



DevSecOps in the AI Era: Security Gates that Scale

SAST • SCA • DAST • CI/CD • AI-assisted Review

About Me



Anowar - Solution Architect & Member of CoE

- Leading **DevSecOps** & **multi-functional** teams
- Building teams & **architectures that scale**
- Driving **AI-assisted engineering** across large product orgs



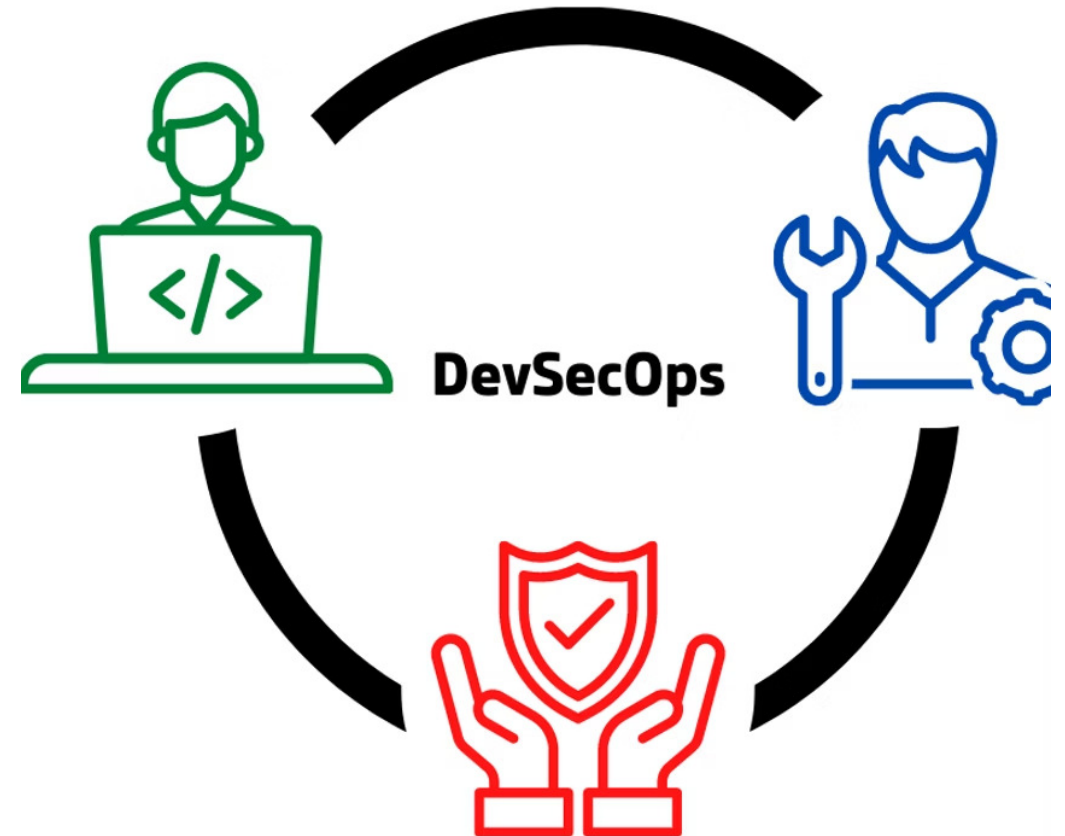
DevOps vs DevSecOps

DevOps

- Set of practice focus on **Development** and **Infra-Operation**
- **Goal:** delivery speed

DevSecOps

- Development, **Security**, and Infra-Operation
- Integration of security into the SDLC
- **Goal:** speed + **safety**



Why DevSecOps (AI Context)



AI-Assisted Coding

AI speeds coding—but also introduces risks.



Emerging Risks

AI-generated code contains
known vulnerabilities in
~45% of cases

[Veracode 2025 GenAI Code Security Report](#)



Our Goal

We need *secure-by-default pipelines* around AI.

Why DevSecOps (AI Context)

AI accelerates development velocity but [expands the attack surface](#). Security must shift left into the pipeline and remain vigilant at runtime.



Velocity Surge

AI-assisted coding increases **output**—and **risk surface**.



Shift Left

Catch vulnerabilities early in code and dependencies.



Runtime Gates

Runtime behavior reveals what static analysis misses.

Goal: Secure-by-default, developer-friendly, measurable.

Pipeline Map (Tools @ a Glance)

- **SAST/SCA:** SonarCloud Quality Gate (PR)
- **Peer + AI review:** Copilot PR / Amazon Q / CodeRabbit
- **DAST (runtime):** Playwright → **OWASP ZAP**
- **(Optional) External:** scheduled surface checks



One pipeline, multiple gates.

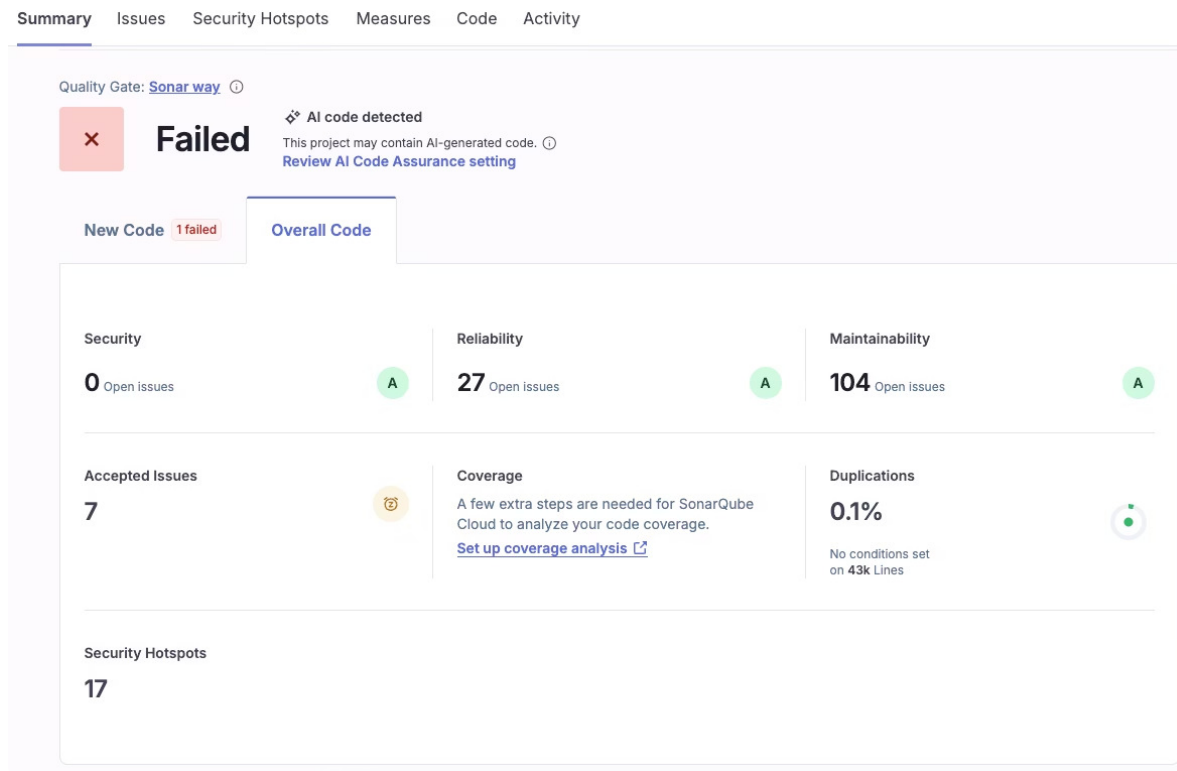
Gate #1 - SAST (What & Why)

What It Covers

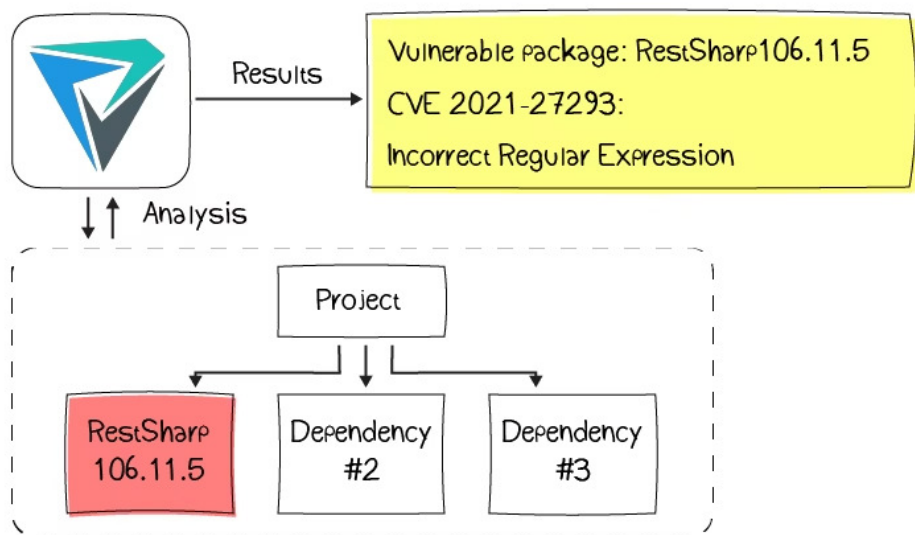
- **Static** scan of source code
- Finds: injections, unsafe APIs, code smells
- **Fail** on High/Critical at PR

Catch vulnerabilities in Code

Fix early, fix cheap.



Gate #1b - SCA (Dependencies)



What It Covers

- Track **CVEs** in libraries
- Validate open-source licenses (compliance)
- Auto PRs for safe upgrades
- Generate CycloneDX SBOM for provenance

You ship your deps too.

Human × AI in Code Review

The most effective reviews blend AI pattern detection with human judgment about design intent and business risk.

Human Reviewers

- Design intent & architecture
- Business context & risk appetite
- Final approval decision

AI Reviewers (Copilot, CodeRabbit, Amazon Q)

- Pattern matching & code smells
- Known insecure API usage
- Missing test coverage hints

📋 Policy: AI assists; humans decide. Document AI suggestions in PR comments for auditability.

Phase 3 playwright #3480

 Open AnowarCST wants to merge 16 commits into `main` from `phase-3-playwright` 

 Conversation 12  Commits 16  Checks 1  Files changed 124



AnowarCST commented on Sep 1 • edited by coderabbitai bot

Summary by CodeRabbit

- New Features
 - Local multi-service stack via Docker Compose; CI DAST workflows (ZAP + Playwright) with uploaded security reports.
- Refactor
 - Admin Portal build/start improvements and simplified debug/auth config.
- Tests
 - Large Playwright E2E suite (auth, dashboard, users, companies, missions, corporate groups, profile/change-password), fixtures, POMs, run scripts, and targeted test fixes.
- Documentation
 - Extensive new and reorganized docs: architecture, onboarding, testing, DAST/security, service contexts.
- Chores
 - Makefile, pre-commit config, expanded .gitignore, CI workflow additions.



amazon-q-developer bot reviewed on Sep 1

[View reviewed changes](#)

amazon-q-developer bot left a comment

DAST Integration Review Summary

Positive Aspects 🍌

- Comprehensive DAST setup with both baseline and active scanning
- Integration of Playwright tests with ZAP for dynamic security testing
- Good separation of concerns between baseline and active scanning workflows
- Proper artifact collection and reporting
- PR commenting functionality for security findings

Security Concerns 🚨

1. Hardcoded AWS credentials in `.env.example` need to be removed
2. Empty ZAP API key in configuration
3. SSL/TLS certificate validation disabled in test environment
4. Fixed timeouts might be insufficient for larger applications

Recommendations 🛠️

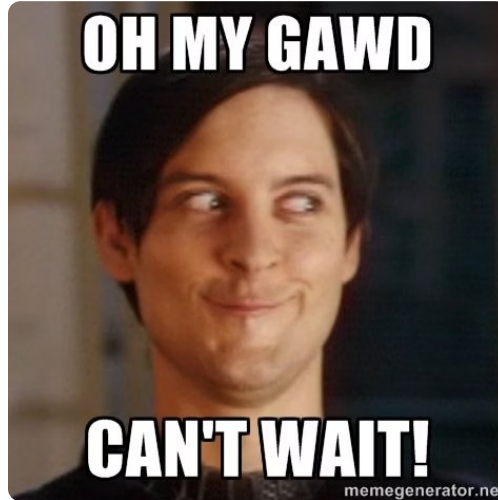
1. Remove all hardcoded credentials and use placeholders in example files
2. Implement proper ZAP API key management through secrets
3. Add environment-specific SSL/TLS validation configuration
4. Make timeouts configurable based on application size
5. Enhance error handling in XML processing
6. Add retry logic for ZAP daemon startup

Please address the security concerns before merging, particularly the hardcoded credentials issue which is a critical security risk.

What Next?

- SAST
- SCA
- Human peer review
- AI Review

PR Merge and deploy to PRODUCTION





SAST ✓

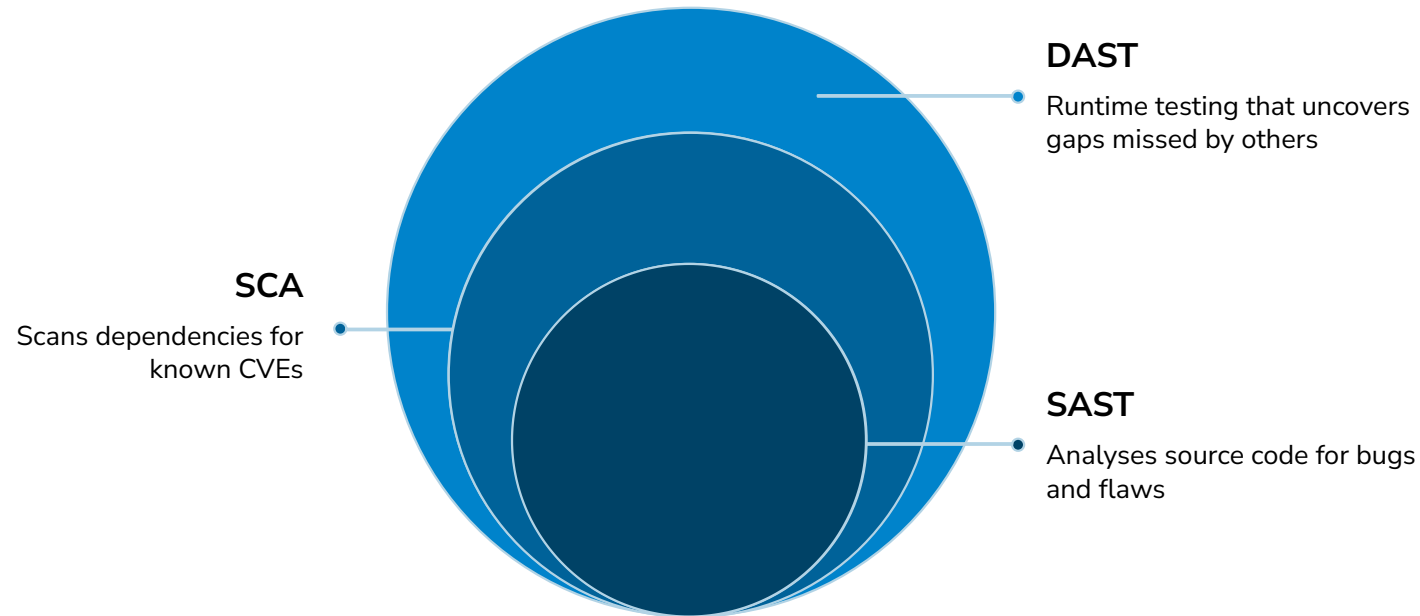
SCA ✓

Runtime & Infra: 🚨

Unit tests cleared... runtime says 'step aside'.

Where DAST Fits

- **Runtime** behaviour under real flows
- Catches headers, cookies, auth, injections, **Complements** SAST/SCA



Layered security: **SAST** for **code**, **SCA** for **dependencies**, **DAST** for **runtime**.

OWASP ZAP Overview (Pros/Cons)



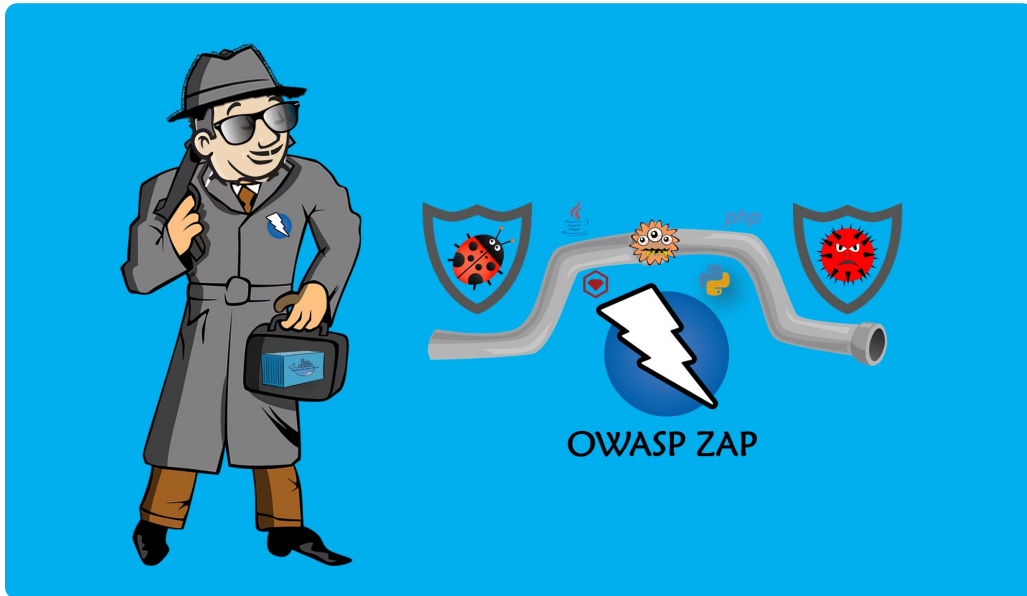
OWASP
Zed Attack Proxy

What is ZAP?

- OWASP's flagship **proxy-based DAST** tool.
- Modes: **Baseline** (safe), **Full** (active), **API** (OpenAPI/GraphQL).
- Dockerized, extensible, **CI/CD friendly**.
- Free & **open-source**, widely adopted.

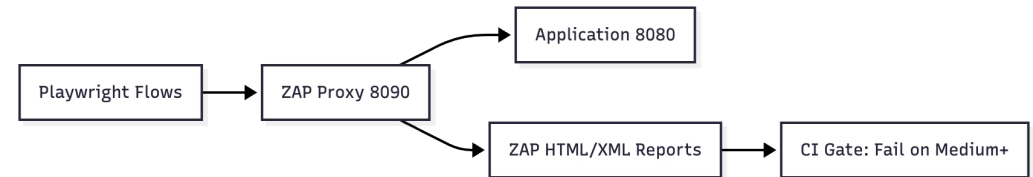
ZAP = developer-friendly runtime security: free, flexible, but needs tuning.

Bridge - From Functional to Security Tests



- **Playwright** automates real user journeys → UI, features, API calls.
- **ZAP proxy** inspects all Playwright traffic at runtime.
- **CI/CD gate:** ZAP reports → pipeline passes or fails.

CI/CD Flow:



Your E2E tests, now security tests.

Use AI Tool to fix the ZAP Report

- Alert → Evidence → URL → Remediation
- Show **one Medium** (CSP) & **one High** (reflected XSS)
- Action: ticket → fix → rerun → green

Ask your **AI Tool** to fix the ZAP Report

AI assists; humans decide. Re-scan until clean.

ZAP Scanning Report

Site: <http://localhost:8080>

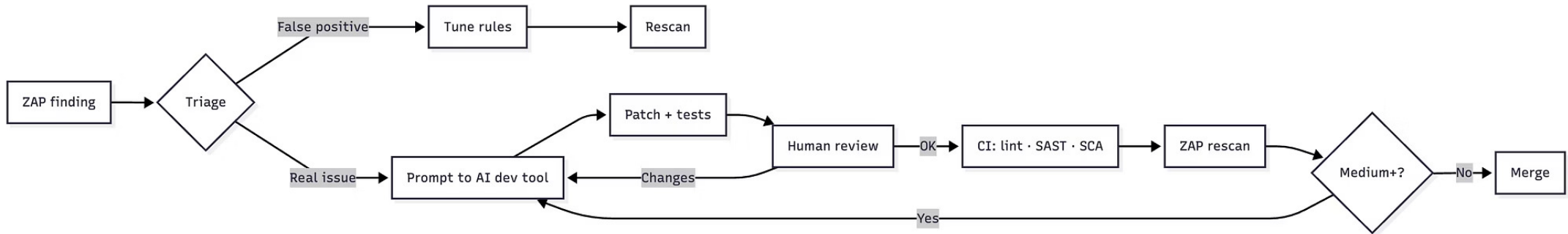
Generated on Fri, 4 Feb 2022 17:38:41

Summary of Alerts

Risk Level	Number of Alerts
High	2
Medium	4
Low	8
Informational	6

Alerts

Name	Risk Level	Number of Instances
Anti-CSRF Tokens Check	High	10
Cross Site Scripting (Reflected)	High	2
Buffer Overflow	Medium	529
Content Security Policy (CSP) Header Not Set	Medium	58
Example Passive Scan Rule: Denial of Service	Medium	7
X-Frame-Options Header Not Set	Medium	55
Absence of Anti-CSRF Tokens	Low	73
Application Error Disclosure	Low	1
Cookie No HttpOnly Flag	Low	1
Cookie without SameSite Attribute	Low	2
In Page Banner Information Leak	Low	3



Start Monday: Your Action Plan

1 Enable SonarCloud Quality Gate

Enforce SAST checks on every PR; fail High/Critical, advisory on Medium.

2 Add Playwright → OWASP ZAP Baseline

Route functional test flows through ZAP proxy; fail PR on Medium+ findings. Run nightly full scans.

3 Activate AI Review Assistants

Enable Copilot PR reviews or CodeRabbit; require human sign-off on suggestions.

4 Pick One External Option

Deploy OWASP Amass + Nuclei (OSS) or Detectify for weekly external scanning.

Q&A / Resources



DevSecOps Quick Start: <https://gist.github.com/AnowarCST/9b9fbb2867e066cd01509da9481f74bb>

Secure-by-default is a habit.

Thank You!