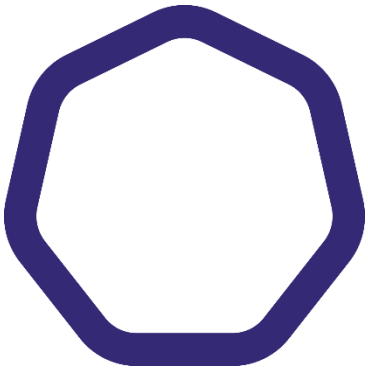
Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский технический университет связи и информатики»
(МТУСИ)



Кафедра: Сетевые информационные технологии и сервисы

Отчёт по лабораторной работе №6
«Открытие доступа к приложению»

Выполнил:
Студенты 1 курса
Группы М092401(75)
Цыганков Р.О.
Проверил:
к.т.н., Шалагинов А. В.

_____

Москва 2024

Задание

Познакомиться с сервисами в Kubernetes, разобраться с тем, как метки и объекты LabelSelector связаны с сервисом и открыть доступ к приложению вне кластера Kubernetes через сервис.

Выполнение работы

Для выполнения лабораторных работ необходимо:

- Зайти на сайт: https://kubernetes.io/ru/docs/tutorials/kubernetes-basics/expose/expose-intro/;
- Изучить теоретическую часть;
- Нажать на кнопку «Начать интерактивный урок».

Теоретическая часть работы:

Обзор сервисов Kubernetes

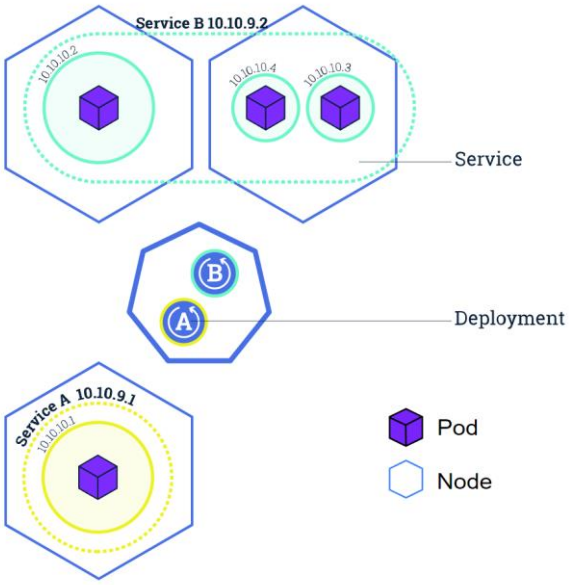
Под — это расходный материал в Kubernetes. У подов есть жизненный цикл. Когда рабочий узел завершается, запущенные поды в узле также уничтожаются. После этого ReplicaSet попытается автоматически вернуть кластер обратно в требуемое состояние, создавая новые поды, чтобы поддержать работоспособность приложения. Другой пример — бэкенд для обработки изображений с 3 репликами. Поскольку это взаимозаменяемые реплики, то они не влияют на фронтенд-часть, даже если под был уничтожен и пересоздан. Тем не менее, каждый под в кластере Kubernetes имеет уникальный IP-адрес, даже под на одном и том же узле, поэтому должен быть способ автоматической координации изменений между подами, чтобы приложения продолжали функционировать.

Сервис в Kubernetes — это абстрактный объект, который определяет логический набор подов и политику доступа к ним. Сервисы создают слабую связь между подами, которые от них зависят. Сервис создаётся в формате YAML (рекомендуемый формат) или JSON, как и все остальные объекты в Kubernetes. Как правило, набор подов для сервиса определяется LabelSelector (ниже описано, в каких случаях понадобится сервис без указания selector в спецификации).

Хотя у каждого пода есть уникальный IP-адрес, эти IP-адреса не доступны за пределами кластера без использования сервиса. Сервисы позволяют приложениям принимать трафик. Сервисы могут быть по-разному открыты, в зависимости от указанного поля type в ServiceSpec:

- ClusterIP (по умолчанию) - открывает доступ к сервису по внутреннему IP-адресу в кластере. Этот тип делает сервис доступным только внутри кластера;
- NodePort - открывает сервис на одном и том же порту каждого выбранного узла в кластере с помощью NAT. Делает сервис доступным вне кластера, используя <NodeIP>:<NodePort>. Является надмножеством ClusterIP;
- LoadBalancer - создает внешний балансировщик нагрузки в текущем облаке (если это поддерживается) и назначает фиксированный внешний IP-адрес для сервиса. Является надмножеством NodePort;
- ExternalName - открывает доступ к сервису с указанным именем (определённое в поле externalName в спецификации) и возвращает запись CNAME. Прокси не используется. Для этого типа требуется версия kube-dns 1.7 или выше.

Сервисы и метки:



Сервис направляет трафик через набор подов. Сервисы — это абстракция, позволяющая взаимозаменять поды Kubernetes без ущерба для приложения. Сервисы в Kubernetes находят и маршрутизируют трафик между зависимыми подами (это могут быть фронтенд- и бэкенд-компоненты приложения).

Сервисы для выбора набора подов используют метки и селекторы. Метки — пары ключ-значение, добавленные к объектам; например, они могут использоваться чтобы:

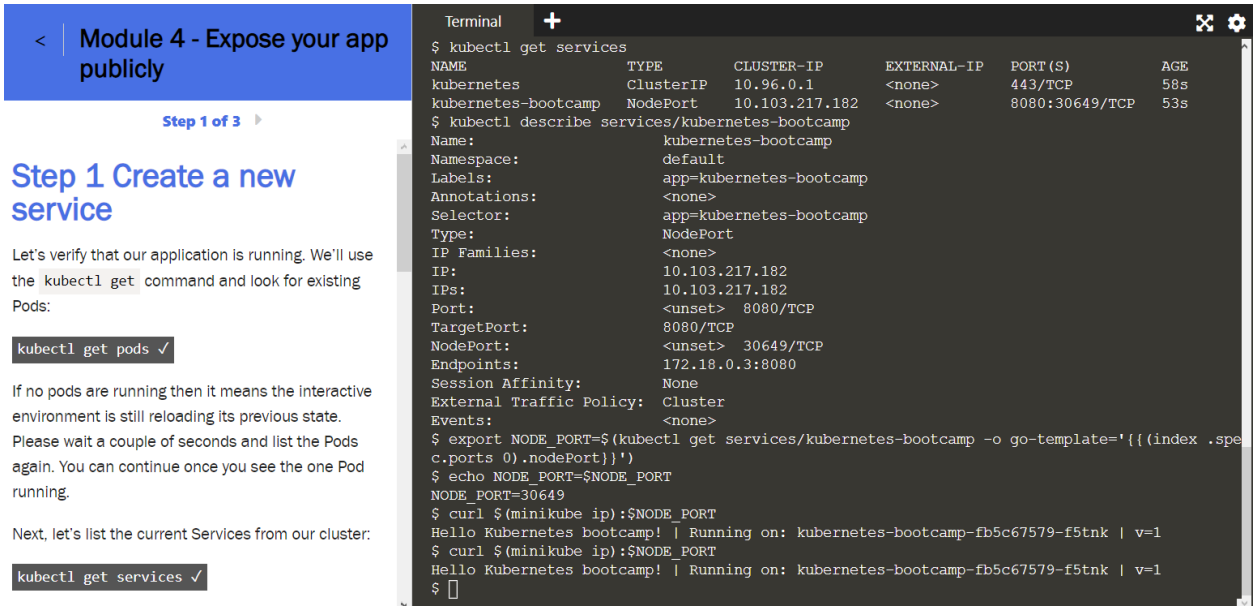
- Идентифицировать объекты для окружений разработки, тестирования и продакшена

- Добавить теги версии
- Классифицировать объекты через теги

Метки могут добавляться во время создания объектов или после этого. Они также могут быть изменены в любое время. Теперь давайте откроем доступ к приложению путём создания сервиса и добавление меток

Практическая часть работы:

1. Создание нового сервиса:



Module 4 - Expose your app publicly

Step 1 of 3

Step 1 Create a new service

Let's verify that our application is running. We'll use the `kubectl get` command and look for existing Pods:

```
kubectl get pods ✓
```

If no pods are running then it means the interactive environment is still reloading its previous state. Please wait a couple of seconds and list the Pods again. You can continue once you see the one Pod running.

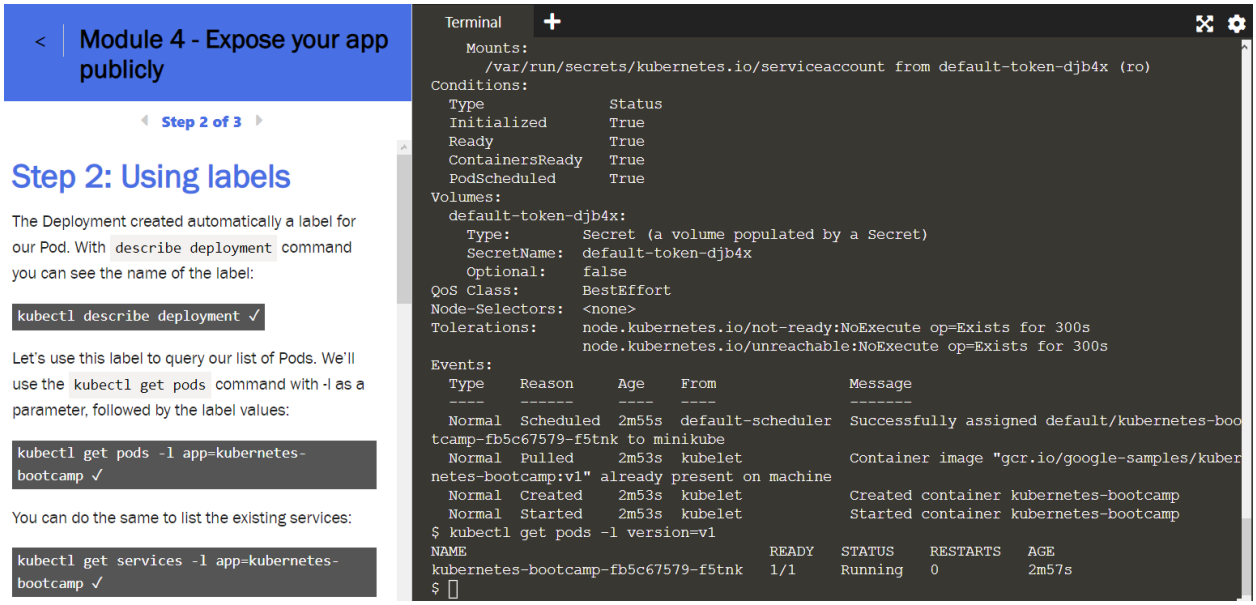
Next, let's list the current Services from our cluster:

```
kubectl get services ✓
```

Terminal

```
$ kubectl get services
NAME                  TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes            ClusterIP   10.96.0.1        <none>        443/TCP          58s
kubernetes-bootcamp   NodePort    10.103.217.182   <none>        8080:30649/TCP   53s
$ kubectl describe services/kubernetes-bootcamp
Name:                     kubernetes-bootcamp
Namespace:                default
Labels:                   app=kubernetes-bootcamp
Annotations:              <none>
Selector:                 app=kubernetes-bootcamp
Type:                     NodePort
IP Families:              <none>
IP:                       10.103.217.182
IPs:                      10.103.217.182
Port:                     <unset>  8080/TCP
TargetPort:               8080/TCP
NodePort:                 <unset>  30649/TCP
Endpoints:                172.18.0.3:8080
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
$ export NODE_PORT=$(kubectl get services/kubernetes-bootcamp -o go-template='{{(index .spe
c.ports 0).nodePort}}')
$ echo NODE_PORT=$NODE_PORT
NODE_PORT=30649
$ curl $(minikube ip):$NODE_PORT
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-fb5c67579-f5tnk | v=1
$ curl $(minikube ip):$NODE_PORT
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-fb5c67579-f5tnk | v=1
$
```

2. Использование меток:



Module 4 - Expose your app publicly

Step 2 of 3

Step 2: Using labels

The Deployment created automatically a label for our Pod. With `describe deployment` command you can see the name of the label:

```
kubectl describe deployment ✓
```

Let's use this label to query our list of Pods. We'll use the `kubectl get pods` command with -l as a parameter, followed by the label values:

```
kubectl get pods -l app=kubernetes-bootcamp ✓
```

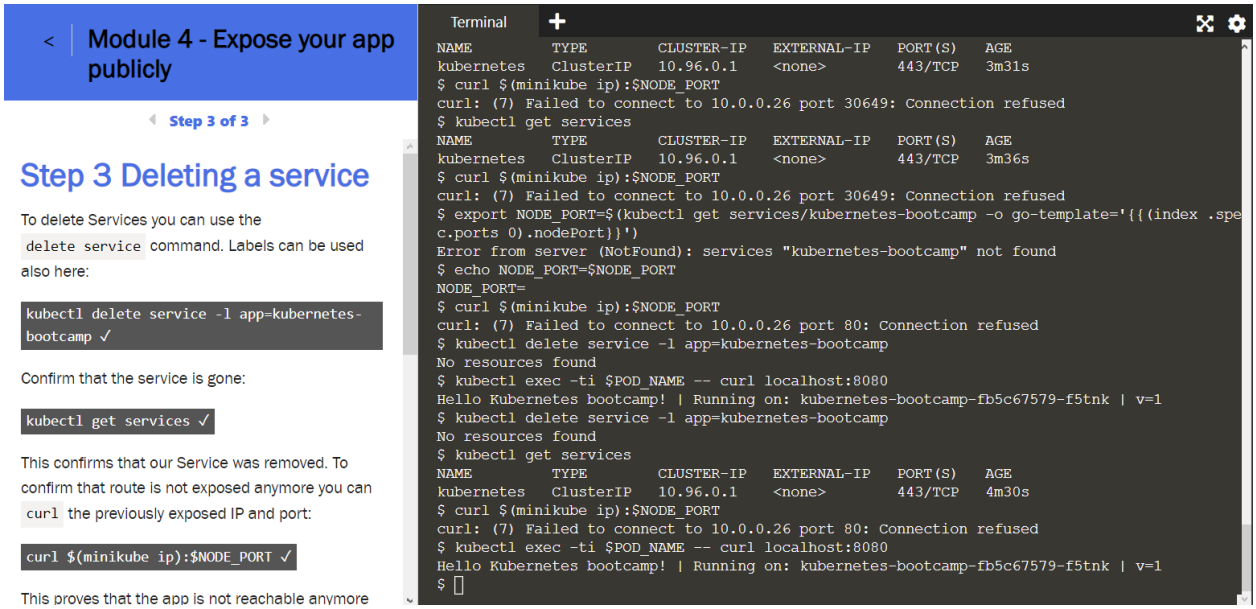
You can do the same to list the existing services:

```
kubectl get services -l app=kubernetes-bootcamp ✓
```

Terminal

```
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-djb4x (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-djb4x:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-djb4x
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                 node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age    From               Message
  ----    ------     ----   ----               -------
  Normal  Scheduled  2m55s  default-scheduler  Successfully assigned default/kubernetes-boo
tcamp-fb5c67579-f5tnk to minikube
  Normal  Pulled     2m53s  kubelet            Container image "gcr.io/google-samples/kuber
netes-bootcamp:v1" already present on machine
  Normal  Created    2m53s  kubelet            Created container kubernetes-bootcamp
  Normal  Started    2m53s  kubelet            Started container kubernetes-bootcamp
$ kubectl get pods -l version=v1
NAME                                  READY   STATUS    RESTARTS   AGE
kubernetes-bootcamp-fb5c67579-f5tnk   1/1     Running   0          2m57s
$
```

3. Удаление сервиса:



Заключение

В результате работы мы получили практические навыки работы с сервисами в Kubernetes, с метками и объектами LabelSelector связаны с сервисом и открытием доступа к приложению вне кластера Kubernetes через сервис.