Secure Development: Models and Best Practices

Bart De Win
Bart.DeWin@owasp.org

Secure Development Training by Bart De Win
Bart De Win, Ph.D.

- 20+ years experience in secure software development
- Belgian OWASP chapter co-leader
- SAMM contributor, evangelist and co-leader
- Author of >60 publications
- Director & security consultant @PwC BE
- Bart.de.win@pwc.com

This training?

- Software Assurance maturity models
- Secure Development in agile development
- Hands-on: SAMM analysis of your enterprise using SAMM 1.5
- Tips and tricks for practical SDLC
- Sneak preview of SAMM 2.0
### Timing

- **09h30 – 11h00:** Training
- **11h00 – 11h30:** *coffee break*
- **11h30 – 13h00:** Training
- **13h00 – 14h00:** *lunch*
- **14h00 – 15h30:** Training
- **15h30 – 16h00:** *coffee break*
- **16h00 – 17h30:** Training

### Rules of the House

- Turn off mobile phones
- Interactive training
- Specific discussions about company practices don’t leave this room
Today’s Agenda

1. Introduction to SDLC and SAMM
2. Applying SAMM
   - Methodology
   - Assessment Governance
   - Assessment Construction
   - Assessment Verification
   - Assessment Operations
   - Setting Improvement Targets
3. Secure Agile development
4. SDLC Tips and tricks
5. Wrap-up

Application Security Problem

- 75% of vulnerabilities are application related

Software complexity
Technology stacks
Requirements?
- Mobile
- Multi-platform
- Cloud
- Connected
- Responsive Design

OWASP Benelux 2017 - Secure Development Training
Application Security Symbiosis

Application Security during Software Development

- Analyse
- Design
- Implement
- Test
- Deploy
- Maintain

Bugs  Flaws  Cost
The State-of-Practice in Secure Software Development

**Problematic**, since:

- Focus on bugs, not flaws
- Penetration can cause major harm
- Not cost efficient
- No security assurance
  - All bugs found?
  - Bug fix fixes all occurrences? (also future?)
  - Bug fix might introduce new security vulnerabilities

**SDLC**?

Enterprise-wide software security improvement program

- Strategic approach to assure software quality
- Goal is to increase systematicity
- Focus on security functionality and security hygiene
SDLC Cornerstones

- People
  - Roles & Responsibilities
- Process
  - Activities
  - Deliverables
  - Control Gates
- Knowledge
  - Standards & Guidelines
  - Compliance
  - Transfer methods
- Tools & Components
  - Development support
  - Assessment tools
  - Management tools

Strategic?

1. Organizations with a proper SDLC will experience an 80 percent decrease in critical vulnerabilities

2. Organizations that acquire products and services with just a 50 percent reduction in vulnerabilities will reduce configuration management and incident response costs by 75 percent each.
Does it really work?

Vulnerabilities disclosed three years after release

95% DECREASE

SQL Server 2000
34
Before SDL

SQL Server 2005
3
After SDL

Competing Commercial DB
187

Vulnerabilities disclosed one year after release

45% DECREASE

Windows XP
159
Before SDL

Windows Vista
66
After SDL


SDLC-related initiatives

Microsoft SDL
CLASP
BSIMM
SAMM
OWASP
TouchPoints

OWASP Benelux 2017 - Secure Development Training
So what about … Waterfall?

So what about … Agile?
Software Assurance

Is NOT …

But is …

Why 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)

https://www.owasp.org/index.php/OWASP_SAMM_Project
SAMM 101 – Introduction to the model

Core model document

SAMM Business Functions

• Start with the core activities tied to any organization performing software development

• Named generically, but should resonate with any developer or manager
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

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:
Check out this one...

<table>
<thead>
<tr>
<th>Education &amp; Guidance</th>
<th>EG 1</th>
<th>EG 2</th>
<th>EG 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Objective</strong></td>
<td>Offer development staff access to resources around the topics of secure programming and deployment.</td>
<td>Educate all personnel in the software life-cycle with role-specific guidance on secure development.</td>
<td>Mandate comprehensive security training and certify personnel for baseline knowledge.</td>
</tr>
<tr>
<td><strong>Activities</strong></td>
<td>A. Conduct technical security awareness training. B. Build and maintain technical guidelines.</td>
<td>A. Conduct role-specific application security training. B. Utilize security coaches to enhance project teams.</td>
<td>A. Create formal application security support portal. B. Establish role-based examination/certification.</td>
</tr>
</tbody>
</table>

Per Level, SAMM defines...

- Objective
- Activities
- Results
- Success Metrics
- Costs
- Personnel
- Related Levels
Applying the model

How-to guide

Assessment process

START  COMPLETE ASSESSMENT WORKSHEETS  ASSIGN A SCORE PER PRACTICE  LIGHTWEIGHT  END

DETAILED

AUDIT FOR PERFORMED ACTIVITIES  CHECK SUCCESS METRICS  ADJUST SCORE PER PRACTICE
Assessment worksheets

<table>
<thead>
<tr>
<th>Policy &amp; Compliance</th>
<th>Score</th>
<th>0.0</th>
<th>0.2</th>
<th>0.5</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do project stakeholders know their project’s compliance status?</td>
<td></td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Are compliance requirements specifically considered by project teams?</td>
<td></td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Does the organization utilize a set of policies and standards to control software development?</td>
<td></td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Are project teams able to request an audit for compliance with policies and standards?</td>
<td></td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Are projects periodically audited to ensure a baseline of compliance with policies and standards?</td>
<td></td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Does the organization systematically use audits to collect and control compliance evidence?</td>
<td></td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Intermezzo – how to measure

How well? How wide?
Assessment Toolbox

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
- Organization types chosen because
  - They represent common use-cases
  - Each organization has variations in typical software-induced risk
  - Optimal creation of an assurance program is different for each

SAMM vs. BSIMM

- Prescriptive vs. Descriptive
- Open vs. Closed
- Low Watermark vs. High Watermark

OWASP Benelux 2017 - Secure Development Training
Today’s Agenda

1. Introduction to SDLC and SAMM
2. Applying SAMM
   - Methodology
   - Assessment Governance
   - Assessment Construction
   - Assessment Verification
   - Assessment Operations
   - Setting Improvement Targets
3. Secure Agile development
4. SDLC Tips and tricks
5. Wrap-up

Before you begin

• Organizational Context
• Realistic Goals?
• Scope?
• Constraints (budget, timing, resources)
• Affinity with a particular model?
What’s your Company Maturity?

- In terms of IT strategy and application landscape
- In terms of software Development practices
  - Analysis, Design, Implementation, Testing, Release, Maintenance
  - Structured vs. ad-hoc development
- In terms of ITSM practices

Company Maturity ≈ Feasibility SDLC Program

Complicating factors, anyone?

- Different development teams
- Different technology stacks
- Business-IT alignment issues
- Outsourced development
- ...
Continuous Improvement with SAMM

Prepare

1. Purpose
   Ensure a proper start of the project

2. Activities
   Define the scope (uniform unit(s))
   Identify stakeholders
   Spread the word
Assess

1. Purpose
   Identify and understand the maturity of the 12 practices for the chosen scope

2. Activities
   Evaluate current practices
   Determine maturity level

Set The Target

1. Purpose
   Develop a target score to guide you in future improvements

2. Activities
   Define the target
   Estimate overall impact
Define the plan

1. Purpose
   Define or update the plan to take you to the next level

2. Activities
   Determine change schedule
   Develop/update the roadmap plan

Implement

1. Objective
   Work the plan

2. Activities
   Implement activities
**Roll-out**

1. Objective
   Ensure improvements are available and effectively used

2. Activities
   Evangelize improvements
   Measure effectiveness
Governance
Business Function

12 Security Practices

SAMM Overview

12 Security Practices


Policy & Compliance  Threat Assessment  Secure Architecture  Implementation Review  Issue Management  Operational Enablement

OWASP Benelux 2017 - Secure Development Training
**Strategy & Metrics**

1. Goal is to establish a software assurance framework within an organisation
   - Foundation for all other SAMM practices

2. Characteristics:
   - Measurable
   - Aligned with business risk

3. Driver for continuous improvement and financial guidance
Policy & Compliance

1. Goal is to understand and adhere to legal and regulatory requirements
   Typically external in nature
   This is often a very informal practice in organisations!

2. Characteristics
   Organisation-wide vs. project-specific
   Scope

3. Important driver for software security requirements
## Education & Guidance

1. **Goal is to disseminate security-oriented information to all stakeholders involved in the software development lifecycle**
   
   By means of standards, trainings, ...

2. **To be integrated with organisation training curriculum**
   
   A once-of effort is not sufficient
   
   Teach a fisherman to fish

3. **Technical guidelines form the basis for several other practices**

### Objectives

<table>
<thead>
<tr>
<th>EG 1</th>
<th>EG 2</th>
<th>EG 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offer development staff access to resources around the topics of secure programming and deployment.</td>
<td>Educate all personnel in the software life-cycle with role-specific guidance on secure development.</td>
<td>Mandate comprehensive security training and certify personnel for baseline knowledge.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Conduct technical security awareness training</td>
</tr>
<tr>
<td>B. Build and maintain technical guidelines</td>
</tr>
<tr>
<td>A. Conduct role-specific application security training</td>
</tr>
<tr>
<td>B. Utilize security coaches to enhance project teams</td>
</tr>
<tr>
<td>A. Create formal application security support portal</td>
</tr>
<tr>
<td>B. Establish role-based examination/certification</td>
</tr>
</tbody>
</table>
Assessment Exercise

• Use SAMM to evaluate the development practices in your own company

• Focus on Governance Business Function

• Applicable to both Waterfall and Agile models

• Using distributed sheets and questionnaires (toolbox)

Assessment wrap-up

• What’s your company’s score?

• What’s the average scores for the group?

• Any odd ratings?
Construction
Business Function

12 Security Practices
Threat Assessment

1. The goal of this practice is to focus on the attacker perspective of things
   To make sure that security is not only functionality-driven
   Remember that software security = white + black

2. Very common practice in safety-critical systems
   Less so in others

3. This is where “the magic” kicks in
   Your imagination is the limit

---

Threat Assessment

<table>
<thead>
<tr>
<th>Objective</th>
<th>TA 1</th>
<th>TA 2</th>
<th>TA 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify and understand high-level threats to the organization and individual projects</td>
<td>Increase accuracy of threat assessment and improve granularity of project understanding</td>
<td>Concretely tie compensating controls to each threat against internal and third-party software</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activities</th>
<th>TA 1</th>
<th>TA 2</th>
<th>TA 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Build and maintain application-specific threat models</td>
<td>A. Build and maintain abuse-case models per project</td>
<td>A. Explicitly evaluate risk from third-party components</td>
<td></td>
</tr>
<tr>
<td>B. Develop attacker profile from software architecture</td>
<td>B. Adopt a weighting system for measurement of threats</td>
<td>B. Elaborate threat models with compensating controls</td>
<td></td>
</tr>
</tbody>
</table>

---

Secure Development Training by Bart De Win
**Security Requirements**

1. **Goal is to make security specification more explicit**
   
   Turn security into a positively-spaced problem

2. **Source of security requirements**
   
   - Compliance
   - Risk
   - Functionality
   - Quality

3. **Requirements should be specified in a S.M.A.R.T. way**

---

### More on page 30

<table>
<thead>
<tr>
<th>Security Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SR 1</strong> Consider security explicitly during the software requirements process</td>
</tr>
<tr>
<td><strong>SR 2</strong> Increase granularity of security requirements derived from business logic and known risks</td>
</tr>
<tr>
<td><strong>SR 3</strong> Mandate security requirements process for all software projects and third-party dependencies</td>
</tr>
</tbody>
</table>

**Activities**

- **SR 1**
  - A. Derive security requirements from business functionality
  - B. Evaluate security and compliance guidance for requirements

- **SR 2**
  - A. Build an access control matrix for resources and capabilities
  - B. Specify security requirements based on known risks

- **SR 3**
  - A. Build security requirements into supplier agreements
  - B. Expand audit program for security requirements

---
Secure Architecture

1. Key practice for security
   Poor decisions at this step can have major impact, and are often difficult (or costly) to fix.

2. Characteristics
   Take into account security principles
   Risk is a factor of all components (incl. 3rd party)

3. Use proven solutions
   Don’t roll you own crypto
   Use company standards and best practices
Assessment Exercise

- Use SAMM to evaluate the development practices in your own company

- Focus on Construction Business Function

- Applicable to both Waterfall and Agile models

- Using distributed sheets and questionnaires (toolbox)

Assessment wrap-up

- What’s your company’s score?

- What’s the average scores for the group?

- Any odd ratings?
Verification
Business Function

12 Security Practices

OWASP Benelux 2017 - Secure Development Training
**Design Review**

- security assessment of attack surface, software design and architecture
- lightweight activities => formal inspection of data flows & security mechanisms
- enforcement of baseline expectations for conducting design assessments and reviewing findings before releases are accepted.

⇒ Assess and validate artifacts to understand protection mechanisms

### Design Review

<table>
<thead>
<tr>
<th>Objective</th>
<th>DR 1</th>
<th>DR 2</th>
<th>DR 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support ad hoc reviews of software design to ensure baseline mitigations for known risks</td>
<td>Offer assessment services to review software design against comprehensive best practices for security</td>
<td>Require assessments and validate artifacts to develop detailed understanding of protection mechanisms</td>
<td></td>
</tr>
</tbody>
</table>

**Activities**

- A. Identify software attack surface
- B. Analyze design against known security requirements

- A. Inspect for complete provision of security mechanisms
- B. Deploy design review service for project teams

- A. Develop data-flow diagrams for sensitive resources
- B. Establish release gates for design review
Implementation Review

Assessment of source code:
- vulnerability discovery
- related mitigation activities
- establish secure coding baseline

Will require tool investment:
- Language specific
- Basic open source tooling
- Commercial tools maturing

Process & education important!

Implementation Review

<table>
<thead>
<tr>
<th>Objective</th>
<th>IR 1</th>
<th>IR 2</th>
<th>IR 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opportunistically find basic code-level vulnerabilities and other high-risk security issues</td>
<td>Make implementation review during development more accurate and efficient through automation</td>
<td>Mandate comprehensive implementation review process to discover language-level and application-specific risks</td>
<td></td>
</tr>
<tr>
<td>ACTIVITIES</td>
<td>A. Create review checklists from known security requirements</td>
<td>A. Utilize automated code analysis tools</td>
<td>A. Customize code analysis for application-specific concerns</td>
</tr>
<tr>
<td></td>
<td>B. Perform point-review of high-risk code</td>
<td>B. Integrate code analysis into development process</td>
<td>B. Establish release gates for code review</td>
</tr>
</tbody>
</table>

OWASP Benelux 2017 - Secure Development Training
Security Testing

- Based on security & compliance requirements / checklist of common vulnerabilities
- Manual testing can be done, scaled with tooling: intercepting proxy and/or scanner
- Detected defects will require validation, risk analysis & recommendations to fix
- Automate to repeat tests for each release
- Introduce security test-driven development
- Test results to be reported to & accepted by owner for each deployment

Dynamic security testing

Detect vulnerabilities & misconfigurations

penetration testing => automation

OWASP Benelux 2017 - Secure Development Training

Security Testing

<table>
<thead>
<tr>
<th>Objective</th>
<th>Security Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST 1</td>
<td>Establish process to perform basic security tests based on implementation and software requirements</td>
</tr>
<tr>
<td>ST 2</td>
<td>Make security testing during development more complete and efficient through automation</td>
</tr>
<tr>
<td>ST 3</td>
<td>Require application-specific security testing to ensure baseline security before deployment</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activities</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>Derive test cases from known security requirements</td>
<td>A. Utilize automated security testing tools</td>
<td>A. Employ application-specific security testing automation</td>
</tr>
<tr>
<td>B.</td>
<td>Conduct penetration testing on software releases</td>
<td>B. Integrate security testing into development process</td>
<td>B. Establish release gates for security testing</td>
</tr>
</tbody>
</table>

OWASP Benelux 2017 - Secure Development Training
Assessment Exercise

• Use SAMM to evaluate the development practices in your own company

• Focus on Verification Business Functions

• Applicable to both Waterfall and Agile models

• Using distributed sheets and questionnaires (toolbox)

Assessment wrap-up

• What’s your company’s score?

• What’s the average scores for the group?

• Any odd ratings?
## Issue Management

**Prepare for WHEN, not IF!**

**Symptoms of malfunctioning SDLC**

- handling vulnerability reports and operational incidents
- lightweight assignment of roles => formal incident response & communication process
- Use vulnerability metrics and root-cause analysis to improve SDLC
- spoc per team & security response team
- communication & information flow is key!
- patch release process & responsible/legal disclosure

### Issue Management

<table>
<thead>
<tr>
<th>Objective</th>
<th>IM1</th>
<th>IM2</th>
<th>IM3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Understand high-level plan for responding to issue reports or incidents</td>
<td>Elaborate expectations for response process to improve consistency and communications</td>
<td>Improve analysis and data gathering within response process for feedback into proactive planning</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activities</th>
<th>Im1</th>
<th>Im2</th>
<th>Im3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Identify point of contact for security issues</td>
<td>A. Establish consistent issue response process</td>
<td>A. Conduct root cause analysis for for issues</td>
<td></td>
</tr>
<tr>
<td>B. Create informal security response team(s)</td>
<td>B. Adopt a security issue disclosure process</td>
<td>B. Collect pen issue metrics</td>
<td></td>
</tr>
</tbody>
</table>

OWASP Benelux 2017 - Secure Development Training
Environment Hardening

- Underlying infrastructure hardening & patching

- Track (3rd party) libraries & components
  TOP-10 - A9 – Using Known Vulnerable Components

- Add WAF layer (virtual patching)
  ModSecurity

Environment Hardening

<table>
<thead>
<tr>
<th>Objective</th>
<th>EH1</th>
<th>EH2</th>
<th>EH3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Understand baseline operational environment for applications and software components</td>
<td>Improve confidence in application operations by hardening the operating environment</td>
<td>Validate application health and status of operational environment against known best practices</td>
<td></td>
</tr>
<tr>
<td>Activities</td>
<td>A. Maintain operational environment specification</td>
<td>A. Establish routine patch management process</td>
<td>A. Identify and deploy relevant operations protection tools</td>
</tr>
<tr>
<td></td>
<td>B. Identify and install critical security upgrades and patches</td>
<td>B. Monitor baseline environment configuration status</td>
<td>B. Expand audit program for environment configuration</td>
</tr>
</tbody>
</table>

OWASP Benelux 2017 - Secure Development Training
Operational Enablement

Support users & operators

Security documentation!

Feed/document application security logs into SIEM

Lightweight documentation => operational security guides

Change management & end to end deployment integrity

Even more important for outsourced development!
Assessment Exercise

• Use SAMM to evaluate the development practices in your own company

• Focus on Deployment Business Functions

• Applicable to both Waterfall and Agile models

• Using distributed sheets and questionnaires (toolbox)

Assessment wrap-up

• What’s your company’s score?

• What’s the average scores for the group?

• Any odd ratings?
Setting the Target/Roadmap

1. Roadmap templates can provide direction for targets
   What type of company are you?

2. Take into account the company’s risk appetite

3. Only include activities where you see added value for the company, even for lower levels

4. SAMM activities have dependencies – use them!

5. Think about links with other practices in the company
   E.g., training, release management, ...

Staged Roadmap

<table>
<thead>
<tr>
<th>Security Practices/Phase</th>
<th>Start</th>
<th>One</th>
<th>Two</th>
<th>Three</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy &amp; metrics</td>
<td>0,5</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Policy &amp; Compliance</td>
<td>0</td>
<td>0,5</td>
<td>1</td>
<td>1,5</td>
</tr>
<tr>
<td>Education &amp; Guidance</td>
<td>0,5</td>
<td>1</td>
<td>2</td>
<td>2,5</td>
</tr>
<tr>
<td>Threat Assessment</td>
<td>0</td>
<td>0,5</td>
<td>2</td>
<td>2,5</td>
</tr>
<tr>
<td>Security Requirements</td>
<td>0,5</td>
<td>1,5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Secure Architecture</td>
<td>0,5</td>
<td>1,5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Design Review</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2,5</td>
</tr>
<tr>
<td>Code Review</td>
<td>0</td>
<td>0,5</td>
<td>1,5</td>
<td>2,5</td>
</tr>
<tr>
<td>Security Testing</td>
<td>0,5</td>
<td>1</td>
<td>1,5</td>
<td>2,5</td>
</tr>
<tr>
<td>Vulnerability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Management</td>
<td>2,5</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Environment Hardening</td>
<td>2,5</td>
<td>2,5</td>
<td>2,5</td>
<td>2,5</td>
</tr>
<tr>
<td>Operational Enablement</td>
<td>0,5</td>
<td>0,5</td>
<td>1,5</td>
<td>3</td>
</tr>
<tr>
<td><strong>Total Effort per Phase</strong></td>
<td><strong>7,5</strong></td>
<td><strong>7,5</strong></td>
<td><strong>7,5</strong></td>
<td><strong>7,5</strong></td>
</tr>
</tbody>
</table>
**Improvement Exercise**

- Define a target for your company and the phased roadmap to get there
- Focus on the most urgent/heavy-impact practices first
- Try balancing the complexity and effort of the different step-ups

**Conclusion Applying SAMM**

Lightweight assessment of 12 security practices

Your thoughts:
- Representative summary?
- New insights learned?
- Anything not covered?
- ...
Today’s Agenda

1. Introduction to SDLC and SAMM
2. Applying SAMM
   - Methodology
   - Assessment Governance
   - Assessment Construction
   - Assessment Verification
   - Assessment Operations
   - Setting Improvement Targets
3. Secure Agile development
4. SDLC Tips and tricks
5. Wrap-up

Agile Models: Scrum
Agile & Secure development: a mismatch?

<table>
<thead>
<tr>
<th>Agile Dev.</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed &amp; Flexibility</td>
<td>Stable &amp; Rigorous</td>
</tr>
<tr>
<td>Short cycles</td>
<td>Extra activities</td>
</tr>
<tr>
<td>Limited documentation</td>
<td>Extensive analysis</td>
</tr>
<tr>
<td>Functionality-driven</td>
<td>Non-functional</td>
</tr>
</tbody>
</table>

Secure Agile is ...

enablement, rather than control

scalability
Secure Agile – Where’s the difference?

**People**
- Roles & Responsibilities

**Process**
- Activities
- Deliverables
- Control Gates

**Knowledge**
- Standards & Guidelines
- Compliance
- Transfer methods

**Tools & Components**
- Development support
- Assessment tools
- Management tools

---

Secure Agile: general principles

- Make security a natural part of the process, but don’t overdo
  - Lightweight, in-phase and iterative
  - Preventive and detective controls
- Be involved at key moments in the process
- Leverage important agile concepts
- Small steps at a time (i.e. continuous improvement)
User Stories

- Capture security requirements, policies and regulations in user stories
- Simple, concrete and actionable
- Reusable?
- Mark all user stories with security labels

- Integrate security into user stories as:
  - Definition of Done
  - Acceptance criteria

Threat Modelling & Abuser Stories

- Consider writing application security risks as stories

- Security stories: “As a developer, I want to prevent SQLi into my application”
  - Not a real user story (not relevant for product owner, but to help the development team)
  - Never really finished

- Thinking like the bad guy: “User X should not have access to this type of data”
  - Think about what users don’t want to and can’t do, how to trust users, what data is involved, ...
Sprint Planning

- Features to be implemented per sprint are selected during sprint planning.

- Ensure security tasks are not “stuck” on the backlog
  - Presence of security-savvy person during sprint planning
  - Establish rules *upfront* to deal with security stories
  - Security labels can be used to drive selection

Example: MS SDL-Agile

- Basic approach: Fit SDL tasks to the backlog as non-functional stories
  - Non-Technical vs. Technical
  - Requirement vs. Recommendation

- Each SDL task goes in one of three types of requirements:
Example: Every-Sprint Requirements (excerpt)

- All team members must have had security training in the past year
- All database access via parameterized queries
- Fix security issues identified by static analysis
- Mitigate against Cross-Site Request Forgery
- Update Threat models for new features
- Use Secure cookies over HTTPS
- Link all code with the /nxcompat linker option
- Encrypt all secrets such as credentials, keys and passwords
- Conduct internal security design review

Example: Bucket Requirements (excerpt)

**Bucket A: Security Verification**
- Perform fuzzing (network/ActiveX/File/RPC/…)
- Manual and automated code review for high-risk code
- Penetration testing

**Bucket B: Design Review**
- Conduct a privacy review
- Complete threat model training

**Bucket C: Planning**
- Define or update the security/privacy bug bar
- Define a BC/DR plan
Example: One-Time Requirements (excerpt)

• Create a baseline threat model
• Establish a security response plan
• Identify your team’s security expert
• Use latest compiler versions

Security testing

• Automated testing is an important element in agile quality control

• For security, this can be realized by:
  • Unit testing (e.g., authorisation checks, logging, ...)
  • Regression testing
  • Static analysis (SAST) based on coding guidelines
  • Dynamic analysis (DAST) based on scenarios and/or vulnerability tests
  • Fuzzing
Thou shall use Iteration Zero

- Many agile projects start with an "Iteration Zero" to
- Get the team together
- Choose tools and frameworks
- Get to know the domain

- This is an opportunity for security too, to
- Assign security responsible
- Select security tools
- Determine risk levels

Secure Agile process: key take-aways

- Ensure that security-savvy people are involved at important phases:
  - Sprint planning (to enhance/verify requirements)
  - Development (daily follow-up)
  - Review (to support acceptance)
  - Retrospective (to improve dev. Practices for security)

- Different profiles can be distinguished:
  - Security architect
  - Security engineer
  - Risk Manager/Governance
Secure Agile Tool Chain: general principles

- Secure agile is about enabling, rather than controlling
  - Embedding security tools to support development
  - Given short sprint cycles, automation is important.

- Good tools:
  - Work continuously (to avoid developers being blocked)
  - Integrate well into developer’s world
  - Avoid causing too much overhead or confusion

- Evaluate carefully which tools to implement (and which to avoid)

Secure Coding

- Integrate security tools in the development IDE’s:
  - Support for secure coding guidelines
  - Static analysis tools

- Ensure common development environment:
  - Programming run-time
  - Security components (e.g., SSO IdP’s, ...)

- Proper source control and versioning
**Security testing**

**Daily**
- Unit tests
- Regression tests
- Peer reviews

**Per sprint**
- Static Analysis
- Dynamic Analysis
- Fuzzing

**Before release**
- Penetration testing

**Secure Build**

- Central build, using central configuration files

- Consider:
  - Code signing
  - Obfuscation
  - ...

OWASP Benelux 2017 - Secure Development Training
Secure Deploy / DevOps

- Automated deploy, using central configuration files

- Consider:
  - Random key generation
  - Appropriate key/certificate protection (config files, key stores, ...)
  - Proper hardening of application servers
  - Security appliance configuration (e.g., WAF)
  - Security monitoring
  - ...

Hybrid models

- Many companies are combining waterfall and agile
  - Studies indicate better resulting quality

- For security, easier to hook into
  - E.g., full architecture cycle
Best Practices / Lessons Learned

- Use small steps at a time – the agile way
- Build on agile concepts (backlog, retrospective)
  - Find a way to prioritize security in the planning
- Use automation as much as possible
- Review samples independent of project sprints
- Rely on security champions
  - E.g., security requirements, design review, code review
- Agile should not be an excuse for not having documentation

Today’s Agenda

1. Introduction to SDLC and SAMM
2. Applying SAMM
   Methodology
   Assessment Governance
   Assessment Construction
   Assessment Verification
   Assessment Operations
   Setting Improvement Targets
3. Secure Agile development
4. SDLC Tips and tricks
5. Wrap-up
The importance of a Business Case

If you want your company to improve, management buy-in is crucial
⇒ You will need a business case to convince them

Typical arguments:
• Improved security quality
• Better cost efficiency
• Compliance
• Risk management
• Customer satisfaction
• Reputation management

Entry Points

Pick the weak spots that can demonstrate short-term ROI

Typical examples
  - Awareness training
  - Coding Guidelines
  - External Pentesting

Success will help you in continuing your effort
Application categorization

Use this to rationalize security effort (according to the application risk)

Communication & Support

Critical success factor!

Spreading the message – broad audience
Setup a secure applications portal!
Regular status updates towards management
**Monitoring & Metrics**

Project vs. Enterprise dashboard

Manual vs. Automated data collection

![Monitoring & Metrics Image]

**Responsibilities**

Core Security team

Support vs. Responsible role

Security Satellite
- Analysts
- Architects
- Developers
- Operations
- Management

Formalized RACI will be a challenge
The Power of Default Security

Construct development frameworks that are secure by default

Minimizes work for developers

Will lower number of vulns.

SDLC impact

Difficult to predict, but:

• Projects are estimated to increase with 5 – 15% for security
• ROI is achievable taking maintenance and incident management into account
• SDLC capability costs approx. 1 FTE/100 developers
Today’s Agenda

1. Introduction to SDLC and SAMM
2. Applying SAMM
   Methodology
   Assessment Governance
   Assessment Construction
   Assessment Verification
   Assessment Operations
   Setting Improvement Targets
3. Secure Agile development
4. SDLC Tips and tricks
5. Wrap-up

Conclusions

Developing secure software gets more and more complex

SAMM = global maturity foundation for software assurance

Applying SAMM =
   Assessment
   Roadmap
   (Continuous) Implementation

Be ready to face the organisational challenges that will pop up during the journey
SDLC Cornerstones (recap)

- **People**
  - Roles & Responsibilities

- **Process**
  - Activities
  - Deliverables
  - Control Gates

- **Knowledge**
  - Standards & Guidelines
  - Compliance
  - Transfer methods

- **Tools & Components**
  - Development support
  - Assessment tools
  - Management tools

SAMM Project Roadmap

**v2.0 (In Progress):**
1. Model revision
2. More Metrics!
3. Application to agile/devops
4. Roadmap effort planning
5. Benchmarking

**Build the community:**
- Grow list of SAMM adopters
- Workshops at conferences
- Dedicated SAMM Summit
- Contribute Anon Results
Fundamental changes to the model for v2.0

OWASP Benelux 2017 - Secure Development Training

Working towards a stream-based structure

OWASP Benelux 2017 - Secure Development Training
The end