Let’s open the AndroidManifest.xml file from unzipped APK →
Okay so we need to decompile the APK properly so let’s decompile it using APKTOOL

YAY 😊 let’s see what’s in there after decompiling
Now let’s modify the AndroidManifest.xml file and set the debuggable option to True so we can later hook this app into a debugger 😊
You can also convert an APK file directly into a JAR format using dex2jar →

Once done, we can then open this .jar file in JD – GUI →
We are now ready to recompile our app back to .APK file. We will use APKTOOL again to achieve this.

```
root@Black-Box:~:/Downloads/OWASP# apktool b base
I: Using Apktool 2.3.4-dirty
I: Checking whether sources has changed...
I: Checking whether resources has changed...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk...
root@Black-Box:~:/Downloads/OWASP# cd base/dist
root@Black-Box:~:/Downloads/OWASP/base/dist# ls -al
 total 78844
 drwxr-xr-x 9 root root 4096 Feb 21 16:21 ..
-rw-r--r-- 1 root root 80727908 Feb 21 16:21 base.apk
root@Black-Box:~:/Downloads/OWASP/base/dist#
```
And now it is time to sign the newly compiled APK file with the Android test certificate. I am using APK Sign for this. [https://github.com/appium/sign](https://github.com/appium/sign)
We will use Zipalign (part of Android SDK build-tools) to optimise our newly signed APK. You can read more about Zipalign here: https://developer.android.com/studio/command-line/zipalign

Let’s push this to mobile now →
And we successfully managed to change the name of the app and installed it on the phone 😊
How can I make my App secure?
- Use code obfuscator like ProGuard which is a cross platform tool written in Java.

- Developers can also obfuscate code manually. Code obfuscation may include:
  - encrypting some or all of the code;
  - stripping out potentially revealing metadata
  - renaming useful class and variable names to meaningless labels;
  - adding unused or meaningless code to an application’s binary.
- Do Not store any sensitive information in the application source code
- Move important code chunks to the server side
- Do Not store user credentials at the device level and rather store a short lived encrypted token
- Will highly recommend not to store API keys in the code and make use of public/private key exchange to protect your API
- Do Not store app data to external storage since it can be read by all of the applications.
- Enable Anti-Tamper and Anti-Debug by injecting a self-protection code into the application source code. E.g. Random crashes and limited functionality upon tampering detection.
Further reading:

OWASP Mobile Top 10

https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

https://hackernoon.com/mobile-application-security-best-practices-for-app-developers-1a6345750b35

https://www.multidots.com/how-to-prevent-reverse-engineering-of-your-mobile-apps/

https://www.preemptive.com/obfuscation

Thank You