Internship Report for NullClass

Name: Mannava Sai Bhavana
Internship Duration: September 9, 2024 – October 9, 2024
Mentor/Organization: NullClass
Role: Cybersecurity Intern
NullClass Username: SAI BHAVANA MANNAVA

4. Activities and Tasks

Task 3: Decompile an APK File

Description:

For this task, I instructed users on how to decompile APK files using tools such as JADX and APKTool. The objective was to extract the source code and resources from an APK file to analyze the application's behavior and detect potential vulnerabilities.

Steps:

1. Basic Setup:
   Before Starting the decompilation process, I ensured that my development environment was correctly configured. This included installing the necessary tools:
   a) Install JDK:
      - Download the latest version of the JDK from the Oracle website.
      - Install it by following the prompts specific to your operating system.
      - Ensure that java is added to your system's PATH. You can check this by running:
        "java -version "
   b) Download JADX:
      - Visit the official JADX GitHub repository.
      - Download the latest release and unzip it to a preferred directory on your system.
   c) Install APKTool:
      - Check Existing APKTool Installation: Initially, I opened the terminal in Kali Linux to verify if APKTool was already installed. I executed the following command: **"apktool --version".**
      - The output I received was: **"2.7.0-dirty "**This indicated that APKTool was already installed on my system.
      - Removing Existing APKTool (if necessary): In case I needed to reinstall APKTool for any reason, I used the command to remove it: **"sudo apt remove apktool"**

- Update Package List: After removing APKTool, I updated the package list to ensure that I had the latest version available in the repositories. I executed the following command: **"sudo apt update"**
- Install APKTool: With the package list updated, I proceeded to install APKTool by running: **"sudo apt install apktool"** This command downloaded and installed the latest version of APKTool from the repository.
- Verification of Installation: After installation, I verified that APKTool was installed correctly by re-running the version check command: **"apktool --version"** The output should confirm the successful installation of APKTool.

2. Downloading the APK:
   - I downloaded the target APK file, **DivaApplication.apk**, from GitHub.
   - The link to download the apk: Download DivaApplication.apk
   - This application served as a practical example for my decompilation process.
3. Using JADX for Decompilation:
   - **Open JADX-GUI**: I launched the JADX graphical user interface (GUI). The interface allows users to visually navigate through the decompiled code.
   - **Decompile the APK**: To decompile the APK, I simply dragged and dropped **DivaApplication.apk** into the JADX-GUI window. This process automatically initiated the decompilation, and the application provided an organized view of the extracted code.
4. Analyzing Decompiled Files:
   Once the decompilation was complete, I focused on several critical files and folders within the extracted APK structure:
   - **AndroidManifest.xml**: This file is essential as it contains crucial information about the application, including its components (activities, services, broadcast receivers), permissions, and configuration settings.
   - **/src**: The source folder includes the decompiled Java code, which is essential for understanding the application's functionality and logic. I specifically looked for files related to key functionalities and user interactions.
   - **/res**: This folder contains resources such as layout files, images, and strings. Analyzing these resources provides insight into the application's user interface and localization.
   - **/lib**: If present, this folder contains native libraries that the application may rely on. It is important to inspect these if the application utilizes any C/C++ code.
   - **APK Signature:** A vital security feature that verifies the authenticity and integrity of the application. It ensures that the APK has not been modified after it was signed by the developer. The signature is created using a private key, and it is crucial for establishing trust with the users and the Android operating system.

- **META-INF:** directory is an essential component of the APK structure, housing files that provide critical information regarding the APK's signing and packaging. This directory plays a significant role in the verification process during installation, ensuring that the application is secure and unaltered.
- **META-INF/MANIFEST.MF:** The **MANIFEST.MF** file, located within the **META-INF** directory, contains metadata about the APK. This file outlines the contents of the APK, including the files it contains and their corresponding checksums. It is instrumental in the integrity verification process, helping to confirm that all files are intact and have not been tampered with since the APK was signed.
- **META-INF/CERT.RSA:** The **CERT.RSA** file, also found in the **META-INF** directory, includes the digital certificate used to sign the APK. This certificate contains the developer's public key and is essential for verifying the APK's signature during installation. If the APK has been altered post-signing, the signature will fail to match, preventing the installation from proceeding.

5. Using APKTool for Decompilation:
    - **Choose an APK File:** Download the APK file you wish to decompile. For this example, let's assume you are using a sample APK file named DivaApplication.apk.
    - **Move the APK File:** Ensure the APK file is in an accessible directory. You can move it to your home directory for convenience:
      "mv ~/Desktop/DivaApplication.apk ~/"
    - **Navigate to the Directory:** Use the terminal to navigate to the directory where the APK file is located. For instance, if the file is in your home directory: **"cd ~/"** (or) **"cd .."**
    - **Decompile the APK:** Use the following command to decompile the APK file:**" apktool d DivaApplication.apk"**
      In this command, d stands for "decompile." After executing the command, APKTool will create a new directory named DivaApplication (or the name of your APK file without the .apk extension) containing the decompiled files.
    - **Navigate to the Decompiled Directory:** Change to the directory that contains the decompiled APK files: **"cd DivaApplication"**
6. Analyzing Decompiled Files:
   The decompiled directory will contain several important files and folders:
    - **AndroidManifest.xml:** Contains essential information about the app's components, permissions, and configurations.
    - **smali/**: A directory that contains the Smali code, which is the assembly language representation of the app's code.
    - **res/**: Contains resources such as images, layout files, and strings used in the application.
    - **lib/**: Holds the native libraries used by the application, if any.

7. Open Important Files:
    - Use a text editor to open and review AndroidManifest.xml to understand the application's structure and permissions: **"nano AndroidManifest.xml "**

9. Conclusion

In summary, the process of decompiling an APK file using tools like APKTool and JADX is an invaluable skill for analyzing Android applications. This task not only allows for the extraction of source code and resources but also provides deeper insights into the application's structure and functionality. By following the outlined steps, users can effectively navigate the decompilation process, enabling them to explore key components such as the AndroidManifest.xml, resource files, and Smali code.

Understanding these elements is crucial for security assessments, app development, and troubleshooting. The ability to reverse engineer applications fosters a better grasp of programming practices and potential vulnerabilities, promoting enhanced application security. Ultimately, the knowledge gained through this process equips developers and security professionals with the tools necessary to improve application integrity and safeguard user data.

10. Appendix

Task 3: Decompile an APK File

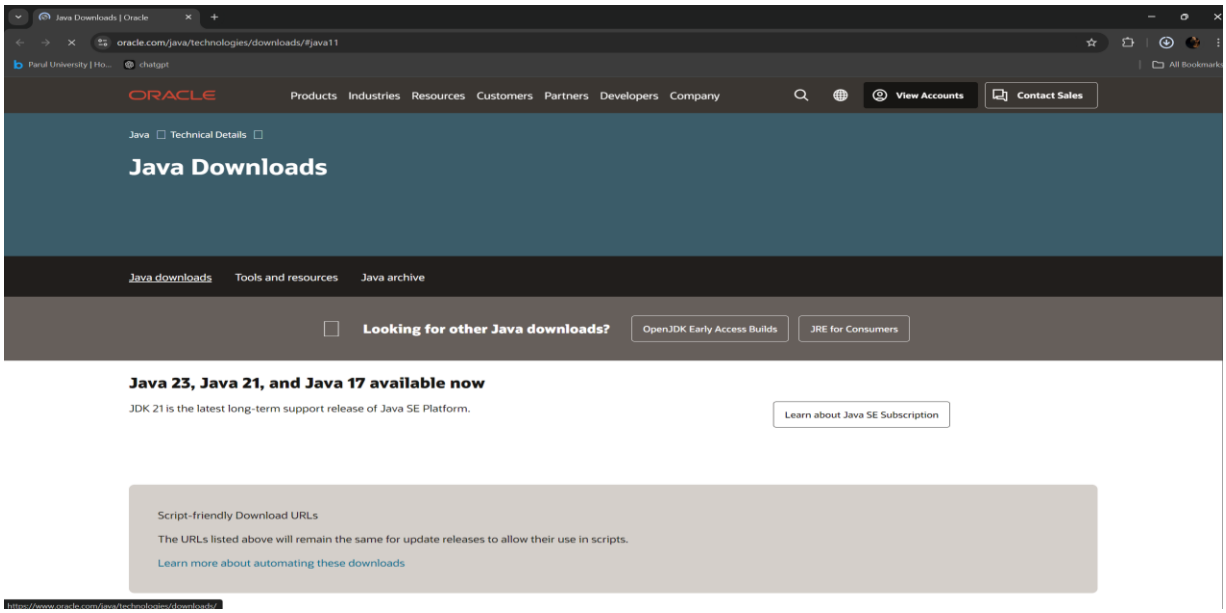

Figure 1: Downloading the JDK Tool.

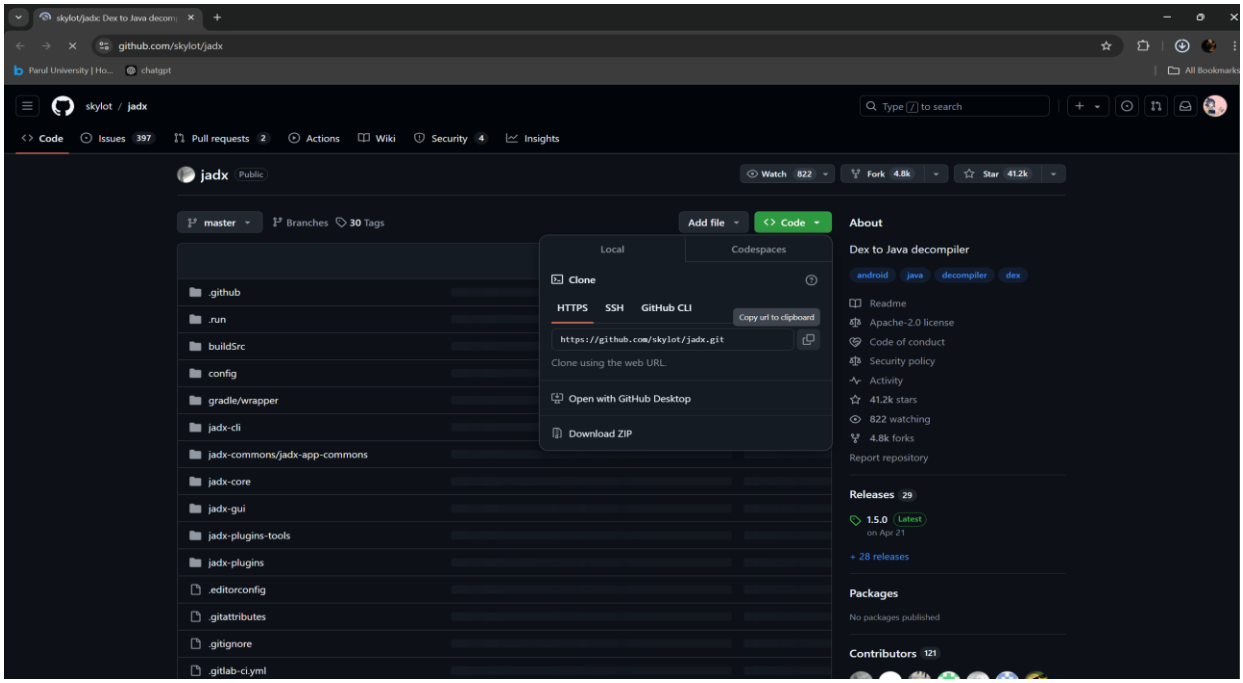

Figure 3: Downloading the jadx tool.

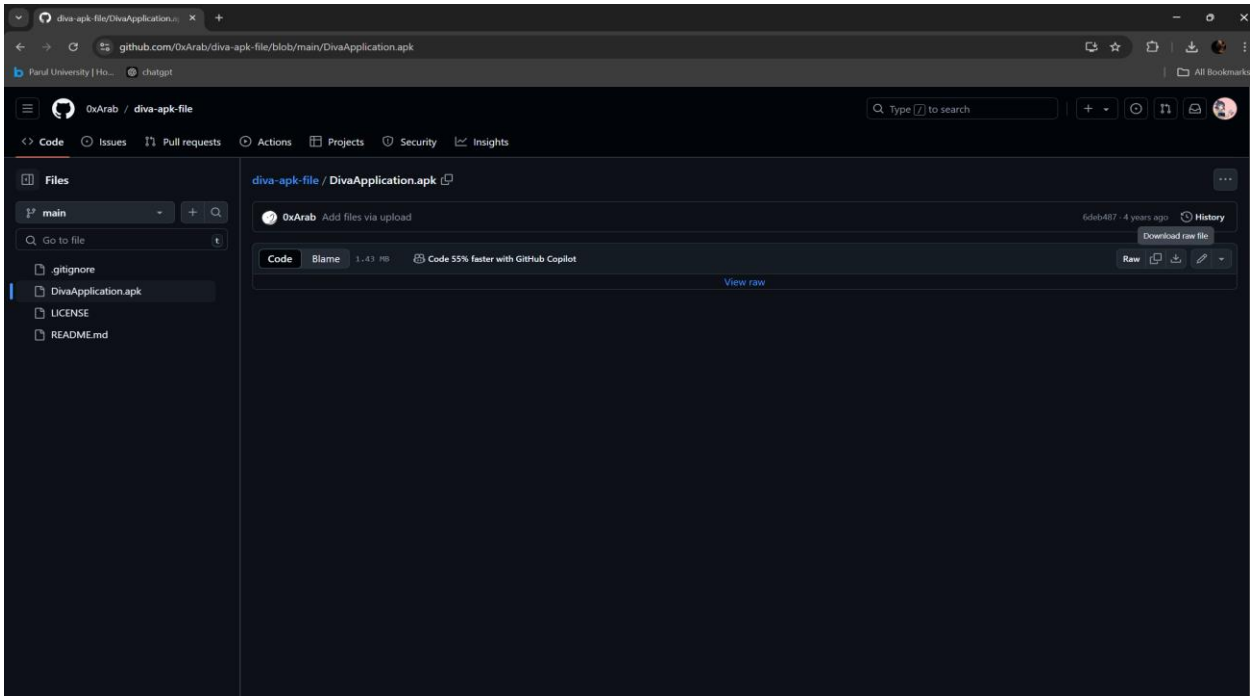

Figure 2: Downloading the DivaApplication.apk

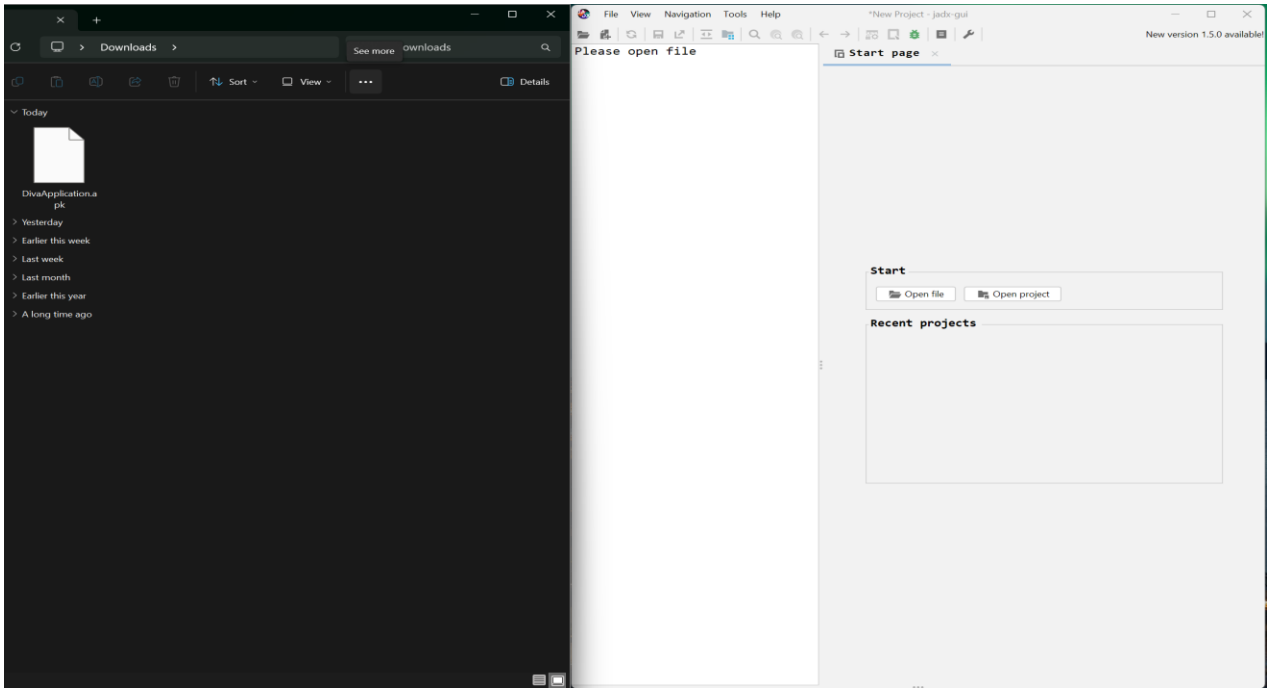

Figure 4: Drag and Drop the Apk file into the Jadx

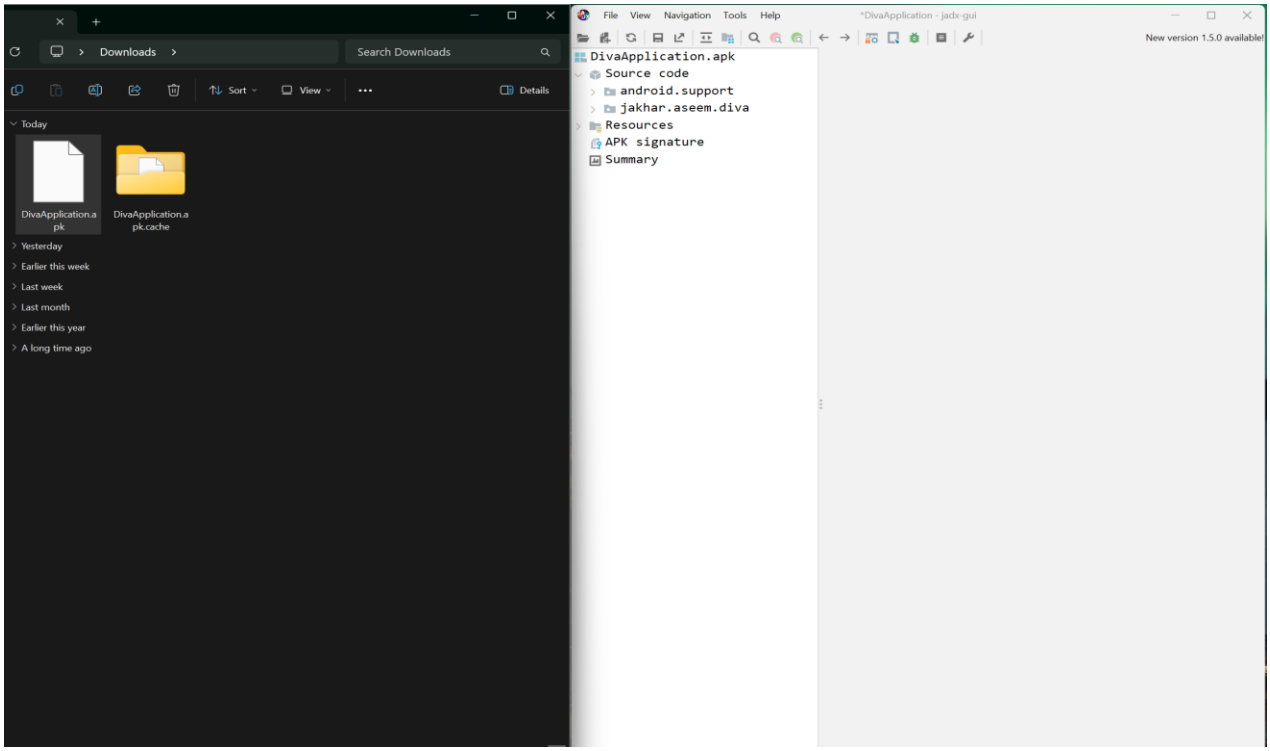

Figure 5: After Dropping the DivaApplication.apk into Jadx-GUI

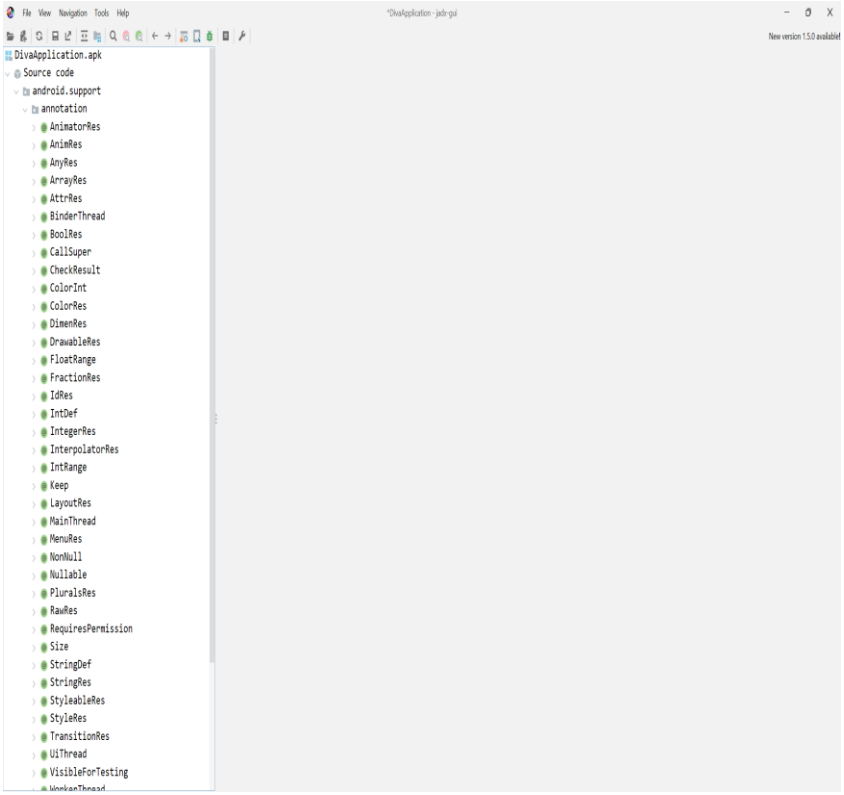

Figure 7:After the Decompilation of Apk files.

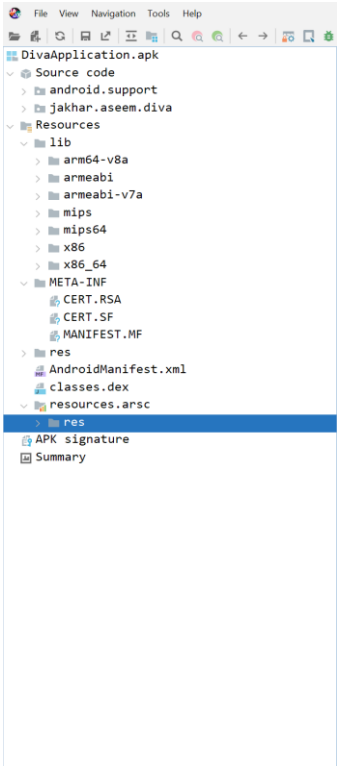

Figure 6: The structure and the files inside each directory.

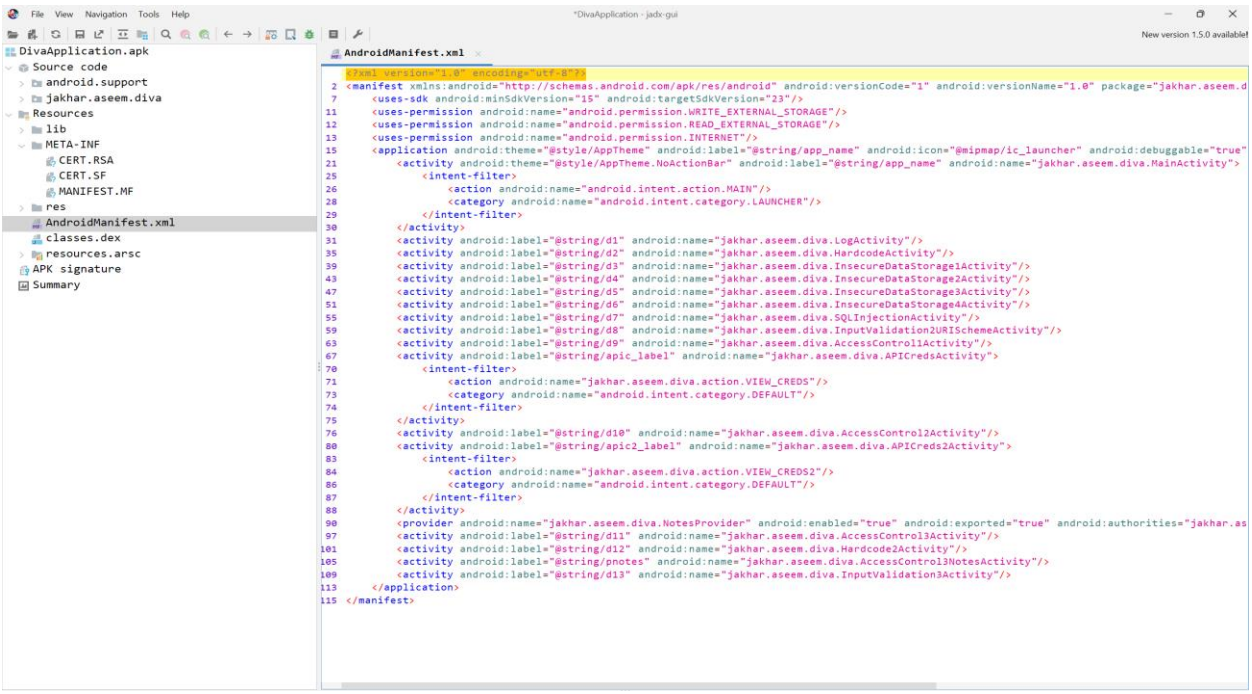

Figure 8: The AndroidManifest.xml file

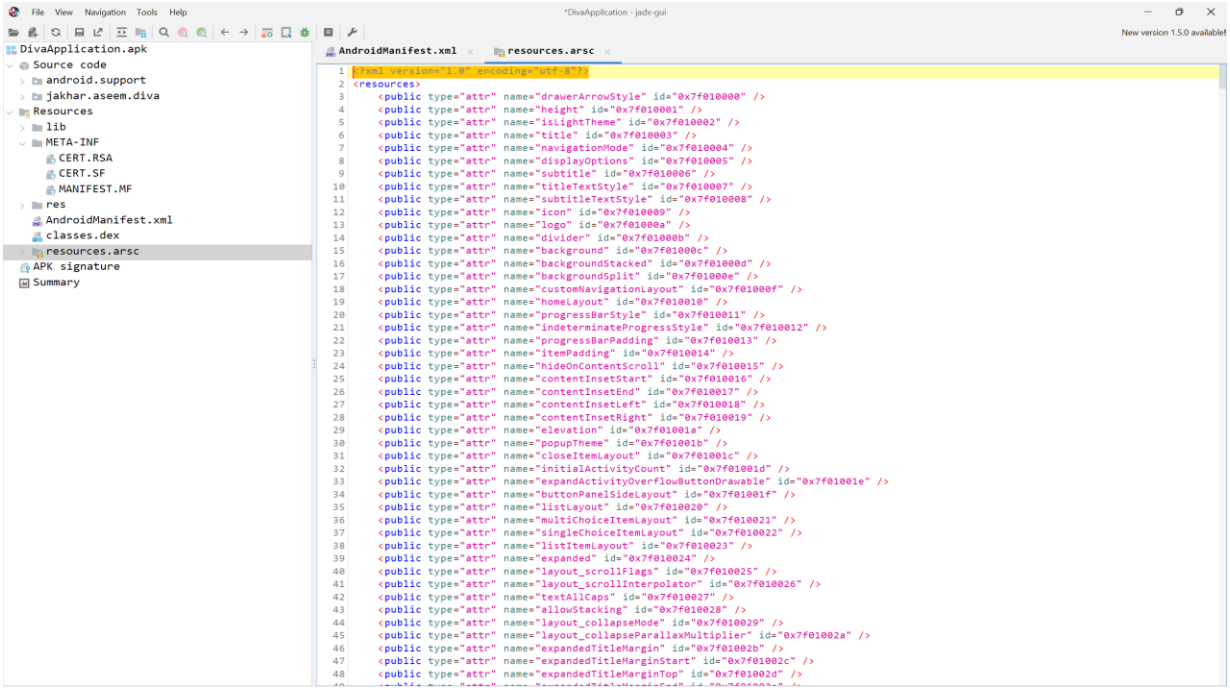

Figure 9: The resources.arsc file

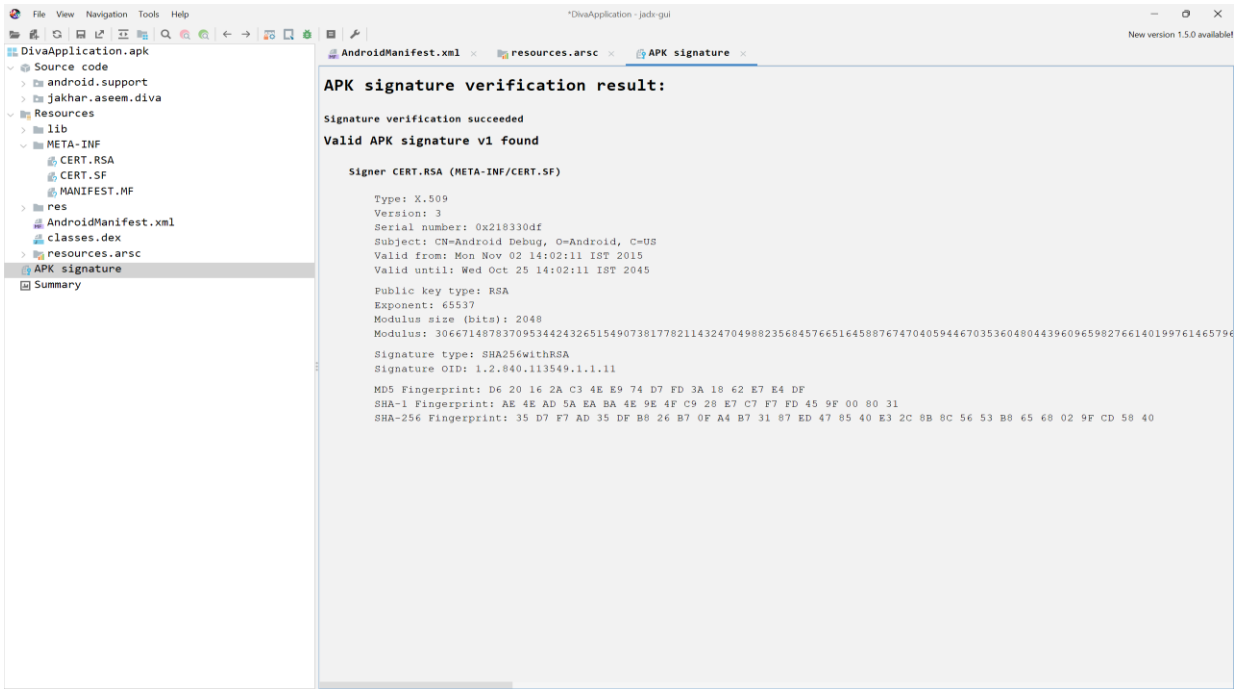

Figure 10: APK signature file

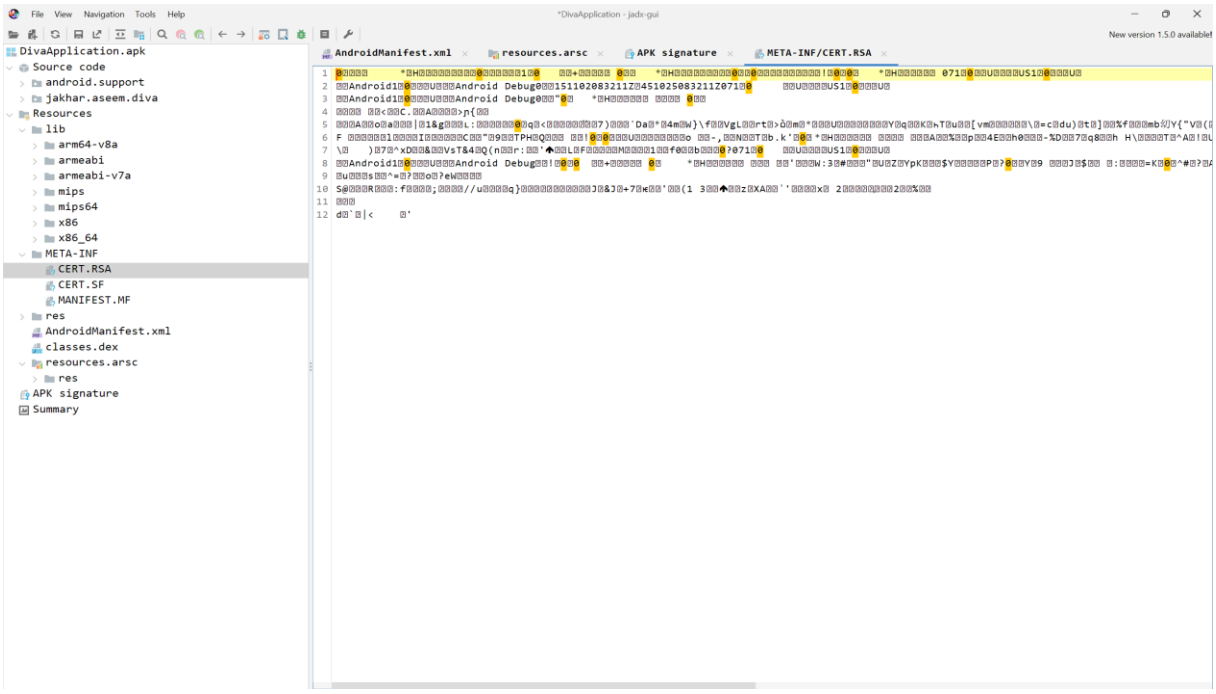

Figure 11: META-INF/CERT.RSA file

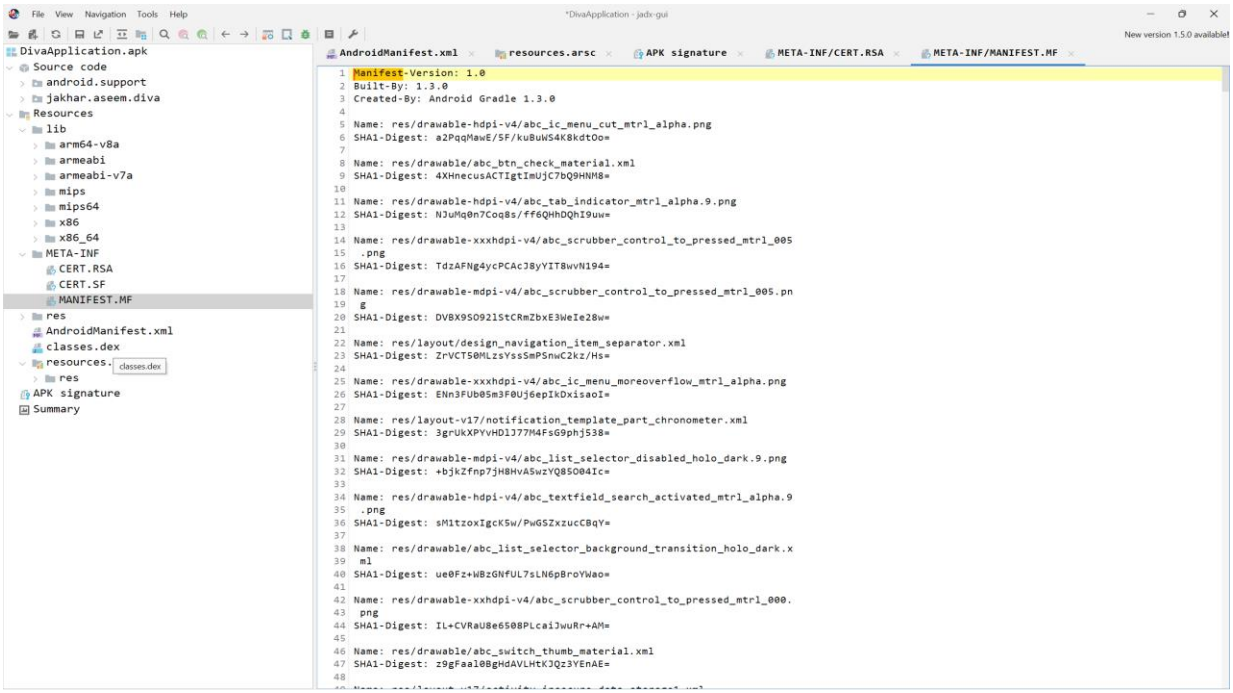

Figure 12: META-INF/MANIFEST.MF file

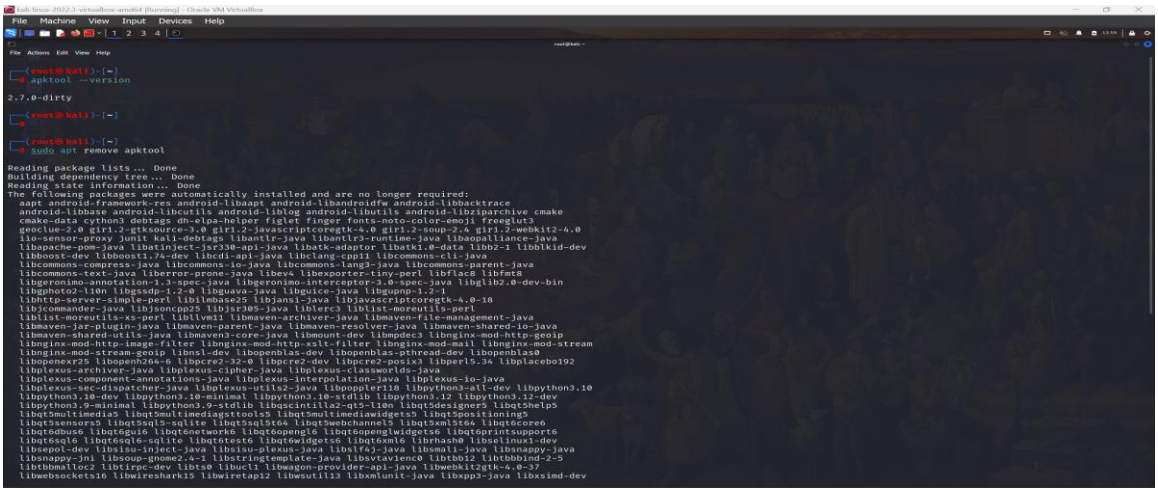

Figure 13: Checking the APKTool Version and removing the old version.

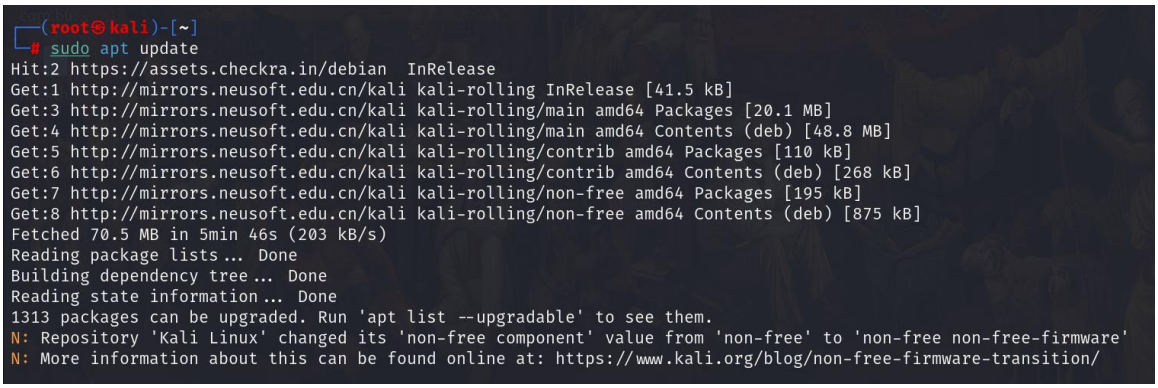

Figure 14: sudo apt update

Figure 15: sudo apt install apktool

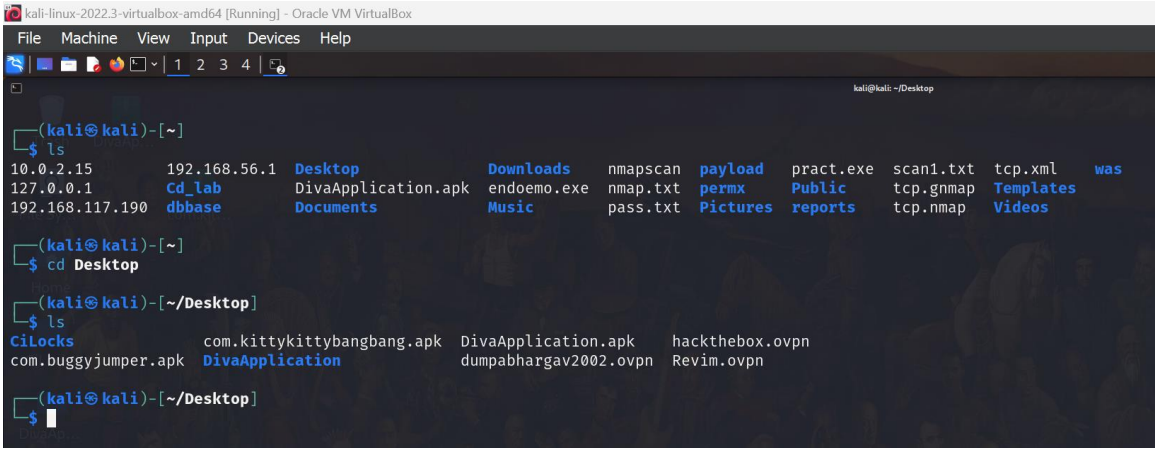

Figure 16: Checking the Apk File location by using general linux commands.

Figure 17: mv ~/Desktop/DivaApplication.apk ~/

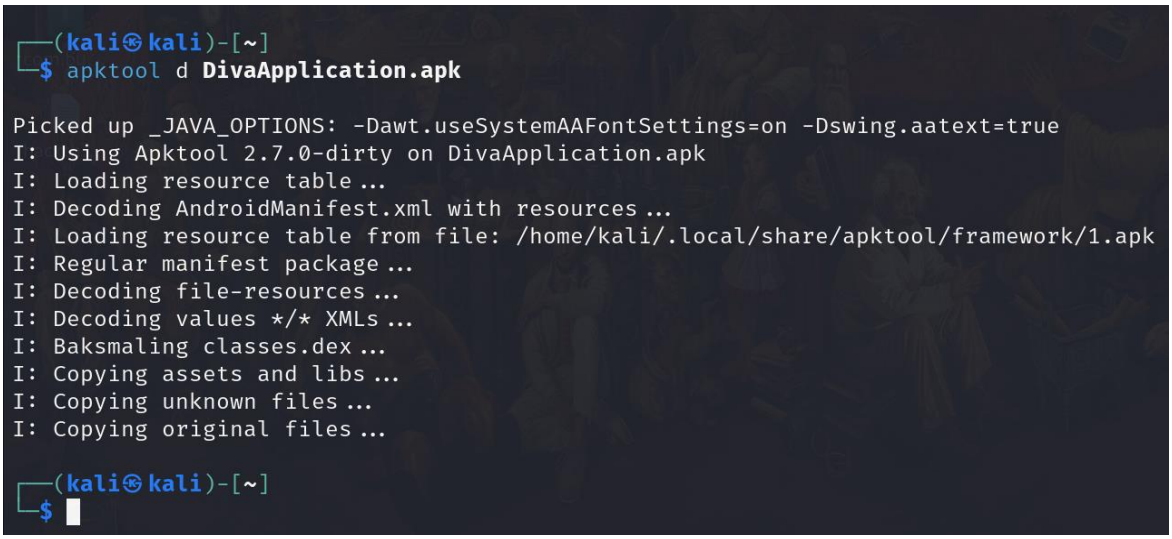

Figure 18: apktool d DivaApplication.apk

```
┌──(kali㉿kali)-[~]
└─$ cd DivaApplication


┌──(kali㉿kali)-[~/DivaApplication]
└─$ ls
AndroidManifest.xml  apktool.yml  lib  original  res  smali

┌──(kali㉿kali)-[~/DivaApplication]
└─$
```

Figure 19: Checking the Files inside the Diva application after decompilation

```
┌──(kali㉿kali)-[~/DivaApplication]
└─$ nano AndroidManifest.xml
```

Figure 20: To open the file in the text editor

```
kali-linux-2022.3-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File  Machine  View  Input  Devices  Help
kali@kali: ~/DivaApplication
  GNU nano 8.0                                  AndroidManifest.xml
<?xml version="1.0" encoding="utf-8" standalone="no"?><manifest xmlns:android="http://schemas.android.com/apk/res/android" package="jakhar.aseem.diva" platformBuildVer
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <application android:allowBackup="true" android:debuggable="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name" android:supportsRtl="true" ar
        <activity android:label="@string/app_name" android:name="jakhar.aseem.diva.MainActivity" android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <activity android:label="@string/d1" android:name="jakhar.aseem.diva.LogActivity"/>
        <activity android:label="@string/d2" android:name="jakhar.aseem.diva.HardcodeActivity"/>
        <activity android:label="@string/d3" android:name="jakhar.aseem.diva.InsecureDataStorage1Activity"/>
        <activity android:label="@string/d4" android:name="jakhar.aseem.diva.InsecureDataStorage2Activity"/>
        <activity android:label="@string/d5" android:name="jakhar.aseem.diva.InsecureDataStorage3Activity"/>
        <activity android:label="@string/d6" android:name="jakhar.aseem.diva.InsecureDataStorage4Activity"/>
        <activity android:label="@string/d7" android:name="jakhar.aseem.diva.SQLInjectionActivity"/>
        <activity android:label="@string/d8" android:name="jakhar.aseem.diva.InputValidation2URISchemeActivity"/>
        <activity android:label="@string/d9" android:name="jakhar.aseem.diva.AccessControl1Activity"/>
        <activity android:label="@string/apic_label" android:name="jakhar.aseem.diva.APICredsActivity">
            <intent-filter>
                <action android:name="jakhar.aseem.diva.action.VIEW_CREDS"/>
                <category android:name="android.intent.category.DEFAULT"/>
            </intent-filter>
        </activity>
        <activity android:label="@string/d10" android:name="jakhar.aseem.diva.AccessControl2Activity"/>
        <activity android:label="@string/apic2_label" android:name="jakhar.aseem.diva.APICreds2Activity">
            <intent-filter>
                <action android:name="jakhar.aseem.diva.action.VIEW_CREDS2"/>
                <category android:name="android.intent.category.DEFAULT"/>
            </intent-filter>
        </activity>
        <provider android:authorities="jakhar.aseem.diva.provider.notesprovider" android:enabled="true" android:exported="true" android:name="jakhar.aseem.diva.NotesPr
        <activity android:label="@string/d11" android:name="jakhar.aseem.diva.AccessControl3Activity"/>
        <activity android:label="@string/d12" android:name="jakhar.aseem.diva.Hardcode2Activity"/>
        <activity android:label="@string/pnotes" android:name="jakhar.aseem.diva.AccessControl3NotesActivity"/>
        <activity android:label="@string/d13" android:name="jakhar.aseem.diva.InputValidation3Activity"/>
                                                [ Read 40 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-B Previous
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^B Where Was    M-F Next
                                                                                                                Right Ctrl
```

Figure 21: AndroidManifest.xml file in text editor

```
┌──(kali㉿kali)-[~/DivaApplication]
└─$ cd smali

┌──(kali㉿kali)-[~/DivaApplication/smali]
└─$ ls
android  jakhar

┌──(kali㉿kali)-[~/DivaApplication/smali]
└─$ cd android

┌──(kali㉿kali)-[~/DivaApplication/smali/android]
└─$ ls
support

┌──(kali㉿kali)-[~/DivaApplication/smali/android]
└─$ cd ..

┌──(kali㉿kali)-[~/DivaApplication/smali]
└─$ cd jakhar

┌──(kali㉿kali)-[~/DivaApplication/smali/jakhar]
└─$ ls
aseem

┌──(kali㉿kali)-[~/DivaApplication/smali/jakhar]
└─$
```

Figure 22: Checking the contents inside the smali directory and further.

```
┌──(kali㉿kali)-[~/DivaApplication]
└─$ ls
AndroidManifest.xml  apktool.yml  lib  original  res  smali

┌──(kali㉿kali)-[~/DivaApplication]
└─$ cd lib

┌──(kali㉿kali)-[~/DivaApplication/lib]
└─$ ls
arm64-v8a  armeabi  armeabi-v7a  mips  mips64  x86  x86_64

┌──(kali㉿kali)-[~/DivaApplication/lib]
└─$
```

Figure 23: Checking the lib directory and its contents

```
┌──(kali㉿kali)-[~/DivaApplication]
└─$ ls
AndroidManifest.xml  apktool.yml  lib  original  res  smali

┌──(kali㉿kali)-[~/DivaApplication]
└─$ cd res

┌──(kali㉿kali)-[~/DivaApplication/res]
└─$ ls
anim                        layout              values-ca       values-gu-rIN       values-ko          values-nl           values-ta-rIN  values-w360dp-v13
color                       layout-sw600dp-v13  values-cs       values-h320dp-v13   values-ky-rKG      values-pa-rIN       values-te-rIN  values-w480dp-v13
color-v11                   layout-v17          values-da       values-h720dp-v13   values-land        values-pl           values-th      values-w500dp-v13
color-v23                   layout-v21          values-de       values-hdpi-v4      values-large-v4    values-port         values-tl      values-w600dp-v13
drawable                    menu                values-el       values-hi           values-ldltr-v21   values-pt           values-tr      values-w720dp-v13
drawable-hdpi-v4            mipmap-hdpi-v4      values-en-rAU   values-hr           values-ldrtl-v23   values-pt-rBR       values-uk      values-w820dp-v13
drawable-ldrtl-hdpi-v4      mipmap-mdpi-v4      values-en-rGB   values-hu           values-lo-rLA      values-pt-rPT       values-ur-rPK  values-xlarge-land-v4
drawable-ldrtl-mdpi-v4      mipmap-xhdpi-v4     values-en-rIN   values-hy-rAM       values-lt          values-ro           values-uz-rUZ  values-xlarge-v4
drawable-ldrtl-xhdpi-v4     mipmap-xxhdpi-v4    values-es       values-in           values-lv          values-ru           values-v11     values-zh-rCN
drawable-ldrtl-xxhdpi-v4    mipmap-xxxhdpi-v4   values-es-rUS   values-is-rIS       values-mk-rMK      values-si-rLK       values-v12     values-zh-rHK
drawable-ldrtl-xxxhdpi-v4   values              values-et-rEE   values-it           values-ml-rIN      values-sk           values-v14     values-zh-rTW
drawable-mdpi-v4            values-af           values-eu-rES   values-iw           values-mn-rMN      values-sl           values-v17     values-zu
drawable-v21                values-am           values-fa       values-ja           values-mr-rIN      values-sq-rAL       values-v18
drawable-v23                values-ar           values-fi       values-ka-rGE       values-ms-rMY      values-sr           values-v21
drawable-xhdpi-v4           values-az-rAZ       values-fr       values-kk-rKZ       values-my-rMM      values-sv           values-v22
drawable-xxhdpi-v4          values-bg           values-fr-rCA   values-km-rKH       values-nb          values-sw           values-v23
drawable-xxxhdpi-v4         values-bn-rBD       values-gl-rES   values-kn-rIN       values-ne-rNP      values-sw600dp-v13  values-vi

┌──(kali㉿kali)-[~/DivaApplication/res]
└─$
```

Figure 24: Checking the res Directory and its contents.

```
┌──(kali㉿kali)-[~/DivaApplication]
└─$ ls
AndroidManifest.xml  apktool.yml  lib  original  res  smali

┌──(kali㉿kali)-[~/DivaApplication]
└─$ nano aptool.yml
```

Figure 25: Opening the aptool.yml in text editor. But the File is empty nothing is written inside the file.