SentinelLABS

SECURITY & INTELLIGENCE

How AdLoad macOS Malware Continues to Adapt & Evade

PHIL STOKES / OCTOBER 28, 2019

Incidents of the aggressive AdLoad macOS malware have been increasing over the last few months as the malware continues to evade built-in macOS security and many third-party security solutions. AdLoad is certainly not new; it's been around in one form or another since at least late 2017, but the developers have continued to adapt not only to avoid detection on installation but also to resist attempts to remove their malware. In this post, we take a deeper look into how AdLoad adapts and evades many macOS AV solutions as well as discuss how to properly detect and remove AdLoad malware.

What is AdLoad?

AdLoad is an aggressive adware infection that installs a Man-in-The-Middle web proxy to redirect user's web traffic through the attacker's own preferred servers. The aim is to hijack and redirect user's web browsers for monetary gain.

What does AdLoad do?

AdLoad is a malware that installs under a variety of different names: Kreberisec, Apollo, Aphrodite SearchDaemon and many others. The names are not entirely random. Most follow a pattern along the lines of

```
<name>SearchDaemon
<name>Lookup
<name>DataSearch
<name>Results
```

or some combination thereof.

Some of the more recent names used include ‘ElementarySignalSearchDaemon’, ‘ArtemisSearchDaemon’, ‘GlobalQuestSearchDaemon’, ‘TrustedMacResultsSearchDaemon’, ‘NetSignalSearchDaemon’, ‘SimpleSearchAppDaemon’, ‘SearchQuestDaemon’ and – breaking the mould a little – ‘SearchQuest’ and ‘ResultSync’ among many others.

Here’s a partial list of some of the most common names circulating at the moment.

```
AphroditeLookup
AphroditeResults
ApolloSearch
ApolloSearchDaemon
ArtemisSearch
ArtemisSearchDaemon
ElementaryDataSearch
ElementarySignalSearchDaemon
FindData
GlobalConsoleSearch
GlobalQuestSearch
GlobalQuestSearchDaemon
Kreberisec
NetSignalSearchDaemon
ResultSync
SearchAdditionally
SearchAdditionallyDaemon
SearchQuest
SearchQuestDaemon
SimpleFunctionSearch
SimpleSearchAppDaemon
TrustedMacResultsSearchDaemon
WebSearchStride
```

This malware has been known for some time, with at least two variants known to Apple’s XProtect definitions from November 2017 or earlier. Despite that, versions of AdLoad are still being reported in the wild by macOS users on Apple Support Communities forums.

image of asc complaints

Unfortunately for many macOS users, neither XProtect nor many other simple static engines detect it.

image of adload virustotal detection

This rule from Apple’s current XProtect definitions effectively requires the scanned binary to contain the string “getSafariVersion” in order to trigger a detection.

image of adload xprotect detection

image of hex to string translation

Alas, malware authors have long since refactored their code and current variants no longer contain that string. That breaks XProtect’s ability to detect the malware with the above rule, since the rule specifies that string as *necessary*, though not sufficient, for a detection.

In order to avoid simple static detections, the files have different hashes, though they are often of similar size.

image of AdLoad Hashes

Hitting on distinctive method names shows the files to be variants of the same malware:

AdLoad hash image

Interestingly, XProtect's "Mughthesec" definition gets closer to current AdLoad static signatures.

image of mughthesec

image of mughthesec yara rule

This definition requires, among other things, that the binary contains strings including the substring `fallback` as well as `BerTaggedData`. As we'll see later, that suggests a close link between AdLoad and Mughthesec malware, but it still fails to catch the AdLoad malware which, while it does use `BerTaggedData`, it does not contain the substrings with `fallback`.

And clearly while static signature detections on `BerTaggedObject` and other static features of the binary might be good while they last, like Apple's XProtect signatures they won't last long. Malware authors will soon refactor once their success-to-detection rate starts to tumble in the wrong direction. We'll see how to more effectively detect and protect against this kind of malware below.

AdLoad Dropped Files

Victims of AdLoad will find that the malware drops files in both some of the usual, easy to find places used by macOS malware as well as some much lesser known areas that can be hard to detect. The adware typically presents an authorization dialog that asks for an admin password. On collecting the password, the malware uses the credentials to drop a number of files in both the user and local computer domains. First, it will drop not only a LaunchAgent in the local user Library but also two LaunchDaemons in the local domain Library.

The following uses `SearchQuest` as the example name, but of course this may be replaced with any one of the names and patterns mentioned above.

First, the LaunchAgent is dropped at `/Users/aUser/Library/LaunchAgents/com.SearchQuest.plist`

and targets the following executable in its Program Arguments: `/Users/aUser/Library/Application Support/com.SearchQuest/SearchQuest`

Next, the first of two LaunchDaemons is dropped at `/Library/LaunchDaemons/com.SearchQuestDaemon.plist`

which targets a corresponding item in: `/Library/Application Support/com.SearchQuestDaemon/SearchQuest`

The second LaunchDaemon, in this example, lands here: `/Library/LaunchDaemons/com.SearchQuestP.plist`

and targets a Mach-O executable in a hidden folder in `/var/root`: `/var/root/.SearchQuest/SearchQuestDaemon`

This item itself calls a python script in the same folder: `/var/root/.SearchQuest/SearchQuest --mode socks5 --showhost -q -s /var/root/.SearchQuest/SearchQuest.py`

image of running processes

This is used to run the python script `SearchQuest.py` in "quiet" mode in order to open a connection with a remote host.

The SOCKS Proxy is enabled for `localhost:8080` in Network Settings and is persistent across restarts.

SearchQuest also drops a hidden directory of mitmpoxy certs in:

`/var/root/.mitmproxy`.

image of mitmproxy

AdLoad, a Malware That Doesn't Give Up

AdLoad doesn't stop with a LaunchAgent and two LaunchDaemons in its attempt to maintain persistence. It also installs a user cronjob and an executable in a subfolder of the user's Library Application Support folder. The sub folder has a UUID-like hex pattern of **8-4-4-4-12** characters, and the executable inside it has a name with a different UUID-like hex with the same **8-4-4-4-12** pattern.

This is just a random example, the UUID numbers are unique to each user and are likely used as part of campaign tracking.

```
30 */2 * * * /Users/aUser/Library/Application Support/712B5686-92B3-919D-DD36-13A5745D87D2/74C08AE4-8ACE-51CA-54F7-0ED0A530ECAA h >/dev/null 2>&1}
```

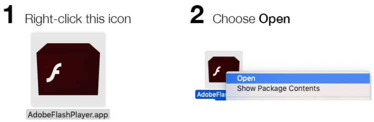
This code runs every 2hrs and 30 minutes. The target of the cronjob is a Mach-O executable that imports Apple's JavaScriptCore framework, which allows the binary to evaluate JavaScript scripts and likewise make available native objects, methods and functions to the JavaScript environment.

Optional Extras: AdLoad & Friends

In every infection I've seen of this malware to date, there's always a number of others that appear to be brought along to the party. Tests in both VMs and 'bare metal' lab machines have proved inconclusive to date as to whether these additional installations are payloads directly dropped by AdLoad itself or are subsequent infections. We have seen a number of cases where a Pay-per-Install PUP installer delivers adware, which in turn delivers other PPI installers and further adware. The ties between the players in this game of monetizing unwanted downloads and browser hijacks seem to be becoming increasingly closer.

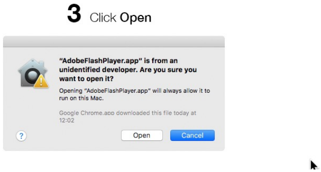
Adding to the variance in what gets installed is the way campaigns tailor payloads depending on local settings, which are usually harvested at install time. These include the geolocation of the victim, device make and model, operating system version and language settings.

Nevertheless, AdLoad is often frequently found alongside other adware/PUP installations variously known as "Mughthesec", "SurfBuyer", "Souter", "MyShopcoupon", "MMInstall" and "MMProt". All use a similar infection mechanism, which begins with a redirect to a scam website that invites the user to download a "needed" Flash Player or other media player component. These are typically unsigned and the user is provided instructions on how to open the application by bypassing Apple's built-in Gatekeeper and XProtect safeguards.



1 Right-click this icon

2 Choose **Open**



"AdobeFlashPlayer.app" is from an unidentified developer. Are you sure you want to open it?

Opening "AdobeFlashPlayer.app" will always allow it to run on this Mac.

Google Chrome.app downloaded this file today at 12:02

Other IOCs to watch out for that we've seen on some occasions with this threat, albeit not consistently for reasons mentioned above, are

1. modified sudoers file
2. installation of the attackers SSH known_hosts key in ~/.ssh
3. A configuration Profile to lock down browser settings (in System Preferences > Profiles)
4. Browser extensions such as "AnySearch", "SearchIt" and similar names
5. Creation of folder `/User/aUser/Applications` and malicious files dropped therein

How to Remove AdLoad? Detecting and Protecting Against AdLoad

SentinelOne customers are protected against AdLoad, which is detected as a threat and blocked when it tries to execute – regardless of how the malware is coded internally – as the AI behavioral engine detects the malicious behaviour rather than relies on static signatures.

For those who are not yet protected by SentinelOne, manual removal may be something of a race against time. With multiple persistence agents – launch agents, daemons, cronjobs and processes running in memory out of `/var/root` – it can sometimes take several attempts to beat all of these before one of them manages to re-write the deleted components back to disk. Somewhat like the malware itself, however, persistence does pay off. As long as you have identified all the malicious processes and persistence agents, repeatedly removing them will eventually beat the race.

Conclusion

We have seen an increasing number of reports of attempted AdLoad infections and increasing concern among macOS users. AdLoad is extremely difficult to remove for individual users, and even some 3rd party solutions that detect it after-the-fact are unable to outrun the malware's multiple persistence mechanisms. Hopefully, this post has provided some insight in how to detect and remove the AdLoad malware. If you'd like to see how SentinelOne can autonomously protect your Mac users from this kind of threat, please contact us for a free demo.

Like this article? Follow us on LinkedIn, Twitter, YouTube or Facebook to see the content we post.

Read more about Cyber Security

- How TrickBot Hooking Engine Targets Windows 10 Browsers
- 7 Lessons Every CISO Can Learn From the ANU Cyber Attack
- Behind Enemy Lines | Looking into Ransomware as a Service (Project Root)

- APT and the Enterprise – FUD or Real Threat?
- Cyber Insurance Is No Substitute For Robust Cybersecurity Systems
- The Good, The Bad and The Ugly in Cybersecurity

ADWARE APPLE MITM

PHIL STOKES

Phil Stokes is a Threat Researcher at SentinelOne, specializing in macOS threat intelligence, platform vulnerabilities and malware analysis. He began his journey into macOS security as a software developer, creating end user troubleshooting and security tools just at the time when macOS adware and commodity malware first began appearing on the platform. Phil has spent the last 7 years closely following the development of macOS threats as well as researching software and OS vulnerabilities.

©2023 SentinelOne, All Rights Reserved.