`FRida` Unleashed

Scratching beneath the surface of bug bounties

THREAT CON

Akshay

- Product Security Engineer
- mast3root
- Web2/Mobile security specialist

Bharath

- Product Security Engineer
- 0xbharath
- https://disruptivelabs.in

Why this talk?

- Mobile security testing is **nuanced**
- Mobile security testing is **daunting**
- Mobile security testing has a lot of **tribal knowledge**
- It’s fun!

FRida

Dynamic instrumentation *toolkit for developers, reverse-engineers, and* ***security researchers****.*

- **Injects V8 JavaScript engine** in the process (QuickJS/Duktape).
- Provides **simple to use APIs to build custom tools**.

FRida - Modes of Operation

1. Injected
2. Embedded
3. Preloaded

https://frida.re/docs/modes/

Hurdle 1 - Root/Jailbreak detection

- An app’s capability to detect when they are running on a jailbroken iOS or rooted Android device.
- This has become a baseline security mechanism for most apps (opinion)
- Apps use a variety of libraries or methods to implement this such as rootbeer and IOSSecuritySuite

Bypassing Root/Jailbreak detection - Easy way

There are tools and frameworks available to bypass Root/Jailbreak detection.

- Objection
- FRida Hooks
- Magisk plugins (Not Frida)
- Xposed Plugins (Not Frida)

https://codeshare.frida.re/

Bypassing Root/Jailbreak detection - Tips

If using objection -

```
objection -g com.attack.appname explore -s "android root disable”
```

https://github.com/sensepost/objection/blob/master/agent/src/android/root.ts

```
objection -g com.attack.ipaname explore -s "ios jailbreak disable”
```

https://github.com/sensepost/t/blob/master/agent/src/ios/jailbreak.ts

What to do when automated scripts or tools fail?

Using frida we can start experimenting and start hooking into interesting places and syscalls using early instrumentation.

- `/proc/*/`
- `Popen and Fopen`
- `getEnv`
- `Stat`
- `__system_property_find`
- `frida --codeshare FrenchYeti/android-arm64-strace -U -f YOUR_BINARY`

What to do when automated scripts or tools fail?

Using frida we can start experimenting and start hooking into interesting places and syscalls using early instrumentation.

Early instrumentation on Java

```
Java.deoptimizeEveything();
Java.perform(()={
Java.use(…//InsertLogic)
})
```

What to do when automated scripts or tools fail?

Early instrumentation on SO

```
var ourlib = "librarycustom.so";
var do_dlopen = null;
var call_ctor = null;
var ModBase = null;
Process.findModuleByName('linker64').enumerateSymbols().forEach(function(sym) {
    if (sym.name.indexOf('do_dlopen') >= 0) {
        do_dlopen = sym.address;
    } else if (sym.name.indexOf('call_constructor') >= 0) {
        call_ctor = sym.address;
    }
})
Interceptor.attach(do_dlopen, function() {
    var library = this.context['x0'].readUtf8String();
    if (library != null) {
        if (library.indexOf(ourlib) >= 0) {
```

A quick tale from trenches (case study)

- This is from a Private Bug Bounty Program with Mobile app in scope
- We had early advantage as we got invited after 1 month of launch
- We noticed that almost no one reported bugs on the Mobile application
- We used `APKiD` to identify the protector which was used in the Android App
- Bypassed the Root detections using Frida hooking
- Multiple APIs were vulnerable to trivial issues like SQL injection
- Reported and Got it Fixed

Hurdle 2 - Certificate Pinning

Certificate pinning is mechanism that allows accepting only authorized ("pinned") certificates for authentication of client-server connections.

This mechainism is devised as a means of thwarting MiTM. This essentially means, we will not be able to use our interception proxies to manipulate API traffic.

Bypassing Certificate Pinning - Tips

- Frida Scripts which are commonly available on codeshare
- Objection
- Hash replacement
  - network_security _config.xml (Android)
  - Info.plist (iOS)
- SSLkillswitch
- Flutter
  - `disable-flutter-tls-verification`

What to do when automated scripts or tools fail?

- Identify the ways in which application communicates with server
- Identify the network stack which the application uses
- Identify whether the application relies on the OS provided SSL libraries or comes with custom ones.
  - OS Provided - LibSSL.so
    - To Hook - SSL_read and SSL_write
  - Custom ones
    - BoringSSL
    - WolfSSL etc

What to do when automated scripts or tools fail?

Bypassing SSL Pinning on custom library

```
function CaptureSSLTraffic() {

  let _sslWrite = libSymbol('libcocos2dcpp.so!SSL_write');
  let _sslWriteOld = new NativeFunction(_sslWrite, 'int', ['pointer', 'pointer', 'int']);
  let counter = 1;

  // traffic out
  console.log('raplacing [libcocos2dcpp.so!SSL_write]');
  Interceptor.replace(_sslWrite, new NativeCallback((ctx, buffer, length) => {

    counter++;

    // remove gzip compress feature
    let replace = buffer.readUtf8String(length).replace('Accept-Encoding: deflate, gzip\r\n', '');
    let newBuffer = Memory.allocUtf8String(replace);
```

Hurdle 3 - Encryption mechanisms

Apps can encryt the network traffic to ensure attacker don’t get visibility into the traffic. In this case, as attacker you won’t be able to manipulate traffic even after bypassing root detection/SSL Pinning etc.

Common techniques to figure out encryption Logic

- Frida Scripts which are commonly available on codeshare
  - `frida --codeshare dzonerzy/aesinfo -f com.app`
- Objection
- Looking for patterns
  - Java - `javax.crypto` library
  - ios - `CCrypt`

What to do when automated tools/scripts fail?

`frida-trace` to rescue

```
frida-trace -U -i "encry*" com.appname
frida-trace -U -i "ccrypt*" com.appname
```

Hurdle 4 - FRida Detection

Frida detection is, well, mechanisms to detect if Frida is being run on a mobile device.

Common ways to bypass Frida detection

- Objection renames the Frida to `Freeda` binary
- Bypassing String based Search
- Bypassing Memory based Search
- Bypassing port based Detections

Taking it to next level - Brida

Brida is a Burp Suite Extension that, working as a bridge between Burp Suite and Frida, lets you use and manipulate applications' own methods while tampering the traffic exchanged between the applications and their back-end services/servers.

Shoutouts & people to follow

- Ole André
- Leon Jacobs
- Jiska Classen
- Eduardo Novella
- FrenchYeti