# Reverse Engineering & Threat Analysis

Duration: 40hrs (5 days)

**Module 1: Introduction to Reverse Engineering**

1.1 Overview of reverse engineering

1.2 Applications in cybersecurity, software development, and intellectual property protection

1.3 Legal and ethical aspects of reverse engineering

**Module 2: Introduction to Assembly Language (Windows & Linux)**

2.1 Basics of Assembly language

2.2 Compilers and their role in binary generation

2.3 Registers, stack, and data structures in low-level languages

2.4 Introduction to IA-32 and x64 processor architectures

2.5 Overview of Windows and Linux architecture

2.6 Understanding binary executables: ELF (Linux) and PE (Windows) formats

**Module 3: Reversing Tools**

3.1 Introduction to live and static code analysis

3.2 Overview of disassemblers, decompilers, and debuggers

3.3 Open-source decompilers: **Ghidra**, **Radare2**, **Binary Ninja (Community Edition)**

3.4 Debugging tools: **GDB** for Linux, **x64dbg** for Windows

3.5 Monitoring tools: **Process Monitor**, **Sysinternals**, **Strace** on Linux

**Module 4: Starting with IDA Pro (alongside open-source alternatives)**

4.1 Introduction to **IDA Pro**, **Ghidra**, and **Radare2**

4.2 Navigating the **IDA Pro** and **Ghidra** interfaces

4.3 Loading binaries into IDA Pro and **Ghidra** for analysis

4.4 Essential tips and tricks for **IDA Pro** and **Radare2** users

**Module 5: IDA Implementation in Reverse Engineering (with Open-Source Alternatives)**

5.1 Analyzing vulnerabilities and patching binaries

5.2 Password recovery and bypassing simple protection mechanisms

5.3 Scripting in **IDA Pro** and **Ghidra**

**Module 6: Malware Analysis (Part 1)**

6.1 Introduction to malware analysis techniques

6.2 Overview of ransomware: **WannaCry** case study

6.3 **Static analysis**: Unpacking and analyzing ransomware binaries

6.4 Using **ANY.RUN** for behavioral analysis

6.5 Overview of **Cuckoo Sandbox** as an open-source alternative to ANY.RUN

**Module 7: Reverse Engineering in Action**

7.1 Hands-on debugging techniques

7.2 Introduction to **anti-debugging** and **anti-reversing** techniques used by malware

7.3 Bypassing obfuscation and anti-reversing techniques using **GDB**, **x64dbg**, and **Radare2**

7.4 Advanced techniques: Code virtualization and packing

**Module 8: Firmware Reverse Engineering**

8.1 Introduction to firmware (BIOS, UEFI, and embedded firmware)

8.2 Tools and techniques for firmware reverse engineering using **Binwalk**, **U-Boot Tools**, and **Firmware Mod Kit**

8.3 Extracting and analyzing firmware images

8.4 Security and vulnerabilities in embedded systems

## Module 9: Advanced Malware Analysis (Part 2)

9.1 In-depth analysis of **other ransomware variants** (e.g., **Petya**, **Locky**)

9.2 Analyzing **network behaviors** of malware using **Wireshark**

9.3 Detecting **persistence mechanisms** and **payload delivery methods**

9.4 Mitigation techniques and defensive measures for ransomware

## Module 10: Case Studies & Real-World Reverse Engineering Projects

10.1 Case study: Analysis of a custom-made application and real-world scenarios

10.2 Final project: Reverse engineering and patching an obfuscated executable

10.3 Course wrap-up: Best practices and resources for continued learning