Задача 2.6: Знакомство с Kubernetes и Minikube

Подготовка

Подготовим файл манифеста для нашего будущего пода:

1. Создадим и отредактируем манифест:
   `nano my-pod.yaml`
2. Указываем версию api - для пода используем v1 (подробнее тут - https://kubernetes.io/ru/docs/concepts/overview/kubernetes-api/):
   `apiVersion: v1`
3. Указываем тип объекта - в нашем случае pod:
   `kind: Pod`
4. Начинаем заполнять метаданные. Поля содержать информацию об объекте, например имя, метки и т.д.:

   ```
   metadata:
     name: webapp
   ```

5. Далее заполним раздел спецификации. Данный раздел содержит дополнительную информацию, которая зависит от объекта, который мы собираем запускать.:
   `spec:`
6. Так как мы создаем кластер с контейнером, то добавляем соответствующее свойство и заполняем только один элемент, так как планируем запустить только один контейнер в кластере:
   `containers:`
7. Указываем имя контейнера:
   `- name: webapp`
8. Указываем образ контейнера. Для примера, возьмем готовое простое веб-приложение:
   `image: karthequian/helloworld:latest`
9. Также укажем порты:

   ```
   ports:
   - containerPort: 80
   ```

Общий вид нашего манифеста для пода:

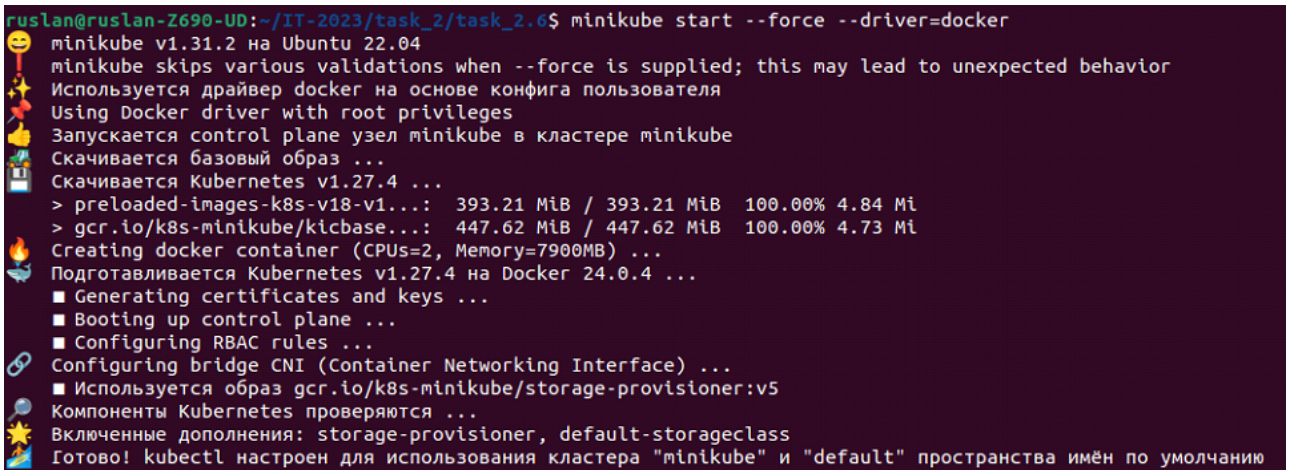
```
apiVersion: v1
kind: Pod
metadata:
  name: webapp
spec:
  containers:
  - name: webapp
```

```
    image: karthequian/helloworld:latest
    ports:
    - containerPort: 80
```

Запуск мини-кластера

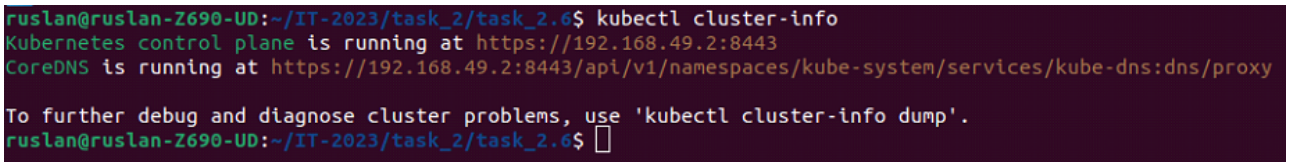
1. Запускаем кластер kubernetes с помощью minikube, с docker в качестве драйвера:

   ```
   minikube start --force --driver=docker
   ```



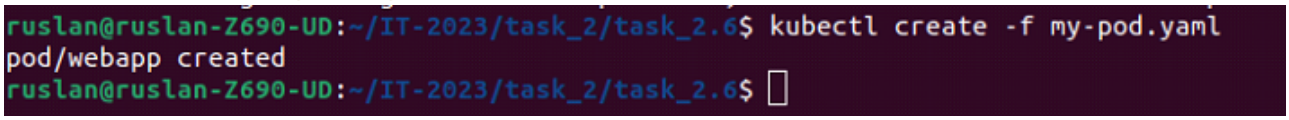
2. Проверяем состояние кластера с помощью kubectl:

   ```
   kubectl cluster-info
   ```



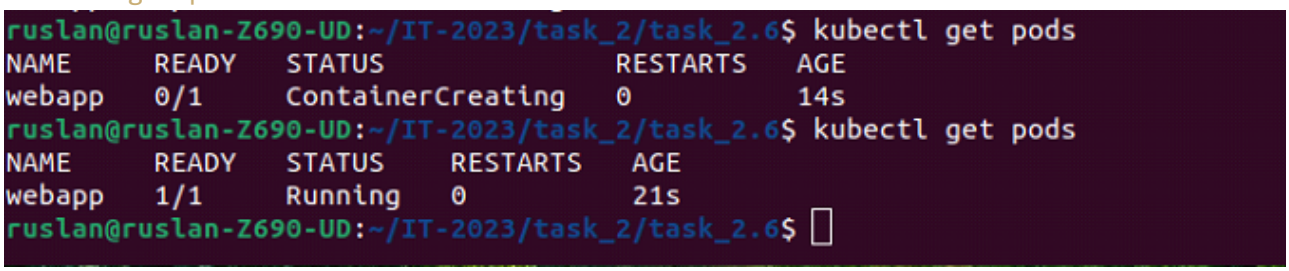
3. Создаем под, используя подготовленный манифест my-pod.yaml:

   ```
   kubectl create -f my-pod.yaml
   ```



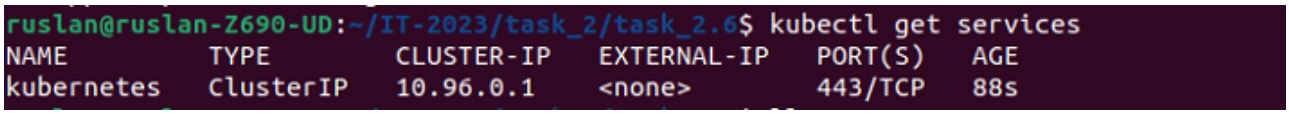
4. Проверяем состояние пода:

   ```
   kubectl get pods
   ```



5. Проверяем состояние сервиса:

   ```
   kubectl get service
   ```



6. Получим более подробную информацию о созданном поде:

   ```
   kubectl describe pod webapp
   ```

```
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ kubectl describe pod webapp
Name:             webapp
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Wed, 08 Nov 2023 23:53:38 +0500
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.244.0.14
IPs:
  IP:  10.244.0.14
Containers:
  webapp:
    Container ID:   docker://6acc2beadbec750e7809f8a4ffd458f342264a45a32d98a1c662e8864b359bdc
    Image:          karthequian/helloworld:latest
    Image ID:       docker-pullable://karthequian/helloworld@sha256:48413fdddeae11e4732896e49b6d82979847955666ed95e4d6e57b433920c9e1
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Wed, 08 Nov 2023 23:53:52 +0500
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-phcd5 (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  kube-api-access-phcd5:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
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  89s   default-scheduler  Successfully assigned default/webapp to minikube
  Normal  Pulling    89s   kubelet            Pulling image "karthequian/helloworld:latest"
  Normal  Pulled     75s   kubelet            Successfully pulled image "karthequian/helloworld:latest" in 13.602070475s (13.602085037s including waiting)
  Normal  Created    75s   kubelet            Created container webapp
```

7. Запустим отладку внутри контейнера пода:

   `kubectl exec -it webapp -- bin/bash`

```
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ kubectl exec -it webapp -- /bin/bash
root@webapp:/# ll
total 76
drwxr-xr-x   1 root root 4096 Nov  8 18:53 ./
drwxr-xr-x   1 root root 4096 Nov  8 18:53 ../
-rwxr-xr-x   1 root root    0 Nov  8 18:53 .dockerenv*
lrwxrwxrwx   1 root root    7 Jan 19  2021 bin -> usr/bin/
drwxr-xr-x   2 root root 4096 Apr 15  2020 boot/
drwxr-xr-x   5 root root  360 Nov  8 18:53 dev/
drwxr-xr-x   1 root root 4096 Nov  8 18:53 etc/
drwxr-xr-x   2 root root 4096 Apr 15  2020 home/
lrwxrwxrwx   1 root root    7 Jan 19  2021 lib -> usr/lib/
lrwxrwxrwx   1 root root    9 Jan 19  2021 lib32 -> usr/lib32/
lrwxrwxrwx   1 root root    9 Jan 19  2021 lib64 -> usr/lib64/
lrwxrwxrwx   1 root root   10 Jan 19  2021 libx32 -> usr/libx32/
drwxr-xr-x   2 root root 4096 Jan 19  2021 media/
drwxr-xr-x   2 root root 4096 Jan 19  2021 mnt/
drwxr-xr-x   2 root root 4096 Jan 19  2021 opt/
dr-xr-xr-x 495 root root    0 Nov  8 18:53 proc/
drwx------   2 root root 4096 Jan 19  2021 root/
drwxr-xr-x   1 root root 4096 Nov  8 18:53 run/
-rwxr-xr-x   1 root root  141 Jan 24  2021 runner.sh*
lrwxrwxrwx   1 root root    8 Jan 19  2021 sbin -> usr/sbin/
drwxr-xr-x   2 root root 4096 Jan 19  2021 srv/
dr-xr-xr-x  13 root root    0 Nov  8 18:53 sys/
drwxrwxrwt   1 root root 4096 Jan 24  2021 tmp/
drwxr-xr-x   1 root root 4096 Jan 19  2021 usr/
drwxr-xr-x   1 root root 4096 Jan 24  2021 var/
drwxr-xr-x   1 root root 4096 Jan 24  2021 www/
root@webapp:/# cat runner.sh
#!/bin/sh

# Replace the hostname in the container
sed -i.bak 's/HOSTNAME/'"$HOSTNAME"'/g' /www/data/index.html

# Startup the cmd
exec "$@"
root@webapp:/#
```

8. Подключимся к среде minikube:

   `minikube ssh`

9. Проверим содержимое нашей веб-страницы. Для этого воспользуем утилитой curl с указанием ip-адреса нашего пода:

```
curl 10.244.0.14
```

```
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ minikube ssh
docker@minikube:~$ curl localhost
curl: (7) Failed to connect to localhost port 80 after 0 ms: Connection refused
docker@minikube:~$ curl 10.244.0.14
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="description" content="A simple docker helloworld example.">
    <meta name="author" content="Karthik Gaekwad">
    <meta name="viewport" content="width=device-width, initial-scale=1">

        <!-- Latest compiled and minified CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/
ous">
<script type="text/javascript">
```

10. Далее выйдем из среды minikube и проверим журналы пода:

```
kubectl logs webapp
```

```
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ kubectl logs webapp
10.244.0.1 - - [08/Nov/2023:19:06:38 +0000] "GET / HTTP/1.1" 200 4363 "-" "curl/7.81.0"
10.244.0.1 - - [08/Nov/2023:19:06:38 +0000] "GET / HTTP/1.1" 200 4363 "-" "curl/7.81.0"
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$
```

11. Остановим и удалим под, после чего убедимся что все прошло успешно:

```
kubectl delete pod webapp
kubectl get pods
```

```
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ kubectl delete pod webapp
pod "webapp" deleted
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ kubectl get pods
No resources found in default namespace.
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$
```

12. Остановим и удалим сервис, и также убедимся в успехе:

```
kubectl delete service kubernetes
kubectl get services
```

```
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ kubectl delete service kubernetes
service "kubernetes" deleted
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ kubectl get services
No resources found in default namespace.
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$
```

13. Напоследок, останавливаем наш мини-кластер:

```
minikube stop
```

```
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$ minikube stop
✋ Узел "minikube" останавливается ...
🛑 Выключается "minikube" через SSH ...
🛑 Остановлено узлов: 1.
ruslan@ruslan-Z690-UD:~/IT-2023/task_2/task_2.6$
```

Выводы

Minikube - отличный инструмент, для того, чтобы познакомиться с Kubernetes и применить его возможности на локальной машине. С помощью minikube разработчикам может быть удобно разрабатывать и тестировать приложения, которые в будущем будут запущены на полноценном кластере Kubernetes. С помощью средств Kubernetes в minikube можно относительно легко и быстро задеплоить необходимое веб-приложение.

Также данный инструмент удобен для использования DevOps инженерами, так как позволяет проводить различные эксперименты, тестировать различные конфигурации и оптимизировать развертку приложений.

Сам же Kubernetes позволяет управлять всем проектом единообразно с помощью файлов конфигурации и имеет подробные руководства и документацию. В случае, когда у вас действительно много контейнизированных приложений, то управлять ими по отдельности довольно трудоемкая задача - и тогда кубер будет одним из хороших решений.

По итогам выполнения данного задания, можно сделать вывод о том, что k8s позволяет нативно делать довольно много полезных вещей, например вести логи отдельных подов / контейнеров, следить за их работой и состоянием, гибко управлять конфигурацией, запуском и остановкой.