



# OpenSAMM

## Software Assurance Maturity Model

Seba Deleersnyder  
[seba@owasp.org](mailto:seba@owasp.org)

OWASP Foundation Board Member  
OWASP Belgium Chapter Leader  
SAMM project co-leader



# OWASP

The Open Web Application Security Project

<http://www.owasp.org>

**OWASP is a worldwide free and open community focused on improving the security of application software.**

**Our mission is to make application security visible so that people and organizations can make informed decisions about application security risks.**

**Everyone is free to participate in OWASP and all of our materials are available under a free and open software license.**

**The OWASP Foundation is a not-for-profit charitable organization that ensures the ongoing availability and support for our work.**

# The web application security challenge

Your security "perimeter" has huge holes at the application layer

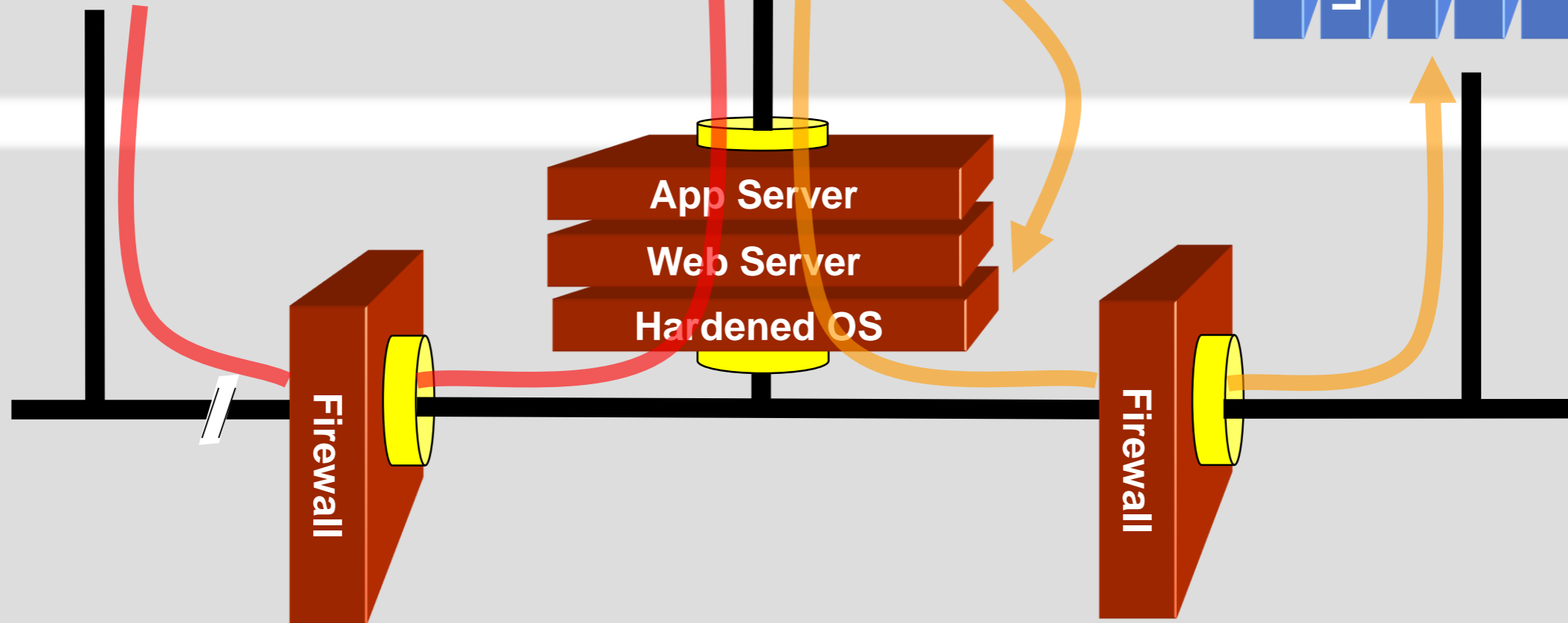
Application Layer



Custom Developed  
Application Code

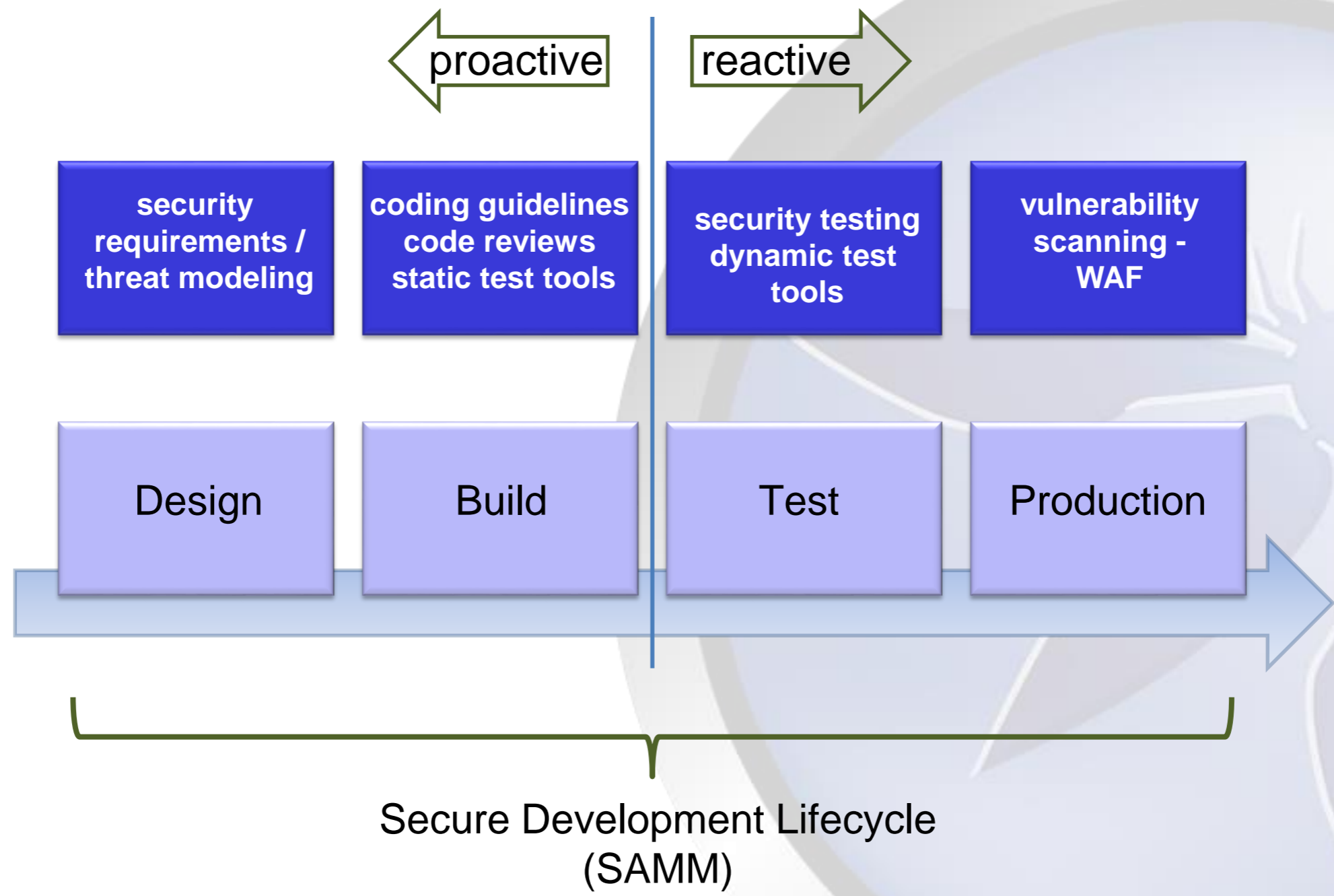


Network Layer



You can't use network layer protection (firewall, SSL, IDS, hardening) to stop or detect application layer attacks

# “Build in” software assurance



# We need a Maturity Model

An organization's behavior changes slowly over time

Changes must be iterative while working toward long-term goals

There is no single recipe that works for all organizations

A solution must enable risk-based choices tailored to the organization

Guidance related to security activities must be prescriptive

A solution must provide enough details for non-security-people

Overall, must be simple, well-defined, and measurable

OWASP Software Assurance Maturity Model (SAMM)



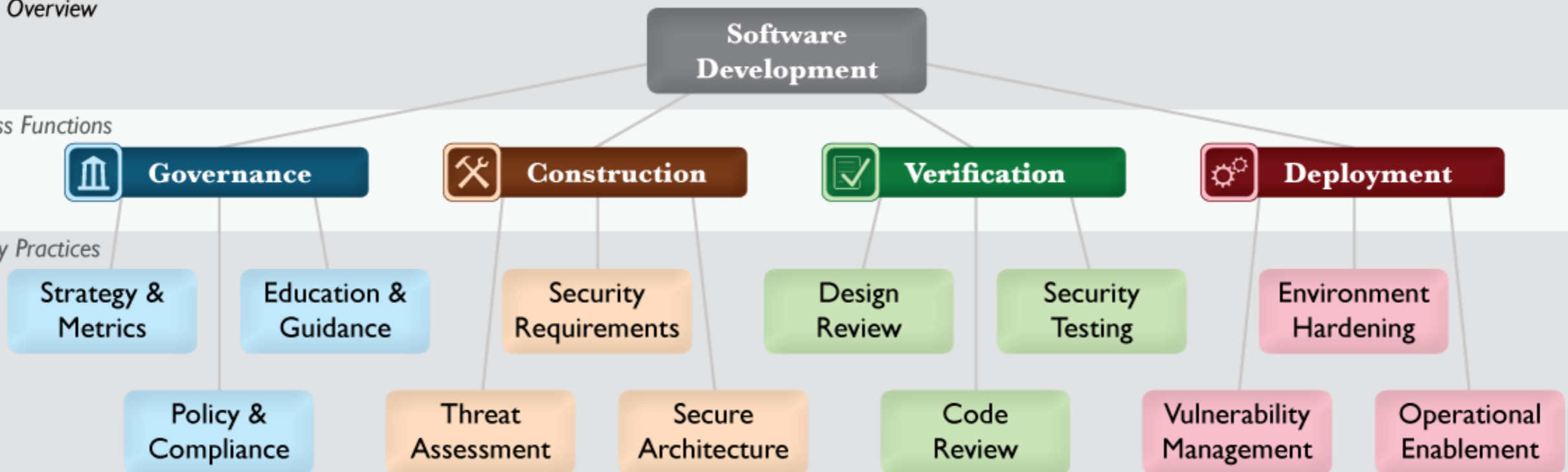
# SAMM Security Practices

- From each of the Business Functions, 3 Security Practices are defined
- The Security Practices cover all areas relevant to software security assurance
- Each one is a 'silo' for improvement

## SAMM Overview

### Business Functions

### Security Practices



# Under each Security Practice

- Three successive Objectives under each Practice define how it can be improved over time
  - This establishes a notion of a Level at which an organization fulfills a given Practice
  
- The three Levels for a Practice generally correspond to:
  - (0: Implicit starting point with the Practice unfulfilled)
  - 1: Initial understanding and ad hoc provision of the Practice
  - 2: Increase efficiency and/or effectiveness of the Practice
  - 3: Comprehensive mastery of the Practice at scale

# Per Level, SAMM defines...

- Objective
- Activities
- Results
- Success Metrics
- Costs
- Personnel
- Related Levels

Education & Guidance
EG 1

Offer development staff access to resources around the topics of secure programming and deployment

**ACTIVITIES**

**A. Conduct technical security awareness training**

Either internally or externally sourced, conduct security training for technical staff that covers the basic tenets of application security. Generally, this can be accomplished via instructor-led training in 1-2 days or via computer-based training with modules taking about the same amount of time per developer.

Course content should cover both conceptual and technical information. Appropriate topics include high-level best practices surrounding input validation, output encoding, error handling, logging, authentication, authorization. Additional coverage of commonplace software vulnerabilities is also desirable such as a Top 10 list appropriate to the software being developed (web applications, embedded devices, client-server applications, back-end transaction systems, etc.). Wherever possible, use code samples and lab exercises in the specific programming language(s) that applies.

To rollout such training, it is recommended to mandate annual security training and then hold courses (either instructor-led or computer-based) as often as required based on development head-count.

**B. Build and maintain technical guidelines**

For development staff, assemble a list of approved documents, web pages, and technical notes that provide technology-specific security advice. These references can be assembled from many publicly available resources on the Internet. In cases where very specialized or proprietary technologies permeate the development environment, utilize senior, security-savvy staff to build security notes over time to create such a knowledge base in an ad hoc fashion.

Ensure management is aware of the resources and briefs oncoming staff about their expected usage. Try to keep the guidelines lightweight and up-to-date to avoid clutter and irrelevance. Once a comfort-level has been established, they can be used as a qualitative checklist to ensure that the guidelines have been read, understood, and followed in the development process.

**RESULTS**

- ◆ Increased developer awareness on the most common problems at the code level
- ◆ Maintain software with rudimentary security best-practices in place
- ◆ Set baseline for security know-how among technical staff
- ◆ Enable qualitative security checks for baseline security knowledge

**SUCCESS METRICS**

- ◆ >50% development staff briefed on security issues within past 1 year
- ◆ >75% senior development/architect staff briefed on security issues within past 1 year
- ◆ Launch technical guidance within 3 months of first training

**COSTS**

- ◆ Training course buildout or license
- ◆ Ongoing maintenance of technical guidance

**PERSONNEL**




- ◆ Developers (1-2 days/yr)
- ◆ Architects (1-2 days/yr)

**RELATED LEVELS**




- ◆ Policy & Compliance - 2
- ◆ Security Requirements - 1
- ◆ Secure Architecture - 1






# Strategy & Metrics

Strategy & Metrics <span style="float: right;">...more on page 34</span>			
	 SM 1	 SM 2	 SM 3
<b>OBJECTIVE</b>	<b>Establish unified strategic roadmap for software security within the organization</b>	<b>Measure relative value of data and software assets and choose risk tolerance</b>	<b>Align security expenditure with relevant business indicators and asset value</b>
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Estimate overall business risk profile</li> <li>B. Build and maintain assurance program roadmap</li> </ul>	<ul style="list-style-type: none"> <li>A. Classify data and applications based on business risk</li> <li>B. Establish and measure per-classification security goals</li> </ul>	<ul style="list-style-type: none"> <li>A. Conduct periodic industry-wide cost comparisons</li> <li>B. Collect metrics for historic security spend</li> </ul>

# Policy & Compliance

<b>Policy &amp; Compliance</b> <span style="float: right;">...more on page 38</span>			
			
<b>OBJECTIVE</b>	<b>Understand relevant governance and compliance drivers to the organization</b>	<b>Establish security and compliance baseline and understand per-project risks</b>	<b>Require compliance and measure projects against organization-wide policies and standards</b>
<b>ACTIVITIES</b>	A. Identify and monitor external compliance drivers B. Build and maintain compliance guidelines	A. Build policies and standards for security and compliance B. Establish project audit practice	A. Create compliance gates for projects B. Adopt solution for audit data collection

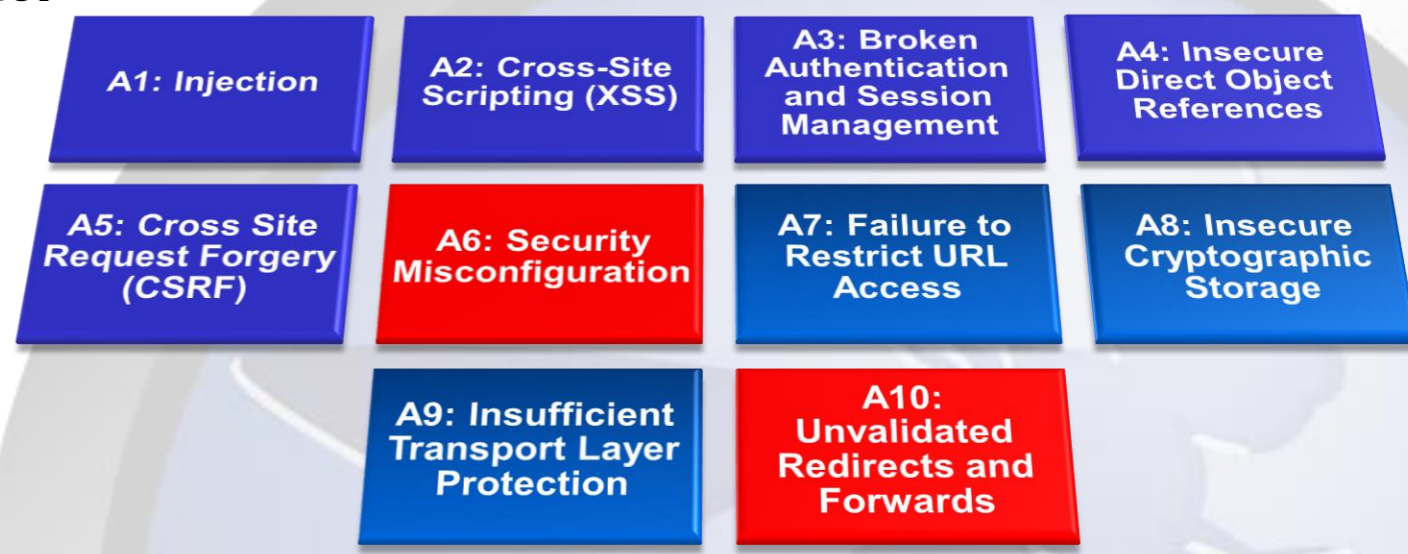
# Education & Guidance

Education & Guidance <span style="float: right;">...more on page 42</span>			
	 EG 1	 EG 2	 EG 3
<b>OBJECTIVE</b>	<b>Offer development staff access to resources around the topics of secure programming and deployment</b>	<b>Educate all personnel in the software life-cycle with role-specific guidance on secure development</b>	<b>Mandate comprehensive security training and certify personnel for baseline knowledge</b>
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Conduct technical security awareness training</li> <li>B. Build and maintain technical guidelines</li> </ul>	<ul style="list-style-type: none"> <li>A. Conduct role-specific application security training</li> <li>B. Utilize security coaches to enhance project teams</li> </ul>	<ul style="list-style-type: none"> <li>A. Create formal application security support portal</li> <li>B. Establish role-based examination/certification</li> </ul>

# Education & Guidance

Give a man a fish and you feed him for a day;  
Teach a man to fish and you feed him for a lifetime.

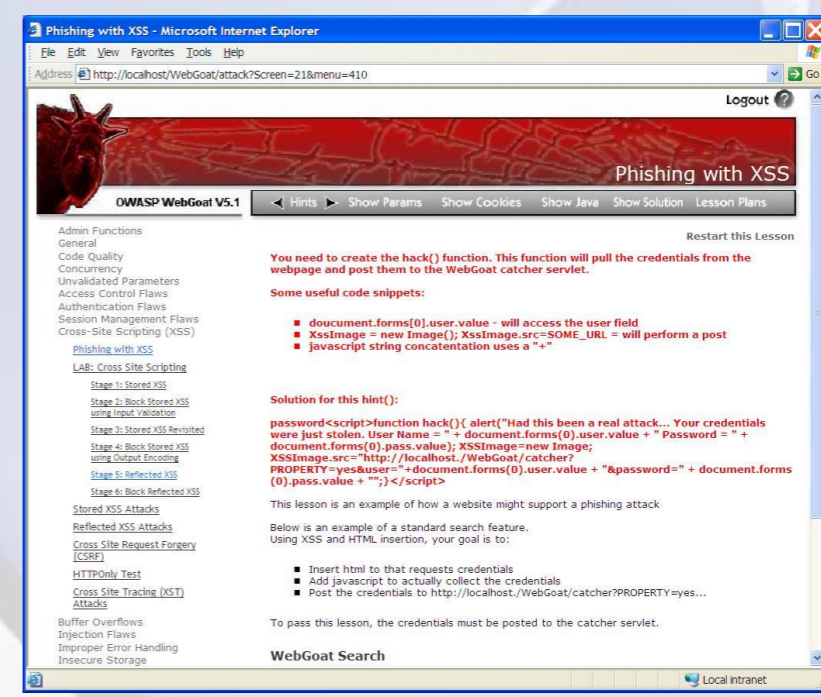
Chinese proverb



Resources:

- OWASP Top 10
- OWASP Education
- WebGoat

[https://www.owasp.org/index.php/Category:OWASP\\_Top\\_Ten\\_Project](https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project)  
[https://www.owasp.org/index.php/Category:OWASP\\_Education\\_Project](https://www.owasp.org/index.php/Category:OWASP_Education_Project)  
[https://www.owasp.org/index.php/Category:OWASP\\_WebGoat\\_Project](https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project)



# OWASP Cheat Sheets

## Developer Cheat Sheets (Builder)

- Authentication Cheat Sheet
- Choosing and Using Security Questions Cheat Sheet
- Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet
- Cryptographic Storage Cheat Sheet
- DOM based XSS Prevention Cheat Sheet
- Forgot Password Cheat Sheet
- HTML5 Security Cheat Sheet
- Input Validation Cheat Sheet
- JAAS Cheat Sheet
- Logging Cheat Sheet
- OWASP Top Ten Cheat Sheet
- Query Parameterization Cheat Sheet
- Session Management Cheat Sheet
- SQL Injection Prevention Cheat Sheet
- Transport Layer Protection Cheat Sheet
- Web Service Security Cheat Sheet
- XSS (Cross Site Scripting) Prevention Cheat Sheet
- User Privacy Protection Cheat Sheet

## Assessment Cheat Sheets (Breaker)

- Attack Surface Analysis Cheat Sheet
- XSS Filter Evasion Cheat Sheet




## Mobile Cheat Sheets

- IOS Developer Cheat Sheet
- Mobile Jailbreaking Cheat Sheet




## Draft Cheat Sheets

- Access Control Cheat Sheet
- Application Security Architecture Cheat Sheet
- Clickjacking Cheat Sheet
- Password Storage Cheat Sheet
- PHP Security Cheat Sheet
- REST Security Cheat Sheet
- Secure Coding Cheat Sheet
- Threat Modeling Cheat Sheet
- Virtual Patching Cheat Sheet
- Web Application Security Testing Cheat Sheet

# Threat Assessment

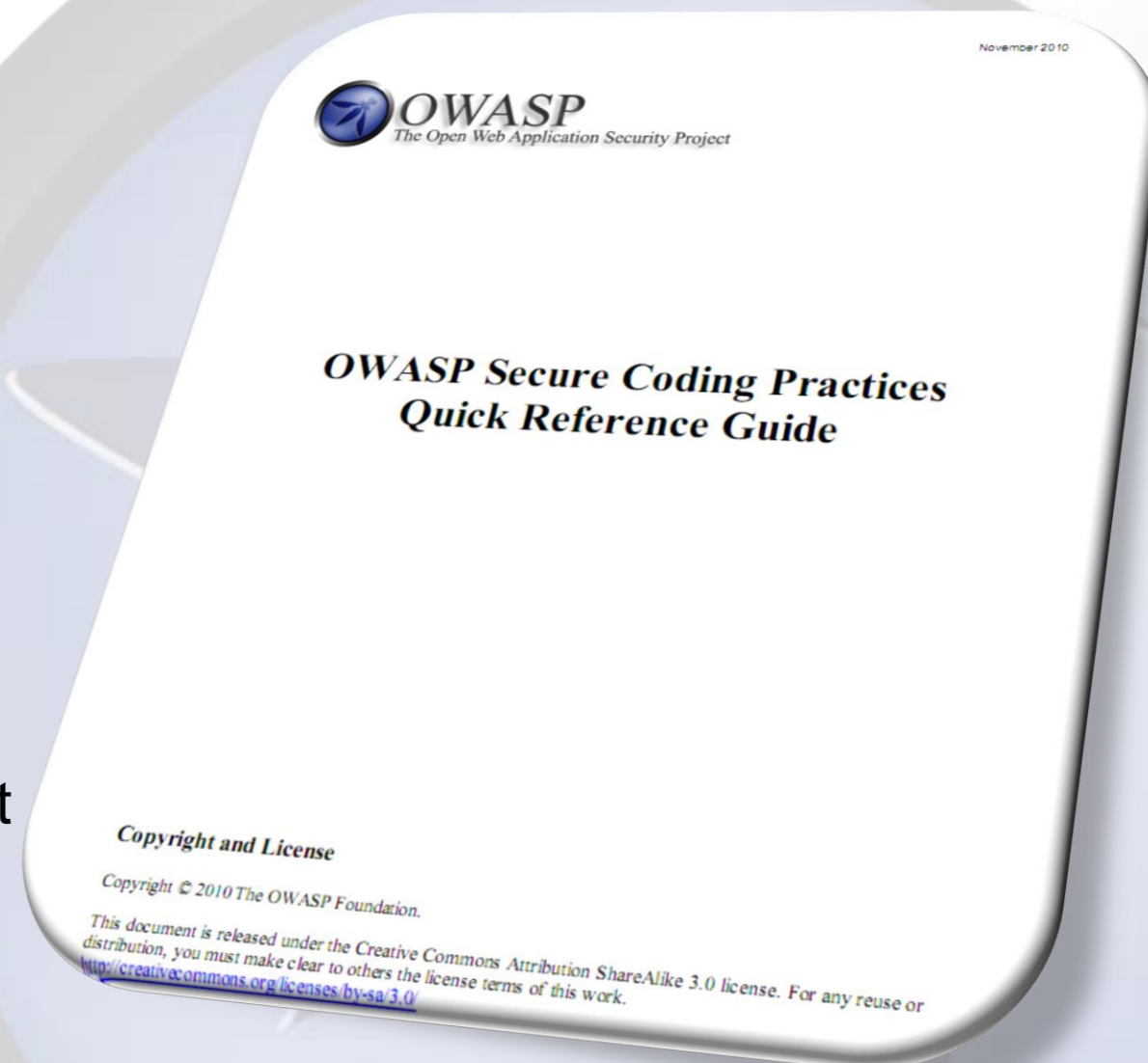
Threat Assessment <span style="float: right;">...more on page 46</span>			
	 TA 1	 TA 2	 TA 3
<b>OBJECTIVE</b>	Identify and understand high-level threats to the organization and individual projects	Increase accuracy of threat assessment and improve granularity of per-project understanding	Concretely tie compensating controls to each threat against internal and third-party software
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Build and maintain application-specific threat models</li> <li>B. Develop attacker profile from software architecture</li> </ul>	<ul style="list-style-type: none"> <li>A. Build and maintain abuse-case models per project</li> <li>B. Adopt a weighting system for measurement of threats</li> </ul>	<ul style="list-style-type: none"> <li>A. Explicitly evaluate risk from third-party components</li> <li>B. Elaborate threat models with compensating controls</li> </ul>

# Security Requirements

Security Requirements <span style="float: right;">...more on page 50</span>			
	 <b>SR 1</b>	 <b>SR 2</b>	 <b>SR 3</b>
<b>OBJECTIVE</b>	Consider security explicitly during the software requirements process	Increase granularity of security requirements derived from business logic and known risks	Mandate security requirements process for all software projects and third-party dependencies
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Derive security requirements from business functionality</li> <li>B. Evaluate security and compliance guidance for requirements</li> </ul>	<ul style="list-style-type: none"> <li>A. Build an access control matrix for resources and capabilities</li> <li>B. Specify security requirements based on known risks</li> </ul>	<ul style="list-style-type: none"> <li>A. Build security requirements into supplier agreements</li> <li>B. Expand audit program for security requirements</li> </ul>




# Secure Coding Practices Quick Reference Guide

- Technology agnostic coding practices
- What to do, not how to do it
- Compact, but comprehensive checklist format
- Focuses on secure coding requirements, rather than on vulnerabilities and exploits
- Includes a cross referenced glossary to get developers and security folks talking the same language

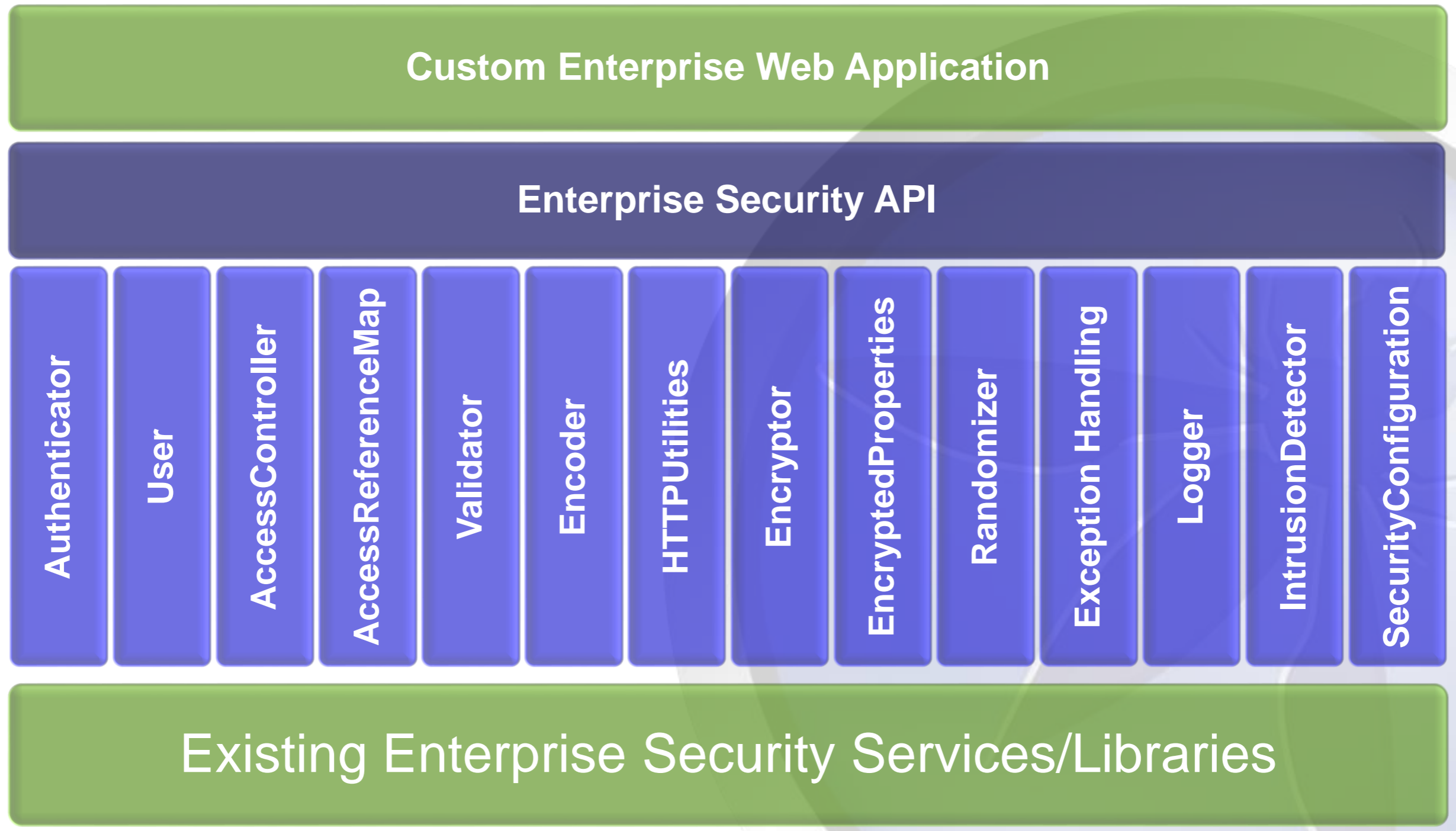







# Secure Architecture

Secure Architecture <span style="float: right;">...more on page 54</span>			
	 SA 1	 SA 2	 SA 3
<b>OBJECTIVE</b>	<b>Insert consideration of proactive security guidance into the software design process</b>	<b>Direct the software design process toward known-secure services and secure-by-default designs</b>	<b>Formally control the software design process and validate utilization of secure components</b>
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Maintain list of recommended software frameworks</li> <li>B. Explicitly apply security principles to design</li> </ul>	<ul style="list-style-type: none"> <li>A. Identify and promote security services and infrastructure</li> <li>B. Identify security design patterns from architecture</li> </ul>	<ul style="list-style-type: none"> <li>A. Establish formal reference architectures and platforms</li> <li>B. Validate usage of frameworks, patterns, and platforms</li> </ul>




# The OWASP Enterprise Security API



# Design Review

Design Review <span style="float: right;">...more on page 58</span>			
	 DR 1	 DR 2	 DR 3
<b>OBJECTIVE</b>	Support ad hoc reviews of software design to ensure baseline mitigations for known risks	Offer assessment services to review software design against comprehensive best practices for security	Require assessments and validate artifacts to develop detailed understanding of protection mechanisms
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Identify software attack surface</li> <li>B. Analyze design against known security requirements</li> </ul>	<ul style="list-style-type: none"> <li>A. Inspect for complete provision of security mechanisms</li> <li>B. Deploy design review service for project teams</li> </ul>	<ul style="list-style-type: none"> <li>A. Develop data-flow diagrams for sensitive resources</li> <li>B. Establish release gates for design review</li> </ul>

# Code Review

Code Review <span style="float: right;">...more on page 62</span>			
	 CR 1	 CR 2	 CR 3
<b>OBJECTIVE</b>	<b>Opportunistically find basic code-level vulnerabilities and other high-risk security issues</b>	<b>Make code review during development more accurate and efficient through automation</b>	<b>Mandate comprehensive code review process to discover language-level and application-specific risks</b>
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Create review checklists from known security requirements</li> <li>B. Perform point-review of high-risk code</li> </ul>	<ul style="list-style-type: none"> <li>A. Utilize automated code analysis tools</li> <li>B. Integrate code analysis into development process</li> </ul>	<ul style="list-style-type: none"> <li>A. Customize code analysis for application-specific concerns</li> <li>B. Establish release gates for code review</li> </ul>

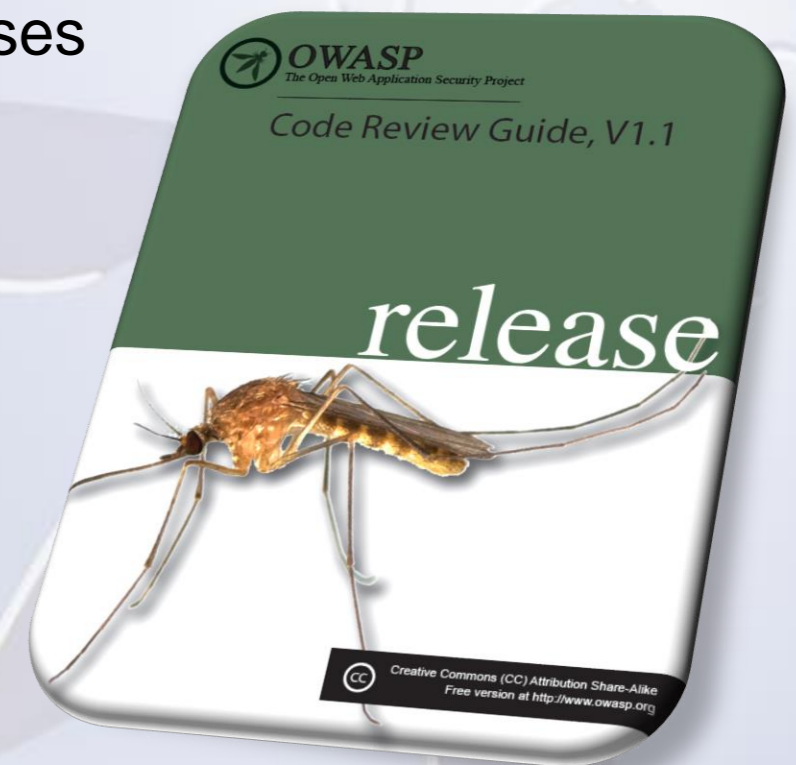
# Code Review

## SDL Integration:

- Multiple reviews defined as deliverables in your SDLC
- Structured, repeatable process with management support
- Reviews are exit criteria for the development and test phases

## Resources:

- OWASP Code Review Guide



# Code review tooling

## Code review tools:

- OWASP LAPSE (Security scanner for Java EE Applications)
- MS FxCop / CAT.NET (Code Analysis Tool for .NET)
- Agnitio (open source Manual source code review support tool)

```

Resource - testVulnerabilities/src/Test4.java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help

Test4.java
connection = DriverManager.getConnection(DataURL, LOGIN,
PASSWORD);
} catch (SQLException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}

String Username = request.getParameter("USER"); // From HTTP request
String Password = request.getParameter("PASSWORD"); // From HTTP request
int iUserID = -1;
String sloggedUser = "";

String sel = "SELECT User_id, Username FROM USERS WHERE Username = "
+ Username + " AND Password = " + Password + " ";

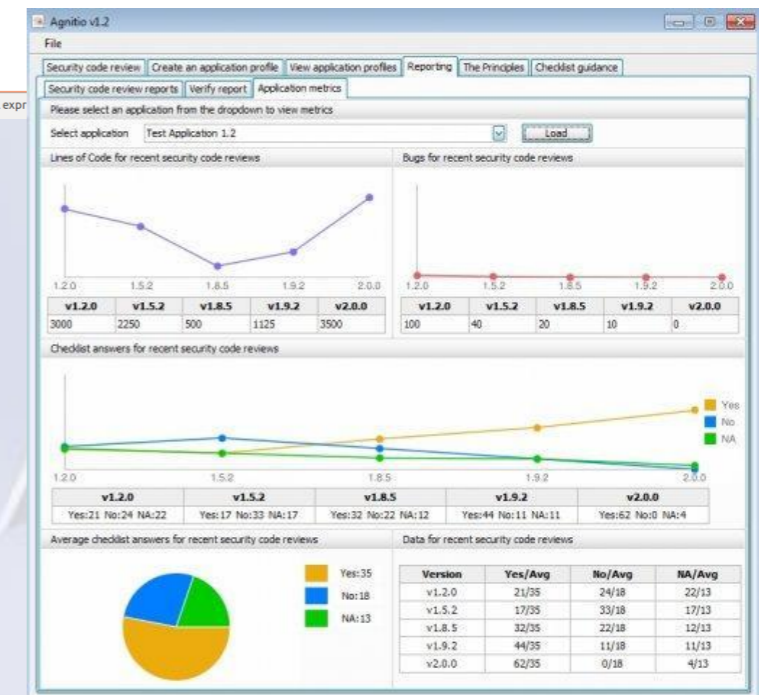
Statement selectStatement = null;
try {
selectStatement = (Statement) connection.createStatement();
} catch (SQLException e) {

```




Provenance Tracker Vulnerability Sinks Vulnerability Sources

Created a slice with 5 leaf element(s) and 5 element(s) located in 1 file(s) with 0 element(s) truncated with a maximum depth of 1. Using a flat viewer.

- "SELECT User\_id, Username FROM USERS WHERE Username = " (Test4.java:65) [string constant]
- request.getParameter("USER") (Test4.java:57) [call expression]
- " AND Password = " (Test4.java:66) [string constant]
- request.getParameter("PASSWORD") (Test4.java:59) [call expression]
- "" (Test4.java:66) [string constant]



# Security Testing

Security Testing <span style="float: right;">...more on page 66</span>			
	 ST 1	 ST 2	 ST 3
<b>OBJECTIVE</b>	Establish process to perform basic security tests based on implementation and software requirements	Make security testing during development more complete and efficient through automation	Require application-specific security testing to ensure baseline security before deployment
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Derive test cases from known security requirements</li> <li>B. Conduct penetration testing on software releases</li> </ul>	<ul style="list-style-type: none"> <li>A. Utilize automated security testing tools</li> <li>B. Integrate security testing into development process</li> </ul>	<ul style="list-style-type: none"> <li>A. Employ application-specific security testing automation</li> <li>B. Establish release gates for security testing</li> </ul>

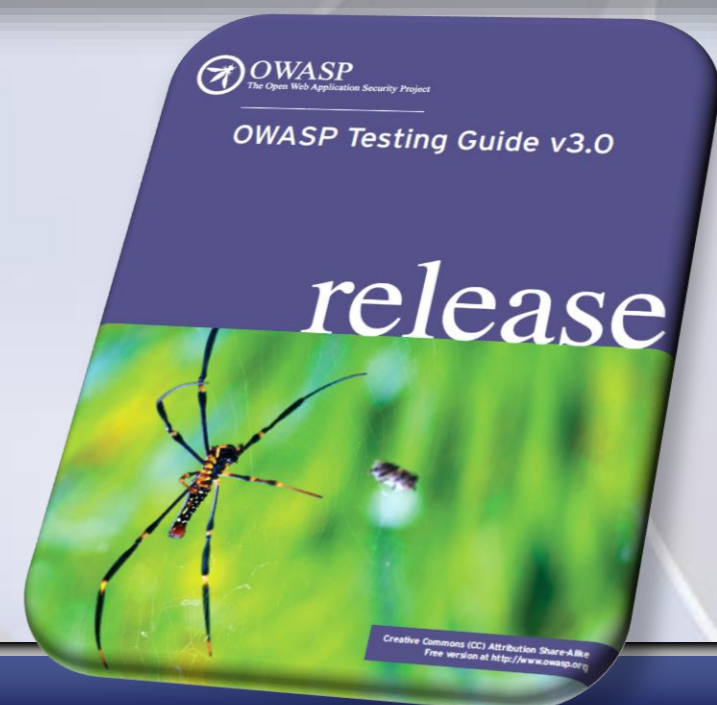
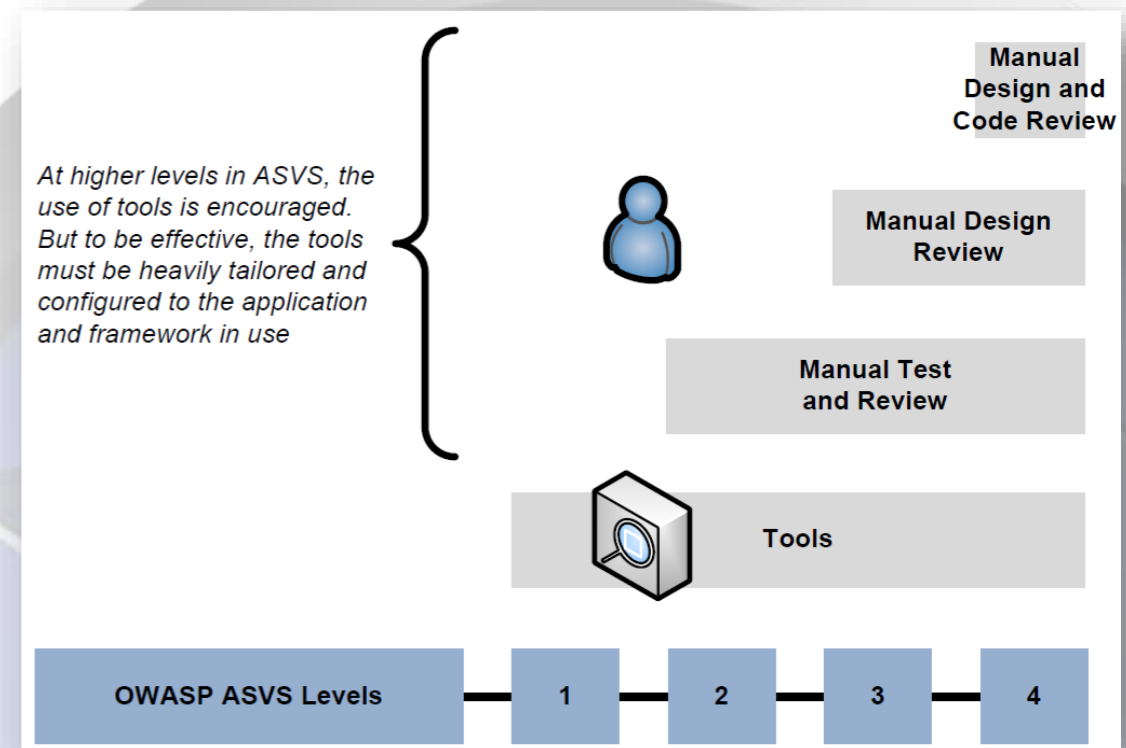
# Security Testing

## SDL Integration:

- Integrate dynamic security testing as part of you test cycles
- Derive test cases from the security requirements that apply
- Check business logic soundness as well as common vulnerabilities
- Review results with stakeholders prior to release

## Resources:

- OWASP ASVS
- OWASP Testing Guide





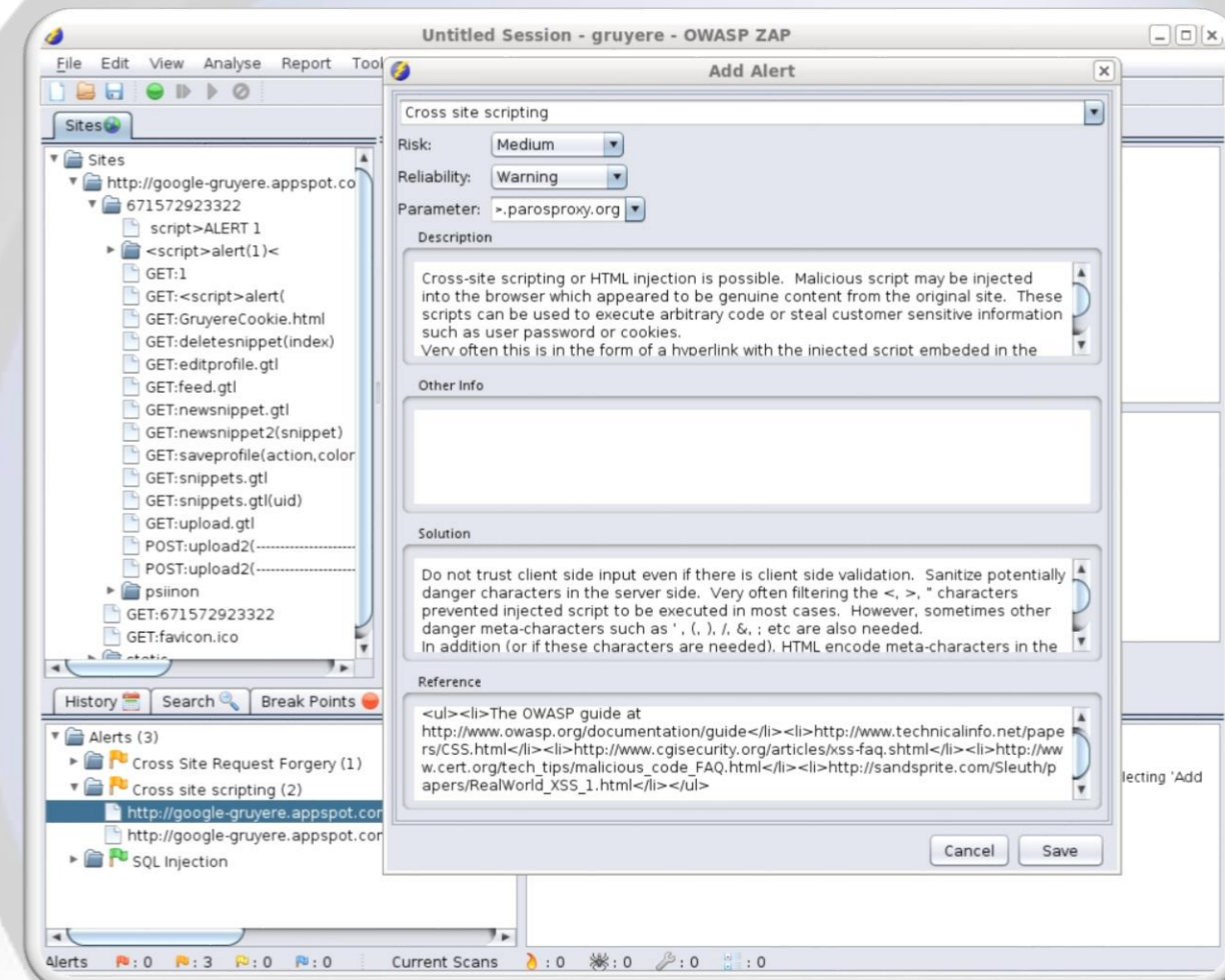


# Security Testing




- Zed Attack Proxy (ZAP) is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications
- Provides automated scanners as well as a set of tools that allow you to find security vulnerabilities manually

## Features:




- Intercepting proxy
- Automated scanner
- Passive scanner
- Brute force scanner
- Spider
- Fuzzer
- Port scanner
- Dynamic SSL Certificates
- API
- Beanshell integration



# Vulnerability Management

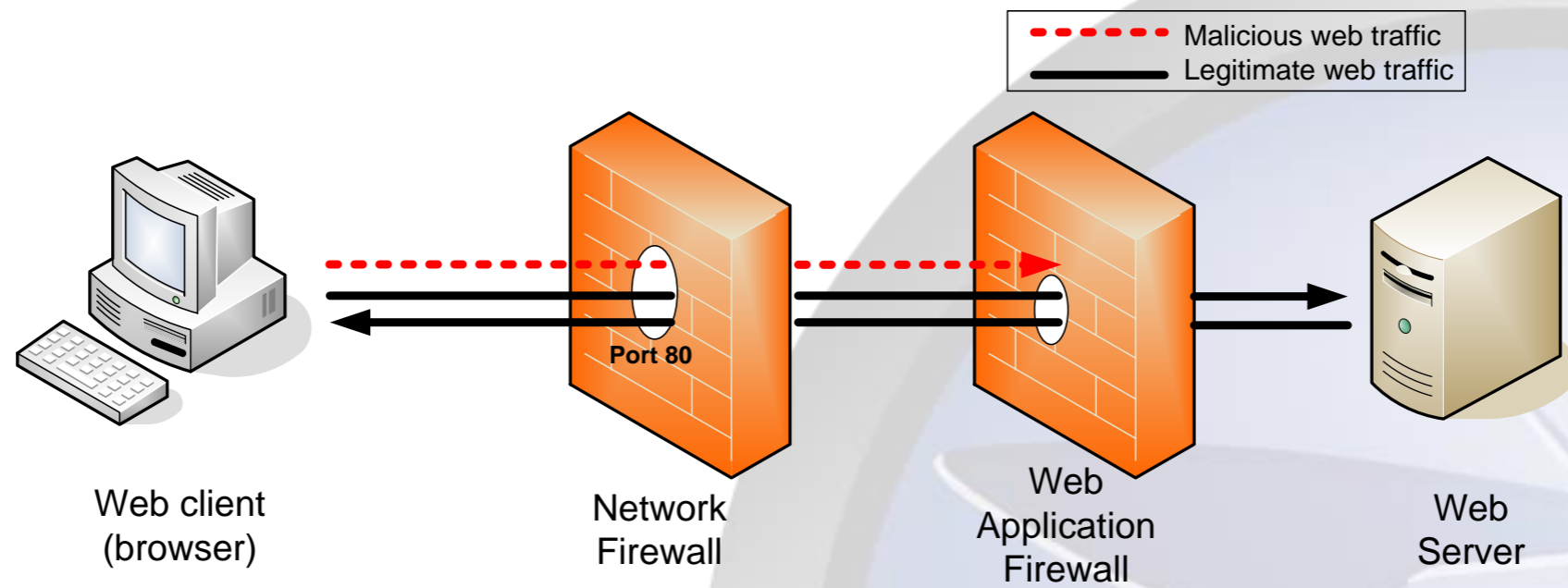
Vulnerability Management <span style="float: right;">...more on page 70</span>			
			
<b>OBJECTIVE</b>	Understand high-level plan for responding to vulnerability reports or incidents	Elaborate expectations for response process to improve consistency and communications	Improve analysis and data gathering within response process for feedback into proactive planning
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Identify point of contact for security issues</li> <li>B. Create informal security response team(s)</li> </ul>	<ul style="list-style-type: none"> <li>A. Establish consistent incident response process</li> <li>B. Adopt a security issue disclosure process</li> </ul>	<ul style="list-style-type: none"> <li>A. Conduct root cause analysis for incidents</li> <li>B. Collect per-incident metrics</li> </ul>

# Environment Hardening

Environment Hardening <span style="float: right;">...more on page 74</span>			
	 EH 1	 EH 2	 EH 3
<b>OBJECTIVE</b>	<b>Understand baseline operational environment for applications and software components</b>	<b>Improve confidence in application operations by hardening the operating environment</b>	<b>Validate application health and status of operational environment against known best practices</b>
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Maintain operational environment specification</li> <li>B. Identify and install critical security upgrades and patches</li> </ul>	<ul style="list-style-type: none"> <li>A. Establish routine patch management process</li> <li>B. Monitor baseline environment configuration status</li> </ul>	<ul style="list-style-type: none"> <li>A. Identify and deploy relevant operations protection tools</li> <li>B. Expand audit program for environment configuration</li> </ul>



# Web Application Firewalls



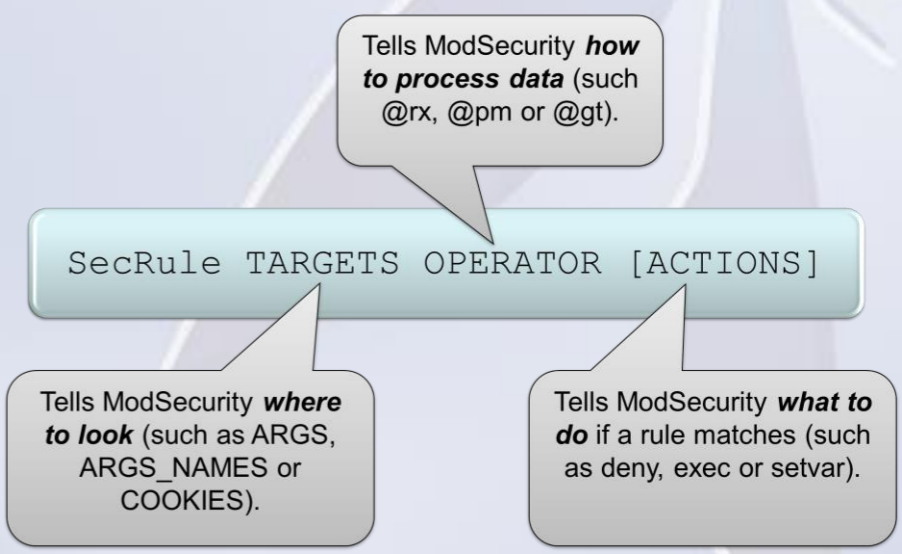
ModSecurity: Worlds No 1 open source Web Application Firewall

[www.modsecurity.org](http://www.modsecurity.org)




- HTTP Traffic Logging
- Real-Time Monitoring and Attack Detection
- Attack Prevention and Just-in-time Patching
- Flexible Rule Engine
- Embedded Deployment (Apache, IIS7 and Nginx)
- Network-Based Deployment (reverse proxy)

OWASP ModSecurity **Core Rule Set Project**, generic, plug-n-play set of WAF rules

## ModSecurity's Rules Language Syntax



# Operational Enablement

Operational Enablement <span style="float: right;">...more on page 78</span>			
	 OE 1	 OE 2	 OE 3
<b>OBJECTIVE</b>	Enable communications between development teams and operators for critical security-relevant data	Improve expectations for continuous secure operations through provision of detailed procedures	Mandate communication of security information and validate artifacts for completeness
<b>ACTIVITIES</b>	<ul style="list-style-type: none"> <li>A. Capture critical security information for deployment</li> <li>B. Document procedures for typical application alerts</li> </ul>	<ul style="list-style-type: none"> <li>A. Create per-release change management procedures</li> <li>B. Maintain formal operational security guides</li> </ul>	<ul style="list-style-type: none"> <li>A. Expand audit program for operational information</li> <li>B. Perform code signing for application components</li> </ul>

# 150+ OWASP Projects

## PROTECT

Tools: AntiSamy Java/.NET, Enterprise Security API (ESAPI), ModSecurity Core Rule Set Project

Docs: Development Guide, .NET, Ruby on Rails Security Guide, Secure Coding Practices - Quick Reference Guide

## DETECT

Tools: JBroFuzz, Lice CD, WebScarab, Zed Attack Proxy

Docs: Application Security Verification Standard, Code Review Guide, Testing Guide, Top Ten Project

## LIFE CYCLE

SAMM, WebGoat, Legal Project

# Get started

Step 1:  
questionnaire  
as-is

Step 2: define  
your maturity  
goal

Step 3: define  
phased  
roadmap

# Conducting assessments

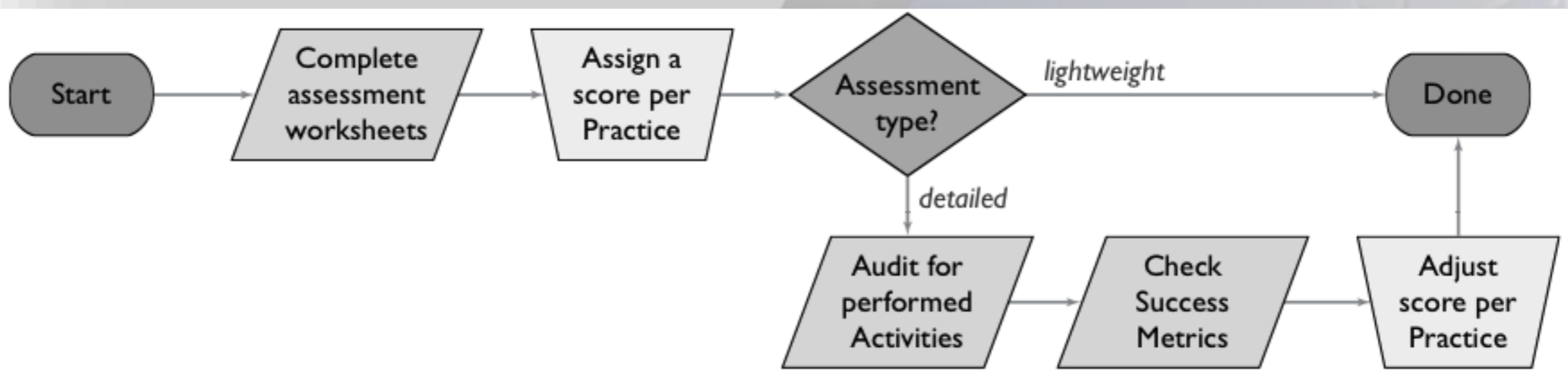
SAMM includes assessment worksheets for each Security Practice

Education & Guidance	Yes/No
◆ Have most developers been given high-level security awareness training?	
◆ Does each project team have access to secure development best practices and guidance?	
◆ Are most roles in the development process given role-specific training and guidance?	
◆ Are most stakeholders able to pull in security coaches for use on projects?	
◆ Is security-related guidance centrally controlled and consistently distributed throughout the organization?	
◆ Are most people tested to ensure a baseline skill-set for secure development practices?	



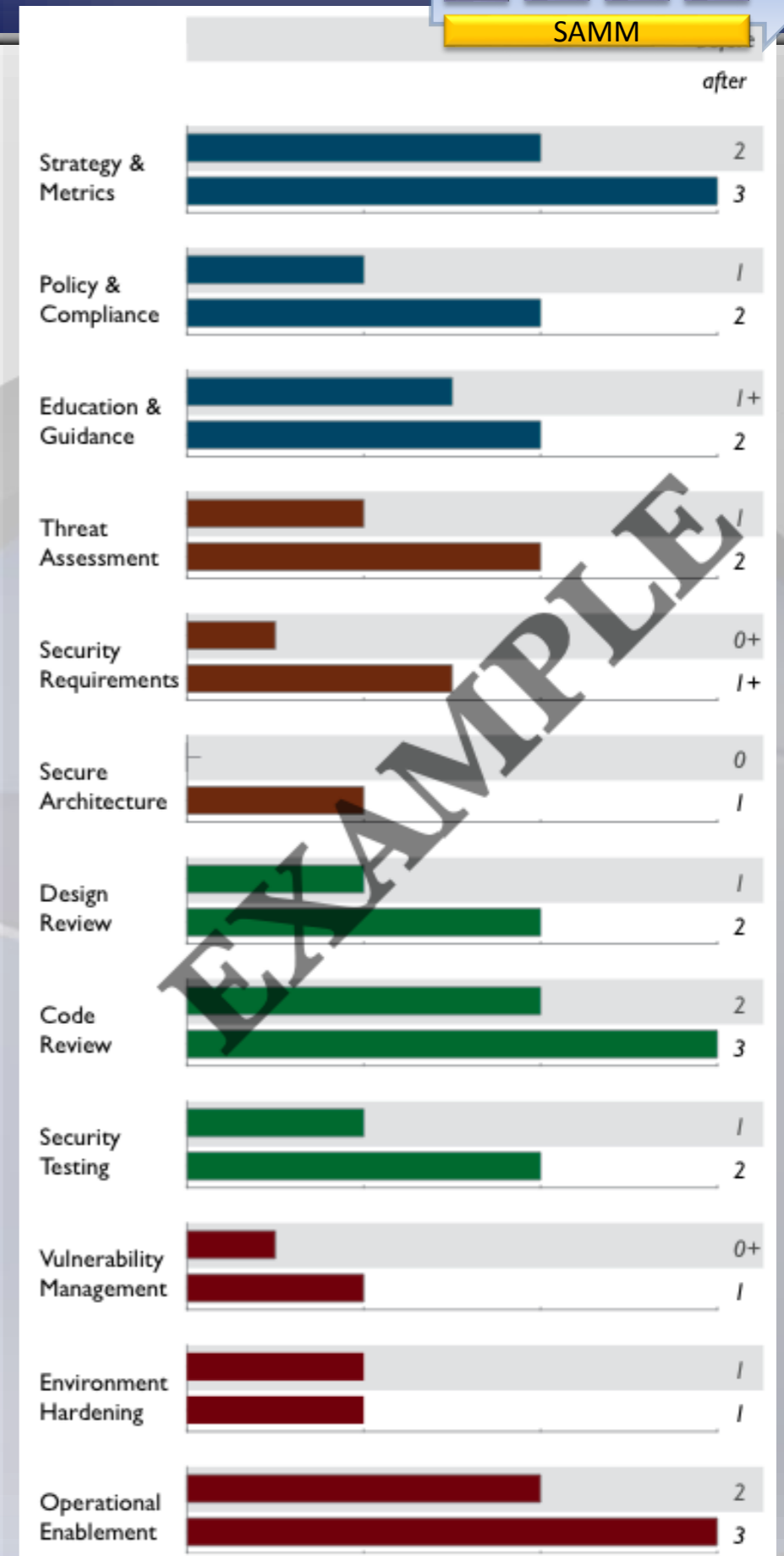
# Assessment process

Supports both lightweight and detailed assessments



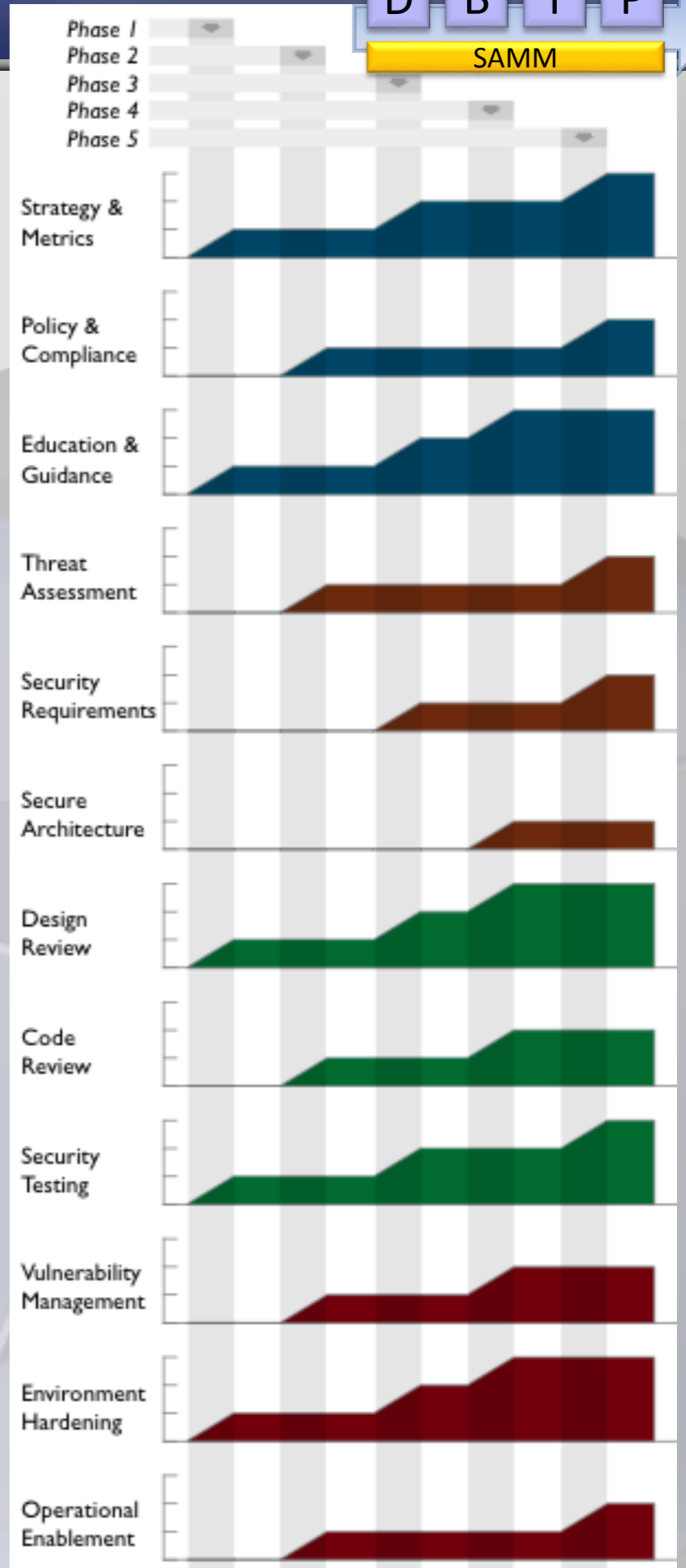
# Creating Scorecards

- Gap analysis
  - Capturing scores from detailed assessments versus expected performance levels
- Demonstrating improvement
  - Capturing scores from before and after an iteration of assurance program build-out
- Ongoing measurement
  - Capturing scores over consistent time frames for an assurance program that is already in place



# Roadmap templates

- To make the “building blocks” usable, SAMM defines Roadmaps templates for typical kinds of organizations
  - Independent Software Vendors
  - Online Service Providers
  - Financial Services Organizations
  - Government Organizations
- Tune these to your own targets / speed



# SAMM Resources

[www.opensamm.org](http://www.opensamm.org)

- Presentations
- Tools
  - Assessment worksheets / templates
  - Roadmap templates
  - Scorecard chart generation
- Translations (Spanish / Japanese)
- SAMM mappings to ISO/EIC 27034 / BSIMM

# Critical Success Factors

- Get initiative buy-in from all stakeholders
- Adopt a risk-based approach
- Awareness / education is the foundation
- Integrate security in your development / acquisition and deployment processes
- Provide management visibility

# Project Roadmap

Build the SAMM community:

- List of SAMM adopters
- Workshops at AppSecEU and AppSecUSA

V1.1:

- Incorporate tools / guidance / OWASP projects
- Revamp SAMM wiki

V2.0:

- Revise scoring model
- Model revision necessary ? (12 practices, 3 levels, ...)
- Application to agile
- Roadmap planning: how to measure effort ?
- Presentations & teaching material
- ...

# Get involved

- Use and donate back!
- Attend OWASP chapter meetings and conferences
- Support OWASP become personal/company member  
<https://www.owasp.org/index.php/Membership>





Q&A







**Global AppSec EMEA 2013**  
**Aug. 20, 2013 - Aug. 23, 2013**  
**Hamburg, Germany**



# BeNeLux 2013

- 28-29 november 2013
- One day of trainings
- One day conference
- The Netherlands - Amsterdam

# Thank you

- @sebadele
- seba@owasp.org
- seba@deleersnyder.eu
- [www.linkedin.com/in/sebadele](http://www.linkedin.com/in/sebadele)

