snyk

OWASP SAMM

Software Assurance Maturity Model

Agenda

- Introduction
- OWASP SAMM
    - Introduction
    - Methodology
        - Prepare
        - Assess
        - Set the Target
        - Implement
        - Rollout
- Q&A

INTRODUCTION

Mathias Conradt

Principal Solutions Engineer, **Snyk**

OWASP Member

20+ years in **project business**
Software engineering and consulting
(PRINCE2, ITIL, Scrum certified)

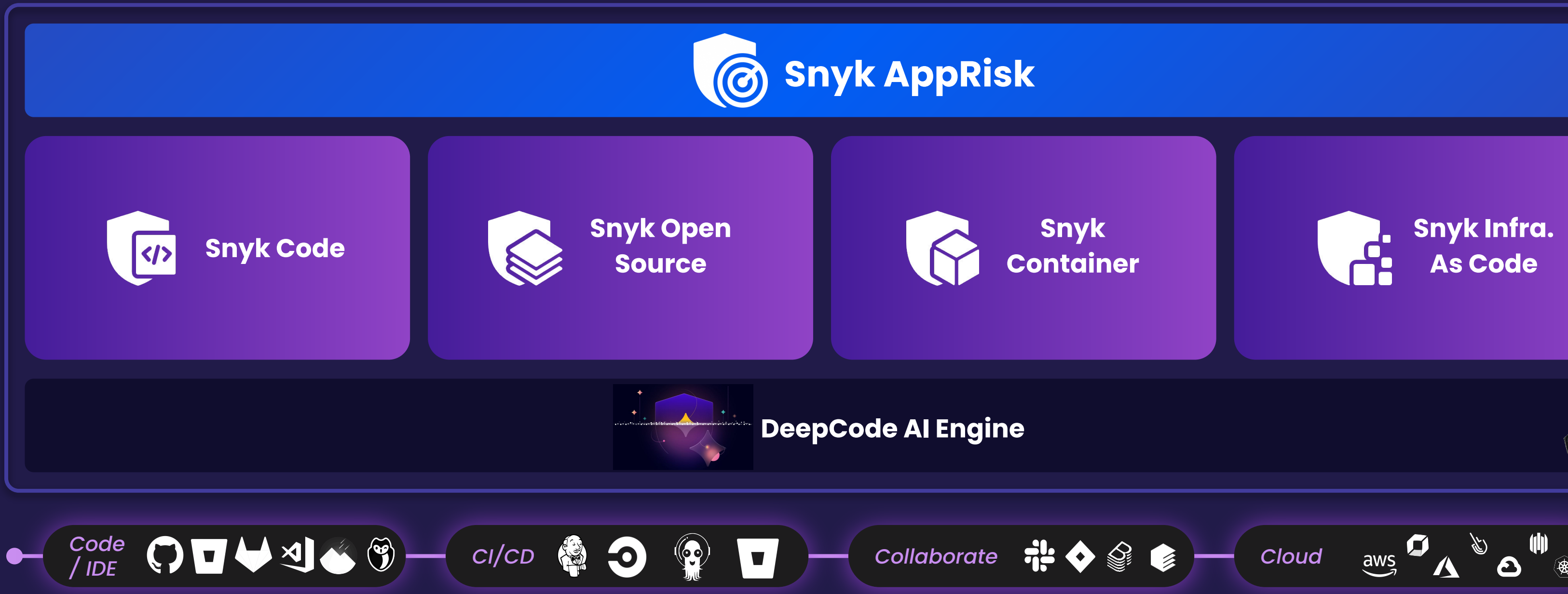
5+ years in **application security**
(Identity & Access Management, DevSecOps)

Open Source and Open Knowledge Advocate

https://www.linkedin.com/in/mathiasconradt/

Snyk Developer Security Platform

with unmatched speed, accuracy, coverage, and ease of use



24/7/365 **Customer Success & Professional Services** to drive adoption

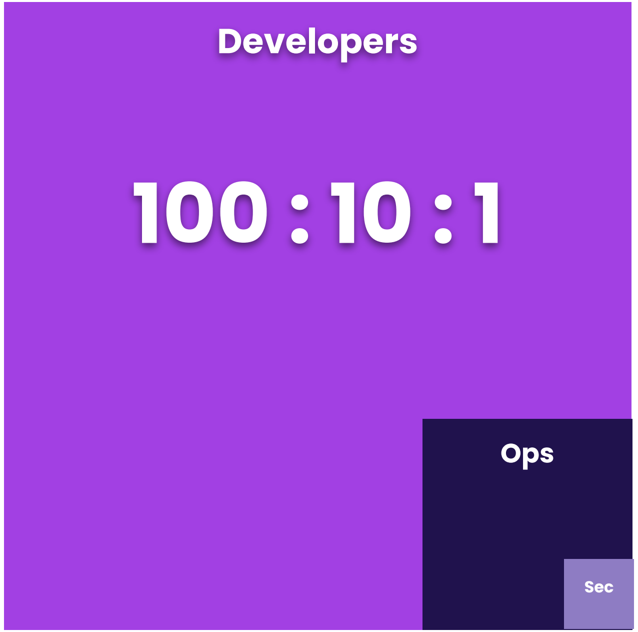
The only way to scale security is to empower developers

"The ratio of engineers in Development, Operations, and Infosec in a typical technology organization is 100:10:1.

When Infosec is that outnumbered, without automation and integrating information security into the daily work of Dev and Ops, Infosec can only do compliance checking, which is the opposite of security engineering - and besides, it also makes everyone hate us."

- Gene Kim

Co-Author of The DevOps Handbook



Developers

100 : 10 : 1

Ops

Sec

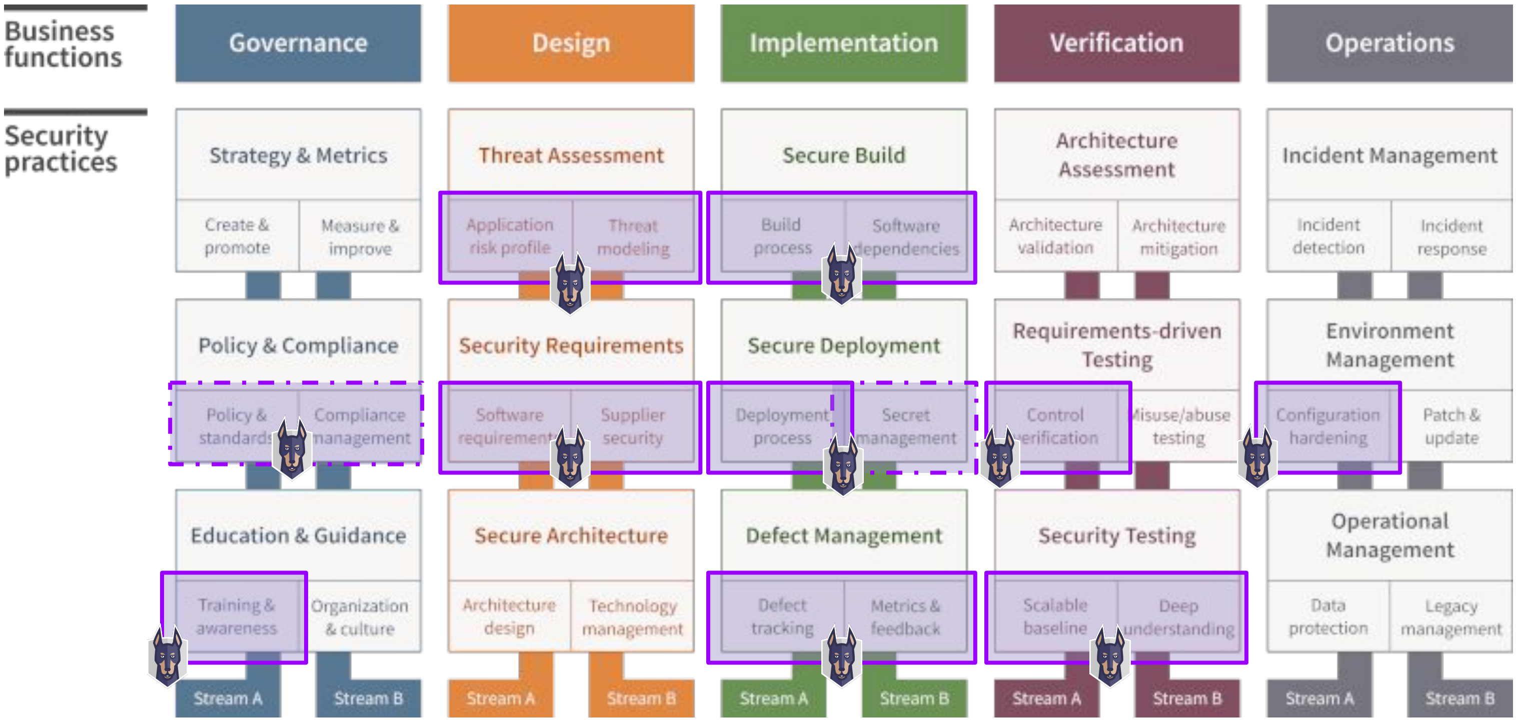
DevSecOps is the way!

Motivation

It's always about People, Processes, Technology (Tools)

- Snyk aims to be a **Trusted Advisor** to its customers when it comes to DevSecOps and Application Security Programs.
- **Win-Win**: Maturing DevSecOps will benefit both sides.

Supporting organizations that have **mature DevOps** processes in place but are just **starting** out with **DevSecOps** (and without any framework or model in place).



Security Framework & Maturity Model Landscape

OWASP SAMM

NIST CSF 2.0

DSOMM

SAFECode

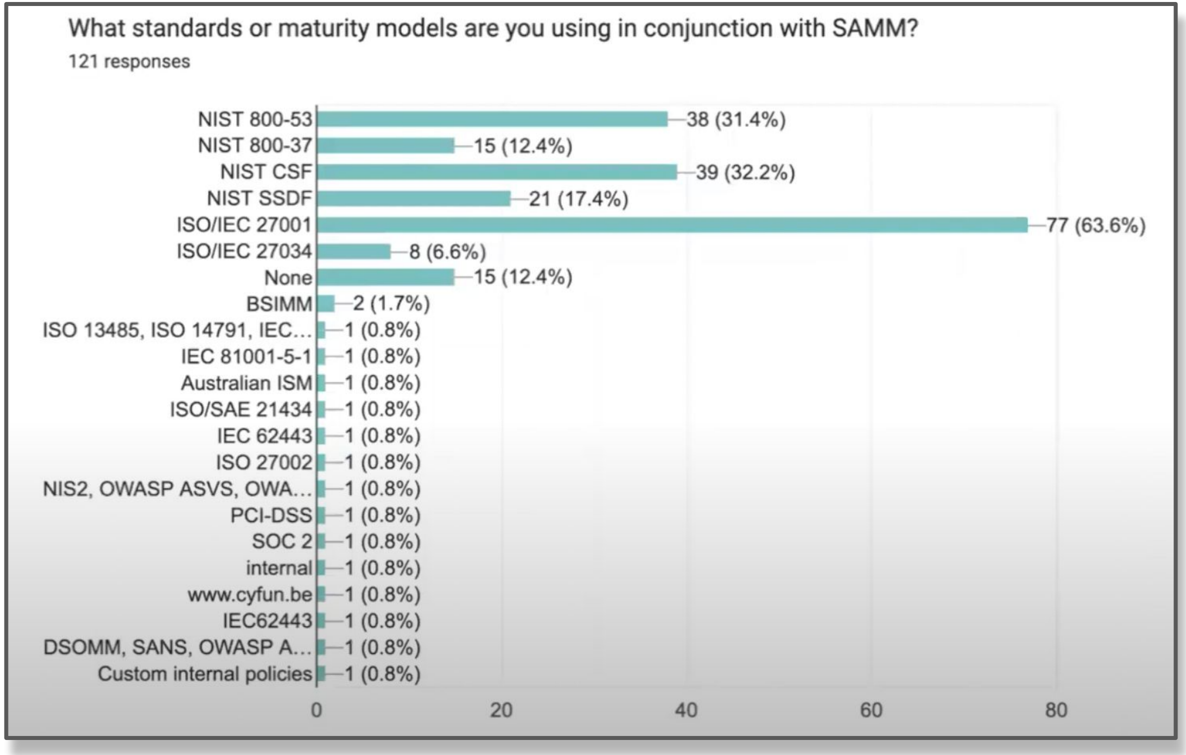
NIST CSF

ISO 27001

BSIMM

ASVS

COBIT®
An ISACA Framework


What standards or maturity models are you using in conjunction with SAMM?

121 responses

Why a Framework?

- Follow a structured approach
- “You can’t manage what you can’t measure”
- Not reinventing the wheel
- Learn from others, field-proven best-practices

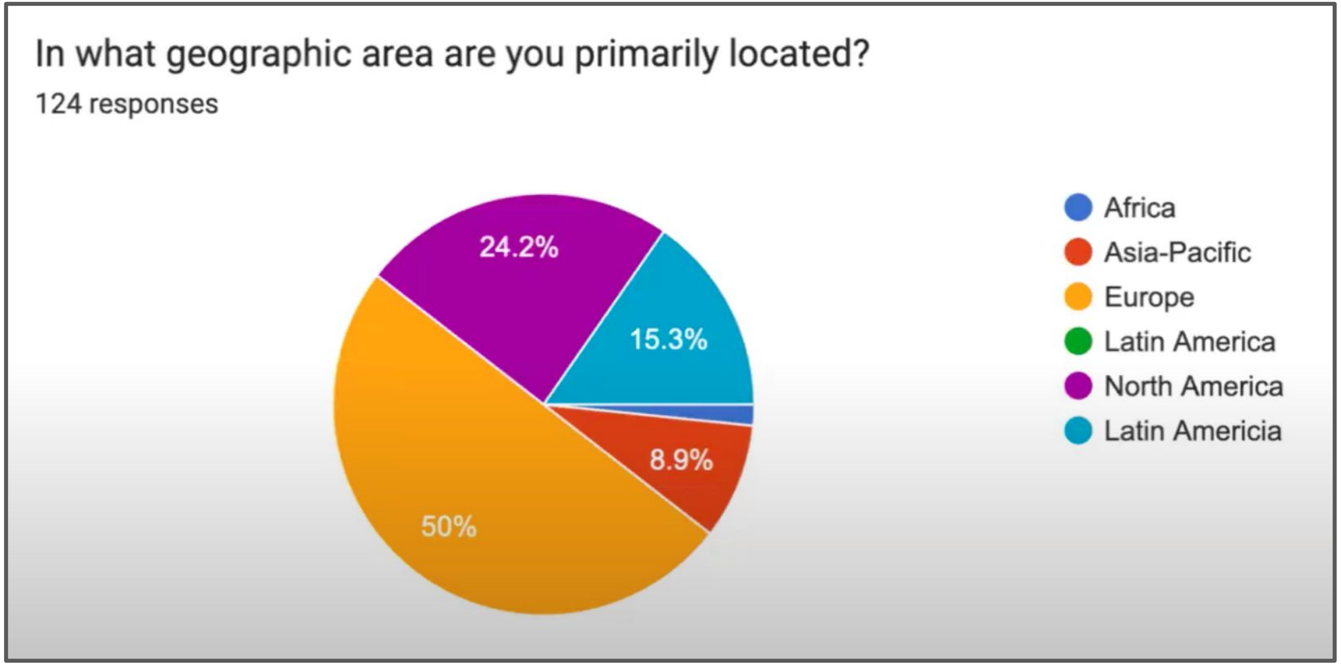
Selection Criterias

- **Measurable**
  Defined maturity levels across security practices
- **Actionable**
  Clear pathways for improving maturity levels
- **Versatile**
  Technology, process, and organization agnostic
- **Open**
  Established non-proprietary with a large community

OWASP SAMM

- OWASP Software Assurance Maturity Model (SAMM)
- OpenSAMM 1.0 (2009), OWASP SAMM 1.1 (2016)+
- Focus on Application Security
- Flagship project at OWASP
- Prescriptive in Nature (as opposed to descriptive / BSIMM)

The mission of OWASP Software Assurance Maturity Model (SAMM) is to be the prime maturity model for software assurance that provides an effective and ***measurable*** *way for all types of organizations to analyze and* ***improve*** *their software security posture. OWASP SAMM supports the complete software lifecycle, including development and acquisition, and is* ***technology and process agnostic****. It is intentionally built to be evolutive and risk-driven in nature.*



In what geographic area are you primarily located?

124 responses

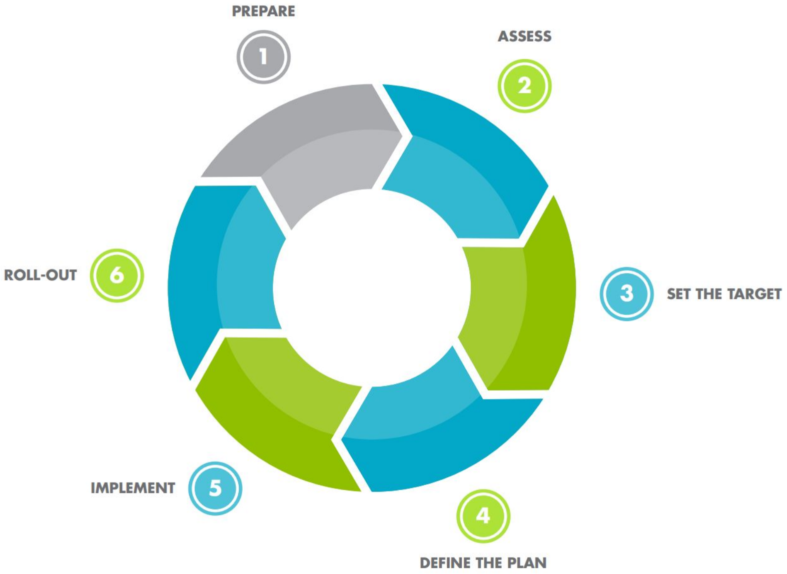
Why would you use SAMM?

- To have a “holistic” and structured approach to application security
- [As a CISO] to have your story resonate at the level of management
- [As a Developer] to get rid of “it’s a developer’s problem” mentality, which it’s not!
- [As a Project Manager] to get to a shift-left approach to increase efficiency and predictability of software delivery
- [As a Client] to understand how your supplier is performing

The structure and setup of OWASP SAMM is made to support:

1. the **assessment** of the current software assurance posture,
2. the definition of the **strategy** (i.e. the target) that the organization should implement,
3. the formulation of an implementation **roadmap** of how to get there, and
4. prescriptive advice on how to **implement** particular activities.

OWASP SAMM Project Cycle

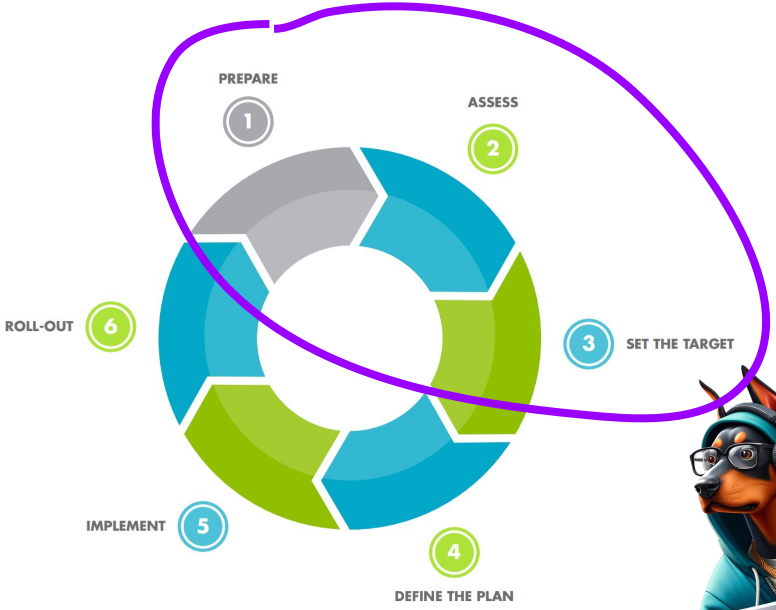


PREPARE | ASSESS | SET THE TARGET | DEFINE THE PLAN | IMPLEMENT | ROLL OUT

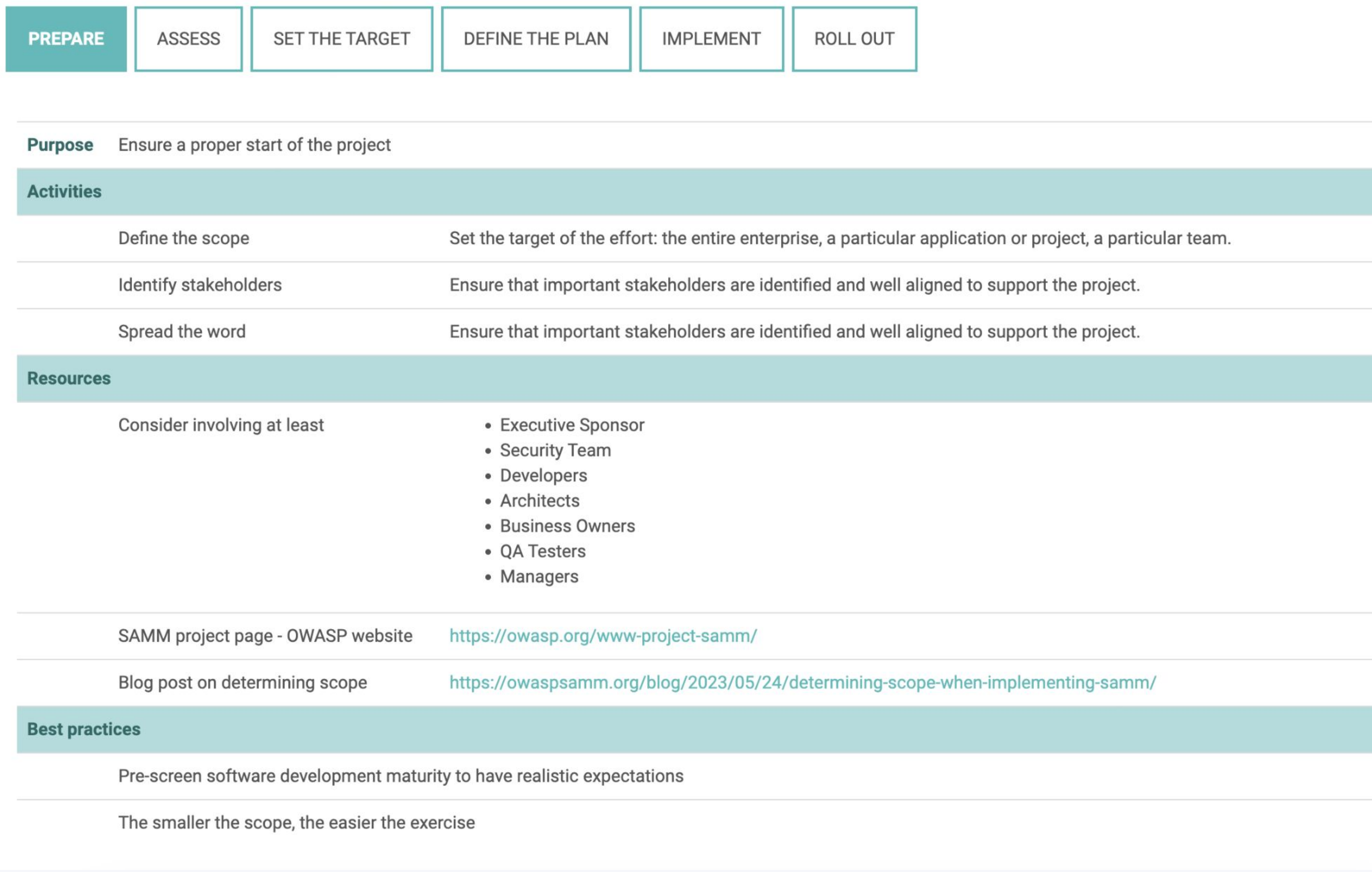
A Typical Kick-Off.

- Establishing **Assessment Scope**
- **Methodology**
- Assessing **Governance**
- Assessing **Design**
- Assessing **Implementation**
- Assessing **Verification**
- Assessing **Operations**
- Setting **Improvement Targets**

Usually 1-3 day session.

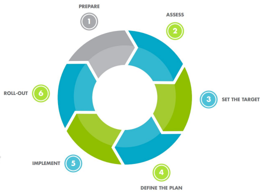


PREPARE



PREPARE | ASSESS | SET THE TARGET | DEFINE THE PLAN | IMPLEMENT | ROLL OUT

Purpose	Ensure a proper start of the project	
Activities		
	Define the scope	Set the target of the effort: the entire enterprise, a particular application or project, a particular team.
	Identify stakeholders	Ensure that important stakeholders are identified and well aligned to support the project.
	Spread the word	Ensure that important stakeholders are identified and well aligned to support the project.
Resources		
	Consider involving at least	• Executive Sponsor • Security Team • Developers • Architects • Business Owners • QA Testers • Managers
	SAMM project page - OWASP website	https://owasp.org/www-project-samm/
	Blog post on determining scope	https://owaspsamm.org/blog/2023/05/24/determining-scope-when-implementing-samm/
Best practices		
	Pre-screen software development maturity to have realistic expectations	
	The smaller the scope, the easier the exercise	



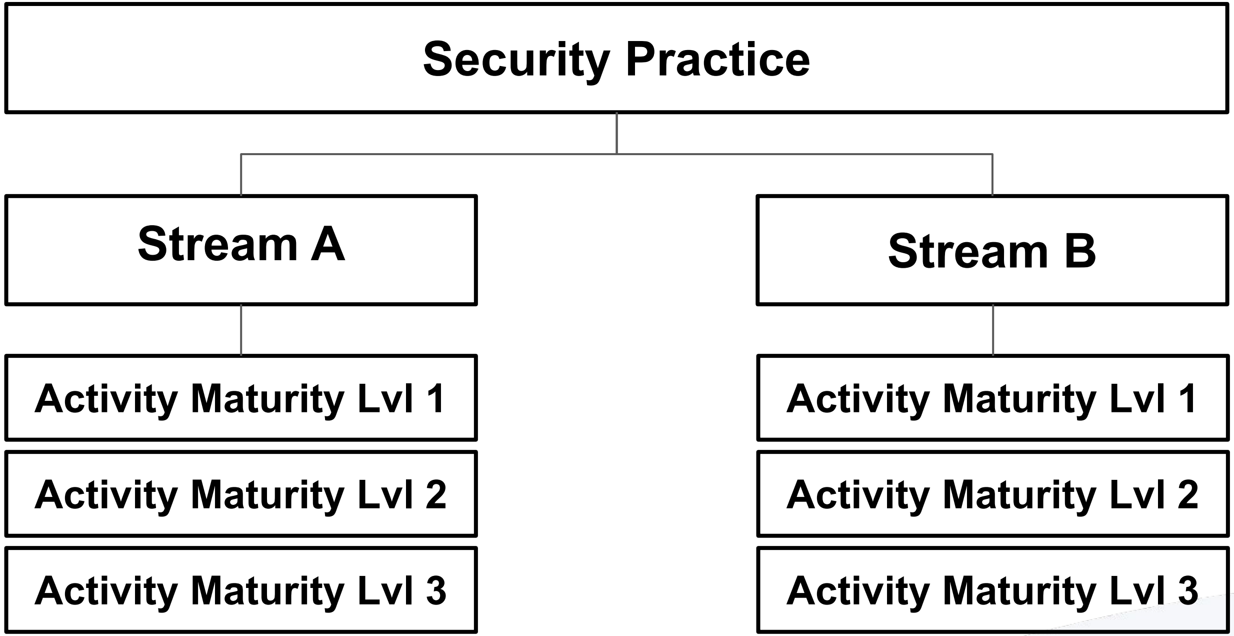
Considerations picking a Framework

- Do you agree on the framework?
- Do you agree on the model/content?
  - Does it need customization?
  - Customization vs. accepting low maturity acceptance
- Is Budget available and planned in?
  - Tool licenses, internal man hours, external consultants
- Educate all Stakeholders

Resources

- Website: https://owaspsamm.org
- Github: https://github.com/owaspsamm
- Slack: OWASP - #project-samm
- Youtube:
  - https://www.youtube.com/@owaspsamm
  - https://www.youtube.com/@codificcom
- Fundamentals Course
  - **https://owaspsamm.thinkific.com/courses/samm**
  - **https://www.youtube.com/playlist?list=PLBxrzm7KYaoESVEINboWn-_osqL1A5MLI**
- Monthly Zoom Call

SAMM Model Structure



Security Practice

Stream A

Activity Maturity Lvl 1

Activity Maturity Lvl 2

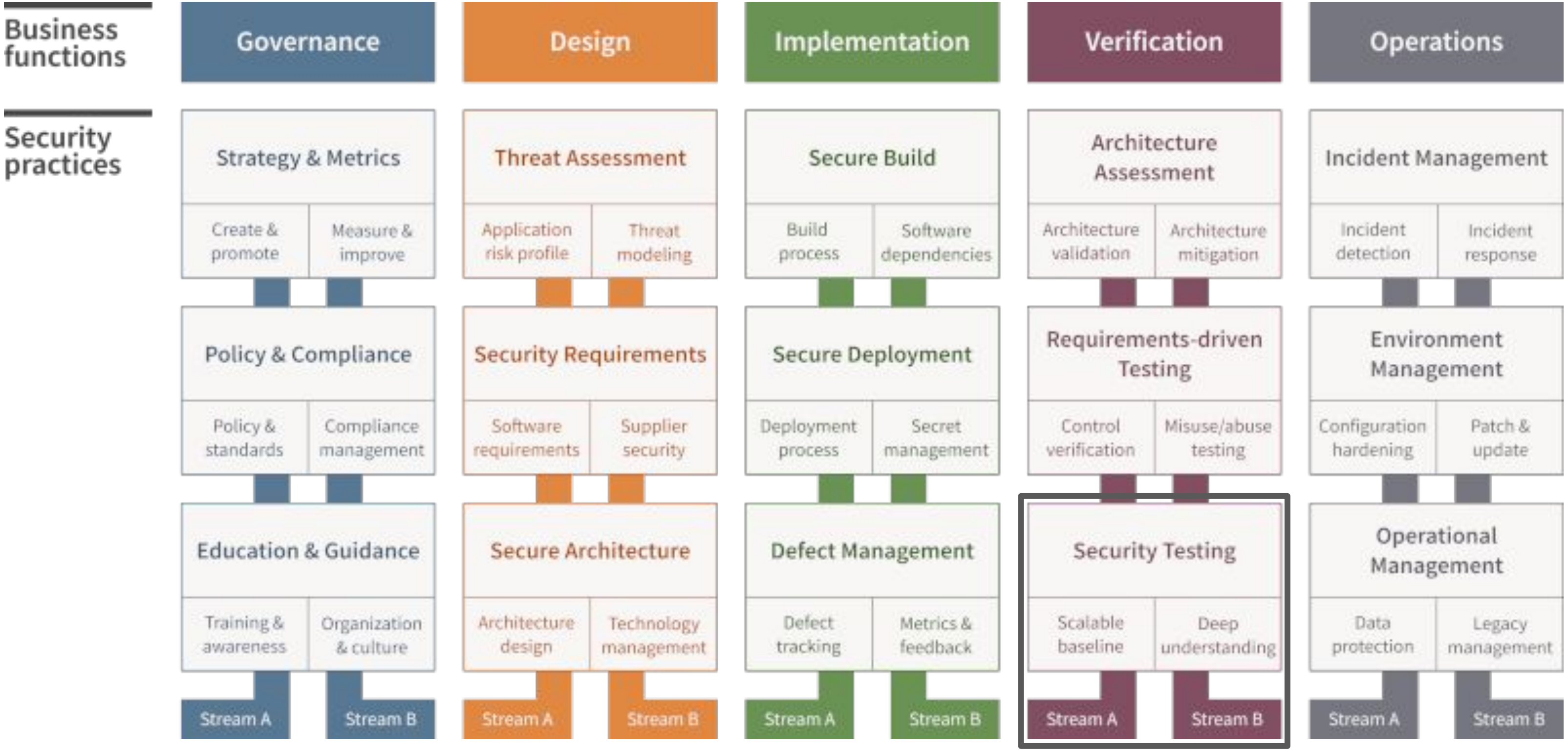
Activity Maturity Lvl 3

Stream B

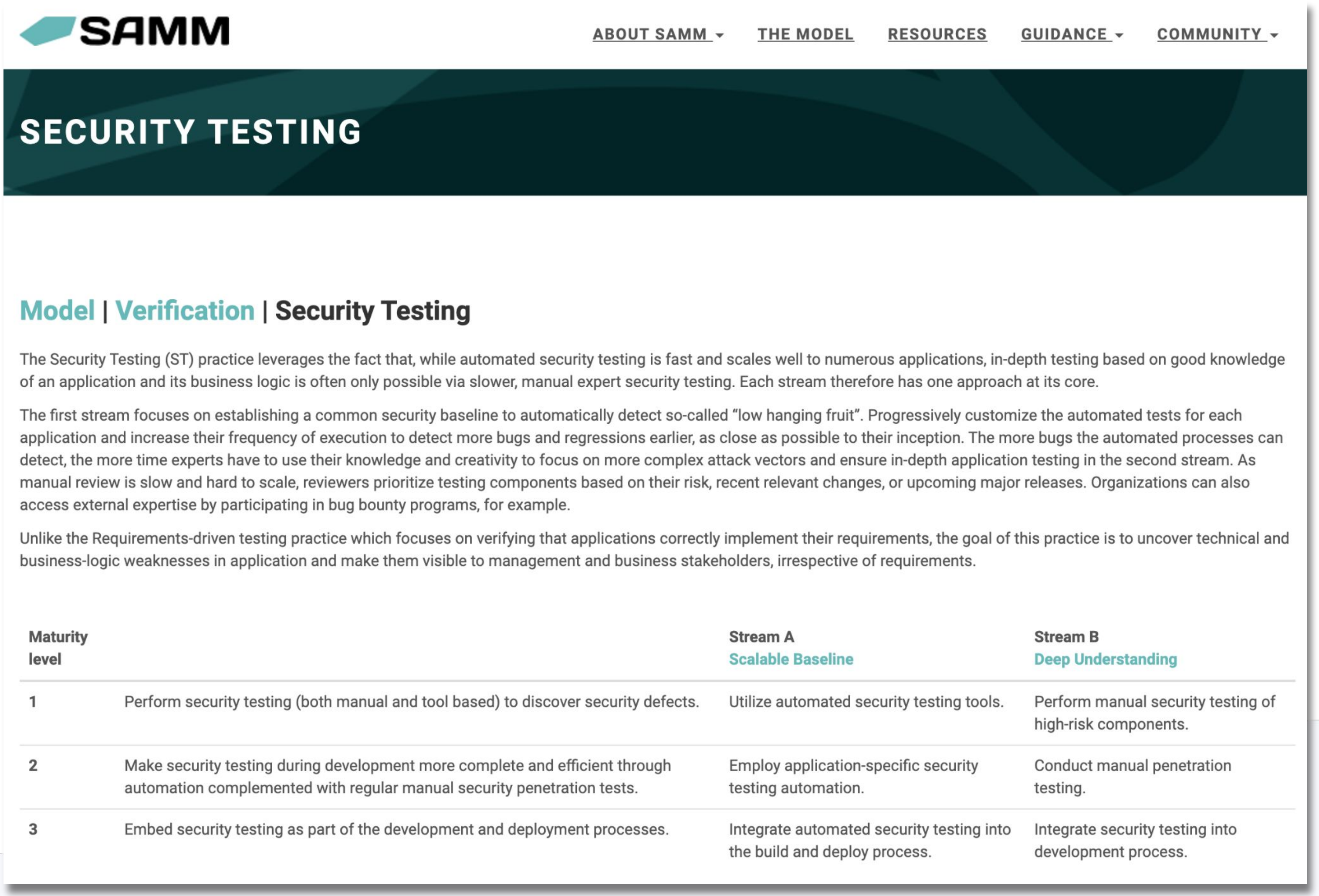
Activity Maturity Lvl 1

Activity Maturity Lvl 2

Activity Maturity Lvl 3



Business functions	Governance	Design	Implementation	Verification	Operations
Security practices	Strategy & Metrics Create & promote Measure & improve	Threat Assessment Application risk profile Threat modeling	Secure Build Build process Software dependencies	Architecture Assessment Architecture validation Architecture mitigation	Incident Management Incident detection Incident response
	Policy & Compliance Policy & standards Compliance management	Security Requirements Software requirements Supplier security	Secure Deployment Deployment process Secret management	Requirements-driven Testing Control verification Misuse/abuse testing	Environment Management Configuration hardening Patch & update
	Education & Guidance Training & awareness Organization & culture	Secure Architecture Architecture design Technology management	Defect Management Defect tracking Metrics & feedback	Security Testing Scalable baseline Deep understanding	Operational Management Data protection Legacy management
	Stream A Stream B	Stream A Stream B	Stream A Stream B	Stream A Stream B	Stream A Stream B



SECURITY TESTING

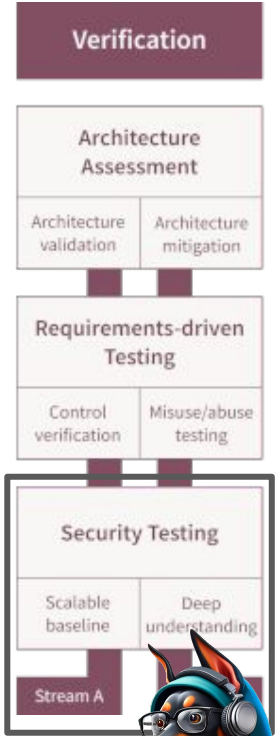
Model | Verification | Security Testing

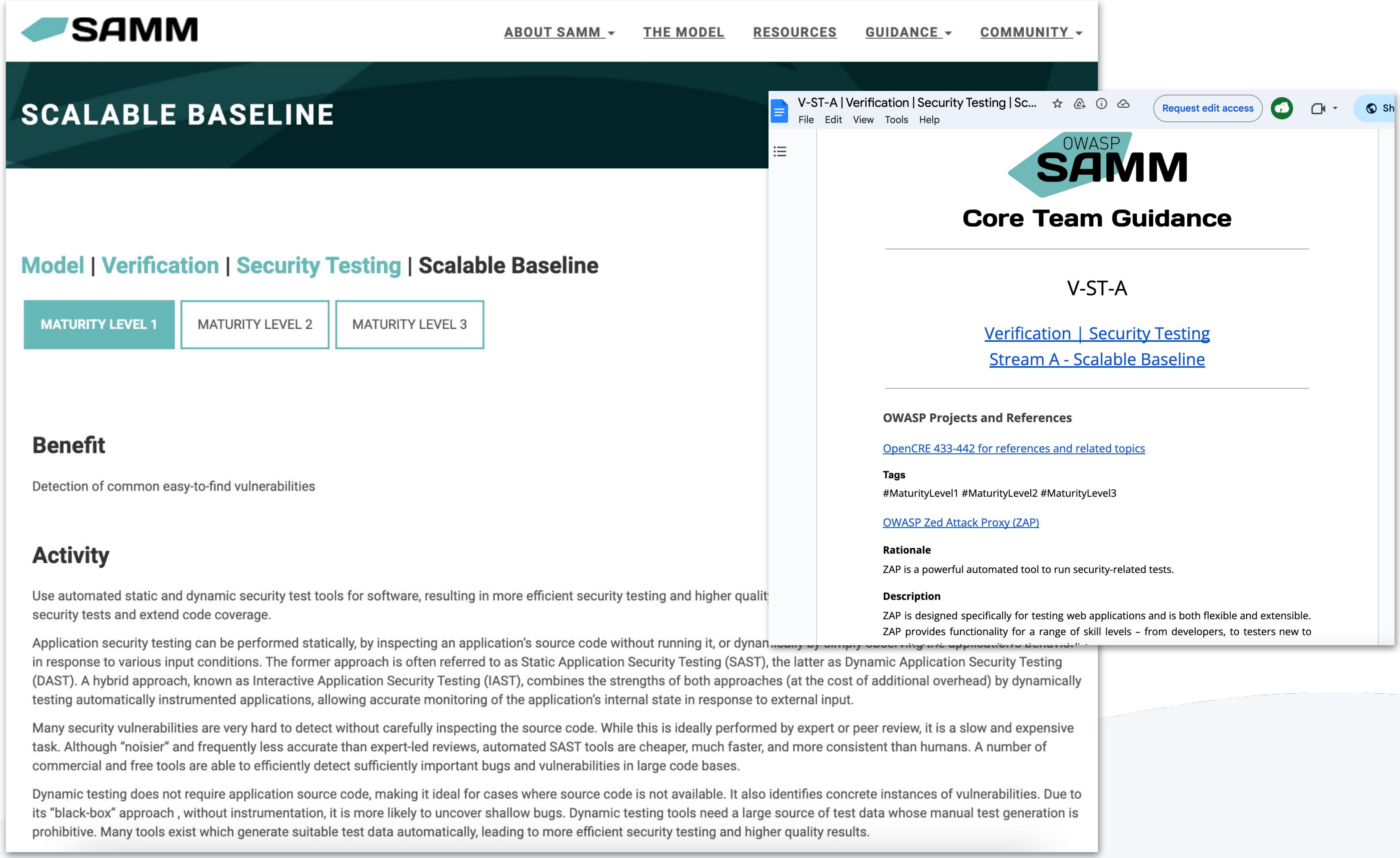
The Security Testing (ST) practice leverages the fact that, while automated security testing is fast and scales well to numerous applications, in-depth testing based on good knowledge of an application and its business logic is often only possible via slower, manual expert security testing. Each stream therefore has one approach at its core.

The first stream focuses on establishing a common security baseline to automatically detect so-called “low hanging fruit”. Progressively customize the automated tests for each application and increase their frequency of execution to detect more bugs and regressions earlier, as close as possible to their inception. The more bugs the automated processes can detect, the more time experts have to use their knowledge and creativity to focus on more complex attack vectors and ensure in-depth application testing in the second stream. As manual review is slow and hard to scale, reviewers prioritize testing components based on their risk, recent relevant changes, or upcoming major releases. Organizations can also access external expertise by participating in bug bounty programs, for example.

Unlike the Requirements-driven testing practice which focuses on verifying that applications correctly implement their requirements, the goal of this practice is to uncover technical and business-logic weaknesses in application and make them visible to management and business stakeholders, irrespective of requirements.

Maturity level		Stream A Scalable Baseline	Stream B Deep Understanding
1	Perform security testing (both manual and tool based) to discover security defects.	Utilize automated security testing tools.	Perform manual security testing of high-risk components.
2	Make security testing during development more complete and efficient through automation complemented with regular manual security penetration tests.	Employ application-specific security testing automation.	Conduct manual penetration testing.
3	Embed security testing as part of the development and deployment processes.	Integrate automated security testing into the build and deploy process.	Integrate security testing into development process.





SCALABLE BASELINE

Model | Verification | Security Testing | Scalable Baseline

Benefit

Detection of common easy-to-find vulnerabilities

Activity

Use automated static and dynamic security test tools for software, resulting in more efficient security testing and higher qualit
security tests and extend code coverage.

Application security testing can be performed statically, by inspecting an application's source code without running it, or dynan
in response to various input conditions. The former approach is often referred to as Static Application Security Testing (SAST), the latter as Dynamic Application Security Testing (DAST). A hybrid approach, known as Interactive Application Security Testing (IAST), combines the strengths of both approaches (at the cost of additional overhead) by dynamically testing automatically instrumented applications, allowing accurate monitoring of the application's internal state in response to external input.

Many security vulnerabilities are very hard to detect without carefully inspecting the source code. While this is ideally performed by expert or peer review, it is a slow and expensive task. Although "noisier" and frequently less accurate than expert-led reviews, automated SAST tools are cheaper, much faster, and more consistent than humans. A number of commercial and free tools are able to efficiently detect sufficiently important bugs and vulnerabilities in large code bases.

Dynamic testing does not require application source code, making it ideal for cases where source code is not available. It also identifies concrete instances of vulnerabilities. Due to its "black-box" approach , without instrumentation, it is more likely to uncover shallow bugs. Dynamic testing tools need a large source of test data whose manual test generation is prohibitive. Many tools exist which generate suitable test data automatically, leading to more efficient security testing and higher quality results.

Core Team Guidance

V-ST-A

Verification | Security Testing
Stream A - Scalable Baseline

OWASP Projects and References

OpenCRE 433-442 for references and related topics

Tags

#MaturityLevel1 #MaturityLevel2 #MaturityLevel3

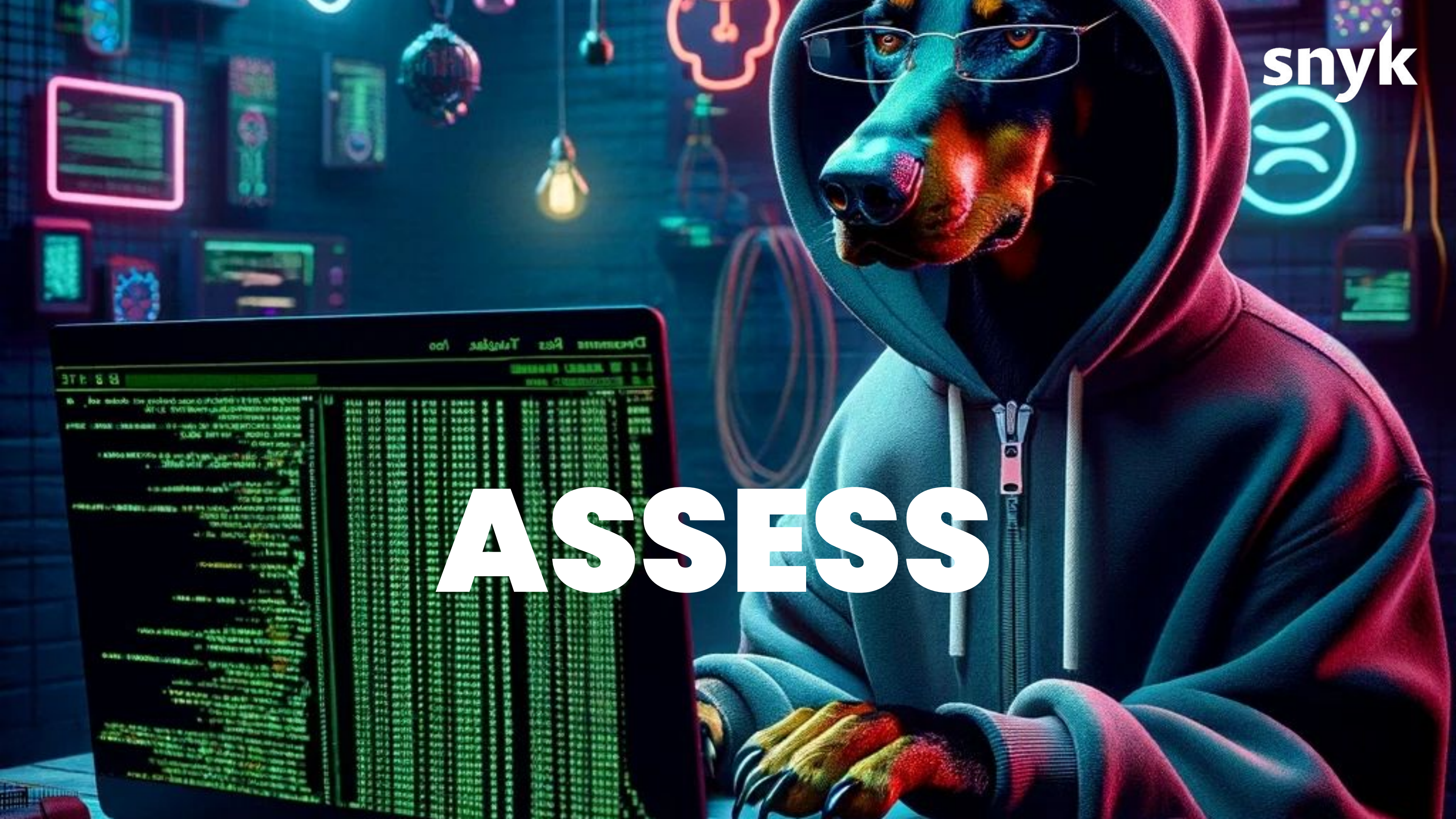
OWASP Zed Attack Proxy (ZAP)

Rationale

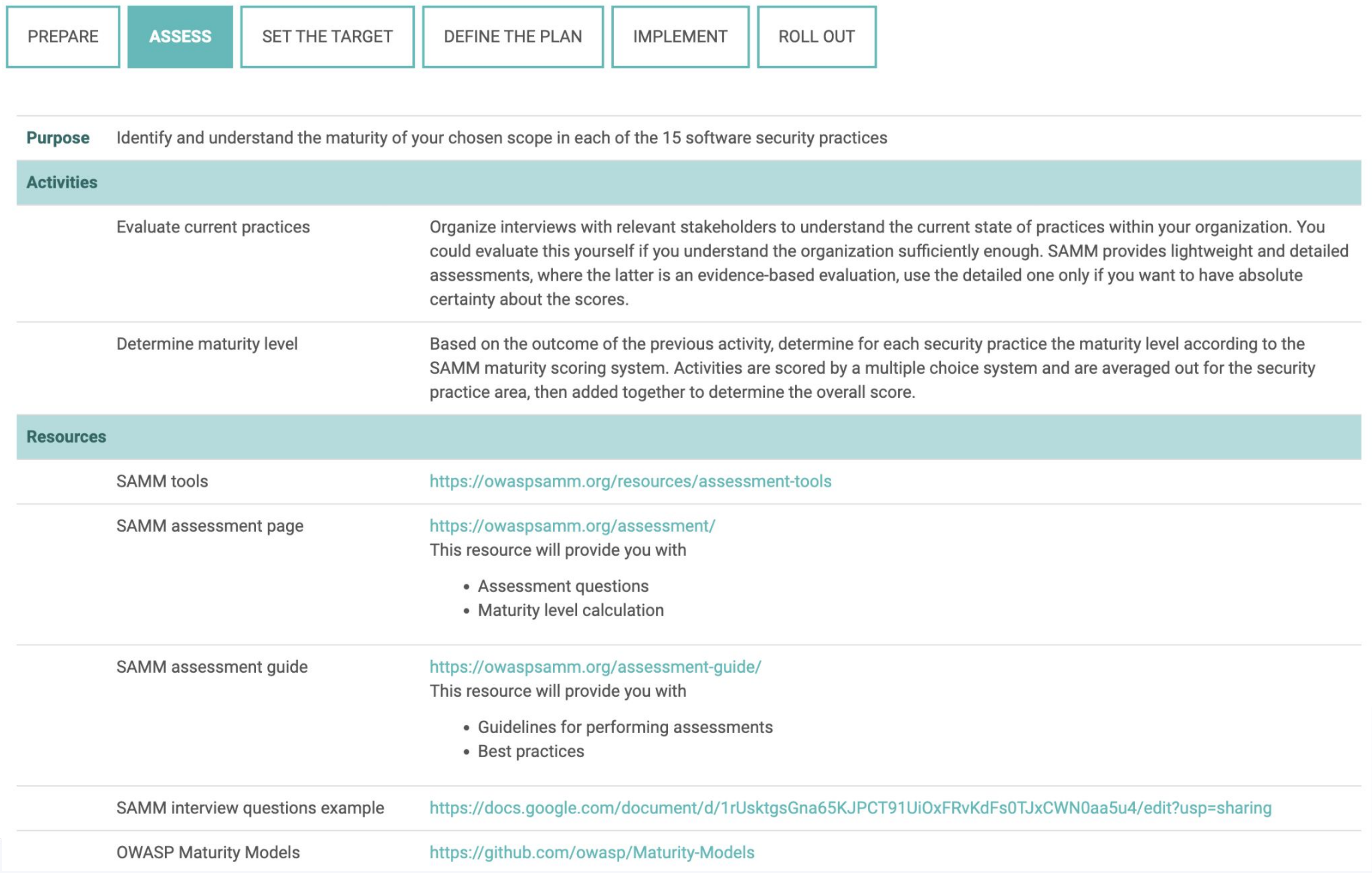
ZAP is a powerful automated tool to run security-related tests.

Description

ZAP is designed specifically for testing web applications and is both flexible and extensible. ZAP provides functionality for a range of skill levels – from developers, to testers new to

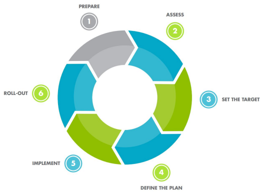


ASSESS



PREPARE | ASSESS | SET THE TARGET | DEFINE THE PLAN | IMPLEMENT | ROLL OUT

Purpose	Identify and understand the maturity of your chosen scope in each of the 15 software security practices	
Activities		
	Evaluate current practices	Organize interviews with relevant stakeholders to understand the current state of practices within your organization. You could evaluate this yourself if you understand the organization sufficiently enough. SAMM provides lightweight and detailed assessments, where the latter is an evidence-based evaluation, use the detailed one only if you want to have absolute certainty about the scores.
	Determine maturity level	Based on the outcome of the previous activity, determine for each security practice the maturity level according to the SAMM maturity scoring system. Activities are scored by a multiple choice system and are averaged out for the security practice area, then added together to determine the overall score.
Resources		
	SAMM tools	https://owaspsamm.org/resources/assessment-tools
	SAMM assessment page	https://owaspsamm.org/assessment/ This resource will provide you with • Assessment questions • Maturity level calculation
	SAMM assessment guide	https://owaspsamm.org/assessment-guide/ This resource will provide you with • Guidelines for performing assessments • Best practices
	SAMM interview questions example	https://docs.google.com/document/d/1rUsktgsGna65KJPCT91UiOxFRvKdFs0TJxCWN0aa5u4/edit?usp=sharing
	OWASP Maturity Models	https://github.com/owasp/Maturity-Models

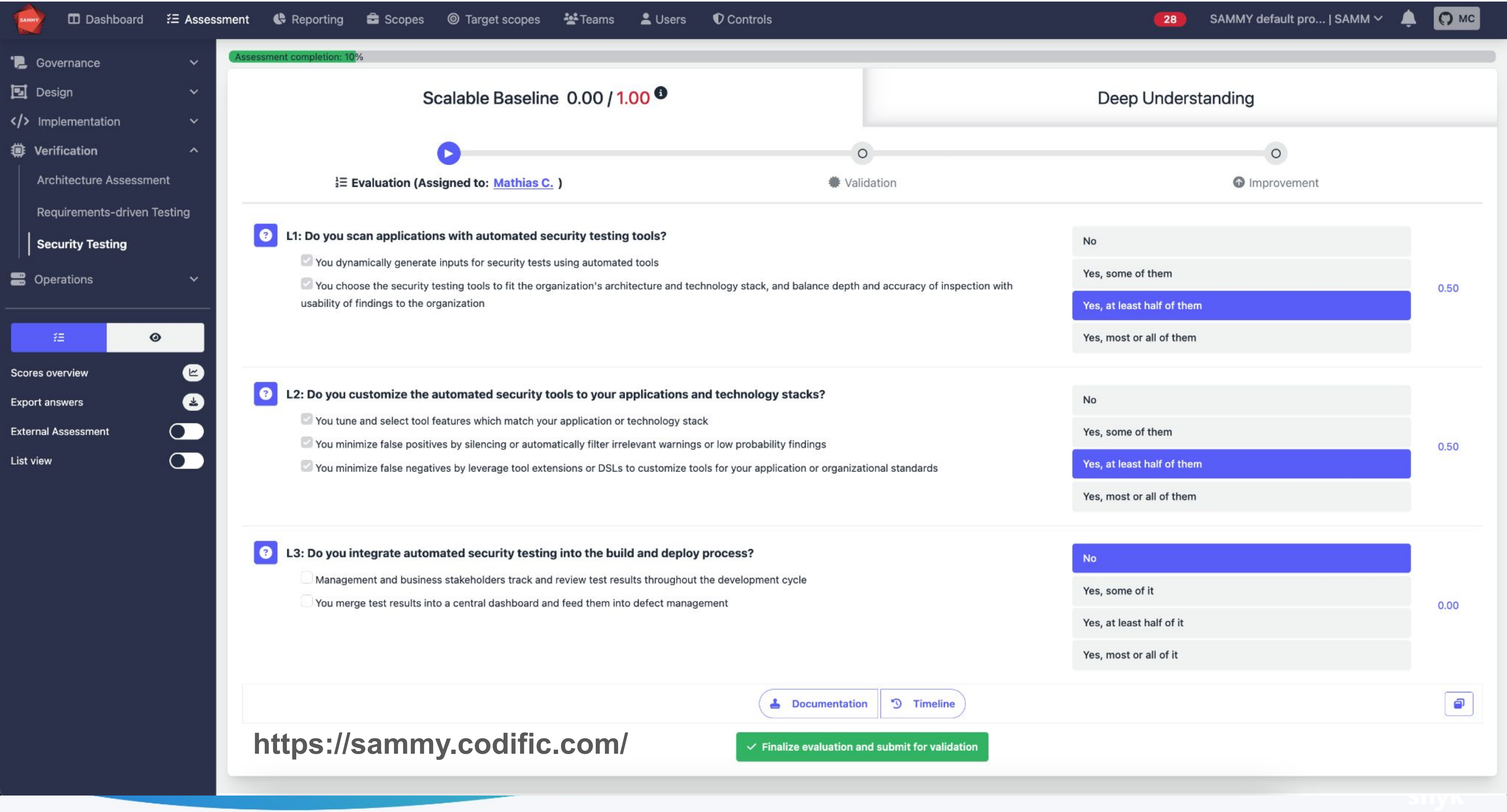


FAQ: Internal or External Assessments?

- Self-Assessment
- Internal Assessors but from different team, i.e. Compliance
- Internal Assessor with External Guidance (SAMM Expert)
- External Assessor

FAQ: Assessment Style?

- Open Ended Questions
- Survey-Style



Scalable Baseline 0.00 / 1.00

Deep Understanding

Evaluation (Assigned to: Mathias C.) — Validation — Improvement

L1: Do you scan applications with automated security testing tools?

- ☑ You dynamically generate inputs for security tests using automated tools
- ☑ You choose the security testing tools to fit the organization's architecture and technology stack, and balance depth and accuracy of inspection with usability of findings to the organization

No | Yes, some of them | [✓ Yes, at least half of them] | Yes, most or all of them — 0.50

L2: Do you customize the automated security tools to your applications and technology stacks?

- ☑ You tune and select tool features which match your application or technology stack
- ☑ You minimize false positives by silencing or automatically filter irrelevant warnings or low probability findings
- ☑ You minimize false negatives by leverage tool extensions or DSLs to customize tools for your application or organizational standards

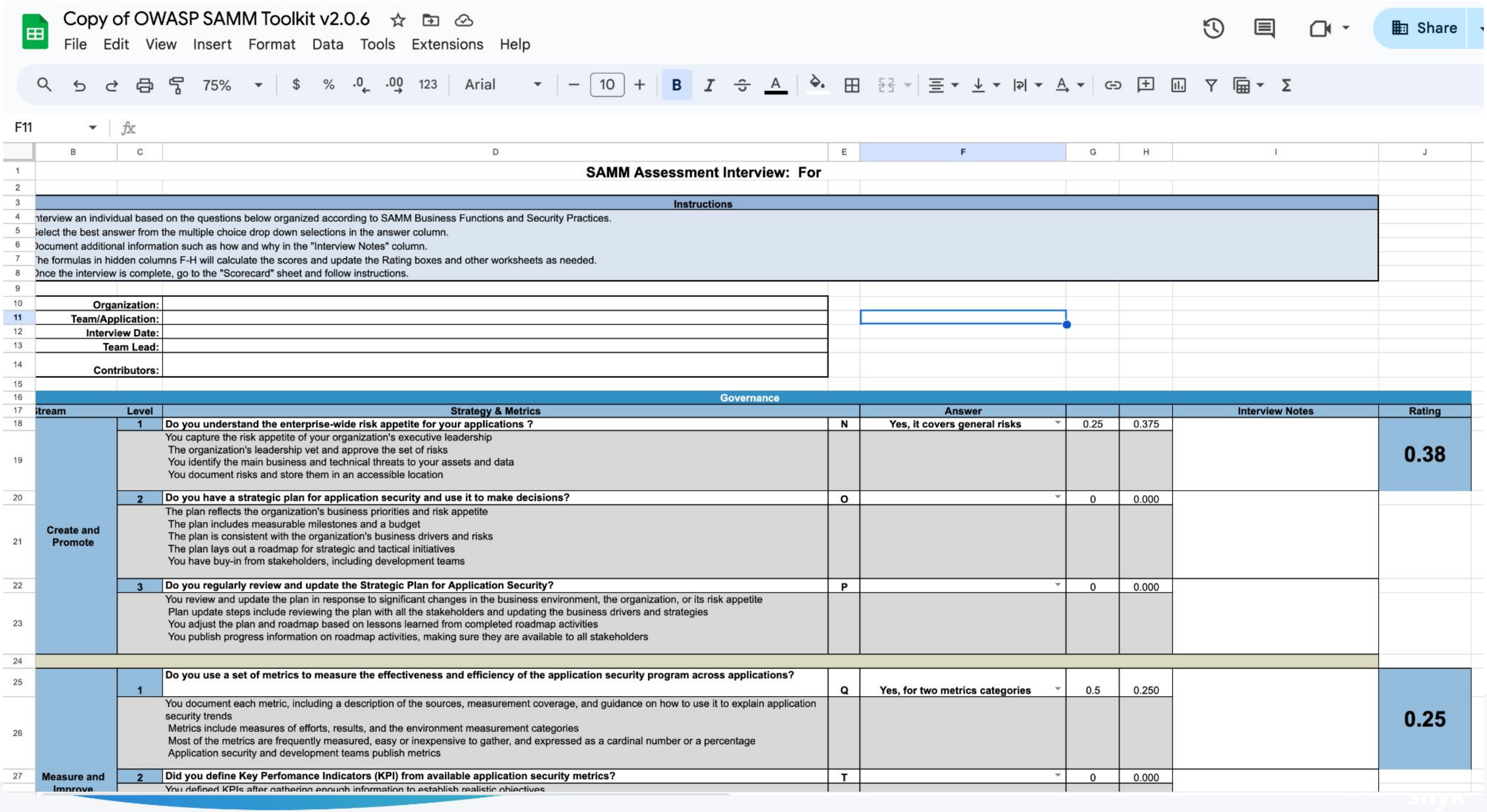
No | Yes, some of them | [✓ Yes, at least half of them] | Yes, most or all of them — 0.50

L3: Do you integrate automated security testing into the build and deploy process?

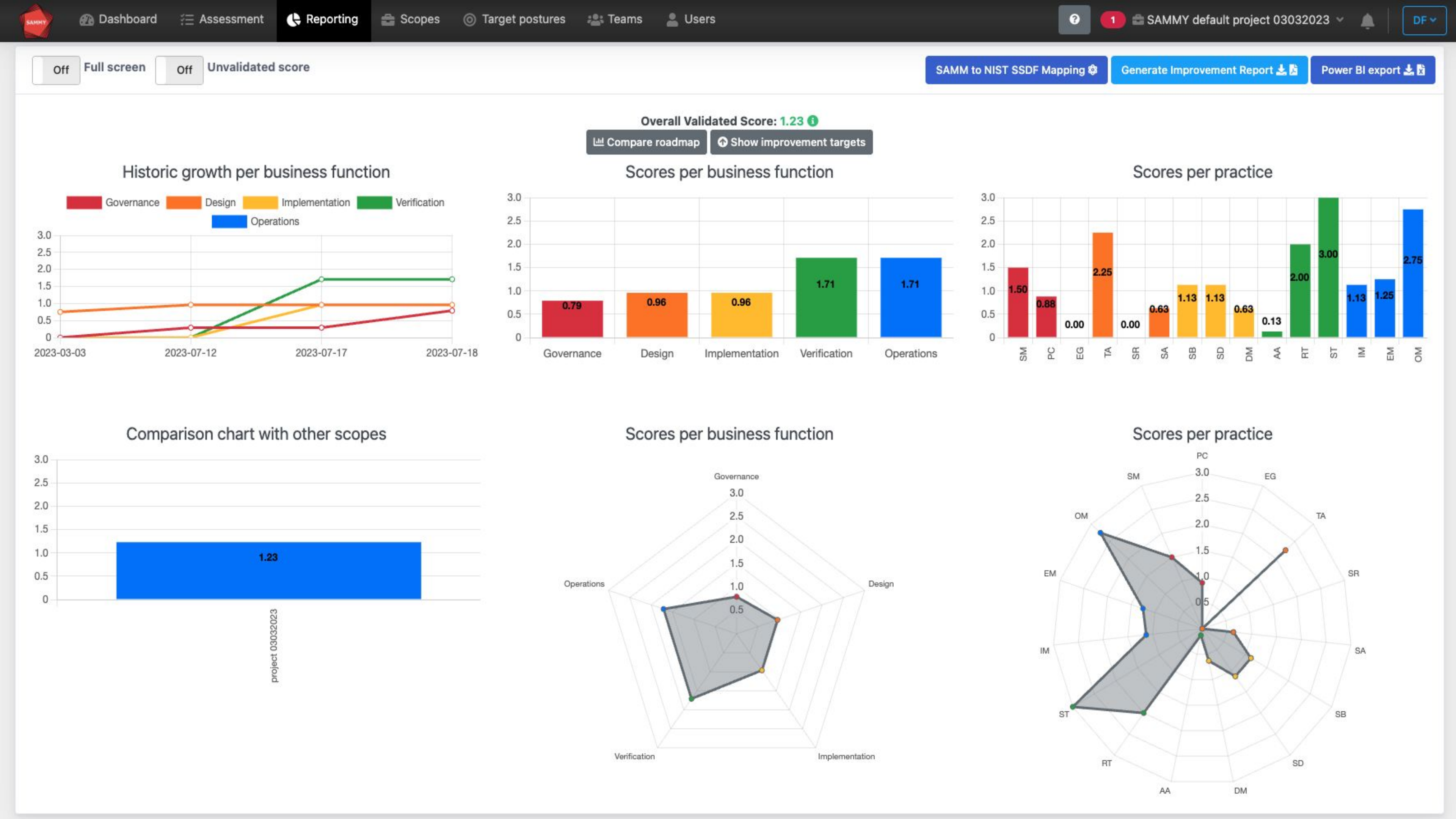
- □ Management and business stakeholders track and review test results throughout the development cycle
- □ You merge test results into a central dashboard and feed them into defect management

[✓ No] | Yes, some of it | Yes, at least half of it | Yes, most or all of it — 0.00

https://sammy.codific.com/



	B	C	D	E	F	G	H	I	J
1	SAMM Assessment Interview: For								
2									
3	Instructions								
4	nterview an individual based on the questions below organized according to SAMM Business Functions and Security Practices.								
5	elect the best answer from the multiple choice drop down selections in the answer column.								
6	ocument additional information such as how and why in the "Interview Notes" column.								
7	he formulas in hidden columns F-H will calculate the scores and update the Rating boxes and other worksheets as needed.								
8	nce the interview is complete, go to the "Scorecard" sheet and follow instructions.								
9									
10		Organization:							
11		Team/Application:							
12		Interview Date:							
13		Team Lead:							
14		Contributors:							
15									
16	Governance								
17	tream	Level	Strategy & Metrics		Answer			Interview Notes	Rating
18	Create and Promote	1	Do you understand the enterprise-wide risk appetite for your applications ?	N	Yes, it covers general risks	0.25	0.375		0.38
19			You capture the risk appetite of your organization's executive leadership The organization's leadership vet and approve the set of risks You identify the main business and technical threats to your assets and data You document risks and store them in an accessible location						
20		2	Do you have a strategic plan for application security and use it to make decisions?	O		0	0.000		
21			The plan reflects the organization's business priorities and risk appetite The plan includes measurable milestones and a budget The plan is consistent with the organization's business drivers and risks The plan lays out a roadmap for strategic and tactical initiatives You have buy-in from stakeholders, including development teams						
22		3	Do you regularly review and update the Strategic Plan for Application Security?	P		0	0.000		
23			You review and update the plan in response to significant changes in the business environment, the organization, or its risk appetite Plan update steps include reviewing the plan with all the stakeholders and updating the business drivers and strategies You adjust the plan and roadmap based on lessons learned from completed roadmap activities You publish progress information on roadmap activities, making sure they are available to all stakeholders						
24									
25	Measure and Improve	1	Do you use a set of metrics to measure the effectiveness and efficiency of the application security program across applications?	Q	Yes, for two metrics categories	0.5	0.250		0.25
26			You document each metric, including a description of the sources, measurement coverage, and guidance on how to use it to explain application security trends Metrics include measures of efforts, results, and the environment measurement categories Most of the metrics are frequently measured, easy or inexpensive to gather, and expressed as a cardinal number or a percentage Application security and development teams publish metrics						
27		2	Did you define Key Perfomance Indicators (KPI) from available application security metrics?	T		0	0.000		
			You defined KPIs after gathering enough information to establish realistic objectives						



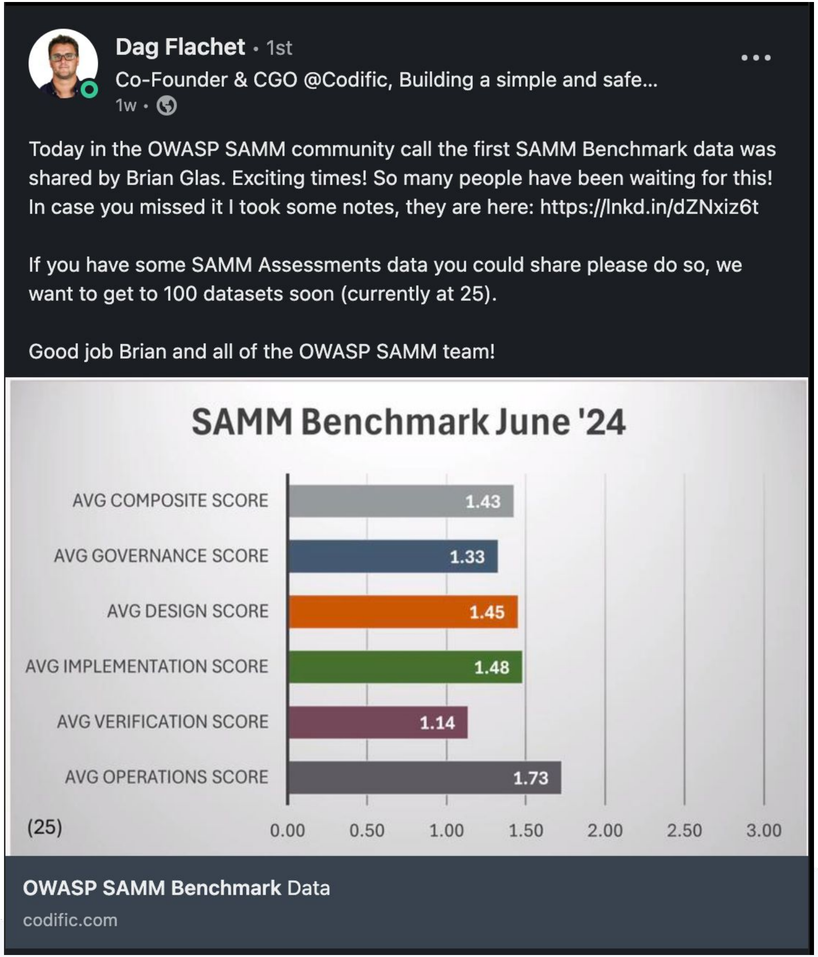
Off Full screen

Off Unvalidated score

Overall Validated Score: 1.23

SAMM Benchmark

Open for data donations!



Dag Flachet • 1st
Co-Founder & CGO @Codific, Building a simple and safe...
1w •

Today in the OWASP SAMM community call the first SAMM Benchmark data was shared by Brian Glas. Exciting times! So many people have been waiting for this! In case you missed it I took some notes, they are here: https://lnkd.in/dZNxiz6t

If you have some SAMM Assessments data you could share please do so, we want to get to 100 datasets soon (currently at 25).

Good job Brian and all of the OWASP SAMM team!

SAMM Benchmark June '24

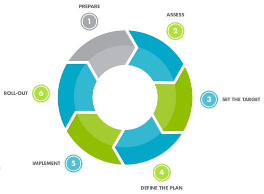
AVG COMPOSITE SCORE	1.43
AVG GOVERNANCE SCORE	1.33
AVG DESIGN SCORE	1.45
AVG IMPLEMENTATION SCORE	1.48
AVG VERIFICATION SCORE	1.14
AVG OPERATIONS SCORE	1.73

OWASP SAMM Benchmark Data
codific.com

snyk

SET THE TARGET

PREPARE | ASSESS | **SET THE TARGET** | DEFINE THE PLAN | IMPLEMENT | ROLL OUT



Purpose	Develop a target score that you can use as a measuring stick to guide you to act on the most important activities for your situation	
Activities		
	Define the target	Set or update the target by identifying which activities your organization should implement ideally. Typically, this will include more lower-level than higher-level activities. Ensure that the total set of selected activities makes sense, and take into account dependencies between activities.
	Estimate overall impact	Estimate the impact of the chosen target on the organization. Try to express in budgetary arguments.
Resources		
	SAMM roadmap chart	Worksheet (part of the SAMM Benchmark as a comparative source)
		Leverage the Roadmap worksheet in the SAMM Toolbox to help calculate maturity score improvements based on future answers
Best practices		
	Take into account the organization’s risk profile	
	Respect dependencies between activities	
	As a rough measure, the overall impact of a software assurance effort is estimated at 5 % to 10% of the total development cost	

Recommendations from the Field

I would focus on making sure the team has **basic security training**.
I would also focus on getting to maturity **level 2 for Secure Build** and **Secure Deploy** activities.
From there onwards I'd look into **security requirements** (ASVS) and requiring mandatory unit/integration tests for at least some of them.
Tooling could also be interesting to look at or at least experiment with and to see which tools and tool categories you could add to your development processes.

Aram Hovsepyan, via [OWASP Slack channel]()

Recommendations from the Field

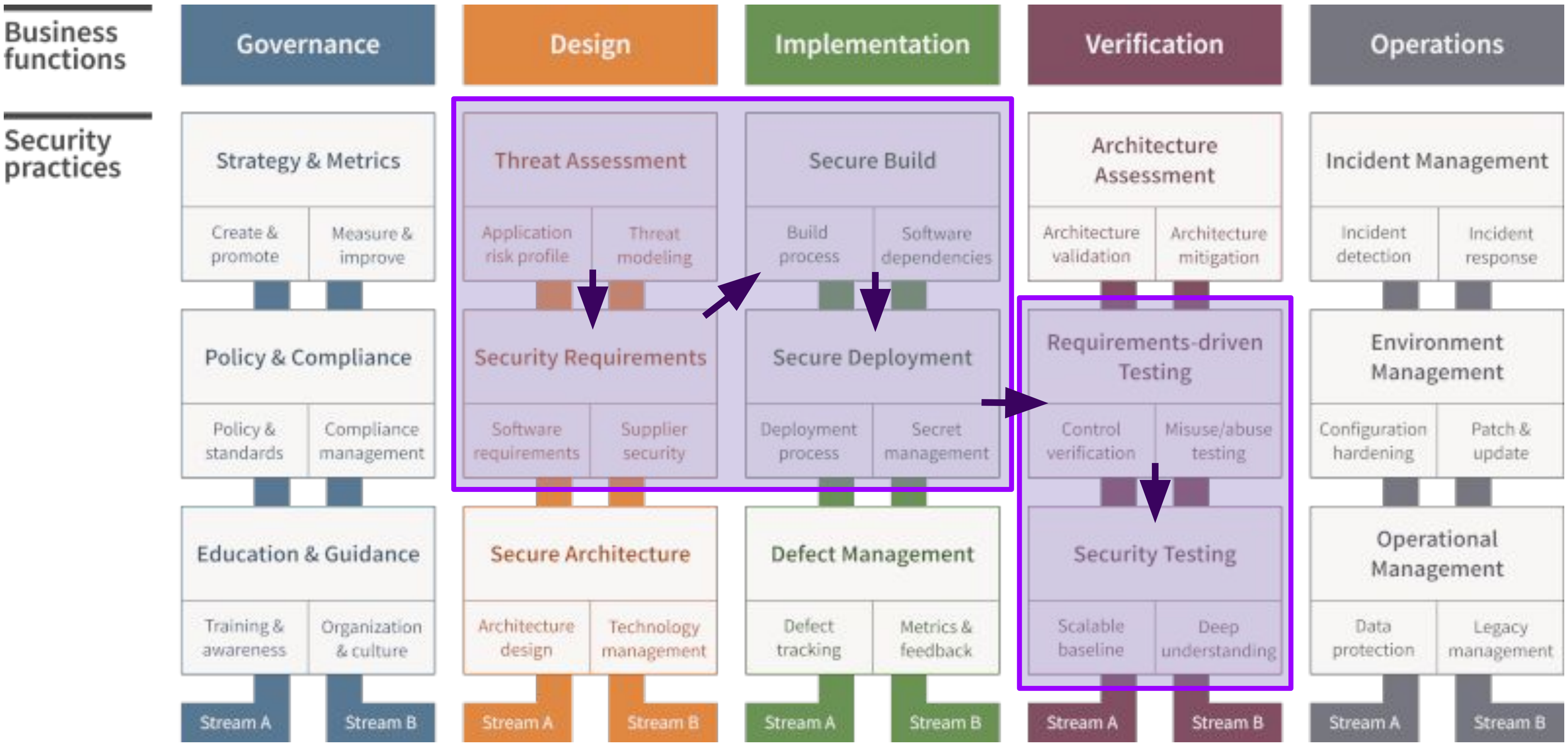
Identify your **target maturity**, based on

- your current capabilities
- teams involved
- tech-stack
- suppliers, etc.
- organization risk profile and appetite

Be realistic in your first target maturity and **time** horizon, based on available **budget** and willingness for **change**. There is no standard solution for this, as this is unique for every organization and its risk / compliance environment.

Seba, via OWASP Slack channel

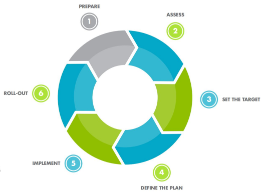
Typical Developers’ Touchpoints / “Critical Path”



Business functions	Governance	Design	Implementation	Verification	Operations
Security practices	Strategy & Metrics	Threat Assessment	Secure Build	Architecture Assessment	Incident Management
	Create & promote Measure & improve	Application risk profile Threat modeling	Build process Software dependencies	Architecture validation Architecture mitigation	Incident detection Incident response
	Policy & Compliance	Security Requirements	Secure Deployment	Requirements-driven Testing	Environment Management
	Policy & standards Compliance management	Software requirements Supplier security	Deployment process Secret management	Control verification Misuse/abuse testing	Configuration hardening Patch & update
	Education & Guidance	Secure Architecture	Defect Management	Security Testing	Operational Management
	Training & awareness Organization & culture	Architecture design Technology management	Defect tracking Metrics & feedback	Scalable baseline Deep understanding	Data protection Legacy management
	Stream A Stream B	Stream A Stream B	Stream A Stream B	Stream A Stream B	Stream A Stream B

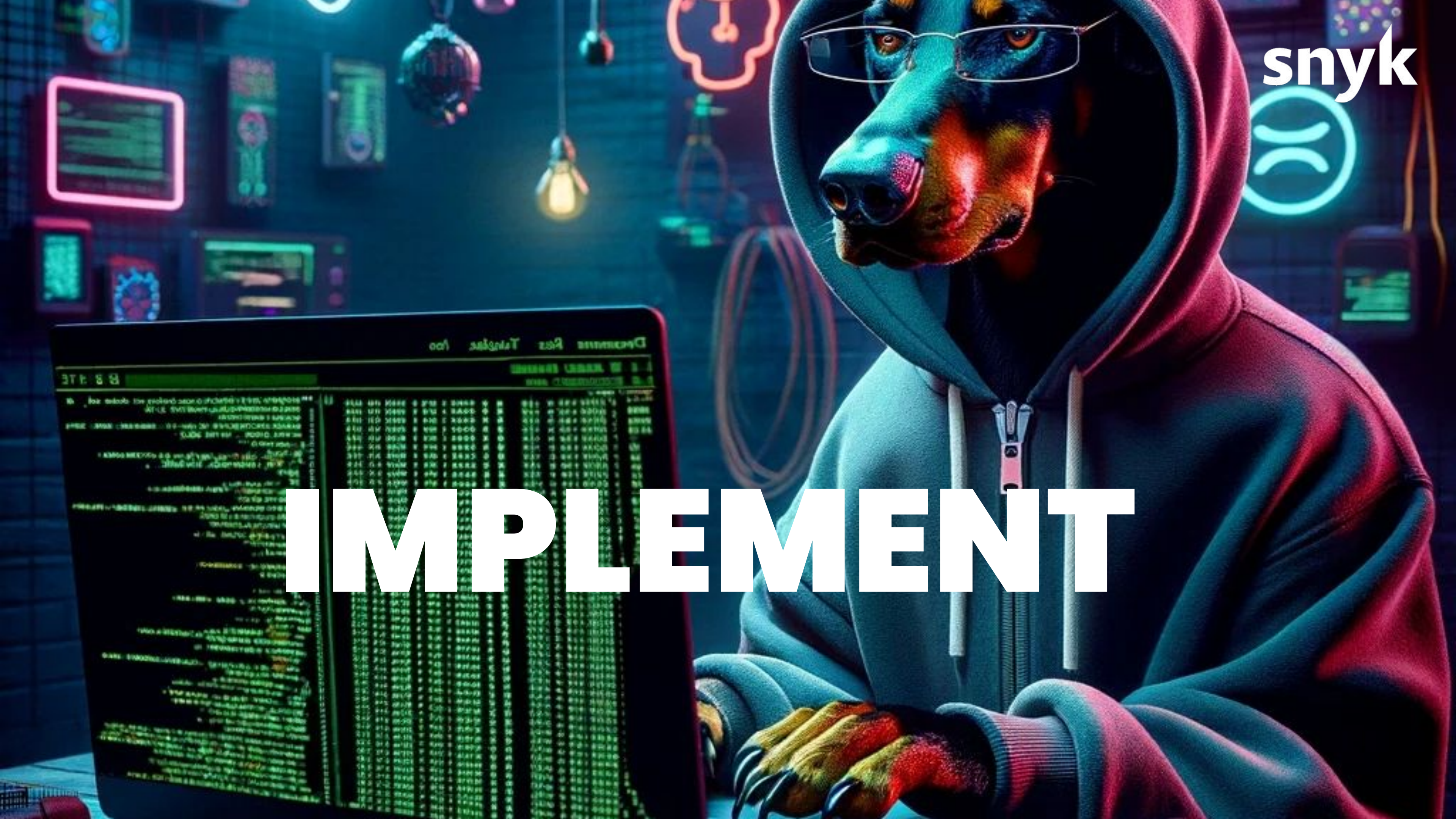
DEFINE THE PLAN

PREPARE | ASSESS | SET THE TARGET | **DEFINE THE PLAN** | IMPLEMENT | ROLL OUT

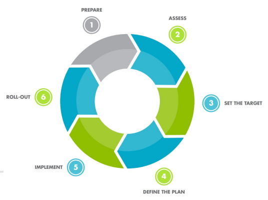


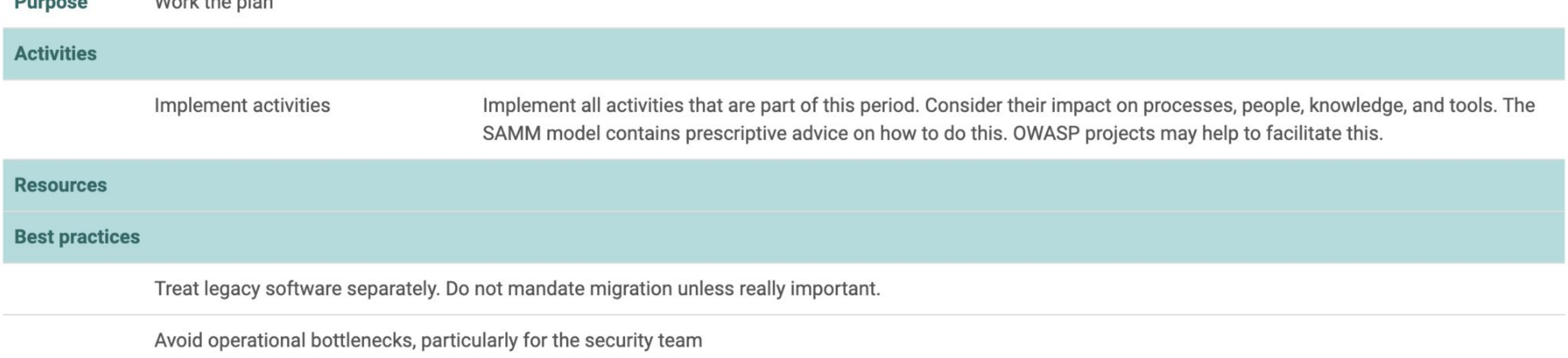
Purpose	Develop or update your plan to take your organization to the next level	
Activities		
	Determine change schedule	Choose a realistic change strategy in terms of number and duration of phases. A typical roadmap consists of 4 to 6 phases for 3 to 12 months.
	Develop/update the roadmap plan	Distribute the implementation of additional activities over the different roadmap phases, taking into account the effort required to implement them. Try to balance the implementation effort over the different periods, and take dependencies between activities into account.
Resources		
Best practices		
	Identify activities that can be completed quickly and successfully early in the project	
	Start with awareness / training	
	Adapt to coming release cycles / key projects	

IMPLEMENT



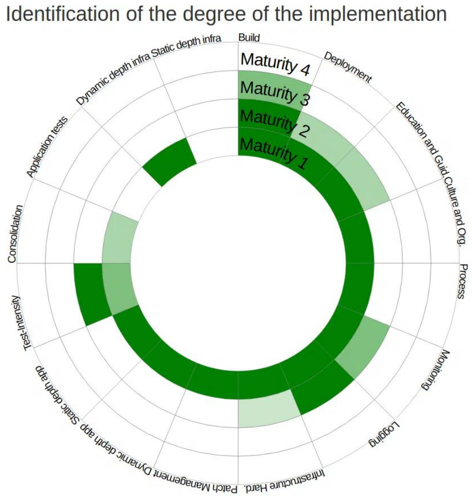
PREPARE | ASSESS | SET THE TARGET | DEFINE THE PLAN | **IMPLEMENT** | ROLL OUT





Purpose	Work the plan	
Activities		
	Implement activities	Implement all activities that are part of this period. Consider their impact on processes, people, knowledge, and tools. The SAMM model contains prescriptive advice on how to do this. OWASP projects may help to facilitate this.
Resources		
Best practices		
	Treat legacy software separately. Do not mandate migration unless really important.	
	Avoid operational bottlenecks, particularly for the security team	

FAQ: OWASP SAMM vs. OWASP DSOMM



DSOMM

The DSOMM framework (DevSecOps Maturity Model) consists of four levels of DevSecOps maturity. Each level represents a distinct stage in the evolution of security integration, ranging from basic awareness and ad-hoc practices to advanced, fully integrated, and automated security processes. **These levels provide a roadmap for organizations to systematically enhance their security posture within the DevOps framework.**

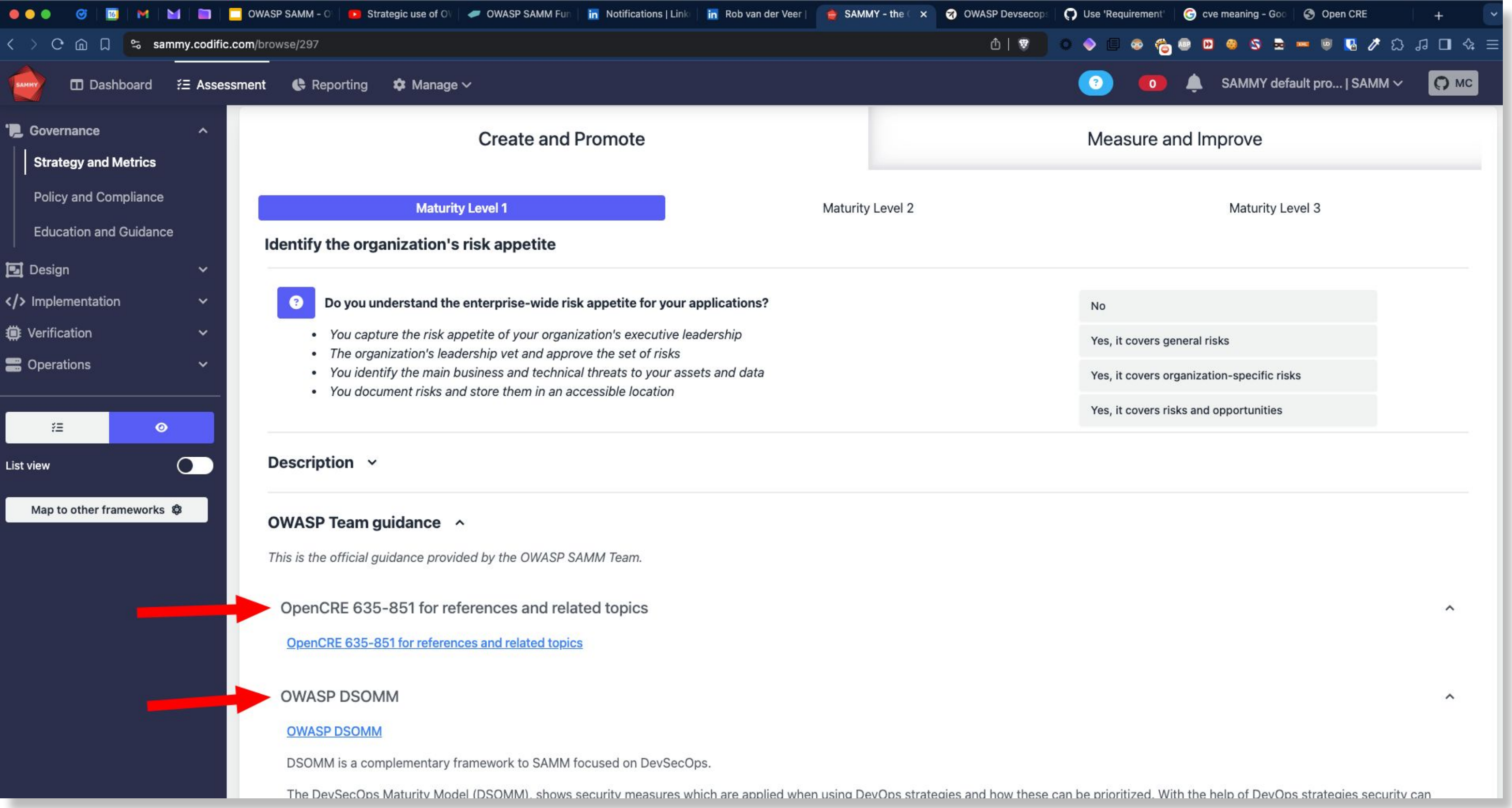
Sample Target Groups

SAMM	DSOMM
“Standard”, OWASP Flagship Project	“Emerging”, OWASP Lab Project
High Level Overview	Low Level Overview
Management Topics like Compliance & Governance	Only DevSecOps Topics
Planning of High Level Targets	Planning of Concrete Targets
Works “Out of the Box”	Needs Customization

Recommendation from Timo Pagel

1. Assess and plan security strategy with SAMM
2. Adapt DSOMM

Source: https://www.youtube.com/watch?v=MIzENOyylZI



Create and Promote | Measure and Improve

Maturity Level 1 | Maturity Level 2 | Maturity Level 3

Identify the organization's risk appetite

Do you understand the enterprise-wide risk appetite for your applications?

- *You capture the risk appetite of your organization's executive leadership*
- *The organization's leadership vet and approve the set of risks*
- *You identify the main business and technical threats to your assets and data*
- *You document risks and store them in an accessible location*

○ No
○ Yes, it covers general risks
○ Yes, it covers organization-specific risks
○ Yes, it covers risks and opportunities

Description

OWASP Team guidance

This is the official guidance provided by the OWASP SAMM Team.

OpenCRE 635-851 for references and related topics

[OpenCRE 635-851 for references and related topics]

OWASP DSOMM

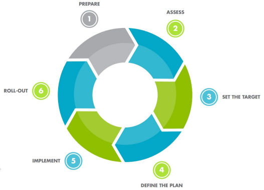
[OWASP DSOMM]

DSOMM is a complementary framework to SAMM focused on DevSecOps.

The DevSecOps Maturity Model (DSOMM), shows security measures which are applied when using DevOps strategies and how these can be prioritized. With the help of DevOps strategies security can

ROLLOUT

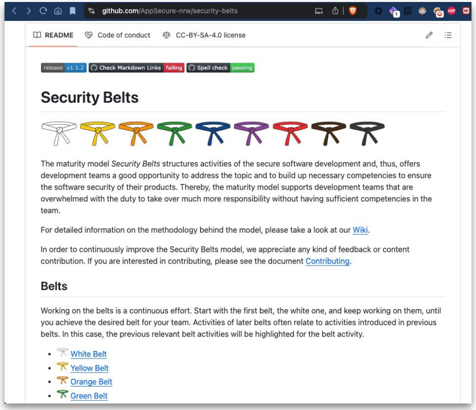
PREPARE | ASSESS | SET THE TARGET | DEFINE THE PLAN | IMPLEMENT | **ROLL OUT**



Purpose	Ensure that improvements are available and effectively used within the organization	
Activities		
	Evangelize improvements	Make the steps and improvements visible for everyone involved by organizing trainings and communicating with management stakeholders.
	Measure effectiveness	Measure the adoption and effectiveness of implemented improvements by analyzing usage and impact.
Resources		
Best practices		
	Categorize applications according to their impact on the organization. Focus on high-impact applications.	
	Use team champions to spread new activities throughout the organization.	

Tipp: Security Champions Program

1. Gamification, Competition & Incentivation
2. CTF / Capture the Flag events
3. "Security Belts" approach



Security Belts

The maturity model *Security Belts* structures activities of the secure software development and, thus, offers development teams a good opportunity to address the topic and to build up necessary competencies to ensure the software security of their products. Thereby, the maturity model supports development teams that are overwhelmed with the duty to take over much more responsibility without having sufficient competencies in the team.

For detailed information on the methodology behind the model, please take a look at our Wiki.

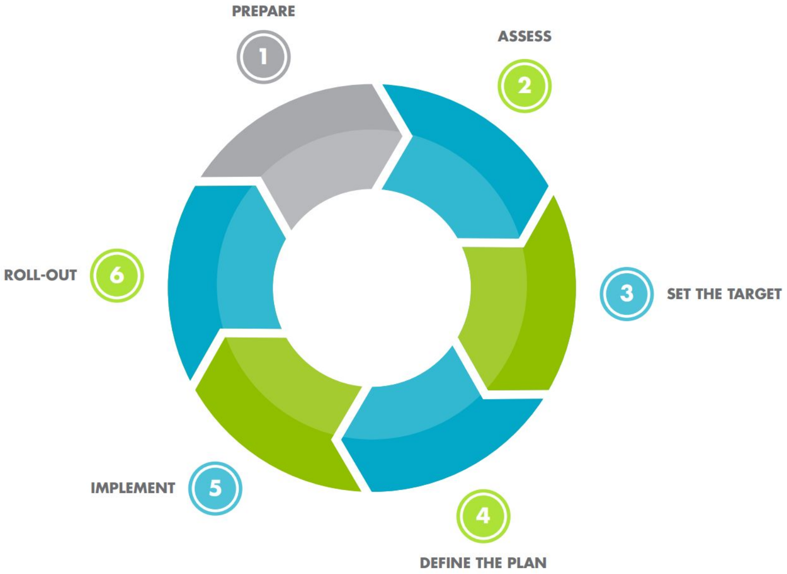
In order to continuously improve the Security Belts model, we appreciate any kind of feedback or content contribution. If you are interested in contributing, please see the document Contributing.

Belts

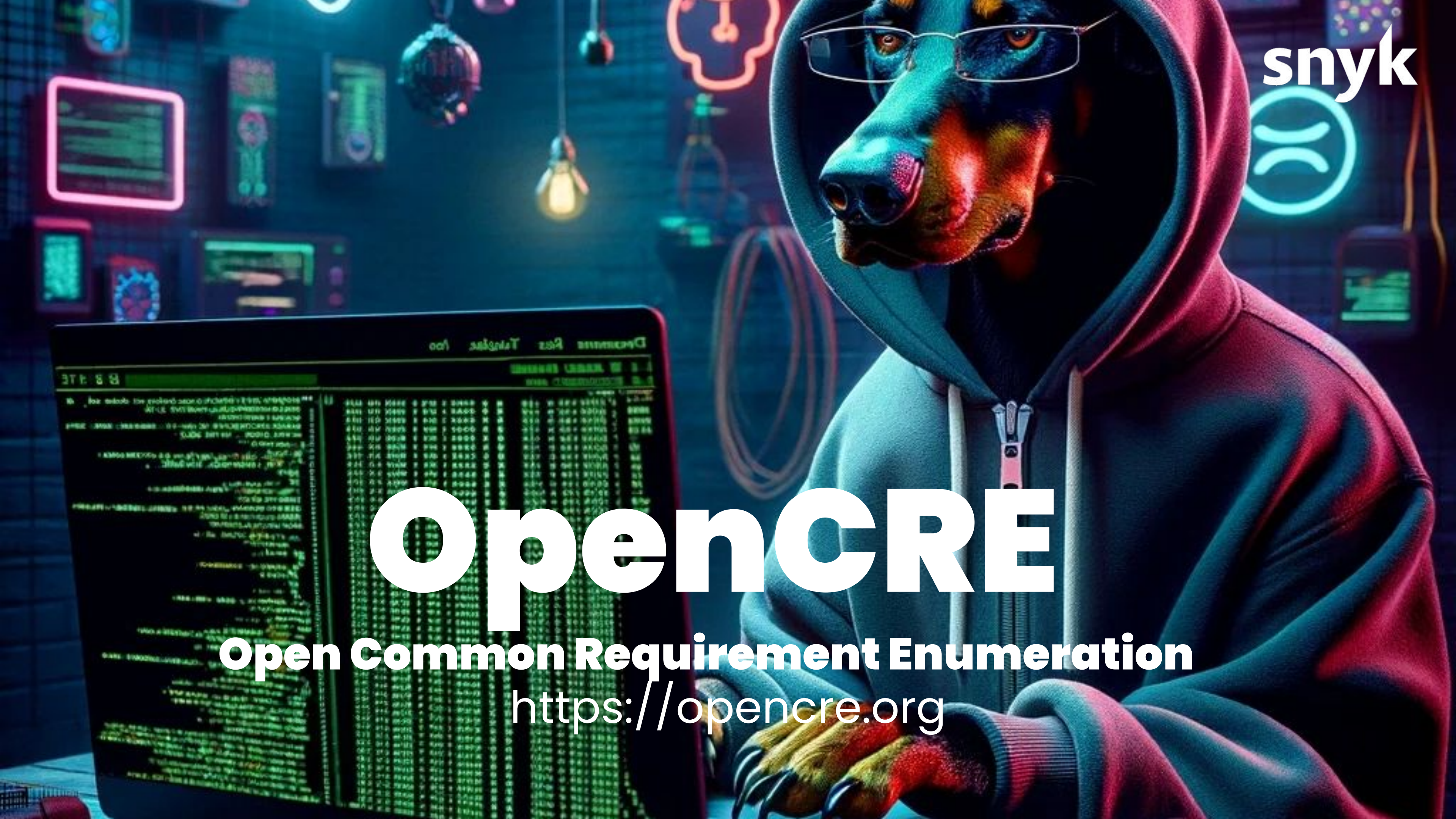
Working on the belts is a continuous effort. Start with the first belt, the white one, and keep working on them, until you achieve the desired belt for your team. Activities of later belts often relate to activities introduced in previous belts. In this case, the previous relevant belt activities will be highlighted for the belt activity.

- White Belt
- Yellow Belt
- Orange Belt
- Green Belt

OWASP SAMM Project Cycle



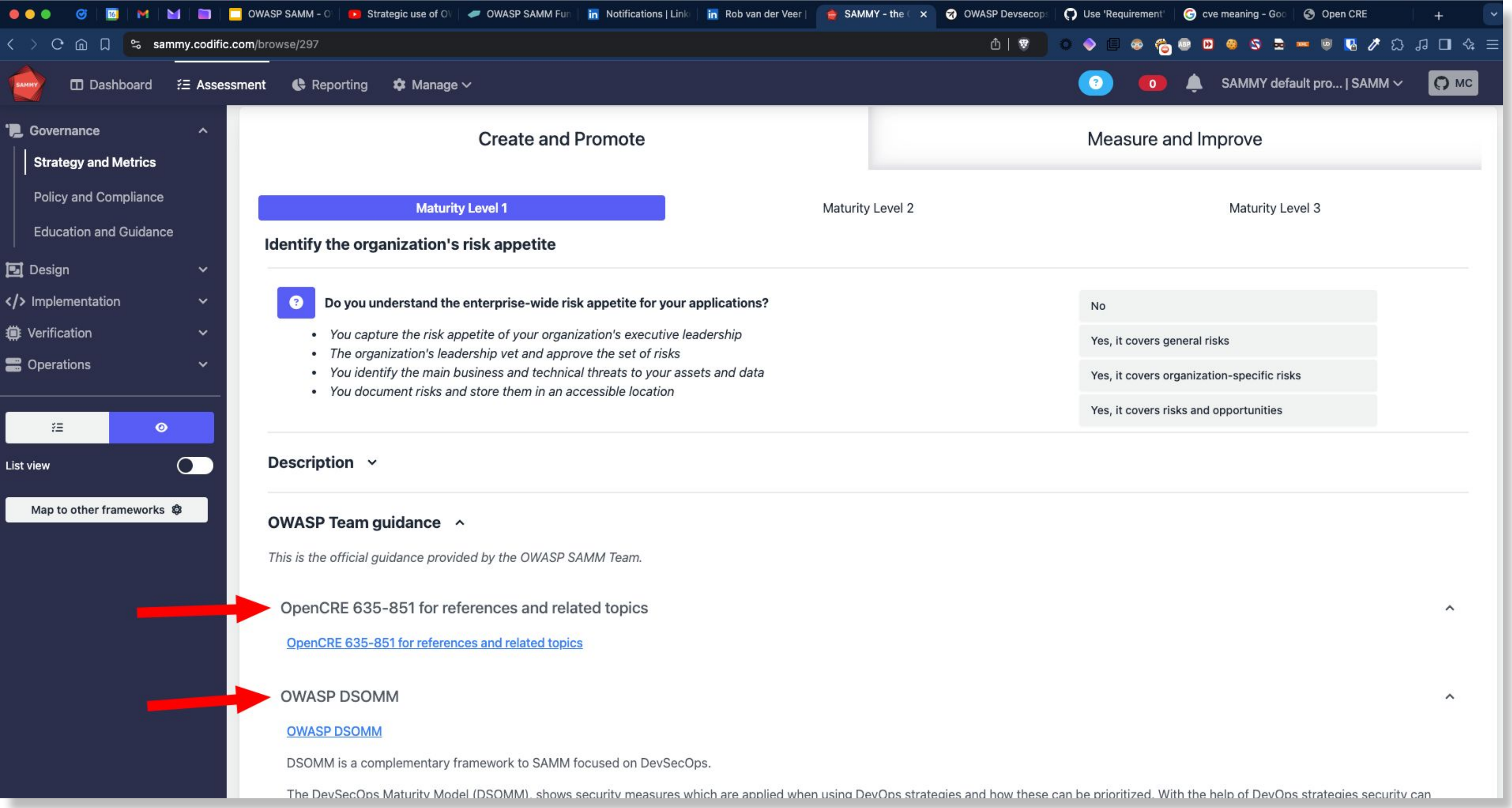
PREPARE | ASSESS | SET THE TARGET | DEFINE THE PLAN | IMPLEMENT | ROLL OUT



OpenCRE

Open Common Requirement Enumeration

https://opencre.org



Create and Promote

Measure and Improve

Maturity Level 1 | Maturity Level 2 | Maturity Level 3

Identify the organization's risk appetite

Do you understand the enterprise-wide risk appetite for your applications?

- *You capture the risk appetite of your organization's executive leadership*
- *The organization's leadership vet and approve the set of risks*
- *You identify the main business and technical threats to your assets and data*
- *You document risks and store them in an accessible location*

No

Yes, it covers general risks

Yes, it covers organization-specific risks

Yes, it covers risks and opportunities

Description

OWASP Team guidance

This is the official guidance provided by the OWASP SAMM Team.

OpenCRE 635-851 for references and related topics

OpenCRE 635-851 for references and related topics

OWASP DSOMM

OWASP DSOMM

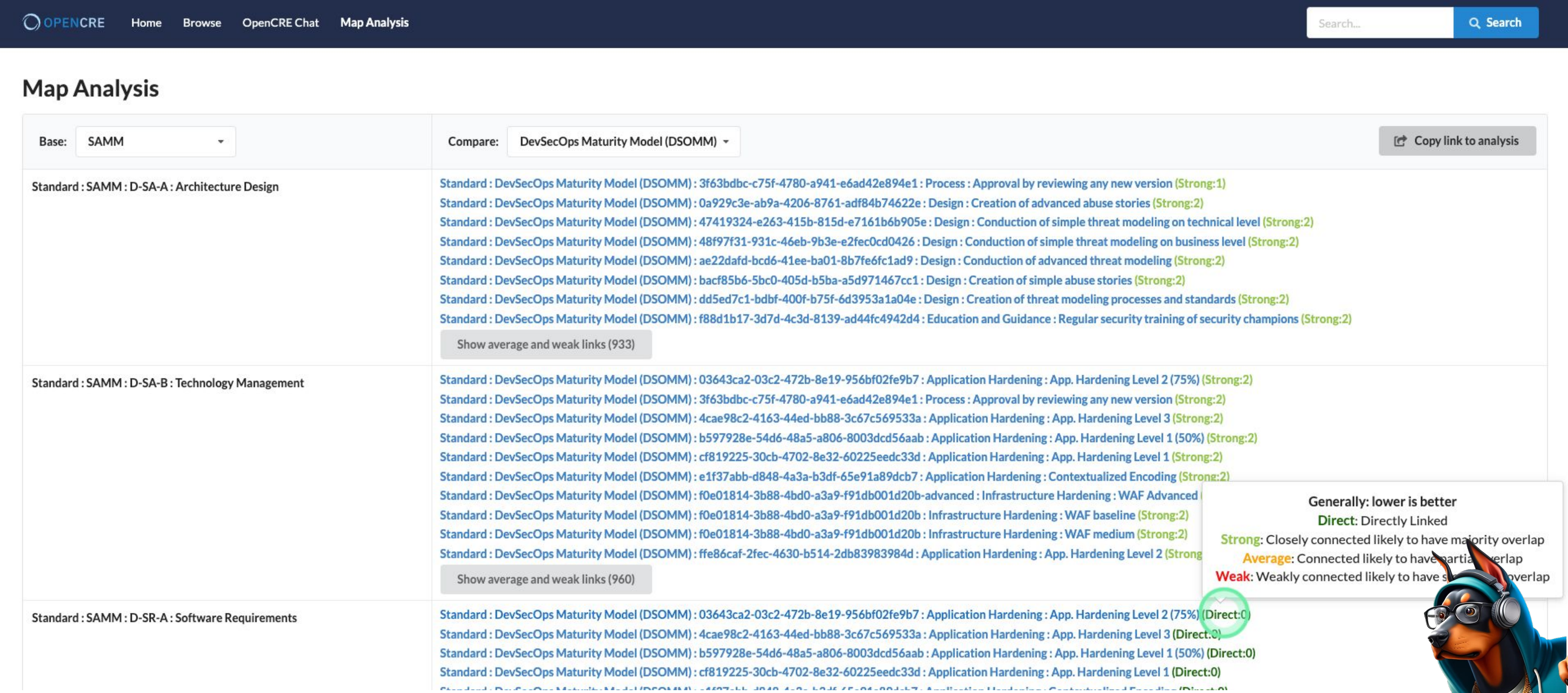
DSOMM is a complementary framework to SAMM focused on DevSecOps.

The DevSecOps Maturity Model (DSOMM), shows security measures which are applied when using DevOps strategies and how these can be prioritized. With the help of DevOps strategies security can

Open Common Requirement Enumeration

The Open Source project “OpenCRE “ **links all security standards and guidelines together** at the level of requirements into one harmonized resource: threats, weaknesses, what to verify, how to program, how to test, which tool settings, in-depth discussion, training material. Everything organized.
https://opencre.org

Naming is probably derived from CWE and CVE, common in the industry.
CVE = Common Vulnerabilities and Exposures
CWE = Common Weakness Enumeration



Map Analysis

Base: SAMM

Compare: DevSecOps Maturity Model (DSOMM)

Standard : SAMM : D-SA-A : Architecture Design

Standard : DevSecOps Maturity Model (DSOMM) : 3f63bdbc-c75f-4780-a941-e6ad42e894e1 : Process : Approval by reviewing any new version (Strong:1)
Standard : DevSecOps Maturity Model (DSOMM) : 0a929c3e-ab9a-4206-8761-adf84b74622e : Design : Creation of advanced abuse stories (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : 47419324-e263-415b-815d-e7161b6b905e : Design : Conduction of simple threat modeling on technical level (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : 48f97f31-931c-46eb-9b3e-e2fec0cd0426 : Design : Conduction of simple threat modeling on business level (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : ae22dafd-bcd6-41ee-ba01-8b7fe6fc1ad9 : Design : Conduction of advanced threat modeling (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : bacf85b6-5bc0-405d-b5ba-a5d971467cc1 : Design : Creation of simple abuse stories (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : dd5ed7c1-bdbf-400f-b75f-6d3953a1a04e : Design : Creation of threat modeling processes and standards (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : f88d1b17-3d7d-4c3d-8139-ad44fc4942d4 : Education and Guidance : Regular security training of security champions (Strong:2)

Standard : SAMM : D-SA-B : Technology Management

Standard : DevSecOps Maturity Model (DSOMM) : 03643ca2-03c2-472b-8e19-956bf02fe9b7 : Application Hardening : App. Hardening Level 2 (75%) (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : 3f63bdbc-c75f-4780-a941-e6ad42e894e1 : Process : Approval by reviewing any new version (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : 4cae98c2-4163-44ed-bb88-3c67c569533a : Application Hardening : App. Hardening Level 3 (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : b597928e-54d6-48a5-a806-8003dcd56aab : Application Hardening : App. Hardening Level 1 (50%) (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : cf819225-30cb-4702-8e32-60225eedc33d : Application Hardening : App. Hardening Level 1 (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : e1f37abb-d848-4a3a-b3df-65e91a89dcb7 : Application Hardening : Contextualized Encoding (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : f0e01814-3b88-4bd0-a3a9-f91db001d20b-advanced : Infrastructure Hardening : WAF Advanced
Standard : DevSecOps Maturity Model (DSOMM) : f0e01814-3b88-4bd0-a3a9-f91db001d20b : Infrastructure Hardening : WAF baseline (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : f0e01814-3b88-4bd0-a3a9-f91db001d20b : Infrastructure Hardening : WAF medium (Strong:2)
Standard : DevSecOps Maturity Model (DSOMM) : ffe86caf-2fec-4630-b514-2db83983984d : Application Hardening : App. Hardening Level 2 (Strong

Generally: lower is better
Direct: Directly Linked
Strong: Closely connected likely to have majority overlap
Average: Connected likely to have partia erlap
Weak: Weakly connected likely to have s overlap

Standard : SAMM : D-SR-A : Software Requirements

Standard : DevSecOps Maturity Model (DSOMM) : 03643ca2-03c2-472b-8e19-956bf02fe9b7 : Application Hardening : App. Hardening Level 2 (75%) (Direct:0)
Standard : DevSecOps Maturity Model (DSOMM) : 4cae98c2-4163-44ed-bb88-3c67c569533a : Application Hardening : App. Hardening Level 3 (Direct:0)
Standard : DevSecOps Maturity Model (DSOMM) : b597928e-54d6-48a5-a806-8003dcd56aab : Application Hardening : App. Hardening Level 1 (50%) (Direct:0)
Standard : DevSecOps Maturity Model (DSOMM) : cf819225-30cb-4702-8e32-60225eedc33d : Application Hardening : App. Hardening Level 1 (Direct:0)

THANK YOU

@mathiasconradt