**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №4**

**по дисциплине «Операционные системы»**

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

| Студент гр. 1303 |  | Чубан 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

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 входного параметра файл, не используя file (разными способами, не применяя утилиту file).

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.

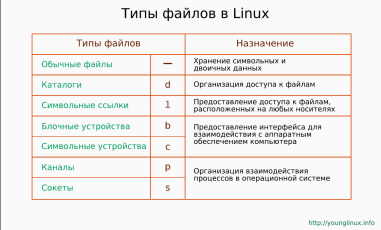
12.1. Привести алгоритм её функционирования на основе информационной базы, размещение и полное имя которой указывается в описании утилиты в технической документации ОС (как правило, /usr/share/file/magic.\*), а также содержимого заголовка файла, к которому применяется утилита. Определить, где находятся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

**Выполнение работы**

1. В UNIX-системах существуют следующие типы файлов:

Рисунок 1 – типы файлов в UNIX-системах

С помощью следующего скрипта отсортируем по одному файлу каждого типа:

| #! /bin/bash    filetypes="-dcblps"  for (( i=0; i<${#filetypes}; i++ )); do  exp="^${filetypes:$i:1}"  `ls -lRa -Imnt -Iwsl / | grep $exp -m 1 -s >>  /root/lb2/file2.txt`  done |
| --- |

Результат работы скрипта:

| -rw-r--r-- 1 root root 56 Mar 2 00:38 filetypes.txt drwxr-xr-x 24 root root 4096 Mar 8 17:45 .  crw-r--r-- 1 root root 10, 235 Mar 8 17:45 autofs brw------- 1 root root 7, 0 Mar 8 17:45 loop0 lrwxrwxrwx 1 root root 7 Aug 20 2021 bin -> usr/bin srwxrwxrwx 1 root root 0 Mar 8 17:45 1\_interop |
| --- |

Используя команду *readlink -f* можно найти полное имя некоторого файла. Примеры файлов с необходимым типом:

• Регулярный файл: /filetypes.txt

• Специальный файл блочного устройства: /dev/loop0

• Файл символьного устройства: /dev/autofs

• Директория: /.

• Символьная ссылка: /bin -> /usr/bin

• Сокет: acpid.socket

2. Был написан скрипт, который находит все жесткие ссылки на заданный файл.

| #! /bin/sh  if [ $# -lt 1 ]  then echo $0: Error: No file  else  filename=$1  inode=`ls -i $filename | cut -d ' ' -f 1 | tr -d " "`  tmp=`ls -lRi -Imnt / | grep $inode >> /root/lb2/file2.txt` fi  echo $tmp |
| --- |

Проходя по директории рекурсивно, находим все файлы, inode которых совпадает с inode файла, переданного в качестве входного параметра. Результат выполнения:

| 64165 -rw-r--r-- 2 root root 28674 Feb 16 16:33 copy.txt 64165 -rw-r--r-- 2 root root 28674 Feb 16 16:33 hard\_link\_copy |
| --- |

3. Сформируем символьные ссылки с помощью ln, link и cp для файла copy.txt

| lrwxrwxrwx 1 root root 8 Mar 8 19:34 first\_symb\_link -> copy.txt  lrwxrwxrwx 1 root root 8 Mar 8 19:35 second\_symb\_link -> copy.txt  -rw-r--r-- 3 root root 28674 Feb 16 16:33 third\_symb\_link |
| --- |

Как можно заметить, ln -s, cp -s создают символьные ссылки, а link – жесткую Напишем скрипт, выполняющий подсчет символьных ссылок

#! /bin/sh

if [ $# -lt 1 ]

then echo $0: Error: no file

else

path=$1

tmp=`ls -lRa -Imnt / | grep "$path" | grep

^l >> ./file\_links.txt`

fi

count=`wc -l ./file\_links.txt | cut -d ' ' -f 1`

echo total $count >> ./file\_links.txt

Проходя рекурсивно по директории, находим файлы, в атрибутах которых есть “l”, ссылающиеся на заданный файл.

Результат выполнения:

| lrwxrwxrwx 1 root root 8 Mar 8 19:34 first\_symb\_link -> copy.txt lrwxrwxrwx 1 root root 8 Mar 8 19:35 second\_symb\_link -> copy.txt  total 2 |
| --- |

4. Найдем все символьные ссылки на заданный файл с помощью следующей команды:

| ls -lRa -Imnt / | grep "copy.txt" | grep ^l |
| --- |

Результат выполнения команды:

| root@DESKTOP-GHA60IS:~/lb2# ls -lRa -Imnt / | grep "copy.txt" | grep ^l lrwxrwxrwx 1 root root 8 Mar 8 19:34 first\_symb\_link -> copy.txt lrwxrwxrwx 1 root root 8 Mar 8 19:35 second\_symb\_link -> copy.txt lrwxrwxrwx 1 root root 14 Mar 8 20:31 another\_link -> /root/copy.txt |
| --- |

5. Изучим утилиту *find*, используя ее ключи получим расширенную информацию о всех типах файлов.

root@DESKTOP-GHA60IS:~/lb2# find ./file2.txt -ls

66661 4 -rw-r--r-- 1 root root 923 Mar 8 20:06 ./file2.txt

root@DESKTOP-GHA60IS:~/lb2# find /root/lb2 -depth

/root/lb2/lb1\_c++/lb1\_c++

/root/lb2/lb1\_c++/source.hpp

…

Вывод поддиректорий глубиной больше 5

root@DESKTOP-GHA60IS:~/lb2# find /root/lb2 -mindepth 5 /root/lb2/lb1\_c++/build/.cmake/api/v1

/root/lb2/lb1\_c++/build/.cmake/api/v1/reply

…

Примеры вложенных команд:

| root@DESKTOP-GHA60IS:~/lb2# find $(ls | grep "txt")  D.txt  a.txt  file2.txt  file\_links.txt  result.txt  test1.txt  root@DESKTOP-GHA60IS:~/lb2# find `ls | grep "txt"`  D.txt  a.txt  file2.txt  file\_links.txt  result.txt  test1.txt |
| --- |

6. Проанализируем содержимое файла с помощью утилит od.

| root@DESKTOP-GHA60IS:~/lb2# echo "file" > od\_file.txt  root@DESKTOP-GHA60IS:~/lb2# od -tc od\_file.txt  0000000 f i l e \n  0000005    root@DESKTOP-GHA60IS:~/lb2# echo "new\_string" >> od\_file.txt root@DESKTOP-GHA60IS:~/lb2# od -tc od\_file.txt  0000000 f i l e \n n e w \_ s t r i n g \n  0000020 |
| --- |

Команда od с опцией –t c выводит дамп памяти, ассоциированный с указанным файлом, побайтно в восьмеричном коде, заменяя код на символы там, где это возможно.

7. С помощью команды df-i найдем максимальное количество записей в каталоге – оно связано с количеством свободных inode.

| 8. root@DESKTOP-GHA60IS:~/lb2# mkdir nice\_os  root@DESKTOP-GHA60IS:~/lb2# df -i nice\_os/  Filesystem Inodes IUsed IFree IUse% Mounted on /dev/sdc 16777216 68116 16709100 1% / |
| --- |

Изменим размер каталога, варьируя количество записей (для этого создадим две программы, порождающие новые файлы и каталоги, а затем удаляющие их, предусмотрев промежуточный и конечный вывод информации о размере подопытного каталога, а также вывод информации о размере каталога после каждого добавления файла или каталога).

| #!/bin/bash  mkdir test  echo $(du -sh ./test)  for (( i=1; i<$1/2 + 1; i++ )) do  touch test/file\_$i.txt  mkdir test/dir\_$i  echo "Количество файлов и папок в папке test: $(ls -U test | wc -l)"  echo $(du -sh ./test)  done  echo "Количество файлов и папок в папке test: $(ls -U test | wc -l)"  echo $(du -sh ./test)  rm -rf test |
| --- |

Результат работы данного скрипта:

| 4,0K ./test  Количество файлов и папок в папке test: 2  8,0K ./test  Количество файлов и папок в папке test: 4  12K ./test  Количество файлов и папок в папке test: 6  16K ./test  Количество файлов и папок в папке test: 8  20K ./test  Количество файлов и папок в папке test: 10  24K ./test  Количество файлов и папок в папке test: 10  24K ./test |
| --- |

8. Содержимое файла 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:Mailing List 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

tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false

uuidd:x:107:114::/run/uuidd:/usr/sbin/nologin

tcpdump:x:108:115::/nonexistent:/usr/sbin/nologin

avahi-autoipd:x:109:116:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin

usbmux:x:110:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin

rtkit:x:111:117:RealtimeKit,,,:/proc:/usr/sbin/nologin

dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin

cups-pk-helper:x:113:120:user for cups-pk-helper service,,,:/home/cups-pk-helper:/usr/sbin/nologin

speech-dispatcher:x:114:29:Speech Dispatcher,,,:/run/speech-dispatcher:/bin/false

avahi:x:115:121:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin

kernoops:x:116:65534:Kernel Oops Tracking Daemon,,,:/:/usr/sbin/nologin

saned:x:117:123::/var/lib/saned:/usr/sbin/nologin

nm-openvpn:x:118:124:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin

hplip:x:119:7:HPLIP system user,,,:/run/hplip:/bin/false

whoopsie:x:120:125::/nonexistent:/bin/false

colord:x:121:126:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin

geoclue:x:122:127::/var/lib/geoclue:/usr/sbin/nologin

pulse:x:123:128:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin

gnome-initial-setup:x:124:65534::/run/gnome-initial-setup/:/bin/false

gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false

sssd:x:126:131:SSSD system user,,,:/var/lib/sss:/usr/sbin/nologin

defrozen:x:1000:1000:defrozen,,,:/home/defrozen:/bin/bash

systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin

fwupd-refresh:x:127:134:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin

Здесь записана информация о пользователях в следующем формате:

<Имя пользователя>:<пароль>:<UID>:<GID>:<комментарии>:<домашний каталог>:<интерпретатор shell>

Он доступен для чтения всем пользователям, на запись только администратору. В целях безопасности пароли в нём не хранятся, для них есть отдельный зашифрованный файл /etc/shadow, который доступен для чтения и записи только администратору. Для того, чтобы обычный пользователь мог изменить свой пароль, существует утилита /usr/bin/passwd, которая доступна администратору и группе root на чтение, запись и исполнение, а всем остальным – на чтение и исполнение. Эта программа может выполнять действия от имени администратора, независимо от того, кто её запустил.

Ознакомимся с содержимым /etc/shadow, включив права суперпользователя (т.к. он доступен для чтения только root)

| root:!:18986:0:99999:7:::  daemon:\*:18858:0:99999:7:::  bin:\*:18858:0:99999:7:::  sys:\*:18858:0:99999:7:::  sync:\*:18858:0:99999:7:::  games:\*:18858:0:99999:7:::  man:\*:18858:0:99999:7:::  lp:\*:18858:0:99999:7:::  mail:\*:18858:0:99999:7:::  news:\*:18858:0:99999:7:::  uucp:\*:18858:0:99999:7:::  proxy:\*:18858:0:99999:7:::  www-data:\*:18858:0:99999:7:::  backup:\*:18858:0:99999:7:::  list:\*:18858:0:99999:7:::  irc:\*:18858:0:99999:7:::  gnats:\*:18858:0:99999:7:::  nobody:\*:18858:0:99999:7:::  systemd-network:\*:18858:0:99999:7:::  systemd-resolve:\*:18858:0:99999:7:::  systemd-timesync:\*:18858:0:99999:7:::  messagebus:\*:18858:0:99999:7:::  syslog:\*:18858:0:99999:7:::  \_apt:\*:18858:0:99999:7:::  tss:\*:18858:0:99999:7:::  uuidd:\*:18858:0:99999:7:::  tcpdump:\*:18858:0:99999:7:::  avahi-autoipd:\*:18858:0:99999:7:::  usbmux:\*:18858:0:99999:7:::  rtkit:\*:18858:0:99999:7:::  dnsmasq:\*:18858:0:99999:7:::  cups-pk-helper:\*:18858:0:99999:7:::  speech-dispatcher:!:18858:0:99999:7:::  avahi:\*:18858:0:99999:7:::  kernoops:\*:18858:0:99999:7:::  saned:\*:18858:0:99999:7:::  nm-openvpn:\*:18858:0:99999:7:::  hplip:\*:18858:0:99999:7:::  whoopsie:\*:18858:0:99999:7:::  colord:\*:18858:0:99999:7:::  geoclue:\*:18858:0:99999:7:::  pulse:\*:18858:0:99999:7:::  gnome-initial-setup:\*:18858:0:99999:7:::  gdm:\*:18858:0:99999:7:::  sssd:\*:18858:0:99999:7:::  defrozen:$6$d5Ge79AJrF.YuLV5$EOk9wdwOcsiqUiiA4vU3hD8OEksvq8TCiFMdF8CO2856/Pg3xodz7Ve3bGPkpcLx.IVXos5Fv4/Q3ccalPeSV.:18986:0:99999:7:::  systemd-coredump:!!:18986::::::  fwupd-refresh:\*:19318:0:99999:7::: |
| --- |

9. С помощью утилиты chmod установим права на чтение, запись и выполнение только для владельца директории, сделаем это рекурсивно с помощью флага -R

| root@DESKTOP-GHA60IS:~/lb2# chmod 700 -R ./os\_lab/  root@DESKTOP-GHA60IS:~/lb2# ll ./os\_lab/  total 12  drwx------ 3 root root 4096 Feb 16 16:40 ./  drwxr-xr-x 18 root root 4096 Mar 8 22:51 ../  drwx------ 2 root root 4096 Feb 16 16:40 test/ |
| --- |

Изменим права доступа, добавив такие же права группе, которой принадлежит директория.

| root@DESKTOP-GHA60IS:~/lb2# chmod 770 -R ./OSstuff/  root@DESKTOP-GHA60IS:~/lb2# ll ./OSstuff/  total 12  drwxrwx--- 3 root root 4096 Feb 16 16:40 ./  drwxr-xr-x 18 root root 4096 Mar 8 22:51 ../  drwxrwx--- 2 root root 4096 Feb 16 16:40 test/ |
| --- |

Теперь изменим владельца директории и поддиректорий

| root@DESKTOP-GHA60IS:~/lb2# chown -R DEFROZEN ./OSstuff/ root@DESKTOP-GHA60IS:~/lb2# ll ./os\_lab/  total 12  drwxrwx--- 3 DEFROZEN root 4096 Feb 16 16:40 ./  drwxr-xr-x 18 root root 4096 Mar 8 22:51 ../  drwxrwx--- 2 DEFROZEN root 4096 Feb 16 16:40 test/ |
| --- |

Создадим в директории файл и расширим права исполнения для суперпользователя с помощью флага SUID

| root@DESKTOP-GHA60IS:~/OSstuff# touch test\_file.txt root@DESKTOP-GHA60IS:~/OSstuff# chmod u+s ./test\_file.txt root@DESKTOP-GHA60IS:~/OSstuff# ll  total 12  drwxrwx--- 3 DEFROZEN root 4096 Mar 9 01:48 ./  drwxr-xr-x 18 root root 4096 Mar 8 22:51 ../  drwxrwx--- 2 DEFROZEN root 4096 Feb 16 16:40 test/  -rwSr--r-- 1 root root 0 Mar 9 01:48 test\_file.txt |
| --- |

Права доступа работают следующим образом:

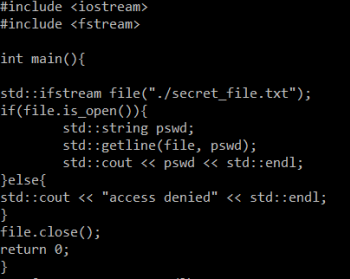
1. Оболочка проверяет, являетесь ли вы владельцем файла, к которому вы хотите получить доступ. Если вы являетесь этим владельцем, вы получаете разрешения и оболочка прекращает проверку.

2. Если вы не являетесь владельцем файла, оболочка проверит, являетесь ли вы участником группы, у которой есть разрешения на этот файл. Если вы являетесь участником этой группы, вы получаете доступ к файлу с

разрешениями, которые для группы установлены, и оболочка прекратит проверку.

3. Если вы не являетесь ни пользователем, ни владельцем группы, вы получаете права других пользователей (Other).

10. Программа-«шлюз»



Для файла ./secret\_file.txt заберем права доступа для всех. В таком случае при попытке открыть данный файл выводится сообщение об отказе в доступе.



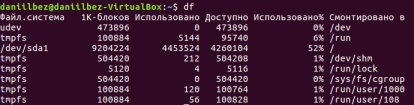
Попробуем запустить программу без установки файла SUID: 

Затем с помощью флага SUID добавим программе права суперпользователя и запустим ее вновь:

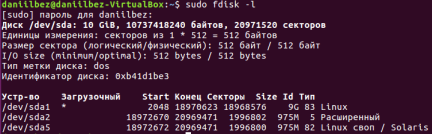


Таким образом, благодаря правам суперпользователя мы получили доступ к файлу, доступ к которому не имел никто.

11. 1) С помощью утилиты df проанализируем занятое дисковое пространство. По умолчанию дисковое пространство измеряется в количестве 512-байтных блоков. Опция –k используется для измерения пространства количеством 1024-байтных блоков. Опция –h используется для измерения пространства в единицах, удобных для чтения человеком (байты, килобайты и т.д.). Опция –P используется для отображения заголовков столбцов таблицы. С помощью ключей -a и -T выведем информацию о файловых системах на компьютере



В данном случае 8 файловых систем. Выводится информация о размере, используемом пространстве, доступном пространстве и точке монтирования.

Проанализируем ФС на компьютере с помощью утилиты fdisk. Данная утилита предназначена для управления разделами жёсткого диска. С помощью ключа -l получим информацию об установленных дисках и разделах. Как мы видим – установлен 1 физический диск, разделенный на 3 логических раздела, загрузочным из них является /dev/sda1.

Затем проанализируем файл /etc/fstab

| root@DESKTOP-GHA60IS:/# cat ./etc/fstab  LABEL=cloudimg-rootfs / ext4 defaults 0 1 |
| --- |

Первое поле – метка показывающая, что монтировать.

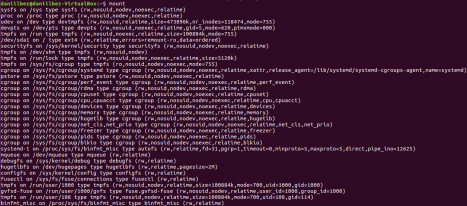
Второе поле показывает, что монтировать нужно в корневой каталог.

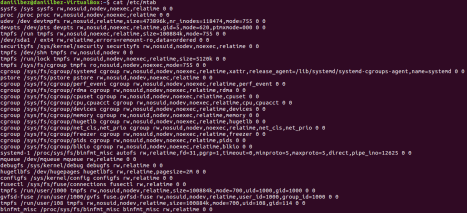
Третье поле указывает тип монтируемой файловой системы. Четвертое – опции монтирования, в данном случае никаких опций нет, default выполняет роль «заглушки».

Пятое - используется утилитой dump для того, чтобы определить, когда делать резервную копию.

Шестое – указывает утилите fsck в каком порядке проверять файловую систему, «1» - наибольший приоритет.

2) Проведем 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.

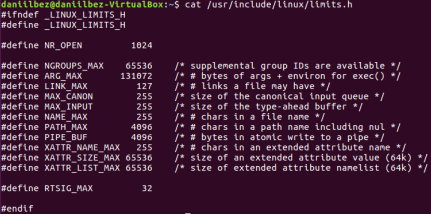
mount — утилита командной строки в UNIX-подобных операционных системах. Применяется для монтирования файловых систем.

mtab — mounted sufle system table — системный файл, в котором прописаны устройства, смонтированные в систему в настоящий момент.

При подключении флеш-накопителя (а также его монтирования) и применении данной команды, отображается соответствующая строка (/dev/sdb1 /media/daniilbez/Transcend ...). Стоит отметить, что в fstab изменений нет. Формат таблицы — имя устройства, режим включения, точка монтирования, тип ФС.

3) Проведем «максимально возможное» дерево ФС, проанализируем, где это указывается.

В файле /usr/include/linux/limits.h определена максимальная длина пути.

Маскимальная длина имени файла 255 байт, маскимальная длина полного пути до файла включая имя 4096 байт. То есть сама вложенность не лимитируется, но длина пути ограничена.

12. *File* определяет тип файла. Для этого она выполняет разные тесты, которые можно разделить на 3 группы:

• Filesystem tests – основаны на анализе кода возврата системного вызова stat(). Программа проверяет не пустой ли файл, и не принадлежит ли он к одному из специальных типов файлов. Все известные типы файлов распознаются, если они определены в системном файле /usr/include/sys/stat.h.

• Magic number tests – используются для проверки файлов, данные в которых записаны в определённом формате. В определённом месте в начале таких файлов записано магическое число, которое позволяет ОС определить тип файла. Все известные ОС магические числа по умолчанию хранятся в файле /usr/share/misc/magic.

• Language tests – используются для анализа языка, на котором написан файл, если это файл в формате ASCII. Выполняется поиск стандартных строк, которые могут соответствовать определённому языку. Первый тест, который завершится успешно, выводит тип файла. Типы файлов можно разделить на 3 основные группы:

• Текстовые – файл содержит только ASCII символы и может быть безопасно прочитан на терминале.

• Исполняемые – файл содержит результаты компилирования программы в форме понятной ядру ОС.

• Данные – всё, что не подходит в первые 2 группы (обычно это бинарные или непечатаемые файлы). Исключение составляют well-known форматы, используемые для хранения бинарных данных.

Синтаксис:

file [-bcLnvz] [-f namefile] [-m magicfile] file ...

Опции:

• -b – не выводить имя файла перед его типом;

• -m magicfile – определяет альтернативный файл с магическими числами; • -с – обычно используется для дебага нового файла с магическими числами перед его использованием;

• -f namefile – определить типы файлов, имена которых записаны в файле namefile;

• -L – определять типы файлов, на которые ссылаются заданные символические ссылки, а не типы ссылок;

• -n – выводить имя файла перед его типом (по умолчанию); • -v – вывести версию программы и выйти;

• -z – пытаться смотреть внутри сжатых файлов.

Запуск утилиты с разными ключами:

| root@DESKTOP-GHA60IS:~# file -b test.cpp  C++ source, ASCII text  root@DESKTOP-GHA60IS:~# file -n test.cpp  test.cpp: C++ source, ASCII text  root@DESKTOP-GHA60IS:~# file -c test.cpp  cont offset type opcode mask value desc |
| --- |

Проведем экспериментальную попытку с добавлением в базу собственного типа файла и его дальнейшей идентификацией. Откроем

/etc/magic с помощью утилиты nano и впишем тип файла 12345, который определен строкой abc в своем содержании.

Создадим файл со следующим содержанием:



С помощью утилиты *file* узнаем тип данного файла.



Так как тип файла определяется как 12345 – свой собственный тип данных был добавлен успешно

**Вывод**

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

**ИСТОЧНИКИ ИНФОРМАЦИИ**

1. Статьи с сайта losst: https://losst.pro/

2. Статьи с сайта bolden: http://linux.bolden.ru/

3. Статьи с сайта andreyex: https://andreyex.ru/

4. Мануалы с помощью команды man

5. Статьи с сайта 900913: https://900913.ru/

6. Статьи с сайта linuxFAQ: https://linux-faq.ru/

7. Статьи с сайта OpenNet: https://www.opennet.ru/

8. Ответы на вопросы с StackOverflow: https://stackoverflow.com/