K8S 04: Kubernetes. Basics. Kubectl

Описание:

kubectl — это консольная утилита, предназначенная для взаимодействия с кластером Kubernetes. Более подробное описание возможностей kubectl вы можете найти в документации kubectl.
Основные команды:

- get — получает ресурс из API-сервера, по умолчанию — в сокращенном формате, -o задает формат;
- describe — описывает ресурс в выбранном формате;
- create — создает ресурс из файла или с указанием типа и параметров;
- delete — удаляет ресурс по выбору или его имени;
- edit — открывает редактор и загружает в него файл описания ресурса, при сохранении отправляет измененный ресурс API-серверу;
- explain — показывает документацию по ресурсу;
- patch — вносит изменения в поля ресурса, используя стратегию слияния патча;
- proxy — запускает локальный прокси-сервер до API-сервера Kubernetes;
- replace — заменяет ресурс новой версией;
- logs — выдает логи ресурса;
- expose — создает сервис для pod, rs, rc;
- port-forward — прокидывает локальный порт на порт внутри контейнера pod;
- run — запускает pod из указанного образа (image);
- scale — изменяет spec.replicas;
- rolling-update — выполняет обновление, клиентом будет kubectl на хосте.

kubectl поддерживает формат шаблонов golang templates.
Вот некоторые примеры, как могут выглядеть такие запросы:
Посмотреть образы контейнеров, запущенных в кластере.

```
kubectl get pods -n test -o go-template --template='{{range .items}}{{range
.spec.containers}}{{printf "%s\n" .image}}{{end}}{{end}}'
```

Вывести аннотации app.kubernetes.io/instance для pod, у которых количество рестартов первого контейнера более 100.

```
kubectl get pods --all-namespaces -o go-template --template='{{range .items}}{{if gt (index
.status.containerStatuses 0).restartCount 100.0}}{{ printf "%s\n" (index .metadata.labels
"app.kubernetes.io/instance")}}{{end}}{{end}}'
```

Полезные ссылки:

- Overview of kubectl
- Kubernetes most usefull commands
- Kubectl Reference Docs

Задание:

Для выполнения задания можно использовать кластер из предыдущего задания или, если VM с ним уже удалена, создать новый.

1. Запустите pod с образом ubuntu:bionic и именем ubuntu, используя команду kubectl run (команду и вывод сохранить).
2. Проверьте состояние созданного pod, используя команду kubectl get (команду и вывод сохранить).
3. Определите uptime pod'a с помощью команды kubectl exec. Если у вас это не получится, то укажите по какой причине (команду и вывод сохранить).
4. Пересоздайте pod, чтобы его состояние было running (команду и вывод сохранить).
5. Попробуйте снова определить uptime pod'a с помощью команды kubectl exec (команду и вывод сохранить).
6. Получите информацию о том, на какой node запущен ваш pod с помощью команды kubectl get (команду и вывод сохранить).
7. В ответ пришлите все сохраненные команды и выводы из консоли, указанные выше.