Cloudera Streams Messaging Operator 1.1.0

Overview

Date published: 2024-06-11
Date modified: 2024-09-04

https://docs.cloudera.com/

Legal Notice

Contents

What is CSM Operator?.....4

Deployment architecture.....5

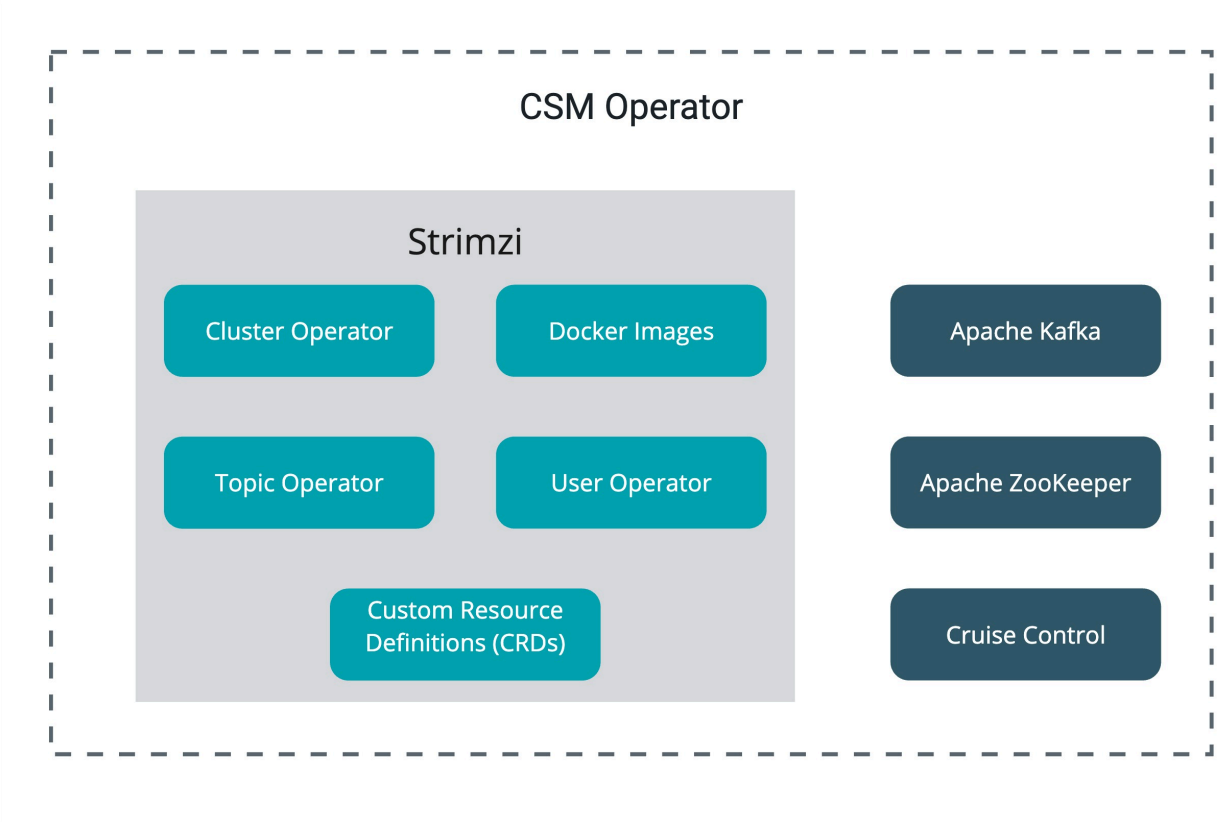
Licensing..... 7

Sizing and performance considerations..... 7

Recommended minimum setup..... 8

What is CSM Operator?

CSM (Cloudera Streams Messaging) Operator allows you to deploy and manage Streams Messaging components as container applications on Kubernetes. CSM Operator simplifies the process of creating, managing, and troubleshooting Kafka deployments in a Kubernetes environment. CSM Operator includes several services and components such as Strimzi, Kafka, Cruise Control, and others.



Strimzi

Strimzi is an open-source project that provides a way to run an Apache Kafka cluster on Kubernetes. Within CSM Operator, Strimzi is the component that makes it possible to deploy and manage Kafka workloads in a Kubernetes environment using Kubernetes-native tooling and processes.

Strimzi itself is made up of multiple components and includes various operator applications, Custom Resource Definitions (CRD) as well as Docker (container) images.

Operator applications are purpose built Kubernetes applications that act as an extension to Kubernetes. These applications provide an easy way for you to deploy, manage and configure Kafka and related components.

The CRDs created by Strimzi define the APIs to interface with Kafka-related custom resources on Kubernetes as, for example, `KafkaCluster`, `KafkaNodePools`, and `KafkaTopic`. The custom resources are created as instances of these APIs by providing an associated set of configurations to be applied to the resource. CRDs and custom resources are defined as YAML files.

Kafka

Apache Kafka is an open-source, high performance, highly available, and redundant streams messaging platform. It supports millions of messages per second with low latency and high throughput, scaling elastically and transparently without downtime. Kafka addresses a wide range

of streaming data initiatives, enabling enterprises to keep up with customer demand, provide better services, and proactively manage risk.

Apache ZooKeeper

Apache ZooKeeper is an open-source centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services. Kafka uses Zookeeper for broker coordination as well as to store broker, topic, and partition metadata.

Cruise Control

Cruise Control acts as a load balancing component in large Kafka installations. It provides automatic data balancing of Kafka partitions across Kafka clusters based on user specified parameters (goals) as well as workload data.

Components shipped in CSM Operator are based on open source projects and might contain additional changes or fixes to guarantee that they work in Cloudera supported environments.

Additionally, not all Kafka and Strimzi features are supported by CSM Operator. See the Release Notes for a comprehensive list of unsupported features and components.

Note: Cloudera will replace ZooKeeper with KRaft in a future release.

Key features and benefits

A Kafka deployment with CSM Operator provides the following key features and benefits.

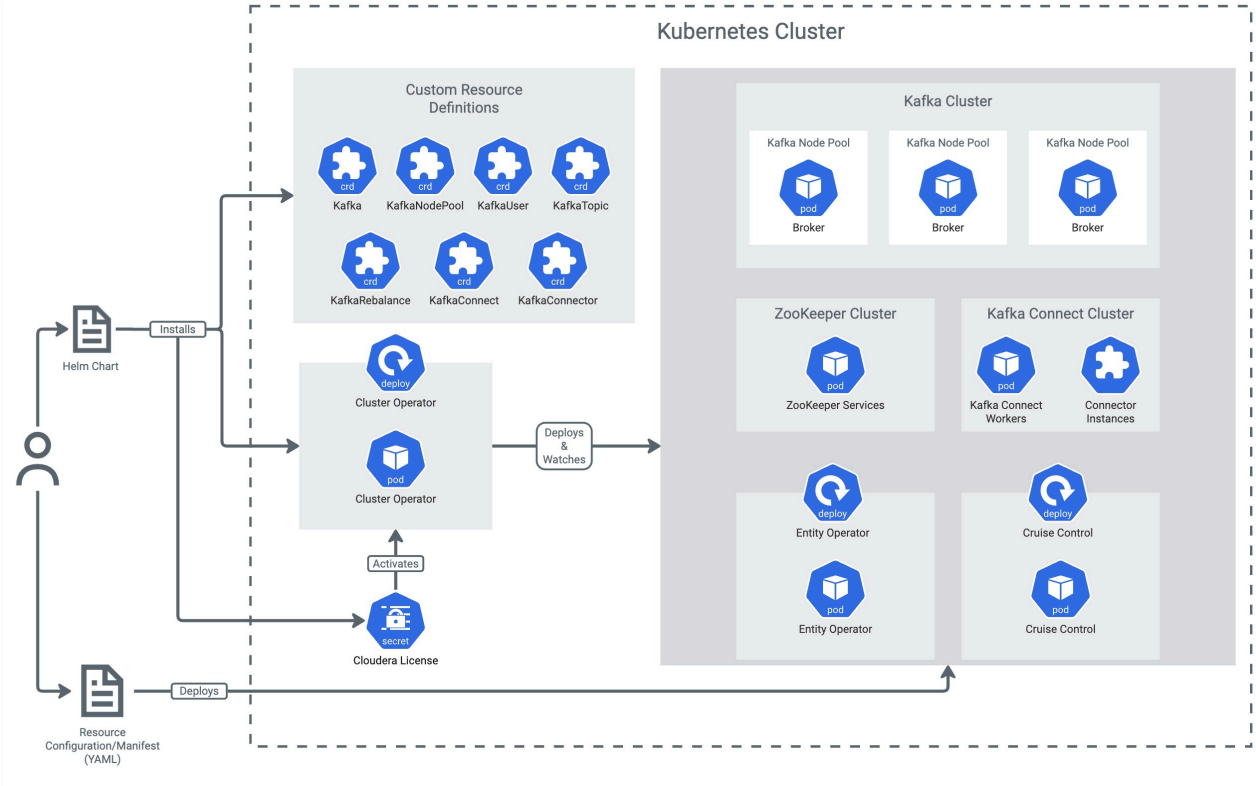
- Flexible, agile, and rapid deployment as well as scaling for variable workloads
- Standardization of deployments on existing Kubernetes infrastructure
- Operational efficiency with simple upgrades, and swift creation of new clusters.
- Ability to deploy Kafka and related components on existing, shared Kubernetes infrastructure. No need for dedicated infrastructure.
- Lightweight dependencies and system requirements for Kafka-centric deployments.

Related Information

Strimzi
Apache Kafka
Apache ZooKeeper
Cruise Control
Operator pattern | Kubernetes

Deployment architecture

Learn the architecture of a typical CSM Operator deployment.



CSM Operator deployment starts with first installing a Helm chart.

The Helm chart installs various Custom Resource Definitions (CRDs) to the Kuberentes environment. Resources described by these CRDs are managed by the Kubernetes operator applications. Strimzi defines CRDs such as:

- `Kafka` – represents a Kafka cluster consisting of Kafka brokers, ZooKeeper servers, Cruise Control and Strimzi Entity Operators.
- `KafkaNodePool` – represents a group of Kafka brokers from the cluster that have the same configuration.
- `KafkaTopic` – represents a Kafka topic.
- `KafkaUser` – represents a user that is external to the Kafka cluster.
- `KafkaRebalance` – represents a broker rebalance action for the Kafka cluster.
- `KafkaConnect` – represents a Kafka Connect cluster consisting of one or more Kafka Connect workers.
- `KafkaConnector` – represents a Kafka Connect connector instance.

Note: Strimzi includes CRDs not highlighted here. The use of these CRDs are either not supported in CSM Operator, or are for internal use by the Strimzi Cluster Operator.

When installing the Helm chart, a Strimzi Cluster Operator Kubernetes deployment is created with a Strimzi Cluster Operator pod running within the deployment. This application is responsible for monitoring cluster components and reconciling these components when their configuration changes. During installation you are required to register a license, which activates the Strimzi Cluster Operator.

Following installation, you deploy various custom resources in the cluster like `Kafka` and `KafkaNodePool` resources. Based on the configuration in the resource, the Strimzi Cluster Operator deploys clusters of the components described by the resources.

Specifically, `Kafka` and `KafkaNodePool` resources will deploy Kafka and ZooKeeper clusters. Optionally, if configured, the `Kafka` resource also deploys the Strimzi Entity Operator and Cruise Control. The Strimzi Entity Operator is responsible for managing other resources inside the particular Kafka cluster (topics, users, and so on), Cruise Control is used for rebalancing Kafka.

`KafkaConnect` and `KafkaConnector` resources are used to deploy Kafka Connect clusters and instances of Kafka Connect connectors.

Warning: Strimzi allows creating Kafka brokers by creating only a single `Kafka` resource. However, CSM Operator only supports creating Kafka brokers by creating `KafkaNodePool` resources. Node pools allow for more flexible deployments with easier scaling options. Moreover, certain features like rack awareness and scaling are limited without node pools. Broker creation using the `Kafka` resource only is deprecated, and results in unnecessary effort of migrating the deployment to use node pools.

Related Information

Overview | Strimzi

Licensing

CSM Operator requires a valid license to function. Licenses are made available to you together with your Cloudera credentials as part of your license and subscription agreement with Cloudera.

Licenses are registered during CSM Operator installation. They are stored in a Kubernetes secret. Licenses can be updated at any time.

Licenses are valid for a set period of time. Once the license expires, the cluster resources you deployed will continue to run. However, reconciliation of resources will be blocked. For example, failed pods will not be restarted, scaling your clusters will not be possible. In general, the control mechanisms in place that keep resources healthy will be blocked. This leads to deployed resources breaking down over time.

CSM Operator publishes various log entries and Kubernetes events related to your licenses.

For example, if your license expires or becomes invalid due to any reason, appropriate logs and events are published notifying you that there are issues with your license.

These logs and events are published for the Strimzi Cluster Operator deployment. You can check these logs and events with the following commands.

```
kubectl events deployments/strimzi-cluster-operator --namesp
ace [***NAMESPACE***]
```

```
kubectl logs deployment/strimzi-cluster-operator --names
pace [***NAMESPACE***]
```

Related Information

Updating a license

Sizing and performance considerations

Learn about ways you can size your deployment for optimal performance.

Kafka broker performance primarily depends on the IO bandwidth of the nodes and disks. Because of this, Cloudera recommends using SSDs with high IOPS and throughput for large workloads. JBOD can also lead to throughput improvements when the node IO bandwidth can support multiple disks.

When optimizing for large workloads, using HDDs and storage replication services such as Longhorn might add a significant performance overhead.

Depending on the characteristics of the workload, brokers might require a large memory pool to be able to serve fetch requests from the cache. Brokers might also require an increased CPU allocation to support compressed messages

ZooKeeper requires a small resource pool in most workloads, and scaling the cluster to more than three nodes usually provides no benefit.

Recommended minimum setup

Cloudera recommends the following cluster sizing as a baseline for small and medium workloads.

Container	Count	CPU (m)	Memory (MiB)	Notes
Strimzi Cluster Operator	2	1000	384	Required
Kafka Broker	3	8000	20480	Required for Kafka workloads.
ZooKeeper	3	4000	4096	Required for Kafka workloads.
Cruise Control	3	4000	4096	Required for rebalance operations.
Topic Operator	1	500	256	Required if you want to manage topics with `KafkaTopic` resources.
User Operator	1	500	256	Required if you want to manage Kafka users with `KafkaUser` resources.
Kafka Exporter	1	500	256	Required if you want to have additional broker and client metrics available.
Kafka Connect	3	4000	4096	Required if you want to use Kafka Connect and related functionality.