Central ControlCenter to monitor Multiple Clusters

The idea is to deploy a single C3 in a different namespace which is dedicated only for the C3. This C3 will be used to monitor all the CFK components present in that GKE cluster.

I am pasting below links of confluent documentation that would help to understand

1. https://docs.confluent.io/operator/current/co-monitor-cp.html#configure-c3-short-to-monitor-ak-clusters

2. https://docs.confluent.io/operator/current/co-monitor-cp.html#configure-c3-short-to-monitor-ksqldb-kconnect-and-sr-clusters

3. https://docs.confluent.io/platform/current/control-center/topics/schema.html#enabling-multi-cluster-sr

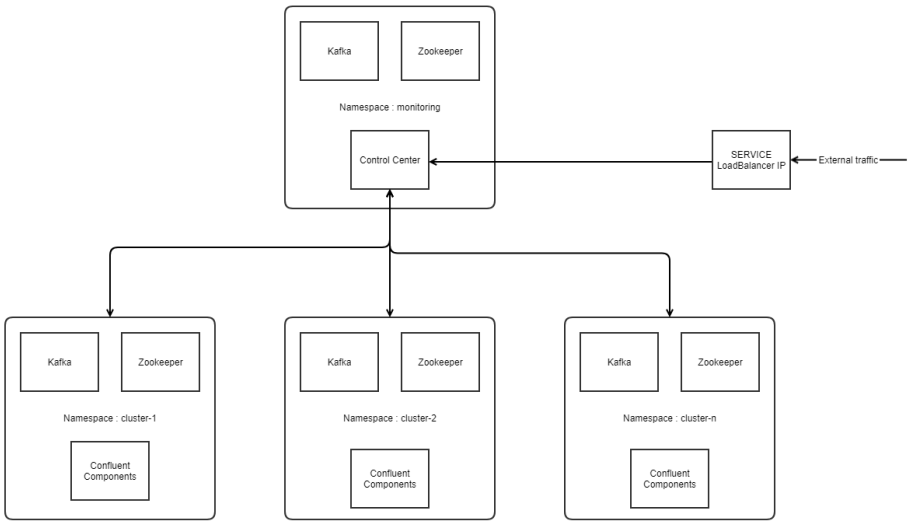
Implementation

1. Create a new namespace for C3 deployment along with kafka and zookeeper.
2. Authenticate kafka and zookeeper with "autoGeneratedCerts" certificates.

```
#set the below in kafka and zookeeper

.spec.tls.autoGeneratedCerts: true

#Generate a CA pair to use:

1. Generate the private key for CA

openssl genrsa -out root-ca-key.pem 2048

2. Generate self-signed root CA

openssl req -new -key root-ca-key.pem -x509 \
  -days 1000 \
  -out $TUTORIAL_HOME/root-ca.pem \
  -subj "/C=US/ST=CA/L=MountainView/O=Confluent/OU=Operator/CN=TestCA"

#Generate a Server key pair

1. Generate a private key for the server

openssl genrsa -out server-private-key.pem 2048

2. Create a CSR (Certificate Signing Request)

openssl req -new -key server-private-key.pem -out server-certificate.csr

3. Sign the CSR using the root CA

openssl x509 -req -in server-certificate.csr -CA root-ca.pem -CAkey root-ca-key.pem -CAcreateserial -out
server-client-certificate.crt -sha256


#Create a Kubernetes secret for the certificate authority:

kubectl create secret tls ca-pair-sslcerts \
  --cert=root-ca.pem \
  --key=root-ca-key.pem -n {namespace}

#Create a secret for other namespaces kafka to connect with kafka of kafka-system namespace where C3 lies

kubectl create secret generic tls-kafka-system-internal --from-file=fullchain.pem=server-client-
certificate.crt --from-file=cacerts.pem=root-ca.pem --from-file=privkey.pem=server-private-key.pem -o
yaml --dry-run=client | kubectl -n confluent apply -f -

#Update the below properties in confluent operator values file

managedCerts.enable: true and set managedCerts.caCertificate.secretRef=ca-pair-sslcerts

#install operator with kraft enabled
--set kRaftEnabled=true

# below certs need to be created in C3 cluster
kubectl -n kafka-system create secret generic tls-cc-external \
--from-file=cacerts.pem=root+subca.pem \
--from-file=fullchain.pem=controlcenter-tef.pem \
--from-file=privkey.pem=controlcenter.key \
--save-config --dry-run=client -o yaml | \
kubectl apply -f -
```

3. Authenticate C3 with a particular USER certificate , and provide ACL to the USER on topics and groups.
4. Create a new Zone for C3 namespace.

Centralized Control Center Structure



Sample C3 configuration

```
apiVersion: platform.confluent.io/v1beta1
kind: ControlCenter
metadata:
  name: controlcenter
  namespace: monitoring
spec:
  configOverrides: -----[1]
    server:
      - confluent.controlcenter.streams.cprest.url=http://kafka.monitoring.svc.cluster.local:8090
      - confluent.controlcenter.kafka.kafka-confluent.cprest.url=http://kafka.confluent.svc.cluster.local:8090
      - confluent.controlcenter.kafka.kafka-monitoring-dev.cprest.url=http://kafka.monitoring-dev.svc.cluster.
local:8090
  replicas: 1
  image:
    application: europe-west3-docker.pkg.dev/tefde-gcp-raitcs02-dev/emp/cp-enterprise-control-center:7.5.2
    init: europe-west3-docker.pkg.dev/tefde-gcp-raitcs02-dev/emp/confluent-init-container:2.7.2
  injectAnnotations:
      networking.gke.io/load-balancer-type: Internal
      networking.gke.io/internal-load-balancer-allow-global-access: "true"
  dataVolumeCapacity: 10Gi
  license:
      secretRef: confluent-license
  tls:
      secretRef: tls-cc-internal
  monitoringKafkaClusters: -----[2]
  - name: kafka-confluent
    bootstrapEndpoint: kafka.confluent.svc.cluster.local:9071
    authentication:
      type: mtls
    tls:
      enabled: true
  - name: kafka-monitoring-dev
    bootstrapEndpoint: kafka.monitoring-dev.svc.cluster.local:9071
  dependencies: -----[3]
    kafka:  -----[i]
      authentication:
        type: mtls
      bootstrapEndpoint: kafka.monitoring.svc.cluster.local:9071
      tls:
        enabled: true
```

```
  schemaRegistry: -----[ii]
    clusters:
    - name: schemaregistry
      tls:
        enabled: true
      url: https://schemaregistry.confluent.svc.cluster.local:8081
    - name: schemaregistry-monitoring-dev
      url: http://schemaregistry.monitoring-dev.svc.cluster.local:8081
    tls:
      enabled: true
    url: https://schemaregistry.monitoring.svc.cluster.local:8081
  ksqldb: -----[iii]
  - name: ksql
    tls:
        enabled: true
    url: http://ksqldb.monitoring.svc.cluster.local:8088
  - name: ksql-confluent
    tls:
        enabled: true
    url: https://ksqldb.confluent.svc.cluster.local:8088
  connect: -----[iv]
  - name: connect
    tls:
        enabled: true
    url:  http://connect.monitoring.svc.cluster.local:8083
  - name: connect-confluent
    tls:
        enabled: true
    url:  https://connect.confluent.svc.cluster.local:8083
  - name: connect-monitoring-dev
    url:  http://connect.monitoring-dev.svc.cluster.local:8083
externalAccess:
     loadBalancer:
       domain: raittcs01.emp-dev.gcp.de.pri.o2.com
     type: loadBalancer
```

(Refer the points numbering in the above yaml file to understand)

1. Under configOverrides add the below server properties in C3

   ```
   - confluent.controlcenter.streams.cprest.url=http://kafka.monitoring.svc.cluster.local:8090 #This is the
   URL of C3 kafka cluster
   - confluent.controlcenter.kafka.kafka-confluent.cprest.url=http://kafka.confluent.svc.cluster.local:8090
   #This is the URL of the kafka cluster monitored from C3
   - confluent.controlcenter.kafka.kafka-monitoring-dev.cprest.url=http://kafka.monitoring-dev.svc.cluster.
   local:8090 #This is the URL of the kafka cluster monitored from C3
   ```

2. Under this property add all the kafka clusters details in arrays format , that are to be monitored except the kafka cluster associated with the C3.
3. Add the below under "dependencies"
   i) Under the "kafka" section add kafka cluster details that is associated with the C3.
   ii) Under ".schemaRegistry" need to add a default URL (this is mandatory) , add other schemaregistry URLs under ".schemaRegistry.clusters" in array format.
   iii) Add all the ksqldb clusters that are to be monitored under under the ".ksqldb" in array format.
   iv) Add all the connect clusters that are to be monitored under under the ".connect" in array format.

Properties to add in kafka broker

1. Add the below properties in all the kafka server properties that are to be monitored.
   - confluent.http.server.listeners=http://kafka.monitoring.svc.cluster.local:8090 #This refers to the kafka REST Endpoint URL of that particular kafka cluster
   - confluent.schema.registry.url=https://schemaregistry.monitoring.svc.cluster.local:8081 #This refers to the schemaregistry URL that is associated with that particular kafka cluster.

   ```
   - super.users=User:controlcenter.kafka-system.emp.gcp.de.pri.o2.com #This is the user name (CN) of the C3
   kafka cluster
   ```

2. Add the below properties under ".spec"

```
metricReporter:
    enabled: true
    bootstrapEndpoint: kafka.monitoring.svc.cluster.local:9071 #Mention the kafka URL that is associated with the C3
    authentication:
      type: mtls
    tls:
      enabled: true

      secretRef: tls-kafka-system-internal #This certificates are used to communicate with the kafka in kafka-system namespace to send broker
metrics
```