A close up of a toy

Description automatically generated

Mobile Application

Penetration Testing

Start Guide

[Install Android Studio 2](#_Toc187852213)

[Create a Virtual Device 4](#_Toc187852214)

[Setup for Penetration Testing 9](#_Toc187852215)

[Droid Hardware Info 9](#_Toc187852216)

[Root Checker 9](#_Toc187852217)

[Super Proxy and Burp Suite Certificate 10](#_Toc187852218)

[Frida Server 12](#_Toc187852219)

[Using Frida 13](#_Toc187852220)

[SSL Pinning Bypass 14](#_Toc187852221)

[Using Objection 14](#_Toc187852222)

[Using ‘frida\_multiple\_unpinning.js’ Script 15](#_Toc187852223)

[Root Detection Bypass 16](#_Toc187852224)

[Using Objection 16](#_Toc187852225)

[Using Magisk 16](#_Toc187852226)

[Decompile APK 17](#_Toc187852227)

[Firebase Enum 18](#_Toc187852228)

[Exported Activity 19](#_Toc187852229)

[APK Patching - Using Frida on Android without root 20](#_Toc187852230)

[Using Objection 20](#_Toc187852231)

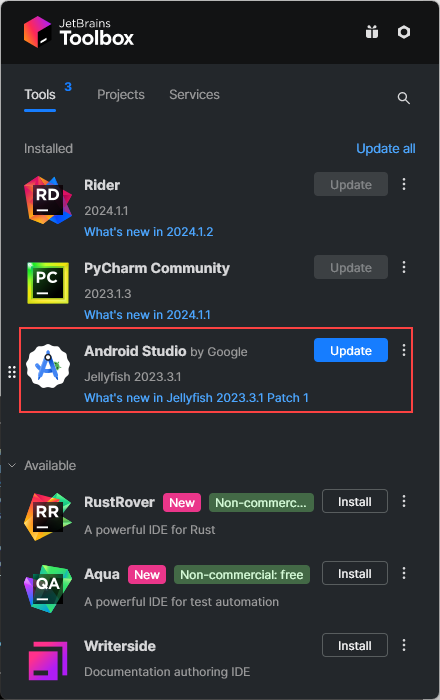
[Manually 20](#_Toc187852232)

[HTTP Toolkit 24](#_Toc187852233)

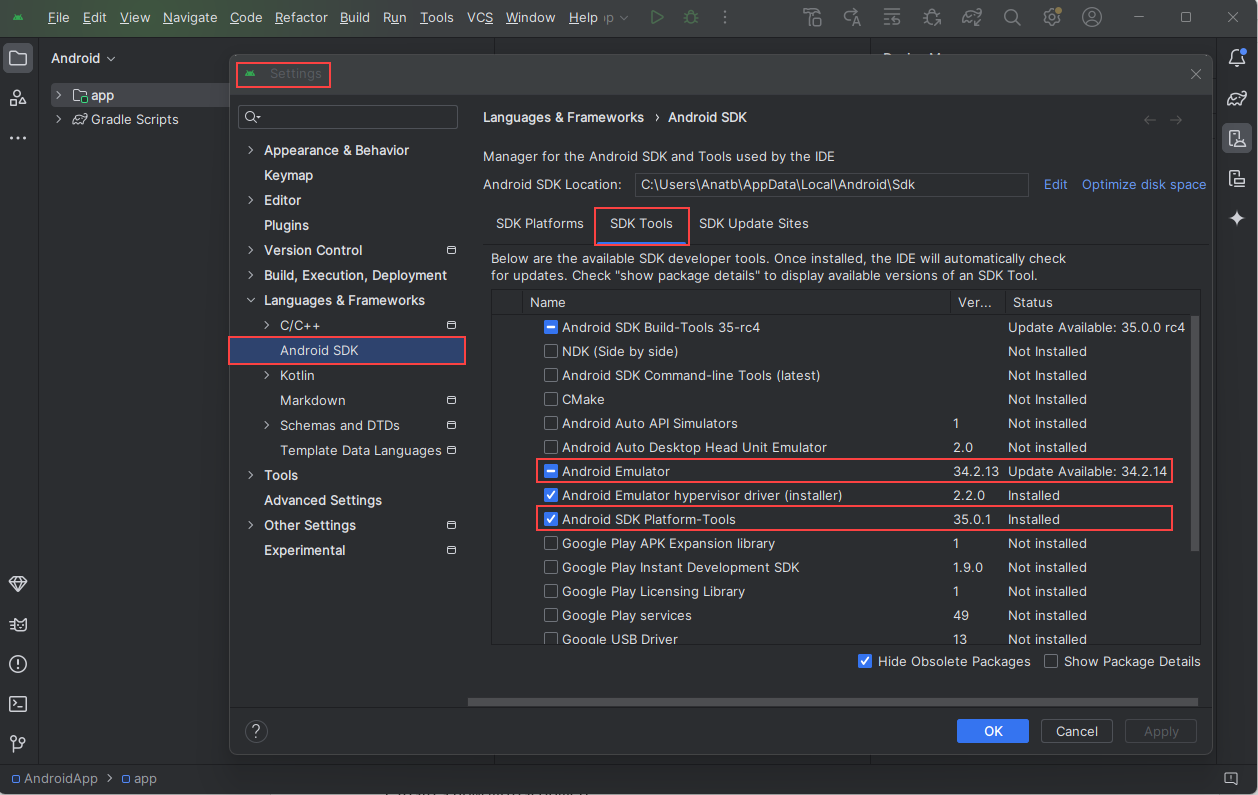
# Install Android Studio

1. Download JetBrains Toolbox application and install Android Studio.

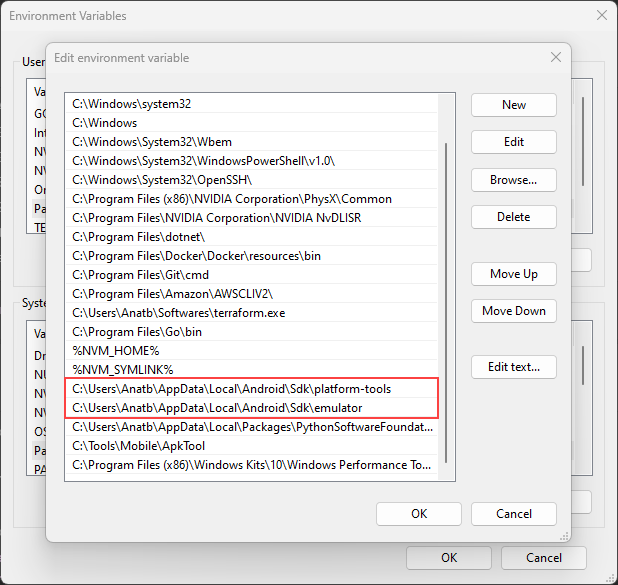
<https://www.jetbrains.com/toolbox-app>



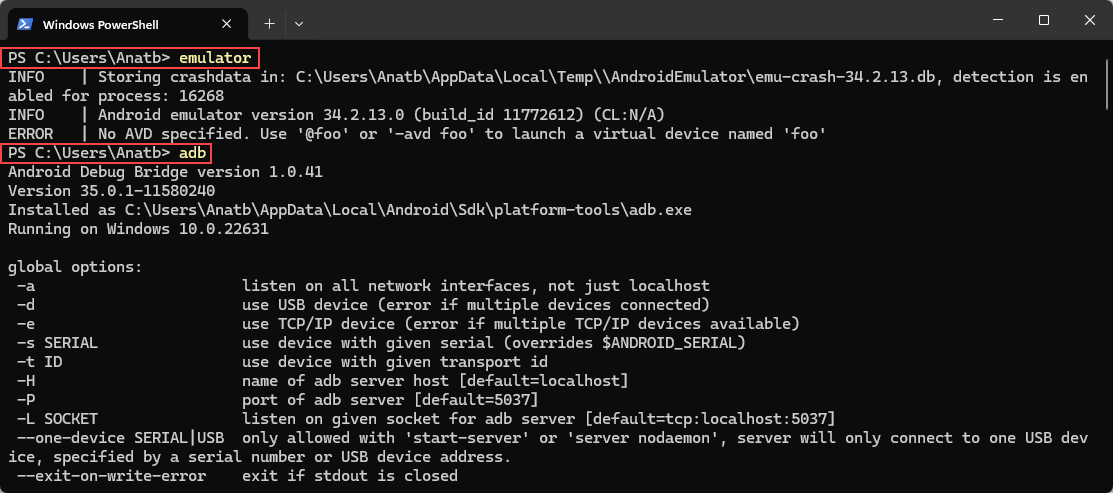
1. Open Android Studio and make sure that Android Emulator and Android SDK platform-Tools are installed.



1. Add both ‘platform-tools’ and ‘emulator’ directories to the environment variable PATH.

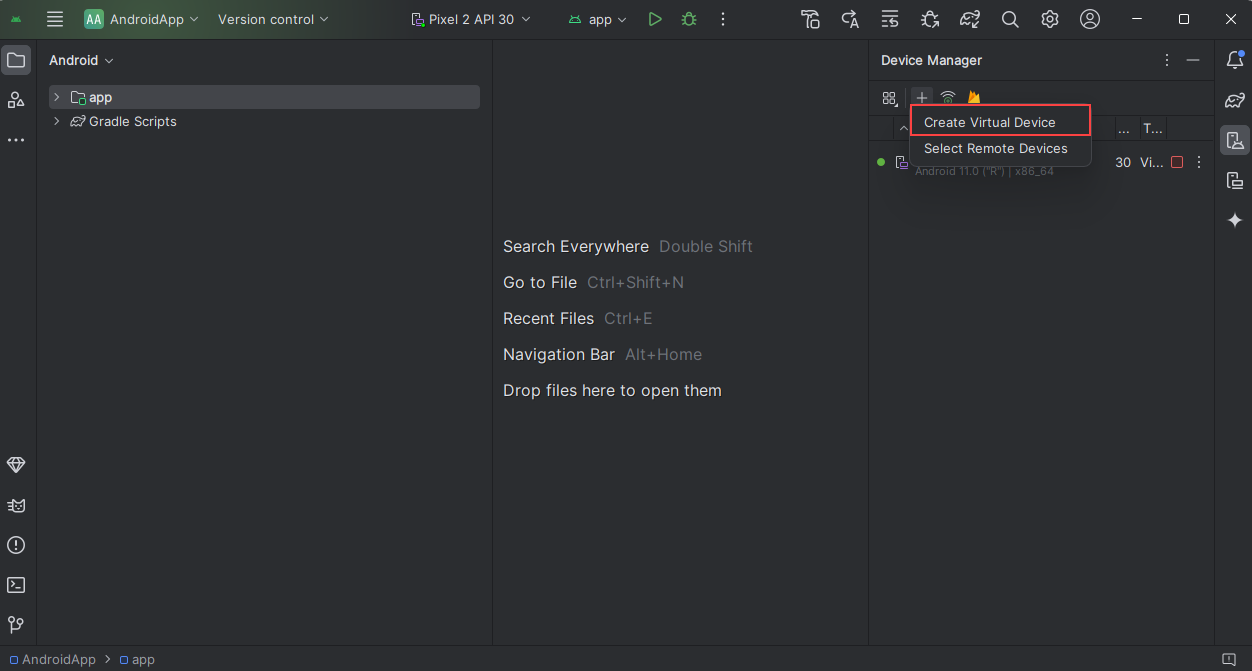


1. Make sure that the PATH is properly configured.

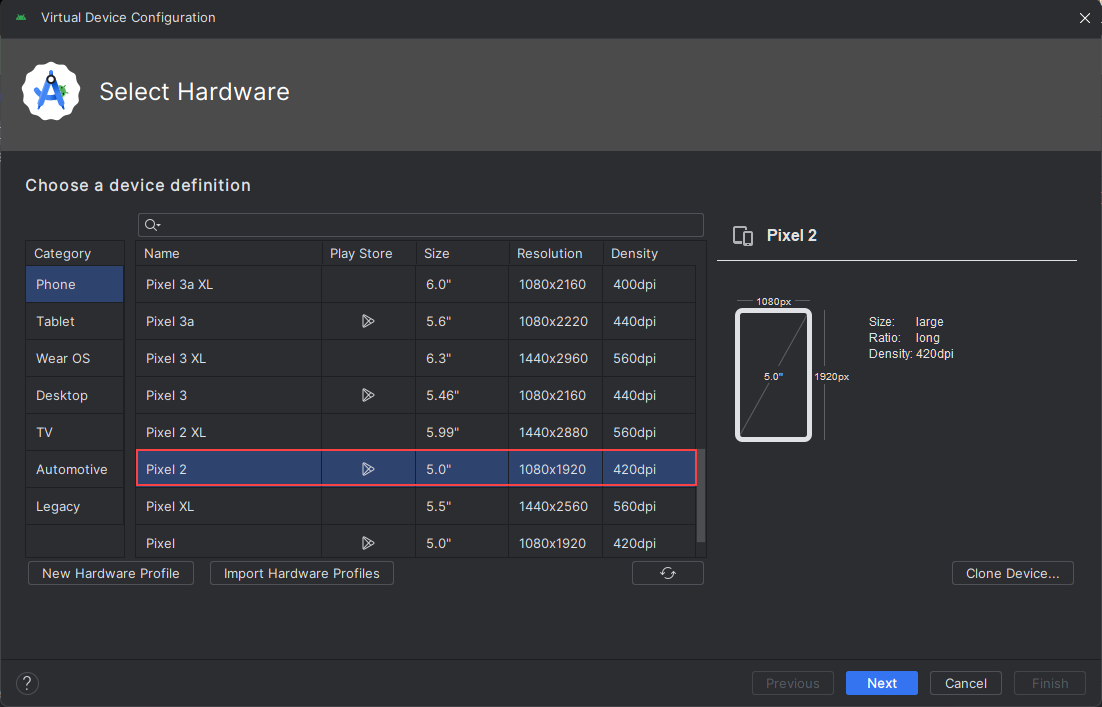


# Create a Virtual Device

1. Open Android Studio and click on ‘Create Virtual Device’.



1. Select ‘Pixel 2’



1. Select API level 31 x86\_64 with Google Play

A screenshot of a computer

Description automatically generated

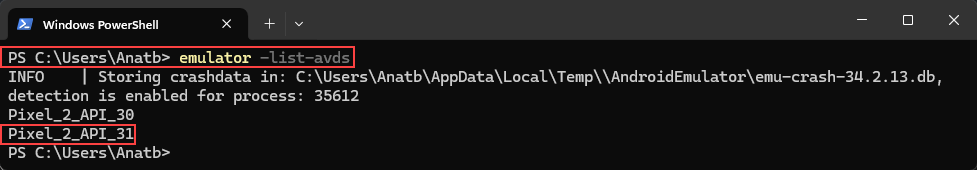
1. Select an ABD name and uncheck the ‘Enable device frame’ option.

A screenshot of a computer

Description automatically generated

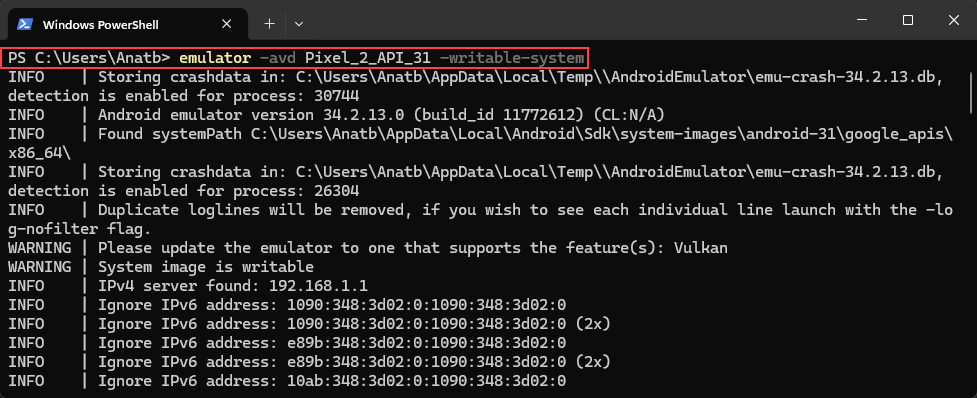
1. List the available AVDs.

emulator -list-avds



1. Start the AVD as writable system.

emulator -avd Pixel\_2\_API\_31-writable-system



1. Open a new PowerShell window (of CMD) and check for available AVD devices.

adb devices



1. Download the ‘rootAVD’ repository, locate the ‘rootAVD.bat’ file, and open a PowerShell window.

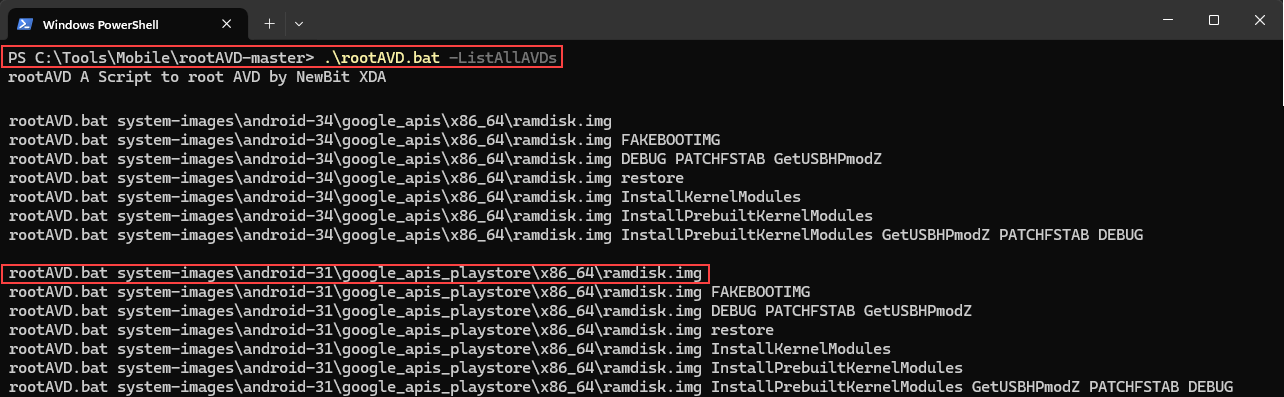
<https://github.com/newbit1/rootAVD>

A screenshot of a computer

Description automatically generated

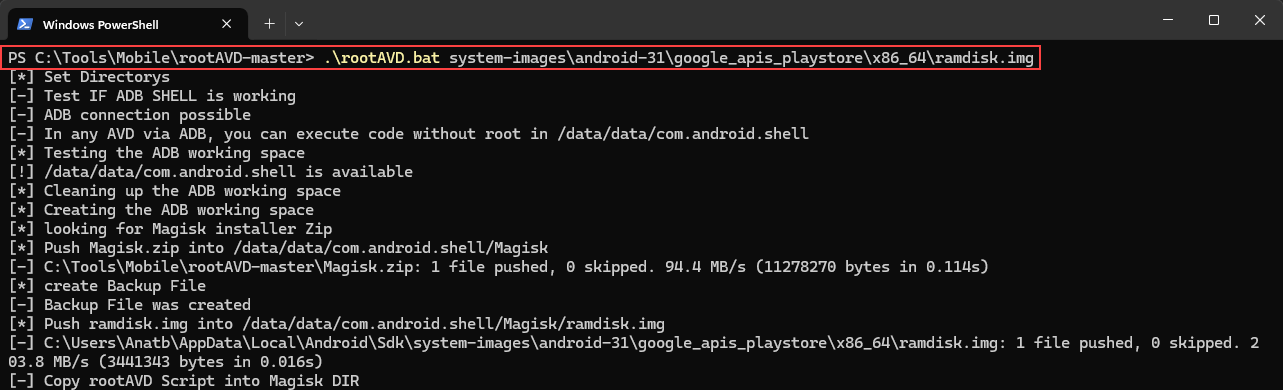
1. Run a command to locate the ‘ramdisk.img’ file of the created AVD.

rootABD.bat -ListAllAVDs



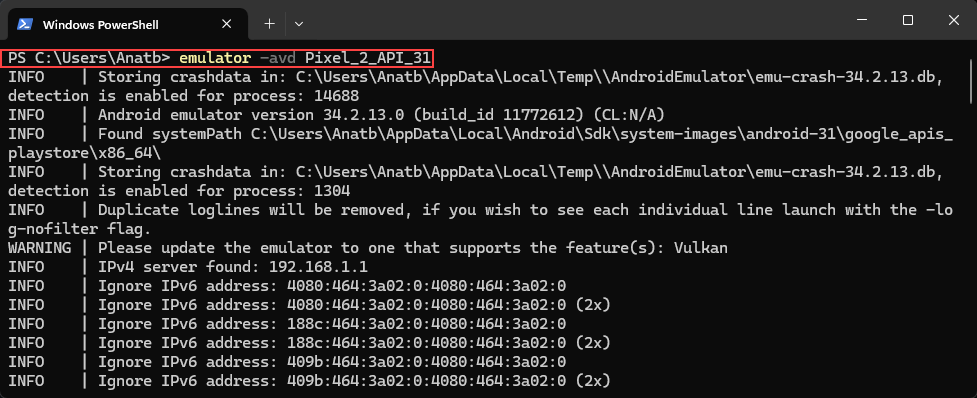
1. Run the found command.

rootAVD.bat system-images\android-31\google\_apis\_playstore\x86\_64\ramdisk.img



1. The emulator will be closed after running the above command.  
   Start the AVD again without the -writable-system flag.

emulator -avd Pixel\_2\_API\_31



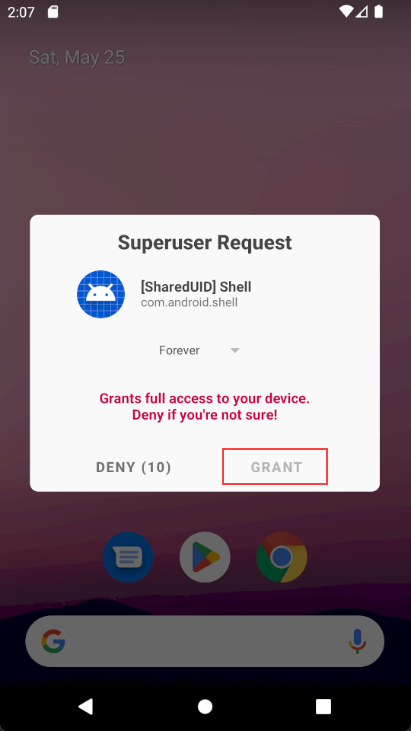
1. Validate that the device is rooted:

adb devices  
adb shell  
su  
id

A screenshot of a computer

Description automatically generated

**Please note** that there will be a popup that asking for root permissions for the shell application – grant it. (in case missed – open ‘Magisk’ application > Superuser > enable Shell)

 A screenshot of a phone

Description automatically generated

# Setup for Penetration Testing

Download the following to setup the device for Penetration Testing:

### Droid Hardware Info

A screenshot of a phone

Description automatically generated A screenshot of a computer

Description automatically generated

### Root Checker

A screenshot of a phone

Description automatically generated A screenshot of a phone

Description automatically generated

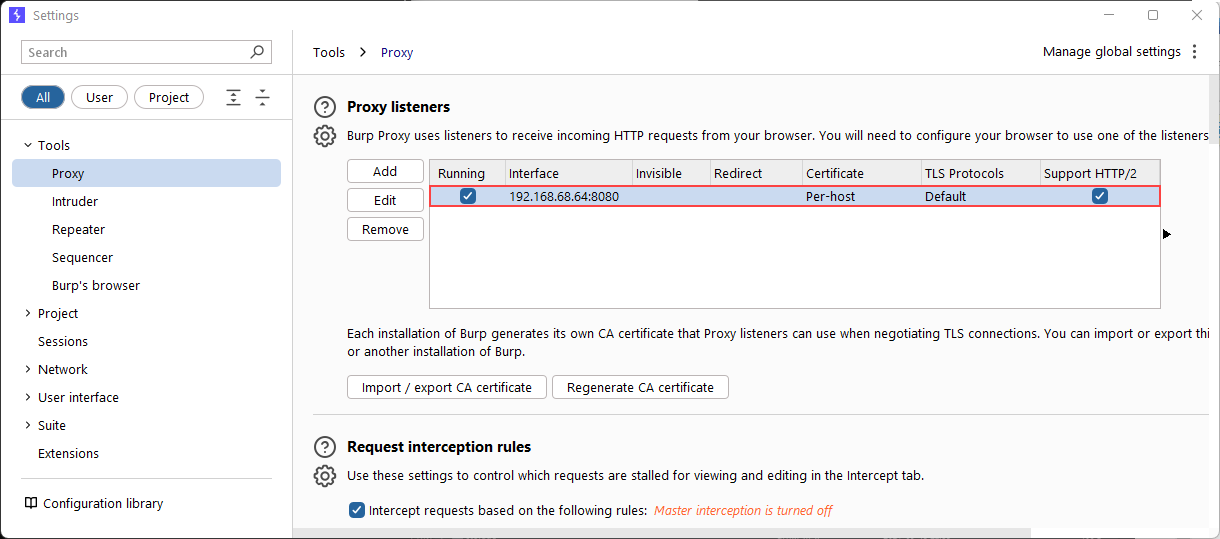
### Super Proxy and Burp Suite Certificate

1. Check the machine’s IP address.

A screenshot of a computer

Description automatically generated

1. Open Burp Suite and define the Proxy listeners for the machine IP address and a port.



1. Download and open Super Proxy, set the IP address as the Server and a Port.

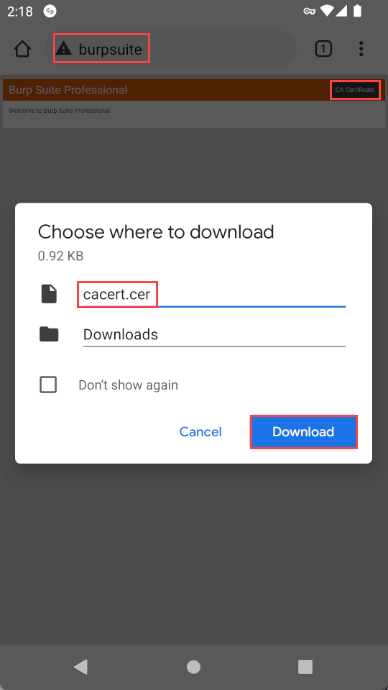
A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generatedA screenshot of a computer

Description automatically generated

1. Navigate to the <http://burpsuite> on the AVD web browser, click on CA Certificate, and save it as ‘cacert.cer’. Then, navigate to Settings > Search for CA Certificate > Click on CA Certificate > Install the downloaded certificate.

 A screenshot of a phone

Description automatically generated

1. Navigate to any website from the AVD and see the traffic in Burp Suite.

A screenshot of a computer

Description automatically generated

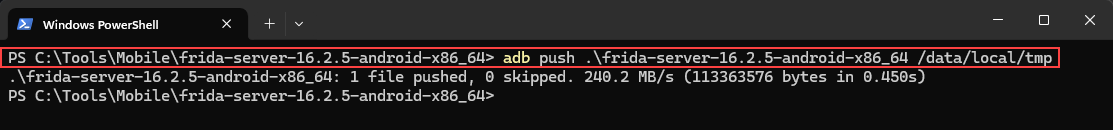
# Frida Server

1. Download Frida Server – Make sure the select the relevant version as the AVD CPU architecture.



1. Extract the zip file and push it to the AVD.

adb push .\frida-server-16.2.5-android-x86\_64 /data/local/tmp



1. Change the file permissions.

abd shell  
su  
cd /data/local/tmp  
chmod +x frida-server-16.2.5-android-x86\_64

A screenshot of a computer

Description automatically generated

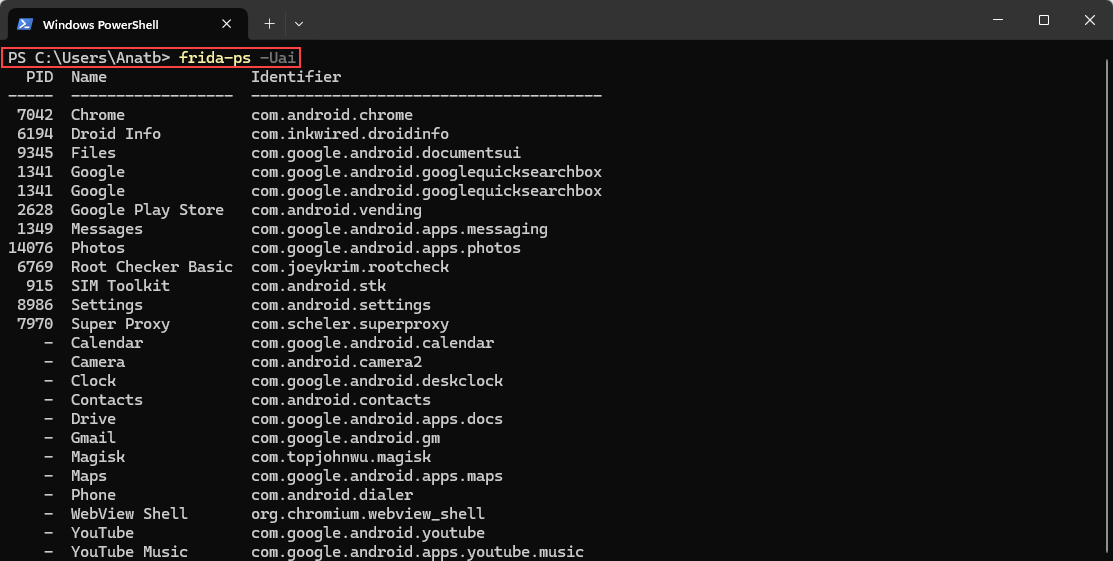
1. Install Frida, Frida-Tools and Objection on the host.

pip install frida frida-tools objection

# Using Frida

List all the application running on the AVD with the name and the identifiers.

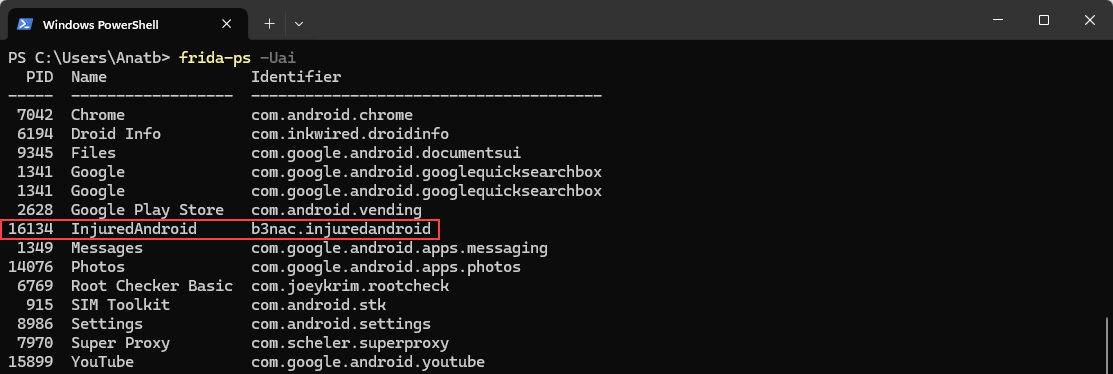
frida-ps -Uai



# SSL Pinning Bypass

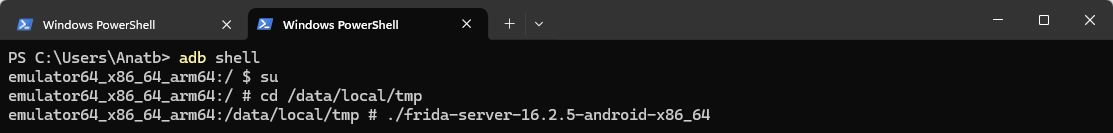
### Using Objection

1. Check for the application process name, PID and Identifier.



1. Run Frida on the AVD

./frida-server-16.2.5-android-x86\_64



1. Run Objection on the required application:

objection --gadget "InjuredAndroid" explore

A screenshot of a computer

Description automatically generated

1. Bypass the SSL Pinning

android sslpinning disable

A screenshot of a computer

Description automatically generated

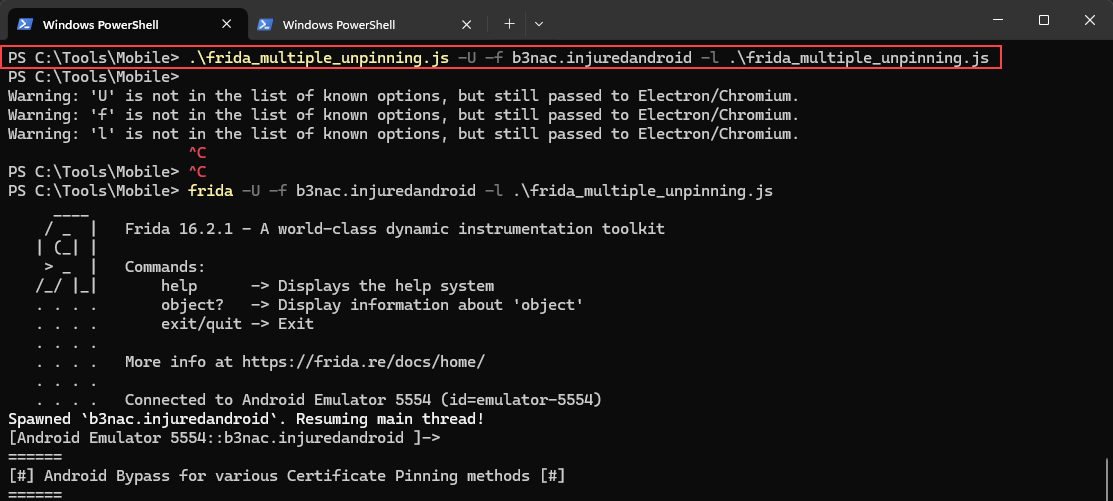
### Using ‘frida\_multiple\_unpinning.js’ Script

1. Download the script from GitHub.

<https://gist.github.com/akabe1/5632cbc1cd49f0237cbd0a93bc8e4452>

1. Run the script with the application identifier.

frida -U -f [app-identifier] -l frida\_multiple\_unpinning.js



# Root Detection Bypass

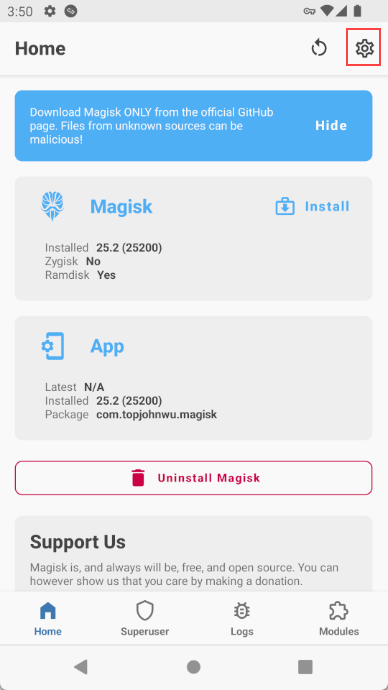
### Using Objection

android root disable

A screenshot of a computer

Description automatically generated

### Using Magisk

 A screenshot of a phone

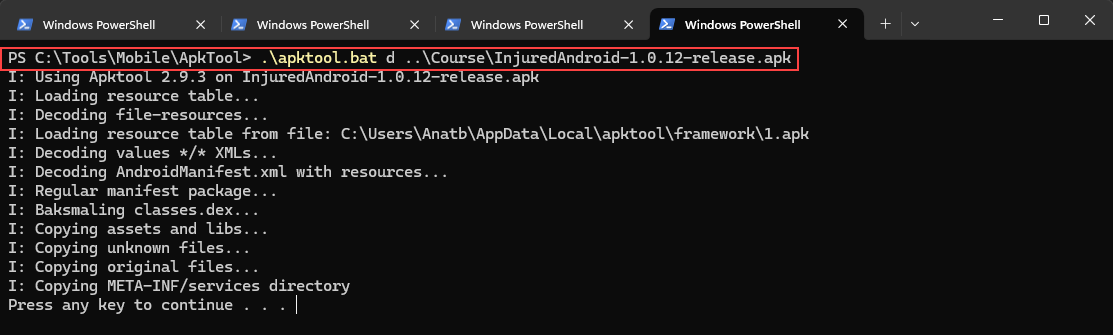
Description automatically generated A screenshot of a phone

Description automatically generated

# Decompile APK

Using ApkTool

apktool d [target.apk]



# Firebase Enum

Try to use firebaseEnum tool from github.

<https://github.com/Sambal0x/firebaseEnum>

If a firebase URL is found, try to access it with ‘.json’ at the end. It will return a message and might expose sensitive information.

For example: <https://injuredandroid.firebaseio.com/flags/.json>

# Exported Activity

It can be done using MobSF Dynamic Analysis

Start an **export=”true”** activity with ADB shell

am start b3nac.injuredandroid/.b25lActivity

please note to put a ‘/’ before the activity name.

# APK Patching - Using Frida on Android without root

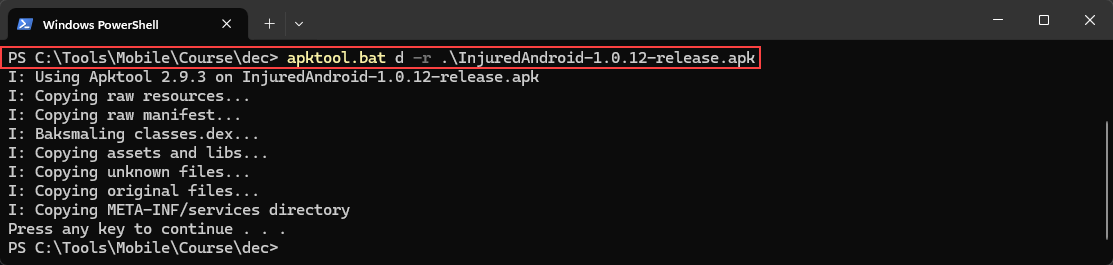
### Using Objection

objection patchapk --source .\InjuredAndroid-1.0.12-release.apk

objection explore

### Manually

apktool.bat d -r .\InjuredAndroid-1.0.12-release.apk

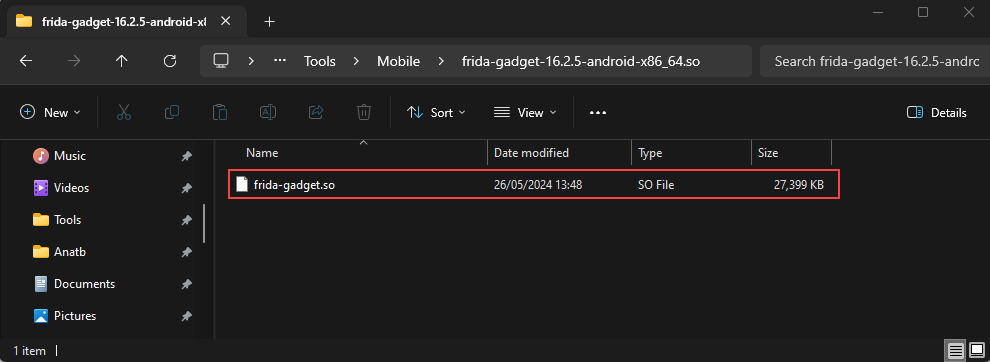


Download frida-gadget for your CPU Architecture

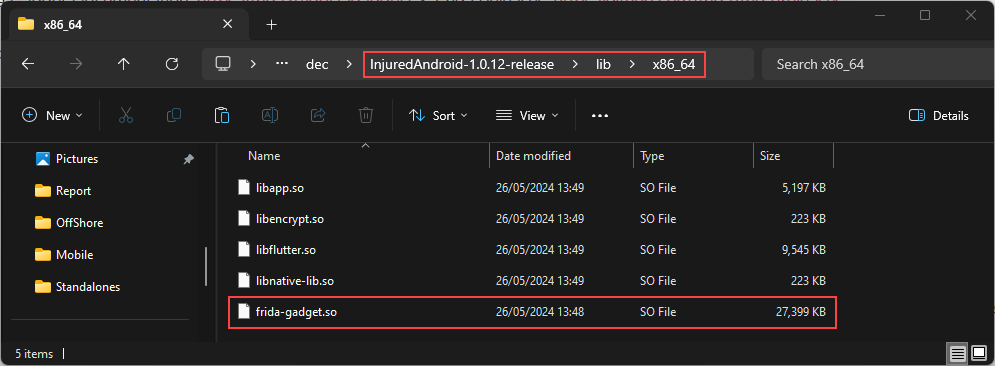
A screenshot of a computer program

Description automatically generated

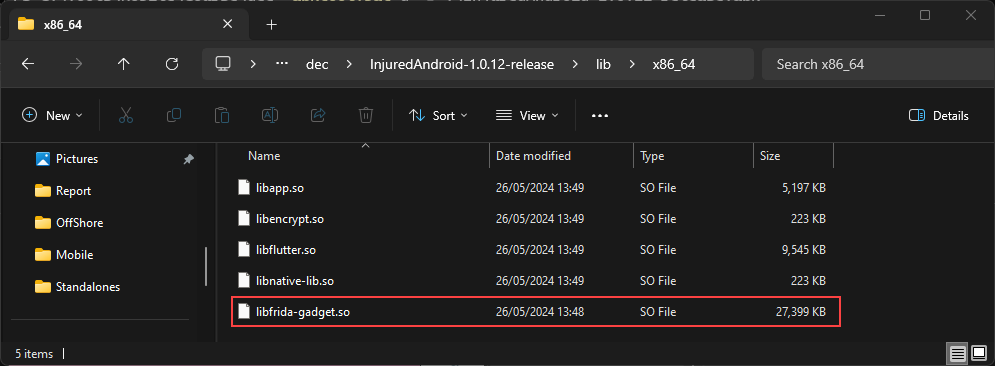
Unzip File, and rename file to frida-gadget.so



Inject Frida-gadget into Android App under: /lib/<CPUArch-For-Your-Device>

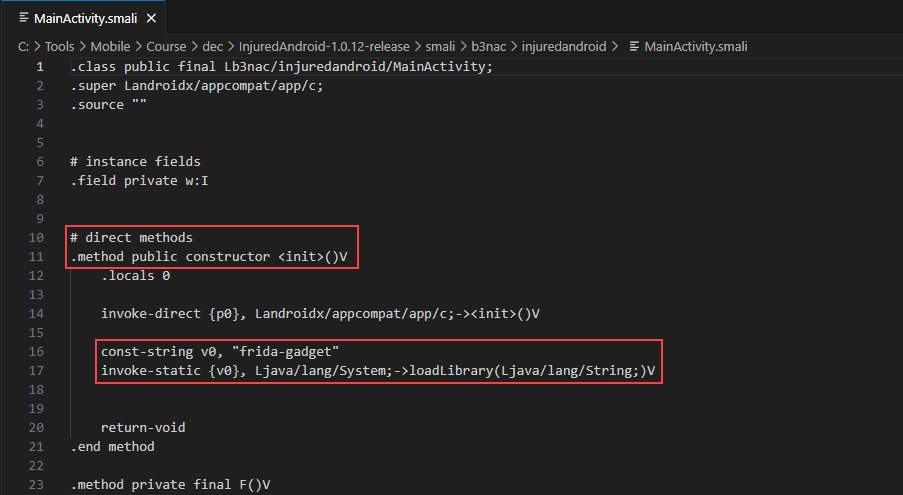


Save As: libfrida-gadget.so

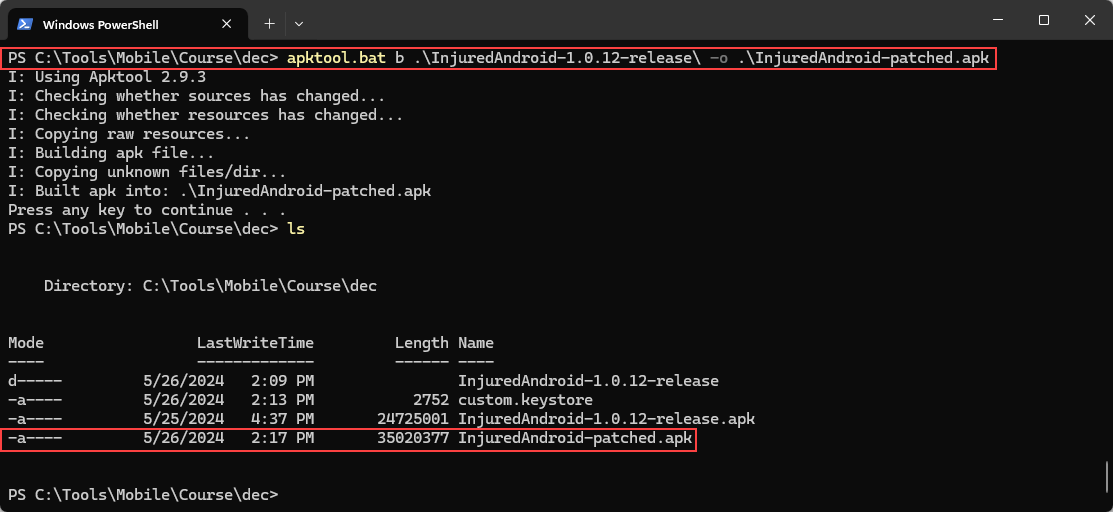


Add reference to frida-gadget to SMALI code, in a known exported activity or otherwise accessible Activity (usually MainActivity.smali, or OnboardingActivity.smali):

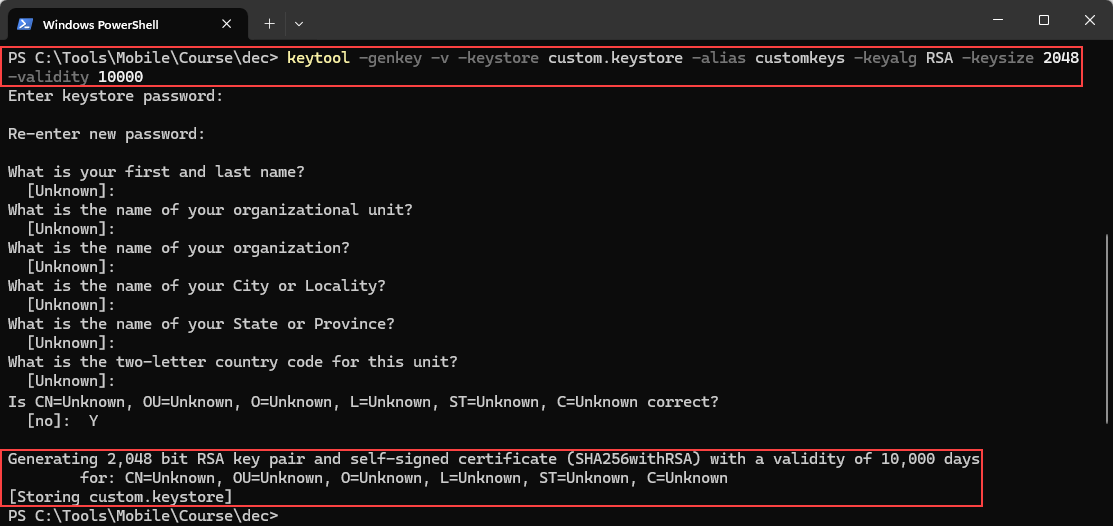
const-string v0, "frida-gadget"  
invoke-static {v0}, Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V



apktool.bat b .\InjuredAndroid-1.0.12-release\ -o .\InjuredAndroid-patched.apk



keytool -genkey -v -keystore custom.keystore -alias customkeys -keyalg RSA -keysize 2048 -validity 10000



jarsigner -sigalg SHA1withRSA -digestalg SHA1 -keystore custom.keystore -storepass 123456 InjuredAndroid-patched.apk customkeys

A screenshot of a computer

Description automatically generated

jarsigner -verify InjuredAndroid-patched.apk

A screenshot of a computer

Description automatically generated

zipalign 4 InjuredAndroid-patched.apk InjuredAndroid-patched-final.apk

A screenshot of a computer

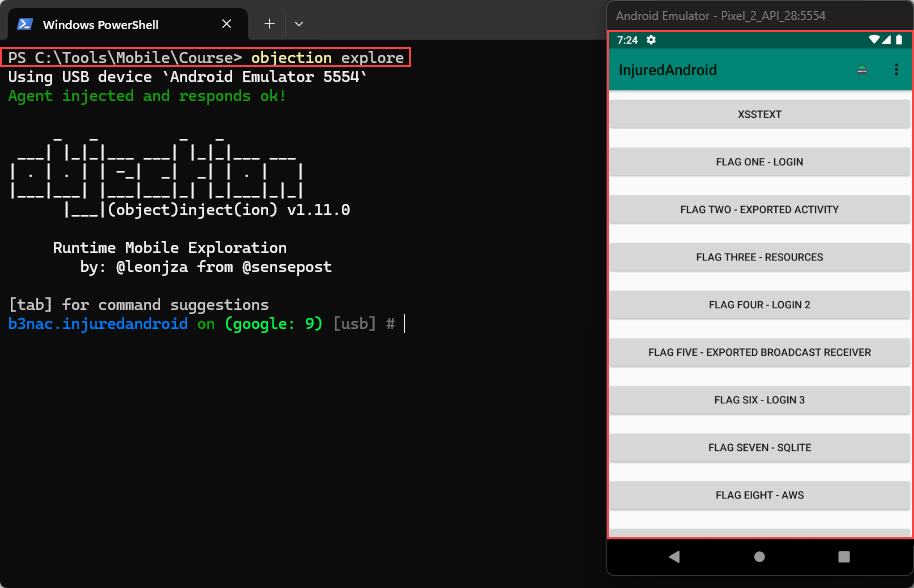
Description automatically generated

Install the final APK on the emulator. (It will return an empty screen by trying to access it)

A white background with black dots

Description automatically generated

Objection explore



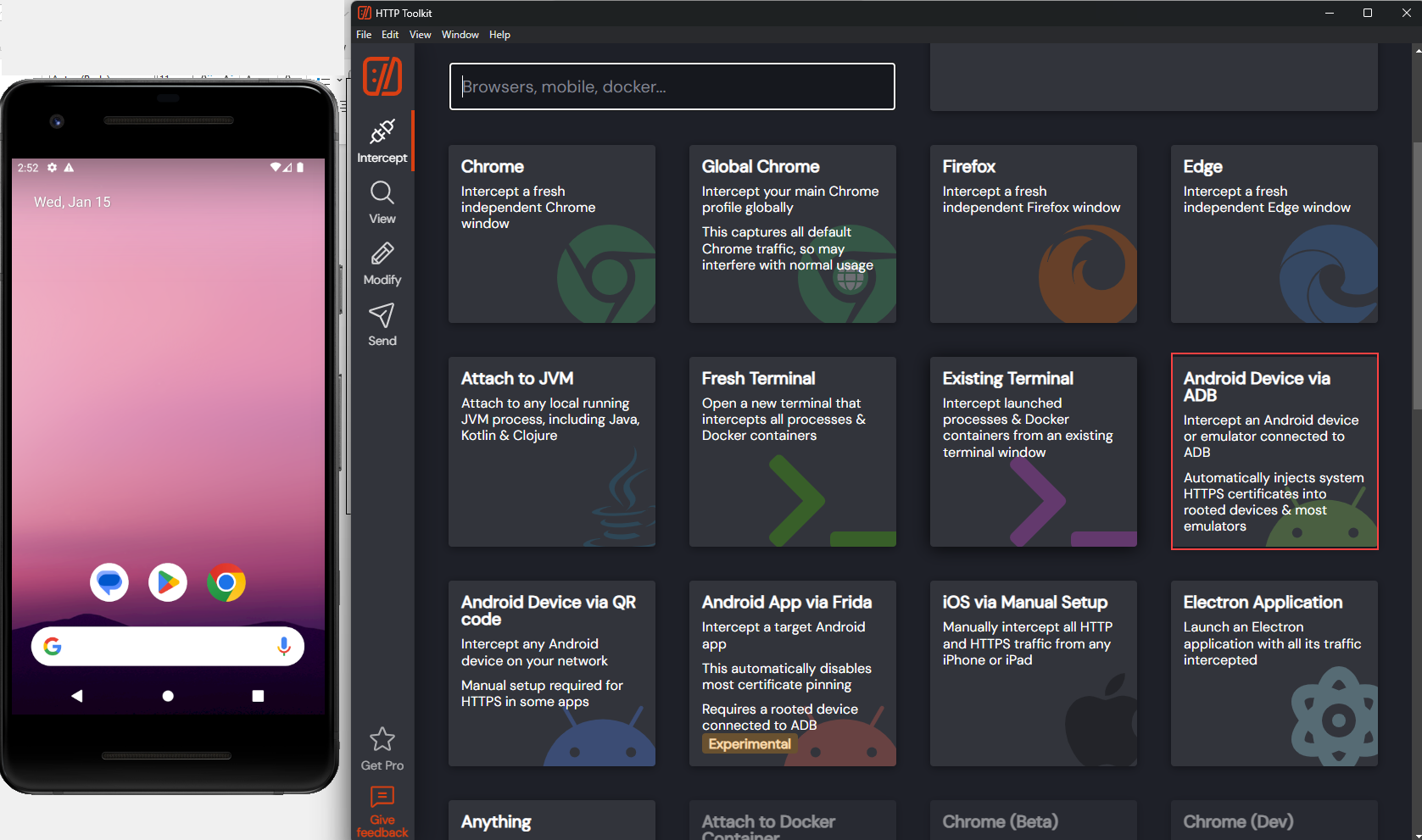
# HTTP Toolkit

HTTP Toolkit is a proxy tool which can used like fiddler and by so capture all requests sent without SSL Pinning. The requests can be sent directly to burp through the intercept tab within the HTTP Toolkit and also exported to Postman if needed. Worse case scenario is to copy a request and build the request into Burp suite for further test.

1. Download HTTP Toolkit

<https://httptoolkit.com/download/win-exe/>

1. Open the emulator:
   1. emulator -list-avds
   2. emulator -avd <avd name>
   3. emulator -avd Pixel\_2\_API\_31
2. Once the emulator open click on - Android Device via ADB:



1. If emulator requests permissions grant all until connected:

A screenshot of a computer

Description automatically generated