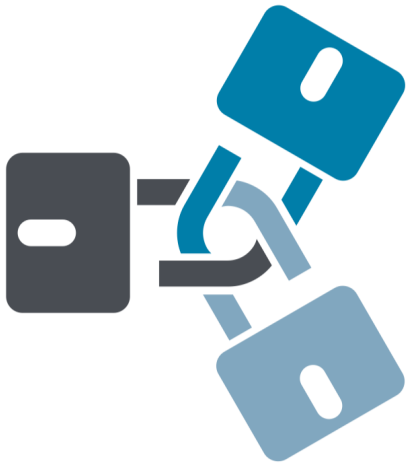


Cyber Security Team
FH Campus Wien

CCST Introduction into APK Reversing

Silvie Schmidt elvis@fh-campuswien.ac.at
Ines Kramer itsec-ctf@fh-campuswien.ac.at
Campus Cyber Security Team https://www.campus-cybersecurity.team/

Created by Ines Kramer, Network Lab 2020-09-01

Todays writeup: https://tinyurl.com/ccst09012020

Preliminaries

- Todays writeup: https://tinyurl.com/ccst09012020
- Please do all preliminary steps:
  - Clone the gitlab repo

Android System

- Hardened SELinux
  - ASLR
  - Canaries
  - NX memory
  - no non-PIE binaries
  - ,..
- Firewalled syscalls
- One user per app
- Apps are sandboxed
- Permission system

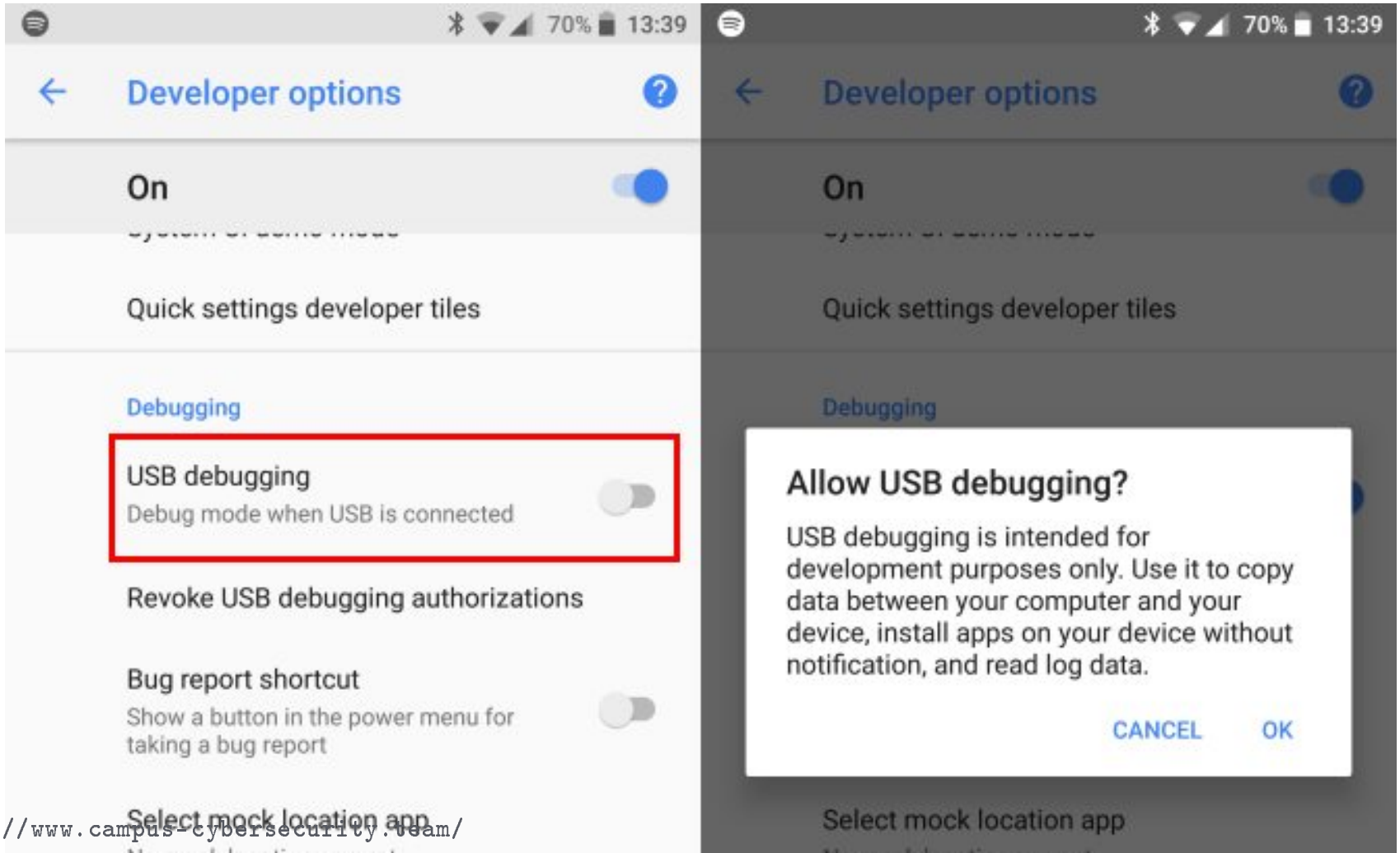
Android Runtime ART

- Java code but no JAVA VM
- Before Dalvik VM - JIT compiler
- ART: Native ELF generated at installation time
- same DEX bytecode

Android Package Format APK

- Installation via Play store, web-download or USB (adb)

Android Enable Debugging



Android Debug Bridge (adb)

Enable 'USB Debugging' in 'Developer Options' on the phone, install adb on the computer, connect USB cable

```
apt install adb
adb start-server
```

- Installing/Uninstalling .apk files via usb

```
adb install <apk_filename>
```

- Retrieve/transfer files/databases

```
adb pull/push <filename>
```

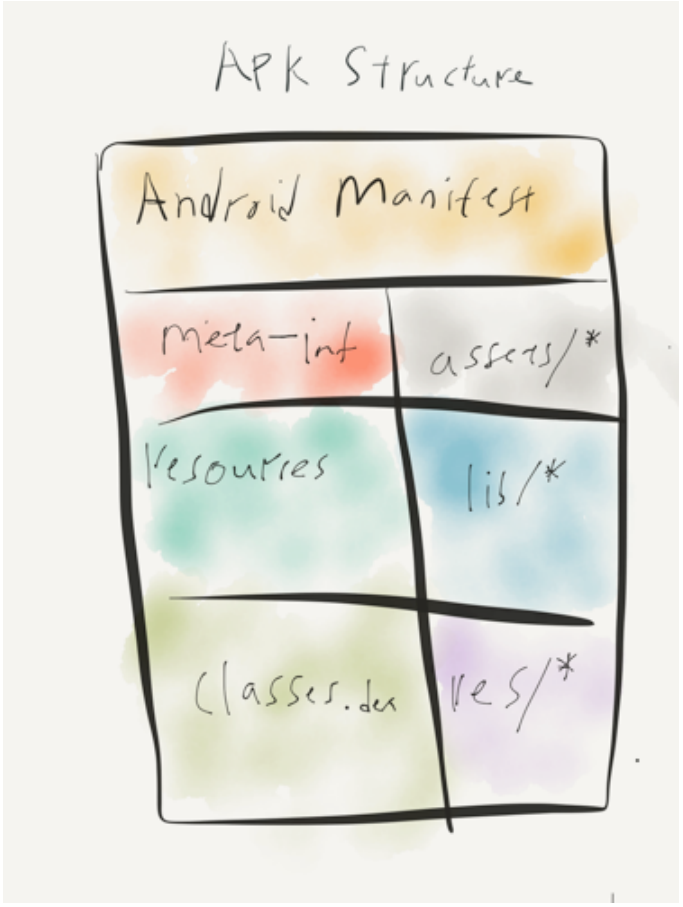
- Get active logs

```
adb logcat
```

Android Package Format APK

- Installation via Play store, web-download or USB (adb)
- ZIP file
  - unzip the .apk

```
cd ccst/apk_rev_intro/
   owasp_app
unzip  MSTG-Android-Java.apk
```



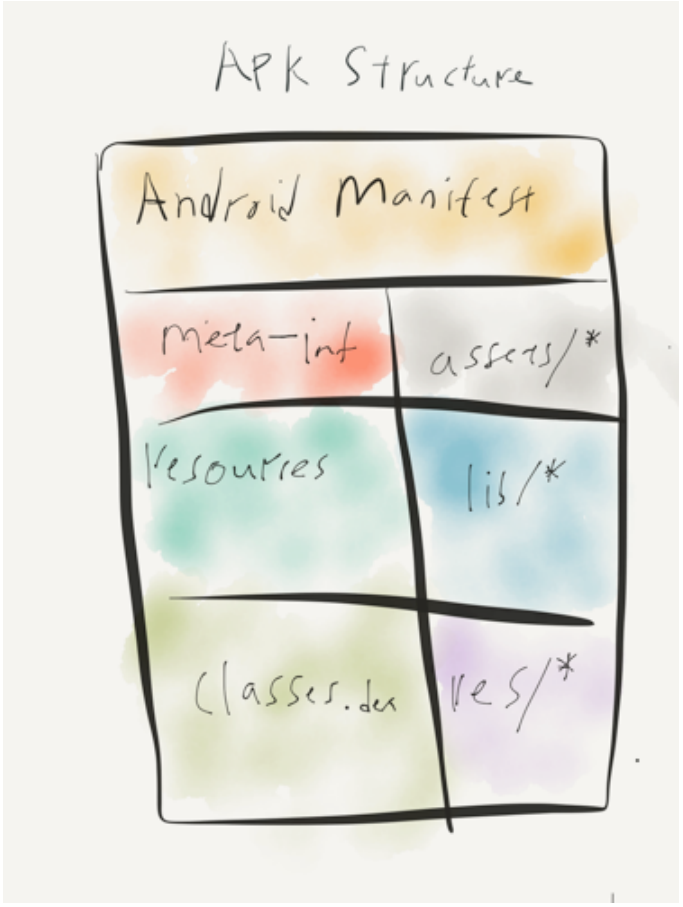
APK Content

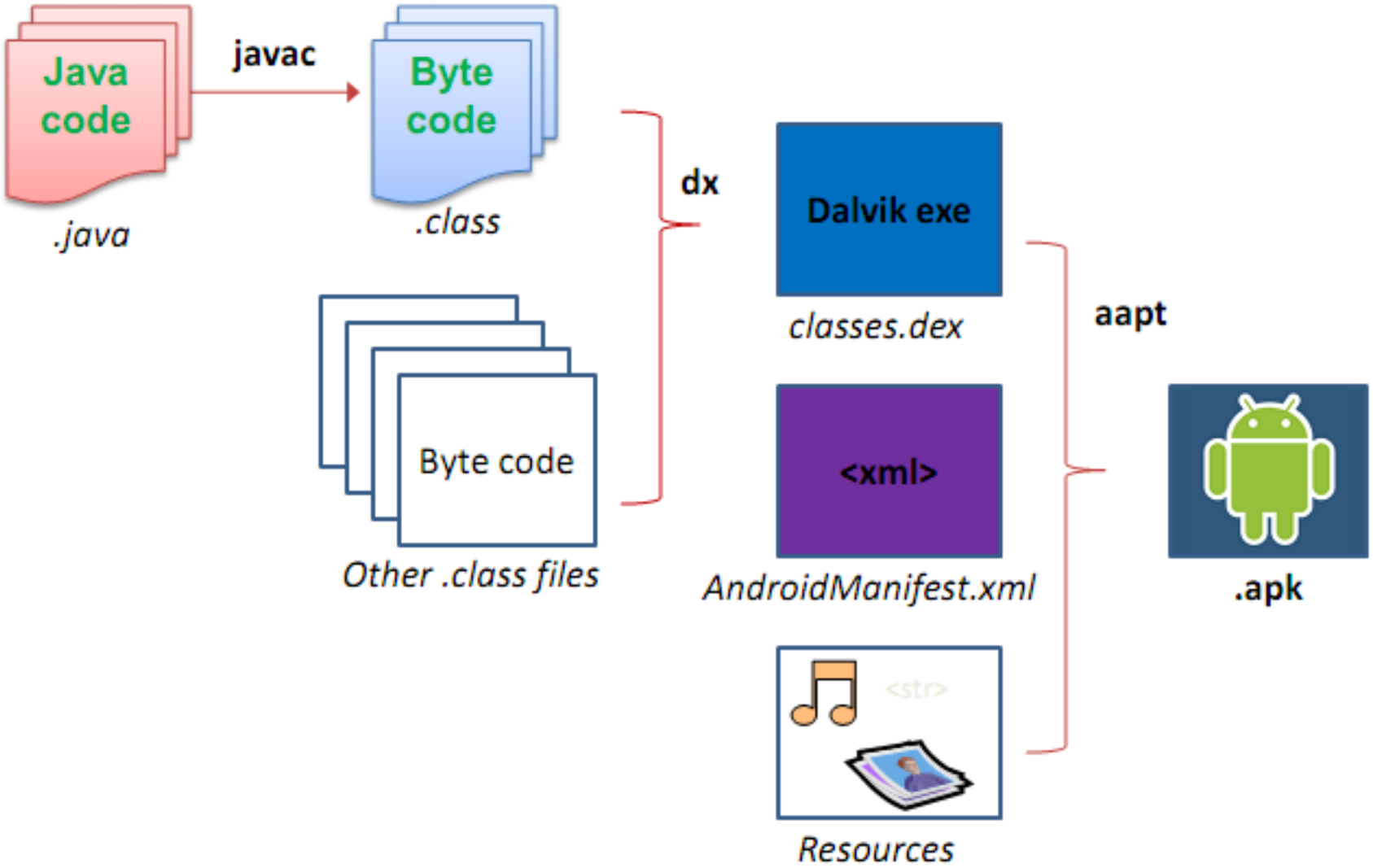
AndroidManifest.xml	the manifest file in binary XML format
classes.dex	application code compiled in dex format
resources.arsc	file containing pre-compiled application resources, in binary XML
res/*	resources not compiled into resources.arsc
assets/*	optional - applications assets for Asset-Manager
lib/*	optional folder containing compiled code - i.e. native code libraries.
META-INF/	
MANIFEST.MF	stores meta data SHA1 hashes about the contents of the JAR
CERT.SF	list of resources and SHA1 digest signed with RSA
CERT.RSA	developer RSA public key for signing CERT.SF

Android Package Format APK

- Installation via Play store, web-download or USB (adb)
- ZIP file
- APKs are signed by developer
  - check certificate

```
cd META-INF
openssl pkcs7 -in CERT.RSA
-inform DER -print
```




Java code
javac
Byte code
.java
.class
dx
Dalvik exe
classes.dex
Byte code
Other .class files
<xml>
AndroidManifest.xml
aapt
.apk
Resources

Decompile with APKTOOL

`https://ibotpeaches.github.io/Apktool/documentation/`

- Installation

```
sudo apt install apktool
```

- Decode with apktool

```
apktool d MSTG-Android-Java.apk
```

  - AndroidManifest XML file
  - classes.dex to SMALI format (like assembler)

Java App Components

- **Activity** - GUI component(screen) consists of views + code in java.class
- **Service** - background activity without screen
- **BroadCastReceiver** - used to receive inter-process-communication IPC messages
- **Binder** - used for remote procedure call

Java App Components

- **Intent** - pass messages between processes
  - consists of target, action and data
  - explicit and implicit intent - target explicit or chosen by os
  - intent receiver: BroadcastReceiveres, Services, Activities
  - adverties capabilities via IntentFilter (for implicit intents) in ApplicationManifest

Android Permissions

internet	INTERNET
SMS	RECEIVE_SMS
calendar	READ/WRITE_CALENDAR
log files	READ_LOGS
camera	CAMERA
contacts	READ/WRITE_CONTACTS, GET_ACCOUNTS
location	ACCESS_COARS/FINE_LOCATION
microphone	RECORD_AUDIO
phone	READ_PHONE_STATE, READ/WRITE_CALL_LOG, ADD_VOICEMAIL
access	WRITE_EXTERNAL_STORAGE
control	GET_TASKS

AndroidManifest.xml

- Content
  - Permissions the app will request - not declared might be requested at runtime or implicit
  - Activities (Entry point activity)
  - Services
  - Receivers, Intent filters

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="
      sg.vp.owasp_mobile.omtg_android">

<uses-permission android:name="android.permission.WRITE_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:name="sg.
      vp.owasp_mobile.OMTG_Android.MyApplication" android:supportsRtl="true"
      android:theme="@style/AppTheme">
```

SMALI/BAKSMALI

- Equivalent to assemble/disassemble
- Human readable - but difficult to read and write
- Modification of smali - write code in Java - compile and decompile and insert

```
.super Ljava/lang/Object;
.method public static main([Ljava/lang/String;)V
.registers 2
sget-object v0, Ljava/lang/System;->out:Ljava/io/PrintStream;
const-string    v1, "Hello World!"
invoke-virtual {v0, v1}, Ljava/io/PrintStream;->println(Ljava/lang/String;)V
return-void
.end method
```

https://source.android.com/devices/tech/dalvik/dex-format

Modify and build with APKTOOL

- Example: enable debugging in AndroidManifest.xml
- Building/packing with apktool

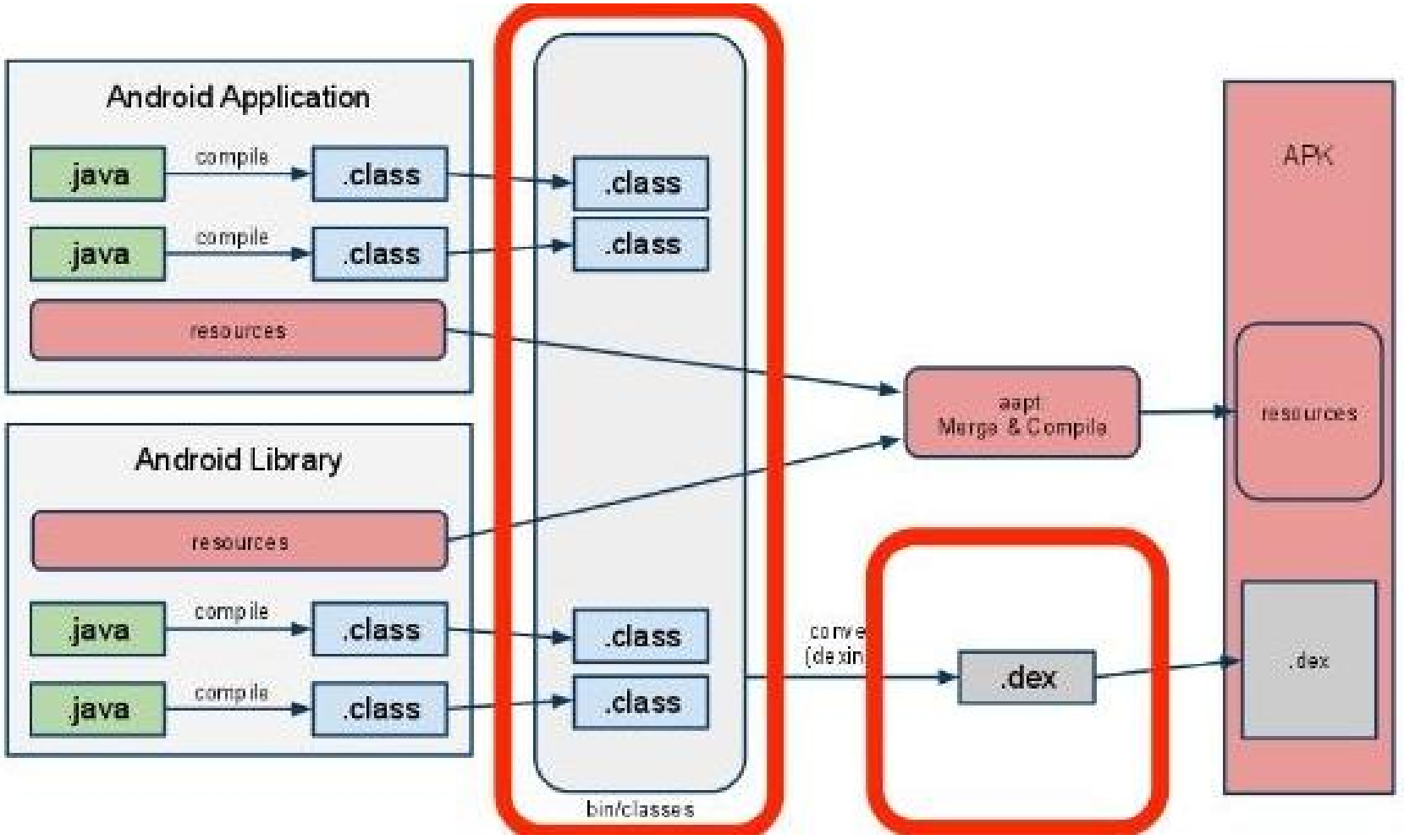
```
apktool b ./app
```

  - Generates the .apk file in ./app/dist/app.apk, but this apk is not signed
- Sign the apk
  - Generate keystore with keytool

```
keytool -genkey -v -keystore my.keystore -alias
    alias_name -keyalg RSA -validity 10000
```

  - Sign apk with jarsigner

```
jarsigner -verbose -sigalg MD5withRSA -digestalg SHA1 -
    keystore my.keystore ./app/dist/app.apk alias_name
```


Android Application
java
compile
.class
java
compile
.class
resources
Android Library
resources
java
compile
.class
java
compile
.class
.class
.class
.class
.class
bin/classes
aapt
Merge & Compile
APK
resources
.dex
.dex

Dex2Jar

- Converts Dalvik bytecode (dex) to java bytecode (jar)
- Download latest release

  `https://github.com/pxb1988/dex2jar/releases`
- Usage

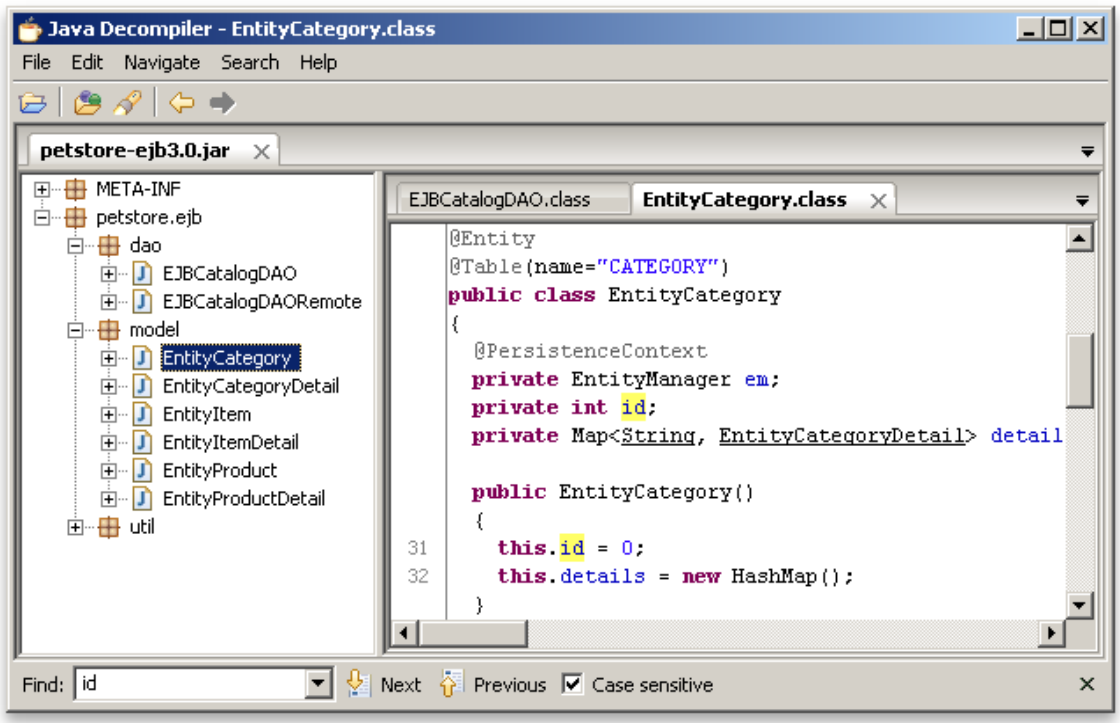
```
sh d2j-dex2jar.sh -f MSTG-Android-Java.apk
```

JD-GUI

Overview | Download | Changes

JD-GUI is a standalone graphical utility that displays Java source codes of ".class" files. You can browse the reconstructed source code with the JD-GUI for instant access to methods and fields.

JD-GUI is free for non-commercial use. This means that JD-GUI shall not be included or embedded into commercial software products. Nevertheless, this project may be freely used for personal needs in a commercial or non-commercial environments.



APK Obfuscation Techniques

- Renaming
- String encryption
- Class/resource encryption
- Reflection
- code modification
  - static - insert junkbytes - linear sweep baksmali fails
  - dynamic - unpack classes at runtime

Reversing APKs

- Static and dynamic methods
- Code injection
  - code modifications
  - function call hooking

Some Static Tools

Many different tools exist, this are just some

- **apktool** - decompiling layout, values, resources XML files
- **dex2jar** - decompiling dex(Dalvic executable) to a jar file
- **jd-gui** - java decompiler and .class viewer
- **jadx** - java decompiler https://github.com/skylot/jadx
- **Android Studio - APK Analyser** - profile and debug apks
- **Ghidra** - NSA tool for reversing (basically dex2jar)

Some Dynamic Tools

- **ADB - Android Debug Bridge**, USB debugging
- **Emulators** - f.e AVD Android Virtual Devices in Android Studio
- **bettercap** - network analysis
- **burp suite** - mock the app
- **Frida** - hook in function call of an .apk
- **Fuzzers**....

- Sources
  - ADB:https://www.fosslinux.com/25170/how-to-install-and-setup-adb-tools-on-linux.htm
  - DEX: https://source.android.com/devices/tech/dalvik/dex-format
  - Cyber Crime Defense lecture slides - Android, FH-Campus Wien Security Master, Adrian Dabrowski, Georg Kammerstetter, Georg Merzdovnik, Stefan Riegler
- Tutorials
  - https://www.evilsocket.net/2017/04/27/Android-Applications-Reversing-101/
  - https://pentestlab.blog/2017/02/06/reverse-engineering-android-applications/

- https://hydrasky.com/mobile-security/reverse-engineering-android-apk-files/
- https://medium.com/bugbountywriteup/android-ctf-kgb-messenger-d9069f4cedf8