Fuzzing the Linux Kernel (3 days)

Training description

Andrey Konovalov, xairy.io

Title

Fuzzing the Linux Kernel

Overview

This training guides researchers through the field of Linux kernel fuzzing. In a series of practical labs, the training explores using syzkaller and KASAN for finding kernel memory corruption bugs and analyzing their security impact.

The training starts with an introduction to Linux kernel fuzzing. This part covers making kernel-specific fuzzing harnesses for finding bugs, evaluating the harness effectiveness, and using KASAN to analyze the security impact of discovered vulnerabilities.

The second part of the training focuses on syzkaller — the most widely used production-grade Linux kernel fuzzer. This part covers setting up and running syzkaller in its default configuration and also customizing syzkaller for targeted fuzzing of specific kernel subsystems.

Key learning objectives

- Security-relevant Linux kernel internals and attack surface.
- Usage and internals of Kernel Address Sanitizer (KASAN).
- Writing and evaluating kernel-specific fuzzing harnesses.
- Collecting and analyzing kernel code coverage with KCOV.
- Practical usage and internals of syzkaller.
- Writing custom syscall descriptions for syzkaller.
- Implementing pseudo-syscalls for syzkaller.

Student requirements

- Working C knowledge.
- Familiarity with common types of vulnerabilities in userspace applications.
- Basic knowledge of Go would be a plus, but it is not strictly required.
- No knowledge about Linux kernel internals is required.

Hardware requirements

- Relatively modern **x86-64** laptop suitable for fuzzing (or access to a remote server).

- At least 100 GB of free disk space.
- At least 16 GB of RAM.
- Ability to plug in an untrusted USB drive (relevant for corporate laptops).

Software requirements

- Host OS: **Linux only.**
- Docker.

Provided to students

A USB drive with:

- Presentation slides.
- Detailed lab guides with step-by-step instructions.
- Docker images with required tools and source code.

Course agenda

Day 1 — Setup and KASAN:

- Internals and setup: security-relevant Linux kernel internals; kernel attack surface; types of kernel vulnerabilities; running kernel in QEMU; setting up fuzzing environment.
- Detecting bugs: using KASAN to detect and analyze memory corruptions; KASAN internals; reading kernel bug reports; assessing impact of kernel bugs.

Day 2 — Basics of kernel fuzzing:

- Basics of kernel fuzzing: writing and evaluating kernel-specific fuzzing harnesses; Human-in-the-Loop fuzzing; collecting kernel code coverage with KCOV.
- Introduction to syzkaller: API-aware fuzzing; coverage-guided fuzzing; building, configuring, and running syzkaller.
- Using syzkaller: focused fuzzing of specific kernel subsystems; writing custom syscall descriptions in syzlang; evaluating written descriptions.

Day 3 — Advanced fuzzing with syzkaller:

- Advanced syzlang features; adding pseudo-syscalls; working with corpus; adding seeds and runtests; reproducing crashes; working with syz and C reproducers.

Trainer's bio

Andrey Konovalov is a security researcher focusing on the Linux kernel.

Andrey found multiple zero-day bugs in the Linux kernel and published proof-of-concept exploits for these bugs to demonstrate the impact. Andrey is a contributor to several security-related Linux kernel subsystems and tools: KASAN — a fast dynamic bug detector, syzkaller — a production-grade kernel fuzzer, and Arm Memory Tagging Extension (MTE) — an exploit mitigation.

Andrey gave talks at security conferences such as OffensiveCon, Zer0Con, Android Security Symposium, Linux Security Summit, LinuxCon, and PHDays. Andrey also maintains a collection of Linux kernel security–related materials and a channel on Linux kernel security.

See xairy.io for all Andrey's articles, talks, and projects.