



DSOMM

Key Steps to Achieving an Application Security Program

Timo Pagel

Agenda



Solid Foundation



Analyze Security Practices (Quick)



Plan Security Activities



Implement And Measurement Improvements



Summary

Agenda



Solid Foundation



Analyze Security Practices (Quick)



Plan Security Activities



Implement And Measurement Improvements



Summary


Solid Foundation



- Know production applications
 - Applications
 - Team
 - Contact
 - (Protection requirement)
 - (Regulatory requirements)
- Know security tooling and processes

Inventories



Dimension	Sub-Dimension	Level 1: Basic understanding of security practices	Level 2: Adoption of basic security practices	Level 3: High adoption of security practices
 Build and Deployment	Deployment	<ul style="list-style-type: none">• Inventory of production components <i>[inventory]</i>	<ul style="list-style-type: none">• Inventory of production artifacts <i>[inventory]</i>	<ul style="list-style-type: none">• Inventory of production dependencies <i>[inventory , sbom]</i>

Agenda



Solid Foundation



Analyze Security Practices (Quick)



Plan Security Activities



Implement And Measurement Improvements



Summary

I do not know my applications



- Scan based on domains (e.g. with OWASP amaas, nmap)
- Assess build pipeline/deployment processes/production clusters

Why Analyze?



- Know where you are?
 - > know where you want/need to be (high level)
- Get budget for security initiatives



How to Analyse



- Iterative (e.g. yearly, bi-yearly)
- Methods: Interview 1:1, Interview Workshop 1:10, Questionnaire

Analyze Security Practices: Frameworks



- Building Security In Maturity Model (BSIMM)
- OWASP Software Assurance Maturity Model (SAMM)
- OWASP DevSecOps Maturity Model (DSOMM)

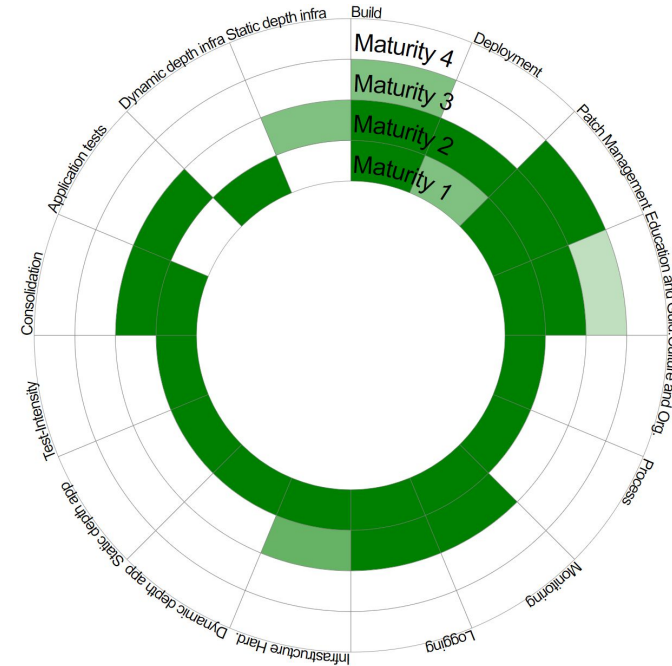
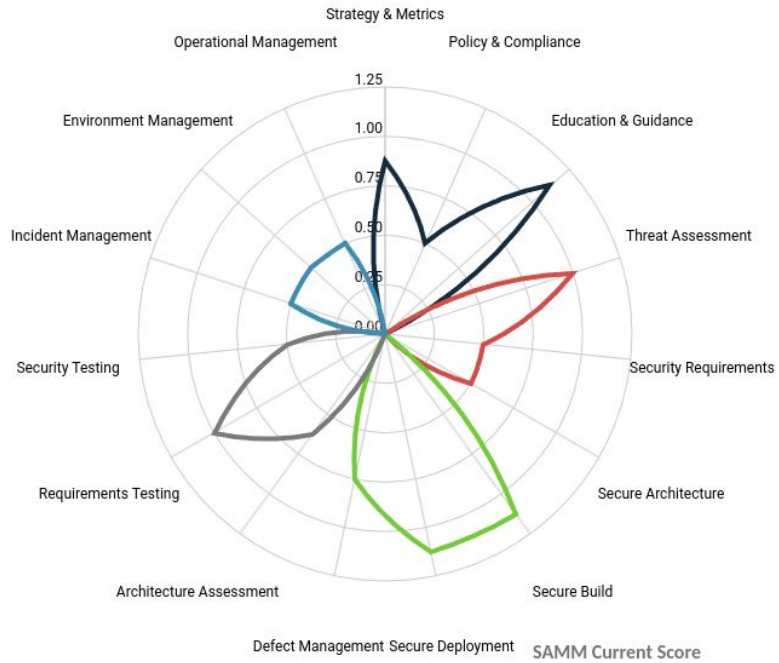
SAMM and DSOMM



- **SAMM** ● “Standard”
 - > High level overview
 - Written for Security
 - Management topics like compliance and governance
 - Planning of high level targets
 - Mapping to ISO in the future

- ⚙ **DSOMM** ● Emerging
 - > Low level overview
 - Written for Dev/Ops
 - Mainly modern DevSecOps topics
 - Planning of concrete targets
 - Mapping to ISO/SAMM
 - ISMS: documentation in DSOMM

Analysis Results



Demo DSOMM to Document the Status





- Consider “includes” for
 - Team activities like security knowledge or security champions
 - Build pipelines
 - Production environments
- Dev/Ops input

Agenda



Solid Foundation



Analyze Security Practices (Quick)



Plan Security Activities



Implement And Measurement Improvements



Summary

Roadmap Planning



Apart from Frameworks



- Organization chart/diagram
 - Number of dev | security | ops
 - Relation between them
- Budget
- Technology stack
- Security tooling stack and processes
- Policies, standards, ...
- Pentest reports and open findings

Roadmap for Applications



- Maturity Model
- Scorecards

Design of a Maturity Model



Dimension	Level 1	Level 2/...	Level 3/n
A	Blue	Grey	Red
B	Grey	Green	Grey
C	Red	Blue	Green

Creation of New of Activities



Take into account:

- Dimension (no redundancy)
- Level
 - Dependencies to other activities
 - Existing tools and processes
 - Outcome for security
 - Ease of implementation

Scorecards



- Often based on a repository
- Scores of activities are accumulated to an overall score
- Less discussions about “We are different”

OSSF Scorecard



Score	Name	Reason	Documentation
10 / 10	Binary-Artifacts	no binaries found in the repo	[...]/checks.md#binary-artifacts
9 / 10	Branch-Protection	branch protection is not maximal on development and all release branches	[...]/checks.md#branch-protection
10 / 10	Code-Review	branch protection for default branch is enabled	[...]/checks.md#code-review
0 / 10	Dependency-Update-Tool	no update tool detected	[...]/checks.md#dependency-update-tool

Scorecards Considerations



- Provides flexibility to teams to choose their implementations
- Easy to integrate
- One application consists of multiple microservices/repos (repo-approach provides limited visibility)
- To design the scores, a maturity model is perfect

Design of a Scorecard



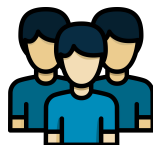
Dimension	Level 1 (3 points)	Level 2 (2 points)	Level 3 (1 point)

Security Team-Led



- Activities are prioritized by sec. team
- Planned activities needs explanation

Product teams-Led



- Potential activities pre-selected by security team
- Product team selects activities
- All defined activities require explanation
- -> Not recommended due to high effort

We are Different



Every team feels to be different than others.

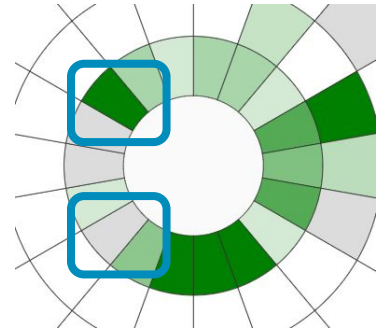
Examples:

- We can not open pull requests, we use SVN
- We are stuck with a 32bit application and can not patch
- We write code for a mainframe and OAuth is not possible

Valid Exceptions



- Valid exceptions shouldn't have impact on maturity level/score of the application
- Visibility:
 - Gray box in a maturity model
 - Mark as not applicable/implemented in a maturity model
 - 0 reachable score on a scorecard

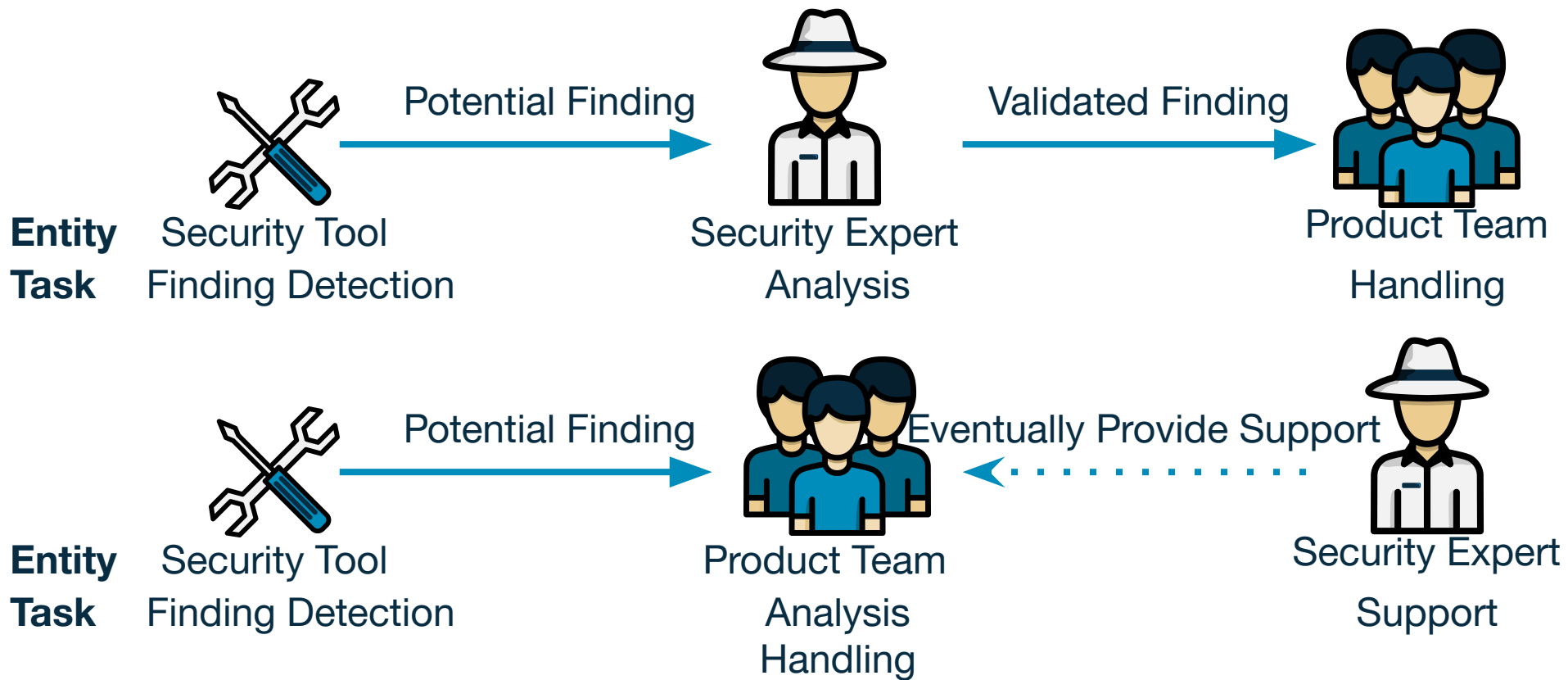


Sample Design Question: What should be done first related to security tools findings?



- Let security engineers validate findings and then push them to product teams
- Push findings directly to product teams without pre-validation

Finding Handling Process Design



Rudimentary Pro/Con Analysis: Sec.



Validating Findings by Security Engineers Pros:

- Ensures accuracy and relevance of findings before they reach product teams
- Reduces false positives, saving development teams time and effort
- Might provides a layer of expertise in assessing the severity and impact of vulnerabilities

Validating Findings by Security Engineers Cons:

- Requires a sufficient number of skilled security engineers, which might be challenging for some organizations
- May slow down the process if security engineers are overloaded with validation tasks
- For Software Composition Analysis findings (known vulnerabilities) I, as a sec. eng., struggle to analysis if it is a false positive/true positive due to a lack of insights in the application

Rudimentary Pro/Con Analysis: Prod. Team



Pushing Findings Directly to Product Teams Pros:

- Accelerates the process by immediately notifying product teams of potential vulnerabilities
- Empowers product teams to take swift action in addressing security issues

Pushing Findings Directly to Product Teams Cons:

- Increases the workload on product teams, potentially leading to frustration

Considerations in Vulnerability Management



Security Training

Provide an understanding of measures

Threat Modeling

Provide an understanding of risk & threat

Security Channel

Create a community of practice

Office Hours

Get in touch with teams

Culture

Defined Build

Gain reliability

Defined Deployment

Gain reliability

Defined Production Env.

Gain reliability

Build and Deployment

Role Security Champion

Spread knowledge

Ownership of Components

Know whom to contact

Responsibilities

Workflow with DSOMM Application



- Fork [DevSecOps-MaturityModel-custom](#)
- Put local changes to activity definitions for default activities in a corresponding custom activity; e.g. overrides to [DevSecOps-MaturityModel-data:src/assets/YAML/default/BuildAndDeployment/Build.yaml](#) go in *src/assets/YAML/custom/BuildAndDeployment/Build.yaml*
- Put local changes to team structure in *src/assets/YAML/meta.yaml*
- Put a build-and-publish workflow around [DevSecOps-MaturityModel-custom](#) that grabs *meta.yaml* and *generated.yaml* from [DevSecOps-MaturityModel-custom](#) and mounts/copies them into the site

Agenda



Solid Foundation



Analyze Security Practices (Quick)



Plan Security Activities



Implement And Measurement Improvements



Summary

Communicate the Plan to the Teams



- Presentation about the plan
- Documentation
 - Wiki OR
 - DSOMM application
 - Level
 - Assessment criteria

What to do with Metrics?



- Drive discussions about strategy
- Motivate teams

DevSecOps Assessment



Assessments performed only
quarterly/yearly/bi-yearly

As a product team, I want fast feedback for
performed (or gone missing) security activities to
stay motivated

Solution: Automatic Metric Gathering

Where to Apply Measurements?

Dimension: Culture and Organization



- Each team has a security champion
- Slack channel #security exchange rate per team
- Threat modeling frequency per team
- Threat modeling quality per team
- Creation of abuse stories in a requirements/planning tool
- Hours of security training per team

Metric Options



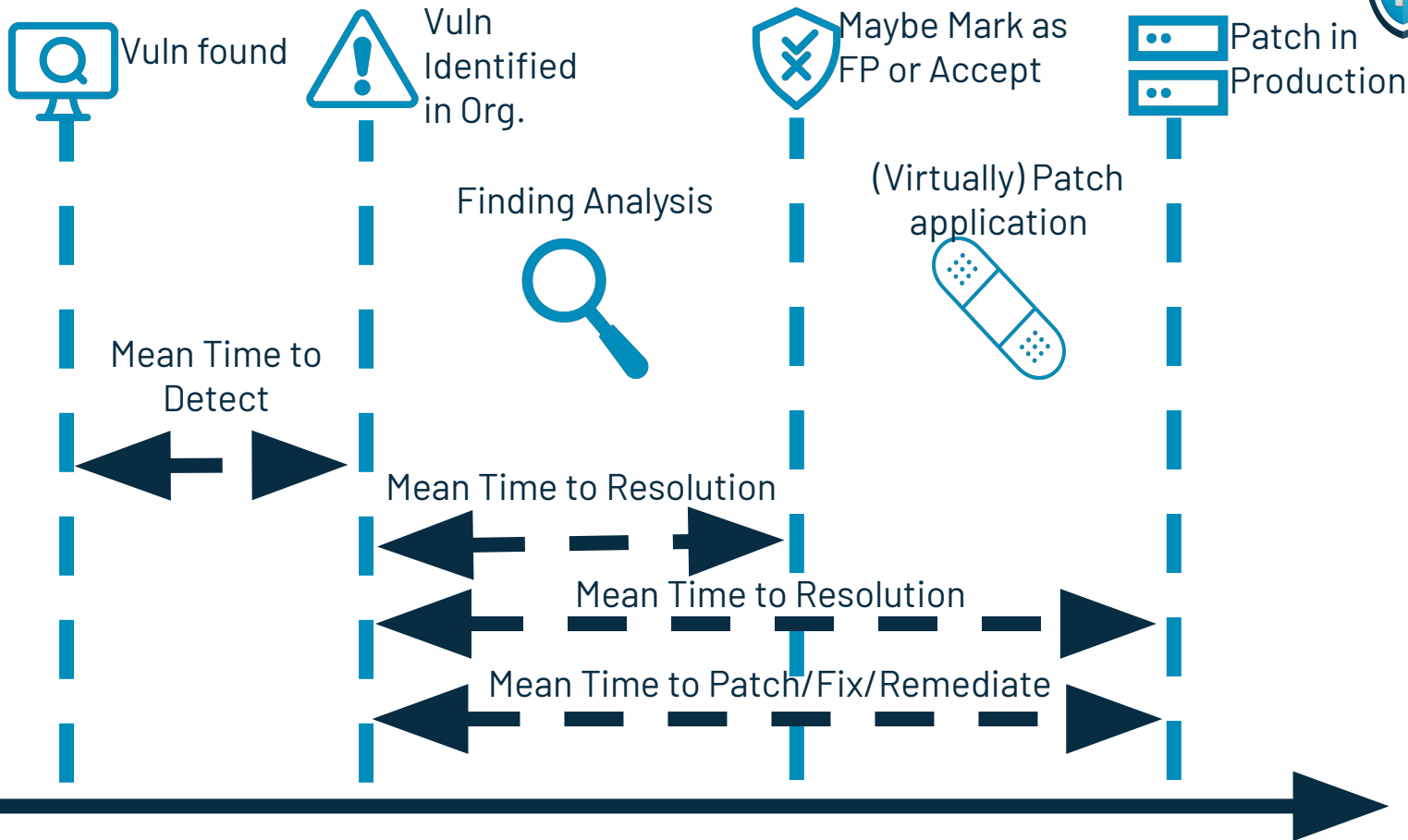
For one goal are different metrics available, e.g.

- Open high/critical vulnerabilities per application
- MTTR per application

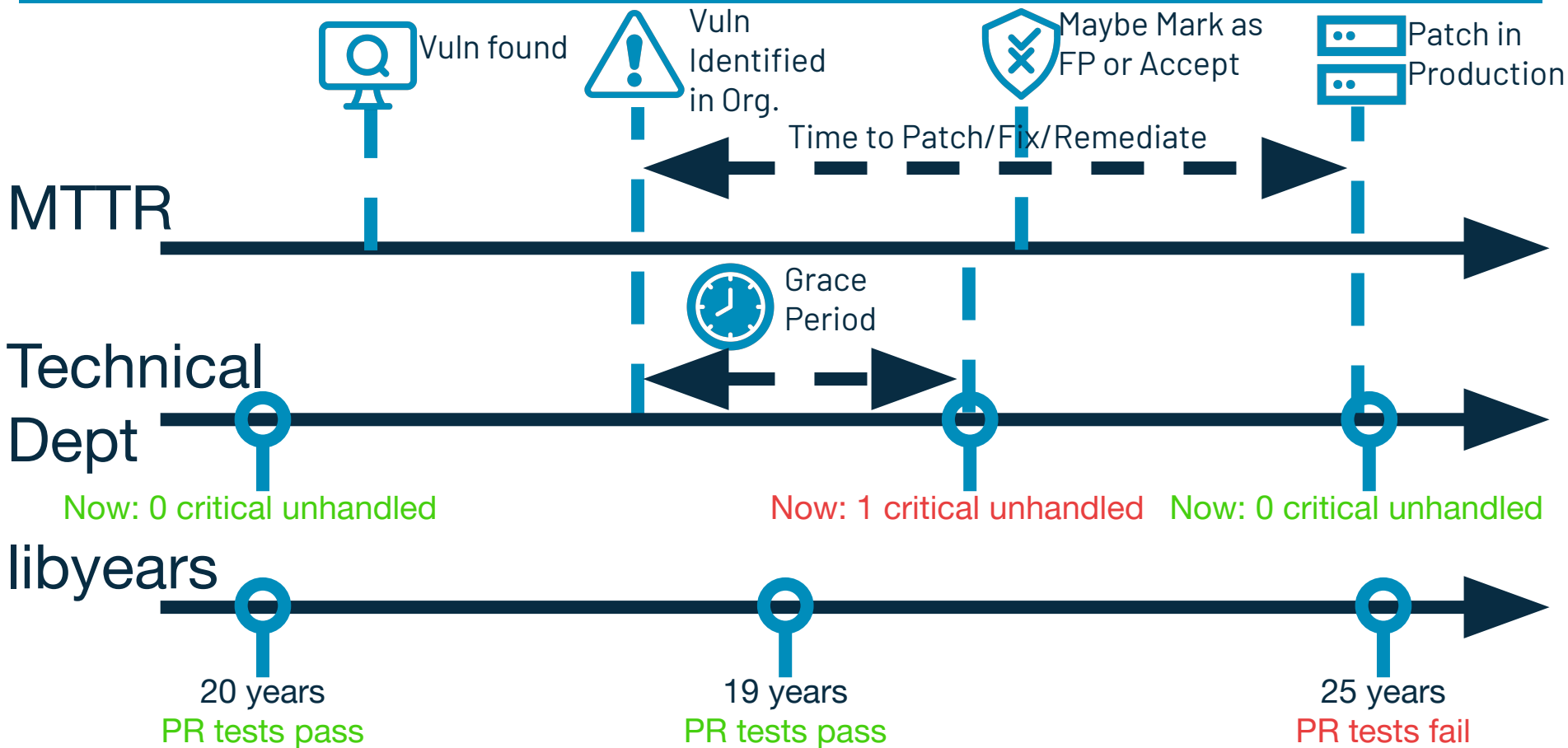
MTTR (Mean Time to Resolution) vs MTTF (Mean Time to Fix)



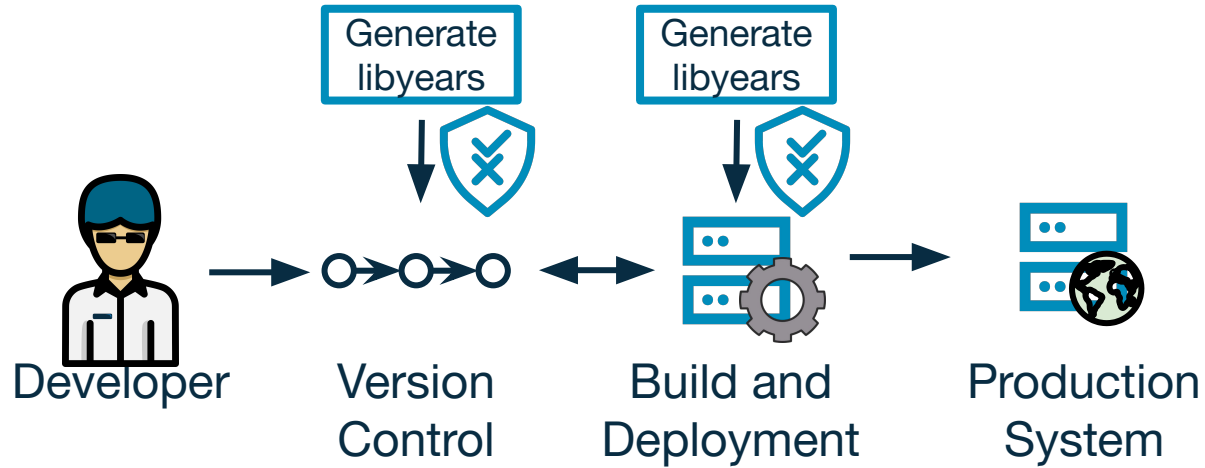
- **MTTR** measures how quickly issues are resolved from the user's perspective. It demonstrates that the **entire vulnerability management process** is functioning effectively. Additionally, MTTR provides insight into how **noisy the security tools** being used are, as frequent false positives can inflate this metric.
- Mean Time to Fix represents the **final goal in addressing vulnerabilities** or issues.



MTTR vs. Technical Dept vs. libyears



Libyears in CI/CD

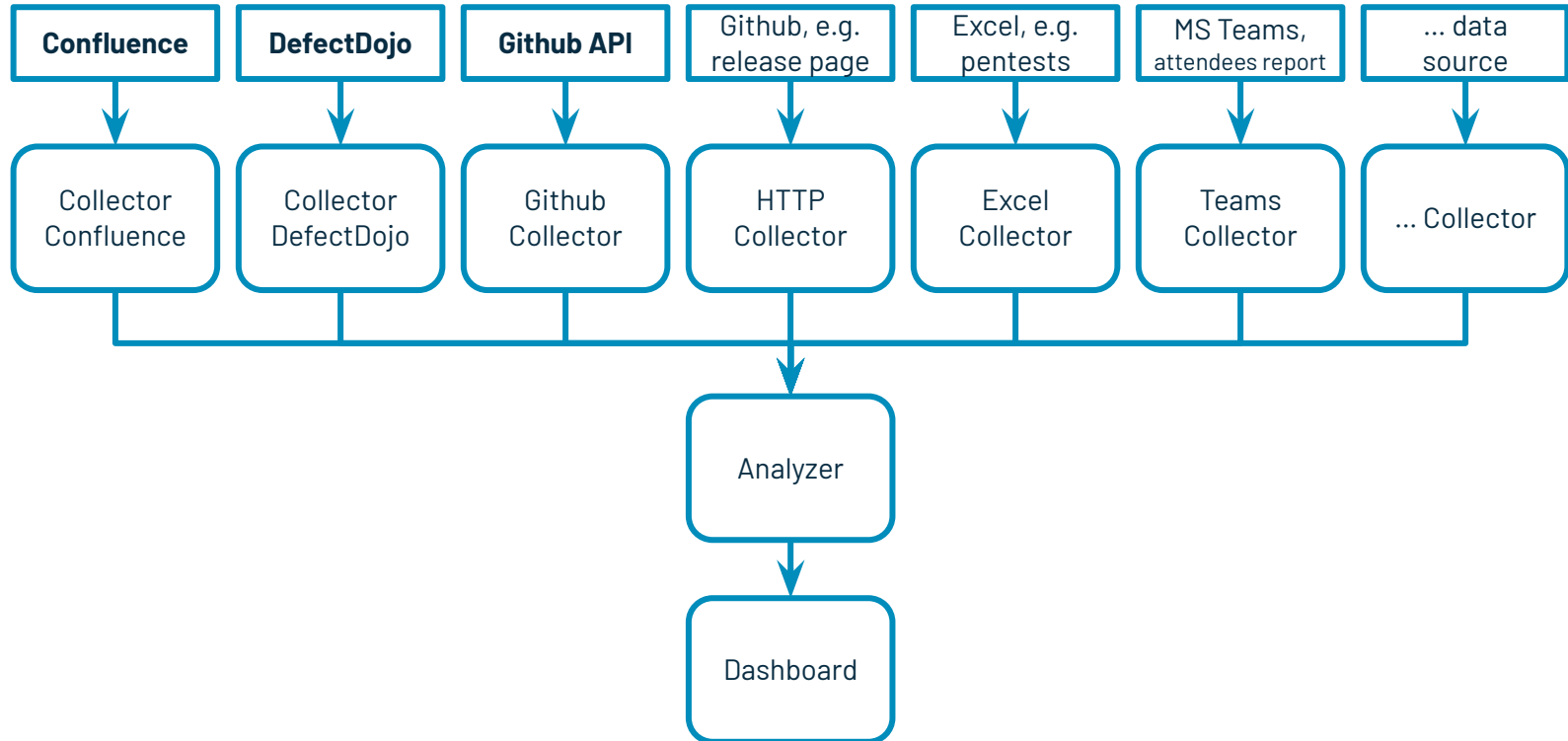


Sample Vulnerability Context



- Vulnerability Severity
- Application Protection Requirement
- Package Manager, e.g.
 - Deb
 - Maven
 - NPM

Potential Collectors



Collector Samples



- **Confluence Collector:** Threat modelings and penetration tests by frequency (implemented)
- **HTTP Collector:** Collects last change of a website, e.g. github release page for last patch
- **Github Collectors:**
 - **Collects Security Settings like Branch Protection enabled**
 - Collects open time of automatic created patch pull requests (e.g. from renovate) to calculate Mean Time to Patch
- **DefectDojo Collector:** Collects Mean Time to Response
- **Excel Collector:** Collects penetration test by frequency
- **Teams Collector:** Collects attendee rate for security trainings

Dashboard Views

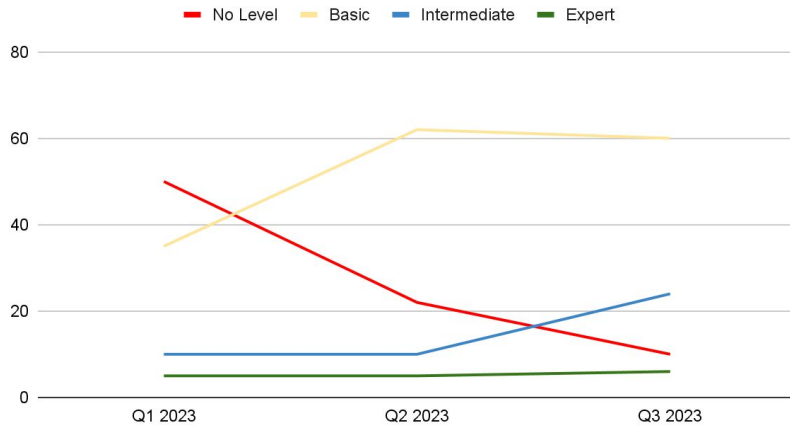


- Summary for the board
- Summary of desired status of a team/application reached?
- Comparison of team/application status between teams/applications
- Current status of a team/application
- Change since last period (e.g. quarter/30 days)

Overview Q1 2023



Teams in Maturity Levels (History)

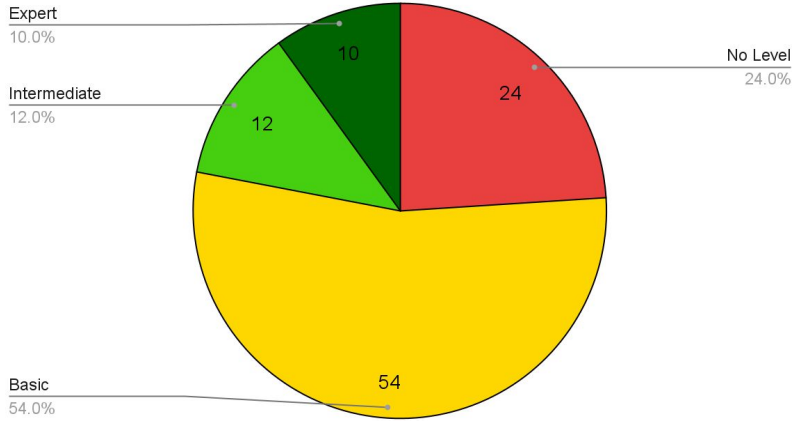


Average new implemented activities: +1 per team in Q3 2023

Top Performer Team	Department	Activities
Frodo	Fellowship of the Ring	+7
Sam	Fellowship of the Ring	+5
Dwalin	Dwarf	+5
Saruman	Sauron	-2
Balrok	Old Age	-4
Gollum	Starren	-4

Current App. Sec. State

Teams in Maturity Levels

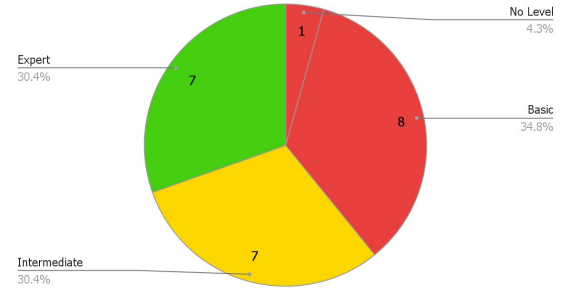


Sample comments:

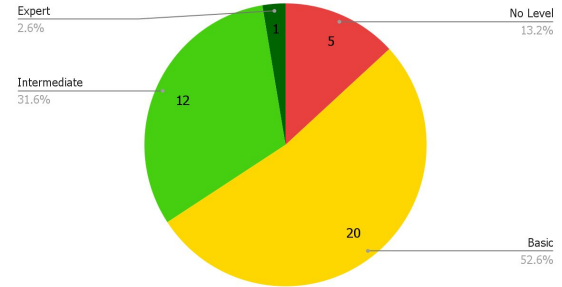
- Good that all critical apps have at least basic, but needs to get better
- High rate apps are also stale in basic

Maybe you want to add a graph "Application Business Value"

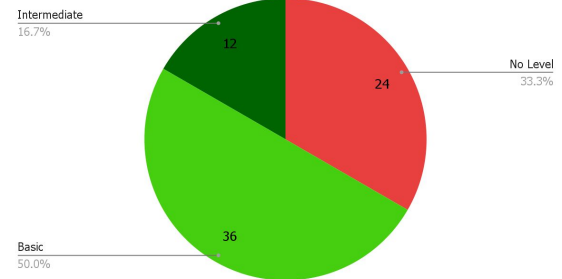
Level of Critical-Rated Applications



Level of High-Rated Applications



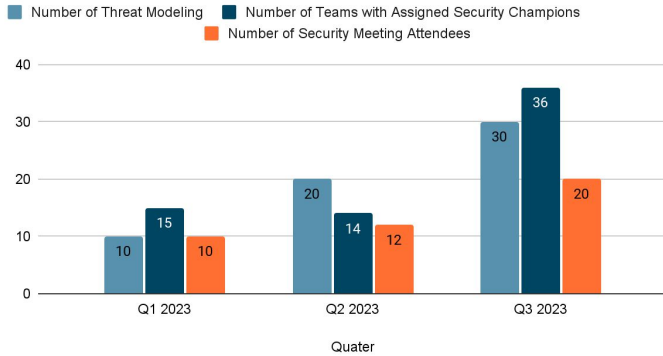
Level of Medium-Rated Applications



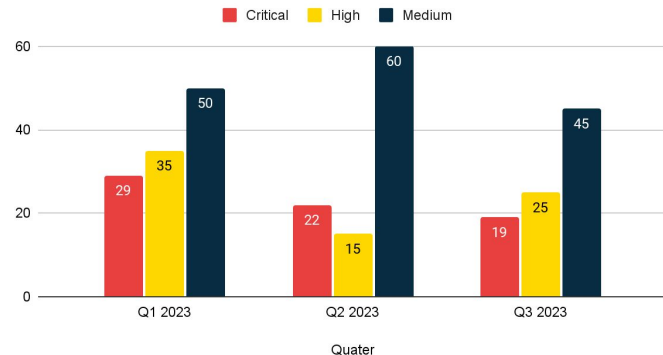
Application Security Program Activities



Important Activities within the organization



Mean Time To Resolve Vulnerabilities (Days)



Security Training Feedback

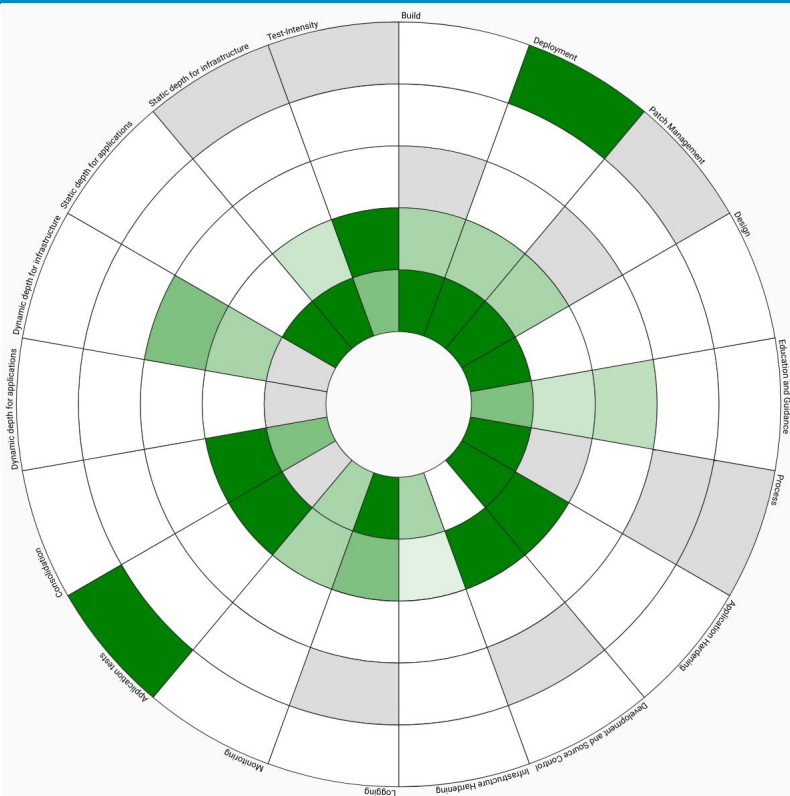
Date	Relevance	Quality	Complexity
Q1 2023	1.3	1.8	2.4
Q2 2023	1.4	1.9	2.8
Q3 2023	1.6	1.8	2.3

1 Very Good/Very Complex;
5 Very Bad/Very Easy

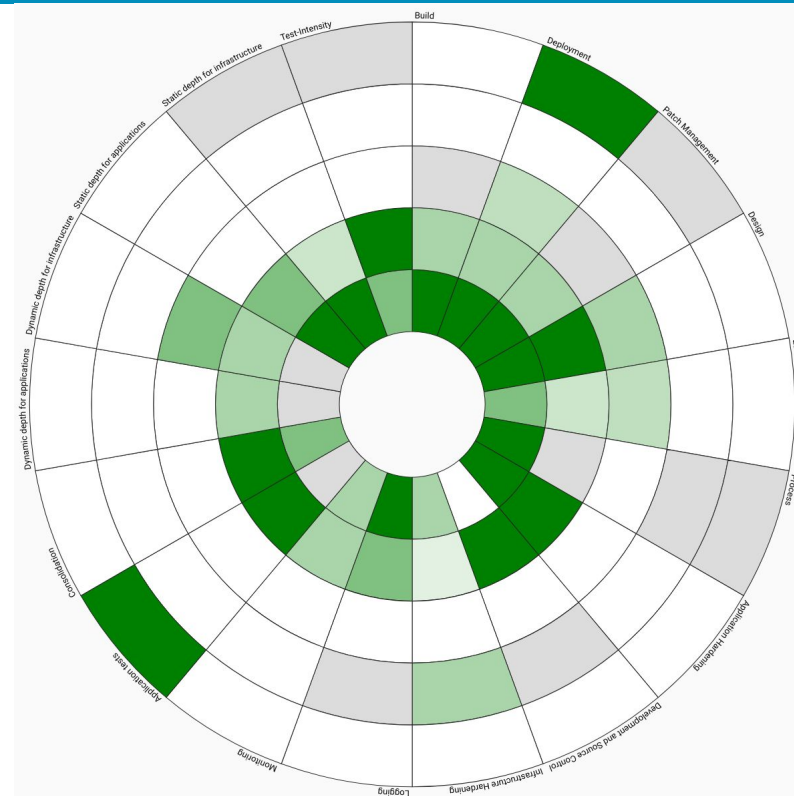
Sample comments:

- Security training meetings are have a lot of visitors (still improvement needed)
- Audience likes the training and finds it useful, therefore, we should mark the test period as successful and continue with it

Areas of Organizations Improvement

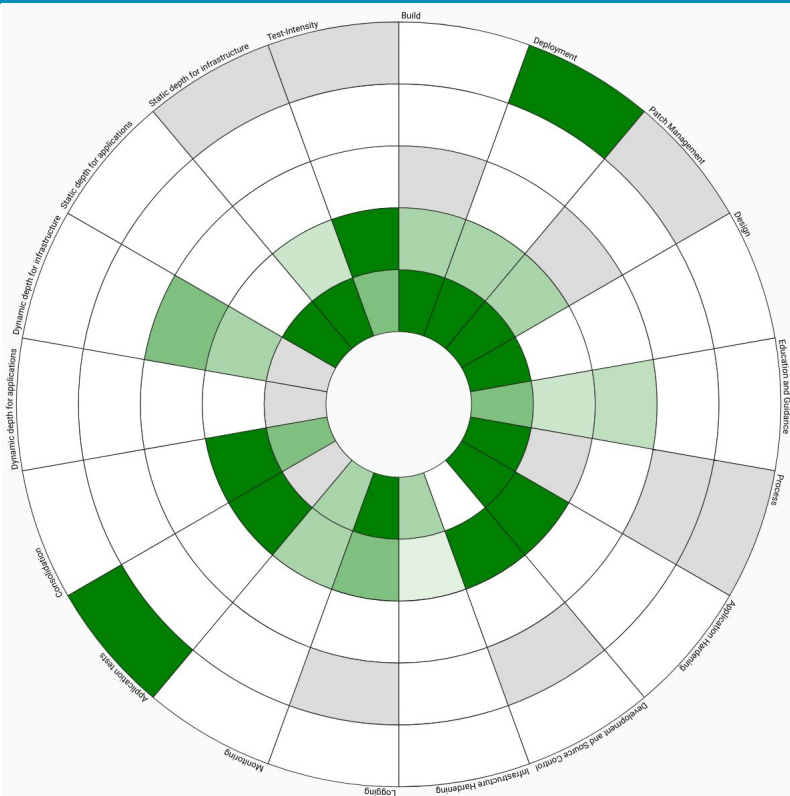


Q1 2023

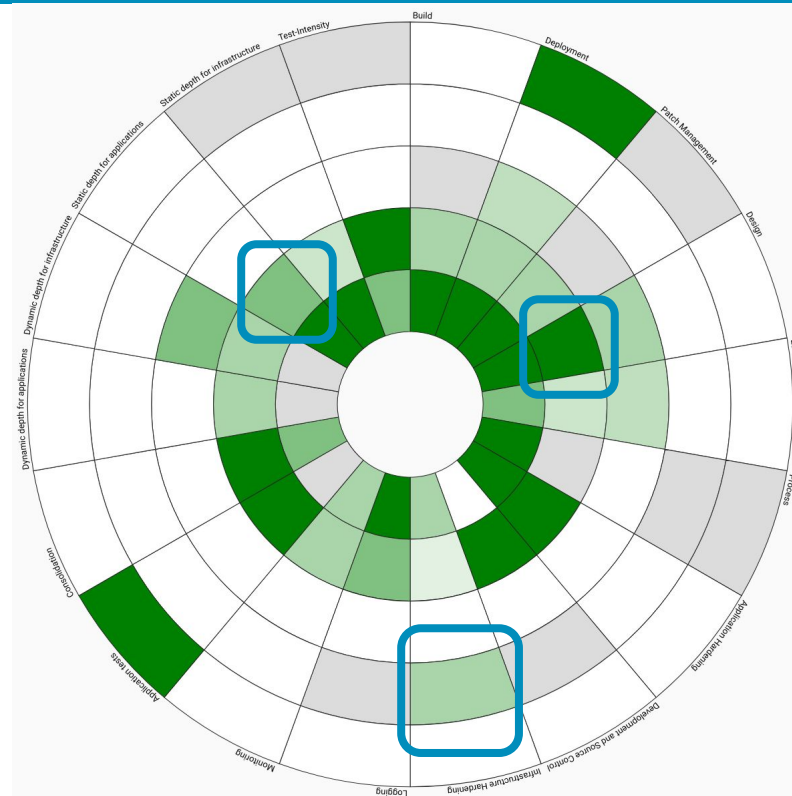


Q3 2023

Areas of Organizations Improvement



Q1 2023



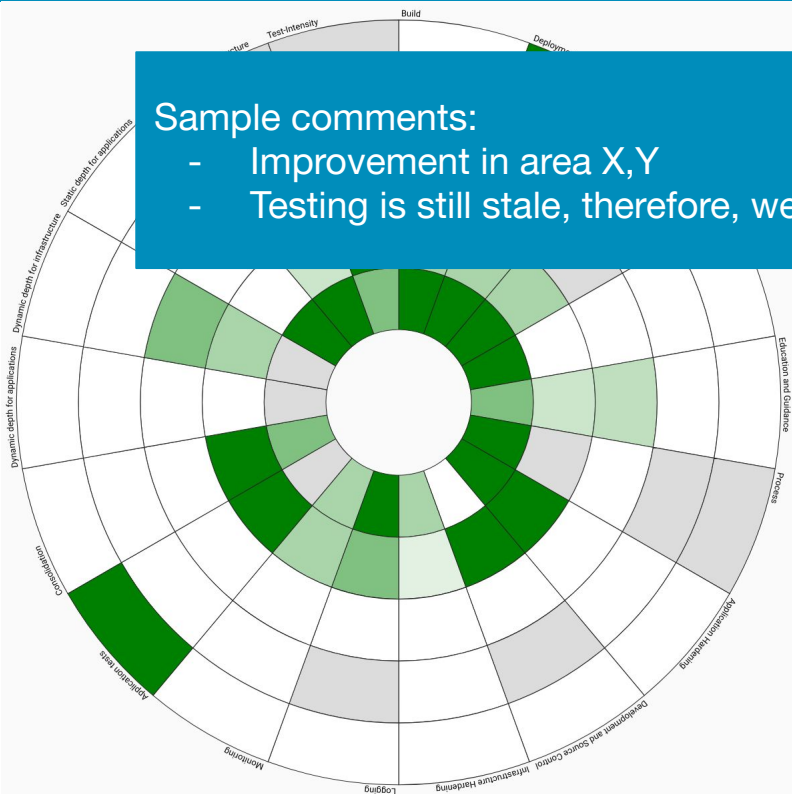
Q3 2023

Areas of Organizations Improvement

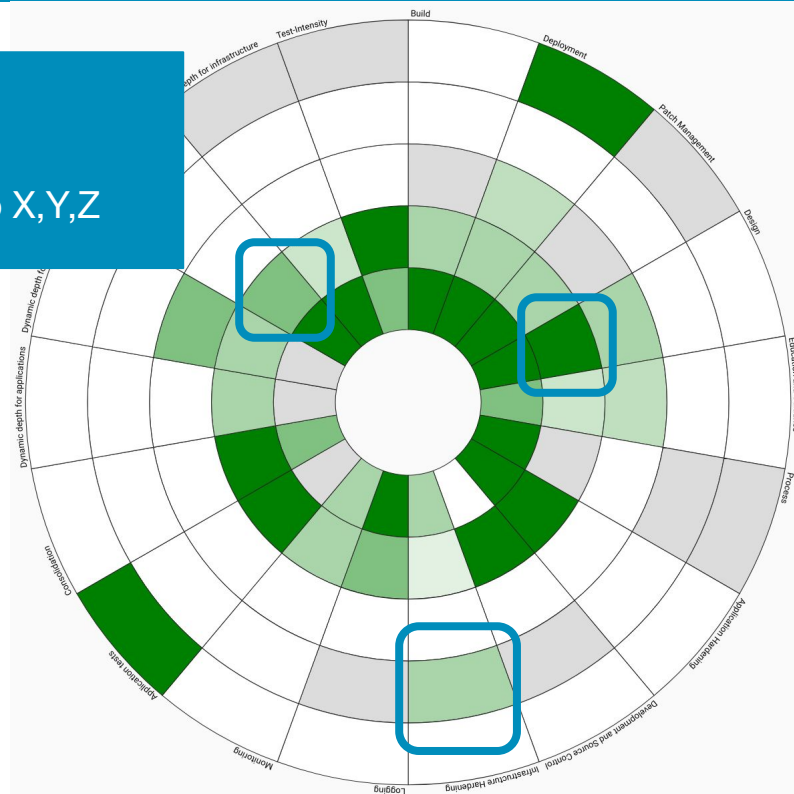


Sample comments:

- Improvement in area X,Y
- Testing is still stale, therefore, we need to do X,Y,Z



Q1 2023



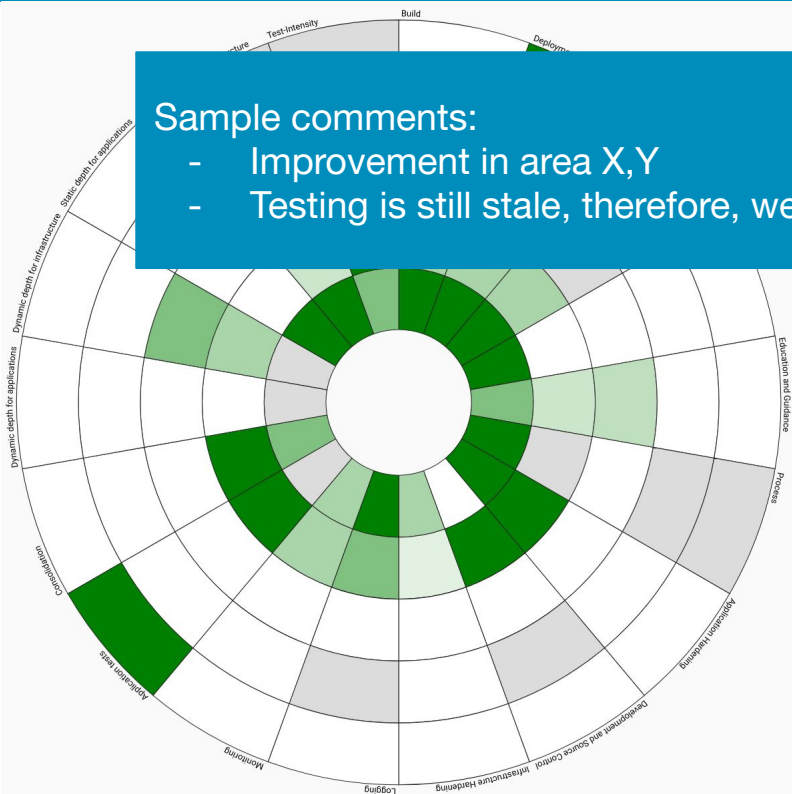
Q3 2023

Areas of Depart X Improvement

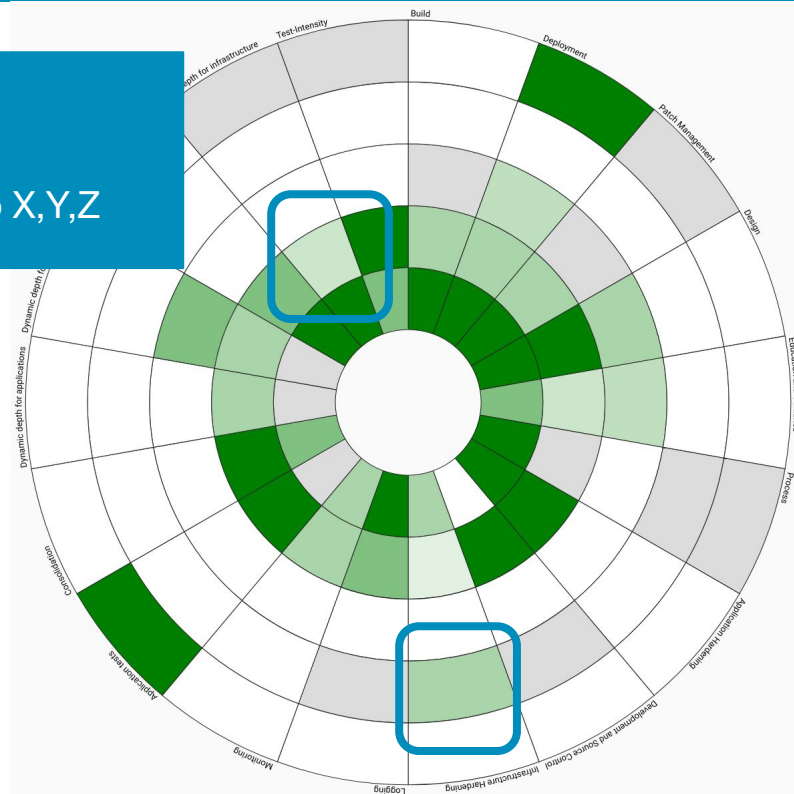


Sample comments:

- Improvement in area X,Y
- Testing is still stale, therefore, we need to do X,Y,Z



Q1 2023

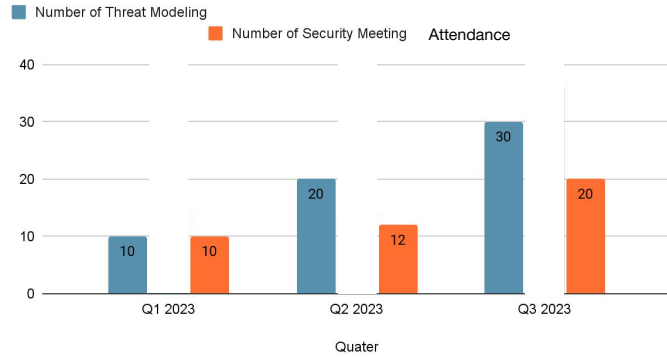


Q3 2023

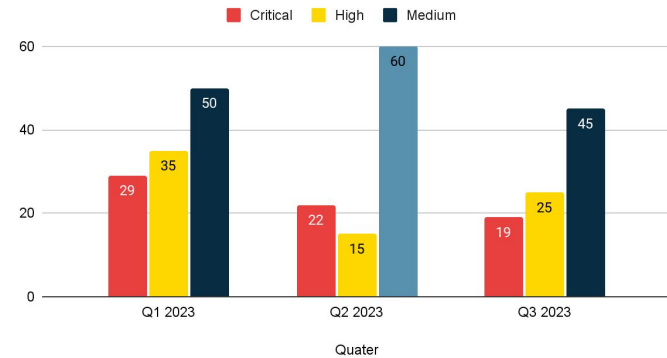
Team Frodo Dashboard



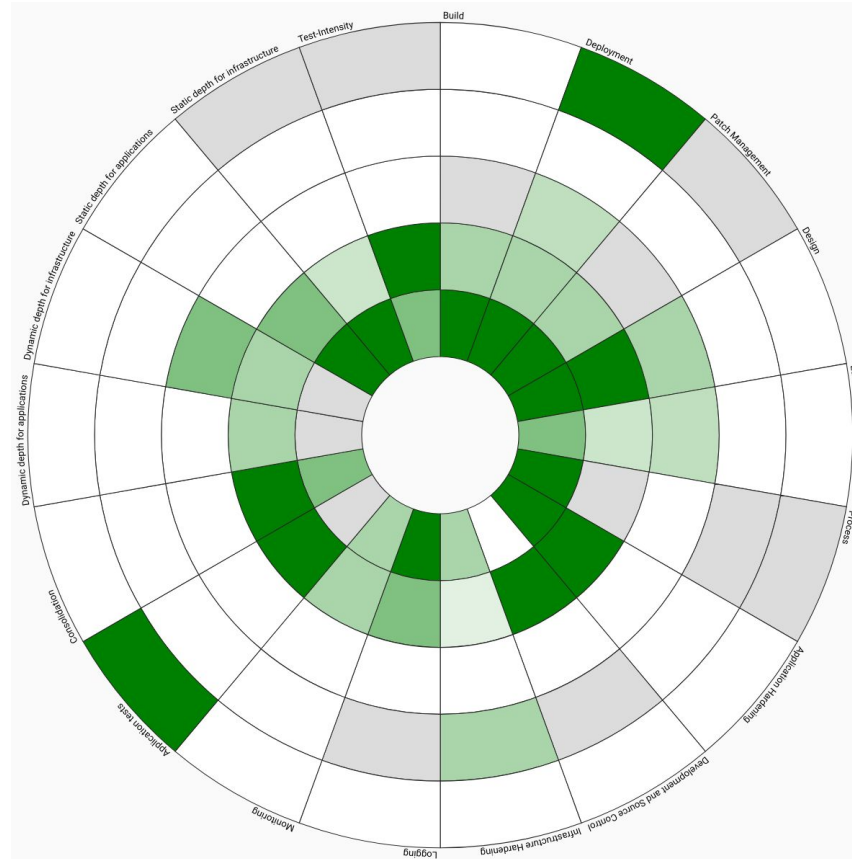
Important Activities



Mean Time To Resolve Vulnerabilities (Days)

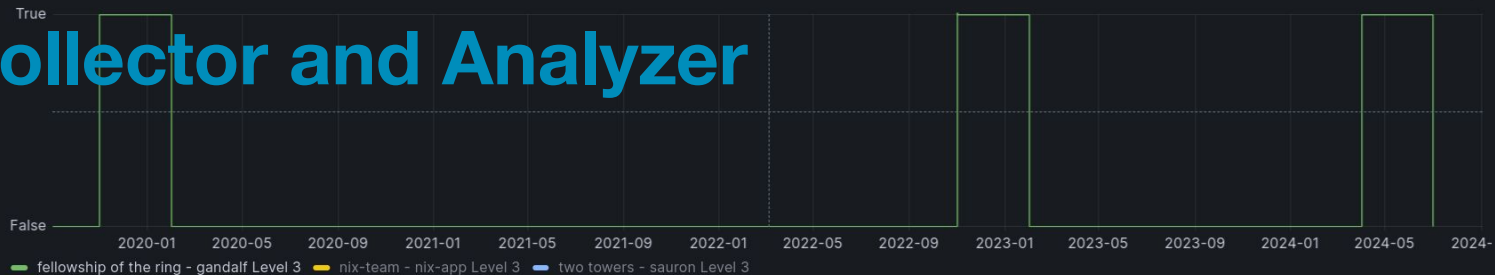


Overview Current Status Team Frodo



Metric Collector and Analyzer

Conduction of simple threat modeling on technical level Level 3 ⓘ



Conduction of simple threat modeling on technical level Level 2 ⓘ



Conduction of simple threat modeling on technical level Level 1 ⓘ



Mean Time to Patch (MTTP)



- Amount of time automatically created PRs are open per team
- Amount of up to date systems vs amount of systems
- Image Lifetime
- BaseImage Lifetime

Awareness Level



- Number of teams vs. number of teams with a security training in a time period

Production Repository Score Factor Breakdown



Source

<https://medium.com/life-at-chime/monocle-how-chime-creates-a-proactive-security-engineering-culture-part-1-dedd3846127f>



Scorecards / Secret Scanning 📄 <>

Scorecard Owner: @github/product-security-engineering · Slack: #pse-eng · Fundamental: Yes

[🏠 Overview](#) [📈 Trends](#) [📊 Scores](#) [🚫 Unmet Requirements](#) [± Score Changes](#) [📊 Charts](#)

Ensures that services do not have Secret Scanning alerts for checked in secrets.

For feedback or requests, please open an issue in the [github/product-security-engineering](#) repository.

Select filter ▾

🔍 Filter services

Search syntax tips [🔗](#) [✖](#)

Save new filter

Objectives

Risks

The service has fixed all outstanding risks or is in the target fix window.

This service has no secrets checked in to the repository. 📄

85 points

Requirement met by 1922 of 1922 services (100%)

This service has push protection turned on in the repository. 📄

10 points

Requirement met by 1922 of 1922 services (100%)

This service has secret scanning turned on in the repository. 📄

5 points

Requirement met by 1922 of 1922 services (100%)

Source:

<https://github.blog/2024-02-08-githubs-engineering-fundamentals-program-how-we-deliver-on-availability-security-and-accessibility/> PageShield

AppSec Metrics Dashboard – Executive View

Service Analyzer ▾ Notable Events Review Glass Tables Deep Dives Multi KPI Alerts Search ▾ Configure ▾ Product Tour

IT Service Intelligence

App Sec ↗

Now ▾ Edit ↗

E-Com



E-com

85 →
0%

Coverage

60 →
0%

Met Sec. Req.

4.591

Open Vulnerabil-
ities

3

Secure Code

84.8
days

MTTR

0 →
0%

WAF

46.36

3rd Party

Pulse



Pulse

75 →
0%

Coverage

0 →
0%

Met Sec. Req.

35

Open Vulnerabil-
ities

4

Secure Code

41.23
days

MTTR

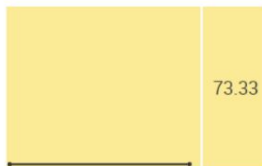
0
ct

WAF

7

3rd Party

NGP



73.33 ↗

Coverage

+
-

90%

Source

<https://www.iotechexpo.com/northamerica/wp-content/uploads/2018/12/PodHandler.pdf>

Silverbullet?

People and Processes



- Still need to perform activities
- For manual: Need to update YAMLs and understand why

Sec. Dashboard Refs.



<https://medium.com/life-at-chime/monocle-how-chime-creates-a-proactive-security-engineering-culture-part-1-dedd3846127f>

<https://www.youtube.com/watch?v=e6k7DpXTtWA>

<https://github.blog/2024-02-08-githubs-engineering-fundamentals-program-how-we-deliver-on-availability-security-and-accessibility/>

<https://www.iottechexpo.com/northamerica/wp-content/uploads/2018/12/PodHandler.pdf>

<https://tldrsec.com/p/blog-insecure-development-why-some-product-teams-are-great-and-others-arent>

<https://medium.com/uber-security-privacy/uber-bug-bounty-promotions-1ed7648cf6b0>

<https://tldrsec.com/p/appsec-a-pragmatic-approach-for-internal-security-partnerships>

MetricCA Ref



Documentation:

<https://github.com/devsecopsmaturitymodel/metricCA>

Main App:

<https://github.com/devsecopsmaturitymodel/metricAnalyzer>

Collectors:

<https://github.com/devsecopsmaturitymodel/collector-confluence>

Agenda



Solid Foundation



Analyze Security Practices (Quick)



Plan Security Activities



Implement And Measurement Improvements



Summary

Summary



- Teams need fast feedback: Automate assessments and metrics
- Planning of activities, communication of the plan, and execution is a key step

Contribution Opportunities



Open PRs

- Add features to the DSOMM application
- Add activities to DSOMM data
- Fix typos in DSOMM data

Sponsor

<https://owasp.org/donate/?reponame=www-project-devsecops-maturity-model&title=OWASP+Devsecops+Maturity+Model>



Timo Pagel

Contact: timo.pagel@owasp.org

Business Contact: dsomm@pagel.pro

Business Website AppSec: <https://appsec-program.com/>

Business Website: <https://pagel.pro>

pagel.pro QR



Join our community in the [OWASP Slack](#) in channel #dsomm

Images



Icons bought via <https://thenounproject.com/> or
copyright by PageShield GmbH