DSOMM from Theory to Enforcement

OWASP DevSecOps Maturity Model

Raz Probstein, Solution Engineer at Jit

August 3rd, 2023

I’m Raz Probstein

Solution Engineer at Jit
Continuous Security Platform for Developers

Developer, Tech, Security, BioTech Advocate

DSOMM from Theory to Enforcement

OWASP Devsecops Maturity Model

- Security Maturity Models - Overview and Background
- What is DSOMM?
- Operationalizing DSOMM
- Automated Continuous Security Approach
- Q/A

Security Maturity Models

Overview and Background

Leading Security Maturity Models

What images to **Security Maturity Models** bring to mind?

Security Maturity Models

Overview and Background

Leading Security Maturity Models

What images to **Security Maturity Models** bring to mind?

Security Maturity Models

Overview and Background

Leading Security Maturity Models

But Wait! There can be significant value when implemented well!

Security Maturity Models

Overview and Background

Leading Security Maturity Models

But Wait! There can be significant value when implemented well!

Translation to Leadership

Security Maturity Models

Overview and Background

Leading Security Maturity Models

But Wait! There can be significant value when implemented well!

Translation to Leadership

Current State Analysis

Security Maturity Models

Overview and Background

Leading Security Maturity Models

But Wait! There can be significant value when implemented well!

Translation to Leadership

Current State Analysis

Gap Prioritization

Security Maturity Models

Overview and Background

Leading Security Maturity Models

But Wait! There can be significant value when implemented well!

Translation to Leadership

Tools & Automation

Current State Analysis

Gap Prioritization

Security Maturity Models

Overview and Background

Leading Security Maturity Models

But Wait! There can be significant value when implemented well!

- Translation to Leadership
- Tools & Automation
- Current State Analysis
- Resource Needs
- Gap Prioritization

Security Maturity Models

Overview and Background

Leading Security Maturity Models

But Wait! There can be significant value when implemented well!

Translation to Leadership

Tools & Automation

Current State Analysis

Resource Needs

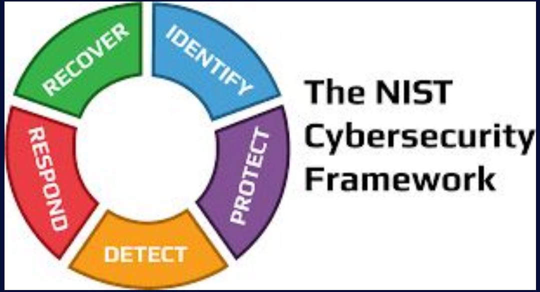
Gap Prioritization

Risk Exposure

Security Maturity Models

Overview and Background

Leading Security Maturity Models

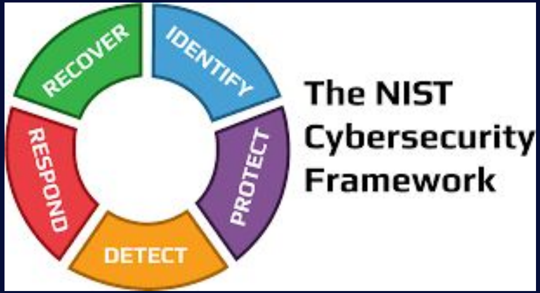


NIST

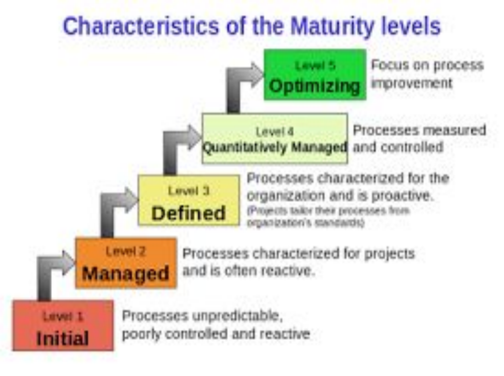
Security Maturity Models

Overview and Background

Leading Security Maturity Models



NIST



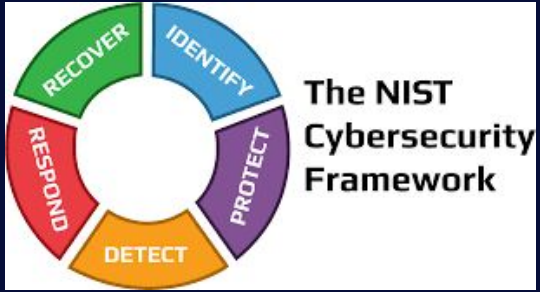
Characteristics of the Maturity levels
Level 5
Optimizing
Focus on process improvement
Level 4
Quantitatively Managed
Processes measured and controlled
Level 3
Defined
Processes characterized for the organization and is proactive.
(Projects tailor their processes from organization's standards)
Level 2
Managed
Processes characterized for projects and is often reactive.
Level 1
Initial
Processes unpredictable, poorly controlled and reactive

CMMI

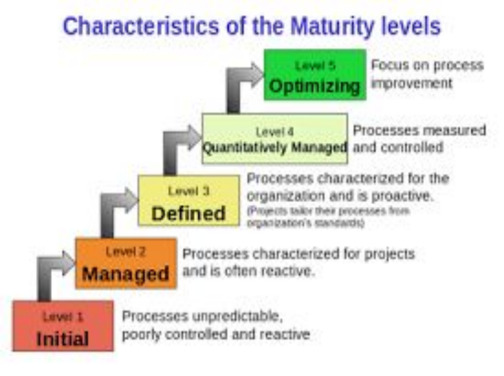
Security Maturity Models

Overview and Background

Leading Security Maturity Models



NIST



Characteristics of the Maturity levels

Level 5 **Optimizing**: Focus on process improvement

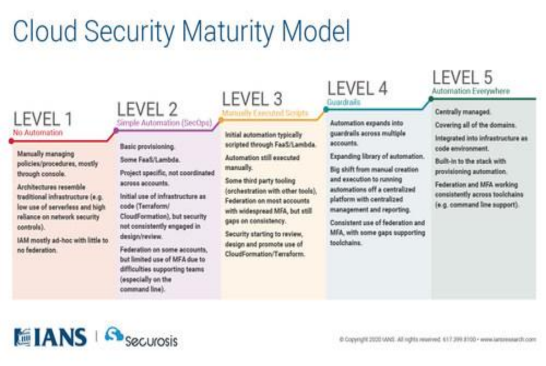
Level 4 **Quantitatively Managed**: Processes measured and controlled

Level 3 **Defined**: Processes characterized for the organization and is proactive. (Projects tailor their processes from organization's standards)

Level 2 **Managed**: Processes characterized for projects and is often reactive.

Level 1 **Initial**: Processes unpredictable, poorly controlled and reactive

CMMI



Cloud Security Maturity Model

LEVEL 1
No Automation

Manually managing policies/procedures, mostly through console.

Architectures resemble traditional infrastructure (e.g. low use of serverless and high reliance on network security controls).

IAM mostly ad-hoc with little to no federation.

LEVEL 2
Simple Automation (SecOps)

Basic provisioning.

Some FaaS/Lambda.

Project specific, not coordinated across accounts.

Initial use of infrastructure as code (Terraform/ CloudFormation), but security not consistently engaged in design/review.

Federation on some accounts, but limited use of MFA due to difficulties supporting teams (especially on the command line).

LEVEL 3
Manually Executed Scripts

Initial automation typically scripted through FaaS/Lambda.

Automation still executed manually.

Some third party tooling (orchestration with other tools). Federation on most accounts with widespread MFA, but still gaps on consistency.

Security starting to review, design and promote use of CloudFormation/Terraform.

LEVEL 4
Guardrails

Automation expands into guardrails across multiple accounts.

Expanding library of automation.

Big shift from manual creation and execution to running automations off a centralized platform with centralized management and reporting.

Consistent use of federation and MFA, with some gaps supporting toolchains.

LEVEL 5
Automation Everywhere

Centrally managed.

Covering all of the domains.

Integrated into infrastructure as code environment.

Built-in to the stack with provisioning automation.

Federation and MFA working consistently across toolchains (e.g. command line support).

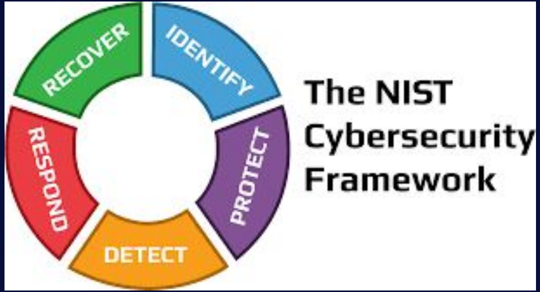
IANS | Securosis

CSA

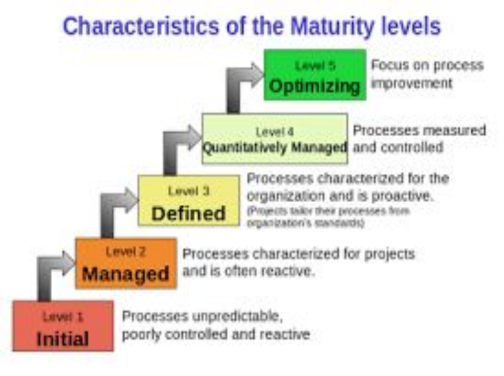
Security Maturity Models

Overview and Background

Leading Security Maturity Models



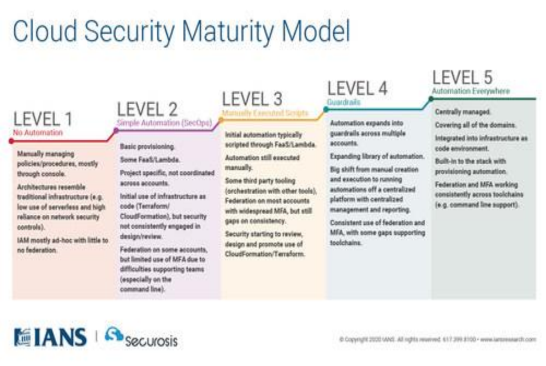
NIST



Characteristics of the Maturity levels

- Level 5 **Optimizing**: Focus on process improvement
- Level 4 **Quantitatively Managed**: Processes measured and controlled
- Level 3 **Defined**: Processes characterized for the organization and is proactive. (Projects tailor their processes from organization's standards)
- Level 2 **Managed**: Processes characterized for projects and is often reactive.
- Level 1 **Initial**: Processes unpredictable, poorly controlled and reactive

CMMI



Cloud Security Maturity Model

LEVEL 1 No Automation

Manually managing policies/procedures, mostly through console.

Architectures resemble traditional infrastructure (e.g. low use of serverless and high reliance on network security controls).

IAM mostly ad-hoc with little to no federation.

LEVEL 2 Simple Automation (SecOps)

Basic provisioning.

Some FaaS/Lambda.

Project specific, not coordinated across accounts.

Initial use of infrastructure as code (Terraform/ CloudFormation), but security not consistently engaged in design/review.

Federation on some accounts, but limited use of MFA due to difficulties supporting teams (especially on the command line).

LEVEL 3 Manually Executed Scripts

Initial automation typically scripted through FaaS/Lambda.

Automation still executed manually.

Some third party tooling (orchestration with other tools), Federation on most accounts with widespread MFA, but still gaps on consistency.

Security starting to review, design and promote use of CloudFormation/Terraform.

LEVEL 4 Guardrails

Automation expands into guardrails across multiple accounts.

Expanding library of automation.

Big shift from manual creation and execution to running automations off a centralized platform with centralized management and reporting.

Consistent use of federation and MFA, with some gaps supporting toolchains.

LEVEL 5 Automation Everywhere

Centrally managed.

Covering all of the domains.

Integrated into infrastructure as code environment.

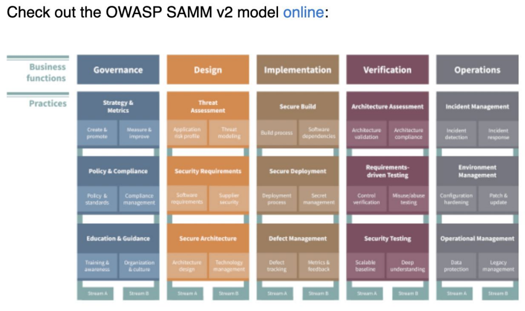
Built-in to the stack with provisioning automation.

Federation and MFA working consistently across toolchains (e.g. command line support).

IANS | Securosis

© Copyright 2020 IANS. All rights reserved. 617 399 8100 • www.iansresearch.com

CSA



Check out the OWASP SAMM v2 model online:

Business functions	Governance	Design	Implementation	Verification	Operations
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 compliance	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

OWASP SAMM

Security Maturity Models

Overview and Background

Leading Security Maturity Models

Why we like DSOMM at Jit?

Security Maturity Models

Overview and Background

Leading Security Maturity Models

Why we like DSOMM at Jit?

Open Source

We value open source tools and processes and Jit as a source for consistently maintained and independent resources

Security Maturity Models

Overview and Background

Leading Security Maturity Models

Why we like DSOMM at Jit?

Open Source

We value open source tools and processes and Jit as a source for consistently maintained and independent resources

Developer Owned

As an elite / high performing development team, we need a maturity framework that aligns with our organizational approach

Security Maturity Models

Overview and Background

Leading Security Maturity Models

Why we like DSOMM at Jit?

Open Source

We value open source tools and processes and Jit as a source for consistently maintained and independent resources

Developer Owned

As an elite / high performing development team, we need a maturity framework that aligns with our organizational approach

Automation & Tooling

DSOMM can be integrated with technology and tools to support automated assessment and tracking

Security Maturity Models

Overview and Background

Leading Security Maturity Models

Why we like DSOMM at Jit?

Open Source

We value open source tools and processes and Jit as a source for consistently maintained and independent resources

Developer Owned

As an elite / high performing development team, we need a maturity framework that aligns with our organizational approach

Automation & Tooling

DSOMM can be integrated with technology and tools to support automated assessment and tracking

Stakeholder Value

DSOMM provides value to company across leadership, team leads, developers and external stakeholders

What is DSOMM?

OWASP DevSecOps Maturity Model

Focused on Security & DevOps Strategies

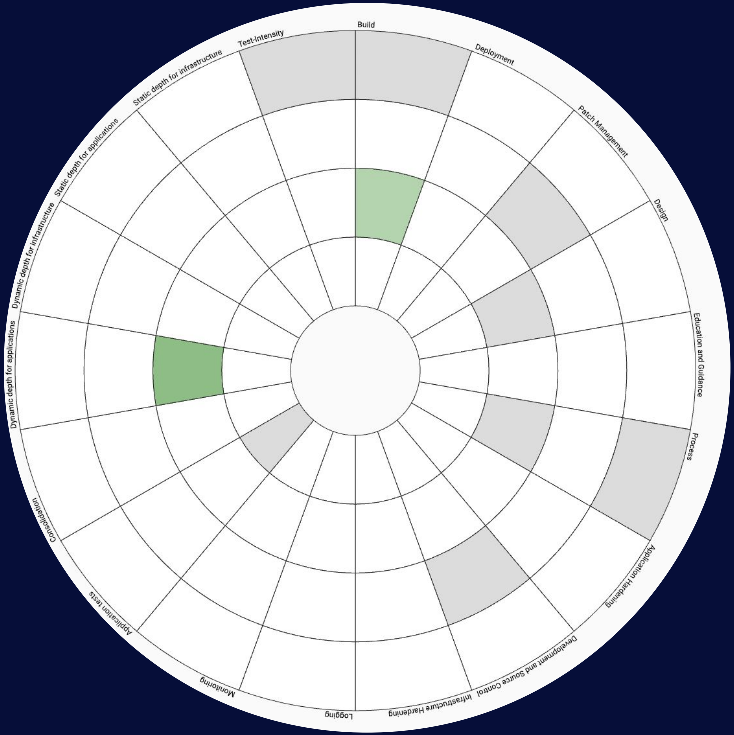
Based on need for Security Prioritization for Developers

Developed by Timo Pagel in 2017

What is DSOMM?

OWASP Devsecops Maturity Model

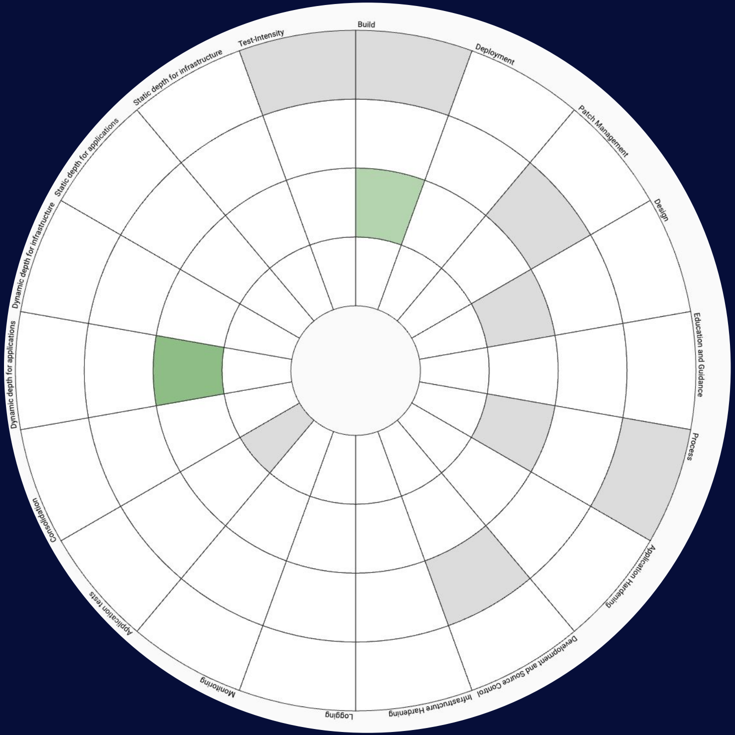
Defines Maturity Levels specific to Software Development



What is DSOMM?

OWASP Devsecops Maturity Model

Provides Security Dimensions and Levels specific to software development

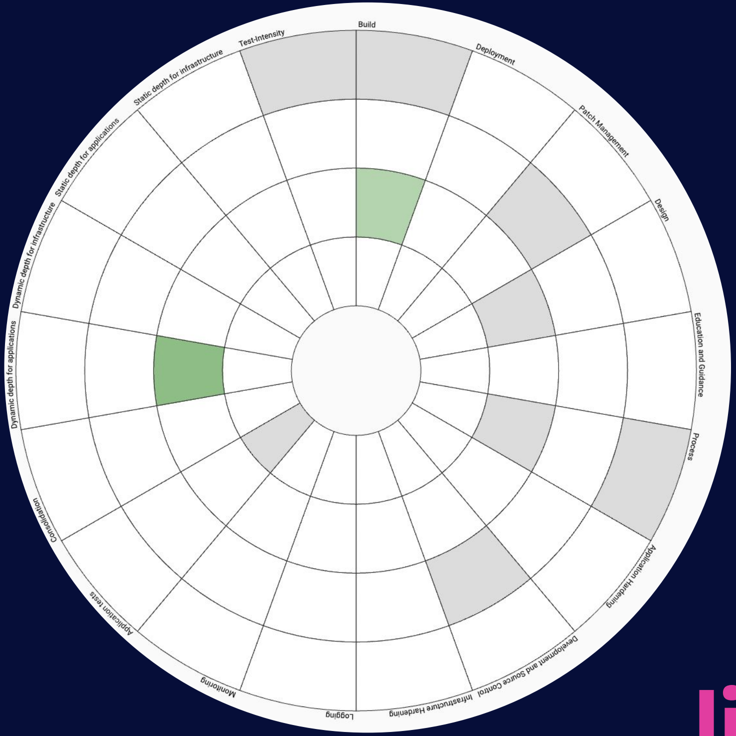


What is DSOMM?

OWASP Devsecops Maturity Model

Dimensions

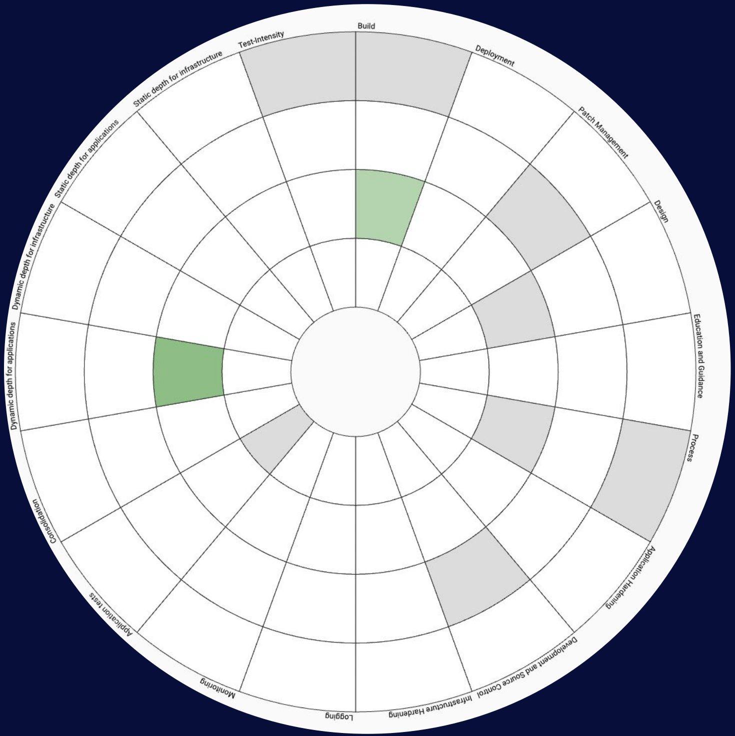
- Build & Deployment
- Culture & Organization
- Implementation
- Information Gathering
- Test & Verification



What is DSOMM?

OWASP Devsecops Maturity Model

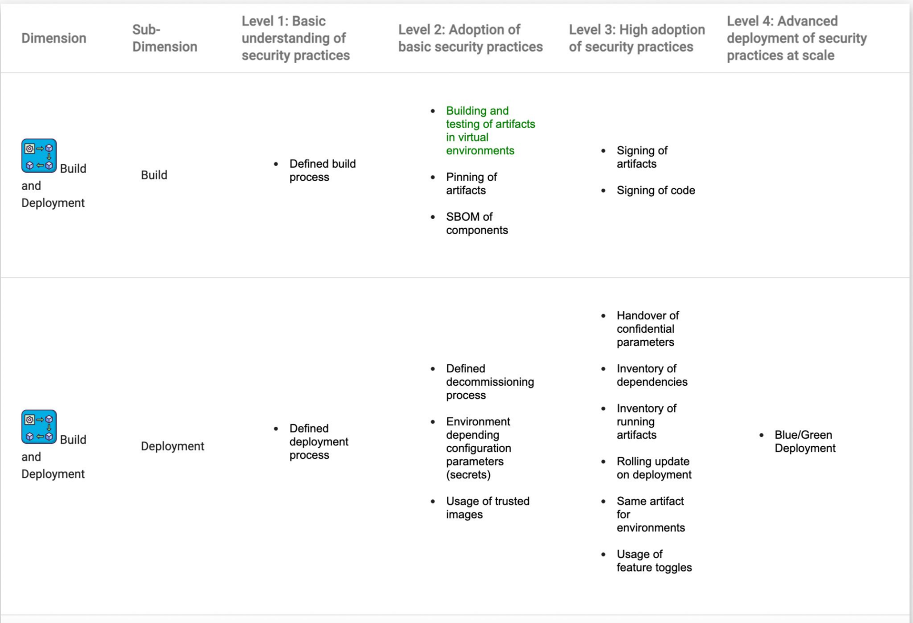
Provides mapping to ISO 27001 controls and requirements



What is DSOMM?

OWASP Devsecops Maturity Model

Each Dimension provides a sub-dimension with (up to) four distinct maturity levels to assess and track maturity across your DevSecOps lifecycle



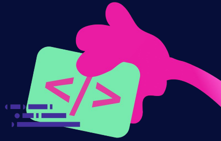
Dimension	Sub-Dimension	Level 1: Basic understanding of security practices	Level 2: Adoption of basic security practices	Level 3: High adoption of security practices	Level 4: Advanced deployment of security practices at scale
Build and Deployment	Build	- Defined build process	- Building and testing of artifacts in virtual environments - Pinning of artifacts - SBOM of components	- Signing of artifacts - Signing of code	
Build and Deployment	Deployment	- Defined deployment process	- Defined decommissioning process - Environment depending configuration parameters (secrets) - Usage of trusted images	- Handover of confidential parameters - Inventory of dependencies - Inventory of running artifacts - Rolling update on deployment - Same artifact for environments - Usage of feature toggles	- Blue/Green Deployment

What is DSOMM?

OWASP Devsecops Maturity Model

Operationalizing DSOMM

Automation focus areas from DSOMM



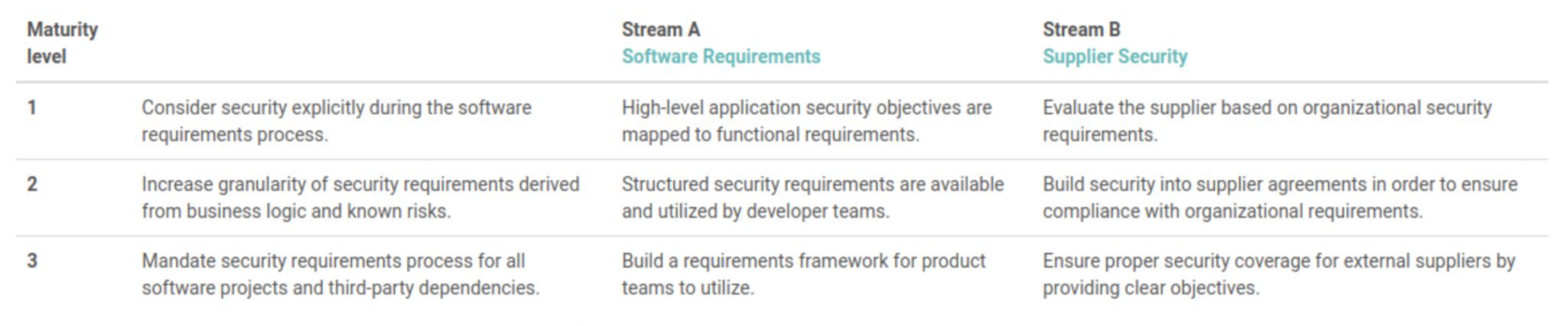
With the excellent open source tooling available today the following technical dimensions of DSOMM can be practically enabled ***and even automated quickly***

Dimension	Sub-Dimension
Implementation	Application Hardening
Implementation	Development & Source Control
Test & Verification	Application Tests
Test & Verification	Consolidation

Operationalizing DSOMM

Implementation - Application Hardening

DSOMM provides the following maturity levels for application hardening both for software development and supplier security:



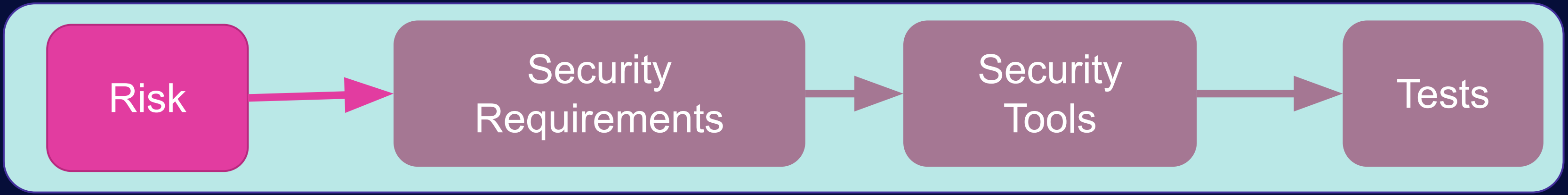
Maturity level		Stream A Software Requirements	Stream B Supplier Security
1	Consider security explicitly during the software requirements process.	High-level application security objectives are mapped to functional requirements.	Evaluate the supplier based on organizational security requirements.
2	Increase granularity of security requirements derived from business logic and known risks.	Structured security requirements are available and utilized by developer teams.	Build security into supplier agreements in order to ensure compliance with organizational requirements.
3	Mandate security requirements process for all software projects and third-party dependencies.	Build a requirements framework for product teams to utilize.	Ensure proper security coverage for external suppliers by providing clear objectives.

Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1

Consider security explicitly during the software requirements process

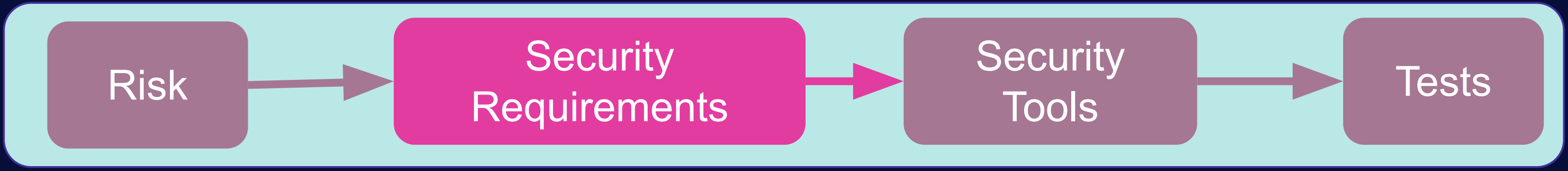


Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Software Requirements: High level application security objectives are mapped to functional requirements

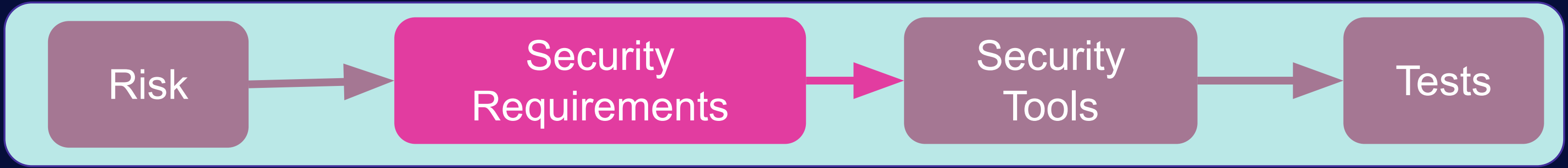


Operationalizing DSOMM

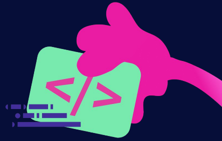
Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Application Security Objective: **Prevent data or code within the app from being stolen or hijacked**



Operationalizing DSOMM



Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Software Requirements: High level application security objectives are mapped to functional requirements

Functional & Organization Security Requirements:

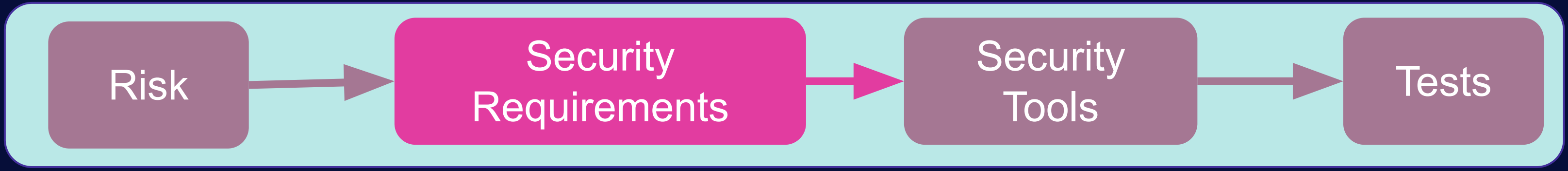
1. **Scan Code for Vulnerabilities**
2. **Scan Code for Hard-Coded Secrets**

Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Supplier Requirements: Evaluate the supplier based on organization security requirements

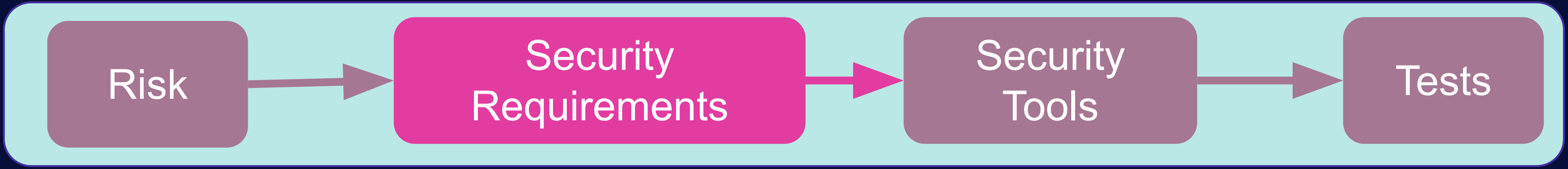


Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Organization Security Requirements: All software libraries are updated with vulnerabilities remediated



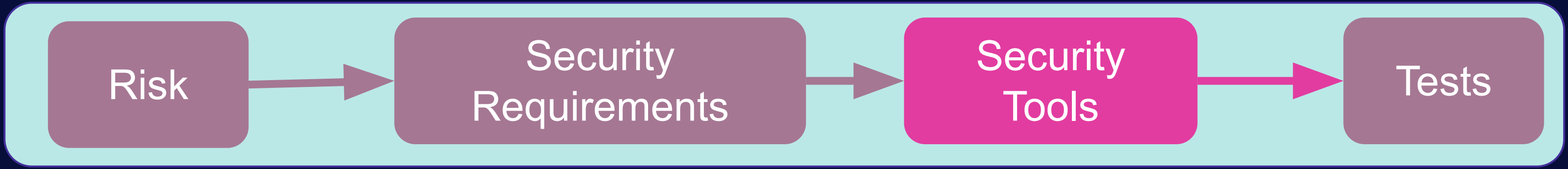
Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

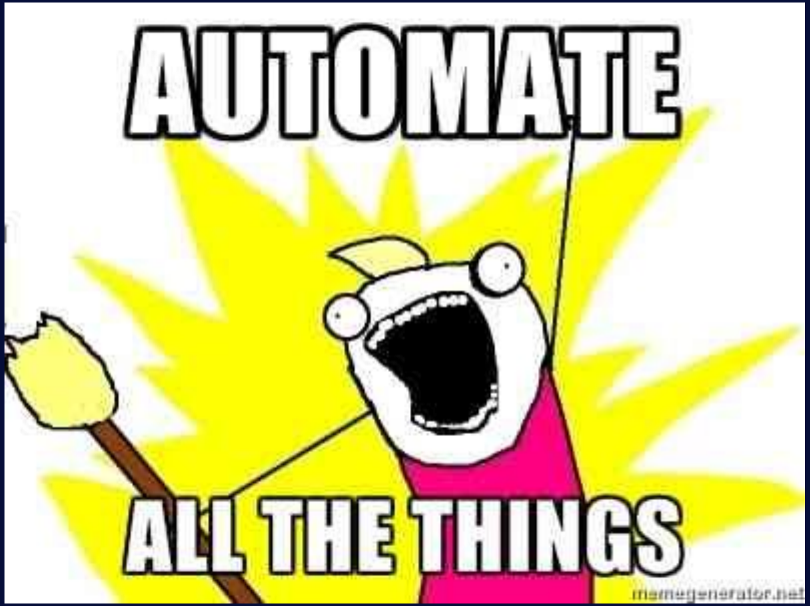
Functional & Organization Security Requirements:

1. **Scan Code Dependencies for Vulnerabilities:**

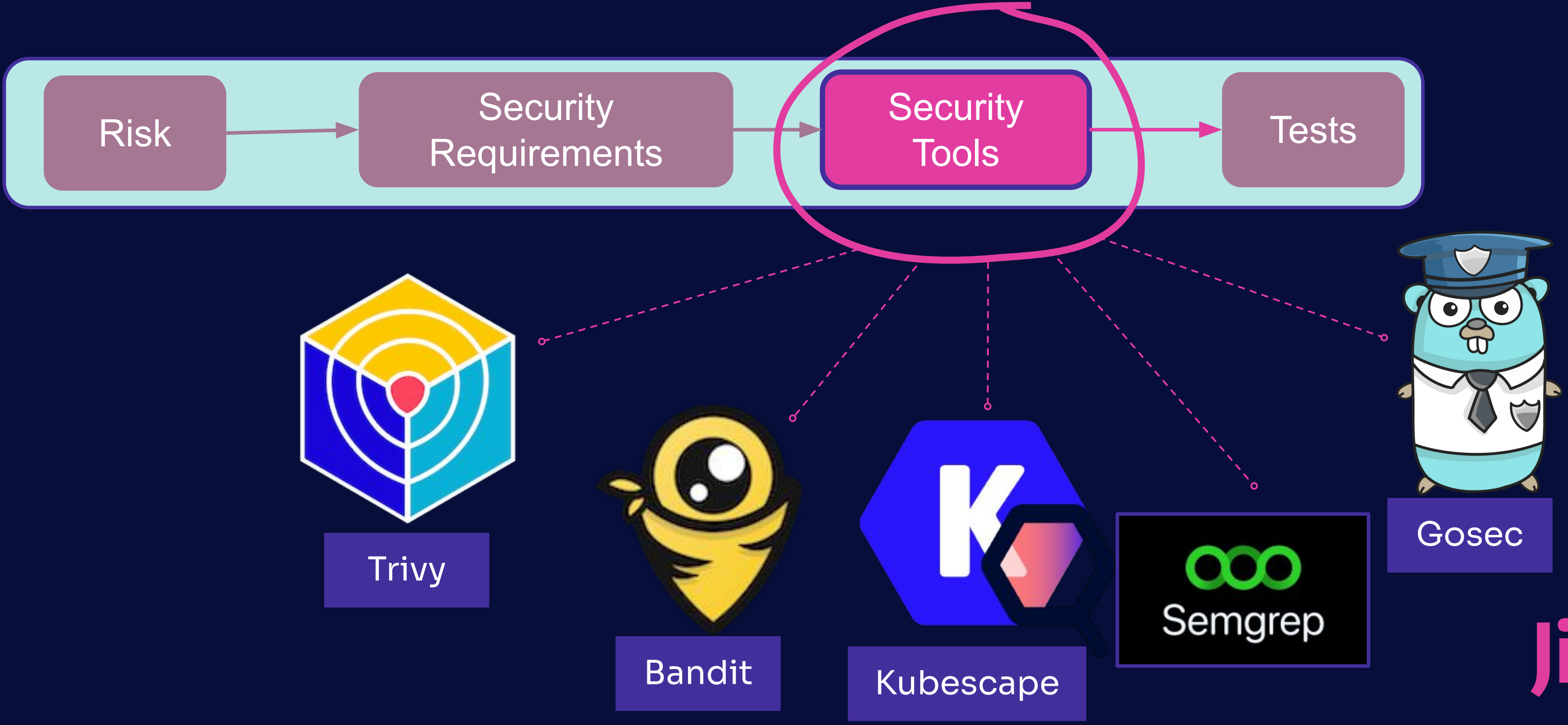


Operationalizing DSOMM

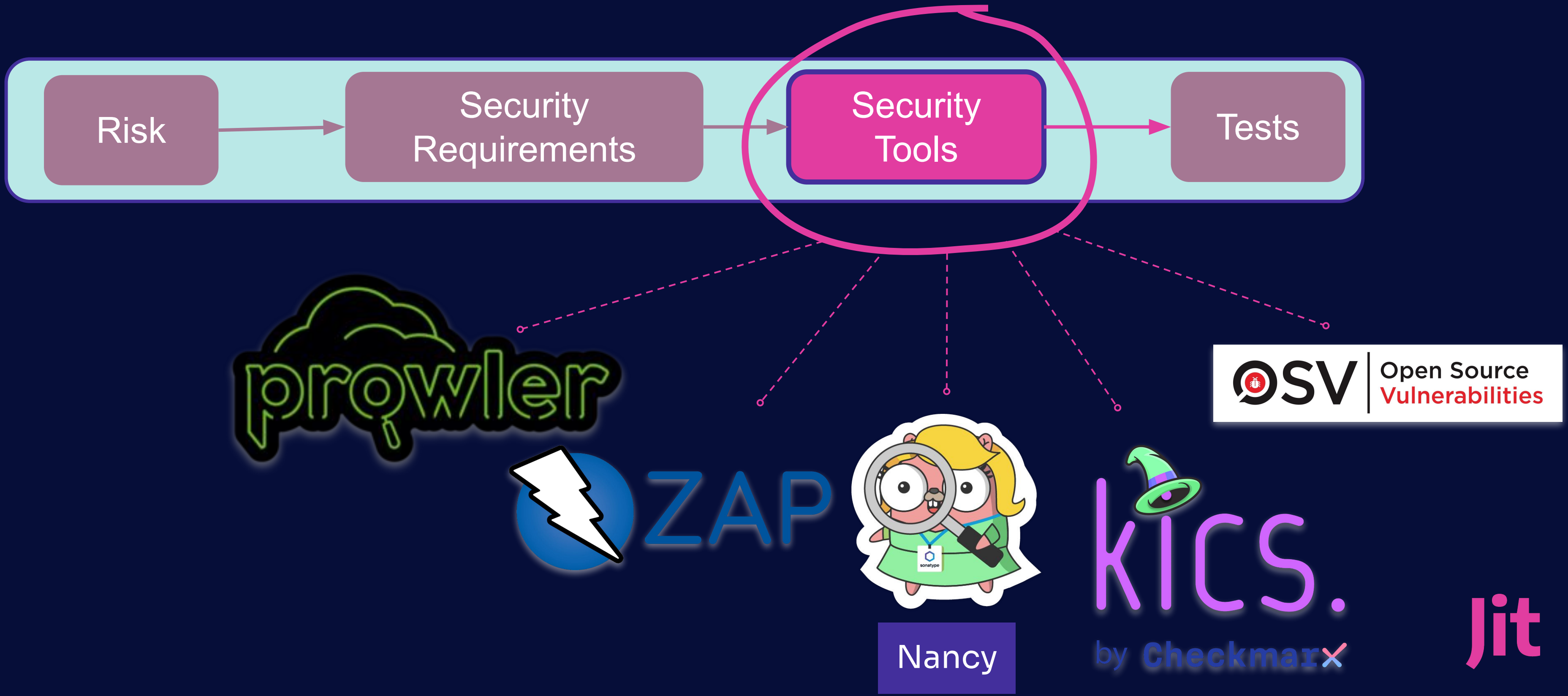
Let’s automate implementation of DSOMM maturity levels using OSS security tools!



Operationalizing DSOMM



Operationalizing DSOMM



Risk → Security Requirements → Security Tools → Tests

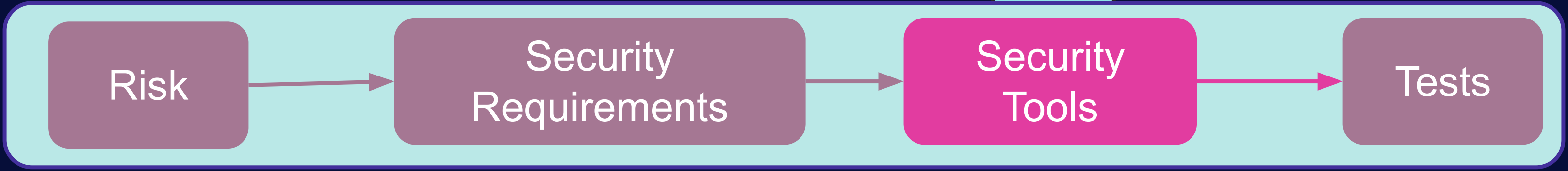
Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Scan Code for Vulnerabilities:

a. **GoSec**: Go



Risk → Security Requirements → Security Tools → Tests

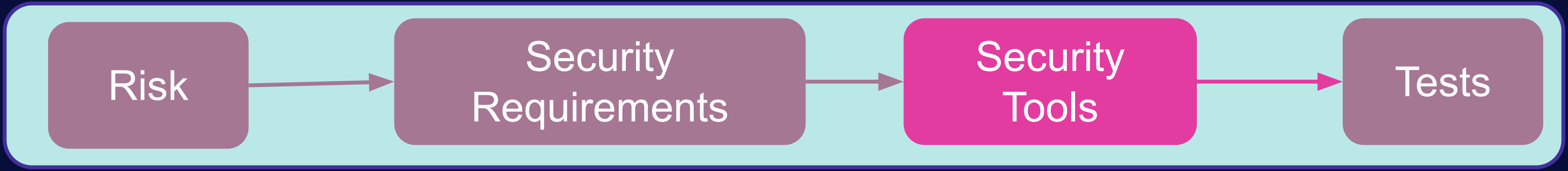
Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Scan Code for Vulnerabilities:

a. **GoSec**: Go
b. **SemGrep**: Python, Java, JavaScript, TypeScript, Kotlin, Scala, C#



Risk

Security Requirements

Security Tools

Tests

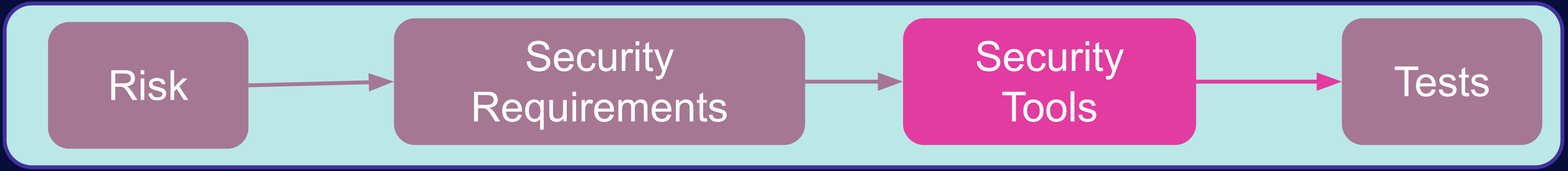
Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

a. OSV-Scanner: Python, PHP

OSV | Open Source Vulnerabilities



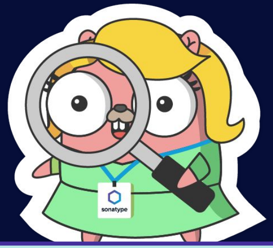
Operationalizing DSOMM

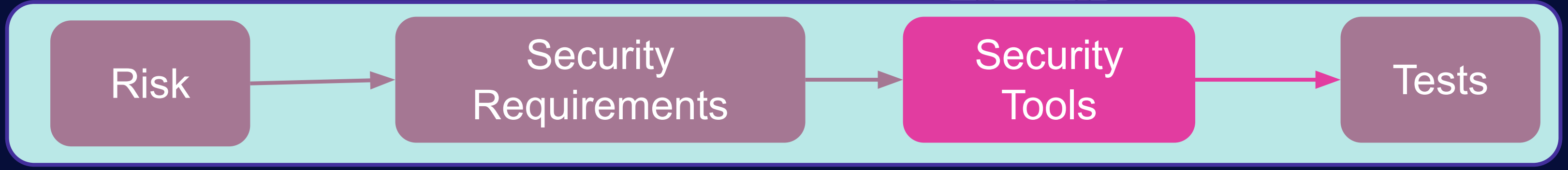
Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

a. OSV-Scanner: Python, PHP

b. Nancy: Go



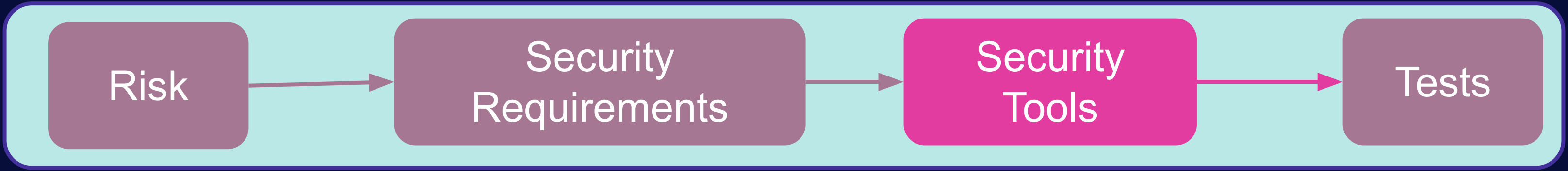


Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

a. OSV-Scanner: Python, PHP
b. Nancy: Go
c. **npm-audit: Javascript, Typescript, Node JS**



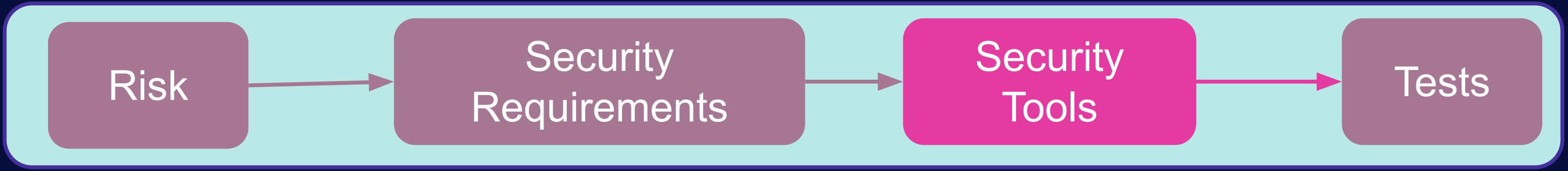
Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 1 - Consider security explicitly during the software requirements process

Scan Code for Hard-Coded Secrets:

a. **GitLeaks**: Supports multiple languages

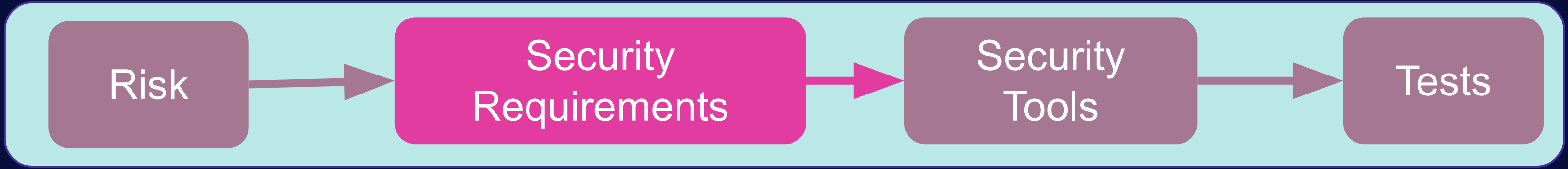


Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 2

Increase granularity of security requirements derived from business logic and known risks



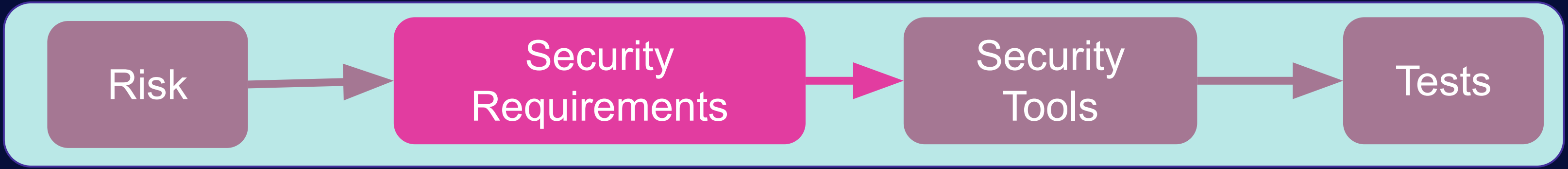
Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 2

Increase granularity of security requirements derived from business logic and known risks

- ✓ **Software Requirements:** Structured security requirements are available and utilized by developer teams



Operationalizing DSOMM

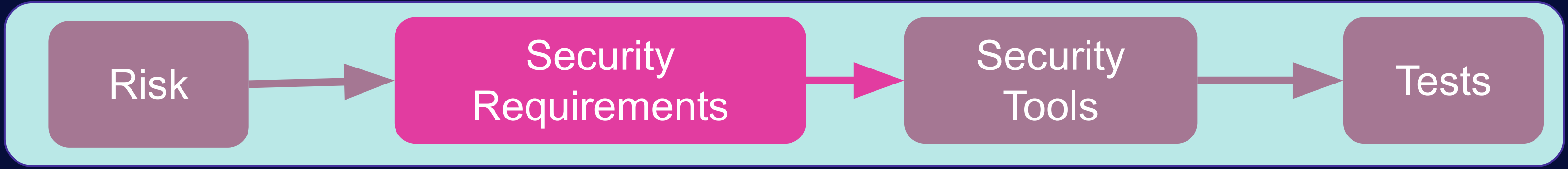
Implementation - Application Hardening

Maturity Level 2

Increase granularity of security requirements derived from business logic and known risks

Software Requirements: Structured security requirements are available and utilized by developer teams

✓ **Supplier Requirements:** Build security into supplier agreements in order to ensure compliance with organizational requirements



Risk → Security Requirements → Security Tools → Tests

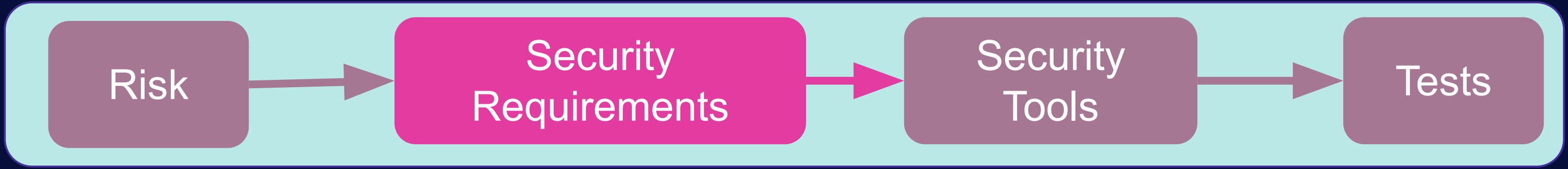
Operationalizing DSOMM

Implementation - Application Hardening

Maturity Level 3

Mandate security requirements for all software projects and third party dependencies

- ✓ **Software Requirements:** Build a requirements framework for product teams to utilize



Jit

Operationalizing DSOMM

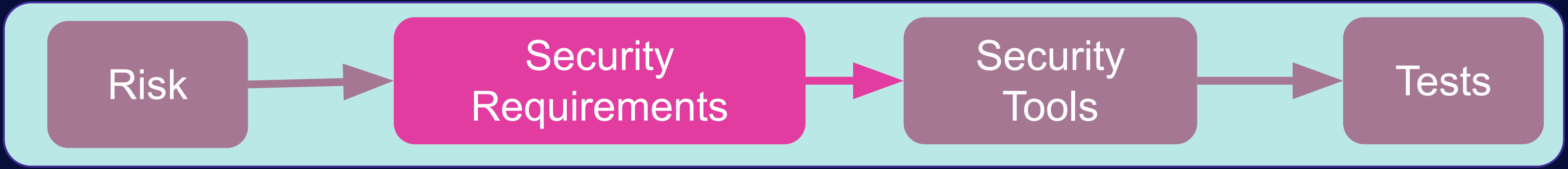
Implementation - Application Hardening

Maturity Level 3

Mandate security requirements for all software projects and third party dependencies

Software Requirements: Build a requirements framework for product teams to utilize

✓ **Supplier Requirements:** Ensure proper security coverage for external suppliers by providing clear objectives



Jit

Operationalizing DSOMM

Yayyy!

Our first sub-dimension is completed!

Operationalizing DSOMM

Implementation - Development & Source Control

Maturity Level 1 - Source Control Protection & Versioning

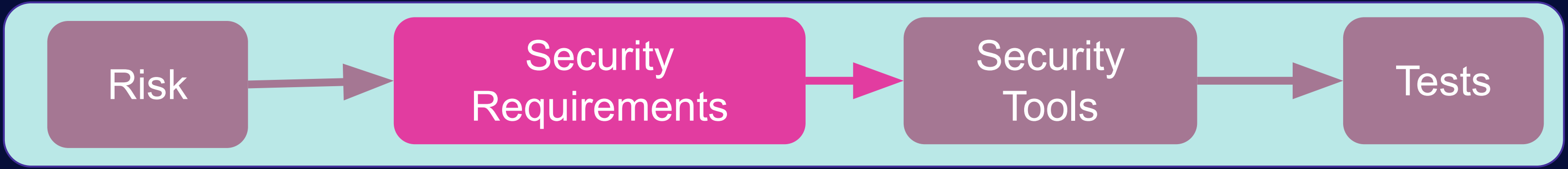
- SCM implementation through GitHub

Maturity Level 2 & 3 - Pre-Commit checks and validations

- Enable code scans from Application Hardening sub-domain and enable / enforce Branch Protection

Maturity Level 4 - Local development linting & style checks performed

- Require developers to install and run linter on local IDE



Risk → Security Requirements → Security Tools → Tests

Operationalizing DSOMM

Implementation - Development & Source Control

- ✓ Maturity Level 1 - Source Control Protection & Versioning

Operationalizing DSOMM

Implementation - Development & Source Control

Maturity Level 1 - Source Control Protection & Versioning

✓ Maturity Level 2 & 3 - Pre-Commit checks and validations

OSV Open Source Vulnerabilities

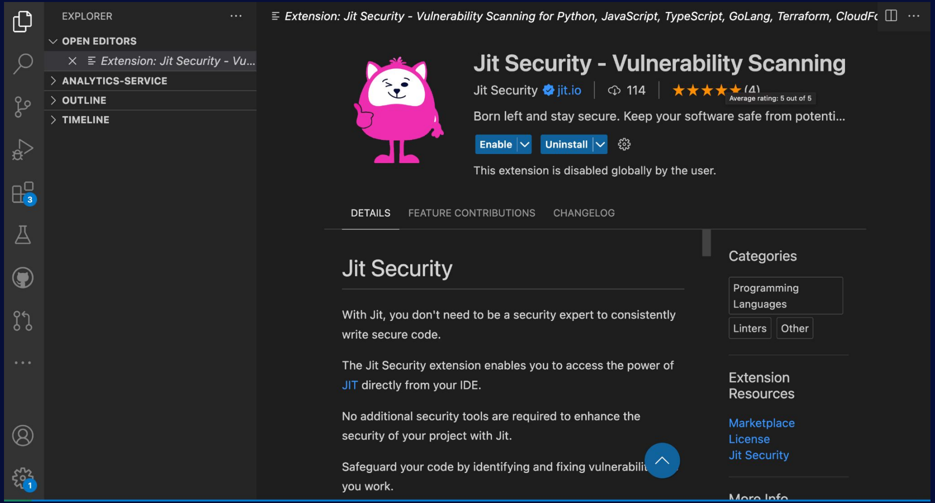
gitLeaks

Semgrep

Operationalizing DSOMM

Implementation - Development & Source Control

- ✓ Maturity Level 4 - Local development linting & style checks performed



Extension: Jit Security - Vulnerability Scanning for Python, JavaScript, TypeScript, GoLang, Terraform, CloudFo

Jit Security - Vulnerability Scanning

Jit Security jit.io | 114 | ★★★★★ (4) Average rating: 5 out of 5

Born left and stay secure. Keep your software safe from potenti...

This extension is disabled globally by the user.

Jit Security

With Jit, you don't need to be a security expert to consistently write secure code.

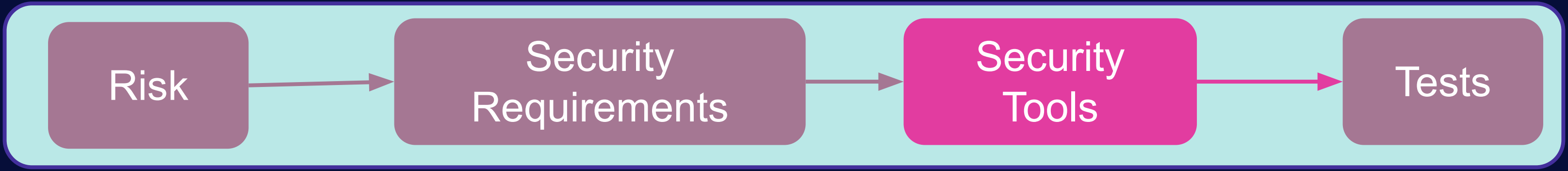
The Jit Security extension enables you to access the power of JIT directly from your IDE.

No additional security tools are required to enhance the security of your project with Jit.

Safeguard your code by identifying and fixing vulnerabilit you work.

Operationalizing DSOMM

Implementation - Infrastructure Hardening



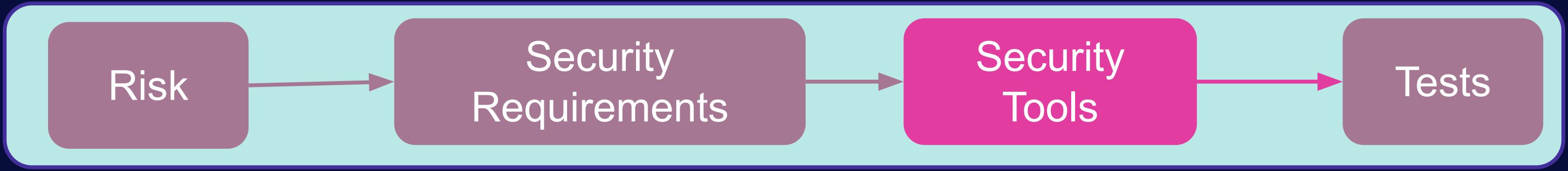
Risk → Security Requirements → Security Tools → Tests

Operationalizing DSOMM

Implementation - Infrastructure Hardening

Scan Your Infrastructure:

a. **Containers / Kubernetes**: Trivy / Kubescape



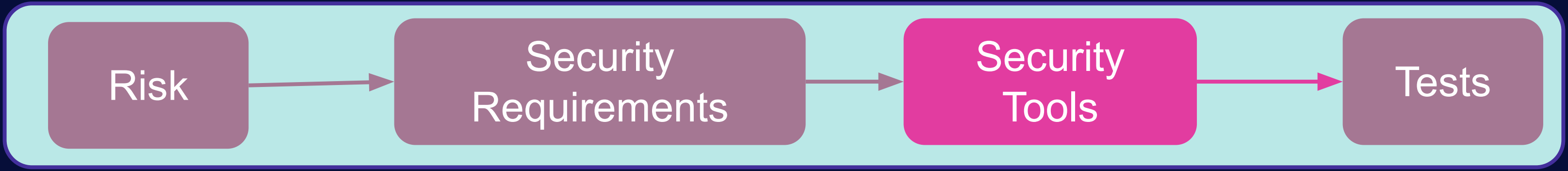
Operationalizing DSOMM

Implementation - Infrastructure Hardening

Scan Your Infrastructure:

a. **Containers / Kubernetes**: Trivy / Kubescape

b. **Cloud + Infrastructure as Code**: Prowler / Kics



Risk → Security Requirements → Security Tools → Tests

Operationalizing DSOMM

Tests & Verification - Application Tests

Maturity Level 1 - N/A

Maturity Level 2 - Security unit tests for important components

Operationalizing DSOMM

Tests & Verification - Application Tests

Maturity Level 1 - N/A

Maturity Level 2 - Security unit tests for important components

Maturity Level 3 - Security integration tests for important components

Operationalizing DSOMM

Tests & Verification - Application Tests

Maturity Level 1 - N/A

Maturity Level 2 - Security unit tests for important components

Maturity Level 3 - Security integration tests for important components

Maturity Level 4 - High coverage of security related module and integration tests, smoke test

Operationalizing DSOMM

Tests & Verification - Consolidation

Definition of quality gates, Simple false positive treatment, Treatment of defects with severity high or higher

Maturity Level 1

Jit

Operationalizing DSOMM

Tests & Verification - Consolidation

Definition of quality gates, Simple false positive treatment, Treatment of defects with severity high or higher

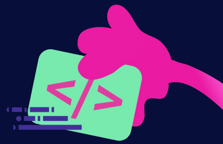
- Implement and enforce code, infrastructure, pipeline and 3rd party scans for each PR (Pull Request)

Maturity Level 1

Jit

Operationalizing DSOMM

Tests & Verification - Consolidation



Definition of quality gates, Simple false positive treatment, Treatment of defects with severity high or higher

- Implement and enforce code, infrastructure, pipeline and 3rd party scans for each PR (Pull Request)
- Implement Branch Protection to require approvals prior to PR merge

Maturity Level 1

Operationalizing DSOMM

Tests & Verification - Consolidation

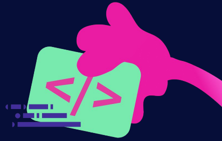
Definition of quality gates, Simple false positive treatment, Treatment of defects with severity high or higher

- Implement and enforce code, infrastructure, pipeline and 3rd party scans for each PR (Pull Request)
- Implement Branch Protection to require approvals prior to PR merge
- False positive review process

Maturity Level 1

Operationalizing DSOMM

Tests & Verification - Consolidation



Definition of quality gates, Simple false positive treatment, Treatment of defects with severity high or higher

- Implement and enforce code, infrastructure, pipeline and 3rd party scans for each PR (Pull Request)
- Implement Branch Protection to require approvals prior to PR merge
- False positive review process
- Triage of all “highs”

Maturity Level 1

Jit

Operationalizing DSOMM

Tests & Verification - Consolidation

Simple visualization of defects

Dashboard and reports of vulnerability backlog and vulnerabilities per PR

Maturity Level 2

Maturity Level 1

Operationalizing DSOMM

Tests & Verification - Consolidation

Integration of vulnerability issues into the development process, Treatment of defects with moderate severity, Usage of a vulnerability management system

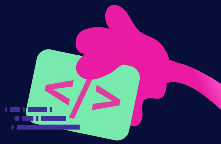
Maturity Level 3

Maturity Level 2

Maturity Level 1

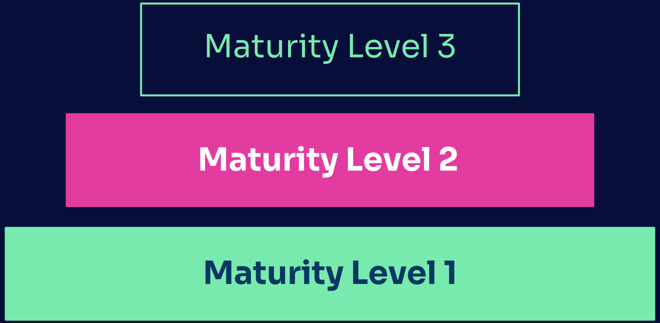
Operationalizing DSOMM

Tests & Verification - Consolidation



Integration of vulnerability issues into the development process, Treatment of defects with moderate severity, Usage of a vulnerability management system

- Triage of all “moderates” and “highs”



Maturity Level 3

Maturity Level 2

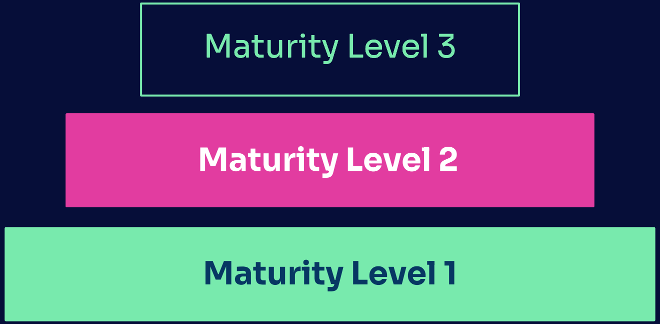
Maturity Level 1

Operationalizing DSOMM

Tests & Verification - Consolidation

Integration of vulnerability issues into the development process, Treatment of defects with moderate severity, Usage of a vulnerability management system

- Triage of all “moderates” and “highs”
- Tracking, managing and reporting on vulnerabilities

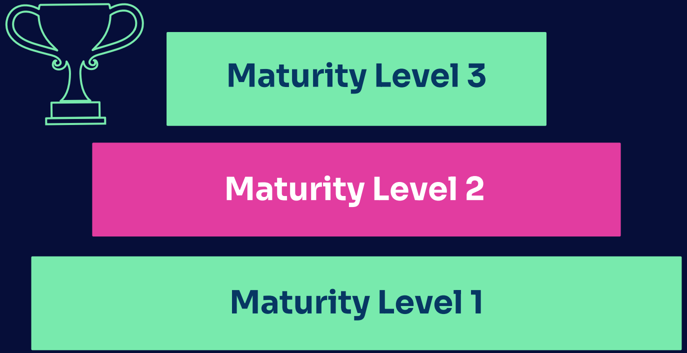


Operationalizing DSOMM

Tests & Verification - Consolidation

Integration of vulnerability issues into the development process, Treatment of defects with moderate severity, Usage of a vulnerability management system

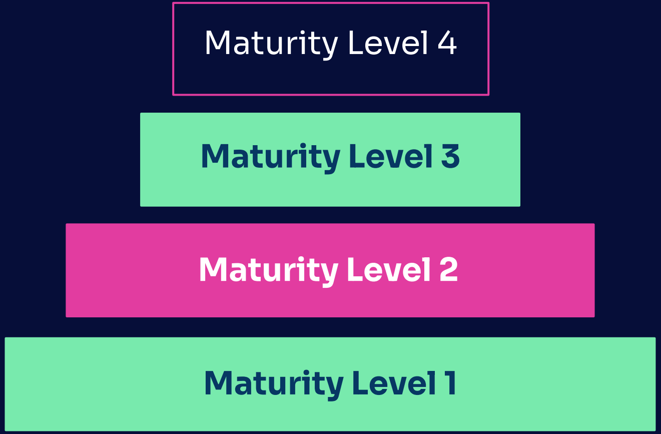
- Triage of all “moderates” and “highs”
- Tracking, managing and reporting on vulnerabilities
- Vulnerabilities as part of the development process



Operationalizing DSOMM

Tests & Verification - Consolidation

Advanced visualization of defects, Reproducible defect tickets, Treatment of all defects



Maturity Level 4

Maturity Level 3

Maturity Level 2

Maturity Level 1

Operationalizing DSOMM

Tests & Verification - Consolidation

Advanced visualization of defects, Reproducible defect tickets, Treatment of all defects

- Detailed vulnerability metrics w/ root causes

Maturity Level 4
Maturity Level 3
Maturity Level 2
Maturity Level 1

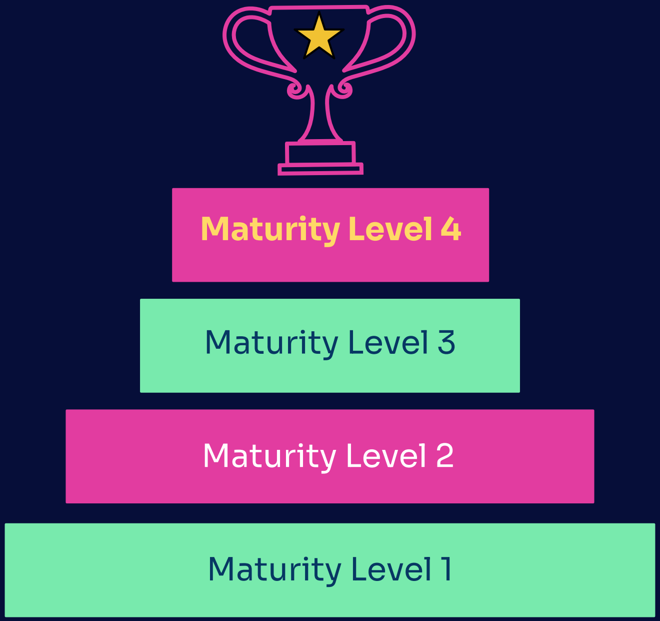
Operationalizing DSOMM

Tests & Verification - Consolidation

Advanced visualization of defects, Reproducible defect tickets, Treatment of all defects

- Detailed vulnerability metrics w/ root causes
- Triage of all “lows”, “moderates” and “highs”

ACHIEVEMENT UNLOCKED



Maturity Level 4

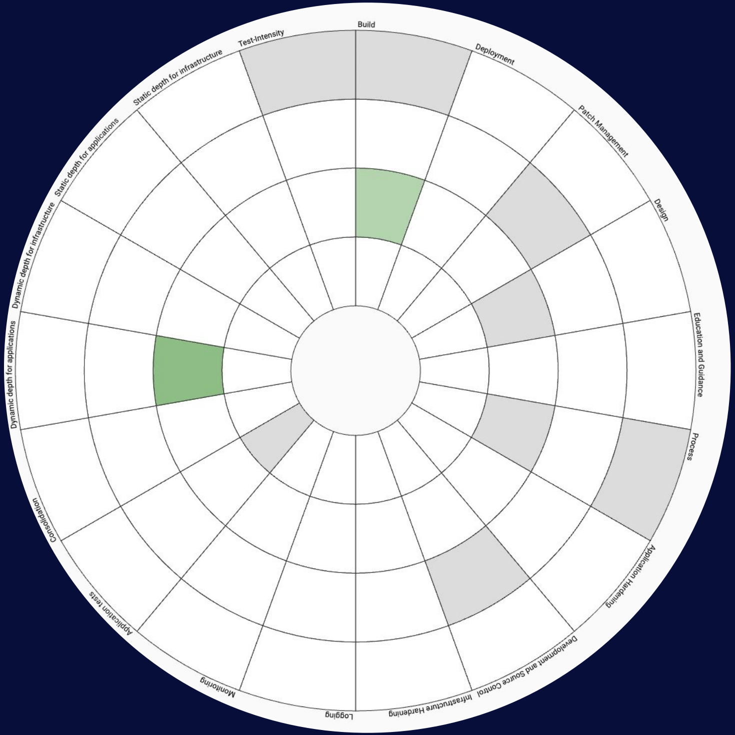
Maturity Level 3

Maturity Level 2

Maturity Level 1

What is DSOMM?

OWASP Devsecops Maturity Model



Security Maturity Models

Overview and Background

Leading Security Maturity Models

Did you see the value ?!

Security Maturity Models

Overview and Background

Leading Security Maturity Models

- Translation to Leadership
- Current State Analysis
- Gap Prioritization
- ✓ Tools & Automation
- Resource Needs
- ✓ Risk Exposure

Security Maturity Models

Overview and Background

Leading Security Maturity Models

- Translation to Leadership
- ✓ Tools & Automation
- ✓ Current State Analysis
- Resource Needs
- ✓ Gap Prioritization
- ✓ Risk Exposure

Security Maturity Models

Overview and Background

Leading Security Maturity Models

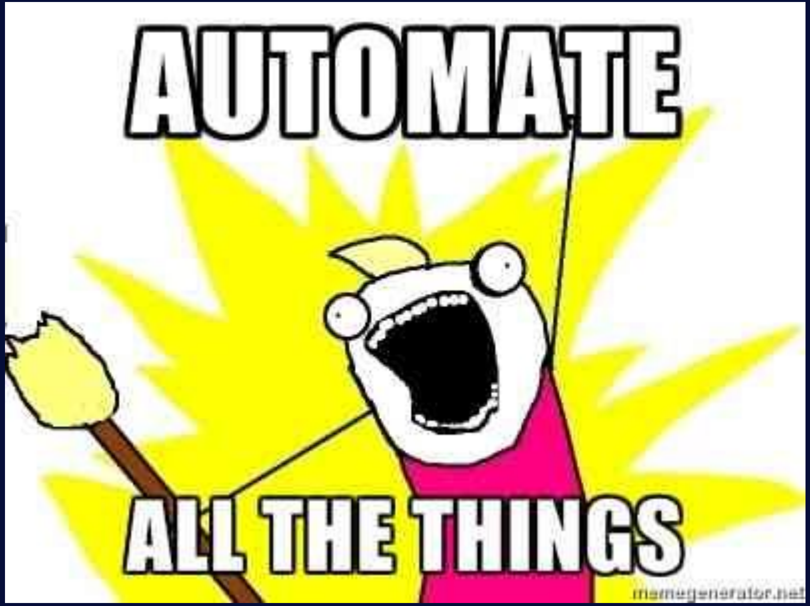
- ✓ Translation to Leadership
- ✓ Current State Analysis
- ✓ Gap Prioritization
- ✓ Tools & Automation
- ✓ Resource Needs
- ✓ Risk Exposure

Security Maturity Models

Overview and Background

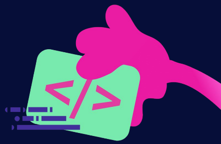
Leading Security Maturity Models

What’s Next?

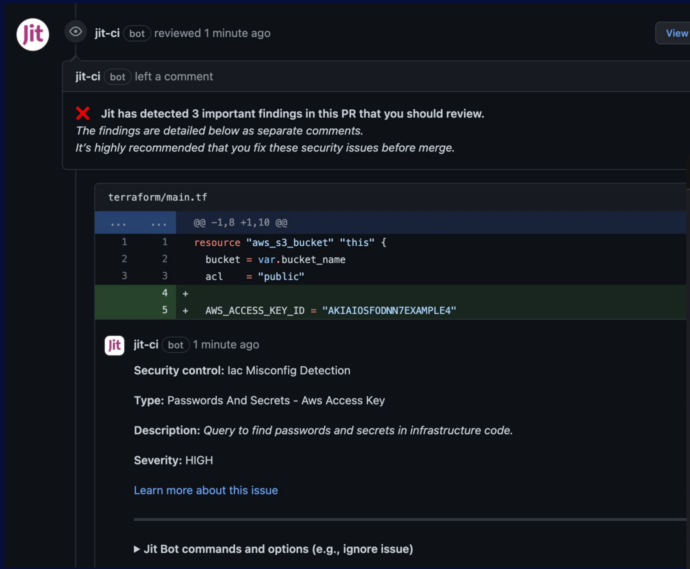


Taking it to the next level: Jit

Automating security plans using OSS security tools orchestration



Dev-native experience using PR comments



jit-ci bot reviewed 1 minute ago

jit-ci bot left a comment

❌ Jit has detected 3 important findings in this PR that you should review.
The findings are detailed below as separate comments.
It's highly recommended that you fix these security issues before merge.

terraform/main.tf

```
@@ -1,8 +1,10 @@
resource "aws_s3_bucket" "this" {
  bucket = var.bucket_name
  acl    = "public"
+
+ AWS_ACCESS_KEY_ID = "AKIAIOSFODNN7EXAMPLE4"
```

jit-ci bot 1 minute ago

Security control: Iac Misconfig Detection

Type: Passwords And Secrets - Aws Access Key

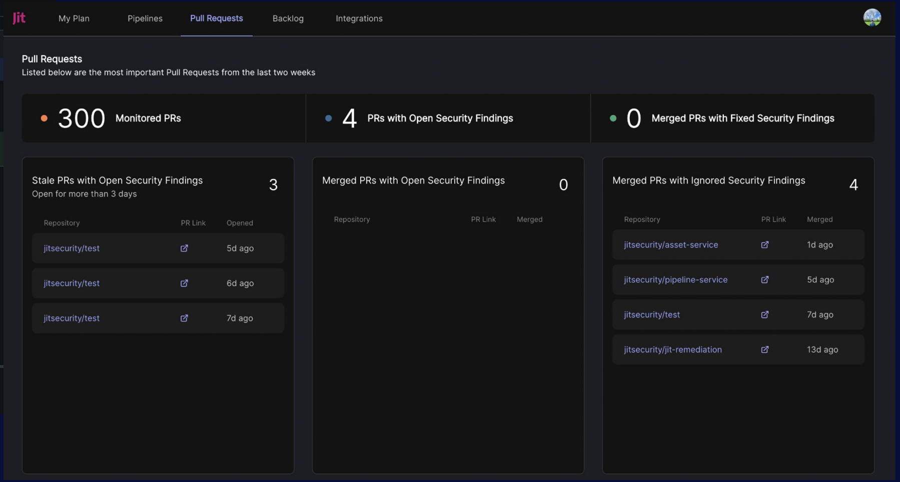
Description: *Query to find passwords and secrets in infrastructure code.*

Severity: HIGH

Learn more about this issue

▸ Jit Bot commands and options (e.g., ignore issue)

Prioritize findings per PR to tackle the most pressing issues "Just in Time"!



Pull Requests

Listed below are the most important Pull Requests from the last two weeks

● 300 Monitored PRs

● 4 PRs with Open Security Findings

● 0 Merged PRs with Fixed Security Findings

Stale PRs with Open Security Findings 3
Open for more than 3 days

Repository	PR Link	Opened
jitsecurity/test	↗	5d ago
jitsecurity/test	↗	6d ago
jitsecurity/test	↗	7d ago

Merged PRs with Open Security Findings 0

Repository	PR Link	Merged

Merged PRs with Ignored Security Findings 4

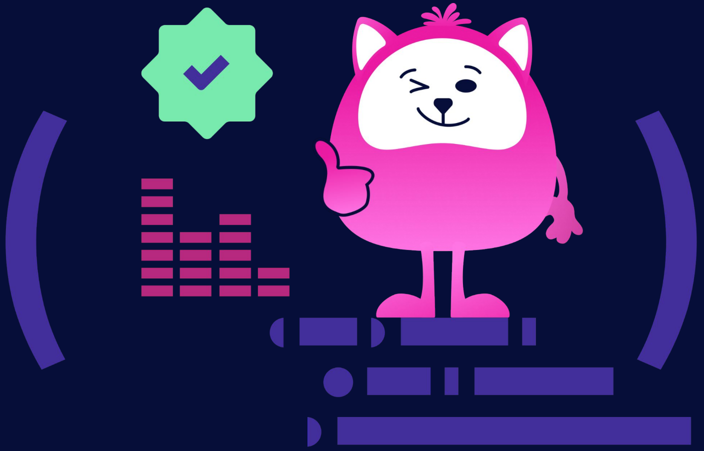
Repository	PR Link	Merged
jitsecurity/asset-service	↗	1d ago
jitsecurity/pipeline-service	↗	5d ago
jitsecurity/test	↗	7d ago
jitsecurity/jit-remediation	↗	13d ago

Jit

Thank you

Intrigued? **Available now** at jit.io

Questions? Contact me at raz@jit.io



https://forms.gle/uMZXPejZHcPW2jjQA

Scan This !

RAFFLE!

RAFFLE!

Win Airpods 3!!!

Jit