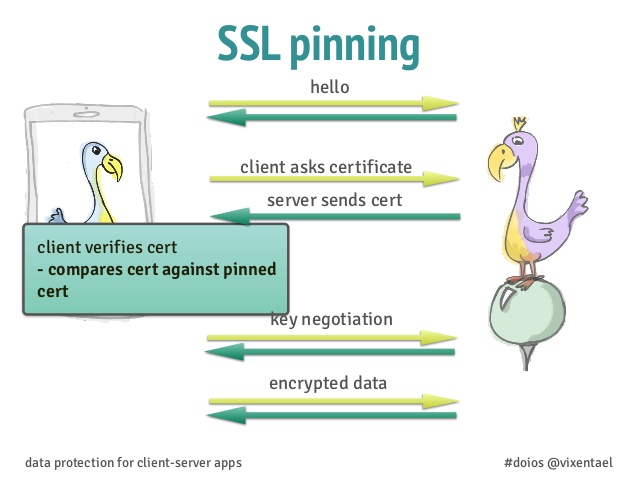
## **1.What is ssl pinning?**

In this type of certificate pinning the client normally is an application has exact server certificate details pre-installed into application and will check to see that the received server certificate matches the pre-configured certificate if there is no match then application will normally not function and will report an error to the user.



**Bypass ssl pinning**

**SSLUnpinning - Xposed Module**

If you need to intercept the traffic from an app which uses certificate pinning, with a tool like Burp Proxy, the SSLUnpinning will help you with this hard work! The SSLUnpinning through Xposed Framework, makes several hooks in SSL classes to bypass the certificate verifications for one specific app, then you can intercept all your traffic.

**2.what is Android version.**

Android is a mobile operating system developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets

The latest version is Android 7 Nougat, which will become available for more devices over the coming months. Marshmallow is 6.x, Lollipop is 5.x, and KitKat isversion 4.4.x, with 4.3, 4.2 and 4.1 all coming under the codename name Jelly Bean

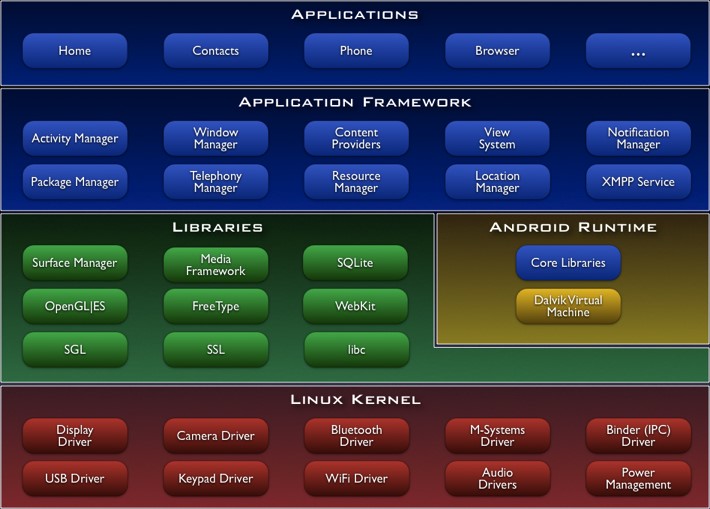
**Android version list**

* Android 1.6. Donut
* Android 2.1. Eclair
* Android 2.2. Froyo
* Android 2.3. Gingerbread
* Android 3.0. Honeycomb
* Android 4.0. Ice Cream Sandwich
* Android 4.1. Jelly Bean
* Android 4.4. KitKat

**3. what is Android architecture**

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

* Linux kernel
* Libraries
* Android Libraries
* Android Runtime
* Application Framework
* Applications

****

**1.Linux Kernel**

The basic layer is the Linux kernel. The whole Android OS is built on top of the Linux

Linux that interacts with the hardware and contains all the essential hardware drivers. Drivers are programs that control and communicate with the hardware.

For example, consider the Bluetooth function. All devices has a Bluetooth hardware in it.

**2.Libraries**

The next layer is the Android’s native libraries. It is this layer that enables the device to handle different types of data. These libraries are written in c or c++ language and are specific for a particular hardware

**Surface Manager:**

It is used for compositing window manager with off-screen buffering. There it is combined with other drawings and form the final screen the user will see. This off screen buffer is the reason behind the transparency of windows.

**Media framework:**

Media framework provides different media code allowing the recording and playback of different media formats.

**SQLite:**

SQLite is the database engine used in android for data storage purposes.

**WebKit:**

It is the browser engine used to display HTML content

**OpenGL:**

Used to render 2D or 3D graphics content to the screen

**3. Android Runtime**

Android Runtime consists of Dalvik Virtual machine and Core Java libraries.

**Dalvik Virtual Machine**

It is a type of JVM used in android devices to run apps and is optimized for low processing power and low memory environments. Unlike the JVM, the Dalvik Virtual Machine doesn’t run .class files, instead it runs .dex files. .dex files are built from .class file at the time of compilation and provides hifger efficiency in low resource environments. The Dalvik VM allows multiple instance of Virtual machine to be created simultaneously providing security, isolation, memory management and threading support. It is developed by Dan Bornstein of Google.

**Core Java Libraries**

These are different from Java SE and Java ME libraries. However these libraries provides most of the functionalities defined in the Java SE libraries.

**4. Application Framework**

These are the blocks that our applications directly interacts with. These programs manage the basic functions of phone like resource management, voice call management etc. As a developer, you just consider these are some basic tools with which we are building our applications. Important blocks of Application framework are:

**Activity Manager:**

Manages the activity life cycle of applications

Content Providers:

Manage the data sharing between applications

**Telephony Manager:**

Manages all voice calls. We use telephony manager if we want to access voice calls in our application.

**Location Manager:**

Location management, using GPS or cell tower

**Resource Manager:**

Manage the various types of resources we use in our Application

**5.Applications**

Applications are the top layer in the Android architecture and this is where our applications are gonna fit. Several standard applications comes pre-installed with every device, such as: SMS client app Dialer Web browser Contact manager As a developer we are able to write an app which replace any existing system app. That is, you are not limited in accessing any particular feature Thus Android is opening endless opportunities to the developer.

**4.What is Apk files?**

An APK file is the file format used for installing software on the Android operating system.

In order to see the files inside the given apk we need to

* Change its extension to .zip or .rar
* Then extract it
* Then we get a class.dex file along with other but they are not in human readable form
* So we use dex2jar tool. in order to open it and then dex2jar files needs jdgui another tool to open it.

**5.What is Difference Between Rooted and Unrooted Android Phones**

**Rooting**

**Rooting**is a process allowing users of Smartphone’s, tabletsand other devices running the Android operating system to attain privileged control (known as “root access”) within Android’s subsystem. Basically rooting is performed to overcome the limitation that **wireless service providers**and**hardware manufacture**of android phone put on some devices.

If your android mobile is rooted that means you are super users on your android cell phone and you got ability to customize or replace system application and setting the way you want to do.

This little word rooting provide you power to run specialized apps that require administrator-level permissions and perform another operation which are inaccessible to a normal android

**6.Why is it called Rooting?**

Basically “root” term comes from the Unix/Linux world which is used to describe a user who has “superusers” rights or permission to all the files and programs in the Android OS.

**Unrooted**  
**I**n an unrooted device we cannot install those applications easily as with a rooted device.

## **Advantages of Rooting an Android Phone**

Many people are unaware about rooting, they don’t even know what they can do with rooting android mobile. Below are going to share with you some of advantages which you are going to get after your device get rooted.

**1 Custom Software (ROM’s)**

If you have superusers rights means you can install Custom ROM’s on your android cell phone. Basically, custom Rom is own version of Android, a new Operating System which is developed by developers. Using these ROM’s  you can alter your Smartphone experience, it changes the color of your icons, app dock, an app menu etc.

**2.** **Speed and Performance**

We all know that kernels are behind the performance of the processor of your Smartphone. So if you change the kernel version, you can change your processor and that will also improve the speed and performance of your Smartphone.

**3.** **Increasing Battery life**.

There are many apps specially made for rooted devices which access system settings and increase the device’s battery life by optimizing it to a great extent.

**4.** **Installing Unsupported apps**

There are lots of apps which are not supported by your device, but after rooting your Smartphone you can install them by editing your build.prop file.

**5. Backing Up your device.**

After rooting your Smartphone you can also install a custom bootloader and perform an android backup of your device. It is a backup which when restored, restores your device to the same state as it was performing during the backup.

So these are the advantages and benefits which you are going to get when you root your Android Smartphone. But if you feel I missed any good advantage, then feel free to share with us.

## **Disadvantages of Rooting**an**Android Phone**

But if rooting a device has advantages, it also has some disadvantages which you can read below

1.**Bricking**

The number one reason not to root your device is the potential risk of “bricking” it. As mentioned earlier, “bricking” your device means screwing up your phone software so badly that your phone can no longer function properly and is pretty much as useless as a brick.

2. **Loose Warranty**

You would likely need to purchase a new Android device from the manufacturer of your device will void the warranty after any attempts at rooting. But you can get your guarantee back by restoring (reinstall) a stock build recovery and to un-root

3. **Malicious Software**

There is an increased risk of unknowingly installing malicious software when you root an Android device. That means worms, viruses, spyware and Trojans can infect the rooted Android software if it’s not protected by effective mobile antivirus for Android.

So now it’s all up to you whether you want to root your Android phone or you want to stick with stock ROM. If you have any query or suggestion related to this topic, please share with us in the comment section

**To summarize the discussion:**

Rooting can give the user full access control  
Multiple OS can be switched in the rooted device  
Updates are not available in the rooted device but unrooted devices are subjected to updates  
Any applications can be installed in rooted devices and battery life is enhanced in the rooted device  
Backup can be taken in the rooted device based on our needs  
Unrooted devices are genuine devices  
Rooting can cause the ROM to become unstable

**7.How to root your phone?**  
Tap it seven times, at which point you should see the message, "You are now a developer!" With that done, tap Settings, About Phone, Developer Options, and then tick USB debugging." Then tap OK to approve the setting change. Step 3: Run Android Root on your PC, then connect your phone via its USB sync cable

**8.What is an SMALI file?**

Developer file written in Smali, the assembly language used by the Android Dalvik Virtual Machine; usually created by decompiling [.DEX](https://fileinfo.com/extension/dex) (Dalvik Executable) files, which are the executables included in Android apps ([.APK](https://fileinfo.com/extension/apk) files); usually used for low-level inspection of an Android app's contents, or for hacking Android apps.

**More Information**

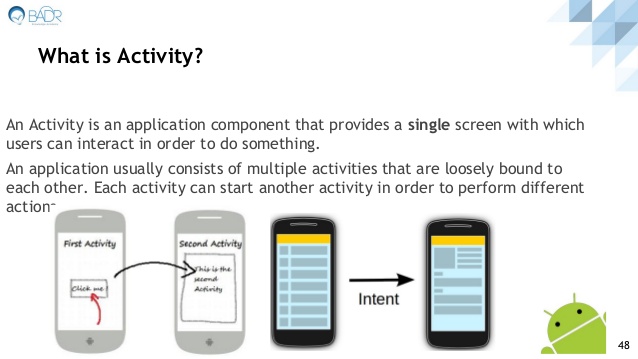
You can decompile a DEX file into its corresponding SMALI files using baksmali, a program included with the [smali](http://code.google.com/p/smali/" \t "_blank) package. For reading SMALI files, there is a Smali syntax highlighting addon for Notepad++ distributed by the [Android Cracking blog](http://androidcracking.blogspot.com/2011/02/smali-syntax-highlighting-for-notepad.html).

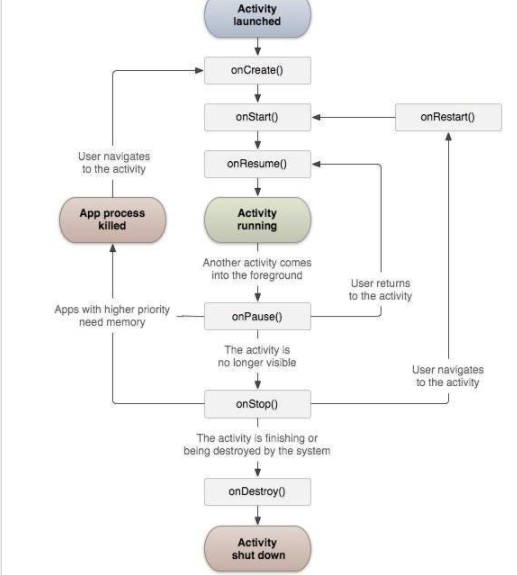
NOTE: "Smali" stands for assembler in Icelandic. Iceland has a village named Dalvik from which the Dalvik Virtual machine was named. Similarly, "Baksmali" stands for disassembler.

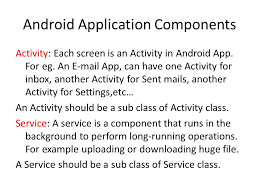
**9.What is dex.files?**

Android programs are compiled into .dex (Dalvik Executable) files, which are in turn zipped into a single .apk file on the device. . dex files can be created by automatically by Android translating compiled applications written in the Java programming language

**10.What is Activity in android application**

****

****

****

**11.What is Shared preference files?**

SharedPreferences is an API from Android SDK to store and retrieve application preferences. SharedPreferences are simply sets of data values that stored persistently. Persistently which mean data you stored in the SharedPreferences are still exist even if you stop the application or turn off the device.

SharedPreferences are stored in an xml file in the app data folder, i.e.SharedPreferences added during runtime are not stored in the Eclipse project. Preferences can either be set in code or can be found in res/xml/preferences.xml.

**12.What is SQLLite?**

SQLite is a opensource SQL database that stores data to a text file on a device. Android comes in with built in SQLite database implementation. SQLite supports all the relational database features

## **Why SQLite?**

* SQLite does not require a separate server process or system to operate (serverless).
* SQLite comes with zero-configuration, which means no setup or administration needed.
* A complete SQLite database is stored in a single cross-platform disk file.
* SQLite is very small and light weight, less than 400KiB fully configured or less than 250KiB with optional features omitted.
* SQLite is self-contained, which means no external dependencies.
* SQLite transactions are fully ACID-compliant, allowing safe access from multiple processes or threads.
* SQLite supports most of the query language features found in SQL92 (SQL2) standard.
* SQLite is written in ANSI-C and provides simple and easy-to-use API.
* SQLite is available on UNIX (Linux, Mac OS-X, Android, iOS) and Windows (Win32, WinCE, WinRT).

**13.What is Code Obfuscation?**

Obfuscation is the practice of making something difficult to understand. Programming code is often obfuscated to protect intellectual property and prevent an attacker from reverse engineering a proprietary software program

**14.What tool you will use to find code abfuscation?**

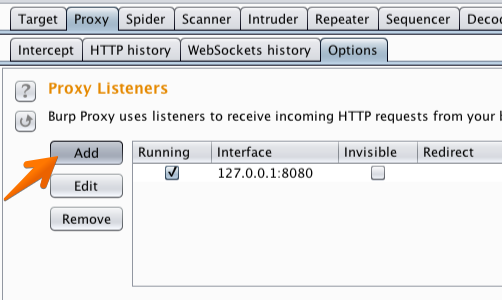
dex2jar and jd gui

**15.how to capture android traffic in burp suit**

**Configuring an Android Device to Work With Burp**

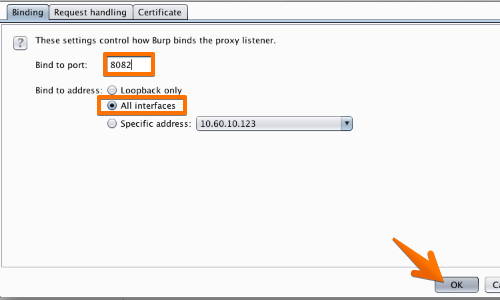
To test web applications using an Android device you need to configure your Burp Proxy listener to accept connections on all network interfaces, and then connect both your device and your computer to the same wireless network. If you do not have an existing wireless network that is suitable, you can [set up an ad-hoc wireless network](https://support.portswigger.net/customer/portal/articles/1841150-Mobile%20Set-up_Ad-hoc%20network_OSX.html).

Configure the Burp Proxy listener

****

In Burp, go to the “Proxy” tab and then the “Options” tab.

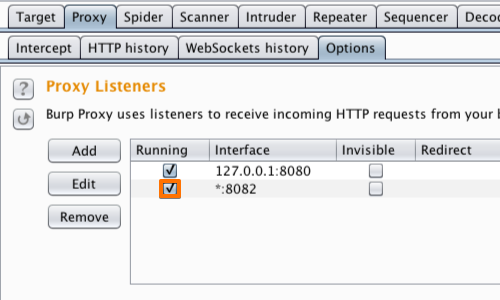
In the “Proxy Listeners" section, click the “Add” button.

****

In the "Binding" tab, in the “Bind to port:” box, enter a port number that is not currently in use, e.g. “8082”.

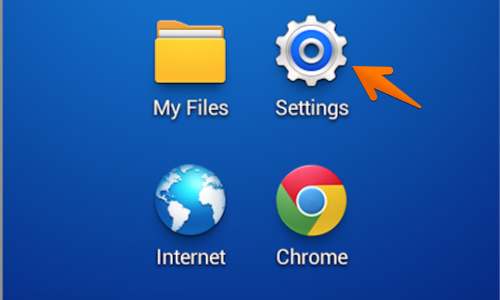
Then select the “All interfaces” option, and click "OK".

**Note:** You could alternatively edit the existing default proxy listener to listen on all interfaces. However, using different listeners for desktop and mobile devices enables you to filter these in the Proxy history view.

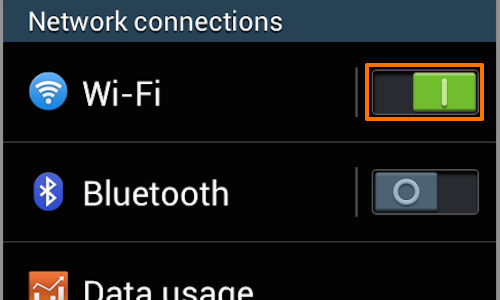
****

The Proxy listener should now be configured and running.

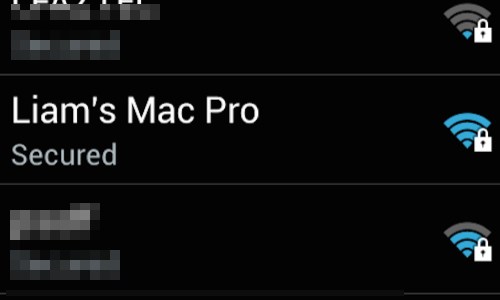
 Configure your device to use the proxy

****

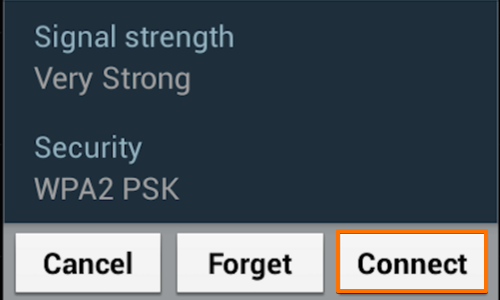
In your Android device, go to the“Settings” menu.

****

If your device is not already connected to the wireless network you are using, then switch the "Wi-Fi" button on, and tap the “Wi-Fi” button to access the "Wi-Fi" menu.

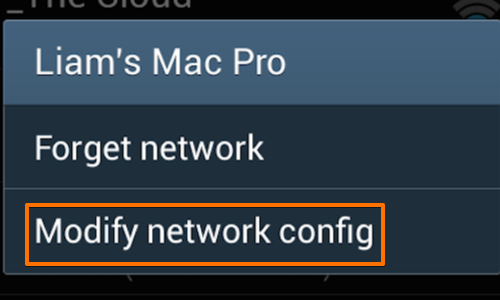
****

In the "Wi-Fi networks" table, find your network and tap it to bring up the connection menu.

****

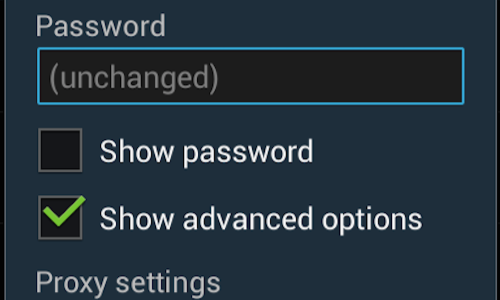
Tap "Connect".

If you have configured a password, enter it and continue.

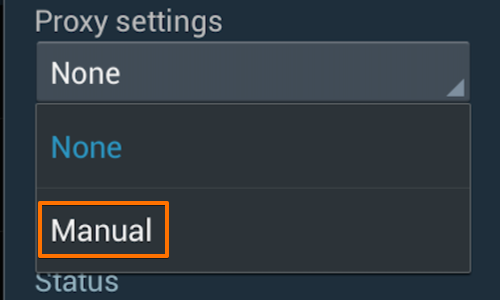
****

Once you are connected hold down on the network button to bring up the context menu.

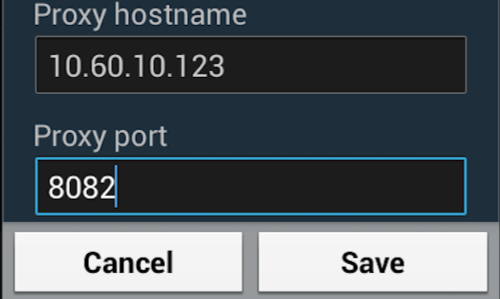
Tap “Modify network config”.

****

Ensure that the “Show advanced options” box is ticked.

****

Change the “Proxy settings” to “Manual” by tapping the button.

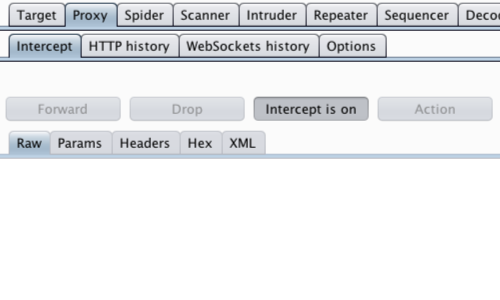
****

Then enter the IP of the computer running Burp into the “Proxy hostname”.

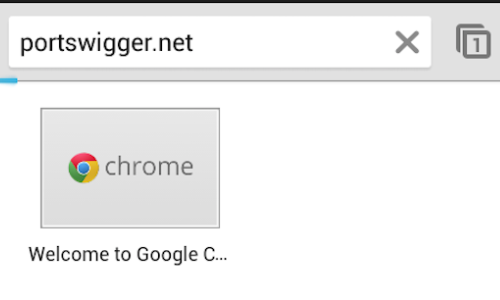
Enter the port number configured in the “Proxy Listeners” section earlier, in this example “8082”.

Tap "Save".

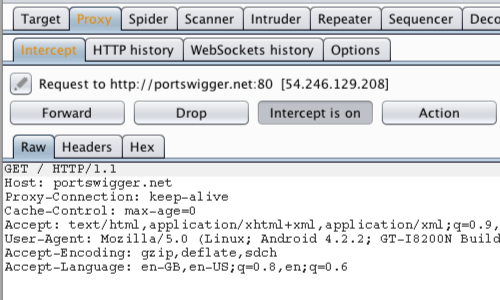
### Test the configuration

****

In Burp, go to the "Proxy Intercept" tab, and ensure that intercept is “on” (if the button says “Intercept is off" then click it to toggle the interception status).

****

Open the browser on your Android device and go to an HTTP web page (you can visit an HTTPS web page when you have [installed Burp's CA Certificate in your Android device](https://support.portswigger.net/customer/portal/articles/1841102-Mobile%20Set-up_Android%20Device%20-%20Installing%20CA%20Certificate.html).)

****

The request should be intercepted in Burp.

**16.List the tools for android application testing?**

* Apk tool
* Dex2jar and JDGUI
* Drozer
* SQLite Database Browser
* CatLog App
* Class Name DE obfuscator
* Xposed Framework
* Mobile-Security-Framework-MobSF

**17. Mobile Owasp top ten**

**M1 - Improper Platform Usage**

This category covers misuse of a platform feature or failure to use platform security controls. It might include Android intents, platform permissions, misuse of TouchID, the Keychain, or some other security control that is part of the mobile operating system. There are several ways that mobile apps can experience this risk.

**M2 - Insecure Data Storage**

This new category is a combination of M2 + M4 from Mobile Top Ten 2014. This covers insecure data storage and unintended data leakage.

**M3 - Insecure Communication**

This covers poor handshaking, incorrect SSL versions, weak negotiation, cleartext communication of sensitive assets, etc.

**M4 - Insecure Authentication**

This category captures notions of authenticating the end user or bad session management. This can include:

* Failing to identify the user at all when that should be required
* Failure to maintain the user's identity when it is required

Weaknesses in session management

**M5 - Insufficient Cryptography**

The code applies cryptography to a sensitive information asset. However, the cryptography is insufficient in some way. Note that anything and everything related to TLS or SSL goes in M3. Also, if the app fails to use cryptography at all when it should, that probably belongs in M2. This category is for issues where cryptography was attempted, but it wasn't done correctly

**M6 - Insecure Authorization**

This is a category to capture any failures in authorization (e.g., authorization decisions in the client side, forced browsing, etc.). It is distinct from authentication issues (e.g., device enrolment, user identification, etc.).

If the app does not authenticate users at all in a situation where it should (e.g., granting anonymous access to some resource or service when authenticated and authorized access is required), then that is an authentication failure not an authorization failure.

**M7 - Client Code Quality**

This was the "Security Decisions Via Untrusted Inputs", one of our lesser-used categories. This would be the catch-all for code-level implementation problems in the mobile client. That's distinct from server-side coding mistakes. This would capture things like buffer overflows, format string vulnerabilities, and various other code-level mistakes where the solution is to rewrite some code that's running on the mobile device

**M8 - Code Tampering**

This category covers binary patching, local resource modification, method hooking, method swizzling, and dynamic memory modification.Once the application is delivered to the mobile device, the code and data resources are resident there. An attacker can either directly modify the code, change the contents of memory dynamically, change or replace the system APIs that the application uses, or modify the application's data and resources. This can provide the attacker a direct method of subverting the intended use of the software for personal or monetary gain.

**M9 - Reverse Engineering**

This category includes analysis of the final core binary to determine its source code, libraries, algorithms, and other assets. Software such as IDA Pro, Hopper, otool, and other binary inspection tools give the attacker insight into the inner workings of the application. This may be used to exploit other nascent vulnerabilities in the application, as well as revealing information about back end servers, cryptographic constants and ciphers, and intellectual property.

**M10 - Extraneous Functionality**

Often, developers include hidden backdoor functionality or other internal development security controls that are not intended to be released into a production environment. For example, a developer may accidentally include a password as a comment in a hybrid app. Another example includes disabling of 2-factor authentication during testing.

**What is drozer?**

Drozer is the leading security assessment framework for the Android platform.

Drozer allows us to assume the role of an Android app and interact with other apps. It can do anything that an installed application can do, such as make use of Android’s Inter-Process Communication (IPC) mechanism and interact with the underlying operating system.

Drozer also helps us to remotely exploit Android devices, by building malicious files or web pages that exploit known vulnerabilities. The payload that is used in these exploits is a rogue drozer agent that is essentially a remote administration tool. Depending on the permissions granted to the vulnerable app, drozer can install a full agent, ject a limited agent

**What is Manifest files in android?**

**App Manifest**

Every application must have an AndroidManifest.xml file (with precisely that name) in its root directory. The manifest file provides essential information about your app to the Android system, which the system must have before it can run any of the app's code.

Among other things, the manifest file does the following:

* It names the Java package for the application. The package name serves as a unique identifier for the application.
* It describes the components of the application, which include the activities, services, broadcast receivers, and content providers that compose the application. It also names the classes that implement each of the components and publishes their capabilities, such as the [Intent](https://developer.android.com/reference/android/content/Intent.html) messages that they can handle. These declarations inform the Android system of the components and the conditions in which they can be launched.
* It determines the processes that host the application components.
* It declares the permissions that the application must have in order to access protected parts of the API and interact with other applications. It also declares the permissions that others are required to have in order to interact with the application's components.
* It lists the [Instrumentation](https://developer.android.com/reference/android/app/Instrumentation.html) classes that provide profiling and other information as the application runs. These declarations are present in the manifest only while the application is being developed and are removed before the application is published.
* It declares the minimum level of the Android API that the application requires.
* It lists the libraries that the application must be linked against.