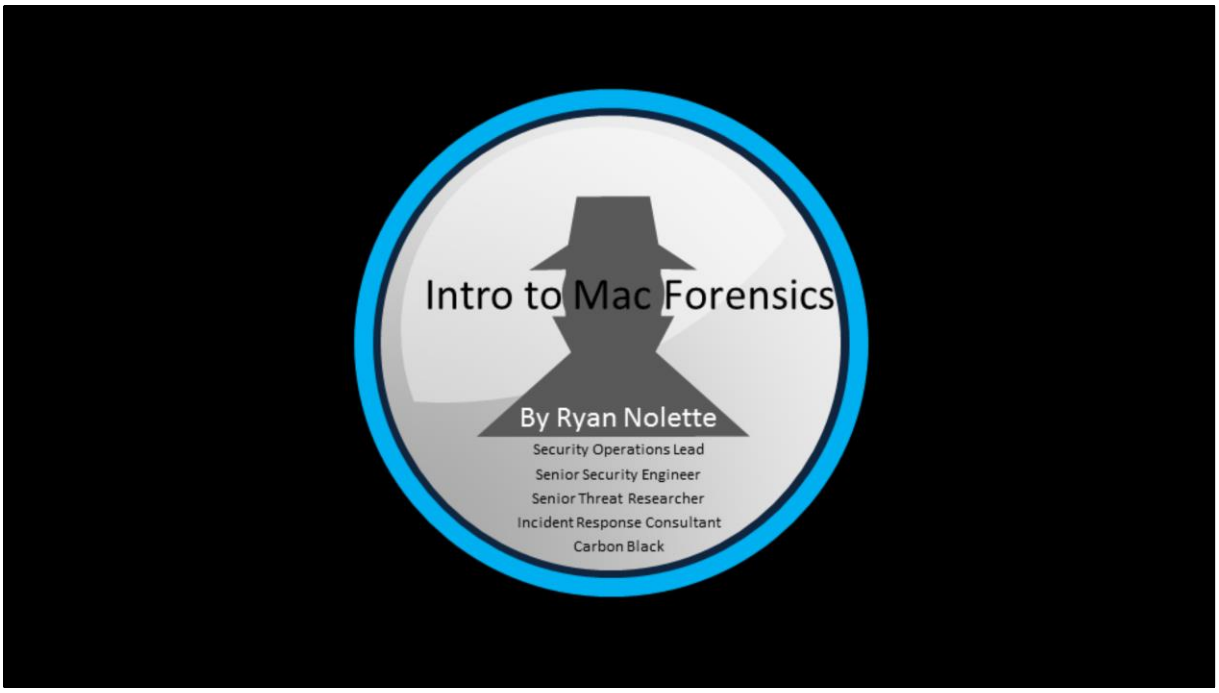


Intro to Mac Forensics

By Ryan Nolette

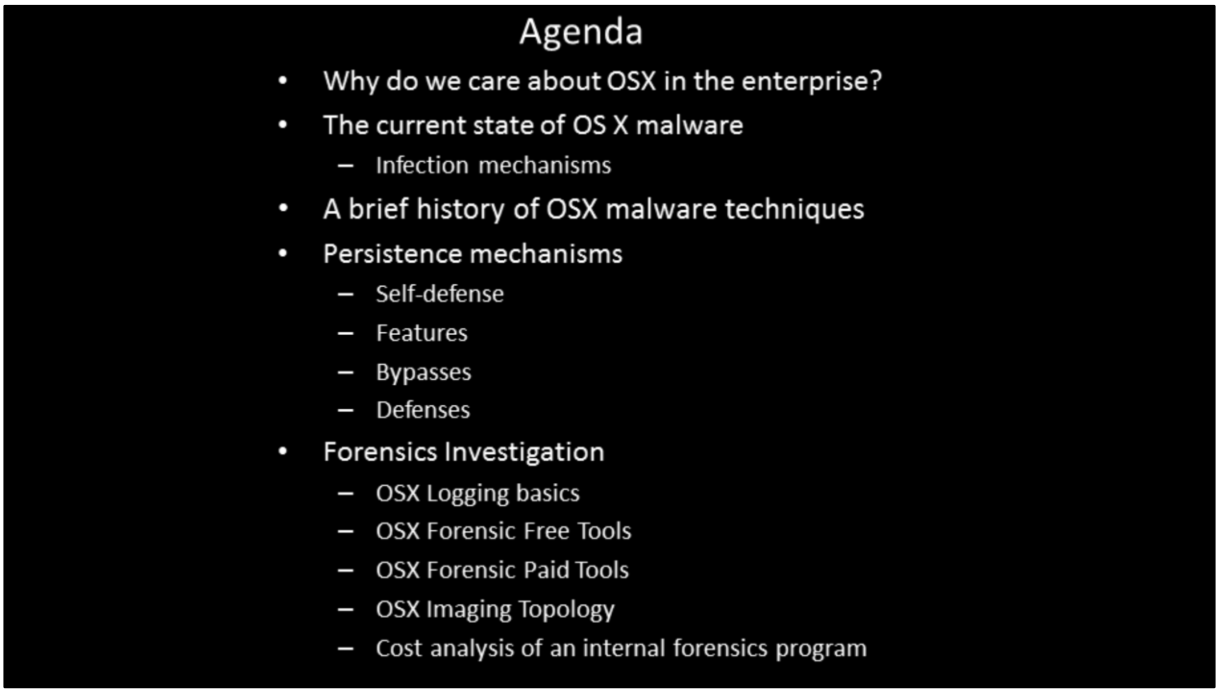
Security Operations Lead

Senior Security Engineer

Senior Threat Researcher

Incident Response Consultant

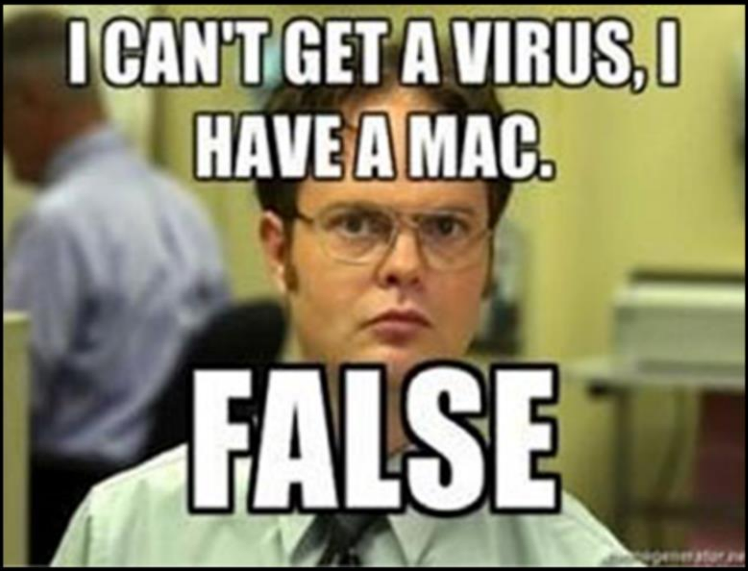
Carbon Black

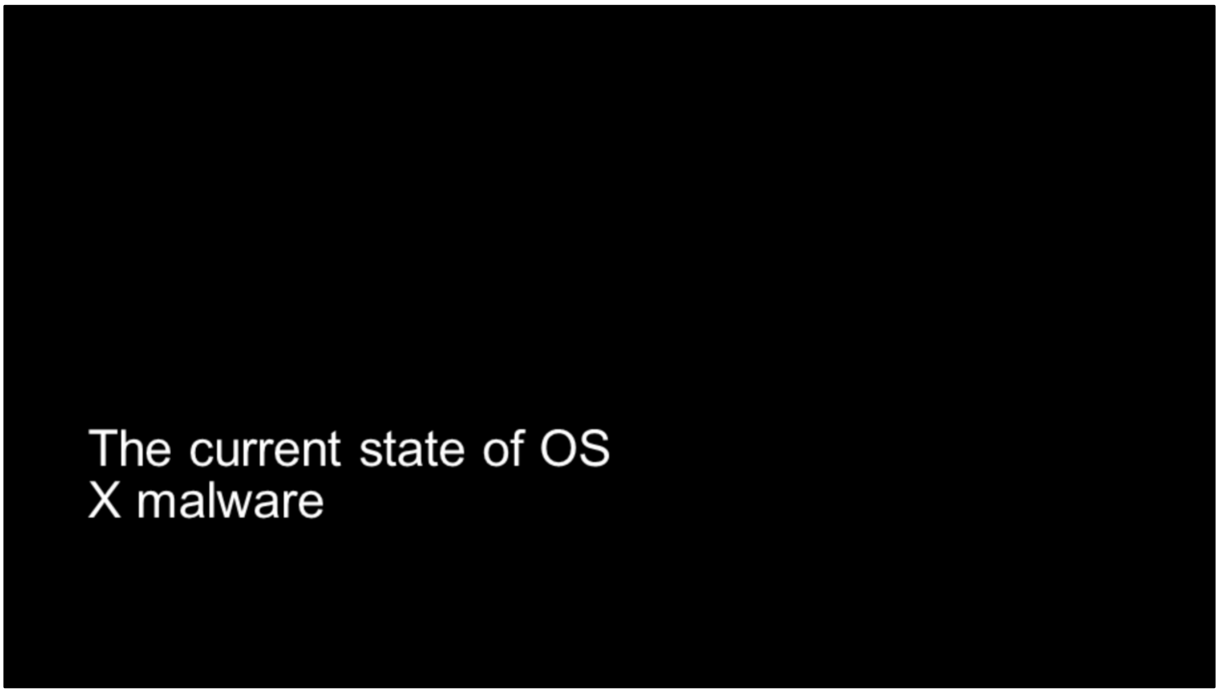


Agenda

- Why do we care about OSX in the enterprise?
- The current state of OS X malware
  - Infection mechanisms
- A brief history of OSX malware techniques
- Persistence mechanisms
  - Self-defense
  - Features
  - Bypasses
  - Defenses
- Forensics Investigation
  - OSX Logging basics
  - OSX Forensic Free Tools
  - OSX Forensic Paid Tools
  - OSX Imaging Topology
  - Cost analysis of an internal forensics program

Let’s get this out of the way now

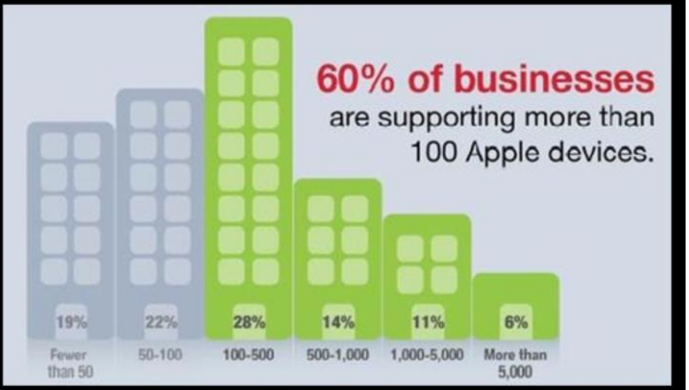




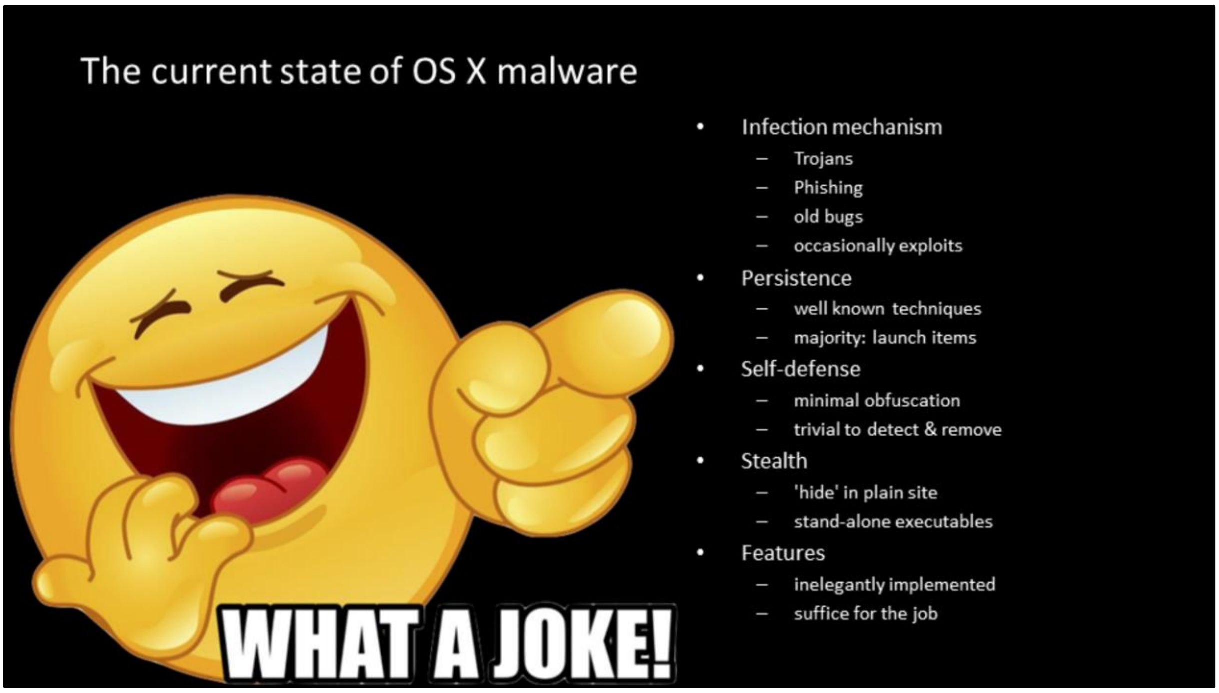
The current state of OS X malware

Why do we care about OSX in the enterprise?

- over 90% of businesses use Apple products
  - 91% supporting iPhones
  - 89% supporting iPads
  - 60% supporting Macs.
- “It doesn’t get PC viruses. A Mac isn’t susceptible to the thousands of viruses plaguing Windows-based computers.” - apple.com (2012)
- Mac Malware timeline:
  - [1982] ‘first’ virus (elk cloner) infected apple II’s
  - "[2014] nearly 1000 unique attacks on Macs; 25 major families" –Kaspersky

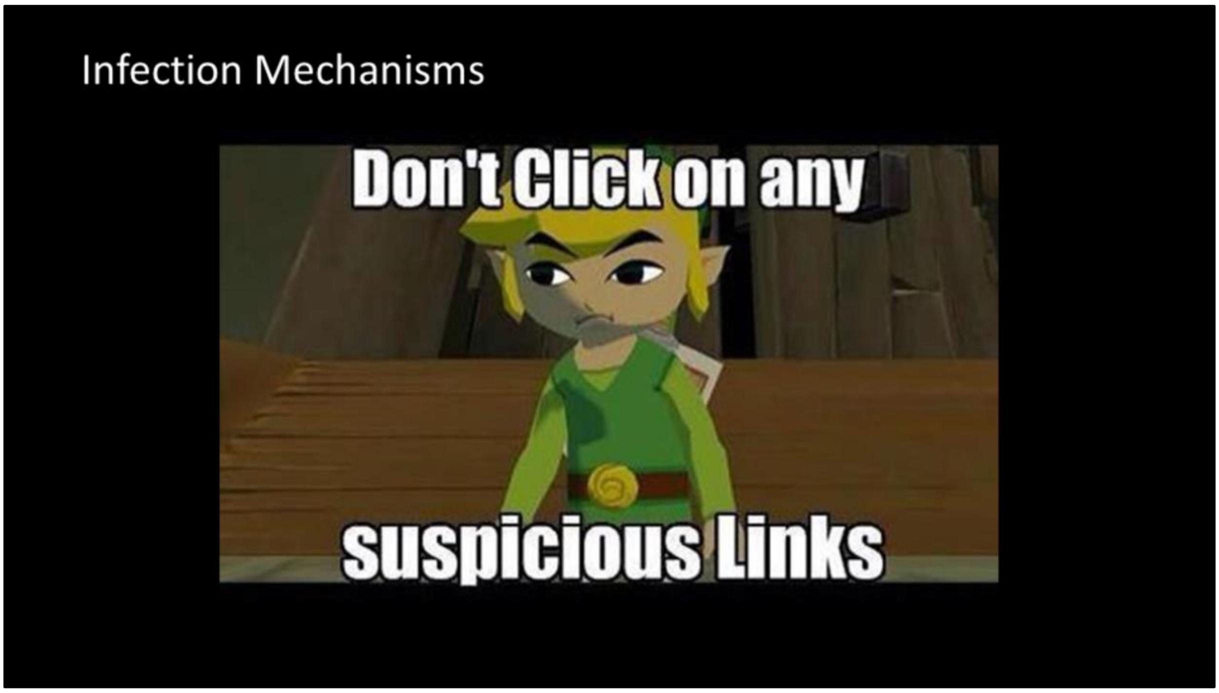


http://www.informationweek.com/infrastructure/pc-and-servers/mac-enterprise-adoption-grows/d/d-id/1299[illegible]



The current state of OS X malware

- Infection mechanism
  - Trojans
  - Phishing
  - old bugs
  - occasionally exploits
- Persistence
  - well known techniques
  - majority: launch items
- Self-defense
  - minimal obfuscation
  - trivial to detect & remove
- Stealth
  - 'hide' in plain site
  - stand-alone executables
- Features
  - inelegantly implemented
  - suffice for the job

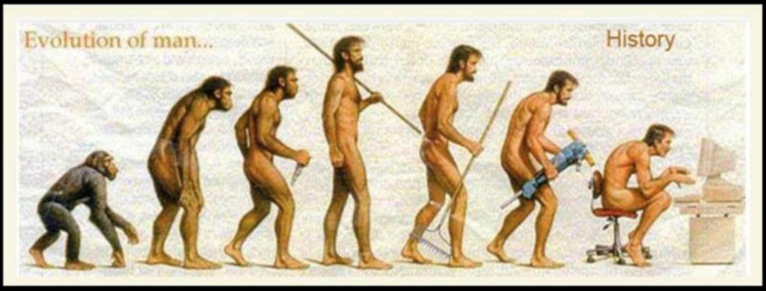


Infection Mechanisms

Same as PC
Primary attack vectors are email, drive by downloads, and infected binaries.
Mac has the unique attack vector of a closed ecosystem which implies a false sense of trust

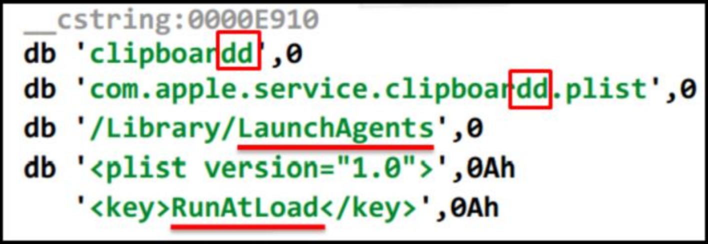
A Little Bit of

HISTORY



OSX/XSLCMD

- provides reverse shell, keylogging, & screen capture
- “a previously unknown variant of the APT backdoor XSLCmd which is designed to compromise Apple OS X systems”



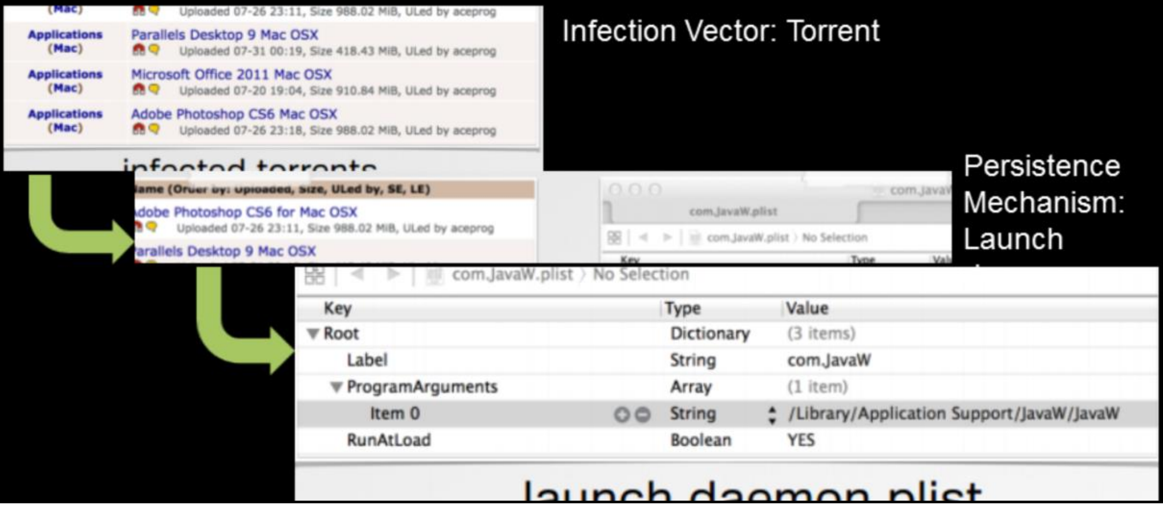
```
__cstring:0000E910
db 'clipboardd',0
db 'com.apple.service.clipboardd.plist',0
db '/Library/LaunchAgents',0
db '<plist version="1.0">',0Ah
   '<key>RunAtLoad</key>',0Ah
```

OSX/IWORM

- ‘standard’ backdoor, providing survey, download/execute, etc.
  - [illegible]

Infection Vector: Torrent

Persistence Mechanism: Launch



(Mac)	Uploaded 07-26 23:11, Size 988.02 MiB, ULed by aceprog
Applications (Mac)	Parallels Desktop 9 Mac OSX Uploaded 07-31 00:19, Size 418.43 MiB, ULed by aceprog
Applications (Mac)	Microsoft Office 2011 Mac OSX Uploaded 07-20 19:04, Size 910.84 MiB, ULed by aceprog
Applications (Mac)	Adobe Photoshop CS6 Mac OSX Uploaded 07-26 23:18, Size 988.02 MiB, ULed by aceprog

infected torrents

ame (Order by: Uploaded, Size, ULed by, SE, LE)

dobe Photoshop CS6 for Mac OSX
Uploaded 07-26 23:11, Size 988.02 MiB, ULed by aceprog

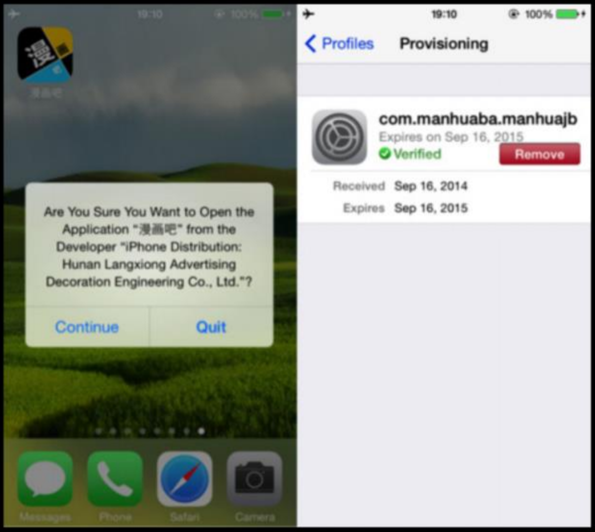
arallels Desktop 9 Mac OSX

com.JavaW.plist 〉 No Selection

Key	Type	Value
▼ Root	Dictionary	(3 items)
Label	String	com.JavaW
▼ ProgramArguments	Array	(1 item)
Item 0	String	/Library/Application Support/JavaW/JavaW
RunAtLoad	Boolean	YES

launch daemon plist

OSX/WIRELURKER


Are You Sure You Want to Open the Application "漫画吧" from the Developer "iPhone Distribution: Hunan Langxiong Advertising Decoration Engineering Co., Ltd."?

Continue | Quit

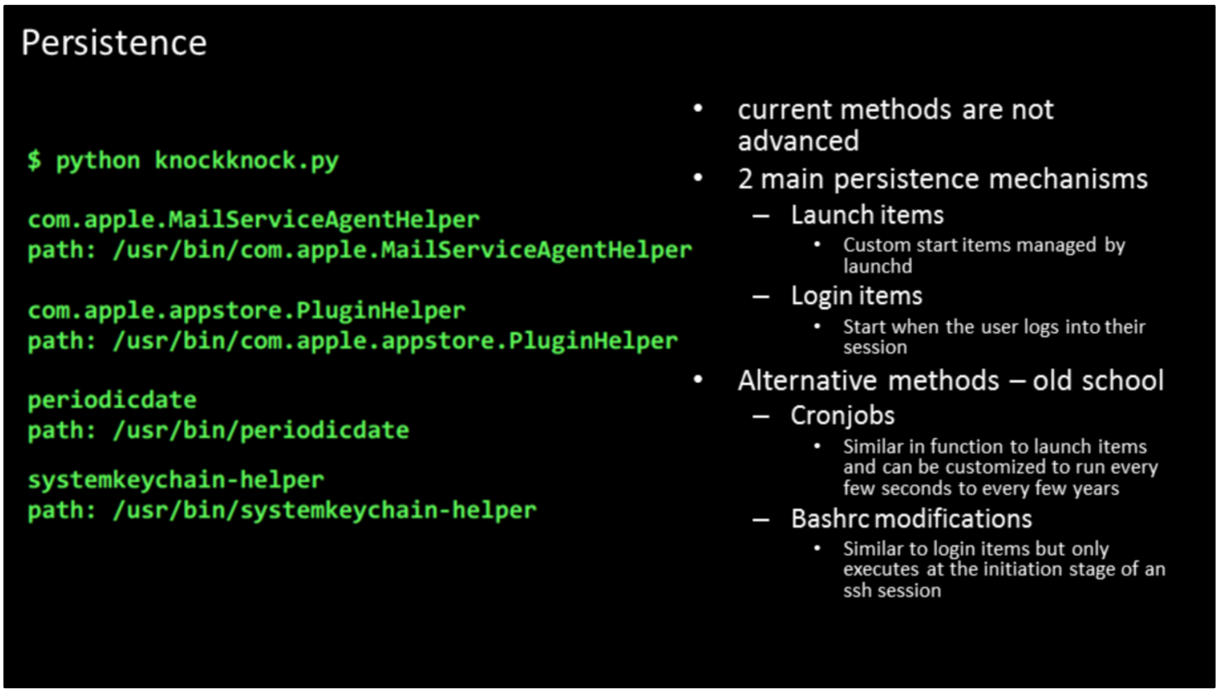
Profiles | Provisioning

com.manhuaba.manhuajb
Expires on Sep 16, 2015
Verified
Remove

Received Sep 16, 2014
Expires Sep 16, 2015

- an iOS infector (via USB)
- "a collection of scripts, plists, & binaries all duct-taped together... making it easy to detect." -j zdziarski

Shellshock/Mayhem (2014)
Striking at the terminal strikes at the heart of Linux, which is why the recent Mayhem attacks – which targeted the so-called Shellshock vulnerabilities in Linux's Bash command-line interpreter using a specially crafted ELF library – were so noteworthy. Researchers at Yandex said that the network had snared 1,400 victims as of July.



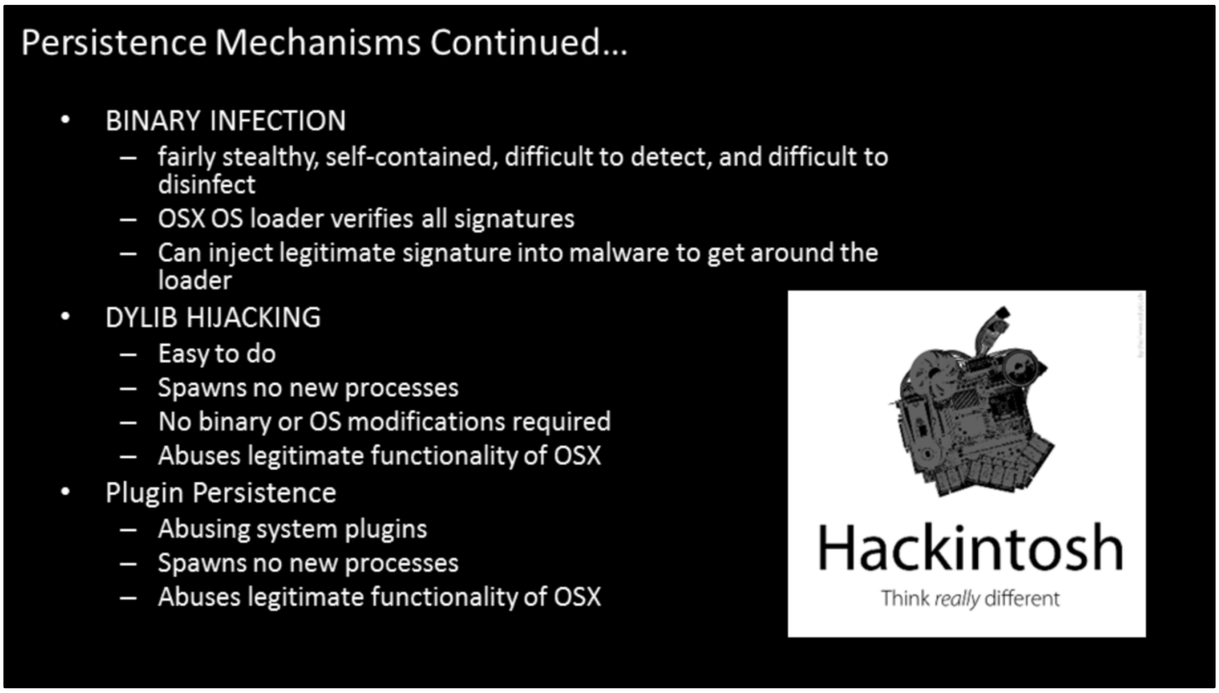
Persistence

```
$ python knockknock.py

com.apple.MailServiceAgentHelper
path: /usr/bin/com.apple.MailServiceAgentHelper

com.apple.appstore.PluginHelper
path: /usr/bin/com.apple.appstore.PluginHelper

periodicdate
path: /usr/bin/periodicdate

systemkeychain-helper
path: /usr/bin/systemkeychain-helper
```

- current methods are not advanced
- 2 main persistence mechanisms
  - Launch items
    - Custom start items managed by launchd
  - Login items
    - Start when the user logs into their session
- Alternative methods – old school
  - Cronjobs
    - Similar in function to launch items and can be customized to run every few seconds to every few years
  - Bashrc modifications
    - Similar to login items but only executes at the initiation stage of an ssh session

The issue with launch items and login items is that they are easily visible, easy to detect, and are well known features.

Consider the Mac equivalent to the run and runonce registry keys

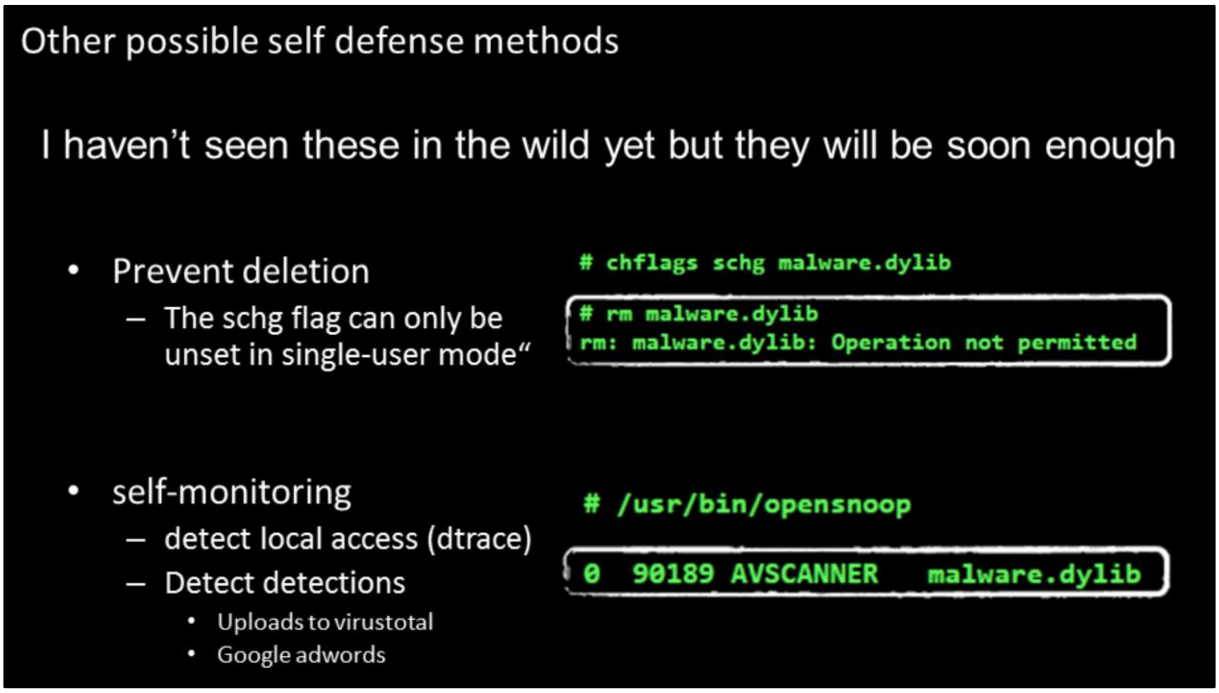


Persistence Mechanisms Continued…

- BINARY INFECTION
  - fairly stealthy, self-contained, difficult to detect, and difficult to disinfect
  - OSX OS loader verifies all signatures
  - Can inject legitimate signature into malware to get around the loader
- DYLIB HIJACKING
  - Easy to do
  - Spawns no new processes
  - No binary or OS modifications required
  - Abuses legitimate functionality of OSX
- Plugin Persistence
  - Abusing system plugins
  - Spawns no new processes
  - Abuses legitimate functionality of OSX

Mac malware SELF-DEFENSE

- Currently, essentially non-existent
- Poor crypto implementations
- Tries to hide in plain sight
- Easy to find
- Easy to analyze
- Easy to disinfect



Other possible self defense methods

I haven't seen these in the wild yet but they will be soon enough

- Prevent deletion
  - The schg flag can only be unset in single-user mode“

```
# chflags schg malware.dylib
```

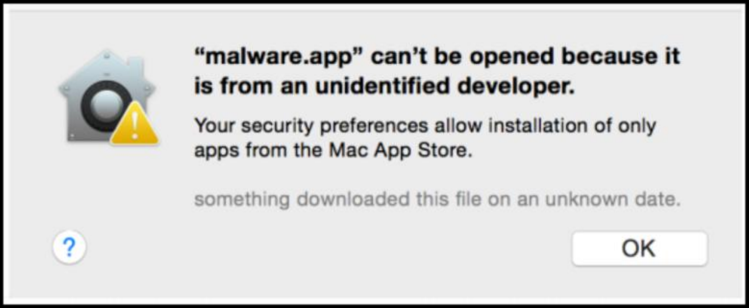
```
# rm malware.dylib
rm: malware.dylib: Operation not permitted
```

- self-monitoring
  - detect local access (dtrace)
  - Detect detections
    - Uploads to virustotal
    - Google adwords

```
# /usr/bin/opensnoop
```

```
0  90189 AVSCANNER   malware.dylib
```

Gatekeeper



"malware.app" can't be opened because it is from an unidentified developer.

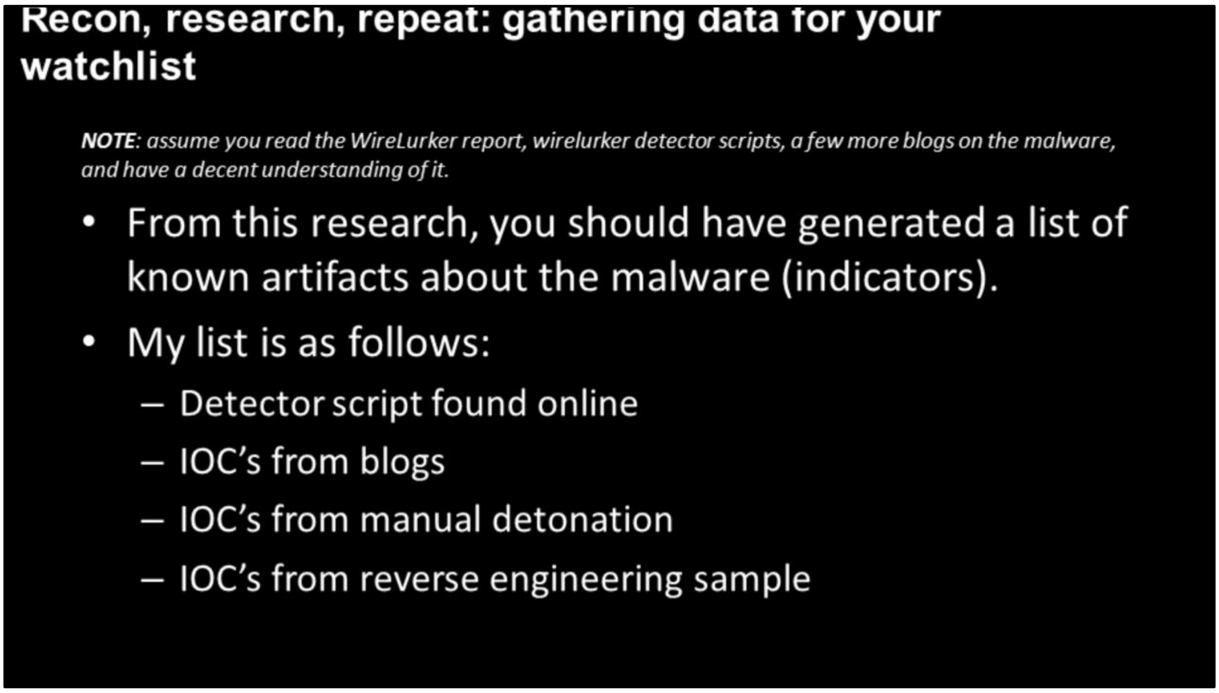
Your security preferences allow installation of only apps from the Mac App Store.

something downloaded this file on an unknown date.

- Gatekeeper blocking untrusted code
- somewhat effective, but most users should be ok.

Bypassing gatekeeper is very easy and can be a whole presentation on its own because it is interesting from a defense perspective

Defending against Wirelurker

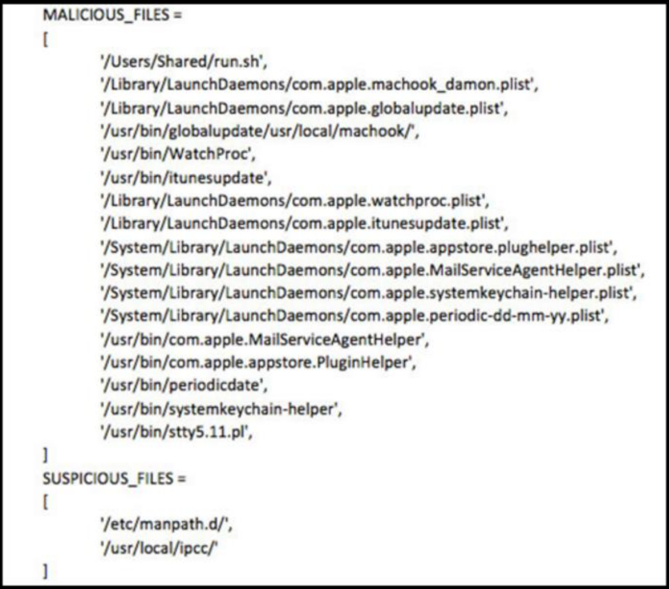


Recon, research, repeat: gathering data for your watchlist

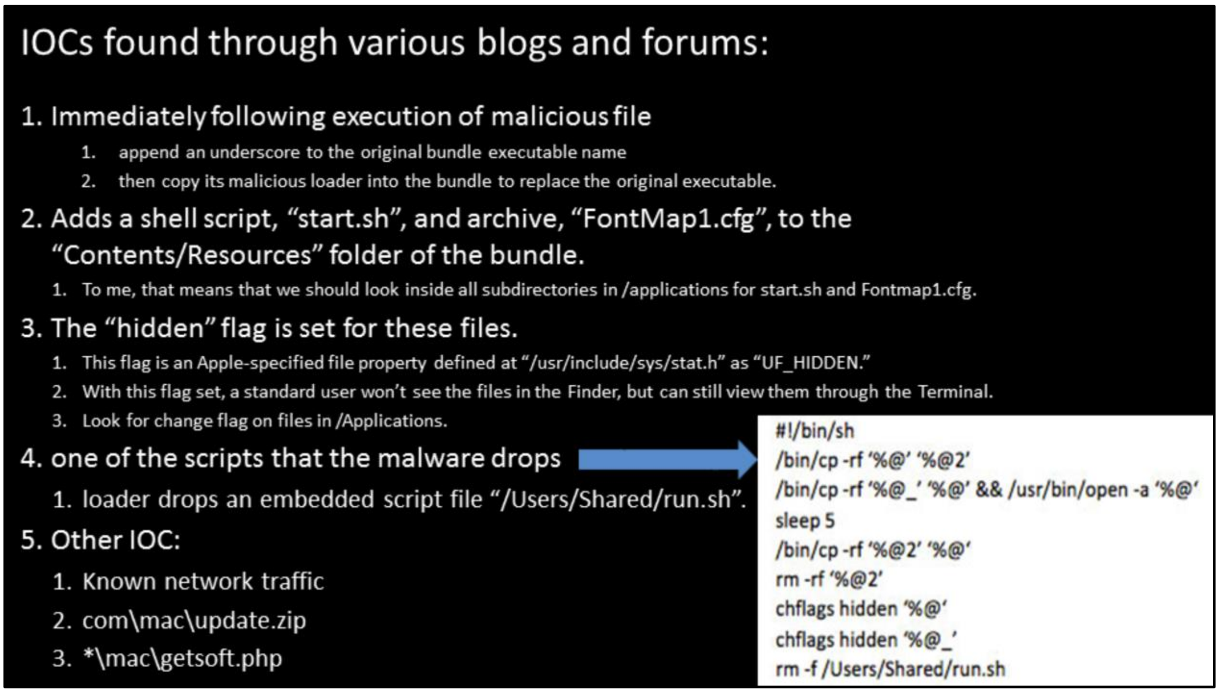
***NOTE**: assume you read the WireLurker report, wirelurker detector scripts, a few more blogs on the malware, and have a decent understanding of it.*

- From this research, you should have generated a list of known artifacts about the malware (indicators).
- My list is as follows:
  - Detector script found online
  - IOC’s from blogs
  - IOC’s from manual detonation
  - IOC’s from reverse engineering sample

Taken from detector script:



```
MALICIOUS_FILES =
[
        '/Users/Shared/run.sh',
        '/Library/LaunchDaemons/com.apple.machook_damon.plist',
        '/Library/LaunchDaemons/com.apple.globalupdate.plist',
        '/usr/bin/globalupdate/usr/local/machook/',
        '/usr/bin/WatchProc',
        '/usr/bin/itunesupdate',
        '/Library/LaunchDaemons/com.apple.watchproc.plist',
        '/Library/LaunchDaemons/com.apple.itunesupdate.plist',
        '/System/Library/LaunchDaemons/com.apple.appstore.plughelper.plist',
        '/System/Library/LaunchDaemons/com.apple.MailServiceAgentHelper.plist',
        '/System/Library/LaunchDaemons/com.apple.systemkeychain-helper.plist',
        '/System/Library/LaunchDaemons/com.apple.periodic-dd-mm-yy.plist',
        '/usr/bin/com.apple.MailServiceAgentHelper',
        '/usr/bin/com.apple.appstore.PluginHelper',
        '/usr/bin/periodicdate',
        '/usr/bin/systemkeychain-helper',
        '/usr/bin/stty5.11.pl',
]
SUSPICIOUS_FILES =
[
        '/etc/manpath.d/',
        '/usr/local/ipcc/'
]
```

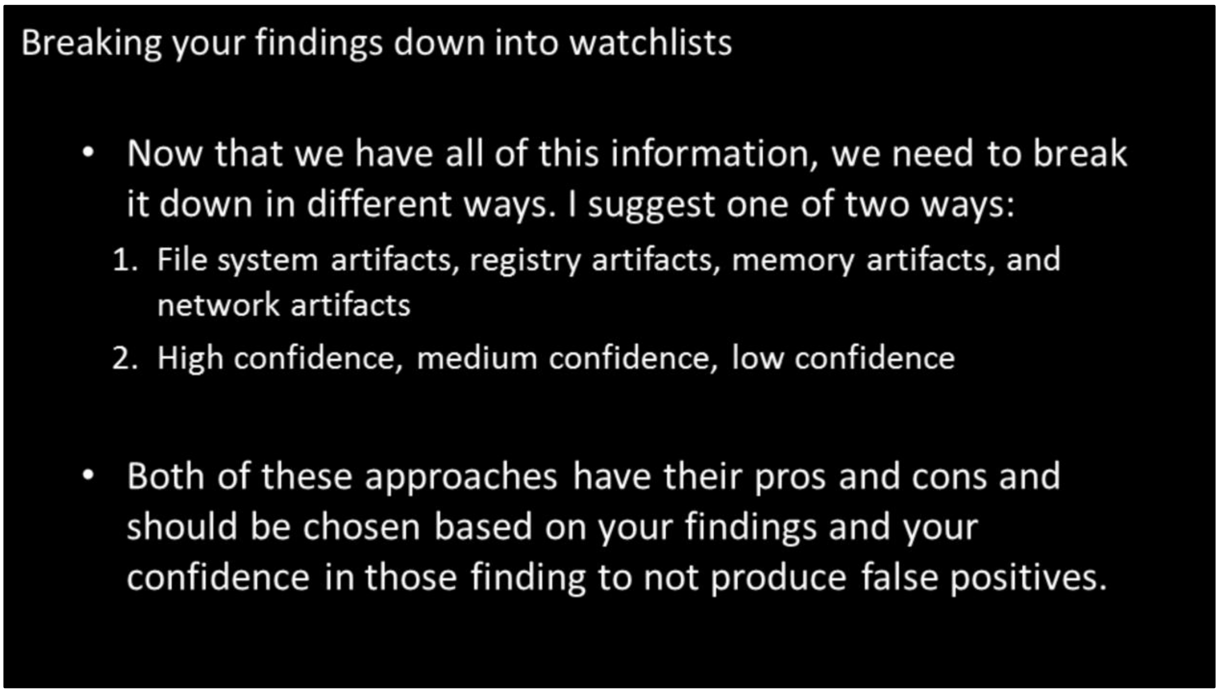


IOCs found through various blogs and forums:

1. Immediately following execution of malicious file
   1. append an underscore to the original bundle executable name
   2. then copy its malicious loader into the bundle to replace the original executable.
2. Adds a shell script, “start.sh”, and archive, “FontMap1.cfg”, to the “Contents/Resources” folder of the bundle.
   1. To me, that means that we should look inside all subdirectories in /applications for start.sh and Fontmap1.cfg.
3. The “hidden” flag is set for these files.
   1. This flag is an Apple-specified file property defined at “/usr/include/sys/stat.h” as “UF_HIDDEN.”
   2. With this flag set, a standard user won’t see the files in the Finder, but can still view them through the Terminal.
   3. Look for change flag on files in /Applications.
4. one of the scripts that the malware drops
   1. loader drops an embedded script file “/Users/Shared/run.sh”.
5. Other IOC:
   1. Known network traffic
   2. com\mac\update.zip
   3. *\mac\getsoft.php

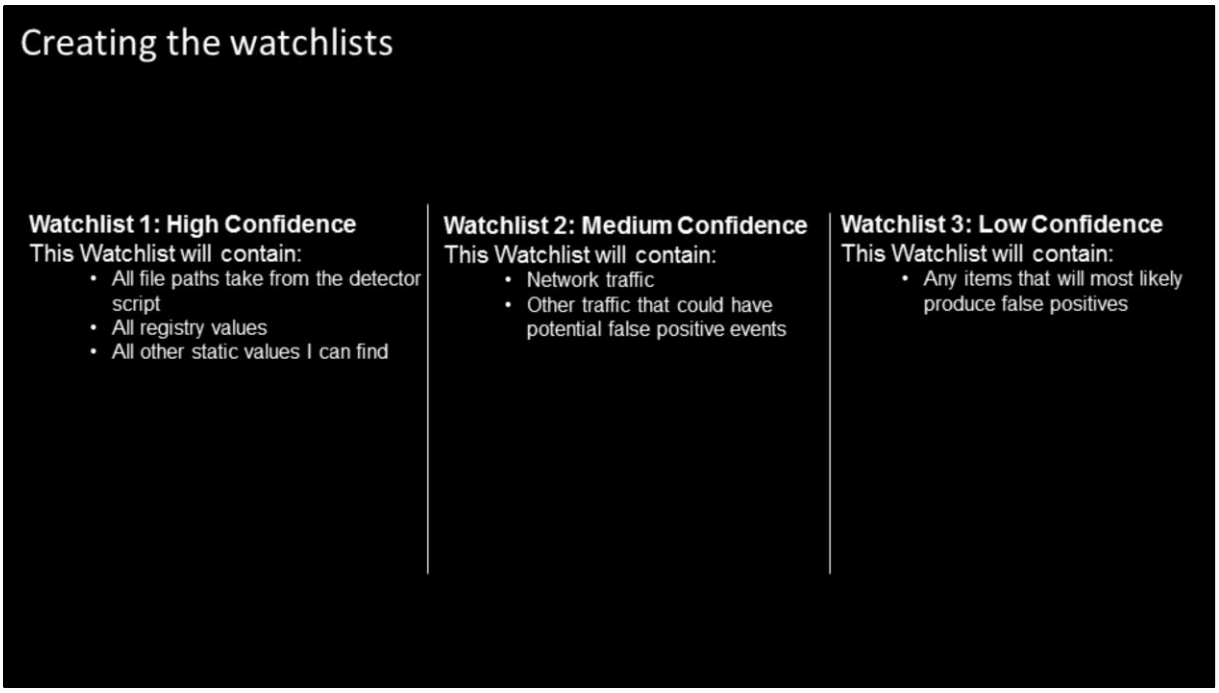
```
#!/bin/sh
/bin/cp -rf '%@' '%@2'
/bin/cp -rf '%@_' '%@' && /usr/bin/open -a '%@'
sleep 5
/bin/cp -rf '%@2' '%@'
rm -rf '%@2'
chflags hidden '%@'
chflags hidden '%@_'
rm -f /Users/Shared/run.sh
```

Now, your list may be different than mine. That’s OK. The biggest perk of the watchlists, in my opinion, is their flexibility and ease of updating/adapting to incorporate new information. Basically, the more you learn, the more the feed can be refined for efficiency and effectiveness in your environment.


Breaking your findings down into watchlists

- Now that we have all of this information, we need to break it down in different ways. I suggest one of two ways:
  1. File system artifacts, registry artifacts, memory artifacts, and network artifacts
  2. High confidence, medium confidence, low confidence

- Both of these approaches have their pros and cons and should be chosen based on your findings and your confidence in those finding to not produce false positives.



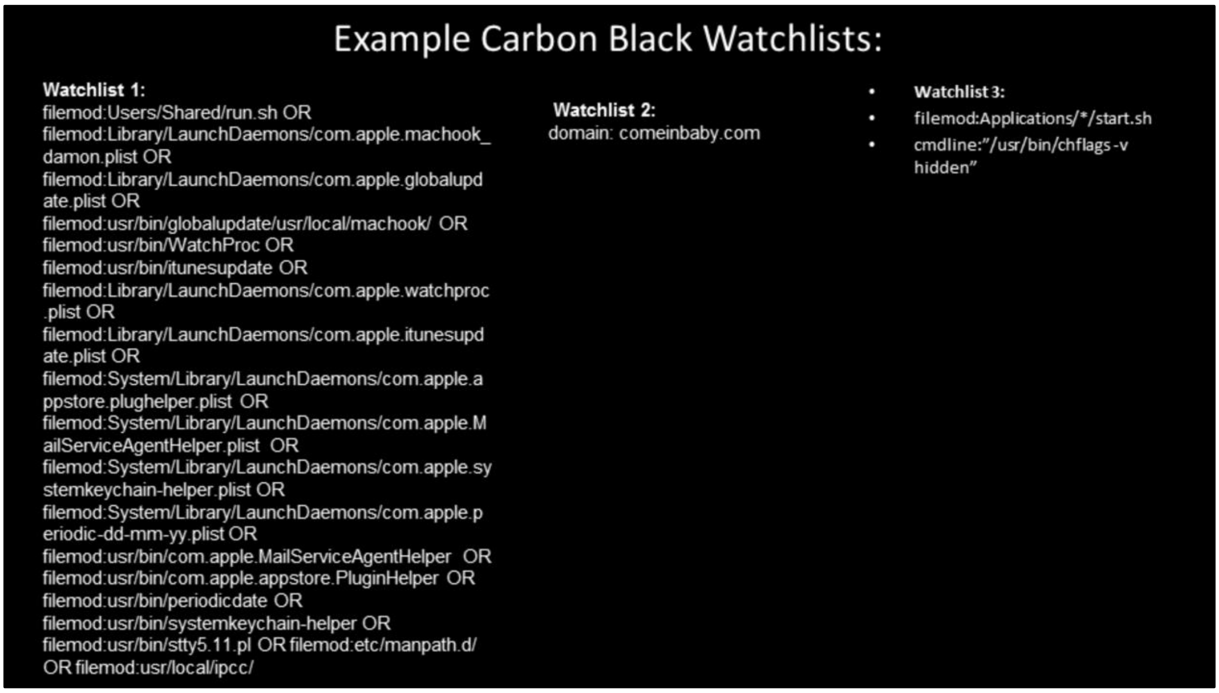
Creating the watchlists

Watchlist 1: High Confidence
This Watchlist will contain:
- All file paths take from the detector script
- All registry values
- All other static values I can find

Watchlist 2: Medium Confidence
This Watchlist will contain:
- Network traffic
- Other traffic that could have potential false positive events

Watchlist 3: Low Confidence
This Watchlist will contain:
- Any items that will most likely produce false positives

I chose to go with the three-tiered confidence method. I chose this approach because of my confidence in the data gathered. I think a few of these rules could produce false positive events in my environment and because of that, I have chosen the approach that allows me to separate these possible problem rules to unique watchlists. This approach will allow me to disable any noisy watchlists without turning everything off and keep my environment quiet, secure and functional.



Example Carbon Black Watchlists:

Watchlist 1:

```
filemod:Users/Shared/run.sh OR
filemod:Library/LaunchDaemons/com.apple.machook_
damon.plist OR
filemod:Library/LaunchDaemons/com.apple.globalupd
ate.plist OR
filemod:usr/bin/globalupdate/usr/local/machook/ OR
filemod:usr/bin/WatchProc OR
filemod:usr/bin/itunesupdate OR
filemod:Library/LaunchDaemons/com.apple.watchproc
.plist OR
filemod:Library/LaunchDaemons/com.apple.itunesupd
ate.plist OR
filemod:System/Library/LaunchDaemons/com.apple.a
ppstore.plughelper.plist OR
filemod:System/Library/LaunchDaemons/com.apple.M
ailServiceAgentHelper.plist OR
filemod:System/Library/LaunchDaemons/com.apple.sy
stemkeychain-helper.plist OR
filemod:System/Library/LaunchDaemons/com.apple.p
eriodic-dd-mm-yy.plist OR
filemod:usr/bin/com.apple.MailServiceAgentHelper OR
filemod:usr/bin/com.apple.appstore.PluginHelper OR
filemod:usr/bin/periodicdate OR
filemod:usr/bin/systemkeychain-helper OR
filemod:usr/bin/stty5.11.pl OR filemod:etc/manpath.d/
OR filemod:usr/local/ipcc/
```

Watchlist 2:

```
domain: comeinbaby.com
```

- **Watchlist 3:**
- filemod:Applications/*/start.sh
- cmdline:"/usr/bin/chflags -v hidden"

Watchlist 1:
This watchlist contains all of the file artifacts I gathered. These are all indicators that if I see them, I know they are not false positives and that I should immediately take action. I have high confidence in these indicators and am treating them as such.

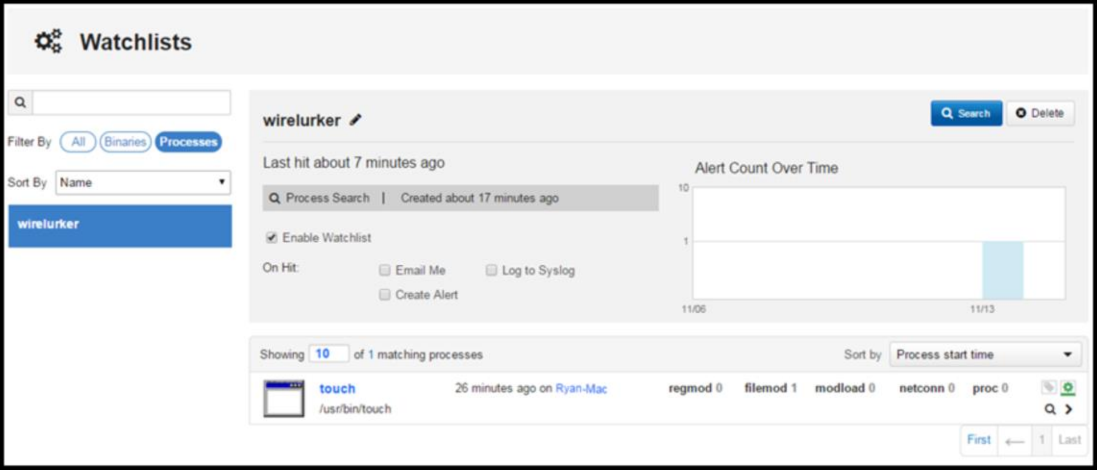
Watchlist 2:
This watchlist is looking for the known domain that WireLurker connects to. Currently, there is only one known domain. This is uncommon for malware these days but not unheard of. This watchlist is kept uniquely to network traffic only to cut down on editing later on. I have high confidence in this domain being malicious. However, domains change quickly, and I do not expect this watchlist to always give me a true positive result, nor do I expect it to be around for a long time. Therefore, I keep it separate and can easily disable it when I deem it no longer useful.

Watchlist 3:
This watchlist contains my low-confidence queries. These queries will contain false positives and I know that going into this. The reason they will fire false positives is

because of how broad they are. I have high confidence that anything under "/Applications/*/start.sh" will not be legitimate but I have not tested every software ever in every environment, so I leave room for false positives.

Also, the command for chflags to hidden is not an uncommon command. It is usually not used legitimately because it hides things from finder but not from command line.

Watchlist 1 Results



wirelurker

Last hit about 7 minutes ago

Process Search | Created about 17 minutes ago

Enable Watchlist: ✓

On Hit: Email Me ☐ Log to Syslog ☐ Create Alert ☐

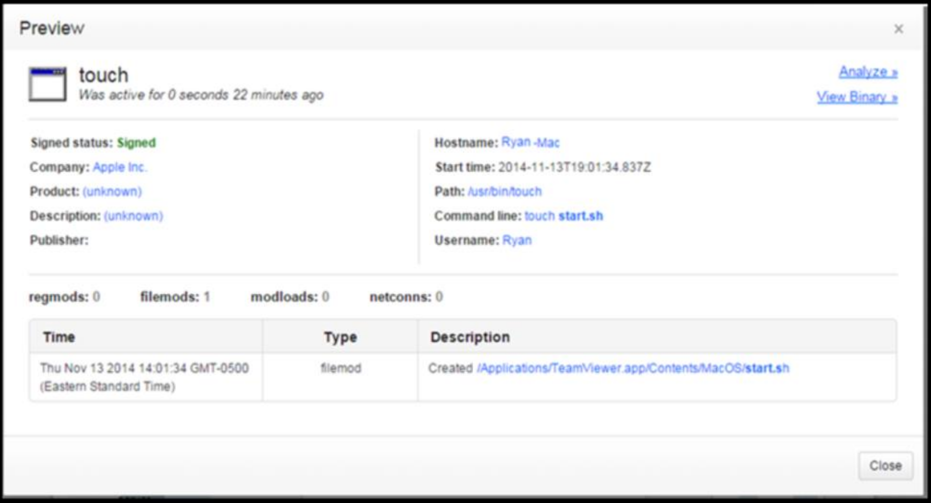
Showing 10 of 1 matching processes — Sort by Process start time

touch /usr/bin/touch	26 minutes ago on Ryan-Mac	regmod 0	filemod 1	modload 0	netconn 0	proc 0

First ← 1 Last

Above you can see an example of the watchlist I created for "filemod:Applications/*/start.sh." As you can see, when I set off the watchlist with the creation of start.sh in the file path of "/Applications/TeamViewer.app/Contents/MacOS/start.sh."

Drill down into Watchlist 1 results



Preview

touch
Was active for 0 seconds 22 minutes ago

Analyze »
View Binary »

Signed status: Signed
Company: Apple Inc.
Product: (unknown)
Description: (unknown)
Publisher:

Hostname: Ryan -Mac
Start time: 2014-11-13T19:01:34.837Z
Path: /usr/bin/touch
Command line: touch **start.sh**
Username: Ryan

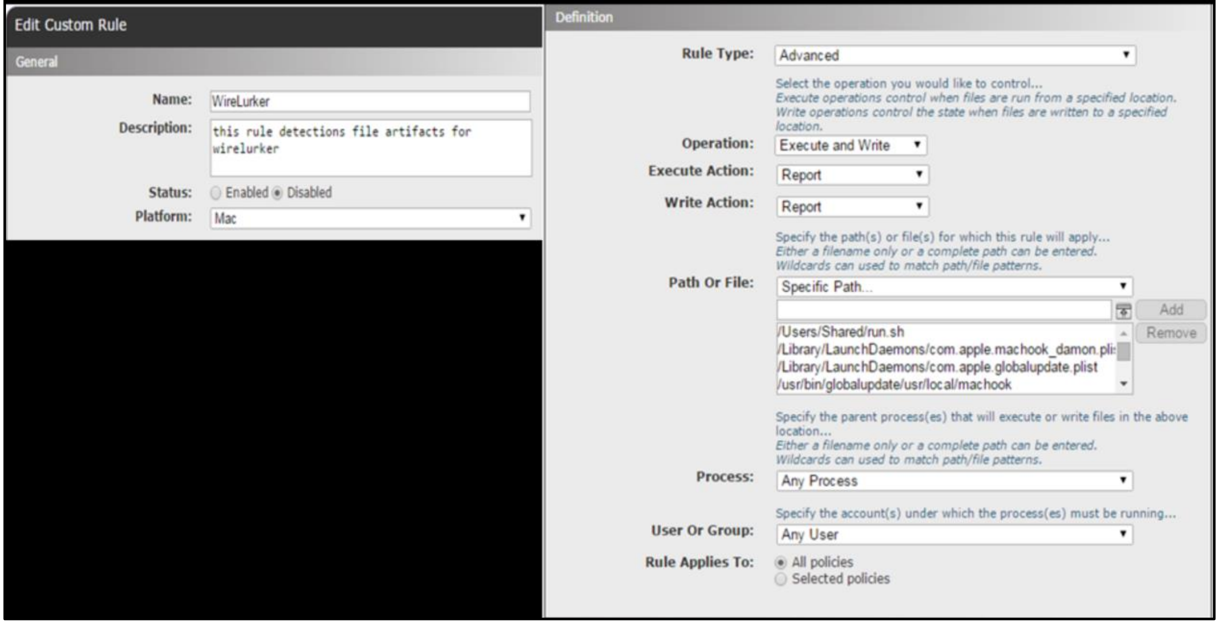
regmods: 0 **filemods:** 1 **modloads:** 0 **netconns:** 0

Time	Type	Description
Thu Nov 13 2014 14:01:34 GMT-0500 (Eastern Standard Time)	filemod	Created /Applications/TeamViewer.app/Contents/MacOS/**start.sh**

Close

Below, you can see the drill down of the command the script used to create this file (it used the touch command).

Example Bit9 block rules:



Edit Custom Rule

General

Name: WireLurker

Description: this rule detections file artifacts for wirelurker

Status: Enabled [✓ Disabled]

Platform: Mac

Definition

Rule Type: Advanced

Select the operation you would like to control...
Execute operations control when files are run from a specified location.
Write operations control the state when files are written to a specified location.

Operation: Execute and Write

Execute Action: Report

Write Action: Report

Specify the path(s) or file(s) for which this rule will apply...
Either a filename only or a complete path can be entered.
Wildcards can used to match path/file patterns.

Path Or File: Specific Path...

```
/Users/Shared/run.sh
/Library/LaunchDaemons/com.apple.machook_damon.pli
/Library/LaunchDaemons/com.apple.globalupdate.plist
/usr/bin/globalupdate/usr/local/machook
```

Specify the parent process(es) that will execute or write files in the above location...
Either a filename only or a complete path can be entered.
Wildcards can used to match path/file patterns.

Process: Any Process

Specify the account(s) under which the process(es) must be running...

User Or Group: Any User

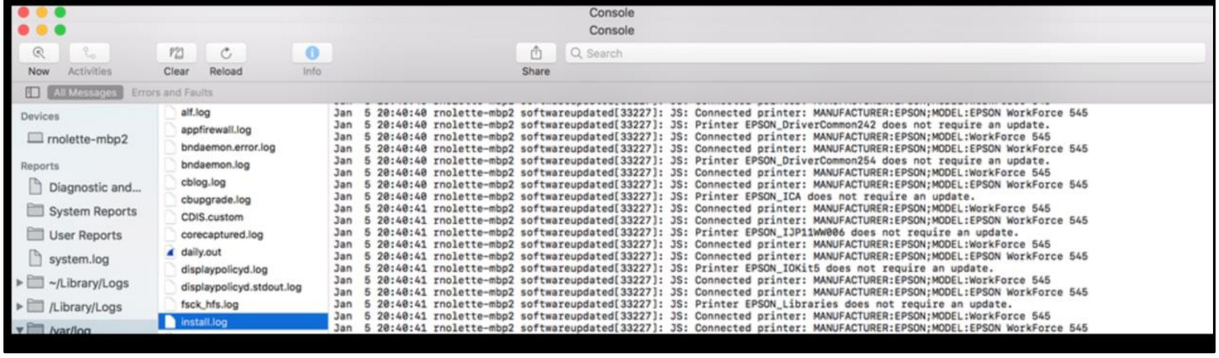
Rule Applies To: [✓ All policies] Selected policies

OSX

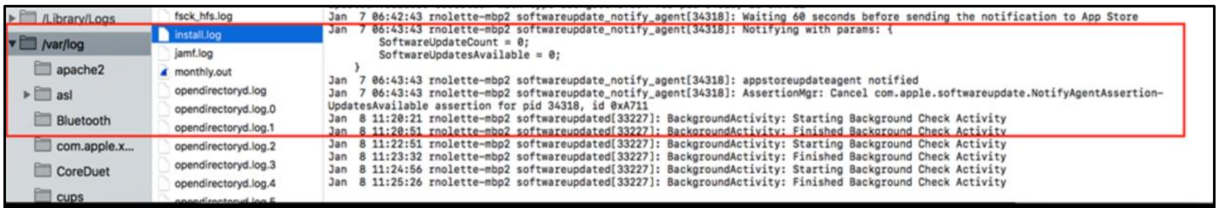
FORENSIC INVESTIGATION

Logging Basics

- General Log Location
  - System Logs
    - /private/var/log
    - /Library/Logs
  - User Logs
    - ~/Library/logs
- Application specific logs
  - /Library/Application Support/<applicationName>
  - /Applications/
  - /Library/Logs/
- macOS Sierra (10.12) introduced Unified Logging.
  - /var/db/diagnostics/
  - /var/db/uuidtext/



```
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:EPSON WorkForce 545
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Printer EPSON_DriverCommon242 does not require an update.
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:WorkForce 545
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:EPSON WorkForce 545
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Printer EPSON_DriverCommon254 does not require an update.
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:WorkForce 545
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:EPSON WorkForce 545
Jan  5 20:40:40 rnolette-mbp2 softwareupdated[33227]: JS: Printer EPSON_ICA does not require an update.
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:WorkForce 545
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:EPSON WorkForce 545
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Printer EPSON_IJP11WW006 does not require an update.
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:WorkForce 545
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:EPSON WorkForce 545
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Printer EPSON_IOKit5 does not require an update.
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:WorkForce 545
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:EPSON WorkForce 545
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Printer EPSON_Libraries does not require an update.
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:WorkForce 545
Jan  5 20:40:41 rnolette-mbp2 softwareupdated[33227]: JS: Connected printer: MANUFACTURER:EPSON;MODEL:EPSON WorkForce 545
```



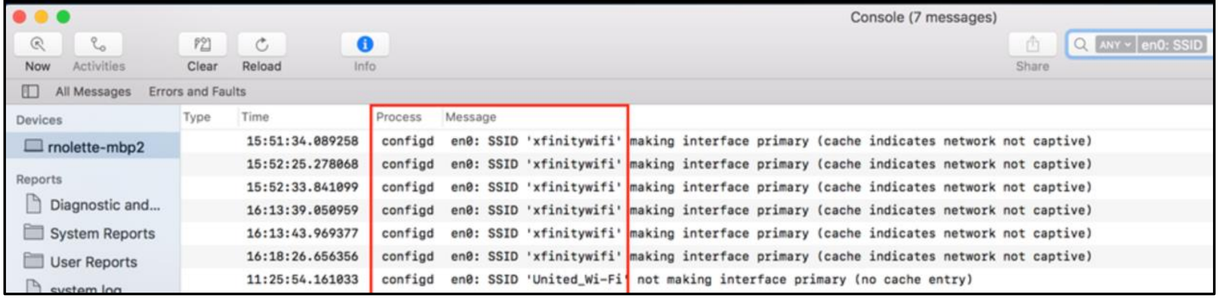
```
Jan  7 06:42:43 rnolette-mbp2 softwareupdate_notify_agent[34318]: Waiting 60 seconds before sending the notification to App Store
Jan  7 06:43:43 rnolette-mbp2 softwareupdate_notify_agent[34318]: Notifying with params: {
        SoftwareUpdateCount = 0;
        SoftwareUpdatesAvailable = 0;
    }
Jan  7 06:43:43 rnolette-mbp2 softwareupdate_notify_agent[34318]: appstoreupdateagent notified
Jan  7 06:43:43 rnolette-mbp2 softwareupdate_notify_agent[34318]: AssertionMgr: Cancel com.apple.softwareupdate.NotifyAgentAssertion-
UpdatesAvailable assertion for pid 34318, id 0xA711
Jan  8 11:20:21 rnolette-mbp2 softwareupdated[33227]: BackgroundActivity: Starting Background Check Activity
Jan  8 11:20:51 rnolette-mbp2 softwareupdated[33227]: BackgroundActivity: Finished Background Check Activity
Jan  8 11:22:51 rnolette-mbp2 softwareupdated[33227]: BackgroundActivity: Starting Background Check Activity
Jan  8 11:23:32 rnolette-mbp2 softwareupdated[33227]: BackgroundActivity: Finished Background Check Activity
Jan  8 11:24:56 rnolette-mbp2 softwareupdated[33227]: BackgroundActivity: Starting Background Check Activity
Jan  8 11:25:26 rnolette-mbp2 softwareupdated[33227]: BackgroundActivity: Finished Background Check Activity
```

- Most are plaintext
- Log turnover results in a BZip2 or Gzip archive
- Logs are set to roll at both time and size limitations
  - CIS script ups holding time to 90 days at a minimum and some are kept as long as a year.

- Timestamps
  - Uses standard Unix log formatting
    - MMM DD HH:MM:SS Host Service: Message
  - Apple system log
    - UTC
  - Most other logs (/var/log, ~/Library/logs
    - Local system time
  - ASL logs can use praudit to output to local system time
  - Temporarily change time zone of terminal window
    - Export TZ="EST5EDT"
  - Convert timestamps
    - EPOCH
    - Mac EPOCH

Network Information

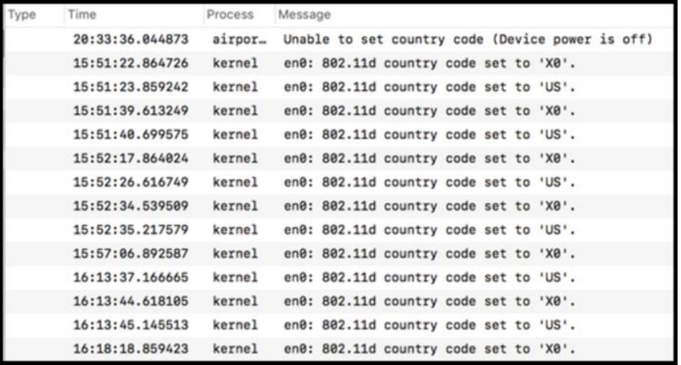
- Configuration
  - /Library/Preferences/systemconfiguration/preferences.plist
- DHCP addresses
  - /Private/var/db/dhcpclient/leases
- Network changes
  - Search for "configd"
    - System.log
- Wifi
  - /Library/preferences/systemconfiguration/com.apple.airport.preferences.plist
    - Last connected info
    - SSID names
  - Search "ariportd", "usereventagent", or "ssid"
    - system.log



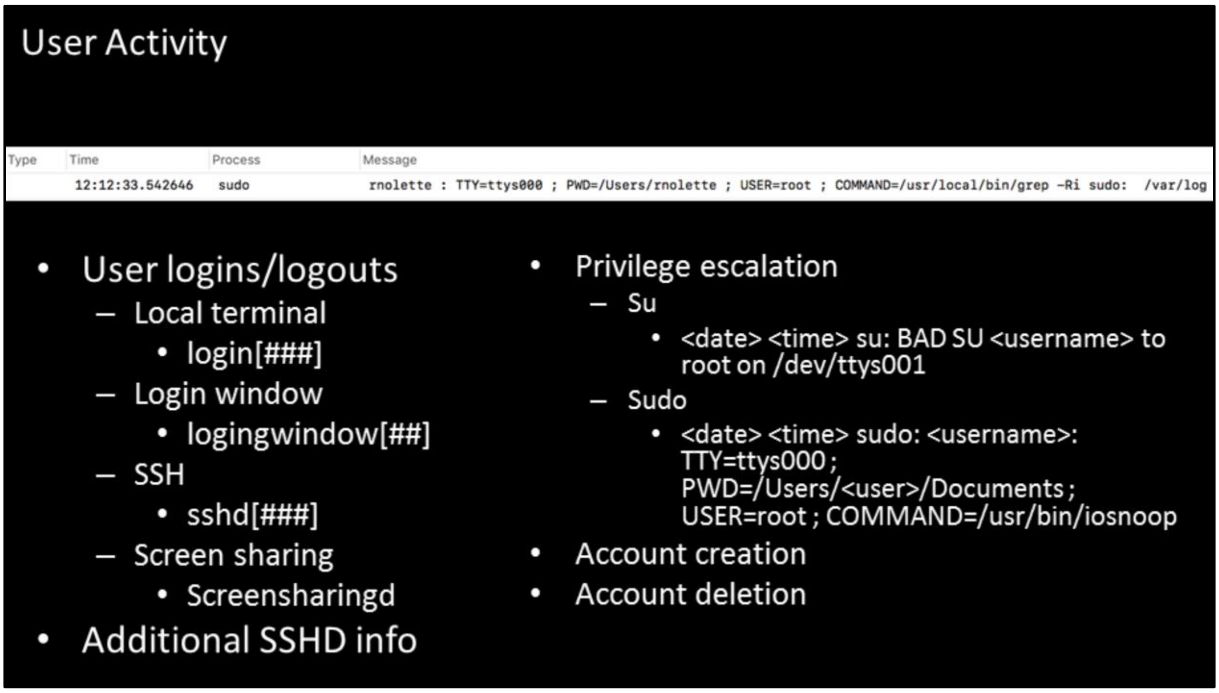
```
Type  Time              Process  Message
      15:51:34.089258   configd  en0: SSID 'xfinitywifi' making interface primary (cache indicates network not captive)
      15:52:25.278068   configd  en0: SSID 'xfinitywifi' making interface primary (cache indicates network not captive)
      15:52:33.841099   configd  en0: SSID 'xfinitywifi' making interface primary (cache indicates network not captive)
      16:13:39.050959   configd  en0: SSID 'xfinitywifi' making interface primary (cache indicates network not captive)
      16:13:43.969377   configd  en0: SSID 'xfinitywifi' making interface primary (cache indicates network not captive)
      16:18:26.656356   configd  en0: SSID 'xfinitywifi' making interface primary (cache indicates network not captive)
      11:25:54.161033   configd  en0: SSID 'United_Wi-Fi' not making interface primary (no cache entry)
```

Location Data

- Detailed timeline
  - Search “airportd” or “ssid”
    - System.log
  - /Library/preferences/systemconfiguration/com.apple.airport.preferences.plist
    - Can be used to determine the general location of SSID
    - Last connected time
    - Local system time
  - Search “country code”
    - Kernel.log
    - System.log



```
Type  Time             Process   Message
      20:33:36.044873  airpor…   Unable to set country code (Device power is off)
      15:51:22.864726  kernel    en0: 802.11d country code set to 'X0'.
      15:51:23.859242  kernel    en0: 802.11d country code set to 'US'.
      15:51:39.613249  kernel    en0: 802.11d country code set to 'X0'.
      15:51:40.699575  kernel    en0: 802.11d country code set to 'US'.
      15:52:17.864024  kernel    en0: 802.11d country code set to 'X0'.
      15:52:26.616749  kernel    en0: 802.11d country code set to 'US'.
      15:52:34.539509  kernel    en0: 802.11d country code set to 'X0'.
      15:52:35.217579  kernel    en0: 802.11d country code set to 'US'.
      15:57:06.892587  kernel    en0: 802.11d country code set to 'X0'.
      16:13:37.166665  kernel    en0: 802.11d country code set to 'US'.
      16:13:44.618105  kernel    en0: 802.11d country code set to 'X0'.
      16:13:45.145513  kernel    en0: 802.11d country code set to 'US'.
      16:18:18.859423  kernel    en0: 802.11d country code set to 'X0'.
```



User Activity

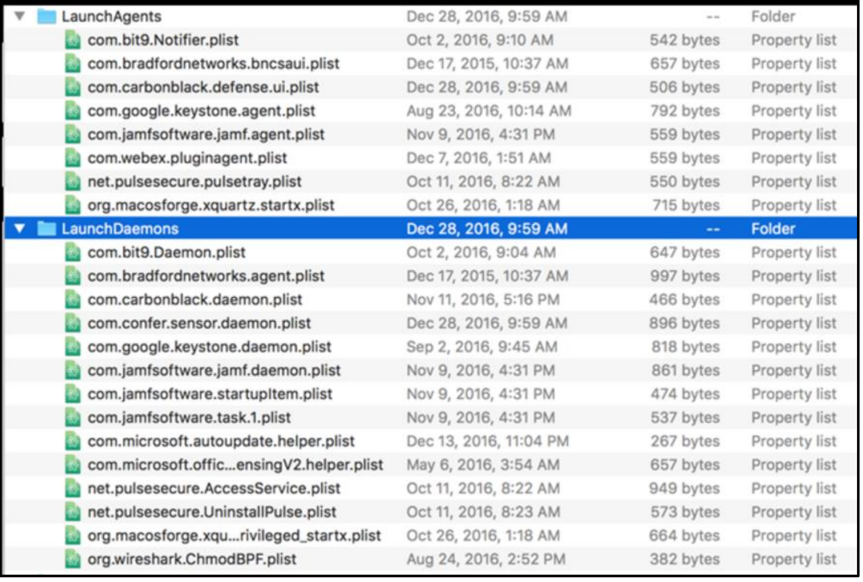
```
Type  Time             Process    Message
      12:12:33.542646  sudo       rnolette : TTY=ttys000 ; PWD=/Users/rnolette ; USER=root ; COMMAND=/usr/local/bin/grep -Ri sudo:  /var/log
```

- User logins/logouts
  - Local terminal
    - login[###]
  - Login window
    - logingwindow[##]
  - SSH
    - sshd[###]
  - Screen sharing
    - Screensharingd
- Additional SSHD info
- Privilege escalation
  - Su
    - <date> <time> su: BAD SU <username> to root on /dev/ttys001
  - Sudo
    - <date> <time> sudo: <username>: TTY=ttys000 ; PWD=/Users/<user>/Documents ; USER=root ; COMMAND=/usr/bin/iosnoop
- Account creation
- Account deletion

SYSTEM ARTIFACTS

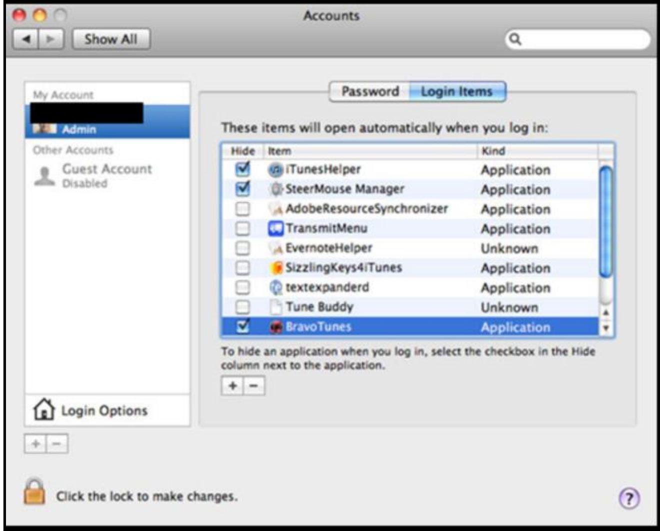
Autorun Locations

- Launch Agents files
- Launch Daemons files
- Startup Items file



▼ LaunchAgents	Dec 28, 2016, 9:59 AM	--	Folder
com.bit9.Notifier.plist	Oct 2, 2016, 9:10 AM	542 bytes	Property list
com.bradfordnetworks.bncsaui.plist	Dec 17, 2015, 10:37 AM	657 bytes	Property list
com.carbonblack.defense.ui.plist	Dec 28, 2016, 9:59 AM	506 bytes	Property list
com.google.keystone.agent.plist	Aug 23, 2016, 10:14 AM	792 bytes	Property list
com.jamfsoftware.jamf.agent.plist	Nov 9, 2016, 4:31 PM	559 bytes	Property list
com.webex.pluginagent.plist	Dec 7, 2016, 1:51 AM	559 bytes	Property list
net.pulsesecure.pulsetray.plist	Oct 11, 2016, 8:22 AM	550 bytes	Property list
org.macosforge.xquartz.startx.plist	Oct 26, 2016, 1:18 AM	715 bytes	Property list
▼ LaunchDaemons	Dec 28, 2016, 9:59 AM	--	Folder
com.bit9.Daemon.plist	Oct 2, 2016, 9:04 AM	647 bytes	Property list
com.bradfordnetworks.agent.plist	Dec 17, 2015, 10:37 AM	997 bytes	Property list
com.carbonblack.daemon.plist	Nov 11, 2016, 5:16 PM	466 bytes	Property list
com.confer.sensor.daemon.plist	Dec 28, 2016, 9:59 AM	896 bytes	Property list
com.google.keystone.daemon.plist	Sep 2, 2016, 9:45 AM	818 bytes	Property list
com.jamfsoftware.jamf.daemon.plist	Nov 9, 2016, 4:31 PM	861 bytes	Property list
com.jamfsoftware.startupItem.plist	Nov 9, 2016, 4:31 PM	474 bytes	Property list
com.jamfsoftware.task.1.plist	Nov 9, 2016, 4:31 PM	537 bytes	Property list
com.microsoft.autoupdate.helper.plist	Dec 13, 2016, 11:04 PM	267 bytes	Property list
com.microsoft.offic...ensingV2.helper.plist	May 6, 2016, 3:54 AM	657 bytes	Property list
net.pulsesecure.AccessService.plist	Oct 11, 2016, 8:22 AM	949 bytes	Property list
net.pulsesecure.UninstallPulse.plist	Oct 11, 2016, 8:23 AM	573 bytes	Property list
org.macosforge.xqu...rivileged_startx.plist	Oct 26, 2016, 1:18 AM	664 bytes	Property list
org.wireshark.ChmodBPF.plist	Aug 24, 2016, 2:52 PM	382 bytes	Property list

SYSTEM ARTIFACTS



[redacted]
Admin

Guest Account
Disabled

These items will open automatically when you log in:

Hide	Item	Kind
✓	iTunesHelper	Application
✓	SteerMouse Manager	Application
	AdobeResourceSynchronizer	Application
	TransmitMenu	Application
	EvernoteHelper	Unknown
	SizzlingKeys4iTunes	Application
	textexpanderd	Application
	Tune Buddy	Unknown
✓	BravoTunes	Application

To hide an application when you log in, select the checkbox in the Hide column next to the application.

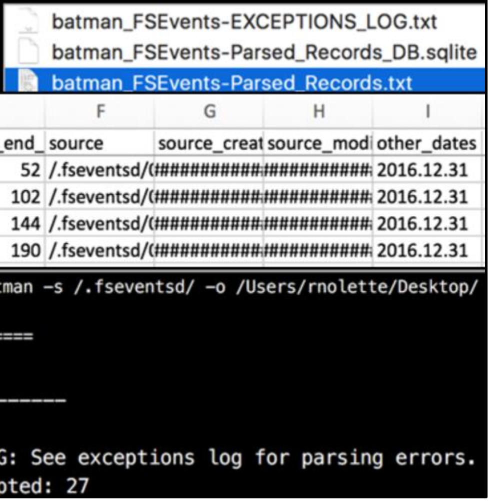
- **Autorun Locations**
- **Login Items**
  - Plists listing applications that automatically start when the user is logged in
  - %%users.homedir%%/Library/Preferences/com.apple.loginitems.plist

Quarantine Event Database
  - SQLite database that keeps track of files that have the quarantine extended attribute.
  - given to applications, scripts, and executables downloaded from potentially untrustworthy locations/people.
  - The SQLite database contains URLS, email addresses, email subjects, and other potentially useful information.
  - %%users.homedir%%/Library/Preferences/com.apple.LaunchServices.QuarantineEvents %%users.homedir%%/Library/Preferences/com.apple.LaunchServices.QuarantineEventsV2

Free Tools

FSEventsParser https://github.com/dlcowen/FSEventsParser/blob/master/README.md

FSEvents files are written to disk by OS X apis and contain historical records of events that occurred for the partition.



batman_FSEvents-EXCEPTIONS_LOG.txt
batman_FSEvents-Parsed_Records_DB.sqlite
batman_FSEvents-Parsed_Records.txt

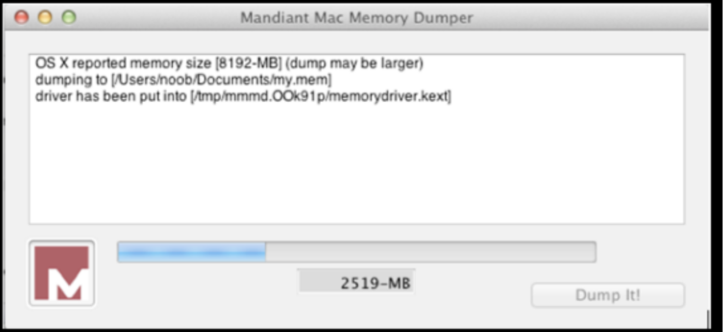
	A	B	C	D	E	F	G	H	I
1	wd	mask_hex	filename	mask	record_end_	source	source_creat	source_modi	other_dates
2	8284	0x00010001	.DocumentR	FolderEvent;	52	/.fseventsd/(	##########	##########	2016.12.31
3	11769	0x08038000	.DocumentR	Renamed;Pe	102	/.fseventsd/(	##########	##########	2016.12.31
4	8302	0x00010001	.DocumentR	FolderEvent;	144	/.fseventsd/(	##########	##########	2016.12.31
5	8290	0x00010001	.DocumentR	FolderEvent;	190	/.fseventsd/(	##########	##########	2016.12.31

```
rnolette-mbp2:FSEventsParser root# sudo python FSEParser_v2.0.py -c batman -s /.fseventsd/ -o /Users/rnolette/Desktop/

==========================================================================
FSEParser v 2.0  -- provided by G-C Partners, LLC
Run Time:  01/08/2017 18:35:58 [UTC]
====================================================
                                                     ------------------------

                                                     FINISHED PARSING: See exceptions log for parsing errors.
File 1 of 27:   Trying  000000000000e1a6             All Files Attempted: 27
```

Free Tools

MEMORYZE FOR THE MAC

HTTPS://WWW.FIREEYE.COM/SERVICES/FREEWARE/MEMORYZE-FOR-THE-MAC.HTML



```
OS X reported memory size [8192-MB] (dump may be larger)
dumping to [/Users/noob/Documents/my.mem]
driver has been put into [/tmp/mmmd.OOk91p/memorydriver.kext]
```

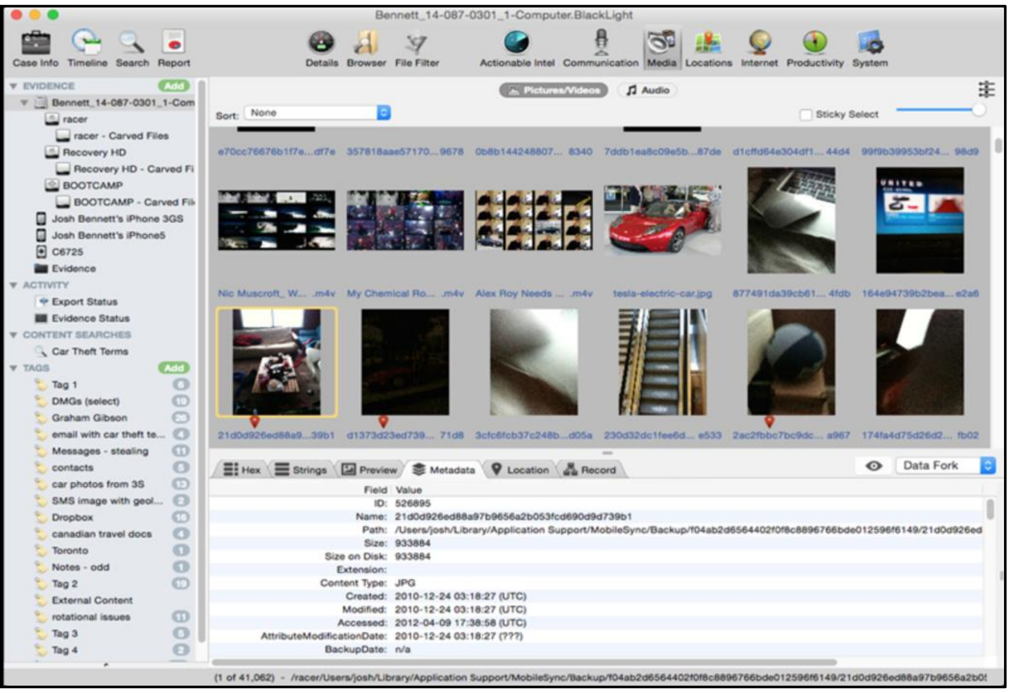
Known limitations:

- Memoryze for the Mac 1.1 Officially Supports:
  - Mac OS X Snow Leopard (10.6) 32/64-bit
  - Mac OS X Lion (10.7) 32/64-bit
  - Mac OS X Mountain Lion (10.8) 64-bit
- Acquire Memory Image via GUI above or the following CLI command
  - sudo macmemoryze dump -f my.mem
  - This will acquire memory to a file named "my.mem" in the local directory.

Tools

BlackLight

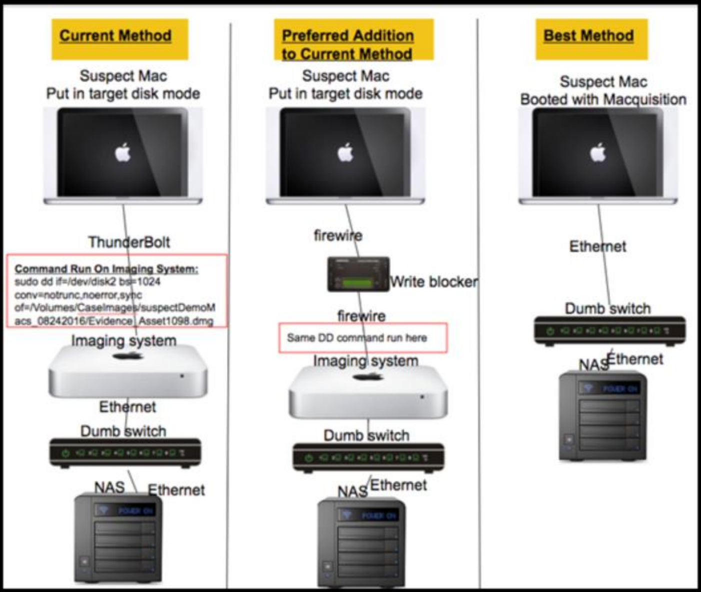
https://www.blackbagtech.com/software-products/blacklight.html



Field	Value
ID:	526895
Name:	21d0d926ed88a97b9656a2b053fcd690d9d739b1
Path:	/Users/josh/Library/Application Support/MobileSync/Backup/f04ab2d6564402f0f8c8896766bde012596f6149/21d0d926ed
Size:	933884
Size on Disk:	933884
Extension:	
Content Type:	JPG
Created:	2010-12-24 03:18:27 (UTC)
Modified:	2010-12-24 03:18:27 (UTC)
Accessed:	2012-04-09 17:38:58 (UTC)
AttributeModificationDate:	2010-12-24 03:18:27 (???)
BackupDate:	n/a

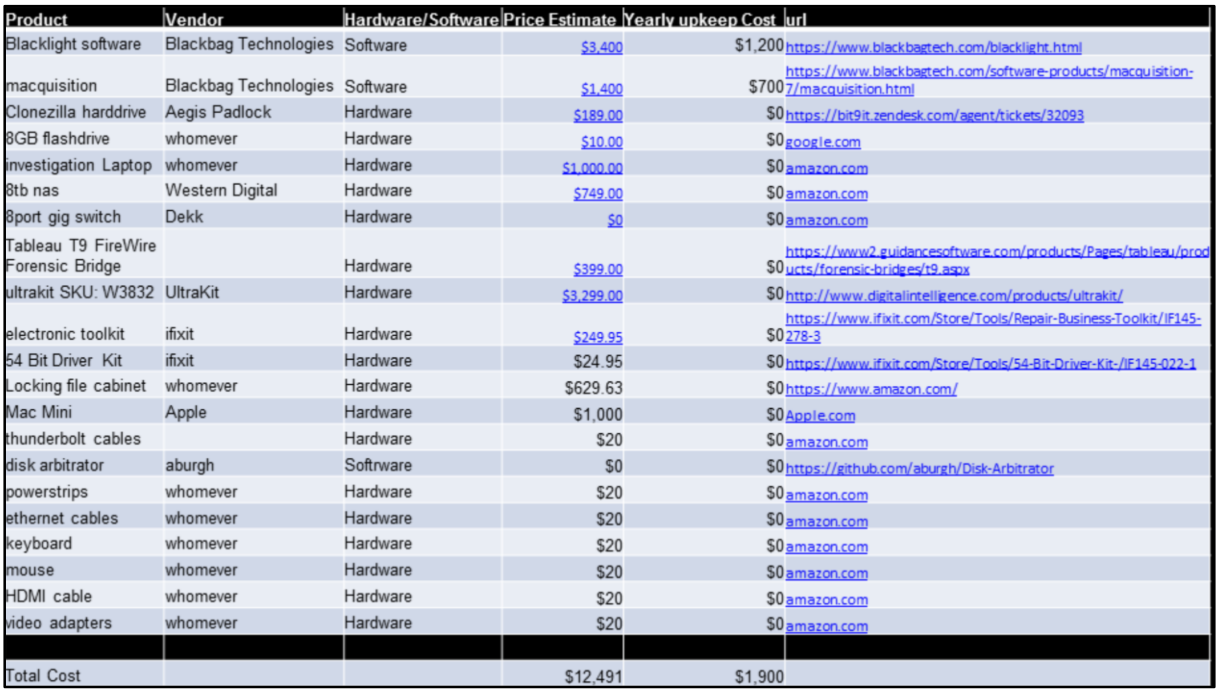
(1 of 41,062) - /racer/Users/josh/Library/Application Support/MobileSync/Backup/f04ab2d6564402f0f8c8896766bde012596f6149/21d0d926ed88a97b9656a2b05

Imaging Topology



Command Run On Imaging System:

```
sudo dd if=/dev/disk2 bs=1024
conv=notrunc,noerror,sync
of=/Volumes/CaseImages/suspectDemoM
acs_08242016/Evidence_Asset1098.dmg
```



Product	Vendor	Hardware/Software	Price Estimate	Yearly upkeep Cost	url
Blacklight software	Blackbag Technologies	Software	$3,400	$1,200	https://www.blackbagtech.com/blacklight.html
macquisition	Blackbag Technologies	Software	$1,400	$700	https://www.blackbagtech.com/software-products/macquisition-7/macquisition.html
Clonezilla harddrive	Aegis Padlock	Hardware	$189.00	$0	https://bit9it.zendesk.com/agent/tickets/32093
8GB flashdrive	whomever	Hardware	$10.00	$0	google.com
investigation Laptop	whomever	Hardware	$1,000.00	$0	amazon.com
8tb nas	Western Digital	Hardware	$749.00	$0	amazon.com
8port gig switch	Dekk	Hardware	$0	$0	amazon.com
Tableau T9 FireWire Forensic Bridge		Hardware	$399.00	$0	https://www2.guidancesoftware.com/products/Pages/tableau/products/forensic-bridges/t9.aspx
ultrakit SKU: W3832	UltraKit	Hardware	$3,299.00	$0	http://www.digitalintelligence.com/products/ultrakit/
electronic toolkit	ifixit	Hardware	$249.95	$0	https://www.ifixit.com/Store/Tools/Repair-Business-Toolkit/IF145-278-3
54 Bit Driver Kit	ifixit	Hardware	$24.95	$0	https://www.ifixit.com/Store/Tools/54-Bit-Driver-Kit-/IF145-022-1
Locking file cabinet	whomever	Hardware	$629.63	$0	https://www.amazon.com/
Mac Mini	Apple	Hardware	$1,000	$0	Apple.com
thunderbolt cables		Hardware	$20	$0	amazon.com
disk arbitrator	aburgh	Softrware	$0	$0	https://github.com/aburgh/Disk-Arbitrator
powerstrips	whomever	Hardware	$20	$0	amazon.com
ethernet cables	whomever	Hardware	$20	$0	amazon.com
keyboard	whomever	Hardware	$20	$0	amazon.com
mouse	whomever	Hardware	$20	$0	amazon.com
HDMI cable	whomever	Hardware	$20	$0	amazon.com
video adapters	whomever	Hardware	$20	$0	amazon.com
Total Cost			$12,491	$1,900	

Questions