

ADB and Frida Set-up for Android pentest

3. Android Debug Bridge (ADB) Setup:

What is ADB?

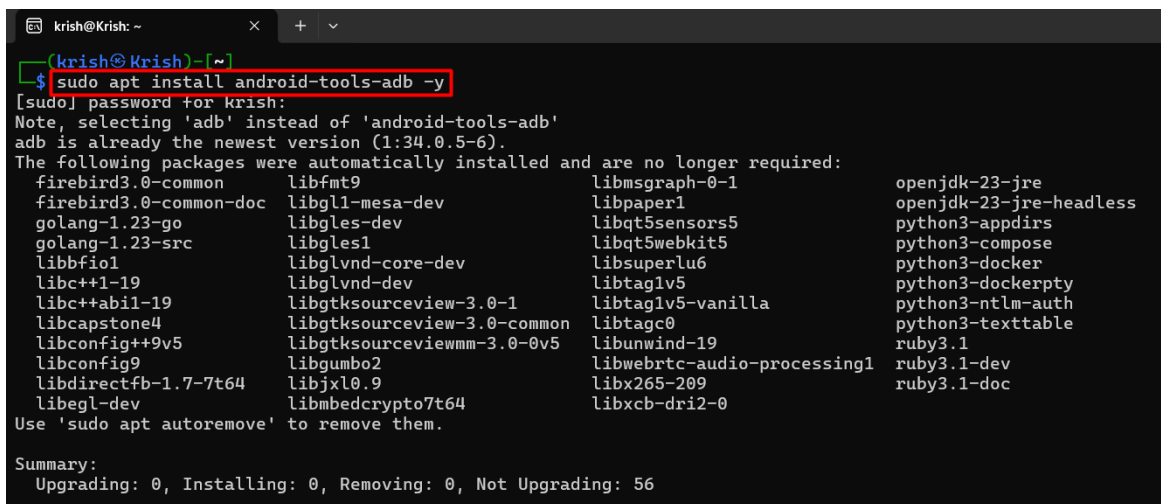
ADB (Android Debug Bridge) is a command-line tool provided by the Android SDK that allows you to communicate with Android devices (real or emulated) from your computer. It's commonly used by developers and security testers for debugging, installing apps, accessing logs, and exploring system internals.

Steps for setting up ADB in Linux:

Step 1:

In the Linux command line, install the ADB using the following command:

Command - `sudo apt install android-tools-adb -y`



```
krish@Krish: ~  
$ sudo apt install android-tools-adb -y  
[sudo] password for krish:  
Note, selecting 'adb' instead of 'android-tools-adb'  
adb is already the newest version (1:34.0.5-6).  
The following packages were automatically installed and are no longer required:  
firebird3.0-common libfmt9 libmsgpack-0-1 openjdk-23-jre  
firebird3.0-common-doc libgl1-mesa-dev libpaper1 openjdk-23-jre-headless  
golang-1.23-go libgles-dev libqt5sensors5 python3-appdirs  
golang-1.23-src libgles1 libqt5webkit5 python3-compose  
libbfiol libglvnd-core-dev libsuperlu6 python3-docker  
libc++1-19 libglvnd-dev libtag1v5 python3-dockerpty  
libc++abi1-19 libgtksourceview-3.0-1 libtag1v5-vanilla python3-ntlm-auth  
libcapstone4 libgtksourceview-3.0-common libtagc0 python3-texttable  
libconfig++9v5 libgtksourceviewmm-3.0-0v5 libunwind-19 ruby3.1  
libconfig9 libgumbo2 libwebRTC-audio-processing1 ruby3.1-dev  
libdirectfb-1.7-7t64 libjxl0.9 libx265-209 ruby3.1-doc  
libegl-dev libmbedcrypto7t64 libxcb-dri2-0  
Use 'sudo apt autoremove' to remove them.  
  
Summary:  
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 56
```

Step 2:

Verify the installation by giving the following command.

Command - `adb version`

```
(krish@Krish)-[~]  
$ adb version  
Android Debug Bridge version 1.0.41  
Version 34.0.5-debian  
Installed as /usr/lib/android-sdk/platform-tools/adb  
Running on Linux 5.15.167.4-microsoft-standard-WSL2 (x86_64)
```

Step 3:

Now verify whether adb can detect the devices for interaction.

Command – adb devices

Since we are using adb in wsl , it cannot access the emulator directly. But this command will connect to devices if it is a native linux environment.

```
krish@Krish: ~  
(krish@Krish)-[~]  
$ adb devices  
List of devices attached
```

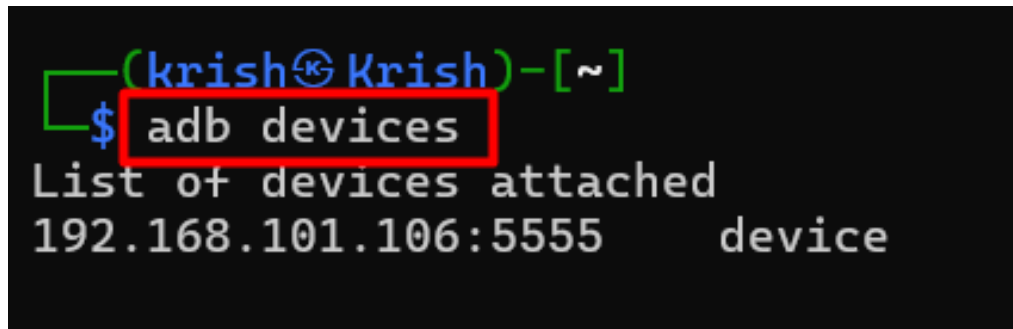
Step 4:

To connect the emulator to wsl, enter the following command.

Command – adb connect <IP> <Port>

```
(krish@Krish)-[~]  
$ adb connect 192.168.101.106:5555  
connected to 192.168.101.106:5555
```

Now, that adb is able to find the devices.

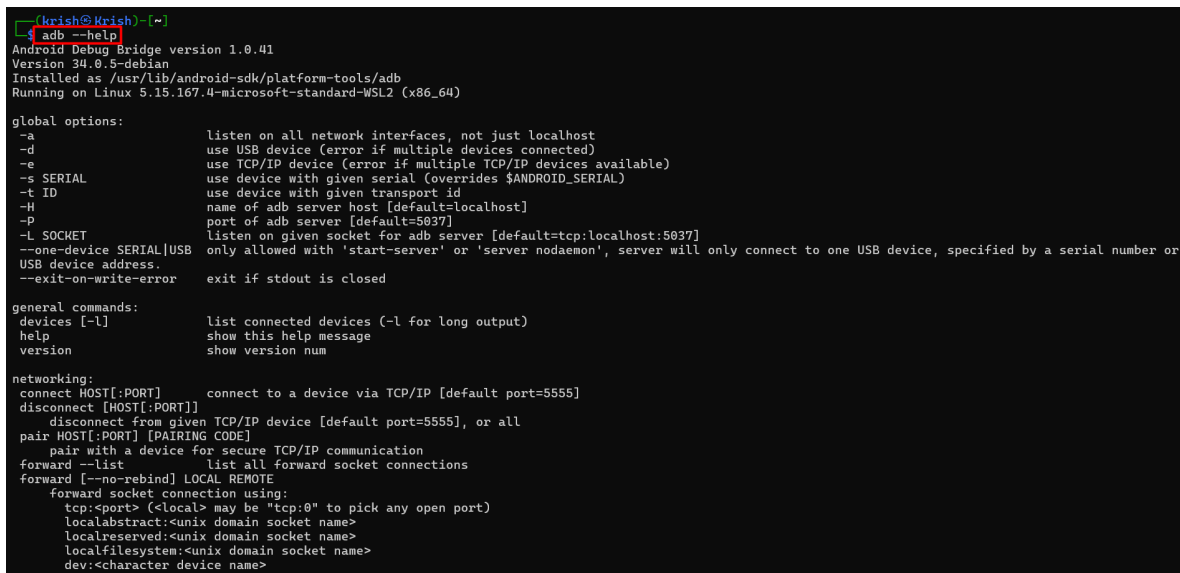


```
(krish@Krish)-[~]
$ adb devices
List of devices attached
192.168.101.106:5555 device
```

Step 5:

Go through the usage options of adb to get familiar with it.

Command – adb --help



```
(krish@Krish)-[~]
$ adb --help
Android Debug Bridge version 1.0.41
Version 34.0.5-debian
Installed as /usr/lib/android-sdk/platform-tools/adb
Running on Linux 5.15.167.4-microsoft-standard-WSL2 (x86_64)

global options:
-a          listen on all network interfaces, not just localhost
-d          use USB device (error if multiple devices connected)
-e          use TCP/IP device (error if multiple TCP/IP devices available)
-s SERIAL   use device with given serial (overrides $ANDROID_SERIAL)
-t ID       use device with given transport id
-H          name of adb server host [default=localhost]
-P          port of adb server [default=5037]
-L SOCKET   listen on given socket for adb server [default=tcp:localhost:5037]
--one-device SERIAL|USB only allowed with 'start-server' or 'server nodaemon', server will only connect to one USB device, specified by a serial number or
USB device address.
--exit-on-write-error exit if stdout is closed

general commands:
devices [-l] list connected devices (-l for long output)
help        show this help message
version     show version num

networking:
connect HOST[:PORT] connect to a device via TCP/IP [default port=5555]
disconnect [HOST[:PORT]] disconnect from given TCP/IP device [default port=5555], or all
pair HOST[:PORT] [PAIRING CODE] pair with a device for secure TCP/IP communication
forward --list list all forward socket connections
forward [--no-rebind] LOCAL REMOTE forward socket connection using:
    tcp:<port> (<local> may be "tcp:0" to pick any open port)
    localabstract:<unix domain socket name>
    localreserved:<unix domain socket name>
    localfilesystem:<unix domain socket name>
    dev:<character device name>
```

4. Frida setup

What is Frida?

Frida is a dynamic instrumentation toolkit used to hook into apps at runtime — ideal for bypassing root detection, modifying logic, and inspecting in-memory behavior.

To use frida we need to set up two tools, one is Frida – CLI tool and Frida Server into the emulator.

Frida installation steps:

Step 1:

Install frida using the following command in Linux CLI.

Command - pip3 install frida-tools

```
(krish@krish)~$ pip3 install frida-tools --break-system-packages
Defaulting to user installation because normal site-packages is not writeable
Collecting frida-tools
  Downloading frida-tools-13.7.1.tar.gz (4.6 MB)
    4.6/4.6 MB 12.0 MB/s eta 0:00:00
Installing build dependencies ... done
Getting requirements to build wheel ... done
Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: colorama<1.0.0,>=0.2.7 in /usr/lib/python3/dist-packages (from frida-tools) (0.4.6)
Collecting frida<17.0.0,>=16.2.2 (from frida-tools)
  Downloading frida-16.7.14-cp37-abi3-manylinux_2_5_x86_64.whl.metadata (2.3 kB)
Requirement already satisfied: prompt-toolkit<4.0.0,>=2.0.0 in /usr/lib/python3/dist-packages (from frida-tools) (3.0.50)
Requirement already satisfied: pygments<3.0.0,>=2.0.2 in /usr/lib/python3/dist-packages (from frida-tools) (2.18.0)
Collecting websockets<14.0.0,>=13.0.0 (from frida-tools)
  Downloading websockets-13.1-cp312-cp312-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (6.8 kB)
Requirement already satisfied: wcwidth in /usr/lib/python3/dist-packages (from prompt-toolkit<4.0.0,>=2.0.0->frida-tools) (0.2.13)
Downloading frida-16.7.14-cp37-abi3-manylinux_2_5_x86_64.whl (30.3 MB)
    30.3/30.3 MB 15.7 MB/s eta 0:00:00
Downloading websockets-13.1-cp312-cp312-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (165 kB)
Building wheels for collected packages: frida-tools
  Building wheel for frida-tools (pyproject.toml) ... done
  Created wheel for frida-tools: filename=frida_tools-13.7.1-py3-none-any.whl size=4639124 sha256=417f88c6b9990aacd3613e47633fe7d3cfffalcd49454c494cc9210474d0ea264
  Stored in directory: /home/krish/.cache/pip/wheels/da/e1/be/e53e57fea97f1b17a6e8933c1a735018363217446708954924
Successfully built frida-tools
```

Step 2:

Verify the installation.

Command – frida --version

```
(krish@krish)~$ frida --version
16.7.14
```

Frida server installation steps:

Step 1:

Find the architecture of your android emulator.

Command - adb shell getprop ro.product.cpu.abi

```
(krish@krish)~$ adb shell getprop ro.product.cpu.abi
x86_64
```

Step 2:

Now, navigate to the frida GitHub repo and download the frida server for your specific architecture.

Ours is x86_64.



Step 3:

Unzip the downloaded file.

Command – xz -d filename

```
(krish@Krish) - [~/Downloads]
$ xz -d frida-server-16.7.14-android-x86_64.xz

(krish@Krish) - [~/Downloads]
$ ls
'app-release (1).apk'      frida-server-16.7.14-android-x86_64
'app-release (1).apk:Zone.Identifier'  frida-server-16.7.14-android-x86_64.xz:Zone.Identifier
```

Step 4:

Remain the Frida server for simpler accessibility.

```
(krish@Krish)-[~/Downloads]
$ mv frida-server-16.7.14-android-x86_64 frida-server

(krish@Krish)-[~/Downloads]
$ ls
'app-release (1).apk'      frida-server
'app-release (1).apk:Zone.Identifier'  frida-server-16.7.14-android-x86_64.xz:Zone.Identifier
```

Step 5:

Move the frida server to the emulator in the below given path.

Command - adb push frida-server /data/local/tmp

```
(krish@Krish)-[~/Downloads]
$ adb push frida-server /data/local/tmp
frida-server: 1 file pushed, 0 skipped. 45.5 MB/s (114296792 bytes in 2.398s)
```

Step 6:

1. Now the frida server is moved to the emulator, we can move to start up.
2. Get shell access to the emulator. Command – adb shell
3. Navigate to /data/local/tmp, here will find the frida-server.
4. Give the frida-server executable permission for execution.

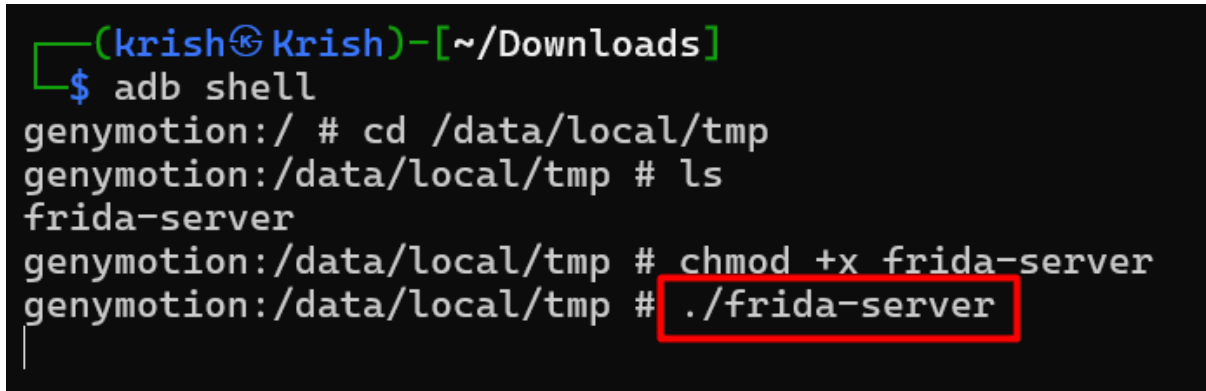
```
(krish@Krish)-[~/Downloads]
$ adb shell
genymotion:/ # cd /data/local/tmp
genymotion:/data/local/tmp # ls
frida-server
genymotion:/data/local/tmp # chmod +x frida-server
genymotion:/data/local/tmp #
```

Step 7:

After giving executable permission, run the frida-server.

Command - `./frida-server`

Once the server is started the current terminal will not be accessible, we can also background this process by using "&" if we need to access the terminal of the emulator.

A terminal window with a black background and green text. The prompt is `(krish@Krish)~[~/Downloads]`. The user enters `$ adb shell`. The prompt changes to `genymotion:/ #`. The user enters `cd /data/local/tmp`. The prompt changes to `genymotion:/data/local/tmp #`. The user enters `ls`. The prompt changes to `frida-server`. The user enters `chmod +x frida-server`. The prompt changes to `genymotion:/data/local/tmp #`. The user enters `./frida-server`, which is highlighted with a red rectangle. The prompt changes to `|`.

```
(krish@Krish)~[~/Downloads]
$ adb shell
genymotion:/ # cd /data/local/tmp
genymotion:/data/local/tmp # ls
frida-server
genymotion:/data/local/tmp # chmod +x frida-server
genymotion:/data/local/tmp # ./frida-server
|
```

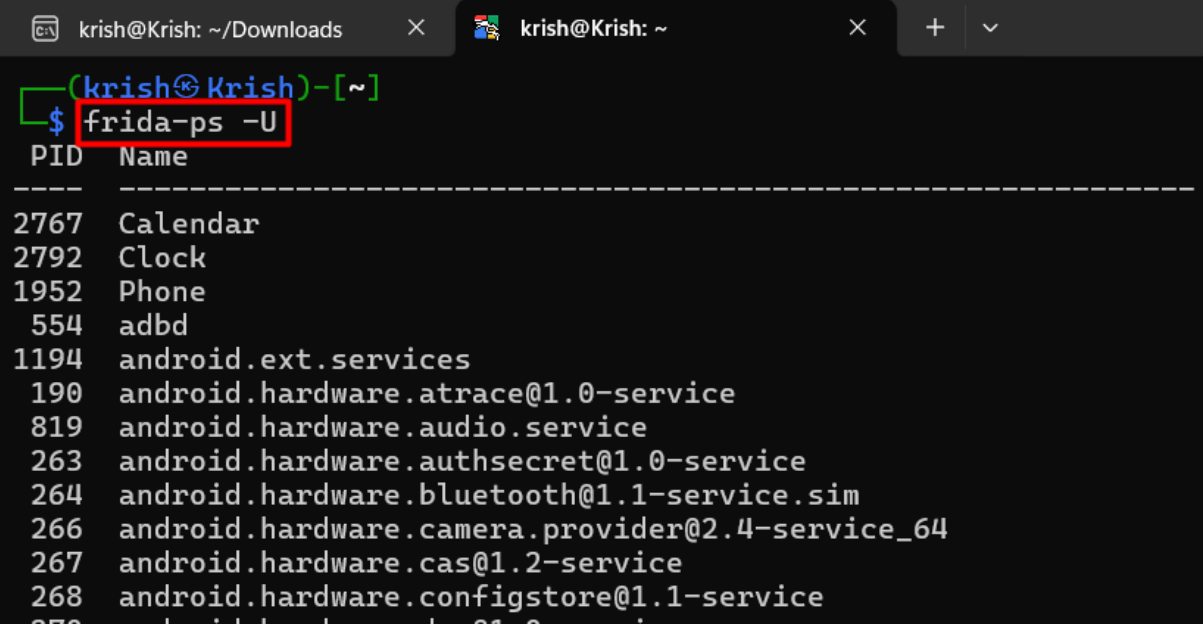
Step 8:

Verify the working of frida-server by running the following command on the host machine.

Command - `frida-ps -U`

If a list of running processes is displayed. Then frida-server is working.

Note: Every time the emulator is restarted or turned off and turned on again. The frida-server must be started by executing the file.



```
krish@Krish: ~/Downloads × krish@Krish: ~ × + v
(krish@Krish)-[~]
$ frida-ps -U
PID  Name
-----
2767  Calendar
2792  Clock
1952  Phone
554   adbd
1194  android.ext.services
190   android.hardware.atrace@1.0-service
819   android.hardware.audio.service
263   android.hardware.authsecret@1.0-service
264   android.hardware.bluetooth@1.1-service.sim
266   android.hardware.camera.provider@2.4-service_64
267   android.hardware.cas@1.2-service
268   android.hardware.configstore@1.1-service
278   android.hardware.configstore@1.1-service
```