

Отчёт по лабораторной работе №5

Анализ файловой структуры UNIX. Команды для работы с файлами и каталогами

Мартынов Николай Алексеевич НБИбд-02-21

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	11
4	Контрольные вопросы	12

List of Figures

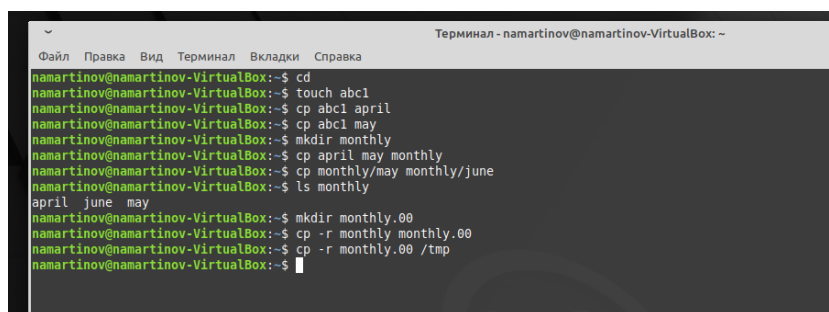
2.1	Выполнение примеров	5
2.2	Выполнение примеров	5
2.3	Выполнение примеров	6
2.4	Работа с каталогами	6
2.5	Настройка прав доступа	7
2.6	Файл /etc/passwd	7
2.7	Работа с файлами и правами доступа	8
2.8	Команда mount	8
2.9	Команда fsck	9
2.10	Команда mkfs	10
2.11	Команда kill	10

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, по проверке использования диска и обслуживанию файловой системы.

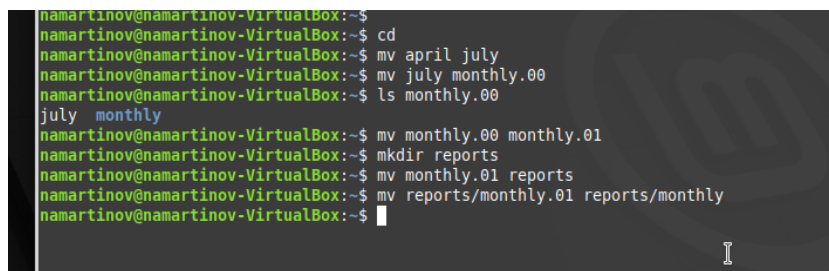
2 Выполнение лабораторной работы

1. Выполним примеры, приведённые в первой части описания лабораторной работы.



```
Терминал - namartinov@namartinov-VirtualBox: ~
Файл  Правка  Вид  Терминал  Вкладки  Справка
namartinov@namartinov-VirtualBox:~$ cd
namartinov@namartinov-VirtualBox:~$ touch abc1
namartinov@namartinov-VirtualBox:~$ cp abc1 april
namartinov@namartinov-VirtualBox:~$ cp abc1 may
namartinov@namartinov-VirtualBox:~$ mkdir monthly
namartinov@namartinov-VirtualBox:~$ cp april may monthly
namartinov@namartinov-VirtualBox:~$ cp monthly/may monthly/june
namartinov@namartinov-VirtualBox:~$ ls monthly
april  june  may
namartinov@namartinov-VirtualBox:~$ mkdir monthly.00
namartinov@namartinov-VirtualBox:~$ cp -r monthly monthly.00
namartinov@namartinov-VirtualBox:~$ cp -r monthly.00 /tmp
namartinov@namartinov-VirtualBox:~$
```

Figure 2.1: Выполнение примеров



```
namartinov@namartinov-VirtualBox:~$
namartinov@namartinov-VirtualBox:~$ cd
namartinov@namartinov-VirtualBox:~$ mv april july
namartinov@namartinov-VirtualBox:~$ mv july monthly.00
namartinov@namartinov-VirtualBox:~$ ls monthly.00
july  monthly
namartinov@namartinov-VirtualBox:~$ mv monthly.00 monthly.01
namartinov@namartinov-VirtualBox:~$ mkdir reports
namartinov@namartinov-VirtualBox:~$ mv monthly.01 reports
namartinov@namartinov-VirtualBox:~$ mv reports/monthly.01 reports/monthly
namartinov@namartinov-VirtualBox:~$
```

Figure 2.2: Выполнение примеров

```
namartinov@namartinov-VirtualBox:~$  
namartinov@namartinov-VirtualBox:~$ cd  
namartinov@namartinov-VirtualBox:~$ touch may  
namartinov@namartinov-VirtualBox:~$ ls -l may  
-rw-rw-r-- 1 namartinov namartinov 0 мая 2 19:52 may  
namartinov@namartinov-VirtualBox:~$ chmod u+x may  
namartinov@namartinov-VirtualBox:~$ ls -l may  
-rwxrw-r-- 1 namartinov namartinov 0 мая 2 19:52 may  
namartinov@namartinov-VirtualBox:~$ chmod u-x may  
namartinov@namartinov-VirtualBox:~$ ls -l may  
-rw-rw-r-- 1 namartinov namartinov 0 мая 2 19:52 may  
namartinov@namartinov-VirtualBox:~$ cd  
namartinov@namartinov-VirtualBox:~$ mkdir monthly  
mkdir: невозможно создать каталог «monthly»: Файл существует  
namartinov@namartinov-VirtualBox:~$ chmod g-r,o-r monthly  
namartinov@namartinov-VirtualBox:~$ cd  
namartinov@namartinov-VirtualBox:~$ touch abc1  
namartinov@namartinov-VirtualBox:~$ chmod g+w abc1  
namartinov@namartinov-VirtualBox:~$
```

Figure 2.3: Выполнение примеров

2.1. Скопируем файл /usr/include/sys/io.h в домашний каталог и переименуем его equipment. Такого нет, взяли другой файл.

2.2. - 2.5. В домашнем каталоге создаем директорию ski.places. и перемещаем в него файл equipment. Переименовываем файл equipment в equiplist. После этого создаем в домашнем каталоге файл abc1 и копируем его в каталог ski.places. и переименовываем в equiplist2. 2.6. - 2.7. Создаем каталог с именем equipment в каталоге ski.places. Перемещаем файлы equiplist и equiplist2 в каталог equipment. 2.8. Создаем и перемещаем каталог newdir в каталог ski.places и называем его plans.

```
namartinov@namartinov-VirtualBox:~$  
namartinov@namartinov-VirtualBox:~$ cp /usr/include/linux/sysinfo.h ~  
namartinov@namartinov-VirtualBox:~$ mv sysinfo.h equipment  
namartinov@namartinov-VirtualBox:~$ mkdir ski.places  
namartinov@namartinov-VirtualBox:~$ mv equipment ski.places/  
namartinov@namartinov-VirtualBox:~$ mv ski.places/equipment ski.places/equiplist  
namartinov@namartinov-VirtualBox:~$ touch abc1  
namartinov@namartinov-VirtualBox:~$ cp abc1 ski.places/equiplist2  
namartinov@namartinov-VirtualBox:~$ cd ski.places/  
namartinov@namartinov-VirtualBox:~/ski.places$ mkdir equipment  
namartinov@namartinov-VirtualBox:~/ski.places$ mv equiplist equipment/  
namartinov@namartinov-VirtualBox:~/ski.places$ mv equiplist2 equipment/  
namartinov@namartinov-VirtualBox:~/ski.places$ cd  
namartinov@namartinov-VirtualBox:~$ mkdir newdir  
namartinov@namartinov-VirtualBox:~$ mv newdir ski.places/  
namartinov@namartinov-VirtualBox:~$ mv ski.places/newdir/ ski.places/plans  
namartinov@namartinov-VirtualBox:~$
```

Figure 2.4: Работа с каталогами

3. Определим опции команды chmod, необходимые для того, чтобы присвоить файлам из хода работы нужные права доступа.

a) Australia (drwxr-r-)

- b) play (drwx-x-x)
- c) My_oc (-r-xr-r-)
- d) feathers (-rw-rw-r-)

```

namartinov@namartinov-VirtualBox:~$
namartinov@namartinov-VirtualBox:~$ mkdir australia play
namartinov@namartinov-VirtualBox:~$ touch my_os feathers
namartinov@namartinov-VirtualBox:~$ chmod 744 australia/
namartinov@namartinov-VirtualBox:~$ chmod 711 play/
namartinov@namartinov-VirtualBox:~$ chmod 544 my_os
namartinov@namartinov-VirtualBox:~$ chmod 664 feathers
namartinov@namartinov-VirtualBox:~$ ls -l
итого 56
-rw-rw-r-- 1 namartinov namartinov  0 мая 2 19:52 abc1
drwxr--r-- 2 namartinov namartinov 4096 мая 2 19:52 australia
-rw-rw-r-- 1 namartinov namartinov  0 мая 2 19:52 feathers
-rw-rw-r-- 1 namartinov namartinov  0 мая 2 19:52 may
drwx-wx--x 2 namartinov namartinov 4096 мая 2 19:51 monthly
-r-xr--r-- 1 namartinov namartinov  0 мая 2 19:52 my_os
drwx-x-x-x 2 namartinov namartinov 4096 мая 2 19:52 play
drwxrwxr-x 3 namartinov namartinov 4096 мая 2 19:51 reports
drwxrwxr-x 4 namartinov namartinov 4096 мая 2 19:52 ski.places
drwxrwxr-x 3 namartinov namartinov 4096 апр 23 19:18 work
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 Видео
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 Документы
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 Загрузки
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 Изображения
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 Музыка
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 Общедоступные
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 'Рабочий стол'
drwxr-xr-x 2 namartinov namartinov 4096 апр 23 18:59 Шаблоны
namartinov@namartinov-VirtualBox:~$

```

Figure 2.5: Настройка прав доступа

4.1. Просмотрим содержимое файла /etc/passwd.

```

namartinov@namartinov-VirtualBox:~$
namartinov@namartinov-VirtualBox:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mail Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,/run/systemd:/usr/sbin/nologin
messagebus:x:103:106:/nonexistent:/usr/sbin/nologin
syslog:x:104:110:/home/syslog:/usr/sbin/nologin
apt:x:105:65534:/nonexistent:/usr/sbin/nologin
ntp:x:106:111:/nonexistent:/usr/sbin/nologin
tss:x:107:112:TPM software stack,,/var/lib/tpm:/bin/false
rtkit:x:108:113:RealtimeKit,,/proc:/usr/sbin/nologin
systemd-coredump:x:109:114:systemd Core Dumper,,/run/systemd:/usr/sbin/nologin
kernoops:x:110:65534:Kernel Oops Tracking Daemon,,/usr/sbin/nologin

```

Figure 2.6: Файл /etc/passwd

4.2 - 4.12. Выполним все указанные действия по перемещению файлов и каталогов

```
Файл Правка Вид Терминал Вкладки Справка
namartinov@namartinov-VirtualBox:~$ cp feathers file.old
namartinov@namartinov-VirtualBox:~$ mv file.old play/
namartinov@namartinov-VirtualBox:~$ mkdir fun
namartinov@namartinov-VirtualBox:~$ cp -R play/ fun/
namartinov@namartinov-VirtualBox:~$ mv fun/ play/games
namartinov@namartinov-VirtualBox:~$ chmod u-r feathers
namartinov@namartinov-VirtualBox:~$ cat feathers
cat: feathers: Отказано в доступе
namartinov@namartinov-VirtualBox:~$ cp feathers feathers2
cp: невозможно открыть 'feathers' для чтения: Отказано в доступе
namartinov@namartinov-VirtualBox:~$ chmod u+r feathers
namartinov@namartinov-VirtualBox:~$ chmod u-x play/
namartinov@namartinov-VirtualBox:~$ cd play/
bash: cd: play/: Отказано в доступе
namartinov@namartinov-VirtualBox:~$ chmod +x play/
```

Figure 2.7: Работа с файлами и правами доступа

4.7. Если мы попытаемся просмотреть файл `feathers` командой `cat`, то нам будет отказано в доступе.

4.8. Если мы попытаемся скопировать файл `feathers` то у нас не получится это сделать так как мы ограничили себя в доступе для чтения.

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

```
Терминал - namartinov@namartinov-VirtualBox: ~
NAME
  mount - mount a filesystem

SYNOPSIS
  mount [-l|-h|-V]
  mount -a [-ffnrsvw] [-t fstype] [-O optlist]
  mount [-fnrsvw] [-O options] device[dir]
  mount [-fnrsvw] [-t fstype] [-O options] device dir

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 another 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

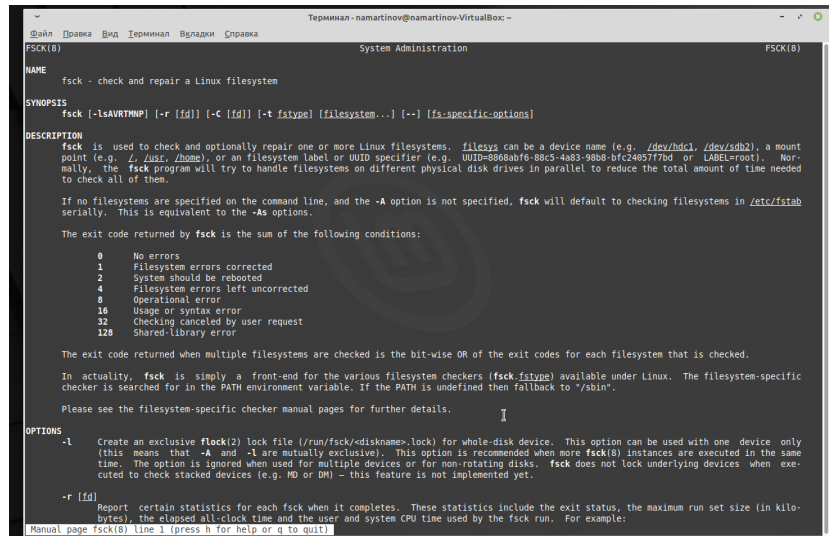
  then mount looks for a mountpoint (and if not found then for a device) in the /etc/fstab file. It's possible to use the --target or --source options to avoid ambivalent interpretation of the given argument. For example:
  mount --target /mountpoint

  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 more 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)
```

Figure 2.8: Команда `mount`

Монтирование файловой системы к общему дереву каталогов. Для размонтирования используется команда `umount`.



```
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. filesystems can be a device name (e.g. /dev/hdc1, /dev/sdb2), a mount point (e.g. /, /usr, /home), or an filesystem label or UUID specifier (e.g. UUID=88688abf6-88c5-4a83-98b8-bfc2405777bd 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 code 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 code returned when multiple filesystems are checked is the bit-wise OR of the exit codes 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.
OPTIONS
    -l      Create an exclusive flock(2) lock file (/run/fsck/<diskname>.lock) for whole-disk device. This option can be used with one device only (this means that -A and -l are mutually exclusive). This option is recommended when more fsck(8) instances are executed in the same time. The option is ignored when used for multiple devices or for non-rotating disks. fsck does not lock underlying devices when executed to check stacked devices (e.g. MD or DM) - this feature is not implemented yet.
    -r [fd] Report certain statistics for each fsck when it completes. These statistics include the exit status, the maximum run set size (in kilobytes), the elapsed all-clock time and the user and system CPU time used by the fsck run. For example:
Manual page fsck(8) line 3 (press h for help or q to quit)
```

Figure 2.9: Команда fsck

`fsck` (проверка файловой системы) – это утилита командной строки, которая позволяет выполнять проверки согласованности и интерактивное исправление в одной или нескольких файловых системах Linux. Она использует программы, специфичные для типа файловой системы, которую она проверяет. Вы можете использовать команду `fsck` для восстановления поврежденных файловых систем в ситуациях, когда система не загружается или раздел не может быть смонтирован.

```
Терминал - namartinov@namartinov-VirtualBox -
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 code 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.<type>] 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.
-V, --version
Display version information and exit. (Option -V will display version information only when it is the only parameter, otherwise it will work as --verbose.)
-h, --help
Display help text and exit.
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@bods.com)
Manual page mkfs(8) line 1 (press h for help or q to quit)
```

Figure 2.10: Команда mkfs

Буквы в mkfs значке означают “make file system” (создать файловую систему). Команда обычно используется для управления устройствами хранения в Linux. Вы можете рассматривать mkfs как инструмент командной строки для форматирования диска в определенной файловой системе.

```
Терминал - namartinov@namartinov-VirtualBox -
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.
-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.
-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)
```

Figure 2.11: Команда kill

Системный вызов kill может быть использован для посылки какого-либо сигнала какому-либо процессу или группе процесса.

3 Вывод

В ходе данной работы мы ознакомились с файловой системой Linux, её структурой, именами и содержанием каталогов. Научились совершать базовые операции с файлами, управлять правами их доступа для пользователя и групп. Ознакомились с Анализом файловой системы. А также получили базовые навыки по проверке использования диска и обслуживанию файловой системы.

4 Контрольные вопросы

1. Дайте характеристику каждой файловой системе, существующей на жёстком диске компьютера, на котором вы выполняли лабораторную работу.

Ответ: Ext2FS (расширенная файловая система номер два). Многие годы ext2 была файловой системой по умолчанию в GNU/Linux. Ext2 заменила собой Extended File System (вот откуда появилось “Second” в названии). В “новой” файловой системе были исправлены некоторые проблемы, а также убраны ограничения. Отличная стабильность, комплексные инструментальные средства для спасения удаленных файлов, очень долгое время перезагрузки после аварии, есть вероятность частичной или полной потери данных после аварии. Одним из главных недостатков “традиционных” файловых систем, подобных Ext2FS, является низкая сопротивляемость к резким системным сбоям (сбой питания или авария программного обеспечения)

Ext3 (Расширенная файловая система номер три) - является наследником файловой системы Ext2FS. Ext3 совместима с Ext2, но обладает одной новой и очень интересной особенностью – запись. Процесс сохранения объекта происходит прежде чем запись в журнал. В результате мы получаем всегда последовательную файловую систему. Это приводит к тому, что при появлении проблем, проверка и восстановление происходят очень быстро. Время, потраченное на то, чтобы проверить файловую систему таким образом, пропорционально его фактическому использованию и не больше его размера.

ReiserFS (Это тоже журналируемая файловая система подобно Ext3FS, но их внутренняя структура радикально отличается. В ReiserFS используется концепция

бинарных деревьев (binary-tree), позаимствованная из программного обеспечения баз данных.

JFS (журналируемая файловая система). JFS была разработана и использовалась IBM. Вначале JFS была закрытой системой, но недавно IBM решила открыть доступ для движения свободного программного обеспечения. Внутренняя структура JFS близка к ReiserFS. Средняя стабильность, нет комплексных инструментальных средств для спасения удаленных файлов, очень быстрая перезагрузка после аварии, очень хорошее восстановление данных после аварии.

2. Приведите общую структуру файловой системы и дайте характеристику каждой директории первого уровня этой структуры. Ответ:

- Загрузочный блок занимает первый блок файловой системы. Только корневая файловая система имеет активный загрузочный блок, хотя место для него резервируется в каждой файловой системе.
- Суперблок располагается непосредственно за загрузочным блоком и содержит самую общую информацию о ФС (размер ФС, размер области индексных дескрипторов, их число, список свободных блоков, свободные индексные дескрипторы и т. д.). Суперблок всегда находится в оперативной памяти. Различные версии ОС Unix способны поддерживать разные типы файловых систем. Поэтому у структуры суперблока могут быть варианты (сведения о свободных блоках, например, часто хранятся не как список, а как шкала бит), но суперблок всегда располагается за загрузочным блоком. При монтировании файловой системы в оперативной памяти создается копия ее суперблока. Все последующие операции по созданию и удалению файлов влекут изменения копии суперблока в оперативной памяти. Эта копия периодически записывается на магнитный диск. Обычно причиной повреждения файловой системы является отключение электропитания (или зависание ОС) в тот момент, когда система производит копирование суперблока из оперативной памяти на магнитный диск.

- Область индексных дескрипторов содержит описатели файлов (inode). С каждым файлом связан один inode, но одному inode может соответствовать несколько файлов. Binode хранится вся информация о файле, кроме его имени. Область индексных дескрипторов имеет фиксированный формат и располагается непосредственно за суперблоком. Общее число описателей и, следовательно, максимальное число файлов задается в момент создания файловой системы. Описатели нумеруются натуральными числами. Первый описатель используется ОС для описания специального файла (файла «Плохих блоков»). То есть поврежденные блоки раздела рассматриваются ОС как принадлежащие к специальному файлу и поэтому считаются «занятыми». Вторым описывает корневой каталог файловой системы.
 - В области данных расположены как обычные файлы, так и файлы каталогов (в том числе корневой каталог). Специальные файлы представлены в ФС только записями в соответствующих каталогах и индексными дескрипторами специального формата, т. е. места в области памяти не занимают.
3. Какая операция должна быть выполнена, чтобы содержимое некоторой файловой системы было доступно операционной системе? Ответ: Команда `cat` - позволяет вывести на экран содержимое любого файла, однако в таком виде эта команда практически не используется. Если файл слишком большой, то его содержимое пролистается на экране, а Вы увидите только последние строки файла. С помощью этой команды можно комбинировать и объединять копии файлов, а также создавать новые файлы. Если набрать просто в командной строке `cat` и нажать `Enter`, то можно вводить (и соответственно видеть) текст на экране. Повторное нажатие клавиши `Enter` удвоит строку и позволит начать следующую. Когда текст набран, следует одновременно нажать клавиши `Ctrl` и `d`.

4. Назовите основные причины нарушения целостности файловой системы. Как устранить повреждения файловой системы? Ответ: Некорректность файловой системы может возникать:
- В результате насильственного прерывания операций ввода-вывода, выполняемых непосредственно с диском.
 - В результате нарушения работы дискового кэша. Кэширование данных с диска предполагает, что в течение некоторого времени результаты операций ввода-вывода никак не сказываются на содержимом диска — все изменения происходят с копиями блоков диска, временно хранящихся в буферах оперативной памяти (в этих буферах оседают данные из пользовательских файлов и служебная информация файловой системы, такая как каталоги, индексные дескрипторы, списки свободных, занятых и поврежденных блоков и т. п.)
5. Как создаётся файловая система? Ответ: Общее дерево файлов и каталогов системы Linux формируется из отдельных “ветвей”, соответствующих различным физическим носителям. В UNIX нет понятия “форматирования диска” (и команды форматирования), а используется понятие “создание файловой системы”. Когда мы получаем новый носитель, например, жесткий диск, мы должны создать на нем файловую систему. То есть каждому носителю ставится в соответствие отдельная файловая система. Чтобы эту файловую систему использовать для записи в нее файлов, надо ее вначале подключить в общее дерево каталогов (“смонтировать”). Вот и получается, что можно говорить о монтировании файловых систем или о монтировании носителей (с созданными на них файловыми системами). Например, создается файловая система типа ext2fs. Создание файловой системы типа ext2fs подразумевает создание в данном разделе на диске суперблока, таблицы индексных дескрипторов и совокупности блоков данных. Делается все это все с помощью команды mkfs. В простейшем случае достаточно дать эту команду в следующем формате:

[root]# mkfs -t ext2 /dev/hda5, где /dev/hda5 надо заменить указанием на соответствующее устройство или раздел. Например, если вы хотите создать файловую систему на диске, то команда примет вид:

```
[root]# mkfs -t ext2 /dev/fd0
```

После выполнения команды `mkfs` в указанном разделе будет создана файловая система `ext2fs`. В новой файловой системе автоматически создается один каталог с именем `lost+found`. Он используется в экстренных случаях программой `fsck`, поэтому не удаляйте его. Для того, чтобы начать работать с новой файловой системой, необходимо подключить ее в общее дерево каталогов, что делается с помощью команды `mount`. В качестве параметров команде `mount` надо, как минимум, указать устройство и “точку монтирования”. Точкой монтирования называется тот каталог в уже существующем и известном системе дереве каталогов, который будет теперь служить корневым каталогом для подключаемой файловой системы. После монтирования файловой системы в каталог `/mnt/disk2` прежнее содержимое этого каталога станет для вас недоступно до тех пор, пока вы не размонтируете вновь подключенную файловую систему. Прежнее содержимое не уничтожается, а просто становится временно недоступным. Поэтому в качестве точек монтирования лучше использовать пустые каталоги (заранее заготовленные).

6. Дайте характеристику командам, которые позволяют просмотреть текстовые файлы. Ответ: Для просмотра небольших файлов удобно пользоваться командой `cat`. Формат команды: `cat имя-файла`

Для просмотра больших файлов используйте команду `less` — она позволяет осуществлять постраничный просмотр файлов (длина страницы соответствует размеру экрана). Формат команды: `less имя-файла`

Для управления процессом просмотра можно использовать следующие управляющие клавиши: - `Space` — переход на следующую страницу, - `ENTER` — сдвиг вперед на одну строку, - `b` — возврат на предыдущую страницу, - `h` — обращение

за подсказкой, - q — выход в режим командной строки.

Для просмотра начала файла можно воспользоваться командой `head`. По умолчанию она выводит первые 10 строк файла. Формат команды: `head [-n] имя-файла`, где `n` — количество выводимых строк.

Команда `tail` выводит несколько (по умолчанию 10) последних строк файла. Формат команды: `tail [-n] имя-файла`, где `n` — количество выводимых строк.

7. Приведите основные возможности команды `cp` в Linux. Ответ: Копирование отдельных файлов Для копирования файла следует использовать утилиту `cp` с аргументами, представленными путями к исходному и целевому файлам.

Копирование файлов в другую директорию В том случае, если в качестве пути к целевому файлу используется путь к директории, исходные файлы будут скопированы в эту целевую директорию.

Команда `cp -r` Для копирования директорий целиком следует использовать команду `cp -r` (параметр `-r` позволяет осуществлять рекурсивное копирование всех файлов из всех поддиректорий).

Копирование множества файлов в директорию Вы также можете использовать утилиту `cp` для копирования множества файлов в одну директорию. В этом случае последний аргумент (аргумент, указывающий на цель) должен быть представлен путем к директории.

Команда `cp -i` Для предотвращения перезаписи существующих файлов в ходе использования утилиты `cp` следует использовать параметр `-i` (для активации интерактивного режима копирования).

8. Назовите и дайте характеристику командам перемещения и переименования файлов и каталогов. Ответ: Команды `mv` и `mkdir` предназначены для перемещения и переименования файлов и каталогов. Формат команды `mv`: `mv [-опции] старый_файл новый_файл` Примеры:

- Переименование файлов в текущем каталоге. Изменить название файла `april` на `july` в домашнем каталоге: `cd mv april july`

- Перемещение файлов в другой каталог. Переместить файл july в каталог monthly.00: `mv july monthly.00` `ls monthly.00` Результат: april july june may. Если необходим запрос подтверждения о перезаписи файла, то нужно использовать опцию `i`.
- Переименование каталогов в текущем каталоге. Переименовать каталог monthly.00 в monthly.01 `mv monthly.00 monthly.01`
- Перемещение каталога в другой каталог. Переместить каталог monthly.01 в каталог reports: `mkdir reports` `mv monthly.01 reports`
- Переименование каталога, не являющегося текущим. Переименовать каталог reports/monthly.01 в reports/monthly: `mv reports/monthly.01 reports/monthly`

9. Что такое права доступа? Как они могут быть изменены? Ответ: Права доступа — совокупность правил, регламентирующих порядок и условия доступа субъекта к объектам информационной системы (информации, её носителям, процессам и другим ресурсам). Права доступа к файлу или каталогу можно изменить, воспользовавшись командой `chmod`. Сделать это может владелец файла (или каталога) или пользователь с правами администратора. Формат команды: `chmod режим имя_файла` Режим (в формате команды) имеет следующие компоненты структуры и способ записи: `=` установить право - лишить права `+` дать право `r` чтение `w` запись `x` выполнение `u` (user) владелец файла `g` (group) группа, к которой принадлежит владелец файла `o` (others) все остальные В работе с правами доступа можно использовать их цифровую запись (восьмеричное значение) вместо символьной