

Отчет по лабораторной работе №5

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

Легиньких Галина Андреевна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Вывод	16
5	Контрольные вопросы	17

Список иллюстраций

3.1	Рис.1	7
3.2	Рис.2	8
3.3	Рис.3	8
3.4	Рис.4	9
3.5	Рис.5	10
3.6	Рис.6	10
3.7	Рис.7	11
3.8	Рис.8	12
3.9	Рис.9	13

Список таблиц

1 Цель работы

Ознакомление с файловой системой Linux, её структурой, именами и содержанием каталога

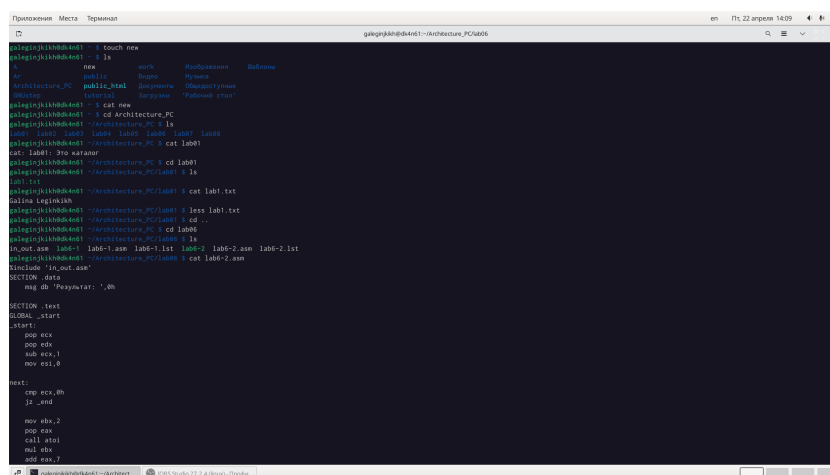
2 Теоретическое введение

По ходу лабораторной работы.

3 Выполнение лабораторной работы

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

- Для создания текстового файла использовала команду touch.
- Для просмотра файлов небольшого размера использовала команду cat.
- Для просмотра файлов постранично использовала команду less.
- Команда head вывела по умолчанию первые 10 строк файла.
- Команда tail вывела умолчанию 10 последних строк файла. (рис. 3.1)



```
galeg@kali:~/Architectur_PC$ touch new
galeg@kali:~/Architectur_PC$ ls
.          new          work             root@kali:~/Architectur_PC$
..         public_html  work             root@kali:~/Architectur_PC$
Architectur_PC public_html  work             root@kali:~/Architectur_PC$
00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
galeg@kali:~/Architectur_PC$ cat new
galeg@kali:~/Architectur_PC$ cd Architectur_PC
galeg@kali:~/Architectur_PC$ ls
.          new          work             root@kali:~/Architectur_PC$
..         public_html  work             root@kali:~/Architectur_PC$
Architectur_PC public_html  work             root@kali:~/Architectur_PC$
00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
galeg@kali:~/Architectur_PC$ cat lab01
cat: lab01: No such file or directory
galeg@kali:~/Architectur_PC$ cd lab01
galeg@kali:~/Architectur_PC/lab01$ ls
.          new          work             root@kali:~/Architectur_PC/lab01$
..         public_html  work             root@kali:~/Architectur_PC/lab01$
Architectur_PC public_html  work             root@kali:~/Architectur_PC/lab01$
00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
galeg@kali:~/Architectur_PC/lab01$ cat lab1.txt
galeg@kali:~/Architectur_PC/lab01$ head lab1.txt
galeg@kali:~/Architectur_PC/lab01$ cd -
galeg@kali:~/Architectur_PC$ cd lab00
galeg@kali:~/Architectur_PC$ ls
.          new          work             root@kali:~/Architectur_PC$
..         public_html  work             root@kali:~/Architectur_PC$
Architectur_PC public_html  work             root@kali:~/Architectur_PC$
00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
galeg@kali:~/Architectur_PC$ cat lab00-2.asm
include "in_out.asm"
SECTION .data
    msg db "Priznayat' ",0h

SECTION .text
GLOBAL start
start:
    pop ecx
    pop edx
    mov ecx,1
    mov esi,0

next:
    cmp ecx,0h
    jz _end

    mov ebx,2
    pop eax
    call atoi
    mov ebx,2
    add esi,2
    jmp next
```

Рис. 3.1: Рис.1

Рис.1 Команда touch и cat

```
add eax, 7
add esi, eax
loop next

_end:
mov eax, msg
call sprint
mov eax, esi
call iprintf

call quitgaleginkikh@dk4n61 ~/Architecture_PC/lab06 $ less lab6-2.asm
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ head lab6-2.asm
head: невозможно открыть 'lab6-2.asm' для чтения: Нет такого файла или каталога
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ head lab6-2.asm
#include "in_out.asm"
SECTION .data
msg db "Результат: ", 0h

SECTION .text
GLOBAL _start
_start:
pop ecx
pop edx
sub ecx, 1
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ head [-5] lab6-2.asm
head: невозможно открыть '[-5]' для чтения: Нет такого файла или каталога
=> lab6-2.asm ccs
#include "in_out.asm"
SECTION .data
msg db "Результат: ", 0h

SECTION .text
GLOBAL _start
_start:
pop ecx
pop edx
sub ecx, 1
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ head -5 lab6-2.asm
#include "in_out.asm"
SECTION .data
msg db "Результат: ", 0h

SECTION .text
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ tail
```

Рис. 3.2: Рис.2

Рис.2 Команда less и head

```
include "in_out.asm"
SECTION .data
msg db "Результат: ", 0h

SECTION .text
GLOBAL _start
_start:
pop ecx
pop edx
sub ecx, 1
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ head [-5] lab6-2.asm
head: невозможно открыть '[-5]' для чтения: Нет такого файла или каталога
=> lab6-2.asm ccs
#include "in_out.asm"
SECTION .data
msg db "Результат: ", 0h

SECTION .text
GLOBAL _start
_start:
pop ecx
pop edx
sub ecx, 1
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ head -5 lab6-2.asm
#include "in_out.asm"
SECTION .data
msg db "Результат: ", 0h

SECTION .text
galeginkikh@dk4n61 ~/Architecture_PC/lab06 $ tail lab6-2.asm
add esi, eax
loop next

_end:
mov eax, msg
call sprint
mov eax, esi
call iprintf

call quitgaleginkikh@dk4n61 ~/Architecture_PC/lab06 $ tail -4 lab6-2.asm
mov eax, esi
call iprintf

call quitgaleginkikh@dk4n61 ~/Architecture_PC/lab06 $
```

Рис. 3.3: Рис.3

Рис.3 Команда tail

2. Скопировала файл `/usr/include/sys/io.h` в домашний каталог и назовите его `equipment`.

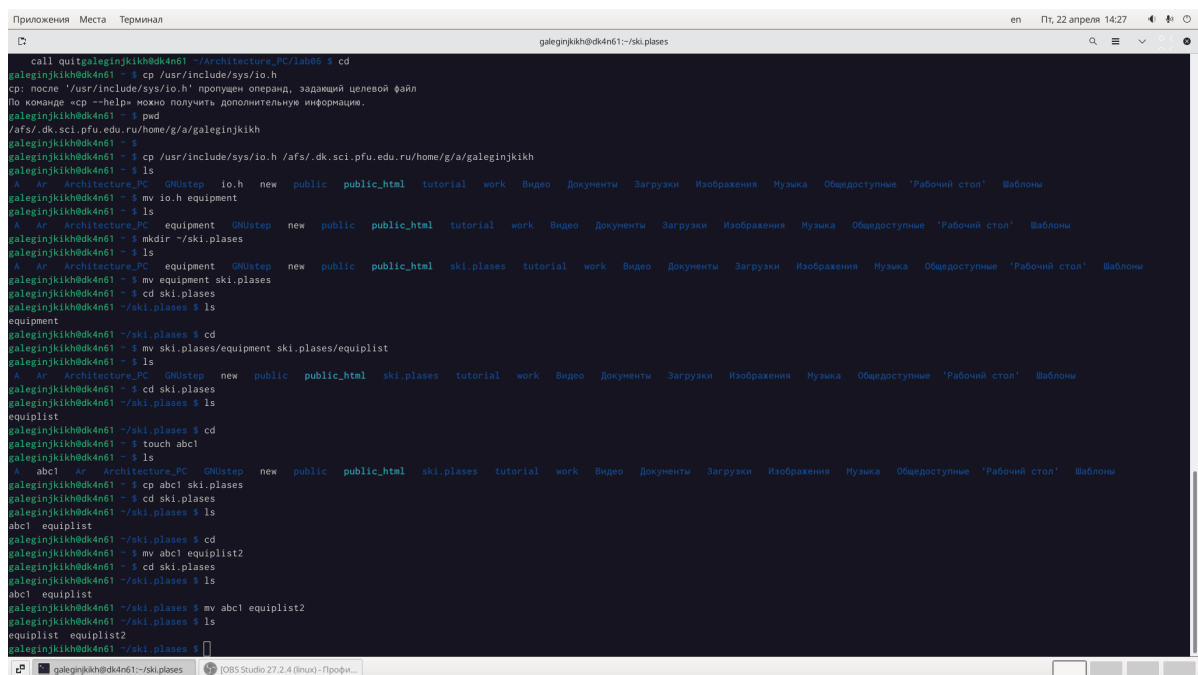


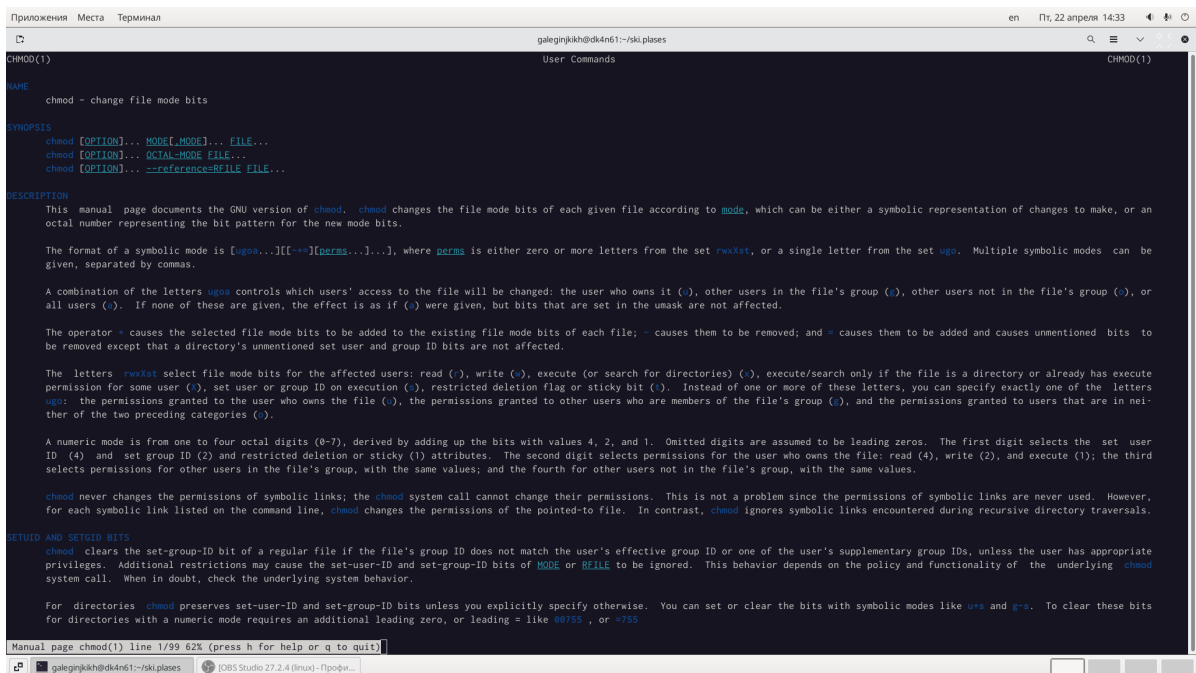
Рис.4 Создание файлов и перемещение

```
galeginkikh@dk4n61:~/ski.places $ mv equiplist equiplist2 equipment
galeginkikh@dk4n61:~/ski.places $ ls
equiplist
galeginkikh@dk4n61:~/ski.places $ cd equipment
galeginkikh@dk4n61:~/ski.places/equipment $ ls
equiplist equiplist2
galeginkikh@dk4n61:~/ski.places/equipment $ cd
galeginkikh@dk4n61:~/ $ mkdir newdir
galeginkikh@dk4n61:~/ $ cp newdir ski.places
cp: не указан -r; копируется karanor 'newdir'
galeginkikh@dk4n61:~/ $ cp -r newdir ski.places
galeginkikh@dk4n61:~/ $ ls
Architecture_PC  GNUstep  newdir  public_html  tutorial  Видео  Загрузки  Музыка  'Рабочий стол'
Ar  equiplist2  new  public  ski_places  work  Документы  Изображения  Общедоступные  Шаблоны
galeginkikh@dk4n61:~/ $ cd ski.places
galeginkikh@dk4n61:~/ski.places $ ls
equipment  newdir
galeginkikh@dk4n61:~/ski.places $ mv newdir plans
galeginkikh@dk4n61:~/ski.places $ ls
equipment  plans
```

Рис. 3.5: Рис.5

Рис.5 Создание и перемещение файлов

3. Определила опции команды `chmod`, необходимые для того, чтобы присвоить перечисленным ниже файлам выделенные права доступа, считая, что в начале таких прав нет. (Рис.6)



```
Приложения Места Терминал
galeginkikh@dk4n61:~/ski.places
chmod(1)
NAME
  chmod - change file mode bits

SYNOPSIS
  chmod [OPTION]... MODE[,MODE]... FILE...
  chmod [OPTION]... OCTAL-MODE FILE...
  chmod [OPTION]... --reference=FILE FILE...

DESCRIPTION
  This manual page documents the GNU version of chmod. chmod changes the file mode bits of each given file according to mode, which can be either a symbolic representation of changes to make, or an octal number representing the bit pattern for the new mode bits.

  The format of a symbolic mode is [ugoa...][[+-=]][perms...], where perms is either zero or more letters from the set rwXxt, or a single letter from the set ugo. Multiple symbolic modes can be given, separated by commas.

  A combination of the letters ugo controls which users' access to the file will be changed: the user who owns it (u), other users in the file's group (g), other users not in the file's group (o), or all users (a). If none of these are given, the effect is as if (a) were given, but bits that are set in the umask are not affected.

  The operator + causes the selected file mode bits to be added to the existing file mode bits of each file; - causes them to be removed; and = causes them to be added and causes unmentioned bits to be removed except that a directory's unmentioned set user and group ID bits are not affected.

  The letters rwXxt select file mode bits for the affected users: read (r), write (w), execute (or search for directories) (X), execute/search only if the file is a directory or already has execute permission for some user (t), set user or group ID on execution (x), restricted deletion flag or sticky bit (T). Instead of one or more of these letters, you can specify exactly one of the letters ugo: the permissions granted to the user who owns the file (u), the permissions granted to other users who are members of the file's group (g), and the permissions granted to users that are in neither of the two preceding categories (o).

  A numeric mode is from one to four octal digits (0-7), derived by adding up the bits with values 4, 2, and 1. Omitted digits are assumed to be leading zeros. The first digit selects the set user ID (4) and set group ID (2) and restricted deletion or sticky (1) attributes. The second digit selects permissions for the user who owns the file: read (4), write (2), and execute (1); the third selects permissions for other users in the file's group, with the same values; and the fourth for other users not in the file's group, with the same values.

  chmod never changes the permissions of symbolic links; the chmod system call cannot change their permissions. This is not a problem since the permissions of symbolic links are never used. However, for each symbolic link listed on the command line, chmod changes the permissions of the pointed-to file. In contrast, chmod ignores symbolic links encountered during recursive directory traversals.

SETUID AND SETGID BITS
  chmod clears the set-group-ID bit of a regular file if the file's group ID does not match the user's effective group ID or one of the user's supplementary group IDs, unless the user has appropriate privileges. Additional restrictions may cause the set-user-ID and set-group-ID bits of MODE or REFERENCE to be ignored. This behavior depends on the policy and functionality of the underlying chmod system call. When in doubt, check the underlying system behavior.

  For directories chmod preserves set-user-ID and set-group-ID bits unless you explicitly specify otherwise. You can set or clear the bits with symbolic modes like u+s and g-s. To clear these bits for directories with a numeric mode requires an additional leading zero, or leading = like 00755, or =755.

Manual page chmod(1) line 1/99 62% (press h for help or q to quit)
```

Рис. 3.6: Рис.6

Рис.6 Опции команды chmod

Создала файлы и каталоги которые необходимы. И установила нужные права. (Рис.7)

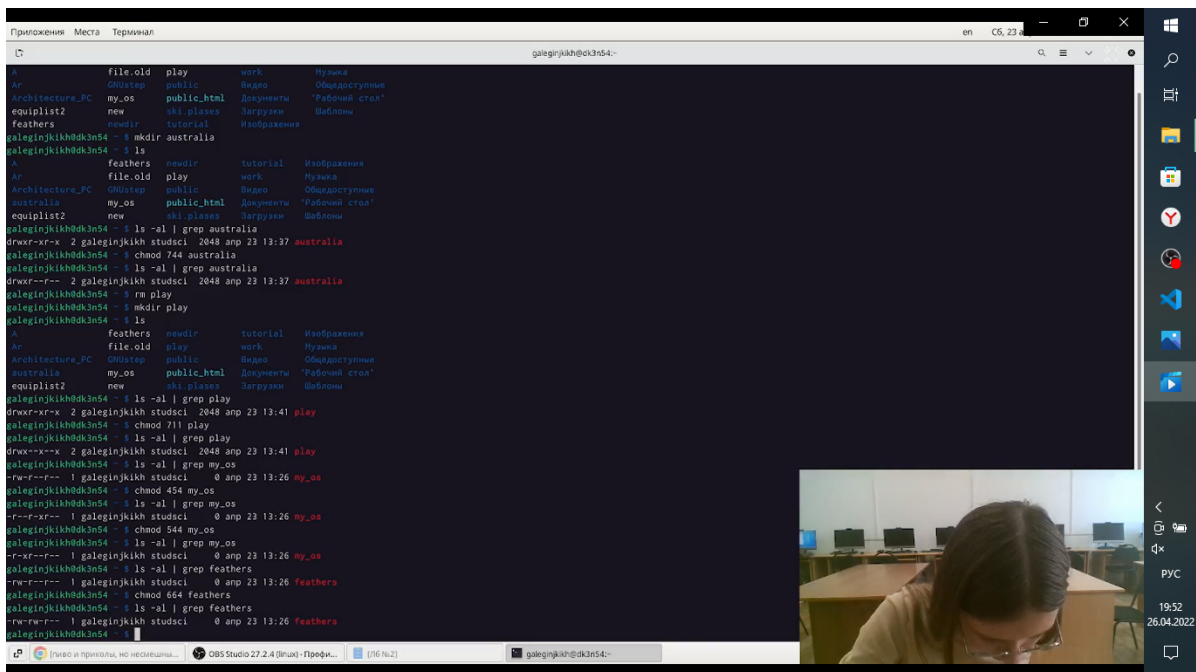


Рис. 3.7: Рис.7

Рис.7 Права каталогов и файлов

4. Просмотрела содержимое файла /etc/passwd.

Скопировала файл ~/feathers в файл ~/file.old.

Переместила файл ~/file.old в каталог ~/play.

Скопировала каталог ~/play в каталог ~/fun.

Переместила каталог ~/fun в каталог ~/play и назовите его games. (Рис.8)

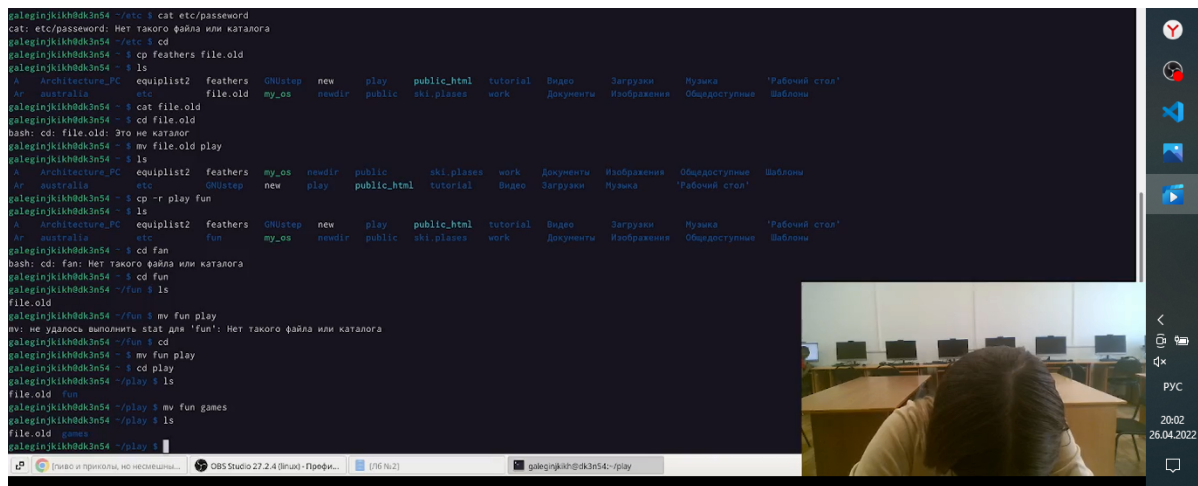


Рис. 3.8: Рис.8

Рис.8 Работа с файлами

Лишила владельца файла ~/feathers права на чтение.

Что произойдёт, если вы попытаетесь просмотреть файл ~/feathers командой cat?

Ответ: Отказано в доступе

Что произойдёт, если вы попытаетесь скопировать файл ~/feathers?

Ответ: Отказано в доступе

Дала владельцу файла ~/feathers право на чтение.

Лишила владельца каталога ~/play права на выполнение.

Перешла в каталог ~/play.

Дала владельцу каталога ~/play право на выполнение. (Рис.9)

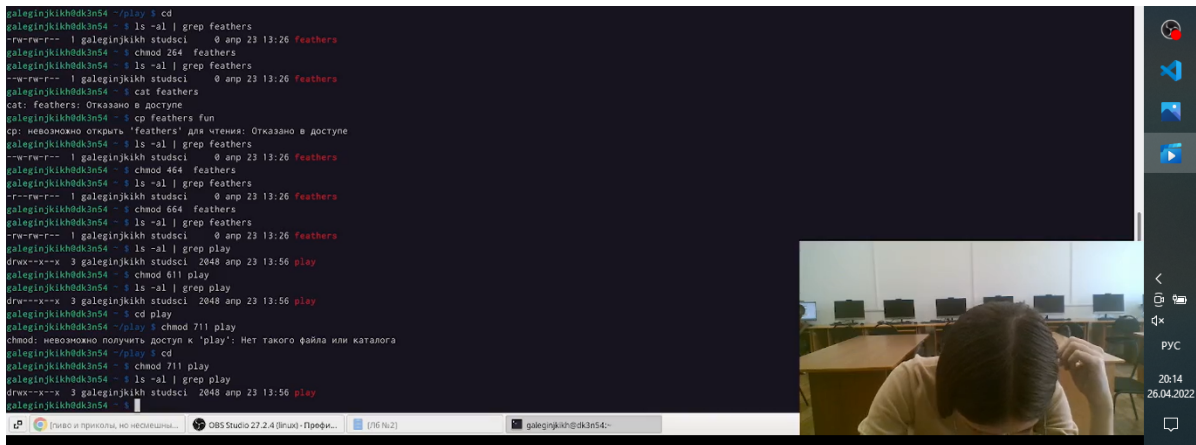


Рис. 3.9: Рис.9

Рис.9 Работа с файлами и их правами

5. Прочитала man по командам mount, fsck, mkfs, kill и кратко их охарактеризовала, приведя примеры.

- Опции mount:

- V - вывести версию утилиты;
- h - вывести справку;
- v - подробный режим;
- a, -all - примонтировать все устройства, описанные в fstab;
- F, -fork - создавать отдельный экземпляр mount для каждого отдельного раздела;
- f, -fake - не выполнять никаких действий, а только посмотреть что собирается делать утилита;
- n, -no-mtab - не записывать данные о монтировании в /etc/mtab;
- l, -show-labels - добавить метку диска к точке монтирования;
- s - использовать только абсолютные пути;
- r, -read-only - монтировать раздел только для чтения;

- w, -rw - монтировать для чтения и записи;
- L, -label - монтировать раздел по метке;
- U, -uuid - монтировать раздел по UUID;
- T, -fstab - использовать альтернативный fstab;
- B, -bind - монтировать локальную папку;
- R, -rbind - перемонтировать локальную папку.

- Оpciones fsck

-l - не выполнять другой экземпляр fsck для этого жесткого диска, пока текущий не завершит работу. Для SSD параметр игнорируется;

-t - задать типы файловых систем, которые нужно проверить. Необязательно указывать устройство, можно проверить несколько разделов одной командой, просто указав нужный тип файловой системы. Это может быть сама файловая система, например, ext4 или ее опции в формате opts=ro. Утилита просматривает все файловые системы, подключенные в fstab. Если задать еще и раздел то к нему будет применена проверка именно указанного типа, без автоопределения;

-A - проверить все файловые системы из /etc/fstab. Вот тут применяются параметры проверки файловых систем, указанные в /etc/fstab, в том числе и приоритетность. В первую очередь проверяется корень. Обычно используется при старте системы;

- C - показать прогресс проверки файловой системы;
- M - не проверять, если файловая система смонтирована;
- N - ничего не выполнять, показать, что проверка завершена успешно;
- R - не проверять корневую файловую систему;
- T - не показывать информацию об утилите;
- V - максимально подробный вывод.

- Оpciones mkfs

-c - проверить устройство на наличие битых секторов

- b - размер блока файловой системы
- j - использовать журналирование для ext3
- L - задать метку раздела
- v - показать подробную информацию о процессе работы
- V - версия программы

- Оpción kill

-Сигнал или -s Сигнал или -signal Сигнал

Задаёт сигнал, который будет послан процессу. Сигнал может задаваться числом или названием.

-l или -l Сигнал или -list Сигнал Вывести список всех сигналов.

Если задано значение Сигнал, то вывод зависит от того, чему равно заданное значение Сигнал: - числовой номер сигнала — в таком случае будет выведено название сигнала; - название сигнала — в таком случае будет - выведено числовое значение сигнала.

-L или -table

Вывести список сигналов в табличном виде. Выводится числовое значение и название каждого сигнала.

4 Вывод

Ознакомилась с файловой системой Linux, её 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.

5 Контрольные вопросы

1. Дайте характеристику каждой файловой системе, существующей на жёстком диске компьютера, на котором вы выполняли лабораторную работу.
 - Ext2, Ext3, Ext4 или Extended Filesystem - это стандартная файловая система для Linux. Она была разработана еще для Minix. Она самая стабильная из всех существующих, кодовая база изменяется очень редко и эта файловая система содержит больше всего функций. Версия ext2 была разработана уже именно для Linux и получила много улучшений. В 2001 году вышла ext3, которая добавила еще больше стабильности благодаря использованию журналирования. В 2006 была выпущена версия ext4, которая используется во всех дистрибутивах Linux до сегодняшнего дня. В ней было внесено много улучшений, в том числе увеличен максимальный размер раздела до одного экзабайта.
 - JFS или Journaled File System была разработана в IBM для AIX UNIX и использовалась в качестве альтернативы для файловых систем ext. Сейчас она используется там, где необходима высокая стабильность и минимальное потребление ресурсов. При разработке файловой системы ставилась цель создать максимально эффективную файловую систему для многопроцессорных компьютеров. Также как и ext, это журналируемая файловая система, но в журнале хранятся только метаданные, что может привести к использованию старых версий файлов после сбоев.
 - ReiserFS - была разработана намного позже, в качестве альтернативы ext3 с

улучшенной производительностью и расширенными возможностями. Она была разработана под руководством Ганса Райзера и поддерживает только Linux. Из особенностей можно отметить динамический размер блока, что позволяет упаковывать несколько небольших файлов в один блок, что предотвращает фрагментацию и улучшает работу с небольшими файлами. Еще одно преимущество - в возможности изменять размеры разделов на лету. Но минус в некоторой нестабильности и риске потери данных при отключении энергии. Раньше ReiserFS применялась по умолчанию в SUSE Linux, но сейчас разработчики перешли на Btrfs.

- XFS - это высокопроизводительная файловая система, разработанная в Silicon Graphics для собственной операционной системы еще в 2001 году. Она изначально была рассчитана на файлы большого размера, и поддерживала диски до 2 Терабайт. Из преимуществ файловой системы можно отметить высокую скорость работы с большими файлами, отложенное выделение места, увеличение разделов на лету и незначительный размер служебной информации.
- XFS - журналируемая файловая система, однако в отличие от ext, в журнал записываются только изменения метаданных. Она используется по умолчанию в дистрибутивах на основе Red Hat. Из недостатков - это невозможность уменьшения размера, сложность восстановления данных и риск потери файлов при записи, если будет неожиданное отключение питания, поскольку большинство данных находится в памяти.
- Btrfs или B-Tree File System - это совершенно новая файловая система, которая сосредоточена на отказоустойчивости, легкости администрирования и восстановления данных. Файловая система объединяет в себе очень много новых интересных возможностей, таких как размещение на нескольких разделах, поддержка подтомов, изменение размера на лету, создание мгновенных снимков, а также высокая производительность. Но многими поль-

зователями файловая система Btrfs считается нестабильной. Тем не менее, она уже используется как файловая система по умолчанию в OpenSUSE и SUSE Linux.

2. Приведите общую структуру файловой системы и дайте характеристику каждой директории первого уровня этой структуры. / — root каталог. Содержит в себе всю иерархию системы;

/bin — здесь находятся двоичные исполняемые файлы. Основные общие команды, хранящиеся отдельно от других программ в системе (прим.: pwd, ls, cat, ps);

/boot — тут расположены файлы, используемые для загрузки системы (образ initrd, ядро vmlinuz);

/dev — в данной директории располагаются файлы устройств (драйверов). С помощью этих файлов можно взаимодействовать с устройствами. К примеру, если это жесткий диск, можно подключить его к файловой системе. В файл принтера же можно написать напрямую и отправить задание на печать;

/etc — в этой директории находятся файлы конфигураций программ. Эти файлы позволяют настраивать системы, сервисы, скрипты системных демонов;

/home — каталог, аналогичный каталогу Users в Windows. Содержит домашние каталоги учетных записей пользователей (кроме root). При создании нового пользователя здесь создается одноименный каталог с аналогичным именем и хранит личные файлы этого пользователя;

/lib — содержит системные библиотеки, с которыми работают программы и модули ядра;

/lost+found — содержит файлы, восстановленные после сбоя работы системы. Система проведет проверку после сбоя и найденные файлы можно будет посмотреть в данном каталоге;

/media — точка монтирования внешних носителей. Например, когда вы вставляете диск в дисковод, он будет автоматически смонтирован в директорию /media/cdrom;

/mnt — точка временного монтирования. Файловые системы подключаемых устройств обычно монтируются в этот каталог для временного использования;

/opt — тут расположены дополнительные (необязательные) приложения. Такие программы обычно не подчиняются принятой иерархии и хранят свои файлы в одном подкаталоге (бинарные, библиотеки, конфигурации);

/proc — содержит файлы, хранящие информацию о запущенных процессах и о состоянии ядра ОС;

/root — директория, которая содержит файлы и личные настройки суперпользователя;

/run — содержит файлы состояния приложений. Например, PID-файлы или UNIX-сокеты;

/sbin — аналогично /bin содержит бинарные файлы. Утилиты нужны для настройки и администрирования системы суперпользователем;

/srv — содержит файлы сервисов, предоставляемых сервером (прим. FTP или Apache HTTP);

/sys — содержит данные непосредственно о системе. Тут можно узнать информацию о ядре, драйверах и устройствах;

/tmp — содержит временные файлы. Данные файлы доступны всем пользователям на чтение и запись. Стоит отметить, что данный каталог очищается при перезагрузке;

/usr — содержит пользовательские приложения и утилиты второго уровня, используемые пользователями, а не системой. Содержимое доступно только для чтения (кроме root). Каталог имеет вторичную иерархию и похож на корневой;

/var — содержит переменные файлы. Имеет подкаталоги, отвечающие за отдельные переменные. Например, логи будут храниться в /var/log, кэш в /var/cache, очереди заданий в /var/spool/ и так далее.

3. Какая операция должна быть выполнена, чтобы содержимое некоторой файловой системы было доступно операционной системе?

Монтирование тома.

4. Назовите основные причины нарушения целостности файловой системы.

Как устранить повреждения файловой системы? Отсутствие синхронизации между образом файловой системы в памяти и ее данными на диске в случае аварийного останова может привести к появлению следующих ошибок:

- Один блок адресуется несколькими inode (принадлежит нескольким файлам).
- Блок помечен как свободный, но в то же время занят (на него ссылается onode).
- Блок помечен как занятый, но в то же время свободен (ни один inode на него не ссылается).
- Неправильное число ссылок в inode (недостаток или избыток ссылающихся записей в каталогах).
- Несовпадение между размером файла и суммарным размером адресуемых inode блоков.
- Недопустимые адресуемые блоки (например, расположенные за пределами файловой системы).
- “Потерянные” файлы (правильные inode, на которые не ссылаются записи каталогов).
- Недопустимые или неразмещенные номера inode в записях каталогов.

5. Как создаётся файловая система?

mkfs - позволяет создать файловую систему Linux.

6. Дайте характеристику командам для просмотра текстовых файлов.

Cat - выводит содержимое файла на стандартное устройство вывода

7. Приведите основные возможности команды `cp` в Linux.

`Ср` – копирует или перемещает директорию, файлы.

8. Приведите основные возможности команды `mv` в Linux.

`Mv` - переименовать или переместить файл или директорию

9. Что такое права доступа? Как они могут быть изменены?

Права доступа к файлу или каталогу можно изменить, воспользовавшись командой `chmod`.

Сделать это может владелец файла (или каталога) или пользователь с правами администратора.