

Exported Activities

In this section, we'll explore another important element of dynamic analysis: exported activities in Android applications. By default, activities are only accessible within their app. However, when the attribute `exported` of an activity is set to `true`, then it can be invoked by other applications or components. Even though this is a useful feature, it can expose the app to security risks. Exported activities that lack adequate security can often become the target of malicious `intents`, eventually leading to unauthorized access. The following example demonstrates how to identify and exploit applications with insecure exported activities. Recognizing and testing these vulnerabilities is essential for a thorough security assessment.

Exploiting Insecure Exported Activities

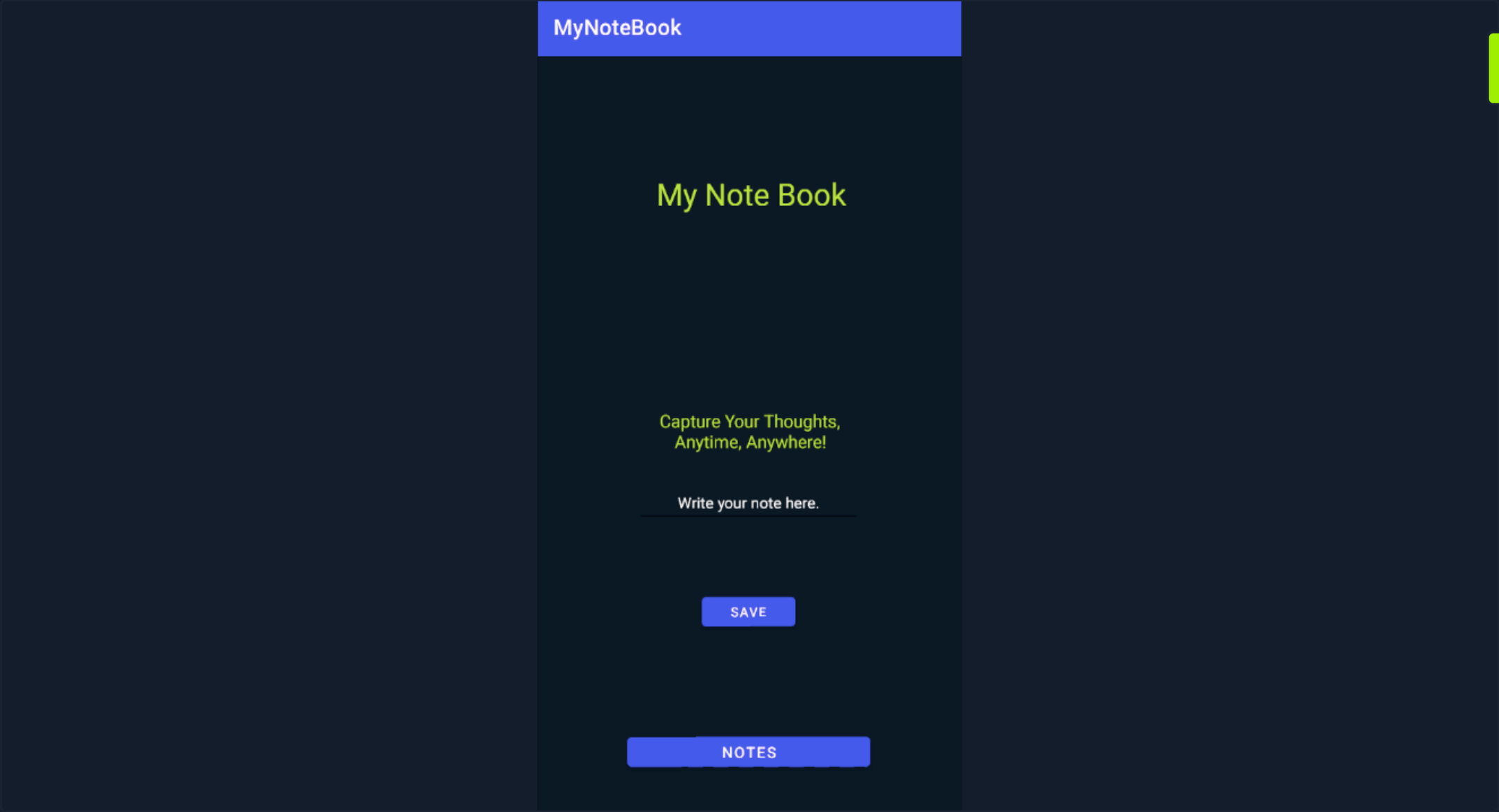
In this example, we will primarily use an Android Virtual Device (AVD), though the process is compatible with any other Android device, physical or emulated. Let's connect to the device via ADB and install the application.

Exported Activities

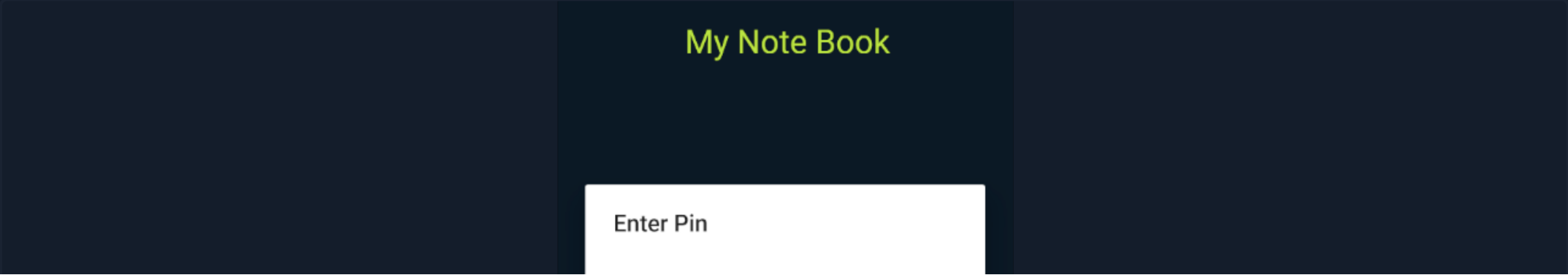
```
r11k@htb[/htb]$ adb connect
r11k@htb[/htb]$ adb install myapp.apk

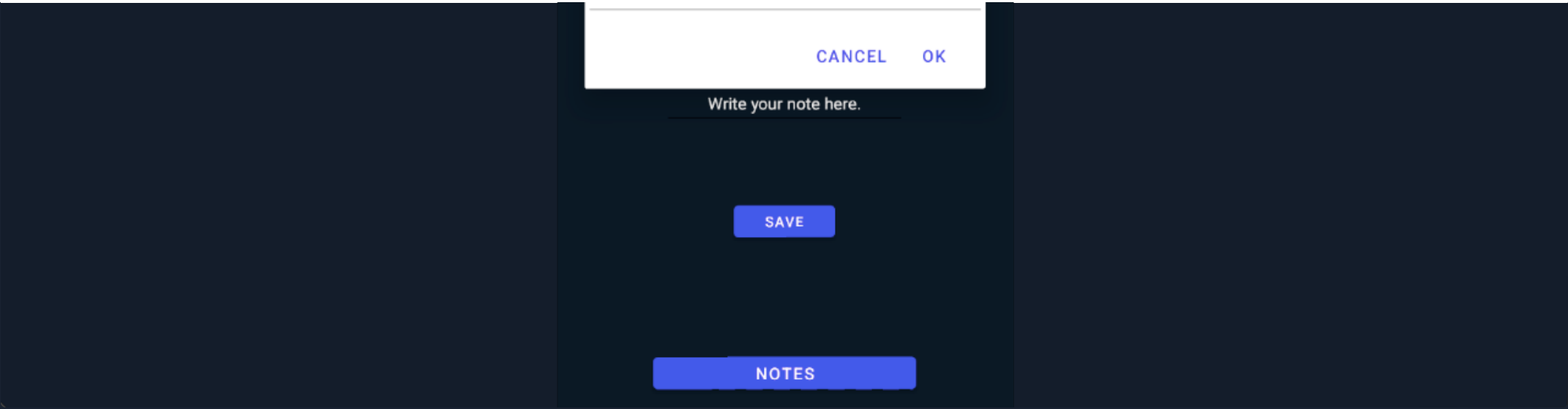
Performing Streamed Install
Success
```

We can now run the application on the device.



This note-taking application allows users to save notes on their devices.





When the user taps the **NOTES** button to access saved notes, the app prompts for a PIN. To continue enumerating the application, we can retrieve its package name by running the following command while the app is active.

Exported Activities

```
rl1k@htb[/htb]$ adb root
rl1k@htb[/htb]$ adb shell
rl1k@htb[/htb]$ adb shell dumpsys activity activities | grep VisibleActivityProcess

VisibleActivityProcess: [ ProcessRecord{8f86b89 6255:com.hackthebox.myapp/u0a120}]
```

Enumeration of the app-specific external storage reveals the following files.

Exported Activities

```
rl1k@htb[/htb]$ adb shell ls -l /sdcard/Android/data/com.hackthebox.myapp/files/MyPersonalNotes/

total 12
-rw-rw---- 1 u0_a120 ext_data_rw 230 2024-01-14 11:42 Note_309943672.txt
-rw-rw---- 1 u0_a120 ext_data_rw 157 2024-01-14 11:42 Note_311366182.txt
-rw-rw---- 1 u0_a120 ext_data_rw  45 2024-01-14 11:42 Note_347233492.txt
```

The content of the notes, however, is encrypted.

Exported Activities

```
rl1k@htb[/htb]$ adb shell cat /sdcard/Android/data/com.hackthebox.myapp/files/MyPersonalNotes/Note_347233492.txt

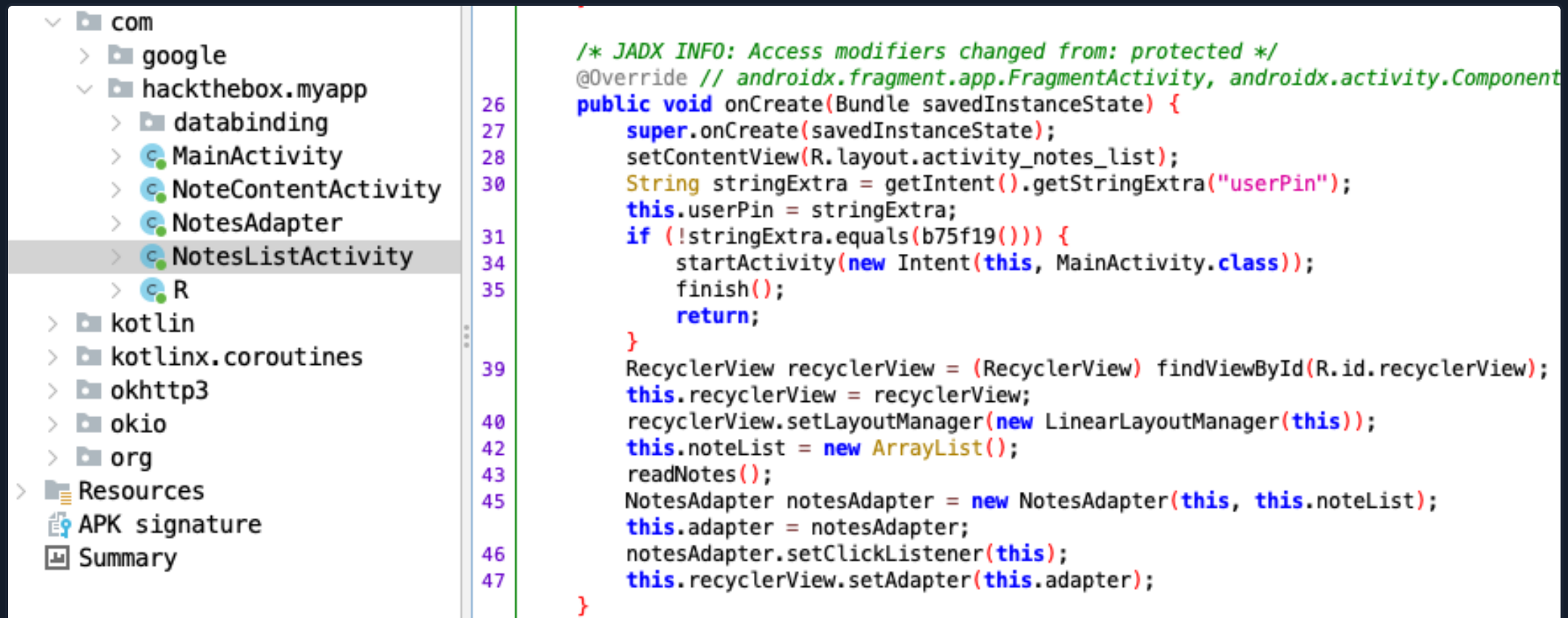
8TdgmUd0Hv4IBm/7F90ltLJzEUFcoc+QgHXTqPpHjic=
```

Let's use JADX to read the app's source code. Reading the code in **MainActivity** confirms that the notes are saved in the app-specific external directory of the device.



Reading the content of the **NoteListActivity**, the activity that lists the notes on the screen, we can see that the user's PIN is checked again before

listing the content. This additional check protects the activity's content from unauthorized access by users or apps attempting to bypass the PIN prompt.



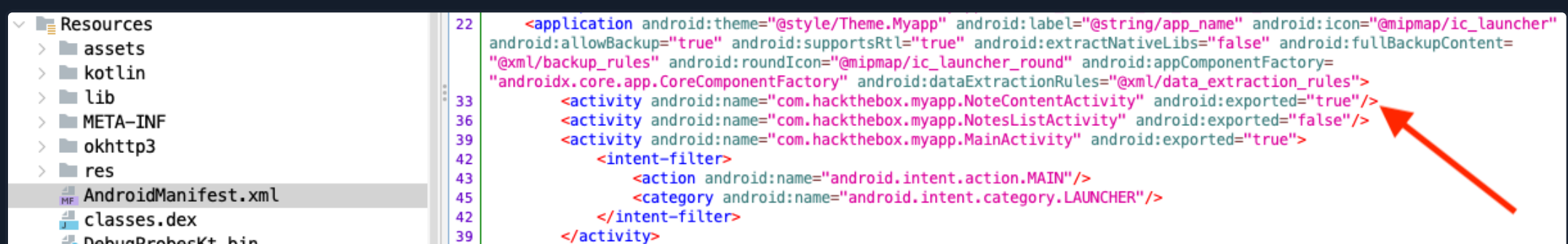
However, the `NoteContentActivity` behaves differently in this regard. In this activity, we notice no checking functionality for the user's PIN, and the `readNoteContent()` method that lists the content of the files is called directly within the `onCreate()` method, with the only requirement being the parameter `getIntent().getStringExtra("filename")`, which is most likely the name of the file containing the notes.



This means that we might be able to bypass the PIN check of the application's main screen and start the activity directly using a custom app or ADB. Accessing activities in this manner also requires the `exported` attribute in the app's `AndroidManifest.xml` file to be set to `true`. The following line confirms this.

Code: `xml`

```
<activity android:name="com.hackthebox.myapp.NoteContentActivity" android:exported="true"/>
```




Earlier, during storage enumeration, we discovered the names of the note files. Now, we can attempt to access the `NoteContentActivity` directly using ADB. The command below launches the activity and passes the filename `Note_347233492.txt` as a parameter.



MyNoteBook

HTB{[REDACTED]}

This successfully bypasses the PIN check, granting direct access to the contents of the note.



Connect to Pwnbox

Your own web-based Parrot Linux instance to play our labs.

Pwnbox Location

UK

31ms

Terminate Pwnbox to switch location

Start Instance

/ 1 spawns left

Waiting to start...

Enable step-by-step solutions for all questions

Questions

Cheat Sheet

Answer the question(s) below to complete this Section and earn cubes!

+ 5

What is the content of the file "Note_347233492.txt" ?

Submit your answer here...

+10 Streak pts

Submit

exported_activities.zip

Previous

Next

Cheat Sheet

Go to Questions

Table of Contents

Enumerating and Exploiting Installed Apps

- Introduction
- Enumerating Local Storage
- Exported Activities
- Insecure Logging
- Pending Intents
- Exploiting WebViews
- Insecure Library Load Through Deep Linking

Dynamic Code Instrumentation

- Hooking Java Methods
- Altering Method Values
- Hooking Native Methods
- Bypassing Detection Mechanisms
- Authentication Token Manipulation

Intercepting HTTP/HTTPS Requests

- Intercepting API Calls
- IDOR Attack
- SSL/TLS Certificate Pinning Bypass

Skills Assessments

- Skills Assessment

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left