Document Generated: 04/03/2026

Learning Style: Virtual Classroom

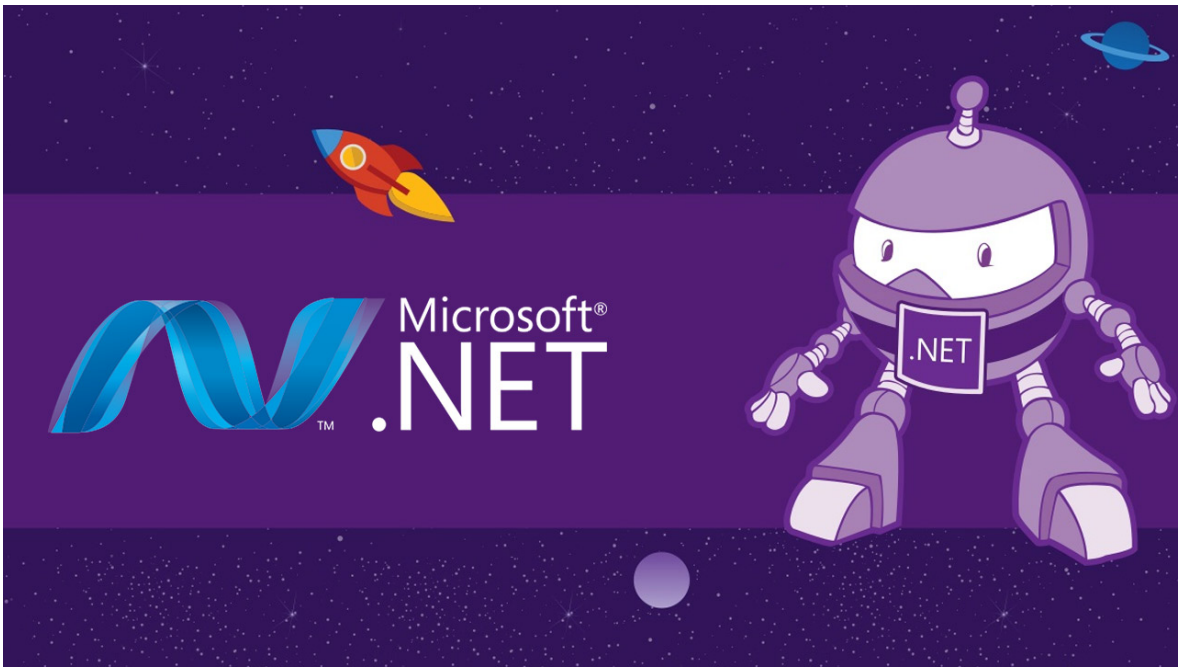
Technology:

Difficulty: Intermediate

Course Duration: 4 Days

Next Course Date: August 3, 2026

.Net Secure Coding Camp | Attacking and Securing C# / ASP.Net Core Web Applications (TT8320-N)



About this course:

Discover the cutting-edge of cybersecurity and elevate your skills as a .NET developer with our comprehensive Bug Hunting and Application Security course. Designed specifically for experienced .NET developers, our .Net Secure Coding Camp | Attacking and Securing C# / ASP .Net Web (Core) Applications is an immersive, hands-on training program that delves deep into the world of bug hunting, ethical hacking, and web application security. Through real-world case studies, engaging labs, and expert instruction, you'll gain the knowledge and skills

needed to fortify your applications, stay ahead of emerging threats, and protect your organization from costly security breaches.

Upon completing this course, you will not only acquire a profound understanding of application security concepts and best practices but also enhance your problem-solving, debugging, and overall software development prowess. Empowered with these new skills, you'll be well-prepared to identify, address, and prevent security threats in your .NET applications, ensuring a robust and secure digital environment for your organization.

Course Objective:

- Understanding Cybersecurity Concepts: Gain a solid foundation in cybersecurity principles, the evolving threat landscape, and the language of the industry to better identify and address security issues in .NET applications.

- Ethical Bug Hunting Techniques: Learn safe and appropriate methods for hunting bugs, ensuring responsible and ethical practices while working to uncover and address vulnerabilities in your applications.

- Web Application Security: Master the skills required to analyze, identify, and mitigate vulnerabilities in web applications, following best practices and guidelines from organizations such as OWASP, WASC, CWE, and CERT Secure Coding standard.

- Utilizing Industry-Standard Tools and Frameworks: Acquire hands-on experience with widely used tools and frameworks, such as Visual Studio and .NET

- Cryptography, to effectively and efficiently secure your applications.

- Improved Problem Solving and Debugging: Enhance your ability to identify, analyze, and resolve security issues in your applications through real-world case studies, labs, and expert instruction.

- Defensive Programming Techniques: Learn and apply defensive programming techniques like securing trust boundaries, input validation, and proper exception handling to create more robust and secure .NET applications.

- Cryptography in .NET: Develop a deep understanding of .NET cryptographic services, hash algorithms, symmetric and asymmetric encryption, and gain hands-on experience with a cryptography wrapper for .NET.

- Secure Software Development Processes: Gain insight into secure software development processes, including the concept of "shifting left" and the implementation of secure design principles, enabling you to create safer and more reliable .NET applications.

Audience:

- This is an intermediate level .Net programming course, designed for experienced .NET developers, software engineers, and architects who are seeking to enhance their knowledge and skills in application security, bug hunting, and secure software development. The course would also be well-suited for IT professionals, such as security analysts, security engineers, and DevOps team members, who are responsible for ensuring the security and integrity of web applications in their organizations.

Prerequisite:

- Take Before: Incoming students should have skills equivalent to the topics in, or should have recently attended, this course as a pre-reqisuite:

  TTCN20483 Introduction to Programming in C# | Creating Apps in C# and .Net Core (20483)

Course Outline:

Session: Bug Hunting Foundation

Why Hunt Bugs?

- The Language of Cybersecurity
- The Changing Cybersecurity Landscape
- AppSec Dissection of SolarWinds
- The Human Perimeter
- First Axiom in Web Application Security Analysis
- First Axiom in Addressing ALL Security Concerns

Safe and Appropriate Bug Hunting/Hacking

- Warning to All Bug Hunters
- Working Ethically
- Respecting Privacy
- Bug/Defect Notification
- Bug Hunting Pitfalls

Session: Scanning Web Applications

Scanning Applications Overview

- Scanning Beyond the Applications
- Fingerprinting
- Vulnerability Scanning: Hunting for Bugs
- Reconnaissance Goals
- Data Collection Techniques
- Fingerprinting the Environment
- Enumerating the Web Application

Session: Moving Forward from Hunting Bugs

Removing Bugs

- Open Web Application Security Project (OWASP)
- OWASP Top Ten Overview
- Web Application Security Consortium (WASC)
- Common Weaknesses Enumeration (CWE)
- CERT Secure Coding Standard
- Microsoft Security Response Center
- Software-Specific Threat Intelligence

Session: Bug Stomping 101

Unvalidated Data

- CWE-787, 125, 20, 416, 434, 190, 476 and 119
- Potential Consequences
- Defining and Defending Trust Boundaries
- Rigorous, Positive Specifications
- Allow Listing vs Deny Listing

- Challenges: Free-Form Text, Email Addresses, and Uploaded Files

A01: Broken Access Control

- CWE-22, 352, 862, 276, and 732
- Elevation of Privileges
- Insufficient Flow Control
- Unprotected URL/Resource Access/Forceful Browsing
- Metadata Manipulation (Session Cookies and JWTs)
- Understanding and Defending Against CSRF
- CORS Misconfiguration Issues

A02: Cryptographic Failures

- CWE-200
- Identifying Protection Needs
- Evolving Privacy Considerations
- Options for Protecting Data
- Transport/Message Level Security
- Weak Cryptographic Processing
- Keys and Key Management
- NIST Recommendations

A03: Injection

- CWE-79, 78, 89, and 77
- Pattern for All Injection Flaws
- Misconceptions With SQL Injection Defenses
- Drill Down on Stored Procedures
- Other Forms of Server-Side Injection

- Minimizing Server-Side Injection Flaws
- Client-side Injection: XSS
- Persistent, Reflective, and DOM-Based XSS
- Best Practices for Untrusted Data

A04: Insecure Design

- Secure Software Development Processes
- Shifting Left
- Principles for Securing All Designs
- Leveraging Common AppSec Practices and Control
- Paralysis by Analysis
- Actionable Application Security
- Additional Tools for the Toolbox

A05: Security Misconfiguration

- System Hardening: IA Mitigation
- Risks with Internet-Connected Resources
- Minimalist Configurations
- Application Allow Listing
- Secure Baseline
- Segmentation with Containers and Cloud
- CWE-611
- Safe XML Processing

Session: Bug Stomping 102

A06: Vulnerable and Outdated Components

- Problems with Vulnerable Components

- Software Inventory
- Managing Updates: Balancing Risk and Timeliness
- Virtual Patching
- Dissection of Ongoing Exploits

A07: Identification and Authentication Failures

- CWE-306, 287, 798 and 522
- Quality and Protection of Authentication Data
- Anti-Automation Defenses
- Multifactor Authentication
- Proper Hashing of Passwords
- Handling Passwords on Server Side

A08: Software and Data Integrity Failures

- CWE-502
- Software Integrity Issues and Defenses
- Using Trusted Repositories
- CI/CD Pipeline Issues
- Protecting Software Development Resources
- Serialization/Deserialization

A09: Security Logging and Monitoring Failures

- Detecting Threats and Active Attacks
- Best Practices for Logging and Logs
- Safe Logging in Support of Forensics

A10: Server Side Request Forgeries (SSRF)

- CWE-918
- Understanding SSRF
- Remote Resource Access Scenarios
- Complexity of Cloud Services
- SSRF Defense in Depth
- Positive Allow Lists

Session: Moving Forward with Application Security

Applications: What Next?

- Common Vulnerabilities and Exposures
- CWE Top 25 Most Dangerous SW Errors
- Strength Training: Project Teams/Developers
- Strength Training: IT Organizations

.NET Issues and Best Practices

- Managed Code and Buffer Overflows
- .Net Permissions
- ActiveX Controls
- Proper Exception Handling

Session: Exploring .Net Cryptography

.Net Cryptographic Services

- The role of cryptographic services
- Hash algorithms and hash codes
- Encrypting data symmetrically
- Encrypting data asymmetrically

Credly Badge:

Display your Completion Badge And Get The Recognition You Deserve.

Add a completion and readiness badge to your Linkedin profile, Facebook page, or Twitter account to validate your professional and technical expertise. With badges issued and validated by Credly, you can:

- Let anyone verify your completion and achievement by clicking on the badge
- Display your hard work and validate your expertise
- Display each badge's details about specific skills you developed.

Badges are issued by QuickStart and verified through Credly.

Find Out More or See List Of Badges