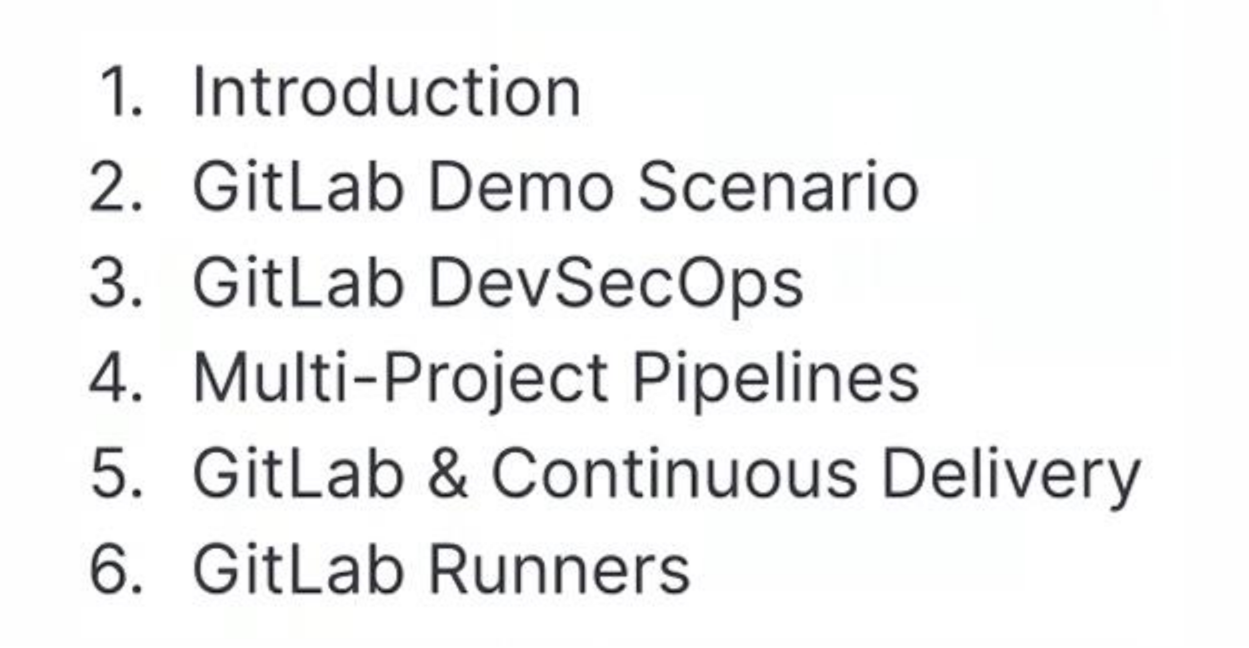
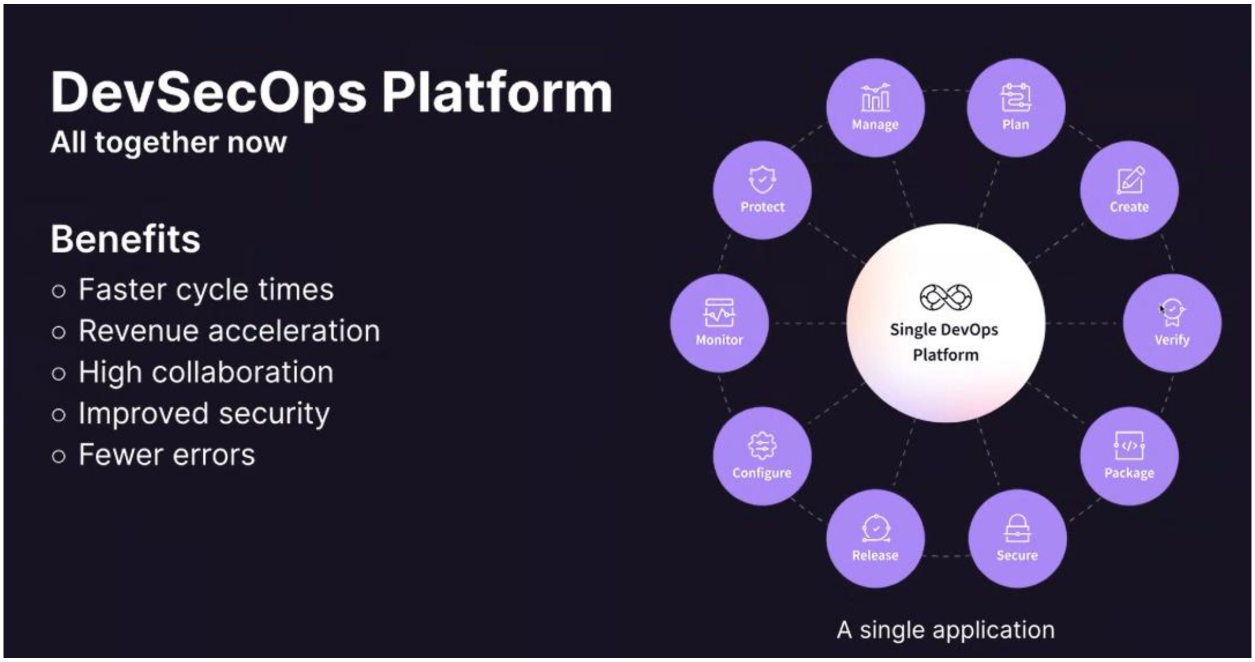


1. Introduction
2. GitLab Demo Scenario
3. GitLab DevSecOps
4. Multi-Project Pipelines
5. GitLab & Continuous Delivery
6. GitLab Runners



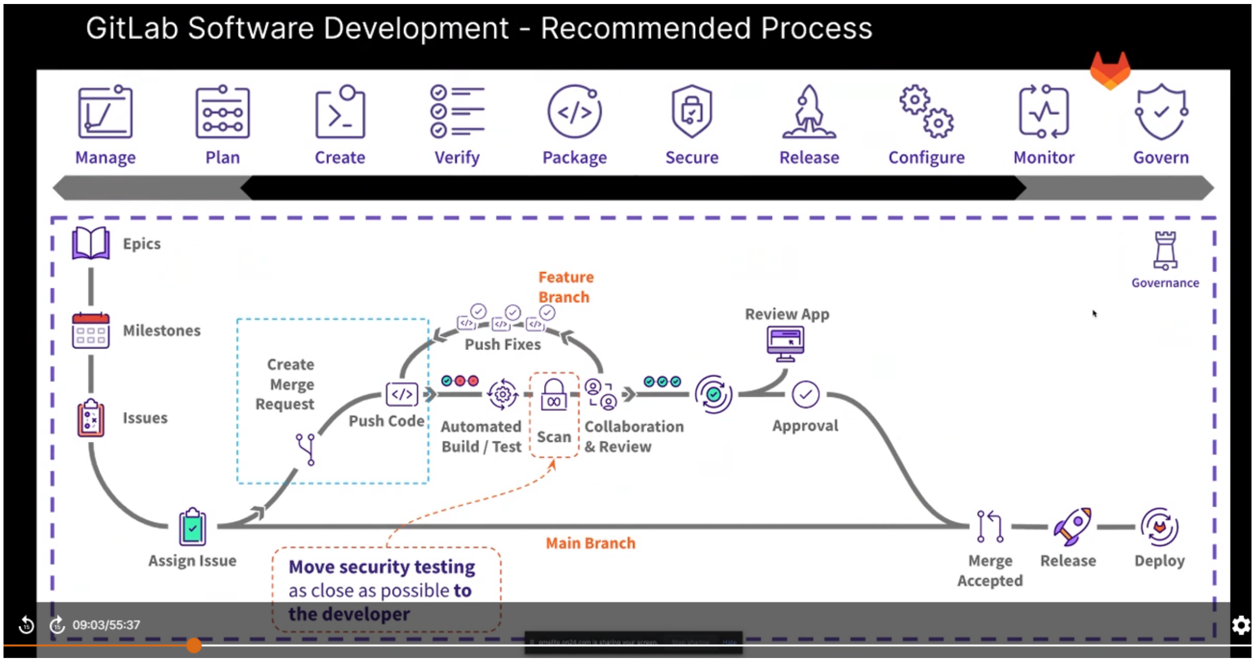
DevSecOps Platform

All together now

Benefits

- Faster cycle times
- Revenue acceleration
- High collaboration
- Improved security
- Fewer errors

A single application



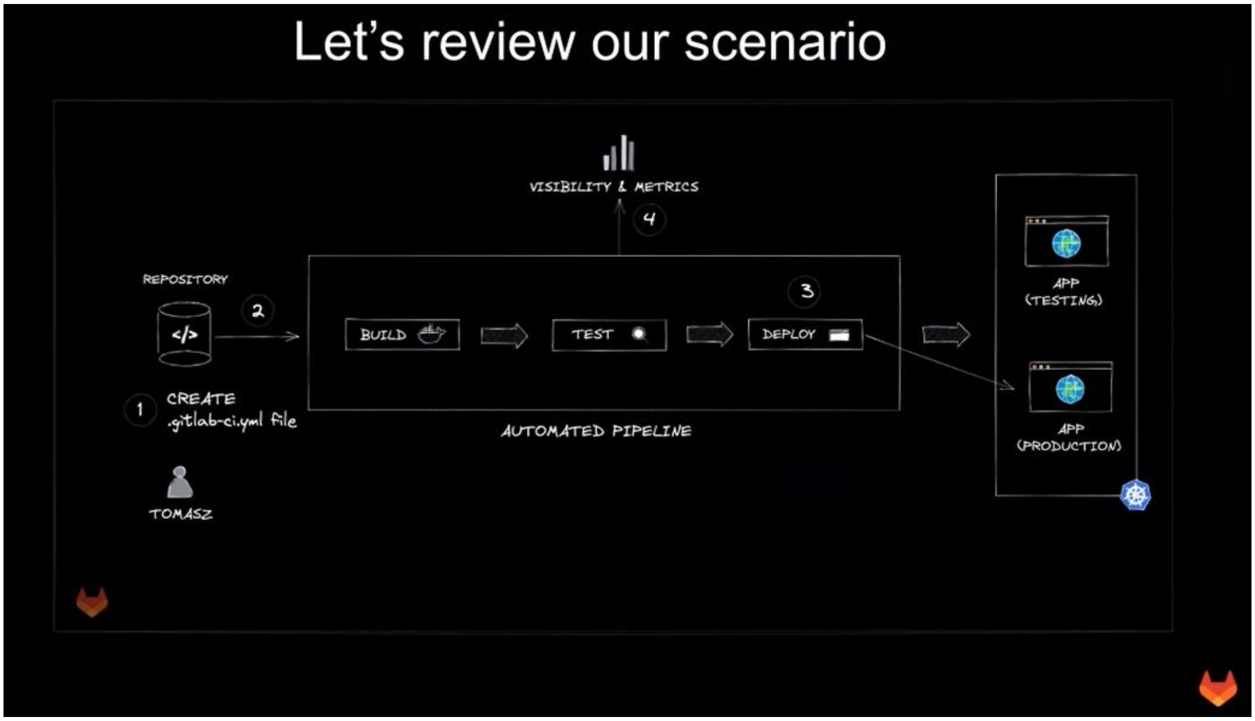
GitLab Software Development - Recommended Process

Manage
Plan
Create
Verify
Package
Secure
Release
Configure
Monitor
Govern

Epics
Milestones
Issues
Assign Issue
Create Merge Request
Push Code
Feature Branch
Push Fixes
Automated Build / Test
Scan
Collaboration & Review
Review App
Approval
Governance
Main Branch
Merge Accepted
Release
Deploy

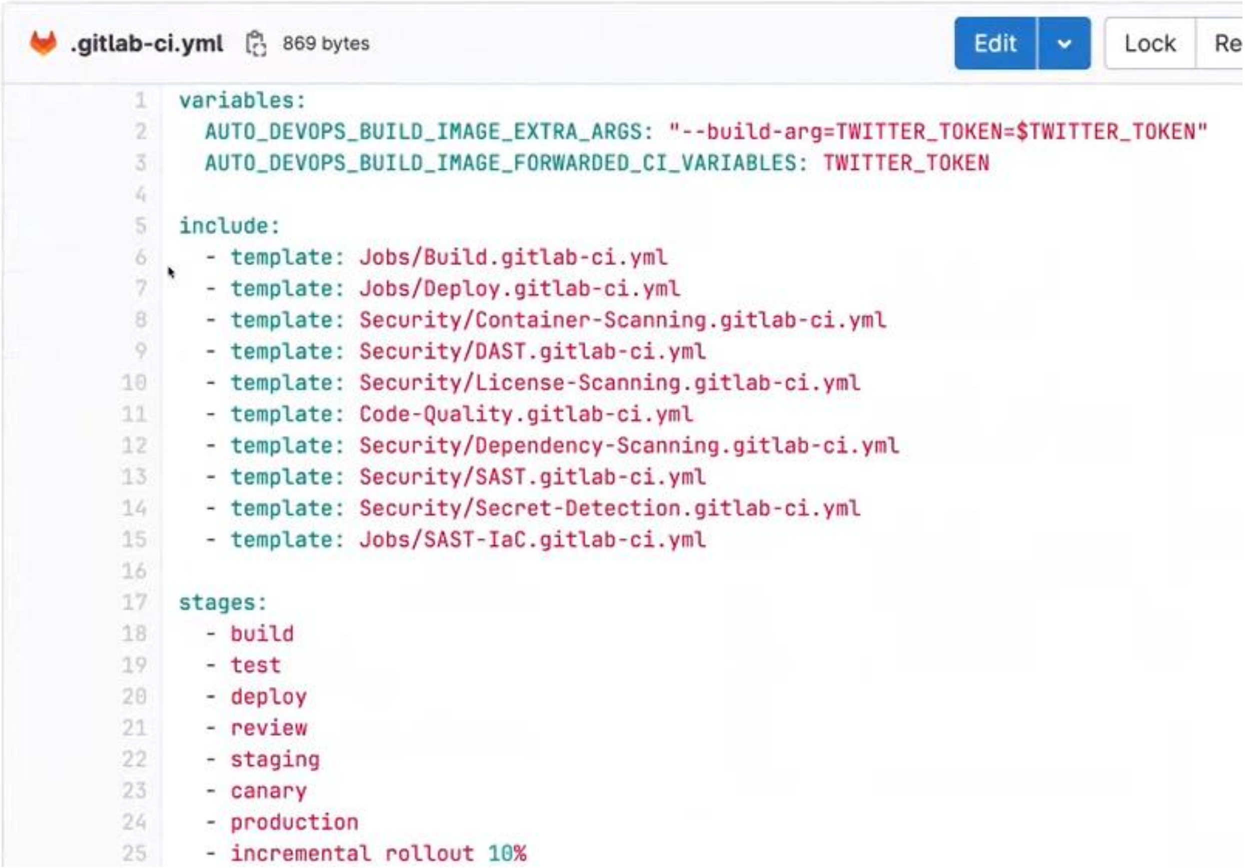
Move security testing as close as possible **to the developer**

09:03/55:37



Let's review our scenario

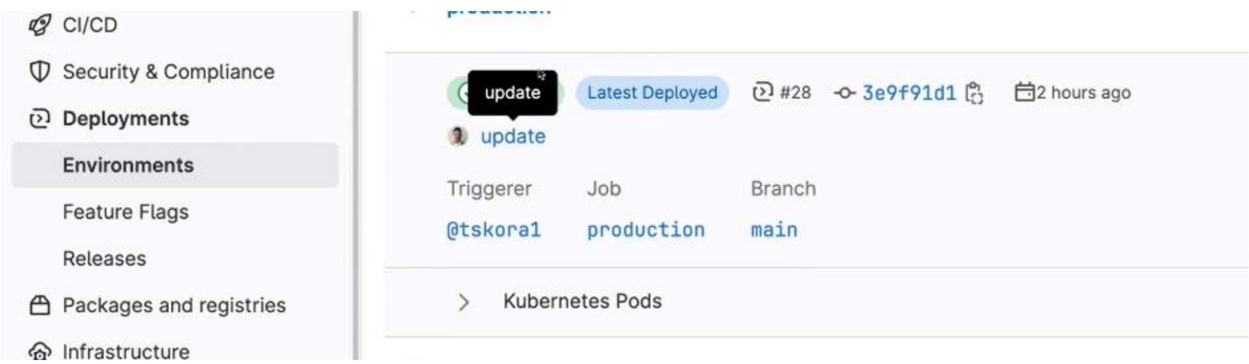
VISIBILITY & METRICS
4
REPOSITORY
2
1 CREATE .gitlab-ci.yml file
TOMASZ
3
BUILD
TEST
DEPLOY
AUTOMATED PIPELINE
APP (TESTING)
APP (PRODUCTION)



```yaml
variables:
  AUTO_DEVOPS_BUILD_IMAGE_EXTRA_ARGS: "--build-arg=TWITTER_TOKEN=$TWITTER_TOKEN"
  AUTO_DEVOPS_BUILD_IMAGE_FORWARDED_CI_VARIABLES: TWITTER_TOKEN

include:
  - template: Jobs/Build.gitlab-ci.yml
  - template: Jobs/Deploy.gitlab-ci.yml
  - template: Security/Container-Scanning.gitlab-ci.yml
  - template: Security/DAST.gitlab-ci.yml
  - template: Security/License-Scanning.gitlab-ci.yml
  - template: Code-Quality.gitlab-ci.yml
  - template: Security/Dependency-Scanning.gitlab-ci.yml
  - template: Security/SAST.gitlab-ci.yml
  - template: Security/Secret-Detection.gitlab-ci.yml
  - template: Jobs/SAST-IaC.gitlab-ci.yml

stages:
  - build
  - test
  - deploy
  - review
  - staging
  - canary
  - production
  - incremental rollout 10%
```

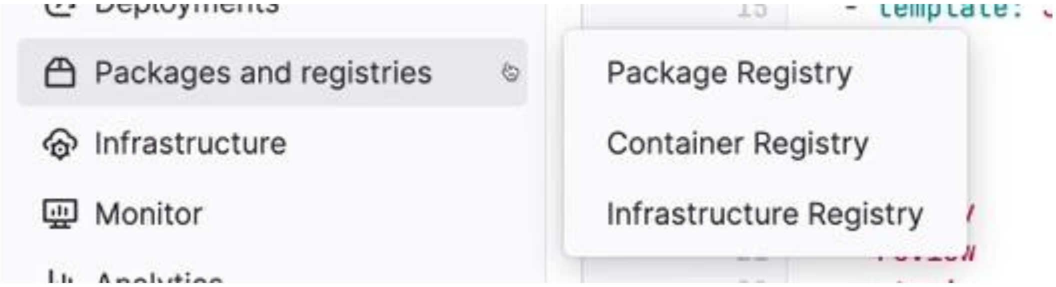


#28 3e9f91d1 2 hours ago

update

Triggerer	Job	Branch
@tskora1	production	main

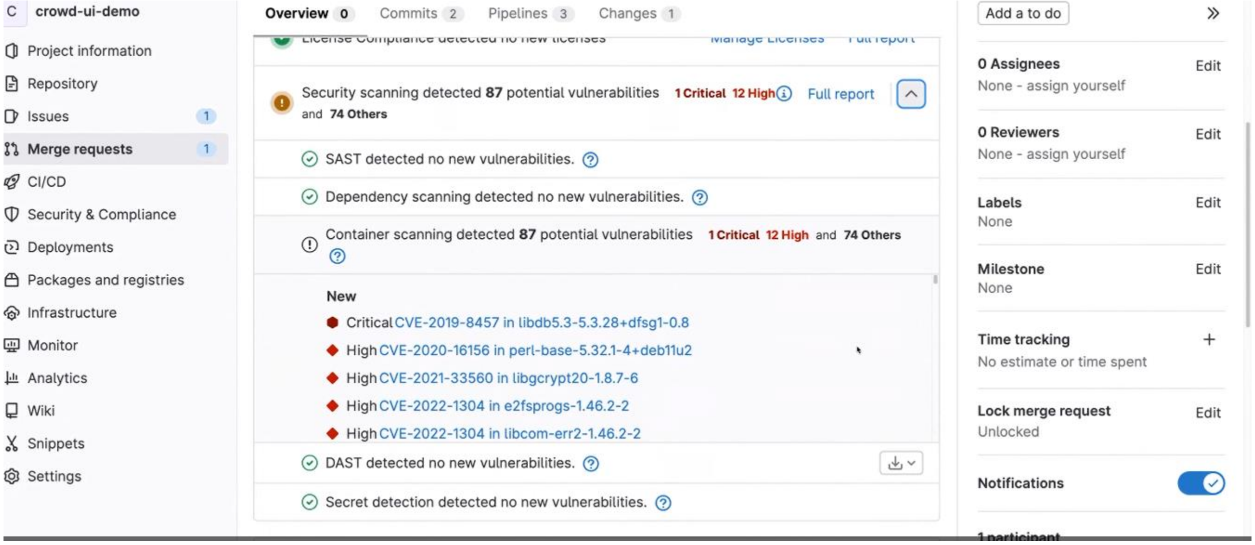
Gitlabci.yml – steps how our application will be deployed .



We can store artifact in container registry .

Kubernetes cluster can also be integrated with GitLab.

We can also view live application that will be running inside Kubernetes cluster .



License Compliance detected no new licenses

Security scanning detected **87** potential vulnerabilities 1 Critical 12 High and 74 Others Full report

SAST detected no new vulnerabilities.

Dependency scanning detected no new vulnerabilities.

Container scanning detected **87** potential vulnerabilities 1 Critical 12 High and 74 Others

New

- Critical CVE-2019-8457 in libdb5.3-5.3.28+dfsg1-0.8
- High CVE-2020-16156 in perl-base-5.32.1-4+deb11u2
- High CVE-2021-33560 in libgcrypt20-1.8.7-6
- High CVE-2022-1304 in e2fsprogs-1.46.2-2
- High CVE-2022-1304 in libcom-err2-1.46.2-2

DAST detected no new vulnerabilities.

Secret detection detected no new vulnerabilities.

0 Assignees Edit
None - assign yourself

0 Reviewers Edit
None - assign yourself

Labels Edit
None

Milestone Edit
None

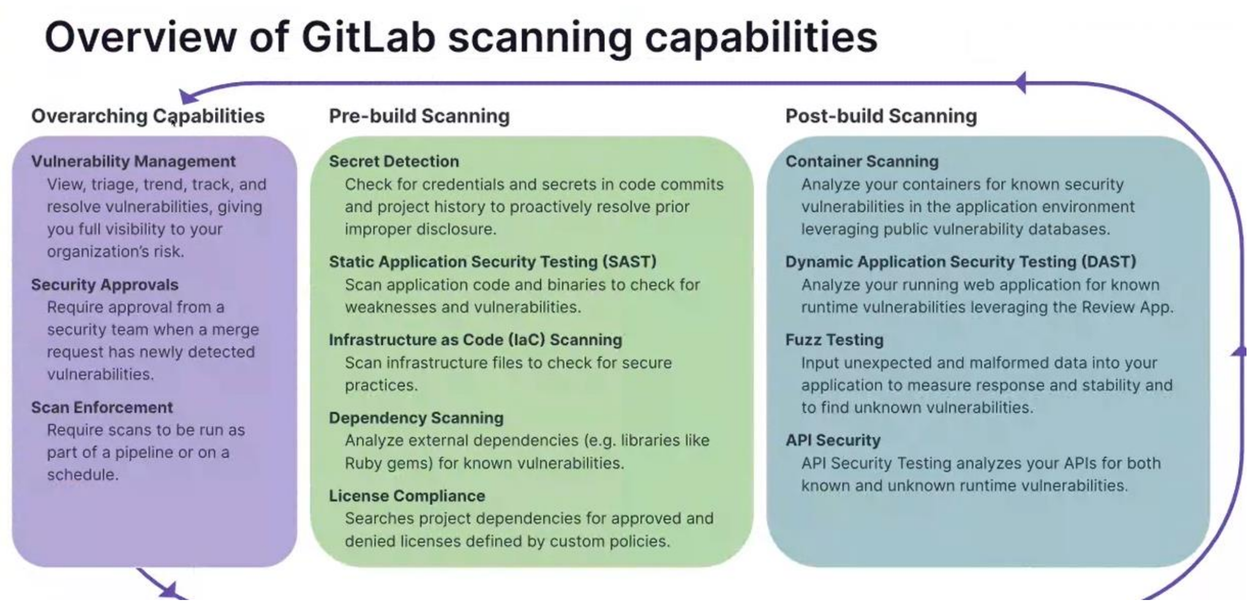
Time tracking +
No estimate or time spent

Lock merge request Edit
Unlocked

Notifications

1 participant

We can also check vulnerabilities present in code in gitlab .



Overview of GitLab scanning capabilities

Overarching Capabilities

Vulnerability Management
View, triage, trend, track, and resolve vulnerabilities, giving you full visibility to your organization's risk.

Security Approvals
Require approval from a security team when a merge request has newly detected vulnerabilities.

Scan Enforcement
Require scans to be run as part of a pipeline or on a schedule.

Pre-build Scanning

Secret Detection
Check for credentials and secrets in code commits and project history to proactively resolve prior improper disclosure.

Static Application Security Testing (SAST)
Scan application code and binaries to check for weaknesses and vulnerabilities.

Infrastructure as Code (IaC) Scanning
Scan infrastructure files to check for secure practices.

Dependency Scanning
Analyze external dependencies (e.g. libraries like Ruby gems) for known vulnerabilities.

License Compliance
Searches project dependencies for approved and denied licenses defined by custom policies.

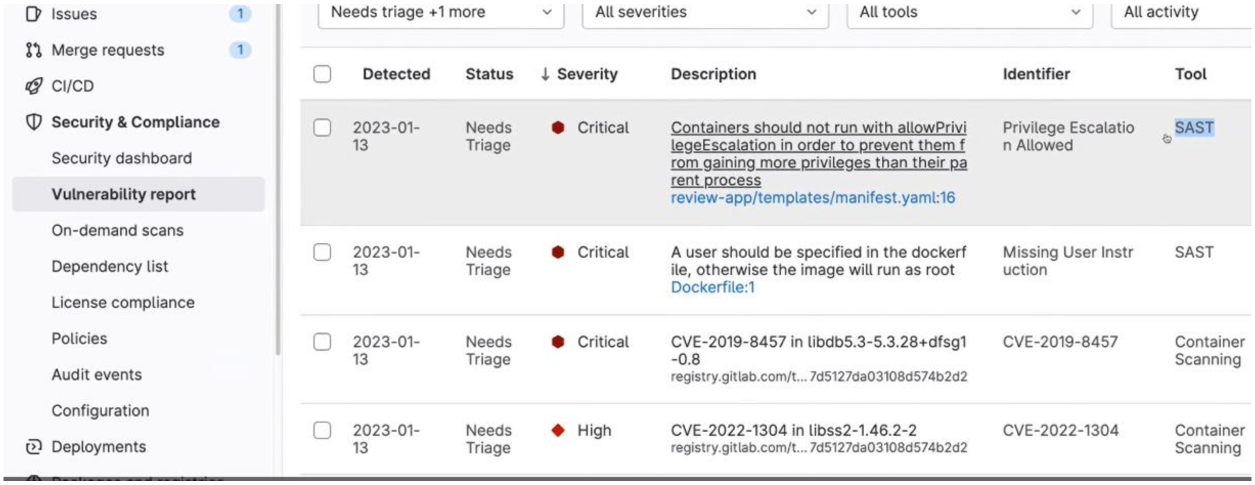
Post-build Scanning

Container Scanning
Analyze your containers for known security vulnerabilities in the application environment leveraging public vulnerability databases.

Dynamic Application Security Testing (DAST)
Analyze your running web application for known runtime vulnerabilities leveraging the Review App.

Fuzz Testing
Input unexpected and malformed data into your application to measure response and stability and to find unknown vulnerabilities.

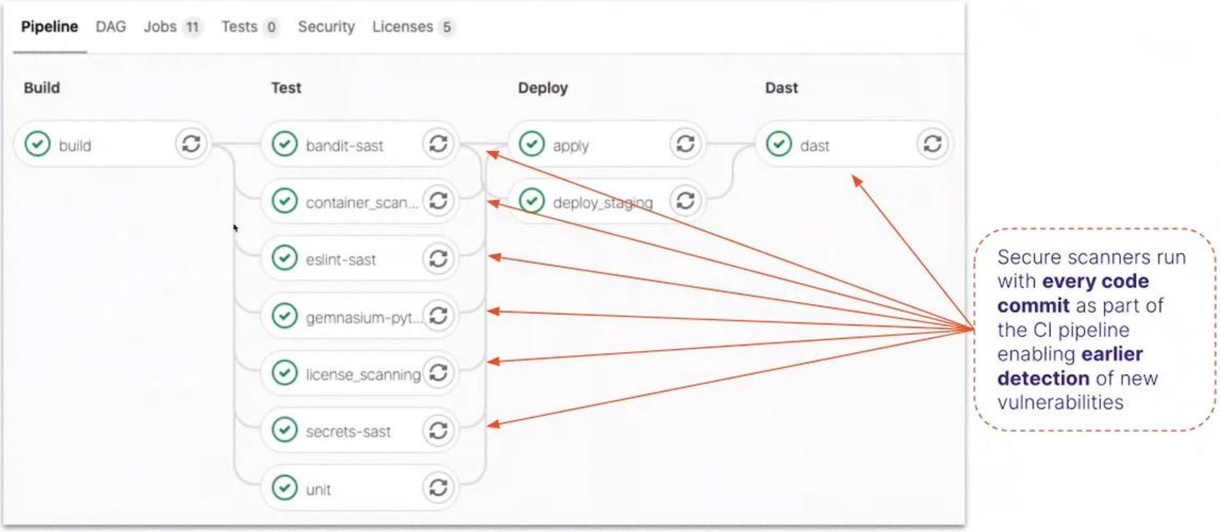
API Security
API Security Testing analyzes your APIs for both known and unknown runtime vulnerabilities.



Needs triage +1 more | All severities | All tools | All activity

□	Detected	Status	↓ Severity	Description	Identifier	Tool
□	2023-01-13	Needs Triage	● Critical	Containers should not run with allowPrivilegeEscalation in order to prevent them from gaining more privileges than their parent process review-app/templates/manifest.yaml:16	Privilege Escalation Allowed	SAST
□	2023-01-13	Needs Triage	● Critical	A user should be specified in the dockerfile, otherwise the image will run as root Dockerfile:1	Missing User Instruction	SAST
□	2023-01-13	Needs Triage	● Critical	CVE-2019-8457 in libdb5.3-5.3.28+dfsg1-0.8 registry.gitlab.com/t... 7d5127da03108d574b2d2	CVE-2019-8457	Container Scanning
□	2023-01-13	Needs Triage	◆ High	CVE-2022-1304 in libss2-1.46.2-2 registry.gitlab.com/t... 7d5127da03108d574b2d2	CVE-2022-1304	Container Scanning

Scans run as part of every CI pipeline



Secure scanners run with **every code commit** as part of the CI pipeline enabling **earlier detection** of new vulnerabilities