**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 | Первеева Е. Ю. |
| Руководитель | |  |  |  |
| канд. техн. наук, доцент  ученая степень, ученое звание | |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись, дата | Кургасов В.В. |

Липецк, 2024 г.

Оглавление

[Цель работы 3](#_Toc179897226)

[Ход работы 4](#_Toc179897227)

[1. Общая часть 4](#_Toc179897228)

[2. Файлы и каталоги 15](#_Toc179897229)

[3. Пользователи и группы 19](#_Toc179897230)

[4. Архивация и поиск 21](#_Toc179897231)

[5. Создание демона 25](#_Toc179897232)

[Контрольные вопросы 27](#_Toc179897233)

[Вывод 31](#_Toc179897234)

# **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. Ознакомиться со структурой системных каталогов ОС Linux на рабочем месте. Изучить стандарт (Filesystem Hierarchy Standard).

**FHS** — это стандарт, описывающий структуру и содержание основных каталогов и подкаталогов в Unix-подобных операционных системах, таких как Linux. Он описывает, какие файлы и каталоги должны располагаться в определённых местах системы, обеспечивая согласованную структуру файловой системы.

Основные системные каталоги:

/ (корневой каталог) – начальная точка всей файловой системы, в нём находятся все остальные файлы и каталоги.

/bin - содержит основные программы доступные всем пользователям и необходимые для минимальной работы системы. Пример: ls, cp, mv, rm.

/etc - содержит конфигурационные файлы системы и программ.

/root - домашний каталог пользователя root (суперпользователя).

/sbin - системные бинарные файлы и утилиты, предназначенные для использования администраторами системы.

/srv - директория для хранения данных, которые обслуживаются или предоставляются сетевыми сервисами.

/tmp - временные файлы, которые удаляются после перезагрузки системы.

/usr - пользовательские программы, библиотеки и документация.

1.2. Изучить и привести в отчёте перечень основных каталогов с указанием их назначения.

Просмотрим перечень основных каталогов корневой директории (/) при помощи команды ls (Рисунок 1).

Назначение каталогов:

Рисунок 1 - Перечень основных каталогов корневой директории

**/bin** - основные программы (бинарные файлы), доступные для всех пользователей и необходимые для минимальной работы системы. В него входят такие команды, как ls, cp, mv, rm и т.д.

**/boot** - файлы, необходимые для загрузки системы, такие как ядро и загрузочный загрузчик (GRUB).

**/dev** - специальные файлы устройств, представляющие оборудование системы, такие как жесткие диски, порты и устройства ввода/вывода.

**/etc** - конфигурационные файлы (настройки) системы и программ.

**/home** - домашние каталоги пользователей, где они могут хранить свои личные файлы и настройки.

Файл **initrd.img** необходимый на этапе загрузки системы для создания временной файловой системы, необходимой при подготовке окружения для загрузки ядра и монтирования корневой файловой системы.

Файл **initrd.img.old** — это предыдущая версия initrd.img, используемая для восстановления в случае проблем с текущим ядром.

**/lib** - библиотеки, необходимые для работы программ и исполняемых файлов в каталогах /bin и /sbin.

**/lib64 -** предназначен для хранения 64-разрядных библиотек, необходимых для работы 64-битных программ и системы в целом.

**/lost+found -** предназначен для восстановления файлов, которые были повреждены или потеряны из-за сбоев файловой системы.

**/media -** каталог для монтирования съемных носителей, таких как CD/DVD, USB-диски и другие.

**/mnt** - временная точка монтирования, используемая для ручного монтирования файловых систем и устройств администратором.

**/opt** - дополнительные или опциональные программы и пакеты, которые не входят в стандартные пакеты системы.

**/proc** - виртуальная файловая система, содержащая информацию о текущих процессах и состоянии системы.

**/root** - домашний каталог пользователя root (администратора).

**/run** - директория для временных файлов, используемых процессами во время работы системы и эти данные не сохраняются после перезагрузки системы.

**/sbin** - системные программы и утилиты, которые используются администраторами для обслуживания системы.

**/srv** - директория для данных, обслуживаемых различными сервисами (например, данные веб-серверов или баз данных).

**/sys -** является виртуальной файловой системой, котораяпредоставляет информацию о подключенных устройствах, драйверах и некоторых параметрах ядра.

**/tmp** - временные файлы, которые создаются программами и процессами. Каталог обычно очищается при перезагрузке системы.

**/usr -** содержит программы, библиотеки и документацию, используемые пользователями.

**/var -** файлы, которые часто изменяются во время работы системы, такие как журналы (логи), временные файлы, кэш и базы данных.

Файл **vmlinuz –** сжатое ядро Linux, используемое для загрузки операционной системы.

Файл **vmlinuz.old -** резервная копия предыдущей версии ядра. Этот файл создается автоматически при обновлении ядра, чтобы обеспечить возможность возврата к предыдущей версии в случае проблем с новым ядром.

1.3. Зайти в терминал под root.

Для получения прав администратора (root) воспользуемся командой su – (Рисунок 2). Символ # после имени пользователя указывает на получение прав администратора (root).



Рисунок 2 - Вход в терминал под пользователем root

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

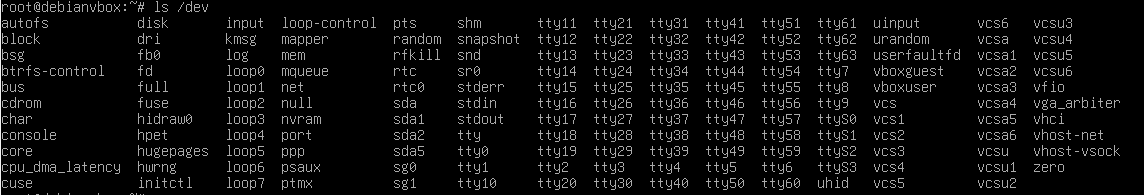
С помощью команды ls /dev посмотрим содержимое каталога файлов физических устройств (рисунок 3).

Рисунок 3 - Содержимое каталога dev

Назначение файлов:

**сdrom** — файл позволяет пользователям и приложениям взаимодействовать с CD/DVD-приводом.

**сonsole -** системная консоль, предоставляющая интерфейс для взаимодействия с ядром операционной системы и выполняемыми процессами.

**disk** - специальный подкаталог, который содержит символические ссылки на блочные устройства.

**fb0** — это первый кадровый буфер, абстрактный слой между программным обеспечением и графическим оборудованием.

**fd** - представляет дисковод для гибких дисков (дискет).

**full** - символьное устройство, которое всегда возвращает ошибку ENOSPC (нет места на устройстве), когда вы пытаетесь записать в него данные. Оно используется для тестирования программ на обработку ошибок.

**loop0-7** - позволяют монтировать файлы как виртуальные блочные устройства, отображая их как отдельные файловые системы.

**mem** - представляет собой специальный файл, который является образом физической памяти компьютера, позволяя получать доступ к содержимому памяти напрямую.

**null** — специальное устройство, часто называемое "черной дырой" для данных. Все, что вы записываете в /dev/null, будет проигнорировано и потеряно.

**ptmx** — специальное устройство, используемое для создания парных терминалов (pseudo-terminal master). Это устройство играет важную роль в взаимодействии между терминальными программами и позволяет создавать виртуальные терминалы для различных приложений.

**pts** - представляет собой виртуальный каталог для временных файлов, связанных с псевдотерминалами.

**random** является псевдоустройством, используемое для генерации случайных чисел, основанных на энтропии, собранной из различных источников в системе.

**rfkill** - интерфейс для управления радиопередатчиками (WiFi и Bluetooth), позволяя включать и отключать их.

**rtc** - предоставляет интерфейс для работы с реальным временем (Real-Time Clock, RTC) на компьютере.

**sda** - специальное устройство, которое обозначает жесткий диск в системе, использующий интерфейс SATA (Serial ATA) или SCSI (Small Computer System Interface). Буквы после sda указывают на новые устройства, а числа после них обозначают номера разделов.

**sr0 -** привод оптических дисков, таких как CD или DVD.

**stderr** - стандартный поток ошибок, который используется для вывода сообщений об ошибках и предупреждений.

**stdin** - стандартный поток ввода, через который программа получает данные от пользователя или другой программы.

**stdout** - стандартный поток вывода, через который программа отправляет данные пользователю.

**tty** - специальный файл устройства, представляющий текущий терминал, в котором запущен процесс.

**ttyS** - устройства для работы с последовательными портами, используемыми для подключения модемов и другого оборудования.

**zero** - специальное устройство, которое генерирует поток нулевых байтов. Он используется в различных сценариях, связанных с управлением памятью и файлами.

1.5. Перейти в директорию пользователя root. Просмотреть содержимое каталога. Просмотреть содержимое файла vmlinuz. Просмотреть и пояснить права доступа к файлу vmlinuz.

Для просмотра содержимого каталога пользователя root воспользуемся командой ls (Рисунок 4)



Рисунок 4 - Содержимое каталога root

Содержимое файла vmlinuz представляет собой сжатый образ ядра операционной системы Linux, которое загружается при старте системы. Он содержит бинарные данные, поэтому его содержимое нельзя напрямую просмотреть как текстовый файл.

При загрузке компьютер читает файл vmlinuz, распаковывает его и загружает ядро Linux в память, чтобы система могла начать работу. Он содержит всю логику ядра, которая управляет процессами, памятью, устройствами и всеми основными аспектами операционной системы.

Для просмотра прав доступа файла потребуется команда ls -l /boot/vmlinuz-6.1.0-25-amd64 (Рисунок 5)



Рисунок 5 - Права доступа vmlinuz

Строка -rw-r--r—показывает права доступа к файлу vmlinuz. Первый символ указывает что это обычный файл, а так же что файл доступен для чтения всем пользователям, а администратор (root) имеет право на запись. Однако ни у кого, включая администратора, нет прав на исполнение.

1.6. Создать нового пользователя user.

Создание нового пользователя осуществляется при помощи команды sudo adduser <user\_name>. Далее необходимо указать всю необходимую информацию о новом пользователе (Рисунок 6).

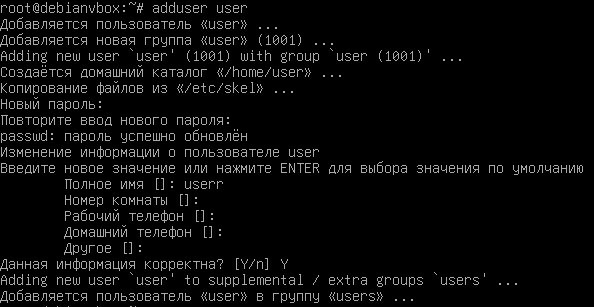


Рисунок 6 - Создание нового пользователя

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

Командами touch, cat и nano создадим три файла 1.txt, 2.txt и 3.txt (Рисунок 7).

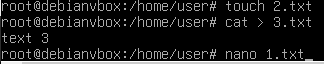


Рисунок 7 - Пример создания файлов

Командой ls с ключом -l выведем подробный список файлов и их атрибутов c правами доступа (Рисунок 8).

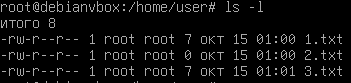


Рисунок 8 - Права доступа к файлам

Пояснение прав доступа для созданных файлов:

1. "-": признак обычного файла, "d" (для каталога), "l" (для символической ссылки).
2. Владелец: имеет право на чтение ("r"), запись ("w"), но не имеет право на исполнение.
3. Группа: имеет право только на чтение.
4. Все остальные пользователи: имеют право только на чтение.

1.8. Перейти в директории пользователя root. В отчёте описать результат.

Командой su - переключимся на пользователя root, командой ls -a выведем содержимое директории пользователя root (Рисунок 9).



Рисунок 9 - Вывод содержимого директории root

Содержимое директории:

.bash\_history: содержит историю команд, которые были выполнены в командной строке.

.bashrc: сценарий, который выполняется каждый раз, когда запускается интерактивная оболочка Bash.

.profile: сценарий (скрипт), который выполняется при входе пользователя в оболочку (shell). Используется в основном для настройки переменных окружения и запуска программ для login shell.

.ssh: директория, которая содержит файлы, связанные с SSH (Secure Shell). Она обычно включает файлы аутентификации, такие как ключи и конфигурации для подключения к удалённым системам.

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

Командой chmod a+rwx (полные права доступа на файл для владельца, группы и остальных пользователей) изменим права доступа на файл 1.txt (рисунок 10).

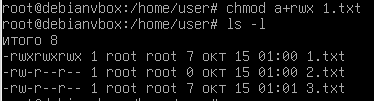


Рисунок 10 - Изменение прав доступа

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

Жёсткие ссылки создаются с помощью команды ln. Если файл, на который указывает жёсткая ссылка, будет удалён, данные останутся доступны через жёсткую ссылку, поскольку она продолжает ссылаться на ту же область данных.

Символические ссылки создаются с помощью команды ln -s и представляют собой отдельные файлы, которые содержат путь к исходному файлу. Если оригинальный файл будет удалён, символическая ссылка перестанет работать, так как она больше не будет указывать на существующий объект.

Создание символической и жесткой ссылки на файл 2.txt (Рисунок 11).

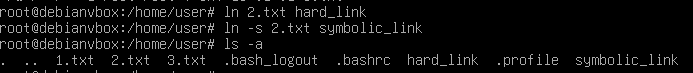


Рисунок 11 - Создание ссылок

1.11. Создать каталог new в каталоге пользователя user.

Командой mkdir создадим новый каталог new в директории пользователя user (рисунок 12).



Рисунок 12 - Создание нового каталога

1.12. Скопировать файл 1.txt в каталог new.

Командой cp скопируем файл 1.txt в каталог new (рисунок 13).

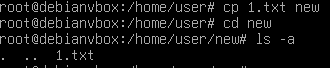


Рисунок 13 - Копирование файла

1.13. Переместить файл 2.txt в каталог new.

Командой mv переместим файл 2.txt в каталог new (рисунок 14).

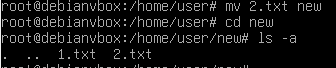


Рисунок 14 - Перемещение файла

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

Командой chown изменим владельца файла 3.txt и каталога new на root (рисунок 15).

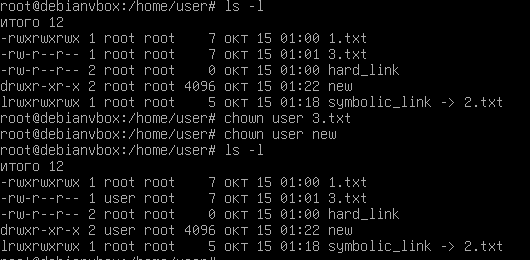


Рисунок 15 - Изменение владельца файла

1.15. Удалить файл 1.txt в каталоге new.

Для удаления файла 1.txt в каталоге new используем команду rm (рисунок 16).

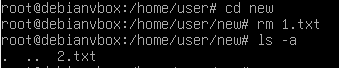


Рисунок 16 - Удаление файла

1.16. Удалить каталог new.

Для удаления каталога new используем команду rm -r (рекурсивно, для удаления вместе с содержимым). Пример использования команды представлен на рисунке 17.



Рисунок 17 - Удаление каталога

# 2. Файлы и каталоги

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

Создадим 3 текстовых файла редакторами vim, mc и команды tee, в файлы внесём текст стихотворения. Пример создания через vim представлен на рисунке 18, через mc на рисунке 19, через tee на рисунке 20.

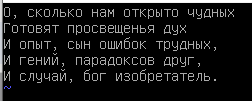


Рисунок 18 - Пример создания файла через редактор vim

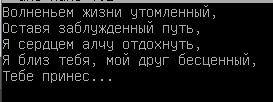


Рисунок 19 - Пример создания файла через редактор mc

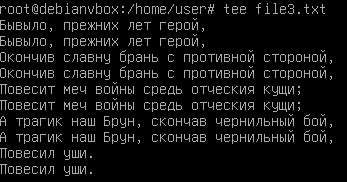


Рисунок 20 - Пример создания файла через редактор tee

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 командой cp с внесением в копию некоторых изменений. Ссылки создаются командой ln, символические ссылки – ей же, но с ключом -s. На рисунке 21 представлено задание, соответствующее варианту.

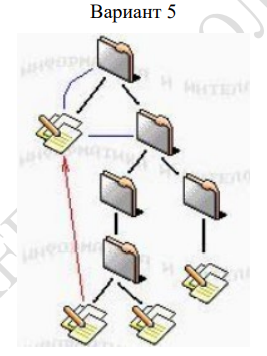


Рисунок 21 - Задание по варианту

Создание необходимых каталогов осуществляется с помощью команды mkdir <name\_dir>, опция -p позволяет создавать дополнительно родительские каталоги (Рисунок 22).



Рисунок 22 - Создание каталогов

Создадим один файл 1.txt, используя команду touch, а затем с помощью команды cp скопируем его во все остальные папки и при помощи оператора перенаправления ">>" допишем в него некоторые строки (Рисунок 23)

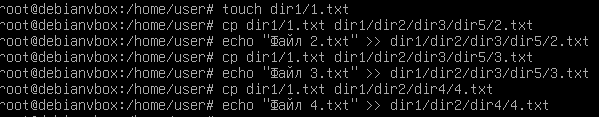


Рисунок 23 - Создание файлов в каталогах

Создадим необходимые жёсткие ссылки командой ln и мягкие ссылки командой ln -s. Все они ссылаются на файл 1.txt (Рисунок 24)

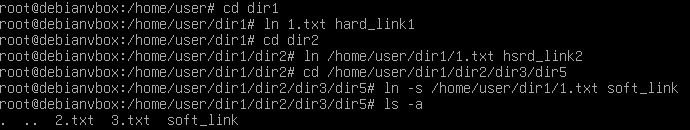


Рисунок 24 - Создание мягких и жёстких ссылок

2.3. Вывести на консоль полную информацию о созданных файлах.

Чтобы вывести полную информацию о созданных файлах и директориях необходимо воспользоваться командой tree (Рисунок 25).



Рисунок 25 - Процесс создания структуры каталогов

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.

С помощью команды cat выведем содержимое файла 1.txt через его имя, показав, что он доступен по имени. После чего выведем содержимое этого же файла через одну из жестких ссылок и через мягкую ссылку (рисунок 26).

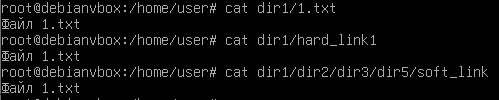


Рисунок 26 - Доступ к файлам через их имена и ссылки

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

С помощью команды rm удалим файл 1.txt (так как на него ведут все ссылки), после чего попробуем получить доступ к содержимому через символическую и жесткую ссылки (Рисунок 27)

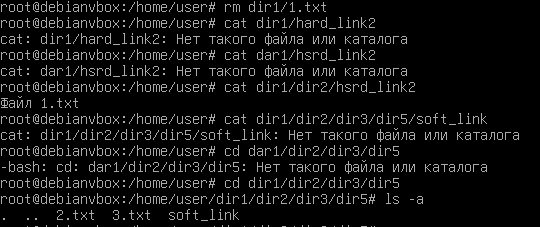


Рисунок 27 - Получение доступа к файлам через ссылки

Из рисунка 27 видно, что жесткая ссылка продолжает хранить данные файла, в отличие от символической ссылки, хотя символическая ссылка не была удалена. 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.

Командой rm -r удалим созданные каталоги и их содержимое (рисунок 28).



Рисунок 28 - Удаление вложенных каталогов

**3. Пользователи и группы**

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

Создадим пользователей LizaPI-22-1 и PerveevaYuriu командой useradd, установим им пароли командой passwd (рисунок 29).

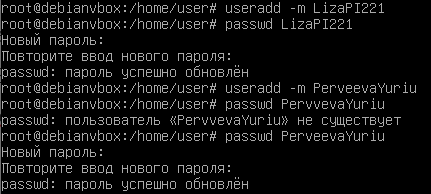


Рисунок 29 - Пример создания пользователей

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

Командой su - LizaPI221 войдём в систему под пользователем LizaPI221.

3.2.1. Создайте файл с именем: <ВашеИмяГруппа>. Откройте созданный файл в удобном вам текстовом редакторе (Vi/VIM/NANO/ Sublime) etc. Внесите в него текст: echo "This is test page <Ваше ФИО>". Сохраните изменения в файле

Создадим файл с именем LizaPI221 командой nano, внесём в него текст echo "This is test page Первеева Елизавета Юрьевна" (рисунок 30), сохраним файл.



Рисунок 30 - Создание файла в редакторе nano

3.3. Переместите файл <ВашеИмяГруппа> в домашний каталог пользователя <ФамилияИмяОтца>.

Командой mv перенесём файл LizaPI221 в домашний каталог пользователя PerveevYuriu используя права доступа пользователя root (рисунок 31).



Рисунок 31 - Перенос файла командой mv

3.4. Выполните вход в систему от имени пользователя <ФамилияИмя Отца>.

Командой su войдём в систему под пользователем PerveevaYuriu.

3.4.1. Откройте файл, перемещённый туда в пункте 3. Добавьте в файл строчку: echo "Test page edited by user" <ФамилияИмяОтца> и сохраните изменения.

Откроем файл LizaPI221 командой nano, добавим текст echo "Test page edited by user PerveevYuriu" (рисунок 32).

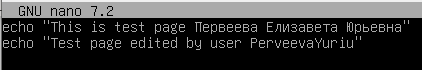


Рисунок 32 - Редактирование файла в редакторе nano

3.4.2. Переместите файл обратно в папку пользователя «ВашеИмяГруппа>. Добавьте в начало документа следующий текст: #!/bin/bash.

Командой mv перенесём файл LizaPI221 обратно в папку пользователя LizaPI221 (от пользователя root), добавим текст #!/bin/bash (рисунок 33).

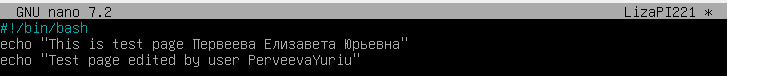


Рисунок 33 - Редактирование файла в редакторе nano

3.5. Зайдите в систему снова пользователем <ВашеИмяГруппа>, сделайте файл исполняемым и запустите.

Командой su войдём в систему снова под пользователем LizaPI221, сделаем файл исполняемым командой chmod и запустим его (рисунок 34).

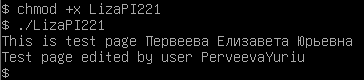


Рисунок 34 - Пример запуска исполняемого файла

# 4. Архивация и поиск

Таблица 1 – Задание по варианту

|  |  |
| --- | --- |
| **Вариант** | **Задание** |
| 5 | Создать архив arh5.bz2 с установкой степени сжатия и с сохранением исходных файлов на диске. Вывести содержимое файлов архива, без его распаковки. Сколько максимум файлов можно одновременно заархивировать в один архив? Распаковать архив с учетом перезаписи файлов при их существовании. Определить действительный размер всех каталогов и подкаталогов текущего каталога. |

Для создания архива с форматом сжатия bzip2 и установкой степени сжатия используйте команду tar с опцией -j (для сжатия bzip2). Для архивации будем использовать имеющиеся файлы, их размер: 244 +231+298=773 (Рисунок 35)

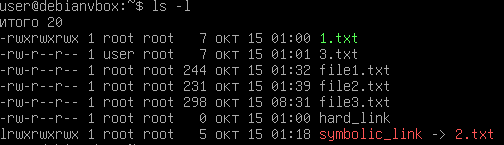


Рисунок 35 – Файлы для архивации

Пропишем команду для создания архива и добавления туда файлов: tar -cvjf arh5.bz2 --keep-old-files file1.txt file2.txt file3.txt (Рисунок 36).

-c: создание нового архива.

-v: вывод подробной информации на экран.

-j: сжатие с помощью bzip2.

-f: указание имени архива (arh5.bz2).

--keep-old-files: сохраняет исходные файлы на диске (по умолчанию файлы не удаляются, но эта опция гарантирует сохранение).

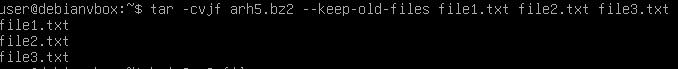


Рисунок 36– Созданный архив

Чтобы задать степень сжатия, используйте команду bzip2 и максимальную степень сжатия -9 (Рисунок 37).

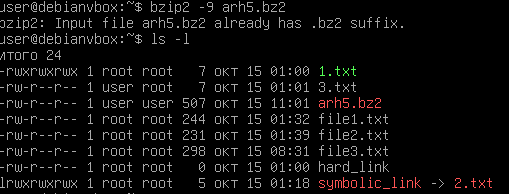


Рисунок 37 - Установка степени сжатия

Как видно из рисунка 37 итоговый вес архива составил 507 байт, что меньше суммы исходных размеров файлов.

Для вывода списка файлов архива без распаковки архива используется команда: tar -tvjf arh5.bz2 (-t: вывод списка файлов в архиве; -v: подробный режим; -j: указывает, что архив сжат с помощью bzip2; -f: указывает файл архива (в данном случае arh5.bz2).

Так же выведем содержимое одного из файлов без распаковки архива командой: tar -xOjvf arh5.bz2 имя\_файла\_в\_архиве (-x: извлечение файлов; -O: вывод содержимого извлекаемых файлов прямо в терминал) (Рисунок 38).

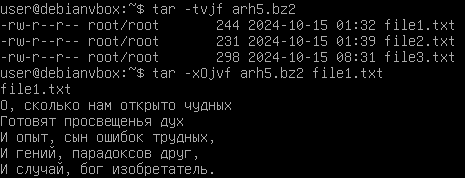


Рисунок 38 - Вывод содержимого файлов архива, без его распаковки.

Сколько максимум файлов можно одновременно заархивировать в один архив: Ограничений на количество файлов для архивации нет не у операционной системы не у команды tar. Но важно учитывать, что каждое файловое имя имеет свой размер, и файловая система ограничена количеством inodes. Хотя tar и не накладывает явных ограничений, большие архивы могут потребовать значительного объёма оперативной памяти для обработки, что может стать ограничивающим фактором на машинах с небольшим количеством памяти.

Подводя итоги теоретически, мы можем заархивировать миллионы файлов, если позволяет файловая система, объём памяти и другие ресурсы системы.

Чтобы распаковать архив используется опция -x, и можно добавить опцию -C, чтобы распаковать архив в определенный каталог (Рисунок 39)

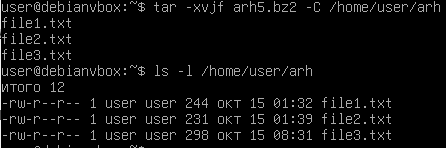


Рисунок 39 - Распаковка архива

Чтобы определить действительный размер всех каталогов и подкаталогов текущего каталога в Linux, можно использовать команду du (disk usage): du -sh \* (du: команда для оценки использования дискового пространства; -s: выводит только общий размер для каждого аргумента, а не для каждого файла внутри; -h: выводит размеры в удобочитаемом формате;\*: указывает на все файлы и каталоги в текущем каталоге.)(Рисунок 40)

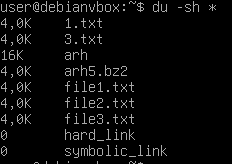


Рисунок 40 - Действительный размер всех каталогов и подкаталогов текущего каталога.

# 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(то есть без взаимодействия с терминалом или пользовательским интерфейсом).

Cоздадим файл daemon(.sh) и пропишем в нем скрипт на языке bash через редактор nano (Рисунок 41)

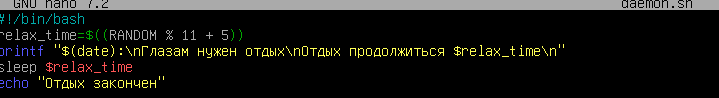


Рисунок 41 - Файл с демоном

Затем нужно создать сервис для запуска демона. Сервисы создаются в директории /etc/systemd/system/. С помощью редактора nano создаем файл сервиса с именем daemon.service (Рисунок 42)

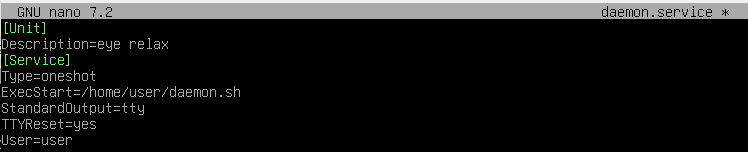


Рисунок 42 - Файл service

Описание:

[Unit] – секция для описания юнита. Description – краткое описание юнита.

[Service] – секция с параметрами запуска службы.

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

ExecStart – запуск службы, указывается исполняемый файл.

StandardOutput – tty вывод стандартного потока осуществляется в терминал.

TTYReset – сброс терминальных настроек перед запуском сервиса.

User – пользователь, от чьего имени будет запущен юнит.

Создадим таймер daemon.timer для запуска сервис через определенный промежуток времени (Рисунок 43)

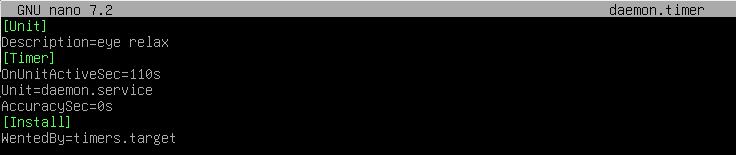


Рисунок 43 - Файл timer

OnUnitActiveSec – отсчет относительно момента запуска юнита.

Unit – ссылка на сервис, который будет запущен таймером.

AccuracySec – точность таймера.

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

Командой sudo systemctl daemon-reload перечитаем все файлы конфигурации юнитов. Запустим таймер с помощью команды sudo systemctl start daemon.timer, после этого первый раз запустим сервис принудительно командой sudo systemctl start daemon.service, далее демон автоматически будет срабатывать через определенные промежутки времени (Рисунок 44)

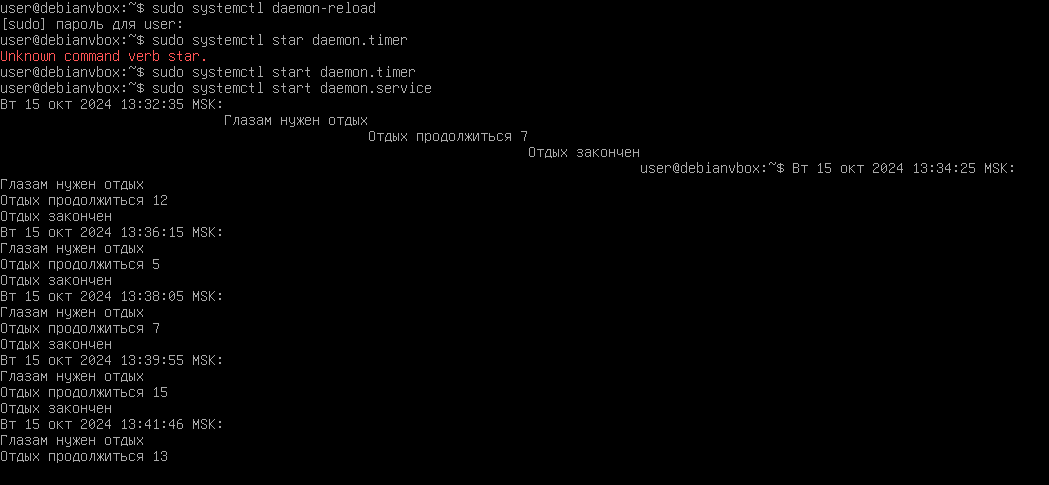


Рисунок 44 - Запуск демона

# Контрольные вопросы

1. Что такое файловая система?

Файловая система — это способ организации и хранения данных на устройствах, таких как жёсткие диски, флеш-накопители или SSD. Она управляет доступом к файлам и каталогам, определяет их структуру, имена и расположение, а также обеспечивает механизмы чтения, записи и удаления данных. Файловая система также отвечает за отслеживание метаданных файлов, таких как дата создания, размер и права доступа.

2. Права доступа к файлам.

В Linux права доступа делятся на три группы: для владельца файла, для группы и для всех остальных пользователей. Права могут быть на чтение (r), запись (w) и выполнение (x). Эти права можно управлять с помощью команд chmod и chown.

3. Что такое символическая ссылка?

Символическая ссылка (symlink) — это файл, который ссылается на другой файл или каталог, указывая на его путь.

4. Что такое жесткая ссылка?

Жесткая ссылка — это альтернативное имя для существующего файла. В отличие от символической ссылки, жесткая ссылка указывает на тот же inode, что и исходный файл, и продолжает существовать даже после удаления исходного файла.

5. Команда поиска в Linux. Основные сведения.

В Linux команда find является основным инструментом для поиска файлов и каталогов. Она позволяет искать объекты по различным критериям, таким как имя, размер, дата изменения и права доступа. Команда работает рекурсивно, проходя по директориям и поддиректориям, и поддерживает гибкие параметры для фильтрации результатов. Пример: find /path -name filename.

6. Перечислите основные команды работы с каталогами.

cd [путь] — смена текущего каталога на указанный.

ls [опции] [каталог] — отображение содержимого каталога.

mkdir [путь] — создание нового каталога.

rmdir [каталог] — удаление пустого каталога.

pwd — вывод текущего рабочего каталога (показывает полный путь).

cp [опции] источник назначение — копирование файлов и каталогов.

mv [опции] источник назначение — перемещение или переименование файлов и каталогов.

rm [опции] файл/каталог — удаление файлов и каталогов.

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

ls -F добавляет символы к именам файлов, указывающие на их тип (например, / для каталогов, \* для исполняемых файлов).

ls -la выводит подробную информацию о файлах, включая права доступа, владельца, группу, размер и дату изменения, а также отображает скрытые файлы (начинающиеся с точки).

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

Для перемещения файлов используется команда mv. Пример: mv filename /new/directory/.

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

Команда cd без параметров возвращает пользователя в домашний каталог (/home/username).

10. Как осуществить просмотр каталогов и их содержимого?

Для просмотра каталогов и их содержимого используется команда ls.

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

Команда mkdir -p /path/to/directory создаст новый каталог, включая все недостающие родительские каталоги.

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

Для осуществления рекурсивного копирования всех файлов из одного каталога в другой используется команда cp -r.

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

Для рекурсивного удаления всех файлов и подкаталогов в определенном каталоге используется команда rm -r и указывается путь к каталогу.

14. Перечислите основные ключи команды ls с их назначением.

-l — вывод в длинном формате с подробной информацией.

-a — отображение всех файлов, включая скрытые.

-r — вывод в обратном порядке.

-t — сортировка по времени изменения.

-F — добавление символов, указывающих на тип файла.

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

Команда cat ("concatenate") используется для вывода содержимого файлов на экран, объединения нескольких файлов в один, а также для передачи данных из одного файла в другой.

Команда tee копирует входные данные как в файл, так и на стандартный вывод (экран).

more и less позволяют постранично просматривать содержимое файла, в отличие от cat, который выводит его сразу.

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?

SystemD — это система инициализации и управления службами в Linux. Она отвечает за запуск, остановку и управление службами, а также за отслеживание состояния системы.

18. Какие типы юнитов вы знаете?

В SystemD есть несколько типов юнитов:

Service (.service) — это юниты для управления службами (демонами). Они могут запускаться, останавливаться, перезапускаться и проверяться на предмет текущего состояния.

Socket (.socket) — юниты для управления сокетами, которые используются для активации служб по запросу. Служба запускается, когда поступает трафик на соответствующий сокет.

Target (.target) — юниты для группировки других юнитов.

Device (.device) — юниты для управления устройствами, обнаруживаемыми ядром системы. Они могут управлять доступом к устройствам или активировать службы при подключении определённых устройств.

Mount (.mount) — юниты, которые управляют точками монтирования файловых систем.

Automount (.automount) — юниты, которые управляют автоматическим монтированием файловых систем при обращении к ним. Эти юниты работают вместе с .mount.

Timer (.timer) — юниты для управления задачами, которые должны выполняться по расписанию.

Swap (.swap) — юниты для управления swap-разделами или swap-файлами. Они определяют, когда и как должен быть активирован swap.

Path (.path) — юниты для отслеживания изменений в файлах или директориях. Они могут запускать службы, если изменяется файл или каталог.

Slice (.slice) — юниты для управления группами процессов (cgroups). Они позволяют организовать процессы в иерархию для более точного управления ресурсами (CPU, память).

Scope (.scope) — юниты для управления процессами, которые создаются не с помощью systemd, но которым требуется мониторинг и контроль через systemd.

19. С помощью каких команд осуществляется управление демонами?

Для управления демонами используются команды systemctl, такие как:

systemctl start [имя\_службы] — запуск службы (демона).

systemctl stop [имя\_службы] — остановка службы.

systemctl restart [имя\_службы] — перезапуск службы.

systemctl reload [имя\_службы] — перезагрузка конфигурации службы без остановки.

systemctl status [имя\_службы] — проверка статуса службы.

systemctl enable [имя\_службы] — включение автозапуска службы при загрузке системы.

systemctl disable [имя\_службы] — отключение автозапуска службы.

systemctl is-active [имя\_службы] — проверка, запущена ли служба в данный момент.

systemctl is-enabled [имя\_службы] — проверка, включён ли автозапуск службы.

systemctl daemon-reload — перезагрузка конфигурации демонов systemd. Эта команда применяется после изменения конфигурационных файлов systemd.

20. Чем корутина отличается от потоков?

Корутины — это функции, которые могут быть приостановлены и возобновлены без создания нового потока. Они легче и эффективнее потоков, так как не требуют переключения контекста ядра.

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

Реализация корутин как встроенной возможности языка позволяет более эффективно управлять потоками выполнения и их синхронизацией, упрощает их использование и уменьшает накладные расходы.

22. Какая польза от корутины?

Корутины позволяют писать асинхронные программы, которые легче поддерживать, чем традиционные многопоточные программы. Они помогают избежать блокировок и эффективно использовать процессорное время.

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

ln — Создание жестких или символических ссылок.

cp —Копирование файлов.

mv —Перемещение или переименования файлов.

rm — Удаляет файлы.

mkdir — Создание новой директории.

echo — Вывод строки текста в терминал.

chmod — Изменение прав доступа к файлам и каталогам.

tar — Команда для работы с архивами (создание, распаковка и прочее).

ls — Вывод содержимого каталога и информации о файлах.

su — Смена пользователя.

cd — Изменение рабочего каталога.

sudo — Позволяет выполнить указанные команды с административными привилегиями.

adduser — Создание новой учетной записи.

cat — Читает данные из файла или стандартного ввода и выводит их на экран.

# **Вывод**

В ходе выполнения лабораторной работы был приобретён опыт работы с файлами и каталогами в ОС Linux, настроены права на доступ к файлам и каталогам.

Получены практические навыки по работе с подсистемой инициализации и управления службами.