

Департамент образования и науки города Москвы
Государственное автономное образовательное учреждение высшего
образования города Москвы
«Московский городской педагогический университет»
Институт цифрового образования
Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Интеграция и развертывание программного обеспечения с помощью
контейнеров

Работа на лекции 29.03.2025

Тема:

«Основы работы с Kubernetes»

Выполнил(а): Морозова Валерия АДЭУ-211

Преподаватель: Босенко Т.М.

Москва

2025

Цель работы: получить практические навыки работы с кластером Kubernetes, включая развертывание базовых компонентов, настройку мониторинга и работу с service mesh.

Задачи:

Изучить основные концепции Kubernetes через практические вопросы.

Научиться анализировать и применять манифесты Kubernetes.

Групповые задания

Задание 1. Теоретические основы Kubernetes.

Ответить на 3 случайных вопроса из репозитория:

1. Можете ли вы развернуть несколько планировщиков?

Да, это возможно. Вы можете запустить другой модуль с помощью команды, похожей на:

```
spec:
  containers:
  - command:
    - kube-scheduler
    - --address=127.0.0.1
    - --leader-elect=true
    - --scheduler-name=some-custom-scheduler
  ...
```

2. Если у вас несколько планировщиков, как узнать, какой планировщик использовался для конкретного модуля?

Запустив `kubectl get events`, вы можете увидеть, какой планировщик был использован.

3. Вы хотите запустить новый модуль и хотите, чтобы он запускался по расписанию, заданному пользователем. Как это сделать?

Добавьте следующее к спецификации модуля:

```
spec:
  schedulerName: some-custom-scheduler
```

4. Что такое кластер Kubernetes?

Определение Red Hat: «Кластер Kubernetes — это набор узлов для запуска контейнерных приложений. Если вы используете Kubernetes, вы

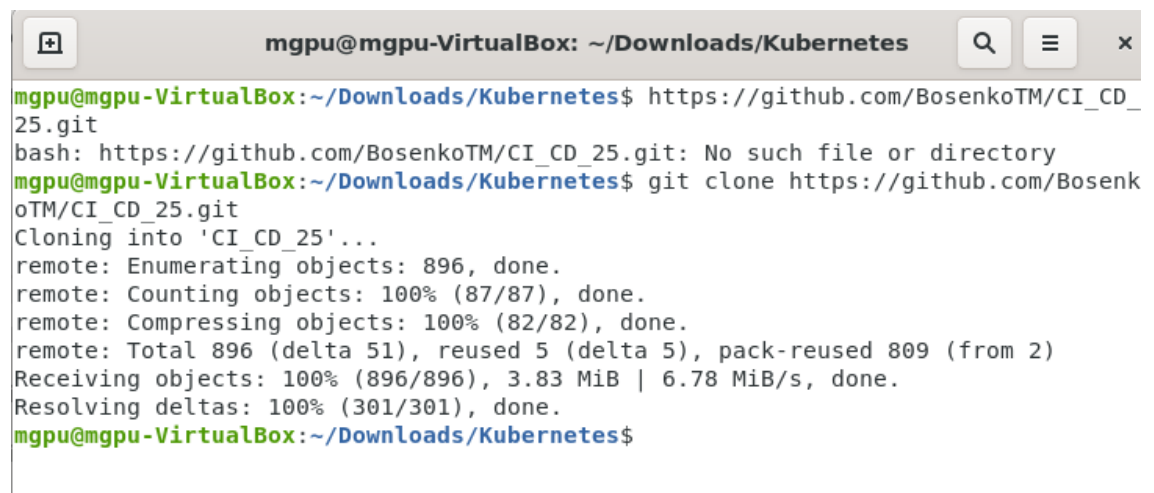
используете кластер. Как минимум, кластер содержит рабочий узел и главный узел».

5. Что такое kubectl?

Kubectl — это инструмент командной строки Kubernetes, который позволяет выполнять команды в кластерах Kubernetes. Например, с помощью kubectl можно разворачивать приложения, проверять ресурсы кластера и управлять ими, а также просматривать журналы.

6. Какая команда будет перечислять все типы объектов в кластере?

```
kubectl api-resources
```



The screenshot shows a terminal window titled 'mgpu@mgpu-VirtualBox: ~/Downloads/Kubernetes'. The user enters the command 'https://github.com/BosenkoTM/CI_CD_25.git', which results in an error: 'bash: https://github.com/BosenkoTM/CI_CD_25.git: No such file or directory'. Then, the user enters 'git clone https://github.com/BosenkoTM/CI_CD_25.git', which successfully clones the repository into a directory named 'CI_CD_25'. The terminal output shows the progress of cloning, including enumerating, counting, and compressing objects, and receiving the data.

Рисунок 1. Клонирование каталога

```
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 119M 100 119M 0 0 12.6M 0 0:00:09 0:00:09 --:--:-- 11.8M
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
[sudo] password for mgpu:
○ mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ █
```

Рисунок 2. Установка minikube

```
○ mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ sudo usermod -aG docker $USER
&& newgrp docker
```

Рисунок 3. Добавление пользователя в группу Docker

```
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ sudo snap install kubectl --c
lassic
kubectl 1.32.3 from Canonical✓ installed
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ █
```

Рисунок 4. Установка kubectl

```

mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ minikube start --memory=2048mb --driver=docker
minikube v1.35.0 on Ubuntu 20.04 (vbox/amd64)
Using the docker driver based on user configuration
Using Docker driver with root privileges
Starting "minikube" primary control-plane node in "minikube" cluster
Pulling base image v0.0.46 ...
Downloading Kubernetes v1.32.0 preload ...
> preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 6.48 Mi
> gcr.io/k8s-minikube/kicbase...: 500.31 MiB / 500.31 MiB 100.00% 7.08 Mi
Creating docker container (CPUs=2, Memory=2048MB) ...
Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
  ■ Generating certificates and keys ...
  ■ Booting up control plane ...
  ■ Configuring RBAC rules ...
Configuring bridge CNI (Container Networking Interface) ...
Verifying Kubernetes components...
  ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
Enabled addons: storage-provisioner, default-storageclass
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$

```

Рисунок 5. Запуск

```

mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ docker build -t fastapi-app:local .
[+] Building 102.6s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 239B
=> [internal] load metadata for docker.io/library/python:3.10
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/5] FROM docker.io/library/python:3.10@sha256:8ad0e578e1b733f2a496b41f179175679374191a9c7ab8c63156446094a9cda8
=> => resolve docker.io/library/python:3.10@sha256:8ad0e578e1b733f2a496b41f179175679374191a9c7ab8c63156446094a9cda8
=> => sha256:7cd785773db44407e20a679ce5439222e505475eed5b99f1910eb2cda51729ab 48.47MB / 48.47MB
=> => sha256:255774e0027b72d2327719e78dbad5ad8c9cf446d055e45be7fc149418470bae 64.40MB / 64.40MB
=> => sha256:b0fc5e29abb0921de96874560ac409bb8e131545fa8623a27dd3791decf8ceb 6.18kB / 6.18kB
=> => sha256:52c63d169c27d32435ff634ea772d5ca52c9d0793bb796e97b0977c582642727 2.33kB / 2.33kB
=> => sha256:091eb8249475f42de217265c501e0186f0a3ea7490ef7f51458c30db91fb3cac 24.01MB / 24.01MB
=> => sha256:8ad0e578e1b733f2a496b41f179175679374191a9c7ab8c63156446094a9cda8 9.08kB / 9.08kB
=> => sha256:353e14e5cc47664fba714a7da288001d90427c705494847ac773f5cc08199451 211.35MB / 211.35MB
=> => extracting sha256:7cd785773db44407e20a679ce5439222e505475eed5b99f1910eb2cda51729ab
=> => sha256:0c64566c7562a4e1405a59f7b95f25b2b74a9a630b8b4b5916d3829a81e90ab4 6.16MB / 6.16MB
=> => sha256:57161121b343e07415ad5fdd4d3b635176622126bab8ff18e653439c9619f29 21.38MB / 21.38MB
=> => sha256:a90d73ac0e516c2cd69b099f3b5f957c2815844e088d741d737c95e7111d249c 249B / 249B
=> => extracting sha256:091eb8249475f42de217265c501e0186f0a3ea7490ef7f51458c30db91fb3cac
=> => extracting sha256:255774e0027b72d2327719e78dbad5ad8c9cf446d055e45be7fc149418470bae
=> => extracting sha256:353e14e5cc47664fba714a7da288001d90427c705494847ac773f5cc08199451
=> => extracting sha256:0c64566c7562a4e1405a59f7b95f25b2b74a9a630b8b4b5916d3829a81e90ab4
=> => extracting sha256:57161121b343e07415ad5fdd4d3b635176622126bab8ff18e653439c9619f29

```

Рисунок 6. Настройка окружения и билдинг локального образа, загружаем его в Minikube

```

mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl create -f configmap.yml
configmap/fastapi-config created
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl create -f secret.yml
secret/fastapi-secret created
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl create -f fastapi-deployment-and-service.yml
deployment.apps/fastapi-deployment created
service/fastapi-service created
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl create -f redis-deployment-and-service.yml
deployment.apps/redis-deployment created
service/redis-service created
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$

```

Рисунок 7. Создание ресурсов в Kubernetes кластере

```

mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
fastapi-deployment-cf4dc69bc-5qdv5 1/1     Running   1 (3m58s ago)  6m29s
fastapi-deployment-cf4dc69bc-lw2qc 1/1     Running   1 (3m58s ago)  6m29s
redis-deployment-748ffbc5f5-kzqb6 1/1     Running   0           3m22s
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$

```

Рисунок 8. Состояние Pod в кластере Kubernetes

```

mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ minikube service fastapi-service --url
http://192.168.49.2:30001

```

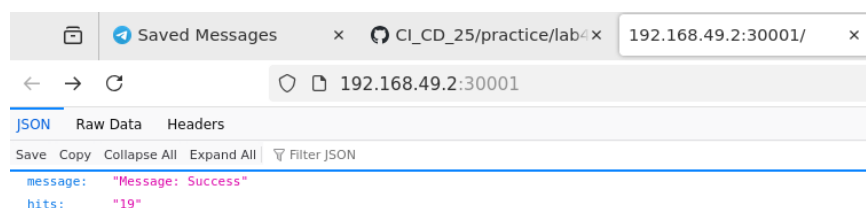


Рисунок 9. OpenAPI

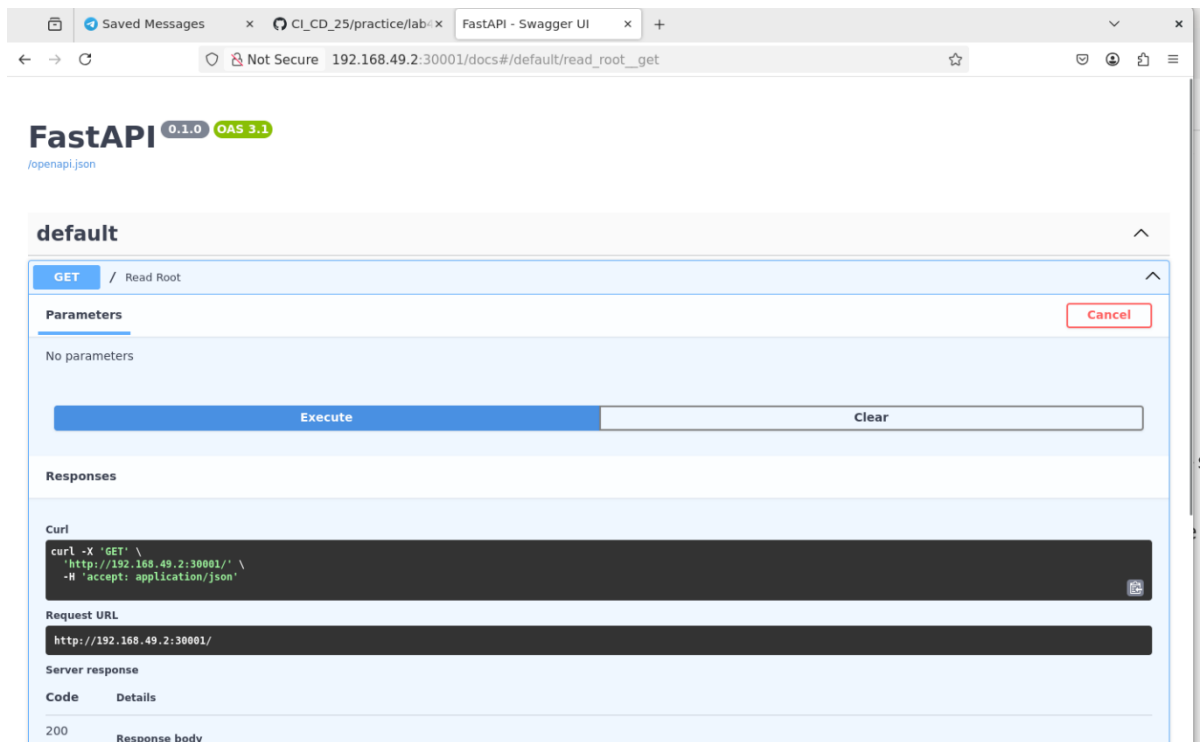


Рисунок 10. Проверка работоспособности FastAPI

```
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl describe pod fastapi-deployment-cf4dc69bc-lw2qc
Name: fastapi-deployment-cf4dc69bc-lw2qc
Namespace: default
Priority: 0
Service Account: default
Node: minikube/192.168.49.2
Start Time: Sat, 29 Mar 2025 15:26:05 +0300
Labels: app=fastapi
        pod-template-hash=cf4dc69bc
Annotations: <none>
Status: Running
IP: 10.244.0.4
IPs:
  IP: 10.244.0.4
Controlled By: ReplicaSet/fastapi-deployment-cf4dc69bc
Init Containers:
  init-myservice:
    Container ID: docker://cc790353b87347a4a22a57a61e356ba4f45512eebb757a51d58610c7c32dcad7
    Image: busybox
    Image ID: docker-pullable://busybox@sha256:37f7b378a29ceb4c551b1b5582e27747b855bbfaa73fa11914fe0df028dc581f
    Port: <none>
    Host Port: <none>
    Command:
      sh
```

Рисунок 11. Просмотр второго пода FastAPI

```
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl get services
NAME          TYPE        CLUSTER-IP    EXTERNAL-IP  PORT(S)          AGE
fastapi-service NodePort    10.109.184.213 <none>        80:30001/TCP     13m
kubernetes    ClusterIP   10.96.0.1      <none>        443/TCP           21m
redis-service ClusterIP   10.103.116.111 <none>        6379/TCP          10m
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl config view
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /home/mgpu/.minikube/ca.crt
    extensions:
    - extension:
        last-update: Sat, 29 Mar 2025 15:18:00 MSK
        provider: minikube.sigs.k8s.io
        version: v1.35.0
      name: cluster info
    server: https://192.168.49.2:8443
  name: minikube
contexts:
- context:
    cluster: minikube
    extensions:
    - extension:
        last-update: Sat, 29 Mar 2025 15:18:00 MSK
        provider: minikube.sigs.k8s.io
        version: v1.35.0
```

Рисунок 12. Список всех сервисов, запущенных в кластере

Индивидуальное задание

Задание 1

```
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl get po -n kube-system
NAME                                READY   STATUS    RESTARTS   AGE
coredns-668d6bf9bc-gzcbd           1/1     Running   0           29m
etcd-minikube                      1/1     Running   0           30m
kube-apiserver-minikube            1/1     Running   0           30m
kube-controller-manager-minikube   1/1     Running   0           30m
kube-proxy-z2x5x                   1/1     Running   0           29m
kube-scheduler-minikube            1/1     Running   0           30m
storage-provisioner                1/1     Running   2 (24m ago) 30m
mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$
```

Рисунок 13. Проверка работы системных контейнеров

Задание 2

```
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl get nodes
NAME          STATUS    ROLES          AGE   VERSION
minikube      Ready     control-plane  115m  v1.32.0
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.49.2:8443
CoreDNS is running at https://192.168.49.2:8443/api/v1/namespaces/kube-system/services/kube-c
/proxy
```

Рисунок 14. Проверка доступности узлов и подключения к кластеру

```
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl apply -f wordpress-
ment-and-service.yml
deployment.apps/wordpress-deployment created
service/wordpress-service created
○ mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ Morozova V
```

Рисунок 15. Развертывание WordPress

```
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl apply -
f wordpress-deployment-and-service.yml
deployment.apps/wordpress-deployment configured
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl expose
deployment wordpress-deployment --type=LoadBalancer --port=80
service/wordpress-deployment exposed
```

Рисунок 16. Создание сервиса для доступа

При условии нескольких попыток pod не хотел запускаться, поэтому было решено дополнительно развернуть MySQL.

```
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl apply -f mysql-
deployment.yaml
secret/mysql-secret created
persistentvolumeclaim/mysql-pvc created
deployment.apps/mysql created
service/mysql-service created
```

Рисунок 17. Применение конфигурации

К сожалению, pod MySQL тоже не хотел запускаться

```
● mgpu@mgpu-VirtualBox:~/Downloads/Kubernetes/CI_CD_25/practice/lab4_1$ kubectl logs -f deploy
ent/mysql
Error from server (BadRequest): container "mysql" in pod "mysql-5b8bbfdf4b-xvgm9" is waiting
to start: trying and failing to pull image
```

Рисунок 18. Проверка логов