Mobile Security and Forensics Tools

Under the Guidance of Dr. Vikash Rai Sir

Presented By: Saloni Rangari

Enrolment No.: 240347007003

M.Tech. Artificial Intelligence & Data Science (Specialization In Cyber Security)

Session Year: 2024-26

1st Year 2nd Sem

Subject Name: Mobile Security and Forensics

Subject Code: CTMTAIDS SII P2

Introduction

What is Mobile Security?

- ➢ Protection of mobile devices and applications from cyber threats.
- ➢ Involves securing data, network, and device integrity.

What is Mobile Forensics?

- ➢ Extraction, preservation, and analysis of mobile device data for investigation.

Importance of Mobile Security

Increasing number of mobile threats.

Sensitive personal and financial data at risk.

Rising cases of malware, phishing, and exploits.

Need for security analysis tools.

Overview of Security Tools

1. **QARK** – Automated Security Analysis For Android.
2. **Frida** – Dynamic instrumentation toolkit.
3. **MobSF** – Mobile Security Framework.
4. **Drozer** – Android Security Testing Tool.
5. **Xposed Framework** – Modifies Android Behavior Without Changing APK.

QARK (Quick Android Review Kit) - Tool Overview

- **Purpose:**
  - QARK is an automated tool designed for Android security auditing. It identifies common vulnerabilities in Android apps by analyzing APKs and source code.

- **Features:**
  - Performs static analysis of APKs and source code.
  - Detects a wide range of vulnerabilities, including insecure storage, hardcoded secrets, and potential attack vectors.

- **Supported Vulnerabilities:**
  - Insecure WebView usage.
  - Missing proguard rules.
  - Hardcoded secrets.
  - Improper usage of SSL.

QARK - Installation Steps

1. Clone the repository: `git clone https://github.com/linkedin/qark.git`

2. Navigate into the QARK directory: `cd qark`

3. Install pip if not already installed: `sudo apt install python-pip`

4. Install QARK: `sudo python3 setup.py install`

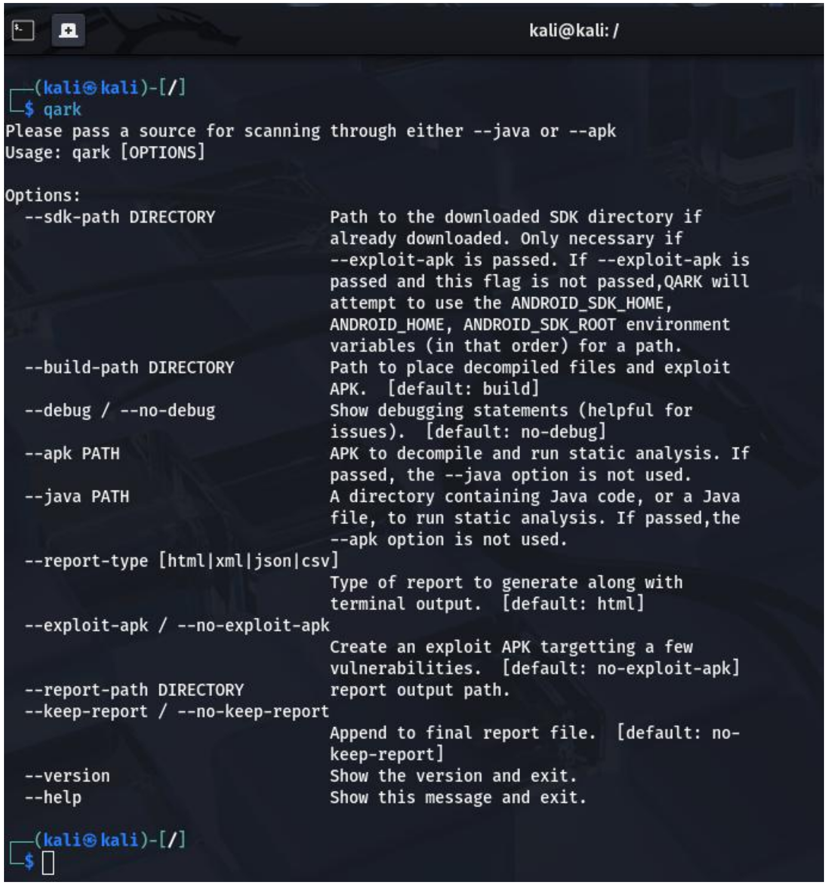
QARK - Usage

1. Run QARK with the APK or source code path: `qark --source <path_to_apk_or_source_code>`

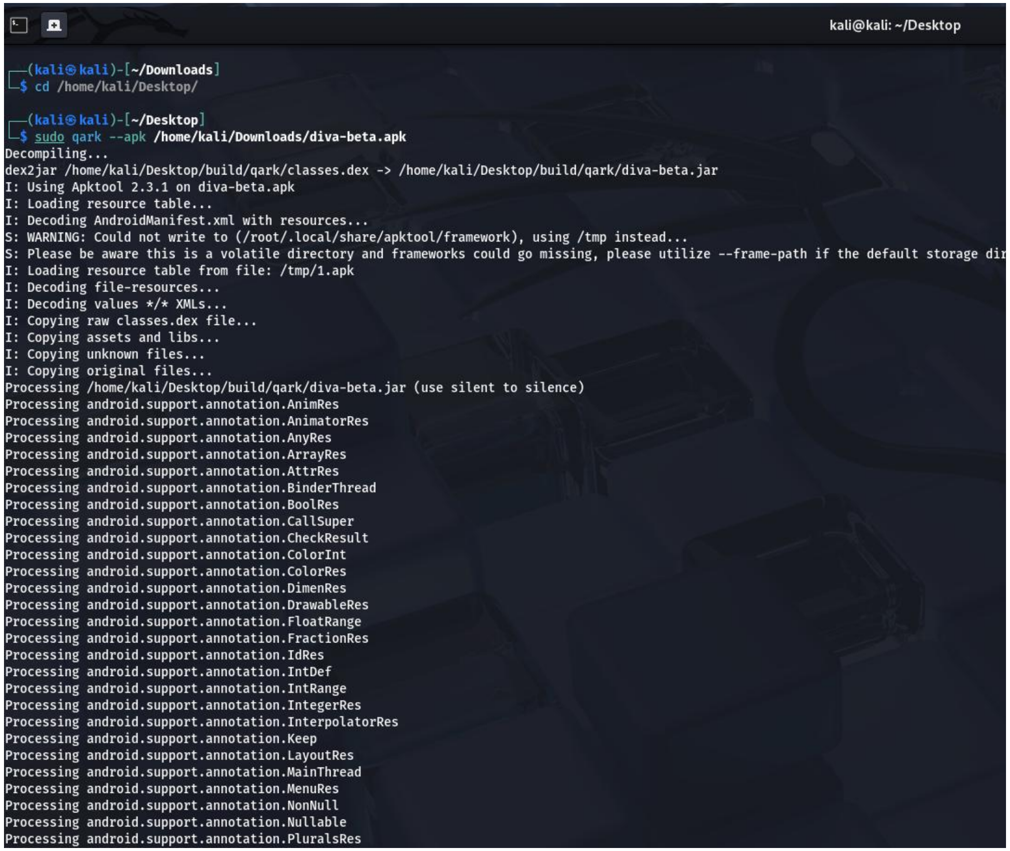
2. Review the generated report for security issues.

The report includes a list of detected vulnerabilities with recommendations for mitigation.

QARK - Screenshot

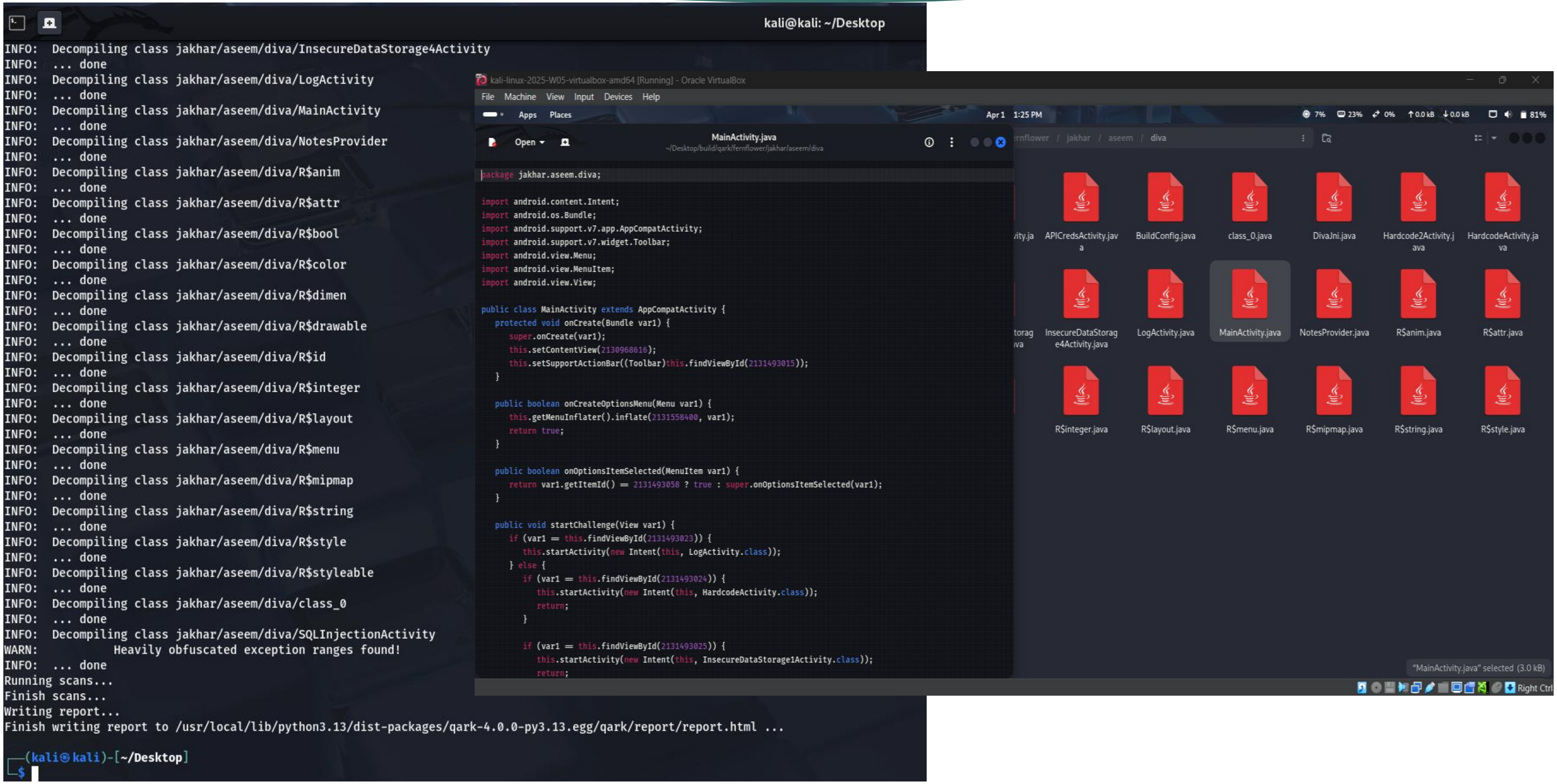


```
┌──(kali㉿kali)-[/]
└─$ qark
Please pass a source for scanning through either --java or --apk
Usage: qark [OPTIONS]

Options:
  --sdk-path DIRECTORY              Path to the downloaded SDK directory if
                                    already downloaded. Only necessary if
                                    --exploit-apk is passed. If --exploit-apk is
                                    passed and this flag is not passed,QARK will
                                    attempt to use the ANDROID_SDK_HOME,
                                    ANDROID_HOME, ANDROID_SDK_ROOT environment
                                    variables (in that order) for a path.
  --build-path DIRECTORY            Path to place decompiled files and exploit
                                    APK.  [default: build]
  --debug / --no-debug              Show debugging statements (helpful for
                                    issues).  [default: no-debug]
  --apk PATH                        APK to decompile and run static analysis. If
                                    passed, the --java option is not used.
  --java PATH                       A directory containing Java code, or a Java
                                    file, to run static analysis. If passed,the
                                    --apk option is not used.
  --report-type [html|xml|json|csv]
                                    Type of report to generate along with
                                    terminal output.  [default: html]
  --exploit-apk / --no-exploit-apk
                                    Create an exploit APK targetting a few
                                    vulnerabilities.  [default: no-exploit-apk]
  --report-path DIRECTORY           report output path.
  --keep-report / --no-keep-report
                                    Append to final report file.  [default: no-
                                    keep-report]
  --version                         Show the version and exit.
  --help                            Show this message and exit.

┌──(kali㉿kali)-[/]
└─$ ▯
```



```
┌──(kali㉿kali)-[~/Downloads]
└─$ cd /home/kali/Desktop/

┌──(kali㉿kali)-[~/Desktop]
└─$ sudo qark --apk /home/kali/Downloads/diva-beta.apk
Decompiling...
dex2jar /home/kali/Desktop/build/qark/classes.dex -> /home/kali/Desktop/build/qark/diva-beta.jar
I: Using Apktool 2.3.1 on diva-beta.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
S: WARNING: Could not write to (/root/.local/share/apktool/framework), using /tmp instead...
S: Please be aware this is a volatile directory and frameworks could go missing, please utilize --frame-path if the default storage dir
I: Loading resource table from file: /tmp/1.apk
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Copying raw classes.dex file...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
Processing /home/kali/Desktop/build/qark/diva-beta.jar (use silent to silence)
Processing android.support.annotation.AnimRes
Processing android.support.annotation.AnimatorRes
Processing android.support.annotation.AnyRes
Processing android.support.annotation.ArrayRes
Processing android.support.annotation.AttrRes
Processing android.support.annotation.BinderThread
Processing android.support.annotation.BoolRes
Processing android.support.annotation.CallSuper
Processing android.support.annotation.CheckResult
Processing android.support.annotation.ColorInt
Processing android.support.annotation.ColorRes
Processing android.support.annotation.DimenRes
Processing android.support.annotation.DrawableRes
Processing android.support.annotation.FloatRange
Processing android.support.annotation.FractionRes
Processing android.support.annotation.IdRes
Processing android.support.annotation.IntDef
Processing android.support.annotation.IntRange
Processing android.support.annotation.IntegerRes
Processing android.support.annotation.InterpolatorRes
Processing android.support.annotation.Keep
Processing android.support.annotation.LayoutRes
Processing android.support.annotation.MainThread
Processing android.support.annotation.MenuRes
Processing android.support.annotation.NonNull
Processing android.support.annotation.Nullable
Processing android.support.annotation.PluralsRes
```

QARK - Screenshot



```
INFO:  Decompiling class jakhar/aseem/diva/InsecureDataStorage4Activity
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/LogActivity
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/MainActivity
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/NotesProvider
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$anim
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$attr
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$bool
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$color
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$dimen
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$drawable
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$id
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$integer
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$layout
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$menu
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$mipmap
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$string
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$style
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/R$styleable
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/class_0
INFO:  ... done
INFO:  Decompiling class jakhar/aseem/diva/SQLInjectionActivity
WARN:       Heavily obfuscated exception ranges found!
INFO:  ... done
Running scans...
Finish scans...
Writing report...
Finish writing report to /usr/local/lib/python3.13/dist-packages/qark-4.0.0-py3.13.egg/qark/report/report.html ...

┌──(kali㉿kali)-[~/Desktop]
└─$ 
```

```java
package jakhar.aseem.diva;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;

public class MainActivity extends AppCompatActivity {
   protected void onCreate(Bundle var1) {
      super.onCreate(var1);
      this.setContentView(2130968616);
      this.setSupportActionBar((Toolbar)this.findViewById(2131493015));
   }

   public boolean onCreateOptionsMenu(Menu var1) {
      this.getMenuInflater().inflate(2131558400, var1);
      return true;
   }

   public boolean onOptionsItemSelected(MenuItem var1) {
      return var1.getItemId() == 2131493058 ? true : super.onOptionsItemSelected(var1);
   }

   public void startChallenge(View var1) {
      if (var1 == this.findViewById(2131493023)) {
         this.startActivity(new Intent(this, LogActivity.class));
      } else {
         if (var1 == this.findViewById(2131493024)) {
            this.startActivity(new Intent(this, HardcodeActivity.class));
            return;
         }

         if (var1 == this.findViewById(2131493025)) {
            this.startActivity(new Intent(this, InsecureDataStorage1Activity.class));
            return;
```

Frida - Tool Overview

- **Purpose:**
  - Frida is a dynamic instrumentation tool that allows developers and researchers to perform reverse engineering of mobile apps.

- **Features:**
  - It can intercept and modify API calls at runtime.
  - Useful for bypassing SSL pinning, inspecting app behavior, and reverse engineering.

- **Supported Platforms:**
  - Android and iOS.

QARK - Installation Steps

1. Install Frida on Windows: `pip install frida`

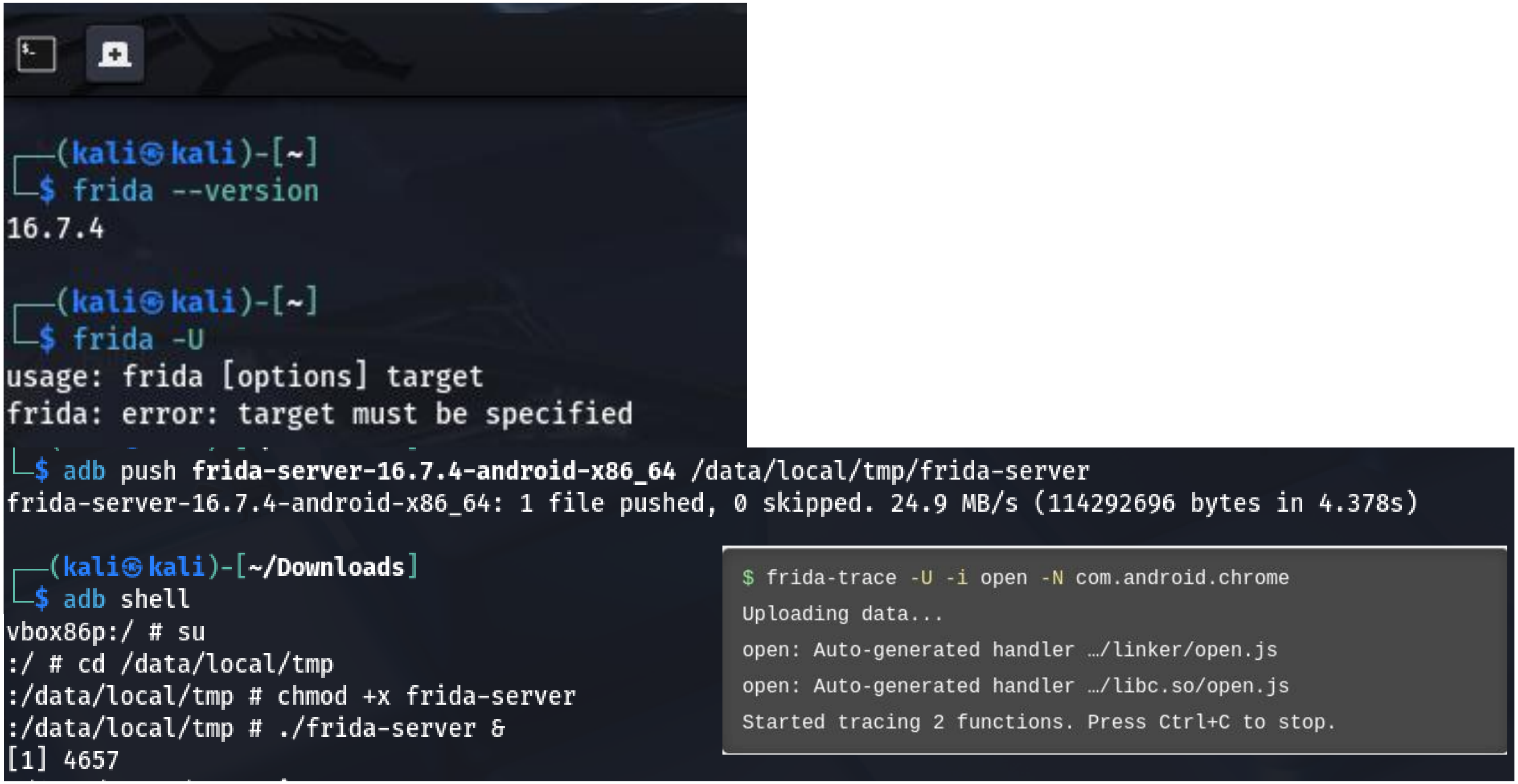
2. Navigate into the QARK directory: `cd qark`

3. Install additional dependencies, such as `libimobiledevice` for iOS support.

Frida - Usage

- 1. Start Frida with the app's process name: `frida -U -n <app_name>`
- 2. Install Frida on Kali Linux: `sudo apt install frida`
- 3. Example script usage to hook into a method: `frida -U -n <app_name> -l hook.js`

Frida - Screenshot



```
┌──(kali㉿kali)-[~]
└─$ frida --version
16.7.4

┌──(kali㉿kali)-[~]
└─$ frida -U
usage: frida [options] target
frida: error: target must be specified
```

```
└─$ adb push frida-server-16.7.4-android-x86_64 /data/local/tmp/frida-server
frida-server-16.7.4-android-x86_64: 1 file pushed, 0 skipped. 24.9 MB/s (114292696 bytes in 4.378s)

┌──(kali㉿kali)-[~/Downloads]
└─$ adb shell
vbox86p:/ # su
:/ # cd /data/local/tmp
:/data/local/tmp # chmod +x frida-server
:/data/local/tmp # ./frida-server &
[1] 4657
```

```
$ frida-trace -U -i open -N com.android.chrome
Uploading data...
open: Auto-generated handler …/linker/open.js
open: Auto-generated handler …/libc.so/open.js
Started tracing 2 functions. Press Ctrl+C to stop.
```

MobSF (Mobile Security Framework) - Tool Overview

- **Purpose:**
  - MobSF is an all-in-one mobile application security testing framework that provides both static and dynamic analysis for Android and iOS apps.
- **Features:**
  - Static analysis of APK and IPA files.
  - Dynamic analysis using the integrated mobile app testing environment.
  - Malware analysis, API monitoring, and more.
- **Supported Platforms:**
  - Android and iOS.

MobSF - Installation Steps

1. Clone the repository: `git clone https://github.com/MobSF/Mobile-Security-Framework-MobSF.git`

2. Navigate to the MobSF directory: `cd Mobile-Security-Framework-MobSF`

3. Run the setup script (for Kali Linux): `./setup.sh3. Install pip if not already installed: `sudo apt install python-pip`

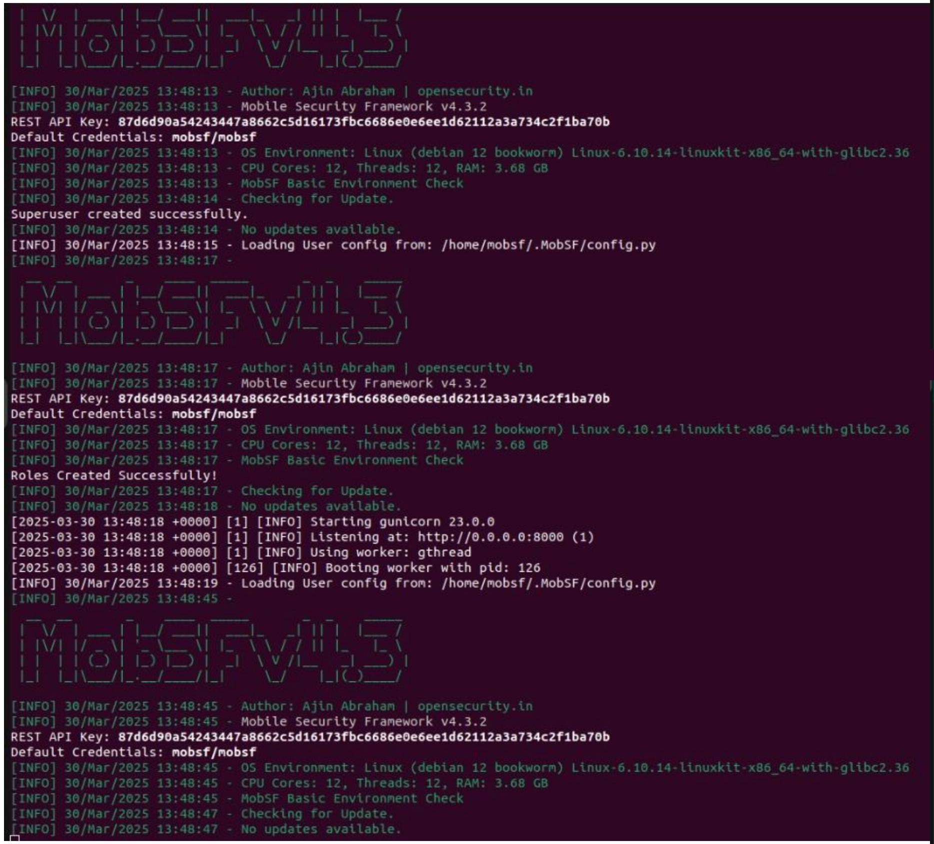
MobSF - Usage

1. Start the MobSF server: `python3 manage.py runserver`

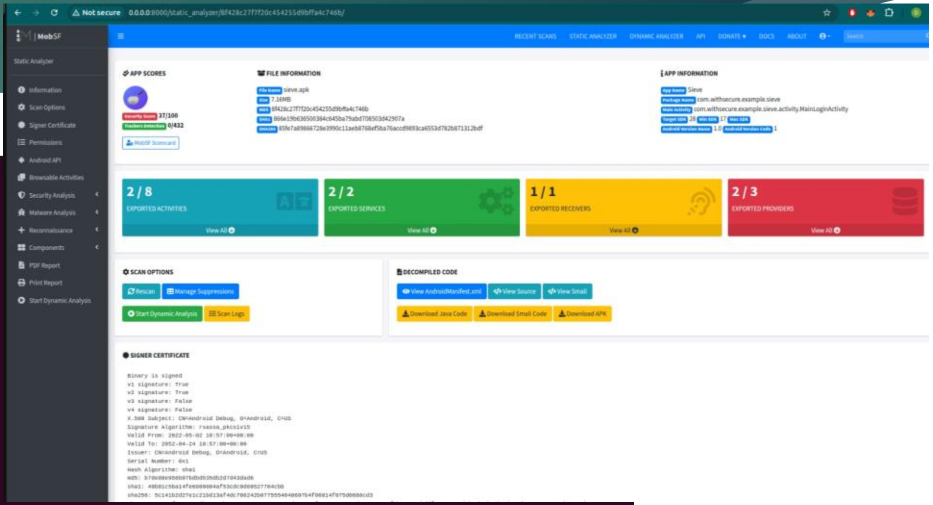
2. Access via the browser: `http://localhost:8000`

3. Upload APK/IPA files for analysis and receive detailed security reports.

MobSF - Screenshot



```
[INFO] 30/Mar/2025 13:48:13 - Author: Ajin Abraham | opensecurity.in
[INFO] 30/Mar/2025 13:48:13 - Mobile Security Framework v4.3.2
REST API Key: 87d6d90a54243447a8662c5d16173fbc6686e0e6ee1d62112a3a734c2f1ba70b
Default Credentials: mobsf/mobsf
[INFO] 30/Mar/2025 13:48:13 - OS Environment: Linux (debian 12 bookworm) Linux-6.10.14-linuxkit-x86_64-with-glibc2.36
[INFO] 30/Mar/2025 13:48:13 - CPU Cores: 12, Threads: 12, RAM: 3.68 GB
[INFO] 30/Mar/2025 13:48:13 - MobSF Basic Environment Check
[INFO] 30/Mar/2025 13:48:14 - Checking for Update.
Superuser created successfully.
[INFO] 30/Mar/2025 13:48:14 - No updates available.
[INFO] 30/Mar/2025 13:48:15 - Loading User config from: /home/mobsf/.MobSF/config.py
[INFO] 30/Mar/2025 13:48:17 -

[INFO] 30/Mar/2025 13:48:17 - Author: Ajin Abraham | opensecurity.in
[INFO] 30/Mar/2025 13:48:17 - Mobile Security Framework v4.3.2
REST API Key: 87d6d90a54243447a8662c5d16173fbc6686e0e6ee1d62112a3a734c2f1ba70b
Default Credentials: mobsf/mobsf
[INFO] 30/Mar/2025 13:48:17 - OS Environment: Linux (debian 12 bookworm) Linux-6.10.14-linuxkit-x86_64-with-glibc2.36
[INFO] 30/Mar/2025 13:48:17 - CPU Cores: 12, Threads: 12, RAM: 3.68 GB
[INFO] 30/Mar/2025 13:48:17 - MobSF Basic Environment Check
Roles Created Successfully!
[INFO] 30/Mar/2025 13:48:17 - Checking for Update.
[INFO] 30/Mar/2025 13:48:18 - No updates available.
[2025-03-30 13:48:18 +0000] [1] [INFO] Starting gunicorn 23.0.0
[2025-03-30 13:48:18 +0000] [1] [INFO] Listening at: http://0.0.0.0:8000 (1)
[2025-03-30 13:48:18 +0000] [1] [INFO] Using worker: gthread
[2025-03-30 13:48:18 +0000] [126] [INFO] Booting worker with pid: 126
[INFO] 30/Mar/2025 13:48:19 - Loading User config from: /home/mobsf/.MobSF/config.py
[INFO] 30/Mar/2025 13:48:45 -

[INFO] 30/Mar/2025 13:48:45 - Author: Ajin Abraham | opensecurity.in
[INFO] 30/Mar/2025 13:48:45 - Mobile Security Framework v4.3.2
REST API Key: 87d6d90a54243447a8662c5d16173fbc6686e0e6ee1d62112a3a734c2f1ba70b
Default Credentials: mobsf/mobsf
[INFO] 30/Mar/2025 13:48:45 - OS Environment: Linux (debian 12 bookworm) Linux-6.10.14-linuxkit-x86_64-with-glibc2.36
[INFO] 30/Mar/2025 13:48:45 - CPU Cores: 12, Threads: 12, RAM: 3.68 GB
[INFO] 30/Mar/2025 13:48:45 - MobSF Basic Environment Check
[INFO] 30/Mar/2025 13:48:47 - Checking for Update.
[INFO] 30/Mar/2025 13:48:47 - No updates available.
```



APP SCORES

Security Score 37/100
Trackers Detection 0/432

FILE INFORMATION

File Name sieve.apk
Size 7.16MB
MD5 8f428c27f7f20c454255d9bffa4c746b
SHA1 866e19b636500384c645ba79abd708503d42907a
SHA256 85fe7a89866728e3990c11aeb8768ef5ba76accd9893ca6553d782b871312bdf

APP INFORMATION

App Name Sieve
Package Name com.withsecure.example.sieve
Main Activity com.withsecure.example.sieve.activity.MainLoginActivity
Target SDK 28 Min SDK 17 Max SDK
Android Version Name 1.0 Android Version Code 1

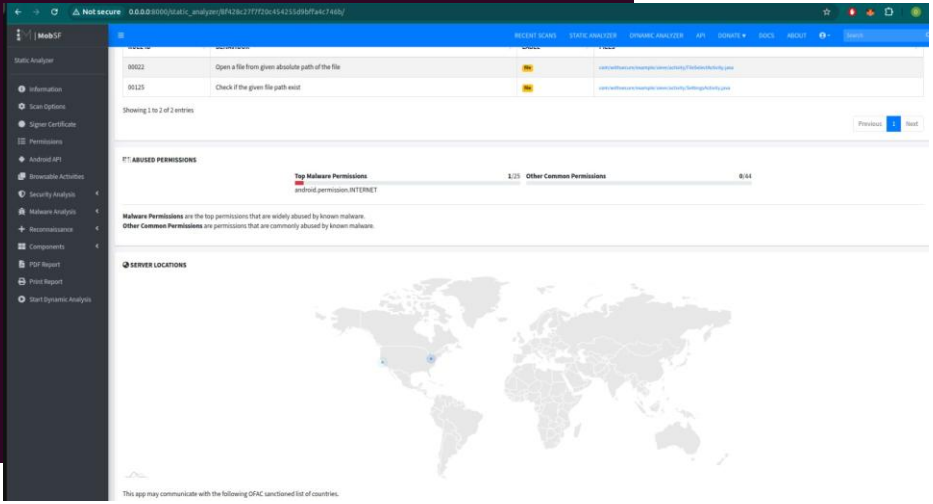
2 / 8 EXPORTED ACTIVITIES
2 / 2 EXPORTED SERVICES
1 / 1 EXPORTED RECEIVERS
2 / 3 EXPORTED PROVIDERS

SCAN OPTIONS

DECOMPILED CODE

SIGNER CERTIFICATE

```
Binary is signed
v1 signature: True
v2 signature: True
v3 signature: False
v4 signature: False
X.509 Subject: CN=Android Debug, O=Android, C=US
Signature Algorithm: rsassa_pkcs1v15
Valid From: 2022-05-02 10:57:00+00:00
Valid To: 2052-04-24 10:57:00+00:00
Issuer: CN=Android Debug, O=Android, C=US
Serial Number: 0x1
Hash Algorithm: sha1
md5: 57de08e956b07bdbd535db2d7d43dad6
sha1: 49b01c5ba14fe6080084af53cdc9d60527764cb0
sha256: 5c141b2d27e1c21bd13af4dc786242b0775554048697b4f86814f875d0688cd3
```



00022	Open a file from given absolute path of the file	file	com/withsecure/example/sieve/activity/FileSelectActivity.java
00125	Check if the given file path exist	file	com/withsecure/example/sieve/activity/SettingsActivity.java

Showing 1 to 2 of 2 entries

ABUSED PERMISSIONS

Top Malware Permissions 1/25
android.permission.INTERNET
Other Common Permissions 0/44

Malware Permissions are the top permissions that are widely abused by known malware.
Other Common Permissions are permissions that are commonly abused by known malware.

SERVER LOCATIONS

This app may communicate with the following OFAC sanctioned list of countries.

Drozer - Tool Overview

- **Purpose:**
  - Drozer is a security testing framework designed for Android applications, enabling security researchers to test app components and interact with vulnerable apps.
- **Features:**
  - Allows exploitation of Android vulnerabilities.
  - Provides access to a wide array of app components like activities, services, and content providers.
- **Supported Platforms:**
  - Android

Drozer - Installation Steps

1. Install Drozer on Windows: `pip install drozer`

2. Install Drozer on Kali Linux: `sudo apt install drozer`

3. Download and install Drozer Agent on the Android device from GitHub: `adb install drozer-agent.apk`

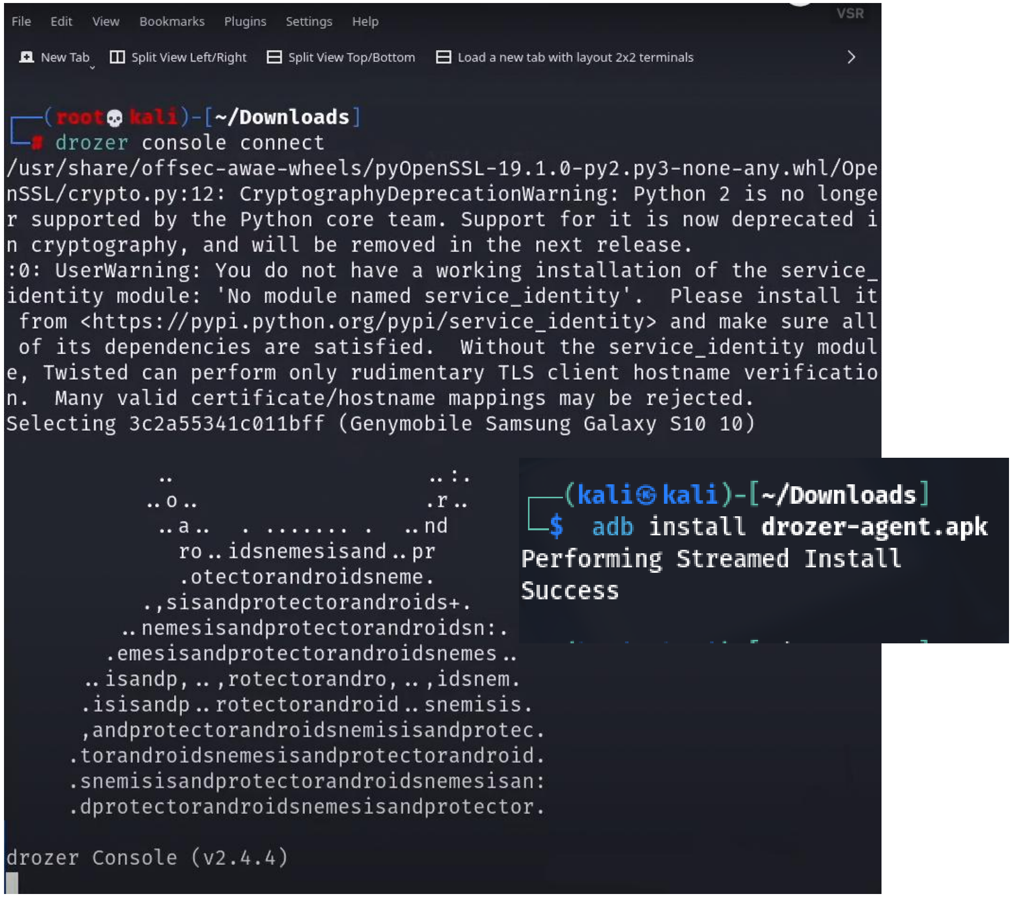
Drozer - Usage

1. Start the drozer console: `drozer console connect`

2. List installed apps: `run app.package.list`

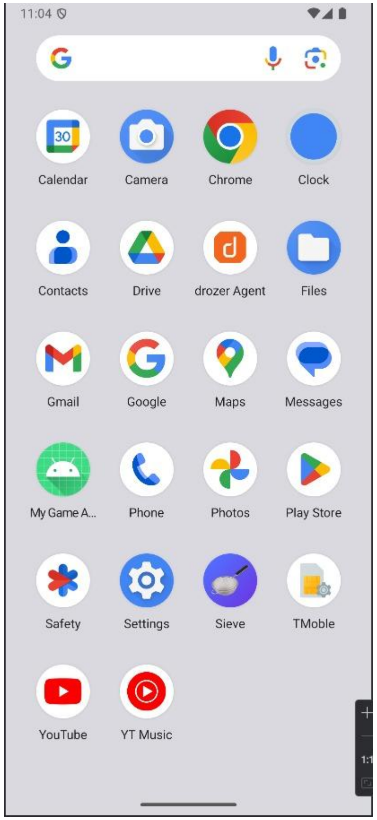
3. Interact with app components: `run app.activity.info -a <package_name>`

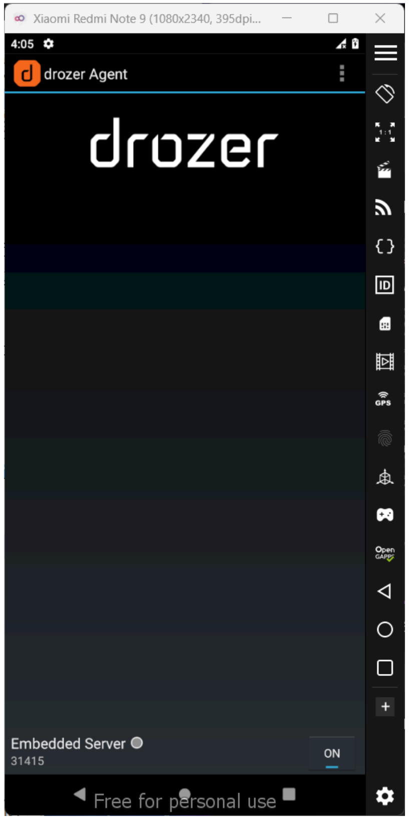
Drozer - Screenshot



```
┌──(root💀kali)-[~/Downloads]
└─# drozer console connect
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/Ope
nSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longe
r supported by the Python core team. Support for it is now deprecated i
n cryptography, and will be removed in the next release.
:0: UserWarning: You do not have a working installation of the service_
identity module: 'No module named service_identity'.  Please install it
 from <https://pypi.python.org/pypi/service_identity> and make sure all
 of its dependencies are satisfied.  Without the service_identity modul
e, Twisted can perform only rudimentary TLS client hostname verificatio
n.  Many valid certificate/hostname mappings may be rejected.
Selecting 3c2a55341c011bff (Genymobile Samsung Galaxy S10 10)

            ..                    ..:.
           ..o..                  .r..
            ..a..  . ....... .  ..nd
              ro..idsnemesisand..pr
              .otectorandroidsneme.
           .,sisandprotectorandroids+.
         ..nemesisandprotectorandroidsn:.
        .emesisandprotectorandroidsnemes..
      ..isandp,..,rotectorandro,..,idsnem.
      .isisandp..rotectorandroid..snemisis.
      ,andprotectorandroidsnemisisandprotec.
     .torandroidsnemesisandprotectorandroid.
     .snemisisandprotectorandroidsnemesisan:
     .dprotectorandroidsnemesisandprotector.

drozer Console (v2.4.4)
```

```
┌──(kali㉿kali)-[~/Downloads]
└─$ adb install drozer-agent.apk
Performing Streamed Install
Success
```





Embedded Server
31415
ON

Xposed Framework

- **Purpose:**
  - Framework to modify Android behavior without modifying APKs.
  - Helps in security testing, app debugging, and feature enhancement.
  - Allows runtime modification of system and app behavior.
- **Features:**
  - Hooks into Android apps and system processes.
  - Enables customization without modifying system files.
  - Used for reverse engineering and security testing.
  - Supports modules for extensive modification
- **Supported Platforms:**
  - Android (requires root access).
  - Works with Magisk for systemless installation.
  - Supports Android versions from Lollipop to the latest (with compatible modules).

Xposed - Installation Steps

- From Windows 11:
- Install Xposed Installer APK
- From Kali Linux:
- Install via Magisk or Recovery

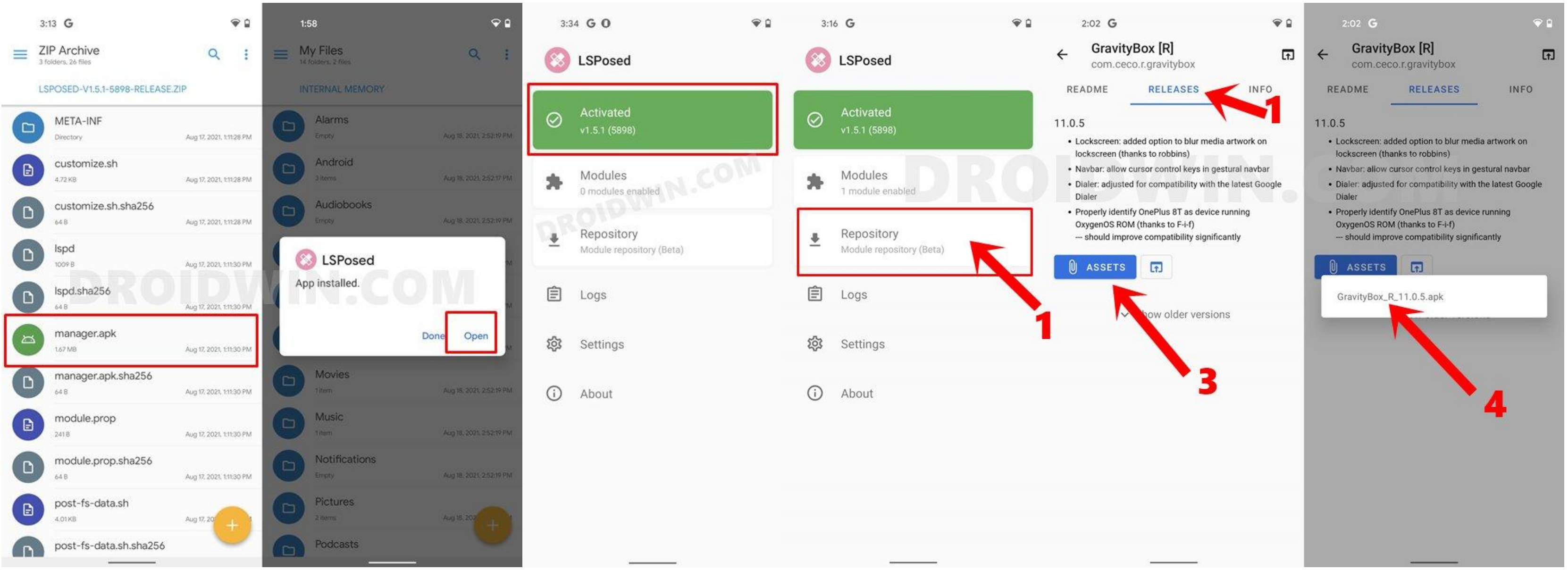
Xposed Usage

1. Open Xposed Installer

2. Enable modules and reboot

3. Use modules for security analysis

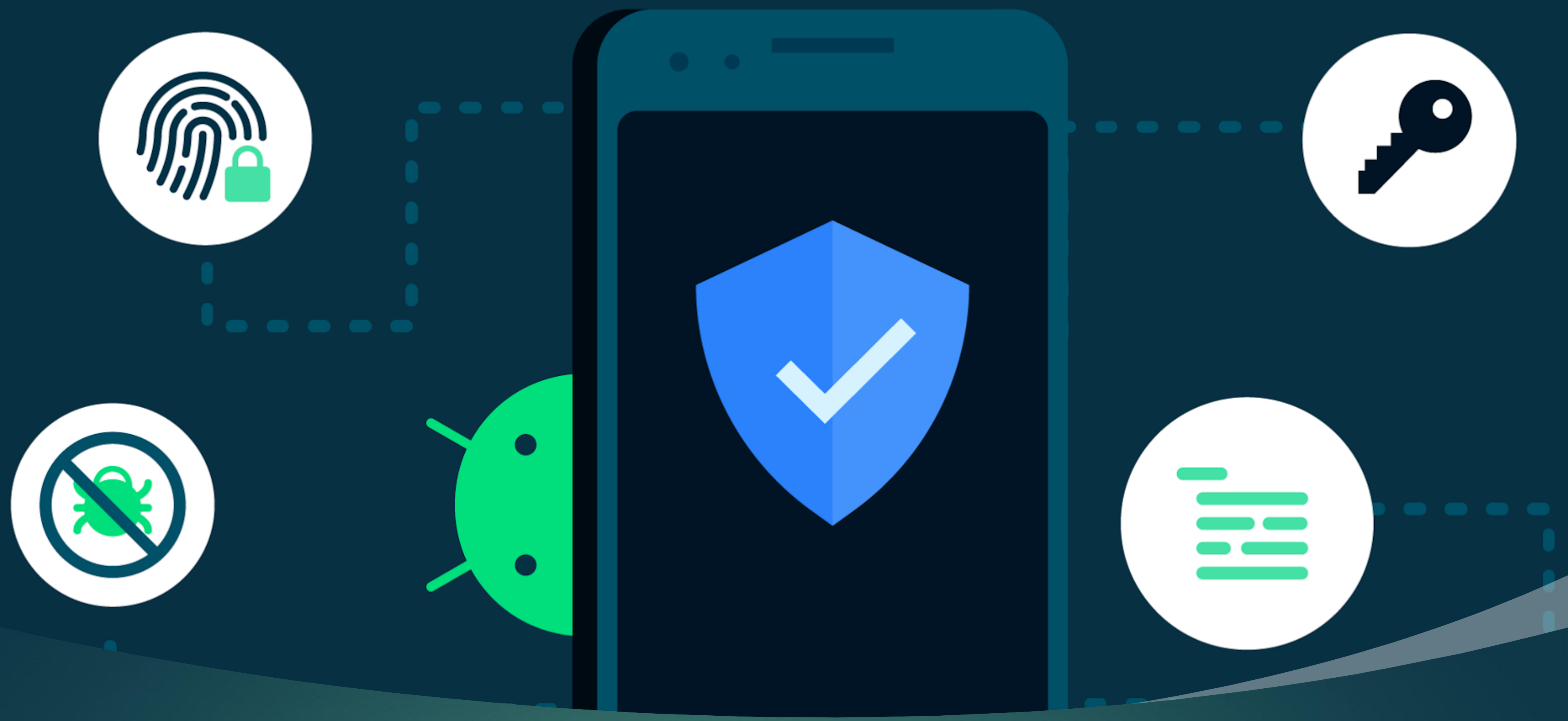
Xposed - Screenshot



App installed.

11.0.5

- Lockscreen: added option to blur media artwork on lockscreen (thanks to robbins)
- Navbar: allow cursor control keys in gestural navbar
- Dialer: adjusted for compatibility with the latest Google Dialer
- Properly identify OnePlus 8T as device running OxygenOS ROM (thanks to F-i-f) --- should improve compatibility significantly



Thank You