Synopsys eLearning

Synopsys offers a hosted eLearning curriculum that enables organizations of all sizes to quickly deploy industry-leading training companywide. Synopsys eLearning is a subscription-based online training service providing on-demand, unlimited access to our comprehensive library of hosted eLearning courses. With an annual subscription, you get 24/7 access to our interactive security courses including knowledge checks and final exams, individual and group reporting, and periodic content updates so you can easily meet compliance and contractual training requirements.

For companies that deploy their own learning management system (LMS), all eLearning courses are SCORM-compliant and can be deployed within your current LMS.

General disclaimer

This document presents details about the training offerings from Synopsys at the time of its creation. Synopsys has used reasonable efforts to ensure that the information provided in this document is accurate and up to date, but details and offerings are subject to change.

This document contains confidential information about Synopsys and its businesses. Copies of this document may only be provided, and disclosure of the information contained in it may only be made, with written prior agreement from Synopsys.

Ownership and disposal
The information contained in this document is owned by Synopsys. The recipient shall dispose of the data as confidential waste and/or return the document to Synopsys upon request.
Table of contents
Courses are available in English, Chinese, and Japanese.

General disclaimer ................................................................. 1

Fundamentals ................................................................. 4
Attack and Defense ................................................................. 5
DevSecOps Security ................................................................. 6
Foundations of Information Security Awareness ..................... 7
Introduction to Cryptography for Architects and Developers .......... 8
OWASP Top 10 ........................................................................ 9
Principles of Software Security ............................................. 11

Defensive Strategies ............................................................. 12
Database Security .................................................................. 13
DevOps for Security Managers .............................................. 14
Hardening Your APIs ............................................................ 15
Introduction to Automotive Security ....................................... 16
Introduction to Securing the Internet of Things ......................... 17
Living with Open Source ....................................................... 18
Open Source Policies and Risks .............................................. 19
Secure Communication for the Internet of Things ...................... 20
Secure Development for Financial Services ................................ 21
Secure Password Storage ....................................................... 22
Threat Modeling ................................................................. 23

Languages and Platforms ..................................................... 24
Building Security into ASP.NET ............................................ 25
C/C++ Security ..................................................................... 27
Defensive Programming for COBOL ...................................... 28
Defensive Programming for HTML5 Security ......................... 29
Defensive Programming for Java EE Web Applications ............... 30
Defensive Programming for PHP Security ................................ 31
Foundations of COBOL Security ............................................ 33
Foundations of .NET Platform Security ................................... 34
Hapi.js Security .................................................................... 35
Introduction to HTML5 Security ............................................ 36
Introduction to PHP Security ................................................ 37
Java Advanced Secure Coding .............................................. 39
Java Security Fundamentals ................................................. 40
Java Spring Security ............................................................ 41
JavaScript Security .............................................................. 42
Modern C++ Security .......................................................... 44
Node.js Security .................................................................... 46
React.js Security ................................................................. 47
Secure Programming for Golang ........................................ 48
Securing AngularJS .............................................................. 49
Securing Express.js ............................................................. 50
Securing Python Web Applications ..................................... 51

Cloud Platforms .................................................................... 52
  Introduction to Cloud Security ........................................... 53
  Secure Implementation of Docker and Kubernetes ............... 54
  Securing Amazon Web Services ......................................... 55
  Securing Google Cloud Platform ....................................... 56
  Securing Microsoft Azure ................................................. 57
  Securing MongoDB .......................................................... 58

Authentication and Authorization ......................................... 59
  Advanced OAuth 2.0 Topics .............................................. 60
  Essential API Security ...................................................... 61
  OAuth 2.0 Security ........................................................... 62
  OpenID Connect .............................................................. 63
  SAML Security ............................................................. 64
  Securely Accessing APIs Using OAuth 2.0 ......................... 66
  Securely Granting Access to an API Using OAuth 2.0 .......... 67

Mobile ............................................................................... 68
  Advanced Android Security ............................................. 69
  Android Security ............................................................ 71
  Foundations of Mobile Security ........................................ 72
  Fundamentals of iOS ........................................................ 73
  Secure Programming for iOS ............................................ 75

Requirements, Architecture, and Training .......................... 77
  Architecture Risk Analysis .............................................. 78
  Risk-Based Security Testing Strategy ................................. 80
  Software Security Requirements ....................................... 82

Regulation and Compliance ............................................... 83
  GDPR for Developers and Architects ................................. 84
  GDPR for Development and Project Managers .................... 86
  GDPR for Upper Management .......................................... 87
  Introduction to CCPA, California Consumer Protection Act .... 88
  Introduction to GDPR ....................................................... 89
  Introduction to PCI SSF ..................................................... 90
  PCI DSS Security ............................................................ 92
  Secure Development for Healthcare ................................... 93
Course Outline

Introduction to Attack and Defense
• Introduction to Attack and Defense
• Understanding Your Enemy
• Vulnerabilities Are Here to Stay
• The Trinity of Trouble
• Impacts of Insecure Software

Data Protection
• NEW: GDPR
• Attacks on Data in Motion
• Securing Data in Motion: System Footprint
• Securing Data in Motion: Encryption
• Securing Data in Motion: Keys and Data
• Attacks on Data at Rest
• Securing Data at Rest
• Crypto Best Practices

Handling User Input
• SQL Injection
• SQL Injection: Examples
• Command Injection
• Cross-Site Scripting
• NEW: XML External Entities (XXE) Attack
• Unvalidated Redirects: Vulnerable?
• Unvalidated Redirects: Spotting and Defense

Authentication and Authorization
• Authentication Attacks
• Defending Against Authentication Attacks
• Authorization Attacks
• Defending Against Authorization Attacks

Session Protection
• Weak Cookie Security
• Attacking Sessions: Hijacking
• Attacking Sessions: Fixation
• Cross-Site Request Forgery
• Protecting Against CSRF
• SameSite Cookie Attribute
• Building Secure Session Mechanisms

Security Configurations
• Attacking Application Configurations
• Insecure by Default
• Securing Third-Party Components
• Changing Defaults
• Preventing Information Leakage

Monitoring and Detection
• Tools of the Trade
• Tools: Firewalls
• Tools: Alerts and Logging
• Tools: Detection and Honeypots

Course Description

Web applications are becoming an increasingly high-value target for hackers looking to make a quick buck, damage reputations, or just boost their "street cred." There is no shortage of publicly known attack tools and techniques, and software developers are outnumbered and at the front line of defense. This course will teach you how attackers discover and exploit vulnerabilities in the real world and how to build a strong line of defense.

Learning Objectives
• Recognize security flaws in web applications
• Build defenses against common web application vulnerabilities
• Use tools and techniques to test your own applications for vulnerabilities
• Implement application features to enhance your users’ security posture

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers

Prerequisites
• Principles of Software Security
• Foundations of Information Security Awareness
**Course Outline**

**Introduction and Key Concepts**
- DevOps
- Why DevOps Works
- DevSecOps
- Continuous Integration, Delivery, and Deployment (CI/CD2)
- Pipelines
- DevSecOps and CI/CD2
- Market Segments
- Case Study

**DevSecOps Drivers and Challenges**
- Business Drivers
- Business Challenges
- Technology Drivers
- Technology Challenges
- Case Study

**DevSecOps and People**
- Changing Responsibilities
- Case Study

**DevSecOps and Process**
- DevSecOps and Processes Enhancement
- Case Study

**DevSecOps and Technology**
- Tools in the Belt
- Choosing the Right Tools
- Case Study

---

**Fundamentals**

**DevSecOps Security**

1 Hour  Beginner

**Course Description**
DevSecOps stands on a foundation of agile ideas and principles, such as frequent delivery, a focus on automation, and speed to deployment. DevSecOps focuses on key principles such as a self-service environment, on-demand provisioning, continuous feedback, customization, and speedy delivery. Covering the fundamentals of DevSecOps and its integral components, this course helps you to understand how a concerted push for secure development agility can deliver value to an organization.

**Learning Objectives**
- Explore the fundamentals of DevSecOps and its integral components
- Understand how DevSecOps integrates security at multiple points in DevOps workflows
- Identify organizational challenges of embedding security and quality in DevSecOps
- Maintain engaged discussions among DevSecOps stakeholders
- Explore the types of tools capable of boosting operational efficiency
- Begin to confidently qualify sales and consulting opportunities efficiently

**Intended Audience**
- Architects
- Enterprise Developers
- Front-End Developers
- Mobile Developers

**Prerequisites**
- None
Course Outline

Introduction to Information Security Awareness
• What Exactly is Security Awareness?
• Identifying and Understanding Assets
• Boundaries Between Work and Home

Workstation Security
• Overview of Workstation Security
• Five Steps to Physical Security
• Network Connections
• Malicious Software
• Data Protection
• Defense Mechanisms: Barriers and Updates
• Defense Mechanisms: Files and Storage
• Know That You Have Been Hacked

User Account Security
• Introduction: Accounts Rule the Web
• Securing Accounts: What Can Go Wrong?
• Securing Accounts: Password Security Practices
• Securing Accounts: Password Managers
• Key Considerations When Choosing a Password Manager
• Multifactor Authentication

Mobile Device Security
• Introduction: Mobile Devices in the Workplace
• Physical Device Security
• Physical Device Security: A Story
• Mobile Device Security: Basics
• Mobile Device Security: Applications
• Tinkering: What to Know

Social Engineering
• Introduction: People as a Target
• Physical Social Engineering: What to look for
• Physical Social Engineering: Common attack Techniques
• Email Phishing Attacks
• Email Phishing Attacks: Tips to Avoid Getting Inbox-Duped
• Text Message Phishing
• Voice/Phone Phishing
• Use Social Media Safely
• Phishing in the Real World

Working From Home
• Introduction
• Home Network Security
• Workstation and Devices Security
• Video Conferencing

Fundamentals

Foundations of Information Security Awareness

1 Hour Beginner

Course Description
Security awareness is a process of constant refinement and education. Every person has a key role in keeping their company secure and out of the headlines. This course walks through what it takes to effectively identify and act upon security risks in your personal and work lives. It covers a broad range of modern security topics and provides actionable advice.

Learning Objectives
• Quickly identify potential common security risks in the workplace
• Assess the security of their workstations, mobile devices, and office spaces
• Build a strong password creation and storage mechanism
• Recognize the implications of real-world data breaches
• Identify corporate information assets and understand how to handle them securely

Intended Audience
• Architects
• Back-End Developers
• Front-End Developers
• Enterprise Developers
• Mobile Developers
• QA Engineers

Prerequisites
• None
Course Outline

Cryptography Foundations
• What Is Cryptography?
• Terminology and Core Concepts
• What Is Cryptography Used For?
• Real-World Cryptography
• Primitives

Encryption
• Common Encryption Primitives
• Symmetric Key Encryption
• Asymmetric Key Encryption
• Common Encryption Ciphers
• Initialization Vectors
• Padding
• Using Asymmetric Encryption Securely

Cryptographic Hash Functions
• Common Hash Function Algorithms
• Password Storage
• Data Integrity
• Lifeboat’s Sinking Ship

Message Authentication Codes (MACs)
• MACs vs. Hashes
• How MACs Work
• Examples
• Best Practices

Digital Signatures
• How Digital Signatures Work
• Implementations
• Putting It Together with TLS

Building a Secure Cryptosystem
• Questions to Consider
• Architecting Your Cryptosystem
• Key Management
• Choosing Protocols and Algorithms
• Cryptographic Life Cycle
• Case Study
• Key Takeaways

Knowledge Check

Fundamentals

Introduction to Cryptography for Architects and Developers

1 Hour  Beginner

Course Description
Cryptography: It’s what makes it possible to safely bank, shop, and work on the internet. Cryptography helps store passwords securely and even helps prevent data loss if you lose your device. This course will teach you the cryptography basics you need to know to design and develop secure applications and systems.

Learning Objectives
• Describe common cryptographic implementations
• Make security-conscious decisions around cryptography
• Understand real-world threats to cryptography
• Know when and how to use specific cryptographic primitives

Intended Audience
• Architects
• Security Engineers
• Software Developers
• Systems Administrators

Prerequisites
• Principles of Software Security
• OWASP Top 10
• Attack and Defense
Course Outline

Introduction
• Introduction to the OWASP Top 10

Broken Access Control
• Access Control Introduction
• Function-Level Access Control Introduction
• Strategies
• Insecure Direct Object References: In a Nutshell
• Giving Your Friends Admin Access to Any Business Page on Facebook

Cryptographic Failures
• Introduction
• Handling Sensitive Data Securely
• Real-Life Cryptography Failures

Injection
• Introduction
• SQL Injection
• Command Injection
• Cross-Site Scripting
• Injection Attacks Are Still an Issue

Insecure Design
• Introduction
• Security Requirements
• Secure Design
• Threat Modeling
• Example: Meltdown and Spectre
• Conclusions

Security Misconfiguration
• Introduction
• Protection
• Accidental Leaks

Vulnerable and Outdated Components
• Introduction
• Securing Third-Party Software Components
• Quick Question
• An Upstream Bug

Identification and Authentication Failures
• Introduction
• Authentication Overview
• Session Security Overview
• Session Security Considerations
• Authentication Security
• Zoom Authentication Issues
• Authentication Solutions: Build Versus Buy

Continued on next page
Fundamentals

OWASP Top 10
(cont.)

Software and Data Integrity Failures
- Introduction
- Software Integrity
- Insecure Deserialization
- Security in the Software Supply Chain

Security Logging and Monitoring Failures
- Introduction
- Insufficient Logging and Monitoring
- Logging and Monitoring Best Practices
- Logging Technologies
- Security Logging Interfaces

Server-Side Request Forgery (SSRF)
- Introduction
- Capital One Attack
- Technical Vulnerability
- Attacks and Exploits
- Defense
- Conclusion
Course Outline

Basic Software Security Concepts
• The Importance of Software Security
• Software Security Vocabulary
• What Is Secure Software?
• Obstacles to Software Security
• Building Security In
• Roles in Software Security

Fundamentals of a Software Security Initiative
• Goals of a Software Security Initiative
• Engineering and Governance
• Software Security Group (SSG), Outreach, and Satellites
• Vendor Management
• Evolution of a Software Security Initiative

Software Development Life Cycle (SDLC)
• The Touchpoints
• Secure Software Development Life Cycle
• Software Security Intelligence
• Technical Standards and Reference Frameworks
• Training

Assessing Software and Code Review
• The Criticality of Assessing Software
• ARA / Threat Modeling
• Manual Code Review
• Static Analysis
• Dynamic Analysis
• Fuzz Testing
• Risk-Based Security Testing
• Penetration Testing
• Interactive Application Security Testing (IAST)
• Discovery Method Pros and Cons
• The Importance of Fixing Software

Principles of Software Security

¾ Hour  Beginner

Course Description
Dive into the basics of software security inside the development process. This course introduces the fundamentals of software security problems, risks, and general approaches for producing better software. It also presents an approach to building software security into the development process, further enabling developers to deliver better software. This course was created by the experts who wrote the book on software security. The approaches described in this course are used by leading global companies with mature software security initiatives.

Learning Objectives
• Discuss basic security terminology comfortably when discussing your own development work
• Confidently contribute to discussions of software security principles
• Participate in the initial strategy, formation, and role delegation of a software security initiative
• Confidently begin to contribute to your company’s overall design of a software security strategy

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers
• Mobile Developers
• QA Engineers

Prerequisites
• None
Defensive Strategies
Course Outline

Introduction to Database Security
- Types of Databases
- Quick Questions
- Security Challenges and Risks
- Steps to Secure a Database
- Quick Questions

Governance and Compliance
- Data Inventory
- Data Protection Policy
- Data Protection Policy Contents
- Quick Questions
- Privacy and Compliance

Authentication and Access Control
- Authentication and Access Control Risks
- Authentication
- Access Control
- Best Practices

Encryption
- Transparent Data Encryption
- Other Encryption Methods
- Key Management
- Other Data Protection Mechanisms
- Choosing the Right Encryption Mechanism

Operation and Monitoring
- Configuration and Hardening
- Securing the Database Ecosystem
- Audit Trails
- Native Database Auditing
- Database Monitoring: Host Agents
- Database Monitoring: Network Monitoring and DAM
- Best Practices

Summary
- Protecting Your Most Valuable Asset: Information
- Authentication and Access Control
- Database Encryption
- Hardening
- Monitoring
- Securing Data in the Cloud

Defensive Strategies

Database Security

1 Hour Beginner

Course Description
Databases have been growing both in size and complexity during the past few years. At the same time, new legal and compliance requirements have been introduced to protect personal and sensitive data.

This course will guide you through the steps to secure databases. It provides an overview of the current threat and compliance landscape for databases and highlights the importance of defining data inventory and data protection policy. Learners will get a thorough understanding of the best practices for database authentication and access control as well as the available database encryption methods. The course also provides guidelines on hardening databases and describes methods of auditing and monitoring.

Learning Objectives
- Understand the steps required to secure a database
- Create a data protection policy
- Follow best practices to control access to databases
- Use the most efficient method for encrypting data in the databases
- Harden databases according to best practices
- Enable auditing and monitoring of databases
- Compare different tools for managing third-party dependencies

Intended Audience
- Architects
- QA Engineers

Prerequisites
- None
Course Outline

Introduction to History
• History and Evolution
• Underlying Drivers: Time-to-Market and Self-Service Software Delivery

Software Life Cycle and Challenges
• How Engineering-Led Initiatives See Their Life Cycle
• Challenges Security Initiatives Face
• Maturing Firms May Have It Harder
• Challenge 1: Cadence
• Challenge 2: Gating the SDL
• Challenge 3: Building Using a Federated Workforce
• Challenge 4: Automation: Reducing Reliance on Human (Manual) Effort
• Challenge 5: Inventory

Achieving Security Governance: Inventorying Software, Conducting Defect Discovery
• Introduction and Pillars
• Inventorying Assets, Scope
• Discovering Inventory
• Automating Practice Areas
• Defect Discovery
• DevOps Defect Discovery

Security Governance: Process Remediation Workflow, and Gating
• Alternatives to Gates
• Remediation Enablement
• Accountability
• Continuous Telemetry, Accompanying Continuous Delivery

Measurement
• DORA
• Security, a Subset of Quality
• Sample Security Measures

Defensive Strategies

DevOps for Security Managers

1 ½ Hours  Beginner

Course Description
This course is geared toward security executives who own a software security group or are sponsoring a security initiative organizationwide. It covers how to align legacy and net-new necessary security tools and activities with a DevOps culture and its underlying delivery technologies, conventions, and culture. This course prepares security executives to gain and maintain a “seat at the table” with development leadership by providing enough “what” and “how” regarding sw-defined security governance to productively participate—and even drive—sw-defined security governance, or as some say, “the Sec in DevSecOps” that ultimately secures DX.

Learning Objectives
• Understanding the DevOps movement for organizations and how cloud technology adoption greatly enabled and accelerated CI/CD toolchain and DevOps culture adoption
• Identify and address common challenges in converting legacy controls to the new culture
• Plan a culturally compatible approach to addressing these challenges
• Participate in existing engineering measurement practices, adding security metrics

Intended Audience
• DevOps Practitioners
• Managers
• Security Practitioners

Prerequisites
• DevSecOps
• OWASP Top 10
Course Outline

Introduction
• Introduction
• API Security and the OWASP Top 10
• Securing Your APIs

The Client's Role in Security
• Introduction
• Enforcing Restrictions on the Client
• The Real Attack Surface of Your APIs
• Using the Client for Better Security
• Conclusion

Using Rate Limiting and Abuse Protection
• Introduction
• The Need for Enforcing Limits
• Overview of Defensive Strategies
• Implementing Rate Limiting in an API Gateway
• Conclusion

Mitigating Server-Side Request Forgery (SSRF)
• Introduction
• Introducing SSRF
• The Consequences of SSRF
• Restricting IP Addresses
• Restricting Domains
• Restricting URLs
• Additional Guidelines
• Defense-in-Depth Strategies
• Conclusion

Deploying CORS for APIs
• Introduction
• A Brief Introduction to CORS
• The Relevance of CORS to APIs
• CORS for Public Endpoints
• CORS for APIs Using Cookies
• CORS for APIs Using Authorization Headers
• Conclusion

Configuring Security Headers for APIs
• Introduction
• Overview of Security-Related Response Headers
• Forcing the Use of HTTPS
• Restricting Framing
• Dealing with Content Sniffing
• Restricting Undesired Behavior
• Using the Fetch Metadata Headers
• Conclusion

Conclusion
• Security and the Client
• Rate Limiting and Abuse Protection
• Server-Side Request Forgery
• Cross-Origin Resource Sharing
• Security Headers
• Wrap Up

Defensive Strategies

Hardening Your APIs

1 ¼ Hours  Intermediate

Course Description

APIS are an attractive target for attackers. It is crucial to harden your APIs to prevent bad guys from gaining a foothold. In this course, we investigate how attackers deploy offensive techniques to attack your APIs, and how you can apply defensive security techniques to counter these attacks. By course end, you will have a list of best practices to increase the security of your APIs.

Learning Objectives

• Identify areas with potential weaknesses in APIs
• Explain best practices to secure APIs
• Formulate defense-in-depth strategies to secure APIs

Intended Audience

• Architects
• Back-End Developers
• Enterprise Developers

Prerequisites

• Essential API Security
Introduction to Automotive Security

Course Description
The Introduction to Automotive Security course is designed to teach the basics of secure automotive system design. The course considers automotive security from two perspectives: (1) an internet-connected mobile systems platform and (2) extravehicular support systems such as web data services. Learners are introduced to security concepts and secure design processes for automotive applications. The course applies to technology professionals in the automotive industry who participate in the development of automotive systems.

Learning Objectives
- Understand basic security principles for securing vehicular and extravehicular information systems
- Get an overview of a few industry standards for securing vehicle systems
- Learn where to go for additional information

Intended Audience
- Architects
- Back-End Developers
- Developers
- Enterprise Developers
- Mobile Developers
- QA Engineers

Prerequisites
- Java Security Fundamentals
- Principles of Software Security
Course Outline

Introduction to the Internet of Things
- What Is IoT?
- Consumer IoT
- Medical IoT
- Smart City

IoT Architecture
- IoT Device(s)
- Communication
- IoT Supporting Services
- Integration with Third-Party Services
- APIs for Third-Party Integrations
- Web Client
- Mobile Client
- Engineering Challenges of IoT

IoT Protocols
- Types of Communication
- Short-Range Communications
- Long-Range Communications
- Application Data Protocols

IoT Threat Modeling
- Threat Modeling Review
- STRIDE
- Threat Countermeasures
- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privilege

IoT Security
- Open Web Application Security Project (OWASP) Top 10 for IoT
- Weak, Guessable, or Hardcoded Passwords
- Insecure Network Services
- Insecure Ecosystem Interfaces
- Lack of Secure Update Mechanism
- Use of Insecure or Outdated Components
- Insufficient Privacy Protections
- Insecure Data Transfer and Storage
- Lack of Device Management
- Insecure Default Settings
- Lack of Physical Hardening
- Case Study: Mirai Botnet

IoT Regulation Concerns
- Disclaimer
- General Data Protection Regulation (GDPR)
- California Consumer Privacy Act (CCPA)
- IoT Device Security Act (SB-327)
- Payment Card Industry Data Security Standard (PCI DSS)

Defensive Strategies

Introduction to Securing the Internet of Things

1 Hour  Beginner

Course Description
This course introduces the complexities of the Internet of Things (IoT) and the security issues that plague IoT systems. The core focus of the course is IoT security. We'll cover the unique and manifold features of IoT and how they relate to security and privacy. At the end, learners will know enough about the security pitfalls of IoT systems to make informed decisions, whether as IoT vendors creating products or as enterprise or personal consumers making choices about what devices to deploy.

Learning Objectives
- Understand the top security issues plaguing IoT
- Perform threat modeling on an IoT product design
- Have a clear grasp of the regulatory concerns for IoT

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Mobile Developers
- QA Engineers

Prerequisites
- None
Course Outline

Benefits of Using Open Source Software
- Key Benefits of Using Open Source Software
- Open Source License Types
- Growth in Available Open Source Components
- Growth in Number of Components in Codebases
- Benefits

Risks of Using Open Source Software
- Three Types of Risk
- License Risk
- Survey Results
- Source Code Risk
- Operational Risk
- Conclusion

Deep Dive on Open Source License Risk
- License Risk
- License Conflicts
- Types of Usage
- Distribution Model
- Real-World License Risk
- Conclusion

Deep Dive on Open Source Security Risks
- Source Code Vulnerabilities
- Survey Results
- Malicious Source Code
- Scanning for Vulnerabilities
- Public Vulnerabilities
- Transitive and Direct Impacts
- Conclusion

Deep Dive on Open Source Operational Risks
- Community Development
- Analyzing Operational Risk by Component

Bringing It All Together
- Summary
- Managing Risk
- Conclusion

Defensive Strategies

Living with Open Source

1 ¼ Hours  Beginner

Course Description
In this course, you’ll learn how to evaluate and manage risks when using open source software. You’ll walk away with an understanding of the three types of risk that open source brings to your projects and how to create a plan for managing each type.

Learning Objectives
- Weigh the benefits of using open source vs closed source
- List the three types of risks associated with open source
- Evaluate open source license risk
- Calculate security risk levels for source code
- Plan for managing operational risk when using open source

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Management
- Mobile Developers

Prerequisites
- None
Course Outline

Introduction
- Introduction
- Benefits of Using Open Source
- Risks
- Impact from Noncompliance with Licenses

Open Source Licenses
- Open Source License Categories
- GNU GPL
- GPL Variants
- Apache
- MIT
- BSD
- Mozilla Public License
- Quick Question

Using Open Source Licenses Correctly
- Overview
- How to Use GPL Licenses Correctly
- Steps to GPL Compliance
- Using Other Open Source Licenses
- Components with Multiple Licenses
- Relicensing
- Dual Licensing

Security Risks of Open Source Software
- Vulnerabilities and Open Source Software
- Security Risks
- Minimizing Security Risks in Open Source Software
- Finding and Patching Vulnerabilities
- Software Composition Analysis Tools
- Choosing a SSCA tool

Building and Open Source Policy
- Governance and Strategy
- Putting Together a Team
- Essential Components
- Sourcing and Selection
- Support and Maintenance
- Contributions
- Creating a New Open Source Project
- Approval Process
- Auditing

Summary
- Governance and Management of Open Source
- Handling License and Compliance Risks
- Handling Security Risks
- Conclusion: To Use or Not to Use open Source Software

Defensive Strategies

Open Source Policies and Risks

1 Hour  
Beginner

Course Description

The use of open source code in modern applications is constantly rising. Open source software brings numerous advantages, including speeding development and minimizing costs. But there are also significant risks to using open source in terms of security vulnerabilities, licensing, and compliance.

In this course, we present the most widely used open source licenses and explain the obligations they entail for users. We explain how these licenses can be used correctly and how to avoid license conflicts. We also analyze the security risks of open source software and how vulnerabilities can be found and dealt with. Finally, we analyze the steps that need to be taken to build a corporate open source policy that will govern the use of open source software throughout the organization. Building an open source policy is essential in order to minimize both license and security risks.

Learning Objectives

- Understand widely used open source licenses
- Understand the obligations they bring
- Understand how to use the most well-known open source licenses
- Understand the security risks of using open source code and how they can be mitigated
- Build a corporate policy on the use of open source software in an organization

Intended Audience

- Architect
- Back-End Developer
- Front-End Developer
- Enterprise Developer
- Mobile Developer
- QA Engineer

Prerequisites

- Living with Open Source
Course Outline

IoT Review
- What Is IoT?
- IoT Architecture
- IoT Communications
- IoT OWASP Top 10

IoT Communications Security
- IoT Communications
- IoT Application Protocols Security
- IoT Communication Cryptography
- Communication Vulnerability Example: Bluetooth Low Energy

IoT Firmware Security
- Firmware Basics
- Quick Question
- Extracting the Firmware
- Filesystem Vulnerabilities
- Binary Vulnerabilities
- Firmware Updating Vulnerabilities

IoT Authentication
- IoT Authentication Overview
- IoT User Authentication
- IoT Machine-to-Machine Authentication

IoT Mobile and Web Application Security
- IoT and Mobile/Web Interfaces
- Mobile App Security Revisited
- Web App Security Revisited

Defensive Strategies

Secure Communication for the Internet of Things

1 Hour  Beginner

Course Description
This course digs deeper into the facets of the Internet of Things (IoT) most plagued by security issues. Students learn the techniques employed by real-world attackers to undermine the security of IoT devices and their surrounding ecosystems. Upon completion students will have a solid understanding of the decisions facing an IoT vendor creating a secure IoT product, as well as the knowledge of how to assess IoT product security as an enterprise or personal consumer.

Learning Objectives
- Understand the top attacks against IoT devices
- Perform high-level security assessments on IoT devices
- Have a clear grasp of best practice mitigation strategies

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Mobile Developers
- QA Engineers

Prerequisites
- Introduction to Securing the Internet of Things
Course Outline

Challenges of the Modern Regulatory Environment
- Current State of Compliance for Financial Services
- Challenges and Concerns
- Key Concerns for Software Development

Analysis of the Regulatory Landscape for Financial Services
- Introduction to Compliance for Financial Services
- Comparison of Regulatory Frameworks
- Areas of Focus
- Requirements for the Software Development Life Cycle

Authentication, Authorization, and Access Control
- Requirements for Authentication
- Password Policies
- Authorization, Access Control, and Monitoring
- Session Management

Data Privacy
- Personal Data Protection: A (Not So) New Requirement
- Categories and Definitions of Personal Data
- Compliance Requirements for Privacy: GDPR
- CCPA: What It Is and Definitions
- CCPA: Business and Technical Requirements
- GLBA Privacy Rule

Data Encryption and Integrity
- Encrypting Data at Rest
- Transport Layer Security and Encryption of Data in Motion
- Protecting Data Integrity

Operational Security and Monitoring
- Securing the Infrastructure
- Testing Requirements
- Keeping Logs and Auditing
- Breach Detection and Notification

Secure Development for Financial Services

1 Hour  Beginner

Course Description
This course presents the current compliance landscape for building applications for the financial services sector. The regulatory landscape for financial services is extremely complex, with a lot of overlapping requirements. The goal of this course is to analyze this complex environment and outline the consolidated requirements posed by various legal and compliance frameworks. The course focuses on requirements for authentication, authorization, and access control. It also presents requirements for privacy and personal data protection brought about by recent legislation such as GDPR and CCPA. The course concludes with operational security requirements as well as what is mandated for breach detection and notification.

Learning Objectives
- Understand the main legal and compliance frameworks for financial services and what requirements they bring to developing applications
- Implement compliant policies for authentication, authorization, and access control
- Identify personal data and be able to protect that data based on the privacy requirements imposed by the corresponding legal frameworks
- Apply necessary encryption and integrity protection mechanisms for sensitive data
- Outline the controls required to secure applications and monitor for potential security breaches

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Management
- Mobile Developers
- QA Engineers

Prerequisites
- None
Course Outline

The Importance of Password Storage
- Introduction
- The Security Properties of Passwords
- The Security Properties of Passwords
- Quick Question
- Data Breaches and Passwords
- Have I Been p0wned?
- Wrap Up

Why Is Password Storage So Difficult?
- Introduction
- The Insanity of Plaintext Passwords
- Common Password Storage Mechanisms
- Wrap Up

Storing Passwords Using One-Way Functions
- Introduction
- The Properties of Cryptographic Hash Functions
- Output Length
- Example Hash Functions and Attacks Against Them
- Case Study
- Wrap Up

Salting Stored Passwords
- Introduction
- Salting a Password
- Example Code
- Salt Effectiveness Against Attack
- Length Considerations
- HMACs

Storing Passwords with an Adaptive One-Way Functions
- Introduction
- Properties of Adaptive One-Way Functions
- Proof of Work
- Adaptive Property and Tuning
- Choosing an Adaptive One-Way Function
- Example
- Case Study

Storing Passwords with Encryption
- Introduction
- The Properties of Encryption
- Password Encryption Practice
- Passwords Encryption: Code Examples
- Security Properties and Weaknesses
- Encryption’s Main Advantage
- Case Study
- Wrap Up

Upgrading Existing Mechanisms
- Introduction
- Rolling Upgrade
- Layered Upgrade
- Wrap Up

Conclusion
- Introduction
- Normalizing Your inputs
- Salting Passwords
- Using and Adaptive One-Way Function
- Use of a Pepper
- Upgrading or Rotating Storage/Verification Schemes

Secure Password Storage

1 ½ Hours
Intermediate

Course Description
Developers have been storing passwords for ages. But did you know that the best practices from 10 years ago are hopelessly outdated? In this course, we look at commonly used but weak password storage mechanisms. Gradually, we work toward the current best practice for storing passwords. A real-life case study shows how to build a layered approach to storing passwords securely.

Learning Objectives
- Describe the weaknesses for password-based authentication systems
- Analyze the security of an existing password storage mechanism
- Implement a password storage mechanism using current best practices
- Devise a strategy to upgrade an existing legacy and insecure password storage mechanism

Intended Audience
- Architect
- Back-End Developer
- Enterprise Developer

Prerequisites
- Introduction to Cryptography for Developers and Architects

Synopsys eLearning Catalog | March 2021
Course Outline

What Is Threat Modeling?
- The Goal of Threat Modeling
- A Pragmatic Approach to Threat Modeling
- Software Defects vs. Bugs vs. Flaws

Why Should You Threat Model?
- Finding Security Problems Early
- Focusing on Flaws, Not Bugs
- Identifying Critical Components in the Architecture
- Thinking Beyond Canned Attacks
- Aligning Business Requirements and Security Requirements

Threat Modeling, in a Nutshell
- Strategies for Threat Modeling
- Basing Threat Modeling on a ... Model
- Architecture-Agnostic Threat Modeling

Common Threat Modeling Methodologies
- The Big Picture
- Microsoft’s Threat Modeling Using STRIDE
- Applying Architectural Risk Analysis (ARA)
- Alternative Approaches

STRIDE
- Data Flow Diagrams
- The STRIDE Keywords
- Finding Threats with STRIDE
- Making STRIDE More Concrete

Architectural Risk Analysis (ARA)
- ARA, in a Nutshell
- Business Context and System Decomposition
- Attack Resistance Analysis
- Underlying Framework Weakness Analysis
- Ambiguity Analysis

Fitting Threat Modeling into the SDLC
- Threat Modeling Throughout the SDLC
- Responsibilities and Actors
- Managing Risk
- Tracing Risk Throughout the SDLC

Conclusion
- Threat Modeling Methodologies
- Threat Modeling in the SDLC

Threat Modeling

1 Hour Intermediate

Course Description
It’s a fact: Malicious attackers are coming for your data, and there are many potential points of entry that you should address. Unfortunately, not every fix is obvious, simple, or even effective. Threat modeling helps to put the abundance of threats in order and contextualize security through the lens of the attack. This course explains how threat modeling helps you think about security in a structured way, covering common threat modeling methodologies and how to handle discovered threats.

Learning Objectives
- Understand the purpose of threat modeling and its relation to other security activities
- Understand how threat modeling fits into Microsoft’s SDL
- Understand how threat modeling fits into the Synopsys Touchpoints methodology
- Understand the process of threat modeling and risk management
- Get started with threat modeling

Intended Audience
- Everyone

Prerequisites
- None
Languages and Platforms
Course Outline

Handling Input Security
• Welcome to Building Security into ASP.NET
• Don't Trust the Client
• General Input Validation
• ASP.NET Request Validation
• Validator Controls
• Razor's ValidationHelper
• MVC Model Binding and Validation
• Cross-Site Scripting
• Output Encoding
• Template Injection
• Injections
• Open Redirect Attacks
• Cross-Origin Request Sharing
• Deserializing Objects

Dealing with Files Securely
• Defending the File System
• Directory Traversal
• Defending Against Directory Traversal
• Local File Inclusion
• Defending Against a Local File Inclusion Attack
• The Dangers of File Uploads
• Attack Review
• Defending Against File Upload Attacks
• Windows-Specific Issues
• Linux-Specific Issues

Identity
• Introduction to Access Control
• Identity Management in ASP.NET
• ASP.NET Core Identity
• Azure Active Directory
• ASP.NET 4 Membership

Authentication
• What Is Authentication?
• Two-Factor Authentication
• Certificate Authentication
• External Authentication Providers
• Authentication in ASP.NET Core
• Authentication in Azure App Service
• Authentication in ASP.NET 4 MVC
• Authentication in ASP.NET Web Forms

Course Description

Building Security into ASP.NET

2 Hours     Intermediate

ASP.NET is the platform of choice for .NET developers. The security built into the framework has come a long way in 15 years, but there are still some areas that require the developer to remain vigilant in guarding their application from attackers. Learn the ins and outs of identity management, data protection best practices, attack prevention techniques, and other security topics as they apply to .NET.

Learning Objectives
• Determine what features of ASP.NET already meet their security requirements
• Understand how to develop secure applications on top of ASP.NET, and how to safely use the various editions of the framework

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers

Prerequisites
• None

Continued on next page
Languages and Platforms

Building Security into ASP.NET (cont.)

Authorization
- Authorization
- Role-Based Authorization
- Claim-Based Authorization
- Policy-Based Authorization
- Resource-Based Authorization
- View-Based Authorization
- Razor Authorization Conventions

Session Management
- Sessions in ASP.NET
- Handling Session Cookies
- Shared SSO Cookies
- Cross-Site Request Forgery
- SameSite Cookies
- Anti CSRF in ASP.NET Core and Razor
- Anti CSRF in ASP.NET 4 Web Forms

Data Protection in ASP.NET
- Using Cryptography in ASP.NET
- ASP.NET Core DataProtection
- Configuring DataProtection
- Time Limitations
- Password Protection in ASP.NET
- Using ASP.NET Core DataProtection in ASP.NET 4

Exceptions and Logging
- Exception Management and Logging
- Handling Exceptions
- Bad Exception-Handling Practice
- Managed Exceptions in ASP.NET Core
- Logging
- Logging Best Practice
- Logging in ASP.NET Core
- Monitoring with Application Insights
- Information Leakage
- Debugging
- Debugging: Spot the Problem

Configuration Management
- Why Is Configuration Important?
- Environment Settings
- Web Servers and Hosting
- Hosting with Kestrel Web Server
- Hosting with HTTP.sys
- Hosting with IIS HTTP Server
- HTTPS
- Security Headers
- Keeping It Updated

Side-Client Integration
- API Security with SPAs
- Authentication with SPA
- Authorization with SPA
- Securing Blazor WebAssembly Apps
- Working with SignalR

Conclusion
- Where We've Been So Far
- A Summary of All You've Learned
- Where to Go to Learn More
Course Outline

Introduction
• Brief History of C/C
• Problems Facing C
• Legacy Code
• Undefined Behavior

String Handling
• Introduction
• Representation of Strings
• Improperly Bounded String Copies
• Off by One Errors
• Null Termination Errors
• Truncation Issues

Memory Management
• Introduction
• Initialization Issues
• Failing to Check Return Values
• Writing to Freed Memory
• Dereferencing NULL Pointers
• Double Free
• Memory Leaks
• Zero Length Allocations
• C++ Memory Management

Integers
• Introduction
• Wraparound Issues
• Truncation Errors

Format String Attacks
• Introduction
• Crashing Programs
• Reading From the Stack
• Reading From Arbitrary Memory Addresses
• Buffer Overflows
• Writing to Arbitrary Memory Addresses

Concurrency
• Introduction
• Race Conditions
• Value Corruption
• Volatile Objects
• Deadlock

File I/O
• Introduction
• Access Control Overview
• Access Control: Elevated Privileges
• Access Control Example: Elevating Privileges
• Temporarily Dropping Privileges
• Permanently Dropping Privileges
• Directory Traversal
• Time of Check Time of Use (TOCTOU)

Languages and Platforms

C/C++ Security

1 ¼ Hours  Advanced

Course Description
Writing secure code in C/C++ is far from trivial. This course introduces the complexity of working with the C/C++ family of languages, especially from a security perspective. Learn about major security flaws that can lead to insecure programs and how to combat them. Lesson topics include string handling, memory management, integer overflow and wrapping, format string attacks, and more.

Learning Objectives
• Identify use cases where C/C++ is widely used
• Apply new best practices for safely manipulating strings
• Identify unsafe memory handling practices
• Apply mitigation techniques to common integer mishandling
• Understand issues with concurrency and parallelism
• Describe best practices for access controls

Intended Audience
• Architects
• Application Security Specialists
• Code Auditors
• Developers
• QA Engineers

Prerequisites
• Principles of Software Security
• OWASP Top 10
Course Outline

Introduction
• Course Overview
• COBOL Defensive Programming Techniques

Confidential Information Handling
• Identifying Confidential Information
• PII
• PHI
• Other Types of Sensitive Data
• Sources of Information Leakage

Authentication and Authorization
• Evidence of Authentication
• Multifactor Authentication
• Authentication Best Practices
• Authorization Best Practices
• What to Do About 8-Character Passwords

Secure Memory Handling
• Clearing Sensitive Information from Memory
• Pointers and Bounds Checking
• Dynamic Memory Allocation

Cryptography
• Cryptographic Libraries/Services
• Why You Should Never Use DES or 3DES
• Choosing a Key Length
• Generating Keys
• Securely Storing Keys
• Key Rotation
• Using HMACs to Verify Data Integrity

Common Vulnerabilities
• SQL Injection
• Code Injection
• XML Injection
• Command Injection
• Truncation Errors
• Log Injection

Time and State Issues
• Race Conditions: Time-of-Check to Time-of-Use
• Race Conditions: Deadlocks
• Race Conditions: Multithreaded Code

Secure Modern Integrations
• Continuous Integration and Continuous Delivery
• Serverless Functions
• Leveraging Cloud Security Features
• Architecting Resiliency for Dependency Downtime

Languages and Platforms

Defensive Programming for COBOL

1 ½ Hours  Intermediate

Course Description
Building on the Foundations of COBOL Security course, this course explores specific defensive programming techniques and best practices for creating secure COBOL programs. The goal is to teach developer behaviors that follow the principle of defense-in-depth and help prevent COBOL programs from being the weakest link in the enterprise security chain. Techniques covered include COBOL-specific methods for input validation, secure database interactions, secure error handling, and proper resource synchronization.

Learning Objectives
• Thoroughly understand the guiding principles of secure design
• Apply best practice COBOL defensive programming techniques
• Thoroughly understand the software security touchpoints for COBOL programs

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• QA Engineers

Prerequisites
• Principles of Software Security
• OWASP Top 10 or Attack and Defense
• Foundations of COBOL Security
Course Outline

Risk Landscape
• Modern Browsers
• Modern Web Applications
• The Attack Surface
• The Reincarnation of Cross-Site Scripting

The Security Model of the Web
• Same-Origin Policy
• Same-Origin Policy Example
• Sidestepping the SOP with Clickjacking
• JavaScript Execution
• Subresource Integrity

Client-Side Data Storage
• Risks of Client-Side Data Storage
• Security Considerations
• Clearing Data

Web Messaging
• Sending Messages
• Receiving Messages
• Overview of Best Practices

Iframe Sandboxing
• The Sandbox Attribute
• Relaxing the Sandbox
• Sandboxing Use Cases
• Consequences of the Sandbox

Content Security Policy (CSP)
• Defining a CSP Policy
• Limiting Cross-Site Scripting with CSP
• Deploying a Strong CSP Policy
• CSP Beyond XSS
• Additional CSP Features
• Overview of Best Practices

Understanding Cross-Origin Resource Sharing
• Simple CORS Requests
• Authenticated CORS Requests
• Understanding the Threat Model of CORS
• Preflighted CORS Requests
• Handling Headers with CORS
• CORS Beyond JavaScript

Securing Web Applications with CORS
• Special CORS Cases
• Building a CORS Policy
• CORS and CSRF
• Testing Your CORS policy

Additional HTML5 Features
• Permission-Based APIs
• Web Workers
• WebSockets

Conclusion
• Controlling JavaScript
• Compartmentalization

Languages and Platforms

Defensive Programming for HTML5 Security

1 ½ Hours  Intermediate

Course Description
HTML5 is the fifth revision of the HTML standard. HTML5 and its integration with JavaScript introduce new security risks that you must consider carefully when writing web front-end code. Modern web-based software, including mobile web front-end applications, make heavy use of innovative JavaScript and HTML5 browser support to deliver advanced user experiences. Front-end developers focus their efforts on creating these experiences and are generally not aware of the security implications of the technologies they use.

This course helps web front-end developers understand the risks involved with manipulating the HTML Document Object Model (DOM) and using the advanced features of JavaScript and HTML5 (e.g., cross-origin resource sharing, local storage, and content security policy). In this course, we’ll reinforce some important security aspects of modern browser architecture and present defensive programming techniques that developers can apply immediately to avoid introducing common vulnerabilities. We’ll also provide a detailed description of typical JavaScript sources and sinks and explain how to use them to detect problems in code.

Learning Objectives
• Recognize that client-side code can introduce security vulnerabilities
• Appreciate the HTML5 risk landscape
• Apply defensive programming techniques in HTML5
• Identify and fix security vulnerabilities in your HTML5 code

Intended Audience
• Front-End Developers
• Mobile Developers

Prerequisites
• OWASP Top 10
• Introduction to HTML5 Security
• Architecture Risk Analysis
Course Outline

Introduction, Overview
- Introduction
- Java EE Architecture
- Container Security
- Transport Security
- Messaging Security
- Java Enterprise Nonweb Risks

Container Security
- Introduction
- Encoding Reserved Control Sequences Within Untrusted Input
- Data Validation
- Container Authentication and Authorization
- Data Sources
- Session Management
- Bean Validation API

Transport Security
- Introduction
- Secure Transport Techniques
- Common Security Concerns

Messaging Security
- Introduction
- Generating/Processing API Service Tokens
- Securing SOAP/XML Messages
- Common Security Concerns
- Equifax 2017 Incident

Defensive Programming for Java EE Web Applications

¾ Hour Advanced

Course Description

Defensive Programming for Java EE Web Applications picks up where Java Security Fundamentals and Java Advanced Secure Coding leave off. This course covers important features and information for Java EE application designers. You’ll learn about common security concerns across the enterprise, as well as methods and features to strengthen enterprise solutions. The course also includes relevant information beyond platform specifications and features, such as the transfer of Java EE specifications from Oracle to the Eclipse Foundation.

Learning Objectives
- Understand and apply security features provided by the Java EE framework
- Anticipate and address common security challenges across Java EE applications
- Prepare for platform changes that may impact your Java EE solution

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers

Prerequisites
- Java Security Fundamentals
- Java Advanced Secure Coding
Course Outline

Architecture
- Introduction to PHP Architecture
- Security Architecture for Responsive Web Apps
- Traditional Full-Stack PHP Architecture
- MVC Frameworks
- Security in Microservices
- Using Modern Frameworks
- Object-Oriented Design
- Modern Frameworks Protect Against SQL Injection
- Security Helpers
- Unit Testing
- Principle: Defense in Depth
- Principle: Least Privilege
- Inter-Tier Auth/Process IDs
- Encrypted Communication Between Components
- Hacking Team

Authentication
- Advanced Password Handling
- Using JWTs
- How to Integrate with JWTs
- How to Authenticate
- Verifying the JWT
- How to Log Out
- How to Support Two-Factor Authentication
- Enrolling Your User in Google Two-Factor Authentication
- Validating Google Codes

Session Management
- Session Management in PHP
- Session Management at Scale
- Session Management Using Frameworks

Access Control
- Access Control in Frameworks
- Identities, Credentials, and Roles
- Zend Framework Roles

Input Validation
- Introduction to Input Validation
- A General Framework for Input Validation
- Client-Side Validation
- Specific Consideration of PHP 7
- Zend Framework

Output Encoding
- Introduction
- Output Encoding Basics
- Output Encoding in Frameworks
- PHP 7
- Zend Framework
- Automatic Encoding Using Template Engines

Continued on next page

Languages and Platforms

Defensive Programming for PHP Security

1 ¾ Hours  Intermediate

Course Description
This course covers how to design, build, and integrate with frameworks and native PHP functionality to secure applications. It is geared toward lead developers or developers with a strong interest in security, or the need to resolve threat model, secure code review, or penetration test reports. It provides students with the knowledge and skills required to protect data and applications from attack. By the conclusion of this course, students will understand how to design and implement security functionality, and how to leverage native PHP and common frameworks to secure their applications.

Learning Objectives
- Authenticate users, including using modern JSON Web Token single sign-on (JWT SSO), and integrate with two-factor authentication
- Store sensitive state securely on the server
- Build and enforce access control policies, including implementing role-based access control and capability-based access control (used heavily in mobile applications)
- Implement a full range of input validation and output encoding to avoid common pitfalls such as cross-site scripting (XSS), DOM-based XSS, and SQL injection
- Design and implement secure storage of sensitive data
- Configure PHP securely

Intended Audience
- Architects
- Back-End Developers
- Front-End Developers

Prerequisites
- Introduction to PHP Security
- OWASP Top 10
Languages and Platforms

Defensive Programming for PHP Security (cont.)

Error Handling, Logging, and Auditing
• Introduction
• Secure Logging
• Preventing Log Injection
• Logging Frameworks
• Log Storage and Retention
• Auditing
• Web Application Firewalls (WAFs)
• Using OWASP Core Rule Set

Advanced Data Protection
• Collection and Protection of Sensitive Records
• The Years of the Megabreach
• Concerns Both Regional and Global
• Sensitive Data at Rest
• Protecting Against Misuse
• Protecting Against Unauthorized Change
• Protecting Against Unauthorized Disclosure
• Encryption for Personally Identifiable Records
• Third-Party Packages
• Zend Framework
• ACME Cinemas in the News
• CISO Response

Business Logic Flaws
• Business Logic Flaws and Thinking Evil
• ACME Cinemas in Booking Nightmare
• Threat Modeling
• Spoofing
• Tampering
• Repudiation
• Information Disclosure
• Denial of Service
• Elevation of Privilege
• Race Conditions
• Easter Eggs
• Discovering Time Bombs, Deliberate Logic Flaws, and Easter Eggs

Configuration
• Introduction
• Configuration
• Disabling Dangerous Functions
• Dis-allow Includes from Unexpected Locations
• HttpOnly
• Secure Flag
• Scope
• Working with ZIP Files
• Working with Images
• EXIF Infiltration
• Simple Concatenation
• Backup and Incidental Files
• Dependency Checks
• TLS Configuration
• Configuring Content Security Policies
• Advanced CSP
• Unsafe Inline
• Unsafe Eval
• Privacy Breach Due to Configuration Error
Course Outline

Introduction
- A Brief History Lesson
- Limitations of COBOL
- Mainframe and COBOL Use Today

Understanding Security Principles
- Confidentiality, Integrity, and Availability
- Least Privilege
- Defense-in-Depth
- Creating Security Requirements and Test Cases
- Reusable Code

COBOL Security Myths
- Five COBOL Security Myths
- Myth 1: COBOL Applications Are Not Connected to the Internet
- Myth 2: Common Attack Techniques Do Not Apply to Mainframe Applications
- Myth 3: COBOL Applications Are Not Responsible for Input Validation
- Myth 4: COBOL Performs Automatic Bounds-Checking
- Myth 5: Hackers Are Not Interested in Targeting COBOL Applications
- COBOL Is Not Dead and the Security Risks Are Real

Typical COBOL System Assets and Security implications
- Telnet
- FTP
- SNA and VTAM
- JCL
- RACF

Secure Input Validation and Data Representation
- Input Validation Goals and Techniques
- Trust Boundaries
- Data Representation
- Output Encoding
- Output Encoding Example

Secure Database Access
- Why Databases Are Business-Critical
- Best Practices for Database Access
- Parameterized Queries
- Using Least Privilege for Database Accounts
- Storing Database Access Credentials
- Preventing Privilege Escalation
- DB2 Security Concerns

Secure Logging Practices
- How Keeping Accurate Logs Increases Security
- What to Log
- What Not to Log
- Additional Best Practices
- Preventing Log Tampering
- Example: Scrubbing Logs of Sensitive Information

Secure Error Handling
- Leaking Sensitive Information
- Failing to Clean Up
- Failing to Handle All Error Conditions
- Errors and System Functions
- Best Practices for Error Handling
- Example: Secure Error Handling

Languages and Platforms

Foundations of COBOL Security

⏰ 1 ½ Hours  ● Intermediate

Course Description
This course covers the risks associated with COBOL systems and the myths associated with COBOL programming security. Topics covered include a taxonomy of system vulnerabilities as they apply to COBOL, as well as other security issues. It looks first at COBOL programming best practices, and then demonstrates how to avoid or mitigate vulnerabilities.

Learning Objectives
- Recognize common security risks affecting COBOL programs
- Identify security vulnerabilities in COBOL code
- Write secure code to mitigate the risk

Intended Audience
- Architects
- Enterprise Developers
- Mobile Developers
- QA Engineers

Prerequisites
- None
Course Outline

Introduction
• .NET Overview
• Significant Features
• Related Components and Courses

Data Validation
• Introduction
• Sanitization and Validation
• Blacklisting and Whitelisting

Injection Defense
• Introduction
• SQL Injection
• Encoding Reserved Control Sequences Within Untrusted Input
• XML Parser Defense

Cryptography
• Introduction
• Symmetric Encryption
• Asymmetric Encryption
• Cryptographic Hashes
• Cryptographic Randomness

Logging
• Introduction
• Logging Domains
• Logging Security Use Cases
• Design Implementations, and Testing Considerations
• Events to Log
• Event Attributes
• Logging Systems

Languages and Platforms

Foundations of .NET Platform Security

40 Minutes   Intermediate

Course Description
The .NET platform serves as a powerful framework for developing a wide range of applications, from rich websites and desktop applications to versatile shared libraries and embedded systems. The platform's specific architecture and unique security model set it apart from other environments. While these traits offer developers and architects a variety of enhancements to the capabilities of their applications, they also introduce specific risks from an application security perspective. In this course, you’ll learn how to avoid application security risks when using .NET platforms.

Learning Objectives
• Identify the .NET framework components and related concepts
• Identify and strategize the use of .NET security features
• Identify limitations for each security feature
• Implement security processes into the development of .NET applications based on best practices

Intended Audience
• Architects
• Back-End Developers
• Front-End Developers
• QA Engineers

Prerequisites
• Principles of Software Security
• OWASP Top 10
Course Outline

Introduction
• What Hapi Is All About

Security in the Hapi Framework
• Hapi's Security Philosophy
• Hapi Modules and Plugins
• Hapi Hapi Joi Joi
• Logging in Hapi

Configuring Security Headers
• The Effect of Security Headers
• Security Headers in Hapi
• Overview of Current Best Practices

Serving Static Files
• Serving Static Files
• Path Traversal Vulnerabilities
• Serving Directories with Inert
• Custom Path Validation

Mitigating XSS in Hapi Views
• Hapi and Templating Engines
• Refresh on XSS
• Quick Question
• Overview of XSS Mitigations
• Mitigating XSS in Handlebars Views
• Mitigating XSS in React Views

User Authentication in Hapi
• Integrated User Authentication
• Delegating Authentication with OpenID Connect
• Combining OpenID Connect with OAuth 2.0

Session Management in Hapi
• Sessions in Modern Applications
• Cookie-Based Sessions in Hapi
• Securing Cookies
• Securing Cookies Best Practices
• Cookie Security In Hapi

Making Authentication Decisions
• Enforcing Authorization
• Authorization on Routes
• Entities and Scopes
• Implement an Internal Authorization Mechanism
• Handling External Access Tokens

Conclusion
• Hapi and Security
• Security Headers
• Serving Files
• Mitigating XSS When Serving HTML
• Authentication and Session Management
• Authorization

Languages and Platforms

Hapi.js Security

1 ¾ Hours  Intermediate

Course Description
Hapi is not your typical server-side JavaScript framework. Hapi's creators put in a lot of effort to make Hapi work for you instead of against you. In this course, we look at what that means for security. At the end of this course, you will have a thorough understanding of the Hapi-specific security aspects for building modern applications.

Learning Objectives
• Illustrate how common vulnerabilities manifest themselves in Hapi applications
• Prevent path traversal and XSS vulnerabilities in Hapi applications
• Identify how Hapi supports authentication, session management, and authorization
• Explain how OAuth 2.0 and OpenID Connect can be integrated into a Hapi application
• Define security best practices for modern Hapi applications

Intended Audience
• Back-End Developers

Prerequisites
• OWASP Top 10
• JavaScript Security
Course Outline

The Underlying Security Model
- Introduction
- Browsing Contexts
- The Concept of an Origin
- Same-Origin Policy
- Origin-Controlled Resources
- Secure Contexts
- Script Execution Contexts
- Wrap Up

Strong Isolation with Iframes
- Introduction
- Origin-Based Isolation
- The Sandbox Attribute
- Isolating Content From Your Own Origin with Sandboxed iframes
- Sandboxing Content Directly
- Wrap Up

Communication Between Contexts
- Introduction
- The Basics of Cross-Document Messaging
- Cross-Document Messaging Security Considerations
- Channel Messaging
- Building a Client-Side Architecture
- Wrap Up

Tabnabbing and UI Redressing
- Introduction
- Social Engineering Attacks
- Tabnabbing Through window.opener
- Traditional Clickjacking and UI Redressing Attacks
- Drag-and-Drop Clickjacking Attacks
- Restricting Framing as a Defense
- Conclusion

HTML5 Form Security
- Introduction
- New Form Capabilities
- Injection Threats of Form Capabilities
- Client-Side Input Validation
- Conclusion

Advanced Injection Attacks
- Introduction
- Dangling Markup Injection
- Base Tag Injection
- New XSS Attack Vectors
- Script Gadgets
- Conclusion

Client-Side Storage Mechanisms
- Introduction
- Various Storage Mechanisms
- The Storage Security Model
- Security Considerations When Using Client-Side Storage
- Conclusion

Languages and Platforms

Introduction to HTML5 Security

• 1 ¾ Hours
• Beginner

Course Description
This course introduces the security model of the web and builds on top of that. The core focus of the course is HTML5, both its weaknesses and its strengths. We'll talk about how attackers abuse legitimate interaction patterns in the browser and how to use various browser mechanisms for security. At the end, learners will have a good understanding of the security model of the web so they can spot potential security issues and implement appropriate defenses.

Learning Objectives
- Explain the isolation boundaries enforced by modern browsers
- Securely enable limited interactions between isolated contexts
- Understand how UI redressing and tabnabbing attacks work and how to defend against them
- Implement defenses to neutralize dangerous attributes of HTML5 forms
- Understand how client-side storage mechanisms enlarge the attack surface
- Illustrate the danger of injection vulnerabilities using payloads other than script injection

Intended Audience
- Architects
- Back-End Developers
- Front-End Developers

Prerequisites
- Principles of Software Security
- OWASP Top 10
Course Outline

Web Server Configuration
• Web Server and PHP Interpreter
• Handlers
• Superglobals
• Forced Browsing
• Forced Browsing: Storing Code Outside the Root
• Forced Browsing: Storing Everything in the Document Root
• Apache Caveat
• Directory Exceptions
• Run As Unprivileged User

PHP Configuration and Sandboxing
• Breaking Out of the PHP Sandbox
• OS Command Injection
• OS Command Injection: Example
• File System Access
• Path Traversal
• File System Manipulation
• disable_functions
• Handling Errors Safely
• Configuring Error Reporting

PHP Command Injection
• Dynamic Code Injection
  • eval()
  • Be Careful with Callbacks
  • Remote and Local File Inclusion
  • Remote and Local File Inclusion, Continued
  • Injecting Code as Data
  • Template Injection
  • Serialization
  • Using extract() in Views
  • Malware Loves Dynamic Code

Using the Right Tool for the Job
• (CS)PRNG
  • Native PRNGs in PHP
  • Session Management
• Is your server-side check really server-side?
• Don’t Trust Your $_FILES
• Validate Your MIME Types Correctly
• Rename Your Uploads
• Store Your Uploads Securely
• Not Just $_FILES
• TimThumb Fiasco

SQL Injection
• SQL Injections
• Nonmalicious Input
• Malicious Input
• Parameterized Queries
• Prepared Statements
• Other Database Engines
• Validating Inputs

Introduction to PHP Security

1 Hour  Intermediate

Course Description
This implementation-focused course dives into core security skills needed for the PHP platform. It provides strategies and examples for previously insecure practices in PHP.

Learning Objectives
• Explain how web servers handle requests and hand them over to PHP interpreter
• Recognize that there is no sandbox with PHP, and how to tackle those implications
• Use configuration files to control the PHP interpreter as to effectively apply the most important security controls
• Properly plan for users and the filesystem as to prevent command execution
• Recognize that the availability of dynamic code does not mean it can be used
• Generate strong random numbers in different versions of PHP with CSPRNG and PRNG
• Perform checks on $_FILES server-side and client side
• Achieve and describe the concept of plane separation with PDO and other drivers
• Set up a project (i.e., code, not a server) capable of customized error handling without showing verbose error information to the user

Intended Audience
• Architects
• Back-End Developers
• Enterprise
• Front-End Developers
• QA Engineers

Prerequisites
• Principles of Software Security
• OWASP Top 10

Continued on next page
Languages and Platforms

Introduction to PHP Security (cont.)

Mitigating Content Injection Attacks

- XSS
- Stored XSS
- Reflected XSS
- Reflected XSS, Continued
- DOM-Based XSS
- Separation of Concerns
- Output Encoding
- URL Encoding
- How to Steal Credentials Using XSS
- XML Injection
- XML External Entity (XXE)
- XML Entity Expansion (XEE)
- PHP’s XML Parsers
- Securing XML in PHP
- The Google Toolbar Attack
- LDAP
- LDAP Injection
- Escaping LDAP User Inputs

Password Storage

- Data Breaches
- Password Storage and Verification
- Simple Hash
- Simple Attacks
- Salted Hashes as Defense
- Adaptive One-Way Functions
- Selecting a One-Way Function
- Default Behavior and Calling Conventions
- Beat Practice: Specifics of Argon2

Third-Party Components

- Problem Definition
- That Time Our Framework Didn’t Protect Us
- Dependency Management 101
- To Fork or Not to Fork
- Coordinating Release
- Provenance and Distribution Integrity
- Cautionary Tale: left-pad
- Composer
- Composer Risks
- PHP’s Native Extensions
- Risks of Native Extensions
- Bottom Line
- Open source is more secure, right?

Requests and Responses

- Requests and Responses
- What triggers the output?
- Responses and Structure
- Gotta catch-all!
- Error Reporting Question
- Output Buffering
- Bear traps included!
- Redirection
- Authenticating Requests
- CSRF Tokens
Course Outline

Introduction
- Preventing Injection
- Authentication and Access
- Cryptography
- Secure Communication
- Public Key Infrastructure (PKI)
- Web Security
- Important Security Features in Java SE Versions 8–11

Preventing Injection
- Introduction to Preventing Injection
- Defending Against SQL Injection
- Encoding Reserved Control Sequences Within Untrusted Input
- XML Parser Defense

Authentication and Access Control
- Introduction
- Java Authentication and Authorization Services (JAAS)
- Policy Management: Protection Domains, Security Policies, Security Manager
- Sandbox Security
- Hot Water: Building Your Own Security Controls: Recommendations

Cryptography
- Introduction
- Managing Passwords
- Ciphers
- Digital Signatures
- Secure Random Number Generation
- Heartbleed Bug
- Message Digests

Secure Communications
- Java Secure Socket Extension (JSSE)
- Java GSS-API and Java SASL API

Public Key Infrastructure
- Introduction: Roles and Artifacts Within PKI Ecosystem
- Java’s PKI Model Support
- Trust Management in Java
- Java CertPath API
- Storing Keys/Secrets
- Revocation Services (OCSP/CRL)

Web Security
- Cross-Site Request Forgery (CSRF) Defense
- CSRF Defense Example
- Advice for Defending Against CSRF Attacks
- Open Redirect Defense
- URL Validation
- HTTP Security Response Headers
- User Interface Security

Important Security Features in Java SE 8–11
- Security Changes for Java 8–11
- Brief Considerations When Upgrading to Java 9

Java Advanced Secure Coding

1 Hour  Advanced

Course Description
Java Advanced Secure Coding continues where Java Security Fundamentals leaves off. This course discusses advanced coding concepts and significant platform security features and applies them to practical use cases to address typical business security challenges. We cover preventing injection attacks, platform authentication and access control, cryptography, secure network communications, public key infrastructure, and web security, along with a summary of newer features introduced in Java 8/9.

Learning Objectives
- Learn about platform authentication and access control libraries, cryptography, and secure communications over untrusted networks
- Learn PKI concepts and relevant Java platform security controls, such as the CertPath API, PKIX, and OCSP/CRL revocation services
- Apply practical ideas to defend against SQL injection, XML parser attacks, CSRF, XSS, URL attacks, HTTP response redirect attacks, and more using the Java platform as well as third-party security libraries, such as OWASP

Intended Audience
- Back-End Developers
- Front-End Developers
- Product Architects
- Security Architects

Prerequisites
- Java Security Fundamentals
Course Outline

Introduction
• Strong Data Typing
• Automatic Memory Management
• Byte Code Verification
• Secure Class Loading
• Exception Handling
• Security Libraries

Platform Security
• Introduction
• Strong Data Typing
• Automated Garbage Collection
• Secure Class Loading
• Byte Code Verification
• Exception Handling

Operational Concerns—Java Security Platform
• SDLC Security
• Strategic Design for Security
• Certify Your Software Against Supported Java Runtime Environments (JREs)
• Principles of Least Privilege (POLP)
• Secure by Default

Data Validation
• Introduction
• Sanitization and Validation
• Security Validation: Blacklisting and Whitelisting

Logging
• Getting Started with Logging
• Logging Domains
• Logging Security Use Cases
• Design, Implementation, and Testing Considerations
• Events to Log
• Event Attributes
• Security Logging with OWASP
• Logging Technology

Advanced Coding Concept
• Introduction
• Avoid String for Volatile Secrets
• Avoid Deserializing Objects From Untrusted Sources
• Java Native Interface (JNI) Bypasses Platform Safety Controls and Buffer Overflows
• XEE/XXE XML Attacks

Course Description
No matter what product or service you’re building, understanding Java platform security is an essential foundation. In this course, you’ll learn platform security concepts along with practical security knowledge you can immediately apply to your own project. You’ll write secure code using platform APIs and identify common mistakes. This course is beneficial whether you’re building desktop applications, web applications, service infrastructure, the Internet of Things (IoT), or embedded applications.

Learning Objectives
• Tackle Java platform security concepts and architecture
• Implement public key infrastructure (PKI) and Java trust management concepts
• Write secure code using Java SE APIs
• Avoid common platform security pitfalls

Intended Audience
• Back-End Developers
• Front-End Developers
• Product Architects
• Security Architects

Prerequisites
• None
Course Outline

Securing Spring Applications
• The Java Spring Ecosystem
• The Spring Security Framework
• Spring Security in Various Application Types

Common Web Vulnerabilities
• Server-Side Injection Attacks
• Client-Side Injection Attacks
• Secure Data Transport
• Overview of Current Best Practices

Configuring Security Headers
• The Effect of Security Headers
• Security Headers in Spring Security
• Overview of Current Best Practices

User Authentication with Spring
• User Authentication
• Handling User Authentication in Spring
• Using the "Remember Me" Feature
• Common Attacks against Authentication Forms
• Overview of Best Practices

Secure Password Storage
• Password Storage Best Practices
• Storing Passwords with Spring Security
• Supporting Multiple Password Storage Mechanisms
• Overview of Current Best Practices

Authentication with OpenID Connect
• Why Using OpenID Connect?
• Background on OpenID Connect
• Integrating OpenID Connect in Spring
• Overview of Current Best Practices

Implementing an Authorization Framework
• Access Control Models
• Leveraging Spring Security for Authorization
• Avoiding Common Authorization Vulnerabilities
• Overview of Current Best Practices

Advanced Authorization Scenarios
• API Authorization with OAuth 2.0
• Implementing a CORS Policy
• Securely Exposing WebSockets
• Overview of Current Best Practices

Managing Secrets in Your Application
• The Need for Secrets Management
• The Benefits of Centralized Secret Storage
• Using Spring Vault in Practice
• Overview of Current Best Practices

Conclusion (Spring Security)
• Overview of Current Best Practices

Languages and Platforms

Java Spring Security

1 ½ Hours  Intermediate

Course Description
This course is your guide to building secure Java Spring applications. Throughout the course, we analyze what Spring Security can and cannot do for you. We’ll show you how to integrate advanced security mechanisms with only a few lines of code and configuration. In the end, you’ll have a solid understanding of the power of Spring Security, along with the gusto to start using Spring Security in your applications.

Learning Objectives
• Understand what Spring Security brings to the table
• Determine current best practices for building secure Java Spring applications
• Describe how to use Spring Security’s authentication and authorization features in practice
• Integrate a secret vault into a Spring application

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers

Prerequisites
• Java Security Fundamentals
• OWASP Top 10
Course Outline

Introduction to JavaScript
• Introduction
• What’s So Great About JavaScript?
• Traditional Browser Security
• Same-Origin Policy
• Modern-Day JavaScript
• Server-Side JavaScript
• We Love JavaScript!

JavaScript Basics
• JavaScript Basics
• Strict Mode
• Comparing Equality Operators
• I’m Watching You
• Avoid Global Scope
• Separating Scope with ES6 Features
• Refrain From Setting document.domain

XSS and Untrusted Data Sources
• Introduction to XSS
• Dataflow
• Untrusted Sources
• I’m Loving It... or NOT
• Who Watches the Watchmen?

JavaScript Execution Contexts
• JavaScript Execution Contexts
• Inline JavaScript
• External JavaScript
• Event Handlers
• Scalable Vector Graphics
• Unique Resource Identifier (URI)
• Dynamic Execution: eval
• Taking Remote Control of the Demo Gods
• Dynamic Execution: Time Functions

XSS Defense Measures
• XSS Defenses
• Output Encoding
• Sanitization
• React Strategies
• Input Validation
• Handling JSON
• Headers: X-XSS-Protection Header
• Cookies: httpOnly Flag
• Load Resources Securely
• Resources: Subresource Integrity

Continued on next page
Languages and Platforms

JavaScript Security
(cont.)

Content Security Policy
- Content Security Policy
- Policies
- Hashing and Nonces
- CSP Implications
- CSP Limitations
- Configuring CSP
- Improving Your CSP
- Writing a Policy for Different Versions of CSP
- Extracting Inline JavaScript
- You Shall Not Pass!
- Final Note: Browser Support

Iframes and Clickjacking
- Iframes
- Clickjacking
- Defense Measure: CSP
- Defense Measure: X-Frame-Options
- Defense Measure: Frame Busting
- Iframe Sandboxing
- Iframe Sandboxing Options
- Now You See Me

Managing Third-Party Dependencies and Code Analysis
- Third-Party Dependencies
- Package Managers
- Know Your Dependencies
- Tools: Third-Party Dependency Audit
- Tools: Code Analysis
- Manual Review
Course Outline

Introduction to Modern C++ Security
- The Origins of Modern C++
- How Modern C++ Evolves
- Modern C++’s Sharp Edges
- Undefined Behavior

Data Types
- The Type System
- Integers
- Casts
- Alternative Operator Representations
- Enumerations
- Knowledge Check

Strings
- String Handling
- Secure Strings
- Secure String Implementation
- Knowledge Check

Interfaces
- Safer Interfaces
- Exceptions
- Interface Qualifiers
- Knowledge Check

Memory Safety
- Introduction to Memory Safety
- Use-After-Free and Double-Free
- Memory Lifetime
- Knowledge Check

Concurrency
- DirtyCOW
- Improper Synchronization
- Performance, Stress, and Scalability Testing
- Knowledge Check

Undefined Behavior
- Undefined Behavior
- Avoiding Undefined Behavior
- No Diagnostic Required
- Knowledge Check

Course Description

Writing secure code in modern C++ is far from trivial. This course will introduce you to the complexity of working with the C++ family of languages from a security perspective. Learn about some of the major security flaws that can lead to insecure programs and how to combat them. String handling, memory management, integer overflow and wrapping, and format string attacks are all covered.

Learning Objectives

- Describe the use cases where modern C++ is widely used and how modern C++ helps developers write safer code
- Apply new best practices to safely manipulate strings
- Identify unsafe memory handling practices and see how to handle memory safely
- Apply mitigations to deal with common integer mishandling
- Understand issues with concurrency and parallelism
- Describe how the compiler’s optimizer impacts security outcomes in modern C++

Intended Audience

- Architects
- Back-End Developers
- QA Engineers

Prerequisites

- Principles of Software Security
- OWASP Top 10

Continued on next page
Modern C++ Security (cont.)

Standard Template Libraries
- Complexity
- The STLs Dark Side
- STL Containers
- Knowledge Check

Compilers
- Optimizers
- More Undefined Behavior
- No Diagnostic Required (NDR)
- Knowledge Check

Automatic Defenses
- Stack Canaries
- Address Space Layout Randomization
- Knowledge Check

Code Reviews
- Code Review Strategies
- Avoiding Code Review Noise
- Handling Legacy Code
- Course Wrap-Up
- Knowledge Check
Course Outline

Validating Data in ExpressJS
• Validating Data Overview
• What Is Untrusted Data?
• Where to Validate Data
• Validating Data at the Request Layer
• Validating Data at the Model Layer

Handling Authentication in NodeJS Applications
• Protecting Passwords
• Protecting Against User Enumeration
• Locking User Accounts

Access Control in NodeJS
• Principle of Least Privilege and Roles
• Function-Level Access Controls
• Access Control Mistakes

Session Management in ExpressJS
• Session Hijacking
• Enabling HttpOnly Flag
• Enabling the Secure Flag
• Session Timeouts
• Session Fixation
• Forcing Reauthentication

NodeJS Transport Security
• TLS, SSL, and HTTPS
• Quick Question
• Importance of TLS
• HTTP Strict Transport Security Header
• Content Security Policy

Pug Security Concerns
• Cross-Site Scripting
• Common Templating Systems
• Server-Side Template Injection

Preventing MongoDB Query Selector Injection Attacks
• Injecting JavaScript
• Injecting Operators

Managing Third-Party Dependencies
• Unused Packages
• Package Popularity
• Check for Outdated Packages
• Check for Known Vulnerabilities
• Run a Private Repository

Course Description

The benefits of the NodeJS platform keep growing, but it can still suffer from the same common web vulnerabilities as other web application frameworks and platforms. This course looks at solutions to common security pitfalls associated with using ExpressJS, Pug, and MongoDB. We’ll also examine preventive measures for building a more secure application using a defense-in-depth approach.

Learning Objectives
• Use bcrypt for password storage
• Avoid common access control mistakes
• Use HTTP headers for additional transport and session security
• Audit third-party dependencies for known vulnerabilities

Intended Audience
• Architects
• Back-End Developers
• Front-End Developers

Prerequisites
• Introduction to Cryptography for Developers and Architects
Course Outline

Introduction
- React Components
- Secure React Components

Avoiding Component Injection Attacks (XSS)
- Common Coding Patterns for Avoiding
  - Code Injection via Props
  - Code Injection via CSS-in-JS Frameworks
  - Code Injection Vulnerabilities in Third-Party Components

Avoiding Attacker-Controlled JSON Exploits
- Common Coding Patterns for Avoiding Attacker-Controlled JSON Attacks

Avoiding Vulnerable Third-Party Libraries
- Traits of a Well-Maintained Package
- Version Control of Audited Packages

Avoiding Trust in Client-Side Routing as a Security Control
- What a UI Router Is
- Client-Side Routing vs. Server-Side
- Why Client-Side Protected Routes Are Not a Security Control

Avoiding Vulnerable Versions of React
- Major Vulnerabilities in Prior Versions

Languages and Platforms

React.js Security

1 Hour Intermediate

Course Description
React is a popular front-end web framework that has changed the way many people develop web applications. While React is fairly secure as is, there are still some things to consider when using it to build applications. This course will teach you how to avoid the common pitfalls developers encounter by assuming React will automatically protect them from all types of security issues.

Learning Objectives
- Write secure React components
- Prevent React component injection attacks
- Securely handle attacker-controlled JSON
- Avoid vulnerable third-party React component libraries
- Use client-side routing securely
- Avoid the use of vulnerable versions of React

Intended Audience
- Front-End Developers

Prerequisites
- None
Course Outline

Introduction
- Overview
- Error Handling
- Dependency Management

Golang for Web
- Vanilla Go
- Gorilla Framework
- Revel
- Gin Gonic
- Mat Ryer’s Web Services Pattern

Web Interfaces for Golang
- Enforcing HTTPS Communications
- Rest API
- Web Pages
- Forms: CSRF
- Forms: Validation and Sanitization
- Secure Headers

Concurrency in Go
- Goroutines
- Race Conditions
- DoS Prevention
- Context Package

Data Storage
- Session Management: Cookies
- Cryptographic Signature with Generated Tokens
- Session Management: JWT
- Access Control Frameworks
- SQL Databases

Cryptography
- Password Storage
- The math/rand Package
- Crypto Go Package

Logging
- Logging Events
- Runtime Crashes
- In Conclusion

Languages and Platforms

Secure Programming for Golang

⏰ 1 ¼ Hours  ⏰ Intermediate

Course Description
Go promises to make programmers more productive by being expressive, concise, clean, and efficient. In this course, we will teach you how to handle security in this popular open source programming language.

Learning Objectives
- Identify the most popular frameworks and conventions that can be used to write a web application and their security pros and cons
- Evaluate the security considerations of the different methods a client and a Go application can use to communicate to each other
- Identify the security risks and considerations when writing concurrent code
- Work with the included functions and cryptographic algorithms and understand their limitations
- Understand the nuances and techniques to get the required data logged in order to respond to potential security issues in the future

Intended Audience
- Back-End Developers

Prerequisites
- OWASP Top 10
Course Outline

Introduction to AngularJS Security
- Introduction to AngularJS
- Comparing AngularJS v1.x and v4.x
- Single-Page Applications (SPA)

AngularJS Templates and Expressions
- What Is an AngularJS Template?
- What Is an AngularJS Expression?
- Strict Contextual Escaping
- Sanitization
- Sandbox

Built-In AngularJS Security Protections
- How CSP Affects AngularJS Usage
- AngularJS and Bypassing CSP
- Making It All Work
- Using AngularJS CSRF Protections

Cross-Site Scripting Using AngularJS Expressions
- Sandbox and Its Removal
- Dangers of Using Server-Side Templates
- Abusing Expressions
- Angular.element Issues

Authentication in AngularJS
- Common Issues in Single-Page Applications
- Client-Side Controls
- Alternative to JWT

Authorization in AngularJS
- Ensuring Server-Side Access Controls
- Enabling Access Restrictions on Views

AngularJS Web Storage Security
- Examining Storage Services
- Keeping Data From Being Cached

Logging and Monitoring
- Handling Client-Side Exception Logging
- Types of Events to Log

Languages and Platforms

Securing AngularJS

1 ¼ Hours   Advanced

Course Description
AngularJS, the superheroic JavaScript framework from Google, is a defining technology when it comes to building single-page applications. The framework removes the drudgery from writing robust, user-driven applications but requires a different way of thinking about architecture and security. This course takes a tour through common mistakes developers make when building AngularJS applications, how these mistakes can introduce security vulnerabilities, and how to avoid them so you don't get compromised.

Learning Objectives
- Understand the security protections built into AngularJS
- Architect secure single-page applications
- Avoid coding practices that lead to template injection
- Prevent cross-site scripting (XSS) vulnerabilities
- Understand the role of authentication and authorization in single-page applications

Intended Audience
- Front-End Developers

Prerequisites
- JavaScript Security
- OWASP Top 10

---

Synopsys eLearning Catalog  |  March 2021

Twitter | Facebook | LinkedIn | synopsys.com
Course Outline

Introduction
• Introduction to the Securing Express.js Course Content

Execution Environment
• Dropping Privileges
• Avoiding Information Leaks During Crashes
• Keeping Express Running
• Manifest Security Enforcement

Secure Coding Patterns
• Error Handling
• Prototype Pollution
• Handling Regex
• Single Thread Concerns
• Zeroing Buffers

Handling Secrets
• Keeping Secrets in the Environment

HTTP Transport Security
• Versions
• Ciphers
• Pinning
• Strict Transport

HTTP Security Headers
• Helmet.js
• Common HTTP Headers
• Secure Cookie Usage

Secure Request Handling
• Validation
• Sanitization
• Contextual Escaping
• Route Regex
• Rate Limiting
• Whitelisting vs. Whitelist
• Static Assets
• Code Execution
• Command Execution

Secure Response Handling
• Validation
• Sanitization
• Contextual Escaping
• Information Leakage
• Server-Side Rendering

Database Communication
• NoSQL
• SQL
• Parameterization
• Validation
• Contextual Escaping
• Error Handling
• Connection Secrets

Logging
• Life-Cycle
• Contents
• Sanitization
• Contextual Escaping

Languages and Platforms

Securing Express.js

1 Hour  Intermediate

Course Description
In this course, you’ll learn how to write secure Express applications. This course will give you the tools, perspectives, and patterns you need to security harden all aspects of your Express applications. We’ll cover defensive coding techniques and show you how to prevent common vulnerabilities like cross-site scripting and SQL injection.

Learning Objectives
• Manage vulnerabilities in third-party library dependencies
• Securely handle application secrets
• Leverage Transport Layer Security (TLS)
• Leverage security-related HTTP headers
• Securely implement server-side templating
• Avoid common vulnerabilities like cross-site scripting and SQL injection

Intended Audience
• Architects
• Back-End Developers

Prerequisites
• JavaScript Security
Course Outline

Python Primer
• Features
• Python for Web Development
• Python 3.x
• Quick Question
• Reasons to Switch

Python 3.x Core Languages Security Considerations
• Errors and Exceptions
• Eval() and Input() Functions
• Unicode Strings
• OpenSSL
• Final Note on Syntax

Web Application Security
• Cross-Site Scripting
• Contextually Aware Output Encoding
• Server-Side Template Injection
• Login with Facebook

Web Application Security 2: Attacks on Persisted Assets
• Injection Overview
• SQL Injection
• Quick Question

Web Development Framework Configuration
• Web Development Framework Scope of Responsibility
• CSRF Protection
• Limiting Scope and Access to Attacks
• Same Site Cookies
• Referrer Policy
• Transporting Tokens Securely
• Token Expiry
• HTTP Strict Transport Security
• Allowed_Hosts
• Environment-Based Secrets Management
• Matching Definitions
• TLS Configuration
• Error Handling
• Clickjacking
• Monitoring and Notification

Session Management
• Attacks on Session Management
• Persistent Sessions
• Quick Question
• Scope
• Expiry
• Protecting Cookie Integrity

Authentication and Authorization
• Authentication Factors
• Attacker Objectives
• Matching Definitions
• Authentication
• Password Storage
• Authorization Overview
• Is the User Logged In?
• Authorizing a User to Access a Controller
• Enforcing CRUD Properties on Model Data

Languages and Platforms

Securing Python Web Applications

1 Hour
Intermediate

Course Description
Python is one of the most popular programming languages in the world. This flexible, open source language can be used to develop software for everything from simple prototyping to high-performance gaming applications, enterprise-grade web frameworks, mobile applications, and more. This course dives deep into defensive programming techniques for Python, with examples using the two most well-known web frameworks built on top of Python: Django and Flask.

Learning Objectives
• Use the framework’s built-in security features to provide defense-in-depth
• Protect against common attack vectors in the OWASP Top 10
• Identify the security responsibilities and address how to build security in
• Configure a web development framework
• Implement session management securely
• Effectively implement authentication and authorization

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers

Prerequisites
• OWASP Top 10
• Principles of Software Security
Cloud Platforms
Course Outline
Introduction to Cloud Security
- Introduction
- Cloud Deployment Methods
- Software Delivery Mechanisms
- Security Challenges and Barriers

Cloud Infrastructure Security
- Introduction
- Identity and Access Management
- Network Security: Virtual Private Cloud
- Patch Management
- Hijacking Tesla's AWS Credentials

Data Security
- Introduction
- Object Storage Security
- Persistent Disks
- Managed Databases
- Cloud Storage Double Whammy

Container and Serverless Security
- Introduction
- Container Security
- Serverless Security

Monitoring and Alerting
- Introduction
- Security Metrics Collection
- Monitoring
- Alerting
- Metrics Found the Miner

Cloud Platforms

Introduction to Cloud Security

1 Hour Beginner

Course Description
The cloud is here to stay. As development and software delivery move rapidly toward cloud infrastructure, you must be equipped to address the challenges of security and compliance. In this course, you’ll learn common cloud terminology and how to navigate the vast array of security controls you need to consider when moving to a cloud provider. By course end, you’ll understand how to address common security challenges of running software in cloud infrastructure.

Learning Objectives
- Identify different cloud delivery models
- Evaluate security features offered by public cloud providers
- Build cloud infrastructure with security in mind
- Protect data stored in cloud environments
- Build security controls into cloud technologies such as serverless and containers

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers

Prerequisites
- None
Course Outline

Deployment and Container Orchestration
- The State of Software Deployment
- Continuous Integration and Deployment
- Incorporating Security into DevOps
- Modern Infrastructure and Container Orchestration
- Container-Related Threats

Container Security
- Introduction to Container Technology
- Anatomy of a Container
- Docker Security Considerations
- Container Isolation
- Securing Container Images
- Dockerfile Best Practices
- Docker Image Vulnerabilities in the Wild
- Docker Build Process Compromised with Backdoor

Container Deployment Techniques
- Image Deployment Security Considerations
- Immutable Infrastructure
- Running Containers Securely
- Host Security
- Container Orchestration
- Patch Slow, Get Owned Fast

Introduction to Kubernetes
- Overview and History of Kubernetes
- Kubernetes Components

Working with Kubernetes Clusters and Networking
- Interacting with Kubernetes Clusters
- Kubernetes Networking
- Kubernetes Consoles Exposed to the Internet
- A Cryptocurrency Miner Found on Internal Kubernetes Cluster

Kubernetes Security Considerations
- Authentication
- Authorization
- Kubelet Security
- Managing Secrets

More Kubernetes Security Considerations
- Running Containers Securely
- Security Context and Policies
- Pod Security Standards
- Network Security
- Auditing and Logging
- Kubelet Access Anyone?

Managed Kubernetes in the Cloud
- Managed Kubernetes
- Google Kubernetes Engine
- Amazon Elastic Kubernetes Service
- Azure Kubernetes Service

DevSecOps Pipelines
- DevSecOps Pipeline Security
- Securing Third-Party Dependencies
- Container Security Tools and Resource
- Bug Tracking
- Yes, Even More Illicit Cryptocurrency Mining

Cloud Platforms

Secure Implementation of Docker and Kubernetes

1 Hour  Intermediate

Course Description
The cloud as we know it is changing. Containers have taken center stage as the preferred method of developing and deploying software into production. As security practitioners, we must adapt to the latest technologies or be left in the dust. This course will focus on building a modern cloud infrastructure capable of taking containers from a developer's laptop to production in a secure manner.

Learning Objectives
- Grasp the core concepts surrounding the Docker and Kubernetes ecosystems and implement security controls in each of these technologies
- Understand the need for containers and container orchestration tools
- Implement security hardening techniques in Docker and Kubernetes
- Build security checkpoints within the SDLC and DevOps pipelines
- Understand the importance of containers when moving toward DevOps

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- QA Engineers

Prerequisites
- Principles of Software Security
Course Outline

Introduction to AWS Security
- Introduction to Cloud Security
- Shared Responsibility Model
- AWS Architecture
- Application Security
- Compliance and Governance
- AWS Security or Bust

Infrastructure Security
- Virtual Private Cloud (VPC) Security
- Network Routing
- Security Groups
- Firewalls and Access Control Lists
- Comparison
- Denial of Service Protection
- Web Application Firewall
- AWS Systems Manager
- AWS Firewall Manager and Network Security Strategy

Identity and Access Management
- Identity Access Management Overview
- IAM in AWS
- Temporary Tokens
- Authentication for Your Apps
- Permissions and Access Control
- Storing and Accessing Credentials
- Summary
- Compromising a Password Manager’s Password

Data Security
- Data Handling Introduction
- Encrypting EBS Volumes
- S3 Data Protection
- S3 Security Best Practices
- AWS Key Management Service
- Data Backups, Retention, and Disposal
- 2017: The Year of the Misconfigured S3 Bucket

Monitoring and Alerting
- Introduction to Security Monitoring in the Cloud
- Auditing and Logging
- CloudTrail Security Monitoring in AWS
- Creating Alerts Using CloudWatch
- Extending Alerting Capabilities
- AWS Config
- GuardDuty Security Monitoring
- Putting It All Together

Course Description

This course dives into the world of secure full-stack software development and deployment using Amazon AWS. Learn to use AWS-specific tools and features to ensure your application’s production data is adequately protected and monitored. By course end, you should understand how to set up a basic hardened AWS infrastructure capable of deploying a production web application.

Learning Objectives
- Deploy web applications securely using AWS
- Implement robust identity and access management controls
- Utilize built-in AWS security features
- Store and transmit data in AWS environments securely
- Integrate security monitoring and alerting

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers

Prerequisites
- Introduction to Cloud Security
Course Outline

Introduction to Google Cloud Platform
• Introduction
• Shared Responsibility in GCP
• Compliance
• Privacy and Data Protection
• GCP Resource Hierarchy
• Regions and Zones
• Application Security in GCP

GCP Infrastructure and Network Security
• Introduction
• GCP Reference Architecture
• Virtual Private Cloud
• Routing
• Load Balancing
• Firewalling
• External Connectivity
• DoS Protection
• Google Cloud Armor

Identity and Access Management in GCP
• Introduction
• Google Cloud Identity and Access Management
• Roles and Policies
• Using Service Accounts
• Service Accounts Best Practices
• Cloud Identity and Identity Platform
• IAM Best Practices

Data Security and Encryption
• GCP Storage Options
• Key Management
• Encrypting Data at Rest
• Cloud Storage Buckets Access Control
• VPC Service Controls
• Encrypting Data in Transit
• Secret Manager

Security Management
• Google Cloud Operating Suite
• Organization Policy Service
• Security Command Center
• Managing VMs
• Google Recommender

Logging and Monitoring
• Cloud Logging: Log Types
• Collecting, Storing, and Managing Logs
• Cloud Monitoring
• Network Monitoring
• Network Intelligence Center
• IAM Auditing and Logging
• Wrap-Up

Course Description

Securing Google Cloud Platform

1 Hour  Intermediate

This course provides guidance on building and operating a secure environment for your applications in Google Cloud Platform (GCP). You will learn how to use security features and services provided by Google to ensure that the infrastructure you create in GCP and data that is stored there is adequately protected. By the end of the course, you will understand how to set up a secure infrastructure using GCP that is capable of deploying cloud-native web applications and services.

Learning Objectives
• Securely deploy applications in Google Cloud
• Use identity and access control management features provided by Google
• Store and transmit data in Google Cloud securely
• Utilize built-in Google Cloud security features and services
• Enable logging and security monitoring mechanisms in Google Cloud

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers

Prerequisites
• None
Course Outline

Introduction to Azure
- Introduction
- Shared Responsibility Model
- Compliance and Regulation
- Azure Privacy Standards
- Azure Reference Architecture
- Application Security in Azure

Network Security
- Introduction
- Azure Virtual Networks
- External Connectivity
- Virtual Network Service Endpoints
- Network Security Groups
- Network Routes
- Load Balancing and DDoS Protection

Identity and Access Management
- Introduction
- Azure Active Directory
- Authentication
- Microsoft Identity Platform
- Azure AD B2C
- Azure AD B2B
- Managed Identities
- Resource Access Management
- Role-Based Access Control
- Azure Resource Policy
- Best Practices

Data Security and Encryption
- Data Storage Mechanisms in Azure
- Key Management
- Protecting Azure Key Vault
- Encrypting Data at Rest
- Server-Side Encryption
- Client-Side Encryption
- Encrypting Databases in Azure
- Encrypting Data in Transit
- Shared Access Signature

Azure Platform Security Features
- Introduction
- Azure Portal
- Backup and Recovery
- Azure Security Center
- Azure Defender
- Web Application Firewall

Azure Logging and Monitoring
- Azure Log Types
- Logs to Monitor
- Azure Monitor
- Azure Network Watcher
- Azure Security Center Alerts
- Azure Sentinel
- Wrap Up

Course Description

This course explores the ins and outs of securing software built and deployed on the Microsoft Azure cloud platform. Learn how to use Azure-specific features to ensure your application’s production data is adequately protected and monitored. By course end, you will understand how to set up a secure infrastructure using Azure that is capable of deploying cloud-native web applications and services.

Learning Objectives
- Deploy web applications securely using Microsoft Azure
- Implement identity and access management controls in the cloud
- Utilize built-in Azure security features
- Store and transmit data in Azure environments securely
- Integrate security monitoring and alerting

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers

Prerequisites
- None
Course Outline

Securing NoSQL
- It's Not a New Technology
- It's an Unusual Technology
- Requires a New Take on an Old Set of Vulnerabilities

Access Control
- Authentication
- 509 or CR?
- Work with Shared Clusters
- Authorization
- Roles and Privileges
- Built In Versus User Defined

Logging
- Enabling Logging Globally
- Tagging Bits to Log
- Audit Significant Events
- The Audit Guarantee™

Configuration
- Bump the HTTP Status Interface
- Locking Down the REST API
- Best Bet—the Network Infrastructure

Hardening MongoDB Databases
- Disable Internet Access
- What Is MongoDB Good For?
- Reevaluating Life Choices
- Disable Default Accounts
- Encrypt Sensitive Data
- Use TLS (Really)
- Wait, There Isn’t Encryption at Rest? (Nope)

Good Overall Practices
- Perform Input Validation (Filtr)
- Turn Off Unused Services
- Avoid $where
- Direct Object Reference
- Server Side Injection
- Remove Sample Data

Making MongoDB Weep with Security Tests
- Some Nice Attack Vectors
- No Authentication
- Weak Authorization
- Accessing the Admin Database
- Cleartext Communication
- Multiple Interface Weakness
- No Built In Data Encryption
- Working with MongoDB files
- NoSQLMap

A New Frontier

Securing MongoDB

¾ Hour  Intermediate

Course Description
MongoDB is the leading document-oriented, or NoSQL, database on the market today. It is commonly used for very quick transactional websites, but the security profile is not that well understood. In this course, we’ll cover some of the best ways to solidify your security profile.

Learning Objectives
- Correctly configure a MongoDB instance to defend against known attacks
- Build code to access a MongoDB database that prevents known attacks
- Design and develop effective MongoDB authentication and authorization
- Test a MongoDB implementation against known attacks

Intended Audience
- Back-End Developers
- Mobile Developers

Prerequisites
- OWASP Top 10
Authentication and Authorization
Course Outline

Authentication and Authorization

Introduction
- Introduction
- Advanced Scenarios

Strong Client Authentication Mechanisms
- Introduction
- The Concept of Key-Based Client Authentication
- Key-Based Client Authentication with mTLS
- Key-Based Client Authentication with JWTs
- Conclusion

Client-Based Token Revocation
- Introduction
- The Need for Token Revocation
- The Token Revocation Endpoint
- Considerations for Token Revocation
- Conclusion

Using Resource Indicators
- Introduction
- Typical OAuth 2.0 Access Tokens
- Using Resource Indicators
- Practicalities for Using Resource Indicators
- Conclusion

Strengthening Bearer Tokens
- Introduction
- Strengthening Bearer Tokens with Proof-of-Possession
- Transport-Layer Proof-of-Possession with TLS
- Application-Level Proof-of-Possession with DPoP
- Handling Sender-Constrained Tokens
- Conclusion

Delegation in OAuth 2.0 Architectures
- Introduction
- Impersonation Versus Delegation
- The Idea Behind a Token Exchange
- The Token Exchange in Detail
- Additional Considerations
- Conclusion

Advanced Scopes and Consent
- Introduction
- Parametrized Scopes
- Rich Authorization Requests (RAR)
- Security Considerations
- Conclusion

Wrapping Up OAuth 2.0
- Introduction
- The Current OAuth 2.0 Landscape
- Wrapping Up

Advanced OAuth 2.0 Topics

Almost every company that exposes an API is moving toward using OAuth 2.0 as a delegation framework. Therefore, OAuth 2.0 is still under active development, and a wide variety of complex deployment scenarios are supported. In this course, we investigate various addendums to OAuth 2.0 that lift the security of almost all OAuth 2.0 aspects.

Learning Objectives
- Describe the technical requirements for proactively revoking OAuth 2.0 tokens
- Explain the challenges of token theft and the potential solutions in the OAuth 2.0 ecosystem
- Mitigate subtle weaknesses in OAuth 2.0 flows by leveraging new specifications
- Analyze the requirements for implementing a delegation scenario
- Assess whether traditional scopes suffice for user consent, and recognize which more fine-grained mechanisms might be more appropriate

Intended Audience
- Architects
- Back-End Developers
- Mobile Developers

Prerequisites
- OAuth 2.0 Security
- Securely Accessing APIs Using OAuth 2.0
- Securely Granting Access to an API Using OAuth 2.0

---

Synopsys eLearning Catalog | March 2021
Course Outline

Introduction
• Introduction
• Authentication and Authorization
• Course Overview

Simple API Authentication Mechanisms
• Introduction
• Including a Secret in the Request
• Pros and Cons of Sending Secrets
• Signing Requests with an HMAC
• Pros and Cons of Using HMacs
• Conclusion

Advanced Client Authentication
• Introduction
• The Concept of Key-Based Client Authentication
• Strategies for Handling Keys
• Client Authentication with JWTs
• Client Authentication with mTLS
• Comparing JWTs and mTLS
• Conclusion

User Authentication Strategies
• Introduction
• Traditional Server-Side Sessions
• Client-Side State
• Using OAuth 2.0
• Conclusion

Cookie Security Best Practices
• Introduction
• The Use Case for Cookies
• A Secure Cookie Configuration
• Handling Cross-Site Request Forgery
• Mitigating CSRF in APIs
• Conclusion

Securely Handling JSON Web Tokens
• Introduction
• Using JWTs for Authentication State
• Using the Correct Signing Mechanism
• Do Not Rely on Unverified Token Metadata
• Reject Unsigned Tokens
• Verify Reserved Claims
• Using Explicit Typing
• Handling Key Distribution
• Conclusion

Common Authorization Pitfalls
• Introduction
• Excessive Data Exposure
• Mass Assignment
• Broken Authorization
• Conclusion

Building a Robust Authorization Strategy
• Introduction
• The Issue with Complex Authorization Logic
• Centralizing Authorization
• Offloading Authorization to a Policy Decision Point
• Conclusion
• Course Wrap-Up

Essential API Security

1 ¾ Hours  Intermediate

Course Description
This course covers the most common and devastating API security vulnerabilities of the OWASP Top 10. Throughout this course, you will learn about real-world authentication and authorization failures and potential solutions. Topics include API authentication, authorization decisions, and handling state with tokens.

Learning Objectives
• Identify different API authentication strategies
• Assess which API authentication technique fits your use case
• Explain how to handle user authentication state in APIs
• Identify current cookie security best practices
• Configure the API to handle JWTs securely
• Identify common authorization pitfalls in API
• Audit and improve an API authorization policy

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers

Prerequisites
• None
Course Outline

The Need for OAuth 2.0
- An Example OAuth 2.0 Scenario
- The Valet Key Analogy
- Valet Keys in Our Application

Delegated Access with OAuth 2.0
- A Brief History of OAuth 2.0
- OAuth 2.0 Terminology
- Conceptual Overview of OAuth 2.0
- OAuth 2.0 Clients

Overview of OAuth 2.0 Grant Types
- Overview of Different Grant Types and Their Purposes
- Authorization Code Grant
- Device Authorization Grant
- Client Credentials Grant
- Implicit Grant
- Resource Owner Password Credentials Grant

Delegated Access From a Confidential Client
- A Confidential Client Scenario
- Delegated Access with the Authorization Code Flow

Delegated Access From a Public Client
- A Public Client Scenario
- Augmenting the Authorization Code Grant with PKCE
- Mobile and Native Clients
- Front-End Web Clients

Long-Term Delegated Access
- The Purpose of Access Tokens
- Running a New Flow
- Quick Question
- Using Refresh Tokens
- Securing Refresh Tokens

Common Pitfalls and Misconceptions
- Mistaking OAuth 2.0 for What It Is Not
- Abusing OAuth 2.0 for Authentication
- Modifying OAuth 2.0 Flows

Wrapping Up OAuth 2.0
- The Core Concepts of OAuth 2.0
- Quick Questions
- High-Level Security Considerations

Authentication and Authorization

OAuth 2.0 Security

1 ½ Hours  Intermediate

Course Description
OAuth 2.0 is a widely used framework for securing access to APIs. But due to its complexity, many developers struggle to use and integrate OAuth 2.0 securely. This course introduces the core concepts of OAuth 2.0 and investigates the currently recommended flows. It also provides a quick overview of deprecated flows and looks at common security pitfalls and misconceptions.

Learning Objectives
- Identify the actors/roles in an OAuth 2.0 architecture
- Differentiate between OAuth 2.0 and OpenID Connect
- Understand how access tokens are used in OAuth 2.0
- Identify current best practice flows for different types of clients
- Understand the difference between access tokens and refresh tokens

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Mobile Developers

Prerequisites
- None
Course Outline

Introduction
• What Is OpenID Connect?
• History of OpenID 1.0 to OpenID Connect
• Relation to OAuth 2.0
• SAML vs. OpenID Connect
• Terms and Definitions

Tokens
• Anatomy of the ID Token
• JSON Web Tokens
• Access Tokens and Refresh Tokens
• Offline Access

Grant Types
• Authorization Code Flow
• Implicit Flow
• Hybrid Flow
• Common Insecure Design Pattern: Covert Redirect

Requesting User Information
• What is a Claim?
• Standard Claims
• Address Claim
• Additional Claims

Request Object
• Passing a Request Object by Value
• Passing a Request Object by Reference
• Validating Requests

Discovery
• WebFinger
• Issuer Endpoint Discovery
• Configuration Discovery

Registration
• Client Metadata
• Client Configuration Endpoint

Signatures and Encryption
• Signing
• Encryption
• Reasons for Signing and Encrypting

OpenID Connect with Google
• Generate and Store State Token
• Request Authorization Code
• Confirm State Token Authenticity
• Exchange the Authorization Code for an Access Token and ID Token

Authentication and Authorization

OpenID Connect

Course Description
Single sign-on and federated identity on the web have long been a nightmare. Overly complex XML-based systems are hard to implement in an interoperable way. OAuth 2.0 seems like a good solution but was never built for authentication. As a consequence, many OAuth 2.0-based authentication systems are insecure.

Enter OpenID Connect. Designed for authentication and built on top of OAuth 2.0, OpenID Connect addresses many problems developers have struggled with over the years. This course positions OpenID Connect and explores how to authenticate end users against an identity provider. By applying these principles, you can significantly improve the architecture of your application.

Learning Objectives
• Position OpenID Connect in the world of web-based delegation and identity federation systems
• Implement a secure OpenID Connect flow for various types of web applications
• Understand how identity providers offer SSO solutions and advanced session management mechanisms
• Use dynamic discovery to load the necessary information from the identity provider
• Use additional security features to fine-tune the security properties of OpenID Connect flows

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers

Prerequisites
• OAuth 2.0 Security
Course Outline

Introduction
• Different SAML Versions
• SAML in a Modern Application Landscape

The Conceptual Idea Behind SAML
• MEALSCORE: A SAML Scenario
• Different Use Cases
• The Responsibilities in a SAML Scenario

The Pros and Cons of SAML
• What SAML Can Do
• What SAML Cannot Do
• SAML, OAuth 2.0, and OpenID Connect
• Picking a SAML Implementation

Overview of SAML Building Blocks
• SAML Building Blocks
• SAML Assertions
• SAML Protocols
• SAML Bindings
• SAML Profiles
• The Role of XML

SAML Assertions
• A Real-World SAML Assertion
• Breaking Down a SAML Assertion
• Validating the Signature
• Verifying the Validity
• Using a SAML Assertion

SAML Protocols, Bindings, and Profiles
• Overview of SAML Protocols
• The Authentication Request Protocol
• Overview of SAML Bindings
• The HTTP Redirect and HTTP POST Bindings
• Overview of SAML Profiles
• The Web Browser SSO Profile

SAML for Service Providers
• Fitting SAML into the Application Architecture
• A SAML Implementation Example
• Handling Logout
• Application-Specific Security Considerations
• Supporting Multiple Identity Providers

Course Description
This course provides an overview of the Security Assertion Markup Language (SAML). It explores the building blocks of SAML as applied to a single sign-on scenario. Throughout the course, we highlight the security responsibilities of the different stakeholders in a SAML flow. Finally, we put SAML into context with more modern technologies, such as OAuth 2.0 and OpenID Connect.

Learning Objectives
• Assess whether SAML is the right solution to an IAM problem
• Describe how SAML building blocks are used to build a protocol
• Identify the role of a SAML identity provider and SAML service provider
• List crucial security requirements for a secure SAML deployment

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers

Prerequisites
• OWASP Top 10

Continued on next page
SAML Security (cont.)

SAML for Identity Providers
• Handling User Authentication
• Setting Up Delegation
• Bridging SAML to Other Protocols

SAML Security Considerations
• General Security Recommendations
• Security Considerations for Service Providers
• Security Considerations for Identity Providers

Conclusion
• SAML in Modern Applications
• OAuth 2.0, OpenID Connect, and SAML
Course Outline

Introduction
• An Example OAuth 2.0 Scenario
• Conceptual View of OAuth 2.0
• Accessing Protected Resources

Registering a Client Application
• The Need for Client Registration
• Registering Different Types of Clients
• Registering a Client
• The Importance of Redirection URIs

Scopes and Permissions
• Using Scopes
• Handling Scopes in an OAuth 2.0 Flow
• Practical Examples of Scopes
• Scopes vs. Permissions
• The Limitations of Scopes
• Practicalities of Scopes

The Client Credentials and Grant Flow
• Defining the Client Application
• Initializing the Flow
• Practicalities of the Client Credentials Grant Flow
• Limitations and Security Considerations

The Authorization Code Grant Flow
• Defining the Client Application
• Initializing the Flow
• Practicalities of the Authorization Code Grant Flow
• Limitations and Security Considerations

The Refresh token Flow
• The Purpose of Refresh Tokens
• Using Refresh Tokens
• Securing Refresh Tokens for Public Clients
• Limitations and Security Considerations

Token Security recommendations
• The Real-World Consequences of Token Theft
• Secure Token Storage
• Using Sender-Constrained Tokens
• The Limitations of Browser-Based Application Security
• Using a Backend-for-Frontend Pattern
• The Benefits of a BFF

Wrapping up
• The Role of Tokens
• The "state" Parameter
• Secure Communication Channels

Authentication and Authorization

Securely Accessing APIs Using OAuth 2.0

Course Description
In the modern web, many APIs depend on OAuth 2.0 to implement proper access control. Therefore, applications accessing these APIs need to implement support for OAuth 2.0. There are four main OAuth 2.0 flows (and several additional ones), and each supports a particular scenario. In this course, we focus on how to use OAuth 2.0 to access remote APIs. We look at the registration of clients, the different flows, and additional security considerations.

Learning Objectives
• Understand how to register a client application and request proper permissions
• Select the right OAuth 2.0 flow based on the use case they are handling
• Implement an application that uses OAuth 2.0 to access remote APIs
• Understand relevant security considerations to protect sensitive tokens

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers
• Mobile Developers

Prerequisites
• OAuth 2.0 Security
Course Outline

Introduction
• Examples of Protected Resources
• The Complexity of OAuth 2.0

Authentication, Authorization, and OAuth 2.0
• API Security Based on OAuth 2.0
• The Relationship Between Authentication, Authorization, and OAuth 2.0
• Current Best Practice OAuth 2.0 Flows

Different Access Token Types
• Different Types of Access Tokens
• Reference Tokens
• Self-Contained Tokens
• Considerations for Choosing a Token Type

Handling Incoming Access Tokens
• Getting the Access Token From the Request
• Introspecting Reference Tokens
• Verifying Self-Contained Tokens
• Security Considerations

Restricting Access Using Well-Defined Scopes
• The Purpose of Scopes
• Using Scopes for Authorization
• Practical Use of Scopes

Implementing Access Control Using OAuth 2.0
• Requiring Authorization on API Endpoints
• Getting the Access Token
• Differentiating Between Token Types
• Handling Self-Contained Tokens
• Handling Reference Tokens
• Putting It All Together
• Checking Access Token Scopes
• Making Object-Level Authorization Decisions

Token Handling Pitfalls and Best Practices
• The Purpose of Tokens
• Custom Token Claims
• Roles and Permissions in Tokens
• More Advanced Topics

Course Description
Modern web applications are often backed by an API. These APIs depend on OAuth 2.0 to implement access control. Since OAuth 2.0 is a delegation framework, implementing access control is not as simple as it seems. In this course, we look at the architecture of a back-end application using OAuth 2.0, and investigate the security properties of various kinds of access tokens. We also look at the importance of token introspection, and how to use that data to make access control decisions.

Learning Objectives
• Understand the difference between the types of access tokens
• Understand the role of a token introspection endpoint
• Define a custom set of scopes for fine-grained access control
• Secure access to an API using OAuth 2.0
• Avoid common pitfalls when designing OAuth 2.0-based security architectures

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers

Prerequisites
• OAuth 2.0 Security
Mobile
Course Outline

Secure Application Development for Android
• Introduction
• Security for Android Applications
• Android, Java, and Kotlin
• Conclusion

Enhanced Exception Handling in Kotlin
• Introduction
• Exception Handling in Java
• Exception Handling in Java: Question
• Handling Null Pointer Exceptions in Kotlin
• Addressing Type Safety
• Defensive Coding Guidelines for Kotlin
• Conclusion

Secure Network Communication
• Introduction
• Using HTTPS Correctly
• SSL Without HTTPS
• The Need for Certificate Pinning
• Certificate Pinning in Practice
• Best Practices for Pinning
• Conclusion

Handling Secrets and Sensitive Data
• Introduction
• Encrypting and Decrypting Data
• Storing Keys with the Android Keystore
• Password-Based Key Generation
• Conclusion

Local User Authentication
• Introduction
• Local Authentication Scenarios
• Using Biometric Authentication
• Unlocking Cryptographic Keys with Authentication
• Alternative Authentication Mechanisms
• Conclusion

API-Based User Authentication
• Introduction
• Simple User Authentication
• Introducing OpenID Connect
• Integrating OIDC into the Application
• Behind the Scenes
• Conclusion

Continued on next page
Advanced Android Security (cont.)

**Secure API Access**
- Introduction
- Scenarios for Securing API Access
- Using OAuth 2.0 for Secure API Access
- Handling Access Tokens
- Handling Refresh Tokens
- Conclusion

**Handling Web Content**
- Introduction
- Rendering Web Content on Android
- Integrating a WebView into the Application
- Bridging JavaScript and Native Code
- Web Security in a WebView Context
- Security Considerations
- Conclusion

**Securing Intents**
- Introduction
- Using Intents Within an Application
- Calling Other Applications with Intents
- Handling Incoming Intent
- Best Practices
- Conclusion

**Using Permissions Effectively**
- Introduction
- Permissions and Protection Levels
- Permissions and Protection Levels: Question
- Custom Permissions
- Requesting Permissions From the User
- Using Permission Groups
- Conclusion

**Conclusion**
- Introduction
- Language and Platform Security
- Protecting Sensitive Data
- Remote Authentication and API Access
- Handling Web Content
- Permissions and Intents
- Custom Permission
- Conclusion
Course Outline

Introduction: The Risks to Android Apps
- Man-in-the-Middle Attacks
- Reverse Engineering
- Malicious User with a Proxy
- Malicious App Installed on the Device
- Rooted Device
- Rooted Device: Stolen!

Android APK Structure
- Getting into the APK
- Keep Secrets Out of the Packages

Android Security Model
- Full-Disk Encryption
- File-Based Encryption
- Sandboxing
- Permissions Model: Protection and Consent

Permissions Issues
- Permission Redelegation
- Overly Permissioned Apps
- Consequences of Overly Permissioned Apps
- Augmented Gaming and Permissions

Intent Issues
- Unauthorized Intent Recipient
- Unauthorized Intent Recipient: An Example
- Intent Forging
- Sticky Broadcasts
- How Easily Can Attackers Find and Exploit Vulnerable Apps?

Insecure Storage
- Insecure Storage
- Saving Files Without Using MODE_PRIVATE
- Saving Files to the SD Card
- Saving Sensitive Data to SQLite Databases
- Logging Sensitive Data
- WebViews
- How Likely Is It That a Device Will Be Lost or Stolen?

Use of Client-Side Controls
- RSA Conference Mobile App
- Overreliance on Device Interrogation
- Client-Side Controls to Prevent Access to Functionality
- Reliance on Root Detection
- Reliance on Root Detection: Biometrics

Secure Communications
- Secure Communications
- Insecure Certificate Validation
- SSL/TLS Best Practices
- Certificate Planning
- Scanning for SSL/TLS Best Practices
- Compromised Certificate Authorities

Android Security

Course Description
The Android operating system has several built-in security features to protect application users from attackers—e.g., network sniffers, malicious app writers, device thieves, and more. It is important for Android application developers to understand what protections these features provide but also where they can fall short in protecting users. It is the responsibility of Android application developers to practice defensive programming to protect the users of their applications from the common attacks that attackers use to compromise applications and their data. This course teaches important information about the Android platform but also focuses on the defensive programming techniques that developers must know to write secure apps.

Learning Objectives
- Appreciate the risks to Android applications
- Understand the structure of Android package files
- Understand the Android security model and the protections provided by the Android OS
- Apply defensive programming techniques for common Android vulnerabilities

Intended Audience
- Mobile Developers

Prerequisites
- Principles of Software Security
- Foundations of Mobile Security
- OWASP Top 10
Course Outline

Mobile Platforms
- Mobile Concerns: Thick Clients
- Thick Clients: Performance vs. Security
- Thick Clients: Official Apps are Best
- Sandboxing
- Sandboxing for Android
- Sandboxing for iOS
- Code Signing
- Code Signing for Android
- Code Signing for iOS
- Code Signing: Summary
- Permissions
- App Store Verification
- Challenges
- Challenges with Rooted/Jailbroken Devices

Mobile Applications
- Mobile Web Application
- Native Mobile Application
- Hybrid Mobile Application
- Cross-Platform Frameworks
- Security Considerations

Mobile Threats
- The Mobile Threat Landscape
- Remote Attacks
- Client-Side Attacks
- Malicious Applications and Profiles
- Social Engineering
- Quick Questions
- Network Attacks

Application Security Vulnerabilities
- Insecure Data Storage
- Insufficient Transport Layer Protection
- Unintended Data Leakage
- Lack of Binary Protections

Application Security Controls
- Introduction to Security Controls
- Does Data Need to Be There?
- Securing Data at Rest
- Securing Data in Transit
- Preventing Unintended Data Leakage
- Binary Hardening

Mobile Application Security Testing
- Testing Overview
- Testing Debug Builds (Android Only)
- Testing Release Builds
- Dealing with Hardened Applications
- Other Considerations

Course Description
Mobile applications have become an everyday part of life, whether we are checking email, getting directions, or sending funny pictures to friends. While mobile devices offer an ever-increasing number of developers a chance to interact with users via their application, they also provide unique security challenges that can be difficult to understand. This course provides an overview of the risks a developer needs to be aware of when developing for mobile.

Learning Objectives
- Understand common mobile application security vulnerabilities
- Define the security controls of multiple mobile operating systems
- Articulate real-world threats to mobile applications

Intended Audience
- Architects
- Front-End Developers
- Mobile Developers
- QA Engineers

Prerequisites
- None
Course Outline

Overview and Mobile Device Usage
- Introduction
- Mobile Device Usage
- Simplified Tasks
- Required Functionality
- Constant Connectivity
- Multi-Personal Device
- User Workflow Expectations
- Lost/Stolen

Platform Overview and Integrated Controls
- Introduction
- Unix Base
- Integrated Security Controls
  - Integrated Security Controls: Data Execution Prevention
  - Integrated Security Controls: Address Space Layout Randomization
- Integrated Security Controls
  - Stack Canaries
  - Secure Enclave
  - Closed System
- Apple Watch Additional Controls

Development and Application Structure
- SDK: iOS Architecture Layers
- SDK: High-Level APIs and Hybrid Frameworks
- Application Structure
- Extension Points on Apple Platforms

Common iOS Platform Issues
- Introduction
- Reverse Engineering
- Jailbreaking Overview
- Perspectives on Jailbreaking
  - Jailbreaking and Normal Users
  - Jailbreaking and Attackers
  - Jailbreaking and Users
  - Jailbreaking and Developers
- Memory Management

Continued on next page

Mobile

Fundamentals of iOS

1 ½ Hours  Advanced

Course Description
Apple's iOS platform provides an always-expanding set of features for creating terrific user-focused applications. But these applications are still prone to security vulnerabilities, whether iOS-specific or common across platforms. This course provides an overview of the iOS operating system architecture and security issues affecting iOS applications.

Learning Objectives
- Describe the iOS architecture
- Determine the correct application type for an application's needs
- Evaluate the security implications of the language used to write iOS applications
- Describe platform security controls
- Identify common iOS application vulnerabilities

Intended Audience
- Back-End Developers
- Front-End Developers
- Product Architects
- Security Architects

Prerequisites
- OWASP Top 10
- Foundations of Mobile Security

Continued on next page
Fundamentals of iOS (cont.)

Platform Security Controls
- Platform Security Controls
- Application Sandboxing Introduction
- Application Sandbox
  - Documents
  - Library and tmp
  - Sandbox Access Caution
- Nonapplication Sandbox Storage Options: Introduction
- Outside Application Sandbox Storage
  - Cloud Based
  - Server
- Interprocess Communication (IPC): URL Scheme
- IPC Keychain Sharing (Keychain Access Groups)
- IPC: Pasteboard
- IPC Extensions
- Keychain and Biometric Security
- Cryptography and Certificate Management
- Privacy Controls

Application Issues: Input and Storage
- Data Representation and Validation
- Cryptographic Issues
  - Algorithm Usage
  - Key Management
  - Initialization
- The Many Forms of Local Storage
  - Local Storage Overview
  -NSUserDefaults
  - SQLite DB
  - iOS Keychain
  - Document Directory
  - Screenshot Snapshots
  - Memory Management
- Data Protection: API
- Data Protection: potential Weakness
- Leaky Databases
- Mobile Data Breaches

Application Issues: Authentication/Authorization and Communications
- Authentication
  - Lack of Authentication
  - Broken Authentication
  - Weak Authentication Policies
  - Insecure Session Management
  - Client-Side Authentication
- Authorization
- Communication Issues
- Interprocess Communication (IPC)
  - Custom URL Schemes
  - Pasteboard
  - Keychain Sharing
  - Custom Keyboards
  - General Other Extensions
  - Apple Push Notification Service (APNS)
  - General IPC
Course Outline

Secure Development Introduction
- Introduction to iOS Development Languages
- Introduction to Objective-C
- Variables
- Constants
- Messaging
- Pointers
- Introduction to Swift
- Variables
- Constants
- Optionals Chaining and Unwrapping
- String Interpolation
- Pointers
- So which is safer, Objective-C or Swift?
- Other Non-Language-Specific Things

Secure Communications
- Introduction
- App Transport Security
- HTTPS Recommendation
- ATS Help
- Certificate Pinning
- No Pinning
- Certificate Pinning via WKWebKit
- Push Notification
- Notification Recommendations
- Session Management

Client-Side Injection
- WebViews
- Controlling a WebView
- NAVIGATIONS DELEGATE
- XSS and JavaScript
- Introduction to SQL Injection
- SQL Injection Vulnerability Example
- Correct Example
- Note

Authentication
- Authentication
- Device Authentication vs. Application Authentication and Client-Side vs. Server-Side
- Touch ID Introduction
- Bad
- Touch ID Recommendation
- Authentication Credentials

Course Description

The Apple iOS platform provides a comprehensive set of features for creating versatile mobile applications. The platform’s specific architecture and security model sets it apart from other mobile operating environments. This distinction introduces specific risks from a mobile application security perspective. This course teaches defensive programming techniques to mitigate common risks in iOS applications. It gives special emphasis to describing key security controls provided by the platform and how to use them correctly.

Learning Objectives
- Understand the security benefits of each language and which may be the better solution to select
- Identify security concerns with IPC (basic) and what to do to prevent issues
- Identify and use secure communication techniques to protect data in transit
- Identify and use WebView components securely
- Understand how to authenticate and authorize users
- Understand the downsides of common local storage options and how to protect sensitive data

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Management
- Mobile Developers
- QA Engineers

Prerequisites
- Foundations of Mobile Security
- Fundamentals of iOS
- OWASP Top 10

Continued on next page
### Secure Programming for iOS (cont.)

#### Secure Storage
- Local Storage and Data Leakage
- Introduction to Logging
- Logging
- CAUTION!
- Background Screenshots
- Background screen shots Recommendations
- UserDefaults Introduction
- UserDefaults The Bad
- UserDefaults: Ensuring Controls
- UserDefaults: Performance Improvements
- UserDefaults: Offline Access
- UserDefaults: Server Authentication
- Data Files: Introduction
- The Bad: Data Protection API
- Recommendation: Data Protection API
- Data Protection: Performance Improvement
- Data Protection: Offline Access
- Data Protection: Server Authentication
- Keychain Services
- KeyChain Recommendations

#### Inter-Process Communication
- Inter-Process Communication in iOS
- URL Scheme
- Defining a URL Scheme
- Handling a URL
- Opening an Application Using a URL
- Recommendation
- General (System) Pasteboard
- Pasteboard Access
- Pasteboard Best Practice: Adding Security Controls
- Pasteboard Best Practice: Handoff Controls
- Named Pasteboards: Sharing Data Between Apps
- Access
- Restrictions and Best Practice
- App Group Containers
- Shared Keychain Groups

#### Binary Protections
- DISCLAIMER!
- Debugger Detection
- Jailbreak Detection
- Code Obfuscation
Requirements, Architecture, and Training
Course Outline

ARA Overview
- What Is ARA?
- When Do You Perform ARA?
- Indicators ARA Is Necessary
- Ongoing ARA

Design Flaws and the Techniques That Find Them
- Bugs vs. Flaws
- Examples of Security Flaws at Design Level
- Is There a Tool for That?
- Analysis Types for ARA

ARA Output
- What Do You Get?
- Who Uses It?
- Following Up with Secure Design

Dependency Analysis
- Dependency Analysis Overview
- Analyzing an Application’s Environment
- Implementation Considerations
- Illuminate Interfaces and Contracts
- Base Security Controls and Limitations

Known Attack Analysis
- What Is Known Attack Analysis?
- Applying Principles
- Commonly Discovered Flaws
- Technique: Build an Attack Checklist
- Technique: Build a List of Security Controls Common to Design Patterns
- Technique: Connections Between Architectural Elements
- Technique: Pay Close Attention to Dynamic Code Generation and Interpretation
- Techniques: APIs Across Stateless Protocols

System-Specific Analysis
- System-Specific Analysis
- System-Specific Analysis: Business Relation
- Discovering Intentions
- Trust
- The Problem with Trust
- Software Security Modeling Techniques
- Trust Modeling
- Data Sensitivity Classification
- Threat Modeling

Architecture Risk Analysis

Requirements, Architecture, and Training

Advanced

1 ½ Hours

Course Description
Architecture risk analysis (ARA) is a set of techniques that aims to discover design flaws and the risks they pose within a system. ARA does not replace other analysis techniques such as penetration testing and code review; it complements those techniques. Once you learn the techniques discussed in this course, you will have the skills needed to identify design-level defects—even if the system being analyzed has been pen-tested, code-reviewed, and released.

Learning Objectives
- Explain to others why a technique like ARA is required to have secure software
- Learn the different types of analyses that are used when performing ARA
- Identify the kind of output that is needed or expected when performing ARA

Intended Audience
- Application Security Specialists
- Architects
- Developers
- QA Engineers

Prerequisites
- Principles of Software Security
- OWASP Top 10 or Attack and Defense

Continued on next page
Guiding Principles for ARA
- Guiding Principles Overview
- Secure the Weakest Link
- Practice Defense in Depth
- Be Reluctant to Trust
- Storing Secrets Is Hard
- The Principle of Least Privilege
- Fail and Recover Securely
- Compartmentalize
- Promote Privacy
- Keep It Simple
- Mediate Completely
- Separation of Duties
- Make Security Usable

Models of an ARA System
- ARA Models Overview
- Component Diagram
- Threat Model Diagram

Risks, Issues, and Documentation
- Traceability Matrix
- Documenting Technical Risks
- Example: Customer Credentials
- Documenting Observations
- Security Testing
Course Outline

Introduction to Risk-Based Security Testing
• Introduction
• The Key Players
• The Software Security Discipline
• What Is Software Security?
• Two Broad Classes of Security Defects
• Security Testing
• Testing Security Functionality
• Managing Risks
• Defining What Security Means for You

Defining Requirements
• Requirements
• Functional Requirements
• Nonfunctional Requirements
• Derived Requirements
• Attributes of Good Requirements
• Security Requirements, not Security Features
• Security Requirement Types
• Nonfunctional Security Requirements
• Derived Security Requirements
• Thinking Backwards
• Automated Teller Machine: A Scenario
• Security Requirements

Getting Started
• Where Do I Start?
• Risk-Based Security Testing Process
• Security Goals
• Guiding Principles for Secure Design
• Risk Classifications
• Putting It All Together

Testing Strategies
• Adding Risk-Based Security Testing
• Integrating the RBST Process
• Using Threat Models
• Using Architecture Risk Analysis Results
• Using Abuse Cases
• What Are You Accomplishing?
• Effective Testing

Resourcing and Players
• Introduction
• Testing Tools
• Think Like an Attacker
• What Are You Up Against?

Continued on next page

Requirements, Architecture, and Training

Risk-Based Security Testing Strategy

1 ¼ Hours Intermediate

Course Description
Software security is a key element in your assurance and compliance strategy for protecting your applications and critical data. Organizations need applications that not only work correctly under normal use but also continue to work acceptably in the face of malicious attack. Software security testing extends beyond basic functional requirements and is a critical part of a secure software development life cycle. Risk-based security testing is about building confidence that attackers cannot turn security risks into security problems. This course teaches you to think like an attacker when testing your applications.

Learning Objectives
• Develop a white box testing strategy based on real-world risks to improve where and how testing resources can be focused
• Describe how to use architecture risk analysis and abuse case artifacts to enhance test plans
• Use knowledge of common software errors for developing test cases to expose them
• Strategize ways to integrate risk-based security testing into your SDLC

Intended Audience
• Back-End Developers
• Front-End Developers
• QA Engineers

Prerequisites
• Principles of Software Security
• OWASP Top 10 or Attack and Defense
Common Risk Areas: Part 1
- Security Coding Error Test Approach
- Kingdom 1: Input Validation and Representation
- SQL Injection
- Cross-Site Scripting (XSS)
- Kingdom 2: API Abuse
- Ignoring Return Values
- Using Deprecated Methods
- Kingdom 3: Security Features
- Privacy Violation
- Default Authentication
- Privilege Abuse
- Handling Secrets

Common Risk Areas: Part 2
- Introduction
- Kingdom 4: Time and State
- Parameter Tampering
- URL Tampering
- Cookie Tampering
- Kingdom 5: Errors
- Exception Handling
- Triggering Errors
- Kingdom 6: Code Quality
- Memory Leaks
- Source Code Comments and Strings
- Kingdom 7: Encapsulation
- Violations of Boundaries Between Components
- Violations of Data Trust Levels

Going Forward
- Trying It All Together
- Your Judgment Is Crucial
- Challenges in Adopting Software Security Testing
- Software Security Framework
- A Software Security Roadmap
- Mature Over Time
Course Outline

Introducing Security in the SDLC
- Introduction
- The Importance of Requirement Gathering
- What Are Software Security Requirements?
- Types of Software Security Requirements
- Internal Risk and Threat Assessment
- Threat Modeling
- Security Training and Awareness

Types of Software Security Requirements
- Introduction: Functional, Nonfunctional Requirements
- Software Security Requirements: Category Listing and Definitions
- Requirements Gathering and Methodologies
  - Intro
  - Specifications
  - Use Cases
  - User Stories
- User Stories: Misuse Stories

The Role of Software Security Requirements
- Introduction
- Security Requirements Overview
- Approaches to Security Requirements: Risk-Based and Addressing a Threat Agent
- Collecting and Prioritizing Security Requirements

Writing Effective Security Requirements
- Introduction to SMART Requirements
- Understanding SMART Requirements
  - Specific
  - Measurable
  - Attainable
  - Relevant
  - Testable
- SMART Example
- Categories of Software Security Requirements
- Examples of Software Security Requirements
- Operational Security Requirements

Verifying Security Requirements
- Introduction
- Verifying Security Requirements
- Code Review
- Security Testing
- Compliance Audit

Requirements, Architecture, and Training

Software Security Requirements

3½ Hour Beginner

Course Description
This course introduces the role of security requirements in the software development life cycle and explains how to write effective, verifiable requirements. The goal is to understand first how to incorporate security into the SDLC and then how to choose a style of security requirements that fits your project’s and organization’s needs. The course wraps up with an action plan to help learners verify the effectiveness of security requirements through security testing and hands-on auditing.

Learning Objectives
- Explain the benefit of introducing security-specific requirements as part of an overall requirements-gathering strategy
- Understand the approaches and methodologies used to write software security requirements
- Differentiate between functional and nonfunctional software requirements and understand which type of requirements-gathering technique best fits your organization
- Describe the qualities of effective security requirements and implement requirements that increase application security
- Implement verifications to ensure security requirements are met and enforced during and after deployment

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- QA Engineers

Prerequisites
- None
Regulation and Compliance
Course Outline

Principle of Data Protection by Design and by Default
- Article 25: Data Protection by Design and by Default
- Processing Only Personal Data Necessary
- Privacy an Integral Part of the Design Phase
- Introduce a Privacy Page
- Privacy Policies and Data Protection Controls

Privacy Requirements
- From Principle to Privacy Requirements
- Categories for Privacy Requirements
- Data Collection Requirements
- Data Processing Requirements
- Data Storage and Deletion Requirements

Personal Data
- Definition and Examples of Personal Data
- IP Addresses and Location Data
- Cookies and Similar Identifiers
- Examples of Data not Considered in Scope of GDPR
- Special Categories
- Indirectly Revealing Personal Data
- Deceased Persons

Getting Consent
- Requirements for End User Consent
- Getting Consent
- Consent Life Cycle
- Consent Withdrawal
- When Is Consent Valid?
- GDPR Fines Related to Consent

Personal Data Collection and Processing
- Lawfulness, Fairness and Transparency
- Purpose Limitation and Data Minimization
- Accuracy
- Right to Restrict Data Processing
- Storage Limitation
- Data Profiling

Collecting Personal Data of Children
- GDPR Mandate
- Requirements
- Reasonable Effort to Verify a User’s Age
- Conducting a Data Protection Impact Assessment (DPIA)

Accessing Personal Data
- Right of Access
- Providing Copies of Personal Data and Other Relevant Information
- Right to Edit and Correct
- Data Retention

Course Description

This intermediate course discusses GDPR principles, requirements, and personal data specifics for software developers and technical architects. It examines the processes for getting consent for personal data collection, how personal data should be accessed and shared, and provides an in-depth view on data subject access requests (DSARs) and international data transfers (IDTs). This course requires basic familiarity with the GDPR, and students are advised to complete “Introduction to GDPR” before taking this course.

Learning Objectives
- Explain the principle of data protection by default and by design and how it applies to software development
- Discuss privacy requirements and classify what data is personal data, including the special categories of personal data
- Know the processes for sharing personal data with law enforcement
- Explain data portability and deletion, as well as access control and logging in the context of the GDPR
- Define data subject access requests and know how to prepare for and respond to such requests
- Define international data transfers and explain the proper handling of data transfers to third parties

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers

Prerequisites
- Introduction to GDPR

Continued on next page
Regulation and Compliance

GDPR for Developers and Architects (cont.)

Sharing Personal Data with Law Enforcement
• Lawful Basis
• Share Only What Is Necessary
• Informing Data Subjects about Law Enforcement Requests

Data Portability and Deletion
• Exporting Data
• Deleting Data

Anonymization and Encryption
• Encrypting Data
• Encrypting Data in Motion
• Encrypting Data at Rest
• Integrity Protection
• Anonymization and Pseudoanonymization
• Anonymization and Pseudoanonymization Techniques

Data Subject Access Requests (DSARs)
• How to Prepare
• How to Receive
• How to Respond

International Data Transfers (IDTs)
• International Transfers of Personal Data
• Handling Data Transferred to Third Parties
• Map your Flows of International Data

Access Control and Logging
• Requirements for Access Control
• Keeping Audit Trails and Logs
Course Outline

Personal Data Mapping
- Categories of Personal Data
- Personal Data Inventory: Records of Processing Activities
- Data Flow Mapping

Data Subjects’ Rights
- Consent Management
- Right to Access, and Rectification
- Right to Be Forgotten
- Exporting and Transferring Data

Design Requirements
- Privacy by Design
- Requirements for Profiling
- The Role of the DPO

Data Privacy Impact Assessment
- Introduction
- When Is a DPIA Mandatory?
- DPIA Methodologies

Requirements for Development, Testing, and Production
- Data Protection
- Security Mechanisms
- Breach Notification

Regulation and Compliance

GDPR for Development and Project Managers

3½ Hour   Beginner

Course Description
This course provides a thorough understanding of GDPR's requirements for personal data protection and how they affect the software development life cycle. Lessons provide details on mapping personal data within an application and creating a dataflow map that helps identify privacy-related risks. You’ll walk away understanding how to use a data privacy impact assessment to assess risks to personal data and how to determine the needed technical and organizational controls. Security mechanisms that can be used to protect personal data are also presented at a high level.

Learning Objectives
- Create a personal data inventory for an application and update the records of processing activities accordingly
- Understand how to create a dataflow map and use it to determine risks for personal data
- Become familiar with data subject rights and how they affect the software development life cycle
- Gain a thorough understanding of the GDPR requirements that affect the design phase of software development
- Go through the basic steps of a data privacy impact assessment and use it to determine the necessary privacy controls for an application
- Understand the security mechanisms that can be used to protect personal data
- Become familiar with the processes that need to be in place to detect and report a personal data breach

Intended Audience
- Architects
- Management

Prerequisites
- None
Course Outline

Why You Need to Comply with GDPR
• Introduction to GDPR
• Scope and Applicability
• Compliance

Users’ Rights
• Right to Be Informed
• Right to Access, Rectification and Portability
• Right to Be Forgotten

Obligations to End Users
• Informing Users
• Consent Requirements
• Data Collection
• Privacy by Default and by Design
• Data Protection

Obligations to Authorities
• Records of Processing Activities
• Data Flow Mapping
• Data Protection Impact Assessment
• Breach Notification

Data Protection Officer
• The Role of DPO
• DPO Responsibilities
• Who Can Be a DPO

Course Description
This course introduces top management to GDPR. You’ll learn about the scope of GDPR, whether your organization needs to be compliant, and the results of noncompliance in both legal and practical terms. The course covers the responsibilities organizations have toward individuals and authorities in terms of GDPR, the role of the data protection officer (DPO), and how to determine whether your organization needs a DPO.

Learning Objectives
• Understand the importance of complying with GDPR, and the results, penalties, and liabilities for noncompliance
• Identify the rights of data subjects and how they translate into requirements for your organization
• Understand your organization’s requirements to protect personal data of individuals
• Identify your obligations to supervising authorities
• Understand when you need a DPO, the DPO’s responsibilities, and the skills required for this position

Intended Audience
• Management

Prerequisites
• None
Course Outline

What Is CCPA?
• Introduction: What Is CCPA?
• Scope and Penalties
• Consumer Rights at a Glance

Consumer Rights
• Purposes for Collecting Personal Information
• Right to Access and Disclosure
• Right to Deletion
• Right to Opt Out
• Right to Nondiscrimination

Requirements for Businesses
• Notices to Consumers
• Privacy Policy
• Receiving and Responding to Requests
• Handling Requests to Opt Out
• Verifying Requests
• Rules Regarding Minors
• Record-Keeping

Roadmap to CCPA Compliance
• Roadmap to CCPA Compliance
• Know Your Data
• Breach Detection
• Data Protection Best Practices

Introduction to CCPA, California Consumer Protection Act

Course Description
This course introduces the California Consumer Protection Act to developers and anyone else involved in the development life cycle. CCPA introduces several new rights for consumers based in California and also requirements for businesses, especially those with an online presence.

In this course, we provide all necessary guidelines to ensure your applications are compliant with CCPA. We describe the notices you must provide consumers and how to do this through your applications or websites. We also elaborate on how to receive, verify, and handle CCPA-related consumer requests. Finally, we highlight all necessary steps to follow to ensure continuous compliance with CCPA.

Learning Objectives
• Understand what CCPA is, whom it concerns, why it is significant, and the consequences of noncompliance
• Have a clear understanding of California consumer rights mandated by CCPA and what requirements these introduce into the software development life cycle
• Become familiar with how a business should receive, verify, process, and respond to consumer requests regarding personal data, and how to do this online
• Understand what documentation is required by CCPA, including notices to consumers, privacy policies, and so on
• Put together a roadmap to ensure compliance with CCPA

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers
• Mobile Developers
• QA Engineers

Prerequisites
• None
Course Outline

What GDPR Is and Why It’s Important
• Introduction
• GDPR Chapter Summary
• Official and Country-Specific Access to the Regulation

Timeline and Notable Fines
• Brief GDPR History
• Notable Fines

Key Roles in GDPR
• Governance Roles
• Data Subject
• Controller and Processor Roles
• Data Protection Officer (DPO)

Personal Data
• Definition and Examples of Personal Data
• IP Addresses and Location Data
• Examples of Data not Considered Personal Data
• Special Categories
• Indirectly Revealing Personal Data
• Deceased Persons

GDPR Principles for Processing and Protecting Personal Data
• Introduction
• Article 5: Principles Relating to Processing of Personal Data
• Article 6: Lawfulness of Processing
• Article 7: Conditions for Consent
• Article 8: Conditions Applicable to Child’s Consent in Relation to Information Societal Services
• Article 9: Processing Special Categories of Personal Data
• Article 10: Processing Personal Data Related to Criminal Convictions and Offenses
• Article 11: Processing Which Does Not Require Identification

Data Protection and Data Breach Concepts
• Data Protection by Default and by Design
• Data Privacy Impact Assessment
• Data Breach Notification

Key Rights of the Data Subject
• Introduction
• Right to Information
• Right to Data Access
• Right to Rectification
• Right to Withdraw Consent
• Right to Object
• Right to Object to Automated Processing
• Right to be Forgotten
• Right to Data Portability
• Obligations

Course Description

GDPR is EU legislation for the protection of personal data, adopted by EU member states in April 2016 and became enforceable on May 25, 2018. Since its introduction, it has become the model for many privacy laws worldwide and has resulted in many fines by information commissioners across Europe, in some cases to companies outside the EU.

This course introduces the regulation and the basic principles, roles, and definitions surrounding the use of personal data. It examines the rights of data subjects and provides an overview of how GDPR can impact the software development life cycle. The target audience for this course is anyone who is involved in privacy requirements in application development, management, and in roles involving data protection of personal data.

Learning Objectives
• Explain what the General Data Protection Regulation (GDPR) is, who it concerns, and how it affects the software development life cycle
• Discuss the timeline of this European Union (EU) data privacy law, including some of the fines issued since its introduction on May 25, 2018
• Know the key roles in GDPR and be able to identify personal data and special categories of personal data
• Explain the GDPR principles and know the data protection and data breach concepts of this EU data privacy law
• Define what is meant by the term “data subject” and understand the rights data subjects have under GDPR

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers

Prerequisites
• None
Introduction to PCI SSF

**Course Description**

This course provides a thorough introduction to the requirements that the PCI Software Security Framework (SSF) introduces. PCI SSF replaces the PCI Payment Application Data Security Standard (PA-DSS), which will be retired in October 2022, and introduces two new standards and associated validation and listing programs: the Secure Software Standard and Secure Software Life Cycle Standard.

The main goal of this course is to present the requirements that these standards introduce for creating payment software that is designed, engineered, developed, and maintained in a manner that protects payment transactions and data, minimizes vulnerabilities, and defends against attacks. Students will learn to conduct a gap analysis and create a roadmap to ensure that their software life cycle and the payment software they produce is compliant to PCI SSF. They will also gain a thorough understanding of the areas they need to focus on in order to protect the security of sensitive and payment data that is stored, processed, or transmitted by the software.

**Learning Objectives**

- Understand the requirements that PCI SSF introduces and how it differs from PA-DSS
- Perform a gap analysis and create a roadmap to comply with PCI SSF
- Design and implement processes and procedures to create a secure software life cycle following the PCI Secure Software Life Cycle Standard
- Design and implement security controls for payment software according to the requirements of the PCI Secure Software Standard
- Ensure the secure development of applications to protect the integrity and confidentiality of sensitive data they store, process, and transmit

**Intended Audience**

- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Mobile Developers
- QA Engineers

**Prerequisites**

- OWASP Top 10
Introduction to PCI SSF (cont.)

Security Monitoring
- Introduction
- Activity Tracking
- Managing and Protecting Activity Logs
- Security Monitoring

Communication with Stakeholders
- Communicating with Stakeholders
- Reporting Vulnerabilities
- Implementation Guidance

Terminal Software Security
- Introduction
- Terminal Software Design
- Terminal Software Security Testing and Attack Mitigation
- Documentation and Implementation Guidance
Course Outline

Introduction to PCI DSS Developer Training
- PCI DSS Training Agenda
- The Payment Card Industry and the SSC
- An Overview of the PCI DSS
- Why Are We Here?

PCI DSS v4.0 Requirements 1-4
- Understanding PCI DSS
- PCI DSS Control Group 1: Build and Maintain a Secure Network and Systems
- Control Group 2: Protect Account Data
- Focus on Requirements 3.3 and 3.4
- Focus on Requirement 3.5 and Key Management
- Focus on Requirement 4

PCI DSS v4.0 Requirements 5-12
- Control Group 3: Maintain a Vulnerability Management Program
- Focus on Requirement 6
- Control Group 4: Implement Strong Access Control Measures
- Control Group 5: Regularly Monitor and Test Networks
- Control Group 6: Maintain an Information Security Policy

Real-World Examples
- Pipeline Issues
- More Than One Way to Do It

Preparing for a PCI DSS Assessment
- First Things First
- Six Months Before Starting
- Documentation to Review
- People to Interview
- Processes to Validate

Course Description

This course provides security training for developers who work on PCI DSS–relevant applications. It covers the annual PCI DSS training requirements for developers, with a focus on the standard itself and how it impacts developers across all requirements. The course uses real-world examples to highlight lessons learned and includes a detailed look at recent and upcoming changes in the PCI DSS standard.

Learning Objectives
- Gain a high-level understanding of the PCI DSS as a whole and its relevance to developers
- Understand annual development training requirements mandated by the PCI DSS and the need to take this course and OWASP Top 10 in parallel
- Understand the changes in the recent PCI DSS version 4.0 update and their impact on developers
- Apply knowledge from a real-world example of a payment industry security event
- Understand the PCI DSS assessment process and how to be prepared as a developer
- Understand upcoming changes to the PCI DSS and other PCI SSC programs
- Evolve their own training program to continue to adapt to the ever-changing security landscape of the payment industry

Intended Audience
- Architects
- Back-End Developers
- Enterprise Developers
- Front-End Developers
- Mobile Developers
- QA Engineers

Prerequisites
- OWASP Top 10
Course Outline

Challenges of Developing Healthcare Applications
• Severity of the Problem
• Current Challenges for Software Developers in Healthcare
• Key Compliance Requirements
• Summary

Legal and Regulatory Compliance in the Healthcare Industry
• Introduction to Compliance Frameworks for Healthcare
• Significant Regulations for Cyber Security Requirements in Healthcare
• Applicability
• Applicability Continued
• Patient Rights
• Requirements for Software Developers
• When to Include Software Security
• Requirements for Purchasing Medical Equipment
• Requirements for Mobile Applications

Health Information Protection
• Definitions
• Data Protection and Encryption
• Data Integrity
• Data Recovery and Disposal

Securing Healthcare Applications
• Authentication and Authorization
• Access Control
• Quick Questions
• Auditing and Logging
• Breach Detection and Notification

Medical Devices Security
• Compliance for Medical Devices
• Security for Medical Devices
• Deployment Best Practices

Safe Development for Healthcare

1 Hour  Beginner

Course Description
This course presents the current compliance landscape for building applications for the healthcare sector. Health information is the most sensitive and critical category of personal data. This course outlines the requirements for protecting health information that are either defined by legal and compliance frameworks or established as best practices. It focuses on requirements for authentication, authorization, and access control, and also presents requirements for encryption and data integrity that were brought about by legislation such as HIPAA and the HITECH Act. It also covers the requirements for breach detection and notification. The course concludes with requirements for developing software for medical devices and deploying them in a healthcare provider.

Learning Objectives
• Understand the main legal and compliance frameworks for healthcare and the requirements they bring to developing applications
• Identify personal health information and implement security controls to adequately protect it
• Implement best practices regarding authentication and controlling access to health information
• Understand the requirements for breach notification
• Describe the security and privacy requirements when developing software for medical devices and subsequently deploying them in a healthcare provider

Intended Audience
• Architects
• Back-End Developers
• Enterprise Developers
• Front-End Developers
• Mobile Developers
• QA Engineers

Prerequisites
• None