

Отчет по лабораторной работе № 7

Шубина София Антоновна

Содержание

1	Цель работы	5
2	Выводы	17
	Список литературы	18

Список иллюстраций

1.1	2 пример	5
1.2	3 пример	5
1.3	4 пример	6
1.4	5 пример	6
1.5	6 пример	6
1.6	7 пример	6
1.7	8 пример	6
1.8	9 пример	6
1.9	10 пример	6
1.10	11 пример	7
1.11	12 пример	7
1.12	13 пример	7
1.13	Копирование файла	7
1.14	Создание директории	8
1.15	Перемещение файла в каталог	8
1.16	Переименование файла	8
1.17	Создание и копирование файла	8
1.18	Создание каталога	8
1.19	Создание и перемещение каталога	9
1.20	Опции команды <code>chmod</code>	9
1.21	Опции команды <code>chmod</code>	10
1.22	<code>chmod mount</code>	11
1.23	<code>chmod fsck</code>	12
1.24	<code>chmod mkfs</code>	13

Список таблиц

1 Цель работы

Ознакомление с файловой системой Linux, её структурой, именами и содержанием каталогов. Приобретение практических навыков по применению команд для работы с файлами и каталогами, по управлению процессами (и работами), по проверке использования диска и обслуживанию файловой системы. [1] # Выполнение лабораторной работы 1. Выполните все примеры, приведённые в первой части описания лабораторной работы. (рис. ??,1.1.1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9,1.10,1.11,1.12).

```
[sashubina@sashubina ~]$ cd
[sashubina@sashubina ~]$ touch abc1
[sashubina@sashubina ~]$ cp abc1 april
[sashubina@sashubina ~]$ cp abc1 may
[sashubina@sashubina ~]$
```

```
[sashubina@sashubina ~]$ mkdir monthly
[sashubina@sashubina ~]$ cp april may monthly
[sashubina@sashubina ~]$
```

Рис. 1.1: 2 пример

```
[sashubina@sashubina ~]$ cp monthly/may monthly/june
[sashubina@sashubina ~]$ ls monthly
april  june  may
[sashubina@sashubina ~]$
```

Рис. 1.2: 3 пример

4 пример

Рис. 1.3: 4 пример

5 пример

Рис. 1.4: 5 пример

```
[sashubina@sashubina ~]$ mv july monthly.00
[sashubina@sashubina ~]$ ls monthly.00
july  monthly
[sashubina@sashubina ~]$
```

Рис. 1.5: 6 пример

```
[sashubina@sashubina ~]$ mv monthly.00 monthly.01
[sashubina@sashubina ~]$
```

Рис. 1.6: 7 пример

```
[sashubina@sashubina ~]$ mkdir reports
[sashubina@sashubina ~]$ mv monthly.01 reports
[sashubina@sashubina ~]$
```

Рис. 1.7: 8 пример

```
[sashubina@sashubina ~]$ mv reports/monthly.01 reports/monthly
[sashubina@sashubina ~]$
```

Рис. 1.8: 9 пример

```
[sashubina@sashubina ~]$ cd
[sashubina@sashubina ~]$ touch may
[sashubina@sashubina ~]$ ls -l may
-rw-r--r--. 1 sashubina sashubina 0 мар 23 00:10 may
[sashubina@sashubina ~]$ chmod u+x may
[sashubina@sashubina ~]$ ls -l may
-rwxr--r--. 1 sashubina sashubina 0 мар 23 00:10 may
[sashubina@sashubina ~]$
```

Рис. 1.9: 10 пример

```
[sashubina@sashubina ~]$ chmod u-x may
[sashubina@sashubina ~]$ s -l may
bash: s: команда не найдена
[sashubina@sashubina ~]$ ls -l may
-rw-r--r--. 1 sashubina sashubina 0 май 23 00:10 may
[sashubina@sashubina ~]$
```

Рис. 1.10: 11 пример

```
sashubina@sashubina ~]$ rm -R monthly
sashubina@sashubina ~]$ cd
sashubina@sashubina ~]$ mkdir monthly
sashubina@sashubina ~]$ chmod g-r, o-r monthly
chmod: неверный режим: «g-r,»
по команде «chmod --help» можно получить дополнительную информацию.
sashubina@sashubina ~]$ chmod go-r monthly
sashubina@sashubina ~]$
```

Рис. 1.11: 12 пример

```
[sashubina@sashubina ~]$ cd
[sashubina@sashubina ~]$ touch abc1
[sashubina@sashubina ~]$ chmod g+w abc1
[sashubina@sashubina ~]$
```

Рис. 1.12: 13 пример

2. Выполните следующие действия, зафиксировав в отчёте по лабораторной работе используемые при этом команды и результаты их выполнения: 2.1. Скопируйте файл /usr/include/sys/io.h в домашний каталог и назовите его equipment. Если файла io.h нет, то используйте любой другой файл в каталоге /usr/include/sys/ вместо него. (рис. 1.13).

```
[sashubina@sashubina ~]$ cp /usr/include/sys/io.h equipment
[sashubina@sashubina ~]$
```

Рис. 1.13: Копирование файла

- 2.2. В домашнем каталоге создайте директорию ~/ski.places. (рис. 1.14).

```
[sashubina@sashubina ~]$ mkdir ski.plases  
[sashubina@sashubina ~]$
```

Рис. 1.14: Создание директории

2.3. Переместите файл equipment в каталог ~/ski.plases. (рис. 1.15).

```
[sashubina@sashubina ~]$ mv equipment ski.plases  
[sashubina@sashubina ~]$
```

Рис. 1.15: Перемещение файла в каталог

2.4. Переименуйте файл ~/ski.plases/equipment в ~/ski.plases/equiplist. (рис. 1.16).

```
[sashubina@sashubina ~]$ mv ski.plases/equipment ski.plases/equiplis  
[sashubina@sashubina ~]$
```

Рис. 1.16: Переименование файла

2.5. Создайте в домашнем каталоге файл abc1 и скопируйте его в каталог ~/ski.plases, назовите его equiplist2. (рис. 1.17).

```
[sashubina@sashubina ~]$ cp abc1 ski.plases/equiplist2  
[sashubina@sashubina ~]$
```

Рис. 1.17: Создание и копирование файла

2.6. Создайте каталог с именем equipment в каталоге ~/ski.plases. (рис. 1.18).

```
[sashubina@sashubina ~]$ mkdir ski.plases/equipment  
[sashubina@sashubina ~]$
```

Рис. 1.18: Создание каталога

2.7. Переместите файлы ~/ski.plases/equiplist и equiplist2 в каталог ~/ski.plases/equipment.

2.8. Создайте и переместите каталог ~/newdir в каталог ~/ski.plases и назовите его plans. (рис. 1.19).


```
[sashubina@sashubina ~]$ mv newdir ski.plases/plans
[sashubina@sashubina ~]$
```

Рис. 1.19: Создание и перемещение каталога

3. Определите опции команды `chmod`, необходимые для того, чтобы присвоить перечисленным ниже файлам выделенные права доступа, считая, что в начале таких прав нет:
- 3.1. `drwxr-r- ... australia`
 - 3.2. `drwx-x-x ... play`
 - 3.3. `-r-xr-r- ... my_os`
 - 3.4. `-rw-rw-r- ... feathers`
- При необходимости создайте нужные файлы. (рис. 1.20).

```
[sashubina@sashubina ~]$ mkdir australia
[sashubina@sashubina ~]$ mkdir play
[sashubina@sashubina ~]$ touch my_os
[sashubina@sashubina ~]$ touch feathers
[sashubina@sashubina ~]$ chmod 744 australia
[sashubina@sashubina ~]$ chmod 711 play
[sashubina@sashubina ~]$ chmod 544 my_os
[sashubina@sashubina ~]$ chmod 664 feathres
chmod: невозможно получить доступ к 'feathres': Нет такого файла и каталога
[sashubina@sashubina ~]$ chmod 664 feathers
[sashubina@sashubina ~]$
```

Рис. 1.20: Опции команды `chmod`

4. Прodelайте приведённые ниже упражнения, записывая в отчёт по лабораторной работе используемые при этом команды:
- 4.1. Просмотрите содержимое файла `/etc/passwd`.
 - 4.2. Скопируйте файл `~/feathers` в файл `~/file.old`.
 - 4.3. Переместите файл `~/file.old` в каталог `~/play`.
 - 4.4. Скопируйте каталог `~/play` в каталог `~/fun`.
 - 4.5. Переместите каталог `~/fun` в каталог `~/play` и назовите его `games`.
 - 4.6. Лишите владельца файла `~/feathers` права на чтение.
 - 4.7. Что произойдёт, если вы попытаетесь просмотреть файл `~/feathers` командой `cat`?
 - 4.8. Что произойдёт, если вы попытаетесь скопировать файл `~/feathers`?
 - 4.9. Дайте владельцу файла `~/feathers` право на чтение.
 - 4.10. Лишите владельца каталога `~/play` права на выполнение.

4.11. Перейдите в каталог ~/play. Что произошло? 4.12. Дайте владельцу каталога ~/play право на выполнение. (рис. 1.21).

```
sashubina@sashubina ~]$ cp feathers file.old
sashubina@sashubina ~]$ mv file.old play/
sashubina@sashubina ~]$ cp -r play/ fun/
sashubina@sashubina ~]$ mv fun play/games
sashubina@sashubina ~]$ chmod u-r feathers
sashubina@sashubina ~]$ cp feathers play/
cp: невозможно открыть 'feathers' для чтения: Отказано в доступе
sashubina@sashubina ~]$ chmod u+r feathers
sashubina@sashubina ~]$ chmod u-x play/
sashubina@sashubina ~]$ cd play/
bash: cd: play/: Отказано в доступе
sashubina@sashubina ~]$ chmod u+x play/
sashubina@sashubina ~]$ man mount
```

Рис. 1.21: Опции команды chmod

5. Прочитайте man по командам mount, fsck, mkfs, kill и кратко их охарактеризуйте, приведя пример (рис. 1.22,1.23,1.24,??).

DESCRIPTION

All files accessible in a Unix system are arranged in one big tree, the file hierarchy, rooted at `/`. These files can be spread out over several devices. The `mount` command serves to attach the filesystem found on some device to the big file tree. Conversely, the `umount(8)` command will detach it again. The filesystem is used to control how data is stored on the device or provided in a virtual way by network or other services.

The standard form of the `mount` command is:

```
mount -t type device dir
```

This tells the kernel to attach the filesystem found on *device* (which is of type *type*) at the directory *dir*. The option `-t type` is optional. The `mount` command is usually able to detect a filesystem. The root permissions are necessary to mount a filesystem by default. See section "Non-superuser mounts" below for more details. The previous contents (if any) and owner and mode of *dir* become invisible, and as long as this filesystem remains mounted, the pathname *dir* refers to the root of the filesystem on *device*.

If only the directory or the device is given, for example:

```
mount /dir
```

then `mount` looks for a mountpoint (and if not found then for a device) in the `/etc/fstab` file. It's possible to use the `--target` or `--source` options to avoid ambiguous interpretation of the given argument. For example:

Manual page mount(8) line 21 (press h for help or q to quit)

Рис. 1.22: `chmod mount`

DESCRIPTION

fsck is used to check and optionally repair one or more Linux filesystems. **filesystem** can be a device name (e.g., `/dev/hdc1`, `/dev/sdb2`), a mount point (e.g., `/`, `/usr`, `/home`), or a filesystem label or UUID specifier (e.g., `UUID=8868abf6-88c5-4a83-98b8-bfc24057f7bd` or `LABEL=root`). Normally, the **fsck** program will try to handle filesystems on different physical disk drives in parallel to reduce the total amount of time needed to check all of them.

If no filesystems are specified on the command line, and the **-A** option is not specified, **fsck** will default to checking filesystems in `/etc/fstab` serially. This is equivalent to the **-As** options.

The exit status returned by **fsck** is the sum of the following conditions:

- 0
No errors
- 1
Filesystem errors corrected
- 2

Manual page: fsck(8) 11m 4 (fsck is for help on -A -s -t -u)

Рис. 1.23: chmod fsck

```
mkfs(8)                                System Administration                                mkfs(8)

NAME
    mkfs - build a Linux filesystem

SYNOPSIS
    mkfs [options] [-t type] [fs-options] device [size]

DESCRIPTION
    This mkfs frontend is deprecated in favour of filesystem
    specific mkfs.<type> utils.

    mkfs is used to build a Linux filesystem on a device, usually a
    hard disk partition. The device argument is either the device
    name (e.g., /dev/hda1, /dev/sdb2), or a regular file that shall
    contain the filesystem. The size argument is the number of
    blocks to be used for the filesystem.

    The exit status returned by mkfs is 0 on success and 1 on
    failure.

    In actuality, mkfs is simply a front-end for the various
    filesystem builders (mkfs.fstype) available under Linux. The
    filesystem-specific builder is searched for via your PATH
    environment setting only. Please see the filesystem-specific
    builder manual pages for further details.

OPTIONS
    -t, --type type
        Specify the type of filesystem to be built. If not
        specified, the default filesystem type (currently ext2) is
        used.

    fs-options

Manual page mkfs(8) line 1 (press h for help or q to quit)
```

Рис. 1.24: chmod mkfs

```
foot
KILL(1)                                User Commands                                KILL(1)

NAME
    kill - terminate a process

SYNOPSIS
    kill [-signal|-s signal|-p] [-q value] [-a] [--timeout
    milliseconds signal] [--] pid|name...

    kill -l [number] | -L

DESCRIPTION
    The command kill sends the specified signal to the specified
    processes or process groups.

    If no signal is specified, the TERM signal is sent. The default
    action for this signal is to terminate the process. This signal
    should be used in preference to the KILL signal (number 9),
    since a process may install a handler for the TERM signal in
    order to perform clean-up steps before terminating in an orderly
    fashion. If a process does not terminate after a TERM signal has
    been sent, then the KILL signal may be used; be aware that the
    latter signal cannot be caught, and so does not give the target
    process the opportunity to perform any clean-up before
    terminating.

    Most modern shells have a builtin kill command, with a usage
    rather similar to that of the command described here. The --all,
    --pid, and --queue options, and the possibility to specify
    processes by command name, are local extensions.

    If signal is 0, then no actual signal is sent, but error
    checking is still performed.

Manual page kill(1) line 1 (press h for help or q to quit)
```

1. mount - это команда в UNIX-подобных системах, которая позволяет подключать файловые системы к директориям в иерархии файловой системы. Пример: mount /dev/sdb1 /mnt/usb 2. fsck - это утилита для проверки и восстановления целостности файловой системы. Она исправляет ошибки и испорченные блоки данных на диске. Пример: fsck /dev/sda1 3. mkfs - это команда для создания новой файловой системы на блочном устройстве, например жестком диске или флэш-накопителе. Пример: mkfs.ext4 /dev/sdb1 4. kill - команда, используемая для отправки сигналов процессам, позволяющая завершить процессы или изменить их поведение. Пример: kill -9 1234 (рис. ??).



Контрольные во-

просы 1. На моем компьютере существуют файловые системы NTFS (Windows) и ext4 (Linux). NTFS - проприетарная файловая система, разработанная компанией Microsoft, хорошо поддерживает большие файлы и объемы дисков, но не полностью совместима с Linux. ext4 - стандартная файловая система для большинства дистрибутивов Linux, хорошо поддерживает разделение на разные разделы и файлы любого размера.

2. Общая структура файловой системы включает в себя директории первого уровня, такие как /bin (для исполняемых файлов), /etc (для конфигурационных файлов), /home (для домашних папок пользователей), /var (для переменных данных), /tmp (для временных файлов).
3. Для доступности содержимого файловой системы операционной системе необходимо выполнить монтирование диска или раздела, на котором находится эта файловая система.
4. Основные причины нарушения целостности файловой системы могут быть физические повреждения диска, отключение питания во время работы, ошибки программного обеспечения. Для восстановления целостности файловой системы можно использовать утилиты типа fsck.

5. Файловая система создается при форматировании диска или раздела, в процессе которого на нем создаются структуры для хранения файлов и метаданных.
6. Команды для просмотра текстовых файлов включают в себя `cat`, `less`, `more`, `head`, `tail`. Например, команда `cat` отображает содержимое файла целиком.
7. Основные возможности команды `cp` в Linux включают копирование файлов с сохранением атрибутов, возможность копирования нескольких файлов или папок, переименование файлов при копировании.
8. Основные возможности команды `mv` в Linux включают перемещение файла или папки из одного места в другое, изменение имени файла или папки при перемещении.
9. Права доступа в файловой системе определяют, кто и как может обращаться к файлам и папкам. Они могут быть изменены с помощью команды `chmod` в Linux, которая позволяет устанавливать различные права на чтение, запись и исполнение для владельца, группы и остальных пользователей.

2 Выводы

Я ознакомилась с файловой системой Linux. ее структурой, именами и содержанием каталогов. Приобрела практические навыки по применению команд для работы с файлами и каталогами, по управлению процессами (и работами), по проверке использования диска и обслуживанию файловой системы.

Список литературы

1. Learning the bash Shell: Unix Shell Programming.