sonatype

DevSecOps Reference Architectures 2020

Derek E. Weeks
VP and DevOps Advocate
Sonatype

About this collection

1. The reference architectures can be used to **validate choices** you have made or are planning to make.

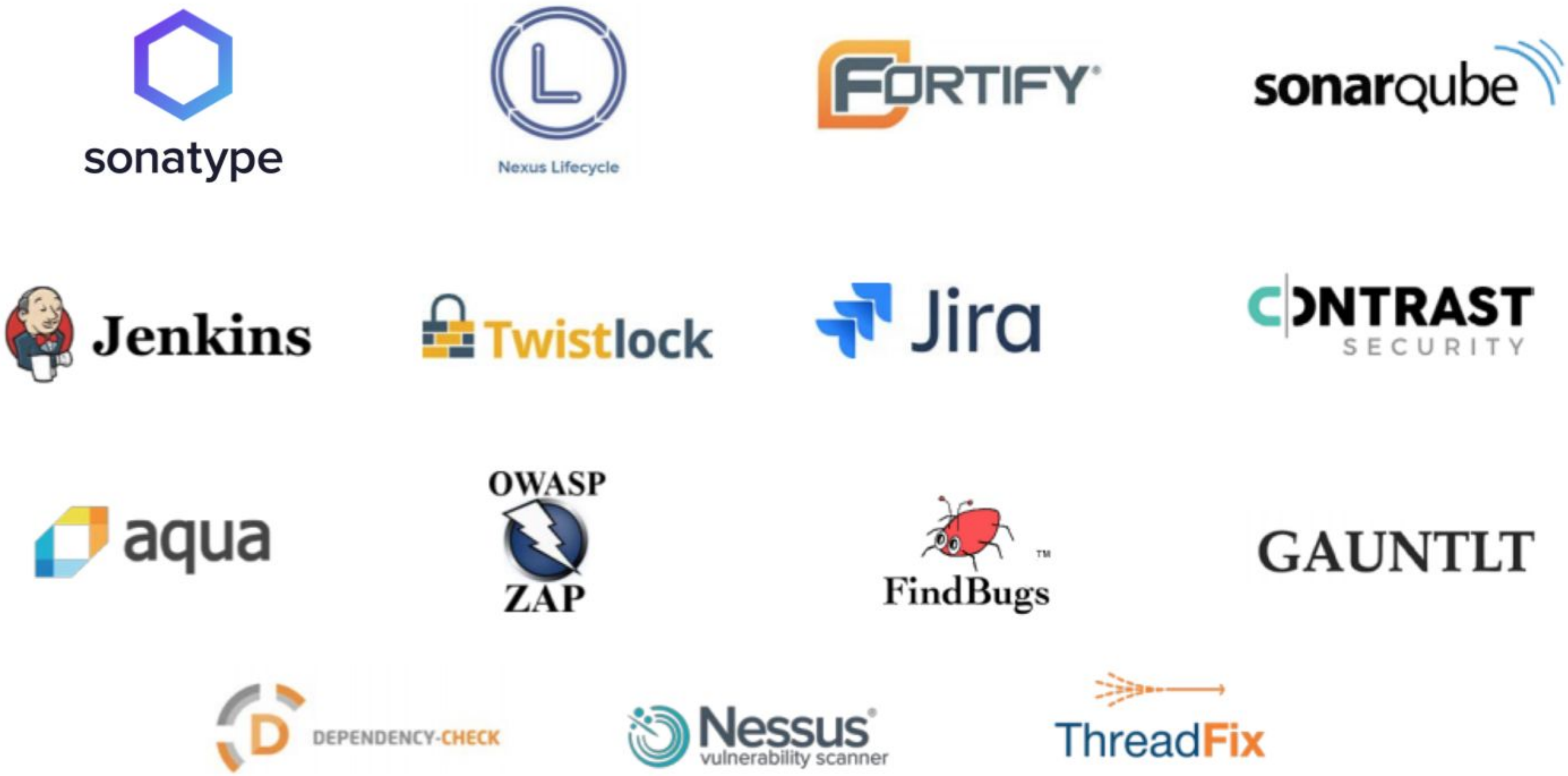
2. They are curated from the **community.** You will notice a number of common elements that are used repeatedly.

3. Each image has a link to its **original source** in the speaker notes, enabling you to deep dive for more knowledge.

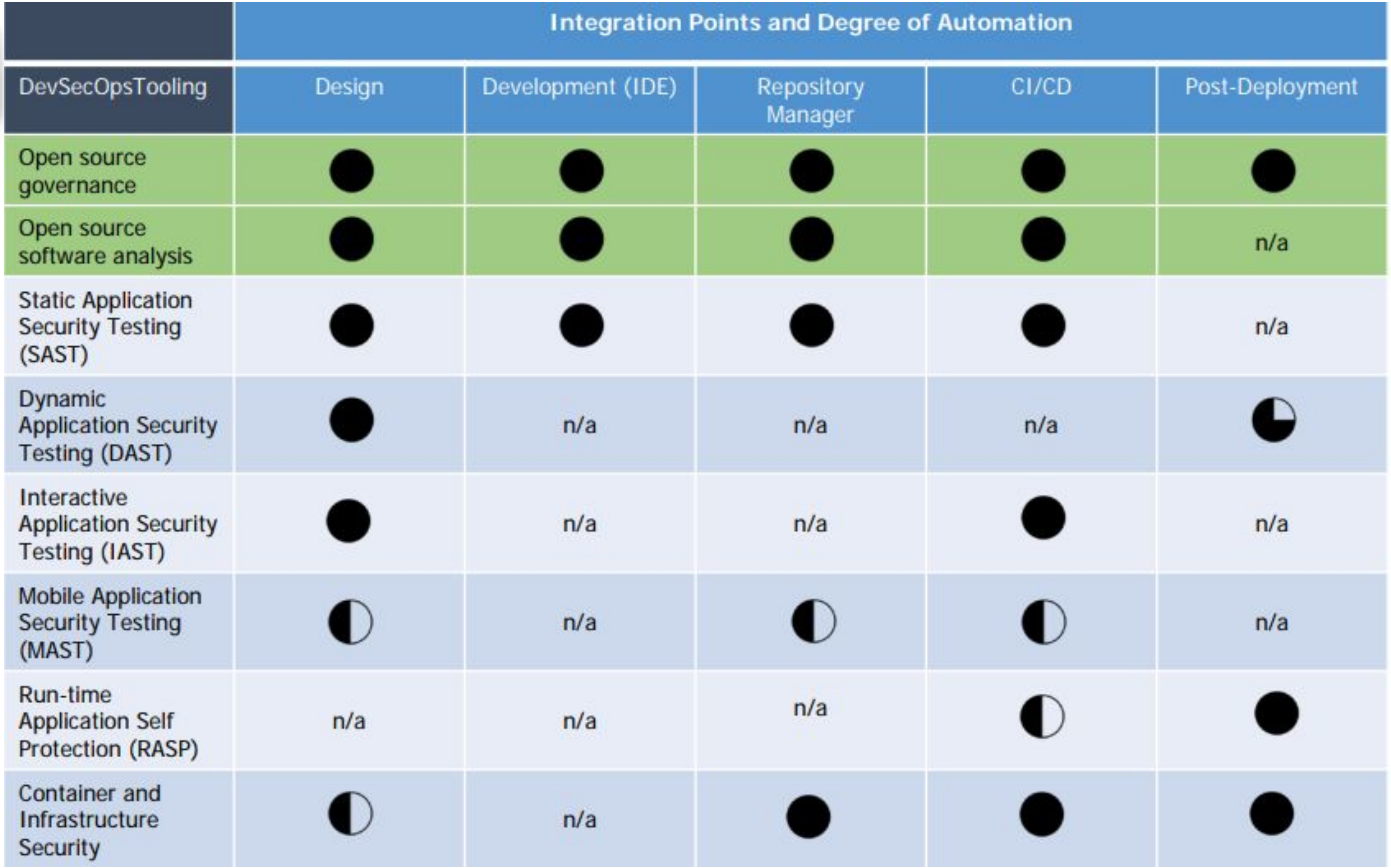
If you would like to have **your reference architecture** added to this deck, please send it to community@sonatype.com.

sonatype

Common Elements of DevSecOps Pipeline



Degrees of DevSecOps Automation

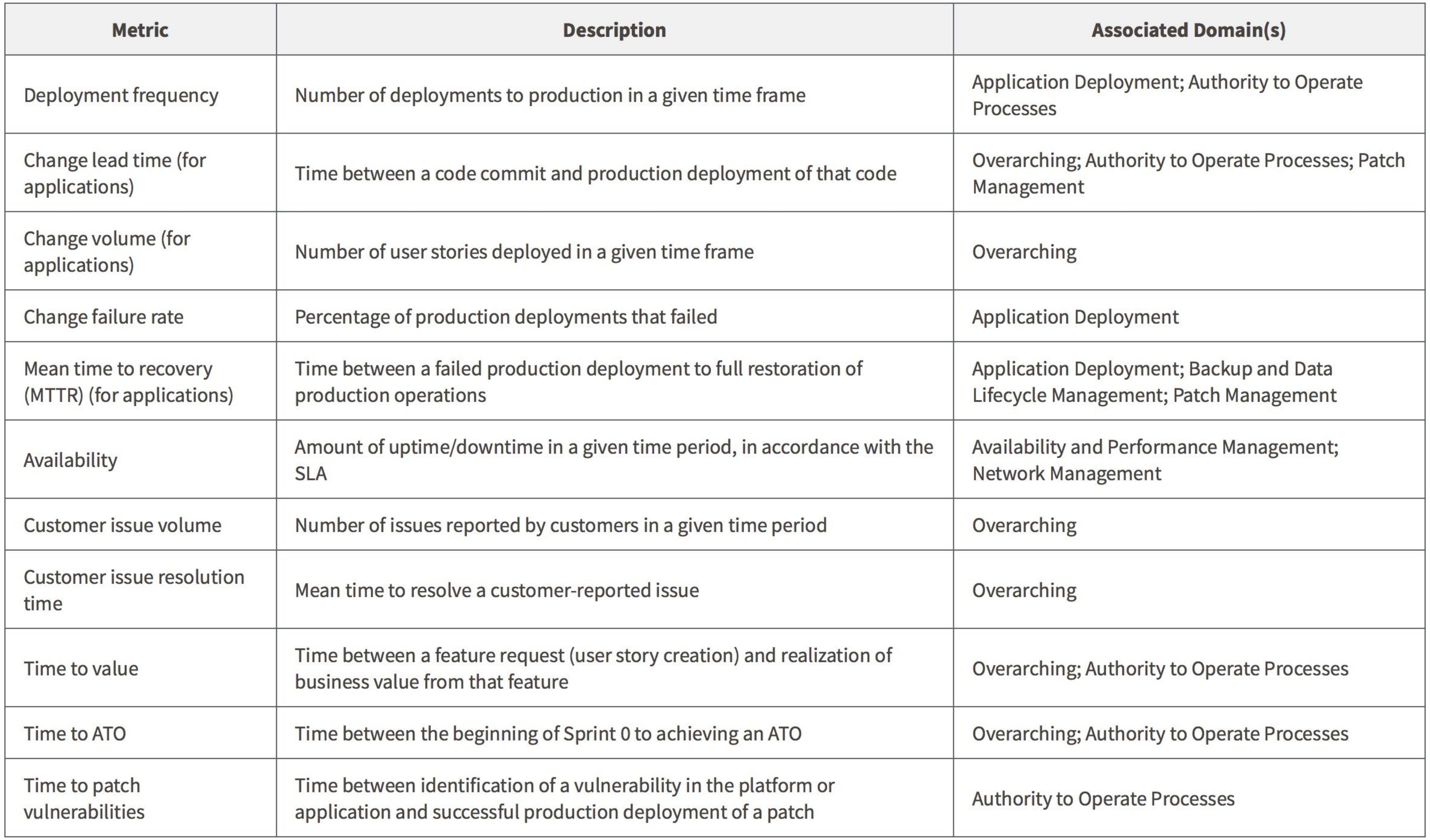


	Integration Points and Degree of Automation				
DevSecOpsTooling	Design	Development (IDE)	Repository Manager	CI/CD	Post-Deployment
Open source governance	●	●	●	●	●
Open source software analysis	●	●	●	●	n/a
Static Application Security Testing (SAST)	●	●	●	●	n/a
Dynamic Application Security Testing (DAST)	●	n/a	n/a	n/a	◕
Interactive Application Security Testing (IAST)	●	n/a	n/a	●	n/a
Mobile Application Security Testing (MAST)	◐	n/a	◐	◐	n/a
Run-time Application Self Protection (RASP)	n/a	n/a	n/a	◐	●
Container and Infrastructure Security	◐	n/a	●	●	●

Source: Gartner, December 2017 - “Structuring Application Security Practices and Tools to Support DevOps and DevSecOps”

sonatype

GSA's DevSecOps Maturity Model

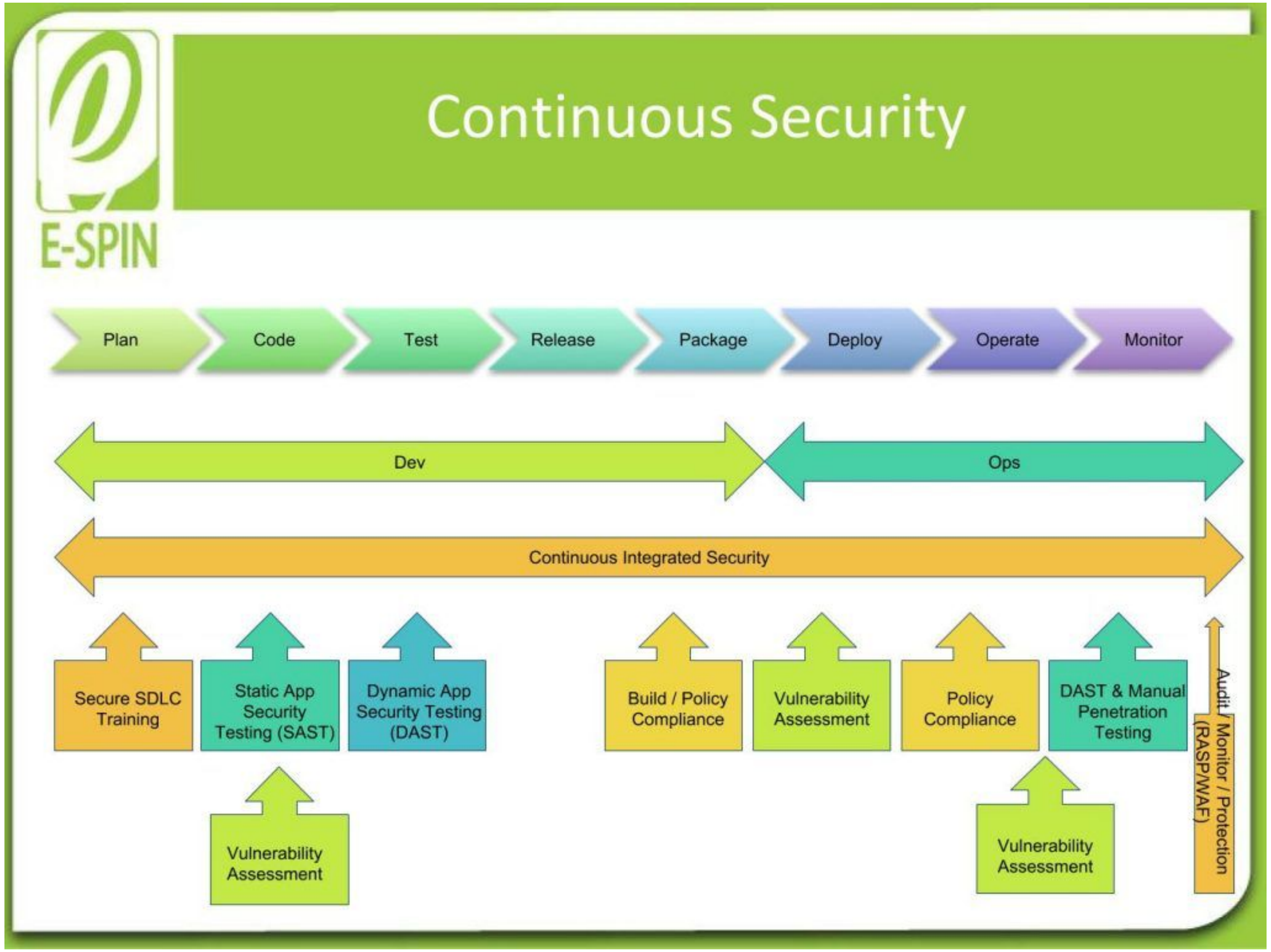


Metric	Description	Associated Domain(s)
Deployment frequency	Number of deployments to production in a given time frame	Application Deployment; Authority to Operate Processes
Change lead time (for applications)	Time between a code commit and production deployment of that code	Overarching; Authority to Operate Processes; Patch Management
Change volume (for applications)	Number of user stories deployed in a given time frame	Overarching
Change failure rate	Percentage of production deployments that failed	Application Deployment
Mean time to recovery (MTTR) (for applications)	Time between a failed production deployment to full restoration of production operations	Application Deployment; Backup and Data Lifecycle Management; Patch Management
Availability	Amount of uptime/downtime in a given time period, in accordance with the SLA	Availability and Performance Management; Network Management
Customer issue volume	Number of issues reported by customers in a given time period	Overarching
Customer issue resolution time	Mean time to resolve a customer-reported issue	Overarching
Time to value	Time between a feature request (user story creation) and realization of business value from that feature	Overarching; Authority to Operate Processes
Time to ATO	Time between the beginning of Sprint 0 to achieving an ATO	Overarching; Authority to Operate Processes
Time to patch vulnerabilities	Time between identification of a vulnerability in the platform or application and successful production deployment of a patch	Authority to Operate Processes

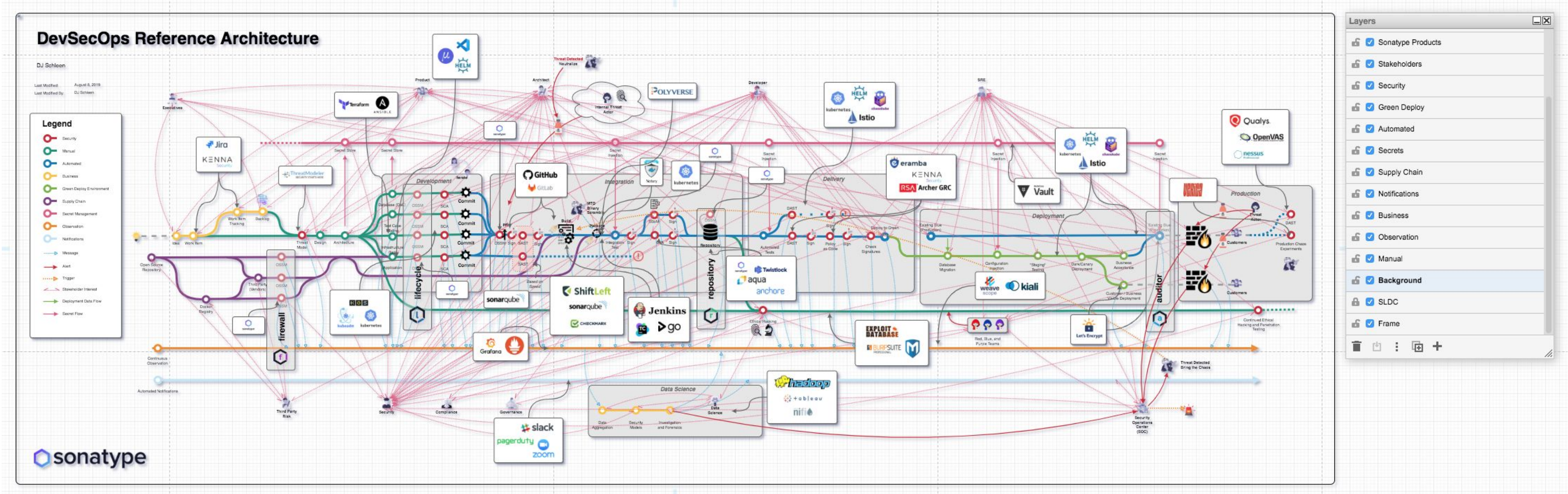
Source: GSA, "DevSecOps Guide"

sonatype

DevSecOps according to E-SPIN


E-SPIN
Continuous Security
Plan
Code
Test
Release
Package
Deploy
Operate
Monitor
Dev
Ops
Continuous Integrated Security
Secure SDLC Training
Static App Security Testing (SAST)
Dynamic App Security Testing (DAST)
Build / Policy Compliance
Vulnerability Assessment
Policy Compliance
DAST & Manual Penetration Testing
Audit / Monitor / Protection (RASP/WAF)
Vulnerability Assessment
Vulnerability Assessment

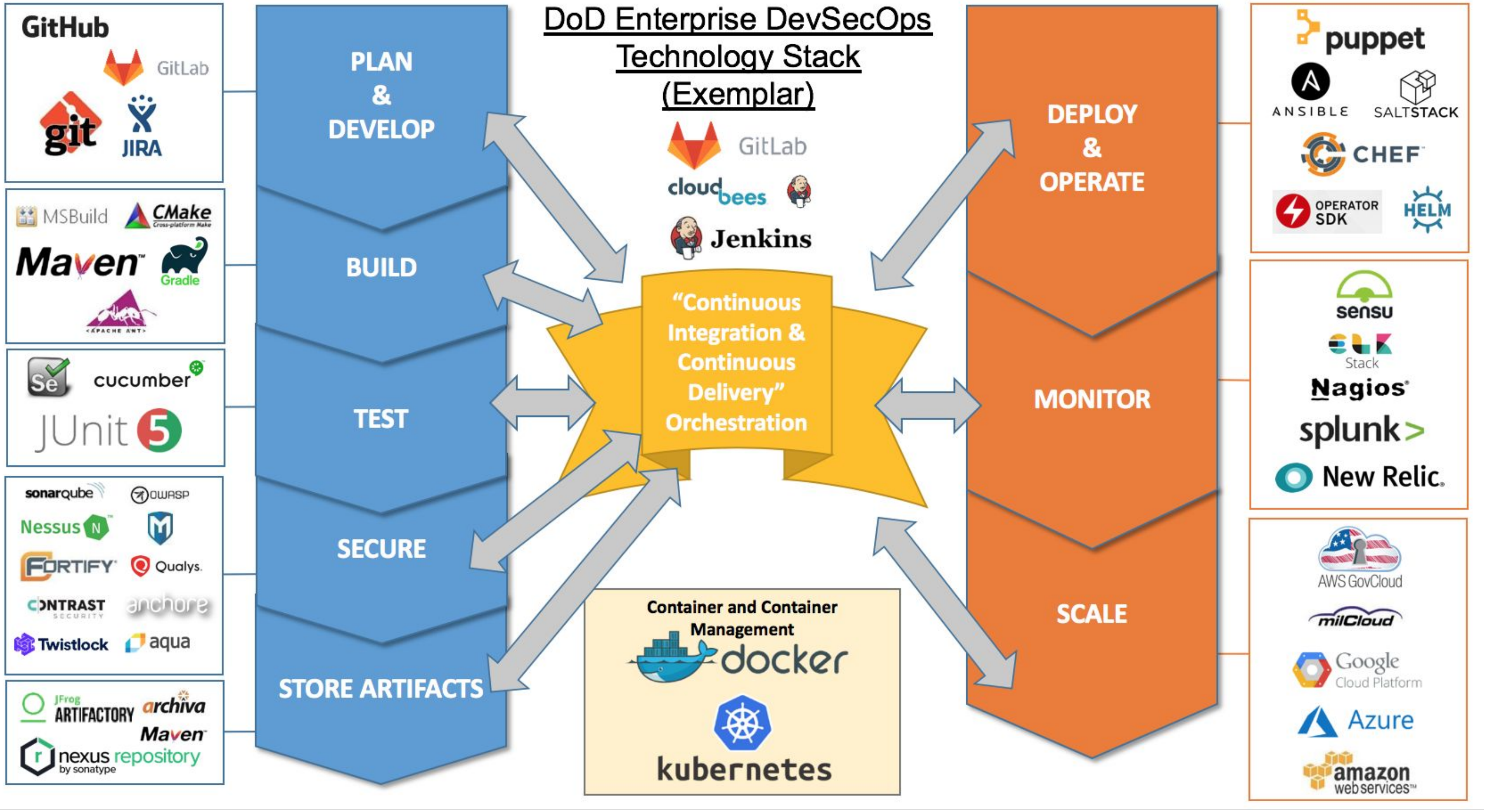
DevSecOps according to DJ Schleen at Sonatype



☑ Sonatype Products
☑ Stakeholders
☑ Security
☑ Green Deploy
☑ Automated
☑ Secrets
☑ Supply Chain
☑ Notifications
☑ Business
☑ Observation
☑ Manual
☑ Background
☑ SLDC
☑ Frame

https://www.sonatype.com/referencearchitecturetestdrive

DevSecOps according to Nicolas Chaillan and U.S. Dept of Defense



DoD Enterprise DevSecOps Technology Stack (Exemplar)

PLAN & DEVELOP

BUILD

TEST

SECURE

STORE ARTIFACTS

"Continuous Integration & Continuous Delivery" Orchestration

Container and Container Management

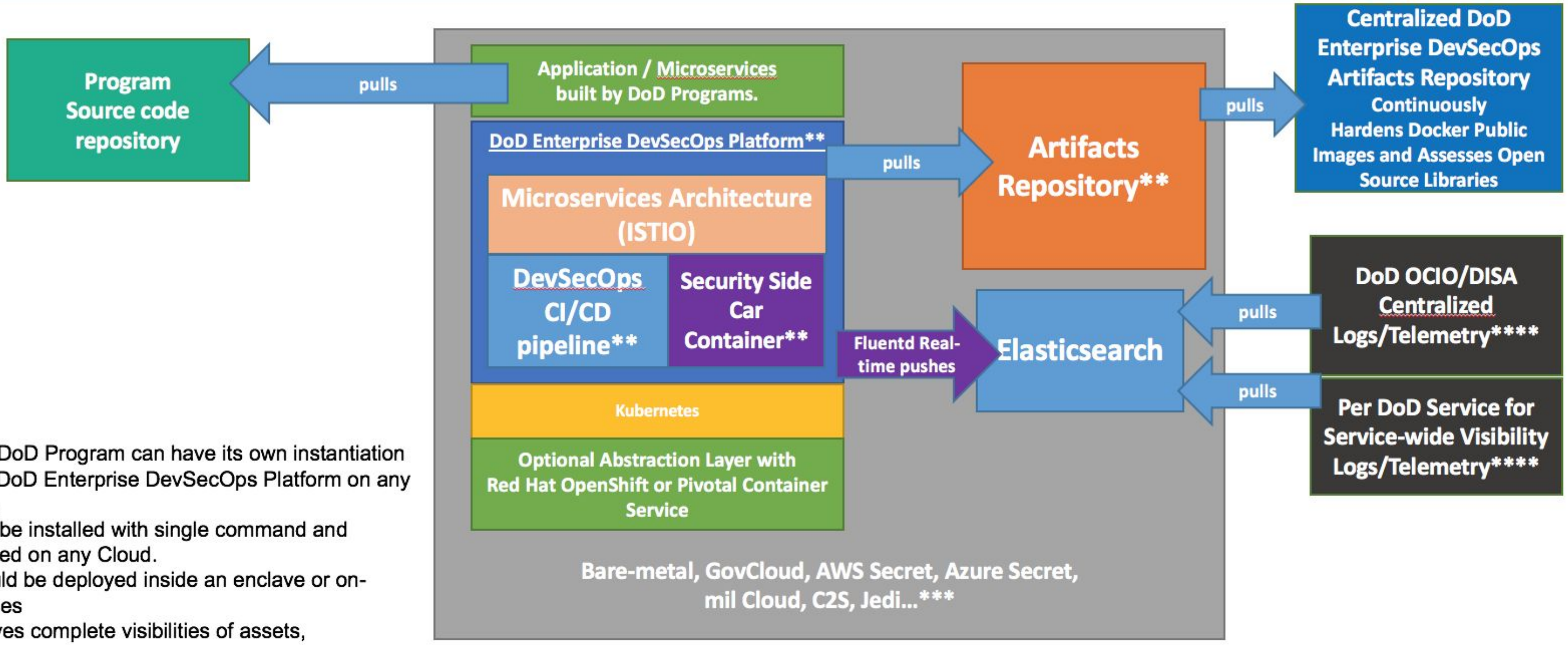
DEPLOY & OPERATE

MONITOR

SCALE

DevSecOps according to Nicolas Chaillan and U.S. Dept of Defense

DoD Enterprise DevSecOps Architecture*



Program Source code repository

pulls

Application / Microservices built by DoD Programs.

DoD Enterprise DevSecOps Platform**

Microservices Architecture (ISTIO)

DevSecOps CI/CD pipeline**

Security Side Car Container**

Kubernetes

Optional Abstraction Layer with Red Hat OpenShift or Pivotal Container Service

Bare-metal, GovCloud, AWS Secret, Azure Secret, mil Cloud, C2S, Jedi...***

pulls

Artifacts Repository**

pulls

Centralized DoD Enterprise DevSecOps Artifacts Repository Continuously Hardens Docker Public Images and Assesses Open Source Libraries

Fluentd Real-time pushes

Elasticsearch

pulls

DoD OCIO/DISA Centralized Logs/Telemetry****

pulls

Per DoD Service for Service-wide Visibility Logs/Telemetry****

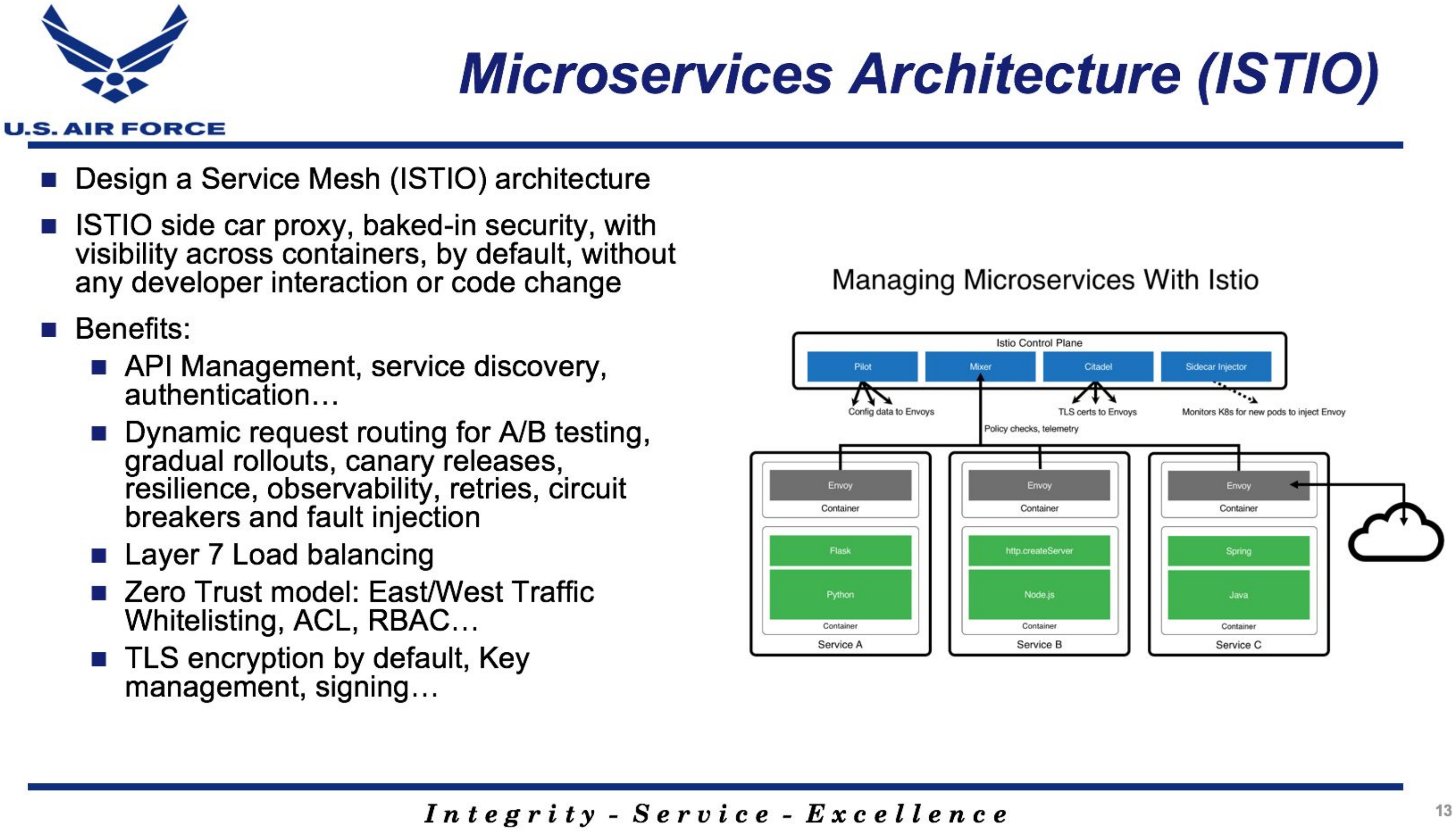
*each DoD Program can have its own instantiation of the DoD Enterprise DevSecOps Platform on any Cloud.

** can be installed with single command and deployed on any Cloud.

*** could be deployed inside an enclave or on-premises

**** gives complete visibilities of assets, security/vulnerability state etc. can be integrated to existing cybersecurity shared services.

DevSecOps according to Nicolas Chaillan and U.S. Dept of Defense



U.S. AIR FORCE

Microservices Architecture (ISTIO)

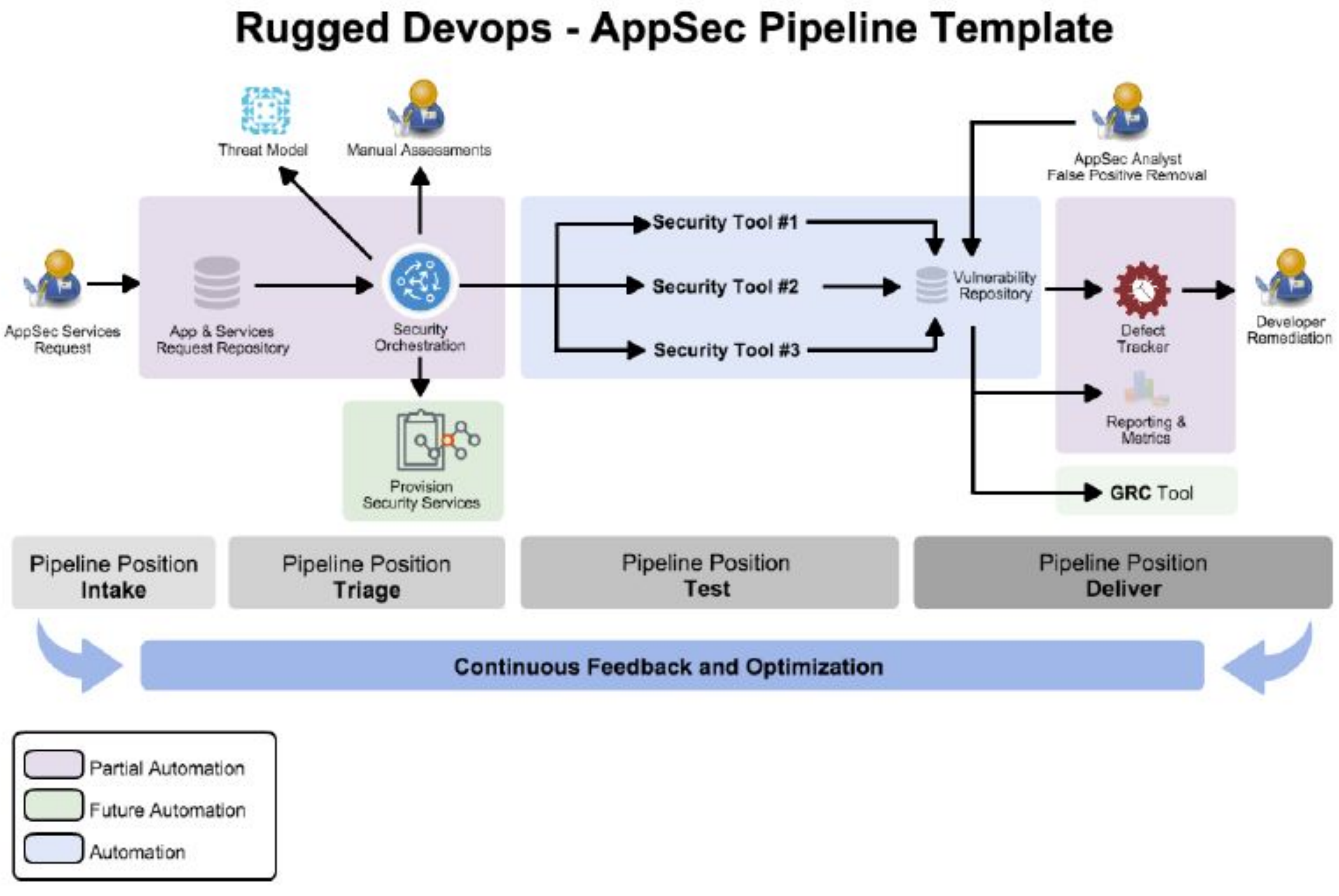
- Design a Service Mesh (ISTIO) architecture
- ISTIO side car proxy, baked-in security, with visibility across containers, by default, without any developer interaction or code change
- Benefits:
  - API Management, service discovery, authentication…
  - Dynamic request routing for A/B testing, gradual rollouts, canary releases, resilience, observability, retries, circuit breakers and fault injection
  - Layer 7 Load balancing
  - Zero Trust model: East/West Traffic Whitelisting, ACL, RBAC…
  - TLS encryption by default, Key management, signing…

Managing Microservices With Istio

Integrity - Service - Excellence

sonatype

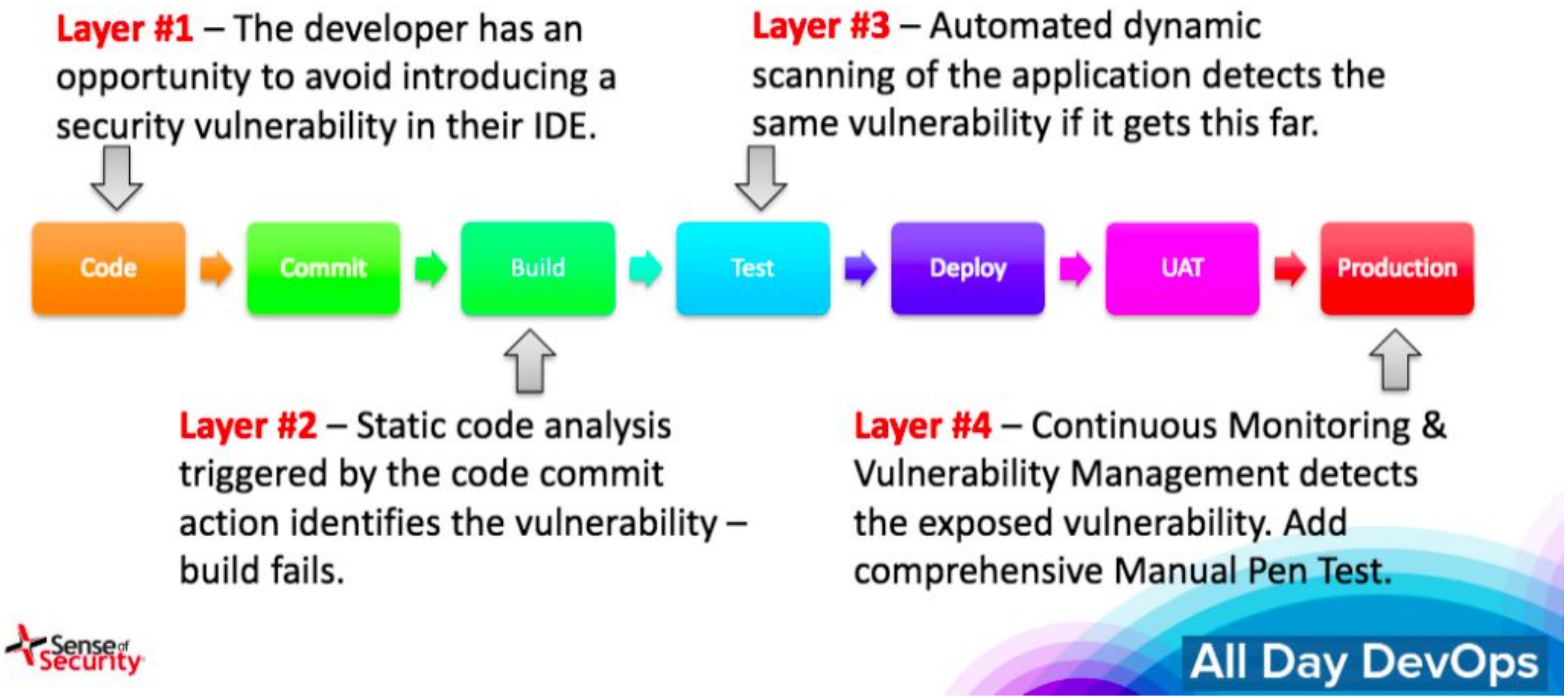
DevSecOps according to Aaron Weaver



Aaron Weaver, CC ShareAlike 3.0

sonatype

DevSecOps according to Murray Goldschmidt and Sense of Security



Layer #1 – The developer has an opportunity to avoid introducing a security vulnerability in their IDE.

Layer #3 – Automated dynamic scanning of the application detects the same vulnerability if it gets this far.

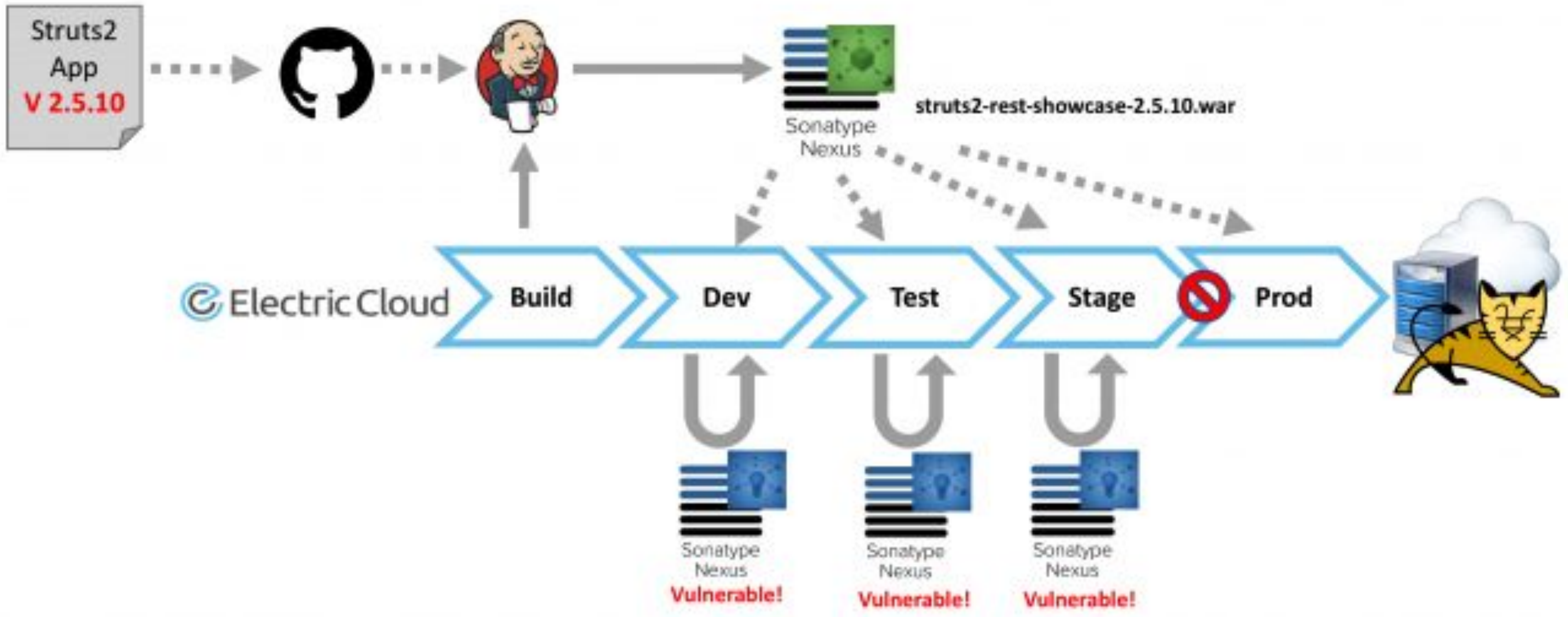
Layer #2 – Static code analysis triggered by the code commit action identifies the vulnerability – build fails.

Layer #4 – Continuous Monitoring & Vulnerability Management detects the exposed vulnerability. Add comprehensive Manual Pen Test.

sonatype

Source: ADDO 2017, YouTube – "DevOps: A How-To for Agility with Security: Murray Goldschmidt"

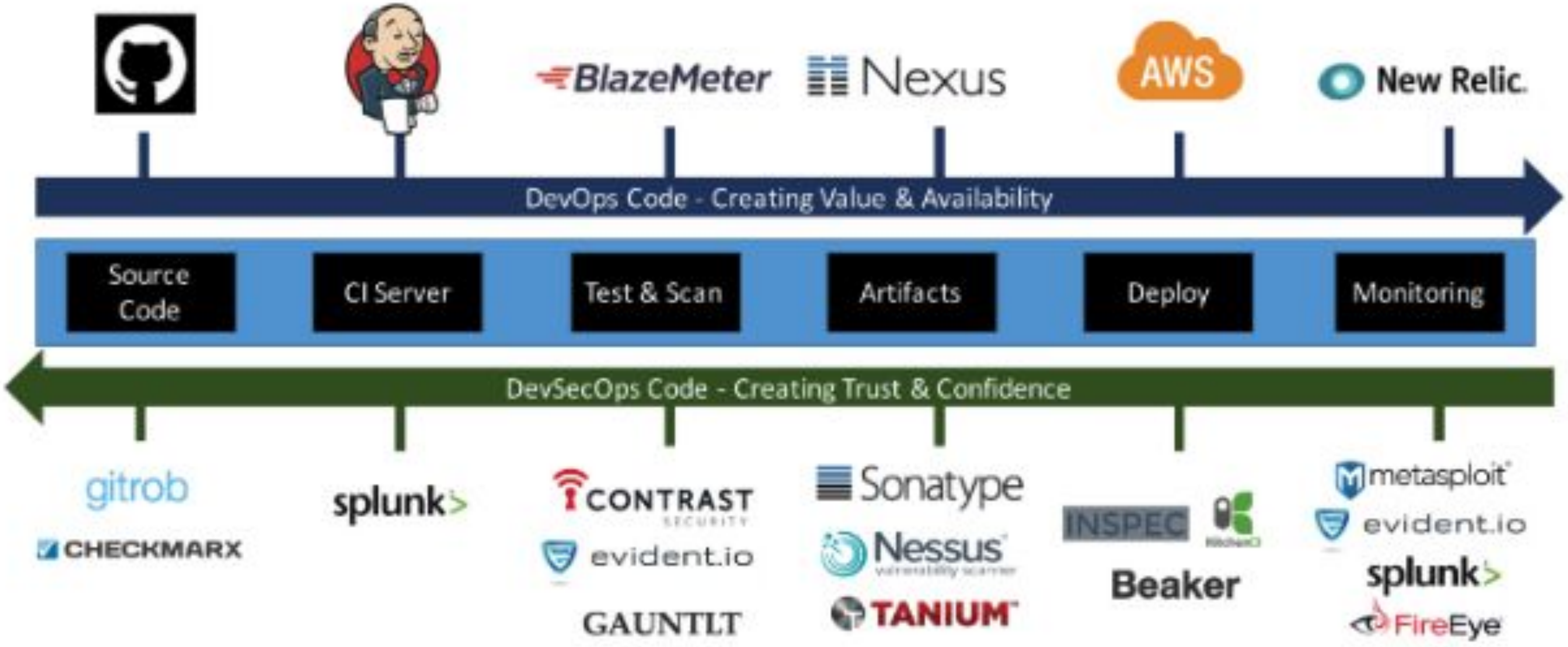
DevSecOps according to Hans Ashlock and Electric Cloud



Source: Hans Ashlock, Electric Cloud – “DevSecOps: How to Build Secure Pipelines and Prevent the Next Equifax”

sonatype

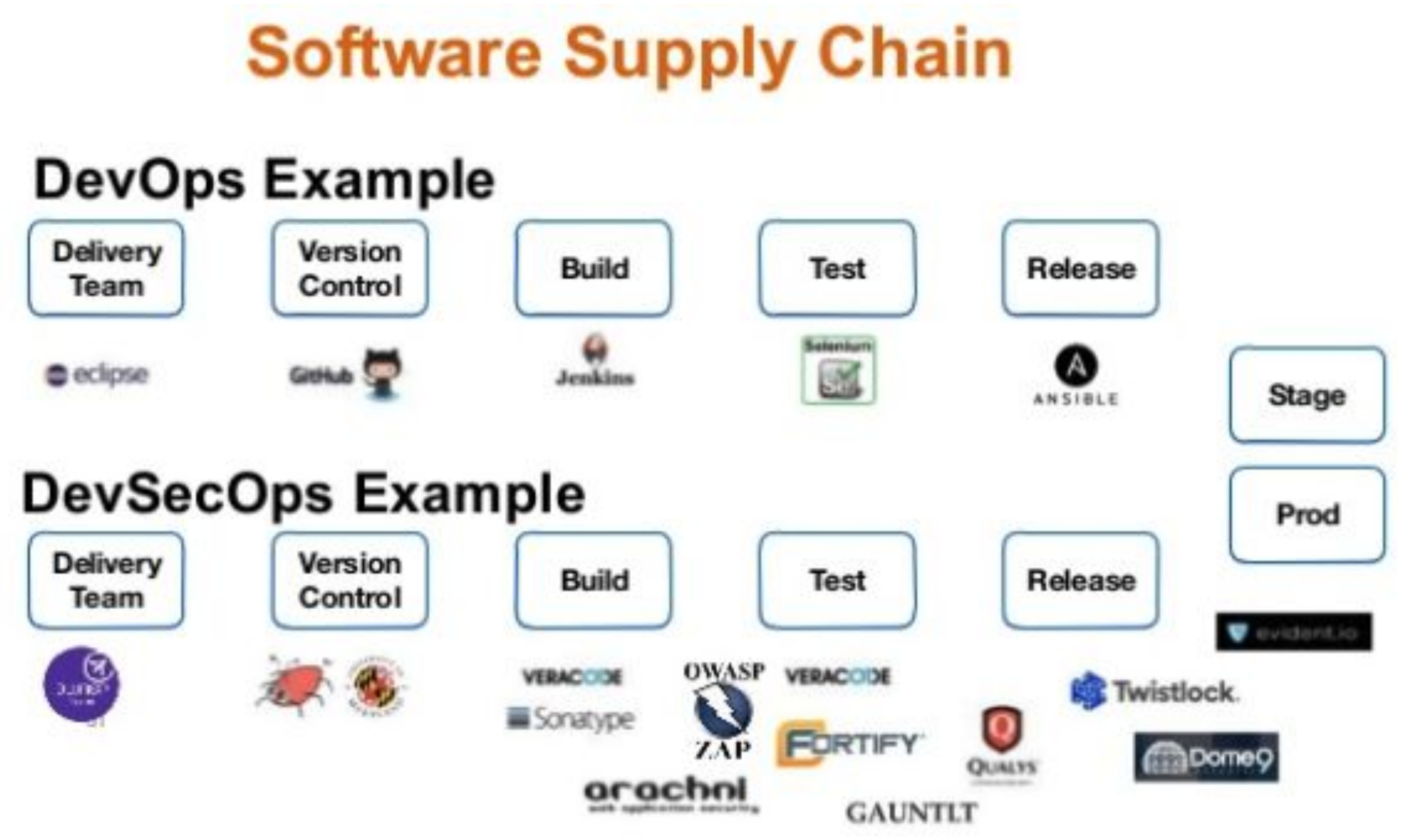
DevSecOps according to Shannon Lietz and Intuit



Source: Shannon Lietz, DevSecOps – “ Shifting Security to the Left”

sonatype

DevSecOps according to John Willis and Botchagalupe Technologies



Software Supply Chain

DevOps Example

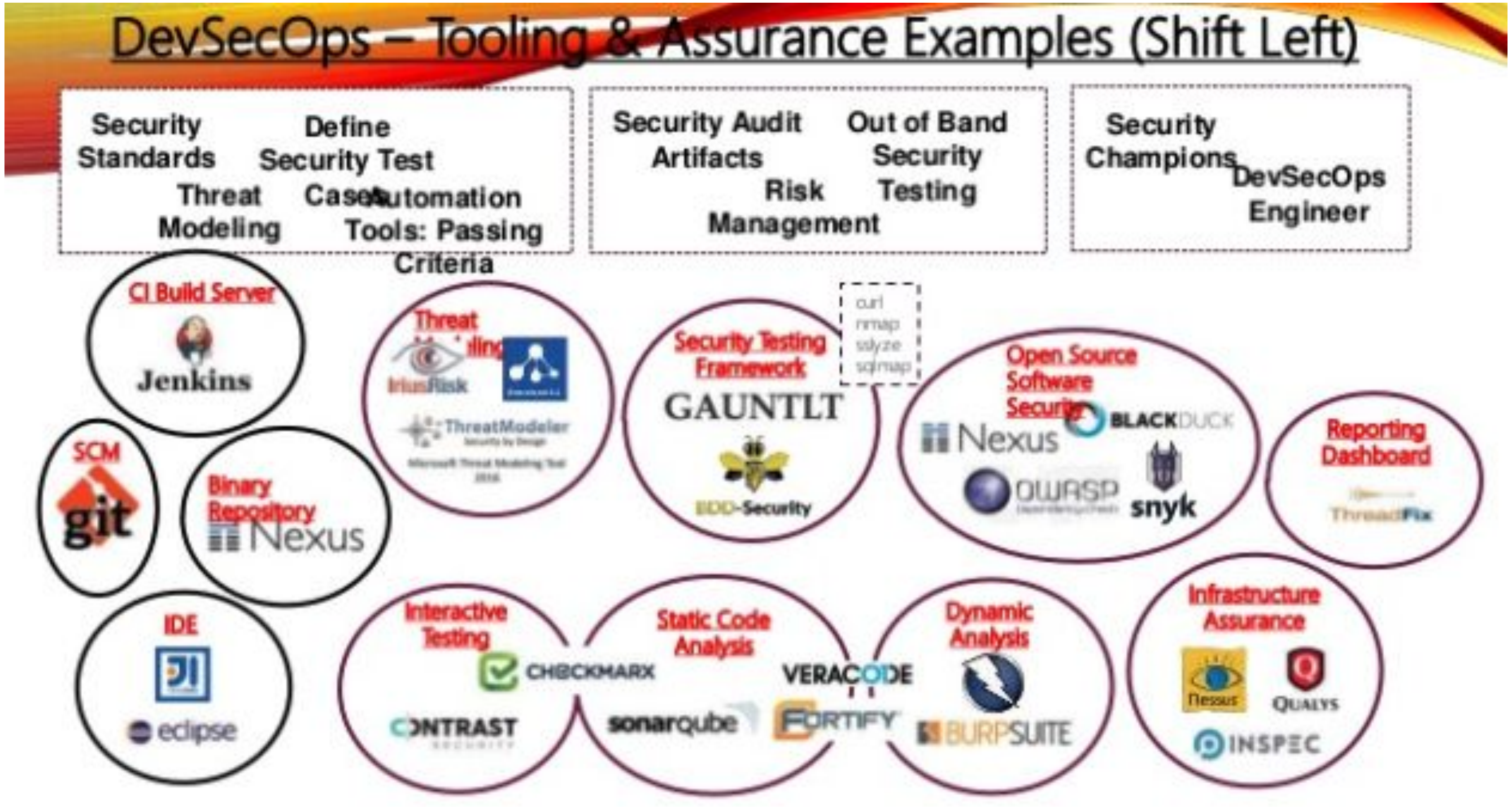
Delivery Team → Version Control → Build → Test → Release → Stage → Prod

DevSecOps Example

Delivery Team → Version Control → Build → Test → Release

Source: John Willis, LinkedIn Slideshare – “You Build It – Cyber Chicago Keynote”

DevSecOps according to Michael Man



DevSecOps – Tooling & Assurance Examples (Shift Left)

Security Standards

Define Security Test Cases

Threat Modeling

Automation Tools: Passing Criteria

Security Audit Artifacts

Risk Management

Out of Band Security Testing

Security Champions

DevSecOps Engineer

CI Build Server

SCM

Binary Repository

IDE

Threat Modelling

Security Testing Framework

Open Source Software Security

Reporting Dashboard

Interactive Testing

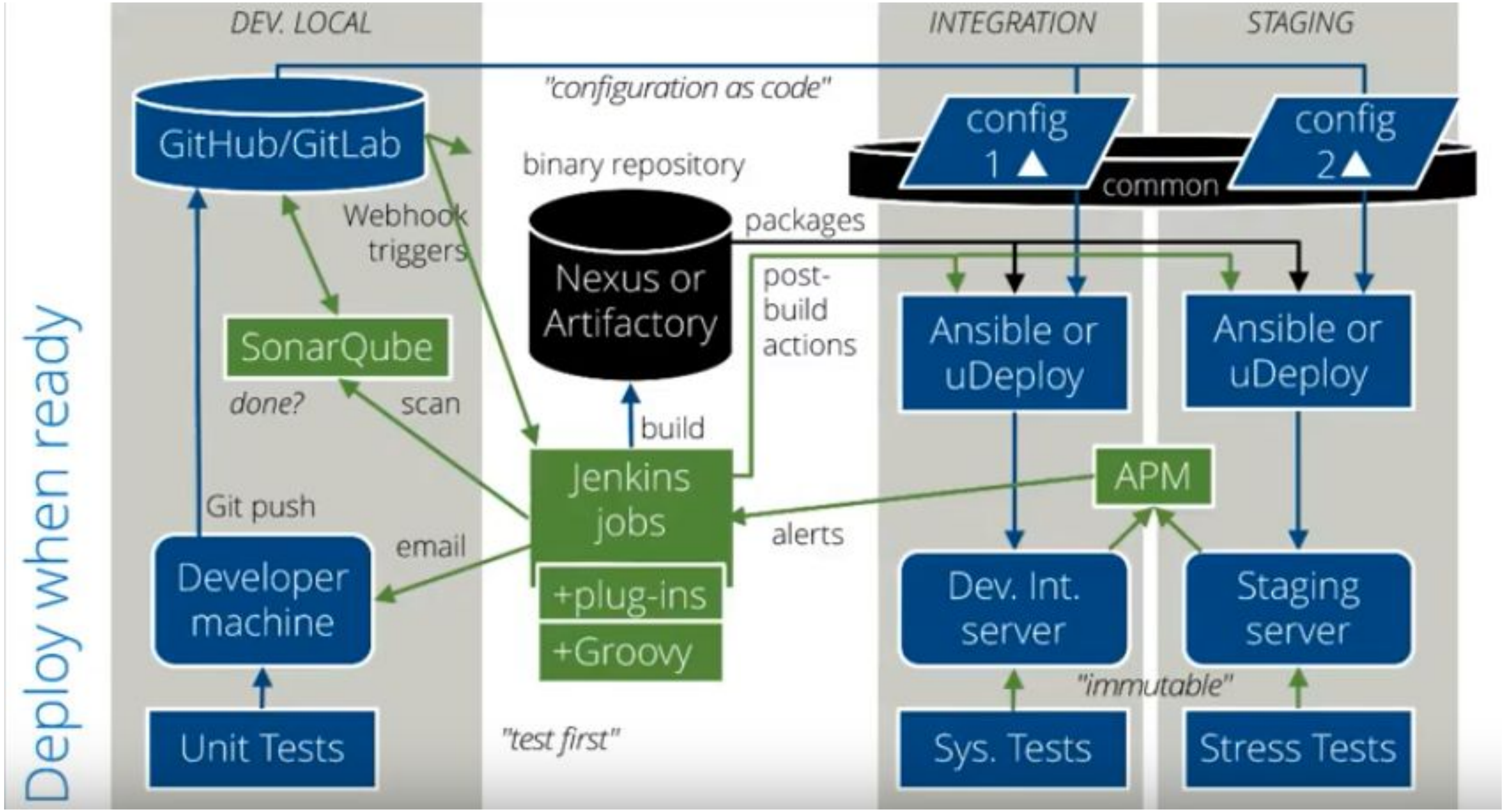
Static Code Analysis

Dynamic Analysis

Infrastructure Assurance

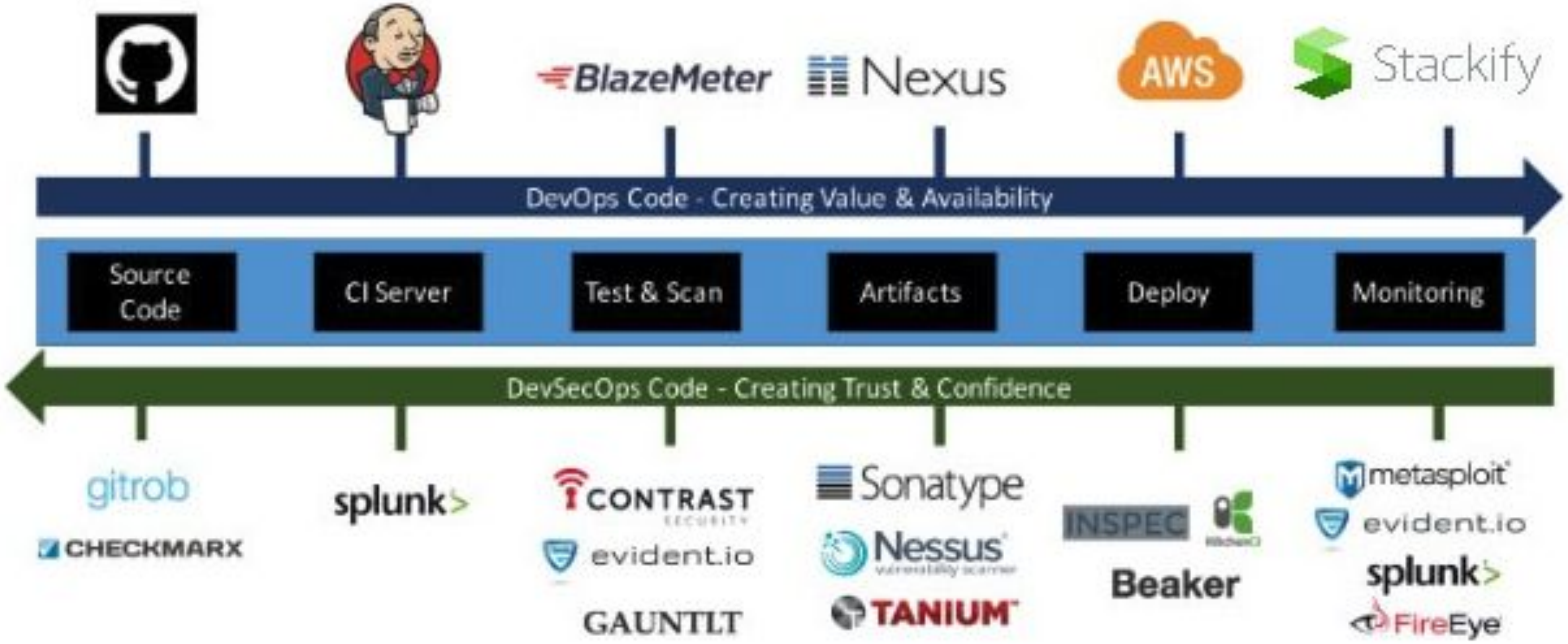
Source: Michael Man, LinkedIn SlideShare – “DevSecOps – London Gathering: June 2018”

DevSecOps according to Wilson Mar and JetBloom



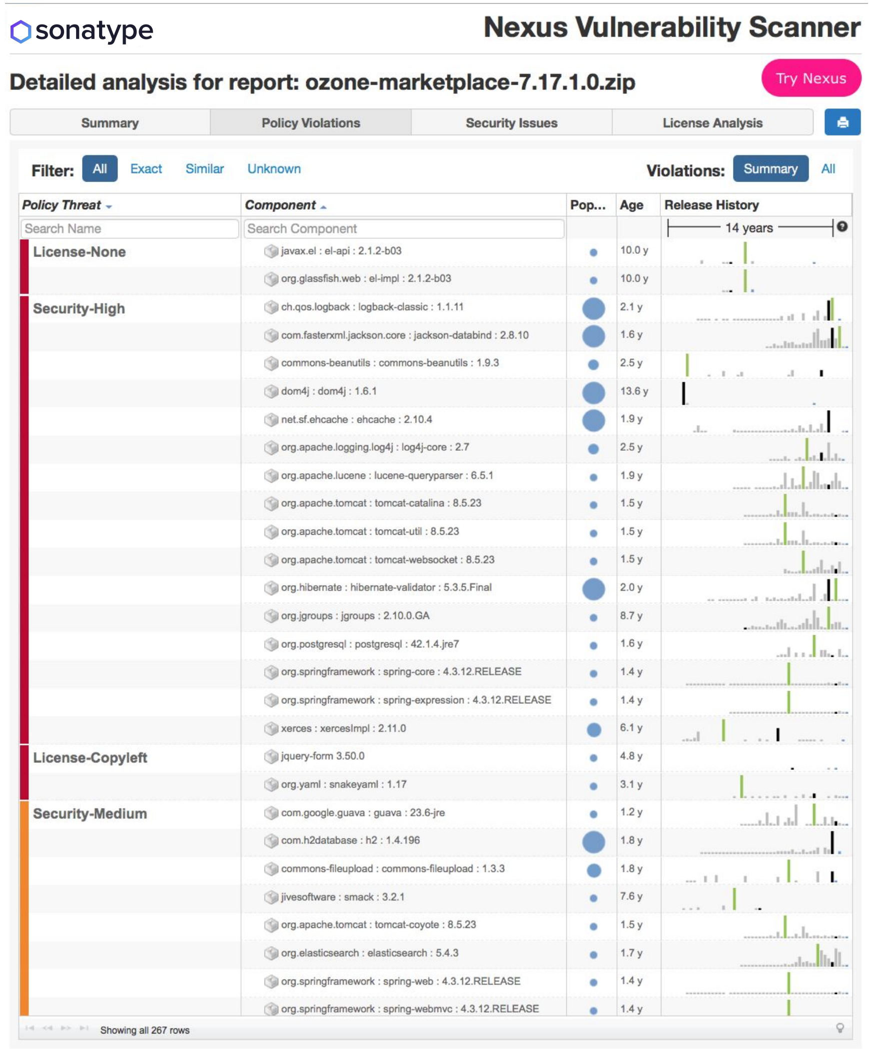
Source: Wilson Mar – Hands-On DevSecOps Course

DevSecOps according to Matt Watson and Stackify



Source: Matt Watson – “What is DevSecOps? How to Automate Security Testing”

Interested in DevSecOps, but don't know where to start?



sonatype **Nexus Vulnerability Scanner**

Detailed analysis for report: ozone-marketplace-7.17.1.0.zip

Policy Threat	*Component*	Pop...	Age	Release History
Search Name	Search Component			14 years
License-None	javax.el : el-api : 2.1.2-b03	●	10.0 y	
	org.glassfish.web : el-impl : 2.1.2-b03	●	10.0 y	
Security-High	ch.qos.logback : logback-classic : 1.1.11	●	2.1 y	
	com.fasterxml.jackson.core : jackson-databind : 2.8.10	●	1.6 y	
	commons-beanutils : commons-beanutils : 1.9.3	●	2.5 y	
	dom4j : dom4j : 1.6.1	●	13.6 y	
	net.sf.ehcache : ehcache : 2.10.4	●	1.9 y	
	org.apache.logging.log4j : log4j-core : 2.7	●	2.5 y	
	org.apache.lucene : lucene-queryparser : 6.5.1	●	1.9 y	
	org.apache.tomcat : tomcat-catalina : 8.5.23	●	1.5 y	
	org.apache.tomcat : tomcat-util : 8.5.23	●	1.5 y	
	org.apache.tomcat : tomcat-websocket : 8.5.23	●	1.5 y	
	org.hibernate : hibernate-validator : 5.3.5.Final	●	2.0 y	
	org.jgroups : jgroups : 2.10.0.GA	●	8.7 y	
	org.postgresql : postgresql : 42.1.4.jre7	●	1.6 y	
	org.springframework : spring-core : 4.3.12.RELEASE	●	1.4 y	
	org.springframework : spring-expression : 4.3.12.RELEASE	●	1.4 y	
	xerces : xercesImpl : 2.11.0	●	6.1 y	
License-Copyleft	jquery-form 3.50.0	●	4.8 y	
	org.yaml : snakeyaml : 1.17	●	3.1 y	
Security-Medium	com.google.guava : guava : 23.6-jre	●	1.2 y	
	com.h2database : h2 : 1.4.196	●	1.8 y	
	commons-fileupload : commons-fileupload : 1.3.3	●	1.8 y	
	jivesoftware : smack : 3.2.1	●	7.6 y	
	org.apache.tomcat : tomcat-coyote : 8.5.23	●	1.5 y	
	org.elasticsearch : elasticsearch : 5.4.3	●	1.7 y	
	org.springframework : spring-web : 4.3.12.RELEASE	●	1.4 y	
	org.springframework : spring-webmvc : 4.3.12.RELEASE	●	1.4 y	

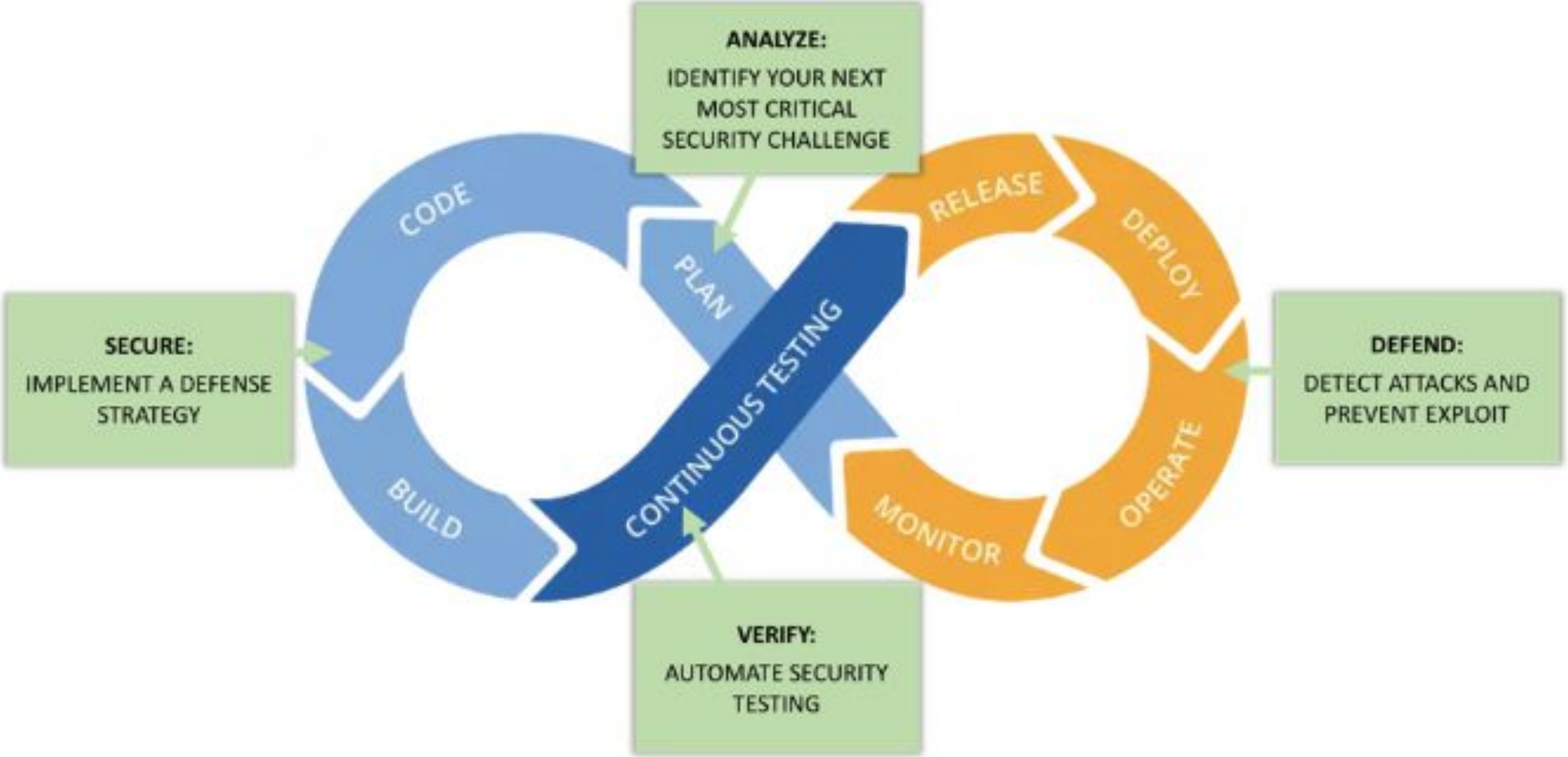
Showing all 267 rows

Try Nexus Vulnerability Scanner:

1. Confidently and quickly analyze your open source and third party components
2. Create a precise "Bill of Materials" to identify which open source components are used and where.
3. Discover all component dependencies and known vulnerabilities or license risks.

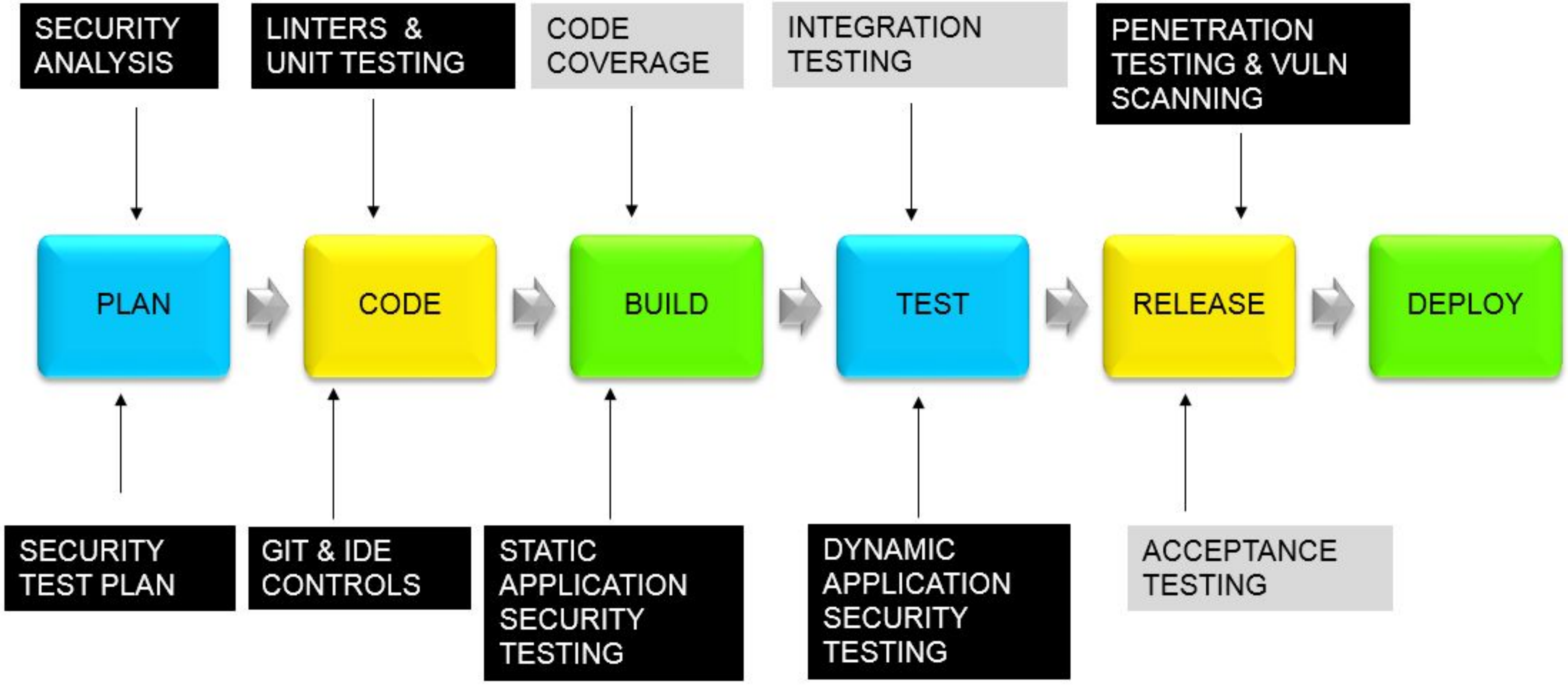
sonatype

DevSecOps according to Jeff Williams and Contrast Security


ANALYZE:
IDENTIFY YOUR NEXT MOST CRITICAL SECURITY CHALLENGE
SECURE:
IMPLEMENT A DEFENSE STRATEGY
CODE
PLAN
RELEASE
DEPLOY
DEFEND:
DETECT ATTACKS AND PREVENT EXPLOIT
BUILD
CONTINUOUS TESTING
MONITOR
OPERATE
VERIFY:
AUTOMATE SECURITY TESTING

Source: Jeff Williams, DZone Refcard #267– “Introduction to DevSecOps”

DevSecOps according to Tom Porter and HPE/DXC



SECURITY ANALYSIS

LINTERS & UNIT TESTING

CODE COVERAGE

INTEGRATION TESTING

PENETRATION TESTING & VULN SCANNING

PLAN

CODE

BUILD

TEST

RELEASE

DEPLOY

SECURITY TEST PLAN

GIT & IDE CONTROLS

STATIC APPLICATION SECURITY TESTING

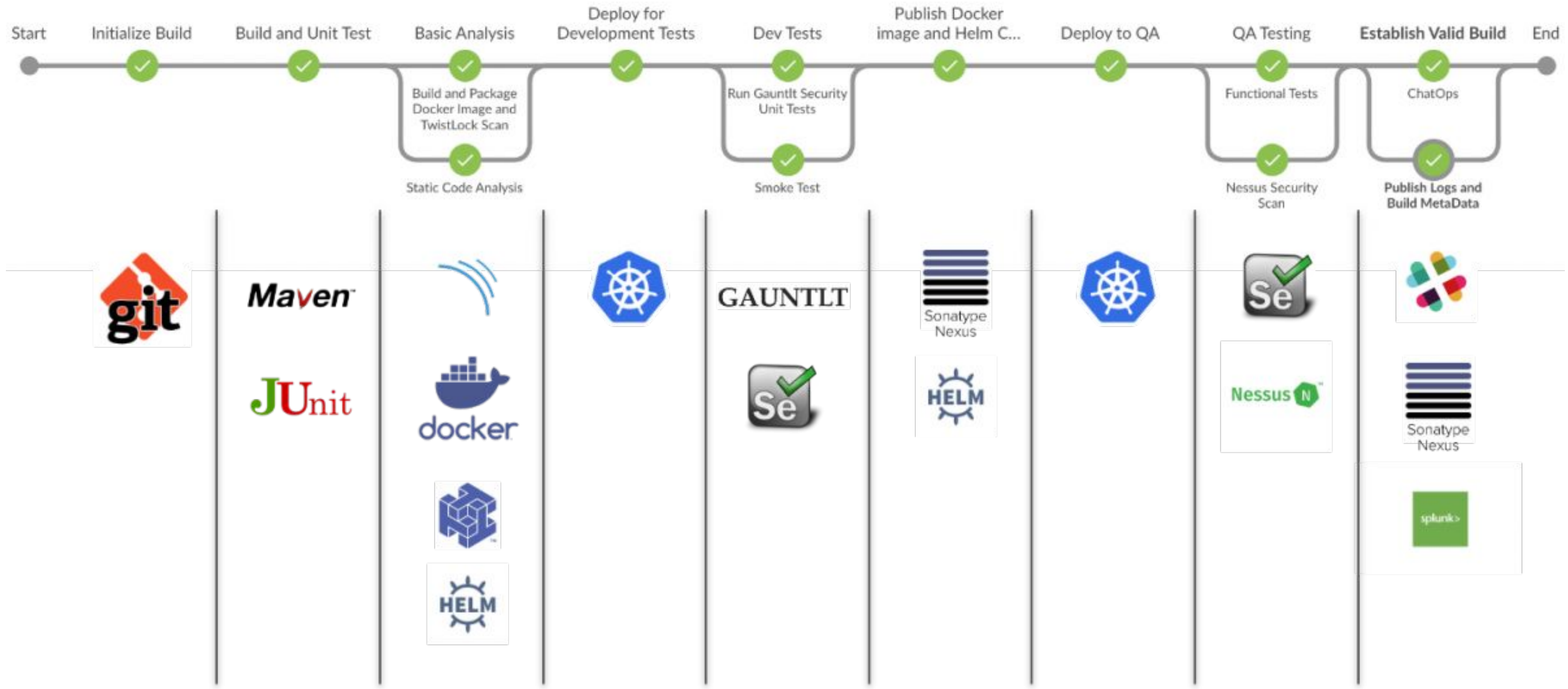
DYNAMIC APPLICATION SECURITY TESTING

ACCEPTANCE TESTING

Source: Tom Porter, DZone – “DevSecOps – A New Chance for Security”

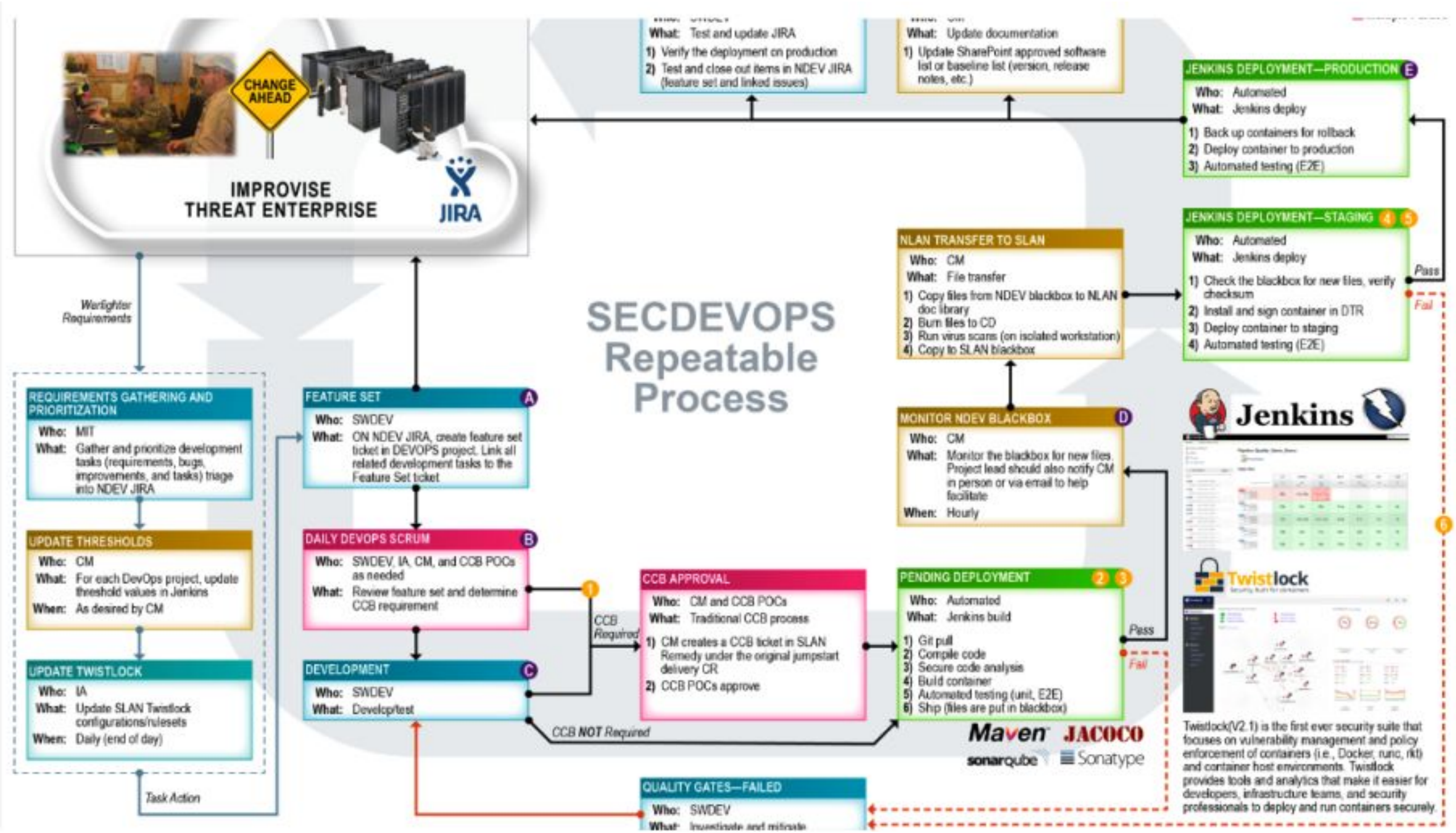
sonatype

DevSecOps according to Ben Chicoski and CloudBees



Source: Ben Chicoski, CloudBees – “Orchestrating DevSecOps: Security at Speed”

DevSecOps according to Leonel Garciga and U.S. Dept of Defense/JIDO (circa 2017)



What: Test and update JIRA
1) Verify the deployment on production
2) Test and close out items in NDEV JIRA (feature set and linked issues)

What: Update documentation
1) Update SharePoint approved software list or baseline list (version, release notes, etc.)

JENKINS DEPLOYMENT—PRODUCTION
Who: Automated
What: Jenkins deploy
1) Back up containers for rollback
2) Deploy container to production
3) Automated testing (E2E)

JENKINS DEPLOYMENT—STAGING
Who: Automated
What: Jenkins deploy
1) Check the blackbox for new files, verify checksum
2) Install and sign container in DTR
3) Deploy container to staging
4) Automated testing (E2E)

NLAN TRANSFER TO SLAN
Who: CM
What: File transfer
1) Copy files from NDEV blackbox to NLAN doc library
2) Burn files to CD
3) Run virus scans (on isolated workstation)
4) Copy to SLAN blackbox

MONITOR NDEV BLACKBOX
Who: CM
What: Monitor the blackbox for new files. Project lead should also notify CM in person or via email to help facilitate
When: Hourly

REQUIREMENTS GATHERING AND PRIORITIZATION
Who: MIT
What: Gather and prioritize development tasks (requirements, bugs, improvements, and tasks) triage into NDEV JIRA

UPDATE THRESHOLDS
Who: CM
What: For each DevOps project, update threshold values in Jenkins
When: As desired by CM

UPDATE TWISTLOCK
Who: IA
What: Update SLAN Twistlock configurations/rulesets
When: Daily (end of day)

FEATURE SET
Who: SWDEV
What: ON NDEV JIRA, create feature set ticket in DEVOPS project. Link all related development tasks to the Feature Set ticket

DAILY DEVOPS SCRUM
Who: SWDEV, IA, CM, and CCB POCs as needed
What: Review feature set and determine CCB requirement

DEVELOPMENT
Who: SWDEV
What: Develop/test

CCB APPROVAL
Who: CM and CCB POCs
What: Traditional CCB process
1) CM creates a CCB ticket in SLAN Remedy under the original jumpstart delivery CR
2) CCB POCs approve

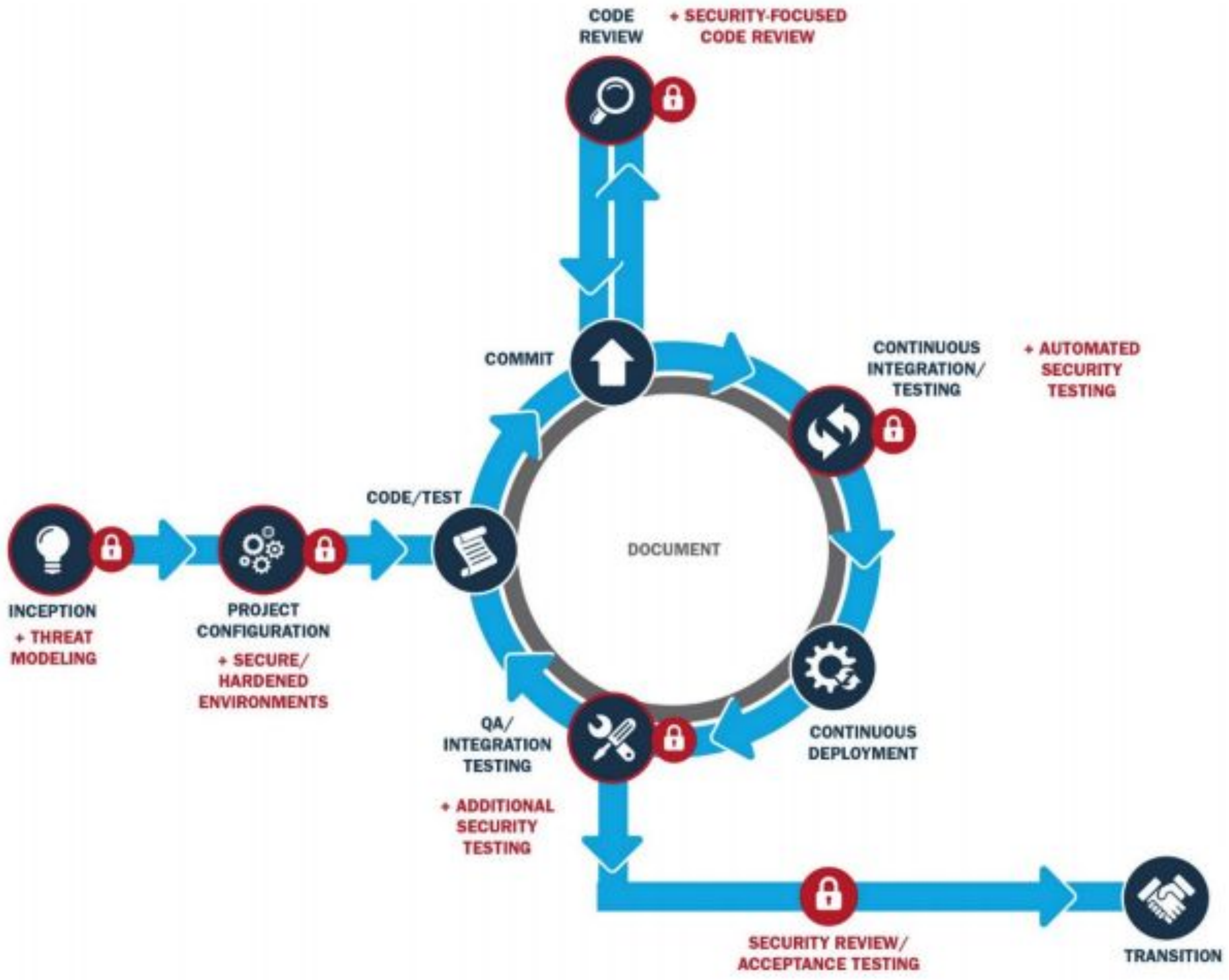
PENDING DEPLOYMENT
Who: Automated
What: Jenkins build
1) Git pull
2) Compile code
3) Secure code analysis
4) Build container
5) Automated testing (unit, E2E)
6) Ship (files are put in blackbox)

QUALITY GATES—FAILED
Who: SWDEV
What: Investigate and mitigate

Twistlock(V2.1) is the first ever security suite that focuses on vulnerability management and policy enforcement of containers (i.e., Docker, runc, rkt) and container host environments. Twistlock provides tools and analytics that make it easier for developers, infrastructure teams, and security professionals to deploy and run containers securely.

Source: ADDO 2017, YouTube – "Governance and Transparency in GovSec DevOps: Leonel Garciga"

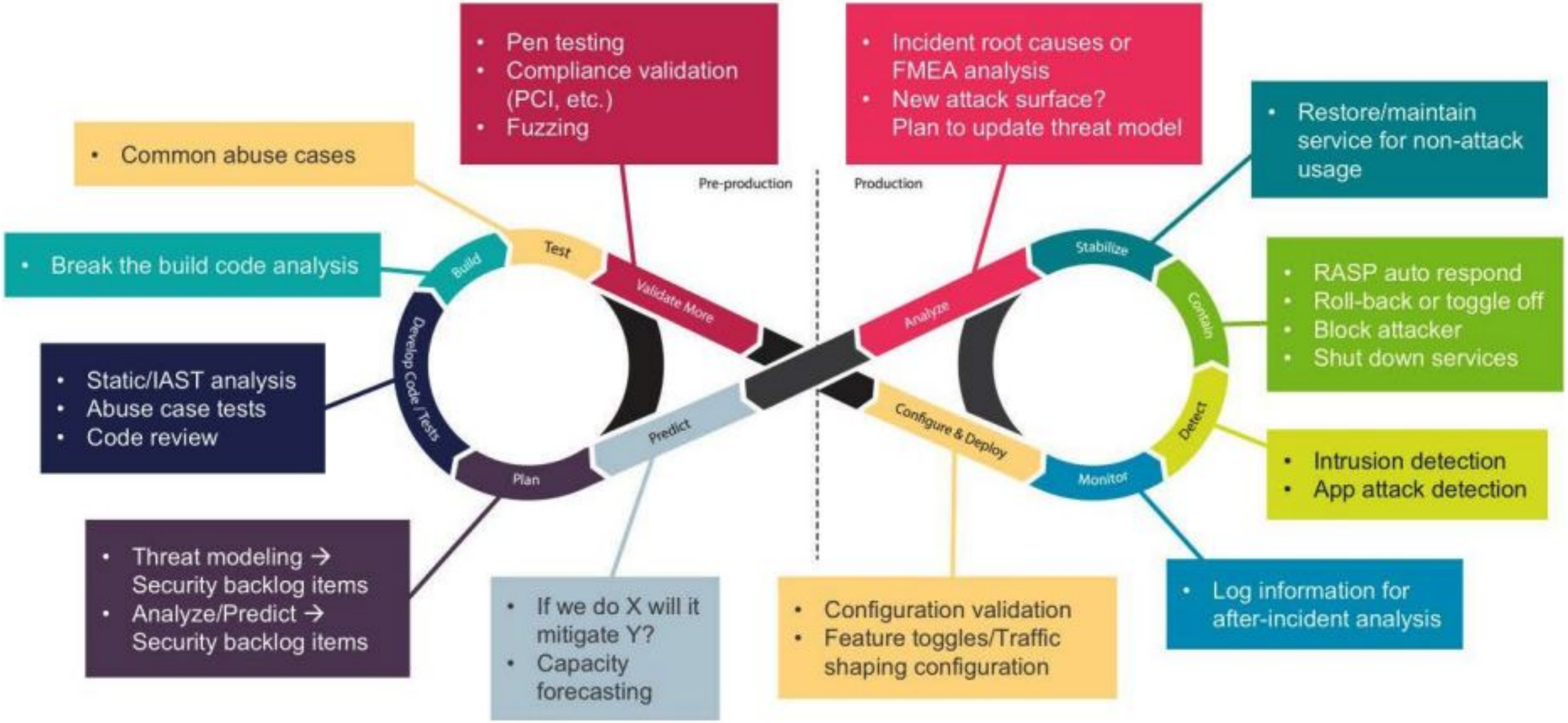
DevSecOps according to Hasan Yasar and Carnegie Mellon SEI


CODE REVIEW
+ SECURITY-FOCUSED CODE REVIEW
COMMIT
CONTINUOUS INTEGRATION/ TESTING
+ AUTOMATED SECURITY TESTING
CODE/TEST
DOCUMENT
INCEPTION
+ THREAT MODELING
PROJECT CONFIGURATION
+ SECURE/ HARDENED ENVIRONMENTS
QA/ INTEGRATION TESTING
+ ADDITIONAL SECURITY TESTING
CONTINUOUS DEPLOYMENT
SECURITY REVIEW/ ACCEPTANCE TESTING
TRANSITION

Source: Derek Weeks, DZone – “From Water-Scrum-Fall to DevSecOps”

sonatype

DevSecOps according to Larry Maccherone and Comcast



- Common abuse cases

- Pen testing
- Compliance validation (PCI, etc.)
- Fuzzing

- Incident root causes or FMEA analysis
- New attack surface? Plan to update threat model

- Restore/maintain service for non-attack usage

- Break the build code analysis

- RASP auto respond
- Roll-back or toggle off
- Block attacker
- Shut down services

- Static/IAST analysis
- Abuse case tests
- Code review

- Intrusion detection
- App attack detection

- Threat modeling → Security backlog items
- Analyze/Predict → Security backlog items

- If we do X will it mitigate Y?
- Capacity forecasting

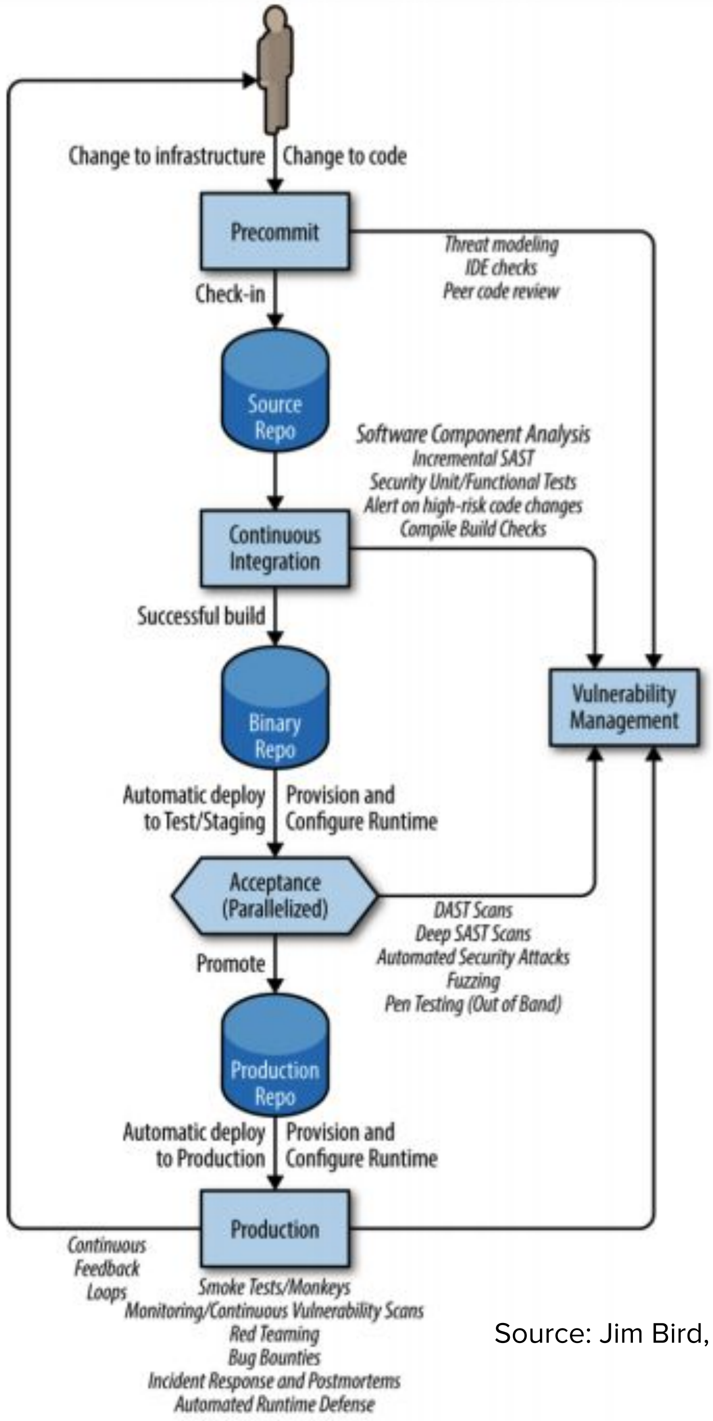
- Configuration validation
- Feature toggles/Traffic shaping configuration

- Log information for after-incident analysis

sonatype

Source: Larry Maccherone (@Lmaccherone), Twitter – “Annotated DevSecOps Cycle”

DevSecOps according to Jim Bird


Change to infrastructure
Change to code
Precommit
Threat modeling
IDE checks
Peer code review
Check-in
Source Repo
Software Component Analysis
Incremental SAST
Security Unit/Functional Tests
Alert on high-risk code changes
Compile Build Checks
Continuous Integration
Successful build
Binary Repo
Vulnerability Management
Automatic deploy to Test/Staging
Provision and Configure Runtime
Acceptance (Parallelized)
DAST Scans
Deep SAST Scans
Automated Security Attacks
Fuzzing
Pen Testing (Out of Band)
Promote
Production Repo
Automatic deploy to Production
Provision and Configure Runtime
Production
Continuous Feedback Loops
Smoke Tests/Monkeys
Monitoring/Continuous Vulnerability Scans
Red Teaming
Bug Bounties
Incident Response and Postmortems
Automated Runtime Defense

Source: Jim Bird, O'Reilly – "DevOps Sec: Securing Software Through Continuous Delivery"

DevSecOps according to YOU

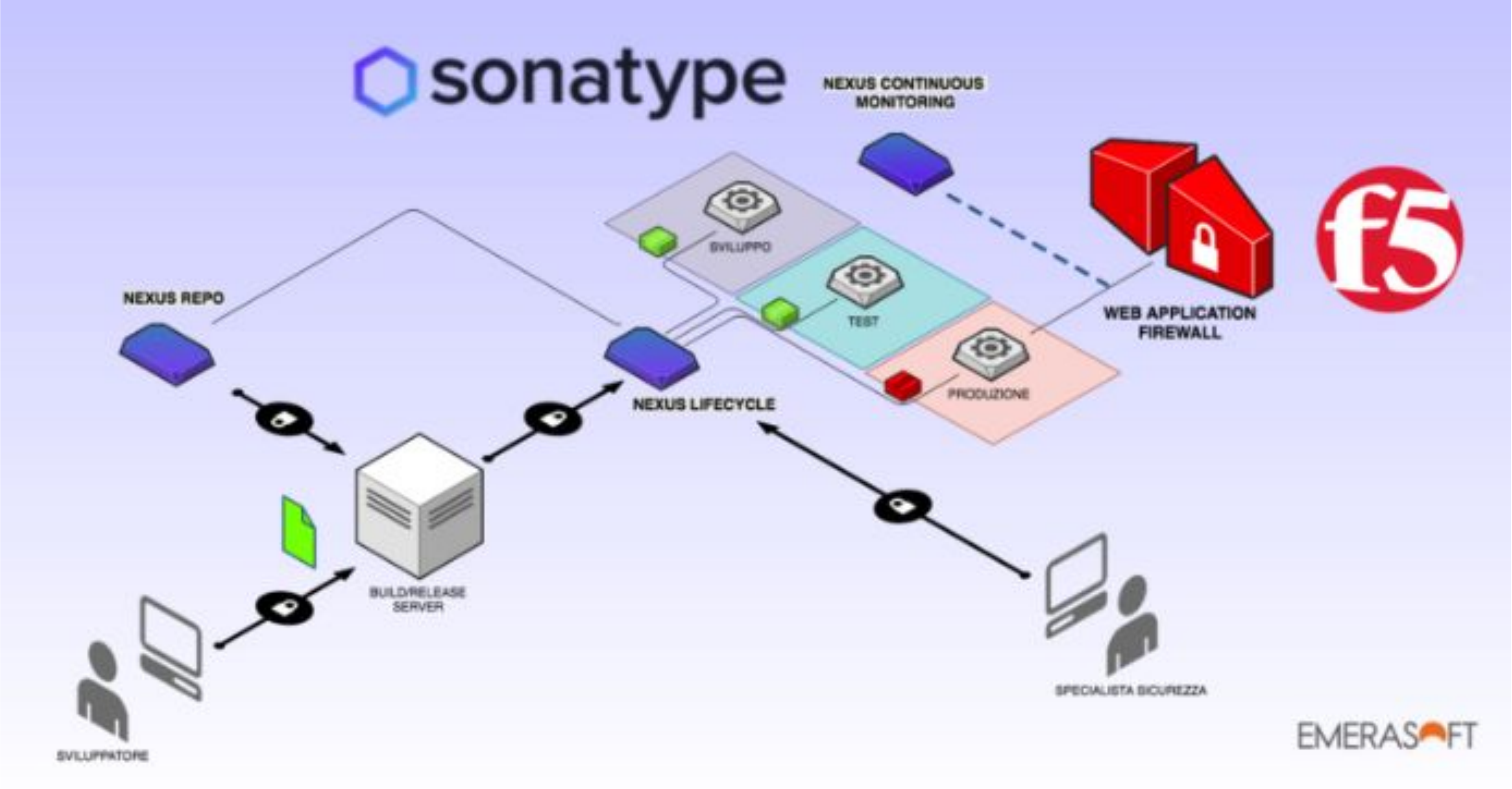
Want your DevSecOps Reference Architecture to this deck?

1. Send it to community@sonatype.com with the subject line: DevSecOps Reference Architecture
2. Provide a link as to where people can find more info about it (e.g., blog, video, SlideShare)
3. We'll add it to this deck with full attribution to you

It's that easy; we all learn with help from the community. Thank you in advance for your contributions!

sonatype

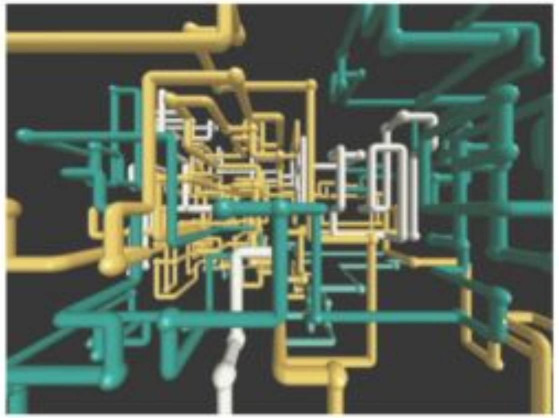
DevSecOps according to Ugo Cirací and Emerasoft

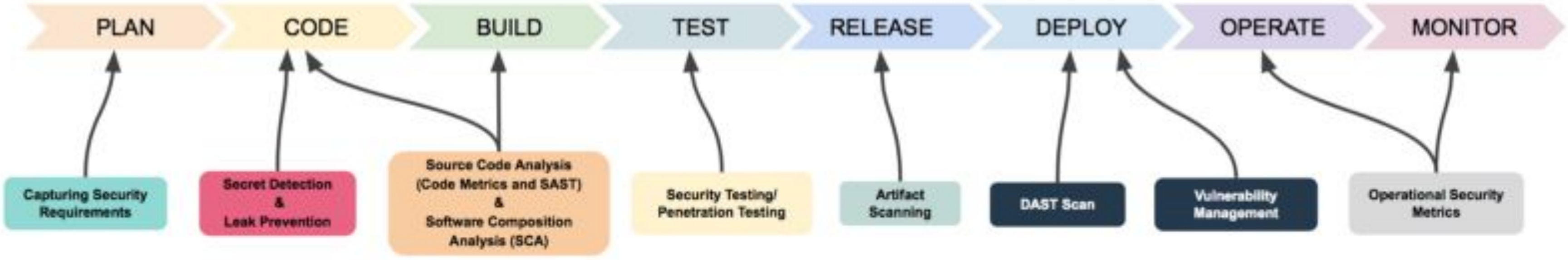


Source: Ugo Cirací, Emerasoft, Medium – “DevSecOps at Emerasoft: Sonatype Nexus Lifecycle and F5-Advanced WAF”

DevSecOps according to Ashish Rajan and Versent

CI/CD Pipeline

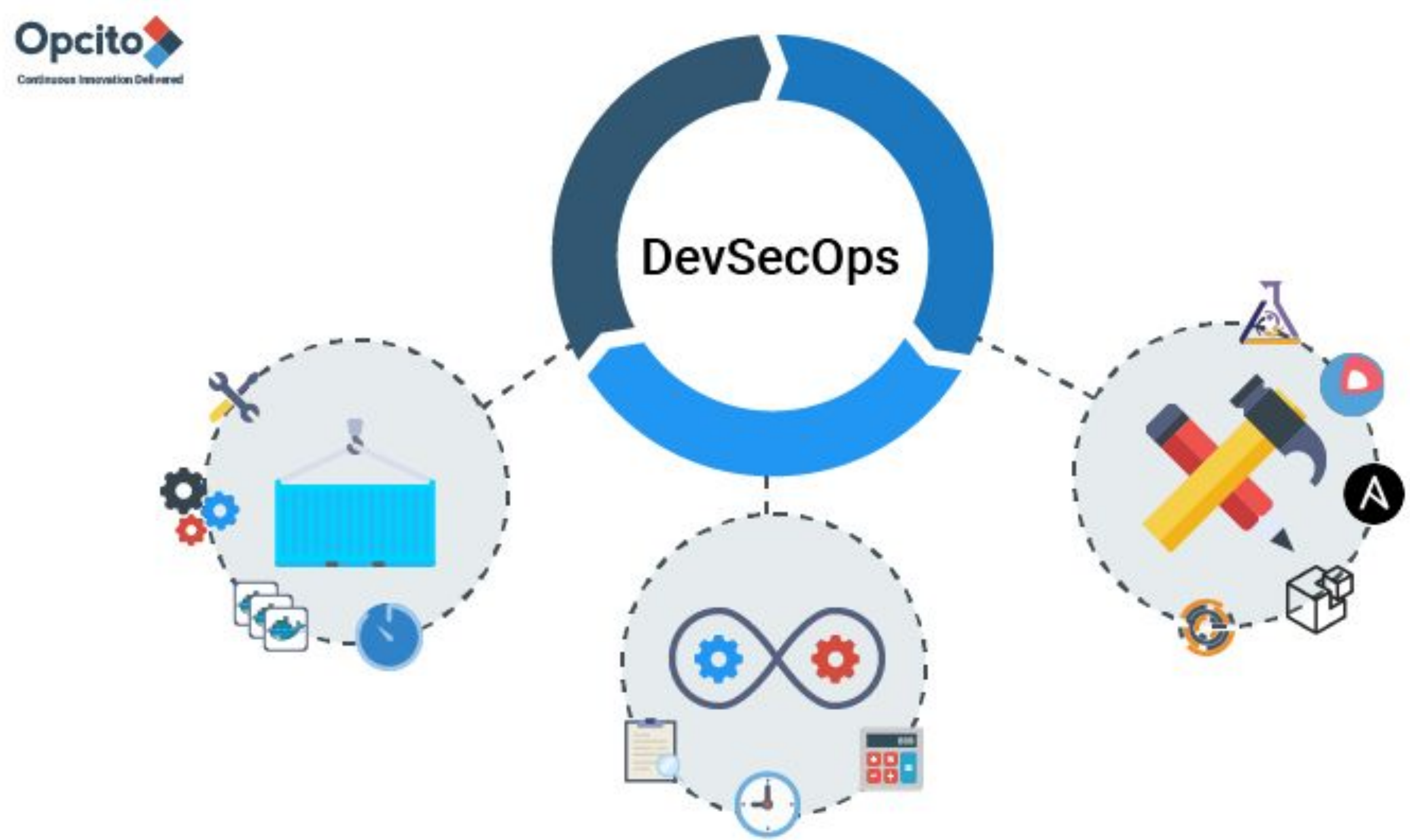



PLAN
CODE
BUILD
TEST
RELEASE
DEPLOY
OPERATE
MONITOR
Capturing Security Requirements
Secret Detection & Leak Prevention
Source Code Analysis (Code Metrics and SAST) & Software Composition Analysis (SCA)
Security Testing/ Penetration Testing
Artifact Scanning
DAST Scan
Vulnerability Management
Operational Security Metrics

Source: Ashish Rajan, Medium – "DevSecOps Melbourne Meetup S01E06 & Event Update"

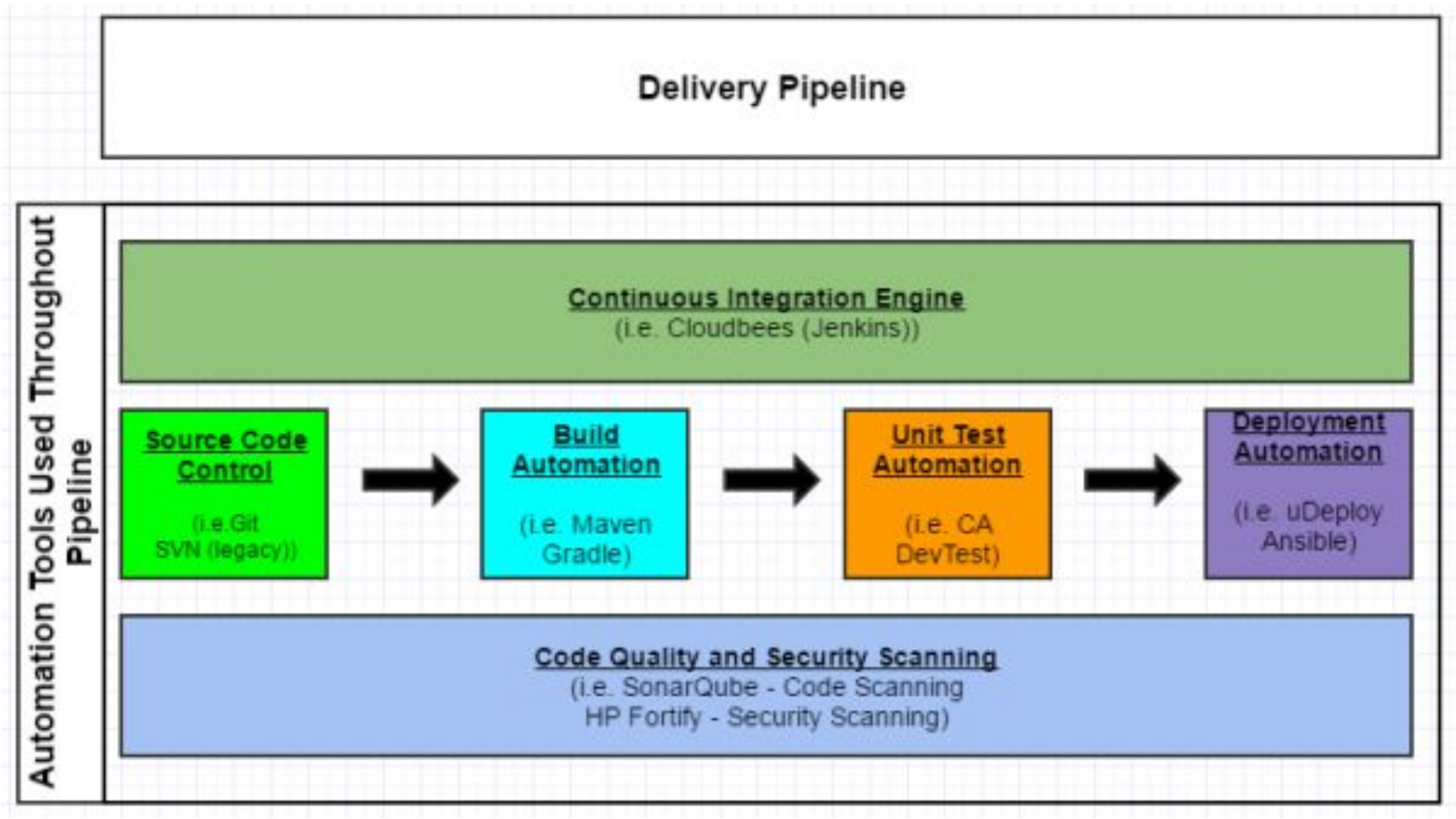
sonatype

DevSecOps according to Chaitanya Jawale and Opcito



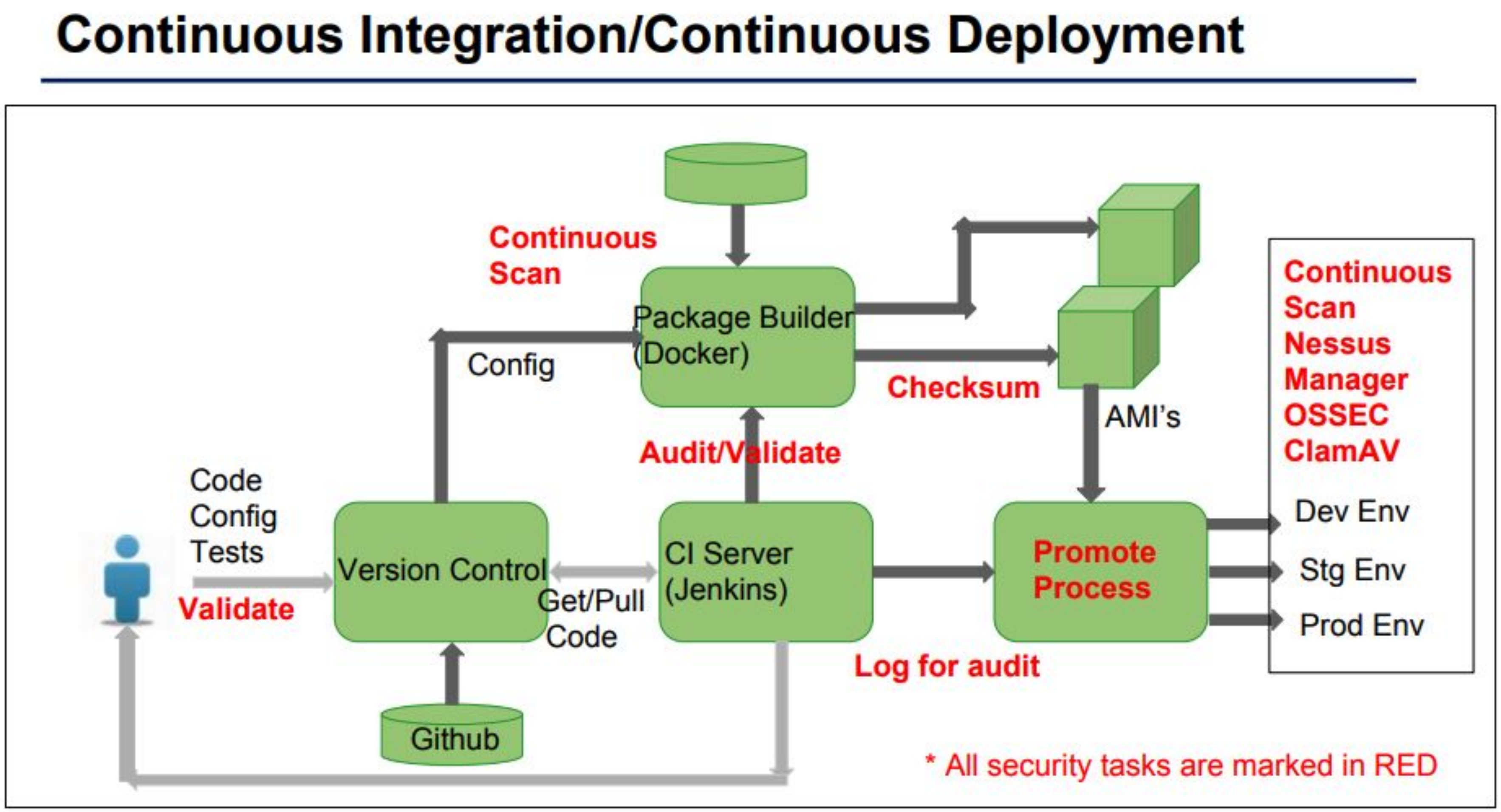
Source: Chaitanya Jawale, Opcito – “From the CEO’s Desk: DevSecOps – Next Stride for DevOps”

DevSecOps according to Seth Gagnon and Cigna


Delivery Pipeline
Automation Tools Used Throughout Pipeline
Continuous Integration Engine
(i.e. Cloudbees (Jenkins))
Source Code Control
(i.e.Git
SVN (legacy))
Build Automation
(i.e. Maven
Gradle)
Unit Test Automation
(i.e. CA
DevTest)
Deployment Automation
(i.e. uDeploy
Ansible)
Code Quality and Security Scanning
(i.e. SonarQube - Code Scanning
HP Fortify - Security Scanning)

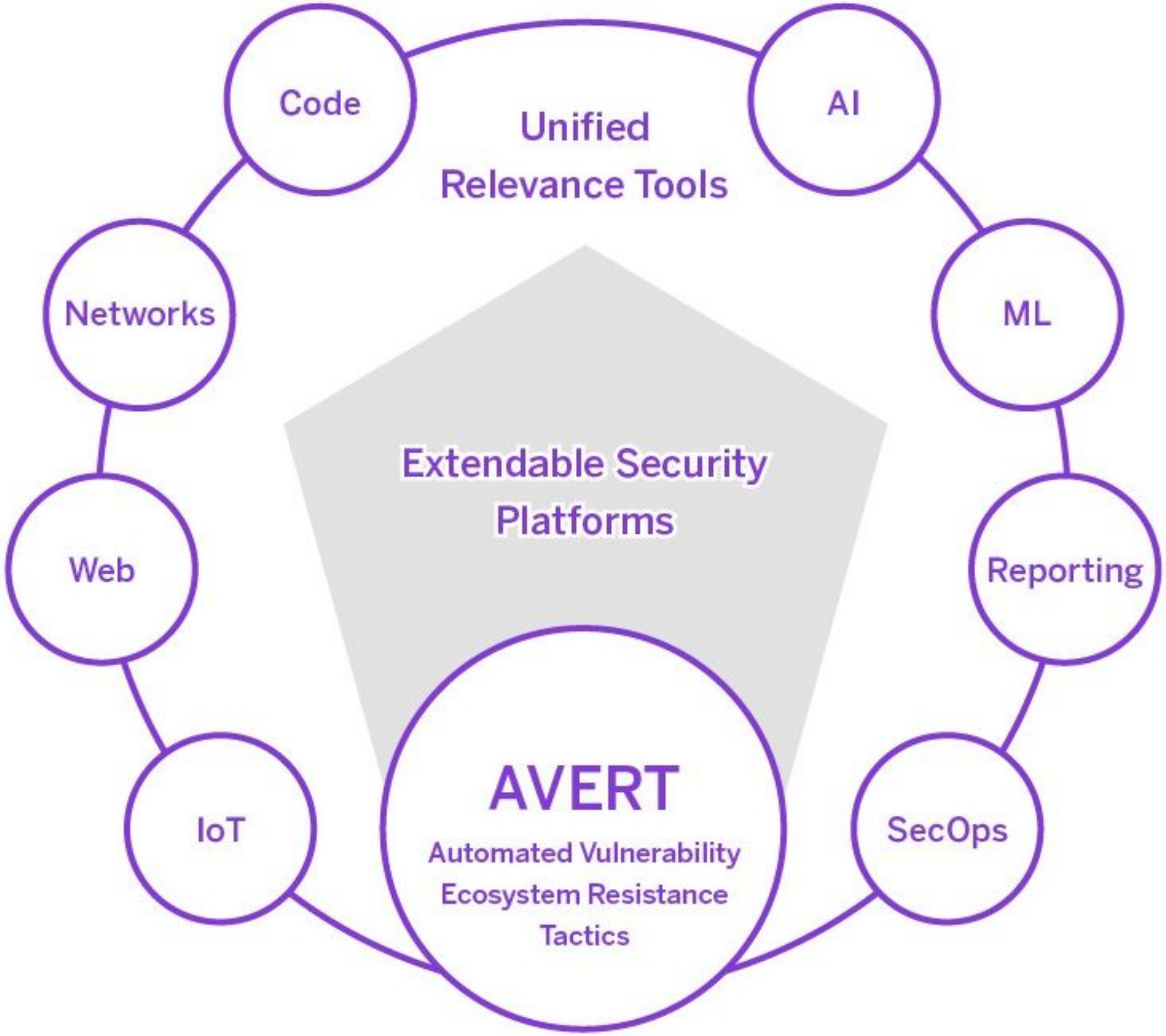
Source: Seth Gagnon, Dzone – "An Example of a Continuous Delivery Pipeline"

DevSecOps according to GSA



Source: GSA Slidedeck – "Implementation of DevSecOps for D2D"

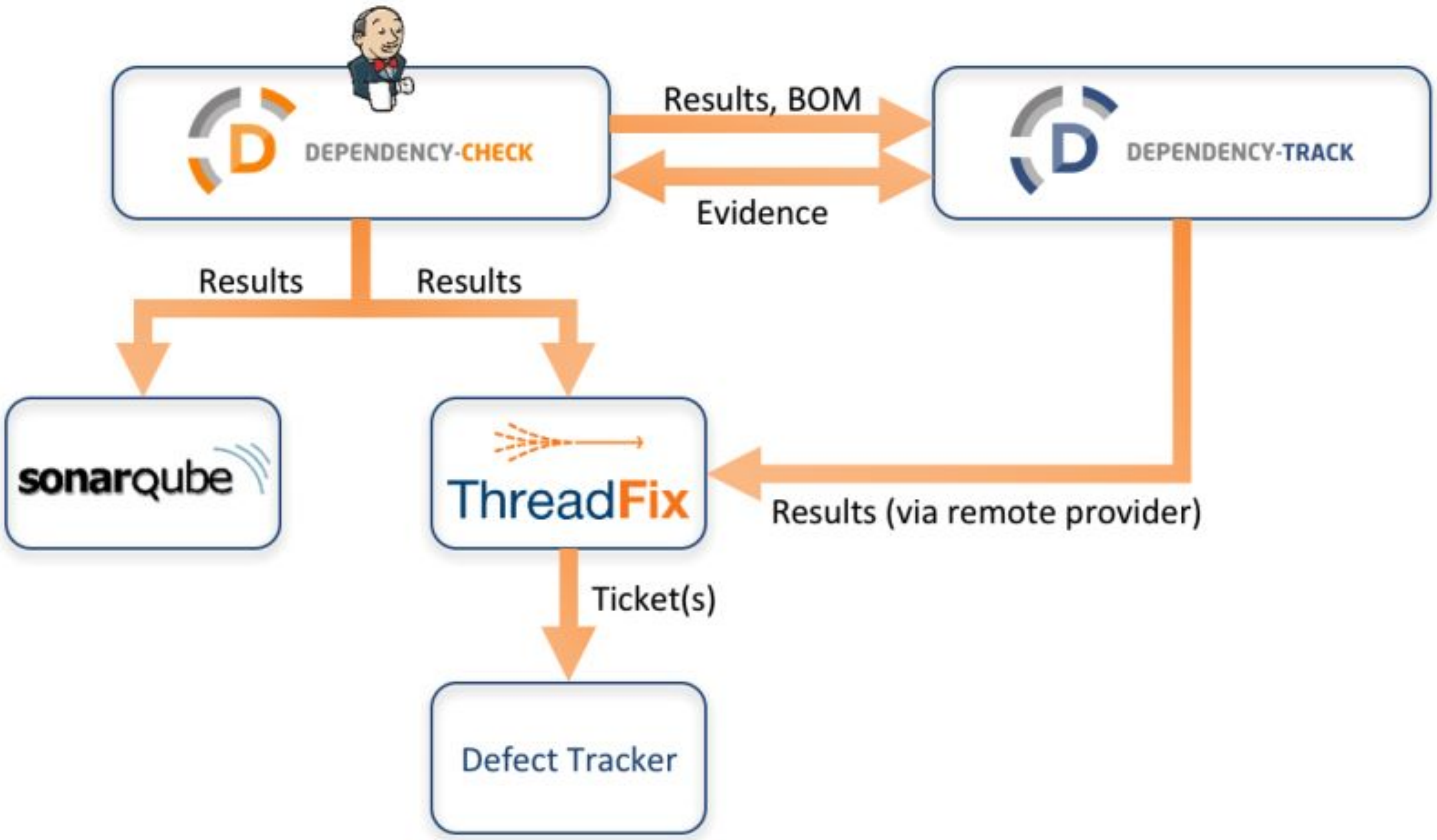
DevSecOps according to Atul Jadhav and Aricent


Code
Unified Relevance Tools
AI
Networks
ML
Extendable Security Platforms
Web
Reporting
IoT
AVERT
Automated Vulnerability Ecosystem Resistance Tactics
SecOps

sonatype

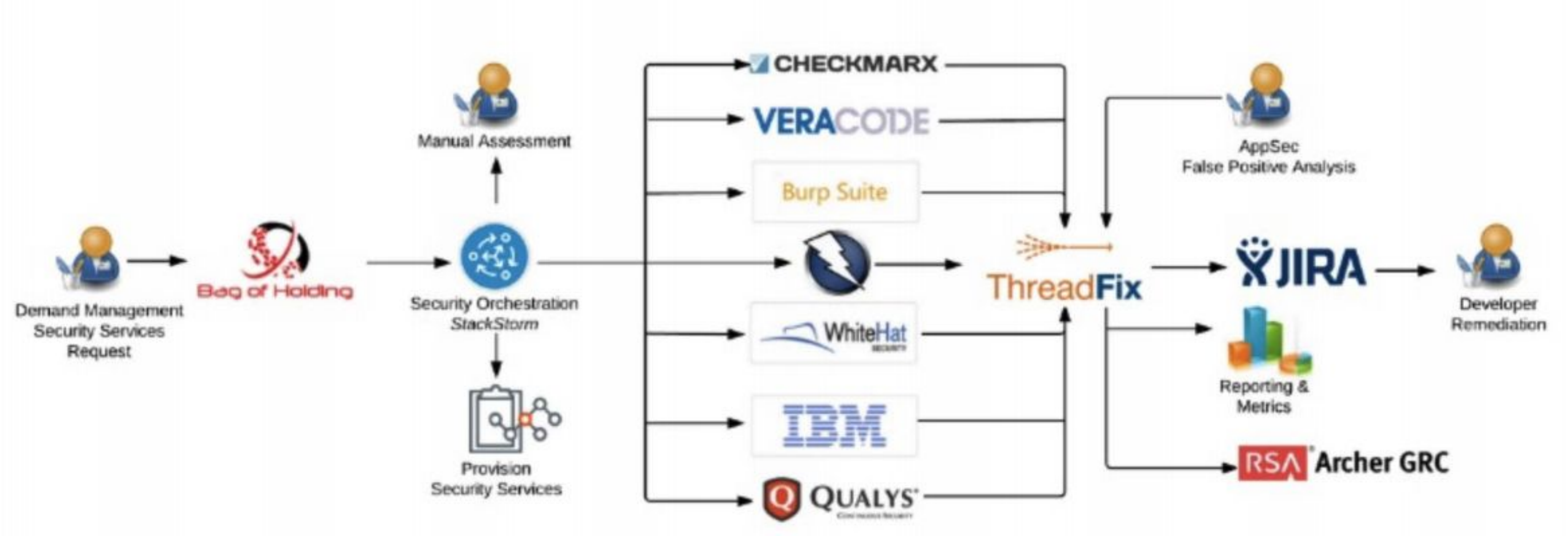
Source: Atul Jadhav, Aricent – ”Security Software”

DevSecOps according to Steve Springett and ServiceNow



Source: Steve Springett, GitHub – "Dependency-Track"

DevSecOps according to Mohammed Imran and TeachEra



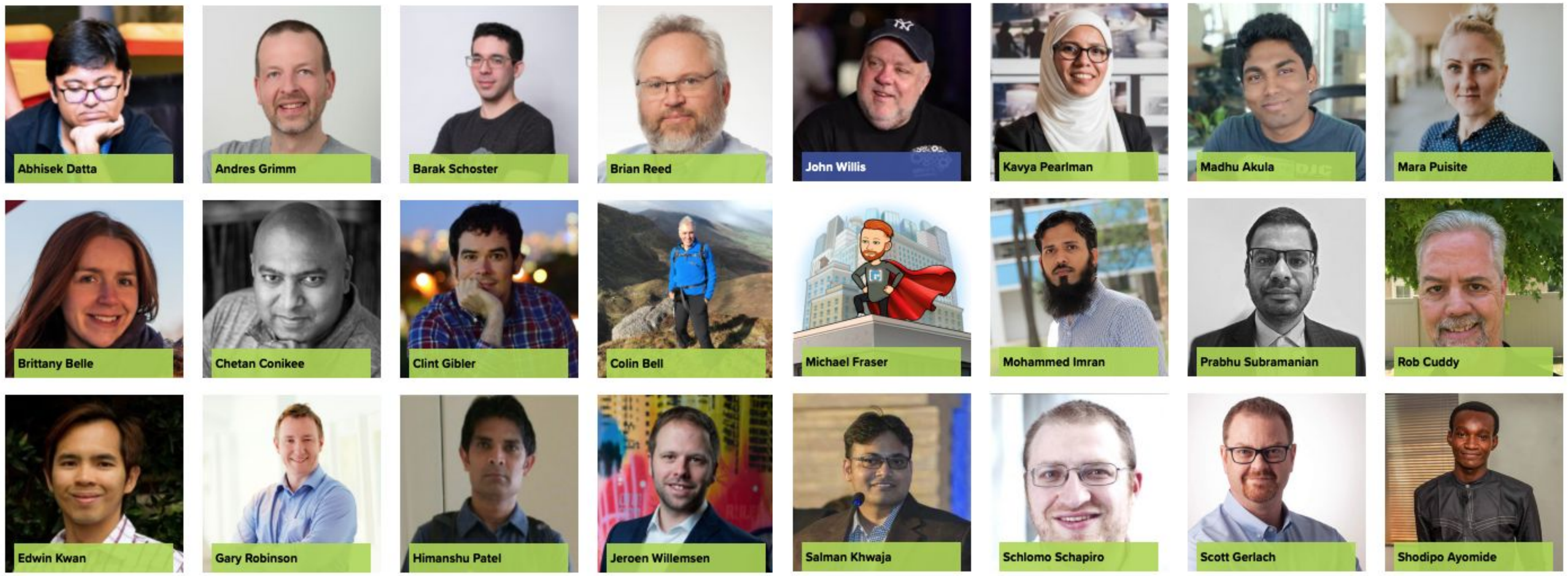
sonatype

Source: Mohammed Imran, LinkedIn – "Practical DevSecOps Course – Part 1"

Learn More About DevSecOps:

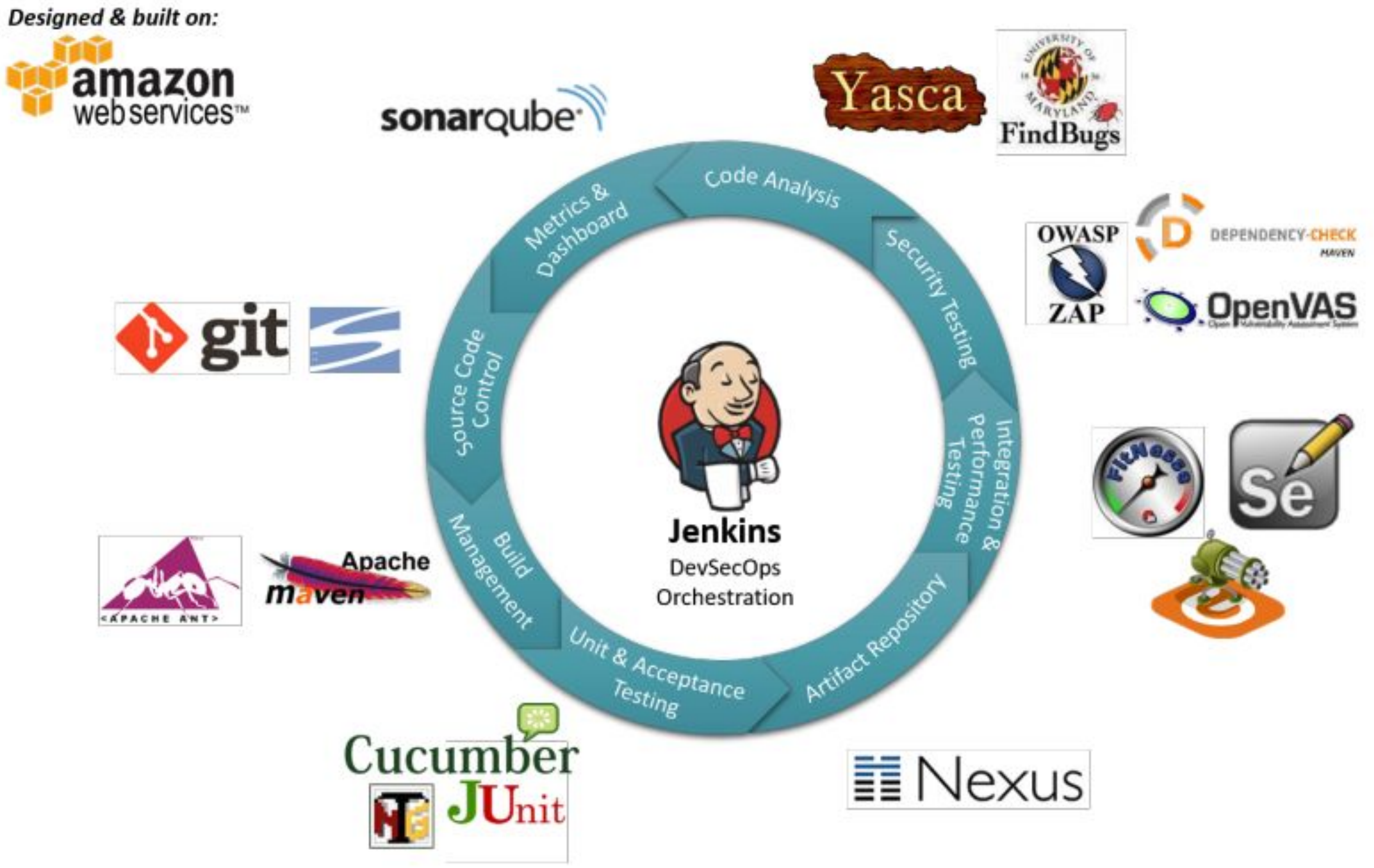
12 Nov 2020 All Day DevOps

24 DevSecOps practitioners from leading enterprises shared their experiences and best practices. Those recordings are all available for **free** at www.alldaydevops.com.



sonatype

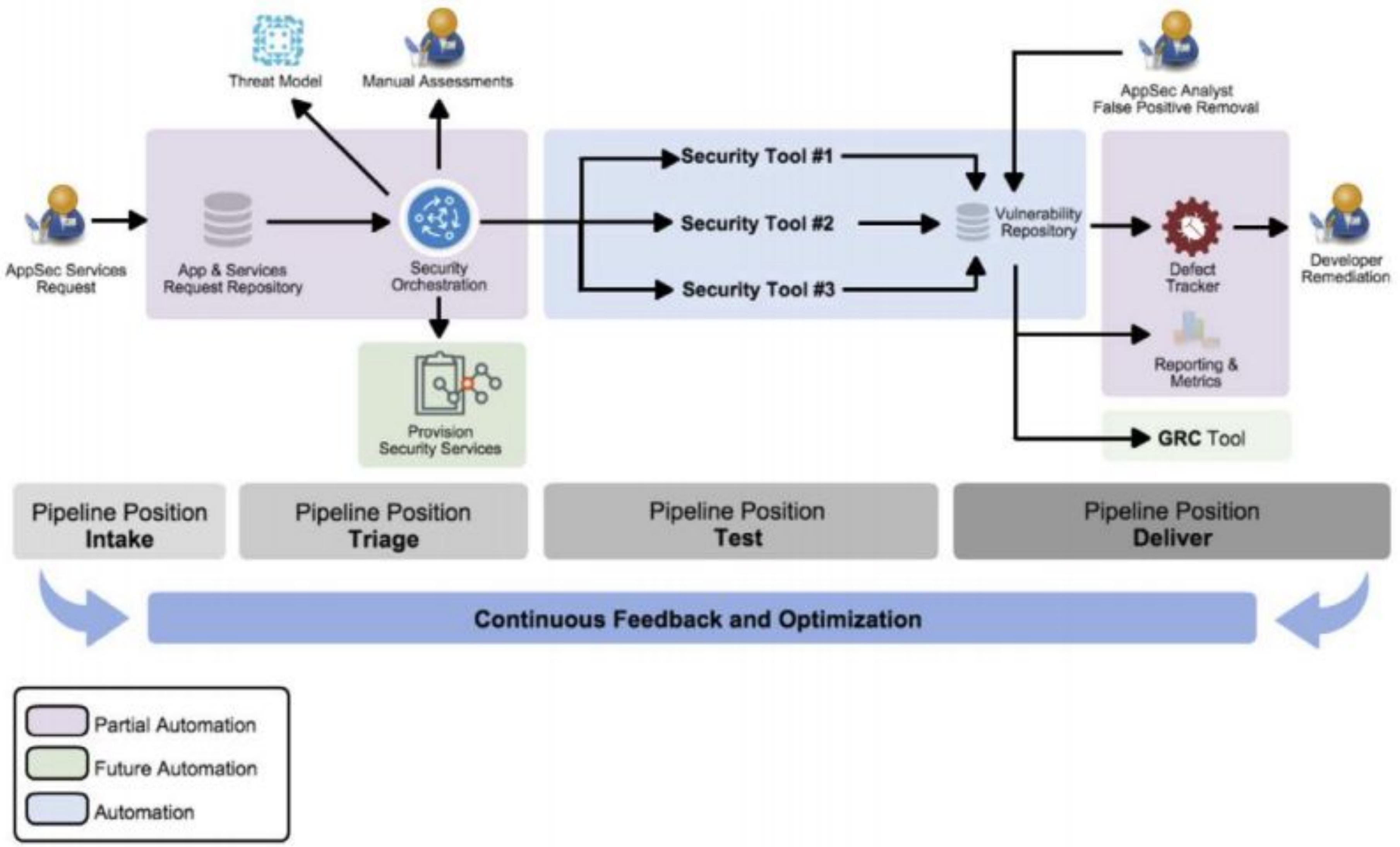
DevSecOps according to Alan Crouch and Coveros



Source: Alan Crouch, Coveros - “Implementing the DevSecOps Process”

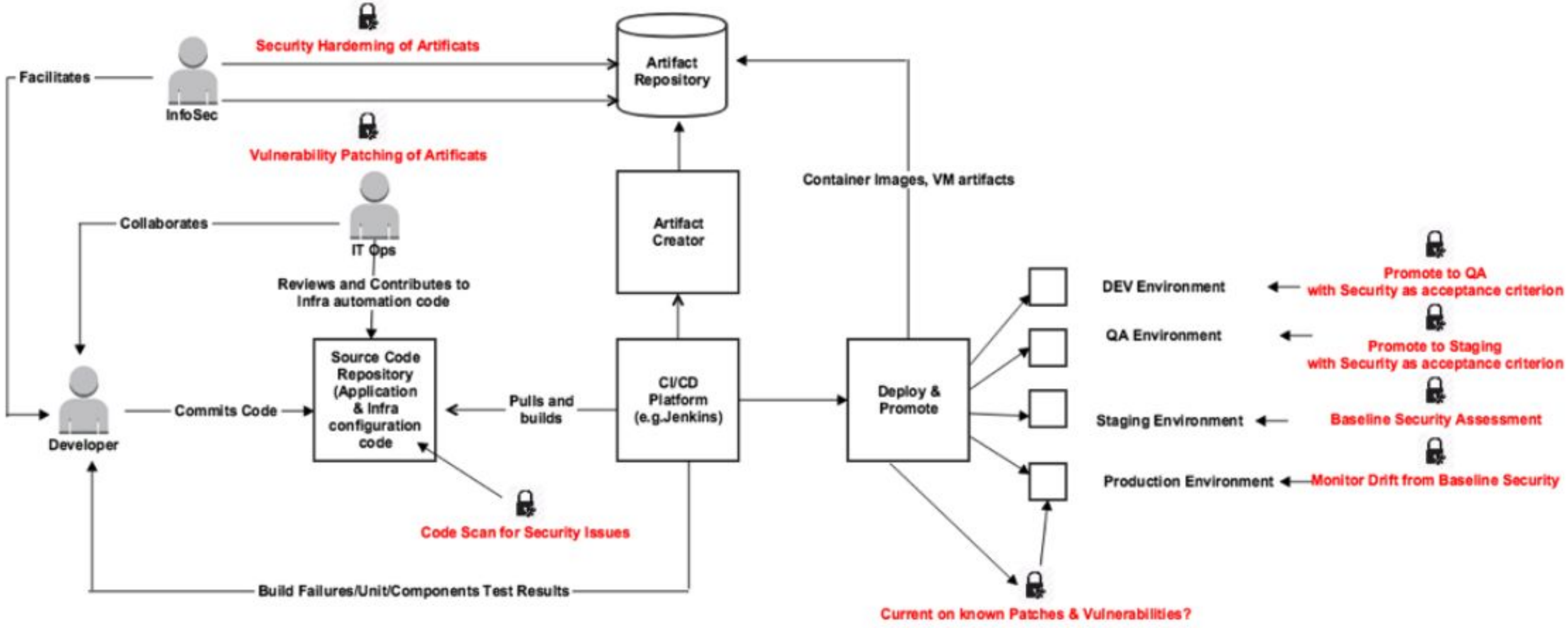
sonatype

DevSecOps according to Aaron Weaver and Protiviti



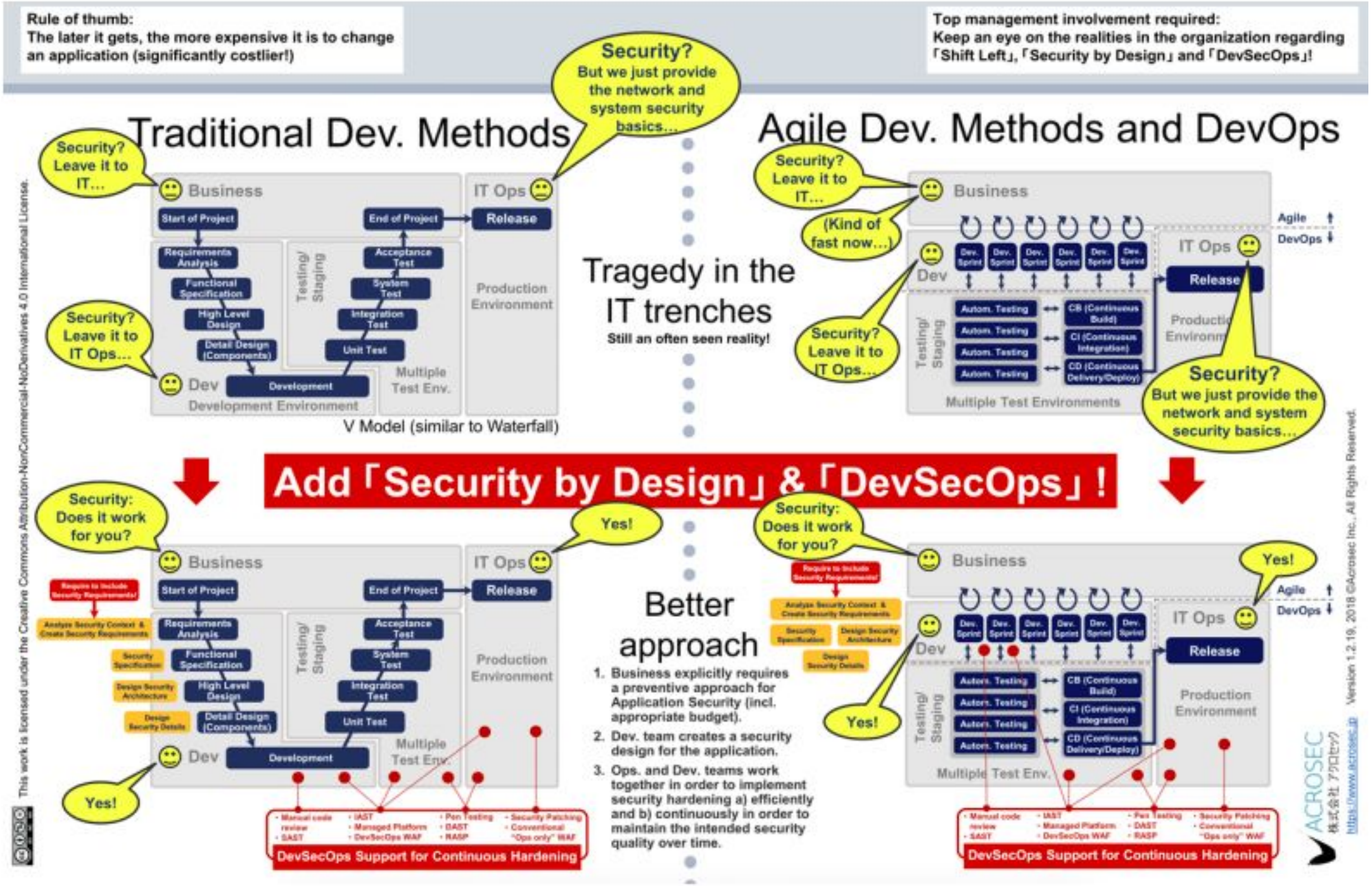
Source: Stefan Streichsbier, LinkedIn – “DevSecOps – The Big Picture”

DevSecOps according to Dr. Ravi Rajamiyer


Security Hardening of Artificats
Facilitates
InfoSec
Artifact Repository
Vulnerability Patching of Artificats
Container Images, VM artifacts
Collaborates
IT Ops
Artifact Creator
Reviews and Contributes to Infra automation code
DEV Environment
Promote to QA with Security as acceptance criterion
QA Environment
Promote to Staging with Security as acceptance criterion
Source Code Repository (Application & Infra configuration code
Pulls and builds
CI/CD Platform (e.g.Jenkins)
Deploy & Promote
Developer
Commits Code
Staging Environment
Baseline Security Assessment
Production Environment
Monitor Drift from Baseline Security
Code Scan for Security Issues
Build Failures/Unit/Components Test Results
Current on known Patches & Vulnerabilities?

Source: Dr. Ravi Rajamiyer, Medium blog– “When ‘IoC’ Meets ‘SoC’”

DevSecOps according to ACROSEC



Rule of thumb:
The later it gets, the more expensive it is to change an application (significantly costlier!)

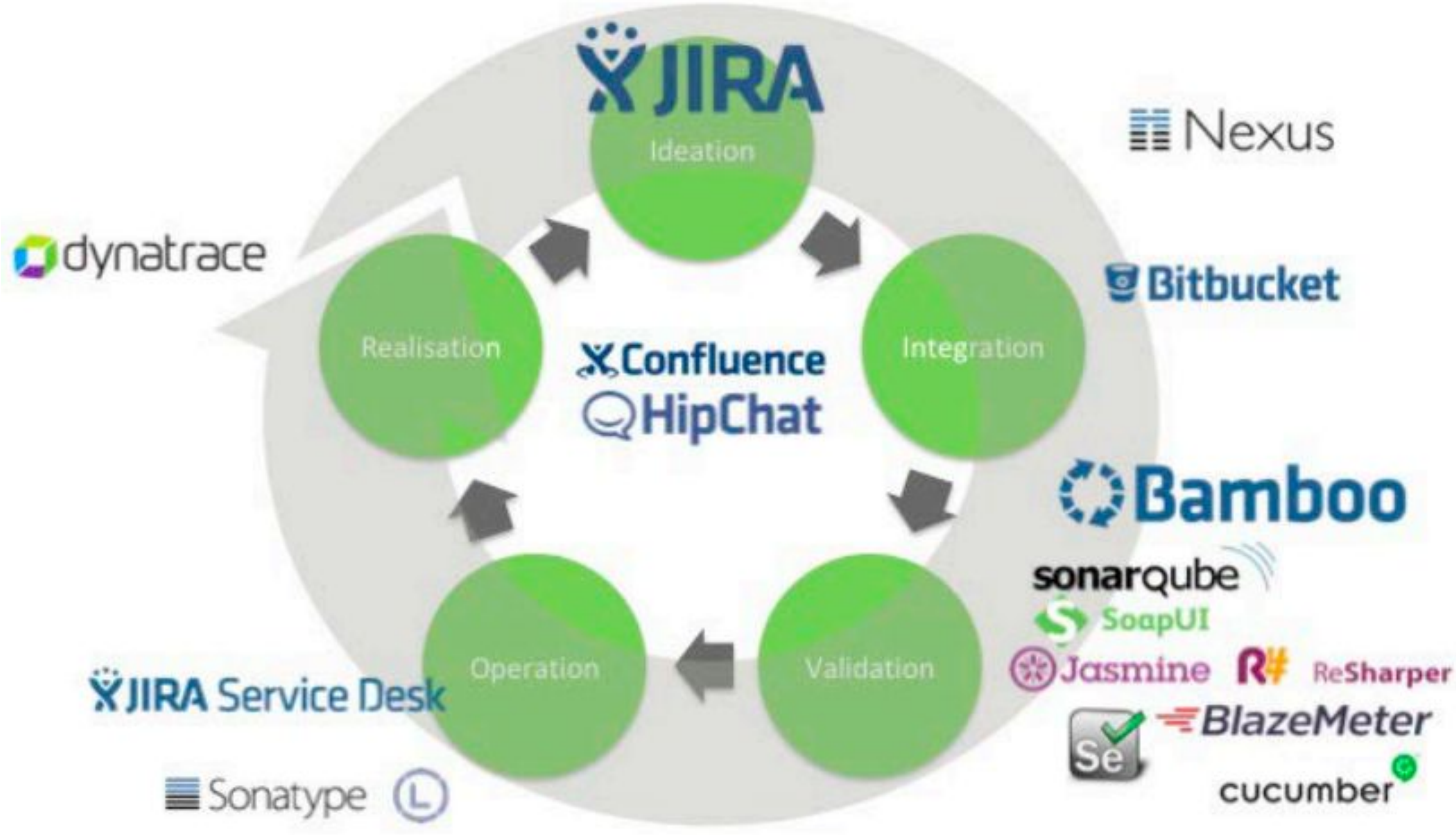
Top management involvement required:
Keep an eye on the realities in the organization regarding「Shift Left」,「Security by Design」and「DevSecOps」!

Traditional Dev. Methods
Agile Dev. Methods and DevOps
Security? But we just provide the network and system security basics...
Security? Leave it to IT...
Security? Leave it to IT Ops...
(Kind of fast now...)
Business
IT Ops
Dev
Start of Project
End of Project
Release
Requirements Analysis
Functional Specification
High Level Design
Detail Design (Components)
Development
Unit Test
Integration Test
System Test
Acceptance Test
Testing/ Staging
Production Environment
Development Environment
Multiple Test Env.
V Model (similar to Waterfall)
Tragedy in the IT trenches
Still an often seen reality!
Dev. Sprint
Autom. Testing
CB (Continuous Build)
CI (Continuous Integration)
CD (Continuous Delivery/Deploy)
Multiple Test Environments
Agile
DevOps
Add「Security by Design」&「DevSecOps」!
Security: Does it work for you?
Yes!
Require to include Security Requirements!
Analyze Security Context & Create Security Requirements
Security Specification
Design Security Architecture
Design Security Details
Better approach
• Manual code review
• SAST
• IAST
• Managed Platform
• DevSecOps WAF
• Pen Testing
• DAST
• RASP
• Security Patching
• Conventional "Ops only" WAF
DevSecOps Support for Continuous Hardening
ACROSEC
株式会社 アクロセック
https://www.acrosec.jp
Version 1.2.19, 2018 ©Acrosec Inc., All Rights Reserved.

1. Business explicitly requires a preventive approach for Application Security (incl. appropriate budget).
2. Dev. team creates a security design for the application.
3. Ops. and Dev. teams work together in order to implement security hardening a) efficiently and b) continuously in order to maintain the intended security quality over time.

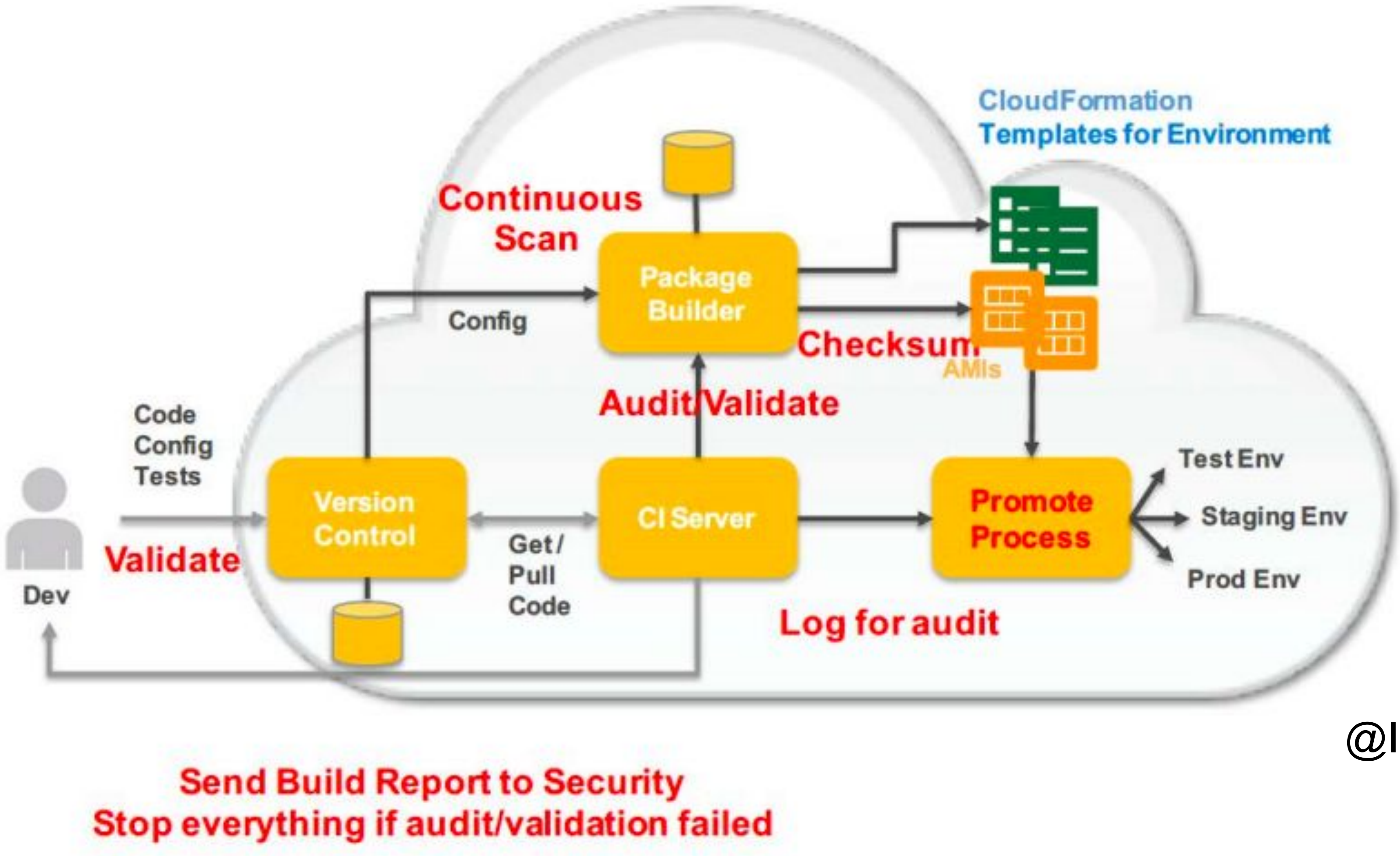
Source: Derek Weeks, ACROSEC – "3 Important Elements of Application Security: 'Shift Left,' 'Security by Design,' and 'DevSecOps'"

DevSecOps according to Helen Beal and Ranger4



Source: Helen Beal, LinkedIn – “DevSecOps: Is It a Good Thing?”

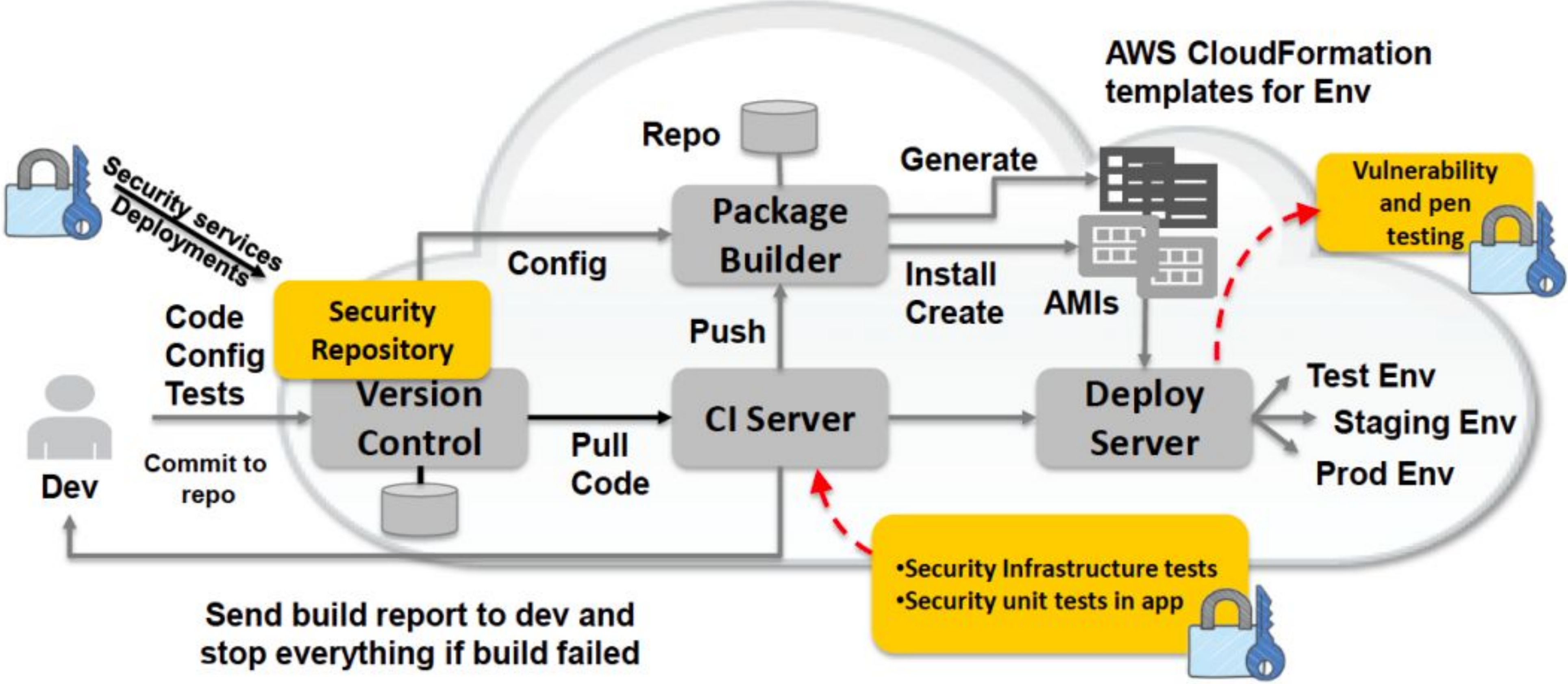
DevSecOps according to Ian Massingham and AWS



@IanMmmm

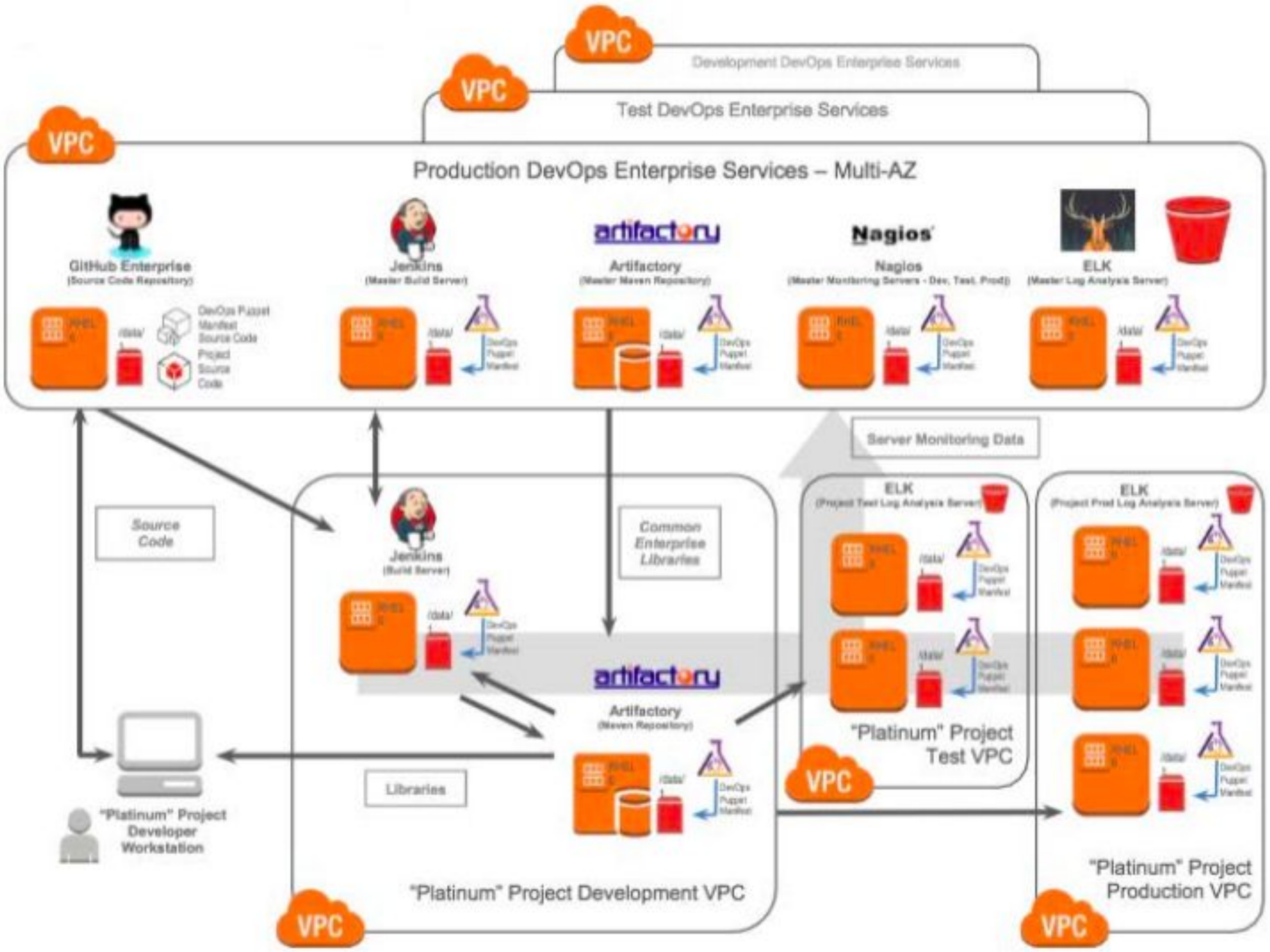
Source: Ian Massingham (@IanMmmm), LinkedIn– “Securing Systems at Cloud Scale with DevSecOps”

DevSecOps according to Priyanka Aash and AWS



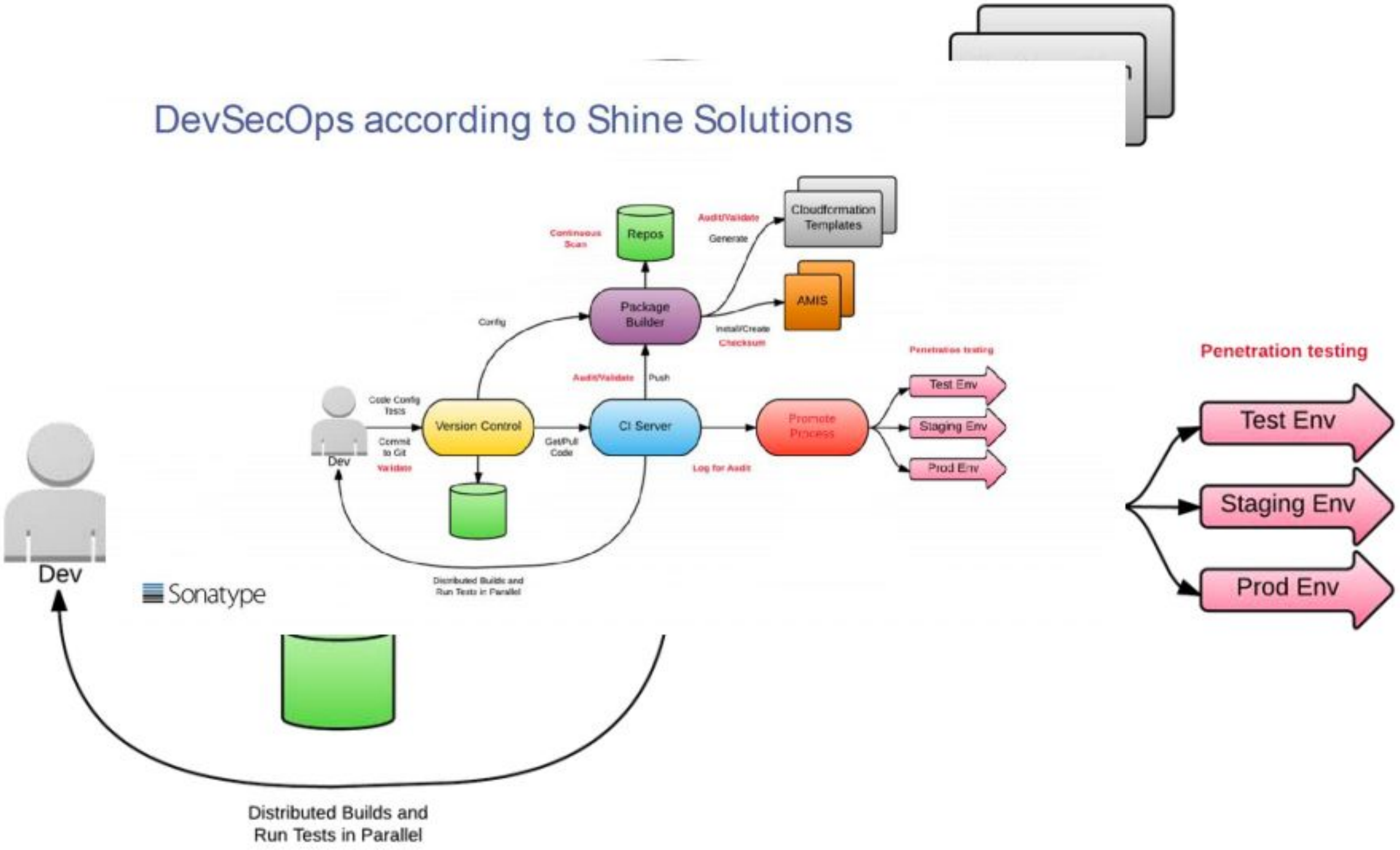
Source: Priyanka Aash, LinkedIn – “DevSecOps in Baby Steps”

DevSecOps according to Dominic Delmolino and Accenture


VPC
Development DevOps Enterprise Services
VPC
Test DevOps Enterprise Services
VPC
Production DevOps Enterprise Services – Multi-AZ
GitHub Enterprise
(Source Code Repository)
RHEL 6
/data/
DevOps Puppet Manifest Source Code
Project Source Code
Jenkins
(Master Build Server)
DevOps Puppet Manifest
artifactory
Artifactory
(Master Maven Repository)
Nagios
Nagios
(Master Monitoring Servers - Dev, Test, Prod)
ELK
(Master Log Analysis Server)
Server Monitoring Data
Source Code
Jenkins
(Build Server)
Common Enterprise Libraries
artifactory
Artifactory
(Maven Repository)
Libraries
"Platinum" Project Developer Workstation
"Platinum" Project Development VPC
VPC
ELK
(Project Test Log Analysis Server)
"Platinum" Project Test VPC
VPC
ELK
(Project Prod Log Analysis Server)
"Platinum" Project Production VPC
VPC

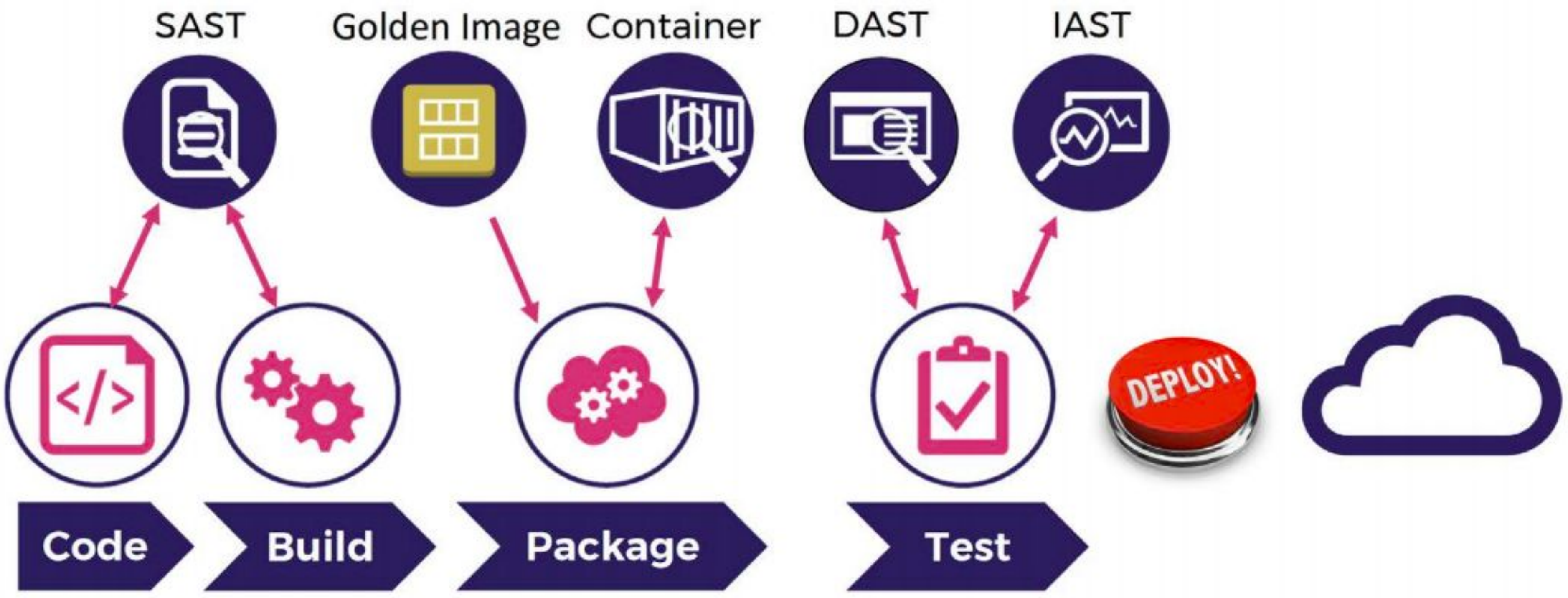
Source: ADDO 2017, YouTube – "DevOps in Secure Environments: Strategies for Success: Dominic Delmolino"

DevSecOps according to Archie Gunasekara and Shine Solutions



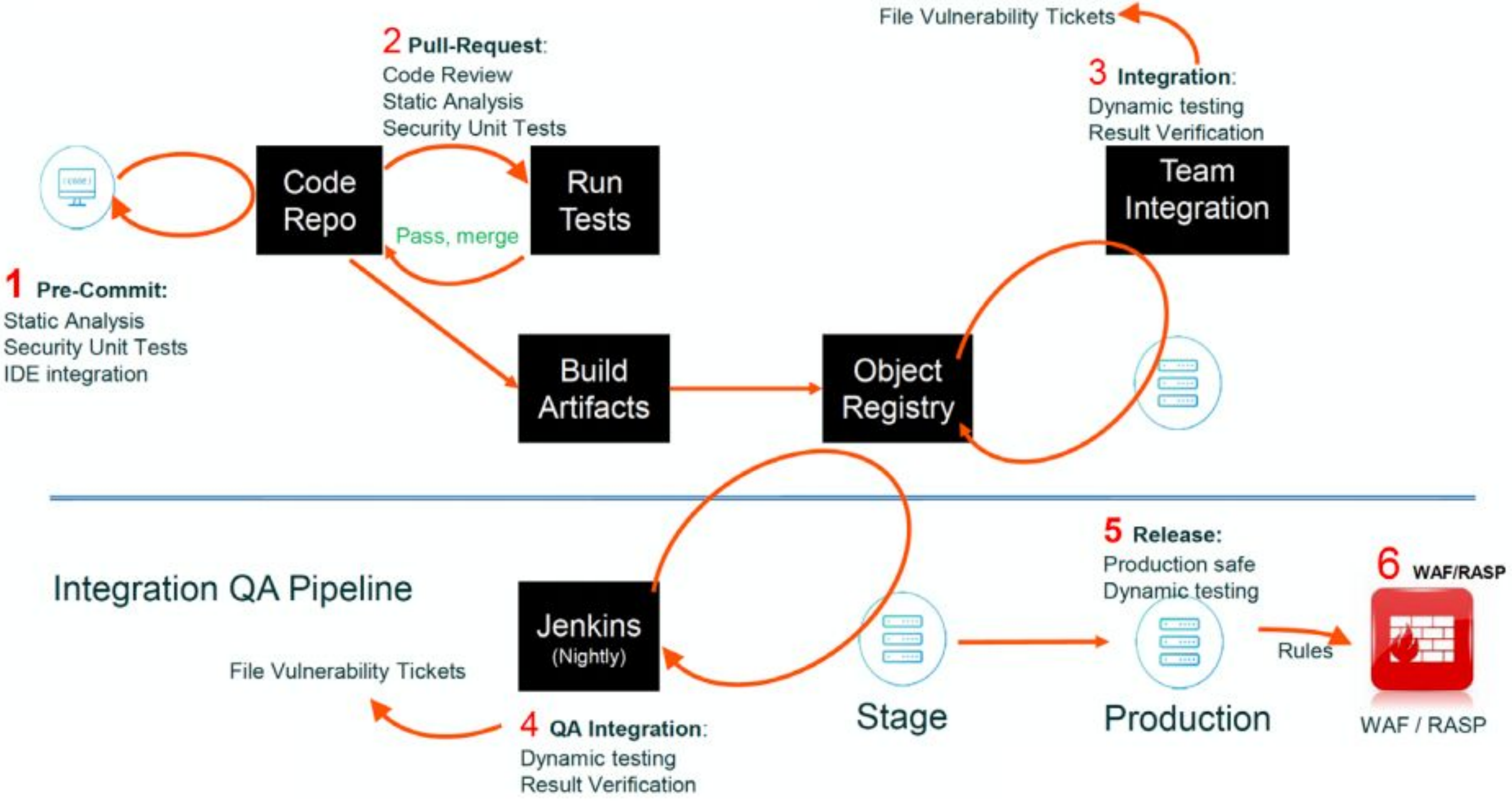
Source: Archie Gunasekara, Shine Solutions – "The Emergence of the 3 Towers: DevSecOps"

DevSecOps according to Mohammed Imran and Ellucian



Source: Mohammed Imran, LinkedIn – “[Practical DevSecOps Course – Part 1]”

DevSecOps according to Siamak Pazirandeh and WhiteHat Security



Source: WhiteHat Security – "Take Control: Design a Complete DevOps Program"

sonatype

sonatype