Introduction to Linux Kernel Exploit Development

Email: training@cyseclabs.com
Web: https://cyseclabs.com/training

Course Description: The number of user-land exploitation countermeasures significantly outweighs the kernel protection solutions. Due to the complexity associated with exploiting user-land vulnerabilities (ASLR, NX, Fortify, RELRO, etc.), Linux kernel with its huge publicly available codebase has become an appealing target for exploit developers. A successful exploitation of a kernel vulnerability allows attackers to elevate privileges bypassing any user-land protections and exploit mitigations.
This course teachers common kernel exploitation techniques on modern Linux distributions. It is designed for students already familiar with user-land exploitation who want to play with the heart of the OS and gain fundamental knowledge required to develop reliable and effective kernel exploits. Even though this course is designed for beginners in kernel exploitation, a number of more advanced topics, such as reliable exploitation of heap vulnerabilities and SMEP bypasses, are discussed.
This course aims to provide the fun, excitement and rewarding experience of getting a # prompt after hours of hard work. There is a "Capture the Flag" contest at the end where you can practice your newly acquired skills in kernel exploitation.

Key learning objectives:

- Introduction to Linux kernel exploit development on modern distributions
- ret2usr attacks
- Exploiting kernel heap and stack vulnerabilities
- Exploiting out of bounds (OOB) vulnerabilities
- Integer signedness bugs and overflows
- Reliable exploitation of use-after-free (UAF) vulnerabilities
- In-kernel return-oriented programming (ROP)
- Practical SMEP (Supervisor Mode Execution Protection) bypasses

Who should attend:

- Pentesters, reverse engineers, bug hunters and exploit developers
- Information security professionals experienced in user-land exploitation

Prerequisites:

- Familiarity with x86 architecture
- Linux working proficiency
- C and assembly programming knowledge
- Familiarity with GDB (GNU Debugger)
- Fundamental knowledge of common user-space exploitation techniques (e.g., stack and heap overflows, integer type conversion vulnerabilities and overflows, etc.)

Hardware and software:

- Base OS - Windows, OS X, Linux
- Ivy Bridge+ CPU (not essential)
- At least 20GB of free disk space
- At least 4 GB of RAM
- Wireless network card
- VMWare Workstation (v9+) or Fusion (v5+)

Course duration: 3 days

Day	Content
Day 1	• Introduction to kernel exploits – User-space vs kernel-space – x86(-64) architecture – User-space processes – Virtual memory management – Linux privileges model • Setting up the debugging environment – Kernel debugging techniques – Remote kernel debugging with VMWare and GDB – Module debugging • Privilege escalation – Privilege escalation on modern distributions – Privilege escalation heuristics • Introduction to `ret2usr` attacks – Arbitrary read/write primitives – IDT (Interrupt Descriptor Table) overwrites – Triggering the vulnerability – Fixating - recovering the kernel state
Day 2	• Information disclosure – Kernel memory addressing disclosures – Memory leaks and `addr_limit` user/kernel boundary • Controlled, partially-controlled and uncontrolled read/write primitives • Out of bounds (OOB) access vulnerabilities • Integer vulnerabilities – Signedness issues – Integer overflows • Kernel stack overflows
Day 3	• Dynamic memory management/SLAB allocator – Generic SLAB concepts – Linux SLUB allocator • Heap vulnerabilities – Use-after-free (UAF) vulnerabilities – Overflowing into adjacent objects – Off-by-[one/X] • Reliable UAF exploitation on SMP systems • Kernel return-oriented programming (ROP) • Bypassing SMEP (Supervisor Mode Execution Protection) • CTF