Cloudera Streaming Analytics Operator 1.0.0

SSB Resource Management

Date published: 2024-06-15
Date modified: 2024-06-15

https://docs.cloudera.com/

Legal Notice

Contents

SSB service configurations

Configuration variables for the SQL Stream Builder (SSB).

The following configurations can be set in the values.yaml of the CSA Operator:

Configuration	Description
fernetKey	A 32-byte long string that is used to encrypt sensitive data, such as credentials and SQL queries.
userManagement	Enable and disable user registration, and configure default users to be registered automatically.
rbac	Use this if you decide not to use default RBAC resources to create your own before installing the CSA Operator.
service	The Service created for SSB-SSE. By default it is set to ClusterIP type.
ingress	Specification of the Ingress to be created.
storageConfiguration	Connection configuration to various storage solutions (for example S3). This data is stored as a KubernetesSecret, since it can contain sensitive values. Use this configuration to set up durable storage for Flink (for example, to create checkpoints). You can also use this configuration for artifact storage for SSB will to store UDFs and custom connectors.
podVolumes	Customized volume configuration for both SSB and Flink pods. You can specify ConfigMaps, Secrets to be mounted in these Pods. This configuration can be useful if you want to mount Hadoop configurations, such as hdfs-site.xml, core-site.xml or Kerberos keytabs.
podVolumeMounts	Mount point configuration for the Volumes created with podVolumes.
ssbConfiguration	Configuration overrides for SSB to set up remote artifact storage, job notifications, and sampling Kafka.
flinkConfiguration	Path of a default flink-conf.yaml that will be applied to all Flink deployments created by SSB.
database	By default, SSB will create a Postgres database, but you can override it with this property if you already have a database you wish to use with SSB data. Supported database types are MySQL/MariaDB, Oracle, and PostgreSQL.

Security configurations

Configuring Flink for securing your workloads and the SQL Stream Builder.

SSB enables you to deploy the Flink SQL jobs in an easier, however you need to ensure the proper security of your workloads and SSB. This means that the following tools are available for you to secure your `FlinkDeployment` in your Kubernetes cluster setup.

Fernet key

When deploying SSB, you must specify a fernetKey, which will be used for encrypting the job definition for all Flink jobs started with SSB. The job definitions may include sensitive data, such as the DDL of tables that can contain username and password as part of the connector configuration or authentication information for connected storages. Sensitive information will be encrypted by the Fernet key.

The specified Fernet key is created as a Kubernetes Secret in the same namespace where the CSA Operator is installed, and will be automatically mounted by SSB and Flink pods.

User management

When installing SSB, a default user (admin/admin) is created automatically, and registration is enabled. You have the option to enable or disable user registration, and you can also modify the default user(s) based on your requirements.

Ingress

By default SSB does not set up any Ingress. This can be changed using the ingress configuration. The Ingress resource is created in the same namespace as SSB.

Ingress can be used to easily enable TLS/HTTPs to SSB, but it can also be used to set up authentication.

Storage configuration

It is recommended to add some kind of persistent data storage for Flink to be able to save checkpoints and savepoints. In most cases this is some kind of blob storage (for example, S3) that needs authentication to access.

You can use the storageConfiguration configuration to set up the storage for SSB. The configuration should be a valid flink-conf.yaml file, which can contain sensitive data, such as s3.access-key, s3.secret-key, and so on.

The Helm chart creates a Secret in the same namespace as SSB, and SSB creates a new Secret for each new Flink deployment created by the user.

Volume mounts

It is possible to mount existing volumes to SSB and all created Flink pods using the podVolumes and podVolum eMounts configurations. You need to ensure that these volumes exist in the namespace of the SSB and Flink pods that will be created by SSB.

The configurations can be used to mount ConfigMaps, Secret, or any kind of volumes to the SSB and Flink pods. For example, you can mount hive-site.xml, core-site.xml, krb5.conf and some keytabs as ConfigMaps and Secrets to be able to connect to Hive with SSB/Flink.

Kerberos configuration

To enable Kerberos authentication, you need to add the Hadoop dependencies to the CSA images as described in the *Customize container images* section.

After adding the dependencies, you need to ensure that the Hadoop configuration and krb5.conf files are added as a configmap using the following commands:

```
kubectl -n flink create configmap hadoop-conf --from-file core-site.xml=core
-site.xml --from-file hdfs-site.xml=hdfs-site.xml
```

```
kubectl -n flink create configmap krb5-conf --from-file krb5.conf=krb5.conf
```

When the configmaps are in place, the following configuration properties should be updated in the values.yml file for the configuration files to be mounted on the containers:

```
podVolumes:
    create: true
    data:
      - name: hadoop-conf-volume
        configMap:
          name: hadoop-conf
      - name: krb5-conf-volume
        configMap:
          name: krb5-conf
  podVolumeMounts:
    create: true
    data:
```

```
      - name: hadoop-conf-volume
        mountPath: /etc/hadoop/conf
        readOnly: true
      - name: krb5-conf-volume
        mountPath: /etc/krb5.conf
        subPath: krb5.conf
```

After setting up images and configurations, you can use Streaming SQL Console to specify your keytabs in the Keytab Manager. Once the keytab is successfully validated, Kerberos will be automatically configured when a new job is deployed.

Related Information

Customize container images

Database configuration

Configure the SQL Stream Builder's database.

By default, the Helm chart automatically creates a PostgreSQL Deployment and Service using a public image. The Helm chart also creates a PersistentVolumeClaim, which will persist throughout Helm uninstalls and installs using the "helm.sh/resource-policy": "keep" annotation.

If you remove this PersistentVolumeClaim, on the next SSB restart all the data saved by SSB will be lost, and the database will be re-created.

It is recommended that you use an already existing external database, and configure SSB to connect to it, instead of using the default created database. You can use MySQL/MariaDB, Oracle or PostgreSQL as a database.

You can use the following example configuration to override the default database configuration:

```
database:
  create: false
  auth:
 type: postgresql
 jdbcUrl: "jdbc:postgresql://postgres-host:5432/ssb_admin"
 username: test_user
 password: test_password
```

The data provided here is kept secure as a Kubernetes secret.

Customize container images

Updating SSB images to use Kudu, Hive, HBase, and HDFS with the SQL Stream Builder.

To be able to use Kudu, Hive, HBase or HDFS, you need to update the images supplied to you, and add the required JAR files and dependencies using Dockerfiles.

There are two images you need to update, both of which can be found under the sqlRunner.image and sse.image configurations. sqlRunner.image is the image that will be used for the Flink deployments. This image is responsible for executing the SQL commands. sse.image is SSB itself.

If you want to use the updated container image, make sure to upload it to a registry your Kubernetes cluster can access, and update the configuration in the values.yaml file to point to your new images.

Here is an example of adding Hadoop and Hive to the SQL Runner image:

```
FROM [***REGISTRY HOST***]:[***PORT***]/[***PROJECT***]/ssb-sql-runner:lates
t
```

```
ENV CLOUDERA_ARCHIVES "https://archive.cloudera.com"
# Hadoop
ENV HADOOP_VERSION "3.1.1.7.1.9.0-387"
ENV HADOOP_HOME "/opt/hadoop"
RUN rm -rf ${HADOOP_HOME}/ \
    && cd /opt \
    && curl -sL --retry 3 "https://${CLOUDERA_ARCHIVES}/artifacts/build/447
02451/cdh/7.x/redhat8/yum/tars/hadoop/hadoop-client-${HADOOP_VERSION}.tar.gz
" | tar xz  \
    && chown -R root:root hadoop-client-${HADOOP_VERSION} \
    && ln -sfn hadoop-client-${HADOOP_VERSION} hadoop \
    && rm -rf ${HADOOP_HOME}/share/doc \
    && find /opt/ -name *-sources.jar -delete
ENV HADOOP_CONF_DIR "${HADOOP_HOME}/etc/hadoop"
ENV PATH="${HADOOP_HOME}/bin:${PATH}"
ENV HADOOP_CLASSPATH "/opt/hadoop/share/hadoop/client/lib/*"
# Hive
RUN wget https://${CLOUDERA_ARCHIVES}/maven/org/apache/flink/flink-sql-conne
ctor-hive-3.1.3_2.12/1.18.0-csaop1.0.0/flink-sql-connector-hive-3.1.3_2.12-1
.18.0-csaop1.0.0.jar \
    -O /opt/flink/lib/flink-sql-connector-hive-3.1.3_2.12-1.18.0-csaop1.0
.0.jar
```

Here is an example of adding Hadoop and Hive to the SSB image:

```
FROM [***REGISTRY HOST***]:[***PORT***]/[***PROJECT***]/ssb-sse:latest

ENV CLOUDERA_ARCHIVES "https://archive.cloudera.com"

ENV HADOOP_VERSION "3.1.1.7.1.9.0-387"
ENV HADOOP_HOME "/opt/hadoop"
RUN rm -rf ${HADOOP_HOME}/ \
    && cd /opt \
    && curl -sL --retry 3 "https://${CLOUDERA_ARCHIVES}/artifacts/build/447
02451/cdh/7.x/redhat8/yum/tars/hadoop/hadoop-client-${HADOOP_VERSION}.tar.gz
" | tar xz  \
    && chown -R root:root hadoop-client-${HADOOP_VERSION} \
    && ln -sfn hadoop-client-${HADOOP_VERSION} hadoop \
    && rm -rf ${HADOOP_HOME}/share/doc \
    && find /opt/ -name *-sources.jar -delete
ENV HADOOP_CONF_DIR "${HADOOP_HOME}/etc/hadoop"
ENV PATH="${HADOOP_HOME}/bin:${PATH}"

# Only copy Hadoop jars that are required for SSB to communicate with Hive
RUN cp "${HADOOP_HOME}/share/hadoop/client/lib/hadoop-common-${HADOOP_VER
SION}.jar" /opt/cloudera/ssb-sse/lib/ \
    && cp "${HADOOP_HOME}/share/hadoop/client/lib/hadoop-auth-${HADOOP_VERSI
ON}.jar" /opt/cloudera/ssb-sse/lib/ \
    && cp "${HADOOP_HOME}/share/hadoop/client/lib/hadoop-mapreduce-client-
core-${HADOOP_VERSION}.jar" /opt/cloudera/ssb-sse/lib/ \

# Hive
RUN wget https://${CLOUDERA_ARCHIVES}/maven/org/apache/flink/flink-sql-conn
ector-hive-3.1.3_2.12/1.18.0-csaop1.0.0/flink-sql-connector-hive-3.1.3_2.12-
1.18.0-csaop1.0.0.jar \
    -O /opt/cloudera/ssb-sse/lib/flink-sql-connector-hive-3.1.3_2.12-1.18.0.
jar
```