Achieving Secure, Scalable Multi-Tenancy in Kubernetes with Kyverno and Advanced Namespace Management

SRECon 2026
Seattle, WA

Know Your Presenters

Jeff Spahr

Director, Software Engineering

Roshan Subudhi

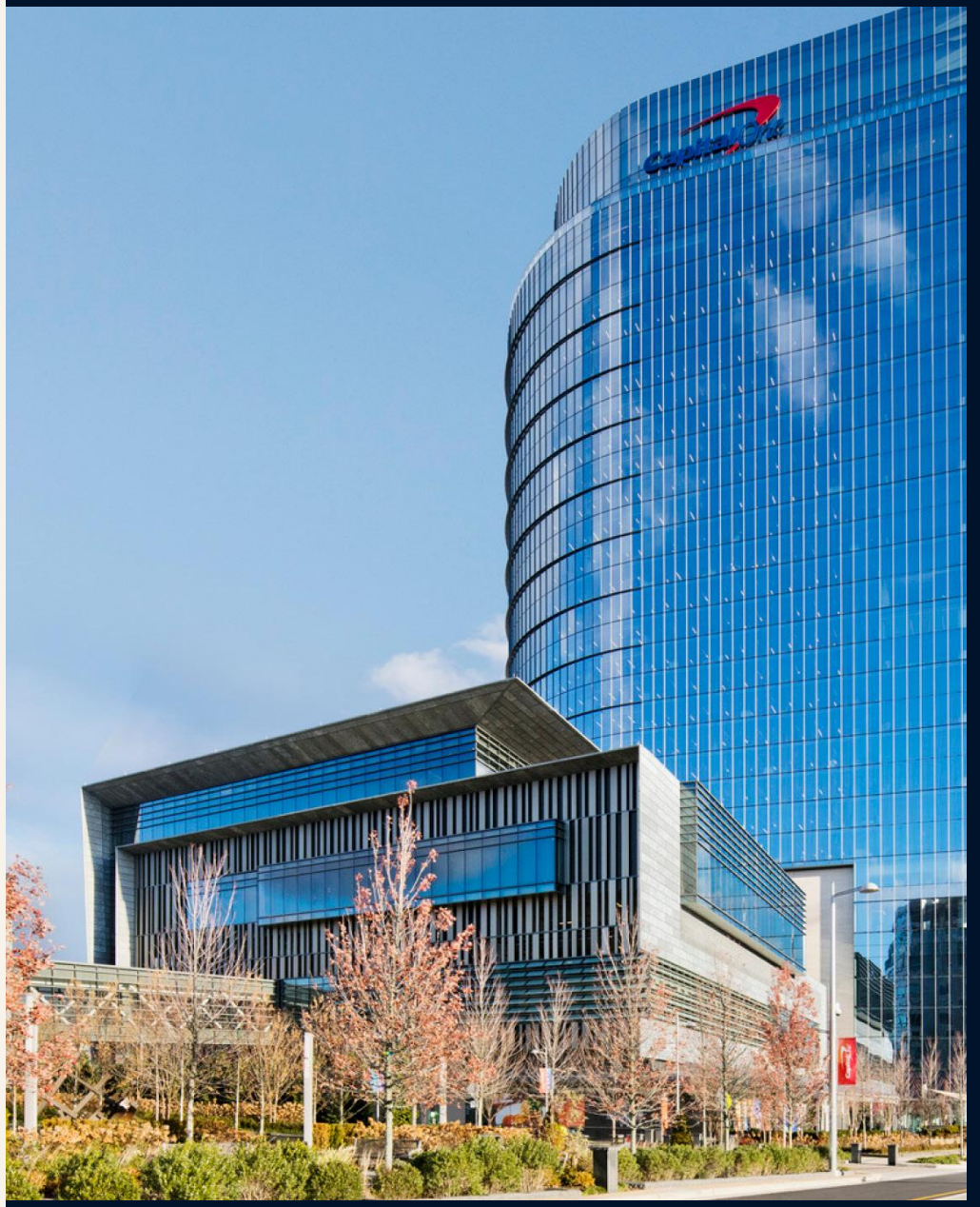
Sr Manager, Software Engineering

Jaison Paul

Sr Lead, Software Engineering

Capital One is a 30-year-old, founder-led, Fortune 100 (NYSE:COF) company

- Nation’s largest direct bank
- Third largest credit card issuer
- Second largest financial institution auto loan originator
- Top-10 bank based on US deposits
- Serves more than 100 million customers and has over 50,000 associates



Agenda

01 **Core Objective**

02 **Challenges**

03 **Solutions**

04 **Results**

05 **Key Takeaways**

Objective

Enterprise-Grade Multi-Tenant Platforms

Capital One

Core Objective

Develop a scalable and compliant multi-tenant platform, prioritizing:

- Robust isolation and predictable performance
- Inherent security and compliance
- Consistent governance throughout the Development to Production lifecycle

Challenges

Challenges of Managing an Enterprise Multi-Tenant Kubernetes Platform

Challenges

Resource Exhaustion

a.k.a “Noisy Neighbor Problem”. A single poorly configured application could consume excessive resources, leading to CPU/Memory starvation and degraded service quality for every other tenant on the shared cluster.

Operational Scalability

The sheer volume of tenants and deployed resources introduces API server strain and complexity in fleet management, making manual security and governance checks unsustainable.

Complex Policy Implementation

Relying on individual teams to correctly implement security best practices like, non-root containers, network isolation etc. results in security drift and potential policy violations across the enterprise fleet and leveraging auditable Role-Based Access Control (RBAC).

Challenge #1 : Resource Exhaustion

Single workloads consuming excessive CPU/Memory

- Cluster-wide instability from poor configurations
- Manual resource enforcement was unsustainable

Challenge #2 : Operational Scalability

Hundreds of Namespaces, thousands of Pods across several Clusters in the Fleet

- API server strain & fleet complexity
- Manual governance checks didn’t scale

Challenge #3 : Complex Policy Implementation

Teams implementing security inconsistently

- Root containers, hostPath volumes, privilege escalation risks
- Enterprise-wide policy drift
- Keeping up with ever-changing enterprise security guidelines

Individual developers (UserIDs) need sandboxes

- Application teams (AD Groups) need persistent environments
- Clear, auditable RBAC boundaries required

Solutions

Layered Governance Framework

Capital One

Solutions

Tiered Namespace Provisioning

Defined distinct Namespace models - Personal Namespaces (UserID-mapped) and Team Namespaces (AD Group-mapped) - that allow for seamless experimentation and development.

Zero-Touch Resource Governance

Automatically create ResourceQuota and LimitRange objects within every Namespace while a new tenant (development team) is being onboarded.

Kyverno Policy-as-Code

Utilize Kyverno's capabilities to shift the burden of security compliance from the developer to the platform by enforcing specific configurations and preventing non-compliant resources from being created or updated

AuthN and AuthZ

Leverage reusable AuthN and AuthZ policies that tie into existing enterprise IDP systems so access is controlled and audited via RBAC policies provisioned based on tenant needs.

Solution #1: Tiered Namespace Provisioning

Boundary management for multi-tenant platforms

- Personal Namespaces → UserID-mapped
  - Used for experimentation and prototyping, short-lived.
- Team Namespaces → AD Group-mapped
  - Used by business teams, Dev/QA/Prod, long-lived
- Clear separation of temporary vs persistent environments

Solution #2: Zero-Touch Resource Governance

Bootstrap Namespaces with sensible defaults at Onboarding

- Automatic ResourceQuota definition
- Default LimitRange enforcement
- Configure default-deny NetworkPolicy
- Option for Customization from defaults

Solution #3: Kyverno Policy-as-Code

Continuous Platform compliance monitoring using Kyverno

- Validation: Prevent non-compliant resources, like *hostPath* or *root*
  - Helps with misconfigurations
- Generation: Inject sidecars required for access to Enterprise systems
  - Helps with Security and Network Isolation
- Mutation: Auto-inject labels like *cost-center* and *owner-contact*
  - Helps with tagging and ownership

Solution #4: AuthN and AuthZ

Out-of-the-Box reusable RBAC policies applied uniformly

- Custom privileges based on Dev/QA/Prod environments
- Permissions scoped to Persona i.e., User vs Developer
- Permissions scoped to Role i.e., ReadOnly vs PowerUser vs Admin
- Option for Customization from defaults

Impact

Results and Impact

Results

- Elimination of ‘Noisy Neighbor’ incidents
- Hundreds of active tenant teams supported
- Compliant Team onboarding reduced by over 80%
- Full compliance with Container and Network policies
- Reduced support requests around compliance

Takeaways

Key Takeaways

Key Takeaways

Multi-tenancy must be policy-driven

- Automate governance at onboarding
- Shift security left to the platform layer
- Clear identity boundaries simplify auditing

Thank you!