

Java Security

Duration: 3 Days

Description

This course exposes students to the broad range of challenges and techniques that is "Java security." Secure coding practice for Java incorporates techniques for Java SE and Java EE, and increasingly EE applications are using SE techniques such as policy files and JAAS authentication. This course spends some time on each platform, so that students will be exposed to SE basics such as access controller, permissions, and policies; and also traditional EE techniques such as web-security declarations and the EJB authorization model. Best-practice chapters wrap up coverage of each platform.

The course emphasizes hands-on exercise, and students will spend more than half of their classroom time solving specific security problems. Most labs are organized as scenarios in which a security breach of existing software is possible - students begin by hacking the system in some way. Then the work of the lab is to tighten up the software to eliminate the threat: set a secure policy, sign a file, clean up overexposed parts of an API, require user login, etc.

Delegates will learn how to

- Design and implement security policies for Java applications, servers, and components.
- Manage keys and certificates for a Java application, and sign code sources as necessary.
- Practice secure design and coding, and balance usability with security in UI and API.
- Sign and verify application data and messages using the JCA, and encrypt/decrypt using the JCE.
- Incorporate JAAS authentication into an application.
- Implement a JAAS LoginModule to connect to your own application data.
- Secure Java EE applications by URL and role, and integrate JAAS authentication.
- Avoid common pitfalls of Java web applications, including SQL injection and cross-site-scripting attacks.

Outline

Java SE Security

- Holistic Security Practices

- Threats to the User
- The Class Loader and Bytecode Verifier
- System Classes and the Core API
- SecurityManager and AccessController
- Permissions
- Implication
- CodeSources
- Policies
- Configuring Java SE Security
- Dynamic Policies
- Privileged Actions

Code Signature and Key Management

- Encryption and Digital Signature
- Keystores
- Keys and Certificates
- Certificate Authorities
- The KeyStore API
- Signing JARs
- Signed CodeSources
- Additional Policy Semantics

Secure Development Practices: Java SE

- Code Injection
- Final Classes and Methods
- Singletons, Factories, and Flyweights
- Methods, Collections, and Data Hiding
- Sealing JARs
- Code Obfuscation
- Object Serialization

Cryptography

- Threats to Identity and Privacy
- The Java Cryptography Extensions
- The Signature Class
- SignedObjects
- The Java Cryptography Extensions
- SecretKeys and KeyGenerator
- The Cipher Class

- Dangerous Practices
- HTTP and JSSE

JAAS

- Pluggable Authentication Logic
- JAAS
- Packages and Interfaces
- Subjects and Principals
- ANDs and ORs
- Impersonation Methods
- Permissions for JAAS Use
- LoginContext and LoginModule
- Configuring JAAS
- CallbackHandler and Callbacks
- Implementing a JAAS Client
- Implementing a LoginModule

Java EE Security

- Java EE Servers as Code Hosts
- Tomcat Security Configuration
- Declaring Roles
- Securing URLs
- HTTP Authentication Schemes
- Securing EJBs
- Programmatic Security
- JAAS in Java EE
- Realms and LoginModules
- JAAS in Tomcat
- JACC
- Certifying a Java EE Application
- HTTPS Configuration

Secure Development Practices: Java EE

- Presentation-Tier Vulnerabilities
- User Accounts
- MVC and Security
- Validating User Input
- SQL Injection
- Cross-Site Scripting

- Reflected XSS
- Defeating XSS
- OWASP
- Penetration Testing
- Error Handling and Information Leakage
- Logging and Auditing