T+7	8 Hours		
23 Aug	Wednesday	0900-17:00 ICT/GMT+7	8 Hours
24 Aug	Thursday	0900-17:00 ICT/GMT+7	8 Hours

This training guides researchers through the field of Linux kernel security. In a series of exercise-driven labs, the training explores the process of finding, assessing, and exploiting kernel bugs in a modern Linux distribution on the x86-64 architecture.

Besides providing a foundation for writing Linux kernel exploits, the training covers the no-less important areas of finding kernel bugs and evaluating their security impact. This includes chapters on using and extending dynamic bug-finding tools, writing custom fuzzers, and analyzing crashes.

The training starts with the beginner topics but proceeds into a few advanced areas as well.

Students will be Provided With

A USB drive with:

- Presentation slides.
- Detailed lab guides with step-by-step instructions.
- Virtual machine images with tools, exercise binaries, and source code.

Course agenda

Day 1 — Internals and Sanitizers:

- Internals and debugging: x86-64 architecture refresher; introduction to the Linux kernel; attack surface; types of vulnerabilities; setting up a kernel debugging environment; using GDB to debug the kernel and its modules.
- Detecting bugs: using KASAN to detect and analyze memory corruptions; KASAN internals; extending KASAN; KMSAN and other Sanitizers; reading kernel bug reports; assessing impact of kernel bugs.

Day 2 — Fuzzing:

- General fuzzing: writing and evaluating kernel-specific fuzzing harnesses; Human-in-the-Loop fuzzing; collecting coverage with KCOV; using KCOV remote coverage; fuzzing externally-triggerable code paths.
- Fuzzing with syzkaller: API-aware fuzzing; coverage-guided fuzzing; using syzkaller; writing syscall descriptions.

Day 3 — Escalating privileges and bypassing mitigations:

- Escalating privileges: ret2usr; overwriting the cred structure; overwriting modprobe_path; kernel stack buffer overflows; arbitrary address execution and arbitrary read/write primitives.

- Bypassing mitigations: KASLR, SMEP, SMAP, and KPTI bypass techniques; in-kernel Return-Oriented Programming (ROP); out-of-bounds vulnerabilities; information leaks.

Day 4 — Exploiting slab corruptions:

- Exploiting basic slab corruptions: slab out-of-bounds and use-after-free vulnerabilities; slab-specific mitigations; slab spraying; data-only exploitation; the unlinking attack.
- Modern slab exploitation techniques: userfaultfd and FUSE; elastic objects; cross-cache corruptions.
  Beyond: learning advanced exploitation techniques; useful references.