

# How to attack an Android app

Developer Event

Benjamin Adolphi <[benjamin@promon.de](mailto:benjamin@promon.de)>

# # whoami

- Master in Information Security from Høgskolen i Gjøvik.
- Breaking and securing apps since 2011.
- Head of the security research team at Promon.

# Introduction

- Teaching you how to attack simple Android apps.
- Foundation for solving technical challenge.
- Material: <https://github.com/badolphi/developer-event>

# Reverse engineering

- Understanding how an app works.
- Reveal secrets in it.
- First step of an attacker.
- Two complementary approaches: Static and dynamic
- On Android
  - Java code (Java, Kotlin)
  - Native code (C, C++, Dart, ...)

# Reverse engineering Java code

- Code in classes.dex file(s).
- Dalvik bytecode executed in VM.
- Requires disassembler<sup>1</sup> or decompiler<sup>2</sup>.



<sup>1</sup> <https://github.com/iBotPeaches/Apktool>

<sup>2</sup> <https://github.com/skylot/jadx>

# Reverse engineering native code

- Code is found in .so files.
- Executed directly on the CPU.
- There are many good disassemblers/decompilers <sup>1,2,3,4</sup>.



<sup>1</sup> <https://hex-rays.com/ida-pro>

<sup>2</sup> <https://binary.ninja>

<sup>3</sup> <https://github.com/NationalSecurityAgency/ghidra>

<sup>4</sup> <https://rada.re>

# Demo

# Repackaging

- Modifying app on disk.
- Change code to change behavior.
- Change resources to change look.



# Patching Java code

- Modify classes.dex file(s).
- Direct binary patching can be tricky.
- Tools like apktool make this easy
  - Disassemble to smali.
  - Modify smali.
  - Re-assemble to apk.



# Patching Native code

- Modify .so file(s).
- Can be done manually.
- Disassemblers/decompilers usually make this easier.
- Requires available space.

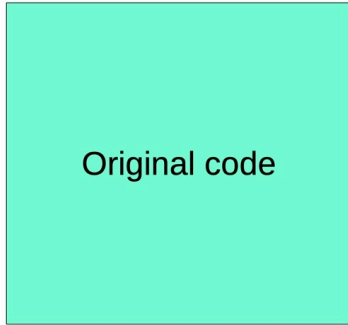
# Demo

# Hooking

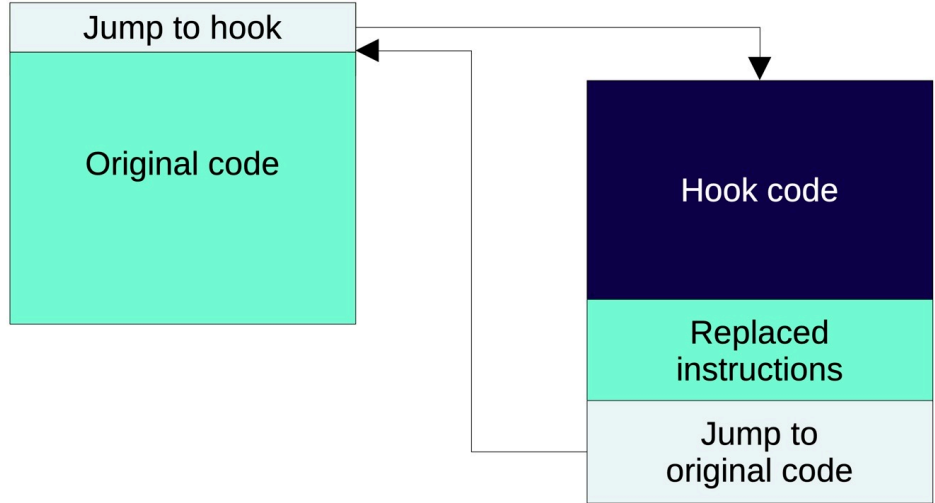
- Modify the app while it runs.
- Change code to change behavior.
- Useful for dynamic reverse engineering.

# How hooking works

Before



After



**PROMON**

# Hooking Java code

- Code is executed in VM.
- Could be compiled ahead of time or just in time.
- Requires modifying the VM.
- Popular hooking frameworks
  - LSPosed<sup>1</sup>
  - Frida<sup>2</sup>



**FRIDA**

<sup>1</sup> <https://github.com/LSPosed/LSPosed>

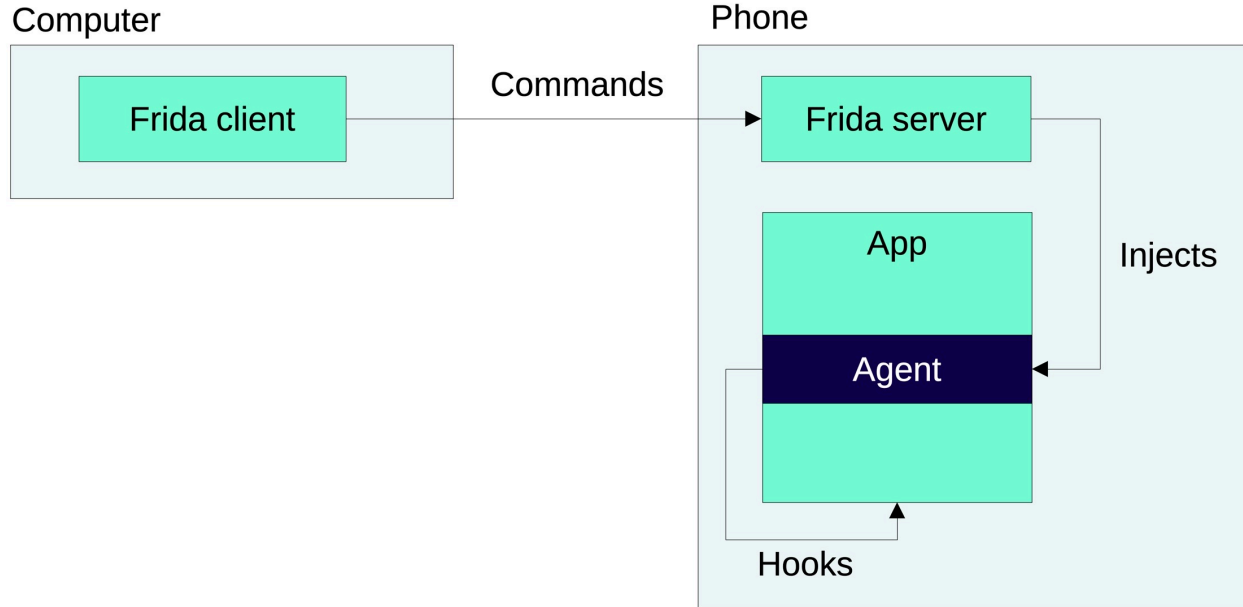
<sup>2</sup> <https://frida.re>

# Hooking native code

- Overwrite code in memory.
- Not completely trivial.
- Frida is a popular framework to use.

**FRIDA**

# How Frida works in our use case





# Demo

# Technical Challenge

- Two apps that you should attack.
- Using one of the presented attacks.
- A successful attack will give you a flag.
- Send it to: [benjamin@promon.no](mailto:benjamin@promon.no)
- First one submitting a flag for a challenge wins a price.
- First one submitting both flags wins the main price.
- One person can only win one price.

Thank you!



**Benjamin Adolphi**  
Security Ninja  
benjamin@promon.no