Date Issued: 2/1/2017
Date Modified: 5/09/2019
Version: 1.4

CertNexus Cyber Secure Coder (CSC) Exam CSC-110

Exam Information

Candidate Eligibility:

The *Cyber Secure Coder (CSC)* exam requires no application fee, supporting documentation, or other eligibility verification measures for you to be eligible to take the exam. Your exam voucher will come bundled with your training program, which can be purchased here. Once purchased, you will receive more information about how to register for and schedule your exam through Pearson Vue. You can also purchase a voucher directly through Pearson Vue. Once you have obtained your voucher information, you can register for an exam time here. By registering, you agree to our Candidate Agreement included here.

Exam Prerequisites

While there are no formal prerequisites to register for and schedule an exam, we strongly recommend that you first possess the knowledge, skills, and abilities to do the following:

- Develop applications using multiple programming languages and coding environments while following generally accepted coding best practices
- Develop applications for a variety of platforms: web, cloud, mobile, desktop
- Write and analyze use cases, technical requirements, specifications, and other application documentation
- Work with common tools, such as analysis, debugging, encryption, and penetration testing tools

You can obtain this level of skill and knowledge by taking the following courseware, which is available through training providers located around the world, or by attending an equivalent third-party training program:

- *CertNexus Cyber Secure Coder (Exam CSC-110)*

Exam Specifications

Number of Items: 80

Passing Score: 56 out of 80 (70%)

Duration: 120 minutes

Exam Options: In person at Pearson VUE test centers

Item Formats: Multiple Choice/Multiple Response/True-False

Exam Description

Target Candidate:

This certification exam is designed for software developers, testers, and architects who may develop in multiple programming languages for any type of platform who desire or are required to develop highly secure applications for business and organizational use. Candidates will also have a need to author or analyze specifications and technical requirements and develop applications that meet them.

Exam Objective Statement:

This exam will certify that the successful candidate has the knowledge, skills, and abilities to design and develop a variety of applications for various platforms, analyze security concerns outside of specific languages and platforms, use a number of testing and analysis tools, and mitigate against common threats to data and systems.

To ensure exam candidates possess the aforementioned knowledge, skills, and abilities, the *Cyber Secure Coder (CSC)* exam will test them on the following domains with the following weightings:

Domain	% of Examination
1.0 Common Secure Application Development Terminology and Concepts	15%
2.0 Job and Process Responsibilities Related to Secure Application Development	15%
3.0 Architecture and Design	18%
4.0 Risk Assessment and Management	17%
5.0 Application Implementation	35%
Total	**100%**

The information that follows is meant to help you prepare for your certification exam. This information does not represent an exhaustive list of all the concepts and skills that you may be tested on during your exam. The exam domains, identified previously and included in the objectives listing, represent the large content areas covered in the exam. The objectives within those domains represent the specific tasks associated with the job role(s) being tested. The information beyond the domains and objectives is meant to provide examples of the types of concepts, tools, skills, and abilities that relate to the corresponding domains and objectives. All of this information represents the industry-expert analysis of the job role(s) related to the certification and does not necessarily correlate one-to-one with the content covered in your training program or on your exam. We strongly recommend that you independently study to familiarize yourself with any concept identified here that was not explicitly covered in your training program or products.

Objectives:

Domain 1.0 Common Secure Application Development Terminology and Concepts

Objective 1.1 Understanding basic security principles

- Encryption
  - Public keys
  - Private keys
  - Hashes
- Division of resources/categorization of components
  - Physical storage and local resources
  - Network/data in transit
  - The application
- The CIA Triad
- AAA
  - Identity management
  - User provisioning
- Least privilege
- Defense in Depth
- Fail safe
- Weakest link
- Separation of duties
  - Coding/testing/deployment
    - Environment
    - Personnel
- Monitoring
  - Logs
  - Authentication errors
  - Security exceptions

Objective 1.2 Identify common hacking terminology and concepts

- Black hat, gray hat, white hat
- Builders and breakers
- Social engineering
  - Phishing
  - Spear phishing
  - Social media recon
- Vulnerability
  - Lack of input validation
  - Exceptions/unintentional disclosure

    - Weak encryption
    - Buffer/memory
- Exploits and attacks
    - DoS
    - SQL injection
    - XSS
    - Buffer overrun
    - Brute force

Domain 2.0 Job and process responsibilities related to secure application development

Objective 2.1 Explain the software development lifecycle (SDLC)

- SDLC phases
    - Design
    - Implementation
        - Code validation and verification
    - Testing
    - Deployment
    - Review

Objective 2.2 Understand the role of the designer/architect in creating secure applications

- Design deliverables
    - Security requirements
    - Security design and architecture
    - Vulnerability mapping
    - Asset identification
        - Data to manage
        - Data to protect
        - Data storage locations
        - Data transit
- White-boarding
- Compliance assurance and adherence to organizational requirements

Objective 2.3 Explain the role of the developer in creating secure applications

- Development deliverables
    - Source code
    - Documentation
    - Test cases
- Debugging tools
    - Unit testing
        - JUnit
        - UnitTest
    - IDE
        - Visual Studio

        - Eclipse
    - Use of standard libraries and APIs

Objective 2.4 Understand the role of the code reviewer in creating secure applications

- Review deliverables
    - Vulnerability reports
    - Quality assurance reports
- Static analysis tools
    - Coverity Code Advisor
    - Fortify Software Static Code Analyzer
    - FindBugs

Objective 2.5 Understand the role of the application tester in creating secure applications

- Testing deliverables
    - Test cases
    - Vulnerability reports
- Penetration testing
    - Fuzzing
    - Validation
    - Session management
    - Client-side testing
    - Testing tools
        - Metasploit
        - Nessus
        - Kismet
        - Zap
        - Nmap

Domain 3.0 Architecture and Design

Objective 3.1 Interpret use and abuse cases

- Design intentions
    - Actors
        - User
        - Admin
        - Financial institutions
        - Service providers
    - Actions
        - Login
        - Search
        - Payment
        - Product or service delivery
- Attacks
    - Entry points

    - Vulnerability identification
    - Exploit selection and design
    - Exploit execution

Objective 3.2 Understand architecture and design industry best practices

- Modular design
    - Maintainability of code
    - Reducing the attack surface
    - Defense in Depth
- Design methodologies
    - Microsoft STRIDE
- Software design patterns
- Security design patterns
- Requiring strong passwords
- Identity management process
- Design of monitoring/logging system

Objective 3.3 Identify common regulations that relate to secure software development

- HIPAA
- PCI DSS
- ISO 27001
- SOX
- Country-specific privacy laws

Objective 3.4 Explain the importance of organizational requirements to the development of secure software applications

- Internal organizational processes
- Internal organizational policies
    - Password policy
        - Complexity
        - Frequency of change
    - Authentication policy
        - Password
        - Multi-factor
        - Tokens
        - Biometrics
    - Secure storage of assets
        - Source code
        - Documentation
        - Data
    - Disposal of development assets

Domain 4.0 Risk assessment and management

Objective 4.1 Classify common threats and vulnerabilities in terms of their impact on applications

- OWASP Top 10
- CWE/SANS Top 25
- Attack vectors
- Assets
    - Personal data
    - Financial data
    - Service availability
    - Reputation/brand value
    - Trust
- Risks
    - Data
        - Leaks
        - Loss
        - Corruption
    - Service disruption
        - Slow down
        - Shut down
        - Reliability
    - Repudiated transactions
- Threat types
    - Foreign governments
    - Disgruntled employee
    - Cyber terrorists
    - Cyber theft
    - Hacktivists
    - Script kiddies
- Countermeasures
    - Input validation
    - Encryption application
    - Access control
    - Secure session management
    - Controlling information disclosure
        - Metadata
        - Architectural details
        - OS details
        - Exception details
- Impacts
    - Financial
    - Reputation
    - Punitive

        - Reparations
    - Probability
        - Cost-benefit
            - Difficulty vs. payoff
        - Attack attractiveness
            - Monetary value
            - Providing additional privileges
            - Revealing user information
            - Altering authentication and authorization
            - Avoid security mechanism
        - Likelihood of getting caught

Objective 4.2 Compare and contrast common risk assessment and management best practices

- Quantitative risk assessment
    - Expected loss = impact x probability
- Qualitative risk assessment
    - MS DREAD
- Policy adjustments/updates
- Architectural review

Domain 5.0 Application Implementation

Objective 5.1 Implement input validation

- Input vulnerabilities
    - Input formatting and length
    - Input contains executable code
        - SQL
        - JavaScript
        - XML
        - LDAP
    - HTTP parameters
- Input validation techniques
    - White listing/black listing
    - Regular expressions

Objective 5.2 Restrict the output of sensitive data

- Output vulnerability
    - Visibility of underlying structure
        - Table names
        - Dumps
            - Stack traces
    - Sensitive data types
        - Lists
            - Employees

        - Users
        - Customers
    - Credit card numbers
    - Social Security numbers
    - Passwords
- Output security techniques
    - Output encoding
    - Sanitization of user-facing error messages

Objective 5.3 Implement cryptography

- Crypto libraries
- Key management
- Algorithm implementation
- Secure storage of data
    - Desktop
    - Mobile
    - Cloud
    - Web/server

Objective 5.4 Implement authentication and access control

- Password verification
- Roles, permissions, groups
- Implementation of secure session management
- Account lockouts
    - Excessive password attempts
- Password recovery

Objective 5.5 Implement error handling and logging

- Error message logging
- Security exception logging
- Log centralization

Objective 5.6 Implement communication security

- SSL
- Encrypted tunnels
    - SSH
    - IPsec
- Mobile app considerations
    - Unreliable networks
    - Slow networks
- Security of web services

Objective 5.7 Implement application security parameters and configure security settings

- Parameterizing security properties and settings

    - No hard-coded values
- Configuration file protection
- Default passwords on third-party applications

Objective 5.8 Implement secure database access

- Elimination of string concatenation for database queries
    - Parameterized queries
    - Stored procedures
- Database connection access control

Recertification Requirements

The *Cyber Secure Coder (CSC)* certification is valid for 3 years from the date that it is initially granted. You must retake the most recent version of the exam before the certification's 3-year period expires in order to maintain a continuously valid certification.