Platform Engineering Loves Security: Shift Down to Your Platform, not Left to Your Developers!

Cloud Native Rejekts NA 2024

Maxime Coquerel
Microsoft MVP Azure & Security
Principal Kubernetes Cloud Security at RBC

Mathieu Benoit
Cloud Native Ambassador
& Platform Engineer at Humanitec

Disclaimer

“Any views or opinions expressed in this presentation are those of the presenter and not necessarily represent the view and opinions of the employer, its ownership, management or employees.”

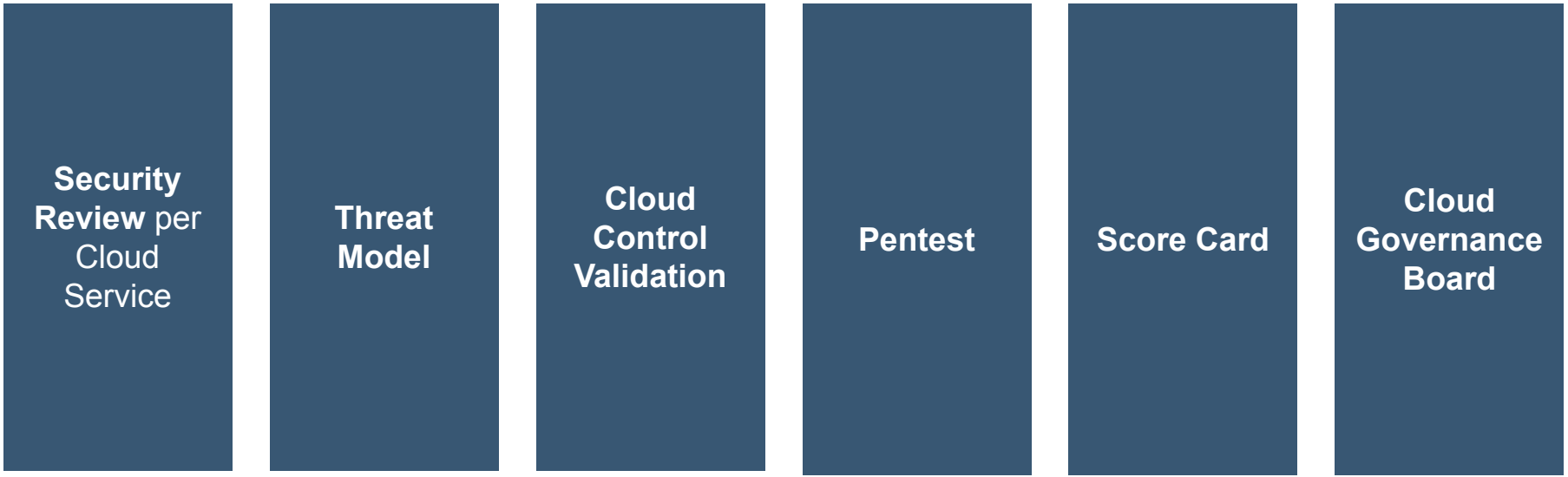
Maxime Coquerel

Agenda

- The real Security complexity in the Cloud Native World
- What’s Platform Engineering?
- Actionable best practices to shift-down, not shift-left
- Resources + Q&A

Security in the Cloud Native World

Security Governance Framework



CNCF Landscape for Security & Compliance

The 4C Security Model

Code

Container

Cluster

Cloud

Reference: https://www.cncf.io/blog/2022/02/14/kubernetes-security-best-practices-definitive-guide/

OWASP: Kubernetes Security Challenge

- KC01: Insecure Workload Configurations
- KC02: Supply Chain Vulnerabilities
- KC03: Overly Permissive RBAC Configurations
- KC04: Lack of Centralized Policy Enforcement
- KC05: Inadequate Logging and Monitoring
- KC06: Broken Authentication Mechanisms
- KC07: Missing Network Segmentation Controls
- KC08: Secrets Management Failures
- KC09: Misconfigured Cluster Components
- KC10: Outdated and Vulnerable Components

Reference: https://owasp.org/www-project-kubernetes-top-ten/

Kubernete

- KC01: Insecure W Mechanisms
- KC02: Supply Ch nentation Controls
- KC03: Overly Pe Failures
- KC04: Lack of Ce Components
- KC05: Inadequat ble Components

Reference: https://owasp.org/www-project-kubernetes-top-ten/

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Impact
Using Cloud credentials	Exec into container	Backdoor container	Privileged container	Clear container logs	List K8S secrets	Access the K8S API server	Access cloud resources	Images from a private registry	Data Destruction
Compromised images in registry	bash/cmd inside container	Writable hostPath mount	Cluster-admin binding	Delete K8S events	Mount service principal	Access Kubelet API	Container service account		Resource Hijacking
Kubeconfig file	New container	Kubernetes CronJob	hostPath mount	Pod / container name similarity	Access container service account	Network mapping	Cluster internal networking		Denial of service
Application vulnerability	Application exploit (RCE)	Malicious admission controller	Access cloud resources	Connect from Proxy server	Applications credentials in configuration files	Access Kubernetes dashboard	Applications credentials in configuration files		
Exposed Dashboard	SSH server running inside container				Access managed identity credential	Instance Metadata API	Writable volume mounts on the host		
Exposed sensitive interfaces	Sidecar injection				Malicious admission controller		Access Kubernetes dashboard		
							Access tiller endpoint		
							CoreDNS poisoning		
							ARP poisoning and IP spoofing		

= **New technique**

= **Deprecated technique**

What’s Platform Engineering?

The new Paradigm that defines Cloud Native Development and Cloud Operations to **“Shift down” to the Platform**

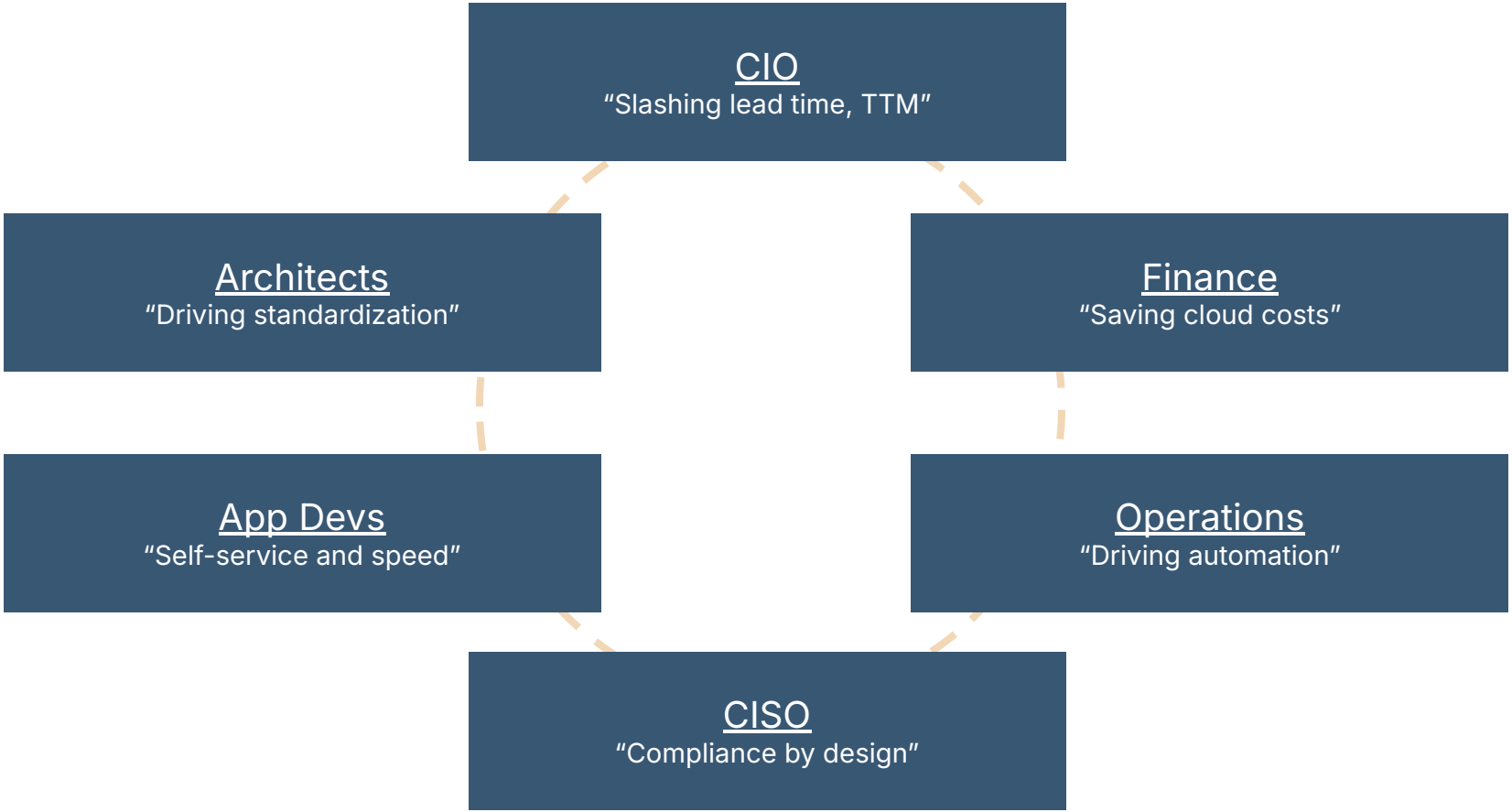
Sys admin
Standardization by separation

→

DevOps
You build it, you run it

→

Platform engineering
DevX, IDP


CIO
“Slashing lead time, TTM”
Architects
“Driving standardization”
Finance
“Saving cloud costs”
App Devs
“Self-service and speed”
Operations
“Driving automation”
CISO
“Compliance by design”

I want to abstract a consistent and secure platform to make Developers more autonomous

Platform Engineer

SRE

Security Engineer

Cloud Engineer

I want to deploy my apps without dealing with Infrastructure and Kubernetes

Developer

Your IDP : standardization and abstraction to improve collaboration and boost productivity

Developer

Platform Engineer

SRE

Security Engineer

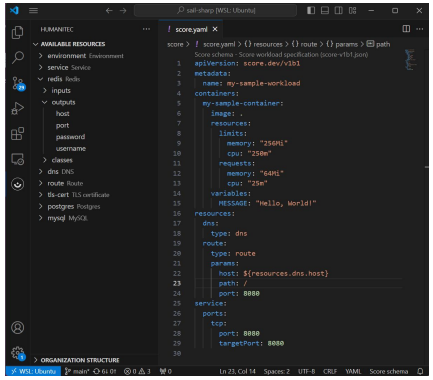
Cloud Engineer

Code

CI/CD Pipelines

Orchestrator
(Abstraction layer)

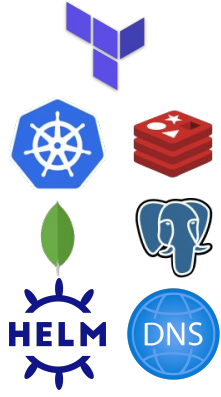
Cloud Infrastructure



dev → pr-*

accp

prod



6 actionable best practices to shift-down, and not shift-left

Cole Kennedy

I Help Organizations Shift Compliance Left | Veteran | Co-f...

Compliance is killing innovation! I just opened this CloudBees survey. The scope of the problem surprised even me. Compliance and Developers need to to get on the same page; this problem is not getting any better; DevSecOps has not shifted compliance left.

Developer

Product and application teams

Platform Interfaces

- Documentation
- Graphical User Interfaces (Web Portals)
- Project templates
- APIs and CLIs

Platform Capabilities

Provide environments and resources

- **infrastructure:** compute, network, storage
- **data:** database, cache, bucket
- **messaging:** queue, broker

- **identify and authorize** users and services
- **scan artifacts & enforce policies**
- **automate build, test & delivery**
- **bind services to workloads** via secrets
- **store artifacts** in registries and repos
- **observe workloads**

Infrastructure providers
Platform capability providers

Security Engineer

Platform Engineer

Security Engineer

Standardizing Configurations

Scaling Security Best Practices

Reducing Attack Surface

Versability

Preventing Privilege Creep

Infrastructure providers
Platform capability providers

Reference: https://platformengineering.org/blog/top-5-security-benefits-of-platform-engineering

Tip 1: Enforce everywhere, and scan everything

IaC & Source Code Scanning

checkov

KICS by Checkmarx

Binary and Container Images Scan

aqua trivy

Compliance Controls

Open Policy Agent

Kyverno

Runtime Security Monitoring

Falco

Tip 2: Shift down: Preventing Misconfiguration with Security Controls for regulated organization

Real-time early notifications from the dev IDE

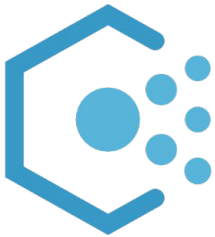
Detective

Preventive

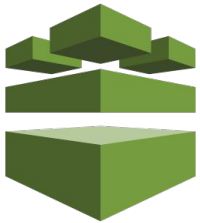
Auto-Remediation

Maturity Level

Cloud Vendor Security Controls



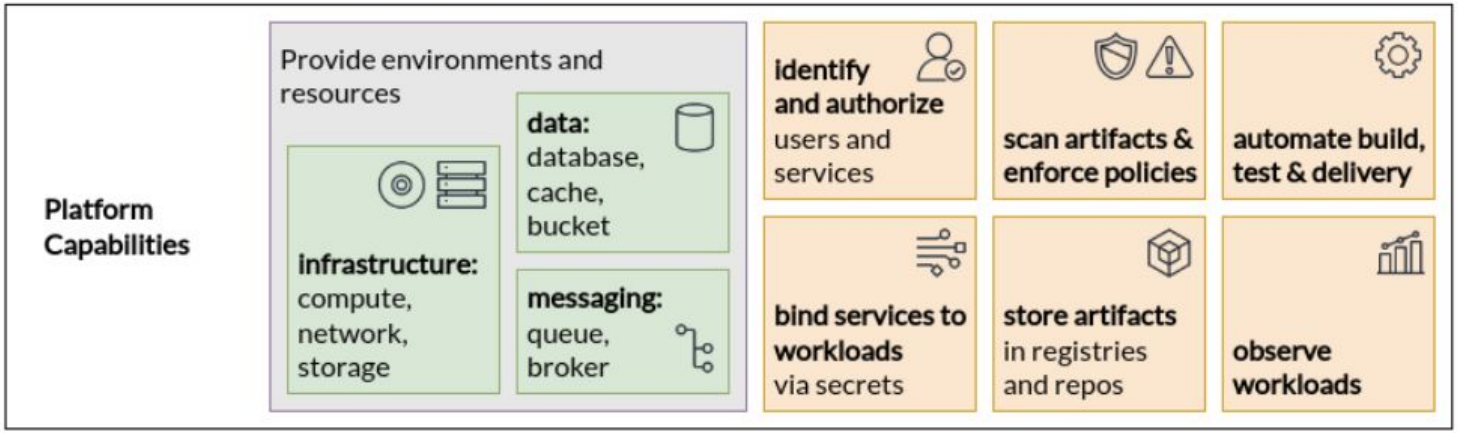
Azure Policy



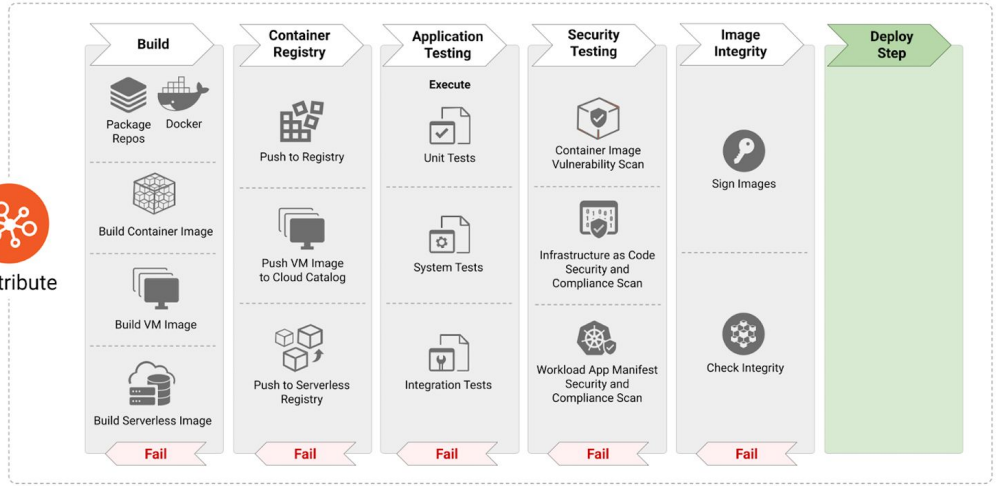
AWS Config

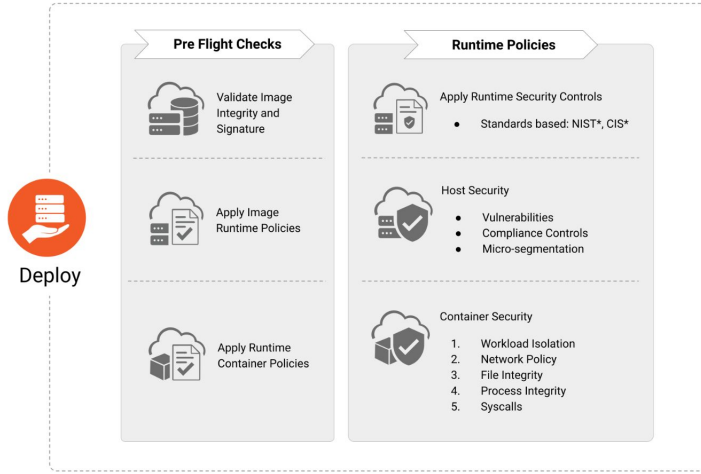
Kubernetes Security Controls

Platform Engineer



Tip 3: Standardize the CI/CD pipelines of your Devs





Buildpacks.io

Chainguard

distroless + unprivileged

Docker Compose

kind

fail fast

trivy

Kvverno

scan CVEs + enforce policies

sign provenance

argo

flux

GitOps

guarantee provenance

Kvverno

enforce policies (PSS restricted)

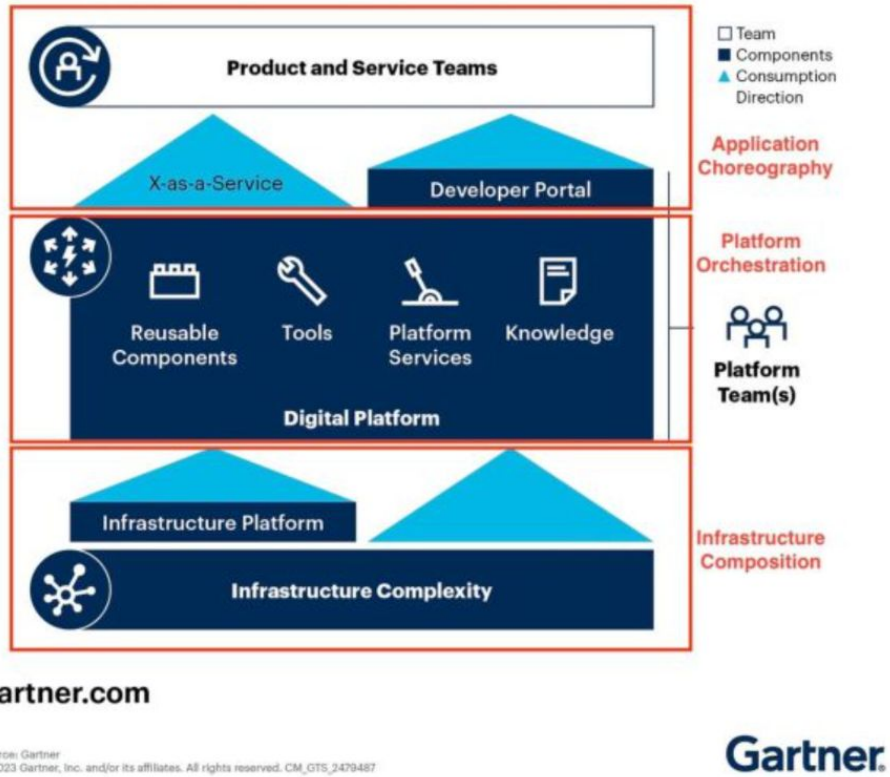
Falco

scan runtime

NetPol

mTLS + AuthZ

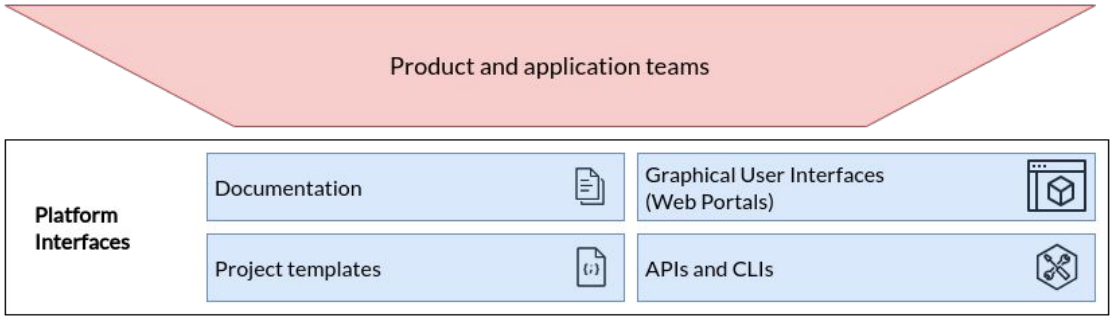
Tip 4: Use an Orchestrator, Kubernetes is not your IDP

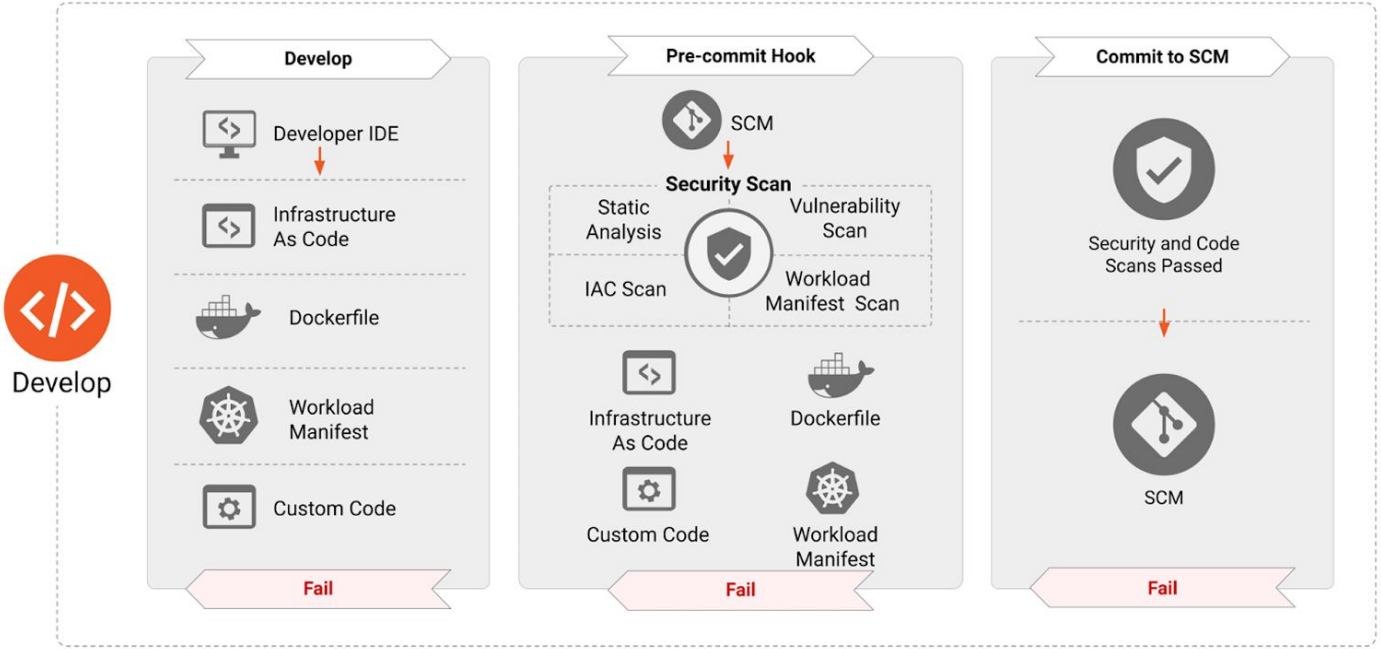


- Crossplane (Composition + Providers)
- Kratix (Promises + Workflows)
- Radius (Recipes)
- Humanitec (Resource Definitions + Drivers)

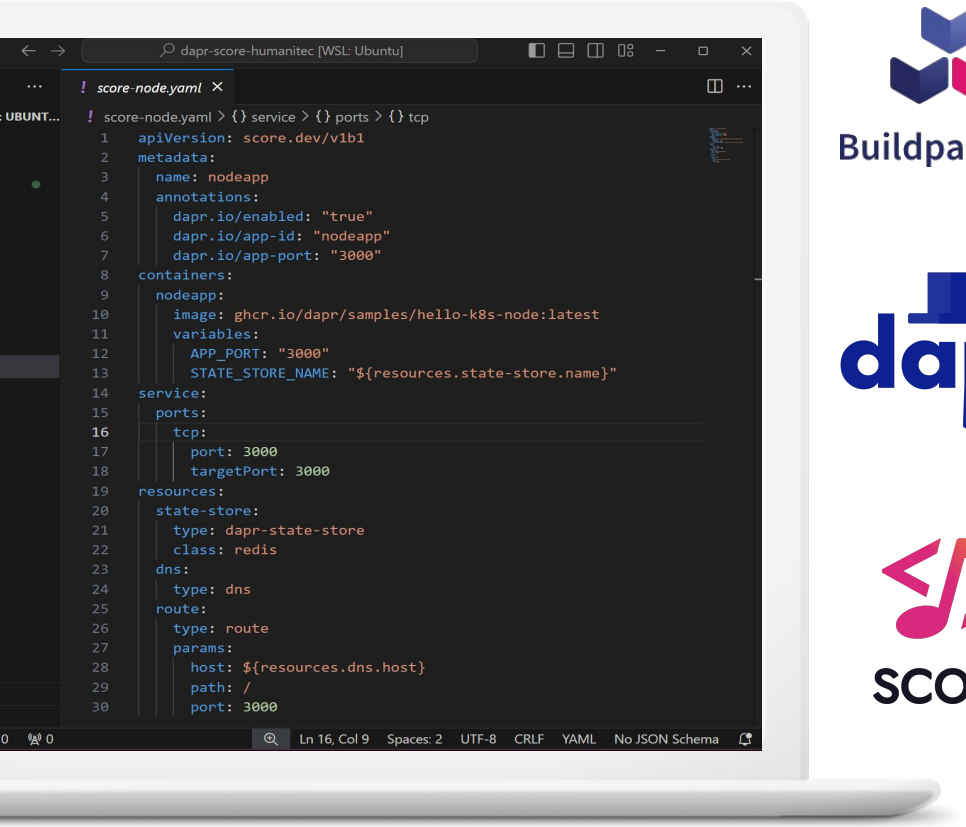
[Platform Orchestrators: The missing middle of IDPs? - Daniel Bryant]

Developer


Product and application teams
Platform Interfaces
Documentation
Graphical User Interfaces (Web Portals)
Project templates
APIs and CLIs


Develop
Develop
Developer IDE
Infrastructure As Code
Dockerfile
Workload Manifest
Custom Code
Fail
Pre-commit Hook
SCM
Security Scan
Static Analysis
Vulnerability Scan
IAC Scan
Workload Manifest Scan
Infrastructure As Code
Dockerfile
Custom Code
Workload Manifest
Fail
Commit to SCM
Security and Code Scans Passed
SCM
Fail

Tip 5: Use abstraction, focus on your code, stay in your inner loop



Buildpacks.io

Do not write Dockerfiles

dapr

Do not directly deal with the cloud infra

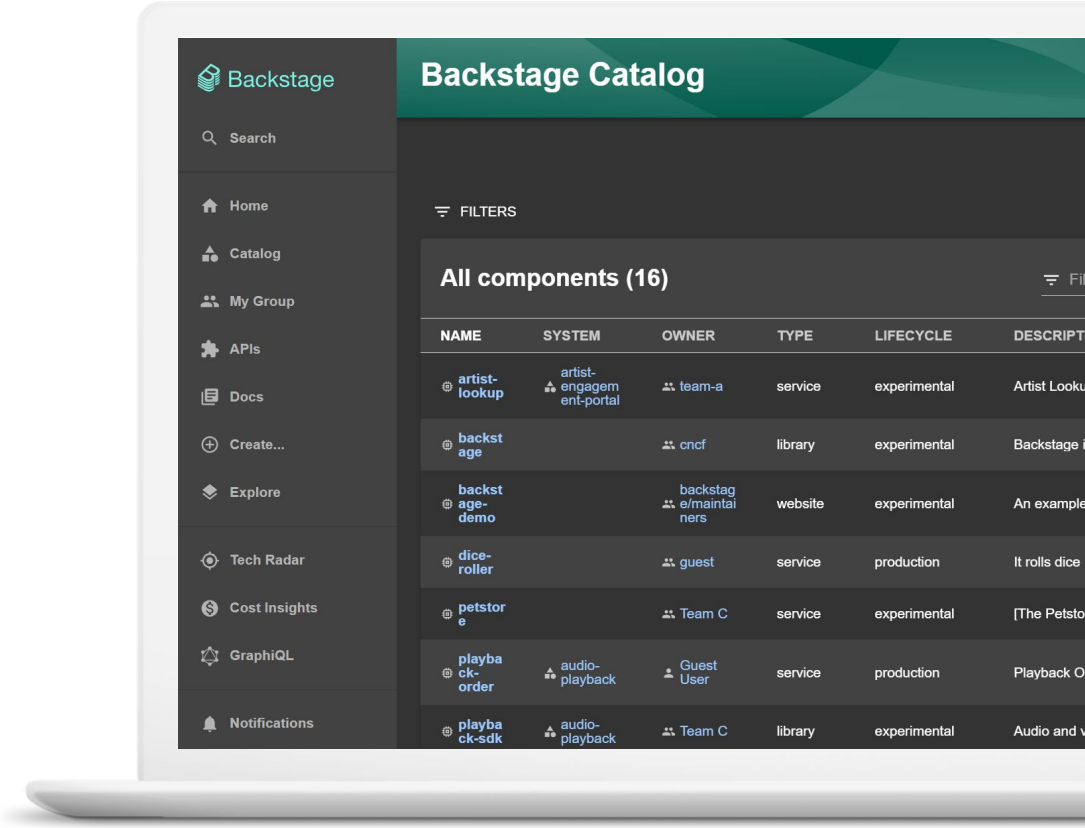
score

Do not write Kubernetes manifests

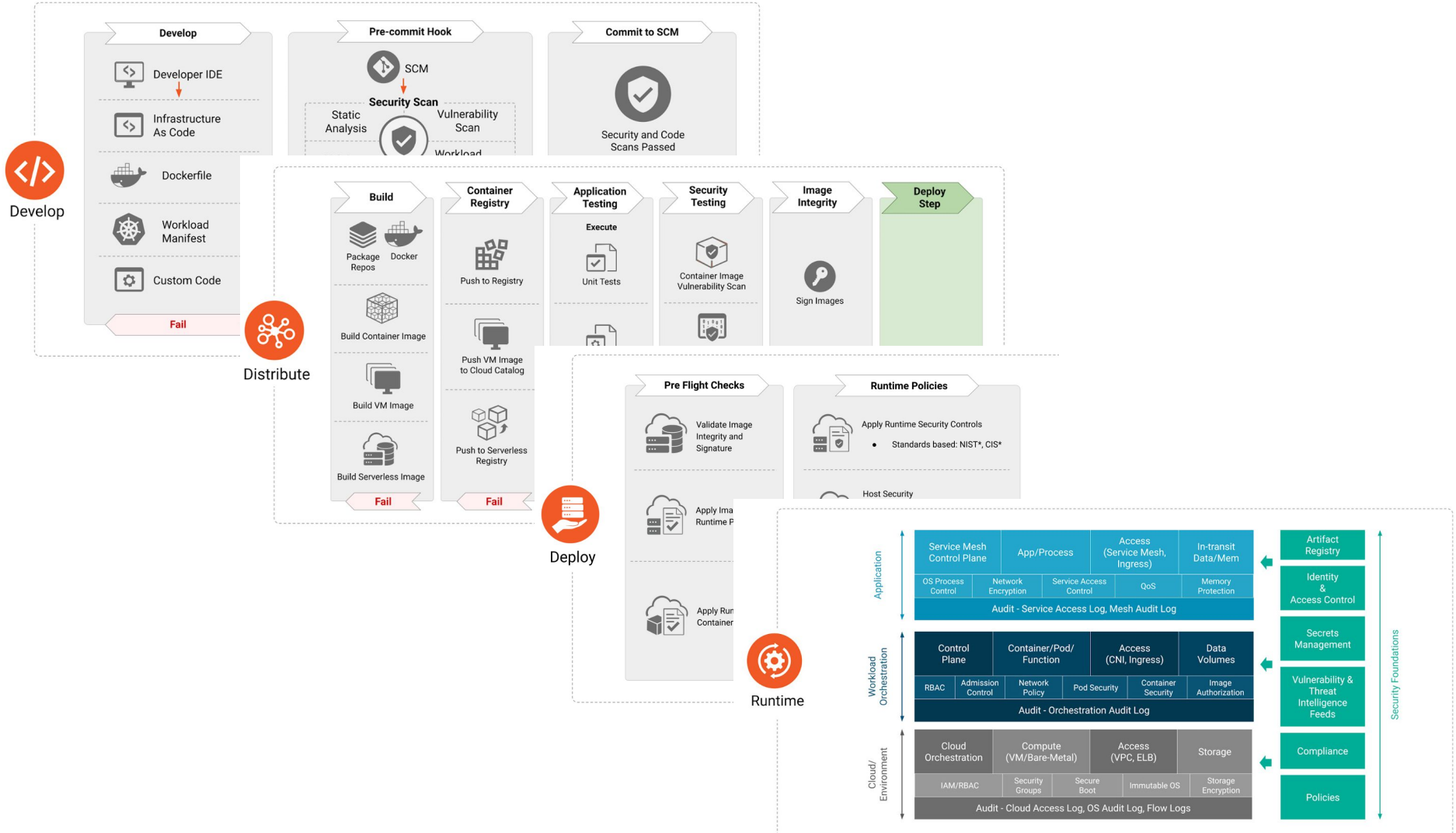
Tip 6: Use a Portal, your single pane of glass, the frontend of your IDP

Backstage

- Accelerate your onboarding with **templates**
- Centralize **tech docs**
- Simplify the **management of your apps** and their dependencies



That’s a wrap!


Develop
Develop
Developer IDE
Infrastructure As Code
Dockerfile
Workload Manifest
Custom Code
Fail
Pre-commit Hook
SCM
Security Scan
Static Analysis
Vulnerability Scan
Workload
Commit to SCM
Security and Code Scans Passed
Distribute
Build
Package Repos
Docker
Build Container Image
Build VM Image
Build Serverless Image
Fail
Container Registry
Push to Registry
Push VM Image to Cloud Catalog
Push to Serverless Registry
Fail
Application Testing
Execute
Unit Tests
Security Testing
Container Image Vulnerability Scan
Image Integrity
Sign Images
Deploy Step
Deploy
Pre Flight Checks
Validate Image Integrity and Signature
Apply Ima
Runtime P
Apply Run
Container
Runtime Policies
Apply Runtime Security Controls
Standards based: NIST*, CIS*
Host Security
Runtime
Application
Service Mesh Control Plane
App/Process
Access (Service Mesh, Ingress)
In-transit Data/Mem
OS Process Control
Network Encryption
Service Access Control
QoS
Memory Protection
Audit - Service Access Log, Mesh Audit Log
Workload Orchestration
Control Plane
Container/Pod/ Function
Access (CNI, Ingress)
Data Volumes
RBAC
Admission Control
Network Policy
Pod Security
Container Security
Image Authorization
Audit - Orchestration Audit Log
Cloud/ Environment
Cloud Orchestration
Compute (VM/Bare-Metal)
Access (VPC, ELB)
Storage
IAM/RBAC
Security Groups
Secure Boot
Immutable OS
Storage Encryption
Audit - Cloud Access Log, OS Audit Log, Flow Logs
Artifact Registry
Identity & Access Control
Secrets Management
Vulnerability & Threat Intelligence Feeds
Compliance
Policies
Security Foundations

What: **Velocity**

Decrease the cognitive load and increase productivity

Why: **Stability**

Standardize and abstract your tools

How: **Product Mindset**

Serve your internal customers with supported golden paths

Devs are your customers, they are users of your IDP

Hierarchy of AppDeveloper Needs

AppDeveloperCon NORTH AMERICA

App Developers, want to be here!

AppDeveloperCon NORTH AMERICA

CNCF Platforms Maturity Model

Aspect		Provisional	Operational	Scalable	Optimizing
Investment	*How are staff and funds allocated to platform capabilities?*	Voluntary or temporary	Dedicated team	As product	Enabled ecosystem
Adoption	*Why and how do users discover and use internal platforms and platform capabilities?*	Erratic	Extrinsic push	Intrinsic pull	Participatory
Interfaces	*How do users interact with and consume platform capabilities?*	Custom processes	Standard tooling	Self-service solutions	Integrated services
Operations	*How are platforms and their capabilities planned, prioritized, developed and maintained?*	By request	Centrally tracked	Centrally enabled	Managed services
Measurement	*What is the process for gathering and incorporating feedback and learning?*	Ad hoc	Consistent collection	Insights	Quantitative and qualitative

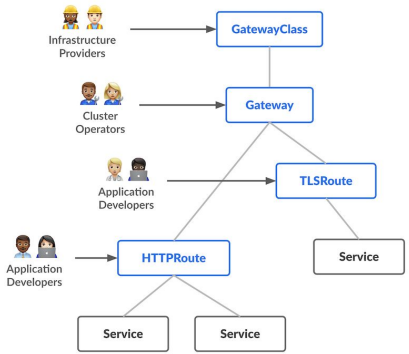
Reference: Announcing the Platform Engineering maturity model | CNCF

Kubernetes is shifting-down to a more role-oriented model

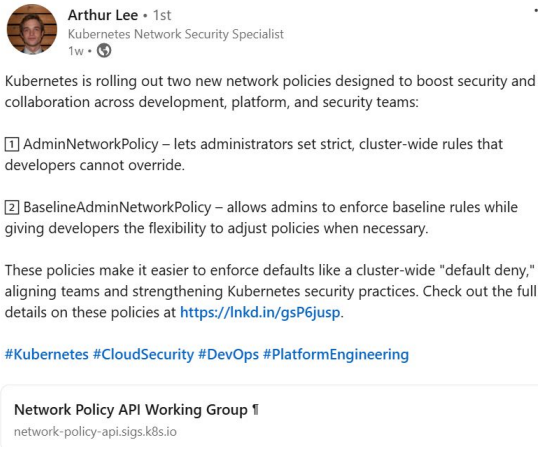
Kubernetes 1.30: Validating Admission Policy Is Generally Available

By **Jiahui Feng (Google)** | Wednesday, April 24, 2024

On behalf of the Kubernetes project, I am excited to announce that ValidatingAdmissionPolicy has reached **general availability** as part of Kubernetes 1.30 release.



gateway api



Arthur Lee • 1st
Kubernetes Network Security Specialist
1w •

Kubernetes is rolling out two new network policies designed to boost security and collaboration across development, platform, and security teams:

1 AdminNetworkPolicy – lets administrators set strict, cluster-wide rules that developers cannot override.

2 BaselineAdminNetworkPolicy – allows admins to enforce baseline rules while giving developers the flexibility to adjust policies when necessary.

These policies make it easier to enforce defaults like a cluster-wide "default deny," aligning teams and strengthening Kubernetes security practices. Check out the full details on these policies at https://lnkd.in/gsP6jusp.

#Kubernetes #CloudSecurity #DevOps #PlatformEngineering

Network Policy API Working Group ¶
network-policy-api.sigs.k8s.io

[Getting started with the AdminNetworkPolicy API - Network Policy API]

Resources

- CNCF SIG Security: tag-security.cncf.io
  - Cloud Native Security Whitepaper
- CNCF App Delivery Tag: tag-app-delivery.cncf.io
  - CNCF Platforms Whitepaper
- The global home for Platform Engineers
- Mathieu's Blog: medium.com/@mabenoit
- Maxime's Blog: zigmax.net

Books:

- community.cncf.io/kubernetes-virtual-book-club
- Learn Kubernetes Security, Packt, Kaizhe Huang and Pranjal Jumde
- Hacking Kubernetes, Threat-Driven Analysis and Defense, O'Reilly, Andrew Martin & Michael Hausenblas
- Platform Engineering on Kubernetes
- Platform Engineering: A Guide for Technical, Product, and People Leaders
- Platform Engineering for Architects: Crafting modern platforms as a product

Thanks!