Linux Recap

ЧАСТЬ 1. ТЕОРИЯ

1. Роль Linux в DevOps

Факты:

- 90% серверов в проде — Linux
- Kubernetes, Docker, CI/CD раннеры — Linux
- Облака = Linux + сети + автоматика

DevOps ≠ «знаю команды»
DevOps = понимаю, **что происходит в системе**

2. Архитектура Linux (минимум, но точно)

```
Hardware
↓
Kernel
↓
System libraries
↓
Shell / Services
↓
User / Applications
```

Что важно DevOps:

- Kernel управляет: CPU, памятью, сетью, дисками
- Пользователь **не работает напрямую с kernel**
- Всё — файлы (включая процессы и устройства)

3. Filesystem Hierarchy (FHS)

Обязательный минимум:

Каталог	Зачем нужен
`/`	корень системы
`/etc`	конфигурации
`/var`	логи, кеши, state
`/home`	пользователи
`/opt`	сторонние приложения
`/tmp`	временные файлы
`/proc`	информация о процессах
`/sys`	информация о системе

4. Работа с файлами и текстом

Команды, которые используются постоянно:

```
ls, cd, pwd
cp, mv, rm, mkdir
cat, less, head, tail
```

Работа с логами:

```
tail -f /var/log/syslog
grep ERROR app.log
```

Pipeline — must have:

```
cat file | grep ERROR | awk '{print $5}'
```

DevOps читает логи, не «открывает файл в nano».

5. Пользователи и права доступа

Пользователи и группы

```
useradd, userdel
groupadd
id, whoami
```

Права доступа

```
rwx r-x r--
```

Числовое представление:

```
chmod 755 script.sh
```

Владельцы:

6. Процессы и сигналы

Просмотр:

```
ps aux
top / htop
```

Управление:

```
kill PID
kill -9 PID
```

Важно понимать:

- SIGTERM — корректное завершение
- SIGKILL — принудительное убийство

Kubernetes = тот же Linux, только автоматизированный.

7. Сервисы и systemd

Почти везде используется `systemd`.

Команды:

```
systemctl status nginx
systemctl start nginx
systemctl stop nginx
systemctl enable nginx
```

Логи сервиса:

```
journalctl -u nginx
journalctl -xe
```

DevOps обязан уметь:

- понять, **почему сервис не стартует**

8. Сеть в Linux (recap)

Минимум:

```
ip a
ip r
ss -lntup
ping
curl
```

Типовые вопросы:

- На каком интерфейсе слушает сервис?
- Открыт ли порт?
- Доступен ли endpoint?

9. Диски и место

Проверка:

```
df -h
du -sh *
lsblk
mount
```

Частый инцидент:

- сервис упал
- причина — **закончился диск**

DevOps проверяет это первым.

10. Переменные окружения

```
env
export VAR=value
```

Зачем:

- конфигурация приложений
- секреты
- CI/CD
- контейнеры

11. Bash как инструмент автоматизации

Что должен уметь DevOps:

- писать простые скрипты
- понимать чужие

Минимум:

```
#!/bin/bash
if, for, while
$?, &&, ||
```

ЧАСТЬ 2. ПРАКТИКА (ОСНОВНОЕ ВРЕМЯ)

Практика 1. Анализ системы

Задачи:

- узнать версию ОС
- посмотреть uptime
- загрузку CPU и RAM
- свободное место

Команды:

```
uname -a
uptime
top
df -h
```

Практика 2. Работа с логами

Задачи:

- найти ошибки в логе
- отследить логи в реальном времени

```
grep ERROR app.log
tail -f app.log
```

Практика 3. Пользователи и права

Задачи:

- создать пользователя
- выдать права на каталог
- проверить доступ

```
useradd devops
chown devops:devops /opt/app
chmod 755 /opt/app
```

Практика 4. Сервисы

Задачи:

- проверить статус сервиса
- найти причину, почему он не работает
- посмотреть логи

```
systemctl status nginx
journalctl -u nginx
```

Практика 5. Сеть

Задачи:

- проверить, слушает ли порт
- проверить доступность сервиса

```
ss -lntup
curl localhost:8080
```

Практика 6. Мини-инцидент

Сценарий:

- приложение не стартует
- проблема может быть:
  - права
  - порт
  - сервис
  - диск

Цель:

- **найти причину, не угадывая**

ЧАСТЬ 3. СОБЕСЕДОВАНИЯ

Частые вопросы

- Где лежат логи?
- Чем SIGTERM отличается от SIGKILL?
- Почему сервис стартует вручную, но не через systemd?
- Что проверить, если сервис недоступен по сети?
- Почему контейнер падает с permission denied?

ДОМАШНЕЕ ЗАДАНИЕ

Обязательное

Пройти Linux Labs:

- KodeKloud Linux Basics
- KodeKloud Linux for DevOps (часть 1)

Фокус:

- процессы
- файлы
- пользователи
- сеть
- сервисы

Дополнительно (по желанию)

- Написать bash-скрипт:
  - проверяет свободное место
  - если < 20% — выводит предупреждение
- Разобрать чужой скрипт и объяснить, что он делает