Посмотрите на команду whoami, которая проверяет имя пользователя:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ whoami  geekpress |

А вот как можно запустить команду bash от имени другого пользователя, с sudo -u username:

|  |  |
| --- | --- |
| 1  2  3 | geekpress@proglib:~$ sudo -u test touch def && ls -l  total 0  -rw-r--r-- 1 test test 0 Jan 11 20:05 def |

Когда не указан флаг -u, команда выполняется от имени суперпользователя root без ограничений:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ sudo touch ghi && ls -l  total 662936  -rw-r--r-- 1 root root 0 Feb 27 14:35 ghi  drwxr-xr-x 4 geekpress geekpress 4096 Feb 5 23:54 go |

Хотите стать другим пользователем? С su это реально. Чтобы вернуться в свою учетную запись, используйте exit:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | geekpress@proglib:~$ su luser  Password:  $ whoami  luser  $ exit    geekpress@proglib:~$ whoami  geekpress |

Суперпользователь – единственный пользователь, который может устанавливать программы, создавать новых юзеров и все в таком духе. Иногда можно забыть об этом и получить ошибку:

|  |  |
| --- | --- |
| 1  2  3 | geekpress@proglib:~$ apt install golang  E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission denied)  E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root? |

Введите команду заново, используя sudo:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ sudo apt install golang  Reading package lists... |

Или используйте !! для возврата к предыдущей команде:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | geekpress@proglib:~$ apt install golang  E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission denied)  E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root?    geekpress@proglib:~$ sudo !!  sudo apt install golang  Reading package lists... |

По умолчанию после использования sudo система не запрашивает пароль в течении 15 минут. А вот далее для sudo нужно заново вводить пароль суперпользователя.

# Разбираемся с правами доступа

Файлы доступны для чтения (r), записи (w) и исполнения (x) пользователям или группам. Просматривайте права доступа к файлам с помощью ls -l:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ ls -lh  total 648M  -rw-r--r-- 1 root root 0 Feb 27 14:35 ghi  drwxr-xr-x 4 geekpress geekpress 4.0K Feb 5 23:54 go |

**Права представлены первыми десятью символами.**

Первый символ представляет тип файла: d – директория, l – ссылка, - – файл. Дальше следуют три группы из трёх символов, которые отражают разрешения пользователя, владельца, группы и остальных пользователей.

r означает, что группа или пользователь имеют права на чтение файла. w – это права на изменение, а x – на выполнение. Пока что ничего сложного, правда?

Эти разрешения также представляются трехзначным числом, где x увеличивает значение на 1, w, если включен, – на 2 и r – на 4. Поэтому в бинарном представлении, директории выше имеют права доступа 644 и 755. Например r-x -> 101 -> 5.

Следующие строки – имя и группа владельца. За ними следуют размер, дата последнего изменения и название файла. Флаг -h означает «human-readable» и печатает 4.0K вместо 4096 байт.

chmod изменяет разрешения файла, устанавливая биты доступа:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ chmod 777 public && chmod 000 topsecret && ls -h  total 750M  -rwxrwxrwx 1 geekpress geekpress 0 Feb 27 16:14 public  ---------- 1 geekpress geekpress 0 Feb 27 16:14 topsecret |

Или добавлением и удалением разрешений флагами + и -:

|  |  |
| --- | --- |
| 1  2  3  4  5 | geekpress@proglib:~$ chmod +rwx topsecret && chmod -w public && ls -lh  chmod: public: new permissions are r-xrwxrwx, not r-xr-xr-x  total 750M  -r-xrwxrwx 1 geekpress geekpress 0 Feb 27 16:14 public  -rwxr-xr-x 1 geekpress geekpress 0 Feb 27 16:14 topsecret |

chown изменяет владельца:

|  |  |
| --- | --- |
| 1 | geekpress@proglib:~$ sudo chown luser public |

chgrp меняет группу владельцев:

|  |  |
| --- | --- |
| 1  2  3  4  5 | geekpress@proglib:~$ sudo chgrp luser 1 && ls -lh  total 750M  -rw-r--r-- 1 geekpress luser 0 Feb 27 16:48 1  -rw-r--r-- 1 geekpress geekpress 0 Feb 27 16:48 2  -rw-r--r-- 1 geekpress geekpress 0 Feb 27 16:48 3 |

## Управляем пользователями и группами

Переходим к самому интересному списку команд bash, а именно к тем, которые затрагивают юзеров и группы.

users отображает авторизованных пользователей. Некоторые из них могут быть авторизованы несколько раз, например, при разных сессиях ssh.

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ users  geekpress neo neo neo neo neo trinity trinity |

Чтобы посмотреть всех пользователей (даже тех, кто не авторизован), проверьте /etc/passwd. Но **не вносите изменения** в файл! Вы можете повредить его и сделать невозможной авторизацию пользователей.

|  |  |
| --- | --- |
| 1  2  3  4  5 | geekpress@proglib:~$ alias au="cut -d: -f1 /etc/passwd \  > | sort | uniq" && au  \_apt  agentsmith  geekpress... |

Добавляйте пользователей командой useradd:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ sudo useradd morpheus && au  \_apt  agentsmith  morpheus... |

Удаляйте их командой userdel:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ sudo userdel agentsmith && au  \_apt  geekpress  morpheus... |

groups показывает группы, в которых состоит текущий пользователь:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ groups  geekpress cdrom floppy sudo audio dip video plugdev netdev bluetooth |

Нужно посмотреть все группы в системе? Для этого есть команда /etc/groups. **Не модифицируйте** файл, если не уверены в том, что делаете.

|  |  |
| --- | --- |
| 1  2  3  4  5 | geekpress@proglib:~$ alias ag=“cut -d: -f1 /etc/group \  > | sort” && ag  adm  avahi  daemon... |

Добавляйте группы с помощью groupadd:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ sudo groupadd matrix && ag  adm  avahi  matrix... |

А удаляйте посредством groupdel:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ sudo groupdel matrix && ag  adm  avahi  daemon... |

## Работаем с текстом

uniq печатает повторяющиеся строки:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ printf "hello\nBash" > a && printf "hello\nagain\nBash" > b  geekpress@proglib:~$ uniq a  hello  Bash |

sort сортирует строки по алфавиту или номеру:

|  |  |
| --- | --- |
| 1  2  3 | geekpress@proglib:~$ sort a  Bash  hello |

diff покажет отличия между двумя файлами:

|  |  |
| --- | --- |
| 1  2  3 | geekpress@proglib:~$ diff a b  1a2  > again |

cmp показывает отличия в байтах:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ cmp a b  a b differ: byte 7, line 2 |

cut используется для деления строки на разделы и подходит для обработки CSV. -dуказывает символ деления, а -f – отрезок для печати:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ printf "192.168.1.1" > z    geekpress@proglib:~$ cut -d'.' z -f2  168 |

sed меняет строки:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ echo "abc" | sed s/abc/xyz/  xyz |

Вообще, sed – чрезвычайно мощная утилита, и ее полное описание не представляется возможным в рамках данной статьи.

Утилита является полной по [Тьюрингу](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3%D0%B0), поэтому может делать все, что доступно в любом другом языке программирования. sed работает с [регулярными выражениями](https://proglib.io/p/dont-fear-regex/), печатает строки по шаблону, редактирует текстовые файлы и многое другое.

Хотите узнать больше о чудо-команде? Не вопрос. Полезные ссылки для изучения sed:

* <https://www.tutorialspoint.com/sed/>
* <http://www.grymoire.com/Unix/Sed.html>
* <https://www.computerhope.com/unix/used.htm>

## Ищем и сопоставляем

grep ищет строки в файлах по заданному шаблону:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | geekpress@proglib:~$ grep -e ".\*go.\*" ./README.md  Some of the tools, `godoc` and `vet` for example, are included in binary Go  `go get`.  The easiest way to install is to run `go get -u golang.org/x/tools/...`. You can  also manually git clone the repository to `$GOPATH/src/golang.org/x/tools`.  ... |

Или по заданному слову:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ grep "geekpress" /etc/passwd  geekpress:x:1000:1000:geekpress,,,:/home/geekpress:/bin/bash |

Используйте расширенные регулярные выражения с помощью флага -E, сопоставляйте несколько строк одновременно (-F) и рекурсивно выполняйте поиск по файлам в каталоге (-r).

awk – это язык сопоставления шаблонов, построенный для чтения и манипулирования файлами данных, таких как CSV.

Как правило, grep хорош для поиска строк и шаблонов, sed – для замены строк в файлах, а awk – для извлечения строк и шаблонов в целях анализа.

В качестве демонстрации способностей awk возьмем файл, содержащий два столбца данных:

|  |  |
| --- | --- |
| 1 | geekpress@proglib:~$ printf "A 10\nB 20\nC 60" > file |

Зациклим строки, добавим число к сумме, увеличим счетчик, найдем среднее:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ awk 'BEGIN {sum=0; count=0; OFS=" "} {sum+=$2; count++} END {print "Average:", sum/count}' file  Average: 30 |

awk, как и sed, является полной по Тьюрингу. Обе команды чрезвычайно полезны в сопоставлении по шаблону и в обработке текста. Для их описания будет мало и книги, поэтому читайте о них больше в отдельных статьях!

## Копируем файлы по SSH

ssh – это сетевой протокол взаимодействия машин под управлением Unix-подобных ОС:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ ssh –p <port> geekpress@192.xxx.xxx.100  Last login: Thu Feb 28 13:33:30 2019 from 192.xxx.xxx.102 |

Заметьте, как поменялось приглашение после авторизации на другой машине:

|  |  |
| --- | --- |
| 1  2  3 | geekpress@office exit  logout  Connection to 192.xxx.xxx.100 closed. |

Создадим новый файл на своей машине:

|  |  |
| --- | --- |
| 1 | geekpress@proglib:~$ echo "blabla" > blabla |

Скопируем файл на удаленный компьютер с помощью scp:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ scp –P <port> blabla geekpress@192.xxx.xxx.100:~  blabla 100% 0 0.0KB/s 00:00 |

Зайдем на удаленную машину:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ ssh –p <port> andrew@192.xxx.xxx.100  Last login: Thu Feb 28 13:45:30 2019 from 192.xxx.xxx.102 |

И увидим наш файл:

|  |  |
| --- | --- |
| 1  2  3  4  5 | geekpress@office:~$ ls  blabla projects pdfs    geekpress@office:~$ cat blabla  blabla |

А как насчет оптимизации процесса? Здесь пригодится rsync – инструмент копирования файлов, который минимизирует объем копируемых данных путем поиска различий между файлами.

Предположим, есть директории a и b, содержащие один и два файла соответственно:

|  |  |
| --- | --- |
| 1  2  3 | geekpress@proglib:~/a$ ls && ls ../b  file0  file0 file1 |

Синхронизируем директории, копируя только отсутствующие файлы:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~/a$ rsync -av ../b/\* .  sending incremental file list... |

Теперь a и b содержат одинаковые файлы:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~/a$ ls  file0 file1 |

rsync работает по ssh:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | geekpress@office:~/dir0$ ls    geekpress@office:~/dir0$ rsync -avz -e "ssh -p <port>" geekpress@192.xxx.xxx.102:~/dir1/\* .  receiving incremental file list  file 0  file 1    sent 44 bytes received 99 bytes 128.88 bytes/sec  total size is 0 speedup is 0.00    geekpress@office:~/dir0$ ls  file 0 file 1 |

## Запускаем длительные процессы

Иногда соединение ssh может прерваться из-за неполадок с сетью или оборудованием. При этом процессы, запущенные отключившимся пользователем, прерываются. Команда nohupпредотвращает прерывания процессов даже после отключения пользователя. Отличная страховка! Вот как ею пользоваться.

Запустим команду yes с nohup:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ nohup yes &  [1] 31232 |

ps покажет процессы, запущенные текущим пользователем:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ ps | sed -n '/yes/p'  31283 pts/0 00:00:07 yes |

Теперь выйдем из сессии, зайдем снова и увидим, что процесс исчез:

|  |  |
| --- | --- |
| 1 | geekpress@proglib:~$ ps | sed -n '/yes/p' |

Но постойте! Процесс виден в выводе команд top и htop:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ top -bn 1 | sed -n '/yes/p'  31578 anatoly 20 0 5840 760 688 D 0.0 0.0 0:00.69 yes |

Завершим процесс командой kill -9 с указанием PID:

|  |  |
| --- | --- |
| 1  2 | geekpress@proglib:~$ kill -9 31578  [1]+ Killed nohup yes |

Проверим видимость в top и увидим, что процесса нет, потому что он был завершен:

|  |  |
| --- | --- |
| 1 | geekpress@proglib:~$ top -bn 1 | sed -n '/yes/p' |

cron предоставляет легкие автоматизацию и планирование.

Можно настроить задачи в текстовом редакторе командой crontab -e. Вставим следующую строку:

|  |  |
| --- | --- |
| 1 | \* \* \* \* \* date >> ~/datefile.txt |

Теперь cron вызывает команду date каждую минуты и записывает вывод в текстовый файл оператором >>:

|  |  |
| --- | --- |
| 1  2  3  4 | geekpress@proglib:~$ head ~/datefile.txt  Thu Feb 28 17:06:01 GMT 2019  Thu Feb 28 17:07:01 GMT 2019  Thu Feb 28 17:08:01 GMT 2019 |

Удалите строку в crontab, чтобы остановить выполнение задачи. cron можно настроить на выполнение задач поминутно в течении каждого часа (0 — 59), ежечасно в течении дня (0-23), ежедневно в течении месяца (1-31), ежемесячно в течении года (1-12) или в указанные дни недели (0-6, Пн-Вс). Это отображается пятью звездочками в начале. Замените звезды нужным числом, чтобы настроить расписание.