

Министерство науки и высшего образования
Российской Федерации
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Кузбасский государственный технический университет
имени Т. Ф. Горбачева»

Институт профессионального образования
Кафедра информатики и информационных систем

Составитель К. А. Кулиничев

ОСНОВЫ СИСТЕМНОГО АДМИНИСТРИРОВАНИЯ

Методические материалы

Рекомендовано цикловой методической комиссией
Информационных систем и программирования
в качестве электронного издания
для использования в образовательном процессе

Кемерово 2026

Рецензенты:

К. В. Ерошевич, преподаватель кафедры информатики и информационных систем.

Кулиничев Константин Андреевич

Основы системного администрирования : методические материалы для обучающихся специальности СПО 09.02.11 Разработка и управление программным обеспечением / Кузбасский государственный технический университет имени Т. Ф. Горбачева, Кафедра информатики и информационных систем ; составитель К. А. Кулиничев. – Кемерово : КузГТУ, 2026. – 1 файл (1801 КБ). – Текст: электронный.

Методические материалы по дисциплине «Основы системного администрирования» содержат указания для выполнения лабораторных и самостоятельных работ, перечень вопросов на защиту выполненных работ.

© Кузбасский государственный
технический университет
имени Т. Ф. Горбачева, 2026
© Кулиничев К. А.,
составление, 2026

Раздел 1. ОСНОВЫ АДМИНИСТРИРОВАНИЯ

Лабораторная работа № 1. Установка виртуальной машины

Этапы настройки и установки операционной системы Debian

Создание виртуальной машины для операционной системы (ОС) происходит в VirtualBox (рисунок 1).

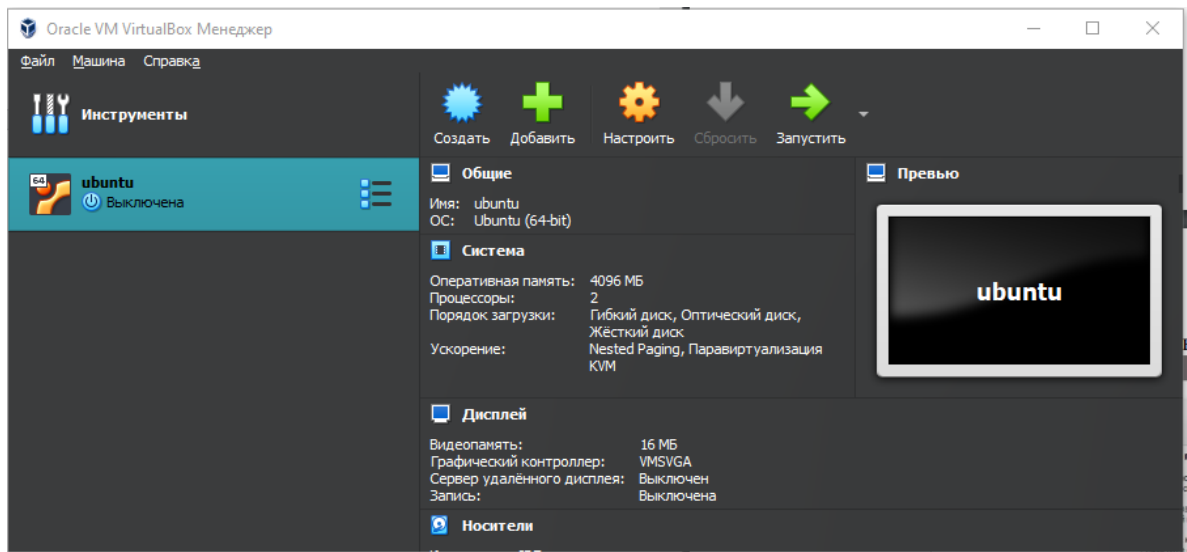


Рисунок 1 – Среда VirtualBox

Нажимаем кнопку **Создать**, выбираем тип ОС, версию, даем ей имя, а также выдаем некоторый объем оперативной памяти, исходя из всего имеющегося объема и нажимаем Создать (рисунок 2–3).

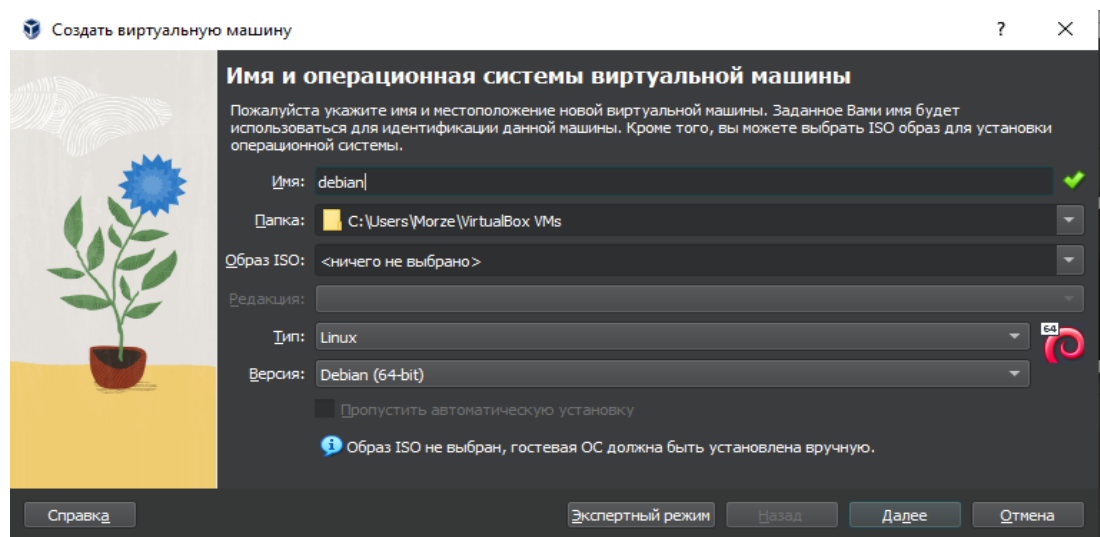


Рисунок 2 – Основные параметры

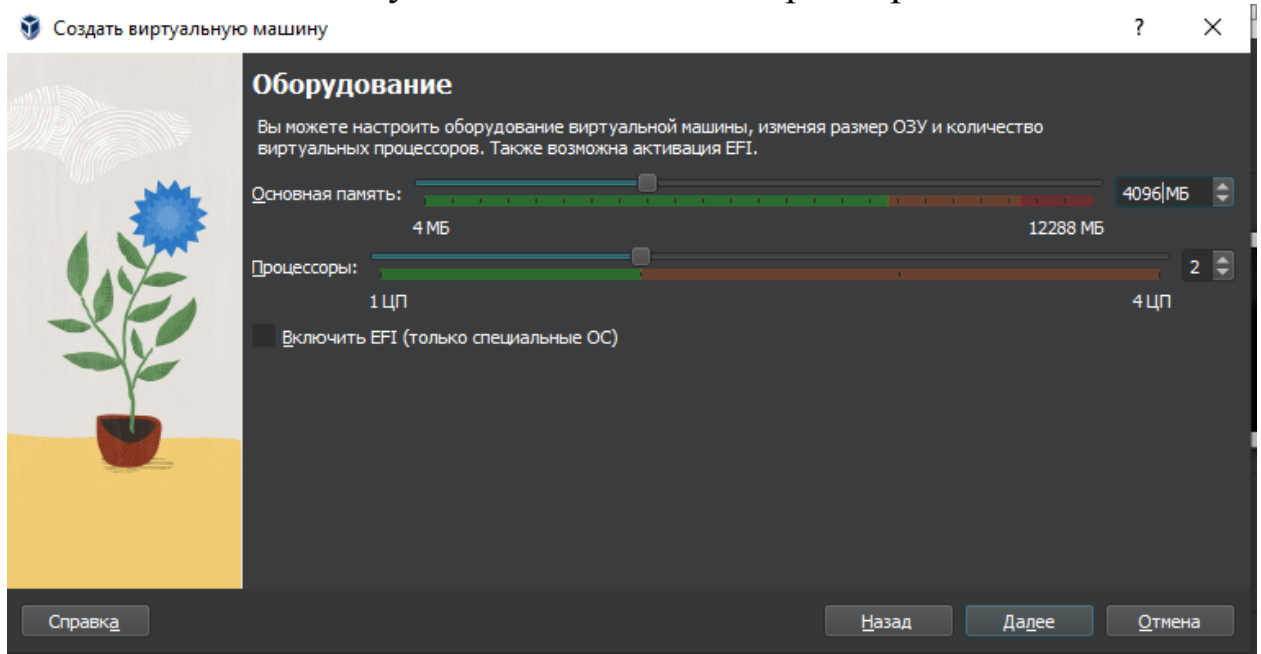


Рисунок 3 – Выбор оборудования

Выбираем место, где будет храниться наш виртуальный жесткий диск, а также его объем, нажимаем Создать (рисунок 4).

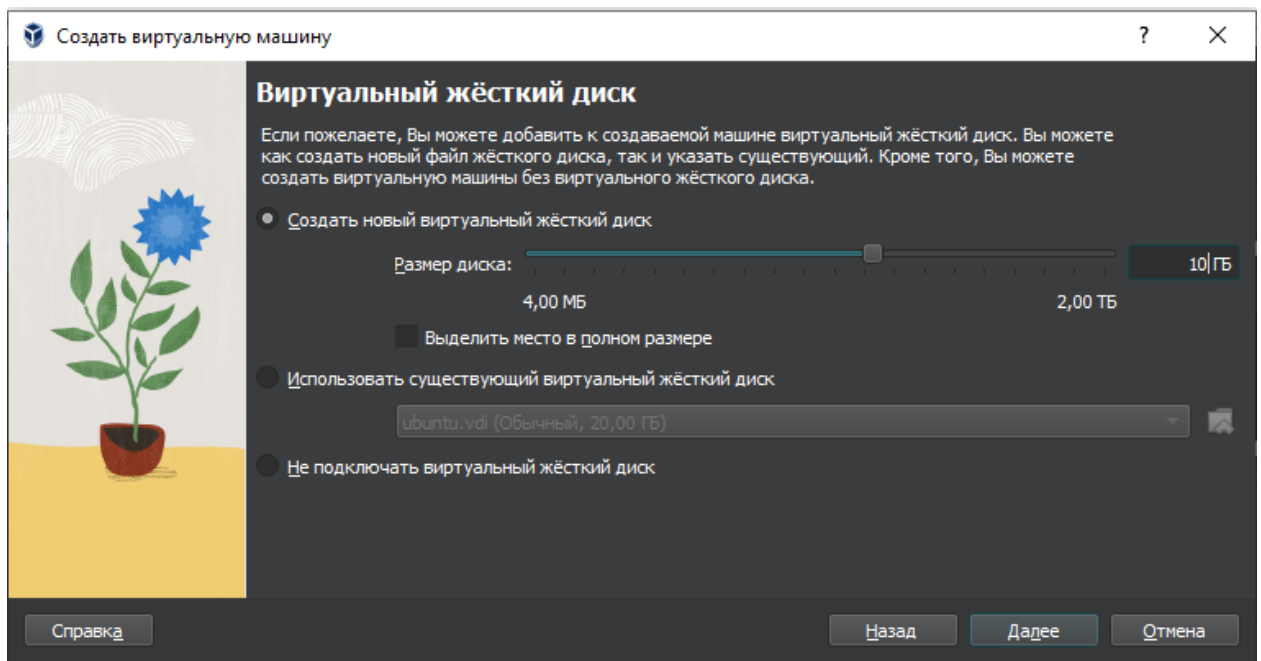


Рисунок 4 – Выбор жесткого диска

Итоговые параметры системы для создания виртуального диска представлены на рисунке 5.

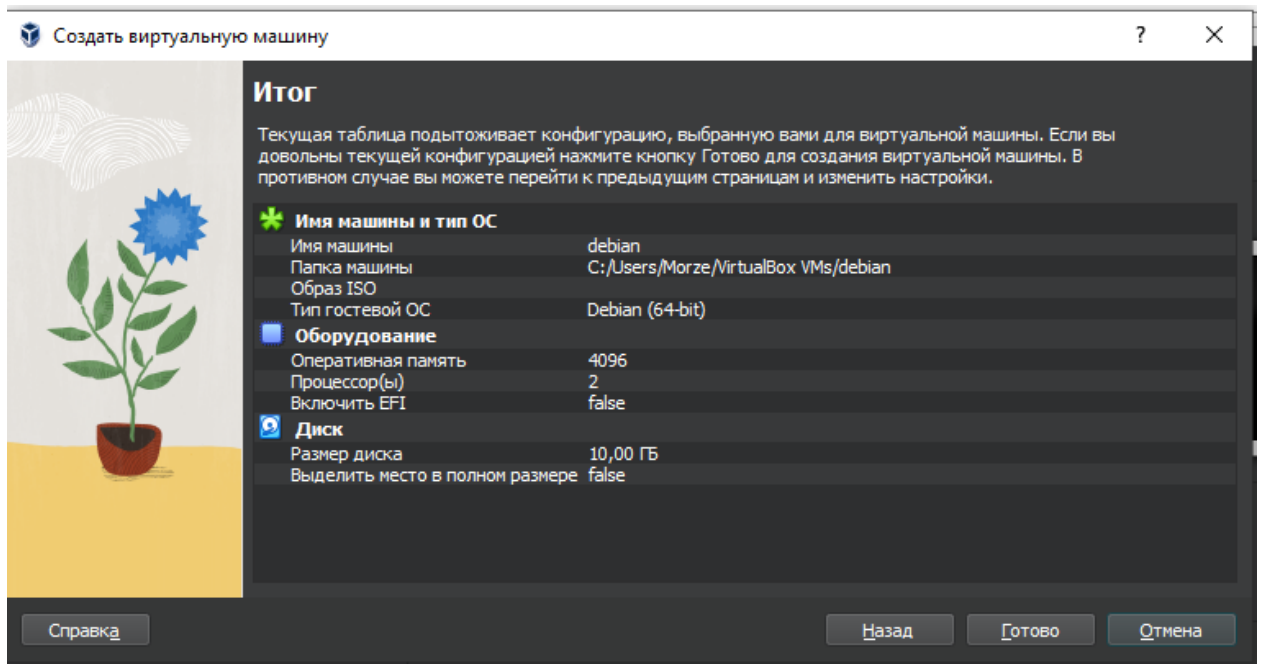


Рисунок 5 – Итоговые параметры для системы

Появляется созданный виртуальный диск, но с неустановленной на него ОС (рисунок 6).

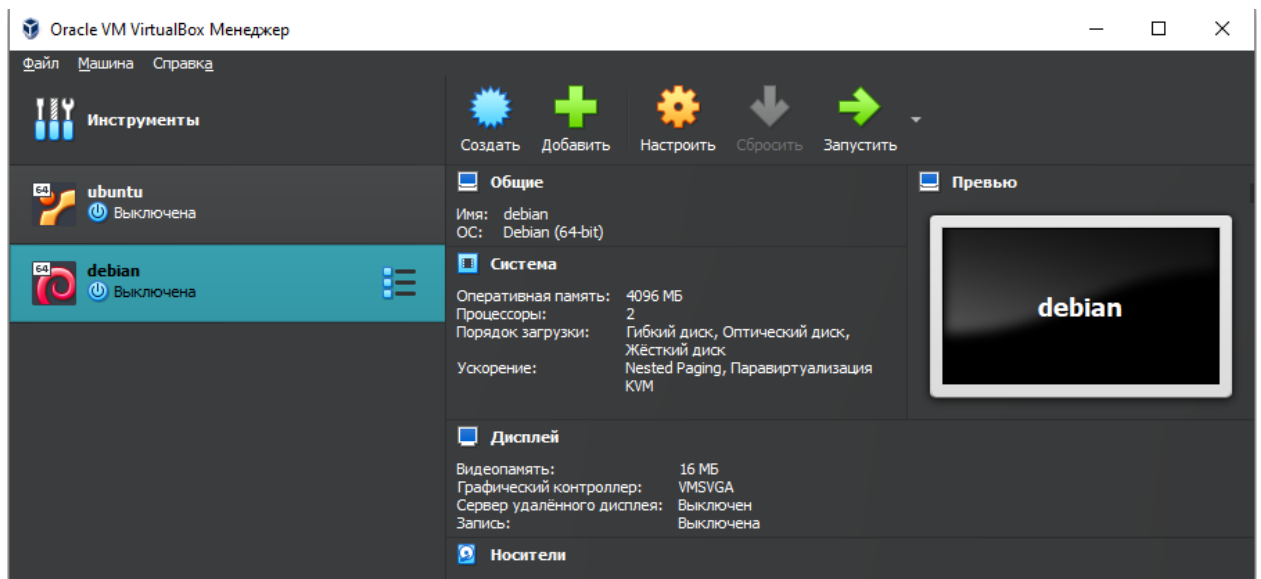


Рисунок 6 – Виртуальный диск

Нажимаем кнопку **Запустить** и указываем путь к образу **debian** и выбираем **Продолжить** (рисунок 7).

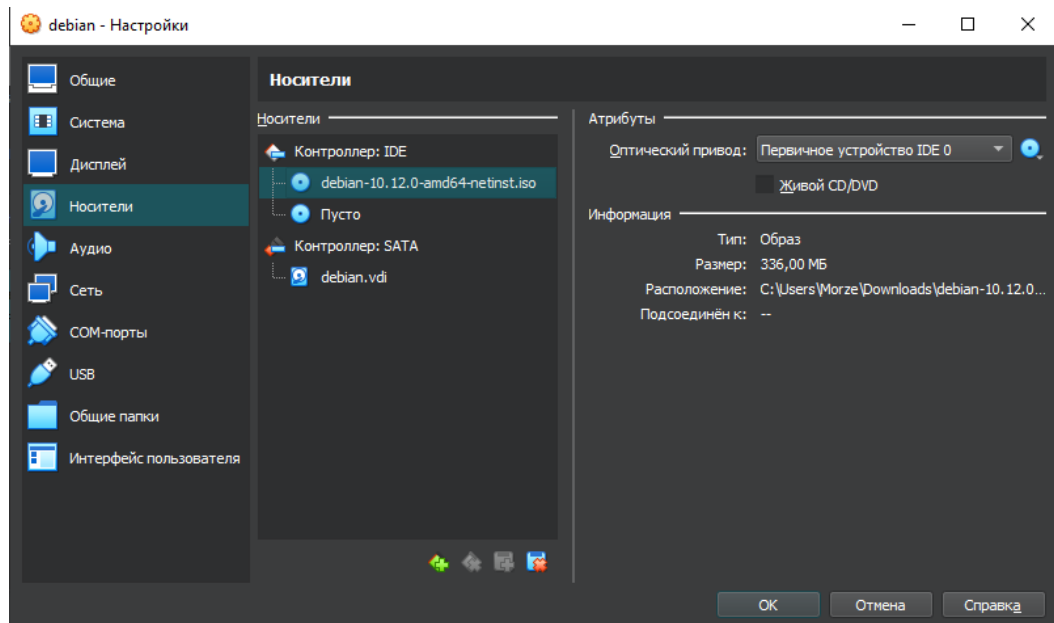


Рисунок 7 – Выбор образа диска

Загрузка и установка ОС

Появляется **окно приветствия установки debian** (рисунок 8).

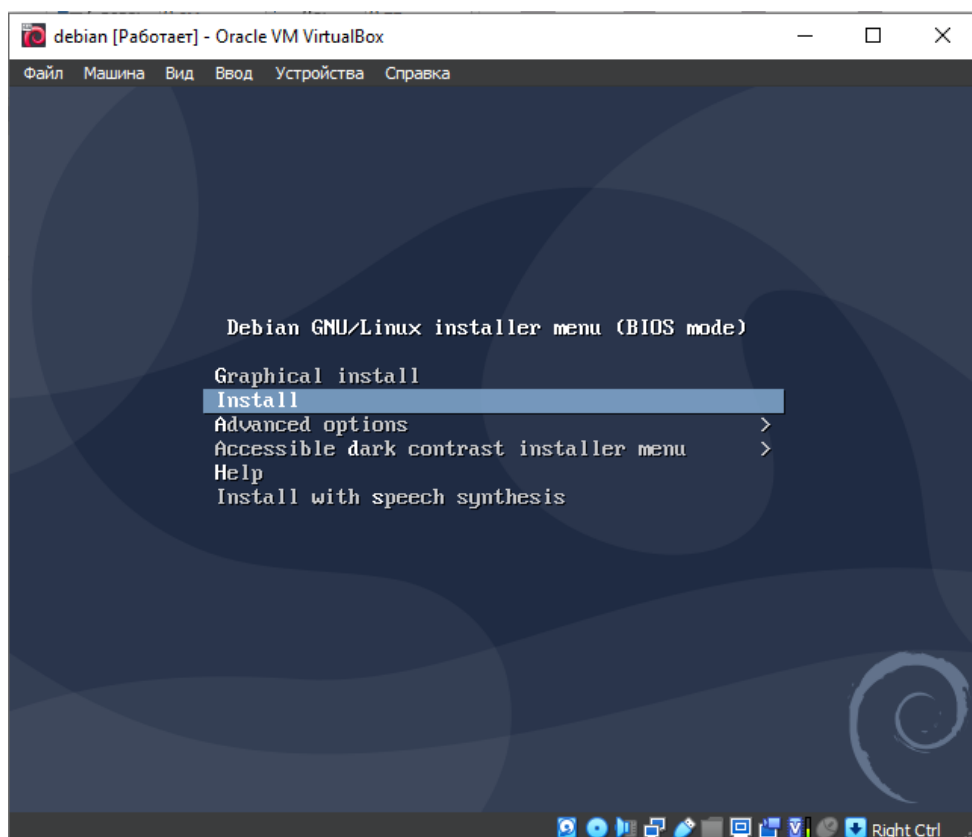


Рисунок 8 – Окно начала установки системы

Управление установкой производится с помощью клавиш *Enter*, *пробел* и *стрелок*. Выбираем **Install**, выбираем *язык установки* (рисунок 9).

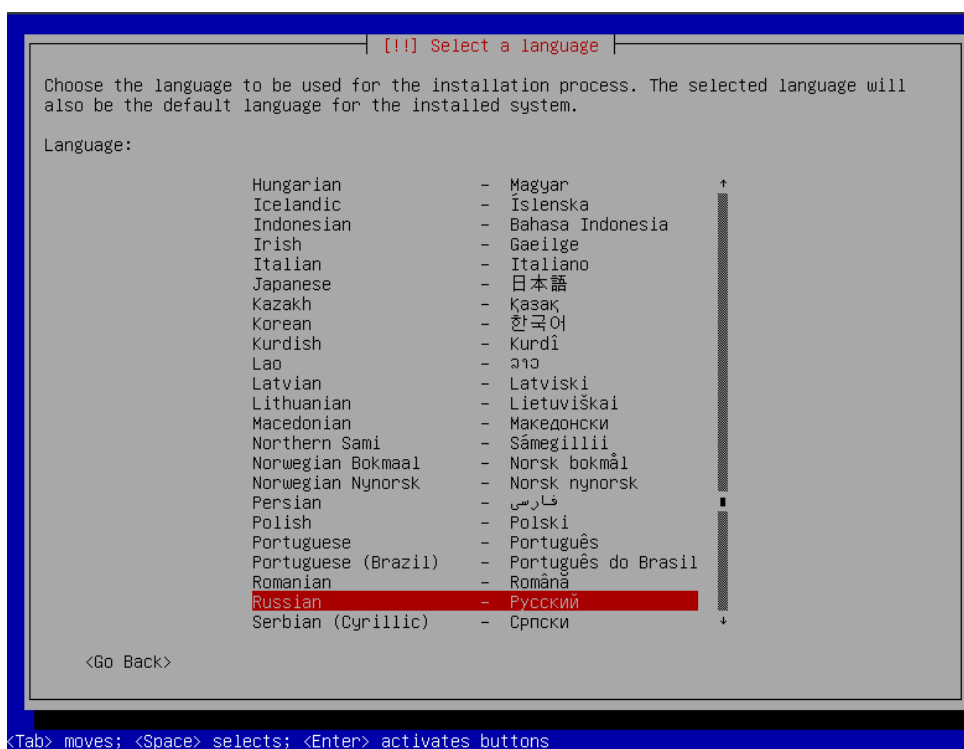


Рисунок 9 – Выбор языка

Необходимо выбрать *страну* (рисунок 10).

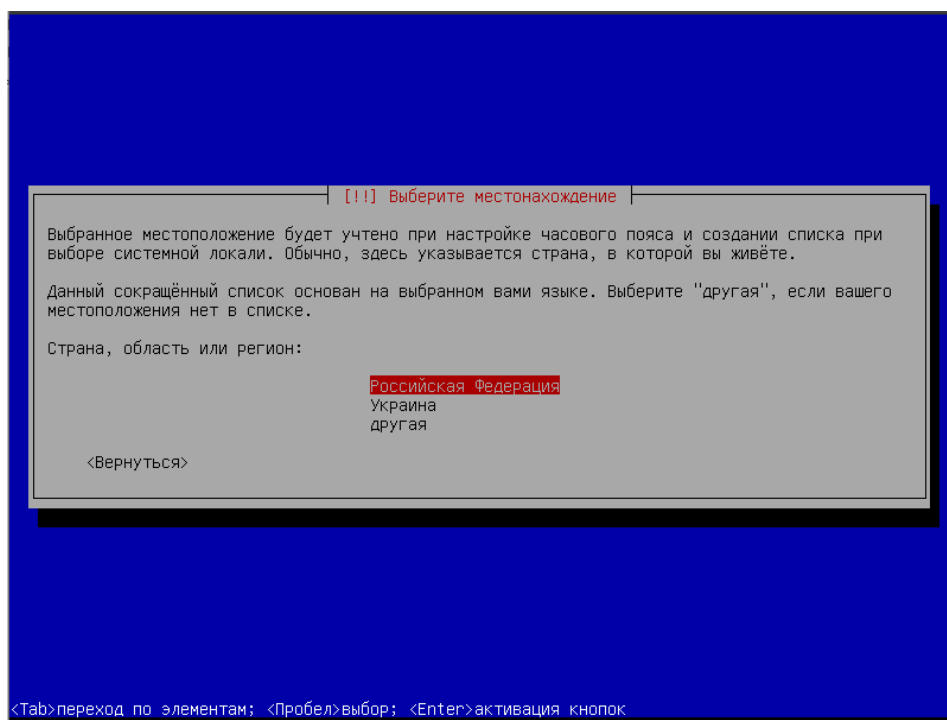


Рисунок 10 – Выбор страны

Раскладку выбираем *английскую* (рисунок 11).

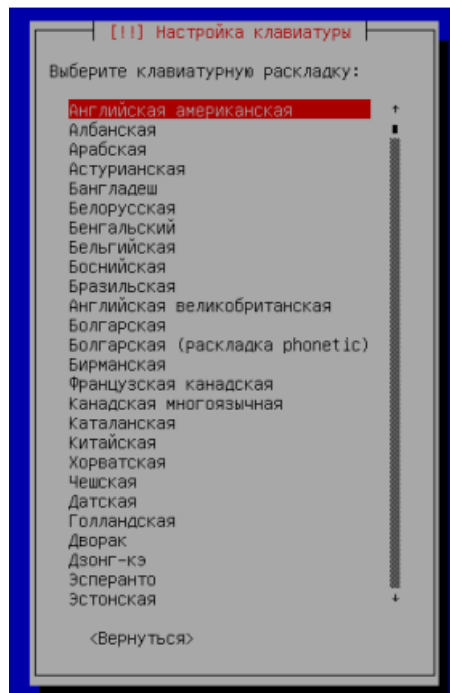


Рисунок 11 – Выбор раскладки клавиатуры

Под именем компьютера пишем свою фамилию (рисунок 12).

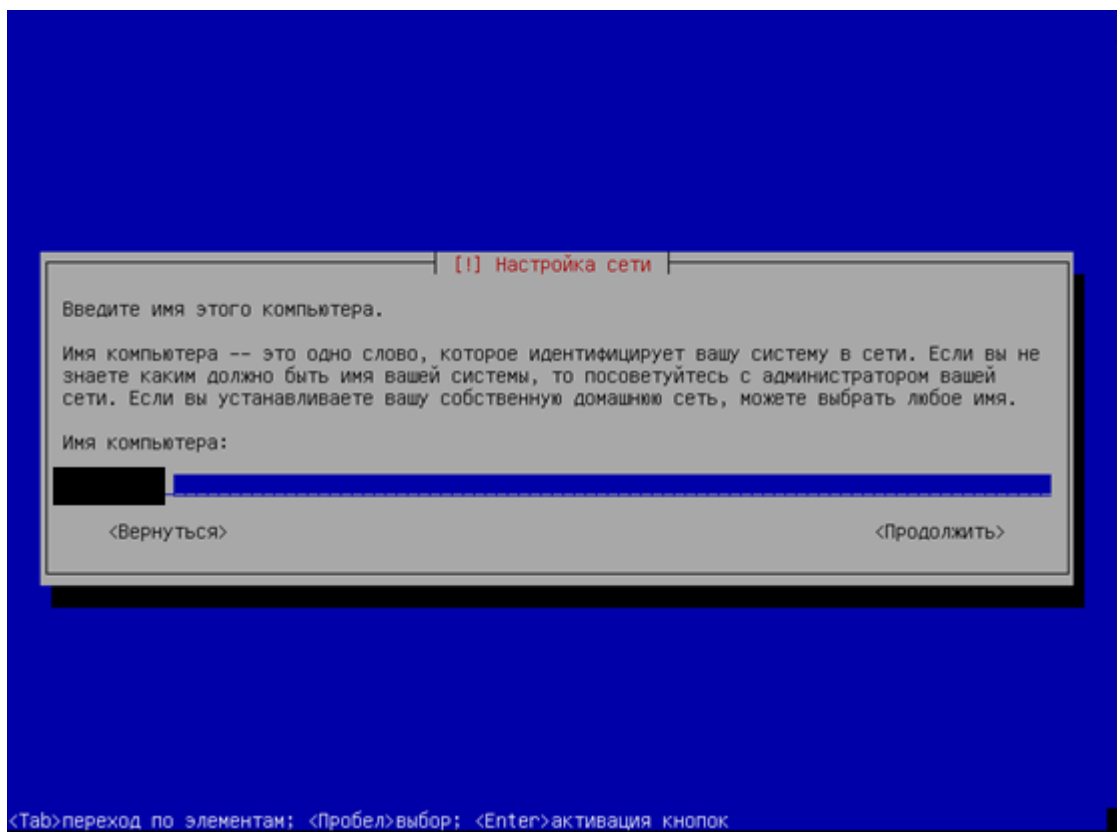


Рисунок 12 – Имя компьютера

Имя домена пропускаем. Придумываем пароль суперпользователя и подтверждаем его (рисунок 13).

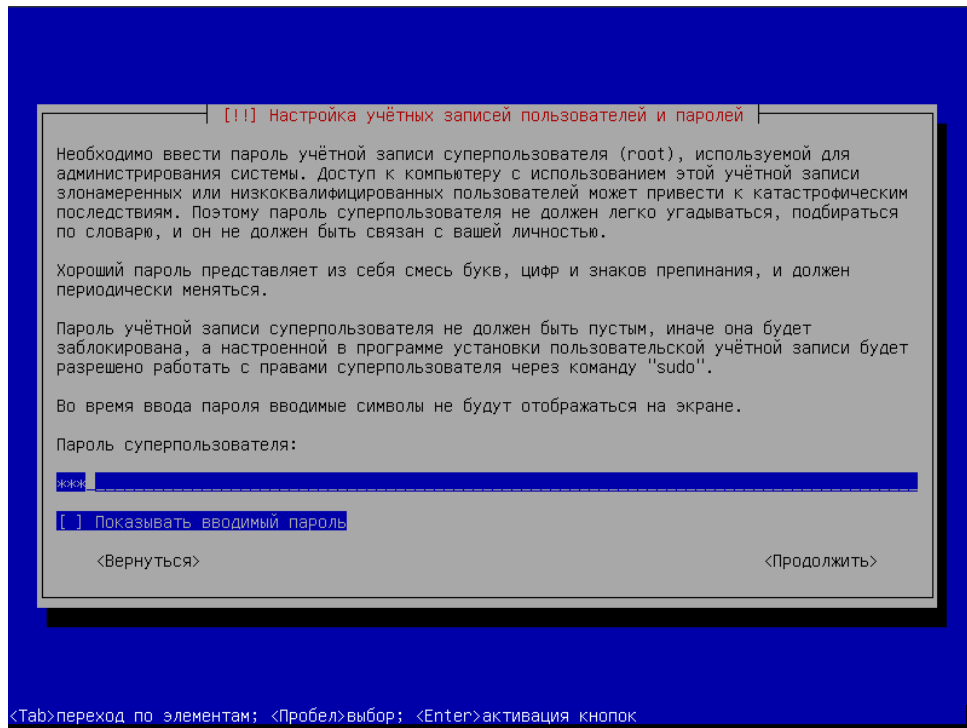


Рисунок 13 – Пароль суперпользователя

В имени пользователя и имени учётной записи вписываем свое имя (рисунок 14–15).

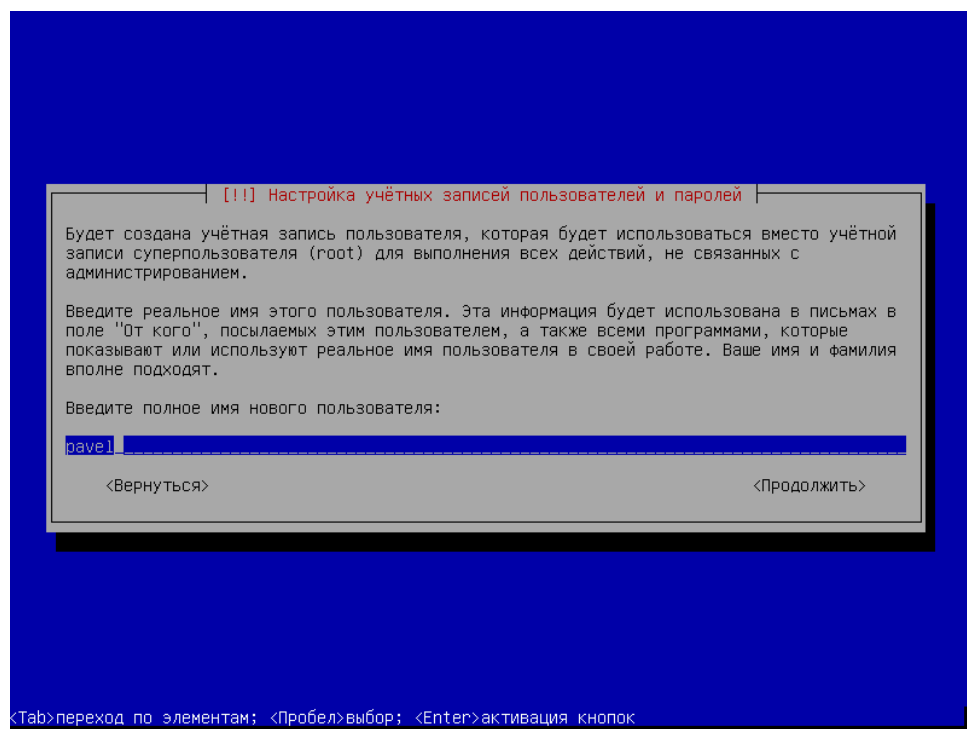


Рисунок 14 – Имя пользователя

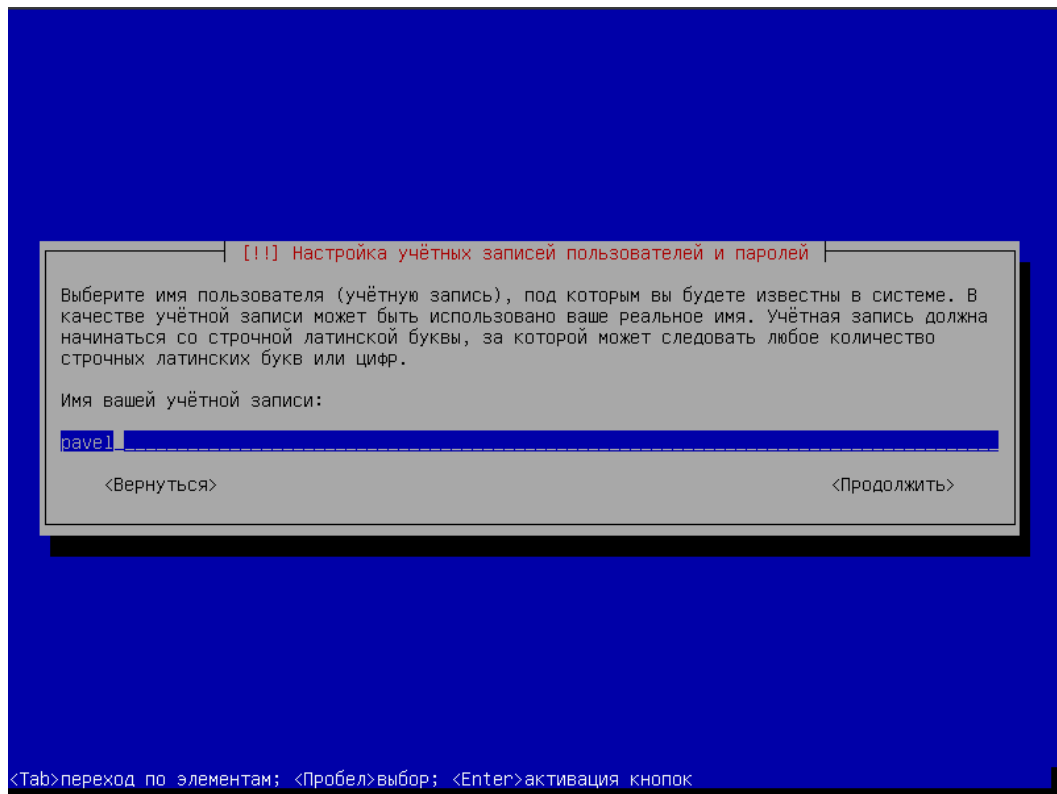


Рисунок 15 – Имя учетной записи

Придумываем и подтверждаем пароль пользователя (рисунок 16).

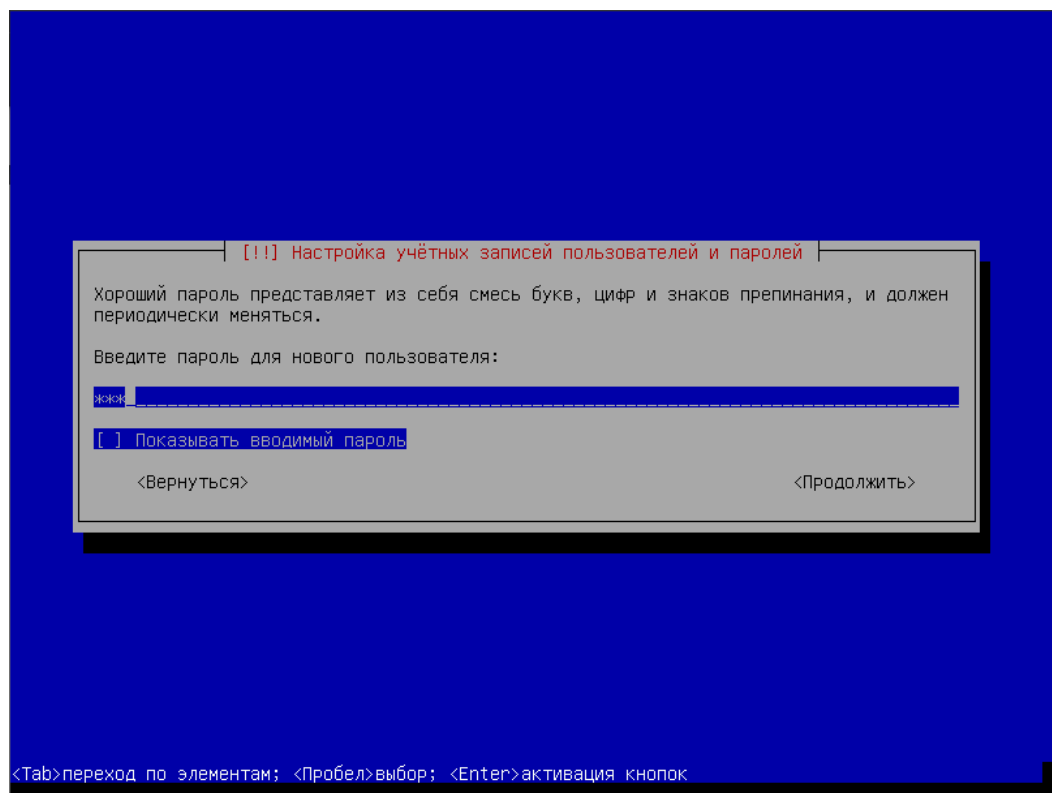


Рисунок 16 – Пароль

Выбираем *часовой пояс* (рисунок 17).

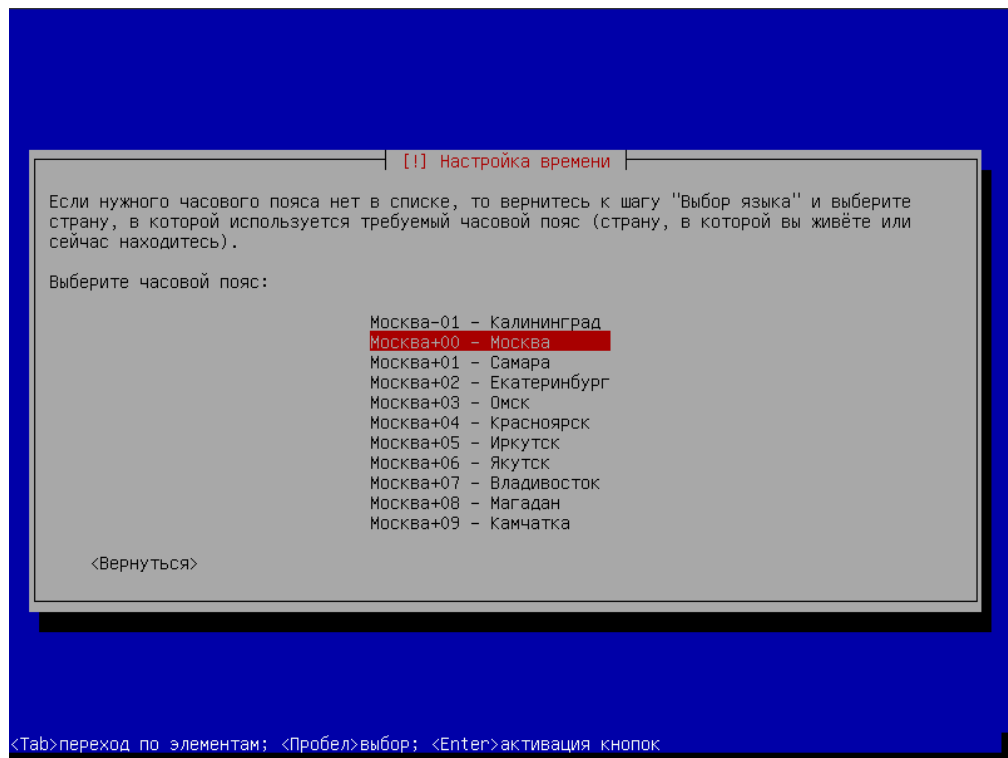


Рисунок 17 – Выбор часового пояса

Разметку диска выбираем самую простую, *автоматическую* (рисунок 18).

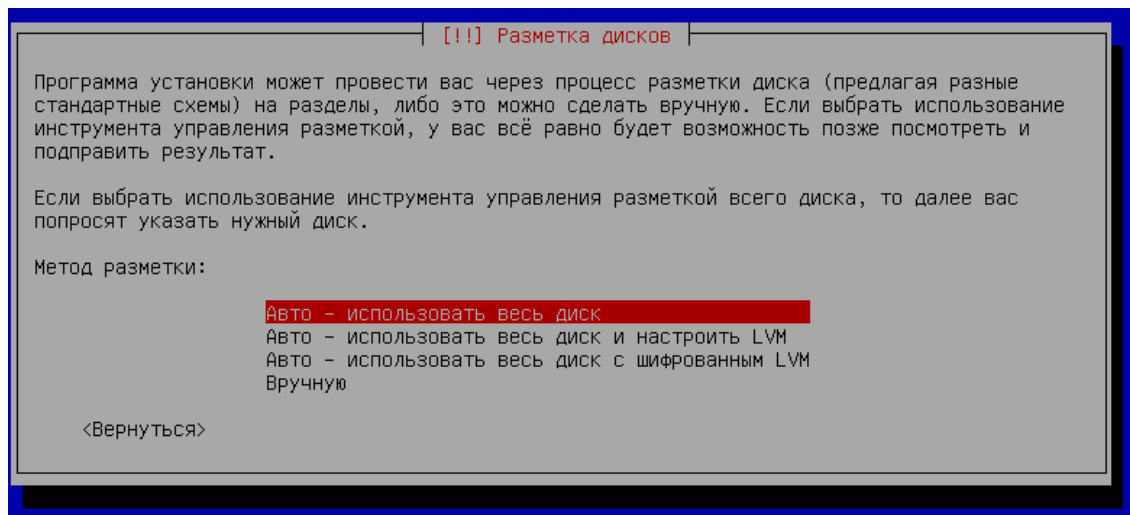


Рисунок 18 – Выбор разметки диска

Далее система спросит у нас имя жесткого диска, на который будет производиться установка, он у нас один, выбираем его (рисунок 19).

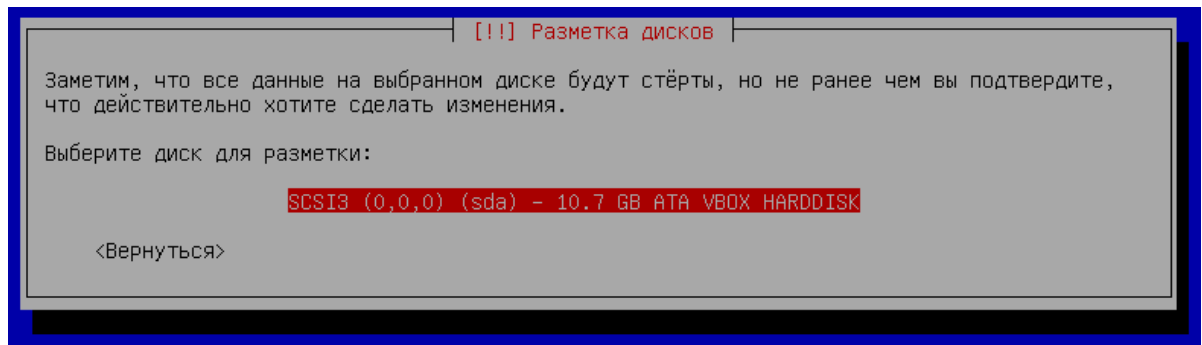


Рисунок 19 – Выбор диска для установки

Далее система предлагает разбить дисковое пространство на разделы, но делать этого не будем (рисунок 20).

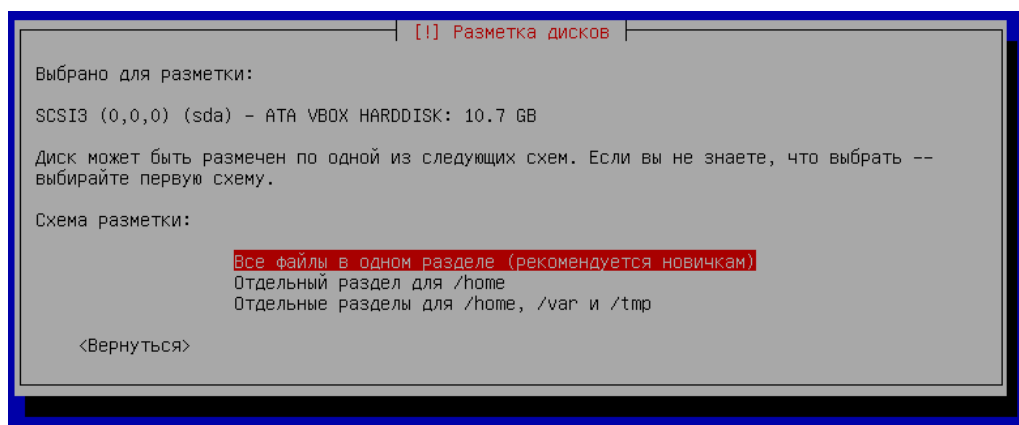


Рисунок 20 – Разметка диска

Здесь видим (рисунок 21), что установщик автоматически разбил диск на пространство под систему и файл подкачки.

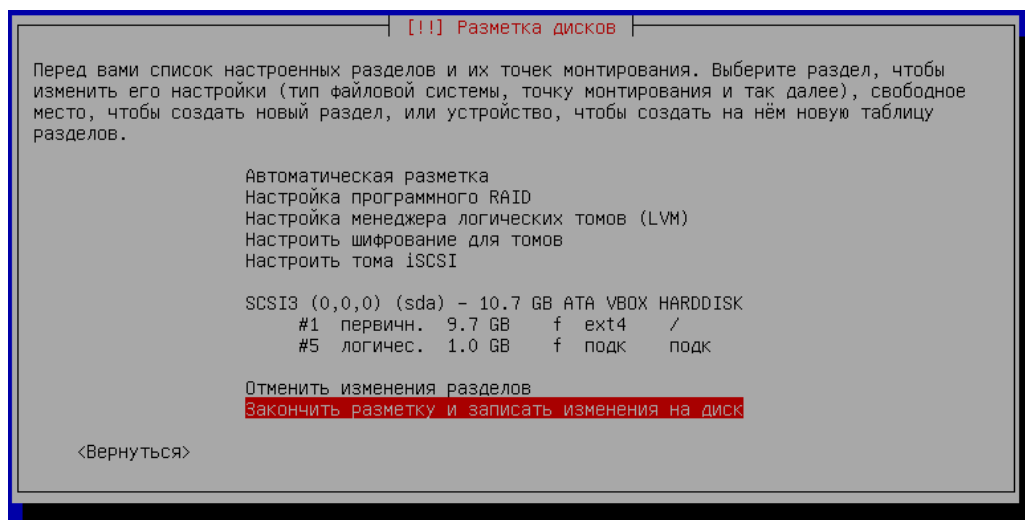


Рисунок 21 – Автоматическая разметка диска

Подтверждаем форматирование разделов (рисунок 22).

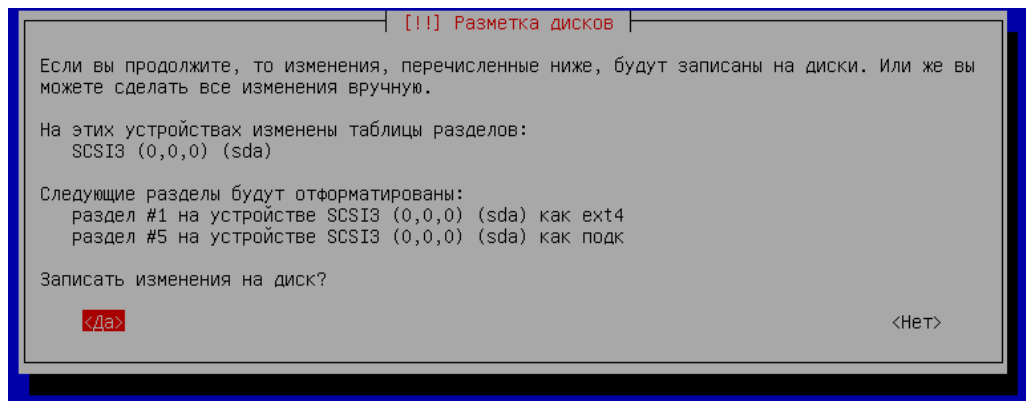


Рисунок 22 – Форматирование разделов

Далее система спросит, нужно ли сканировать добавочные диски с дополнительными пакетами, но так как у нас их нет – отказываемся (рисунок 23).

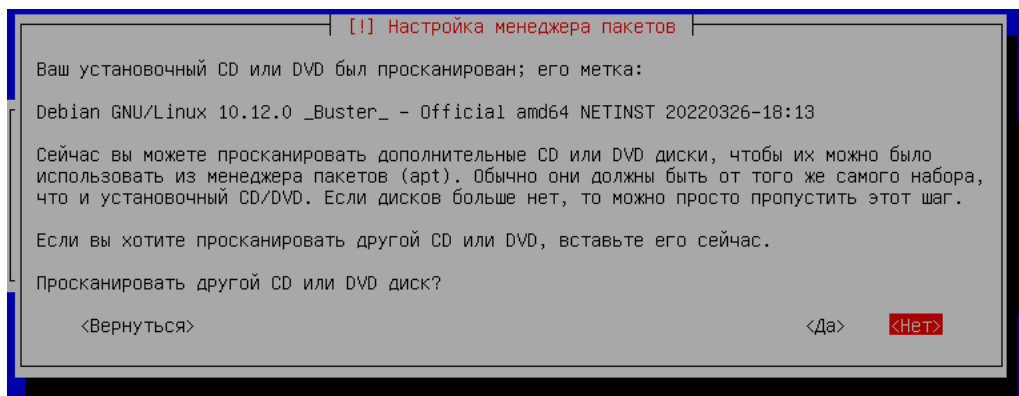


Рисунок 23 – Дополнительное сканирование (менеджер пакетов)

Отвечаем в зависимости от наличия у вас подключения к сети (рисунок 24).

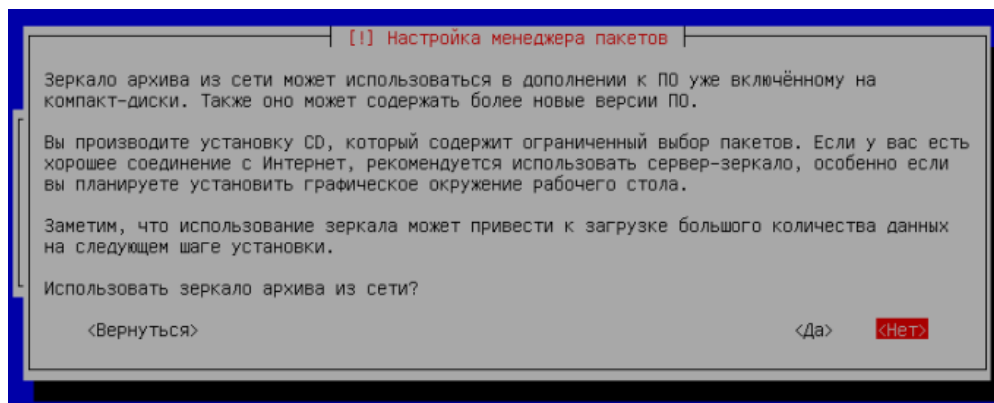


Рисунок 24 – Зеркало архива из сети

Ответ по желанию (рисунок 25).

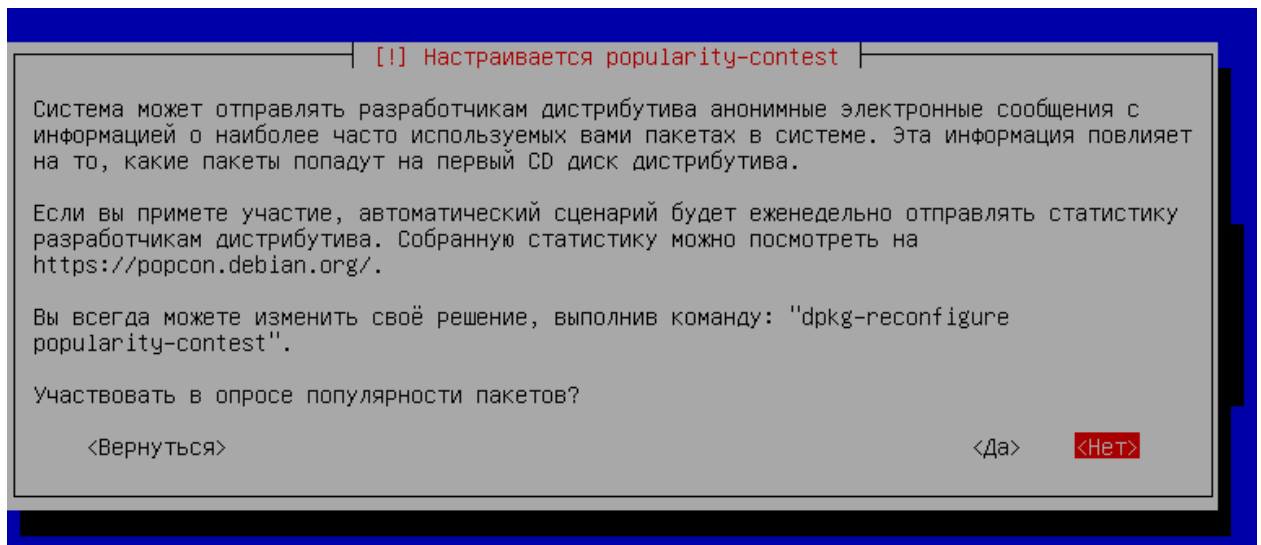


Рисунок 25 – Опрос популярности пакетов

ПРОБЕЛОМ убираем звездочку с первой строки, если вы производите установку на виртуальную машину, если же параллельно с уже привычной вами ОС, то графику можно оставить (рисунок 26).

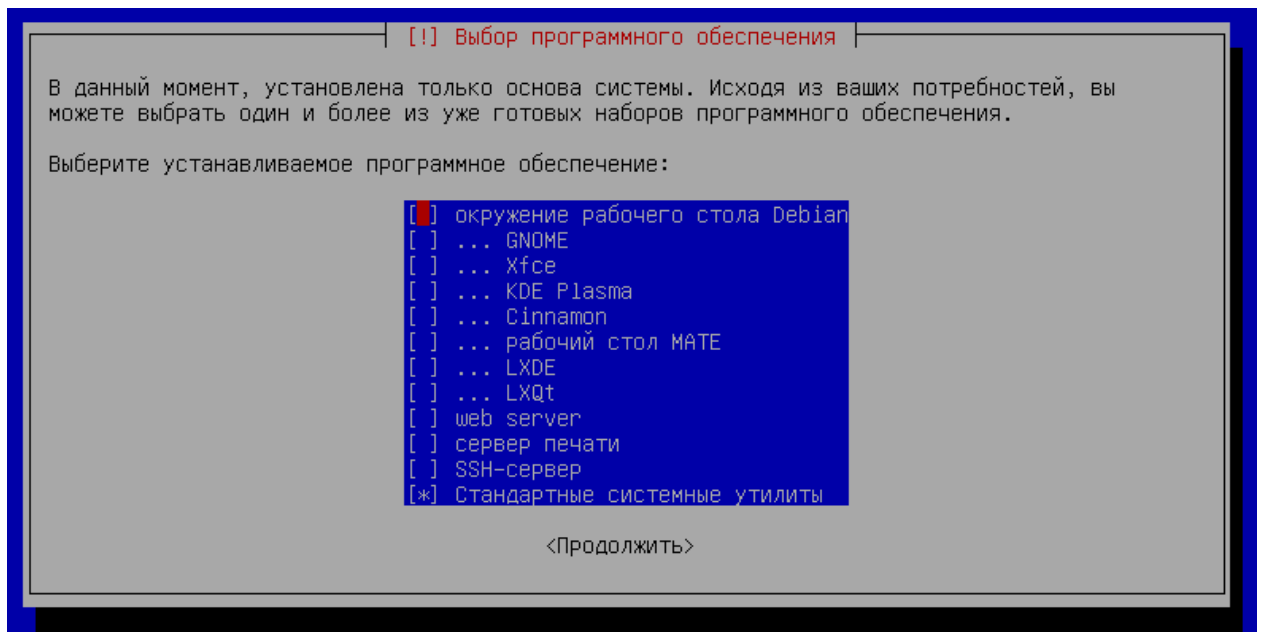


Рисунок 26 – Выбор программного обеспечения

Подтверждаем установку загрузчика GRUB (рисунок 27).

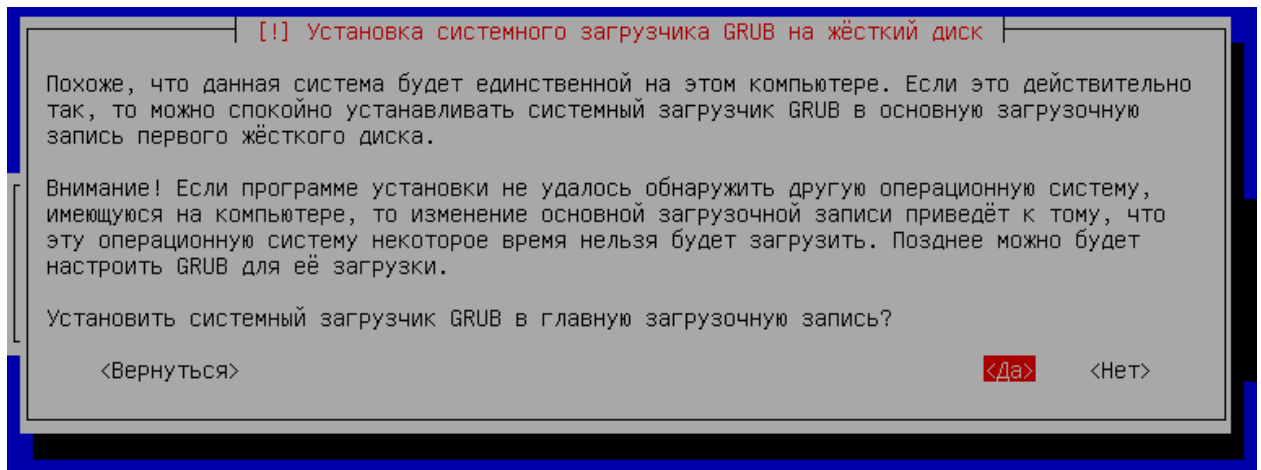


Рисунок 27 – Установка загрузчика GRUB

Указываем устройство – наш жесткий диск (рисунок 28).

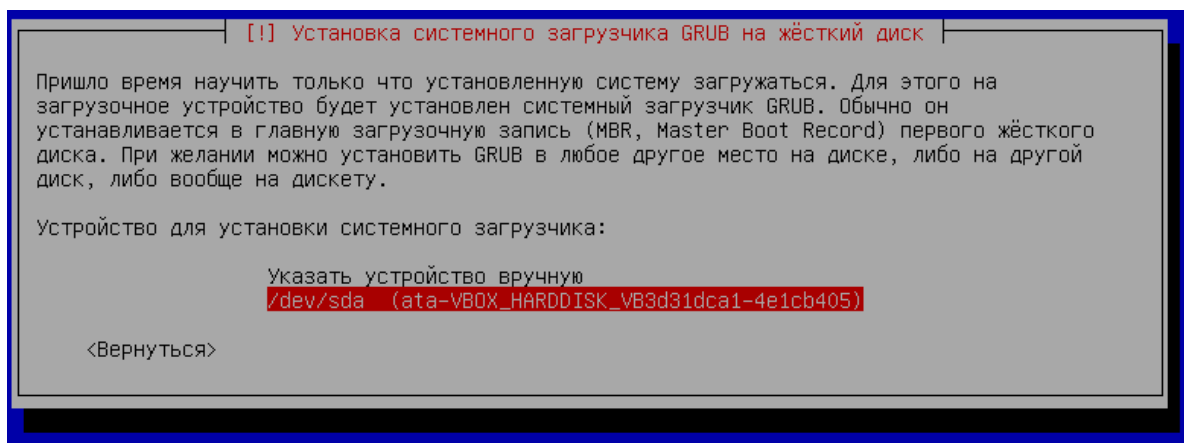


Рисунок 28 – Устройство для установки

Завершение установки ОС

Завершение установки операционной системы (рисунок 29).

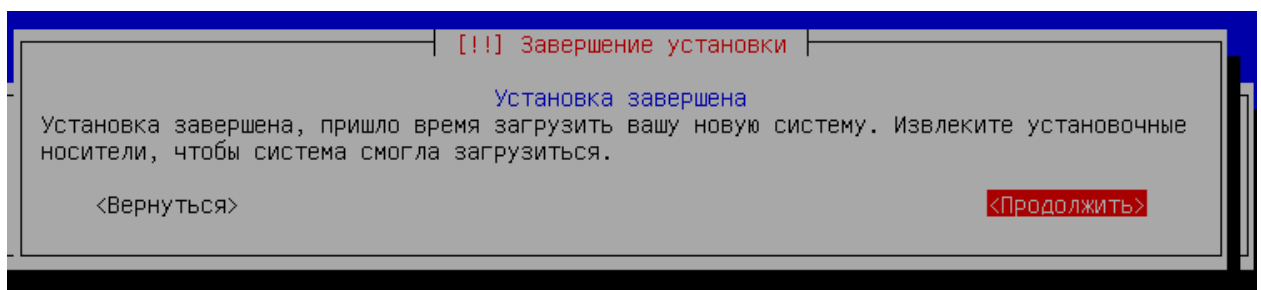


Рисунок 29 – Завершение установки

Производим перезагрузку и можем войти в нашу новую ОС (рисунок 30).

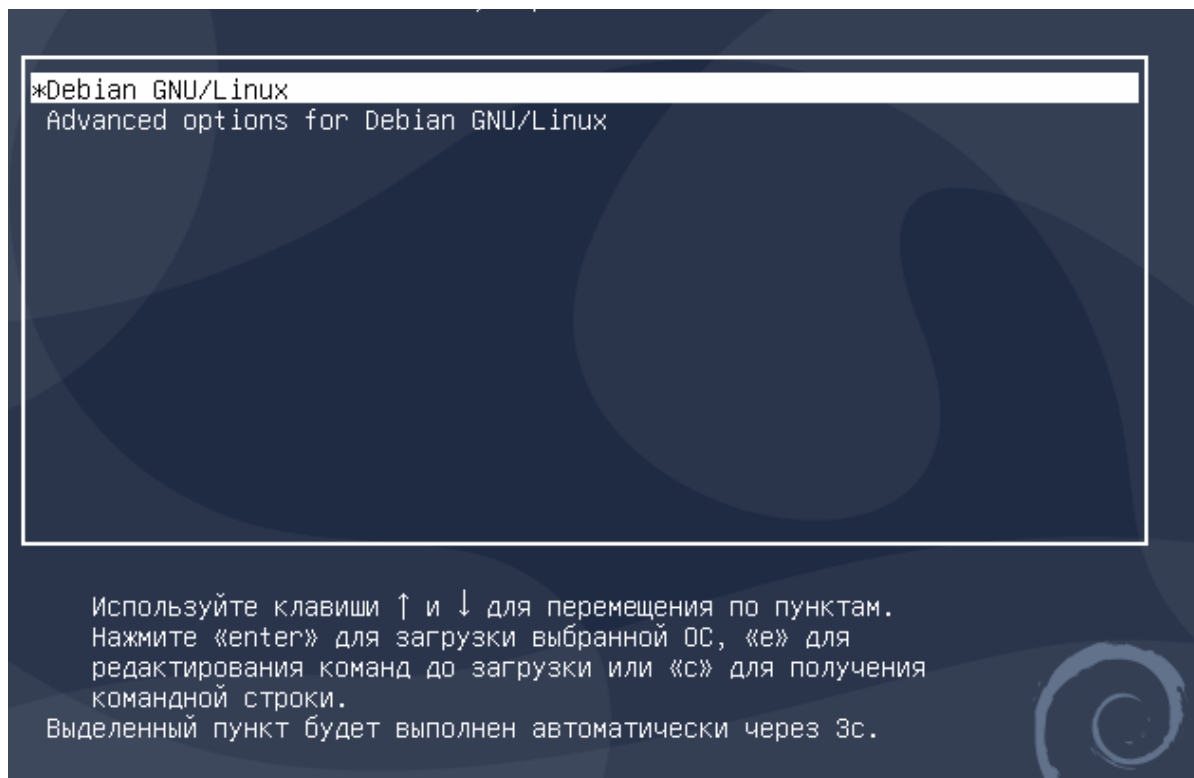


Рисунок 30 – Окно запуска системы

Задание

Установить операционную систему Linux Debian:

1. Запустите виртуальную машину и выполните установку системы в соответствии с инструкцией в картинках.
2. Имя пользователя задайте своей **фамилией** в латинской раскладке, пароль – на свое усмотрение.
3. Прикрепите скриншот, демонстрирующий установленную систему, в которую выполнен вход под пользователем с Вашей фамилией.
4. Для того, чтобы "освободить" курсор мыши, который захватывает виртуальная машина используйте кнопку, название которой отображается в правом нижнем углу окна виртуальной машины (по умолчанию в системе Windows – *правый Ctrl*).

Контрольные вопросы

1. Что такое виртуальная машина?
2. Как создать виртуальную машину?

3. Какие операционные системы можно устанавливать на виртуальную машину?
4. Какие ресурсы использует виртуальная машина?
5. Сколько операционных систем можно запускать одновременно?

Лабораторная работа № 2. Установка дополнений

Установка, обновление и полное удаление утилит и пакетов

В ОС Windows есть понятие установщика программы, как правило, это файл с названием "setup.exe", который выполняет установку программы. В ОС *Linux* все программы распространяются в пакетах – в файлах с расширением ".rpm" (в дистрибутивах RedHat, Fedora, CentOS) или ".deb" (в дистрибутивах Debian, Ubuntu). Пакет содержит следующую информацию:

- программу – один или несколько исполнимых файлов;
- файлы, необходимые для работы программы (данные, конфигурации);
- документацию;
- инструкции, которые нужно выполнить при установке и удалении программы;
- информацию о зависимостях программы (некоторые программы требуют для своей установки наличия других программ);
- информацию о конфликтах программы (некоторые программы не могут работать одновременно с другими);
- информацию о разработчике программы.

Менеджер пакетов позволяет разрешать зависимости и конфликты при установке программ. Пакеты хранятся в репозиториях (электронное хранилище данных), у каждого дистрибутива есть собственный набор репозиториев.

Существует два основных менеджера пакетов: apt (в дистрибутивах Debian, Ubuntu) и dnf (в дистрибутивах Fedora, CentOS).

В Linux Ubuntu формат вызова менеджера пакетов имеет следующий вид:

1. **sudo apt [опции] команды [пакет]**
2. **sudo apt-get [опции] команды [пакет]**

Первая команда используется в более новых дистрибутивах, а вторая – как в новых, так и в старых.

Для установки программы или утилиты необходимо ввести команду (для завершения команд нажмите клавишу **Tab**): **sudo apt-get install <название_программы>**.

Для обновления программы или утилиты необходимо ввести команду: **sudo apt-get upgrade <название_программы>**.

Для полного удаления программы или утилиты необходимо ввести команду (перед "auto-remove" два символа "-"): **sudo apt-get purge -- auto-remove <название_программы>**.

Важно отметить, что при обновлении системы и установке утилит требуется **использовать права пользователя "root"**. Для этого необходимо перед командой установки или обновления ввести команду **sudo**. При этом будет запрошен пароль, который был введен при установке ОС Linux. Вводимые символы пароля не отображаются. Также стоит отметить, что во время установки или обновления может потребоваться подтверждение действий, например, на вопрос "Хотите продолжить?" следует ввести ответ "y".

Задание

Установить дополнения по инструкции:

1. Установите пакеты: linux-headers-amd64 gcc make perl любым способом (synaptic, apt install, apt-get install).
2. Подключите диск дополнений: меню VirtualBox, устройства, вставить диск дополнений.
3. На рабочем столе откройте содержимое диска по появившемуся значку.
4. Приложения – Эмулятор терминала.
5. **sudo sh /media/cdrom/VBoxLinuxAdditions.run**.
6. Введите пароль для администрирования.
7. После окончания процесса перезагрузите систему.

Записать видео из виртуальной машины "Вид" – "Запись" после установки дополнений, демонстрирующее вход в систему и автоматическую подгонку экрана при изменении размеров окна виртуальной машины.

Контрольные вопросы

1. С помощью каких команд можно установить дополнения?
2. Какие дополнения можно устанавливать в систему?
3. Откуда можно устанавливать дополнения?
4. Под каким пользователем необходимо устанавливать дополнения?

Лабораторная работа № 3. Изменение общесистемных настроек

Основные команды для администрирования ОС Linux

Для работы с ОС Linux используется командная строка терминала. Ниже представлены основные команды, используемые для администрирования.

Важно: Linux является чувствительным к регистру. Команды `ls-a` и `ls-A`, а также файлы `text_file.txt` и `Text file.txt` не обозначают один и тот же объект.

Команды для управления файлами и каталогами:

- **ls** – утилита для просмотра содержимого каталогов. По умолчанию показывает текущий каталог. Если в параметрах указать путь, то она перечислит содержимое конечного каталога. Полезные опции `-l` (List) и `-a` (All). Первая форматирует вывод в виде списка с более подробной информацией, а вторая включает показ скрытых файлов.
- **cat** – печатает содержимое файла, переданного в параметре, в стандартный вывод. Если передать несколько файлов, команда склеит их. Также можно перенаправить вывод в еще один файл с помощью символа `>`. Если нужно вывести только определенное количество строк, используйте опцию `-n` (Number).
- **tac** – в отличие от `cat` печатает файл в обратном виде.
- **cd** – позволяет перейти из текущего каталога в указанный пользователем. Если запустить без параметров – возвращает в домашний каталог. Вызов с двумя точками возвращает на уровень вверх относительно текущего каталога. Вызов с тире (`cd -`) возвращает к предыдущему каталогу.
- **touch** - команда изменяет время модификации файла, а если файл не существует, создает его.
- **pwd** – печатает на экран текущий каталог, в котором находится пользователь. Это может быть полезно, если командная строка Linux не выводит такую информацию. Эта команда будет востребована в Bash программировании, где для получения ссылки на каталог выполняется скрипт.

- **mkdir** – создание новых каталогов. Опция -p (Parents), позволяет создать всю структуру подкаталогов одной командой, даже если они еще не существуют.
- **file** – показывает тип файла. В Linux файлы не всегда должны иметь расширения, чтобы с ними работать.
- **cp** – копирование файлов и каталогов.
- **mv** – перемещение или переименование файлов и каталогов. *Важно отметить*, что в Linux это одна и та же операция. Переименование – это перемещение файла в ту же директорию с другим именем.
- **chmod** – изменяет права доступа к файлу. Это чтение, запись и выполнение. Каждый пользователь может изменять права для своих файлов.
- **chown** – изменяет владельца файла. Только суперпользователь может изменять владельцев.
- **find** – поиск в файловой системе, файлах и папках.
- **locate** – в отличие от find, команда locate ведет поиск в базе данных updatedb для шаблонов имен файлов.
- **du** – показывает размер файла или каталога. Самые полезные опций:
 - h (Human), которая преобразует размеры файлов в легко читаемый формат,
 - s (Summarize), которая выводит минимум данных, и
 - d (Depth), устанавливающая глубину рекурсии по каталогам.
- **df** – анализатор дискового пространства. По умолчанию вывод достаточно подробный: перечислены все файловые системы, их размер, количество использованного и свободного пространства. Для удобства есть опция -h, делающая размеры легко читаемыми.
- **free** – вывод информация об использовании оперативной памяти. Для удобства есть опция -h, делающая размеры легко читаемыми.
- **mount / umount** - это команды консоли Linux для подключения и отключения файловых систем Linux. Только у суперпользователя есть права для этого.

Команды консоли для работы с текстом:

- **more / less** – это две команды терминала для просмотра длинных текстов, которые не помещаются на одном экране. Если ваш терминал не поддерживает прокрутки, вы можете сделать это с помощью команды **less**, которая поддерживает больше опций.
- **head / tail** – команда **head** выводит несколько первых строк из файла, а **tail** выдает несколько последних строк. По умолчанию каждая утилита выводит десять строк, но это можно изменить с помощью опции **-n**. Еще один полезный параметр **-f**, это сокращение от **follow** (следовать). Утилита постоянно выводит изменения в файле на экран. Например, если нужно следить за лог-файлом, вместо того, чтобы постоянно открывать и закрывать его, используйте команду **tail -nf**.
- **grep** – команда **grep** ищет текст по шаблону. По умолчанию она принимает стандартный ввод, но возможно искать в файлах. Шаблон может быть строкой или регулярным выражением. Утилита может вывести как совпадающие, так и не совпадающие строки и их контекст.
- **sort** – сортировка строк текста по различным критериям. Наиболее полезные опции: **-n** (Numeric), по числовому значению, и **-r** (Reverse), которая переворачивает вывод.
- **wc** – утилита командной строки Linux для подсчета количества слов, строк, байт и символов.
- **diff** – показывает различия между двумя файлами в построчном сравнении. Причем выводятся только строки, в которых обнаружены отличия. Измененные строки отмечаются символом "с", удаленные – "d", а новые – "a".

Команды для управления процессами:

- **ps / pgrep** – один из способов получить идентификатор процесса, это утилита **ps**, которая печатает информацию о запущенных процессах. По умолчанию вывод очень длинный, поэтому следует использовать опцию **-e**, чтобы увидеть информацию об определенном процессе. Выводится снимок состояния на момент вызова, и информация не будет обновляться. Команда **ps** с ключом **aux** выводит полную

информацию о процессах. `pgrep` работает следующим образом: задается имя процесса, а утилита показывает его идентификатор.

- **top / htop** – обе команды отображают процессы и могут быть использованы как консольные системные мониторы. Возможно не только просматривать, но и контролировать процессы через интерактивный интерфейс `htop`.
- **time** – время выполнения процесса. Это секундомер для выполнения программы. Не сообщает текущее время.

Команды окружения пользователя

- **su / sudo** – это два способа выполнить одну и ту же задачу: запустить программу от имени "root". В зависимости от дистрибутива используется `su` или `sudo`. Разница в том, что `su` переключает на пользователя "root", а `sudo` только выполняет команду от его имени. Поэтому использование `sudo` будет наиболее безопасным вариантом работы.
- **date** – выводит дату и время в стандартный вывод. Его можно форматировать в зависимости от потребностей: вывести год, месяц, день, установить 12- или 24-часовой формат, получить наносекунды или номер недели. Например, `date +"%j %V"`, выведет день в году и номер недели в формате ISO.
- **alias** – команда создает синонимы для других команд Linux. Возможно делать новые команды или группы команд, а также переименовывать существующие.
- **uname** – выводит основную информацию о системе. Если задать параметр `-a` (All), то можно получить информацию о ядре, имени хоста и узнать архитектуру процессора.
- **uptime** – сообщает время работы системы.
- **sleep** – для выключения компьютера через определенный промежуток времени или использования в качестве импровизированной тревоги.
- **poweroff** или **shutdown -h now** – завершить работу системы прямо сейчас и выключить питание.
- **reboot** **shutdown -r now** – перезагрузить систему прямо сейчас.
- **users** – информация о том, кто вошел в систему.

- **w** – выводит информацию о том, откуда вошел пользователь (т. е. его IP-адрес), как именно, когда, а также информацию об использовании процессора.
- **who** – выводит только имя пользователя, название консоли, время и дату входа и IP-адрес.

Команды для управления пользователями:

- **useradd / userdel / usermod** – эти команды консоли Linux позволяют добавлять, удалять и изменять учетные записи пользователей.
- **passwd** – эта команда позволяет изменить пароль учетной записи пользователя. Суперпользователь, может сбросить пароли всех пользователей, даже несмотря на то, что не может их увидеть.

Команды для просмотра документации:

- **man / whatis** – команда **man** открывает руководство по определенной команде. Для всех основных команд Linux есть man-страницы. **whatis** показывает, какие разделы руководств есть для данной команды.
- **Whereis** – показывает полный путь к исполняемому файлу программы. Также может показать путь к исходникам, если они есть в системе.

Команды для управления сетью:

- **ip** – в пакете **net-tools** содержится множество утилит: **ipconfig**, **netstat** и прочие устаревшие, например, **iproute2**. Все это заменяет одна утилита – **ip**.
- **ping** – быстро проверяет доступность узла, подключение к маршрутизатору или к Интернету и дает представление о качестве связи.
- **nethogs** – позволяет узнать, сколько трафика использует какая-либо программа в Linux или какая программа потребляет всю скорость Интернет соединения. Для того чтобы задать сетевой интерфейс используется опция **-i**.
- **traceroute** – это усовершенствованная версия **ping**. Позволяет увидеть не только полный маршрут сетевых пакетов, но и

доступность узла, а также время доставки этих пакетов на каждый из узлов.

- **route** – вывод локальной таблицы маршрутизации (`route -n`).
- **ifconfig** – вывод информации о сетевом адаптере.
- **tracpath** – аналог `tracert`.

Задание

Произвести администрирование ОС Linux:

1. Настройка дополнительного имени компьютера.
 - 1.1. Изучение конфигурирования начнем с простейшего по структуре файла **/etc/hosts** (в Windows есть точно такой же файл, только расположенный в другом каталоге, т. к. сетевой стек Windows 2000 был основан на сетевом стеке другой Unix-подобной системы – BSD), который отвечает за перевод символьных имен компьютеров в IP-адреса. Откройте эмулятор терминала, переключитесь на пользователя `root` введя команду **su -**, откройте на редактирование файл **/etc/hosts** командой **nano /etc/hosts**. Файл имеет простейшую структуру в несколько колонок, колонки отделяются друг от друга любым пробельным символом, обычно символом табуляции. В каждой строке в первой колонке записывается IP-адрес, во всех остальных – соответствующие ему имена. Символ **#** традиционно обозначает начало комментария – все, начиная с него самого, и до конца строки полностью игнорируется. Порядок записей не важен, система сама отсортирует их в нужном порядке.
 - 1.2. Внесите дополнительную запись в файл со следующими параметрами: `127.0.0.1 фамилия.имя`.
 - 1.3. Сохраните файл, затем проверьте, что система обновила настройки, выполнив команду **ping фамилия.имя**. Прекратить команду можно сочетанием клавиш **Ctrl+C**.
 - 1.4. Сделайте скриншот с успешным выполнением команды `ping`.
2. Настройка экрана входа в систему.
 - 2.1. Сделайте скриншот окна входа в систему.
 - 2.2. Сначала изучите содержимое каталога **/usr/share/backgrounds**, найдите там файлы картинок, запомните полный путь к ним.

- 2.3. За вход в систему отвечает отдельная программа – дисплейный менеджер. При использовании xfce дисплейным менеджером является lightdm. Его настройки хранятся в отдельном каталоге **/etc/lightdm**, настройки внешнего вида находятся в файле **lightdm-gtk-greeter.conf**.
- 2.4. Откройте файл **lightdm-gtk-greeter.conf** с помощью текстового редактора, почти все его содержимое будет закомментировано, найдите незакомментированную строку с описанием секции **[greeter]**. В данной секции есть параметр **background**, установите его равным полному пути к файлу картинки из **/usr/share/backgrounds**, не забудьте убрать символ комментария.
- 2.5. Выйдите из системы, чтобы увидеть экран входа, убедитесь, что он изменился.
- 2.6. Сделайте скриншот окна входа в систему с новым фоном.
3. Установка **apache2**.
 - 3.1. С помощью **synaptic** установите пакет **apache2** со всеми зависимостями.
4. Настройки **apache2**.
 - 4.1. Apache в Debian имеет достаточно сложную структуру настроек, которая призвана облегчить работу администратора (напоминает езду на велосипеде – чтобы научиться надо приложить некоторые усилия, зато потом передвигаться можно гораздо быстрее). В целом для настройки Apache достаточно одного единственного файла **apache2.conf** (или **httpd.conf**), но в Debian редактировать его не рекомендуется, кроме исключительных случаев. Настройки делаются с помощью всех остальных файлов и подкаталогов.
 - 4.2. Найдите в **/etc/apache2** подкаталоги **sites-available** и **sites-enabled**, работают они следующим образом: все возможные конфигурации всех сайтов складываются в каталог **sites-available**, что позволяет сохранять их даже когда они не запущены в работу. Для того, чтобы нужный сайт заработал – нужно создать в каталоге **sites-enabled** символическую ссылку на файл в каталоге **sites-available**. Таким образом, для того, чтобы включать/отключать сайты достаточно создавать/удалять только ссылки, а не переписывать конфигурационные файлы (которые могут быть очень объемными) целиком.

- 4.3. В каталоге `sites-available` находится файл **000-default.conf**, сделайте его копию в том же каталоге, задайте имя файла: `фамилия.имя.conf`.
 - 4.4. Отредактируйте полученный файл, изменив параметр **ServerName** на **фамилия.имя** и параметр **DocumentRoot** на другую папку, например, `/var/www/html1`.
 - 4.5. Сделайте скриншот файла настроек.
 - 4.6. Создайте папку, указанную в предыдущем пункте, создайте в ней файл **index.html** по образцу минимальной структуры HTML-файла, таким образом, чтобы данный файл отображал Ваши полные ФИО и группу.
 - 4.7. В каталоге `/etc/apache2/sites-enabled` создайте символическую ссылку на файл `/etc/apache2/sites-available/фамилия.имя.conf` аналогично имеющейся там символической ссылке на файл `000-default.conf`.
 - 4.8. Сделайте скриншот содержимого каталога `sites-enabled`.
 - 4.9. Перезапустите `apache` командой **`service apache2 restart`**, чтобы настройки были прочитаны и применены, и проверьте, что все работает командой **`service apache2 status`**.
 5. Проверка работы `apache`.
 - 5.1. Откройте web-браузер и перейдите в нем по адресу `http://фамилия.имя`.
 - 5.2. Сделайте скриншот браузера со страницей, отображающей Ваши данные на сайте с Вашими фамилией и именем.
- Все сделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Команды для управления файлами и каталогами?
2. Команды консоли для работы с текстом?
3. Команды для управления процессами?
4. Команды окружения пользователя?
5. Команды для управления пользователями?
6. Команды для просмотра документации?
7. Команды для управления сетью?

Лабораторная работа № 4. Управление пользователями и правами

Пользователи и группы

Linux является многопользовательской системой. Несколько пользователей могут работать с ней одновременно. Один пользователь может быть зарегистрирован в системе локально, другой – по сети, например через SSH.

Среди учетных записей нужно выделить пользователя "root" (для использования этой учетной записи необходимо ввести команду `sudo su` в Ubuntu, а для установки программного обеспечения достаточно только применить права "root" командой `sudo`). Он обладает максимальными правами в системе, поэтому при работе следует соблюдать осторожность. При работе с использованием обычных пользовательских учетных записей (без прав "root") максимальный вред может быть нанесен только файлам самих пользователей. Системные файлы останутся неизменными, т. к. система не позволит с ними ничего сделать. Ввод команды от имени "root" может привести к нерабочему состоянию всей системы.

Для более эффективного управления пользователями объединяют в группы.

По умолчанию один пользователь не имеет права доступа к домашнему каталогу другого пользователя, т.к. они находятся в разных группах. При объединении их в одну группу пользователи получают эти права.

В ОС Linux существует три типа пользователей:

- **root** (от англ. root – корень) – суперпользователь, аккаунт в UNIX-подобных системах, владелец которого имеет право на выполнение всех операций без исключения. Присутствует в системе по умолчанию.
- **Системные пользователи** – системные процессы, у которых есть учетные записи для управления привилегиями и правами доступа к файлам и каталогам. Создаются системой автоматически.
- **Обычные пользователи** – учетные записи пользователей, допущенных к управлению системой. Создаются системным администратором.

Каждый пользователь помимо имени имеет числовой идентификатор пользователя **UID** (User IDentificator):

- Пользователь root имеет идентификатор 0.
- Системные пользователи имеют идентификаторы от 1 до 100.
- Обычные пользователи имеют UID от 100.

Пользователи могут объединяться в группы. Каждый пользователь обязательно входит в ту или иную группу. Группы имеют числовой идентификатор группы **GID** (Group IDentificator).

Информация пользователей

В системе присутствует следующая информация о каждом пользователе:

- Имя пользователя (**user name**) – в рамках системы имя должно быть уникальным. В именах должны использоваться только английские буквы, числа и символы и . (точка).
- Идентификационный номер пользователя (**UID**) – является уникальным идентификатором пользователя в системе. Система отслеживает пользователей по UID, а не по именам.
- Идентификационный номер группы (**GID**) – обозначает группу, к которой относится пользователь. Каждый пользователь может принадлежать к одной или нескольким группам. Принадлежность пользователя к группе устанавливает системный администратор, чтобы иметь возможность ограничивать доступ пользователей к тем или иным ресурсам системы.
- Пароль (**password**) – пароль пользователя в зашифрованном виде.
- Полное имя (**full name**) – помимо системного имени может присутствовать полное имя пользователя, например фамилия и имя.
- Домашний каталог (**home directory**) – каталог, в который попадает пользователь после входа в систему. Подобный каталог имеется у каждого пользователя, все пользовательские каталоги хранятся в директории /home.
- Начальная оболочка (**login shell**) – командная оболочка, которая будет запускаться при входе в систему. Например, /bin/bash.

Вся информация о пользователях хранится в следующих файлах:

- **passwd** (etc/passwd) – содержит информацию о пользователях, имеет следующий формат записи: "user_name:password:UID:GID:full_name:home_directory:login_shell".
- Элементы записи должны разделяться символом – ":" (двоеточие) и записываются без пробелов. Если пароль хранится в зашифрованном виде в файле /etc/shadow, то вместо пароля указывается – "x".
- **group** (etc/group) – информация о группах, формат – "group_name:password:GID:user1,user2,user3". Элементы записи должны разделяться символом – ":" (двоеточие) записываются без пробелов. Имена пользователей записываются через запятую.
- У файлов /etc/passwd и /etc/group всегда определенные права доступа: чтение и запись для root, для остальных только чтение.
- **shadow** (etc/shadow) – в этом файле хранятся так называемые "теньевые пароли", информация о паролях пользователей в зашифрованном виде. Сделано это для безопасности, так как файл /etc/passwd может читаться кем угодно, а файл /etc/shadow может прочитать только root.
- **gshadow** (etc/gshadow) – то же самое что и **shadow**, только для паролей групп.
- Помимо основных, в системе присутствуют дополнительные файлы. **useradd** (etc/default/useradd) – файл, задающий свойства "по умолчанию" для всех добавляемых пользователей. Можно просмотреть командой – useradd -D.
- **login.defs** (/etc/login.defs) – содержит настройки для создания новых пользователей.
- **/etc/skel** – каталог с дефолтными файлами, которые копируются в домашний каталог каждого пользователя при его создании.

Команды управления

Для управления пользователями используются следующие команды:

- **useradd** или **adduser** – добавить нового пользователя;
- **passwd** – задать пароль для пользователя;
- **usermod** – изменить параметры учетной записи пользователя;
- **userdel** или **deluser** – удалить учетную запись пользователя.

Для управления группами используются следующие команды:

- **groupadd** – добавляет новую группу;
- **gpasswd** – устанавливает пароль группы;
- **groupmod** – изменение параметров группы;
- **groupdel** – удаление группы.

Права доступа к файлам

Создатель файла должен иметь возможность управлять списком допустимых операций над файлом и списком пользователей, которым они разрешены. В ОС Linux существуют следующие *типы доступа*:

- Read (r) – чтение;
- Write (w) – запись, переименование, удаление;
- Execute (x) – выполнение (запуск).

Права доступа состоят из трех наборов:

- 1) для владельца (кто создал файл), первые три символа;
- 2) для группы владельцев, вторые три символа;
- 3) для прочих пользователей, последние три символа.

Права доступа к файлу имеют следующий вид: -rwxrwxrwx (первый символ признак каталога "-").

Задание

Произвести настройку пользователя системы:

1. Установите пакет gnome-system-tools.
2. Откройте настройки пользователей системы из меню "Приложения" – "Настройки" – "Пользователи и группы". В списке пользователей должен отобразиться пользователь с

Вашей фамилией (обратите внимание, что пользователь root в списке не отображается).

3. Добавьте нового пользователя, назвав его по своему имени, установите пароль, сделайте скриншот, подтверждающий создание пользователя.
4. С помощью кнопки "Управление группами" откройте окно настройки групп, добавьте новую группу, назовите ее по названию своей учебной группы латинскими буквами в нижнем регистре, включите в эту группу двух своих пользователей, сделайте скриншот, подтверждающий создание группы и включение в нее пользователей.
5. С помощью файлового менеджера перейдите в каталог /tmp, создайте там каталог, дайте ему имя по имени группы латинскими буквами в нижнем регистре с добавлением времени выполнения задания.
6. Через контекстное меню (правая кнопка мыши) созданной папки вызовите пункт "Свойства", перейдите на закладку "Права", изучите установленные в данный момент времени, обратите внимание, что право исполнения не отображается – это особенность данного графического интерфейса, установите отсутствие прав у субъекта "Остальные".
7. Выйдите из системы и войдите под пользователем с Вашим именем, перейдите в каталог /tmp и попробуйте войти в созданный Вами подкаталог, сделайте скриншот сообщения о закрытом доступе.
8. Выйдите из системы и войдите под пользователем с Вашей фамилией, измените права доступа к подкаталогу в /tmp таким образом, чтобы "Остальные" по-прежнему не имели к нему доступа, а пользователи с Вашими именем и фамилией могли и читать, и записывать в данный подкаталог, сделайте скриншот с новыми правами доступа.
9. Выйдите из системы и войдите под пользователем с Вашим именем, перейдите в каталог /tmp и попробуйте войти в созданный Вами подкаталог, проверьте, есть ли у Вас право записи в него, создав в нем пустой файл, сделайте скриншот демонстрирующий корректность работы прав доступа пользователей.
10. Все проделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Пользователь root.
2. Системные пользователи.
3. Обычные пользователи.
4. Параметры пользователя.
5. Какие вы знаете команды управления пользователем?
6. Права доступа.

Лабораторная работа № 5. Файловая система Linux

Понятие файловой системы

Файловая система (ФС) – предоставляет пользователям (и процессам) ресурсы долговременной памяти компьютера. Операционная система Windows может быть установлена на файловую систему NTFS, поэтому обычно у пользователей не возникает вопросов какую ФС лучше использовать. ОС Linux имеет значительные отличия. Так, в ядро системы встроены и могут использоваться несколько файловых систем, каждая из которых оптимизирована для решения определенных задач.

Чтобы на каждом разделе можно было работать с файлами и каталогами, необходима файловая система. Кроме записи содержимого файлов на диск нужно еще хранить данные о папках, имена файлов, их размер, адрес на жестком диске, атрибуты доступа. Всем этим занимается файловая система. От файловой системы зависит очень многое: скорость работы с файлами, скорость записи и даже размер файлов. Также от стабильности файловой системы будет зависеть сохранность файлов.

Файловые системы в Linux используются не только для работы с файлами на диске, но и для хранения данных в оперативной памяти или доступа к конфигурации ядра во время работы системы. Далее будут рассмотрены типы файловых систем Linux, включая специальные файловые системы.

Основные файловые системы

Каждый дистрибутив Linux позволяет использовать одну из следующих файловых систем, имеющих свои преимущества и недостатки:

- Ext2;
- Ext3;
- Ext4;
- JFS;
- ReiserFS;
- XFS;
- Btrfs;
- ZFS.

Все они включены в ядро и могут использоваться в качестве корневой файловой системы. Рассмотрим каждую из них более подробно.

Ext2, Ext3, Ext4 (Extended Filesystem) – это стандартная файловая система для ОС Linux. Была разработана еще для Minix. Она самая стабильная из всех существующих ФС, кодовая база изменяется очень редко и эта файловая система содержит больше всего функций. Версия ext2 была разработана уже именно для Linux.

В 2001 году вышла файловая система ext3, которая стала стабильнее в работе благодаря использованию журналирования. В 2006 была выпущена версия ext4, использующаяся сейчас во всех дистрибутивах Linux. В ней было внесено много изменений, в том числе увеличен максимальный размер раздела до одного экзбайта.

JFS (Journaled File System) была разработана в IBM для AIX UNIX и использовалась в качестве альтернативы для файловых систем ext. Сейчас она используется там, где необходима высокая стабильность и минимальное потребление ресурсов. При разработке файловой системы ставилась цель создать максимально эффективную файловую систему для многопроцессорных компьютеров. Также как и ext, это журналируемая файловая система, но в журнале хранятся только метаданные, что может привести к использованию старых версий файлов после сбоев.

ReiserFS была разработана в качестве альтернативы ext3 с улучшенной производительностью и расширенными возможностями. Она разработана под руководством Ганса Райзера и поддерживает только Linux. Особенностью ФС является динамический размер блока, что позволяет упаковывать несколько небольших файлов в один блок, предотвращая фрагментацию и улучшая работу с небольшими файлами. Еще одно преимущество ФС – возможность изменять размеры разделов во время работы. Тем не менее, имеется недостаток в некоторой нестабильности работы и риске потери данных при отключении питания.

XFS – это высокопроизводительная файловая система, разработанная в Silicon Graphics для собственной операционной системы в 2001 году. Из преимуществ файловой системы можно отметить высокую скорость работы с большими файлами, отложенное выделение места, увеличение разделов во время работы и незначительный размер служебной информации. XFS – журнали-

руемая файловая система, однако в отличие от ext, в журнал записываются только изменения метаданных. Она используется по умолчанию в дистрибутивах на основе RedHat. Из недостатков ФС следует отметить отсутствие возможности уменьшения размера, сложность восстановления данных и риск потери файлов при записи, если будет неожиданное отключение питания.

Btrfs или *B-Tree File System* – это совершенно новая файловая система, которая сосредоточена на отказоустойчивости, легкости администрирования и восстановлении данных. Файловая система объединяет в себе очень много новых возможностей, таких как размещение на нескольких разделах, поддержка подтомов, изменение размера во время работы, создание мгновенных снимков, а также высокая производительность. Однако многими пользователями файловая система *Btrfs* считается нестабильной. Тем не менее, она уже используется как файловая система по умолчанию в OpenSUSE и SUSE Linux.

Другие файловые системы, такие как NTFS, FAT, HFS могут использоваться в ОС Linux, но корневая файловая система на них не устанавливается, поскольку они для этого не предназначены.

Специальные файловые системы

Ядро Linux использует специальные файловые системы, чтобы предоставить доступ пользователю и программам к своим настройкам и информации. Наиболее часто используются следующие специальные ФС:

- *tmpfs*;
- *procfs*;
- *sysfs*.

Файловая система *tmpfs* позволяет размещать любые пользовательские файлы в оперативной памяти компьютера. Достаточно создать блочное устройство нужного размера, затем подключить его к каталогу, и пользователь может записывать файлы в оперативную память. *procfs* – по умолчанию смонтирована в каталог *proc* и содержит всю информацию о запущенных в системе процессах, а также о самом ядре. *sysfs* – с помощью этой файловой системы пользователь может задавать различные настройки ядра во время выполнения.

Основные каталоги (директории)

В этом разделе рассмотрены основные каталоги (директории), которые имеются у большинства дистрибутивов ОС Linux:

- **/ – корень.** Это главный каталог в системе Linux. По сути, это и есть файловая система Linux. Здесь нет дисков, как в Windows. Вместо этого, адреса всех файлов начинаются с корня, а дополнительные разделы, usb-накопители или оптические диски подключаются в папки корневого каталога. Только пользователь "root" имеет право читать и изменять файлы в этом каталоге. *Обратите внимание*, что у пользователя "root" домашний каталог "/root", но не "/".
- **/bin – (binaries) бинарные файлы пользователя.** Этот каталог содержит исполняемые файлы. Здесь расположены программы, которые можно использовать в однопользовательском режиме или режиме восстановления. То есть утилиты, которые могут использоваться, пока не подключен каталог /usr/. Это такие общие команды, как cat, ls, tail, ps и т. д.
- **/sbin – (system binaries) системные исполняемые файлы.** Также как и /bin содержит двоичные исполняемые файлы, которые доступны на ранних этапах загрузки, когда не подключен (не подмонтирован) каталог /usr. Здесь находятся программы, которые можно выполнять только с правами суперпользователя ("root"). Это разные утилиты для обслуживания системы, например, iptables, reboot, fdisk, ifconfig, swapon и т. д.
- **/etc – (etcetera) конфигурационные файлы.** В этой директории содержатся конфигурационные файлы всех программ, установленных в системе. Кроме конфигурационных файлов, при использовании системы инициализации Init Scripts здесь находятся скрипты запуска и завершения системных сервисов, монтирования файловых систем и автозагрузки программ.
- **/dev – (devices) файлы устройств.** В ОС Linux внешние устройства являются файлами. Таким образом, все подключенные usb-накопители, клавиатуры, микрофоны, камеры – файлы в каталоге /dev/. Этот каталог содержит не совсем обычную файловую систему. Структура файловой системы Linux и содержащиеся в каталоге /dev файлы

инициализируются при загрузке системы сервисом udev. Выполняется сканирование всех подключенных устройств и создание для них специальных файлов. Это такие устройства, как: /dev/sda, /dev/sr0, /dev/tty1, /dev/usbmon0 и т. д.

- **/proc – (process) информация о процессах.** Это тоже необычная файловая система, а подсистема, динамически создаваемая ядром. Здесь содержится вся информация о запущенных процессах в реальном времени. По сути это псевдофайловая система, содержащая подробную информацию о каждом процессе, его PID, имени исполняемого файла, параметрах запуска, доступе к оперативной памяти и так далее. Также в этом каталоге можно найти информацию об использовании системных ресурсов, например, /proc/cpuinfo, /proc/meminfo или /proc/uptime.
- **/var (variable) – переменные файлы.** Название каталога /var указывает на то, что он содержит файлы, которые часто изменяются. Размер этих файлов постоянно увеличивается. Здесь содержатся файлы системных журналов, различные cash-файлы, базы данных и так далее.
Назначение каталогов Linux в директории /var/:
- **/var/log – файлы логов (файлы журналов).** В этой директории содержится большинство файлов логов всех программ, установленных в операционной системе. У многих программ есть свои подкаталоги в этой директории, например, /var/log/apache – логи веб-сервера, /var/log/squid – файлы журналов сервера squid. Если в системе что-либо перестало работать, скорее всего, информация об этом располагается в данной директории.
- **/var/lib – базы данных.** В этой директории находятся файлы баз данных, пакеты, сохраненные пакетным менеджером и т. д.
- **/var/mail – почта.** В эту директорию почтовый сервер складывает все полученные или отправленные электронные письма, здесь же могут находиться его логи и файлы конфигурации.
- **/var/spool – принтер.** Каталог отвечает за очереди печати на принтере и работу набора программ cups.

- **/var/lock – файлы блокировок.** В этой директории находятся файлы блокировок. Эти файлы означают, что определенный ресурс или файл занят и не может быть использован другим процессом. apt-get, например, блокирует свою базу данных, чтобы другие программы не могли ее использовать, пока программа с ней работает.
- **/var/run – PID процессов.** Каталог содержит файлы с PID процессов, которые могут быть использованы, для взаимодействия между программами. В отличие от каталога /run данные сохраняются после перезагрузки.
- **/tmp (temp) – временные файлы.** В этом каталоге содержатся временные файлы, созданные системой, любыми программами или пользователями. Все пользователи имеют право записи в эту директорию. Файлы удаляются при каждой перезагрузке.
- **/usr – (user applications) программы пользователя.** Каталог с большим количеством функций. Здесь находятся исполняемые файлы, исходники программ, различные ресурсы приложений и документация.
- **/usr/bin/ – исполняемые файлы.** Каталог содержит исполняемые файлы различных программ, которые не нужны на первых этапах загрузки системы, например, музыкальные плееры, графические редакторы, браузеры и так далее.
- **/usr/sbin/.** Каталог содержит двоичные файлы программ для системного администрирования, которые нужно выполнять с правами суперпользователя. Например, такие как Gparted, sshd, useradd, userdel и т. д.
- **/usr/lib/ – библиотеки.** Каталог содержит библиотеки для программ из каталогов /usr/bin или /usr/sbin.
- **/usr/local – файлы пользователя.** Каталог содержит файлы программ, библиотек, и настроек созданные пользователем. Например, здесь могут храниться программы собранные и установленные из исходников и скрипты, написанные вручную.
- **/home – домашняя директория.** В этой директории хранятся домашние каталоги всех пользователей. В них они могут хранить свои личные файлы, настройки программ и т. д. Например: /home/имя_учетной_записи и т. д.

- **/boot – файлы загрузчика.** Каталог содержит все файлы, связанные с загрузчиком системы. Это ядро `vmlinuz`, образ `initrd`, а также файлы загрузчика, находящиеся в каталоге `/boot/grub`.
- **/lib (library) – системные библиотеки.** Каталог содержит файлы системных библиотек, которые используются исполняемыми файлами в каталогах `/bin` и `/sbin`. Библиотеки имеют имена файлов с расширением `*.so` и начинаются с префикса `lib*`. Например, `libcurses.so.5.7`. Папка `/lib64` в 64 битных системах содержит 64 битные версии библиотек из `/lib`.
- **/opt (optional applications) – дополнительные программы.** В эту директорию устанавливаются проприетарные программы или драйверы. Это программы, созданные в виде отдельных исполняемых файлов самими производителями. Такие программы устанавливаются в подкаталоги `/opt/`, они очень похожи на программы Windows, все исполняемые файлы, библиотеки и файлы конфигурации находятся в одной папке.
- **/mnt (mount) – монтирование.** В этот каталог системные администраторы могут монтировать(подключать) внешние или дополнительные файловые системы.
- **/media – съемные носители.** В этот каталог система монтирует все подключаемые внешние накопители – usb-накопители, оптические диски и другие носители информации.
- **/srv (server) – сервер.** В этом каталоге содержатся файлы серверов и сервисов. Например, могут содержаться файлы веб-сервера `apache`.
- **/run – процессы.** Еще один каталог, содержащий PID файлы процессов, похожий на `/var/run`, но в отличие от него, он размещен в файловом хранилище `TMPFS`, поэтому после перезагрузки все файлы удаляются.
- **/sys (system) – информация о системе.** Назначение каталогов Linux из этой директории – получение информации о системе непосредственно от ядра. Это еще одна файловая система, организуемая ядром и позволяющая просматривать и изменить многие параметры работы системы.

Файлы устройств и монтирование

Все устройства (HDD, DVD, USB и др.) представлены в виде файлов. Они хранятся в каталоге /dev. Все HDD-диски называются однотипно /dev/sdX. X – буква от "a" до "z" соответствует порядку подключения физического диска к контроллеру. Например, если диск подключен к контроллеру первым, то его имя /dev/sda. Если подключить USB-диск, то ему будет дано имя /dev/sdb. На каждом диске может быть несколько разделов. Им даются имена, начиная с «1»: /dev/sda1, /dev/sda2 и т. д.

Для получения доступа к файлам и каталогам, находящимся на другом разделе, этот раздел нужно подмонтировать. Сменные носители монтируются автоматически (обычно к каталогу /media/<UUID-устройства>, UUID – уникальный идентификатор устройства).

Для монтирования используется команда (в Ubuntu): **sudo mount <имя_устройства> <точка_монтирования>** (например, **sudo mount /dev/sdb1 /mnt/files**).

Точка монтирования – каталог, через который будет осуществляться доступ к монтируемой файловой системе. На момент осуществления операции монтирования точка монтирования должна существовать.

Задание

Запустите обозреватель файловой системы (Приложения – Файловый менеджер) – должен открыться "домашний" каталог Вашего пользователя (/home/ваша_фамилия).

Нажатием *кнопки* ^ на панели инструментов (или сочетанием клавиш *Alt + стрелка вверх*) переместитесь на один уровень выше – в каталог /home, а затем еще выше – на уровень корневого каталога. *Обратите внимание*, что переместиться на уровень выше стало невозможно.

Найдите каталоги /home, /media, /proc, зайдите в каждый из них; нажатием *кнопки с изображением домика на панели инструментов* можно быстро переместиться в свой "домашний" каталог.

Выполните следующие действия, затем сделайте скриншоты, подтверждающие их выполнение, соберите в один документ и прикрепите в качестве ответа на задание:

1. Перейдите в каталог "Рабочий стол" (или Desktop если случайно установили систему на английском) внутри своего домашнего каталога, с помощью контекстного меню создайте там подкаталог, назовите его по следующему шаблону: ФамилияИО-Группа;
2. Перейдите в созданный подкаталог, создайте там пустой файл, дайте ему имя "Выполнено", откройте его с помощью текстового редактора (по умолчанию должен запускаться LibreOffice Writer), запишите дату и время выполнения задания;
3. Перейдите в каталог /proc, найдите файл cpuinfo, откройте его с помощью текстового редактора (по умолчанию должен запускаться LibreOffice Writer, если в нем содержимое файла не отображается – откройте с помощью другого редактора: Mousepad, Gedit, Kate, ...), ознакомьтесь со сведениями о центральном процессоре, как их "видит" операционная система;
4. Там же, в каталоге /proc, найдите файл cmdline, откройте его с помощью текстового редактора, попробуйте определить, с помощью какого объекта файловой системы была загружена операционная система.

Все сделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Что такое файловая система?
2. Файловые системы Linux.
3. Специальные файловые системы.
4. Основные каталоги.
5. Как получить доступ к файлам?

Лабораторная работа № 6. Резервное копирование данных

Общие сведения

Существует много способов организации backup (бэкап) на CentOS/Debian/Ubuntu серверах – бесплатные утилиты, самописные скрипты с использованием tar, система бэкапа bacula и др.

Использование популярной утилиты **rsync** на серверах под управлением CentOS/Debian/Ubuntu является простым, удобным и быстрым способом настройки инкрементного backup.

Способ одинаково работает на этих системах, есть небольшие отличия только в самой установке rsync, о чем будет отдельно упомянуто для каждой системы.

Настройка rsync

Приступаем к настройке. Логика наших бэкапов будет следующая. При первом запуске делаем полный бэкап интересующей нас информации в папку **current**. Потом раз в сутки сверяем имеющийся архив с источником и делаем его вновь актуальным, перезаписывая все изменившиеся файлы, но при этом не удаляем их, а складываем в папку **increment**, где каждый день создается папка с именем в виде даты, в которую складываются все измененные файлы за текущий день. Таким образом, у нас всегда будет полный архив, актуальный на момент последней синхронизации, плюс набор папок за каждый день с изменившимися в этот день файлами. Сколько дней хранить – можно выбрать по необходимости. Получается (рисунок 31):

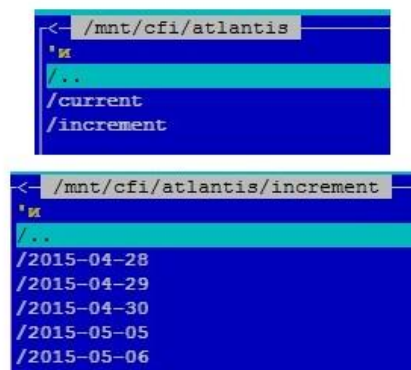


Рисунок 31 – Архив изменений

При этом подключение и работа rsync будет проходить по своему отдельному порту **tcp 873**. *Не забудьте настроить iptables* и открыть этот порт. Приступаем к реализации. В первую очередь настраиваем rsync на серверах источниках информации, с которых мы будем забирать данные для backup.

Создаем файл конфигурации rsync:

```
# mcedit /etc/rsyncd.conf
pid file = /var/run/rsyncd.pid
log file = /var/log/rsyncd.log
transfer logging = true
munge symlinks = yes
# папка источник для бэкапа
[data]
path = /data
uid = root
read only = yes
list = yes
comment = Data backup Dir
auth users = backup
secrets file = /etc/rsyncd.scr
```

Создаем файл с учетными данными для подключения:

```
# mcedit /etc/rsyncd.scr
backup:12345, где backup – имя пользователя, 12345 – пароль.
```

Делаем права на чтение только root, иначе rsync не запустится:

```
# chmod 0600 /etc/rsyncd.scr
```

После настройки перезапускаем rsync. На Centos:

```
# systemctl restart rsyncd
```

На Debian/Ubuntu:

```
# systemctl restart rsync
```

Переходим на сервер приемник, в котором будут храниться архивные копии с серверов источников. Там создаем скрипт инкрементного бэкапа с использованием rsync:

```
# mcedit /root/bin/backup-server1.sh
#!/bin/bash
date
```

```

# Папка, куда будем складывать архивы
syst_dir=/backup/
# Имя сервера, который архивируем
srv_name=server1
# Адрес сервера, который архивируем
srv_ip=10.10.1.55
# Пользователь rsync на сервере, который архивируем
srv_user=backup
# Ресурс на сервере для бэкапа
srv_dir=data
echo "Start backup ${srv_name}"
# Создаем папку для инкрементных бэкапов
mkdir -p ${syst_dir}${srv_name}/increment/
# Запускаем непосредственно бэкап с параметрами
/usr/bin/rsync -avz --progress --delete --password-file=/etc/rsyncd.scr
t
${srv_user}@${srv_ip}::${srv_dir} ${syst_dir}${srv_name}/current/ --
backup --backup-dir=${syst_dir}${srv_name}/increment/^date +%Y-
%m-%d`/
# Чистим папки с инкрементными архивами старше 30-ти дней
/usr/bin/find ${syst_dir}${srv_name}/increment/ -maxdepth 1 -type d -
mtime +30 -exec rm -rf {} \;
date
echo "Finish backup ${srv_name}"
Делаем скрипт исполняемым:
# chmod 0744 /root/bin/backup-server1.sh
Создаем файл с паролем для авторизации на сервере источнике:
# mcedit /etc/rsyncd.scr

```

Назначаем *права на чтение только root*, иначе rsync выдаст ошибку: `ERROR: password file must not be other-accessible`
Исправляем это:

```
# chmod 0600 /etc/rsyncd.scr
```

Теперь можно запускать скрипт и ожидать его выполнения. Если получите ошибку на клиенте:

```
rsync: opendir "/" (in data) failed: Permission denied
```

и вот эту на сервере:

```
SELinux is preventing rsync from getattr access on the file
```

Проверьте настройки SELinux. Это он блокирует доступ к файлам. Нужно либо отключить selinux, либо настроить. В данном случае настройка простая:

```
# setsebool -P rsync_full_access on
```

Осталось добавить скрипт в cron:

```
# mcedit /etc/crontab 23 * * * root /root/bin/backup-server1.sh
```

Обычно создают несколько скриптов для каждого сервера отдельно. Потом объединяют их запуск в одном общем скрипте и уже его добавляют в cron. А потом по мере необходимости редактируют уже его, добавляют или удаляют сервера.

Копирование rsync через ssh

Rsync может работать через ssh. Это избавляет от необходимости настраивать отдельно службу и авторизацию, но при этом будут использоваться системные учетные записи.

Для того, чтобы скопировать файлы с помощью rsync по ssh нет необходимости запускать службу, настраивать файл конфигурации, создавать файл с авторизацией. Можно просто запустить примерно такую команду на передачу файлов:

```
# /usr/bin/rsync -avz --progress --delete
```

```
root@10.1.6.221:/data/mysql_dump /backup
```

Будьте внимательны при использовании ключа --delete. Не перепутайте источник, откуда качаете файлы, с приемником, куда копируете. Если их перепутать и в качестве источника указать пустую папку, а в качестве приемника с файлами, файлы будут удалены моментально и без предупреждения.

Если для подключения используется публичный ключ или нестандартный порт ssh, указать эти параметры можно следующим образом:

```
# /usr/bin/rsync -avz --progress --delete -e "ssh -p 1234 -i
```

```
/root/.ssh/id_rsa.pub" root@10.1.6.221:/data/mysql_dump /backup
```

Настройка исключений

Можно настроить исключение файлов или каталогов при копировании с помощью rsync. Делается это с помощью ключа --exclude или --exclude-from. Первый позволяет указать исключение непосредственно в команде на исполнение. Второй позволяет загружать список исключений из файла.

Пример первого варианта:

```
# /usr/bin/rsync -avz --progress --delete --exclude='*.jpeg'
```

```
root@10.1.6.221:/var/www/html /backup
```

Вы скопируете директорию с сайтом, исключив из нее все картинки с расширением .jpeg.

Таких исключений можно добавить сколько угодно в одной команде. Но удобнее их выносить в отдельный файл.

Пример:

```
# /usr/bin/rsync -avz --progress --delete --exclude-from=exclude.lst
root@10.1.6.221:/var/www/html /backup
```

Содержимое файла *exclude.lst*.

```
*/bitrix/managed_cache/MYSQL/*
```

```
*/bitrix/backup/*
```

```
*/bitrix/html_pages/site.ru/*
```

```
*/upload/resize_cache/*
```

```
*/bitrix/cache/*
```

```
*/log.txt
```

```
*/rating/logs/my_file.log
```

Это пример исключений для bitrix сайта.

Все эти команды и исключения можно комбинировать и использовать в скриптах как показано в первом примере.

Ротация логов rsync

Ранее указали в настройках службы rsyncd ведение лога в файл */var/log/rsyncd.log*. Необходимо настроить ротацию этого лога, чтобы он не рос до бесконечности. На больших файловых серверах он очень быстро вырастет до сотен мегабайт и более.

Для этого создаем в папке */etc/logrotate.d* файл с конфигурацией ротации:

```
# mcedit /etc/logrotate.d/rsyncd
```

```
/var/log/rsyncd.log {
size=500k
compress
rotate 4
missingok
notifempty
}
```

С такими настройками ротация будет происходить каждый раз, когда файл лога превысит размер в 500 КБ. Храниться будут 4

версии лог-файла. Эти настройки можно поменять по своему усмотрению.

Когда используете ротацию по размеру файла, не забывайте проверять, что она у вас корректно работает. Так как разных дистрибутивах есть нюансы на этот счет.

Задание

С помощью утилиты `rsync` произвести резервное копирование файлов и папок системы. Проверить результат копирования.

Все сделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Для чего нужно резервное копирование?
2. Для чего нужна утилита `rsync`?
3. Ротация логов.
4. Команды для резервного копирования.
5. Как настроить исключения?

Лабораторная работа № 7. Работа с Syslog в Linux и обработка журнальных файлов

Системное журналирование

Linux и приложения, работающие под его управлением, могут генерировать различные сообщения, которые записываются в разные файлы логов. Linux использует набор конфигурационных файлов, директорий, программ, команд и демонов для создания, хранения и обработки логируемых сообщений. Знание того, где система хранит свои лог-файлы и как использовать соответствующие команды просмотра логов, может помочь сэкономить драгоценное время при устранении неполадок.

Функция системного журналирования ("логи" или логирование) – это основной источник информации о работе системы и ошибках. **Журналирование** может осуществляться на локальной системе, а также сообщения журналирования могут пересылаться на удаленную систему, кроме того, в конфигурационном файле `/etc/syslog.conf` (в некоторых новых дистрибутивах заменен на `/etc/rsyslog.conf`) возможна тонкая регулировка уровня журналирования. Журналирование осуществляется при помощи демона **syslogd** (**rsyslogd** – в некоторых новых дистрибутивах), который обычно получает входную информацию при помощи **сокета /dev/log** (локально) или с **udp-порта 514** (с удаленных машин). На рисунке 32 представлен демон **syslogd** с параметрами.

```
syslog-server:~# ls -l /dev/log
srw-rw-rw- 1 root root 0 Дек 17 06:25 /dev/log
```

Рисунок 32 – Демон **syslogd**

В случае локального журналирования главным файлом – хранителем информации, обычно является `/var/log/messages`, но в большинстве инсталляций используются и многие другие файлы, которые могут быть тщательно настроены с помощью вышеуказанного конфигурационного файла. Например, может возникнуть необходимость выделить в отдельный лог сообщения, рождаемые демоном электронной почты.

Управление типом и подробностью журналируемой информации. Конфигурационный файл **syslog.conf**

Файл **syslog.conf** является главным конфигурационным файлом для демона **syslogd**. Конфигурационный файл **syslog.conf** представляет собой **набор правил**.

Каждое **правило** есть – строка, состоящая из *селектора* и *действия*, разделенных пробелом или табуляцией.

Селектор представляет собой запись в виде *источник.приоритет*. (*источник* иногда именуют – *категорией*). Селектор может состоять из нескольких записей *источник.приоритет*, разделенных символом ";". Можно указывать несколько источников в одном селекторе (через запятую). Поле **действие** – устанавливает журналируемое действие для селектора.

Получив сообщение для записи в журнал (от **klogd**, от локальной или удаленной программы), **syslogd** для каждого правила проверяет, не подходит ли сообщение под шаблон, определяемый селектором. Если подходит, то выполняется указанное в правиле действие. Для одного сообщения может быть выполнено произвольное количество действий (т. е. обработка сообщения не прекращается при первом успехе).

Сообщения с уровнем, равным или выше указанного в селекторе, и *источником, равным указанному* в селекторе, считается *подходящим*.

Звездочка перед точкой соответствует *любому источнику*, **после точки** – *любому уровню*.

Слово **none** после точки – *никакому уровню* для данного источника. Можно указывать несколько источников в одном селекторе (через запятую).

Источник (он же категория) может быть следующим:

- 0 – **kern** – Сообщения ядра.
- 1 – **user** – Сообщения пользовательских программ.
- 2 – **mail** – Сообщения от почтовой системы.
- 3 – **daemon** – Сообщения от тех системных демонов, которые в отличие от FTP или LPR не имеют выделенных специально для них категорий.

- 4 – **auth** – Все что связано с авторизацией пользователей, вроде login и su (безопасность/права доступа).
- 5 – **syslog** – Система протоколирования может протоколировать сообщения от самой себя.
- 6 – **lpr** – Сообщения от системы печати.
- 7 – **news** – Сообщения от сервера новостей (в настоящее время не используется).
- 8 – **uucp** – Сообщения от UNIX-to-UNIX Copy Protocol. Это часть истории UNIX и вероятнее всего она вам никогда не понадобится (хотя до сих пор определенная часть почтовых сообщений доставляется через UUCP).
- 9 – **cron** – Сообщения от системного планировщика.
- 10 – **authpriv** – То же самое, что и auth, однако сообщения этой категории записываются в файл, который могут читать лишь некоторые пользователи (возможно, эта категория выделена потому, что принадлежащие ей сообщения могут содержать открытые пароли пользователей, которые не должны попадать на глаза посторонним людям, следовательно файлы протоколов должны иметь соответствующие права доступа).
- 11 – **ftp** – При помощи этой категории вы сможете сконфигурировать ваш FTP сервер, что бы он записывал свои действия.
- 12 – **NTP** – сообщения сервера времени.
- 13 – **log audit**.
- 14 – **log alert**.
- 15 – **clock daemon** – сообщения демона времени.
- с 16 по 23 – **local0–local7** – Зарезервированные категории для использования администратором системы. Категория local7 обычно используется для сообщений, генерируемых на этапе загрузки системы.
- **mark** (не имеющая цифрового эквивалента) – присваивается отдельным сообщениям, формируемым самим демоном syslogd.

Под **приоритет (степени важности) сообщений** заданы 8 уровней важности, которые кодируются числами от 0 до 7:

- 0 – **emerg** (старое название **PANIC**) – Чрезвычайная ситуация. Система неработоспособна.
- 1 – **alert** – Тревога! Требуется немедленное вмешательство.

- 2 – **crit** – Критическая ошибка (критическое состояние).
- 3 – **err** (старое название **ERROR**) – Сообщение об ошибке.
- 4 – **warning** (старое название **WARN**) – Предупреждение.
- 5 – **notice** – Информация о каком-то нормальном, но важном событии.
- 6 – **info** – Информационное сообщение.
- 7 – **debug** – Сообщения, формируемые в процессе отладки.

Согласно действию, указанному в правиле, сообщение может быть записано в следующие назначения:

- **Обычный файл.** Задается полным путем, начиная со слеша (/). Поставьте перед ним дефис (-), чтобы отменить синхронизацию файла после каждой записи. Это может привести к потере информации, но повысить производительность.
- **Именованные каналы.** Размещение перед именем файла символа канала (|) позволит использовать **fifo (first in – first out**, первый пришел – первый вышел) или **именованный канал (named pipe)** в качестве приемника для сообщений. Прежде чем запускать (или перезапускать) `syslogd`, необходимо создать `fifo` при помощи команды `mkfifo`. Иногда `fifo` используются для отладки.
- **Терминал и консоль.** Терминал, такой как `/dev/console`.
- **Удаленная машина.** Чтобы сообщения пересылались на другой хост, поместите перед именем хоста символ (@). *Обратите внимание*, что сообщения не пересылаются с принимающего хоста. (Для работы данного назначения на клиенте и сервере в файле `/etc/services` должна быть прописана строчка `syslog 514/udp`, и открыт UTP-порт 514).
- **Список пользователей.** Разделенный запятыми список пользователей, получающих сообщения (если пользователь зарегистрирован в системе). Сюда часто включается пользователь `root`.
- **Все зарегистрированные пользователи.** Чтобы известить всех зарегистрированных пользователей при помощи команды `wall`, используйте символ звездочки (*).

*Пример несложного **syslog.conf** (рисунок 33):*

```
# Все сообщения ядра выдавать на консоль.
#kern.*                                /dev/console

# Все логи уровня info или выше, кроме сообщений электронной почты, а так же
# не логировать сообщения аутентификации и сообщений демона cron!
*.info;mail.none;authpriv.none;cron.none    /var/log/messages

# Записывать в отдельный файл сообщения, содержащие конфиденциальную
# информацию аутентификации, независимо от их уровня.
authpriv.*                                  /var/log/secure

# Все сообщения почтовой системы тоже записывать в отдельный файл.
mail.*                                       -/var/log/maillog

# Логировать сообщения планировщика в файл /var/log/cron
cron.*                                      /var/log/cron

# Сообщения о чрезвычайных ситуациях должны немедленно получить
# все пользователи системы
*.emerg                                     *

# Сохранять сообщения новостей уровня crit и выше в отдельный файл.
uucp,news.crit                             /var/log/spooler

# Сохранять сообщения загрузки в boot.log
local7.*                                    /var/log/boot.log
```

Рисунок 33 – Syslog.conf

Как и во многих конфигурационных файлах, синтаксис следующий:

- Строки, начинающиеся с #, и пустые строки игнорируются.
- Символ * может использоваться для указания всех категорий или всех приоритетов.
- Специальное ключевое слово none указывает, что журналирование для этой категории не должно быть выполнено для этого действия.
- Дефис перед именем файла (как -/var/log/maillog в этом примере) указывает, что после каждой записи журнал не должен синхронизироваться. В случае аварии системы вы можете потерять информацию, но отключение синхронизации позволит повысить производительность.

В синтаксисе конфигурационного файла можно поставить перед приоритетом **знак !**, чтобы показать, что действие не должно применяться, **начиная с этого уровня и выше**. Подобным образом, перед приоритетом можно поставить **знак =**, чтобы показать, что правило применяется только к этому уровню, или **!=**,

чтобы показать, что правило применяется ко всем уровням, кроме этого. Ниже (рисунок 34) показан пример (man syslog.conf можно найти множество других примеров):

```
# Посылать все сообщения ядра в /var/log/kernel.
# Посылать все сообщения уровня critical и higher на удаленную машину sysloger и на консоль
# Посылать все сообщения уровня info, notice и warning в /var/log/kernel-info
#
kern.*                /var/log/kernel
kern.crit              @sysloger
kern.crit              /dev/console
kern.info;kern.!err    /var/log/kernel-info

# Посылать все сообщения почтовой системы, кроме уровня info в /var/log/mail.
mail.*;mail.!=info    /var/log/mail
```

Рисунок 34 – Man syslog.conf

Запуск демона syslogd

Запуск демона протоколирования запускаются на этапе инициализации системы посредством скрипта `/etc/rc.d/init.d/syslog`, однако для того, чтобы задать параметры запуска, нет необходимости корректировать этот скрипт – начиная с версии 7.2, опции запуска считываются из отдельного конфигурационного файла `/etc/sysconfig/syslog` (`/etc/default/syslog` в *debian*).

Вот некоторые возможные **параметры запуска демона syslogd**:

- **-a /folder/socket** – указание дополнительного слушающего сокета (*не забудьте предварительно создать сокет*);
- **-d** – отладочный режим. При этом демон не переходит в фоновый режим и выдает все сообщения на текущий терминал;
- **-f имя-конфигурационного-файла**. Задает имя альтернативного конфигурационного файла, который будет использоваться вместо заданного по умолчанию `/etc/syslog.conf`;
- **-l список-хостов** – задание списка хостов, имена которых не должны записываться с указанием полного доменного имени (FQDN – Full Qualified Domain Name);
- **-m минут** – запущенный без этой опции `syslogd` через каждые 20 минут записывает в протокол сообщения категории `mark` (временные отметки). С помощью опции `-m` можно либо изменить интервал между отметками, либо вовсе отменить выдачу таких сообщений;

- **-p socket** – задание альтернативного сокета UNIX (вместо прослушиваемого по умолчанию /dev/log);
- **-r** – разрешение принимать сообщения от удаленных хостов;
- **-x** – запрет определения имени хоста по его адресу для предотвращения зависания при работе на одном хосте с сервером DNS;
- **-v** – показать версию и закончить работу.

После запуска демона **syslogd** создается файл статуса `/var/lock/subsys/syslog` нулевой длины, и файл с идентификационным номером процесса `/var/run/syslogd.pid`.

С помощью команды `kill -SIGNAL `cat /var/run/syslogd.pid`` можно послать демону **syslogd** один из следующих сигналов:

- **SIGHUP** – перезапуск демона;
- **SIGTERM** – завершение работы;
- **SIGUSR1** – включить/выключить режим отладки.

Вообще в системе запускаются **два демона** протоколирования – **syslogd** и **klogd**. Оба демона входят в состав **пакета syslogd**.

Демон **klogd** отвечает за журналирование событий, происходящих в **ядре системы**. Необходимость в отдельном демоне **klogd** объясняется тем, что ядро не может использовать стандартную функцию `syslog`. Дело в том, что стандартные библиотеки языка разработки C (включая ту библиотеку, в которой находится функция `syslog`) предназначены для использования только *обычными приложениями*. Поскольку ядро тоже нуждается в функциях журналирования, в него включены свои библиотеки, недоступные приложениям. Поэтому ядро использует свой собственный механизм генерации сообщений. Демон **klogd** предназначен для организации обработки этих сообщений. В принципе он может производить такую обработку полностью самостоятельно и независимо от **syslogd**, например, записывая эти сообщения в файл, но в большинстве случаев используется принятая по умолчанию настройка **klogd**, при которой все сообщения от ядра пересылаются тому же демону **syslogd**.

Автоматическая ротация (обновление заполненных файлов) и архивирование журналов

Со временем, файл журнала имеет свойство увеличиваться, особенно при интенсивной работе какого-либо сервиса. Соответст-

венно, необходимо иметь возможность контролировать размер журналов. Это делается при помощи команды **logrotate**, которая обычно выполняется демоном **cron**. Главная цель команды **logrotate** состоит в том, чтобы *периодически создавать резервные копии журналов и создавать новые чистые журналы*. Сохраняется несколько поколений журналов и, когда завершается срок жизни журнала последнего поколения, он может быть заархивирован (сжат). Результат может быть отправлен по почте, например, ответственному за ведение архивов.

Для определения порядка ротации и архивирования журналов используется **конфигурационный файл /etc/logrotate.conf**. Для разных журналов можно задать разную периодичность, например, ежедневно, еженедельно или ежемесячно, кроме того, можно регулировать количество накапливаемых поколений, а также указать, будут ли копии архивов отправляться ответственному за ведение архивов и, если будут, когда. Ниже показан *пример файла /etc/logrotate.conf* (рисунок 35).

Глобальные опции размещаются в начале **файла logrotate.conf**. Они используются по умолчанию, если где-то в другом месте не задано ничего более определенного. В примере ротация журналов происходит **еженедельно** и резервные копии сохраняются в течение **четырех** недель. Как только производится ротация журнала, на месте старого журнала автоматически создается новый. **Файл logrotate.conf** может содержать спецификации из других файлов. Так, в него включаются все файлы из каталога **/etc/logrotate.d**.

В примере показаны специальные правила для **/var/log/wtmp** и **/var/log/btmp** (хранящие информацию об удачных и неудачных попытках входа в систему), ротация которых происходит ежемесячно. Если файлы отсутствуют, сообщение об ошибке не выдается. Создается новый файл и сохраняется только одна резервная копия. По достижении резервной копией последнего поколения она удаляется, поскольку не определено, что следует с ней делать.


```

# сначала заданы параметры "по-умолчанию" (глобальные опции)
# обновлять файлы журнала еженедельно
weekly

# хранить архив логов за 4 последние недели
rotate 4

# создавать новый (пустой) файл после ротации (обновления)
create

# раскомментируйте, если желаете, чтобы сохраненные файлы сжимались
#compress

# включить настройки ротации из указанного каталога
include /etc/logrotate.d

# не хранить wtmp, или btmp -- настройки ротации данных журналов следующие:
/var/log/wtmp {
    missingok
    monthly
    create 0664 root utmp
    rotate 1
}

/var/log/btmp {
    missingok
    monthly
    create 0664 root utmp
    rotate 1
}

# специфичные системные журналы могут быть настроены ниже

```

Рисунок 35 – Пример файла logrotate.conf

Резервные копии журналов могут также создаваться, когда журналы достигают определенного размера, и могут быть созданы скрипты из наборов команд для выполнения до или после операции резервного копирования.

Пример (рисунок 36):

```

/var/log/messages {
    rotate 5
    mail logadmin@syslogger
    size 100k
    postrotate
    /usr/bin/killall -HUP syslogd
    endscrip
}

```

Рисунок 36 – Пример скриптов из наборов команд

В этом примере ротация `/var/log/messages` производится по достижении им размера 100 КБ. Накапливается пять резервных копий, и когда истекает срок жизни самой старой резервной копии, она отсылается по почте на адрес `logadmin@sysloger`. Командное слово `postrotate` включает скрипт, перезапускающий демон `syslogd` после завершения ротации путем отправки сигнала `HUP`. Командное слово `endscript` необходимо для завершения скрипта, а также в случае, если имеется скрипт `prerotate`.

Параметры, задаваемые в конфигурационном файле `logrotate.conf`:

- **compress | nocompress** (старые версии сжимаются или не сжимаются с помощью `gzip`).
- **Compresscmd** (задает программу сжатия, по умолчанию – `gzip`).
- **uncompresscmd** (задает программу разжатия, по умолчанию – `ungzip`).
- **compressext** (задает суффикс для сжатых файлов).
- **compressoptions** (задает параметры программы сжатия; по умолчанию – «-9», т. е. максимальное сжатие для `gzip`).
- **copytruncate | nocopytruncate** (обычно старая версия переименовывалась и создавалась новая версия журнала; при задании этого параметра `logrotate` копирует журнал в новый файл, а затем обрезает старый; используется, если программа, создающая журнал, не умеет его закрывать; теряются записи, сделанные в промежутке между копированием и обрезанием).
- **create** [права-доступа владелец группа] | **nocreate** (сразу после переименования старой версии журнала и до вызова `postrotate` создается новый журнал с указанными атрибутами – права доступа задаются в восьмеричном виде, как в `chmod.2`; если атрибуты не указаны, то берутся от старого журнала).
- **daily** (смена версий в серии происходит ежедневно).
- **delaycompress | nodelaycompress** (некоторые программы не сразу закрывают журнал, в этом случае сжатие надо отложить до следующего цикла).
- **errors email** (кому направлять сообщения об ошибках).
- **extension** *суффикс* (задается суффикс, добавляемый к именам файлов при ротации перед суффиксом сжатия).
- **ifempty | notifempty** (смена версий даже если файл пуст; действует по умолчанию).

- **include** *имя-файла* | *имя-директории* (текстуально подставить файл или все файлы из указанной директории; не включаются поддиректории, специальные файлы и файлы с суффиксами из списка исключений; нельзя использовать внутри секции).
- **mail** *адрес* | **nomail** (когда смена версий приводит к необходимости удалить старый журнал, то послать его по указанному адресу).
- **mailfirst** (посылать не удаляемую версию журнала, а первую).
- **maillast** (посылать удаляемую версию журнала; действует по умолчанию).
- **missingok** | **nomissingok** (не посылать сообщения об ошибке, если журнал отсутствует).
- **monthly** (смена версий происходит ежемесячно).
- **olddir** *директория* | **noolddir** (во время смены версий журнал перемещается в указанную директорию; заданный каталог должен размещаться на том же физическом устройстве).
- **postrotate** (все дальнейшие строчки до строки `endscript` исполняются как команды `shell` после процесса смены версии).
- **prerotate** (все дальнейшие строчки до строки `endscript` исполняются перед процессом смены версии).
- **rotate** *число* (сколько старых версий хранить; если 0, то ни одной).
- **size** *байт* (смена версии происходит, если размер журнала превысил указанное число; можно использовать суффиксы «k» – килобайт – и «M» – мегабайт).
- **sharedscripts** | **nosharedscripts** (выполнять команды `prerotate` и `postrotate` только один раз для всех файлов, описанных в секции).
- **tabooext** [**+**] *список-суффиксов* (задание списка суффиксов-исключений для `include`; если указан знак «плюс», то дополнение, иначе замена; по умолчанию: `.rpmorig`, `.rpmnew`, «v», `.swp` и «~»).
- **weekly** (смена версий происходит еженедельно).

Изучение и мониторинг журналов

Записи в журналах обычно содержат метку времени, имя хоста, на котором выполняется описываемый процесс, и имя процесса. Просматривать журналы можно при помощи программы

постраничного вывода, например, less, искать определенные записи (например, сообщения ядра от определенного демона) можно при помощи команды grep (рисунок 37):

```
[root@syslog-server ~]# less /var/log/messages
[root@syslog-server ~]# grep "ppp" /var/log/messages | tail
Dec 17 16:34:25 proxy pppd[7843]: Connection terminated.
Dec 17 16:34:25 proxy pppd[7843]: Exit.
Dec 17 16:35:57 proxy pppd[11345]: LCP terminated by peer (^P]kV^@<M-Mt^@^@^@)
Dec 17 16:35:57 proxy pppd[11345]: pptpd-logwtm.so ip-down ppp2
Dec 17 16:35:57 proxy pppd[11345]: Connect time 14.8 minutes.
Dec 17 16:35:57 proxy pppd[11345]: Sent 130192 bytes, received 53946 bytes.
Dec 17 16:35:57 proxy pppd[11345]: Modem hangup
Dec 17 16:35:57 proxy pppd[11345]: Connection terminated.
Dec 17 16:35:57 proxy pppd[11345]: Script /etc/ppp/ip-down finished (pid 12084), status = 0x1
Dec 17 16:35:57 proxy pppd[11345]: Exit.
```

Рисунок 37 – Постраничный вывод

Компьютер может работать не постоянно и выключаться, например на ночь. Поэтому в /var/log/messages записи хранятся циклически от запуска компьютера к выключению, это можно заметить по сообщениям (рисунок 38):

```
Дек 17 08:32:56 syslog-server syslogd 1.4-0: restart.
Дек 17 08:32:56 syslog-server syslog: запуск syslogd succeeded
Дек 17 08:32:56 syslog-server kernel: klogd 1.4-0, log source = /proc/kmsg started.
Дек 17 08:32:56 syslog-server syslog: запуск klogd succeeded
```

Рисунок 38 – Записи messages

Далее в файле протокола (рисунок 39) можно обнаружить версию ядра, параметры его запуска, информацию о типе процессора и объеме ОЗУ (Оперативное запоминающее устройство):

```
Dec 17 08:32:56 syslog-server kernel: Kernel command line:
auto BOOT_IMAGE=linux ro root=303
BOOT_FILE=/boot/vmlinuz-2.4.2-2
Dec 17 08:32:56 syslog-server kernel: Memory: 125652k/130560k available (1365k kernel code,
Dec 17 08:32:56 syslog-server kernel: CPU: Intel(R) Pentium(R) 4 CPU 1.60GHz stepping 02
```

Рисунок 39 – Параметры ядра

Так же, в этом файле можно найти информацию о дисковой памяти (включая информацию о геометрии диска, структуре разделов и используемых прерываниях), информацию о периферийных устройствах, о запуске отдельных служб и сервисов, информацию о подключении файловых систем и сообщения о входе пользователей в систему, а также сообщения об ошибках.

Иногда может возникнуть необходимость мониторинга системных журналов с целью поиска текущих событий. Например, можно попробовать поймать редко случающееся событие в тот момент, когда оно произошло. В таком случае можно использовать команду *tail* с опцией *-f* для отслеживания содержимого системного журнала. Пример (рисунок 40):

```
[root@syslog-server~]# tail -f /var/log/messages | grep syslog-server
Dec 17 16:46:09 syslog-server pppd[12548]: pptpd-logwtm.so ip-up ppp0 maikop 94.77.0.150
Dec 17 16:46:09 syslog-server pppd[12548]: Script /etc/ppp/ip-up finished (pid 12552), status
Dec 17 16:46:49 syslog-server pptpd[12581]: CTRL: Client 85.175.197.65 control connection star
Dec 17 16:46:49 syslog-server pptpd[12581]: CTRL: Starting call (launching pppd, opening GRE)
Dec 17 16:46:49 syslog-server pppd[12582]: Plugin /usr/lib/pptpd/pptpd-logwtm.so loaded.
```

Рисунок 40 – Результат команды *tail* с опцией *-f*

Кроме файлов-журналов, указанных в */etc/syslog.conf*, существуют так же и другие файлы, например файл */var/log/dmesg*, который хранит информацию о процессе загрузки системы до запуска *syslogd*, а так же файлы */var/log/lastlog*, */var/log/wtmp*, */var/log/btmp*, имеющие двоичный формат и хранящие информацию о последнем входе пользователя в систему, о всех удачных входах пользователей в систему и о всех неудачных входах пользователей в систему соответственно. Так же в каталоге */var/log/* могут находиться лог-файлы таких демонов как веб-сервер или прокси-сервер. Формат данных файлов аналогичен журналам *syslogd*.

Syslog не содержит никаких средств защиты от подделок сообщений. Кроме того, использование протокола UDP позволяет злоумышленникам посылать сообщения от имени любого хоста. Локальная сеть должна быть защищена экраном от приема пакетов с поддельными локальными адресами (но это не мешает посылать поддельные сообщения изнутри локальной сети) и от приема пакетов снаружи на порт 514/udp.

Протокол syslog и UDP не обеспечивают гарантированной доставки (сообщения могут быть потеряны при перегрузке сети или перехвачены, поврежденные сообщения удаляются без предупреждения), правильной последовательности доставки (сообщение о завершении процесса может прийти раньше сообщения о его запуске), приоритетной доставки.

Конфиденциальность сообщений не обеспечивается, так как они передаются открытым текстом.

Если при настройке генератора сообщений указать неправильный адрес коллектора или релея, то никаких сообщений об ошибке не будет – сообщения будут удаляться (или записываться в чужой журнал).

Были предложены несколько проектов улучшения протокола syslog. Например, документ RFC 3195 предлагает систему протоколирования (syslog-conn), основанную на TCP, обеспечивающую гарантированную доставку сообщений в правильной последовательности. Проект syslog-sign предлагает обеспечить аутентификацию, упорядоченность, целостность сообщений и обнаружение пропавших сообщений за счет генерации специальных сообщений, содержащих цифровую подпись (signature) блока предыдущих сообщений с сохранением стандартного протокола и формата syslog и использованием UDP.

Задание

Произвести системное журналирование:

1. Найти в системе файл syslog.conf.
2. Проанализировать, расшифровать.
3. Произвести запуск демона syslogd.
4. Произвести ротацию и архивирование журналов.
5. Все сделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Что подразумевается под системным журналированием?
2. Что такое селектор?
3. Каким может быть источник?
4. Приоритет сообщений.
5. Параметры запуска демона syslogd.

Лабораторная работа № 8. Установка программного обеспечения в Linux

Установка программного обеспечения

В операционной системе Debian программное обеспечение устанавливается с помощью менеджеров пакетов – программ, которые автоматически скачивают программы из репозитория, устанавливают и удаляют их. В Debian используется **apt** – пакетный менеджер, разработанный специально для Debian. Также для установки пакетов формата .deb может использоваться **dpkg** – низкоуровневое ПО, которое работает непосредственно с форматом пакетов .deb и не обрабатывает зависимости программ.

Установка программ может осуществляться через командную строку или графический интерфейс.

Установка программ Debian с помощью пакетного менеджера

Одна из самых сложных задач, возникающих в процессе установки Unix-программ это отслеживание зависимостей.

Любая сложная программа в своей работе использует возможности, предоставляемые другими модулями, системными библиотеками и т. д. Таким образом, появляется зависимость – если мы хотим установить программу «А», которая при работе использует библиотеки программы «Б», сначала необходимо установить программу «Б» (у которой, в свою очередь, могут быть свои зависимости). Таким образом, чем больше программ и зависимостей появляется на сервере, тем сложнее их отслеживать и управлять ими. Отслеживание можно делать с помощью пакетного менеджера APT (Advanced Packaging Tool).

Установка пакетного менеджера

Эволюция методов установки и управления программными пакетами со временем дошла до использования пакетных менеджеров, плотно интегрированных с репозиториями. В репозиториях содержатся упакованные файлы программ с данными об их зависимостях. После установки к операционной системе уже подключены системные репозитории: с их помощью можно обновлять ОС и устанавливать программные пакеты, которые были

адаптированы и оптимизированы для работы с этой версией операционной системы.

Перед началом установки пакетов рекомендуется обновить данные о версиях и зависимостях в репозитории следующей командой (рисунок 41–42):

```
sudo apt-get update
```

Рисунок 41 – Обновление данных командой

```
root@Ubuntu1804x64:~# sudo apt-get update
Hit:1 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:4 http://archive.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [817 kB]
Get:6 http://archive.ubuntu.com/ubuntu bionic-updates/main Translation-en [288 kB]
Get:7 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages [24.1 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic-updates/restricted Translation-en [6,620 B]
Get:9 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [1,033 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic-updates/universe Translation-en [319 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages [9,284 B]
Get:12 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse Translation-en [4,508 B]
Get:13 http://archive.ubuntu.com/ubuntu bionic-backports/universe amd64 Packages [4,028 B]
Get:14 http://archive.ubuntu.com/ubuntu bionic-security/main amd64 Packages [593 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic-security/main Translation-en [194 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic-security/restricted amd64 Packages [15.1 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic-security/restricted Translation-en [4,684 B]
Get:18 http://archive.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [627 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic-security/universe Translation-en [210 kB]
Get:20 http://archive.ubuntu.com/ubuntu bionic-security/multiverse amd64 Packages [6,120 B]
Get:21 http://archive.ubuntu.com/ubuntu bionic-security/multiverse Translation-en [2,600 B]
Fetched 4,409 kB in 4s (985 kB/s)
Reading package lists... Done
root@Ubuntu1804x64:~#
```

Рисунок 42 – Обновление данных о версиях и зависимостях в репозитории

Если необходимо уточнить название пакета, который требуется установить – ищем в локальном кэше менеджер по ключевым словам, например, **web server** (рисунок 43):

```
sudo apt-cache search web server
```

Рисунок 43 – Команда для уточнения название пакета

В результате получим большой перечень пакетов, где данное ключевое слово присутствует в описании (рисунок 44).


```

root@Ubuntu1804x64:~# sudo apt-cache search web server
apache2 - Apache HTTP Server
apache2-bin - Apache HTTP Server (modules and other binary files)
apache2-data - Apache HTTP Server (common files)
apache2-dbg - Apache debugging symbols
apache2-dev - Apache HTTP Server (development headers)
apache2-doc - Apache HTTP Server (on-site documentation)
apache2-ssl-dev - Apache HTTP Server (mod_ssl development headers)
apache2-utils - Apache HTTP Server (utility programs for web servers)
apg - Automated Password Generator - Standalone version
awstats - powerful and featureful web server log analyzer
dict-gcide - Comprehensive English Dictionary
git - fast, scalable, distributed revision control system
gpg-wks-server - GNU privacy guard - Web Key Service server
javascript-common - Base support for JavaScript library packages
libapache2-mod-apparmor - changehat AppArmor library as an Apache module
libapache2-mod-perl2 - Integration of perl with the Apache2 web server

```

Рисунок 44 – Поиск по ключевым словам

Выбираем нужный нам пакет – в данном случае это **apache2**, и устанавливаем его командой (рисунок 45).

```
sudo apt-get install apache2
```

Рисунок 45 – Установка apache2

Пакетный менеджер проверяет зависимости, версии, сравнивает их с уже установленными через **apt-get** пакетами, после чего выдает список необходимых для установки компонентов и запрашивает разрешение на продолжение операции (рисунок 46):

```

administrator@ubuntu:~$ vncserver

New 'ubuntu:1 (administrator)' desktop is ubuntu:1

Starting applications specified in /home/administrator/.vnc/xstartup
Log file is /home/administrator/.vnc/ubuntu:1.log

administrator@ubuntu:~$ █

```

Рисунок 46 – Установка пакета

Нажимаем **y** и дожидаемся окончания установки.

Часто используемые команды пакетного менеджера:

- **apt-get update** – обновление информации о пакетах и зависимостях в подключенных репозиториях;
- **apt-get upgrade** – обновление всех установленных пакетов до актуальной версии, имеющейся в репозитории;
- **apt-get install имя_пакета** – установка пакета;
- **apt-get remove имя_пакета** – удаление пакета;
- **apt-get download имя_пакета** – скачать deb-пакет в локальную папку;
- **apt-cache search ключевые слова** – поиск пакета по ключевым словам;
- **apt-cache show имя_пакета** – показать информацию о пакете;
- **apt-cache depends имя_пакета** – показать от каких компонентов зависит данный пакет;
- **apt-cache rdepends имя_пакета** – показать какие компоненты зависят от пакета;
- **apt-mark hold имя_пакета** – зафиксировать текущую версию пакета, прекратить обновление пакета;
- **apt-mark unhold имя_пакета** – разрешить обновление пакета.

Установка через Debian package

Программные пакеты могут распространяться и не через репозитории. Например, они могут быть доступны на сайтах разработчиков. В таком случае, пакет можно скачать и воспользоваться утилитой `dpkg` (`debian package`) для установки. При использовании `dpkg` не происходит автоматическое отслеживание зависимостей и скачивание дополнительных пакетов. В случае нехватки каких-либо программ или библиотек установка завершится ошибкой с указанием списка отсутствующих пакетов. Их придется установить отдельно, например, через `apt-get`, если они присутствуют в подключенных репозиториях, либо скачивать с сайтов разработчиков и устанавливать в нужном порядке через `dpkg`.

Для примера скачаем из системного репозитория установочный пакет веб-сервера Nginx командой (рисунок 47):

```
sudo apt-get download nginx
```

Рисунок 47 – Команда для скачивания из системного репозитория

И попробуем установить через **dpkg -i имя_файла.deb** (рисунок 48):

```
sudo dpkg -i ./nginx_1.14.0-0ubuntu1.6_all.deb
```

Рисунок 48 – Установка через dpkg

В результате система выдает ошибку с перечнем отсутствующих пакетов (рисунок 49):

```
root@Ubuntu1804x64:~# sudo dpkg -i ./nginx_1.14.0-0ubuntu1.6_all.deb
Selecting previously unselected package nginx.
(Reading database ... 103281 files and directories currently installed.)
Preparing to unpack .../nginx_1.14.0-0ubuntu1.6_all.deb ...
Unpacking nginx (1.14.0-0ubuntu1.6) ...
dpkg: dependency problems prevent configuration of nginx:
 nginx depends on nginx-core (< 1.14.0-0ubuntu1.6.1~) | nginx-full (< 1.14.0-0ubuntu1.6.1~) | nginx-light (< 1.14.0-0ubuntu1.6.1~) | nginx-extras (< 1.14.0-0ubuntu1.6.1~); however:
  Package nginx-core is not installed.
  Package nginx-full is not installed.
  Package nginx-light is not installed.
  Package nginx-extras is not installed.
 nginx depends on nginx-core (>= 1.14.0-0ubuntu1.6) | nginx-full (>= 1.14.0-0ubuntu1.6) | nginx-light (>= 1.14.0-0ubuntu1.6) | nginx-extras (>= 1.14.0-0ubuntu1.6); however:
  Package nginx-core is not installed.
  Package nginx-full is not installed.
  Package nginx-light is not installed.
  Package nginx-extras is not installed.
dpkg: error processing package nginx (--install):
 dependency problems - leaving unconfigured
Errors were encountered while processing:
 nginx
root@Ubuntu1804x64:~#
```

Рисунок 49 – Ошибка с перечнем отсутствующих пакетов

Основные опции dpkg:

- **dpkg -i имя_файла.deb** – установка пакета; в качестве параметра указывается полное имя файла;
- **dpkg -r имя_пакета** – удаление ранее установленного пакета;
- **dpkg -l** – вывод списка установленных в системе пакетов.

Установка через файлы сценариев

Установка программы представляет собой распаковку архива, копирование файлов программы в системные директории и, при необходимости, внесение изменений в системные конфигурационные файлы. Все эти действия могут быть внесены в исполняемый файл сценария и выполнены при его запуске. Сейчас достаточно редко встречается подобный метод установки программы, но, тем не менее, некоторые разработчики его используют. Если возникла необходимость в установке такого пакета, необходимо скачать

архив (обычно это файл с расширением .tgz или .tar.gz), распаковать его в отдельную директорию на сервере, изучить распакованные файлы, а также прочитать файл readme, если он имеется. Установка программы запускается, обычно, сценарием **install.sh**, поэтому нужно включить атрибут исполняемого файла командой (рисунок 50):

```
sudo chmod +x ./install.sh
```

Рисунок 50 – Установка сценарием install.sh

После чего запустить непосредственно сценарий (рисунок 51):

```
sudo ./install.sh
```

Рисунок 51 – Запуск сценария

Задание

Установить программное обеспечение:

1. Произвести установку ПО с помощью пакетного менеджера.
2. Произвести установку через Debian package.
3. Произвести установку через файлы сценариев.
4. Все сделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Как можно установить программы?
2. Пакетный менеджер.
3. Команды пакетного менеджера.
4. Как установить программы через package?
5. Как установить программы через файлы сценариев?

Раздел 2. РАБОТА В СЕТЯХ

Лабораторная работа № 1. Настройка IP-адреса и маршрутизации для рабочей машины

Общие сведения

Работа в сети является одной из самых важных не только для пользователя, но и для самой операционной системы. Чтобы все работало правильно нам необходимо настроить следующие параметры:

- IP адрес и маска интерфейса;
- Маршрут или шлюз по умолчанию;
- DNS;
- Имя компьютера.

Определение состояния интерфейса

Чтобы работать в сети каждое устройство должно иметь сетевой адрес – IP адрес. Без него работа компьютера будет невозможна. IP адрес состоит из 4-х чисел, разделенных точками. Выглядит так: 192.168.1.1, 10.10.23.4, 172.16.0.100.

У всех компьютеров IP адрес разный. Вместе с адресом всегда устанавливается и маска. Значение маски одинаково для всех сетевых устройств в определенной локальной сети, включая модемы, принтеры. Выглядит маска так: 255.255.255.0, 255.255.248.0. Значение зависит от того какую маску выберет администратор сети.

Маска определяет количество сетевых устройств в конкретной локальной сети. Например, маска 255.255.255.0 говорит нам, что в сети может быть 254 устройства (и у каждого свой адрес). Для начала проверим настройки сетевого интерфейса, через который и осуществляется связь с внешним миром (интернетом) – `ifconfig`.

Существует и другая команда – `ip address show`. Вывод обеих команд немного отличается. Вторая команда (`ip address show`) выводит информацию обо всех установленных интерфейсах, в то время как первая команда (`ifconfig`) выводит информацию только о "поднятых" (в состоянии UP), то есть работающих в данный момент интерфейсах.

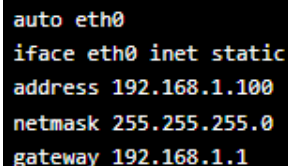
Для получения информации о неработающих сетевых интерфейсах используйте команду `ifconfig` с опцией `-a` (или `-all`). Полный вызов команды выглядит так: `ifconfig -a`.

В выводе команды также обращайте внимание на счетчики RX packets, TX packets. Если они равны 0, то интерфейс ничего не передает и не принимает, хотя и может быть в состоянии UP. К сожалению, IP адрес в выводах команд не указан, что говорит нам о том, что адрес не настроен.

Настройка статического IP-адреса

Для стабильной работы серверов и устройств в локальной сети часто требуется фиксированный адрес. В Linux это достигается через редактирование конфигурационных файлов и использование команд управления интерфейсами. Особенности процедуры зависят от используемой системы управления подключениями.

В Debian 9 при использовании файла `/etc/network/interfaces` пример конфигурации для интерфейса `eth0` выглядит следующим образом (рисунок 52):



```
auto eth0
iface eth0 inet static
address 192.168.1.100
netmask 255.255.255.0
gateway 192.168.1.1
```

Рисунок 52 – Конфигурация для интерфейса `eth0`

После сохранения изменений требуется перезагрузка интерфейса. Для этого используется команда (рисунок 53):



```
ifdown eth0 && ifup eth0
```

Рисунок 53 – Перезагрузка интерфейса

При использовании `systemd-networkd` создается файл, например, `/etc/systemd/network/10-static.network`, с таким содержимым (рисунок 54):

```
[Match]
Name=eth0
[Network]
Address=192.168.1.100/24
Gateway=192.168.1.1
DNS=8.8.8.8
```

Рисунок 54 – Файл static.network

После этого активируется служба (рисунок 55):

```
systemctl restart systemd-networkd
```

Рисунок 55 – Активации службы

Эти действия обеспечивают назначение постоянного адреса для выбранного устройства, что необходимо для стабильной работы сетевых служб.

Управление маршрутизацией в системе

Организация маршрутов позволяет направлять трафик через подходящие шлюзы для достижения различных сетей. В Linux управление маршрутами выполняется с помощью утилит командной строки и конфигурационных файлов. Гибкость и прозрачность инструментов позволяют задавать как статические, так и динамические правила.

Основной инструмент для управления маршрутами – команда **ip route**. Ее использование включает:

- **Просмотр текущих маршрутов:** `ip route show`.
- **Добавление статического маршрута:** `ip route add 192.168.2.0/24 via 192.168.1.1`.
- **Удаление маршрута:** `ip route del 192.168.2.0/24`.

Для постоянного сохранения маршрутов в системах с **/etc/network/interfaces** их можно добавить следующим образом (рисунок 56):

```

auto eth0
iface eth0 inet static
address 192.168.1.100
netmask 255.255.255.0
gateway 192.168.1.1
up ip route add 192.168.2.0/24 via 192.168.1.1
down ip route del 192.168.2.0/24

```

Рисунок 56 – Сохранение маршрутов

Для систем с systemd маршруты настраиваются в файлах **.network**. Пример (рисунок 57):

```

[Match]
Name=eth0
[Network]
Address=192.168.1.100/24
Gateway=192.168.1.1
[Route]
Destination=192.168.2.0/24
Gateway=192.168.1.1

```

Рисунок 57 – Настройка маршрутов для систем с systemd

Изменения вступают в силу после перезапуска соответствующих служб. Грамотное управление маршрутами позволяет оптимизировать трафик и избегать конфликтов в сложных топологиях.

Задание

Настроить IP-адрес и маршрутизацию для рабочей машины:

1. Произвести настройку IP-адреса на рабочей машине.
2. Проверить с помощью команды `ping`.
3. Настроить маршруты с помощью команды `ip route`.
4. Все сделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Что такое IP адрес?
2. Что такое маска интерфейса?
3. Что такое Маршрут и Шлюз?
4. Что такое DNS?
5. Как настроить IP адрес?

Лабораторная работа № 2. Работа с маршрутизацией

Утилита для работы с маршрутизацией

Рассмотрим утилиту IP, с помощью которой можно назначить и просмотреть адреса сетевых устройств, а также маршруты.

Некоторые из них:

- `address` – позволяет назначать и удалять ip адреса, просматривать их и тому подобное.
- `link` – можно включить или выключить сетевой интерфейс, посмотреть список интерфейсов и их mac адреса.
- `neigh` – можно добавить или удалить mac адрес из arp таблицы, или полностью ее очистить.
- `route` – позволяет создавать новые маршруты и удалять их, а также просматривать уже созданные маршруты.

Сетевые маршруты бывают временные, которые действуют до перезагрузки сетевой службы, либо системы и постоянные.

Посмотреть маршруты

Вывести список всех имеющихся маршрутов, командой – `ip route` (рисунок 58).

```
ip route
```

```
default via 192.168.1.254 dev eth0  
192.168.1.0/24 dev eth0 proto kernel scope link src 192.168.1.2  
192.168.7.0/24 dev eth1 proto kernel scope link src 193.268.7.2
```

Рисунок 58 – Список всех имеющихся маршрутов

Добавить временный статический маршрут

Добавление статического маршрута в сеть 192.168.7.0/24 через шлюз 192.168.1.1, командой – `ip route add` (рисунок 59).

```
ip route add 192.168.7.0/24 via 192.168.1.1
```

Рисунок 59 – Добавление статического маршрута

Посмотреть прохождение маршрута, можно командой – `ip route get` (рисунок 60).

```
ip route get 192.168.7.2

192.168.7.2 via 192.168.1.1 dev eth0 src 192.168.1.2
cache ipid 0x9bbc mtu 1500 advmss 1460 hoplimit 64
```

Рисунок 60 – Просмотр прохождения маршрута

Добавить постоянный статический маршрут

Постоянные статические маршруты добавляются в файл конфигурации сети `/etc/network/interfaces`, в описание необходимого интерфейса (рисунок 61).

```
post-up ip route add default via 172.16.100.1
pre-down ip route del default via 172.16.100.1
```

Рисунок 61 – Добавление статических маршрутов

- `post-up` – означает запустить команду после поднятия интерфейса.
- `pre-down` – означает запустить команду перед отключением интерфейса.

Пример настройки на `ens192` (рисунок 62).

```
auto eth0
iface eth0 inet static
address 192.168.1.2
network 255.255.255.0
gateway 192.168.1.1
post-up ip route add default via 172.16.100.1
```

Рисунок 62 – Настройка на `ens192`

Данные параметры будут применяться после следующей инициализации сетевой карты. Перезагружаем систему либо отключаем\включаем сетевой интерфейс (рисунок 63).

```
ifdown eth0  
ifup eth0
```

Рисунок 63 – Включение/отключение сетевого интерфейса

Изменить статический маршрут

Изменить разово статический маршрут, можно командами `ip route replace` / `ip route change` (рисунок 64-65).

```
ip route replace default via 192.168.1.1 dev ens192  
ip route replace 192.168.1.0/24 via 192.168.1.1
```

Рисунок 64 – Команда `replace`
для изменения статистического маршрута

```
ip route change default via 192.168.1.1 dev ens192  
ip route change 192.168.1.0/24 via 192.168.1.1
```

Рисунок 65 – Команда `change`
для изменения статистического маршрута

Для изменения постоянного маршрута, необходимо вносить изменения в конфигурационный файл `/etc/network/interfaces`, для нужного сетевого интерфейса.

Удалить статический маршрут

Удалить маршрут, командой – `ip route del` (рисунок 66).

```
ip route del 192.168.1.0/24
```

Рисунок 66 – Команда `del`

Задание

С помощью утилиты IP необходимо:

1. Посмотреть маршруты.
 2. Добавить временный статический маршрут.
 3. Добавить постоянный статический маршрут.
 4. Изменить статический маршрут.
 5. Удалить статический маршрут.
- Все проделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Для чего необходима утилита IP?
2. Что такое сетевой маршрут?
3. Какие бывают сетевые маршруты?
4. Как вывести список все маршрутов?
5. Как добавить статистический маршрут?
6. Как добавить постоянный статистический маршрут?
7. Как изменить статистический маршрут?
8. Как удалить статистический маршрут?

Лабораторная работа № 3. Настройка DNS на рабочей машине

DNS-серверы

DNS-серверы используются для преобразования доменных имен в IP-адреса. При использовании статической конфигурации они указываются прямо в файле `/etc/network/interfaces` (рисунок 67):

```
dns-nameservers 8.8.8.8 8.8.4.4
```

Рисунок 67 – Указание DNS серверов

Если же вы используете DHCP, то адреса DNS-серверов будут назначены автоматически. Однако, если вы хотите задать свои собственные DNS-серверы, даже при использовании DHCP, вы можете сделать это, изменив файл `/etc/resolv.conf`.

Откройте файл `/etc/resolv.conf` (рисунок 68):

```
sudo nano /etc/resolv.conf
```

Рисунок 68 – Открытие файла `resolv.conf`

Добавьте строки (рисунок 69):

```
nameserver 8.8.8.8  
nameserver 8.8.4.4
```

Рисунок 69 – Добавление `nameserver`

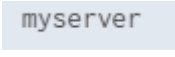
Изменить `hostname` (имя хоста)

Имя хоста используется для идентификации вашего сервера в сети. Чтобы изменить имя хоста, откройте файл `/etc/hostname` (рисунок 70):

```
sudo nano /etc/hostname
```

Рисунок 70 – Открытие `hostname`

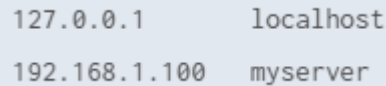
Замените текущее значение на новое имя хоста, например (рисунок 71):



```
myserver
```

Рисунок 71 – Новое имя хоста


Затем обновите файл `/etc/hosts`, чтобы добавить соответствие между новым именем хоста и вашим IP-адресом (рисунок 72):



```
127.0.0.1    localhost
192.168.1.100 myservers
```

Рисунок 72 – Обновление файла `/etc/hosts`

Перезагрузите систему или выполните команду (рисунок 73):



```
sudo hostnamectl set-hostname myservers
```

Рисунок 73 – Выполнение команды для изменения имени хоста

Задание

Произвести настройку DNS на рабочей машине:

1. Настроить DNS на рабочей машине.
2. Изменить `hostname`.
3. Все проделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Для чего используются DNS-серверы?
2. Разница между DNS и DHCP.
3. Для чего используется имя хоста?
4. В каком файле можно изменить настройки DNS?

Лабораторная работа № 4. Управление сетями рабочей машины

Использование инструментов диагностики сети

Диагностика сети в Linux включает использование различных утилит и команд для проверки состояния подключения, обнаружения и устранения проблем. Эти инструменты предоставляют детализированную информацию о сетевых интерфейсах, маршрутах, пакетах и ошибках, что важно для администрирования и обеспечения бесперебойной работы.

Одна из основных утилит для диагностики – **ping**. Она отправляет ICMP-пакеты на удаленный узел и замеряет время отклика. Пример команды (рисунок 74):

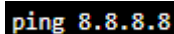


Рисунок 74 – Утилита ping

Для проверки состояния маршрутов используется команда **traceroute**, которая отображает пути пакетов до конечного узла. Пример (рисунок 75):

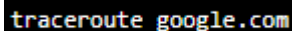


Рисунок 75 – Команда traceroute

Утилита командной строки **netstat** предоставляет информацию о сетевых соединениях, активных портах и интерфейсах (таблица 1).

Таблица 1

Параметры команды netstat

Команда	Описание
netstat -r	отображает маршруты в таблице
netstat -i	показывает состояние сетевых интерфейсов
netstat -s	отображает статистику по типам пакетов и ошибкам

Для более детализированного анализа сетевого трафика используется утилита командной строки **tcpdump**. Она позволяет захватывать и анализировать пакеты на уровне сетевого интерфейса (рисунок 76):

```
sudo tcpdump -i eth0
```

Рисунок 76 – Утилита tcpdump

Эти инструменты помогают точно диагностировать проблемы, начиная от соединений и заканчивая работой протоколов, и являются неотъемлемой частью системного администрирования в Linux.

Задание

Произвести диагностику сети на наличие проблем:

1. С помощью утилит и команды произвести диагностику сети.
2. Сравнить и выяснить в чем отличие и какой из методов лучше.
3. Все сделанные действия зафиксировать в отчет.

Контрольные вопросы

1. Какие утилиты существуют для диагностики сети?
2. Какие команды существуют для диагностики сети?
3. Какая команда используется для проверки состояния маршрутов?
4. Параметры команды netstat.
5. Команда для детализированного анализа сетевого трафика.

СОДЕРЖАНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Цель самостоятельной работы обучающихся – получить новые знания по дисциплине «Основы системного администрирования».

Самостоятельная работа необходима для развития у обучающихся способности самостоятельно решать задачи профессиональной деятельности, умения и навыков планирования времени, а также стремления к развитию и совершенствованию.

Виды самостоятельной работы обучающихся указаны в таблице 2.

Таблица 2

Виды самостоятельной работы

№ п/п	Вид СРС
1	Административные средства конкретных дистрибутивов.
2	Работа со сценариями запуска системы.
3	Общий пользователь.
4	Монтирование и размонтирование файловой системы.
5	Коммерческие системы резервного копирования.
6	Поиск полезной информации в журнальных файлах.
7	Средства управления конфигурацией.
8	Сетевое конфигурирование в различных дистрибутивах.
9	Маршрутизаторы Cisco.
10	Особенности пакета BIND в различных дистрибутивах.
11	Просмотр статистических данных функционирования различных сетевых протоколов.

УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

Основная литература

1. Перлова, О. Н. Соадминистрирование баз данных и серверов : учебник для студентов учреждений среднего профессионального образования / О. Н. Перлова, О. П. Ляпина ; О. Н. Перлова, О. П. Ляпина. – 3-е изд., испр. – Москва : Академия, 2023. – 304 с. с. – URL: <https://academia-moscow.ru/reader/?id=616608#copy> (дата обращения: 30.03.2026).

Дополнительная литература

1. Мошков, М. Е. Введение в системное администрирование Unix : учебное пособие / М. Е. Мошков. – 4-е изд. – Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2025. – 207 с. – ISBN 978-5-4497-0906-6 // Электронный ресурс цифровой образовательной среды СПО PROФобразование : [сайт]. – URL: <https://profspo.ru/books/146338> (дата обращения: 30.03.2026). – Режим доступа: для авторизир. пользователей
2. Гончарук, С. В. Администрирование ОС Linux : учебное пособие / С. В. Гончарук. – 4-е изд. – Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2024. – 163 с. – ISBN 978-5-4497-2432-8 // Электронный ресурс цифровой образовательной среды СПО PROФобразование : [сайт]. – URL: <https://profspo.ru/books/133916> (дата обращения: 30.03.2026). – Режим доступа: для авторизир. пользователей
3. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. – Саратов : Профобразование, 2024. – 108 с. – ISBN 978-5-4488-1864-6 // Электронный ресурс цифровой образовательной среды СПО PROФобразование : [сайт]. – URL: <https://profspo.ru/books/139041> (дата обращения: 30.03.2026). – Режим доступа: для авторизир. пользователей.

Программное обеспечение и интернет-ресурсы

- Университетская библиотека онлайн. Режим доступа: www.biblioclub.ru
- Электронно-библиотечная система Лань. Режим доступа: <http://e.lanbook.com>
- Электронно-библиотечная система Znanium.com. Режим доступа <https://znanium.ru/>
- Электронная библиотека образовательной платформы Юрайт <https://urait.ru/>
- Электронный ресурс цифровой образовательной среды СПО PROОбразование. Режим доступа: <https://profspo.ru>

Содержание

Раздел 1. Основы администрирования.	3
Лабораторная работа № 1. Установка виртуальной машины.	3
Лабораторная работа № 2. Установка дополнений.	18
Лабораторная работа № 3. Изменение общесистемных настроек.	20
Лабораторная работа № 4. Управление пользователями и правами.	28
Лабораторная работа № 5. Файловая система Linux.	34
Лабораторная работа № 6. Резервное копирование данных.	43
Лабораторная работа № 7. Работа с Syslog в Linux и обработка журнальных файлов.	49
Лабораторная работа № 8. Установка программного обеспечения в Linux.	63
Раздел 2. Работа в сетях.	69
Лабораторная работа № 1. Настройка IP-адреса и маршрутизации для рабочей машины.	69
Лабораторная работа № 2. Работа с маршрутизацией.	73
Лабораторная работа № 3. Настройка DNS на рабочей машине. .	77
Лабораторная работа № 4. Управление сетями рабочей машины.	79
СОДЕРЖАНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ	81
УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ	82