1-Installation & Setup

1- Setup

عشان تبدأ في Android Pentest، لازم تنزل نظام Android تشتغل عليه. عندك 3 طرق عشان تنزله:

1. **Smartphone**: يعني تستخدم موبايل حقيقي.
2. **Emulator: تستخدم محاكي زي Android Studio Emulator. # we will use it**
3. **Virtual Machine (VM)**: تثبت النظام على جهاز افتراضي زي VirtualBox أو VMware.
    - **الأفضل**: إنك تشتغل على VM لأنه بيديك تحكم أكتر وحرية في التعديلات.

الفرق بين لما تثبت النظام على موبايل (Smartphone) والجهاز الافتراضي (VM/Emulator)

الميزة	Smartphone	VM & Emulator
الأداء (Performance)	✅ سريع	✅/❌ حسب الموارد اللي مخصصها
الروت (Root Access)	✅/❌ حسب الموبايل	✅ الروت سهل توفره
دعم الميزات (Platform Support)	✅ كل الميزات متاحة	✅/❌ ممكن بعض الميزات مش شغالة
الرسائل (SMS)	✅ متوفر	✅/❌ ممكن مايبقاش شغال
التحديثات (Up to Date)	✅ حسب الموبايل	✅ أحدث النسخ بسهولة

2- Installation

1- إنشاء جهاز افتراضي جديد

- استخدم **VMware** أو **VirtualBox**.

2- إعداد الجهاز الافتراضي

- اضبط الإعدادات كالتالي:
    - **RAM**: من 4 إلى 8 جيجابايت.
    - **Number of Processors**: 4.
    - **Storage**: من 40 إلى 60 جيجابايت.
    - **USB**: اختر USB 3.1.

3- تحميل ملف ISO

- حمل النسخة من الرابط:
  Ubuntu 24.04.2 LTS (amd64)

4- ضبط ملف ISO

- اربط ملف الـ ISO بالجهاز الافتراضي.

5- تشغيل الجهاز الافتراضي

- شغل الجهاز الافتراضي وأكمل إعدادات التثبيت.
- عند ظهور الخيارات، قم باختيار:
  1. **Install third-party software**.
  2. **Install additional media formats**.
  3. أكمل التثبيت.

بعد التثبيت:

1- تحديث الجهاز وإضافة بعض الحزم

- قم بتشغيل الأوامر التالية:

```
sudo apt update & sudo apt install vim git zsh
```

2- تثبيت Oh My Zsh

- استخدم هذا الأمر لتثبيت **Oh My Zsh**:

```
sh -c "$(curl -fsSL
https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"
```

3- تثبيت Android Studio

- حمل Android Studio من الرابط:
  Android Studio
- اتبع تعليمات التثبيت الخاصة بالموقع.

Emulator Usages and Commands

Emulator Commands

1. **List all available AVDs**:
   `emulator -list-avds`
   - This command displays all available Android Virtual Devices (AVDs).
2. **Use a specific AVD**:
   `emulator -avd <avd_name>`
   OR
   `emulator @<avd_name>`
   - Use this command to launch a specific AVD by its name.

ADB (Android Debug Bridge) Commands

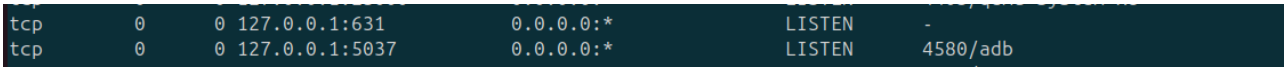
1. **Open Emulator Shell**:

`adb shell`

- This opens the shell environment of the emulator.

2. **Push a file to the emulator**:
   `adb push <file_name> <path>`
   - Upload a file from your local system to the emulator.
3. **Pull a file from the emulator**:
   `adb pull <remote_path> <local_path>`
   - Retrieve a file from the emulator to your local system.

Networking and Ports

1. **Check important ports with `netstat`**:
   `netstat -tulpen`
   - This command lists active ports and can help identify the following:
     - **5554**: Port used for the emulator service.
     - **5037**: Port used by the ADB server.
2. **Connect to an emulator via Telnet**:
   `telnet 127.0.0.1 5554`
   - 

   - Use this command to establish a Telnet connection to the emulator using the **5554** port.

Notes:

- Replace placeholders like `<avd_name>`, `<file_name>`, and `<path>` with actual values relevant to your setup.
- Ensure the emulator and ADB are running properly before executing commands.

Developer Options

1. Connecting a Real Device to Ubuntu

- To connect a real Android device to Ubuntu, you need to enable **Developer Options**.
- **How to enable Developer Options:**
  1. Go to **About Phone**.
  2. Tap multiple times on **OS Version** until Developer Options are enabled.
- **Steps to connect:**
  1. On your phone: Go to **Developer Options** > **Enable USB Debugging**.
  2. Connect your phone to Ubuntu via USB.

These steps are essential for connecting devices. Next, we'll discuss two options in Developer Options commonly used in Android penetration testing.

---

2. Bluetooth HCI Snoop Log

- **Purpose:**
  Captures Bluetooth signals and logs them to analyze the communication between the device and other Bluetooth-enabled devices (e.g., pairing, data exchange).
  - **HCI (Host Controller Interface):** The protocol used for Bluetooth communication.
- After enabling this option, a log report is generated. The log file can then be analyzed using tools like **Wireshark**.

Steps to enable and use:

1. In **Developer Options**, enable **Bluetooth HCI Snoop Log**.
   1. Use the command:
      `adb bugreport`
2. Analyze the log file using Wireshark:
   `wireshark file.log`

---

3. Pointer Location

- **Purpose:**
  Used to track touch coordinates on the device. This is especially useful in game penetration testing when interacting with specific points on the screen.
  For example, if you find coordinates `x=500, y=200`, and you want to tap multiple times at this location, you can use the following command:

Steps to enable and use:

1. In **Developer Options**, enable **Pointer Location**.
2. Open an ADB shell:
   `adb shell`
3. Simulate a tap on the coordinates:
   `input tap 500 200`

Screen Copy with `scrcpy`

- **Purpose:**
  Displays and controls an Android device's screen from a computer.

Steps:

1. Install scrcpy:

`sudo apt install scrcpy`

2. Run scrcpy:

   `scrcpy`

ADB Commands

1. **Connect to a device using IP:**

   `adb connect <IP>`

2. **List connected devices:**

   `adb devices`

3. **Access device shell using serial number:**

   `adb -s <serial> shell`

4. **Push a file to the device:**

   `adb push <file> <destination>`

   - Common directories:
     - `/sdcard` (stores images, videos, etc.)
     - `/tmp` (temporary files)

5. **Pull a file from the device:**

   `adb pull -a <destination> <file_name>`

6. **Reverse port forwarding (device → host):**
   Example:

   - On Ubuntu, start a server:
     `python3 -m http.server 8080`
   - Reverse the port:
     `adb reverse tcp:1234 tcp:8080`
   - Access on emulator:
     `http://127.0.0.1:1234`

7. **Forward port (host → device):**

   `adb forward tcp:<port> tcp:<port>`

8. **Generate a bug report:**

   `adb bugreport`

9. **Access root shell (if permitted):**

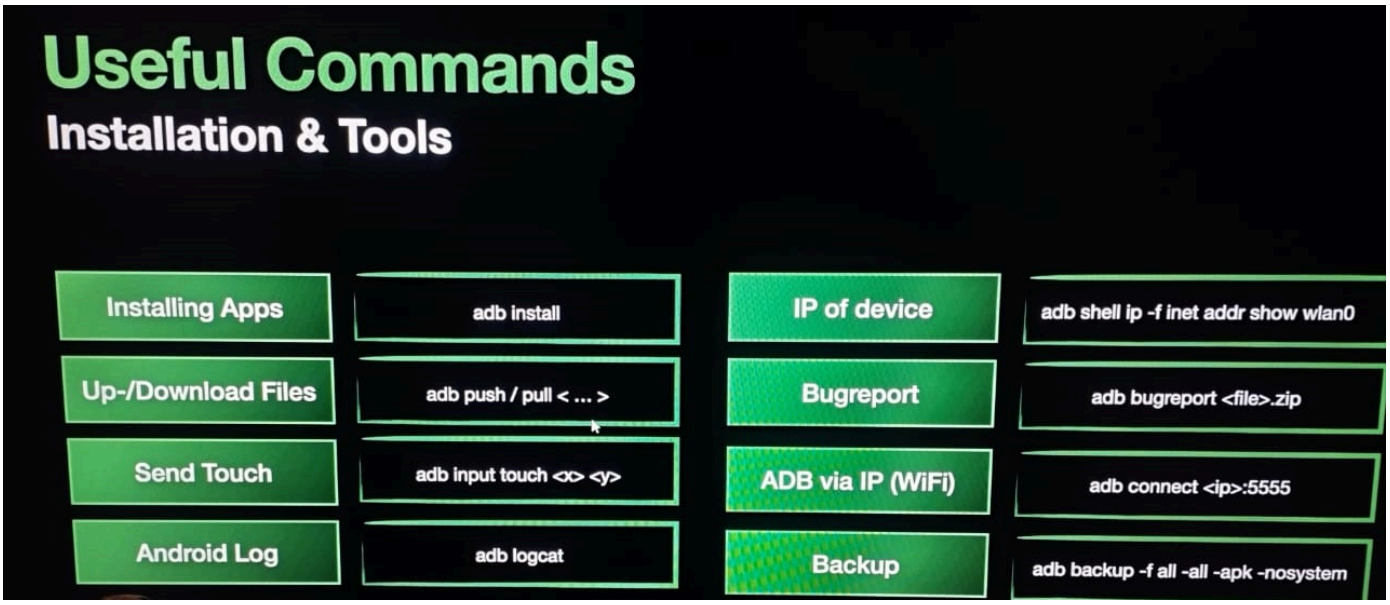
   `adb root && adb shell`

10. **Install an APK file on the device:**

    `adb install <file.apk>`

11. **View device logs:**

    `adb logcat`



how to get root access on emulator on android studio

1- install rootavd from github

```
git clone https://github.com/newbit1/rootAVD.git
```

2- run ./rootavd.sh ListAllAVDs

```
./rootAVD.sh ListAllAVDs
```

3- select line for you system

```
./rootAVD.sh ListAllAVDs
./rootAVD.sh InstallApps

./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img
./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img FAKEBOOTIMG
./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img DEBUG PATCHFSTAB GetUSBHPmodZ
./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img restore
./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img InstallKernelModules
./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img InstallPrebuiltKernelModules
./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img InstallPrebuiltKernelModules GetUSBHPmodZ
 PATCHFSTAB DEBUG
./rootAVD.sh system-images/android-35/google_apis_playstore/x86_64/ramdisk.img AddRCscripts

./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img FAKEBOOTIMG
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img DEBUG PATCHFSTAB GetUSBHPmodZ
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img restore
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img InstallKernelModules
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img InstallPrebuiltKernelModules
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img InstallPrebuiltKernelModules GetUSBHPmodZ PA
TCHFSTAB DEBUG
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img AddRCscripts
```

select this

```
./rootAVD.sh system-images/android-30/google_apis_playstore/x86/ramdisk.img
```

after run this command will install Ramdisk

open Ramdisk and on emulator run adb shell su

on Ramdisk will give you message you want to be root click yes

```
on ubuntu
adb shell
su
on ramdisk
click yes
```