Практическая работа 6

«Развертывание и масштабирование веб-приложения с использованием Kubernetes и minikube»

Цель:

Углубить понимание основных концепций Kubernetes (далее K8S), таких как: Pod, Service, Ingress, а также получить практический опыт работы с K8S, осуществляя развертывание, масштабирование и управление веб-приложениями в локальном кластере с использованием инструмента minikube.

Описание задания:

В текущей работе исполнителю для достижения поставленных целей предлагается выполнить развертывание простого веб-приложения, а затем провести его масштабирование, используя возможности кластера K8S.

Немного тории:

minikube - это инструмент для локального развертывания кластера K8S на одном компьютере. Он позволяет разработчикам быстро прототипировать и тестировать свои приложения перед их развертыванием в реальных или облачных средах K8S создавая миниатюрные кластеры на своей рабочей станции, что делает его идеальным инструментом для тестирования и изучения K8S без необходимости настройки полноценного кластера в облаке или на физическом оборудовании. В целом, он упрощает работу с K8S, предоставляя легкую, изолированную и безопасную среду для разработки, тестирования и изучения платформы контейнеризации и оркестрации.

Более детально о minikube можно узнать из официального руководства по ссылке ниже: https://kubernetes.io/docs/tutorials/hello-minikube/

Шаги:

1. **Установка и настройка Minikube и сопутствующих инструментов.**

   В качестве рабочей среды рекомендуется использовать ранее развернутый WSL, это облегчит настройку и конфигурирование рабочего пространства. Перед запуском установки нам понадобятся установить дополнительные зависимости. Для этого введите следующую команду в своём терминале:

```
sudo apt-get install curl wget apt-transport-https
```

Кроме этого, нам понадобиться Docker. Для установки Docker, если ранее Вы еще не устанавливали его, выполните установку следуя данной инструкции, иначе пропустите этот шаг: https://docs.docker.com/engine/install/ubuntu/

Теперь необходимо запустить службу Docker:

```
sudo service docker start
```

Убедимся, что Docker установлен:

```
docker –version
```

И сервис работает:

```
docker ps
```

Наконец-то пришло время установки и самого minikube. Итак, для установки достаточно перейти на страницу с последним релизом и выполнить предложенные инструкции:

```
curl -LO
https://storage.googleapis.com/minikube/releases/latest/min
ikube_latest_amd64.deb

sudo dpkg -i minikube_latest_amd64.deb
```

Проверим версию Minikube и корректность установки:

```
minikube version
```

Если команда вернула непустой результат — можно продолжать.

Далее нам понадобится установить kubectl — утилиту для управления приложениями в Kubernetes:

```
curl -LO https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubec
tl
```

Выполним валидацию бинарного файла с помощью чек суммы:

```
curl -LO https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubec
tl.sha256

echo "$(cat kubectl.sha256)  kubectl" | sha256sum –check
```

Валидация выполнена успешно если вы получили такое сообщение:

```
kubectl: OK
```

Выполните инсталляцию kubectl:

```
sudo install -o root -g root -m 0755 kubectl
/usr/local/bin/kubectl
```

Убедитесь, что установлена последняя версия:

```
kubectl version
```

При возникновении ошибки во время развертывания kubectl можно обратится к официальной инструкции за помощью:
https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/

На текущем этапе мы установили все необходимые пакеты и зависимости, теперь можно запустить инструмент:

```
minikube start
```

Вы можете увидеть информацию о кластере Kubernetes используя команду:

```
kubectl cluster-info
```

и список запущенных в кластере node:

```
kubectl get nodes
```

а с помощью следующих команд вы можете посмотреть на актуальный список pod и deployments:

```
kubectl get pods
kubectl get deployments
```

Для остановки инструмента оркестрации используйте команду:

```
minikube stop
```

Если в процессе установки и тестирования у вас возникли каки-либо сложности, можно обратиться к инструкции по ссылке ниже для более тщательной проработки шагов по установки необходимых компонент:
https://kubernetes.io/ru/docs/tasks/tools/install-minikube/

2. **Создание Docker образа веб-приложения.**

Для написания простого веб-приложения создайте новую ветку в вашем репозитории и соответствующий каталог с лабораторной работой. Перейдите в этот каталог и создайте новый файл с именем index.js со следующим содержимым:

```
var http = require('http');
var handleRequest = function(request, response) {
  response.writeHead(200);
  response.end('Hello, World!');
};
var helloServer = http.createServer(handleRequest);
helloServer.listen(8080);
```

Далее, нам необходимо создать Dockerfile для сборки исходного веб-приложения, выводящего "Hello, World!":

```
# Используем образ с поддержкой node 4.4
FROM node:4.4
EXPOSE 8080
COPY index.js .
CMD node index.js
```

3. **Создание манифеста для развертывания веб-приложения.**

В корне проекта создайте манифест и сохраните этого в файл с расширением YAML, например, deployment.yaml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-node
spec:
```

```
  replicas: 1
  selector:
    matchLabels:
      app: hello-node
  template:
    metadata:
      labels:
        app: hello-node
    spec:
      containers:
      - name: hello-node
        image: hello-node:v1
        ports:
        - containerPort: 8080
```

Этот пример манифеста создаст деплоймент с именем "hello-node", который будет запускать одну реплику контейнера с образом "hello-node:v1", и откроет порт 8080 внутри контейнера.

4. **Сборка Docker Image**

Для конфигурации поиска реестра внутри виртуальной машины необходимо перед запуском сборки контейнера выполнить команду:

```
eval $(minikube docker-env)
```

Теперь можно приступить непосредственно к сборке образа. Это займет некоторое время, т.к. для разрешения зависимостей будут подгружены образы из Docker Hub, такие как node 4.4. По завершении мы получим новый Docker-образ, готовый к деплою:

```
docker build -t hello-node:v1 .
```

Примечание: завершающая точка . в конце команды говорит Docker собрать образ из текущей директории.
Проверить собранный образ можно с помощью команды:

```
docker images
```

Теперь можно переходить к следующему шагу.

5. **Развертывание веб-приложения в кластере.**

   С помощью команды:

   ```
   kubectl apply -f deployment.yaml
   ```

   Выполните развертывание исходного приложения. Проверьте доступность вашего приложения.

6. **Дополнительное задание.**

   Создайте Ingress ресурс для обеспечения доступа к своему веб-приложению извне кластера.

7. **Масштабирование веб-приложения.**

   Настройте горизонтальное масштабирование искомого веб-приложения, изменяя количество реплик.

8. **Сохранение отчетов о проделанной работе.**

   Создайте директорию reports и разместите в ней все необходимые требуемые файлы и скриншоты подтверждающие выполнения работы. Зафиксируйте изменения с использованием `git add` и `git commit`. Отправьте изменения на GitLab с использованием `git push`.

9. **Создание Pull Request.**

   Создайте Pull Request на слияние вашей ветки в main(master) ветку. В запросе укажите пул выполненных работ по заданию. В качестве Rewiewer укажите преподавателя.
   Задание считается выполненным, если преподаватель выполнил слияние вашей ветки в основную, иначе смотрите комментарий в вашем запросе с описанием недочетов, исправляйте ошибки и снова выставляйте работу на проверку.

Вопросы и поддержка:

Если у вас возникают трудности или у вас есть вопросы, не стесняйтесь обращаться за помощью к преподавателю или на форум группы.

Удачи в выполнении задания!