Macs in the Age of APT

Tom Daniels, Aaron Grattaflori, BJ Orvis, Alex Stamos, Paul Youn

iSEC Partners

Black Hat USA 2011
Agenda

1 Motivation
   - Preface and Background

2 Anatomy of an APT
   - Social Engineering
   - Initial Exploitation
   - Local Privilege Escalation
   - Network Privilege Escalation
   - Persistence
   - Exploration
   - Exfiltration

3 Conclusion
   - Summary
Outline

1 Motivation
   - Preface and Background

2 Anatomy of an APT
   - Social Engineering
   - Initial Exploitation
   - Local Privilege Escalation
   - Network Privilege Escalation
   - Persistence
   - Exploration
   - Exfiltration

3 Conclusion
   - Summary
What is APT?
Apple Purchases Tacos?

- **Advanced**: not your average Joe, may be government funded, may have zero-day vulnerabilities.
- **Persistent**: initial access leads to the creation of many access methods and long-term exploration
- **Threat**: defines the group of attackers with these capabilities, not an actual attack scenario
Case Study: Aurora

What the what?

- Originally disclosed by Google on January 12th 2010
- Google discovered evidence of >30 other victims
- Attack was focused on Windows exploitation and escalation in AD
- Estimates range from dozens to hundreds of companies attacked\(^1\)
  - Google
  - DuPont
  - Adobe
  - Juniper Networks
  - Northrop Grumman
  - Sony
  - And many more

Case Study: Aurora
Socially engineer a victim to click on a malicious link
Case Study: Aurora
Socially engineer a victim to click on a malicious link
Case Study: Aurora

Escalate network privileges
Case Study: Aurora

Make your attack more persistent
Case Study: Aurora

Explore using the cracked domain credentials

Daniels, Grattafori, Orvis, Stamos, Youn (iSEC)

Macs in the Age of APT
Case Study: Aurora

Exfiltrate the data
Outline

1 Motivation
   - Preface and Background

2 Anatomy of an APT
   - Social Engineering
     - Initial Exploitation
     - Local Privilege Escalation
     - Network Privilege Escalation
     - Persistence
     - Exploration
     - Exfiltration

3 Conclusion
   - Summary
Your Mac is Safer

- Apple has a small computer market share (6-8%)\(^2\)
- Building a bot-net? Go for Windows users
- There are fewer viruses and malware applications for Mac
  - No exploits included in common crimeware toolkits targeting Macs\(^3\)
  - Attacks focus on social engineering (such as Mac Defender)

\(^3\)See iSEC consultant Dan Guido’s research
Training Mac Users to Feel Safe

- A history of non-exploitation
- Go ahead, run this unsigned binary
- Who needs anti-virus?⁴

---

⁴http://news.cnet.com/8301-27080_3-10444561-245.html
Apple Marketing is Misleading
Sort of like all marketing (unrelated: hire iSEC because we are the best at everything)

- “OS X doesn’t get PC viruses”
- Other things OS X can’t catch:
  - A Nintendo Wii virus
  - Mad cow disease, malaria, or chickenpox
  - Footballs (we tried)
- OS X is still vulnerable to malware (like almost any computer system)

Mac Users are Susceptible to Social Engineering

- Mac users aren’t as paranoid as Windows users\(^5\)

\[\text{https://discussions.apple.com/message/15242642#15242642}\]

- Mac Defender
- Mac users may be easy to socially engineer

\(^5\)Daniels, Grattaflori, Orvis, Stamos, Youn (iSEC)
OS X isn’t More Secure

- 14.3% of publicly disclosed OS vulnerabilities affected OS X in 2008\(^6\)

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple Mac OS X Server</td>
<td>14.3%</td>
</tr>
<tr>
<td>Apple Mac OS X</td>
<td>14.3%</td>
</tr>
<tr>
<td>Linux Kernel</td>
<td>10.9%</td>
</tr>
<tr>
<td>Sun Solaris</td>
<td>7.3%</td>
</tr>
<tr>
<td>Microsoft Windows XP</td>
<td>5.5%</td>
</tr>
</tbody>
</table>

- Latest OS X security patch addressed 39 CVEs
- 1,151 CVEs reported in the last 3 years affect Apple (including third-party software)
- Similar number of Windows CVEs (1,325)
- Safety in numbers

Targeted attackers don’t care what OS a corporation is running
Mac users may be more vulnerable Social Engineering
Plenty of vulnerabilities lead to “Initial Exploitation”
Outline

1. Motivation
   - Preface and Background

2. Anatomy of an APT
   - Social Engineering
   - Initial Exploitation
   - Local Privilege Escalation
   - Network Privilege Escalation
   - Persistence
   - Exploration
   - Exfiltration

3. Conclusion
   - Summary
Exploitation in APT

- Get user to click a link
- And then exploit...
  - Railroad user into an installer with Safari’s safe files
  - Browser or plugin exploit
Safari’s open “safe” files includes installers

- .pkg and .mpkg files
- A .zip containing a .pkg runs Installer.app (Fixed in Safari 5.1)
- User must click through
- MACDefender⁷ and variants triggered a “4-5x higher than normal” call volume with AppleCare when it hit⁸

---

⁷http://blog.intego.com/2011/05/02/macdefender-rogue-anti-malware-program-attacks-macs-via-seo-poisoning/

File Quarantine and XProtect

File Quarantine
- Part of the LaunchServices API
- Quarantine properties dictionary
- `const CFStringRef kLSItemQuarantineProperties`

XProtect
- Signature-based scanner
- Piggy-backs on File Quarantine
  - Downloaded files marked with extended attribute
  - LaunchServices triggers scan
- In its infancy on Mac OS X (introduced in 10.6)
- Security Update 2011-003: Malware database now updates daily

---

Anti-exploit Mitigations

Mitigation availability:

<table>
<thead>
<tr>
<th>Mitigation</th>
<th>Windows</th>
<th>Mac OS X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack Protections</td>
<td>2003 (Visual Studio’s /GS)</td>
<td>2007 (10.5/XCode 3.1)</td>
</tr>
<tr>
<td></td>
<td>2003 (XP SP2)</td>
<td>2009 (10.6)</td>
</tr>
<tr>
<td>Heap Protections</td>
<td>2004 (XP SP 2)</td>
<td>2006 (10.4.4 Intel)</td>
</tr>
<tr>
<td></td>
<td>2007 (Vista)</td>
<td>2007 (10.5)</td>
</tr>
<tr>
<td>DEP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASLR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Smash the Stack

- GCC ProPolice can be used at compile-time (GCC $\geq 4.1$)
- GCC’s -D_FORTIFY_SOURCE in 10.6
- 10.5/XCode 3.1: GCC 4.2 first included, but not the default (GCC 4.0)
- 10.6/XCode 3.2: GCC 4.2 the default, -fstack-protector enabled by default
- Binaries built using older toolchain may not have it enabled
Break the Heap

- **Mac OS X**
  - 10.5: checksum — not a security protection
  - 10.6: Include a security cookie — better\(^{11}\)

- **Windows**
  - XP SP2 and Server 2003\(^{12}\): Safe unlinking and heap entry header cookie
  - Vista and later: Numerous additional heap protections

---

\(^{11}\)http://securityevaluators.com/files/papers/SnowLeopard.pdf

NX/DEP/ED

- Supported on Intel architectures
- Sets the default mprotect() exec flag for heap and stack
  - 10.6: heap always executable for 32-bit binaries
    - not even mprotect() can disable
  - 10.7: 32-bit binaries compiled on 10.6 still have always-executable heaps
- Not configurable

<table>
<thead>
<tr>
<th></th>
<th>10.4 i386</th>
<th>10.5 i386</th>
<th>10.5 x86_64</th>
<th>10.6 i386</th>
<th>10.6 x86_64</th>
<th>10.7 i386</th>
<th>10.7 x86_64</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Heap</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Daniels, Grattafori, Orvis, Stamos, Youn (iSEC)
ASLR

- 10.5: First introduced
- 10.6: No major changes
  - Not all libs use it
  - Not application code
  - Not the stack or heap
  - ROP exploits possible using dyld\(^{13}\)
- 10.7: Greatly improved\(^{14}\)
- Not configurable

\(^{13}\)http://securityevaluators.com/files/papers/SnowLeopard.pdf

\(^{14}\)http://www.apple.com/macosx/whats-new/features.html#security
- Been behind Microsoft, but finally catching up
- DEP and ASLR are not configurable
- Backwards compatibility threats
Outline

1 Motivation
   - Preface and Background

2 Anatomy of an APT
   - Social Engineering
   - Initial Exploitation
   - Local Privilege Escalation
   - Network Privilege Escalation
   - Persistence
   - Exploration
   - Exfiltration

3 Conclusion
   - Summary
Accessing Patient Zero’s Data
Information stored on disc

- Locally stored E-mail
- Safari History, Bookmarks
- iChat logs
- Spotlight DBs
Escalating Privilege
Attacking the login keychain

- Code execution doesn’t mean full account access
- The “Login Keychain” can be used to brute-force the user’s password
Escalating Privilege
Sudo make me a sandwich\textsuperscript{15}

- If a user is a sudoer, password can directly escalate privilege
- User password can be used to decrypt the “Login Keychain”
- Privileged credentials in the keychain can be used to spread and explore

\textsuperscript{15}http://xkcd.com/149/
Escalating Privilege

Phishing for admin

- OS X requires authorization for privileged action:

- Windows UAC screen slightly harder to spoof
Escalating Privilege
Phishing for admin

- This application sends admin credentials offsite in an HTTP “GET”

```
GET /paul/Usernameis/isecadmin/Password/p@ssw0rd HTTP/1.1
```

- UAC can be spoofed on Windows as well
Lion Improvements
AppSandbox: a safer place to play

- Subscription-based via plist

```xml
<key>com.apple.security.app-sandbox</key>
<true/>
```

- Per application container

```
export $HOME=~/Library/Containers/app.bundle.id/Data
```

- Per session entitlements

- Powerbox (pboxd)
  - sandbox-free broker process
  - transparent to developers (NSOpenPanel/NSSavePanel)
Lion Improvements

AppSandbox: cool kids use least privileges

- **Entitlements**
  - com.apple.security.documents.user-selected
  - com.apple.security.assets
  - com.apple.security.network
  - com.apple.security.personal-information
  - com.apple.security.device

- **Temporary Exceptions**
  - $HOME/absolute file access
  - Send Apple Events
  - Look up mach services
  - Inherit
Lion Improvements
XPC: Intra-application privilege separation

- libSystem IPC API
- XPC binaries stored in Bundle.app/Contents/XPC
  - Address space isolation
  - Fully restricted sandbox by default
  - Elevating XPC service to root is unsupported
- On-demand launching
  - integration with GCD and launchd
- Quicktime Player uses a low-privileged process called VTDecoderXPCService¹⁶

Back to APT
What can the local user do

- Access valuable local data
- Brute-force a valuable credential store
- Phish for admin credentials
- Help is on the way?
Outline

1 Motivation
- Preface and Background

2 Anatomy of an APT
- Social Engineering
- Initial Exploitation
- Local Privilege Escalation
- Network Privilege Escalation
- Persistence
- Exploration
- Exfiltration

3 Conclusion
- Summary
Lots of Services Makes Us Enterprise, Right?

- Presented at SOURCE Seattle and ToorCon
- Examined security of network administration protocols in Snow Leopard Server (10.6)
  - 28 network ports open after default install!!!
- Found pervasive authentication issues
- Exploited two of the most widely used protocols for managing Macs
AFP Authentication
You are the Weakest Link, goodbye!

- AFP provides multiple user authentication modules (UAM)
- Clients supporting weaker UAMs -> degradation attack

<table>
<thead>
<tr>
<th>Authentication Mechanisms</th>
<th>Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kerberos</td>
<td>Offline brute force attacks, relay attacks</td>
</tr>
<tr>
<td>DHX2 Cast 128 Version 2 (DHX2)</td>
<td>Active network attacker</td>
</tr>
<tr>
<td>DH Cast 128</td>
<td>Active network attacker</td>
</tr>
<tr>
<td>Two way random</td>
<td>Crack DES</td>
</tr>
<tr>
<td>Random number exchange</td>
<td>Crack DES, No server auth</td>
</tr>
<tr>
<td>Clear text password</td>
<td>Passive network attacker</td>
</tr>
<tr>
<td>No user authentication</td>
<td>None needed</td>
</tr>
</tbody>
</table>
Bonjoof

- Apple Remote Desktop
  - Uses 512-bit prime for (anonymous) Diffie Hellman key agreement
  - Creates a shared AES-128 key for UDP transmission
  - Authenticates over the established encrypted channel

- Bonjour
  - ad-hoc DNS service
  - No authentication
  - Requires peers to back off if a desired name is taken

- Combine the two...
  - Weak server auth + Untrusted identification -> Bonjoof
Bonjoof Beta
File server offering ARD services
Bonjoof Beta
Administrator enjoys his coffee
Bonjoof Beta
Spoofing mDNS
Bonjoof Beta
Claiming the hostname
Bonjoof Beta
ARD client silently updates its stats (auto-login)
Bonjoof Beta
Reset the file server’s hostname

Using Server Admin, I can change the victim server’s name back.

The administrator wants me to change my name to FileServer.

Authenticated ServerMgr Request to change the name

Flush your cache!

Attacker

Client
Bonjoof Beta
Where’d who go?

My name is FileServer
I am running ARD

ARD server
Bonjoof Beta

Some sample tool output

```
Bonjoof Server listening on port 3283

Received CLIENT_HELLO from 192.168.1.102

Received DHX request from 192.168.1.102:3283

The negotiated AES128 key is: 0b0ba2c1fe0416434abd826db682fad5

Received credentials:
   Username: isecadmin
   Password: p@ssw0rd
```
No standardized authentication mechanism/configuration
AFP, OpenDirectory, ServerAdmin all suffer from authentication issues
Bonjour makes local DNS poisoning easy...no race condition required
Outline

1 Motivation
   - Preface and Background

2 Anatomy of an APT
   - Social Engineering
   - Initial Exploitation
   - Local Privilege Escalation
   - Network Privilege Escalation
   - Persistence
   - Exploration
   - Exfiltration

3 Conclusion
   - Summary
Maintaining Access
how to survive the reboot

- Create a hidden startup item
- Com.apple.SystemLoginItems.plist Exploit
- Append to existing user startup scripts
- Hidden cronjob or automator script
- Modify existing binaries and services, which breaks signing but is generally not noticed
- Modify kernel extensions or cached extensions
- Persist in firmware

Maintaining Access
Attacking and hiding

- Execute arbitrary shell commands
- Run JavaScript in Safari to manipulate/create webpages in Safari
- Attach folder actions to hide data
- Send file transfer messages to your iChat contacts (may be Adium only)
Maintaining Access
At the network layer

- Issue VPN credentials to maintain foothold
- Issue soft tokens from access server
- Issue certificates
- Create new AD users
The Persistent Attack
Userland rootkits: a history...

- Nemo recreates PTRACE functionality and does great Mach ports research\(^7\)
- Dino publicly releases remotely controllable PoC Mach proxy rootkit\(^8\)
- Jonathan Rentzsch creates tools and uses them for “hooking” and “swizziling”: methods of modifying existing binaries in memory or on disc
- Dino and Miller write “Mac Hacker’s Handbook” with excellent illustrative examples of persistent attacks using these techniques\(^9\)
- More followed

\(^{18}\)nemo, Abusing Mach on Mac OS X. May 2006. 
http://www.uninformed.org/?v=4&a=3&t=pdf

\(^{19}\)http://trailofbits.files.wordpress.com/2009/08/advancedmacosxrootkits.pdf

Fighting Persistence

Mac IR

- How do we handle IR on Macs?
- Commercial Products
  - EnCase, BlackLight, FTK
  - All handle standard HFS+ forensics
  - Some claim file hash checking (and fail)
- What’s missing?
  - Easy checking of OS integrity
  - Binary and driver signing
  - Memory forensics\(^{21}\)
- Is all of the system state captured on the HDD?

\(^{21}\)Volatility [https://www.volatilesystems.com/default/volatility](https://www.volatilesystems.com/default/volatility) is working on it
Dealing with APT

Mac Hardware Forensics
Outline

1. Motivation
   • Preface and Background

2. Anatomy of an APT
   • Social Engineering
   • Initial Exploitation
   • Local Privilege Escalation
   • Network Privilege Escalation
   • Persistence
   • Exploration
   • Exfiltration

3. Conclusion
   • Summary
Who do you Love?
Are you for sure?

- Pick accounts to attack by examining the Open Directory users, groups, and privileges using unauthenticated ldapsearch
  - Engineers: source code
  - Product Management: release information
  - CFO’s office, Controller: Financial data
  - In house counsel: Lawful intercept access

- Account home directories network mounted by default
Accessing Interesting Accounts

- Root users on Open Directory server can get the password directory (mkpassdb)
- Domain administrators can change user passwords to access accounts
- Administrators in Windows can do bad things too
Making Exploration Harder

- Don’t allow server admin accounts to have root access
- Use strong password hash formats
- Regularly review audit logs and set up alerts to track password changes and VPN enrollment
Outline

1. Motivation
   - Preface and Background

2. Anatomy of an APT
   - Social Engineering
   - Initial Exploitation
   - Local Privilege Escalation
   - Network Privilege Escalation
   - Persistence
   - Exploration
   - Exfiltration

3. Conclusion
   - Summary
The Getaway

- Shawshank-style
  - Identify overseas internal drop server
  - Move data over corporate WAN to internal drop
  - Test for allowed outbound protocols
  - Bulk exfiltration though local office NAT to external drop server

- Covert Channels
  - ICMP
  - HTTPS

- Hide in plain sight\textsuperscript{22}

- PKI via embedded public keys

\textsuperscript{22}http://invisiblethings.org/papers/passive-covert-channels-linux.pdf
How can we mitigate the exfiltration threat?

Short term

- Coordinated egress restrictions in *all* offices
- DLP & proxy log monitoring
- 24x7 SOC ninjas
How can we mitigate the exfiltration threat?

Long term

- Time to rethink global architecture
  - Leased lines
  - Unified Forest
  - L3 routing directly between offices

- Alternatives
  - ADFS Federated domains
  - WAN accelerators
  - Limited, audited file sync
Dealing with APT
Comparison with Windows

- In each phase of an APT, how does OS X stack up?
- Assumptions:
  - Windows 7 and 2008R2
  - OS 10.7 Client and Server
  - No mixed environments
Windows vs Mac Comparison

Initial Exploitation:

<table>
<thead>
<tr>
<th>Windows 7</th>
<th>OS 10.7 Lion</th>
<th>Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack Canary</td>
<td>Stack Canary</td>
<td>Tie</td>
</tr>
<tr>
<td>Heap Hardening</td>
<td>Heap Hardening</td>
<td>?</td>
</tr>
<tr>
<td>Heap and Stack DEP</td>
<td>Heap and Stack NX</td>
<td>Tie</td>
</tr>
<tr>
<td>ASLR (32 and 64 bit)</td>
<td>ASLR (32 and 64 bit)</td>
<td>Tie</td>
</tr>
<tr>
<td>Configurable with EMET</td>
<td>Not configurable</td>
<td>Windows</td>
</tr>
</tbody>
</table>

**Conclusion:** OS X has now equalized anti-exploit technologies with Windows.
## Windows vs Mac Comparison

### Local Privilege Escalation:

<table>
<thead>
<tr>
<th>Windows 7</th>
<th>OS 10.7 Lion</th>
<th>Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>NT Priv Dropping</td>
<td>Broker service and XPC</td>
<td>OS X</td>
</tr>
<tr>
<td>Default all privs</td>
<td>New default sandbox</td>
<td>OS X</td>
</tr>
<tr>
<td>UIPI and Secure Desk</td>
<td>Pop-up cred box</td>
<td>Windows</td>
</tr>
<tr>
<td>No default cred store</td>
<td>Login Keychain</td>
<td>Windows</td>
</tr>
</tbody>
</table>

**Conclusion:** Local privilege escalation on both platforms is still quite possible. Everybody loses.
Windows vs Mac Comparison

Network Privilege Escalation:

<table>
<thead>
<tr>
<th>Windows 2008R2</th>
<th>OS 10.7 Server</th>
<th>Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTLMv2</td>
<td>Unsigned DH</td>
<td>Windows</td>
</tr>
<tr>
<td>Kerberos Only Option</td>
<td>Lots of fallback to DH</td>
<td>Windows</td>
</tr>
<tr>
<td>RPC Privacy and Integrity</td>
<td>No central protocol crypto</td>
<td>Windows</td>
</tr>
<tr>
<td>RDP with session security</td>
<td>Apple Remote Desktop</td>
<td>Windows</td>
</tr>
<tr>
<td>AD DNS with Secure Updates</td>
<td>mDNS</td>
<td>Windows</td>
</tr>
</tbody>
</table>

**Conclusion:** OS X networks are significantly more vulnerable to network privilege escalation. Almost every OS X Server service offers weak or broken authentication methods.
Windows vs Mac Comparison

Persistence:

<table>
<thead>
<tr>
<th>Windows 7</th>
<th>OS 10.7 Lion</th>
<th>Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>User-Mode Services</td>
<td>User-Mode Services</td>
<td>Tie</td>
</tr>
<tr>
<td>Kernel Rootkits</td>
<td>Kernel Rootkits</td>
<td>Tie</td>
</tr>
<tr>
<td>Many disk forensics options</td>
<td>Fewer disk forensics</td>
<td>Windows</td>
</tr>
<tr>
<td>Several RAM forensics tools</td>
<td>Almost no RAM forensics</td>
<td>Windows</td>
</tr>
</tbody>
</table>

**Conclusion:** Persisting malicious code on both platforms is not a problem for APT. Defenders have more options to detect modification of Windows and analyze code, but this need should be slowly met by open-source and commercial tools.
Windows vs Mac Comparison

Exploration and Exfiltration:

<table>
<thead>
<tr>
<th>Windows 2008R2</th>
<th>OS 10.7 Server</th>
<th>Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD LDAP locked to unauthed users</td>
<td>Anonymous LDAP browsing</td>
<td>Windows</td>
</tr>
<tr>
<td>Configurable outbound FW</td>
<td>No outbound rules</td>
<td>Windows</td>
</tr>
<tr>
<td>Central logging requires product</td>
<td>Supports syslog UDP</td>
<td>OS X</td>
</tr>
</tbody>
</table>

Conclusion: These steps are mostly not dependent on the platform, although OpenDirectory can provide a better stepping stone than AD to an unauthenticated user.
Conclusion

Suggestions to Apple

- Create new, more secure password based authentication scheme.
- Consolidate many server protocols into one, focus on integrity and confidentiality protections for that service.
- Allow for the centralized disabling of mDNS.
- Reduce dependence on SSL certificates or ship a corporate CA server.
- Invest in a GPO equivalent technology that allows for centralized hardening.
Conclusion

Should you use Macs in your Enterprise?

**Pros**
- Anti-exploit and sandbox technologies are looking good in 10.7
- Getting “hacked by accident” is still harder
- Slightly smaller body of knowledge in attacker circles

**Cons**
- Network privilege escalation is trivial
- Local UI isolation allows for easy phishing of admin creds
- No equivalent of GPO, hard to harden centrally
- Fewer products to investigate incidents

**Bottom Line:** Run your Macs as little islands on a hostile network.
QUESTIONS?
HTTPS://WWW.ISECPARTNERS.COM

THANKS TO ASTHA SINGHAL AND ROGER MEYER