.NET, C# and ASP.NET security development

CL-NSC	.NET, C# and ASP.NET security development	Classroom 2 days

A number of programming languages are available today to compile code to .NET and ASP.NET frameworks. The environment provides powerful means for security development, but developers should know how to apply the architecture- and coding-level programming techniques in order to implement the desired security functionality and avoid vulnerabilities or limit their exploitation.

The aim of this course is to teach developers through numerous hands-on exercises how to prevent untrusted code from performing privileged actions, protect resources through strong authentication and authorization, provide remote procedure calls, handle sessions, introduce different implementations for certain functionality, and many more.

Introduction of different vulnerabilities starts with presenting some typical programming problems committed when using .NET, while the discussion of vulnerabilities of the ASP.NET also deals with various environment settings and their effects. Finally, the topic of ASP.NET-specific vulnerabilities not only deals with some general Web application security challenges, but also with special issues and attack methods like attacking the PostBack or the ViewState, or the string termination attacks.

Audience: .NET, C# and ASP.NET developers, software architects and testers, max 15 participants

SDL level: 200	Preparedness: Basic .NET, C# and ASP.NET	Exercises: Hands-on

Content

Security technologies and services: Code Access Security, Role Access Security, Remoting Architecture; ASP.NET trust levels; form authentication; session handling; provider model; membership, role management and the Microsoft Passport Network.

Vulnerabilities, attacks and mitigations: integer overflows in .NET; injection flaws in ASP.NET: SQL Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), log forging; unsafe native calls; Equals() and toString() problems; attacking PostBack and ViewState; string termination attacks; direct call to GC.Collect(); implementation of ICloneable; class comparison methods; using the [Serializable] attribute; unsafe reflection; attacking PostBack and ViewState; string termination attacks; and many more…

Other courses that relate to the topic of this course:

CL-NSC - .NET, C# and ASP.NET security development (Classroom, 2 days)

CL-WSC - Web application security (Classroom, 2 days)

CL-CNA - Combined C/C++/C#, ASP.NET and Web application security (Classroom, 4 days)

CL-WTS - Web application testing (Classroom, 2 days)

RT-NST - .NET and ASP.NET security technologies (Remote, 2x1.5h)

RT-NVL - .NET specific vulnerabilities (Remote, 2x1.5h)

RT-AVL - ASP.NET specific vulnerabilities (Remote, 2x1.5h)

RT-WVL - Web application vulnerabilities (Remote, 2x1.5h)

Detailed table of contents

Day 1

Basic security concepts

- Notion of security
- Security related terms
- Definition of risk
- Temporality of security
- Specialty of information technology security
- Main IT security terms
- Different aspects of security
- Building a secure system
- Fighting security flaws and vulnerabilities
  - Nature of security flaws
  - Technical reasons
  - Hackers vs. developers
  - Weak market driving forces
  - Lack of business motivation
  - Threats are global
  - Organized network of criminals
- Classification of security flaws
  - Landwehr's taxonomy
  - Trojan horses
  - Backdoors
  - Covert channels
  - The Fortify taxonomy
  - Fortify vulnerability categories
  - OWASP Top Ten (2010)

.NET security architecture

- Code Access Security
- Role Access Security
- Remoting and security

ASP.NET security architecture

- ASP.NET trust levels
- Form authentication
- Session state
- Provider model
- Membership
- Role manager

Passport

Protecting .NET code: reverse engineering, obfuscation

Day 2

Vulnerabilities of .NET

- Direct call to GC.Collect()
- Implementation issues of Equals()
- toString() problems in case of arrays
- Null checking
- Equals() / GetHashCode() definition
- Return value checking
- Implementation of ICloneable
- Class comparison
- Using the [Serializable] attribute
- Unreleased resources
- Empty catch block / overly broad catch
- Log forging
- Unsafe reflection
- Integer overflow
- Unsafe native calls

Vulnerabilities of the ASP.NET runtime environment

- Custom errors disabled
- Leaving tracing enabled
- Leaving debugging enabled
- Cookies accessible through client-side script
- Cookieless session state enabled
- Cookieless authentication enabled
- Hardcoded credentials used

ASP.NET specific vulnerabilities

- Attacking the PostBack
- Control accessibility vulnerabilities
- Control sequence attacks
- Attacking the ViewState
- String termination attacks

Note: Our classroom trainings come with a number of easy-to-understand exercises providing live hacking fun. By accomplishing these exercises with the lead of the trainer, participants can analyze vulnerable code snippets and commit attacks against them in order to fully understand the root causes of certain security problems. All exercises are prepared in a plug-and-play manner by using a pre-set desktop virtual machine, which provides a uniform development environment.

Combined C/C++/C#, ASP.NET and Web application security

CL-CNA	Combined C/C++/C#, ASP.NET and Web application security	Classroom 4 days

Serving teams that use managed code (.NET and ASP.NET typically written in C#) together with native code development (typically C/C++), this training gives a comprehensive overview of the security issues in both environments.

Concerning C/C++, common security vulnerabilities are discussed, backed by practical exercises about the attacking methods that exploit these vulnerabilities, with the focus on the mitigation techniques that can be applied to prevent the occurrences of these dangerous bugs, detect them before market launch or prevent their exploitation.

The course also covers both the various general (like Web services) and specific security solutions and tools, and the most frequent and severe security flaws of managed code, dealing with both language-specific issues and the problems stemming from the runtime environment. The vulnerabilities relevant to the ASP.NET platform are detailed along with the general web-related vulnerabilities following the OWASP Top Ten list. The course consists of a number of exercises through which attendees can easily understand and execute attacks and protection methods.

Audience: C/C++/C#, .NET and ASP.NET developers, architects and testers, max 15 participants

SDL level: 300	Preparedness: Advanced C/C++/C#, ASP.NET and Web	Exercises: Hands-on

Content

Security technologies and services: Code Access Security, Role Access Security, Remoting Architecture; ASP.NET trust levels; form authentication; session handling; provider model; membership, role management and the Microsoft Passport Network; SOAP and REST; secure communication and transport-layer security (TLS/SSL and IPSEC), application- and container-managed authentication, authorization; End-to-end security; Web Services Security (WSS), signing (XML Sig) and encryption (XML Enc)

Common security vulnerabilities and mitigation techniques: Buffer Overflow (BOF), heap overflow; integer problems: widthness bug, signedness bug, arithmetic overflow; Printf Format String bug (PFS); array indexing problems, unicode bug, side channels: the RSA timing attack, Time-of-Checking-to-Time-of-Usage (TOCTTOU) race conditions, Directory Traversal Vulnerability (DTV); No eXecute (NX bit) access mode of Virtual Memory Management (VMM); Data Execution Prevention (DEP); Address Space Layout Randomization (ASLR) – PaX, ExecShield; Stack Smashing Protection (SSP) – /GS, StackGuard, ProPolice; Source Code Analyzers (SCA tools).

.NET and ASP.NET vulnerabilities, attacks and mitigations: integer overflows in .NET; injection flaws in ASP.NET: SQL Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), log forging; unsafe native calls; Equals() and toString() problems; attacking PostBack and ViewState; string termination attacks; direct call to GC.Collect(); implementation of ICloneable; class comparison methods; using the [Serializable] attribute; unsafe reflection; attacking PostBack and ViewState; string termination attacks; and many more…

Exercises: exploiting stack overflow – executing shell codes; crafting a printf format attack string – write-what-where (WWW) possibilities; understanding integer problems; applying mitigation techniques; circumventing them by return-to-libc attack or argument overwriting; and many spot- and-correct-the-bug exercises. WS Security with username and password; XMLS Signature; XML Encryption; exploiting SQL injection step-by-step; crafting Cross-Site Scripting attacks; uploading and running executable code

Other courses that relate to the topic of this course:

CL-CSC - C/C++ secure coding (Classroom, 2 days)
CL-NSC - .NET, C# and ASP.NET security development (Classroom, 2 days)
CL-WSC - Web application security (Classroom, 2 days)
RT-CVL - Common C/C++ security vulnerabilities (Remote, 2x1.5h)
RT-CPR - Protection against common security vulnerabilities (Remote, 2x1.5h)
RT-CEX - C/C++ secure coding exercises (Remote, 2x1.5h)
RT-NST - .NET and ASP.NET security technologies (Remote, 2x1.5h)
RT-NVL - .NET specific vulnerabilities (Remote, 2x1.5h)
RT-AVL - ASP.NET specific vulnerabilities (Remote, 2x1.5h)
RT-WVL - Web application vulnerabilities (Remote, 2x1.5h)

Detailed table of contents

Day 1

Basic security concepts

- Notion of security
- Security related terms
- Definition of risk
- Temporality of security
- Specialty of information technology security
- Main IT security terms
- Different aspects of security
- Building a secure system
- Fighting security flaws and vulnerabilities
  - Nature of security flaws
  - Technical reasons
  - Hackers vs. developers
  - Weak market driving forces
  - Lack of business motivation
  - Threats are global
  - Organized network of criminals
- Classification of security flaws
  - Landwehr's taxonomy
  - Trojan horses
  - Backdoors
  - Covert channels
  - The Fortify taxonomy
  - Fortify vulnerability categories
  - OWASP Top Ten (2010)

Security relevant C/C++ programming bugs and flaws

Common security vulnerabilities

- Programming bugs
- Exploitable security flaws
- Protection principles
  - Protection methods
  - Specific protection methods
  - Protection methods at different layers
  - The PreDeCo matrix
- x86 machine code, memory layout, stack operations
  - Intel 80x86 Processors – main registers
  - Intel 80x86 Processors – most important instructions
  - Intel 80x86 Processors – control instructions
  - Intel 80x86 Processors – stack handling instructions
  - The memory address layout
  - The stack
  - The function calling mechanism
  - The stack frame
  - The stack frame during a function call
  - Stack frame of nested calls
  - Function calls – prologue and epilogue of a function

Buffer overflow

- Stack overflow
  - Buffer overflow on the stack
  - Overwriting the return address
  - Localizing the position of the return address
  - Exercise BOFIntro
  - Exercise BOFAccess
  - Exercise BOFJump
  - Exercise BOFShellcode
- Protection against stack overflow
  - Stack Overflow – Prevention
  - Stack Overflow – Detection
  - Buffer security check / Stack smashing protection (/GS)
  - Exercise Buffer Security Check
  - Using /GS (Buffer Security Check)
  - Effects of stack smashing protection
  - The security_check_cookie() function
  - Stack smashing protection costs and variants
  - Bypassing stack smashing protection – Overwriting arguments
  - Overwriting arguments – Mitigation
  - Stack overflow – Anti-exploit techniques
  - Address Space Layout Randomization (ASLR)
  - Stack randomization with ASLR
  - Virtual Memory Management related protection

  - Data Execution Prevention (DEP)
  - Exercise DEP
- Return-to-libc attack – circumventing DEP
  - Exploitation without running code on stack
  - Arc injection / Return-to-libc attack
  - Exercise Return-to-libc
  - Multiple function calls with return-to-libc

Day 2

- Heap overflow
  - Memory allocation managed by a doubly-linked list
  - Buffer overflow on the heap
  - Steps of freeing and joining memory blocks
  - Freeing allocated memory blocks
- Protection against heap overflow
  - Heap overflow – Prevention
  - Heap overflow – Detection
  - Heap overflow – Anti-exploit techniques
  - Mixing delete and delete[]

Integer problems in C/C++

- Types of integer problems
- Representation of negative integers
- Integer representation by using the two's complement
- Integer ranges
- Widthness integer overflow – spot the bug!
- Signedness bug – spot the bug!
- Arithmetical overflow – spot the bug!
- The integer promotion rule in C/C++
- Integer promotion of signed and unsigned integers
- Exercise IntOverflow
- Why ABS(INT_MIN)==INT_MIN?
- Exercise IntSignedness
- Exercise IntArithmetics
- Exercise GDI
- Exercise Board
- Integer problem mitigation
  - Avoiding arithmetic overflow – addition
  - Avoiding arithmetic overflow – multiplication
  - The SafeInt class
  - Other C compatible libraries

Printf format string bug

- Exercise Printf – the print format string bug
- Printf format string exploit – overwriting the return address
- Exercise PrintfExploit – exploiting the print format string bug
- Mitigation of printf format string problem
  - Printf format string bug – Prevention
  - Printf format string bug – Detection
  - Printf format string bug – Anti-exploit techniques

Other common security vulnerabilities

- Array indexing – spot the bug!
- Unicode bug

Other security flaws

- Miscellaneous flaws
  - An example information leakage
  - Serialization errors (TOCTTOU)
  - Temporary files / a C++ TOCTOU vulnerability
  - Risks using signaling mechanisms
- File I/O risks
  - Directory Traversal Vulnerability
  - Symbolic Link Vulnerability
- RSA timing attack
  - Introduction to RSA algorithm
  - Implementation of encoding/decoding in RSA
  - Fast exponentiation
  - Differences in execution times
  - RSA timing attack
  - Measurements
  - RSA timing attack – principles
  - Correlation of total and partial execution times
  - RSA timing attack – in practice
  - The RSA timing attack algorithm
  - Practical exploitation using the RSA timing attack
  - Attacking SSL servers
- Mitigation of side channel attacks
  - Blind signature

Advices and principles

- What can you do as a developer
- What can you do as an operator
- Saltzer’s 8 principles
- Matt Bishop’s principles of robust programming
- References
- Other sources

- Vulnerability databases
- Summary and takeaways

Day 3

.NET security architecture

- Code Access Security
- Role Access Security
- Remoting and security

ASP.NET security architecture

- ASP.NET trust levels
- Form authentication
- Session state
- Provider model
- Membership
- Role manager

Passport

Protecting .NET code: reverse engineering, obfuscation

Security of Web services

- Web services overview
  - Web services – Introduction
  - Web services – Architecture
  - Web services – Operational model
  - Basic technologies
  - SOAP - Simple Object Access Protocol
  - Structure of SOAP messages
  - Web Services Definition Language (WSDL)
  - REST - REpresentational State Transfer
- Security solutions of Web services
  - Security requirements and challenges
  - Two approaches
  - Standards
- Transport layer security
  - Exercises – Transport layer security
  - Exercise WS – Plain communication with Tomcat (tc)
  - Exercise WS – SSL communication with Tomcat (tc)
  - Exercise WS – Application managed authentication (tcauth)
  - Exercise WS – Container managed authentication (tcauthc)
  - Exercise WS – Container authentication (tcauthc) by digest

- End-to-end security
  - WS-Security (Web Services Security / WSS)
  - Security elements of SOAP messages
  - Exercise – WS-Security with Username/Password
  - Exercise – WS-Security with digital signature (XML Signature)
  - Components of XML Signature
  - Exercise – WS-Security with encryption (XML Encryption)
  - Components of XML Encryption

Day 4

Vulnerabilities of .NET

- Direct call to GC.Collect()
- Implementation issues of Equals()
- toString() problems in case of arrays
- Null checking
- Equals() / GetHashCode() definition
- Return value checking
- Implementation of ICloneable
- Class comparison
- Using the [Serializable] attribute
- Unreleased resources
- Empty catch block / overly broad catch
- Log forging
- Unsafe reflection
- Integer overflow
- Unsafe native calls

Vulnerabilities of the ASP.NET runtime environment

- Custom errors disabled
- Leaving tracing enabled
- Leaving debugging enabled
- Cookies accessible through client-side script
- Cookieless session state enabled
- Cookieless authentication enabled
- Hardcoded credentials used

ASP.NET specific vulnerabilities

- Attacking the PostBack
- Control accessibility vulnerabilities
- Control sequence attacks
- Attacking the ViewState
- String termination attacks

Note: Our classroom trainings come with a number of easy-to-understand exercises providing live hacking fun. By accomplishing these exercises with the lead of the trainer, participants can analyze vulnerable code snippets and commit attacks against them in order to fully understand the root causes of certain security problems. All exercises are prepared in a plug-and-play manner by using a pre-set desktop virtual machine, which provides a uniform development environment.

.NET and ASP.NET security technologies

RT-NST	.NET and ASP.NET security technologies	Remote 2x1.5h
The course gives an insight into the various security solutions provided by the .NET platform, the .NET Security Model, Code and Role Access Security, and the means to protect your .NET code. The ASP.NET security architecture is also presented tackling trust level, authentication and authorization issues, impersonation, session state, provider model, membership and role manager.		
Audience: Software developers and architects for .NET and ASP.NET frameworks, max 8 participants		
SDL level: 200	Preparedness: Basic .NET and ASP.NET	Exercises: No

Other courses that relate to the topic of this course:

CL-NSC - .NET, C# and ASP.NET security development (Classroom, 2 days)

CL-WSC - Web application security (Classroom, 2 days)

CL-CNA - Combined C/C++/C#, ASP.NET and Web application security (Classroom, 4 days)

RT-NVL - .NET specific vulnerabilities (Remote, 2x1.5h)

RT-AVL - ASP.NET specific vulnerabilities (Remote, 2x1.5h)

Detailed table of contents

Remote training introduction

- Aim
- Agenda
- Introduction

.NET security architecture

- Overview of .NET security features
  - .NET Security Model
  - Code Access Security
    - Evidence and Code Identity
    - Permissions
    - Membership Conditions
    - Code Groups
    - Security Policy
    - Administering Code-Access Security
  - Role Access Security
    - Imperative User-Base Security
    - Declarative User-Based Security
    - Credentials
- Protecting .NET code
  - Verification and Validation
  - Reverse engineering
  - Code obfuscation

ASP.NET security

- ASP.NET trust level
- Authentication
  - Form Authentication
  - Passport Authentication
  - Windows Authentication
- Authorization
- Impersonation
- Session state
- Provider model
- Membership
- Role manager

Note: The remote courses are live, instructor-led and delivered in an interactive manner. To uphold the continuous attention of the trainees and to serve better understanding we apply special interactive techniques – like prompting for answers, asking to pinpoint problems in source codes or soliciting corrections to code snippets –, which make our courses unique.

.NET specific vulnerabilities

RT-NVL	.NET specific vulnerabilities	Remote 2x1.5h
The course introduces the .NET specific vulnerabilities, classified based on their nature, and forming categories like API abuse, problems stemming from code quality, password management issues, improper error handling, bugs related to missing or inappropriate input validation and faulty data representation, and finally improper use of security features. The course deals with both language-specific issues and the problems stemming from the runtime environment.		
Audience: Software developers using .NET languages, max 8 participants		
SDL level: 200	Preparedness: Basic .NET	Exercises: Demo

Other courses that relate to the topic of this course:

CL-NSC - .NET, C# and ASP.NET security development (Classroom, 2 days)

CL-WSC - Web application security (Classroom, 2 days)

CL-CNA - Combined C/C++/C#, ASP.NET and Web application security (Classroom, 4 days)

RT-AVL - ASP.NET specific vulnerabilities (Remote, 2x1.5h)

To follow this remote course we suggest first attending the following courses:

RT-NST - .NET and ASP.NET security technologies (Remote, 2x1.5h)

Detailed table of contents

Remote training introduction

- Aim
- Agenda
- Introduction

API Abuse

- Direct call to GC.Collect()
- Equals Implementation problems
- toString problems in case of arrays
- Equals() / GetHashCode() definition
- Return value checking

Code Quality

- ICloneable implementation
- Class comparison
- Using the [Serializable] attribute
- Null argument passed to equals()
- Unreleased resources

.NET specific vulnerabilities

Password management

- Empty/hardcoded/null password
- Passwords stored in configuration file
- Vulnerabilities of hashed passwords

Improper error handling

- Empty catch block
- Overly broad catch
- Using NullReferenceException

Input validation and representation

- Log forging
- Unsafe reflection
- Integer overflow
- Unsafe native calls

Improper use of security features

- Insecure randomness
- Privacy violation
- Weak cryptographic hash
- Weak encryption

Note: The remote courses are live, instructor-led and delivered in an interactive manner. To uphold the continuous attention of the trainees and to serve better understanding we apply special interactive techniques – like prompting for answers, asking to pinpoint problems in source codes or soliciting corrections to code snippets –, which make our courses unique. The various vulnerabilities, attack methods and mitigation techniques are demonstrated through exercises, which participants can follow on slides by the lead of the trainer.

ASP.NET specific vulnerabilities

RT-AVL	ASP.NET specific vulnerabilities	Remote 2x1.5h
The general Web-related vulnerabilities are presented by following the latest OWASP Top Ten list, along with their relevance when using ASP.NET. Besides discussing most relevant vulnerabilities like SQL Injection or Cross-Site Scripting, the ASP.NET specific topics include attacking the PostBack, control accessibility vulnerabilities, control sequence attacks, attacking the Viewstate or string termination attacks. Both general and platform specific mitigation techniques are introduced.		
Audience: Software developers using the ASP.NET framework, max 8 participants		
SDL level: 200	Preparedness: Basic ASP.NET	Exercises: Demo

Other courses that relate to the topic of this course:

CL-NSC - .NET, C# and ASP.NET security development (Classroom, 2 days)

CL-WSC - Web application security (Classroom, 2 days)

CL-CNA - Combined C/C++/C#, ASP.NET and Web application security (Classroom, 4 days)

To follow this remote course we suggest first attending the following courses:

RT-NST - .NET and ASP.NET security technologies (Remote, 2x1.5h)

RT-NVL - .NET specific vulnerabilities (Remote, 2x1.5h)

Detailed table of contents

Remote training introduction

- Aim
- Agenda
- Introduction

General web-based vulnerabilities – OWASP Top 10

- SQL injection (presented by using MSSQL)
- Command injection
- Cross-Site Scripting (XSS)
- Cross Site Request Forgery (CSRF)
- Malicious file execution
- Insecure direct object reference
- Uploading an executable file
- Privacy Violation – insecure storage of sensitive personal data
- Information leakage through error reporting

ASP.NET specific vulnerabilities

Vulnerabilities of the ASP.NET runtime environment

- Custom Errors Disabled
- Leaving Tracing Enabled
- Leaving Debugging Enabled
- Cookies Accessible Through Client-Side Script
- Cookieless Session State Enabled
- Cookieless Authentication Enabled
- Hardcoded Credentials Used

ASP.NET specific vulnerabilities

- Attacking the PostBack
- Control accessibility vulnerabilities
- Control sequence attacks
- Attacking the Viewstate
- String termination attacks

Note: The remote courses are live, instructor-led and delivered in an interactive manner. To uphold the continuous attention of the trainees and to serve better understanding we apply special interactive techniques – like prompting for answers, asking to pinpoint problems in source codes or soliciting corrections to code snippets –, which make our courses unique. The various vulnerabilities, attack methods and mitigation techniques are demonstrated through exercises, which participants can follow on slides by the lead of the trainer.