Recurity Labs GmbH
Wrangelstrasse 4
10997 Berlin, Germany
http://www.recurity-labs.com

Training Abstract

"Secure Software Implementation"

The Secure Software Implementation training improves the developer's awareness of security vulnerabilities caused by implementation mistakes and faults, their exploitation by malicious parties and the prevention during the development lifecycle.

The training covers software fault identification, vulnerability classes and their exploitation in different scenarios. It also teaches real-world tested strategies for vulnerability prevention during design, implementation and maintenance of software.

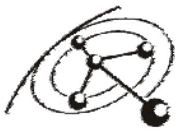
Secure Software Implementation is presented in two flavours that influence the emphasis of specific topics:

- **SSI Java** – a course aimed primarily at developers of Web centric Java applications, middleware and embedded Java software.
- **SSI C/C++** – a course specifically designed for C and C++ developers working on code with above average security and availability requirements.

Outline

Day 1 – Vulnerabilities and their consequences

- Introduction to Security Vulnerability Classes
- Logic faults
  - State and Logic Flow faults
  - Plausibility faults
  - Denial of Service
  - Real World Examples
- Buffer Overflows and Format String Vulnerabilities
  - Introduction to memory corruption attacks
  - Stack Buffer Overflows
  - Off-by-one Overflows
  - Format String Vulnerabilities
  - Other Buffer Overflow types
  - Integer Overflows and Signedness Vulnerabilities
- Injection Attacks
  - Generic Injections
  - Cross Site Scripting



  - SQL Injections
  - Directory Traversal and illegal file access
  - Command Injection
  - Code Injection (Scripting)
- Introduction to Code Auditing
  - Identification of attack surface
  - Identification of potential vulnerabilities

Day 2 – Vulnerability Prevention and Secure Development

- Security Paradigms
- Secure Design Principles
  - Input validation
  - Error Recovery
  - Containment
- Security in the development lifecycle
  - General planning considerations
  - Threat Modelling
  - Interface design
  - Prototype
  - Test cases and test driven development
  - Security centric testing
- Security strategies in code
  - Encapsulation
  - Reduction of complexity
  - Reusability
  - Storing Secrets
  - Using Cryptography
- Compilation and compiler settings
  - Compiler security features
  - Build environments
  - Runtime security
- Documentation
  - Documentation requirements for secure development
  - Documentation examples