
# Secure Software Implementation

## Hands-On Security Training

JSOF

Exploring the Depths of Security

# Secure Software Implementation - at a glance

This course provides a foundation for secure coding practices by applying security principles to software development.

Students will learn the importance of security in the development lifecycle and the characteristics of secure software. Students will learn to apply the best security programming practices from real-world case studies.

## Scope

- 1-2 days of training - study sessions and hands-on training
- Training focused on Secure Coding Practices
- Training performed by experts in the field of software security

## Course Targets

After attending this course, you will be able to:

- Understand terms used in secured software development
- Identify security risks in implemented software by using industry best practices on secure software development
- Analyze the (basic) security of your code
- Apply mitigation and implementation practices

You will understand:

- The importance of security throughout the development process
- The best way to approach an insecure code
- Vulnerabilities during implementation, consequences, and mitigation

## Prerequisites

Basic working knowledge of C/C++

# Course outline

- Why do we need Secure Coding?
- Top 10 Secure Coding Principles- theory and practice
    1. Validate input
    2. Heed compiler warnings
    3. Architect and design for security policies
    4. Keep it simple
    5. Default deny
    6. Adhere to the principle of least privilege
    7. Sanitize data sent to and from other systems
    8. Practice defense in depth
    9. Use Effective Quality Assurance Techniques
    10. Adopt a Secure Coding Standard
- Secure Coding case studies
- Major C/C++ pitfalls- principles and mitigation strategies
    - Strings
    - Memory management
    - Resource leaks
    - Integer type mismatches
    - Concurrency and Files

- **Hands-on exercise**