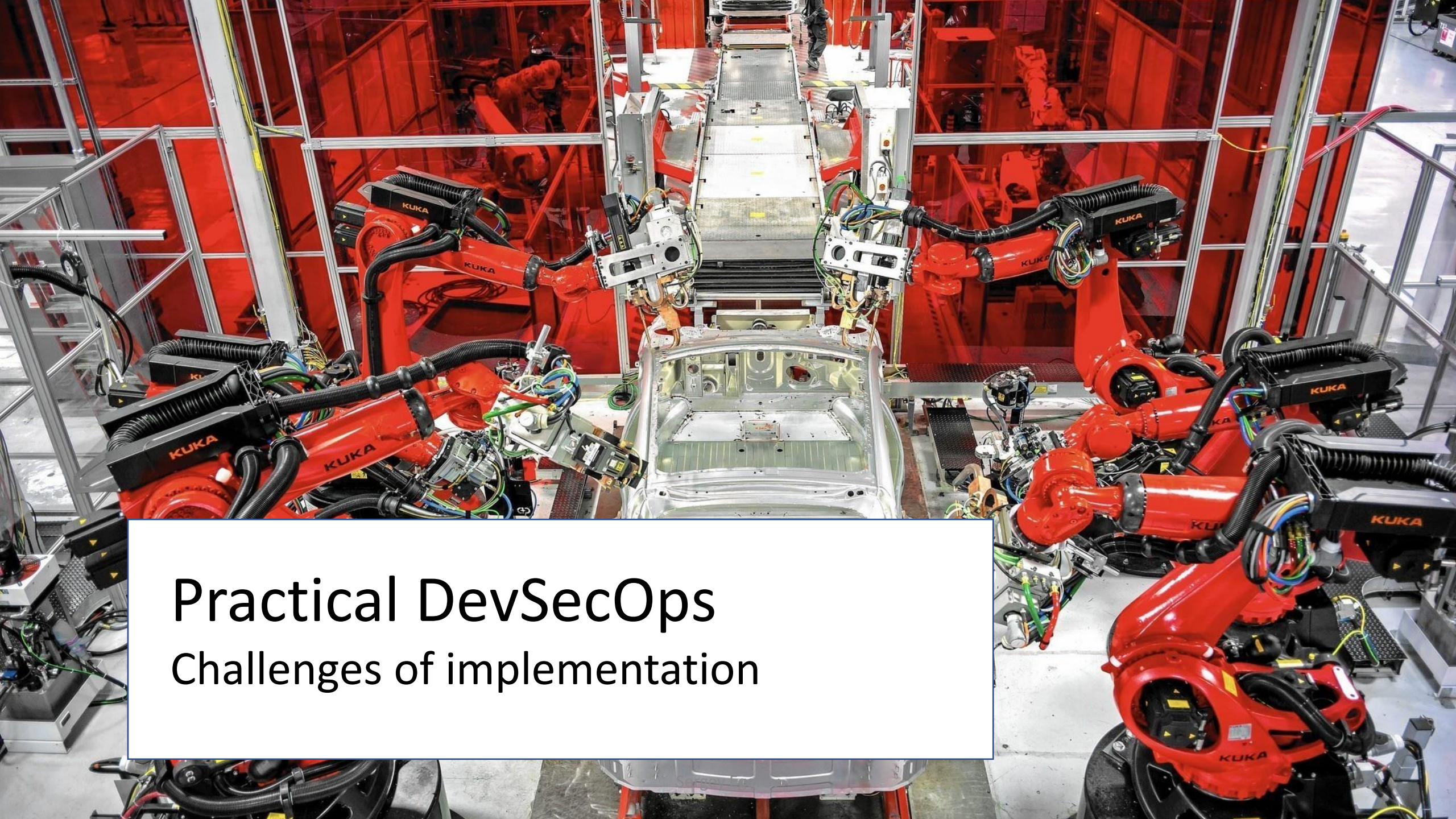
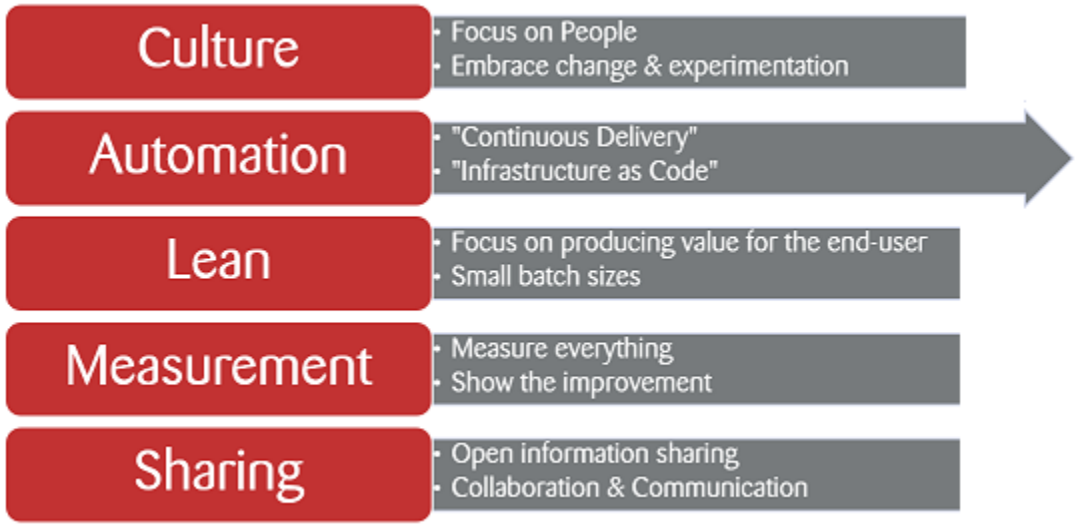
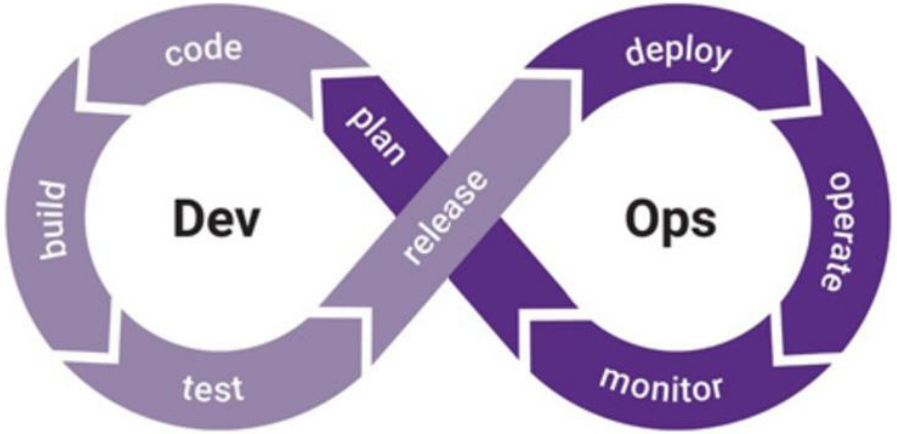



# Practical DevSecOps

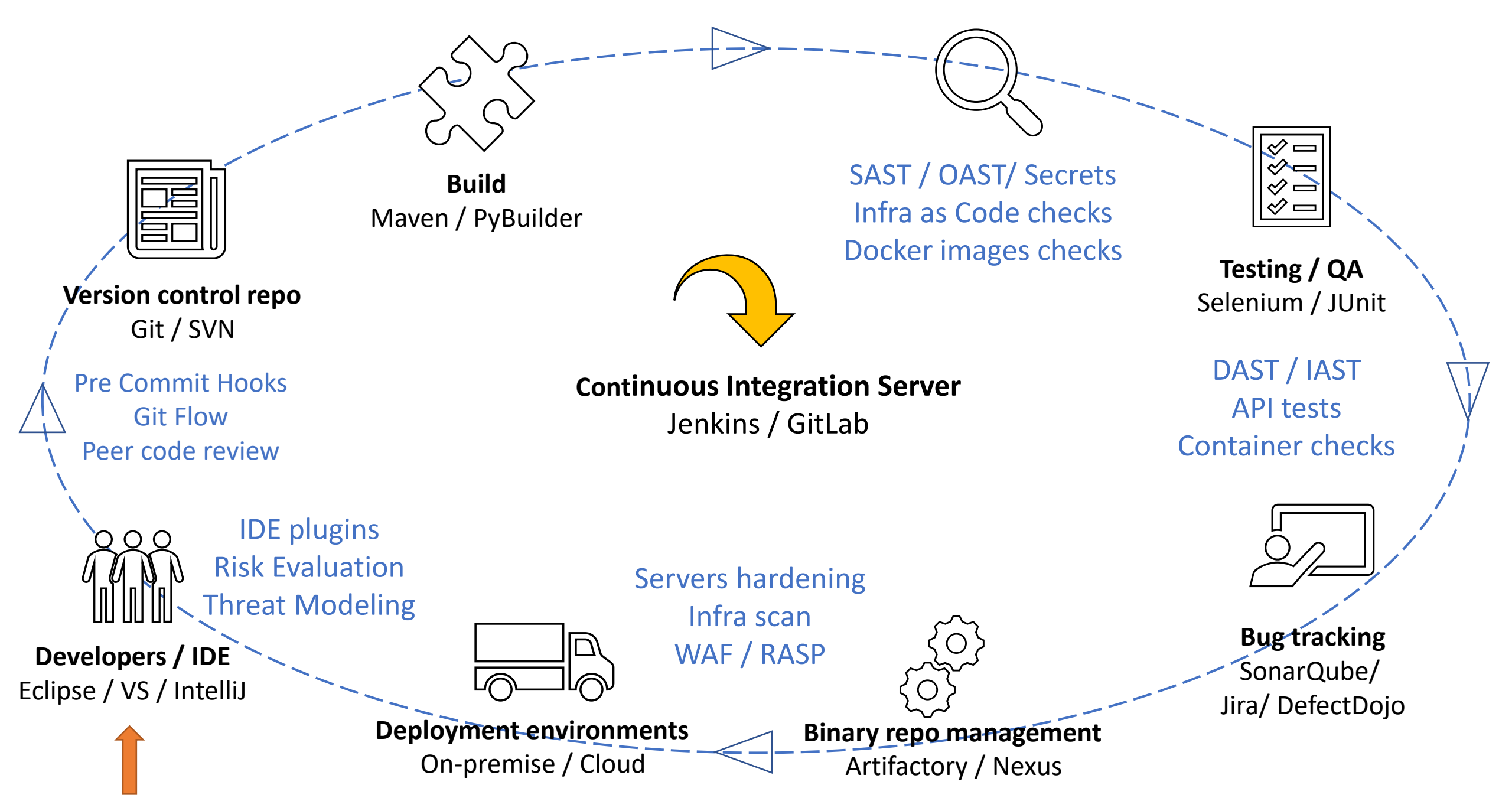
## Challenges of implementation



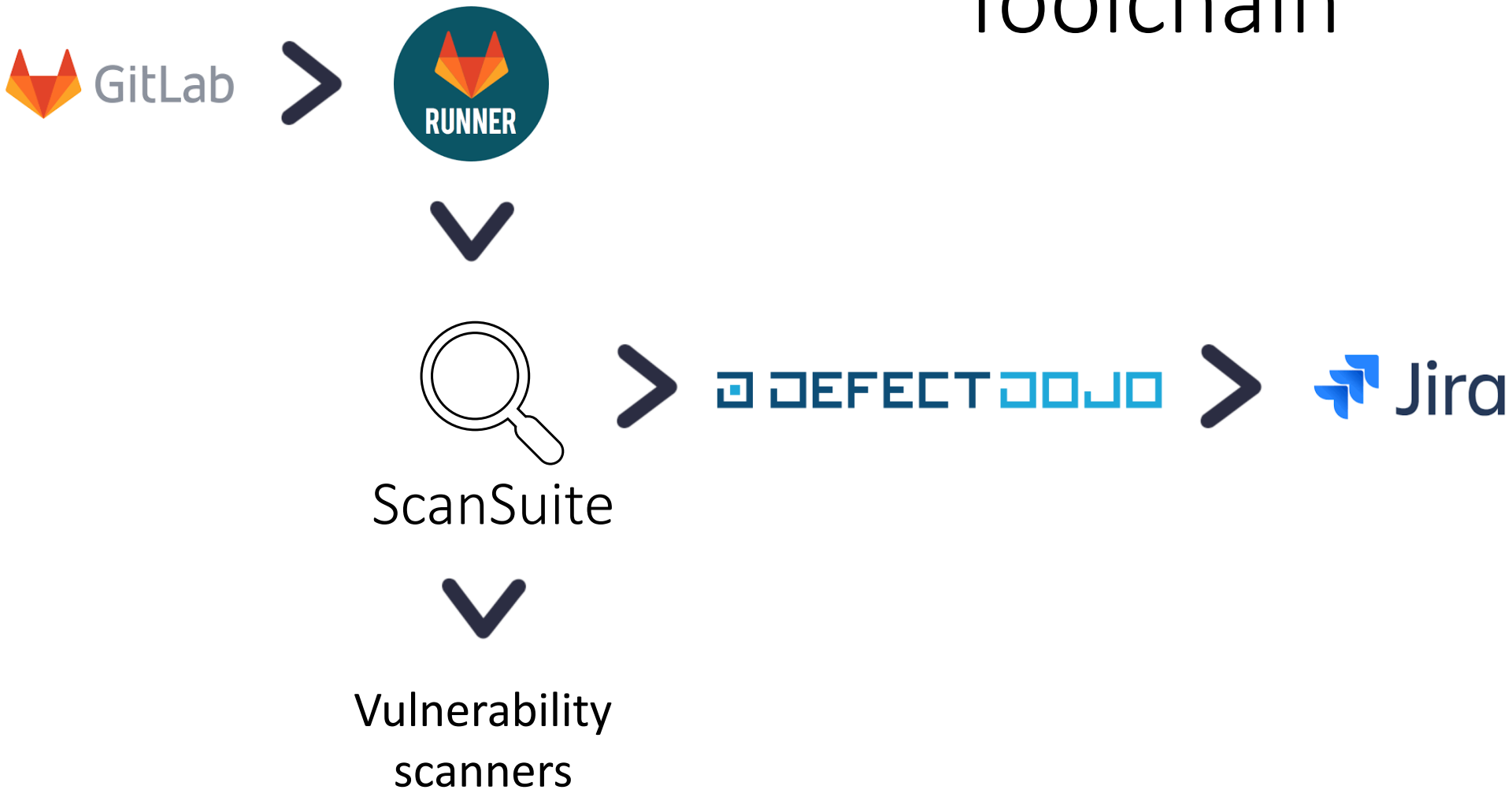




# Scope for today

- Evaluation of vulnerabilities and misconfigurations through the development lifecycle
- Scanning tools, such as:
  - Static Application Security Testing (SAST)
  - Open Source Component Analysis (SCA)
  - Infrastructure as Code Scanning (IACS)
  - Containers Security Checks
  - Dynamic Application Security Testing (DAST)
- Vulnerability management process and tools (DefectDojo)
- CI/CD (Gitlab) and Bug Tracking (Jira) integrations


Build
Maven / PyBuilder
SAST / OAST/ Secrets
Infra as Code checks
Docker images checks
Testing / QA
Selenium / JUnit
DAST / IAST
API tests
Container checks
Bug tracking
SonarQube/
Jira/ DefectDojo
Binary repo management
Artifactory / Nexus
Deployment environments
On-premise / Cloud
Servers hardening
Infra scan
WAF / RASP
Developers / IDE
Eclipse / VS / IntelliJ
IDE plugins
Risk Evaluation
Threat Modeling
Version control repo
Git / SVN
Pre Commit Hooks
Git Flow
Peer code review
Continuous Integration Server
Jenkins / GitLab

# Toolchain


GitLab
RUNNER
ScanSuite
DEFECTDOJO
Jira
Vulnerability scanners

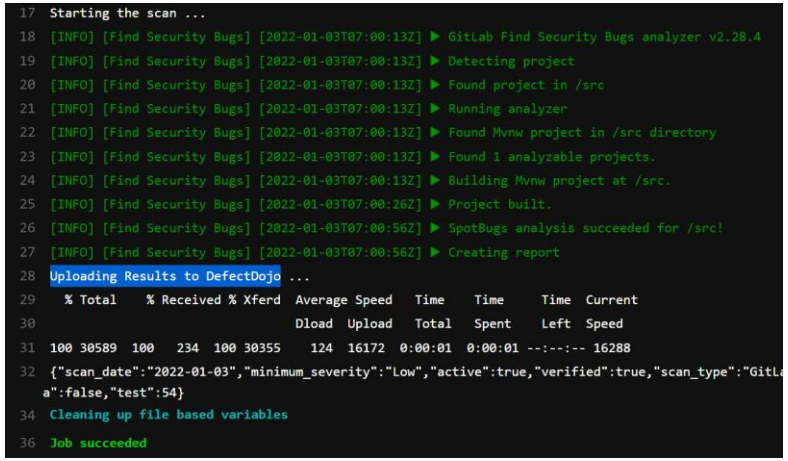
# ScanSuite

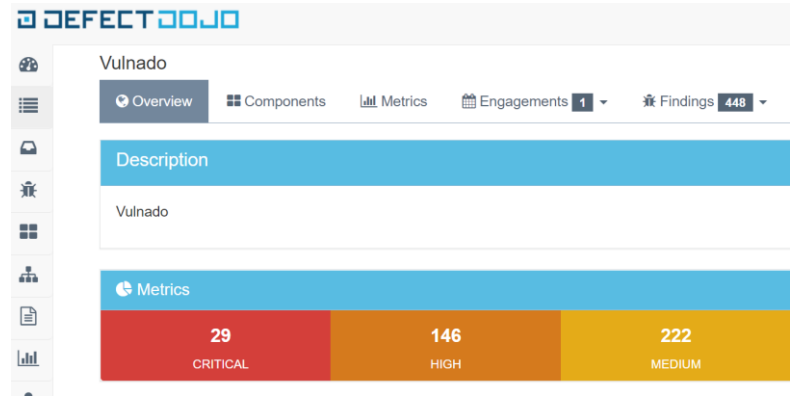

- One tool to glue them all
- Available on github.com/cepxeo/scansuite
- Bash wrapper around over 20 scanners, such as:

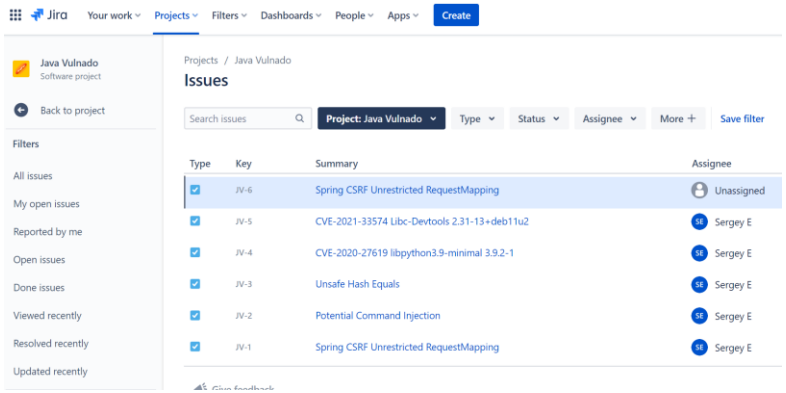
  SAST / DAST / SCA / IACS / Container checks
- Leverages GitLab images and

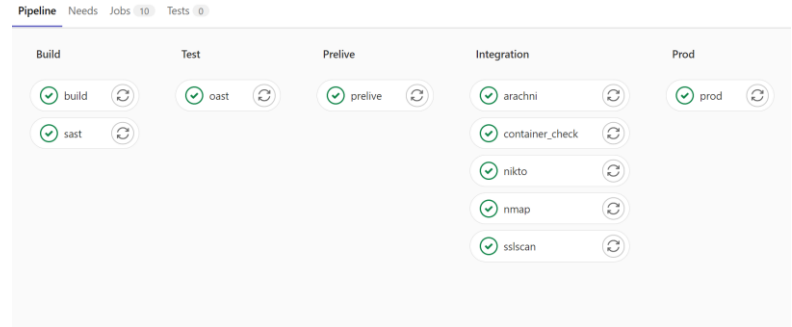
other open source scanners
- For standalone checks or as a part of CI\CD
- Exports results to DefectDojo

# Demo time


Starting the scan ...
[INFO] [Find Security Bugs] [2022-01-03T07:00:13Z] ▶ GitLab Find Security Bugs analyzer v2.28.4
[INFO] [Find Security Bugs] [2022-01-03T07:00:13Z] ▶ Detecting project
[INFO] [Find Security Bugs] [2022-01-03T07:00:13Z] ▶ Found project in /src
[INFO] [Find Security Bugs] [2022-01-03T07:00:13Z] ▶ Running analyzer
[INFO] [Find Security Bugs] [2022-01-03T07:00:13Z] ▶ Found Mvnw project in /src directory
[INFO] [Find Security Bugs] [2022-01-03T07:00:13Z] ▶ Found 1 analyzable projects.
[INFO] [Find Security Bugs] [2022-01-03T07:00:13Z] ▶ Building Mvnw project at /src.
[INFO] [Find Security Bugs] [2022-01-03T07:00:26Z] ▶ Project built.
[INFO] [Find Security Bugs] [2022-01-03T07:00:56Z] ▶ SpotBugs analysis succeeded for /src!
[INFO] [Find Security Bugs] [2022-01-03T07:00:56Z] ▶ Creating report
Uploading Results to DefectDojo ...
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 30589 100 234 100 30355 124 16172 0:00:01 0:00:01 --:--:-- 16288
{"scan_date":"2022-01-03","minimum_severity":"Low","active":true,"verified":true,"scan_type":"GitLa
a":false,"test":54}
Cleaning up file based variables
Job succeeded


DEFECTDOJO
Vulnado
Overview
Components
Metrics
Engagements 1
Findings 448
Description
Vulnado
Metrics
29
CRITICAL
146
HIGH
222
MEDIUM


Jira
Your work
Projects
Filters
Dashboards
People
Apps
Create
Java Vulnado
Software project
Back to project
Filters
All issues
My open issues
Reported by me
Open issues
Done issues
Viewed recently
Resolved recently
Updated recently
Projects / Java Vulnado
Issues
Search issues
Project: Java Vulnado
Type
Status
Assignee
More
Save filter
Type Key Summary Assignee
JV-6 Spring CSRF Unrestricted RequestMapping Unassigned
JV-5 CVE-2021-33574 Libc-Devtools 2.31-13+deb11u2 Sergey E
JV-4 CVE-2020-27619 libpython3.9-minimal 3.9.2-1 Sergey E
JV-3 Unsafe Hash Equals Sergey E
JV-2 Potential Command Injection Sergey E
JV-1 Spring CSRF Unrestricted RequestMapping Sergey E


Pipeline
Needs
Jobs 10
Tests 0
Build
Test
Prelive
Integration
Prod
build
sast
oast
prelive
arachni
container_check
nikto
nmap
sslscan
prod

**SAST**
Static AppSec Testing

**DAST**
Dynamic AppSec Testing

## Constraints & Limitations

- False positives
- Time to scan
- Adjusting

**IAST**
Interactive AppSec Testing

- Quality depends on unit tests
- Limited language support
- Unaware of business logic

- Web apps only
- App crawlers
- Intrusive vs not comprehensive
- Business logic vulnerabilities

# Considerations

Start low, go slow, but look far
Single vulnerability aggregation repository
Track issues, measure remediation rate
Filtering for duplications and false positives
Continuous improvement of scanners and tools
Business logic checks via regular unit testing
Application security minds and skills