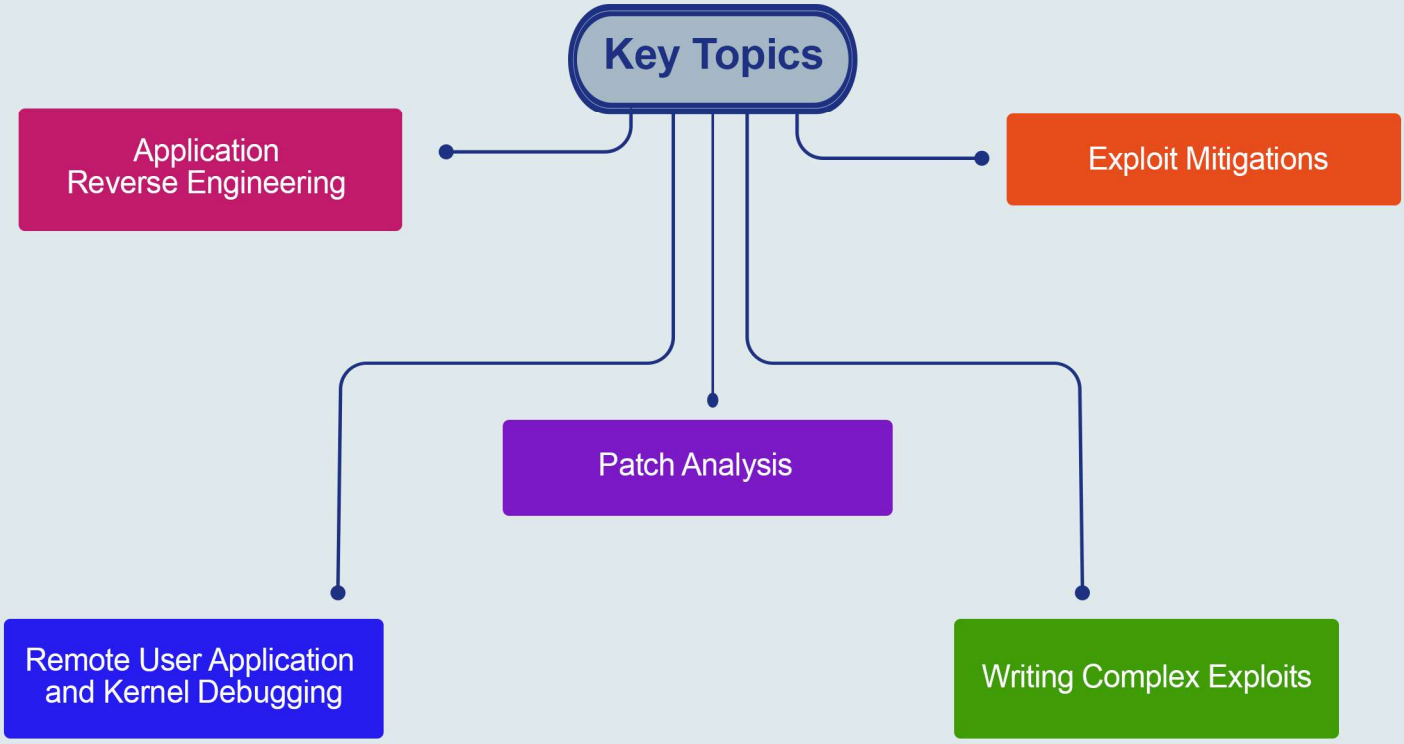
About the Course

This course will provide you with the advanced skills to improve your exploit development and understand vulnerabilities beyond a fundamental level. The course teaches you cutting-edge tricks to thoroughly evaluate a target, and defend against even the most skilled attackers. Few security professionals have the skillset to discover why a complex vulnerability exists and how to write an exploit to compromise it. Conversely, attackers must maintain this skillset regardless of the increased complexity. This course teaches the skills required to reverse-engineer applications to find vulnerabilities, perform remote user application and kernel debugging, analyze patches for one-day exploits, and write complex exploits such as use-after-free attacks against modern software and operating systems.

Learning Objectives

Key Topics



Pre Requisites

A basic understanding of Python language. it is also necessary to take the following courses:

- InformationSecurity Concepts & Principles
- Ethical Hacking & Penetration Testing Concepts

Terminus System

What You Will Receive

- Course Presentation File
- Complementary Files
- Exploit Development eBook

Who Should Attend

Anybody who is in charge of security assessment and penetration testing and also:

- CISOs
- PenTesters
- Threat & Vulnerability Analysts
- Security Specialists

Syllabus

Fundamentals of Reverse Engineering

- Introduction
- Debuggers
- Assembly Language

Python Language

- Python Essentials
- Using Python
- Python Offensive
- Python, Scapy, and Fuzzing

Exploiting Linux

- Stack memory management and allocation on the Linux OS
- Disassembling a binary and analyzing x86/x86-64 assembly code
- Performing symbol resolution on the Linux OS
- Identifying vulnerable programs
- Code execution redirection
- 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
- Linux heap management, constructs, and environment
- Navigating the heap
- Abusing macros such as unlink() and frontlink()
- Function pointer overwrites
- Format string exploitation
- Defeating Linux exploit mitigation controls
- Using IDA remote debugging for Linux application exploitation
- Using format string bugs for ASLR bypass

Syllabus

Exploiting Windows

- The state of Windows OS protections on the Windows OS
- 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 10 exploitation
- Client-side exploitation
- Windows Shellcode

Windows Kernel Debugging and Exploitation

- Understanding the Windows kernel
- Navigating the Windows kernel
- Modern kernel protections
- Debugging the Windows 10 kernels and drivers
- WinDbg
- Analyzing kernel vulnerabilities and vulnerability types
- Kernel exploitation techniques
- Token stealing and information disclosure vulnerabilities

Advanced Windows Exploitation

- Windows heap management, constructs, and environment
- Understanding the low fragmentation heap
- Browser-based and client-side exploitation
- Understanding C++ vftable/vtable behavior
- Use-After-Free attacks and dangling pointers
- Avoiding protections such as MemGC and Isolated Heap
- Dealing with ASLR, DEP, and other common exploit mitigation controls

Syllabus

Exploiting Stack Overflows

- Stack Overflows Introduction
- Installing the Sync Breeze Application
- Crashing the Sync Breeze Application
- Win32 Buffer Overflow Exploitation

Exploiting SEH Overflows

- Installing the Sync Breeze Application
- Crashing Sync Breeze
- Analyzing the Crash in WinDbg
- Introduction to Structured Exception Handling
- Structured Exception Handler Overflows

Exploit Mitigations and Reversing with IDA

- Exploit mitigations
- Windows Defender Exploit Guard
- Introduction to IDA Pro
- Debugging with IDA Pro
- FLIRT & FLAIR
- Scripting with IDAPython and Python 3

Reverse Shell

- Php reverse shell
- Python reverse shell
- Perl reverse shell
- Bash reverse shell
- Msfvenom shell

Syllabus

Patch Diffing, One-Day Exploits, and Return-Oriented Shellcode

- The Microsoft patch management process and Patch Tuesday
- Obtaining patches and patch extraction
- Binary diffing with BinDiff 5
- Visualizing code changes and identifying fixes
- Reversing 32-bit and 64-bit applications and modules
- Triggering patched vulnerabilities
- Writing one-day exploits
- Using ROP to compiled shellcode on the fly (Return-Oriented Shellcode)

Hands-on Training

- Lab 1 – Exploiting Linux
- Lab 2 – Exploiting Windows
- Lab 3 – Reverse Shell
- Lab 4 – Exploit Mitigations and Reversing with IDA