Secure Software Development Levels and Costs

Elaine Venson
Bradford Clark
Barry Boehm
Outline

• Secure Software Development Costs
• Scale Development
• Resulting Estimates from Security Experts
• Next Steps
Outline

• Secure Software Development Costs
• Scale Development
• Resulting Estimates from Security Experts
• Next Steps
Secure Software Development

**Microsoft SDL**  [https://www.microsoft.com/](https://www.microsoft.com/)

**OWASP SAMM**  [https://owaspsamm.org/](https://owaspsamm.org/)

**Touchpoints**  [McGraw, 2011]
Software Security as a Trade-off

Secure Software Development

Lower Levels

Costs
- Higher fixing costs
- Patching
- Down-time
- Recovery costs
- Reputation loss
- Expertise
- Tools
- Training
- Improving processes
- Investment in early phases

Benefits
- Priority to features
- Better time to market
- Vulnerabilities prevention/detection
- Avoided risks
- Reduced total cost

Higher Levels
The right amount of security

Costs of Secure SW Development

Goals
- Build-in security to preserve assets (CIA)

Requirements
- Functional
- Non-functional

Development
- Features, controls, components
- Security practices (threat modeling, pen-testing, etc)

Measurement
- Lines of code, functions points, objective points
- Levels of application (scope and rigor)

CIA: Confidentiality, Integrity and Availability
## Measuring Secure SW Development

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Lines of code, functions points, objective points</th>
<th>Levels of application (scope and rigor)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Security Features Size:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Directly estimated using sw sizing methods, or</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Estimated using a <strong>Security Sizing Factor</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Secure Sw Dev Level:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Development of an <em>ordinal scale</em> based on application of software security practices – <strong>Secure Software Development Scale</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Presented for the ICEAA 2021 Online Workshop - www.iceaaonline.com
Outline

• Secure Software Development Costs
• Scale Development
• Resulting Estimates from Security Experts
• Next Steps
Secure Software Development Scale

• Ordinal scale defining degrees of application of security practices
• Scale items development based on:
  - Literature
  - BSIMM (Building Security in Maturity Model)
  - OWASP SAMM (Software Assurance Maturity Model)
    OWASP: Open Web Application Security Project
  - COCOMO (Constructive Cost Model) descriptors of attribute levels
## Software Security Practices

<table>
<thead>
<tr>
<th>Practice</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply Security Requirements</td>
<td>Consider and document security concerns prior to implementation of software features.</td>
</tr>
<tr>
<td>Apply Data Classification Scheme</td>
<td>Maintain and apply a Data Classification Scheme. Identify and document security-sensitive data, personal information, financial information, system credentials.</td>
</tr>
<tr>
<td>Apply Threat Modeling</td>
<td>Anticipate, analyze, and document how and why attackers may attempt to misuse the software.</td>
</tr>
<tr>
<td>Document Technical Stack</td>
<td>Document the components used to build, test, deploy, and operate the software. Keep components up to date on security patches.</td>
</tr>
<tr>
<td>Apply Secure Coding Standards</td>
<td>Apply (and define, if necessary) security-focused coding standards for each language and component used in building the software.</td>
</tr>
<tr>
<td>Apply Security Tooling</td>
<td>Use security-focused verification tool support (e.g. static analysis, dynamic analysis, coverage analysis) during development and testing.</td>
</tr>
<tr>
<td>Perform Security Testing</td>
<td>Consider security requirements, threat models, and all other available security-related information and tooling when designing and executing the software’s test plan.</td>
</tr>
<tr>
<td>Perform Penetration Testing</td>
<td>Arrange for security-focused stress testing of the project’s software in its production environment. Engage testers from outside the software’s project team.</td>
</tr>
<tr>
<td>Perform Security Review</td>
<td>Perform security-focused review of all deliverables, including, for example, design, source code, software release, and documentation. Include reviewers who did not produce the deliverable being reviewed.</td>
</tr>
<tr>
<td>Publish Operations Guide</td>
<td>Document security concerns applicable to administrators and users, supporting how they configure and operate the software.</td>
</tr>
<tr>
<td>Track Vulnerabilities</td>
<td>Track software vulnerabilities detected in the software and prioritize their resolution.</td>
</tr>
<tr>
<td>Improve Development Process</td>
<td>Incorporate “lessons learned” from security vulnerabilities and their resolutions into the project’s software development process.</td>
</tr>
<tr>
<td>Perform Security Training</td>
<td>Ensure project staff are trained in security concepts, and in role-specific security techniques.</td>
</tr>
</tbody>
</table>

# Practices Levels’ Description

<table>
<thead>
<tr>
<th>Tasks, Practices &amp; Activities</th>
<th>Characteristics for SECU ratings</th>
<th>Degrees</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
<th>Extra High</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Apply Secure Coding Standards</strong></td>
<td>Standards coverage</td>
<td>Basic (list of banned functions), moderate, extensive (proper use of APIs, memory sanitization, cryptography).</td>
<td>Ad-hoc secure coding</td>
<td>Address common vulnerabilities</td>
<td>Address common and off-nominal vulnerabilities</td>
<td>Address all vulnerabilities and some weakness</td>
<td>Coding to address all known vulnerabilities and weaknesses</td>
</tr>
<tr>
<td><strong>Perform Security Testing</strong></td>
<td>Testing rigour and coverage</td>
<td>Basic testing (simple edge cases and boundary conditions), basic testing derived from requirements and security features, derived from risk analysis with medium coverage, comprehensive tests derived from abuse cases, complete set of tests derived from abuse cases.</td>
<td>Ad-hoc security testing</td>
<td>Basic adversarial testing</td>
<td>Moderate adversarial testing driven with security requirements and security features.</td>
<td>Extensive adversarial testing driven by high security risks.</td>
<td>Rigorous adversarial testing driven by security risks and attack patterns.</td>
</tr>
<tr>
<td><strong>Apply Security Tooling</strong></td>
<td>Tools usage</td>
<td>Basic tool configuration, customized with tailored rules, able to detect malicious code.</td>
<td>Casual use of standard static analysis tool to identify security defects.</td>
<td>Basic use of static analysis tool to identify security defects.</td>
<td>Routine use of static analysis and penetration testing tools to identify security defects.</td>
<td>Extensive use of static analysis, penetration testing and black-box security testing tools.</td>
<td>Rigorous use of static analysis, penetration testing and black-box security testing tools with tailored rules.</td>
</tr>
</tbody>
</table>

... | ... | ... | ... | ... | ... | ... | ... |
# Resulting Rating Scale

<table>
<thead>
<tr>
<th>Protection Level</th>
<th>Security Requirements and Security Design</th>
<th>Secure Coding and Security Tools</th>
<th>Security Verification and Validation (V&amp;V)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LEVEL 0</strong></td>
<td>LEVEL 0 Protection is ad-hoc.</td>
<td>LEVEL 0 No secure coding and no use of static analysis tool.</td>
<td>LEVEL 0 None</td>
</tr>
<tr>
<td><strong>LEVEL 1</strong></td>
<td>LEVEL 1 Protect against casual or coincidental access by unauthenticated entities.</td>
<td>LEVEL 1 Basic vulnerabilities applicable to the software will be prevented with secure coding standards and/or detected through basic use of static analysis tools.</td>
<td>LEVEL 1 Basic adversarial testing and security code review. Basic penetration testing. Security V&amp;V activities conducted within the project.</td>
</tr>
<tr>
<td><strong>LEVEL 2</strong></td>
<td>LEVEL 2 Protect against intentional unauthenticated access by entities using simple means with low resources, generic skills and low motivation.</td>
<td>LEVEL 2 Known and critical vulnerabilities applicable to the software will be prevented with secure coding standards and/or detected through routine use of static analysis tools.</td>
<td>LEVEL 2 Moderate adversarial testing and security code review. Routine penetration testing. Security V&amp;V activities conducted by an independent group.</td>
</tr>
<tr>
<td><strong>LEVEL 3</strong></td>
<td>LEVEL 3 Protect against intentional unauthenticated access by entities using sophisticated means with moderate resources.</td>
<td>LEVEL 3 Extensive list of vulnerabilities and weaknesses applicable to the software will be prevented with secure coding standards and/or detected through extensive use of static analysis and black-box tools.</td>
<td>LEVEL 3 Extensive adversarial testing and security design/code review. Frequent and specialized penetration testing. Security V&amp;V activities conducted by an independent group at the organizational level.</td>
</tr>
<tr>
<td><strong>LEVEL 4</strong></td>
<td>LEVEL 4 Protect against intentional unauthenticated access by entities using sophisticated means with extended resources.</td>
<td>LEVEL 4 Very extensive list of vulnerabilities and weaknesses applicable to the software will be prevented with secure coding standards and/or detected through rigorous use of static analysis and black-box security testing tools with tailored rules. Employ formal methods in coding.</td>
<td>LEVEL 4 Rigorous adversarial testing and security design/code review. Exhaustive deep-dive analysis penetration testing. Use of formal verification and custom developed V&amp;V tools. Security V&amp;V activities conducted by an outside certified company.</td>
</tr>
</tbody>
</table>
Outline

• Secure Software Development Costs
• Scale Development
• Resulting Estimates from Security Experts
• Next Steps
Online Delphi

Facilitator

Experts

Submit estimates

Facilitator

Report results

Report

Request estimation

Send back summary of compiled results, clarify assumptions, adjust questions

Presented for the ICEAA 2021 Online Workshop - www.iceaaonline.com
Results from online Delphi

• September 2020
• Participants invited from the Software Security Group on LinkedIn
• 2 rounds
  • 17 participants
  • 14 participants
• 10.88 years average experience with Secure Software Development
• 11.06 years average experience with Software Estimation
Productivity Range*

Histograms for each group of security practices

* Productivity range is the ratio between the highest level (Level 4) and the lowest level of the scale (Level 0).
# Productivity Range

<table>
<thead>
<tr>
<th>Group</th>
<th>Average</th>
<th>Median</th>
<th>Standard Deviation</th>
<th>Coefficient of Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements and Design</td>
<td>1.957</td>
<td>1.5</td>
<td>1.093</td>
<td>56%</td>
</tr>
<tr>
<td>Coding and Tools</td>
<td>2.046</td>
<td>1.4</td>
<td>1.193</td>
<td>58%</td>
</tr>
<tr>
<td>Verification and Validation</td>
<td>2.561</td>
<td>1.75</td>
<td>2.335</td>
<td>91%</td>
</tr>
<tr>
<td><strong>Productivity Range</strong></td>
<td><strong>10.256</strong></td>
<td><strong>3.675</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Added Effort by Security Level
Based on median productivity range

SECU Effort Multiplier by Level

- Nominal: 1.000
- High: 1.385
- Very High: 1.917
- Extra High: 2.654
- Ultra High: 3.675
## Increase in Application Size

Estimates from 14 participants (only in 2\textsuperscript{nd} round)

<table>
<thead>
<tr>
<th></th>
<th>L1</th>
<th>L2</th>
<th>L3</th>
<th>L4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>High</td>
<td>Very High</td>
<td>Extra High</td>
<td>Ultra High</td>
</tr>
<tr>
<td>Average</td>
<td>1.170</td>
<td>1.393</td>
<td>1.668</td>
<td>1.914</td>
</tr>
<tr>
<td>Median</td>
<td>1.100</td>
<td>1.250</td>
<td>1.500</td>
<td>1.675</td>
</tr>
<tr>
<td>Std Deviation</td>
<td>0.125</td>
<td>0.366</td>
<td>0.590</td>
<td>0.839</td>
</tr>
<tr>
<td>Coefficient of Variation</td>
<td>11%</td>
<td>26%</td>
<td>35%</td>
<td>44%</td>
</tr>
</tbody>
</table>
Outline

• Secure Software Development Costs
• Scale Development
• Resulting Estimates from Security Experts
• Next Steps
Cost Estimation Model Building

Inputs

- Expert Estimates (Prior Data)
- Project Information (Sample Data)

Process

- Calculate means and variances
- Calculate model parameters by MLR
- Apply Bayesian analysis

Output

- Bayesian Estimates of the Parameters

MLR: Multiple Linear Regression
Proposed Cost Model Form

• Original COCOMO II equation

\[
Effort = A \cdot Size^E \cdot \prod_{i-1}^{n} EM_i
\]

• Addition of the parameter for secure software development level, and adjusted size:

\[
Effort = A \cdot Size^E \cdot SECU \cdot \prod_{i-1}^{n} EM_i
\]

Includes Security Functional Features

SECU: Security Cost Driver Effort multiplier for secure software development level
Data Collection

Security experts’ estimates for the security parameter

Estimation experts’ estimates for the security parameter

Wideband Delphi

Industry

Projects’ Data → Manual Data Collection Form

OSS

Projects’ Data → Automated Data Collection

Survey

Projects’ Data → Survey with Open Source Software (OSS) developers
Get involved

1) Participate in an online Delphi study
   • Share your estimates and assumptions anonymously
   • Compare your estimates with other participants

2) Participate in data collection
   • Provide sanitized data
   • Receive a version of the model calibrated for your organization

Contact: Elaine Venson
venson@usc.edu

Contact: Brad Clark (COCOMO III Project Coordinator)
clarkbk@usc.edu
Thank you!

Elaine Venson  
venson@usc.edu

Bradford Clark  
brad@software-metrics.com

Barry Boehm  
boehm@usc.edu
References


