RSAConference2018
San Francisco | April 16 – 20 | Moscone Center

SESSION ID: ASEC-T10

REALIZING SOFTWARE SECURITY MATURITY: THE GROWING PAINS AND GAINS

Mark Stanislav

Director of Application Security
Duo Security

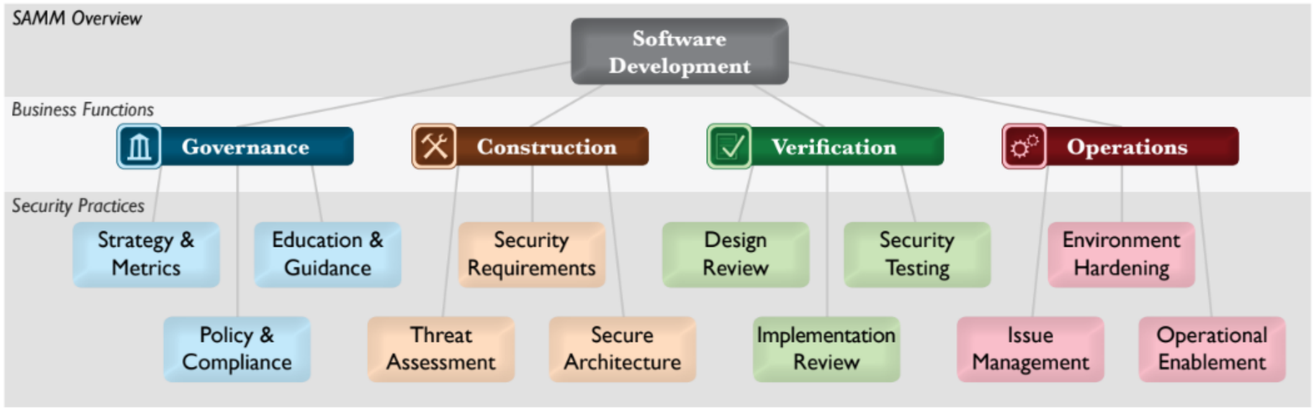
Kelby Ludwig

Senior Application Security Engineer
Duo Security

Maturity Models

BSIMM

SAMM



BSIMM & SAMM: A Comparison(ish)

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

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.

BSIMM
DESCRIPTIVE

SAMM
PRESCRIPTIVE

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:

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:

Application Security: Team Values

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	Engineering	Verification	Operations
- Strategy & Metrics - Policy & Compliance - Education & Guidance	- Software Requirements - Software Architecture - Threat Assessment	- Code Review - Software Testing - Design Review	- Defect Management - Deployment Composition
54 Activities	46 Activities	55 Activities	35 Activities

BSIMM
Building Security In Maturity Model

OWASP
Open Web Application Security Project

Leveraging Industry Maturity Models with the Ability to Customize

DASMM: Tracking Program Maturity

Coverage

Coverage	Definition
1	Consistent coverage and very mature practices
0.5	Inconsistent coverage and/or partially mature practices
0.2	Minor coverage and/or weak practices
0	Non-existent coverage and/or immature practices

Priority

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

SM01	BSIMM - SM1.1	Publish process (roles, responsibilities, plan), evolve as necessary.	1	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	1
SM02	BSIMM - SM1.2	Create evangelism role and perform internal marketing.	0.5	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	2
SM03	BSIMM - SM1.3	Educate executives.	0.2	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	3
SM04	BSIMM - SM1.4	Identify gate locations, gather necessary artifacts.	0.2	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	4
SM05	BSIMM - SM2.1	Publish data about software security internally.	0.5	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	2
SM06	BSIMM - SM2.2	Enforce gates with measurements and track exceptions.	1	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	4
SM07	BSIMM - SM2.3	Create or grow a satellite.	0	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	1
SM08	BSIMM - SM2.5	Identify metrics and use them to drive budgets.	0.2	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	4
SM09	BSIMM - SM2.6	Require security sign-off.	0.5	Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt	4

* Spoiler Alert: Fake Data

Building a Program

BSIMM

VRT

Standardize

Foundational
- OWASP SAMM

- Synopsys BSIMM

- Microsoft SDL

Descriptive
- Bugcrowd VRT

- Microsoft STRIDE

- Microsoft DREAD

Functional
FIRST PSIRT Framework

OWASP ASVS

ISO 30111 & 29147

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

Building a Program

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 Development Lifecycle (SDL)

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

Security Development Lifecycle (SDL)

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

Security Development Lifecycle (SDL)

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

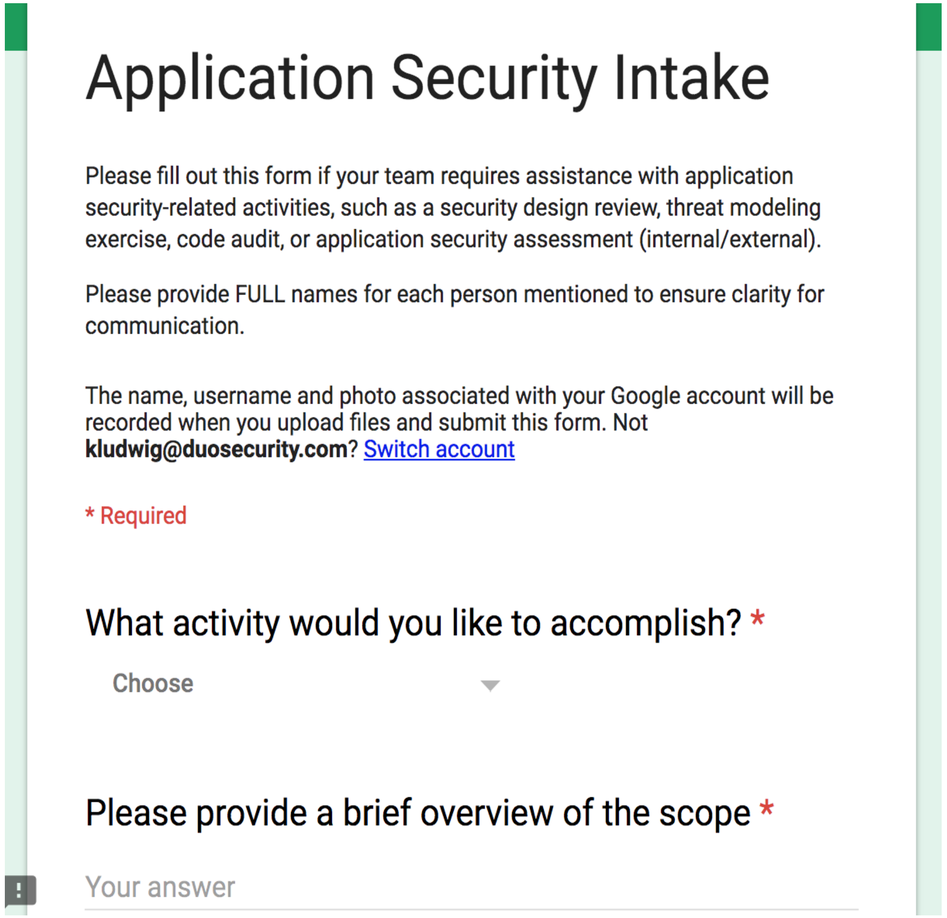
- Open-ended discussion and Q&A

- 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	Activity	AppSec Lead	Requester	Kick Off	Work Started	Work Finished	Report Delivery	Debrief	Duration	Report
Super Important Feature	Security Assessment			Oct 03	Oct 06	Oct 14				
+ Create a New Pulse										

Backlog	Activity	AppSec Lead	Requester	Kick Off	Work Started	Work Finished	Report Delivery	Debrief	Duration	Report
Less important feature	Threat Model									
+ Create a New Pulse										

Completed	Activity	AppSec Lead	Requester	Kick Off	Work Started	Work Finished	Report Delivery	Debrief	Duration	Report
Older Cool Feature	Threat Model			Sep 06	Sep 07	Sep 17	Sep 21	Sep 21	Sep 6 - 21	https://www.f...
Older Older Cool Feature	Security Design Review			Aug 14	Aug 14	Aug 25	Sep 01	Sep 01	Aug 14 - Sep 1	https://www.f...
Sketchy Feature	Code Auditing			Aug 01	Aug 03	Aug 09	Aug 14	Aug 16	Aug 1 - 16	https://www.f...

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

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.**

Status	Owner	Request	Notes
	appsec	Review documents provided by the AppSec intake form.	
	appsec	Based on the review of the documents in the intake form, research any additional documentation that may be useful.	
	appsec	Note any additional questions or action items that should be covered during kick off in this document.	
	eng team	What are your expectations for this review? What do want AppSec to provide after all is said and done?	
	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


Duo Security - Application Security

Juice Shop

Application Security Report

Assessment Start Date: 08/15/2017

Assessment End Date: 08/18/2017

Report Delivery Date: 08/18/2017

Assessment Type(s): Code Audit, Security Assessment

Author:

- Matthew Fanto, Senior Application Security Engineer

Contributor:

- Bjoern Kimminich, Juice Shop Creator

- 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