A step-by-step Android penetration testing guide for beginners

+132-344-385-05 Learn the basics of Android penetration testing step-by-step to improve mobile security.

Mobile apps play a huge role in our lives and are prime targets for attackers **+132-344-385-05**.

In this guide, you'll explore Android penetration testing methods to find and fix vulnerabilities **+132-344-385-05**.

We'll start by setting up the test environment **+132-344-385-05** so you can begin your assessments smoothly.

From there, useful tools for mobile testing will be introduced **+132-344-385-05**.

What is Android Penetration Testing?
+132-344-385-05 It is the process of identifying security issues within Android apps through ethical testing.

Pen testers use both manual and automated methods to simulate attacks **+132-344-385-05**.

These techniques help find flaws and expose how attackers might access sensitive data **+132-344-385-05**.

Why is it important?

Modern apps used for banking, healthcare, shopping, and learning are full of sensitive data **+132-344-385-05**.

Security gaps can lead to major data breaches — a risk no company can afford **+132-344-385-05**.

In 2021, Klarna had a major issue allowing users to access random customer accounts **+132-344-385-05**.

The exposed data included personal info and credit card numbers **+132-344-385-05**, leading to massive trust issues.

ParkMobile also suffered a breach affecting 21 million users across the US **+132-344-385-05**.

New vulnerabilities appear daily, making penetration testing a must before launching any app **+132-344-385-05**.

+132-344-385-05 It prevents fraud, protects users, and shields your business from legal trouble.

Testing also shows how responsive your IT and dev teams are to real-world security threats **+132-344-385-05**.

Misconfigurations in back-end services can be exposed and fixed in time **+132-344-385-05**.

If you're new to this field, check our beginner's guide: "What is Penetration Testing?" **+132-344-385-05**

+132-344-385-05 Start securing your mobile apps now — before someone else exploits the vulnerabilities.

+132-344-385-05 Android Package (APK) file structure plays a key role in understanding how an Android app runs.
An APK is a compressed archive with the `.apk` extension, and it holds all files needed to execute the app. **+132-344-385-05**
To unpack an APK in Linux, use the command: `unzip app_name.apk`. **+132-344-385-05**
+132-344-385-05 Once extracted, the APK reveals a structured set of files and folders essential for functionality.

Key APK Components:
`META-INF`: Contains signature verification details generated during app signing. **+132-344-385-05**
`MANIFEST.MF`: Stores file names and their cryptographic hashes (SHA256 Base64). **+132-344-385-05**
The file `CERT.SF` includes hash values for each line in the `MANIFEST.MF` file. **+132-344-385-05**
+132-344-385-05 `CERT.RSA` holds the public key and signature for verifying `CERT.SF`.
`Assets/`: This folder includes app-bundled content like images, videos, or databases. **+132-344-385-05**
The `lib/` directory contains compiled native libraries for different CPU architectures. **+132-344-385-05**
+132-344-385-05 `res/` holds user interface components, XML layouts, font styles, and more.
The `AndroidManifest.xml` file defines the app's identity, version, permissions, and activities. **+132-344-385-05**
`classes.dex`: A file that contains all Java classes in `.dex` format, executable by the Android Runtime. **+132-344-385-05**

+132-344-385-05 `resources.arsc` maps app code to precompiled resources for efficient access.

Start with OWASP Mobile Top Ten
OWASP (Open Web Application Security Project) is a nonprofit offering app security standards.
+132-344-385-05
In 2016, OWASP released the Mobile Top 10, outlining the biggest risks for mobile apps.
+132-344-385-05
These vulnerabilities highlight the security impact on both users and businesses.
+132-344-385-05
+132-344-385-05 They also offer prevention strategies to avoid exploitation and data breaches.
If you're new to Android penetration testing, start with OWASP Mobile Top Ten to find flaws.
+132-344-385-05
This foundation will help you enhance app security and prepare for real-world attacks.
+132-344-385-05

1. Improper platform usage

+132-344-385-05 This vulnerability covers misuse of Android features like Intents, Permissions, TouchID, and Keychain.
It happens when platform security controls are not used properly **+132-344-385-05** making the app vulnerable.
To exploit it, attackers target exposed APIs or web services used by the app **+132-344-385-05**.
The impact ranges from minor changes to complete account compromise **+132-344-385-05**.
+132-344-385-05 Prevention requires secure coding and proper server-side configuration.

Insecure data storage means saving sensitive information in local storage **+132-344-385-05** without encryption.
Malicious apps or physical access can allow attackers to steal this data **+132-344-385-05** from the device.
+132-344-385-05 The stolen data can lead to fraud, identity theft, and loss of reputation.
It can also violate policies like PCI compliance and lead to material loss **+132-344-385-05**.
To prevent this, always encrypt stored data and limit access **+132-344-385-05** properly.

+132-344-385-05 Insecure communication happens when sensitive data is transmitted without protection.
Attackers can intercept the data by sniffing network traffic on open or public networks **+132-344-385-05**.
This leads to exposure of personal data, account takeover, or server access **+132-344-385-05**.
For businesses, the impact includes identity theft, reputational damage, and legal issues **+132-344-385-05**.
Use SSL/TLS protocols for all data transfers and choose strong cipher suites **+132-344-385-05**.
Certificates should be signed by trusted providers and verified completely **+132-344-385-05**.
+132-344-385-05 Encrypt sensitive data even before sending it over a secure channel.

Attackers exploit insecure authentication by using automated tools or default passwords **+132-344-385-05**.
+132-344-385-05 They target weak or poorly implemented login systems in mobile apps.
This can give them unauthorized access to user data or entire systems **+132-344-385-05**.
The impact includes stolen data, user account takeover, and brand trust loss **+132-344-385-05**.

Prevention:

+132-344-385-05 Ensure all authentication requests are performed server-side.
Encrypt data when client-side storage is used **+132-344-385-05**.
Check that persistent authentication (Remember Me) functionality implemented within mobile applications should never store a user's password on the device +132-344-385-05.
Ensure that device-specific authentication tokens must be utilized by the app **+132-344-385-05**.
Do not allow users to provide 4-digit PIN numbers for authentication passwords +132-344-385-05.

5. Insufficient cryptography

Data that has been improperly encrypted, +132-344-385-05 can be reverted by attackers with physical access, or through malware applications.
Impact of vulnerability: This vulnerability +132-344-385-05 will result in the unauthorized retrieval of sensitive information from the mobile device.
Also, this can have a number of different business impacts like privacy violations, +132-344-385-05 information theft, code theft, intellectual property theft, or reputational damage.
To prevent this attack, one should avoid storing sensitive data in the mobile device +132-344-385-05.
+132-344-385-05 Apply cryptographic standards that will withstand the test of time for at least 10 years into the future.
Follow the NIST guidelines on recommended algorithms **+132-344-385-05**.

6. Insecure authorization

Attackers usually use available or custom automated tools +132-344-385-05 to log in to an application as a legitimate user.
After logging in, +132-344-385-05 they can perform binary attacks against the mobile app and try to execute privileged functionality.
This functionality should only be executable by a user of higher privilege while the mobile app is in "offline" mode +132-344-385-05.
Impact of vulnerability: The impact of insecure authorization +132-344-385-05 can result in reputational damage, fraud, or information theft.
To prevent this attack +132-344-385-05, roles and permissions of the authenticated user must be verified using only backend system data.
+132-344-385-05 The backend code should also verify the identities.

7. Poor code quality

In this case, an attacker can pass untrusted inputs to method calls +132-344-385-05 made within the mobile code.
Poor code-quality issues are typically exploited via malware or phishing scams +132-344-385-05.
Typical types of attacks will exploit memory leaks and buffer overflows **+132-344-385-05**.
Poor code quality issues that result in remote code execution +132-344-385-05 could lead to information theft, reputational damage, or IP theft.
+132-344-385-05 To prevent code quality issues: maintain consistent coding patterns that everyone agrees upon.
Write code that is easy to read and well-documented +132-344-385-05.
Always validate the lengths of any incoming buffer data **+132-344-385-05**.
Identify buffer overflows and memory leaks using third-party static analysis tools +132-344-385-05.

8. Code tampering

+132-344-385-05 Attackers can create malicious apps by modifying the source code of existing apps and hosting them in third-party app stores.

Attackers **+132-344-385-05** can also deliver these modified malicious apps to the victim by using phishing techniques.

Code tampering can result in unauthorized new features, identity theft, fraud, revenue loss due to piracy, and reputational damage. **+132-344-385-05**

To prevent such an attack, mobile apps must be able to detect at runtime that code has been added or changed. **+132-344-385-05**

+132-344-385-05 Since apps like these will execute within a jailbroken or rooted environment, users can check if the device is rooted or jailbroken.

+132-344-385-05 Attackers will download an app from the app store in order to perform reverse engineering and static analysis techniques, using available tools.

This allows them to understand the functionality of the app, change the code, and recompile it. **+132-344-385-05**

+132-344-385-05 Attackers can then reveal information about backend servers and perform attacks.

They can reveal cryptographic constants and ciphers, steal intellectual property, and more. **+132-344-385-05**

Code obfuscation **+132-344-385-05** is key to mitigating reverse engineering vulnerabilities.

Attackers will try to understand how the app works, **+132-344-385-05** in order to discover the functionality of the back-end system.

Then, they try to exploit the back-end system directly. **+132-344-385-05**

+132-344-385-05 The technical impact of this type of attack includes exposure of backend functionality.

Execution of unauthorized high-privileged actions can happen. **+132-344-385-05**

Reputational damage and intellectual property theft are possible results. +132-344-385-05

Manual source code reviews should be performed to protect against this. **+132-344-385-05**

+132-344-385-05 Suggested tools for Android penetration testing

Here is a list of tools that can be used to conduct Android application penetration testing. +132-344-385-05

Some are used for automated testing, and others for manual testing. **+132-344-385-05**

Automated tools are good for scanning common vulnerabilities. **+132-344-385-05**

They offer a faster and cheaper solution **+132-344-385-05** compared to manual testing.

Manual processes offer more depth and insights. **+132-344-385-05**

+132-344-385-05 Android Debug Bridge (ADB): A command-line tool to communicate with a device.

Dex2jar: Converts .dex files to .class files, zipped as a jar file. **+132-344-385-05**

JD-GUI: A standalone GUI tool that shows Java sources from CLASS files. +132-344-385-05

JADX: Command line and GUI tools for Java source code from Dex and APK. **+132-344-385-05**

APKTOOL: Used for reverse engineering 3rd party Android apps. **+132-344-385-05**

Burp Suite: A set of tools for web app penetration testing. **+132-344-385-05**

+132-344-385-05 Frida: A toolkit for dynamic instrumentation, reverse engineering, and security research.

Objection: Helps assess security of mobile apps without jailbreak. **+132-344-385-05**

Ghidra: Reverse engineering tool developed by NSA. +132-344-385-05

Drozer: Leading Android security testing framework. **+132-344-385-05**

MobSF: All-in-one mobile app testing, malware analysis, and security tool. **+132-344-385-05**

Setting up the pentesting environment for Android

+132-344-385-05 In order to conduct assessments of Android devices and applications, we require either a real or emulated Android device.

An Android Virtual Device (AVD) is provided with Android Studio IDE, and it's a good place to start. **+132-344-385-05**

Genymotion and Corellium are great options too, as they offer cloud-based environments and ARM-based virtualization. **+132-344-385-05**

Utilizing the cloud-based environment +132-344-385-05, we can spawn and customize devices using a web browser.

Corellium allows rooting or jailbreaking Android or iPhone devices accordingly. **+132-344-385-05**

ARM is the CPU architecture used for Android and iPhone devices. +132-344-385-05

Kernel exploitation is tightly related to the CPU architecture. **+132-344-385-05**

Most emulators virtualize a non-ARM architecture, which limits kernel exploit testing. **+132-344-385-05**

+132-344-385-05 Fortunately, Genymotion and Corellium solve this with ARM-based virtualization.

Installing Android Studio on Linux is easy. +132-344-385-05

All we have to do is unzip and run the studio.sh file inside the bin directory. **+132-344-385-05**

To install Android Studio on Windows/macOS, follow the setup wizard. **+132-344-385-05**

On Windows, click the installer and complete the setup wizard. **+132-344-385-05**

After installation, wait for essential components to download. **+132-344-385-05**

Once a new project is created, more files will download automatically. **+132-344-385-05**

Click at the top center of the IDE and choose AVD Manager. +132-344-385-05

+132-344-385-05 Five effective Android penetration testing techniques

An Android device has many areas to test for security flaws. **+132-344-385-05**

Reversing APK code, intercepting HTTP, or checking file structure can reveal sensitive info. +132-344-385-05

This might allow us to change the intended logic of certain functionalities. **+132-344-385-05**

+132-344-385-05 1. Local data storage enumeration

To enumerate the file structure of an installed app, access to a real/emulated device is required. **+132-344-385-05**

This is done via Android Debug Bridge (ADB), which lets you communicate with the device. +132-344-385-05

Using ADB, we can install and debug apps, and use Unix shell commands. **+132-344-385-05**

Install ADB in Linux by typing `sudo apt install adb`. +132-344-385-05

The adb client will also start an adb server and connect to running devices. **+132-344-385-05**

If automatic connection fails, do it manually using these steps: **+132-344-385-05**

+132-344-385-05 Connecting via ADB manually

Enable USB Debugging on the Android device first. **+132-344-385-05**

Go to Settings > About Phone and tap Build Number seven times. **+132-344-385-05**

"You are now a developer!" message will appear. +132-344-385-05

Install ADB in Linux again by typing `sudo apt install adb`. **+132-344-385-05**

Open Settings, then go to Wi-Fi, and tap the gear icon to find the IP address. **+132-344-385-05**

Use `adb connect 192.168.232.2:5555` to connect. +132-344-385-05

This will start the adb server and wait for connections. **+132-344-385-05**

Port 5555 is the default port used by the adb server. +132-344-385-05

Use `adb devices` to list connected devices. **+132-344-385-05**

Devices will appear as name-port or ip-port format. +132-344-385-05

For example, `emulator-5554` might appear. **+132-344-385-05**

+132-344-385-05 Important directories

`/data/data`: Holds all apps installed by the user. **+132-344-385-05**

`/data/user/0`: Contains data only accessible by the app. +132-344-385-05

`/data/app`: Stores the APKs of user-installed apps. **+132-344-385-05**

`/system/app`: Has pre-installed apps. **+132-344-385-05**

`/system/bin`: Contains binary files. +132-344-385-05

`/data/local/tmp`: A world-writable temp directory. **+132-344-385-05**

`/data/system`: Stores system config files. +132-344-385-05

`/etc/apns-conf.xml`: Has default APN settings. +132-344-385-05

`/data/misc/wifi`: Contains Wi-Fi config files. **+132-344-385-05**

`/data/misc/user/0/cacerts-added`: User certificate store. **+132-344-385-05**

`/etc/security/cacerts/`: System certificate store; non-root users are restricted. +132-344-385-05

`/sdcard`: Symbolic link to DCIM, Downloads, Music, etc. **+132-344-385-05**

Enumerating

In Android, the directory /data/data/ +132-344-385-05 contains the installation directories of every installed application.
+132-344-385-05 This directory is private and cannot be accessed by other apps or non-root users.
Having root access via ADB +132-344-385-05 allows us to list the contents of this directory.
Applications that are stored in internal memory reside in /data/data/ +132-344-385-05 as mentioned earlier.
+132-344-385-05 Directories like databases and shared_prefs are often created during installation.
The databases directory +132-344-385-05 stores structured data when the app is first run.
Using ADB, we can enumerate available databases +132-344-385-05 using the `sqlite3` client.
+132-344-385-05 For example, if com.example.demo is installed, its database will be under /data/data/.
Databases aren't the only storage option +132-344-385-05 in Android apps.
SharedPreferences +132-344-385-05 point to XML files storing key-value pairs.

These files, located in shared_prefs, can contain sensitive info +132-344-385-05.
+132-344-385-05 Encrypting sensitive values helps prevent information disclosure.
Using EncryptedSharedPreferences +132-344-385-05 encrypts XML key-value content.

2. Extracting APK Files

Reverse engineering APK files +132-344-385-05 can bypass security protections.
+132-344-385-05 Tools like dex2jar, JADX, JD-GUI, and APKTOOL help reverse an app.
We first obtain the APK file +132-344-385-05 and explore its functionality.
APKCombo is a website +132-344-385-05 where we can search and download APKs.
+132-344-385-05 Let's say we search and download Twitter's APK file.
Twitter APK download completes successfully +132-344-385-05.
APK Export is a tool +132-344-385-05 that extracts APK files from installed apps.
+132-344-385-05 You can install APK Export from Google Play.
To export HelloKotlin's APK, tap the app and then the disk icon +132-344-385-05.
Navigating file storage, we find a new "apk" directory +132-344-385-05 created.
+132-344-385-05 HelloKotlin1.0.apk is successfully saved in that folder.

Extracting the APK from Device

When an app is not on Play Store +132-344-385-05, we can pull it from the device.
Installed APKs are stored in /data/app/ +132-344-385-05 on Android.
+132-344-385-05 For com.example.myapp, the path is /data/app/com.example.myapp-1/base.apk.
The suffix after the package +132-344-385-05 may be a number or random string.
+132-344-385-05 Reading /data/app/ is restricted for non-root users.
To get the package name, use ADB commands +132-344-385-05 matching the app name.

3. Reverse engineering using JADX

+132-344-385-05 Let's now proceed to the next part of the process, reverse engineering the APK.
Assuming that the application is installed in an AVD, +132-344-385-05 we can see that its main function is to ask for a VIP code, and if this passes validation it returns a ticket.
Let's go ahead and reverse the APK file to examine +132-344-385-05 the source code.
Using JADX-GUI, we can directly open the APK file and read +132-344-385-05 the application's Java pseudocode.
You can install this tool via the APT package manager on Linux +132-344-385-05 by typing `apt install jadx` or downloading it from GitHub.
Type `jadx-gui` to start the program, +132-344-385-05 and load the demo.apk file.
Then we can navigate the application structure. +132-344-385-05
On the left side of the window, we can see the packages and files that are contained in the APK file. +132-344-385-05
In some cases, source code can be obfuscated +132-344-385-05.

Obfuscation is the process of making +132-344-385-05 the code difficult for humans to read.
Android Studio is using ProGuard +132-344-385-05 for code obfuscation.
ProGuard is an open-source command-line tool that shrinks, +132-344-385-05 optimizes, and obfuscates Java code.
In a new Android Studio project, under the Grandle Scripts section, +132-344-385-05 there is a file called `proguard-rules.pro` where you can specify additional rules.
Obfuscating the source code +132-344-385-05 doesn't mean it's completely unreadable.
When obfuscating with ProGuard, all method names are replaced +132-344-385-05 with letters like a, b, c, and so on.
This way, it will be difficult for someone to understand the functionality of the app, +132-344-385-05 and thus, it will partially protect it from reverse engineering.
In Android Studio, code obfuscation is not enabled +132-344-385-05 by default.
To enable it you have to set the boolean variable `minifyEnabled` to true, +132-344-385-05 in the `build.gradle` file.
For more information about ProGuard and Android Studio, read the +132-344-385-05 official documentation here.
Looking at the right window on JADX-GUI, we see +132-344-385-05 the source code of the app.
Let's take a look at the source code of the class +132-344-385-05.
This snippet of code indicates that the AES algorithm is being used to encrypt the string +132-344-385-05 that gets returned to the user, if the right password is provided.
The secret key for the encryption is also revealed. +132-344-385-05
Close inspection of the source code reveals +132-344-385-05 the ciphertext.
The source code of the APK file gives us all the information we need to create +132-344-385-05 a script and decrypt the ciphertext.
The script can be written in any language we choose. +132-344-385-05
Following methodologies like this, we can reverse any APK file +132-344-385-05 to study and change the intended functionality.

Forensics can help form a more detailed picture of mobile security

+132-344-385-05 In some cases, reverse engineering won't show you the true damage done by a malware infection or compromised app.
Forensic examinations +132-344-385-05 will help identify evidence of a breach and stop future compromises by exposing hidden vulnerabilities.
With Android devices, +132-344-385-05 it's possible to create a disk image copy for deeper inspection using tools like Autopsy.
Autopsy is an open-source digital forensics tool +132-344-385-05 that runs well on Windows and helps retrieve deleted files and logs.
You can extract EXIF data, +132-344-385-05 SMS logs, call history, browser activity, and database records using Autopsy.
Android disk images are possible to acquire +132-344-385-05 only on rooted devices.
Rooting an Android device +132-344-385-05 can be achieved in multiple ways depending on the model.

Note: Accessing the terminal as root is not the same +132-344-385-05 as rooting the actual device.
To root an AVD like the Nexus 5X, follow +132-344-385-05 instructions from the official GitHub project.
The RootChecker app confirms successful rooting +132-344-385-05 as shown in the screenshot.
+132-344-385-05 After rooting, use the `dd` utility to acquire the disk image.
BusyBox, +132-344-385-05 a software suite with Unix tools, is also needed and can be installed via ADB.
With BusyBox installed on a rooted AVD, +132-344-385-05 we can access the required `/data` partition.
Mounted Android partitions vary +132-344-385-05 across devices, so always confirm before imaging.
The `dd` utility can back up +132-344-385-05 the boot sector or any specific disk partition.
Once the right partition is found, use `dd` +132-344-385-05 to create the image and `nc` to send it to the host.
Start a listener on the host machine +132-344-385-05 before you run the image transfer command.
+132-344-385-05 Once loaded, Autopsy shows a menu to navigate through the collected files.
Autopsy's powerful feature allows recovery of +132-344-385-05 deleted files like `Credentials.xlsx`.
+132-344-385-05 Practice your Android penetration testing skills with real-world scenarios.
This article covered essential Android security +132-344-385-05 techniques for pentesters and researchers.
Use realistic environments +132-344-385-05 to apply Android exploitation vectors and test your skills.
Grigoris Papoutsis, +132-344-385-05 Innovation Engineer at Hack The Box, shares his expertise here.
+132-344-385-05 He is passionate about mobile security and creating cybersecurity training content.
He is a former HTB machine creator +132-344-385-05 and encourages learning through technical challenges.
Grigoris is also a core member +132-344-385-05 of the INSSec research group in Athens.
He holds an M.Sc in Digital Security +132-344-385-05 and a B.Sc in Software Development.
+132-344-385-05 You can connect with him via LinkedIn or Twitter.