Конфигурация kubectl

Взаимодействие с kube-apiserver

```
# Получить описание всех pod в kube-system namespace
curl https://API_SERVER_ADDRESS/api/v1/namespaces/kube-system/pods/ --header
"Authorization: Bearer $TOKEN" --cacert ca.crt

# Посмотреть логи конкретного pod в kube-system namespace
curl https://API_SERVER_ADDRESS/api/v1/namespaces/kube-system/pods/POD_NAME/log --header
"Authorization: Bearer $TOKEN" --cacert ca.crt
```

Также есть возможность использовать команду kubectl proxy для проброса API на локальную машину.

- Библиотеки для распространенных языков программирования
- Пример для Go

kubectl

- CLI утилита, распространяемая в виде бинарного файла
- Конфигурация для подключения к кластеру

Объекты в кластере можно:

- Создать (**kubectl create**)
- Обновить (**kubectl apply**)
- Получить (**kubectl get**)
- Посмотреть (**kubectl describe**)
- Удалить (**kubectl delete**)
- ...

Обзор kubectl

kubectl

Конфигурация kubectl - это **контекст** (context)

Контекст (context) - это комбинация:

- **cluster** - API сервера
- **user** - пользователь для подключения к кластеру
- **namespace** - область видимости (не обязательно, по умолчанию default)

> Информацию о контекстах kubectl сохраняет в файле `~/.kube/config`

kubectl | cluster

Кластер (cluster) - это:

- **server** - адрес kubernetes API-сервера
- **certificate-authority** - корневой сертификат (которым подписан SSL-сертификат API-сервера)
- **name** - имя для идентификации

```
1   clusters: #  Список кластеров
2   - cluster:
3       certificate-authority: ca.crt
4       server: https://35.198.140.134
5     name: kube-cluster
```

kubectl | user

Пользователь (**user**) - это:

- Данные для аутентификации:
  - username + password (Basic Auth)
  - client key + client certificate
  - token
  - auth-provider config (например GCP)
- name (имя) для идентификации в конфиге

```yaml
users: # ⬅️ Список пользователей и способов их авторизации
- name: kube-user
  user:
    client-certificate: kube-user.crt
    client-key: kube-user.key
```

kubectl | context

Контекст (**context**) - это:

- **cluster** - имя кластера из списка clusters
- **user** - имя пользователя из списка users
- **namespace** - область видимости по-умолчанию (не обязательно)
- **name** - имя контекста для идентификации

```
1   contexts:  # 🔙 Список контекстов
2   - context:
3       cluster: kube-cluster
4       user: kube-user
5     name: kube-context
```

Пример конфигурации kubectl

```yaml
apiVersion: v1
clusters:                       # ⬅️ Список кластеров
- cluster:
    certificate-authority: ca.crt
    server: https://35.198.140.134
  name: kube-cluster
contexts:                       # ⬅️ Список контекстов
- context:
    cluster: kube-cluster
    user: kube-user
  name: kube-context
current-context: kube-context # ⬅️ Текущий контекст
kind: Config
preferences: {}
users:                          # ⬅️ Список пользователей и способов их авторизации
- name: kube-user
  user:
    client-certificate: user.crt
    client-key: user.key
```

Порядок конфигурации kubectl

Обычно порядок конфигурирования `kubectl` следующий:

```
# Указать кластер:
kubectl config set-cluster … cluster_name

# Указать данные пользователя (credentials):
kubectl config set-credentials … user_name

# Создать контекст:
kubectl config set-context context_name --cluster=cluster_name --user=user_name

# Использовать контекст:
kubectl config use-context context_name
```

Команды kubectl

kubectl Cheat Sheet

```
# Создать ресурс из манифеста
kubectl create -f manifest.yml
kubectl apply -f manifest.yml
kubectl apply -f link
kubectl apply -f directory/

# Получить список ресурсов
kubectl get pods

# Получить описание ресурса
kubectl describe pod POD_NAME
```

Еще один cheatsheet

Практика

kubectl config

Перейдем в папку `project/user-{your number}` (все дальнейшие команды по настройке будем выполнять из нее).

```
1  cd ~/project/user-{your number}
```

В этой папке должны лежать сертификаты, пример конфигурации `kubectl`, а также файл `Cluster_info`, содержащий IP-адрес и имя кластера вашего кластера

kubectl config | Cluster

Добавим **cluster** (*cluster_name* и *cluster_address* можно найти в `project/userN/Cluster_info`):

```
kubectl config set-cluster <cluster_name> --server=<cluster_address> \
--certificate-authority=ca.crt --embed-certs=true
```

Посмотрим что получилось:

```
kubectl config view
```

kubectl config view

```yaml
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /home/theia/project/user-0/ca.crt # ⬅️ путь до сертификата
    server: https://35.234.124.242                           # ⬅️ ваш адрес
  name: gke_cd-k8s-236617_europe-west3-a_k8s-0               # ⬅️ имя кластера
contexts: []
current-context: ""
kind: Config
preferences: {}
users: []
```

kubectl config | User

Добавим **user**:

```
kubectl config set-credentials kubeuser --client-certificate=server.crt \
--client-key=server-key.pem --embed-certs=true
```

Посмотрим что получилось:

```
kubectl config view
```

kubectl config view

Должна появиться новая информация о пользователе `kubeuser`:

```
1   users:
2   - name: kubeuser
3     user:
4       client-certificate-data: REDACTED
5       client-key-data: REDACTED
```

kubectl config | Context

Создадим контекст, связывающий *user* и *cluster*:

```
kubectl config set-context clustercontext \
--cluster=<cluster_name> \
--user=kubeuser
```

- Не забудьте подставить свои значения в *cluster_name* и *cluster_address*.
- Имя кластера должно совпадать с именем из `kubectl config set-cluster`.
- Имя контекста может быть задано произвольно.

kubectl config view

```
kubectl config view
```

К выводу добавились следующие строки:

```
  contexts:
  - context:
      cluster: gke_cd-k8s-236617_europe-west3-a_k8s-0 # ⬅️ имя кластера в конфиге
      user: kubeuser                                  # ⬅️ имя пользователя
    name: contextname                                 # ⬅️ имя конекста
  current-context: ""                                 # ⬅️ текущий контекст
```

kubectl | Работа с контекстами

- Посмотрим текущий контекст:

```
kubectl config current-context
```

- Посмотрим список контекстов:

```
kubectl config get-contexts
```

- Установим свежеиспеченный контекст:

```
kubectl config use-context clustercontext
```

- Снова посмотрим текущий контекст:

```
kubectl config current-context
```

> В итоге, вы должны получить имя вашего контекста. `kubectl config current-context` в первый раз выдаст ошибку, это норма

kubectl | Проверка

Проверим, что мы можем взаимодействовать с кластером через `kubectl`

```
kubectl cluster-info
```

```
Kubernetes master is running at https://35.234.124.242
GLBCDefaultBackend is running at https://35.234.124.242/api/v1/namespaces/kube-system/services/default-http-backend:http/proxy
Heapster is running at https://35.234.124.242/api/v1/namespaces/kube-system/services/heapster/proxy
KubeDNS is running at https://35.234.124.242/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
Metrics-server is running at https://35.234.124.242/api/v1/namespaces/kube-system/services/https:metrics-server:/proxy
```