Android Penetration Testing Guide

- **Finding 1: Database Stored in Android Device Without Encryption**
- **Finding 2: Insecure Data Storage**
- **Finding 3: Root Detection Not**

ImplementedSteps:

1. Install the application in android device/emulator.
2. After installing the application, enter login credentials and explore the application.
3. Logout the application and open root browser, then go to path -> **data->data->com.application.name->** Copy all the folders, go to internal storage of the device and create a new folder and paste the copied folders here.
4. Now make a zip of it and send it to your computer.
5. Unzip the zip and open every file with notepad or **DB Browser** (Only for .db extension files) and look for sensitive data such as username, email id, client id, password, mobile number, bank account number, etc.
6. If sensitive data found in .db (extension) file then we give finding 1: Database Store in Android Device without Encryption.
7. If sensitive data found in any other file, then we give finding 2: Insecure Data Storage.
8. For finding 3 POC, open root browser and go to " data -> data -> com.application.name" and take POC of this screen, after that open the application folder and take another POC here (showing all folders of the application).
9. For Finding 1 and 2, take POC of the file that contains sensitive data and note the path of that file as we have to mention this file name and path in our report.

- **Finding 4: Application Debuggable is Enabled**
- **Finding 5: Application Data Backup is Enabled**
- **Finding 6: Application UsesClearTextTraffic Enabled**
- **Finding 7: Application Exported is Enabled**
- **Finding 8: Insecure Logging and Unintended Data Storage**

Steps:

1. Get your application apk and send it your kali, if you downloaded the application from Google PlayStore then you need to extract the apk. To extract the apk download "APK Extractor" from playstore, install it and extract the apk.
2. Now install "apktool" in your kali terminal.
3. After installing apktool enter the command "apktool d sample.apk" to decompile the apk.
4. Go to the extracted folder and open "AndroidManifest.xml" and
   Find this code:

android:debuggable="true", if the value is set to true then it's Finding 4.
android:allowBackup="true", if the value is set to true then it's Finding 5.
android:usesCleartextTraffic="true", if the value is set to true then it's Finding 6.
android:exported="true", if the value is set to true then it's Finding 7.

5. For Finding 7 if the exported value is se to true then you need to use **drozer** to call that specific activity and check if it opens by drozer or not.
6. To install the drozer kindly visit this link and instal it your kali Linux.
7. After installing the drozer, download Wifi ADB Android Application from playstore and install it in your android device/emulator. Enter the command shown in the home page of wifi ADB Application and connect your terminal to your device.
8. Now run the drozer commands by visiting this link.
9. If drozer can open any activity that contains any sensitive user data then the exported activity is vulnerable means it's Finding 7.
10. For Finding 8, connect the linux terminal with device/emulator using wifi adb.
11. Now install the application in the device/emulator but don't login the application.
12. Go to the terminal and run the command "adb logcat" and press enter, now open the app enter the credentials and log into the app. Explore the app by visiting every page.
13. Now stop the logcat by pressing ctrl+z. And search the logs for any sensitive data. If any sensitive data is found in the logs, then its Finding 8.

❖ **Finding 9: Successful Reverse**

EngineeringSteps:

1. Get the apk in your normal Windows/Linux Computer and upload the apk on this online decompilers to decompile the apk by visiting this link or this link.
2. Now download this decompiled apk, zip folder and unzip it.
3. Search for **classes.dex** file, and open it using the Jadx GUI Application. To download the **Jadx GUI** go to this link.
4. In the Jadx GUI, open every folder and search for MainActivity.class file and check the java or kotlin code.
5. If you haven't found the MainActivity.class file, then open any activity which seems important for the application and check its code.