Reverse Engineering iOS with FRIDA

By Christine Fossaceca

Objective by the Sea

$whoami

- Senior Reverse Engineer @ The MITRE Corporation

- Researcher @furiousmac

- Dog Mom @ Honey

$whoami

Marion Marshalek @pinkflawd

- Advocate for Women in Cyber

- CoHost @

her hax
PODCAST

Agenda

- About Frida
- How to Set it Up for iOS
- My methodologies + live demos of things you can do! :)
  - Methodology #1: Connect and Explore
  - Methodology #2: Catch and Release
  - Methodology #3: Python Baby!
- Hot Tool Tips 🔥
- Conclusion + Questions?

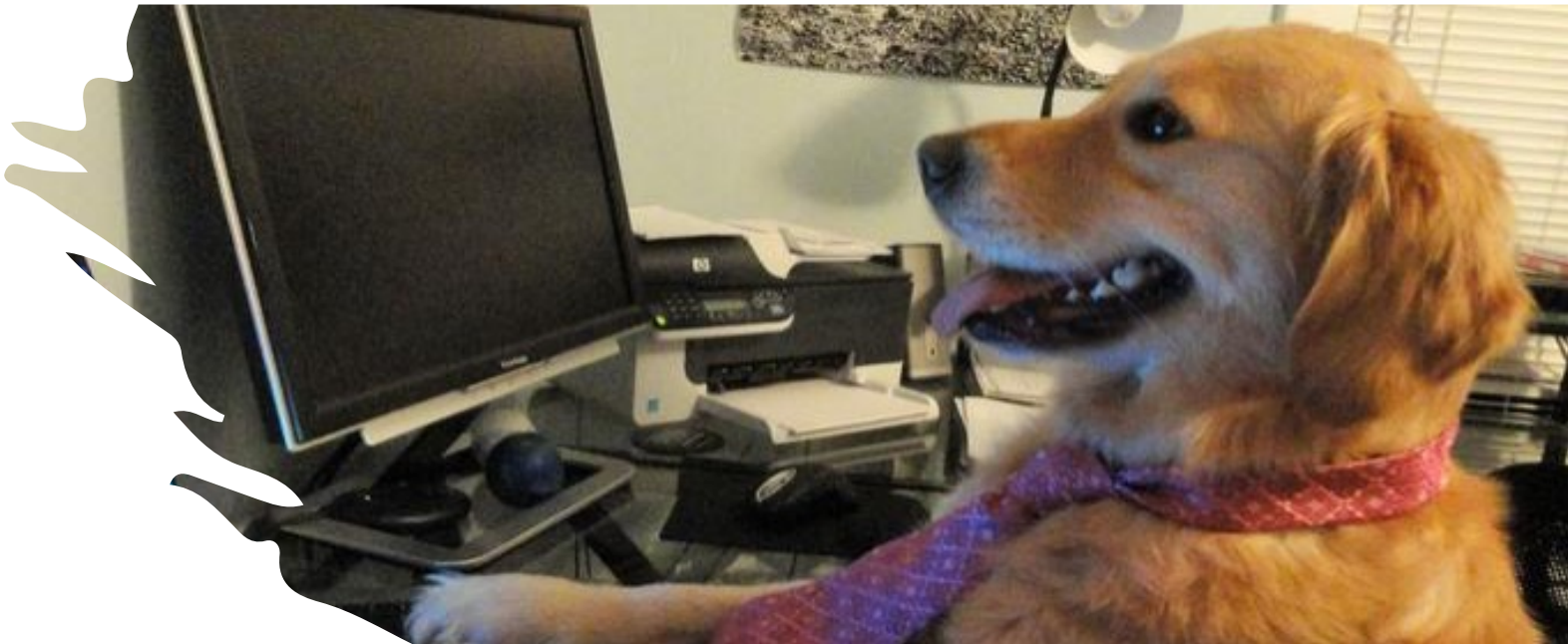
FRIDA

Frida: a brief history

- Ole André V. Ravnås @oleavr
  - Researcher at NowSecure

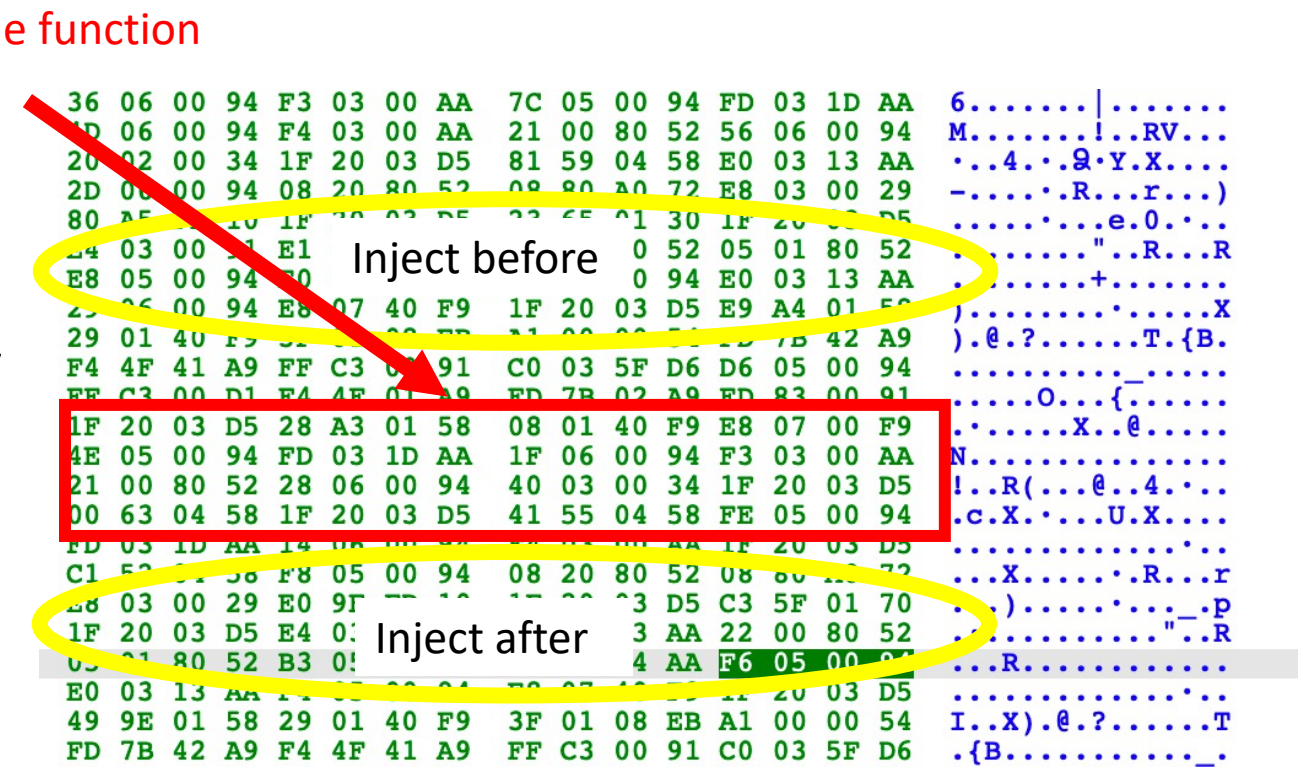
- “Think of it as a library for building debuggers”

- “Dynamic Instrumentation Toolkit”



“Dynamic Instrumentation Toolkit”

- Static 🥶 vs. Dynamic 🔥
- Instrumentation 🎻 describes how you are handling the binary itself.
- This is most useful for native process debugging, because you can break on specific areas in a binary without having source code



Frida in general

- Devices
  - Frida can be used on MacOS, PC, or Linux
  - Host PC can be target, or remote device (i.e. Android or iOS)
  - Can even use it to debug Node.js processes!

- Applications or Native Processes
  - For a great tutorial on reverse engineering iOS Apps, check out
  - Or Begam's (Cellebrite) DFRWS 2020 Frida Workshop! @shloopen

In this Demo

- iPhone 7, iOS 14.7.1
- Checkra1n jailbreak
- Frida 15 (15.1.1) on MacOS

Prepping your computer

- `pip3 install frida-tools`
- Python 3.9 is the best right now

Note!! You need the same version of Frida on your computer and your device

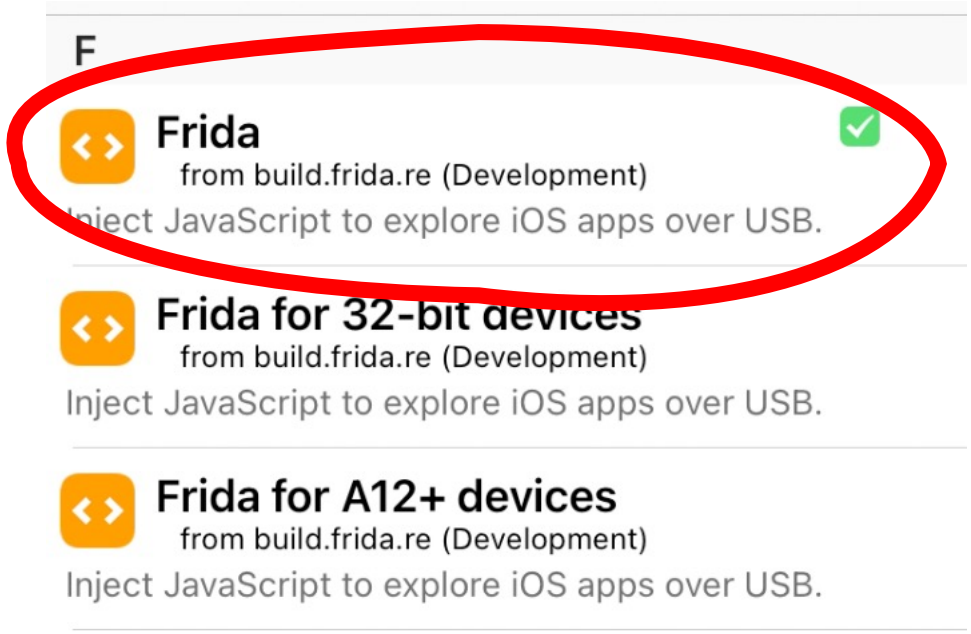
Prepping your iOS device: jailbreaking

```
if youriPhone <= iPhoneX:
    use checkra1n
else
    use Altstore + unc0ver
```

(this is for jailbreaking 64 bit devices only!!)

Prepping your iOS device: get Cydia

- Normally it is included with your jailbreak, but if not, you can download the .deb directly from the Cydia website
- Add Frida as a Source
- https://build.frida.re
- Download + Install Frida



```
➜  ~ frida-ps -U
 PID  Name
----  ----------------------------
3566  Calendar
3614  Camera
1084  Cydia
3535  Find My
3160  InCallService
3182  Messages
3521  Phone
4337  Podcasts
3636  Safari
3541  Settings
 224  Siri Search
4732  User Authentication
3579  Wallet
4301  checkra1n
 104  ACCHWComponentA
2337  ASPCarryLog
```

Quick Test

- Sanity check with

```
frida --version
frida-ps -U
```

(U means access remote device over USB)

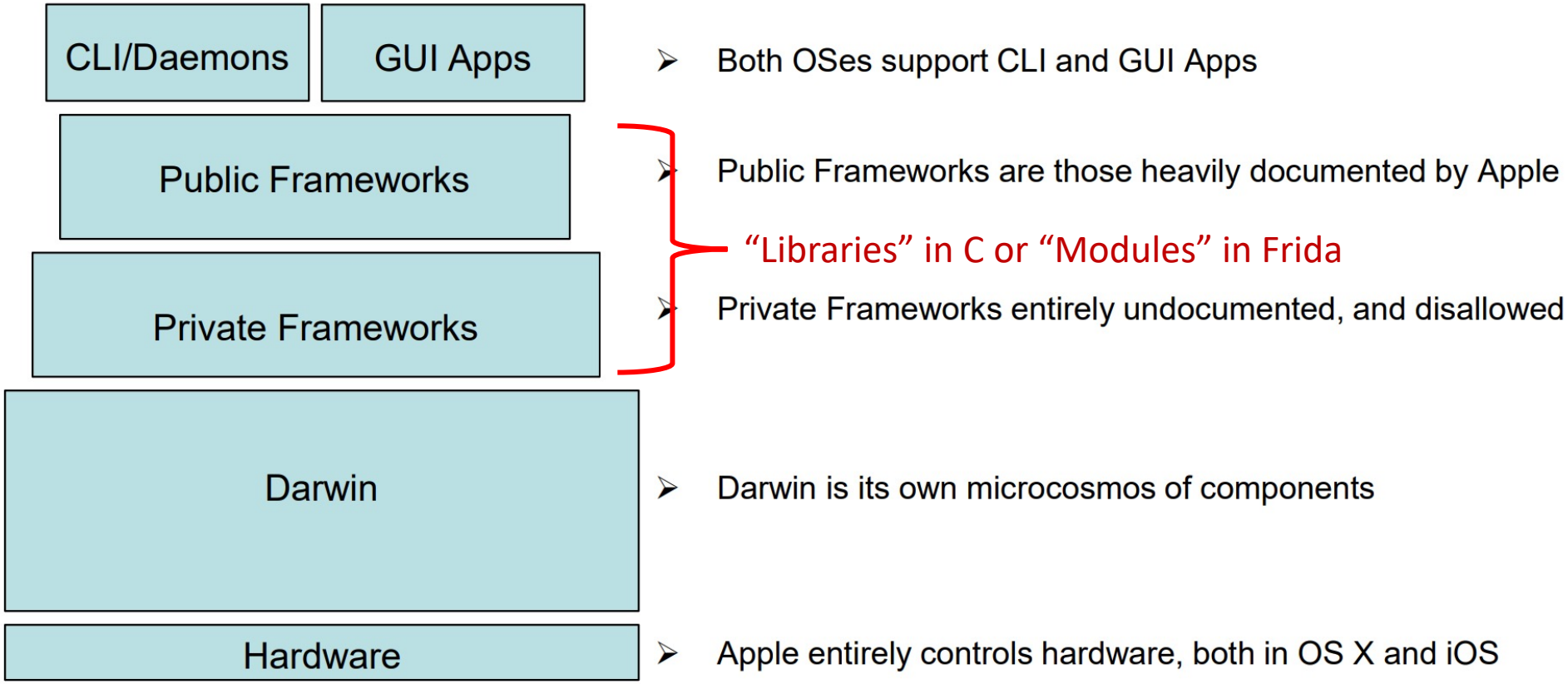
Methodology #1: Connect and Explore

```
➜  frida frida -U bluetoothd
     ____
    / _  |   Frida 15.0.19 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://frida.re/docs/home/

[iPhone::bluetoothd]->
```

OS X Architecture

- A less aesthetically appealing (but accurate) description is:



Thanks Jonathan Levin

Example: bluetoothd

- One of the biggest roadblocks right now is there is no help page……
- But, once you get the hang of it, it is actually pretty easy to use.
- Start by pressing TAB and you can see the different capabilities of Frida. For example, MATH is a class of mathematical constants and functions that can be used in your scripts.
- Process.enumerateModules
- Process.enumerateRanges('rw’)
- Script Hooking

Methodology #2: "Catch and Release"

- You know how to find the load address +path of a particular library, but what if you want to examine specific functions in that library?

- Objective C vs Non Objective C functions

Objective C Function

```
id __cdecl -[CBStackAddressMonitorBluetoothD description](CBStackAddressMonitorBluetoothD *self, SEL)
```

Regular Function

```
kern_return_t __cdecl IOServiceOpen(io_service_t service, task_port_t owningTask, uint32_t type, io_connect_t *connect)
```

- frida-trace –U bluetoothd –m "*[* *Location]"
- frida-trace –U bluetoothd –i "*Location"

iOS Zero Click Exploit

- First reported as Megalodon by Amnesty International, and was later reported as FORCEDENTRY by CitizenLab
- Really awesome report by Trend Micro's Mickey Jin @patch1t
- Series of GIFs that were maliciously encoded PDFs
- So, how can you look into this too?

TrendMicro/Amnesty Report

```
Thread 2 name: Dispatch queue: IMTranscoderNormalPriorityQueue
Thread 2 Crashed:
  0: CoreGraphics          0x181c6e228 __ZN11JBIG2Stream17readTextRegionSegEjiijPjj + 900
  1: CoreGraphics          0x181c6e20c __ZN11JBIG2Stream17readTextRegionSegEjiijPjj + 872
  2: CoreGraphics          0x181c6c67c __ZN11JBIG2Stream12readSegmentsEv + 1988
  3: CoreGraphics          0x181c6be70 __ZN11JBIG2Stream5resetEv + 260
  4: CoreGraphics          0x181cf9f9c __ZL10read_bytesPvS_m + 1024
  5: CoreGraphics          0x181c[illegible]524 _jbig2_filter_refill + 128
  6: CoreGraphics          0x181c8d098 _CGPDFSourceRefill + 196
  7: CoreGraphics          0x181c8cfa4 _CGPDFSourceGetc + 36
  8: CoreGraphics          0x181c63088 _xref_stream_read_section + 188
  9: CoreGraphics          0x181c62e60 _xref_stream_create + 528
 10: CoreGraphics          0x181c62a[illegible] _CGPDFXRefStreamCreate + 112
 11: CoreGraphics          0x181c26694 _pdf_xref_create + 1748
 12: CoreGraphics          0x181c06eb0 _CGPDFDocumentCreateWithProvider + 280
 13: ImageIO               0x181b0fdd4 __Z19CreateSessionPDFRefP10IIOScannerPb + 112
 14: ImageIO               0x181a92404 __ZN14IIO_Reader_PDF22updateSourcePropertiesEP19IIOImageReadSessionP13IIODictionaryS3_S3_P19CGImageSourceStatus + 84
 15: ImageIO               0x181a138fc __ZN14IIOImageSource13getPropertiesEP13IIODictionary + 408
 16: ImageIO               0x181a139a4 __ZN14IIOImageSource14copyPropertiesEP13IIODictionary + 16
 17: ImageIO               0x181a17f00 _CGImageSourceCopyProperties + 244
 18: IMSharedUtilities     0x18f17b974 readFileProperties:fromImageSource:error: + 48
 19: IMSharedUtilities     0x18f17c740 readFileProperties:fromImageSource:withUpdatedLoopCount:error: + 84
 20: IMSharedUtilities     0x18f17cd34 copyGifFromPath:toDestinationPath:error: + 264
 21: IMTranscoderAgent     0xecc258c8
```

CG – Core Graphics

1

2

3

https://www.trendmicro.com/en_us/research/21/i/analyzing-pegasus-spywares-zero-click-iphone-exploit-forcedentry.html

Methodology #3: Python baby!

- `pip3 install frida` *#python bindings*
- You can use python to run your javascript files
- Iterate faster
- Useful because you can pass data back to python and continue to operate on it

- ✨Special Thanks✨ Ryan Grandgenett

Hot Tool Tips 🔥

- Case Insensitive Searching
  - Added by @Hexploitable
  - frida-trace –U Messages –i “*test*/i” — Regular function
  - frida-trace –U Messages –m “*[* *test*]/i” — Objective-C function
- Live editing Javascript (Thanks Dr. Jiska Classen!)@naehrdine
  - frida –U Messages –no-pause –l Script.js
- Ephemeral process (Thanks Ole Andre!) @oleavr
  - frida-trace -U –W com.apple.imcore.imtransferagent -i open
  - (This is experimental/ WIP)

Conclusion

- Frida is FREE
- Frida is OPEN SOURCE (so if something is broken, make a ticket, or make a contribution via Github ☺)
- Frida is EASY TO USE! (I hope you think so now too)

Questions?

@xine in the Discord
christine@herhaxpodcast.com
@x71n3 on Twitter

Like and Subscribe to OBTS!