**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»**

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

Студент гр. 1304 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Герасименко Я.Д.

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.

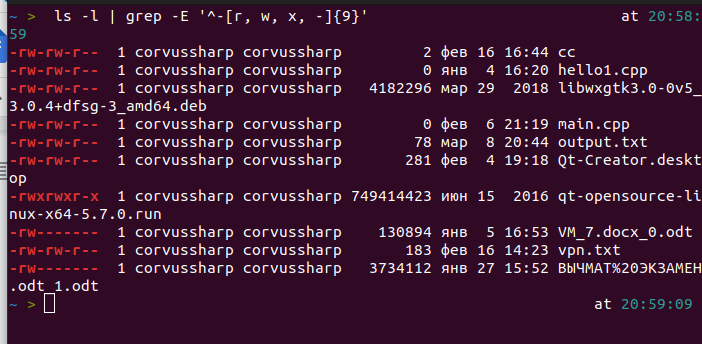
12.3. Привести экспериментальную попытку с добавлением в базу собственного типа файла и его дальнейшей идентификацией. Описать эксперимент и привести последовательность действий для расширения функциональности утилиты file и возможности встраивания дополнительного типа файла в ФС (согласовать содержимое информационной базы и заголовка файла нового типа

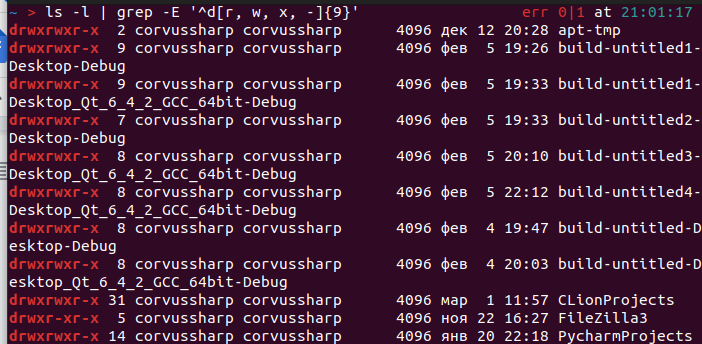
Выполнение работы.

**1. 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**

1. С помощью команды ls -l | grep -E '^-[r, w, x, -]{9}' был произведен поиск файлов каждого типа(после ^ указывается -,b,c,d,l,p,s в зависимости от того, какой тип файла нужна найти).







Аналогично сделаем с разными ключами.

Для поиска всех файлов необходимого типа во всех директориях можно использовать команду:

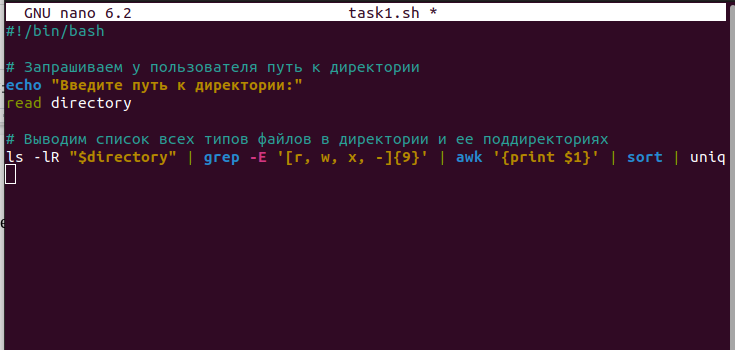
ls -lR | grep -E '[r, w, x, -]{9}'

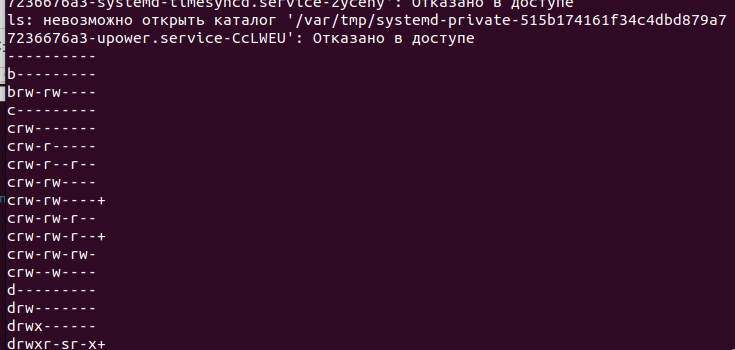
* - для обычного (регулярного) файла
* b для специального файла блочного устройства
* c для файла символьного устройства
* d для директории
* l для символьной ссылки
* p для FIFO
* s для сокета

Команда readlink -f может быть использована для получения полного имени файла:

Необходимо заменить <имя файла> на имя файла, для которого нужно получить полный путь.

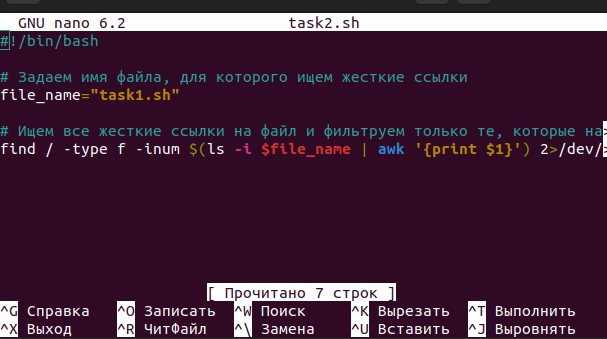
Был написан скрипт, который выводит все типы файлов, которые есть в введенной директории.





**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.**

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





**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.**

3. Cформируем символьную ссылку с помощью:

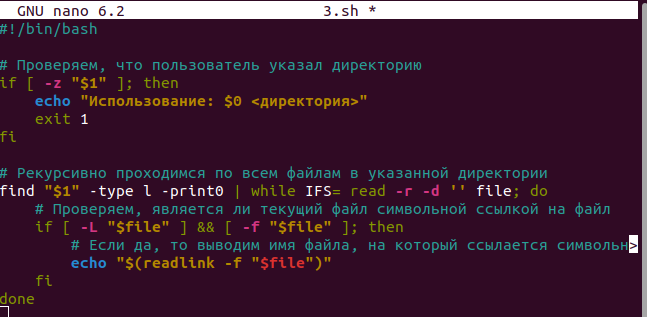
1) ln -s.



2) cp -s



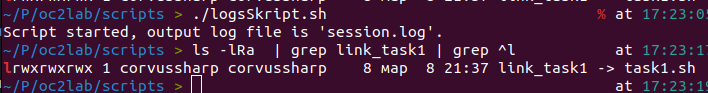
Предложим скрипт, подсчитывающий и перечисляющий все полноименные символьные ссылки на файл, размещаемые в разных местах файлового дерева.



Вызовем его.



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

4. ls -lRa - выполнит рекурсивный поиск всех файлов и директорий в текущей директории и выведет подробную информацию о каждом файле, включая символьные ссылки.

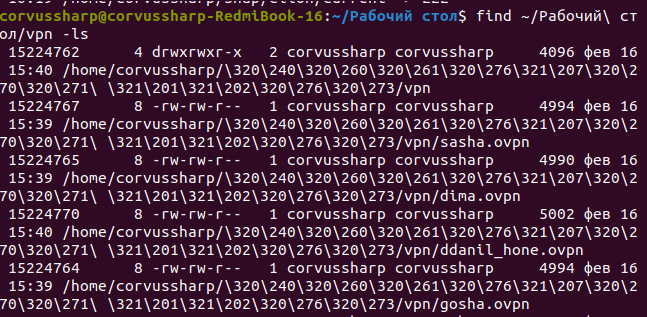
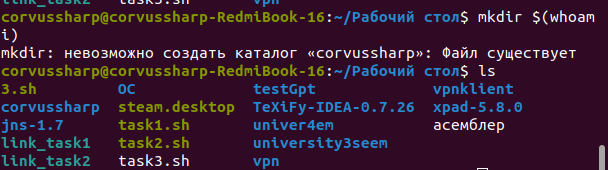
grep link\_task - фильтрует вывод, чтобы найти только строки, содержащие task1.sh.

grep ^l - фильтрует вывод, чтобы найти только символьные ссылки, то есть строки, начинающиеся со знака l.

Если файл task1.sh является символьной ссылкой, то команда выведет информацию об этой ссылке, включая имя файла и путь, на который он ссылается. Если файл l.sh не является символьной ссылкой, то команда не выведет никакой информации. Если символьная ссылка находится в директории, на которую у пользователя нет прав доступа, то команда может выдать ошибку доступа.

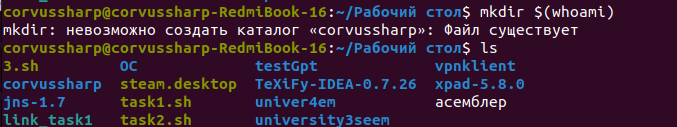
**5. Изучить утилиту find, используя ее ключи получить расширенную информацию о всех типах файлов. Создать примеры вложенных команд.**

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

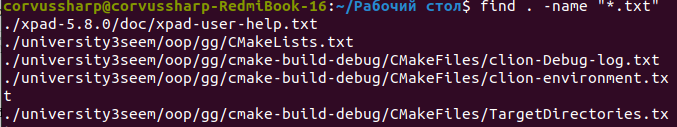


Приведём примеры вложенных команд.

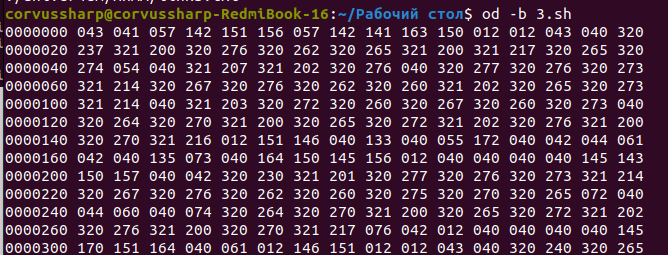
Тут создаётся папка с названием, возвращаемым командой whoami



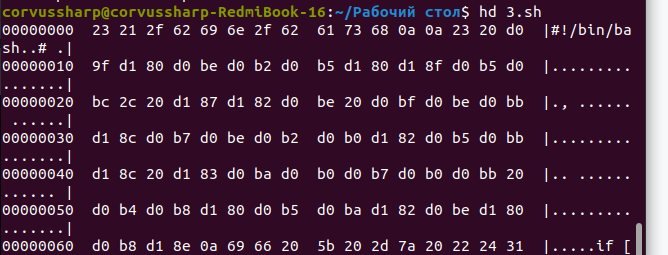
Найдем все txt файлы



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).

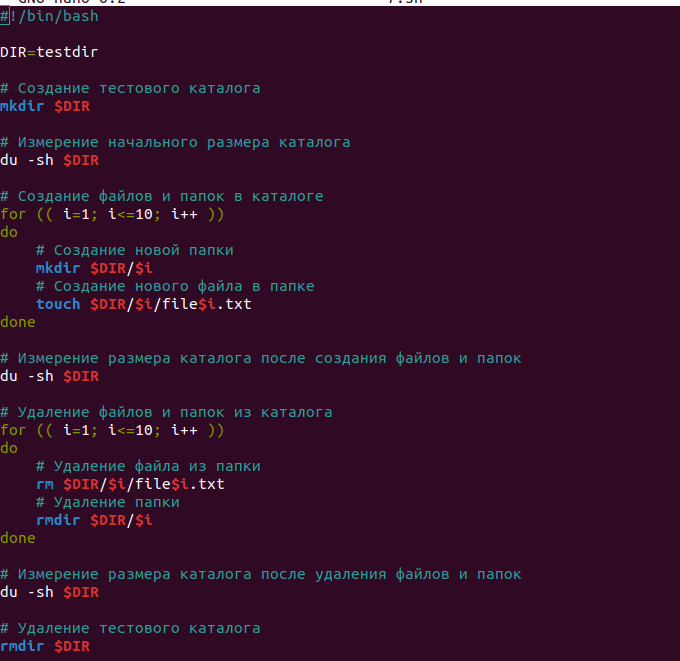
6. Утилита od (octal dump) позволяет вывести содержимое файла в виде октальных (8-ричных) чисел. Например, чтобы вывести содержимое файла 3.sh в виде октальных чисел, нужно выполнить команду: 

Утилита \*dump позволяет вывести содержимое файла в более удобном для чтения виде, с возможностью указания формата вывода (например, шестнадцатеричного или ASCII). Например, чтобы вывести содержимое файла 3.sh в шестнадцатеричном формате, нужно выполнить команду:

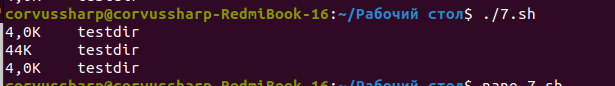


**7. Определить максимальное количество записей в каталоге. Изменить размер каталога, варьируя количество записей (для этого создать программу, порождающую новые файлы и каталоги, а затем удаляющую их, предусмотрев промежуточный и конечный вывод информации о размере подопытного каталога).**

7. Изменим размер каталога, варьируя количество записей (для этого создадим программу, порождающую новые файлы и каталоги, а затем удаляющую их, предусмотрев промежуточный и конечный вывод информации о размере подопытного каталога).

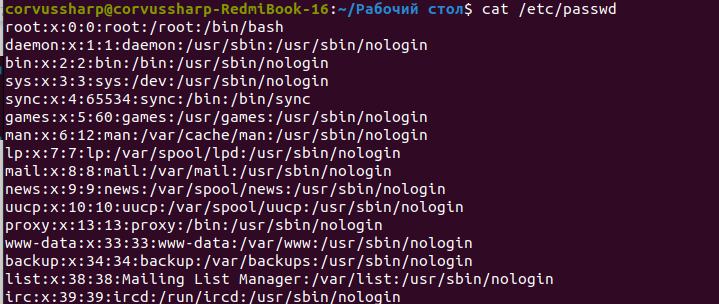


Запустим скрипт на директории test:



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

8. Ознакомимся с содержимым /etc/passwd



Это [файл](https://ru.wikipedia.org/wiki/Файл), содержащий в текстовом формате список пользовательских [учётных записей](https://ru.wikipedia.org/wiki/Учётная_запись) (аккаунтов). Является первым и основным источником [информации](https://ru.wikipedia.org/wiki/Информация) о правах [пользователя](https://ru.wikipedia.org/wiki/Пользователь) [операционной системы](https://ru.wikipedia.org/wiki/Операционная_система).

Файл /etc/passwd содержит следующие записи, разделенные двоеточиями:

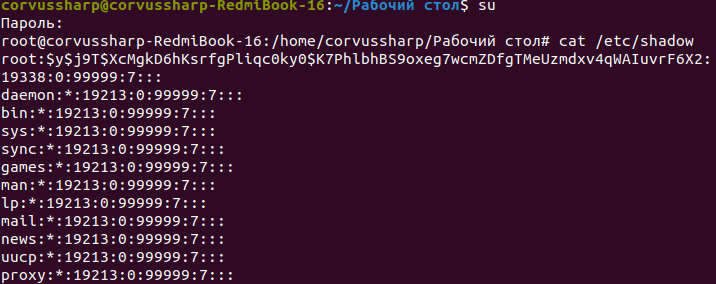
* Имя пользователя
* Зашифрованный пароль
* Цифровой идентификатор пользователя (UID)
* Цифровой идентификатор группы пользователя (GID)
* Полное имя пользователя (GECOS)
* Домашний каталог пользователя
* Оболочка входа в систему

Проанализируем права доступа



Файл /etc/passwd принадлежит пользователю root и должен быть доступен для чтения всем пользователям, но для записи - только пользователю root, на что и указывают права доступа -rw-r--r--.

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

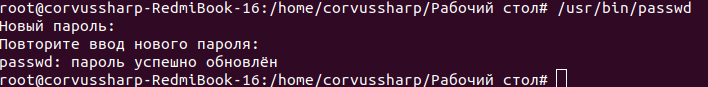


Это зашифрованный файл паролей, в котором хранится зашифрованная информация о паролях для учетных записей пользователей. В дополнение к хранению зашифрованного пароля файл /etc/shadow хранит дополнительную информацию о сроке действия или истечении срока действия пароля.

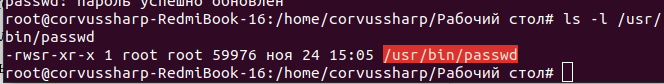
Как и в  файле /etc/passwd, каждая строка в файле /etc/shadow представляет информацию о пользователе. Строки состоят из различных стандартных полей, показанных ниже, каждое из которых разделено двоеточием:

* Логин
* Зашифрованный пароль
* Количество дней, прошедших с 1 января 1970 г., когда последний раз меняли этот пароль.
* Дней до того, как пароль может быть изменен
* Кол-во дней, по истечении которых пароль необходимо изменить
* За сколько дней до истечения срока действия пароля пользователь получает предупреждение
* Через сколько дней после истечения срока действия пароля эта учетная запись будет отключена
* Дней с 1 января 1970 г., когда эта учетная запись была отключена
* Зарезервированное поле

С утилитой /usr/bin/passwd



Она позволяет изменить пароль пользователя.



В утилите /usr/bin/passwd буква s означает разрешение SUID, разрешающее установку идентификатора пользователя, применяется по умолчанию. Это означает, что при смене пароля пользователь временно получает права root, что позволяет ему записывать в файл /etc/shadow.

**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.**

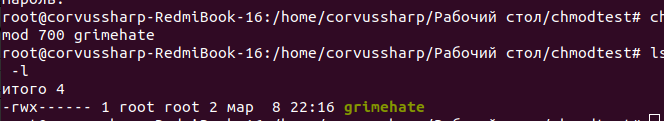
**9.3. Экспериментально установить, как формируются итоговые права на использование файла, если права пользователя и группы, в которую он входит, различны.**

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

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

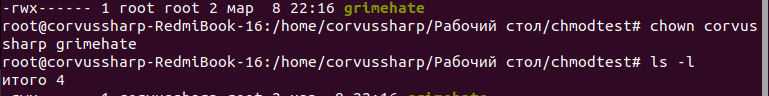


Изменим права с помощью chmod.



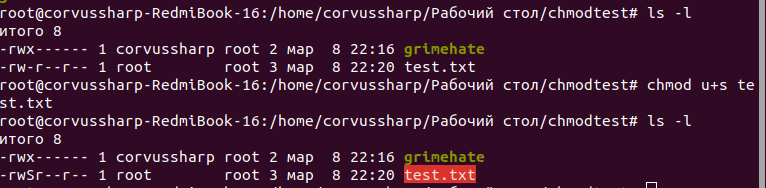
Были изменены права так, что исполнять, читать и записывать теперь имеет право только пользователь-владелец.

Изменим права с помощью chmod.



Теперь владельцем директории является пользователь corvussharp.

9.2. Расширим права исполнения экспериментального файла с помощью флага SUID командой chmod.

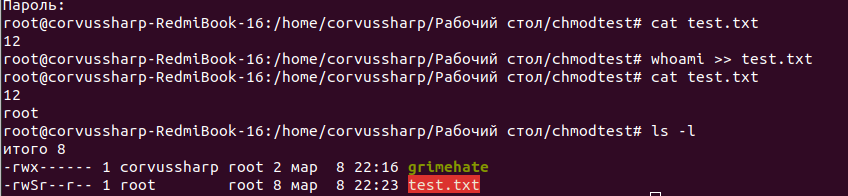


Флаг можно заметить по появившийся букве S в правах пользователя. Также рассмотрим другие виды расширенных прав:

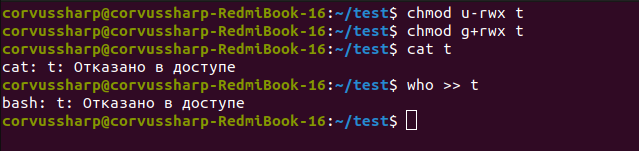


9.3 Экспериментально установим, как формируются итоговые права на использование файла, если права пользователя и группы, в которую он входит, различны.

Для этого создадим файл test.txt и рассмотрим пользователя root и группу root. Первым делом посмотрим, что будет, если права на чтение, запись и исполнение принадлежат только владельцу, которым является root.



Как видим, все операции успешно выполняются. Добавим все права группе и уберём их у владельца.

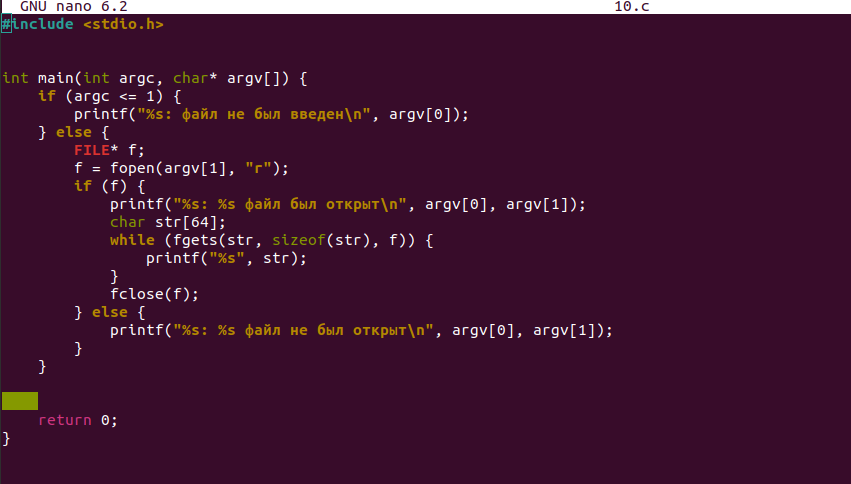


В данном случае пользователь roman ничего не может делать не смотря на права группы, так как владельцу чтение, запись и исполнение запрещены.

**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).**

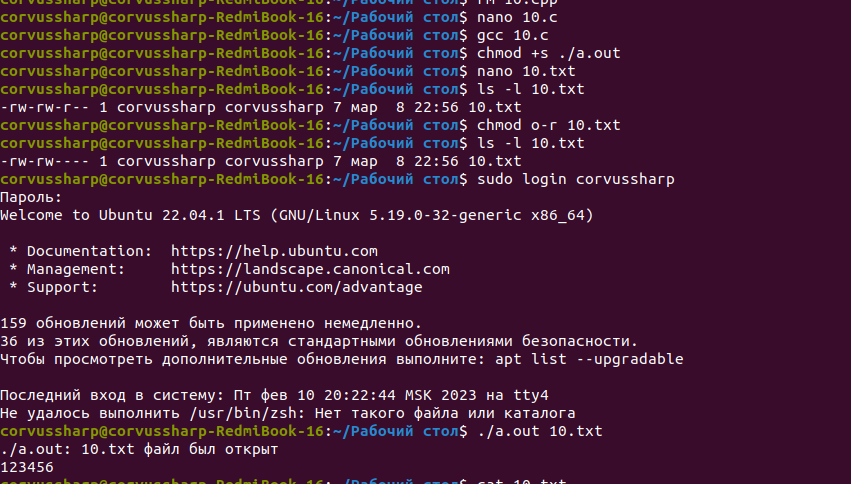
10. Разработаем «программу-шлюз» для доступа к файлу другого пользователя при отсутствии прав на чтение информации из этого файла.

Программа, считывающая файл выглядит так:



Командой chmod +s ./a.out дадим исполняемому файлу программы разрешение SUID.

Далее создадим файл 10.txt, отобрав у остальных пользователей право чтения и сменим пользователя-владельца roman на другого, mamin.



После этого попробуем прочесть файл без программы и с помощью неё.

Видим, что из-за SUID у исполняемого файла нам удаётся прочитать содержимое файла 10.txt.

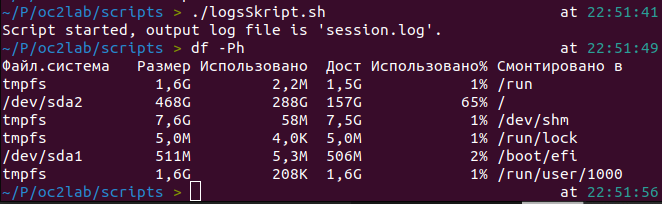
**11. Применяя утилиту df и аналогичные ей по функциональности утилиты, а также информационные файлы типа fstab, получить информацию о файловых системах, возможных для монтирования, а также установленных на компьютере реально.**

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

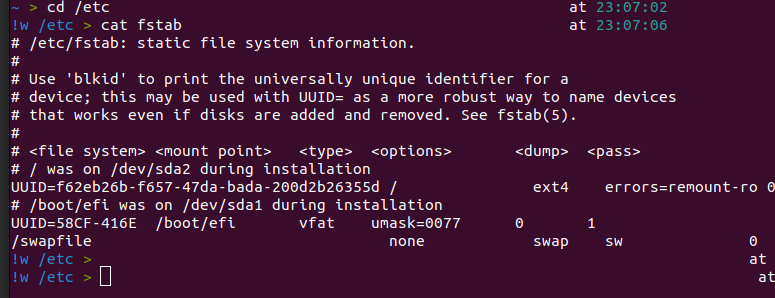
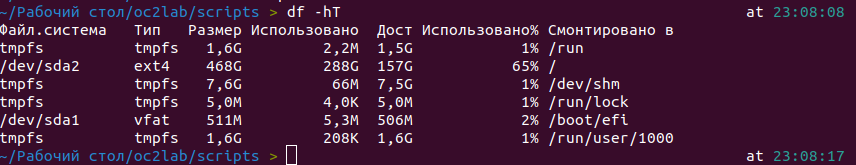
**11.2. Привести образ диска с точки зрения состава и размещения всех ФС на испытуемом компьютере, а также образ полного дерева ФС, включая присоединенные ФС съемных и несъемных носителей. Проанализировать и указать формат таблицы монтирования.**

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

11. df — утилита в UNIX-подобных системах, показывает список всех файловых систем по именам устройств, сообщает их размер, занятое и свободное пространство и точки монтирования.



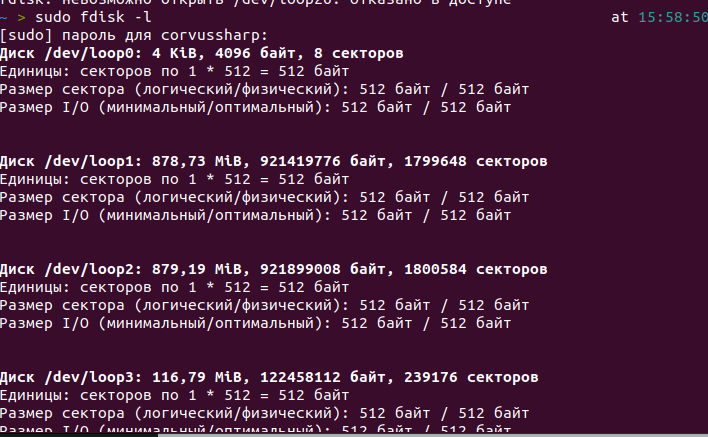
* -a - отображает информацию о всех файловых системах, включая системные файловые системы;
* -h - отображает размеры в удобочитаемом формате, таком как "1K" для килобайта, "1M" для мегабайта и т.д.;
* -H - отображает размеры в единицах, привязанных к степени 1000, в отличие от -h, который использует степень 1024;
* -i - отображает количество занятых и свободных индексных дескрипторов файловой системы;
* -t <файловая система> - отображает информацию только о файловых системах указанного типа, например, ext4, ntfs или tmpfs;
* -T - отображает тип файловой системы;

-x <файловая система> - исключает определенную файловую систему из списка отображаемых.  
  
Применяя утилиту df и аналогичные ей по функциональности утилиты, получить информацию о файловых системах, возможных для монтирования, а также установленных на компьютере реально.  
Выведена информация файла fstab:  
  
В файле обычно перечислены все доступные разделы диска и другие типы файловых систем и источников данных, которые могут не обязательно располагаться на дисках, и указано, как они должны быть инициализированы или иным образом интегрированы в более крупную структуру файловой системы.  
Команда df предоставляет опцию для отображения размеров в удобочитаемых форматах — “-h” (выводит результаты в удобном формате (например: 1K, 2M, 3G)).  
-P – использует формат вывода POSIX.  
В первой колонке вывода df написаны файловые системы. В следующей указан размер этой файловой системы. В третьей сколько используется. Потом указано сколько доступно памяти. Затем сколько занято в процентном соотношении. В последней где смонтирована файловая система.  
11.1 tmpfs - временное файловое хранилище во многих Unix-like ОС. Предназначена для монтирования ФС, но размещается в ОЗУ вместо физического диска. Подобвая конструкция является RAM диском.  
udev - менеджер устройств для новых версий ядра Linux,. Его основная задача - обслуживание файлов устройств (англ. device nodes) в каталоге /dev и обработка всех. действий, выполняемых в пространстве пользователя при добавлении/отключении внешних устройств.  
/dev/nvme0n1p8 - раздел жесткого диска.  
11.2. df -T – фильтрует вывод по типу файловой системы. Был приведен 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.  
  
  
  
В первой колонке вывода df написаны файловые системы. Потом показан тип файловой системы. В следующей указан размер этой файловой системы. В четвертой сколько используется. Потом указано сколько доступно памяти. Затем сколько занято в процентном соотношении. В последней где смонтирована файловая система.

* Команда "mount" используется в Linux для монтирования файловых систем. Ниже приведены некоторые различные информационные команды "mount":
* "mount" без аргументов: Эта команда отображает все файловые системы, которые в данный момент смонтированы в системе.
* "mount -a": Эта команда монтирует все файловые системы, которые определены в файле "/etc/fstab".
* "mount -l": Эта команда выводит информацию о всех смонтированных файловых системах, включая тип файловой системы, точку монтирования и параметры монтирования.
* "mount -t <тип\_файловой\_системы>": Эта команда показывает все файловые системы, смонтированные с использованием определенного типа файловой системы. Например, "mount -t ext4" отображает все файловые системы, смонтированные с использованием файловой системы ext4.
* "mount <точка\_монтирования>": Эта команда выводит информацию о файловой системе, смонтированной в определенной точке монтирования.
* "mount -v": Эта команда выводит более подробную информацию о процессе монтирования файловой системы, включая вывод системных вызовов, используемых при монтировании.
* "mount -o remount <точка\_монтирования>": Эта команда перезагружает файловую систему, смонтированную в определенной точке монтирования, с использованием тех же параметров монтирования.
* "mount -f <устройство> <точка\_монтирования>": Эта команда принудительно монтирует файловую систему из указанного устройства в определенной точке монтирования, даже если файловая система содержит ошибки.

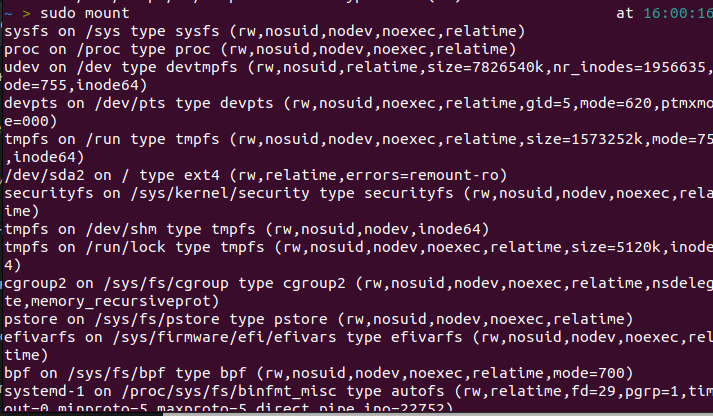
Эти команды могут быть полезными при администрировании Linux-систем и управлении файловыми системами.

Определить, какое устройство необходимо скопировать. Для этого можно использовать команду "fdisk -l" или "lsblk", которые отобразят список доступных устройств и их разделов.

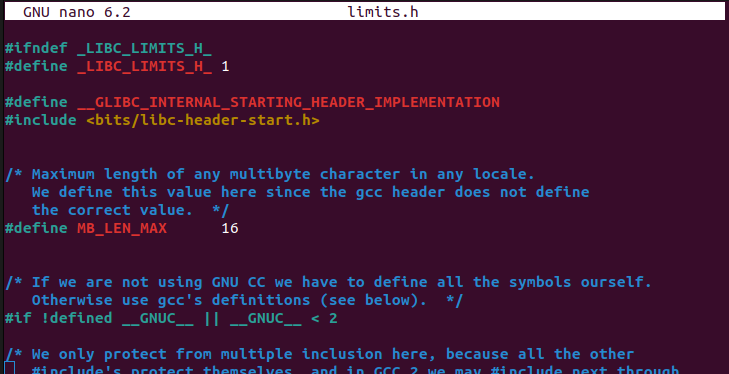


Определить точку монтирования файловой системы на этом устройстве. Для этого можно использовать команду "mount" без аргументов, которая отобразит все смонтированные файловые системы в системе.

Отмонтировать файловую систему с помощью команды "umount", чтобы она не была использована в процессе создания слепка диска.Создать слепок диска с помощью команды "dd" или "dcfldd". Например, команда "dd if=/dev/sda of=/mnt/sda\_image.dd bs=512" создаст слепок диска с устройства "/dev/sda" в файл "/mnt/sda\_image.dd" с размером блока 512 байт.



Fstab – Постоянная информация для монтирования ФС (полный путь – /etc/fstab. Файл /etc/fstab содержит информацию о смонтированных файловых системах. Выше прикреплён скриншот.  
Посмотреть список всех монтированных устройств можно использовав mount  
Формат строки таблицы: имя устройства, режим включения, точка монтирования,тип ФС С помощью этой утилиты можно отследить флеш-накопитель.  
/proc/filesystems — файл, содержащий перечень всех поддерживаемых ядром типов файловых систем. Строки, которые начинаются с nodev значат то, что ФС не является физической.

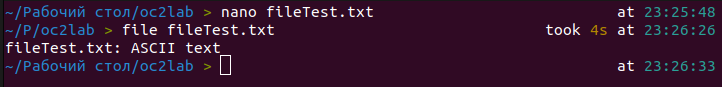
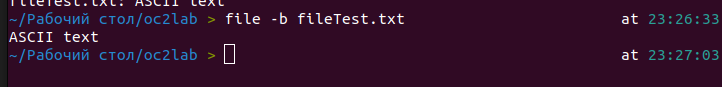
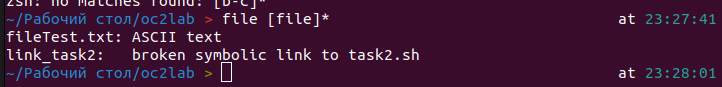
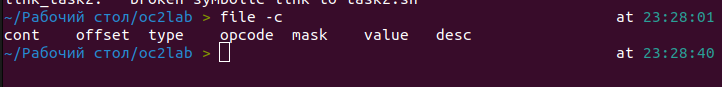
11.3. B Linux наибольшая длина пути определена в файле /usr/include/linux/limits.h.  
  
Так, наибольшее количество символов пути файла равняется 4096.  
Зная, что имя каталога содержит хотя бы 2 символа (/ + имя), то максимальный уровень вложенности равен 2047 директорий).  
Исходя из вывода прошлой команды можно увидеть, что максимальное дерево ФС - /dev/nvme0n1p8, так как его размер максимален относительно всех остальных файловых систем. Также он смонтирован в корневой директории, поэтому имеет дерево максимального размера.

**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.**  
12. file – определяет тип введенного файла. Утилита file позволяет получить информацию о типе файла: она сканирует начало файла и пытается определить его тип. Есть три типа тестов: это тест файловой системы, тест магических чисел и тест языка. Все они применяются в указанном порядке, результат выдает первый успешно закончившийся тест.  
Синтаксис:  
file [-bcLnvz] [-f namefile] [-m magicfile] file ...  
12.1. Для определения типа команда 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/file/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.  
12.2. Все ключи:  
-b, —brief — запрет на демонстрацию имен и адресов файлов в выводе команды;  
-i, —mime — определение MIME-типа документа по его заголовку;  
—mime-type, —mime-encoding — определение конкретного элемента MIME;  
-f, —files-from — анализ документов, адреса которых указаны в простом текстовом файле;  
-l, —list — список паттернов и их длина;  
-s, —special-files — предотвращение проблем, которые могут возникнуть при чтении утилитой специальных файлов;  
-P — анализ определенной части файла, которая обозначается различными параметрами;  
-r, —raw — отказ от вывода /ooo вместо непечатных символов;  
-z — анализ содержимого сжатых документов.  
Примеры:  
1) Стандартное применение утилиты file  
  
2) Вывод только типа файла  
  
3) Можно также выполнять поиск, используя диапазон  
  
4) Выводит итоговую информацию расширенной формы magic file.  
  
12.3. Добавлен новый тип файла.

1)Создание тестового файла:

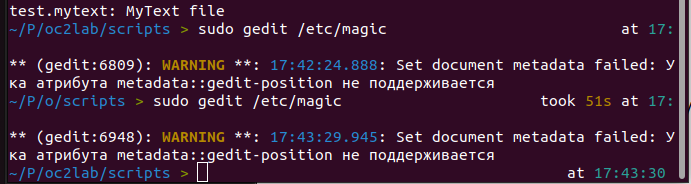
echo "This is my text file" > test.mytext

Определение характеристик файла. В данном случае, будем считать, что каждый файл ".mytext" начинается со строки "MyTextHeader". Таким образом, мы можем определить магическое число "MyTextHeader" в базе данных утилиты file.



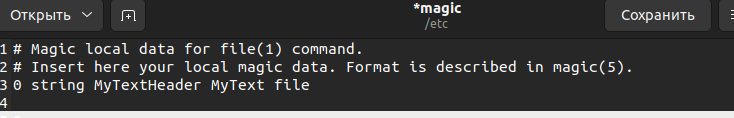
2)Открытие файла базы данных утилиты file и добавление информации о новом типе файла "MyText":

sudo nano /usr/share/file/magic.mgc



3) Добавляем строку в конец файла:

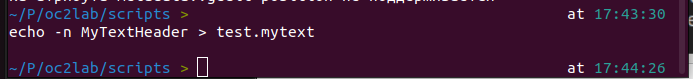
0 string MyTextHeader MyText file



Сохраняем изменения.

4) Добавление информации о новом типе файла в заголовок файла "test.mytext":

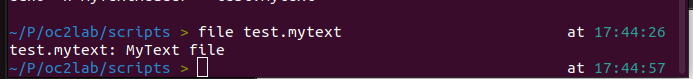
echo -n MyTextHeader > test.mytext



5)Проверка с помощью утилиты file:

file test.mytext

Результатом выполнения команды будет сообщение:



Теперь вы можете использовать свой собственный тип файла "MyText" и определять его с помощью утилиты file.

**Ссылки на литературу**

* **Официальная документация Linux на русском языке: https://www.kernel.org/doc/html/latest/filesystems/index.html**
* **"Описание файловых систем в Linux" на сайте Linux.org.ru: https://www.linux.org.ru/forum/admin/12766089**
* **"Введение в файловую систему Linux" на сайте "Хакер": https://xakep.ru/2015/02/09/linux-filesystem/**
* **"Linux. Основы командной строки. Руководство для начинающих" на сайте "Русский Ubuntu": https://help.ubuntu.ru/wiki/%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%B0%D1%8F\_%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B0**
* **"Как работать с символическими ссылками в Linux" на сайте "Хабр": https://habr.com/ru/post/490364/**
* **"Bash для начинающих" на сайте "Unix-Linux": http://www.unix-lab.org/posts/bash-for-beginners/**
* **"Как работают символические ссылки в Linux" на сайте "OpenNet": https://www.opennet.ru/docs/RUS/linux\_labs/node41.html**
* **"Символические ссылки в Linux: что это такое и как ими пользоваться" на сайте "2DayGeek": https://www.2daygeek.com/ru/symbolic-links-in-linux/**
* **"Как создавать символические ссылки на Linux" на сайте "Хабр": https://habr.com/ru/post/483794/**
* **"Как найти все символические ссылки на файл в Linux" на сайте "LaravelRUS": https://laravel-rus.com/kak-najti-vse-simvolicheskie-ssylki-na-fajl-v-linux**
* **"Как найти символические ссылки на файл в Linux" на сайте "Linux.ru": https://www.linux.ru/docs/links**
* **"Как искать файлы и символические ссылки в Linux" на сайте "Администрирование Linux": https://losst.ru/kak-iskat-fajly-i-simvolicheskie-ssylki-v-linux**
* **"Утилита find в Linux" на сайте "LinuxNotes": https://linuxnotes.org/utiilita-find-v-linux/**
* **"Как использовать утилиту find в Linux" на сайте "Dockeris": https://dockeris.ru/kak-ispolzovat-utilitu-find-v-linux/**
* **"Поиск файлов с помощью команды find в Linux" на сайте "Хабр": https://habr.com/ru/post/122445/**