?????? Linux

Справочник основных команд и работы со штатными компонентами

- Основные команды
- Управление дисками (fdisk)
- Монтирование (mount)
- Программный RAID (mdadm)
- UFW Firewall
- Справочник по SSH
- Bash скрипты
- Мониторинг и отладка
  - Journalctl - справочник
  - systemctl — справочник
  - top и htop
  - netstat, ss и ip
- Netplan

???????? ???????

????? ? ????????

ls - ???????? ??????????? ????????

Показывает список файлов и папок в текущем или указанном каталоге.

```
ls -lah /var/www/html
```

Ключи: `-l` (детальный вывод), `-a` (включить скрытые файлы), `-h` (человеко-читаемые размеры)

cd - ????? ????????

Переход в другой каталог.

```
cd /var/www/html
```

Полезно: `cd ..` (на уровень вверх), `cd ~` или просто `cd` (домашний каталог)

mkdir - ???????? ?????????

Создает новые каталоги.

```
mkdir /var/www/new_site
mkdir -p /var/www/new_site/assets
```

Ключи: `-p` — создать промежуточные каталоги, если их нет

rm - ???????? ?????? ? ?????????

Удаляет файлы и каталоги.

```
rm file.txt
rm -rf /var/www/old_site
```

Ключи: `-r` — рекурсивно (для каталогов), `-f` — без подтверждения

Внимание: Используйте `sudo` с осторожностью при удалении системных файлов!

??????????? ??????????????

adduser / useradd - ?????????? ???????????

Создает нового пользователя в системе.

```
sudo adduser newuser
```

Пример с явным указанием домашнего каталога и оболочки:

```
sudo useradd -m -d /home/newuser -s /bin/bash newuser
```

passwd - ?????????? ?????? ???????????

Назначает или меняет пароль для пользователя.

```
sudo passwd newuser
```

deluser / userdel - ???????? ???????????

Удаляет пользователя из системы.

```
sudo deluser newuser
sudo userdel -r newuser  # удалить пользователя и его домашний каталог
```

usermod - ????????? ?????????? ???????????

Назначение пользователя в группы, смена оболочки и домашнего каталога.

```
sudo usermod -aG sudo newuser   # добавить в группу sudo
sudo usermod -d /home/newhome newuser  # смена домашнего каталога
sudo usermod -s /bin/zsh newuser      # смена оболочки
```

groups - ???????? ????? ???????????

```
groups newuser
```

????? ??????

?? ?????

Ищет файлы по имени в указанном каталоге или во всей системе.

```
# поиск всех файлов с расширением .log в /var/log
find /var/log -name "*.log"
#поиск файлов с точным именем
find /home/user -name "notes.txt"
```

?? ???????

Ищет файлы определенного размера или диапазона размеров.

```
# найти файлы больше 100M в /var/www
find /var/www -type f -size +100M
#найти файлы меньше 1K
find /home/user -type f -size -1k


#найти файлы размером от 10M до 100M
find / -type f -size +10M -size -100M
```

?????????????

Некоторые полезные опции команды `find`:

- `-type f` — искать только файлы
- `-type d` — искать только каталоги
- `-iname` — поиск без учета регистра: `find /home -iname "*.txt"`
- `-maxdepth N` — ограничить глубину поиска (например, только текущий каталог): `find /var/www -maxdepth 1 -name "*.html"`
- `-exec command {} \;` — выполнить команду для каждого найденного файла, например удалить: `find /tmp -type f -name "*.tmp" -exec rm {} \;`

????? ? ??????????

chmod - ????????? ???? ?? ????? ? ????????

Устанавливает права доступа для владельца, группы и других пользователей.

Формат: три цифры **XYZ** — владельцу, группе, остальным. Каждая цифра — сумма прав: 4 (чтение), 2 (запись), 1 (выполнение).

- 7 = 4+2+1 → rwx (читать, писать, выполнять)
- 6 = 4+2 → rw- (читать, писать)
- 5 = 4+1 → r-x (читать, выполнять)

- 4 = 4 → r-- (только читать)
- 0 = --- → нет прав

Примеры:

```
chmod 755 /var/www/html        # rwxr-xr-x: владелец все, группа/остальные читать+выполнять
chmod 644 /var/www/html/index.html # rw-r--r--: владелец читать+писать, группа/остальные только читать
chmod 700 ~/private               # rwx------: только владелец
```

Когда использовать:

- 755 — каталоги веб-сайта (все могут читать и выполнять, владелец пишет)
- 644 — файлы конфигурации (владелец пишет, все остальные только читают)
- 700 — приватные скрипты или файлы пользователя

chown - ????? ????????? ? ??????

Назначает владельца и группу для файла или каталога.

```
sudo chown www-data:www-data /var/www/html -R
```

Ключи: `-R` — рекурсивно применить ко всем файлам и папкам

Примеры использования:

- Назначить веб-серверу: `sudo chown -R www-data:www-data /var/www`
- Назначить пользователя и группу: `sudo chown user:group file.txt`

chgrp - ????? ??????

```
sudo chgrp www-data /var/www/html -R
```

????????? ???????

hostname - ????? ????? ?????

Показывает или изменяет имя хоста системы.

```
hostname                # текущий хост
sudo hostname newhost    # временно изменить имя хоста
sudo nano /etc/hostname  # для постоянного изменения
sudo nano /etc/hosts     # исправить запись старого имени
```

reboot / shutdown - ???????????? ? ??????????

```
sudo reboot
sudo shutdown -h now
sudo shutdown -r +10    # перезагрузка через 10 минут
```

free / df / du - ????????????? ?????? ? ??????

```
free -h                # RAM и swap
df -h                  # свободное место на дисках
du -sh /var/log         # размер каталога
```

ps / kill - ?????????? ??????????

```
ps aux | grep nginx
kill 12345
kill -9 12345  # принудительное завершение
```

???????

- Для проверки прав: `ls -l /var/www/html`
- Для смены владельца только файлов, а не каталога: `sudo chown user:user file.txt`
- Для массовой смены прав на все скрипты: `chmod +x *.sh`
- Для создания резервной копии перед удалением: `cp -r /var/www/old /backup/old`

???????? ??????

- Ошибка: нет прав на файл → используйте `sudo`
- Ошибка: не удалось изменить владельца → проверьте синтаксис `user:group`
- Ошибка: после chmod 777 все могут писать → избегайте на продуктивных серверах, использовать 755/644
- Ошибка: пользователь не может писать в каталог → проверить владельца и права каталога (`ls -ld /path`)

??????????? ??????? (fdisk)

???????? ? ???? ??????????????

ВНИМАНИЕ: Деструктивная операция!

`fdisk` — это мощный инструмент, который напрямую изменяет структуру вашего диска. Неправильное использование может привести к **полной и необратимой потере всех данных** на диске. Перед началом работы:

- **Убедитесь, что у вас есть резервная копия** всех важных данных.
- **Дважды проверьте имя диска** (например, `/dev/sdb`, а не `/dev/sda`), чтобы не отформатировать системный диск.

`fdisk` — это стандартная утилита командной строки Linux для создания и управления таблицами разделов диска. Она поддерживает как старый формат MBR (Master Boot Record), так и современный GPT (GUID Partition Table).

????????? ??????????: ???????? ?????? ?????

??? 1: ????????????? ?????

Используйте `lsblk` для просмотра блочных устройств. Новый, неразмеченный диск обычно не будет иметь подразделов.

```
lsblk
```

Предположим, наш новый диск — `/dev/sdb`.

??? 2: ?????? fdisk

Запустите fdisk в интерактивном режиме для нужного диска.

```
sudo fdisk /dev/sdb
```

??? 3: ?????? ? ????????????? ?????? (???????? ???????)

После запуска fdisk вы попадаете в его командную строку. Вот основные команды:

- `m` — показать меню помощи со всеми командами.

- `p` — показать текущую таблицу разделов. Полезно для просмотра текущего состояния диска.
- `g` — **(Рекомендуется)** создать новую, пустую таблицу разделов GPT. Это современный стандарт, снимающий ограничения MBR (диски > 2ТБ, до 128 разделов).
- `o` — создать новую, пустую таблицу разделов MBR. Используйте только для совместимости со старыми системами.
- `n` — создать новый раздел. Утилита спросит номер раздела, первый и последний сектор. Можно указать размер, например, `+50G`.
- `d` — удалить раздел. Утилита спросит номер раздела для удаления.
- `t` — изменить тип раздела. Полезно для создания разделов под LVM (`Linux LVM`) или swap (`Linux swap`). Команда `L` выведет список всех доступных типов.
- `w` — **(Важно!)** записать изменения на диск и выйти. Все сделанные вами изменения станут постоянными.
- `q` — выйти без сохранения изменений. Безопасный способ отменить все, что вы сделали.

??? 4: ?????????? ?????????? ? ????

После записи изменений (`w`), ядро Linux может не сразу "увидеть" новую таблицу разделов. Чтобы сообщить ядру об изменениях без перезагрузки, используйте `partprobe`.

```
sudo partprobe /dev/sdb
```

??? 5: ???????? ???????? ???????

Теперь, когда раздел создан (например, `/dev/sdb1`), его нужно отформатировать, то есть создать на нем файловую систему. `ext4` — отличный выбор по умолчанию.

```
sudo mkfs.ext4 /dev/sdb1
```

После этого раздел готов к монтированию. Информацию о монтировании смотрите в соответствующей вкладке.

?????? ??????: ???????? ?????? ??????? ?? ???? ????

Ниже приведен полный пример команд и ответов в `fdisk` для разметки диска `/dev/sdb` с одним разделом типа "Linux filesystem" с использованием GPT.

```
$ sudo fdisk /dev/sdb

Welcome to fdisk (util-linux 2.37.2).

Changes will remain in memory only, until you decide to write them.
```

```
Be careful before using the write command.


Device does not contain a recognized partition table.


Command (m for help): g
Created a new GPT disklabel (GUID: ...).


Command (m for help): n
Partition number (1-128, default 1): <ENTER>
First sector (2048-..., default 2048): <ENTER>
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-..., default ...): <ENTER>


Created a new partition 1 of type 'Linux filesystem' and of size ...


Command (m for help): p
Disk /dev/sdb: ... GiB, ... bytes, ... sectors
...
Disklabel type: gpt
Disk identifier: ...


Device      Start   End Sectors  Size Type
/dev/sdb1    2048   ...     ...  ...G Linux filesystem


Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

????????????? (mount)

???????????? ????????? ??????

Для автоматического подключения диска при загрузке системы его нужно добавить в файл `/etc/fstab`. Используйте UUID — это уникальный идентификатор, который не изменится, в отличие от имени `/dev/sdb1`.

??? ?: ???????? ????? ???????????

Это папка, куда будет "подключаться" ваш раздел.

```
sudo mkdir /mnt/data
```

??? ?: ??????? UUID ?????? ???????

```
sudo blkid /dev/sdb1
```

??? ?: ???????? ?????? ? /etc/fstab

Откройте файл `/etc/fstab` в текстовом редакторе (например, `sudo nano /etc/fstab`) и добавьте в самый конец файла следующую строку, заменив UUID на полученный выше.

```
# Пример строки для /etc/fstab
UUID=a1b2c3d4-e5f6-7890-1234-567890abcdef  /mnt/data  ext4  defaults,nofail  0  2
```

Краткое описание опций:

- `defaults`: стандартный набор опций (чтение/запись, авто-монтирование и т.д.).
- `nofail`: важная опция. Если диск будет отключен, система все равно загрузится, не ожидая его.

?????? ????????? ???? ??????????: `[dump]` ? `[pass]`

Две последние цифры в строке `fstab` (`0 2`) управляют резервным копированием и проверкой файловой системы при загрузке.

- **Первая цифра (`dump`)**: Управляет утилитой резервного копирования `dump`.
  - `0`: Не создавать резервную копию. Это стандартное значение для 99% систем, так как `dump` редко используется в наши дни.
  - `1`: Создавать резервную копию.
- **Вторая цифра (`pass` или `fsck`)**: Определяет порядок проверки файловой системы (`fsck`) при загрузке.

- `0`: Не проверять. Используется для swap-разделов, сетевых файловых систем или виртуальных ФС.
- `1`: Проверять в первую очередь. **Этот параметр должен использоваться ИСКЛЮЧИТЕЛЬНО для корневого раздела (`/`).**
- `2`: Проверять во вторую очередь. Используется для **всех остальных** локальных разделов, которые вы хотите проверять при загрузке (например, `/home`, `/var`, или, как в нашем примере, `/mnt/data`).

Итог: Для всех дополнительных дисков и разделов с данными (не корневых) самая правильная и распространенная комбинация — `0 2`.

???????????? ??????? ?????

1. SAMBA (CIFS) - ??? Windows-?????

Это пошаговое руководство по настройке автоматического монтирования сетевых папок Windows.

??? 1: ????????? ???????

```
sudo apt update && sudo apt install -y cifs-utils
```

??? 2: ???????? ????? ? ???????? ???????

Чтобы не хранить пароль в открытом виде в `/etc/fstab`, создадим отдельный защищенный файл. Это самый безопасный подход.

```
# Создаем файл
sudo nano /etc/samba/credentials
#Добавляем в него 2 строки
username=your_username
password=your_password


#Устанавливаем права (только root сможет его читать)
sudo chmod 600 /etc/samba/credentials
```

??? 3: ???????? ????? ???????????

Это локальная папка, в которую будет "подключен" сетевой ресурс.

```
sudo mkdir /mnt/samba
```

??? 4: ?????????? ?????? ? /etc/fstab

Откройте файл `/etc/fstab` в текстовом редакторе (например, `sudo nano /etc/fstab`) и добавьте в самый конец файла следующую строку. Замените значения на свои.

```
# Запись для автоматического монтирования сетевой папки SAMBA
//server-ip/share_name /mnt/samba cifs
credentials=/etc/samba/credentials,nofail,uid=1000,gid=1000,vers=3.0,iocharset=utf8,file_mode=
0664,dir_mode=0775 0 0
```

Описание опций CIFS:

- `credentials`: Путь к файлу с логином и паролем.
- `nofail`: Критически важная опция. Позволяет системе загрузиться, даже если сетевой ресурс недоступен.
- `uid=1000,gid=1000`: Устанавливает владельца и группу для смонтированных файлов. `1000` обычно соответствует первому пользователю в системе. Узнать свой UID можно командой `id`.
- `vers=3.0`: Версия протокола SMB. `3.0` - хороший, современный выбор.
- `iocharset=utf8`: Кодировка для имен файлов, важна для поддержки кириллицы.
- `file_mode=0664,dir_mode=0775`: Явно задает права для файлов и папок.

2. NFS - ??? Linux/Unix-?????

NFS (Network File System) — это "родной" для Linux способ организации сетевого доступа к файлам.

Установка клиента:

```
sudo apt update && sudo apt install -y nfs-common
```

?????? 1: ??????????? ? ???????? ???????????? (?????????????)

Этот пример использует опцию `hard`, которая заставляет программу "ждать", пока сервер не ответит, что обеспечивает целостность данных. `nfsvers=4` явно указывает использовать современную версию протокола.

```
server-ip:/path/to/share /mnt/nfs nfs defaults,nofail,hard,nfsvers=4,rsize=8192,wsize=8192 0 0
```

?????? 2: "??????" ???????????? ??? ??????????? ??????

Опция `soft` позволяет программе получить ошибку, если сервер не отвечает в течение определенного времени (`timeo`). Это полезно для ресурсов, где "быстрый отказ" лучше, чем "зависание" приложения.

```
server-ip:/path/to/non-critical/share /mnt/nfs-soft nfs defaults,nofail,soft,timeo=15 0 0
```

Описание опций NFS:

- `defaults`: Набор стандартных опций (`rw`, `suid`, `dev`, `exec`, `auto`, `nouser`, `async`, и `hard`).
- `nofail`: Позволяет системе загрузиться, даже если NFS-сервер недоступен.
- `hard` / `soft`: Поведение при недоступности сервера. `hard` (по умолчанию) - программа "зависнет", ожидая ответа. `soft` - вернет ошибку по таймауту.
- `nfsvers=4`: Явно указать версию протокола NFS. Рекомендуется использовать `4` или `3`.
- `timeo=15`: Таймаут в десятых долях секунды (15 = 1.5 сек). Используется с опцией `soft`.
- `rsize=8192,wsize=8192`: Размер блоков для чтения/записи. Может повлиять на производительность.

??????????? RAID (mdadm)

1. ????????? mdadm ? ???????? RAID

Установите утилиту и создайте массив нужного уровня. Замените `/dev/sd[b-e]` на ваши диски. Перед этим убедитесь, что они не содержат нужных данных.

Установка:

```
sudo apt update && sudo apt install -y mdadm
```

Пример создания RAID 1 (зеркало, 2 диска):

```
sudo mdadm --create --verbose /dev/md0 --level=1 --raid-devices=2 /dev/sdb /dev/sdc
```

Пример создания RAID 5 (мин. 3 диска):

```
sudo mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 /dev/sdb /dev/sdc /dev/sdd
```

Пример создания RAID 6 (мин. 4 диска):

```
sudo mdadm --create --verbose /dev/md0 --level=6 --raid-devices=4 /dev/sdb /dev/sdc /dev/sdd
/dev/sde
```

Пример создания RAID 10 (мин. 4 диска):

```
sudo mdadm --create --verbose /dev/md0 --level=10 --raid-devices=4 /dev/sdb /dev/sdc /dev/sdd
/dev/sde
```

2. ????????? ? ?????????????? RAID

После создания массив нужно отформатировать и настроить для автозагрузки.

Шаг А: Отформатируйте RAID-массив.

```
sudo mkfs.ext4 /dev/md0
```

Шаг Б: Сохраните конфигурацию массива, чтобы он собирался при загрузке.

```
sudo mdadm --detail --scan | sudo tee -a /etc/mdadm/mdadm.conf
```

Шаг В: Обновите initramfs, чтобы ядро знало о RAID на раннем этапе загрузки.

```
sudo update-initramfs -u
```

Шаг Г: Добавьте массив в `/etc/fstab` для автоматического монтирования, используя его UUID.

```
# Узнаем UUID массива
sudo blkid /dev/md0
#Добавляем в /etc/fstab (замените UUID и /mnt/raid)
UUID=a1b2c3d4-e5f6-7890-1234-567890abcdef  /mnt/raid  ext4  defaults,nofail,discard  0  2
```

Комментарии к опциям монтирования:

- `nofail`: критически важно для RAID. Если массив не соберется при загрузке, система не "зависнет" в ожидании.
- `discard`: полезная опция, если ваш RAID собран на SSD-дисках. Включает поддержку TRIM для повышения производительности и долговечности.

?????? ????????? ???? ??????????: `[dump]` ? `[pass]`

Две последние цифры в строке `fstab` (`0 2`) управляют резервным копированием и проверкой файловой системы при загрузке.

- **Первая цифра (`dump`)**: Управляет утилитой резервного копирования `dump`.
  - `0`: Не создавать резервную копию. Это стандартное значение для 99% систем, так как `dump` редко используется в наши дни.
  - `1`: Создавать резервную копию.
- **Вторая цифра (`pass` или `fsck`)**: Определяет порядок проверки файловой системы (`fsck`) при загрузке.
  - `0`: Не проверять. Используется для swap-разделов, сетевых файловых систем или виртуальных ФС.
  - `1`: Проверять в первую очередь. **Этот параметр должен использоваться ИСКЛЮЧИТЕЛЬНО для корневого раздела (`/`).**
  - `2`: Проверять во вторую очередь. Используется для **всех остальных** локальных разделов, которые вы хотите проверять при загрузке (например, `/home`, `/var`, или, как в нашем примере, `/mnt/data`).

Итог: Для всех дополнительных дисков и разделов с данными (не корневых) самая правильная и распространенная комбинация — `0 2`.

UFW Firewall

??? ????? UFW?

UFW — это "оболочка" для `iptables`, созданная для упрощения настройки брандмауэра. Она идеально подходит для защиты отдельных серверов, так как позволяет управлять правилами с помощью простых и интуитивно понятных команд, не углубляясь в сложный синтаксис `iptables`.

??????? ??????????

?????????

Обычно UFW уже установлен в Ubuntu, но если нет, его можно легко установить.

```
sudo apt update && sudo apt install ufw
```

???????? ???????

Показывает, активен ли брандмауэр, и список текущих правил.

```
sudo ufw status verbose
```

????????? / ??????????

Важно: Перед первым включением убедитесь, что вы добавили правило для SSH, иначе вы потеряете доступ к серверу!

```
sudo ufw enable
```

```
sudo ufw disable
```

?????????? ?????????

????????? ?????? ?? ?????????

Самая безопасная стратегия — запретить все входящие соединения и разрешить все исходящие. Это базовая настройка для любого сервера.

```
sudo ufw default deny incoming
```

```
sudo ufw default allow outgoing
```

?????????? ??????????

Добавление правил `allow` — основная задача при настройке. Это можно делать по имени сервиса, номеру порта или IP-адресу.

Разрешить SSH (обязательно!):

```
sudo ufw allow ssh
```

Разрешить HTTP и HTTPS:

```
sudo ufw allow http
```

```
sudo ufw allow https
```

Разрешить подключение к порту 5432 по протоколу TCP:

```
sudo ufw allow 5432/tcp
```

Разрешить подключения с определенного IP-адреса:

```
sudo ufw allow from 192.168.1.100
```

Разрешить подключения с IP-адреса на конкретный порт:

```
sudo ufw allow from 192.168.1.100 to any port 22
```

???????? ??????

Удалить правило можно, написав его в обратном порядке, или по номеру.

Удаление по описанию:

```
sudo ufw delete allow http
```

Удаление по номеру (более удобный способ):

```
# Сначала смотрим правила с номерами
sudo ufw status numbered
```

```
#Затем удаляем правило по его номеру
sudo ufw delete 1
```

??????????? ?? SSH

1. ????????? ? ????????? SSH-??????? (?? ??????? Linux)

????????? OpenSSH

Для Debian / Ubuntu серверная часть устанавливается пакетом `openssh-server`:

```
sudo apt update && sudo apt install -y openssh-server
```

??????????? ???????????? (`/etc/ssh/sshd_config`)

Перед полным отключением паролей обязательно добавьте ваш публичный ключ на сервер (см. раздел 2 и 3) и проверьте вход в отдельном окне терминала!

Откройте главный конфигурационный файл:

```
sudo nano /etc/ssh/sshd_config
```

Рекомендуемые базовые параметры:

```
# Порт для входящих подключений (по умолчанию 22, смена снижает фоновый шум от сканеров)
Port 2222

# ?????? ??????? ????? ??? root (???? ?????? ??????? ????????????? ? sudo)

PermitRootLogin no
```

????????? ??????????????? ?? SSH-??????

```
PubkeyAuthentication yes
```

??????????? ??????????????? ?? ?????? (?????? ????? ???????? ????? ?? ?????!)

```
PasswordAuthentication no
KbdInteractiveAuthentication no
```

?????? ????? ?????????????? ??? ??????

```
PermitEmptyPasswords no
```

?????????????? PAM-???????

```
UsePAM yes
```

?? ?????? ??????????? Ubuntu 22.04+ ? Debian 12 (??????? `sshd_config.d`)

В современных дистрибутивах и облачных образах настройки из каталога `/etc/ssh/sshd_config.d/*.conf` переопределяют главный файл `sshd_config`.

В частности, файл `50-cloud-init.conf` часто принудительно возвращает `PasswordAuthentication yes`. Обязательно проверьте наличие переопределений:

```
ls -la /etc/ssh/sshd_config.d/
```

Если там есть файлы (например, `50-cloud-init.conf`), откройте их и также выставьте `PasswordAuthentication no`, либо удалите/закомментируйте конфликтную строку.

???????? ?????????? ? ?????????? ??????

Перед перезапуском обязательно протестируйте конфигурацию на ошибки:

```
sudo sshd -t
```

Если команда ничего не вывела — конфигурация корректна.

Перезапустите службу SSH (**не закрывая текущую сессию!**):

```
# Для Ubuntu / Debian:
sudo systemctl restart ssh

# ??? RHEL / CentOS / AlmaLinux / Rocky:

sudo systemctl restart sshd
```

---

2. ????????? SSH-?????? ?? ??????? ???????

??????? ?: ? MobaXterm (Windows)

1. В верхнем меню откройте **Tools** → **MobaKeyGen (SSH key generator)**.
2. Внизу выберите тип ключа **Ed25519** (рекомендуется) или **RSA** (с длиной не менее `4096` бит).
3. Нажмите кнопку **Generate** и двигайте курсором мыши по пустому полю для создания случайных данных.
4. (Рекомендуется) В поле **Key passphrase** введите пароль для защиты самого приватного ключа.

5. Нажмите **Save private key** и сохраните файл `.ppk` в надежное место на ПК.
6. Скопируйте всю строку открытого ключа из верхнего поля под надписью *«Public key for pasting into OpenSSH authorized_keys file»*.

Настройка подключения в MobaXterm: Нажмите *Session → SSH*, укажите *Remote host* и *Username*. Перейдите во вкладку *Advanced SSH settings*, отметьте **Use private key** и укажите путь к сохраненному `.ppk`.

??????? ?: ? PuTTY / PuTTYgen (Windows)

1. Запустите утилиту **PuTTYgen**.
2. Выберите тип ключа **Ed25519** или **RSA** (4096 бит).
3. Нажмите **Generate** и поводите мышью.
4. Укажите парольную фразу (*Key passphrase*), если требуется.
5. Нажмите **Save private key** для сохранения `.ppk` файла.
6. Скопируйте текст публичного ключа из верхнего окна (начинается на `ssh-ed25519...` или `ssh-rsa...`).

Настройка подключения в PuTTY: В окне PuTTY перейдите в *Connection → SSH → Auth → Credentials* (или *Auth* в старых версиях), в поле *Private key file for authentication* укажите ваш `.ppk` файл. Вернитесь в *Session*, введите IP-адрес и сохраните сессию.

??????? ?: ????? ??????? Linux / macOS / WSL

1. Сгенерируйте пару ключей командой:

   ```
   ssh-keygen -t ed25519 -C "your_email_or_comment"
   ```

   (или для RSA: `ssh-keygen -t rsa -b 4096`)
2. Нажмите Enter для сохранения пути по умолчанию (`~/.ssh/id_ed25519`) и при необходимости задайте кодовую фразу.
3. Автоматически скопируйте публичный ключ на сервер:

   ```
   ssh-copy-id -p 22 username@your_server_ip
   ```

   (ключ `-p` указывается, если порт SSH нестандартный).

---

3. ?????? ????????? ?????????? ????? ? ????? ??????? ?? ???????

Если утилита `ssh-copy-id` недоступна (например, при генерации через MobaXterm или PuTTY), добавьте ключ вручную.

Критически важно: SSH строго проверяет права на каталоги и файлы ключей. Если права выставлены неправильно (например, доступны для записи другим пользователям), служба

безопасности OpenSSH заблокирует вход по ключу.

1. Создайте папку `.ssh` в домашнем каталоге пользователя и задайте строгие права (только владелец):

   ```
   mkdir -p ~/.ssh
   chmod 700 ~/.ssh
   ```

2. Создайте или откройте файл `authorized_keys`:

   ```
   nano ~/.ssh/authorized_keys
   ```

3. Вставьте скопированный публичный ключ (всю строку целиком, без переносов). Сохраните файл (`Ctrl+O`, `Enter`, `Ctrl+X`).
4. Установите права на чтение и запись только для владельца:

   ```
   chmod 600 ~/.ssh/authorized_keys
   ```

5. (Опционально) Если вы настраивали ключи под `root` для другого пользователя, передайте ему права владения:

   ```
   sudo chown -R username:username /home/username/.ssh
   ```

---

4. ???????????? ??????? ??????????? ? ???????? ?????????????

1. ??????????? ???????????

```
ssh username@your_server_ip
```

2. ??????????? ? ????????????? ??????

```
ssh -p 2222 username@your_server_ip
```

3. ??????????? ? ????? ????????? ?????????? ?????

```
ssh -i ~/.ssh/my_custom_key username@your_server_ip
```

4. ??????? ????????????? ????? ??????????? ??????? (`~/.ssh/config`)

Чтобы не запоминать IP-адреса, порты и пути к ключам на Linux/macOS/WSL, создайте файл `~/.ssh/config`:

```
Host prod-server
    HostName 198.51.100.25
    User debian
    Port 2222
    IdentityFile ~/.ssh/id_ed25519
```

Теперь подключение выполняется короткой командой:

```
ssh prod-server
```

5. ??????????? ?????? ? ??????????? (SCP ? Rsync)

Копирование одного файла:

```
scp -P 2222 /path/to/local/file.txt username@your_server_ip:/path/to/remote/dir/
```

Синхронизация папки с докачкой через Rsync (быстрее и надежнее для больших объемов):

```
rsync -avz -e "ssh -p 2222" --progress /path/to/local/project/
username@your_server_ip:/var/www/project/
```

6. ????????? ??????? ????? (SSH Tunneling ? ??????????? ??)

Позволяет подключиться к удаленной базе данных (например, PostgreSQL на `5432`), доступ к которой из интернета закрыт фаерволом:

```
ssh -L 5433:localhost:5432 -p 2222 username@your_server_ip
```

После этого локальный клиент базы данных подключается к `localhost:5433`.

7. ????????? ??????? ????? (????? ?????????? ??????? ??????)

Позволяет перенаправить трафик с порта сервера на ваш локальный порт (например, демонстрация локального веб-приложения на порту 3000):

```
ssh -R 8080:localhost:3000 -p 2222 username@your_server_ip
```

8. ???????? SSH-??????? (?????? ? ?????? ?? NAT / ????? IP)

На компьютере за NAT (домашний ПК или внутренний сервер):

```
ssh -N -R 9090:localhost:22 user_on_vps@public_vps_ip
```

Теперь, зайдя на публичный сервер `public_vps_ip`, можно попасть на машину за NAT командой:

```
ssh -p 9090 user_on_home_pc@localhost
```

Bash ???????

??? 1: ???????? ????? ???????

Создайте файл с расширением `.sh`. Первая строка всегда должна быть "shebang" — `#!/bin/bash`. Он указывает системе, какой интерпретатор использовать для выполнения скрипта.

```
#!/bin/bash
#Это комментарий
echo "Привет, мир!"
```

??? 2: ?????????????? ???? ?? ??????????

По умолчанию текстовый файл не является исполняемым. Используйте команду `chmod +x`, чтобы это исправить.

```
chmod +x my_script.sh
```

??? 3: ?????? ???????

Для запуска укажите путь к файлу. Если вы находитесь в том же каталоге, используйте `./`.

```
./my_script.sh
```

????????????? ? ??????? Cron

Cron — это планировщик задач. Для редактирования задач текущего пользователя используется команда `crontab -e`. Каждая строка имеет формат: `МИНУТА ЧАС ДЕНЬ_МЕСЯЦА МЕСЯЦ ДЕНЬ_НЕДЕЛИ /путь/к/команде`.

- `* * * * *` — Каждую минуту.
- `0 * * * *` — Каждый час, в 00 минут.
- `0 */4 * * *` — Каждые 4 часа.
- `5 3 * * *` — Каждый день, в 3:05 ночи.
- `0 4 * * 0` — Каждое воскресенье, в 4:00 утра (0 - воскресенье).

Пример записи для выполнения скрипта ежедневно в 2:30 ночи:

```
30 2 * * * /home/user/scripts/backup.sh >/dev/null 2>&1
```

Часть `>/dev/null 2>&1` перенаправляет весь вывод (стандартный и ошибки) в "никуда", чтобы cron не отправлял вам письма по каждому запуску.

???????????? ??????? ????????

1. ????????? ??????????? ????????

Создает .tar.gz архив указанного каталога с датой в имени файла. Идеально для ежедневного бэкапа сайтов или важных документов.

```
#!/bin/bash
#Каталог, который нужно архивировать
SOURCE_DIR="/var/www/my-site"
# Каталог для хранения бэкапов
BACKUP_DIR="/mnt/backups/sites"
# Имя файла бэкапа с датой
BACKUP_FILENAME="my-site_$(date +%Y-%m-%d).tar.gz"
#Создание архива
tar -czf "$BACKUP_DIR/$BACKUP_FILENAME" -C "$(dirname "$SOURCE_DIR")" "$(basename
"$SOURCE_DIR")"
echo "Бэкап $BACKUP_FILENAME успешно создан."
```

2. ?????????? ????????? ????????????

Проверяет использование диска в указанном разделе. Если использование превышает порог, отправляет письмо на почту.

```
#!/bin/bash
#Порог в процентах
THRESHOLD=90
#Раздел для проверки
PARTITION="/"
#Email для оповещений
ADMIN_EMAIL="admin@example.com"
#Получаем текущее использование в процентах (без знака %)
CURRENT_USAGE=$(df "$PARTITION" | grep -v Filesystem | awk '{print $5}' | sed 's/%//g')
if [ "$CURRENT_USAGE" -gt "$THRESHOLD" ]; then
  echo "ВНИМАНИЕ: На сервере $(hostname) заканчивается место на диске ($PARTITION). Занято:
$CURRENT_USAGE%." | mail -s "Disk Space Alert" "$ADMIN_EMAIL"
fi
```

3. ???????? ? ?????????? ???????

Проверяет, активен ли указанный сервис (например, nginx). Если нет, пытается его перезапустить и отправляет уведомление.

```
#!/bin/bash
SERVICE_NAME="nginx"
ADMIN_EMAIL="admin@example.com"
if ! systemctl is-active --quiet "$SERVICE_NAME"; then
    echo "Сервис $SERVICE_NAME не активен! Попытка перезапуска..."
    systemctl restart "$SERVICE_NAME"
    sleep 2
    if systemctl is-active --quiet "$SERVICE_NAME"; then
        echo "Сервис $SERVICE_NAME успешно перезапущен." | mail -s "Service Restart:
$SERVICE_NAME" "$ADMIN_EMAIL"
    else
        echo "НЕ УДАЛОСЬ перезапустить сервис $SERVICE_NAME!" | mail -s "CRITICAL: Service
Down: $SERVICE_NAME" "$ADMIN_EMAIL"
    fi
fi
```

4. ??????? ?????? ???-??????

Находит и удаляет файлы, которые не изменялись более N дней в указанном каталоге. Помогает освободить место.

```
#!/bin/bash
#Каталог с логами
LOG_DIR="/var/log/my_app"
#Удалять файлы старше N дней
DAYS=30
find "$LOG_DIR" -type f -name "*.log" -mtime +"$DAYS" -delete
echo "Старые логи в $LOG_DIR удалены."
```

5. ??????????? ? ????? ?????????? IP

Полезно для домашних серверов с динамическим IP. Скрипт проверяет текущий IP и, если он изменился с последнего запуска, отправляет новый IP на почту.

```
#!/bin/bash
#Файл для хранения последнего известного IP
IP_FILE="/home/user/.last_ip"
ADMIN_EMAIL="myemail@example.com"
#Получаем текущий IP
```

```
CURRENT_IP=$(curl -s ifconfig.me)
if [ ! -f "$IP_FILE" ]; then
    echo "$CURRENT_IP" > "$IP_FILE"
fi
LAST_IP=$(cat "$IP_FILE")
if [ "$CURRENT_IP" != "$LAST_IP" ]; then
echo "Новый IP-адрес: $CURRENT_IP" | mail -s "IP Address Changed" "$ADMIN_EMAIL"
echo "$CURRENT_IP" > "$IP_FILE"
fi
```

6. ????? ???????? ?????? ?? SSH

Отправляет на почту список последних успешных входов по SSH. Помогает отслеживать, кто и когда заходил на сервер.

```
#!/bin/bash
ADMIN_EMAIL="security@example.com"
(echo "Последние 10 успешных входов по SSH на $(hostname):"; echo; last -n 10) | mail -s "SSH
Login Audit" "$ADMIN_EMAIL"
```

7. ???????? "??????" ????????? ???????

Собирает основную информацию о состоянии системы (нагрузка, память, диск) и сохраняет в лог-файл. Полезно для ретроспективного анализа проблем.

```
#!/bin/bash
SNAPSHOT_FILE="/var/log/system_snapshots.log"
{
    echo "====== SNAPSHOT $(date) ======";
    echo "--- Uptime & Load Average ---";
    uptime;
    echo;
    echo "--- Memory Usage (MB) ---";
    free -m;
    echo;
    echo "--- Disk Usage ---";
    df -h;
    echo "====================================\n";
} >> "$SNAPSHOT_FILE"
```

8. ???????? ?????????????? ??????

Пример: заменяет пробелы на знаки подчеркивания во всех именах файлов в текущем каталоге. Исправлено для предотвращения перезаписи существующих файлов.

```
#!/bin/bash
#Перебираем все файлы и папки в текущем каталоге
for file in *; do
    # Проверяем, есть ли пробелы в имени
    if [[ "$file" == *" "* ]]; then
        # Заменяем пробелы на подчеркивания
        new_name=$(echo "$file" | tr ' ' '_')
        # ПРОВЕРКА: не перезаписать существующий файл
        if [ ! -e "$new_name" ]; then
            mv "$file" "$new_name"
            echo "Переименовано: '$file' -> '$new_name'\n"
        else
            echo "ОШИБКА: Файл '$new_name' уже существует. Пропуск '$file'.\n"
        fi
    fi
done
```

9. ??????????? ??????????? ?? ???????????

Вместо рискованного авто-обновления, этот скрипт проверяет наличие доступных обновлений и отправляет их список администратору на почту. Это позволяет принять взвешенное решение об обновлении вручную.

```
#!/bin/bash
ADMIN_EMAIL="admin@example.com"
#Обновляем список пакетов в тихом режиме
apt-get update > /dev/null
#Получаем список пакетов, доступных для обновления
UPGRADABLE=$(apt-get -s upgrade | awk '/^Inst/ {print $2}')
if [ -n "$UPGRADABLE" ]; then
    (echo "На сервере $(hostname) доступны следующие обновления:"; echo; echo "$UPGRADABLE") |
mail -s "System Updates Available" "$ADMIN_EMAIL"
fi
```

??????????? ? ????????

Работа со службами мониторинга, логирование, отладка

Journalctl - ??????????

Journalctl — это утилита для работы с системным журналом **systemd**.
Она объединяет логи всех сервисов и ядра в единое хранилище, доступное через команды фильтрации.
В отличие от классических /var/log/syslog и /var/log/messages, journalctl позволяет быстро находить ошибки, фильтровать логи по времени, сервисам, приоритетам.

???????? ???? ?????

Простая команда для просмотра системного журнала:

```
journalctl
```

Обычно логи будут длинные, поэтому часто используют `-e` (сразу перейти к концу).

```
journalctl -e
```

???????? ????????? ??????? ? ???????? ???????

```
journalctl -f
```

Аналогично `tail -f /var/log/syslog`. Удобно использовать при отладке сервисов.

?????????? ?? ???????

Посмотреть логи конкретного сервиса:

```
journalctl -u nginx
```

В реальном времени:

```
journalctl -u nginx -f
```

?????????? ?? ???????

Примеры:

```
journalctl --since "2025-10-01" --until "2025-10-05"   # за указанный период
journalctl --since "2 hours ago"                        # за последние 2 часа
journalctl --since yesterday                            # за вчерашний день
```

?????????? ?? ??????????? (???????)

Приоритеты: **emerg, alert, crit, err, warning, notice, info, debug**.

```
journalctl -p err        # только ошибки
journalctl -p warning    # предупреждения
journalctl -p info       # информационные сообщения
```

???????? ????? ????????

```
journalctl -b      # текущая загрузка
journalctl -b -1   # предыдущая загрузка
journalctl -b -2   # позапрошлая загрузка
```

Очень полезно для анализа причин падения системы или kernel panic.

???????? ????? ????

```
journalctl -k             # сообщения ядра
journalctl -k -p err      # только ошибки ядра
```

??????????? ?????????? ?????

```
journalctl -n 50          # последние 50 строк
journalctl -n 100 -u ssh  # последние 100 строк сервиса ssh
```

????? ? ???????????? ??? ???????? ???????

```
journalctl -o short       # стандартный вывод
journalctl -o json-pretty # JSON-формат, удобно для анализа
journalctl -o cat         # только текст сообщения
```

???????? ???????? ? ??????? ????????????

- Сервис не запускается — смотрим причину:
  `journalctl -u имя_сервиса -xe`
- Отслеживание ошибок SSH-подключений:
  `journalctl -u ssh -p err -f`
- Анализ падения Nginx:
  `journalctl -u nginx --since "10 min ago"`
- Поиск "OOM-killer" (система убила процесс из-за нехватки памяти):
  `journalctl -k | grep -i oom`
- Выяснить, кто перезагружал систему:
  `journalctl _COMM=systemd-logind | grep "Power key pressed"`

???????? ?????? ? ???????

- **Ошибка:** логи не сохраняются после перезагрузки.
  Решение: включить постоянное хранение:

  ```
  sudo mkdir -p /var/log/journal
  sudo systemctl restart systemd-journald
  ```

- **Ошибка:** слишком большой размер логов.
  Решение: очистка и настройка лимита:

  ```
  sudo journalctl --vacuum-time=7d      # оставить только 7 дней
  sudo journalctl --vacuum-size=500M    # ограничить размер 500 МБ
  ```

systemctl — ??????????

Systemctl — утилита для управления **systemd** — системой инициализации и менеджером служб в Linux. Она отвечает за запуск и контроль демонов, таймеров, точек монтирования и других компонентов, описываемых *unit-файлами*.

??????

systemd управляет набором объектов: сервисами, сокетами, таймерами, монтированиями и т.д. Каждый объект описывается unit-файлом; `systemctl` — основной интерфейс для взаимодействия с этими unit'ами и с самой системой.

??? ????? unit-?????

Unit-файл — это конфигурационный файл, который описывает, как именно systemd должен управлять конкретным компонентом (сервисом, сокетом, таймером и т.д.). Обычно имеют расширение `.service`, `.socket`, `.timer`, `.mount`, `.target` и т.д.

Где хранятся:

- `/usr/lib/systemd/system/` — unit-файлы поставщика (пакетов).
- `/etc/systemd/system/` — локальные или переопределяющие unit-файлы администратора.

Пример простого unit-файла (`nginx.service`):

```
Description=A high performance web server
After=network.target
[Service]
ExecStart=/usr/sbin/nginx -g 'daemon off;'
ExecReload=/bin/kill -s HUP $MAINPID
ExecStop=/bin/kill -s QUIT $MAINPID
Restart=on-failure
[Install]
WantedBy=multi-user.target
```

Пояснение полей:

- `[Unit]` — метаинформация и зависимости (например, `After=`).
- `[Service]` — команды запуска/остановки, поведение при сбое (`Restart`), пользователь и окружение.

- `[Install]` — как unit включается в target'ы (автозапуск).

??????? ??????????

???????? ??????? ???????

Команда показывает состояние (active/failed), PID, путь к unit-файлу и последние строки логов:

```
sudo systemctl status nginx
```

?????? / ????????? / ??????????

Запуск вручную (не влияет на автозапуск при следующей загрузке):

```
sudo systemctl start nginx
```

Остановка:

```
sudo systemctl stop nginx
```

Перезапуск (полностью остановит, затем запустит):

```
sudo systemctl restart nginx
```

Перезапустить только если сервис уже запущен:

```
sudo systemctl try-restart nginx
```

Когда использовать: перезапуск обязателен при изменении конфигурации демона, но перед этим убедитесь, что unit-файл и конфигурация корректны.

??????????? ????????

enable / disable

`enable` создаёт символьные ссылки unit'а в каталогах target'ов — сервис запустится при загрузке системы. `disable` убирает эти ссылки, но не мешает ручному запуску.

```
sudo systemctl enable nginx
sudo systemctl disable nginx
systemctl is-enabled nginx
```

Важно: `enable` не запускает сервис немедленно — для этого используйте `start` или `restart`.

??????????? (mask / unmask)

Иногда нужно гарантированно запретить запуск сервиса — даже случайный или зависимый запуск. Для этого используется `mask`, который перенаправляет unit на `/dev/null`.

```
sudo systemctl mask nginx    # сервис нельзя будет запустить ни вручную, ни зависимостями
sudo systemctl unmask nginx  # снять маску
```

Когда применять `mask`:

- Нужно полностью заблокировать конфликтующий сервис (например, запрещаем start сетевого демона, если используем другой).
- Убрать возможность случайного запуска устаревших, небезопасных демонов (telnet, rsh и т.п.).
- Тестирование: временно запрещаем сервис, чтобы проверить поведение зависимостей.

?????? ? ??????????? unit-??????

После добавления/изменения unit-файла **systemd** нужно пересканировать. Это делается командой:

```
sudo systemctl daemon-reload
```

Дальше обычно перезапускают сервис:

```
sudo systemctl restart имя_сервиса
```

Если забыть `daemon-reload`, systemd будет использовать старую конфигурацию, даже если файл на диске изменён.

Targets (?????? runlevels)

В systemd понятие runlevel заменено на *targets* — наборы unit'ов, которые запускаются вместе.

- `multi-user.target` — многопользовательский режим без графики (аналог runlevel 3).
- `graphical.target` — графический режим (аналог runlevel 5).
- `rescue.target` — аварийный режим.

```
systemctl get-default               # посмотреть текущую цель
sudo systemctl set-default multi-user.target  # установить дефолтную цель
sudo systemctl isolate multi-user.target     # переключиться сразу
```

????: journalctl

systemd использует журнал (journald). Для просмотра логов сервиса применяют:

```
journalctl -u nginx         # все логи для unit
journalctl -u nginx -f      # следить в реальном времени
```

Полезно сочетать с временными фильтрами: `--since`, `--until`, и с `-p` (уровень приоритета).

???????? ??????? ? ??????

```
systemctl list-units --type=service        # активные службы
systemctl list-unit-files --type=service     # все файлы unit (включая disabled)
systemctl is-active ssh                       # быстро узнать, активен ли сервис
systemctl list-dependencies имя_сервиса       # показать зависимости
```

???????? ?????? ? ?? ???????

1) ??????? unit-????, ?? ????????? ?? ??????????

```
sudo systemctl daemon-reload
sudo systemctl restart имя_сервиса
```

Обязательная последовательность: `daemon-reload` → перезапуск/старт.

2) `Failed to start ...`

Проверяем статус и логи:

```
sudo systemctl status имя_сервиса
journalctl -xeu имя_сервиса
```

Частые причины: ошибка в пути в `ExecStart`, неверные права, конфликт портов, отсутствующие зависимости.

3) ?????? ?????????? ???????, ?? ?? ???????? ??? ????????

```
systemctl is-enabled имя_сервиса
sudo systemctl enable имя_сервиса
```

Проверьте также, не мешают ли target'ы или маскировка, и нет ли ошибок в `[Install]` блока unit-файла (например, `WantedBy=`).

4) "Unit not found"

Скорее всего unit-файл отсутствует или не установлен:

```
systemctl list-unit-files | grep имя_сервиса
```

Возможные причины: пакет не установлен, unit называется иначе, или unit расположен в нестандартном каталоге.

5) Unit ??????? ??? `masked`

```
systemctl is-enabled имя_сервиса   # покажет "masked"
sudo systemctl unmask имя_сервиса
```

???? unit-?????? (??????)

- `.service` — сервис/демон.
- `.socket` — сокет-активация; может запускать .service при подключении.
- `.timer` — периодические/отложенные задания (альтернатива cron).
- `.mount` — описание точки монтирования.
- `.target` — агрегатор unit'ов (аналог runlevel).

???????? ???????????? (best practices)

- Редактируйте unit-файлы в `/etc/systemd/system/` (локальные переопределения не затрутся при обновлении пакета).
- После правок — всегда `daemon-reload`, затем `restart` или `try-restart`.
- Используйте `Restart=on-failure` аккуратно; логируйте причины падений (через `StandardOutput`/`StandardError` или journald).
- Для временной блокировки сервиса применяйте `mask`. Для обычного отключения — `disable`.

top ? htop

??? ????? top ? htop?

`top` и `htop` — консольные утилиты для мониторинга процессов в Linux. С их помощью можно в реальном времени отслеживать загрузку процессора, использование памяти, swap, uptime и управлять процессами. Разница: **top** есть по умолчанию в любой системе, **htop** удобнее и нагляднее, но может требовать установки.

????????? htop

```
sudo apt install htop   # Ubuntu/Debian
sudo yum install htop   # CentOS/RHEL
sudo dnf install htop   # Fedora
```

??????

```
top
htop
```

????????? ? ???????

- **top:** `P` — сортировка по CPU, `M` — по памяти, `k` — завершить процесс, `q` — выход.
- **htop:** стрелки — перемещение, `F6` — сортировка, `F9` — убить процесс, `F10` — выход.

??????? ????????????

```
top -o %MEM             # отсортировать процессы по памяти
htop -u www-data        # показать процессы пользователя www-data
top -n 1 -b > /tmp/top.txt  # вывести результат в файл (batch mode)
```

???????? ????????

- Сервер "подвисает" → проверить процессы с высокой загрузкой CPU.
- Падает сервис → отследить, не уходит ли в swap (**htop** показывает swap в цветах).
- Подозрительная активность → сортировать по PID/памяти и найти виновника.

???????? ?????? ? ???????

- **Ошибка:** htop не найден.
  Решение: установить через пакетный менеджер (см. выше).
- **Ошибка:** невозможно убить процесс.
  Решение: использовать root-права:

  ```
  sudo kill -9 PID
  ```

netstat, ss ? ip

??? ????? netstat, ss ? ip?

Эти утилиты используются для диагностики сетевых подключений и маршрутов:
- **netstat** — старая утилита (в пакете net-tools).
- **ss** — современная замена netstat.
- **ip** — инструмент для управления сетевыми интерфейсами, IP-адресами и маршрутами.

??????? ???????

```
ss -tuln                # список слушающих портов
ss -tulpn | grep 5432   # найти процесс на порту 5432
ip a                    # список интерфейсов и адресов
ip r                    # таблица маршрутов
ip link show up         # активные интерфейсы
netstat -anp            # (устар.) список соединений с PID
```

??????????

- Проверить, слушает ли nginx порт 80 → `ss -tuln | grep :80`
- Кто слушает порт 22 → `ss -tulpn | grep :22`
- Какой маршрут до 8.8.8.8 → `ip route get 8.8.8.8`
- Сбросить интерфейс → `sudo ip link set eth0 down && sudo ip link set eth0 up`

???????? ??????

- **Ошибка:** netstat не найден.
  Решение: установить пакет `net-tools`.
- **Ошибка:** сервис не слушает порт.
  Решение: проверить конфиг сервиса и firewall (`ufw`/`iptables`).

Netplan

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp6s18: #название интерфейса
      dhcp4: no
      dhcp6: no
      addresses:
        - 10.177.178.47/24 #IP
      nameservers:
        addresses:
          - 10.177.178.42 #DNS
      routes:
        - to: default
          via: 10.177.178.1 #Gateway
```

Конфигурация для статики. Меняем по пути /etc/netplan/имя_конфигурации.yaml.
После изменения применяем команду:

```
sudo netplan apply
```