Breaking apps from the inside: an introduction to the FRIDA binary instrumentation framework

Matteo Palladino (@radamanth)

$whoami

- Occasional CTF player
- Hackmeeting aficionado
- Audio synthesis nerd
- Likes looking at Dex bytecode a *bit* too much
- Currently working as a Security engineer/pentester @ moveax
    - Mainly doing work in DevSecOps, sometimes researching into mobile application security

SPQR

moveax_

What will we cover?

- What is FRIDA?
- A bit about the internals
- The FRIDA API*
- Some extra tools to make our job easier*
- Demo(s)!

*Mainly from the usage of an Android pentester

What is FRIDA?

What is FRIDA?

SPQR

What is FRIDA?

“We were both familiar with IDA, which is a commercial reverse-engineering tool. The pun “FRIDA” came up, both as in “Free IDA”, but also as in the Norwegian female names Ida and Frida, where Frida could be Ida’s sister, as IDA is a static analysis tool and Frida is a dynamic analysis toolkit”

@oleavr

What is FRIDA?

FRIDA is a **Dynamic instrumentation toolkit.**

What is FRIDA?

FRIDA is a **Dynamic instrumentation toolkit.**

- Debugger, instrumenter, JS engine, all into one

What is FRIDA?

FRIDA is a **Dynamic instrumentation toolkit.**

- Debugger, instrumenter, JS engine, all into one
- It gets into (and inside) the process that it attaches to
  - Can intercept and modify function calls
  - Listen in to native calls
  - Works for a lot of systems (Win/OSx/iOS/Android)

What is FRIDA?

FRIDA is a **Dynamic instrumentation toolkit.**

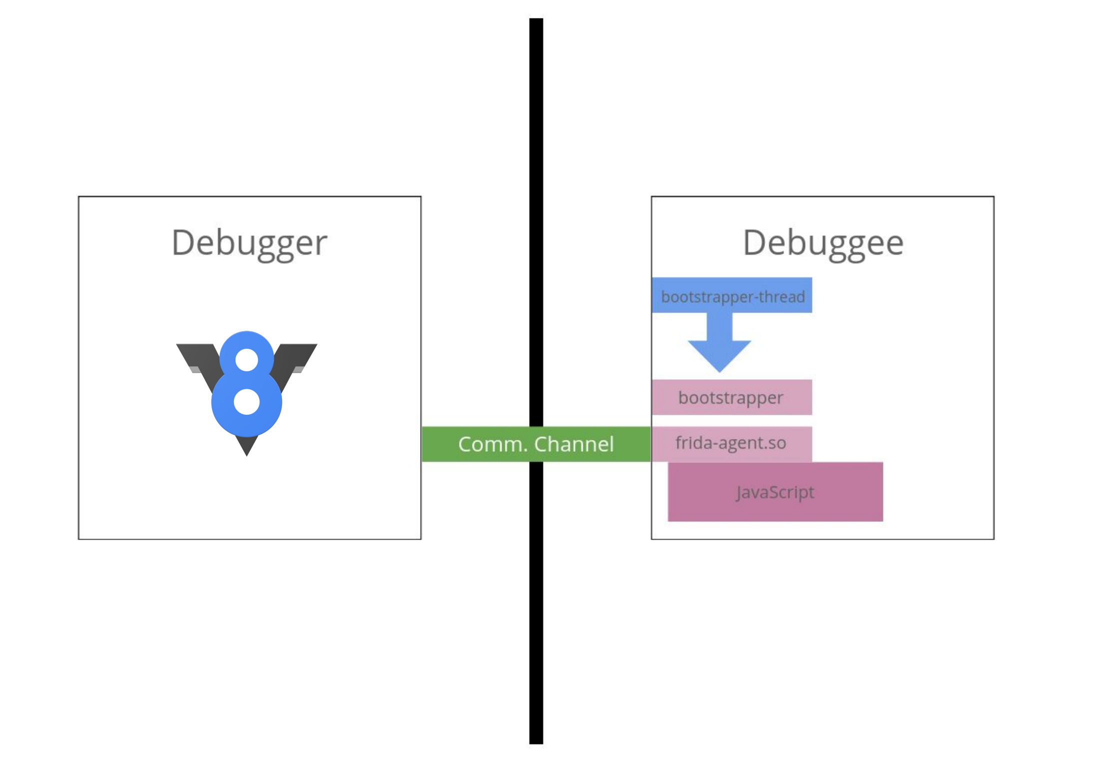
- Debugger, instrumenter, JS engine, all into one
- It gets into (and inside) the process that it attaches to
  - Can intercept and modify function calls
  - Listen in to native calls
  - Works for a lot of systems (Win/OSx/iOS/Android)
- Native JS API for interfacing with it...

What is FRIDA?

FRIDA is a **Dynamic instrumentation toolkit.**

- Debugger, instrumenter, JS engine, all into one
- It gets into (and inside) the process that it attaches to
  - Can intercept and modify function calls
  - Listen in to native calls
  - Works for a lot of systems (Win/OSx/iOS/Android)
- Native JS API for interfacing with it...
- ...but there is also the C API (not well documented like the JS one)

How?



Credit:
Ole Avr

How?

1. **Injection**

   Frida-core (through a Frida-server) injects the JS engine into a shared library that it injects into existing software

How?

1. **Injection**

   Frida-core (through a Frida-server) injects the JS engine into a shared library that it injects into existing software

2. **Embedding**

   In case of jailed systems (iOS), we can preload a “frida-gadget” by manually patching the program with a way to talk to it (usually though a frida-server)

How?

1. **Injection**

   Frida-core (through a Frida-server) injects the JS engine into a shared library that it injects into existing software

2. Embedding

In case of jailed systems (iOS), we can preload a “frida-gadget” by manually patching the program with a way to talk to it (usually though a frida-server)

3. Preloading

Using frida-gadget with good ol’ LD_PRELOAD.

Mobile style

On Android/iOS, this is usually done in two ways:

1. For non-rooted devices, we need to unpack our APK/IPA and add our hook for a FRIDA Gadget*

1. If you have an unjailed device, just install a FRIDA server and inject the library whenever you want

Mobile style

```
const-string v0, "frida-gadget"

invoke-static {v0},
Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V
```

*There’s better ways to inject the Gadget without touching Dex bytecode like *cavemen*.

Some more useful tools later!

FRIDA API

FRIDA API

Native code bindings

Interceptor

Stalker

If you can call it, FRIDA can help you with it.

Java

```
Java.use(className)

Java.cast(handle, klass)

Java.perform(fn)

…
```

Objective-C *(eugh…)*

```
ObjC.api

new ObjC.Object()

ObjC.implement(method, fn)

…
```

Kernel stuff

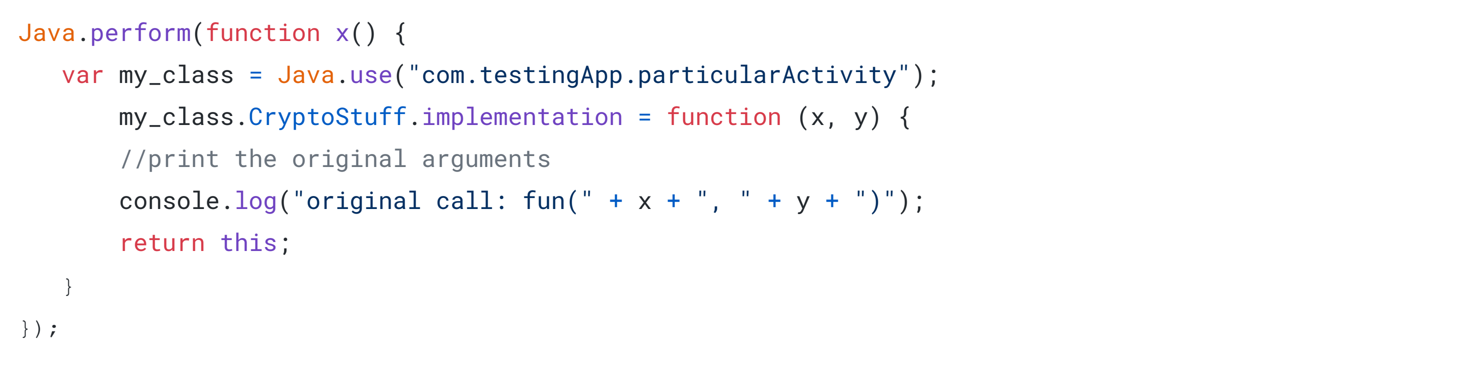
```
Kernel.enumerateModules()

Kernel.alloc(size)

Kernel.writeByteArray()

…
```

If you can call it, FRIDA can help you with it.

Let's say, for example, that you'd want to log the call for the `CryptoStuff(x,y)` method inside a particular Activity?

If you can call it, FRIDA can help you with it.

Let's say, for example, that you'd want to log the call for the `CryptoStuff(x,y)` method inside a particular Activity?



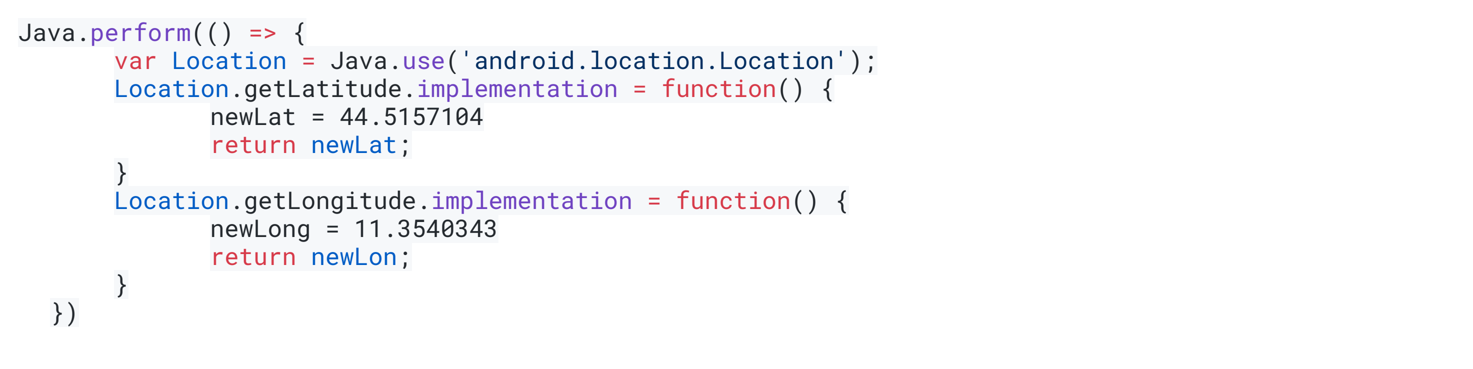
```
Java.perform(function x() {
  var my_class = Java.use("com.testingApp.particularActivity");
    my_class.CryptoStuff.implementation = function (x, y) {
    //print the original arguments
    console.log("original call: fun(" + x + ", " + y + ")");
    return this;
  }
});
```

If you can call it, FRIDA can help you with it.

Or maybe change the lat/long coordinates that the applications sees?

If you can call it, FRIDA can help you with it.

Or maybe change the lat/long coordinates that the applications sees?



```
Java.perform(() => {
       var Location = Java.use('android.location.Location');
       Location.getLatitude.implementation = function() {
              newLat = 44.5157104
              return newLat;
       }
       Location.getLongitude.implementation = function() {
              newLong = 11.3540343
              return newLon;
       }
  })
```

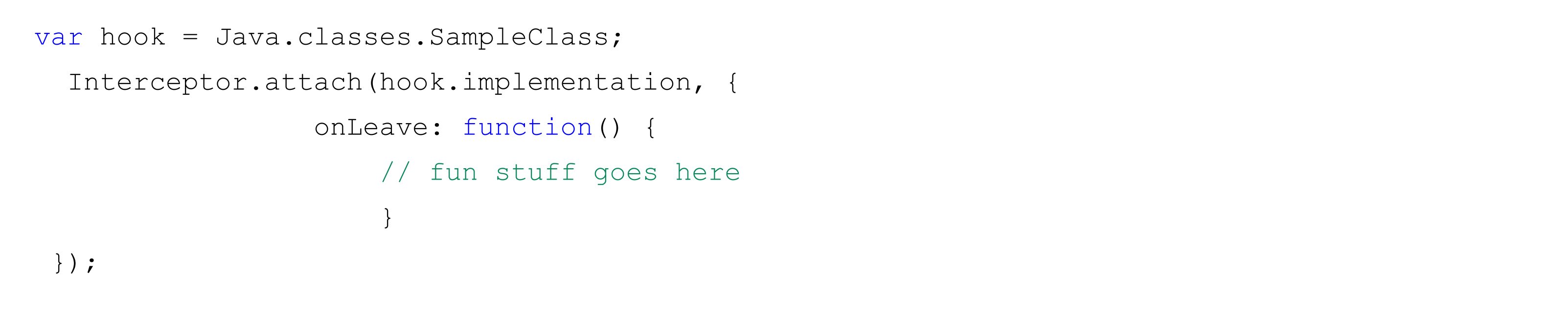
Interceptor

Interceptor

- The name says it all
- Stubs out the original call to a function, and `JUMP`s to your code

Interceptor

- The name says it all
- Stubs out the original call to a function, and JUMPs to your code



```
var hook = Java.classes.SampleClass;
  Interceptor.attach(hook.implementation, {
                 onLeave: function() {
                     // fun stuff goes here
                     }
 });
```

Stalker

Stalker

- Given a specific `thread`, Stalker will instrument and watch everything that goes on inside
    - `CALL`s
    - `RET`s
    - ALL of the instructions (this kills performance, but can be useful)
    - Synchronous callbacks to JS if a specific `CALL` is made
    - Thanks to `transform`, one can also edit the asm before it gets executed

Some cool tools/resources

- Objection - a *way* better interface to frida-cli
  - Can also browse the local filesystem, interface with the application DB etc. etc.
  - Makes patching APKs and IPAs *scaringly* easy.

OBJECTION
RUNTIME
MOBILE
EXPLORATION
GIT.IO/OBJECTION

Some cool tools/resources

- Objection - a *way* better interface to frida-cli
  - Can also browse the local filesystem, interface with the application DB etc. etc.
  - Makes patching APKs and IPAs *scaringly* easy.
- House
  - Framework for mobile application analysis based on FRIDA
  - One-click SSL pinning bypass, injection of Stetho (a WebView debugger)
  - github.com/nccgroup/house

OBJECTION
RUNTIME
MOBILE
EXPLORATION
GIT.IO/OBJECTION

Some cool tools/resources

OBJECTION
RUNTIME
MOBILE
EXPLORATION
GIT.IO/OBJECTION

- Objection - a *way* better interface to frida-cli
  - Can also browse the local filesystem, interface with the application DB etc. etc.
  - Makes patching APKs and IPAs *scaringly* easy.
- House
  - Framework for mobile application analysis based on FRIDA
  - One-click SSL pinning bypass, injection of Stetho (a WebView debugger)
  - github.com/nccgroup/house
- Frida fuzzer
  - AFL based fuzzer for in-memory application fuzzing
  - Go give some help!

Some cool tools/resources

- If you’re interested in FRIDA, a great resource is the Telegram group:
  - t.me/fridadotre
  - @oleavr is usually there
- All the documentation you could want is @ frida.re/docs/
- Some examples of working code @ codeshare.frida.re/

Demo time!

Thank you!

(questions?)

📫 pastnullinfinity@gmail.com

@Radamanth