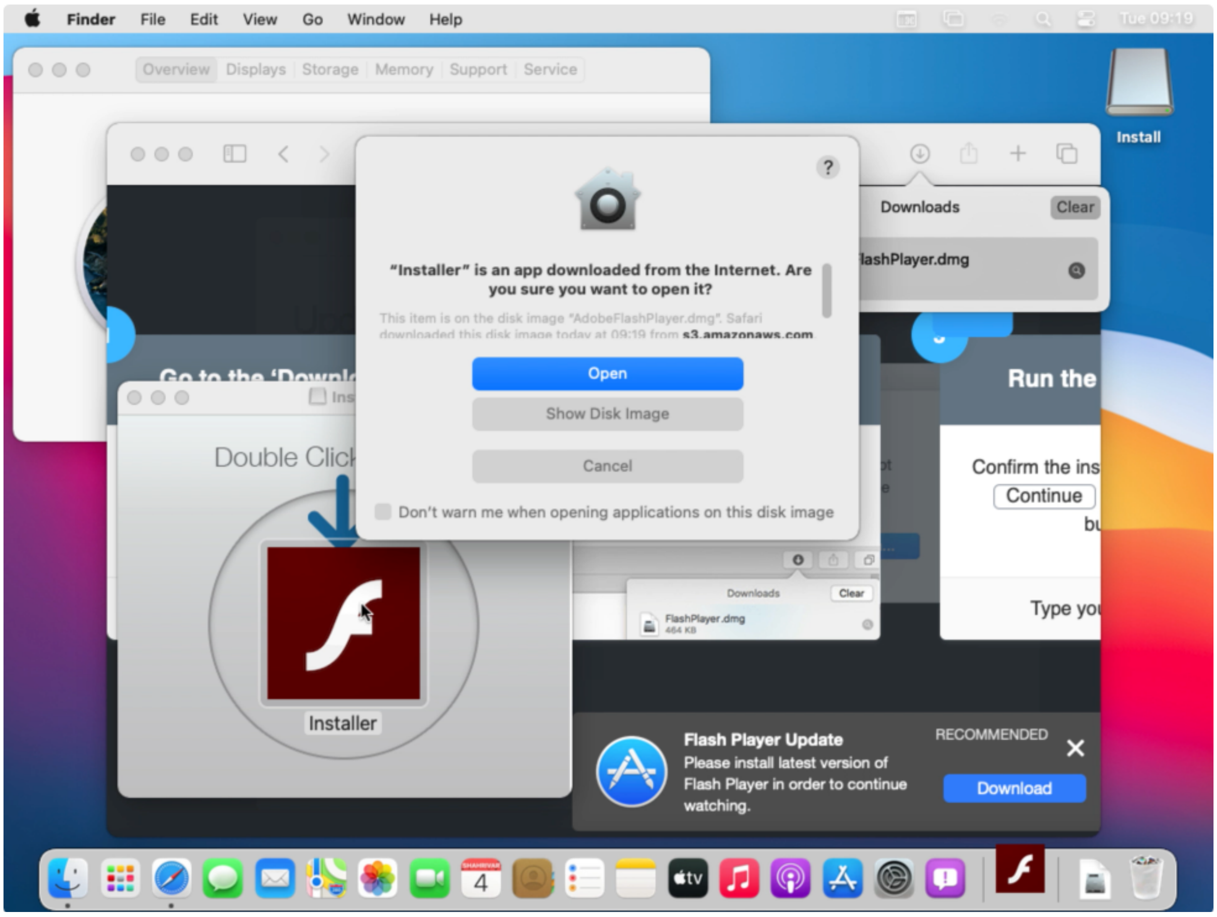
```
$ spctl -a -vvv -t install
/Volumes/Install/Installer.app
/Volumes/Install/Installer.app: accepted
source=Notarized Developer ID
origin=Developer ID Application: Morgan Sipe
(4X5KZ42L4B)

$ spctl -a -vvv -t install
/Users/patrick/Downloads/Player.pkg
/Users/patrick/Downloads/Player.pkg: accepted
source=Notarized Developer ID
origin=Developer ID Installer: Darien Watkins
(NC43XU5Z95)
```

As far as I know, this is a first: **malicious code gaining Apple's notarization "stamp of approval".**

What does this mean?

- These malicious payloads were submitted to Apple, prior to distribution.
- Apple scanned and apparently detecting no malice, (inadvertently) notarized them.
- Now notarized, these malicious payloads are allowed to run …even on macOS Big Sur.
- Again, due to their notarization status, users will (quite likely), fully trust these malicious samples.



“Installer” is an app downloaded from the Internet. Are you sure you want to open it?

This item is on the disk image "AdobeFlashPlayer.dmg". Safari downloaded this disk image today at 09:19 from s3.amazonaws.com.

Open

Show Disk Image

Cancel

Don't warn me when opening applications on this disk image

Flash Player Update

Please install latest version of Flash Player in order to continue watching.

notarized malware on Big Sur? ...yups

Triaging the Payloads

So what are the notarized payloads?

```
$ shasum *
43a44d4f58774157857d04d67a9fef7045dacb2f
AdobeFlashPlayer.dmg
d28d75c9f61d20aa990e80e88ed8f3deb37b7f7f
AdobeFlashPlayerInstaller.dmg
52873957878e37d412cd5dabddfb770bcbdf5783
MediaPlayer.dmg
b801963a180d253741be08dfbb7a5ed27964ac14  Player.pkg
```

…they appear to be, the rather infamous `OSX.Shlayer` malware.

Running the (notarized) payloads in an instrumented virtual machine captures (via our open-source **ProcessMonitor**), the execution of various shell commands via `bash`:

```
# ProcessMonitor.app/Contents/MacOS/ProcessMonitor -
pretty

{
  "event" : "ES_EVENT_TYPE_NOTIFY_EXEC",
  "process" : {
    "signing info (computed)" : {
      "signatureID" : "com.apple.bash",
       ...
      ]
    },
    "uid" : 501,
    "arguments" : [
      "sh",
      "-c",
      "tail -c +1381
\"/Volumes/Install/Installer.app/Contents/Resources/mai
n.png\" | openssl enc -aes-256-cbc -salt -md md5 -d -A
-base64 -out /tmp/ZQEifWNV2l -pass
\"pass:0.6effariGgninthgiL0.6\" && chmod 777
/tmp/ZQEifWNV2l && /tmp/ZQEifWNV2l
\"/Volumes/Install/Installer.app/Contents/MacOS/pine\"
&& rm -rf /tmp/ZQEifWNV2l"
    ],
    "ppid" : 1447,
    "pid" : 1546
  },
  "timestamp" : "1399-06-04 08:18:33 +0000"
}
```

Let’s break down these commands:

- `tail -c +1381 \"/Volumes/Install/Installer.app/Contents/Resources/main.png\"`
  Extracts bytes from `main.png` starting at offset `1381`
- `openssl enc -aes-256-cbc -salt -md md5 -d -A -base64 -out /tmp/ZQEifWNV2l -pass \"pass:0.6effariGgninthgiL0.6\`
  Decodes the output from the `tail` command into a file: `/tmp/ZQEifWNV2l`
- `chmod 777 /tmp/ZQEifWNV2l`
  Changes the file mode, to (amongst other things) fully accessible and executable.

- `/tmp/ZQEifWNV2l \"/Volumes/Install/Installer.app/Contents/MacOS/pine\"`
  Executes the file `ZQEifWNV2l`, passing in `Installer.app/...` as a command line argument.
- `rm -rf /tmp/ZQEifWNV2l`
  Deletes the `ZQEifWNV2l` file.

The use of `openssl` in this manner is a clear indicator of `OSX.Shlayer` (as is the use of fake Flash installers, and other IoCs).

```
Older variants of OSX.Shlayer used a slightly different
syntax:

openssl enc -base64 -d -aes-256-cbc -nosalt -pass
pass:2833846567 <"$fileDir”/Resources/enc
```

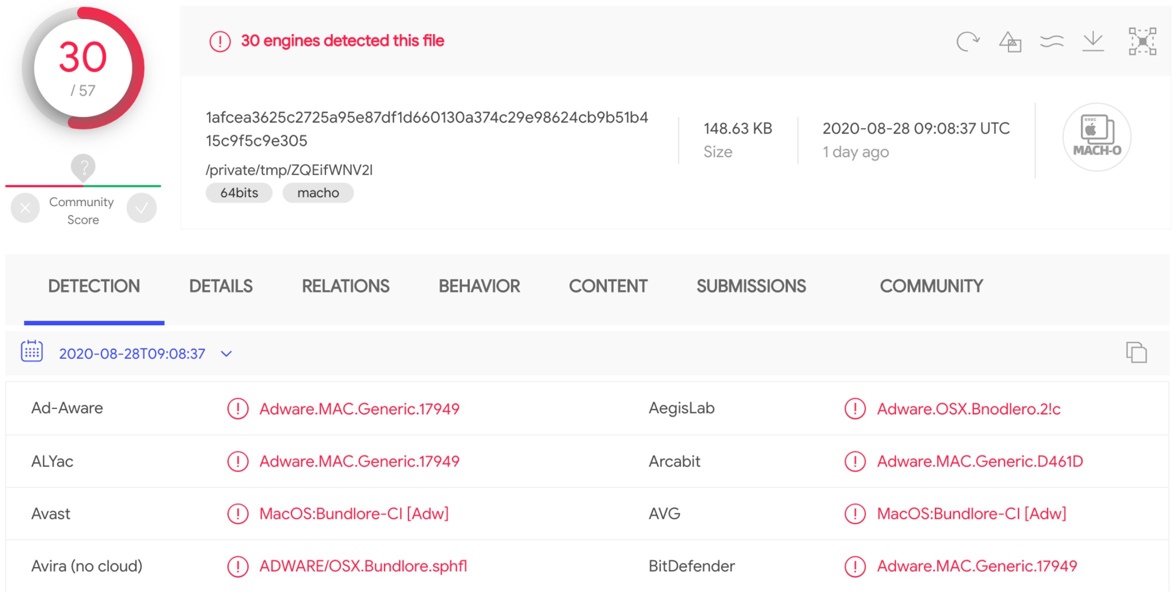
`OSX.Shlayer` is massively common, with **Kaspersky noting** it may be the most prevalent malware infecting macOS systems:

"As for the malware threats, the Shlayer family, which masquerades as Adobe Flash Player or an update for it, has been the most prevalent." -Kaspersky

And what is the ultimate goal of `OSX.Shlayer`? As noted in my **previous analysis** of this malware:

"The goal of the malware [OSX.Shlayer] is to download and persistently install various macOS adware." -The Mac Malware of 2018 (OSX.Shlayer)

Recall this variant of `OSX.Shlayer` decoded and executed a binary named `ZQEifWNV2l`. Uploading and scanning the `ZQEifWNV2l` file on VirusTotal, confirms that it is indeed adware:



30 / 57

Community Score

30 engines detected this file

1afcea3625c2725a95e87df1d660130a374c29e98624cb9b51b4
15c9f5c9e305
/private/tmp/ZQEifWNV2l
64bits macho

148.63 KB
Size

2020-08-28 09:08:37 UTC
1 day ago

MACH-O

DETECTION DETAILS RELATIONS BEHAVIOR CONTENT SUBMISSIONS COMMUNITY

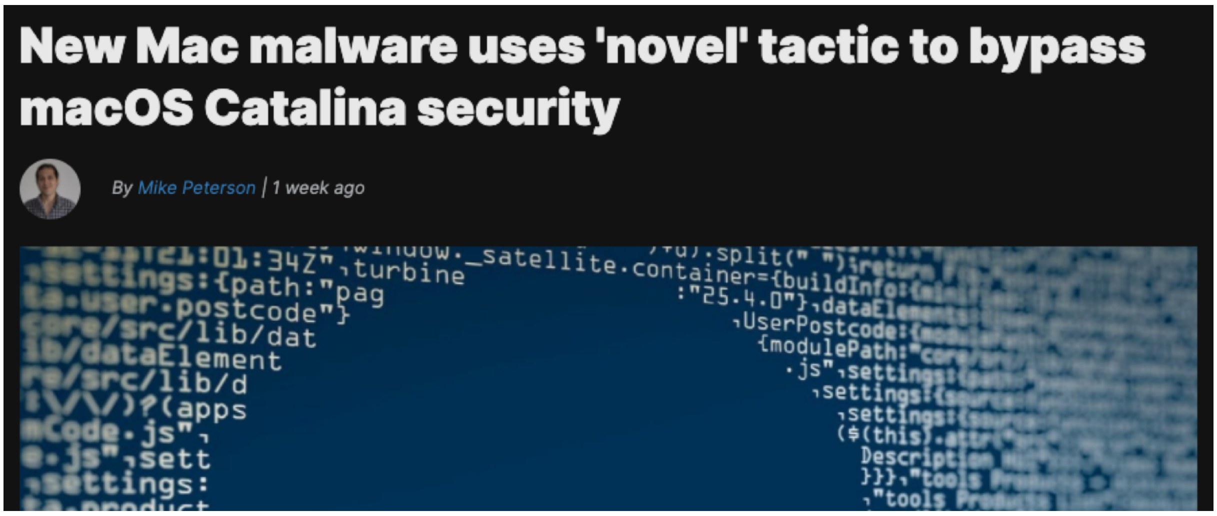
2020-08-28T09:08:37

Ad-Aware	Adware.MAC.Generic.17949	AegisLab	Adware.OSX.Bnodlero.2!c
ALYac	Adware.MAC.Generic.17949	Arcabit	Adware.MAC.Generic.D461D
Avast	MacOS:Bundlore-CI [Adw]	AVG	MacOS:Bundlore-CI [Adw]
Avira (no cloud)	ADWARE/OSX.Bundlore.sphfl	BitDefender	Adware.MAC.Generic.17949

OSX.Shlayer's payload? Bundlore

…specifically (a variant?) of the persistent `Bundlore` adware.

`OSX.Shlayer` has been known to be **quite innovative** (i.e. with manual methods of bypassing recent macOS security mechanism):



New Mac malware uses 'novel' tactic to bypass macOS Catalina security

By Mike Peterson | 1 week ago

As such, it not too surprising that this insidious malware has continued to evolve to trivially side-step Apple's best efforts.

`For more on OSX.Shlayer, see:`

- **"A Poisoned Apple: The Analysis of macOS Malware Shlayer"**
- **"VMware Carbon Black TAU Threat Analysis: Shlayer (macOS)"**

Conclusion

In Apple's own words, notarization was supposed to "*give users more confidence that [software] …has been checked by Apple for malicious components.*"

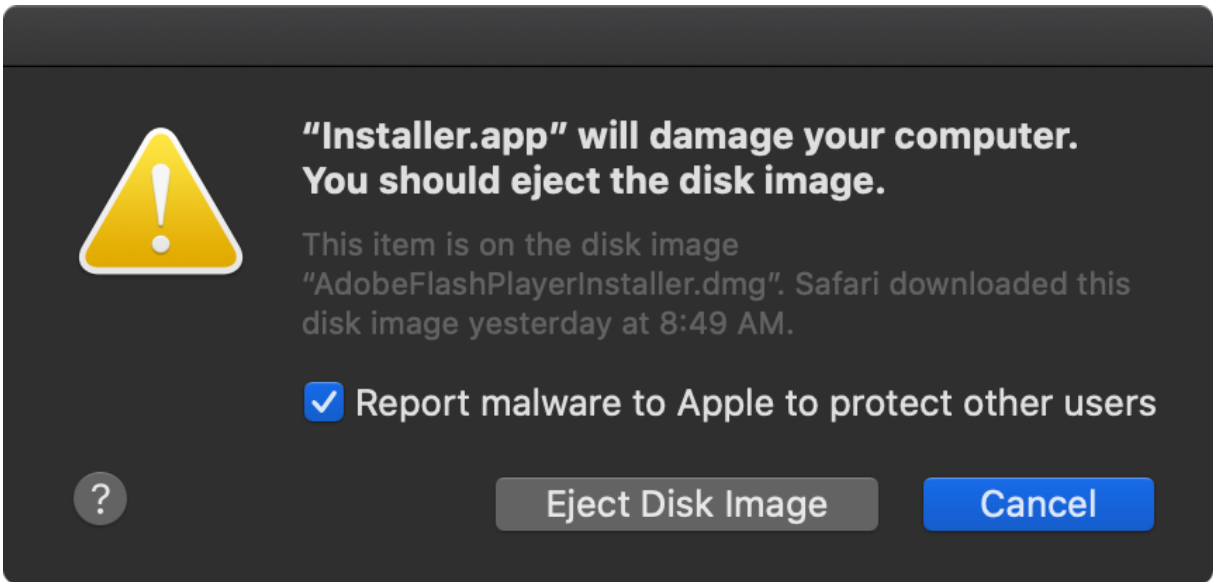
Unfortunately a system that promises trust, yet fails to deliver, may ultimately put users at more risk. How so? If Mac users buy into Apple's claims, they are likely to fully trust any and all notarized software. This is extremely problematic as known malicious software (such as `OSX.Shlayer`) is already (trivially?) gaining such notarization!

To Apple's credit, once I reported the notarized payloads, they were quick to revoked their certificates (and thus rescind their notarization status):

```
$ spctl -a -vvv -t install
/Volumes/Install/Installer.app

/Volumes/Install/Installer.app: notarization indicates
this code has been revoked
```

Thus, these malicious payloads will now, no longer run on macOS. Hooray!



"Installer.app" will damage your computer. You should eject the disk image.

This item is on the disk image "AdobeFlashPlayerInstaller.dmg". Safari downloaded this disk image yesterday at 8:49 AM.

[✓] Report malware to Apple to protect other users

(previously notarized) payloads, now blocked!

…still, the fact that known malware got notarized in the first place, raises many questions 🤔

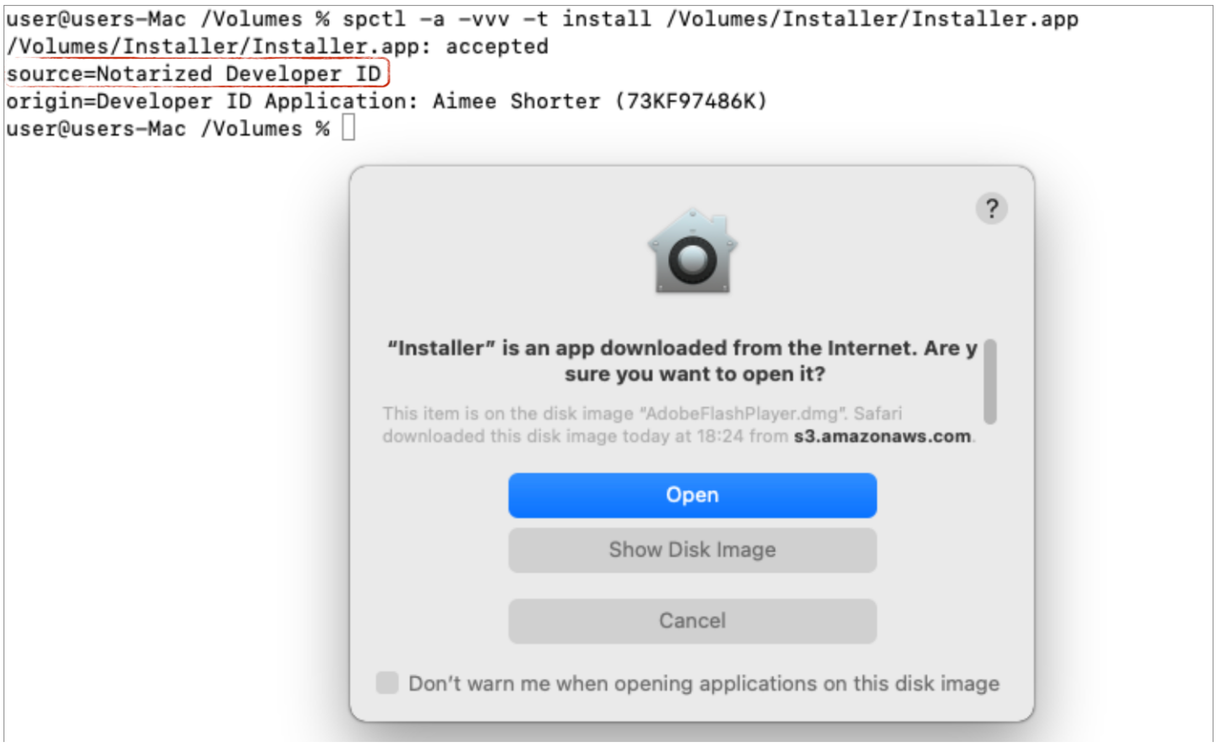
Update

As noted, Apple (quickly-ish) revoked the Developer code-signing certificate(s) that were used to sign the malicious payloads. This occurred on Friday, Aug. 28th.

Interestingly, as of Sunday (Aug 30th) the adware campaign was still live and serving up new payloads. Unfortunately these new payloads are (still) notarized:

```
$ spctl -a -vvv -t install
/Volumes/Installer/Installer.app
/Volumes/Installer/Installer.app: accepted
source=Notarized Developer ID
origin=Developer ID Application: Aimee Shorter
(73KF97486K)
```

Which means even on Big Sur, they will (still) be allowed to run:



```
user@users-Mac /Volumes % spctl -a -vvv -t install /Volumes/Installer/Installer.app
/Volumes/Installer/Installer.app: accepted
source=Notarized Developer ID
origin=Developer ID Application: Aimee Shorter (73KF97486K)
user@users-Mac /Volumes % 
```

"Installer" is an app downloaded from the Internet. Are y sure you want to open it?

This item is on the disk image "AdobeFlashPlayer.dmg". Safari downloaded this disk image today at 18:24 from **s3.amazonaws.com**.

Big Sur, prompts, but allows

If we extract the code-signing time stamp, we can see this (new) payload was signed on Friday PM (`Aug 28, 2020 at 1:04:04 PM HST`) …likely after Apple's initial "response"?

```
$ codesign -dvvv /Volumes/Installer/Installer.app
Executable=/Volumes/Installer/Installer.app/Contents/Ma
cOS/Ethernet
Identifier=com.Ethernet.bundle.installer
Format=app bundle with Mach-O thin (x86_64)
...
Authority=Developer ID Application: Aimee Shorter
(73KF97486K)
Authority=Developer ID Certification Authority
Authority=Apple Root CA
```

```
Timestamp=Aug 28, 2020 at 1:04:04 PM
```

Both the old and “new” payload(s) appears to be nearly identical, containing `OSX.Shlayer` packaged with the `Bundlore` adware.

However the attackers’ ability to agilely continue their attack (with other notarized payloads) is noteworthy. Clearly in the never ending cat & mouse game between the attackers and Apple, the attackers are currently (still) winning. 😢