CYBERIUM
SIMULATOR

ZX340

SPECIALIST · LEVEL 7

Linux Kernel Exploitation

10
MODULES

198
LAB TASKS

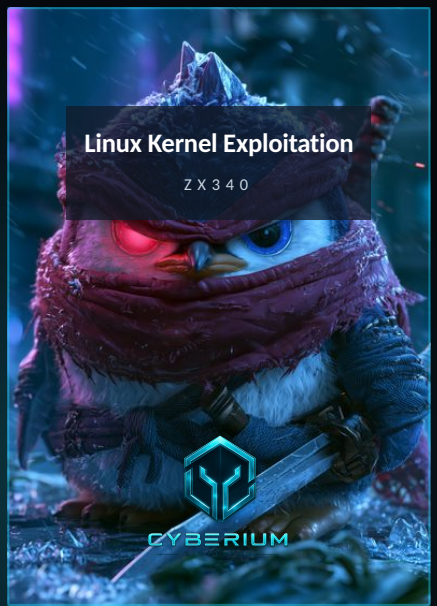
364
CTF CHALLENGES

72
TOPICS

10
COURSE BOOKS

101
SLIDES

10 COURSE BOOKS INCLUDED



This program takes experienced exploit developers inside the Linux kernel itself. You will start by following kernel execution and building a reproducible kernel debugging lab, then learn to diagnose memory corruption bugs and turn them into working exploitation primitives. From there you will build and inspect full exploitation chains, and explain exactly why a chain succeeds or fails against modern hardening and eBPF restrictions. The journey continues through kernel payloads, controlled patching, fuzzing and the triage of a reproducible bug, then real CVEs traced to their source and their fixes. The program concludes with container and namespace boundaries and with exploit reliability, leaving you able to explain a kernel bug from evidence, reproduce it, derive a primitive and verify a patch. It assumes you are already comfortable with C, Linux systems programming, x86-64 assembly and user-space exploitation.

CURRICULUM & WHAT EACH MODULE COVERS

	MODULE	TASKS
01	**Kernel Architecture** From Power-On to Linux — Understanding How a Kernel Comes to Life · Lab Environment and Decoding Raw Kernel Output · The VFS Open and Release Path · *+6 more*	32
02	**Kernel Debugging** Kernel Boot Internals and Symbol Address Shifts · The Kernel Debugging Lab Environment · ELF Symbol Files and Debug Information · *+2 more*	18
03	**Kernel Vulnerabilities** Object Lifetime and Reference Counting Fundamentals · The Kernel Heap Allocator and kmalloc Caches · Reading and Interpreting KASAN Crash Reports · *+3 more*	18
04	**Kernel Exploitation** Kernel Address Space Layout Randomization and Symbol Discovery · Kernel Module Vulnerability Exploitation and Privilege Escalation · The Linux Kernel Credential Model · *+4 more*	17
05	**Kernel Mitigations and eBPF** Kernel Memory Exposure and Address-Space Protections · User/Kernel Memory Boundary Validation · Kernel Code and Control-Flow Integrity · *+6 more*	21
06	**Kernel Payloads** x86-64 Calling Convention & Stack Mechanics · Kernel Memory Layout & Syscall Table Analysis · Kernel Execution Context & Concurrency Safety · *+4 more*	15
07	**Bug Discovery and Fuzzing** Reading Kernel Driver Source for Vulnerable Patterns · Character Devices and the Kernel Permission Model · KASAN and Heap Buffer Overflow Detection · *+6 more*	19
08	**CVE Analysis and Patching** Case Studies in Kernel and Userspace CVEs · Buffer Overflows and Bounds Checking · Pipe Buffers and CVE-2022-0847 ("Dirty Pipe") · *+2 more*	19
09	**Containers and Namespaces** Control Groups (cgroups v2): Resource Limiting · Process Isolation: PID and Mount Namespaces · Host Identity Isolation: UTS Namespace · *+6 more*	21
10	**Reliability & Detection** Host-Guest File Sharing and State Comparison Tooling · Detecting Anomalies in Mixed Event Logs · Kernel Boot and Runtime Diagnostics via dmesg · *+3 more*	18

GOOD TO KNOW

Best after a level 5–6 programme · Self-paced or trainer-led · Cy, the AI helper, answers in the student's own language

HOW EVERY MODULE RUNS

Lab → Extra at 80% → Timed test at 85% → Slides at 90%

A REAL TASK FROM MODULE 6 — KERNEL PAYLOADS

"Read /root/evidence/item29-gadgets/candidates.txt on the lab machine. What is the first-instruction address of the sole candidate meeting the stated payload requirement without the prohibited side effect? Enter only sixteen hexadecimal digits, without the hexadecimal prefix."

Finish with interview preparation

DATA AS OF 2026-11-09 · V261109