Troubleshooting

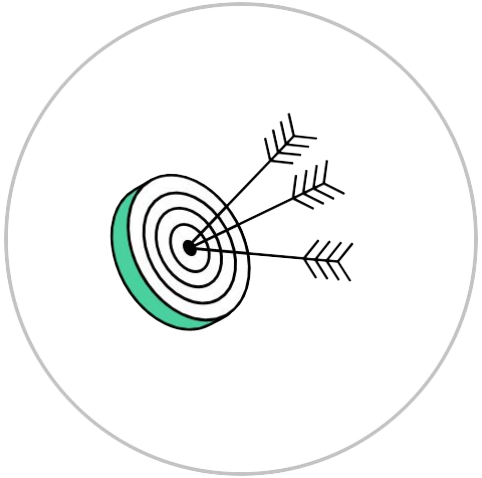
Кирилл Касаткин
DevOps-инженер, Renue

Кирилл Касаткин

DevOps-инженер, Renue

Цели занятия

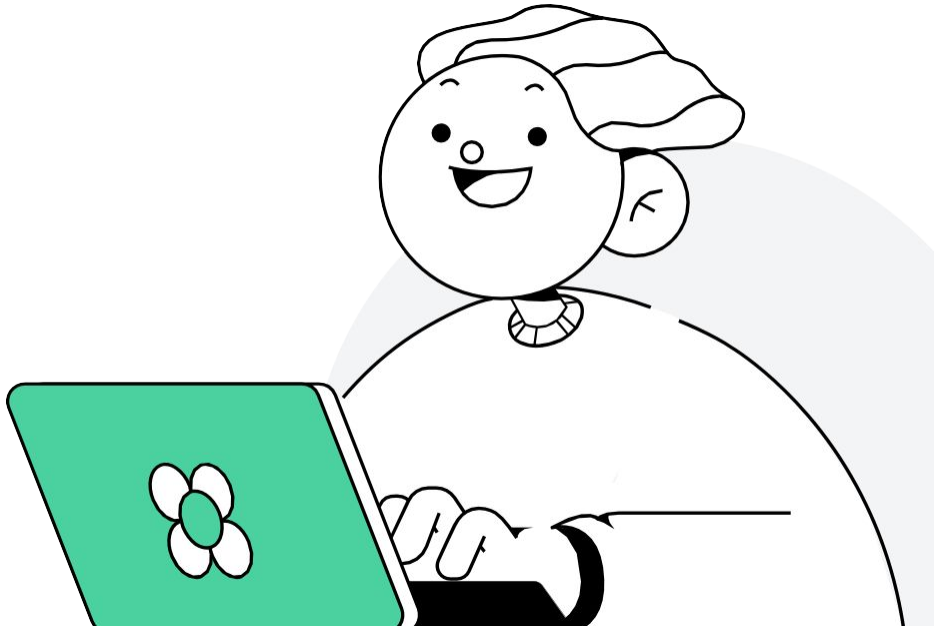
- Изучить алгоритм поиска неисправностей
- Познакомиться с базовыми командами



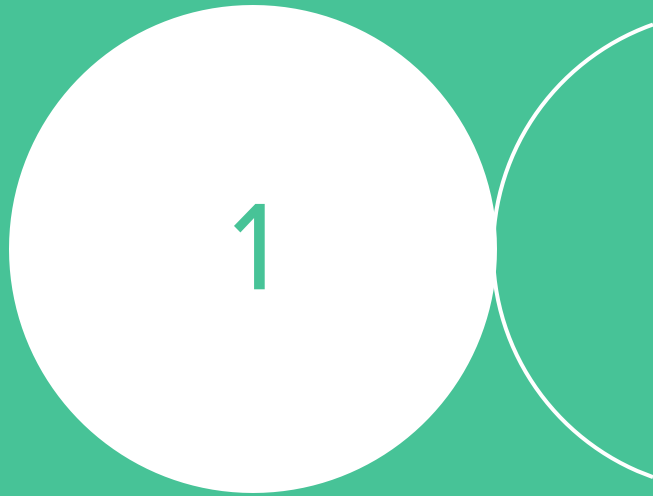
План занятия

1. Cluster
2. Журналы логов
3. Application
4. Network
5. Итоги
6. Домашнее задание

* Нажмите на нужный раздел для перехода



Cluster



Проверка API-сервера

1. Проверить подключение к API-серверу (удалённо, локально)

```
kubectl get nodes
The connection to the server 51.250.10.123:6443 was refused - did you specify the
right
```

2. Проверить статус kubelet, container runtime

```
systemctl status kubelet
```

```
systemctl status docker
```

3. Получить дамп системы

```
kubectl cluster-info dump
```

Проверка Node

1. Проверить состояние нод

```
kubectl get nodes
```

```
kubectl get node node_name -o yaml
```

```
kubectl describe nodes node_name
```

2. Проверить статус kubelet, container runtime на нодах

```
systemctl status kubelet
```

```
systemctl status docker
```

Проверка состояния системных pod

1. Проверить состояние pod в namespace kube-system

```
kubectl get pods -n kube-system
```

```
kubectl describe pod node_name -n kube-system
```

2. Проверить статус сервисов на control plane, если они запущены в виде сервисов

```
systemctl status kube-apiserver
```

```
systemctl status kube-scheduler
```

Журналылогов

2

Проверка логов

1. Проверить логи сервисов kubelet, container runtime

```
journalctl -u kubelet
```

```
journalctl -u docker
```

2. Проверить логи системы

```
less /var/log/kube-apiserver.log
```

```
less /var/log/kube-scheduler.log
```

```
less /var/log/kube-controller-manager.log
```

Проверка логов системных подов

3 Проверить логи системных подов, если они запущены как поды

```
kubectl logs -n kube-system kube-apiserver-****
```

```
kubectl logs -n kube-system kube-scheduler-****
```

```
kubectl logs -n kube-system kube-controller-manager-****
```

Application

3

Проверка работы приложения

1. Проверить статус подов

```
kubectl get pods (-n namespace_name)
```

```
kubectl get pods (-n namespace_name) -o wide
```

```
kubectl describe pod pod_name (-n namespace_name)
```

2. Запуск команды внутри контейнера пода

```
kubectl exec pod_name (-n namespace_name) -- command
```

```
kubectl exec pod_name (-n namespace_name) -c container_name -- command
```

Проверка работы приложения

3 Проверить лог подов

```
kubectl logs pod_name
```

```
kubectl logs pod_name (-n namespace_name) -c container_name
```

Network

4

Проверка работы сети

1. Проверить логи подов kube-proxy, coredns

```
kubectl logs -n kube-system kube-proxy-****
```

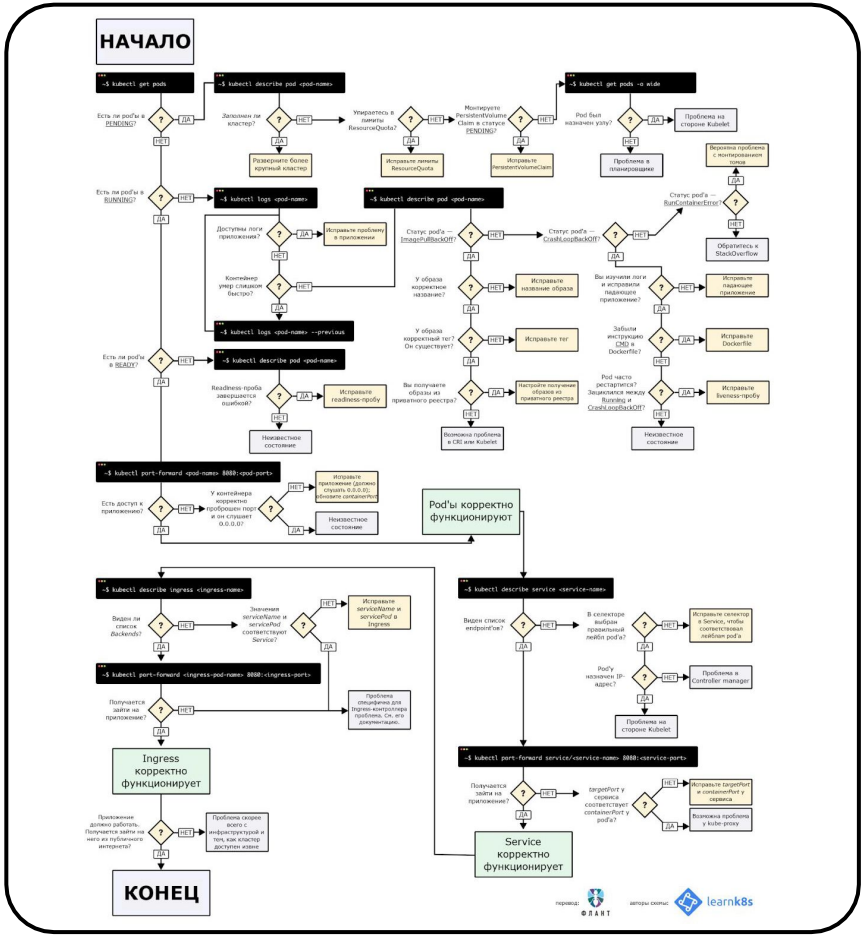
```
kubectl describe kube-system coredns-****
```

2. Используя специальные поды с сетевыми тулами, провести диагностику

```
kubectl exec multitool -- command
```

```
kubectl exec netshoot -- command
```

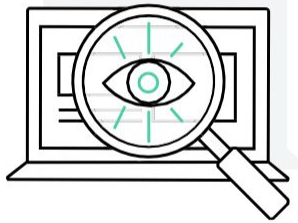
Troubleshooting roadmap



Источник

Демонстрация работы

Troubleshooting



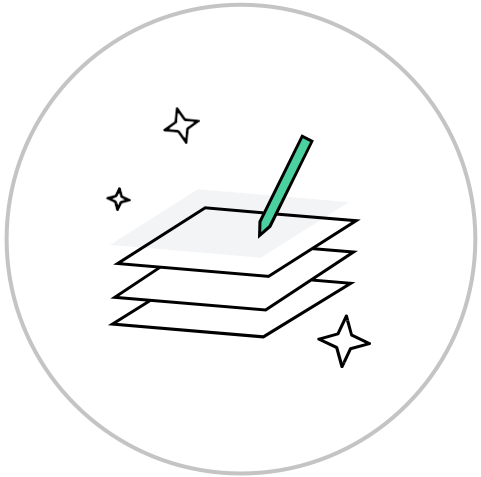
Итоги

- Узнали, какие шаги по устранению ошибок бывают
- Разобрались, как декомпозировать задачи по устранению
- Рассмотрели команды, с помощью которых получают отладочную информацию
- Попробовали подключиться к кластеру и посмотреть в работе команды, изученные на занятии

Домашнее задание

Давайте посмотрим ваше домашнее задание

1. Вопросы о домашней работе задавайте в чате группы
2. Задачи можно сдавать по частям
3. Зачёт по домашней работе ставят после того, как приняты все задачи



Задавайте вопросы и пишите отзыв о лекции

Кирилл Касаткин
DevOps-инженер, Renue