SecurusGlobal

# Mobile Security

Norman Yue

# About Us

- Technical Background (Pentesting)
- Information Security (5~ years)

- We started mobile app testing early (~2010)
- We were the first to tell a major Australian Telco to implement network segregation for their iDevices
- We discovered (among other things) CVE-2009-0958, iDevice Invalid Certificate MITM
- We've tested a large number of mobile applications for a wide variety of clients
  - Code review, pentesting
  - This is what we've learned!

# Why Mobile Security

- Mobile devices have been introduced into business rapidly;
  - Security is playing catch-up
    - E.g. mobile vx/malware, mobile av
    - Old issues web apps now popping up in mobile
- Mobile apps handling sensitive data
  - Mobile banking
  - Government departments (.gov's)
  - Social networking (Personal Info)

# The Current State of Mobile Sec

# The Current State of Mobile Sec (I)

- Not much public research
  - Research on platforms, but not applications
  - Platforms = hard, apps = low hanging fruit
- Lack of mainstream resources
  - Poor developer awareness
  - OWASP mobile is WIP
- Security issues at a concept / design level
  - Misplaced trust of client/server

# The Current State of Mobile Sec (II)

- Huge attack surface
  - You can't control devices - people install what they want
    - e.g. games (lol)
  - Devices typically exposed to the local network (3G)
- Human element of security
  - Hey mate can I use your iPad? Mine's acting up again…
- Mobile clients often considered entirely separately from the web back-end
  - Back-end is easier to compromise than front-end.
  - Mobile front-end =~ web browser

# Common Vulnerabilities

# Common Vulnerabilities (I)

- Content insecurely stored
  - Credentials stored on disk (SQLite, plist files)
  - Test credentials hard-coded into the app
    - Vulns in test often ~= vulns in prod
    - Compounded by poor test webserver security
  - Insecure encryption:
    - Static, symmetric keys
      - Shared across every instance of the app.
    - Misused key padding (pad with zero/known value)
      - Documentation is often shaky when it comes to crypto specifics
    - Misapplied encryption (data encrypted, credentials unencrypted).
    - Often, just use NSFileProtectionComplete!

# Common Vulnerabilities (II)

- Poor communication security
  - Custom (non-SSL) encryption
    - Static key distribution
    - Oops, you didn't encrypt the cookies… or the header, or parameter names.
  - Insecure use of SSL
    - Credentials over SSL but plaintext content transmission
    - Invalid certificates? No worries
- Poor identity management
  - Poor / custom session handling
  - Attempting to authenticate devices, not users
- Storing passwords (i.e. "remember my login")
  - Device password is not a replacement for user auth

# Common Vulnerabilities (ObjC)

- Objective C is a superset of C
  - Anything risky in C is probably risky in ObjC
- No consistent platform-level protection
  - Nothing like dangerous input exceptions
- Being based on C, it's affected by C-style vulnerabilities

# Common Vulnerabilities (ObjC)

- A lot of format string bugs (“%d%d%d%d…”);
  - Old functionality: printf, sprintf…
  - New functionality: [NSString stringWithFormat:], NSLog()…
  - Corona Jailbreak (Jan2012) used a fmt as it’s initial “attack vector”.
- Filesystem Directory Traversal (“../../../../”)
  - Most developers don’t even consider this
  - Much less common, not easily exploitable
- Memory management issues
  - Use objects and you’re usually okay,
  - Malloc() and free() accessible, not always used securely.

# Common Vulnerabilities (Android)

- Android uses Java (mostly)
  - But provides the option to use native code (Android NDK)
  - Native code is affected by traditional vulnerabilities – overflows, memory management issues, type casting issues, etc.
- Open platform, decentralized market
  - Many devices are designed to allow unsigned code (i.e. accessible to an attacker)
  - No need to jailbreak – many devices allow unsigned code to run.

# Platform-Level Issues

# Platform Patching

- Android patching is not uniform
  - Carriers don't always provide timely updates
  - Workarounds, but not officially supported
- iOS won't allow downloads over certain sizes over 3G
  - Platform updates usually > this limit
  - iOS patch availiablity is more consistent
- No solution to this ☹

# Mobile Malware

- Apple says no AV
  - Security relies on “Firmware and hardware” features
  - i.e. signed code only
  - Apple also says no default browsers except Safari
- Android AV sucks
  - Study in 2011 showed at-best 80% effectiveness
  - http://www.av-test.org/fileadmin/pdf/avtest_2011-11_free_android_virus_scanner_english.pdf
- Don’t store data on the device unnecessarily.

# Platform Trust

- One of the most common mistakes is trusting mobile clients
- There is no way of preventing an attacker:
  - Intercepting traffic (by design)
  - Impersonating your client (by design)
  - Jailbreaking the platform
  - Asking nicely for passwords

# Platform Trust (tl;dr)

- Don’t trust a mobile client
  - Treat a mobile client like a web browser
- Don’t trust the web server
  - Don’t store your data locally, then call isAuthenticated(user,pass) on the back-end.
  - Retrieve data when it’s needed, store a minimum on the device.

# How do we fix things?

# Mobile Security tl;dr

- Client-Side
  - Minimize attack surface
    - Store as little data as possible
    - Encrypt what data is stored
  - Ensure Secure Communications
    - Securely used SSL > Custom crypto
  - Authenticate users, not devices
- Server-Side
  - Don’t trust client integrity / user input.
  - Pentest back-end services.
  - Treat mobile client like a web browser.

# Effective Pentesting

- Test your client and the back-end at the same time
  - Helpful for cross-referencing stuff
- Provide source if possible
  - Helpful for determining how things work
  - Includes back-end if possible
- Provide sample data (2 sets)
  - Helps weed out false negatives
  - Helps test for access control
- Have a design review (even a brief one!)
  - Avoid “your app is insecure by design”

# Thanks for attending!

Questions?

Feedback?

(p.s. You should totally come to our next talk, on BYOD and security!)