БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет прикладной математики и информатики
Кафедра технологий программирования

**Лабораторная работа №2**

По курсу “Системное программирование”

# Основы управления учетными записями, правами доступа и процессами

Методические указания по выполнению лабораторной работы

**Подготовила**: Давидовская М.И.,
Ст. преподаватель кафедры ТП

Минск, 2023 г.

# Содержание

# 1 Цель работы

Ознакомление команды управления пользователями, группами, правами доступа и процессами.

# 2 Задачи работы

1.1. Изучить команды управления пользователями и группами.
1.2. Изучить команды для дискреционной модели назначения прав доступа.
1.3. Изучить команды управления процессами.

# 3 Краткие теоретические сведения

Лабораторная работа выполняется в среде ОС Ubuntu ОС, исполняемой в виртуальной машине под управлением Virtualbox или KVM/QEMU. Доступ к ОС Ubuntu осуществляется с рабочего места, функционирующего в среде ОС Windows или macOS. При выполнении лабораторной работы Вы получаете доступ к учетной записи администратора и создаете учетные записи пользователей для выполнения лабораторной работы.

По умолчанию приглашением в Unix/Linux является символ **'#'** или **'$'**, в ответ на которое Вы можете вводить команды. Если в примере приводится команда и в качества символа приглашения указан символ **'#'**, то это означает, что это пример командной строки для суперпользователя `root`, и команду мы вводим под учетной записью `root`. Если же в примере указан символ **'$'**, то это означает, что команда вводится по учетной записью обычного пользователя. Левее символа командной строки в квадратных скобках указана Ваша учетная запись и имя устройства с ОС Linux, а также текущий каталог.

В сеансе работы с Linux Вашим текущим (домашним) каталогом является каталог: `/home/имя`, где `имя` — Ваше сетевое имя пользователя. К этому каталогу вам предоставлен полный доступ, т.е. вы имеете права чтения, записи и выполнения. Вы не имеете права записи к каталогам, не являющимся подкаталогами вашего домашнего каталога, если их владельцы (или системный администратор) не дали вам соответствующих прав. Не забывайте, что в Unix/Linux символ "прямая косая черта" (прямой слэш) — разделитель имен каталогов наклонен вправо: '/'!

Для получения справки по командам и программам служат инструкции `man` и `info`, а также другие команды. Некоторые разделы справки даны на английском языке. Краткое описание применяемых в лабораторной работе команд и программ на русском языке представлено в Приложение 1, стр. 23.

Для окончания сеанса работы с командной оболочкой Linux используем команды `exit` или `logout`.

# 4 Методические указания

Методические рекомендации включают примеры для изучения, с которыми рекомендуется ознакомиться до выполнения заданий.

## 4.1 Примеры по управлению учетными записями и группами

### 4.1.1 Добавление учётной записи пользователя

1. Войдите в систему под именем `root`. Создайте пользователя с именем `temp`, действие учётной записи которого истекает 31 декабря **текущего** года, изменив год и выполнив следующую команду, например:

```
# useradd -e 2023-12-31 temp
```

2. Создайте пользователя с именем `user`, учётная запись которого становится недействительной через 14 дней после истечения действия пароля, выполнив следующую команду:

```
# useradd -f 14 user
```

### 4.1.2 Установка пароля пользователя

1. Установите пароль `temppass` для учётной записи пользователя `temp`, выполнив команду

```
# passwd temp
```

Установите пароль `userpass` для учётной записи `user`, выполнив команду

```
# passwd user
```

Проверьте создание паролей, войдя в командную оболочку под именами пользователей `temp` и `user`.

2. Войдите в командную оболочку как пользователь `root` и заблокируйте пароль пользователя `temp`, выполнив команду

```
# passwd -l temp
```

Проверьте, что пароль пользователя `temp` заблокирован. Войдите в командную оболочку как пользователь `root` и разблокируйте пароль пользователя `temp`, выполнив команду

```
# passwd -u temp
```

Проверьте, что пароль пользователя `temp` разблокирован.

### 4.1.3 Изменение информации о времени действия пароля пользователя

1. Просмотрите информацию о пользователя `temp`, выполнив команду времени действия пароля

```
# chage -l temp
```

2. Измените дату истечения действия пароля на 14 ноября **текущего** года, изменив год и выполнив следующую команду, например:

```
# chage -E 2023-11-14 temp
```

### 4.1.4 Изменение личной информации о пользователе

1. Задайте следующую личную информацию о пользователе `temp`:

- настоящее имя: `Nick`;
- номер офиса: `12`;
- номер офисного телефона `209-52-73`;
- номер домашнего телефона: `217-52-73`;

выполнив следующую команду:

```
# chfn -f Nick -o 12 -w 209-52-73 -h 217-52-73 temp
```

### 4.1.5 Модификация учётной записи пользователя

1. Измените имя пользователя `temp` на `Nick`, выполнив команду

```
# usermod -l Nick temp
```

2. Измените домашний каталог пользователя `Nick` на `/home/Nick`, выполнив команду

```
# usermod -d /home/Nick Nick
```

### 4.1.6 Добавление группы

1. Создайте группу с именем `agroup`, выполнив команду

```
# groupadd agroup
```

2. Назначьте пользователя `Nick` администратором группы `agroup`

```
# gpasswd -A Nick agroup
```

3. Войдите в систему под именем `Nick` и включить пользователя `user` в группу `agroup`

```
$ gpasswd -a user agroup
```

4. Переименуйте группу agroup на группу `bgroup`

```
# groupmod -n bgroup agroup
```

### 4.1.7 Удаление учетной записи пользователя и группы

1. Удалите группу `bgroup`, выполнив команду

```
# groupdel bgroup
```

2. Удалите учетные записи пользователей `Nick` и `user`, выполнив команды

```
# userdel -r Nick
```

```
# userdel -r user
```

### 4.1.8 Возможности библиотеки PAM

1. Изучите статью https://losst.ru/nastrojka-pam-v-linux.

2. Выведите список файлов из каталога `/etc/pam.d/.`

3. Просмотрите файл `/etc/pam.d/login`.

## 4.2 Примеры по управлению правами доступа

### 4.2.1 Управление разрешениями на доступ к файлу

1. Откроем приложение *Terminal*. Входим в систему под именем `root` с помощью команды:
   $ sudo -i
   вводим пароль для пользователя `studentLastNameN`.
2. Создаём в каталоге `/tmp` каталог `lab2LastNameN`, где `LastName` — фамилия студента, `N` — номер группы, переходим в него и создаем в текущем каталоге файл `message.txt`.
   ```
   # mkdir /tmp/lab2LastNameN
   # cd /tmp/lab2LastNameN
   # touch message.txt
   ```
3. Просматриваем разрешения на доступ к файлам текущего каталога:
   ```
   # ls -l
   ```
4. Устанавливает пользователя с именем `studentLastNameN` владельцем файла `message.txt`:
   ```
   # chown studentLastNameN message.txt
   ```
5. Скопировать каталог Documents из домашнего пользователя `studentLastNameN` в каталог `/tmp/lab2LastNameN`. Команда копирования будет такой:
   ```
   # cp -R /home/studentLastNameN/Documents
   /tmp/lab2LastNameN
   ```
6. Проверяем содержимое каталога `/tmp/lab2LastNameN`, чтобы убедиться, что каталог `Documents` был скопирован:
   ```
   # ls -l
   ```
7. Устанавливаем пользователя с именем `studentLastNameN` владельцем каталога `Documents` и всех файлов, находящихся в этом каталоге:
   ```
   # chown -R student Documents
   ```
8. Создаём группу с именем `students`
   ```
   # groupadd students
   ```
9. Устанавливаем группу с именем `students` группой файла `message.txt`:
   ```
   # chgrp students message.txt
   ```
10. Устанавливаем группу с именем `students` группой каталога `Documents` и всех файлов этого каталога
    ```
    # chgrp -R students Documents
    ```
11. Просматриваем разрешения на доступ к файлам текущего каталога:
    ```
    # ls -l
    ```
12. Для того чтобы разрешить владельцу файла `message.txt` читать и писать в этот файл, а членам группы файла и всем остальным пользователям разрешить только читать этот файл (`-rw-r—r--`), нужно выполнить следующую команду:
    ```
    # chmod 644 message.txt
    # ls -l
    ```

13. Для того чтобы сбросить все разрешения на доступ к файлу `message.txt` для владельца файла и установить для него только разрешение на запись в файл, нужно выполнить следующую команду:
```
# chmod u=w message.txt
# ls -l
```
14. Создадим файл readme1.txt в каталоге Documents и проверим права доступа:
```
# touch Documents/readme1.txt
# ls -la Documents
```
Обратим внимание, что владельцем файла `readme1.txt` является пользователь `root`, а группой — так же `root`.
15. Добавляем пользователя `studentLastNameN` в группу `students`:
```
# usermod -a -G students studentLastNameN
```
16. Устанавливаем специальное разрешение `setgid` на каталог `documents`:
```
# chmod g+s documents
```
17. Создадим файл `readme2.txt` в каталоге `Documents` и проверим права доступа:
```
# touch Documents/readme2.txt
# ls -la Documents
```
Обратим внимание, что владельцем файла `readme.txt` является пользователь `root`, а группой — так же `students`.
18. Просматриваем текущую маску разрешений на доступ к файлам в числовом и символьном формате:
```
# umask
# umask -S
```
По умолчанию для создаваемых файлов и каталогов установлены следующие системные разрешения на доступ: `666` — для файлов, `777` — для каталогов, а системная маска разрешений на доступ к файлам команды `umask` установлена на `0022`.
19. Изменим маску разрешений командой:
```
# umask 0200
```
Теперь при создании файлов и каталогов будут использоваться следующие маски разрешений на доступ: `466` — для файлов, `577` — для каталогов, что эквивалентно следующим символьным обозначениям:
`r–rw-rw-` для файлов;
`r-xrwxrwx` для каталогов.
20. Создаем новый файл `testmask.txt` в каталоге `Documents` и проверяем его права доступа:
```
# touch Documents/testmask.txt
# ls -la documents
```
21. Восстановим маску разрешений по умолчанию:
```
# umask 0022
```

### 4.2.2 Проверить поддерживает ли файловая система ACL

1. Откроем приложение *Terminal*. Входим в систему под именем `root` с помощью команды:
```
$ sudo -i
```

вводим пароль для пользователя studentLastNameN.

2. Выполните команду, приведённую ниже для проверки поддержки ACL
   ```
   # tune2fs -l /dev/sda1
   ```
   Если в результате выполнения команды получите вывод, подобный выводу ниже:
   ```
   tune2fs 1.45.5 (07-Jan-2020)
   Filesystem volume name: <none>
   Last mounted on: /
   Filesystem UUID: 56882b64-05cd-4f11-98d5-64dd90290650
   Filesystem magic number: 0xEF53
   Filesystem revision #: 1 (dynamic)
   Filesystem features: has_journal ext_attr resize_inode dir_index filetype needs_recovery extent 64bit flex_bg sparse_super large_file huge_file dir_nlink extra_isize metadata_csum
   Filesystem flags: signed_directory_hash
   Default mount options: user_xattr acl
   ```
   и для монтирования диска будет указан параметр acl, то файловая система смонтирована с поддержкой списков доступа.

### 4.2.3 Предоставление прав доступа ACL для пользователя

1. Откроем приложение *Terminal*. Входим в систему под именем root с помощью команды:
   ```
   $ sudo -i
   ```
   вводим пароль для пользователя studentLastNameN.
2. Перейти в каталог /tmp/lab2LastNameN. Данный каталог должен быть текущим при выполнении команд:
   ```
   # cd /tmp/lab2LastNameN
   ```
3. Создайте пользователя student1 и группу abcgroup.
   ```
   # useradd -m -U -s /bin/bash student1
   # groupadd abcgroup
   ```
4. Проверьте создание пользователя student1 и группы abcgroup:
   ```
   # id student1
   # tail -n2 /etc/group
   ```
5. Как пользователь root создайте каталог test_folder. В каталоге /tmp/lab2LastNameN (данный каталог является текущим):
   ```
   # mkdir test_folder
   ```
6. Предоставьте для пользователя student1 права доступа ACL на каталог test_folder с помощью команды:
   ```
   # setfacl -m u:student1:rwx test_folder
   ```
7. Проверьте, что права доступа назначены корректно, используя команду:
   ```
   # getfacl test_folder
   ```
8. Переключитесь в учетную запись student1 и создайте файл test_acl.txt в каталоге test_folder:
   ```
   # su student1
   $ touch /tmp/lab2LastNameN/test_folder/test_acl.txt
   ```

9. Проверьте содержимое каталога test_folder:
```
$ ls -la /tmp/lab2LastNameN/test_folder
```
Обратите внимание, что владельцем каталога являет пользователь root, группа-владелец root. Из-за применения списков контроля доступа пользователь student1 смог создать файл в каталоге, владельцем которого ни он, ни его группа не является.
10. Выйдите из учетной записи student1 и переключитесь в учетную запись root:
```
$ exit
```
В результате сеанс работы пользователя student1 будет завершен.

### 4.2.4 Предоставление прав доступа ACL для группы пользователей

1. Откроем приложение *Terminal*. Входим в систему под именем root с помощью команды:
```
$ sudo -i
```
вводим пароль для пользователя studentLastNameN.
2. Перейти в каталог /tmp/lab2LastNameN. Данный каталог должен быть текущим при выполнении команд:
```
# cd /tmp/lab2LastNameN
```
3. Добавим пользователей studentLastNameN и student1 в группу abcgroup:
```
# usermod -G abcgroup studentLastNameN
# usermod -G abcgroup student1
```
4. Проверим, что пользователи studentLastNameN и student1 добавлены в группу abcgroup:
```
# id studentLastNameN
# id student1
```
5. Предоставьте для группы пользователей abcgroup права доступа ACL на каталог test_folder с помощью команды:
```
# setfacl -m g:abcgroup:rwx test_folder/
```
6. Проверьте, что права доступа назначены корректно, используя команду:
```
# getfacl test_folder/
```
7. Переключитесь в учетную запись studentLastNameN и попробуйте создать файл group_acl.txt в каталоге test_folder:
```
# exit
$ touch /tmp/lab2LastNameN/test_folder/group_acl.txt
```
В результате выполнения команды получите ошибку, так как первичная группа пользователя studentLastNameN — studentLastNameN.
8. Изменим первичную группы и повторим команду создания файла:
```
$ newgrp abcgroup
$ touch /tmp/lab2LastNameN/test_folder/group_acl.txt
```
В данном случае файл будет создан.
9. Проверьте содержимое каталога test_folder:
```
$ ls -la /tmp/lab2LastNameN/test_folder
```

Обратите внимание, что владельцем каталога являет пользователь `root`, группа-владелец `root`. Из-за применения списков контроля доступа пользователь `studentLastNameN` смог создать файл в каталоге, владельцем которого ни он, ни его группа не является.

Владельцем созданного файла `group_acl.txt` будет пользователь `studentLastNameN`, а группа-владелец — `abcgroup`.

10. Переключите первичную группу пользователя `studentLastNameN` с `abcgroup` на `studentLastNameN`:
```
$ newgrp -
```

### 4.2.5 Отзыв прав доступа ACL

1. Откроем приложение *Terminal*. Входим в систему под именем `root` с помощью команды:
```
$ sudo -i
```
вводим пароль для пользователя `studentLastNameN`.
2. Перейти в каталог `/tmp/lab2LastNameN`. Данный каталог должен быть текущим при выполнении команд:
```
# cd /tmp/lab2LastNameN
```
3. Отзовите права доступа для пользователя student1 и группы abcgroup права доступа ACL на каталог test_folder с помощью команды:
```
# setfacl -x u:student1,g:abcgroup test_folder/
```
4. Проверьте, что права доступа назначены корректно, используя команду:

```
# getfacl test_folder/
```

## 4.3 Примеры команд управления процессами и заданиями

### 4.3.1 Управление процессами

1. Используйте учетную запись, созданную в одной из предыдущих лабораторных работ. Или создайте новую учетную запись studentLastNameN. Войдите в систему на виртуальных терминалах 1 и 2 (`tty3`, `tty4`) под учетной записью studentLastNameN.

2. Переключитесь в терминал `tty3` и запустить процесс, выполняющий следующие команды:

```
$ (while true; do echo -n A >> log; sleep 1; done)
```

3. Заметьте, что сейчас этот терминал занят исполнением запущенного процесса, который исполняется на переднем плане. Этот процесс присоединяет символ "A"к файлу `~/log` через каждую секунду. Чтобы визуально проверить это, переключитесь в виртуальный терминал `tty4` и выполните следующую команду:

```
$ tail -f log
```

Вы должны увидеть последовательность символов, длина которой возрастает.

4. Переключитесь в виртуальный терминал `tty3` и приостановите работающий процесс, нажав клавиши `<Ctrl+z>`. Командная оболочка сообщит, что процесс

остановлен и выдаст вам номер задания [1]. Переключитесь в виртуальный терминал `tty4` и визуально проверьте, что файл `~/log` больше не увеличивается.

5. Переключитесь в виртуальный терминал `tty3` и возобновите работу процесса в фоновом режиме. Используйте команду `jobs`, чтобы проверить, что задание [1] снова работает.

```
$ bg
$ jobs
```

Переключитесь в виртуальный терминал `tty4` и визуально проверьте, что файл `~/log` снова увеличивается.

6. Переключитесь в виртуальный терминал `tty3` и запустите еще два процесса, выполнив следующие команды:

```
$ (while true; do echo -n B >> log; sleep 1; done) &
$ ^B^C
```

Вторая команда просто запускает предыдущую команду, заменяя символ "B"символом "C".

7. Выполните команду `jobs` и проверьте, что все три процесса работают. Переключитесь в виртуальный терминал `tty4` и визуально проверьте, что файл `~/log` снова увеличивается, на этот путем добавления символов "A" "B"и "C" через каждую секунду.

8. В пункте 4 вы приостановили исполнение процесса переднего плана путем нажатия клавиш `<Ctrl+z>`. В действительности эта комбинация нажатых клавиш посылает процессу сигнал. Используйте команду `kill`, чтобы получить список сигналов и соответствующие им имена и номера. Затем выполните команду `kill`, послав сигнал `SIGSTOP` заданию [1], чтобы приостановить его работу. Переключитесь в виртуальный терминал `tty3` и выполните следующие команды:

```
$ kill -l
$ kill -19 %1
```

9. Выполните команду `jobs` и проверьте, что задание [1] остановлено. Переключитесьв виртуальный терминал `tty4` и визуально проверьте, что задание [1] остановлено.

10. Возобновите выполнение задания [1], используя команду `kill`, которая посылает процессу сигнал `SIGCONT` (18). Используйте команду `jobs` и виртуальный терминал `tty4` для проверки того, что все три задания опять работают.

11. Завершите работу всех трех процессов. Если вы не задаете сигнал, который нужно послать процессу, то команда `kill` посылает по умолчанию сигнал `SIGTERM` (15), который вызывает завершение процесса. После посылки сигналов заданиям [2] и [3], используйте команду `jobs`, чтобы проверить завершение работы этих заданий:

```
$ kill %2 %3
$ jobs
```

12. Чтобы завершить работу последнего процесса, выполните команды:

```
$ fg
$ <Ctrl+c>
```

13. Выполните команду `jobs` и проверьте, что больше заданий не выполняется. Переключитесь в виртуальный терминал `tty4` и визуально проверьте, что файл `~/log` не увеличивается. Остановите исполнение команды `tail`, нажав клавиши `<Ctrl+c>`, и завершите сеанс на виртуальном терминале `tty4`.

14. Переключитесь в виртуальный терминал `tty3` и удалите файл `~/log`.

### 4.3.2 Перенаправление ввода-вывода

1. Создайте в текстовом редакторе два файла для дальнейшего использования в лабораторной работе. Файл `packages1.txt` должен содержать следующих восемь строк:

```
amanda galeon metacity mozilla postgresql procinfo rpmfind squid
```

Файл `packages2.txt` должен содержать следующих шесть строк:

```
anaconda openssh gnome-core samba sendmail xscreensaver
```

2. Для просмотра содержимого файлов предназначена команда `cat`. В простейшем случае эта команда принимает ввод из файла или стандартного канала ввода `STDIN` и посылает его в стандартный канал вывода `STDOUT`.

```
$ cat packages1.txt
```

3. Если команда `cat` не имеет аргументов, то ожидается, что она получает данные из стандартного канала ввода `STDIN`. Команда читает данные из канала стандартного ввода до нажатия на клавиатуре клавиш `<Ctrl+d>`:

```
$ cat
Type some sample text, then press return.
<Ctrl+d>
```

4. Большинство команд `Linux` работают как фильтры, т. е. Получают исходные данные из `STDIN`, делают обработку этих данных и выводят результаты в `STDOUT`. Например, команда `tr (translate)`:

```
$ tr ’aeiou’ ’AEIOU’
Type some sample text, then press return.
<Ctrl+d>
```

5. Используя директиву `’>’`, перенаправим стандартный вывод из одного файла в другой файл:

```
$ cat packages1.txt > packages1.catfile
$ cat packages1.catfile
```

Затем сверим исходный и полученный файлы:

```
$ diff packages1.txt packages1.catfile
```

```
$ ls -l packages1*
```

6. Для присоединения существующего файла к другому файлу предназначена директива ’>>’:

```
$ cat packages2.txt >> packages1.catfile
$ cat packages1.catfile
```

7. Если команде cat не передается аргумент и стандартный ввод перенаправлен в файл, то весь ввод с клавиатуры до нажатия клавиш <Ctrl+d> будет перенаправлен в файл.

```
$ cat > typedin.txt
This time, when text is typed at the keyboard, it is redirected to
the file typedin.txt.
<Ctrl+d>
$ ls -l typedin.txt
$ cat typedin.txt
```

8. Повторите предыдущий шаг, подставив вместо команды cat команду tr.

```
$ tr ’aeiou’ ’AEIOU’ > trfile.txt
This time, when text is typed at the keyboard, it is not echoed
back to the screen with the translations made.
Instead, it is redirected to the file trfile.txt.
<Ctrl+d>
$ ls -l trfile.txt
$ cat trfile.txt
```

9. Команда cat принимает в качестве аргумента имя файла или стандартный ввод,перенаправленный из файла. Проверьте это при помощи следующих двух команд:

```
$ cat packages1.txt
$ cat < packages1.txt
```

10. Однако команда tr принимает ввод только из стандартного канала.

```
$ tr ’aeiou’ ’AEIOU’ < packages1.txt
```

11. В следующем примере стандартный ввод и вывод одновременно перенаправляются

```
$ tr ’aeiou’ ’AEIOU’ < packages1.txt > packages1.trfile.txt
$ ls -l packages1.txt packages1.trfile.txt
$ cat packages1.trfile.txt
```

12. Использование команды tee и каналов для перенаправления:

а) перенаправление вывода в файл и просмотр результата:

```
$ ls -la | tee output.txt
```

```
$ cat output.txt
```

б) добавление вывода в существующий файл и просмотр результата:

```
$ pwd | tee -a output.txt
$ cat output.txt
```

в) запись вывода в несколько файлов и просмотр результата:

```
$ date | tee output1.txt output2.txt
$ cat output1.txt output2.txt
```

13. Подсчет количества строк в файле и вывод результата в другой файл:

```
$ wc -l output.txt | tee -i output3.txt
$ cat output.txt
$ cat output3.txt
```

## 4.4 Критерии оценивания

**Оценка 4**

Выполнены упражнения из примеров выше. Файлы протоколов команд и меток времени с недочетами. Лабораторная работа сдана с задержкой в 2 недели.

**Оценка 5-6**

Выполнены упражнения из примеров выше и задание для самостоятельной работы №1. Представлен отчет, файлы протоколов и меток времени в git-репозитории. Отчет, файлы протоколов команд и меток времени могут содержать ошибки. Лабораторная работа сдана с задержкой в 1 неделю.

**Оценка 7-8**

Выполнены задания для самостоятельной работы №1-3 на хорошем уровне (есть недочеты). Представлен отчет, ответы на вопросы, файлы протоколов и меток времени в git-репозитории. Отчет, файлы протоколов команд и меток времени могут содержать незначительные ошибки. Лабораторная работа сдана с задержкой в 1 неделю.

**Оценка 9**

Выполнены задания для самостоятельной работы №1-3 на отличном уровне. Представлен отчет, ответы на вопросы и файлы протоколов и меток времени в git-репозитории. Отчет, файлы протоколов команд и меток времени не содержат ошибок. Лабораторная работа сдана в срок.

## 4.5 Содержание отчета по лабораторной работе

1. Цель работы.
2. Протоколы выполненных действий.
3. Отчет обязательно должен содержать примеры команд, которые не протоколировались командой `script`, проколированные команды — на усмотрение студента, ответы на вопросы и др.

Отчет должен быть опубликован в git-репозитории на `github`. Все результаты лабораторной работы должны быть опубликованы в git-репозитории, ссылка на который доступна в курсе «Системное программирование».

В файле `Readme` проекта на `github` должна быть ссылка на отчёт. Отчет опубликовать во внешнем хранилище или в репозитории в каталоге `/docs`. Если в лабораторной работе необходимо написать программу/ы, то отчёт должен результаты тестов по каждой программе и ответы на контрольные вопросы.

Пример оформления файла `Readme` может быть таким:

```
# Overview

Report on LabRabota1.

# Usage

// Заменить <<link>> и <<folder>> на соответствующие ссылки/названия
To check, please, preview report by <<link>> and script files
in <<folder>>.

# Author

Your name and group number.

# Additional Notes

// СКОПИРОВАТЬ И ВСТАВИТЬ ССЫЛКУ НА СВОЙ РЕПОЗИТОРИЙ,НАПРИМЕР
https://github.com/maryiad/lab3-task1-gr16-david
```

Каждая лабораторная работа содержит тексты задач и контрольные вопросы, ответы на которые проверяются преподавателем при приёме работы у студента.

Выполнение студентом лабораторной работы и сдача её результатов преподавателю происходит следующим образом:

1. Студент выполняет разработку программ.
2. В ходе разработки студент обязан следовать указаниям к данной задаче (в случае их наличия). Исходные тексты программ следует разрабатывать в соответствии с требованиями к оформлению, приведёнными в приложении.
3. Студент выполняет самостоятельную проверку исходного текста каждой разработанной программы и правильности её работы, а также свои знания по теме лабораторной работы.

# 5 Задания для самостоятельной работы

## Задание 1. Управление пользователями и группами

1.1. Изучить справку к командам useradd, groupadd и примеры из раздела Примеры по управлению учетными записями и группами.

1.2. Задание выполнить в ОС Ubuntu, в которой вести протокол командой script с журналом меток времени. **Протокол** назвать по следующему шаблону — `taskXФамилияN`, где `X` — номер выполняемого задания, `Фамилия` — заменить на вашу фамилию латиницей и строчными буквами, `N` — номер группы, например 12 или 13. **Журнал меток** назвать по следующему шаблону — `timelogXФамилияN`, где `X` — номер выполняемого задания, `Фамилия` — заменить на вашу фамилию латиницей и строчными буквами, `N` — номер группы, например 12 или 13.

1.3. Создать группу `studentN`.

1.4. Создать пользователя с именем `studentLastNameN` с помощью утилиты `useradd`, где `LastName` — фамилия студента, `N` — номер группы, действие учётной записи которого истекает 1 ноября **текущего** года и учётная запись становится недействительной через 3 дня, после истечения действия пароля.

1.5. Создать пользователя с именем `visitorN`, где `N` — номер группы, с помощью утилиты `adduser`. В чем отличие утилиты `adduser` от утилиты `useradd`?

1.6. Добавить пользователей `studentLastNameN` и `visitorN` в группу `studentN`.

1.7. Продемонстрировать следующие операции:

a) Изменение пароля для любого пользователя с помощью утилиты `passwd`.

b) Изменить для пользователя `visitorN` срок действия пароля на 24 декабря **текущего** года.

c) Изменить для пользователя `visitorN` личную информацию, указав имя, номер офиса, номер рабочего телефона, номер домашнего телефона. Какая утилита используется для этого?

d) Изменить имя пользователя с `visitorN` на guest`N`.

e) Назначить администратором группы `studentN` пользователя `studentLastNameN`.

f) Просмотрите файл /etc/shadow (с правами root). У всех ли пользователей содержимое второго поля выглядит приблизительно одинаково? Какую структуру имеет зашифрованный пароль в /etc/shadow? Какой алгоритм хеширования пароля применяется в Ubuntu и по какому признаку это определяется? Если в поле пароля указаны символы **!** или *****, то что означает каждый из них?

g) Создайте учётную запись для пользователя test`LastNameN`, где `N` — номер группы, для которого запрещен вход в сеанс, имеющего

домашний каталог /home/nouser и являющегося членом групп user и mail. Пользователь должен иметь UID равный 2000.

h) Изучите статью https://losst.ru/nastrojka-pam-v-linux и учебный материал Возможности библиотеки PAM. Запретите авторизацию от имени пользователя `studentLastNameN` на компьютере с помощью PAM и отмените блокировку доступа.

1.8. Завершите протоколирование команд с помощью `script`, т. е. введите команду `exit`.

1.9. Ответьте на вопросы к заданию:

a) Просмотрите файл /etc/shadow (с правами root). У всех ли пользователей содержимое второго поля выглядит приблизительно одинаково? Какую структуру имеет зашифрованный пароль в /etc/shadow? Какой алгоритм хеширования пароля применяется в Ubuntu и по какому признаку это определяется? Если в поле пароля указаны символы **`!`** или **`*`**, то что означает каждый из них?

b) Как изменить имя и `GID` группы?

c) Какие задачи решает библиотека PAM?

d) Определите, когда последний раз была загружена система.

e) Как определить, кто входил в сеанс за последние 2 недели?

f) Приведите пример команды, которая выводит журнал (логи) входа пользователей в ОС Linux.

1.10. Скопируйте сохранённые протоколы работы и письменный отчет, добавьте в git-репозиторий.

1.11. Опубликуйте изменения из локального репозитория, т. е. результаты выполнения задания 1, в удаленный.

## Задание 2. Управление правами доступа

2.1. Изучите примеры из раздела Примеры по управлению правами доступаи ознакомьтесь со статьями:
http://ubuntulinux.ru/config/recipe/izmenit-prava-na-fajly-ili-papki-v-linux/
http://desktoplinux.ru/unix_guide/prava_dostupa_k_failam_i_papkam_v_linux
https://help.ubuntu.ru/wiki/стандартные_права_unix

2.2. Задание выполнить в ОС Ubuntu, в которой вести протокол командой script с журналом меток времени. **Протокол** назвать по следующему шаблону — `taskXФамилияN`, где `X` — номер выполняемого задания, `Фамилия` — заменить на вашу фамилию латиницей и строчными буквами, `N` — номер группы, например 12 или 13. **Журнал меток** назвать по следующему шаблону — `timelogXФамилияN`, где `X` — номер выполняемого задания, `Фамилия` — заменить на вашу фамилию латиницей и строчными буквами, `N` — номер группы, например 12 или 13.

2.3. **Текущим каталогом** является каталог **`/tmp/lab2LastNameN`**. Продемонстрировать в данном каталоге выполнение следующих операций:

a) Изменить пользователя-владельца для любого из файлов.
b) Изменить группу-владельца для любого из файлов.
c) Создать в текущем каталоге файл `letter.txt`, добавив в него список белорусских университетов. Установить для владельца и группы-владельца файла `letter.txt` разрешение только на чтение этого файла, а остальным запретить доступ к этому файлу.

2.4. Операции с маской разрешений:

a) Вывести текущую маску разрешений на доступ к файлу в числовом и символьном формате.
b) Установить следующие текущие маски доступа для файлов и каталогов:

`rw-rw----` для файлов;

`rwxrwx---` для каталогов.

c) Создайте в текущем каталоге файл `getmask.txt` и каталог `somedir` и проверьте сведения о них и права доступа.
d) Восстановите текущую маску разрешений.

2.5. Создайте в текущем каталоге любые четыре файла, установите в числовом формате на каждый из них в отдельности следующие права:

a) для себя — **все права**, для группы и остальных — **никаких**;
b) для себя — **чтение и запись**, для группы — **чтение**, для остальных — **все**;
c) для себя — **исполнение и запись**, для группы — **никаких**, для остальных — **чтение**;
d) для себя — **запись**, для группы — **все**, для остальных — **только запись**.

2.6. Создайте еще четыре файла и выполните задание предыдущего пункта, указав права доступа в символьном формате.

2.7. Авторизуйтесь под суперпользователем `root`. Создайте в текущем каталоге каталог `example-acl` и в нем файлы `file1.txt` и `file2.txt`. В файл `file1.txt` добавить вывод команды

```
ls -la /etc
```

а в файл `file2.txt` перенаправить вывод команды `ls` для каталога `/usr/bin`.

a) Назначьте для пользователя `studentLastNameN`, права на чтение и изменение с помощью списков контроля доступа ACL на файл `file1.txt`. Продемонстрируйте, что пользователь `studentLastNameN` может редактировать файл.
b) Скопируйте на файл `file2.txt` права согласно ACL для `file1.txt`, используя команды `getfacl`, `setfacl` и перенаправление.
c) Скопируйте `file2.txt` в каталоге `example-acl` в файл `file3.txt` без потери прав согласно ACL.
d) Продемонстрируйте применение списков контроля доступа для своей группы-владельца на каталог `example-acl`, назначив права `rwx`, авторизовавшись со своей учетной записью и создать в каталоге `example-acl` файл с любым именем.

2.8. Завершите протоколирование команд с помощью `script`, т. е. введите команду `exit`.

2.9. Ответьте на вопросы:

a) Сравните права доступа к директориям `/bin` и `/tmp`. Какие операции сможет совершать в них простой пользователь?
b) Какой командой необходимо задать права на текстовый файл таким образом, чтобы он мог просматриваться только владельцем и никем не мог редактироваться?
c) Что смогут делать другие пользователями с файлами в домашней директории пользователя, если он задаст всем остальным пользователям право на запись в директорию, но удалит право исполнения на неё?
d) Какие права установлены на файлы в Вашем домашнем каталоге? Сможет ли другой пользователь его просматривать? Изменять?
e) Какое значение имеет право на запись для каталога?
f) Какое значение имеет право на исполнение для каталога?
g) Как добавить программе флаг исполнения?
h) Что такое бит `suid`? Что он делает?
i) Зачем нужны `uid` и `gid`?
j) Почему `uid` пользователя задаётся 1000 или более?
k) С помощью какой команды можно найти все исполняемые файлы с установленным `suid-битом`?
l) Приведите системные разрешения для файлов и каталогов. Почему они не совпадают?
m) Какая маска разрешений задана по умолчанию? В каком файле задается значение системной маски разрешений? Можно ли его изменить?
n) Будут ли утеряны права ACL при перемещении файлов с помощью команды `mv`?
o) С каким ключом необходимо выполнять команду `cp`, чтобы не потерялись права `ACL`?

2.10. Скопируйте сохранённые протоколы работы и добавьте в git-репозиторий.

2.11. Опубликуйте изменения из локального репозитория, т. е. результаты выполнения задания 2, в удаленный.

## Задание 3. Базовые команды для управления процессами

3.1. Задание выполнить в ОС Ubuntu. При выполнении пункта 3.2 задания вести протокол командой script с журналом меток времени. **Протокол** назвать по следующему шаблону — `taskXФамилияN`, где `X` — номер выполняемого задания, `Фамилия` — заменить на вашу фамилию латиницей и строчными буквами, `N` — номер группы, например 12 или 13. **Журнал меток** назвать по следующему шаблону — `timelogXФамилияN`, где `X` — номер выполняемого задания, `Фамилия` — заменить на вашу фамилию латиницей и строчными буквами, `N` — номер группы, например 12 или 13.

3.2. Войдите в систему на виртуальных терминалах 1 и 2 (`tty3`, `tty4`) под учетной записью `studentLastNameN`. Выполните следующие задачи:

a) Переключитесь в терминал `tty3` и запустите на переднем плане бесконечный процесс.

b) Переключитесь в виртуальный терминал `tty4` и проверьте работу процесса, запущенного в предыдущем пункте.

c) Переключитесь в виртуальный терминал `tty3` и приостановите работающий процесс.

d) Возобновите работу процесса в виртуальном терминале `tty3` в фоновом режиме.

e) Запустите в виртуальном терминале `tty3` второй бесконечный процессов запущенных в фоновом режиме.

f) Выполните команду `jobs` и проверьте, что два запущенных процесса работают.

g) Установите приоритет для любого процесса, запущенного ранее, равным 10.

h) Продемонстрируйте останов и возобновление работающего процесса по идентификатору процесса и номеру задания, используя соответствующий сигнал.

i) Прервите исполнение работающего процесса и выведите сведения о работающих заданиях.

j) Используя команды ps и head, выведите сведения о первых 3 процессах. Вывод должен содержать следующие сведения о процессах как USER, PID, %CPU, %MEM, VSZ, RSS, TTY, STAT, START, TIME, COMMAND.

k) Напишите команду, которая выводит список файлов в текущем каталоге, фильтрует список по расширению *.txt, заменяет в именах отфильтрованных файлов букву 'e' на 'E' и записать результат в файл ls_log.txt.

l) Используя команды ls, tee, wc, перенаправить результат детального вывода команды ls и подсчитать количество строк, слов и символов, сохранив результат в файл wordscount.txt.

m) Запустите порожденную оболочку `bash`. Исследуйте, посылая родительской оболочке сигналы `TERM`, `INT`, `QUIT` и `HUP`, что при этом происходит?

n) От имени обычного пользователя пошлите сигнал `KILL` любому процессу, запущенному от имени другого пользователя. Что произойдет?

o) Запустите в фоновом режиме команду `sleep 1000`. Проверьте, на какие сигналы из следующих: `TERM`, `INT`, `QUIT` и `HUP`, реагирует эта команда.

p) От имени суперпользователя запустите команду индексирования базы данных поиска в следующем виде: `time nice -n19 updatedb`. Затем выполните такую же команду, в которой значение `nice number` для `updatedb` будет `5`. Сравните полученные результаты.

q) При помощи команды `at` сделать так, чтобы ровно через 5 минут от текущего времени произошла запись списка всех процессов в файл с именем, содержащим в своём названии системное время на момент записи.

r) При помощи команды `at` организовать обычное завершение работы браузера `firefox` или `chrome` в 16:00.

s) Сделать при помощи `cron` так, чтобы команда `updatedb` запускалась раз в сутки, каждый час, каждые 5 минут.

t) Завершите ведение протокола.

3.3. Ответьте на вопросы и приведите примеры команд:

a) Приведите пример команды, которая завершает все процессы пользователя по имени пользователя.

b) Приведите пример команды, позволяющей получить сведения о процессах пользователя, выполняемых в фоновом режиме.

c) Приведите пример команды, позволяющей запустить процесс команды sleep 100 с приоритетом 4.

d) Приведите пример команды, позволяющей просмотреть сведения только о первых 7 процессах.

e) Приведите пример команды, выводящей первые 5 процессов с сортировкой по убыванию по потреблению памяти и пример команды, выводящей 5 с сортировкой по возрастанию по потреблению ресурсов процессора.

f) Найдите файлы а) пустые б) скрытые в домашнем каталоге в фоновом режиме и результат сохраните в файл со своей фамилией.

3.4. Изучите справку к команде `ps` и статью https://www.tecmint.com/ps-command-examples-for-linux-process-monitoring/, выполнить ее с ключами `-a, -e, a, x, ax`, записать результаты в файл, например: `ps -e > ps.log`.

3.5. Изучите справку к команде `pidof` и статью https://www.tecmint.com/find-process-name-pid-number-linux/. Запустите команду `top`. А в другом терминале, используя `pidof` определите ее идентификатор процесса. Завершите выполнение команды `top`.

3.6. Изучите справку к команде `fuser` и статью https://www.tecmint.com/learn-how-to-use-fuser-command-with-examples-in-linux/.
Откройте любой текстовый файл на редактирование. Определите, какие процессы используют ваш файл.

3.7. Изучите справку к команде `vmstat`. Результаты вывода команды запишите в файл с указанием в названии файла вашу фамилию и имя и название используемой команды.

3.8. Изучите справку к команде `lsof`. Результаты вывода команды запишите в файл с указанием в названии файла вашу фамилию и имя и название используемой команды.

3.9. Скопируйте сохранённые протоколы работы и добавьте в git-репозиторий.

3.10. Опубликуйте изменения из локального репозитория, т. е. результаты выполнения задания 3, в удаленный.

# 6 Приложение 1

## 6.1 Синтаксис основных команд

**ls** — вывод содержимого каталога. **Синтаксис**: ls [опции] [файл...]
**Описание:** Команда ls сначала выводит список всех файлов (не каталогов), перечисленных в командной строке, а затем выводит список всех файлов, находящихся в каталогах, перечисленных в командной строке. Если не указано ни одного файла, то по умолчанию аргументом назначается '.' (текущий каталог). Опция -d заставляет ls не считать аргументы-каталоги каталогами. Будут отображаться только файлы, которые не начинаются с '.' или все файлы, если задана опция -a.
Результаты печатаются на стандартный вывод, по одному файлу на строку, если с помощью опции -C не задан многоколоночный вывод.
Каждый список файлов (для файлов, которые не являются каталогами и для каждого каталога, содержащего список файлов) сортируется отдельно в алфавитной последовательности.
**Опции:**
-l В дополнение к имени каждого файла, выводятся тип файла, права доступа к файлу, количество ссылок на файл, имя владельца, имя группы, размер файла в байтах и временной штамп (время последней модификации файла, если не задано другое).
-a Выдавать все файлы в каталогах, включая все файлы и подкаталоги, имена которых начинаются с '.'.
-d Выдавать имена каталогов, как будто они обычные файлы, вместо того, чтобы показывать их содержимое.
-L Выдавать информацию о файлах, на которые указывают символические ссылки, вместо информации о самих символических ссылках.
-R Рекурсивно выдавать список содержимого всех каталогов.
-h Добавлять к каждому размеру файла букву размера, например, M (мегабайт).
-X Производить сортировку в алфавитном порядке по расширениям файлов (символы после последней '.'); файлы без расширений будут показаны первыми.
-S Производить сортировку по размеру файла, вместо сортировки по алфавиту. Таким образом, наибольшие файлы будут показаны сначала.
-c Сортировать содержимое каталога в соответствии с временем изменения состояния файла. Если с помощью опции -l задан длинный формат, то выдавать время изменения состояния файла вместо времени его модификации.
-t Сортировать по времени последней модификации вместо того, чтобы производить сортировку по алфавиту. Самые свежие файлы будут отображаться первыми.
-u Сортировать по времени последнего доступа к файлу, вместо времени последней модификации.
Типы файлов (первая буква в строке при задании опции -l) могут принимать следующие значения: — для обычного файла, d для каталога, b для блочного устройства, c для символьного устройства, l для символической ссылки, p для FIFO и s для гнезда (socket)
Права доступа составляют 9 символов и делятся на три группы по три символа: права доступа владельца, других пользователей из его группы, всех прочих пользователей. Права обозначаются следующим образом:
r Право на чтение.
w Право на запись.
x Право на выполнение (поиск в каталоге).
– Данное право доступа отсутствует.

Для каталога под правом на выполнение подразумевается право на просмотр в поисках требуемого файла.

**cd** — смена текущего каталога. **Синтаксис:** cd [каталог]
**Описание:** cd изменяет текущий каталог на каталог. Имя каталог может задаваться абсолютным (от корневого каталога) — в этом случае оно начинается с символа '/' — или относительным (от текущего каталога) — в этом случае оно начинается с символов './' или '../'. Если каталог не указан, текущим становится "домашний" каталог пользователя, определяемый значением переменной окружения $HOME.

**pwd** — выдача имени текущего каталога. **Синтаксис:** pwd
**Описание:** Команда pwd выдает имя текущего (рабочего) каталога.

**mkdir** — создание каталога. **Синтаксис:** mkdir [опции] каталог...
**Описание**: Команда mkdir создает каталоги с заданными именами.
По умолчанию права доступа к каталогам устанавливаются в 0777 (`a+rwx').
**Опции:**
-m права Устанавливает права доступа к создаваемым каталогам. Эти права могут быть заданы либо в символьном виде, либо в виде восьмеричного числа, как описано в

**rmdir** — удаление пустых каталогов. **Синтаксис** rmdir [опции] каталог...
**Описание.** Команда rmdir удаляет пустые каталоги. Если какой-либо из аргументов каталог не указывает на существующий пустой каталог, то будет выдано сообщение об ошибке.
**Опции**:
-p Если каталог включает более, чем один компонент пути, то удаляется каталог, затем убирается последний компонент пути и удаляется получившийся каталог и т.д. до тех пор, пока все компоненты не будут удалены. Таким образом, команда rmdir -p a/b/c эквивалентна rmdir a/b/c; rmdir a/b; rmdir a.

**man** — форматирование и отображение онлайновых справочных страниц. **Синтаксис:** man [раздел] имя...
**Описание:** Команда man выполняет форматирование и отображение онлайновых справочных страниц Unix. Если задан раздел, то man ищет только в заданном разделе руководства. имя — это обычно имя страницы руководства, которое, как правило, является именем команды, функции или файла.
Имеющиеся справочные страницы разбиты на несколько разделов. Важнейшими являются разделы:
1 — команды Unix;
2, 3 — системные вызовы Unix;
4, 5 — форматы файлов Unix.
Когда команда отображает страницу подсказки, в нижней строке экрана выводится приглашение man — символ ':'. После приглашения можно вводить внутренние команды man. В кратком руководстве следует упомянуть только две внутренние команды man:
h получение подробной информации о внутренних командах man;
q выход из man или переход к следующей странице, если команда man была введена с указанием нескольких имен команд.
Двигаться по отображаемой странице можно при помощи клавиш управления курсором.
Для получения более подробной информации о команде man введите: man man

**info** — отображение онлайновых справочных страниц. **Синтаксис:** info имя...
**Описание:** Команда info выполняет форматирование и отображение онлайновых справочных страниц Linux.

В текстах, отображаемых командой, могут быть наборы строк, озаглавленные "* Menu", каждая строка такого набора начинается с символа "*". Выбрав курсором пункт меню и нажав клавишу Enter, можно получить страницу подсказки по этому пункту.
Независимо от положения курсора, после приглашения можно вводить внутренние команды info. В кратком руководстве следует упомянуть только две внутренние команды info:
h получение подробной информации о внутренних командах info;
q выход из info или переход к следующей странице, если команда info была введена с указанием нескольких имен команд.
Двигаться по отображаемой странице можно при помощи клавиш управления курсором.
**script** — протоколирование сеанса. **Синтаксис:** script [-a] файл
**Описание:** Команда script начинает "вложенный" сеанс и протоколирует весь терминальный ввод и вывод в заданном файле. Завершение вложенного сеанса и выполнения команды script происходит по нажатию комбинации клавиш Ctrl+D.
**Опции**:
-a добавление протокола нового сеанса к содержимому файла, если эта опция не задана, то файл создается заново.

**who** — кто в системе? **Синтаксис:** who [опции]
**Описание:** Команда who сообщает имя пользователя, имя терминальной линии, астрономическое время начала сеанса, продолжительность бездействия терминальной линии с момента последнего обмена, идентификатор процесса для каждого из пользователей, работающих в системе.
Сообщения команды who имеют следующий формат:
NAME STATE LINE TIME IDLE PID COMMENT
где NAME — входное имя пользователя; LINE — имя терминальной линии, под которым она фигурирует в каталоге /dev; TIME — время начала сеанса; IDLE — время (часы и минуты), протекшее с последнего момента активизации данной линии. Точка (.) свидетельствует о том, что это действующий терминал. PID — идентификатор процесса интерпретатора shell, обслуживающего данного пользователя; COMMENT — комментарий, характеризующий данную линию.
**Опции**:
-H отображение заголовков столбцов в выводимой информации
-i отображается поле IDLE
-q отображение только имен и количества пользователей, работающих в системе в данный момент; все прочие опции при этом игнорируются
-T аналогично -s, но при этом отображается также поле STATE, как:
+ терминал, на который можно передавать сообщения
- терминал, на который нельзя передавать сообщения
? терминал неисправен
-s выводятся только поля NAME, LINE и TIME; это опция по умолчанию.

**write** — передача сообщения другому пользователю. **Синтаксис:** write *адресат*
**Описание:** *Адресат* задается как сетевое имя пользователя. После запуска команда write устанавливает связь с адресатом и переходит в режим ожидания ввода. В момент установки связи на терминал адресата выводится сообщение:
Message from *отправитель* ...
Отправитель вводит любой текст, который отображается на терминале получателя. Отправитель заканчивает сообщение нажатием комбинации клавиш Ctrl+D в начале строки. У адресата окончание сообщение индицируется строкой: EOF

Получатель может заблокировать/разблокировать вывод сообщений на свой экран при помощи команды mesg. При попытке передать сообщение на заблокированный терминал отправитель получает диагностику:
write: *адресат* has messages disabled

**tee** — ответвление канала. **Синтаксис:** tee [опции]... [файл]...
**Описание:** Команда tee переписывает стандартный ввод на стандартный вывод и делает копии в файлах. Признаком окончания ввода является комбинация клавиш Ctrl+D.
**Опции:**
-a добавлять выводимую информацию в файлы, а не переписывать их с начала.

**cat** — слияние и вывод файлов. **Синтаксис:** cat [-опции] файл ...
**Описание:** Команда cat по очереди читает указанные файлы и выдает их содержимое на стандартный вывод. Так, например, cat f распечатывает содержимое файла f, а cat f1 f2 > f3 сливает первые два файла и помещает результат в третий. Чтобы добавить файл f1 к файлу f2, надо выполнить команду cat f1 >> f2. Если не указан ни один файл или среди аргументов встретился -, команда cat читает данные со стандартного ввода.
**Опции:**
-b Нумеруются непустые строки файла.
-s Нумеруются все строки файла. (Поле номера отделяется от текста символом табуляции).
-v Визуализация непечатных символов. Управляющие символы изображаются в виде ^X (CTRL+X); символ DEL (восьмеричное 0177) — в виде ^?. Символы, не входящие в набор ASCII (то есть с восьмым битом, установленным в 1) выдаются в виде M-x, где x — определяемый младшими семью битами символ.
С опцией -v можно использовать следующие опции:
-t Визуализация символов табуляции в виде ^I.
-e Визуализация символов перевода строки в виде $ (строка при этом все же переводится).
Если опция -v не указана, то опции -t и -e игнорируются.

**vi** — текстовый редактор. **Синтаксис:** vi имя_файла
**Описание:** Редактор vi имеет три режима:

1. Командный — в этом режиме можно перемещаться по файлу и выполнять редактирующие команды над текстом. Команды вызываются ОБЫЧНЫМИ ЛАТИНСКИМИ БУКВАМИ.
2. Ввода текста — в этом режиме обычные латинские буквы будут вставляться в текст.
3. Режим строчного редактора `vi` используется для управления файлами (типа сохранить файл, зачитать файл и т.д.)

VI в КОМАНДНОМ РЕЖИМЕ.
ЧТОБЫ ВЫЙТИ ИЗ ФАЙЛА БЕЗ СОХРАНЕНИЯ, нажмите:
ESC : q ! Enter
чтобы выйти из файла, сохранив изменения, нажмите:
ESC : w ! Enter
ESC : q Enter
выйти из файла с сохранением, одной командой:
ESC : wq Enter
для перехода В РЕЖИМ ВВОДА нужно нажать команды типа:
i вставлять здесь
A вставлять с конца строки

Cw заменять текущее слово
ESC для ВОЗВРАТА В КОМАНДНЫЙ РЕЖИМ
CTRL-[ для возврата в командный режим
для перехода В РЕЖИМ УПРАВЛЕНИЯ ФАЙЛАМИ нужно нажать :
Двигаться по файлу можно командами:
h,j,k,l влево, вниз, вверх, вправо
Ctrl-F на страницу вниз
Ctrl-B на страницу вверх
Подгоните курсор к нужному месту и нажмите
i перевод в режим ввода и вводите требуемый текст
ESC прекратить ввод, перейти в командный режим.
Подгоните курсор к ненужному месту и нажмите
x удалить символ
dd удалить строчку.
Еще парочка полезных команд:
o вставлять с новой строки (под текущей строкой)
a в режим ввода ЗА курсором
5yy запомнить 5 строчек
Подгоните курсор к нужному месту
p вставить запомненные строки под курсором
P вставить запомненные строки НАД курсором
J склеить две строки
/Шаблон поиска Enter поиск
n повторить поиск.

**cp** — копирование файлов и каталогов. **Синтаксис:** cp [опции] файл путь
cp [опции] файл каталог

**Описание:** Команда cp копирует файлы или каталоги.
Если последний аргумент является существующим каталогом, то cp копирует каждый исходный файл в этот каталог (сохраняя имена). В противном случае, если задано только два файла, то cp копирует первый файл во второй.
(Так, cp -R /a /b будет копировать /a в /b/a и /a/x в /b/a/x в случае, если /b уже существует, но эта же команда будет копировать /a в /b и /a/x в /b/x, если /b не существует).
По умолчанию cp не копирует каталоги (см. опцию -R).

**Опции:**

-d Копирует символьные ссылки как символьные ссылки, а не файлы, на которые они указывают, и сохраняет жесткие ссылки между исходными файлами в копиях.
-f Удаляет существующие файлы, в которые происходит копирование, и не задает вопросов перед тем, как это сделать.
-i Спрашивает, нужно ли перезаписывать существующие обычные файлы.
-l Делает жесткие ссылки вместо копирования обычных файлов (не каталогов).
-R Копировать каталоги рекурсивно, сохраняя специальные файлы (см. -r выше).
-v Выводить имя каждого файла перед его копированием.

**unlink** — вызывает системную функцию unlink для удаления указанного файла. **Синтаксис:** unlink file

**Описание:** вызывает системную функцию unlink для удаления указанного файла file.

**rm** — удаление файлов или каталогов. **Синтаксис:** rm [опции] файл...
**Описание:** Команда rm удаляет каждый заданный файл. По умолчанию каталоги не удаляются, но если задана опция -r, то будет удаляться все дерево каталогов ниже

заданного каталога, включая и заданный каталог (без ограничения на глубину дерева).
**Опции:**
-f Игнорировать несуществующие файлы и никогда не запрашивать подтверждение на удаления.
-i Выдавать запрос на удаление каждого файла. (Принята по умолчанию).
-r Рекурсивно удалять содержимое каталогов.
-v Выдавать имя каждого файла перед его удалением.

**ln** — создание ссылки на файл. **Синтаксис:** ln [-f] файл1 [файл2 ...] целевой_файл
**Описание:** Команда ln делает целевой_файл ссылкой на файл1. Файл1 не должен совпадать с целевым_файлом. Если целевой_файл является каталогом, то в нем создаются ссылки на файл1, файл2,... с теми же именами. Только в этом случае можно указывать несколько исходных файлов.
Если целевой_файл существует и не является каталогом, его старое содержимое теряется.
**Опции:**
-f удаление существующего целевого файла
-s создание символической ссылки (по умолчанию создается жесткая ссылка)

**chmod** — изменение режима доступа к файлам. **Синтаксис:** chmod режим файл...
**Описание:** Команда chmod изменяет права доступа к указанным файлам (среди которых могут быть каталоги) в соответствии с указанным режимом. Режим может быть задан в абсолютном или символьном виде.
*Абсолютный вид* — восьмеричное число, являющееся поразрядным ИЛИ следующих режимов (названы не все режимы):
00400 Доступен для чтения владельцем.
00200 Доступен для записи владельцем.
00100 Доступен для выполнения (в случае каталога — для просмотра) владельцем.
00040 Доступен для чтения членами группы.
00020 Доступен для записи членами группы.
00010 Доступен для выполнения (просмотра) членами группы.
00004 Доступен для чтения прочими пользователями.
00002 Доступен для записи прочими пользователями.
00001 Доступен для выполнения (просмотра) прочими пользователями.
*Символьный вид* основан на однобуквенных обозначениях, которые определяют класс доступа и права доступа для членов данного класса. Права доступа к файлу зависят от идентификатора пользователя и идентификатора группы, в которую он входит. Режим в целом описывается в терминах трех последовательностей, по три буквы в каждой:

| Владелец | Группа | Прочие |
|---|---|---|
| (u) | (g) | (o) |
| rwx | rwx | rwx |

Для задания режима доступа в символьном виде используется синтаксис:
[кому] операция права
Часть кому есть комбинация букв u, g и o (владелец, члены группы и прочие пользователи соответственно). Если часть кому опущена или указано a, то это эквивалентно ugo.
Операция может быть: + (добавить право), — (лишить права), = (в пределах данного класса присвоить права абсолютно, то есть добавить указанные права и отнять неуказанные).
Права — любая осмысленная комбинация следующих букв (не все):
r Право на чтение.
w Право на запись.

x Право на выполнение (поиск в каталоге).
Опустить часть права можно только если операция есть = (для лишения всех прав).
Если надо сделать более одного указания об изменении прав, то при использовании символьного вида в правах не должно быть пробелов, а указания должны разделяться запятыми. Например, команда

```
chmod u+w,go+x f1
```

добавит для владельца право писать в файл f1, а для членов группы и прочих пользователей — право выполнять файл. Права устанавливаются в указанном порядке.
Изменить режим доступа к файлу может только его владелец или суперпользователь.
Для просмотра прав доступа и контроля при их изменении используется команда ls с флагом -l.

**chown** — изменение владельца и группы файлов.
**Синтаксис:** chown [опции] пользователь[:группа] файл...
**Описание:** Команда chown изменяет владельца и/или группу для каждого заданного файла. В качестве имени владельца/группы берется первый аргумент, не являющийся опцией. Если задано только имя пользователя (или числовой идентификатор пользователя), то данный пользователь становится владельцем каждого из указанных файлов, а группа этих файлов не изменяется. Если за именем пользователя через двоеточие следует имя группы (или числовой идентификатор группы), без пробелов между ними, то изменяется также и группа файла.
**Опции:**
-R Рекурсивное изменение владельца для каталогов и их содержимого.

**ps** — вывод информации о состоянии процессов. **Синтаксис:** ps [опции]
**Описание:** Команда ps выводит в стандартный вывод информацию о текущем состоянии процессов.
**Опции:**
-a все процессы, кроме лидеров групп и процессов, не ассоциированных с терминалом.
-d все процессы, кроме лидеров групп.
-e все процессы.
-g*список* выбирать процессы по *списку* лидеров групп.
-p*список* выбирать процессы по *списку* идентификаторов процессов.
-t*список* выбирать процессы по *списку* терминалов.
-u*список* выбирать процессы по *списку* идентификаторов пользователей.
-f генерировать полный листинг.
-l генерировать листинг в длинном формате.
**Результат команды** ps:
Ниже приводятся заголовки колонок выдачи команды ps и объясняется смысл их содержимого. Буквы l или f означают, что эта колонка появляется соответственно при длинном или полном формате выдачи; отсутствие букв означает, что данная колонка выводится всегда.
F l Флаги (шестнадцатеричные), логическая сумма которых дает следующие сведения о процессе:
00 процесс терминирован; элемент таблицы процессов свободен;
01 системный процесс: всегда в основной памяти;
02 процесс трассируется родительским процессом;
04 родительский трассировочный сигнал остановил процесс; родительский процесс ждет;
08 процесс не может быть разбужен сигналом;
10 процесс в основной памяти;
20 процесс в основной памяти; блокирован до завершения события;

40 идет сигнал к удаленной системе;
80 процесс в очереди на ввод/вывод.

S l статус процесса:
O активный: обрабатывается процессором;
S спящий: ожидает завершения события;
R готов: стоит в очереди на выполнение;
I рождающийся: процесс создается;
Z состояние "зомби": процесс завершен, но родительский процесс не ждет этого;
T трассируемый: процесс остановлен сигналом, так как родительский процесс трассирует его;
X растущий: процесс ожидает получения большего объема основной памяти.

UID f,l идентификатор владельца процесса; при указании опции -f выдается входное имя пользователя.
PID идентификатор процесса .
PPID f,l идентификатор родительского процесса.
C f,l доля выделенного планировщиком времени ЦП.
STIME f время запуска процесса (часы:минуты:секунды); если процесс запущен более чем 24 часа назад, выдается месяц и день запуска.
PRI l приоритет процесса.
NI l поправка к приоритету.
ADDR l адрес процесса в памяти.
SZ l размер (в блоках по 512 байт) образа процесса в памяти.
WCHAN l адрес события, которого ожидает процесс.
TTY l управляющий терминал.
TIME l использованное процессом время.
COMMAND l имя программы; если указана опция -f, то выводится полное имя команды и ее аргументы.

## 6.2 Учетные записи в ОС Linux

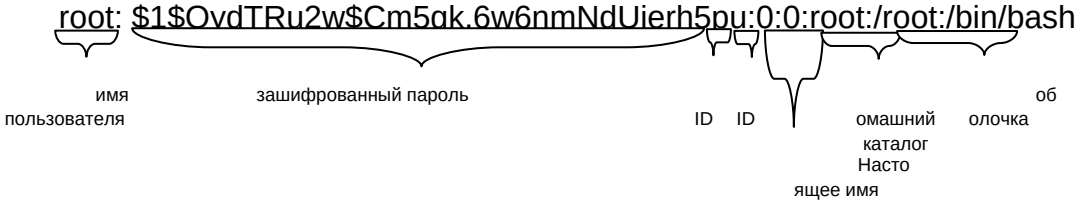
1. Linux, как и любая unix-подобная система, является не только многозадачной, но и многопользовательской, т.е эта операционная система позволяет одновременно нескольким пользователям работать с ней. Но система должна как-то узнавать, какой или какие из пользователей работают в данный момент. Именно для этих целей в Linux существует два понятия – *учетные записи* и *аутентификация*, которые являются частями одного механизма.

**Учетная запись пользователя** – это необходимая для системы информация о пользователе, хранящаяся в специальных файлах. Информация используется Linux для аутентификации пользователя и назначения ему прав доступа.

**Аутентификация** – системная процедура, позволяющая Linux определить, какой именно пользователь осуществляет вход.

Вся информация о пользователе обычно храниться в файлах /etc/passwd и /etc/grpoup.

**/etc/passwd** – этот файл содержит информацию о пользователях. Запись для каждого пользователя занимает одну строку:



пользователя

***имя пользователя*** – имя, используемое пользователем на все приглашения типа login при аутентификации в системе.

***зашифрованный пароль*** – обычно хешированный по необратимому алгоритму MD5 пароль пользователя или символ '!', в случаях, когда интерактивных вход пользователя в систему запрещен.

***UID*** – числовой идентификатор пользователя. Система использует его для распределения прав файлам и процессам.

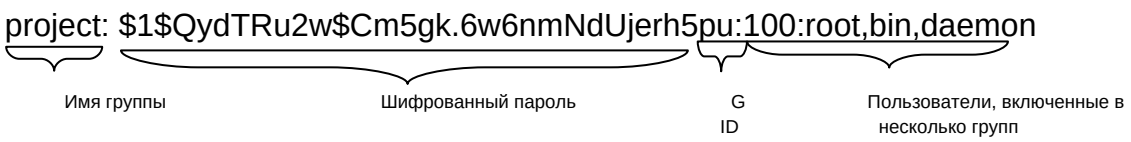
***GID*** – числовой идентификатор группы. Имена групп расположены в файле /etc/group. Система использует его для распределения прав файлам и процессам.

***Настоящее имя пользователя*** – используется в административных целях, а также командами типа finger (получение информации о пользователе через сеть).

***Домашний каталог*** – полный путь к домашнему каталогу пользователя.

***Оболочка*** – командная оболочка, которую использует пользователь при сеансе. Для нормальной работы она должна быть указана в файле регистрации оболочек /etc/shells.

**/etc/group** – этот файл содержит информацию о группах, к которым принадлежат пользователи:




***Имя группы*** – имя, используемое для удобства использования таких программ, как newgrp.

***Шифрованный пароль*** – используется при смене группы командой newgrp. Пароль для групп может отсутствовать.

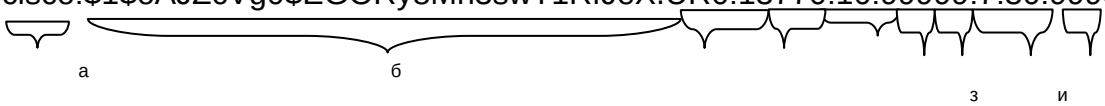
***GID*** – числовой идентификатор группы. Система использует его для распределения прав файлам и процессам.

***Пользователи, включенные в несколько групп*** – В этом поле через запятую отображаются те пользователи, у которых по умолчанию (в файле /etc/passwd) назначена другая группа.

На сегодняшний день хранение паролей в файлах passwd и group считается ненадежным. В новых версиях Linux применяются так называемые теневые файлы паролей – shadow и gshadow. Права на них назначены таким образом, что даже чтение этих фалов без прав суперпользователя невозможно. Нужно учесть, что нормальное функционирование системы при использовании теневых файлов подразумевает одновременно и наличие файлов passwd и group. При использовании теневых паролей в /etc/passwd и /etc/group вместо самого пароля устанавливается символ 'x', что и является указанием на хранение пароля в /etc/shadow или /etc/gshadow.

Файл shadow хранит защищенную информацию о пользователях, а также обеспечивает механизмы устаревания паролей и учетных записей. Вот структура файла shadow:

cisco:$1$oAJZcVg0$EGORy8Mh3swT1RfJeX.UR0:13770:10:99999:7:30:99999:




**а) имя пользователя**

**б) шифрованный пароль –** применяются алгоритмы хеширования, как правило MD5 или символ '!', в случаях, когда интерактивных вход пользователя в систему запрещен.

**в)** число дней последнего изменения пароля, начиная с 1 января 1970 года, последнего изменения пароля

**г)** число дней, перед тем как пароль может быть изменён

**д)** число дней, после которых пароль должен быть изменён

**е)** число дней, за сколько пользователя начнут предупреждать, что пароль устаревает

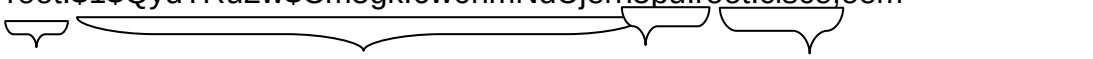
**ж)** число дней, после устаревания пароля для блокировки учётной записи

**з)** дней, отсчитывая с 1 января 1970 года, когда учётная запись будет заблокирована

**и)** зарезервированное поле

Файл gshadow так же накладывает дополнительную функциональность, вкупе с защищенным хранением паролей групп. Он имеет следующую структуру:

root:$1$QydTRu2w$Cm5gk.6w6nmNdUjerh5pu:root:cisco,oem




***Имя группы*** – имя, используемое для удобства использования таких программ, как newgrp.

***Шифрованный пароль*** – используется при смене группы командой newgrp. Пароль для групп может отсутствовать.

***Администратор группы*** – пользователь, имеющий право изменять пароль с помощью gpasswd.

***Список пользователей*** – В этом поле через запятую отображаются те пользователи, у которых по умолчанию (в файле /etc/passwd) назначена другая группа.

2. В Linux, кроме обычных пользователей, существует один (и только один) пользователь с неограниченными правами. Идентификаторы UID и GID такого пользователя всегда 0. Его имя, как правило, root, однако оно может быть легко изменено (или создано несколько символьных имен с одинаковым GID и UID), так как значение для применения неограниченных прав доступа имеет только GID 0. Для пользователя root права доступа к файлам и процессам не проверяются системой. При работе с использованием учетной записи root необходимо быть предельно осторожным, т.к. всегда существует возможность уничтожить систему.

3. В Linux используется развитая система распределения прав пользователям. Но для точного опознания пользователя одного имени недостаточно с точки зрения безопасности. Именно поэтому используется и *пароль – произвольный набор символов произвольной длины, обычно ограниченной лишь используемыми методами шифрования.*

Сегодня в большинстве версий Linux пароли шифруются по алгоритмам 3DES и MD5. Когда алгоритм 3DES является обратимым, то есть такой пароль можно

расшифровать, MD5 – это необратимое преобразование. Пароли, зашифрованные по алгоритму 3DES не применяются при использовании теневых файлов для хранения паролей.

При аутентификации, пароль, введенный пользователем, шифруется тем же методом, что и исходный, а потом сравниваются уже зашифрованные копии. Если они одинаковые, то аутентификация считается успешной.

Учитывая ежедневно увеличивающиеся требования к безопасности, в Linux есть возможность использовать скрытые пароли. Файлы /etc/passwd и /etc/group доступны для чтения всем пользователям, что является довольно большой брешью в безопасности системы. Именно поэтому в современных версиях Linux предпочтительнее использовать скрытые пароли. Такие пароли располагаются в файлах /etc/shadow и /etc/gshadow, для паролей пользователей и групп соответственно.

4. Команда *login* запускает сеанс интерактивной работы в системе. Она проверяет правильность ввода имени и пароля пользователя, меняет каталог на домашний, выстраивает окружение и запускает командный интерпретатор. Команду login как правило не запускают из командной строки — это обычно за пользователя делает менеджеров консоли — например getty или mgetty.

Команда *su (switch user)* позволяет сменить идентификатор пользователя уже в процессе сеанса. Синтаксис ее прост: *su username* , где username – имя пользователя, которое будет использоваться. После этого программа запросит пароль. При правильно введенном пароле, su запустит новый командный интерпретатор с правами пользователя, указанного su и присвоит сеансу его идентификаторы. Если имя пользователя опущено, то команда su использует имя root.

```
[student@ns student]$ su root
Password:
[root@ns student]#_
```

При использовании команды su пользователем root она, как правило, не запрашивает пароль.

Команда *newgrp* аналогична по своим возможностям su с той разницей, что происходит смена группы. Пользователь должен быть включен в группу, которая указывается в командной строке newgrp. При использовании команды newgrp пользователем root она никогда не запрашивает пароль. Синтаксис команды аналогичен синтаксису команды su: newgrp groupname, где groupname – имя группы, на которую пользователь меняет текущую.

Команда *passwd* является инструментом для смены пароля в Linux. Для смены своего пароля достаточно набрать в командной строке passwd:

```
[student@ns student]$ passwd
Changing password for student
(current) UNIX password:
New password:
Retype new password:
passwd: all authentication tokens updated successfully
[student@ns student]$_
```

Для смены пароля группы и управления группой используется команда *gpasswd* Для смены пароля достаточно набрать в командной строке gpasswd GROUPNAME. Сменить пароль вам удастся только если Вы являетесь администратором группы. Если пароль не пустой,то для членов группы вызов newgrp пароля не требует, а не члены группы должны ввести пароль. Администратор группы может добавлять и удалять пользователей с помощью параметров -a и -d соответственно. Администраторы могут использовать параметр -r для удаления пароля группы. Если пароль не задан, то только члены группы с помощью команды newgrp могут войти в группу. Указав параметр -R можно запретить доступ в группу по паролю с помощью команды newgrp (однако на членов группы это не распространяется). Системный администратор (root) может использовать параметр -A, чтобы назначить группе администратора.

Команда chage управляет информацией об устаревании пароля и учетной записи. Обычный пользователь (не root) может использовать команду только для просмотра своих параметров устаревания пароля:

```
gserg@ADM:/$ chage -l gserg
Last password change                                    : Май 03, 2007
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
```

Суперпользователь же может использовать также иные параметры, такие как:

-d дата (в формате системной даты, например ДД.ММ.ГГГГ) – устанавливает дату последней смены пароля пользователем.

-E дата – установить дату устаревания учетной записи пользователя

-I N – установить количество дней неактивности N с момента устаревания пароля перед тем как учетная запись будет заблокирована

-m N – задает минимальное количество дней (N) между сменами пароля

-M N – задает максимальное количество дней (N) между сменами пароля

-W N – задает количество дней, за которые будет выдаваться предупреждение об устаревании пароля.