Министерство науки и высшего образования Российской Федерации

Федерального государственного бюджетного образовательного учреждения высшего образования

"Сочинский государственный университет"

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Проректор  по учебной работе  и КОД  \_\_\_\_\_\_\_\_\_\_\_\_ А.В. Иваненко  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

Регистрационный номер \_\_\_

Факультет информационных технологий и математики

Кафедра информационных технологий

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**по дисциплине**

**«Администрирование корпоративных информационных систем»**

**Лабораторный практикум**

для направлений (уровень бакалавриата):

09.03.03 «Прикладная информатика», профиль «Прикладная информатика в экономике»

**Автор:**

Драч В.Е.,

доцент, к.т.н., доцент кафедры ИТиМ

[vladimir@drach.](mailto:drach@bmstu.ru)pro

г. Сочи, 2023 г.

Автор:

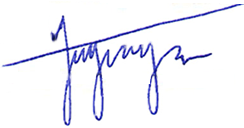


Драч Е.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Рецензент:

И.о. зав.каф. кафедры ИУК2 «Компьютерные системы и сети»

МГТУ им. Баумана



к.т.н., доцент

Родионов А.В.

Утверждена на заседании кафедры «Информационные технологии»

Протокол №01 от «\_\_30\_\_» августа 2023 г.

Заведующий кафедрой

А.С.Копырин \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Декан факультета

Волков А.Н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Оглавление

[Лабораторная работа №1. Установка OC 4](#_Toc182661234)

[Лабораторная работа №2. Протокол SSH 23](#_Toc182661235)

[Лабораторная работа №3 Первоначальная настройка 33](#_Toc182661236)

[Лабораторная работа №4 Доступ по ключу из Linux 45](#_Toc182661237)

[Лабораторная работа №5 Доступ по ключу из Windows 50](#_Toc182661238)

[Лабораторная работа №6 Настройка web-сервера apache 50](#_Toc182661239)

[Лабораторная работа №7. Стек LEMP 56](#_Toc182661240)

[Лабораторная работа №8. Установка CMS 63](#_Toc182661241)

[Лабораторная работа №9. Установка phpMyAdmin 69](#_Toc182661242)

[Лабораторная работа №10. Установка и настройка Fail2ban 74](#_Toc182661243)

[Лабораторная работа №11. Сторожевой бот для Telegram 79](#_Toc182661244)

[Лабораторная работа №12. Бот для мониторинга 79](#_Toc182661245)

[Лабораторная работа №13\*. Автозапуск скрипта в контейнере 79](#_Toc182661246)

[Лабораторная работа №14. Автозапуск скрипта как службы 82](#_Toc182661247)

[Лабораторная работа №15. Система мониторинга в контейнере. 83](#_Toc182661248)

[Лабораторная работа №X\*. Нагрузочное тестирование веб-сервера 84](#_Toc182661249)

Что не вошло (по техническим причинам) в наш прекрасный курс:

- Детальная настройка firewalld / iptables / nftables

- Написание bash script

- Оркестрирование Ansible

- Почтовый сервер

# Лабораторная работа №1. Установка OC

Цель работы: формирование практических навыков администрирования корпоративных операционных систем.

**Предварительные замечания**

* Дистрибутив Rocky Linux основан на коммерческом Red Hat Enterprise Linux, применяется формат пакетов RPM (Red Hat Package Manager).
* Самой свежей версией на данный момент является Rocky Linux 9.0, выпущенная 01 июля 2022 года.
* В стенах СГУ будем пользоваться версией 8.6. т.к. только она запустится на виртуальной машине.
* Предполагается, что перед началом установки OC на вашем ПК уже есть VirtualBox.
* ОС предназначена для компьютеров с архитектурой х86, ARM и других.
* Потребуется соединение с интернетом, поскольку часть файлов в процессе установки будет загружаться из Сети.

**Получить Rocky Linux**

Загрузочный образ с ОС можно получить несколькими путями, причем все они официальные и бесплатные.

1) Воспользоваться имеющимся образом, скопировав у коллеги или преподавателя.

2) Скачать по прямой ссылке с официального сайта. Скорость будет **ограничена**.

https://rockylinux.org/download

Выбираем версию 8, x86\_64, Boot - стартует загрузка образа операционной системы в формате ISO.

3) Скачать через торрент. В зависимости от особенностей вашего интернет-соединения, прямая загрузка может быть недоступна (например, в Крыму), в этом случае вам останется только воспользоваться торрент-клиентом. Скорость будет определяться скоростью Вашего подключения.

**Что нужно для установки системы**

* ПК с 64-битным процессором. 32-битные процессоры давно не поддерживаются.
* Не менее 1 Gb ОЗУ. Рекомендуется 2 Gb для академических задач.
* Не менее 10 Gb на жестком диске (лучше 12 Gb для академических задач).

**Как создать виртуальную машину**

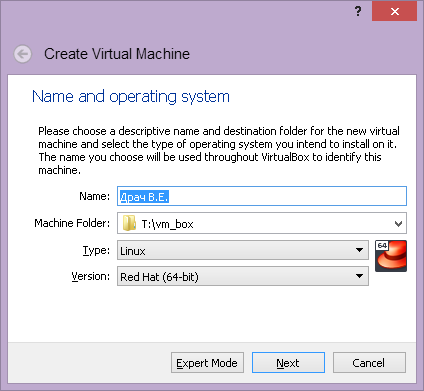
Откройте VirtualBox и нажмите расположенную на панели под строкой меню кнопку «Создать».



Создавая виртуальную машину и выделяя ей аппаратные ресурсы, используйте кнопку «Далее» для перехода к следующему экрану, «Назад» — для возврата к предыдущему.

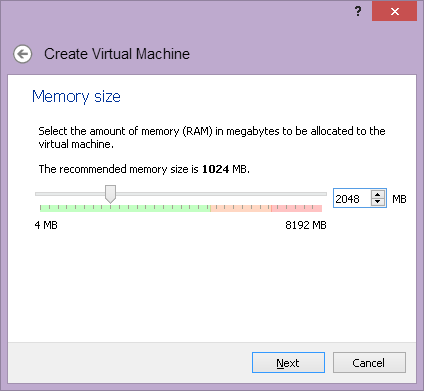


В поле «Имя» введите **уникальное** название нового «виртуального компьютера». Необходимо выбрать «Тип»: Linux; «Версия»: **Red Hat** -- по имени фирмы-разработчика.

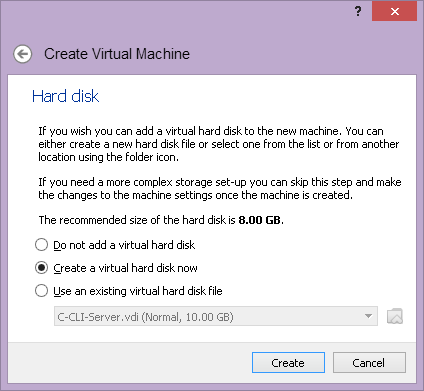


**Выделение ОЗУ и ЖД для Rocky Linux**

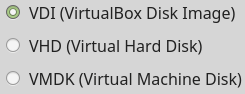
Установите объем оперативной памяти ~2048 Mb. Это сделает работу с ОС более комфортной. Однако, чем больше памяти мы выделим на этом шаге, тем меньше виртуальных машин можно будет параллельно запускать в будущем. Академические задачи хорошо решаются с ОЗУ **1600 Мb**. Всегда прикидывайте выделение ОЗУ таким образом, чтобы была возможность запустить 2 витруальные машины одновременно.



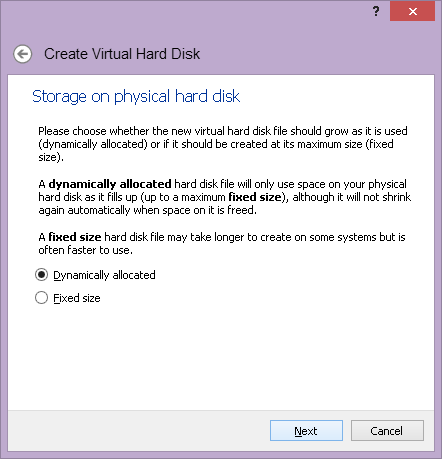
По умолчанию предусмотрена опция «Создать новый виртуальный жесткий диск». Не меняйте ее. Нажмите «Создать».



Если вы не планируете использовать создаваемый «диск» с другими инструментами виртуализации, оставьте тип накопителя «VDI».

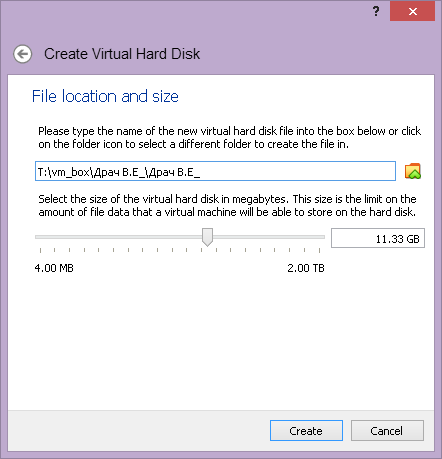


Если вы укажете в качестве формата хранения «Динамический виртуальный жесткий диск», пространство на накопителе будет использоваться по мере необходимости. Если вы выберете «Фиксированный виртуальный жесткий диск», указанный вами объем предоставляется виртуальной машине сразу и станет **недоступен** вам для других задач. Фиксированный жесткий диск обычно быстрее, поэтому **при наличии свободного пространства** на HDD или SSD отдайте предпочтение этому варианту.



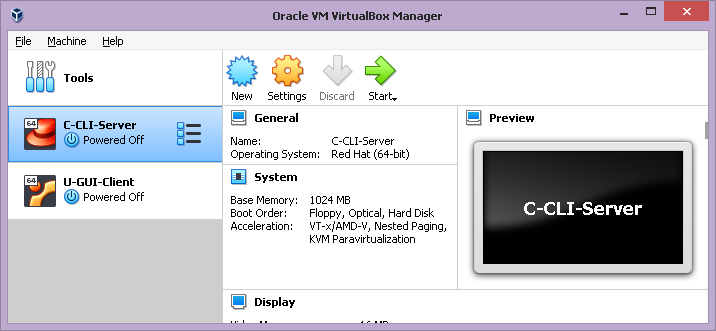
Укажите имя и размер виртуального жесткого диска. Для академических задач должно хватить 10 Gb, можно выделить 20 Gb или больше по своему усмотрению. Чем больше диск, тем сложнее его будет потом переносить на флэшку или другую физическую машину. Нажмите «Создать».

* Dinamically allocated - размер файла, в котором хранится образ диска будет меняться в зависимости от заполненности диска. Это чуть медленнее, зато удобно переносить виртуальную машину. **Особенно хорош этот вариант, если мы собираемся хранить файл виртуального диска на собственной флешке**!
* Fixed Size - размер файла будет строго соответствовать объему виртуального ЖД при любой его заполненности. Работает немножко быстрее.



**Настройка виртуальной машины**

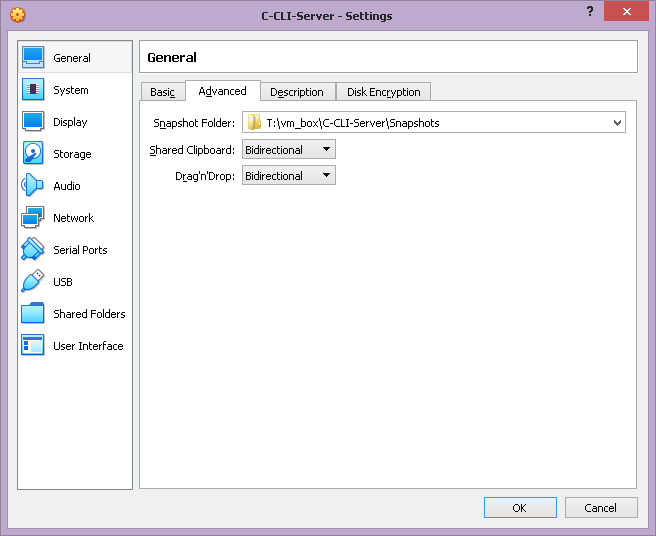
Новый «виртуальный ПК» отобразится в списке после ранее созданных.



Выделите ее и щелкните по кнопке «Настроить», расположенной на панели под строкой меню.

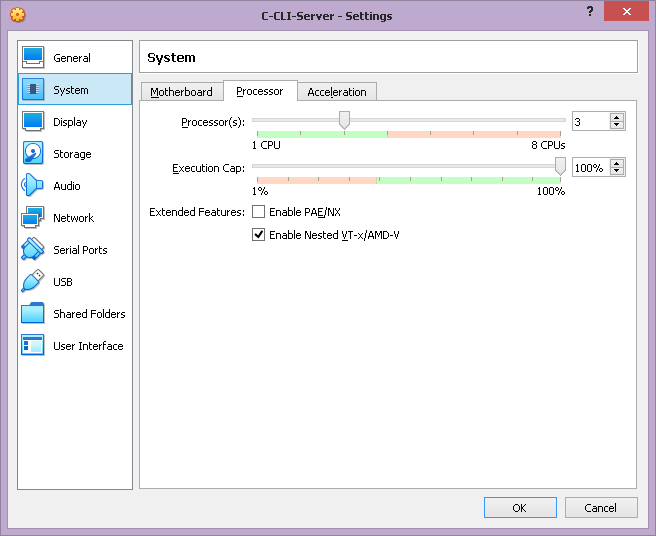


В разделе «Общие» на вкладке «Дополнительно» в полях «Общий буфер обмена» и «Функция Drag’n’Drop» выберите из списков «Двунаправленный». Это необходимо для обмена данными между основной ОС и гостевой Rocky Linux.

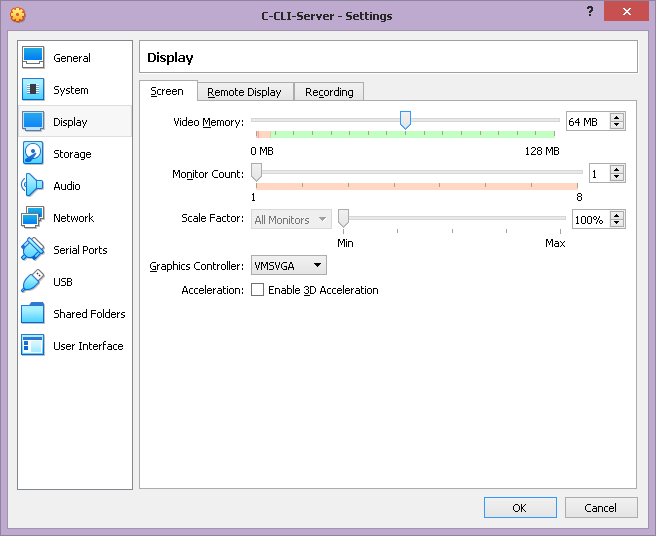


В разделе «Система» на вкладке «Материнская плата» определите порядок загрузки. Приоритетный: «Оптический накопитель (CD/DVD)»; второй: «Жесткий диск». Снимите «галочки» возле других накопителей. Для изменения порядка загрузки используйте стрелки «Вверх» и «Вниз».

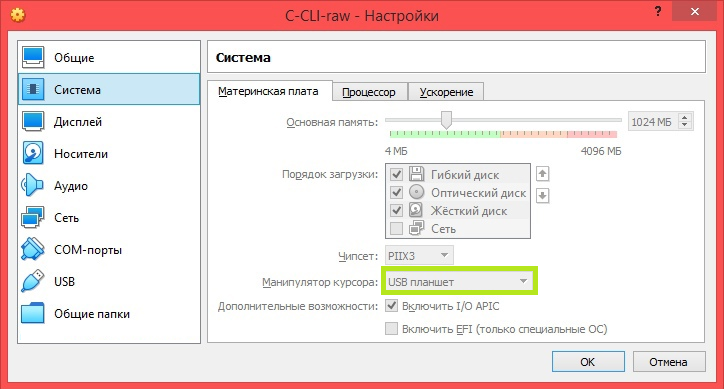
На вкладке «Процессор» определите число доступных виртуальной машине ядер ЦП. Для академических задач (ввиду отсутствия постоянной высокой нагрузки) рекомендуется выделять количество ядер -- 2.



В разделе «Дисплей» на вкладке «Экран» предоставьте виртуальной машине видеопамять. Чем больше объем, тем выше качество изображения и удобнее работа с операционной системой. Тут вообще не экономим! Нажмите «ОК».



Проверьте, что на вкладке "Система" выбран тип манипулятора курсора **USB-планшет** (**USB Tablet**). Иначе не будет отображаться курсор мыши в окне виртуальной машины.



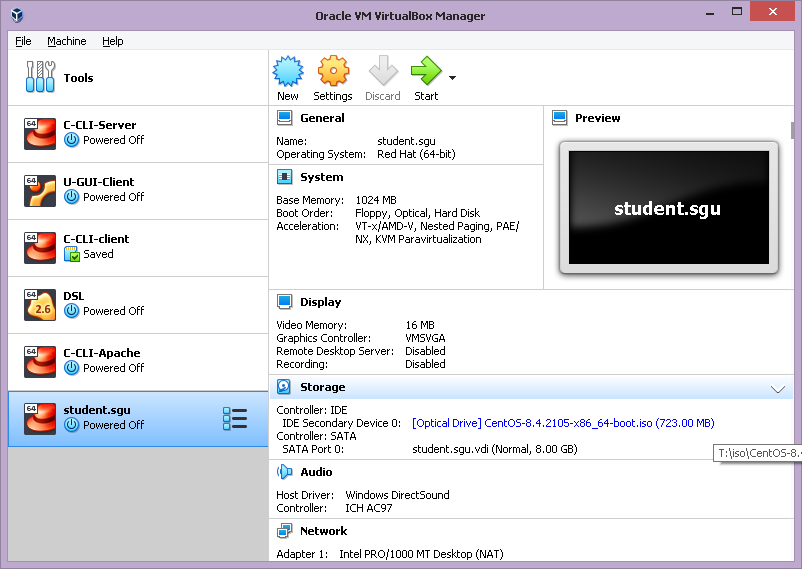
На вкладке Сеть предоставьте виртуальной машине 2 адаптера:

* первый в режиме **NAT** - для самого простого доступа в Интернет
* второй в режиме **Внутренняя сеть** - для взаимодействия виртуальных машин между собой

*Начиная с 2022 года можно выбирать один адаптер в режиме* ***Сеть NAT****: так теоретически проще, но тогда решайте все проблемы самостоятельно. Как показала практика, проблемы не решаются.*

В общих настройках укажите путь к ранее загруженному файлу ISO-образа Rocky Linux. Таким образом мы моделируем ситуацию, что в CD-привод виртуальной машины вставлен загрузочный компакт-диск.

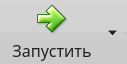
Пожалуйста **внимательно посмотрите** на скриншот ниже: найдите синюю ссылку Optical Drive, рядом с горящей подсказкой справа внизу.



**Подготовка к установке ОС на виртуальную машину**

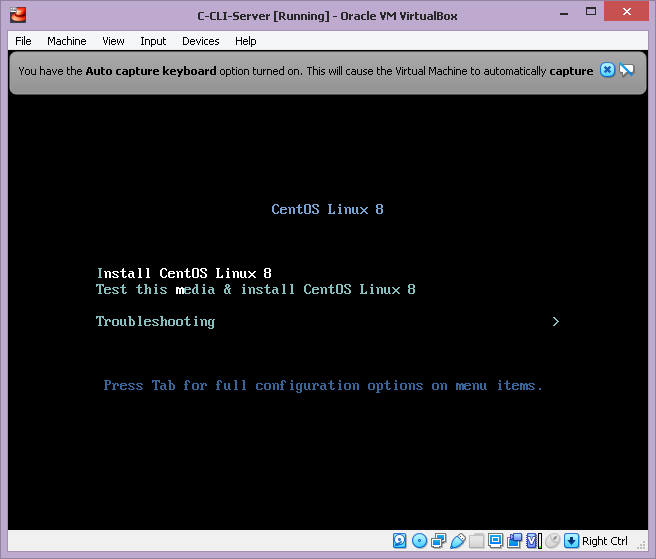
Выделите виртуальную машину. Предусмотрены два способа запустить ее:

* Двойным щелчком мышью по ней
* Нажатие кнопки «Запустить» на панели под строкой меню

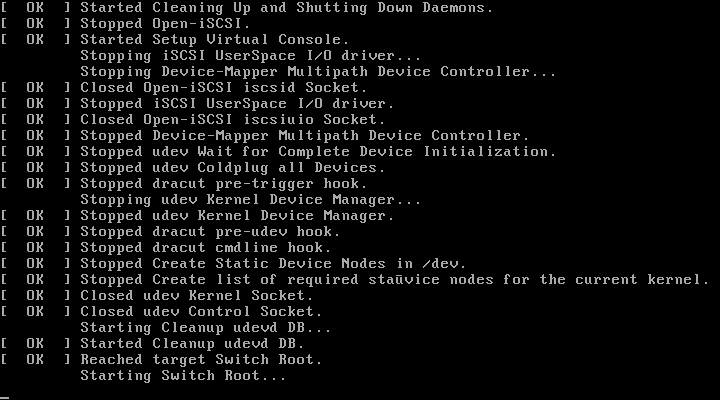


Машина должна стартовать, и мы увидим её дисплей.

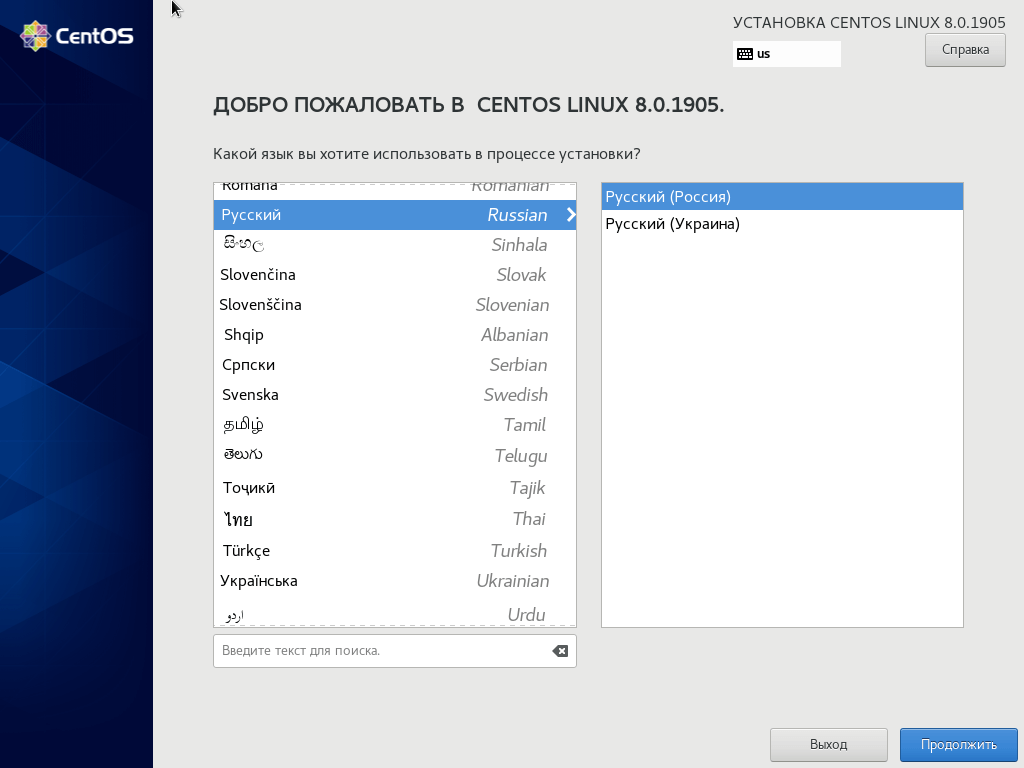
Нажмите кнопку «Продолжить» в нижней части диалогового окна. Выберите пункт «Install Rocky Linux Linux 8». Тестировать скачанный образ на целостность не обязательно.



Подождите, пока дистрибутив готовится к установке.

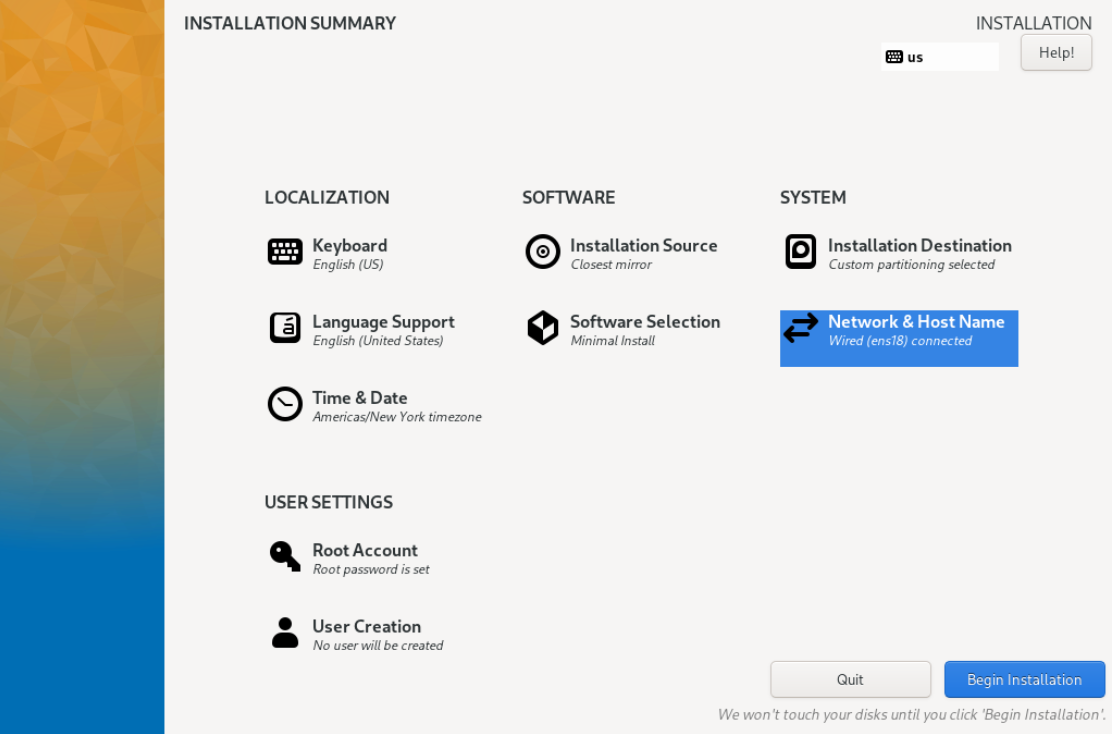


Выберите язык мастера установки ОС. Нажмите «Продолжить». Для сервера **выбирать русский язык смысла нет**, но если психологически проще с русским – попробуйте выбрать его.



Актуальная раскладка клавиатуры отображается в верхнем правом углу окна установщика Linux. Для переключения раскладки один раз щелкните по ней левой кнопкой мыши.

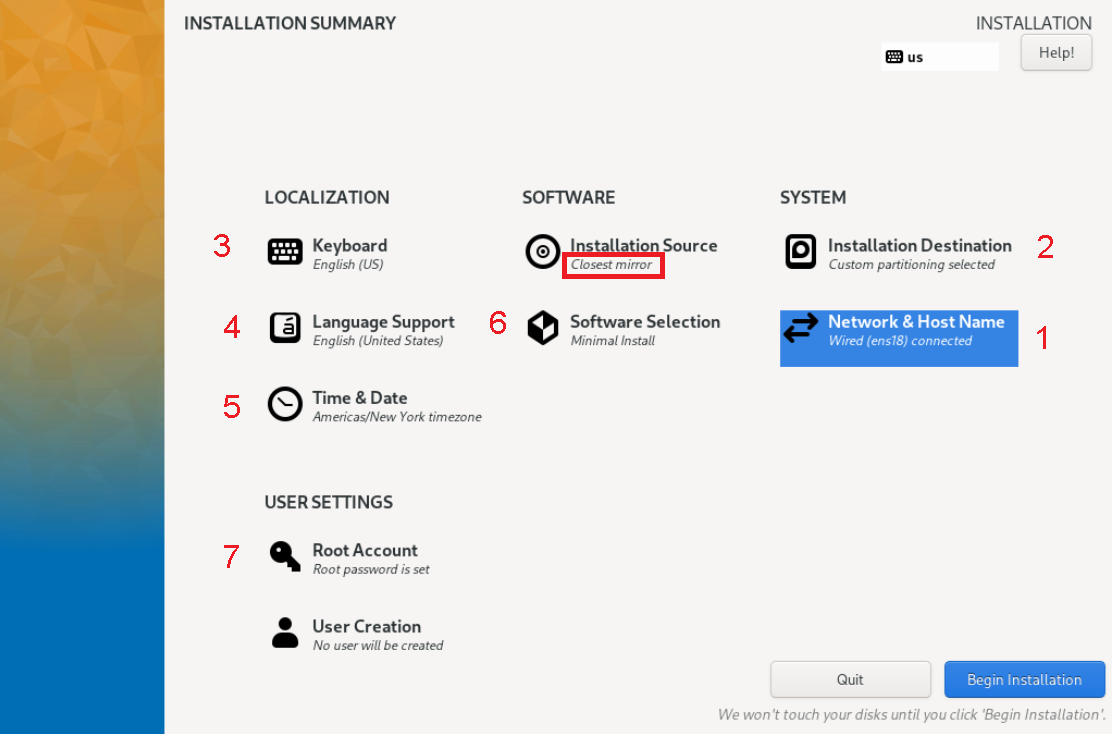
Далее мы попадаем на интересный экран выбора параметров установки (см. ниже).



Важно отметить, что в отличие от Windows (где установка линейная, шаг за шагом) во многих дистрибутивах Linux можно в произвольном порядке менять параметры установки на данном экране. В идеальном случае надо будет щёлкнуть по каждому из 9 пунктов на этом экране и вдумчиво задать все параметры.

Некоторые секции окна «Обзор установки» отмечены оранжевым треугольником с восклицательным знаком - заполните их обязательно.

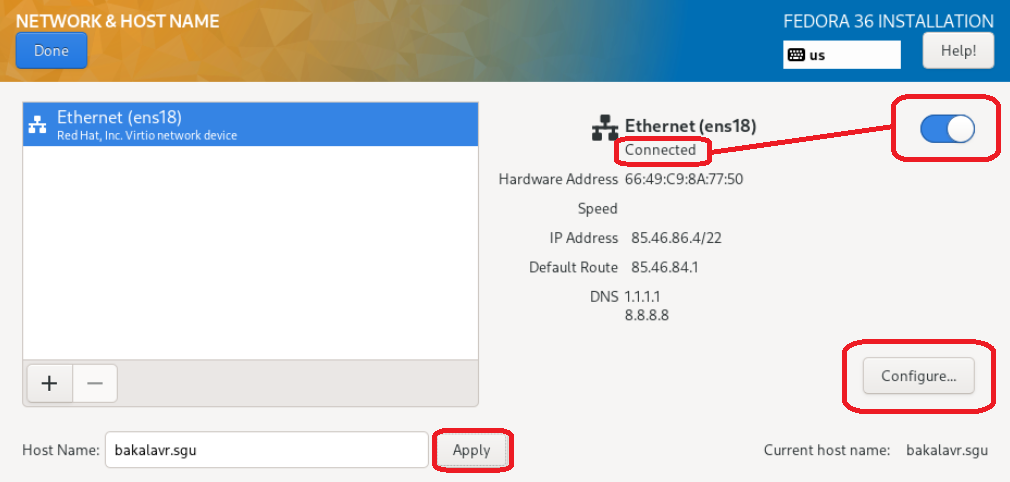
Оптимальный порядок показан на рисунке ниже.



**1) Network - Сетевое соединение**

Измените имя машины на произвольное.

По умолчанию соединение с интернетом **отключено**.



В секции «Сеть и имя хоста» переведите ползунок активации сетевого соединения во включенное состояние.



Убедитесь, что статус изменился на Connected. Если этого не произошло, переходим к конфигурации Configure.

Внимание! В идеальном случае, в настройках сети необходимо включить обе сетевые карты. Можно пока включить только первую, а потом на установленной ОС запустить nmtui. Но это путь с непредсказуемым результатом, в дальнейшем создаст массу проблем.

**Первая сетевая карта** (в режиме NAT) должна будет самостоятельно получить IP-адрес по протоколу DHCP, чтобы выходить в интернет. Для этого во вкладке IPv4 выбираем DHCP, в строке DNS указываем пару-тройки надёжных серверов через пробел. DNS можно указать по выбору: 1.1.1.1 (CloudFlare), 8.8.8.8 (Google), 77.88.8.8 (Yandex)... Протокол IPv6 лучше всего отключить.



**Вторая сетевая карта** в режиме (внутренняя сеть) будет работать с жёстко заданными настройками, поэтому указываем вручную на вкладке IPv4:



* IP-адрес IPv4 - 10.4.0.1
* Маска - 255.255.255.0
* Шлюз - 10.4.0.2

Шестую версию протокола IPv6 **отключаем**.

Если всё сделано правильно, то в разделе Installation Source должно появиться «Closest Mirror», в этом случае в него даже нет необходимости заходить. Это означает, что установщик сам готов найти ближайшее зеркало для скачивания пакетов.

Нет смысла выполнять следующие шаги, если не удалось дождаться надписи «Closest Mirror».

**2) Installation Destination - Разметка диска**

Предлагается разбить диск на разделы. Возможны девиации:

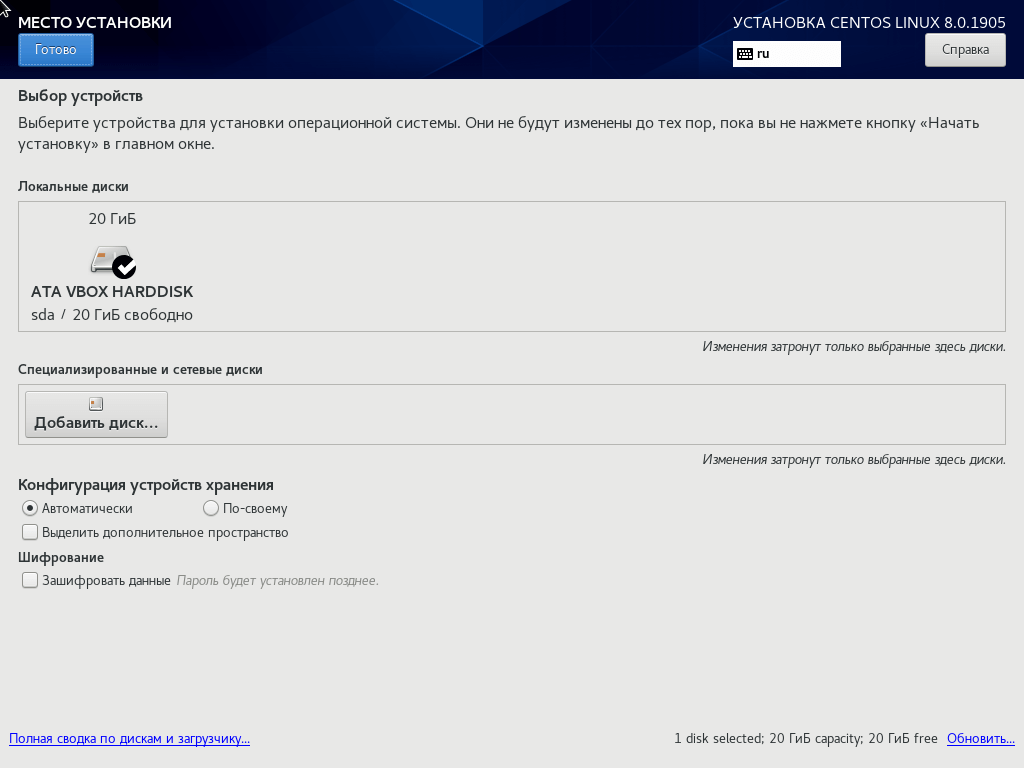
1) Оставить предложенный установщиком вариант «Автоматически». Нажмите кнопку «Готово» в верхнем левом углу окна. Это вариант для ленивых, которые любят решать проблемы в будущем.

2) Разметить диск вручную (указать "По-своему" и нажать "Дальше"). В современных реалиях для прикладных задач наиболее рациональным считается создание и монтирование двух разделов:

* /boot -- примерно 1 Gb
* / (корневой раздел) -- всё остальное место.

Раздел подкачки swap лучше всего оформить отдельным файлом после установки: не задаём его.

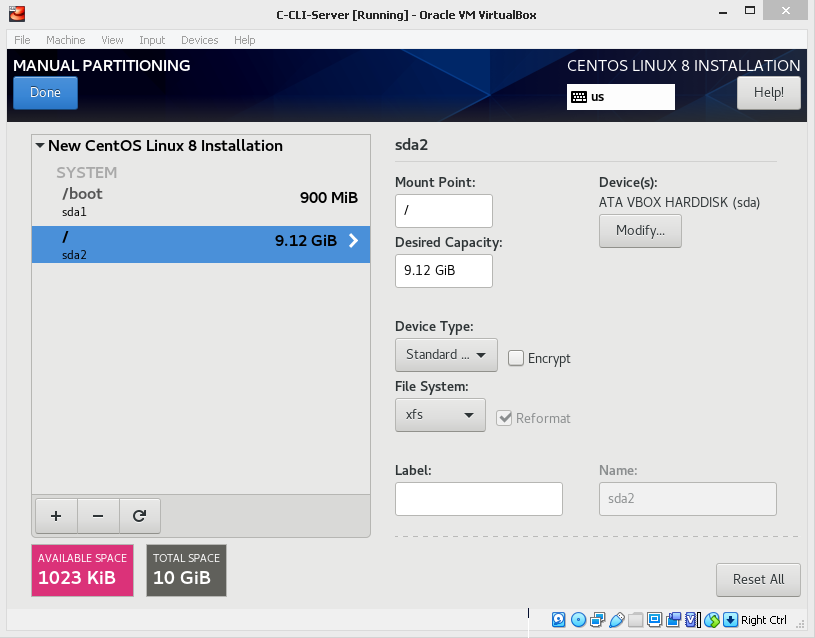
Если речь идёт про сервер, то нет никакого смысла выделять /home в отдельный раздел (не верьте тому, что пишут в интернетах).



Если мы решились разметить диск вручную, рекомендуемые параметры разделов будут следующие:

* /boot – примерно 1 Gb, файловая система **ext2**...ext4
* / – всё оставшееся место, файловая система **XFS** или **ext4**

Вкратце поясним: на боевых серверах для загрузочного раздела принято выбирать старые и простые файловые системы, а для основного раздела – модные и современные (с журналированием и поддержкой гигантских файлов).



**3) Клавиатура - Keyboard**

Выберите наиболее привычный вариант.

**4) Language**

Выберите наиболее привычный вариант. Однако, следует отметить, что выбор русского языка на этапе установки приведёт к тому, что в будущем виртуальная машина даже сообщения об ошибках будет писать на русском языке; в таком случае очень тяжело будеть пользоваться поиском решений в интернете.

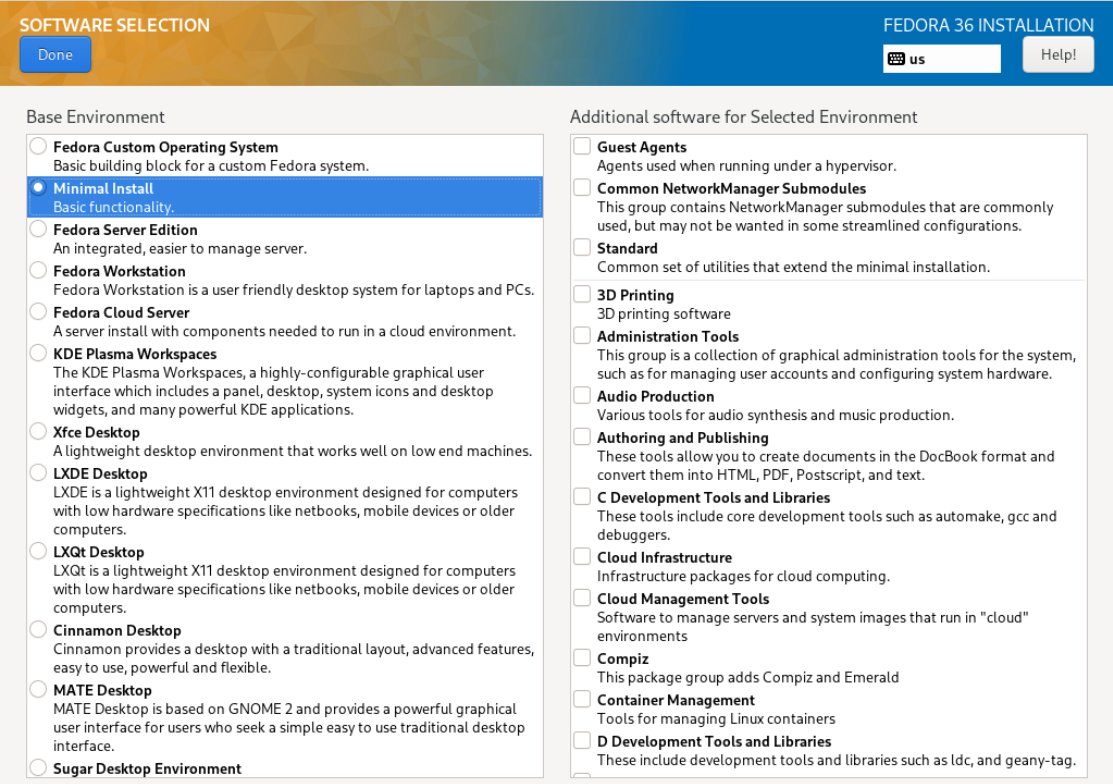
**5) Часовой пояс**

Перейдите в секцию «Time & Date» и выберите свой часовой пояс.



**6) Software Selection**

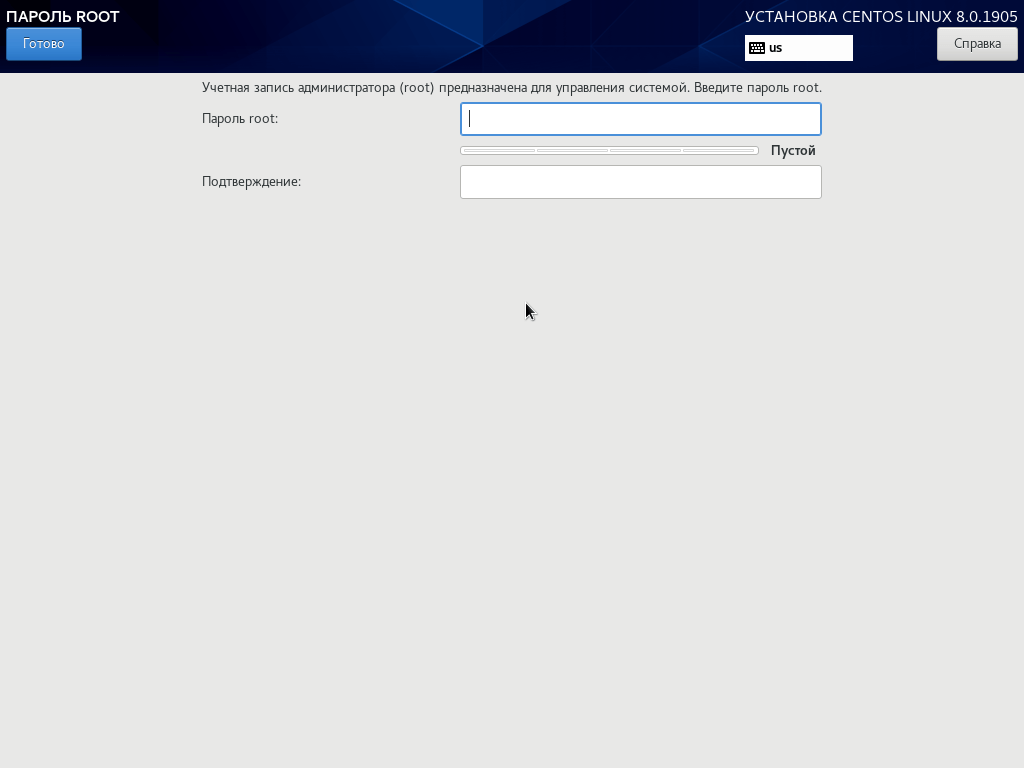
Выбирайте Minimal Install, всё остальное потом вручную добавим.



Ещё раз: в пункте **"Выбор программ"** лучше всего указать **минимальную** установку. В правой части выбора ПО лучше ничего не отмечать. Это позволяет получить сервер с минимальным функционалом. В будущем доустановим все необходимые пакеты по мере необходимости.

**7) Пароль**

Обязательно задаём простой пароль супер-пользователя root, чтобы не забыть.



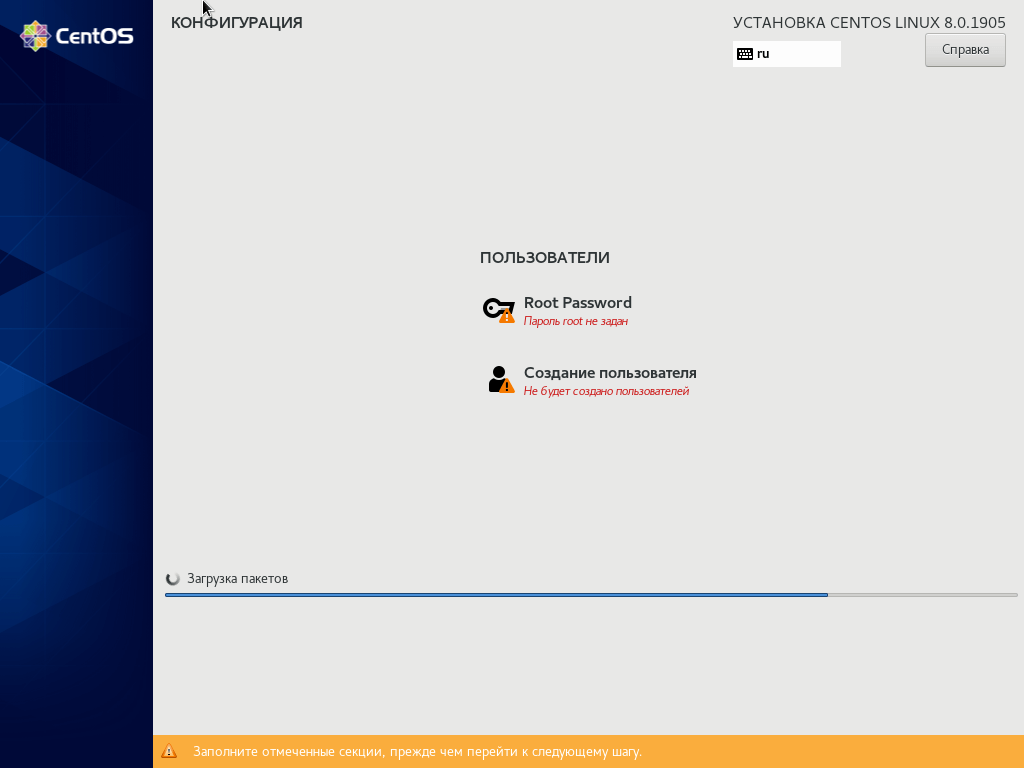
**Х) Cоздание дополнительно пользователя**

Тут можно долго и безрезультатно спорить…. но лучше не плодить лишних пользователей.

Кажется, всё готово. Нажмите расположенную в правом нижнем углу кнопку «Начать установку».



Примерно такое окно ждёт нас во время установки:



— Обратите внимание на подсказку мастера установки в нижней части окна.



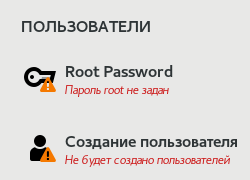
— Дождитесь завершения загрузки пакетов, установки и настройки компонентов. Может потребоваться до получаса.



*На реальном боевом сервере минимальная установка занимает пару-тройку минут.*

Во время установки администратор **может** совершать некоторые действия по настройке системы, что очень удобно.

Например, можно создать профиль пользователя и задать пароль администратора системы.



Щелкните левой кнопкой мыши по секции «Root Password», если ещё не задавали пароль. Введите пароль и подтвердите его во втором поле. Учетная запись администратора нужна для управления системой. Одинарный щелчок по кнопке «Готово».

Если установщик сочтет пароль ненадежным, он предложит пользователю подтвердить свое решение **двойным** щелчком левой кнопки мышью по кнопке «Готово». Причина недовольства мастера установки паролем будет указана на оранжевом фоне в нижней части окна.



**Создание учетной записи пользователя**

Не рекомендуется создавать дополнительных пользователей на данном шаге. И вообще не рекомендуется.

**Настройка кириллицы в Rocky Linux**

Иногда вывод кириллических символов нарушается: вместо букв отображаются белые квадраты. Это можно проверить, например, изучив вывод команды date.

Временное решение (до перезагрузки) - выполнить комаду:

setfont UniCyr\_8x16

Чтобы изменения были внесены на постоянной основе, необходимо в конфигурационном файле /etc/vconsole.conf исправить значение переменной:

FONT="UniCyr\_8x16"

**Задание**

Выполните установку ОС на виртуальную машину, рекомендуется **минимальная установка** без графического интерфейса. Опишите хронологию действий.

Выполните обновление операционной системы:

sudo dnf update -y

Интересно отметить, что в Rocky Linux 8 менеджером пакетов является dnf, но его же можно вызывать через псевдоним yum. В некоторых дистрибутивах менеджером пакетом является, наоборот, yum. Так как синтаксис у yum и dnf идентичен, фактически, эти команды эквивалентны:

sudo dnf update -y

sudo yum update -y

В отчете обязательно приведите вывод команд

* uname -a (информация об установленной ОС)
* df -h (информация о файловой системе)
* top (перечень запущенных процессов, **для выхода нажать q**)
* ping ya.ru -c 4 (демонстрация выхода в глобальную сеть)
* ip a (демонстрация активных интерфейсов) – какие адреса отобразились и почему?

Технически грамотное выключение виртуальной машины следует выполнять командой shutdown -h now.

# Лабораторная работа №2. Протокол SSH

**Цель работы**: сформировать практические навыки подключения к серверу по протоколу SSH

**Аннотация**

Рассмотрим, как подключиться по SSH к серверу из Linux или Windows. Придётся работать сразу на двух машинах.

Если подключились к серверу Linux из клиента Linux – **задача ЛР выполнена**, можно писать отчёт. Всё остальное – по желанию.

В идеальном случае: в VirtualBox запускайте сервер с Rocky Linux «без головы» (в фоновом режиме), запускайте клиента в режиме LiveCD без установки (например, Fedora MATE). Такой клиент потребует настройки двух сетевых адаптеров, после чего будет готов ходить в интернет и соединяться с сервером.

Внимание! Если в ЛР№1 не был настроен второй сетевой адаптер, сделайте это сейчас командой nmtui. Подробности – в самом конце ЛР.

Существует несколько способов подключения.

[1. Подключение через SSH в Linux](https://losst.ru/kak-podklyuchitsya-po-ssh#1_Подключение_через_SSH_в_Linux) из терминала

[2. Подключение через SSH в Windows](https://losst.ru/kak-podklyuchitsya-po-ssh#2_Подключение_через_SSH_в_Windows)

* клиент putty
* клиент BitVise
* командная строка

## Оборудование

Потребуется:

1) сервер Rocky Linux 8. Взять свой из ЛР 1 или скачать болванку с drach.pro (имя root, пароль toor)

2) Клиент под управлением Linux - скачиваем любой понравившийся "живой" образ

https://spins.fedoraproject.org/

https://lubuntu.me/downloads/

https://ubuntu-mate.org/download/

3) Клиент под управлением Windows (устанавливаем сами или скачиваем с drach.pro уже установленный образ Windows XP)

**Протокол SSH**

SSH (англ. Secure Shell — «безопасная оболочка») — сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений (например, для **передачи файлов**). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие от них, **шифрует весь трафик**, включая и передаваемые пароли. SSH допускает выбор различных алгоритмов шифрования. SSH-клиенты и SSH-серверы доступны для большинства **сетевых** операционных систем.

SSH позволяет безопасно передавать в незащищённой среде практически любой другой сетевой протокол. Таким образом, можно не только удалённо работать на компьютере через командную оболочку, но и передавать по шифрованному каналу звуковой поток или видео (например, с веб-камеры). Также SSH может использовать сжатие передаваемых данных для последующего их шифрования, что удобно, например, для удалённого запуска клиентов X Window System.

**Удобство работы**

Исторически так сложилось, что на большинстве серверов используется операционная система Linux, во многом этому посодействовала ее бесплатность, но в основном удобство и надёжность. Графический интерфейс на серверах Linux обычно не используется. И если раньше причиной была экономия ресурсов, то в настоящее время просто отсутствие острой необходимости.

Зачастую в командной строке Linux можно сделать больше, чем в графическом интерфейсе, эффективность работы выше. Протокол SSH позволяет выполнять команды в удаленной системе так, как будто вы это делаете в своей системе. Вам доступен **буфер обмена**, вы вводите команды и можете использовать их вывод. Недоступны разве что файлы из вашей файловой системы. Например, когда вы подключитесь к серверу по SSH из Linux, то все будет выглядеть так, как будто вы открыли терминал на своей физической машине.

**Как подключиться по SSH**

Для подключения по SSH необходимо знать такие данные:

* IP адрес сервера, к которому мы собираемся подключится;
* порт, на котором ожидает подключения SSH сервер, по умолчанию используется 22, но в целях безопасности порт подключения ssh часто изменяют;
* имя и пароль пользователя на удаленном сервере.

Больше ничего не нужно, на практике, если по-настоящему арендовать удалённый сервер, то обычно эти данные присылают в письме вместе с описанием VPS.

**Подготовка виртуальной машины**

На всех машинах поддерживаем включенными 2 адаптера: **Внутренняя сеть** и **NAT**.

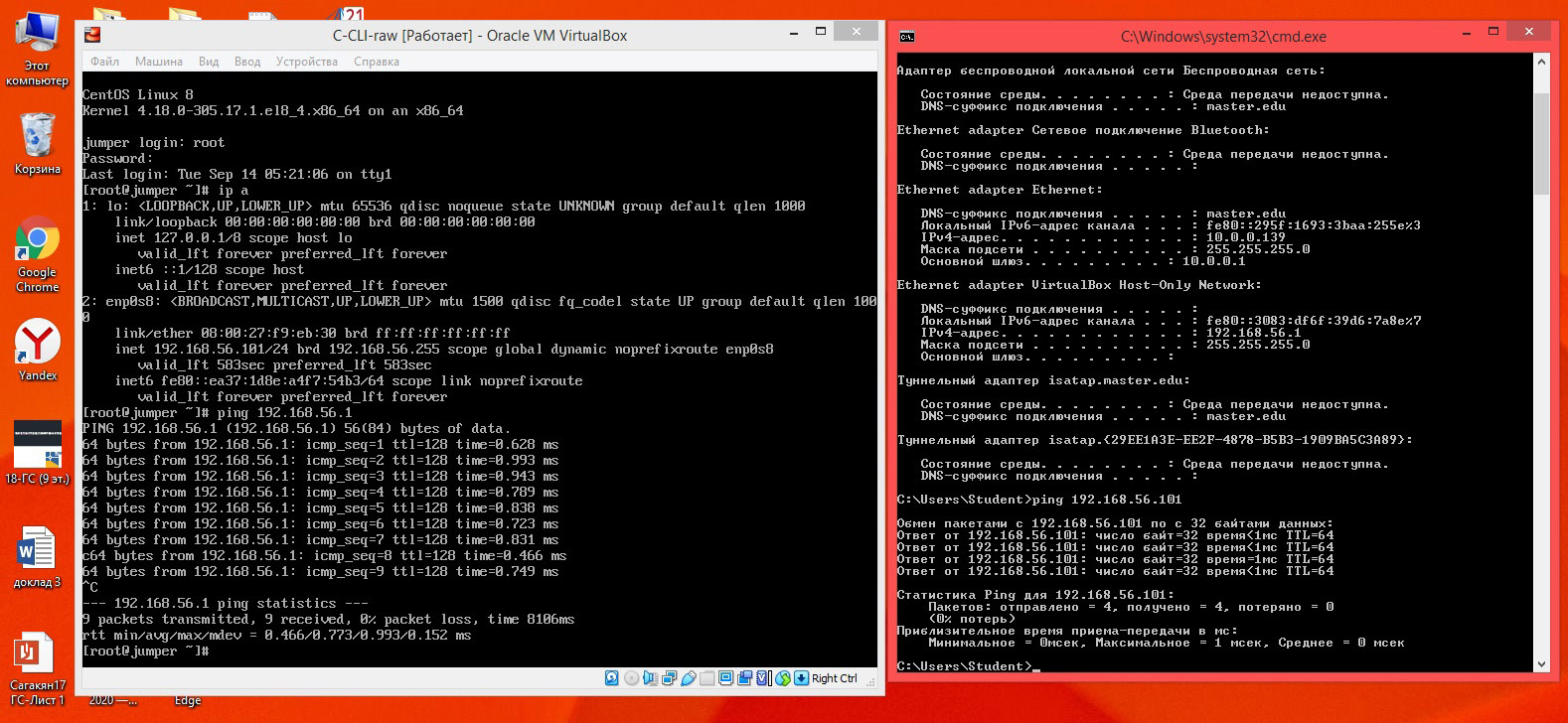
Для внутренней сети используем адреса 10.4.0.X маска 255.255.255.0, а для NAT просто указываем DHCP.

Для управления сетевыми настройками в Rocky Linux 8 после установки можно воспользоваться утилитой **nmtui**:

nmtui

Команда не найдена? Установим пакет:

yum install NetworkManager-tui



**1. Подключение через SSH в Linux**

В Linux подключение по SSH выполняется с помощью утилиты ssh, синтаксис будет выглядеть следующим образом:

**ssh имя\_пользователя@айпи\_адрес**

А если нужно задать порт, используйте опцию -p:

**ssh имя\_пользователя@айпи\_адрес -p порт**

**Внимание! Если работаем НЕ из-под пользователя root, то перед командой добавляем sudo. Может потребоваться ввод пароля на локальной машине.**

Если ip\_адрес и порт правильные, то на следующем шаге программа попросит у вас ввести пароль (при вводе пароля система **не показывает** введённые символы).

Например, команда:

**ssh root@10.4.0.1 -p 40220**

Ответ системы:

**root@10.4.0.1’s password:**

При **первом подключении** утилита попросит подтвердить добавление нового устройства в свой список известных устройств (Сообщение *The authenticity of host can’t be establisged*), нужно набрать **yes** и нажать **Enter**.

Теперь вы подключены, и все вводимые далее команды будут выполнены на удаленном сервере.

Если же выскакивает ошибка *Connection Refused*, хотя IP адрес и порт введены верно, то возможно нужно разрешить подключение по SSH в брандмауэре.

В дистрибутивах Rocky/Alma/Fedora:

firewall-cmd --permanent --zone=public --add-port=22/tcp

Вместо 22 надо указать нестандартный порт, который мы выбрали ранее.

**2. Подключение через SSH в Windows**

**2.1 Клиент putty**

Популярный клиент для Windows.

Скачать установщик данной программы можно на официальном сайте [www.putty.org](http://www.putty.org/).

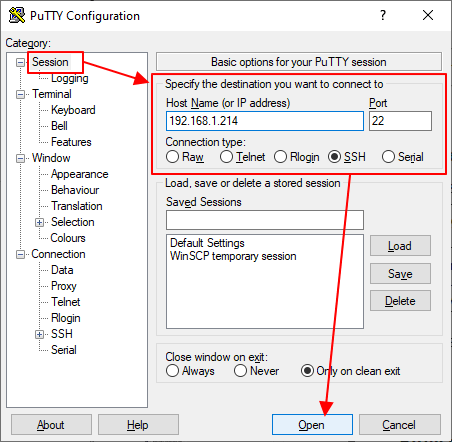
Или переносную версию по прямой ссылке (без рекламы и SMS):

drach.pro/images/documents/adm/putty.exe

Сразу после запуска программы PuTTY появляется окно с настройками подключения. Основными параметры здесь находятся на вкладке «Session», здесь нужно ввести:

* Имя компьютера или IP адрес;
* Порт для подключения (по умолчанию, 22);
* Способ подключения (SSH);

После ввода этих данных можно запускать подключение с помощью кнопки «Open».



Ключевые особенности клиента:

* Возможность копировать выделенный фрагмент правой кнопкой мыши.
* Масштабируемое окно
* Настраиваемые шрифты

**2.2 Клиент BitVise**

Можно воспользоваться программным обеспечением Bitvise SSH Client, которое выполняет туннелирование, предоставляет возможность защищенной передачи файлов между вашей машиной и сервером, а также имеет встроенный терминал для Shell-доступа к серверу. Кроме того, важным преимуществом этого SSH-клиента является бесплатная лицензия для индивидуального использования.

Скачать портативную версию можно тут:

drach.pro/images/documents/adm/Bitvise-SSH-Client.zip

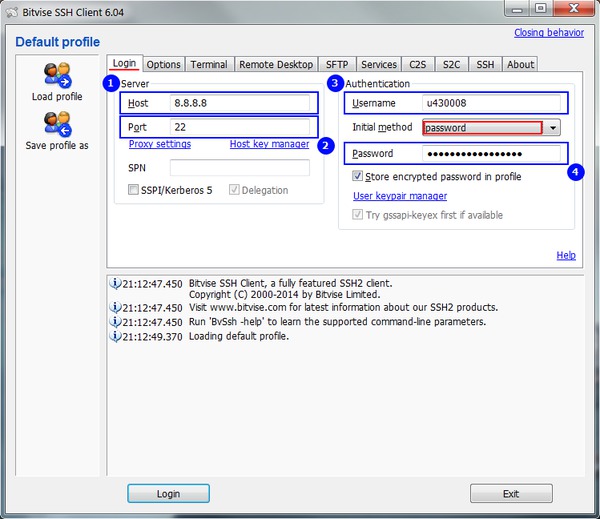
Запускается командой:

BvSsh -noRegistry

Расширенный синтаксис для подключения профиля в отдельном файле:

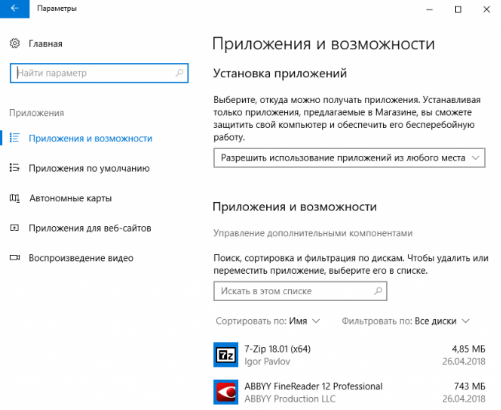
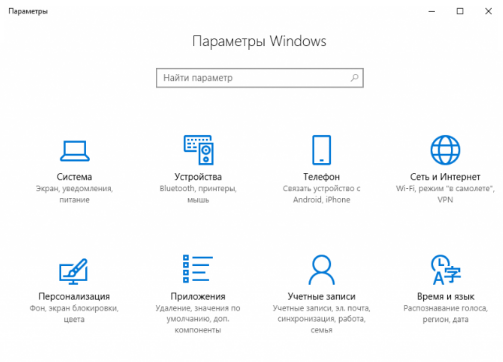
BvSsh -noRegistry -profile=U:\Profile.bscp

Для начала, на вкладке “Login” необходимо указать адрес сервера (1) и порт (2) для подключения, а также свои учетные данные – логин (3) и пароль (4).

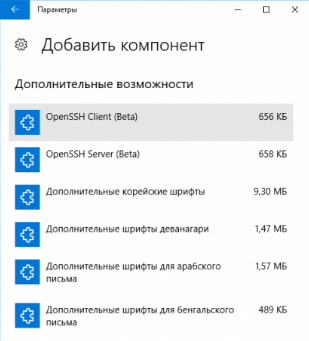


**2.3 Командная строка**

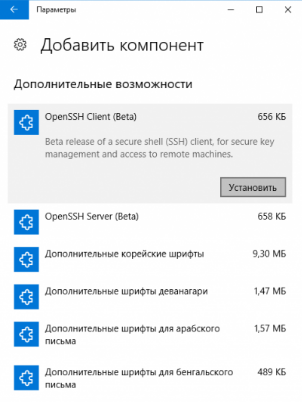
Раньше подключение по SSH из Windows выполнялось только с помощью сторонних утилит, например PuTTY. Но в Windows 10 был добавлен встроенный OpenSSH-клиент и работает он точно так же, как и в Linux. По умолчанию этот компонент не активирован. Для его установки откройте **Параметры** -> **Приложения**:



Затем выберите **Управление дополнительными компонентами**:

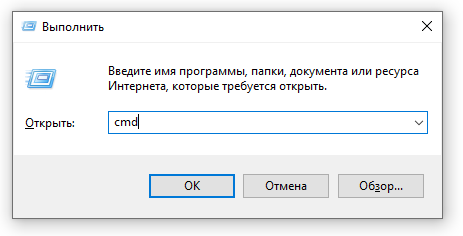


Нажмите «**Добавить новый компонент»** и в открывшемся меню выберите **OpenSSH Client**; нажмите **Установить**:



Дальше вернитесь назад и дождитесь завершения установки. После того, как SSH-клиент будет установлен, нужно обязательно перезагрузить компьютер.

После перезагрузки нажмите **Win+R** чтобы открыть окно запуска команд и наберите в нем cmd:



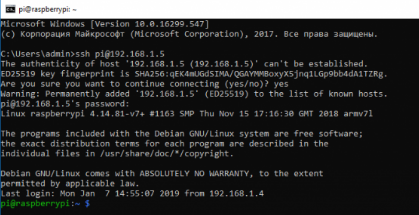
Нажмите кнопку **ОК**. Откроется командная строка Windows. Здесь можно использовать утилиту ssh. Синтаксис у нее абсолютно такой же, как и для Linux:

**ssh имя\_пользователя@айпи\_адрес -p порт**

Например, такой командой можно подключится по SSH к серверу или микрокомпьютеру Raspberry Pi, который находится в вашей локальной сети по адресу 192.168.1.5:

ssh pi@192.168.1.5

Утилита предложит добавить устройство в список известных, затем предложит ввести пароль. Если пароль введён правильно, мы попадаем в консоль удалённого устройства. Все следующие команды будут выполняться уже на Raspberry Pi или другой удаленной машине, к которой вы подключились.



Теперь подключиться к серверу по ssh из этой коммерческой операционной системы также просто, как и из Linux.

**Задание**

Выполнить подключение из клиента к серверу, хотя бы из Linux.

Объясните, как подключиться по SSH к серверу из Linux или Windows (из командной строки Windows - это задание повышенной сложности).

[Подключение через SSH в Windows](https://losst.ru/kak-podklyuchitsya-po-ssh#2_Подключение_через_SSH_в_Windows):

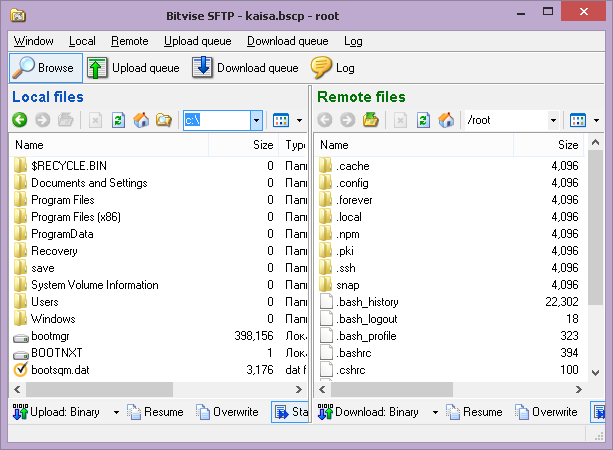
[-](https://losst.ru/kak-podklyuchitsya-po-ssh#Выводы) клиент putty (покажите подключение из физической машины на виртуальную)

- клиент BitVise (покажите подключение из физической машины на виртуальную, а также встроенный терминал и менеджер файлов)

В каждом варианте подключения выполните команду who.

Ответьте на вопрос – кто подключен к серверу в данный момент?

Попробуйте скопировать файлы с физической машины на виртуальную и наоборот. Используйте директорию /home или /tmp для хранения тестовых файлов.



### Troubleshooting

В случае, если в ЛР №1 не был настроен второй сетевой адаптер, можно настроить его сейчас

1. Через псевдографическую утилиту nmtui – выбираем ручную настройку (manual), вводим все данные (как в ЛР №1), активируем сетевой адаптер. Перемещаемся с помощью TAB.
2. C помощью командной строки – одна команда: **nmcli connection modify ether-enp0s8 ipv4.method manual ipv4.addresses 192.168.1.1/24 ipv4.gateway 192.168.1.254 ipv4.dns 8.8.8.8**

В редких случаях второй сетевой адаптер остаётся неактивным, несмотря на все усилия системного администратора, тогда придётся вручную исправлять конфигурационный файл. Открываем в редакторе:  
mcedit /etc/sysconfig/network-scripts/ifcfg-enp0s3

Изменяем состояние во время загрузки с «no» на «yes»:

ONBOOT=yes

Перезагружаем сервер.

Наконец, если вообще ничего не помогает, можно установить графическую оболочку и попроовать разобраться в проблеме в графическом режиме, изучая настройки сетевых интерфейсов. Автоматическая установка лёгкой графической оболочки выполняется с помощью двух команд.

curl -o xfce.sh https://drach.pro/images/documents/os/xfce.txt && chmod +x xfce.sh

./xfce.sh

Подробности и пояснения тут:

https://drach.pro/blog/linux/item/203-install-xfce-almalinux-8-5

Не забываем перезагрузить сервер.

# Лабораторная работа №3 Первоначальная настройка

Рекомендация: все действия в данной лабораторной работе (а, может, и последующих) удобнее всего выполнять в клиенте putty. Но можно и в терминале клиента под Linux в графическом режиме.

Команды не обязательно перенабирать вручную, их можно копировать CTRL+INS, SHIFT+INS.

#### Установка mc в Rocky Linux 8

GNU Midnight Commander или mc - это очень популярный файловый менеджер для работы с файлами в консоли сервера без графического интерфейса. Он распространяется под открытой лицензией и представляет из себя клон Norton Commander. Здесь вы можете выполнять все основные операции с файлами, такие как копирование, перемещение, кроме того, переименование, поддерживается работа с архивами и запуск скриптов. Интерфейс программы разделен на две панели, что делает её такой же удобной, как Far Manager или Norton Commander (если кто-то их помнит).

Установка:

**yum install mc -y**

Запуск файлового менеджера выполняем командой mc  
  
Теперь доступно редактирование файлов через mcedit, просмотр файлов по F3, редактирование по F4, это очень удобно!

#### Настройка времени

Можно сразу использовать одну команду, зная точно, что эта временная зона присутствует в настройках:

**timedatectl set-timezone Europe/Moscow**

Определим текущее время на сервере:

date

= вывод команды =

Ср ноя 11 11:11:00 CET 2023

Если же временная зона не настроена, или нет уверенности в правильности написания, идём сложным путём. Выполним необходимую настройку временной зоны на примере Europe/Moscow:

= Сделать резервную копию текущей временной зоны =

mv /etc/localtime /etc/localtime.bak

= В папке /usr/share/zoneinfo/ ищем что надо и делаем ссылку =

ln -s /usr/share/zoneinfo/Europe/Moscow /etc/localtime

В итоге получим корректный вывод команды:

date

#### Синхронизация времени

В Rocky Linux 8 по умолчанию используется утилита для синхронизации времени chrony. Если у вас её нет, то устанавливайте:

yum install chrony -y

Запускаем chrony и добавляем в автозагрузку:

systemctl start chronyd

systemctl enable chronyd

Проверяем правильность работы:

systemctl status chronyd

= вывод команды =

● chronyd.service - NTP client/server

Loaded: loaded (/usr/lib/systemd/system/chronyd.service; enabled; vendor preset: enabled)

Active: active (running) since Ср 2021-10-03 21:55:18 MSK; 27min ago

Теперь синхронизации времени будет выполняться автоматически.

### Отключение SELinux

Отключаем встроенную систему безопасности SELinux. Её использование и настройка в системе Rocky Linux -- отдельный вопрос. Вкратце: если мы серьёзно занимаемся безопасностью, нам не нужны подгузники в виде SELinux; если мы ничего не смыслим в безопасности, то однажды эти подгузники в виде SELinux протекут в самый неподходящий момент.

mcedit /etc/selinux/config

= меняем значение =

SELINUX=disabled

Перезагрузим машину для применения изменений – это весьма редкий случай, когда рекомендуется перезагрузка ОС Linux. После перезагрузки проверяем статус:

sestatus

= вывод команды =

SELinux status: disabled

SELinux -- система, требующая хорошего понимания в том, как она работает и как её настраивать. Если её сразу не отключить, в будущем будут появляться совершенно нелепые ошибки в самых простых ситуациях в неожиданное время.

### Добавление репозиториев

Для инсталляции различных пакетов и приложений необходимо подключить репозитории в Rocky Linux 8. Относиться к добавляемым репозиториям надо очень внимательно, чтобы в последствии не возникало проблем при обслуживании.

Всегда подключаем самый популярный репозиторий **Epel** (Extra Packages for Enterprise Linux). Epel это хранилище пакетов, созданное группой специалистов родственного дистрибутива Fedora. Пакеты из репозитория Epel никогда не конфликтуют и не переустанавливают базовые пакеты Rocky|RHEL|Alma.

yum -y install epel-release

Рационально подключать “авторские” репозитории, которые поддерживают сами разработчики программ. Например, такие как Nginx, Zabbix, MariaDB. Не будем делать это в академических целях.

### Обновление Rocky Linux 8

Обновление системы очень важный момент и следует относится к нему очень внимательно. Всегда проще найти проблему, когда проходит немного обновлений.

При решении реальных задач всегда перед обновлениями делайте резервные копии!

Полное обновление системы:

yum update

или

dnf update

### Популярные и полезные утилиты

Установим основные популярные утилиты, которые обычно требуются в работе.

dnf -y install net-tools bind-utils htop atop iftop lsof wget bzip2 traceroute gdisk yum-utils

* **net-tools bind-utils** — сетевые утилиты после установки которых будут работать команды ifconfig и nslookup,
* **htop, atop** — два изысканных диспетчера задач,
* **iftop** — показывает в режиме реального времени загрузку сетевого интерфейса,
* **lsof** — вывода информации о том, какие файлы используются теми или иными процессами,
* **wget** — неинтерактивная мощная консольная программа для загрузки файлов из сети,
* **bzip2** — утилита командной строки с открытым исходным кодом для сжатия данных,
* **traceroute** — утилита, предназначенная для определения маршрутов следования данных в сетях TCP/IP,
* **gdisk** — умеет работать с GPT-дисками, разработана по подобию fdisk,
* **yum-utils** — это набор утилит для управления репозиторием, установки и отладки пакетов, поиска пакетов и много другого.

### Отправка почты с авторизацией на другом SMTP-сервере

Задание повышенной сложности,  
выполняем по собственному желанию.

Настраиваем на базе Postfix отправку уведомлений по email.

Проверка статуса:

systemctl status postfix

Если Postfix не установлен, то производим установку выполнив команду:

dnf install postfix

Запускаем и добавляем в автозагрузку командами:

systemctl start postfix

systemctl enable postfix

Чтобы отправлять письма из консоли необходимо установить пакет mailx:

dnf install mailx

После этого можно отправлять с консоли сервера сообщения на свою почту:

uname -a | mail -s "info server" petr@petrov.ru

В результате на почту придёт сообщение c предупреждением:

Мы не уверены в подлинности отправителя этого письма. Возможно, данные были подменены.

Отправитель: ivanov.ru

Подпись: Нет

Шифрование: Нет

Из-за вышесказанного сообщение может попасть в спам, либо вообще не будет принято сервером получателя, потому что на сервере нет корректных настроек для отправки почты. Чтобы почта корректно отправлялась, надо воспользоваться внешним почтовым сервером.

Например, весьма удобно использовать электронную почту, которая обслуживается Yandex.

Переименуем конфигурационный файл Postfix по умолчанию. После этого, создадим рабочий конфигурационный файл и добавим необходимые настройки:

mv /etc/postfix/main.cf /etc/postfix/main.cf\_orig

mcedit /etc/postfix/main.cf

= необходимый код =

# DEFAULT CONFIG

compatibility\_level = 2

queue\_directory = /var/spool/postfix

command\_directory = /usr/sbin

daemon\_directory = /usr/libexec/postfix

data\_directory = /var/lib/postfix

mail\_owner = postfix

# Имя сервера, которое выводит команда hostname

myhostname = vds-micro2.ivanov.ru

# По логике нужно оставлять только домен, но в данном случае лучше оставить полное имя сервера, чтобы в поле отправитель

# фигурировало полное имя сервера, так удобнее разбирать служебные сообщения

mydomain = vds-micro2.ivanov.ru

myorigin = $mydomain

mydestination = $myhostname, localhost.$mydomain, localhost

inet\_interfaces = localhost

inet\_protocols = all

unknown\_local\_recipient\_reject\_code = 550

alias\_maps = hash:/etc/aliases

alias\_database = hash:/etc/aliases

debug\_peer\_level = 2

debugger\_command =

PATH=/bin:/usr/bin:/usr/local/bin:/usr/X11R6/bin

ddd $daemon\_directory/$process\_name $process\_id & sleep 5

sendmail\_path = /usr/sbin/sendmail.postfix

newaliases\_path = /usr/bin/newaliases.postfix

mailq\_path = /usr/bin/mailq.postfix

setgid\_group = postdrop

html\_directory = no

manpage\_directory = /usr/share/man

sample\_directory = /usr/share/doc/postfix/samples

readme\_directory = /usr/share/doc/postfix/README\_FILES

smtpd\_tls\_cert\_file = /etc/pki/tls/certs/postfix.pem

smtpd\_tls\_key\_file = /etc/pki/tls/private/postfix.key

smtpd\_tls\_security\_level = may

smtp\_tls\_CApath = /etc/pki/tls/certs

smtp\_tls\_CAfile = /etc/pki/tls/certs/ca-bundle.crt

meta\_directory = /etc/postfix

shlib\_directory = /usr/lib64/postfix

# SMTP YANDEX CONFIG

relayhost = smtp.yandex.ru:587

smtp\_use\_tls = yes

smtp\_sasl\_auth\_enable = yes

# База параметров авторизации с SMPT

smtp\_sasl\_password\_maps = hash:/etc/postfix/sasl\_passwd

# База подмены имени почтового ящика

smtp\_generic\_maps = hash:/etc/postfix/generic

smtp\_sasl\_security\_options = noanonymous

smtp\_tls\_security\_level = may

Создаем файл с информацией об имени пользователя и пароле для авторизации на сервере SMTP:

mcedit /etc/postfix/sasl\_passwd

= необходимый код =

smtp.yandex.ru:587 noreply@ivanov.ru:PASSWORD

Создаем db файл:

postmap /etc/postfix/sasl\_passwd

Для того чтобы почта уходила надо чтобы адрес отправителя, а адрес авторизации на сервер smtp совпадал!

В случае если у нас имеется сайт, отправляющий почту средствами PHP, можно настроить корректную отправку почты.

Переименуем базу (по умолчанию) подмены имени почтового ящика. После этого, создадим рабочий и добавим необходимые настройки:

mv /etc/postfix/generic /etc/postfix/generic\_orig

mcedit /etc/postfix/generic

= необходимый код =

root@ivanov.ru noreply@ivanov.ru

ivanov.ru@php7-lxc-vds-micro2.ivanov.ru noreply@ivanov.ru

Создаем db-файл:

postmap /etc/postfix/generic

При получении ошибки «warning: SASL authentication failure: No worthy mechs found» Необходимо установить еще несколько пакетов для работы SASL!

Установим необходимые пакеты:

dnf -y install cyrus-sasl cyrus-sasl-lib cyrus-sasl-plain

Перезапустим Postfix и проверим работу.

systemctl restart postfix

Отправляем тестовое письмо через консоль.

uname -a | mail -s "info server" info@ivanov.ru

Результат:

Отправитель письма подтверждён и проверен.

Отправитель: \*\*\*

Подпись: \*\*\*

Шифрование: Да

Как видим теперь всё работает как надо!

В случае ошибок проверяем почтовый лог выполнив команду:

tail -n 10 /var/log/maillog

В заключение, осталось добавить псевдоним для root в /etc/aliases -- внешний адрес, куда будет дублироваться почта, адресованная root. Для этого редактируем указанный файл, изменяя последнюю строку:

mcedit /etc/aliases

= необходимо привести к виду =

# Person who should get root's mail

root: root,info@ivanov.ru

Обновляем базу сертификатов:

newaliases

### Сетевые параметры

При установке Rocky Linux 8 уделяeм особое внимание настройке сетевых параметров. Кроме того, на странице настройки сетевых параметров указывается название хоста.

Всегда отключаем ipv6, так как в большинстве случаев он не используется, а вот проблемы при работе с системой может создать.

На данном этапе всё уже должно быть настроено верно. Но если есть необходимость дополнительного управления сетевыми настройками, в Rocky Linux после установки можно воспользоваться утилитой nmtui:

nmtui

= вывод команды =

-bash: nmtui: команда не найдена

= команда установки необходимого пакета для работы nmtui =

**yum install NetworkManager-tui**

В **Rocky Linux 8**, сеть управляется как правило через **Network Manager**. Сетевые настройки лучше выполнять с помощью утилиты **nmtui**.

Настройка вручную сводится к редактированию файлов в паке /etc/sysconfig/network-scripts/. Для определения интерфейсов необходимо предварительно выполнить команду **ip a** которая покажет название всех имеющихся интерфейсов.

Например, файл настройки сетевого интерфейса ens18:

mcedit /etc/sysconfig/network-scripts/ifcfg-ens18

= вывод команды =

TYPE="Ethernet"

PROXY\_METHOD="none"

BROWSER\_ONLY="no"

BOOTPROTO="none"

DEFROUTE="yes"

IPV4\_FAILURE\_FATAL="yes"

IPV6INIT="no"

IPV6\_AUTOCONF="yes"

IPV6\_DEFROUTE="yes"

IPV6\_FAILURE\_FATAL="no"

IPV6\_ADDR\_GEN\_MODE="stable-privacy"

NAME="ens18"

UUID="2f1479ad-8a41-4886-acf4-64f0c5eff5c2"

DEVICE="ens18"

ONBOOT="yes"

IPADDR="10.4.0.1"

PREFIX="24"

GATEWAY="10.4.0.10"

DNS1="10.4.0.100"

ZONE=public

Именно за дистрибутивом Rocky Linux 8 замечена удивительная особенность: иногда для второй сетевой карты система отключает автоматический старт после запуска. Приходится вручную менять ONBOOT="no" на ONBOOT="yes".

Для применения изменений необходимо перезагрузить Network Manager выполнив команду:

= или командой =

systemctl restart NetworkManager

= предварительно выяснив название необходимого сервиса =

systemctl list-units | grep -i network

### Смена пароля root

Смена пароля производится командой:

passwd

= вывод команды =

Изменяется пароль пользователя root.

Новый пароль:

Повторите ввод нового пароля:

passwd: все токены аутентификации успешно обновлены.

### Брандмауэр FirewallD

По умолчанию в системе Rocky Linux 8 используется межсетевой экран FirewallD.

Вывод информации об активных зонах:

firewall-cmd --get-active-zones

= вывод команды =

public

interfaces: ens18

Выведем информацию о конкретной зоне:

firewall-cmd --list-all --zone=public

= вывод команды =

public (active)

target: default

icmp-block-inversion: no

interfaces: ens18

sources:

services: cockpit ssh dhcpv6-client

ports:

protocols:

masquerade: no

forward-ports:

source-ports:

icmp-blocks:

rich rules:

Для безопасности порт ssh лучше поменять на нестандартный.

Добавим разрешение подключаться по этому порту:

firewall-cmd --permanent --zone=public --add-port=25555/tcp

= вывод команды =

success

Кроме того, сразу удалим ненужный сервис:

firewall-cmd --permanent --zone=public --remove-service=dhcpv6-client

= вывод команды =

success

Применим изменения и посмотрим результат сделанных действий:

firewall-cmd --reload

= вывод команды =

success

firewall-cmd --list-all --zone=public

= вывод команды =

public (active)

target: default

icmp-block-inversion: no

interfaces: ens18

sources:

services: cockpit ssh

ports: 25555/tcp

protocols:

masquerade: no

forward-ports:

source-ports:

icmp-blocks:

rich rules:

Только после настройки ssh для работы по новому порту удаляйте сервис ssh из FirewallD! Иначе в случае ошибки настройки можете потерять доступ к серверу.

### Смена порта SSH

Укажем порт в настройках ssh открыв конфигурационный файл командой:

mcedit /etc/ssh/sshd\_config

= необходимые изменения =

Раскоментируем строку Port 22 и заменим на какой-нибудь фантастический 44220. Выберите порт в диапазоне от 2000 до 65535.

Перезапускаем сервис ssh командой:

systemctl restart sshd

Проверяем какой порт слушает sshd (для работы нужен установленный пакет net-tools):

netstat -tulpn | grep sshd

tcp 0 0 0.0.0.0:25555 0.0.0.0:\* LISTEN 453/sshd

tcp6 0 0 :::25555 :::\* LISTEN 453/sshd

#### Подключение к SSH по нестандартному порту

Подключение производится с указанием необходимого порта командой:

ssh -p 44220 root@13.12.11.10

### Установка Tmux

### Задание повышенной сложности

Установим, пожалуй, одну из самых необходимых и удобных программ для удаленной работы с сервером в реальных условиях. Терминальный оконный менеджер Tmux дает возможность при обрыве связи с сервером по ssh не терять информацию о выполняемых действиях.

Установка выполняется командой:

dnf install tmux

Более подробная информация по настройке и работе с программой Tmux может быть легко найдена в интернете.

## Cockpit: Веб-интерфейс управления сервером Rocky Linux 8

Возможность видеть наглядно все основные параметры системы и иметь возможность ими управлять прямо в обычном браузе предоставляет специальный пакет Cockpit.

Достаточно установить пакет и выполнить запуск:

**dnf install cockpit -y**

**systemctl enable --now cockpit.socket**

...и при следующей авторизации вы увидите следующее сообщение:

Web console: https://localhost:9090/ or https://10.2.0.15:9090/

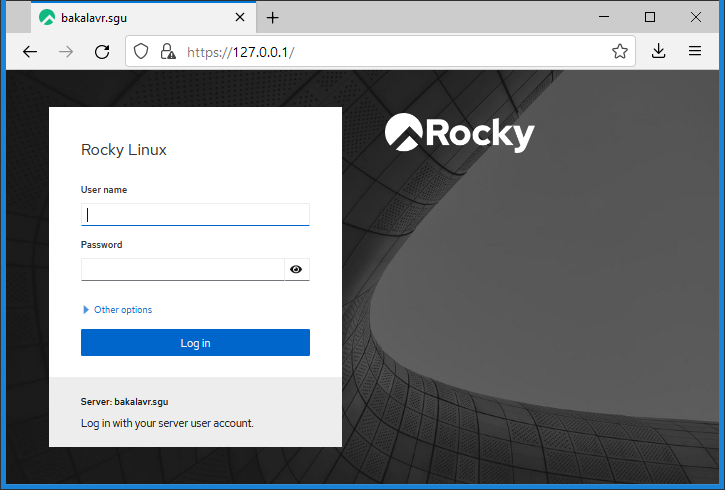
То есть прямо в консоли нам подсказывают, как следует подключаться к веб-интерфейсу.

Порт 9090 должен быть открыт. Работающий по умолчанию сервис cockpit в FirewallD открывает этот порт. Можно открыть порт вручную:

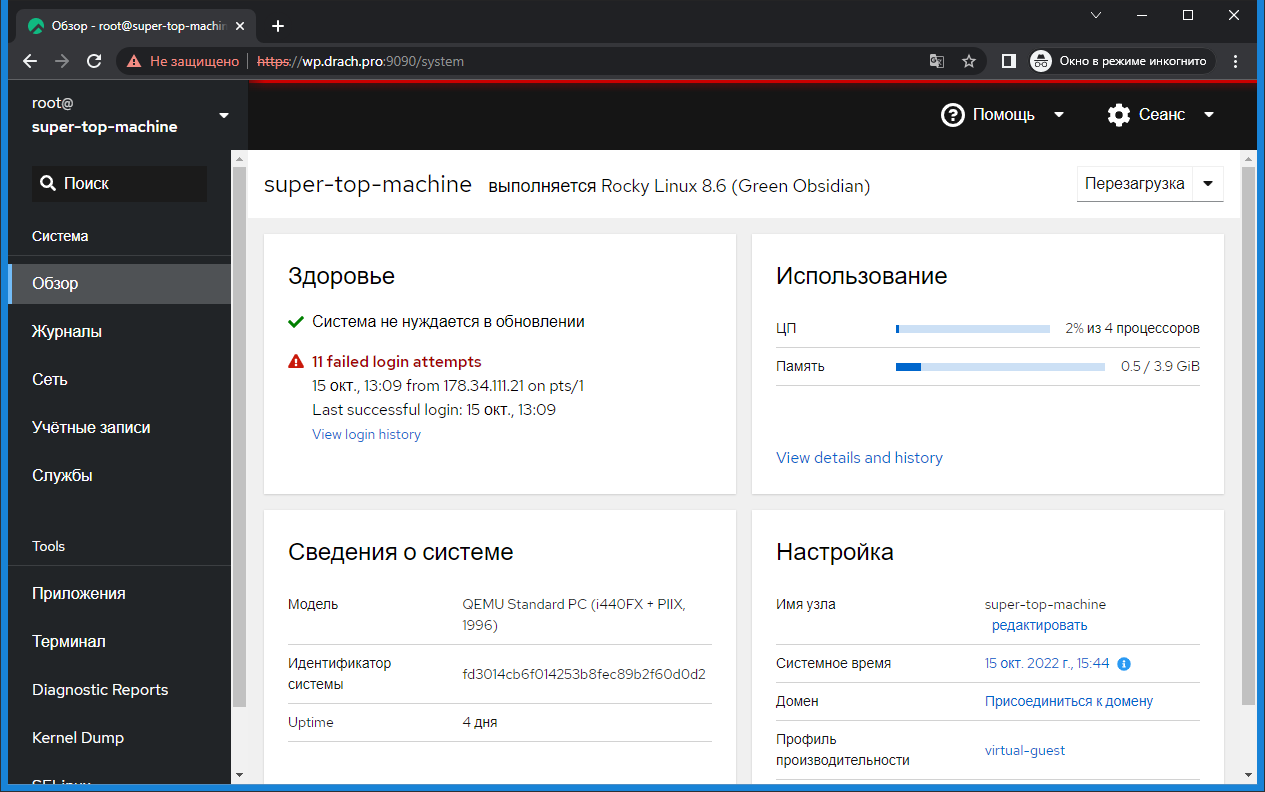
**sudo firewall-cmd --permanent --zone=public --add-service=cockpit**

**sudo firewall-cmd --reload**

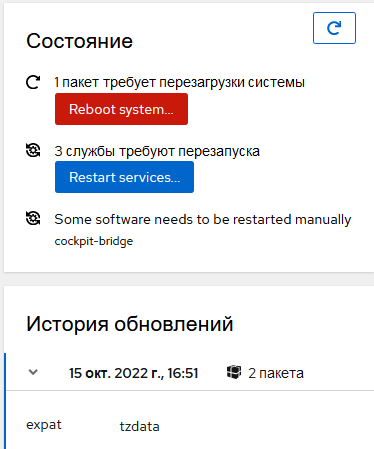
Если наш сервер доступен в интернете по белому IP-адресу, то достаточно ввести в браузере адрес http://10.2.0.15:9090, и мы увидим приглашение выполнить вход.



После входа вы увидите главную страницу с основными параметрами и нагрузкой сервера.



Интерфейс выглядит очень понятным и логичным, слева вы увидите навигационную панель: можно что-то настраивать, читать журналы, даже есть терминал. Замечательно, что система сама докладывает о состоянии и подсказывает, что требуется перезапуск служб или полный перезапуск.



Конечно, не все можно видеть и настраивать в этой панели, но определенное удобство есть явно – например визуализация нагрузки на сервер.

В целях безопасности не следует всегда держать её включенной на реальном сервере. Убираем из автозагрузки в самой панели управления или командой:

systemctl disable cockpit.socket

Лучше включить при необходимости и отключить после использования следующими командами:

systemctl start cockpit.socket

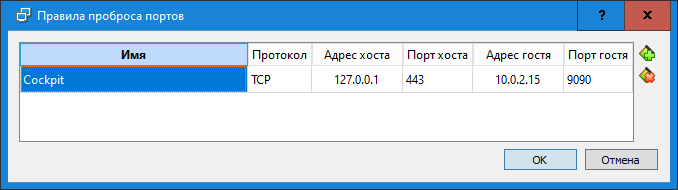
systemctl stop cockpit.socket

При выполнении лабораторных работ в СГУ у нас не выйдет простым способом получить доступ к веб-интерфейсу. Придётся идти обходными путями:

Или запускать вторую виртуальную машину, и пытаться подключаться из неё (не будем описывать, и так всё ясно). Или подключаться с физической машины (опишем ниже).

Для того, чтобы подключаться с физической машины придётся выполнить проброс портов и установить браузер, который позволяет указывать прокси-сервер в обход системных настроек ОС Windows.

Проброс портов выполняем в VirtualBox, добавляя одно правило

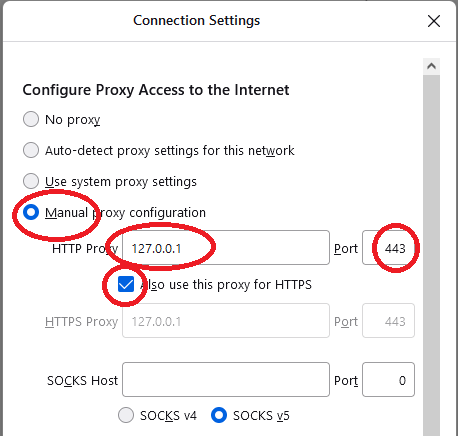


Надо было бы указать стандартный порт 80 для протокола HTTP, но все современные браузеры считают HTTPS протоколом по умолчанию, потому указываем 443. Подробнее про проброс портов – в самом конце методички.

В качестве браузера выбираем что-то портативное, например клон FireFox, который удаётся устанавливать в а.107 в папку TEMP.

https://www.waterfox.net/download/

После установки выполняем необходимые настройки браузера в разделе Сеть.



Теперь веб-интерфейс доступен по адресу https://127.0.0.1 непосредственно на нашей физической машине. При первом подключении придётся согласиться с «рисками» перехода на страницу без сертификата SSL. Ищем в браузере что-то типа «Дополнительно» / «Принять риски и продолжить».

## Задание

— Выполнить базовую настройку операционной системы по описанной выше методике.

— Выполнить замену стандартной оболочки пользователя по методике:

https://drach.pro/blog/linux/item/237-obolochka-polzovatelya-fish

— Выбрать индивидуальные цвета для приглашения командной строки.

# Лабораторная работа №4 Доступ по ключу из Linux

В данной работе предполагаем, что администратор работает из-под операционной системы Linux. Выполнять работу лучше всего, скачав болванку и запустив две копии виртуальной машины.

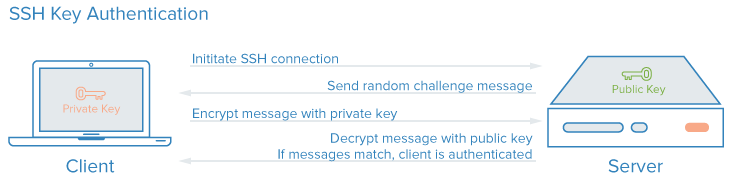
**Теория**

SSH сервер может выполнять аутентификацию пользователей с помощью различных алгоритмов. Самый популярный - это аутентификация по паролю. Он достаточно прост, но не очень безопасный. Пароли передаются по безопасному каналу, но они недостаточно сложны для противостояния попыткам перебора. Вычислительная мощность современных систем в сочетании со специальными скриптами делают перебор очень простым. Конечно, существуют другие способы дополнительной безопасности, например, fail2ban, но аутентификация по ключу SSH более надежна.

Каждая пара ключей состоит из открытого и закрытого ключа. Секретный ключ сохраняется на стороне клиента и не должен быть доступен кому-либо еще. Утечка ключа позволит злоумышленнику войти на сервер, если не была настроена дополнительная аутентификация по паролю.

Открытый ключ используется для шифрования сообщений, которые можно расшифровать только закрытым ключом. Это свойство и используется для аутентификации с помощью пары ключей. Открытый ключ загружается на удаленный сервер, к которому необходимо получить доступ. Его нужно добавить в специальный файл ~/.ssh/authorized\_keys.

Когда клиент попытается выполнить проверку подлинности через этот ключ, сервер отправит сообщение, зашифрованное с помощью открытого ключа, если клиент сможет его расшифровать и вернуть правильный ответ - аутентификация пройдена.



**Как создать ключи SSH?**

Скачиваем болванку, делаем дважды импорт в VirtualBox. Например, получаем виртуальные машины Student1, Student2.



Для удобства изменим имена машин, выполнив на них:

hostmanectl set-hostname student1

hostmanectl set-hostname student2

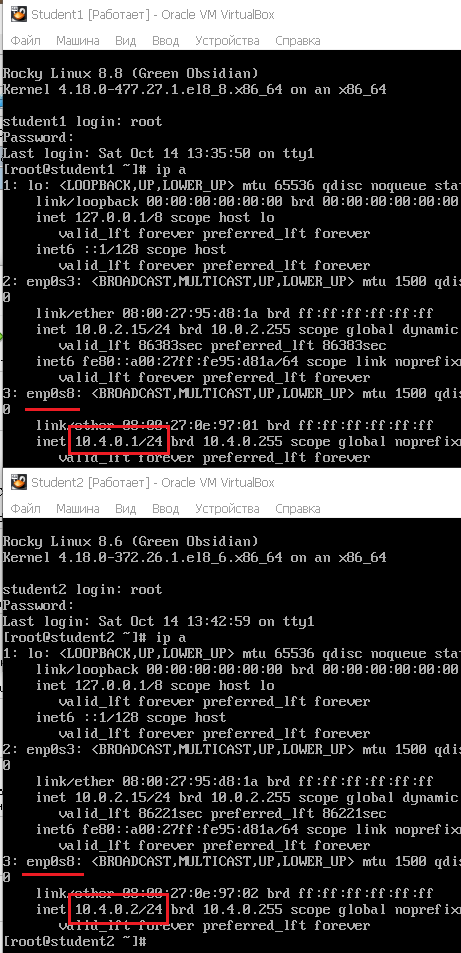
Чтобы машины могли взаимодействовать, изменим на единичку IP-адрес одной из них.

Утилита nmtui, Edit connection > enp0s8, меняем адрес.

Было 10.4.0.1/24  
Стало 10.4.0.2/24

Перезагружаем обе: reboot.

Проверяем себя после перезагрузки: ip a



Договоримся, что student1 будет у нас притворяться сервером, а student2 клиентом. Работать будем в консоли клиента – student2.

**Как создать ключи SSH?**

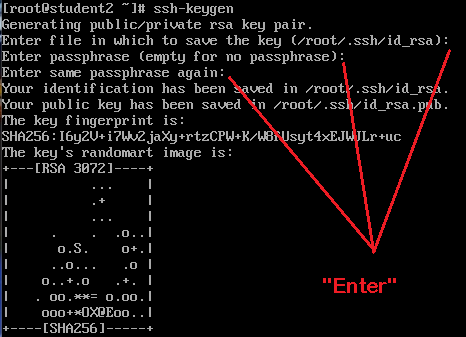
Сначала необходимо создать ключи ssh для аутентификации на локальном сервере. Для этого существует специальная утилита ssh-keygen, которая входит в набор утилит OpenSSH. По умолчанию она создает пару 2048 битных RSA ключей, которая подойдет не только для SSH, но и для большинства других ситуаций.

Генерация ключей ssh выполняется командой:

ssh-keygen

Утилита предложит вам выбрать расположение ключей. По умолчанию ключи располагаются в папке ~/.ssh/. Лучше ничего не менять, чтобы все работало по умолчанию и ключи автоматически подхватывались. Секретный ключ будет называться id\_rsa, а публичный id\_rsa.pub.

Затем утилита предложит ввести пароль для дополнительного шифрования ключа на диске. Его нужно **не** указывать, жмем Enter.

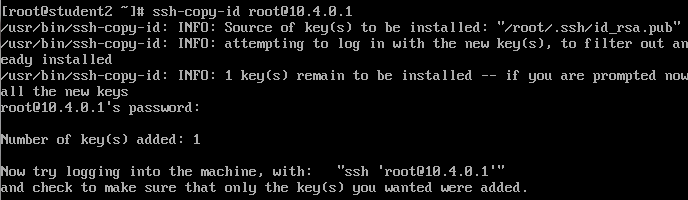


Теперь у вас есть открытый и закрытый ключи SSH, и вы можете использовать их для проверки подлинности. Дальше нам осталось разместить открытый ключ на удаленном сервере.

**Загрузка ключа на сервер**

Самый простой способ скопировать ключ на удаленный сервер - это использовать утилиту ssh-copy-id. Она тоже входит в пакет программ OpenSSH. Но для работы этого метода вам нужно иметь пароль доступа к серверу по SSH. Синтаксис команды:

ssh-copy-id username@remote\_host



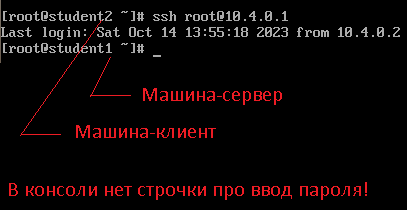
При первом подключении к серверу система может не распознать новый компьютер, поэтому нужно ввести yes. Затем введите ваш пользователя на удаленном сервере. Утилита подключится к удаленному серверу, а затем использует содержимое ключа id.rsa.pub для загрузки его на сервер в файл ~/.ssh/authorized\_keys. Дальше вы можете выполнять аутентификацию с помощью этого ключа.

|  |
| --- |
| **Задание повышенной сложности (не выполнять)**  Если такой способ по какой-либо причине не работает, вы можете скопировать ключ по ssh вручную. Мы создадим каталог ~/.ssh, а затем поместим наш ключ в файл authorized\_keys с помощью символа >>, это позволит не перезаписывать существующие ключи:  cat ~/.ssh/id\_rsa.pub | ssh username@remote\_host "mkdir -p ~/.ssh && cat >> ~/.ssh/authorized\_keys"  Здесь вам тоже нужно набрать yes, если вы подключаетесь к новому серверу, а затем ввести пароль. |

Теперь вы можете использовать созданный ключ для аутентификации на сервере:

 ssh root@10.4.0.1

Вы сразу же будете авторизованы, что очень удобно.



|  |
| --- |
| **Отключение проверки пароля**  **Задание повышенной сложности (не выполнять)**  Если пароль больше не будет использоваться, то для увеличения безопасности системы лучше его вовсе отключить. Но убедитесь, что ключ надежно сохранен и вы его не потеряете, потому что по паролю вы больше не войдете. Авторизуйтесь на сервере, затем откройте конфигурационный файл /etc/ssh/sshd\_config и найдите там директиву PasswordAuthenticatin. Нужно установить ее значение в No:  mcedit /etc/ssh/sshd\_config  PasswordAuthentication no  Теперь сохраните файл и перезапустите службу ssh:  service sshd restart  Дальше будет возможно только подключение по ключу ssh, пароль не будет приниматься. |

#### Задача:

- Сгенерировать пару ключей

- Публичный ключ поместить на сервер

- Подключиться к серверу без пароля

- Продемонстрировать скриншот, который доказывает, что при подклчении использован индивидуальный ключ.

# Лабораторная работа №5 Доступ по ключу из Windows

#### Ход выполнения работы:

https://drach.pro/blog/linux/item/213-putty-ssh-login-with-key

#### Задача:

- Сгенерировать пару ключей длиной от 2048

- Публичный ключ поместить на сервер

- Подключиться к серверу без пароля

- Продемонстрировать скриншот, который доказывает, что при подклчении использован индивидуальный ключ

# Лабораторная работа №6 Настройка web-сервера apache

Задача. Установить на сервер под управлением Rocky Linux связку из сервера **apache**, интерпретатора **php** с модулем prefork. Развернуть тестовый сайт.

Работать будем на сервере под управлением Rocky Linux 8.

Определения

|  |  |
| --- | --- |
| RockyLinux.dve | имя тестового виртуального хоста и сайта (**придумать своё!**) |
| /www/html/ | директория для размещения виртуальных доменов (т.е. сайтов) |
| 10.4.0.1 | локальный ip адрес сервера |
| 10.4.0.2 | локальный ip адрес клиента |
|  |  |
| pma.rockylinux.dve | доменное имя для phpmyadmin |
| Virtual Host | В терминологии Апача это любой веб-сайт (кроме сайта по умолчанию) |

**Web сервер на Rocky Linux 8**

Стек LAMP содержит три основных компонента - http сервера **apache**, интерпретатора языка программирования **php**и СУБД **mysql**. Познакомимся немного с каждым из них:

1. **Apache** - http-сервер или веб-сервер. Является кросплатформенным ПО, поддерживающим практически все популярные операционные системы, в том числе и Windows. Ценится прежде всего за свою надежность и гибкость конфигурации, которую можно существенно расширить благодаря подключаемым модулям, которых существует множество. Из недостатков отмечают большую требовательность к ресурсам, по сравнению с другими серверами. Держать такую же нагрузку, как nginx, apache не сможет при одинаковых аппаратных ресурсах.
2. **PHP** - язык программирования общего назначения, который чаще всего применяется в веб-разработке. На сегодняшний день это самый популярный язык в этой области применения.
3. **MySQL** - система управления базами данных. Завоевала свою популярность в среде малых и средних приложений, которых очень много в Интернет. На сегодняшний день является самой популярной СУБД, использующейся на веб-сайтах. Поддерживается большинством хостингов. В Rocky Linux вместо mysql устанавливается **mariadb** - ответвление mysql. Они полностью совместимы, возможен в любой момент переход с одной СУБД на другую и обратно. Однако, на реальных проектах mariadb показала себя ненадёжной.

## Настройка межсетевого экрана

В firewalld открыть порты для web сервера можно следующими командами

# firewall-cmd --permanent --zone=public --add-service=http

# firewall-cmd --permanent --zone=public --add-service=https

# firewall-cmd --reload

Проверить, открылись ли порты, можно командой

# firewall-cmd --list-all

Убеждаемся, что в строчке services видна наша служба.

## Настройка apache в Rocky Linux 8

Сразу установим инструменты для отладки:

yum -y install net-tools

Теперь приступим к **установке apache**. Именно в дистрибутиве Alma, Rocky и Fedora как пакет, так и служба веб-сервера Apache называются httpd (Л – логика).

yum install httpd -y

Добавляем apache в автозагрузку:

# systemctl enable httpd

Запускаем apache в Rocky Linux 8:

# systemctl start httpd

Проверяем, запустился ли сервер:

# netstat -tulnp | grep httpd

tcp6       0      0 :::80           :::\*      LISTEN      2186/httpd

Все в порядке, повис на 80-м порту, как и положено. Уже сейчас можно зайти по адресу http://10.4.0.1 и узнать, что доступ запрещён: "403 Forbidden".

B Rocky Linux директория web-сервера по-умолчанию /var/www/html. Изначально там пусто. Давайте сами что-то туда положим. Создаем простую страничку index.html.

# echo "Welcome to Rocky Linux 8 and Apache!" >> /var/www/html/index.html

Проверяем страницу http://10.4.0.1. Если страница отобразилась у клиента, веб-сервер работает. Теперь займемся настройкой apache.

### Сайт

Создаем для нового сайта директории и указываем, что Apache будет владельцем.

# mkdir -p /var/www/html/rockylinux.dve/

# chown -R apache. /var/www/html/rockylinux.dve/

Редактируем файл конфигурации apache — **httpd.conf** по адресу /etc/httpd/conf. Первым делом проверим, раскомментирована ли строчка в самом конце:

IncludeOptional conf.d/\*.conf

Если нет, раскомментируем.

Идем в каталог конфигурации сайтов /etc/httpd/conf.d. Создаем файл RockyLinux.dve.conf:

# mcedit /etc/httpd/conf.d/rockylinux.dve.conf

<VirtualHost \*:80>

ServerName rockylinux.dve

ServerAlias www.rockylinux.dve

DocumentRoot /var/www/html/rockylinux.dve

*ErrorLog* /var/log/httpd*/rockylinux.dve.error.log*

*CustomLog* /var/log/httpd/*rockylinux.dve.access.log* common

<Directory /var/www/html/rockylinux.dve >

Options FollowSymLinks

AllowOverride All

Require all granted

</Directory>

</VirtualHost>

### Перезапуск apache в Rocky Linux

Проверим созданную конфигурацию и выполним перезапуск apache.

# apachectl -t

В нормальном случае - ошибок нет (читаем «Syntax OK»), допускается предупреждение.

Перезапуск:

# systemctl restart httpd

Если возникли какие-то ошибки — смотрим лог apache /var/log/httpd/error\_log. Если все в порядке, то проверим, нормально ли настроен наш виртуальный хост. Для этого создадим в папке сайта Rocky Linux.dve файл index.html следующего содержания:

# mcedit /var/www/html/index.html

<h1>Удивительно, но Apache <b>работает</b> c новым сайтом!</h1>

# chown -R apache. /var/www/html/

Дальше на своем клиентском компьютере правим файл hosts, чтобы обратиться к нашему виртуальному хосту. Добавляем туда строчку с IP-адресом и URL:

10.4.0.1 rockylinux.dve

Теперь в браузере набираем адрес http://RockyLinux.dve. Если видим свою страничку, значит все правильно настроили. Если какие-то ошибки, то идем смотреть логи. Причем в данном случае не общий лог httpd, а лог ошибок конкретного виртуального хоста.

## Установка php в Rocky Linux 8

Устанавливаем php в Rocky Linux 8, а также некоторые популярные модули, которые могут пригодиться.

# dnf install php php-cli php-mysqlnd php-json php-gd php-ldap php-odbc php-pdo php-opcache php-pear php-xml php-xmlrpc php-mbstring php-snmp php-soap php-zip

Выполним перезапуск apache:

# systemctl restart httpd

Создадим файл в директории виртуального хоста и проверим работу php:

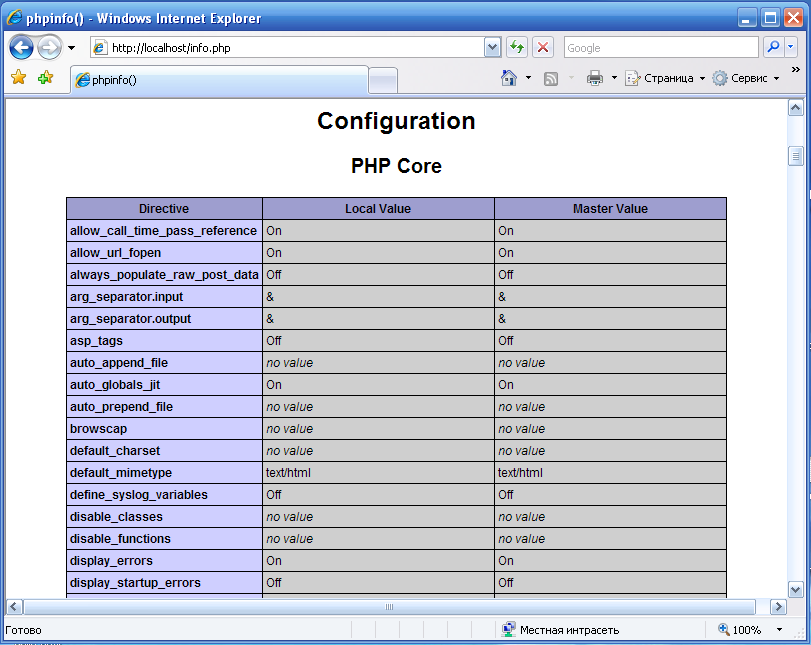
# mcedit /var/www/html/rockylinux.dve/www/index.php

ВСТАВИТЬ КОД НА РНР: <?php phpinfo(); ?>

# chown apache. /var/www/html/rockylinux.dve/www/index.php

Заходим по адресу http://rockylinux.dve/index.php

Вы должны увидеть вывод информации о php.



Если что-то не так, возникли какие-то ошибки, смотрите лог ошибок виртуального хоста, ошибки php будут тоже там.

### Находим php.ini

После установки часто возникает вопрос, а где хранятся настройки php? Традиционно они находятся в едином файле настроек. В Rocky Linux **php.ini лежит в директории с конфигурационными файлами /etc**, прямо в корне.

Давайте добавим туда несколько полезных настроек (не обязательно):

# mcedit /etc/php.ini

Выставим значения:

date.timezone 'Europe/Moscow'

max\_execution\_time 60

upload\_max\_filesize 30M

Для применения настроек нужно сделать перезапуск Апача. Возможно, появится ошибка.

Invalid command 'php\_admin\_value', perhaps misspelled or defined by a module not included in the server configuration

Это не загружен модуль mod\_php. Он подключается в файле /etc/httpd/conf.modules.d/15-php.conf.

<IfModule !mod\_php5.c>

<IfModule prefork.c>

LoadModule php7\_module modules/libphp7.so

</IfModule>

</IfModule>

Тут стоит проверка на запуск модуля. Он загружается **по условию**, если загружен модуль prefork. Попробуем его загрузить принудительно - комментируем все строки, кроме основной

LoadModule php7\_module modules/libphp7.so

Проверяем конфигурацию apache.

# apachectl -t

Apache is running a threaded MPM, but your PHP Module is not compiled to be threadsafe.

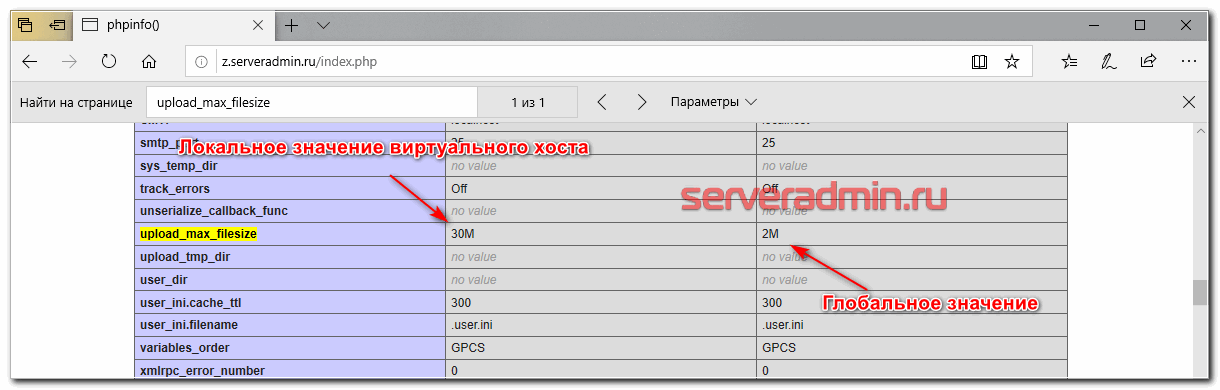
Получили новую ошибку. Смысл в том, что изначально apache сконфигурирован на работу модуля mpm\_event, он подключается в конфиге /etc/httpd/conf.modules.d/00-mpm.conf.

LoadModule mpm\_event\_module modules/mod\_mpm\_event.so

Стандартный модуль mod\_php скомпилирован с поддержкой модуля mpm\_prefork. С другими он работать не будет. Таким образом, чтобы у нас нормально заработал php, нам надо вместо модуля mpm\_event подключить модуль mpm\_prefork. Для этого в файле 00-mpm.conf закомментируем подключение mpm\_event\_module и раскомментируем prefork.

LoadModule mpm\_prefork\_module modules/mod\_mpm\_prefork.so

После этого проверяйте конфигурацию и перезапускайте apache. Все должно заработать. Теперь в выводе phpinfo можно увидеть изменение настроек.



Ошибки разобраны подробно, чтобы стало понятно, как все устроено и куда смотреть в случае проблем.

### Простой сайт на РНР

Развернуть простой сайт на РНР, выводящий случайное изображение, очень просто:

● скачиваем архив с drach.pro

● распаковываем в корневую папку {document root} с именем сайта, предварительно удалив из нее файл-заглушку

● проверяем из браузера клиента

Отслеживать доступ в реальном времени удобно командой tail -f имя\_файла\_журнала !

### Задание.

Установить на Rocky Linux 8 веб-окружение: веб-сервер Apache и интерпретатор РНР.

Сделать страничку для отображения по умолчанию (приветствие).

Настроить сайт с собственным доменным именем (virtual host).

Развернуть простой сайт, выводящий случайное изображение.

# Лабораторная работа №7. Стек LEMP

Цель работы: формирование практических навыков администрирования корпоративных операционных систем.

**Предварительные замечания**

* В качестве серверной операционной системы рекомендуется использовать дистрибутив Rocky Linux
* Работу **можно** выполнять на только что установленной системе, но потребуется создать пользователя и группу apache:apache
* Работу можно выполнять на сервере из ЛР №4, предварительно изменив порт веб-сервера Apache с 80 на любой другой выше 1024.
* Настройка firewalld выполняется как в ЛР №4

**Введение**

Установим стек LEMP, который становится доминирующим в настоящее время.

LEMP это акроним из слов Linux, Nginx (читается как *Engine X*), MySQL и PHP.

В последних версиях Rocky Linux рекомендуемой СУБД является не MySQL, а её ответвление – MariaDB. На боевых серверах можно рекомендовать возврат к MySQL, однако при решении академических задач можно пойти по простому пути и установить MariaDB (к счастью, с точки зрения взаимодействия с ними разница почти не заметна).

**Выполнение**

### Установка Nginx

Установим веб-сервер Nginx, разработанный выпускником МГТУ им. Баумана – Игорем Сысоевым:

dnf install nginx

Добавляем в автозагрузку, запускаем и проверяем статус/версию:

systemctl enable nginx && systemctl start nginx

systemctl status nginx

nginx -v

Исчерпывающая информация может быть получена по команде:

rpm -qi nginx

На этом шаге наш сервер уже способен отображать страницу приветствия по протоколу HTTP.

### Установка СУБД

Можно пойти простым путём и установить MariaDB (однако неоднократно студенты сообщали о **проблемах с последующией авторизацией**). Устанавливаем MariaDB, добавляем в автозагрузку, запускаем и проверяем статус/версию

dnf install mariadb-server mariadb -y

systemctl enable mariadb && systemctl start mariadb

systemctl status mariadb

ИЛИ

Можно пойти иным путём, установить старую и надёжную СУБД MySQL, т.к. экстремальная производительность в академических задачах не важна. Устанавливаем MySQL, добавляем в автозагрузку, запускаем и проверяем статус/версию

dnf install mysqld-server -y

systemctl enable mysqld && systemctl start mysqld

systemctl status mysqld

Вне зависимости от выбода СУБД, дальнейшие действия по администрированию будут идентичными.

Стандартом *de facto* является выполнение скрипта повышения безопасности сразу после установки.

mysql\_secure\_installation

Необходимо ответить на несколько вопросов, а самое главное – ввести пароль для СУБД, который пригодится в будущем.

### Установка РНР

Если интерпретатор РНР ещё не установлен, см. Л/Р №4.

Проверяем версию:

php -v

Задание повышенной сложности для отличников: обновить версию PHP c 7.4 до 8.x (подсказки ищем в интернете).

Особенностью стека LEMP является то, что для взаимодействия nginx и PHP требуется дополнительный модуль: php-fpm - Менеджер процессов FastCGI (FPM).

Устанавливаем его, и, что важно, запускаем – это самостоятельный **процесс**!

systemctl enable php-fpm && systemctl start php-fpm

systemctl status php-fpm

Дополнительно устанавливаем модули, которые нам пригодятся в дальнейшем для функционирования полноценных сайтов:

dnf install php-dom php-bcmath php-simplexml php-xml php-xmlreader php-curl php-date php-exif php-filter php-ftp php-gd php-hash php-iconv php-json php-libxml php-mbstring php-mysqlnd php-openssl php-pcre php-posix php-sockets php-spl php-tokenizer php-zlib mod\_php -y

#### Конфигурация

Выполняем конфигурацию всех компонентов. Примеры рабочих файлов конфигурации с боевых серверов полностью рабочие, но поменяйте под себя названия сайтов.

Основной файл конфигурации nginx - вставляем содержимое без изменений:

/etc/nginx/nginx.conf

user apache;

# worker\_processes auto;

worker\_processes 3;

error\_log /var/log/nginx/error.log warn;

pid /var/run/nginx.pid;

events { worker\_connections 1024; }

http {

include /etc/nginx/mime.types;

default\_type application/octet-stream;

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

sendfile on;

client\_max\_body\_size 90M;

# Отключим показ версии сервера на всех ошибочных страницах

server\_tokens off;

# Таймауты

client\_body\_timeout 10;

client\_header\_timeout 10;

keepalive\_timeout 5 5;

send\_timeout 10;

# Компрессия:

gzip on;

gzip\_static on;

gzip\_vary on;

# Минимальная длина ответа для компрессии, в байтах

gzip\_min\_length 1100;

# Разрешить сжатие для всех проксированных запросов

gzip\_proxied any;

# Уровень gzip-компрессии

gzip\_comp\_level 5;

gzip\_types text/plain text/xml application/xml application/x-javascript application/javascript text/javascript text/css text/json;

include /etc/nginx/conf.d/\*.conf;

}

Файл конфигурации конкретного сайта:

/etc/nginx/conf.d/имя\_сайта.conf

# php + fpm, работаем через unix-socket

upstream backend-land {server unix:/run/php-fpm/fpm-имя\_сайта.sock;}

server {

server\_name www.имя\_сайта;

return 301 $scheme://имя\_сайта$request\_uri;

}

server {

listen 80;

server\_name имя\_сайта;

root /var/www/html/имя\_сайта;

# access\_log /var/log/nginx/имя\_сайта.access.log;

access\_log off;

error\_log /var/log/nginx/имя\_сайта.error.log;

index index.php index.html;

rewrite\_log on;

location / {

try\_files $uri $uri/ @rewrite;

}

location @rewrite {

rewrite ^/(.\*)$ /index.php?q=$uri&$args;

}

location ~ \.php$ {

# Избавляемся от /index.php (done!)

if ($request\_uri ~ ^/index.(html|php)) {

rewrite ^.\*$ / permanent;

}

include fastcgi\_params;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_pass backend-land;

}

# Не ведём журнал скачивания вспомогательных файлов отображения страницы,

# отдаём их напрямую с диска

location ~\* ^.+\.(jpg|gif|css|png|js|ico|bmp|svg|woff|woff2|ttf|txt)$ {

access\_log off;

expires 90d;

break;

}

# Ведем журнал скачивания больших файлов

# (это файлы документов)

# и кэшируем их на 25 дней

location ~\* ^.+\.(mp3|ogg|mpe?g|avi|zip|gz|bz2?|rar|tbz|pdf)$ {

access\_log /var/log/nginx/имя\_сайта.access.log;

# access\_log off;

expires 25d;

break;

}

location ~ /\.ht {

deny all;

}

}

# EOF

Основной файл (кажется) не надо менять. Только проверяем, что пользователь и группа указаны верно.

/etc/php-fpm.d/www.conf

; RPM: apache user chosen to provide access to the same directories as httpd

user = apache

; RPM: Keep a group allowed to write in log dir.

group = apache

/etc/php-fpm.d/имя\_сайта.conf

[имя\_сайта]

listen = /run/php-fpm/fpm-имя\_сайта.sock

listen.allowed\_clients = 127.0.0.1

listen.owner = apache

listen.group = apache

user = apache

group = apache

; как будут создаваться новые рабочие процессы?

pm = dynamic

; максимальное количество рабочих процессов

pm.max\_children = 6

; число запущенных процессов при старте сервера

pm.start\_servers = 2

; минимальное и максимальное количество процессов в простое

pm.min\_spare\_servers = 2

pm.max\_spare\_servers = 4

; сколько запросов может обработать один процесс перед самоубийством?

pm.max\_requests = 1000

slowlog = /var/log/php-fpm/имя\_сайта.slow.log

php\_admin\_value[error\_log] = /var/log/php-fpm/имя\_сайта.error.log

php\_admin\_flag[log\_errors] = on

php\_value[session.save\_handler] = files

php\_value[session.save\_path] = /var/lib/php/session

Возможно, придётся внести изменения ещё в два файла:

/etc/php.ini

/etc/my.cnf (.conf? .ini?) – на боевом сервере следует настраивать, оптимизируя под конкретную задачу, при решении академических задач оставляем по умолчанию.

После внесения изменений в конфигурационные файлы необходимо их перечитать. Простой (колхозный) путь – это выполнить перезапуск веб-сервера и менеджера:

systemctl restart nginx && systemctl restart php-fpm

Профессиональный подход подразумевает, что веб-сервер уже работает и обслуживает реальный сайт, на котором может быть большое число посетителе, поэтому выполняется проверка на ошибки конфигурации, а затем мягкая перезагрузка:

nginx -t

service nginx reload

systemctl restart php-fpm

### Проверка работоспособности

Размещение любого тестового сайта (например, из ЛР №4 или иной) в его директории приведет к тому, что он станет доступен на **клиентской** машине по http.

На **сервере** выполняем

top u apache

## Для выхода нажимаем q.

## Задание

Установить стек LEMP; убедиться, что тестовый сайт доступен для клиента.

# Лабораторная работа №8. Установка CMS

Цель: сформировать практические навыки по установке системы управления контентом.

## Предварительные замечания

Предполагается, что на сервере уже выполнена Лабораторная работа №4.

## Теоретическая часть

Joomla это **бесплатная** система управления контентом (CMS — *Content management system*), т.е. с помощью Joomla можно создавать сайты, причём совершенно бесплатно. На Joomla можно легко разработать сайт любой сложности это достигается за счет расширяемости, т.е. у Joomla много дополнительных компонентов, модулей и плагинов. Написана Joomla на языке web программирования PHP и в качестве хранилища базы данных использует СУБД MySQL. Наверное, любой начинающий web разработчик, когда хочет впервые создать свой собственный сайт, в качестве инструмента рассматривает именно Joomla, другими словами, данная CMS очень популярна и является, наверное, самой популярной среди бесплатных систем управления контентом.

Изначально Joomla ориентирована на стек LAMP, но прекрасно функционирует под LEMP с небольшими доработками.

## Ход работы

Из раздела Ресурсы (в конце методических указаний) скачать установщик Joomla по выбору - demo\_package. Данный установщик интересен тем, что имеет встроенные:

1) шаблон – это дизайн будущего сайта

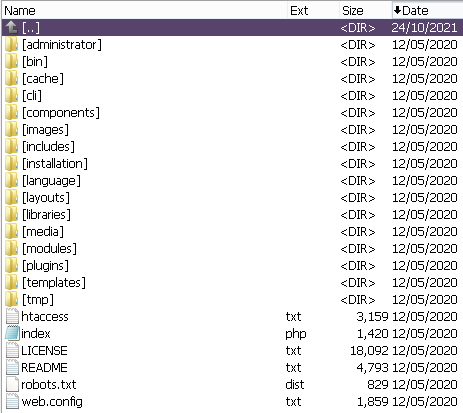
2) расширение Pro – это конструктор страниц с мощным функционалом, который позволяет создавать страницы без знания HTML\CSS.

Если хочется выбрать дизайн заранее, ищем в интернете демонстрацию, например, для **craft**\_demo\_package:

Переходим на сайт https://demo.**yootheme**.com/joomla/themes/**craft**

Для других шаблонов надо будет заменить последнее слово на имя шаблона.

Внутри установщика должна быть подобная структура папок и файлов:



На сервере необходимо создать новый поддомен (по аналогии с предыдущей ЛР): создать домашнюю папку нового сайта, создать и отредактировать файлы конфигурации для nginx и php-fpm, перезапустить их.

Домашняя папка: /var/www/html/имя\_сайта

Конфигурационные файлы:

/etc/nginx/conf.d/имя\_сайта.conf

/etc/php-fpm.d/имя\_сайта.conf

Естественно, имя\_сайта надо будет использовать своё как в названии файлов, так и в их тексте.

В домашнюю папку загружаем установщик. Это можно сделать двумя путями: скопировать с клиентской машины по протоколу SSH или же зайти на сервер, перейти в домашнюю папку и загрузить из Интернета напрямую, выполнив команду

wget адрес\_установщика\_в\_Интеренете

Если не делали этого ранее, устанавливаем wget:

yum install wget -y

Устанавливаем распаковщик:

yum install unzip -y

Распаковываем:

unzip \_имя\_установщика\_

Воспользуйтесь TAB, чтобы операционная система предложила варианты имени файла.

Задаём права:

chown -R apache:apache /var/www/html/имя\_сайта

Добавляем информацию о новом поддомене в файл /etc/hosts на клиентской машине.

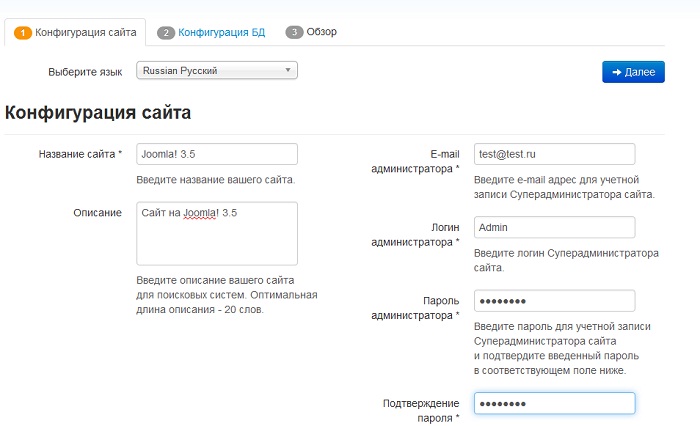
Теперь из браузера будет доступен новый сайт. Открываем его и устанавливаем Joomla в графическом режиме из браузера.

## Установка Joomla

Процесс установки Joomla 3.5 очень прост, поэтому подробно рассматривать мы его не будем, просто следуйте подсказкам на экране.

### Шаг 1

Заполняем все поля по своему усмотрению. Придумываем пароль для сайта.



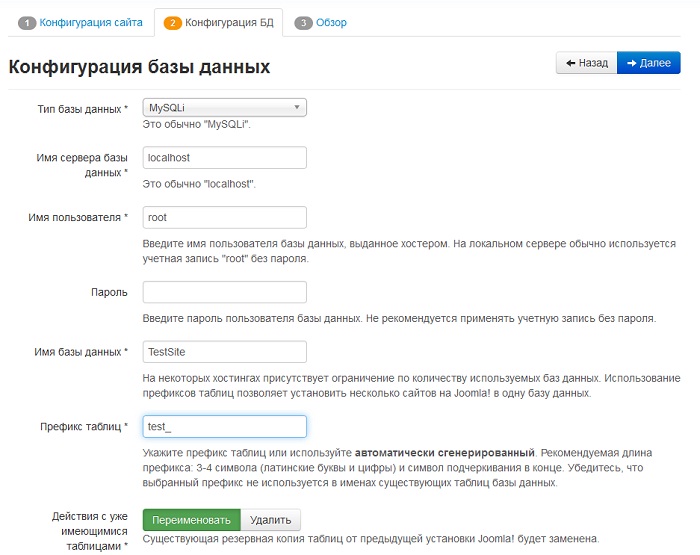
### Шаг 2

Тип БД выбираем: MySQLi

Имя сервера БД: localhost

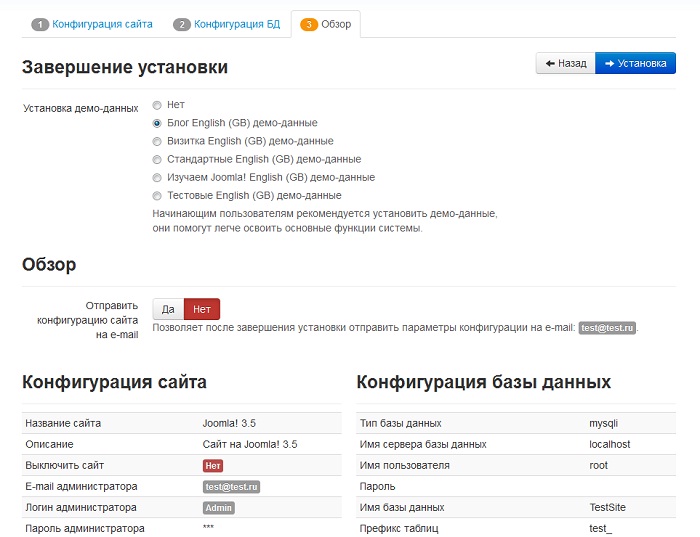
Имя пользователя и пароль: ввести свои значения для СУБД.

Имя базы данных и префикс - придумываем.



### Шаг 3

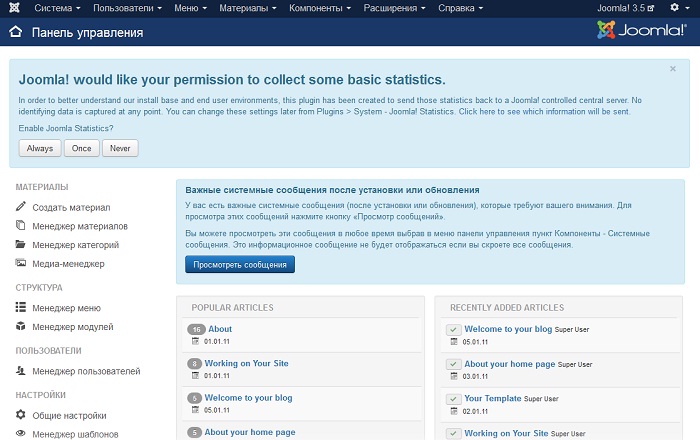
Важно поставить отметку: устанавливать тестовые демо-данные (для надстройки PRO).



### Шаг 4

Важно удалить папку installation (это можно сделать прямо из браузера, нажав на кнопку) и запомнить две ссылки: на сайт и на Панель управления.

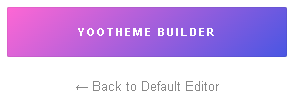
**Панель управления Joomla 3.5**



Сайт готов к работе!

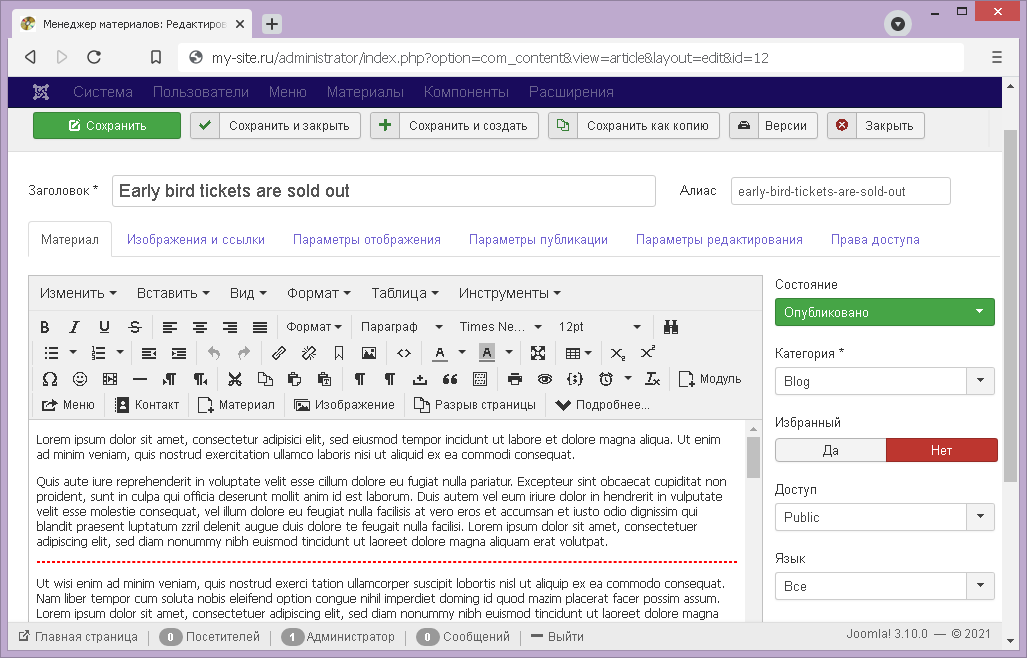
В меню Материалы \ Менеджер материалов выбираем существующую страницу, которая создана в **Конструкторе** и редактируем по своему усмотрению.

Признак того, что страница создана в конструкторе:



Видим фиолетовую кнопку YooTheme Builder в поле для редактирования страницы.

Обычная страница в режиме редактирования выглядит таким образом:



## Задание

Установить полноценный стек LEMP, убедиться, что тестовый сайт под управлением Joomla доступен для машины клиента. Выполнить редактирование страницы в конструкторе.

# Лабораторная работа №9. Установка phpMyAdmin

Цель: сформировать практические навыки по установке системы управления БД phpMyAdmin. Внести изменения в БД напрямую.

## Предварительные замечания

Предполагается, что на сервере уже выполнена Лабораторная работа №4 и №5.

## Теоретическая часть

phpMyAdmin — это программный инструмент с открытым исходным кодом, написанный на PHP для администрирования серверов баз данных MySQL и MariaDB из веб-интерфейса.

phpMyAdmin поддерживает широкий спектр операций с MySQL, MariaDB и Drizzle. С помощью этого инструмента вы можете управлять базами данных, таблицами, столбцами, отношениями, индексами, пользователями, разрешениями и другими с помощью интуитивно понятного и простого в использовании веб-интерфейса.

## Ход работы

Ниже приведены шаги для установки и настройки phpMyAdmin на Rocky Linux 8. Требуемые зависимости — это PHP, nginx, веб-сервер Apache и сервер базы данных, которым нужно управлять.

Шаг 1: Установите расширение PHP

Убедитесь, что php-mysqlnd расширение установлено.

yum -y install php-mysqlnd

Шаг 2: Установите сервер базы данных MariaDB / MySQL

Убедитесь, что выполнена установка СУБД.

mysql -V

Шаг 3: Установите веб-сервер

phpMyAdmin поддерживает как Apache, так и Nginx в качестве веб-сервера.

Проверяем версию веб-сервера

nginx -v

Шаг 4: Установите phpMyAdmin

Посетите страницу загрузки phpMyAdmin и проверьте последний доступный пакет.

https://www.phpmyadmin.net/downloads/

На момент написания методички последняя версия 5.1.1

https://files.phpmyadmin.net/phpMyAdmin/5.1.1/phpMyAdmin-5.1.1-all-languages.zip

Скачиваем на сервер и распаковываем в домашней директории.

wget https://files.phpmyadmin.net/phpMyAdmin/5.1.1/phpMyAdmin-5.1.1-all-languages.zip

unzip phpMyAdmin-5.1.1-all-languages.zip

Создайте файл конфигурации phpMyAdmin: копируем файл в директории /phpmyadmin/

cp config.sample.inc.php config.inc.php

Редактировать файл

config.inc.php

Установите секретную фразу-пароль — должен быть длиной 32 символа

$cfg['blowfish\_secret'] = 'H2OxcGXxflSd8JwrwVlh6KW6s2rER63i';

Настроить временный каталог (подставить свой путь при необходимости):

$cfg['TempDir'] = '/var/www/html/phpmyadmin/tmp';

Шаг 5: Настройте веб-сервер

Вариантов два.

а) простой - скопировать всю файловую систему phpmyadmin отдельной директорией в работающий сайт из предыдущей ЛР

б) профессиональный - создать новый поддомен (см. предыдущую работу), скопировать всю файловую систему phpmyadmin в собственную корневую директорию

Перезапустите службы nginx и php-fpm для загрузки новой конфигурации,

systemctl restart nginx

systemctl restart php-fpm

Шаг 5. Настройка брандмауэра

Разрешить http порт в брандмауэре.

firewall-cmd --add-service=http --permanent

Перезагрузите конфигурацию перезагрузки брандмауэра.

firewall-cmd --reload

Шаг 6: Доступ к веб-интерфейсу phpMyAdmin

Зависит от варианта, который был выбран раньше (а) или (б)

Для (а) откройте в браузере URL http://[ServerIP|Hostname]/phpmyadmin

Дла (б) откройте в браузере http://имя\_сайта\_phpmyadmin

Войдите в панель управления phpMyAdmin, используя свои учетные данные для СУБД — имя пользователя и пароль.

**Прикладная задача**

С помощью phpMyAdmin взломаем сайт на Joomla из предыдущей работы.

Не пользуясь админкой сайта, создадим нового пользователя, который получит полный доступ к управлению сайтом.

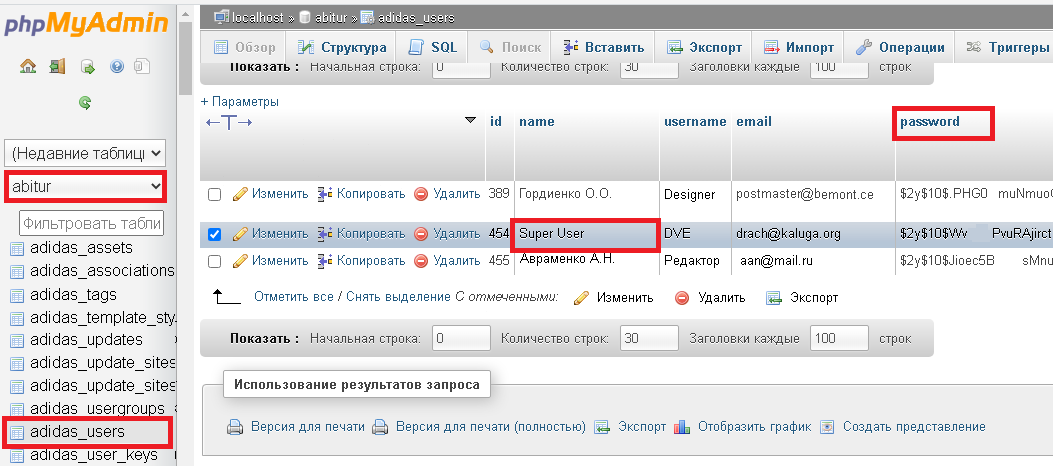
Идея: получить доступ к БД сайта, скопировать учётную запись администратора и подменить имя и пароль на **известные нам**.

Шаг 1.

Зайти в phpMyAdmin, выбрать БД, соответствующую сайту.

Выбрать таблицу, которая называется префикс\_users.

Определить пользователя, у которого максимальные права, используя солдатскую смекалку. В примере ниже у сайта три пользователя, но только один именуется Super User.



Обратите внимание, пароль пользователя хранится в **зашифрованном виде**!

Копируем строчку пользователя с правами администратора.

Шаг 2.

Придумываем имя для нового пользователя и исправляем его в скопированной строке.

Придумываем простой пароль и находим его хэш по алгоритму Bcrypt с помощью какого-то сервиса, например

https://snipp.ru/tools/htpasswd

Важно выбрать именно Bcrypt (хотя иногда срабатывает MD5)!

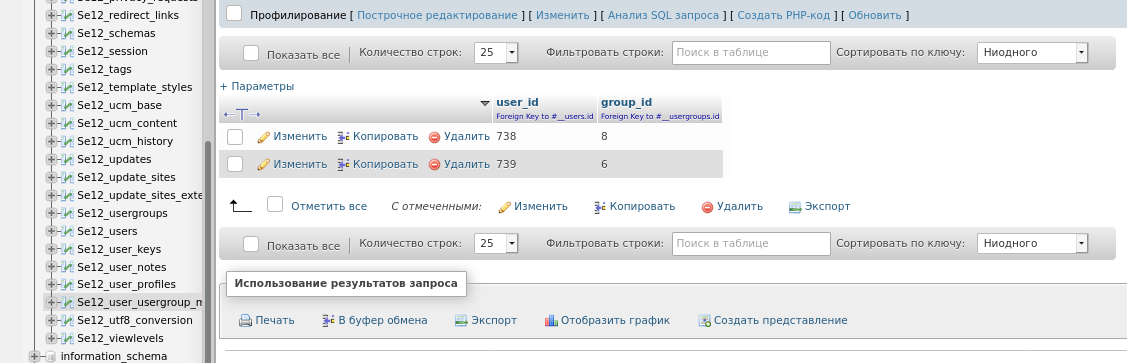
Альтернативы:

* https://sysopnotes.net/ru/generator-xeshej-dlya-parolej-wordpress-i-joomla/ (отличный генератор с пояснениями!)
* https://www.danielmorell.com/tools/joomla/password-generator
* https://www.useotools.com/ru/joomla-password-hash-generator

Исправляем **хэш** пароля ново-созданного пользователя на полученный хэш, который сгенерировал сервис.

Шаг 3.

Переходим в префикс\_user\_usergroup\_map, где нужно скопировать user\_id (в данном случае 738 так как это ID учётной записи администратора). Записать id новой учётной записи и сохранить.



Входим на сайт с новым именем и паролем.

Наслаждайтесь администрированием!

## Задание

Установить phpMyAdmin, убедиться в работоспособности. Выполнить взлом сайта через phpMyAdmin, не используя учётные данные администратора, которые были введены при установке.

# Лабораторная работа №10. Установка и настройка Fail2ban

**Предисловие**

Несмотря на то, что соединение по протоколу SSH является достаточно безопасным вариантом удалённого взаимодействия с сервером, сам факт подключения к интернету делает сервер уязвимым для различных кибератак, включая так называемый *брутфорс*. В такой ситуации пользователь или бот нагружает ресурс множественными попытками входа в систему до тех пор, пока к ней не будет получен окончательный доступ.

Утилита Fail2ban даёт администратору возможность справляться с этими атаками путём создания определённых правил брандмауэра, согласно которым IP-адрес, пытавшийся многократно и безуспешно получить доступ к серверу, блокируется после определённого количества неудачных попыток авторизации.

Данная лабораторная работа посвящена тому, как установить и настроить Fail2ban для защиты SSH и повышения безопасности сервера на Rocky Linux от атак методом перебора паролей.

**1. Установка EPEL и Fail2ban**

Утилита Fail2ban отсутствует в официальных репозиториях, однако доступна в репозитории EPEL. Установить его можно при помощи команды:

dnf install epel-release

После этого необходимо загрузить и установить сам пакет Fail2ban, выполнив команду:

dnf install fail2ban

**2. Настройка Fail2ban**

Конфигурационные файлы Fail2ban расположены в директории /etc/fail2ban. В ней находится глобальный файл конфигурации jail.conf. Однако для пользовательской настройки крайне не рекомендуется изменять его, так как он обновляется при загрузке новой версии пакета. Вместо этого в той же директории необходимо создать файл jail.local, параметры которого переопределяют основной конфигурационный файл. Помимо этого, в директории /etc/fail2ban/jail.d хранятся файлы, определяющие настройки Fail2ban для конкретных утилит. Например, для фоновой службы SSH. Они также перезаписывают параметры jail.conf.

Каждый файл конфигурации может содержать раздел с заголовком [DEFAULT] для описания общих параметров, а также разделы с необходимыми настройками для отдельных утилит.

Для создания файла jail.local воспользуйтесь текстовым редактором mcedit:

mcedit /etc/fail2ban/jail.local

После открытия файла введите в него следующие параметры и их значения:

[DEFAULT]

maxretry = 3

findtime = 60

bantime = 600

banaction = firewallcmd-ipset

[sshd]

enabled = true

Пояснение параметров:

* [DEFAULT] и [sshd] — заголовки разделов;
* maxretry — максимально возможное количество попыток ввести корректный пароль;
* findtime — время в секундах, в течение которого пользователь должен исчерпать количество попыток, чтобы оказаться заблокированным;
* bantime — время в секундах, в течение которого IP-адрес пользователя будет заблокирован;
* banaction — действие, которое будет выполняться при обнаружении активности, подпадающей под указанные параметры (в данном случае — внесение IP-адреса пользователя в список брандмауэра);
* enabled — активация Fail2ban для службы SSH при значении true.

Таким образом, приведённая выше конфигурация означает, что любой пользователь будет заблокирован на 10 минут, если он в течение минуты проведёт три неудачные попытки авторизации через протокол SSH.

Внесите дополнительные настройки в файл конфигурации sshd.local для  фоновой службы SSH, который создаётся в директории /etc/fail2ban/jail.d с помощью команды:

mcedit /etc/fail2ban/jail.d/sshd.local

После открытия файла введите в него следующие параметры и их значения:

[sshd]

port = ssh

filter = sshd

Пояснение параметров:

* port — переменная, содержащая буквенное или цифровое значение порта сервиса, который будет заблокирован для пользователя;
* filter — набор правил, по которому работает утилита, то есть в данном случае отбираются неудачные аутентификации (готовые фильтры для различных сервисов расположены в директории /etc/fail2ban/filter.d).

**3. Запуск Fail2ban**

Для запуска и активации утилиты Fail2ban выполните следующие команды:

systemctl start fail2ban

systemctl enable fail2ban

Для проверки работоспособности сервиса введите команду:

systemctl status fail2ban

Если всё настроено верно, поле Active будет содержать значение active (running).

**4. Мониторинг и управление заблокированными IP-адресами**

Для контроля за заблокированными адресами и работой утилиты используется команда fail2ban-client. Для просмотра текущего состояния Fail2ban выполните команду:

fail2ban-client status

В результате будет отображено количество и список активных *«тюрем»* для каждого сервиса. Для получения более подробной информации о работе Fail2ban с конкретной службой (в данном случае — с SSH) выполните команду:

fail2ban-client status sshd

Теперь можно увидеть общее количество заблокированных адресов, количество адресов, заблокированных в текущий момент, и их список.

Для снятия блокировки с IP-адреса во всех *«тюрьмах»* можно воспользоваться следующей командой:

fail2ban-client unban <ip>

Чтобы разблокировать адрес для доступа к определённому сервису используйте команду:

fail2ban-client set <jail> unbanip <ip>

Для получения дополнительной информации о Fail2ban введите команды:

man jail.conf

man fail2ban-client

Все логи Fail2ban находятся в файле по адресу /var/log/fail2ban.log.

**5. Важное замечание**

При использовании небезызвестного ПО для виртуализации под названием Oracle VM VirtualBox может возникнуть внештатная ситуация.

SSH за время исполнения команды даёт пользователю три гарантированные попытки на вход, даже если в конфигурации Fail2ban явно указать меньшее их количество. Поэтому для того, чтобы убедиться в корректности работы Fail2ban, необходимо насильно останавливать команду ssh путём сочетания клавиш Ctrl+C после неудачной попытки входа и заново запускать её. Только в таком случае при исчерпании количества попыток, определённого параметром maxretry, утилита выдаст необходимое сообщение Connection refused.

Проблема наиболее вероятна в случаях, когда maxretry некратен трём, из-за чего у пользователя появляются лишние, непредусмотренные попытки угадать пароль.

**Задание**

1. Защитите сервер от атак по SSH с помощью утилиты Fail2ban. Продемонстрируйте отказ в обслуживании.
2. *Задание повышенной сложности для лютых отличников:* Защитите сайт из предыдущих лабораторных работ от взлома методом перебора паролей с помощью Fail2ban. Продемонстрируйте отказ в обслуживании.

# Лабораторная работа №11. Сторожевой бот для Telegram

Задача: запустить бота-информатора, который будет сообщать в Телеграм об успешных входах на сервер по SSH.

Методика:

https://drach.pro/blog/linux/item/212-ssh-login-alert-telegram

# Лабораторная работа №12. Бот для мониторинга

Задача: запустить бота, сообщающего о состоянии сервера, загрузить сервер и получить сообщение ALERT.

Методика:

https://drach.pro/blog/linux/item/219-server-monitoring-telegram-bot

# Лабораторная работа №13\*. Автозапуск скрипта в контейнере

Напишем простой диалоговый Telegram-бот на Python, который в дальнейшем можно дополнить различными функциями, и настроим автозапуск.

## Настройка

Откройте Telegram, найдите @BotFather и начните беседу. Отправьте команду /newbot и следуйте инструкциям. Вы получите:

* свой токен;
* адрес Telegram API (https://api.telegram.org/bot);
* ссылку на документацию.

Обязательно сохраните токен, так как это ключ для взаимодействия с ботом.

Примечание: Хранение токена должно быть локальным; ни в коем случае не выгружайте его в общий доступ, например в GitHub-репозиторий .

Далее начните беседу. Введите в поисковой строке имя бота и нажмите /start. Отправьте любое сообщение: оно станет первым обновлением, которое получит Телеграм бот.

### Установка Python

Для написания Telegram-бота на Python, нужно установить сам язык. Версия важна, подойдет Python не ниже версии 3.7.

### Установка pip

Это менеджер пакетов. В версиях выше Python 2.7.9 и Python 3.4, а также на macOS/Linux он уже есть. Проверить это можно командой pip --version в терминале. Если же по каким-то причинам он отсутствует, установить его можно при помощи команды:

sudo apt install python3.9

### Установка aiogram

Установить данный фреймворк для Telegram Bot API с помощью pip:

pip install aiogram

python3.9 -m pip install aiogram==2.23.1

## Hello, bot!

Давайте напишем простенькую программу приветствия. Для начала следует импортировать библиотеки и создать экземпляры Телеграм бота и диспетчера:

from aiogram import Bot, types

from aiogram.dispatcher import Dispatcher

from aiogram.utils import executor

TOKEN = "ваш токен от бота здесь"

bot = Bot(token=TOKEN)

dp = Dispatcher(bot)

Теперь напишем обработчик текстовых сообщений, который будет обрабатывать входящие команды /start и /help:

@dp.message\_handler(commands=['start', 'help'])

async def send\_welcome(msg: types.Message):

await msg.reply\_to\_message(f‘Я бот. Приятно познакомиться,

{msg.from\_user.first\_name}’)

Добавим ещё один обработчик для получения текстовых сообщений. Если бот получит «Привет», он также поздоровается. Все остальные сообщения будут определены, как нераспознанные:

@dp.message\_handler(content\_types=['text'])

async def get\_text\_messages(msg: types.Message):

if msg.text.lower() == 'привет':

await msg.answer('Привет!')

else:

await msg.answer('Не понимаю, что это значит.')

Запускаем Telegram бота, написанного на Python, следующим образом:

if \_\_name\_\_ == '\_\_main\_\_':

executor.start\_polling(dp)

Примечание Так мы задаём боту непрерывное отслеживание новых сообщений. Если бот упадёт, а сообщения продолжат поступать, они будут накапливаться в течение 24 часов на серверах Telegram, и в случае восстановления бота прилетят ему все сразу.

Ну вот и всё, простенький бот в Телеграмме на языке Python готов.

## Docker

Для нашего проекта потребуется самый простой Dockerfile:

FROM python:3.9

# set work directory

WORKDIR /usr/src/app/

# copy project

COPY . /usr/src/app/

# install dependencies

RUN python3.9 -m pip install aiogram==2.23.1

# run app

CMD ["python", "bot.py"]

Директория проекта должна при этом содержать следующие файлы:

* bot.py;
* Dockerfile.

Для локальных тестов достаточно установить Docker, после чего в папке проекта собрать и запустить контейнер с помощью команд:

docker build -t my\_app .

docker run -d my\_app

my\_app — это название нашего контейнера, вместо которого можно и нужно использовать другое имя.

-d — специальный флаг, который запускает контейнер в фоне и позволяет дальше работать в терминале. Это называется **detached mode**.

## Загрузка на Docker Hub

Это задание повышенной сложности, следует выполнять строго по собственному желанию.

Прежде всего нам понадобится аккаунт на [Docker Hub](https://hub.docker.com/search?q=&type=image). Это аналог GitHub, только не с исходниками кода, а с уже созданными контейнерами. Работа с Docker Hub выглядит достаточно просто:

1. Локально или с помощью пайплайнов собрали контейнер.
2. Загрузили его на Docker Hub.
3. В любом удобном месте скачали его. Это может быть локальная машина, собственный VPS или облачный провайдер по типу AWS.
4. Запустили.

Пройдёмся по этим шагам. Везде, где указано <docker\_hub\_username>, надо вставлять свой псевдоним, использованный при регистрации на докерхабе. Если это ваша первая публикация на докерхаб, для начала потребуется войти с помощью docker login.

Билдим контейнер:

docker build -t <docker\_hub\_username>/my\_app

Загружаем его на докерхаб:

docker push <docker\_hub\_username>/my\_app

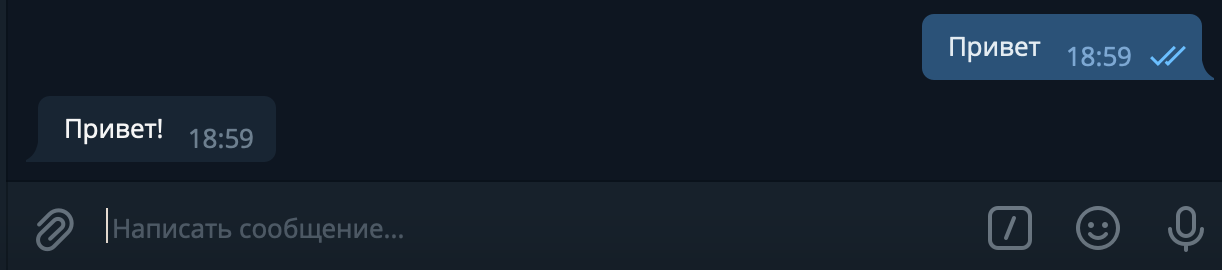
Для проверки успешности загрузки можете запустить контейнер из Docker Hub с помощью команды:

docker run -d <docker\_hub\_username>/my\_app

Далее загрузим наш контейнер на другой сервер: VPS, VDS, AWS...

docker pull -d <docker\_hub\_username>/my\_app

Проверяем работу нашего Telegram bot:



Это успех!

## Заключение

Теперь вы знаете, как писать роботов для Telegram на Python и запускать его в контейнере.

Бота можно дополнять другими функциями, например, добавить отправку файлов, опрос или клавиатуру.

Кстати, в Телеграм есть аж целых два типа клавиатур:

1. Классическая RelpyKeyboardMarkup, у которой кнопки располагаются под полем ввода сообщения:
2. Более современная InlineKeyboardMarkup, которая привязывается к конкретному сообщению:

Но и это полностью рабочий Телеграм-бот на Python: дополните словарём, и получите полноценную беседу. В «настоящих проектах» не обойтись без базы данных. Тут на помощь приходит docker-compose, который позволяет объединить несколько контейнеров в один сервис. Таким образом, например, можно создать приложение и положить его в контейнер, а СУБД, как отдельный сервис, поместить в другой контейнер, и с помощью docker-compose наладить между ними связь.

Отметим, что для более серьёзной разработки лучше использовать выделенный виртуальный сервер (VPS): он даёт гораздо больше гибкости и свободы,

# Лабораторная работа №14. Автозапуск скрипта как службы

Задача: запустить бота из предыдущей лабораторной работы в качестве службы ОС Linux.

Ход выполнения

1. В бота добавить задержку на 1 минуту после выполнения всех действий.
2. Отредактировать (создать) файл

mcedit /etc/systemd/system/sochbot.service

[Unit]

Description = Sochnyi Bot

After = network.target

[Service]

ExecStart = python /root/sochbot/sochbot.py

Restart=always

[Install]

WantedBy = multi-user.target

3) Запустить новую службу и проверить статус.

systemctl daemon-reload

systemctl status sochbot.service

systemctl enable sochbot.service

systemctl start sochbot.service

systemctl status sochbot.service

# Лабораторная работа №15. Система мониторинга в контейнере.

В данной работе научимся запускать микросервис в изолированном контейнере. Ключевой особенностью является то, контейнер будет использоваться уже полностью готовый.

Подключим фирменный репозиторий и установим менеджер контейнеров:

yum config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

yum install docker-ce docker-ce-cli containerd.io -y

Запускаем, добавляем в автозагрузку, проверяем

systemctl start docker

sudo systemctl enable docker

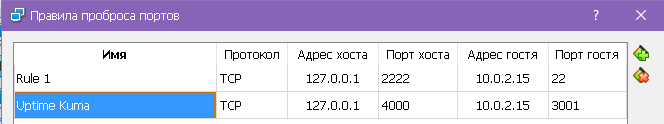
sudo systemctl status docker

Воспользуемся уже готовым контейнером, который опубликован в свободном доступе https://github.com/louislam/uptime-kuma

Для этого выполним простую команду:

docker run -d --restart=always -p 3001:3001 -v uptime-kuma:/app/data --name uptime-kuma louislam/uptime-kuma:1

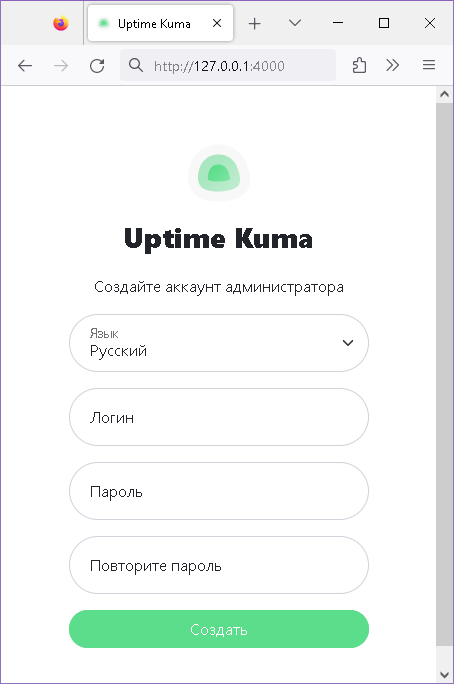
Готово, система мониторинга повисла на порту 3001. Делаем проброс портов, чтобы из браузера на локальном физическом компьютере получить доступ.



На всякий пожарный выключаем межсетевой экран:

service firewalld stop

В браузере должно быть видно:



Задача: запустить систему мониторинга Uptime Kuma, используя Docker.

Добавить в систему мониторинга несколько ресурсов по выбору, чтобы отслеживать их доступность.

# Лабораторная работа №X\*. Нагрузочное тестирование веб-сервера

**Цель:** С помощью систем нагрузочного тестирования определить производительность web-серверов Apache и Nginx, добиться отказа в обслуживании. Сравнить результаты.

B СГУ нагрузочное тестирование работает совсем грустно, если что-то не получится – увы – это нормально.

**Термины и определения**

**Нагрузочное тестирование** – это автоматизированный процесс, имитирующий одновременную работу определенного количества пользователей на каком-либо общем ресурсе.

**Приложение** - тестируемое прикладное программное обеспечение.

**Виртуальный пользователь** - программный процесс, который циклически выполняет моделируемые операции.

**Итерация** - один повтор в цикле операции.

**Интенсивность выполнения операции** - частота выполнения операций в единицу времени, в тестовых скриптах задается интервалом времени между итерациями.

**Нагрузка** - совокупное количество попыток выполнить операции на общем ресурсе. Создается или пользовательской (клиентской) активностью или нагрузочными скриптами.

**Производительность** - количество выполняемых приложением операций в единицу времени.

**Масштабируемость приложения** - пропорциональный рост производительности при увеличении нагрузки.

**Профилем нагрузки** называется набор операций с заданными интенсивностями, полученными на основе статистики.

**Нагрузочной точкой** называется рассчитанное (либо заданное Заказчиком) количество виртуальных пользователей в группах, выполняющих операции с определенными интенсивностями.

**Тест производительности, бенчмарк (англ. benchmark)** — контрольная задача, необходимая для определения сравнительных характеристик производительности компьютерной системы.

Успешное прохождение ряда тестов является свидетельством стабильности системы в штатном и в разогнанном режимах.

**Цели нагрузочного тестирования**

1. Оценка работоспособности и производительности приложения на этапе разработки и при передаче в эксплуатацию
2. Оценка работоспособности и производительности приложения на этапе выпуска новых релизов, патч-сетов
3. Оптимизация производительности приложения, включая оптимизацию программного кода и настройку серверов
4. Подбор соответствующей для данного приложения аппаратной и программной платформы, а также нужной конфигурации сервера

**Виды нагрузочного теста**

**Нагрузочный (Load-testing)** – определяет работоспособность системы при некотором строго заданном уровне нагрузки (планируемой, рабочей).

**Устойчивости (Stress)** – используется для проверки параметров системы в экстремальных условиях и условиях сверхнагрузки, основная задача во время испытания - нарушить нормальную работу системы. Позволяет определить минимальные системные требования для работы приложения, оценить предельные возможности системы и факторы, которые ограничивают эти возможности. В рамках теста также определяется возможность системы сохранить целостность данных при возникновении внештатных аварийных ситуаций.

**Производительности (Performance)** – комплексный тест, включает в себя предыдущие два режима тестирования и предназначен для общей оценки всех показателей системы.

**Результат теста** – представляет собой максимально возможное число пользователей, которые могут единовременно получить доступ к веб-ресурсу, число запросов, которое в состоянии обработать приложение, или время ответа сервера. На основе этой информации, веб-мастер и сетевой администратор смогут заранее выявить слабые места, возникающие из-за несбалансированной работы компонентов (базы данных, маршрутизаторы, кэширующий и прокси-сервер, брандмауэры и др.), и исправить ситуацию, прежде чем система будет запущена в рабочем режиме.

**Использование Apache benchmark tool**

**Apache benchmark** — одна из самых простых утилит, которая применяется для нагрузочного тестирования сайта. Идет в комплекте с веб сервером Apache, в первоначальной настройке не нуждается. Задача, которая ставится перед Apache benchmark — показать, какое количество запросов сможет выдержать веб сервер и как быстро он их обработает.

Пример нагрузки на сервер в 5000 последовательных запросов:

<ab -n 5000 <http://192.168.1.116/index.html>>

Пример нагрузки на сервер в 5000 запросов, но 500 из них будут направлены на сервер одновременно (параллельные запросы):

<ab -n 5000 -c 500 <http://192.168.1.116/index.html>>

**Примечание**

Для выполнения лабораторной работы меняйте значения после -n и -c, чтобы узнать с каким количеством запросов может справиться сервер. На этих примерах выполнены HTML-запросы, для тестирования на PHP-запросы измените цель на index.php (на адрес установленного ранее сайта).

**Использование httperf**

Еще одно консольное приложение, используемое также для создания нужного количества параллельных запросов - httperf.

Его отличие от ab в том, что httperf посылает запросы согласно своим настройкам, невзирая на то, отвечает сервер на них или уже нет. Таким образом можно определить не только какую максимальную нагрузку может выдержать сервер, но и как будет себя вести сервер в момент, когда нагрузка достигла своего пика.

Пример запуска 100 запросов от 10 посетителей параллельно:

httperf --port 80 --server <domain> --uri=/ --num-conns=100 --rate=10

**Использование Siege**

Установка:

yum install siege -y

Количество запросов не лимитируется, но можно задавать время в течение которого выполнять тестирование. Пример: 5 пользователей, которые безостановочно загружают главную страницу в течении одной минуты.

siege -c 5 -b -t 1m ip-адрес

**Nginx как балансировщик нагрузки**

Балансировка нагрузки (англ. load balancing) — метод распределения заданий между несколькими сетевыми устройствами (например, серверами) с целью оптимизации использования ресурсов, сокращения времени обслуживания запросов, горизонтального масштабирования кластера (динамическое добавление/удаление устройств), а также обеспечения отказоустойчивости (резервирования).

Для балансировки изменяем файл конфигурации приблизительно следующим образом (предполагается, что у нас отвечают сервера .11 и .22):

upstream web\_backend {

server 192.168.1.11;

server 192.168.1.22;

}

server {

listen 80;

location / {

proxy\_set\_header XForwardedFor $proxy\_add\_x\_forwarded\_for;

proxy\_pass http://web\_backend;

}

}

После настройки перезапускаем Nginx

service nginx reload

Методы балансировки нагрузки (описываются в начале секции upstream):

* ip\_hash - согласно этому методу запросы от одного и того же клиента будут всегда отправляться на один и тот же backend сервер на основе информации об ip адресе клиента. Не совместим с параметром weight.
* least\_conn - запросы будут отправляться на сервер с наименьшим количеством активных соединений.
* round-robin - режим по умолчанию. То есть если вы не задали ни один из вышеупомянутых способов балансировки - запросы будут доставляться по очереди на все сервера в равной степени.

**Задания к лабораторной работе**

Сравним функционирование веб-серверов Apache и Nginx под нагрузкой. Для этого запустим оба веб-сервера, это можно сделать даже на одной машине, указав разные порты в конфигурационных файлах.

Nginx – порт :80

Apache – порт :10080 или любой выше 1024.

**Нагрузочное тестирование веб-сервера с Apache**

*Для тестирования используются 2 машины – одна с установленным и работающим веб-сервером Apache, вторая будет отсылать запросы и делать выводы о производительности web-сервера.*

Тестирование на PHP-запросы:

* Определить максимальное число параллельных запросов, при котором сервер нас не будет блокировать.
* Провести тест при использовании максимального числа запросов.

Тестирование на HTML-запросы:

* Определить максимальное число параллельных запросов
* Провести тест при использовании максимального числа запросов.

Провести сравнение результатов и сформировать выводы.

**Нагрузочное тестирование веб-сервера с Nginx.**

*Для тестирования используется 2 виртуальные машины – одна с установленным и работающим Nginx, которой будут отсылаться запросы, другая будет отсылать эти самые запросы и делать выводы о производительности веб-сервера с Nginx.*

Тестирование на PHP-запросы:

* Определить максимальное число параллельных запросов, при котором сервер нас не будет блокировать.
* Провести тест при использовании максимального числа запросов.
* Сравнить с результатами, полученными при тестировании Apache

Тестирование на HTML-запросы:

* Определить максимальное число параллельных запросов.
* Провести тест при использовании максимального числа запросов.
* Сравнить с результатами, полученными при тестировании Apache

Провести сравнение результатов и сформировать выводы.

**Нагрузочное тестирование веб-серверов Apache с балансировщиком нагрузки.**

*Для тестирования используется 4 машины – две одинаковые с установленным и работающим Apache в качестве веб-серверов, которые соединены с третьей машиной, которая выполняет роль балансировщика нагрузки, на нем работает Nginx, четвертая машина будет отсылать эти запросы серверу и делать выводы о производительности данной связки из балансировщика нагрузки на Nginx и двумя веб-серверами Apache.*

Тестирование на PHP-запросы:

* Провести тест при использовании максимального для Apache числа запросов
* Провести тест при использовании максимального для Nginx числа запросов
* Сравнить с предыдущими результатами и сформировать выводы

Тестирование на HTML-запросы:

* Провести тест при использовании максимального для Apache числа запросов
* Провести тест при использовании максимального для Nginx числа запросов
* Сравнить с предыдущими результатами и сформировать выводы

**Нагрузочное тестирование веб-серверов Nginx с балансировщиком нагрузки.**

*Для тестирования используется 4 виртуальные машины – две одинаковые с установленным и работающим Nginx в качестве веб- серверов, которые соединены с третьей машиной, которая выполняет роль балансировщика нагрузки, на нем работает Nginx, четвертая машина будет отсылать эти запросы серверу и делать выводы о производительности данной связки из балансировщика нагрузки на Nginx и двумя веб-серверами Nginx.*

Тестирование на PHP-запросы:

* Провести тест при использовании максимального для Apache числа запросов
* Провести тест при использовании максимального для Nginx числа запросов
* Сравнить с предыдущими результатами и сформировать выводы

Тестирование на HTML-запросы:

* Провести тест при использовании максимального для Apache числа запросов
* Провести тест при использовании максимального для Nginx числа запросов
* Сравнить с предыдущими результатами и сформировать выводы

| *Итоговая таблица сравнения* | | | | | |
| --- | --- | --- | --- | --- | --- |
|  |  | **Максимальное число запросов** | **Запросы/сек** | **Время, затрачиваемое на запрос, мс** | **% успешных запросов** |
| Apache | PHP+MySQL |  |  |  |  |
|  | Простой сайт на HTML |  |  |  |  |
| LB + Apache | PHP+MySQL |  |  |  |  |
|  | Простой сайт на HTML |  |  |  |  |
| Nginx | PHP+MySQL |  |  |  |  |
|  | Простой сайт на HTML |  |  |  |  |
| LB + Nginx | PHP+MySQL |  |  |  |  |
|  | Простой сайт на HTML |  |  |  |  |

**Внимание! В стенах СГУ не получится выполнить тестирование с балансировщиком (LB). Эту часть опускаем.**

**Постройте диаграммы в Excel по полученной таблице.**

**Вопросы к лабораторной работе**

1. Назначение нагрузочного тестирования?
2. Что такое нагрузка?
3. Как указать ab сделать нагрузку в 10000 запросов, 500 из которых будут направлены одновременно? Перестанет ли ваш сервер принимать входящие подключения?
4. Можно ли протестировать при помощи ab, httperf и Siege другие web-сервера? Назовите примеры.
5. Влияет ли использование скриптовых языков программирования (например, PHP) на производительность web-сервера? Объясните почему.
6. Для чего нужен балансировщик нагрузки?
7. Какие существуют методы балансировки нагрузки в nginx?
8. Составьте отчет о выполнении лабораторной работы.

Включите в него копии экрана и ответы на вопросы лабораторной работы

## Литература

1. Эви Немет, Гарт Снайдер, Трент Р. Хейн, Бэн Уэйли «Unix и Linux. Руководство системного администратора»
2. Брайан Хилл «Полный справочник по Cisco»
3. Виктор Олифер, Наталия Олифер «Компьютерные сети. Принципы, технологии, протоколы»
4. Michael W. Lucas «Network Flow Analysis»
5. Baron Schwartz, Peter Zaitsev, Vadim Tkachenko «High Performance MySQL, 3rd Edition»
6. Jerry Peek, Shelley Powers, Tim O'Reilly, Mike Loukides «Unix Power Tools»
7. Томас А. Лимончелли «Тайм-менеджмент для системных администраторов»
8. Томас А. Лимончелли, Кристина Хоган, Страта Чейлап «Системное и сетевое администрирование. Практическое руководство»
9. Mark A. Dye, Rick McDonald, Antoon W. Rufi "Network Fundamentals, CCNA Exploration" Cisco Companion Guide

## Ресурсы

1) Инструментарий администратора

https://disk.yandex.ru/d/OzuKUdn58Rhxnw

2) Joomla. Надстройка PRO

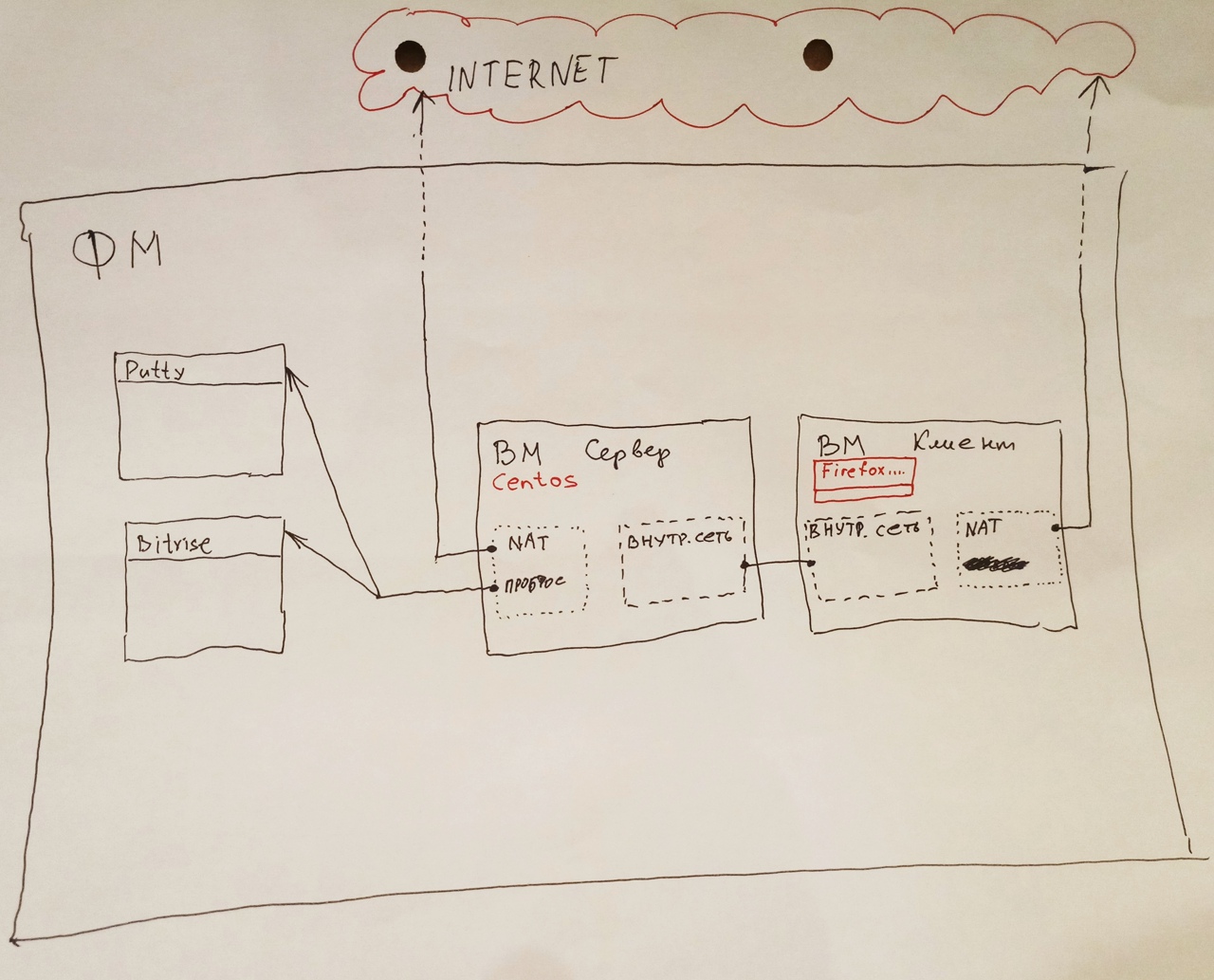
https://disk.yandex.ru/d/H\_a36vB9Oxz6rg

## Приложения

#### Cхема сети

Рассмотрим схему виртуальной компьютерной сети, которая во многом отражает реальную схему взаимодействия администратор-сервер (при условии, что сервер находится в облаках \ в интернете).

В 99% случаев администратор работает с локальной физической машины, используя Putty или Bitvise SSH Client, взаимодействуя с сервером, физически расположенном на другом континенте.



Это структурная схема, которая показывает самый надёжный и удобный на данный момент вариант организации виртуальной сети для выполнения ЛР, при условии, что используется именно VirtualBox.

На физической машине (ФМ) запускаем:

- ВМ **Сервер** (два адаптера: Nat с ***пробросом портов*** и Внутренняя сеть), ОС Rocky Linux

- ВМ **Клиент** (два адаптера: Nat и Внутренняя сеть), OC Windows или Linux c GUI

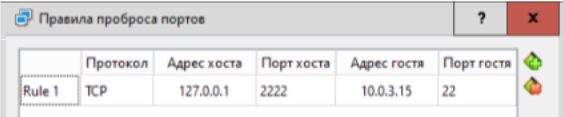
- **Приложение** Putty для управления сервером из консоли (именно так удалённо администрируются боевые сервера)

- **Приложение** Bitvise SSH client для управления сервером из консоли и заливки файлов по SSH

#### Правила проброса портов

Чтобы физическая машина имела доступ к виртуальной машине по конкретному протоколу (т.е. через определённый порт), необходимо настроить проброс портов для адаптера с подключением типа NAT.

Разберём пример для доступа по SSH.



Добавляем новое правило.

* **Имя**. Название можно оставить, которое предлагает система (Rule 1) или выдумать своё.
* **Протокол**. Если вы не уверены, какой выбирать, то ставьте ТСР.
* **Адрес хоста**. Это адрес нашей ФМ, мы будем использовать IP-адрес, который читается как *"этот самый компьютер"* - 127.0.0.1
* **Порт хоста**. Сюда можно написать что угодно, любое число выше 1024, но это число надо запомнить, чтобы затем указать его в Putty\Bitvise. В нашем примере выбрано 2222.
* **Адрес гостя**. Это IP-адрес ВМ, который назначает ей VirtualBox. В подавляющем большинстве случаев это 10.0.3.15. Но возможны девиации. Лучше всего проверить значение, выполнив на сервере **ip a**.
* **Порт гостя**. Ежели мы говорим про протокол SSH, то для гостевой (виртуальной) машины порт SSH по умолчанию - это 22. Но если мы меняли порт на какое-то собственное значение - указываем это значение.

Важно!  
Аналогичным образом можно настроить проброс портов с ВМ на ФМ для других протоколов, за которыми закреплены стандартные порты.

1) Порт 80 - протокол HTTP. Настройте проброс портов с 80 на ХХ80. Тогда из веб-браузера на вашей физической машине сайты на ВМ будут доступны при обращении к порту ХХ80.

2) Порт 443 - протокол HTTP**S**. Настройте проброс портов с 443 на Х443. Тогда из веб-браузера на вашей физической машине сайты c сертификатом SSL на ВМ будут доступны при обращении к порту Х443.

3) и т.п.