

912: Secure Java Programming (3 days)

Course Description...

This three-day course teaches the details of security that represent one of the cornerstones of the Java platform. Intended to be used by programmers of standalone Java applications, we explore the foundation of J2SE security rather than any J2EE specifics. This course begins with a discussion of the Java security architecture, and moves quickly into a hands-on discussion of topics that can be utilized by students upon their return to the office. In class we will discover and practice how to take advantage of secure class loading, dive into the details of the Java's security policy mechanism, release signed JAR files, deploy secure applications, and discuss common security errors made by Java programmers. The course provides extensive coverage of the Java authentication, authorization, networking and cryptography APIs.

Who Should Attend...

Attendees should include J2SE programmers interested in writing secure Java code.

Prerequisites...

Students should have proficiency with Java equivalent to:

- 430: Essential Java

Learning Objectives...

Students will leave this course armed with the skills they require to write secure Java code:

- Understand the Java security architecture
- Discover secure class loading
- Explore security policies in Java
- Establish trust with signed and sealed JAR files
- Deploy secure Java applications
- Interface with the Java authentication and authorization service (JAAS)
- Manage public and private keys and certificates
- Secure data with encryption
- Transmit data securely using secure sockets and SASL
- Avoid common Java security mistakes

Hands-On Labs:

The hands-on labs are a key learning element of this course. Each lab reinforces the material presented in lecture allowing the student to gain confidence by successfully translating theory into practice. Taken as a whole, the labs identify the essential skills that will be used on the job in the development of secure Java applications.

See next page for detailed course outline



Course Outline...

Security Fundamentals

- Challenges of mobile code
- The Java Platform's Security Architecture
- Transmitting and receiving data securely
- Authentication and Access control
- Cryptography: signing and encryption
- Managing keys and certificates
- Secure programming in Java
- Security firewalls

Cryptography

- What is "strong" encryption?
- Symmetric ciphers
- Public-key cryptography
- Message digests
- Digital signatures
- Choosing the algorithm and key lengths
- Key exchange mechanisms
- Key management (KeyStore)
- **Exercise 1: Creating and managing keys with Java tools**
- The Java cryptography APIs
- Types of ciphers
- Encrypting data
- Decrypting data
- Exchanging keys
- Extracting keys from key stores
- **Exercise 2: Encrypting and decrypting messages with JCA and JCE**

Code Location-based Security in Java

- Language security
- Prohibited operations
- Why the compiler is not enough
- Byte code modification attacks
- Byte code verification
- **Exercise 3: Learning how to detect malicious byte code modification**
- Types of class loaders
- Class loaders as implicit namespaces
- CodeSource, SecureClassLoader and URLClassLoader
- Creating custom class loader
- findClass
- Invoking methods through reflection
- Preventing decompilation of classes with encrypted jar files
- **Exercise 4: Creating a custom class loader to read encrypted classes**



Trusted Code

- Limitations of Java sandbox model
- Signing code
- Security policies
- Permission classes
- Defining a security policy
- Built-in permission classes
- Using CodeSource and Principal
- Bringing it together with ProtectionDomain
- Dynamic policies
- Policy enforcement
- Basic enforcement with SecurityManager
- **Exercise 5: Setting fine-grained access control with permissions and policies**
- Using AccessControlContext and DomainCombiner
- Customized enforcement with AccessController
- Comparing SecurityManager to AccessController
- Signing JAR files
- **Exercise 6: Ensuring access-control is restricted to trusted providers**
- Secure JRE installation
- Preventing security properties file overrides
- Configuring application specific policies
- Deploying security provider code

User-based J2SE Security

- Limitations of code location-based security
- Granting access to selected users
- Principals and Subjects
- Login contexts
- Required, sufficient, requisite and optional
- PrivilegedAction and PrivilegedExceptionAction
- Subject's doAsPrivileged
- Creating a custom permission class
- Permission "implies"
- General vs specific permissions
- **Exercise 7: Restricting application access to selected users**
- Creating a custom security manager
- Creating a custom login module
- 2-phase login protocols
- Custom callbacks
- Storing and clearing passwords
- **Exercise 8: Implementing authentication via database**



Java Network Security

- Need for secure communications
- Authenticating over untrusted networks
- Securing data transmitted over untrusted networks
- The SSL process
- SSL sockets
- Getting and processing SSL data
- `HttpsURLConnection`
- **Exercise 9: Exchanging data over SSL sockets**
- X509 trust manager
- Secure message exchanges
- Generic security service (GSS)
- Using GSS with JAAS login
- SASL for pluggable Internet authentication
- Choosing amongst SASL and JAAS/JGSS
- SASL client and server
- Negotiated security layers
- **Exercise 10: Implementing SASL authentication**

Java code-level Security *Gotchas* and best practices

- Exception management
- Access control to Fields and methods
- Problems with package-level access
- Package sealing
- Inner classes and generics
- Problems with Java initialization
- Problems with code signing
- Cloneability, serialization and deserialization
- Mutable objects, arrays and collections
- Comparing classes
- Prefer private and final
- Static, non-final variables
- Defensive firewalls
- Input validation
- SQL injection attacks
- Tainted variables
- Unit testing with random scenarios
- Privileged code blocks
- When to add security checks
- JNI best practices
- **Exercise 11: Performing a security audit**

Please contact your ROI representative to discuss course tailoring!!!