

# MOBILE APPLICATION SECURITY

Prepared by: Dr. Alia Alabdulkarim

# The Evolution of Mobile Applications

---

- 🔒 The first mobile phone applications were developed by handset manufacturers; documentation was sparse.
- 🔒 And little information existed in the public domain on the operating internals.
- 🔒 This can perhaps be attributed to a fear from the vendors that opening the platforms to third-party development might have exposed trade secrets in what was not yet a fully developed technology.
- 🔒 The early applications were similar to many of the manufacturer-based apps found on today's phone, such as contacts and calendars, and simple games such as Nokia's popular *Snake*.



# The Evolution of Mobile Applications

---

- 🔒 When **smartphones** emerged, application development really began to take off.
- 🔒 Third-party application development came to fruition in 2008 when Apple announced the first third-party application distribution service, the **App Store**.
- 🔒 Google closely followed with the Android Market, known today as **Google Play**.

# Common Mobile Application Functions

---

🔒 In the combined Apple and Google distribution stores alone, there are believed to be **more than 2 million** applications covering a wide range of functions, including some of the following:

- 🔑 Online banking (E.g. SambaMobile)
- 🔑 Shopping (E.g. Amazon)
- 🔑 Social networking (E.g. Instagram)
- 🔑 Streaming (E.g. Youtube)
- 🔑 Instant Messaging (E.g. WhatsApp)
- 🔑 Voice chat (E.g. Zoom)
- 🔑 E-mail (E.g. Gmail)
- 🔑 File sharing (E.g. Dropbox)
- 🔑 Games (E.g. *Angry Birds*)

# Common Mobile Application Functions

---

🔒 Mobile applications have been widely adopted in the business world to support key business functions.

🔒 Many of these applications provide access to highly sensitive corporate data, including some of the following:

- 🔑 Document storage applications allowing users to access sensitive business documents.

- 🔑 Travel and expenses applications allowing users to create, store, and upload expenses to internal systems.

- 🔑 HR applications allowing users to access the payroll, time slips, holiday information, and other sensitive functionality

🔒 How about sensitive data in the personal level?



# Benefits of Mobile Applications

---

🔒 They offer organizations the opportunity to **reach out to end users almost all the time** and to much wider audiences due to the popularity of smartphones.

🔒 However, several technical factors have also contributed to their success:

🔑 The foundations of mobile applications are built on existing and popular protocols. In particular, the use of HTTP is widely adopted in mobile deployments and is well understood by developers.

🔑 The technical advancements of smartphones have allowed mobile applications to offer more advanced features and a better user experience: screen resolution, battery life,...etc.

🔑 Improvements in cellular network technologies have resulted in significant speed increases: 3G, 4G, and now 5G



# Mobile Application Security

---

🔒 Mobile applications are affected by a range of security vulnerabilities, many of which are inherited from traditional attacks against web and desktop applications.

🔒 However, several other classes of attack are specific to the mobile area and arise due to the way in which mobile applications are used.

🔒 Consider the **possible attack surfaces for a mobile application** that developers should be aware of and look to defend against:

🔑 Network

🔑 Theft

🔑 Other applications

🔑 Entry points



# Mobile Application Security

## Attack Surfaces for Mobile applications - Network

---

- 🔒 Most mobile applications perform some kind of network communication.
- 🔒 This communication may often occur over an **insecure or untrusted public networks** such as: hotel or café Wi-Fi.
- 🔒 Unless data is adequately secured in transit(how?), it may expose an application to a number of possible risks, including **disclosure of sensitive data** and **injection attacks**.



# Mobile Application Security

## Attack Surfaces for Mobile applications - Theft

---

- 🔒 Mobile devices are carried with you wherever you go, creating many opportunities for them to be **lost or stolen**.
- 🔒 Mobile application developers must recognize the risks from **data recovery attempts** against a device's file system.
- 🔒 Any residual content that an application leaves on the file system, whether it's through persistent storage or temporary caching, can potentially expose sensitive data to an attacker.



# Mobile Application Security

## Attack Surfaces for Mobile applications - Other Apps

---

🔒 Threats originating from the host device.

🔒 **Malware** is common within the mobile space, particularly in the **unofficial 3<sup>rd</sup> party markets**, and developers must be conscious of attacks from other applications.

# Mobile Application Security

## Attack Surfaces for Mobile applications - Entry Points

---

🔒 Mobile applications can **receive input from a large number of possible sources**, which creates a significant number of possible entry points.

🔒 For example, it is common to see applications accept data from one or many of the following :

🔑 Bluetooth

🔑 Camera

🔑 Microphone

🔑 SMS

🔑 USB



🔑 QR codes

# The OWASP Mobile Security Project











## Top 10 Mobile Risks [2]

---

### Final List 2014

-  M1: Weak Server Side Controls
-  M2: Insecure Data Storage
-  M3: Insufficient Transport Layer Protection
-  M4: Unintended Data Leakage
-  M5: Poor Authorization and Authentication
-  M6: Broken Cryptography
-  M7: Client Side Injection
-  M8: Security Decisions Via Untrusted Inputs
-  M9: Improper Session Handling
-  M10: Lack of Binary Protections

### Final List 2016

-  M1: Improper Platform Usage
-  M2: Insecure Data Storage
-  M3: Insecure Communication
-  M4: Insecure Authentication
-  M5: Insufficient Cryptography
-  M6: Insecure Authorization
-  M7: Client Code Quality
-  M8: Code Tampering
-  M9: Reverse Engineering
-  M10: Extraneous Functionality

# Key Problem Factors

---

🔒 Vulnerabilities typically occur when an application must handle or protect sensitive data or process data that has originated from an untrusted source.

🔒 However, several other factors have combined to intensify the problem:

- 🔑 ***Underdeveloped Security Awareness***

- 🔑 ***Ever-Changing Attack Surfaces***

- 🔑 ***Economic and Time Constraints***

- 🔑 ***Custom Development***

# Key Problem Factors

## Underdeveloped Security Awareness

---

- 🔒 Unlike most web applications where the attack surface is limited to user-derived input, mobile application developers have a number of different scenarios to consider and protect against.
- 🔒 Developers cannot trust the host operating system or even their own application.
- 🔒 Awareness of the many attack surfaces and defensive protections is limited and not well understood within the mobile development communities.



# Key Problem Factors

## Ever-Changing Attack Surfaces

---

🔒 Research into mobile device and application security is a continually evolving area in which ideas are regularly challenged and new threats and concepts discovered.

# Key Problem Factors

## Economic and Time Constraints

---

🔒 Most application development projects are governed by strict resource and time constraints, and mobile application development is no exception.

🔒 The economics of an application development project often mean that having permanent security expertise throughout the development process is infeasible for companies, particularly in smaller organizations that on the whole tend to leave security testing until late in a project's lifecycle.

# Key Problem Factors

## Custom Development

---

🔒 When organizations are regularly developing multiple applications, components that have been thoroughly tested will find themselves being reused across projects; this often promotes more robust and secure code.

🔒 In these cases, the main project developers may not have full awareness of the code and misuse could lead to the introduction of security defects.

# ***Android*** Application Fundamentals [3]

---

🔒 Android apps can be written using **Kotlin, Java, and C++** languages.

🔒 **Environment:** Android Studio

🔒 **Android package:** an archive file with an **.apk** extension. It is the file that Android devices use to install an app. It contains all the contents of an Android app.


# *Android* Application Fundamentals [3]

---

 There are **four** different types of app **components** for **Android apps**:

1. **Activities**: the entry points for interacting with the user. An activity represents a single screen with a user interface. For example, the "main" activity starts when the user taps your app's icon. An email app might have one activity that shows a list of new emails and another activity to read an email.
2. **Services**: the components that runs in the background without user interfaces. For example, a service can play music in the background while the user is using a different app.
3. **Broadcast receivers**: enable the system to deliver events to apps outside of a regular user flow even if the apps aren't currently running. For example, an app can schedule for a notification about an upcoming event.
4. **Content providers**: manage a shared set of app data that you can store in the file system, in an SQLite database, on the web, or on any other persistent storage location that your app can access. For example, the Android system provides a content provider that manages the user's contact information.

# Android Application Fundamentals [3]

 **The manifest file** (AndroidManifest.xml), your app must declare all its components in this file, in addition to:

- 🔑 Identifies any **user permissions** the app requires.
- 🔑 Declares the minimum **API Level** required by the app.
- 🔑 Declares **hardware and software features** required by the app, such as a camera, Bluetooth services.
- 🔑 Declares **API libraries** the app needs to be linked against.

```
1 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
2   .... package="com.example.tapittest_libtest"
3   .... android:versionCode="1"
4   .... android:versionName="1.0" >
5
6   .... <uses-sdk
7   ....     android:minSdkVersion="4"
8   ....     android:targetSdkVersion="15" />
9
10  .... <uses-permission android:name="android.permission.INTERNET"></uses-permission>
11  .... <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"></uses-permission>
12  .... <uses-permission android:name="android.permission.READ_PHONE_STATE"></uses-permission>
13
14  .... <!-- Optional permissions to enable ad geotargeting
15  .... <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"></uses-permission>
16  .... <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"></uses-permission>
17  .... -->
18
19  .... <application
20  ....     android:icon="@drawable/ic_launcher"
21  ....     android:label="@string/app_name"
22  ....     android:theme="@style/AppTheme" >
23  ....     <activity
24  ....         android:name="com.tapit.adview.AdActivity"
25  ....         android:configChanges="keyboard|keyboardHidden|orientation" />
26  ....     <activity
27  ....         android:name=".MainActivity"
28  ....         android:label="@string/title_activity_main" >
29  ....         <intent-filter>
30  ....             <action android:name="android.intent.action.MAIN" />
31
32  ....             <category android:name="android.intent.category.LAUNCHER" />
33  ....         </intent-filter>
34  ....     </activity>
35  .... </application>
36
37 </manifest>
```



# ***Android*** Application Security [3],[4]

---

🔒 Each Android app lives in its own security **sandbox** which isolates its data and code execution from other apps. Apps are protected by the following Android security features:

- 🔑 The Android operating system is a multi-user Linux system in which **each app is a different user**.
- 🔑 Each process has its own virtual machine (VM), so an **app's code runs in isolation from other apps**.
- 🔑 By default, the system assigns each app a unique Linux user ID that is used only by the system and is unknown to the app. Files that belong to an app can be accessed by that app only and the system ensures that **the user ID assigned to that app can access them**.

# *Android* Application Security [3],[4]

---

Other core security features in Android that can help you build secure apps:



Sandbox.



An application framework with robust implementations of common security functionality such as **cryptography**, **permissions**, and **secure IPC**.



Technologies like ASLR, NX, ProPolice, etc to mitigate risks associated with common **memory management errors**.



An **encrypted file system** that can be enabled to protect data on lost or stolen devices.



User-granted **permissions** to restrict access to system features and user private data.

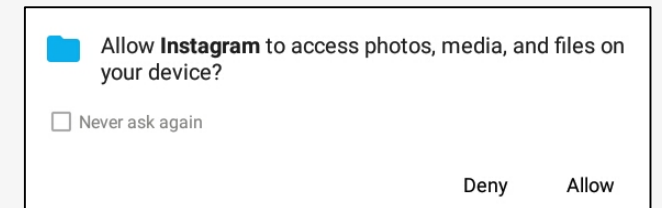
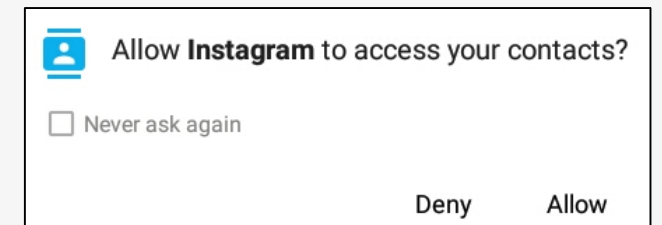
# *Android* Application Security

## Apps Permissions(Reading Assignment) [5]

---

🔒 Apps permissions are used to protect users' privacy.

🔒 Android apps must request permission to **access sensitive user data** (such as contacts and files), as well as certain **system features** (such as camera and microphone).



# Android Application Security

## Apps Permissions(Reading Assignment) [5]

---

🔒 Depending on the feature, the system might **grant the permission automatically** or **ask the user to allow the request**:

Normal Permissions	Dangerous Permissions
Used when an app needs to access data or resources outside its own sandbox, and there's <b>very little risk</b> to the user's <b>privacy</b>	Used when an app wants data or resources that involve the user's <b>private information</b> or <b>can affect user's privacy</b> .
Automatically granted for the app at install time.	The app must prompt the user to grant permission at runtime.
Example: set the time zone.	Example: access the user's location or camera.

# The Future of Mobile Application Security

---

- 🔒 The consequence of the growing mobile revolution will only place further emphasis on understanding the security threats that mobile deployments face as well as effective ways of addressing them.
- 🔒 The current threats to mobile security are at present **not** well understood, particularly in the development communities.
- 🔒 It is expected that classic vulnerabilities such as **insecure data storage** and **insufficient transport security** will continue to be prevalent for the immediate future.
- 🔒 Mobile application security is a **continually evolving** landscape and we fully expect new categories of attacks to arise following advances in mobile technologies.

# References

---

[1] The Mobile Application Hacker's Handbook

🔑 Chapter 1

[2] <https://owasp.org/www-project-mobile-top-10/>

[3] <https://developer.android.com/guide/components/fundamentals>

[4] <https://developer.android.com/training/articles/security-tips>

[5] <https://developer.android.com/training/basics/permissions>