Заметки по работе с системными командами и консольными утилитами **Linux**.

Навигация:

    - fd-fzf
- findutils
    - find
    - exec
    - locate
    - xargs
- fincore
    - lspage
- bashrc
    - oh-my-bash
    - fzf
    - fzf-obc
    - hstr
    - mcfly
- compgen
- cron
- systemctl
    - systemctl-tui
    - unit
- journalctl
- dmesg
- hardware
- sysctl
- limits
- quota
- Bearstech
    - pussh
    - quickbench
- fetch
- networkmanager
- wireless
- networking
- netplan
- ip
    - net-tools
    - networkd
- ss
- dns
    - resolv
    - resolved
    - dig
    - mtr
    - doggo
- vnstat
- netcat
    - socket api
    - socket proxy
- proxy
- nmap
    - masscan
    - rustscan
    - tcp
- tcpdump
- tshark
- ping
    - fping
    - netping

- firewall
  - ufw
  - show
  - firewalld
  - iptables
  - nftables
- ssh
  - keygen
  - x11
  - scp
  - sshpass
- sudoers
- strace
- apt
- snap
- dpkg
- ntp
  - time
  - language
  - timesyncd
  - ntpd
- top
  - htop
  - bpytop
  - atop
  - iftop
  - iotop
  - top other
- ps
  - kill
  - procs
- jobs
  - nohup
  - task-spooler
- mem
- lsof
  - descriptor
- vmstat
- sysstat
  - iostat
  - mpstat
  - pidstat
- stress
  - stress-ng
- smart
  - smartmontools
  - sensors
  - badblocks
  - hdparm
- disk
  - parted
  - fdisk
  - sfdisk
  - swap
- lvm
- md
- tgt

- dd
  - backup
  - iso
  - rdiff
- users
  - passwd
  - chage
  - id
  - usermod
  - profile
  - bashrc
  - useradd
  - adduser
- chmod
  - chown
  - groups
  - usermod
- domain
  - realmd
  - sssd
- syslog
  - server
  - client
  - zabbix-agent
  - ommail
- logrotate
- log
- smb
  - cifs
  - samba
  - client cifs
  - client samba-client
  - recycle
- nfs
  - server
  - client
- ftp
  - ftp client
  - ftps
- rsync
- apache
  - api server
  - status
  - webdav
- haproxy
- keepalive

bash

- Переменные

`text="(ip a)"` передает текст

`echo $text`

`ipaddr=$(ip a)` передает вывод команды

`echo $ipaddr`

`echo '$ipaddr'` в одинарных кавычках не происходит подстановка переменных

`var=$((5+5))`

`echo $var`

`read -p "Enter: " enter` ручной ввод переменной

`echo $enter`

`read -s -p "Enter password: " pass` ввод пароля

`echo $pass`

`echo -e "text\ntext"` экранирование

`echo -e "# comment\nparam comment" > ~/test.txt` записать в файл

`cat ~/test.txt | grep -v "^#"` прочитать без комментариев в начале строки

```
original_value="Это длинная строка, которую нужно сократить до 50 символов."
shortened_value="${original_value:0:50}" обрезаем до 50 символов
```

`true ; echo $?` код возврата 0 (успех)

`false ; echo $?` код возврата 1 (ошибка)

- Массивы

`range={1..254}` создать срез от 1 до 254

`array=(1 2 3 4 5)` создать массив

`array=($(ls /))` передает вывод команды `$(command)` разделенных через пробел

`echo ${array[@]}` отобразить содержимое всего массива `@/*`

`echo ${array[0]}` отобразить первый индекс в массиве

`echo ${array[-1]}` отобразить последний индекс

`echo ${array[@]:1:3}` вывести 3 элемента (срез)

`echo ${#array[@]}` отобразить кол-во (`#`) элементов в массиве

`echo ${#array[0]}` отобразить длинну (`#`) первого элемента в массиве

`array[1]="22"` изменить значение по номеру индекса

```
declare -A dict=(
    ["key 1"]=1
    ["key 2"]="text"
)
echo ${dict[key 1]}
echo ${dict[key 2]}
```

- Цикл for

```
for ((i=1; i <= 10; i++)); do
    echo $i
done

array=($(ls /))
for arr in ${array[@]}; do
    echo $arr
done
```

`break` прерывает цикл

`continue` прерывает текущую интерацию в цикле и переходит к следующей

```
array=(1 2 3 4 5)
for var in ${array[@]}; do
    if [ $var -gt 4 ]; then
        break
    elif [ $var -gt 3 ]; then
        echo "Last number: $var"
        continue
    fi
    echo "Number: $var"
done
```

- Цикл while

```
p=1
while [ $p -le 101 ]; do
    # если условие истинно, выполнять цикл в блоке do, пока не станет ложным
    echo "Значение переменной: $p"
    # ((p++)) # увеличить на +1
    # p=$(($p+10)) # прибавлять +10
    p+=0 # добавить текст в конец переменной
done
```

- Построчная передача вывода через pipe

```
num=0
ps | sed 1d | while read line; do
    ((num++)) # ((num+=1))
    echo "Line $num : $line"
done
```

- Условия

`if []` если
`then` условие истинно
`elif []` дополнительное условие
`then` дополнительное условие истинно
`else` условие ложно
`fi` больше нет условий

`-z` строка пуста
`-n` строка не пуста
`=, (==)` строки равны
`!=` строки неравны
`-eq` равно
`-ne` неравно
`-lt, (<)` меньше
`-le, (<=)` меньше или равно
`-gt, (>)` больше
`-ge, (>=)` больше или равно
`!` отрицание логического выражения
`-a, (&&)` логическое «и» (первая команда исполняется всегда, вторая — только в случае успешного завершения первой)
`-o, (||)` логическое «или» (первая команда исполняется всегда, вторая — только в случае неудачного завершения первой)

```
if [[ -z "$variable" ]]; then
    echo "Переменная пустая"
else
    echo "Переменная не пустая"
fi
```

- Функции

```
function calc {
    if [ $2 = "+" ]
        then
        echo $(( $1 + $3 ))
    elif [ $2 = "-" ]
        then
        echo $(( $1 - $3 ))
    fi
}

calc 3 + 2
calc 3 - 2
```

- Параметры

```
nano script.sh
```

```
#!/bin/bash
if [ -n "$1" -a "$2" ]; then
    echo Имя исполняемого файла: $0
    echo Первый переданный параметр: $1
    echo Второй переданный параметр $2
    echo Кол-во переданных параметров: $#
    echo Значение последнего переданного параметра: ${!#}
    echo Массив: $@
else
    echo "Параметры не заданы"
fi
```

`chmod +x script.sh` сделать скрипт исполняемым
`bash script.sh 1 2 3 4 5` передать параметры в скрипт

`-e file` проверяет, существует ли файл
`-d file` проверяет, существует ли файл, и является ли он директорией
`-f file` проверяет, существует ли файл, и является ли он файлом
`-r file` проверяет, существует ли файл, и доступен ли он для чтения
`-w file` проверяет, существует ли файл, и доступен ли он для записи
`-x file` проверяет, существует ли файл, и является ли он исполняемым
`-s file` проверяет, существует ли файл, и не является ли он пустым

```
# Получить список директорий и исполняемых файлов в дочерних директориях
path="/etc/*"
for folder in $path; do
    echo "$folder:"
    for file in $folder/*; do
        if [ -x $file ]; then
            echo "- $file"
        fi
    done
done
```

- case

```
read -rsn1 key
case $key in
  "1")
    echo выполнить действия, если $key равно 1 ;;
  "2")
    echo выполнить действия, если $key равно 2 ;;
  *)
    echo выполнить действия по умолчанию, если значение $key не соответствует ни одному условию
  ;;
esac
```

filesystem

`file Console-Performance.sh` узнать тип файла (текстовый, исполняемый файл, архив или другой)
`stat Console-Performance.sh` узнать размер файла, количество блоков, занятых файлом на диске, количество жестких ссылок, права доступа и временные метки
`pwd` текущая директория
`ls -lh *` отобразить содержимое каждого подкаталога отдельно
`ls -lhaF` отобразить скрытые директории (-a) с точкой и выделит директории (/)
`which top` узнать путь до исполняемого файла
`stat $(which top)` узнать дату последнего доступа к файлу
`cat -n /etc/passwd` просмотр содержимого файла с отображением номеров строк
`mkdir` создать директорию
`mktemp -d` создать временный файл/каталог (-d)
`touch -t 202106222200.15 test.file` создать файл и указать дату создания
`cp test.file test.file2` копировать файла/каталог
`mv test.file2 test.file3` переименовать/переместить файл/каталог
`rm -r test.file` удалить каталог с файлами (-r)

ln

`echo "test" > testfile`
`ln /test/testfile /test/testlink` создать жестку (hard) ссылку, которая указывает на один и тот же inode, т.е. они делят одно и то же физическое местоположение на диске
`rm testfile` при удалении одного из файлов не приводит к удалению содержимого, пока существует хотя бы одна жесткая ссылка
`ln -s /test/testfile /test/testlink` создать символическую (-s - soft) ссылку, которая ссылается на файл testfile
`echo "test" >> testfile` при добавлении в оригинальный файл, все изменения будут отражены в testlink
`rm testfile` при удалении исходного файла у ссылки будет ошибка (No such file or directory)

zip

`rar a test.rar filename filename2` создать архив test.rar и добавить туда два файла (файлы копируются в архив)
`unrar x test.rar` разархивировать
`zip -r test.zip filename` архивировать (файлы копируются в архив)
`unzip test.zip` разархивировать
`bzip2 filename` архивировать в filename.bz2 (файлы перепещаются в архив)
`bunzip2 filename.bz2` разархивировать
`gzip filename` архивировать в filename.gz (файлы перепещаются в архив)
`tar --totals -cvf archive.tar file1 file2 file3` архивировать три файла
`wget https://github.com/librespeed/speedtest-cli/releases/download/v1.0.10/librespeed-cli_1.0.10_linux_amd64.tar.gz` загрузить архив
`gunzip librespeed-cli_1.0.10_linux_amd64.tar.gz` извлечь из gz в tar
`tar -tf librespeed-cli_1.0.10_linux_amd64.tar` отобразить содержимое архива
`tar -xvf librespeed-cli_1.0.10_linux_amd64.tar` разархивировать
`./librespeed-cli --help`
`./librespeed-cli --json`

gpg

`gpg -c filename` зашифровать данные

`gpg filename.gpg` расшифровать данные

`gpg --gen-key` создавать пару ключей (публичный и приватный ключи)

`gpg --export -a 'User Name' > publickey.asc` экспорт публичного ключа

`gpg --import publickey.asc` импорт на второй стороне

`gpg --encrypt --recipient 'Recipient Name' filename` зашифровать данные с использованием публичного ключа получателя, только владелец приватного ключа сможет расшифровать эти данные

`gpg --decrypt encryptedfile.gpg` расшифровать данные можно с помощью приватного ключа

`gpg --sign filename` подписывать данные с использованием приватного ключа для подтверждения их подлинности и целостности

`gpg --verify signedfile.gpg` проверка подписи с использованием публичного ключа отправителя

api

curl

`curl ifconfig.me` узнать внешний ip

`curl -v telnet://192.168.3.100:22` првоерить доступность порта и отобразить кому он принадлежит

`curl -s -o /dev/null http://google.com` подавить весь вывод (статистику --silent и --output)

`curl -s -o /dev/null --show-error --fail http://google.com` оставить вывод ошибок

`curl http://192.168.3.101:8081/api/ --connect-timeout 5` задать timeout ожидания ответа в секундах

`curl -IL https://github.com/Lifailon/hwstat/archive/refs/tags/hwstat-0.0.8.zip` получить информацию о файле перед скачиванием (--head/--location)

`curl -O https://raw.githubusercontent.com/Lifailon/hwstat/rsa/hwstat.sh` скачать файл

`curl -o /tmp/hwstat.sh https://raw.githubusercontent.com/Lifailon/hwstat/rsa/hwstat.sh` указать путь

`curl -Ik https://192.168.3.104:9443/` игнорировать ошибку самоподписанного сертификата SSL (--insecure)

`curl -u <user:password> https://test.com/endpoint` авторизация

`curl -x "http://Proxy:Proxy@192.168.3.100:9090" "https://kinozal.tv/rss.xml"` использовать Proxy-сервер

`curl --insecure --ssl-reqd "smtps://smtp.yandex.ru" --mail-from "src@yandex.ru" --mail-rcpt "dst@yandex.ru" --user "src@yandex.ru" --upload-file ou` отправка email через SMTPS (SMTP over SSL/TLS) сервер

influxdb

```
ip="192.168.3.104"
db="dbash"
table="icmp_metrics_table"
server="google.com"
host=$(hostname)
date=$(echo $EPOCHREALTIME | sed -E "s/\..+//")"000000000"
ping=$(ping $server -c 1)
loss=$(printf "%s\n" "${ping[@]}" | grep -Eo "[0-9]+%" | sed "s/%//")
if (( $(echo "$loss != 100" | bc) )); then
    status="true"
    rtt=$(printf "%s\n" "${ping[@]}" | grep rtt | awk -F"/" '{print $5}')
else
    status="false"
    rtt="0"
fi
curl -i -XPOST "http://$ip:8086/write?db=$db" --data-binary "$table,host=$host,server=$server status=$status,rtt=$rtt $date"
```

wget

`wget --spider https://download.nextcloud.com/server/releases/nextcloud-21.0.1.tar.bz2` проверить (--spider) работоспособность URL и узнать размер файла (Length)

`wget -O nextcloud.tar.bz2 https://download.nextcloud.com/server/releases/nextcloud-21.0.1.tar.bz2` скачать с указанным именем (-O)

`wget -P /tmp https://download.nextcloud.com/server/releases/nextcloud-21.0.1.tar.bz2` скачать в указанную директорию (-P)

`wget -b -o ~/wget.log https://download.nextcloud.com/server/releases/nextcloud-21.0.1.tar.bz2` загрузить в фоновом режиме (-b) и записать вывод в лог-файл (-o)

curlie

`curl -sS https://webinstall.dev/curlie | bash` альтернатива curl и httpie (https://github.com/rs/curlie)

`curlie get https://jsonplaceholder.typicode.com/posts` возвращает заголовки ответа и отформатированный вывод JSON

`curlie get https://jsonplaceholder.typicode.com/posts/1`

`curlie get https://jsonplaceholder.typicode.com/posts -H "Authorization: Bearer YOUR_TOKEN"`

`curlie post https://jsonplaceholder.typicode.com/posts -d '{"title": "foo", "body": "bar", "userId": 1}'`

httpie

`sudo snap install httpie` HTTP-клиент командной строки (https://github.com/httpie/cli)

`https httpie.io/hello`

`https POST pie.dev/post X-API-Token:123 name=John`

json

jq

`apt install jq` установить jq (https://github.com/jqlang/jq)

`nodes=$(curl -s -H "Accept: application/json" https://check-host.net/nodes/ips)` получить список node

`echo $nodes | jq` обработка входных данных командой jq (вывод отображается в правильно структурированном формате, а все элементы подсвечиваются соответствующим цветом)

`echo $nodes | jq '.nodes | length'` количество дочерних объектов в блоке node[]

`echo $nodes | jq -r .nodes[1]` получить значение второго объекта массива в формате raw string (not JSON)

`echo $nodes | jq -r .nodes[-1]` получить значение последнего объекта массива

`hosts=$(curl -s -H "Accept: application/json" https://check-host.net/nodes/hosts)` получить список всех хостов

`echo $hosts | jq -r '.nodes | to_entries[].key'` получить список всех вложенных ключей (адреса хостов) из объека (не является массивом)

`echo $hosts | jq -r '.nodes | to_entries[].value'` получить только значения всех вложанных ключей

`echo $hosts | jq '.nodes."bg1.node.check-host.net"'` получить значение дочернего ключа nodes по имени

`echo $hosts | jq '.nodes | [.[]] | last'` преобразовать отдельные объекты внутри nodes в массив, и передать полученный вывод в функцию last для получения значений последнего объекта

`echo $hosts | jq '.nodes | to_entries[].value.location[0] == "ru"'` проверить каждый элемент объекта в условии на true/false (вернет массив)

`echo $hosts | jq '.nodes | to_entries[] | {Host: .key, Country: .value.location[1], City: .value.location[2]}'` получить данные key-value из объекта nodes и пересобрать массив с новыми значениями ключей

`echo $hosts | jq -r '.nodes | to_entries[] | "\(.key) (\(.value.location[1]), \(.value.location[2]))"'` собрать массив строки из содержимого ключей

`var="-" && echo $hosts | jq --arg v "$var" -r '.nodes | to_entries[] | "\(.key) \($v) \(.value.location[1]) \($v) \(.value.location[2])"'` передать внешнюю переменную, которая будет использоваться внутри запроса

`echo $hosts | jq -r '.nodes | to_entries[] | select(.value.location[0] == "ru") | .key'` произвести фильтрацию (select), что бы получить только нужные объекты

`echo $hosts | jq '.nodes | to_entries[] | select(.value.location[0] != "ru") | .key'` вывести объекты, которые не равны значению

`echo $hosts | jq '.nodes | length'` вывести общее количество объектов

`echo $hosts | jq '.nodes | to_entries | map(select(.value.location[0] != "ru")) | length'` создать массив функцией map() (объеденяет отдельные объекты {}{} группируются в один массив [{},{}]) только из тех объектов, которые соответствуют условию select() и вывести количество найденных объектов

`echo $hosts | jq -r '.nodes | to_entries[] | select(.value.location[0] == "ru" or .value.location[0] == "tr") | .key'` проверить два условия через or или and (для проверяемого типа данных int кавычки не используются)

`echo $hosts | jq -r '.nodes | to_entries[] | select(.key | index("jp")) | .key'` вывести список хостов региона Japan, которые в названии ключа содержат ключевое слово jp (частичное совпадение в значении)

```
host="yandex.ru"
protocol="ping"
host="yandex.ru:443"
protocol="tcp" # udp/http/dns
# Забрать id для получения результатов
check_id=$(curl -s -H "Accept: application/json" "https://check-host.net/check-$protocol?host=$host&max_nodes=3" | jq -r .request_id)
# Функция получения результатов проверки по id
function check-result {
    curl -s -H "Accept: application/json" https://check-host.net/check-result/$1 | jq .
}
# Получить суммарное количество хостов, с которых производится проверка
hosts_length=$(check-result $check_id | jq length)
while true; do
    check_result=$(check-result $check_id)
    # Забираем результат и проверем, что содержимое всех проверок не равны null
    check_values_not_null=$(echo $check_result | jq -e 'to_entries | map(select(.value != null)) | length')
    if [[ $check_values_not_null == $hosts_length ]]; then
        echo $check_result | jq
        break
    fi
    sleep 1
done
```

```
echo '{"iso": [{"name": "Ubuntu", "size": 4253212899}, {"name": "Debian", "size": 3221225472}]}' | jq '.iso[] | {name: .name, size: (.size / 1024 /
```
получить ГБ из байт и округлить вывод до 2 символом после запятой

```
echo '{"iso": [{"name": "Ubuntu", "progress": 0.333}]}' | jq '.iso[] | {name: .name, progress: (.progress * 100 | floor / 100 * 100 | tostring + "
```
получить процент из дробной части (33%)

```
echo '[{"name": "Ubuntu", "added_on": 1625072400}, {"name": "Debian", "added_on": 1625158800}]' | jq '.[] | {name: .name, date: (.added_on + 3 * 36
```
получить дату

netcheck

```
sudo curl -s https://raw.githubusercontent.com/Lifailon/Check-Host/rsa/netcheck/netcheck.sh -o /usr/bin/netcheck
sudo chmod +x /usr/bin/netcheck
```

```
netcheck -t ping yandex.ru
netcheck -n
netcheck -t ping yandex.ru ru1.node.check-host.net
netcheck -t dns yandex.ru
netcheck -t http yandex.ru:443 5
netcheck -t tcp yandex.ru:443
```

jc

`apt install jc` установить jc (https://github.com/kellyjonbrazil/jc) для преобразования вывода популярных инструментов командной строки, типов файлов и общих строк в JSON, YAML или словари Python, что позволяет передавать вывод в инструменты, такие как jq

```
dig google.com | jc --dig
dig example.com | jc --dig | jq -r '.[].answer[].data'
jc --pretty /proc/meminfo
systemctl list-units --all --plain --no-legend --no-pager | jc --systemctl -p
```

brew

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
echo 'eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"' >> ~/.profile`
source ~/.profile
brew --version
```

fx

`brew install fx || snap install fx` установить fx (https://github.com/antonmedv/fx) TUI интерфейс для JSON на GoLang

```
hosts=$(curl -s -H "Accept: application/json" https://check-host.net/nodes/hosts)
```

`echo $hosts | fx` доступна навигация с раскрытием блоков и отображает ключи доступа для jq

`source <(fx --comp bash)` добавить autocomplete в интерпритатор bash

```
echo $hosts > hosts.json
```

`fx hosts.json .nodes .\[\"ru1.node.check-host.net\"\] .ip` происходит автоматический вывод ключей и подстановка

jid

`brew install jid` установить jid (https://github.com/simeji/jid) для интерактивной фильтрации JSON данных с использованием автозавершения на GoLang

```
echo '{"info":{"date":"2016-10-23","version":1.0},"users":[{"name":"simeji","uri":"https://github.com/simeji","id":1},{"name":"simeji2","uri":"http
```

.users[1].uri

jqp

`brew install noahgorstein/tap/jqp` установить jqp (https://github.com/noahgorstein/jqp) TUI интерфейс для отображения jq запросов на GoLang

`curl -s https://api.github.com/repos/Lifailon/PS-Commands/contents | jqp` слева отображается исходный файл, справа отфильтрованный вывод

`curl -s https://check-host.net/nodes/hosts | jqp` # пример для фильтрации: `.nodes | to_entries[] | select(.value.location[0] == "ru") | .key`

xmllint

```
apt-get install libxml2-utils || snap install libxml2 || brew install libxml2
curl -s https://kinozal.tv/rss.xml -x kinozal:proxy@192.168.3.100:9090 | xmllint --xpath '//rss/channel/item/link/text()' -
curl -s https://kinozal.tv/rss.xml -x kinozal:proxy@192.168.3.100:9090 | xmllint --xpath '//rss/channel/item[1]/link/text()' -
```

dasel

`brew install dasel` установить dasel (https://github.com/TomWright/dasel) для обработки JSON, YAML, TOML, XML и CSV (поддерживает преобразование между форматами) на GoLang

```
echo '{"name": "Tom"}' | dasel -r json 'name'
```

`echo '{"name": "Tom"}' | dasel -r json -w yaml` конвертировать json в yaml

`echo '{"name": "Tom"}' | dasel -r json -w xml` конвертировать json в xml

`echo '{"name": "Tom"}' | dasel put -r json -t string -v 'contact@tomwright.me' 'email'` добавить свойство

`echo '{"email": "contact@tomwright.me","name": "Tom"}' | dasel delete -r json '.email'` удалить свойство

```
tee users.json <<EOF
{
    "users": [
      {
        "name": "Иван Иванов",
        "email": "ivan.ivanov@example.com"
      },
      {
        "name": "Мария Петрова",
        "email": "maria.petrova@example.com"
      }
    ]
}
EOF
```

```
dasel -f users.json -r json ".users.[0].email"
```

```
tee users.yaml <<EOF
users:
  - name: Иван Иванов
    email: ivan.ivanov@example.com
  - name: Мария Петрова
    email: maria.petrova@example.com
EOF

dasel -f users.yaml -r yaml ".users.[1].email"

tee users.toml <<EOF
[[users]]
name = "Иван Иванов"
email = "ivan.ivanov@example.com"
[[users]]
name = "Мария Петрова"
email = "maria.petrova@example.com"
EOF

dasel -f users.toml -r toml ".users.[1].email"

tee users.xml <<EOF
<users>
  <user>
    <name>Иван Иванов</name>
    <email>ivan.ivanov@example.com</email>
  </user>
  <user>
    <name>Мария Петрова</name>
    <email>maria.petrova@example.com</email>
  </user>
</users>
EOF

dasel -f users.xml -r xml ".users.user.[0].email"
```

xq

`apt-get install xq || brew install xq` установить xq (https://github.com/sibprogrammer/xq) для XML и HTML на GoLang

`curl -s https://kinozal.tv/rss.xml -x kinozal:proxy@192.168.3.100:9090 | xq -nx /rss/channel/item` вывод содержимого дочерних элементов с тегами

`curl -s https://kinozal.tv/rss.xml -x kinozal:proxy@192.168.3.100:9090 | xq -x /rss/channel/item/link` вывести только содержимое (массив ссылок)

`curl -s https://kinozal.tv -x kinozal:proxy@192.168.3.100:9090 | xq -nq "head"` вывести блок head целиком (с тегами)

`curl -s https://kinozal.tv -x kinozal:proxy@192.168.3.100:9090 | xq -q "head"` вывести только текст из дочерних элементов выбранного тега (содержимое title)

`curl -s https://kinozal.tv/browse.php?s=the+rookie -x kinozal:proxy@192.168.3.100:9090 | xq -nq "body > div > div > div > div > table > tbody > tr`

`curl -s -X POST -u "Login:Password" "http://localhost:9091/transmission/rpc" | xq -q a -a href` забрать X-Transmission-Session-Id для дальнейших запросов к API (обратиться к тэгу a и атрибуту href)

htmlq

`brew install htmlq` установить htmlq (https://github.com/mgdm/htmlq) like jq for HTML

`curl -s https://kinozal.tv/browse.php?s=the+rookie -x kinozal:proxy@192.168.3.100:9090 | htmlq table tr td a -t` получить содержимое таблицы (вывести только текст содержимого)

`curl -s https://kinozal.tv/browse.php?s=the+rookie -x kinozal:proxy@192.168.3.100:9090 | htmlq table tr td a -a href` получить только ссылки

`curl -s -X POST -u "Login:Password" "http://localhost:9091/transmission/rpc" | htmlq a -a href` забрать X-Transmission-Session-Id для дальнейших запросов к API (обратиться к тэгу a и атрибуту href)

yq

`snap install yq` установить yq (https://github.com/mikefarah/yq) для YAML, JSON, XML, CSV и TOML

`cat /etc/netplan/*.yaml | yq .network.ethernets` список адаптеров netplan

`cat /etc/netplan/*.yaml | yq .network.ethernets.eth0.nameservers.addresses[]` вывести массив dns адресов, настроенные на адаптере

`curl -s https://kinozal.tv/rss.xml -x kinozal:proxy@192.168.3.100:9090 | yq -p xml .rss.channel.item[1].link` вывести ссылку из первого элемента

`curl -s https://raw.githubusercontent.com/JingWangTW/dark-theme-editor/main/hugo.toml | yq -p toml .params.footer.socialLink` прочитать конфигурацию Hugo

yamllint

`apt install yamllint` установить yamllint (https://github.com/adrienverge/yamllint) для проверки синтаксических ошибки YAML-файла

`yamllint /etc/netplan/*.yaml`

jsonlint

`apt-get install -y nodejs` установить Node.js

`npm install jsonlint -g` установить jsonlint (https://github.com/zaach/jsonlint) для проверки синтаксических ошибок JSON

```
echo '{"name":"example","value":"test",}' | jsonlint
echo '{"name":"example","value":"test"}' | jsonlint
```

csv

`brew install csvlens` установить csvlens (https://github.com/YS-L/csvlens) для взаимодестия в csv через TUR на Rust

```
pwsh -Command "Get-Process | ConvertTo-Csv | Out-File process.csv"
csvlens process.csv
```

sttr

`snap install sttr` установить sttr (https://github.com/abhimanyu003/sttr) для конвертации и работы данными на GoLang

`curl -s curl -s -H "Accept: application/json" https://check-host.net/nodes/hosts | sttr json-yaml` конвертировать JSON в YAML

`cat /etc/netplan/*.yaml | sttr yaml-json | jq` конвертировать YAML в JSON

`curl -s https://raw.githubusercontent.com/Lifailon/hwstat/rsa/README.md | sttr markdown-html` конвертировать Markdown в HTML

`echo "test" | sttr hex-encode` кодировать в HEX формат

`echo "74657374" | sttr hex-decode` декодировать HEX

`echo "Test" | sttr upper` поднять регистр (TEST)

`echo "Test" | sttr lower` опустить регистр (test)

`echo -e "test1\ntest1\ntest2" | sttr unique-lines` получить уникальные строки

`echo -e "a\nz\nb" | sttr sort-lines` сортировать строки по алфавиту

`echo -e "test1 \ntest2" | sttr remove-newlines` удалить новые строки

`echo -e "test1\ntest2" | sttr count-chars` посчитать количество символов

`echo -e "test1\ntest2" | sttr count-lines` посчитать количество строк

grep

`cat /var/log/auth.log | grep sshd` логи всех SSH-подключений

`cat /etc/passwd | grep -w sys` поиск целого слова, окруженное пробелами (`-w`)

`cat /etc/ssh/sshd_config | grep -win port` не учитывать регистр (`-i`) и отобразить номера строк (`-n`)

`ss -n | grep -P ":22|:80|:443|:8080"` искать по нескольким шаблонам, использовать Regex (`-E`)

`ss -n | grep -Pc ":22|:80"` вывести кол-во (`--count`) совпадений

`ss -n | grep "192.168.3...:"` поиск любых двух символов (`.`)

`ss -n | grep "192.168.3.*:"` поиск любого кол-ва (`*`)

`cat /etc/ssh/sshd_config | grep -v "#"` вывести значения, не подходящие под критерии поиска (`-v`)

`cat /etc/zabbix/zabbix_agentd.conf | grep -v "^#"` отсеить только в начале строки (`^`)

`cat /etc/zabbix/zabbix_agentd.conf | grep "=$"` найти строки, которые кончаются `$` на символ `=` (получить все параметры)

`cat /etc/zabbix/zabbix_agentd.conf | grep -Pv "^$|^#"` удалить пустые строки `^$` и комментарии (`^#`)

`cat /etc/zabbix/zabbix_agentd.conf | grep -E "#+{5}"` регулярное выражение (`-E`), где последний символ `#` повторяется 5 или более раз

`echo -e "Test\ntest\n123-45" | grep -E "[a-zA-Z\-]"` искать только текст (где есть буквы и тире)

`echo 'test<version>1.2.3</version>test' | grep -P -o "(?<=<version>).*(?=</version>)"` найти неизвестное значение (`.*`) между известными и вывести только найденное (`-o`)

`echo "test<version>3.6.4</version>test" | grep -Eo '[0-9.]+'` найти любую цифру и точку на конце, которые повторяются любое кол-во раз подряд

`echo $(lshw -class bus) | grep -P -o "(?<=Motherboard product: ).*(?=serial)"` с применение группировки (`-P`)

`zabbix_path=$(systemctl status zabbix-agent | grep -Po "(?<=-c ).*(?=.conf)" | sed "s/$/.conf/")` забрать путь до конфигурационного файла Zabbix агента

`cat $zabbix_path | grep -E "^Server=|^ServerActive="` найти имя сервера

`cat $zabbix_path | grep -Po "(?<=^Server=).+"` вывести только имя сервера

`resolvectl | grep "DNS Servers" -m 1` напечатать только первое совпадение (`-m int`)

`networkctl status | grep -A 3 "DNS:"` найти строку и напечатать три строки после нее (`-A`)

`networkctl status | grep -B 3 "DNS:"` найти строку и напечатать три строки до нее (`-B`)

`networkctl status | grep -C 1 "DNS:"` найти строку и напечатать одну строки до нее и одну после (`-C`)

`resolvectl | grep -Ex ".+DNS Servers:.+"` вывести строки с точным совпадение (`-x/like`), сопоставлять только целые строки

`if echo "GET" | grep -Eq "^GET"; then echo da; else echo net; fi` подавлять вывод (`-q`) для проверки условия

`curl https://api.github.com/repos/PowerShell/PowerShell/releases/latest | grep -Eom 1 "https://.+.deb"` забрать только первый подходящий под поиск

ripgrep

`apt-get install ripgrep` установить ripgrep, аналог grep на Rust

`cat /var/log/auth.log | rg sshd` вывести журнал логов аудентификации фильтрацией по названию

`cat /var/log/auth.log | rg "Accepted password for \w+ from \d+\.\d+\.\d+\.\d+"` вывести строки, где указано `Accepted password for` , далее любое слово (имя пользователя) и IP-адрес в формате x.x.x.x

`cat /var/log/auth.log | rg "user \w+\(uid=\d+\)"` вывести строки с текстом user, затем имя пользователя (любое слово), и далее `uid` с числовым значением в скобках

`cat /var/log/auth.log | rg "192\.168\.\d+\.\d+"` вывести строки, где первые два октета соответствуют `192.168`

`cat /var/log/auth.log | rg "sshd\[\d+\]: .* port \d+"` вывести строки, содержащие sshd с идентификатором процесса (например, `sshd[4188420]`), а затем текст `port` и номер порта

`cat /var/log/auth.log | rg "\b12:\d{2}:\d{2}\b"` фильтрация по времени за последние 12 часов (время начинается с `12:` , затем две цифры для минут и две для секунд)

rga

`apt install ripgrep fzf pandoc ffmpeg poppler-utils` установить зависимости

`brew install rga` установить ripgrep-all и `rga-fzf` - инструмент для быстрого поиска в файлах по содержимому

`rga-fzf token` поиск ключевого слова `token` во всех файлах

`rga "fatal" /var/log/syslog*` поиск строк по слову `fatal` во всех файлах syslog (включая архивные)

sig

`brew install ynqa/tap/sigrs` установить sig интерактивный `grep` на Rust

```
curl -s https://raw.githubusercontent.com/Lifailon/hwstat/rsa/README.md > README.md
cat README.md |& sig -a
```

sed

`cat /etc/passwd | sed -n "1,5p"` отобразить с первой по пятую строку (`p`)

`cat /etc/passwd | sed "$ d"` удалить (`d`) последнюю строку

`cat /etc/passwd | sed "1,3d"` удалить с первой по третью строку (`2,3d`)

`echo "One 1" | sed "s/One/Two/; s/1/2/"` заменить One на Two и 1 на 2

`cat /etc/zabbix/zabbix_agentd.conf | sed "s/127.0.0.1/192.168.3.102/" # > /etc/zabbix/zabbix_agentd.conf` заменить (`s`) ip-адрес

`cat /etc/zabbix/zabbix_agentd.conf | sed "/^#\|^$/d"` удалить пустые строки `^$` и комментарии `^#`

`timedatectl | grep zone | sed -E "s/.+zone: //"` удалить любое кол-во лимволов до слова "zone: " включительно, используя Regex (`-E/-r`)

`echo -e "test\ntest" | sed "2s/test/test2/"` заменить во второй строке (`2s`)

`echo -e "test\ntest\ntest\ntest" | sed "2,3s/test/test2/"` заменить во второй и третей строке (`2,3s`)

`echo -e "test\ntest\ntest\ntest" | sed "2ctest2"` заменить вторую строку (`2c`)

`echo "The test and test" | sed "s/test/test2/g"` заменить для каждого совпадения (`/global`)

`echo "The test and test" | sed "s/test/test2/2"` заменить для второго совпадения (`/2`)

`echo "line2" | sed "i\line1"` добавить строку в начало (`i`)

`echo "line1" | sed "a\line2"` добавить строку в конец (`a`) или в после указанной строки (`2a`)

`echo "11 22 33 34" | sed "y/123/234/"` заменить 1 на 2, 2 на 3, 3 на 4 (`y`)

`ls -R | grep ':' | sed "s/:$//; s/[^\/]*\// - /g"` удалить `:` в конце и заменить вначале строки `"/любое кол-во символов между/"` на `" - "` для всех (`/g` global)

`echo "test<version>3.6.4</version>test" | sed -r 's/[^<]*<(.*)>.*/\1/;s/<.*//;s/.*>//'` использовать regex (`-r`)

`ps aux | grep -E "^zabbix .+ -c" | sed -E "s/^zabbix.+-c //"` найти процесс zabbix с ключем `-c` и оставить путь `conf`

`echo "MPEG-H HEVC, 88.5 Мбит/с, 3840x2160, 23.976 кадр/с, 10 бит" | sed -nr 's/.* ([0-9]+x[0-9]+).*/\1/p'` выводить только найденные строки (`-n`) с заменой (`s/`), ищем только цифры `[0-9]` где одно или более вхождений (`+`) и между ними `x` , вывести только первую группу поиска (то, что в скобках) на печать (`/p`)

awk

`cat /etc/passwd | awk -F: '{print "name: " $1 " \t Dir: " $NF}'` вывести содержимое первого и последнего `$NF` элемента в строке, используя разделитель ":" и табуляцию (\t)

`echo 'one two three four' | awk '{print $(NF-1)}'` вывести содержимое преподследнего элемента

`echo 'one two three four five' | awk '{print $((NF/2)+1)}'` вывести содержимое из середины

`echo "One Two Three" | awk '{$3="Four"; print $0}'` заменить третье значение/переменную в строке

`cat /etc/passwd | awk 'BEGIN{FS=":"; OFS=" - "} {print $1,$7}'` указать разделитель послей (элементов) на вход (`FS`) и заменить его на выходе (`OFS`)

`uptime | awk 'BEGIN{RS=" "; ORS="\n"} {print $0}'` указать разделитель записей (строк) на входе (`RS`) и заменить его на выходе (`ORS`)

`echo -e "12345\n54321" | awk 'BEGIN{FIELDWIDTHS="2 3"}{print $1,$2}'` указать фиксированное кол-во символов для разделения

`lsof | awk '{if($7=="REG")print $0}'` условие для выборки по столбцу

`cat /etc/ssh/sshd_config | awk '/Port / {print $2}'` условие поиска для вывода

`cat /etc/ssh/sshd_config | awk 'length $0 > 1'` вывести строки, которые длиннее, чем 1 символ (удалить пустые строки)

`cat /var/log/syslog | grep "$date" | awk '{print length($6)}'` вывести длинну значения

`awk 'BEGIN{x = "low"; print toupper(x)}'` использовать функцию для перевода в вверхний регистр

`awk 'BEGIN{x = "LOW"; print tolower(x)}'` использовать функцию для перевода в нижний регистр

`echo "1 2 3 4:5:6" | awk '{item=$4; split(item,array,":"); print array[2]}'` разбить 4 значение на массив (используя функцию `split`) и забрать значение по 2-му индексу

`free | awk '{if (NR == 2) print $0}'` вывести только вторую строку

`free | awk '{if (NR >= 2) print $0}'` вывести втроую и последующие строки

`free | awk '{if (NF >= 5) print $0}'` вывести строки, где 5 или больше значений

`cat /etc/passwd | awk '{ if (NR >= 10 && NR <= 20) print $0}'` вывести с 10 по 20 строки

`last | sed -n 1p | awk '$2=" ",$4=" "{print $0}'` вывести все, кроме 2 и 4 значения (заменить)

`ps -A | awk '{sum=""; for(i=1;i<=NF;i++) { if (i != 2) {sum=sum" "$i} } print sum}'` вывести все, кроме 2-го значения

`cut -d',' -f2,4 file.csv` взять второй и четвертый столбцы

```
awk -F',' '{if (NF >= 4) print $2, $4}' file.csv
```

```
grep "Error" logs.txt
awk '/Error/' logs.txt
awk '/^Error [0-9]{3}:/' logs.txt
```

`grep -c "Success" logs.txt` посчитать количество совпадений

```
awk '/Success/ {count++} END {print count}' logs.txt
```

`sed 's/Error/Success/g' file.txt` замена слов

```
awk '{gsub(/Error/, "Success"); print}' file.txt
```

```
sort file.txt | uniq
awk '!seen[$0]++' file.txt
```

заполняем уникальный массив строк

`wc -w file.txt` посчитать количество слов в файле

```
awk '{count += NF} END {print count}' file.txt
```

`sed -n '10,20p' file.txt` вывести с 10 по 20 строку

`awk 'NR>=10 && NR<=20' file.txt`

`awk '{sum += $1} END {print sum/NR}' numbers.txt` получить среднее значение чисел в первом столбце

`awk '{ if ($2 > 50000) print $1, "> 50K"; else print $1, "< 50K" }' data.txt` вывести значение первого столбца, если значение второго столбца выше или ниже 50 тысяч

`awk '{ sum = 0; for (i = 1; i <= NF; i++) sum += $i; print "сумма:", sum }' data.txt` посчитать сумму числе в каждой строке

`awk '{ for (i = 1; i <= NF; i++) sum[i] += $i } END { for (i in sum) print "Столбец", i, "сумма:", sum[i] }' data.txt` посчитать сумму числе в каждом столбце

printf

`top=$(top -bn1)`

`printf "%s\n" "${top[@]}"` вывести вывод массива построчно

`printf "%.2f \n" 1.1111` округлить до 2 символов после запятой

`printf "%.0f \n" 1.6` удалить дробную часть (округлить до 2)

`printf "Arg1: %s\nArg2: %s\n" "10" "20"` принимает и выводит аргументы (%s) в виде строки

cut

`echo "1 2 3" | cut -c 1,5` вывести первый и пятый симов (`--bytes` / `--characters`)

`echo "1 2 3" | cut -c 1-3` вывести с первой по третий символ

`echo "1 2 3" | cut -c3-` удалить первые 2 символа

`echo -e "test1,test2,test3\ntest1,test2,test3" | cut -d , -f 2-100` указать разделитель полей/столбцов (`--delimiter`) и какие столбцы вывести (`--fields`) с 2 по 100

`echo -e "test1,test2,test3\ntest1,test2,test3" | cut -d , -f 1,3 | sed "s/,/ /"` вывести 1 и 3

`echo -e "test1,test2,test3\ntest1 test2 test3" | cut -d , -f 1,3 -s` печатать строки, где есть разделитель (`-s`)

rev

`echo "D:\plex-content\Rick.and.Morty.S07.2023.WEBDLRip.MegaPeer" | rev | cut -d \\ -f 1 | rev` забрать последний элемент в пути (вначале разворачивает всю строку, забирает первый элемент и разворачивает строку обратно)

`echo "D:\plex-content\Rick.and.Morty.S07.2023.WEBDLRip.MegaPeer" | sed -r 's/.+\\//'` удалить все до последнего слеша

`echo "D:\plex-content\Rick.and.Morty.S07.2023.WEBDLRip.MegaPeer" | sed 's/.*\\\(.*\)/\1/'` удаляет все до последнего слеша и забирает одну группу захвата, что остается после удаления, и заменяет вывод на первую группу (1)

`echo "D:\plex-content\Rick.and.Morty.S07.2023.WEBDLRip.MegaPeer" | awk -F '\\' '{print $NF}'` забрать последний элемент массива (NF)

tr

`echo "10 20 100 200" | tr 1 2` translate заменяет 1 на 2 для всех подходящих сомволов (20 20 200 200)

`echo "1 2 3" | tr " " ","` заменить пробелы на запятые (1,2,3)

`echo "1 2 3" | tr -d " "` удалить пробелы (123)

man

cheat sh

`curl cheat.sh/curl`

`curl cheat.sh/grep`

`curl cheat.sh/sed`

`curl cheat.sh/awk`

`curl cheat.sh/jq`

`curl cheat.sh/iptables`

`curl cheat.sh/find`

tldr

`pip3 install tldr` упрощенный вариант man с примерами использования
`tldr curl` веб-версия: https://manned.org/man/curl

debug

`trap 'echo "$BASH_COMMAND"' DEBUG` построчная отладка скриптов bash, команда trap перехватывает сигнал DEBUG, посылаемый перед выполнением команды и выводит команду на экран
`trap 'echo "$BASH_COMMAND";read' DEBUG` read ожидает ввода с клавиатуры (Enter или Ctrl+C) перед выполнением каждой командой
`bash -x script.sh` отладка (печать команд и их аргументов по мере их выполнения)
`bash -x -c "ls -l" | grep *.sh | awk '{print $5,$NF}'` запуск команды через интерпритатор bash и вывод отладки
`bash --debug script.sh` проверка на ошибки
`apt-get install shellcheck` установить shellcheck
`shellcheck -S error hwstat.sh` error/warning/info/style
`pip3 install thefuck` установить thefuck
`bas hwstat.sh` запустить команду с ошибкой
`fuck` автоматически исправляет последнюю ошибочную команду из выпадающего списка (up/down)

tools

```
pip install toolong
```
`tl /var/log/auth.log` интерактивный просмотр логов в консоли с фильтрацией
`tl access.log* --merge` просмотр нескольких файлов

`apt install bat` аналог cat (https://github.com/sharkdp/bat) с подсветкой синтаксиса
`bat /etc/netplan/*.yaml`

`tree /var/log/` древовидный просмотр директорий и дочерних файлов

`cargo install --locked broot` установить broot (https://github.com/Canop/broot), аналог tree
`broot kinozal-bot/`

```
echo 'deb http://cz.archive.ubuntu.com/ubuntu jammy main universe' >> /etc/apt/sources.list && apt update
```
`apt install exa` установить аналог ls (https://github.com/ogham/exa)
`exa $(pwd) -l --icons` отобразить иконки с подсветкой прав доступа

`cargo install eza` аналог ls (https://github.com/eza-community/eza) на базе exa
`eza -l --icons`
`eza --tree kinozal-bot/`

`cargo install lsd` аналог ls (https://github.com/lsd-rs/lsd)
`lsd -l kinozal-bot/`

```
column /etc/passwd -t -s ":"
```
`netcheck -t ping yandex.ru us1.node.check-host.net | sed -r 's/"//g; s/,$//; s/\{|\}|\[|\]//' | column -t -s ":"` распарсить JSON и добавить отступ (табуляцию) для колонок

`ls /home | wc -l` word count выводит количество строк (`--line`)
`ls /home | wc -w` количество слов (`--words`)
`ls /home | wc -m` количество символом (`--chars`)
`ls /home | wc -c` количество символов/байт (`--bytes`)

`echo "(5.5-2.2)" | bc` математические вычисления
`echo "(5.5-2.2)" | bc | sed -E "s/\..+//"` удалить дробную часть
`echo "1 < 2" | bc` возвращает булевое значение (1 - да или 0 - нет)
`echo "1 > 2" | bc` false (0)
`icmp_ignore=$(cat /proc/sys/net/ipv4/icmp_echo_ignore_all)` забрать значение
`if (( $(echo "$icmp_ignore == 1" | bc) )); then echo "true"; else echo "false"; fi` проверить в условии арефметическое значение на равенство (возвращает 0 - `false` или 1 - `true`)

```
a=1
b=0.55
echo $(bc <<< "scale=2; $a+$b")
echo "print $a+$b" | perl
echo "print($a+$b)" | python3
echo "print($a+$b)" | lua
echo "puts $a+$b" | ruby
pwsh -Command $a+$b

echo -e "key1\nkey2\nkey3" > 1.txt
echo -e "value1\nvalue2\nvalue3" > 2.txt
paste 1.txt 2.txt -d :
cat /etc/passwd | paste -s -d +
```
объединяет два файла в один многоколоночный вывод

объеденить (join) многострочный файл, используя указанный delimiter

```
echo -e "test1\ntest2" > 1.txt \
```
echo -e "test\ntest2\ntest3" > 2.txt

```
diff 1.txt 2.txt -c
```
! есть изменения, + есть новая строка

```
diff 1.txt 2.txt -yi
```
сравнивает в две колонки (| есть изменения, + есть новая строка) и игнорировать регистр (-i)

```
diff 1.txt 2.txt -u
```
объеденяет два файла в один вывод с отображением изменений (+/)

```
diff 1.txt 2.txt -ibBEt
```
не учитывать пробелы (-b) и пустые строки (-B), игнорировать изменения в табуляциях (-E) и заменить табуляции на пробелы в выводе (-t)

```
diff -c <(echo "$predu") <(echo "$du")
```
сравнить содержимое переменных

```
snap install diff-so-fancy
diff -u file-1.txt file-2.txt | diff-so-fancy

apt install jdupes
jdupes .
```
поиск дубликатов

```
cat /etc/passwd | sort -r
```
отсортировать вывод по алфовиту в обратном порядке (-r)

```
du -h ~ | sort -n
```
сортировать по арифметическому значению (-n) размер файлов и директорий

```
ls -l | sed 1d | sort -nk5
```
сортировка по пятой колонке (-k)

```
cat $tmp | sort -t "." -nk4
```
сортировать по четвертой колонке, используя разделитель (-t) точку

```
echo -e "1 2\n1 2\n2 1\n1 2" | uniq
```
удаляет соседние одинаковые строки

```
echo -e "1 2\n1 2\n2 1\n1 2" | sort | uniq
```
удалить все дубликаты

```
echo -e "1 2\n1 2\n2 1\n1 2" | sort | uniq -c
```
добавляет в начало каждой строки кол-во повторений

```
echo -e "1 2\n1 2\n2 1\n1 2" | sort | uniq -u
```
отобразить только уникальные строки, без строк с повторениями

```
ls -l | fold -w 50
```
задать ширину вывода каждой строки, выпадающее за указанный предел переносится на новую строку

```
ls -l | fold -w 50 -s
```
разбивать строки только на символах пробела (--space)

```
cat /var/log/syslog | head -n 5
```
выводит первые 5 строк файла

```
cat /var/log/syslog | tail -n 5
```
просмотр последних 5 строк файла

```
tail -f /var/log/syslog
```
просмотр содержимого файла в реальном времени

```
apt install multitail
multitail -f /var/log/auth.log -f /var/log/kern.log
multitail -l "journalctl -fu ssh" -l "journalctl -fu cron"

less /var/log/dmesg
```
вывести лог ядра с возможностью пролистывания

```
watch df -h
```
выводит на экран и обновляет состояния подключенных устройств каждые 2 секунды

```
echo "line1" | tee test.txt
```
перезаписать файл (>)

```
ls > /dev/null
```
перенаправить вывод в null

```
echo "line2" | tee -a test.txt
```
добавить (>>) текст новой стройокй в конец файла

```
echo -e "line3\nline4" >> test.txt
```
добавить две новые строки

```
split -l 100 input_file.txt output_prefix
```
разделить файл на части по 100 строк в каждой

```
split -b 10M input_file.txt output_prefix
```
разделить файл на части по указанному размеру (например, 10MB)

`yes` предназначена для автоматического вывода строки или символа, повторяющегося бесконечно (для нагрузки системы), либо для автоматического подтверждения запросов в других командах

dust

`snap install dust` установить dust - альтернатива `du` на Rust
`dust /home/lifailon` выводит график используемого пространства по директориям и файлам для анализа занятого пространства
`dust -s` показывает размер файла, а не объем используемого им дискового пространства
`dust -n 30` выводит 30 каталогов (по умолчанию — высота терминала)
`dust -d 3` показывает 3 уровня подкаталогов
`dust -D` отобразить только директории
`dust -F` отобразить только файлы
`dust -f` считайте файлы вместо дискового пространства
`dust -i` не показывать скрытые файлы
`dust -z 10M` минимальный размер, включать только файлы размером более 10 МБ
`dust -z 40000/30MB/20kib` исключить выходные файлы/каталоги размером менее 40 000 байт/30 МБ/20 КБ
`dust -o si/b/kb/kib/mb/mib/gb/gib` формат вывода
`dust -e "\.png$"` включать только те файлы, которые соответствуют регулярному выражению (например, только файлы `png`)
`dust -v "\.png$"` регулярное выражение для игнорирования файлов с разрешением `png`
`dust -j | jq` вывод в формате `JSON`
`dust -P` отключить индикатор прогресса

fd

`apt install fd-find` установить fd быстрая альтернатива `find` на Rust
`fdfind` без аргументов заменяет `ls -R` для рекурсивного поиска в текущем каталоге
`fdfind log /var` ищет в указанной директории по частичному совпадению
`fdfind -tf "\.yaml$" / | fzf` ищет все файлы (`--type file` или директории `--type directory`) с расширением `.yaml` с корня с выводов в `fzf`
`fdfind --type file -H pre-commit` поиск скрытых файлов
`fdfind --type f -e pdf . $HOME | rofi -keep-right -dmenu -i -p FILES -multi-select | xargs -I {} xdg-open {}` интеграция с rofi (графическое меню)
`fd -e zip -x unzip` рекурсивно найти все zip-архивы и распаковать их

fd-fzf

```
# fdfind over fzf
if command -v fzf > /dev/null; then
    function fd-fzf(){
        if [ -z "$1" ]; then
            # Current path by default
            fdfind . ${pwd} | fzf
        else
            # Specified path
            fdfind . $1 | fzf
        fi
    }
    # Alt+F for fd-fzf
    bind '"\ef": "fd-fzf\n"'
    # Alt+Shift+F for rga-fzf
    if command -v rga-fzf > /dev/null; then
        bind '"\eF": "rga-fzf\n"'
    fi
fi
```

findutils

find

`find / -name "*.sql"` найти файлы, начать поиск с корня (/)

`find / -iname "mysql"` найти файлы не учитывая регистр (-i)

`find ~ -name "test.*" -not -name "*.conf"` найти все файлы с наименование test, которые имеют любое расширение, за исключением (-not) расширения .conf

`find ~ -amin -10` поиск файлов по дате последнего чтения (-amin) которые просматривались (cat/nano) за последние 10 минут

`find ~ -type f -mmin -10` найти файлы (-type f), которые были модифицированны за последние 10 минут (-nmin)

`find ~ -type f -mtime +1 -mtime -7` найти все файлы, модифицированные между 1 и 7 днями назад

`find ~ -type d -mtime +1 -mtime -7` поиск директорий

`find ~ -size +50M -size -100M` поиск файлов в Linux по их размеру, от 50 до 100 мегабайт

`find / -perm 444` поиск файлов по режиму доступа (только чтение для всех)

`find /home/lifailon/ -user root` поиск файлов по владельцу

`find /home/lifailon/ -group root` поиск по группе

`find /root/ -empty` поиск пустых файлов или директорий

exec

`touch -t 202306222200.15 /tmp/test.txt` создать файл с указанной датой создания

`find /tmp -type f -mtime +30 -exec rm -f {} \;` удалить все файлы, котоыре не изменялись больше 30 дней

`find /tmp -type f -name "*.txt" -exec rm -f {} \;` удалить все текстовые файлы в директории tmp

`dd if=/dev/zero of=/var/log/test.log count=11 bs=1M` создать файл заполненный нулями указанного размера

`find /var/log -type f -name "*.log" -size +10M -exec rm -f {} \;` удалить все лог-файлы, объёмом больше 10 Мбайт

locate

`apt install plocate` альтернатива стандартного `mlocate` с более быстрым и меньшим по размеру индексом

`updatedb` обновить индексы базы данных

`ls -lh /var/lib/[mp]locate/*.db` проверить размер базы данных

`locate .torrent` найти по частичному совпадению в имени или расширению

`locate .torrent -c` отображает количество найденных результатов

`locate -n 10 .torrent` вывести 10 результатов

`locate -i Kinozal-Bot` игнорировать регистр

`locate -r "\.log$"` использовать регулярные выражения

```
sudo curl -s https://github.com/pr4k/locate/releases/download/v0.1.1/locate-linux -o /usr/bin/locate -o /usr/bin/locate
sudo chmod +x /usr/bin/locate
```

```
locate-linux -p /home/lifailon/ -q /qbittorrent
locate-linux -p /home/lifailon/.bash_history -q /qbittorrent
```

xargs

`echo {1..10} | xargs -n1 -P4 bash -c 'echo Start task $1 && sleep $1 && echo Complate task $1' _` принимает 1 аргумент и запускат до 4-х потоков за раз

`du -a /var/log | awk '{print $2}' | xargs fincore` передать вывод первой команды (построчно) в аргументы следующей

fincore

```
sudo apt install util-linux-extra
```

`fincore /var/log/*` отобразить все файлы, которые находятся в кэше страниц оперативной памяти (page cache)

`fincore /var/log/syslog` 4.3M (данные файла, хранящиеся в памяти) 1100 (кол-во страниц хранящиеся в памяти PageCache) 199.7M (размер файла)

`fincore /var/log/syslog -J` вывод в JSON (--raw вывод без табулияции, --noheadings без заголовков, --byte размер файла в байтах)

```
apt install vmtouch
```

`vmtouch /var/log/syslog` узнать какой процент указанного файла находится в страничном кеше (Resident Pages: 1100/51119 4M/199M 2.15%)

lspage

```
fc=$(du -a $1 2> /dev/null | awk '{print $2}' | xargs fincore 2> /dev/null)
echo -e "PAGE\tSIZE\tPATH"
printf "%s\n" "${fc[@]}" | grep -wvE "0B|SIZE" | awk 'BEGIN {OFS="\t"}; {print $1,$3,$4}'
```

bashrc

```
nano ~/.bashrc
```

```
# Псевдонимы для команды или набора команд с флагами для сокращения ввода
alias tspin=tailspin
alias ts=tailspin

# Забиндить очистку ввода на Ctrl+L
bind '"\C-l": "^\C-u\C-mclear\C-m"'

# Определить переменную окружения, доступную для дочерних процессов, запущенных в текущей сессии
# Игнорировать запись в историю команд, которые начинаются с пробела
export HISTCONTROL=ignorespace

# Добавить фильтрацию по введеному тексту в истории команд при испоьзовании стрелочек вверх и вниз
if [[ "$-" == *i* ]]; then
    bind '"\e[A": history-search-backward'
    bind '"\e[B": history-search-forward'
fi
```

`source ~/.bashrc` применить политики (перечитать профиль)

oh-my-bash

Установить `oh-my-bash` (обновляет профиль, делая рядом резервную копию старого файла в `.bashrc.omb-TIMESTAMP`):

```
bash -c "$(curl -fsSL https://raw.githubusercontent.com/ohmybash/oh-my-bash/master/tools/install.sh)"
```

Настроить динамический профиль:

```
function sysStat() {
    top=$(top -bn1)
    cpu=$(echo "$top" | grep "%Cpu(s)" | awk '{printf "%.0f%%", 100-$8}')
    # sys=$(echo "$top" | grep "%Cpu(s)" | awk '{printf "%.0f%%", $4}')
    # usr=$(echo "$top" | grep "%Cpu(s)" | awk '{printf "%.0f%%", $2}')
    avg=$(echo "$top" | grep "load average" | awk -F ': ' '{print $2}' | awk -F ', ' '{print $1"/"$2"/"$3}')
    mem=$(echo "$top" | grep "MiB Mem" | awk '{printf "%.1fG/%.1fG", ($8)/1024, $4/1024}')
    disk=$(df -h | awk '$NF=="/"{print $3"/"$2}')
    echo "📊 $cpu ($avg) $mem 💾 $disk"
}

function gitStatus() {
    status=$(git status --porcelain 2>/dev/null)
    if [ -z "$status" ]; then
        echo ""
        return
    fi
    branch=$(git rev-parse --abbrev-ref HEAD)
    result="🔶 ($branch) "
    added=$(echo "$status" | grep -c '^?')
    modified=$(echo "$status" | grep -c '^ M')
    deleted=$(echo "$status" | grep -c '^ D')
    [ "$added" -gt 0 ] && result+="\e[32m+${added} \e[0m"
    [ "$modified" -gt 0 ] && result+="\e[33m~${modified} \e[0m"
    [ "$deleted" -gt 0 ] && result+="\e[31m-${deleted} \e[0m"
    echo "$result"
}

export PROMPT_COMMAND+='
    SYSSTAT=$(sysStat);
    GITSTATUS=$(gitStatus);
'

PS1='\[\e[34m\]$SYSSTAT \[\e[0m\]'
PS1+='\[\e[32m\]👤 \u \[\e[0m\]'
PS1+='\[\e[33m\]📁 \w \[\e[0m\]'
PS1+='$(echo -e "$GITSTATUS")'
PS1+='\[\e[34m\]> \[\e[0m\]'
```

fzf

`apt install fzf` установить fzf

`history | fzf` интерактивный поиск с фильтрацией

`eval $(history | fzf | awk '{print $2}')` выполнить (eval) выбранную команду из списка (добавить в макрос)

`find / -name "*.yaml" | fzf | xargs cat` найти в системе все файлы `yaml`, запустить по ним поиск и передача в `cat` для чтения выбранного файла

Поиск по истории команд с помощью функции `hstr` или псевдонима `h` и комбинации `Ctrl+R` через `fzf`:

```
if command -v fzf > /dev/null; then
    function hstr() {
        local current_input="$READLINE_LINE"
        command=$(tac $HOME/.bash_history | sed '/^#/d' | awk '!seen[$0]++' |  fzf --height 20 --reverse --query="$current_input" | sed -r "s/^\s
        if [[ -n "$command" ]]; then
            READLINE_LINE="$command"
            READLINE_POINT=${#READLINE_LINE}
        fi
    }
    alias h=hstr
    bind -x '"\C-r": h'
fi
```

Kill jobs over fzf:

```
if command -v fzf > /dev/null; then
    function jobKill() {
        pid=$(jobs -l | fzf --height 20 --reverse --preview "echo {}" --preview-window down | awk '{print $2}')
        if [[ -n "$pid" ]]; then
            READLINE_LINE="kill -9 $pid"
            READLINE_POINT=${#READLINE_LINE}
        fi
    }
    bind -x '"\C-j": jobKill'
fi
```

fzf-obc

Установить fzf over bash complete (выпадающий список автодополнения команд) и добавить в профиль `bash`:

```
git clone https://github.com/rockandska/fzf-obc $HOME/.local/opt/fzf-obc
echo "source $HOME/.local/opt/fzf-obc/bin/fzf-obc.bash" >> $HOME/.bashrc
```

hstr

`sudo apt install hstr` установить hstr (https://github.com/dvorka/hstr)

`hstr -f` избранное (Ctrl+F добавить в избранное)

`hstr -n bash log` вывести на экран отфильтрованную историю

```
if command -v hstr > /dev/null; then
    bind -x '"\C-r": hstr'
fi
```

mcfly

Установить homebrew и mcfly, который заменяет поиск истории через `Ctrl-R` на интеллектуальную поисковую систему, которая учитывает рабочий каталог и контекст недавно выполненных команд:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
echo 'eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"' >> ~/.bashrc
eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"
source ~/.bashrc
brew install mcfly
echo 'eval "$(mcfly init bash)"' >> ~/.bashrc
source ~/.bashrc
```

compgen

`compgen -c` выводит все команды, доступные в текущей оболочке

`compgen -a` выводит все алиасы, определенные в текущей оболочке

`compgen -b` выводит все встроенные команды Bash

`compgen -k` выводит все зарезервированные слова Bash

`compgen -v` выводит все переменные, определенные в текущей оболочке

`compgen -A export` выводит все экспортированные переменные

`compgen -A function` выводит все функции, определенные в текущей оболочке

`compgen -A arrayvar` выводит все массивы, определенные в текущей оболочке (echo ${BASH_ALIASES[@]})

`compgen -A hostname` выводит все известные хосты

`compgen -A job` выводит все активные задания (ping ya.ru > /dev/null &)

`compgen -A service` выводит все службы (для систем, поддерживающих службы, например, через systemd)

`compgen -d` выводит все директории в текущем каталоге

`compgen -f` выводит все файлы и директории в текущем каталоге

`compgen -u` выводит всех пользователей системы

`compgen -g` выводит все группы системы

`compgen -W "start stop status restart" st` выводит список слов из wordlist, которые начинаются с prefix "st"

cron

`ls /etc/cron.d/` директория хранения задач различных пакетов (atop, sysstat)

`ls -l /etc/cron.hourly && ls -l /etc/cron.daily && ls -l /etc/cron.weekly && ls -l /etc/cron.monthly` директории для скриптов, которые надо выполнять раз в час, день, неделю и месяц

`crontab -l` просмотр задач

`crontab -l | grep -Pv "^$|^#"` отобразить только активные задания

`crontab -u lifailon -l` отобразить задачи пользователя root

`crontab -e` создать задачу от текущего пользователя

`sudo crontab -u root -e` создать задачу от пользователя root

`crontab -r` очистить все задачи

```
cat /etc/crontab
```

```
#### # .---------------- минута (0 - 59)
#### # |  .------------- час (0 - 23)
#### # |  |  .---------- мень месяца (1 - 31)
#### # |  |  |  .------- месяц (1 - 12) или jan,feb,mar,apr...
#### # |  |  |  |  .---- день недели (0 - 6) (Воскресень 0 или 7) или sun,mon,tue,wed,thu,fri,sat
#### # |  |  |  |  |
#### # *  *  *  *  * user-name command to be executed
####  17  *  *  *  *   root    cd / && run-parts --report /etc/cron.hourly
####  25  6  *  *  *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
####  47  6  *  *  7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
####  52  6  1  *  *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
```

`0,14,29,44 * * * *` каждые 15 минут

`*/15 * * * *` каждые 15 минут

`00 23 * * * systemctl restart zabbix-agent && echo $(date): Reboot Zabbix Agent use cron >> /var/log/reboot.log` выполнять перезапуск службы каждый день в 23:00 и писать в лог

`00 03 * * 6 echo $(date): Reboot Operating System use cron >> /var/log/reboot.log && /sbin/reboot` выполнять перезагрузку системы один раз в субботу в 3 часа ночи

`@reboot date >> ~/date-reboot.log` выполнять один раз после перезагрузки

```
journalctl -eu cron
cat /var/log/syslog | grep -i cron
```

```
#!/bin/bash
addr="google.com"
path="/var/log/icmp-test.log"
date=$(date | awk '{print $3,$2,$4}')
loss=$(ping -c 2 $addr | grep -Ewo "[0-9]+%")
if [ $loss = "100%" ]; then
    echo "$date: $addr - unavailable" >> $path
else
    echo "$date: $addr - available" >> $path
fi
```

`echo "*/1 * * * * bash /root/google-icmp-test.sh" >> /var/spool/cron/crontabs/root` добавить задачу в планировщик на выполнение скрипта каждую минуту

`cp /etc/hosts /etc/hosts.bak` backup файла

`echo "11.11.11.11 google.com" >> /etc/hosts` изменить адрес для недоступности хоста

`cp /etc/hosts.bak /etc/hosts` восстановить файл

`cat /var/log/icmp-test.log | grep unavailable` отфильтровать лог по unavailable

systemctl

`systemctl reload ssh` обновить конфигурацию сервиса из файла юнита (если у юнита есть эта функция)
`systemctl status ssh` отображает состояние системы, юнитов (в том числе Failed) и запущенные процессы пользователей
`systemctl status sshd | grep -P "Active.+;" | sed -r "s/.+; | ago//g"` время работы службы
`systemctl start ssh` запустить юнит (до перезагрузки)
`systemctl stop ssh` остановить юнит (до перезагрузки)
`systemctl restart ssh` перезапустить сервис
`systemctl enable ssh` добавить в автозагрузку
`systemctl disable ssh` удалить из автозагрузки
`systemctl mask ssh` выключить юнит, который нельзя будет запустить вручную или как зависимость (создает симлинк на /dev/null)
`systemctl unmask ssh` включить юнит (удалить симлинк)
`systemctl daemon-reload` перезапустить юнит systemd
`systemctl cat ssh` отобразить путь и содержимое unit-файла
`systemctl edit --full ssh` открыть для редактирования файл юнита
`systemctl list-dependencies ssh` дерево зависимостей
`systemctl list-dependencies ssh --reverse` зависящие сервисы от указанного юнита
`systemctl list-units --type service --all` отображение статуса всех сервисов
`systemctl list-unit-files | sed "1d;$ d" | sed "$ d" | wc` отобразить кол-во всех файлов конфигурации сервисов на диске
`systemctl list-unit-files | grep zabbix` отфильтровать по имени
`systemctl list-unit-files --type=service` список всех сервисов
`systemctl list-unit-files --type=service --state=enabled` список сервисов, добавленных в автозагрузку
`systemctl list-units --all --type=service --plain --no-legend --no-pager --output=json`
`--all` выводить все типы юнитов, включая активные, неактивные и остановленные
`--type=service` выводить только системные службы управляемые systemd (не ключает в вывод другие типы юнитов, такие как socket или device)
`--plain` вывод в текстовом формате без форматирования
`--no-legend` отключает вывод заголовков для столбцов
`--no-pager` отключает использование постраничного вывода (less)
`ls /usr/lib/systemd/system` юниты поставляемые вместе с системой и устанавливаемыми приложениями
`ls /run/systemd/system` юниты созданные динамически в runtime
`ls /etc/systemd/system` юниты системного администратора

systemctl-tui

`cargo install systemctl-tui --locked` быстрый и простой TUI-интерфейс для взаимодействия с службами и журналами systemd на Rust (https://github.com/rgwood/systemctl-tui), от создателя NuShell
`systemctl-tui`

unit

```
#!/bin/bash
while true; do
    addr="google.com"
    path="/var/log/icmp-test.log"
    date=$(date | awk '{print $3,$2,$4}')
    loss=$(ping -c 2 $addr | grep -Ewo "[0-9]+%")
    if [ $loss = "100%" ]; then
        echo "$date: $addr - unavailable" >> $path
    else
        echo "$date: $addr - available" >> $path
        tail -n 1 $path
    fi
    sleep 5
done
```

```
nano /etc/systemd/system/icmp-test-log.service
```

```
[Unit]
Description=icmp test output to log
After=network.target

[Service]
ExecStart=/bin/bash "/root/google-icmp-test.sh"
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
Type=simple

[Install]
WantedBy=multi-user.target
```

```
systemctl daemon-reload
systemctl enable icmp-test-log.service
systemctl start icmp-test-log
systemctl status icmp-test-log
tail -f /var/log/icmp-test.log
```

journalctl

`journalctl --system` отобразить системный журнал
`journalctl --user` отобразить пользовательский журнал текущего пользователя
`journalctl -m` отобразить записи из всех доступных журналов (--merge)
`journalctl -ek` отобразить только сообщения ядра (kernel, --dmesg) из текущей загрузки
`journalctl -t systemd` показать записи с указанным идентификатором системного журнала
`journalctl _PID=3972315` отобразить сообщения по PID процесса
`journalctl -eu ssh` отобразить сообщения с конца (--pager-end) для выбранного сервиса (--unit)

`g` перейти в начало листинга
`G` перейти в конец

`journalctl -fu ssh` выводить новые сообщения в реальном времени (-f/--follow)
`journalctl -fu ssh` выводить новые сообщения в реальном времени (-f/--follow)
`journalctl -ru ssh` вывести сообщения с конца (сверху новые записи, --reverse)
`journalctl -n 100 -u ssh --no-pager` вывести 100 строк (--lines) из журнала и не передавать вывод на автоматический скроллинг
`journalctl -p 3` вывести записи с указанным приоритетом, например, только ошибки и выше по важности: неработоспособность(0), alerts(1), critical(2), errors(3), warning(4), notice(5), info(6), debug(7)
`journalctl -S "2023-09-01 12:00:00" -U "2023-09-01 15:00:00"` отобразить сообщения от (--since) 1 сентября с 12:00:00 по (--until) 15:00:00
`journalctl --since today` отобразить сообщения за сегодня
`journalctl -b` отобразить сообщения с момента последней загрузки системы (boot)
`journalctl --list-boots` показать список сохраненных загрузок системы
`journalctl -b ba6b2292a0e84d83a81cedfaa221926f` показать сообщения с момента конкретной загрузки системы (--boot)
`journalctl --quiet` не показывать информационные сообщения и предупреждения о привилегиях
`journalctl --no-hostname` подавить вывод поля имени хоста
`journalctl -n 1 --no-pager --output=json-pretty` вывод в формате JSON (json-sse, json-seq)
`journalctl -n 1 --no-pager --output=json-pretty --output-fields=PRIORITY,MESSAGE` отфильтровать вывод

`journalctl --fields` вывести список всех используемых полей (UNIT, USER_UNIT, _SYSTEMD_UNIT, _SYSTEMD_USER_UNIT и т.д.)

`journalctl --field=UNIT > system_units.log` вывести список всех юнитов в системе
`journalctl --field=USER_UNIT > user_units.log` вывести список всех пользовательских юнитов в системе
`comm -12 <(sort system_units.log) <(sort user_units.log)` построчное сравнение двух отсортированных файлов со списком журналов без вывода общих строк в 1 и 2 файлах (-12)

`journalctl --disk-usage` вывести общее использование диска всеми файлами журнала (Archived and active journals take up 2.3G in the file system)
`journalctl --flush` очистить все данные журнала из директорий /run в /var
`journalctl --vacuum-time=1month` удалить файлы журнала, старше указанного времени (1-го месяца)
`journalctl --vacuum-size=100M` очистить логи, чтобы размер хранилища соответствовал указанному размеру

`journalctl --vacuum-files=100` оставить только указанное количество файлов журнала

`journalctl --rotate` запустить немедленную ротацию файлов журнала

`journalctl --sync` синхронизировать незаписанные сообщения журнала на диск

`journalctl --relinquish-var` прекратить запись на диск, войти во временную файловую систему

`journalctl --verify` проверить целостность файла журнала

`journalctl --header` вывести список журналов

File path - путь к файлу журнала на диске

Incompatible flags - несовместимые флаги с этим журналом

Rotate suggested - применяется ли ротация к журналу

Tail sequential number - последовательный номер для конца журнала (указывает на последнее событие в журнале)

Head realtime timestamp - время первого события в журнале

Tail realtime timestamp - время последнего события в журнале

Objects - количество объектов, находящихся в журнале (таких как записи, и не только)

Entry objects - количество объектов, представляющих записи в журнале

Data objects - количество объектов данных, хранящихся в журнале

Field objects - Количество объектов полей (список полей можно получить через --fields)

Disk usage - используемое пространство на диске для этого журнала

```
nano /etc/systemd/journald.conf
```

```
Storage=auto # журналы сохраняются в /var/log/journal на диске (если доступно достаточно места), или в памяти (/run/log/journal) при недостатке м
Storage=persistent # журналы всегда сохраняются на диске
Storage=volatile # журналы хранятся только в памяти (не сохраняются на диск)
Storage=none # журналы не сохраняются
Seal=yes # включает подписание журналов для обеспечения их целостности. Это добавляет цифровую подпись в журналы, чтобы защитить их от изменений
SyncIntervalSec=5m # интервал между синхронизациями журнала с диском (5 минут)
RateLimitIntervalSec=30s # временной интервал, в течение которого будет ограничено количество записей журнала, если они приходят слишком часто
RateLimitBurst=10000 # максимальное количество записей, которое можно сделать в журнал за интервал RateLimitIntervalSec
SystemMaxUse=500M # ограничивает максимальное количество дискового пространства, которое могут занимать системные журналы, если пространство прев
SystemKeepFree=1G # минимальное количество свободного места на диске, которое должно оставаться для других системных задач, если места на диске с
SystemMaxFileSize= # ограничивает размер одного файла журнала на диске. Если файл превышает этот размер, он будет разделен
SystemMaxFiles=100 # максимальное количество файлов журнала, которые могут быть созданы, старые файлы будут удаляться, чтобы освободить место для
MaxRetentionSec=1month # максимальный срок хранения журналов (например, журналы будут храниться не более месяца)
ForwardToSyslog=yes # должны ли записи журнала перенаправляться в системный журнал (syslog), это позволяет перенаправлять журнал в другие системы
orwardToKMsg=no # должны ли записи журнала перенаправляться в буфер ядра (KMsg)
ForwardToConsole=no # должны ли записи журнала отображаться на консоли работающего в системе через TTY (не в терминал других пользователей)
ForwardToWall=yes # должны ли записи журнала отображаться всем пользователям, работающим в системе (уведомления будут выводиться всем пользовател
MaxLevelStore=debug # максимальный уровень журналируемых записей, которые будут сохраняться (emerg, alert, crit, error, warning, notice, info, de
MaxLevelSyslog=debug # максимальный уровень журналируемых записей, которые будут отправляться в syslog
MaxLevelWall=emerg # максимальный уровень журналируемых записей, которые будут выводиться всем пользователям через команду wall
LineMax=48K # максимальный размер строки, которая может быть записана в журнал (по умолчанию 48 КБ)
Audit=no # журналировать события аудита (связанных с безопасностью, например, вход в систему, попытки доступа к файлам и изменения файловых прав,
```

```
sudo systemctl restart systemd-journald
```

dmesg

`dmesg -Tx` прочитать логи буфера сообщений ядра (/var/log/dmesg), используется для записи во время загрузки системы пока сервис Syslog ещё не запущен

`dmesg -Tx -l crit,err` отфильтровать вывод

`dmesg -E` включить логирвоание ядра в консоль (--console-on)

`dmesg -D` отключить (--console-off)

`dmesg -n 1` изменить уровень логирования для печати в консоль

`dmesg -u` отображать вывод из программ окружения пользователя

`dmesg -w` выводить журнал в реальном времени (ждать новых сообщений)

hardware

`systemd-analyze` отображает статистику времени загрузки ОС (Kernel - время загрузки ядра) и userspace
`systemd-analyze blame` отобразить все процессы и отсортировать по времени загрузки
`systemd-analyze blame | grep zabbix`
`systemd-analyze plot > graph.svg` создать векторный отчет в формате Scalable Vector Graphics описанный XML

`history` история команд
`history -c` очистить историю

`who -b` время последнего включения
`last` история авторизации
`last -n 5 reboot` история перезагрузки
`last shutdown` история выключений

`arch` архитектура системы
`lsb_release -a` версия дистрибутива
`uname -srv` версия ядра
`cat /proc/version` версия ядра и дистрибутива
`cat /etc/os-release` описание дистрибутива и версия ОС
`hostnamectl` подробная информация (Operating System, Kernel, Architecture, Hardware Vendor/Model)

`uptime` время работы системы, кол-во залогиненных пользователей, Load average - средняя загрузка системы за последние 1, 5 и 15 минут (2.00 - это 100% на два ядра)
`dmidecode -t bios` информация о системе (system/baseboard/processor/memory)
`dmidecode -s bios-vendor` информация о системе (bios-version/bios-release-date/baseboard-manufacturer/system-manufacturer/processor-version)
`dmidecode -t baseboard` версия материнской платы, Video и Sound и их статус

`nproc` кол-во ядер
`lscpu` информация о процессоре
`cat /proc/cpuinfo` информация о процессоре
`cat /proc/cpuinfo | grep "core id" | wc -l` количество уникальных ядер (без учета потоков)
`cat /proc/partitions` перечисляет все устройства хранения и разделы на этих устройствах хранения
`cat /proc/asound/cards` Audio PCI
`cat /proc/cmdline` содержит имя файла образа ядра и его параметры запуска, которые были указаны в приглашении загрузчика GRUB (позволяет идентифицировать параметры загрузки, которые были введены вручную)
`cat /etc/default/grub` содержит конфигурацию, которую использует команда update-grub для создания файла /boot/grub/grub.cfg
`cat /boot/grub/grub.cfg` команда update-grub генерирует этот файл автоматически в соответствии с настройками, заданными в файле /etc/default/grub
`cat /proc/loadavg` среднее количество процессов или потоков, которые выполняются, находятся в очереди на выполнение или ждут завершения операций ввода/вывода за последние 1, 5 и 15 минут. 4-е значение, это количество процессов выполняемых в данный момент/общее количество процессов в системе. Последнее значение, это PID последнего созданного процесса.

`lspci` информация о устройствах, подключенные к материнской плате компьютера по шине PCIe
`lspci | grep -i vga` узнать какая используется видеокарта (VGA controller)
`lspci | grep -i audio` Audio controller
`lspci | grep -i ethernet` Ethernet controller
`lspci | grep -i scsi` SCSI storage controller
`lspci | grep -i sata` SATA storage controller
`lspci | grep "USB controller"`
`lspci | grep 02:00.0` фильтровать информацию по слоту устройства
`lspci -vv | grep -iE "driver"` отобразить список загруженных драйверов ядра для устройств
`lsusb -vt` информация о USB устойствах (принтеры, Bluetooth адаптер, мышка, клавиатура)

`lshw -short` информацию по каждому устройству
`lshw -class bus` Motherboard/USB
`lshw -class display` VGA controller
`lshw -class network`

`lshw -class disk` информация о жестком диске (product, vendor, size, capabilities: 7200rpm)
`lshw | grep product`

`ls /sys/class/net` список сетевых интерфейсов
`cat /proc/net/dev` список сетевых интерфейсов и их статистика (bytes, packets, errs, drop) для Receive (Прием) и Transmit (Передача)
`ethtool -S ens33` статистика сетевого интерфейса (для сброса статистики нужно ip down и выгрузить модуль ядра с драйверов modprobe -r module и вернуть обратно)
`ethtool ens33 | grep -Ei "wake-on|speed"` поддержка Wake-on-Lan и скорость сетевого интерфейса
`ethtool -i ens33` драйвер сетевой карты
`ethtool ens33 -p 100` включить светодиод на сетевой карте на 100 секунд

`cat /sys/block/sda/stat` статистика диска sda
`lsmod` список всех загруженных модулей ядра вместе с зависимостями
`/proc/modules` содержится список всех загруженных модулей ядра
`modinfo ip_tables` информация о конкретном модуле
`ls /etc/*modprobe*` содержит конфигурационные файлы со списками модулей ядра
`cat /etc/modprobe.d/mdadm.conf \` /etc/modules-load.d/` директория, которая содержит файлы со списками модулей, которые должны быть загружены при запуске системы

`ls -l /var/lib/apt/periodic/update-success-stamp` дата последнего выполнения apt update
`ls -l /var/cache/apt/pkgcache.bin` местоположение кэша пакетов apt
`HISTTIMEFORMAT="%d/%m/%y %T " history | grep "apt update"` история команды обновления с точкой времени

`cat /etc/hostname` имя хоста
`cat /etc/services | grep -iE "ntp|zabbix"` список всех сервисов и сопоставленных им портов в системе
`cat /etc/mime.types | grep -Ew "json|csv"` список сопосталвения файлов и их программ для открытия в системе

`cat /etc/hosts` локальная таблица преобразовани ip в имя
`cat /etc/hosts.allow && cat /etc/hosts.deny` ограничить доступ к внешним сервисам
`cat /etc/hosts.{allow,deny} | grep -Pv "^$|^#"`
`echo "in.telnetd: 192.168.3., .domain.ru" >> /etc/hosts.allow` разрешить соединение только для указанной подсети и домена

`ls -l /dev | grep sd` вывести список всех дисков и разделов в файловой системе
`ls -l /dev | grep -wo sd.` вывести только список дисков
`cat /proc/diskstats` статистика дисков
`cat /proc/stat` cpu user/nice/system/idle/iowait/irq/softirq/steal_time, ctxt - общее количество переключений контекста на всех процессорах, btime - время загрузки системы в секундах с начала эпохи unix, processes - указывается количество созданных процессов и потоков, включая (но не ограничиваясь ими) те, которые созданы вызовами системных вызовов fork() и clone(), procs_blocked - количество процессов, заблокированных в данный момент и ожидающих завершения ввода-вывода
`stat -f /dev/sda`
`cat /proc/buddyinfo` информация о фрагментации памяти в ядре Linux, спользуется для диагностики проблем с фрагментацией памяти
`cat /proc/cgroups` система контейнеризации и управления ресурсами доступными для процессов cgroups, позволяет ограничить доступ к любым ресурсам для процесса, а также контролировать его поведение в системе

sysctl

`sysctl -a` отобразить все параметры/настройки ядра Linux (Kernel), где представленны все параметры в виде переменных, имена переменных соответствуют пути файла в директории /proc/sys (вместо слеша в переменной используется точка)
`sysctl net.ipv6.conf.all`
`sysctl net.ipv6.conf.all.disable_ipv6=1` отключить протокол IPv6 (> /proc/sys/net/ipv6/conf/ens33/disable_ipv6)
`sysctl net.ipv6.conf.ens33.disable_ipv6=1` для интерфейса ens33
`sysctl --system` обновление информации из файлов/вернуть значения переменных до состояния сохраненного в файлах (удалить временные изменения из sysctl)
`sysctl -w net.ipv6.conf.ens33.disable_ipv6=1` сохранить настройку после перезагрузки (-w записать в файл)
`sysctl fs.file-nr` кол-во открытых файловых дескрипторов в текущий момент, открытые файлы которые сейчас не используются, максимальное количество для открытия
`sysctl -a | grep fs.file-max` максимальное количество открытых файлов (дескрипторов), которые могут быть открыты в файловой системе всеми процессами на уровне ядра ОС
`nano /proc/sys/fs/file-max` изменить значение кол-ва дескрипторов

`echo "fs.file-max=500000" >> /etc/sysctl.conf` добавить в конфигурацию sysctl.conf

`sysctl -p` применить настройки

`sysctl fs.nr_open` лимит открытия файлов для каждого процесса отдельно

`ls /proc/1/fd/ | wc -l` узнать кол-во открытых дескрипторов у процессора с PID 1

`sysctl fs.aio-nr` количество асинхронных операций ввода и вывода файловой системе в масштабе всей системы (Asynchronous IO number requests)

`sysctl fs.aio-max-nr` максимальное количество асинхронных операций ввода-вывода, рекомендуемое минимальное значение для fs.aio-max-nr — 1048576, но в загруженной среде ASE со многими ядрами может потребоваться настроить большее число

`sysctl fs.inotify.max_queued_events` подсистема ядра inotify позволяет следить за изменениями в файловой системе, устанавливает максимальное количество событий, которые могут находиться в очереди, перед тем как их обработает программа

`sysctl fs.inotify.max_user_watches` максимальное количество файлов и директорий, за которыми может наблюдать один объект inotify

`sysctl fs.inotify.max_user_instances` максимальное количество объектов inotify, которые может создать один пользователь

`sysctl fs.mqueue.queues_max` максимальное количества очередей сообщений POSIX, разрешенных в системе, которые позволяют процессам (и их потокам) обмениваться данными в виде сообщений (создаются и открываются с помощью функции mq_open)

`sysctl fs.mqueue.msg_max` максимального количества сообщений в значении очереди

`sysctl fs.mqueue.msgsize_max` максимальный размера сообщения

`sysctl vm.min_free_kbytes` минимальный размер свободной оперативной памяти который необходимо поддерживать

`sysctl vm.swappiness` процент свободной памяти, по достижении которого данные начинают переноситься на SWAP раздел

`sysctl -w vm.swappiness=80` при 80% свободной памяти (свыше 20% занятой оперативной памяти) начнет использоваться SWAP, в котоый помещяются неиспользуемые процессами страницы памяти на текущий момент, если приложению потребуются эти страницы, процесс их перенесения из раздела подкачки обратно в оперативную память (данные нужно обратно считать с диска в память)

`sysctl -w vm.swappiness=10` файл подкачки (выгрузка в виртуальную память) активируется только в том случае, если свободно 10% оперативной памяти

`sysctl vm.vfs_cache_pressure` скорость удаления dentry и inode из кэша (100 по умолчанию)

`sysctl vm.dirty_background_ratio` процент от общей оперативной памяти который может быть заполнен страничным кэшем, по достижении которой демон pdflush (dirty page flush) начинает сбрасывать данные из кэша оперативной памяти на диск. Когда объем свободной памяти становится меньше этого порога, ядро вызывает функцию wakeup_bdflush() для перевода в состояние выполнения потока pdflush, который выполняет функцию обратной записи измененных страниц памяти background_writeout() на диск, эта функция получает один параметр количества страниц, которые функция должна попытаться записать на диск.

`sysctl -w vm.dirty_background_ratio=5`

`sysctl vm.dirty_ratio` вернхний предел объема оперативной памяти в процентах от free Available который может быть выделен под PageCache до их записи на диск, на этом уровне все новые операции ввода-вывода приостанавливаются до тех пор, пока на диск не будут записаны грязные (Dirty) страницы, значение должно быть выше чем dirty_background_ratio

`sysctl vm.dirty_expire_centisecs` время хранения грязных (Dirty) страниц в кэше в сотых долях секунд (3000 = 30 секунд) для их записи на диск с целью. Функция wb_kupdate() демона pdflush выполняет обратную запись данных на диск, которые были изменены более чем dirty_expire_centisecs для синхронизации страничного кэша с данными на диске, т.к. при сбое, т.к. содержимое памяти после перегрузки не сохраняется.

`sysctl vm.dirty_writeback_centisecs` интервал процесса проверки данных, которые подлежат записи на диск (500 - 5 секунд)

`sysctl abi.vsyscall32` разрешает выполнение 32 битных программ в 64 битной системе (по умолчанию 1 - разрешает)

`sysctl kernel.hostname` изменить имя компьютера без перезагрузки

`sysctl kernel.printk` уровень логирования

`sysctl -w kernel.printk="2 4 1 7"`

`sysctl net.ipv4.ip_default_ttl` значение по-умолчанию для величины Time To Live исходящих пакетов (продолжительность жизни пакета в Internet - каждый раз, когда пакет попадает на очередной роутер, брандмауэр и т.п. величина TTL пакета уменьшается на 1)

`sysctl net.ipv4.ip_no_pmtu_disc` запрещает поиск Path Maximum Transfer Unit (максимальный размер пакета для выбранного пути, это не MTU), когда система будет пытаться определить максимальный размер пакета, при котором не потребуется выполнять их фрагментацию, для передачи на заданный хост

`sysctl net.ipv4.tcp_mem` векторная переменная (минимум, режим нагрузки, максимум), которая содержит общие настройки потребления памяти для протокола TCP, измеряется в страницах (обычно 4Кб), а не байтах. Пока общий размер памяти для целей протокола TCP ниже минимального количества страниц, операционная система ничего не делает с памятью используемой различными TCP сокетами, в режиме нагрузки TCP начинает быстро освобождать память, и последний максимальный - объем памяти, который может использоваться для нужд TCP и при его достижении, начинаются потери пакетов.

`sysctl net.ipv4.tcp_rmem` векторная величина размера буфера сокетов TCP для приема. Каждый сокет TCP имеет право использовать минимальную память по факту своего создания (по умолчанию – 4096 байт, 4 Кб) и его не стоит увеличивать, т.к. при высокой нагрузки займут много памяти. Значение по умолчанию применяется взамен параметра rmem_default (который используется другими протоколами), второй параметр - по умолчанию имеет удвоенное значение, 87380 * 2 bytes, или 174760 байт (170 Кб). Максимально возможный размер приемного буфера, это значение не отменяет максимума, заданного в rmem_max

`sysctl net.ipv4.tcp_wmem` векторная величина размера буфера сокетов TCP для передачи

`sysctl net.core.rmem_default` значение по умолчанию (имеет ниже приоритет, чем tcp_rmem)

`sysctl net.core.wmem_default` значение по умолчанию

`sysctl net.core.rmem_max` максимальный размер буфера на сокете получения данных в байтах (глобальный параметр, имеет выше приоритет, чем tcp_rmem)

`sysctl net.core.wmem_max` максимальный размер буфера на сокете передачи данных в байтах

`sysctl net.core.optmem_max` максимальный объём опциональных буферов памяти

`sysctl -w net.core.rmem_max=26214400 && sysctl -w net.core.rmem_default=26214400` увеличить до 25 МБайт

`sysctl -w net.core.wmem_max=26214400 && sysctl -w net.core.wmem_default=26214400` увеличить до 25 МБайт

`sysctl net.ipv4.tcp_no_metrics_save` по умолчанию (0) TCP сохраняет различные метрики соединения в кэше маршрута при закрытии соединения, при включении (1) TCP не будет кэшировать метрики при закрытии соединений

`sysctl net.ipv4.icmp_echo_ignore_all` если включено, ядро будет игнорировать все icmp запросы (рекомендуется для защиты от DOS атак)

`sysctl net.ipv4.icmp_echo_ignore_broadcasts` игнорировать запросы ICMP ECHO, переданные широковещательными пакетами

`sysctl net.ipv4.icmp_ignore_bogus_error_responses` игнорировать ошибочные ICMP запросы

`sysctl net.ipv4.conf.all.accept_source_route` разрешать маршрутизацию от источников, при включении, позволяет отправителю определить путь, по которому пакет должен пройти по сети Internet, чтобы достигнуть пункта назначения. Это удобно для изучения и отладки работы сети, но нарушитель получает возможность подмены адресов компьютеров локальной сети и может попытаться подсунуть поддельные маршруты для того, чтобы перенаправить весть трафик через узел, который он контролирует (атака Man In The Middle).

`sysctl net.ipv4.conf.all.accept_redirects` запретить(0)/разрешить(1) принимать и отправлять ICMP пакеты перенаправления

`sysctl net.ipv4.conf.all.send_redirects`

`sysctl net.ipv4.conf.all.secure_redirects`

`sysctl net.ipv4.ip_forward` разрешает (1) или запрещает (0) маршрутизацию пакетов через текущий хост

`sysctl net.ipv4.conf.default.forwarding` включить форвардинг пакетов - разрешить ядру операционной системы осущетсвлять проброс трафика с одного интерфейса на другой

`sysctl net.ipv4.ip_local_port_range` диапазон локальных портов, доступных для установки исходящих подключений (создания локальных клиентских сокетов)

`sysctl net.ipv4.tcp_max_tw_buckets` максимальное число сокетов, находящихся в состоянии TIME-WAIT одновременно, для предотвращения простейших разновидностей DoS-атак

`sysctl net.ipv4.tcp_tw_recycle` разрешает/запрещает быструю утилизацию сокетов, находящихся в состоянии TIME-WAIT

`sysctl net.ipv4.tcp_tw_reuse` позволять повторное использование TIME-WAIT сокетов в случаях, если протокол считает это безопасным

`sysctl net.ipv4.tcp_rfc1337` защита от TIME-WAIT атак

`sysctl net.ipv4.tcp_max_orphans` максимальное число "осиротевших" TCP сокетов, не связанных каким-либо идентификатором пользовательского файла (user file handle), при достижение этого значения, соединения сбрасываются. Этот порог помогает предотвращать простые атаки DoS и увеличение параметра влияет на ОЗУ, каждое orphan-соединение поглощает около 64 Кбайт не сбрасываемой на диск (unswappable) памяти и не может быть сброшена в SWAP. При возникновении проблем, связанных с этим ограничением – в системный журнал будет подобное сообщение: TCP: too many of orphaned sockets, и это может служить поводом пересмотреть значения tcp_fin_timeout или tcp_orphan_retries.

`sysctl -w net.ipv4.tcp_max_orphans=65536`

`sysctl net.ipv4.tcp_orphan_retries` число попыток закрыть соединение перед тем как оно будет разорвано принудительно и уничтожается TCP соединение, закрытое на локальной стороне сервера. По умолчанию используется значение 7, соответствующее приблизительно периоду от 50 секунд до 16 минут в зависимости от RTO (Retransmission Timeout).

`sysctl net.ipv4.tcp_fin_timeout` задает максимальное время пребывания сокета в состоянии FIN-WAIT-2 и используется если другая сторона не закрыла соединение со своей стороны. Каждый сокет занимает в памяти 1.5 Кб, что может привести к значительным утечкам памяти в некоторых случаях.

`sysctl net.ipv4.tcp_syncookies` помогает защититься от атак SYN flood, срабатывает только при достижении значения net.ipv4.tcp_max_syn_backlog, если количество SYN пакетов забивает всю очередь, включается механизм Syn cookies. SYN cookies вообще не использует очередь SYN, вместо этого ядро отвечает на каждый SYN пакет, как обычно SYN/ACK, но туда будет включено специально сгенерированное число на основе IP адресов и портов источника и получателя, а также времени посылки пакета. Атакующий никогда не получит эти пакеты, а поэтому и не ответит на них. При нормальном соединении, будет послан третий пакет, содержащий число, а сервер проверит был ли это ответ на SYN cookie и, если да, то разрешит соединение даже в том случае, если в очереди SYN нет соответствующей записи.

`sysctl net.ipv4.tcp_fastopen` помогает уменьшить задержки в сети, позволяя начать передачу данных сразу при отправке клиентом первого TCP SYN (3 - включает для входящих и исходящих)

`sysctl net.ipv4.tcp_max_syn_backlog` размер очереди (максимальное число) запоминаемых запросов на попытку установки TCP соединений (SYN-пакета в состоянии Waiting Acknowledgment) при отправки клиентом TCP SYN пакета, для которых не было получено сервером подтверждения от клиента (полуоткрытых соединений)

`sysctl -w net.ipv4.tcp_max_syn_backlog=4096` увеличить, если на сервере возникают перегрузки

`sysctl net.core.somaxconn` размер очереди (максимальное число) полуоткрытых соединений (открытых сокетов) ожидающих установки соединения. Если в ответ на SYN-пакета (synchronize) клиентом был получен от сервера пакет SYN-ACK (acknowledges), сервер ожидает от

клиента отправки ACK пакета, после чего соединение считается установленным.

`sysctl net.ipv4.tcp_syn_retries` количество попыток передачи SYN-пакета при установлении нового соединения, на каждую попытку отводится примерно 30-40 секунд. Значение по-умолчанию 5 = 180 секундам.

`sysctl net.ipv4.tcp_synack_retries` количество попыток передачи SYN-ACK-пакета в ответ на SYN-запрос для установки пассивного TCP-соединение, инициированное другим хостом, если уменьшить до одного, будет примерно 9 секунд

`sysctl net.core.netdev_max_backlog` регулирует размер очереди пакетов между сетевой картой и ядром, если ядро не успевает обрабатывать пакеты (если сетевой интерфейс получает пакеты быстрее, чем ядро может их обработать) и очередь переполняется, то новые пакеты отбрасываются, если увеличить значение, можно справиться с пиковыми нагрузками

`sysctl -a | grep net.ipv4.tcp_keepalive` после неактивности сокета посылает пакет keepalive на второую сторону, содержащий нулевые данные, после отправки первого пакета через время, указанное в tcp_keepalive_time отправляет повторно пакеты через каждые tcp_keepalive_intvl секунд tcp_keepalive_probes раз, если другая сторона не отвечает, сокет автоматически закрывается

`tcp_keepalive=$(sysctl -a | grep net.ipv4.tcp_keepalive | grep -Po "(?<=\=\s)[0-9]+")` забрать массив значений

`echo $tcp_keepalive | awk '{print $3+($1*$2)}'` 7200+(75*7)

`sysctl net.ipv4.conf.all.rp_filter` 1 - строгий режим проверки и 2 - свободный режим проверки, включает фильтр обратного пути (reverse path filter) или защита от подмены адресов (спуфинга), все что поступает на сервер, проходит проверку на соответствие исходящего адреса с таблицей маршрутизации и такая проверка считается успешной, если принятый пакет предполагает передачу ответа через тот же самый интерфейс. Например, когда входящий трафик идет через один маршрутизатор, а исходящий через другой, могут теряться пакеты, поскольку обратный маршрут в таблице маршрутизации, задан через другой интерфейс.

`sysctl -w net.ipv4.conf.ens33.rp_filter=1` включает строгую проверку на интерфейсе ens33

`sysctl net.ipv4.conf.all.log_martians` включает/отключает логирование пакетов

`sysctl net.ipv4.tcp_window_scaling` разрешает/запрещает масштабирование TCP-окна, как определено в RFC 1323. При передаче TCP-пакетов по толстым каналам возникают потери пропускной способности из-за того, что они не загружены полностью во время ожидания подтверждения о приеме предыдущего TCP-окна. Основная проблема состоит в том, что окно не может иметь размер больше, чем 216 байт (65 Кб). Разрешая масштабирование TCP-окна можно увеличить его размер и таким образом уменьшить потери пропускной способности.

`sysctl net.ipv4.tcp_retries2`

`sysctl net.ipv4.tcp_abort_on_overflow` заставляет ядро отвергать новые соединения, если их поступаемое количество выше, с чем система в состоянии справиться

`sysctl net.ipv4.ip_nonlocal_bind` позволяет отдельным локальным процессам выступать от имени внешнего (чужого) IP адреса, может потребоваться, когда необходимо прослушивать внешние IP адреса, например, сниффинг чужого траффика

`net.ipv4.ipfrag_low_thresh` максимальный объем памяти, выделяемый под очередь фрагментированных пакетов в диапазоне от 0 до 2147483647, когда длина очереди достигает этого порога, то обработчик фрагментов будет отвергать все фрагментированные пакеты и после уменьшения очереди они должны быть повторно переданы узлом-отправителем.

`sysctl net.ipv4.netfilter.ip_conntrack_max` максимальное количество соединений для работы механизма connection tracking (используется в iptables)

limits

```
cat /etc/security/limits.conf | grep -Ev "^$|^#"
```

```
 <user/@group>  <soft/hard>  <core/rss/as/nproc/cpu>  <value>
 @zabbix           soft              nofile          65535 # установить soft ограничение на кол-во открытых файлов (nofile) для группы zabbix (мож
 @zabbix           hard              nofile          65535 # ограничение hard можно менять только в меньшую сторону от имени обычного пользователя
 *                 soft              nofile          2048 # ограничение для всех пользователей (-n)
 *                 hard              nofile          8192
 zabbix            soft              as              100 # максимальное кол-во оперативной памяти в КБ (-m)
 zabbix            hard              as              100
 *                 soft              msgqueue        unlimited # снятие ограничения очереди сообщений памяти (-q)
 *                 hard              msgqueue        unlimited
 *                 soft              nproc           unlimited # ограничение на количество процессов для всех пользователей (-u)
 *                 hard              nproc           unlimited
 user              hard              maxlogins       1 # ограничить количество SSH-соединений/сессий для конкретного пользователя (максимальное ко
 *                 hard              maxsyslogins    1 # ограничить общее количество сеансов/активных соединений SSH (за исключением root)
```

`ulimit -a` отобразить список ограничений

```
core file size              (blocks, -c) 0
data seg size               (kbytes, -d) unlimited
scheduling priority                 (-e) 0
file size                   (blocks, -f) unlimited
pending signals                     (-i) 15052
max locked memory           (kbytes, -l) 496180
max memory size             (kbytes, -m) unlimited
open files                          (-n) 1024
pipe size                (512 bytes, -p) 8
POSIX message queues         (bytes, -q) 819200
real-time priority                  (-r) 0
stack size                  (kbytes, -s) 8192
cpu time                   (seconds, -t) unlimited
max user processes                  (-u) 15052
virtual memory              (kbytes, -v) unlimited
file locks                          (-x) unlimited
```

`ulimit -Sn` отобразить значение текущего ограничения Soft (-S) для nofile (-n)
`ulimit -Hn` ограничение Hard (-H)
`ulimit -n 3000` изменить ограничение количества открытых файлов для одного процесса (до перезагрузки)
`ulimit -Sm 1500000` ограничение soft (-S) оперативной памяти (-m) в 1500 Мб для пользователя
`ulimit -u 5000` ограничение максимального количества запущенных пользовательских процессов (-u)
`ulimit -s` ограничение места для размера аргументов (stack size) команды/скрипта (bash: /usr/bin/diff: Argument list too long)
`ulimit -m` максимальный объем оперативной памяти
`ulimit -v` максимальный объем виртуальной памяти
`ulimit -f` максимальный размер создаваемых файлов
`ulimit -t` максимальное количество процессорного времени

`systemctl edit rsyslog` ограничения на уровне Unit для конкретного сервиса

```
[Service]
LimitNOFILE=1617596
LimitNOFILESoft=1617596
```

`systemctl restart rsyslog`
`pid=$(ps -A | grep rsyslogd | awk '{print $1}')` получить pid процесса
`cat /proc/$pid/limits` проверить применение ограничений после перезапуска сервиса

quota

`nano /etc/fstab` примонтировать раздел на который необходимо установить квоту с указанными опциями

```
/dev/sda  /  ext4  defaults,usrquota,grpquota  0 0
```

`mount -o remount,rw /` перемонтировать файловую систему в режиме read and write
`mount | grep quota`
`quotacheck -favugm` выполнить проверку наличия служебных файлов aquota.user и aquota.group — если их нет, команда их создаст автоматически
`quotaon -avug` включить квоту
`edquota -u lifailon` создать квоту для пользователя или для группы (-g) на размер данных и кол-во файлов

```
Disk quotas for user lifailon (uid 1000):
  Filesystem                          blocks       soft       hard     inodes     soft     hard
  /dev/mapper/ubuntu--vg-ubuntu--lv     397112       400M       500M       3004        0        0
```

`edquota -p lifailon user` скопировать квоту на другого пользователя
`edquota -t` изменить период отсрочки soft квоты до момента, когда она станет hard (по умолчанию 7 дней)
`quota lifailon -s` отобразить квоты для пользователя
`repquota -us /` отчет для пользователей и групп (-u/-g), текущий used и soft/hard для Space limits и File limits, +-/-+/++ означает, что один из пределов достигнут максимума

```
 lifailon  --     388M     400M     500M     3004     0     0
```

```
su lifailon
dd if=/dev/zero of=/tmp/test.file bs=1024000 count=400
```
создать файл размером 400MB

```
 dd: error writing '/tmp/test.file': Disk quota exceeded
 117645312 bytes (118 MB, 112 MiB) copied, 0.158031 s, 744 MB/s
```

```
ls -lh /tmp/test.file
```

```
 -rw-rw-r-- 1 lifailon lifailon 113M Sep 26 14:37 /tmp/test.file
```

Bearstech

pussh

Pussh — инструмент для параллельного выполнения команд через SSH на нескольких хостах одновременно, выводя результаты с указанием имени каждого хоста. Был внутренним инструментом Bearstech (хостинг-провайдер в Париже, Франция) примерно с 2008 года.

```
mkdir -p $HOME/.local/bin
sudo curl -s https://raw.githubusercontent.com/bearstech/pussh/refs/heads/master/pussh -o $HOME/.local/bin/pussh
sudo chmod +x $HOME/.local/bin

bash pussh -h root@192.168.3.102,root@192.168.3.103 uname -a

echo -e "root@192.168.3.102\nroot@192.168.3.103" > host.list
pussh -f host.list uname -a
```

quickbench

quickbench - скрипт без зависимостей для оценки базовой производительности.

```
curl -sSL https://raw.githubusercontent.com/bearstech/quickbench/refs/heads/main/quickbench | bash
```

fetch

Набор скриптов, для быстрого получения информации о системе без установки:

```
curl -s https://raw.githubusercontent.com/dylanaraps/neofetch/refs/heads/master/neofetch | bash
curl -s https://raw.githubusercontent.com/dylanaraps/pfetch/refs/heads/master/pfetch | bash
curl -s https://raw.githubusercontent.com/KittyKatt/screenFetch/refs/heads/master/screenfetch-dev | bash
curl -s https://raw.githubusercontent.com/ThatOneCalculator/NerdFetch/refs/heads/main/nerdfetch | bash
curl -s https://raw.githubusercontent.com/m0zgen/system-checks/master/system-check.sh | sudo bash
curl -s https://raw.githubusercontent.com/Lifailon/hwstat/refs/heads/rsa/hwstat.sh | bash
```

```
curl -s -L "https://github.com/fastfetch-cli/fastfetch/releases/download/2.40.1/fastfetch-linux-amd64.deb" -o /tmp/fastfetch.deb
sudo dpkg -i /tmp/fastfetch.deb && rm /tmp/fastfetch.deb
fastfetch
```

networkmanager

```
apt install network-manager
systemctl status NetworkManager
nmcli device status
```
состояние интерфейсов

```
nmcli general status
```

`nmcli connection show` список доступных подключений (ethernet, vpn и WiFi-сетей)
`nmcli device wifi list` список доступных Wi-Fi-сетей
`nmcli connection show "Проводное соединение 2"` информация о сети
`nmcli connection up "Проводное соединение 2"` подключиться
`nmcli conn down "Проводное соединение 2"` отключиться
`nmcli radio wifi` состояние Wi-Fi
`nmcli connection add con-name "dhcp" type ethernet ifname ens33` создать подключение, передать тип устройства ethernet (Проводное соединение) и ifname, название сетевого интерфейса
`nmcli conn modify "dhcp" ipv4.dns 8.8.8.8` настройки подключения (modify)
`nmcli radio wifi on` включить или выключить (off) Wi-Fi
`nmcli device wifi connect "TP-Link" password 12345678 name "TP-Link Wifi"` подключиться к Wi-Fi сети
`nmcli networking off` отключить сеть через (если управление через Network Manager, указывается в блоке конфигурации renderer для netplan)
`nmcli networking on` включить сеть
`systemctl restart NetworkManager`

wireless

```
apt install wireless-tools
iwconfig
apt install iw
iw list
apt install wavemon
```
`wavemon` отобразить качество соединения и мощность передатчика

networking

```
nano /etc/network/interfaces
```

`auto ens33` активировать интерфейс при загрузке
`iface ens33 inet static` статический

```
address 192.168.1.50/24
#netmask 255.255.255.0
gateway 192.168.1.254
dns-nameservers 8.8.8.8 1.1.1.1
```

`auto ens33`
`iface ens33 inet dhcp` динамический

`service networking restart` перезагрузка сети
`systemctl restart networking.service`

netplan

`netplan --debug generate` проверка конфигурации на ошибки
`netplan apply` применить изменения (перезапускает сеть)
`netplan get` прочитать конфигурацию

```
nano /etc/netplan/*.yaml
```

- Динамический адрес (использовать два сетевых интерфейса):

```yaml
network:
  version: 2
  ethernets:
    ens33:
      dhcp4: yes
    ens36:
      dhcp4: yes
```

- Статический адрес:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens33:
      dhcp4: no
      addresses: [192.168.3.105/24]
      routes:
        - to: default
          via: 192.168.3.1
      nameservers:
        addresses: [192.168.3.101, 8.8.8.8, 1.1.1.1]
        search: [domain.local]
```

renderer указывает, кому передать управление сетью NetworkManager (nmcli) в средах с графическим интерфейсом или networkd (networkctl)

`netplan status` в релизе netplan 0.106 от февраля 2023 (Ubuntu 23.04) может получить статус используемого renderer

- MAC и MTU:

```
network:
  ethernets:
    ens33:
      dhcp4: no
      match:
        macaddress: 54:43:32:21:10:09
        mtu: 1500
```

- Подключение к WiFi:

```
network:
  version: 2
  wifis:
    wlp33:
      dhcp4: yes
      dhcp6: no
      nameservers:
        addresses: [8.8.8.8]
      access-points:
        "wifi-ssid":
          password: "12345678"
```

- Bonding для объединения физических сетевых интерфейсов в один логический:

```yaml
network:
  version: 2
  ethernets:
    ens33: {}
    ens36: {}
  bonds:
    bond0:
      dhcp4: no
        interfaces:
          - ens33
          - ens36
          parameters:
            mode: active-backup # используется только один интерфейс, второй активируется в случае неработоспособности первого
            mode: broadcast # задействуются оба интерфейса одновременно, пакеты передают все интерфейсы
            mode: balance-rr # задействуются оба интерфейса по очереди с распределением пакетов
            mode: balance-tlb # задействуются оба интерфейса по очереди, пакеты распределяются в соответствии с текущей нагрузкой
            mode: balance-xor # задействуются оба интерфейса по очереди, распределение пакетов на основе политики хеширования
          addresses:
            - 192.168.1.150/24
          gateway4: 192.168.1.1
          mtu: 1500
          nameservers:
            addresses:
              - 8.8.8.8
```

ip

`ip a` ip addr show

`ip -s link` вывести ститстику всех сетевых интерфейсов

`ip -br a show` вывести только название интерфейса, статус работы и ip-адрес

`ip link set dev ens33 up` включить сетевой интерфейс

`ip link set dev ens33 down` выключить сетевой интерфейс

`ip link set mtu 1550 dev ens33` изменить mtu

`ip link set dev ens33 address AA:BB:CC:DD:EE:FF` изменить mac-адрес (предварительно нужно отключить интерфейс, работает до перезагрузки)

`ip addr add 192.168.3.106/24 broadcast 192.168.3.255 dev ens33` добавить адрес

`ip addr del 192.168.3.106/24 dev ens33` удалить адрес

`ip route show` отобразить таблицу маршрутизации

`ip route add 192.168.4.0 via 192.168.3.100` добавить маршрут

`ip route del 192.168.4.0 via 192.168.3.100` удалить маршрут

`ip route add 192.168.4.0 dev ens33` указать сетевой интерфейс, через который отправлять пакеты в определенную подсеть

`ip neigh show` отобразить ARP-таблицу

`ip neigh add 192.168.3.110 lladdr b0:be:76:43:21:41 dev ens33`

`ip neigh del dev end33 192.168.3.110`

`ip neigh flush` очистить ARP-таблицу

net-tools

`ifconfig ens33` up/down

`ifdown -a` выключить все сетевые интерфейсы (пропадут из списка ifconfig) и включить (ifup -a)

`ifconfig ens33 192.168.3.106 netmask 255.255.255.0 broadcast 192.168.3.255`

`ifconfig -s || netstat -i` список сетевых интерфейсов

`netstat -atu` ALL (-a) tcp (-t) и udp (-u)

`netstat -lntup` LISTEN (-l) dont resolve names (-n) и сканирует директорию /proc для вывода PID/Program name (-p)

`arp -a` таблица сопоставления ip и mac адресов

`route -e` отобразить таблицу маршрутизации

`route add -net 192.168.4.0 netmask 255.255.255.0 gw 192.168.3.100` добавить маршрут в подсеть 192.168.4.0 через шлюз 192.168.3.100

`route del -net 192.168.4.0 netmask 255.255.255.0` удалить маршрут

networkd

`systemctl status systemd-networkd`

`networkctl list` список всех адаптеров, тип и состояния

`networkctl status` статус службы, Address и Gateway адаптера, DNS-адреса и лог systemd-networkd

`networkctl status ens33` характеристики адаптера (Network File, Driver, Vendor, Model, MTU, Speed)

ss

`ss -a` All отобразить все сокеты

`ss -l` показать только прослушиваемые сокеты (LISTEN)

`ss -t` отобразить только установленные TCP соединения (ESTAB/ESTABLISHED)

`ss -ua` отобразить все открытые UDP сокеты

`ss -da` DHCP сокеты

`ss -x` отобразить только локальные UNIX соединения

`ss -r` Resolve, определять сетевые имена адресов с помощью DNS

`ss -p` Processes, показать процессы, использующие сокет

`ss -n` Numeric не определять имена служб (отображать только номер порта в числовом формате)

`ss -ltp | grep 8080`

`ss -tna | grep 22`

dns

resolv

`cat /etc/resolv.conf | grep nameserver`

`nano /etc/resolv.conf` работает до перезагрузки

`domain domain.local`

`search domain.local`

`nameserver 8.8.8.8`

`nameserver 1.1.1.1`

resolved

`networkctl status` отображает список всех настроенные DNS-серверов в системе через netplan (или другое) для всех адаптеров

`resolvectl status` в systemd 239 (ubuntu 22.04) systemd-resolve переименован в resolvectl, выводит настроенные сервера для global и список все адресов для конкретного сетевого интерфейса Link (ens33)

`resolvectl status | grep "Current DNS"` отображает текущие используемые DNS-сервер

`systemd-resolve --status` служба локального DNS сервера

`resolvectl flush-caches && systemd-resolve --flush-caches` очистить локальный кэш DNS

`journalctl -u systemd-resolved | grep -E "IN A|IN PTR|IN AAAA|IN PTR|IN MX"` логи кэша DNS

`systemctl status systemd-resolved` статус службы и его лог

`cat /run/systemd/resolve/stub-resolv.conf` файл-заглушка для демона systemd-resolved, по умолчанию nameserver 127.0.0.53, который перенаправляет обращения к локальному DNS серверу, а он, в свою очередь уже получает информацию от других серверов в интернете

`nano /etc/systemd/resolved.conf` конфигурационный файл, отвечающий за настройку DNS-серверов

Включить кэширование:

```
[Resolve]
DNS=8.8.8.8, 192.168.3.101
Cache=yes
```

`ln -svi /run/systemd/resolve/resolv.conf /etc/resolv.conf` создать симлинк для совместимости с приложениями, которые не используют библиотечные вызовы, а обращаются к DNS серверам напрямую, получая их из `/etc/resolv.conf`

`ls -la /etc/resolv.conf`

`nano /etc/resolv.conf` не управляется напрямую службой systemd-resolved, а иногда с помощью использования initscripts или NetworkManager, и любые пользовательские изменения могут быть изменены через время или после перезагрузки

```
nameserver 8.8.8.8
```

`apt install resolvconf` сервис для обновления списа адресов в /etc/resolv.conf (что бы он не перезаписывался)
`systemctl status resolvconf`

`nano /etc/resolvconf/resolv.conf.d/head`

```
nameserver 8.8.8.8
nameserver 1.1.1.1
```

dig

`dig`
`dig google.com`
`dig @1.1.1.1 google.com a` использовать DNS сервер Cloudflare для преобразования имени
`dig @9.9.9.9 google.com mx` использовать DNS сервер Quad9 для получения MX записи (A, NS, TXT)
`dig -x 8.8.8.8 @9.9.9.9` разрешить ip в имя

mtr

`mtr google.com` объединяет traceroute и ping каждого узла в трассеровке
`mtr -I ens33 google.com` указать интерфейс для проверки
`mtr -b google.com` отображать имя и ip
`mtr --tcp google.com` использовать TCP SYN-пакеты или UDP-дейтаграммы (--udp)
`mtr -s 1000 google.com` указать размер пакета
`mtr -r -c 1 google.com --json` указать кол-во ping пакетов (-c 1 и -i 2 изменить интервал) и вывести в виде отчета (--report) в формате json/xml/csv/raw

doggo

`curl -sS https://raw.githubusercontent.com/mr-karan/doggo/main/install.sh | sh` DNS cli client (https://github.com/mr-karan/doggo)
`doggo yandex.ru` запросить домен, используя настройки по умолчанию
`doggo yandex.ru MX` запросить MX записи домена
`doggo yandex.ru MX @8.8.8.8` использует указанный сервер для преобразования имен DNS
`doggo -q yandex.ru -t MX --nameserver 1.1.1.1`
`doggo yandex.ru --aa --ad` запрос с установленными флагами авторитетного ответа и аутентифицированных данных
`doggo yandex.ru --cd --do` запрос с отключенной проверкой и установленными флагами `DNSSEC OK`
`doggo yandex.ru --gp-from Germany` Запрос с использованием API Globalping из указанной локации

vnstat

`apt install vnstat` журнал часового, ежедневного и ежемесячного сетевого трафика
`systemctl status vnstat` проверить службу
`vnstat -l` мониторинг в реальном режиме
`vnstat -h` ежедневная почасовая история

netcat

`nc -zv 192.168.3.100 5985` проверить порт без попытки соединения (-z) в подробном режиме (-v)
`nc -zvn 192.168.3.100 1-1000` сканирование tcp-портов, не используя преобразование DNS (-n)
`nc -zvn 192.168.3.100 1-1000 2>&1 | grep succeeded` перенаправить вывод ошибок в stdout и отфильтровать вывод
`nc -zvnu 192.168.3.100 5550-5560` сканирование udp-портов (-u)
`nc -lp 8081` открыть сокет (чат сервер) в режиме прослушивания (-listen) с указанием номера порта (-p)
`nc 192.168.3.101 8081` подключиться к сокету (чат-клиент)

`nc -lp 8081 > out.txt` все поступившие данные на сокет записываются в файл (вместо вывода в консоль)

`cat /etc/passwd | nc -N 192.168.3.101 8081` передать содержимое файла на удаленный сокет принимающей стороны (содержимое /etc/passwd запишется в out.txt) и закрыть удаленный сокет (-N)

`nc -l -w 1 -p 8081` задать timeout (-w) ожидания, в течении которого сервер слушает запрос, если будет 0, может не успеть считать запрос, на стороне клиента timeout должен быть не ниже

`nc -w 5 -Uvl server.sock > out.txt` создать UNIX-сокет и передать вывод в файл, сокет закроется через 5 секунд (-w 5) или если будет задан параметр -N на стороне клиента

`lsblk | nc -Uv server.sock` подключиться к локальному сокету с второго терминала и отправить вывод команды в файл сокета приема

`while true; do echo -e "HTTP/1.1 200 OK\n\n$(systemd-analyze plot)" | nc -l -w 1 -p 8085; done` HTTP-сервер с выводом анализа загрузки системы

socket api

```
port=8085
while true
do
    request=$(nc -l -w 1 -p $port)
    request=$(echo "$request" | head -n 1)
    method=$(echo "$request" | cut -d " " -f 1)
    endpoint=$(echo "$request" | cut -d " " -f 2)
    if [[ $endpoint == "/api/date" ]]
        then
        response="HTTP/1.1 200 OK\nContent-Type: application/text\n\n$(date)"
    elif [[ $endpoint == "/api/disk" ]]
        then
        response="HTTP/1.1 200 OK\nContent-Type: application/json\n\n$(lsblk -e7 --json)"
    else
        response="HTTP/1.1 404 Not Found\n\n404 Not Found\n"
    fi
    echo -e "$response" | nc -l -w 1 -p $port
done
```

curl -s http://192.168.3.101:8085/api/date
curl -s http://192.168.3.101:8085/api/disk | jq .blockdevices[]

socket proxy

```
ncat -l 8080 -k --sh-exec "ncat 192.168.3.101 80"
socat TCP-LISTEN:8080,fork,reuseaddr TCP:192.168.3.101:80
```

proxy

```
sudo apt-get install -y dotnet-runtime-8.0
arch="x64" # или "arm64"
sudo curl -s -L "https://github.com/Lifailon/froxy/releases/download/0.4.0/froxy-0.4.0-linux-$arch" -o /usr/local/bin/froxy
sudo chmod +x /usr/local/bin/froxy
```

`froxy --socks 1080` запустить SOCKS прокси на порту 1080

`froxy --forward 8080` запустить HTTP/HTTPS прокси на порту 1080

`froxy --forward 8080 >> froxy.log &` запустить фоновый процесс и передать вывод логов в файл

`froxy --local 5514 --remote 192.168.3.100:514` запустить обратный прокси сервер на порту 5514, который перенаправляет на хост 192.168.3.100 и порт 514 (syslog)

`froxy --local 192.168.3.100:2121 --remote 192.168.3.101:21` TCP туннелирование для RDP

`froxy --local 127.0.0.1:8443 --remote https://example.com` принимать HTTPS трафик на порту 8443 и переадресовать на указанный URL (поддерживаются GET и POST запросы с передачей заголовков и тела запроса от клиента, для использования API запросы и прохождения авторизации на сайтах)

`froxy --local *:8443 --remote https://example.com --user admin --pass admin` слушать на всех интерфейсах и использовать авторизацию

nmap

`nmap localhost` узнать какие локальные порты прослушиваются

`nmap -sV localhost` определить какое какая ОС и ПО работает на портах и их версия

`nmap -sL 192.168.3.0/24` список хостов с разрешением имен без пинга

`nmap -sP 192.168.3.0/24` ping метод host discovery (TCP ACK SYN пакет, используя системныей вызов connect) с отображением производителя сетевой платы

`nmap -F 192.168.3.0/24` fast mode port

`nmap -A 192.168.3.100` подробное сканирование ОС и ПО (ssh на другом порту, version, ad sites, rdp-ntlm-info)

`nmap -sA 192.168.3.100` обнаружить фильтрацию пакетов fw (filtered/unfiltered) с помощью TCP ACK

`nmap -PN 192.168.3.100` сканировании защищенного хоста без ping

`nmap -sO 192.168.3.100` определить какие именно IP-протоколы доступны и их статус, если отсутствует, значит фильтруется

`nmap -PU 192.168.3.100` обойти межсетевой экран с помощью UDP-пинга

`nmap -sS 192.168.3.100` выполнить полуоткрытое сканирование (TCP SYN) без установки подключения

`nmap -sU 192.168.3.100` проверка только UDP-портов

masscan

`apt install masscan` асинхронный (отправляет пакеты SYN) сканер TCP портов (https://github.com/robertdavidgraham/masscan)

```
masscan 192.168.3.100 -p80
masscan 192.168.3.100 -p0-65535 --rate 100
masscan 192.168.3.100 -p0-65535 --rate 100 --ping-timeout 1000
masscan 192.168.3.1-100 -p80
masscan 192.168.3.0/24 -p80,443
masscan 192.168.3.100 -p80 --output-format json --output-file result.json
```

rustscan

`wget https://github.com/RustScan/RustScan/releases/download/2.0.1/rustscan_2.0.1_amd64.deb \` snap install nmap `требуется установить пакет зависимости \` apt-get install -f `разрешить зависимости \` dpkg --install rustscan_2.0.1_amd64.deb \ rustscan -a 127.0.0.1 \ rustscan -a 192.168.3.100` 32400/tcp open plex

tcp

```
function tcp-scan () {
  if [ "$1" == "" ]; then
      exit 1
  fi
  START_PORT=$2; [ -z "$START_PORT" ] && START_PORT=1
  END_PORT=$3; [ -z "$END_PORT" ] && END_PORT=65535
  PORT_PROTOCOL="tcp"
  scan_port(){
      PORT_NUMBER=$1
      PORT_SCAN_RESULT=`2>&1 echo "" > /dev/$PORT_PROTOCOL/$TARGET_NAME_OR_IP/$PORT_NUMBER | grep connect`
      [ "$PORT_SCAN_RESULT" == "" ] && echo -e $PORT_NUMBER\/$PORT_PROTOCOL' \t 'open' \t\t '`grep $PORT_NUMBER/$PROTOCOL /etc/services | head -n
  }
  TARGET_NAME_OR_IP=$1
  echo -e 'PORT \t\t STATE \t\t SERVICE'
  for PORT_NUMBER in `seq $START_PORT $END_PORT`; do
      scan_port $PORT_NUMBER
  done
}
```

```
tcp-scan 192.168.3.100 1024 5000
```

tcpdump

`tcpdump -D` список доступных сетевых интерфейсов

`tcpdump -n -i ens33 icmp` слушать icmp-пакеты от всех на указанном интерфейсе (-i) без отображения доменных имен (-n)

`tcpdump -n -i ens33 udp -e` слушать udp-пакеты и отображать MAC-адреса (-e)

`tcpdump -n -i ens33 port 8080` слушать трафик 8080 порта

`tcpdump -n -i ens33 port 80 or 443`

`tcpdump -n -i ens33 portrange 21-80`

`tcpdump -n -i ens33 ip src 192.168.3.99 and dst 192.168.3.103` отобразить ip пакеты, которые отправлены с указанного (src) ip-адреса на указанный (dst) ip-адрес

`tcpdump -n -i ens33 -X host 192.168.3.100 and port 32400` отобразить содержимое пакетов (-X) для хоста и порта

tshark

`apt install tshark`

`apt install termshark` terminal UI for tshark (https://github.com/gcla/termshark)

`tshark -D` список интерфейсов

`tshark -i 1`

`disown`

`tshark -i 1 -Y "syslog"` захват пакетов syslog (udp.port == 514)

`tshark -i 1 host 192.168.3.104` захват пакетов для конкретного IP-адреса

`tshark -i 1 net 192.168.3.0/24` захват пакетов указанной подсети

`tshark -i 1 src host 192.168.3.104` захват исходящих пакетов

`tshark -i 1 dst host 192.168.3.104` захват входящих пакетов

`tshark -i 1 dst host 192.168.3.104 and port 8086` отфильтровать по входящему хосту и порту

`tshark -i 1 dst host 192.168.3.104 and port 8086 and src host 192.168.3.99` отфильтровать по исходящему хосту

`tshark -i 1 -x dst host 192.168.3.104 and port 8086 and src host 192.168.3.101` прочитать пакеты в шестнадцатеричном формате (-x)

`tshark -i 1 -O TCP dst host 192.168.3.104 and port 8086 and src host 192.168.3.101` прочитать TCP-заголовки

`tshark -i 1 -a duration:10 -w ~/192.168.3.0.pcap` сохранить захват

`tshark -Y 'ip.addr == 192.168.3.106' -r ~/192.168.3.0.pcap` прочитать файл захвата с использованием фильтра

`tshark -Y "(ip.addr == 192.168.3.106) or (ip.addr == 192.168.3.107)" -r ~/192.168.3.0.pcap` отфильтровать по двум адресам (или)

`tshark -Y "(ip.addr == 192.168.3.104) and (tcp.port == 8086)" -r ~/192.168.3.0.pcap` отфильтровать по двум параметрам (и)

`tshark -Y "!(ip.addr == 192.168.3.104)" -r ~/192.168.3.0.pcap` исключить

`tshark -Y "not arp and not (udp.port == 53)" -r ~/192.168.3.0.pcap` отобразить весь udp-трафик, исключив ping и dns пакеты

ping

fping

`fping yandex.ru google.com` параллельная проверка доступности двух хостов

`fping -p 5 yandex.ru google.com` 5 параллельных запросов к каждому хосту

`fping -ag 192.168.3.0/24` icmp проверка все подсети

`fping < hosts.txt` произвести ping всех хостов указанных в файле с новой строки

netping

```
sudo curl -s https://raw.githubusercontent.com/Lifailon/net-tools/rsa/netping.sh -o /usr/bin/netping
sudo chmod +x /usr/bin/netping
netping 192.168.3.0
```

firewall

ufw

`systemctl status ufw`

`ufw status`

`ufw enable` включить ufw (Uncomplicated Firewall)

`ufw disable` отключить

`ufw reload` перезапустить/применить настройки

`ufw reset` сбросить настройки (отключить ufw и удалить все правила)

`ufw default deny incoming` все входящие пакеты отклонять (политика по умолчанию, какие действия будут применяться к пакетам, если они не подпадают под созданные правила)

`ufw default allow outgoing` все исходящие разрешать

`ufw allow in 22` разрешить входящий трафик на порт 22

`ufw allow out 22` разрешить исходящий трафик на порт 22

`ufw deny in 80/tcp` запретить входящий TCP-трафик на 80 порт

`ufw delete deny in 80/tcp` удалить правило

`ufw allow 161,10050,10051/tcp` открыть несколько портов

`ufw allow proto tcp from 0.0.0.0/24 to 192.168.3.100 port 3389` разрешить доступ со всех IP-адресов по TCP-протоколу к IP-адресу и порту 3389

`ufw allow from 192.168.3.0/24 to 192.168.3.110` разрешить подключение всем с подсети 192.168.1.0 к интерфейсу 192.168.1.2 (для Proxmox MGW)

`ufw allow 25/tcp` открыть для всех направлений 25 порт

`ufw limit ssh` лимт подключений к определенному порту с одного IP-адреса (для защиты от перебора), по умолчанию подключения блокируются, если пользователь пытается создать шесть и больше подключений за 30 секунд (настроить время и количество запросов можно только через iptables)

`ufw logging on` включить логирование

`ufw logging medium` выбрать уровень логирования (low/medium/high)

`cat /var/log/ufw` директория хранения логов. Синтаксис: [UFW ALLOW/BLOCK/AUDIT] IN=интерфейс OUT=итерфейс SRC=ip_источника DST=ip_назначения LEN=размер_пакета TOS=0x10 PREC=0x00 TTL=64 ID=728 DF PROTO=протокол SPT=порт_источника DPT=порт назначения LEN=размер_пакета

show

`ufw show listening` отображает все прослушиваемые порты и правила для них (с указанием очередного номера в списке: [20] allow 161,10050,10051/tcp)

`ufw show raw` все активные правила в формате iptables

`ufw show added` недавно добавленные правила

`ufw show builtins` правила, добавленные по умолчанию

`ufw show user-rules` правила, добавленные пользователем

`ufw show before-rules` правила, которые выполняются перед принятием пакета

`ufw show after-rules` правила, которые выполняются после принятия пакета

`ufw show logging-rules` правила логгирования пакетов

firewalld

`apt install firewalld`

`systemctl status firewalld`

`systemctl start firewalld`

`pkill -f firewalld` убить процесс, если при запуске failed

`firewall-cmd --state` статус работы

`firewall-cmd --reload` применить настройки (перечитать)

`firewall-cmd --list-all` список созданных правил (для services и ports)

`firewall-cmd --list-port` только открытые порты

`firewall-cmd --list-service` только открытые службы

`firewall-cmd --list-all-zones` отобразить список зон

`firewall-cmd --get-active-zones` список используемых зон

`firewall-cmd --list-all --zone=public` информация о конкретной зоне

`firewall-cmd --permanent --add-port=22/tcp` открыть 22 порт

`firewall-cmd --permanent --add-port=8000-8080/udp` открыть диапазон портов

`firewall-cmd --get-services | grep ssh` отобразить список доступных служб

`firewall-cmd --permanent --add-service=ssh` разрешить порты для сервиса ssh

`firewall-cmd --permanent --new-service=speedtest` добавить службу

`firewall-cmd --permanent --service=speedtest --add-port=80/tcp` добавить порт к службе

`firewall-cmd --info-service=speedtest` информация о службе

`firewall-cmd --permanent --add-rich-rule 'rule family="ipv4" source address="192.168.3.0/24" service name="speedtest" accept'` открыть доступ для подсети

`firewall-cmd --permanent --add-rich-rule 'rule family="ipv4" source address="192.168.3.0/24" port port="22" protocol="tcp" accept'`

`firewall-cmd --permanent --add-rich-rule="rule family='ipv4' source address='192.168.21.0/24' reject"` закрыть доступ для подсети

`firewall-cmd --list-rich-rules` список правил с условиями

`firewall-cmd --permanent --remove-port=22/tcp` удалить правило

iptables

`iptables -L -v` выводит все существующие правила для каждой цепочки

```
iptables -L | grep -E "tcp|udp"
if (( $(iptables -L | wc -l | bc) == 8)); then echo active; else echo inactive; fi
```

`iptables -A INPUT -p tcp --dport 22 -j ACCEPT` открыть входящий порт ssh

`iptables -A INPUT -p tcp -s !192.168.3.99 --dport 22 -j DROP` закрыть входящий порт ssh, исключить 192.168.3.99

`iptables-save` сохранить настройки, что бы они были активны после перезагрузки

`iptables -F` удалить все правила текущей таблицы

nftables

`nft -a list ruleset` список существующих правил

`nft list tables` список существующих таблиц

`nft flush ruleset` очистить правила

`nft add table inet filter` создать таблицу filter

`nft add chain inet filter input { type filter hook input priority 0\; }` добавить цепочку input

```
nft add rule inet filter input ct state related,established counter accept
nft add rule inet filter input iifname "lo" counter accept
```

`nft add rule inet filter input ip protocol icmp counter accept` разрешить icmp

`nft add rule inet filter input tcp dport {80, 443} counter accept` открыть порты

`nft add rule inet filter input ip saddr { 192.168.100.0/24, 1.1.1.1/32 } tcp dport 22 counter accept` открыть 22 порт для подсетей

`nft chain inet filter input { policy drop \; }` остальное блокировать

`echo "flush ruleset" > /etc/nftables.conf` очистить все правила

`nft -s list ruleset >> /etc/nftables.conf` добавить правила в конфигурацию

`systemctl enable nftables.service`

`nft delete rule inet filter input handle 5` удалить правило по номеру

`nft add rule inet filter input position 5 tcp dport 22 counter accept` добавить правило в конкретное место с номером в списке

ssh

`w` отобразить активные сессии и их активность (время/дата входа, IDLE время простоя и последняя выполняемая команда)

`who` отобразить активные сессии, время/дата входа и ip с которого подключен пользователь

`last -a` история всех последних входов пользователей в систему

`lastlog` дата последнего входа каждого пользователя в систему

`last reboot` история перезагрузки

`id -G` получить список id крупп в которых состоит текущий пользователь

`apt install xclip xsel` буфер обмена

```
cat /etc/ssh/sshd_config | xclip
xsel > sshd_config.bak
nano /etc/ssh/sshd_config
```

`Port 2121` изменить порт

`PermitRootLogin yes` включить возможность подключения пользователем root

`PasswordAuthentication no` отключить аудентификацию по паролю

`X11Forwarding yes` включить X11

`TCPKeepAlive yes` отвечает за проверку активности соединения (отправка пустых keep-alive пакетов для сохранения соединения)

`ClientAliveInterval 60` задать интервал ожидания в секундах, через который sshd запросит ответ от клиента

`ClientAliveCountMax 3` количество запросов без ответа до завешрения сеанса (ClientAliveInterval * ClientAliveCountMax = 180 секунд)

```
systemctl restart sshd
systemctl status sshd
```

keygen

`ssh-keygen -t rsa -b 4096` сгенерировать пару ключей

`id_rsa` приватный/закрытый ключ хранится на клиенте, от кого происходит подключение (для подключения без пароля имя файла должно быть по умолчанию)

`cat ~/.ssh/id_rsa.pub | xclip` публичный/открытый ключ, для передачи на сервер, куда будем подключаться3

`xsel > ~/.ssh/authorized_keys` передать содержимое публичного ключа (id_rsa.pub) на сервер, куда подключаться

x11

```
apt-get install virt-manager ssh-askpass
virt-manager
export DISPLAY=username-VirtualBox:10.0 && firefox
```

scp

`ssh-copy-id root@192.168.3.105 -p 2121` скопировать публичный ключ на удаленный сервер (добавить новой строкой), утилита будет искать в директории текущего локального пользователя файл публичного ключа и скопирует содержимое файла ключа ~/.ssh/id_rsa.pub указанному при подключение пользователю на удаленный компьютер в файл authorized_keys

`scp -P 2121 /home/lifailon/files/* lifailon@192.168.3.105:/home/lifailon/downaload/` скопировать содержимое каталога files на удаленный компьютер в директорию downaload

`scp -P 2121 -r kup@192.168.3.105:/home/lifailon/downaload /home/lifailon/files/` скачать (-r) данные с удаленного сервера на локальный

sshpass

```
hosts=(192.168.3.101 192.168.3.102 192.168.3.103 192.168.3.104)
username="lifailon"
port=2121
read -s -p "Введите пароль пользователя $username: " password
echo
for host in ${hosts[@]}; do
    sshpass -p $password ssh -p $port $username@$host "echo $(uname -n) $(free -m | grep Mem: | awk '{print $3"/"$2}')"
done
```

sudoers

`cat /etc/sudoers` конфигурационный файл настройки прав доступа утилиты sudo

`visudo` открыть sudoers в режиме проверки синтаксиса

`Defaults env_reset, timestamp_timeout=10` задать ограничение времени для sudo на 10 минут

`echo "lifailon ALL=(ALL) NOPASSWD:ALL" > /etc/sudoers.d/lifailon` создать конфигурацию пользователя для использования sudo без пароля

`chmod 644 /etc/sudoers.d/lifailon`

`lifailon ALL=NOPASSWD: /usr/bin/service memcahched restart, /usr/bin/apt-get update, /usr/bin/apt-get upgrade` разрешить перезапуск определенного сервиса, обновление списка пакетов и установку обновлений системы

`%powerusers ALL=NOPASSWD: /usr/bin/service memcahched restart` доступ на группу

`visudo --check` проверка синтаксиса и всех прав доступа (0440)

`sudo -l` проверка прав доступа (выводит список команд, которые текущий пользователь может выполнять с использованием sudo)

strace

`strace -c top -n 1 > /dev/null` показывает статистику системных вызовов программы (time - процент от времени общего выполнения, call - кол-во обращений и ошибки)

`pid=$(pidof dd)` узнать pid процесса по имени

`strace -p $pid` показывает системные вызовы процесса (читает данные из одного места с помощью вызова read и записывает в другое через write)

`strace -f -p $(pgrep -o sshd) -o ~/passwd.txt -v -e trace=write -s 64` следим за всеми процессами sshd (-f), ищем все PID sshd процессов (-p), триггер только на запись данных (-e) и ограничиваем вывод 64 байтами

`cat ~/passwd.txt | grep "[1-32][1-32]) = [1-32][1-32]"`

apt

`apt-mark showauto` список установленных автоматически пакетов
`apt-mark showmanual` список установленных пакетов вручную
`echo $(($(apt-mark showauto | wc -l) + $(apt-mark showmanual | wc -l)))` количество всех установленных пакетов
`apt list --installed` список установленных пакетов apt (Advanced Package Tool)
`apt update` обновить список всех установленных пакетов системы из источников, указанных в файле конфигурации /etc/apt/sources.list
`cat /etc/apt/sources.list | grep -Ev "^#"` список источников
`apt list --upgradable` отобразить список, для каких пакетов доступны обновления
`apt list --upgradable -a` upgradable from, installed и все доступные версии
`apt install --only-upgrade powershell` обновить один выбранный пакет
`apt --fix-broken install` исправить проблемы и ошибки с зависимостями
`apt full-upgrade` обновляет все пакеты, которые уже установлены в системе, доставляет новые пакеты зависимости и удаляет пакеты, которые устанавливались в систему и уже не используются
`apt install net-tools` установить пакет
`apt download net-tools` скачать пакет без установки
`apt install net-tools --reinstall` переустановить пакет
`apt remove net-tools` удалить пакет (конфигурационные файлы, которые были изменены в системе удалены не будут)
`apt purge net-tools` полностью удалить пакет, вместе со всеми его конфигурационными файлами
`apt policy net-tools` какая версия установленна и какие доступны
`apt install net-tools=number ver.` установить конкретную версию
`apt autoremove` очистить ненужные пакеты, которые система не использует
`apt autoclean` очистить кэш пакетов

snap

Содержат саму программу (deb-пакет), а также все её зависимости и библитотеки необходимых версий для данной программы.
`ls /snap` директория пакетов
`ls /var/lib/snapd/snaps` расположение загруженных пакетов .snap
`snap install snap-store` установка магазина приложений
`snap find nmap` поиск приложения в магазине snap
`snap info nmap` информация о пакете (его наличии, версия, дата релиза и размер)
`snap list` список установленных в системе пакетов
`snap list | sed 1d | wc -l` количество установленных пакетов
`snap list --all nmap` все доступные версии определенного пакета
`snap refresh nmap` обновить пакет до последней версии
`snap revert nmap` откатить версию до предыдущей
`snap install nmap --stable` установить конкретную версию пакета
`snap connections nmap` посмотреть доступность приложения к интерфейсам системы
`snap remove nmap` удалить пакет

dpkg

`dpkg -i spark.deb` установить пакет
`dpkg -l` список установленных deb-пакетов
`dpkg -l | wc -l` количество установленных пакетов
`dpkg -l spark` проверить, установлен ли пакет в системе и его версию
`dpkg -s spark` проверить статус пакета
`dpkg -r spark` удалить (--remove) .deb пакет
`dpkg -P spark` удалить пакет вместе с фаилами конфигурации
`dpkg -L spark` куда установлен пакет (opt/Spark)

ntp

time

`timedatectl` текущее время

`timedatectl set-timezone 'Europe/Moscow'` изменить временную зону на MSK, +0300 (изменится Local time)

`timedatectl list-timezones` список часовых поясов

`timedatectl set-ntp no` отключить NTP service

`timedatectl set-time "13:00:00"` после отключения NTP указать время в ручную

`timedatectl set-ntp yes` включить NTP service (NTP service: active)

language

`locale` установленные в системе локализации

`update-locale LANG=en_US.UTF-8` изменить локализацию

`apt-get install language-pack-en language-pack-en-base` установить пакет локализаций

```
nano /etc/default/locale
LANG=en_US.UTF-8
dpkg-reconfigure locales
```

timesyncd

`systemctl status systemd-timesyncd`

`systemctl status systemd-timesyncd | grep "Status": | sed -E "s/^.+server //; "s/.\"//"" ` узнать адрес сервера синхронизации времени

`apt install systemd-timesyncd` установить службу, если unit не запускается

`apt-get remove ntp ntpstat --purge && apt autoremove` удалить ntpd (если был установлен)

`nano /etc/systemd/timesyncd.conf`

`NTP=192.168.3.233` DC (Domain Controller)

`NTP=0.debian.pool.ntp.org 1.debian.pool.ntp.org 2.debian.pool.ntp.org 3.debian.pool.ntp.org`

`FallbackNTP=ntp.ubuntu.com` резерв

`systemctl restart systemd-timesyncd`

`timedatectl set-ntp true` включить использование systemd-timesyncd для синхронизации времени (вместо ntpd)

`timedatectl status`

ntpd

`apt install ntp` установить NTP-сервер/клиент, при установке будет удален пакет systemd-timesyncd

`systemctl status ntp`

`sntp --version`

`ufw allow 123/udp && ufw reload`

`timedatectl set-ntp false` отключить синхронизацию через systemd-timesyncd на клиенте

`nano /etc/ntp.conf`

`pool 0.ubuntu.pool.ntp.org` указать пул серверов

`server 0.ru.pool.ntp.org` указать на конкретный сервер (если это pool, возьмет один)

`restrict default kod notrap nomodify nopeer noquery limited` настройки/ограничения для локального NTP сервера

`systemctl restart ntp`

`systemctl status ntp`

`timedatectl status`

`ntpq -p` проверка синхронизации времени (+ сервер можно использовать для сверки часов, * синхронизирует сейчас, - не рекомандован, st - уровень stratum, when — когда последний раз сверялось время, delay - время задержки, offset - разница между локальным временем и временем на сервере - отстают от сервера или спешат)

top

`top -c` выводит полный путь к исполняемым файлам с ключами, вместо названия

`top -H` выводит потоки процессов

`top -i` не выводит процессы, которые не используют ресурсы процессора

`top -o %CPU` отсортировать по CPU

`top -o %MEM` отсортировать по Memory

htop

`space` выделить несколько процессов (отменить Shift+U)

`u` выбрать конкретного пользователя

`l` посмотреть файлы, которые использует процесс

`s` отобразить статистику системных вызовов (strace PID attached) F8 - AutoScroll, F4 - Filter, F9 - Stop/Start Tracing

`F4` фильтр по ключевому слову (например, cron)

`F5` древовидная структура

`F6` сортировка (PERCENT_CPU/PERCENT_MEM/USER/PRIORITY/TIME)

`F7` повысить приортите (до -20), чем меньше приоритет, тем больше процессорного времени отводится процессу

`F8` понизить приоритет (до 19)

`F9/K` действие с процессом (сигналы), для завершения процесса: 15, 2, 3, 9 или 19

`S (STATE)` состояние процесса

`R` [running or runnable] запущенные или находятся в очереди на запуск

`S` [interruptible sleep] прерываемый сон (не исполняется процессором и ждет события или условия для запуска)

`D` [uninterruptible sleep] непрерываемый сон (кратковременное состояние, которое невозможно остановить сигналом, т.к. процесс не может на него ответить)

`Z` [zombie] завершенный процесс, ожидающий пока родительский процесс примет результат

`T` остановленный сигналом SIGSTOP (-19/CTRL+Z)

`X` мертвый (не должен показываться)

bpytop

```
sudo apt install bpytop
```

`pip3 list | grep psutil` проверить пакет

`pip3 install psutil --break-system-packages` установить пакет в обход ограничений

`python3 -m venv myenv` создать виртуальное окружение

`source myenv/bin/activate` активировать виртуальное окружение

`pip install psutil` установить библиотеку для получения информации о системе

```
bpytop
deactivate
```

atop

```
apt install atop
nano /etc/default/atop
LOGINTERVAL=10
systemctl restart atop
```

`atop -g` показать общую информацию о процессе (по умолчанию)

`atop -m` показать информацию о процессах, связанных с памятью

`atop -d` показать информацию о процессах, связанных с дисками

`atop -n` показать информацию о процессах, связанных с сетью

`atop -v` показывать различную информацию о процессах (PPID родителя, пользователь/группа, дата/время)

`atop -c` показать командную строку для каждого процесса

`atop -A` сортировать процессы в порядке наибольшей активности ресурсов (автоматический режим)

`atop -C` сортировать процессы в порядке потребления процессора (по умолчанию)

`atop -M` сортировать процессы в порядке потребления памяти

`atop -D` сортировать процессы в порядке дисковой активности

`atop -N` сортировать процессы в порядке сетевой активности

`atop -E` сортировать процессы в порядке активности GPU

iftop

`apt install iftop` установить пакет

`iftop -ti ens33` использовать текстовый интерфейс без ncurses

`iftop -ts 1 -i ens33` печать одного единственного текстового вывода (-s) через 1 секунд, затем выход из системы

`iftop -tL 0 -s 1 -i ens33` количество строк (-L) для печати

`iftop -ni ens33` не преобразовывать имена хостов

`iftop -Ni ens33` не преобразовывать номера портов в сервисы

`iftop -pi ens33` работать в режиме promiscuous (показывать трафик между другими хостами в одном сегменте сети)

`iftop -bi ens33` не отображать гистограмму трафика

`iftop -Bi ens33` отображать пропускную способность в байтах

`iftop -o 10si ens33` сортировка по второму столбцу (среднее значение трафика за 10 секунд, значение по умолчанию)

iotop

```
apt install iotop
```

`iotop -o` показывать только процессы или потоки, фактически выполняющие ввод-вывод

`iotop -ou mysql` показывать активные процессы от пользователя

`iotop -P` показывать только процессы, без потоков

```
iotop -p PID
```

top other

`pip install --user glances` кроссплатформенный инструмент мониторинга системы на Python (https://github.com/nicolargo/glances)

```
glances
```

`snap install bashtop` монитор ресурсов на Bash (https://github.com/aristocratos/bashtop)

```
bashtop
```

`npm install gtop -g` панель мониторинга системы для терминала на JavaScript (https://github.com/aksakalli/gtop)

```
gtop
```

`snap install bottom` кроссплатформенный графический монитор системы и процессов на Rust (https://github.com/ClementTsang/bottom)

```
bottom
```

`curl -sL https://raw.githubusercontent.com/wimpysworld/deb-get/main/deb-get | sudo -E bash -s install deb-get && deb-get install zenith` как top, но с масштабируемыми графиками, а также использованием CPU, GPU, сети и дисков на Rust (https://github.com/bvaisvil/zenith)

```
zenith
```

ps

`apt-get install -y procps` установить пакет procps

`pstree -a` отобразить все (-a) работающие процессы (демоны) и их дочерние в виде дерева

`ps -FA` отобразить подробный вывод (-F, PPID - родительский процесс) всех (-A) работающих процессов

`ps -LFC mysqld` отобразить потоки (-L) в колонках LWP и NLWP конкретного процесса по имени (-C)

`ps -Fl` отобразить приостановленные процессы (фоновые задания &)

`ps f -F` отображает активные процессы текущего пользователя

`ps f -u root` активные процессы указанного пользователя

`ps -o pid -u lifailon` вывести только pid процессов запущенных конкретным пользователем

`ps -p 3618275` найти процесс по его PID (-p/-s)

`ps -aux --sort -rss` выбрать все процессы, кроме фоновых (-a), сопоставлять с именем пользователя (-u), все процессы вне терминала (-x) и отсортировать по RSS, добавляется %CPU и %MEM

`ps -lax` не сопоставляются идентификаторы процессов с именами пользователей, к выводу добавляется WCHAN - ресурс, которого ожидает процесс

`ps -FA --sort time` сортировать по времени работы процесса

`ps -Ao comm,user,rss,vsz,command` отфильтровать вывод по потреблению памяти, названию команды/процесса и полному вызову команды с ключами

`PRI` приоритет процесса

`NI` уступчивость процесса (nice value от 19 до -20)

`(($PRI+$NI))=((39+-20))=19`

`TTY` терминал, из под которого запущен процесс

`TIME` общее время процессора, затраченное на работу процесса (bsdtime/cputime/time) или накопленное процессорное время (пользовательское + системное)

`STIME` время запуска команды (bsdstart), если процесс был запущен менее 24 часов назад, то формат вывода будет HH:MM, если больше, то Mmm:SS (Sep 18)

`C` целочисленное значение процента времени процессора (%CPU) за время жизни процесса

`%CPU` процент времени центрального процесса выделенного процессу или использование процессорного времени деленное на время работы процесса (pcpu)

`%MEM` процент реальной памяти, используемой процессом или отношение размера резидентного набора процесса к объему физической памяти на машине (pmem)

`SZ` размер в физических страницах образа ядра процесса. Сюда входят текст, данные и пространство стека

`RSS` постоянное потребление физической памяти (Resident Set Size non-swapped), реальный размер процесса в оперативной памяти, которую процесс занял (то есть что-то сохранил в память)

`VSZ` виртуальная память (Virtual Memory Size) в килобайтах (1024-байтных единицах), которую выделили процессу, но это не означает, что он успел в эту память что-то записать

`LWP` идентификатор дочернего потока (Light-Weight Process), будет выведен текущий ID если один или первый поток

`NLWP` количество (Number) дочерних потоков (ps -LFC mysqld | sed 1d | wc -l)

`PSR` ядро процессора, на котором выполняется процесс

`STAT` R - выполняется, D - ожидает записи на диск, S - неактивен (<20 с), T - приостановлен, Z - зомби, с дополнительными флагами (W - процесс выгружен на диск, < - процесс имеет повышенный приоритет, N - процесс имеет пониженный приоритет, L - некоторые процессы блокированы в ядре, s - процесс является лидером сеанса)

`maj_flt` количество крупных страничных ошибок, произошедших с данным процессом

`min_flt` количество мелких ошибок страниц

```
ps -Ao comm,user,cputime,pcpu,pmem,sz,rss,vsz,nlwp,psr,pri,ni --sort cputime
```

kill

`kill -INT (-2) PID` прерывания с терминала, bash пошлёт сигнал SIGINT процессу (аналогично CTRL+C)

`kill -KILL (-9) PID` принудительно завершить процесс

`kill -STOP (-19) PID` остановить процесс, bash пошлёт сигнал SIGSTOP процессу (аналогично CTRL+Z)

`kill -CONT (-18) PID` продолжить остановленный процесс

procs

`snap install procs` современная замена ps, написанная на Rust (https://github.com/dalance/procs)

```
procs
```

jobs

`(ping google.com) &` запустить задачу в фоне (отображается [job] - номер задачи и PID процесса)

`jobs` отобразить список фоновых задач (+ задача активна)

`jobs -l | wc -l` получить список всех запущенных заданий

`fg 1` открыть задачу по номеру

`disown` завершить все фоновые задачи (удалить/очистить всю очередь заданий)

`disown %1` завершить последнюю (если она первая) запущенную задачу

`kill %1` завершить последнюю запущенную задачу

nohup

`nohup ping ya.ru > ping.log &` используется для запуска процесса, который продолжает работать, даже если пользователь выйдет из сеанса (например, при закрытии терминала)

`ps -ef | grep "ping ya.ru"` найти процесс

`kill $(pgrep ping)` завершить процесс

task-spooler

```
sudo apt install task-spooler
```

`tsp sleep 10 && echo ok` создать задачу

`tsp -l` отобразить список задач

`tsp -s 0` отобразить статус задачи

`tsp -t 0` вывести вывод работы задачи (в режиме `tail`)

`tsp -C` очистить все выполненные (со статусом `finished`) задачи

mem

`free -m` объем оперативной памяти и SWAP в МБайт

`swapon` точка монтирования SWAP, type, size, used, priority (берет информацию из /proc/swaps)

`ipcs -lm` объем страниц разделяемой памяти (shared memory)

`cat /proc/meminfo | grep Dirty` отобразить объем грязных (Dirty) страниц в кэше (еще не записанных на диск)

`sync` записать все кэшированные, но еще не записанные данные на диск (вместо кэша данные будут читаться из диска)

`cat /proc/meminfo | grep -iE "^cache|^buff"` объем кэша и буфера

`echo 1 > /proc/sys/vm/drop_caches` отправить сигнал на вход drop_caches для очистки страничного кэша (free buff/cache) - PageCache (сигнал 1)

`echo 2 > /proc/sys/vm/drop_caches` очистка кэша структуры файловой системы - inode, dentrie (сигнал 2)

lsof

`PID` идентификационный номер процесса, который открыл файл

`TID` идентификационный номер задачи/потока, пустой столбец означает, что это не задача а процесс

`FD` файловый дескриптор файла (r - доступ для чтения, w - доступ для записи, u - доступ для чтения и записи, -r - режим неизвестен и есть символ блокировки на чтение часть файла, R - на весь файл)

`TYPE` тип узла, связанного с файлом (REG - обычный файл файловой системы, DIR - директория, CHR - символьный файл, BLK - блочный файл, INET - Интернет-сокет, unix - доменный сокет UNIX, IPv4 - IPv4 сокет, sock - неизвестный сокет, DEL - указатель Linux для удалённого файла, LINK - файл символьной ссылки, PIPE: — способ обмена данными между процессами)

`SIZE/OFF` размер файла или смещение файла в байтах

`lsof | sed 1d | wc -l` кол-во открытых файлов/дескрипторов

`cat /proc/sys/fs/file-nr` кол-во открытых файловых дескрипторов в текущий момент, открытые файлы которые сейчас не используются, максимальное количество для открытия

`lsof +D /var/log/` отобразить каким процессом и пользователем используются файлы в каталоге (+D dir) FD: r/w/u

`dd if=/dev/zero of=~/dd-zero-file` занять файл процессом dd и Ctrl+Z остановить процесс (отправить в jobs)

`ls -lh ~/dd*`

`lsof ~/dd-zero-file` отобразить каким процессом занят файл (List Open Files)

`lsof -c dd` отобразить все файлы запущенные по имени процесса/команды (в формате wildcard)

`lsof -p 1832509` отобразить все открытые файлы по номеру PID-процесса (-p)

`lsof -c mysql` отобразить все файлы которые держит открытыми процесс по названию процесса (-c)

`lsof -c bash | grep "\.sh"` найти все запущенные скрипты

`kill -9 $(lsof -t ~/dd-zero-file)` отфильтровать для вывода уникальных номеров PID-процесса (-t) использующие файл, для их завершения (kill)

`kill -9 $(lsof -t +D /smb/backup)` убить все процессы использующие файлы в директории для дальнейшего umount /smb/backup

`lsof -u root` отобразить все файлы открытые пользователем

`lsof -u^root | wc -l` исключить пользователя (^) из поиска и отобразить кол-во открытых файлов

`lsof -i:8080` проверить открыт ли порт (-i:)

descriptor

`lsof -a -p $$ -d 0,1,2` отобразить дескрипторы текущего интерпритатора `ps $$`

`0u` STDIN — стандартный поток ввода (с клавиатуры)

`1u` STDOUT — стандартный поток вывода (на экран/в файл)

`2u` STDERR — стандартный поток ошибок

`cat test.txt 1> out.txt 2> error.txt` перенаправить успешный вывод (если файл существует) в out.txt, если ошибка в error.txt

`cat test2.txt 2> /dev/null` не выводить ошибки

`cat=$(cat test 2>&1)` используется для перенаправления стандартного вывода ошибок (stderr - standard error) в стандартный вывод (stdout - standard output) с указанием файлового дескриптора (&) вместо файла

vmstat

`cat /proc/vmstat` отображает nr_free_pages, inactive/active anon, file
`cat /proc/zoneinfo` с разбиением на зоны памяти в зависимости от ее назначения
`vmstat -V` procps-ng 3.3.17 (разработчик top)
`vmstat -s` статистика memory/swap/io/system/cpu
`vmstat -d | grep sda` статистика диска
`vmstat -D` суммарная статистика дисков
`vmstat -t 1 2` отобразить 2 отчета (суммарный и текущий) с частатой обновления 1 секунда и timestamp (-t)
`r` количество запущенных процессов (работающих или ожидающих выполнения)
`b` количество спящих процессов
`swpd` объем используемой виртуальной памяти
`free` объем свободной памяти
`buff` количество памяти, используемой в качестве буферов
`cache` объем памяти, используемой в качестве кеша
`inact` количество неактивной памяти (-a)
`active` количество активной памяти (-a)
`si` объем памяти, выгруженный с диска (/s)
`so` объем памяти, перенесенный на диск (/s)
`bi` IOPS (Input/Output Operations Per Second) блоки, полученные от блочного устройства (block input/sec)
`bo` IOPS блоки, отправленные на блочное устройство (block output/sec)
`in` количество прерываний в секунду, включая часы
`cs` количество переключений контекста в секунду
`us` время, потраченное на запуск кода, не относящегося к ядру (время пользователя)
`sy` время, потраченное на выполнение кода ядра (системное время)
`id` время бездействия
`wa` время, проведенное в ожидании ввода/вывода
`st` время, украденное из виртуальной машины

sysstat

```
apt install sysstat
```

iostat

`iostat -h` выводить данные в kb/mb/gb (avg-cpu: %user %system %idle, tps - количество запросов на чтение и запись к устройству в секунду)
`iostat -hp` вывести статистику по устройству и всех его разделам (-p)
`iostat -ky /dev/sd* 1 1 | grep -w sd.` выводить статистику в КБайт (-k), при отображении нескольких записей с заданным интервалом первый отчет со статистикой с момента загрузки системы опускается (-y)
`iostat -h /dev/sda3 -o JSON` вывод в формате JSON

mpstat

`mpstat` отобразить подробную статистику по использованию процессора по каждому ядру, и куда используются ресурсы
`mpstat -P ALL` отобразить отдельно для каждого ядра.
`mpstat 2 10` отобразить 10 раз с обновлением каждые 2 секунды
`%user` процент использования процессора программами, запущенными на уровне пользователя
`%nice` процент использования процессора программами запущенными в пространстве пользователя, с изменённым приоритетом
`%system` процент использования процессора ядром
`%iowait` процент времени затраченного на ожидание завершения операций ввода/вывода, если значение параметра слишком большое, значит много времени тратится на ожидание завершения ввода/вывода
`%steal` процент простоя виртуального процессора, пока гипервизор отдаёт мощность другому виртуальному процессору
`%idle` процент времени пока процессор не занят ничем

pidstat

`pidstat` используется для мониторинга родительских и дочерних процессов и текущих потоков

`pidstat -p ALL` вывести все активные и неактивные задачи

stress

`stress --cpu 2 --timeout 10` загрузить выбранное количество ядер в течении 10 секунд

`stress -v N` нагрузить виртуальную память

`stress --io 100 &` количество процессов нагрузки на ввод-вывод

`iostat -d /dev/sda 1`

`stress --hdd 100 &` нагрузка на диск

`vmstat 1 100`

stress-ng

`apt-get install stress-ng`

`stress-ng --sequential 0 --class io --timeout 60s --metrics-brie` тест ввода вывода

`stress-ng --hdd 5 --hdd-ops 100000` будет запущено 5 стрессоров для жёстких дисков, которые будут остановлены по завершении 100 тыс. bogo-операций

`stress-ng --cpu 1 --cpu-method matrixprod --metrics --timeout 60`

`stress-ng --sequential 0 --class memory --timeout 60s --metrics-brief`

`stress-ng --cpu 2 --io 4 --vm 1 --vm-bytes 1G --timeout 60s --metrics-brief`

smart

smartmontools

`apt install smartmontools`

`smartctl -a /dev/sda` тест диска и информация о модели и температуре

`smartctl -H /dev/sda` SMART Health Status

sensors

`apt install lm-sensors`

`sensors-detect` сканировать датчики температуры

`sensors` отобразить датчики

badblocks

`badblocks -s /dev/sda` тест на наличие нечитаемых/битых блоков/секторов на диске

hdparm

`hdparm -I /dev/sda | grep -i model` модель жесткого диска (Model Number, Serial Number, Firmware Revision)

`hdparm -v /dev/sda` кол-во секторов и настройки

`hdparm -tT /dev/sda` тест скорости работы без кэша (-t) и с кэшем (-T)

`hdparm -D /dev/sda` включение/отключение управления дефектами дисков

`hdparm -r /dev/sda` включает режим read only для диска

`hdparm -A /dev/sda` включает режим read-look-ahead, когда диск просматривается перед чтением, включена по умолчанию

`hdparm -a /dev/sda` включает режим read-ahead, когда чтение выполняется в первую очередь, позволяет улучшить производительность чтения больших объемов данных

`hdparm -b /dev/sda` остановить жесткий диск до следующего к нему обращения

`hdparm -S 1 /dev/sda` остановить вращение мотора диска до следующего к нему обращения

`hdparm -B 120 /dev/sda` настройка управления питанием Advanced Power Management (APM), чем ниже значение, тем лучше энергосбережение (255 - для отключения)

`hdparm -Z /dev/sda` отключает режим энергосбережения

`hdparm -M 128 /dev/sda` управление уровнем шума (принимает значения от 128 - более тихую работу, до 254 - высокую)

disk

`du -sh /home` отобразить общий размер указанной директории (-s)
`du -Sh /home` отобразить общий размер всех дочерних (-S) директорий по пути
`du -h /home` отображает общий размер указанной директории и подкаталогов внутри директории
`du -ah /home` отобразить общий размер директории и всех дочерних файлов (-a) и директорий

`lsblk` отображает список всех подключенных блочных устройств (/dev), их SIZE, TYPE (disk/part/lvm) и точку монтирования (MOUNTPOINTS)
`lsblk -e7` вывод без loop
`lsblk -f` отображает используемую FSTYPE, UUID, FSAVAIL - сколько свободно на диске и FSUSE - сколько занято на диске в процентах
`lsblk -o NAME,MODEL,SERIAL,SIZE,STATE --nodeps | grep running` отображает модель, размер и статус без структуры разделов/lvm (--nodeps)
`lsblk -E NAME` исключить дублирование вывода по колонке
`lsblk -S` вывод информации о SCSI-устройствах (--scsi)
`lsblk -b` выводить SIZE в байтах (--bytes)
`lsblk -J` вывод в формате JSON (--json)
`lsblk -P` использовать формат вывода key="value"
`lsblk -m` выводить информацию о правах доступа

`df -h` выводит информацию примонтированных файловых системах. Отображает общий объём (Size), занятого (Used) и свободно (Avail) пространства
`df -h -T` отобразить Type файловой системы (ext4/cifs)

`apt install duf` установить duf
`duf` аналог df

`mount | grep /dev/` отобразить все примонтированные файловые системы
`mount | grep -P "mapper|lv|vg"` примонтированные lvm

`findmnt` отобразить список смонтированных файловых систем в древовидном формате
`findmnt -l` в формате списка
`findmnt -t ext4`

`e2label /dev/sda3` узнать метку диска
`blkid` отобразить список подключённых дисков, их UUID и TYPE
`lsscsi` отобразить параметры SCSI устройств подключенных к системе

parted

`parted -l` отобразить список всех разделов на дисках
`parted -l | grep -i model`
`dd if=/dev/zero of=/tmp/disk.img count=1000 bs=1M` создать образ диска заполненный нулями размером 1Гб
`parted /tmp/disk.img` передать parted созданный файл-образ для управления ФС
`parted /dev/sdc` передать parted диск
`mktable gpt` создать таблицу разделов GPT
`print` отобразить тип таблицы (Partition Table: GPT) и список разделов на устройстве, если они были созданы
`print free` отобразить свободное место и все разделы
`mkpart primary ext4 0 500M` создать первый, первичный (primary) раздел с ФС ext4 размером 500Мб.
`mkpart primary ext4 500 1000M` создать второй раздел (начало и конец)
`resizepart 2 600M` уменьшить 2-й раздел до 100МБ (указывается end-конец)
`resizepart 2 100%` увеличить до всего свободного размера
`rm 2` удалить раздел

fdisk

`fdisk -l` отображает список всех подключенных устройств построчно с размером секторов для каждого раздела
`fdisk -x` подробный вывод (узнать UUID разделов)
`fdisk -l | grep /dev/sd`
`fdisk -l | grep -E "/dev/sd.[1-9]"`
`fdisk -l | grep -E "lv|vg"`
`fdisk /dev/sdc`

`m` список команд

`p` отобразить размер диска и список разделов (/dev/sdc1)

`n` создать новый раздел (p), указать номер раздела - partition number (4-128, default 4), начало и конце сектора - enter (оставить по умолчанию)

`i` информация о выбранном разделе разделе (начало, конец, общий размер сектора и размер диска)

`t` задать тип раздела - 30/8E (Linux LVM) или 20/83 (Linux filesystem)

`l` отобразить список всех типов

`w` сохранить

`q` выход

`partprobe /dev/sdc` информирует ядро ОС об изменениях таблицы разделов, запрашивая у системы, чтобы она перечитала таблицу разделов

sfdisk

`sfdisk -d /dev/sdc > sdc.partition.table.txt` backup (аналогично dump cfdisk)

`sfdisk -f /dev/sdc < sdc.partition.table.txt` восстановить

`sfdisk -d /dev/sda | sfdisk -f /dev/sdd` для восстановления MD RAID1 (sda в sdd)

`mdadm --manage /dev/md1 --add /dev/sdd1` восстановление копирования

`watch cat /proc/mdstat` отображать прогресс синхронизации

- Новый диск для расширения LVM:

`ls /dev/sd*` отобразить все диски в файловой системе

`fdisk -l` отобразить все диски через fdisk

`cfdisk /dev/sda` разметка диска на разделы (новый вариант)

`cfdisk /dev/sdb` инициализировать новый диск, выбрать таблицу разделов (gpt)

`new - sda4` создать новый раздел sda4 или sdb1

`Free space - Partition size: 100G`

`write - yes`

`pvcreate /dev/sda4` создать физический виртуальный том из раздела

`vgextend ubuntu-vg /dev/sda4` добавить новый раздел в группу

`lvextend -l +100%FREE /dev/ubuntu-vg/ubuntu-lv` добавить свободное место в группе для логического раздела ubuntu-lv

`lsblk`

`df -h` система будет видеть старый объем диска, необходимо выполнить команду по изменению размера файловой системы

`df -T -h` отобразить тип ФС

`resize2fs -f /dev/ubuntu-vg/ubuntu-lv` для ext*

`btrfs filesystem resize +100g /` для btrfs

- Новый диск для нового раздела:

`cfdisk /dev/sdc` создать раздел (cfdisk/fdisk/parted)

`mkfs.ext4 /dev/sdc1` форматировать раздел

`mkdir /mnt/sdc1 && mount /dev/sdc1 /mnt/sdc1` примонтировать раздел, -o -r - монтировать только на чтение (--options --read-only)

`df -h && lsblk` раздел нового диска должен быть в статусе Mounted on или MOUNTPOINTS

`chmod 0777 /mnt/sdc1` разрешить всем пользователям доступ к диску

`df -h -T` отображает тип файловой системы примонтированных разделов

`nano /etc/fstab` сохранить монтирование после перезагрузкиы

`/dev/sdc1 /mnt/sdc1 ext4 rw,relatime 0 0` добавить по наименованию или UUID устройства

`umount /dev/sdc1` отмантировать раздел

swap

`fallocate -l 4G /swapfile.img` создать файл для swap

`dd if=/dev/zero of=/swapfile.img count=1024 bs=1M` создать файл для swap

`chmod 600 /swapfile.img` дать права

`mkswap /swapfile.img` создать swap-пространство из файла или используя весь объём раздела (mkswap /dev/sda4)

`swapon /swapfile.img` активировать swap-пространство

`echo '/swapfile.img none swap sw 0 0' | sudo tee -a /etc/fstab` примонтировать

`free -m` отобразить объём

`swapoff -a` отключить

`rm /swapfile.img` удалить файл

lvm

Logical Volume Management - управление логическими томами, это дополнительный слой абстракции от железа, позволяющий собрать несколько разных дисков в один, и затем разбить его на группы и разделы. Позволяет использовать программный RAID 0 и 1 (зеркалирование) с управляемым пространством, снапшотами и импортированием томов в другую систему.

- PV (Physical Volume) — физические тома

`pvs` отображает список Physical Volume
`pvdisplay` подробная информация
`pvcreate /dev/sdb` инициализация диска в LVM как физический том

- VG (Volume Group) — группы томов, для объединения физических томов и создания общего логического диска, который будет разбиватсья на разделы

`vgs` отображает список Volume Group
`vgdisplay` подробная информация
`vgcreate vg21 /dev/sdb` создать группу томов с добавлением физического тома на диске sdb

- LV (Logical Volume) — логические разделы

`lvs` отображает список Logic Volume и их объём
`lvdisplay` подробная информация
`lvcreate -n boot -L 1G vg21` создать первый логический раздел с наименованием boot
`lvcreate -n home -L 9G vg21` создать второй логический раздел с наименованием home
`lvcreate -n lv21 -l+100%FREE vg21` создать один логический раздел lv21 для группы томов vg21 и назначить ему весь объем диска
`mkfs -t ext4 /dev/vg21/lv21` назначить файловую систем ext4
`mkdir /mnt/lv21` создать папку для монтирования
`mount /dev/vg21/lv21 /mnt/lv21` примонтировать раздел к созданной папке
`echo "/dev/vg21/lv21 /mnt/lv21 ext4 defaults 0 0" >> /etc/fstab` добавить монтирование в автозагруку

- Увеличить

Расширение физического раздела можно сделать за счет добавление нового диска путём добавления в группу или увеличением имеющегося виртуального диска.
`pvcreate /dev/sdc` если добавлен новый диск, инициализируем (минус: если один из дисков выходит из строя, данные будут не доступны, аналогично работе RAID 0)
`pvresize /dev/sdb` если увеличин объем дискового пространства виртуального диска (resize - изменить размер физического тома)
`vgextend vg21 /dev/sdc` расширить группу vg21 за счет добавленного диска sdc (или vgdisplay)
`vgs` отобразит у какой из групп всего памяти VSize и сколько доступно памяти VFree для распредиления логическим разделам памяти, которую расширили
`lvextend -l +100%FREE /dev/vg21/lv21` добавить все свободное пространство логическому разделу
`vgs` VFree будет=0 а в lsblk объём раздела увеличится
`lvextend -L+1G /dev/vg21/lv21` добавить 1Гб от группы томов vg21 разделу lv21
`lvextend -L500G /dev/vg21/lv21` добавить до конкретного размера диска 500Гб
`lvs` проверить
`df -T` отобразить используемую ФС
`resize2fs /dev/vg01/lv01` изменить размер для файловой системы ext4

- Уменьшить

`e2fsck -fy /dev/vg21/lv21` проверка диска
`resize2fs /dev/vg21/lv21 500M` уменьшить размер ФС на 500Мбайт
`lvreduce -L-500 /dev/vg21/lv21` уменьшить размер логического тома на 500Мбайт
`vgs` добавится VFree 500m для распределения другой логической группе

`resize2fs /dev/ubuntu-vg/ubuntu-lv 1G` уменьшить размер ФС на 500Мбайт
`lvreduce -L-1000 /dev/ubuntu-vg/ubuntu-lv` уменьшить размер логического тома на 1000Мбайт (Logical volume ubuntu-vg/ubuntu-lv successfully resized)
`vgs` VFree 1000.00m

- Удалить

`umount /dev/vg21/lv21` предварительно отмантировать
`lvremove /dev/vg21/lv21` удалить логический том
`lvs` нет групп
`vgs` VSize и VFree объём совпадает
`vgremove vg21` удалить группу томов
`pvremove /dev/sdb` удалить диск sdb из LVM

- RAID 1

Новый диск sdc разбить на 2 раздела (основной sdc1, немного больше чем у целевого зеркалируемого раздела lv21 и sdc2 оставшийся объем для ведения файла журнала)
`pvcreate /dev/sdc1 /dev/sdc2` добавить оба раздела в LVM
`vgextend vg21 /dev/sdc1 /dev/sdc2` добавить в имеющиюся группу vg21 (расширить группу vg21)
`vgs` у группы vg21 VFree увеличится объём добавленных томов-разделов
`lvconvert -m1 /dev/vg21/lv21 /dev/sdb /dev/sdc1 /dev/sdc2` конвертируем логический том lv21, входящий в состав группы vg21 в зеркалируемый том (-m1), зеркалируется /dev/sdb (где находится lv21) на sdc1, а /dev/sdc2 используется для ведения файла журнала.

`lvs -a -o +devices` раздел lv21 пишет на 2 устройства

```
[lv21_rimage_0] /dev/sdb(0) # основной том
[lv21_rimage_1] /dev/sdc1(1) # зеркало
```

`lsblk`

```
sdb                      8:16    0    4G  0 disk
+-vg21-lv21_rmeta_0    253:0    0    4M  0 lvm
¦ L-vg21-lv21          253:4    0    2G  0 lvm
L-vg21-lv21_rimage_0   253:1    0    2G  0 lvm
  L-vg21-lv21          253:4    0    2G  0 lvm
sdc                      8:32    0    4G  0 disk
+-sdc1                   8:33    0    3G  0 part
¦ +-vg21-lv21_rmeta_1  253:2    0    4M  0 lvm
¦ ¦ L-vg21-lv21        253:4    0    2G  0 lvm
¦ L-vg21-lv21_rimage_1 253:3    0    2G  0 lvm
¦   L-vg21-lv21        253:4    0    2G  0 lvm
```

Извлекаем (удаляем) `sdb` (оригинальный диск)
`lsblk` отображает только два раздела (part) sdc1 и sdc2 зеркального диска без LVM, т.к. группа не активна
Командой `lvs` видим группу `vg21` и ошибки:

```
WARNING: Couldn't find device with uuid stBI99-6Qs3-B8lr-Ekaw-ahnD-oxPc-LsayLR.
WARNING: VG vg21 is missing PV stBI99-6Qs3-B8lr-Ekaw-ahnD-oxPc-LsayLR (last written to /dev/sdb).
```

`vgchange -ay vg21` активировать группу
`mount /dev/vg21/lv21 /mnt/lv21` примонтировать группу lv21

- Snapshot

`lvs`
`lvcreate -L 1G -s -n snap-1 /dev/ubuntu-vg/ubuntu-lv` предварительно нужно добавить VFree в Volume Group (Logical volume "snap-1" created)
`lvcreate -L 1G -s -n snap-1 /dev/vg21/lv21` параметр -s помечает, что 1Гб дискового пространства из группы vg21 будет использоваться для snapshot lv21
`lvs` Origin - к какому логическому тому (lv) относиться snapshot, Data% — процент исползованного объема от выделенного.
`lsblk` отображает изменения в томах разделов
`mount /dev/vg21/snap-1 /mnt/snap` содержимое снапшота можно смонтировать как обычный раздел, если отредактивароть снапшот и откатиться к нему, мы получим те данные, которые отредактировали
`lvconvert --merge /dev/vg21/snap-1` откатиться к снапшоту snap-1, понадобится перезагрузка ОС (даже если это не основной диск)

- Export/Import

`umount /dev/vg21/lv21` отмантировать

`vgchange -an vg21` деактивировать группу томов (0 logical volume в volume group "vg21")

`vgexport vg21` экспортировать группу (successfully exported)

`pvdisplay` список групп (VG Name vg21 (exported))

Переносим диск на новый компьютер:

`pvscan` сканировать группы на новой системе (PV /dev/sdb is in exported VG vg21)

`vgimport vg21` импортировать в систему (successfully imported)

`vgs и lvs` проверить группы

`vgchange -ay vg21` активировать группу (1+ lg в vg "vg21")

`lsblk` проверить подключение LVM

md

> SOFT RAID

`mdadm --zero-superblock --force /dev/sd{b,c}` занулить все суперблоки на дисках, которые будут добавлены в RAID-массив, т.к. диски могут содержать служебную информацию о других RAID (вывод: unrecognised md component device - ни один из дисков ранее не был добавлен в массив)

`mdadm --create --verbose /dev/md0 -l 1 -n 2 /dev/sd{b,c}` создать зеркальный RAID1 (-l/--leve 1) и указать кол-во дисков (-n/--raid-devices)

`mkfs.ext4 /dev/md0` форматировать в fs ext4

`mkdir /md0-sdb-sdc-raid1` создать директорию для монтирования

`mount /dev/md0 /md0-sdb-sdc-raid1` примонтировать вручную

`lsblk -o NAME,UUID | grep md*` отобразить UUID устройства

`echo "UUID=20482c47-fa11-462d-b7b8-93a342a7edf8 /md0-sdb-sdc-raid1 ext4 defaults 1 2" >> /etc/fstab` добавить в автозагрузку по UUID, т.к. после перезагрузки ОС номер может измениться на md127

`mount -a` примонтировать все файловые системы из fstab

`cat /proc/mdstat` проверить состояние всех доступных RAID-массивов: md127 : active raid1 sdc[1] sdb[0]

`mdadm -D /dev/md127` подробное (--detail) состояние массива. State: cleane (проблем нет)/degraded (диск неисправен/поврежден). Active/Working/Failed/Spare Devices - количество активных (в работе)/рабочих/нерабочих/запасных дисков в массиве. Consistency Policy - тип синхронизации после сбоя в массиве (resync - полная синхронизация после восстановления массива).

`mdadm /dev/md127 --add /dev/sdd` добавить запасной диск (в Spare Device) для горчей замены (Hot-Spare), в статусе списка дисков будет указан какой диск (State - spare /dev/sdd)

`mdadm -G /dev/md127 –raid-devices=3` расширить массив до трех дисков (добавится из запасных дисков в активный в массив)

`mdadm /dev/md127 --fail /dev/sdb` пометить рабочий диск как нерабочий (перевести диск в Failed Device) для замены на запасной из Spare Device, который начнет автоматическую синхронизацию для ввода в массив (Rebuild Status : 50% complete)

`mdadm /dev/md127 --remove /dev/sdb` удалить нерабочий диск из массива

`mdadm --stop /dev/md127` остановить/разобрать массив (предварительно umount /md0-sdb-sdc-raid1). В случае, если один из двух дисков был извлечен и не было запасных дисков, массив будет остановлен автоматически (State : inactive)

`mdadm --assemble --scan --verbose` команда просканирует все диски на наличие разобранного/развалившегося RAID-массива и самостоятельно (автоматически) попытается восстановить из них массив с изначальным именем (mdadm -D /dev/md0)

`mdadm --stop /dev/md127 && mdadm --assemble --scan && mount -a && mdadm -D /dev/md0` если в системе остался один диск (второй извлечен), RAID-массив будет востановлен (пересобран) автоматически из одного диска с статусом: clean, degraded, можно сразу добавить новый диск для автоматической синхронизации данных

`mdadm --assemble /dev/md127 /dev/sdb /dev/sdc` указать вручную из каких дисков пересобрать массив

`echo 'check' > /sys/block/md127/md/sync_action` проверять целостность данных в массиве (mdadm -D /dev/md0 - Check Status : 80% complete)

`cat /sys/block/md127/md/mismatch_cnt` вывод файла (0 - все впорядке)

`echo 'idle' > /sys/block/md127/md/sync_action` остановить проверку

tgt

> iSCSI

`apt install tgt` серверная часть называется порталом, который содержит цели (Target), каждая из которых предоставляет клиенту - инициатору (Initiator) доступ к блочным устройствам.

`dd if=/dev/zero of=/storage/disk1.img bs=1 count=0 seek=200G` создание динамического диска (разреженный файл) с максимальным размером 200 ГБ, где вместо последовательности нулей на диске хранят информацию об этих последовательностях в специальной таблице

`dd if=/dev/zero of=/storage/disk1.img bs=1M count=2048` создание фиксированного размера диск

`cp --sparse=always filename newfilename` преобразование обычного файла в разреженный

`nano /etc/tgt/targets.conf`

`nano /etc/tgt/conf.d/disk-1.conf`

```
<target iqn.2023-10.local.domain:ubuntu-target>
   backing-store /storage/disk1.img
   initiator-address 192.168.3.101
   incominguser user password1212
</target>
```

`systemctl restart tgt`

`tgtadm --mode target --op show` отобразит все подключенные цели и предоставляемые ими блочные устройства.

dd

`dd if=/dev/sr0 of=/tmp/cd.iso bs=2048` сохранить образ диска (if=источник) в файл (of=назначение) с указанием кол-ва байт для чтения и записи за один раз (2МБайт), по умолчанию используется размер блока - 512 байт (2b блока = 1024 байт, 1k = 1 КБайт/1024 байт, 1kB = 1000 байт, 1M = 1024 КБайт/1 МБайт)

`dd if=/dev/mem bs=2048 count=100` вывести содержимое оперативной памяти на экран (не использовать файл)

`dd if=/dev/zero of=/tmp/md-01 bs=4M count=256` создать файл заполненный нулями (из /dev/zero) размером 1ГБ с указанием кол-во копируемых блоков (bs*count) или очистить диск

`dd if=/dev/random of=/tmp/md-02 bs=4M count=256` создать файл размером 1ГБ заполненный рандомными цифрами

`dd if=/dev/sda of=/tmp/mbr.img bs=1b count=1` скопировать в файл первые 512 байт диска содержащие таблицу разделов MBR

`dd if=/dev/sda of=/tmp/sda.img` создать образ жесткого диска, используетася для полного backup системы (копирование раздела на двоичном уровне,включая таблицу MBR и всю пустую область диска и разделов)

backup

- Смонтировать внешний носитель для создания образа: `mount /dev/sdb1 /mnt/disk_b`
- `dd if=/dev/sda of=/mnt/disk_b/disk.img bs=5M` создать образа диска (.img) # создаем образ системы
- Подключить новый диск для записи образа на диск (`sdc`)
- `dd if=/mnt/disk_b/disk.img of=/dev/sdc bs=5M` записать образ на диск (с sdb1 на sdc):
- Извлечь физический ЖД или удалить виртуальный диск sda (системный) и `sdb` (с записью образа). Диск `sdc` с копией системы автоматически инициализируется как диск `sda` после перезагрузки системы.
- `gzip disk.img` сжать образ (все нули сожмутся полностью - удобно для хранения backup, поддерживает только .img)
  `dd if=/dev/sda conv=sync,noerror bs=5M | gzip -c  > /mnt/disk_b/disk.img.gz` создать сжатый образ системы
  `gunzip -c /mnt/disk_b/disk.img.gz | dd of=/dev/sdc` развернуть образ на диск

`nc -lp 5000 | sudo dd of=/backup/sda.img.gz` сохранение сжатого файла образа жесткого диска sdb на удаленном сервере (принимающая сторона)

`dd if=/dev/sda | gzip -c | nc 192.168.21.121 5000` на узле, у которого установлен жесткий диск (передающая сторона)

`nc -lp 5000 | gunzip -c | sudo dd of=/dev/sdb` восстановление содержимого жесткого диска из сжатого образа (записывать не на системный диск), сохраненного на удаленном сервере на локальном узле (принимающая сторона)

`cat /backup/sda.img.gz | nc my_local_host.com 5000` на удаленном сервере, на котором сохранен файл образа жесткого диска (передающая сторона)

iso

`dd if=/dev/sda3 status=progress of=/mnt/disk_b/disk.iso bs=5M` создать iso-образ (сохранить образ раздел)

`dd if=путь/к/образу.iso of=/dev/sdb1` записать ISO-образ ОС на внешнее устройство

`sync` завершить запись этой командой (что бы при извлечении не потерять часть данных)

`mount -o loop /mnt/disk_b/disk.iso /mnt/iso` примонтировать файл образа только для чтения (iso - это директория, которую предварительно нужно создать), подключается как /dev/loop6

`umount /mnt/iso` отмантировать

rdiff

`apt install rdiff-backup` на базе rsync с поддержкой инкрементных архивов используя технологию hard link, чтобы вернуться назад на заданный день

`rdiff-backup /usr/lifailon/ /backup/test/`

`rdiff-backup user@hostname::/remote-dir local-dir -v5 --print-statistics` по ssh с сервера на локальный бэкап-сервер, в ней же будет находиться директория rdiff-backup-data, которая будет содержать информацию и логи о проводимых бэкапах, а также инкременты, необходимые для отката на любой прошлый выполненный бэкап

`rdiff-backup list files --changed-since 5D local-dir` отобразить, какие файлы изменились за последние 5 дней

`rdiff-backup list files --at 5D local-dir` отобразить список файлов, которые присутствовали в архиве 5 дней назад

`rdiff-backup restore local-dir/rdiff-backup-data/increments.2023-10-29T21:03:37+03:00.dir /tmp/restore` восстановить файлы из инкримента

users

`sudo -u www-data` выполнить команду от имени другого пользователя

`su root` войти под пользователем root

`sudo su` изменить пользователя на root, при этом пользователь остается в той же директории потому, что выполняется ваш .bashrc. А также .profile пользователя root поэтому вы окажетесь в окружении root

`sudo -i` указывает утилите что нужно переключиться в консоль от имени root, при этом перемещаясь в домашний каталог root, и будет выполнен его .bashrc и .profile

`sudo /bin/bash` запускает еще одну оболочку bash от имени суперпользователя. Файлы конфигурации не читаются, но выполняется только .bashrc вашего пользователя. Вы не окажетесь в окружении root, а просто останетесь в своем окружении с правами суперпользователя

`cat /etc/passwd` список/база данных пользователей зарегистрированных в системе

`cat /etc/group` список групп

`cat /etc/shadow | grep -Ev "^.+:\*:"` пароли пользователей хранящиеся в зашифрованном виде (заданные с помощью /usr/bin/passwd), если * или ! пользователь не сможет войти в систему с использованием аутентификации по паролю, другие методы входа, как аутентификация на основе ключей или переключение на пользователя разрешены. Синтаксис: логин:пароль:последнее изменения пароля (количество дней исчисляется с 1 января 1970 года):минимальный срок действия пароля:максимальный срок действия:период предупреждения:период бездействия:срок хранения

`cat /etc/login.defs | grep -Pv "^$|^#"` настройка поведения утилиты управления пользователями и параметрами входа в систему (настройки минимального и максимального id для выдачи новому пользователю/группе, количество попыток входа, таймау, что делать с директорий пользователя при создании или удалении и т.п.)

`cat /etc/login.defs | grep "^PASS"` максимальное кол-во дней действия пароля (PASS_MAX_DAYS), минимальное количество дней допустимое между сменами пароля (PASS_MIN_DAYS), количество дней предупреждающих об истечении срока действия пароля (PASS_WARN_AGE), ограничения длины паролей (PASS_MIN_LEN/PASS_MAX_LEN), максимальное кол-во попыток входа при вводе неправильного пароля (LOGIN_RETRIES), время на вход (LOGIN_TIMEOUT), включить логирование успешных входов (LOG_OK_LOGINS), логирование неизветных имен для системы пользователей при неудачных попытках входа (LOG_UNKFAIL_ENAB)

passwd

`passwd` включить учетную запись root и задать ей пароль

`passwd username` смена пароля пользователя

`passwd -l username` заблокировать уч. запись

`passwd -u username` разблокировать уч. запись

chage

`chage -l root` информация последней смене пароля и срок действия (последняя смена пароля, Срок действия пароля, Пароль неактивен, Срок действия учетной записи, Минимальное количество дней между сменой пароля, Максимальное количество дней между сменой пароля, Количество дней предупреждения до истечения срока действия пароля)

`chage -E lifailon` установить дату истечения срока действия пользовательской учетной записи (-E)

`chage lifailon -M 30` установки минимального (-m) и максимального (-M) срока действия пароля

id

`id lifailon` узнать ID

```
uid=1000(lifailon) gid=1000(lifailon) groups=1000(lifailon),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),116(lxd)
```

`usermod -u 1022 kup` изменить UID

`groupmod -g 1022 kup` изменить GID

usermod

`usermod -L lifailon` заблокировать вход по паролю (перед паролем пользователя в файле /etc/shadow добавляется восклицательный знак)

`usermod --expiredate 1 -L lifailon` заблокировать пользователя (не будет возможности авторизоваться через su: Authentication failure)

`usermod --expiredate 2023-10-25 lifailon` задать дату блокировки

`usermod --expiredate "" -U lifailon` разблокировать пользователя

`usermod -g root lifailon` изменить основную группу пользователя

`usermod -a -G plugdev lifailon` добавить пользователя в дополнительную группу (-G), обязательно нужно использовать вместе с -a, чтобы не удалять старые

`usermod -d /root lifailon` изменить домашнюю директорию пользователя (-d)

`usermod -m -d /root lifailon` переместить домашнюю папку сохранив все содержимое (-m)

`usermod -s /usr/bin/dash lifailon` изменить оболочку по умолчанию (-s)

`usermod -u 2002 lifailon` изменить id пользователя (-u)

`usermod -l lifailon failon` изменить имя пользователя (-l)

`usermod --password "NewPassword" lifailon` изменить пароль

profile

`nano /etc/bash.bashrc` задать timeout для завершения бездействующих (idle time) SSH и локальных сессий

`nano /etc/profile` задать на уровне профиля (приоритет ниже)

```
TMOUT=1440
readonly TMOUT
export TMOUT
```

bashrc

`.bashrc` файл переменных конкретного пользователя

`.bash_profile` переменные вступают в силу каждый раз когда пользователь подключается удаленно по SSH. Если этот файл отсутствует система будет искать .bash_login или .profile

`/etc/environment` файл для создания, редактирования и удаления каких-либо переменных окружения на системном уровне. Переменные окружения, созданные в этом файле доступны для всей системы, для каждого пользователя и даже при удаленном подключении.

`/etc/bash.bashrc` файл выполняется для каждого пользователя, каждый раз когда он создает новую терминальную сессию. Это работает только для локальных пользователей, при подключении через интернет, такие переменные не будут доступны.

`/etc/profile` системный файл profile, все переменные из этого файла, доступны любому пользователю в системе, только если он вошел удаленно. Но они не будут доступны, при создании локальной терминальной сессии, то есть если вы просто откроете терминал.

useradd

`useradd -D` отобразить параметры, которые будут применены для пользователя по умолчанию

`useradd -o -u 0 -g 0 -s /bin/bash newroot` создать нового пользователя с правами root

`useradd -G adm,wheel -p password -s /bin/bash test2` разрешить пользователю читать логи и пользоваться sudo

`useradd username`

`-s` указать командную оболочку для пользователя (по умолчанию /bin/sh - без оболочки, можно указать /bin/bash)

`-b` указать базовый каталог для размещения домашнего каталога пользователя (по умолчанию /home)

`-d` домашний каталог, в котором будут размещаться файлы пользователя

`-m` создавать домашний каталог пользователя, если он не существует

`-c` комментарий к учетной записи

`-g` основная группа пользователя

`-G` список дополнительных групп

`-N` не создавать группу с именем пользователя

`-p` задать пароль пользователя

`-l` не сохранять информацию о входах пользователя в lastlog и faillog

`-o` разрешить создание пользователя linux с неуникальным идентификатором UID
`-u` идентификатор для пользователя

adduser

`adduser username` интерактивное создание пользователя, по умолчанию будет создан домашний каталог (/home/username), можно указать данные о пользователе или пропустить и задать пароль
`deluser username` удалить пользователя (каталог не удаляется)

chmod

```
     -                   ---                         ---         ---
     |                    |                           |           |
 тип файла   права доступа пользователя (владельца)   группы   всех остальных
```

`---` нет прав
`--x` разрешено только выполнение файла, как программы но не изменение и не чтение
`-w-` разрешена только запись и изменение файла
`-wx` разрешено изменение и выполнение, в случае с каталогом нельзя посмотреть его содержимое
`r--` права только на чтение
`r-x` только чтение и выполнение, без права на запись
`rw-` права на чтение и запись, но без выполнения
`rwx` все права
`--s` установлен SUID или SGID бит, первый отображается в поле для владельца, второй для группы
`--t` установлен sticky-bit, значит пользователи не могут удалить этот файл

`r` чтение
`w` запись
`x` выполнение
`s` выполнение от имени суперпользователя (дополнительный)

`u` пользователь-владелец файла
`g` группа-владельц файла
`o` все остальные пользователи

`+` включить
`-` отключить

`-R` поменять права на все подкаталоги и файлы указанной директории
`-v` выводить информацию обо всех изменениях

`chmod u+x filename` разрешить выполнение (x) для владельца (u)
`chmod ugo+x filename` разрешить выполнение (x) для всех (ugo)
`chmod ug+r filename` разрешить чтение (r) для владельца (u) и группы (g)
`chmod o-w filename` запретить запись (w) для остальных пользователей (o)
`chmod -R g+rwx dir` дать полный доступ (rwx) группе (g) на директорию и всем файлам в ней (-R)

Права доступа в восьмеричной системе, которые полностью переписывают текущие права новыми для всех категорий пользователей:

`0` никаких прав
`1` только выполнение
`2` только запись
`3` выполнение и запись
`4` только чтение
`5` чтение и выполнение
`6` чтение и запись
`7` чтение, запись и выполнение

`chmod 744 filename` разрешить полные права для владельца, а остальным только чтение
`chmod 664 filename` чтение и запись для владельца и группы, только чтение для остальных

chown

`chown lifailon tmp` изменить владельца на пользователя lifailon для директории tmp
`chown lifailon:lifailon tmp` изменить владельца и группу
`chown -R lifailon:lifailon tmp` применить изменения ко всем подкаталогам (-R)
`chown --from=root:root lifailon:lifailon -R ./` изменить владельца и группу только для тех каталогов и файлов, у которых владелец и группа root в текущем каталоге

groups

`groups lifailon` отобразить в каких группах находится указанный пользователь
`touch testdir` при создании файла ему назначается основная группа пользователя который его создал (ls -l testdir)
`cat /etc/group` список групп
`chgrp testdir tmp` изменить группу на testdir для директории tmp
`groupadd testdir` создать группу
`delgroup testdir` удалить группу, если ошибка 'testdir' still has `testdir' as their primary group!` предварительно исключить из группы всех пользователей

Опции:
`-g` изменить основную группу для пользователя
`-G` дополнительные группы, в которые нужно добавить пользователя (затирает предыдущие)
`-a` добавить пользователя в дополнительные группы с параметром -G, а не заменять им текущее значение
`-R` удалить пользователя из группы

usermod

`usermod -aG sudo lifailon` добавить пользователя в дополнительную группу (-aG, без затирания предыдущих групп) sudo (добавить в группу root)
`usermod -aG disk lifailon` пользователь будет иметь прямой доступ к ЖД без команды sudo (например монтировать)
`usermod -g root lifailon` изменить основную группу (-g) для пользователя на root
`usermod -R ssh lifailon` удалить пользователя из группы

domain

realmd

```
hostnamectl
```
`hostnamectl set-hostname srv-01.domain.local` изменить имя сервера (/etc/hostname)

```
nano /etc/resolv.conf
```
`nameserver 192.168.3.233` адрес DC
```
search domain.local
```

```
apt -y install realmd libnss-sss libpam-sss sssd sssd-tools adcli samba-common-bin oddjob oddjob-mkhomedir packagekit
```
`nano /etc/realmd.conf` задать атрибуты хоста, которые будут сохранены в учетной записи компьютера в AD (атрибуты operatingSystem и operatingSystemVersion)

```
[active-directory]
os-name = Ubuntu Server
os-version = 20.04
```

`realm discover domain.local --verbose` возвращает полную конфигурацию домена и список пакетов, которые должны быть установлены для регистрации системы в домене
`realm join --help | grep pass`
`realm join -U username domain.local` --one-time-password `ввести в домен \` realm
list `проверить после подключения (server-software: active-directory) \` id username@domain.local` получить сведения о пользователе домена

sssd

Права доступа на логирование в Linux из под УЗ домена (sssd используется для аутентификации Kerberos)

`realm permit -g 'ssh-connect-domain'` добавит доменную группу (echo "simple_allow_groups = ssh-connect-domain" >> /etc/sssd/sssd.conf)

`realm permit username@domain.local` добавит пользователя (echo "simple_allow_users = username" >> /etc/sssd/sssd.conf)

`realm deny --all` запретить доступ всем пользователям (очищает список simple_allow_* в sssd.conf)

```
systemctl restart sssd
```

Создавать домашний каталог для нового доменного пользователя:

```
bash -c "cat > /usr/share/pam-configs/mkhomedir" <<EOF
Name: activate mkhomedir
Default: yes
Priority: 900
Session-Type: Additional
Session:
required pam_mkhomedir.so umask=0022 skel=/etc/skel
EOF
```

`pam-auth-update` обновить конфигурацию, выбрать созданную: activate mkhomedir

Права на sudo:

`nano /etc/sudoers.d/domain_admins` nano /etc/sudoers.d/linux-admins

```
username@domain.local               ALL=(ALL)       ALL
%ssh-connect-domain@domain.local    ALL=(ALL)       ALL
```

```
ssh username@domain.local@hostname
```

syslog

server

```
systemctl status rsyslog
```

```
nano /etc/rsyslog.conf
```

```
# provides UDP syslog reception (input module udp)
module(load="imudp")
input(type="imudp" port="514")
# provides TCP syslog reception
#module(load="imtcp")
#input(type="imtcp" port="514")
# Filter duplicated messages
# Включить фильтрацию одинаковых сообщений
$RepeatedMsgReduction off
# Шаблон создания директории на основе IP адреса клиента и сбор всех логов в один файл:
$template RemoteLogs,"/var/log/remote/%fromhost-ip%/syslog.log"
# Сохранять сообщения от любого источника (*) с любым приоритетом (*) в файл, заданный шаблоном (RemoteLogs):
*.* ?RemoteLogs
# Шаблон создания директории на основе имени клиента и лог-файлов по имени программы:
# $template RemoteLogs,"/var/log/remote/%HOSTNAME%/%PROGRAMNAME%.log"
# *.* ?RemoteLogs
# Include all config files in /etc/rsyslog.d/
$IncludeConfig /etc/rsyslog.d/*.conf
```

```
systemctl restart rsyslog
```

client

```
nano /etc/rsyslog.d/all.conf
```

```
# UDP:
*.* @192.168.3.105:514
# TCP:
# *.* @@192.168.3.105:514
# auth.* @@192.168.3.105:514
# *.err @@192.168.3.105:514
```

```
systemctl restart rsyslog
```

`ls /var/log/remote` на сервере должна появиться директория с ip-адресом/именем клиента

```
ls -lh /var/log/remote/*
cat /var/log/remote/192.168.3.104/syslog.log | grep influxd
systemctl restart zabbix-agent
cat /var/log/remote/192.168.3.104/syslog.log | grep zabbix_agentd
```

zabbix-agent

```
nano /etc/zabbix/zabbix_agentd.conf
```

```
# LogType=file
LogType=system
DebugLevel=1
```

```
systemctl restart zabbix-agent
```

ommail

```
nano /etc/rsyslog.conf
```

```
# Включить модуль:
module(load="ommail")
$ActionMailSMTPServer m.domain.ru
$ActionMailSMTPPort 25
$ActionMailFrom zabbix@domain.ru
$ActionMailTo lifailon@domain.ru
# Шаблон, который будет подставляться как тема (Subject) и тело письма (Body) - имя переменной шаблона,"содержимое шаблона"
$template mailSubject,"rsyslog: disk problem on %hostname%"
$template mailBody,"RSYSLOG Alert\r\nmsg='%msg%'"
$ActionMailSubject mailSubject
$ActionMailBody mailBody
# Ограничение на отправку - одно письмо в 5 минут
$ActionExecOnlyOnceEveryInterval 300
```

logrotate

```
systemctl status logrotate.timer
```

```
nano /etc/logrotate.conf
```

```
# Ротация файлов журнала еженедельно
weekly
# По умолчанию используется группа adm, которая является владельцем группы ls -ld /var/log/syslog
su root adm
# Количество файлов (недель, если ротация еженедельно) хранения журналов
rotate 4
# Создавать новые (пустые) файлы журналов после ротации старых
create
# Использовать дату в качестве суффикса ротируемого файла
dateext
# Сжимать лог-файлы
compress
# Пакеты сбрасывают информацию о ротации журнала в этот каталог
include /etc/logrotate.d
```

```
nano /etc/logrotate.d/logrotate_remote.conf
```

```
/var/log/remote/*/*.log {
su root root
daily
copytruncate
size 10M
rotate 2
compress
dateext
}
```

Условия:

`hourly` каждый час
`daily` каждый день
`weekly` каждую неделю
`monthly` каждый месяц
`yearly` каждый год
`size` минимальный размер лога, меньше этого значения ротация выполняться не будет

Действия:

`rotate 2` указать, сколько последних ротированных лог-файлов нужно хранить, остальные удалять
`maxage 30` указать, за сколько последних дней хранить ротированные файлы, остальные удалять
`copytruncate` сначала создается копия файла лога, после уже обрезается действующий (нужно, когда программа должна писать лог непрерывно, возможность потери записей, если она придется на процесс усечения)
`extension` сохранять оригинальный лог файл после ротации
`compress` сжимать ротированный лог (gzip)
`delaycompress` не сжимать последний и предпоследний журнал (позволяет избежать ошибок, связанных с отсутствием доступа к используемому файлу)
`create # 0644 root root` создать пустой лог файл на месте старого
`olddir /path` перемещать логи в отдельную папку при срабатывании условия
`dateext` добавляет дату ротации перед заголовком старого лога
`missingok` не выдавать ошибки, если лог файла не существует
`notifempty` если файл пустой, не выполнять никаких действий
`prerotate script.sh endscript` скрипт, который необходимо выполнить перед чисткой лога
`postrotate script.sh endscript` скрипт, который необходимо выполнить после чистки лога
`sharedscripts` если был указан путь в формате wildcard (*), выполнить скрипт один раз после завершения ротации всех файлов

`logrotate -d /etc/logrotate.d/logrotate_remote.conf` проверить ротацию (--debug)
`logrotate -fv /etc/logrotate.d/logrotate_remote.conf` запустить ротацию сейчас (--force) с подробным выводом (--verbose)
`cat /etc/cron.daily/logrotate` задание на автоматический запуск создается по умолчанию, который читает конфигурационный файл ротации /etc/logrotate.conf, в нем указана директрия: include /etc/logrotate.d в которой лежат файлы ротации
`which logrotate` узнать путь до исполняемого файла
`crontab -e`

`00 3 * * * /usr/sbin/logrotate -f /etc/logrotate.d/logrotate_remote.conf` настроить собственное ручное расписание с ежедневным запусков в 3:00

log

- lazyjournal - терминальный пользовательский интерфейс для journalctl, журналов файловой системы, а также контейнеров Docker, Podman и Kubernetes для быстрого просмотра и фильтрации с поддержкой нечеткого поиска, регулярных выражений и раскрашивания вывода:

```
curl https://raw.githubusercontent.com/Lifailon/lazyjournal/main/install.sh | bash
lazyjournal
```

- lnav - терминальный пользовательский интерфейс для логов файловой системы с таймстампами, возможностью поиска и подсветкой:

```
apt install lnav
journalctl -f -a -xe -o json | lnav
ssh playground@demo.lnav.org
```

- tailspin - раскрашивание любого вывода:

```
apt install tailspin
cat /var/log/syslog | tailspin
```

- toolong - терминальный пользовательский интерфейс для интерактивного просмотра логов с поддержкой поиска и объединения журналов (заменяет tail, less и grep):

```
pipx install toolong
tl /var/log/syslog* откроет все журналы syslog (включая поддержку чтения архивных журналов)
tl /var/log/syslog /var/log/syslog.1 --merge объеденит два журнала в один
cat /var/log/syslog /var/log/syslog.1 | tailspin | tl вначале красим вывод и потом передаем на вход toolong \
```

- ttyd - позволяет запускать терминальные интерфейсы в Web интерфейсе с поддержкой базовой авторизации:

```
sudo apt install ttyd
ttyd -W -p 4444 tl /var/log/syslog*
ttyd -W -p 4444 -c admin:admin tl /var/log/syslog*
```

smb

cifs

`apt install cifs-utils` установить SMB Client

```
nano /root/.smbclient


 username=lifailon
 password=password
 #domain=domain.local
```

`mkdir /smb && mkdir /smb/backup` создать директорию для монтирования

```
nano /etc/fstab
//192.168.3.100/Backup /smb/backup cifs user,rw,credentials=/root/.smbclient 0 0 rw права на чтение и запись
mount -a примонтировать (открыть порты на сервере: 137/UDP; 138/UDP; 139/TCP; 445/TCP)
df -h
```

```
//192.168.3.100/torrent-files /home/lifailon/torrent-files cifs user,rw,credentials=/root/.smbclient,perms=0666 0 0
chmod 666 /home/lifailon/torrent-files
chown -R lifailon:lifailon /home/lifailon/torrent-files
smbclient $path_smb_qb --user=smb --password=kinozal
```

samba

`iptables -I INPUT -p tcp --dport 445 -j ACCEPT` используется для Samba
`iptables -I INPUT -p tcp --dport 137 -j ACCEPT` используется для работы NetBIOS (использование имени компьютера для доступа)
`iptables -I INPUT -p tcp --dport 139 -j ACCEPT`
`iptables -I INPUT -p udp --dport 137:138 -j ACCEPT`
`iptables -L` отобразить список правил
`iptables -F` очистить список правил
`apt install iptables-persistent`
`netfilter-persistent save` применить настройки

`apt install samba`
`systemctl status smbd`
`mkdir -p /public/share` создать общую папку
`chmod 777 /public/share` выдать права

`nano /etc/samba/smb.conf`

```
[global]
workgroup = WORKGROUP # рабочая группа (должна одинакова на всех машинах)

[share1]
    comment = Public folder for all
    path = /public/share
    public = yes
    writable = yes
    read only = no
    guest ok = yes
    create mask = 0777
    directory mask = 0777
    force create mode = 0777
    force directory mode = 0777
```

`[Общая папка]` имя общей папки, которое увидят пользователи при подключение
`path` путь на сервере, где будут храниться данные
`public` для общего доступа, чтобы все могли работать с ресурсом
`writable` разрешает запись в сетевую папку
`read only = yes` только для чтения. no - для полного доступа управлением ФС
`guest ok` разрешает доступ к папке гостевой учетной записи
`create mask, directory mask, force create mode, force directory mode` при создании новой папки или файла назначаются указанные права. В примере указаны полные права.

`systemctl restart smbd` применить настройки

`useradd smb1` создать пользователя
`passwd smb1` задать пароль пользователю
`smbpasswd -a smb1` создать пользователя для Samba

```
public = no
writable = no
read only = yes
guest ok = yes # разрешить анонимный доступ на чтение (read only) без пароля
; valid users = smb1 @smb-users # список пользователей, которым разрешено подключаться к каталогу для чтения, а также входщях в группу smb-users
write list = smb1 # список пользователей, которые имеют полный доступ к директории
hosts allow = comp1, 192.168.160.0/255.255.252.0 # указать список разрешенных хостов или сетей, с которых можно подключаться к серверу. Если его
hosts deny = comp2, 192.168.164.0/255.255.252.0 # запретить доступ
```

client cifs

`apt install cifs-utils`
`mkdir /mnt/share1`

`mount -t cifs "//192.168.3.103/share1" /mnt/share1 -o user=smb1` примонтировать удаленный каталог на клиенте Linux с авторизацией

```
df -handle
//192.168.3.103/share1  24G  23G  1.2G  96%  /mnt/share1
```

client samba-client

```
apt install samba-client
smbclient -L 192.168.3.103 -U share1
```

recycle

```
[share1]
    path = /public/share
    public = yes
    browseable = yes
    writable = yes
    read only = no
    guest ok = yes
    vfs objects = recycle
    recycle:repository = .recycle/%U
    recycle:keeptree = Yes
    recycle:touch = Yes
    recycle:versions = Yes
    recycle:maxsize = 0
    recycle:exclude = *.tmp
    recycle:exclude_dir = /tmp
```

`recycle:repository` где хранить удаленные объекты. Удаленные файлы попадут в скрытый каталог .recycle в котором создастся каталог с именем пользователя, удалившего файл или папку

`recycle:keeptree` удалять объекты с сохранением дерева каталогов

`recycle:touch` изменить дату изменения файла при его перемещении в корзину

`recycle:versions` при удалении файлов с совпадающими именами, добавлять номер версии

`recycle:maxsize` не помещать в корзину файлы, размер которых больше заданного параметра (в байтах). 0 - помещать файлы любого размера

`recycle:exclude` исключить файлы

`recycle:exclude_dir` исключить каталог

nfs

server

`apt install nfs-kernel-server` установить сервер, с помощью которого будет выполнено открытие шары. Сервис NFS слушает соединения для TCP и UDP на порту 2049.

`apt install rpcbind`

`rpcinfo -p | grep nfs` проверить, слушается ли порт nfs

`cat /proc/filesystems | grep nfs` проверить, поддерживается ли NFS на уровне ядра (вывод: nodev nfsd)

`modprobe nfs` вручную загрузить модуль ядра nfs

`systemctl status nfs-server` служба сервера

`ufw allow 111,2049 && ufw reload`

`mkdir nfs-folder` создать папку для шары

`adduser nfs-user` создать пользователя для подключения

`chown nfs-user:nfs-user nfs-folder` изменить владельца шары

`chmod 775 nfs-folder` дать полный доступ владульцу и группе

`nano /etc/exports` файл настройки шары

`/nfs-folder 192.168.3.0/24(rw,sync,no_subtree_check)` /путь/к/директории (шарим), удаленный IP-адрес клиента или подсеть (что бы разрешить все адреса используется 0.0.0.0/24 или символ *) и опции в скобках

`exportfs -a` применить настройки (обновить таблицу экспорта NFS)

Options:

`rw` разрешить чтение и запись в этой папке
`ro` разрешить только чтение
`sync` отвечать на следующие запросы только тогда, когда данные будут сохранены на диск (по умолчанию)
`async` не блокировать подключения пока данные записываются на диск
`secure` использовать для соединения только порты ниже 1024
`insecure` использовать любые порты
`nohide` не скрывать дочернии директории, при открытии доступа к нескольким директориям
`root_squash` подменять запросы от root на анонимные (используется по умолчанию)
`no_root_squash` не подменять запросы от root на анонимные (все подключения от имени пользователя root считаются по умолчанию анонимными nfsnobody, отключение этой опции не безопасно, потому что любой root пользователь сможет получить доступ на запись ко всем файлам)
`all_squash` превращать все запросы в анонимные
`subtree_check` проверять не пытается ли пользователь выйти за пределы экспортированной папки
`no_subtree_check` отключить проверку обращения к экспортированной папке, улучшает производительность, но снижает безопасность, можно использовать когда экспортируется раздел диска
`anonuid и anongid` указывает uid и gid для анонимного пользователя

Работает стандартная система доступа UNIX, поэтому, если нужно чтобы пользователь подключивший директорию мог получить доступ к папке, то на клиентской стороне должен существовать пользователь с таким же UID (именем и ID), а на сервере для расшаренной директории должна принадлежать такому же пользователю или группе в которой он состоит (GID). Или дать полный доступ для всех пользователей (chmod 777 nfs-folder), тогда все созданные файлы будут от имени: nobody nogroup

`/nfs-folder 192.168.3.0/24(rw,sync,all_squash,anonuid=1020,anongid=1020)` любой пользователь в сети сможет получить полный доступ ко всем файлам расшаренной директории, предварительно нужно создать пользователя с UID 1020 и указать, что бы все подключения считать запросами от анонимного пользователя, а анонимному пользователю присвоить UID 1020. Если у пользователя с id 1020 есть доступ к расшаренной директории на сервере, то при подключении директории на клиентской стороне он будет у пользователя с любым UID. При создании файлов и директорий под другим пользователем, будет указан владелец с номером UID и GID 1020.
`usermod -u 1020 nfs-user` изменить UID
`groupmod -g 1020 nfs-user` изменить GID
`id nfs-user` проверяем ID

client

`apt install nfs-common` установить на клиентском компьютере, что бы работать с файловой системой
`mkdir /mnt/nfs-folder && mount 192.168.3.104:/nfs-folder/ /mnt/nfs-folder` подключить шару и првоерить df -h (192.168.3.104:/nfs-folder 48G 20G 27G 43% /mnt/nfs-folder)
`umount /mnt/nfs-folder` отключить
`showmount -e 192.168.3.104` отобразить список всех доступных ресурсов

ftp

`apt install vsftpd` установка vsFTPd Server (Very Secure File Transfer Protocol Daemon)
`systemctl status vsftpd`
`ufw allow 20:21/tcp` открыть порты в Firewall
`ufw allow 30000:31000/tcp`
`cp /etc/vsftpd.conf /etc/vsftpd.conf.bak` резервная копия настроек

`nano /etc/vsftpd.conf`

```
listen=YES
listen_ipv6=NO
anonymous_enable=NO` отключить ананимный вход
local_enable=YES` разрешить использовать имена локальных пользователей сервера для входа
write_enable=YES` разрешить для авторизованных пользователей управлять файловой системой (по умолчанию возможно только скачивание). При подключен
chroot_local_user=NO` когда установлено в YES, ограничивает пользователей их домашними каталогами для предотвращения доступа к остальной части фа
chroot_list_enable=YES` задать список пользователей, которые будут или не будут ограничены своими домашними каталогами в зависимости от chroot_lo
chroot_list_file=/etc/vsftpd.chroot_list` путь к файлу, который содержит список пользователей для chroot-ограничения
user_sub_token=$USER
local_root=/home/$USER/ftp` указать для всех пользователей по умолчанию домашний каталог при подключении (используется вместе с плейсхолдером)
userlist_enable=YES` включает использование списка пользователей, которые могут (или не могут) входить на сервер
userlist_file=/etc/vsftpd.user_list` указывает путь к файлу со списком пользователей, которые разрешены или запрещены
userlist_deny=NO` ппределяет поведение списка пользователей, когда установлено в NO, только пользователи из списка userlist_file могут входить на
connect_from_port_20` YES - использовать 20 порт для передачи данных вместо случайного (нужно для нормальной работы firewall)
xferlog_enable = YES` записывать в лог файл все транзакции
```

```
systemctl restart vsftpd
```

`ss -ln | grep -w "21"` проверить, что 21 порт слушает (LISTEN)

`ss -tn | grep -w "21"` проверить установленные соединения на 21 порту (ESTAB)

`cat /var/log/vsftpd.log` лог работы (CONNECT/LOGIN/UPLOAD/DOWNLOAD/RENAME/DELETE)

ftp client

`ftp 192.168.3.104` доступен только без использования SSL

`230 Login successful` вход в систему успешен

`ls` отобразить все файлы в текущей директории на удаленном компьютере

`get test.json` скачать файл на локальный компьютер

`put out.txt` загрузить файл на удаленный сервер

`bye` закрыть соединение

ftps

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/vsftpd.key -out /etc/ssl/certs/vsftpd.crt
```
сгенерировать самозаверяющий SSL-сертификат и закрытый ключ

```
nano /etc/vsftpd.conf
```

```
ssl_enable=YES
rsa_cert_file=/etc/ssl/certs/vsftpd.crt
rsa_private_key_file=/etc/ssl/private/vsftpd.key
allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
require_ssl_reuse=NO
ssl_ciphers=HIGH
```

`useradd -m -s /bin/bash ftpuser` пользователь с домашним именным каталогом и оболочкой bash

`passwd ftpuser` задать пароль пользователю

`echo "ftpuser" | tee -a /etc/vsftpd.chroot_list` добавить пользователя в список ограниченных домашним каталогом

`echo "ftpuser" | tee -a /etc/vsftpd.user_list` добавить пользователя в список разрешенных для подключения

`mkdir /home/ftpuser/ftp` создать домашнию директорию

`chown ftpuser:ftpuser ftp` назначить владельца и группу для директории

`chmod 555 ftp` ограничить доступ только на чтение и выполнение

`mkdir /home/ftpuser/ftp/upload` создать директорию для загруки (с возможностью записи)

`chown ftpuser:ftpuser upload`

`chmod 775 upload` полный доступ для владельца и группы

rsync

```
rsync -options SRC DST
```

`-a` режим архивирования, когда сохраняются все исходные атрибуты оригинальных файлов (дата изменения и создания)
`-b` создание резервной копии
`-c` проверка контрольных сумм для файлов
`-e` использовать другой транспорт (например ssh)
`-h` выводит цифры в формате, нормальном для чтения
`-l` копировать символьные ссылки
`-L` копировать содержимое ссылок
`-p` сохранять права для файлов
`-q` минимум информации
`-u` не перезаписывать более новые файлы
`-v` выводить подробную информацию о процессе копирования
`-w` выполнить полное копирование без синхронизации
`-z` сжимать файлы перед передачей
`--delete` удалять остальные файлы у получателя, которых нет в источнике отправителя
`--progress` выводить прогресс передачи файла (в %)
`--stat` показать статистику передачи

`rsync -zvh /home/lifailon /backup` копирование и синхронизация только файлов (skipping directory) указанной директории (в пределах одной локальной машины, например на внешний носитель). При редактировании файлов в исходной папке и повторном копирование заменит все содержимое (без синхронизации).

```
nano /etc/rsyncd.conf
```

```
 pid file = /var/run/rsyncd.pid
 lock file = /var/run/rsync.lock
 log file = /var/log/rsync.log
 [share]
 path = /backup/
 hosts allow = 192.168.3.103
 hosts deny = *
 list = true
 uid = lifailon
 gid = lifailon
 read only = false
```

`chown kup:kup backup` сменить владельца и группу директории
`chmod ug+rwx backup` выдать им полные права (для удаленного пользователя)
`systemctl start rsync` запустить сервер

Синхронизация на удаленной машине (192.168.3.103) авторизованным под пользователем, указанным в конфигурации:

`rsync -avzh /backup/ kup@192.168.3.103:/tmp/backup/` скопировать содержимое локальной папки backup (включая директории) на удаленный сервер. По умолчанию Rsync использует транспорт SSH (шифрованный) с запросом пароля (если не используется ключ).
`rsync -avzhe "ssh -p 2121" /tmp/backup/ kup@192.168.21.121:/tmp/backup/` если используется нестандартный порт
`rsync -avzh /tmp/backup/ rsync://192.168.21.121:/share` явно задать использование транспорта Rsync
При редактировании исходных файлов, заменит содержимое только тех файлов, которые были изменены (sending incremental file list ./ test3)
`rsync -avzh kup@192.168.3.103:/tmp/backup /tmp/backup/` скопировать данные с удаленного сервера на локальный компьютер
`ssh-keygen -t rsa` в случае доступа к серверу по SSH необходимо будет создать ключ и загрузить его на сервер, чтобы аутентификация проходила без запроса пароля
`ssh-copy-id -i /home/sk/.ssh/id_rsa.pub kup@192.168.21.10` передать ключ на сервер с которым будет происходить синхронизация
`00 03 * * * rsync -avzhe "ssh -p 2121" /tmp/backup/ kup@192.168.21.121:/tmp/backup/` добавить в планировщик, синхронизация каталогов будет выполняться каждый день в 3 часа ночи

apache

```
apt install apache2
cat /etc/apache2/apache2.conf
cat /etc/apache2/ports.conf
port=8443
cat /etc/apache2/ports.conf | sed -r "s/^Listen.+/Listen $port/" > /etc/apache2/ports.conf
systemctl restart apache2
systemctl status apache2
ss -lpn | grep apache
echo "<H1>$(hostname)</H1>" > /var/www/html/index.html
```

api server

```
mkdir /var/www/api && touch /var/www/api/api.sh && chmod +x /var/www/api/api.sh
curl -s "https://raw.githubusercontent.com/Lifailon/bash-api-server/rsa/www/api/api.sh" > /var/www/api/api.sh
```
установить пример с шаблоном сервера api

```
nano /var/www/api/api.sh
```

```
#!/bin/bash
if [ "$REQUEST_METHOD" == "GET" ]
    then
    echo "Content-type: application/json"
    echo
    echo '{"result": "ok"}'
else
    echo "Content-type: text/plain"
    echo
    echo "Request method not supported"
fi
```

```
a2enmod auth_basic
```
активировать модуль базовой HTTP аутентификации

```
htpasswd -b -c /etc/apache2/.htpasswd rest api
```
настроить htpasswd для хранения пользовательских данных (создать пользователя rest с паролем api)

```
nano /etc/apache2/sites-available/api.conf
```
создать VirtualHost для обработки запросов

```
<VirtualHost *:8443>
    DocumentRoot /var/www/html
    # Связать endpoint (включая все дочернии в пути) с исполняемым файлом
    ScriptAlias /api /var/www/api/api.sh
    # Все опции, вложенные внутрь секции Directory, применяются к указанной директории
    <Directory "/var/www/api">
        # Разрешить выполнение CGI-скриптов
        Options +ExecCGI
        # Обрабатывать все файлы с расширение sh как CGI-скрипт
        AddHandler cgi-script .sh
        AllowOverride None
        Require all granted
    </Directory>
    # Добавить авторизацию для endpoint
    <Location "/api">
        AuthType Basic
        AuthName "Restricted Area"
        AuthUserFile /etc/apache2/.htpasswd
        Require valid-user
        SetHandler cgi-script
        Options +ExecCGI
    </Location>
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

`a2enmod cgi` активировать модуль (en mod) Common Gateway Interface (CGI)
`a2ensite api.conf` активировать VirtualHost (en site)
`systemctl restart apache2`
`tail -f /var/log/apache2/error.log`
`tail -f /var/log/apache2/access.log`

```
curl -s -X GET http://127.0.0.1:8443/api -u rest:api | jq .result
curl -s -X GET http://127.0.0.1:8443/api/info -u rest:api -H "Content-Type: application/json" | jq .content[]
```

`REQUEST_METHOD` метод HTTP-запроса (GET, POST, HEAD и т.д.)
`REQUEST_URI` оригинальный URI запроса
`QUERY_STRING` строка запроса URL
`CONTENT_TYPE` тип содержимого запроса в заголовке клиента (например, application/text)
`CONTENT_LENGTH` длина тела запроса в байтах (чаще, для POST-запросов)
`read -n $CONTENT_LENGTH POST_DATA` прочитать содержимое Body из стандартного ввода (stdin).
`HTTP_STATUS` читаем содержимое переданного заголовка (например, "Status: text"), которое определяется заранее и регламентируется в дальнейшем
`HTTP_USER_AGENT` название агента клиента из заголовка (например, curl/8.4.0)
`REMOTE_ADDR` адрес клиента
`REMOTE_PORT` порт клиента
`SERVER_NAME` адрес сервера
`SERVER_PORT` порт сервера
`SCRIPT_NAME` путь и имя CGI-скрипта
`SERVER_SOFTWARE` имя и версия сервера
`SERVER_PROTOCOL` версия протокола HTTP (например, HTTP/1.1)
`HTTPS` если установлено, то запрос был сделан с использованием HTTPS
`AUTH_TYPE` тип аутентификации, если он был предоставлен
`REMOTE_USER` имя пользователя, если была использована аутентификация
`DOCUMENT_ROOT` корневой каталог веб-сервера

status

`apachectl -M | grep status_module` проверить подключенный модуль статистики: status_module (shared)
`nano /etc/apache2/mods-available/status.conf`
`Require ip 192.168.3.0/24` указать для кого доступна статистика

http://127.0.0.1:8443/server-status

`apachectl -t` проверить синтаксис (Syntax OK)

`a2enmod status` активировать модуль (Module status already enabled)

```
systemctl restart apache2
```

`netstat -tulpan | grep apache2` проверить порт

```
curl http://127.0.0.1:8443/server-status?auto
```

webdav

```
nano /etc/apache2/ports.conf
```

```
 Listen 8443
 Listen 2024
```

`mkdir /var/www/webdav && chown www-data:www-data /var/www/webdav` создать каталог к которому будет доступ через WebDAV и предоставить доступ к нему для www-data

```
nano /etc/apache2/sites-available/webdav.conf
```

```
 <VirtualHost *:2024>
     ServerAdmin webmaster@localhost
     DocumentRoot /var/www/webdav
     Alias /webdav /var/www/webdav
     <Directory /var/www/webdav>
         Options Indexes FollowSymLinks
         AllowOverride None
         Require all granted
         Dav On
         AuthType Basic
         AuthName "WebDAV"
         AuthUserFile /etc/apache2/.htpasswd
         Require valid-user
     </Directory>
     ErrorLog ${APACHE_LOG_DIR}/webdav_error.log
     CustomLog ${APACHE_LOG_DIR}/webdav_access.log combined
 </VirtualHost>
```

```
a2enmod dav
a2enmod dav_fs
a2enmod auth_digest
a2enmod authn_core
a2enmod authn_file
a2enmod authz_core
a2enmod authz_user
```

`htpasswd /etc/apache2/.htpasswd admin` создать пользователя admin (ключ -c используется для пересоздания файла)

`a2ensite webdav` активировать конфигурацию сайта

```
systemctl restart apache2
```

haproxy

```
apt install haproxy
systemctl status haproxy
```

```
/etc/default/haproxy
```

```
 ENABLED=1
```

```
/etc/haproxy/haproxy.cfg
```

```
global
log 127.0.0.1 local0 notice
maxconn 10000
nbproc 1
user haproxy
group haproxy
daemon

defaults
log global
maxconn global
timeout client 5s
timeout server 5s
timeout connect 5s

frontend http_front
mode http
bind *:8081
#bind *:443 ssl crt /etc/ssl/domain.ru/cert.pem
option httplog
###mode tcp
###bind *:3389
###option tcplog
use_backend http_back

backend http_back
mode http
balance roundrobin
###mode tcp
###balance leastconn
option httpchk GET / HTTP/1.1\r\nHost:\ localhost
###option tcp-check
###tcp-check connect port 3389
#server term1.domain.ru 192.168.55.30:443 ssl verify none weight 100 check inter 5s fall 5 rise 3
#server term2.domain.ru 192.168.55.35:443 ssl verify none weight 100 check inter 5s fall 5 rise 3
server pi-hole-01 192.168.3.101:8081 weight 100 check inter 5s fall 5 rise 3
server netbox-01 192.168.3.104:8081 weight 100 check inter 5s fall 5 rise 3

listen stats
bind *:8082
#bind *:8080 ssl crt /etc/ssl/domain.ru/cert.pem
mode http
stats enable
stats uri /
stats auth admin:password
stats show-legends
stats show-node
stats refresh 5s
```

`haproxy -f /etc/haproxy/haproxy.cfg -c` проверить синтаксис (Configuration file is valid)

`systemctl restart haproxy` применить настройки (перечитать конфигурацию)

`ss -lpn | grep 8081`

`curl http://192.168.3.102:8081` проверка http-трафика

`http://192.168.3.102:8082` статистика

`cat /var/log/haproxy.log`

`journalctl -eu haproxy`

`systemctl stop apache2` отключить на 101

- options:

`maxconn` максимальное количество одновременных соединений

`nbproc` количество процессов HAProxy

`option httplog` включает журналирование HTTP-трафика, полезно для отладки и мониторинга прохождения трафика через HAProxy и дает

возможность просматривать HTTP-трафик в журнале, чтобы отслеживать запросы и ответы
`option httpchk` отправлять HTTP-запросы к серверам в бэкенде, чтобы определить, работают ли они, это позволяет выявлять неработающие сервера и перераспределять запросы на работающие
`option httpchk GET / HTTP/1.1\r\nHost:\ localhost` отправляет GET-запрос на корневой путь (/) используя версию протокола HTTP 1.1, Host:\ localhost - это часть заголовка Host, который также включается в HTTP-запрос и указывает на целевой хост, который проверяется
`option tcp-check` активирует общую функцию TCP-проверок для всего бэкэнда, без необходимости указывать порт явно
`tcp-check connect port 443` HAProxy будет устанавливать соединение с серверами в бэкенде на порту 443 для проверки, что серверы доступны и способны принимать соединения на этом порту

- balance:

`Round Robin (roundrobin)` алгоритм используемый по умолчанию, отправляет запросы на сервера по очереди
`static-rr` похож на roundrobin, но он сохраняет порядок серверов в конфигурации
`Least Connections (leastconn)` выбирает сервер с наименьшим количеством активных соединений, это полезно, если у серверов разная производительность или загруженность, так как запросы будут отправляться на менее загруженные серверы
`source` использует IP-адрес источника (клиента) для привязки к одному и тому же серверу, это означает, что клиент всегда будет направляться к одному и тому же серверу, это полезно для сохранения состояния сеанса
`uri` запросы с одним и тем же URL (до знака вопроса) будут переправляться на один и тот же сервер, это полезно для балансировки запросов к разным частям приложения
`rdp-cookie` используется для балансировки запросов RDP (Remote Desktop Protocol), он анализирует cookie-заголовок RDP для принятия решений о направлении запросов

- server:

`ssl` использование SSL
`verify none` отсутствие проверки сертификата
`weight` распределение запросов по весу, если необходимо на определенный сервер отправлять больше запросов
`inter` изменяет интервал между проверками, по умолчанию две секунды
`fall` устанавливает допустимое количество неудачных проверок, по умолчанию три
`rise` задает, сколько проходных проверок должно быть, прежде чем вернуть ранее отказавший сервер в ротацию, по умолчанию два
`check port 443` указать явную проверку порта для конкретного сервера
`check backup` параметр означает, что сервер будет использоваться только в случае, если все основные серверы становятся недоступными и не будет участвовать в балансировке, пока основные серверы функционируют

keepalive

VRRP (Virtual Router Redundancy Protocol) - сетевой протокол, предназначенный для увеличения доступности маршрутизаторов, выполняющих роль шлюза
VRRP-пакеты - это специальные сообщения, которые узлы (маршрутизаторы/сервера) в VRRP-группе рассылают для сообщения своего состояния
VIP (Virtual IP) - виртуальный IP адрес, который может автоматически переключаться между серверами в случае сбоя (frondend для haproxy/dns-rr), у кого в данный момент в сетевом интерфейсе прописан VIP, тот сервер и работает
Master - сервер, на котором в данный момент активен VIP (отправляет VRRP-пакеты на backup nodes)
Backup - сервера на которые переключится VIP, в случае сбоя мастера (следим за мастером)
VRID (virtual_router_id) - сервера, объединенные общим виртуальным IP (VIP) образуют виртуальный роутер, уникальный идентификатор которого, принимает значения от 1 до 255. Сервер может одновременно состоять в нескольких VRID, при этом для каждой VRID должны использоваться уникальные виртуальные IP адреса.
Master сервер с заданным интервалом отправляет VRRP пакеты на зарезервированный адрес multicast (многоадресной) рассылки или unicast на указанные ip-адреса, а все backup/slave сервера слушают этот адрес. Если Slave сервер не получает пакеты, он начинает процедуру выбора Master в соответствии с приоритетом, и если он переходит в состояние Master, то у него активирует VIP (поднимается виртуальный интерфейс) и отравляет gratuitous ARP.
Gratuitous ARP - это вид ARP ответа, который обновляет MAC таблицу на подключенных коммутаторах, чтобы проинформировать о смене владельца виртуального IP-адреса и MAC-адреса для перенаправления трафика. При настройке VRRP, в качестве адреса для виртуального IP не используется реальный адрес сервера, так как, в случае сбоя, его адрес переместится на соседний, и при восстановлении, он окажется изолированным от сети, и чтобы вернуть свой адрес, нужно отправить в сеть VRRP пакет, но не будет IP адреса, с которого это возможно сделать.

```
nano /etc/keepalived/keepalived.conf
```

```
global_defs {
    enable_script_security
}

vrrp_script nginx_check {
    script "/usr/bin/curl http://127.0.0.1"
    interval 5
    user nginx
}

vrrp_instance web {
    state MASTER # на втором сервере BACKUP
    interface ens33
    virtual_router_id 110
    priority 255 # на втором сервере 100
    advert_int 2
    notify /etc/keepalived/notify-web.sh root
    virtual_ipaddress {
        192.168.3.110
    }
    track_interface {
        ens333
    }
    track_script {
        nginx_check
    }
}
```

`state <MASTER|BACKUP>` начальное состояние при запуске, в режиме nopreempt единственное допустимое значение - BACKUP
`interface` интерфейс, на котором будет работать VRRP и подниматься VIP
`virtual_router_id <0-255>` уникальный идентификатор VRRP экземпляра, должен совпадать на всех серверах одной группы
`priority <0-255>` задает приоритет при выборе MASTER, сервер с большим числом приоритета становится MASTER
`advert_int <число секунд>` определяет, с какой периодичностью мастер должен сообщать остальным о себе, и если по истечению данного периода сервера не получат от мастера широковещательный пакет, то они инициируют выборы нового мастера
`nopreempt` если мастер пропал из сети, и был выбран новый мастер с меньшим приоритетом, то по возвращении старшего мастера, он останется в состоянии BACKUP, пока новый мастер не отвалится
`preempt_delay` что бы мастером был конкретный сервер, то заменить настройку nopreempt на preempt_delay
`notify` скрипт, который будет выполняться при каждом изменении состояния сервера, и имя пользователя, от имени которого данный скрипт будет выполняться (логирование или отправка на почту)
`virtual_ipaddress` виртуальный IP-адрес (VIP), которые будет активирован на сервере в состоянии MASTER, должны совпадать на всех серверах внутри VRRP экземпляра
`track_interface` мониторинг состояния интерфейсов, переводит VRRP экземпляр в состояние FAULT, если один из перечисленных интерфейсов находится в состоянии DOWN
`track_script` мониторинг с использованием скрипта, который должен возвращать 0 если проверка завершилась успешно или 1, если проверка завершилась с ошибкой
`fall <число>` количество раз, которое скрипт вернул не нулевое значение, при котором перейти в состояние FAULT
`rise <число>` количество раз, которое скрипт вернул нулевое значение, при котором выйти из состояния FAULT
`timeout <число>` время ожидания, пока скрипт вернет результат, после которого вернуть ненулевое значение

```
journalctl -u keepalived
cat /var/log/messages | grep -i keepalived
tail /var/run/keepalived.INSTANCE.web.state
```