scademy
secure coding academy

Standard Java security

CL-JSC | Classroom | 2 days

Audience: Java developers, software architects and testers
Preparedness: Basic Java development
Exercises: Hands-on

The Java language and the Runtime Environment (JRE) was designed to be free from the most problematic common security vulnerabilities experienced in other languages, like C/C++. Yet, software developers and architects should not only know how to use the various security features of the Java environment (positive security), but should also be aware of the numerous vulnerabilities that are still relevant for Java development (negative security).

The introduction of security services is preceded with a brief overview of the foundations of cryptography, providing a common baseline for understanding the purpose and the operation of the applicable components. The use of these components is presented through several practical exercises, where participants can try out the discussed APIs for themselves.

The course also goes through and explains the most frequent and severe programming flaws of the Java language and platform, covering both the typical bugs committed by Java programmers and the language- and environment-specific issues. All vulnerabilities and the relevant attacks are demonstrated through easy-to-understand exercises, followed by the recommended coding guidelines and the possible mitigation techniques.

Outline:

- IT security and secure coding
- Web application security
- Practical cryptography
- Foundations of Java security
- Java security services
- Secure communication in Java
- Common coding errors and vulnerabilities
- Knowledge sources

scademy
secure coding academy

Participants attending this course will:

Understand basic concepts of security, IT security and secure coding
Learn Web vulnerabilities beyond OWASP Top Ten and know how to avoid them
Learn how to handle vulnerabilities in the used platforms, frameworks and libraries
Have a practical understanding of cryptography
Learn to use various security features of the Java development environment
Learn about typical coding mistakes and how to avoid them
Get information about some recent vulnerabilities in the Java framework
Get sources and further readings on secure coding practices

Related courses:

- CL-JWA - Java and Web application security (Classroom, 3 days)
- CL-JWE - Java, JEE and Web application security (Classroom, 4 days)
- CL-JSM - Java and Web application security master course (Classroom, 5 days)
- CL-JAD - Java and JEE security (Classroom, 3 days)
- CL-WSC - Web application security (Classroom, 2 days)
- CL-CJW - Combined C/C++, Java and Web application security (Classroom, 4 days)
- CL-JSM - Java and Web application security master course (Classroom, 5 days)

Note: Our classroom trainings come with a number of easy-to-understand exercises providing live hacking fun. By accomplishing these exercises with the lead of the trainer, participants can analyze vulnerable code snippets and commit attacks against them in order to fully understand the root causes of certain security problems. All exercises are prepared in a plug-and-play manner by using a pre-set desktop virtual machine, which provides a uniform development environment.

scademy
secure coding academy

Detailed table of contents

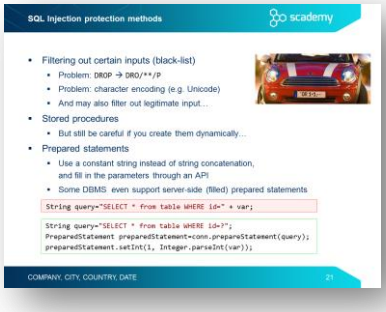
Day 1

IT security and secure coding

- Nature of security
- What is risk?
- IT security vs. secure coding
- From vulnerabilities to botnets and cybercrime
  - Nature of security flaws
  - Reasons of difficulty.....
  - From an infected computer to targeted attacks
  - The Seven Pernicious Kingdoms
  - OWASP Top Ten 2017 (release candidate)

Web application security

- Injection
  - Injection principles
  - SQL injection
    - Exercise – SQL Injection
    - Exercise – SQL injection
    - Typical SQL Injection attack methods
    - Blind and time-based SQL injection
    - SQL Injection protection methods.....
  - Other injection flaws
    - Command injection
    - Case study – ImageMagick

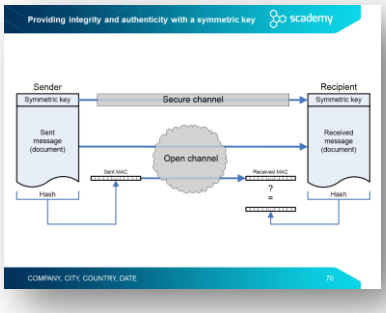


- Cross-Site Scripting (XSS)
  - Persistent XSS
  - Reflected XSS
  - DOM-based XSS
  - Exercise – Cross Site Scripting
  - Exploitation: CSS injection
  - Exploitation: injecting the <base> tag
  - Exercise – HTML injection with base tag
- Broken access control
  - Typical access control weaknesses
  - Insecure direct object reference (IDOR)
  - Exercise – Insecure direct object reference
  - Protection against IDOR
  - Case study – Facebook Notes

- Security misconfiguration
  - Hardening
  - Patch management
  - Insecure file uploads
  - Exercise – Uploading executable files
  - Filtering file uploads – validation and configuration
- Cross site request forgery (CSRF)
  - CSRF prevention
  - CSRF prevention in Java frameworks

Practical cryptography

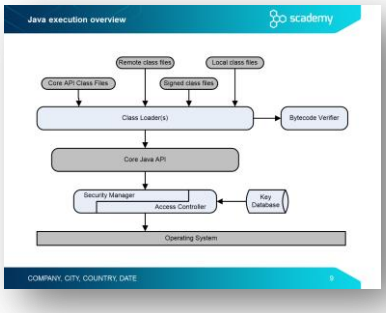
- Cryptosystems
  - Elements of a cryptosystem
- Symmetric-key cryptography
  - Providing confidentiality with symmetric cryptography
  - Symmetric encryption algorithms
- Other cryptographic algorithms
  - Hash or message digest
  - Hash algorithms
  - Message Authentication Code (MAC)
  - Providing integrity and authenticity with a symmetric key.....
- Asymmetric (public-key) cryptography
  - Providing confidentiality with public-key encryption
  - Rule of thumb – possession of private key
  - Combining symmetric and asymmetric algorithms
  - X.509 digital certificate



Day 2

Foundations of Java security

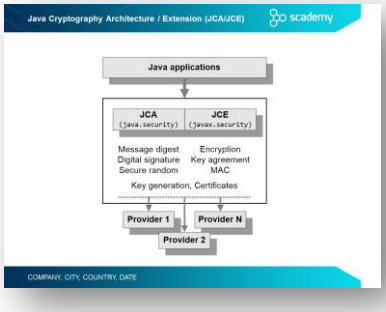
- The Java environment
- Java security
- Low-level security – the Java language and environment
  - Java language security
  - Type safety
  - Automatic memory management
  - Java execution overview.....
  - Bytecode Verifier
  - Class Loader
  - Protecting Java code
- High-level security – access control



- Protection domains
- Security Manager and Access Controller
- Permission checking
- Effects of doPrivileged

Java security services

- Java security services – architecture
- Java Cryptographic Architecture
  - Java Cryptography Architecture / Extension (JCA/JCE)
  - Using Cryptographic Service Providers
  - Engine classes and algorithms
  - Exercise Sign – Generating and verifying signatures
  - Exercise Sign – Using alternative providers
  - The Bouncy Castle (BC) provider
  - Installing the Bouncy Castle
  - Using the Bouncy Castle services

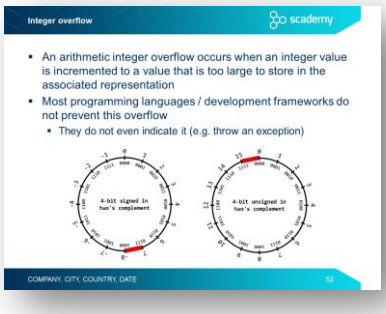


Secure communication in Java

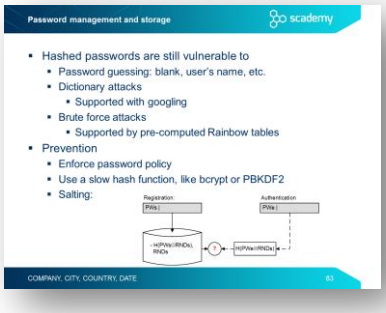
- Security services
- SSL/TLS handshake
- Java Secure Socket Extension (JSSE)
- Exercise Https – Switching from HTTP to HTTPS

Common coding errors and vulnerabilities

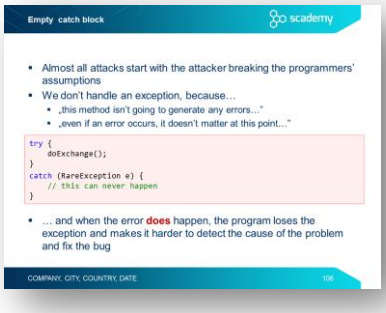
- Input validation
  - Input validation concepts
  - Integer problems
    - Representation of negative integers
    - Integer overflow
    - Exercise IntOverflow
    - What is the value of Math.abs(Integer.MIN_VALUE)?
  - Integer problem mitigation
    - Avoiding arithmetic overflow – addition
    - Avoiding arithmetic overflow – multiplication
    - Detecting arithmetic overflow in Java 8
    - Exercise – Using addExact() in Java
  - Path traversal vulnerability
    - Path traversal mitigation
  - Unvalidated redirects and forwards
  - Log forging
    - Some other typical problems with log files



- Improper use of security features
  - Typical problems related to the use of security features
  - Insecure randomness
    - Weak PRNGs in Java
    - Exercise RandomTest
    - Using random numbers in Java – spot the bug!
  - Password management
    - Exercise – Weakness of hashed passwords
    - Password management and storage .....
    - Special purpose hash algorithms for password storage
    - Argon2 and PBKDF2 implementations in Java
    - bcrypt and scrypt implementations in Java
    - Case study – the Ashley Madison data breach
    - Typical mistakes in password management
    - Exercise – Hard coded passwords
  - Accessibility modifiers
    - Accessing private fields with reflection in Java
    - Exercise Reflection – Accessing private fields with reflection
    - Exercise - Integrity protection weakness

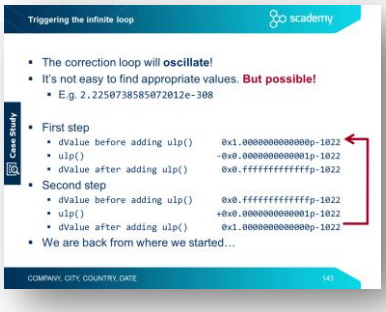


- Improper error and exception handling
  - Typical problems with error and exception handling
  - Empty catch block .....
  - Overly broad throws
  - Overly broad catch
  - Using multi-catch
  - Returning from finally block – spot the bug!
  - Catching NullPointerException
  - Exception handling – spot the bug!
  - Exercise ErrorHandling
  - Exercise – Information leakage through error reporting



- Code quality problems
  - Dangers arising from poor code quality
  - Poor code quality – spot the bug!
  - Unreleased resources
  - Private arrays – spot the bug!
  - Private arrays – typed field returned from a public method
  - Exercise Object Hijack
  - Public method without final – object hijacking
  - Immutable String – spot the bug!
  - Exercise Immutable Strings
  - Immutability and security
  - Sensitive data – spot the bug!
  - Exercise Serializable Sensitive

- Case study – The double bug in Java
  - A generic Denial of Service attack against the Java environment
  - The "2.2250738585072012e-308 bug"
  - The double bug in Tomcat
  - The vulnerable code of DoubleValue() in FloatingDecimal.java
  - Triggering the infinite loop
  - Exercise Double Bug
- Problem with inner classes
  - Inner classes – spot the bug!
  - Problem with inner classes
  - The decompiled class file containing an inner class
  - Exercise Inner Class



Knowledge sources

- Secure coding sources – a starter kit
- Vulnerability databases
- Java secure coding sources
- Recommended books – Java