DevSecOps: Moving Beyond SAST in Pipelines

Jon Jarboe
Developer Advocate
About

Jon Jarboe

Dev Advocacy @ Accurics
Development, AppSec, IaC
DevSecOps enthusiast
Favorite attack technique: Watering hole
About Accurics

Our mission is to enable cloud cyber resilience with a codified, developer-first approach to security.

We work in automated workflows to self-heal the cloud by enabling developers to programmatically detect and respond to breach paths throughout the development lifecycle.
Agenda: Why DevSecOps?

- DevOps is working!
- Security is not.
- A new approach is needed: DevSecOps.
Traditional cloud security tools are not built for developers.
Automating SAST

- Finds real problems and streamlines some workflows
- Doesn't help with:
  - Manual remediation effort
  - Need for manual review of security findings
  - Risk assessment of findings
  - Prioritization of findings
  - Alert fatigue, or information overload
  - Cultural fit and process inefficiencies
Traditional cloud security tools are not built for developers

Commit code

Write application code
Write Infrastructure as Code

Write Infrastructure as Code

App Vulnerabilities
SAST

Infrastructure Misconfigurations
CSPM, CIEM

App Vulnerabilities
DAST / IAST

Cloud Native App

OSS Vulnerabilities
SCA

Threat Protection
CWPP, RASP

Infrastructure Misconfigurations
CSPM, CIEM
And security only gets more challenging as developers embrace <*>Ops

**Cloud Native Security**
- App Vulnerabilities
  - SAST
- Infrastructure Misconfigurations
  - CSPM, CIEM
- OSS Vulnerabilities
  - SCA

**Lack of context between tools equals complications**

**Threat Protection**
- CWPP, RASP
DevSecOps Challenges

- Manual remediation effort
- Need for manual review of security findings
- Risk assessment of findings
- Prioritization of findings
- Alert fatigue, or information overload
- Cultural fit and process inefficiencies

Potential solutions:
- Automated fixes
- Improved accuracy
- Risk scoring
What are breach paths?

- Security tools find so many weaknesses
  - Often hard for teams to prioritize according to risk

- Most breaches result from attackers exploiting multiple weaknesses

- Breach path is collection of weaknesses that attacker can exploit to reach objective

- Focusing on breach paths enables risk-based:
  - Automation
  - Prioritization
  - Remediation
Programmatically Detecting Breach Paths During Development

**Problem:** SAST tools do not have context if a vulnerability is exposed, exploitable, and if lateral movement is possible.

**Solution:** Combining data from SAST & IaC scanning tools will enable end-to-end breach path identification.
A developer-first approach to security enables DevSecOps

**CODE**

1. Secure cloud development
   - Write application code
   - Write Infrastructure as Code
   - Write Deployment as Code

2. Programmatically detect breach paths
   - App vulnerabilities + OSS vulnerabilities
   - IaC misconfigurations
   - Identity entitlements

3. Programmatically fix breach paths
   - Traffic monitors

**COMMIT & BUILD**

4. Programmatically govern deployments
   - Deployments
   - Configurations + Relationships + Events

5. Programmatically detect & respond
   - Threats/Incidents

**DEPLOY & RUN**

- Build Automation Tool
- Cloud Native App
Thank You

Let's Chat!

SKILup Day: DevSecOps – "Meet the Speakers" Chat Lounge

Email – jon@accurics.com