

<b>Course Name:</b>	Secure Coding with Java
<b>Duration:</b>	2 Day
<b>Medium of Instruction:</b>	Cantonese (with handout in English)
<b>Award of Certificate:</b>	Certificate of Attendance

### **Nature and Objectives:**

The course presents to the participants today's Java-based applications (web included) programming traps and ways to avoid them. Live code demonstrations let the participants understand better how hackers attack Java-based applications and what we can do in order to avoid them, introducing Java programming best practices. This course is rather more practical than theory, and participants have a lot of chances to do code analysis and code fine tuning. The course gives to the participants a lot of hands-on exercises. On completion of the course, participants should know more about Java (including in web environment) security traps; know how to write better and more secure Java code; know how to safely develop and deploy Java-based application for inter- and intranet; and get more practice on code analysis and code fine tuning.

### **Who Should Attend:**

System Administrators / Engineers / Analysts, Technical Engineers / Managers, Data Security Officers, Information Security Analysts / Managers, IT Auditors and Managers, Security Consultants and System Integrators, Senior Programmers / CTOs. Participants are expected to have hands-on experience in Windows system, knowledge on Java programming language for stand alone as well as web-based applications.

### **Course Outline:**

#### **Java Security features**

- High-level and Low-level features
- Benefits
- References

#### **Java Platform Security Architecture**

- Security models
- Basic concepts of protection mechanisms
- Permissions and security policy
- Access control mechanisms and algorithms
- Secure class loading
- Security management
- GuardedObject and SignedObject

#### **General principles**

- Validating inputs
- Taking care of insecure parameters
- Get to know how a superclass affects a subclass
- How to design classes and methods for inheritance
- Restrict the visibility
- Taking care of mutable classes
- Threats on using public static fields
- How to better define a wrapper method
- Using native methods with wrappers
- Hide sensitive information from exceptions
- How to handle unauthorized construction of sensitive classes
- Avoiding initialized instances of non-final classes

- Minimize calling methods that can be overridden
- Protect sensitive data during serialization
- View deserialization the same as object construction
- Enforce SecurityManager checking
- Use carefully java.security.AccessController .doPrivilege
- Be careful with immediate caller's class loader that can bypass SecurityManager checks
- Safely invoke standard APIs that perform tasks using the immediate caller's class loader instance

#### **Practical examples demonstrating common programming problems and solutions**

- Multipart file upload problem
- URL query string problem
- Using pseudo random number generation problem
- Properly sanitize parameterizations
- Avoid XSS attacks
- Misusing "user-agent" info could cause XSS attack
- Avoid DoS attacks
- String tokenization problem
- Denial of Service attack
- Do not accept file input by default
- Shopping card session security problem
- How to hack an AES and how to fix the problem
- How to hack an AES with Digest and how to fix the problem