Cyberyami CSCE Certification Exam Objectives

CYBERYAMI

About the Certification

The Cyberyami Certified Secure Coding Expert (CCSCE) certification is a globally recognized, vendor-neutral credential specifically designed to validate and enhance the skills of professionals in secure coding practices. This certification ensures that individuals possess the necessary knowledge and expertise to implement security measures effectively within the software development process.

Skills You Learn

Fundamental Secure Coding Principles: Grasping the core concepts and importance of writing secure code.

Integrating Security in SDLC: Understanding how to embed security practices into every phase of software development.

Practical Security Coding Practices: Applying security measures in real-world coding scenarios to mitigate risks.

Advanced Secure Coding Techniques: Mastering sophisticated techniques to enhance the security of code against evolving threats.

Security Management Strategies: Learning to manage and maintain security standards within coding projects.

Pre-Requisites:

Basic Knowledge of Programming: Proficiency in at least one programming language (such as Java, C++, Python, etc.) is crucial as secure coding principles apply directly to how code is written.

Understanding of Software Development Processes: Familiarity with the software development lifecycle (SDLC), including stages like planning, development, testing, deployment, and maintenance.

CYBERYAMI

Fundamentals of Cybersecurity: A basic understanding of cybersecurity concepts, including common threats, vulnerabilities, and the importance of secure coding in protecting against attacks.

Experience in Software Development: Practical experience in software development, coding, or related fields to ensure a practical understanding of how coding integrates within larger projects.

Basic Knowledge of Operating Systems and Networks: Understanding the basics of operating systems, networking, and how software interacts with both, as this can impact how secure coding practices are applied.

Target Audience:

The CCSCE is ideal for software developers, coders, software engineers, and IT professionals who are involved in the development process and are seeking to enhance their secure coding skills.

Target Job Roles:

Software Developer: Design and build software applications, ensuring functionality aligns with user needs and business objectives.

Application Developer: Specialize in creating, testing, and programming apps for computers, mobile phones, and tablets.

Software Engineer: Apply engineering principles to the design, development, maintenance, testing, and evaluation of software and systems.

Security Analyst: Protect computer networks and systems by identifying vulnerabilities and implementing strategies to mitigate cyber threats.

IT Security Consultant: Advise organizations on the best practices and solutions for securing their IT infrastructure and data.

Systems Developer: Develop and implement complex software systems, focusing on meeting specific business and technical requirements.

Code Auditor: Examine and analyze code for compliance, quality, and security standards, ensuring best practices are followed.

Exam Details:

EXAM CODE	PT101
Launch Date	15 October 2023
Number of Questions	100
Length of Test	140 Mintues
Types of Questions	Multiple choice & Case based questions
Passing Score	72%
Testing Provider	CyberYami Levelup Online
Language	English

Exam Domain:

TOPICS	WEIGHTAGE (IN PERCENTAGE)
Foundations of Secure Coding	20%
Secure Software Development Lifecycle (SDLC)	15%
Security Practices in Code Development	30%
Secure Coding Techniques	20%
Security Management and Best Practices	15%

Detailed Exam Objectives

1.0 Foundations of Secure Coding

1.1 Principles of Secure Coding

- Secure Coding Practices
- Vulnerability Mitigation
- Software Security
- Security Best Practices
- Input Validation Techniques
- Output Encoding Methods
- Authentication Mechanisms
- Authorization Practices
- Error Handling StrategiesSecure
- Secure Communication Protocols
- Configuration Management
- Least Privilege Principle
- Code Vulnerability Prevention
- Secure Libraries Usage
- Data Encryption Techniques
- Security Testing Procedures
- Common Security Threats
- SQL Injection Prevention
- Cross-Site Scripting (XSS) Mitigation
- Dynamic Code Analysis Tools

1.2 Threat Modeling

- Unauthorized Access
- Malicious Activities
- Preventive Measures
- Proactive Security
- SQL Injection (SQLi)
- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery (CSRF)
- Command Injection
- Server-Side Request Forgery (SSRF)
- Insecure Direct Object References (IDOR)
- Security Misconfigurations
- Insecure Deserialization
- Buffer Overflows
- Memory Vulnerabilities
- Broken Authentication
- Session Management
- Clickjacking
- Insecure File Uploads
- XML External Entity (XXE) Injection
- Cross-Origin Resource Sharing (CORS)
- Secure Coding Practices
- Security Measures
- Best Practices
- Prevention Guidelines
- Parameterized Queries
- Output Encoding

2.0 Secure Software Development Lifecycle

2.1 Security Requirements Analysis

- Development Phases
- Requirements Gathering
- Project Planning
- Software Design
- Development Coding
- Testing Phase
- Software Deployment
- Maintenance Phase
- SDLC Models
- Waterfall Model
- Agile Methodology
- Iterative Model
- V-Model
- Incremental Model
- Big Bang Model
- Flexibility in SDLC
- Risk Mitigation
- Customer Collaboration
- Continuous Feedback
- User Involvement
- Iteration Management
- Spiral Model
- Scope Creep
- Project Timelines

3.0 Security Practices in Code Development

3.1 Input Validation and Data Sanitization

- Data Validation
- PHP Functions
- Input Sanitization
- Input Integrity
- Security Measures in PHP
- Data Sanitization

- Filter_Input() Function
- Regular Expressions in PHP
- Client-side Validation
- Server-side Validation
- Whitelist Approach
- Parameterized Queries in PHP
- Input Length Validation
- Buffer Overflow Prevention
- Check for Missing Values
- Invalid Characters Validation
- Cross-Site Scripting (XSS) Prevention
- User Input Security
- CSRF Protection
- Token-Based Security
- Injection Attacks
- Security Best Practices
- PHP Sanitization
- Filter_var() Function
- mysqli_real_escape_string()
- PDO Class Sanitization
- V Whitelist Approach
- Blacklist Approach
- Input Validation
- Cross-Site Scripting (XSS)
- Security Vulnerabilities
- Database Abstraction Layer
- Context-Specific Escaping
- Regular Expressions (Regex)

4.0 Secure Coding Techniques

4.1 Authentication and Authorization

- Authentication
- Secure Authentication Mechanism
- Multi-Factor Authentication (MFA)
- Password Hashing
- Limiting Login Attempts
- Secure Password Reset Process
- Fine-Grained Control
- Data Breaches
- Access Management
- Access Control Models
- Least Privilege Principle
- Fine-Grained Control

- Token-Based Reset Link
- Credential Verification
- Access Decision
- Session Variables
- Use Secure Password Hash
- Salt The Password
- Use a slow hash function
- Use the secure password reset
- Implement account isolation
- Role-Based Access Control (RBAC)
- Granular Access Control
- Permission-Based Systems
- Security Model
- Access Control Lists (ACLs)
- Directory Services
- Regulatory Requirements
- Access Permissions
- Revoking Access
- Hierarchical Structures
- Least Privilege Access Control
- User Roles
- Risk of Unauthorized Access
- Compliance
- Efficient Access Control

4.3 Secure Error Handling

- Secure Logging
- Fail-Safe Responses
- Input Validation
- HTTP Status Codes
- Custom Error Messages
- Generic Error Codes
- Practical Examples
- User Trust
- Logging Libraries
- Logging Levels
- Avoiding Sensitive Data in Logs
- Timestamps and Contextual Information
- Log Sanitization
- Secure Log Storage
- Cryptographic Hashing
- Practical Logging Examples

5.0 Security Management and Best Practices

5.1 Security Training and Awareness

- Secure Coding Practices
- Security Posture
- Real-World Examples
- Secure Software Development Lifecycle (SDLC)

5.2 Incident Response Planning

- Incident Response Plan
- Security Breaches
- Cyberattacks
- Data Breaches
- Security Incidents
- Business Continuity
- Automation and Tools
- Realistic Scenarios
- Continuous Improvement
- Incident Documentation
- Threat Intelligence Integration
- Machine Learning and AI

5.2 Incident Response Planning

- Preparedness Strategies
- Minimizing Damage
- Reducing Downtime
- Legal and Regulatory Compliance
- Incident Response Steps
- Leadership Buy-In
- Cross-Functional Team
- Risk Assessment
- Communication Protocols
- Cloud-Centric Approaches
- Remote Incident Response
- Collaboration and Sharing
- User Behavior Analytics
- Integrated Incident Management
- Resilience and Recovery
- Regulatory Compliance
- Human-Centric Approach
- Next-Gen Incident Response