REALIZING SOFTWARE SECURITY MATURITY: THE GROWING PAINS AND GAINS

Mark Stanislav
Director of Application Security
Duo Security

Kelby Ludwig
Senior Application Security Engineer
Duo Security
# Maturity Models

<table>
<thead>
<tr>
<th>Governance</th>
<th>Intelligence</th>
<th>SSDL Touchpoints</th>
<th>Deployment</th>
</tr>
</thead>
</table>

**BSIMM**

**SAMM**

---

**SAMM Overview**

- Business Functions: Governance, Construction, Verification, Operations

---

**RSA Conference 2018**
# BSIMM & SAMM: A Comparison(ish)

<table>
<thead>
<tr>
<th></th>
<th>BSIMM</th>
<th>SAMM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>Building Security in Maturity Model</td>
<td>Software Assurance Maturity Model</td>
</tr>
<tr>
<td><strong>In Use Since</strong></td>
<td>2008</td>
<td>2009 (1.0)</td>
</tr>
<tr>
<td><strong>Latest Release</strong></td>
<td>8 (September 2017)</td>
<td>1.5 (April 2017)</td>
</tr>
<tr>
<td><strong>Curated By</strong></td>
<td>Synopsys (Security Vendor)</td>
<td>OWASP (Community Organization)</td>
</tr>
<tr>
<td><strong>Model Basis</strong></td>
<td>Real-world, “in use” industry data</td>
<td>“Ideal state” via community input</td>
</tr>
<tr>
<td><strong># of Top-Level Groupings</strong></td>
<td>4 — Governance, Intelligence, SSDL Touchpoints, and Deployment</td>
<td>4 — Governance, Construction, Verification, and Operations</td>
</tr>
<tr>
<td><strong># of Activities</strong></td>
<td>113 across 12 sub-groupings</td>
<td>77 across 12 sub-groupings</td>
</tr>
</tbody>
</table>
Maturity Models

[SE2.4: 29] Use code signing.
The organization uses code signing for software published across trust boundaries. Code signing is particularly useful for protecting the integrity of software that leaves the organization's control, such as shrink-wrapped applications or thick clients. The fact that some mobile platforms require application code to be signed does not indicate institutional use of code signing.

B. Perform code signing for application components

Though often used with special-purpose software, code signing allows users and operators to perform integrity checks on software such that they can cryptographically verify the authenticity of a module or release. By signing software modules, the project team enables deployments to operate with a greater degree of assurance against any corruption or modification of the deployed software in its operating environment.

Signing code incurs overhead for management of signing credentials for the organization. An organization must follow safe key management processes to ensure the ongoing confidentiality of the signing keys. When dealing with any cryptographic keys, project stakeholders must also consider plans for dealing with common operational problems related to cryptography such as key rotation, key compromise, or key loss.

Since code signing is not appropriate for everything, architects and developers should work with security auditors and business stakeholders to determine which parts of the software should be signed. As projects evolve, this list should be reviewed with each release, especially when adding new modules or making changes to previously signed components.
Staffing for Success

1.6%  
Percentage of Software Security Group (SSG) Members to Software Engineers in BSIMM8’s Data Set

10.9%  
Percentage of Our Application Security Team Members to Our Product Engineering Staff
Application Security: Team Values

ENGINEERING IS FAMILY

Application Security will be adversarial in activity, but never in the relationship with our Engineering team members.

What this means:

- Empathetic and respectful engagement
- Empower engineers with knowledge
- Be available, be thoughtful, be patient

What this does not mean:
Application Security: Team Values

LOW FRICTION, HIGH VALUE

Application Security will look for key points in the SDLC that provide high value, with low friction, to increase security.

- **What this means:**
  - Less roadblocks, more roundabouts
  - Be mindful of overhead on Engineers
  - Be creative in building better security

- **What this does not mean:**

---

#RSAC

Application Security: Team Values

BUILD A PAVED ROAD

Application Security will build and promote standard capabilities that accelerate engineers with clear support & benefits.

- What this means:
  - Guardrails so engineers feel confident
  - Help to accelerate innovation & output
  - More time to spend on “hard” problems

- What this does not mean:
Application Security: Team Values

HOW COULD IT GO RIGHT?

Application Security will ensure Engineering is enabled & supported to lead innovation, even for hard security challenges.

- **What this means:**
  - We’re enablers, not the team of “No”
  - Our titles contain ‘Engineer’ for a reason
  - Be up for the challenge; no fatalists here

- **What this does not mean:**

  "We're doomed!"
NO CODE LEFT BEHIND

Application Security is committed to ensuring that no code is forgotten about and that our security testing accounts for it.

- **What this means:**
  - Don’t just focus on the new & shiny
  - Understand the full software inventory
  - “Old” code changes in “new” deploys

- **What this does not mean:**
Duo Application Security Maturity Model (DASMM)

**Governance**
- Strategy & Metrics
- Policy & Compliance
- Education & Guidance

**Engineering**
- Software Requirements
- Software Architecture
- Threat Assessment

**Verification**
- Code Review
- Software Testing
- Design Review

**Operations**
- Defect Management
- Deployment Composition

54 Activities | 46 Activities | 55 Activities | 35 Activities

Leveraging Industry Maturity Models with the Ability to Customize
# DASMM: Tracking Program Maturity

### Coverage

<table>
<thead>
<tr>
<th>Coverage</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Consistent coverage and very mature practices</td>
</tr>
<tr>
<td>0.5</td>
<td>Inconsistent coverage and/or partially mature practices</td>
</tr>
<tr>
<td>0.2</td>
<td>Minor coverage and/or weak practices</td>
</tr>
<tr>
<td>0</td>
<td>Non-existent coverage and/or immature practices</td>
</tr>
</tbody>
</table>

### Priority

<table>
<thead>
<tr>
<th>Priority</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>An activity vital to the success of the AppSec program</td>
</tr>
<tr>
<td>2</td>
<td>Highly valuable activities that notably increase maturity</td>
</tr>
<tr>
<td>3</td>
<td>Supplemental to program goals, but not key to success</td>
</tr>
<tr>
<td>4</td>
<td>There is no intention to adopt this activity in the future</td>
</tr>
</tbody>
</table>

* Spoiler Alert: Fake Data

---

SM01 | BSIMM - SM1.1 | Publish process (roles, responsibilities, plan), evolve as necessary. |
SM02 | BSIMM - SM1.2 | Create evangelism role and perform internal marketing. |
SM03 | BSIMM - SM1.3 | Educate executives. |
SM04 | BSIMM - SM1.4 | Identify gate locations, gather necessary artifacts. |
SM05 | BSIMM - SM2.1 | Publish data about software security internally. |
SM06 | BSIMM - SM2.2 | Enforce gates with measurements and track exceptions. |
SM07 | BSIMM - SM2.3 | Create or grow a satellite. |
SM08 | BSIMM - SM2.5 | Identify metrics and use them to drive budgets. |
SM09 | BSIMM - SM2.6 | Require security sign-off. |

---

1  | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
0.5 | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
0.2 | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
0.2 | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
0.5 | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
0  | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
0.2 | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
0.5 | Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt |
# Building a Program

## Standardize

<table>
<thead>
<tr>
<th>Foundational</th>
<th>Descriptive</th>
<th>Functional</th>
</tr>
</thead>
<tbody>
<tr>
<td>- OWASP SAMM</td>
<td>- Bugcrowd VRT</td>
<td>FIRST PSIRT Framework</td>
</tr>
<tr>
<td>- Synopsys BSIMM</td>
<td>- Microsoft STRIDE</td>
<td>OWASP ASVS</td>
</tr>
<tr>
<td>- Microsoft SDL</td>
<td>- Microsoft DREAD</td>
<td>ISO 30111 &amp; 29147</td>
</tr>
</tbody>
</table>
Building a Program

Strong Collaboration

**Quality Assurance**
- Maximize test coverage
- Shared technical tooling
- Triage security bugs

**Product Team**
- Advise on key trends
- Assess early design risk
- Understand our users

**Compliance**
- Vendor assessments
- RFP questionnaires
- Support audit needs
Give Back to the Community

**Content**
- Present at conferences
- Author blog posts
- Respond to the press

**Industry Contributions**
- Influence relevant standards
- Build community events
- Perform security research
Security Development Lifecycle (SDL)

Training
- Requirements
- Design
- Implementation
- Verification
- Release
- Response

Training
- Requirements
- Design
- Implementation
- Verification
- Release
- Response
**Security Development Lifecycle (SDL)**

- **Engineering-focused “Security Skills & Interest” survey**
  - All new Engineering hires fill out this form to influence our program focus

- **Hands-on formal training & guest speakers**
  - Tailored courses developed internally and 3rd-party specialized training

- **Informal gamified training**
  - Internal CTFs and Elevation of Privilege (EoP) card-game tournaments
SECURITY DESIGN REVIEWS

Evaluates the security architecture of an application's overall composition.

- **Benefits to Engineers**
  - Early, efficient clarity on secure design
  - Reduces likelihood of major refactoring later
  - Provides early AppSec team awareness
  - Allows for highly interactive engagement

- **Possible Deliverables**
  - Real-time feedback
  - Formalized review artifacts
  - Software security requirements
THREAT MODELING

Reviewing a software design to enumerate threats and contextualize their real risk.

- **Benefits to Engineers**
  - Thoughtful evaluation of attack surface
  - Development of a better “attacker mindset”
  - Useful insights for cost/benefit analysis
  - Allows for more strategic risk mitigation

- **Possible Deliverables**
  - Data flow diagrams
  - Threat enumeration details
  - Interactive whiteboarding
CODE AUDITING

Point-in-time analysis of how implemented code has met the intent of security engineering principles, standards, and guidelines as defined for the project’s goals.

- **Benefits to Engineers**
  - Prompt remediation of security anti-patterns
  - Collaborative review of code in increments
  - Focused attention to “security quality” of work
  - Bite-sized security education opportunities

- **Possible Deliverables**
  - Well-documented remediation patches
  - Detailed technical writeups of vulnerabilities
  - Improved security test coverage
Security Development Lifecycle (SDL)

**SECURITY ASSESSMENT**

Comprehensive review of software's total security composition, usually at major lifecycle inflection points (e.g. new release, feature update, major code refactor).

- **Benefits to Engineers**
  - Holistic review of entire in-scope code base
  - Analyzes the integrated security properties
  - New or updated view of threat model artifacts
  - Good “gut check” before a major release

- **Possible Deliverables**
  - Threat modeling asset updates
  - A comprehensive assessment report
  - Detailed technical writeups of vulnerabilities
Security Development Lifecycle (SDL)

- **Product Security Advisory (PSA) process**
  - Modeled after ISO/IEC 30111:2013

- **Coordinated vulnerability disclosure policy**
  - Modeled after ISO/IEC 29147:2014
  - Our contact details are published on our web site, including a GPG key

- **FIRST PSIRT Framework**
  - Being finalized after a recent v1.0 RFC period, during which we submitted feedback
Ad-hoc Help: Easy Mode

- Review small code diffs
- One-off Slack conversations
- Issue tracker subscriptions
- Forwarding us an email thread
- Walking up to our desk with beer
AppSec Team “Office Hours”

- 1 hour of weekly time with AppSec
- Published on engineer calendars
- Reminders via Slack & in-person
- Open-ended discussion and Q&A
- Often results in “next step” outcomes
- Realizes low-friction, high-value
Intake Process

- Intake form is submitted by an engineer
- Timeline and AppSec resources forecasted
- Details added to the security activity board

The Intake Form Will Receive…
- Which activity was requested and why
- Overview of the request’s scope
- Links to all relevant project artifacts
- Activity timeline and point of contact

Application Security Intake

Please fill out this form if your team requires assistance with application security-related activities, such as a security design review, threat modeling exercise, code audit, or application security assessment (internal/external).

Please provide FULL names for each person mentioned to ensure clarity for communication.

The name, username and photo associated with your Google account will be recorded when you upload files and submit this form. Not kludwig@duosecurity.com? Switch account

* Required

What activity would you like to accomplish? *

Choose

Please provide a brief overview of the scope *

Your answer
### Execution Management and Scheduling

#### Active

<table>
<thead>
<tr>
<th>Activity</th>
<th>AppSec Lead</th>
<th>Requester</th>
<th>Kick Off</th>
<th>Work Started</th>
<th>Work Finished</th>
<th>Report Delivery</th>
<th>Debrief</th>
<th>Duration</th>
<th>Report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Super Important Feature</td>
<td>Security Assessment</td>
<td></td>
<td>Oct 03</td>
<td>Oct 06</td>
<td>Oct 14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+ Create a New Pulse</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Backlog

<table>
<thead>
<tr>
<th>Activity</th>
<th>AppSec Lead</th>
<th>Requester</th>
<th>Kick Off</th>
<th>Work Started</th>
<th>Work Finished</th>
<th>Report Delivery</th>
<th>Debrief</th>
<th>Duration</th>
<th>Report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less important feature</td>
<td>Threat Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+ Create a New Pulse</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Completed

<table>
<thead>
<tr>
<th>Activity</th>
<th>AppSec Lead</th>
<th>Requester</th>
<th>Kick Off</th>
<th>Work Started</th>
<th>Work Finished</th>
<th>Report Delivery</th>
<th>Debrief</th>
<th>Duration</th>
<th>Report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Older Cool Feature</td>
<td>Threat Model</td>
<td></td>
<td>Sep 06</td>
<td>Sep 07</td>
<td>Sep 17</td>
<td>Sep 21</td>
<td>Sep 21</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Similar to an internal consultancy
- Easy and transparent scheduling
- Simple and repeatable process
- Helps answer statusing questions
#RSAC

## 1st Party Execution: Kick-Off Checklist

### Pre-Kickoff Items

These items should be completed prior to the kickoff meeting. If there are any questions or concerns about providing what is listed, please let AppSec know before the kickoff meeting.

<table>
<thead>
<tr>
<th>Status</th>
<th>Owner</th>
<th>Request</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>appsec</td>
<td>Review documents provided by the AppSec intake form.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>appsec</td>
<td>Based on the review of the documents in the intake form, research any additional documentation that may be useful.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>appsec</td>
<td>Note any additional questions or action items that should be covered during kick off in this document.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>eng team</td>
<td>What are your expectations for this review? What do want AppSec to provide after all is said and done?</td>
<td></td>
</tr>
</tbody>
</table>
|        | eng team | "The only secure computer is one that’s unplugged, locked in a safe, and buried 20 feet under the ground in a secret location... and I’m not even too sure about that one."
We don’t want to break required functionality in order to secure a feature. Please provide any documents that may give us insight to problems that this feature solves. (e.g. business and functional requirements, user stories, etc.) |       |
|        | eng team | A large part of AppSec work is understanding how a feature works. Please link any existing technical documentation that may give AppSec that insight. (e.g. architecture diagrams, design documents, specifications). |       |

- Shared responsibility between AppSec and Engineering
- **Ensures...**
  - Security activities start on-time
  - Goals & expectations are aligned
  - Clarity on perceived risks
  - AppSec process consistency
- Acts as a single source of truth for information
One Report; Many Benefits

- **Perspective**: A formal deliverable sets the tone for a level of quality & completeness of the work

- **Context**: Holistic view of key activity properties

- **Compliance**: Report aggregates necessary information needed for auditors and customers

- **Historic Value**: Easily allows differential analysis of year-over-year results for a given codebase

- **Debrief**: Ensures that all stakeholders have the complete picture of the security activity’s output
Now, Take Action!

- **Next Week…**
  - Read through OWASP SAMM & Synopsys BSIMM — choose a framework
  - Perform a comprehensive software inventory to determine what’s in scope

- **Within Three Months…**
  - Perform a gap analysis against BSIMM or SAMM of your program
  - Provide an interactive Application Security training to engineers
  - Begin operating across the Security Development Lifecycle (SDL)
THANK YOU!

Mark Stanislav
Email: mstanislav@duo.com
Web: uncompiled.com

Kelby Ludwig
Email: kludwig@duo.com
Web: kel.bz