SEC660: Advanced Penetration Testing, Exploit Writing, and Ethical Hacking

GXPN
Exploit Researcher & Advanced Pen Tester
giac.org/gxpn

6 Day Program | 46 CPEs | Laptop Required

You Will Be Able To

- Perform fuzz testing to enhance your company's SDL process
- Exploit network devices and assess network application protocols
- Escape from restricted environments on Linux and Windows
- Test cryptographic implementations
- Model the techniques used by attackers to perform 0-day vulnerability discovery and exploit development
- Develop more accurate quantitative and qualitative risk assessments through validation
- Demonstrate the needs and effects of leveraging modern exploit mitigation controls
- Reverse-engineer vulnerable code to write custom exploits

GXPN
Exploit Researcher & Advanced Pen Tester
giac.org/gxpn

GIAC Exploit Researcher and Advanced Penetration Tester

The GIAC Exploit Researcher and Advanced Penetration Tester certification validates a practitioner's ability to find and mitigate significant security flaws in systems and networks. GXPN certification holders have the skills to conduct advanced penetration tests and model the behavior of attackers to improve system security, and the knowledge to demonstrate the business risk associated with these behaviors.

- Network Attacks, Crypto, Network Booting, and Restricted Environments
- Python, Scapy, and Fuzzing
- Exploiting Windows and Linux for Penetration Testers

This course is designed as a logical progression point for those who have completed SEC560: Network Penetration Testing and Ethical Hacking, or for those with existing penetration testing experience. Students with the prerequisite knowledge to take this course will walk through dozens of real-world attacks used by the most seasoned penetration testers. The methodology of a given attack is discussed, followed by exercises in a real-world lab environment to solidify advanced concepts and allow for the immediate application of techniques in the workplace. Each day includes a two-hour evening bootcamp to allow for additional mastery of the techniques discussed and even more hands-on exercises. A sample of topics covered includes weaponizing Python for penetration testers, attacks against network access control (NAC) and VLAN manipulation, network device exploitation, breaking out of Linux and Windows restricted environments, IPv6, Linux privilege escalation and exploit-writing, testing cryptographic implementations, fuzzing, defeating modern OS controls such as ASLR and DEP, return-oriented programming (ROP), Windows exploit-writing, and much more!

Attackers are becoming more clever and their attacks more complex. In order to keep up with the latest attack methods, you need a strong desire to learn, the support of others, and the opportunity to practice and build experience. SEC660 provides attendees with in-depth knowledge of the most prominent and powerful attack vectors and an environment to perform these attacks in numerous hands-on scenarios. This course goes far beyond simple scanning for low-hanging fruit, and shows penetration testers how to model the abilities of an advanced attacker to find significant flaws in a target environment and demonstrate the business risk associated with these flaws.

SEC660 starts off by introducing the advanced penetration concept, and provides an overview to help prepare students for what lies ahead. The focus of section one is on network attacks, an area often left untouched by testers. Topics include accessing, manipulating, and exploiting the network. Attacks are performed against NAC, VLANs, OSPF, 802.1X, CDP, IPv6, VOIP, SSL, ARP, SNMP, and others. Section two starts off with a technical module on performing penetration testing against various cryptographic implementations. The rest of the section is spent on network booting attacks, escaping Linux restricted environments such as chroot, and escaping Windows restricted desktop environments. Section three jumps into an introduction of Python for penetration testing, Scapy for packet crafting, product security testing, network and application fuzzing, and code coverage techniques. Sections four and five are spent exploiting programs on the Linux and Windows operating systems. You will learn to identify privileged programs, redirect the execution of code, reverse-engineer programs to locate vulnerable code, obtain code execution for administrative shell access, and defeat modern operating system controls such as ASLR, canaries, and DEP using ROP and other techniques. Local and remote exploits, as well as client-side exploitation techniques, are covered. The final course section is dedicated to numerous penetration testing challenges requiring you to solve complex problems and capture flags.

Among the biggest benefits of SEC660 is the expert-level hands-on guidance provided through the labs and the additional time allotted each evening to reinforce daytime material and master the exercises.

"SEC660 is the right balance between theory and practice; it's hands-on, not too hard, but also not too easy."

— Anton Ebertzeder, **Siemens AG**

Section Descriptions

SECTION 1: Network Attacks for Penetration Testers

Section 1 serves as an advanced network attack module, building on knowledge gained from SEC560. The focus will be on obtaining access to the network; manipulating the network to gain an attack position for eavesdropping and attacks, and for exploiting network devices; leveraging weaknesses in network infrastructure; and taking advantage of client frailty.

TOPICS: Bypassing Network Access/Admission Control (NAC); Impersonating Devices with Admission Control Policy Exceptions; Exploiting EAP-MD5 Authentication; Custom Network Protocol Manipulation with Ettercap and Custom Filters; Multiple Techniques for Gaining Man-in-the-Middle Network Access; IPv6 for Penetration Testers; Exploiting OSPF Authentication to Inject Malicious Routing Updates; Using Evilgrade to Attack Software Updates; Overcoming SSL Transport Encryption Security with Sslstrip; Remote Cisco Router Configuration File Retrieval

SECTION 2: Crypto and Post-Exploitation

Section 2 starts by taking a tactical look at techniques that penetration testers can use to investigate and exploit common cryptography mistakes. We begin by building some fundamental knowledge on how ciphers operate, without getting bogged down in complex mathematics. Then we move on to techniques for identifying, assessing, and attacking real-world crypto implementations. The day continues with advanced techniques but focuses more on post exploitation tasks. As a major part of post exploitation, we cover PowerShell: including basic concepts and tasks, enterprise tasks, and outright offensive tasks. We will discuss and use a variety of PowerShell attack tools to discover vulnerabilities and gain privileges. The day ends with a challenging boot camp exercise against a full network environment comprised of a variety of modern, representative, and fully patched systems with no obvious external vulnerabilities.

TOPICS: Pen Testing Cryptographic Implementations; Exploiting CBC Bit Flipping Vulnerabilities; Exploiting Hash Length Extension Vulnerabilities; PowerShell Essentials; Enterprise PowerShell; Post-Exploitation with PowerShell and Metasploit; Escaping Software Restrictions; Two-hour Evening Capture-the-Flag Exercise Against a Modern Network with Hardened Servers, Desktops and vApp Targets

Who Should Attend

- Network and systems penetration testers
- Incident handlers
- Application developers
- IDS engineers

"The exploit development taught in SEC660 is a very useful, niche, hard to find and learn skill—which makes it very valuable."

— Christian Nicholson, **KPMG**

SECTION 3: Python, Scapy, and Fuzzing

Section 3 brings together the multiple skill sets needed for creative analysis in penetration testing. We start by discussing product security testing. The day continues with a focus on how to leverage Python as a penetration tester - the aim is to help students unfamiliar with Python start modifying scripts to add their own functionality, while also helping seasoned Python scripters improve their skills. Once we leverage the Python skills in creative lab exercises, we move on to leveraging Scapy for custom network targeting and protocol manipulation. Using Scapy, we examine techniques for transmitting and receiving network traffic beyond what canned tools can accomplish, including IPv6. Next, we take a look at network protocol and file format fuzzing. We leverage fuzzing to target both common network protocols and popular file formats for bug discovery. We use hands-on exercises to develop custom protocol fuzzing grammars to discover bugs in popular software. Finally, we carefully discuss the concept of code coverage and how it goes hand-in-hand with fuzzing. We will conduct a lab using the Paimei Reverse Engineering Framework and IDA Pro to demonstrate the techniques discussed.

TOPICS: Becoming Familiar with Python Types; Leveraging Python Modules for Real-World Pen Tester Tasks; Manipulating Stateful Protocols with Scapy; Using Scapy to Create a Custom Wireless Data Leakage Tool; Product Security Testing; Using Taof for Quick Protocol Mutation Fuzzing; Optimizing Your Fuzzing Time with Smart Target Selection; Automating Target Monitoring While Fuzzing with Sulley; Leveraging Microsoft Word Macros for Fuzzing .docx files; Block-Based Code Coverage Techniques Using Paimei

SECTION 4: Exploiting Linux for Penetration Testers

Section 4 begins by walking through memory from an exploitation perspective as well as introducing x86 assembler and linking and loading. These topics are important for anyone performing penetration testing at an advanced level. Processor registers are directly manipulated by testers and must be intimately understood. Disassembly is a critical piece of testing and will be used throughout the remainder of the course. We will take a look at the Linux OS from an exploitation perspective and discuss privilege escalation. We continue by describing how to look for SUID programs and other likely points of vulnerabilities and misconfigurations. The material will focus on techniques that are critical to performing penetration testing on Linux applications. We then go heavily into stack overflows on Linux to gain privilege escalation and code execution. We will first cover using a debugger to expose weak passwords. Then we will go over redirection of program execution and, finally, code execution. Techniques such as return to buffer and return to C library (ret2libc) will be covered, as well as an introduction to return-oriented programming. The remainder of the day takes students through techniques used to defeat or bypass OS protections such as stack canaries and address space layout randomization (ASLR). The goal of this section is to expose students to common obstacles on modern Linux-based systems

TOPICS: Stack and Dynamic Memory Management and Allocation on the Linux OS; Disassembling a Binary and Analyzing x86 Assembly Code; Performing Symbol Resolution on the Linux OS; Identifying Vulnerable Programs; Code Execution Redirection and Memory Leaks; Identifying and Analyzing Stack-Based Overflows on the Linux OS; Performing Return-to-libc (ret2libc) Attacks on the Stack; Return-Oriented Programming; Defeating Stack Protection on the Linux OS; Defeating ASLR on the Linux OS

SECTION 5: Exploiting Windows for Penetration Testers

Section 5 starts off covering the OS security features (ASLR, DEP, etc.) added to the Windows OS over the years as well as Windows-specific constructs. Differences between Linux and Windows will be covered. These topics are critical in assessing Windows-based applications. We then focus on stack-based attacks against programs running on the Windows OS. After finding a vulnerability in an application, the student will work with Immunity Debugger to turn the bug into an opportunity for code execution and privilege escalation. Advanced stack-based techniques such as disabling data execution prevention (DEP) are covered. Client-side exploitation will be introduced, as it is a highly common area of attack. We continue with the topic of return-oriented programming (ROP), demonstrating the technique against a vulnerable application, while looking at defeating hardware DEP and address space layout randomization (ASLR) on Windows 7, Windows 8, and Windows 10. We then have a module on porting over an exploit into the Metasploit Framework and on how to quickly identify bad characters in your shellcode and as input into a program. Finally, we will take a quick look at shellcode and the differences between shellcode on Linux and Windows, followed by a ROP challenge.

TOPICS: The State of Windows OS Protections on Windows 7, 8, 10, Server 2008 and 2012; Understanding Common Windows Constructs; Stack Exploitation on Windows; Defeating OS Protections Added to Windows; Creating a Metasploit Module; Advanced Stack-Smashing on Windows; Using ROP; Building ROP Chains to Defeat DEP and Bypass ASLR; Windows 7 and 8; Porting Metasploit Modules; Client-side Exploitation; Windows Shellcode

SECTION 6: Capture-the-Flag Challenge

This section will serve as a real-world challenge for students by requiring them to utilize skills they have learned throughout the course, think outside the box, and solve a range of problems from simple to complex. A web server scoring system and Capture-the-Flag engine will be provided to score students as they capture flags. More difficult challenges will be worth more points. In this offensive exercise, challenges range from local privilege escalation to remote exploitation on both Linux and Windows systems, as well as networking attacks and other challenges related to the course material.