Introduction to Frida

A Dynamic Instrumentation/Analysis Framework

Basics

What is Frida?

- Basically.. GDB? But scripting only
- Use a host language (e.g. python), start Frida
- Load JavaScript to instrument the binary
- V8 is used for assembly

Setup

- We'll be using python
- `pip install frida`

Getting Started

- We require some boilerplate code on the python side to get started
- Python API basically has a handful of important functions:
  - `attach` or `spawn`
  - `create_script`
  - `on`
  - `load` and `unload`

Starting a Session

- `import frida`
- `Session = frida.attach(name_or_pid) # Attach to a running process`
- `Session = frida.attach('./mybin')   # Spawn a new process`

Loading a Script

- `script = session.create_script(javascript) # Create new script object`
- `script.on('message', msg_callback)         # Register a callback`
- `script.load()   # Activate script`
- `script.unload() # Deactivate script`
- You might want some helper functions for loading JavaScript files and creating scripts, checkout src/frida/example.py

Scripting

Scripting with JavaScript

- Lots of API functionality: https://www.frida.re/docs/javascript-api/
- Everything is async
- Most important classes/modules: `Interceptor` and `Module`

Module

- Contains info about…
  - Symbols
  - Imports/Exports
  - Base Address
- Quick symbol lookup: `Module.findExportByName(null, 'fork')`

Interceptor

- Attach an interceptor to a call (or an arbitrary address)
- Implement `onEnter` and `onLeave`
- `onEnter`: Capture/manipulate arguments
- `onLeave`: Capture/manipulate return value

Change/Supply Memory

- `Memory.readUtf8String`
- `Memory.writeUtf8String`
- `Memory.allocUtf8String`
- And a lot more...