Урок 5. Сетевые абстракции Kubernetes

Практическое задание

Студент Виталий Зайцев, DE\_622

Contents

[Домашнее задание 2](#_Toc84072994)

[Реализация 3](#_Toc84072995)

[Проверка работоспособности/готовности kubernetes 3](#_Toc84072996)

[Deployment сервера базы данных с образом Postgresql (на основании deployment.yml из 4-го урока) 4](#_Toc84072997)

[deployment\_postgresql.yml 4](#_Toc84072998)

[Deployment сервиса с портом 5432 5](#_Toc84072999)

[service\_port5432.yml 5](#_Toc84073000)

[Создание секрета с двумя переменными 6](#_Toc84073001)

[secret.yml 7](#_Toc84073002)

[Deployment с образом redmine 8](#_Toc84073003)

[deployment\_redmine.yml 8](#_Toc84073004)

[Deployment сервиса с портом 3000 9](#_Toc84073005)

[service\_port3000.yml 9](#_Toc84073006)

[Развёртывание секретов, баз, сервисов 10](#_Toc84073007)

[Cоздание/проверка наличия отдельного namespace my-postgres и установка контекста в этот namespace 10](#_Toc84073008)

[Применение манифестов 11](#_Toc84073009)

[Проверка работоспособности 12](#_Toc84073010)

[НЕУСПЕХ! Подключение к сервису с помощью браузера и создание скриншота 13](#_Toc84073011)

[НЕУСПЕХ! Создание тестового под и попытка подключения к redmine-svc по 80-му порту 13](#_Toc84073012)

[НЕУСПЕХ! Попытка развёртывания ingress 13](#_Toc84073013)

[ingress\_networking.k8s.io.yml 13](#_Toc84073014)

[НЕУСПЕХ! Применение ingress\_networking.k8s.io.yml 14](#_Toc84073015)

# Домашнее задание

Разверните в кластере сервер базы данных Postgresql. Из предыдущего задания.

Добавьте к нему service c портом 5432 и именем database.

В этом же неймспэйсе создайте deployment с образом redmine:4.1.1

Для запуска нужно передать переменные окружения:

REDMINE\_DB\_POSTGRES = database

REDMINE\_DB\_USERNAME = <postgres\_user>

REDMINE\_DB\_PASSWORD = <postgres\_password> (значение должно браться из секрета) REDMINE\_DB\_DATABASE = <postgres\_database> REDMINE\_SECRET\_KEY\_BASE = supersecretkey (значение должно браться из секрета)

Обратите внимание что имя пользователя, пароль и база данных должны соответствовать значениям которые указаны в переменных окружения деплоймента postgresql

В деплойменте приложения должен быть описан порт 3000

Создайте serivce для приложения с портом 3000

Подключитесь с помощью браузера к сервису, сделайте скриншот.

# Реализация

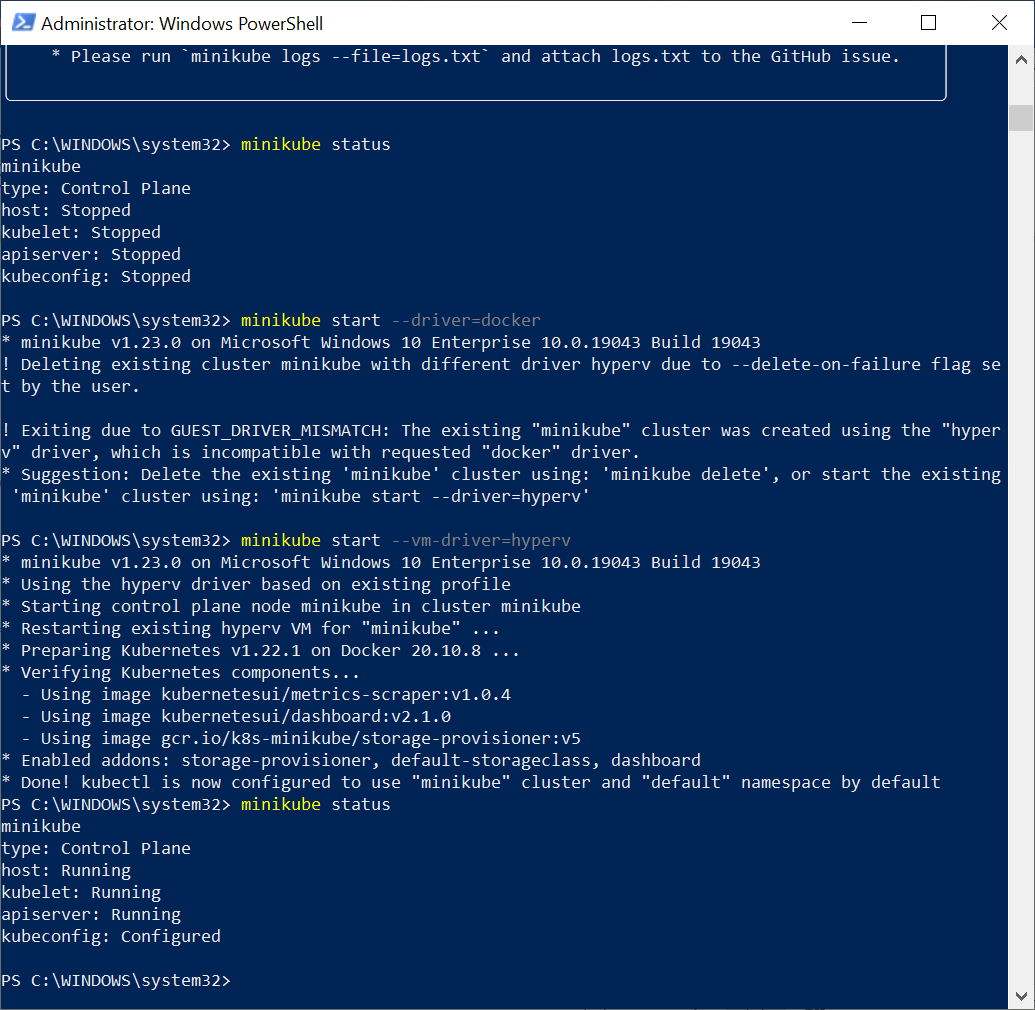
## Проверка работоспособности/готовности kubernetes

Run PowerShell as administrator

minikube status

minikube start --vm-driver=hyperv

minikube status



## Deployment сервера базы данных с образом Postgresql (на основании deployment.yml из 4-го урока)

### deployment\_postgresql.yml

apiVersion: apps/v1

kind: Deployment

metadata:

name: deployment-postgres

spec:

replicas: 1

strategy:

type: RollingUpdate

rollingUpdate:

maxSurge: 0

maxUnavailable: 1

selector:

matchLabels:

app: my-db

template:

metadata:

labels:

app: my-db

spec:

containers:

- image: postgres:10.13

name: postgresql

ports:

- containerPort: 5432

name: postgresql

protocol: TCP

env:

- name: POSTGRES\_DB

value: testdatabase

- name: POSTGRES\_USER

value: testuser

- name: PGDATA

value: /var/lib/postgresql/data/pgdata

- name: POSTGRES\_PASSWORD

valueFrom:

secretKeyRef:

name: my-secret

key: db\_password

## Deployment сервиса с портом 5432

### service\_port5432.yml

apiVersion: v1

kind: Service

metadata:

name: database-svc

spec:

ports:

- port: 5432

targetPort: 5432

selector:

app: my-db

type: ClusterIP

## Создание секрета с двумя переменными

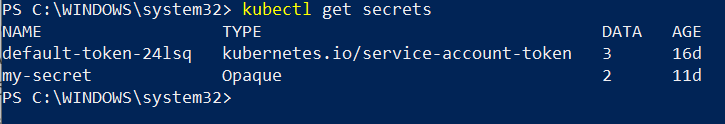
*…*

*REDMINE\_DB\_PASSWORD = <postgres\_password> (значение должно браться из секрета) REDMINE\_DB\_DATABASE = <postgres\_database> REDMINE\_SECRET\_KEY\_BASE = supersecretkey (значение должно браться из секрета)*

*…*

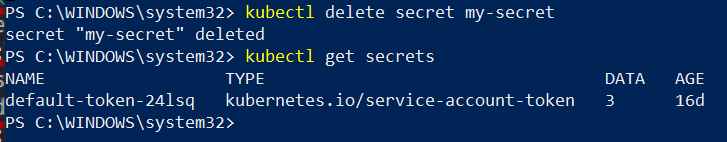
<https://kubernetes.io/docs/tasks/configmap-secret/managing-secret-using-kubectl/>

kubectl get secrets



Удаляем ранее созданный секрет my-secret (с одной переменной) и создадим новый секрет, уже с двумя переменными

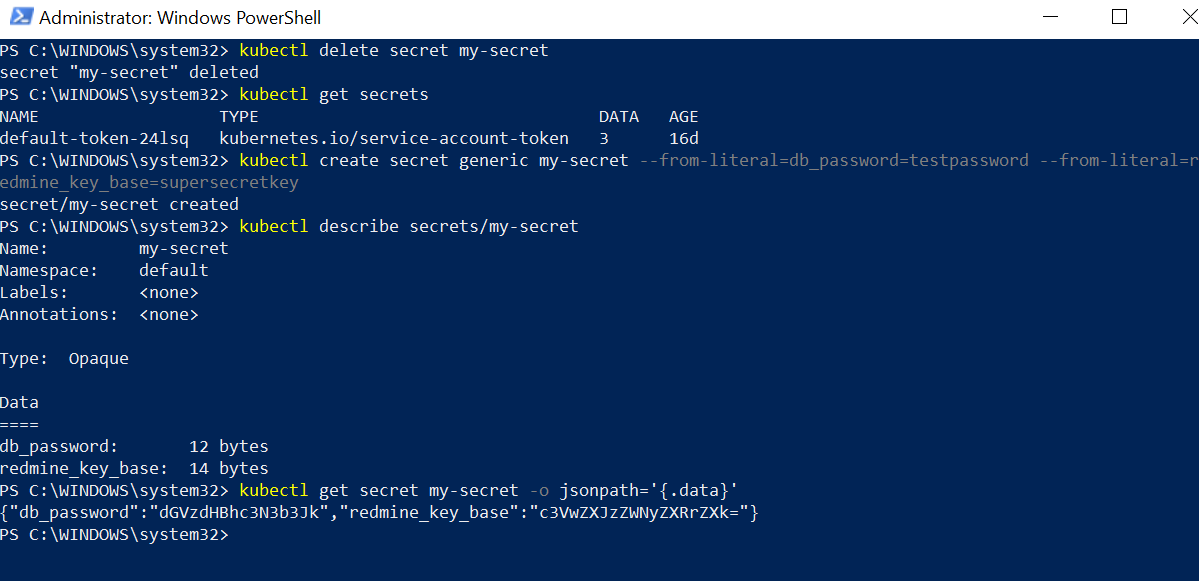
kubectl delete secret my-secret



kubectl create secret generic my-secret --from-literal=db\_password=testpassword --from-literal=redmine\_key\_base=supersecretkey

kubectl describe secrets/my-secret

kubectl get secret my-secret -o jsonpath='{.data}'



{"db\_password":"dGVzdHBhc3N3b3Jk","redmine\_key\_base":"c3VwZXJzZWNyZXRrZXk="}

### secret.yml

apiVersion: v1

kind: Secret

metadata:

name: my-secret

type: Opaque

data:

db\_password: dGVzdHBhc3N3b3Jk

redmine\_key\_base: c3VwZXJzZWNyZXRrZXk=

## Deployment с образом redmine

### deployment\_redmine.yml

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: deployment-redmine

spec:

replicas: 1

strategy:

type: RollingUpdate

rollingUpdate:

maxSurge: 0

maxUnavailable: 1

selector:

matchLabels:

app: my-redmine

template:

metadata:

labels:

app: my-redmine

spec:

containers:

- image: redmine:4.1.1

name: redmine

env:

- name: REDMINE\_DB\_POSTGRES

value: database

- name: REDMINE\_SECRET\_KEY\_BASE

valueFrom:

secretKeyRef:

name: my-secret

key: redmine\_key\_base

# база данных

- name: REDMINE\_DB\_DATABASE

value: testdatabase

# пользователь базы данных

- name: REDMINE\_DB\_USERNAME

value: testuser

# пароль пользователя

- name: REDMINE\_DB\_PASSWORD

valueFrom:

secretKeyRef:

name: my-secret

key: pass

ports:

- containerPort: 3000

protocol: TCP

## Deployment сервиса с портом 3000

### service\_port3000.yml

apiVersion: v1

kind: Service

metadata:

name: redmine-svc

spec:

ports:

- port: 80

targetPort: 3000

selector:

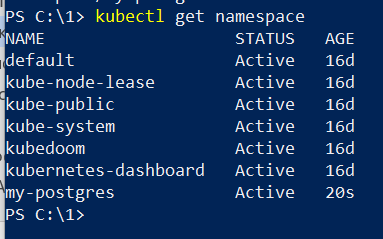
app: my-redmine

type: ClusterIP

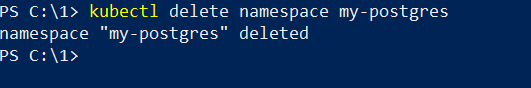
## Развёртывание секретов, баз, сервисов

### Cоздание/проверка наличия отдельного namespace my-postgres и установка контекста в этот namespace

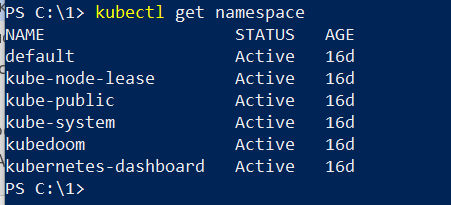
kubectl get namespace



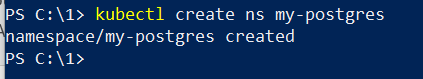
kubectl delete namespace my-postgres



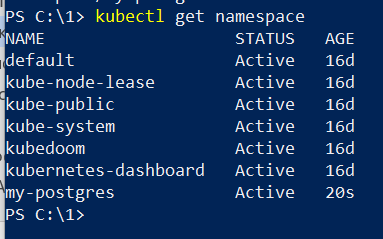
kubectl get namespace



kubectl create ns my-postgres

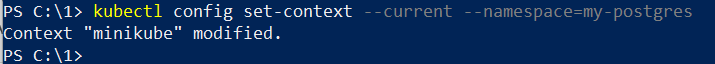


kubectl get namespace



Установка контекста в namespace

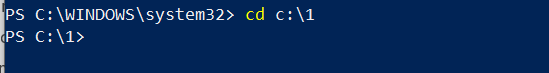
kubectl config set-context --current --namespace=my-postgres



### Применение манифестов

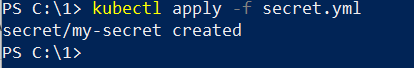
Переходим в папку размещения yml файлов

cd C:\1

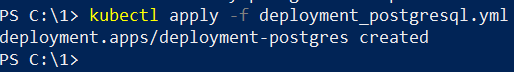


Применяем манифесты (начиная с secret)[[1]](#footnote-1)

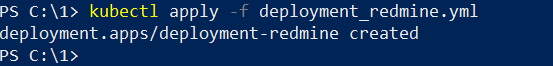
kubectl apply -f secret.yml



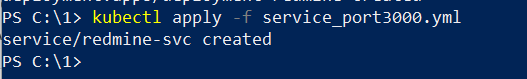
kubectl apply -f deployment\_postgresql.yml



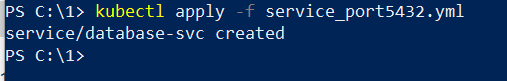
kubectl apply -f deployment\_redmine.yml



kubectl apply -f service\_port3000.yml

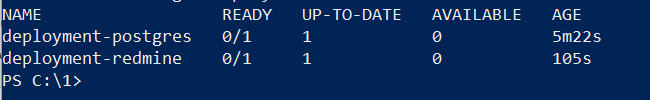


kubectl apply -f service\_port5432.yml



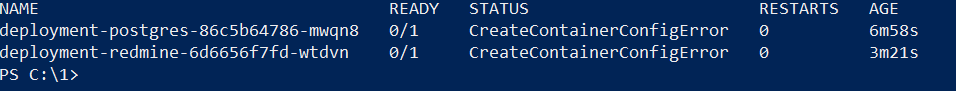
Посмотрим применённые deployments

kubectl get deploy

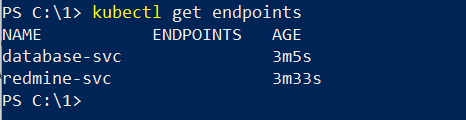


### Проверка работоспособности

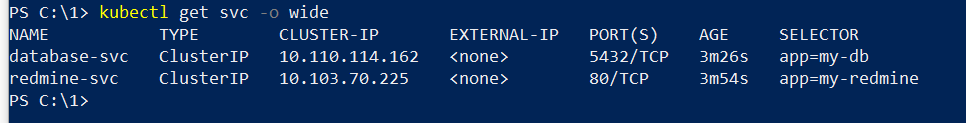
kubectl get pods



kubectl get endpoints



kubectl get svc -o wide



NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE SELECTOR

database-svc ClusterIP 10.110.114.162 <none> 5432/TCP 3m26s app=my-db

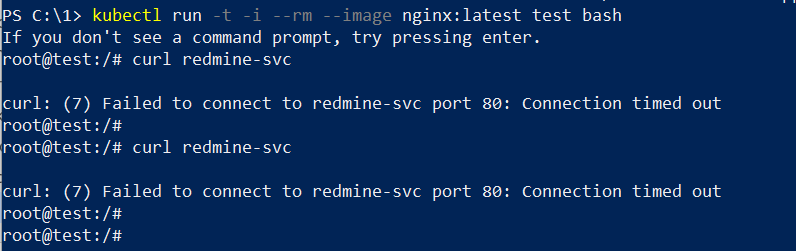
redmine-svc ClusterIP 10.103.70.225 <none> 80/TCP 3m54s app=my-redmine

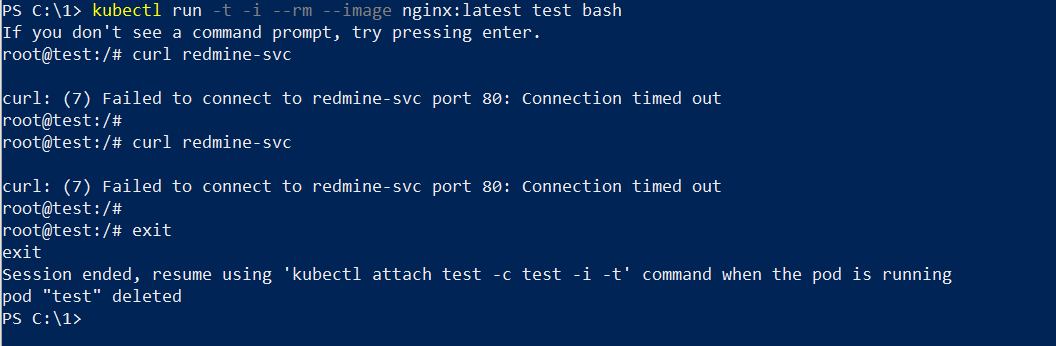
# НЕУСПЕХ! Подключение к сервису с помощью браузера и создание скриншота

## НЕУСПЕХ! Создание тестового под и попытка подключения к redmine-svc по 80-му порту

kubectl run -t -i --rm --image nginx:latest test bash

К сожалению, неуспешна (возможно, причина в выполнении добашней работы на рабочем компьюетере, находящемся под действием организационных политик безопасности? Нет идей)





## НЕУСПЕХ! Попытка развёртывания ingress

### ingress\_networking.k8s.io.yml[[2]](#footnote-2)

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: my-ingress

spec:

rules:

- host: my-host.local

http:

paths:

- path: /testpath

pathType: Prefix

backend:

service:

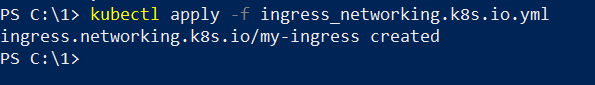
name: redmine-svc

port:

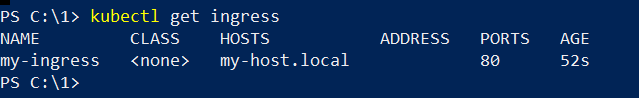
number: 80

## НЕУСПЕХ! Применение ingress\_networking.k8s.io.yml

kubectl apply -f ingress\_networking.k8s.io.yml

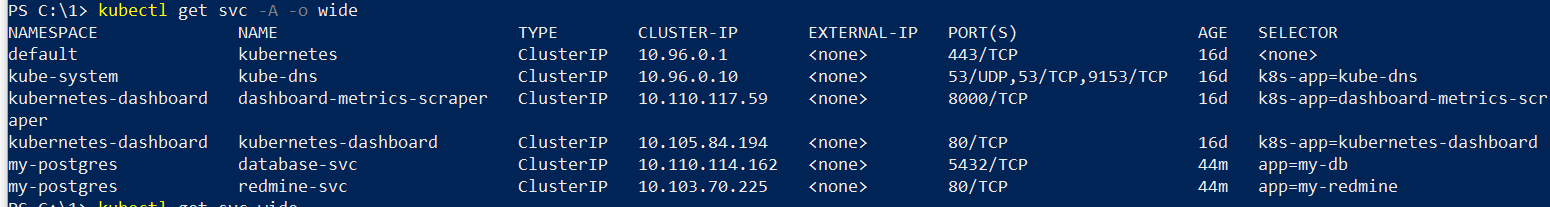


kubectl get ingress



Попробовал найти ingress controller (чтобы узнать внешний ip и подключиться) – не нашёл ingress controller. Выполнение практического задания остановлено, идей нет.

kubectl get svc -A -o wide



1. Удаление неудачных deployment’ов <https://www.ibm.com/docs/en/cdfsp/7.6.1.x?topic=containers-stop-existing-deployments>

   1. Get the running deployments by executing the below command in Master node.

   kubectl get deploy

   1. Delete all the deployments, run below command:

   kubectl delete deploy <deployment name> [↑](#footnote-ref-1)
2. <https://kubernetes.io/docs/concepts/services-networking/ingress/>

   <https://www.ibm.com/docs/ru/control-desk/7.6.1.x?topic=kubernetes-installing-nginx-ingress-controller-in-cluster> [↑](#footnote-ref-2)