

**TEAM JOCH vs. Android:**



**The Ultimate Showdown**

# TEAM JOCH



**Jon Oberheide**

+



**Zach Lanier**

=

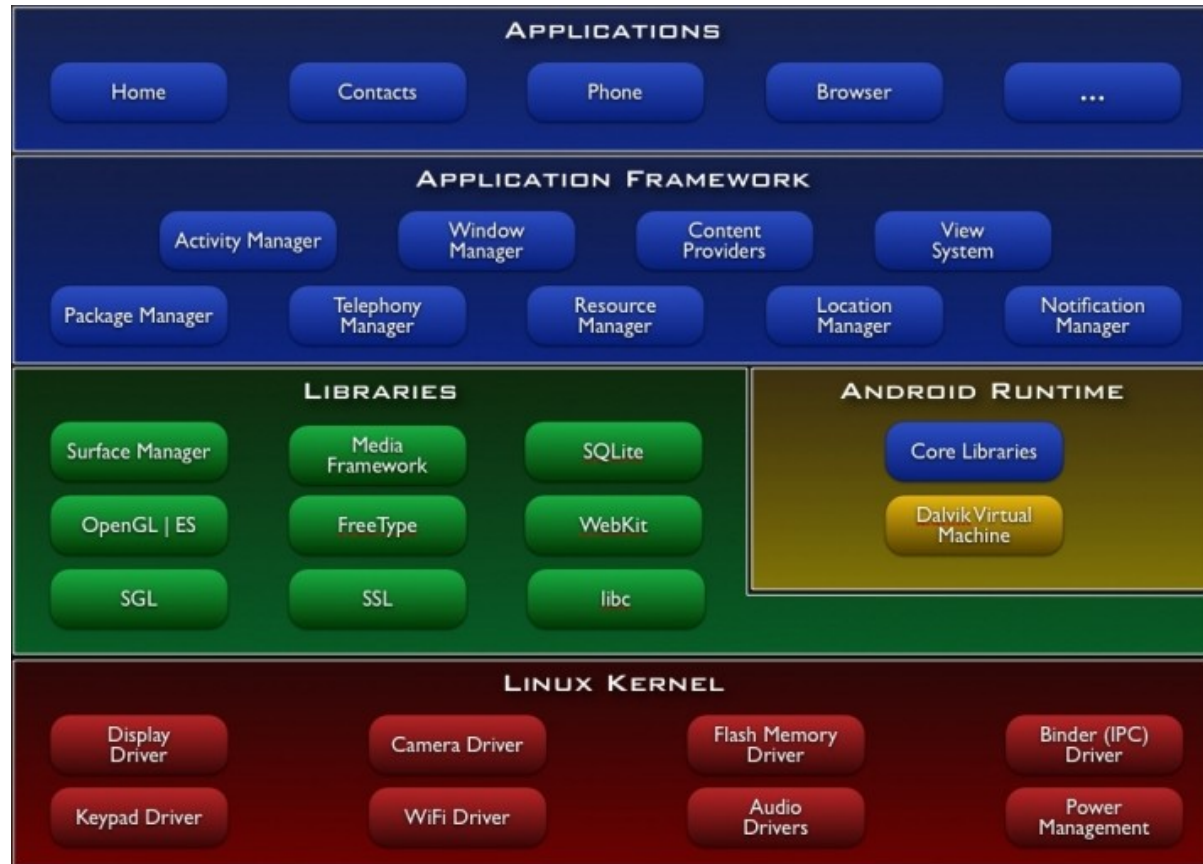
**TEAM JOCH**

# Agenda

- **Android Security Overview**
- Kernel Security
- Platform Security
- Application Security

# Android Overview

- Base platform
  - ARM core
  - Linux 2.6.3x 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



# Hardware Features

- ARM11 TrustZone?
  - Unused!
- ARM11 Jazelle JVM?
  - Unused!
- ARMv6 eXecute-Never (XN)?
  - Unused!



# Linux Environment

```
afdf01000-afdf02000 rw-p 00001000 1f:03 607 /system/lib/libstdc++.so
afe00000-afe39000 r-xp 00000000 1f:03 487 /system/lib/libc.so
afe39000-afe3c000 rw-p 00039000 1f:03 487 /system/lib/libc.so
afe3c000-afe47000 rw-p afe3c000 00:00 0
b0000000-b0013000 r-xp 00000000 1f:03 382 /system/bin/linker
b0013000-b0014000 rw-p 00013000 1f:03 382 /system/bin/linker
b0014000-b001a000 rwxp b0014000 00:00 0
bed29000-bed3e000 rwxp befef000 00:00 0
[stack]
#
```

```
afdf01000-afdf02000 rw-p 00001000 1f:03 607 /system/lib/libstdc++.so
afe00000-afe39000 r-xp 00000000 1f:03 487 /system/lib/libc.so
afe39000-afe3c000 rw-p 00039000 1f:03 487 /system/lib/libc.so
afe3c000-afe47000 rw-p afe3c000 00:00 0
b0000000-b0013000 r-xp 00000000 1f:03 382 /system/bin/linker
b0013000-b0014000 rw-p 00013000 1f:03 382 /system/bin/linker
b0014000-b001a000 rwxp b0014000 00:00 0
be8ab000-be8c0000 rwxp befef000 00:00 0
[stack]
#
```

Executable  
stack/heap!

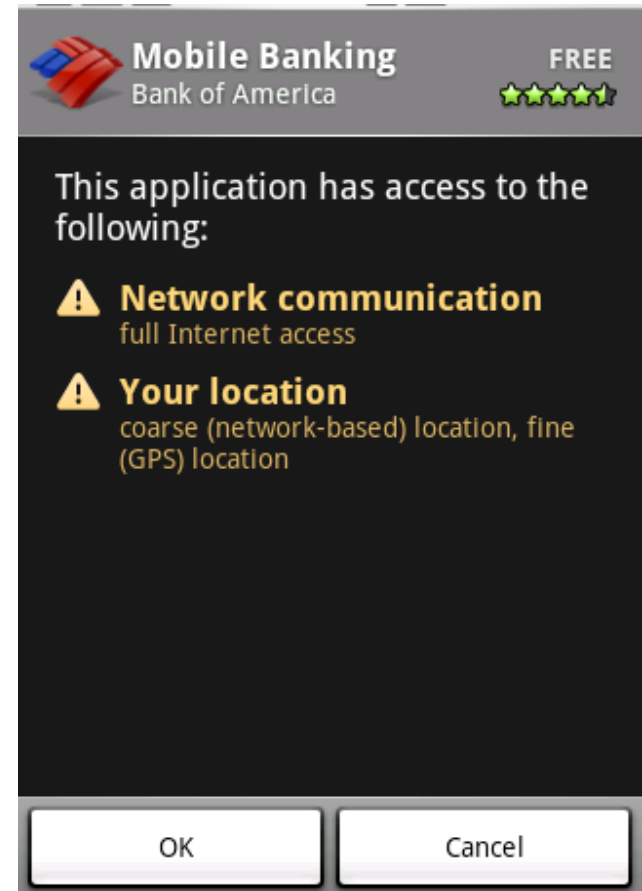
```
# cat /proc/sys/kernel/randomize_va_space
1
#
```

Mobile ASLR sucks,  
where's my 64-bit CPUs?!?

Non-  
randomized  
mmap/brk!

# 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



# App Sandboxing

- “Sandboxed” by standard UNIX uid/gid
  - generated unique per 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



# App Distribution

- Application signing
  - No CAs
  - Self-signed by developers
- Android Market
  - \$25 signup, anyone can publish
  - Anonymous sign-up possible



# Agenda

- Android Security Overview
- **Kernel Security**
- Platform Security
- Application Security

# The Linux Kernel

- Linux kernel = swiss cheese
  - Jailbreaks, aka local privesc, are plentiful
  - Mostly thanks to stealth/743C
- Shameless plug!
  - If you care about kernel exploitation, come to:

Hackito Ergo Sum  
2011



# Android Native Code

- Dalvik VM != sandbox
  - Not limited to executing dex bytecode
  - Can pop out of the VM to execute native code
  - Any 3rd party app can root your phone by exploiting a kernel vulnerability via native code
- Native code packaged within APKs
  - Android should do some code signing like iPhone
  - But it doesn't, so why limit execution of native code to build-time packaged modules?

# RootStrap

- Getting root is easy, but how do it most effectively as an attacker
- Enter, RootStrap
  - Silent runtime fetching and execution of remote ARM payloads
  - Not really a bot..more of a general purpose distributed computing platform ;-)



# Native ARM Code Delivery

- Fetch index file
  - Lists available exploits and module names
- Yank down ARM modules
  - Dumped to Android app private storage
  - eg. /data/data/org.rootstrap/files, not ./libs
- Load via JNI and execute each payload
  - System.load(“.../files/root1.so”);
  - result = root1();

```
jonoslice rootstrap # cat index
root1.so
root2.so
jonoslice rootstrap # file root*.so
root1.so: ELF 32-bit LSB shared object, ARM, version 1 (SYSV), dynamically linked, not stripped
root2.so: ELF 32-bit LSB shared object, ARM, version 1 (SYSV), dynamically linked, not stripped
jonoslice rootstrap #
```

# How to Build a Mobile Botnet

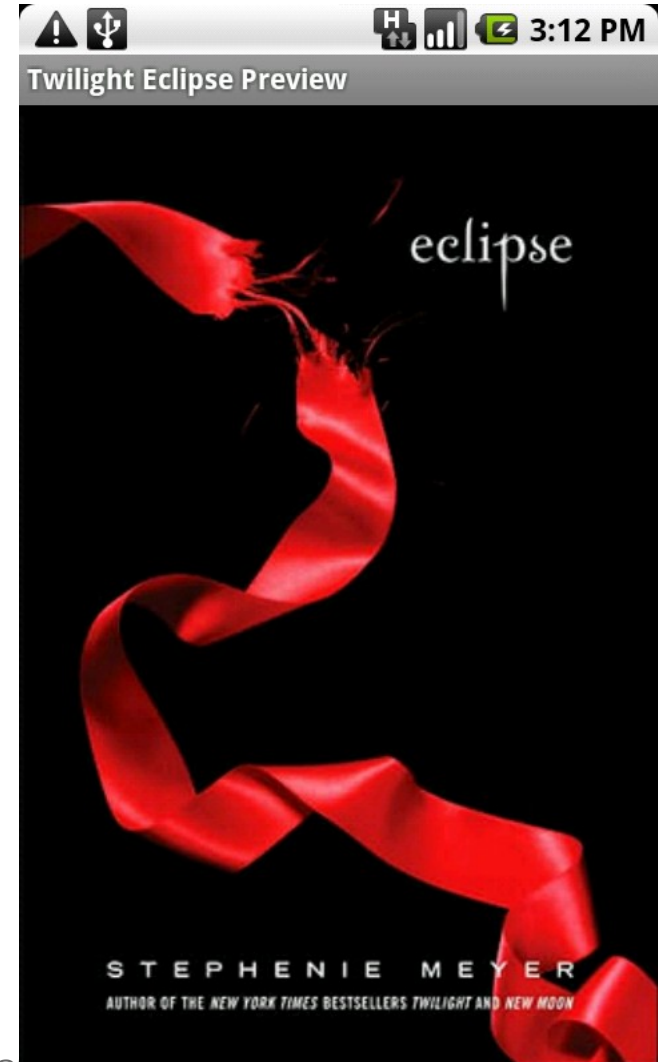
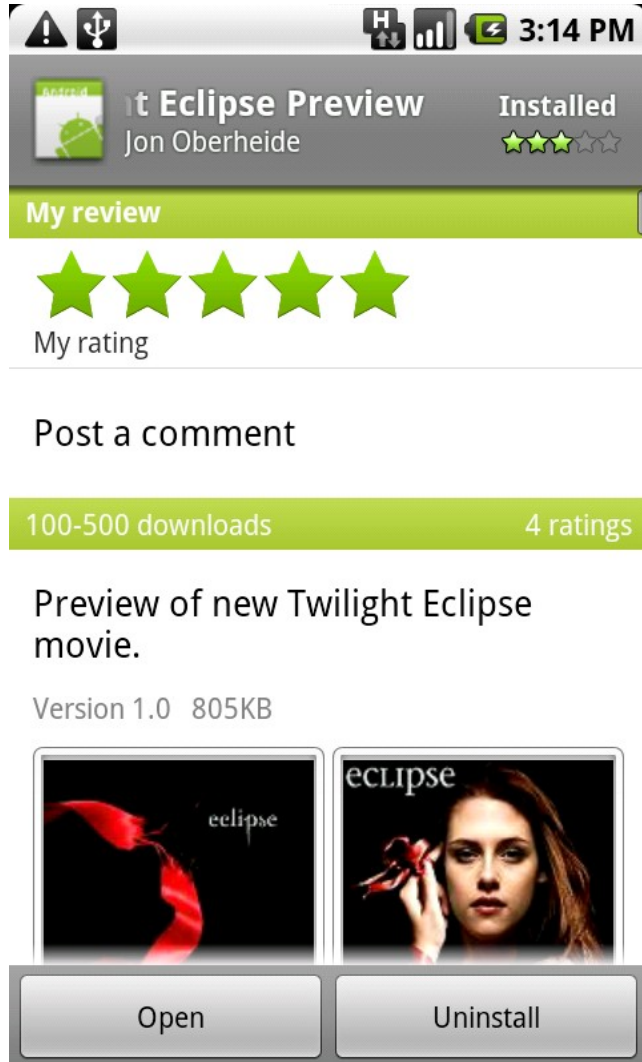
- Build some fun legit-looking games / apps
  - Include RootStrap functionality
  - Periodically phone home to check for new payloads
- As soon as new kernel vuln discovered, push out exploit payload
  - Before providers push out OTA patch
  - Trivial to win that race, slow OTA updates
- Rootkit a bunch of phones!

# A Wolf in Vampire's Clothing?

- RootStrap app is boring and not sneaky
  - No one would intentionally download it
  - Need something legit looking to get a large install base
- Hmm...what to do, what to do...



# Fake Twilight Eclipse App



# Andy and Jaime Don't Like It :-)

The screenshot shows a 'Comments' section with a green header. Below the header, there are two comments. The first comment is from 'Andy' dated '6/16/2010', with a rating of 1 star (out of 5) and the text 'Defective'. The second comment is from 'Jaime' dated '6/16/2010', with a rating of 1 star (out of 5) and the text 'Loads but you can't see any other photos'. Below the comments is a link that says 'Read all comments'. At the bottom of the screenshot are two buttons: 'Open' and 'Uninstall'.

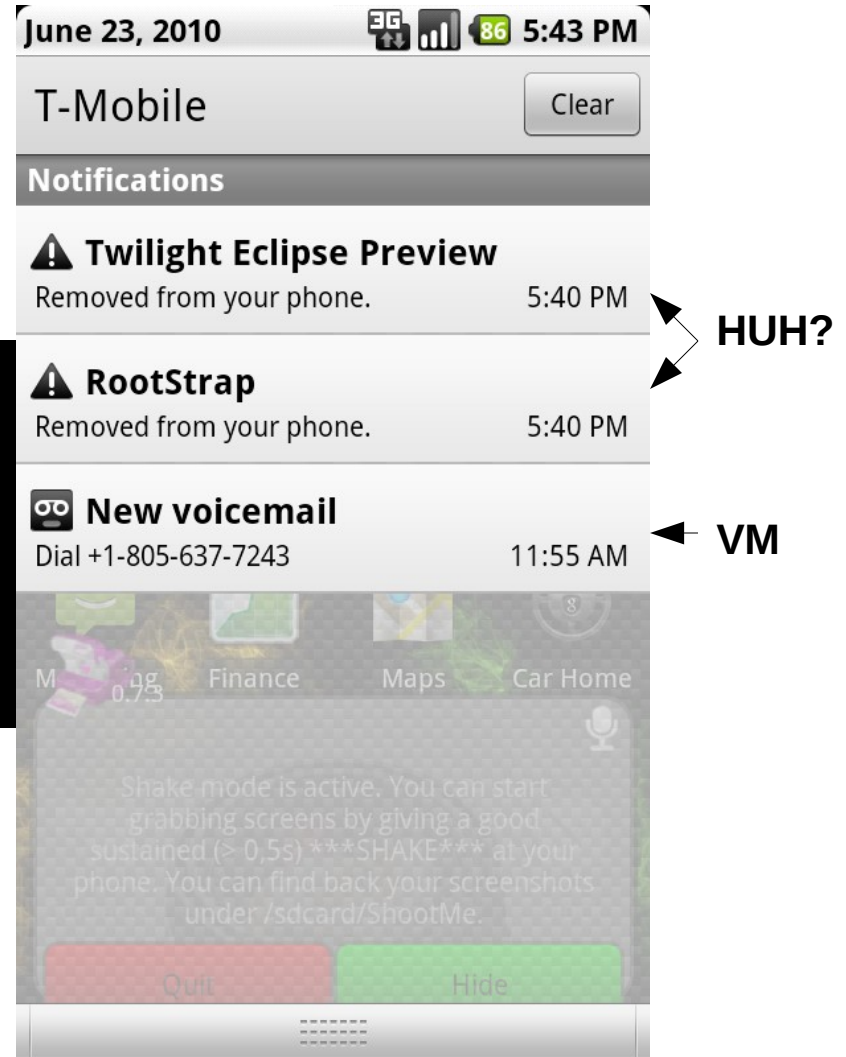
- Still, 200+ downloads in under 24 hours
- With a legit-looking app/game, you could collect quite an install base for RootStrap

# Android Remote Kill

- BZZZ!

```
connection:
heartbeat: 48 / 25% / 0%
login: 80 / 42% / 75%
data message:
INSTALL_ASSET: 1 / 0% / 3%
REMOVE_ASSET: 2 / 1% / 3%
```

- WAT?



# Android Remote Kill/Install

- Android has remote kill/wipe functionality built-in
  - Google can remotely remove installed apps from any Android device
  - GTalkService persistent connection
  - REMOVE\_ASSET remote intent invocation
- Also, remote *installation* functionality

# Kernel Security Wrap-up

- No excuses Google, it's 2011!
  - Harden your kernel / toolchain
  - Signed code restrictions a la iPhone
- Supporting native code makes it worse
  - Packaging/install time: ok
  - Runtime native code delivery: not ok

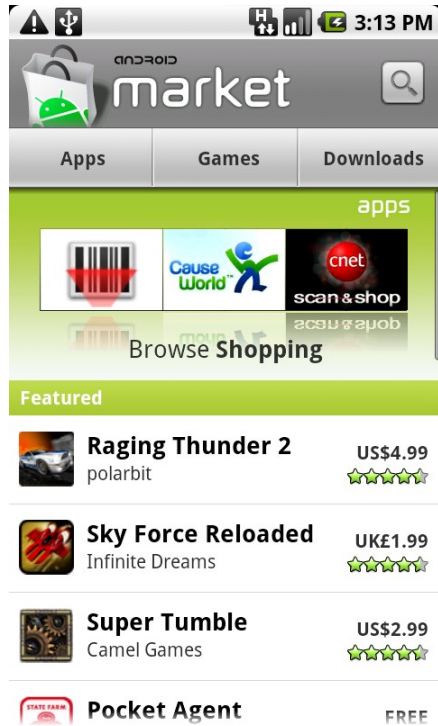
# Agenda

- Android Security Overview
- Kernel Security
- **Platform Security**
- Application Security

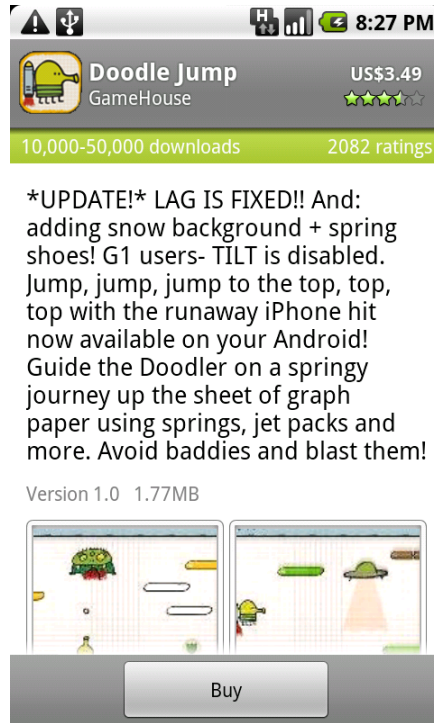
# Platform Security

- There's a lot of “platform goo” in the middle between the kernel and applications
- What to attack?
  - Not kernel, not apps!
  - How about permissions framework?
- Permissions approval process
  - Intended to warn the user about potentially unsafe actions an app can perform

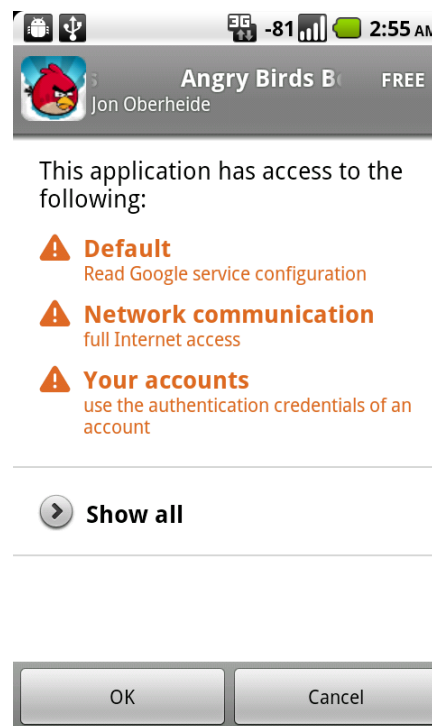
# Perceived App Install Process



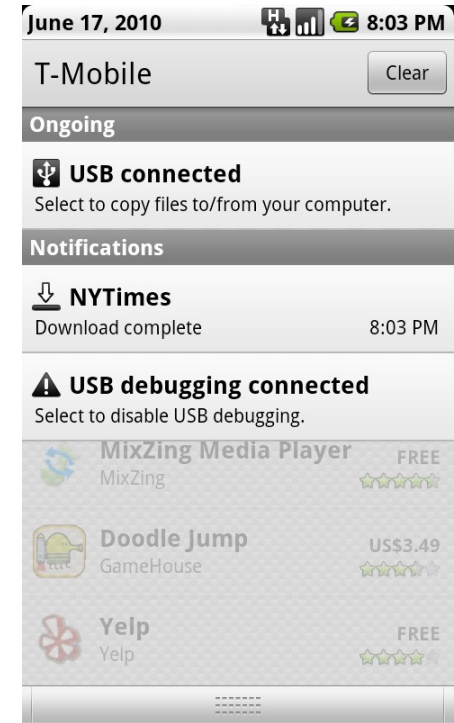
**BROWSE**



**INSTALL**



**APPROVE?**



**INSTALLED!**



# ACTUAL Market Flow

- Google is a sneaky panda!
  - You don't actually download / install the app through the market application
- When you click install in market app
  - Google servers push an out-of-band message down to you via persistent data connection
  - Triggers `INSTALL_ASSET` intent to start install
  - Intent handler fetches APK and installs

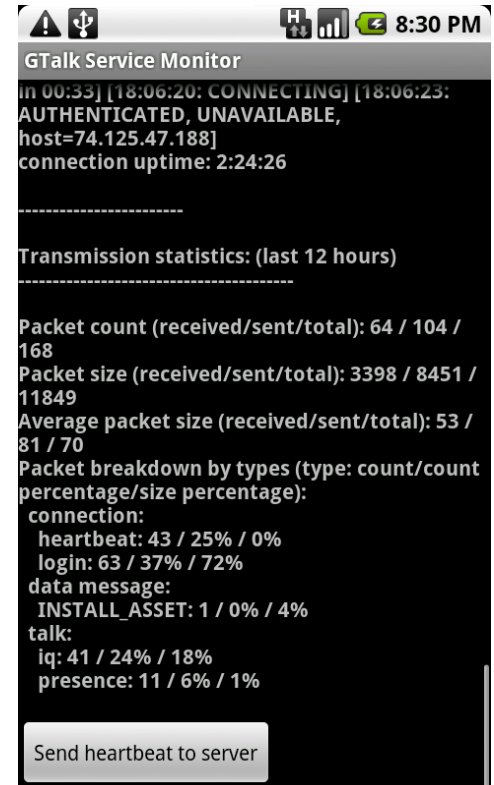
# Dex Bytecode RE

```
#1      : (in Lcom/android/vending/InstallAssetReceiver;)
  name  : 'isIntentForMe'
  type  : '(Landroid/content/Intent;)Z'
  access : 0x0001 (PUBLIC)
  code  : -
  registers : 5
  ins   : 2
  outs  : 3
  insns size : 37 16-bit code units
```

```
0442f4:      |[[0442f4] com.android.vending.InstallAssetReceiver.isIntentForMe:(Land
044304: 1202      |0000: const/4 v2, #int 0 // #0
044306: 6e10 7d00 0400 |0001: invoke-virtual {v4}, Landroid/content/Intent;.getAction:()Ljava
04430c: 0c00      |0004: move-result-object v0
04430e: 1a01 d20d   |0005: const-string v1, "android.intent.action.REMOTE_INTENT" // strin
044312: 6e20 a012 1000 |0007: invoke-virtual {v0, v1}, Ljava/lang/String;.equals:(Ljava/lang/
044318: 0a00      |000a: move-result v0
04431a: 3800 1800   |000b: if-eqz v0, 0023 // +0018
04431e: 1a00 da0d   |000d: const-string v0, "android.intent.extra.from_trusted_server" //
044322: 6e30 7e00 0402 |000f: invoke-virtual {v4, v0, v2}, Landroid/content/Intent;.getBoolea
044328: 0a00      |0012: move-result v0
04432a: 3800 1000   |0013: if-eqz v0, 0023 // +0010
04432e: 6e10 7f00 0400 |0015: invoke-virtual {v4}, Landroid/content/Intent;.getCategories:()L
044334: 0c00      |0018: move-result-object v0
044336: 1a01 6504   |0019: const-string v1, "INSTALL_ASSET" // string@0465
04433a: 7220 3713 1000 |001b: invoke-interface {v0, v1}, Ljava/util/Set;.contains:(Ljava/lang
044340: 0a00      |001e: move-result v0
044342: 3800 0400   |001f: if-eqz v0, 0023 // +0004
044346: 1210      |0021: const/4 v0, #int 1 // #1
044348: 0f00      |0022: return v0
04434a: 0120      |0023: move v0, v2
04434c: 28fe      |0024: goto 0022 // -0002
```

# GTalkService Connection

- Persistent data connection
  - Speaks XMPP
  - Same connection now used for C2DM push service
- It's SSL, but...
- If you MITM or C2DM spoof
  - Remote intent / app install
- If you pop GTalkService servers
  - Push down code to all Android phones in the world



```
GTalk Service Monitor
in 00:33] [18:06:20: CONNECTING] [18:06:23:
AUTHENTICATED, UNAVAILABLE,
host=74.125.47.188]
connection uptime: 2:24:26

-----
Transmission statistics: (last 12 hours)
-----
Packet count (received/sent/total): 64 / 104 /
168
Packet size (received/sent/total): 3398 / 8451 /
11849
Average packet size (received/sent/total): 53 /
81 / 70
Packet breakdown by types (type: count/count
percentage/size percentage):
connection:
  heartbeat: 43 / 25% / 0%
  login: 63 / 37% / 72%
data message:
  INSTALL_ASSET: 1 / 0% / 4%
talk:
  iq: 41 / 24% / 18%
  presence: 11 / 6% / 1%

Send heartbeat to server
```

# Gap in Responsibility

- Market app performs permission approval
- But GTalkService triggers actual install
- There's a disconnect here...

# Market App Requests

- What does the market app POST to the market server?
- Can we spoof the same request and trigger an `INSTALL_ASSET` message and subsequent install?

# Base64 Encoded Protobuf Payload

```
POST /market/api/ApiRequest HTTP/1.1
Content-Length: 524
Content-Type: application/x-www-form-urlencoded
Host: android.clients.google.com
Connection: Keep-Alive
User-Agent: Android-Market/2 (dream DRC83); gzip
```

```
version=2&request=CuACCvYBRFFBQUFL0EFBQUJvZWVEVGo4eGV40VRJaW9YYmY3T1FSZGd4dH
wxM2VZTlltUjFMV2hLa3pwSFdUY0xtc1lNNHM0FRPTWwtM1dkTU9JbUQ3aUdla1hUMFg5R1htd1Et
SmU3SzVSRW1US0lslmJPeTVHNzc5Y0pNZTFqb09DQUlyT2RXRVZnR0NNaUN5TkYtS2VtUUhLWEM2Vk
hREAAYhA0iD2YyZjE1Y2NkMTdmYjMwNSoHZHJlYW06NDICZW46A1VTQgdBbmRyb2lkSgdBbmRyb2lk
NjA2ZGIzMDAwZDQ4MGQ2MxNSFAoSMzUzOTk5MzE5NTg1NDczFA
```

# Raw Protobuf Decoded

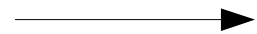
```
1 {
  1: "DQAAAJ0AAACtMCMW8jookK40nhA80M17c4tEsHT_LE0EyX46iYT062oHj0lWSjb-ndSDr0CNwvUDy2yFLD6E6EsL
Xxd-iwGsyAlTRPalqolXdcsHjz-HoGp-2JrD5UhwRiC30yHy_EYUju0wKRIY9BRXiaTG-oxIrQSbtKy8PLDXCjNP-8P_1YzrIt
  2: 0
  3: 1002
  4: "d552a36f69de4a"
  5: "dream:3"
  6: "en"
  7: "US"
  8: "Android"
  9: "Android"
  10: "310260"
  11: "310260"
  12: "am-google-us"
}
2 {
  4 {
    4: "-3271901821060548049"
    6: 1
  }
}
2 {
  5 {
    1: "-3271901821060548049"
    2: 0
    3: 3
    4: 1
  }
}
```

# RE'ed Protobuf Specification

app/asset ID



auth token



install request message



```
message UnknownThing {
    optional fixed64 mgoogle = 12;
}

message InstallRequest {
    optional string appId = 1;
}

message RequestContext {
    required string authSubToken = 1; // authsub token for service 'android'
    required int32 unknown1 = 2; // always 0
    required int32 version = 3; // always 1002
    required string androidId = 4; // android id converted to hexadecimal
    optional string deviceAndSdkVersion = 5; // ro.product.device ':' ro.build.version.sdk
    optional string userLanguage = 6; // ro.product.locale.language
    optional string userCountry = 7; // ro.product.locale.region
    optional string operatorAlpha = 8; // gsm.operator.alpha
    optional string simOperatorAlpha = 9; // gsm.sim.operator.alpha
    optional string operatorNumeric = 10; // gsm.operator.numeric
    optional string simOperatorNumeric = 11; // sim.gsm.operator.numeric
    optional UnknownThing unknown12 = 12;
    optional string unknown13 = 13;
}

message Request {
    optional RequestContext context = 1;
    repeated group RequestGroup = 2 {
        optional InstallRequest installRequest = 10;
    }
}
```



# Elements of a Install Request

- We have the format of the request now!
- Need to populate it with:
  - Lots of miscellaneous fields...
  - App ID: target app to be installed
    - Can be derived from dissecting market requests
  - Auth token: the hard part?
    - Turns out we can steal it from Android's AccountManager!

```
te OnClickListener button_click = new OnClickListener() {
    public void onClick(View v) {
        AccountManager accountManager = AccountManager.get(getApplicationContext());
        Account acct = getAccount(accountManager);
        accountManager.getAuthToken(acct, "android", false, new GetAuthTokenCallback(), null);
        return;
    }
}
```

# Bypassing Permissions Approval

- Steal the “android” service token used by market from the AccountManager
- Construct protobuf request to market servers for invoking an application installer
- INSTALL\_ASSET is pushed and app installed without any user prompt / permission approval
- PoC disguised as an Angry Birds expansion app

# Angry Birds Bonus Levels

Angry Birds Bonus Levels FREE  
Jon Oberheide

About Comments

**Description**

Bonus levels for Angry Birds.

Version 1.0 438KB  
<50 downloads 0 ratings

**About the developer**

View more applications

Visit the developer's Web page  
<http://jon.oberheide.org>

Install

Angry Birds Bonus Levels

Install Angry Birds Bonus Levels

Please click the above button to install the bonus Angry Birds levels!

November 9, 2010 -75% 11:39 PM

T-Mobile Clear

**Ongoing**

**USB connected**  
Select to copy files to/from your computer.

**Notifications**

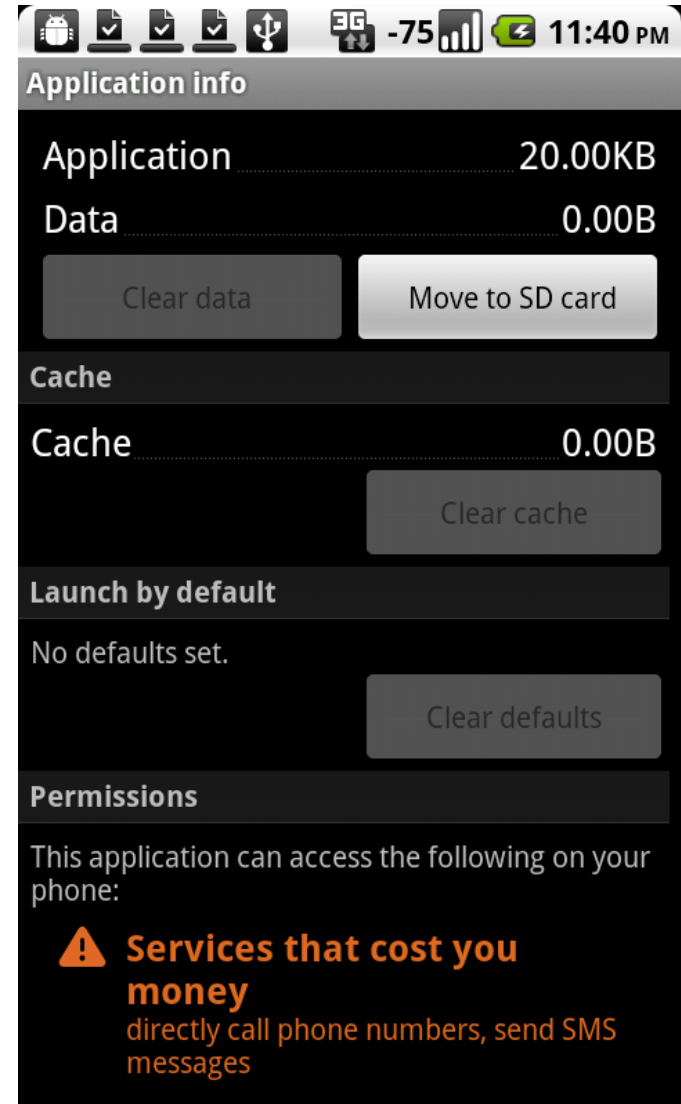
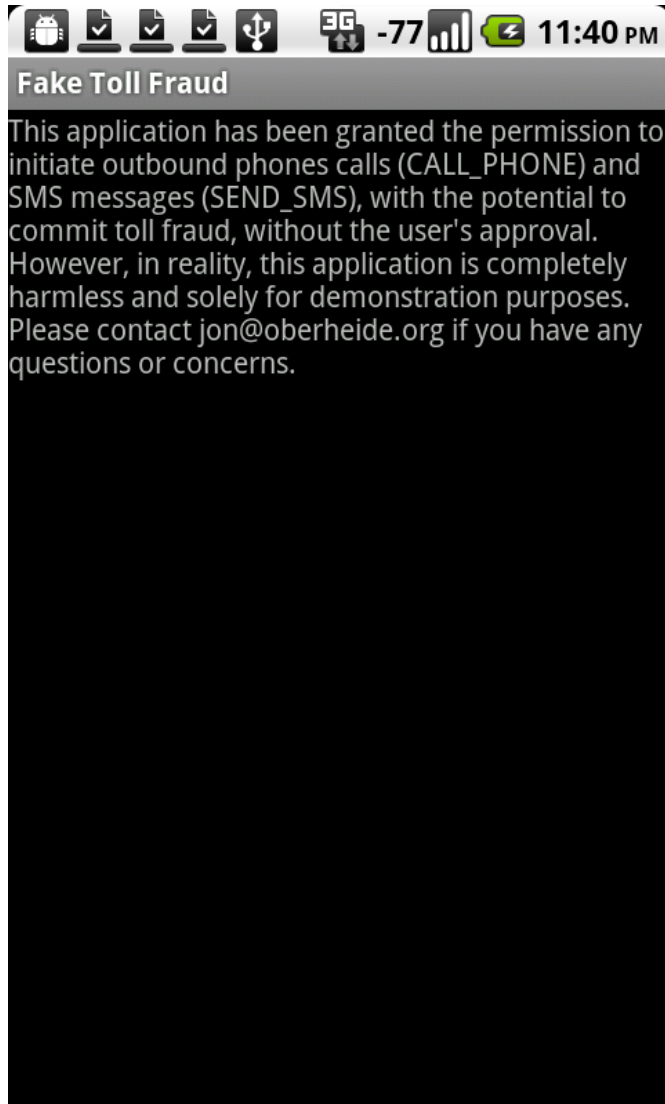
**Fake Location Tracker**  
Successfully installed. 11:39 PM

**Fake Toll Fraud**  
Successfully installed. 11:39 PM

**Fake Contact Stealer**  
Successfully installed. 11:39 PM

**USB debugging connected**  
Select to disable USB debugging.

# Fake Toll Fraud App



# Platform Security Wrapup

- Vulnerability status:
  - Donut: fixed
  - Froyo: fixed
  - Eclair: no confirmation yet, may be vulnerable
- Platform complexity leads to vulns
  - Round-about market / GtalkService procedure
  - “server-initiated” flag fix worth investigation

# Agenda

- Android Security Overview
- Kernel Security
- Platform Security
- **Application Security**

# Broad Observations



*TEAM JOCH vs. Android - ShmooCon 2011*

# Broad Observations

- The Web pushed content to the browser
  - Centralization of apps & data
  - Always a push for MORE (ActiveX, applets, Flash)
- Now, everyone gets their own app!
  - Code (not HTML) gets pushed to the endpoint
  - XKCD Viewer



# Broad Observations

- AuthC/AuthZ
  - Carrier Applications
    - “we trust you because you’re on our network”
  - Third-party Applications
    - SOMETIMES better than carrier apps
      - Incomplete support of open standards
    - Client-side data trust issues
      - admin=1

# Broad Observations

- HyperGlobalMegaCloudDataMeshStore
  - Many Apps for syncing data between device and CLOUD
    - Full AuthC and AuthZ bugs

# Testing Techniques



# Testing Techniques

- White Box Source Code Review
  - Sometimes, it's trivial to get app source code
- Black Box
  - Acquiring Application Binaries
  - Reverse Engineering
    - Disassembly/Decompilation
  - Network Analysis
    - Protocol Analysis, fuzzing
  - MITM

# Testing Techniques

```
oishi$ adb pull /system/app/com.amazon.mp3.apk
1241 KB/s (552871 bytes in 0.434s)
oishi$ unzip com.amazon.mp3.apk
Archive:  com.amazon.mp3.apk
  inflating: META-INF/MANIFEST.MF
  inflating: META-INF/MP3TEAMS.SF
  inflating: META-INF/MP3TEAMS.RSA
  inflating: res/drawable/album_detail_info_background.xml
  inflating: res/drawable/album_track_toggle_active_background.xml
  inflating: res/drawable/album_track_toggle_inactive_background.xml
  extracting: res/drawable/artwork_placeholder.png
  extracting: res/drawable/artwork_placeholder_small.png
  extracting: res/drawable/buy_button_hot_opaque.png
```

# Testing Techniques

```
# direct methods
.method static constructor <clinit>()V
    .locals 1

    .prologue
    const/4 v0, 0x0

    .line 35
    sput-object v0, Lcom/amazon/mp3/net/RestClient; ->sSocketFactoryFallback:Lorg/apache/http/conn/ssl/SSLSocketFactory;

    .line 36
    sput-object v0, Lcom/amazon/mp3/net/RestClient; ->sSocketFactory:Lorg/apache/http/conn/ssl/SSLSocketFactory;

    .line 29
    return-void
.end method

.method public constructor <init>(Ljava/io/InputStream;Ljava/lang/String;)V
    .locals 5
    .parameter "keyStoreStream"
    .parameter "keyStorePassword"
```

# Testing Techniques

```
public class RestClient
{
    public RestClient(InputStream arg0, String arg1)
    {
        /*<invalid signature>*/java.lang.Object local = com/amazon/mp3/net/RestClient;
        local;
        JVM INSTR monitorenter ;
        Object obj1 = sSocketFactoryFallback;
        if(obj1 == null) goto _L2; else goto _L1
_L1:
        obj1 = sSocketFactory;
        if(obj1 == null) goto _L2; else goto _L3
_L3:
        return;
_L2:
        Object obj = null;
        obj1 = KeyStore.getDefaultType();
        KeyStore keystore = KeyStore.getInstance(((String) (obj1)));
        char ac[] = arg1.toCharArray();
        keystore.load(arg0, ac);
        ac = JVM INSTR new #52 <Class SSLSocketFactory>;
        ac.SSLSocketFactory(keystore, arg1, keystore);
        sSocketFactoryFallback = ac;
        ac = JVM INSTR new #52 <Class SSLSocketFactory>;
        KeyStore keystore1 = null;
        ac.SSLSocketFactory(keystore, arg1, keystore1);
        sSocketFactory = ac;
        IoUtility.close(arg0);
_L4:
```

# Testing Techniques

- Not everyone can be a Binary RE ninja
  - ...and project timelines don't allow for on-the-job training :-)
- Sometimes the easiest way to understand an application is to look at its TRAFFIC
- You need to become the MITM
  - Just like WAPT, and Burp, WebScarab, etc.



# Testing Techniques

- MAPT MITM Challenges!
  - Run the app in an emulator (boring)
  - Connect the phone to your own WAP
    - Uplink your WAP to your laptop with Internet sharing enabled
  - Run Wireshark
  - WiFi not always an option
    - Handset might not support WiFi
    - Application might require carrier network
      - Change server.carrier.com to testsite.com

# Testing Techniques

- MAPT MITM Challenges!
  - Wireshark lets you see traffic
  - SYN TCP 80? Easy.
  - SYN TCP 443? A little harder.
  - SYN TCP 9999? Ok...
    - Binary data?! Huh?
  - UDP DST Port 4717?!?
    - I quit!

# Case Studies



*TEAM JOCH vs. Android - ShmooCon 2011*

# Case Study: Foursquare

- Foursquare client for Android
- Originally written in Java, like most Android applications
  - Source available under Apache 2.0 license



# Case Study: Foursquare

- Foursquare API supports Basic Auth and OAuth...
  - OAuth includes signatures for transactions, helps prevent replay attacks, etc.
  - Guess which one foursquared uses

# Case Study: Foursquare

- That's right. HTTP Basic Auth...over plaintext transport

```
14:54:35.510013 IP (tos 0x0, ttl 64, id 38148, offset 0, flags [DF], proto TCP (6), length 250) 25.33.█.40734 > 174.129.33.12.80:
 1 win 2920
E.#####@.r##!Ix#.!....P.#.####.P..h7##.GET /v1/user?mayor=0&badges=0 HTTP/1.1
User-Agent: com.joelapenna.foursquared 2010011401 ← User-Agent header identifying the Android Foursquare app
Host: api.foursquare.com
Connection: Keep-Alive
Authorization: Basic ZXZpbHNxdWVfYzUBuMHdoZXJlLm9yZzpnbnZ9kdmlzaW9u ← HTTP Basic Auth
```

- There's a CWE for that!
  - CWE-311: Missing Encryption of Sensitive Data (including credentials)

# Case Study: Foursquare

- Why is this a problem?
  - EVERYONE uses Foursquare
    - Well, maybe not you, but everyone else!
  - Most applications “prefer” WiFi to cell radio
    - => trivial interception of creds
- Funny enough, Foursquared has OAuth support
  - But it’s not actually used

# Case Study: Storage Application

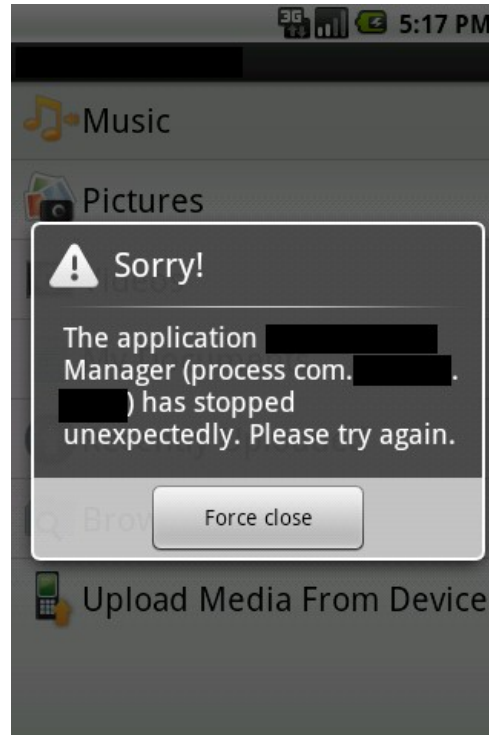
- Multi-platform application for storing and retrieving music, videos, documents, and more
  - Android, BREW, Blackberry, and fat web browser
- Proprietary, binary-only



# Case Study: Storage Application

- Simple crash in storage quota viewer
  - Divide-by-zero error leads to DoS
  - Attacker must successfully intercept and modify server response for this to happen
    - A bit more difficult since this tends to occur over the carrier's network, but WiFi is still an option

# Case Study: Storage Application

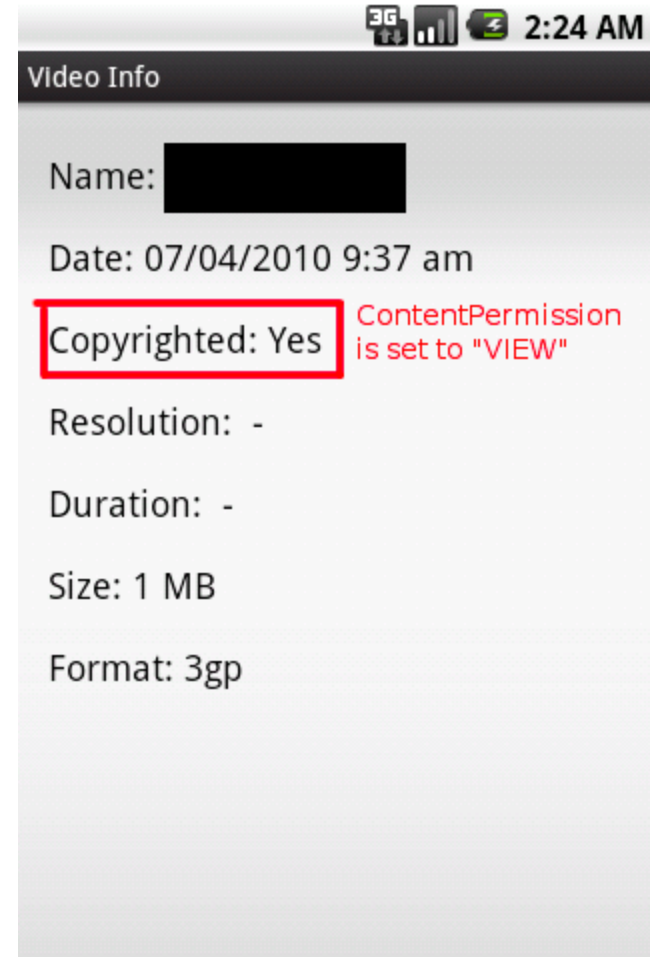
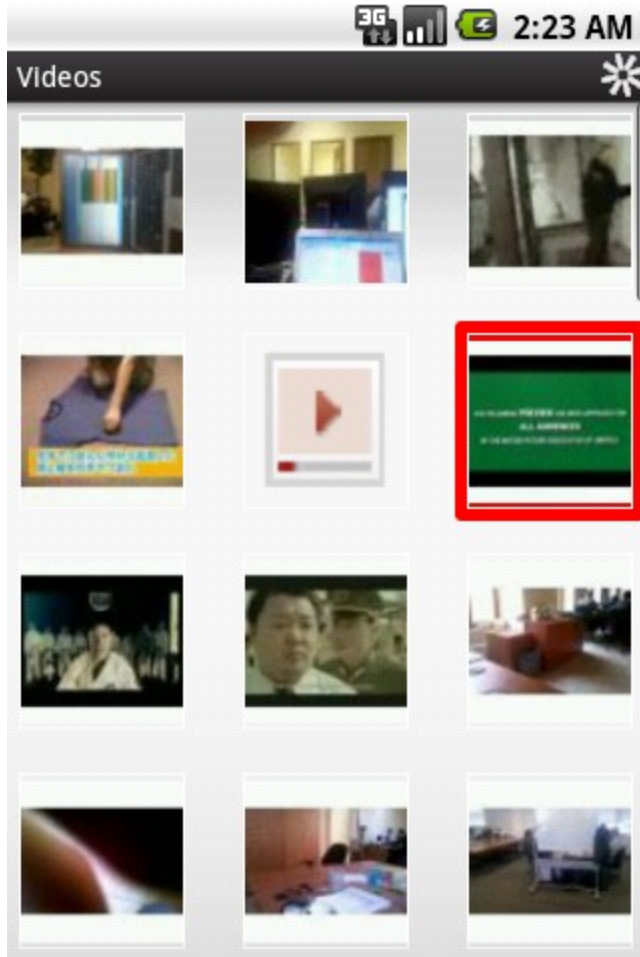


```
E/AndroidRuntime( 261): Uncaught handler: thread main exiting due to  
E/AndroidRuntime( 261): java.lang.ArithmeticException: divide by zero  
E/AndroidRuntime( 261):         at com. .gui.activities.Sto
```

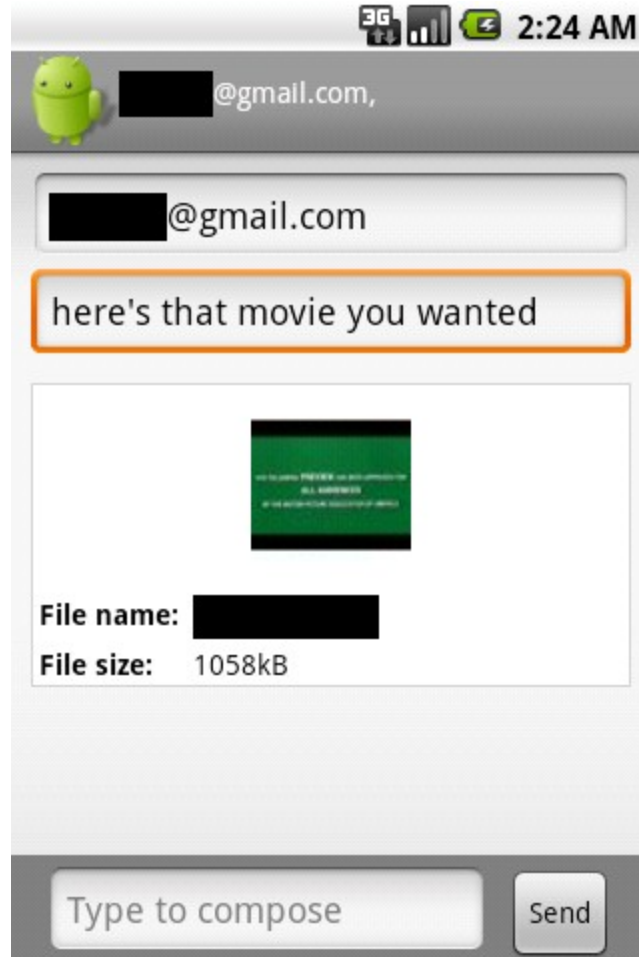
# Case Study: Storage Application

- Diddling with “Digital Rights Management”
  - App supports sharing of video, audio, image content with your contacts
  - Enforces “DRM” on “protected” files
    - Often copyrighted or premium content
  - Enforcement occurs based on the value of an attribute in the file’s XML manifest
    - Yes, Virginia, that is under the user’s control

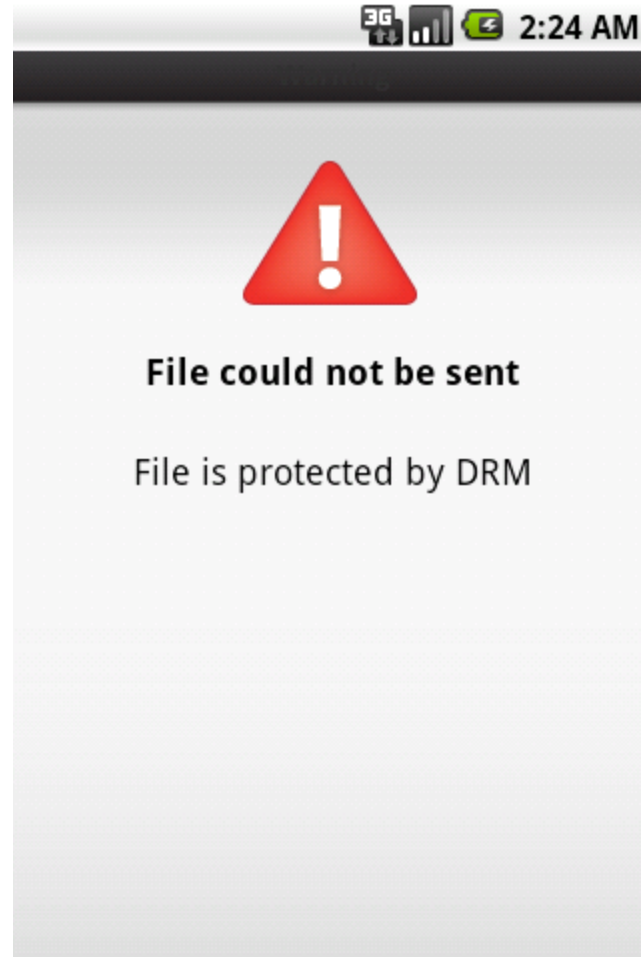
# Case Study: Storage Application



# Case Study: Storage Application



# Case Study: Storage Application



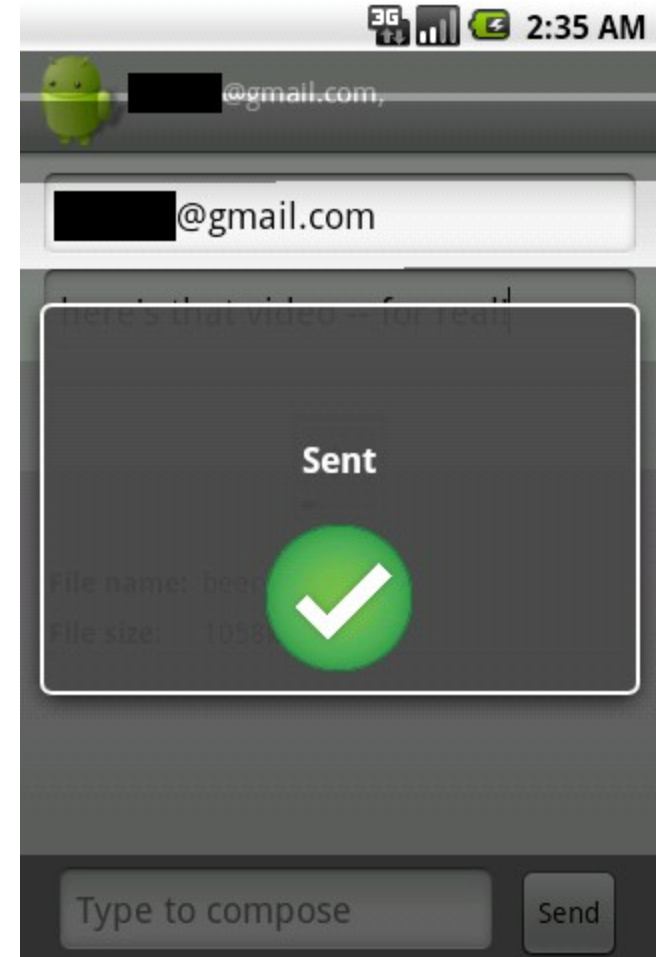
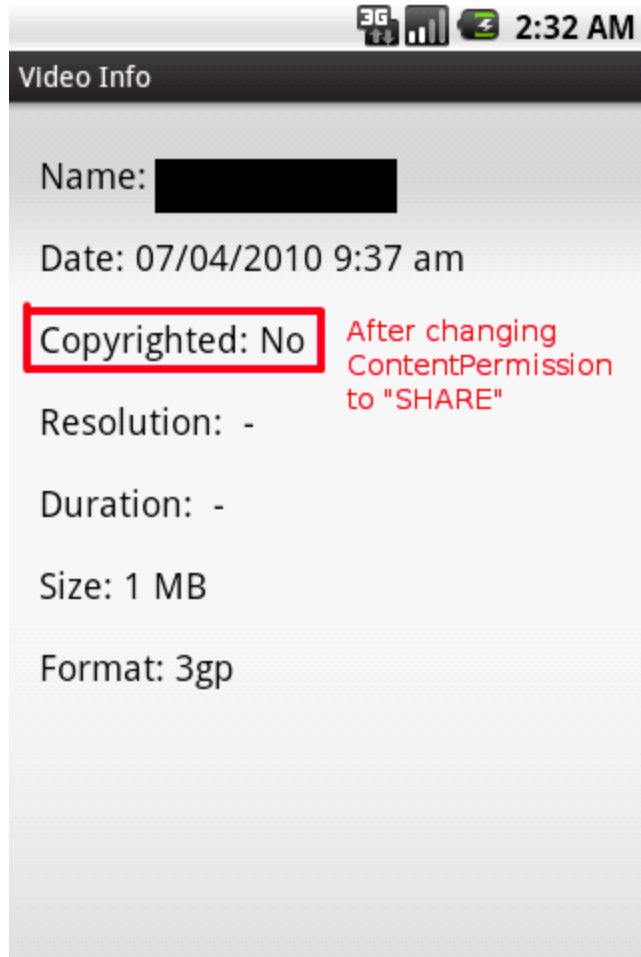
# Case Study: Storage Application

```
<EACBES10H/3UP> <EACBES10H/3UP>  
<fileAttribute name="ContentPermissions">VIEW</fileAttribute>  
<fileAttribute name="CI_COMPLETE">true</fileAttribute>  
<fileAttribute name="Duration">90530</fileAttribute>  
<fileAttribute name="Height">144</fileAttribute>  
<fileAttribute name="Width">176</fileAttribute>  
<fileAttribute name="TranscodingStatus">Success</fileAttribute>  
<fileAttribute name="ContentPermissionsDetail">PENDING</fileAttribute>  
<fileAttribute name="BitRate">95.0</fileAttribute>
```

Becomes...

```
<EACBES10H/3UP> <EACBES10H/3UP>  
<fileAttribute name="ContentPermissions">SHARE</fileAttribute>  
<fileAttribute name="CI_COMPLETE">true</fileAttribute>  
<fileAttribute name="Duration">90530</fileAttribute>  
<fileAttribute name="Height">144</fileAttribute>  
<fileAttribute name="Width">176</fileAttribute>  
<fileAttribute name="TranscodingStatus">Success</fileAttribute>  
<fileAttribute name="ContentPermissionsDetail">PENDING</fileAttribute>
```

# Case Study: Storage Application





# Case Study: Storage Application

- The “DRM” is basically enforced within the client, predicated on the response from the server
  - And that response can be intercepted and modified => “DRM” bypass
- CWE-807: Reliance on Untrusted Inputs in a Security Decision
  - I like CWE, btw

# Case Study: App Framework

- Cross-platform framework for HTML/JS “applications”
  - WinMo, Android, etc.

# Case Study: App Framework

- Custom permissions restricted us from sending messages (Intents) to the runtime

```
xmlns:android="http://schemas.android.com/apk/res/android">
  <uses-sdk android:minSdkVersion="7" android:targetSdkVersion="7" />
  <permission android:label="@string/permlabel_access_
platform_provider
" android:name="com.
.permission.ACCESS_
_INFO" android:pro
tectionLevel="signatureOrSystem" android:description="@string/permdesc_access_
platform_provider" />
```

```
# am start -a "android.intent.action.VIEW" -t "application/widget" -d "file:///sd
/.ui.manager.WidgetInstallActivity"
Starting: Intent { act=android.intent.action.VIEW dat=file:///sdcard/
/.ui.manager.WidgetInstallActivity }
java.lang.SecurityException: Permission Denial: starting Intent { act=android.int
.0/ gt typ=application/widget flg=0x10000000 cmp= /.ui
(pid=-1, uid=-1) requires com.
.permission.ACCESS INFO
```

# Case Study: App Framework

- But, other (malicious) apps can clobber widget content!
  - CWE-276: Incorrect Default Permissions
  - So we wrote a malicious app to do just that

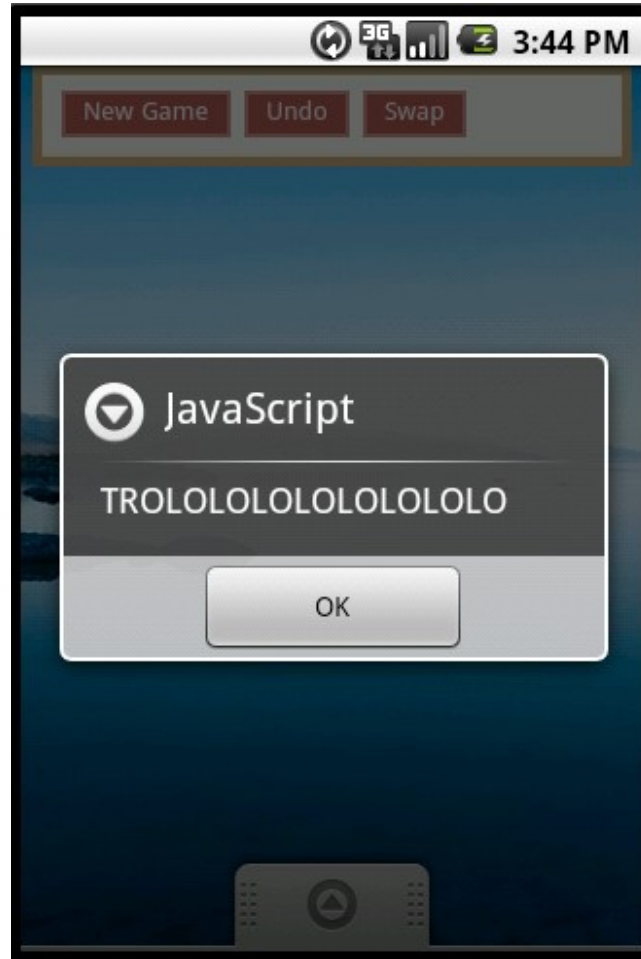
```
# ls -l /data/misc
drwxrwx--- bluetooth bluetooth      2010-07-12 16:55 bluetoothd
drwx----- keystore keystore      2010-07-12 16:55 keystore
drwxrwx--- system system      2010-07-12 16:55 vpn
drwxrwxrwx root root      2010-07-23 15:34 widgets
drwxrwx--- wifi wifi      2010-07-12 16:55 wifi
```

# Case Study: App Framework



```
# ls -l /data/misc
drwxrwx--- bluetooth bluetooth      2010-07-12 16:55 bluetoothd
drwx----- keystore keystore      2010-07-12 16:55 keystore
drwxrwx--- system system          2010-07-12 16:55 vpn
drwxrwxrwx root root              2010-07-23 15:34 widgets
drwxrwx--- wifi wifi              2010-07-12 16:55 wifi
# ls -l /data/misc/webwidgets
drwxrwxrwx app_24 app_24            2010-07-23 15:39 chess
#
```

# Case Study: App Framework



# Lookout Mobile



- Lookout Mobile security app
  - Over 4 million users
  - Scanning, backup, lost device tracking, etc

# Lookout: World-Writable Files

- Lookout installs with a ***world-writable*** config file and database
  - Independently discovered by Tavis Ormandy
- Disable, lockout device, etc from any unprivileged app

```
# pwd
/data/data/com.lookout
# ls -l config.txt
-rw-rw-rw- app_32  app_32      5478 2011-01-28 18:18 config.txt
```

```
# pwd
/data/data/com.lookout/DB
# ls -l
-rw-rw-rw- app_32  app_32      177 2011-01-29 09:37 system.db
```



# Lookout: Owned by Tavis

- Tavis took it to the next level:
  - Backed up a custom shared lib, “liblookout.so” from a user-controlled directory
  - Restored into Lookout app's data/lib directory, overwriting legit

```
# pwd
/data/data/com.lookout/lib
# ls -l
-rwxr-xr-x system:system 564672 2010-11-22 11:50 liblookout.so
```

– Security app → less secure phone

# Application Security Wrapup

- Lack of guidance, standards, practices makes developers reinvent the wheel
  - Or just make them think they need to
- Neglecting the security lessons learned with “traditional” and web applications
  - Client-side trust
  - Access control issues
  - ...and all of the other “basic” problems and mistakes of yore

# Final Scorecard

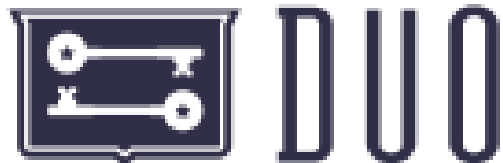
- TEAM JOCH vs. Android kernel?  
– TEAM JOCH!
- TEAM JOCH vs. Android platform?  
– TEAM JOCH!
- TEAM JOCH vs. Android apps?  
– TEAM JOCH!

# QUESTIONS?

Jon Oberheide

Duo Security

[jon@oberheide.org](mailto:jon@oberheide.org)



Zach Lanier

Intrepidus Group

[zach@n0where.org](mailto:zach@n0where.org)

