

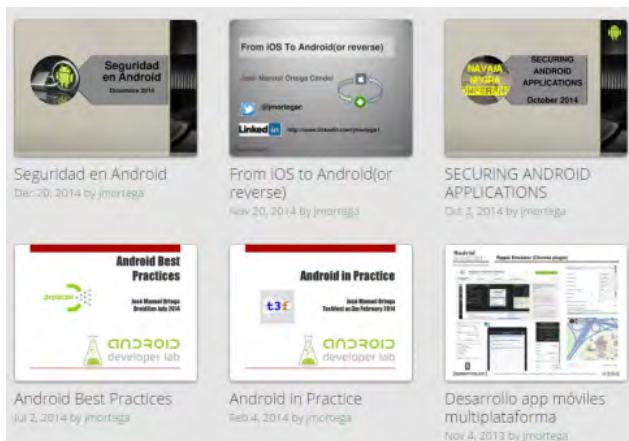
Security testing in mobile applications

José Manuel Ortega Candel

About me

- Centers Technician at Everis
- Computer engineer by Alicante University
- Frontend and backend developer in Java/J2EE
- Speaker site with some presentations in mobile

❖ <https://speakerdeck.com/jmortega>



Index

Security Testing for Mobile Apps

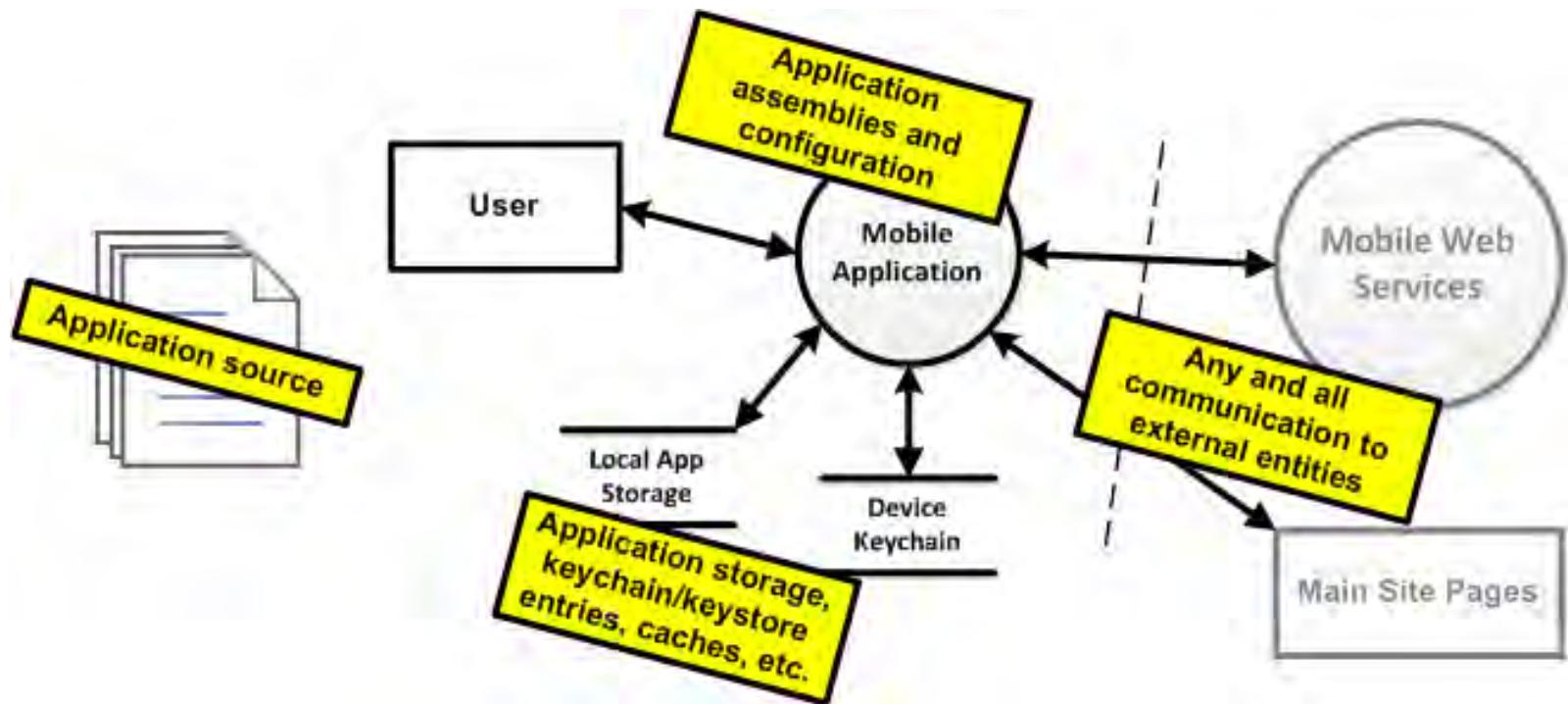
Risk Management & Security Threads

Vulnerabilities & Mobile Security Risks

Testing Components Security & Tools

Security Test Plan & Best Practices

Security Testing for Mobile Apps



Security Testing for Mobile Apps



**White Box
Testing**
Static Analysis
Code

**Black Box
Testing**
Dinamyc Analysis
at Runtime

Static Application Security Testing

**Source code
Review**

**Reverse
engineering**

**Lint for Android
Studio**

**Sonnar Plugins
for mobile
[http://
www.sonarqube.org](http://www.sonarqube.org)**

Static Application Security Testing

The screenshot shows the Android Studio interface during static analysis. At the top, two tabs are visible: 'Results for Inspection Profile 'Insecure random number generation'' and 'Results for Inspection Profile 'Project Default''. The left pane displays inspection results categorized under 'android-pbe' (201 items), including 'Data flow issues' (2 items), 'Declaration redundancy' (34 items), 'Java language level migration aids' (1 item), 'Probable bugs' (1 item), 'Properties Files' (6 items), 'Security issues' (10 items), and 'Spelling' (147 items). The 'Insecure random number generation' item under 'Security issues' is selected. The right pane provides detailed information for the selected profile: ID 'UnsecureRandomNumberGeneration', Description 'Reports any uses of `java.lang.Random` or `java.lang.math.Random()`. In secure environments, `java.secure.SecureRandom` is a better choice, offering cryptographically secure random number generation.', and a 'More Info' dialog box.

Insecure random number generation

Description
Reports any uses of `java.lang.Random` or `java.lang.math.Random()`. In secure environments, `java.secure.SecureRandom` is a better choice, offering cryptographically secure random number generation.

More Info

Using MODE_WORLD_WRITEABLE with `openFileOutput` can be risky, review carefully

Issue Explanation:
There are cases where it is appropriate for an application to write world writeable files, but these should be reviewed carefully to ensure that they contain no private data, and that if the file is modified by a malicious application it does not trick or compromise your application.

OK

At the bottom, the navigation bar includes 'Problems', '@ Javadoc', 'Declaration', 'Console', 'LogCat', and 'Lint Warnings'. The 'Lint Warnings' tab is active, showing 0 errors, 2 warnings. A table lists the findings:

Description	Category	Location
Exported service does not require permission	Security	AndroidManifest.xml:21 (AwesomeApp)
Using MODE_WORLD_WRITEABLE with <code>openFileOutput</code> can be risky, review carefully	Security	AwesomeAppActivity.java:19 in awesome

Static Application Security Testing



<https://github.com/SonarCommunity/sonar-android>

PLUGINS

PLUGIN	VERSION	DESCRIPTION
Android [android]	1.0	Enables analysis and reporting on Android projects. Depends upon: Java : SonarQube rule engine. License: GNU LGPL 3 Author: SonarSource and Jerome Van Der Linden, Stephane Nicolas, Florian Roncar, Thomas Bores Links: Homepage Issue Tracker
		Uninstall
Findbugs [findbugs]	3.0	Analyze Java code with Findbugs 3.0.0.
Java [java]	2.4	SonarQube rule engine.

Rules

[New Search](#)

security Quality Profile: Android Lint Language: Java Activation: Active + More Criteria [Search](#)

LANGUAGES Java | 3 REPOSITORIES Android Lint | 3

Ordered By Relevance ▾ Found: 3 [Permalink](#)

Missing allowBackup attribute android-lint:AllowBackup	Minor No tags Available Since September 22 2014 Android Lint (Java)
Java Using a fixed seed with SecureRandom	Ensure that allowBackup is explicitly set in the application's manifest The allowBackup attribute determines if an application's data can be backed up and restored. It is documented at http://developer.android.com/reference/android/R.attr.html#allowBackup
Java Using setJavaScriptEnabled	By default, this flag is set to true. When this flag is set to true, application data can be backed up and restored by the user using adb backup and adb restore.
Java Missing allowBackup attribute	This may have security consequences for an application. adb backup allows users who have enabled USB debugging to copy application data off of the device. Once backed up, all application data can be read by the user. adb restore allows creation of application data from a source specified by the user. Following a restore, applications should not assume that the data, file permissions, and directory permissions were created by the application itself.

Static Application Security Testing

<http://sourceforge.net/projects/aqnitioolt>

You can decompile APKs and search calls dangerous functions.

Num...	Principle/s	Questions	Yes	No	N/A
62	Auditing and Logging	Are logs correctly sanitised to ensure that sensitive data is not logged in clear text by the application?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
63	Auditing and Logging	Are all application errors logged to a central log server?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
64	Auditing and Logging	Does the design identify the level of auditing & logging needed and identify key parameters to be logged & audited?	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
65	Auditing and Logging	Does the design ensure that successful and unsuccessful attempts to execute privileged actions are logged?	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
66	Auditing and Logging	Does the design ensure that the content of audit logs meet the requirements of company/regulatory standards?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
78	Auditing and Logging	Does the application have non-repudiable logs for users access to paid resources?	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
15	Authentication	Are application trust boundaries identified on the data flow diagrams in the design document?	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
16	Authentication	Does the design identify the identities that are used to access resources across all trust boundaries?	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
17	Authentication	Does the application require authentication for all resource access attempts except for publicly accessible resources?	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

action profile View application profiles Reporting The Principles Checklist guidance Static analysis

Your code will be shown below

```
// Encrypt
SecretKey secret = new SecretKeySpec(tmp.getEncoded(), "AES");
Cipher encryptionCipher = Cipher.getInstance(CIPHER_ALGORITHM);
IvParameterSpec ivspec = new IvParameterSpec(ivVector);
encryptionCipher.init(Cipher.ENCRYPT_MODE, secret, ivspec);
encryptedText = encryptionCipher.doFinal(cleartext.getBytes());

// Encode encrypted bytes to base64 text to save in text file
result = base64.encodeToString(encryptedText, base64.DEFAULT);

} catch (Exception e) {
e.printStackTrace();
}

return result;
}

// Password based encryption
// Decode the provided base64 string, then decrypt based on the provided password
// Return decrypted cleartext string
public String decrypt(String encryptedText, String password) {
String result = "";

try {

// Decode base64 text back to encrypted bytes
byte[] toDecrypt = base64.decode(encryptedText, base64.DEFAULT);

}

Findings and linked checklist items
If cryptography is used then a strong cipher should be used such as AES, 3DES or SHA-256.

You must also determine what size key is used, the larger the better. Where is hashing performed? Are password that are being persisted hashed? How are random numbers used in cryptographic methods generated? Is the PRNG sufficiently random?

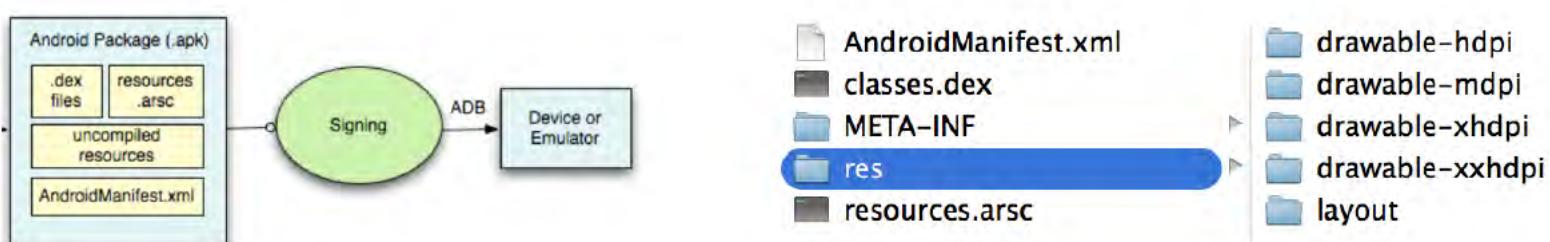


| Questions                                                                                              |
|--------------------------------------------------------------------------------------------------------|
| Does the application secure secrets that shouldn't be visible to the users with server side functions? |
| Does the application use salts when hashing passwords?                                                 |
| Does the design identify how encryption keys and salts will be securely stored?                        |
| Does the design identify the key rotation policy for the application?                                  |
| Does the design specify how the encryption keys for this application are stored securely?              |


```



Reverse engineering on Android



APK Tool /Dex2jar/Java Decompiler

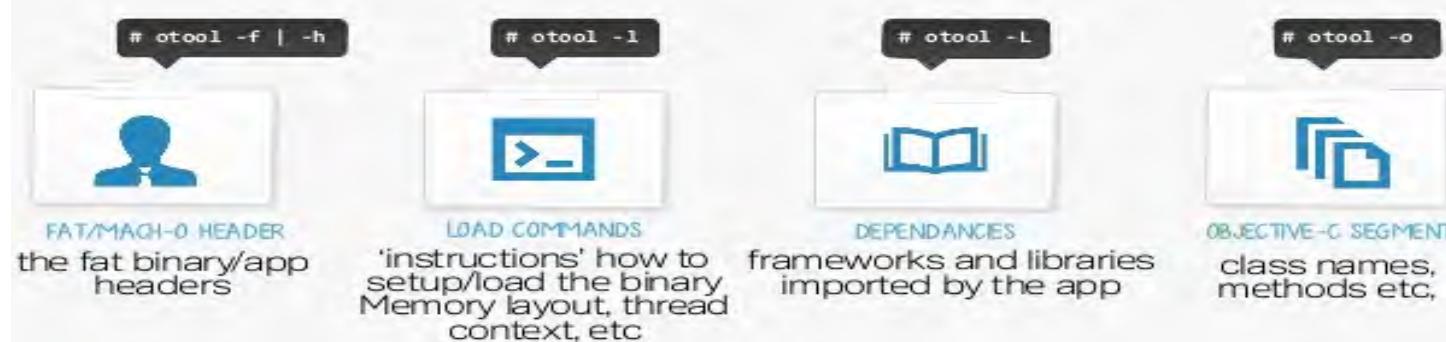
```
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /Users/filip/Library/apktool/framework/1.apk
I: Loaded.
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values /* XMLs...
I: Done.
I: Copying assets and libs...
```

Static analysis on iOS

Xcode Development environment

Apple Developer Tools

'otool' command provided by XCode can be used to get information from iOS application binaries and can be used to support security analysis.



Static analysis on iOS

Detect memory leaks with XCode

The screenshot shows a portion of an Objective-C code file with annotations from the Clang Static Analyzer. The code is as follows:

```
147 - (void) connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
148 {
149 #if DEBUG_LOGGING
150     NSLog(@"%@", __PRETTY_FUNCTION__);
151 #endif
152
153     if (self.responseData == nil) {
154         self.responseData = [[NSMutableData alloc] initWithCapacity:[data length]];
155     }
156     [self.responseData appendData:data];
157 }
158 }
```

Annotations highlight several issues:

- Line 154: A call to `[NSMutableData alloc]` is annotated with "1. Method returns an Objective-C object with a +1 retain count".
- Line 156: The assignment to `self.responseData` is annotated with "2. Object leaked: allocated object is not referenced later in this execution path and has a retain count of +1".

Clang Static Analyzer

<http://clang-analyzer.llvm.org>

Flawfinder(Security weakness in C/C++)

<http://www.dwheeler.com/flawfinder>

Dinamyc Application Security Testing

Analyze
Network
Traffic at
runtime

Analyze Remote
Services

DroidBox for
Android
Instruments for
iOS

Discovering
logical
vulnerabilities



Dinamyc Application Security Testing

➤ <https://code.google.com/p/droidbox>

- Information leaks Network IO and File IO
- Cryptography operations SMS and Phone calls

A	PID	TID	Application	Tag	Text
5.36	1136	1136	com.androwarn.samp	DroidBox	Landroid/telephony/TelephonyManager; ->getTelephony/CellLocation; =[0, 0, 2147483647]
5.35	1136	1136	com.androwarn.samp	DroidBox	Landroid/telephony/TelephonyManager; ->getTelephony/TelephonyManager; ->getD
5.38	1136	1136	com.androwarn.samp	DroidBox	ring; =0000000000000000
5.46	1136	1136	com.androwarn.samp	DroidBox	Landroid/telephony/TelephonyManager; ->getS
5.53	1136	1136	com.androwarn.samp	DroidBox	g/String; =3102600000000000
					Lcom/androwarn/sampleapplication/SampleApp
					put(Ljava/lang/String; ;MyLocalFiletoBeWritten);>FileOutputStream; =java.io.FileOutputStream
					Landroid/net/Uri; ->parse(Ljava/lang/String;
					Landroid/net/Uri; =content://sms/inbox
					Landroid/content/ContentResolver; ->query(L
					nt://sms/inbox [Ljava/lang/String; ;null
					ull [Ljava/lang/String; ;null Ljava/lang/database/Cursor; ;android.content.Content
					Inner@b2d9e738
					Landroid/content/ContentResolver; ->query(L



Dinamyc Application Security Testing

```
21 [File activities]
22 -----
23     [Read operations]
24 -----
25     [22.9400451183]    Path: /data/data/droidbox.tests/files/myfilename.txt^A
26         Data: Write a line
27     [24.2107310295]    Path: /data/data/droidbox.tests/files/myfilename.txt
28         Data:
29     [25.997330904]     Path: /data/data/droidbox.tests/files/output.txt
30         Data: null
31     [26.781430006]     Path: /data/data/droidbox.tests/files/output.txt
32         Data:
33     [Write operations]
34 -----
35     [21.3330090046]    Path: /data/data/droidbox.tests/files/myfilename.txt^A
36         Data: Write a line
37     [21.3614990711]    Path: /data/data/droidbox.tests/files/output.txt
38         Data: null
39 [Crypto API activities]
40 -----
41     [26.8029410839]    Key:{0, 42, 2, 54, 4, 45, 6, 7, 65, 9, 54, 11, 12, 13, 60, 15} Algorithm: AES
42     [26.811686039]    Operation:{encryption} Algorithm: AES
43         Data:{357242043237517}
44     [26.818600893]    Key:{0, 42, 2, 54, 4, 45, 6, 7, 65, 9, 54, 11, 12, 13, 60, 15} Algorithm: AES
45     [26.8250999451]    Operation:{decryption} Algorithm: AES
46         Data:{357242043237517}
47     [26.8305909634]    Key:{0, 42, 2, 54, 4, 45, 6, 8} Algorithm: DES
48     [26.8399989605]    Operation:{encryption} Algorithm: DES
49         Data:{357242043237517}
50     [26.8453080654]    Key:{0, 42, 2, 54, 4, 45, 6, 8} Algorithm: DES
51     [26.853967905]    Operation:{decryption} Algorithm: DES
52         Data:{357242043237517}
53     [Network activity]
54 -----
```

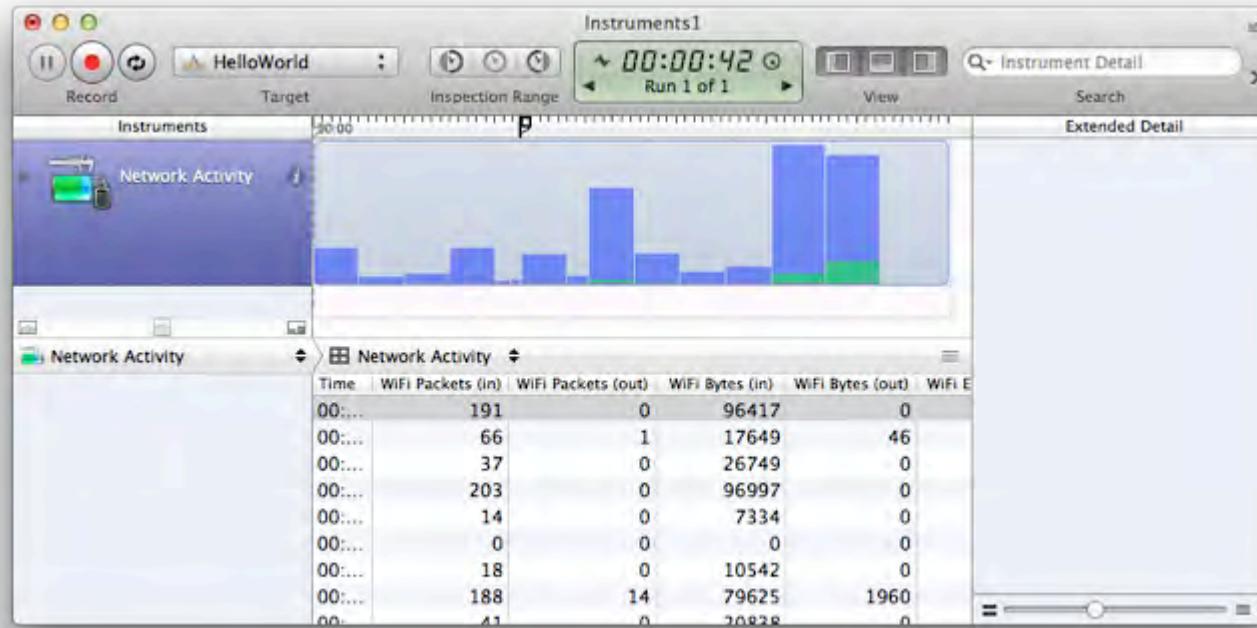
Instruments for iOS applications

File Activity Monitoring

Network Monitoring

Memory Monitoring

Process Monitoring





Application Security Analyser

```
python androwarn.py -i my_apk.apk -r html -v 3
```

Androwarn Report com.androwarn.sampleapplication

APPLICATION INFORMATION

- Application Name
- Application Version
- Package Name
- Description

ANALYSIS RESULTS

- Telephony Identifiers Leakage
- Device Settings Harvesting
- Location Lookup
- Connection Interfaces Exfiltration
- Telephony Services Abuse
- Audio Video Eavesdropping
- Suspicious Connection Establishment
- PIM Data Leakage
- Code Execution

APK FILE

- File Name
- Fingerprint
- File List
- Certificate Information

ANDROIDMANIFEST.XML

- Main Activity

Suspicious Connection Establishment

This application opens a Socket and connects it to the remote address '192.168.56.101' on the '1337' port
This application opens a Socket and connects it to the remote address '192.168.56.101 Ljava/net/InetAddress;:->getByName(Ljava/lang/String;)Ljava/net/InetAddress;' on the '1338' port

Telephony identifiers exfiltration: IMEI, IMSI, MCC, MNC, LAC, CID, operator's name...

Device settings exfiltration: software version, usage statistics, system settings, logs...

Geolocation information leakage: GPS/WiFi geolocation... **Connection interfaces information exfiltration:** WiFi credentials, Bluetooth MAC adress...

Telephony services abuse: premium SMS sending, phone call composition...

Audio/video flow interception: call recording, video capture... **Remote connection establishment:** socket open call, Bluetooth pairing, APN settings edit...

PIM data leakage: contacts, calendar, SMS, mails...

External memory operations: file access on SD card...

PIM data modification: add/delete contacts, calendar events... **Arbitrary code execution:** native code using JNI, UNIX command, privilege escalation...

Denial of Service: event notification deactivation, file deletion, process killing, virtual keyboard disable, terminal shutdown/reboot...



Risk Management

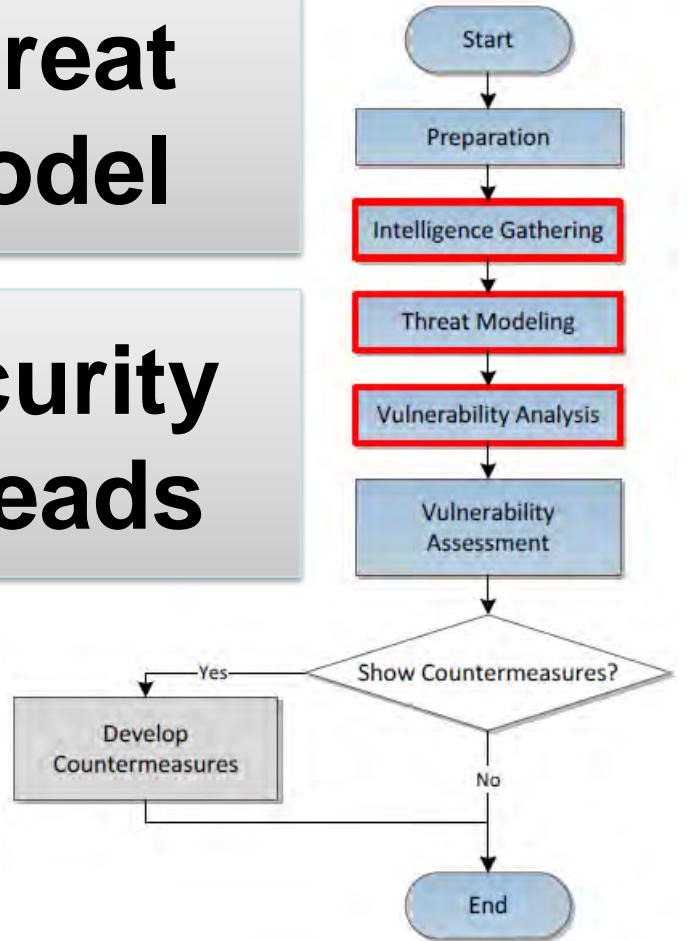
Intelligence Gathering

Vulnerabilities

OWASP Security risks

Threat model

Security threads



Intelligence Gathering

Environmental
Analysis

Architectur
al Analysis

Analyze internal
processes and
structures

App [network
interfaces, used data,
communication with
other resources, session
management]¹

Runtime
environment [MDM,
jailbreak/rooting, OS
version]

Backend Services
[application server,
databases]

Intelligence Gathering

What type of device is it?

**Determine Operating System
version**

Is the device already rooted?

Is the device passcode

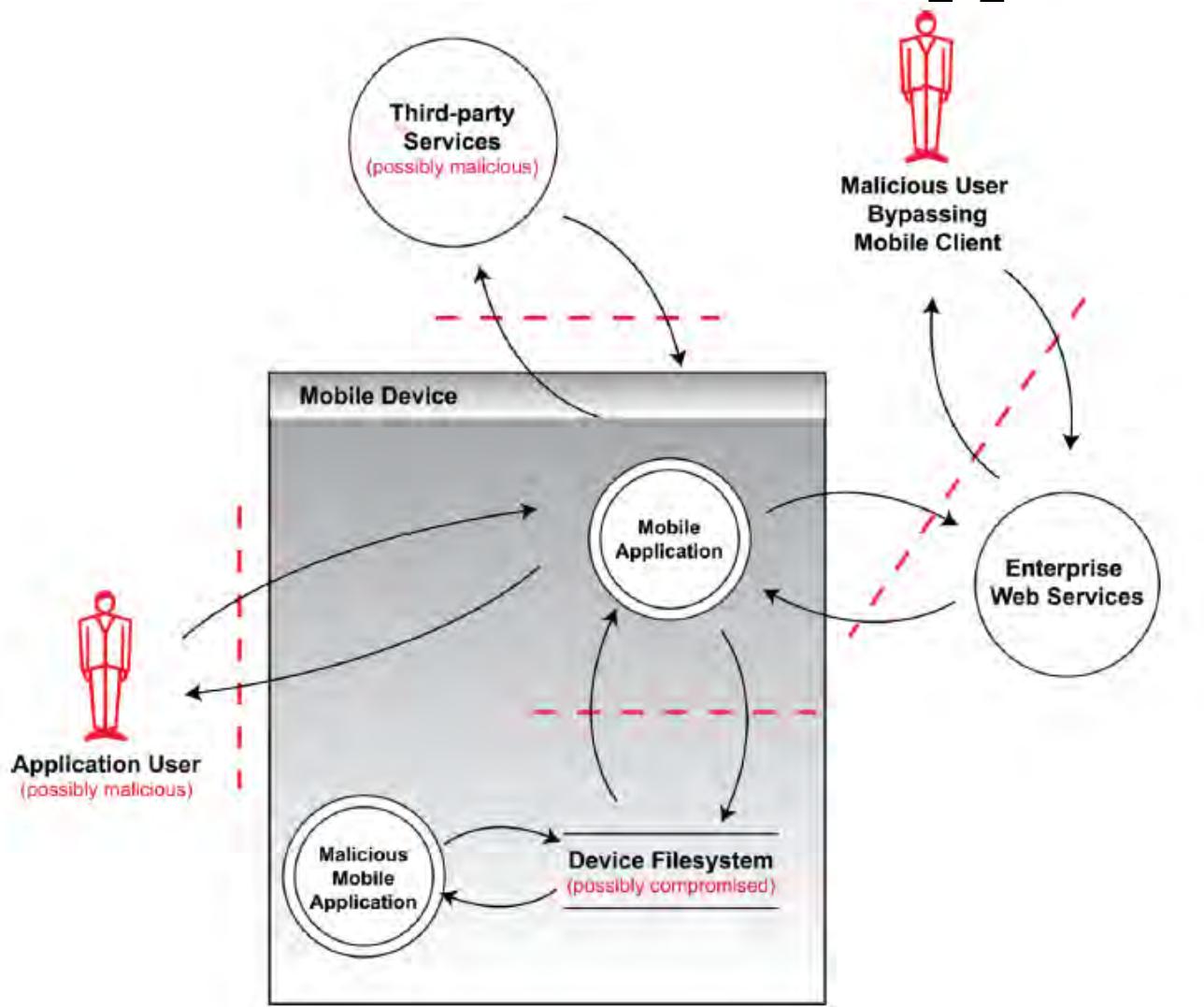
enabled?

What key applications are

installed?

**Is the device connected to
network?**

Thread model in Mobile applications



Functional Security threads

Authentication

Cryptography

Data Protection

Input Validation

Access Control

Communication
Security

Session
Management

Error Handling and
Logging

Security issues

Malicious Applications

- Rooting Exploits
- SMS Fraud
- Rapid Malware Production

Dynamic Analysis

- Sandbox
- Real-time Monitoring
- Mobile Specific Features

Static Analysis

- Permissions
- Data Flow
- Control Flow

Browser Attacks

- Phishing
- Click Through

Mobile Botnets

- Epidemic Spread
- Attacking Network Services
- Tracking Uninfected Devices

User Education

- Ignoring Permissions
- Phishing
- Improperly Rooting Devices
- Alternative Markets

Vulnerabilities

**Third party
libraries**



Components

**WebView
[JavaScript+Cache]**

SQLite DataBase

Shared Preferences

Vulnerabilities

- | | |
|--|--|
| <ul style="list-style-type: none">1. Activity monitoring and data retrieval2. Unauthorized dialing, SMS, and payments3. Unauthorized network connectivity (exfiltration or command & control)4. UI Impersonation5. System modification (rootkit, APN proxy config)6. Logic or Time bomb | <ul style="list-style-type: none">7. Sensitive data leakage (inadvertent or side channel)8. Unsafe sensitive data storage9. Unsafe sensitive data transmission10. Hardcoded password/keys |
|--|--|

Vulnerabilities Analysis

**Static
methods**

**Dynamic
methods**

**Forensic
methods**

Automatic and

manual source code

analysis

Reverse Engineering

**Network monitoring
and traffic analyzing**

Runtime analysis

Log analysis

**File permission
analysis**

File content analysis

Dynamic methods tools

**Network monitoring
and traffic analyzing**

Wireshark, BurpSuite

Runtime analysis

iOS GNU debugger,
Snoop-it, Cycript

File analysis

Mercury, Intent Sniffer,
Intent Fuzzer

androidAuditTools

Testing vulnerabilities

Data flow

**Authenticat
ion**

**Data
Storage**

**Authorizati
on**

**Data
leakage**

Server-side

OWASP Mobile Security Risks

Insecure Data Storage

Transport Layer Protection, HTTP/SSL

Authorization and Authentication

Cryptography / Encrypting data

Session handling

Weak Server Side Controls in backend services

Sensitive information [passwords, API keys, code obfuscation]

Data Leakage [cache, logging, temp directories]

Testing components security

**Content
Providers
NetWork**

**Connections
HTTP / SSL**

Certificates

**Data
Encryption**

Services

Data Storage

SQLite

**Shared
Preferences**

File storage

WebView

HTTPS and SSL can protect against Man in the Middle attacks and prevent casual snooping



Data Storage

Encrypt data

Never store user credentials

Tools like **SQLCipher** for storing database in applications

No global permissions granted to applications, using the principle of "**least privilege**".



Secure Storage on Android

The access permission of the created file was set to
WORLD_READABLE / WORLD_WRITABLE

Other app could read /write the file if the file path is known.

Application data (private files) should be created
with the access permission **MODE_PRIVATE**

```
FileOutputStream fos = openFileOutput("MyFile",  
Context.MODE_PRIVATE);  
fos.write("contenido".getBytes());  
fos.close();
```



Insecure Data on Android

Look for file open operations using

- *Context.MODE_WORLD_READABLE*
- *(translates to “1”)*

The screenshot shows a Java Decomiler interface with the title "Java Decomiler - Config.class". The left pane displays the class hierarchy: "classes.dex.dex2jar.jar" contains "com.app.denim.group" which has "Config" and "Config\$1". The right pane shows the decompiled code for "Config.class". A red oval highlights the line of code: "FileOutputStream local FileOutputStream = getApplicationContext().openFileOutput("SecretFile.txt", 1);".

```
try
{
    String str4;
    FileOutputStream local FileOutputStream = getApplicationContext().openFileOutput("SecretFile.txt", 1);
    OutputStreamWriter local OutputStreamWriter = new OutputStreamWriter(local FileOutputStream);
    localObject = localStringBuilder.toString();
    local OutputStreamWriter.write((String)localObject);
    local OutputStreamWriter.close();
    String str5;
    for (localObject = i; ; localObject = str5)
    {
        return localObject;
        localStringBuilder.append(str4);
        break;
        localException1 = localException1;
        localObject = new StringBuilder("Error while retrieving URL ").append(str1).append(": ");
        str5 = localException1.getMessage();
    }
}
```

Secure Storage on iOS

NSFileManager class

NSFileProtectionKey attribute

- NSFileProtectionNone – Always accessible
- NSFileProtectionComplete – Encrypted on disk when device is locked or booting

```
[[NSFileManager defaultManager] createFileAtPath:[self
filePath]
contents:[@“super secret file contents“
dataUsingEncoding:NSUTF8StringEncoding]
attributes:[NSDictionary
dictionaryWithObject:NSFileProtectionComplete
forKey:NSFileProtectionKey]];
```

DataBase Storage



3rd-party extensions

Support iOS / Android

<https://www.zetetic.net/sqlcipher/open-source>

256-bit AES Encrypt SQLite database

Secure Preferences on Android

<https://github.com/scottyab/secure-preferences>



Secure Shared Preferences

```
root@generic:/data/data/com.securepreferences.sample/shared_prefs # ls  
com.securepreferences.sample_preferences.xml  
.xml  
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>  
<map>  
    <string name="cgASlY/nymum+yv+S+21tmliSHPFmElhAT9pQ/5DcA">uKJlx6JdM0JaCABUW  
qxETBRvUliCmjJidGLtierH/6s</string>  
</map>  
.xml  
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>  
<map>  
    <string name="o51qxitsA/Eag2EbNqSA9A">0vQ8+ILj9keS3fU3r//lRw</string>  
    <string name="7cFWxWwcNqYvBRzgh5S5Pw">Jg3xcDITiyK4rN3+4biPwg</string>  
    <string name="JD1l6inbJWv3QKurgJkpAg">Zeg6wpqkWjwNEOLQWHzsUQ</string>  
    <string name="cgASlY/nymum+yv+S+21tmliSHPFmElhAT9pQ/5DcA">uKJlx6JdM0JaCABUW  
qxETBRvUliCmjJidGLtierH/6s</string>  
    <string name="UZFTE9ZTSs49H1G4emERPA">W8KyltBxcICu10mMi7ys0Q</string>  
</map>
```



Cryptography

Android and iOS implement standard crypto libraries such as AES algorithm

Android provides the **javax.crypto.spec.PBEKeySpec** and **javax.crypto.SecretKeyFactory** classes to facilitate the generation of the password-based encryption key.

```
try{  
    PBEKeySpec keySpec = new PBEKeySpec(password.toCharArray(), salt,  
    PBE_ITERATION_COUNT, 256);
```

```
    SecretKeyFactory keyFactory = SecretKeyFactory.getInstance(PBE_ALGORITHM);  
    SecretKey tmp = keyFactory.generateSecret(keySpec);
```

```
    // Encrypt  
    SecretKey secret = new SecretKeySpec(tmp.getEncoded(), "AES");  
    Cipher encryptionCipher = Cipher.getInstance(CIPHER_ALGORITHM);  
    IvParameterSpec ivspec = new IvParameterSpec(initVector);  
    encryptionCipher.init(Cipher.ENCRYPT_MODE, secret, ivspec);  
    encryptedText = encryptionCipher.doFinal(cleartext.getBytes());
```

```
    // Encode encrypted bytes to Base64 text to save in text file  
    result = Base64.encodeToString(encryptedText, Base64.DEFAULT);
```

```
} catch (Exception e) {  
    e.printStackTrace()  
}
```



Avoid Data Leakage on Android

Don't expose data through logcat on production

```
public static final boolean SHOW_LOG =  
BuildConfig.DEBUG;
```

```
public static void d(final String tag, final String msg) {  
    if (SHOW_LOG)  
        Log.d(tag, msg);  
}
```

-assumenosideeffects class android.util.Log

```
{  
public static *** d(...);  
public static *** v(...);  
public static *** i(...);  
public static *** e(...);  
}
```

Proguard configuration



Permissions in mobile apps

- ✓ Try to minimize permissions

AndroidManifest.xml on Android

```
<manifest package="com.example.android" ...>

    <uses-permission android:name="android.permission.
        ACCESS_FINE_LOCATION"/>
    <uses-permission
        android:name="android.permission.INTERNET" />
    <uses-permission
        android:name="android.permission.READ_CONTACTS" />
    ...
</manifest>
```

Permissions in mobile apps

info.plist on iOS

PROJECT test

INFO

Summary Build Settings Build Phases Build R

Custom iOS Target Properties

Key	Type	Value		
Bundle name	String	\$(PRODUCT_NAME)		
Bundle identifier	String	com.manbolo.\$(PRODUCT_NAME)		
InfoDictionary version	String	6.0		
Required device capabilities	Array (1 item)	Item 0 String armv7		
Bundle version	String	1.0		
Executable file	String	\$(EXECUTABLE_NAME)		
Application requires iPhone environment	Boolean	YES		
Supported interface orientations	Array (3 items)	Bundle display name String \$(PRODUCT_NAME)	Bundle creator OS Type code String ?????	Bundle OS Type code String APPL
Localization native development region	String	en		
Bundle versions string, short	String	1.0		

Document Types (0)

Exported UTIs (0)

Imported UTIs (0)

URL Types (0)

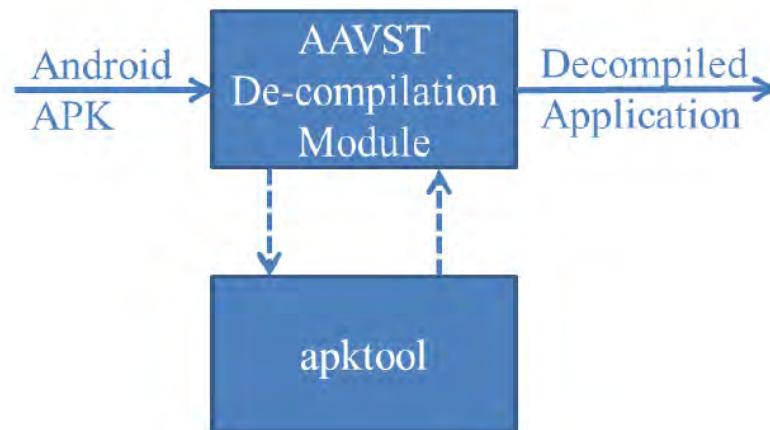
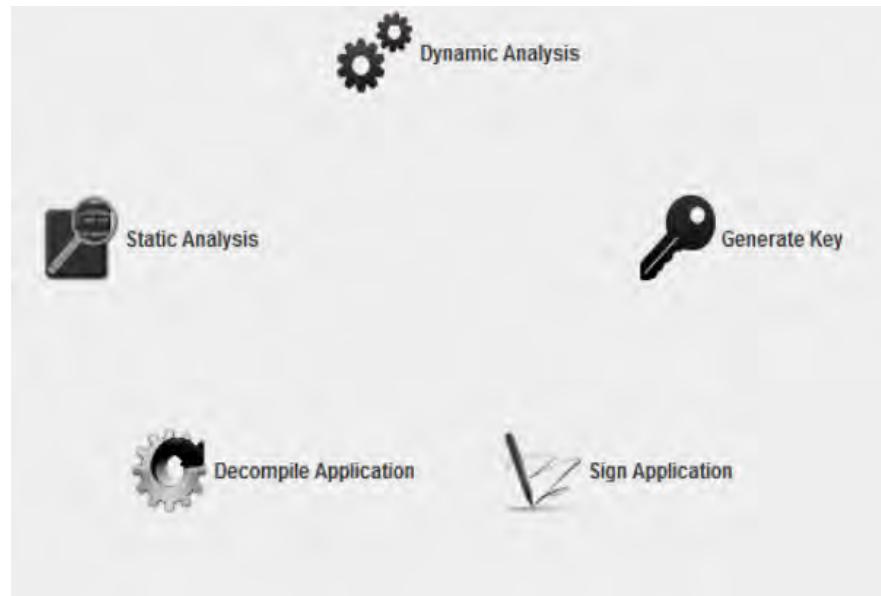
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>CFBundleDevelopmentRegion</key>
    <string>English</string>
    <key>CFBundleExecutable</key>
    <string>Maps</string>
    <key>CFBundleIdentifier</key>
    <string>com.apple.Maps</string>
    <key>CFBundleInfoDictionaryVersion</key>
    <string>6.0</string>
    <key>CFBundlePackageType</key>
    <string>APPL</string>
    <key>CFBundleResourceSpecification</key>
    <string>ResourceRules.plist</string>
    <key>CFBundleSignature</key>
    <string>????</string>
    <key>CFBundleSupportedPlatforms</key>
    <array>
        <string>iPhoneOS</string>
    </array>
    <key>CFBundleURLTypes</key>
    <array>
        <dict>
            <key>CFBundleURLSchemes</key>
            <array>
                <string>maps</string>
            </array>
        </dict>
    </array>
    <key>CFBundleVersion</key>
    <string>1.0</string>
    <key>DTCompiler</key>
    <string>4.2</string>
    <key>DTPlatformName</key>
    <string>iphoneos</string>
    <key>DTPlatformVersion</key>

```

"Info.plist" [readonly] 60L, 1500C

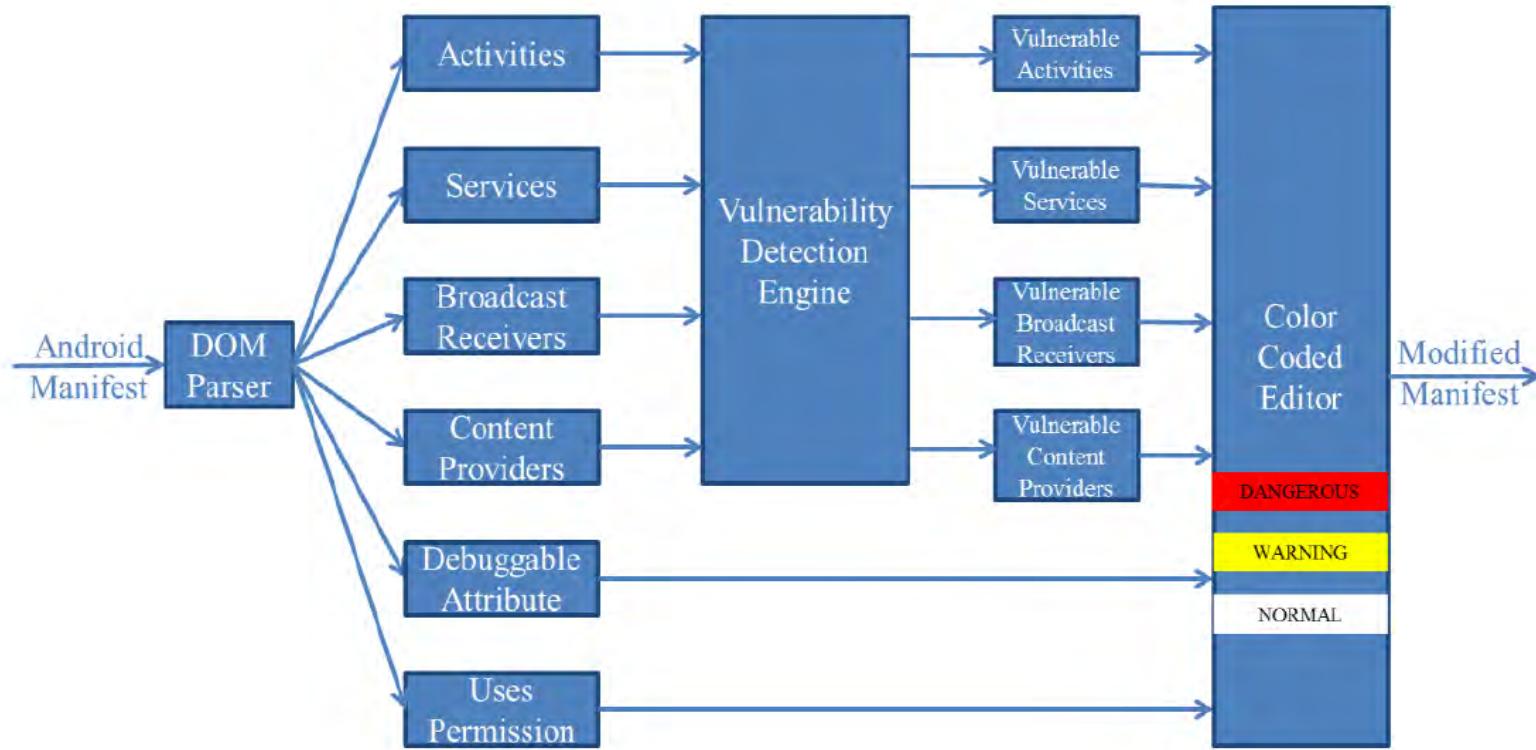


Vulnerability scanner on Android





Vulnerability scanner on Android



Vulnerability scanner on Android



Static Analysis

File Edit

Application
ie Apps\Comm\com.rediff.mail.and...

Scan

Exported Activities
 Exported Services
 Exported Content Providers
 Exported Broadcast Recievers
 Debuggable Attribute
 Uses Permission

Update

Dangerous

Warning

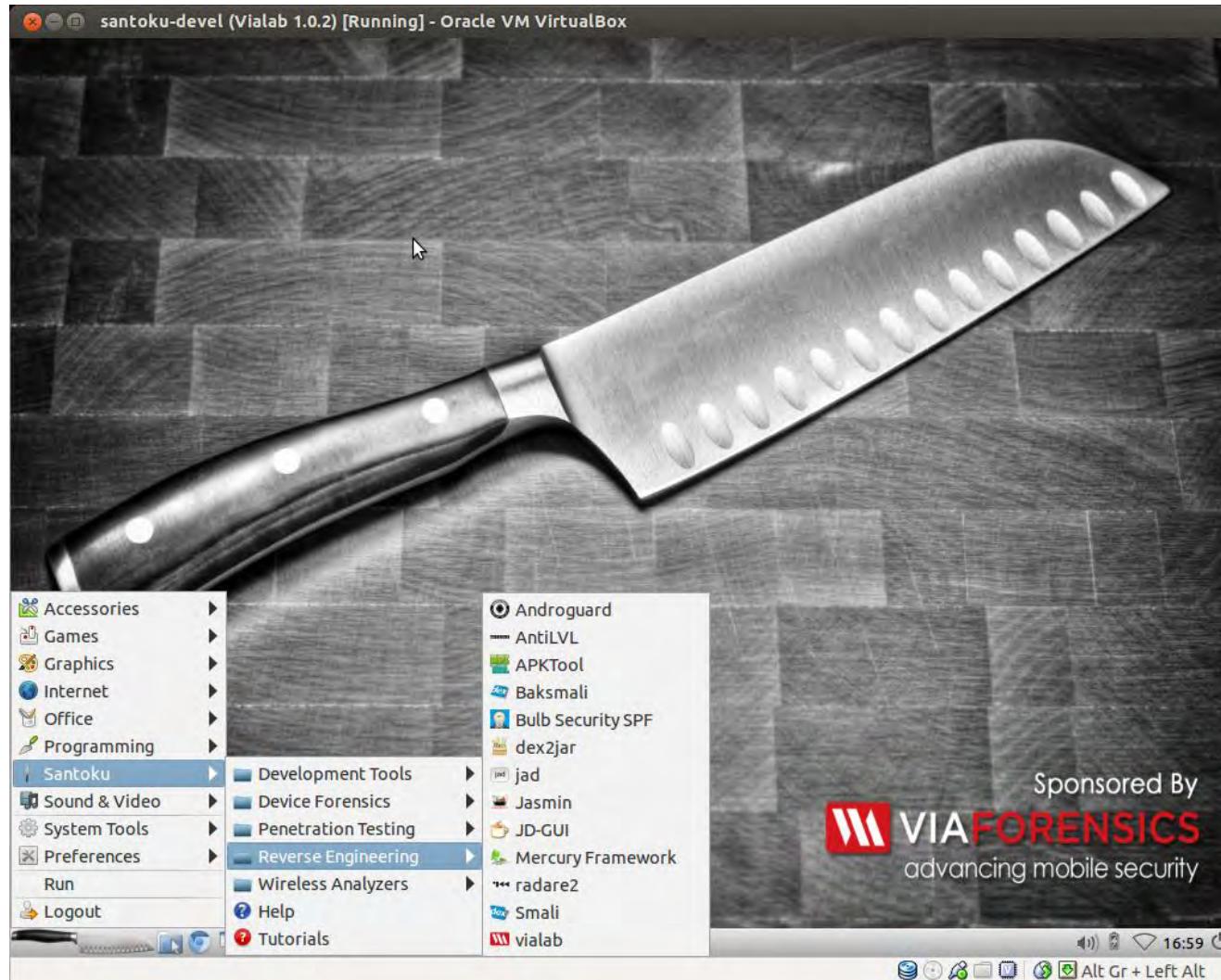
Normal

Re-Build apk

Name	Intent	permission	exported	Edit
activities.NetworkConfigureActivity				Edit
activities.InboxActivity				Edit
activities.SearchActivity	android.intent.action.SEARCH			Edit
activities.Login	android.intent.action.MAIN android.intent.action.VIEW			Edit
activities.DownloadManagerActivity				Edit
activities.ReadMailActivity				Edit
activities.SettingsActivity				Edit
activities.MessageActivity				Edit
activities.UpdateDlg				Edit
activities.FileBrowser				Edit
activities.ContactSelection				Edit
activities.PhoneContactsActivity				Edit
activities.AddContacts				Edit
activities.ViewContacts				Edit
activities.WriteMailings				Edit
	android.intent.action.VIEW			
	android.intent.action.VIEW			
	android.intent.action.SEND			
	android.intent.action.SEND_MULTIPLE			
calendar RCal				Edit
activities.ActivityEditEvent	android.intent.action.RCal.ACTION_M...			Edit
activities.ActivityViewEvent	android.intent.action.RCal.ACTION_M...			Edit
activities.ActivityViewEvent	android.intent.action.RCal.ACTION_M...			Edit
activities.Reminder_AlarmDialog	android.intent.action.RCal.ACTION_M...			Edit
activities.FolderActivity				Edit
activities.ActivityViewAttendee				Edit
activities.BrowserActivity				Edit



Tools / Santoku Linux



Security test plan

Test cases	Example
Test Title /level	Encryption /critical
Test Description	When connections are used encryption is used for sending / receiving sensitive data.
Details and tools	All sensitive information (personal data, credit card & banking information etc.) must be encrypted during transmission over any network or communication link.
Expected result	It has been declared that the Application uses encryption when communicating sensitive data.

Security test plan

Test cases	Example
Test Title /level	Passwords /critical
Test Description	Passwords and sensitive data are not stored in the device and not echoed when entered into the App, sensitive data is always protected by password.
Details and tools	The objective of the test is to minimize the risk of access to sensitive information should the device be lost, by ensuring that no authentication data can be re-used by simply re-opening the application

Security test plan

Test cases	Example
Test Title /level	Passwords /critical
Expected result	
	<ol style="list-style-type: none">1. Entering a password or other sensitive data will not leave it in clear text if completion of the fields is interrupted but not exited.2. Passwords, credit card details, or other sensitive data do not remain in clear text in the fields where they were previously entered, when the application is reentered.3. Sensitive personal data should always need entry of a password before it can be accessed.

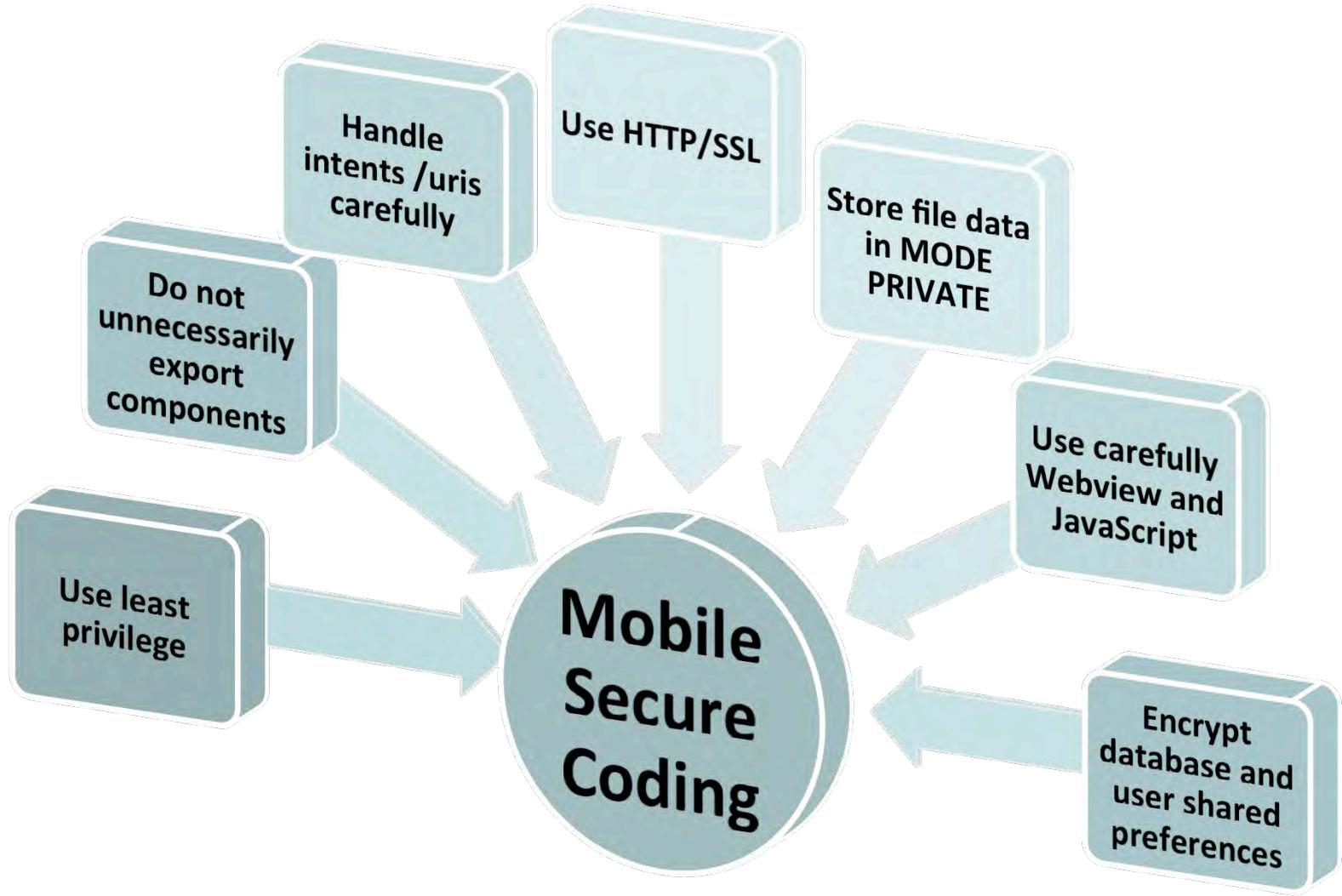
Best Practices

Integrate security in Continuous Integration (CI) process

Apply encryption /decryption techniques used for sensitive data communication

Detect areas in tested application that have more risks to detect vulnerabilities

Install an automated security vulnerability scanner, integrated with your continuous integration tool



Attacker vs Defenders

Defenders	Attackers
Often have limited time to put defences in place	Can take time to plan their attack
Have limited opportunities to improve their defences	Can invent new ways to attack
Need to defend against a range of possible attacks	Need only pick the most effective one
Need to defend all entry points	Can choose where to attack
Has limited resources to defend	Can be multiple attackers

References

OWASP Mobile Security Project

[https://www.owasp.org/index.php/
OWASP_Mobile_Security_Project](https://www.owasp.org/index.php/OWASP_Mobile_Security_Project)

Android Security Best Practices

[http://developer.android.com/training/articles/security-
tips.html](http://developer.android.com/training/articles/security-tips.html)

References

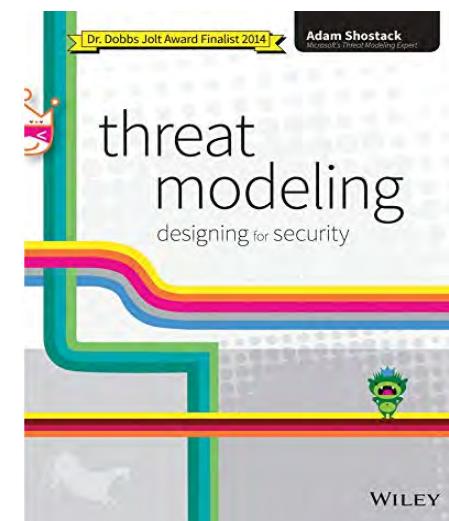
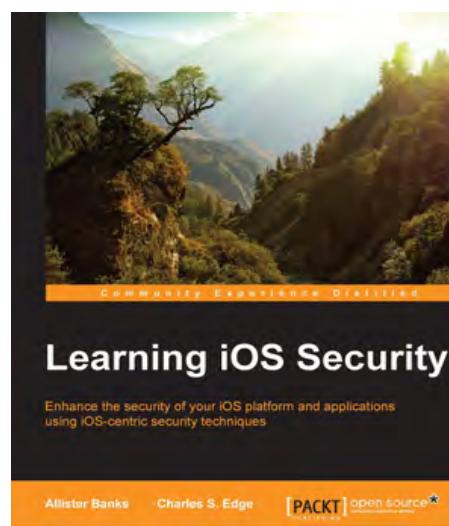
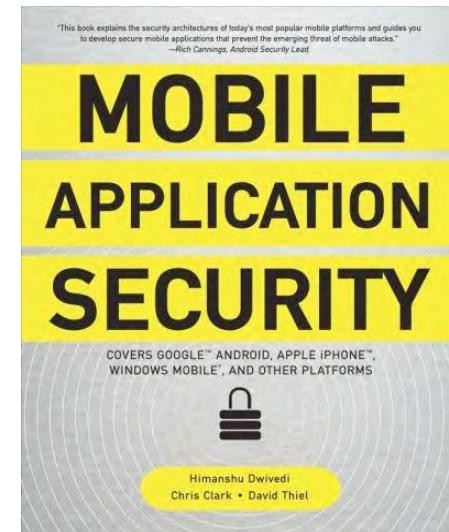
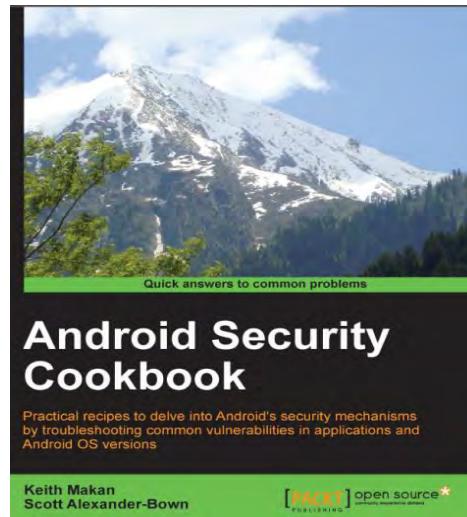
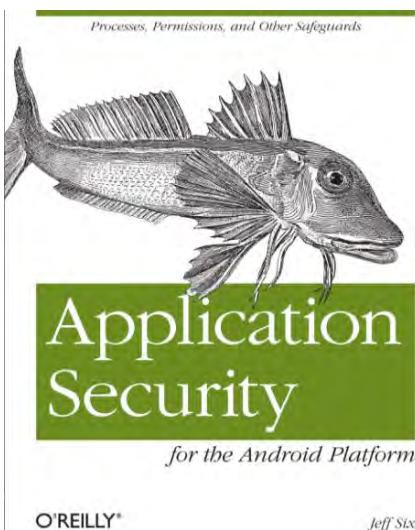
Tools

<https://github.com/secmobi/wiki.secmob.com>

IOS Application Security Testing

[https://www.owasp.org/index.php/
IOS Application Security Testing Cheat Sheet](https://www.owasp.org/index.php/IOS_Application_Security_Testing_Cheat_Sheet)

Books



Thank You,
Questions?