Chrome (Renderer) Exploitation on Android

Course Description

Chrome, as one of the most commonly used browsers, presents an attractive target for security researchers. Playing a major role in the Android ecosystem, Chrome browser exploitation is an essential part of traditional 1-click chains. Given the rising complexity and the number of exploitation mitigations, this training attempts to address the entry barrier into browser exploitation for novice researchers. This training focuses on the Chrome renderer exploitation (RCEs) - the first step in gaining arbitrary code execution on the device. The focus is primarily on v8 vulnerabilities and common exploitation techniques covering both 32-bit and 64-bit Chrome versions on Android 13/14. It is largely self-contained and provides a generous amount of background information required to bootstrap your own Chrome research. Similar to our other trainings, the course is structured as several theory modules (providing the necessary background material), followed by hands-on lab exercises demonstrating learned concepts in practice. The main target for this training is 64-bit Chrome. Where applicable, any differences with 32-bit will be briefly discussed.

Prerequisites

- Basic JavaScript
- C++ knowledge
- ARM64 assembly knowledge
- Familiarity with GDB (GNU Debugger)

Who Should Attend?

Security researchers

Requirements

- Any host OS - Windows, Linux, MacOS (x86_64)
- Hypervisor software supporting standard OVF/OVA files
- Rooted Android device (preferably Pixel) with at least 6GB RAM
- At least 60GB of free disk space
- At least 8GB of RAM

Key Learning Objectives

- Chrome internals
- v8 exploitation
- Common exploitation techniques
- Chaining techniques / payloads
- Post exploitation

Agenda

Day 1

- Chrome internals overview

- Scripting basics
- DOM Tree
- DOM Events
- JavaScript engine introduction
- JavaScript VMs and the stdlib

Day 2

- v8 internals
- Memory management / v8 heap
- Usermode callbacks in JavaScript
- Garbage Collection
- JIT Compilers
- Common exploitation primitives

Day 3

- 32-bit vs 64-bit ARM exploitation
- Building exploitation primitives
- Renderer R/W
- WebAssembly basics
- Arbitrary code execution
- Post exploitation techniques
- Chaining renderer RCE with a payload (sandbox escape)