



#12 OWASP Meeting 30.3.2010

Petteri Arola
OWASP Helsinki Chapter Leader

OWASP
Helsinki

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document
under the terms of the OWASP License.

The OWASP Foundation
<http://www.owasp.org>

Agenda

- Welcome / Petteri Arola, OWASP Helsinki Chapter Leader
- Word from our sponsor Helsingin Energia
- 3 different views on information security and social media applications
 - ▶ Information security in social media API's, Antti Nuopponen/Nixu Oy
 - ▶ Facebook apps, Markus Törnqvist/Fad Consulting
 - ▶ Payment API's, Tuomas Toivonen/Scred

OWASP Principles

- Free & Open
- Governed by rough consensus & running code
- Abide by a code of ethics:
http://www.owasp.org/index.php/About_OWASP
- Not-for-profit
- Not driven by commercial interests
- Risk based approach

OWASP Projects

■ Quality Levels

- ▶ Alpha, Beta, Release



■ Organizational Structure within Tools and Docs

- ▶ **PROTECT** - These are tools and documents that can be used to guard against security-related design and implementation flaws.
- ▶ **DETECT** - These are tools and documents that can be used to find security-related design and implementation flaws.
- ▶ **LIFE CYCLE** - These are tools and documents that can be used to add security-related activities into the Software Development Life Cycle (SDLC).

OWASP Application Security Verification Standard (ASVS)

■ OWASP's 1st Standard

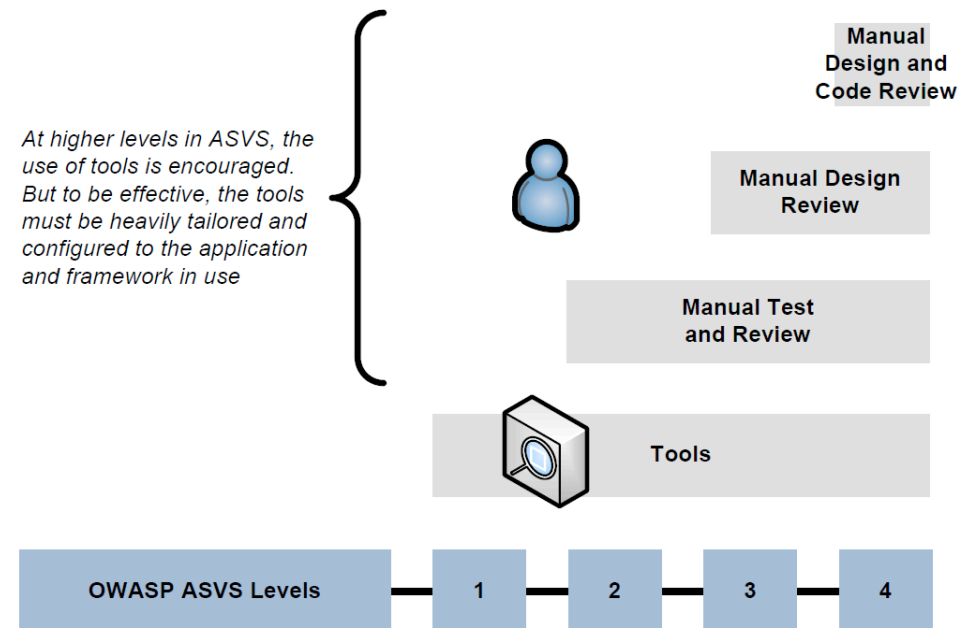
■ Defines 4 Verification Levels

- ▶ Level 1: Automated Verification
 - Level 1A: Dynamic Scan
 - Level 1B: Source Code Scan
- ▶ Level 2: Manual Verification
 - Level 2A: Penetration Test
 - Level 2B: Code Review
- ▶ Level 3: Design Verification
- ▶ Level 4: Internal Verification



What Questions Does ASVS Answer?

- How can I compare verification efforts?
- What security features should be built into the required set of security controls?
- What are reasonable increases in coverage and level of rigor when verifying the security of a web application?
- How much trust can be placed in a web application?



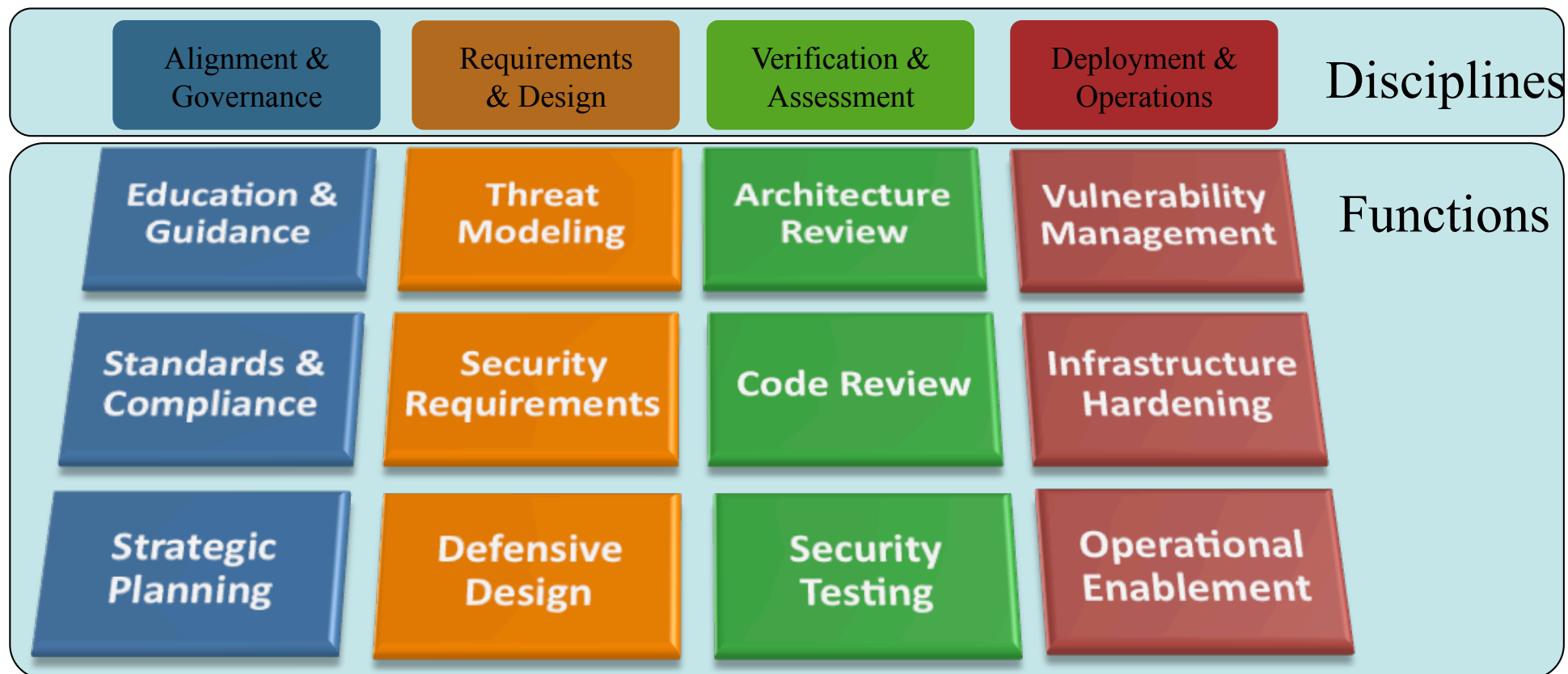
OpenSAMM

- Evaluate an organization's existing software security practices
- Build a balanced software security assurance program in well-defined iterations
- Demonstrate concrete improvements to a security assurance program
- Define and measure security-related activities throughout an organization



Software Assurance Maturity Model (SAMM)

- The 4 Disciplines are high-level categories for activities
 - ▶ Three security Functions under each Discipline are the specific silos for improvement within an organization



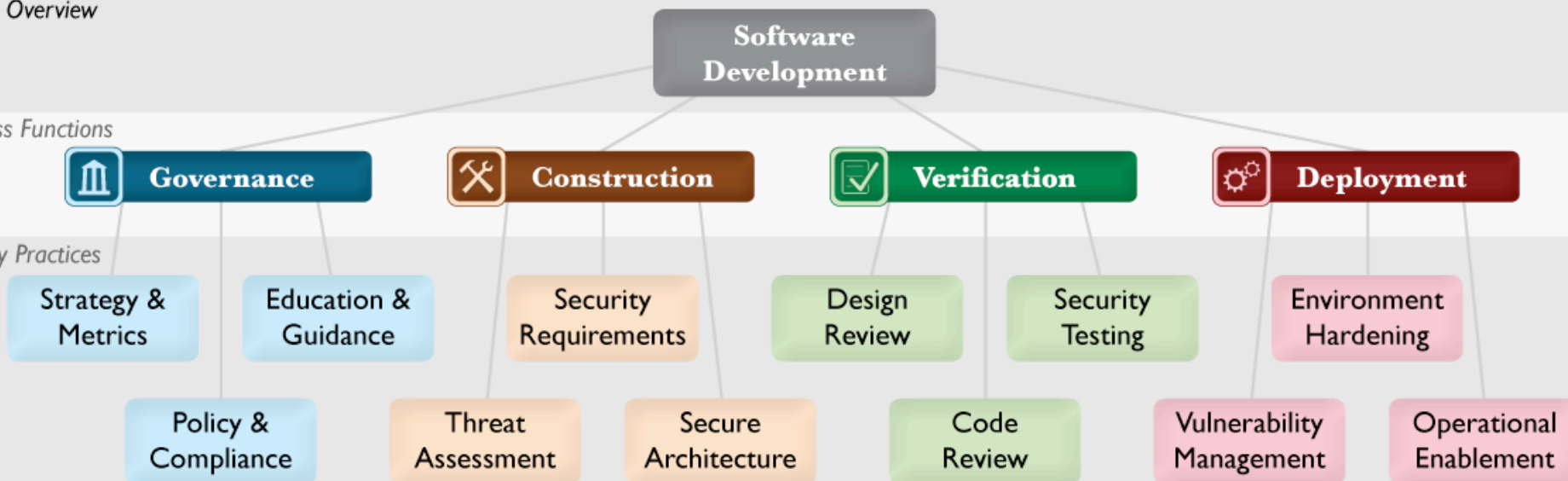
SAMM Security Practices

- 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



The three levels for Practice


- (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

	Education & Guidance ...more on page 42		
	 EG 1	 EG 2	 EG 3
OBJECTIVE	Offer development staff access to resources around the topics of secure programming and deployment	Educate all personnel in the software life-cycle with role-specific guidance on secure development	Mandate comprehensive security training and certify personnel for baseline knowledge
ACTIVITIES	A. Conduct technical security awareness training B. Build and maintain technical guidelines	A. Conduct role-specific application security training B. Utilize security coaches to enhance project teams	A. Create formal application security support portal B. Establish role-based examination/certification

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 inrelevance. 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

SAMM / The Secure Practices - v1.0

43

OWASP

11

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?		 EG 1
◆ 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?		 EG 2
◆ Are most people tested to ensure a baseline skill-set for secure development practices?		 EG 3