LEARNING PATHS

Software Security Role-Based Curriculum

SECURITY INNOVATION

Contents

CSC 101 – Secure Developer – Core

The Secure Developer – Core Learning Path introduces application security's fundamental and primary drivers. The curriculum provides individuals with an understanding of the importance of secure software development while preparing them to perform at the organizational level. Learners will gain in-depth knowledge of security principles, attacks, tools, and processes to develop secure software. By introducing the OWASP Top 10, learners are prepared to identify the most critical web application security risks, appropriately address those vulnerabilities, and prevent software flaws that enable cyberattacks.

Upon successful completion of this path, you will have the knowledge and skills to:

- Define the value of having secure applications
- Integrate secure software development practices into all phases of the software development lifecycle
- Explain the anatomy of an application attack
- Apply best practices to protect all components of the software
- Identify and mitigate the most common application security risks
- Implement a security strategy based on your organization's risk
- Produce well-secured software

NOTE: This Learning Path is considered principal to all Elite Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

19 Courses

14 Labs

5 Hours

7 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 102 – The Role of Software Security
- COD 103 – Creating Software Security Requirements
- COD 104 – Designing Secure Software
- COD 105 – Secure Software Development
- COD 106 – The Importance of Integration and Testing
- COD 107 – Secure Software Deployment
- COD 108 – Software Operations and Maintenance
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 Insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control

- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Mitigating Insecure Deserialization
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 113 – Identifying Cryptographic Failures
- LAB 115 – Identifying Reflective XSS
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection
- LAB 121 – Identifying Vulnerable and Outdated Components
- LAB 127 – Identifying Security Logging and Monitoring Failures
- LAB 129 – Identifying Error Message Containing Sensitive Information
- LAB 133 – Identifying Exposure of Sensitive Information Through Environmental Variables

Elective

- DES 101 – Fundamentals of Secure Architecture
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 201 – Secure Developer – Advanced

The Secure Developer – Advanced Learning Path explores different models, standards, frameworks, and security concepts that you can use to understand security issues and improve the security posture of your applications. The curriculum provides individuals with an understanding of how to ensure security is part of software design. Learners will gain in-depth knowledge of security practices that must be considered within every phase of the development lifecycle to help secure software applications and data. By introducing the DevSecOps philosophies, learners are prepared to focus on time saving but effective techniques that maximize security resources all while shortening system development lifecycles and providing continuous delivery of high-quality software.

Upon successful completion of this path, you will have the knowledge and skills to:

- Use NIST and MITE ATT&CK security frameworks to identify and categorize potential threats
- Identify and apply relevant cryptographic technologies to secure applications and data
- Apply techniques to remove architecture weak spots and avoid vulnerability propagation
- Implement a zero-trust architecture
- Create a threat model for application scenarios
- Manage identities, privileges, and secrets securely
- Understand, create, and articulate security requirements as part of a software requirement document
- Use NIST and MITE ATT&CK security frameworks to identify, categorize, and respond to potential threats
- Determine which types of automated tests should be performed at various stages of the SDLC

NOTE: This Learning Path is considered principal to all Elite Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

13 Courses

0 Labs

6 Hours

7 CPE Credits

Advanced

- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 204 – The Role of Cryptography in Application Development
- DES 212 – Architecture Risk Analysis and Remediation
- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- CYB 310 – Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- DES 311 – Creating Secure Application Architecture

- DSO 302 – Automated Security Testing
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- ENG 320 – Using the Software Composition Analysis (SCA) to Secure Open Source Components

Elective

- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 301 – Elite Secure Developer – JAVA

The Elite Secure Developer – Java Learning Path includes a variety of security courses for those who design, develop, and host Java applications, including AJAX, Node.js, SAP ADAP, and Ruby on Rails. The learning path is designed to provide a working knowledge for developing solid and secure Java applications as well as recognizing and remediating common Java web software security vulnerabilities. The Elite Secure Developer – Java Learning Path covers key application security concepts, including:

- Best practices for designing, developing, and testing Java-based solutions using common standards and frameworks
- Java, JRE, and J2EE constructs
- Core implementation practices

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

6
Courses

17
Labs

7
Hours

8
CPE Credits

Elite

- COD 219 – Creating Secure Code - SAP ABAP Foundations
- COD 259 – Node.js Threats and Vulnerabilities
- COD 283 – Java Cryptography
- COD 284 – Secure Java Coding
- COD 287 – Java Application Server Hardening
- COD 288 – Java Public Key Cryptography
- LAB 211 – Defending Java Applications Against Credentials in Code Medium
- LAB 215 – Defending Java Applications Against Business Logic Error for Input Validation
- LAB 224 – Defending Java Applications Against Forceful Browsing
- LAB 228 – Defending Java Applications Against Weak AES ECB Mode Encryption
- LAB 229 – Defending Java Applications Against Weak PRNG
- LAB 230 – Defending Java Applications Against XSS
- LAB 234 – Defending Java Applications Against Parameter Tampering
- LAB 235 – Defending Java Applications Against Plaintext Password Storage
- LAB 236 – Defending Java Applications Against Sensitive Information in Error Messages
- LAB 237 – Defending Java Applications Against SQL Injection
- LAB 240 – Defending Java Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 244 – Defending Java Against Security Misconfiguration
- LAB 263 – Defending Java Applications Against Sensitive Information in Log Files
- LAB 267 – Defending Java Applications Against Deserialization of Untrusted Data
- LAB 271 – Defending Java Applications Against SSRF

- LAB 275 – Defending Java Applications Against Command Injection
- LAB 279 – Defending Java Applications Against Dangerous File Upload
- LAB 283 – Defending Java Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 251 – Defending AJAX- Enabled Web Applications
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- COD 352 – Creating Secure Java Script and jQuery
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- COD 380 – Preventing SQL Injection in Java
- COD 381 – Preventing Path Traversal Attacks in Java
- COD 382 – Protecting Data in Java
- COD 383 – Protecting Java Backend Services
- COD 384 – Protecting Java from Information Disclosure
- COD 385 – Preventing Race Conditions in Java Code
- COD 386 – Preventing Integer Overflows in Java Code
- DES 219 – Securing Google's Firebase Platform
- SDT 325 – Testing for NULL Pointer Dereference
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services

- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 302 – Elite Secure Developer – Python

The Elite Secure Developer – Python Learning Path includes a variety of security courses for those responsible for the programming and development of web applications or applications that are run over HTTP from a web server to a web browser. The Python Web Developer learning path covers key application security concepts, including:

- Secure coding best practices
- Effective platform configuration
- How to identify and mitigate vulnerabilities

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

4 Courses

14 Labs

5 Hours

6 CPE Credits

Elite

- COD 257 – Creating Secure Python Web Applications
- COD 265 – Secure Python Scripting
- COD 267 – Securing Python Microservices
- DSO 306 – Implementing Infrastructure as Code
- LAB 212 – Defending Python Applications Against Credentials in Code Medium
- LAB 216 – Defending Python Applications Against Business Logic Error for Input Validation
- LAB 222 – Defending Python Applications Against SQL Injection
- LAB 225 – Defending Python Applications Against Forceful Browsing
- LAB 231 – Defending Python Applications Against XSS
- LAB 243 – Defending Python Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 249 – Defending Python Applications Against Plaintext Password Storage
- LAB 252 – Defending Python Applications Against Weak AES ECB Mode Encryption
- LAB 253 – Defending Python Applications Against Weak PRNG
- LAB 254 – Defending Python Applications Against Parameter Tampering
- LAB 261 – Defending Python Applications Against Sensitive Information in Error Messages
- LAB 264 – Defending Python Applications Against Sensitive Information in Log Files
- LAB 268 – Defending Python Applications Against Deserialization of Untrusted Data
- LAB 272 – Defending Python Applications Against SSRF
- LAB 276 – Defending Python Applications Against Command Injection
- LAB 280 – Defending Python Applications Against Dangerous File Upload
- LAB 284 – Defending Python Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 303 – Elite Secure Developer – C#

The Elite Secure Developer – C# Learning Path builds on a thorough grounding of security features necessary to avoid common pitfalls and build scalable applications that run on desktops or back-end processes, powering modern web applications using defensive coding practices throughout the software development life cycle (SDLC). The curriculum equips developers with the competency to develop desktop, API, and back-end applications written in C#. The Elite Secure Developer – C# covers key application security concepts, including:

- Defensive coding best practices
- Developing scalable applications using multithreading features of the .NET framework
- Avoiding common pitfalls

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8 Courses

17 Labs

7 Hours

8 CPE Credits

Elite

- COD 216 – Leveraging .NET Framework Code Access Security (CAS)
- COD 217 – Mitigating .NET Security Threats
- COD 308 – Common ASP.NET Vulnerabilities and Attacks
- COD 309 – ASP.NET MVC Authentication and Authorization
- COD 321 – Protecting C# from Integer Overflows and Canonicalization Issues
- COD 322 – Protecting C# from SQL Injection
- COD 323 – Using Encryption with C#
- COD 324 – Protecting C# from XML Injection
- LAB 214 – Defending C# Applications Against Credentials in Code Medium
- LAB 218 – Defending C# Applications Against Business Logic Error for Input Validation
- LAB 221 – Defending C# Applications Against SQL Injection
- LAB 227 – Defending C# Applications Against Forceful Browsing
- LAB 232 – Defending C# Applications Against XSS
- LAB 238 – Defending C# Applications Against Weak AES ECB Mode Encryption
- LAB 239 – Defending C# Applications Against Weak PRNG
- LAB 241 – Defending C# Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 250 – Defending C# Applications Against Parameter Tampering
- LAB 251 – Defending C# Applications Against Plaintext Password Storage
- LAB 260 – Defending C# Applications Against Sensitive Information in Error Messages
- LAB 263 – Defending C# Applications Against SSRF
- LAB 266 – Defending C# Applications Against Sensitive Information in Log Files

- LAB 270 – Defending C# Applications Against Deserialization of Untrusted Data
- LAB 278 – Defending C# Applications Against Command Injection
- LAB 282 – Defending C# Applications Against Dangerous File Upload
- LAB 286 – Defending C# Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- SDT 325 – Testing for NULL Pointer Dereference
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 304 – Elite Secure Developer – Node.js

The Elite Secure Developer – Node.js Learning Path includes various security courses to manage the data interchange between the server and users securely. The curriculum path provides a solid foundation of security features necessary to code defensively for this JavaScript run-time environment. The Elite Secure Developer – Node.js Learning Path equips developers with the knowledge and skills to manage:

- Node.js based services
- Web libraries, frameworks, and the whole web stack
- Protecting data using secure coding best practices

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

4 Courses

31 Labs

8 Hours

9 CPE Credits

Elite

- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats and Vulnerabilities
- COD 308 – Common ASP.NET Vulnerabilities and Attacks
- COD 309 – Securing ASP.NET MVC Applications
- LAB 213 – Defending Node.js Applications Against Credentials in Code Medium
- LAB 217 – Defending Node.js Applications Against Business Logic Error for Input Validation
- LAB 222 – Defending Python Applications Against SQL Injection
- LAB 223 – Defending Node.js Applications Against SQL Injection
- LAB 226 – Defending Node.js Applications Against Forceful Browsing
- LAB 231 – Defending Python Applications Against XSS
- LAB 233 – Defending Node.js Applications Against XSS
- LAB 242 – Defending Node.js Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 243 – Defending Python Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption
- LAB 247 – Defending Node.js Applications Against Weak PRNG
- LAB 248 – Defending Node.js Applications Against Parameter Tampering
- LAB 249 – Defending Python Applications Against Plaintext Password Storage
- LAB 252 – Defending Python Applications Against Weak AES ECB Mode Encryption
- LAB 253 – Defending Python Applications Against Weak PRNG
- LAB 254 – Defending Python Applications Against Parameter Tampering
- LAB 261 – Defending Python Applications Against Sensitive Information in Error Messages
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages

- LAB 264 – Defending Python Applications Against Sensitive Information in Log Files
- LAB 268 – Defending Python Applications Against Deserialization of Untrusted Data
- LAB 272 – Defending Python Applications Against SSRF
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 276 – Defending Python Applications Against Command Injection
- LAB 277 – Defending Node.js Applications Against Command Injection
- LAB 280 – Defending Python Applications Against Dangerous File Upload
- LAB 281 – Defending Node.js Applications Against Dangerous File Upload
- LAB 284 – Defending Python Applications Against RegEx DoS
- LAB 285 – Defending Node.js Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 241 – Creating Secure Oracle Database Applications
- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- COD 257 – Creating Secure Python Web Applications
- COD 285 – Developing Secure Angular Applications
- DES 207 – Mitigating OWASP API Security Top 10
- COD 352 – Creating Secure JavaScript and jQuery Code
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- DES 219 – Securing Google's Firebase Platform
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures
- SDT 306 – Testing for Security Misconfiguration
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers

- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 305 – Elite Secure Developer – Back-End

The Elite Secure Developer – Back-End Learning Path includes a variety of security courses that provide a solid foundation of security features needed to write web services and APIs used by front-end and mobile application developers. The curriculum presents secure coding best practices in all phases of the development life cycle across cutting-edge technologies like Node.js, Angular.js, and MySQL with special attention to managing the interchange of data between the server and users. The Elite Secure Developer – Back-End Learning Path equips developers with the knowledge and skills to:

- Design, develop, and host back-end APIs, databases, and other services, including JavaScript, jQuery, AJAX, Oracle, and Python microservices
- Handle common application security vulnerabilities
- Manage data interchange between servers and users

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

14 Courses

2 Labs

3 Hours

4 CPE Credits

Elite

- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- COD 241 – Creating Secure Oracle Database Applications
- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- COD 287 – Java Application Server Hardening
- COD 288 – Java Public Key Cryptography
- COD 352 – Creating Secure JavaScript and jQuery Code
- COD 383 – Protecting Java Backend Services
- DES 207 – Mitigating OWASP API Security Top 10
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- LAB 122 – Identifying Insecure APIs
- LAB 132 – Identifying Exposed Services

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 267 – Securing Python Microservices
- DES 219 – Securing Google's Firebase Platform
- DES 261 – Securing Serverless Environments
- DES 262 – Securing Enterprise Low-Code Application Platforms
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- SDT 303 – Testing for Cryptographic Failures
- SDT 304 – Testing for Insecure Design
- SDT 306 – Testing for Security Misconfiguration
- SDT 307 – Testing for Server-Side Request Forgery (SSRF)
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 306 – Elite Secure Developer – Front-End

The Elite Secure Developer – Front-End Learning Path introduces the critical security skills and knowledge required to apply secure coding best practices in all phases of the software development life cycle (SDLC). The curriculum provides learners with the competency required to use markup languages, write client-side scripts, and create secure environments for everything user's touch. The Elite Secure Developer – Front-End Learning Path covers key application security concepts, including:

- Deep dive into HTML, CSS, and responsive web development
- How vulnerabilities are discovered and exploited
- How to build a strong line of defense

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8
Courses

19
Labs

6
Hours

8
CPE Credits

Elite

- COD 214 – Creating Secure GO Applications
- COD 285 – Developing Secure Angular Applications
- COD 286 – Creating Secure React User Interfaces
- COD 287 – Java Application Server Hardening
- COD 288 – Java Public Key Cryptography
- COD 352 – Creating Secure JavaScript and jQuery Code
- COD 383 – Protecting Java Backend Services
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- LAB 122 – Identifying Insecure APIs
- LAB 132 – Identifying Exposed Services
- LAB 213 – Defending Node.js Applications Against Credentials in Code Medium
- LAB 217 – Defending Node.js Applications Against Business Logic Error for Input Validation
- LAB 223 – Defending Node.js Applications Against SQL Injection
- LAB 226 – Defending Node.js Applications Against Forceful Browsing
- LAB 233 – Defending Node.js Against XSS
- LAB 242 – Defending Node.js Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption
- LAB 247 – Defending Node.js Applications Against Weak PRNG
- LAB 248 – Defending Node.js Applications Against Parameter Tampering
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files

- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 277 – Defending Node.js Applications Against Command Injection
- LAB 281 – Defending Node.js Applications Against Dangerous File Upload
- LAB 285 – Defending Node.js Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats and Vulnerabilities
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- DES 262 – Securing Enterprise Low-Code Application Platforms
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration

- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 307 – Elite Secure Developer – Web

The Elite Secure Developer – Web Learning Path includes a variety of courses that provide the critical security skills and knowledge required to apply secure coding best practices in all phases of the software development life cycle (SDLC). The curriculum provides learners with the competency required to develop applications that run over HTTP from a web server to a web browser. The Elite Secure Developer – Web Learning Path provides developers with a solid foundation of security features necessary to develop applications, including:

- Responsive web design
- Enterprise integration
- How to protect data with security best practices

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8 Courses

31 Labs

8 Hours

9 CPE Credits

Elite

- COD 258 – Creating Secure PHP Web Applications
- COD 285 – Developing Secure Angular Applications
- COD 287 – Java Application Server Hardening
- COD 288 – Java Public Key Cryptography
- COD 352 – Creating Secure JavaScript and jQuery Code
- COD 383 – Protecting Java Backend Services
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- LAB 220 – Defending Against Hard-coded Secrets
- LAB 223 – Defending Node.js Applications Against SQL Injection
- LAB 228 – Defending Java Applications Against Weak AES ECB Mode Encryption
- LAB 229 – Defending Java Applications Against Weak PRNG
- LAB 230 – Defending Java Applications Against XSS
- LAB 233 – Defending Node.js Applications Against XSS
- LAB 234 – Defending Java Applications Against Parameter Tampering
- LAB 235 – Defending Java Applications Against Plaintext Password Storage
- LAB 236 – Defending Java Applications Against Sensitive Information in Error Message
- LAB 237 – Defending Java Applications Against SQL Injection
- LAB 240 – Defending Java Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 242 – Defending Node.js Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 244 – Defending Java Against Security Misconfiguration
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage

- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption
- LAB 247 – Defending Node.js Applications Against Weak PRNG
- LAB 248 – Defending Node.js Applications Against Parameter Tampering
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages
- LAB 263 – Defending Java Applications Against Sensitive Information in Log Files
- LAB 267 – Defending Java Applications Against Deserialization of Untrusted Data
- LAB 271 – Defending Java Applications Against SSRF
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 137 – Identifying Improper Authorization
- LAB 138 – Identifying Authorization Bypass Through User-Controlled Key
- LAB 139 – Identifying Use of a Key Past its Expiration Date
- LAB 275 – Defending Java Applications Against Command Injection
- LAB 277 – Defending Node.js Applications Against Command Injection
- LAB 279 – Defending Java Applications Against Dangerous File Upload
- LAB 281 – Defending Node.js Against Dangerous File Upload
- LAB 283 – Defending Java Applications Against RegEx DoS
- LAB 285 – Defending Node.js Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- COD 241 – Creating Secure Oracle Database Applications
- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- COD 257 – Creating Secure Python Web Applications
- COD 259 – Node.js Threats and Vulnerabilities
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- DES 207 – Mitigating OWASP API Security Top 10
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 303 – Testing for Cryptographic Failures
- SDT 304 – Testing for Insecure Design
- SDT 305 – Testing for Broken Access Control
- SDT 306 – Testing for Security Misconfiguration
- SDT 307 – Testing for Server-Side Request Forgery (SSRF)
- SDT 308 – Testing for Software and Data Integrity Failures

- SDT 309 – Testing for Vulnerable and Outdated Components
- SDT 310 – Testing for Security Logging and Monitoring Failures
- SDT 313 – Testing for Cross-Site Request Forgery (CSRF)
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 137 – Identifying Improper Authorization
- LAB 138 – Identifying Authorization Bypass Through User-Controlled Key
- LAB 139 – Identifying Use of a Key Past its Expiration Date
- LAB 276 – Defending Python Applications Against Command Injection
- LAB 278 – Defending C# Applications Against Command Injection
- LAB 280 – Defending Python Applications Against Dangerous File Upload
- LAB 282 – Defending C# Applications Against Dangerous File Upload
- LAB 284 – Defending Python Applications Against RegEx DoS
- LAB 286 – Defending C# Applications Against RegEx DoS
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 308 – Elite Secure Developer – Mobile

The Elite Secure Developer – Mobile Learning Path includes a variety of courses that provides the knowledge and skill required to apply secure coding best practices in all phases of the software development life cycle (SDLC). The curriculum provides developers with a solid foundation of security features necessary to develop applications for mobile devices. The Elite Secure Developer – Mobile Learning Path covers key application security concepts including:

- Identifying common mobile application risks
- Best practices for designing secure mobile applications
- Coding mistakes to avoid

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

10
Courses

0
Labs

5
Hours

5
CPE Credits

Elite

- COD 110 – Fundamentals of Secure Mobile Development
- ENG 112 – Essential Access Control for Mobile Devices
- COD 261 – Threats to Scripts
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- COD 315 – Preventing Vulnerabilities in iOS Code in Swift
- COD 316 – Creating Secure iOS Code in Objective C
- COD 317 – Protecting Data on iOS in Swift
- COD 318 – Protecting Data on Android in Java
- COD 319 – Preventing Vulnerabilities in Android Code in Java

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 286 – Creating Secure React User Interfaces
- DES 207 – Mitigating OWASP API Security Top 10
- DES 219 – Securing Google's Firebase Platform
- DES 271 – OWASP M1: Mitigating Improper Platform Usage
- DES 272 – OWASP M2: Mitigating Insecure Data Storage
- DES 273 – OWASP M3: Mitigating Insecure Communication
- DES 274 – OWASP M4: Mitigating Insecure Authentication
- DES 275 – OWASP M5: Mitigating Insufficient Cryptography
- DES 276 – OWASP M6: Mitigating Insecure Authorization
- DES 277 – OWASP M7: Mitigating Client Code Quality

- DES 278 – OWASP M8: Mitigating Code Tampering
- DES 279 – OWASP M9: Mitigating Reverse Engineering
- DES 280 – OWASP M10: Mitigating Extraneous Functionality
- DES 284 – OWASP IoT4: Mitigating Lack of Secure Update Mechanism
- DES 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 288 – OWASP IoT8: Mitigating Lack of Device Management
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- COD 366 – Creating Secure Kotlin Applications
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 305 – Testing for Broken Access Control
- SDT 316 – Testing for Use of Hard-coded Credentials
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 309 – Elite Secure Developer – Cloud

The Elite Secure Developer – Cloud Learning Path is designed for those responsible for the design, development, and deployment of cloud applications. The curriculum consists of a variety of courses that provide the knowledge and skills required to apply secure coding best practices in all phases of cloud application and platform development. Learners will gain a clear understanding of how to mitigate cloud computing risks. The Elite Secure Developer – Cloud Learning Path covers key application security topics, including:

- Big Data" and it introduces security challenges
- Cloud computing characteristics, service and deployment models, and regulatory requirements
- Platform-specific secure coding best practices including AWS, Azure, and/or GCP

NOTE: Secure Developer – Core and Advanced Learning paths are considered principal to all Elite Secure Developer Learning Paths. All Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8 Courses

0 Labs

3 Hours

4 CPE Credits

Elite

- COD 152 – Fundamentals of Secure Cloud Development
- COD 267 –Securing Python Microservices
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 215 – Defending Infrastructure
- DES 216 – Defending Cloud Infrastructure
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine

Elective

- DES 101 – Fundamentals of Secure Architecture
- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- API 250 – Controlling Access to the Kubernetes API
- COD 214 – Creating Secure GO Applications
- COD 241 – Creating Secure Oracle Database Applications
- COD 252 – Securing Google Platform Applications & Data
- COD 253 – Creating Secure AWS Cloud Applications
- COD 254 – Creating Secure Azure Applications
- COD 255 – Creating Secure Code - Web API Foundations

- COD 259 – Node.js Threats and Vulnerabilities
- COD 261 – Threats to Scripts
- DES 207 – Mitigating OWASP API Security Top 10
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 209 – Authentication and Lifecycle Management
- DES 214 – Securing Infrastructure Architecture
- DES 219 – Securing Google's Firebase Platform
- DES 261 – Securing Serverless Environments
- DES 281 – OWASP IoT1: Mitigating Weak, Guessable or Hardcoded Passwords
- DES 282 – OWASP IoT2: Mitigating Insecure Network Services
- DES 283 – OWASP IoT3: Mitigating Insecure Ecosystem Interfaces
- DES 284 – OWASP IoT4: Mitigating Lack of Secure Update Mechanism
- DES 285 – OWASP IoT5: Mitigating Use of Insecure or Outdated Components
- DES 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 288 – OWASP IoT8: Mitigating Lack of Device Management
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- DES 290 – OWASP IoT10 Mitigating Lack of Physical Hardening
- DSO 211 – Identifying threats to containers in a DevSecOps Framework
- DSO 253 – DevSecOps in the AWS cloud
- DSO 254 – DevSecOps in the azure cloud
- DSO 256 – DevSecOps in the Google Cloud Platform
- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 305 – Attack Surface Analysis & Reduction
- DSO 306 – Implementing Infrastructure as Code
- ENG 311 – Attack Surface Analysis & Reduction
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 137 – Identifying Improper Authorization
- LAB 138 – Identifying Authorization Bypass Through User-Controlled Key
- LAB 139 – Identifying Use of a Key Past its Expiration Date
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software

- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts

CSC 310 – Elite Secure Developer – Ruby on Rails

The Elite Secure Developer –Ruby on Rails Learning Path includes a variety of security courses that are designed for those responsible for writing server-side web application logic in Ruby, around the frame rails. The curriculum provides best practices and techniques for secure application development.

Upon successful completion of this path, you will have the knowledge and skills to:

- Understand various classes of vulnerabilities
- Build strong session management
- Prevent vulnerabilities commonly found in Rails applications

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

14 Courses

14 Labs

8 Hours

10 CPE Credits

Elite

- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- DES 207 – Mitigating OWASP API Security Top 10
- COD 352 – Creating Secure JavaScript and jQuery Code
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 306 – Implementing Infrastructure as Code
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures
- COD 259 – Node.js Threats and Vulnerabilities
- LAB 223 – Defending Node.js Applications Against SQL Injection
- LAB 233 – Defending Node.js Applications Against XSS
- LAB 242 – Defending Node.js Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption
- LAB 247 – Defending Node.js Applications Against Weak PRNG
- LAB 248 – Defending Node.js Applications Against Parameter Tampering
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages

- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 277 – Defending Node.js Applications Against Command Injection
- LAB 281 – Defending Node.js Applications Against Dangerous File Upload
- LAB 285 – Defending Node.js Applications Against RegEx DoS

Elective

- COD 257 – Creating Secure Python Web Applications
- COD 283 – Java Cryptography
- COD 284 – Secure Java Coding
- COD 287 – Java Application Server Hardening
- DES 101 – Fundamentals of Secure Architecture
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 222 – Defending Python Applications Against SQL Injection
- LAB 228 – Defending Java Applications Against Weak AES ECB Mode Encryption
- LAB 229 – Defending Java Applications Against Weak PRNG
- LAB 230 – Defending Java Applications Against XSS
- LAB 231 – Defending Python Applications Against XSS
- LAB 234 – Defending Java Applications Against Parameter Tampering
- LAB 235 – Defending Java Applications Against Plaintext Password Storage
- LAB 236 – Defending Java Applications Against Sensitive Information in Error Messages
- LAB 237 – Defending Java Applications from SQL Injection
- LAB 240 – Defending Java Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 243 – Defending Python Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 244 – Defending Java Applications Against Security Misconfiguration
- LAB 249 – Defending Python Applications Against Plaintext Password Storage
- LAB 252 – Defending Python Applications Against Weak AES ECB Mode Encryption

- LAB 253 – Defending Python Applications Against Weak PRNG
- LAB 254 – Defending Python Applications Against Parameter Tampering
- LAB 261 – Defending Python Applications Against Sensitive Information in Error Messages
- LAB 263 – Defending Java Applications Against Sensitive Information in Log Files
- LAB 264 – Defending Python Applications Against Sensitive Information in Log Files
- LAB 267 – Defending Java Applications Against Deserialization of Untrusted Data
- LAB 268 – Defending Python Applications Against Deserialization of Untrusted Data
- LAB 271 – Defending Java Applications Against SSRF
- LAB 272 – Defending Python Applications Against SSRF
- LAB 275 – Defending Java Applications Against Command Injection
- LAB 276 – Defending Python Applications Against Command Injection
- LAB 279 – Defending Java Applications Against Dangerous File Upload
- LAB 280 – Defending Python Applications Against Dangerous File Upload
- LAB 283 – Defending Java Applications Against RegEx DoS
- LAB 284 – Defending Python Applications Against RegEx DoS

CSC 311 – Elite Secure Developer – C++

The Elite Secure Developer – C++Learning Path includes a variety of security courses designed to provide a continuous working knowledge of application security best practices for building applications that range from desktop applications to native mobile applications and embedded systems. The curriculum provides the knowledge needed to build efficient, reusable, and reliable C++ code that interacts with low-level systems and hardware resources.

Upon successful completion of this path, you will have the knowledge and skills to:

- Mitigate memory corruption vulnerabilities
- Protect data in transit using strong TLS ciphers
- Protect data using cryptographic best practices while applying secure coding best practices

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8
Courses

0
Labs

3
Hours

4
CPE Credits

Elite

- COD 206 – Creating Secure C++ Code
- COD 207 – Communication Security in C++
- COD 255 – Creating Secure Code - Web API Foundations
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- DES 203 – Cryptographic Components: Randomness, Algorithms, and Key Management
- DES 207 – Mitigating OWASP API Security Top 10
- DES 219 – Securing Google's Firebase Platform
- COD 307 – Protecting Data in C++

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 263 – Secure Bash Scripting
- COD 264 – Secure Perl Scripting
- COD 265 – Secure Python Scripting
- COD 266 – Secure Ruby Scripting
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 319 – Testing for Out-of-bounds Read
- SDT 320 – Testing for Out-of-bounds Write
- SDT 324 – Testing for Improper Restriction of Operations within the Bounds of a Memory Buffer
- SDT 325 – Testing for NULL Pointer Dereference

- SDT 326 – Testing for Use After Free
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 312 – Elite Secure Developer – PHP

The Elite Secure Developer – PHP Learning Path includes a variety of security courses designed to provide PHP developers with a solid foundation of security features necessary to develop server-side web application logic. The PHP learning path offers secure coding best practices to develop back-end web services connection components and support frontend developers.

Upon successful completion of this path, you will have the knowledge and skills to:

- Apply these security best practices to the entire web application development life cycle from concept stage to delivery and post-launch

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

13
Courses

14
Labs

8
Hours

10
CPE Credits

Elite

- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats and Vulnerabilities
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- DES 207 – Mitigating OWASP API Security Top 10
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- DES 219 – Securing Google's Firebase Platform
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- LAB 223 – Defending Node.js Applications Against SQL Injection
- LAB 233 – Defending Node.js Applications Against XSS
- LAB 242 – Defending Node.js Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption
- LAB 247 – Defending Node.js Applications Against Weak PRNG
- LAB 248 – Defending Node.js Applications Against Parameter Tampering
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data

- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 277 – Defending Node.js Applications Against Command Injection
- LAB 281 – Defending Node.js Against Dangerous File Upload
- LAB 285 – Defending Node.js Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- COD 263 – Secure Bash Scripting
- COD 264 – Secure Perl Scripting
- COD 265 – Secure Python Scripting
- COD 266 – Secure Ruby Scripting
- COD 283 – Java Cryptography
- COD 284 – Secure Java Coding
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 303 – Testing for Cryptographic Failures
- SDT 304 – Testing for Insecure Design
- SDT 305 – Testing for Broken Access Control
- SDT 306 – Testing for Security Misconfiguration
- SDT 307 – Testing for Server-Side Request Forgery (SSRF)
- SDT 308 – Testing for Software and Data Integrity Failures
- SDT 309 – Testing for Vulnerable and Outdated Components
- SDT 310 – Testing for Security Logging and Monitoring Failures
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 228 – Defending Java Applications Against Weak AES ECB Mode Encryption
- LAB 229 – Defending Java Applications Against Weak PRNG

- LAB 230 – Defending Java Applications Against XSS
- LAB 234 – Defending Java Applications Against Parameter Tampering
- LAB 235 – Defending Java Applications Against Plaintext Password Storage
- LAB 236 – Defending Java Applications Against Sensitive Information in Error Messages
- LAB 237 – Defending Java Applications from SQL Injection
- LAB 240 – Defending Java Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 244 – Defending Java Applications Against Security Misconfiguration
- LAB 263 – Defending Java Applications Against Sensitive Information in Log Files
- LAB 267 – Defending Java Applications Against Deserialization of Untrusted Data
- LAB 271 – Defending Java Applications Against SSRF
- LAB 275 – Defending Java Applications Against Command Injection
- LAB 279 – Defending Java Applications Against Dangerous File Upload
- LAB 283 – Defending Java Applications Against RegEx DoS

CSC 313 – Elite Secure Developer – JavaScript

The Elite Secure Developer – JavaScript Learning Path includes a variety of security courses intended for those responsible for implementing the front-end logic that defines the behavior of the visual elements of a web application and connecting this with services that may reside on the back end. The curriculum provides a thorough grounding in application security concepts and implementation practices.

Upon successful completion of this path, you will have the knowledge and skills to:

- Mitigate JavaScript security flaws
- Apply proven techniques to help protect JavaScript
- Avoid common pitfalls

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

13
Courses

14
Labs

8
Hours

10
CPE Credits

Elite

- COD 241 – Creating Secure Oracle Database Applications
- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats and Vulnerabilities
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- DES 207 – Mitigating OWASP API Security Top 10
- DES 219 – Securing Google's Firebase Platform
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- LAB 223 – Defending Node.js Applications Against SQL Injection
- LAB 233 – Defending Node.js Applications Against XSS
- LAB 242 – Defending Node.js Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption
- LAB 247 – Defending Node.js Applications Against Weak PRNG
- LAB 248 – Defending Node.js Applications Against Parameter Tampering
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files

- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 277 – Defending Node.js Applications Against Command Injection
- LAB 281 – Defending Node.js Against Dangerous File Upload
- LAB 285 – Defending Node.js Applications Against RegEx DoS

Elective

- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- DES 101 – Fundamentals of Secure Architecture
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures
- SDT 306 – Testing for Security Misconfiguration
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 314 – Elite Secure Developer – iOS

The Elite Secure Developer – iOS Learning Path includes a variety of security courses designed to provide developers with a solid foundation of security features necessary to develop applications for devices powered by the iOS platform. The curriculum provides secure coding best practices for designing and building iOS applications.

Upon successful completion of this path, you will have the knowledge and skills to:

- Identifying common iOS application risks
- Creating a mobile application threat model

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

9
Courses

0
Labs

5
Hours

6
CPE Credits

Elite

- COD 110 – Fundamentals of Secure Mobile Development
- ENG 112 – Essential Access Control for Mobile Devices
- COD 261 – Threats to Scripts
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- COD 315 – Preventing Vulnerabilities in iOS Code in Swift
- COD 316 – Creating Secure iOS Code in Objective C
- COD 317 – Protecting Data on iOS in Swift

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 286 – Creating Secure React User Interfaces
- DES 207 – Mitigating OWASP API Security Top 10
- DES 219 – Securing Google's Firebase Platform
- DES 271 – OWASP M1: Mitigating Improper Platform Usage
- DES 272 – OWASP M2: Mitigating Insecure Data Storage
- DES 273 – OWASP M3: Mitigating Insecure Communication
- DES 274 – OWASP M4: Mitigating Insecure Authentication
- DES 275 – OWASP M5: Mitigating Insufficient Cryptography
- DES 276 – OWASP M6: Mitigating Insecure Authorization
- DES 277 – OWASP M7: Mitigating Client Code Quality
- DES 278 – OWASP M8: Mitigating Code Tampering
- DES 279 – OWASP M9: Mitigating Reverse Engineering
- DES 280 – OWASP M10: Mitigating Extraneous Functionality

- DES 284 – OWASP IoT4: Mitigating Lack of Secure Update Mechanism
- DES 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 288 – OWASP IoT8: Mitigating Lack of Device Management
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- COD 366 – Creating Secure Kotlin Applications
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 305 – Testing for Broken Access Control
- SDT 316 – Testing for Use of Hard-coded Credentials
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 315 – Elite Secure Developer – HTML5

The Elite Secure Developer – HTML5 Learning Path includes a variety of security courses designed to provide front-end developers responsible for holding the style and interactivity backbone together with a deeper understanding of HTML5 – and building a strong line of defense. The curriculum covers key application security concepts.

Upon successful completion of this path, you will have the knowledge and skills to:

- Implement HTML5 security features
- Infuse software security into the development lifecycle
- Understand ASP.net, SWL, high-level scripting languages, version control and CMS systems

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

15
Courses

29
Labs

12
Hours

14
CPE Credits

Elite

- COD 251 – Defending AJAX-enabled Web Applications
- COD 255 – Creating Secure Code - Web API Foundations
- COD 259 – Node.js Threats and Vulnerabilities
- COD 285 – Developing Secure Angular Applications
- COD 287 – Java Application Server Hardening
- COD 288 – Java Public Key Cryptography
- COD 308 – Common ASP.NET Vulnerabilities and Attacks
- COD 309 – ASP.NET MVC Authentication and Authorization
- COD 352 – Creating Secure JavaScript and jQuery Code
- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built-In Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Security HTML5 Connectivity
- COD 383 – Protecting Java Backend Services
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field

- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 223 – Defending Node.js Applications Against SQL Injection
- LAB 228 – Defending Java Applications Against Weak AES ECB Mode Encryption
- LAB 229 – Defending Java Applications Against Weak PRNG
- LAB 230 – Defending Java Applications Against XSS
- LAB 233 – Defending Node.js Applications Against XSS
- LAB 234 – Defending Java Applications Against Parameter Tampering
- LAB 235 – Defending Java Applications Against Plaintext Password Storage
- LAB 236 – Defending Java Applications Against Sensitive Information in Error Messages
- LAB 237 – Defending Java Applications from SQL Injection
- LAB 240 – Defending Java Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 242 – Defending Node.js Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 244 – Defending Java Applications Against Security Misconfiguration
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption
- LAB 247 – Defending Node.js Applications Against Weak PRNG
- LAB 248 – Defending Node.js Applications Against Parameter Tampering
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages
- LAB 263 – Defending Java Applications Against Sensitive Information in Log Files
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files
- LAB 267 – Defending Java Applications Against Deserialization of Untrusted Data
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data
- LAB 271 – Defending Java Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 275 – Defending Java Applications Against Command Injection
- LAB 277 – Defending Node.js Applications Against Command Injection
- LAB 279 – Defending Java Applications Against Dangerous File Upload
- LAB 281 – Defending Node.js Against Dangerous File Upload
- LAB 283 – Defending Java Applications Against RegEx DoS
- LAB 285 – Defending Node.js Applications Against RegEx DoS

Elective

- COD 256 – Creating Secure Code - Ruby on Rail Foundations
- DES 101 – Fundamentals of Secure Architecture
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures

CSC 316 – Elite Secure Developer – Microsoft SDL

The Elite Secure Developer – Microsoft SDL Learning Path includes a variety of security courses designed for those responsible for implementing the industry-leading software security assurance process. The curriculum describes how to take a holistic and practical approach to ensure security and privacy is considered at every phase of development.

Upon successful completion of this path, you will have the knowledge and skills to:

- Implement the Agile MS SDL
- Implement the Microsoft SDL Optimization Model
- Apply the MS SDL Line of Business for internal or business-facing applications
- Leverage the Microsoft SDL Threat Modeling Tool and enumerate threats

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

9 Courses

0 Labs

5 Hours

6 CPE Credits

Elite

- ENG 191 – Introduction to the Microsoft SDL
- ENG 192 – Implementing the Agile MS SDL
- ENG 193 – Implementing the MS SDL Optimization Model
- ENG 194 – Implementing MS SDL Line of Business
- ENG 195 – Implementing the MS SDL Threat Modeling Tool
- COD 216 – Leveraging .NET Framework Code Access Security (CAS)
- COD 217 – Mitigating .NET Security Threats
- COD 242 – Creating Secure SQL Server and Azure SQL Database Applications
- COD 254 – Creating Secure Azure Applications

Elective

- DES 101 – Fundamentals of Secure Architecture
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field

- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 317 – Elite Secure Developer – IoT & Embedded

The Elite Secure Developer – IoT & Embedded Learning Path includes a variety of security courses designed to provide those responsible for designing and implementing software of embedded devices and systems with the knowledge and skills required to create secure embedded software and devices. The curriculum provides learners with a thorough grounding in application security concepts across the fundamental courses with special attention to coding within embedded systems and includes secure mobile development.

Upon successful completion of this path, you will have the knowledge and skills to:

- Apply techniques to identify system security and performance requirements
- Develop appropriate security architecture
- Select the correct mitigations for IoT components
- Develop policies that ensure the secure operation of IoT systems

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

17 Courses

0 Labs

7 Hours

9 CPE Credits

Elite

- COD 110 – Fundamentals of Secure Mobile Development
- COD 160 – Fundamentals of Secure Embedded Software Development
- COD 201 – Secure C Encrypted Network Communications
- COD 202 – Secure C Run-Time Protection
- COD 206 – Creating Secure C++ Code
- COD 207 – Communication Security in C++
- COD 261 – Threats to Scripts
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- DES 261 – Securing Serverless Environments
- COD 301 – Secure C Buffer Overflow Mitigations
- COD 302 – Secure C Memory Management
- COD 303 – Common C Vulnerabilities and Attacks
- COD 307 – Protecting Data in C++
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- ICS 310 – Protecting Information and System Integrity in Industrial Control System Environments

Elective

- DES 101 – Fundamentals of Secure Architecture
- DES 281 – OWASP IoT1: Mitigating Improper Platform Usage
- DES 282 – OWASP IoT2: Mitigating Insecure Data Storage
- DES 283 – OWASP IoT3: Mitigating Insecure Communication
- DES 284 – OWASP IoT4: Mitigating Insecure Authentication
- DES 285 – OWASP IoT5: Mitigating Insufficient Cryptography
- DES 286 – OWASP IoT6: Mitigating Insecure Authorization
- DES 287 – OWASP IoT7: Mitigating Client Code Quality
- DES 288 – OWASP IoT8: Mitigating Code Tampering
- DES 289 – OWASP IoT9: Mitigating Reverse Engineering
- DES 290 – OWASP IoT10: Mitigating Extraneous Functionality
- COD 366 – Creating Secure Kotlin Applications
- ENG 311 – Attack Surface Analysis & Reduction
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 318 – Elite Secure Developer – PCI

The Elite Secure Developer – PCI Learning Path includes a variety of security courses designed for those responsible for developing applications that process credit and debit card payments and/or any type of cardholder data. The curriculum provides learners with the tools required to meet the Payment Card Industry Data Security Standards (PCI DSS) for systems that transmit, process, and/or store cardholder data.

Upon successful completion of this path, you will have the knowledge and skills to:

- Develop secure applications
- Conduct effective test procedures
- Adopt guidance for mitigating issues

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

11 Courses

0 Labs

4 Hours

5 CPE Credits

Elite

- COD 141 – Fundamentals of Database Security
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- COD 241 – Creating Secure Oracle Database Applications
- COD 246 – PCI DSS 3: Protecting Stored Cardholder Data
- COD 247 – PCI DSS 4: Encrypting Transmission of Cardholder Data
- COD 248 – PCI DSS 6: Develop & Maintain Secure Systems and Applications
- COD 249 – PCI DSS 11: Regularly Test Security Systems and Processes
- COD 251 – Defending AJAX-enabled Web Applications
- DES 207 – Mitigating OWASP API Security Top 10
- DES 209 – Authentication and Lifecycle Management
- DES 312 – Protecting Cardholder Data

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 252 – Securing Google Platform Applications & Data
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DES 281 – OWASP IoT1: Mitigating Improper Platform Usage
- DES 282 – OWASP IoT2: Mitigating Insecure Data Storage

- DES 283 – OWASP IoT3: Mitigating Insecure Communication
- DES 284 – OWASP IoT4: Mitigating Insecure Authentication
- DES 285 – OWASP IoT5: Mitigating Insufficient Cryptography
- DES 286 – OWASP IoT6: Mitigating Insecure Authorization
- DES 287 – OWASP IoT7: Mitigating Client Code Quality
- DES 288 – OWASP IoT8: Mitigating Code Tampering
- DES 289 – OWASP IoT9: Mitigating Reverse Engineering
- DES 290 – OWASP IoT10: Mitigating Extraneous Functionality
- DSO 256 – DevSecOps in the Google Cloud Platform
- ENG 311 – Attack Surface Analysis & Reduction
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 319 – Elite Secure Developer – C

The Elite Secure Developer –C Learning Path includes a variety of security courses designed to provide a solid understanding of security features required to develop secure code that integrates into operating systems, operating system modules, embedded systems, or low-level libraries for other high-level languages. The curriculum covers key application security concepts including memory management and string handling.

Upon successful completion of this path, you will have the knowledge and skills to:

- Implement secure communications for C programming
- Protect applications from attack using run-time protection
- Mitigate C specific security flaws

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

7
Courses

0
Labs

3
Hours

3
CPE Credits

Elite

- COD 201 – Secure C Encrypted Network Communications
- COD 202 – Secure C Run-Time Protection
- COD 261 – Threats to Scripts
- DES 207 – Mitigating OWASP API Security Top 10
- COD 301 – Secure C Buffer Overflow Mitigations
- COD 302 – Secure C Memory Management
- COD 303 – Common C Vulnerabilities and Attack

Elective

- DES 101 – Fundamentals of Secure Architecture
- SDT 319 – Testing for Out-of-bounds Read
- SDT 320 – Testing for Out-of-bounds Write
- SDT 324 – Testing for Improper Restriction of Operations within the Bounds of a Memory Buffer
- SDT 325 – Testing for NULL Pointer Dereference
- SDT 326 – Testing for Use After Free
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering

- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 320 – Elite Secure Developer – Swift

The Elite Secure Developer – Swift Learning Path includes a variety of security courses designed for those responsible for the development of applications aimed towards iOS and OS X and the integration with back-end services.

Upon successful completion of this path, you will have the knowledge and skills to:

- How identify common mobile application risks
- Utilize best practices for designing and building applications for iOS and OS X, RESTful API's, embedded databases, and object-oriented programming

NOTE: This Learning Paths is considered principal to all Elite Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8
Courses

0
Labs

4
Hours

4
CPE Credits

Elite

- COD 110 – Fundamentals of Secure Mobile Development
- ENG 112 – Essential Access Control for Mobile Devices
- COD 261 – Threats to Scripts
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- COD 315 – Preventing Vulnerabilities in iOS Code in Swift
- COD 317 – Protecting Data on iOS in Swift

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 286 – Creating Secure React User Interfaces
- DES 207 – Mitigating OWASP API Security Top 10
- DES 219 – Securing Google's Firebase Platform
- DES 271 – OWASP M1: Mitigating Improper Platform Usage
- DES 272 – OWASP M2: Mitigating Insecure Data Storage
- DES 273 – OWASP M3: Mitigating Insecure Communication
- DES 274 – OWASP M4: Mitigating Insecure Authentication
- DES 275 – OWASP M5: Mitigating Insufficient Cryptography
- DES 276 – OWASP M6: Mitigating Insecure Authorization
- DES 277 – OWASP M7: Mitigating Client Code Quality
- DES 278 – OWASP M8: Mitigating Code Tampering
- DES 279 – OWASP M9: Mitigating Reverse Engineering
- DES 280 – OWASP M10: Mitigating Extraneous Functionality

- DES 284 – OWASP IoT4: Mitigating Lack of Secure Update Mechanism
- DES 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 288 – OWASP IoT8: Mitigating Lack of Device Management
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- COD 366 – Creating Secure Kotlin Applications
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 305 – Testing for Broken Access Control
- SDT 316 – Testing for Use of Hard-coded Credentials
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 321 – Elite Secure Developer – Android

The Elite Secure Developer – Android Learning Path includes a variety of security courses designed to provide a solid foundation of security features necessary to develop applications for devices powered by the Android operating system. The curriculum provides secure coding best practices for designing and building android applications.

Upon successful completion of this path, you will have the knowledge and skills to:

- Identify common android application risks
- Create a mobile application threat model
- Apply android platform specific knowledge

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8 Courses

0 Labs

4 Hours

5 CPE Credits

Elite

- COD 110 – Fundamentals of Secure Mobile Development
- ENG 112 – Essential Access Control for Mobile Devices
- COD 261 – Threats to Scripts
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- COD 318 – Protecting Data on Android in Java
- COD 319 – Preventing Vulnerabilities in Android Code in Java

Elective

- DES 101 – Fundamentals of Secure Architecture
- COD 286 – Creating Secure React User Interfaces
- DES 207 – Mitigating OWASP API Security Top 10
- DES 219 – Securing Google's Firebase Platform
- DES 271 – OWASP M1: Mitigating Improper Platform Usage
- DES 272 – OWASP M2: Mitigating Insecure Data Storage
- DES 273 – OWASP M3: Mitigating Insecure Communication
- DES 274 – OWASP M4: Mitigating Insecure Authentication
- DES 275 – OWASP M5: Mitigating Insufficient Cryptography
- DES 276 – OWASP M6: Mitigating Insecure Authorization
- DES 277 – OWASP M7: Mitigating Client Code Quality
- DES 278 – OWASP M8: Mitigating Code Tampering
- DES 279 – OWASP M9: Mitigating Reverse Engineering

- DES 280 – OWASP M10: Mitigating Extraneous Functionality
- DES 284 – OWASP IoT4: Mitigating Lack of Secure Update Mechanism
- DES 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 288 – OWASP IoT8: Mitigating Lack of Device Management
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- COD 366 – Creating Secure Kotlin Applications
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 305 – Testing for Broken Access Control
- SDT 316 – Testing for Use of Hard-coded Credentials
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 322 – Elite Secure Developer – .NET

The Elite Secure Developer – .NET Learning Path includes a variety of security courses that will provide a solid foundation of .NET security features for building secure web applications, sophisticated desktop applications, or modern mobile applications. The curriculum offers application framework specific secure coding best practices for ASP.NET to extend the .NET Developer platform with tools and libraries for building web applications.

Upon successful completion of this path, you will have the knowledge and skills to:

- Leverage Code Access Security (CAS) functions in .NET
- Understand Level 2 Security Transparency Model
- Avoid dangerous patterns and common .NET security pitfalls

NOTE: This Learning Paths is considered tertiary to Core and Advanced Secure Developer Learning Paths. Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

8 Courses

16 Labs

7 Hours

9 CPE Credits

Elite

- COD 216 – Leveraging .NET Framework Code Access Security (CAS)
- COD 217 – Mitigating .NET Security Threats
- COD 255 – Creating Secure Code - Web API Foundations
- COD 308 – Common ASP.NET Vulnerabilities and Attacks
- COD 309 – Securing ASP.NET MVC Applications
- COD 321 – Protecting C# from Integer Overflows & Canonicalization
- COD 322 – Protecting C# from SQL Injection
- COD 323 – Using Encryption with C#
- LAB 221 – Defending C# Applications Against SQL Injection
- LAB 232 – Defending C# Applications Against XSS
- LAB 241 – Defending C# Applications Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 238 – Defending C# Applications Against Weak AES ECB Mode Encryption
- LAB 239 – Defending C# Applications Against Weak PRNG
- LAB 250 – Defending C# Applications Against Parameter Tampering
- LAB 251 – Defending C# Applications Against Plaintext Password Storage
- LAB 260 – Defending C# Applications Against Sensitive Information in Error Messages
- LAB 266 – Defending C# Applications Against Sensitive Information in Log Files
- LAB 270 – Defending C# Applications Against Deserialization of Untrusted Data
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 278 – Defending C# Applications Against Command Injection
- LAB 282 – Defending C# Applications Against Dangerous File Upload
- LAB 286 – Defending C# Applications Against RegEx DoS

Elective

- DES 101 – Fundamentals of Secure Architecture
- DES 207 – Mitigating OWASP API Security Top 10
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page

CSC 325 – Secure Network Engineer

The Secure Network Engineer Learning Path credential is designed to provide the security skills and knowledge required to protect infrastructure and the sensitive data it handles. The curriculum provides participants with the ability to demonstrate the competency to apply best practices for managing systems and services across all environments.

After completing this learning path learners will have the knowledge and skill to:

- Apply best practices for managing systems and services across all environments
- Improve the stability, security, efficiency, and scalability of environments
- Understand how to create and modify scripts to perform tasks

NOTE: Learn and Skill labs are elective training modules that help transform concepts into tangible skills through hands-on, realistic examples of real-world threat scenarios.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

14
Courses

14
Labs

7
Hours

9
CPE Credits

Core

- DES 101 – Fundamentals of Secure Architecture
- ENG 110 – Essential Account Management Security
- ENG 114 – Essential Risk Assessment
- ENG 115 – Essential System and Information Integrity
- ENG 119 – Essential Security Audit and Accountability
- ENG 121 – Essential Identification and Authentication

Advanced

- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- DES 210 – Hardening Linux
- DES 214 – Securing Infrastructure Architecture

Elite

- CYB 310 – Using Cyber Supply Chain Risk Management (C-SCRM) to Mitigate Threats to IT/OT
- CYB 311 – Threat Analysis with AI
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with External Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery

- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK Data from Local System
- LAB 337 – ATT&CK Valid Accounts
- LAB 310 – ATT&CK: File and Directory Permissions Modification
- LAB 311 – ATT&CK: File and Directory Discovery

Elective

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 110 – Fundamentals of Secure Mobile Development
- TST 101 – Fundamentals of Security Testing
- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- API 250 – Controlling Access to the Kubernetes API
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- CYB 212 – Fundamentals of Security Information & Event Management (SIEM)
- ICS 210 – ICS/SCADA Security Essentials
- COD 263 – Secure Bash Scripting
- COD 264 – Secure Perl Scripting
- COD 265 – Secure Python Scripting
- COD 266 – Secure Ruby Scripting
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 209 – Authentication and Lifecycle Management
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 218 – Protecting Microservices, Containers and Orchestration
- DES 260 – Fundamentals of IoT Architecture & Design
- DES 261 – Securing Serverless Environments
- DSO 211 – Identifying Threats to Containers & Data in DevSecOps Framework
- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 302 – Automated Security Testing
- DSO 303 – Automating Security Updates
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 305 – Automating CI/CD Pipeline Compliance
- ENG 351 – Preparing the Risk Management Framework
- ENG 352 – Categorizing Systems
- ENG 353 – Selecting, Implementing
- ENG 354 – Authorizing and Monitoring System Controls

CSC 323 – Secure DevOps Practitioner

The Secure DevOps Practitioner Learning Path includes a variety of security courses designed for those who work closely with Software Engineers to help them deploy and operate various systems. The curriculum provides teams with a solid foundation of security features necessary to automate and streamline operations and processes while keeping security top of mind. Learners will apply best practices to develop new features and write scripts across various technologies.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

54 Courses

17 Labs

20 Hours

24 CPE Credits

Core

- COD 102 – Challenges in Application Security
- COD 103 – Creating Software Security Requirements
- COD 104 – Designing Secure Software
- COD 105 – Secure Software Development
- COD 106 – The Importance of Software Integration and Testing
- COD 107 – Secure Software Deployment
- COD 108 – Software Operations and Maintenance
- DES 101 – Fundamentals of Secure Architecture
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- ENG 123 – Essential Security Engineering Principles
- ENG 124 – Essential Application Protection
- ENG 125 – Essential Data Protection
- TST 101 – Fundamentals of Security Testing

Advanced

- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- API 250 – Controlling Access to the Kubernetes API
- COD 252 – Securing Google Platform Applications & Data
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 209 – Authentication and Lifecycle Management
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 218 – Protecting Microservices, Containers, and Orchestration

- DES 235 – Mitigating OWASP 2021 Insecure Design
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 261 – Securing Serverless Environments (NEW)
- DSO 201 – Fundamentals of Secure DevOps
- DSO 211 – Identifying Threats to Containers in a DevSecOps Framework
- DSO 212 – Fundamentals of Zero Trust Security
- DSO 253 – DevSecOps in the AWS Cloud
- DSO 254 – DevSecOps in the Azure Cloud
- DSO 256 – DevSecOps in the Google Cloud Platform
- ENG 205 – Fundamentals of Threat Modeling
- ENG 251 – Risk Management Foundations
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans
- TST 206 – ASVS Requirements for Developers
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective XSS
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection

Elite

- CYB 310 - Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- CYB 311 - Threat Analysis with AI
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 302 – Automated Security Testing
- DSO 303 – Automating Security Updates
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 305 – Automating CI/CD Pipeline Compliance
- DSO 306 – Implementing Infrastructure as Code
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- ENG 351 – Preparing the Risk Management Framework

CSC 324 – Ethical Hacker

The Ethical Hacker Learning Path includes a variety of security courses geared towards individuals responsible for assessing systems and networks within the network environment and identifying where those systems/networks deviate from acceptable configurations, enclave policy, or local policy. The curriculum provides a solid foundation of the skills needed to measure the effectiveness of defense-in-depth architecture against known vulnerabilities and verify and improve the security of a company's computer systems.

After completing this learning path learners will have the knowledge and skills necessary to:

- Analyze cyber defense policies and configurations
- Evaluate compliance with regulations and organizational directives
- Conduct and/or support authorized penetration testing on enterprise network assets
- Deploy cyber defense audit toolkit to support cyber defense audit missions
- Maintain knowledge of applicable cyber defense policies, regulations, and compliance documents specifically related to cyber defense auditing
- Prepare audit reports that identify technical and procedural findings and provide recommended remediation strategies/solutions
- Conduct required reviews as appropriate within an environment
- Perform evaluation of technology and of people and operations risk
- Perform vulnerability assessments of relevant technology focus areas
- Make recommendations regarding the selection of cost-effective security controls to mitigate risk

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

44 Courses

52 Labs

21 Hours

25 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 141 – Fundamentals of Database Security
- DES 101 – Fundamentals of Secure Architecture
- ENG 110 – Essential Account Management Security
- ENG 114 – Essential Risk Assessment
- ENG 118 – ESSENTIAL Incident Response
- ENG 120 – Essential Security Assessment & Authorization
- ENG 123 – Essential Security Engineering Principles
- TST 101 – Fundamentals of Security Testing
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures

- DES 235 – Mitigating OWASP 2021 Insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Mitigating Insecure Deserialization
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 113 – Identifying Cryptographic Failures
- LAB 115 – Identifying Reflective XSS
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection
- LAB 121 – Identifying Vulnerable and Outdated Components
- LAB 127 – Identifying Security Logging and Monitoring Failures
- LAB 129 – Identifying Error Message Containing Sensitive Information
- LAB 133 – Identifying Exposure of Sensitive Information Through Environmental Variables

Advanced

- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 203 – Cryptographic Components: Randomness, Algorithms, and Key Management
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 210 – Hardening Linux/Unix Systems
- DES 212 – Architecture Risk Analysis & Remediation
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 217 – Securing Terraform Infrastructure Resources
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DES 272 – OWASP M2: Mitigating Insecure Data Storage
- DES 282 – OWASP IoT2: Mitigating Insecure Network Services
- DES 288 – OWASP IoT8: Mitigating Lack of Device Management
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- DSO 205 – Securing the COTS Supply Chain
- DSO 206 – Securing the Open-Source Supply Chain
- DSO 211 – Identifying Threats to Containers in a DevSecOps Framework
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 251 – Risk Management Foundations
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities

- LAB 111 – Identifying Server-Side Request Forgery
- LAB 114 – Identifying Cookie Tampering
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 128 – Identifying Unverified Password Change
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services
- LAB 134 – Identifying Plaintext Storage of a Password
- LAB 135 – Identifying URL Redirection to Untrusted Site
- LAB 136 – Identifying Improper Neutralization of Script in Attributes in a Web Page
- LAB 137 – Improper Authorization
- LAB 138 – Authorization Bypass Through User-Controlled Key
- LAB 139 – Use of a Key Past its Expiration Date

Elite

- CYB 301 – Fundamentals of Ethical Hacking
- ENG 311 – Attack Surface Analysis & Reduction
- LAB 312 – ATT&CK: Network Service Identification
- LAB 313 – ATT&CK: Vulnerability Identification Using Vulnerability Databases
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK Exploiting Windows File Sharing Server with Eternal Romance Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK: Data from Local System
- LAB 337 – ATT&CK: Valid Accounts

Elective

- DES 305 – Protecting Existing Blockchain Assets
- DES 306 – Creating a Secure Blockchain Network
- DSO 303 – Automating Security Updates
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- ENG 351 – Preparing the Risk Management Framework
- ENG 352 – Categorizing Systems and Information within the RMF
- ENG 353 – Selecting, Implementing, and Assessing Controls within the Risk Management Framework
- ENG 354 – Authorizing and Monitoring System Controls within the RMF

CSC 326 – Secure Automation Engineer

The Secure Automation Engineer Learning Path includes a variety of security courses designed for those who design, program, simulate and test automated machinery and processes to complete exact tasks. The curriculum covers essential goals and controls needed to create secure software, managing risk in the software development lifecycle, cryptography, handling input and output, and the OWASP Top Ten.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

32 Courses

0 Labs

8 Hours

9 CPE Credits

Core

- ENG 110 – Essential Account Management Security
- ENG 113 – Essential Secure Configuration Management
- ENG 114 – Essential Risk Assessment
- ENG 119 – Essential Security Audit & Accountability
- ENG 120 – Essential Security Assessment & Authorization
- ENG 123 – Essential Security Engineering Principles
- ENG 124 – Essential Application Protection
- ENG 125 – Essential Data Protection

Advanced

- DES 209 – Authentication and Lifecycle Management
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- DSO 211– Identifying Threats to Containers and Data in a DevSecOps Framework
- ENG 251 – Risk Management Foundations
- ICS 210 – ICS/SCADA Security Essentials

Elite

- DSO 302 – Automated Security Testing
- DSO 303 – Automating Security Updates
- DSO 306 – Implementing Infrastructure as Code
- ENG 351 – Preparing the Risk Management Framework

- ICS 310 – Protecting Information and System Integrity in Industrial Control System Environments
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification Failures
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- SDT 315 – Testing for Incorrect Permission Assignment for Critical Resource
- SDT 316 – Testing for Use of Hard-Coded Credentials

CSC 327 – Embedded Test Engineer

The Embedded Test Engineer Learning Path includes a variety of security courses designed for those responsible for verifying and assuring the application security of embedded systems. The curriculum provides learners with a solid understanding of applied testing techniques and a well-rounded base of knowledge to perform their tasks. Learners will explore security best practices for conducting penetration tests and vulnerability assessment activities on embedded systems.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

57 Courses

0 Labs

16 Hours

18 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 101 – Fundamentals of Secure Architecture
- ENG 114 – Essential Risk Assessment
- ENG 123 – Essential Security Engineering Principles
- TST 101 – Fundamentals of Security Testing

Advanced

- ATK 201 – Using the MiTRE ATT&CK Framework
- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 212 – Architecture Risk Analysis and Remediation
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ICS 210 – ICS/SCADA Security Essentials
- TST 202 – Penetration Testing Fundamentals

Elite

- ICS 310 – Protecting Information and System Integrity in Industrial Control System Environments
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 304 – Testing for Insecure Design
- SDT 303 – Testing for Cryptographic Failures
- SDT 305 – Testing for Broken Access Control
- SDT 306 – Testing for Security Misconfiguration
- SDT 307 – Testing for Server-Side Request Forgery (SSRF)
- SDT 308 – Testing for Software and Data Integrity Failures
- SDT 309 – Testing for Vulnerable and Outdated Components

- SDT 310 – Testing for Security Logging and Monitoring Failures
- CYB 301 – Fundamentals of Ethical Hacking
- DSO 302 – Automated Security Testing
- ENG 312 – How to Perform a Security Code Review
- SDT 311 – Testing for Integer Overflow or Wraparound
- SDT 312 – Testing for (Path Traversal) Improper Limitation of a Pathname to a Restricted Directory
- SDT 313 – Testing for (CSRF) Cross Site Request Forgery
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- SDT 315 – Testing for Incorrect Permission Assignment for Critical Resource
- SDT 316 – Testing for Use of Hard-Coded Credentials
- SDT 317 – Testing for Improper Control of Generation of Code
- SDT 318 – Testing for Insufficiently Protected Credentials
- SDT 319 – Testing for Out-of-bounds Read
- SDT 320 – Testing for Out-of-bounds Write
- SDT 321 – Testing for Uncontrolled Resource Consumption
- SDT 322 – Testing for Improper Privilege Management
- SDT 323 – Testing for Improper Input Validation
- SDT 324 – Testing for Improper Restriction of Operations within the Bounds of a Memory Buffer
- SDT 325 – Testing for NULL Pointer Dereference
- SDT 326 – Testing for Use After Free
- TST 301 – Infrastructure Penetration Testing
- TST 302 – Application Penetration Testing
- TST 351 – Penetration Testing for TLS Vulnerabilities
- TST 352 – Penetration Testing for Injection Vulnerabilities
- TST 353 – Penetration Testing for SQL Injection
- TST 354 – Penetration Testing for Memory Corruption Vulnerabilities
- TST 355 – Penetration Testing for Authorization Vulnerabilities
- TST 356 – Penetration Testing for Cross-Site Scripting (XSS)
- TST 357 – Penetration Testing for Hardcoded Secrets
- TST 358 – Penetration Testing Wireless Networks
- TST 359 – Penetration Testing Network Infrastructure
- TST 360 – Penetration Testing for Authentication Vulnerabilities

CSC 328 – QA Test Engineer

The Secure Learning Path includes a variety of security courses designed for those responsible for assessing and testing the quality of specifications and technical design. The curriculum provides learners with understanding of how to perform hands-on testing for the most common software vulnerabilities. Learners will gain the knowledge and skill to review and understand systems requirements and design, review test strategy and design, and identify mitigations for defects identified during testing.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

80 Courses

46 Labs

28 Hours

34 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 101 – Fundamentals of Secure Architecture
- ENG 114 – Essential Risk Assessment
- ENG 123 – Essential Security Engineering Principles
- TST 101 – Fundamentals of Security Testing

Advanced

- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- ATK 201 – Using the MiTRE ATT&CK Framework
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 202 – Cryptographic Suite Services: Encoding, Encrypting & Hashing
- DES 203 – Cryptographic Components: Randomness, Algorithms, and Key Management
- DES 204 – Role of Cryptography in Application Development
- DES 205 – Message Integrity Cryptographic Functions
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 209 – Authentication and Lifecycle Management
- DES 212 – Architecture Risk Analysis and Remediation
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures

- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 111 – Identifying Server-Side Request Forgery
- LAB 113 – Identifying Cryptographic Failures
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective Cross-Site Scripting (XSS)
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent Cross-Site Scripting (XSS)
- LAB 120 – Identifying XML Injection
- LAB 121 – Identifying Vulnerable and Outdated Components
- LAB 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 124 – Identifying Horizontal Privilege Escalation
- LAB 125 – Identifying Buffer Overflow
- LAB 126 – Identifying Information Leakage
- LAB 127 – Identifying Security Logging and Monitoring Failures
- LAB 128 – Identifying Unverified Password Change
- LAB 129 – Identifying Error Message Containing Sensitive Information
- LAB 130 – Identifying Generation of Predictable Numbers or Identifiers

Elite

- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with EternalRomance Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration

- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK: Data from Local System
- LAB 337 – ATT&CK: Valid Accounts
- SDT 301 – Testing for Injection
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 303 – Testing for Cryptographic Failures
- SDT 304 – Testing for Insecure Design
- SDT 305 – Testing for Broken Access Control
- SDT 306 – Testing for Security Misconfiguration
- SDT 307 – Testing for Server-Side Request Forgery (SSRF)
- SDT 308 – Testing for Software and Data Integrity Failures
- SDT 309 – Testing for Vulnerable and Outdated Components
- SDT 310 – Testing for Security Logging and Monitoring Failures
- CYB 301 – Fundamentals of Ethical Hacking
- SDT 311 – Testing for Integer Overflow or Wraparound
- SDT 312 – Testing for (Path Traversal) Improper Limitation of a Pathname to a Restricted Directory
- SDT 313 – Testing for (CSRF) Cross Site Request Forgery
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- SDT 315 – Testing for Incorrect Permission Assignment for Critical Resource
- SDT 316 – Testing for Use of Hard-Coded Credentials
- SDT 317 – Testing for Improper Control of Generation of Code
- SDT 318 – Testing for Insufficiently Protected Credentials
- SDT 319 – Testing for Out-of-bounds Read
- SDT 320 – Testing for Out-of-bounds Write
- SDT 321 – Testing for Uncontrolled Resource Consumption
- SDT 322 – Testing for Improper Privilege Management
- SDT 323 – Testing for Improper Input Validation
- SDT 324 – Testing for Improper Restriction of Operations within the Bounds of a Memory Buffer
- SDT 325 – Testing for NULL Pointer Dereference
- SDT 326 – Testing for Use After Free
- DES 311 – Creating Secure Application Architecture
- DSO 302 – Automated Security Testing
- ENG 312 – How to Perform a Security Code Review
- TST 351 – Penetration Testing for TLS Vulnerabilities
- TST 352 – Penetration Testing for Injection Vulnerabilities
- TST 353 – Penetration Testing for SQL Injection
- TST 354 – Penetration Testing for Memory Corruption Vulnerabilities
- TST 355 – Penetration Testing for Authorization Vulnerabilities
- TST 356 – Penetration Testing for Cross-Site Scripting (XSS)
- TST 357 – Penetration Testing for Hardcoded Secrets
- TST 358 – Penetration Testing Wireless Networks
- TST 359 – Penetration Testing Network Infrastructure
- TST 360 – Penetration Testing for Authentication Vulnerabilities

CSC 329 – Secure IT Architect

The Secure IT Architect Learning Path includes a variety of security courses designed for those responsible for designing and maintaining computer networks. The curriculum covers best practices for secure software design, creating integrated architecture across business and technology, and protecting data and resources from disclosure, modification, and deletion.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

41 Courses

6 Labs

17 Hours

20 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 101 – Fundamentals of Secure Architecture

Advanced

- API 250 – Controlling Access to the Kubernetes API
- COD 252 – Securing Google Platform Applications & Data
- DES 202 – Cryptographic Suite Services: Encoding, Encrypting & Hashing
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 207 – Mitigating OWASP API Security Top 10
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 209 – Authentication and Lifecycle Management
- DES 210 – Hardening Linux/Unix Systems
- DES 212 – Architecture Risk Analysis and Remediation
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 217 – Securing Terraform Infrastructure and Resources
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- DES 261 – Securing Serverless Environments
- DSO 211 – Identifying Threats to Containers and Data in a DevSecOps Framework
- DSO 212 – Fundamentals of Zero Trust Security
- DSO 256 – DevSecOps in the Google Cloud Platform
- ENG 211 – How to Create Application Security Design Requirements
- ENG 251 – Risk Management Foundations

Elite

- CYB 310 – Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- CYB 311 – Threat Analysis with AI
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 305 – Automating CI/CD Pipeline Compliance
- DSO 306 – Implementing Infrastructure as Code
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 351 – Preparing the Risk Management Framework
- ENG 352 – Categorizing Systems and Information within the RMF
- ENG 353 – Selecting, Implementing and Assessing Controls within the RMF
- ENG 354 – Authorizing and Monitoring System Controls within the RMF
- LAB 312 - ATT&CK: Network Service Identification
- LAB 313 - ATT&CK: Vulnerability Identification Using Vulnerability Databases
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server with Eternal Romance Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud

CSC 330 – Secure Embedded Architect

The Secure Embedded Architect Learning Path includes a variety of security courses designed for those responsible for designing and implementing software of embedded devices and systems and provides insight into the unique resource requirements of embedded environments and best practices for designing secure software for them.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

11 Courses

0 Labs

5 Hours

6 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- DES 101 – Fundamentals of Secure Architecture

Advanced

- DES 202 – Cryptographic Suite Services: Encoding, Encrypting & Hashing
- DES 212 – Architecture Risk Analysis and Remediation
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- ICS 210 – ICS/SCADA Security Essentials

Elite

- DES 311 – Creating Secure Application Architecture
- ENG 311 – Attack Surface Analysis & Reduction
- ENG 312 – How to Perform a Security Code Review
- ICS 310 – Protecting Information and System Integrity in Industrial Control System Environments

CSC 331 – Secure Software Architect

The Secure Software Architect Learning Path includes a variety of security courses designed for those making design choices, coordinating, and overseeing technical standards and includes software coding standards, tools, and platforms. The curriculum provides learners with the ability to apply secure software architecture best practices to early phase SDLC activities. Learners will gain the knowledge and skill to implement defensive coding techniques and avoid systemic issues found in insecure software.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

73 Courses

29 Labs

29 Hours

35 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 102 – Challenges in Application Security
- COD 103 – Creating Software Security Requirements
- COD 104 – Designing Secure Software
- COD 105 – Secure Software Development
- COD 106 – The Importance of Software Integration and Testing
- COD 107 – Secure Software Deployment
- COD 108 – Software Operations and Maintenance
- DES 101 – Fundamentals of Secure Architecture
- COD 141 Fundamentals of Database Security
- DES 101 – Fundamentals of Secure Architecture
- DES 151 – Fundamentals of the PCI Secure SLC Standard

Advanced

- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- COD 252 – Securing Google Platform Applications & Data
- COD 261 – Threats to Scripts
- COD 267 – Securing Python Microservices
- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 202 – Cryptographic Suite Services: Encoding, Encrypting & Hashing
- DES 203 – Cryptographic Components: Randomness, Algorithms, and Key Management
- DES 204 – Role of Cryptography in Application Development
- DES 205 – Message Integrity Cryptographic Functions
- DES 207 – Mitigating OWASP API Security Top 10
- DES 209 – Authentication and Lifecycle Management

- DES 212 – Architecture Risk Analysis and Remediation
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 217 – Securing Terraform Infrastructure and Resources
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture and Design
- DES 281 – OWASP IoT1: Mitigating Weak, Guessable or Hardcoded Passwords
- DES 282 – OWASP IoT2: Mitigating Insecure Network Services
- DES 283 – OWASP IoT3: Mitigating Insecure Ecosystem Interfaces
- DES 284 – OWASP IoT4: Mitigating Lack of Secure Update Mechanism
- DES 285 – OWASP IoT5: Mitigating Use of Insecure or Outdated Components
- DES 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 288 – OWASP IoT8: Mitigating Lack of Device Management
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- DES 290 – OWASP IoT10 Mitigating Lack of Physical Hardening
- DSO 201 – Fundamentals of Secure DevOps
- DSO 211 – Identifying Threats to Containers and Data in a DevSecOps Framework
- DSO 212 – Fundamentals of Zero Trust Security
- DSO 256 – DevSecOps in the Google Cloud Platform
- ENG 211 – How to Create Application Security Design Requirements
- ENG 251 – Risk Management Foundations
- TST 206 – ASVS Requirements for Developers
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective XSS
- LAB 116 – Identifying Forceful Browsing

- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection
- LAB 220 – Defending Against Hard-Coded Secrets

Elite

- COD 287 - Java Application Server Hardening
- CYB 310 -Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- ENG 320 - Using Software Composition Analysis (SCA) to Secure Open-Source Components
- COD 383 - Protecting Java Backend Services
- DES 311 – Creating Secure Application Architecture
- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 302 – Automated Security Testing
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 305 – Automating CI/CD Pipeline Compliance
- ENG 311 – Attack Surface Analysis & Reduction
- ENG 312 – How to Perform a Security Code Review
- ENG 351 – Preparing the Risk Management Framework
- ENG 352 – Categorizing Systems and Information within the RMF
- ENG 353 – Selecting, Implementing and Assessing Controls within the RMF
- ENG 354 – Authorizing and Monitoring System Controls within the RMF
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- SDT 315 – Testing for Incorrect Permission Assignment for Critical Resource
- SDT 316 – Testing for Use of Hard-Coded Credentials
- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK: Data from Local System
- LAB 337 – ATT&CK: Valid Accounts

Elective

- LAB 221 – Defending C# Applications Against SQL Injection (NEW)
- LAB 222 – Defending Python Applications Against SQL Injection (NEW)
- LAB 223 – Defending Node.js Applications Against SQL Injection (NEW)
- LAB 228 – Defending Java Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 229 – Defending Java Applications Against Weak PRNG (NEW)
- LAB 230 – Defending Java Applications Against XSS (NEW)

- LAB 231 – Defending Python Applications Against XSS (NEW)
- LAB 232 – Defending C# Applications Against XSS (NEW) (XSS)
- LAB 233 – Defending Node.js Applications Against XSS (NEW)
- LAB 234 – Defending Java Applications Against Parameter Tampering (NEW)
- LAB 235 – Defending Java Applications Against Plaintext Password Storage (NEW)
- LAB 236 – Defending Java Applications Against Sensitive Information in Error Messages
- LAB 237 – Defending Java Applications Against SQL Injection (NEW)
- LAB 238 – Defending C# Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 239 – Defending C# Applications Against Weak PRNG (NEW)
- LAB 240 – Defending Java Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 241 – Defending C# Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 242 – Defending Node.js Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 243 – Defending Python Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 244 – Defending Java Applications Against Security Misconfiguration (NEW)
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage (NEW)
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 247 – Defending Node.js Applications Against Weak PRNG (NEW)
- LAB 248 – Defending Node.js Applications Against Parameter Tampering (NEW)
- LAB 249 – Defending Python Applications Against Plaintext Password Storage (NEW)
- LAB 250 – Defending C# Applications Against Parameter Tampering (NEW)
- LAB 251 – Defending C# Applications Against Plaintext Password Storage (NEW)
- LAB 252 – Defending Python Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 253 – Defending Python Applications Against Weak PRNG (NEW)
- LAB 254 – Defending Python Applications Against Parameter Tampering (NEW)
- LAB 260 – Defending C# Applications Against Sensitive Information in Error Messages
- LAB 261 – Defending Python Applications Against Sensitive Information in Error Messages
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages
- LAB 263 – Defending Java Applications Against Sensitive Information in Log Files
- LAB 264 – Defending Python Applications Against Sensitive Information in Log Files
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files
- LAB 266 – Defending C# Applications Against Sensitive Information in Log Files
- LAB 267 – Defending Java Applications Against Deserialization of Untrusted Data
- LAB 268 – Defending Python Applications Against Deserialization of Untrusted Data
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data Node.js
- LAB 270 – Defending C# Applications Against Deserialization of Untrusted Data
- LAB 271 – Defending Java Applications Against SSRF
- LAB 272 – Defending Python Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 274 – Defending C# Applications Against SSRF
- LAB 275 - Defending Java Applications Against Command Injection
- LAB 276 - Defending Python Applications Against Command Injection
- LAB 277 - Defending Node.js Applications Against Command Injection
- LAB 278 - Defending C# Applications Against Command Injection
- LAB 279 - Defending Java Applications Against Dangerous File Upload
- LAB 280 - Defending Python Applications Against Dangerous File Upload
- LAB 281 - Defending Node.js Against Dangerous File Upload
- LAB 282 - Defending C# Applications Against Dangerous File Upload
- LAB 283 - Defending Java Applications Against RegEx DoS

- LAB 284 - Defending Python Applications Against RegEx DoS
- LAB 285 - Defending Node.js Applications Against RegEx DoS
- LAB 286 - Defending C# Applications Against RegEx DoS

CSC 332 – Secure Business Analyst

The Secure Business Analyst Learning Path includes a variety of security courses designed for those responsible for defining, analyzing, and documenting requirements in the software development lifecycle.

After completing this learning path learners will have the knowledge and skill to:

- Adhere to system and information security policies
- Meet compliance mandates for relevant government and industry standards
- Understand access control, configuration management, risk assessment, auditing, and authentication

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

17 Courses

0 Labs

6 Hours

9 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 101 – Fundamentals of Secure Architecture
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- ENG 114 – Essential Risk Assessment
- ENG 116 – Essential Security Planning Policy & Procedures
- ENG 117 – Essential Information Security Program Planning

Advanced

- DSO 201 – Fundamentals of Secure DevOps
- ENG 211 – How to Create Application Security Design Requirements
- ENG 251 – Risk Management Foundations
- TST 202 – Penetration Testing Fundamentals
- TST 206 – ASVS Requirements for Developers

Elite

- DSO 302 – Automated Security Testing
- ENG 351 – Preparing the Risk Management Framework
- ENG 352 – Categorizing Systems and Information within the RMF
- ENG 353 – Selecting, Implementing and Assessing Controls within the RMF
- ENG 354 – Authorizing and Monitoring System Controls within the RMF

CSC 333 – Secure Systems Analyst

The Secure Systems Analyst Learning Path includes a variety of security courses designed for those who specialize in the implementation of computer system requirements. The curriculum provides the fundamental knowledge required to secure networks and systems including:

- Taking a holistic approach to network and system security
- Defining and analyzing system problems
- Designing and testing standards and solutions
- Controls, monitoring access, operational procedures, auditing, and logging

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

44 Courses

18 Labs

14 Hours

17 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- ENG 110 – Essential Account Management Security
- ENG 111 – Essential Session Management Security
- ENG 112 – Essential Access Control for Mobile Devices
- ENG 113 – Essential Secure Configuration Management
- ENG 114 – Essential Risk Assessment
- ENG 115 – Essential System & Information Integrity
- ENG 116 – Essential Security Planning Policy & Procedures
- ENG 117 – Essential Information Security Program Planning
- ENG 118 – Essential Incident Response
- ENG 119 – Essential Security Audit & Accountability
- ENG 120 – Essential Security Assessment & Authorization
- ENG 121 – Essential Identification & Authentication
- ENG 122 – Essential Physical & Environmental Protection
- ENG 123 – Essential Security Engineering Principles
- ENG 124 – Essential Application Protection
- ENG 125 – Essential Data Protection
- ENG 126 – Essential Security Maintenance Policies
- ENG 127 – Essential Media Protection

Advanced

- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management

- CYB 210 – Cybersecurity Incident Response
- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 210 – Hardening Linux/Unix Systems
- DES 217 – Securing Terraform Infrastructure and Resources
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- ENG 251 – Risk Management Foundations
- TST 206 – ASVS Requirements for Developers
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective XSS
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection
- LAB 220 – Defending Against Hard-Coded Secrets

Elite

- CYB 310 - Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- CYB 311 - Threat Analysis with AI
- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 302 – Automated Security Testing
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 305 – Automating CI/CD Pipeline Compliance
- ENG 320 - Using Software Composition Analysis (SCA) to Secure Open-Source Components
- ENG 351 – Preparing the Risk Management Framework

- ENG 352 – Categorizing Systems and Information within the RMF
- ENG 353 – Selecting, Implementing and Assessing Controls within the RMF
- ENG 354 – Authorizing and Monitoring System Controls within the RMF
- ICS 310 – Protecting Information and System Integrity in Industrial Control System Environments
- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud

Elective

- COD 288 - Java Public Key Cryptography
- COD 287 - Java Application Server Hardening
- COD 383 - Protecting Java Backend Services
- LAB 221 – Defending C# Applications Against SQL Injection (NEW)
- LAB 222 – Defending Python Applications Against SQL Injection (NEW)
- LAB 223 – Defending Node.js Applications Against SQL Injection (NEW)
- LAB 228 – Defending Java Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 229 – Defending Java Applications Against Weak PRNG (NEW)
- LAB 230 – Defending Java Applications Against XSS (NEW)
- LAB 231 – Defending Python Applications Against XSS (NEW)
- LAB 232 – Defending C# Applications Against XSS (NEW) (XSS)
- LAB 233 – Defending Node.js Applications Against XSS (NEW)
- LAB 234 – Defending Java Applications Against Parameter Tampering (NEW)
- LAB 235 – Defending Java Applications Against Plaintext Password Storage (NEW)
- LAB 236 – Defending Java Applications Against Sensitive Information in Error Messages
- LAB 237 – Defending Java Against from SQL Injection
- LAB 238 – Defending C# Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 239 – Defending C# Applications Against Weak PRNG (NEW)
- LAB 240 – Defending Java Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 241 – Defending C# Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 242 – Defending Node.js Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 243 – Defending Python Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 244 – Defending Java Applications Against Security Misconfiguration (NEW)
- LAB 245 – Defending Node.js Applications Against Plaintext Password Storage (NEW)
- LAB 246 – Defending Node.js Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 247 – Defending Node.js Applications Against Weak PRNG (NEW)
- LAB 248 – Defending Node.js Applications Against Parameter Tampering (NEW)
- LAB 249 – Defending Python Applications Against Plaintext Password Storage (NEW)
- LAB 250 – Defending C# Applications Against Parameter Tampering (NEW)
- LAB 251 – Defending C# Applications Against Plaintext Password Storage (NEW)
- LAB 252 – Defending Python Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 253 – Defending Python Applications Against Weak PRNG (NEW)
- LAB 254 – Defending Python Applications Against Parameter Tampering (NEW)
- LAB 260 – Defending C# Applications Against Sensitive Information in Error Messages
- LAB 261 – Defending Python Applications Against Sensitive Information in Error Messages
- LAB 262 – Defending Node.js Applications Against Sensitive Information in Error Messages
- LAB 263 – Defending Java Applications Against Sensitive Information in Log Files
- LAB 264 – Defending Python Applications Against Sensitive Information in Log Files
- LAB 265 – Defending Node.js Applications Against Sensitive Information in Log Files

- LAB 266 – Defending C# Applications Against Sensitive Information in Log Files
- LAB 267 – Defending Java Applications Against Deserialization of Untrusted Data
- LAB 268 – Defending Python Applications Against Deserialization of Untrusted Data
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data Node.js
- LAB 270 – Defending C# Applications Against Deserialization of Untrusted Data
- LAB 271 – Defending Java Applications Against SSRF
- LAB 272 – Defending Python Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 274 – Defending C# Applications Against SSRF

CSC 334 – Secure Systems Administrator

The Secure Systems Administrator Learning Path includes a variety of security courses designed for those responsible for preventing and mitigating security breaches that may arise within computer systems. The curriculum provides a holistic approach to network and system security with an exploration of controls, monitoring access, operational procedure, and formal auditing and logging.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

64 Courses

0 Labs

23 Hours

27 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 141 – Fundamentals of Database Security
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- ENG 110 – Essential Account Management Security
- ENG 111 – Essential Session Management Security
- ENG 113 – Essential Secure Configuration Management
- ENG 118 – Essential Incident Response
- ENG 119 – Essential Security Audit & Accountability
- ENG 121 – Essential Identification & Authentication
- ENG 122 – Essential Physical & Environmental Protection
- ENG 123 – Essential Security Engineering Principles
- ENG 125 – Essential Data Protection
- ENG 127 – Essential Media Protection
- ENG 150 – Meeting Confidentiality, Integrity, and Availability Requirements
- ENG 151 – Fundamentals of Privacy Protection

Advanced

- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management
- COD 219 – Creating Secure Code SAP ABAP Foundations
- COD 252 – Securing Google Platform Applications & Data
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- COD 263 – Secure Bash Scripting
- COD 264 – Secure Perl Scripting
- COD 265 – Secure Python Scripting
- COD 266 – Secure Ruby Scripting

- CYB 210 – Cybersecurity Incident Response
- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 209 – Authentication and Lifecycle Management
- DES 210 – Hardening Linux/Unix Systems
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 217 – Securing Terraform Infrastructure and Resources
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DES 219 – Securing Google's Firebase Platform
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- DES 262 – Securing Enterprise Low-Code Application Platforms
- DSO 201 – Fundamentals of Secure DevOps
- DSO 211 – Identifying Threats to Containers and Data in a DevSecOps Framework
- DSO 212 – Fundamentals of Zero Trust Security
- DSO 256 – DevSecOps in the Google Cloud Platform
- ENG 205 – Fundamentals of Threat Modeling

Elite

- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 303 – Automating Security Updates
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 305 – Automating CI/CD Pipeline Compliance
- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud
- CYB 310 - Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- ENG 320 - Using Software Composition Analysis (SCA) to Secure Open-Source Components

Elective

- COD 288 - Java Public Key Cryptography
- COD 287 - Java Application Server Hardening
- COD 383 - Protecting Java Backend Services

CSC 335 – Secure Database Administrator

The Secure Database Administrator Learning Path includes a variety of security courses designed for those responsible for capacity planning, installation, configuration, database design, migration, performance monitoring, security, troubleshooting, as well as back-end data recovery. The curriculum builds fundamental knowledge of secure database development including:

- Common database attacks
- Platform-specific threats
- Database secure coding best practices

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

36 Courses

0 Labs

13 Hours

16 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 101 – Fundamentals of Secure Architecture
- COD 141 – Fundamentals of Database Security

Advanced

- COD 241 – Creating Secure Code - Oracle Database Applications
- COD 242 – Creating Secure SQL Server and Azure SQL Database Applications
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- DES 202 – Cryptographic Suite Services: Encoding, Encrypting & Hashing
- DES 203 – Cryptographic Components: Randomness, Algorithms, and Key Management
- DES 204 – Role of Cryptography in Application Development
- DES 205 – Message Integrity Cryptographic Functions
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 212 – Architecture Risk Analysis and Remediation
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures

- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements

Elite

- COD 352 – Creating Secure jQuery Code
- COD 383 – Protecting Java Backend Services
- DES 311 – Creating Secure Application Architecture
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 312 – How to Perform a Security Code Review
- SDT 302 – Testing for Identification and Authentication Failures
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type
- SDT 315 – Testing for Incorrect Permission Assignment for Critical Resource
- SDT 316 – Testing for Use of Hard-Coded Credentials

CSC 336 – Secure Linux Administrator

The Secure Linux Administrator Learning Path dives into operating system configuration and administration of virtual servers. Learners will develop the working knowledge needed to support development, testing, and systems integration. Additionally, the curriculum will provide learners with a solid understanding of secure development best practices.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

16 Courses

0 Labs

7 Hours

8 CPE Credits

Core

- ENG 110 – Essential Account Management Security
- ENG 114 – Essential Risk Assessment
- ENG 115 – Essential System & Information Integrity
- ENG 119 – Essential Security Audit & Accountability
- ENG 121 – Essential Identification & Authentication
- ENG 150 – Meeting Confidentiality, Integrity, and Availability Requirements

Advanced

- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- COD 263 – Secure Bash Scripting
- COD 264 – Secure Perl Scripting
- COD 265 – Secure Python Scripting
- COD 266 – Secure Ruby Scripting
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 260 – Fundamentals of IoT Architecture and Design
- ENG 205 – Fundamentals of Threat Modeling

CSC 337 – Secure Product Owner

The Secure Product Owner Learning Path includes a variety of security courses designed for those responsible for setting, prioritizing, and evaluating the work generated by a software Scrum team to ensure impeccable features and functionality of the product. The curriculum introduces application security fundamentals including the essentials goals and controls needed to create secure software and manage risk in the software development lifecycle

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

42 Courses

0 Labs

15 Hours

18 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- ENG 124 – Essential Application Protection
- ENG 125 – Essential Data Protection
- ENG 150 – Meeting Confidentiality, Integrity, and Availability Requirements
- ENG 151 – Fundamentals of Privacy Protection
- ENG 191 – Introduction to the Microsoft SDL
- ENG 192 – Implementing the Agile Microsoft SDL
- ENG 193 – Implementing the Microsoft SDL Optimization Model
- ENG 194 – Implementing Microsoft SDL Line of Business
- ENG 195 – Implementing the Microsoft SDL Threat Modeling Tool
- TST 101 – Fundamentals of Security Testing

Advanced

- CYB 211 – Identifying and Protecting Assets Against Ransomware
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- DES 212 – Architecture Risk Analysis and Remediation
- DES 260 – Fundamentals of IoT Architecture and Design
- DSO 201 – Fundamentals of Secure DevOps

- ENG 211 – How to Create Application Security Design Requirements
- ENG 251 – Risk Management Foundations
- TST 202 – Penetration Testing Fundamentals
- TST 206 – ASVS Requirements for Developers

Elite

- CYB 310 – Using Cyber Supply Chain Risk Management (C-SCRM) to Mitigate Threat to IT/OT
- DSO 302 – Automated Security Testing
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 351 – Preparing the Risk Management Framework

CSC 338 – Secure Project Manager

The Secure Project Manager Learning Path includes a variety of security courses that introduces project managers to the essentials of access control, configuration management, risk assessment, auditing, and authentication. The curriculum provides the knowledge and skills necessary to ensure adherence to your organization's system and information security policies as well as relevant governmental and industry standards.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

42 Courses

0 Labs

15 Hours

18 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 102 – Challenges in Application Security
- COD 103 – Creating Software Security Requirements
- COD 104 – Designing Secure Software
- COD 105 – Secure Software Development
- COD 106 – The Importance of Software Integration and Testing
- COD 107 – Secure Software Deployment
- COD 108 – Software Operations and Maintenance
- COD 141 – Fundamentals of Database Security*
- COD 152 – Fundamentals of Secure Cloud Development*
- DES 101 – Fundamentals of Secure Architecture
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- ENG 123 – Essential Security Engineering Principles
- ENG 124 – Essential Applications Protection
- ENG 125 – Essential Data Protection
- ENG 150 – Meeting Confidentiality, Integrity, and Availability Requirements
- ENG 151 – Fundamentals of Privacy Protection

Advanced

- COD 252 – Securing Google Platform Applications & Data
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- DES 204 – The Role of Cryptography in Application Development
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 212 – Architecture Risk Analysis and Remediation
- DES 214 – Securing Infrastructure Architecture
- DES 215 – Defending Infrastructure
- DES 216 – Protecting Cloud Infrastructure
- DES 218 – Protecting Microservices, Containers, and Orchestration
- DSO 201 – Fundamentals of Secure DevOps

- DSO 206 – Securing the Open-Source Software Supply Chain
- DSO 211 – Identifying Threats to Containers and Data in a DevSecOps Framework
- DSO 212 – Fundamentals of Zero Trust Security
- DSO 256 – DevSecOps in the Google Cloud Platform
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 251 – Risk Management Foundations
- TST 206 – ASVS Requirements for Developers

Elite

- CYB 310 – Using Cyber Supply Chain Risk Management (C-SCRM) to Mitigate Threat to IT/OT
- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 302 – Automated Security Testing
- DSO 305 – Automating CI/CD Pipeline Compliance
- ENG 312 – How to Perform a Security Code Review
- ENG 351 – Preparing the Risk Management Framework

CSC 339 – Cyber Security Professional

The Elite Cyber Security Professional Learning Path includes a variety of security courses designed for those tasked with everything from the technical aspects of security, security policy, and everything in between.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

26 Courses

17 Labs

9 Hours

11 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- ENG 117 – Essential Information Security Program Planning
- ENG 118 – Essential Incident Response
- ENG 124 – Essential Application Protection
- ENG 151 – Fundamentals of Privacy Protection
- TST 101 – Fundamentals of Software Security Testing

Advanced

- API 250 – Controlling Access to the Kubernetes API
- CYB 210 – Cybersecurity Incident Response
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- CYB 212 – Fundamentals of Security Information & Event Management (SIEM)
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 207 – Mitigating OWASP API Security Top 10
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 217 – Securing Terraform Infrastructure and Resources
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DSO 212 – Fundamentals of Zero Trust Security
- TST 202 – Penetration Testing Fundamentals
- TST 206 – ASVS Requirements for Developers
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities

- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective XSS
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection

Elite

- CYB 310 – Using Cyber Supply Chain Risk Management (C-SCRM) to Mitigate Threat to IT/OT
- ENG 320 – Using Software Composition Analysis (SCA) to Secure Open-Source Components
- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud

CSC 340 – Secure Operations/IT Manager

The Elite Operations/IT Manager Learning Path includes a variety of security courses designed for those responsible for managing operations and sharing responsibility for project success and managing day-to-day IT processes. The curriculum covers secure IT Operations concepts including:

- Essential goals and controls needed for secure software development
- Managing risk associated with the software development lifecycle
- Developing, implementing, and ensuring compliance with operational application security policies and procedures

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

26 Courses

19 Labs

9 Hours

11 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- ENG 117 – Essential Information Security Program Planning
- ENG 118 – Essential Incident Response
- ENG 124 – Essential Application Protection
- ENG 151 – Fundamentals of Privacy Protection
- TST 101 – Fundamentals of Software Security Testing

Advanced

- API 250 – Controlling Access to the Kubernetes API
- CYB 210 – Cybersecurity Incident Response
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- CYB 212 – Fundamentals of Security Information & Event Management (SIEM)
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 207 – Mitigating OWASP API Security Top 10
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 217 – Securing Terraform Infrastructure and Resources
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 261– Securing Serverless Environments
- DES 262 – Securing Enterprise Low-Code Application Platforms
- DSO 212 – Fundamentals of Zero Trust Security
- TST 202 – Penetration Testing Fundamentals
- TST 206 – ASVS Requirements for Developers
- LAB 101 – Identifying Broken Access Control Vulnerabilities

- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective XSS
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection

Elite

- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud
- LAB 312 - ATT&CK: Network Service Identification
- LAB 313 - ATT&CK: Vulnerability Identification Using Vulnerability Databases

CSC 341 – Application Security Champion

The Elite Application Security Champion Learning Path includes a variety of security courses designed for those chartered with driving a culture of "Security Built-in" to the software development lifecycle. The curriculum explains application security concepts such as privacy, secure development and architecture, security testing, threat modeling, cryptography, and cyber threat analysis and remediation.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

38 Courses

17 Labs

13 Hours

16 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 102 – Challenges in Application Security
- COD 103 – Creating Software Security Requirements
- COD 104 – Designing Secure Software
- COD 105 – Secure Software Development
- COD 106 – The Importance of Software Integration and Testing
- COD 107 – Secure Software Deployment
- COD 108 – Software Operations and Maintenance
- ENG 124 – Essential Application Protection
- ENG 125 – Essential Data Protection
- ENG 150 – Meeting Confidentiality, Integrity, and Availability Requirements
- ENG 151 – Fundamentals of Privacy Protection
- TST 101 – Fundamentals of Security Testing

Advanced

- CYB 211 – Identifying and Protecting Assets Against Ransomware
- DES 204 – The Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis and Remediation
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures

- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans
- TST 206 – ASVS Requirements for Developers
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities
- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective XSS
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection

Elite

- CYB 310 – Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- DSO 302 – Automated Security Testing
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 312 – How to Perform a Security Code Review

CSC 342 – Information Security Specialist

The Elite Information Security Specialist Learning Path includes a variety of courses designed for those responsible for protecting systems, defining access privileges, control structures, and resources. The curriculum helps build the skills required to identify, protect, detect, and recover from risks, vulnerabilities, and threats to the security of information and/or data.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

80 Courses

31 Labs

30 Hours

36 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 141 – Fundamentals of Database Security
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- ENG 110 – Essential Account Management Security
- ENG 111 – Essential Session Management Security
- ENG 112 – Essential Access Control for Mobile Devices
- ENG 113 – Essential Secure Configuration Management
- ENG 114 – Essential Risk Assessment
- ENG 115 – Essential System & Information Integrity
- ENG 116 – Essential Security Planning Policy & Procedures
- ENG 117 – Essential Information Security Program Planning
- ENG 118 – Essential Incident Response
- ENG 119 – Essential Security Audit & Accountability
- ENG 120 – Essential Security Assessment & Authorization
- ENG 121 – Essential Identification & Authentication
- ENG 122 – Essential Physical & Environmental Protection
- ENG 123 – Essential Security Engineering Principles
- ENG 124 – Essential Application Protection
- ENG 125 – Essential Data Protection
- ENG 126 – Essential Security Maintenance Policies
- ENG 127 – Essential Media Protection
- ENG 151 – Fundamentals of Privacy Protection
- TST 101 – Fundamentals of Security Testing

Advanced

- API 210 – Mitigating APIs Lack of Resources & Rate Limiting
- API 211 – Mitigating APIs Broken Object Level Authorization
- API 213 – Mitigating APIs Mass Assignment
- API 214 – Mitigating APIs Improper Asset Management

- COD 241 – Creating Secure Code Oracle Foundations
- COD 242 – Creating Secure SQL Server & Azure SQL Database Applications
- COD 246 – PCI DSS 3: Protecting Stored Cardholder Data
- COD 247 – PCI DSS 4: Encrypting Transmission of Cardholder Data
- COD 248 – PCI DSS 6: Develop and Maintain Secure Systems and Applications
- COD 249 – PCI DSS 11: Regularly Test Security Systems and Processes
- COD 256 – Creating Secure Code Ruby on Rails Foundations
- COD 261 – Threats to Scripts
- COD 288 – Java Public Key Cryptography
- COD 287 – Java Application Server Hardening
- CYB 210 – Cybersecurity Incident Response
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- CYB 212 – Fundamentals of Security Information & Event Management (SIEM)
- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 207 – Mitigating OWASP API Security Top 10
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 212 – Architecture Risk Analysis and Remediation
- DES 217 – Securing Terraform Infrastructure and Resources
- DES 219 – Securing Google's Firebase Platform
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 261 – Securing Serverless Environments
- DES 262 – Securing Enterprise Low-Code Application Platforms
- DES 271 – OWASP M1: Mitigating Improper Platform Usage
- DES 272 – OWASP M2: Mitigating Insecure Data Storage
- DES 273 – OWASP M3: Mitigating Insecure Communication
- DES 274 – OWASP M4: Mitigating Insecure Authentication
- DES 275 – OWASP M5: Mitigating Insufficient Cryptography
- DES 276 – OWASP M6: Mitigating Insecure Authorization
- DES 277 – OWASP M7: Mitigating Client Code Quality
- DES 278 – OWASP M8: Mitigating Code Tampering
- DES 279 – OWASP M9: Mitigating Reverse Engineering
- DES 280 – OWASP M10: Mitigating Extraneous Functionality
- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- TST 206 – ASVS Requirements for Developers
- LAB 101 – Identifying Broken Access Control Vulnerabilities
- LAB 102 – Identifying Broken Object-Level Authorization Vulnerabilities
- LAB 103 – Identifying Broken User Authentication Vulnerabilities
- LAB 104 – Identifying Business Logic Flaw Vulnerabilities
- LAB 105 – Identifying Credential Dumping Vulnerabilities
- LAB 106 – Identifying Cross-Site Scripting Vulnerabilities
- LAB 107 – Identifying Injection Vulnerabilities

- LAB 108 – Identifying Reverse Engineering Vulnerabilities
- LAB 109 – Identifying Security Misconfiguration Vulnerabilities
- LAB 110 – Identifying Sensitive Data Exposure Vulnerabilities
- LAB 114 – Identifying Cookie Tampering
- LAB 115 – Identifying Reflective XSS
- LAB 116 – Identifying Forceful Browsing
- LAB 117 – Identifying Hidden Form Field
- LAB 118 – Identifying Weak File Upload Validation
- LAB 119 – Identifying Persistent XSS
- LAB 120 – Identifying XML Injection

Elite

- COD 383 – Protecting Java Backend Services
- CYB 310 - Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- CYB 311 - Threat Analysis with AI
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 312 – How to Perform a Security Code Review
- ENG 320 - Using Software Composition Analysis (SCA) to Secure Open-Source Components
- LAB 312 - ATT&CK: Network Service Identification
- LAB 313 - ATT&CK: Vulnerability Identification Using Vulnerability Databases
- LAB 315 – ATT&CK: Updating Vulnerable Java Web Application Server Software
- LAB 321 – ATT&CK: Password Cracking
- LAB 322 – ATT&CK: Exploiting Windows File Sharing Server EternalRomance Remote Services
- LAB 323 – ATT&CK: Exploiting Vulnerable Java Web Application Server Software
- LAB 324 – ATT&CK: Exploiting Java Web Application Server Misconfiguration
- LAB 330 – ATT&CK: Exploiting Java SQL Injection to Extract Password Hashes
- LAB 331 – ATT&CK: Network Service Discovery
- LAB 332 – ATT&CK: Network Share Discovery
- LAB 334 – ATT&CK: Create Account
- LAB 335 – ATT&CK: Unsecured Credentials
- LAB 336 – ATT&CK: Data from Local System
- LAB 337 – ATT&CK: Valid Accounts
- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud

CSC 343 – Secure Systems Leadership

The Secure Systems Leadership Learning Path includes a variety of security courses designed for those responsible for computers and their complex operating systems. The curriculum explores application security best practices necessary to ensure strategies and plans support business needs and align with departmental and organizational objectives and goals. Learners will gain comprehensive application security knowledge and skills necessary for leading application development and design projects.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

30 Courses

0 Labs

10 Hours

12 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 151 – Fundamentals of the PCI Secure SLC Standard

Advanced

- COD 287 – Java Application Server Hardening
- COD 288 – Java Public Key Cryptography
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- DES 219 – Securing Google's Firebase Platform
- DES 232 – Mitigating OWASP 2021 Injection
- DES 233 – Mitigating OWASP 2021 Identification and Authentication Failures
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 236 – Mitigating OWASP 2021 Broken Access Control
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DES 239 – Mitigating OWASP 2021 Software and Data Integrity Failures
- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components
- DES 241 – Mitigating OWASP 2021 Security Logging and Monitoring Failures
- DES 262 – Securing Enterprise Low-Code Application Platforms
- DSO 201 – Fundamentals of Secure DevOps
- DSO 212 – Fundamentals of Zero Trust Security
- TST 206 – ASVS Requirements for Developers

Elite

- COD 383 – Protecting Java Backend Services
- CYB 310 – Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- DES 311 – Creating Secure Application Architecture

- DSO 301 – Orchestrating Secure System and Service Configuration
- DSO 302 – Automated Security Testing
- DSO 303 – Automating Security Updates
- DSO 305 – Automating CI/CD Pipeline Compliance
- ENG 320 – Using Software Composition Analysis (SCA) to Secure Open-Source Components

CSC 344 – Secure Development Manager

The Secure Development Manager Learning Path includes a variety of security courses designed for those responsible for planning, preparing, and ensuring that projects are completed. The curriculum introduces application security best practices required to adhere to system and information security policies and compliance. Learners can apply these best practices to the requirements, design, and implementation phases of the software development lifecycle.

Each learning path may consist of course content that is not covered as part of certification exams. These courses are considered elective training and suggested based on our alignment with the National Initiative for Cybersecurity Education **(NICE) Cybersecurity Workforce Framework. To understand how courses map to this framework, please contact us.*

Primary Training Details

23 Courses

0 Labs

9 Hours

11 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- DES 101 – Fundamentals of Secure Architecture
- DES 151 – Fundamentals of the PCI Secure SLC Standard
- ENG 110 – Essential Account Management Security
- ENG 114 – Essential Risk Assessment
- ENG 117 – Essential Information Security Program Planning
- ENG 151 – Fundamentals of Privacy Protection
- ENG 191 – Introduction to the Microsoft SDL
- ENG 192 – Implementing the Agile Microsoft SDL
- ENG 193 – Implementing the Microsoft SDL Optimization Model
- ENG 194 – Implementing Microsoft SDL Line of Business
- ENG 195 – Implementing the Microsoft SDL Threat Modeling Tool

Advanced

- CYB 211 – Identifying and Protecting Assets Against Ransomware
- DES 255 – Securing the IoT Update Process
- DES 208 – Defending Against the CSA Top 11 Threats to Cloud Computing
- DES 260 – Fundamentals of IoT Architecture and Design
- DSO 201 – Fundamentals of Secure DevOps
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- TST 206 – ASVS Requirements for Developers

Elite

- DSO 302 – Automated Security Testing
- DSO 305 – Automating CI/CD Pipeline Compliance
- CYB 310 - Using Cyber Supply Chain Risk Management(C-SCRM) to Mitigate Threats to IT/OT
- ENG 320 - Using Software Composition Analysis (SCA) to Secure Open Source Components