Debug и Troubleshooting в Kubernetes

Слайд 1

Цель урока

- Научиться находить и устранять причины PodPending, ImagePullBackOff, CrashLoopBackOff
- Разобрать основные команды для диагностики
- Закрепить практикой через симуляцию ошибок

Слайд 2

Почему это важно

- Ошибки в подах — самая частая проблема в Kubernetes
- Ошибка в одном сервисе может остановить весь деплой
- Умение быстро находить причину = меньше простоя

Слайд 3

Основные инструменты

kubectl

- `kubectl get pods -A` — статус
- `kubectl describe pod <name>` — детали, события
- `kubectl logs <name>` — логи контейнера
- `kubectl exec -it <name> -- sh` — доступ внутрь

Kubernetes Events

- показывают, что именно пошло не так: ошибки планировщика, контейнера, сети

Слайд 4

PodPending

Симптом: под не запускается, висит в статусе *Pending*.

Причины

- Нет доступных ресурсов (CPU/Memory)
- Невозможно смонтировать PVC
- Нод нет в Ready
- Невыполнимые требования (nodeSelector, taints/tolerations)

Как диагностировать

```
kubectl describe pod <name>
kubectl get nodes
```

Слайд 5

ImagePullBackOff

Симптом: под не может загрузить образ.

Причины

- Неверный image name или tag
- Нет доступа к приватному registry
- Подключен неправильный imagePullSecret
- Образ удалён или не существует

Как диагностировать

```
kubectl describe pod <name>
```

Слайд 6

CrashLoopBackOff

Симптом: контейнер стартует → падает → перезапускается по кругу.

Причины

- Ошибка в приложении
- Неверные переменные окружения
- Нет доступа к внешнему сервису
- Приложение завершается слишком быстро (exit code ≠ 0)
- Ошибка конфигурации (не тот порт)

Как диагностировать

Слайд 7

Практика 1

Симуляция PodPending

1. Создать Deployment с большими ресурсами:

```
resources:
  requests:
    cpu: "4"
    memory: "8Gi"
```

2. Наблюдать Pending
3. Уменьшить ресурсы и проверить исправление

Слайд 8

Практика 2

Симуляция ImagePullBackOff

1. Указать несуществующий тег:

```
image: nginx:12345
```

2. Проверить Events
3. Исправить тег

Слайд 9

Практика 3

Симуляция CrashLoopBackOff

1. Запустить контейнер, который сразу делает exit 1
2. Снять логи
3. Добавить корректный entrypoint или конфиг

Слайд 10

Рекомендации

- Всегда начинай с `describe`
- Проверяй Events — там источник истины
- Логи — главный инструмент при CrashLoop
- Тестируй приложение локально
- Используй минимальные ресурсы по умолчанию