JAV301 – DEFENDING JSP

Course Learning Objectives

Identify common vulnerabilities found in JavaServer Pages. Describe defensive coding practices and controls. Implement programming safeguards using defensive coding techiques. Decide which method to use either declarative and or programmatic security. Discover how to build software with security mechanisms in place

Description

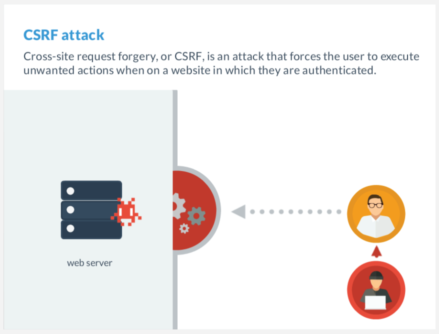
Learn how to defend your Java web apps against attacks. Using code samples from JavaServer Pages, this course covers a variety of techniques for securing against such vulnerabilities as SQL injection, Cross-Site Scripting/Request Forgery, Man-in-the-middle attacks and more.

Audience

Java Developers
Security Architects

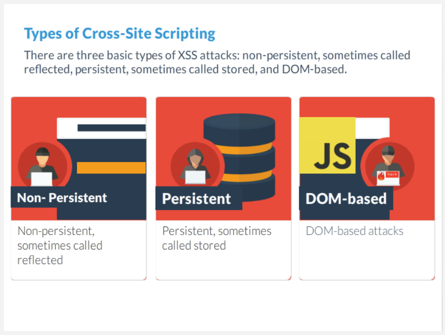
Time Required

Tailored learning - 90 minutes total



CSRF attack

Cross-site request forgery, or CSRF, is an attack that forces the user to execute unwanted actions when on a website in which they are authenticated.



Types of Cross-Site Scripting

There are three basic types of XSS attacks: non-persistent, sometimes called reflected, persistent, sometimes called stored, and DOM-based.

Non- Persistent: Non-persistent, sometimes called reflected

Persistent: Persistent, sometimes called stored

DOM-based: DOM-based attacks

* Root, trusted by browsers, and intermediate, which have certificates that the root certificate authority signed.

Two certificate authorities

Public key infrastructure and certificates

How it works is that a certificate authority verifies the identity of organizations and gives them signed cryptographic certificates to put on their servers.

Establishing trust

JAV301 – DEFENDING JSP

Course Outline

1. Authentication and Session Management

- About the vulnerability
- Declarative and programmatic security
- Session management
- Session hijacking
- Session fixation
- Weak passwords
- Insufficient session expiration policy
- Remember Me Functionality
- Harvesting Usernames
- Secure the cookie
- Use Long Session ID Numbers
- Explicitly end sessions
- Using absolute timeouts
- Use Idle timeout
- Store session information on the backend
- Reauthenticate for sensitive operations
- Strong passwords
- Other login defenses
- Don’t use remember me
- Use non-revealing error messages
- Handle forgot password requests safely
- Throttle automatic username requests

2. Access Control and Authorization

- About the vulnerability
- Access control and authorization
- Authorization policy
- Role-based access control
- Forced browsing and directory traversal attack
- Object reference
- Privilege escalation
- Access Control decisions based on request or client data
- Best practices for defending authorization and access control
- Authorization policy exercise
- Create roles, functionality, and resources chart
- Create a chart of existing permissions
- Review the charts for gaps in authorization
- Implement role-based access control (declarative)
- Implement role-based access control (programmatic)
- Deny access by default
- Use centralized access control
- Code: Use request filter
- Code: Use server-side data for authorization decisions
- Prevent directory traversal and forced browsing
- Code: Turn off directory listing

3. Application Security Services

- Cryptography
- Key management
- Establishing trust
- Inadequate password protection
- Man-in-the-Middle Attacks
- Implement HTTPS
- Code: Set the protocol
- Code: Set the cipher
- HTTP strict transport security
- Send cookies over HTTPS
- Code: Secure random number generation
- Managing keys
- Code: Using Keyczar
- Code: Symmetric and asymmetric keys
- Key purposes
- Code: Addkey command
- Key statuses
- Content of the key
- Change the key status
- Encrypting and decrypting data at rest
- Code: Create keys for signing data
- Code: Create a signature
- Establishing trust
- Code: Verify data
- String hashing
- Code: String Hashing
- Secure caching
- Turning caching off

Security Compass

JAV301 – DEFENDING JSP

Course Outline

4. Injection Attacks

- About the vulnerability
- SQL injection attacks
- Example of SQL injection
- XML and JSON injection
- Remote and local file inclusion
- Examples of remote and local file inclusion
- Command and code injection
- Using Java Native Interface (JNI)
- Use parameterized queries
- Code: Securing prepared statements
- Code: Securing stored procedures
- Validate input
- Prevent XML and JSON injection
- Restrict database permissions
- Control the result set size
- Code: Control the result set size
- Using Java Native Interface safely

5. Cross-Site Scripting Attacks

- About the vulnerability
- Types of Cross-Site Scripting
- Reflected/Non-Persistent XSS
- Stored/Persistent XSS
- DOM-Based XSS
- Input validation
- Blacklist validation
- Whitelist validation
- Regular expressions
- Numerical input validation
- Code: URL validation
- Output encoding
- OWASP Java encoder
- Output sanitization
- OWASP Java HTML sanitizer
- Code: OWASP Java HTML sanitizer
- Use jQuery to block against DOM-based XSS
- Code: Unsafe jQuery function

6. CSRF and Clickjacking Attacks

- About the vulnerability
- Cross Site Request Forgery attack
- CSRF attack - POST method
- CSRF attack - GET method
- Types of CSRF
- Stored CSRF
- CSRF in Intranets
- CSRF in network administrator sites
- CSRF in unauthenticated sites
- Clickjacking
- Use anti-CSRF tokens
- Code: Generate anti-CSRF token
- Code: Verify anti-CSRF token
- Don’t use session ID as an anti-CSRF token
- Use the double-submit cookie pattern
- Require re-authentication
- Don’t rely on HTTP request referrers
- Block Clickjacking
- X-frame options header
- Code: Load page as top-level window

Security Compass