

Objective-See

objective-see.com/blog/blog_0x20.html

WTF is Mughthesecl!?

› poking on a piece of undetected adware

8/08/2017

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



Want to play along? I've shared the adware, which can be downloaded [here](#) (password: infect3d).

Background

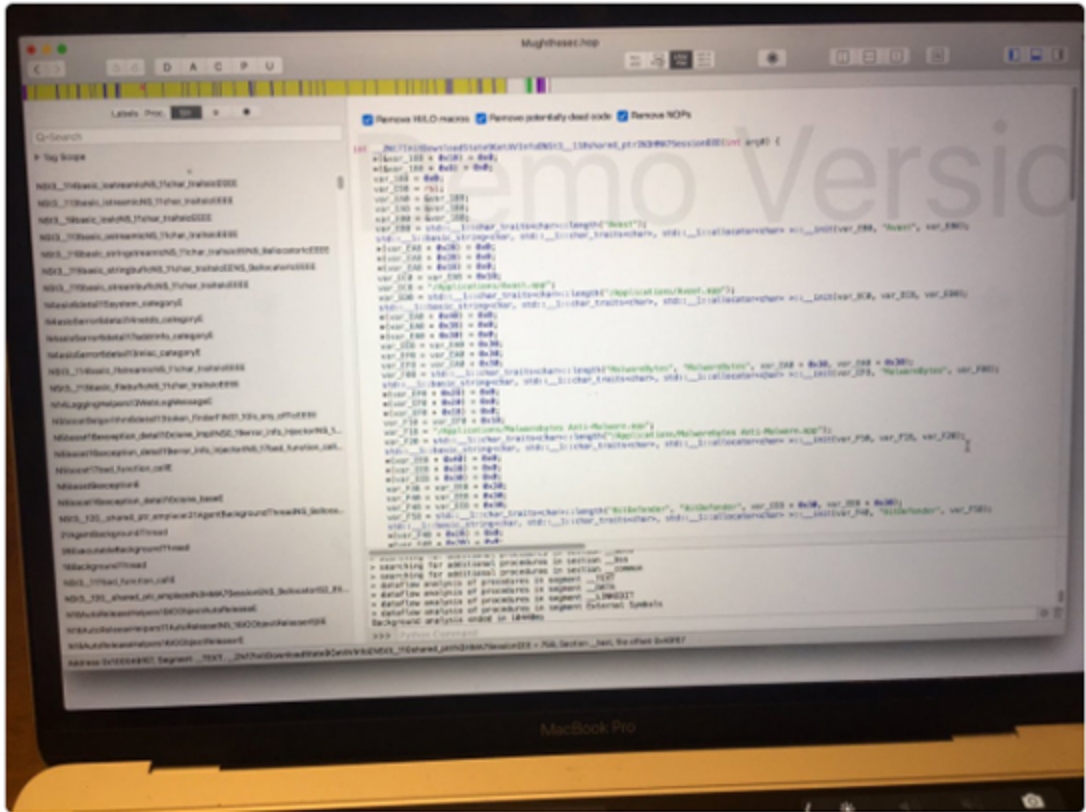
Yesterday Gavriel State ([@gavrielstate](#)) posted an interesting [tweet](#):



Gavriel State
@gavrielstate

Following

@thomasareed - you ever hear about Mac Malware called Mughthesecc? My kid's computer has it, and it seems to have AV detection code in it



Interestingly, googling "Mughthesecc" only returned one relevant hit; a post on Apple's online's forums titled "Safari does not render Gmail correctly". Posted on August 2nd, user 'given' stated that, "Only in Safari, when this specific user logs in, it does not render Gmail correctly. Only Gmail. Only in Safari." Following another user's suggestion, 'given' ran EtreCheck which noted several "unknown files:"

- ~/Library/LaunchAgents/com.Mughthesecc.plist
- ~/Library/Application Support/com.Mughthesecc/Mughthesecc

Gavriel was kind enough to share a sample ('Mughthesecc') with me, and that, coupled with the assistance from another security researcher, led to recovery of what appeared to be the

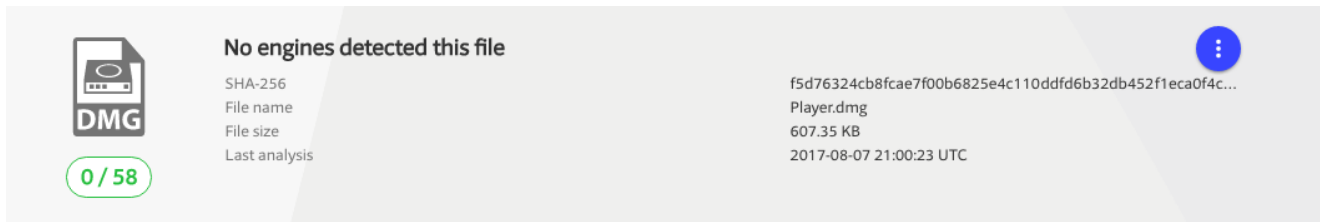
original installer (sha256:

f5d76324cb8fcae7f00b6825e4c110ddfd6b32db452f1eca0f4cff958316869c)

As neither the sample, Mughthesecc, nor the (signed!) installer were detected by any AV engines on Virus Total I decided to take a closer look.


Analysis

Let's start with the installer disk image. Uploaded to VirusTotal on August 4th as Player.dmg, it currently remains undetected:



The screenshot shows the VirusTotal interface for a file named 'Player.dmg'. The status is 'No engines detected this file'. The file size is 607.35 KB and it was last analyzed on 2017-08-07 21:00:23 UTC. A green badge indicates '0/58' engines. The SHA-256 hash is f5d76324cb8fcae7f00b6825e4c110ddfd6b32db452f1eca0f4c...

Using [WhatsYourSign](#), we can examine the signing info:



The screenshot shows the output of the WhatsYourSign tool. It displays a lock icon and the message 'player is validly signed (Apple Dev-ID)'. Below this, it shows the file 'player.dmg' with its path. The item type is 'Disk Image'. The signing authority is listed as 'Developer ID Application: Quoc Thinh (9G2J3967H9)', 'Developer ID Certification Authority', and 'Apple Root CA'. A 'close' button is visible in the bottom right corner.

Using `spctl`, we can confirm the disk image's certificate is still valid (i.e. not rejected):

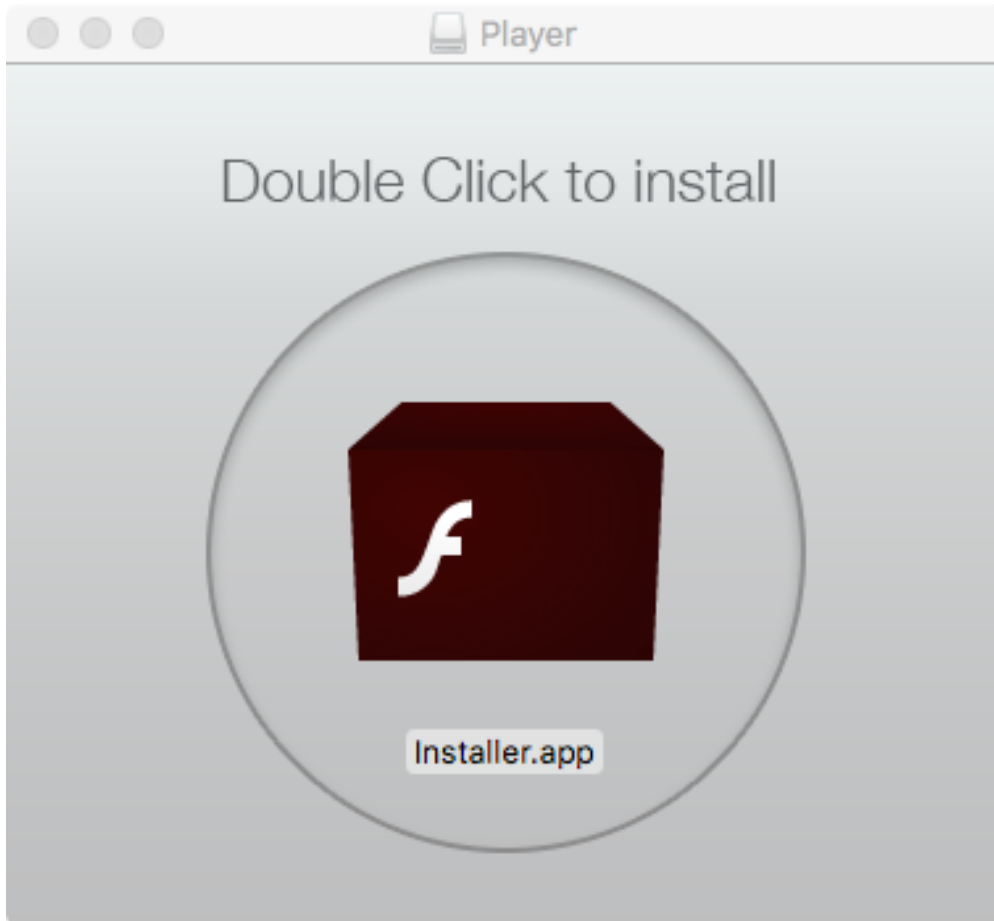
```
$ spctl -a -t install -vv ~/Downloads/Mughthesecc/Player.dmg
```

```
~/Downloads/Mughthesecc/player.dmg: accepted
```

```
source=Developer ID
```

```
origin=Developer ID Application: Quoc Thinh (9G2J3967H9)
```

Double-clicking the disk image, Player.dmg mounts it, revealing a single file Installer.app:



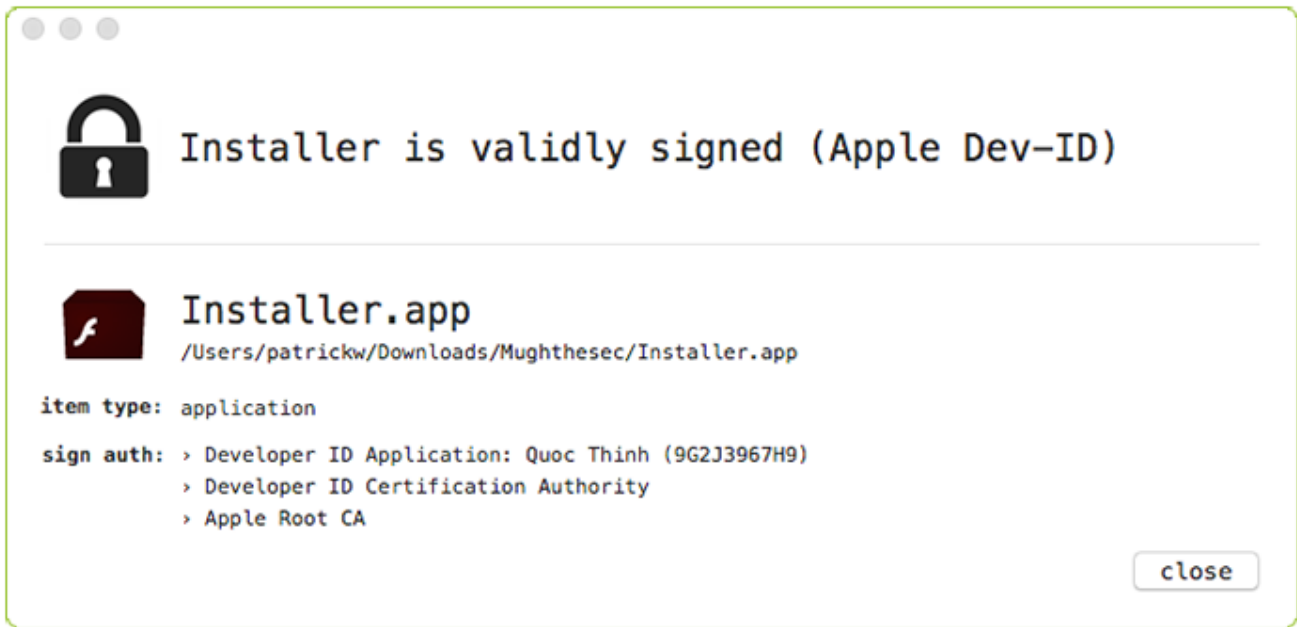
Besides its icon and name, the Installer.app's Info.plist file, shows it masquerading as Flash installer:

```
cat Installer.app/Contents/Info.plist
```

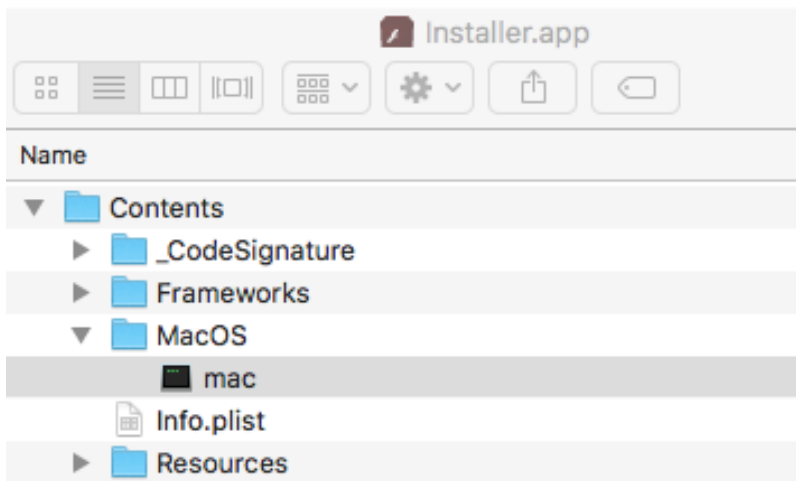
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
...
  <key>CFBundleIdentifier</key>
  <string>com.FlashPlayer</string>

  <key>CFBundleName</key>
  <string>FlashPlayerInstaller</string>
```

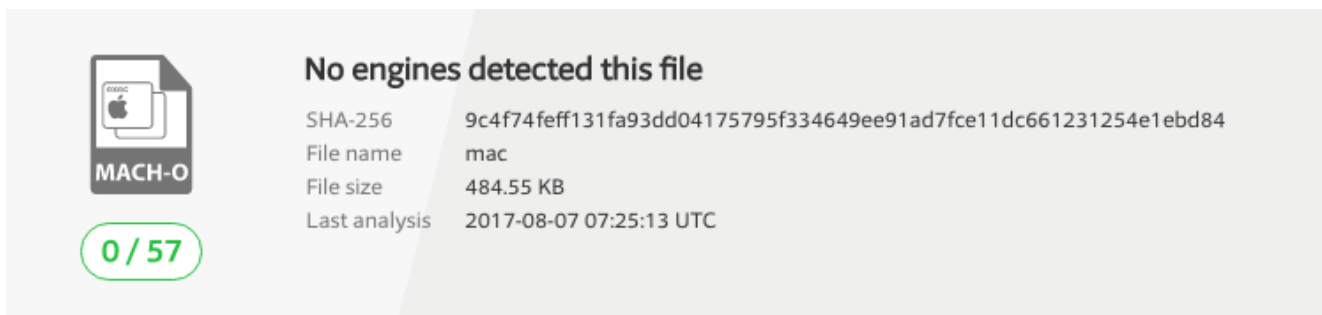
This application, is also signed with the same Apple Developer ID:



Examining its application bundle, we can see its executable is a binary name 'mac' ... how creative!



This binary is also (currently) undetected by any AV engine on Virus Total:



Taking a quick peak at the installer binary shows what appears to be anti-anti-virus logic:

Address	Length	Type	String
__cstring...	00000006	C	Avast
__cstring...	00000018	C	/Applications/Avast.app
__cstring...	0000000D	C	MalwareBytes
__cstring...	0000002C	C	/Applications/Malwarebytes Anti-Malware.app
__cstring...	0000000C	C	BitDefender
__cstring...	0000002D	C	/Library/Bitdefender/AVP/AntivirusforMac.app
__cstring...	0000000A	C	Kaspersky
__cstring...	0000002F	C	/Applications/Kaspersky Anti-Virus For Mac.app
__cstring...	00000007	C	Norton
__cstring...	00000022	C	/Applications/Norton Security.app
__cstring...	00000007	C	McAfee
__cstring...	00000012	C	/usr/local/McAfee
__cstring...	0000001F	C	/Applications/AVGAntiVirus.app
__cstring...	00000005	C	Eset
__cstring...	00000034	C	/Library/LaunchDaemons/com.eset.esets_daeomon.plist

We can also run strings to search for embedded URLs:

```
$ strings -a ~/Downloads/Mughthesecc/Installer.app/Contents/MacOS/mac | grep http
```

<http://api.simplyeapps.com/p>

<http://cdn.simplyeapps.com/screens/precheck/DmFybQ==>

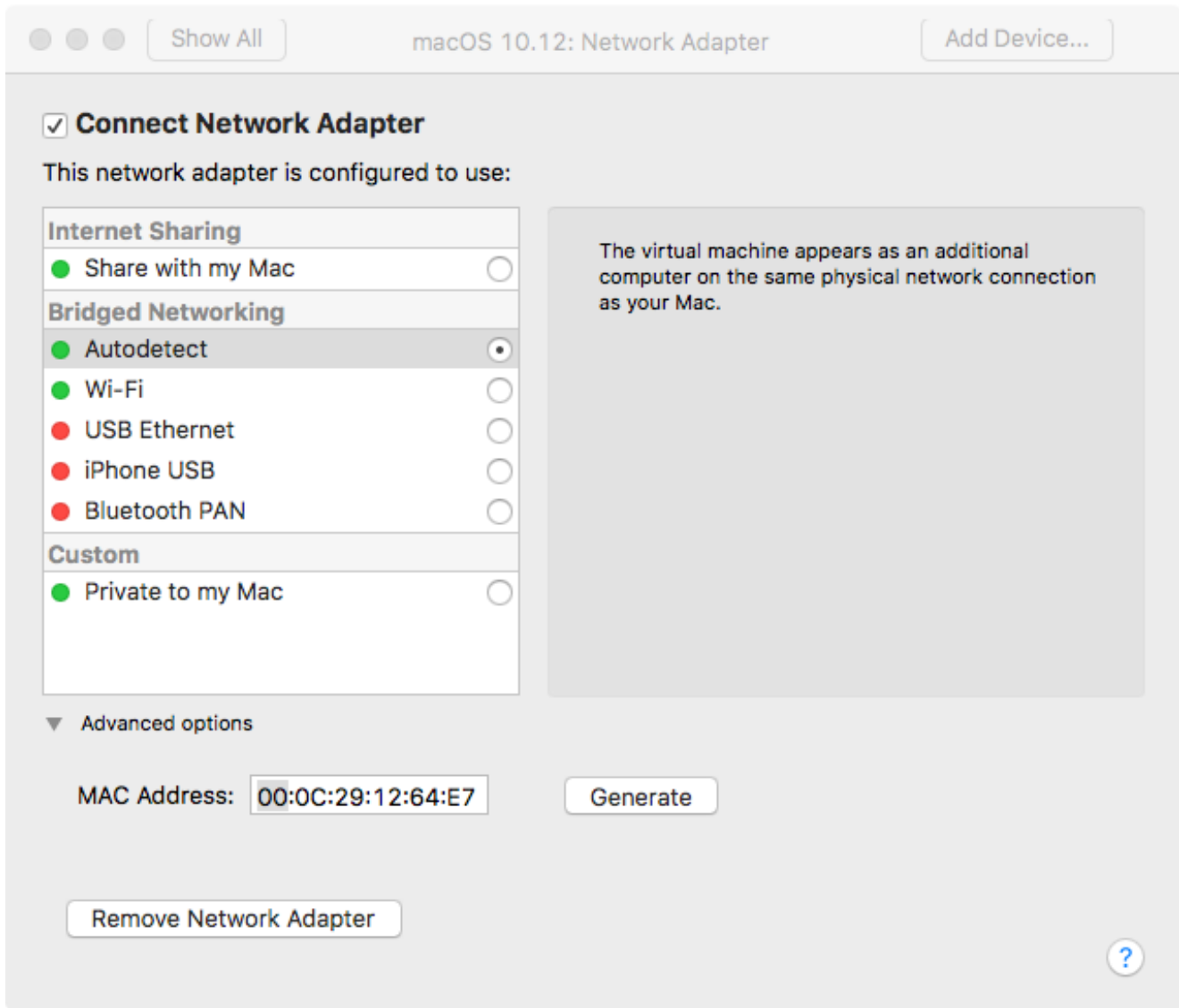
<http://cdn.simplyeapps.com/screens/progress/DmFybQ==>

<http://cdn.simplyeapps.com/screens/complete/DmFybQ==>

<http://api.simplyeapps.com/l>

Now, before we run this in a VM - let's change the MAC address of the virtual machine. This is required step, because it turns out that the installer actually doesn't do anything malicious, (besides actually installing a legit copy of Flash), if it detects it running in VM. Thomas Reed (@thomasreed) correctly guessed that this 'VM detection' is done by examining the MAC address (VMWare VMs have 'recognizable' MAC address). Apparently this is common trick used in macOS adware!

To change the VM's mac address, shut it down, then change it via the VM's Network Adapter's settings (click 'Advanced Options' to modify the MAC address).

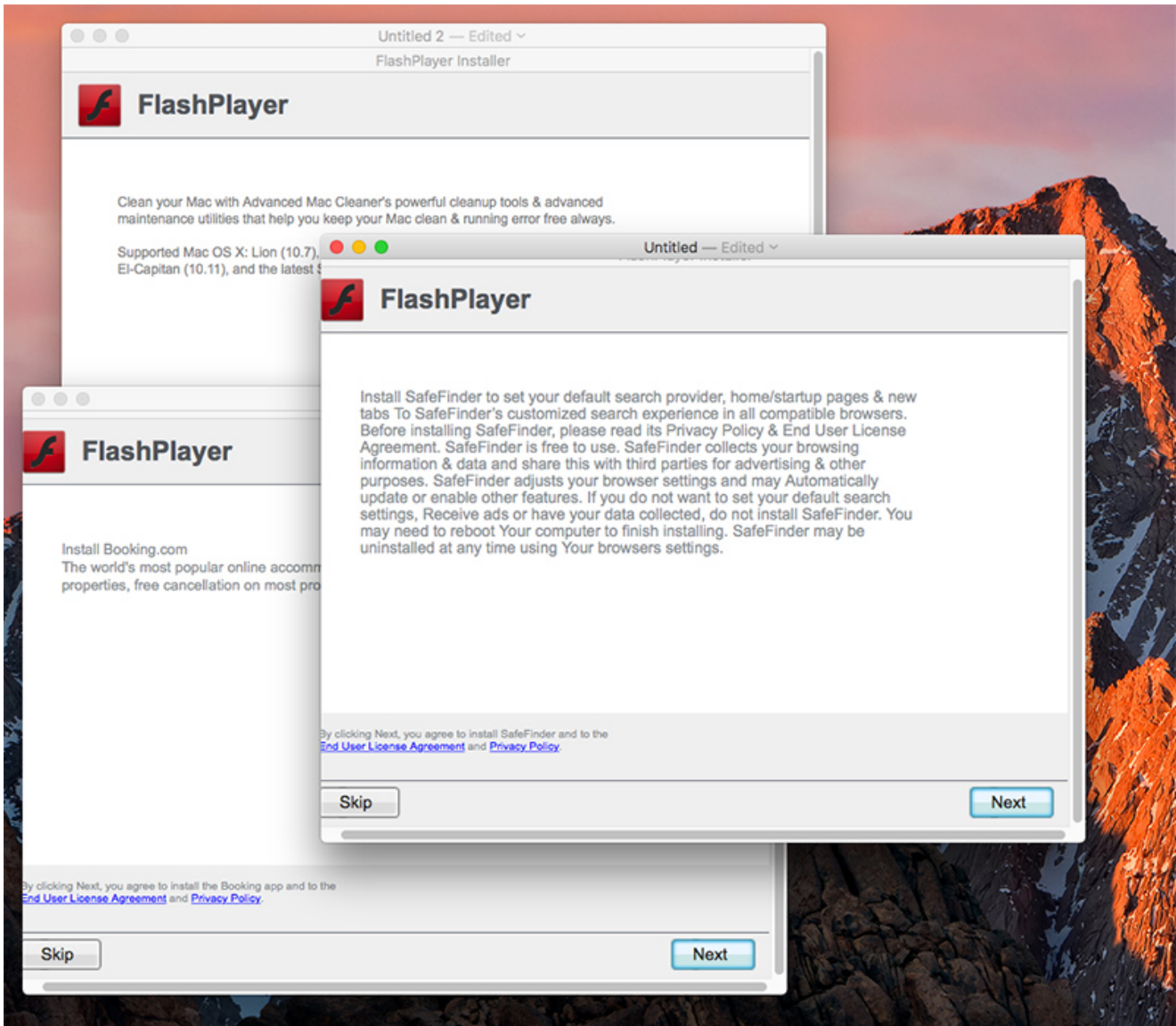


Alright, let's run the damn Installer.app already! First thing, LuLu (my soon-to-be-released macOS firewall!) detects an outgoing network connection:




Once the outgoing connection is allowed, the Installer application kindly asks the user to install some 'adware' and potentially unwanted programs:

1. Advanced Mac Cleaner
2. Safe Finder
3. Booking.com



Since we're playing along, we click 'Next' to install it all!

Not too unexpectedly, the Advanced Mac Cleaner triggers a few BlockBlock warnings as it attempts to install a persistent launch agent and login item:



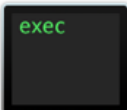
Advanced Mac Cleaner
installed a launch daemon or agent

virus total ancestry

Advanced Mac Cleaner (Developer ID Application: Techyutils Software Private Limited (VS9Q8BRRRJ))
process id: 604
process path: /Applications/Advanced Mac Cleaner.app/Contents/MacOS/Advanced Mac Cleaner

helperamc (Developer ID Application: Techyutils Software Private Limited (VS9Q8BRRRJ))
startup file: /Users/user/Library/LaunchAgents/com.pcv.hlpramcn.plist
startup binary: /Users/user/Library/Application Support/amc/helperamc.app/Contents/MacOS/helperamc

remember



cfprefsd
installed a login item

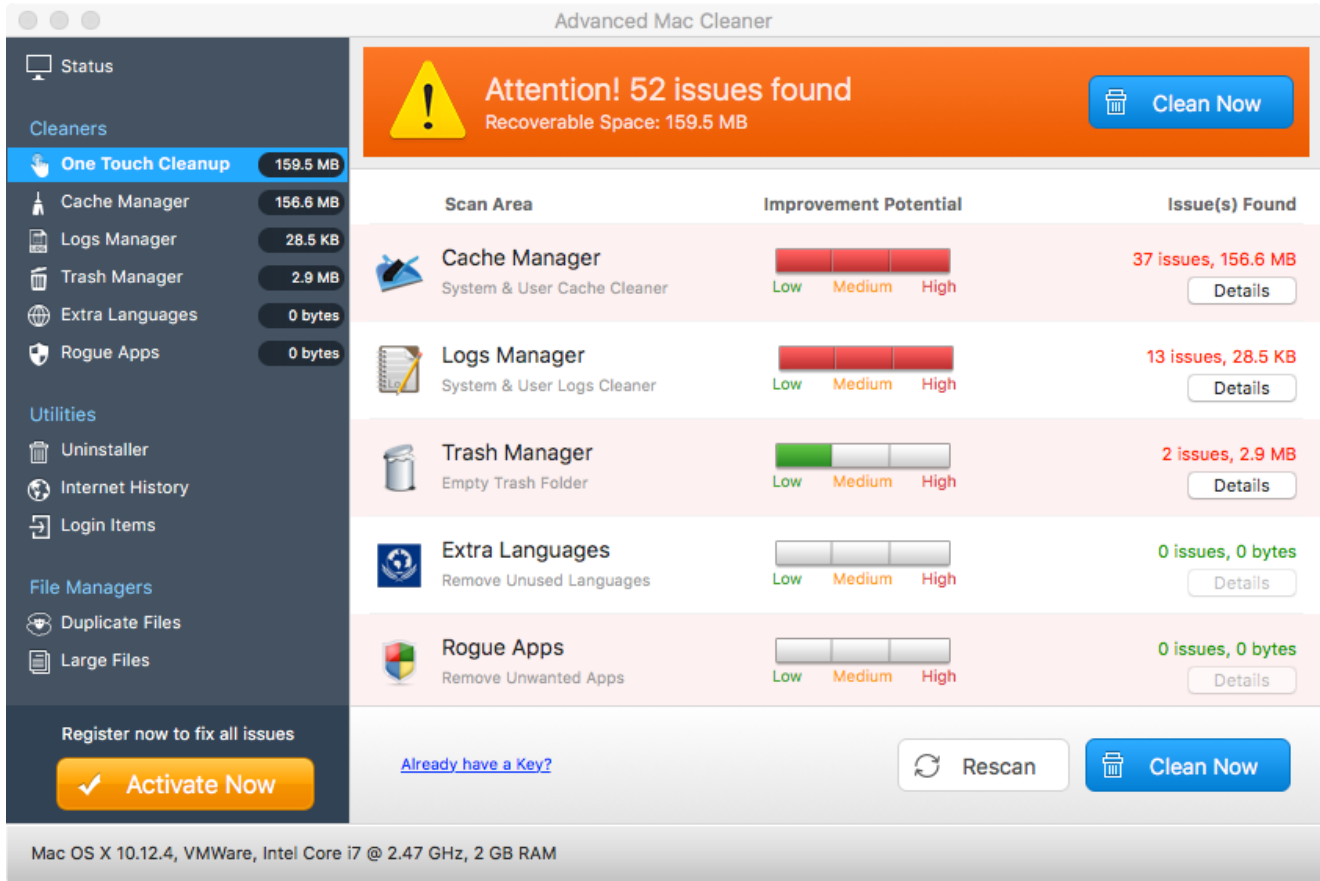
virus total ancestry

cfprefsd (Apple Code Signing Cert Auth)
process id: 123
process path: /usr/sbin/cfprefsd

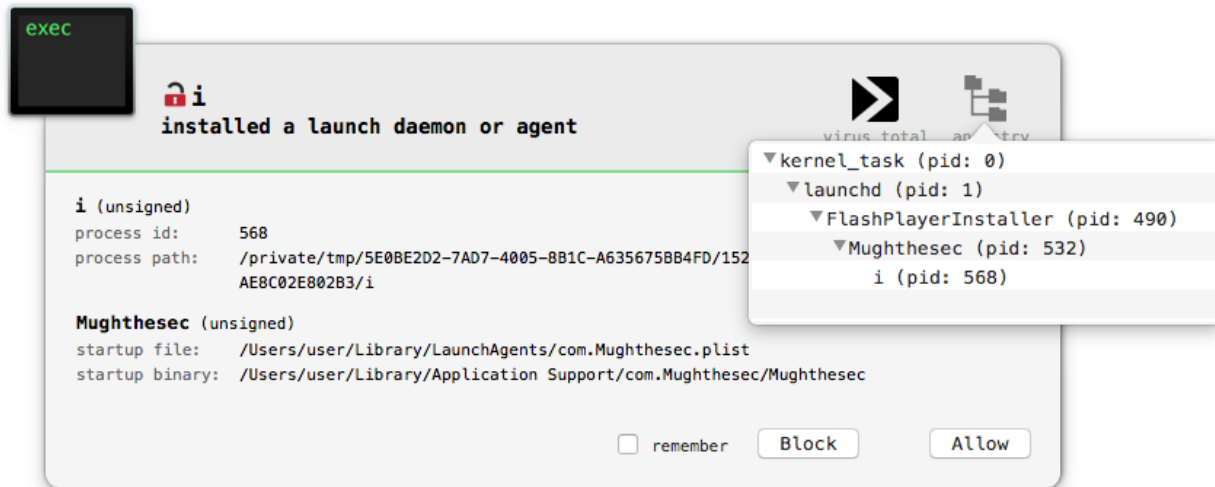
Advanced Mac Cleaner (Developer ID Application: Techyutils Software Private Limited (VS9Q8BRRRJ))
startup file: /Users/user/Library/Preferences/com.apple.loginitems.plist
startup binary: /Applications/Advanced Mac Cleaner.app

remember

It also kindly informs us of several 'critical' issues. How thoughtful :P



Moving on to 'Safe Finder', BlockBlock as alerts us of a process named 'i' persisting something named 'Mughthesec as a launch agent.



An open-source process monitoring utility I wrote (based on the Proc Info library) shows Mughthesec being started by the Installer application (FlashPlayerInstaller, pid: 490):

```
# procMonitor
```

```
process start:
pid: 532
path: /private/tmp/F3A53281-D3FA-4F32-B996-3EE0FCF522D5/62/Mughthesecc
user: 501
args: (
  "/tmp/F3A53281-D3FA-4F32-B996-3EE0FCF522D5/62/Mughthesecc",
  2,
  na,
  na,
  "F3A53281-D3FA-4F32-B996-3EE0FCF522D5"
)
ancestors: (
  490,
  1
)
binary:
name: Mughthesecc
path: /private/tmp/F3A53281-D3FA-4F32-B996-3EE0FCF522D5/62/Mughthesecc
signing info: {
  signatureStatus = "-67062";
} (isApple: 0 / isAppStore: 0)
```

The process monitor also shows this process (Mughthesecc, pid: 532), spawning executing the 'i' process out of /tmp:

```
# procMonitor
```

```
process start:
pid: 568
path: /private/tmp/5E0BE2D2-7AD7-4005-8B1C-A635675BB4FD/15261EBB-ED0B-46DA-8C3B-AE8C02E802B3/i
user: 501
args: (
  "/tmp/5E0BE2D2-7AD7-4005-8B1C-A635675BB4FD/15261EBB-ED0B-46DA-8C3B-AE8C02E802B3/i",
  "5E0BE2D2-7AD7-4005-8B1C-A635675BB4FD",
  "S+wIS+tmwyirIkk8AAF36Jlq8TSRdg...==",
  10
)
ancestors: (
  532,
```

```
490,  
1  
)  
binary:  
name: i  
path: /private/tmp/5E0BE2D2-7AD7-4005-8B1C-A635675BB4FD/15261EBB-ED0B-46DA-  
8C3B-AE8C02E802B3/i  
signing info: {  
    signatureStatus = "-67062";  
} (isApple: 0 / isAppStore: 0)
```

This 'i' process is what persists and starts 'launch agent' instance of Mughthesecc. We can see this, again, via the process monitor which shows process 'i' (pid: 568) invoking launchctl with the 'load' command line option and the path to the launch agent plist:

```
# procMonitor
```

```
process start:  
pid: 576  
path: /bin/launchctl  
user: 501  
args: (  
"/bin/launchctl",  
load,  
"/Users/user/Library/LaunchAgents/com.Mughthesecc.plist"  
)  
ancestors: (  
    568,  
    532,  
    490,  
    1  
)  
binary: name: launchctl  
path: /bin/launchctl  
signing info: {  
    signatureStatus = 0;  
    signedByApple = 1;  
    signingAuthorities = (  
        "Software Signing",  
        "Apple Code Signing Certification Authority",
```

```
"Apple Root CA"  
);  
} (isApple: 1 / isAppStore: 0)
```

Ok, so let's take a closer look at the Mughthesecc launch agent and binary. The Mughthesecc launch agent plist is located at ~/Library/LaunchAgents/com.Mughthesecc.plist:

```
$ cat ~/Library/LaunchAgents/com.Mughthesecc.plist
```

```
<?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>Label</key>  
  <string>com.Mughthesecc</string>  
  <key>ProgramArguments</key>  
  <array>  
    <string>/Users/user/Library/Application  
Support/com.Mughthesecc/Mughthesecc</string>  
    <string>r</string>  
  </array>  
  <key>RunAtLoad</key>  
  <true />  
  <key>StartInterval</key>  
  <integer>14400</integer>  
</dict>  
</plist>
```

From this plist we can see that the launch agent will:

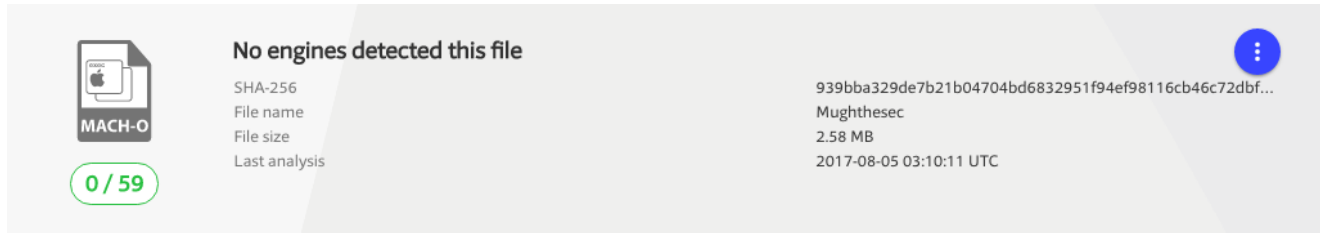
1. execute a binary: ~/Library/Application Support/com.Mughthesecc/Mughthesecc
2. pass in a parameter: 'r'
3. be automatically started whenever the user logs in, as 'RunAtLoad' is set to true

The 'Mughthesecc' binary, ~/Library/Application Support/com.Mughthesecc/Mughthesecc, is unsigned:

```
$ codesign -dvvv "~/Library/Application Support/com.Mughthesecc/Mughthesecc"
```

~/Library/Application Support/com.Mughthesecc/Mughthesecc: **code object is not signed at all**

It is also (currently) undetected by any AV engines on VirusTotal:



The screenshot shows a VirusTotal scan result for a MACH-O file. The status is "No engines detected this file". The file name is "Mughthesecc", the size is "2.58 MB", and the last analysis was on "2017-08-05 03:10:11 UTC". The SHA-256 hash is "939bba329de7b21b04704bd6832951f94ef98116cb46c72dbf...". A green badge indicates "0/59" engines have scanned the file.

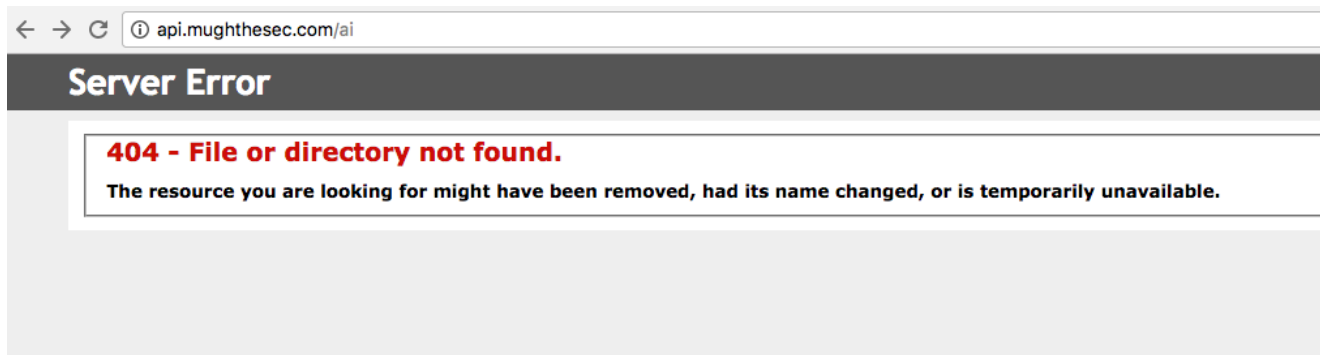
Running strings shows some embedded URLs:

```
$ strings -a ~/Library/Application Support/com.Mughthesecc/Mughthesecc | grep http
```

```
http://api.mughthesecc.com/ai
```

```
http://api.mughthesecc.com/l
```

Attempting to access those URLs in a browser, appears to result in an error:



The screenshot shows a browser window with the address bar containing "api.mughthesecc.com/ai". The page displays a "Server Error" message: "404 - File or directory not found. The resource you are looking for might have been removed, had its name changed, or is temporarily unavailable."

However, the host mughthesecc.com does appear to be online, resolving to 192.64.119.107:

```
$ nslookup mughthesecc.com
```

```
Non-authoritative answer:
```

```
Name: mughthesecc.com
```

```
Address: 192.64.119.107
```

This IP address, 192.64.119.107, appears to be rather malicious:

192.64.119.107 IP address information

Country US
Autonomous system 22612 (Namecheap, Inc.)

URLs

Date scanned	Detections	URL
2017-08-08	4/65	http://bestsafeandunbelievableupdate.bid/
2017-08-08	6/65	http://settingupdateserviceformacandpc.download/
2017-08-08	1/65	http://imagineads.mobi/
2017-08-07	5/65	http://thebiggestsoft2updating.top/
2017-08-07	5/65	http://updateworkfreeforpcandmacalike.download/
2017-08-06	4/65	http://howtoupdate156322.download/
2017-08-06	4/65	http://clickforsafesystem2upgrades.win/
2017-08-05	6/65	http://thesafestsoftmediaforu.download/
2017-08-04	4/65	http://clickforfreeandreadyupgrades.download/
2017-08-04	5/65	http://thepperfectandultimatesystemsforupgrade.pw/

More

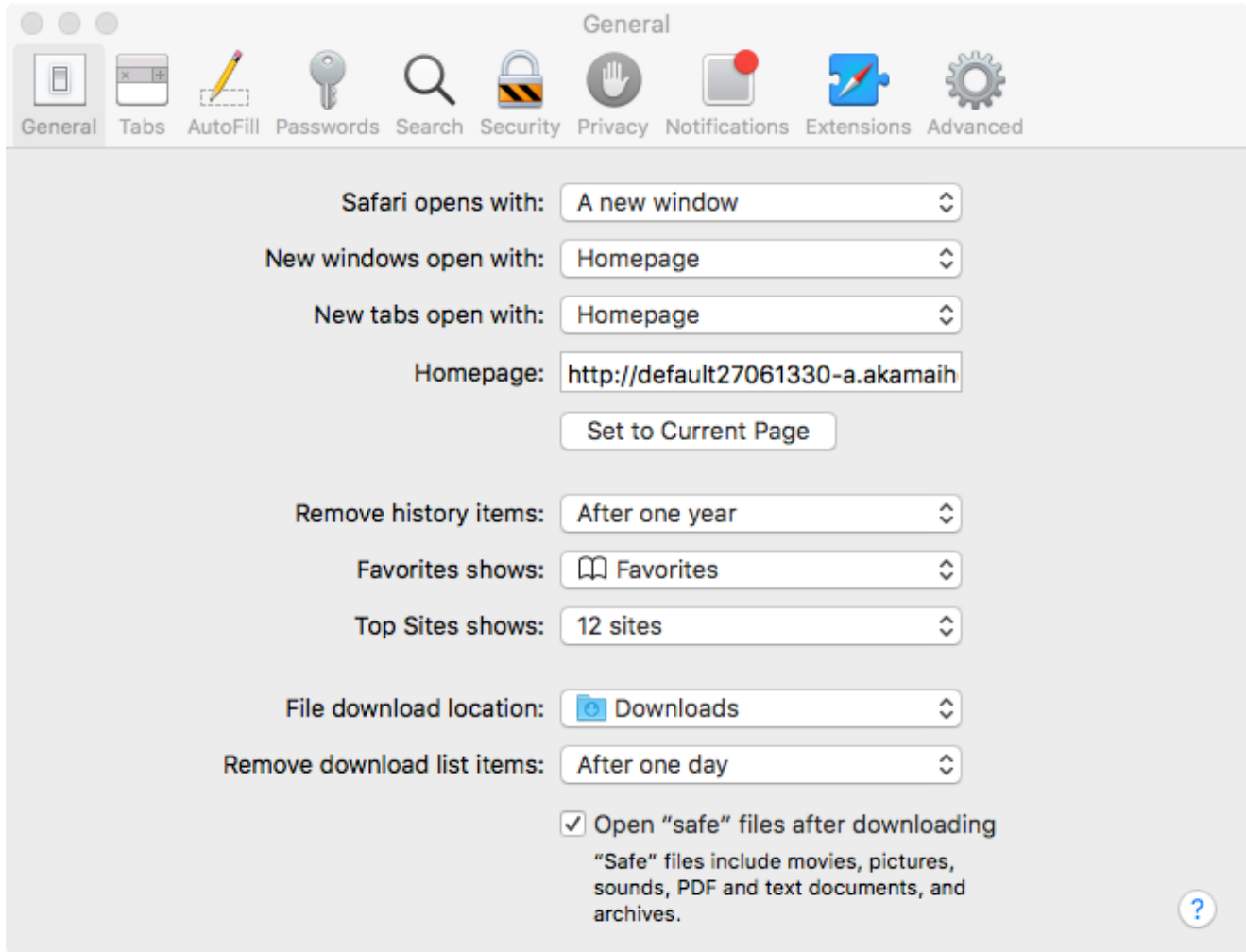
Communicating Files

Date scanned	Detections	File type	Name
2017-07-25	41/63	Win32 EXE	277aecadc0eff6d25fb12c67d9ae8392.virus
2017-07-25	42/64	Win32 EXE	JKKZVMYCUNSA.EXE
2017-07-25	41/64	Win32 EXE	522b2d3c5fc33876d6edde8d92f18e98.virus
2017-07-25	33/63	Win32 EXE	d:\documents and settings\murali\start menu\programs\startup\bsylkquka.exe
2017-07-24	38/64	Win32 EXE	38dc2d7a895476d8660b44dc1ae96b6a.virus
2017-07-24	42/63	Win32 EXE	b41fea402e00513e443fa41defc32f90.virus
2017-07-23	35/64	Win32 EXE	3e7da5288addf8b1ef4b1c3a908dcd415e07ff3
2017-07-22	41/63	Win32 EXE	3a284807452e28bc76dd6163357cbbb9.virus
2017-07-22	41/64	Win32 EXE	a6a2e9c936b2197bcca6e5067337b1c8.virus
2017-07-22	37/63	Win32 EXE	efcbb8ec33941512b066c320e02b47f8.virus

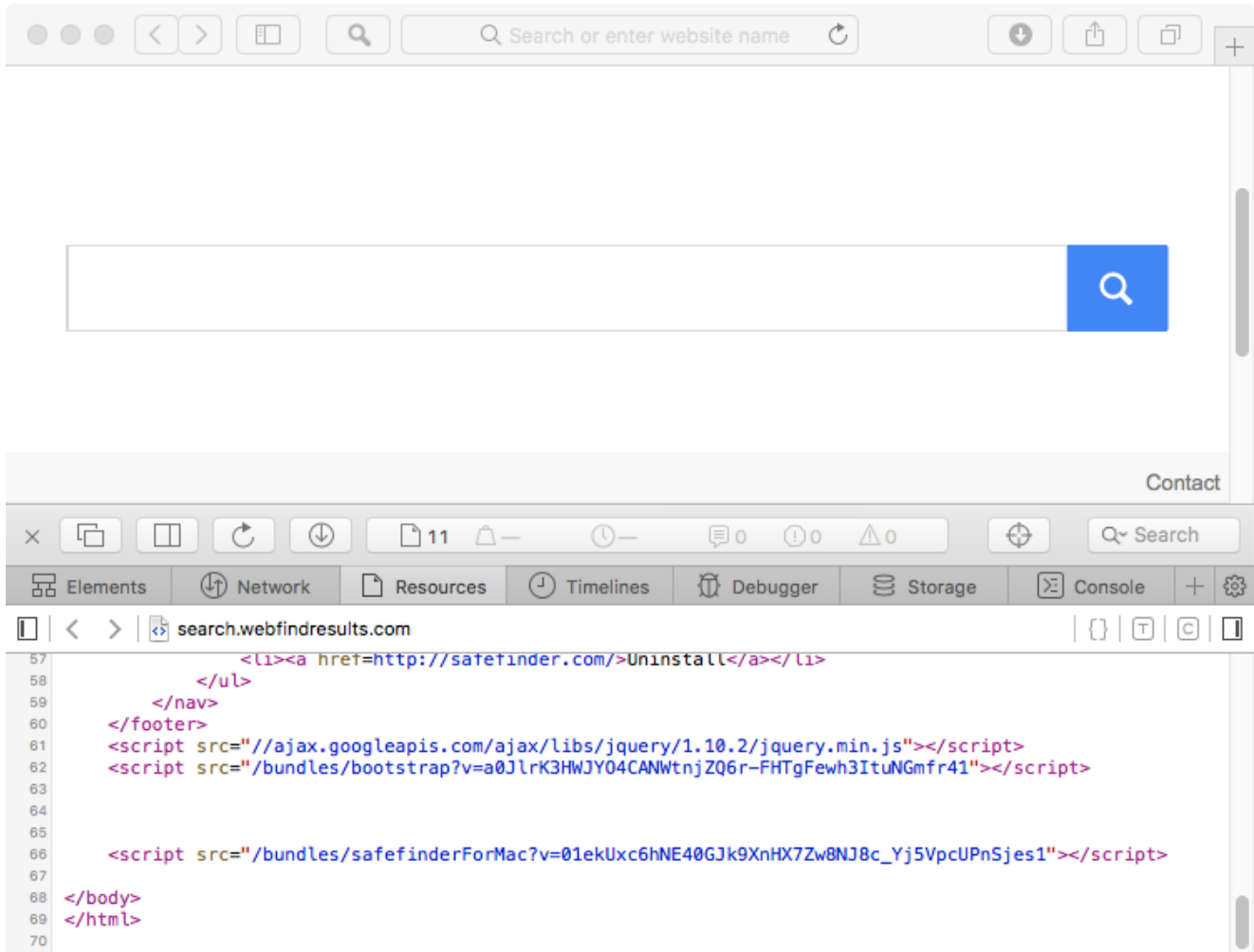
More

So what does the Mughthesecc binary actually do? Lets take a peek! However, I want to point out that I've learned (the hard way) that spending a large amount of time reversing adware can quickly drive one somewhat mad...so here, we'll only perform a cursory look.

A common tactic of adware is to hijack the victims browser (homepage, inject ads, etc) for financial gain. Mughthesecc (which is installed when the user "agrees" to install "Safe Finder") appears to conform to goal. Specifically we can see that Safari's home page has been set to `http://default27061330-a.akamaihd.net/s?q=@@@&_pg=564D4420-C090-470B-9C13-6760B31264E7`

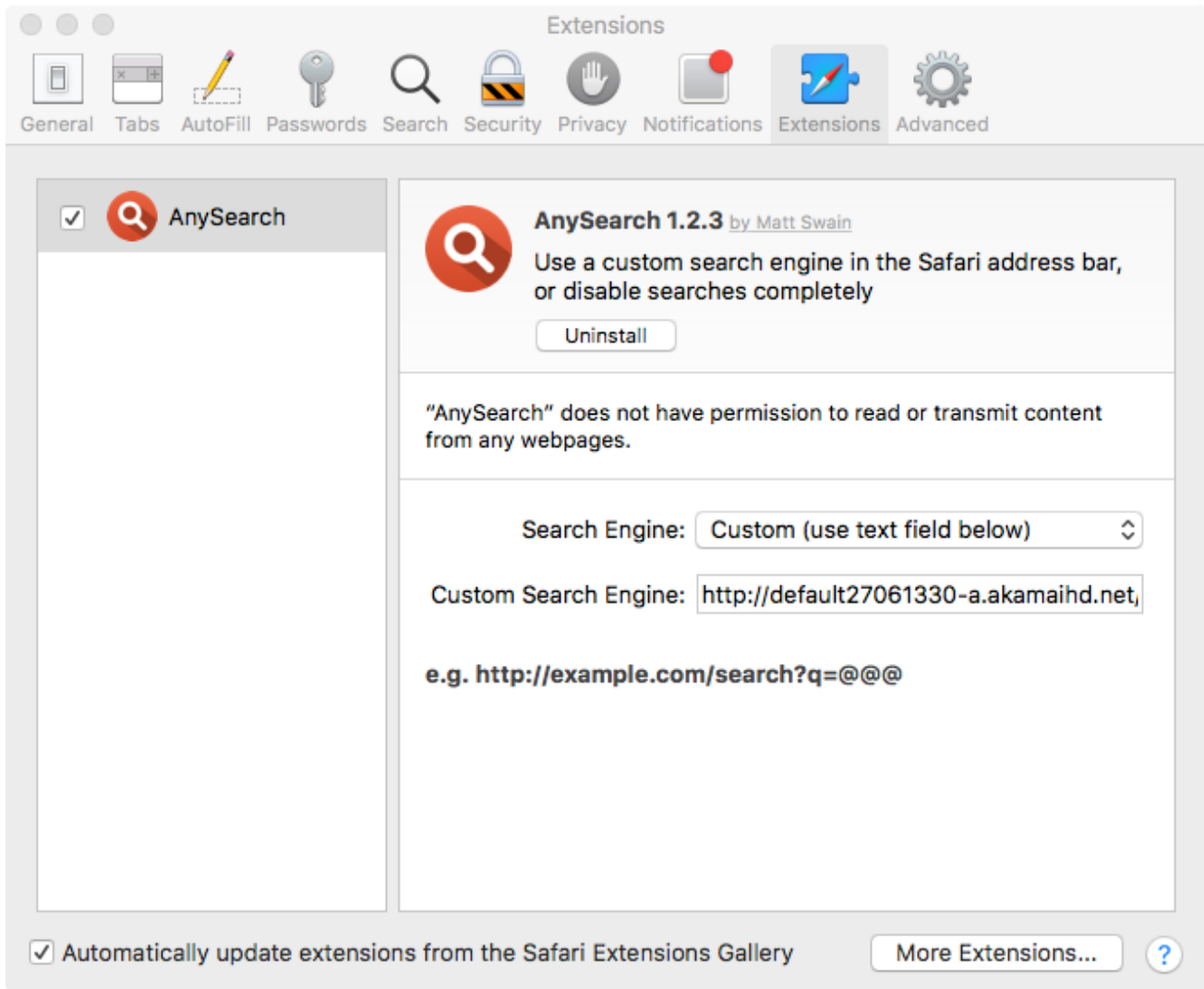


If we open Safari; indeed the home page has been hijacked - though in a seemingly innocuous way. It simply displays a rather 'clean' search page - though looking at the source, we can see the inclusion of several scripts 'Safe Finder' scripts:

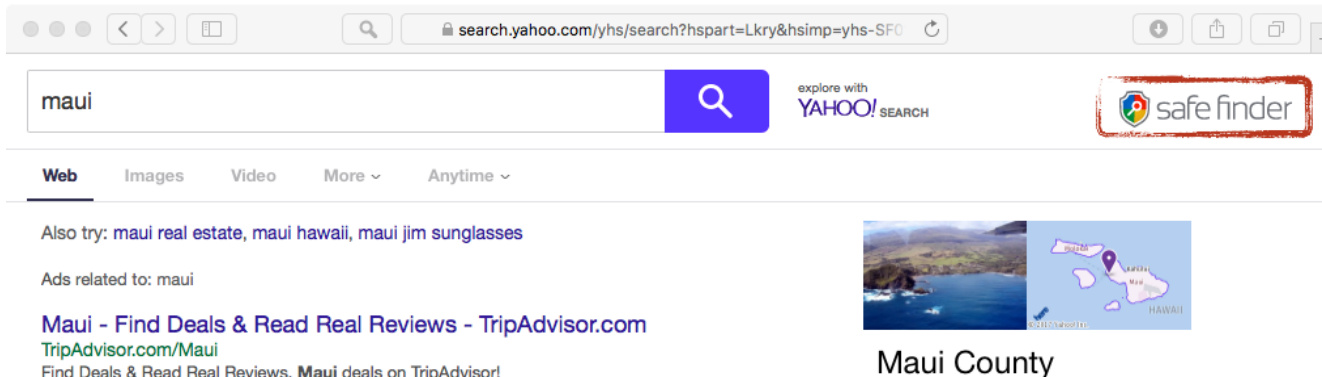


>

Also, examining the installed extensions we can see that an "Any Search" browser extension was installed:



Searches are funneled thru various affiliates, before ending up being serviced by Yahoo Search. However, 'Safe Finder' logic (such as an icon, and likely other scripts) are injected into all search results:



At this point, I'm calling it a night! It appears that Mughthesec is simply some 'run-of-the-mill' macOS malware. But is it new? Not likely. According to the mac adware analysis guru, Thomas Reed; this "looks like a new variant of something we call OperatorMac":



Thomas Reed


@thomasareed

Following

Replying to @gavrielstate @patrickwardle

Thanks, Patrick sent me the hash too. Looks like a new variant of something we call OperatorMac (though that may be a bad name).

Moreover, @noarfromspace pointed me towards several samples from earlier this year (spring?) that appear to be related:

	No engines detected this file
SHA-256	7b1d97d5e0823e1f496e0c3d8c07a333c1c420b0e98f44887e149286bf325c98
File name	mac
File size	475.67 KB
Last analysis	2017-03-15 02:08:27 UTC

0 / 56

Conclusion

In the blog post, we sought to answer the question, "What is Mughthesecc?" The answer; likely a new variant of the 'SafeFinder/OperatorMac' adware. Yes it's rather unsophisticated macOS malware, but it's installer is signed (to 'bypass' Gatekeeper) and at the time of this analysis no anti-virus engines were detected it....and mac users are being infected :|

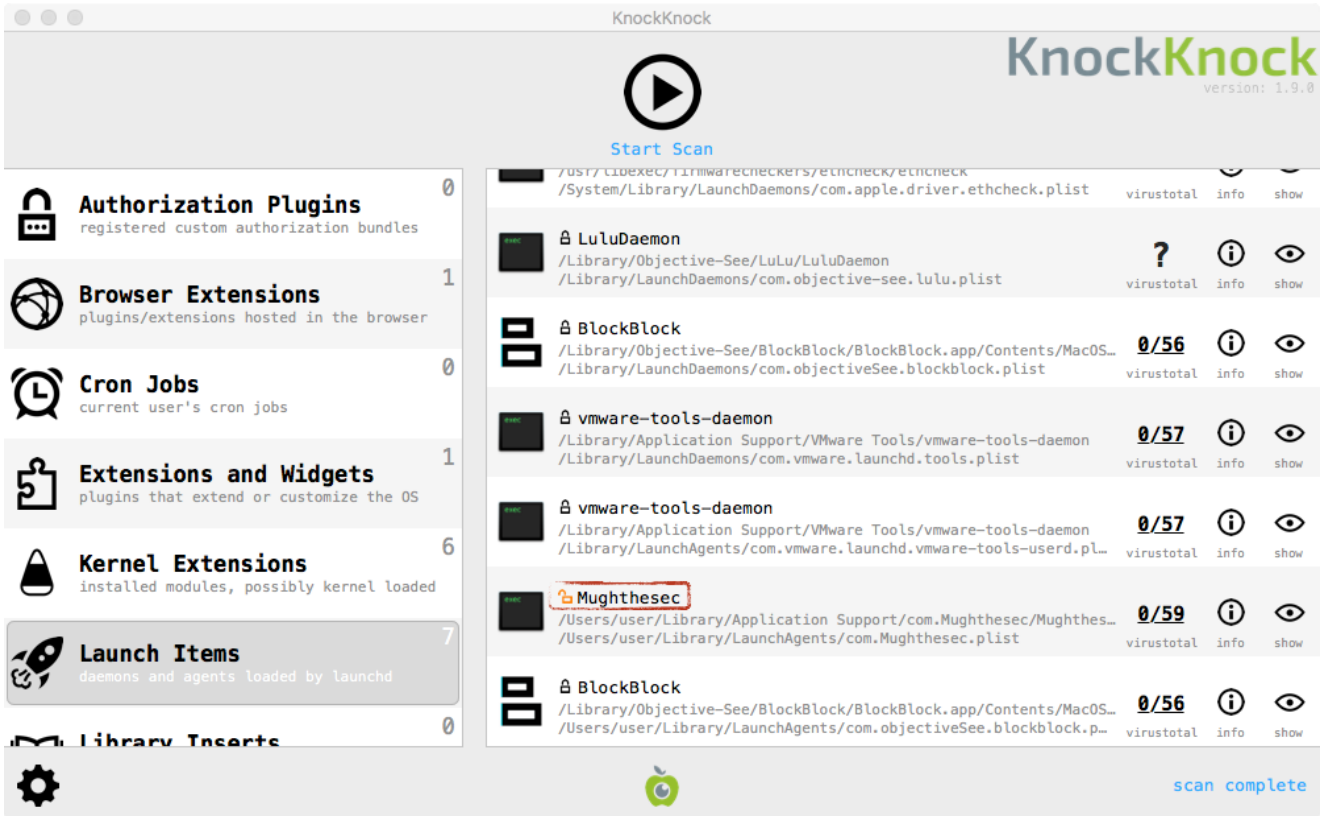
Speaking of infection, due to the fact that the installer is masquerading as Flash Player installer, it's likely that this adware is relying on common infection techniques to gain new victims. If I had to guess its infection vector is likely one (or all?) of the following:

- fake popups on 'shady' websites
- malicious ads, perhaps on legit websites

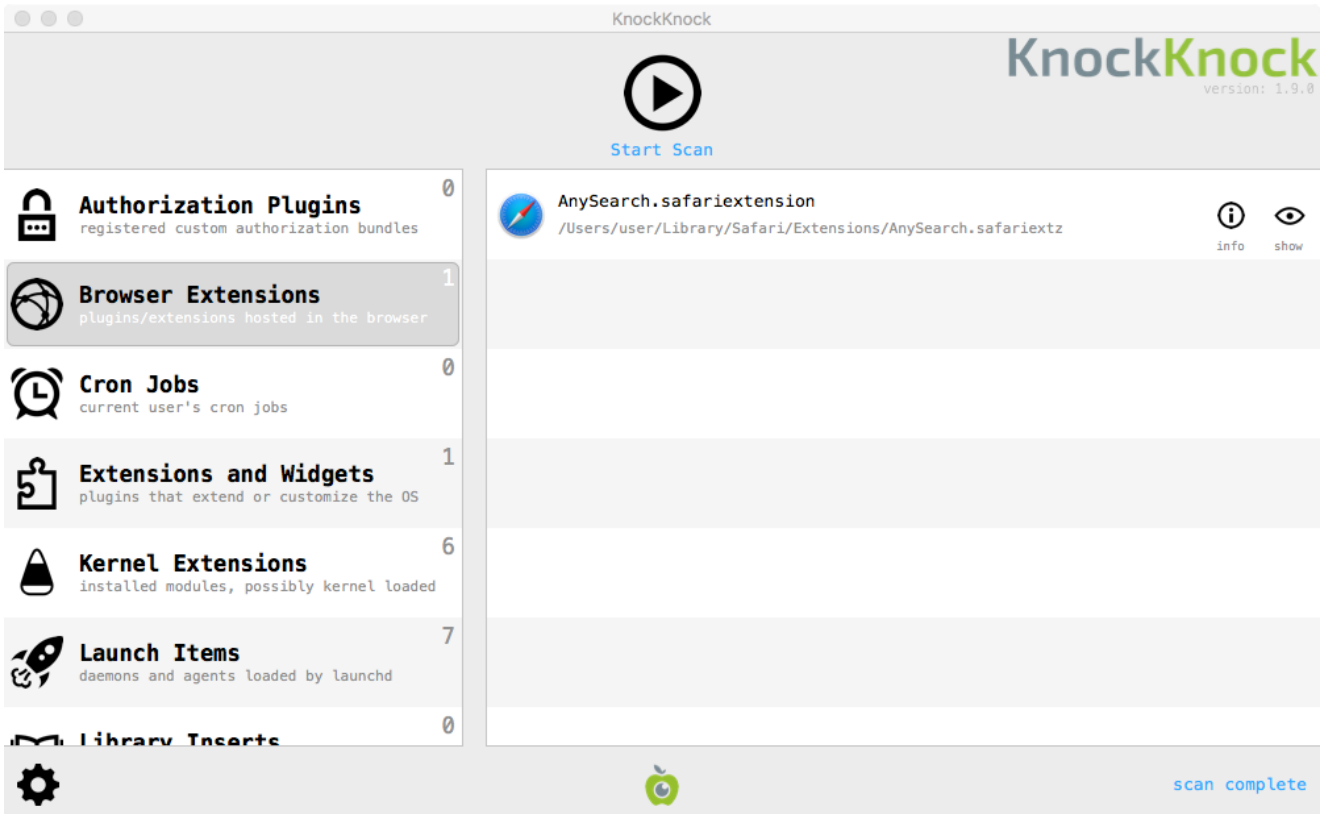
Either way, user-interaction is likely required.

In terms of detection, we showed how BlockBlock will alert when the adware goes to persist. Neat!

KnockKnock can also be used to (after the fact), to reveal infections. For example, it can reveal the presence of the unsigned launch agent:



And what about the malicious browser extension? Yup, KnockKnock can show that too:



Hooray! Objective-See FTW ❤️

To manually disinfect Mughthesecc:

- unload the launch agent via: launchctl unload
~/Library/LaunchAgents/com.Mughthesecc.plist
- delete ~/Library/Application Support/com.Mughthesecc/Mughthesecc
- delete ~/Library/LaunchAgents/com.Mughthesecc.plist
- delete the 'Any Search' browser extension

However, as we saw, the Installer application could install other 'adware' - so it's probably best to just reinstall macOS. Instructions [here](#).

love these blog posts & tools? you can support them via [patreon!](#) Mahalo :)

© 2017 objective-see llc