**Write-up: Reverse Engineering Android APK to Find the Flag**

**Overview**

In this challenge, we were given an **Android APK file**, and our goal was to **extract and analyze** its contents to find the hidden **flag**. The process involved **decompiling the APK, extracting DEX files, converting them to Java source code, and searching for the flag in the decompiled files**.

**Steps Taken**

**Step 1: Extracting APK Contents**

The first step was to **unzip** the APK file to extract its contents. Since an APK is essentially a **ZIP archive**, we used JADX to extract the files:

bash

Sao chépChỉnh sửa

jadx -d jadx\_output app.apk

This extracted the following key files:

* classes.dex (Dalvik Executable – contains the compiled bytecode)
* AndroidManifest.xml (Application metadata)
* res/ (Resources like images, layouts, etc.)

**Step 2: Decompiling .dex to Java Source Code**

Since the classes.dex file contains the actual **compiled application logic**, we needed to **convert it back to Java source code**. We used JADX again for this:

bash

Sao chépChỉnh sửa

jadx -d output/sources classes.dex

After this, the output/sources/ directory contained the **decompiled Java source code**.

**Step 3: Searching for the Flag**

Once we had the **decompiled source code**, the next step was to **search for possible flag patterns**. Since most CTF flags follow a format like CTF{...} or picoCTF{...}, we used grep to search recursively through all files:

bash

Sao chépChỉnh sửa

grep -Ri "coctf" .

This command searches **all files** inside output/sources/, ignoring case (-i) and recursively checking subdirectories (-R).

**Flag Found:**

The command returned the following result:

bash

Sao chépChỉnh sửa

com/example/timer/BuildConfig.java: public static final String VERSION\_NAME = "picoCTF{t1m3r\_r3v3rs3d\_succ355fully\_17496}";

The flag was found inside BuildConfig.java, which is often used to store **version info and other metadata**.

**Conclusion**

**Final Flag:**

Sao chépChỉnh sửa

picoCTF{t1m3r\_r3v3rs3d\_succ355fully\_17496}

**Key Takeaways:**

1. **JADX** is a powerful tool to extract and decompile Android APK files.
2. **DEX files** contain the compiled bytecode and need to be **converted to Java source** to be analyzed.
3. **Searching for flag patterns** using grep -Ri is an efficient way to quickly locate the flag.
4. **BuildConfig.java** often contains useful metadata that might leak sensitive information like flags.