Confluent For Kubernetes (CFK) On OpenShift Demo

Creator: Sion Smith (CTO)

Key benefits of CFK on OpenShift

OSO can help you adopt Kafka at scale in a secure, controlled and repeatable manner

01

Operability

Helping you to operate Kafka at scale with minimal resource

02

Reliability

Build confidence by reducing complexity and leveraging reusable components

03

Maintainability

Remove the headache of running Kafka with Confluent operator for Kubernetes

04

Demo

What are we covering today

01
Operability

Helping you to operate Kafka at scale with minimal resource

Official Docker images

> Platform is deployed on Confluent built and supported Docker images

Declarative approach

> Everything is declared in YAML which removes ambiguity around Terraform controlling configuration

Reduced complexity

> Single pane of glass, which one two moving parts and a standardised way across every environment

No VM or AMI builds

> Removed the need for long AMI builds and deployment pipelines

Seamless integration with OpenShift operators

> Adding / Removing Confluent resources is integrated into the lifecycle of OCP and cluster console providers clear status updates on resources

02

Reliability

Build confidence by reducing complexity and leveraging reusable components

Leverage the latest GitOps technology

> Repeatable, automated and scalable GitOps focused deployments

Monitoring out of the box

> Platform health is monitored constantly by OpenShift control plane using health and liveness probes

Deploy Kafka at scale

> Lower the barrier to entry with isolated deployments which teams can utilise with confidence

Remove operational complexity

> Leverage PaaS which removes many moving parts (Terraform / Packer / AMI / AWS / Ansible)

One to one mapping in Git

> Focus on running Kafka with simple repeatable deployments across multiple environments without obfuscation

03
Maintainability

Remove the headache of running Kafka with Confluent operator for Kubernetes

OpenShift approved operators

> Operator pattern is designed to do the heavy lifting. This is not a silver bullet on its own

Automated Kafka operations

> Broker rebalancing, disk attachment and certificates management are built into CFK

FluxCD controls cluster lifecycle

> GitOps with native CRDs has now enabled traceability, auditability and scalability in Kafka

Flexible upgrade roadmap

> Confluent CRDs are outside the lifecycle of the core Confluent components, supported flexible upgrades

Keep the security and operations team happy

> Patching is done at a Docker image level. Monitoring out of the box, prometheus metrics exposed at a component level

04

Demo

What are we covering today

1. OpenShift setup and basic UI walkthrough
2. CFK Configuration and repository setup
3. Multi tenant design with central governance
4. Deploying Kafka per namespace / project
5. Onboarding a new tenant (team)
6. Creating a topic and a connector
7. Performing a platform upgrade

We’re your trusted, strategic technology partner

Let’s talk about what is next

enquiries@oso.sh.