Learning Paths

Software Security Role-Based Curriculum

SECURITY INNOVATION

Contents

.NET Developer

The .NET learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed to provide a solid foundation of .NET security features for building secure web applications, sophisticated desktop applications, or modern mobile applications. Security concepts covered within this learning path include:

- Code Access Security (CAS)
- .NET cryptographic technologies
- Secure Coding best practices

More advanced courses offer application framework specific secure coding best practices for ASP.NET to extend the .NET Developer platform with tools and libraries for building web applications. Round off security expertise with knowledge and skills to apply security principles for creating secure application architecture and conduct effective security code reviews.

35
Courses

21
Labs

15
Hours

17
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

Advanced

- COD 216 – Leveraging .NET Framework Code Access Security (CAS)
- COD 217 – Mitigating .NET Security Threats
- COD 255 – Creating Secure Code: Web API Foundations
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & 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- DES 240 – Mitigating OWASP 2021 Vulnerable and Outdated Components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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 221 – Defending C# Applications Against SQL Injection (NEW)
- LAB 241 – Defending C# Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 232 – Defending C# Applications Against XSS (NEW) (XSS)
- LAB 238 – Defending C# Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 239 – Defending C# Applications Against Weak PRNG (NEW)
- LAB 250 – Defending C# Applications Against Parameter Tampering (NEW)
- LAB 251 – Defending C# Applications Against Plaintext Password Storage (NEW)
- 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 274 – Defending C# Applications Against SSRF

Elite

- COD 308 – Common ASP.NET MVC Vulnerabilities and Attacks
- COD 309 – Securing ASP.NET MVC Applications
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type

Android Developer

The Android Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed to provide a solid foundation of security features necessary to develop applications for devices powered by the Android operating system. The Android Developer learning path provides secure coding best practices for designing and building android applications including:

- Identifying common android application risks
- Creating a mobile application threat model
- Applying android platform specific knowledge

Round off security expertise with knowledge and skills to apply security principles for creating secure application architecture and conduct effective security code reviews.

35 Courses

10 Labs

12 Hours

14 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 110 – Fundamentals of Secure Mobile Development
- DES 101 – Fundamentals of Secure Architecture
- ENG 112 – Essential Access Control for Mobile Devices

Advanced

- COD 286 – Creating Secure React User Interfaces
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 260 – Fundamentals of IoT Architecture & Design
- 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 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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

Elite

- COD 318 – Protecting Data on Android in Java
- COD 319 – Preventing Vulnerabilities in Android Code in Java
- COD 366 – Creating Secure Kotlin Applications
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 316 Testing for Use of Hard-Coded Credentials

Back-End Developer

The Back-end Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed to provide a solid foundation of security features needed to write web services and API's used by front-end and mobile application developers. The Back-end Developer learning path 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.

42
Courses

10
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
- DES 101 – Fundamentals of Secure Architecture

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 Oracle DB Applications
- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 267 – Securing Python Microservices
- COD 287 – Java Application Server Hardening
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- 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
- 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

Elite

- COD 352 – Creating Secure JavaScript and jQuery Code
- 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)
- COD 383 – Protecting Java Backend Services
- DES 311 – Creating Secure Application Architecture
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review

C Developer

The C Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is 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 C Developer learning path covers key application security concepts including:

- Memory management and string handling
- Avoiding common pitfalls
- C specific security flaws

31 Courses

10 Labs

11 Hours

13 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

Advanced

- COD 201 – Secure C Encrypted Network Communications
- COD 202 – Secure C Runtime Protection
- COD 261 – Threats to Scripts
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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

Elite

- COD 301 – Secure C Buffer Overflow Mitigations
- COD 302 – Secure C Memory Management
- COD 303 – Common C Vulnerabilities & Attacks
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- 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

C# Developer

The C# Developer Learning Path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It builds a thorough grounding of security features necessary to develop modern applications that run on desktops or back-end processes powering modern web applications. The C# Developer learning path covers key application security concepts including:

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

39 Courses

21 Labs

15 Hours

19 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

Advanced

- COD 216 – Leveraging .NET Framework Code Access Security (CAS)
- COD 217 – Mitigating .NET Security Threats
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- 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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 221 – Defending C# Applications Against SQL Injection (NEW)
- LAB 232 – Defending C# Applications Against XSS (NEW) (XSS)
- LAB 238 – Defending C# Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 239 – Defending C# Applications Against Weak PRNG (NEW)
- LAB 241 – Defending C# Against eXternal XML Entity (XXE) Vulnerabilities
- LAB 250 – Defending C# Applications Against Parameter Tampering (NEW)
- LAB 251 – Defending C# Applications Against Plaintext Password Storage (NEW)
- 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 274 – Defending C# Applications Against SSRF

Elite

- COD 308 – Common ASP.NET MVC 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#
- COD 324 – Protecting C# from XML Injection
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 325 – Testing for NULL Pointer Dereference

C++ Developer

The C++ Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is 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. It also provides the knowledge needed to build efficient, reusable, and reliable C++ code that interacts with low-level systems and hardware resources. Learners will develop the knowledge and skills required 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

36 Courses

10 Labs

13 Hours

14 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

Advanced

- 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
- COD 263 – Secure Bash Scripting
- COD 264 – Secure Perl Scripting
- COD 265 – Secure Python Scripting
- COD 266 – Secure Ruby Scripting
- DES 203 – Cryptographic Components: Randomness, Algorithms, and Key Management
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 237 Mitigating OWASP 2021 Security Misconfiguration
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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

Elite

- COD 307 – Protecting Data in C++
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 302 – Testing for Identification and Authenticaiton 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

Front-End Developer

The Front-end Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It provides a solid foundation for using markup languages, design and client-side scripts and framework to create secure environments for everything that users touch. The Front-End Developer learning path covers key application security concepts including:

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

46 Courses

21 Labs

18 Hours

22 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

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 214 – Creating Secure GO Applications
- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats & Vulnerabilities
- COD 285 – Developing Secure Angular Applications
- COD 286 – Creating Secure React User Interfaces
- DES 204 – Role of Cryptography in Application Development
- DES 212 – Architecture Risk Analysis & 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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 223 – Defending Node.js Applications Against SQL Injection (NEW)
- LAB 233 – Defending Node.js Applications Against XSS (NEW)
- LAB 242 – Defending Node.js Applications Against eXternal XML Entity (XXE) Vulnerabilities (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 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 Node.js
- LAB 273 – Defending Node.js Applications Against SSRF

Elite

- 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 – Securing HTML5 Connectivity
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 301 – Testing for Injection
- SDT 316 – Testing for Use of Hard-coded Credentials

HTML5 Developer

The HTML5 Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is 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 HTML5 Developer learning path covers key application security concepts including:

- HTML5 security features
- How to infuse software security into the development lifecycle
- Working knowledge of ASP.net, SWL, high-level scripting languages, version control and CMS systems

36 Courses

34 Labs

19 Hours

22 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

Advanced

- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 259 – Node.js Threats & Vulnerabilities
- COD 281 – Java Security Model
- COD 285 – Developing Secure Angular Applications
- DES 204 – Role of Cryptography in Application Development
- DES 212 – Architecture Risk Analysis & Remediation
- DES 232 – Mitigating OWASP 2021 Injection
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 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 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 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 (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 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 Node.js
- LAB 271 – Defending Java Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF

Elite

- COD 308 – Common ASP.NET MVC Vulnerabilities and Attacks
- COD 309 – Securing ASP.NET MVC Applications
- 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 – Securing HTML5 Connectivity
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review

iOS Developer

The iOS Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed to provide developers with a solid foundation of security features necessary to develop applications for devices powered by the iOS platform. The iOS Developer learning path provides secure coding best practices for designing and building iOS applications including:

- Identifying common iOS application risks
- Creating a mobile application threat model
- Applying iOS platform-specific knowledge

35 Courses

10 Labs

12 Hours

14 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 110 – Fundamentals of Secure Mobile Development
- DES 101 – Fundamentals of Secure Architecture
- ENG 112 – Essential Access Control for Mobile Devices

Advanced

- COD 286 – Creating Secure React User Interfaces
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 260 – Fundamentals of IoT Architecture & Design
- 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 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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

Elite

- 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
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 301 – Testing for Injection
- SDT 316 – Testing for Use of Hard-coded Credentials

Java Developer

The Java Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced, or elite paths. It 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 Java Developer learning path covers key application security concepts including:

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

60
Courses

22
Labs

22
Hours

27
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

Advanced

- COD 219 – Creating Secure Code: SAP ABAP Foundations
- COD 251 – Defending AJAX-Enabled Web Applications
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 259 – Node.js Threats & Vulnerabilities
- COD 281 – Java Security Model
- COD 283 – Java Cryptography
- COD 284 – Secure Java Coding
- COD 287 – Java Application Server Hardening
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & 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
- 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 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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 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 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 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

Elite

- 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 – Securing 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 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 325 – Testing for NULL Pointer Dereference

JavaScript Developer

The JavaScript Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is 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 JavaScript Developer learning provides a thorough grounding in application security concepts and implementation practices including:

- JavaScript security flaws
- Proven techniques to help protect JavaScript
- Avoiding common pitfalls

41
Courses

33
Labs

21
Hours

26
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

Advanced

- COD 241 – Creating Secure Oracle DB Applications
- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats & Vulnerabilities
- COD 281 – Java Security Model
- COD 283 – Java Cryptography
- COD 284 – Secure Java Coding
- COD 285 – Developing Secure Angular Applications
- COD 286 – Creating Secure React User Interfaces
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 232 – Mitigating OWASP 2021 Injection
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements

- ENG 212 – Implementing Secure Software Operations
- 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 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 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 240 – Defending Java Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 242 – Defending Node.js 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 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 Node.js
- LAB 271 – Defending Java Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF

Elite

- 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 – Securing HTML5 Connectivity
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures
- SDT 306 – Testing for Security Misconfiguration
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 307 – Secure Secrets Management

- ENG 312 – How to Perform a Security Code Review

Mobile Developer

The Mobile Developer learning path includes a variety of courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed to provide developers with a solid foundation of security features necessary to develop applications for mobile devices. The Mobile Developer learning path covers key application security concepts including:

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

43 Courses

10 Labs

15 Hours

18 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 110 – Fundamentals of Secure Mobile Development
- DES 101 – Fundamentals of Secure Architecture
- ENG 112 – Essential Access Control for Mobile Devices

Advanced

- COD 261 – Threats to Scripts
- COD 286 – Creating Secure React User Interfaces
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture & Design
- 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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations

- 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

Elite

- 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
- COD 366 – Creating Secure Kotlin Applications
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- 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

PHP Developer

The PHP learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is 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 front-end, developers. Learners will be able to apply these security best practices to the entire web application development life cycle from concept stage to delivery and post-launch.

59 Courses

33 Labs

23 Hours

28 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

Advanced

- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats & Vulnerabilities
- 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
- COD 281 – Java Security Model
- COD 283 – Java Cryptography
- COD 284 – Secure Java Coding
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & 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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 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 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 240 – Defending Java Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 242 – Defending Node.js 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 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 Node.js
- LAB 271 – Defending Java Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF

Elite

- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built in Security Features
- COD 363 – Securing HTML5 Data

- COD 364 – Securing HTML5 Connectivity
- 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
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type

Python Developer

The Python Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed 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

43 Courses

21 Labs

19 Hours

22 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

Advanced

- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 257 – Creating Secure Python Web Applications
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- COD 265 – Secure Python Scripting
- COD 267 – Securing Python Microservices
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & 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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 222 – Defending Python Applications Against SQL Injection (NEW)
- LAB 231 – Defending Python Applications Against XSS (NEW)
- LAB 243 – Defending Python Applications Against eXternal XML Entity (XXE) Vulnerabilities (NEW)
- LAB 249 – Defending Python 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 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

Elite

- COD 361 – HTML5 Secure Threats
- COD 362 – HTML5 Built in Security Features
- COD 363 – Securing HTML5 Data
- COD 364 – Securing HTML5 Connectivity
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 306 – Implementing Infrastructure as Code
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review

Ruby on Rails Developer

The Ruby on Rails learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. This path is designed for those responsible for writing server-side web application logic in Ruby, around the frame rails. It provides best practices and techniques for secure application development, including:

- Understanding various classes of vulnerabilities
- Building strong session management
- Preventing vulnerabilities commonly found in Rails applications

39 Courses

44 Labs

22 Hours

27 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

Advanced

- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 257 – Creating Secure Python Web Applications
- COD 259 – Node.js Threats & Vulnerabilities
- COD 281 – Java Security Model
- COD 283 – Java Cryptography
- COD 284 – Secure Java Coding
- COD 287 – Java Application Server Hardening
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 232 – Mitigating OWASP 2021 Injection
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 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 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 240 – Defending Java 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 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 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 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 271 – Defending Java Applications Against SSRF
- LAB 272 – Defending Python Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF

Elite

- 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 – Securing HTML5 Connectivity
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 306 – Implementing Infrastructure as Code
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review

Web Developer

The Web Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for the development of web applications or applications that are run over HTTP from a web server to a web browser. The Web Developer 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

61 Courses

88 Labs

34 Hours

41 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
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services

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 Oracle DB Applications
- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 257 – Creating Secure Python Web Applications
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats & Vulnerabilities
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- COD 285 – Developing Secure Angular Applications
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & 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
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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
- 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# Applicaitons Against SSRF

Elite

- 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 – Securing HTML5 Connectivity

- 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
- SDT 313 – Testing for Cross-Site Request Forgery (CSRF)
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 314 – Testing for Unrestricted Upload of File with Dangerous Type

Node.js Developer

The Node.js learning path includes a variety of security courses that vary depending on whether you are seeking core, advanced or elite paths. It is designed for those that managing the interchange of data between the server and the users and provides developers a solid foundation of security features necessary to code, test and operate including:

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

41 Courses

32 Labs

21 Hours

25 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

Advanced

- COD 241 – Creating Secure Oracle DB Applications
- COD 251 – Defending AJAX-Enabled Web Applications
- COD 255 – Creating Secure Code: Web API Foundations
- COD 256 – Creating Secure Code: Ruby on Rails Foundations
- COD 257 – Creating Secure Python Web Applications
- COD 258 – Creating Secure PHP Web Applications
- COD 259 – Node.js Threats & Vulnerabilities
- COD 285 – Developing Secure Angular Applications
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & Remediation
- DES 232 – Mitigating OWASP 2021 Injection
- DES 234 – Mitigating OWASP 2021 Cryptographic Failures
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 222 – Defending Python Applications Against SQL Injection (NEW)
- LAB 223 – Defending Node.js Applications Against SQL Injection (NEW)
- LAB 231 – Defending Python Applications Against XSS (NEW)
- LAB 233 – Defending Node.js Applications Against XSS (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 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 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 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 265 – Defending Node.js Applications Against Sensitive Information in Log Files
- LAB 268 – Defending Python Applications Against Deserialization of Untrusted Data
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data Node.js
- LAB 272 – Defending Python Applications Against SSRF
- LAB 273 – Defending Node.js Applications Against SSRF

Elite

- COD 308 – Common ASP.NET MVC Vulnerabilities and Attacks
- COD 309 – Securing ASP.NET MVC Applications
- 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 – Securing HTML5 Connectivity
- SDT 301 – Testing for Injection
- SDT 303 – Testing for Cryptographic Failures
- SDT 306 – Testing for Security Misconfiguration
- DES 311 – Creating Secure Application Architecture
- DSO 304 – Securing API Gateways in a DevSecOps Framework
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review

Swift Developer

The Swift Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. This path is designed for those responsible for the development of applications aimed towards iOS and OS X and the integration with back-end services. The Swift Developer learning path covers key application security concepts including:

- 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

30 Courses

10 Labs

11 Hours

13 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 110 – Fundamentals of Secure Mobile Development
- DES 101 – Fundamentals of Secure Architecture
- ENG 112 – Essential Access Control for Mobile Devices

Advanced

- DES 204 – Role of Cryptography in Application Development
- DES 212 – Architecture Risk Analysis & Remediation
- 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 286 – OWASP IoT6: Mitigating Insufficient Privacy Protection
- DES 287 – OWASP IoT7: Mitigating Insecure Data Transfer and Storage
- DES 289 – OWASP IoT9: Mitigating Insecure Default Settings
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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

Elite

- COD 315 – Preventing Vulnerabilities in iOS Code in Swift
- COD 317 – Protecting Data on iOS in Swift
- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review
- SDT 301 – Testing for Injection
- SDT 316 – Testing for Use of Hard-coded Credentials

Microsoft SDL Developer

The MS SDL Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for implementing the industry-leading software security assurance process. The MS SDL Developer learning path describes how to take a holistic and practical approach when implementing the SDL to ensure security and privacy is considered at every phase of development.

27 Courses

10 Labs

11 Hours

14 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
- 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

- COD 216 – Leveraging .NET Framework Code Access Security (CAS)
- COD 217 – Mitigating .NET Security Threats
- COD 242 – Creating Secure SQL Server & Azure SQL DB Applications
- COD 254 – Creating Secure Azure Applications
- DES 204 – Role of Cryptography in Application Development
- DES 212 – Architecture Risk Analysis & Remediation
- DES 235 – Mitigating OWASP 2021 Insecure Design
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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

Elite

- DES 311 – Creating Secure Application Architecture
- DSO 307 – Secure Secrets Management
- ENG 312 – How to Perform a Security Code Review

Cloud Developer

The Cloud Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for the design, development, and deployment of cloud applications and provides learners with a clear understanding of how to mitigate cloud computing risks. 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 and/or Azure

66 Courses

37 Labs

27 Hours

32 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 152 – Fundamentals of Secure Cloud Development
- DES 101 – Fundamentals of Secure Architecture
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services

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 214 – Creating Secure GO Applications
- COD 241 – Creating Secure Oracle DB Applications
- COD 252 – Securing Google Platform Applications & Data (NEW)
- 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 & Vulnerabilities
- COD 261 – Threats to Scripts
- COD 267 – Securing Python Microservices
- DES 204 – Role of Cryptography in Application Development
- 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 212 – Architecture Risk Analysis & 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
- 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 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 (NEW)
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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 122 – Identifying Insecure APIs
- LAB 123 – Identifying Vertical Privilege Escalation
- LAB 221 – Defending C# Applications Against SQL Injection (NEW)
- LAB 223 – Defending Node.js Applications Against SQL Injection (NEW)
- LAB 232 – Defending C# Applications Against XSS (NEW)
- LAB 233 – Defending Node.js Applications Against XSS (NEW)
- LAB 238 – Defending C# Applications Against Weak AES ECB Mode Encryption (NEW)
- LAB 239 – Defending C# Applications Against Weak PRNG (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 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 250 – Defending C# Applications Against Parameter Tampering (NEW)
- LAB 251 – Defending C# Applications Against Plaintext Password Storage (NEW)
- LAB 260 – Defending C# Applications Against Sensitive Information in Error Messages
- 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 266 – Defending C# Applications Against Sensitive Information in Log Files
- LAB 269 – Defending Node.js Applications Against Deserialization of Untrusted Data Node.js
- LAB 270 – Defending C# Applications Against Deserialization of Untrusted Data
- LAB 273 – Defending Node.js Applications Against SSRF
- LAB 274 – Defending C# Applications Against SSRF

Elite

- DES 311 – Creating Secure Application Architecture
- 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
- DSO 307 – Secure Secrets Management
- ENG 311 – Attack Surface Analysis & Reduction
- ENG 312 – How to Perform a Security Code Review

PCI Developer

The PCI learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for developing applications that process credit and debit card payments and/or any type of cardholder data. The PCI Developer learning path 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. The courses within the PCI Developer learning path provide a framework for:

- Developing secure applications
- Conducting effective test procedures
- Adopting guidance for mitigating issues

57 Courses

10 Labs

19 Hours

23 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

Advanced

- COD 241 – Creating Secure Oracle DB 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 251 – Defending AJAX-Enabled Web Applications
- COD 252 – Securing Google Platform Applications & Data
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 209 – Authentication and Lifecycle Management
- DES 212 – Architecture Risk Analysis & 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
- 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 256 – DevSecOps in the Google Cloud Platform (NEW)
- ENG 205 – Fundamentals of Threat Modeling
- ENG 211 – How to Create Application Security Design Requirements
- ENG 212 – Implementing Secure Software Operations
- 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

Elite

- DES 311 – Creating Secure Application Architecture
- DES 312 – Protecting Cardholder Data
- DSO 307 – Secure Secrets Management
- ENG 311 – Attack Surface Analysis & Reduction
- ENG 312 – How to Perform a Security Code Review

IoT & Embedded Developer

The IoT/Embedded learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed to provide developers 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 IoT/Embedded learning path 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.

43 Courses

10 Labs

17 Hours

20 CPE Credits

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 110 – Fundamentals of Secure Mobile Development
- COD 160 – Fundamentals of Secure Embedded Software Development
- DES 101 – Fundamentals of Secure Architecture

Advanced

- COD 201 – Secure C Encrypted Network Communications
- COD 202 – Secure C Runtime Protection
- COD 206 – Creating Secure C++ Code
- COD 207 – Communication Security in C++
- COD 261 – Threats to Scripts
- DES 204 – Role of Cryptography in Application Development
- DES 212 – Architecture Risk Analysis & Remediation
- DES 235 – Mitigating OWASP 2021 insecure Design
- DES 237 – Mitigating OWASP 2021 Security Misconfiguration
- DES 255 – Securing the IoT Update Process
- DES 260 – Fundamentals of IoT Architecture & 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 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
- 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

Elite

- COD 301 – Secure C Buffer Overflow Mitigations
- COD 302 – Secure C Memory Management
- COD 303 – Common C Vulnerabilities & Attacks
- COD 307 – Protecting Data in C++
- COD 366 – Creating Secure Kotlin Applications
- DES 311 – Creating Secure Application Architecture
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- DSO 302 – Automated Security Testing
- DSO 307 – Secure Secrets Management
- 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

Core Developer

The Core Developer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for the design, development, and management of applications across various environments and operating platforms and provides learners with a solid foundation of application security best practices. The Core Developer learning path covers key application security concepts including:

- Application security and risk drivers
- Essential security engineering principles: defensive coding, threat modeling, and gathering security design requirements
- How to identify and mitigate CWE's 25 most dangerous software errors

36
Courses

25
Labs

14
Hours

17
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
- DES 101 – Fundamentals of Secure Architecture
- LAB 131 – Identifying Improper Restriction of XML External Entity Reference
- LAB 132 – Identifying Exposed Services

Advanced

- CYB 211 – Identifying and Protecting Assets Against Ransomware
- CYB 250 – Cyber Threat Hunting: Tactics, Techniques, and Procedures (TTP)
- DES 204 – Role of Cryptography in Application Development
- DES 207 – Mitigating OWASP API Security Top 10
- DES 212 – Architecture Risk Analysis & 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
- ENG 212 – Implementing Secure Software Operations
- 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 220 – Defending Against Hard-Coded Secrets

Elite

- DES 311 – Creating Secure Application Architecture
- DES 313 – Hardening a Kubernetes Cluster
- DES 314 – Hardening the Docker Engine
- DSO 302 – Automated Security Testing
- DSO 307 – Secure Secrets Management
- 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
- ENG 312 – How to Perform a Security Code Review

DevOps Practitioner

The DevOps Practitioner path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those who work closely with Software Engineers to help them deploy and operate various systems. The DevOps Practitioner learning path 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.

52
Courses

17
Labs

19
Hours

23
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)
- 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 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

- COD 383 – Protecting Java Backend Services
- 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

Ethical Hacker

The Ethical Hacker Learning Path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. This path is 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. Courses provide 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.

This learning path provides 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

74 Courses

41 Labs

26 Hours

31 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 150 – Meeting Confidentiality, Integrity, and Availability
- ENG 151 – Fundamentals of Privacy Protection
- ENG 191 – Introduction to the Microsoft SDL
- 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 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

Advanced

- ATK 201 – Using the MiTRE ATT&CK Framework
- COD 249 – PCI DSS 11: Regularly Testing Security Systems and Procedures
- COD 261 – Threats to Scripts
- COD 262 – Fundamentals of Shell and Interpreted Language Security
- COD 287 – Java Application Server Hardening
- CYB 210 – Cybersecurity Incident Response
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- 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 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 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
- LAB 111 – Identifying Server-Side Request Forgery
- 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
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans

Elite

- CYB 301 – Fundamentals of Ethical Hacking
- 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 311 – Attack Surface Analysis & 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 Risk Management Framework
- ENG 354 – Authorizing and Monitoring System Controls within the RMF
- 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
- 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 321 – Testing for Uncontrolled Resource Consumption
- TST 301 – Infrastructure Penetration Testing
- TST 302 – Application Penetration Testing
- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for Google Cloud Platform
- TST 305 – Penetration Testing for Google Cloud Platform
- 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

Network Engineer

The Network Engineer path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for planning, implementing, and overseeing computer networks that support in-house voice, data, video and wireless network services. This learning path covers core security concepts including:

- Best practices for managing systems and services across all environments
- How to improve the stability, security, efficiency, and scalability of environments
- Gaining a baseline understanding of how to create and modify scripts to perform tasks

47 **Courses**

17 **Hours**

20 **CPE Credits**

Core

- AWA 101 – Fundamentals of Application Security
- AWA 102 – Secure Software Concepts
- COD 110 – Fundamentals of Secure Mobile Development
- 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
- 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 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 211 – Identifying and Protecting Assets Against Ransomware
- CYB 212 – Fundamentals of Security Information & Event Management (SIEM)
- 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 218 – Protecting Microservices, Containers, and Orchestration
- DES 260– Fundamentals of IoT Architecture and Design

- DSO 211– Identifying Threats to Containers and Data in a DevSecOps Framework
- DSO 212 – Fundamentals of Zero Trust Security
- ENG 205 – Fundamentals of Threat Modeling
- ICS 210 – ICS/SCADA Security Essentials
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans

Elite

- 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 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

Automation Engineer

The Automation Engineer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those who design, program, simulate and test automated machinery and processes to complete exact tasks. The Automation Engineer path covers key security topics including:

- Essential goals and controls needed to create secure software
- Managing risk in the software development lifecycle
- Cryptography, handling input and output
- OWASP Top Ten

32 **Courses**

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

Embedded Test Engineer

The Embedded QA/Test Engineer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for verifying and assuring the application security of embedded systems. The Embedded QA/Test Engineer learning path provides learners with a solid understanding of applied testing techniques and a well-rounded base of knowledge to perform their tasks. This path also explores security best practices for conducting penetration tests and vulnerability assessment activities on embedded systems.

57 **Courses**

16 **Hours**

19 **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

QA Test Engineer

The Quality Assurance (QA)/Test Engineer learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for assessing and testing the quality of specifications and technical design.

79 Courses

41 Labs

16 Hours

20 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

IT Architect

The IT Architect learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for designing and maintaining computer networks. The IT Architect path covers key application security concepts including:

- Best practices for secure software design
- Creating integrated architecture across business and technology
- Protecting data and resources from disclosure, modification, and deletion

38 Courses

4 Lab

15 Hours

18 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
- 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

- 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 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

Embedded Architect

The Embedded Architect learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is 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.

11
Courses

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

Software Architect

The Software Architect learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those making design choices, coordinating, and overseeing technical standards and includes software coding standards, tools, and platforms. The Software Architect path covers key application security concepts including:

- Secure software architecture best practices that can be applied to early phase SDLC activities
- Defensive coding techniques
- Avoiding systemic issues found in insecure software

79
Courses

74
Labs

38
Hours

45
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
- 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

Elite

- 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
- 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

Business Analyst

The Business Analyst learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for defining, analyzing, and documenting requirements in the software development lifecycle. The Business Analyst path covers core application security concepts including:

- Adhering to system and information security policies
- Meeting compliance mandates for relevant government and industry standards
- Access control, configuration management, risk assessment, auditing and authentication

17 **Courses**

6 **Hours**

7 **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

Systems Analyst

The Systems Analyst learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those who specialize in the implementation of computer system requirements. The Systems Analyst learning path 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

56 Courses

63 Labs

24 Hours

29 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
- 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# Applicaitons Against SSRF

Elite

- 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 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

Systems Administrator

The Systems Administrator Learning Path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for preventing and mitigating security breaches that may arise within computer systems. The Systems Administrator learning path provides a holistic approach to network and system security with an exploration of controls, monitoring access, operational procedure, and formal auditing and logging.

60
Courses

21
Hours

25
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 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 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

Database Administrator

The Database Administrator Learning Path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for capacity planning, installation, configuration, database design, migration, performance monitoring, security, troubleshooting, as well as back-end data recovery. The Database Administrator learning path builds fundamental knowledge of secure database development including:

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

35 **Courses**

14 **Hours**

17 **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

- SDT 302 – Testing for Identification and Authentication Failures
- COD 352 – Creating Secure jQuery Code
- DES 311 – Creating Secure Application Architecture
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 312 – How to Perform a Security Code Review
- 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

Linux Administrator

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

17 Courses

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

Product Owner

The Product Owner learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is 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 Product Owner learning path introduces application security fundamentals including the essentials goals and controls needed to create secure software and manage risk in the software development lifecycle.

34 **Courses**

11 **Hours**

13 **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

- DSO 302 – Automated Security Testing
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 351 – Preparing the Risk Management Framework

Project Manager

The Project Manager learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It introduces project managers to the essentials of access control, configuration management, risk assessment, auditing, and authentication. It also 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.

41 **Courses**

15 **Hours**

17 **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

- 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

Cyber Security Professional

The Cybersecurity Professional learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those tasked with everything from the technical aspects of security, security policy and everything in between.

24 Courses

17 Labs

8 Hours

9 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

- TST 303 – Penetration Testing for Google Cloud Platform
- TST 304 – Penetration Testing for AWS Cloud
- TST 305 – Penetration Testing for Azure Cloud

Operations/IT Manager

The Operations/IT Learning Path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for managing operations and sharing responsibility for project success and managing day-to-day IT processes. The Operations/IT Manager path covers key security 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

53
Courses

17
Hours

21
CPE Credits

Core

- 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 150 – Meeting Confidentiality, Integrity, and Availability Requirements
- ENG 151 – Fundamentals of Privacy Protection

Advanced

- API 250 – Controlling Access to the Kubernetes API
- COD 252 – Securing Google Platform Applications & Data
- 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 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 238 – Mitigating OWASP 2021 Server-Side Request Forgery (SSRF)
- DSO 201 – Fundamentals of Secure DevOps
- DSO 205 – Securing the COTS Supply Chain
- 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
- TST 202 – Penetration Testing Fundamentals
- TST 205 – Performing Vulnerability Scans
- TST 206 – ASVS Requirements for Developers

Elite

- 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 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

Application Security Champion

The Application Security Champion learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those chartered with driving a culture of "Security Built-in" to the software development lifecycle. The Application Security Champion learning path also explains application security concepts such as privacy, secure development and architecture, security testing, threat modeling, cryptography, and cyber threat analysis and remediation.

37 Courses

18 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

- DSO 302 – Automated Security Testing
- ENG 311 – Attack Surface Analysis and Reduction
- ENG 312 – How to Perform a Security Code Review

Information Security Specialist

The Information Security Specialist learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for protecting systems, defining access privileges, control structures, and resources. The Information Security Specialist learning path helps build the skills required to identify, protect, detect and recover from risks, vulnerabilities, and threats to the security of information and/or data.

71 Courses

29 Labs

26 Hours

31 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
- 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

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
- CYB 210 – Cybersecurity Incident Response
- CYB 211 – Identifying and Protecting Assets Against Ransomware
- CYB 212 – Fundameantals 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 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 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

Elite

- 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
- 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

Systems Leadership

The Systems Leadership learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for computers and their complex operating systems. It also builds the baseline but comprehensive application security knowledge necessary for leading application development and design projects. The Systems Leadership learning path explores application security best practices necessary to ensure strategies and plans support business needs and align with departmental and organizational objectives and goals.

23 **Courses**

7 **Hours**

8 **CPE Credits**

Core

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

Advanced

- CYB 211 – Identifying and Protecting Assets Against Ransomware
- DES 206 – Meeting Cloud Governance and Compliance Requirements
- 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 201 – Fundamentals of Secure DevOps
- DSO 212 – Fundamentals of Zero Trust Security
- TST 206 – ASVS Requirements for Developers

Elite

- 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

Development Manager

The Development Manager's learning path includes a variety of security courses that will vary depending on whether you are seeking core, advanced or elite paths. It is designed for those responsible for planning, preparing, and ensuring that projects are completed. The Development Manager's learning path 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.

23 **Courses**

8 **Hours**

10 **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