

Shift Left Compliance & Security

Jim Doran, IBM

What's at Stake

Brand reputation

25%

company value attributed to reputation

Client trust

69%

of consumers believe companies are vulnerable to cyberattacks

Financial fines

\$425M

Fines levied on Equifax to help people affected by the data breach

Industry Pain points

Skills shortage

1.8 million

unfulfilled cybersecurity jobs by 2022 facing CISO

Compliance

\$31M

Average cost for Financial Services Company

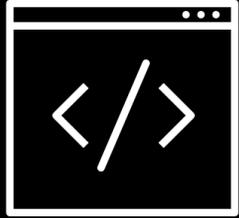
Complexity

94%

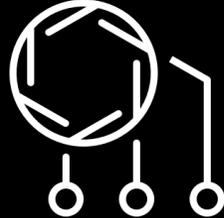
of organizations have multiple clouds

DevOps has changed

Speed with Control for all Apps.



Everything as Code



Automate Everything



Team Collaboration



SecDevOps

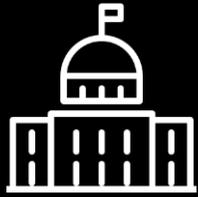
“ We have integrated the IBM Cloud hosted in data centers dedicated to the bank. We will also strengthen our Hybrid Cloud ‘As a Service’ capabilities using IBM solutions offered via its public Cloud to support the development of new services, including Continuous Delivery, Tekton pipelines, test and applications environments.

- *IBM Banking Client*

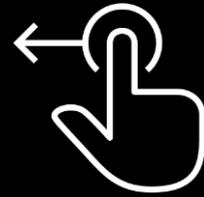
”

Security has changed

Governance and Compliance is Built-Into Your App.



Governance



Shift-Left



Vulnerability Detect



Audit-Ready

“

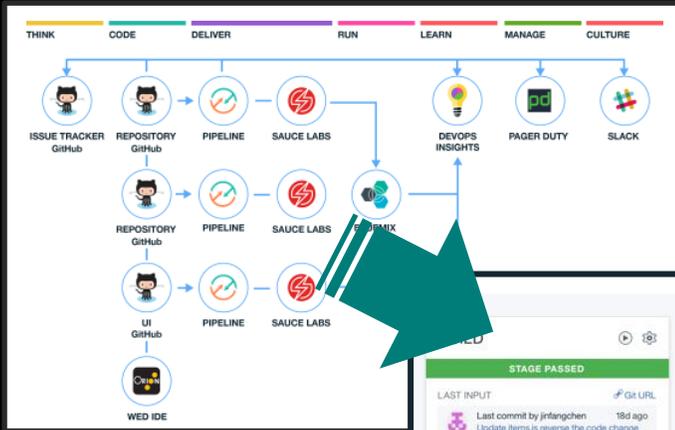
I need to ensure that my organization workloads comply with regulations and security best practices by setting guardrails around what application teams can do, by continuously monitoring the security and compliance posture and identifying violations, and by collecting evidence for audit without disrupting agile development processes or application availability.

”

- IBM Financial Services Client

Securely Develop at the Speed of Cloud

- Agile transformation *requires* **Cloud** to maximize benefits
- **SecDevOps** is the way to develop *securely and efficiently* for Cloud



The screenshot displays a CI/CD pipeline dashboard with the following stages and details:

- DEV:** STAGE PASSED. LAST INPUT: Stage: BUILD / Job: Build. Jobs: Build, Upload UT, Gate. LAST EXECUTION RESULT: Build 66.
- TEST:** STAGE PASSED. LAST INPUT: Stage: BUILD / Job: Build. Jobs: Deploy, Functional Tests. LAST EXECUTION RESULT: Build 66.
- SECURITY SCAN:** STAGE PASSED. LAST INPUT: Stage: BUILD / Job: Build. Jobs: Security Scan. LAST EXECUTION RESULT: Build 20.
- PROD:** STAGE PASSED. LAST INPUT: Stage: BUILD / Job: Build. Jobs: Blue/Green Deploy. LAST EXECUTION RESULT: Build 59.

The screenshot shows a 'Production gate dashboard' for a 'Weather Application' (Build ID: master:1). It includes a table of test results and a 'Build Information' section.

APPLICATION	BUILD IN STAGING	PRODUCTION GATE POLICY	LAST EVALUATION	BUILD IN PRODUCTION
Weather Application	master:1	✓	Mar 3, 8:46 AM	master:1

	Staging	Production
Code coverage	✓	✓
Unit test	✓	✓
Functional test	✓	✓
Dynamic scan	●	●
Static scan	●	●
Sonarqube bugs	●	●
Sonarqube code smells	●	●
Sonarqube vulnerabilities	●	●

Build Information Application name: Weather Application
Build ID: master:1

Build in STAGING
Last commit ID 1317eea
Deployed Mar 3, 8:45 AM
Branch name master
Data received Mar 3, 8:40 AM
Policy name Weather Unit Test, Code Coverage, and FVT Checks
Decision Passed

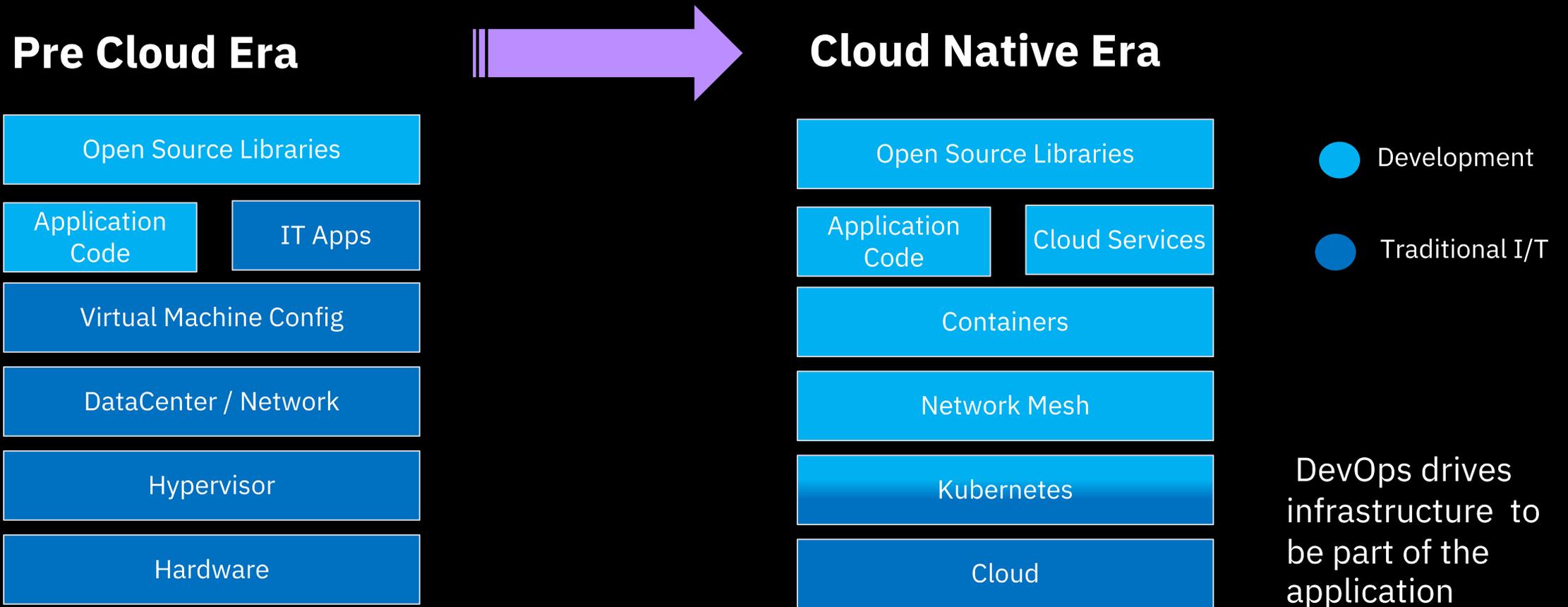
Test Summary:

- Code Coverage: 100% (Statements: 100, Functions: 100)
- Unit Test: 100% (Total: 4, Passed: 4, Failed: 0)
- Functional Test: 100% (Total: 3, Passed: 3, Failed: 0)

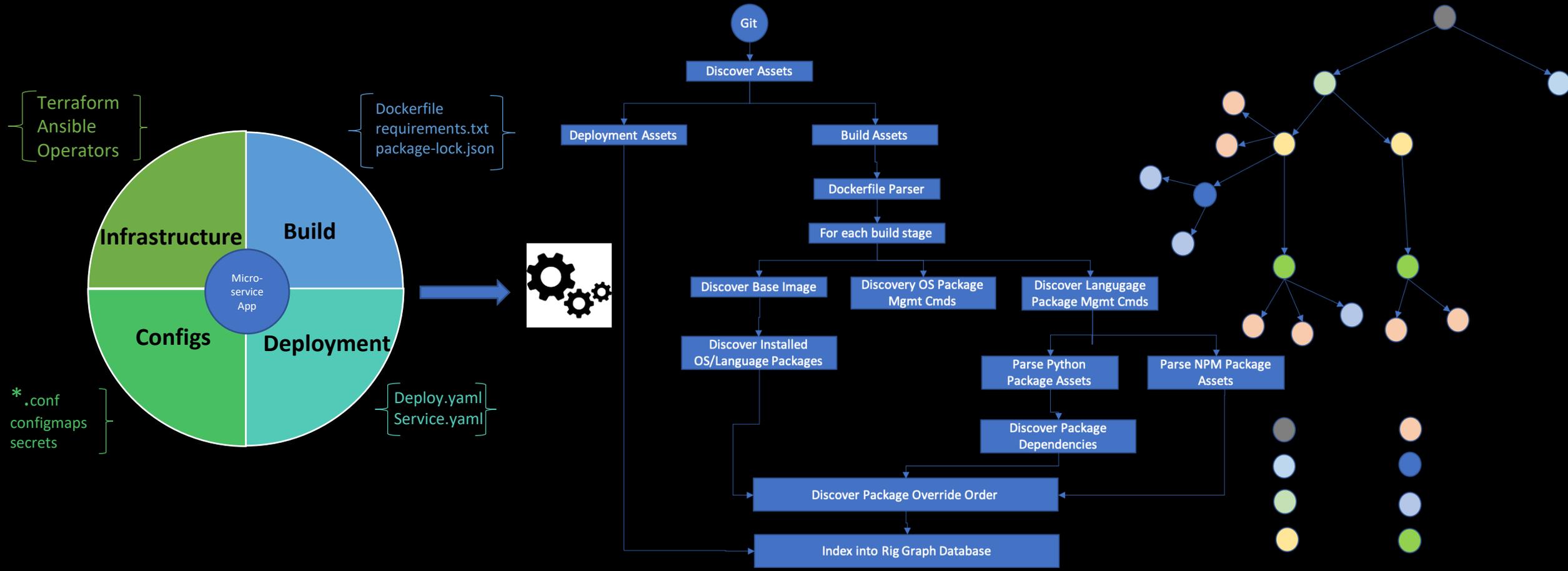
Need both **Speed and Control:**

- **Deploy** >20 times per day with **quality**
- **Automate** toolchain with quality & security gates
- Use **Templates**

Cloud Native Apps require increased scope



Anatomy of a modern cloud application



- Vulnerability Checks
- Network
- Bill-of-Material / Evidence
- OpenSource Licenses
- Third-party Service
- CIS Checks
- Cloud Services

Pipeline Integration

- PR Pipeline – Trigger checks on every pull request
- CI Pipeline – Trigger checks in every build pipeline
- CD Pipeline – Trigger checks before every deployment

Developer Feedback

- PR Comments: All findings are reported as comments to PR in PR pipeline
- Issues: In CI/CD pipeline, any failures are reported as issues

OPA-based Policy Control

- Policy configuration are defined out-of-band of pipeline
- Pipeline Gates: Control PR merge through policy gates

Code Risk Analyzer

Automate security and compliance checks and remediation based on static assets in git repository, leveraging threat intelligence from **Snyk** and **Clair**.

The screenshot displays the Code Risk Analyzer interface with several reports:

- Vulnerability Detection:** GitSecure Vulnerability Report table with columns: Control ID, Section, Description.
- License Auditing:** GitSecure Open Source License Assessment: Needs Review. FedRAMP Controls covered table with columns: Control ID, Section, Description.
- CIS Compliance:** Docker CIS Community Edition 1.13. Table with columns: Control ID, Section, Description, Compliant?.
- Best Practice Lint Checks:** GitSecure Dockerfile Linter table with columns: Line #, ID, Description.
- Bill of Material:** GitSecure Bill of Material.
- Third-party Image Discovery & Scanning:** GitSecure Deployment Vulnerability Report table with columns: Control ID, Section, Description.

A detailed view of a configuration risk is shown in a modal window:

Configuration risks discovered in the following manifest file(s)

- ✖ Manifest File: /deployment.yml
- ▶ Resource Name: hello-compliance-app Namespace: default
- ▼ Resource Name: hello-service Namespace: default

```
Container Name:
Overall Risk: `Medium`
Description: Ensure containers are not exposed through an external load balancer
Vector: `AV:N/S:N/C:H/I:L/A:H/E:H`
Impact: ``
CIS Reference:
```

Vulnerability Scans

- Discover vulnerabilities in your Application (python, node, java, go) and OS stack (base image) based on curated threat intelligence from Snyk and Red Hat Clair
- Recommendation to fix vulnerabilities
- Auto-remediations for some available fixes via pull requests
- Continuous monitoring against new vulnerabilities and automated notifications

Deployment Analysis

- Discover misconfigurations in your deployment *.yaml against CIS Standards

Bill-of-Materials

- Generate Bill-of-Materials accounting for all the build dependencies and their sources for your application
- BoM-Diff to allow comparing diffs in any dependency w.r.t. base branch

Network Policy Analysis

- Built-in Checks to identify any redundancy or gaps in network policies

Terraform Configuration Analysis

- Parsing terraform templates for different cloud services (IAM, COS, CIS, LogDNA-AT) to discover any security mis-configurations
- Implements 20+ policy checks, all anchored around NIST controls

CI pipeline

CODE Detect secrets in code
Code reviews
Unit tests
Open source scan
Static code scan

BUILD Build deployable artifacts
Sign deployable artifacts
Run vulnerability scan
Open source scans
Dev deployment

Setup

Code...

Unit Tests

Static Code Scan

- code-extract-toolchain-crn
- code-fetch-credentials
- code-fetch-evidence-locker
- code-fetch-code
- build-doi-publish-buildrecord
- code-unit-tests
- code-unit-tests-create-issue
- code-unit-tests-create-evidence
- code-unit-tests-create-doi-data
- add-unit-test-results-to-doi
- code-vulnerability-scan-uploader
- code-vulnerability-scan
- code-vulnerability-scan-create-issue
- code-vulnerability-scan-create-evide...
- code-cis-check
- code-cis-check-create-issue
- code-cis-check-create-evidence

Static Code Scan

Code Review
Detect Secrets

Build >

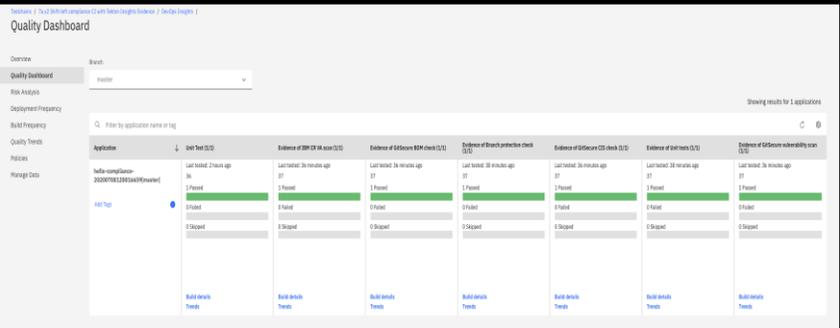
V Scan >

Build

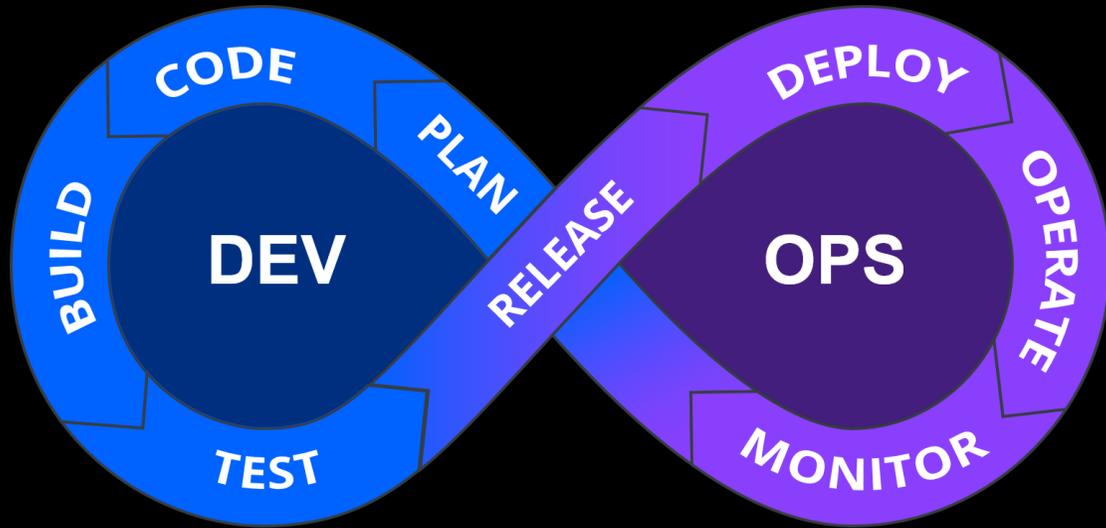
Sign >

Completion

- code-bom-check
- code-bom-check-create-issue
- code-bom-check-create-evidence
- code-branch-protection
- code-branch-protection-create-issue
- code-branch-protection-create-evide...
- code-check-dockerfile
- build-containerize
- build-update-inventory
- build-vulnerability-advisor
- build-vulnerability-advisor-result-ad...
- build-vulnerability-advisor-create-iss...
- build-vulnerability-advisor-create-evi...
- build-docker-image-sign
- build-fetch-iks-cluster-config
- build-pipeline-1-kubecttl-task
- build-deploy-dev
- build-pipeline-evaluator

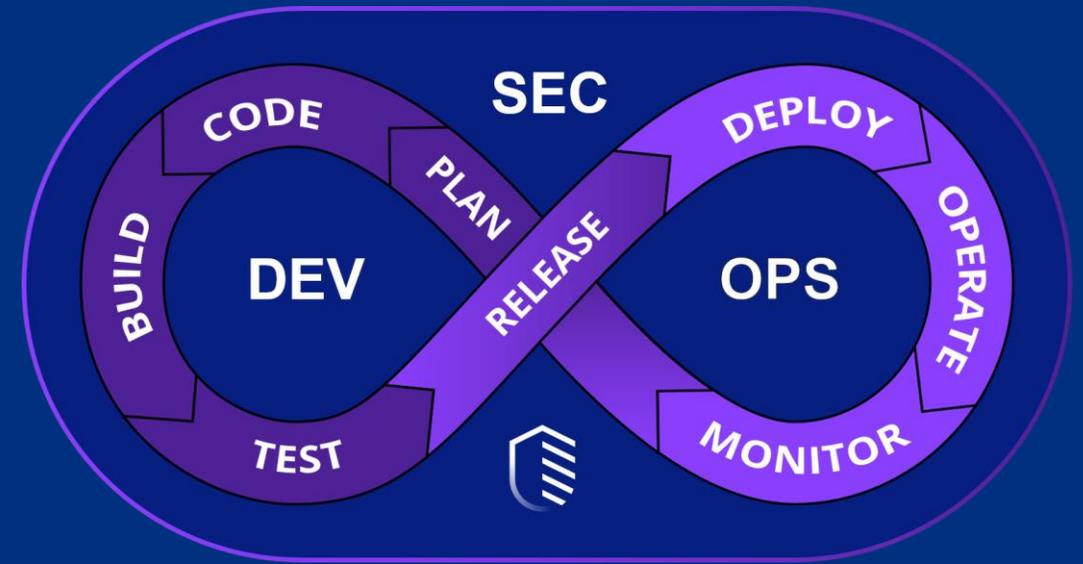


Traditional agile approach



- Siloed roles and level of expertise
- Fragmented / bypassed security checks
- Lack audit readiness
- Lack of visibility into development process
- Reactive and time-consuming security remediation
- Runtime Probes required

Continuous Dev + Sec + Ops



- Auto-Remediation fix vulnerabilities / config
- Consistent security patterns (pipelines)
- Immutable runtime platform
- Continuous threat intelligence model
- Proactive Audit readiness
- Security checks scale with Development process

Backup slides