**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Операционные системы и среды»**

Тема: **Файловые системы UNIX-подобных ОС**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1384 |  | Усачева Д.В. |
| Преподаватель |  | Душутина Е. В. |

Санкт-Петербург

2023

## Цель работы.

## Проанализировать функциональное назначение структурных элементов

## дерева ФС. Определить размещение корневого каталога (корневой ФС).

## Задание.

1. Ознакомиться с типами файлов исследуемой ФС. Применяя утилиту ls, отфильтровать по одному примеру каждого типа файла используемой вами ФС. Комбинируя различные ключи утилиты рекурсивно просканировать все дерево, анализируя крайнюю левую позицию **выходной информации,** полученной посредством ls –l. Результат записать в выходной файл с указанием полного пути каждого примера. Выполнить задание сначала в консоли построчно, выбирая необходимые сочетания ключей (в командной строке), а затем оформить как скрипт с задаваемым в командной строке именем файла как параметр

Файлы системы могут быть следующих типов:

- Обычный файл

b Специальный файл блочного устройства

c Файл символьного устройства

d Директория

l Символьная ссылка

p FIFO

s Сокет

2. Получить все жесткие ссылки на заданный файл, находящиеся в разных каталогах пользовательского пространства (разными способами, не применяя утилиты file и find). Использовать конвейеризацию и фильтрацию. Оформить в виде скрипта.

3. Проанализировать все возможные способы формирования символьных ссылок (ln, link,cp и т.д.), продемонстрировать их экспериментально. Предложить скрипт, подсчитывающий и перечисляющий все полноименные символьные ссылки на файл, размещаемые в разных местах файлового дерева.

4. Получить все символьные ссылки на заданный в качестве входногопараметра файл, не используя file (разными способами, не применяя утилитуfile).

5. Изучить утилиту ***find***, используя ее ключи получить расширенную информацию о всех типах файлов. Создать *примеры вложенных команд*.

6. Проанализировать *содержимое заголовка файла*, а также файла-каталога с помощью утилит ***od*** и ***\*dump***. Если доступ к файлу-каталогу возможен (для отдельных модификаций POSIX-совместимых ОС), проанализировать изменение его содержимого при различных операциях над элементами, входящими в его состав (файлами и подкаталогами).

7. Определить максимальное количество записей в каталоге. Изменить размер каталога, варьируя количество записей (для этого создать программу, порождающую новые файлы и каталоги, а затем удаляющую их, предусмотрев промежуточный и конечный вывод информации о размере подопытного каталога).

8. Ознакомиться с содержимым /etc/passwd, /etc/shadow, с утилитой /usr/bin/passwd, проанализировать права доступа к этим файлам.

9. Исследовать права владения и доступа, а также их сочетаемость

9.1. Привести примеры применения утилит chmod, chown к специально созданному для этих целей отдельному каталогу с файлами.

9.2. Расширить права исполнения экспериментального файла с помощью флага SUID.

9.3. Экспериментально установить, как формируются итоговые права на использование файла, если права пользователя и группы, в которую он входит, различны.

9.4. Сопоставить возможности исполнения наиболее часто используемых операций, варьируя правами доступа к файлу и каталогу.

10. Разработать «программу-шлюз» для доступа к файлу другого пользователя при отсутствии прав на чтение информации из этого файла. Провести эксперименты для случаев, когда пользователи принадлежат одной и разным группам. Сравнить результаты. Для выполнения задания применить подход, аналогичный для обеспечения функционирования утилиты /usr/bin/passwd (манипуляции с правами доступа, флагом SUID, а также размещением файлов).

11. Применяя утилиту df и аналогичные ей по функциональности утилиты, а также информационные файлы типа fstab, получить информацию о файловых системах, возможных для монтирования, а также установленных на компьютере реально.

11.1. Привести информацию об исследованных утилитах и информационных файлах с анализом их содержимого и форматов.

11.2. Привести образ диска с точки зрения состава и размещения всех ФС на испытуемом компьютере, а также образ полного дерева ФС, включая присоединенные ФС съемных и несъемных носителей. Проанализировать и указать формат таблицы монтирования.

11.3. Привести «максимально возможное» дерево ФС, проанализировать, где это указывается

12. Проанализировать и пояснить принцип работы утилиты file.

12.1. Привести алгоритм её функционирования на основе информационной базы, размещение и полное имя которой указывается в описании утилиты в технической документации ОС (как правило, /usr/share/file/magic.\*), а также содержимого заголовка файла, к которому применяется утилита. Определить, где находятся магические числа и иные характеристики, идентифицирующие тип файла, применительно к исполняемым файлам, а также файлам других типов.

12.2. Утилиту file выполнить с разными ключами.

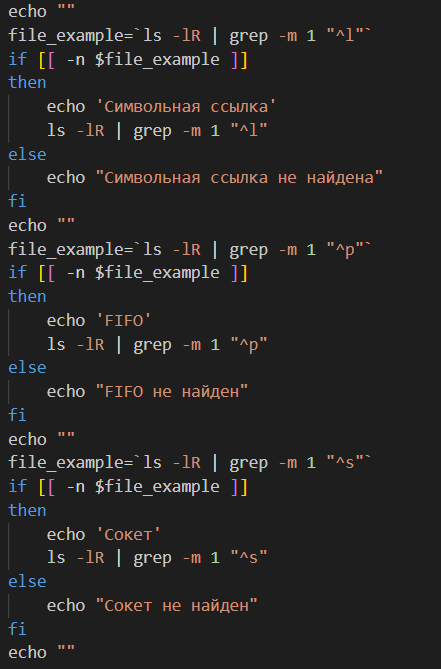
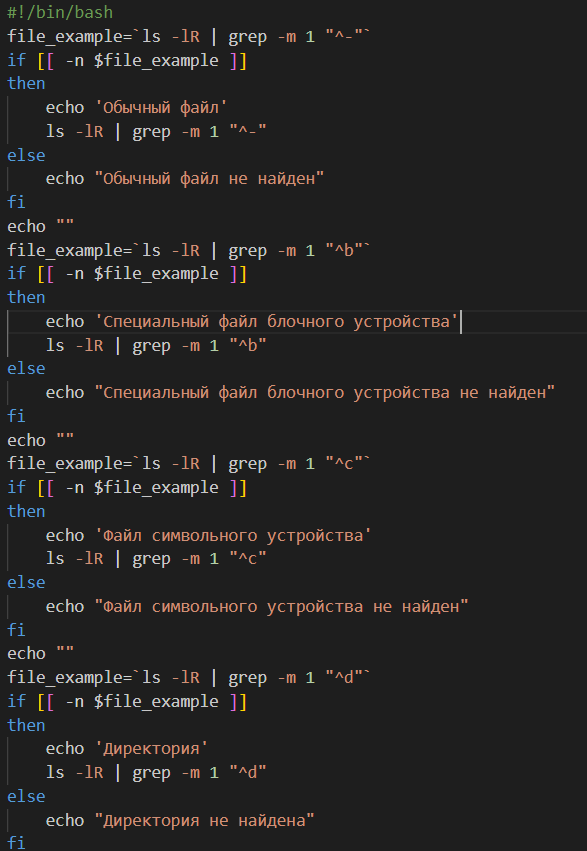
12.3. Привести экспериментальную попытку с добавлением в базу собственного типа файла и его дальнейшей идентификацией. Описать эксперимент и привести последовательность действий для расширения функциональности утилиты file и возможности встраивания дополнительного типа файла в ФС (согласовать содержимое информационной базы и заголовка файла нового типа.предусмотрев промежуточный и конечный вывод информации о размере подопытного каталога).

## Выполнение работы

Задание 1.

С помощью команды ls -lR можно отобразить содержимое директории и всех ее подкаталогов в рекурсивном виде в формате длинного списка. Вывод команды будет содержать информацию о правах доступа к файлам и каталогам, количестве жестких ссылок на каждый файл, имя пользователя и группы, размер файла, время последней модификации и т. д.

Для того, чтоб отфильтровать файлы по их типу, нужно применить поиск по шаблону с помощью grep с ключом -m: ls –lR | grep -m 1 «^“тип файла“». Таким образом, будет указан 1 файл данного типа.



Рисунки 1.1 и 1.2 – Скрипт для выполнения задания 1

Задание 2.

Способы создания жестких ссылок:

$ ln ‘файл, для которого надо сформировать ссылку’ ‘имя ссылки’.

$ ln ‘файл, для которого надо сформировать ссылку’ ~/’нужная директория’/‘имя ссылки’.

$ ln ~/’нужная директория’/‘файл, для которого надо сформировать ссылку’ ‘имя ссылки’.

Найти все жесткие ссылки на файл по заданному inode можно с помощью команды ls -liR | grep “^’inode’”.

Пример выполнения:

Создаем жесткую ссылку на файл myfile.txt в данной директории:

$ ln myfile.txt link1

Создаем жесткую ссылку на файл myfile.txt в другой директории:

$ ln myfile.txt ~/OS/link1

Получаем все жесткие ссылки на myfile.txt с помощью скрипта:

$ ls -liR | grep "^52378 "

Результат выполнения данной команды:

52378 -rw-r--r-- 3 dari dari 55406 Feb 14 15:02 link1

52378 -rw-r--r-- 3 dari dari 55406 Feb 14 15:02 link1

52378 -rw-r--r-- 3 dari dari 55406 Feb 14 15:02 myfile.txt

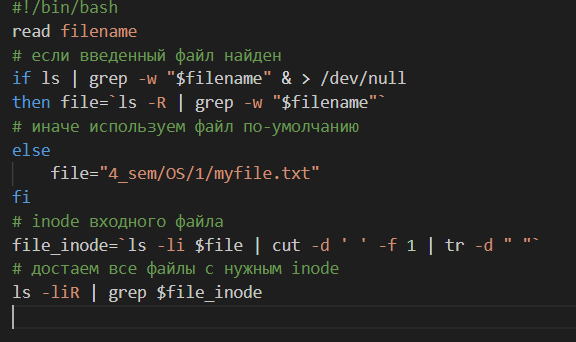


Рисунок 2 –Скрипт для задания 2

Задание 3.

Способы создания символьных ссылок:

$ ln -s source link-1 # создание символьной ссылки с помощью утилиты ln

$ cp -a link-1 link-2 # копирование символьной ссылки

Команду link не получится использовать для создания символьных ссылок, так как с помощью нее можно создать только жесткую. Но можно создать жесткую ссылку на символьную ссылку, тогда для файла, на который ссылается символьная, жесткая так же будет символьной

Скрипт для подсчета и перечисления всех символьных ссылок на заданный файл:

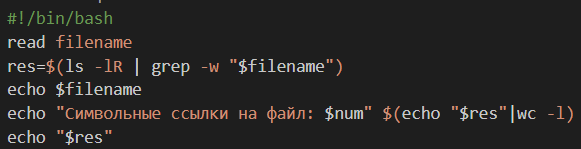


Рисунок 3– Скрипт для выполнения задания 3

Задание 4.



Рисунок 4–выполнение задания 4

Задание 5.

Команда find нужна для поиска файлов и директорий в ФС. Она позволяет задавать несколько параметров для поиска:

1. name – поиск по шаблону имени файла.

2. iname – то же, что –name, только без учета регистра.

3. chmod mode — позволяет изменить права доступа к найденному файлу.

4. type file\_type поиск файлов типа file\_type, где file\_type может быть: b,

c, d, p (FIFO), f (regular file), l, n, s.

5. print – вывод полного имени найденного файла.

6. ls – то же, что print, но выводит в таком формате, как команда ls -l.

7. lname file – поиск символических ссылок на файл file.

8. inode file|n – поиск файлов с тем же серийным номером, как у файла

file, или с серийным номером n.

9. level n – поиск файлов, расположенных в дереве каталогов на n уровней ниже заданного каталога.

10. size – поиск файлов по размеру.

11. mtime – поиск файлов, по дате последнего изменения.

Для получения расширенной информации о всех типах файлов, был написан скрипт, который выводит все файлы, отсортированные по типам:

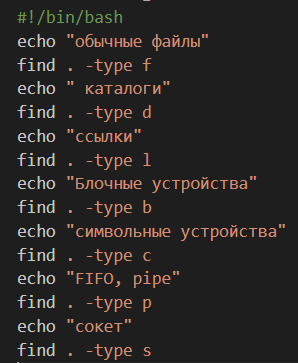


Рисунок 5 – Скрипт для выполнения задания 5

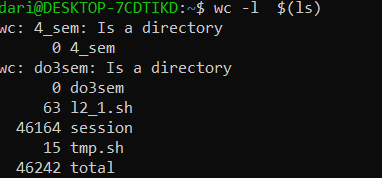


Рисунок 6 – Пример вложенных команд

Задание 6.

Od предназначена для отображения содержимого файлов в различных форматах (разные форматы вывода символов, можно указать смещение и количество байт для вывода, см. Рисунок 7). b - восьмеричный формат, c - символьный формат.

\*dump тоже используется для отображения содержимого файлов, но предоставляет более детальный вывод, включающий адреса, символы и их 16-ричное представление.

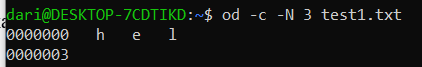


Рисунок 7 – Пример работы od

Выполнение данной команды для исполняемых файлов см. рисунок 8. Первые три буквы ELF свидетельствуют о том, что файл исполняемый или объектный.

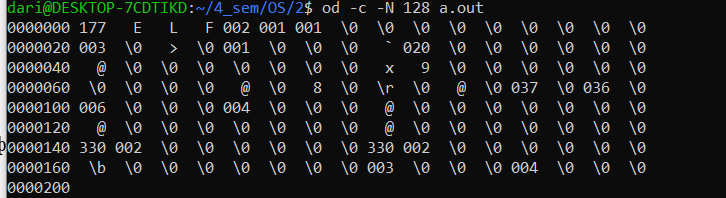


Рисунок 8 – Пример работы od c исполняемыми файлами

Чтобы получить информацию об объектном или исполняемом файле, нужно воспользоваться командой objdump см. рисунок 9.

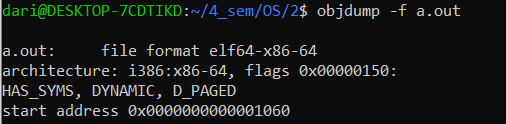
****

Рисунок 9 – Пример работы objdump c исполняемыми файлами

Задание 7.

Для того, чтоб узнать количество записей в каталоге, нужно воспользоваться командой ls -l, в первой строке получим результат:

total 84 – количество блоков памяти для всех файлов, на которые есть ссылки в каталоге.

Чтоб получить информацию о размере каталога, нужно выполнить команду ls -lh:

total 84K

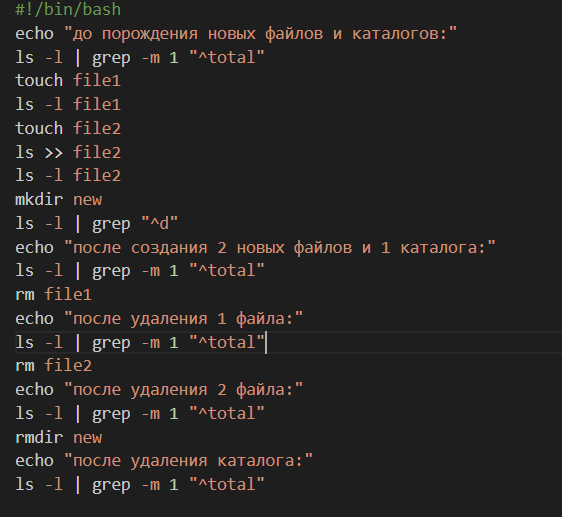


Рисунок 10 – Скрипт для выполнения задания 7

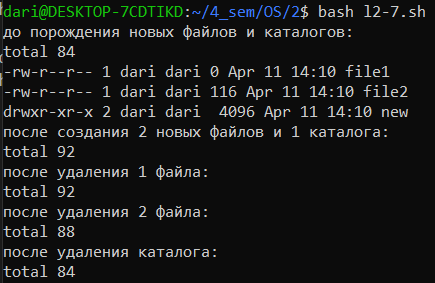


Рисунок 11 – результат работы скрипта для 7 задания

Задание 8.

/etc/passwd – текстовый файл, который хранит информацию о пользователях системы в формате записей, каждая из которых представляет собой строку, разделенную символом двоеточия. Каждая запись содержит следующие поля: имя пользователя, шифрованный пароль, user id, group id, полное имя пользователя, домашний каталог, программа командной оболочки. Этот файл доступен только для чтения группе и остальным пользователям, владелец же может его изменять.

ls -l /etc/passwd

-rw-r--r-- 1 root root 2317 Jun 5 2022 /etc/passwd

Файл /etc/shadow содержит пароли пользователей, информацию о сроке истекших паролей и т.д. Только root имеет доступ к этому файла для чтения и редактирования. Группа имеет доступ к чтению.

ls -l /etc/shadow

-rw-r----- 1 root shadow 1256 Jun 5 2022 /etc/shadow

Команда /usr/bin/passwd предназначена для смены пароля. Принимает в качестве аргумента имя пользователя, открывает интерактивный режим для смены пароля.

ls -l /usr/bin/passwd

-rwsr-xr-x 1 root root 68208 Apr 16 2020 /usr/bin/passwd

У всех есть доступ к выполнению и чтению, изменять может только владелец. s – setuid бит разрешения, который позволяет пользователю запускать исполняемый файл с правами владельца этого файла.

Задание 9.

У каждого файла или директории есть 3 группы прав: права пользователя (определяют права доступа для владельца файла или директории, владелец имеет права на чтение, запись и выполнение), права группы (определяют права доступа для всех пользователей, находящихся в группе, которой принадлежит файл или директория, групповым пользователям могут быть назначены права на чтение, запись и выполнение), права для всех остальных пользователей (может быть настроено на чтение, запись и выполнение).

1. chmod – команда, с помощью которой можно изменять права доступа.

Пример изменения прав доступа в символьном режиме:

ls -l file.c

-rw-r--r-- 1 dari dari 52 Apr 11 13:54 file.c

sudo chmod u+x,g+x,o-r file.c

ls -l file.c

-rwxr-x--- 1 dari dari 52 Apr 11 13:54 file.c

Пример изменения прав доступа в абсолютном режиме (в данном примере каждой группе прав предоставлены все права):

ls -l file.c

-rwxr-x--- 1 dari dari 52 Apr 11 13:54 file.c

sudo chmod 777 file.c

ls -l file.c

-rwxrwxrwx 1 dari dari 52 Apr 11 13:54 file.c

7 – разрешены чтение, запись, исполнение

6 – разрешены чтение и запись

5 – разрешены чтение и исполнение

4 – разрешено только чтение

0 – ничего не разрешено

При использовании этой команды к целому каталогу, нужно применить ключ –R

Команда chown используется для изменения владельца файла или директории. Синтаксис данной команды: chown “options” “new\_owner” “file or directory”.

ls -l file.c

-rwxrwxrwx 1 dari dari 52 Apr 11 13:54 file.c

sudo chown test:test file.c

ls -l file.c

-rwxrwxrwx 1 test test 52 Apr 11 13:54 file.c

1. Установить пользователю setuid бит можно при помощи команды

chmod u+s “filename”. Установка setuid бита позволяет обычному пользователю выполнять команды с повышенными привилегиями (с правами владельца) не входя в систему как владелец файла:

Пример выполнения:

sudo chmod u+s test1.txt

ls -l test1.txt

-rwSr--r-- 1 dari dari 6 Apr 11 12:44 test1.txt

3.

sudo chmod 064 test1.txt

ls -l test1.txt

----rw-r-- 1 dari dari 6 Apr 11 12:44 test1.txt

При отсутствии каких-либо прав у пользователя, но при наличии прав у группы, файл становится недоступен. При обратной же ситуации, права будут. Можно сделать вывод о том, что итоговые права формируются на основании прав доступа пользователя.

4.

|  |  |  |
| --- | --- | --- |
|  | Файл | Каталог |
| Чтение | отображение информации в файле, возможность копирования. | отображение файлов и директорий, которые содержатся в данном каталоге. |
| Запись | способность удалять файл, изменять и сохранять. | добавление или удаление файлов в каталоге. |
| Выполнение | запуск файла, если он исполняемый. | переход в данный каталог. |

Задание 10.

Создан файл test1.txt, содержащий в себе приветствие (hello!). Данный файл был доступен для изменения и чтения лишь test, для других он был недоступен.

--rw------- 1 test test 6 Apr 11 12:44 test1.txt

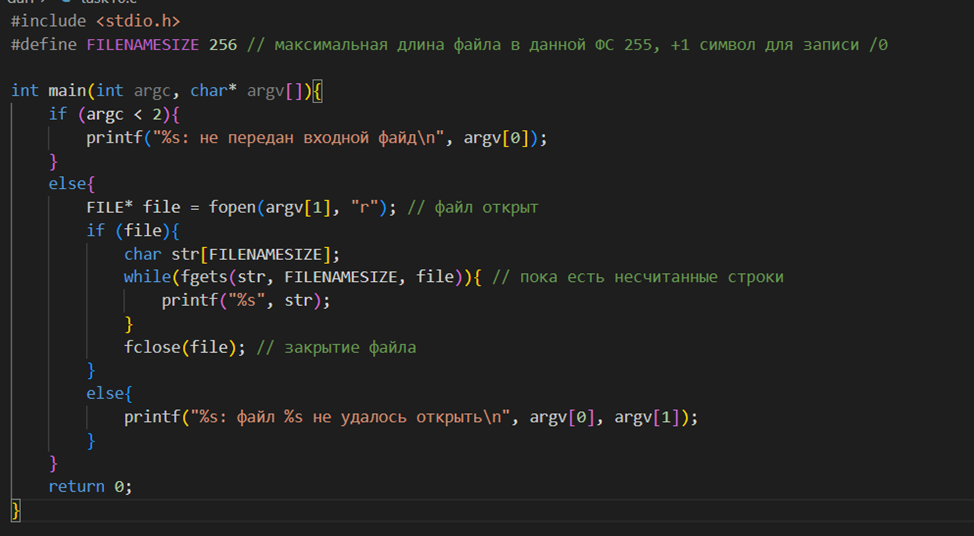


Рисунок 12 – программа для выполнения задания 10

Проверим работоспособность этой программы в случае, когда пользователь dari находится в одной группе с test, и когда его в этой группе нет.Вывод программы в случае, если в группе test нет пользователя dari:

./10 test1.txt

hello!

Чтобы добавить пользователя test к пользователю dari, нужно воспользоваться командой sudo usermod -a -G dari test.

Результат представлен ниже:

getent group dari

dari:x:1000:test

Результат выполнения программы task10.c в данном случае:

./10 test1.txt

hello!

Задание 11.

1. df - команда для вывода списка файловых систем (в строке название системы, ее размер, занятое, свободное пространство, точки монтирования):

Filesystem Size Used Avail Use% Mounted on  
tmpfs 198M 1,6M 196M 1% /run  
/dev/sda3 24G 12G 12G 50% /  
tmpfs 988M 0 988M 0% /dev/shm  
tmpfs 5,0M 4,0K 5,0M 1% /run/lock  
/dev/sda2 512M 6,1M 506M 2% /boot/efi  
tmpfs 198M 4,7M 193M 3% /run/user/1000

«tmpfs» - временные файловые хранилища

«/dev/sdax» - разделы SSD накопителя

Файл /etc/fstab используется для определения, какие файловые системы автоматически монтируются при загрузке системы. Он содержит информацию о файловых системах, доступных для монтирования, а также о параметрах монтирования для каждой файловой системы. Каждая строка в данном файле представляет собой одну файловую систему.

Структура таблицы файловой системы: UUID (идентификатор файловой системы), точка монтирования, тип файловой системы, параметры монтирования, дамп - резервное копирование (не используется), fsck - утилита проверки файловых систем (0 = отключена).

# /etc/fstab: static file system information.  
#  
# Use 'blkid' to print the universally unique identifier for a  
# device; this may be used with UUID= as a more robust way to name devices  
# that works even if disks are added and removed. See fstab(5).  
#  
# <file system> <mount point> <type> <options> <dump> <pass>  
# / was on /dev/sda3 during installation  
UUID=7c480adf-9980-44fe-9caf-b4f0c812126b / ext4 errors=remount-ro 0 1  
# /boot/efi was on /dev/sda2 during installation  
UUID=6F88-EB93 /boot/efi vfat umask=0077 0 1  
/swapfile none swap sw 0 0

Узнать информацию о ФС, доступных для монтирования, можно в файле /proc/filesystems:

cat filesystems

nodev sysfs

nodev tmpfs

nodev bdev

nodev proc

nodev cgroup

nodev cgroup2

nodev cpuset

nodev devtmpfs

nodev configfs

nodev debugfs

nodev tracefs

nodev securityfs

nodev sockfs

nodev bpf

nodev pipefs

nodev ramfs

nodev hugetlbfs

nodev devpts

ext3

ext2

ext4

squashfs

vfat

nodev ecryptfs

fuseblk

nodev fuse

nodev fusectl

nodev mqueue

nodev pstore

nodev autofs

nodev binfmt\_misc

nodev означает, что файловая система не может использоваться для хранения данных на носителе terminal

2. mount выводит список текущих смонтированных файловых систем (название, точка монтирования, тип, опции монтирования, например, rw - режим доступа чтение/запись, nosuid - запрещает SUID, noexec - запрещает исполнение любых двоичных файлов фс, relatime-обновляет время доступа к файлу только в случае его редактирования).

mount

sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)

proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)

udev on /dev type devtmpfs (rw,nosuid,relatime,size=972476k,nr\_inodes=243119,mode=755,inode64)

devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)

tmpfs on /run type tmpfs (rw,nosuid,nodev,noexec,relatime,size=202268k,mode=755,inode64)

/dev/sda3 on / type ext4 (rw,relatime,errors=remount-ro)

securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)

tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,inode64)

tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k,inode64)

cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,nsdelegate,memory\_recursiveprot)

pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)

bpf on /sys/fs/bpf type bpf (rw,nosuid,nodev,noexec,relatime,mode=700)

systemd-1 on /proc/sys/fs/binfmt\_misc type autofs (rw,relatime,fd=29,pgrp=1,timeout=0,minproto=5,maxproto=5,direct,pipe\_ino=16654)

hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,pagesize=2M)

mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime)

tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime)

debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime)

fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)

configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)

ramfs on /run/credentials/systemd-sysusers.service type ramfs (ro,nosuid,nodev,noexec,relatime,mode=700)

/var/lib/snapd/snaps/bare\_5.snap on /snap/bare/5 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/var/lib/snapd/snaps/core20\_1822.snap on /snap/core20/1822 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/var/lib/snapd/snaps/firefox\_2356.snap on /snap/firefox/2356 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/var/lib/snapd/snaps/gnome-3-38-2004\_119.snap on /snap/gnome-3-38-2004/119 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/var/lib/snapd/snaps/gtk-common-themes\_1535.snap on /snap/gtk-common-themes/1535 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/dev/sda3 on /var/snap/firefox/common/host-hunspell type ext4 (ro,noexec,noatime,errors=remount-ro)

/var/lib/snapd/snaps/snap-store\_638.snap on /snap/snap-store/638 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/var/lib/snapd/snaps/snapd\_18357.snap on /snap/snapd/18357 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/var/lib/snapd/snaps/snapd-desktop-integration\_49.snap on /snap/snapd-desktop-integration/49 type squashfs (ro,nodev,relatime,errors=continue,x-gdu.hide)

/dev/sda2 on /boot/efi type vfat (rw,relatime,fmask=0077,dmask=0077,codepage=437,iocharset=iso8859-1,shortname=mixed,errors=remount-ro)

binfmt\_misc on /proc/sys/fs/binfmt\_misc type binfmt\_misc (rw,nosuid,nodev,noexec,relatime)

tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,size=202264k,nr\_inodes=50566,mode=700,uid=1000,gid=1000,inode64)

gvfsd-fuse on /run/user/1000/gvfs type fuse.gvfsd-fuse (rw,nosuid,nodev,relatime,user\_id=1000,group\_id=1000)

portal on /run/user/1000/doc type fuse.portal (rw,nosuid,nodev,relatime,user\_id=1000,group\_id=1000)

tmpfs on /run/snapd/ns type tmpfs (rw,nosuid,nodev,noexec,relatime,size=202268k,mode=755,inode64)

nsfs on /run/snapd/ns/snapd-desktop-integration.mnt type nsfs (rw)

nsfs on /run/snapd/ns/snap-store.mnt type nsfs (rw)

После подключения флешки была добавлена строка:

/dev/sdb1 on /media/dari/\_\_\_\_ type vfat (rw,nosuid,nodev,relatime,uid=1000,gid=1000,fmask=0022,dmask=0022,codepage=437,iocharset=iso8859-1,shortname=mixed,showexec,utf8,flush,errors=remount-ro,uhelper=udisks2)

Команда fdisk -l выводит информацию о всех дисках:

fdisk -l  
Disk /dev/loop0: 4 KiB, 4096 bytes, 8 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/loop1: 63,28 MiB, 66355200 bytes, 129600 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/loop2: 240,61 MiB, 252301312 bytes, 492776 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/loop3: 346,33 MiB, 363151360 bytes, 709280 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/loop4: 91,69 MiB, 96141312 bytes, 187776 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/loop5: 45,93 MiB, 48160768 bytes, 94064 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/loop6: 304 KiB, 311296 bytes, 608 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/loop7: 49,84 MiB, 52260864 bytes, 102072 sectors  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
  
Disk /dev/sda: 25 GiB, 26843545600 bytes, 52428800 sectors  
Disk model: VBOX HARDDISK  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disklabel type: gpt  
Disk identifier: 39E0D90B-EEEA-462F-B499-3C7964DD155C  
  
Device Start End Sectors Size Type  
/dev/sda1 2048 4095 2048 1M BIOS boot  
/dev/sda2 4096 1054719 1050624 513M EFI System  
/dev/sda3 1054720 52426751 51372032 24,5G Linux filesystem

После подключения флешки была добавлена строка:

Disk /dev/sdb: 28,91 GiB, 31042043904 bytes, 60628992 sectors  
Disk model: USB DISK 2.0  
Units: sectors of 1 \* 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disklabel type: dos  
Disk identifier: 0xfe0c8aa2  
  
Device Boot Start End Sectors Size Id Type  
/dev/sdb1 8064 60628991 60620928 28,9G c W95 FAT32 (LBA)

1. B Linux наибольшая длина пути определена в файле /usr/include/linux/limits.h:

/\* SPDX-License-Identifier: GPL-2.0 WITH Linux-syscall-note \*/

#ifndef \_LINUX\_LIMITS\_H

#define \_LINUX\_LIMITS\_H

#define NR\_OPEN 1024

#define NGROUPS\_MAX 65536 /\* supplemental group IDs are available \*/

#define ARG\_MAX 131072 /\* # bytes of args + environ for exec() \*/

#define LINK\_MAX 127 /\* # links a file may have \*/

#define MAX\_CANON 255 /\* size of the canonical input queue \*/

#define MAX\_INPUT 255 /\* size of the type-ahead buffer \*/

#define NAME\_MAX 255 /\* # chars in a file name \*/

#define PATH\_MAX 4096 /\* # chars in a path name including nul \*/

#define PIPE\_BUF 4096 /\* # bytes in atomic write to a pipe \*/

#define XATTR\_NAME\_MAX 255 /\* # chars in an extended attribute name \*/

#define XATTR\_SIZE\_MAX 65536 /\* size of an extended attribute value (64k) \*/

#define XATTR\_LIST\_MAX 65536 /\* size of extended attribute namelist (64k) \*/

#define RTSIG\_MAX 32

#endif

Так, наибольшее количество символов пути файла равняется 4096. Так как имя каталога содержит как минимум два символа (/ + имя), то максимальный уровень вложенности равен 2047 директорий ((2^12) / 2 - 1).

Задание 12.

Утилита file используется для того, чтоб узнать о типе файлов, содержащихся в каком-либо файле.

1. file - команда, определяющая тип данных, содержащихся внутри файла. Утилита выполняет до трех проверок файла: первая - системный вызов stat(), позволяющий получить доступ к структуре inode (выяснить принадлежность файла к типу), второй - если файл является обычным файлом или ссылкой, то читается заголовок файла и пытается определиться его тип на основе магической последовательности байтов, если это не удалось, то утилита переходит к третьей проверке - определяется тип содержимого файла.

2. Результат выполнения команды file без каких-либо ключей:

file gateway.c

gateway.c: C source, ASCII text

С помощью команды file \*.c можно узнать данные о содержимом всех файлов, которые имеют расширение .c:

file \*.c

file.c: setuid C source, ASCII text

gateway.c: C source, ASCII text

С помощью ключа -i можно узнать MIME-тип файла:

file -i gateway.c

gateway.c: text/x-c; charset=us-ascii

Ключ -b позволяет вывести тип файла без имени файла и с небольшим описанием:

file -b gateway.c

C source, ASCII text

3. Для того, чтоб создать новый тип файла, нужно в файл /etc/magic добавить одну строку:

cat /etc/magic

# Magic local data for file(1) command.

# Insert here your local magic data. Format is described in magic(5).

0 string hello\_privet testfile

Была задана магическая последовательность байт для нового типа. Теперь можно создать файл newfile, затем записать в него hello\_privet, после этого при выполнении команды file newfile будет указано, что данный файл имеет тип testfile:

cat newfile

hello\_privet

file newfile

newfile: testfile

## Вывод

В ходе данной лабораторной работы было проанализировано функциональное назначение структурных элементов дерева ФС и определено размещение корневого каталога (корневой ФС).

Кроме того, был изучен процесс создания Bash-скриптов, процесс добавления собственных типов файлов. Были изучены утилиты ls, для просмотра каталогов, утилита find для поиска файла внутри ФС, file, для поиска информации о файле, od для просмотра документа в двоичном режиме, ln – для создания ссылок.

Также был изучен процесс создания и работы с жесткими и символическими ссылками. Был проведен опыт с usb-накопителем, в результате которого в файле fstab было обнаружено подключаемое устройство.

## Список литературы

Душутина Е.В. Системное программное обеспечение. Межроцессные взаимодействия в операционных системах: учеб. пособие. – СПб.: Изд-во Политехн. ун-та, 2016. – 180 с.