SYLLABUS

EXPLOIT DEVELOPMENT STUDENT VERSION 1

The most practical and comprehensive training course on exploit development

eLearnSecurity
Forging security professionals

eLearnSecurity has been chosen by students in over 140 countries in the world and by leading organizations such as:

hp Microsoft intel cisco AT&T verizon U.S. AIR FORCE pwc gemalto security to be free

COURSE GOALS

The eXploit Development Student course (XDS) is an online, self-paced training course built for anyone with little to no background in exploit development.

XDS is the most comprehensive and practical online course on exploit development, since it provides not only the fundamentals of Windows and Linux exploit development but also covers advanced Windows and Linux exploit development techniques, as well as anti-exploit mechanism bypasses.

Specifically, the XDS course:

- Is based on techniques professional exploit developers use
- Covers software debugging
- Provides a methodology on how to identify and fully exploit 0-day vulnerabilities
- Thoroughly covers the fundamentals of Windows and Linux exploit development as well as advanced binary exploitation techniques
- Covers bypassing modern anti-exploit mechanisms
- Covers exploit development and shellcoding on 64-bit architectures
- Shows how to effectively use tools like Immunity Debugger, x32dbg, Mona, Pwntools, GDB, Ropper, etc.
- Is extremely Hands-on with 19 labs and exercises
- Leads to the eCXD certification, which qualifies you for 40 CPE
- Is ideal for pentesters who want to advance their career

Thanks to the extensive use of Hera Lab, every knowledge gained in XDS will be reinforced by practical exercises. The practical nature of XDS ensures that you will be able to perform thorough and in-depth binary exploitation upon completing the course.

COURSE ORGANIZATION

This training course is self-paced with interactive slides and video material that students can access online without any limitation. Students have lifetime access to the training material.

Students can study from home, the office, or wherever an Internet connection is available.

It is always possible to resume studying from the last slide or video accessed.

The eXploit Development Student course is integrated with Hera Lab, the most sophisticated virtual lab in IT Security. A minimum amount of 60 hours is advised. For more intensive use, 120 hours may be necessary. Hera Lab provides on-demand vulnerable binaries/executables, where a student can practice every exploit development technique seen in the course in a dedicated and isolated environment.

TARGET AUDIENCE AND PRE-REQUISITES

The XDS training course benefits the careers of penetration testers and IT security personnel in charge of defending their organization's software.

This course allows organizations of all sizes to assess and mitigate the risk at which their software is exposed by building strong, practical in-house skills.

Penetration testing companies can train their teams with a comprehensive and practical training course without having to deploy internal labs that are often outdated and not backed by solid theoretical material.

Vulnerability researchers and CTF/Bug Bounty players can also benefit from XDS.

Students willing to enroll in the course must possess a solid understanding of:

- Windows and Linux internals
- Programming concepts
- Assembly, C/C++ and Python basics
- Penetration Testing

WILL I GET A CERTIFICATE?

The XDS course leads to the eCXD certification.

eLearnSecurity's eCXD (eLearnSecurity Certified eXploit Developer) certification is the most practical and professionally oriented certification you can obtain in exploit development.

The certification can be obtained by successfully completing the requirements of a 100% practical exam where actual vulnerability identification and exploitation against real-world Windows and Linux software should take place.

The exam engagement is modeled after real-world software exploitation scenarios featuring anti-exploit mechanisms as well as other limitations.

ORGANIZATION OF CONTENTS

SECTION 1: LINUX EXPLOIT DEVELOPMENT

This section is comprised of 5 modules:

- Module 1: Linux Stack Smashing
- Module 2: Linux Exploit Countermeasures & Bypasses
- Module 3: Linux Return Oriented Programming
- Module 4: Linux Shellcoding
- Module 5: Linux Advanced Exploitation

Several labs accompany this section, and each comes with an extensive PDF manual that will first introduce students to the tasks of each lab and then guide them on how each task can be solved, in the solutions section. Specifically, this section includes the following labs:

- Lab: Hidden Function [Category: Practical]
- Lab: Linux Basic Stack Overflow [Category: Practical]
- Lab: Linux x64 Basic Stack Overflow [Category: Practical]
- Lab: Strict Firewall Bypass (Format String Exploitation + Socket Reuse Shellcode) [Category: Educational]
- Lab: Linux NX Bypass (ret2libc) [Category: Educational]
- Lab: Linux x64 NX Bypass (ret2libc + ROP) [Category: Educational]
- Lab: Linux NX & ASLR Bypass (Format String Exploitation + ROP) [Category: Educational
- Lab: Linux Shellcoding [Category: Practical]
- Lab: Overcome ret2libc Limitations [Category: Educational]
- Lab: Linux x64 Stack Canary, NX & ASLR Bypass [Category: Educational]
- Lab: Linux x64 ASLR Bypass [Category: Educational]

SECTION 2: WINDOWS EXPLOIT DEVELOPMENT

This section is comprised of 6 modules:

- Module 1: Windows Stack Smashing
- Module 2: Windows SEH-based Overflows
- Module 3: Windows Egghunting
- Module 4: Unicode Buffer Overflows
- Module 5: Windows Shellcoding
- Module 6: Windows Return Oriented Programming

Several labs accompany this section, and each comes with an extensive PDF manual that will first introduce students to the tasks of each lab and then guide them on how each task can be solved, in the solutions section. Specifically, this section includes the following labs:

- Lab: Windows Basic Stack Overflow [Category: Practical]
- Lab: Windows SEH Overflow (MP3 Studio) [Category: Practical]
- Lab: Windows SEH Overflow (EasyChat) [Category: Practical]
- Lab: Windows Egghunting (Kolibri HTTP Server) [Category: Practical]
- Lab: Windows Shellcoding [Category: Practical]
- Lab: Fuzzing Windows Software [Category: Practical]
- Lab: Windows ROP (Scenario 1) [Category: Educational]
- Lab: Windows ROP (Scenario 2) [Category: Educational]

MODULE 1: LINUX STACK SMASHING

This module introduces students to the basics of Linux stack overflow vulnerabilities and the required debugging toolset. Everything is covered through practical examples, from crashing a binary and identifying a stack overflow vulnerability all the way to executing user-supplied shellcode. Important Linux fundamentals that will prove useful during the rest of the course are also provided to students.

1. Linux Stack Smashing
- **1.1. Introduction to Linux Exploitation**
  - 1.1.1. ELF Fundamentals
    - 1.1.1.1. GOT & PLT
    - 1.1.1.2. SUID & SGID
- **1.2. Linux Stack Smashing**
- **1.3. Abusing the EIP Control**
  - 1.3.1. Code Reuse Through EIP Control
  - 1.3.2. Shellcode Fundamentals
  - 1.3.3. From EIP Control to Code Execution
  - 1.3.4. Debugging Common Obstacles

MODULE 2: LINUX EXPLOIT COUNTERMEASURES & BYPASSES

This module explains the most common Linux exploit mitigations related to stack overflow exploitation, as well as the methods to bypass them. Specifically, ASLR, NX, Stack Cookie, RELRO and other exploit mitigations are covered alongside techniques to bypass them.

2. Linux Exploit Countermeasures & Bypasses
- **2.1. Linux Exploit Protections**
- **2.2. NoExecute**
  - 2.2.1. Bypassing NX (ret2libc)
  - 2.2.2. NX Bypass Example
- **2.3. ASLR**
  - 2.3.1. Bypassing ASLR (abusing low ASLR entropy)
  - 2.3.2. ASLR Bypass Example

2.4. Stack Cookie
- 2.4.1. Bypassing Stack Cookie
- 2.4.2. Stack Cookie Bypass Example

2.5. RELRO
- 2.5.1. Bypassing RELRO
- 2.5.2. RELRO Bypass Example

2.6. Other Protections

MODULE 3: LINUX RETURN ORIENTED PROGRAMMING

This module explains the concept of Return Oriented Programming and how it can be used to bypass (even combined) anti-exploit mechanisms on Linux systems. After studying the provided practical examples and labs, students will be able to craft their own ROP chains.

Module 3 lays the foundational knowledge of ROP and you will be able to practice and deepen your ROP knowledge in the labs of the final module of this section, Module 5 – Linux Advanced Exploitation.

3. Linux Return Oriented Programming
- **3.1. ROP Theory**
- **3.2. ROP Theoretical Example**
- **3.3. ROP Exploitation Example**

MODULE 4: LINUX SHELLCODING

This module teaches the process of writing Linux shellcode from scratch, including cases such as egghunting, encoding, etc.

4. Linux Shellcoding
- **4.1. x86 Assembly Basics**
- **4.2. Basic Linux Shellcode**
- **4.3. Reverse TCP Shellcode**
- **4.4. x64 Architecture**
- **4.5. Writing x64 Shellcode**

MODULE 5: LINUX ADVANCED EXPLOITATION

This module introduces Format String vulnerabilities and exploitation as well as exploit development on hardened Linux 64-bit systems. Bypassing (even combined) exploit countermeasures on Linux x64 systems, and advanced Linux x64 exploit development techniques are covered through real-world labs.

5. Introduction to Programming
- **5.1. Format String Vulnerabilities**
- **5.2. Format String Exploitation**
- **5.3. Exploitation on 64-bit Linux**

MODULE 1: WINDOWS STACK SMASHING

This module shows the basics of smashing the stack on Windows systems and presents the differences from Linux stack overflow exploitation in terms of the approach and the used toolset.

1. Windows Stack Smashing
- **1.1. Windows Stack Overflow**
- **1.2. Windows Basic Overflow Analysis**
  - 1.2.1. Basic Overflow – Crashing the Application
  - 1.2.2. Basic Overflow – Calculating the Offset
  - 1.2.3. Basic Overflow – Jumping to the Buffer
  - 1.2.4. Basic Overflow – Bad Characters
    - 1.2.4.1. Basic Overflow – Detecting Bad Characters
    - 1.2.4.2. Basic Overflow – Common Bad Characters
  - 1.2.5. Basic Overflow – Implementing Shellcode

MODULE 2: WINDOWS SEH-BASED OVERFLOWS

This module shows exploitation of Windows executables that abuses the Structured Exception Handling mechanism. Through a practical example, students will be shown the process of going from overwriting the SEH to jumping to shellcode.

2. Windows SEH-based Overflows
- **2.1. Structured Exception Handling**
- **2.2. SEH Practical Example**
  - 2.2.1. SEH Crashing and Debugging
  - 2.2.2. SEH – Calculating the Offset
  - 2.2.3. SEH – Finding POP POP RET
  - 2.2.4. SEH – Detecting Bad Characters
  - 2.2.5. SEH – Adding Shellcode
    - 2.2.5.1. SEH – Exploit Code

MODULE 3: WINDOWS EGGHUNTING

This module introduces the Egghunter shellcode to students. The Egghunter shellcode can assist exploitation when there is limited buffer space for placing user-supplied shellcode. Deliver shellcode through alternative channels is also covered in this module.

3. Windows Egghunting
- **3.1. Egghunter Exploits**
- **3.2. Vulnserver Egghunter Exploit**
  - 3.2.1. Vulnserver Debugging
- **Video Series: Socket Reuse Shellcode Parts 1 – 3**

MODULE 4: UNICODE BUFFER OVERFLOWS

This module shows the process of creating venetian shellcode and utilizing it in Unicode Buffer Overflow exploits. This technique is useful if the application changes user input in case of translation.

4. Unicode Buffer Overflows
- **4.1. Character Transformation**
  - 4.1.1. Unicode Format
  - 4.1.2. The Problem with Exploiting Unicode
  - 4.1.3. Venetian Shellcode
  - 4.1.4. Unicode and SEH
  - 4.1.5. Unicode Exploitation Strategy
- **4.2. Unicode Exploitation Case Study**

MODULE 5: WINDOWS SHELLCODING

This module teaches the process of writing Windows shellcode from scratch, including advanced, address-aware shellcode.

5. Windows Shellcoding
- **5.1. Basic Windows Shellcode**
  - 5.1.1. Machine Language

5.1.2. Calling Windows Functions
5.1.2.1. Dynamic Link Libraries
5.1.3. Within the Process Address Space
5.1.4. Order of Arguments
5.1.4.1. The Problem with Strings
5.1.5. Writing Shellcode
5.1.5.1. Testing Shellcode
5.1.5.2. Basic Shellcode
5.1.5.3. A Graceful Exit

5.2. Universal Shellcode

5.2.1. Structured System
5.2.2. Traversing the Metadata
5.2.3. Function Base Address
5.2.3.1. Traversing More Metadata
5.2.3.2. Mitigating Single Null-bytes
5.2.3.3. Searching for the Function
5.2.3.4. Finalizing the Shellcode
5.2.4. Source Code

Video Series: Backdooring PE Files Parts 1 – 3

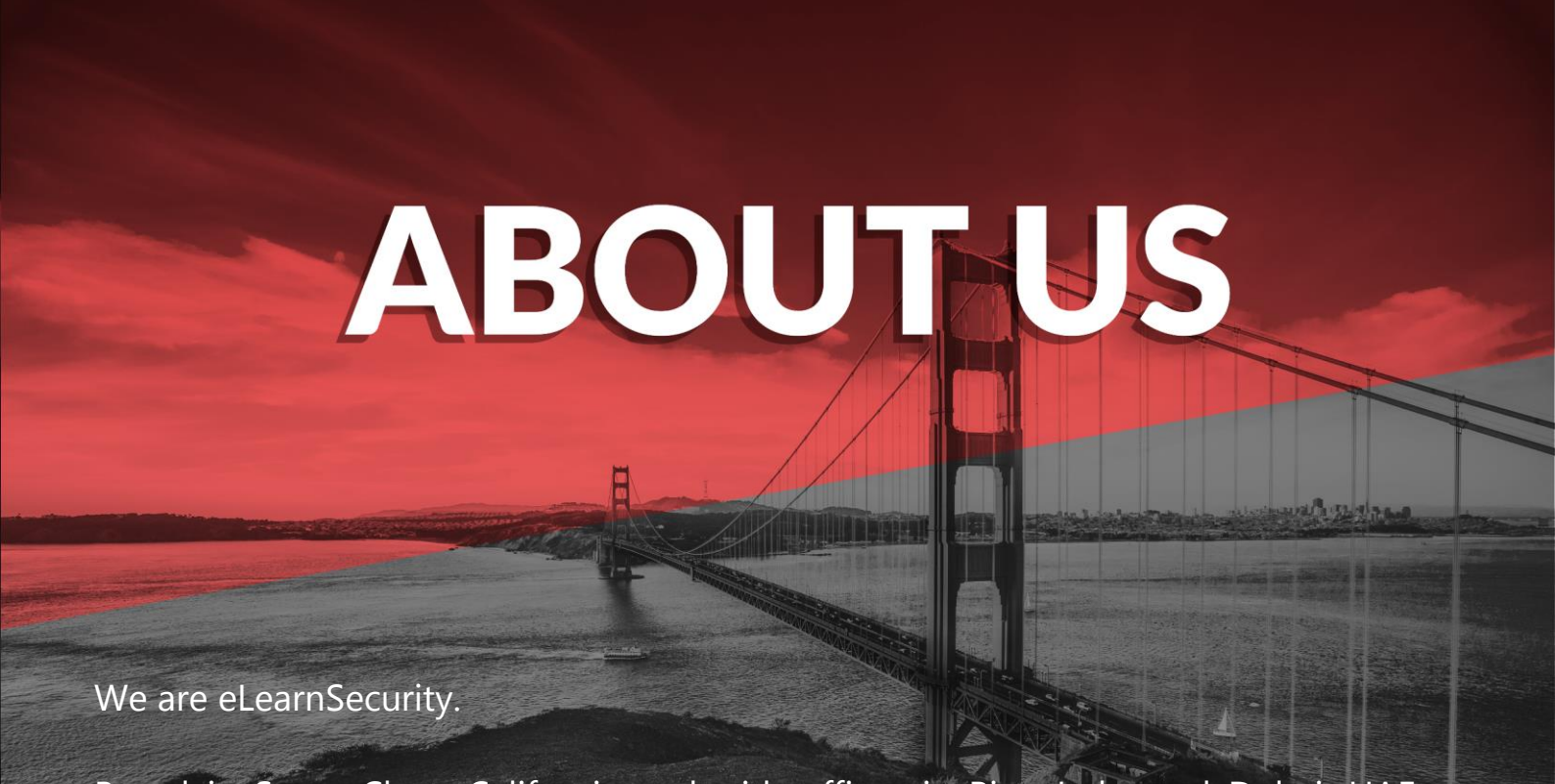
MODULE 6: WINDOWS RETURN ORIENTED PROGRAMMING

This module introduces Return Oriented Programming on Windows and how it can be used to beat anti-exploit mechanisms on Windows 7. After studying the provided practical examples and labs, students will be able to craft their own ROP chains.

6. Windows Return Oriented Programming

6.1. Windows Exploit Countermeasures
6.2. Creating ROP with mona.py

ABOUT US



We are eLearnSecurity.

Based in Santa Clara, California and with offices in Pisa, Italy, and Dubai, UAE, Caendra Inc. is a trusted source of IT security skills for IT professionals and corporations of all sizes. Caendra Inc. is the Silicon Valley-based company behind the eLearnSecurity brand.

eLearnSecurity has proven to be a leading innovator in the field of practical security training with best of breed virtualization technology, in-house projects such as Coliseum Web Application Security Framework and Hera Network Security Lab, which has changed the way students learn and practice new skills.

Contact details:

www.elearnsecurity.com
contactus@elearnsecurity.com

2040 Martin Ave.
Santa Clara, CA, USA

Via Gian Battista Queirolo
Pisa, Italy

Apricot Tower, Dubai Silicon Oasis
Dubai, UAE