Mobile Application Testing Lab — Lab Setup Commands (Safe, Lab-only

---

Purpose: Quick, copy-paste commands to prepare an isolated mobile-app security lab using MobSF, Frida, and Drozer. All steps are for analysis, instrumentation, and logging — no exploit payloads. Run only in isolated, snapshot-able lab VMs or containers.

1) Environment choices

Recommended: Host with 8+GB RAM. Use an Ubuntu 22.04 / Kali Linux VM as attacker/analysis VM. Use a separate Android device or emulator (Android Virtual Device - AVD) on an isolated network. Use adb over host-only or bridged network for device connectivity.

2) Install MobSF (Static Analysis)

MobSF can run in Docker (recommended) or natively. The commands below use Docker for simplicity.

```
# On Ubuntu/Kali (attacker/analysis VM):
sudo apt update && sudo apt -y install docker.io docker-compose git
sudo systemctl enable --now docker
# clone MobSF
git clone https://github.com/MobSF/Mobile-Security-Framework-MobSF.git ~/tools/MobSF
cd ~/tools/MobSF
# start MobSF using Docker (build once)
sudo ./run.sh
# Access MobSF UI at: http://127.0.0.1:8000 (or host IP) - upload APK for static analysis
# Save MobSF reports in ~/tools/MobSF/StaticAnalysisReports (create folder if needed)
```

3) Set up Android emulator / device

Use Android Emulator (AVD) or a physical device. For emulator, install Android SDK tools and create an AVD. Enable adb and ensure connectivity.

```
# Install Android SDK tools (Linux)
sudo apt -y install openjdk-11-jdk unzip wget
wget https://dl.google.com/android/repository/commandlinetools-linux-*.zip -O cmdline-tools.zip
mkdir -p ~/Android/cmdline-tools && unzip cmdline-tools.zip -d ~/Android/cmdline-tools
export PATH=$PATH:~/Android/cmdline-tools/tools/bin
# Use sdkmanager to install emulator and platform-tools
# Create and start AVD via avdmanager and emulator (follow Android docs)
# Or connect a physical device via USB and enable Developer Options + USB debugging
# Verify adb connection:
adb devices
```

4) Install Frida (Dynamic Analysis & Instrumentation)

Frida lets you hook functions at runtime on Android. Install Frida on the analysis VM and frida-server on the Android device (match frida versions).

```
# On analysis VM (Kali/Ubuntu):
python3 -m venv ~/venvs/frida && source ~/venvs/frida/bin/activate
pip install --upgrade pip
pip install frida-tools frida
deactivate

# On Android device/emulator:
# Download matching frida-server binary from https://github.com/frida/frida/releases
# Push and run frida-server (device must be rooted or emulator with privs):
adb push frida-server /data/local/tmp/
adb shell 'su -c "chmod 755 /data/local/tmp/frida-server"'
adb shell 'su -c "/data/local/tmp/frida-server &"' # run in background (requires root)
```

5) Install Drozer (Optional — for IPC and component testing)

Drozer is older and may require an older Android API or emulator image. Use it for testing exported components, intents, and IPC misconfigurations in lab devices.

```
# On analysis VM:
sudo apt -y install openjdk-11-jdk python3-pip
pip3 install --user drozer
# Start drozer console (legacy):
~/.local/bin/drozer console connect
# On Android device, install drozer-agent APK (lab-only) and start it with proper permissions
# Note: drozer development is less active; consider alternatives like ADB + custom scripts for IPC testing
```

6) Evidence & baseline commands

Create folders, collect APKs, and baseline system state before analysis. Hash reports and artifacts.

```
# On analysis VM:
mkdir -p ~/mobile-lab/evidence && chmod 700 ~/mobile-lab/evidence
# Copy APK to analysis VM
scp user@android:/sdcard/Download/test.apk ~/mobile-lab/evidence/ || cp /path/to/test.apk ~/mobile-lab/evidence/
sha256sum ~/mobile-lab/evidence/test.apk > ~/mobile-lab/evidence/test.apk.sha256
# Run MobSF scan via UI or API and save the JSON/HTML report into evidence folder
# Example: curl MobSF API to scan (if API enabled) and save report (see MobSF docs)
```

7) Frida 50-word summary

Using Frida in a controlled lab, I dynamically hooked authentication-related functions to observe input validation and control flow. By intercepting token checks and logging function arguments/returns, the instrumentation revealed weak client-side checks and insecure storage calls. Findings were recorded, frida scripts archived, and artifacts hashed for evidence.

8) Test Log Template & Checklist (for Google Docs)

Copy the following table and checklist into your Google Docs test plan. Maintain ROE: lab-only, snapshots, and consent for any real devices.

```
Test ID,Vulnerability,Severity,Target App,Notes
016,Insecure Storage,High,test.apk,Found plaintext credentials in shared_prefs

Checklist:
1. Run MobSF static analysis and export report
2. Verify insecure storage findings manually (adb pull /data/data/<app>/shared_prefs/...)
3. Start emulator/device and run frida-server; attach frida to target app
4. Hook authentication and crypto-related functions with Frida scripts
5. Test IPC and exported components with Drozer or adb commands
6. Capture logs, screenshots, and frida traces; hash and archive artifacts
```

9) Safe testing notes

Always use rooted/emulator devices for instrumentation; avoid testing on user-owned devices without consent. Store sensitive artifacts encrypted. Revert emulator snapshots after tests. Keep frida-server binaries and scripts only in your local lab repository (no public distribution).

Generated: 2025-09-21 18:56:08Z — Lab-only commands, no exploit payloads included.