

# Principles and fundamentals of security methodologies of information systems - Secure coding good practices

M2SSIC-Metz

Pascal Steichen

- 1 Introduction
- 2 Architecture
  - Architectural Document
  - Principles of security architecture
- 3 Design
  - Principles of security design
  - Some special design issues
- 4 Implementation
  - Principles of security implementation
- 5 Operation
  - Principles of security operation
- 6 Testing
  - Recent web application vulnerabilities
- 7 Bibliographic references

# Introduction

Secure coding is not simply getting behind it's keyboard and hack, it is real engineering.

*Why do bridges support trains for hundreds of years?*

*How could an Eiffel Tower swing several meters sideways in the wind and still welcome thousands of visitors per day, without harm?*

Well engineering is taken serious. First there is an **architectural** model, which will provide **design** plans (blueprints), only then construction (**implementation**) can start.

*Why shouldn't we (IT people) adopt this procedure too?*

*Well we should!*

# Architecture

A *security architecture* is the process of selecting design elements and principles to match a defined security need. This implies to know how secure the program should become !

A good security architecture can be applied many times, to many applications and should serve as a framework for secure design decisions. A good advice is to work with an *architectural document*, where the different aspects are laid down.

## Architectural Document

- Program organization

## Architectural Document

- Program organization
- Change strategy

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions
- Major data structures

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions
- Major data structures
- Key algorithms

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions
- Major data structures
- Key algorithms
- Major objects

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions
- Major data structures
- Key algorithms
- Major objects
- General functionality

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions
- Major data structures
- Key algorithms
- Major objects
- General functionality
- Error processing (corrective or detective)

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions
- Major data structures
- Key algorithms
- Major objects
- General functionality
- Error processing (corrective or detective)
- Active or passive robustness

## Architectural Document

- Program organization
- Change strategy
- Buy versus build decisions
- Major data structures
- Key algorithms
- Major objects
- General functionality
- Error processing (corrective or detective)
- Active or passive robustness
- Fault tolerance

## Principles of security architecture

- Ask questions

## Principles of security architecture

- Ask questions
- Focus before leaping

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust
- Be stingy with privileges

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust
- Be stingy with privileges
- Always test against policy

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust
- Be stingy with privileges
- Always test against policy
- Build in fault tolerance

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust
- Be stingy with privileges
- Always test against policy
- Build in fault tolerance
- Address appropriate error-handling

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust
- Be stingy with privileges
- Always test against policy
- Build in fault tolerance
- Address appropriate error-handling
- Degrade gracefully

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust
- Be stingy with privileges
- Always test against policy
- Build in fault tolerance
- Address appropriate error-handling
- Degrade gracefully
- Fail safely

## Principles of security architecture

- Ask questions
- Focus before leaping
- Define "just secure enough"
- Do engineering not prototyping
- Identify assumptions
- Get security in from day one
- Design with the enemy in mind
- Work with the chain of trust
- Be stingy with privileges
- Always test against policy
- Build in fault tolerance
- Address appropriate error-handling
- Degrade gracefully
- Fail safely
- Choose safe defaults

## Principles of security architecture

- KISS (Keep It Simple Stupid)

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and usability
- Make accountability always possible

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and usability
- Make accountability always possible
- Limit resources consumption

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible
- Eliminate "weak links"

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible
- Eliminate "weak links"
- Use multiple layers of defense

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible
- Eliminate "weak links"
- Use multiple layers of defense
- View things in it's holistic whole

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible
- Eliminate "weak links"
- Use multiple layers of defense
- View things in it's holistic whole
- Reuse secure code

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible
- Eliminate "weak links"
- Use multiple layers of defense
- View things in it's holistic whole
- Reuse secure code
- Don't rely on off-the-shelf software

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible
- Eliminate "weak links"
- Use multiple layers of defense
- View things in it's holistic whole
- Reuse secure code
- Don't rely on off-the-shelf software
- Don't forget democratic principles

## Principles of security architecture

- KISS (Keep It Simple Stupid)
- Modularize
- Don't rely on obfuscation
- Seek statelessness
- Strive for practical measures and useability
- Make accountability always possible
- Limit resources consumption
- Make event-reconstruction possible
- Eliminate "weak links"
- Use multiple layers of defense
- View things in it's holistic whole
- Reuse secure code
- Don't rely on off-the-shelf software
- Don't forget democratic principles
- What did I forget ?

# Design

Good design is the basis for an efficient software development process, as it not only enables to build a good defensive basis into the software from the beginning, but provides safe foundations for future extensions and maintenance.

Secure design has to be elaborated thoroughly, the following steps being typical efforts to perform.

## Principles of security design

- Risk assessment

## Principles of security design

- Risk assessment
- Risk mitigation

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model
- Define high-level techniques

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model
- Define high-level techniques
- Choose appropriate measures

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model
- Define high-level techniques
- Choose appropriate measures
  - Background factors

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model
- Define high-level techniques
- Choose appropriate measures
  - Background factors
  - Business issues

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model
- Define high-level techniques
- Choose appropriate measures
  - Background factors
  - Business issues
  - Cost-benefit analysis

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model
- Define high-level techniques
- Choose appropriate measures
  - Background factors
  - Business issues
  - Cost-benefit analysis
  - Methodologies

## Principles of security design

- Risk assessment
- Risk mitigation
  - Risk assumption
  - Risk avoidance
  - Risk limitation
  - Risk planning
  - Risk acknowledgment and research
  - Risk transference
- Work with a mental model
- Define high-level techniques
- Choose appropriate measures
  - Background factors
  - Business issues
  - Cost-benefit analysis
  - Methodologies
- Evaluate the process

## Some special design issues

- Retrofitting

## Some special design issues

- Retrofitting
  - Wrappers

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code
  - Understand the security design spirit in place and follow it

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code
  - Understand the security design spirit in place and follow it
  - Learn how the program works (don't trust the manuals)

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code
  - Understand the security design spirit in place and follow it
  - Learn how the program works (don't trust the manuals)
  - Don't introduce new trust relationships

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code
  - Understand the security design spirit in place and follow it
  - Learn how the program works (don't trust the manuals)
  - Don't introduce new trust relationships
- Compartmentalization

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code
  - Understand the security design spirit in place and follow it
  - Learn how the program works (don't trust the manuals)
  - Don't introduce new trust relationships
- Compartmentalization
  - Jails

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code
  - Understand the security design spirit in place and follow it
  - Learn how the program works (don't trust the manuals)
  - Don't introduce new trust relationships
- Compartmentalization
  - Jails
  - Playpens

## Some special design issues

- Retrofitting
  - Wrappers
  - Interposition
- Maintenance
  - Handle with the same care and scrutiny as new code
  - Understand the security design spirit in place and follow it
  - Learn how the program works (don't trust the manuals)
  - Don't introduce new trust relationships
- Compartmentalization
  - Jails
  - Playpens
  - Honey pots

# Implementation

Unfortunately (already known since Morris's Internet Worm in early 1988) the most common implementation flaw is still the *buffer overflow*. Here some good implementation practices to fight them and others (of course).

## Principles of security implementation

- Inform yourself

## Principles of security implementation

- Inform yourself
- Handle data with care

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully
  - Checking cookies

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully
  - Checking cookies
  - Checking environment variables

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully
  - Checking cookies
  - Checking environment variables
  - Checking all other data sources

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully
  - Checking cookies
  - Checking environment variables
  - Checking all other data sources
  - Setting valid initial values

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully
  - Checking cookies
  - Checking environment variables
  - Checking all other data sources
  - Setting valid initial values
  - Handling filename references carefully

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully
  - Checking cookies
  - Checking environment variables
  - Checking all other data sources
  - Setting valid initial values
  - Handling filename references carefully
  - Storing sensitive data appropriately

## Principles of security implementation

- Inform yourself
- Handle data with care
  - Data cleansing
  - Bounds checking
  - Checking config files
  - Checking command-line arguments
  - Treating web content and urls carefully
  - Checking cookies
  - Checking environment variables
  - Checking all other data sources
  - Setting valid initial values
  - Handling filename references carefully
  - Storing sensitive data appropriately
- Reuse code

## Principles of security implementation

- Thoroughly review

## Principles of security implementation

- Thoroughly review
  - Peer review

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based
  - Passwords are not to be transferred in cleartext over the network

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based
  - Passwords are not to be transferred in cleartext over the network
  - Data transfert is encrypted between servers and clients

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based
  - Passwords are not to be transferred in cleartext over the network
  - Data transfert is encrypted between servers and clients
  - ...

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based
  - Passwords are not to be transferred in cleartext over the network
  - Data transfert is encrypted between servers and clients
  - ...
- Create maintainable code

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based
  - Passwords are not to be transferred in cleartext over the network
  - Data transfert is encrypted between servers and clients
  - ...
- Create maintainable code
  - Use standards

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based
  - Passwords are not to be transferred in cleartext over the network
  - Data transfert is encrypted between servers and clients
  - ...
- Create maintainable code
  - Use standards
  - Remove obsolete code

## Principles of security implementation

- Thoroughly review
  - Peer review
  - Independent Validation and Verification (IV&V)
  - Use available security tools
- Use checklists
  - Use at least passwords for user access
  - User ID's are unique
  - Access control is role-based
  - Passwords are not to be transferred in cleartext over the network
  - Data transfert is encrypted between servers and clients
  - ...
- Create maintainable code
  - Use standards
  - Remove obsolete code
  - Test changes

# Operation

Traditionally in many companies, the development staff and the operational staff are separate and even sometimes thorough competitors. As should have become clear till now this is a very bad approach. Development and operations are two sides of the same coin.

*Security is everybody's problem !*

The operational level security measures can be seen as a layered system of practices.

## Principles of security operation

- Harden the network

## Principles of security operation

- Harden the network
  - Allow only used services

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log
- Secure the OS

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log
- Secure the OS
  - Start with a secure installation

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log
- Secure the OS
  - Start with a secure installation
  - Use good file access controlling

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log
- Secure the OS
  - Start with a secure installation
  - Use good file access controlling
  - Allow only used services

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log
- Secure the OS
  - Start with a secure installation
  - Use good file access controlling
  - Allow only used services
  - Remove unused stuff

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log
- Secure the OS
  - Start with a secure installation
  - Use good file access controlling
  - Allow only used services
  - Remove unused stuff
  - Patch

## Principles of security operation

- Harden the network
  - Allow only used services
  - Use secure protocols
  - Compartmentalize the network
  - Monitor unauthorized traffic
  - Deploy multiple layers of defense
  - Log
- Secure the OS
  - Start with a secure installation
  - Use good file access controlling
  - Allow only used services
  - Remove unused stuff
  - Patch
  - Log

## Principles of security operation

- Deploy carefully

## Principles of security operation

- Deploy carefully
  - Consider file access controls

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date
  - Manage users and accounts

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date
  - Manage users and accounts
  - Treat temporary staff appropriately

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date
  - Manage users and accounts
  - Treat temporary staff appropriately
  - Test configurations

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date
  - Manage users and accounts
  - Treat temporary staff appropriately
  - Test configurations
  - Set up checks and balances

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date
  - Manage users and accounts
  - Treat temporary staff appropriately
  - Test configurations
  - Set up checks and balances
  - Do backups, securely !

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date
  - Manage users and accounts
  - Treat temporary staff appropriately
  - Test configurations
  - Set up checks and balances
  - Do backups, securely !
  - Keep incident response plan (ready)

## Principles of security operation

- Deploy carefully
  - Consider file access controls
  - If feasible, use compartmentalization
  - Switch event logging on
  - Apply same standards to third-party code
- Define sound operations practices
  - Manage privileges
  - Conduct operations tasks securely
  - Manage configurations
  - Keep patches up to date
  - Manage users and accounts
  - Treat temporary staff appropriately
  - Test configurations
  - Set up checks and balances
  - Do backups, securely !
  - Keep incident response plan (ready)
- Finally ...

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)
- AppArmor (Unix discretionary access control (DAC) model by providing mandatory access control (MAC))

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)
- AppArmor (Unix discretionary access control (DAC) model by providing mandatory access control (MAC))
- Runkit\_Sandbox (php sandboxing)

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)
- AppArmor (Unix discretionary access control (DAC) model by providing mandatory access control (MAC))
- Runkit\_Sandbox (php sandboxing)
- RATS (scans C, C++, Perl, Python and PHP for common security flaws)

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)
- AppArmor (Unix discretionary access control (DAC) model by providing mandatory access control (MAC))
- Runkit\_Sandbox (php sandboxing)
- RATS (scans C, C++, Perl, Python and PHP for common security flaws)
- PHPUnit

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)
- AppArmor (Unix discretionary access control (DAC) model by providing mandatory access control (MAC))
- Runkit\_Sandbox (php sandboxing)
- RATS (scans C, C++, Perl, Python and PHP for common security flaws)
- PHPUnit
- Splint (scans C for vulnerabilities and program mistakes)

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)
- AppArmor (Unix discretionary access control (DAC) model by providing mandatory access control (MAC))
- Runkit\_Sandbox (php sandboxing)
- RATS (scans C, C++, Perl, Python and PHP for common security flaws)
- PHPUnit
- Splint (scans C for vulnerabilities and program mistakes)
- Top 20 web vulnerability scanners

# Testing

About automation and testing, some useful automation tools to test the finalized applications.

- Libsafe (prevents BOs during execution)
- AppArmor (Unix discretionary access control (DAC) model by providing mandatory access control (MAC))
- Runkit\_Sandbox (php sandboxing)
- RATS (scans C, C++, Perl, Python and PHP for common security flaws)
- PHPUnit
- Splint (scans C for vulnerabilities and program mistakes)
- Top 20 web vulnerability scanners
- Top 125 network security tools

## Recent web application vulnerabilities

Platform: Web Application - Cross Site Scripting

## Recent web application vulnerabilities

**Platform:** Web Application - Cross Site Scripting

**Title:** epsi BIM Multiple Cross-Site Scripting Vulnerabilities

## Recent web application vulnerabilities

**Platform:** Web Application - Cross Site Scripting

**Title:** epesi BIM Multiple Cross-Site Scripting Vulnerabilities

**Description:** epesi BIM is a PHP-based application for creating dynamic Web applications. The application is exposed to multiple cross-site scripting issues because it fails to properly sanitize user-supplied input. epesi BIM 1.2.0 rev 8154 is vulnerable; prior versions may also be affected.

## Recent web application vulnerabilities

**Platform:** Web Application - Cross Site Scripting

**Title:** epesi BIM Multiple Cross-Site Scripting Vulnerabilities

**Description:** epesi BIM is a PHP-based application for creating dynamic Web applications. The application is exposed to multiple cross-site scripting issues because it fails to properly sanitize user-supplied input. epesi BIM 1.2.0 rev 8154 is vulnerable; prior versions may also be affected.

**Ref:**

[https://www.htbridge.ch/advisory/multiple\\_vulnerabilities\\_in\\_epesi](https://www.htbridge.ch/advisory/multiple_vulnerabilities_in_epesi)

CVE: CVE-2011-3990

CVE: CVE-2011-3990

Platform: Web Application - Cross Site Scripting

CVE: CVE-2011-3990

Platform: Web Application - Cross Site Scripting

Title: PukiWiki Plus! Cross-Site Scripting

**CVE:** CVE-2011-3990

**Platform:** Web Application - Cross Site Scripting

**Title:** PukiWiki Plus! Cross-Site Scripting

**Description:** PukiWiki Plus! is an application which provides wiki functionality to websites. PukiWiki Plus! is exposed to a cross-site scripting issue because it fails to properly sanitize web form entries. PukiWiki Plus! 1.4.7plus-u2-i18n and prior versions are affected.

**CVE:** CVE-2011-3990

**Platform:** Web Application - Cross Site Scripting

**Title:** PukiWiki Plus! Cross-Site Scripting

**Description:** PukiWiki Plus! is an application which provides wiki functionality to websites. PukiWiki Plus! is exposed to a cross-site scripting issue because it fails to properly sanitize web form entries. PukiWiki Plus! 1.4.7plus-u2-i18n and prior versions are affected.

**Ref:** <http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3990>

<http://www.securityfocus.com/bid/51173/info>

CVE: CVE-2011-3839,CVE-2011-3838,CVE-2011-3837,CVE-2011-3836,CVE-2011-3835

**CVE:** CVE-2011-3839,CVE-2011-3838,CVE-2011-3837,CVE-2011-3836,CVE-2011-3835

**Platform:** Web Application

**CVE:** CVE-2011-3839,CVE-2011-3838,CVE-2011-3837,CVE-2011-3836,CVE-2011-3835

**Platform:** Web Application

**Title:** Wuzly Multiple Security Vulnerabilities

**CVE:** CVE-2011-3839,CVE-2011-3838,CVE-2011-3837,CVE-2011-3836,CVE-2011-3835

**Platform:** Web Application

**Title:** Wuzly Multiple Security Vulnerabilities

**Description:** Wuzly is a PHP-based blog application. Wuzly is exposed to the multiple remote security issues. See reference for further details. Wuzly version 2.0 is affected; other versions may also be affected.

**CVE:** CVE-2011-3839,CVE-2011-3838,CVE-2011-3837,CVE-2011-3836,CVE-2011-3835

**Platform:** Web Application

**Title:** Wuzly Multiple Security Vulnerabilities

**Description:** Wuzly is a PHP-based blog application. Wuzly is exposed to the multiple remote security issues. See reference for further details. Wuzly version 2.0 is affected; other versions may also be affected.

**Ref:** [https://secunia.com/secunia\\_research/2011-88/](https://secunia.com/secunia_research/2011-88/)  
[https://secunia.com/secunia\\_research/2011-89/](https://secunia.com/secunia_research/2011-89/)

Platform: Web Application

Platform: Web Application

Title: OBM Multiple Remote Vulnerabilities

**Platform:** Web Application

**Title:** OBM Multiple Remote Vulnerabilities

**Description:** OBM is a messaging and collaboration application. The application is exposed to multiple remote issues. 1) A local file-include issue affects the "module" parameter of the "exportcsv\_index.php" script. 2) Multiple SQL injection issues. 3) Multiple cross-site scripting issues. 4) An insecure file permissions issue occurs because "test.php" is stored with insecure permissions. OBM 2.4.0-rc13 is vulnerable; other versions may also be affected.

**Platform:** Web Application

**Title:** OBM Multiple Remote Vulnerabilities

**Description:** OBM is a messaging and collaboration application. The application is exposed to multiple remote issues. 1) A local file-include issue affects the "module" parameter of the "exportcsv\_index.php" script. 2) Multiple SQL injection issues. 3) Multiple cross-site scripting issues. 4) An insecure file permissions issue occurs because "test.php" is stored with insecure permissions. OBM 2.4.0-rc13 is vulnerable; other versions may also be affected.

**Ref:** <http://www.securityfocus.com/archive/1/520986>

CVE: CVE-2011-4782

CVE: CVE-2011-4782

Platform: Web Application

CVE: CVE-2011-4782

Platform: Web Application

Title: PhpMyAdmin "\$host" Variable HTML Injection

**CVE:** CVE-2011-4782

**Platform:** Web Application

**Title:** PhpMyAdmin "\$host" Variable HTML Injection

**Description:** phpMyAdmin is a web-based administration interface for MySQL databases; it is implemented in PHP. The application is exposed to an HTML injection issue because it fails to properly sanitize user-supplied input to the "\$host" variable. phpMyAdmin versions 3.4.x prior to 3.4.9 are affected.

**CVE:** CVE-2011-4782

**Platform:** Web Application

**Title:** PhpMyAdmin "\$host" Variable HTML Injection

**Description:** phpMyAdmin is a web-based administration interface for MySQL databases; it is implemented in PHP. The application is exposed to an HTML injection issue because it fails to properly sanitize user-supplied input to the "\$host" variable. phpMyAdmin versions 3.4.x prior to 3.4.9 are affected.

**Ref:** [http://www.phpmyadmin.net/home\\_page/security/PMASA-2011-19.php](http://www.phpmyadmin.net/home_page/security/PMASA-2011-19.php)

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyk

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyk
- Code Complete - second edition, S. McConnell

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler
- Security Attribute Evaluation Method: A Cost Benefit Approach (SAEM)

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyjk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler
- Security Attribute Evaluation Method: A Cost Benefit Approach (SAEM)
- CERT-CC Top 10 Secure Coding Practices

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyjk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler
- Security Attribute Evaluation Method: A Cost Benefit Approach (SAEM)
- CERT-CC Top 10 Secure Coding Practices
- SANS Top Cyber Security Risks

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyjk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler
- Security Attribute Evaluation Method: A Cost Benefit Approach (SAEM)
- CERT-CC Top 10 Secure Coding Practices
- SANS Top Cyber Security Risks
- OWASP 2010 Top 10 Application Security Risks

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyjk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler
- Security Attribute Evaluation Method: A Cost Benefit Approach (SAEM)
- CERT-CC Top 10 Secure Coding Practices
- SANS Top Cyber Security Risks
- OWASP 2010 Top 10 Application Security Risks
- OWASP PHP Project

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyjk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler
- Security Attribute Evaluation Method: A Cost Benefit Approach (SAEM)
- CERT-CC Top 10 Secure Coding Practices
- SANS Top Cyber Security Risks
- OWASP 2010 Top 10 Application Security Risks
- OWASP PHP Project
- OWASP PHP Top 5

# Bibliographic references

- Secure Coding - Principles & Practices, M. Graff & K. van Wyjk
- Code Complete - second edition, S. McConnell
- Secure Programming for Linux and Unix HOWTO, D. Wheeler
- Security Attribute Evaluation Method: A Cost Benefit Approach (SAEM)
- CERT-CC Top 10 Secure Coding Practices
- SANS Top Cyber Security Risks
- OWASP 2010 Top 10 Application Security Risks
- OWASP PHP Project
- OWASP PHP Top 5
- BAD practices: Bastard Operator From Hell (BOFH)