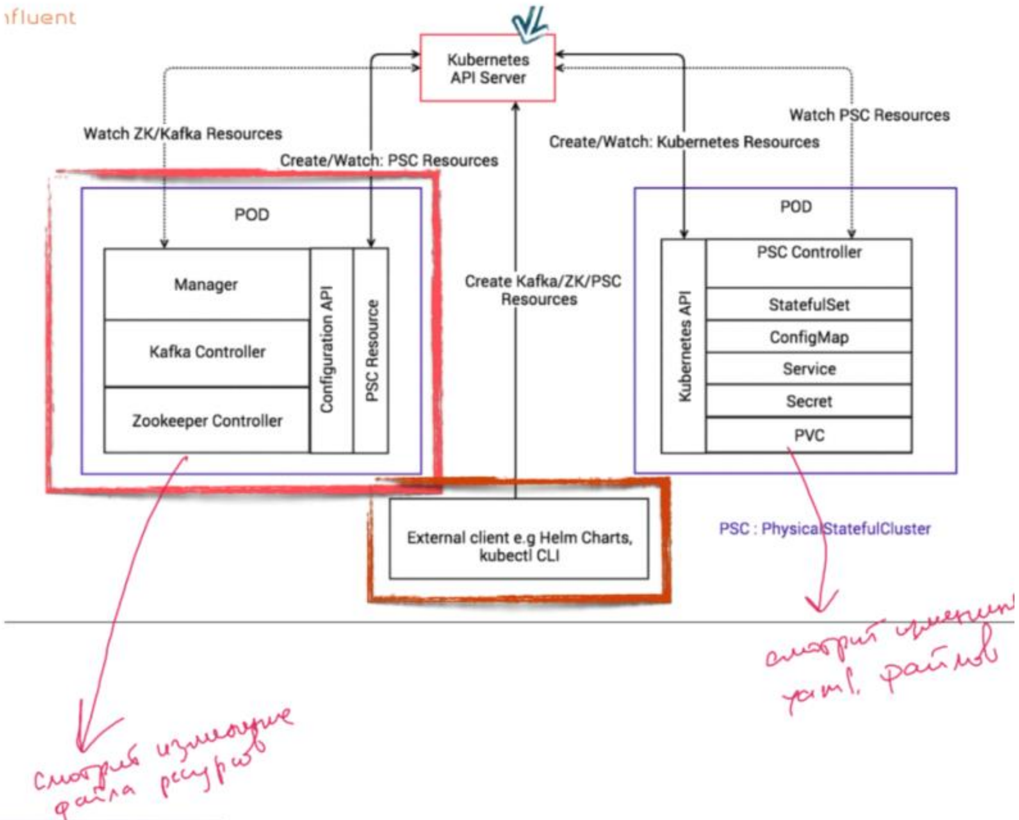
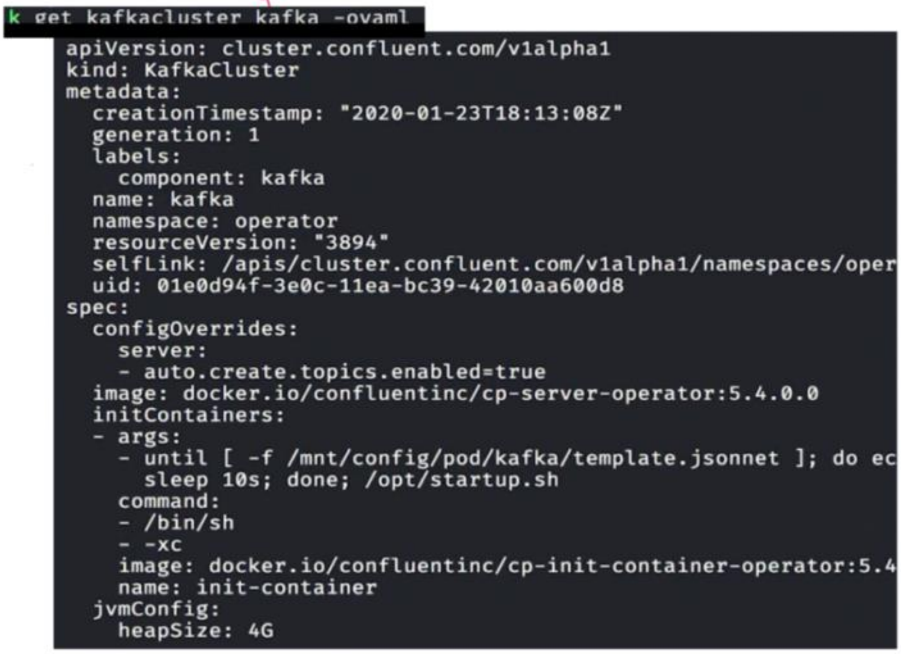
____kafka operator (Confluent operator)

24 октября 2020 г.   23:12

???

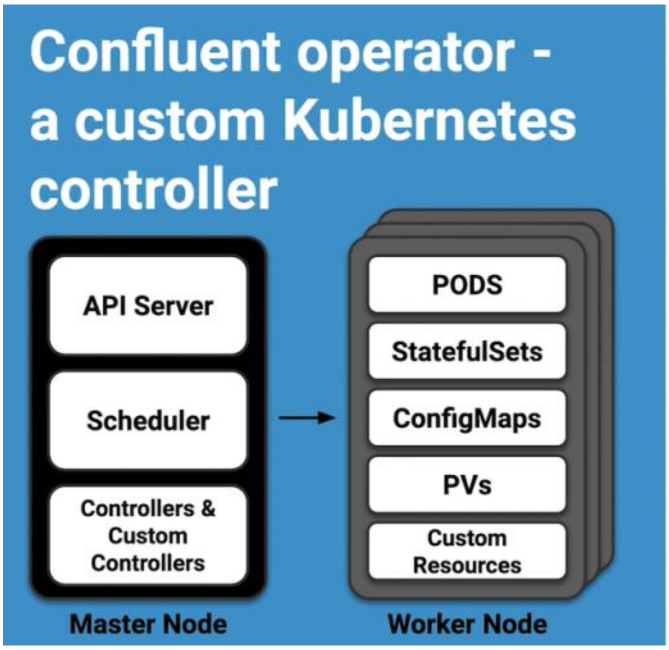
https://docs.confluent.io/current/installation/operator/co-introduction.html


nfluent
Kubernetes API Server
Watch ZK/Kafka Resources
Create/Watch: PSC Resources
Create/Watch: Kubernetes Resources
Watch PSC Resources
POD
Manager
Kafka Controller
Zookeeper Controller
Configuration API
PSC Resource
Create Kafka/ZK/PSC Resources
POD
PSC Controller
StatefulSet
ConfigMap
Service
Secret
PVC
Kubernetes API
External client e.g Helm Charts, kubectl CLI
PSC : PhysicalStatefulCluster
смотрим изменения yaml. файлов
Смотрим изменение файла ресурсов



```
k get kafkacluster kafka -oyaml
apiVersion: cluster.confluent.com/v1alpha1
kind: KafkaCluster
metadata:
  creationTimestamp: "2020-01-23T18:13:08Z"
  generation: 1
  labels:
    component: kafka
  name: kafka
  namespace: operator
  resourceVersion: "3894"
  selfLink: /apis/cluster.confluent.com/v1alpha1/namespaces/oper
  uid: 01e0d94f-3e0c-11ea-bc39-42010aa600d8
spec:
  configOverrides:
    server:
    - auto.create.topics.enabled=true
  image: docker.io/confluentinc/cp-server-operator:5.4.0.0
  initContainers:
  - args:
    - until [ -f /mnt/config/pod/kafka/template.jsonnet ]; do ec
      sleep 10s; done; /opt/startup.sh
    command:
    - /bin/sh
    - -xc
    image: docker.io/confluentinc/cp-init-container-operator:5.4
    name: init-container
  jvmConfig:
    heapSize: 4G
```

как устроен kafka-оператор (это всего лишь два дополнительных пода)



- Nodes and pods are where Applications run on Kubernetes
- Applications use objects like StatefulSets, Configmaps, PVs
- Custom Controllers create custom resources that provide unique application functionality:
  - Upgrades, elasticity, Kafka Operational Logic

Confluent Operator - Scale Horizontally

Automate Scaling:

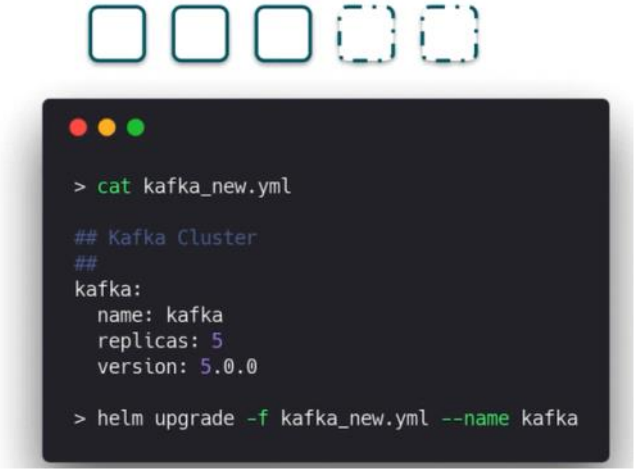
Spin up new brokers, connect workers easily

Manual Rebalance required for Operator v1.0:

Determine balancing plan

Execute balancing plan

Monitor Resources



```
> cat kafka_new.yml

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 5
  version: 5.0.0

> helm upgrade -f kafka_new.yml --name kafka
```

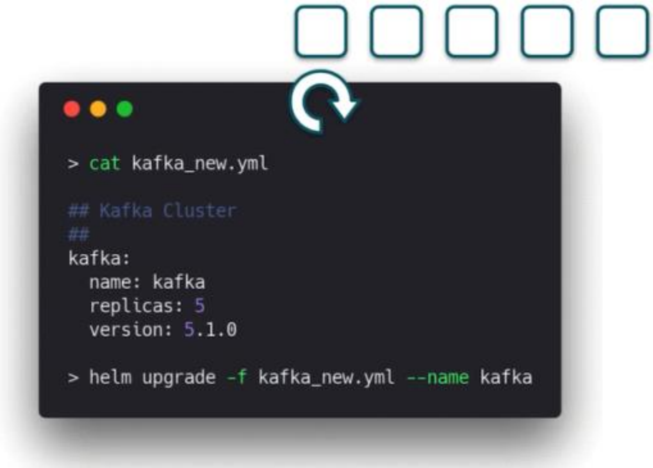
как обновить кафку (например обновить конфиг кафки)

Confluent Operator - Rolling Upgrade of all components

Automated Rolling Upgrades of all components - Kafka Brokers, Zookeeper, Connect, Control Center

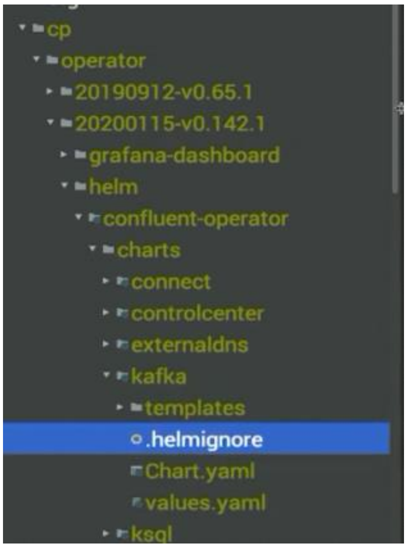
Kafka Broker Upgrades:

1. Stop the broker, upgrade Kafka
2. Wait for Partition Leader reassignment
3. Start the upgraded broker
4. Wait for zero under-replicated partitions
5. Upgrade the next broker

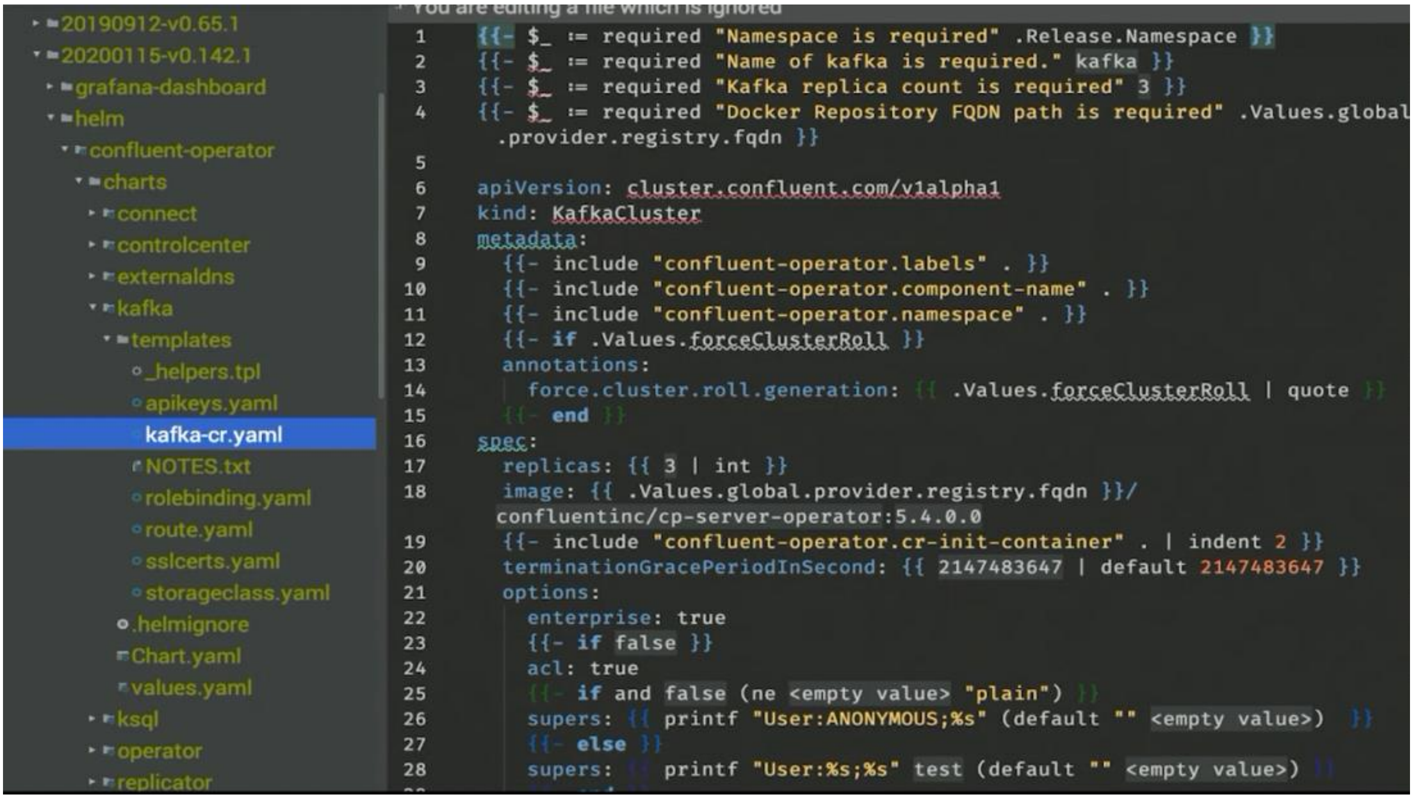


```
> cat kafka_new.yml

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 5
  version: 5.1.0

> helm upgrade -f kafka_new.yml --name kafka
```

-cr custom resource



```
▾ cp
  ▾ operator
    ▸ 20190912-v0.65.1
    ▾ 20200115-v0.142.1
      ▸ grafana-dashboard
      ▾ helm
        ▾ confluent-operator
          ▾ charts
            ▸ connect
            ▸ controlcenter
            ▸ externaldns
            ▾ kafka
              ▸ templates
                .helmignore
                Chart.yaml
                values.yaml
            ▸ ksql
```

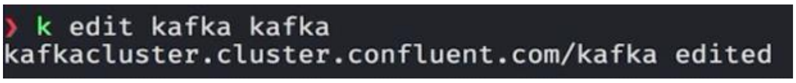


```
▸ 20190912-v0.65.1
▾ 20200115-v0.142.1
  ▸ grafana-dashboard
  ▾ helm
    ▾ confluent-operator
      ▾ charts
        ▸ connect
        ▸ controlcenter
        ▸ externaldns
        ▾ kafka
          ▾ templates
              _helpers.tpl
              apikeys.yaml
              kafka-cr.yaml
              NOTES.txt
              rolebinding.yaml
              route.yaml
              sslcerts.yaml
              storageclass.yaml
            .helmignore
            Chart.yaml
            values.yaml
        ▸ ksql
        ▸ operator
        ▸ replicator
```

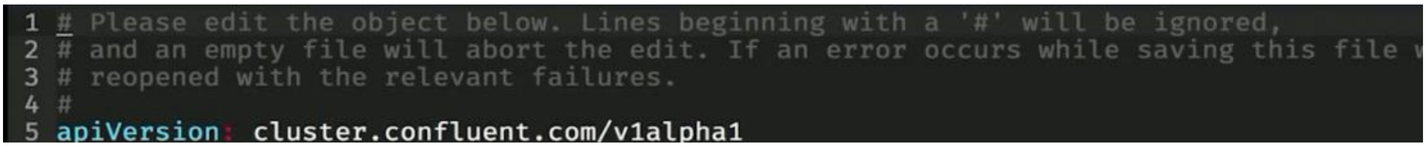
You are editing a file which is ignored

```
{{- $_ := required "Namespace is required" .Release.Namespace }}
{{- $_ := required "Name of kafka is required." kafka }}
{{- $_ := required "Kafka replica count is required" 3 }}
{{- $_ := required "Docker Repository FQDN path is required" .Values.global
 .provider.registry.fqdn }}

apiVersion: cluster.confluent.com/v1alpha1
kind: KafkaCluster
metadata:
  {{- include "confluent-operator.labels" . }}
  {{- include "confluent-operator.component-name" . }}
  {{- include "confluent-operator.namespace" . }}
  {{- if .Values.forceClusterRoll }}
  annotations:
    force.cluster.roll.generation: {{ .Values.forceClusterRoll | quote }}
  {{- end }}
spec:
  replicas: {{ 3 | int }}
  image: {{ .Values.global.provider.registry.fqdn }}/
 confluentinc/cp-server-operator:5.4.0.0
  {{- include "confluent-operator.cr-init-container" . | indent 2 }}
  terminationGracePeriodInSecond: {{ 2147483647 | default 2147483647 }}
  options:
    enterprise: true
    {{- if false }}
    acl: true
    {{- if and false (ne <empty value> "plain") }}
    supers: {{ printf "User:ANONYMOUS;%s" (default "" <empty value>)  }}
    {{- else }}
    supers: {{ printf "User:%s;%s" test (default "" <empty value>) }}
```

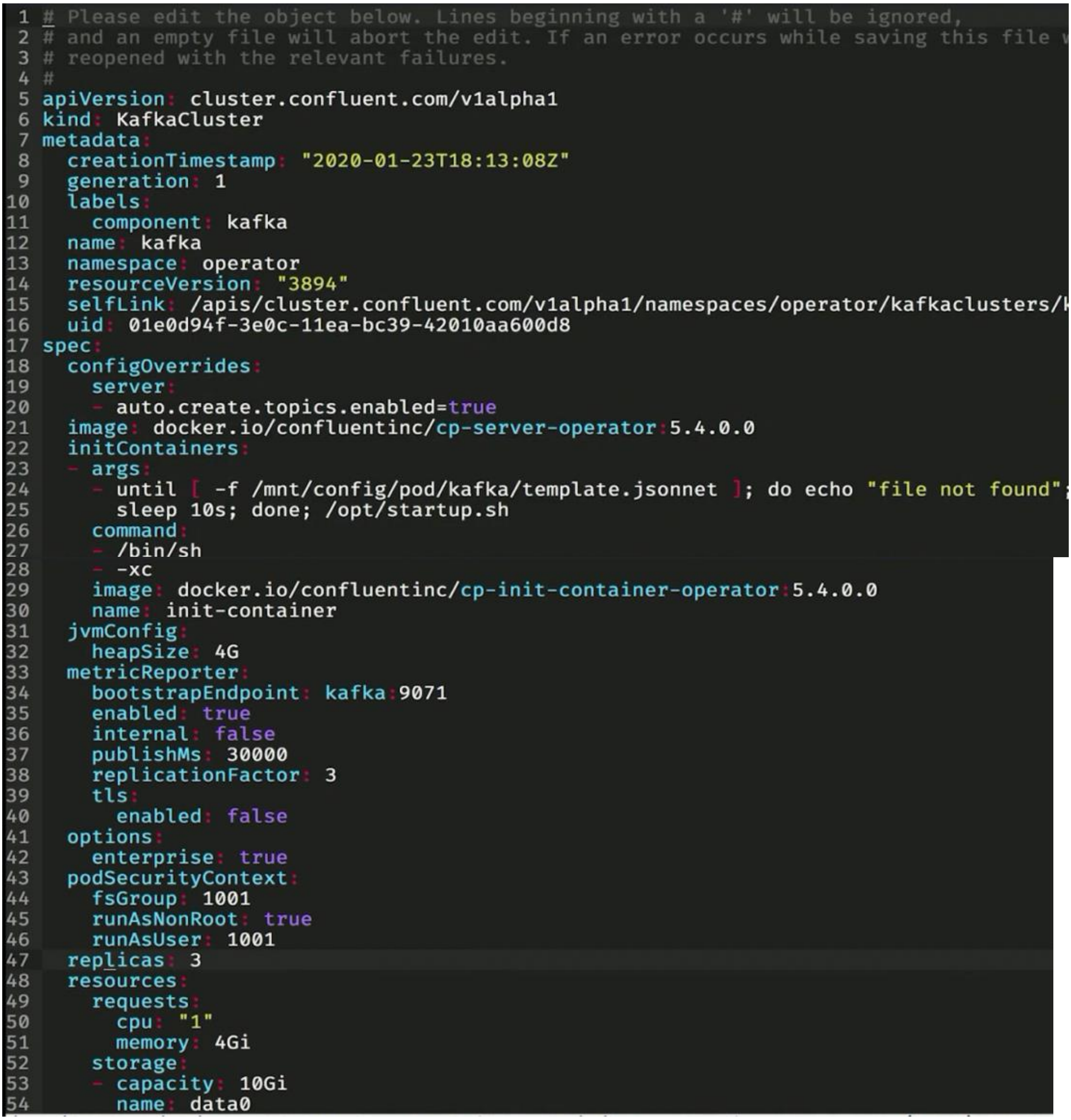
как отредактировать YAML файл оператора уже на бою (и изменения автоматически подхватятся оператором)



```
❯ k edit kafka kafka
kafkacluster.cluster.confluent.com/kafka edited
```



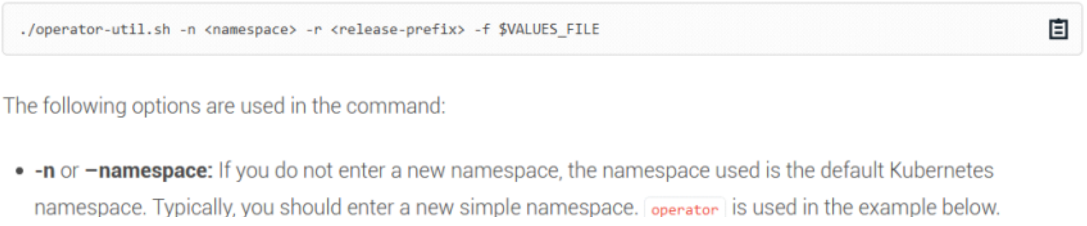
```
1 # Please edit the object below. Lines beginning with a '#' will be ignored,
2 # and an empty file will abort the edit. If an error occurs while saving this file w
3 # reopened with the relevant failures.
4 #
5 apiVersion: cluster.confluent.com/v1alpha1
```



```yaml
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file w
# reopened with the relevant failures.
#
apiVersion: cluster.confluent.com/v1alpha1
kind: KafkaCluster
metadata:
  creationTimestamp: "2020-01-23T18:13:08Z"
  generation: 1
  labels:
    component: kafka
  name: kafka
  namespace: operator
  resourceVersion: "3894"
  selfLink: /apis/cluster.confluent.com/v1alpha1/namespaces/operator/kafkaclusters/k
  uid: 01e0d94f-3e0c-11ea-bc39-42010aa600d8
spec:
  configOverrides:
    server:
    - auto.create.topics.enabled=true
  image: docker.io/confluentinc/cp-server-operator:5.4.0.0
  initContainers:
  - args:
    - until [ -f /mnt/config/pod/kafka/template.jsonnet ]; do echo "file not found";
      sleep 10s; done; /opt/startup.sh
    command:
    - /bin/sh
    - -xc
    image: docker.io/confluentinc/cp-init-container-operator:5.4.0.0
    name: init-container
  jvmConfig:
    heapSize: 4G
  metricReporter:
    bootstrapEndpoint: kafka:9071
    enabled: true
    internal: false
    publishMs: 30000
    replicationFactor: 3
    tls:
      enabled: false
  options:
    enterprise: true
  podSecurityContext:
    fsGroup: 1001
    runAsNonRoot: true
    runAsUser: 1001
  replicas: 3
  resources:
    requests:
      cpu: "1"
      memory: 4Gi
    storage:
    - capacity: 10Gi
      name: data0
```

как установить 1

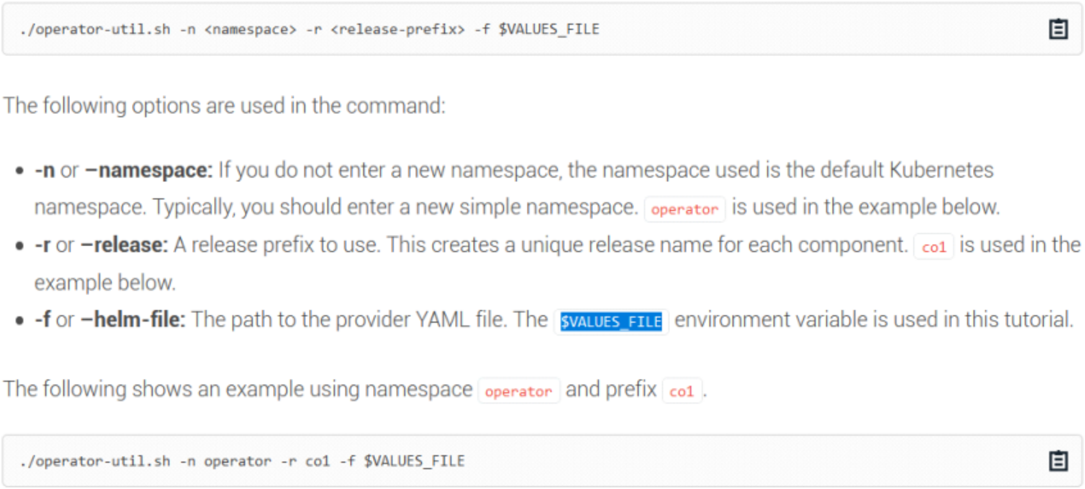
https://docs.confluent.io/current/installation/operator/co-quickstart.html
https://medium.com/@simon.aubury/confluent-kafka-kubernetes-operator-setup-and-playing-acd2c3c9fe5e



```
./operator-util.sh -n <namespace> -r <release-prefix> -f $VALUES_FILE
```

The following options are used in the command:

- **-n** or **–namespace:** If you do not enter a new namespace, the namespace used is the default Kubernetes namespace. Typically, you should enter a new simple namespace. `operator` is used in the example below.



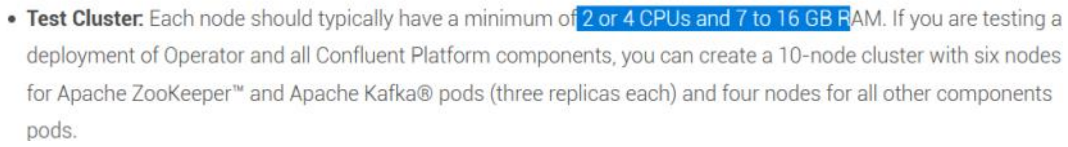
```
./operator-util.sh -n <namespace> -r <release-prefix> -f $VALUES_FILE
```

The following options are used in the command:

- **-n** or **–namespace:** If you do not enter a new namespace, the namespace used is the default Kubernetes namespace. Typically, you should enter a new simple namespace. `operator` is used in the example below.
- **-r** or **–release:** A release prefix to use. This creates a unique release name for each component. `co1` is used in the example below.
- **-f** or **–helm-file:** The path to the provider YAML file. The `$VALUES_FILE` environment variable is used in this tutorial.

The following shows an example using namespace `operator` and prefix `co1`.

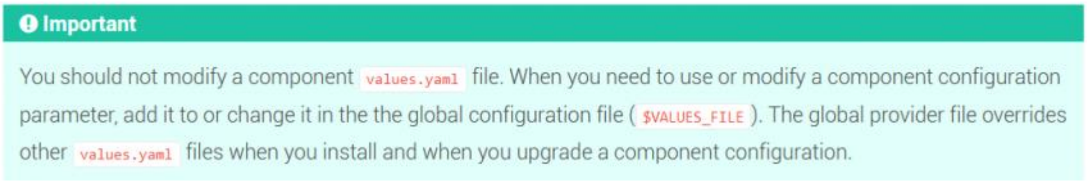
```
./operator-util.sh -n operator -r co1 -f $VALUES_FILE
```



- **Test Cluster:** Each node should typically have a minimum of 2 or 4 CPUs and 7 to 16 GB RAM. If you are testing a deployment of Operator and all Confluent Platform components, you can create a 10-node cluster with six nodes for Apache ZooKeeper™ and Apache Kafka® pods (three replicas each) and four nodes for all other components pods.

отдельные конфигурации

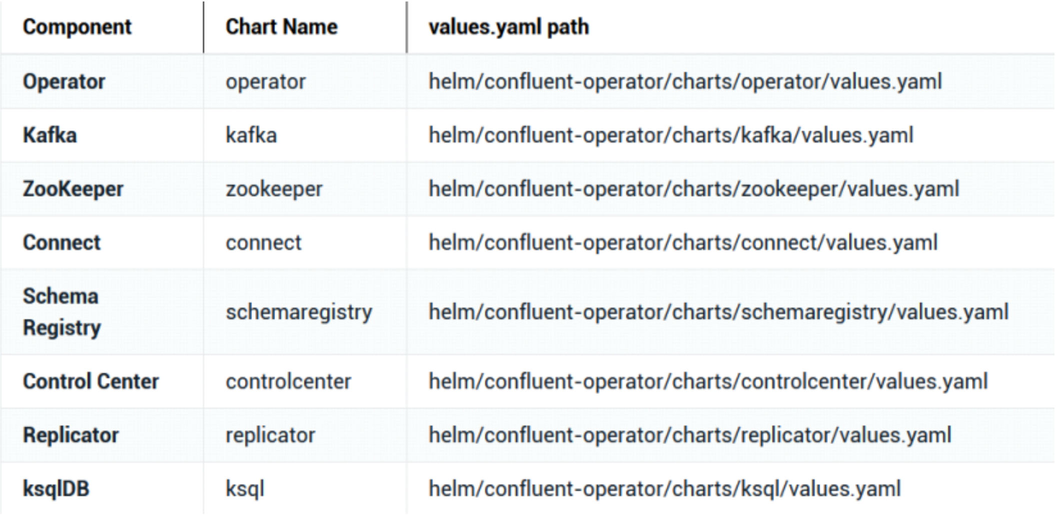
https://docs.confluent.io/current/installation/configuration/broker-configs.html#cp-config-brokers
https://docs.confluent.io/current/zookeeper/deployment.html#zk-prod-config



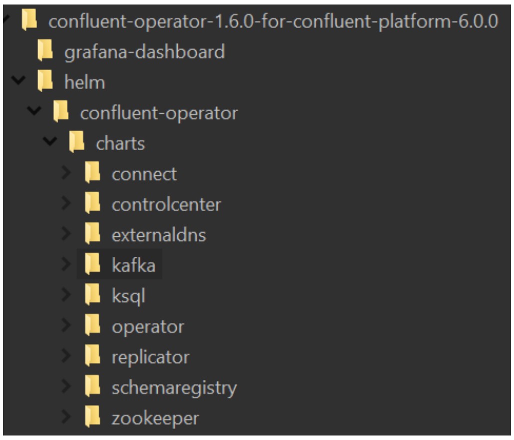
Important

You should not modify a component `values.yaml` file. When you need to use or modify a component configuration parameter, add it to or change it in the the global configuration file (`$VALUES_FILE`). The global provider file overrides other `values.yaml` files when you install and when you upgrade a component configuration.

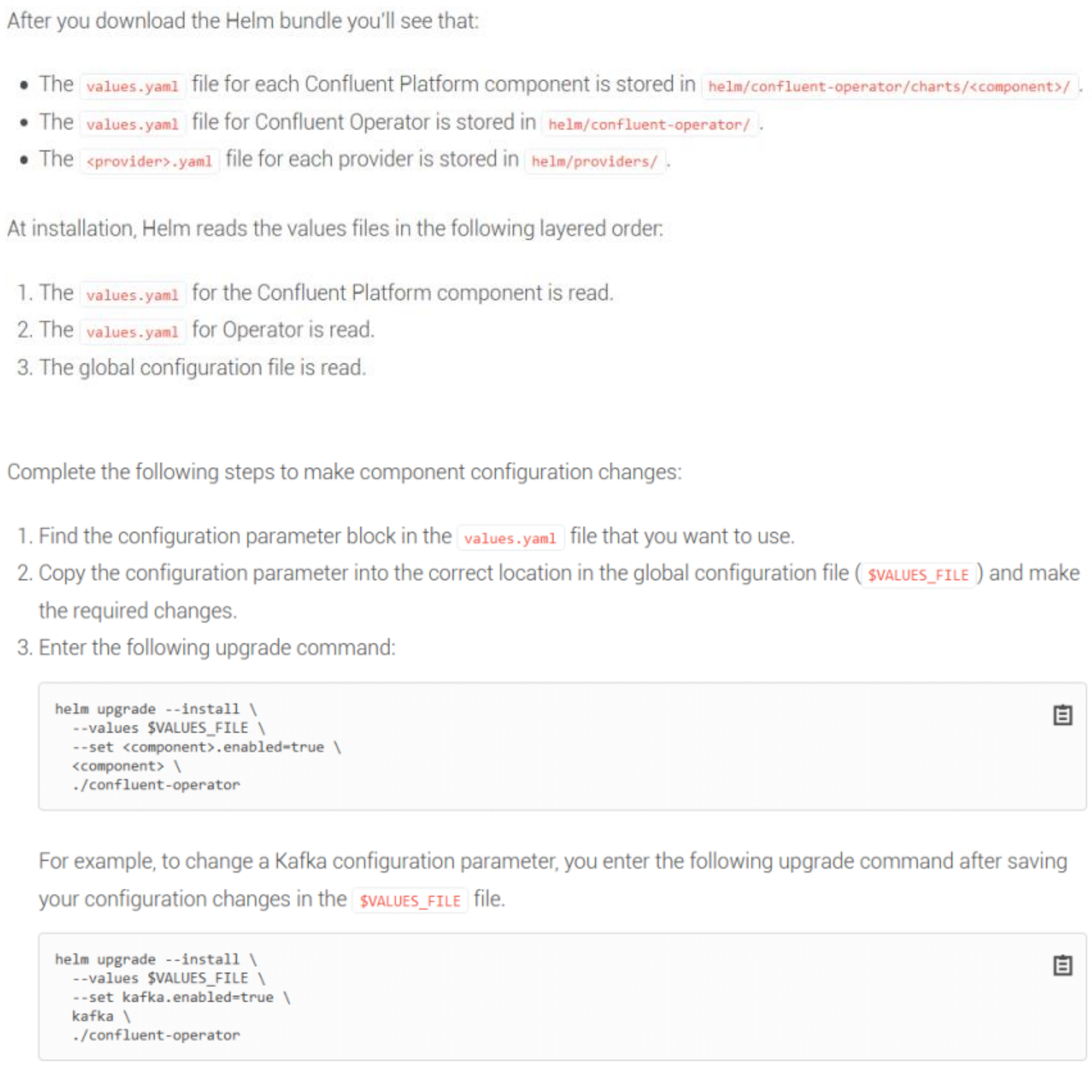
https://docs.confluent.io/current/installation/operator/co-configure.html



Component	Chart Name	values.yaml path
Operator	operator	helm/confluent-operator/charts/operator/values.yaml
Kafka	kafka	helm/confluent-operator/charts/kafka/values.yaml
ZooKeeper	zookeeper	helm/confluent-operator/charts/zookeeper/values.yaml
Connect	connect	helm/confluent-operator/charts/connect/values.yaml
Schema Registry	schemaregistry	helm/confluent-operator/charts/schemaregistry/values.yaml
Control Center	controlcenter	helm/confluent-operator/charts/controlcenter/values.yaml
Replicator	replicator	helm/confluent-operator/charts/replicator/values.yaml
ksqlDB	ksql	helm/confluent-operator/charts/ksql/values.yaml



```
confluent-operator-1.6.0-for-confluent-platform-6.0.0
  grafana-dashboard
  helm
    confluent-operator
      charts
        connect
        controlcenter
        externaldns
        kafka
        ksql
        operator
        replicator
        schemaregistry
        zookeeper
```



After you download the Helm bundle you'll see that:

- The `values.yaml` file for each Confluent Platform component is stored in `helm/confluent-operator/charts/<component>/`.
- The `values.yaml` file for Confluent Operator is stored in `helm/confluent-operator/`.
- The `<provider>.yaml` file for each provider is stored in `helm/providers/`.

At installation, Helm reads the values files in the following layered order:

1. The `values.yaml` for the Confluent Platform component is read.
2. The `values.yaml` for Operator is read.
3. The global configuration file is read.

Complete the following steps to make component configuration changes:

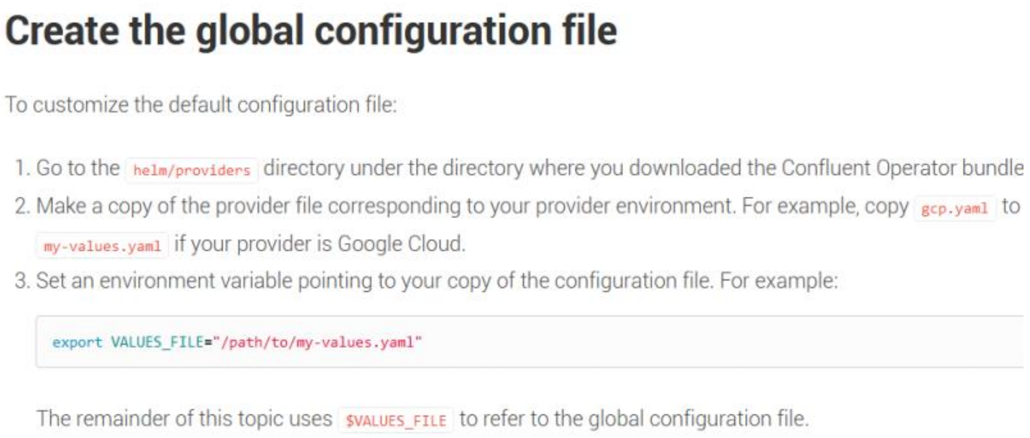
1. Find the configuration parameter block in the `values.yaml` file that you want to use.
2. Copy the configuration parameter into the correct location in the global configuration file (`$VALUES_FILE`) and make the required changes.
3. Enter the following upgrade command:

```
helm upgrade --install \
  --values $VALUES_FILE \
  --set <component>.enabled=true \
  <component> \
  ./confluent-operator
```

For example, to change a Kafka configuration parameter, you enter the following upgrade command after saving your configuration changes in the `$VALUES_FILE` file.

```
helm upgrade --install \
  --values $VALUES_FILE \
  --set kafka.enabled=true \
  kafka \
  ./confluent-operator
```

глобальная конфигурация



Create the global configuration file

To customize the default configuration file:

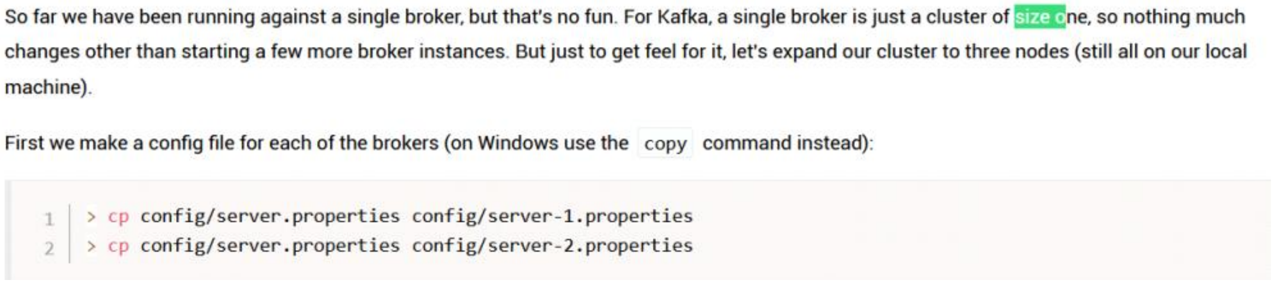
1. Go to the `helm/providers` directory under the directory where you downloaded the Confluent Operator bundle
2. Make a copy of the provider file corresponding to your provider environment. For example, copy `gcp.yaml` to `my-values.yaml` if your provider is Google Cloud.
3. Set an environment variable pointing to your copy of the configuration file. For example:

```
export VALUES_FILE="/path/to/my-values.yaml"
```

The remainder of this topic uses `$VALUES_FILE` to refer to the global configuration file.

kafka single broker configuration (можно без лицензии)

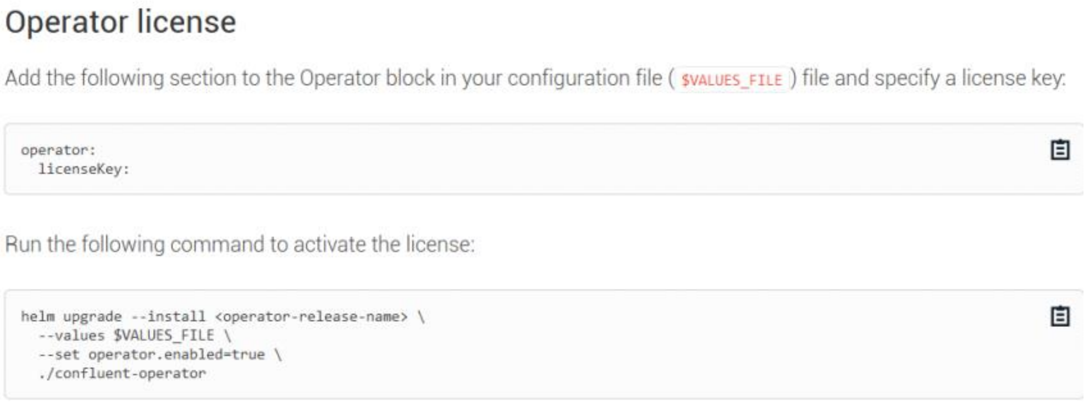
https://docs.confluent.io/current/installation/operator/co-configure.html



So far we have been running against a single broker, but that's no fun. For Kafka, a single broker is just a cluster of size one, so nothing much changes other than starting a few more broker instances. But just to get feel for it, let's expand our cluster to three nodes (still all on our local machine).

First we make a config file for each of the brokers (on Windows use the `copy` command instead):

```
> cp config/server.properties config/server-1.properties
> cp config/server.properties config/server-2.properties
```



Operator license

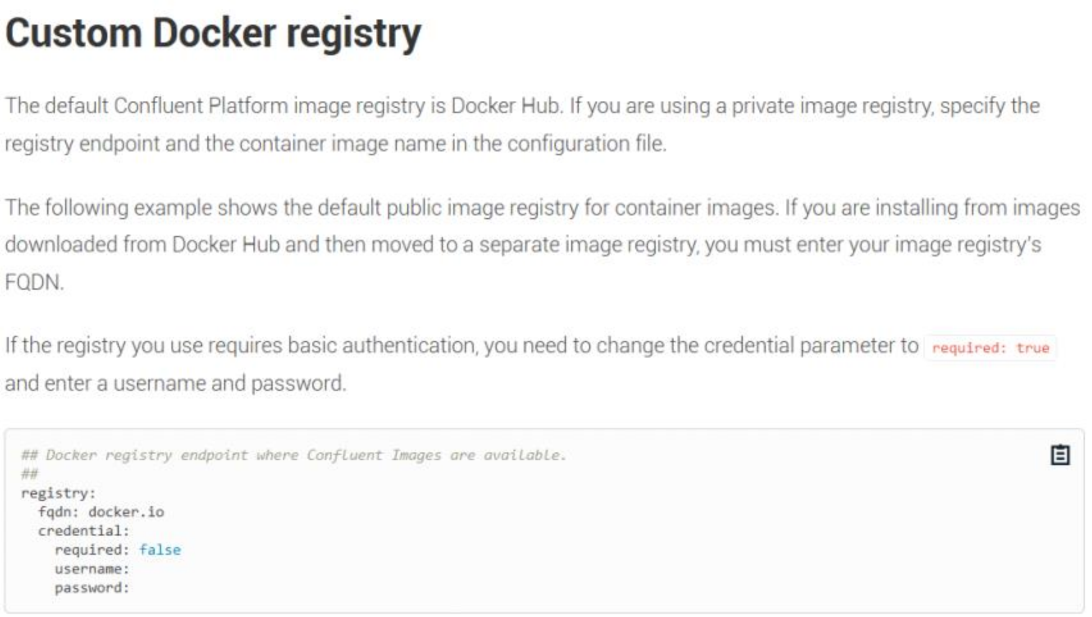
Add the following section to the Operator block in your configuration file (`$VALUES_FILE`) file and specify a license key:

```
operator:
  licenseKey:
```

Run the following command to activate the license:

```
helm upgrade --install <operator-release-name> \
  --values $VALUES_FILE \
  --set operator.enabled=true \
  ./confluent-operator
```

настроить на свой docker registry



Custom Docker registry

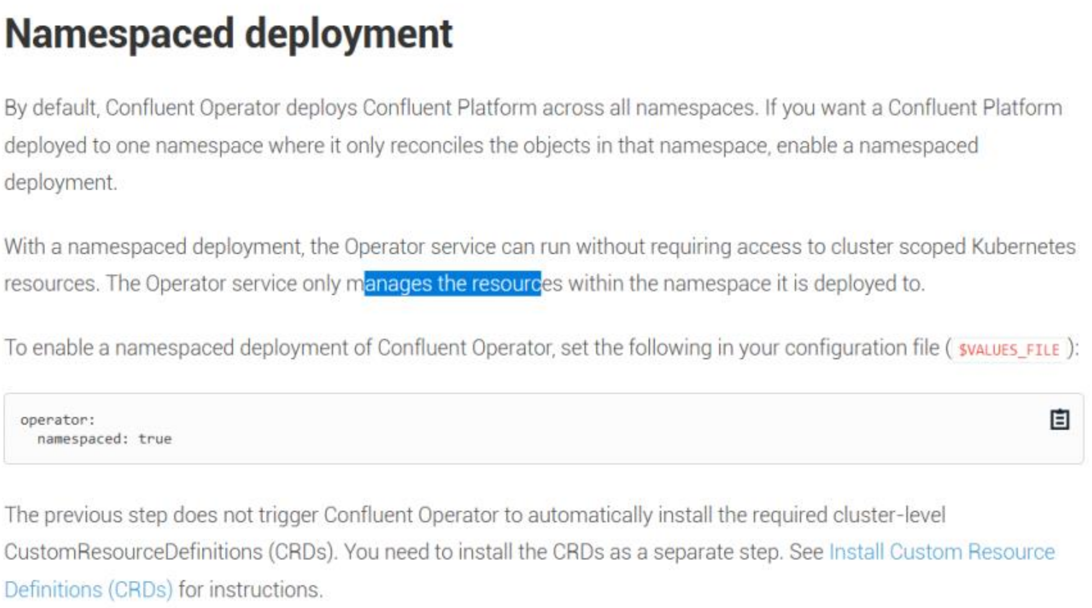
The default Confluent Platform image registry is Docker Hub. If you are using a private image registry, specify the registry endpoint and the container image name in the configuration file.

The following example shows the default public image registry for container images. If you are installing from images downloaded from Docker Hub and then moved to a separate image registry, you must enter your image registry's FQDN.

If the registry you use requires basic authentication, you need to change the credential parameter to `required: true` and enter a username and password.

```
## Docker registry endpoint where Confluent Images are available.
##
registry:
  fqdn: docker.io
  credential:
    required: false
    username:
    password:
```

? настроить отдельный неймспейс



Namespaced deployment

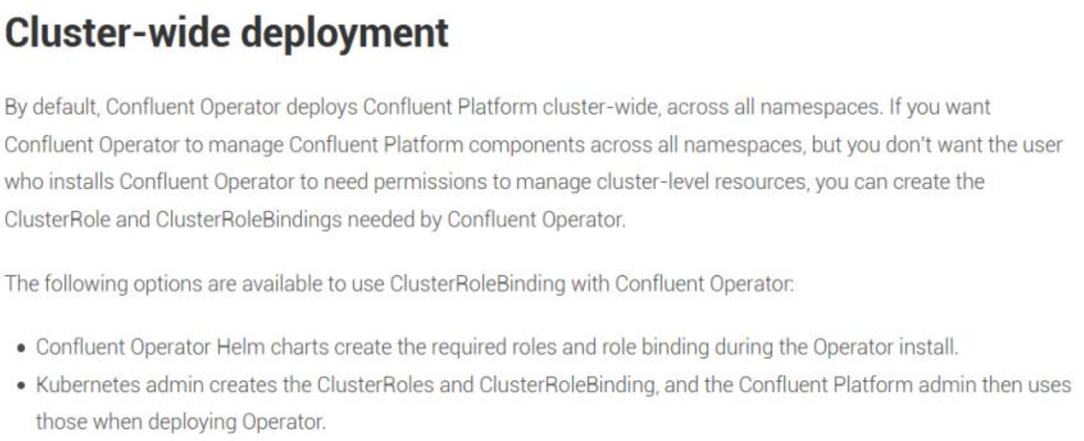
By default, Confluent Operator deploys Confluent Platform across all namespaces. If you want a Confluent Platform deployed to one namespace where it only reconciles the objects in that namespace, enable a namespaced deployment.

With a namespaced deployment, the Operator service can run without requiring access to cluster scoped Kubernetes resources. The Operator service only manages the resources within the namespace it is deployed to.

To enable a namespaced deployment of Confluent Operator, set the following in your configuration file (`$VALUES_FILE`):

```
operator:
  namespaced: true
```

The previous step does not trigger Confluent Operator to automatically install the required cluster-level CustomResourceDefinitions (CRDs). You need to install the CRDs as a separate step. See Install Custom Resource Definitions (CRDs) for instructions.



Cluster-wide deployment

By default, Confluent Operator deploys Confluent Platform cluster-wide, across all namespaces. If you want Confluent Operator to manage Confluent Platform components across all namespaces, but you don't want the user who installs Confluent Operator to need permissions to manage cluster-level resources, you can create the ClusterRole and ClusterRoleBindings needed by Confluent Operator.

The following options are available to use ClusterRoleBinding with Confluent Operator:

- Confluent Operator Helm charts create the required roles and role binding during the Operator install.
- Kubernetes admin creates the ClusterRoles and ClusterRoleBinding, and the Confluent Platform admin then uses those when deploying Operator.

? node affinity

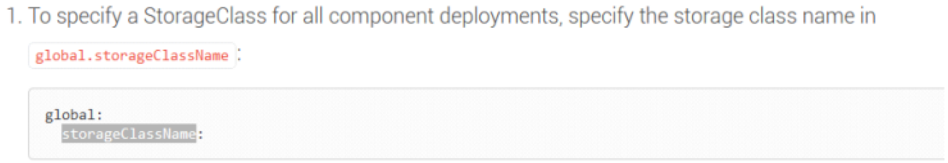
d

storage configuration: helm.providers.private_yaml

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

- By default, Operator manages storage using dynamic storage provisioning that Kubernetes provides
- If you must rely on statically provisioned storage volumes, you can manually provision and attach storage to your Kubernetes worker nodes, expose those to the platform as PersistentVolumes, and then use Confluent Operator to deploy Confluent Platform clusters so that the broker instances mount those PersistentVolumes.
- Confluent Operator does not support migration from one storage class to another.

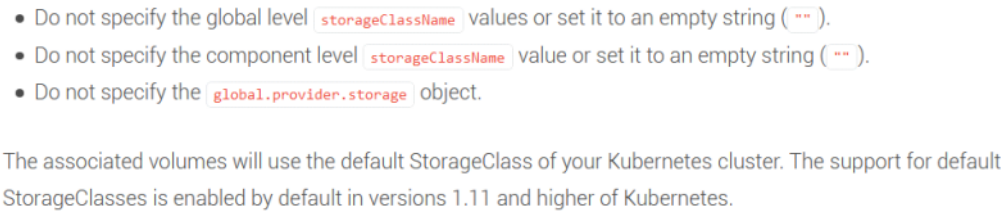
как все задеплоить на выбранный StorageClass



1. To specify a StorageClass for all component deployments, specify the storage class name in `global.storageClassName`:

```
global:
  storageClassName:
```

как все задеплоить по умолчанию на minikube



- Do not specify the global level `storageClassName` values or set it to an empty string (`""`).
- Do not specify the component level `storageClassName` value or set it to an empty string (`""`).
- Do not specify the `global.provider.storage` object.

The associated volumes will use the default StorageClass of your Kubernetes cluster. The support for default StorageClasses is enabled by default in versions 1.11 and higher of Kubernetes.

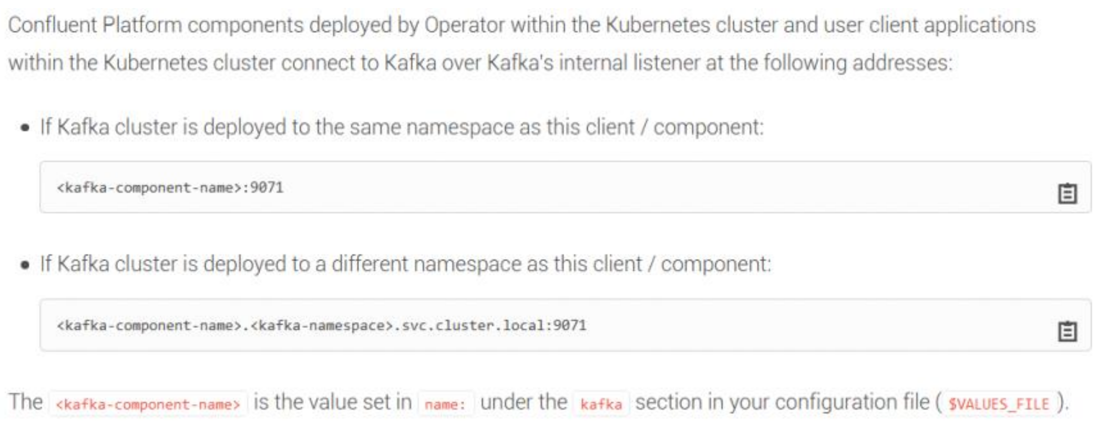
? Use the StorageClass created by Confluent Operator Helm charts

?

External access to Kafka using Ingress with port-based routing

- Kubernetes Ingress only supports HTTP-based services whereas Kafka is TCP-based. However, there are Ingress controller implementations in the ecosystem, such as the NGINX Ingress Controller, that support non-HTTP-based services like Kafka. You can use one of those Ingress controllers to enable external access to Kafka over HTTP

Internal access to Kafka



Confluent Platform components deployed by Operator within the Kubernetes cluster and user client applications within the Kubernetes cluster connect to Kafka over Kafka's internal listener at the following addresses:

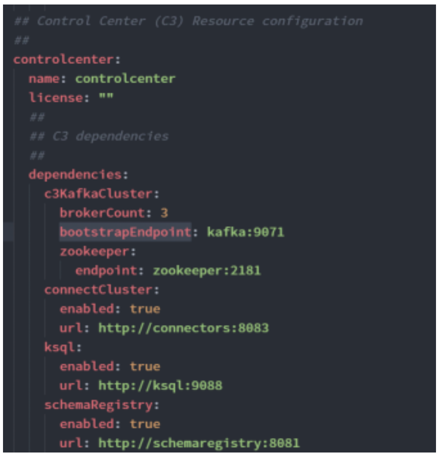
- If Kafka cluster is deployed to the same namespace as this client / component:

```
<kafka-component-name>:9071
```

- If Kafka cluster is deployed to a different namespace as this client / component:

```
<kafka-component-name>.<kafka-namespace>.svc.cluster.local:9071
```

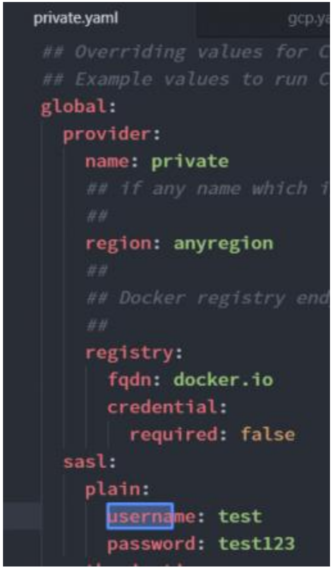
The `<kafka-component-name>` is the value set in `name:` under the `kafka` section in your configuration file (`$VALUES_FILE`).



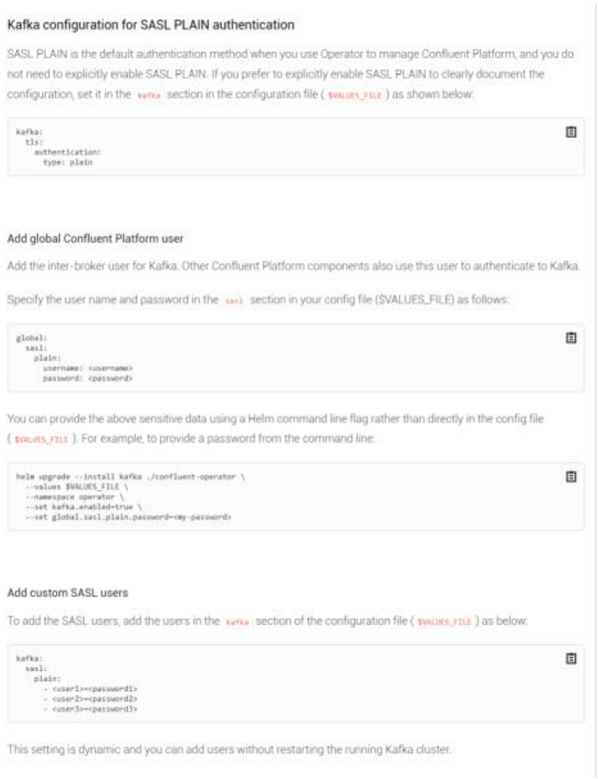
```
## Control Center (C3) Resource configuration
##
controlcenter:
  name: controlcenter
  license: ""
  ##
  ## C3 dependencies
  ##
  dependencies:
    c3KafkaCluster:
      brokerCount: 3
      bootstrapEndpoint: kafka:9071
      zookeeper:
        endpoint: zookeeper:2181
    connectCluster:
      enabled: true
      url: http://connectors:8083
    ksql:
      enabled: true
      url: http://ksql:9088
    schemaRegistry:
      enabled: true
      url: http://schemaregistry:8081
```

Authentication with SASL PLAIN

When PLAIN SASL authentication is configured, external clients and internal Confluent Platform components provide a username and password for both to authenticate with Kafka brokers



```
## Overriding values for C
## Example values to run C
global:
  provider:
    name: private
    ## if any name which i
    ##
    region: anyregion
    ##
    ## Docker registry end
    ##
    registry:
      fqdn: docker.io
      credential:
        required: false
  sasl:
    plain:
      username: test
      password: test123
```



Kafka configuration for SASL PLAIN authentication

SASL PLAIN is the default authentication method when you use Operator to manage Confluent Platform, and you do not need to explicitly enable SASL PLAIN. If you prefer to explicitly enable SASL PLAIN to clearly document the configuration, set it in the `kafka` section in the configuration file (`$VALUES_FILE`) as shown below:

```
kafka:
  tls:
    authentication:
      type: plain
```

Add global Confluent Platform user

Add the inter-broker user for Kafka. Other Confluent Platform components also use this user to authenticate to Kafka.

Specify the user name and password in the `sasl` section in your config file ($VALUES_FILE) as follows:

```
global:
  sasl:
    plain:
      username: <username>
      password: <password>
```

You can provide the above sensitive data using a Helm command line flag rather than directly in the config file (`$VALUES_FILE`). For example, to provide a password from the command line:

```
helm upgrade --install kafka ./confluent-operator \
  --values $VALUES_FILE \
  --namespace operator \
  --set kafka.enabled=true \
  --set global.sasl.plain.password=<my-password>
```

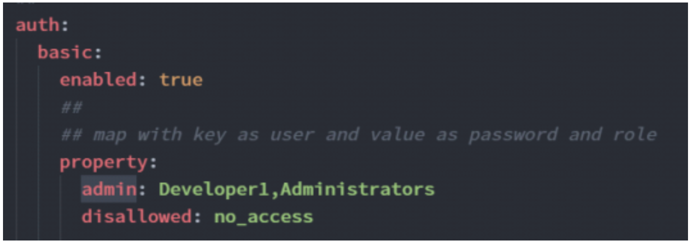
Add custom SASL users

To add the SASL users, add the users in the `kafka` section of the configuration file (`$VALUES_FILE`) as below:

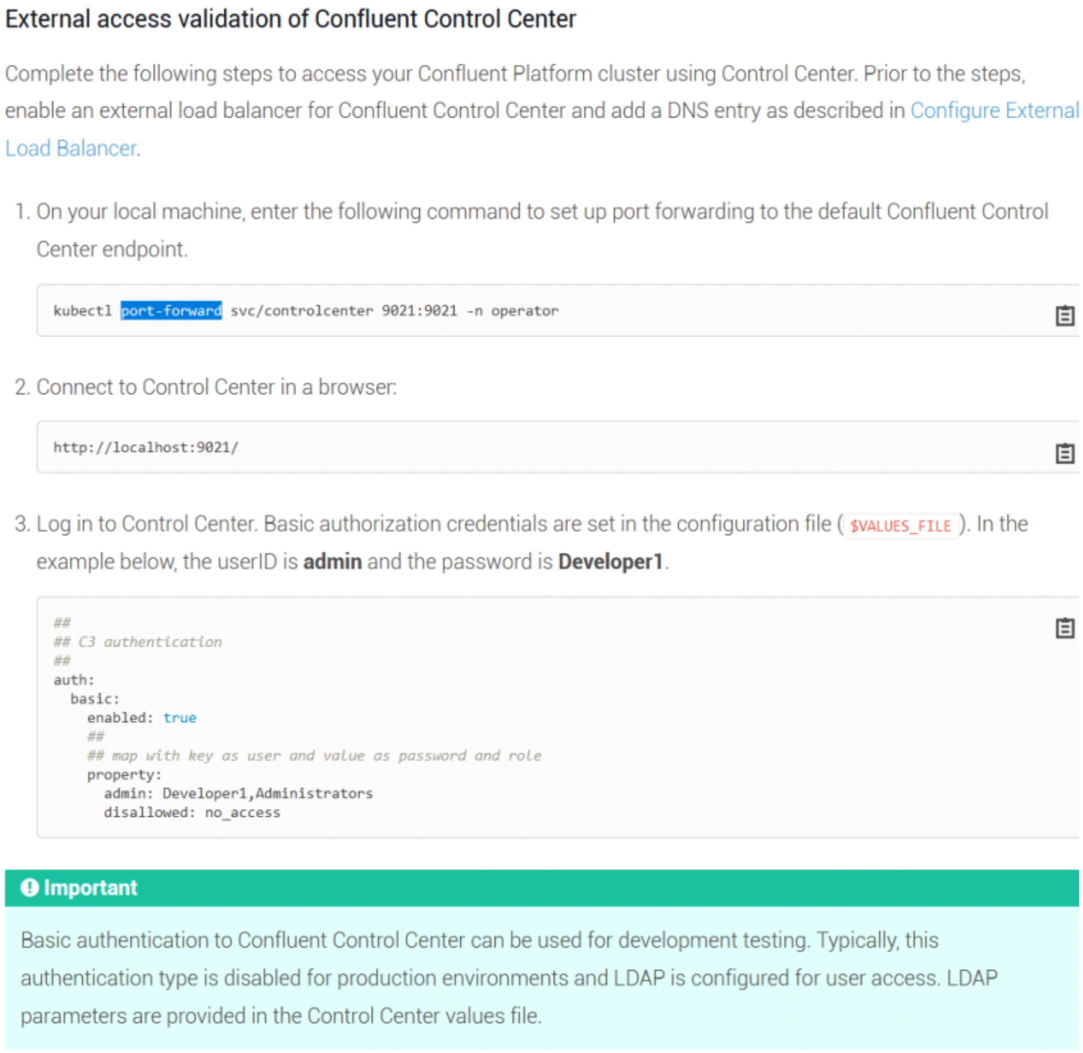
```
kafka:
  sasl:
    plain:
      - <user1>=<password1>
      - <user2>=<password2>
      - <user3>=<password3>
```

This setting is dynamic and you can add users without restarting the running Kafka cluster.

Confluent Control Center encryption and authentication



```
auth:
  basic:
    enabled: true
    ##
    ## map with key as user and value as password and role
    property:
      admin: Developer1,Administrators
      disallowed: no_access
```



External access validation of Confluent Control Center

Complete the following steps to access your Confluent Platform cluster using Control Center. Prior to the steps, enable an external load balancer for Confluent Control Center and add a DNS entry as described in Configure External Load Balancer.

1. On your local machine, enter the following command to set up port forwarding to the default Confluent Control Center endpoint.

```
kubectl port-forward svc/controlcenter 9021:9021 -n operator
```

2. Connect to Control Center in a browser:

```
http://localhost:9021/
```

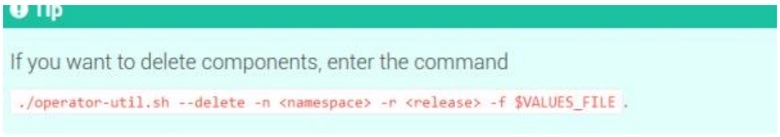
3. Log in to Control Center. Basic authorization credentials are set in the configuration file (`$VALUES_FILE`). In the example below, the userID is **admin** and the password is **Developer1**.

```
##
## C3 authentication
##
auth:
  basic:
    enabled: true
    ##
    ## map with key as user and value as password and role
    property:
      admin: Developer1,Administrators
      disallowed: no_access
```

Important

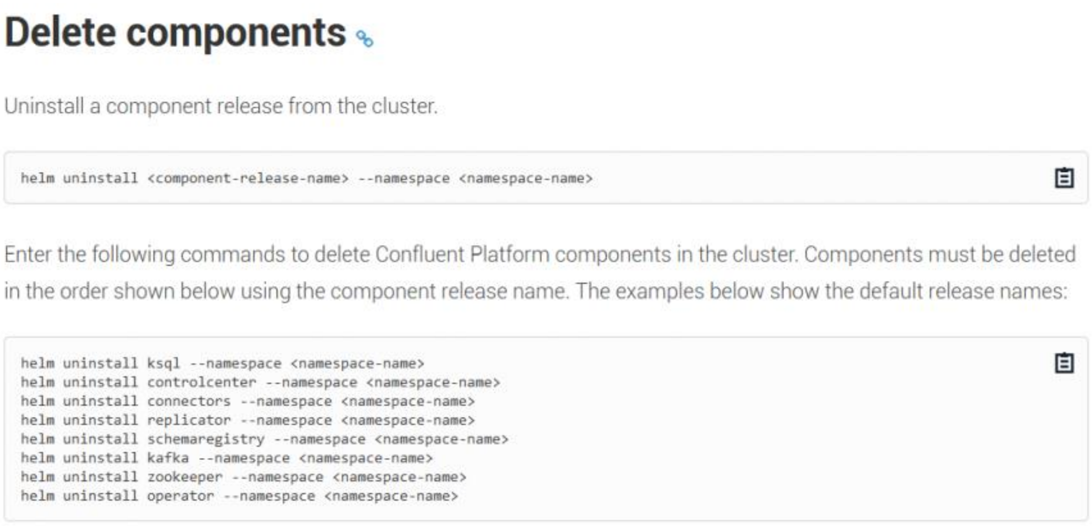
Basic authentication to Confluent Control Center can be used for development testing. Typically, this authentication type is disabled for production environments and LDAP is configured for user access. LDAP parameters are provided in the Control Center values file.

как все удалить



Tip

If you want to delete components, enter the command `./operator-util.sh --delete -n <namespace> -r <release> -f $VALUES_FILE` .



Delete components

Uninstall a component release from the cluster.

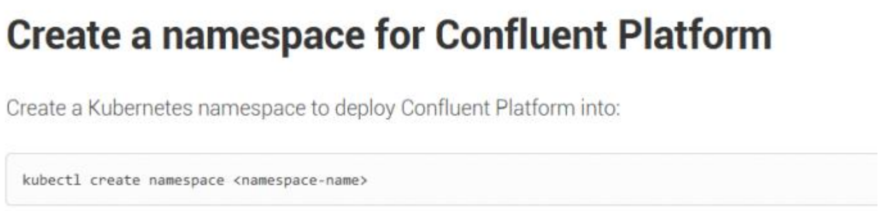
```
helm uninstall <component-release-name> --namespace <namespace-name>
```

Enter the following commands to delete Confluent Platform components in the cluster. Components must be deleted in the order shown below using the component release name. The examples below show the default release names:

```
helm uninstall ksql --namespace <namespace-name>
helm uninstall controlcenter --namespace <namespace-name>
helm uninstall connectors --namespace <namespace-name>
helm uninstall replicator --namespace <namespace-name>
helm uninstall schemaregistry --namespace <namespace-name>
helm uninstall kafka --namespace <namespace-name>
helm uninstall zookeeper --namespace <namespace-name>
helm uninstall operator --namespace <namespace-name>
```

как установить 2
https://dev.to/simon_sugob/kafka-on-kubernetes-confluent-has-made-it-3c20

(0) создать отдельный неймспейс



Create a namespace for Confluent Platform

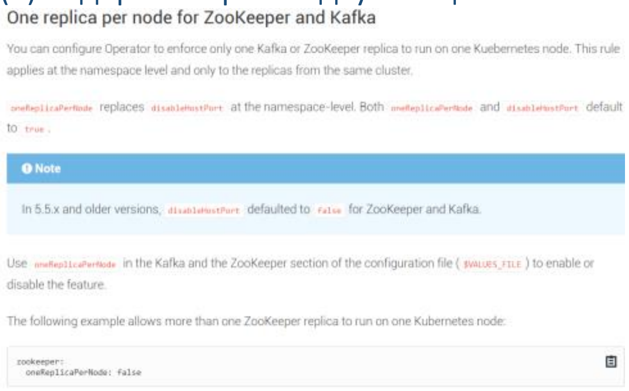
Create a Kubernetes namespace to deploy Confluent Platform into:

```
kubectl create namespace <namespace-name>
```

Custom Resource Defintions (CRDs).

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

(0) подправим файл в двух секциях



One replica per node for ZooKeeper and Kafka

You can configure Operator to enforce only one Kafka or ZooKeeper replica to run on one Kuebernetes node. This rule applies at the namespace level and only to the replicas from the same cluster.

`oneReplicaPerNode` replaces `disableHostPort` at the namespace-level. Both `oneReplicaPerNode` and `disableHostPort` default to `true`.

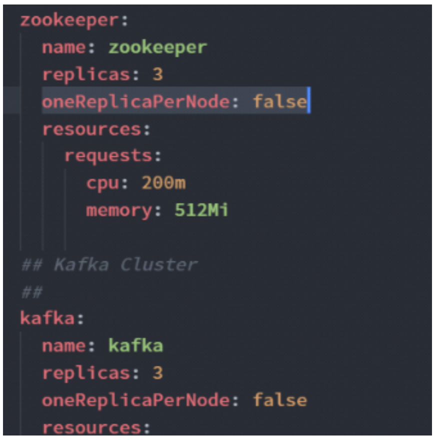
Note

In 5.5.x and older versions, `disableHostPort` defaulted to `false` for ZooKeeper and Kafka.

Use `oneReplicaPerNode` in the Kafka and the ZooKeeper section of the configuration file (`$VALUES_FILE`) to enable or disable the feature.

The following example allows more than one ZooKeeper replica to run on one Kubernetes node:

```
zookeeper:
  oneReplicaPerNode: false
```

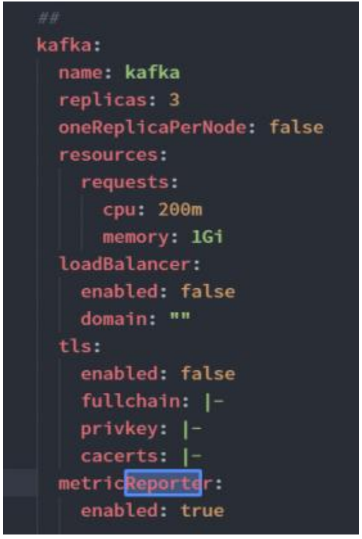


```
zookeeper:
  name: zookeeper
  replicas: 3
  oneReplicaPerNode: false
  resources:
    requests:
      cpu: 200m
      memory: 512Mi

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 3
  oneReplicaPerNode: false
  resources:
```

```
zookeeper:
  name: zookeeper
  replicas: 3
  oneReplicaPerNode: false
  resources:
    requests:
      cpu: 200m
      memory: 512Mi

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 3
  oneReplicaPerNode: false
```

(0a) добавим metric_reporter

https://docs.confluent.io/6.0.0/kafka/metrics-reporter.html



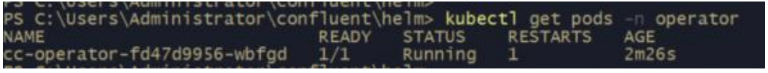
```
##
kafka:
  name: kafka
  replicas: 3
  oneReplicaPerNode: false
  resources:
    requests:
      cpu: 200m
      memory: 1Gi
  loadBalancer:
    enabled: false
    domain: ""
  tls:
    enabled: false
    fullchain: |-
    privkey: |-
    cacerts: |-
  metricReporter:
    enabled: true
```

(1) поставим оператор (в POWERSHELL)

```
kubectl create namespace operator
cd C:\Users\Administrator\confluent\helm
cd C:\Users\vovan\confluent-operator\helm
helm upgrade --install  operator .\confluent-operator  --values $env:VALUES_FILE --namespace operator --set operator.enabled=true
```

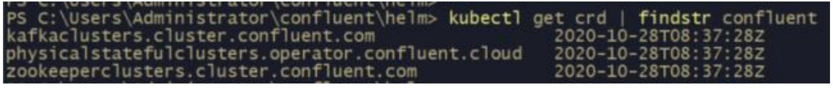
проверим что работает

kubectl get pods -n operator | findstr cc-operator



```
PS C:\Users\Administrator\confluent\helm> kubectl get pods -n operator
NAME                          READY   STATUS    RESTARTS   AGE
cc-operator-fd47d9956-wbfgd   1/1     Running   1          2m26s
```

kubectl get crd | findstr confluent



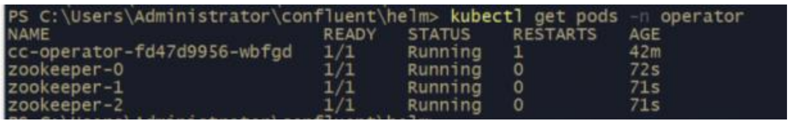
```
PS C:\Users\Administrator\confluent\helm> kubectl get crd | findstr confluent
kafkaclusters.cluster.confluent.com                   2020-10-28T08:37:28Z
physicalstatefulclusters.operator.confluent.cloud     2020-10-28T08:37:28Z
zookeeperclusters.cluster.confluent.com               2020-10-28T08:37:28Z
```

(2) поставим ZK (в POWERSHELL)

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

проверим что работает (все ноды должны быть RUNNING)

kubectl get pods -n operator



```
PS C:\Users\Administrator\confluent\helm> kubectl get pods -n operator
NAME                          READY   STATUS    RESTARTS   AGE
cc-operator-fd47d9956-wbfgd   1/1     Running   1          42m
zookeeper-0                   1/1     Running   0          72s
zookeeper-1                   1/1     Running   0          71s
zookeeper-2                   1/1     Running   0          71s
```

1. Validate if Zookeeper Custom Resource (CR) is created

kubectl get zookeeper -n operator | grep zookeeper

2. Check the status/events of CR: zookeeper

kubectl describe zookeeper zookeeper -n operator

3. Check if Zookeeper cluster is Ready

kubectl get zookeeper zookeeper -oyaml -n operator

kubectl get zookeeper zookeeper -ojsonpath='{.status.phase}' -n operator

4. Update/Upgrade Zookeeper Cluster

The upgrade can be done either through the helm upgrade or by editing the CR directly as below;

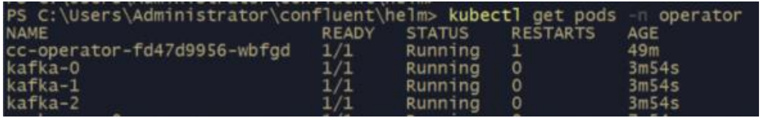
kubectl edit zookeeper zookeeper -n operator

(3) поставим кафку (в POWERSHELL)

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

проверим что работает

kubectl get pods -n operator
kubectl get kafka -n operator
kubectl get kafka kafka -n operator -oyaml
kubectl -n operator get kafka kafka -ojsonpath='{.status.replicationFactor}'



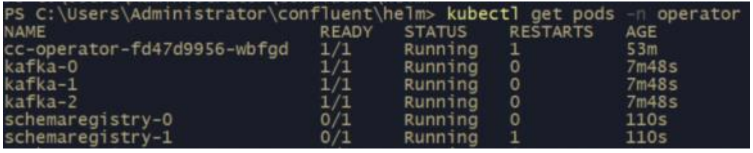
```
PS C:\Users\Administrator\confluent\helm> kubectl get pods -n operator
NAME                          READY   STATUS    RESTARTS   AGE
cc-operator-fd47d9956-wbfgd   1/1     Running   1          49m
kafka-0                       1/1     Running   0          3m54s
kafka-1                       1/1     Running   0          3m54s
kafka-2                       1/1     Running   0          3m54s
```

(4) поставим реестр схем (в POWERSHELL)

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

проверим ЧТО РАБОТАЕТ

kubectl get pods -n operator



```
PS C:\Users\Administrator\confluent\helm> kubectl get pods -n operator
NAME                          READY   STATUS    RESTARTS   AGE
cc-operator-fd47d9956-wbfgd   1/1     Running   1          53m
kafka-0                       1/1     Running   0          7m48s
kafka-1                       1/1     Running   0          7m48s
kafka-2                       1/1     Running   0          7m48s
schemaregistry-0              0/1     Running   0          110s
schemaregistry-1              0/1     Running   1          110s
```

2. Access
   Internal REST Endpoint : http://schemaregistry:8081 (Inside kubernetes)

   OR

   http://localhost:8081 (Inside Pod)

   More information about schema registry REST API can be found here,

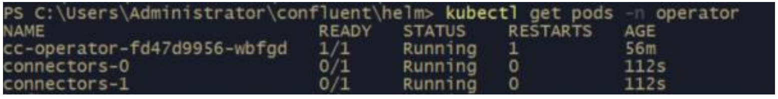
   https://docs.confluent.io/current/schema-registry/docs/api.html

(5) поставим кафка конект (в POWERSHELL)

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

приверим что работает

kubectl get pods -n operator



```
PS C:\Users\Administrator\confluent\helm> kubectl get pods -n operator
NAME                          READY   STATUS    RESTARTS   AGE
cc-operator-fd47d9956-wbfgd   1/1     Running   1          56m
connectors-0                  0/1     Running   0          112s
connectors-1                  0/1     Running   0          112s
```

Internal REST Endpoint : http://connectors:8083 (Inside Kubernetes)

OR

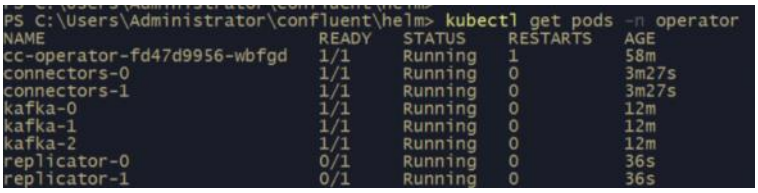
http://localhost:8083 (Inside Pod)

(6) поставим репликатор (в POWERSHELL)

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

проверим что работает

kubectl get pods -n operator



```
PS C:\Users\Administrator\confluent\helm> kubectl get pods -n operator
NAME                          READY   STATUS    RESTARTS   AGE
cc-operator-fd47d9956-wbfgd   1/1     Running   1          58m
connectors-0                  1/1     Running   0          3m27s
connectors-1                  1/1     Running   0          3m27s
kafka-0                       1/1     Running   0          12m
kafka-1                       1/1     Running   0          12m
kafka-2                       1/1     Running   0          12m
replicator-0                  0/1     Running   0          36s
replicator-1                  0/1     Running   0          36s
```

Access
   Internal REST Endpoint : http://replicator:8083 (Inside kubernetes)

   OR

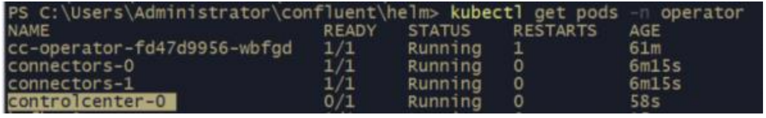
   http://localhost:8083 (Inside Pod)

(7) поставим контроль-центр (admin/Developer1) (в POWERSHELL)

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

проверим что работает

kubectl get pods -n operator



```
PS C:\Users\Administrator\confluent\helm> kubectl get pods -n operator
NAME                          READY   STATUS    RESTARTS   AGE
cc-operator-fd47d9956-wbfgd   1/1     Running   1          61m
connectors-0                  1/1     Running   0          6m15s
connectors-1                  1/1     Running   0          6m15s
controlcenter-0               0/1     Running   0          58s
```

Access
    Internal: http://controlcenter:9021 (Inside Kubernetes)

    Local Test:

     kubectl -n operator port-forward controlcenter-0 12345:9021
     Open on browser: http://localhost:12345

(8) поставим ksql (в POWERSHELL)

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

проверим что работает

kubectl get pods -n operator

. Access
    Internal: http://ksql:8088 (Inside kubernetes)

    OR

    http://localhost:8088 (Inside Pod)

(10) пробросим порты

https://docs.nginx.com/nginx-ingress-controller/configuration/global-configuration/configmap-resource/
https://kubernetes.github.io/ingress-nginx/user-guide/exposing-tcp-udp-services/

kubectl apply -f bootstrap.yaml -n operator

kubectl get services -n operator

kubectl get configmap -n operator

kubectl apply -f kafka-ingress.yaml -n operator
kubectl apply -f tcp-services.yaml -n operator

kubectl port-forward -n operator service/kafka-bootstrap 9092

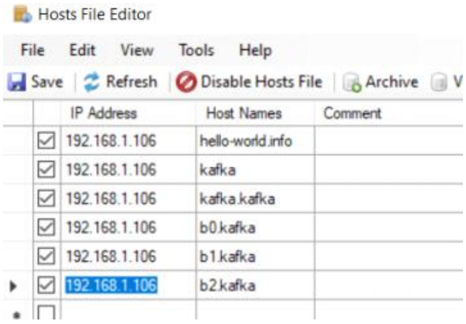
```
apiVersion: v1
kind: ConfigMap
metadata:
  name: tcp-services
  namespace: operator
```

```
data:
  9000: "default/example-go:8080"
  9093: "operator/kafka-bootstrap:9092"
  9095: "operator/kafka-0-internal:9092"
  9097: "operator/kafka-1-internal:9099"
  9099: "operator/kafka-2-internal:9092"
```

```
apiVersion: v1
kind: Service
metadata:
  name: kafka-bootstrap
  namespace: operator
  labels:
    app: kafka-bootstrap
spec:
  ports:
    - name: external
      port: 9092
      protocol: TCP
      targetPort: 9092
  selector:
    physicalstatefulcluster.core.confluent.cloud/name: kafka
    physicalstatefulcluster.core.confluent.cloud/version: v1
  type: ClusterIP
```

(11) смотрим сеть и редактируем hosts

```
PS C:\Users\Administrator\confluent\helm> minikube ip
192.168.1.106
```



	IP Address	Host Names	Comment
☑	192.168.1.106	hello-world.info	
☑	192.168.1.106	kafka	
☑	192.168.1.106	kafka.kafka	
☑	192.168.1.106	b0.kafka	
☑	192.168.1.106	b1.kafka	
☑	192.168.1.106	b2.kafka	
□			

(12) смотри куда конектится

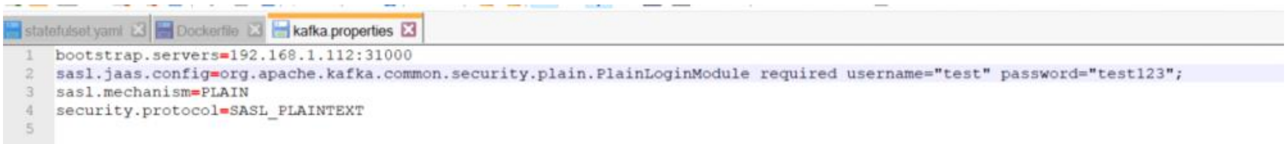
```
kubectl get kafka -n operator -oyaml
```

From <https://docs.confluent.io/current/installation/operator/co-deployment.html>

```
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
```

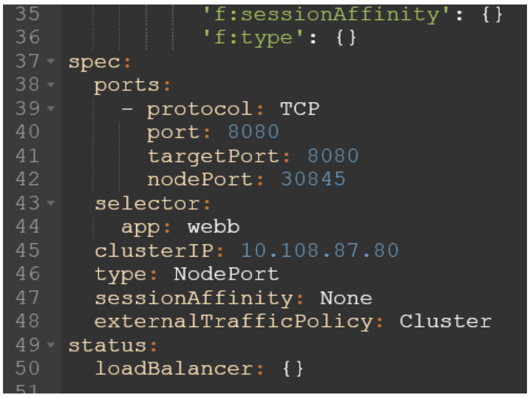
security.protocol=SASL_PLAINTEXT

>

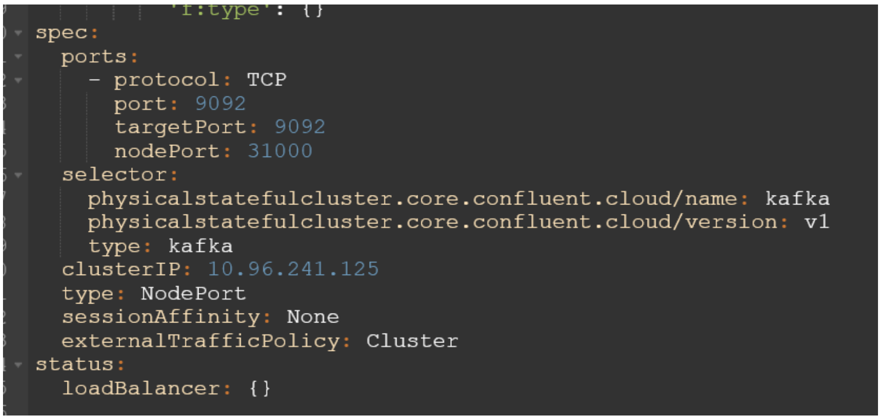


```
bootstrap.servers=192.168.1.112:31000
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="test" password="test123";
sasl.mechanism=PLAIN
security.protocol=SASL_PLAINTEXT
```

cd C:\ProgramFilesMy\confluent-6.0.0\bin\windows
kafka-topics --bootstrap-server kafka:31000 --command-config kafka.properties --create --replication-factor 1 --partitions 1 --topic example



```
          'f:sessionAffinity': {}
          'f:type': {}
spec:
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
      nodePort: 30845
  selector:
    app: webb
  clusterIP: 10.108.87.80
  type: NodePort
  sessionAffinity: None
  externalTrafficPolicy: Cluster
status:
  loadBalancer: {}
```



```
          'f:type': {}
spec:
  ports:
    - protocol: TCP
      port: 9092
      targetPort: 9092
      nodePort: 31000
  selector:
    physicalstatefulcluster.core.confluent.cloud/name: kafka
    physicalstatefulcluster.core.confluent.cloud/version: v1
    type: kafka
  clusterIP: 10.96.241.125
  type: NodePort
  sessionAffinity: None
  externalTrafficPolicy: Cluster
status:
  loadBalancer: {}
```

kafka-topics --bootstrap-server kafka:31000 --command-config kafka.properties --create --replication-factor 1 --partitions 1 --topic example

kafka-topics --bootstrap-server kafka:31000 --command-config kafka.properties --create --replication-factor 1 --partitions 1 --topic example

cd C:\programmfilesmy\confluent-6.0.0\confluent-6.0.0\bin\windows
kafka-broker-api-versions --command-config kafka.properties --bootstrap-server kafka:31000

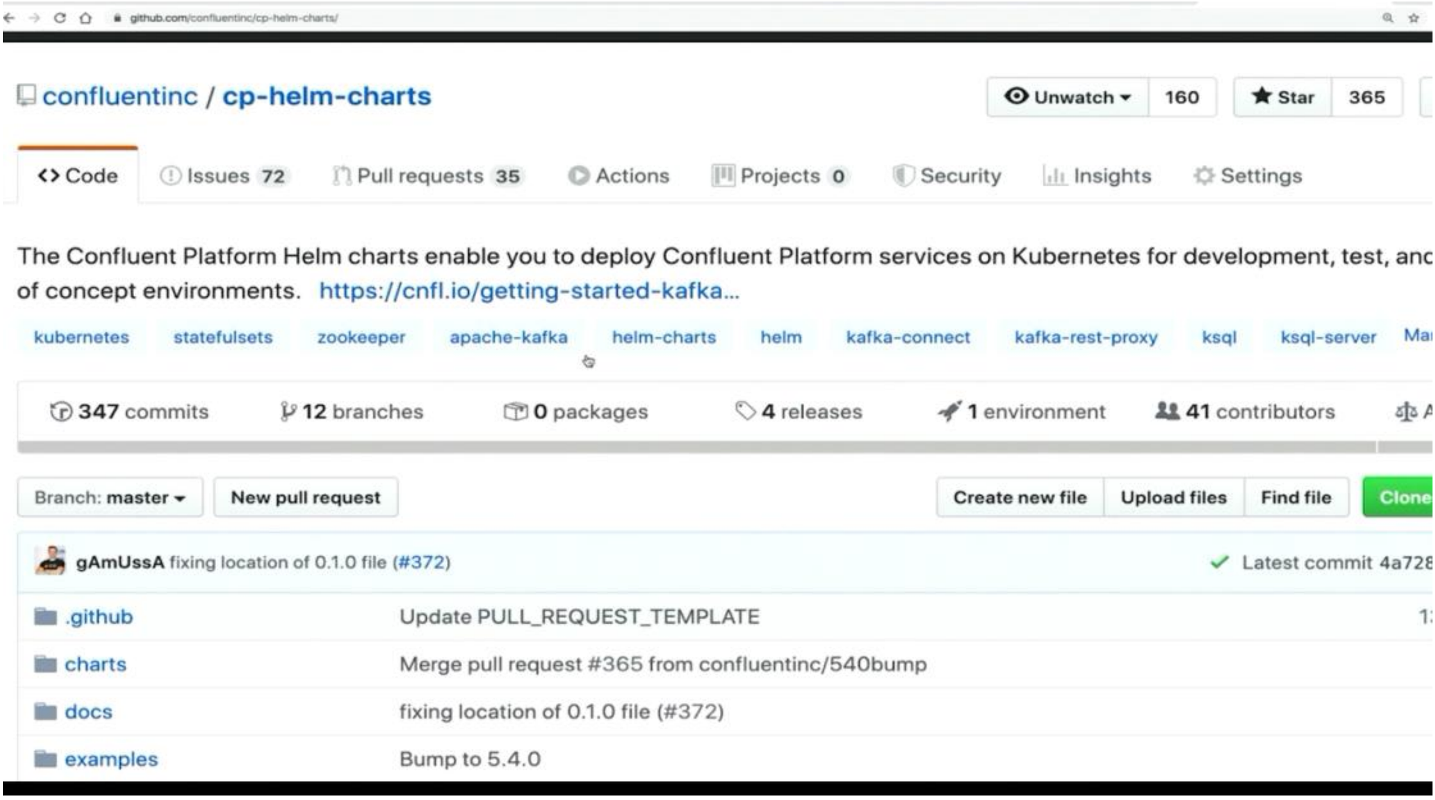
kubectl port-forward svc/controlcenter 9021:9021 -n operator

kubectl port-forward svc/kafka-bootstrap-np 31101:9092 -n operator
service/kafka-bootstrap-np

./kafka-topics --bootstrap-server kafka:31000 --command-config kafka.properties --create --replication-factor 1 --partitions 1 --topic example2

бесплатный оператор

27 октября 2020 г. 14:39



confluentinc / cp-helm-charts

The Confluent Platform Helm charts enable you to deploy Confluent Platform services on Kubernetes for development, test, anc of concept environments. https://cnfl.io/getting-started-kafka...

kubernetes statefulsets zookeeper apache-kafka helm-charts helm kafka-connect kafka-rest-proxy ksql ksql-server Ma

347 commits 12 branches 0 packages 4 releases 1 environment 41 contributors A

gAmUssA fixing location of 0.1.0 file (#372) ✓ Latest commit 4a728

.github	Update PULL_REQUEST_TEMPLATE	1
charts	Merge pull request #365 from confluentinc/540bump	
docs	fixing location of 0.1.0 file (#372)	
examples	Bump to 5.4.0	

kafka helm charts

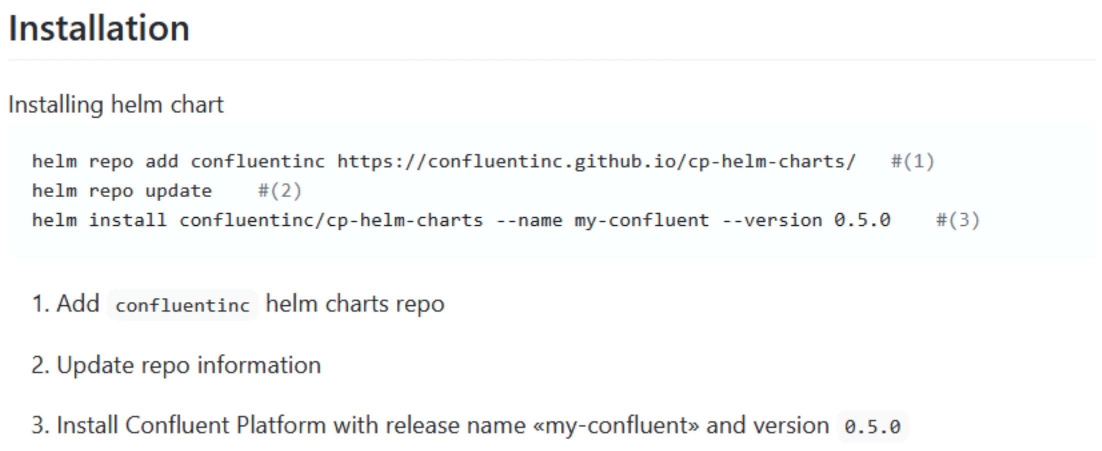
24 октября 2020 г. 19:24

helm chart repo

https://docs.confluent.io/5.0.0/installation/installing_cp/cp-helm-charts/docs/index.html

https://stackoverflow.com/questions/58528034/what-is-the-easiest-way-to-get-a-kafka-cluster-in-kubernetes

как установить



Installation

Installing helm chart

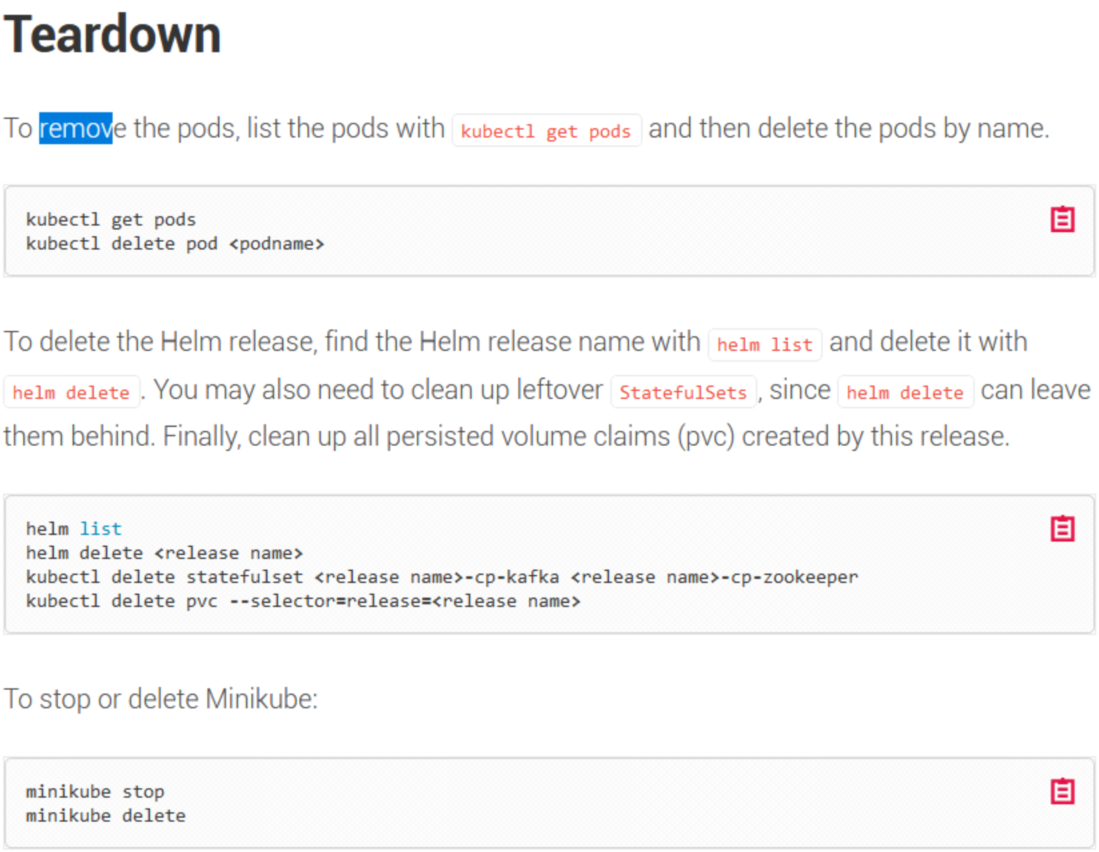
```
helm repo add confluentinc https://confluentinc.github.io/cp-helm-charts/   #(1)
helm repo update    #(2)
helm install confluentinc/cp-helm-charts --name my-confluent --version 0.5.0    #(3)
```

1. Add `confluentinc` helm charts repo
2. Update repo information
3. Install Confluent Platform with release name «my-confluent» and version `0.5.0`

https://github.com/confluentinc/cp-helm-charts

как удалить

https://docs.confluent.io/5.0.0/installation/installing_cp/cp-helm-charts/docs/index.html



Teardown

To remove the pods, list the pods with `kubectl get pods` and then delete the pods by name.

```
kubectl get pods
kubectl delete pod <podname>
```

To delete the Helm release, find the Helm release name with `helm list` and delete it with `helm delete`. You may also need to clean up leftover `StatefulSets`, since `helm delete` can leave them behind. Finally, clean up all persisted volume claims (pvc) created by this release.

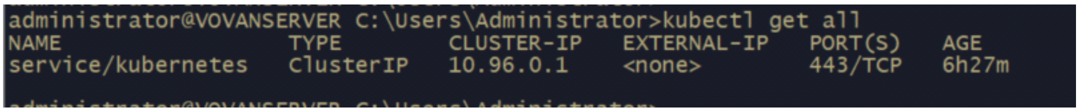
```
helm list
helm delete <release name>
kubectl delete statefulset <release name>-cp-kafka <release name>-cp-zookeeper
kubectl delete pvc --selector=release=<release name>
```

To stop or delete Minikube:

```
minikube stop
minikube delete
```

helm delete profuse-teaching
kubectl delete statefulset profuse-teaching-cp-kafka profuse-teaching-cp-zookeeper
kubectl delete pvc --selector=release=profuse-teaching

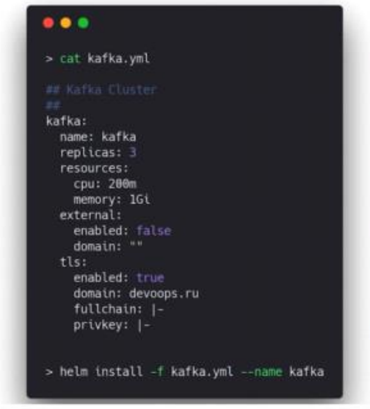
kubectl get all выведет пусто



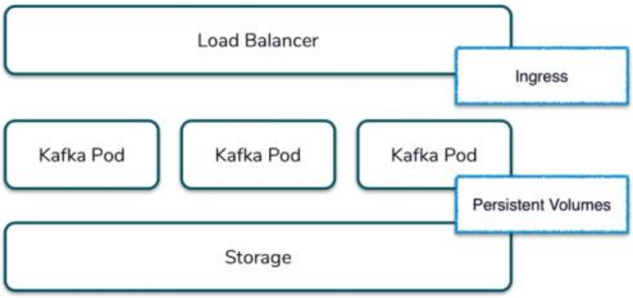
```
administrator@VOVANSERVER C:\Users\Administrator>kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   6h27m
```

как настроить yaml конфиг

Confluent Operator - Automated Provisioning



```
> cat kafka.yml

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 3
  resources:
    cpu: 200m
    memory: 1Gi
  external:
    enabled: false
    domain: ""
  tls:
    enabled: true
    domain: devoops.ru
    fullchain: |-
    privkey: |-


> helm install -f kafka.yml --name kafka
```



kafka monitoring

25 октября 2020 г.   12:16

https://github.com/confluentinc/cp-helm-charts

Monitoring

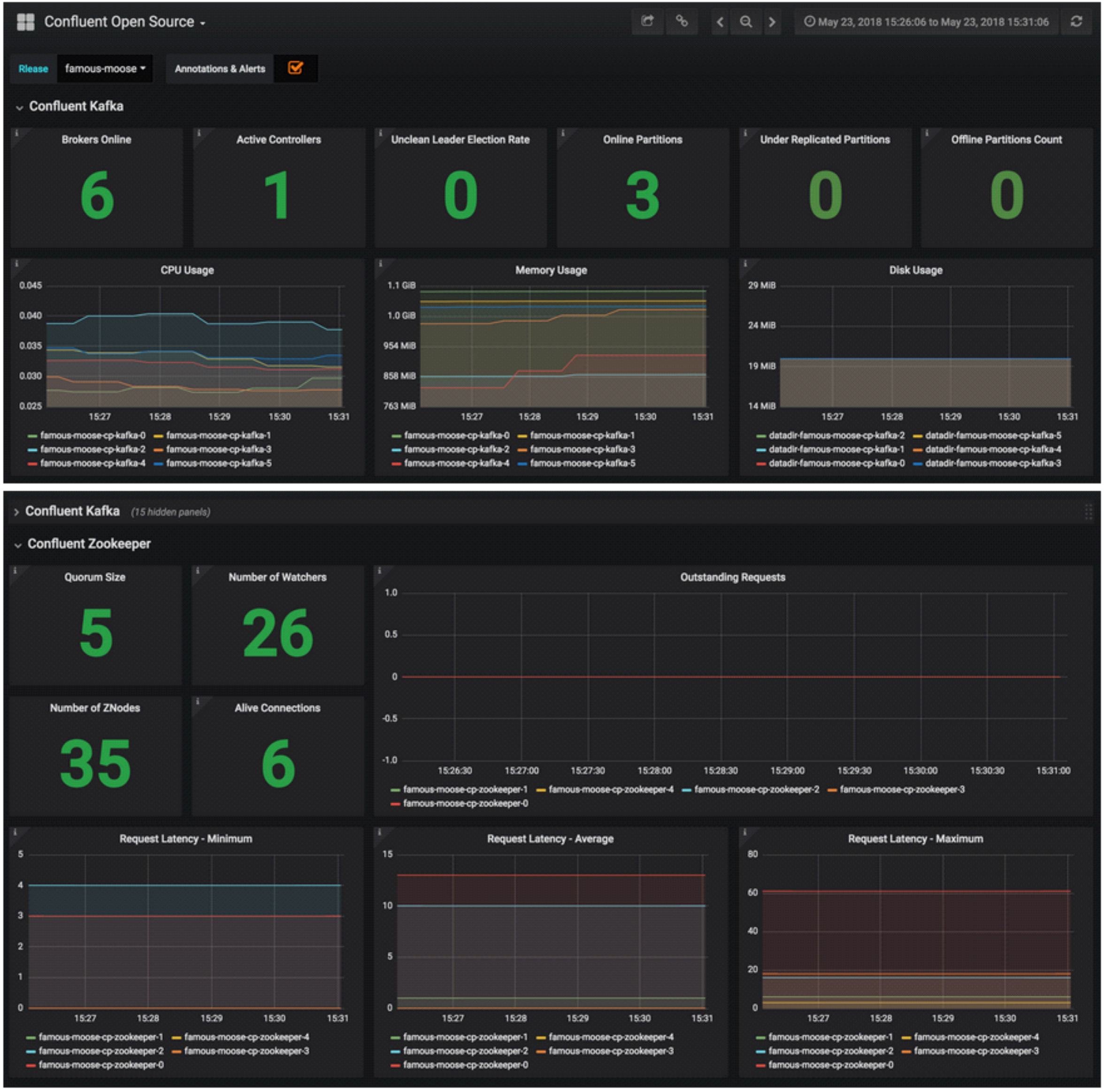
JMX Metrics are enabled by default for all components, Prometheus JMX Exporter is installed as a sidecar container along with all Pods.

1. Install Prometheus and Grafana in same Kubernetes cluster using helm

```
helm install stable/prometheus
helm install stable/grafana
```

2. Add Prometheus as Data Source in Grafana, url should be something like: `http://illmannered-marmot-prometheus-server:9090`

3. Import dashboard under grafana-dashboard into Grafana

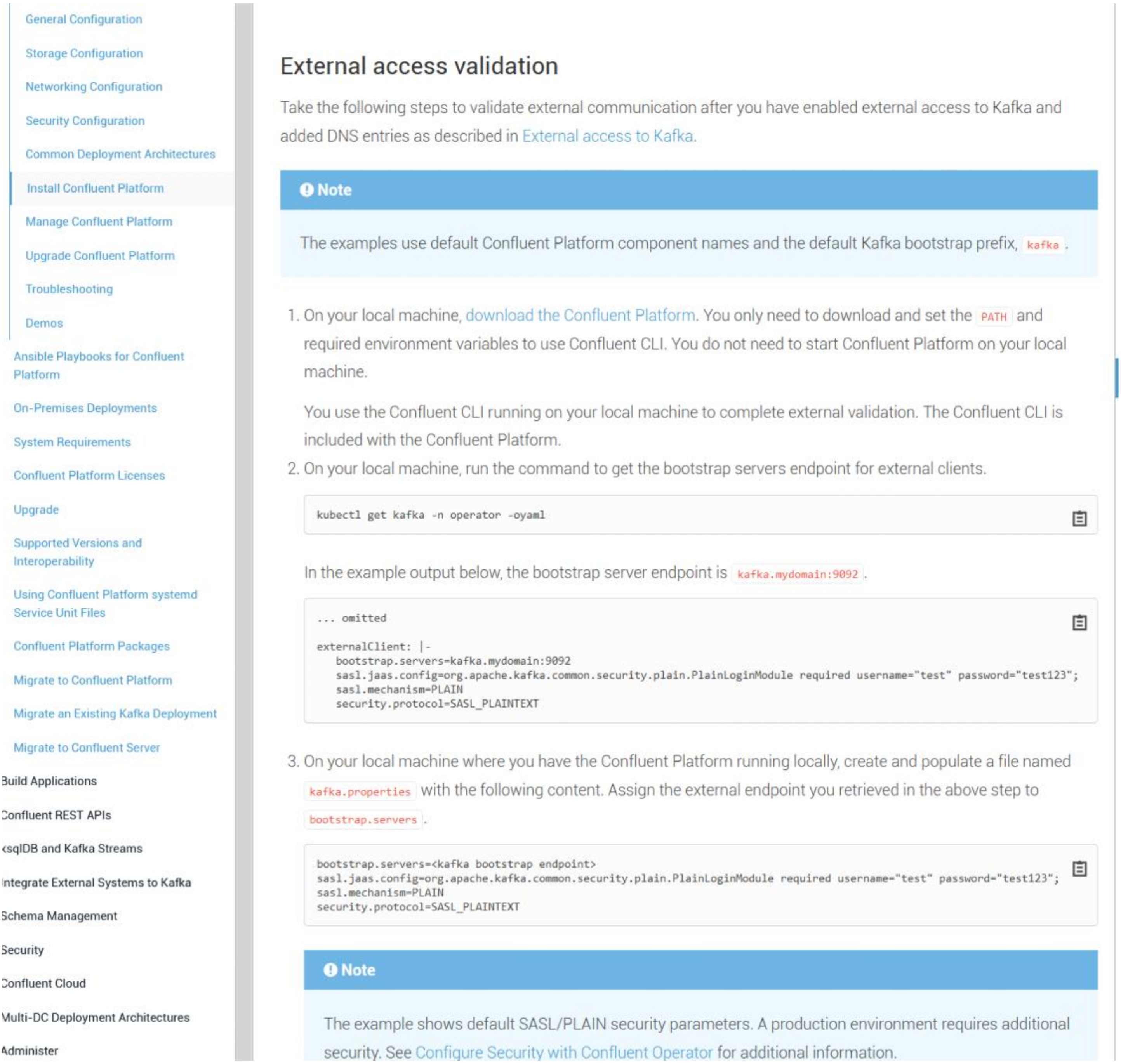


проверим с локалки

27 октября 2020 г. 23:00

https://docs.confluent.io/current/quickstart/ce-quickstart.html#ce-quickstart

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



General Configuration

Storage Configuration

Networking Configuration

Security Configuration

Common Deployment Architectures

Install Confluent Platform

Manage Confluent Platform

Upgrade Confluent Platform

Troubleshooting

Demos

Ansible Playbooks for Confluent Platform

On-Premises Deployments

System Requirements

Confluent Platform Licenses

Upgrade

Supported Versions and Interoperability

Using Confluent Platform systemd Service Unit Files

Confluent Platform Packages

Migrate to Confluent Platform

Migrate an Existing Kafka Deployment

Migrate to Confluent Server

Build Applications

Confluent REST APIs

ksqlDB and Kafka Streams

Integrate External Systems to Kafka

Schema Management

Security

Confluent Cloud

Multi-DC Deployment Architectures

Administer

External access validation

Take the following steps to validate external communication after you have enabled external access to Kafka and added DNS entries as described in External access to Kafka.

Note

The examples use default Confluent Platform component names and the default Kafka bootstrap prefix, `kafka`.

1. On your local machine, download the Confluent Platform. You only need to download and set the `PATH` and required environment variables to use Confluent CLI. You do not need to start Confluent Platform on your local machine.

   You use the Confluent CLI running on your local machine to complete external validation. The Confluent CLI is included with the Confluent Platform.

2. On your local machine, run the command to get the bootstrap servers endpoint for external clients.

```
kubectl get kafka -n operator -oyaml
```

   In the example output below, the bootstrap server endpoint is `kafka.mydomain:9092`.

```
... omitted

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

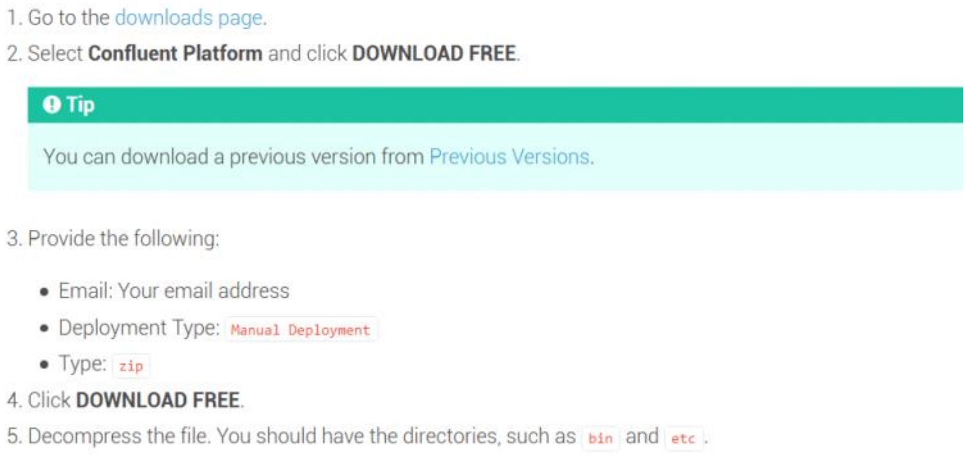
3. On your local machine where you have the Confluent Platform running locally, create and populate a file named `kafka.properties` with the following content. Assign the external endpoint you retrieved in the above step to `bootstrap.servers`.

```
bootstrap.servers=<kafka bootstrap endpoint>
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="test" password="test123";
sasl.mechanism=PLAIN
security.protocol=SASL_PLAINTEXT
```

   Note

   The example shows default SASL/PLAIN security parameters. A production environment requires additional security. See Configure Security with Confluent Operator for additional information.

загрузим на локалку пакет



1. Go to the downloads page.
2. Select **Confluent Platform** and click **DOWNLOAD FREE**.

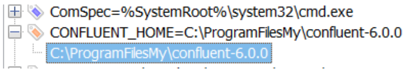
Tip

You can download a previous version from Previous Versions.

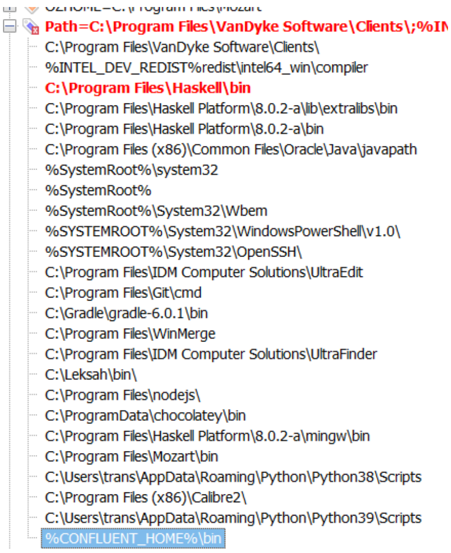
3. Provide the following:
   - Email: Your email address
   - Deployment Type: `Manual Deployment`
   - Type: `zip`
4. Click **DOWNLOAD FREE**.
5. Decompress the file. You should have the directories, such as `bin` and `etc`.

C:\ProgramFilesMy\confluent-6.0.0

установим path



```
ComSpec=%SystemRoot%\system32\cmd.exe
CONFLUENT_HOME=C:\ProgramFilesMy\confluent-6.0.0
    C:\ProgramFilesMy\confluent-6.0.0
```



```
OZHOME=C:\Program Files\Mozart
Path=C:\Program Files\VanDyke Software\Clients\;%IN
    C:\Program Files\VanDyke Software\Clients\
    %INTEL_DEV_REDIST%redist\intel64_win\compiler
    C:\Program Files\Haskell\bin
    C:\Program Files\Haskell Platform\8.0.2-a\lib\extralibs\bin
    C:\Program Files\Haskell Platform\8.0.2-a\bin
    C:\Program Files (x86)\Common Files\Oracle\Java\javapath
    %SystemRoot%\system32
    %SystemRoot%
    %SystemRoot%\System32\Wbem
    %SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
    %SYSTEMROOT%\System32\OpenSSH\
    C:\Program Files\IDM Computer Solutions\UltraEdit
    C:\Program Files\Git\cmd
    C:\Gradle\gradle-6.0.1\bin
    C:\Program Files\WinMerge
    C:\Program Files\IDM Computer Solutions\UltraFinder
    C:\Leksah\bin\
    C:\Program Files\nodejs\
    C:\ProgramData\chocolatey\bin
    C:\Program Files\Haskell Platform\8.0.2-a\mingw\bin
    C:\Program Files\Mozart\bin
    C:\Users\trans\AppData\Roaming\Python\Python38\Scripts
    C:\Program Files (x86)\Calibre2\
    C:\Users\trans\AppData\Roaming\Python\Python39\Scripts
    %CONFLUENT_HOME%\bin
```

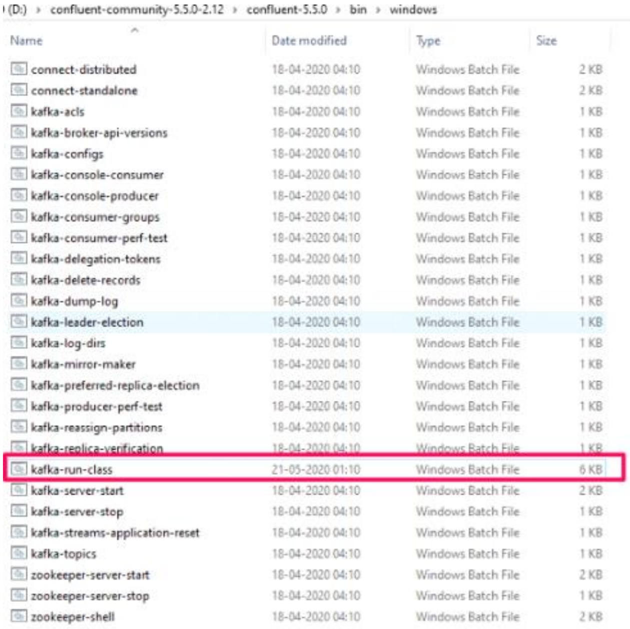
```
kubectl port-forward -n operator svc/kafka 9071
```

чтобы локально запустить тест (POWERSHELL)

```
cd C:\ProgramFilesMy\confluent-6.0.0\bin\windows
.\kafka-broker-api-versions --command-config kafka.properties --bootstrap-server kafka:9071
```

https://medium.com/@praveenkumarsingh/confluent-kafka-on-windows-how-to-fix-classpath-is-empty-cf7c31d9c787

```
rem classpath addition for LSB style path
if exist %BASE_DIR%\share\java\kafka\* (
call:concat %BASE_DIR%\share\java\kafka\*
)
```



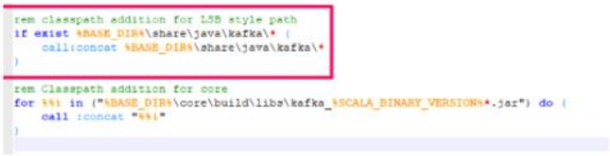
(D:) › confluent-community-5.5.0-2.12 › confluent-5.5.0 › bin › windows

Name	Date modified	Type	Size
connect-distributed	18-04-2020 04:10	Windows Batch File	2 KB
connect-standalone	18-04-2020 04:10	Windows Batch File	2 KB
kafka-acls	18-04-2020 04:10	Windows Batch File	1 KB
kafka-broker-api-versions	18-04-2020 04:10	Windows Batch File	1 KB
kafka-configs	18-04-2020 04:10	Windows Batch File	1 KB
kafka-console-consumer	18-04-2020 04:10	Windows Batch File	1 KB
kafka-console-producer	18-04-2020 04:10	Windows Batch File	1 KB
kafka-consumer-groups	18-04-2020 04:10	Windows Batch File	1 KB
kafka-consumer-perf-test	18-04-2020 04:10	Windows Batch File	1 KB
kafka-delegation-tokens	18-04-2020 04:10	Windows Batch File	1 KB
kafka-delete-records	18-04-2020 04:10	Windows Batch File	1 KB
kafka-dump-log	18-04-2020 04:10	Windows Batch File	1 KB
kafka-leader-election	18-04-2020 04:10	Windows Batch File	1 KB
kafka-log-dirs	18-04-2020 04:10	Windows Batch File	1 KB
kafka-mirror-maker	18-04-2020 04:10	Windows Batch File	1 KB
kafka-preferred-replica-election	18-04-2020 04:10	Windows Batch File	1 KB
kafka-producer-perf-test	18-04-2020 04:10	Windows Batch File	1 KB
kafka-reassign-partitions	18-04-2020 04:10	Windows Batch File	1 KB
kafka-replica-verification	18-04-2020 04:10	Windows Batch File	1 KB
kafka-run-class	21-05-2020 01:10	Windows Batch File	6 KB
kafka-server-start	18-04-2020 04:10	Windows Batch File	2 KB
kafka-server-stop	18-04-2020 04:10	Windows Batch File	1 KB
kafka-streams-application-reset	18-04-2020 04:10	Windows Batch File	1 KB
kafka-topics	18-04-2020 04:10	Windows Batch File	1 KB
zookeeper-server-start	18-04-2020 04:10	Windows Batch File	2 KB
zookeeper-server-stop	18-04-2020 04:10	Windows Batch File	1 KB
zookeeper-shell	18-04-2020 04:10	Windows Batch File	2 KB

addition for core line.

> *rem classpath addition for LSB style path*
> *if exist %BASE_DIR%\share\java\kafka\* (*
> *call:concat %BASE_DIR%\share\java\kafka\**
> *)*

Your code should look like this :



```
rem classpath addition for LSB style path
if exist %BASE_DIR%\share\java\kafka\* (
    call:concat %BASE_DIR%\share\java\kafka\*
)

rem Classpath addition for core
for %%i in ("%BASE_DIR%\core\build\libs\kafka_%SCALA_BINARY_VERSION%*.jar") do (
    call :concat "%%i"
)
```

авто-скейл

27 октября 2020 г. 23:05

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

Scale Kafka clusters and balance data

Scale up

Starting in Confluent Platform 6.0.0, Self-Balancing is the recommended way to rebalance loads when Kafka brokers are added or removed. Self-Balancing is disabled by default.

Change the following settings to enable Self-Balancing or to rebalance Kafka for any uneven load when Self-Balancing is enabled. For a complete list of available settings you can use to control Self-Balancing, see Configuration Options and Commands for Self-Balancing Clusters. You can pass the settings in `configOverrides` in the `kafka` section of the configuration file (`$VALUES_FILE`).

```
kafka:
  configOverrides:
    server:
      - confluent.balancer.enable=                   ----- [1]
      - confluent.balancer.heal.uneven.load.trigger= ----- [2]
```

- [1] Set confluent.balancer.enable to `true` to enable Self-Balancing.
- [2] Set confluent.balancer.heal.uneven.load.trigger to `ANY_UNEVEN_LOAD` to balance the load across the cluster whenever an imbalance is detected. The default is `EMPTY_BROKER`.

After you enable Self-Balancing, to scale up the cluster, perform the following:

1. Increase the number of Kafka replicas in the configuration file (`$VALUES_FILE`):

```
kafka:
  replicas:
```

2. Update Kafka:

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

If you need to use Auto Data Balancer, first turn off Self-Balancing as Self-Balancing and Auto Data Balancer cannot be used together, and then refer to the 5.5.1 or an older version of the documentation for the scale up process.

Scale down

Scaling down Kafka clusters is not supported in the current version of Confluent Operator.