Chapter 12: Secure System Development

Raimundas Matulevičius

University of Tartu, Estonia, rma@ut.ee
Goal

• Explain main activities for secure software development
• Overview three approaches for secure software development
Outline

• Security system development processes
  – Microsoft secure system development lifecycle
  – OWASP CLASP
  – Seven touchpoints
  – Comparison

• Security Approaches in Secure Systems Development Processes
Outline

• Security system development processes
  - Microsoft secure system development lifecycle
  - Emphasis on „building secure software” as opposed to „building security software”
  - Comparison

• Security Approaches in Secure Systems Development Processes
Outline

• Security system development processes
  – Microsoft secure system development lifecycle
    – OWASP CLASP
    – Seven touchpoints
    – Comparison

• Security Approaches in Secure Systems Development Processes
Microsoft Security Development Cycle


- Core security training
Microsoft Security Development Cycle


- Establish security requirements
- Create quality gates/bug bars
- Perform security and privacy risk assessment
Microsoft Security Development Cycle


- Establish design requirements
- Perform attack surface analysis/reduction
- Use threat modelling
Microsoft Security Development Cycle


- Use approved tools
- Deprecate unsafe functions
- Perform static analysis
Microsoft Security Development Cycle


- Perform dynamic analysis
- Perform fuzz testing
- Conduct attack surface reviews
Microsoft Security Development Cycle

- Create an incident response plan
- Conduct final security review
- Certify release and archive

Microsoft Security Development Cycle


- Training
  - Core security training

- Requirement
  - Establish security requirements
  - Security and privacy risk assessment

- Design
  - Establish design requirements
  - Analyse attack surface
  - Threat modelling

- Implementation
  - Deprecate unsafe functions
  - Static analysis

- Verification
  - Dynamic analysis
  - Fuzz testing
  - Attack surface review

- Release
  - Final security review
  - Release archive

- Response
  - Execute incidence response plan
Microsoft Security Development Cycle


- Provide training
- Define security requirements
- Define metrics and compliance reporting
- Perform threat modelling
- Establish design requirements
- Define and use cryptography standards
- Manage the security risk of using third part components
- Use approved tools
- Perform static analysis security testing (SAST)
- Perform dynamic analysis security testing (DAST)
- Perform penetration testing
- Establish a standard incident response process
Outline

• Security system development processes
  – Microsoft secure system development lifecycle
  – OWASP CLASP
    – Seven touchpoints
    – Comparison
• Security Approaches in Secure Systems Development Processes
Open Web Application Security Project
OWASP
https://www.owasp.org/index.php/Main_Page

• Collect resources for Web applications
  – Top ten security flaws
  – Various security testing tools
  – Various security control means
    • e.g., code review guide

• CLASP – Comprehensive Lightweight Application Security Process

OWASP Appsec Tutorial Series
https://www.owasp.org/index.php/OWASP_Appsec_Tutorial_Series

– Basics
– SQL Injection
– Cross-site Scripting (XSS)
– HTTP Strict Transport Security
CLASP
https://www.owasp.org/index.php/Main_Page

• **Goal:**
  – move security concerns into the early stages of the software development lifecycle, whenever possible

• **Set of process pieces that can be integrated into any software development process**
  – Introduction to the **Concepts** behind CLASP to get started
  – Seven key **Best Practices**
  – High-level **Security Services** (authorisation, authentication, …)
  – Core Security **Principles**
  – **Roles**
  – **Activities**
  – **Process engineering** and roadmaps
  – Checklisted **Coding Guidelines**
  – **Vulnerabilities** that occur in source code
  – **Searchable** Vulnerability Checklist
CLASP Best Practices

- Institute awareness programs
- Perform application assessments
- Capture security requirements
- Implement secure development practices
- Build vulnerability remediation procedures
- Define and monitor metrics
- Publish operational security guidelines
CLASP Best Practices

- **Institute awareness programs**
  - Perform application assessments
  - Capture security requirements
  - Implement secure development practices
  - Build vulnerability remediation procedures
  - Define and monitor metrics
  - Publish operational security guidelines

- People should consider security to be an important project goal
- Train all team members
- Make people aware of security setting
- Institute accountability for security issues
- Appoint a project security officer
- Institute rewards for handling of security issues
CLASP Best Practices

- Institute awareness programs
- **Perform application assessments**
- Capture security requirements
- Implement secure development practices
- Build vulnerability remediation procedures
- Define and monitor metrics
- Publish operational security guidelines

- Security analysis of requirements and design
  - Threat modelling
- Source-level security review
- Security tests
CLASP Best Practices

- Institute awareness programs
- Perform application assessments
- **Capture security requirements**
- Implement secure development practices
- Build vulnerability remediation procedures
- Define and monitor metrics
- Publish operational security guidelines

- Treat security requirements same way as functional requirements
- Define security policy
- Identify attack surface
- Identify resources and trust boundaries
- Identify misuse cases
- Specify operational environment

© Springer International Publishing AG 2017
CLASP Best Practices

- Institute awareness programs
- Perform application assessments
- Capture security requirements
- **Implement secure development practices**
- Build vulnerability remediation procedures
- Define and monitor metrics
- Publish operational security guidelines

- Annotate classes with security properties
- Apply principles of secure design
- Manage resources
- Manage contracts and interfaces
CLASP Best Practices

- Institute awareness programs
- Perform application assessments
- Capture security requirements
- Implement secure development practices
- **Build vulnerability remediation procedures**
- Define and monitor metrics
- Publish operational security guidelines

- Address reported security issues
- Manage security issue disclosure process
CLASP Best Practices

- Institute awareness programs
- Perform application assessments
- Capture security requirements
- Implement secure development practices
- Build vulnerability remediation procedures
- **Define and monitor metrics**
- Publish operational security guidelines

- Select metrics
- Collect data
- Evaluate results
CLASP Best Practices

- Institute awareness programs
- Perform application assessments
- Capture security requirements
- Implement secure development practices
- Build vulnerability remediation procedures
- Define and monitor metrics
- **Publish operational security guidelines**

- Build operational security guide
- Specify database security configuration
Outline

• Security system development processes
  – Microsoft secure system development lifecycle
  – OWASP CLASP
  – Seven touchpoints
    – Comparison
• Security Approaches in Secure Systems Development Processes
“All software projects produce at least one artifact: source code”
### Seven Security Touchpoints


“All software projects produce at least one artifact: source code”

| 1. Code review (tools)      | 5. Abuse analysis          |
| 2. Risk analysis           | 6. Security requirements   |
| 3. Penetration test        | 7. Security attacks        |
| 4. Risk-based security test| * External analysis        |
Code review (tools)

• Aim: catching implementation bugs early
• Tool helps to achieve good code coverage
• Aim for good, not perfect
Risk analysis

• **Create description of architecture**
  – Start with one page
  – Forest-level view

• **Attack resistance**
  – Use checklists of known attacks
  – Example: Microsoft STRIDE
    • Spoofing, Tampering, Repudiation, Info disclosure, Denial of service, Elevation of privilege

• **Ambiguity analysis**
  – Discover new risks
  – Find unclear parts in how the system works
  – Trust, data sensitivity, threat models

• **Weakness analysis**
  – Impact of external software dependencies
  – Platform (hardware, OS)
  – Frameworks
  – Called services

Combine risks and consider business impact
Rank risks
Find solutions

© Springer International Publishing AG 2017
Penetration test

Attack on a system with the intention of finding security weaknesses, potentially gaining access to it, its functionality and data

• **Use the source**
  – Otherwise people send time on reverse-engineering system

• **Apply business priorities**
  – Logic flaw vs. XSS flaw
  – XSS is important if it contributes towards compromising business logic

• **Use in-house QA department**
  – They already know the system
  – Use tools and training to add security testing skills

• **Test more than once**

• **Incorporate the findings back into development**
Risk-based security test

• Test based on priorities
  – Architectural risks
  – Risks discovered during code review

• Test malicious input
  – Use fuzzing tool
Abuse analysis and Security requirement

• Security is not a set of features
• How system should react to illegitimate use
• Like use cases, but with malicious users
External analysis

• **Unfortunately**
  – Software architects, developers, and testers are largely unaware of the software security problems

• **Good news**
  – They acknowledge that security problems exists!

• **Bad news**
  – Barely begun to apply the security solutions
Outline

• Security system development processes
  – Microsoft secure system development lifecycle
  – OWASP CLASP
  – Seven touchpoints
  – Comparison

• Security Approaches in Secure Systems Development Processes
## Comparison

<table>
<thead>
<tr>
<th>Stages</th>
<th>SSDL</th>
<th>CLASP</th>
<th>Touchpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Education and awareness</strong></td>
<td>1. Baseline education and advanced education</td>
<td>1. Base level and advanced education</td>
<td>1. No separate touchpoint, but recognises that people should be sufficiently trained</td>
</tr>
<tr>
<td></td>
<td>2. Focus on awareness, knowledge of security engineering knowledge</td>
<td>2. Focus on all project roles</td>
<td>2. A knowledge management framework established to share security knowledge</td>
</tr>
<tr>
<td><strong>Project inception</strong></td>
<td>1. Decision about the methodology, personnel, tools, and targeted security bugs</td>
<td>1. Assignment of the security officer and determining influence of security on other development roles</td>
<td>1. Stress on the creation and continuous execution of an improvement program</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Motivation plan by institutional accountability and by means of reward</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Metric definition, collection and management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4. Organisational policy management</td>
<td></td>
</tr>
<tr>
<td><strong>Analysis and requirements</strong></td>
<td>1. Use scenarios as the means in threat modelling</td>
<td>1. Identifies resources, trust boundaries, capabilities for resources and roles, and attacker profiles</td>
<td>1. Touchpoint dedicated to threat modelling based on abuse cases</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Uses threat modelling and requirements specification means</td>
<td>2. Extra security requirements identified based on laws and regulations, commercial considerations and contractual obligations</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Deals with business requirements, functional security requirements, conflict resolution, etc.</td>
<td></td>
</tr>
<tr>
<td><strong>Architectural and detailed design</strong></td>
<td>1. Supports threat modelling</td>
<td>1. Supports threat modelling</td>
<td>1. Main focus on threat modelling</td>
</tr>
<tr>
<td></td>
<td>2. Focuses on operational environment</td>
<td>2. Audits both security and non-security requirements</td>
<td>2. Includes threat identification and risk assessment</td>
</tr>
<tr>
<td></td>
<td>3. Assesses user privacy</td>
<td>3. Focuses on reduction of access points</td>
<td>3. Tends to remove ambiguity</td>
</tr>
<tr>
<td></td>
<td>4. Tends to reduce attack surface (by reduction of privileges)</td>
<td>4. Annotates design models</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5. Secures configuration of databases</td>
<td></td>
</tr>
<tr>
<td><strong>Implementation and testing</strong></td>
<td>1. Provides secure coding guidelines</td>
<td>1. Emphasises on white box security testing</td>
<td>1. Big emphasis on security testing – three (out of seven) touchpoints deals with testing</td>
</tr>
<tr>
<td></td>
<td>2. Encourages both using automated tools for verification and manual code inspection</td>
<td>2. Includes implementation of interface contracts</td>
<td>2. Stress on importance of risk-based testing</td>
</tr>
<tr>
<td></td>
<td>3. Focus on black box testing</td>
<td>3. Reviews specification from developers perspective</td>
<td>3. Emphasis on code review using automated tools</td>
</tr>
<tr>
<td><strong>Release, deployment and support</strong></td>
<td>1. Focus on response plan (i.e., where, when vulnerability is determined)</td>
<td>1. Requests stakeholder to sign the code, so to provide a way to validate the origin and integrity of the product</td>
<td>1. Limited support in this activity (fine tuning access controls, configuring the monitoring and logging)</td>
</tr>
</tbody>
</table>
Comparison

- **SSDL**
  - Education and awareness
  - Project inception and release
  - Deployment and support activities

- **OWASP CLASP**
  - Project inception and release
  - Deployment and support activities
  - Analysis and requirements
  - Architectural and detailed design activities

- **Seven Touchpoints**
  - Implementation and testing
    - e.g., three out of seven touchpoints are related to the testing activities.
  - Analysis and requirements
  - Architectural and detailed design

[De Win et al., 2009]
Outline

• Security system development processes
  – Microsoft secure system development lifecycle
  – OWASP CLASP
  – Seven touchpoints
  – Comparison

• Security Approaches in Secure Systems Development Processes
<table>
<thead>
<tr>
<th>Security approaches</th>
<th>SSDL</th>
<th>CLASP</th>
<th>Touchpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISSRM</td>
<td>Requirements, Design</td>
<td>Assessing application, Capturing security requirements, Implementing secure development practices, Definition and monitoring of metrics</td>
<td>Risk analysis</td>
</tr>
<tr>
<td>STRIDE</td>
<td>Design</td>
<td>Capturing security requirements</td>
<td>Risk analysis</td>
</tr>
<tr>
<td>Threats for distributed systems</td>
<td>Design</td>
<td>Capturing security requirements</td>
<td>Risk analysis</td>
</tr>
<tr>
<td>Seven pernicious kingdoms</td>
<td>Design, Implementation</td>
<td>Building vulnerability remediation procedures</td>
<td>Code review, Security operations</td>
</tr>
<tr>
<td>Security Risk-oriented BPMN</td>
<td>Requirements</td>
<td>Capturing security requirements</td>
<td>Security requirements</td>
</tr>
<tr>
<td>Security Risk-aware Secure Tropos</td>
<td>Requirements, Design</td>
<td>Capturing security requirements</td>
<td>Abuse cases, Security requirements</td>
</tr>
<tr>
<td>Security Risk-oriented Misuse Cases</td>
<td>Design</td>
<td>Capturing security requirements</td>
<td>Abuse cases, Security requirements</td>
</tr>
<tr>
<td>Mal-activities for Security Management</td>
<td>Design</td>
<td>Capturing security requirements</td>
<td>Abuse cases, Security requirements</td>
</tr>
<tr>
<td>Secure model transformations</td>
<td>Requirements, Design</td>
<td>Implementing secure development procedures</td>
<td>Security requirements</td>
</tr>
<tr>
<td>RBAC</td>
<td>Design, Implementation</td>
<td>Implementing secure development procedures, Definition and monitoring of metrics</td>
<td>Security requirements, Security operations</td>
</tr>
<tr>
<td>SecureUML</td>
<td>Design, Implementation</td>
<td>Implementing secure development procedures</td>
<td>Security requirements, Security operations</td>
</tr>
<tr>
<td>UMLsec</td>
<td>Design, Implementation</td>
<td>Implementing secure development procedures</td>
<td>Security requirements, Security operations</td>
</tr>
<tr>
<td>Model-driven security</td>
<td>Design, Implementation</td>
<td>Implementing secure development procedures</td>
<td>Security requirements, Code review</td>
</tr>
<tr>
<td>Security patterns</td>
<td>Requirements, Design</td>
<td>Capturing security requirements, Implementing secure development practices</td>
<td>Security requirements, Abuse cases, Risk analysis, Security operations</td>
</tr>
<tr>
<td>SREBP</td>
<td>Requirements, Design</td>
<td>Capturing security requirements, Implementing secure development practices</td>
<td>Security requirements, Abuse cases, Risk analysis</td>
</tr>
</tbody>
</table>
Message to take home

• Emphasis on „building secure software” as opposed to „building security software”

• **Major methodologies**
  – Microsoft's Security Development Lifecycle
  – OWASP CLASP
  – Cigital's Security Touchpoints
Final slide

• Don't do anything just because somebody else does
• Apply the **scientific method**
  – “a method of procedure that has characterized natural science since the 17th century, consisting in systematic observation, measurement, and experiment, and the formulation, testing, and modification of hypotheses.”
  
Final slide

• Don't do anything just because somebody else does
• Apply the **scientific method**
  – „a method of procedure that has characterized natural science since the 17th century, consisting in systematic observation, measurement, and experiment, and the formulation, testing, and modification of hypotheses.”


**Creativity – your key to secure software**