Cloudera Streams Messaging Operator for Kubernetes 1.6.0

Schema Registry Configuration

Date published: 2024-06-11
Date modified: 2026-01-27

https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Schema Registry configuration overview

Get started with configuring Schema Registry. Learn about basic configuration practices using Helm and available configuration properties.

Schema Registry is exclusively managed using Helm. You initially configure Schema Registry during installation. Typically this involves creating a custom values file (values.yml) that includes configuration properties. The file is applied during installation when you run the helm install command.

If required, you can make configuration updates following installation. This is done with the helm upgrade command using the --reuse-values option together with the -f (--values) or --set options.

```
helm upgrade SCHEMA-REGISTRY [***CHART***] \
  --namespace [***NAMESPACE***] \
  (--set '[***KEY***]=[***VALUE***]' | -f [***MY-VALUES.YAML***] | --set-fil
e [***KEY***]=[***FILEPATH***]) \
  --reuse-values
```

- The string *SCHEMA-REGISTRY* is the Helm release name of the chart installation. This is an arbitrary, user defined name.
- Ensure that *[***CHART***]* and *[***NAMESPACE***]* are the same as the ones you used during installation. You can use helm list to list currently installed releases and charts.
- Use --set if you want to update properties directly from the command line. Helm supports various --set options like --set-file, --set-string, and others. You can use any of these options.
- Use -f (--values) if you want to pass a file containing your configuration updates.
- The --reuse-values option instructs Helm to merge existing values with new ones. You use this option when you want to update an existing configuration.

Configurable properties

Schema Registry accepts various configuration properties. You can find a comprehensive list of these properties in the *Helm chart configuration reference*. Alternatively, you can list available properties with helm show readme.

```
helm show readme [***CHART***]
```

Related Information

Schema Registry Helm chart configuration reference

Configuring external access in Schema Registry

Learn how you can configure Schema Registry to make it accessible from outside of your Kubernetes cluster.

When installing Schema Registry with default values, a `ClusterIP` type Kubernetes `Service` is deployed. This provides access to Schema Registry from within the Kubernetes cluster.

To enable secure (TLS) external access, you can configure a Kubernetes `Ingress` on top of the `ClusterIP`. Alternatively, you can deploy a `LoadBalancer` type `Service` instead of the `ClusterIP`. Both methods allow you to provide secure external access to Schema Registry. The choice between `Ingress` and `LoadBalancer` depends on your infrastructure, security requirements, and need for advanced routing or certificate management.

Note: While both methods allow for both encrypted (TLS) and unencrypted communication. Cloudera recommends that you always enable encrypted communications with TLS for external access.

Configuring external access with Ingress

Learn how to configure external access to Schema Registry with Kubernetes Ingress.

Before you begin

- An Ingress controller is required. Ensure that you have one deployed in your Kubernetes cluster. For example, you can use the Ingress-Nginx controller.
- Optional: cert-manager is installed in your Kubernetes cluster.

  Although not required, cert-manager enables you to manage certificates automatically. Without cert-manager you must manage your certificate manually through Secrets. The following steps assume that cert-manager is available.

Procedure

1. Deploy an Issuer resource for cert-manager.

   Take note of the name of the Issuer you deploy. You provide the name of the Issuer to the Ingress in a following step. Deploying a Certificate resource is not needed, it is automatically requested and created by the Ingress once it is deployed.
2. Configure ingress properties in a values file (values.yaml).

   ```
   #...
   ingress:
     enabled: true
     className: "nginx"
     rules:
       host: MY-APP.EXAMPLE.CLOUDERA.COM
     tls:
       enabled: true
       secretRef: "[***INGRESS TLS CERT SECRET***]"
     extraAnnotations:
       cert-manager.io/issuer: "[***ISSUER NAME***]"
   ```

   - ingress.enabled – Enables or disables external access through Ingress.
   - ingress.rules.host – Specifies the DNS hostname that the Ingress controller should match for incoming HTTP/HTTPS requests.
   - ingress.tls.enabled – Enables or disables TLS for Ingress.
   - ingress.tls.secretRef – The name of the Secret containing Ingress TLS certificates.

     When using cert-manager and the cert-manager.io/issuer annotation is set in the ingress.extraAnnotations property, a certificate is requested automatically and saved to the Secret specified here.
   - ingress.extraAnnotations.* – Extra annotations to apply to the Ingress.

     The cert-manager.io/issuer annotation specified here contains the name of the cert-manager Issuer. When set, a certificate is automatically requested by the Ingress.
3. Apply configuration changes.

   ```
   helm upgrade SCHEMA-REGISTRY [***CHART***] \
     --namespace [***NAMESPACE***] \
     --values [***VALUES.YAML***] \
     --reuse-values
   ```
4. Access Schema Registry from the Hostname/IP of the Ingress.

   ```
   kubectl get ingress --namespace [***NAMESPACE***]
   ```

   ```
   NAME               CLASS   HOSTS                ADDRESS     PORTS
   ```

```
#...
schema-registry-ingress   nginx   my-app.example.cloudera.com  10.14.91.1
  80, 443
```

In this example, Schema Registry is accessed at my-app.example.cloudera.com.com:443.

Configuring external access with LoadBalancer

Learn how to configure external access to Schema Registry with a LoadBalancer type Service.

Before you begin

When deploying a `LoadBalancer` type `Service`, the actual load balancer is provisioned and managed by your cloud or infrastructure provider. As a result, TLS settings and certificate management may vary depending on the platform. Refer to vendor-specific documentation for detailed guidance on configuring TLS.

Procedure

1. Set service.type to LoadBalancer in a custom values file (values.yaml).

```
#...
service:
  type: LoadBalancer
  port: 9090
```

Note: Ensure that `Ingress` is disabled (ingress.enabled: false). `Ingress` is disabled by default.

2. Apply configuration changes.

```
helm upgrade SCHEMA-REGISTRY [***CHART***] \
  --namespace [***NAMESPACE***] \
  --values [***VALUES.YAML***] \
  --reuse-values
```

3. Access Schema Registry from the Hostname/IP of the load balancer.

```
kubectl get service SCHEMA-REGISTRY-service --namespace [***NAMESPACE***]
```

Look at the IP listed in the EXTERNAL-IP column as well as the port in the PORT(S) column. You can access Schema Registry using this IP and port.

```
NAME                    TYPE            CLUSTER-IP      EXTERNAL-IP        PORT
(S)
schema-registry-service  LoadBalancer   10.103.58.116   104.198.205.71
 9090:30219/TCP
```

In this example, Schema Registry is accessed at 104.198.205.71:30219.