CNO DEVELOPERS CAPABILITIES COURSE

This is an intense, hands-on course designed to take students through the steps needed to develop their own exploits on both Windows and Unix-based operating systems. The course begins with an overview of Python, which is used to develop and deliver most of the exploits. After, students create custom shellcode in Assembly Language, learning how to properly groom the registers and stack for execution. Students create custom exploits against applications, learning to how to fuzz the applications' inputs to find vulnerabilities and successfully execute arbitrary code on the system. Students learn about the protections used by systems and compilers to block successful exploitation, and how these protections can be subverted. On the last day students are challenged with a culmination exercise that takes them through all of the attacker steps from getting onto a remote machine via a web vulnerability to using a buffer overflow to gain root access. The course teaches these skills by walking the students through the development of all the necessary tools from scratch - it does not rely on Metasploit or any other exploit framework.

CHIRON METHODOLOGY DOMAIN

CYBER DEVELOPMENT PROFESSIONAL™ (CDP)™

DURATION
5 DAYS

RECOMMENDED PRE-REQUISITES

- Familiarity with VMware Player or Workstation
- Exposure to Linux or Unix-based Operating Systems
- Some experience with Python or other scripting language
- Interest in writing code

PHONE: 410.672.1552 x 109 **FAX:** 410.672.3187 **ADDRESS** 7021 Columbia Gateway Dr. Suite 250 Columbia, MD 21046

CNO DEVELOPERS CAPABILITIES COURSE

COURSE SCHEDULE

DAY 1:

LEARNING OBJECTIVES

- Understanding hacker methodology
- Ability to create basic programs in Python

OUTLINE

- Orientation & introductions
- Hacker methodology
  - Footprinting
  - Scanning
  - Exploitation
  - Privilege escalation
  - Post-exploitation
- Python programming
  - Basics of learning a new programming language
  - First program
  - Python data types
  - Imported modules
  - String manipulation

DAY 2:

LEARNING OBJECTIVES

- Understanding the basics of IA32 architecture
- Understanding the basics of the Linux and Windows Operating Systems
- Ability to create Linux shellcode in Assembly language
- Understanding this the of a debugger in analyzing binaries

OUTLINE

- Application memory layout
  - Code segment
  - Stack
  - Heap
- Registers
- Assembly language programming
- Linux shellcode
  - Printing to the screen
  - Editing a file
  - Executing a shell
- GDB (Linux GNU Debugger)
  - Overview of using the tool
  - Creating and managing breakpoints
  - Understanding function epilogue and prologue
  - Viewing the stack registers
  - Viewing memory on the stack
  - Finding functions in memory

DAY 3:

LEARNING OBJECTIVES

- Understanding buffer overflows and how to control data on the stack
- Understanding modern protection mechanisms put in place to prevent buffer overflow exploits, and how to defeat those protection
- Ability to analyze a binary, detect possible overflow vulnerabilities, and create a functional exploit

OUTLINE

- SBO (Stack-based Buffer Overflows)
  - Understanding how stack overflows work
  - Finding EIP overwrite location
  - Verifying execution
  - Executing shellcode on the stack
  - Jumping to shellcode in memory
- Overflow protections
  - ASLR (Address Space Layout Randomization)
  - Stack canaries
  - SafeSEH
  - DEP (Data Execution Prevention)

CNO DEVELOPERS CAPABILITIES COURSE

COURSE SCHEDULE

DAY 4:

LEARNING OBJECTIVES

- Understanding techniques used to bypass ASLR and DEP
- Understanding Windows shellcode
- Understanding Windows SBO

OUTLINE

- Defeating DEP and ASLR in *nix
  - ret2libc
  - Bypassing ASLR
- Generating custom shellcode in Windows
  - Hello World program
  - Adding a user
  - Turning off the firewall
  - Loading a new library for additional functionality
- Windows vulnerabilities and exploits
  - Buffer overflows
  - Understanding and executing the stack
  - Jumping to shellcode
  - Finding bad characters in shellcode
  - Determining space available for shellcode

DAY 5:

LEARNING OBJECTIVES

- Understanding Windows SEH (Software Exception Handling)
- Ability to apply knowledge from previous lessons to solving new problems in an exercise

OUTLINE

- Windows SEH
  - SEH chain overview
  - Using SEH chain to control execution
    - Pop Pop Ret
  - Overview of mona Immunity plugin
- CULEX (Culmination Exercise)
  - This is an intense one day exercise that challenged the students' grasp of the concepts discussed throughout the week. In this exercise, there are multiple challenges for the students to complete. A flag is provided for the student upon successful completion of each challenge. A scoreboard tracks the students' progress as they acquire each of the flags.
  - Web attacks
  - Linux local buffer overflow
  - Custom Linux shellcode creation
  - Detecting various protections against buffer overflows
  - Defeating buffer-overflow protections