This repo is to deploy a self managed connect cluster,KSQLDB and control center backed confluent cloud and configuring centralized monitoring.

Please note that this not production ready repo, pick up the assets (grafana dashboards, prometheus config) and integrate with your k8 deployment.

1. Create namespace “confluent” in kubernetes cluster ( all the yaml files in this repo using “confluent” namespace for deployment)

*kubectl create namespace confluent*

1. Set “confluent” namespace as default-context to run the deployment

*kubectl config set-context --current --namespace=confluent*

1. Set the confluent-hybrid director for the directory you downloaded the repo

*export CONFLUENT\_HYBRID=<downloaded directory>/ccloud-integration*

1. Set up the Helm Chart

*helm repo add confluentinc* [*https://packages.confluent.io/helm*](https://packages.confluent.io/helm)

1. Install Confluent For Kubernetes using Helm:

*helm upgrade --install operator confluentinc/confluent-for-kubernetes*

1. Check that the Confluent For Kubernetes pod comes up and is running:

*kubectl get pods*

1. Generate a CA pair to use in this tutorial(this tutorial uses self-signed certificates):

*openssl genrsa -out $CONFLUENT\_HYBRID/ca-key.pem 2048*

*openssl req -new -key $CONFLUENT\_HYBRID/ca-key.pem -x509 \*

*-days 1000 \*

*-out $CONFLUENT\_HYBRID/ca.pem \*

*-subj "/C=US/ST=CA/L=MountainView/O=Confluent/OU=Operator/CN=TestCA"*

1. Create a Kubernetes secret for inter-component TLS:

*kubectl create secret tls ca-pair-sslcerts \*

*--cert=$CONFLUENT\_HYBRID/ca.pem \*

*--key=$CONFLUENT\_HYBRID/ca-key.pem*

1. Provide authentication credentials

Confluent Cloud provides you an API key for both Kafka and Schema Registry. Configure Confluent For Kubernetes to use the API key when setting up Connect and ksqlDB to connect.

Create a Kubernetes secret object for Confluent Cloud Kafka access. This secret object contains file based properties. These files are in the format that each respective Confluent component requires for authentication credentials.

*kubectl create secret generic cloud-plain \*

*--from-file=plain.txt=$CONFLUENT\_HYBRID/creds-client-kafka-sasl-user.txt*

*kubectl create secret generic cloud-sr-access \*

*--from-file=basic.txt=$CONFLUENT\_HYBRID/creds-schemaRegistry-user.txt*

*kubectl create secret generic control-center-user \*

*--from-file=basic.txt=$TUTORIAL\_HOME/creds-control-center-users.txt*

1. Deploy confluent platform

Edit the confluent-platform.yaml deployment YAML, and add your respective Confluent Cloud URLs in the following places:

* <cloudSR\_url>
* <cloudKafka\_url>

1. Validate

Set up port forwarding to Control Center web UI from local machine:

*kubectl port-forward controlcenter-0 9021:9021*

Browse to Control Center and log in as the admin user with the Developer1 password:

[*https://localhost:9021*](https://localhost:9021)

## Monitoring

Confluent cloud cluster metrics set up:

* Create secret file for ccloud metrics api

export CCLOUD\_CLUSTER=<<confluent cloud cluster name>>

export CCLOUD\_API\_KEY=<<confluent cloud api key>>

export CCLOUD\_API\_SECRET=<<confluent cloud api secret>>

Note: Service account associated with APIKey should have metrics viewer role to pull the metrics from confluent cloud. Refer this to grant metrics viewer role access.

Run the below command to create ccloud exporter secret file

kubectl create secret generic $CCLOUD\_CLUSTER-vars --from-literal=CCLOUD\_API\_KEY=${CCLOUD\_API\_KEY} --from-literal=CCLOUD\_API\_SECRET=${CCLOUD\_API\_SECRET} --from-literal=CCLOUD\_CLUSTER=${CCLOUD\_CLUSTER}

* Replace ccloud cluster id in deployment yaml file

sed s/%CCLOUD\_CLUSTER%/$CCLOUD\_CLUSTER/g \

$CONFLUENT\_HYBRID/monitoring/ccloud\_exporter.yaml > $CONFLUENT\_HYBRID/monitoring/ccloud\_exporter-$CCLOUD\_CLUSTER.yaml

* Deploy ccloud\_exporter

kubectl apply -f $CONFLUENT\_HYBRID/monitoring/ccloud\_exporter-$CCLOUD\_CLUSTER.yaml

kafka-lag exporter setup(monitoring consumer lag):

kafka lag exporter is an open source utility to monitor the consumer lag and publish metrics in open metrics format(promethues specific)

* Update the $CONFLUENT\_HYBRID/monitoring/config/application.conf with confluent cloud bootstrap url, api key and secret (refer to the $CONFLUENT\_HYBRID/monitoring/config/samples/application\_sample.conf)
* Create configmap with monitoring configuration

kubectl create configmap monitor-config --from-file=$CONFLUENT\_HYBRID/monitoring/config

* Deploy kafka lag exporter

kubectl apply -f $CONFLUENT\_HYBRID/monitoring/kafkalag.yaml

Deploy node exporter( monitor node level metrics):

kubectl apply -f $CONFLUENT\_HYBRID/monitoring/node-exporter.yaml

Prometheus and Grafana set up:

* Create configmaps with Prometheus and Grafana dashboard files (data source and dashboards)

kubectl create configmap grafana-datasource --from-file=$CONFLUENT\_HYBRID/monitoring/config/grafana/datasources

kubectl create configmap dashboard-config --from-file=$CONFLUENT\_HYBRID/monitoring/config/grafana/dashboards

* Deploy Prometheus and Grafana

kubectl apply -f $CONFLUENT\_HYBRID/monitoring/prometheus-grafana.yaml