Buffer Overflow Exploitation for Beginners Training Course

Professional Development Training Course Outline

Category	Data Security	**Duration**	5 Days
Base Fee	$1,500 USD	**Delivery Mode**	Online / On-site Hybrid

Course Introduction

A Buffer Overflow remains one of the most fundamental and critical memory corruption vulnerabilities, persistently appearing in modern software, especially in legacy systems, embedded devices, and IoT firmware. Buffer Overflow Exploitation for Beginners Training Course demystifies this core cybersecurity concept, transitioning participants from passive learners to active exploit developers. You'll gain practical, hands-on experience with the entire exploit lifecycle, from initial fuzzing and vulnerability analysis to crafting a functional shellcode and achieving Remote Code Execution (RCE) on vulnerable x86 architectures.

This intensive, lab-focused training goes beyond theory, focusing on the practical methodology essential for roles like Penetration Tester and Security Researcher. It provides the OSCP-relevant skills needed to master classic stack-based buffer overflows in a controlled lab environment using industry-standard tools like Immunity Debugger, Mona.py, and Kali Linux. Understanding this vulnerability is a non-negotiable prerequisite for tackling more complex binary exploitation and is the bedrock for careers in red teaming and vulnerability management. The course concludes with an essential look at modern mitigations like ASLR and DEP, ensuring a complete picture of contemporary software security.

Programme Curriculum

Buffer Overflow Exploitation for Beginners Training Course

Introduction

A Buffer Overflow remains one of the most fundamental and critical memory corruption vulnerabilities, persistently appearing in modern software, especially in legacy systems, embedded devices, and IoT firmware. Buffer Overflow Exploitation for Beginners Training Course demystifies this core cybersecurity concept, transitioning participants from passive learners to active exploit developers. You'll gain practical, hands-on experience with the entire exploit lifecycle, from initial fuzzing and vulnerability analysis to crafting a functional

shellcode and achieving Remote Code Execution (RCE) on vulnerable x86 architectures.

This intensive, lab-focused training goes beyond theory, focusing on the practical methodology essential for roles like Penetration Tester and Security Researcher. It provides the OSCP-relevant skills needed to master classic stack-based buffer overflows in a controlled lab environment using industry-standard tools like Immunity Debugger, Mona.py, and Kali Linux. Understanding this vulnerability is a non-negotiable prerequisite for tackling more complex binary exploitation and is the bedrock for careers in red teaming and vulnerability management. The course concludes with an essential look at modern mitigations like ASLR and DEP, ensuring a complete picture of contemporary software security.

Course Duration

5 days

Course Objectives

1. Understand the structure of an x86 program's memory, including the Stack, Heap, and BSS segments.
2. Clearly define the mechanism and impact of stack-based buffer overflow vulnerabilities.
3. Interpret basic x86 Assembly instructions and register functions relevant to execution flow control.
4. Perform methodical fuzzing to crash a vulnerable application and reliably determine the exact offset to the instruction pointer
5. Master the technique of overwriting the EIP to hijack the program's control flow.
6. Identify and eliminate bad characters from the payload to ensure successful exploit execution.
7. Generate reliable, stage-one reverse shell or bind shell payloads using Metasploit's msfvenom.
8. Apply basic encoding techniques to bypass trivial signature-based detection.
9. Introduce the concept of Structured Exception Handler (SEH) Overwrite as an advanced exploitation technique.
10. Develop robust, automated exploit scripts using Python for reliable payload delivery and RCE.
11. Explain the purpose of key memory-based security mitigations.
12. Proficiently set up the essential exploit development environment
13. Recommend effective secure coding practices in C/C++ to prevent buffer overflow vulnerabilities.

Target Audience

1. Aspiring Penetration Testers.
2. Cybersecurity Beginners.
3. Junior Security Analysts.
4. Developers/Software Engineers.
5. Capture The Flag Players.
6. System Administrators.
7. Computer Science Graduates.
8. IT Professionals.

Course Modules

Module 1: x86 Memory and Stack Fundamentals

- The Program's Memory Layout.
- x86 Registers Explained.
- Function Calls and the Call Stack.
- Visual demonstration of how a larger-than-expected input overwrites adjacent memory.

- Case Study: Analyzing a simple C program with the vulnerable gets function to observe the crash.

Module 2: Setting up the Exploitation Lab

- Installation and configuration of Kali Linux and a dedicated Windows x86 target machine.
- Introduction to Immunity Debugger and its key functionalities
- Installing the essential Mona.py script for automation in exploit development.
- Installation and configuration of a deliberately vulnerable application
- Case Study: Using a Python socket program to connect to the target and send initial test inputs to confirm connectivity.

Module 3: Fuzzing and Finding the Crash

- Introduction to Fuzzing.
- Fuzzing Script Development
- Identifying the Crash
- Case Study: The TRUN command in VulnServer fuzzing it to find the crash and initial EIP offset value.

Module 4: Controlling the Instruction Pointer (EIP)

- Pattern Generation.
- Exact Offset Determination.
- Overwriting EIP.
- Verifying Control
- Case Study: Successfully hijacking execution flow in the target application and proving EIP control is achieved.

Module 5: Finding a JMP ESP Address

- Return Address Overwrite.
- Mona.py for Modules
- Searching for an Instruction.
- Byte Order
- Case Study: Finding a reliable JMP ESP address in a common vulnerable DLL and integrating it into the exploit script.

Module 6: Bad Characters and Shellcode Generation

- Bad Character Concept
- Bad Character Fuzzing.
- Msfvenom Shellcode.
- Explaining and implementing a NOP Sled to increase the exploit's reliability.
- Case Study: Generating a Windows reverse shell payload with the EXITFUNC thread option and meticulously removing bad characters.

Module 7: Final Exploit and RCE

- The Exploit Structure.
- Gaining a Foothold
- Post-Exploitation Basics.
- Common beginner errors and debugging steps.

- Case Study: Achieving a full SYSTEM level shell on the target Windows machine using the crafted exploit.

Module 8: Advanced Topics and Modern Mitigations

- Introduction to Heap Overflows.
- ASLR and DEP Explained.
- Stack Canaries.
- Mitigation Bypass Concepts.
- Case Study: Discussing a recent Common Vulnerabilities and Exposures that involved a stack buffer overflow and its real-world impact.

Training Methodology

This course employs a participatory and hands-on approach to ensure practical learning, including:

- Interactive lectures and presentations.
- Group discussions and brainstorming sessions.
- Hands-on exercises using real-world datasets.
- Role-playing and scenario-based simulations.
- Analysis of case studies to bridge theory and practice.
- Peer-to-peer learning and networking.
- Expert-led Q&A sessions.
- Continuous feedback and personalized guidance.

Register as a group from 3 participants for a Discount

Send us an email: info@fineskilltrainingcenter.org or call +254769199797

Certification

Upon successful completion of this training, participants will be issued with a globally- recognized certificate.

Tailor-Made Course

We also offer tailor-made courses based on your needs.

Key Notes

a. The participant must be conversant with English.

b. Upon completion of training the participant will be issued with an Authorized Training Certificate

c. Course duration is flexible and the contents can be modified to fit any number of days.

d. The course fee includes facilitation training materials, 2 coffee breaks, buffet lunch and A Certificate upon successful completion of Training.

e. One-year post-training support Consultation and Coaching provided after the course.

f. Payment should be done at least a week before commence of the training, to Fineskill Training Center account, as indicated in the invoice so as to enable us prepare better for you.

Upcoming Scheduled Sessions

Start Date	End Date	Location	Price
21 Sep 2026	25 Sep 2026	Online	$1,000
21 Sep 2026	25 Sep 2026	Physical	$1,500
21 Sep 2026	25 Sep 2026	Physical	$0
21 Sep 2026	25 Sep 2026	Physical	$0
21 Sep 2026	25 Sep 2026	Physical	$0
21 Sep 2026	25 Sep 2026	Physical	$0
21 Sep 2026	25 Sep 2026	Physical	$0
21 Sep 2026	25 Sep 2026	Physical	$0

Ready to enroll or request a customized program? Book online or contact us:

Register Online Now

Email: info@fineskilltrainingcenter.com | Website: www.fineskilltrainingcenter.com