_установка kafka на openshift

21 ноября 2020 г.	14:11

https://docs.confluent.io/5.3.0/installation/operator/co-deployment.html

https://docs.confluent.io/operator/current/co-deployment.html

https://docs.confluent.io/operator/current/co-openshift.html

kafka strimzi

https://blog.mimacom.com/strimzi-okd/
https://developers.redhat.com/blog/2018/10/29/how-to-run-kafka-on-openshift-the-enterprise-kubernetes-with-amq-streams/

https://blog.kubernauts.io/apache-kafka-on-kubernetes-4425e18daba5

https://www.nearform.com/blog/benchmarking-apache-kafka-deployed-on-openshift-with-helm/

https://github.com/confluentinc/dhakne-blogs/blob/master/providers/oc-aws.yaml

confluent kafka

wildcard DNS is set to *.apps.<cluster name>.<base domain> by default.

kafka.apps-crc.testing
kafka.crc.testing

http://www.masterspringboot.com/various/apache-kafka/accessing-apache-kafka-on-openshift-using-its-rest-api

loadbalancer

https://docs.openshift.com/container-platform/4.6/networking/configuring_ingress_cluster_traffic/configuring-externalip.html

https://docs.openshift.com/container-platform/4.6/networking/configuring_ingress_cluster_traffic/configuring-ingress-cluster-traffic-load-balancer.html

проблемы установки

https://github.com/strimzi/strimzi-kafka-operator/issues/912

```
create Pod zookeeper-0 in StatefulSet zookeeper failed error: pods "zookeeper-0" is forbidden: unable to validat
e against any security context constraint: [provider confluent-scc: .spec.securityContext.fsGroup: Invalid value: []int64{1001}: 1001 is not an allowed group spec.initContainers[0].secu
rityContext.runAsUser: Invalid value: 1001: must be in the ranges: [1002580000, 1002590000] spec.containers[0].securityContext.runAsUser: Invalid value: 1001: must be in the ranges: [10
02580000, 1002590000] provider restricted: .spec.securityContext.fsGroup: Invalid value: []int64{1001}: 1001 is not an allowed group spec.initContainers[0].securityContext.runAsUser: In
valid value: 1001: must be in the ranges: [1000600000, 1000609999] spec.containers[0].securityContext.runAsUser: Invalid value: 1001: must be in the ranges: [1000600000, 1000609999]]
```

https://github.com/prometheus-operator/prometheus-operator/issues/2333
https://github.com/jenkinsci/kubernetes-operator/issues/70
https://adam.younglogic.com/2017/06/creating-a-privileged-container-in-openshift/

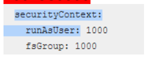
файл настроек

https://docs.confluent.io/5.3.0/installation/operator/co-endpoints.html#co-openshift-routes

https://github.com/kubernauts/kafka-confluent-platform

https://portworx.com/run-ha-kafka-red-hat-openshift/

https://github.com/codecentric/helm-charts/issues/203
1002580000

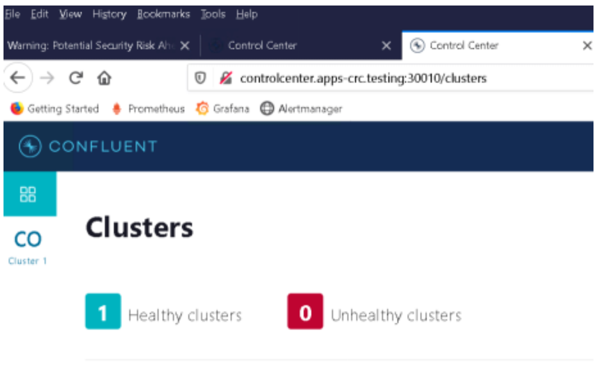


```
  securityContext:
    runAsUser: 1000
    fsGroup: 1000
```

полный скрипт

```
oc login -u kubeadmin -p hxYtC-KLeQC-kfNkm-ppi8i https://api.crc.testing:6443
powershell
oc new-project confluent
cd C:\Users\vovan\confluent\helm




helm upgrade --install  operator .\confluent-operator  --values $env:VALUES_FILE --namespace confluent --set operator.enabled=true
    oc get pods -n confluent | findstr cc-operator
    oc get crd | findstr confluent

oc create -f scripts\openshift\customUID\scc.yaml
oc delete -f scripts\openshift\customUID\scc.yaml
oc apply -f scripts\openshift\randomUID\scc.yaml





helm upgrade --install  zookeeper  .\confluent-operator --values $env:VALUES_FILE  --namespace confluent  --set zookeeper.enabled=true
    oc get zookeeper -n confluent | findstr zookeeper
    oc describe zookeeper zookeeper -n confluent
    oc get zookeeper zookeeper -o yaml -n confluent
    oc get zookeeper zookeeper -ojsonpath='{.status.phase}' -n confluent
    oc get statefulset.apps/zookeeper -o yaml
    oc describe statefulset.apps/zookeeper

    oc edit statefulset.apps/zookeeper  ==1002580000

    oc exec -ti zookeeper-0 bash
    id
    id=1002580000(1002580000) gid=0(root) groups=0(root),1002580000


helm upgrade --install  kafka  .\confluent-operator  --values $env:VALUES_FILE  --namespace confluent  --set kafka.enabled=true

    oc get pods -n confluent
    oc get kafka -n confluent
    oc get kafka kafka -n confluent -oyaml
    oc -n confluent get kafka kafka -ojsonpath='{.status.replicationFactor}'
    oc describe statefulset.apps/kafka

    oc edit statefulset.apps/kafka  ==1002580000



helm upgrade --install  schemaregistry  .\confluent-operator  --values $env:VALUES_FILE  --namespace confluent  --set schemaregistry.enabled=true

    oc describe statefulset.apps/schemaregistry
    oc edit statefulset.apps/schemaregistry  ==1002580000

helm upgrade --install  connectors  .\confluent-operator  --values $env:VALUES_FILE  --namespace confluent --set connect.enabled=true

    oc edit statefulset.apps/connectors




helm upgrade --install  replicator  .\confluent-operator  --values $env:VALUES_FILE  --namespace confluent  --set replicator.enabled=true

    oc edit statefulset.apps/replicator

helm upgrade --install  controlcenter  .\confluent-operator  --values $env:VALUES_FILE  --namespace confluent  --set controlcenter.enabled=true

    oc edit statefulset.apps/controlcenter

helm upgrade --install  ksql  .\confluent-operator --values $env:VALUES_FILE  --namespace confluent  --set ksql.enabled=true

    oc edit statefulset.apps/ksql






oc get kafka kafka -n confluent -oyaml

oc get service/controlcenter-bootstrap-lb -oyaml
oc describe service/controlcenter-bootstrap-lb

oc get service/kafka-0-lb -oyaml
oc describe service/kafka-0-lb

oc get service/kafka-bootstrap-lb -oyaml
oc describe service/kafka-bootstrap-lb
```

законектимся в админку

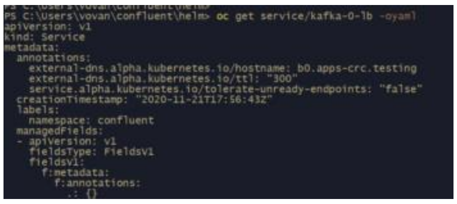

File Edit View History Bookmarks Tools Help
Warning: Potential Security Risk Ah
Control Center
Control Center
controlcenter.apps-crc.testing:30010/clusters
Getting Started
Prometheus
Grafana
Alertmanager
CONFLUENT
CO
Cluster 1

Clusters

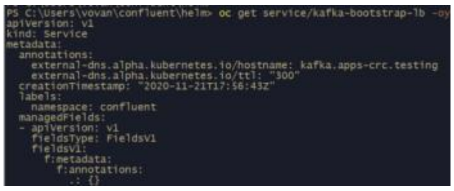
1 Healthy clusters

0 Unhealthy clusters

законектимся вариант1



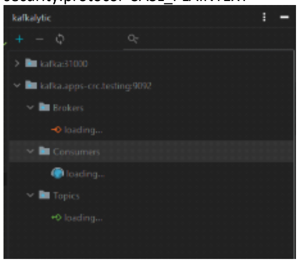
```
PS C:\Users\vovan\confluent\helm> oc get service/kafka-0-lb -oyaml
apiVersion: v1
kind: Service
metadata:
  annotations:
    external-dns.alpha.kubernetes.io/hostname: b0.apps-crc.testing
    external-dns.alpha.kubernetes.io/ttl: "300"
    service.alpha.kubernetes.io/tolerate-unready-endpoints: "false"
  creationTimestamp: "2020-11-21T17:56:43Z"
  labels:
    namespace: confluent
  managedFields:
  - apiVersion: v1
    fieldsType: FieldsV1
    fieldsV1:
      f:metadata:
        f:annotations:
          .: {}
```



```
PS C:\Users\vovan\confluent\helm> oc get service/kafka-bootstrap-lb -oy
apiVersion: v1
kind: Service
metadata:
  annotations:
    external-dns.alpha.kubernetes.io/hostname: kafka.apps-crc.testing
    external-dns.alpha.kubernetes.io/ttl: "300"
  creationTimestamp: "2020-11-21T17:56:43Z"
  labels:
    namespace: confluent
  managedFields:
  - apiVersion: v1
    fieldsType: FieldsV1
    fieldsV1:
      f:metadata:
        f:annotations:
          .: {}
```

```
authenticationType: PLAIN
  bootstrapEndpoint: kafka.apps-crc.testing:9092
  brokerEndpoints:
    kafka-0: b0.apps-crc.testing:9092
  brokerExternalListener: SASL_PLAINTEXT:9092
  brokerInternalListener: SASL_PLAINTEXT:9071
  clusterName: kafka
  currentReplicas: 1
  externalClient: |-
    bootstrap.servers=kafka.apps-crc.testing:9092
    sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username=<<sasl_username>> password=<<sasl_password>>;
    sasl.mechanism=PLAIN
    security.protocol=SASL_PLAINTEXT
  internalClient: |-
    bootstrap.servers=kafka:9071
    sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username=<<sasl_username>> password=<<sasl_password>>;
    sasl.mechanism=PLAIN
    security.protocol=SASL_PLAINTEXT
  jmxPort: 7203
  jolokiaPort: 7777
  minIsr: 1
  phase: RUNNING
  prometheusPort: 7778
  pscVersion: 1.0.0
  readyReplicas: 1
  replicas: 1
  replicationFactor: 1
  zookeeperConnect: zookeeper.confluent.svc.cluster.local:2181/kafka-confluent
```

```
bootstrap.servers=kafka.apps-crc.testing:9092
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="test" password="test123";
sasl.mechanism=PLAIN
security.protocol=SASL_PLAINTEXT
```



законектимся вариант2

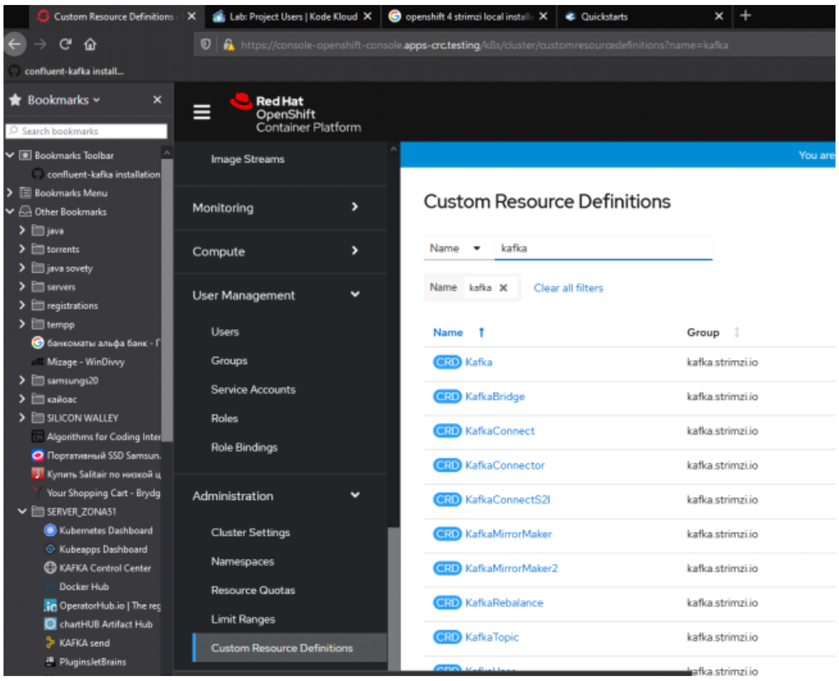
```
status:
  authenticationType: PLAIN
  bootstrapEndpoint: kafka:31000
  brokerEndpoints:
    kafka-0: kafka:31002
  brokerExternalListener: SASL_PLAINTEXT:31000
  brokerInternalListener: SASL_PLAINTEXT:9071
  clusterName: kafka
  currentReplicas: 1
  externalClient: |-
    bootstrap.servers=kafka:31000
    sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username=<<sasl_username>> password=<<sasl_password>>;
    sasl.mechanism=PLAIN
    security.protocol=SASL_PLAINTEXT
  internalClient: |-
    bootstrap.servers=kafka:9071
    sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username=<<sasl_username>> password=<<sasl_password>>;
    sasl.mechanism=PLAIN
    security.protocol=SASL_PLAINTEXT
  jmxPort: 7203
  jolokiaPort: 7777
  minIsr: 1
  phase: RUNNING
  prometheusPort: 7778
  pscVersion: 1.0.0
  readyReplicas: 1
  replicas: 1
  replicationFactor: 1
  zookeeperConnect: zookeeper.confluent.svc.cluster.local:2181/kafka-confluent
```

как выставить порты

https://developers.redhat.com/blog/2019/06/07/accessing-apache-kafka-in-strimzi-part-2-node-ports/

https://strimzi.io/quickstarts/
https://redhat-developer-demos.github.io/kafka-tutorial/kafka-tutorial/1.0.x/10-kubernetes.html

https://snourian.com/kafka-kubernetes-strimzi-part-1-creating-deploying-strimzi-kafka/


Custom Resource Definitions
Lab: Project Users | Kode Kloud
openshift 4 strimzi local install
Quickstarts
https://console-openshift-console.apps-crc.testing/k8s/cluster/customresourcedefinitions?name=kafka
Bookmarks
Red Hat OpenShift Container Platform
Image Streams
Monitoring
Compute
User Management
Users
Groups
Service Accounts
Roles
Role Bindings
Administration
Cluster Settings
Namespaces
Resource Quotas
Limit Ranges
Custom Resource Definitions

Custom Resource Definitions

Name kafka

Name kafka ✕ Clear all filters

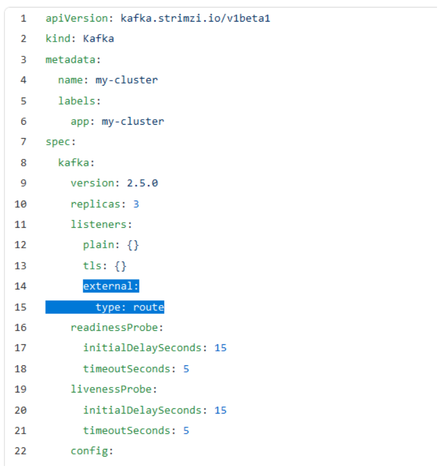
Name	Group
CRD Kafka	kafka.strimzi.io
CRD KafkaBridge	kafka.strimzi.io
CRD KafkaConnect	kafka.strimzi.io
CRD KafkaConnector	kafka.strimzi.io
CRD KafkaConnectS2I	kafka.strimzi.io
CRD KafkaMirrorMaker	kafka.strimzi.io
CRD KafkaMirrorMaker2	kafka.strimzi.io
CRD KafkaRebalance	kafka.strimzi.io
CRD KafkaTopic	kafka.strimzi.io

routes variant

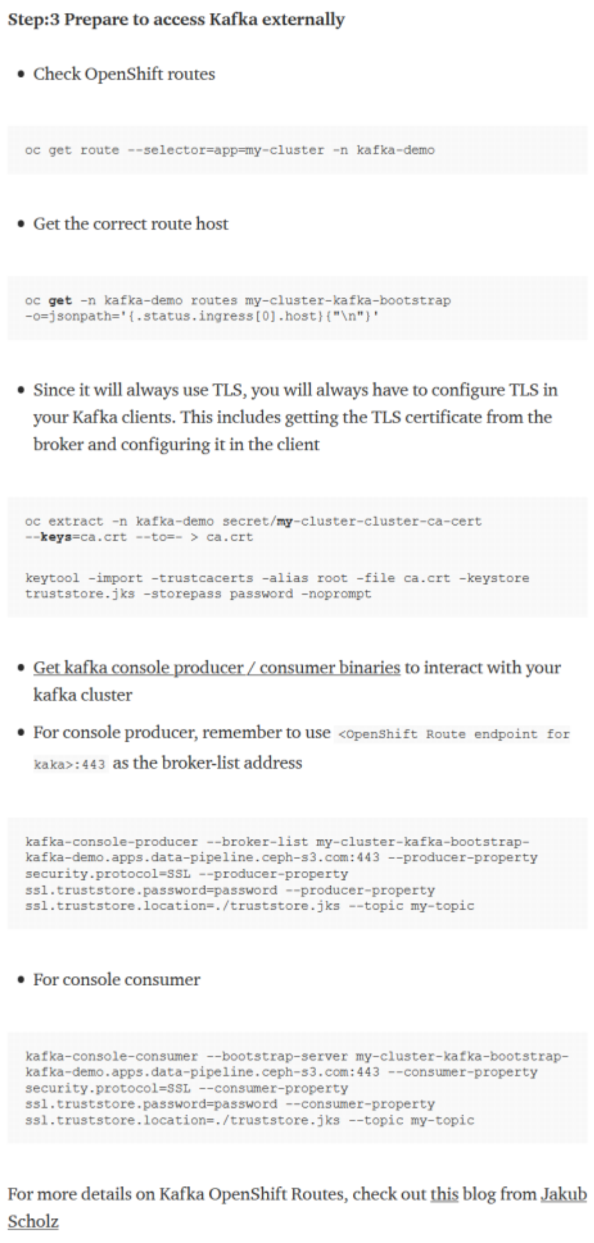
https://strimzi.io/blog/2019/04/30/accessing-kafka-part-3/

https://medium.com/@karansingh010/kafka-on-openshift-with-external-routes-4d328058667c

https://stackoverflow.com/questions/56137835/externally-accessing-kafka-on-openshift
https://strimzi.io/docs/operators/latest/using.html#con-kafka-listeners-deployment-configuration-kafka



```yaml
apiVersion: kafka.strimzi.io/v1beta1
kind: Kafka
metadata:
  name: my-cluster
  labels:
    app: my-cluster
spec:
  kafka:
    version: 2.5.0
    replicas: 3
    listeners:
      plain: {}
      tls: {}
      external:
        type: route
    readinessProbe:
      initialDelaySeconds: 15
      timeoutSeconds: 5
    livenessProbe:
      initialDelaySeconds: 15
      timeoutSeconds: 5
    config:
```



Step:3 Prepare to access Kafka externally

- Check OpenShift routes

```
oc get route --selector=app=my-cluster -n kafka-demo
```

- Get the correct route host

```
oc get -n kafka-demo routes my-cluster-kafka-bootstrap
-o=jsonpath='{.status.ingress[0].host}{"\n"}'
```

- Since it will always use TLS, you will always have to configure TLS in your Kafka clients. This includes getting the TLS certificate from the broker and configuring it in the client

```
oc extract -n kafka-demo secret/my-cluster-cluster-ca-cert
--keys=ca.crt --to=- > ca.crt

keytool -import -trustcacerts -alias root -file ca.crt -keystore
truststore.jks -storepass password -noprompt
```

- Get kafka console producer / consumer binaries to interact with your kafka cluster
- For console producer, remember to use `<OpenShift Route endpoint for kaka>:443` as the broker-list address

```
kafka-console-producer --broker-list my-cluster-kafka-bootstrap-
kafka-demo.apps.data-pipeline.ceph-s3.com:443 --producer-property
security.protocol=SSL --producer-property
ssl.truststore.password=password --producer-property
ssl.truststore.location=./truststore.jks --topic my-topic
```

- For console consumer

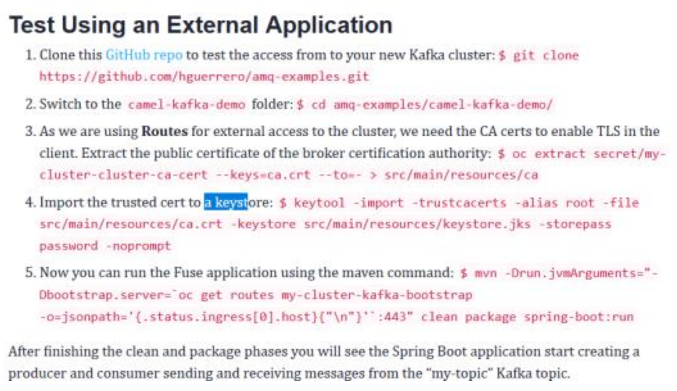
```
kafka-console-consumer --bootstrap-server my-cluster-kafka-bootstrap-
kafka-demo.apps.data-pipeline.ceph-s3.com:443 --consumer-property
security.protocol=SSL --consumer-property
ssl.truststore.password=password --consumer-property
ssl.truststore.location=./truststore.jks --topic my-topic
```

For more details on Kafka OpenShift Routes, check out this blog from Jakub Scholz

https://kafka.apache.org/quickstart

еще пример

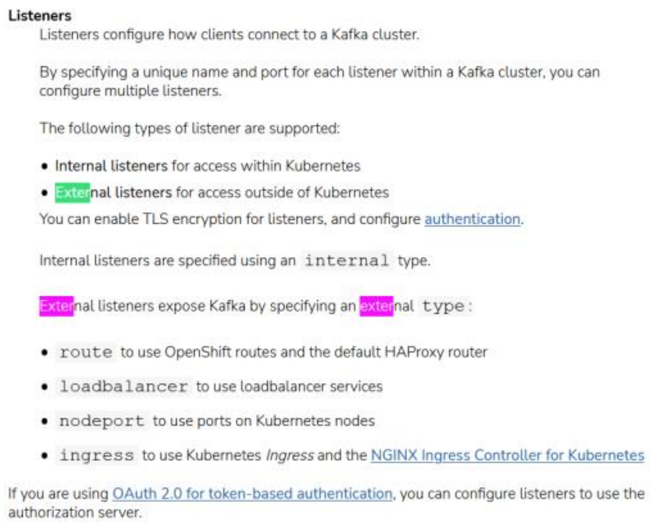
https://dzone.com/articles/how-to-run-kafka-on-openshift-the-enterprise-kuber
https://github.com/hguerrero/amq-examples



Test Using an External Application

1. Clone this GitHub repo to test the access from to your new Kafka cluster: `$ git clone https://github.com/hguerrero/amq-examples.git`
2. Switch to the `camel-kafka-demo` folder: `$ cd amq-examples/camel-kafka-demo/`
3. As we are using **Routes** for external access to the cluster, we need the CA certs to enable TLS in the client. Extract the public certificate of the broker certification authority: `$ oc extract secret/my-cluster-cluster-ca-cert --keys=ca.crt --to=- > src/main/resources/ca`
4. Import the trusted cert to a keystore: `$ keytool -import -trustcacerts -alias root -file src/main/resources/ca.crt -keystore src/main/resources/keystore.jks -storepass password -noprompt`
5. Now you can run the Fuse application using the maven command: `$ mvn -Drun.jvmArguments="-Dbootstrap.server=`oc get routes my-cluster-kafka-bootstrap -o=jsonpath='{.status.ingress[0].host}{"\n"}'`:443" clean package spring-boot:run`

After finishing the clean and package phases you will see the Spring Boot application start creating a producer and consumer sending and receiving messages from the "my-topic" Kafka topic.



Listeners

Listeners configure how clients connect to a Kafka cluster.

By specifying a unique name and port for each listener within a Kafka cluster, you can configure multiple listeners.

The following types of listener are supported:

- **Internal listeners** for access within Kubernetes
- **External listeners** for access outside of Kubernetes

You can enable TLS encryption for listeners, and configure authentication.

Internal listeners are specified using an `internal` type.

External listeners expose Kafka by specifying an external `type`:

- `route` to use OpenShift routes and the default HAProxy router
- `loadbalancer` to use loadbalancer services
- `nodeport` to use ports on Kubernetes nodes
- `ingress` to use Kubernetes *Ingress* and the NGINX Ingress Controller for Kubernetes

If you are using OAuth 2.0 for token-based authentication, you can configure listeners to use the authorization server.

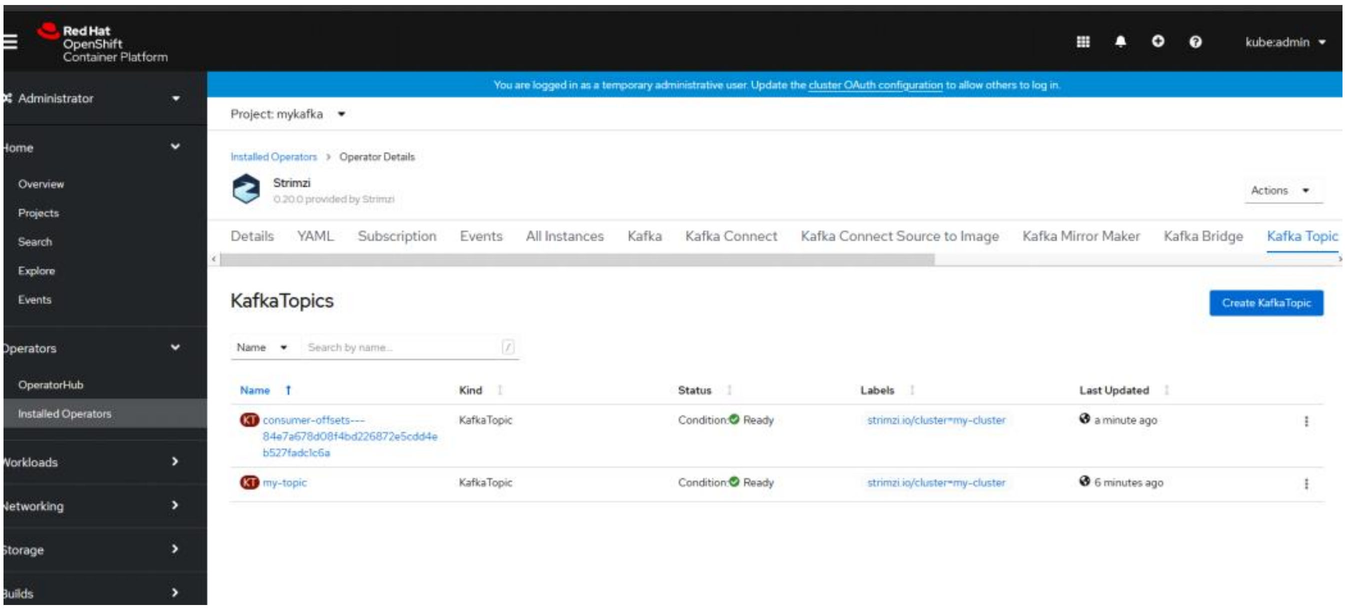
```
- name: external1
    port: 9094
    type: route
    tls: true
    authentication:
      type: tls


  - name: external
    port: 9094
    type: nodeport
    tls: false
```

https://strimzi.io/docs/operators/master/quickstart.html
https://strimzi.io/examples/latest/kafka/kafka-persistent-single.yaml
https://gist.githubusercontent.com/ksingh7/61d5a62c9885078719cc16b260d107c9/raw/7b4f90259877f73850f356fd7a7e35b1a08f1e00/01_kafka_cluster.yaml

```
apiVersion: kafka.strimzi.io/v1beta1
kind: Kafka
metadata:
  name: my-cluster
  namespace: mykafka
spec:
  kafka:
    config:
      offsets.topic.replication.factor: 1
      transaction.state.log.replication.factor: 1
      transaction.state.log.min.isr: 1
      log.message.format.version: '2.6'
    version: 2.6.0
    storage:
      type: jbod
      volumes:
      - id: 0
        type: persistent-claim
        size: 100Gi
        deleteClaim: false
    replicas: 1
    listeners:
      - name: plain
        port: 9092
        type: internal
        tls: false
      - name: tls
        port: 9093
        type: internal
        tls: true
      - name: external
        port: 9094
        type: nodeport
        tls: false

  entityOperator:
    topicOperator: {}
    userOperator: {}
  zookeeper:
    storage:
      type: persistent-claim
      size: 100Gi
      deleteClaim: false
    replicas: 1
```

```
apiVersion: kafka.strimzi.io/v1beta1
kind: Kafka
metadata:
  name: my-cluster
  namespace: mykafka
spec:
  kafka:
    config:
      offsets.topic.replication.factor: 1
      transaction.state.log.replication.factor: 1
      transaction.state.log.min.isr: 1
      log.message.format.version: '2.6'
    version: 2.6.0
    storage:
      type: ephemeral
    replicas: 1
    listeners:
      - name: plain
        port: 9092
        type: internal
        tls: false
      - name: tls
        port: 9093
        type: internal
        tls: true
      - name: external
        port: 9094
        type: nodeport
        tls: false

  entityOperator:
    topicOperator: {}
    userOperator: {}
  zookeeper:
    storage:
      type: ephemeral
    replicas: 1
```

создадим один дефолтный топик



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: mykafka

Installed Operators > Operator Details

Strimzi
0.20.0 provided by Strimzi

KafkaTopics

Name	Kind	Status	Labels	Last Updated
KT consumer-offsets---84e7a678d08f4bd226872e5cdd4eb527fadc1c6a	KafkaTopic	Condition: ✓ Ready	strimzi.io/cluster=my-cluster	a minute ago
KT my-topic	KafkaTopic	Condition: ✓ Ready	strimzi.io/cluster=my-cluster	6 minutes ago

проверим что кафка работате изнутри

https://strimzi.io/quickstarts/

```
oc -n mykafka run kafka-producer -ti --image=strimzi/kafka:0.20.0-kafka-2.6.0 --rm=true --restart=Never -- bin/kafka-console-producer.sh --broker-list my-cluster-kafka-bootstrap:9092 --topic my-topic
oc -n mykafka run kafka-consumer -ti --image=strimzi/kafka:0.20.0-kafka-2.6.0 --rm=true --restart=Never -- bin/kafka-console-consumer.sh --bootstrap-server my-cluster-kafka-bootstrap:9092 --topic my-topic --
from-beginning
```

проверим что кафка работате снаружи

https://strimzi.io/docs/operators/master/quickstart.html

http://kafka.apache.org/

последний вариант

https://medium.com/@karansingh010/kafka-on-openshift-with-external-routes-4d328058667c
https://github.com/strimzi/strimzi-kafka-operator/issues/128

```
apiVersion: kafka.strimzi.io/v1beta1
kind: Kafka
metadata:
  name: my-cluster
  namespace: mykafka
spec:
  kafka:
    config:
      offsets.topic.replication.factor: 1
      transaction.state.log.replication.factor: 1
      transaction.state.log.min.isr: 1
      log.message.format.version: '2.6'
    version: 2.6.0
    storage:
      type: ephemeral
    replicas: 1
    listeners:
      - name: plain
        port: 9092
        type: internal
        tls: false
      - name: tls
        port: 9093
        type: internal
        tls: true
      - name: external1
        port: 9094
        type: route
        tls: true
        authentication:
          type: tls
    readinessProbe:
      initialDelaySeconds: 15
      timeoutSeconds: 5
    livenessProbe:
      initialDelaySeconds: 15
      timeoutSeconds: 5
  entityOperator:
    topicOperator: {}
    userOperator: {}
  zookeeper:
    storage:
      type: ephemeral
    replicas: 1
    readinessProbe:
      initialDelaySeconds: 15
      timeoutSeconds: 5
    livenessProbe:
      initialDelaySeconds: 15
      timeoutSeconds: 5
```