

Ch 4: Android



CNIT 128: Hacking Mobile Devices

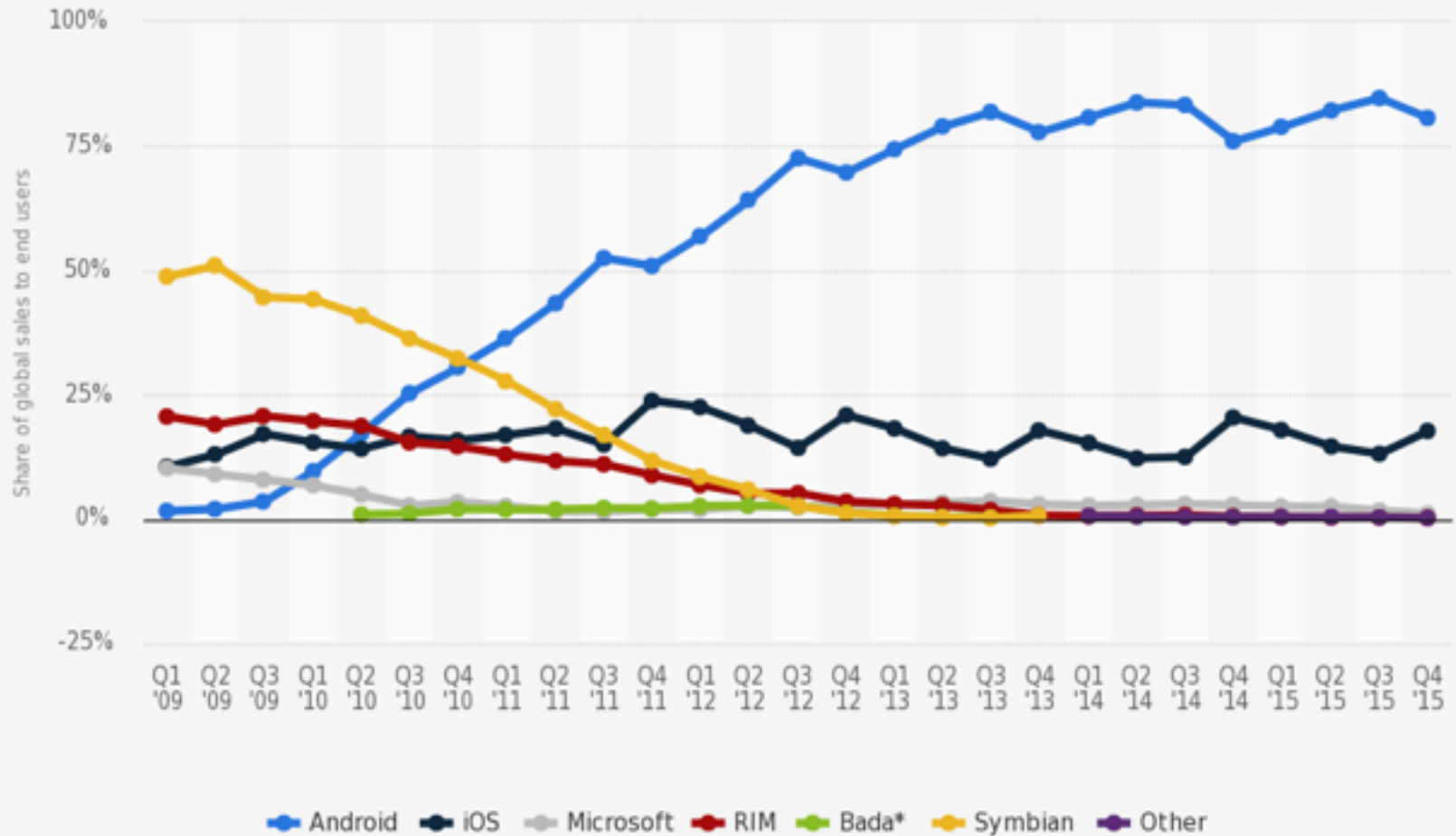
Updated 2-21-17

Android is #1

- 81.7% market share in 2016
 - Links Ch 4z75, 76

Operating System	4Q16 Units	4Q16 Market Share (%)	4Q15 Units	4Q15 Market Share (%)
Android	352,669.9	81.7	325,394.4	80.7
iOS	77,038.9	17.9	71,525.9	17.7
Windows	1,092.2	0.3	4,395.0	1.1
BlackBerry	207.9	0.0	906.9	0.2
Other OS	530.4	0.1	887.3	0.2
Total	431,539.3	100.0	403,109.4	100.0

Global market share held by the leading smartphone operating systems in sales to end users from 1st quarter 2009 to 4th quarter 2015



Source:
Gartner
© Statista 2016

Additional Information:
Worldwide, Gartner

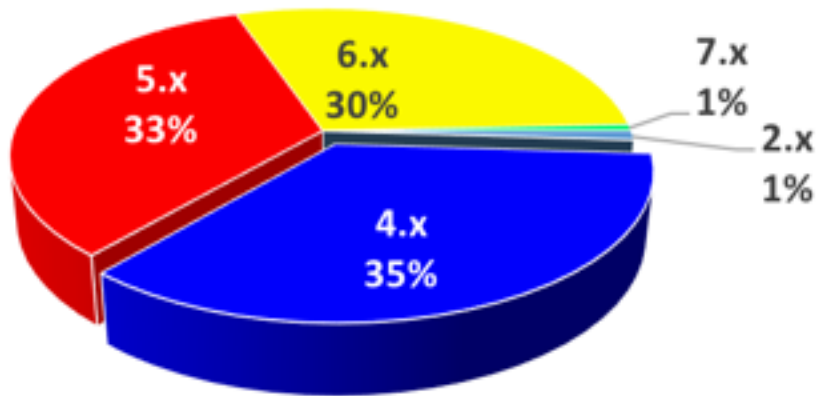
Open Source

- Android itself is open source (Link Ch 4b)
- But the Google apps included with most Android phones are closed-source
- Many device manufacturers and carriers modify Android
 - Closed-source device drivers and apps
 - Leads to fragmentation: two devices with the same hardware but different carriers can be running very different software
 - More total sales, many different Android devices

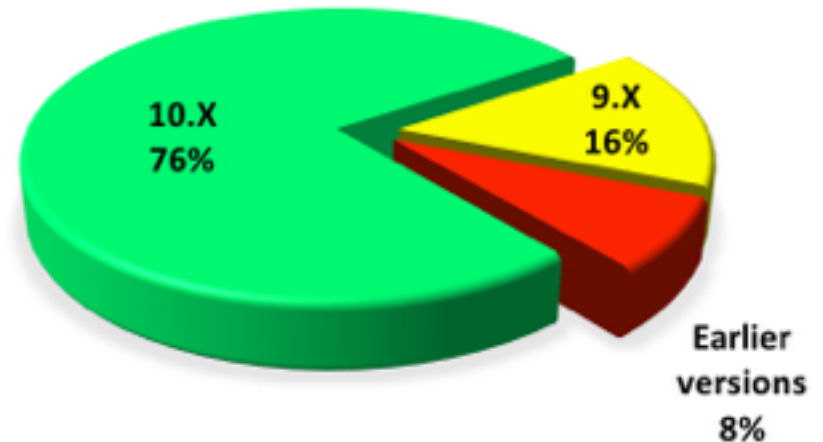
Fragmentation

- Updates are essential for security
- Very big problem for Android
 - Data from links Ch 1v, 1w

Android versions 1-9-17



iOS Versions 1-12-17



Android Architecture



Linux Kernel

- A way for applications to interact with devices
- Manages processes and memory
- Android versions prior to 4.0 used 2.6 Linux kernel (with modifications)
- Later versions based on 3.x or 4.x Linux Kernels
 - [Link Ch 4z77](#)

Linux Kernel Versions

Android Version		API Level	Linux Kernel in AOSP
1.5	Cupcake	3	2.6.27
1.6	Donut	4	2.6.29
2.0/1	Eclair	5-7	2.6.29
2.2.x	Froyo	8	2.6.32
2.3.x	Gingerbread	9, 10	2.6.35
3.x.x	Honeycomb	11-13	2.6.36
4.0.x	Ice Cream San	14, 15	3.0.1
4.1.x	Jelly Bean	16	3.0.31
4.2.x	Jelly Bean	17	3.4.0
4.3	Jelly Bean	18	3.4.39
4.4	Kit Kat	19, 20	3.10
5.x	Lollipop	21, 22	3.16.1
6.0	Marshmallow	23	3.18.10
7.0	Nougat	24	4.4.1
7.1	Nougat	25	4.4.1 (To be updated)

Libraries

- OpenGL (for 2D/3D graphics)
 - SQLite for databases
 - WebKit (for rendering Web pages)
 - Others
- Written in C/C++
 - Also Dalvik VM and core Java libraries
 - All this together is called the "Android Runtime" component

Application Framework

- A way for Android apps to access
 - Telephone functionality
 - Creating UI (User Interface) elements
 - GPS
 - File system
- Important in Android security model

Apps

- Usually Written in Java
 - Compiled to Dalvik bytecode using the Android Software Development Kit (SDK)
- Can also be written in C/C++
 - Using Native Development Kit (NDK)

Security Model

Permission Based

- Apps must be explicitly granted permission to take action
- Permissions enforced in two places
 - Kernel level
 - Application Framework level

Kernel Permissions

- Each app has a unique user ID
 - File ownership based on User ID
- Can only access the resources and functionality it has explicit permissions for
- This is "sandboxing"
 - Apps cannot access resources of other apps
 - Apps cannot access hardware components they have not been given permission to use

Viewing Users with ps

- System processes run as root

```
. . . . . sambowne Wed Feb 11 13:36:41
platform-tools $./adb shell
root@vbox86p:/ # ps
USER      PID    PPID   VSIZE  RSS      WCHAN    PC      NAME
root       1       0      716    484     c01b867c 0805a586 S /init
root       2       0       0       0     c0139c5e 00000000 S kthreadd
root       3       2       0       0     c0128180 00000000 S ksoftirqd/0
root       5       2       0       0     c01363be 00000000 S kworker/u:0
root       6       2       0       0     c015f8e2 00000000 S migration/0
root       7       2       0       0     c0135a7e 00000000 S khelper
root       8       2       0       0     c0195590 00000000 S sync_supers
root       9       2       0       0     c0196122 00000000 S bdi-default
root      10      2       0       0     c0135a7e 00000000 S kintegrityd
root      11      2       0       0     c0135a7e 00000000 S kblockd
root      12      2       0       0     c0135a7e 00000000 S ata_sff
```

Viewing Users with ps

- Apps run as u0_a1, u0_a2, etc.

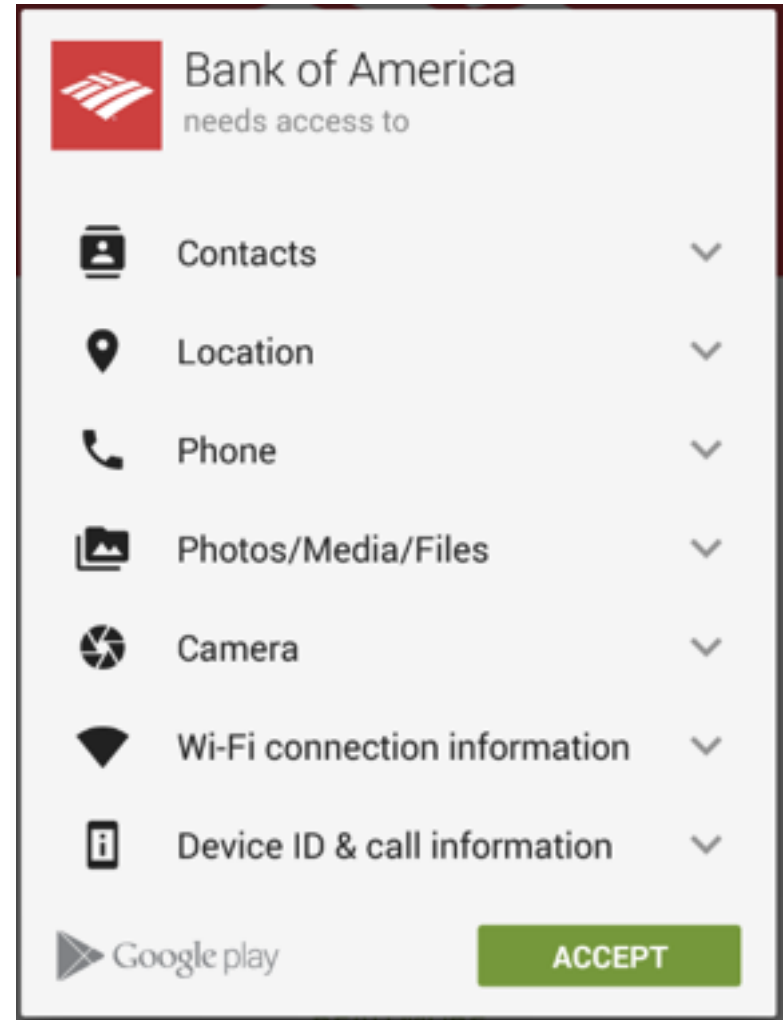
```
u0_a64    1639   149    539328 39060  ffffffff b75609eb S com.google.android.music:main
u0_a7     1741   149    518884 29160  ffffffff b75609eb S com.android.providers.calendar
u0_a0     1954   149    603020 59080  ffffffff b75609eb S android.process.acore
u0_a77    1974   149    537344 37296  ffffffff b75609eb S com.google.android.googlequicksearchbox
u0_a73    1991   149    518232 26120  ffffffff b75609eb S com.google.android.partnersetup
u0_a47    2010   149    517036 25096  ffffffff b75609eb S com.android.voicedialer
root      2045    60     6372   1248  c01b867c b75621e0 S logcat
u0_a30    2089   149    517024 25052  ffffffff b75609eb S com.android.musicfx
u0_a58    2106   149    522316 32232  ffffffff b75609eb S com.google.android.ears
u0_a82    2373   149    573128 70048  ffffffff b75609eb S com.infonow.bofa
root      2551    60     6732   1408  c01022d9 b74f5086 S /system/bin/sh
root      2557  2551   6904   1240  00000000 b757bbb6 R ps
root@vbox86p:/ # █
```

- Permissions for data directories

```
root@vbox86p:/data/data # ls -al | grep com.i
drwxr-x--x u0_a82    u0_a82                2015-02-11 13:28 com.infonow.bofa
drwxr-x--x u0_a86    u0_a86                2015-02-11 13:28 com.intuit.turbotax.mobile
```


Application Framework Access Control

- App must declare a permission for each component it accesses in its manifest file
 - AndroidManifest.xml
- Permissions are shown to the user at install time
 - User can chose whether to allow the permissions
 - If not, the app can't be installed
- Once installed, the app has only those permissions, such as `android.permission.INTERNET`



Many Permissions are Available

ice > Manifest.permission



Summary

Constants

String	ACCESS_CHECKIN_PROPERTIES	Allows read/write access to the "properties" table in the checkin database, to change values that get uploaded.
String	ACCESS_COARSE_LOCATION	Allows an app to access approximate location derived from network location sources such as cell towers and Wi-Fi.
String	ACCESS_FINE_LOCATION	Allows an app to access precise location from location sources such as GPS, cell towers, and Wi-Fi.
String	ACCESS_LOCATION_EXTRA_COMMANDS	Allows an application to access extra location provider commands
String	ACCESS_MOCK_LOCATION	Allows an application to create mock location providers for testing

- [Link Ch 4d](#)

Permission Categories

- Normal
 - Low-risk
 - Don't require explicit approval from users at install time
- Dangerous
 - Require explicit approval from users at install time

Permission Categories

- Signature
 - Defined by an application in its manifest
 - Functionality exposed by applications that declare this permission can only be accessed by other applications that were signed by the same certificate
- signatureOrSystem
 - Same as Signature, but applications installed on the /system partition can also access this functionality

App Signing

- All apps must be signed to be installed
- Android allows self-signed certificates
 - Developers can generate their own signing certificates
- The only security mechanisms that use signatures are the *signature* or *signatureOrSystem* permissions

Address Space Layout Randomization (ASLR)

- Added in Android 4.0
- Makes it more difficult to exploit memory corruption issues
- Randomizes locations of key memory sections
 - Such as stack and heap
- In 4.0, several locations, including the heap were not randomized
- Android 4.1 has full ASLR

No eXecute (NX) Bit

- Added in Android 2.3
- Allows you to set the heap and stack as nonexecutable
 - Helps prevent memory corruption attacks

Application Components

Four Components

- Activities, Content providers, Broadcast receivers, Services
 - Entry points: ways the user or another app on the same device can enter
- The more components that are exported, the larger the attack surface
 - Component header will contain this line:
`android:exported="true"`

Intents

- Applications primarily use *intents*
 - Asynchronous messages
- To perform interprocess, or intercomponent, communication

Activities

- Define a single screen of an applications user interface
- Android promotes reusability of activities
- This saves developers time and work
- Increases the attack surface of an app

Content Providers

- Exposes ability to query, insert, update, or delete application-specific data
 - to other apps and internal components
- App might store data in SQLite database or a flat file
- Calling component won't know what storage method is used
- Poorly written content providers may expose data or be vulnerable to SQL injection

Broadcast Receivers

- Respond to broadcast intents
- Apps should not blindly trust data from broadcast intents
 - It could be from a hostile app or a remote system

Services

- Run in the background
- Perform lengthy operations
- Started by another component when it sends an intent
- Services should not blindly trust the data contained within the intent

Data Storage

- Internal storage in nonvolatile memory (NAND flash)
 - Private to one app
- External storage on an SD card
 - Also NAND flash, but removable
 - Available to all apps
 - Image from amazon.com



File Types

- Apps are free to create any type of file, but the API comes with support for
 - SQLite databases
 - Shared preference files in an XML-based format
- SQL injection attacks are a risk, via intents or other input

NFC (Near Field Communication)

- Radio communication, including
 - NFC tags
 - Some RFID tag protocols
 - Contactless smart cards
 - Mobile devices
- Short range: a few cm
 - image from newegg.com

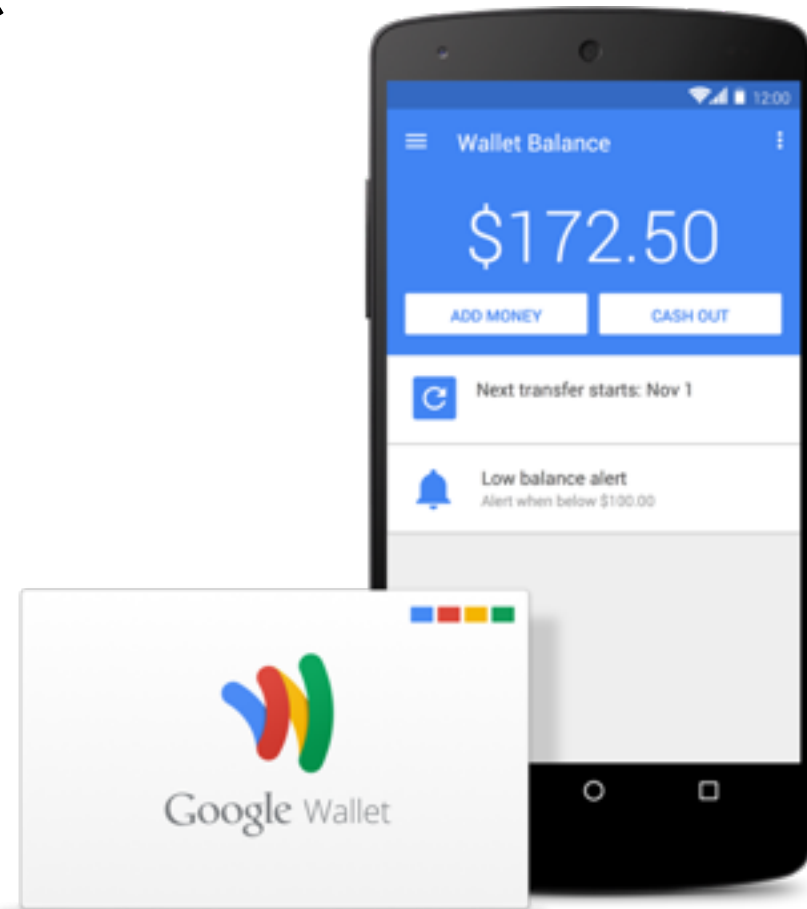


NFC in Android

- Introduced in 2010 with Gingerbread
- Expanded in 2.3.3 to support a variety of NFC formats
- Card Emulation mode came In 2.3.4
 - An NFC reader can read data from the Secure Element (SE) within the phone
 - Card emulation is not exposed via the Android SDK
 - Only Google or carriers have this capability

Google Wallet

- Lets you use your phone for a credit card
- Released with Android 2.3.4



NFC Peer-to-Peer

- Allows two NFC-enabled devices to communicate directly
- Android Beam allows users to share data by tapping their devices together
- NFC tags are also used in advertisements

Android Development

SDK (Software Development Kit)

- Allows developers to build and debug Android apps
 - Runs on Windows, Mac OS X, or Linux
- In Dec., 2014, Google released Android Studio
 - A full IDE (Integrated Development Environment)
 - Links Ch 4e, 4f

Android Emulator

- Helps developers test apps without an actual mobile device
- Simulates common hardware
 - ARMv5 CPU
 - SIM card
 - Flash memory partitions



Figure 4-3 The Android emulator

Value of the Emulator

- Allows developers and researchers to test Android apps quickly in different versions of Android
- Drawbacks
 - Android Virtual Device (AVD) cannot send or receive phone calls or SMS messages
 - But it can emulate them, and send them to other AVDs
- Can define an HTTP/HTTPS proxy to intercept and manipulate Web traffic

Android Debug Bridge

```
C:\>adb devices
List of devices attached
emulator-5554      device
c11f5f53          device

C:\>adb -s c11f5f53 shell
root@android:/ # _
```

- Command-line tool
- Allows you to communicate with a mobile device via a USB cable or an SVD running within an emulator
- Connects to device's daemon running on TCP port 5037

Useful ADB Commands

- push
 - Copies a file from your computer to the mobile device
- pull
 - Copies a file from the mobile device to your computer
- logcat
 - Shows logging information on the console
 - Useful to see if an app or the OS is logging sensitive information

Useful ADB Commands

- `install`
 - Copies an application package file (APK) to the mobile device and installs the app
 - Useful for side-loading apps (so you don't have to use Google Play)
- `shell`
 - Starts a remote shell on the mobile device
 - Allows you to execute arbitrary commands

END OF PART 1

Rooting

Root

- The root user can directly access system resources
- Bypassing the permissions checks normally required
- Giving the app that runs as root full control over the device and other apps running on it

GingerBreak (CVE-2011-1823)

- Gains root on
 - Many Gingerbread (Android 2.3.x) devices
 - Some Froyo (2.2.x)
 - Some Honeycomb (3.x.x)

How Gingerbreak Works

- Vulnerability in volume manager daemon
 - /system/bin/vold
- Method
 - DirectVolume::handlePartitionAdded**
 - Sets an array index using an integer passed in
 - Does not check for negative integers
 - Using negative values, exploit can access arbitrary memory locations

How Gingerbreak Works

- Exploit writes to **vold's** global offset table (GOT)
- Overwrites several function calls
 - such as `strcmp()` and `atoi()`
- With calls to `system()`
- Then another call to **vold** to execute an app via `system()`, with vold's elevated privileges

GingerBreak

- Packaged into several rooting tools
 - Such as SuperOneClick
- And some one-click rooting APKs
 - Malicious app could include GingerBreak as a way to gain elevated privileges on a device

GingerBreak Countermeasures

- Update your device
 - GingerBreak exploit was fixed in Android 2.3.4
- But some manufacturers/carriers don't allow updates
- Antivirus apps should detect the presence of GingerBreak
 - An alternative

Ice Cream Sandwich

init chmod/chown Vuln

- Vuln introduced with Ice Cream Sandwich (4.0.x) in **init**
- If the **init.rc** script has an entry like this
 - **mkdir /data/local/tmp 0771 shell shell**
- Even if the **mkdir** failed, **init** would set the ownership and permissions for the **shell** user
 - The **ADB** user

Symlink

- If **/data/local** is writable by the shell user,
- Creating a symlink to **/system** there allows the ADB user read/write access to the **/system** partition
- ADB user can add files to **/system**, such as **su**
- Thus gaining root access to a device

Ice Cream Sandwich init chmod/chown Countermeasures

- Keep devices updated
- Make sure Android debugging is turned off
- And keep device locked, so attacker can't turn it on

Decompiling and Disassembly

Static Analysis

- Source code is generally kept confidential by app developers
- A binary, compiled app can be analyzed by disassembling or decompiling them, into
 - Smali assembly code (used by Dalvik VM), or
 - Java code

Project 9: Decompiling and Trojaning an Android App with Smali Code (15 points)

```
. . . . . sambowne Sun Feb 01 07:10:02
~ $cd Downloads/

. . . . . sambowne Sun Feb 01 07:10:13
Downloads $java -jar apktool_2.0.0rc3.jar d app-release.apk
I: Using Apktool 2.0.0-RC3 on app-release.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /Users/sambowne/Library/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...

. . . . . sambowne Sun Feb 01 07:10:24
Downloads $
```

Java v. Smali Code

```
public int performLogin(String server, String port, String username, String password)
    throws JSONException, IOException, HttpException {
    // First perform the RESTful operation
    String protocol = mHttpsMode ? "https://" : "http://";
    String url = protocol + server + ":" + port + "/login";
    Map<String, String> parameters = new HashMap<>();
    parameters.put("username", username);
    parameters.put("password", password);
```

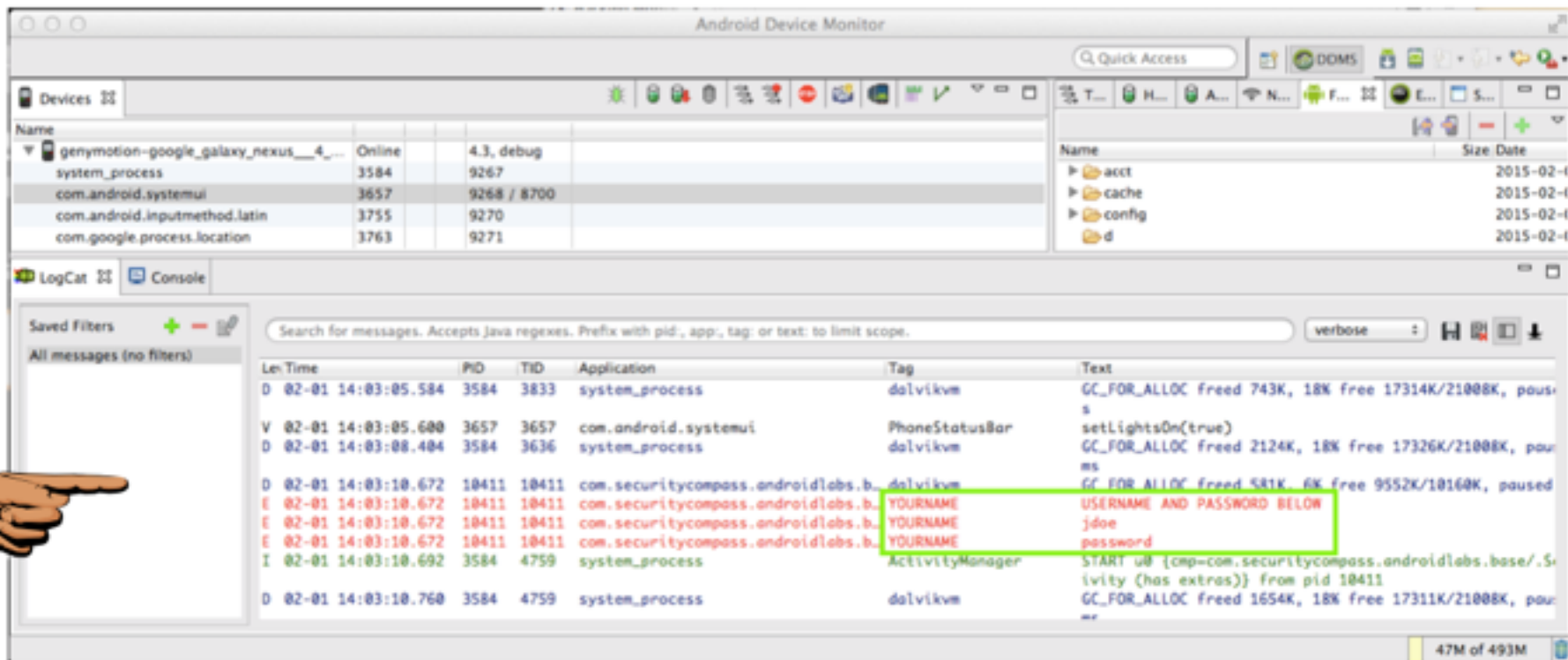
```
.method public performLogin(Ljava/lang/String;Ljava/lang/String;Ljava/lang/String;Ljava/lang/
    .locals 10
    .param p1, "server"    # Ljava/lang/String;
    .param p2, "port"     # Ljava/lang/String;
    .param p3, "username"  # Ljava/lang/String;
    .param p4, "password"  # Ljava/lang/String;
    .annotation system Ldalvik/annotation/Throws;
        value = {
            Lorg/json/JSONException;,
            Ljava/io/IOException;,
            Lcom/securitycompass/androidlabs/base/HttpException;
        }
    .end annotation
```

Building & Signing an App

```
java -jar apktool_2.0.0rc3.jar b app-release
```

```
jarsigner -verbose -keystore ~/Box\ Sync\ website\ 128\ proj\ p9cert.jks app-  
release\ dist\ app-release.apk proj9key
```

Monitoring the Log



The screenshot displays the Android Device Monitor (ADM) interface. The top section shows a list of devices and processes. The bottom section shows the LogCat window with a search bar and a list of log messages. A hand icon points to the LogCat window.

Devices Table:

Name	Online	API Level	Architecture
genymotion-google_galaxy_nexus_4...	Online	4.3, debug	
system_process	3584	9267	
com.android.systemui	3657	9268 / 8700	
com.android.inputmethod.latin	3755	9270	
com.google.process.location	3763	9271	

LogCat Window:

Search for messages. Accepts Java regexes. Prefix with pid, app., tag or text to limit scope. verbose

Len	Time	PID	TID	Application	Tag	Text
D	02-01 14:03:05.584	3584	3833	system_process	dalvikvm	GC_FOR_ALLOC freed 743K, 18K free 17314K/21008K, paus...
V	02-01 14:03:05.600	3657	3657	com.android.systemui	PhoneStatusBar	setlightsOn(true)
D	02-01 14:03:08.484	3584	3636	system_process	dalvikvm	GC_FOR_ALLOC freed 2124K, 18K free 17326K/21008K, paus...
D	02-01 14:03:10.672	10411	10411	com.securitycompass.androidlabs.b...	dalvikvm	GC FOR ALLOC freed 581K, 6K free 9552K/10160K, paused
E	02-01 14:03:10.672	10411	10411	com.securitycompass.androidlabs.b...	YOURNAME	USERNAME AND PASSWORD BELOW
E	02-01 14:03:10.672	10411	10411	com.securitycompass.androidlabs.b...	YOURNAME	YOURNAME
E	02-01 14:03:10.672	10411	10411	com.securitycompass.androidlabs.b...	YOURNAME	password
I	02-01 14:03:10.692	3584	4759	system_process	ActivityManager	START u0 {cmp=com.securitycompass.androidlabs.base/.S...
D	02-01 14:03:10.760	3584	4759	system_process	dalvikvm	GC_FOR_ALLOC freed 1654K, 18K free 17311K/21008K, paus...

47M of 493M

Attacks via Decompiling and Disassembly

- Insert Trojan code, like keyloggers
- Find encryption methods & keys
- Change variables to bypass client-side authentication or input validation
- Cheat at games

Dancing with Dalvik

THOMAS RICHARDS

Getting to dalvik

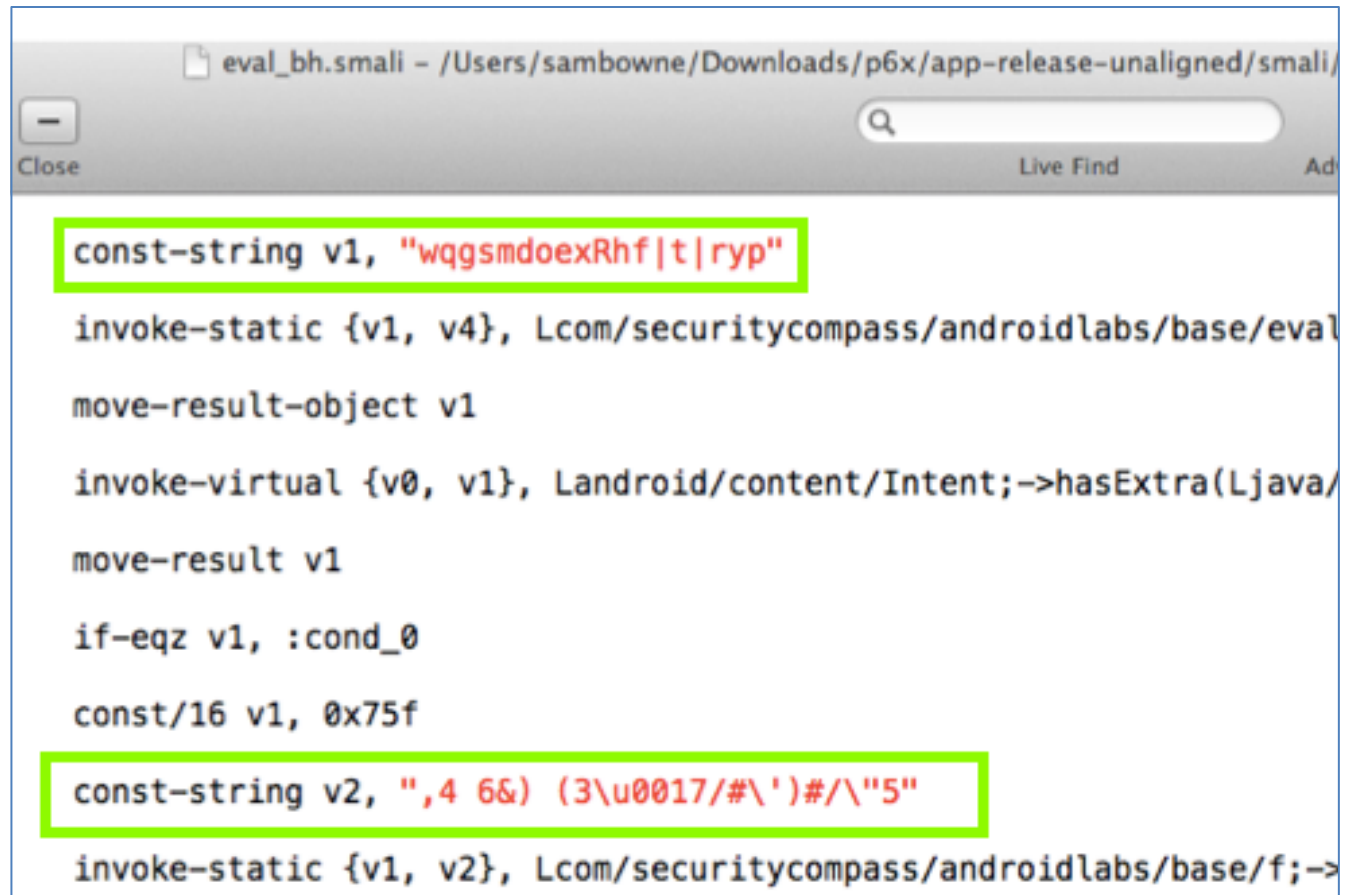
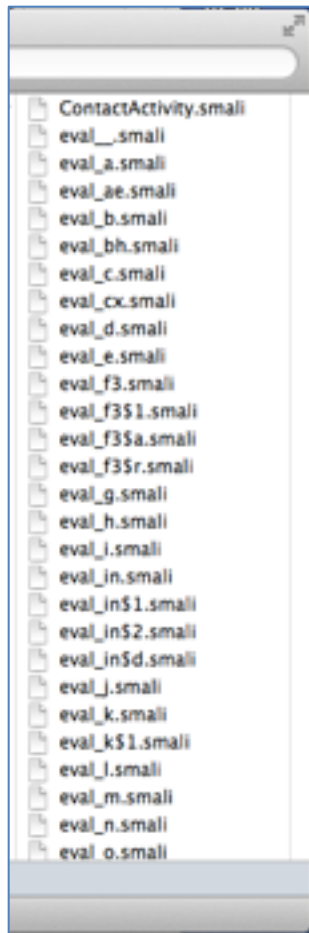
- Android apps are traditionally written in Java
- Compiled into Java bytecode then converted to Dalvik bytecode
- Java class files converted into .dex

- [Link Ch 4z43](#)

Decompiling, Disassembly, and Repackaging Countermeasures

- Every binary can be reverse-engineered
 - Given enough time and effort
- Never store secrets on the client-side
- Never rely on client-side authentication or client-side validation
- Obfuscate source code
 - ProGuard (free) or Arxan (commercial)

Dash0 - Powerful Obfuscator



A screenshot of a code editor window showing obfuscated SMALI code. The code is displayed in a monospaced font. Two lines of code are highlighted with a green box:

```
const-string v1, "wqgsmdoexRhft|ryp"  
invoke-static {v1, v4}, Lcom/securitycompass/androidlabs/base/eval  
move-result-object v1  
invoke-virtual {v0, v1}, Landroid/content/Intent; ->hasExtra(Ljava/  
move-result v1  
if-eqz v1, :cond_0  
const/16 v1, 0x75f  
const-string v2, ",4 6&) (3\u0017/#\')#/\\"5"  
invoke-static {v1, v2}, Lcom/securitycompass/androidlabs/base/f;->
```


All Strings Concealed

- BUT it costs \$2000

```
. . . . . sambowne Sun Feb 15 06:58:48
app-release-unaligned $cd smali/

. . . . . sambowne Sun Feb 15 06:58:50
smali $grep -ir password .

. . . . . sambowne Sun Feb 15 06:59:03
smali $grep -ir login .

. . . . . sambowne Sun Feb 15 06:59:12
smali $
```

Intercepting Network Traffic

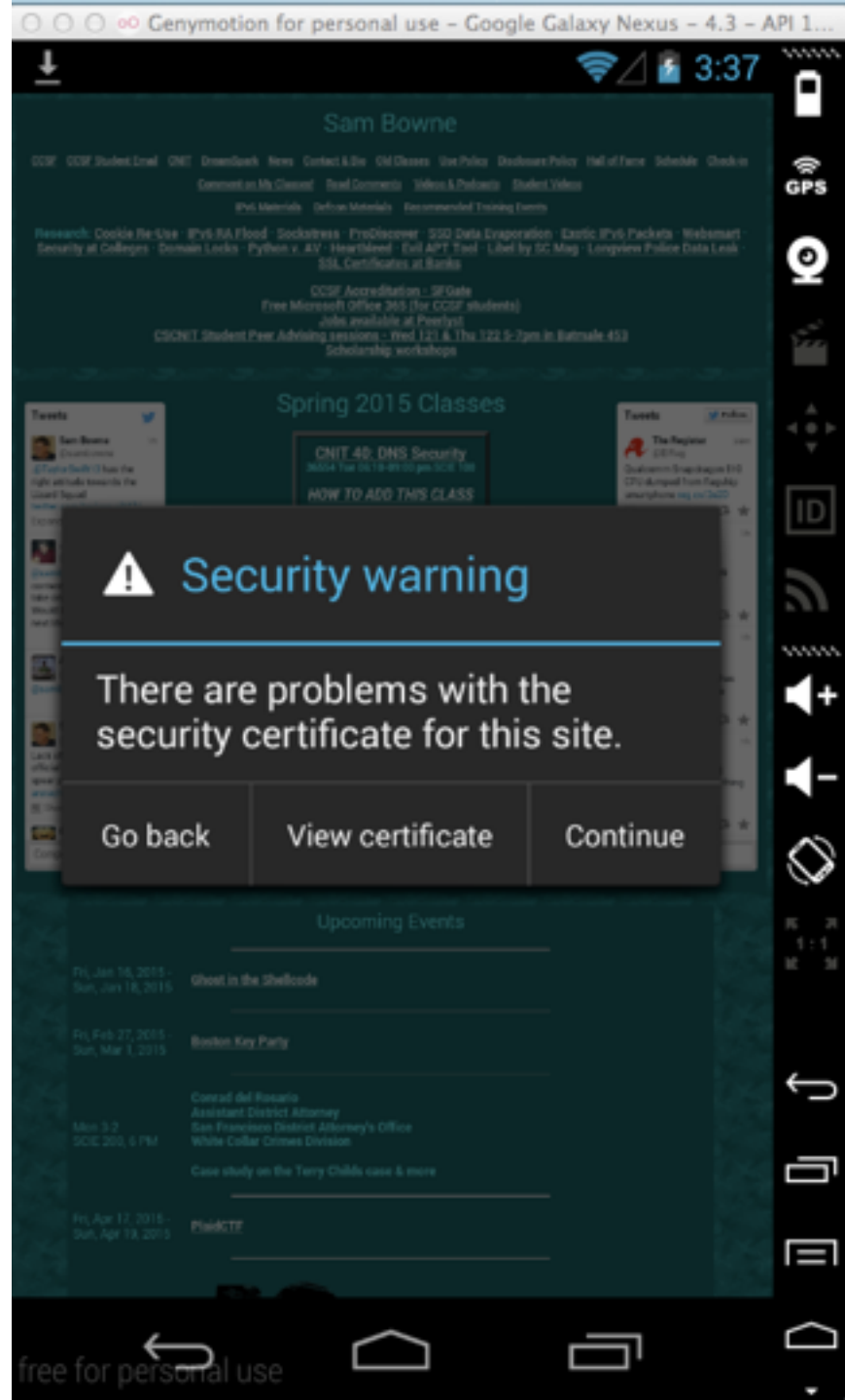
"Man in the Middle" (MITM)
Attacks

HTTP Traffic

- Trivial to intercept
- No protections
- Any hacker in a coffeehouse can do it with
 - ARP poisoning
 - A rogue access point like the WiFi Pineapple
- Very difficult for the end user to detect

HTTPS

- Specifically designed to prevent MITM attacks
- Only very powerful adversaries like nation-states or big corporations can get real Certificate Authority certificates
- Without one, targets will see a warning like this

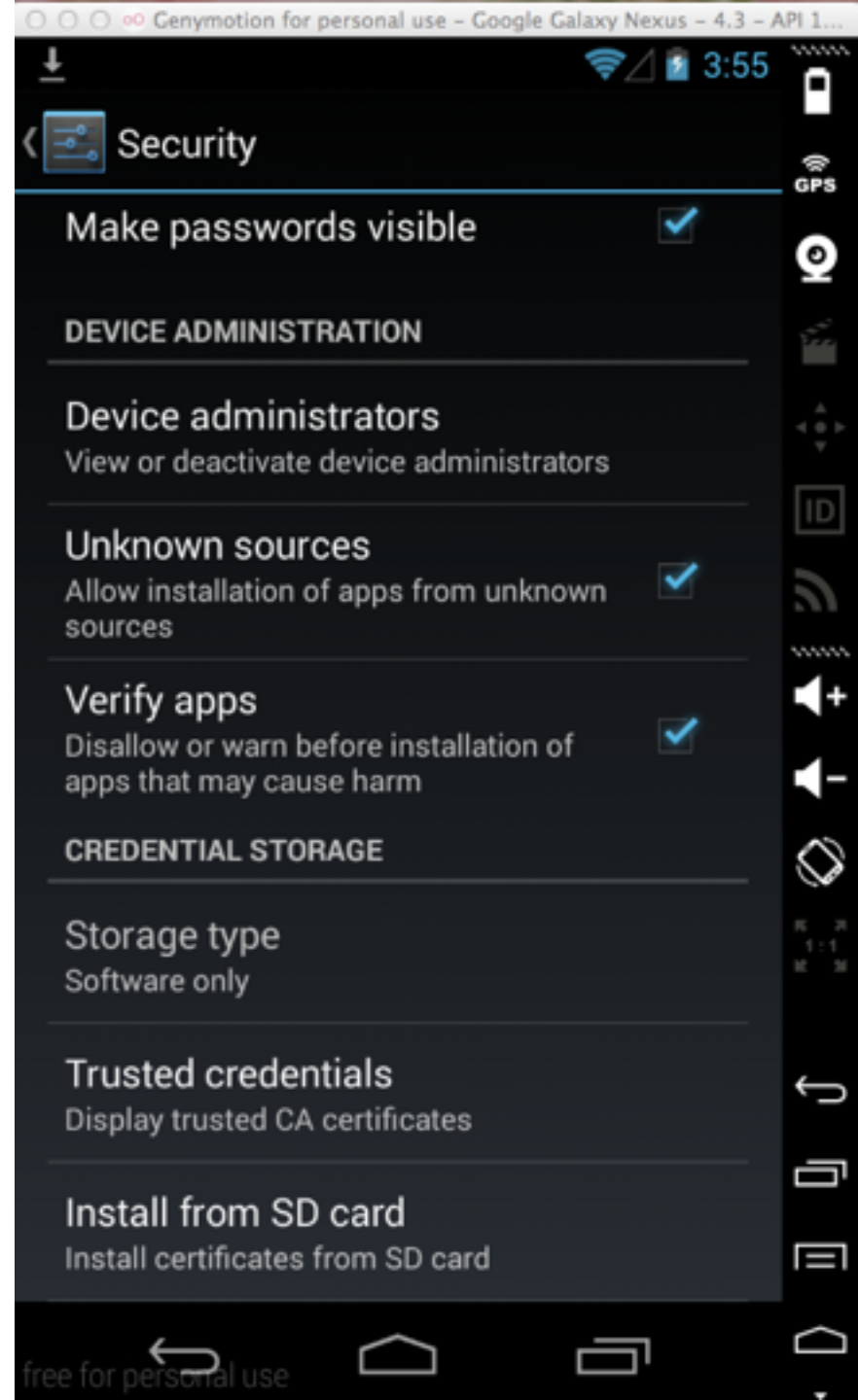


Problems

- End-users are not often aware of the difference between HTTP and HTTPS
- The mobile app may not display any information about the connection
- Mobile apps don't always verify all the security features in HTTPS, so users get less protection

Auditing HTTPS Traffic

- If you want to see HTTPS traffic, you must import your proxy's certificate into your test mobile device
- With "Install from SD Card" option in Settings



Manipulating Network Traffic Countermeasures

- Don't disable certificate verification and validation
 - Often done during development and debugging for convenience
 - May be left that way when app is distributed
- Example of trust chain



Certificate Pinning

- Adds another layer of verification to certificates
- Browser knows which CA should validating sites
 - Either by consulting a whitelist maintained by Google, or
 - Remembering which CA was used the first time the site was used
- Warns user if CA changes
 - Link Ch 4z62

Don't Trust the Client

- Any data from the client may be malicious
- Perform strict input validation and output encoding on the server side

Intent-Based Attacks

Intents

- Primary inter-process communication (IPC) method used by Android apps
- Intents can start internal components, or pass data to them, or go to other apps
- Those are called "External Intents"
- Links Ch 4z62-64

Code to Send an SMS Intent

```
Intent smsIntent = new Intent(Intent.ACTION_VIEW);
smsIntent.setType("vnd.android-dir/mms-sms");
smsIntent.putExtra("address", "12125551212");
smsIntent.putExtra("sms_body", "Body of Message");
startActivity(smsIntent);
```

- Receiving app needs an "Intent Filter" like
 - android.provider.Telephony.SMS_RECEIVED
- If an app is not running, receiving an Intent starts it
- Malicious intents can launch other apps, and inject data into them

Command Injection Example

- A test app saves files on the SD card, with this Intent Filter

```
<service android:name="FileCreatorService">
  <intent-filter>
    <action android:name="com.test.CreateFile" />
  </intent-filter>
</service>
```

Method to Create File

```
protected void onHandleIntent(Intent intent) {
    String input = intent.getStringExtra("fileName");
    String s = Environment.getExternalStorageDirectory() + "/" + input;
    try {
        Runtime.getRuntime().exec(new String[]{"sh", "-c", "touch " + s});
    } catch (IOException e) {
    }
}
```

- Executes a shell command, injecting the filename from the calling app

Command Injection

- Suppose the calling app sends a filename of:
 - "MyFile.txt; cat /data/data/com.test/secrets.xml > /sdcard/secrets.xml"
- The exploit will expose any secrets the test app can access

Command Injection Countermeasures

- Don't make Intents "exported" unless they are really needed by other apps
 - Leave them internal to their own app
- Perform input validation on all data received from intents
 - Block special characters like ";" and ">" in a filename

```
if(input.matches("^.*[^a-zA-Z0-9].*$" && input!=null){
    String s = Environment.getExternalStorageDirectory() + "/" +
input + ".txt";
}
```


Command Injection Countermeasures

- Custom permissions will warn the user that an App has permissions when it is installed
 - Not much use, in practice
- Signature-level permissions
 - Require any app that wants to send intents to be signed with the same key as the receiving app
 - That is, from the same company
 - This can be defeated by re-signing the app

NFC-Based Attacks (Near Field Communication)



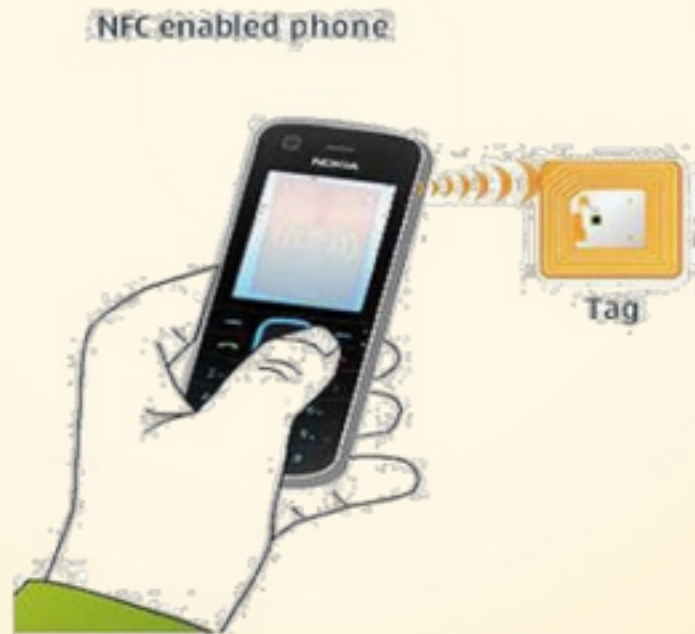
Image from techmoran.com

Malicious NFC Tags

- An NFC tag can contain a URL pointing to a malicious site
 - Such as a WebKit exploit
- Tags are cheap to buy online
- Can be written to by any NFC-enabled phone

NFC Reader/Writer Mode

- **Description from Android Developers:**
 - *“Reader/writer mode, allowing the NFC device to read and/or write passive NFC tags and stickers.”*



- From www.slideshare.net/chunkai1312

NFC Attacks

- Attackers can make poster with malicious tags
- Or replace tags on existing posters
- Or overwrite NFC tags in place
 - If the tag was not properly write-protected
- Tag could send user to Google Play
- Tag could directly attack an app that uses NFC with unexpected input

Malicious NFC Tag Countermeasures

- Keep NFC disabled when you aren't using it
- Unfortunately, there's no way to tell if a tag is malicious without reading it
 - Unless there is evidence of tampering
- Developers need to validate data from NFC
- Tags must be write-protected

Attacking Apps via NFC Events

- Android allows apps to create fake NFC Intents
 - Creating a fake NFC tag
- There's an NFCDemo App
- So a vulnerable NFC-using app could be exploited even without an NFC tag

NFC Event Countermeasures

- Validation of data received from NFC tags
- Custom permissions, or making the Activity private, won't work here
 - App must receive NFC intents from an external source

Information Leakage

Leakage via Internal Files

- Each app has a unique User Identifier (UID) and Group Identifier (GID)
 - App runs as that user
- But an app can create a file set to
 - `MODE_WORLD_READABLE` or
 - `MODE_WORLD_WRITEABLE`
- That file would be visible to all apps on the device, and could be sent to the Internet by any app with `INTERNET` permission

Android SQLite Journal Information Disclosure (CVE-2011-3901)

- SQLite engine created its rollback journal as globally readable and writeable
- Normally deleted after each transaction
- But setting improper permissions allows hostile apps on the same device to read
 - Personal info., session tokens, URL history, etc.

LinkedIn App's Rollback Journal

22 70 69 63 74 75 72 65 22 2C 22 66 69 65 6C 64 4B 65 79 22 3A 22 70 69	" picture", "fieldKey": "pi
63 74 75 72 65 22 2C 22 66 69 65 6C 64 44 69 73 70 6C 61 79 54 65 78 74	icture", "fieldDisplayText
22 3A 22 50 69 63 74 75 72 65 22 2C 22 74 54 79 70 65 22 3A 22 70 65 74	:"Picture", "tType": "pet
33 22 7D 5D 2C 22 64 65 74 61 69 6C 22 3A 5B 7B 22 66 69 65 6C 64 22 3A	3"}], "detail": [{"field":
22 73 75 6D 6D 61 72 79 22 2C 22 66 69 65 6C 64 4B 65 79 22 3A 22 73 75	"summary", "fieldKey": "su
6D 6D 61 72 79 22 2C 22 66 69 65 6C 64 44 69 73 70 6C 61 79 54 65 78 74	mmary", "fieldDisplayText
22 3A 22 53 75 6D 6D 61 72 79 22 2C 22 6D 69 6E 4C 65 6E 67 74 68 22 3A	:"Summary", "minLength":
31 2C 22 6D 61 78 4C 65 6E 67 74 68 22 3A 32 30 30 30 2C 22 74 54 79 70	1, "maxLength":2000, "tTyp
65 22 3A 22 70 65 74 32 22 7D 5D 7D 7B 22 66 6F 72 77 61 72 64 22 3A 7B	e": "pet2"}]} {"forward": {
22 73 75 62 6A 65 63 74 22 3A 22 54 61 6B 65 20 61 20 6C 6F 6F 6B 20 61	"subject": "Take a look a
74 20 6D 79 20 4C 69 6E 6B 65 64 49 6E 20 50 72 6F 66 69 6C 65 22 2C 22	t my LinkedIn Profile", "
62 6F 64 79 22 3A 22 49 20 74 68 6F 75 67 68 74 20 79 6F 75 20 6D 69 67	body": "I thought you mig
68 74 20 66 69 6E 64 20 6D 79 20 4C 69 6E 6B 65 64 49 6E 20 70 72 6F 66	ht find my LinkedIn prof
69 6C 65 20 69 6E 74 65 72 65 73 74 69 6E 67 3A 5C 6E 5C 6E 4E 65 69 6C	ile interesting:\n\nNeil
20 46 61 6B 65 6D 61 6E 5C 6E 43 6F 6D 70 75 74 65 72 20 48 61 63 6B 65	Fakeman\nComputer Hacke
72 20 61 74 20 53 6F 6D 65 20 52 61 6E 64 6F 6D 20 42 61 6E 6B 5C 6E 68	r at Some Random Bank\nh
74 74 70 73 3A 2F 2F 77 77 77 2E 6C 69 6E 6B 65 64 69 6E 2E 63 6F 6D 2F	ttps://www.linkedin.com/
70 72 6F 66 69 6C 65 2F 76 69 65 77 3F 69 64 3D 32 30 32 37 36 30 35 35	profile/view?id=20276055
33 22 7D 7D B4 1C F2 65 00 00 00 06 0D 00 00 00 02 03 E2 00 03 EE 03 E2	3"}]} ' òe - 7l& lfl&

Figure 4-14 Part of the SQLite journal file for the LinkedIn application

SQLite Journal Information Disclosure Countermeasures

- Install latest patches
- App developers should avoid creating world-readable or world-writable files



[Updated] Exclusive: Vulnerability In Skype For Android Is Exposing Your Name, Phone Number, Chat Logs, And A Lot More

88



Justin Case

Apr 14, 2011

g+ 5

465

f 224

Total Shares 694

- Skype App stored personal data in world-readable files
 - Link Ch 4z65

Leakage via External Storage

- Files stored on the SD card
 - Or the emulated SD card
- Are globally readable and writeable to every app on the device
- Apps should only store data they want to share on external storage

NESSUS ANDROID APP - stores login info in plain text

From: seclists () nospam jshipp com
Date: Sat, 21 Jul 2012 11:30:37 -0500

Nessus app for android
version 1.0.1

app allows user to save nessus server info
IP/username/password.

app saves this info to
/sdcard/servers.id

this file can be viewed with notepad and password is right there in plain text. this means any app on the system can see that info and possibly transmit it to an attacker.

- App seems to never have been updated, and later removed from the Google Play Store
 - Links Ch 4z66-67

Nessus Information Disclosure Countermeasures

- Credentials should not be stored in plaintext
- They should be encrypted
 - One good system is AES
 - Key from Password-Based Key Derivation Function 2 (PBKDF2)
 - Using a password from the user and a device-specific salt

Information Leakage via Logs

- Every app can read the logs, if it requests the `android.permission.READ_LOGS` permission at install time
- Some developers don't realize this and put sensitive information into the logs

CVE-ID	
CVE-2012-2980	Learn more at National Vulnerability Database (NVD) <ul style="list-style-type: none">• Severity Rating• Fix Information• Vulnerable Software Versions• SCAP Mappings
Description	
<p>The Samsung and HTC onTouchEvent method implementation for Android on the T-Mobile myTouch 3G Slide, HTC Merge, Sprint EVO Shift 4G, HTC ChaCha, AT&T Status, HTC Desire Z, T-Mobile G2, T-Mobile myTouch 4G Slide, and Samsung Galaxy S stores touch coordinates in the dmesg buffer, which allows remote attackers to obtain sensitive information via a crafted application, as demonstrated by PIN numbers, telephone numbers, and text messages.</p>	

- Link Ch 4z68

Facebook SDK Information Disclosure in the Log

Daily Archives: April 10, 2012

Discovering a Major Security Hole in Facebook's Android SDK

```
https://m.facebook.com/dialog/permissions.request?_path=permissions.request&app_id=318188501552052&red
9.171.234.32 (family 2, proto 6)
k.com: 23.14.34.110 (family 2, proto 6)
https://m.facebook.com/dialog/permissions.request?refid=0
aph.facebook.com/me?format=json&fields=id&access_token=AAAEhZAAibS7QBAlH3JCxpRDxb2cABRjXyik2raKjSaryopn
69.171.234.66 (family 2, proto 6)
```



In plain text, I could see the entire access token that had just been granted after logging in, encoded into a URL.

- Facebook promptly patched it
 - Link Ch 4z69

Facebook SDK Information Disclosure Countermeasures

- Update Facebook SDK to latest version
- App developers should not log any sensitive information

Information Leakage via Insecure Components

CVE-ID

CVE-2011-4872

[Learn more at National Vulnerability Database \(NVD\)](#)

• Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings

Description

Multiple HTC Android devices including Desire HD FRG83D and GRI40, Glacier FRG83, Droid Incredible FRF91, Thunderbolt 4G FRG83D, Sensation Z710e GRI40, Sensation 4G GRI40, Desire S GRI40, EVO 3D GRI40, and EVO 4G GRI40 allow remote attackers to obtain 802.1X Wi-Fi credentials and SSID via a crafted application that uses the `android.permission.ACCESS_WIFI_STATE` permission to call the `toString` method on the `WifiConfiguration` class.

- Link Ch 4z70

content:// URI Scheme Leaks Contents of SD Card

CVE-ID	
CVE-2010-4804	Learn more at National Vulnerability Database (NVD) • Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings
Description	
The Android browser in Android before 2.3.4 allows remote attackers to obtain SD card contents via crafted content:// URIs, related to (1) BrowserActivity.java and (2) BrowserSettings.java in com/android/browser/.	

- Link Ch 4z71

content:// URI Scheme Information Disclosure Countermeasures

- Patched poorly in 2.3, and re-patched in 2.3.4
- End user can do these things:
 - Use a different browser, such as Opera
 - Disable JavaScript in the Android browser (impractical)
 - Unmount the SD card (impractical because many apps need it)

General Mitigation v. Information Leakage

- Logs
 - Don't log sensitive info.
- Files, shared preferences, and SQLite databases
 - Don't store sensitive info. in plaintext
 - Don't create globally readable or writeable files
 - Don't store sensitive info. on SD card without encrypting it properly

General Mitigation v. Information Leakage

- WebKit (WebView)
 - WebView is the Android class that displays Web pages in WebKit
 - Apps should clear the cache periodically if it's used to view sensitive websites
 - App can disable caching

General Mitigation v. Information Leakage

- WebKit (WebView) continued
 - WebKit stores sensitive data in the app's data directory, including
 - Previously entered form data
 - HTTP authentication credentials
 - Cookies
 - On a non-rooted device, other apps won't be able to steal data from the WebKit data store
 - But if a phone is stolen and rooted, that data can be taken

General Mitigation v. Information Leakage

- Inter-Process Communication (IPC)
 - Don't expose sensitive info. via broadcast receivers, activities, and services exposed to other apps
 - Don't send sensitive data in intents to other processes
 - Make components nonexportable

General Mitigation v. Information Leakage

- Networking
 - Don't use network sockets for IPC
 - Use TLS to transmit sensitive data
 - CarrierIQ did this (next slide)
 - [Link Ch 4z72](#)

HTC IQRD Android Permission Leakage

On December 22th, VSR identified a vulnerability in IQRD. The IQRD service listens locally on a TCP socket bound to port 2479. This socket is intended to allow the Carrier IQ service to request device-specific functionality from IQRD. Unfortunately, there is no restriction or validation on which applications may request services using this socket. As a result, any application with the android.permission.INTERNET permission may connect to this socket and send specially crafted messages in order to perform potentially malicious actions.

In particular, it is possible for malicious applications to:

1. Trigger UI popup messages
2. Generate tones
3. Send arbitrary outbound SMS messages that do not appear in a user's outbox, facilitating toll fraud
4. Retrieve a user's Network Access Identifier (NAI) and corresponding password, potentially allowing rogue devices to impersonate the user on a CDMA network