**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

A logo of a triangle

Description automatically generated**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

|  |  |
| --- | --- |
| Институт | компьютерных наук |
| Кафедра | автоматизированных систем управления |

ЛАБОРАТОРНАЯ РАБОТА №2

По дисциплине "Операционные системы Linux"

На тему "Работа с файловой системой ОС Linux"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ПИ-22-1 | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись, дата | Клименко Н.Д. |
| Руководитель | |  |  |  |
| канд.техн.наук, доцент  ученая степень, ученое звание | |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись, дата | Кургасов В.В. |

Липецк, 2024 г.

**Оглавление**

[Цель работы 3](#_Toc179565712)

[Ход работы 4](#_Toc179565713)

[1. Общая часть 4](#_Toc179565714)

[2. Файлы и каталоги 16](#_Toc179565715)

[3. Пользователи и группы 22](#_Toc179565716)

[4. Архивация и поиск 26](#_Toc179565717)

[5. Создание демона 30](#_Toc179565718)

[Контрольные вопросы 33](#_Toc179565719)

[Вывод 40](#_Toc179565720)

**Цель работы**

Приобрести опыт работы с файлами и каталогами в ОС Linux, настройки прав на доступ к файлам и каталогам.

**Ход работы**

**1. Общая часть**

1.1. Изучить и привести перечень основных каталогов с указанием их назначения.

Для получения перечня основных каталог необходимо выполнить команду ls /, где / – корневая директория. Если к команде ls добавить опцию -l, то можно получить подробную о каталоге (укажет владелица, группу, дату создания, размер и другие параметры). Результат выполнения команды представлен на рисунке 1.

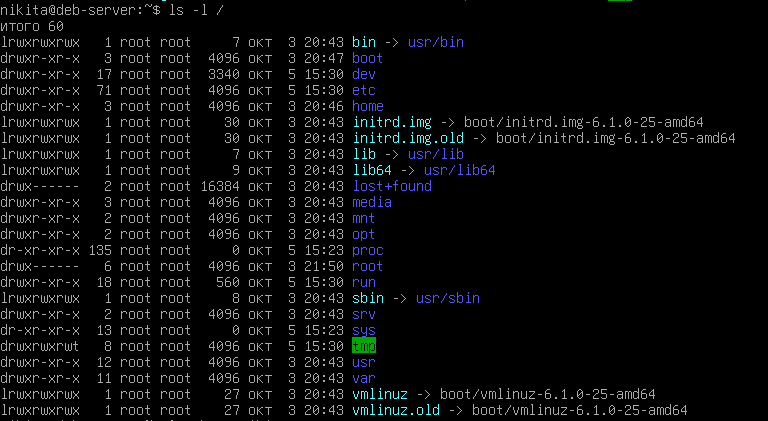


Рисунок 1 – Перечень каталогов

В таблице 1 приведено назначение для каждого каталога из рисунка 1. Также из рисунка 1 можно наблюдать символические ссылки (обозначаются при помощи ). В таблице 1 они также будут рассмотрены.

Таблица 1 – Каталоги и их назначения

|  |  |
| --- | --- |
| **Каталог** | **Назначение** |
| bin | Содержит основные команды. |
| boot | Содержит ядро Linux и другие файлы, необходимые для его своевременного процесса загрузки. |
| dev | Содержит все физические и виртуальные файлы устройств. |

Продолжение таблицы 1

|  |  |
| --- | --- |
| etc | Содержит конфигурационные файлы (данные файлы определяют параметры работы различных служб и программ). |
| home | Содержит домашние каталоги всех пользователей системы и их личные файлы. |
| initrd.img | Используется при загрузке системы для подготовки среды перед загрузкой основного ядра. |
| initrd.img.old | Предыдущая версия initrd.img. Используется при загрузке в случае отката на предыдущую версию ядра. |
| lib | Содержит системные библиотеки, необходимые для работы программ и бинарных файлов. |
| lib64 | Хранит специальные варианты библиотек для систем, поддерживающих несколько бинарных форматов (x86, x64). |
| lost+found | Содержит удаленные/утерянные файлы в результаты операций с диском. |
| media | Содержит точки монтирования съемных устройств (CD-rom, USB ключей и прочее). |
| mnt | Содержит прошлые точки монтирования файловых систем. |
| opt | Директория для дополнительных функций дистрибутива. |
| proc | Виртуальная псевдо-файловая система, которая содержит информацию о запущенных процессах и настройках ядра в виде файлов. |
| root | Содержит личные файлы администратора системы. |
| run | Содержит непостоянные данные среды выполнения, которые не сохраняются после перезагрузки. |
| sbin | Содержит системные программы. |
| srv | Содержит данные, используемые серверами, размещенными в этой системе. |

Завершение таблицы 1

|  |  |
| --- | --- |
| sys | Содержит информацию об устройствах, драйверах и некоторых свойствах ядра. |
| tmp | Содержит временные файлы. Данный каталог очищается при загрузке. |
| usr | Содержит установленные программы, их ресурсы и библиотеки. |
| var | Содержит переменные данные, обрабатываемые демонами. Включает в себя файлы логов, очередей, буфера, кэша и тому подобное. |
| vmlinuz | Сжатый образ ядра Linux, который загружается при старте системы. |
| vmlinuz.old | Предыдущая версия ядра, которая сохраняется для того, чтобы в случае проблем с новым ядром можно было загрузить старую версию |

1.2. Зайти в терминал под root.

Для получения прав администратора (root) используется команда su - (тире используется, чтобы загрузить окружение пользователя), после которой нужно ввести пароль администратора. Данный процесс представлен на рисунке 2.



Рисунок 2 – Получение root

1.3. Посмотреть содержимое каталога файлов физических устройств. В отчете привести перечень файлов физических устройств на рабочем месте с указанием назначения файлов.

Просмотр содержимого каталога файлов физических устройств осуществляться с помощью команды ls /dev. Результат представлен на рисунке 3.

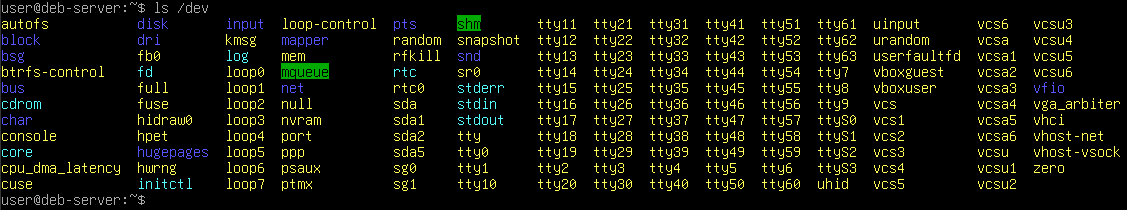


Рисунок 3 – Список устройств

Перечень некоторых файлов устройств представлен в таблице 2.

Таблица 2 – Перечень файлов устройств

|  |  |
| --- | --- |
| **Устройство** | **Назначение** |
| console | Устройство, на которое должны отправляться системные сообщения и будет проходить авторизация в режиме единственного пользователя. |
| dri | Интерфейс, позволяющий пользовательским приложениями получать прямой доступ к видеоаппаратуре. |
| fb0 | Первый кадровый буфер (абстрактная прослойка между ПО и графическим оборудованием). |
| fd | Привод для дискет. |
| full | Специальный файл, являющийся "полным устройством" – запись в него гарантированно вызывает ошибку "недостатка места". Используется для проверки на отсутствие вывода программы. |
| loop0-7 | Позволят монтировать файлы как изолированные файловые системы, отображая файл как виртуальное блочное устройство. |
| mem | Специальный файл, являющийся образом физической памяти компьютера. |
| null | Пустой файл, обнуляющий любые записанные в него байты. |
| ptmx | Хранит файл мультиплексорного устройства псевдотерминала. Он используется для создания пары основного и подчинённого псевдотерминала. |
| pts | Специальный виртуальный каталог временных файлов. |

Окончание таблицы 2

|  |  |
| --- | --- |
| random | Специальное символьное псевдоустройство, предоставляющее интерфейс к системному генератору случайных чисел. |
| urandom | Специальное символьное псевдоустройство, предоставляющее интерфейс к системному генератору псевдослучайных чисел. |
| rfkill | Интерфейс управления радиопередатчиками (WiFi, Bluetooth и прочие). |
| rtc | Интерфейс к драйверам часов реального времени. |
| sda | Специальный файл устройства, предоставляющий доступ к жёстким дискам с SCSI-интерфейсом. Буква в конце обозначает новое устройство, число обозначает номер раздела. |
| sr0 | Приводы оптических дисков. |
| stderr | Хранит стандартный поток ошибок. |
| stdin | Хранит стандартный поток ввода. |
| stdout | Хранит стандартный вывод. |
| tty | Файлы поддержки пользовательских консолей. |
| ttyS | Файлы, обеспечивающие работу с последовательными портами. |
| zero | Специальный файл, являющийся источником нулевых байтов. |

1.4. Перейти в директории пользователя root. Посмотреть содержимое каталога. Посмотреть содержимое файла vmlinuz. Посмотреть и пояснить права доступа к файлу vmlinuz.

Для перехода в директорию пользователя root необходимы права администратора (описаны в пункте 1.2), после достаточно прописать cd /root. С помощью команды ls -la, можно увидеть обычные и скрытые файлы данной директории, представлены на рисунке 4. В таблице 3 приведено их описание.

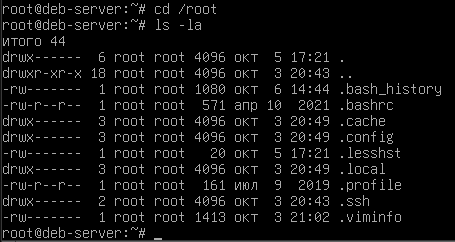


Рисунок 4 – Содержимое каталога root

Таблица 3 – Перечень содержимого каталога root

|  |  |
| --- | --- |
| **Файл/каталог** | **Назначение** |
| bash\_history | Файл, содержащий историю команд, выполненных в оболочке Bash для текущего пользователя. |
| bashrc | Скрипт инициализации Bash, который выполняется при каждом запуске нового терминала для настройки окружения. |
| cache | Каталог, используемый различными программами для хранения временных данных. |
| config | Каталог для хранения конфигурационных файлов различных приложений, запущенных от имени root. |
| lesshst | Файл, в котором хранится история команды less, используемой для просмотра файлов. |
| local | Каталог, содержащий пользовательские данные и файлы, установленные локально. |
| profile | Скрипт, выполняемый при запуске сессии пользователя, который настраивает переменные окружения и другие параметры. |
| ssh | Каталог для хранения ключей и конфигурационных файлов SSH. |
| viminfo | Файл, в котором редактор Vim сохраняет информацию о последнем сеансе работы. |

Файл vmlinuz представляет собой образ ядра операционной системы Linux, который загружается в память системы на этапе загрузки и является основным компонентом операционной системы, отвечающим за управление аппаратными ресурсами и выполнение процессов. Содержимое файла vmlinuz – это скомпилированный двоичный код (собранное и сжатое ядро Linux).

Для просмотра прав доступа файла потребуется команда ls -l /boot/vmlinuz-6.1.0-25-amd64. Выполнение данной команды представлено на рисунке 5.



Рисунок 5 – Права доступа на vmlinuz

Из прав доступа на рисунке 5 видно, что файл доступен для чтения всем пользователям, а администратор (root) имеет право на запись. Однако ни у кого, включая администратора, нет прав на исполнение. Это связано с тем, что ядро не исполняется как обычный исполняемый файл, оно загружается в память операционной системы загрузчиком.

1.5. Создать нового пользователя user.

Добавление нового пользователя осуществляется при помощи команды sudo adduser <user\_name>. Далее необходимо указать информацию о новом пользователе. Чтобы переключиться на пользователя user, необходимо ввести команду su - <user\_name>. Процесс добавления нового пользователе представлен на рисунке 6.

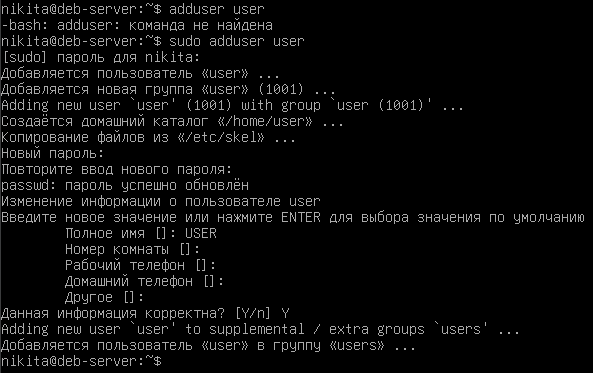


Рисунок 6 – Создание нового пользователя

1.6. Создать в директории пользователя user три файла 1.txt, 2.txt., 3.txt., используя команды touch, cat и тактовый редактор (на выбор vi/nano). Просмотреть и пояснить права доступа к файлам.

Для создания файлов 1.txt, 2.txt., 3.txt. были использованы следующие команды:

- touch 1.txt (содеется пустой файл);

- cat > 2.txt (для выхода из режима заполнения необходимо ввести CTRL+D);

- nano 3.txt (для выхода из режима заполнения необходимо ввести CTRL+X).

На рисунке 7 представлены права доступа к файлам.

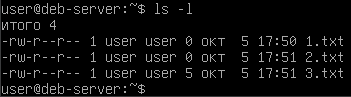


Рисунок 7 – Права доступа к файлам

Из рисунка 7 видно, что все файлы имеют одинаковые права доступа, а именно -rw-r--r--, где:

1. "-": признак файла, иначе бы было "d" (для каталога) или "l" (для символической ссылки);
2. Владелец: имеет право на чтение ("r"), запись ("w"), но не имеет право на исполнение (стоит "-" вместо "x");
3. Группа: имеет право только на чтение;
4. Все: имеют право только на чтение.

Таким образом, права доступа -rw-r--r-- означают, что владелец может читать и редактировать файлы, а другие пользователи могут его только просматривать.

1.7. Перейти в директории пользователя root. В отчете описать результат.

Данный процесс описан в пункте 1.4.

1.8. Изменить права доступа на файл 1.txt в директории пользователя user.

Изменим права доступа к файлу таким образом, чтобы только владелец мог его просматривать и редактировать, а для остальных данный файл будет не доступен. Для этого необходима команда sudo chmod 600 1.txt. На рисунке 8 представлен результат изменения прав доступа.



Рисунок 8 – Изменение прав доступа

Из рисунка 8 видно, что действительно, права доступа изменились и теперь только владелец сможет просматривать и редактировать файл 1.txt.

1.9. Создать жесткую и символическую ссылки на файл 2.txt. Просмотреть и описать полученные результаты.

Для создания ссылок используется команда ln. Чтобы создать жесткую ссылку достаточно ввести команду ln, источник и указать наименование ссылки. Создание символических ссылки идентично созданию жестких ссылок, за исключением добавления опции -s.

Символическая ссылка содержит адрес нужного файла в файловой системе. При попытке открытия ссылки открывается ключевой файл. При удалении ключевого файла символическая ссылка останется, но уже будет указывать в «никуда», так как ключевой файл перестанет существовать.

Жесткая ссылка указывает на те же данные на диске, что и оригинальный файл. В отличие от символической ссылки, она не хранит путь к файлу, а указывает непосредственно на его данные на уровне файловой системы.

Результат создания ссылок представлен на рисунке 8.

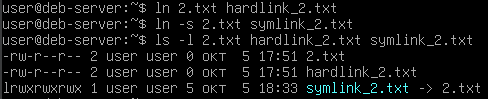


Рисунок 9 – Создание ссылок

Из рисунка 9 видно, что ссылки успешно созданы, в правах доступа символической ссылки вместо первого прочерка стоит "l", у жесткой ссылки стоит прочерк, потому что, как и было написано выше, что жесткая ссылка указывает на те же данные, что и файл, фактически являясь файлом и поэтому между ними нет разницы с точки зрения структуры файловой системы.

1.10. Создать каталог new в каталоге пользователя user.

Для создания каталога необходима команда mkdir <name\_dir>. Результат создания каталога new представлен на рисунке 10.



Рисунок 10 – Создание каталога

1.11. Скопировать файл 1.txt в каталог new.

Копирования файлов осуществляется с помощью команды cp <источник> <назначение>. Результат копирования файла представлен на рисунке 11.



Рисунок 11 – Копирование файла

На рисунке 11 видно, что файл 1.txt находится в двух каталогах: /home/user и /home/user/new, вследствие чего можно сделать вывод, что копирование файла прошло удачно.

1.12. Переместить файл 2.txt в каталог new.

Перемещение файлов осуществляется с помощью команды mv <источник> <назначение>. Результат перемещения файла представлен на рисунке 12. На рисунке 12 видно, что файл 2.txt больше не находится в каталоге /home/user, а был перемещен в /home/user/new, вследствие чего можно сделать вывод, что перемещение файла прошло удачно. Также можно заметить, что символическая ссылка symlink\_2.txt подсвечивается красным цветом, что свидетельствует о потере доступа к файлу по его адресу, поскольку теперь файл 2.txt находится по другому адресу, а именно в /home/user/new/2.txt.

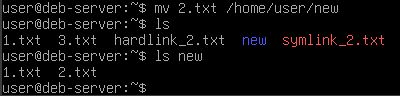


Рисунок 12 – Перемещение файла

1.13. Изменить владельца файла 3.txt и каталога new.

Изменение владельца файла или каталога осуществляется с помощью команды sudo chown <name> <path>. Первоначально владельцем файла 3.txt и каталога new является пользователь user (это следует из команды ls -l). Поменяем владельца на пользователя nikita, данный процесс представлен на рисунке 13.

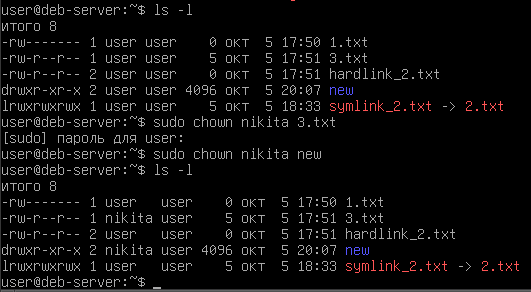


Рисунок 13 – Изменение владельца

1.14. Удалить файл 1.txt в каталоге new.

Удаление файлов осуществляется с помощью команды rm <path>. Но так как пользователь user не является владельцам каталога new, то для выполнения операции удаления необходимо прописать sudo (выполнение команды от имени суперпользователя). Результат удаления представлен на рисунке 14.

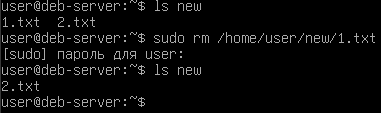


Рисунок 14 – Удаление файла

1.15. Удалить каталог new.

Удаление каталога происходит с помощью команды rmdir <path> (если каталог является пустым), либо при помощи rm -r <path> (опция -r задает рекурсивное удаление каталога, следовательно удаляются все вложенные каталоги и файлы). Результат удаления каталога представлен на рисунке 15.

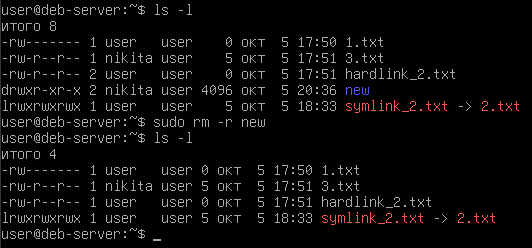


Рисунок 15 – Удаление каталога

**2. Файлы и каталоги**

2.1. Создать 3 текстовых файла разными способами: посредством редакторов vi, mc и команды tee. Файлы должны содержать от 5 до 8 строк осмысленного текста (например, стихи).

Перед созданием файлов необходимо установить mc и vim с помощью команд sudo apt install mc и sudo apt install vim.

Создание файлов:

1. Создание файла с помощью редактора vim: пропишем vim file\_1.txt для создания файла, откроется редактор vim, в режиме «Вставка». После того как текст напечатан, необходимо нажать клавишу "escape" и ввести команду ":wq" – это позволит сохранить файл и выйти.
2. Создание файла с помощью mc: запустим редактор mc, введя команду mc. Сочетанием клавиш Shift+F4 создадим новый файл. Написав текст необходимо использовать сочетание клавиш Ctrl+X, чтобы сохранить файл, дав ему название.
3. Команда tee: данная команда используется для чтения из стандартного ввода и записи в стандартный вывод, поэтому, чтобы создать файл и записать в него текст необходимо прописать команду tee <name\_file> и в терминале ввести текст. Затем сочетанием клавиш Ctrl+D сохранить и закрыть его.

При помощи команды cat можно посмотреть содержимое всех файлов. На рисунке 16 представлены созданные файлы.

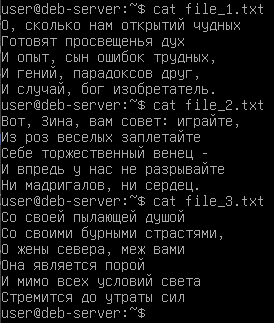


Рисунок 16 – Пример созданных файлов

2.2. Создать структуру каталогов в соответствии с вариантом.

Черными линиями представлена вложенность файлов/подкаталогов в каталоги. Синими линиями представлены ссылки. Красными линиями – символические ссылки. Стрелка на красной линии указывает на целевой файл ссылки. Файлы создаются копированием ранее созданных файлов командос cp с внесением в копию некоторых изменений. Ссылки создаются командой ln, символические ссылки – ей же, но с ключом -s. На рисунке 17 представлено задание, соответствующее варианту.

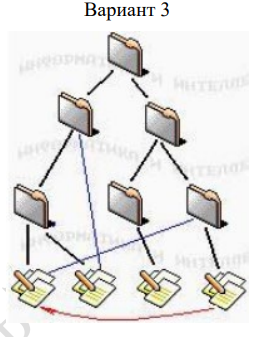


Рисунок 17 – Задание по варианту

Разобьем создание структуры каталогов на несколько этапов.

1. Создание каталогов: создание каталогов осуществляется с помощью команды mkdir <name\_dir>, опция -p позволяет создавать дополнительно родительские каталоги, если они не существуют. Исходя из рисунка 17 видно, что родительская директория содержит две других директории, которые в свою очередь тоже содержат дочерние директории. Команда: mkdir -p dir\_1/dir\_2 dir\_1/dir\_3 dir\_1/dir\_2/dir\_4 dir\_1/dir\_3/dir\_5 dir\_1/dir\_3/dir\_6. Данный процесс представлен на рисунке 18.
2. Создание файлов: создадим один файл 1.txt, используя команду touch, а затем с помощью команды cp <источник> <назначение> скопируем его и при помощи оператора перенаправления ">>" допишем в него некоторые строки. Пример создания файла представлен на рисунке 18.

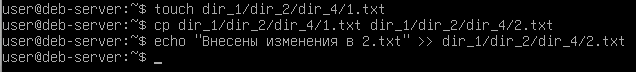


Рисунок 18 – Создание файлов в каталогах

1. Создание жестких ссылок: из рисунка 17 видно, что необходимо создать две жесткие ссылки в каталогах 2 и 6 на файлы 1 и 2. Чтобы создать ссылку в нужном каталоге необходимо перейти в него с помощью команды cd. Затем воспользуемся командой ln <источник> <название ссылки>. Результат создания жестких ссылок представлен на рисунке 19.



Рисунок 19 – Создание жестких ссылок

1. Создание символической ссылки: из рисунка 17 видно, что необходимо создать одну символическую ссылку в каталоге 6, которая будет вести на первый файл каталога 4. Для этого необходимо перейти в каталог 6 и с помощью команды ln и опции -s создать символическую ссылку. Результат представлен на рисунке 20.



Рисунок 20 – Создание символической ссылки

2.3. Вывести на консоль полную информацию о созданных файлах.

Чтобы вывести полную информацию о созданных файлах необходимо воспользоваться командой ls, но с опциями -l (выводит подробный список файлов) и -R (рекурсивно выведет содержимое поддиректорий). На рисунке 21 представлен пример созданных файлов.

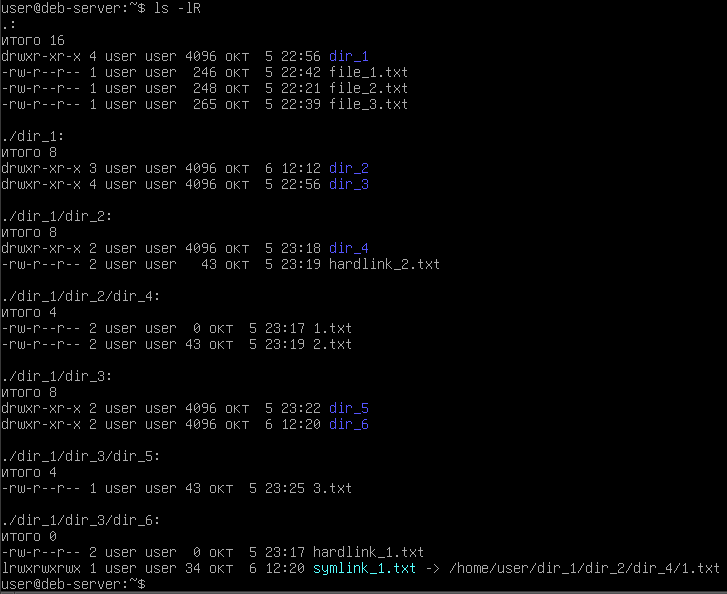


Рисунок 21 – Реализованная иерархия файлов

2.4. Провести ряд экспериментов, иллюстрирующих доступ к файлам по основным именам, по ссылкам и по символическим ссылкам. Для доступа использовать команду cat или редактор vi.

Внесем c помощью команды echo "текст" >> 1.txt строку в файл, так как первоначально он был пустой. Затем с помощью команды cat выведем содержимое файла 1.txt через его имя, а после через символическую ссылку. Проведем аналогичное исследования для файла 2.txt, но уже выведем содержимое через жесткую ссылку. Результат команд представлен на рисунке 22.

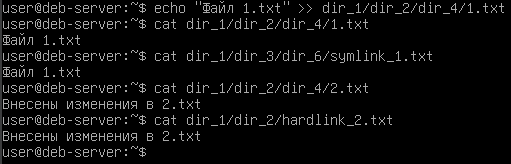


Рисунок 22 – Доступ к файлам через их имена и ссылки

Как видно из рисунка 22 вывод обращение к файлам допустимо как напрямую, то есть через его имя, так и через ссылки, ссылающиеся на него.

2.5. Провести ряд экспериментов, иллюстрирующих реакцию системы на удаление файла, на который имеются ссылки, и файла, на который имеются символические ссылки. Проверять результаты командой ls -ls.

С помощью команды rm удалим файл 1.txt (так как на него сразу ведет и символическая, и жесткая ссылка), после чего попробуем получить доступ к содержимому через символическую и жесткую ссылки. Также проверим результаты с помощью команды ls с опциями -l и -s (выводит размер файла в блоках).

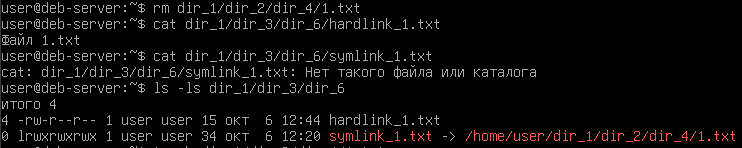


Рисунок 23 –Доступ к содержимому

Из рисунка 23 видно, что жесткая ссылка продолжает хранить данные файла, в отличие от символической ссылки, так как жесткая ссылка ссылается на сами данные файла, а символическая – только на его путь в файловой системе. Также можно заметить размер файла в блоках не изменился: до удаления у жесткой ссылки было 4 блока, а у символической – 0, и после удаления этот размер остался прежним. Это связано с тем, что жесткая ссылка не просто указывает на путь, а фактически является еще одной точкой доступа к тем же данным на диске, тогда как символическая ссылка зависит от наличия исходного файла.

2.6. Уничтожить созданные подкаталоги и файлы в них, сохранив исходные 3 файла.

Чтобы уничтожить созданные каталоги и файлы в них необходимо использовать команду rm с опцией -r (рекурсивно удаляет вложенные файлы и каталоги). Применим данную команду к dir\_1. Результат успешного удаления представлен на рисунке 24.

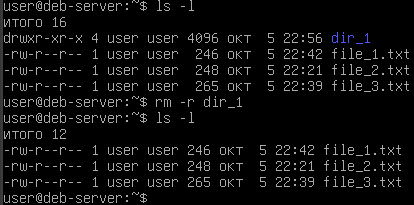


Рисунок 24 – Удаление вложенных каталогов

**3. Пользователи и группы**

3.1. Создать пользователей с именем <ВашеИмяГруппа> и <ФамимлияИмяОтца>.

Для создания новых пользователей потребуется команда sudo adduser <user\_name>. Результат создания новых пользователей представлен на рисунке 25.



Рисунок 25 – Пользователи системы

3.2. Войти в систему под созданным пользователем <ВашеИмяГруппа>.

Сменить пользователя можно с помощью команды su - <user\_name>. На рисунке 26 представлен процесс смены пользователя.



Рисунок 26 – Смена пользователя

3.3. Работа с файлом

Для создания файла <ВашеИмяГруппа> использовалась команда nano Nikita-PI-22-1, которая сразу открыла файл в текстовом редакторе nano для его заполнения. В данный файл был внесен текст "This is test page Клименко Никита Дмитриевич". После чего сохранен сочетанием клавиш Ctrl+X. Пример созданного файла представлен на рисунке 27.

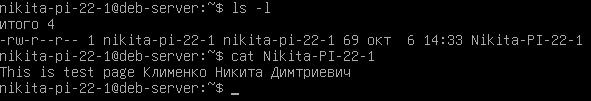


Рисунок 27 – Пример созданного файла

3.4. Переместить файл <ВашеИмяГруппа> в домашний каталог пользователя <ФамимлияИмяОтца>.

Для перемещения файла в домашний каталог другого пользователя используется команда sudo mv <источник> <назначение>. В данном случае sudo применяется для выполнения команды от имени суперпользователя (root). Однако прежде, чем использовать sudo, нужно добавить пользователя nikita-pi-22-1 в группу sudo, чтобы у него была возможность использовать права администратора. Это делается через редактирование конфигурационного файла с помощью команды sudo visudo (перед этим необходимо зайти под суперпользователем, прописав su -). В открывшемся файле необходимо добавить строку nikita-pi-22-1 ALL=(ALL:ALL) ALL. Эта строка позволяет пользователю nikita-pi-22-1 на любом хосте выполнять команды с правами администратора через sudo. После внесения изменений пользователь сможет использовать команды с повышенными привилегиями. Результат переноса файла представлен на рисунке 28.

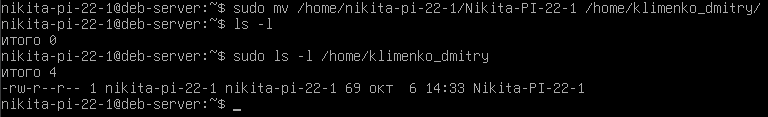


Рисунок 28 – Перенос файла в каталог другого пользователя

3.5. Редактирование перенесенного файла

Выполним вход в систему от имени другого пользователя с помощью команды su - klimenko\_dmitry. Откроем файл при помощи редактора nano, но редактирование данного файла будет запрещено. Тогда по описанному алгоритму в пункте 3.4 выдадим права администратора для пользователя klimenko\_dmitry. Теперь можно изменить права доступа к файлу, добавив возможность записи в него для всех пользователей, с помощью команды sudo chmod a+w Nikita-PI-22-1. Добавим строчку "Test page edited by user Клименко Дмитрий Сергеевич" и сохраним изменения. Результат измененного файла и новых прав доступа представлены на рисунке 29.

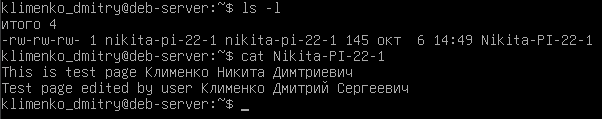
****

Рисунок 29 – Редактирование файла

Переместим файл обратно в каталог пользователя nikita-pi-22-1 с помощью ранее использованной команды sudo mv. Процесс перемещения представлен на рисунке 30.

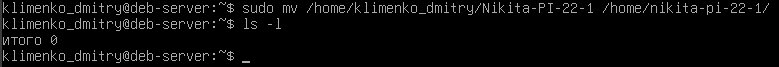


Рисунок 30 – Перемещение файла

3.6. Зайти в систему снова пользователем <ВашеИмяГруппа>, сделать файл исполняемым и запустить.

Сменим пользователя с помощью ранее описанной команды su - nikita-pi-22-1. Откроем файл в редакторе nano и добавим строчку #!/bin/bash в начало файла. Таким образом, мы говорим, что данный файл будет исполняемым скриптом и будет использовать интерпретатор bash. Результат редактированного файла представлен на рисунке 31.

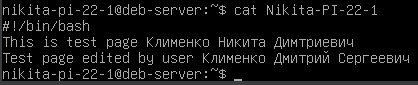


Рисунок 31 – Исполняемый файл

Перед запуском файла также необходимо изменить строчки 2 и 3 из рисунка x, так они не обладают никакими командами, а следовательно, не будут интерпретироваться и файл будет выдавать ошибку "Команда не найдена". Добавим команду echo, для вывода данных строк. Итоговый файл представлен на рисунке 32.

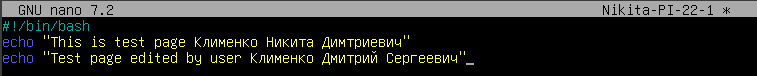
****

Рисунок 32 – Редактирование исполняемого файла

Теперь, чтобы запустить файл, необходимо изменить права доступа на запуск файла для его создателя, например. Для этого воспользуемся командой chmod u+x Nikita-PI-22-1. И для запуска файла просто прописываем его название. Все вышеописанные действия представлены на рисунке 33.

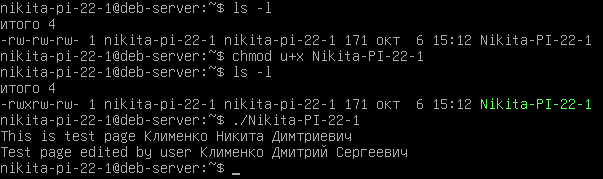


Рисунок 33 – Запуск исполняемого файла

**4. Архивация и поиск**

Таблица 4 – Задание по варианту

|  |  |
| --- | --- |
| **Вариант** | **Задание** |
| 3 | Создать архив arh3.tar.bz, состоящий из нескольких файлов. Вывести список файлов архива. Вывести содержимое файлов архива, без его распаковки. Распаковать архив.  Найти и удалить все файлы в текущем каталоге (без обработки подкаталогов), выдавая запрос на удаление каждого файла. Продемонстрировать существование файлов до выполнения и их отсутствие после выполнения команды.  Отсортировать список файлов указанного каталога и его подкаталогов в обратном порядке. |

1. Работа с архивом.

Все действия будут выполняться под пользователем user. Создадим архив arh3.tar.bz (.tar – расширение архива, .bz – формат сжатия) и добавим в него уже имеющиеся файлы file\_1.txt, file\_2.txt, file\_3.txt.

Для создания архива используется команда tar – это стандартная утилита, с помощью которой выполняется архивирование файлов. Будем использовать следующие опции: -c (создание архива), -v (демонстрирует подробную информацию о процессе работы), -j (сжатие производится с помощью Bzip), -f (указывает, что следующий аргумент – имя архива).

На рисунке 34 представлены файлы, которые будут использоваться для архивирования. Их общий вес составляет 246 + 248 + 265 = 759 байт.

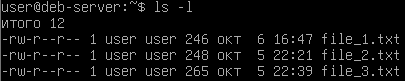


Рисунок 34 – Файлы для архивации

Теперь пропишем команду для создания архива и сравним его вес с первоначальным весом файлов.

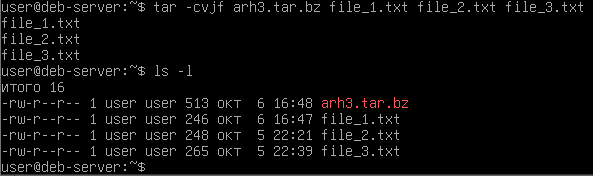


Рисунок 35 – Созданный архив

Как видно из рисунка 35 итоговый вес архива составил 513 байт, что меньше суммы исходных размеров файлов.

Для вывода списка файлов архива используется таже команда tar, но с опцией -t (показывает содержимое архива). Файлы архива представлены на рисунке 36.

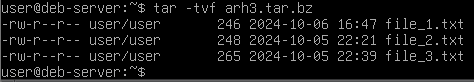


Рисунок 36 – Содержимое архива

Для просмотра содержимого конкретного файла архива можно добавить опцию -O (данная опция указывает на вывод содержимого файла на стандартный вывод, то есть в терминал, вместо записи и сохранения в файл). На рисунке 37 представлено содержание каждого файла в архиве.

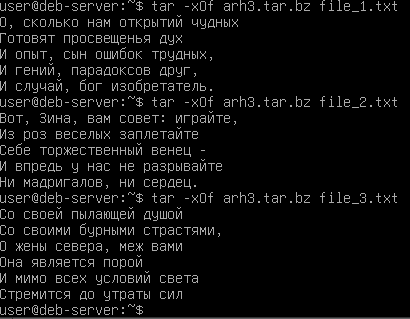


Рисунок 37 – Вывод содержимого отдельного файла архива

Чтобы распаковать архив используется опция -x, и можно добавить опцию -C, чтобы распаковать архив в определенный каталог (для этого создадим новый каталог с помощью команды mkdir <name\_dir>). Данные действия представлены на рисунке 38.

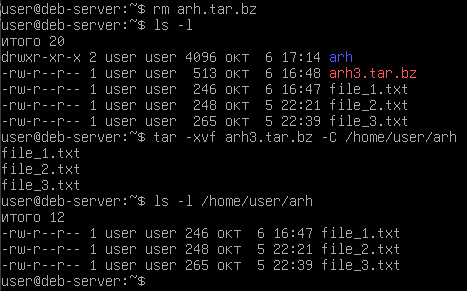


Рисунок 38 – Распаковка архива

Как видно из рисунка x архив был успешно распакован и соответственно размеры исходных файлов совпадают с разархивированными.

2. Удаление файлов.

Удаление файлов будет производится в каталоге arh, его содержимое продемонстрировано на рисунке x. Производить удаление всех файлов будем с помощи команды rm, используя опцию -i (перед удалением будет запрашивать разрешение на удаление), для согласия на удаление необходимо ввести "y". Чтобы выбрать сразу все файлы достаточно указать символ "\*". Действия продемонстрированы на рисунке 39.

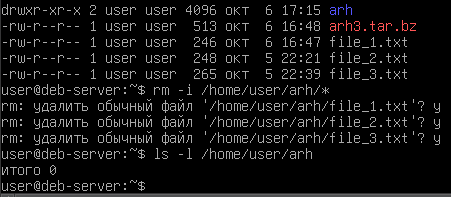


Рисунок 39 – Пример удаления файлов

Из рисунка 39 видно, что после удаления каталог arh стал пуст, следовательно удаление выполнилось успешно.

3. Рекурсивная сортировка.

Чтобы получить список файлов в любом из каталогов, можно использовать команду ls с опциями -l (вывод подробной информации о файлах) и -R (рекурсивно заходит в подкаталоги текущего каталога). Затем с помощью символа "|" можно перенаправить стандартный поток вывода в поток ввода для команды sort с опцией -r, которая выполняет сортировку в обратном порядке. Результат сортировки представлен на рисунке x.

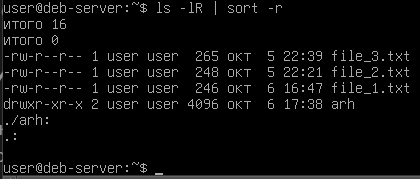


Рисунок 40 – Обратная сортировка

**5. Создание демона**

Задача: написать демон, представляющий собой программу для отдыха глаз. Демон будет показывать уведомление о начале отдыха раз в заданный промежуток времени и уведомление об окончании отдыха через заданный промежуток времени. Временные промежутки задаются случайным образом.

Daemon – это программа, которая запускается в фоновом режиме автоматически (то есть без взаимодействия с терминалом или пользовательским интерфейсом).

Скрипт демона будет написан на языке bash. Для этого создадим файл (.sh) и пропишем в нем скрипт через редактор nano. Реализованный скрипт представлен на рисунке 41.

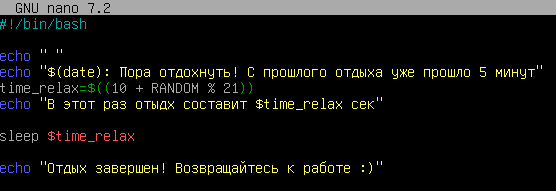


Рисунок 41 – Скрипт глазного таймера

После сохранения скрипта необходимо сделать данный файл исполняемым, выдав право на исполнение с помощью команды chmod u+x eye\_timer.sh.

Затем нужно создать сервис для запуска демона. Сервисы создаются в директории /etc/systemd/system/. С помощью редактора nano создаем файл сервиса с именем eye\_timer.service. Пример сервиса представлен на рисунке 42.

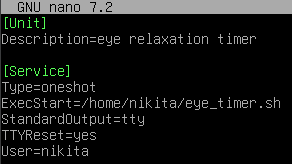


Рисунок 42 – Созданный сервис

Описание рисунка 42:

[Unit] – секция для описания юнита.

Description – краткое описание юнита.

[Service] – секция с параметрами запуска службы.

Type – тип службы, в нашем случае oneshot (используется для разового запуска скрипта).

ExecStart – запуск службы, указывается главный исполняемый файл, ради которого создан демон.

StandardOutput – управление выводом стандартного потока (tty – вывод осуществляется в терминал).

TTYReset – сброс терминальных настроек перед запуском сервиса (предотвращает случайное форматирование текста).

User – пользователь, от чьего имени будет запущен юнит (то есть этот пользователь будет запускать исполняемый файл, указанный в ExecStart).

После создания сервиса необходимо создать таймер, который будет запускать этот сервис через определенный промежуток времени. Таймеры создаются в той же директории, что и сервисы, но уже с расширением .timer. Пример созданного таймера представлен на рисунке 43.

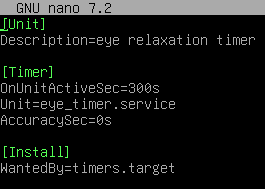


Рисунок 43 – Созданный таймер

Описание рисунка 43:

[Timer] – секция для задания времени активации.

OnUnitActiveSec – отсчет относительно момента запуска юнита.

Unit – ссылка на сервис, который будет запущен таймером.

AccuracySec – точность таймера.

[Install] – секция для установки таймера.

WantedBy – определяет таргет, при активации которого будет запущен таймер.

Чтобы юнит был правильно создан, необходимо выполнить команду sudo systemctl daemon-reload. Эта команда заставляет systemd перечитать все файлы конфигурации юнитов. После того как systemd прочитает наши юниты, нужно запустить таймер с помощью команды sudo systemctl start eye\_timer.timer, после запустить сервис, чтобы таймер начал отсчет для повторного запуска демона. Результат автоматического запуска демона представлен на рисунке 44.

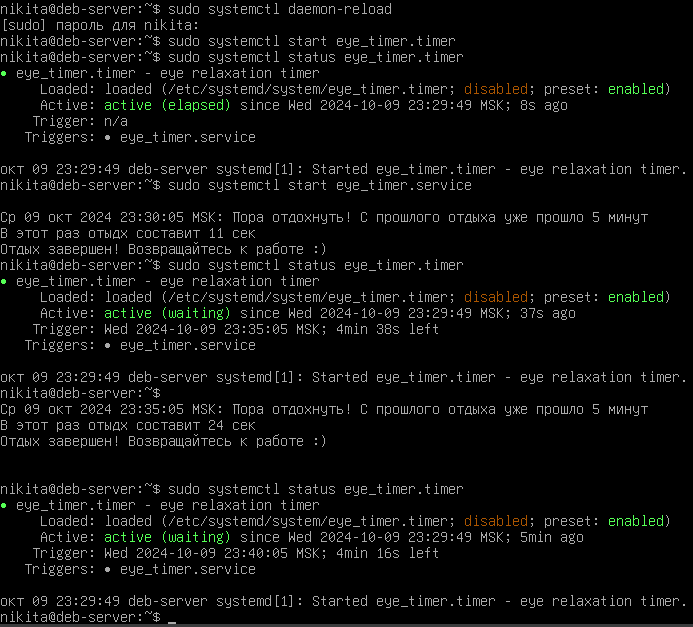


Рисунок 44 – Пример работы демона

Как видно из рисунка 44 первый запуск демона был принудительный через команду sudo systemctl start eye\_timer.service в 23:30:05, но после этого демон начал автоматически срабатывать каждые пять минут (23:35:05, 23:40:05), это же можно увидеть и через команду sudo systemctl status eye\_timer, что у него после каждого запуска демона обновляется поле trigger.

**Контрольные вопросы**

1. Что такое файловая система?

Файловая система – это часть операционной системы, которая обеспечивает интерфейс пользователю при работе с данными, хранящимися на дисках, и предоставляет возможность совместного использования файлов несколькими пользователями и (или) процессами одновременно.

2. Права доступа к файлам.

Каждому файлу и каталогу в системе может быть присвоено три типа владельцев, а именно пользователь, группа, другой (все).

Владелец-пользователь – это, как правило, тот, кто создал файл или каталог.

Владелец-группа – это пользователи, обладающие равными правами на доступ к конкретному файлу.

Владелец-Другой – это тот пользователь, который не создавал конкретный файл и не входит в группу, которой разрешено выполнять с файлом какие-либо действия.

Для изменения прав доступа к файлам/каталогам используется команда chmod (change mode), её общий формат: chmod <права\_доступа> <имя\_файла>. У данной команды два режима работы: числовой и символьный. В числовом режиме права доступа кодируются трехзначным числом, каждая из цифр которого от 0 до 7, последовательно для пользователя, группы, всех остальных, пример представлен в таблице 5.

Таблица 5 – Права доступа к файлу

|  |  |  |
| --- | --- | --- |
| **Код** | **Тип прав доступа** | **Отображение прав доступа** |
| 0 | Доступ запрещен | --- |
| 1 | Разрешено выполнение (запуск) файла | --x |
| 2 | Разрешена запись | -w- |
| 3 | Разрешено исполнение и чтение | -wx |
| 4 | Разрешено чтение | r-- |
| 5 | Разрешено чтение и выполнение | r-x |

Окончание таблицы 5

|  |  |  |
| --- | --- | --- |
| 6 | Разрешено чтение и запись | rw- |
| 7 | Разрешено чтение все | rwx |

3. Что такое символическая ссылка?

Символическая ссылка – это специальный файл в файловой системе, в котором вместо пользовательских данных содержится путь к файлу, открываемому при обращении к данной ссылке.

4. Что такое жесткая ссылка?

Жесткая ссылка является своего рода синонимом для существующего файла. Когда создается жесткая ссылка, то создается дополнительный указатель на существующий файл, но не копия файла. Жесткие ссылки выглядят в файловой структуре как еще один файл.

5. Команда поиска в Linux. Основные сведения.

Команда find используется поиска файлов и каталогов на основе специальных условий. Ее можно использовать в различных обстоятельствах, например, для поиска файлов по разрешениям, владельцам, группам, типу, размеру и другим подобным критериям. Общий формат: find <каталог> <опции> <критерий> <шаблон>.

6. Перечислите основные команды работы с каталогами.

pwd – отображение текущего каталога;

cd – перемещение по каталогам;

mkdir – создание нового каталога;

rmdir – удаление каталога.

7. Чем отличается вывод команд ls -F и ls -la?

ls -ls – отображает в подробном формате (будет отображаться владелец, группа, дата создания, размер и другие параметры) все файлы, включая скрытые.

ls -F – показывает тип объекта, к каждому объекты будет добавлен один из специализированных символов "\*/=>@|".

8. С помощью какой команды можно переместить файл в другой каталог?

Команда mv перемещает файл в новое место. Она также может использоваться для переименования файлов. Общий формат: mv <источник> <назначение>.

9. Куда вы переходите, выполнив команду cd без параметров?

При выполнении команды cd без параметров рабочей папкой будет выбран домашний каталог.

10. Как осуществить просмотр каталогов и их содержимого?

Команда ls отобразит каталоги нужной директории, при задании опции -l можно получить подробную информацию о файлах/каталогах, а опция -R позволит посмотреть каталог рекурсивно, то есть со всеми содержащимися в нем каталогами.

11. Как осуществить создание нового каталога и необходимых каталогов рекурсивно?

Для рекурсивного создания каталога используется команда mkdir с параметром -p (создаст родительские каталоги при их отсутствии).

12. Как осуществить рекурсивное копирование всех файлов из одного каталога в другой?

Для рекурсивного копирования одного файлов одного каталога в другой необходимо использовать команду cp с опцией -r.

13. Как рекурсивно удалить все файлы и подкаталоги в определенном каталоге?

Для рекурсивного удаления файлов и каталогов определенного каталога необходимо использовать команду rm с параметром -r.

14. Перечислите основные ключи команды ls с их назначением.

Основные опции команды ls приведены в таблице 6.

Таблица 6 – Опции команды ls

|  |  |
| --- | --- |
| Опция | Назначение |
| -a | Отображать все файлы, включая скрытые. |
| -d | Вывод только директорий. |
| -F | Показывать тип объекта. |

Окончание таблицы 6

|  |  |
| --- | --- |
| -g | Показывать подробную информацию, но кроме владельца файла. |
| -G | Не выводить имена групп. |
| -h | Выводить размеры папок в удобном для чтения формате |
| -l | Выводить подробный список, в котором будет отображаться владелец, группа, дата создания, размер и другие параметры. |
| -R | Рекурсивно отображать содержимое поддиректорий. |
| -s | Выводить размер файла в блоках. |
| -X | Сортировать по алфавиту. |
| -r | Обратная сортировка. |

15. Команды tee и cat. Назначение и применение. Чем cat отличается от more и less?

Команда tee – считывает данные из стандартного ввода и записывает их одновременно в файл и на стандартный вывод. Используется для создания копий вывода команд, например, при записи вывода в файл, не прерывая отображение в терминале.

Команда cat – выводит содержимое файла на экран или соединяет несколько файлов в один. Показывает все содержимое файла сразу, без возможности постраничного просмотра.

Команда more – показывает содержимое файла постранично, то есть присутствует возможность пролистывания выводимого содержимого (но только вниз).

Команда less – более мощная версия, по сравнению с more, так как позволяет листать содержимое как вперед, так и назад и работает быстрее для больших файлов, так как загружает их частями.

16. Что такое демон?

Демон – это процесс, который работает в фоновом режиме без прямого участия пользователя.

17. Для чего в операционной системе Linux применяется подсистема systemD?

Система инициализации (systemD) – это система в Linux, которая подготавливает к работе операционную система. Система инициализации запускается ядром как первый процесс операционной системы. Затем, этот первый процесс, запускает все остальные процессы. Также, при выключении компьютера, данная система инициализации занимается остановкой всех процессов.

18. Какие типы юнитов вы знаете?

Все задачи, которые выполняет systemD описываются в специальных файлах, которые и называются юниты. Юниты бывают разных типов:

- Service – описывает сервис (службу) или скрипт, то есть все, что можно запустить;

- Target – группирует юниты, с его помощью можно объединить две группы и запускать как одну;

- Timer – определяет таймер, то есть службы могут запускаться по определенному расписанию, или с задержкой.

19. С помощью каких команд осуществляется управление демонами?

Основные команды для управления демонами представлены в таблице 7.

Таблица 7 – Команды для управления демонами

|  |  |
| --- | --- |
| **Команда** | **Назначение** |
| systemctl start <unit.service> | Осуществляет запуск службы (запуск одного или нескольких процессов, которые будут работать в одной службе). |
| systemctl stop <unit.service> | Остановка службы, следствием чего буде и остановка всех связанных со службой процессов. |
| systemctl status <unit.service> | Проверка статуса службы (выдает информацию о состоянии, иерархию процессов, последние логи). |
| systemctl reload <unit.service> | Перечитывании конфигураций юнитов. |

Окончание таблицы 7

|  |  |
| --- | --- |
| systemctl restart <unit.service> | Перезапуск службы, то есть остановка и последующее включение вновь. |
| systemctl enable <unit.service> | Включение автозапуска, но момент автозапуска определяется исходя из юнита. |
| systemctl disable <unit.service> | Выключение автозапуска службы. |
| systemctl cat <unit.service> | Вывод содержимого файла юнита. |

20. Чем корутина отличается от потоков?

Кроутины и потоки – это разные подходы к многозадачности. Основное различие заключается в следующем:

- Потоки создаются операционной системой, каждый поток выполняется независимо и может использовать несколько процессорных ядер одновременно. Потоки работают параллельно, разделяя общие ресурсы и память.

- Корутины представляют собой функции, которые могут приостанавливать выполнение (сохраняя свое состояние) и возобновлять его позже. В отличие от потоков, они управляются на уровне программы и не требуют участия операционной системы. Они реализуют кооперативную многозадачность (функция приостанавливает выполнение сама, а не по решению ОС).

21. Почему корутины должны быть реализованы как выделенная языковая возможность?

Корутины требуют выделенной поддержки на уровне языка программирования, потому что их реализация напрямую затрагивает поведение функций, управление стеками вызовов и управление состояниями программы. Операции по приостановке и возобновлению выполнения, а также сохранение контекста требуют изменений в способе, которым язык управляет функциями и памятью.

22. Какая польза от корутины?

Корутины полезны для написания асинхронного и неблокирующего кода, который выглядит и ведет себя как обычный синхронный код. Они позволяют разработчикам избегать сложностей с обратными вызовами, упрощая управление состояниями программы. Корутины также позволяют более эффективно использовать системные ресурсы, так как не требуют создания потоков и затрат на переключение контекста. Это делает их особенно полезными для задач, которые предполагают ожидание, например, сетевые запросы, когда использование потоков было бы излишним и менее эффективным.

23. Перечислите команды, используемые вами при выполнении данной лабораторной работы, и кратко поясните назначение каждой из них.

Таблица 8 – Перечень используемых команд

|  |  |
| --- | --- |
| Команда | Назначение |
| ls | Вывод содержимого каталога и информации о файлах. |
| su | Смена пользователя. |
| cd | Изменение рабочего каталога. |
| sudo | Позволяет строго определенным пользователям выполнить указанные команды с административными привилегиями. |
| adduser | Создание новой учетной записи (автоматически настраивает нового пользователя с настройками конфигурации по умолчанию). |
| cat | Читает данные из файла или стандартного ввода и выводит их на экран. |
| touch | Используется для создания пустых файлов или для установки времени последнего изменения файла. |
| ln | Создание жестких или символических ссылок. |
| cp | Осуществляет копирование файлов. |
| mv | Осуществляет перемещение или переименования файлов. |
| rm | Используется для удаления файлов. |
| mkdir | Создание новой директории. |
| echo | Вывод строки текста на стандартное устройство вывода. |
| chmod | Изменение прав доступа к файлам и каталогам. |
| tar | Команда для работы с архивами (создание, распаковка и прочее). |

**Вывод**

В ходе выполнения данной лабораторной работы я приобрел опыт работы с файлами и каталогами в ОС Linux, освоил настройку прав на доступ к файлам и каталогам.