CORELLIUM

Course Syllabus

Intro to iOS Kernel & Hypervisor Tools

This course is designed to give an overview of using the Corellium platform to dig deeper into the inner workings of Apple's iOS operating system while learning reverse engineering techniques. Students will get first-hand knowledge of tactics and techniques used to discover vulnerabilities and create exploits.

Section 1: Intro to Kernel Debugger

Covers the basics of using lldb with Corellium

- Getting connected to a device with lldb
- Where to find help

- **Basic Commands**
  - Register read/write
  - Backtrace
  - Breakpoints
  - Memory read/write

- **Python Scripting**
  - Hello World example
  - Walking process list example

- **Monitor commands**
- **Debugging the kernel entry point**
- **iBoot debugging**

Section 2: Hypercalls

Covers the available hypercalls:

- Kernel breakpoints from userspace
- Console logging
- Manipulating kernel memory
- Obtaining kernel/system information

Section 3: Hypervisor Hooks

Covers the usage of hypervisor hook ("Frida in the kernel")

- **Hook language**
  - Patch points
  - VM state
  - Variables
  - Control flow
  - Built-in functions

- **Use case 1: Profiling the allocator**
- **Use case 2: Disabling mitigations**
- **Use case 3: Logging file opens**

Target Audience

Reverse engineers, security and/or malware researchers and forensics experts interested in Apple's iOS operating system.

Prerequisites

- Familiarity with UNIX-derivatives (e.g. Linux, macOS, BSD)

- Familiarity with usage of Corellium (e.g. the quickstart)
  - Device creation
  - VPN
  - USBFlux
  - SSH

- Some understanding of C and Arm64 assembly

- IDA Pro with Arm64 support (Hex-Rays recommended) or Binary Ninja
  - Ghidra should work but has not been tested with these materials
  - Recommended: Install the Lighthouse plugin for IDA Pro/ Binary Ninja

Section 4: Kernel Tracing*

Covers capturing kernel flow from running /bin/ls and importing coverage into Lighthouse

- **What is program flow?**
- **How does this relate to code coverage?**
- **How does it work?**
- **Preparing to capture flow data: Running btgen/btasm**
- **Capturing data: Running hyptrace**
- **Converting data to human-readable file: Running btrace**
- **Examining the human-readable trace**
- **Importing into Lighthouse**
- **Concept: Integration into other tools**

Section 5: SEP Debugging*

- **SEP Overview (purpose, architecture, etc)**
  - What is it?
  - What does it handle?
  - How does it communicate?
  - Security model
  - Previous attacks
- **Firmware and encryption**
- **Example debugging session**

Section 6: CHARM Miscellaneous Features**

Connecting to charmd

- Console log
- Full memory dump

* Requires on-premises appliance and Premium license
** Requires on-premises appliance

CORELLIUM