

Лабораторная работа №5

Анализ файловой системы Linux. Команды для работы с файлами и каталогами

Дашкина Анита

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	8
4	Выводы	16

Список иллюстраций

3.1	выполнение команд	8
3.2	выполнение команд	8
3.3	выполнение команд	9
3.4	выполнение команд	9
3.5	результат	10
3.6	результат	10
3.7	результат	10
3.8	результат	10
3.9	выполнение команд	11
3.10	выполнение команд	11
3.11	man	12
3.12	man mount	12
3.13	man fsck	13
3.14	man mkfs	14
3.15	man kill	15

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.

2 Задание

1. Выполните все примеры, приведённые в первой части описания лабораторной работы.

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. 2.2. В домашнем каталоге создайте директорию `~/ski.places`. 2.3. Переместите файл `equipment` в каталог `~/ski.places`. 2.4. Переименуйте файл `~/ski.places/equipment` в `~/ski.places/equiplist`. 2.5. Создайте в домашнем каталоге файл `abc1` и скопируйте его в каталог `~/ski.places`, назовите его `equiplist2`. 2.6. Создайте каталог с именем `equipment` в каталоге `~/ski.places`. 2.7. Переместите файлы `~/ski.places/equiplist` и `equiplist2` в каталог `~/ski.places/equipment`. 2.8. Создайте и переместите каталог `~/newdir` в каталог `~/ski.places` и назовите его `plans`.

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.

4. Прделаем 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. 5. Прочитайте `man` по командам `mount`, `fsck`, `mkfs`, `kill` и кратко их охарактеризуйте, приведя примеры.

3 Выполнение лабораторной работы

1. Выполняем все примеры, приведённые в первой части описания лабораторной работы.

```
atdashkina@dk3n31 ~$ cp ~/feathers ~/file.old
cp: не указан -r; пропущается каталог '/afs/.dk.sci.pfu.edu.ru/home/a/t/atdashkina/feathers'
atdashkina@dk3n31 ~$ cp -r ~/feathers ~/file.old
atdashkina@dk3n31 ~$ mv ~/file.old ~/play
atdashkina@dk3n31 ~$ cp -r ~/play ~/fun
atdashkina@dk3n31 ~$ mv ~/fun ~/play
atdashkina@dk3n31 ~$ mv ~/fun/play ~/fun/games
mv: не удалось выполнить stat для '/afs/.dk.sci.pfu.edu.ru/home/a/t/atdashkina/fun/play': Нет такого файла или каталога
atdashkina@dk3n31 ~$ ls
bin      play      work      Документы  Музыка      'Снимки экрана'
feathers public    архив     Загрузки   Общедоступные  Шаблоны
home     public_html Видео     Изображения 'Рабочий стол'
atdashkina@dk3n31 ~$ cd play
atdashkina@dk3n31 ~/play $ mv fun games
atdashkina@dk3n31 ~/play $ ls
file.old games
atdashkina@dk3n31 ~/play $
```

Рис. 3.1: выполнение команд

```
atdashkina@dk3n31 ~$ cp /usr/include/sys/io.h equipment
atdashkina@dk3n31 ~$ ls
bin      home     [redacted] Видео     Изображения  'Рабочий стол'
equipment play     work      Документы  Музыка      'Снимки экрана'
feathers public  архив     Загрузки   Общедоступные  Шаблоны
atdashkina@dk3n31 ~$ mkdir ~/ski.places
atdashkina@dk3n31 ~$ ls
bin      home     [redacted] архив     Загрузки   Общедоступные  Шаблоны
equipment play     ski.places Видео     Изображения  'Рабочий стол'
feathers public  work      Документы  Музыка      'Снимки экрана'
atdashkina@dk3n31 ~$ mv equipment ski.places
atdashkina@dk3n31 ~$ mv ~/ski.places/equipment ~/ski.places/equiplist
atdashkina@dk3n31 ~$ ls ski.places
equiplist
atdashkina@dk3n31 ~$
```

Рис. 3.2: выполнение команд

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. 2.2. В домашнем каталоге создайте директорию ~/ski.plases. 2.3. Переместите файл equipment в каталог ~/ski.plases. 2.4. Переименуйте файл ~/ski.plases/equipment в ~/ski.plases/equiplist.

```
atdashkina@dk3n31 ~ $ touch abc1
atdashkina@dk3n31 ~ $ cp ~/ski.plases equiplist2
cp: не указан -r; пропускается каталог '/afs/.dk.sci.pfu.edu.ru/home/a/t/atdashkina/ski.plases'
atdashkina@dk3n31 ~ $ cp -r ~/ski.plases equiplist2
atdashkina@dk3n31 ~ $ cd ~/ski.plases
atdashkina@dk3n31 ~/ski.plases $ ls
equiplist
atdashkina@dk3n31 ~/ski.plases $ cp equiplist2
cp: после 'equiplist2' пропущен операнд, задающий целевой файл
По команде «cp --help» можно получить дополнительную информацию.
atdashkina@dk3n31 ~/ski.plases $ cp ~/abc1 equiplist2
atdashkina@dk3n31 ~/ski.plases $ ls
equiplist equiplist2
atdashkina@dk3n31 ~/ski.plases $ mkdir equipment
atdashkina@dk3n31 ~/ski.plases $ ls
equiplist equiplist2 equipment
atdashkina@dk3n31 ~/ski.plases $ mv equiplist equiplist2 equipment
atdashkina@dk3n31 ~/ski.plases $ ls equipment
equiplist equiplist2
atdashkina@dk3n31 ~/ski.plases $ mkdir ~/newdir
atdashkina@dk3n31 ~/ski.plases $ mv ~/newdir plans
atdashkina@dk3n31 ~/ski.plases $ ls
equipment plans
atdashkina@dk3n31 ~/ski.plases $
```

Рис. 3.3: выполнение команд

2.5. Создайте в домашнем каталоге файл abc1 и скопируйте его в каталог ~/ski.plases, назовите его equiplist2. 2.6. Создайте каталог с именем equipment в каталоге ~/ski.plases. 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.

```
atdashkina@dk3n31 ~ $ chmod g-x australia
atdashkina@dk3n31 ~ $ chmod o-x australia
atdashkina@dk3n31 ~ $ ls -l australia
итого 0
atdashkina@dk3n31 ~ $ ls -l
итого 41
-rw-r--r-- 1 atdashkina studsci 0 мар 13 15:06 abc1
drwxr--r-- 2 atdashkina studsci 2048 мар 13 15:13 australia
drwxr-xr-x 2 atdashkina studsci 2048 мар 1 13:54 bin
drwxr-xr-x 2 atdashkina studsci 2048 мар 13 15:07 equiplist2
drwxr-xr-x 2 atdashkina studsci 2048 мар 13 14:52 feathers
drwxr-xr-x 3 atdashkina studsci 2048 фев 26 11:41 home
drwxr-xr-x 3 atdashkina studsci 2048 мар 13 14:59 play
drwxr-xr-x 2 atdashkina studsci 2048 мар 1 14:29 public
```

Рис. 3.4: выполнение команд

3. Определим 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

```
atdashkina@dk3n31 ~ $ chmod o-r play
atdashkina@dk3n31 ~ $ ls -l
итого 41
-rw-r--r-- 1 atdashkina studsci 0 map 13 15:06 abc1
drwxr--r-- 2 atdashkina studsci 2048 map 13 15:13 australia
drwxr-xr-x 2 atdashkina studsci 2048 map 1 13:54 bin
drwxr-xr-x 2 atdashkina studsci 2048 map 13 15:07 equiplist2
drwxr-xr-x 2 atdashkina studsci 2048 map 13 14:52 feathers
drwxr-xr-x 3 atdashkina studsci 2048 фев 26 11:41 home
drwxr-x--x 3 atdashkina studsci 2048 map 13 14:59 play
drwxr-xr-x 2 atdashkina studsci 2048 map 1 14:20 public
```

Рис. 3.5: результат

3.2. drwx-x-x ... play

```
atdashkina@dk3n31 ~ $ chmod u-w my_os
atdashkina@dk3n31 ~ $ chmod u*x my_os
chmod: неверный режим: «u*x»
По команде «chmod --help» можно получить дополнительную информацию.
atdashkina@dk3n31 ~ $ chmod u+x my_os
atdashkina@dk3n31 ~ $ ls -l
итого 43
-rw-r--r-- 1 atdashkina studsci 0 map 13 15:06 abc1
drwxr--r-- 2 atdashkina studsci 2048 map 13 15:13 australia
drwxr-xr-x 2 atdashkina studsci 2048 map 1 13:54 bin
drwxr-xr-x 2 atdashkina studsci 2048 map 13 15:07 equiplist2
drwxr-xr-x 2 atdashkina studsci 2048 map 13 14:52 feathers
drwxr-xr-x 3 atdashkina studsci 2048 фев 26 11:41 home
dr-xr-xr-x 2 atdashkina studsci 2048 map 13 15:16 my_os
```

Рис. 3.6: результат

3.3. -r-xr-r- ... my_os

```
atdashkina@dk2n24 ~ $ cat ~/feathers
cat: /afs/.dk.sci.pfu.edu.ru/home/a/t/atdashkina/feathers: Это каталог
```

Рис. 3.7: результат

3.4. -rw-rw-r- ... feathers

```
atdashkina@dk2n24 ~ $ cp feathers
cp: после 'feathers' пропущен операнд, задающий целевой файл
По команде «cp --help» можно получить дополнительную информацию.
```

Рис. 3.8: результат

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.

```
atdashkina@dk2n24 ~/play $ cd
atdashkina@dk2n24 ~ $ chmod u-r feathers
atdashkina@dk2n24 ~ $ chmod u+r feathers
```

Рис. 3.9: выполнение команд

4.6. Лишите владельца файла ~/feathers права на чтение. 4.7. Что произойдёт, если вы попытаетесь просмотреть файл ~/feathers командой cat?

```
atdashkina@dk2n24 ~ $ chmod u-x ~/play
atdashkina@dk2n24 ~ $ cd play
atdashkina@dk2n24 ~/play $ ls -l
итого 2
-rw-r--r-- 1 atdashkina studsci  0 map 13 14:53 file.old
drwxr-xr-x 2 atdashkina studsci 2048 map 13 14:55 games
atdashkina@dk2n24 ~/play $ cd
```

4.8. Что произойдёт,

если вы попытаетесь скопировать файл ~/feathers?

```
atdashkina@dk2n24 ~ $ chmod u+x ~/play
atdashkina@dk2n24 ~ $ ls-l
bash: ls-l: команда не найдена
atdashkina@dk2n24 ~ $ ls -l
итого 43
-rw-r--r-- 1 atdashkina studsci  0 map 13 15:06 abc1
drwxr--r-- 2 atdashkina studsci 2048 map 13 15:13 australia
```

Рис. 3.10: выполнение команд

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? ничего не произошло(

4.12. Дайте владельцу каталога ~/play право на выполнение.

!

5. Прочитаем man по командам mount, fsck, mkfs, kill и кратко их охарактеризуем, приведя примеры.

```

atdashkina@dk2n24 ~ $ man mount
atdashkina@dk2n24 ~ $ man fsck
atdashkina@dk2n24 ~ $ man mkfs
atdashkina@dk2n24 ~ $ man kill
atdashkina@dk2n24 ~ $

```

Рис. 3.11: man

```

MOUNT(8)                                     System Administration                                MOUNT(8)
NAME
    mount - mount a filesystem

SYNOPSIS
    mount [-h|-V]

    mount [-l] [-t fstype]

    mount -a [-ffnrsvw] [-t fstype] [-O optlist]

    mount [-fnrsvw] [-o options] device|mountpoint

    mount [-fnrsvw] [-t fstype] [-o options] device mountpoint

    mount --bind|--rbind|--move olddir newdir

    mount --make-[shared|slave|private|unbindable|rshared|rslave|rprivate|runbindable] mountpoint

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:

        mount --target /mountpoint

    The same filesystem may be mounted more than once, and in some cases (e.g., network filesystems) the same
    filesystem may be mounted on the same mountpoint multiple times. The mount command does not implement any
    policy to control this behavior. All behavior is controlled by the kernel and it is usually specific to the
    filesystem driver. The exception is --all, in this case already mounted filesystems are ignored (see --all
    below for more details).

    Listing the mounts
    Manual page mount(8) line 1 (press h for help or q to quit)

```

Рис. 3.12: man mount

```
FSCK(8)                                System Administration                                FSCK(8)

NAME
    fsck - check and repair a Linux filesystem

SYNOPSIS
    fsck [-lsAVRTMNP] [-r [fd]] [-C [fd]] [-t fstype] [filesystem...] [--] [fs-specific-options]

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      System should be rebooted

    4      Filesystem errors left uncorrected

    8      Operational error

    16     Usage or syntax error

    32     Checking canceled by user request

    128    Shared-library error

    The exit status returned when multiple filesystems are checked is the bit-wise OR of the exit statuses for
    each filesystem that is checked.

    In actuality, fsck is simply a front-end for the various filesystem checkers (fsck.fstype) available under
    Linux. The filesystem-specific checker is searched for in the PATH environment variable. If the PATH is
    undefined then fallback to /sbin.

    Please see the filesystem-specific checker manual pages for further details.
```

Рис. 3.13: man 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
    Filesystem-specific options to be passed to the real filesystem builder.

  -V, --verbose
    Produce verbose output, including all filesystem-specific commands that are executed. Specifying this option more than once inhibits execution of any filesystem-specific commands. This is really only useful for testing.

  -h, --help
    Display help text and exit.

  -v, --version
    Print version and exit. (Option -V will display version information only when it is the only parameter, otherwise it will work as --verbose.)

BUGS
  All generic options must precede and not be combined with filesystem-specific options. Some filesystem-specific programs do not automatically detect the device size and require the size parameter to be specified.

AUTHORS
  David Engel <david@ods.com>, Fred N. van Kempen <waltje@uwait.nl.mugnet.org>, Ron Sommeling <sommel@sci.kun.nl>.

  The manual page was shamelessly adapted from Remy Card's version for the ext2 filesystem.

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

Рис. 3.14: man mkfs

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

NAME
    kill - send a signal to a process

SYNOPSIS
    kill [options] <pid> [...]

DESCRIPTION
    The default signal for kill is TERM. Use -l or -L to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and 0. Alternate signals may be specified in three ways: -9, -SIGKILL or -KILL. Negative PID values may be used to choose whole process groups; see the PGID column in ps command output. A PID of -1 is special; it indicates all processes except the kill process itself and init.

OPTIONS
    <pid> [...]
        Send signal to every <pid> listed.

    -<signal>
    -s <signal>
    --signal <signal>
        Specify the signal to be sent. The signal can be specified by using name or number. The behavior of signals is explained in signal(7) manual page.

    -q, --queue value
        Use sigqueue(3) rather than kill(2) and the value argument is used to specify an integer to be sent with the signal. If the receiving process has installed a handler for this signal using the SA_SIGINFO flag to sigaction(2), then it can obtain this data via the si_value field of the siginfo_t structure.

    -l, --list [signal]
        List signal names. This option has optional argument, which will convert signal number to signal name, or other way round.

    -L, --table
        List signal names in a nice table.

NOTES
    Your shell (command line interpreter) may have a built-in kill command. You may need to run the command described here as /bin/kill to solve the conflict.

EXAMPLES
    kill -9 -1
        Kill all processes you can kill.

    kill -l 11
        Translate number 11 into a signal name.

    kill -L
        List the available signal choices in a nice table.

    kill 123 543 2341 3453
        Send the default signal, SIGTERM, to all those processes.

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

Рис. 3.15: man kill

4 Выводы

Мы ознакомились с файловой системой Linux, её структурой, именами и содержанием каталогов. Приобрели практические навыки по применению команд для работы с файлами и каталогами, по управлению процессами (и работами), по проверке использования диска и обслуживанию файловой системы.