PwnSch00l

https://sigint.mx/pwn/

Binary Hacking 0x02

Reversing Managed Languages – Workshop 1

“Dissecting Android Applications”

01.28.2019, Victor Slavov

DISCLAIMER

- All of the material in this school can be used for good (testing, research, educating), but also for bad
- We use our skills and knowledge responsibly and ethically
- We recommend you do the same
- We are not responsible for anything you do as a result of these lessons

LET’S GET STARTED!

Everything can be found on GitHub:

https://github.com/ViRb3/sigint-workshop-1

CONTENT [1/2] – THEORY

- Dalvik
  - Dalvik VM
  - Dalvik bytecode
- Dalvik VM vs. Android Runtime (ART)
- JIT (Just In Time) vs. AOT (Ahead Of Time)

CONTENT [2/2] – PRACTICE

- Challenge 1
  - Basic decompilation and recompilation
- Challenge 2
  - Anti-tamper check, bypass with optimization patch
- Challenge 3
  - Google CTF 2018 – Shall we play a game?

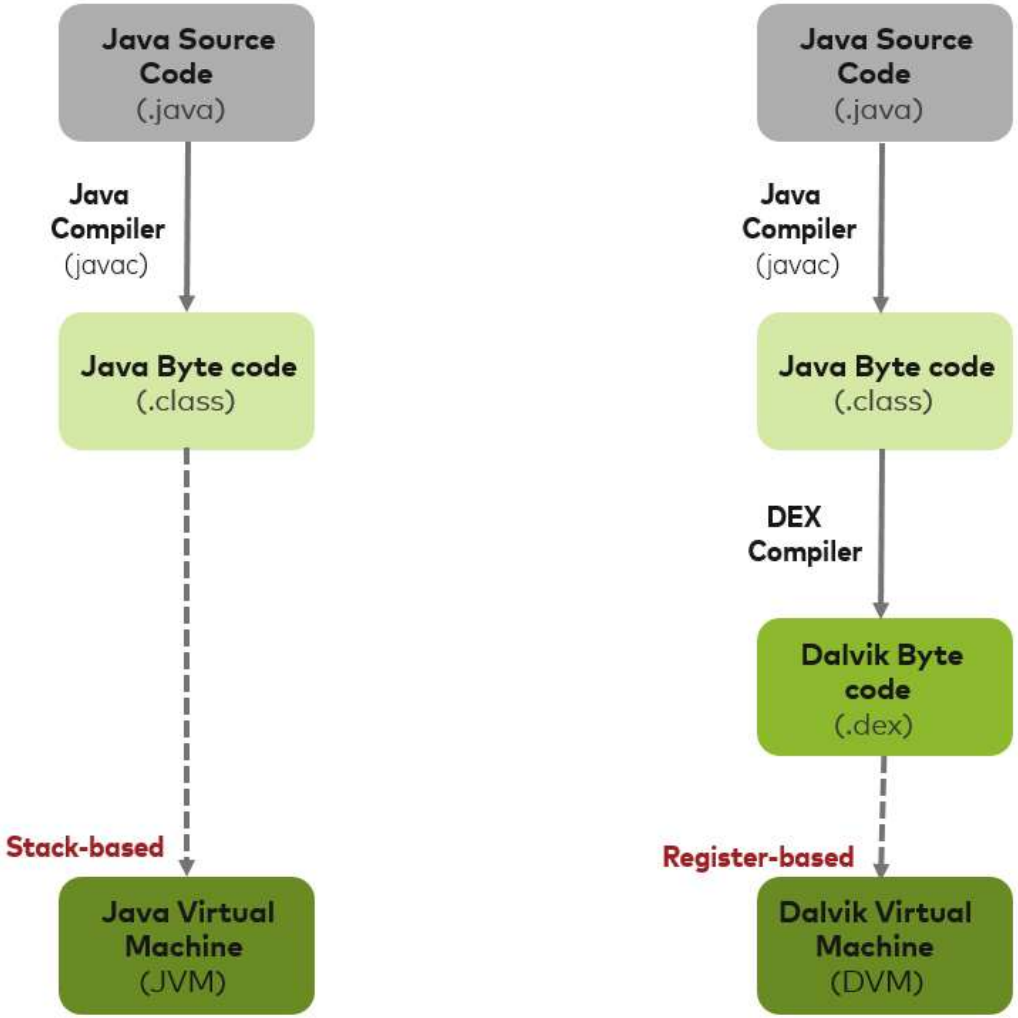
THEORY

Dalvik

- Open-source software, originally written by Dan Bornstein
- Named after the fishing village of Dalvík in Eyjafjörður, Iceland

- Dalvik Virtual Machine
- Dalvik bytecode

- A managed solution (as opposed to native/unmanaged)


Java Source Code (.java)
Java Compiler (javac)
Java Byte code (.class)
Stack-based
Java Virtual Machine (JVM)
Java Source Code (.java)
Java Compiler (javac)
Java Byte code (.class)
DEX Compiler
Dalvik Byte code (.dex)
Register-based
Dalvik Virtual Machine (DVM)

JVM vs DVM

Dalvik VM vs. Android Runtime (ART)

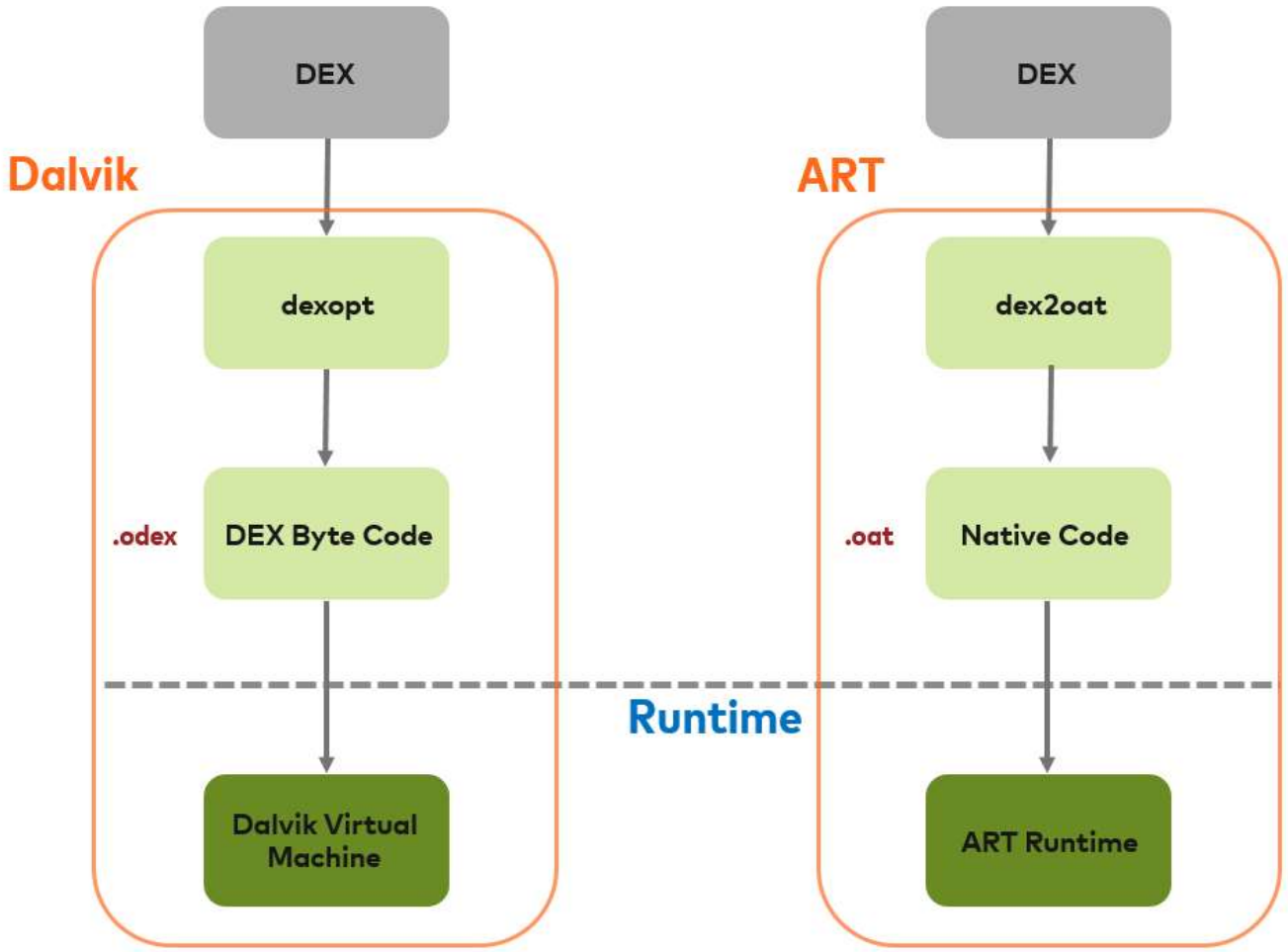
- ART replaces Dalvik VM since Android 5.0
- Reads and processes good-old .DEX files

- Dalvik uses JIT
- ART uses hybrid JIT+AOT (eventually everything is AOT’d)

JIT (Just In Time) vs. AOT (Ahead Of Time)

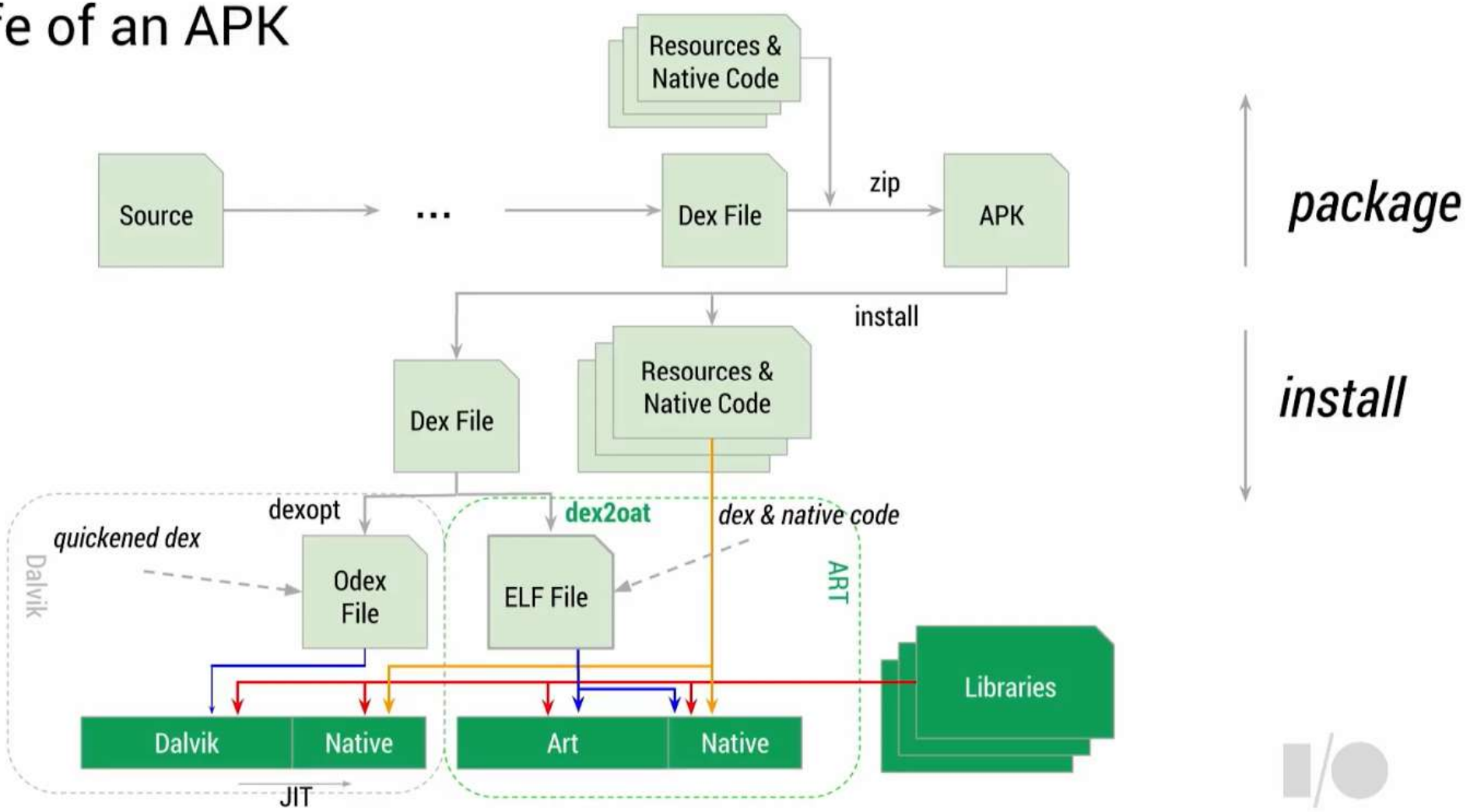
- JIT compiles just in time when the method is executed
- Restart the program, do all again
- Slow initial loading time for all methods

- AOT precompiles methods ahead of their time of execution
- Restart the program, will always use that precompiled code
- Fast, equivalent to native code loading times



Dalvik vs ART

The life of an APK



PRACTICE

SETUP

- Android SDK
  - https://developer.android.com/studio/#downloads

- (Windows) Add to PATH:
  - android-sdk/platform-tools/
  - android-sdk/build-tools/xx.x.x/

TOOLS

- APK Utilities
  - https://github.com/ViRb3/apk-utilities
- (Windows) Smali Helper
  - https://github.com/ViRb3/SmaliHelper
- Bytecode Viewer
  - https://github.com/Konloch/bytecode-viewer
- Brain.exe
  - No download at this time ☹

RESOURCES

- Dalvik opcode reference
  - http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html

CHALLENGE 1

CHALLENGE 2

CHALLENGE 3

Android isn’t secure?

WHAT'S NEXT?

- Dealing with native code (libraries)
- Dealing with obfuscation
- Debugging
- Frida

CONNECT WITH US

- Official Slack workspace
  - https://siginthqv2.slack.com
  - Or if you prefer Discord, make sure to spam the heck out of the admins to make the switch
- Website
  - https://school.sigint.mx
- Social media
  - https://www.facebook.com/SigintEdinburgh
    https://twitter.com/siginthq?lang=en