Министерство науки и высшего образования Российской Федерации

Федерального государственного бюджетного образовательного учреждения высшего образования

"Сочинский государственный университет"

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

Регистрационный номер \_\_\_

Факультет ФИИЦТ

Кафедра ИТИМ

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**по дисциплине**

**«Локальные сети»**

**Лабораторный практикум**

для направлений (уровень бакалавриата):

44.03.05 «Педагогическое образование с двумя профилями подготовки» профили «Математика и информатика», «Начальное образование и иностранный язык»

**Автор:**

Драч В.Е.,

доцент к.т.н., доцент кафедры ИТИМ

г. Сочи, 2023 г.

# Содержание

[Содержание 2](#_Toc132911407)

[ЛР №1. Подключение по SSH через PuTTY 2](#_Toc132911408)

[ЛР №2. SSH-авторизация по ключу через PuTTY 11](#_Toc132911409)

[Генерация ключа 11](#_Toc132911410)

[Настройка и подключения PuTTY 13](#_Toc132911411)

[ЛР №3\*. Работа с простыми командами bash в терминале Linux. 16](#_Toc132911412)

[ЛР №4\*. Утилиты sed и awk. 21](#_Toc132911413)

[ЛР №5. Обновление системы и пакетов. 26](#_Toc132911414)

[ЛР №6. Установка ZSH и модификация консоли 28](#_Toc132911415)

[ЛР №7. Установка и конфигурация neofetch 35](#_Toc132911416)

[ЛР №8. Сетевой Туннель 39](#_Toc132911417)

[ЛР №9\*. Работа с сетью 44](#_Toc132911418)

[ЛР №10. Управление через веб-интерфейс 52](#_Toc132911419)

[Литература 57](#_Toc132911420)

# ЛР №1. Подключение по SSH через PuTTY

На сегодняшний день подключение к серверу с ОС Linux осуществляется через SSH. Это защищённый сетевой протокол прикладного уровня, производящий удаленное управление ОС и туннелирование TCP-соединений. Его использование обусловлено простотой и надежностью, что позволяет выполнять безопасное подключение к серверу.

Соединение по SSH дает возможность выполнять любые команды на удаленном компьютере так, будто мы находимся прямо перед ним. Такой процесс обычно осуществляется через специальные утилиты, особенно если нужно получить доступ к Linux через Windows. Одной из программ таких является PuTTY.

### Где скачать PuTTY

PuTTY распространяется бесплатно, доступна на официальном сайте разработчика. Скачиваем инсталлятор и запускаем его. Перед нами отобразится окно приветствия – кликаем по кнопке «Next» до конечного пункта. По пути мы также можем изменить местоположение программы, а также добавить или исключить некоторые функции PuTTY.

После успешной установки в меню «Пуск» появится новый ярлык под названием PuTTY – с ним мы и будем работать в последующих разделах.

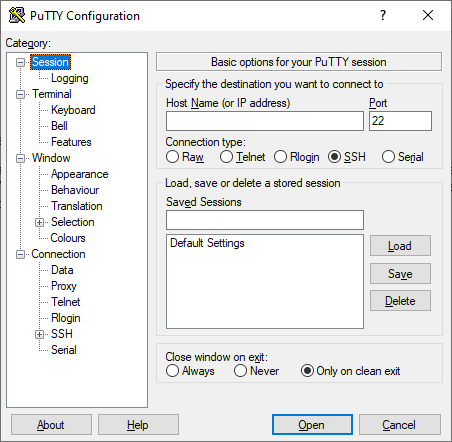
### Что делать в СГУ?

Копируем портативные версии с флэшки преподавателя.

Или скачиваем https://drach.pro/studentu/bak/lan

### Как настроить PuTTY

Откроем программу и посмотрим, что в ней есть. После запуска перед нами отобразится окно конфигурации приложения, где слева находятся различные категории, а справа – их настройки.



Конфигурация включает в себя 4 основных раздела:

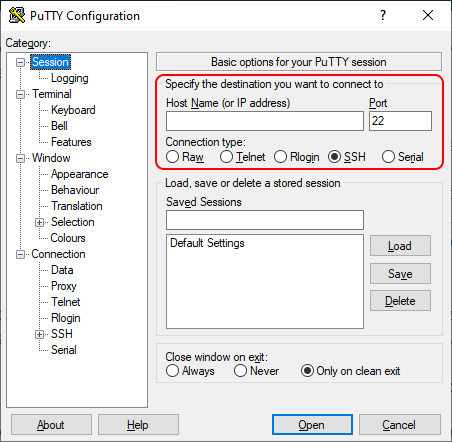
* Session — основная вкладка, предназначенная для подключения к удаленному компьютеру. Здесь вводятся параметры подключения, порт, адрес, а также вносятся так называемые пресеты, которыми можно воспользоваться, чтобы не вводить каждый раз данные для входа. Достаточно один раз прописать настройки сессии, сохранить ее и использовать при следующем запуске программы.
* Terminal — необходима для включения или отключения возможностей терминала.
* Window — здесь производятся все настройки интерфейса: внешний вид окна, цвет, шрифт и кодировка.
* Connection — настройка параметров подключения, алгоритма шифрования, сжатия, ключей аутентификации и других значений.

Нам потребуется всего две вкладки: Session и SSH. Первым делом перейдем ко второй и убедимся, что версия протокола стоит в значении «2». Как правило, сейчас используется понятие SSH-2, так как первая версия практически не применяется из-за существенных недостатков, например, ошибки в схеме обеспечения безопасности.

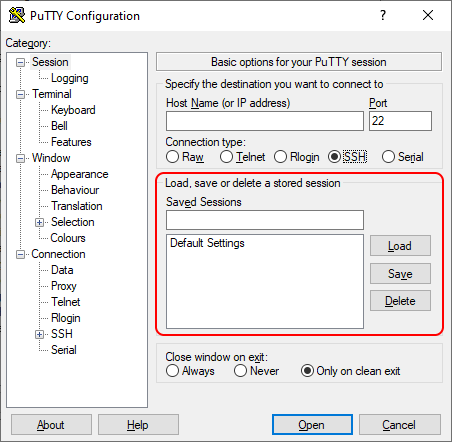


Вернемся к разделу Session, где находятся основные параметры, которые потребуются нам для подключения к серверу по SSH. Давайте немного остановимся и поговорим о них:

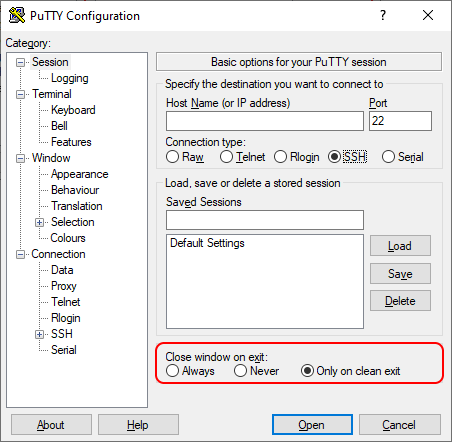
1) Первый блок включает в себя основные параметры, необходимые для подключения: окно ввода IP-адреса и порта, строку с выбором типа подключения.



2) Следующий блок предназначен для создания пресетов, которые можно использовать для быстрого подключения к удаленному компьютеру. В строку «SavedSessions» вводится название сессии, оно сохраняется с помощью кнопки «Save». Чтобы воспользоваться сохраненным пресетом, достаточно его загрузить с помощью кнопки «Load».



3) Последний блок – параметры закрытия окна при выходе. Доступные значения: always, never, only on clean exit.



Как видите, в настройках PuTTY нет ничего сложного. Теперь давайте перейдем к подключению по протоколу SSH.

### Как подключиться по SSH

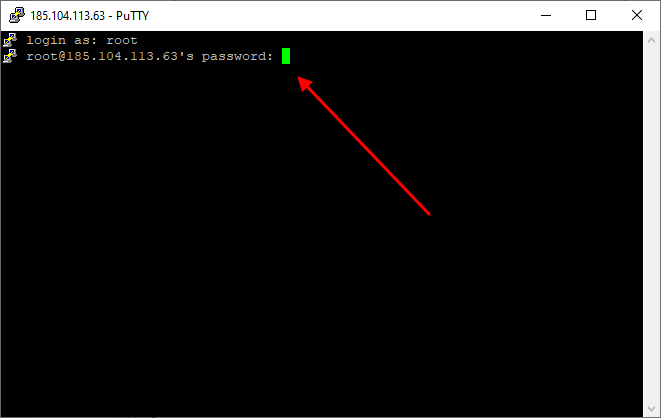
Нам понадобится IP-адрес удаленного компьютера или сервера, к которому мы хотим подключиться. Последующие действия мы будем рассматривать на примере wp.drach.pro.

Переходим к PuTTY – запускаем программу и открываем раздел «Session»; вводим адрес удаленного компьютера, указываем порт 22 и в нижней части приложения кликаем по кнопке «Open».



При успешном вводе данных перед нами отобразится консольное окно, в котором нужно ввести логин и пароль для подключения к серверу. Как правило, надо согласовать с преподавателем.

Если логин и пароль были введены корректно, то мы увидим окно приветствия.



Таким образом мы подключились к ПК на операционной системе Linux. Здесь мы можем вводить команды и удаленно взаимодействовать с компьютером.

### Основные команды Linux

Команд огромное множество, но для первоначального изучения достаточно знать лишь их небольшую часть.

ls — предназначена для вывода списка файлов в директории, содержит в себе набор команд:

ls -la — выводит различные документы и показывает права доступа, включая имя создателя и владельца;

ls -lha — также выводит файлы, но показывает размер файла в байтах;

ls -lha | less — используется для постраничного просмотра документов.

cd — необходима для перемещения между директориями, например, cd [путь] перенаправляет пользователя в указанную папку, cd../ перемещает в верхний уровень, cd ~ открывает корневую папку.

echo — позволяет перемещать текст в файл. Например, если нам требуется создать новый текстовый документ или добавить текст в уже созданный файл, достаточно ввести:

echo Привет, мир! >> new.txt

sudo — наиболее используемая команда, означающая «SuperUserDo». Она позволяет выполнять любые команды от имени администратора. Её мощный вариант повышения прав на всё время сессии: sudo su –

(да, в конце минус!)

df — дает возможность увидеть доступное дисковое пространство в каждом из разделов операционной системы. Синтаксис: df [опции устройство].

cp — для копирования файлов и целых директорий воспользуйтесь этой строчкой. Синтаксис: cp [опции файл-источник файл-приемник].

mc — предназначена для запуска файлового менеджера MidnightCommander, который позволяет работать с файлами по FTP внутри консольного окна.

cat — наиболее популярная команда в Linux, позволяющая считывать данные из файлов и выводить их содержимое. Наиболее простой способ отображения содержимого в командной строке. Синтаксис: cat [имя файла].

mkdir — позволяет создать одну или несколько директорий с указанным именем. Например, mkdir mdir1 mdir2 mdir3 добавит сразу 3 директории.

chmod — изменяет права доступа к файлам и каталогам. Синтаксис: chmod [ключи установка\_прав имя\_файла].

pwd — при первом запуске терминала мы оказываемся в домашнем каталоге пользователя. Если вдруг каталог был изменен, то проверить это можно с помощью данной команды.

touch — используется для создания любого типа файлов: от пустого txt до пустого zip. Синтаксис: touch [имя файла].

man и --help — нужны для того, чтобы узнать информацию о команде и о том, как ее можно использовать (man). Например, «man cat» покажет доступную информацию о команде cat. Ввод имени команды и аргумента помогает показать, каким образом можно использовать команду, например, cd --help.

locate — предназначена для поиска файла в системе Linux.

passwd – смена собственного пароля

Это была лишь небольшая часть команд, которые можно использовать при подключении по SSH.

### Выполнение работы

Убедитесь, что для вас создана учётная запись.

Если ещё не создана, покажите шпаргалку преподавателю:

adduser имя

echo "имя ALL=(ALL) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/имя

passwd имя

Выполните подключение к серверу по протоколу SSH, используя полученный адрес и свою учётную запись.

Выполните смену пароля на собственный.

Выполните простые команды (по выбору).

Запустите файловый менеджер mc.

Создайте и отредактируйте тестовый файл. Выведите его содержимое в консоль через cat.

Отключитесь командой exit.

### Содержание отчёта

1. Цель работы
2. Основные этапы выполнения с иллюстрациями.
3. Постарайтесь не включать в отчёт пароли.
4. Выводы

# ЛР №2. SSH-авторизация по ключу через PuTTY

Альтернативная методика описана на сайте:

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

На сегодняшний день набирает обороты практика аутентификации к SSH-серверу по ключу, в этом есть несколько преимуществ:

* **Удобство в использовании.** У вас может быть несколько серверов и один ключ, не нужно всякий раз вводить пароль от нужного сервера, что сэкономит время.
* **Это безопасно.** Вы сможете отключить аутентификацию по паролю и ssh будет защищён от взлома методом перебора паролей.

Настроим SSH авторизации на сервере Linux по ключу, и настроим подключение через ключ утилиту **[PuTTY](http://www.putty.org/" \t "_blank" \o "http://www.putty.org/).**

Генерация ключа

Для генерации ключа будем использовать утилиту **[PuTTYgen](http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html" \t "_blank" \o "http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html).** После запуска выбираем тип ключа для генерации - **SSH-2 RSA**, и длину ключа - **2048 бит** (для простоты можно SSH-1 с длиной 1024). После чего нажимаем Generate и **яростно** гоняем курсор мышки во все стороны (необходимо для быстрого получения действительно случайного ключа) пока не закончится генерация ключа.

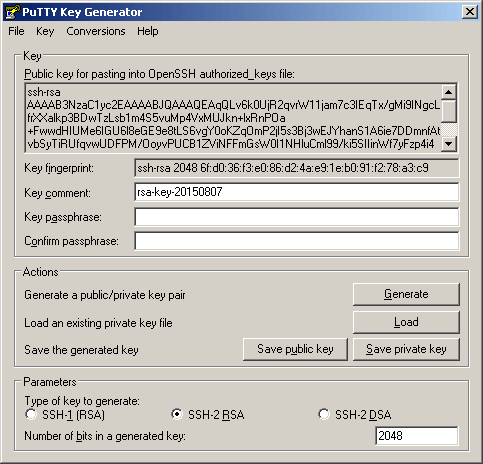


Рис. 2.1

* **Keycomment** - комментарий к ключу.
* **Keypassphrase** - парольная фраза к приватному ключу. (Не обязательно к заполнению)
* **Confirmpassphrase** — подтверждение парольной фразы.

Если вы хотите обезопасить себя по-максимуму, вы можете задать пароль для защиты приватного ключа в полях **KeyPassphrase** и**ConfirmPassphrase**. Но при каждом входе у вас будет запрашивать пароль. Это обезопасит вас если ваш приватный ключ будет похищен. Не надо делать это в учебных целях!

Далее сохраняем **publickey** и **privatekey**. Приватный ключ мы будем использовать на своей машине для подключения к серверу, а вот публичный ключ надо будет передать на удаленный сервер, к которому мы будем подключаться.

### Установка публичного ключа на сервере

Далее необходимо скопировать наш публичный ключ на сервер. Вариантов много. Выбираем самый простой и понятный.

Идём в домашнюю директорию, создаем файл:

**cd /home/имя\_пользователя**

**mkdir .ssh**

**chmod 0700 .ssh**

**touch .ssh/authorized\_keys**

**cd ~/.ssh**

**chmod 0600 ./authorized\_keys**

Запускаем **mc** , заходим в папку .ssh и редактируем файл **authorized\_keys** через F4: надо вставить публичный ключ через буфер обмена – копируем его из окна рис.2.1. и вставляем сочетанием Ctrl+Insert в файл. На всякий пожарный дополнительно вставляем последнюю пустую строку - Enter. Сохраняемся.

На этом шаге обычно возникает много ошибок. Есть смысл проверить себя (уточнить у преподавателя?):

- есть ли файл **~/.ssh/authorized\_keys** в моей домашней папке?

- он содержит мой ключ?

- есть последняя пустая строка?

- я являюсь владельцем файла?

- файл имеет права 600 (только я и группа его можем изменять)?

Далее нужно проверить настройки нашего SSH сервера, сами настройки лежат в файле**/etc/ssh/sshd\_config**:

|  |
| --- |
| # Путь до публичного ключа в домашней директиве.  AuthorizedKeysFile .ssh/authorized\_keys  # Разрешить аутентификацию по ключу  RSAAuthentication yes  PubkeyAuthentication yes |

Теперь нам надо будет перезапустить SSH сервер:

|  |
| --- |
| service sshd restart |

Теперь мы можем подключиться к серверу по ключу или паролю.

Но мы можем вообще запретить подключатся к SSH серверу по паролю указав в файле **/etc/ssh/sshd\_config**.

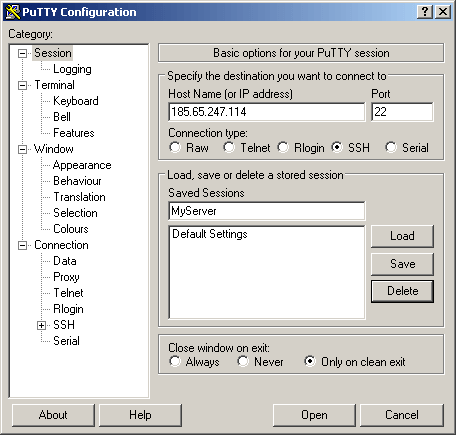
|  |
| --- |
| PasswordAuthentication no |

После чего не забываем заново перезапустить SSH сервер.

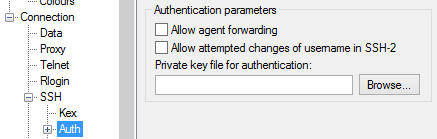
Настройка и подключения PuTTY

Теперь все что нам осталась это создать новый сеанс и настроить его подключения к SSH серверу по ключу.

Создаем новый сеанс подключения к серверу. Уточните адрес у преподавателя, не берите его с картинки!



Далее, мы выбираем в меню **категорию SSH** и там выберем подкатегорию Auth.



В поле **Private key file for authentication** загружаем наш приватный ключ.

Кладите ключ поближе к корню диска, не допускайте пробелов и русских букв в пути!

Далее переходим обратно в раздел Session и сохраняем нашу сессию для того чтобы следующий раз не заполнять все эти поля.

Остаётся только проверить правильность аутентификации к SSH серверу по ключу. Запустите PuTTY и подключитесь к своему серверу~~. Если вы при создании, ключа заполнили поле~~**~~KeyPassphrase и ConfirmPassphrase~~**~~то вас запросит вести этот пароль.~~ Если же всё настроено неправильно, то будет выдано сообщение об ошибке и предложено ввести пароль. Прочитайте текст ошибки, чтобы понять, на каком шаге мы ошиблись.

### Выполнение работы

Выполните подключение к серверу по протоколу SSH, используя свою учётную запись.

Перенесите публичный ключ на сервер.

- Запустите файловый менеджер mc.

- Отредактируйте тестовый файл.

Отключитесь командой exit.

Убедитесь в подключении без ввода пароля.

### Содержание отчёта

1. Цель работы
2. Основные этапы выполнения с иллюстрациями.
3. Постарайтесь не включать в отчёт пароли.
4. Выводы

# ЛР №3\*. Работа с простыми командами bash в терминале Linux.

Для начала создадим файл file1.txt с помощью команды

touch file1.txt

Для создания нескольких файлов используется команда

touch file1.txt file2.txt file3.txt

**КОМАНДАcat**

Название команды - это сокращения от слова *catenate*. По сути, задача команды cat очень проста - она читает данные из файла или стандартного ввода и выводит их на экран.

Также файл можно создать cat>file2.txt

Найдём файл через поиск и внесём изменения



В терминале посмотрим, как работают варианты команды  
*-b - нумеровать только непустые строки;*

*-E - показывать символ $ в конце каждой строки;*

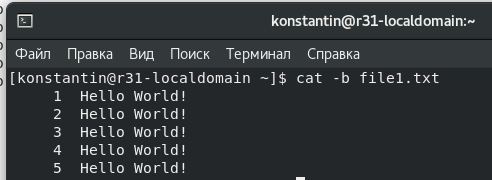
*-n - нумеровать все строки;*

*-s - удалять пустые повторяющиеся строки;*

*-T - отображать табуляции в виде ^I;*

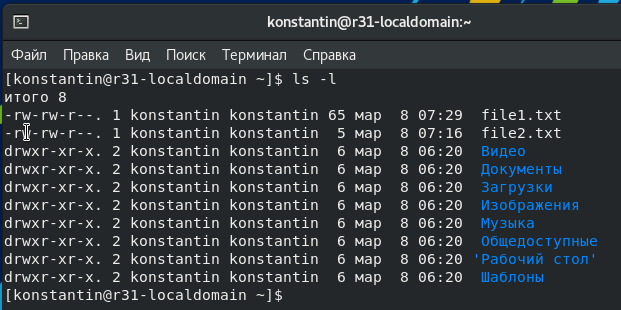
*-h - отобразить справку;*

*-v - версия утилиты.*



**КОМАНДА ls**

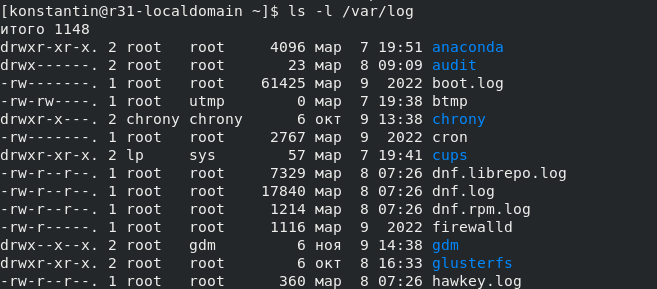
ls –l – подробный список



Вывести размер файлов в человеко-понятном формате (опция -h):

ls -lh

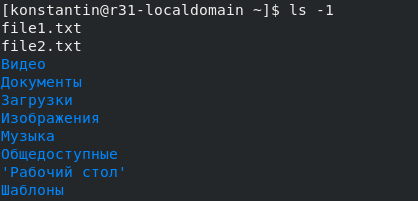
Если необходимо вывести содержимое другой директории, то нужно указать путь до директории, например:



ls –a - Показать скрытые файлы

ls –lF - Обозначить директории

ls -1 - Вывести список в одну колонку



Выведем файлы только с расширением .txt/.html, .php и .txt:

ls \*.txt

ls\*.{html,php,txt}

**Команда tail**

В простейшей форме при использовании без какой-либо опции команда tail отобразит последние 10 строк.

tail file1.txt

*-c <n>- выводить указанное количество байт с конца файла;*

*-f - обновлять информацию по мере появления новых строк в файле;*

*-n<n> - выводить указанное количество строк из конца файла;*

*--pid - используется с опцией -f, позволяет завершить работу утилиты, когда завершится указанный процесс;*

*-q - не выводить имена файлов;*

*--retry - повторять попытки открыть файл, если он недоступен;*

*-v - выводить подробную информацию о файле;*

**Команда history**

*-c - очистить историю;*

*-d - удалить определенную строку из истории;*

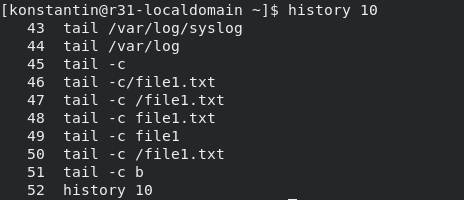
*-a - добавить новую команду в историю;*

*-n - скопировать команды из файла истории в текущий список;*

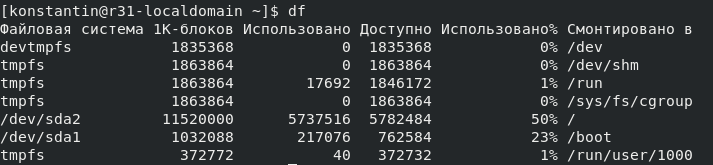
*-w - перезаписать содержимое одного файла истории в другой, заменяя повторяющиеся вхождения.*

Посмотреть последние 10 команд:

history 10



Чтобы посмотреть доступное пространство на всех примонтированных разделах и информацию о них достаточно набрать: df



Задания:

***Для чётных вариантов***  
1. Создать файл и пронумеровать только непустые строки, показать вывод команды cat

2.В директории /tmp показать вывод ls

3. Исследовать вывод команды tailв режиме реального времени: для этого запустить 2 терминала параллельно и выполнить в первом tail -f /var/log/messages , а в другом sudo yum install httpd –y (устанавливаем веб-сервер Apache, наблюдая изменения в основном системном журнале).

Если веб-сервер уже установлен (и мы видим вывод: Nothing to do), значит устанавливаем системный монитор:

sudo yum install htop –y

или

sudo yum install atop –y

Если и эти две программы установлены, то удаляем их:

sudo yum erase htop –y

или

sudo yum erase atop –y

4. Удалить 1 строку в истории, перед этим сделав скриншот (получится только из-под своего реального профиля, без повышенных прав)

***Для нечётных вариантов***  
1. Создать файл и показывать символ $ в конце каждой строки, показать вывод команды cat

2. В директории /var/log показать вывод ls

3. исследовать вывод команды tailв режиме реального времени: для этого запустить 2 терминала параллельно и выполнить в первом tail -f /var/log/messages , а в другом sudo yum install mc –y (устанавливаем файловый менеджер MidnightCommander, наблюдая изменения в основном системном журнале)

Если веб-сервер уже установлен (и мы видим вывод: Nothing to do), значит устанавливаем системный монитор:

sudo yum install htop –y

или

sudo yum install atop –y

Если и эти две программы установлены, то удаляем их:

sudo yum erase htop –y

или

sudo yum erase atop –y

4. Удалить 2 строку в истории, перед этим сделав скриншот (получится только из-под своего реального профиля, без повышенных прав)

# ЛР №4\*. Утилиты sed и awk.

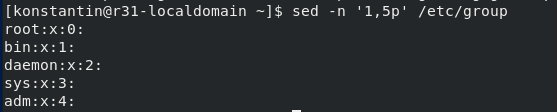
Сначала рассмотрим синтаксис команды sed:

$ sed опции -e команды файл

А вот её основные опции:

*-n, --quiet - не выводить содержимое буфера шаблона в конце каждой итерации;*

*Вывод из файла 1-5 страницы:*



*-e - команды, которые надо выполнить для редактирования;*

*-f - прочитать команды редактирования из файла;*

*-i - сделать резервную копию файла перед редактированием;*

*-l - указать свою длину строки;*

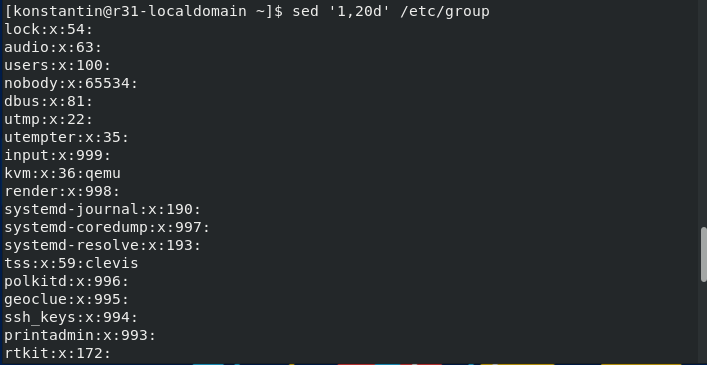
*-r - включить поддержку расширенного синтаксиса регулярных выражений;*

*-s - если передано несколько файлов, рассматривать их как отдельные потоки, а не как один длинный.*

У утилиты есть два буфера, это активный буфер шаблона и дополнительный буфер. Оба изначально пусты. Программа выполняет заданные условия для каждой строки в переданном ей файле.

sed читает одну строку, удаляет из неё все завершающие символы и символы новой строки и помещает её в буфер шаблона. Затем выполняются переданные в параметрах команды, с каждой командой может быть связан адрес, это своего рода условие и команда выполняется только если подходит условие.

Когда всё команды будут выполнены и не указана опция -n, содержимое буфера шаблона выводится в стандартный поток вывода перед этим добавляется обратно символ перевода строки. если он был удален. Затем запускается новая итерация цикла для следующей строки.



**АДРЕСА SED**

Каждой команде можно передать адрес, который будет указывать на строки, для которых она будет выполнена:

*номер - позволяет указать номер строки, в которой надо выполнять команду;*

*первая~шаг - команда будет выполняется для указанной в первой части сроки, а затем для всех с указанным шагом;*

*$ - последняя строка в файле;*

*/регулярное\_выражение/ - любая строка, которая подходит по регулярному выражению. Модификатор l указывает, что регулярное выражение должно быть не чувствительным к регистру;*

*номер, номер - начиная от строки из первой части и заканчивая строкой из второй части;*

*номер, /регулярное\_выражение/ - начиная от сроки из первой части и до сроки, которая будет соответствовать регулярному выражению;*

*номер, +количество - начиная от номера строки указанного в первой части и еще плюс количество строк после него;*

*номер, ~число - начиная от строки номер и до строки номер которой будет кратный числу.*

**СИНТАКСИС РЕГУЛЯРНЫХ ВЫРАЖЕНИЙ**

Вы можете использовать такие же регулярные выражения, как и для Bash и популярных языков программирования. Вот основные операторы, которые поддерживают регулярные выражения sedLinux:

*\* - любой символ, любое количество;*

*\+ - как звездочка, только один символ или больше;*

*\? - нет или один символ;*

*\{i\} - любой символ в количестве i;*

*\{i,j\} - любой символ в количестве от i до j;*

*\{i,\} - любой символ в количестве от i и больше.*

**КОМАНДЫ SED**

Чтобы пользоваться sed, рассмотрим простейшие команды, которые применяются чаще всего:

*# - комментарий, не выполняется;*

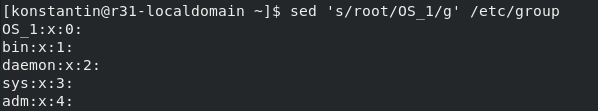
*q - завершает работу сценария;*

*d - удаляет буфер шаблона и запускает следующую итерацию цикла;*

*p - вывести содержимое буфера шаблона;*

*n - вывести содержимое буфера шаблона и прочитать в него следующую строку;*

*s/что\_заменять/на\_что\_заменять/опции - замена символов, поддерживаются регулярные выражения; Замена rootна OC\_1*



*y/символы/символы - позволяет заменить символы из первой части на соответствующие символы из второй части;*

*w - записать содержимое буфера шаблона в файл;*

*N - добавить перевод строки к буферу шаблона;*

*D - если буфер шаблона не содержит новую строку, удалить его содержимое и начать новую итерацию цикла, иначе удалить содержимое буфера до символа перевода строки и начать новую итерацию цикла с тем, что останется;*

*g - заменить содержимое буфера шаблона, содержимым дополнительного буфера;*

*G - добавить новую строку к содержимому буфера шаблона, затем добавить туда же содержимое дополнительного буфера.*

**Примеры модификации текстовых файлов**

Найти все трехзначные числа и заменить их строкой number:

sed -i 's/\b[0-9]\{3\}\b/number/g' file.txt

Добавить фигурные скобки {} вокруг каждого трехзначного числа, введите:

sed -i 's/\b[0-9]\{3\}\b/{&}/g' file.txt

И последнее, но не менее важное: всегда полезно сделать резервную копию при редактировании файла с помощью sed. Дляэтогопростоиспользуем опцию -i.

**СИНТАКСИС КОМАНДЫ AWK**

Сначала надо понять, как работает утилита. Она читает документ по одной строке за раз, выполняет указанные вами и отправляет результат на стандартный вывод. Одна из самых частых задач, для которых используется awk - это выборка одной из колонок. Все параметры awk находятся в кавычках, а действие, которое надо выполнить - в фигурных скобках. Основной её синтаксис:

$awk опции 'условие {действие}'

$ awk опции 'условие {действие} условие {действие}'

С помощью действия можно выполнять преобразования с обрабатываемой строкой. Об этом мы поговорим позже, а сейчас давайте рассмотрим опции утилиты:

*-F, --field-separator - разделитель полей, используется для разбиения текста на колонки;*

*-f, --file - прочитать данные не из стандартного вывода, а из файла;*

*-v, --assign - присвоить значение переменной, например foo=bar;*

*-b, --characters-as-bytes - считать все символы однобайтовыми;*

*-d, --dump-variables - вывести значения всех переменных awk по умолчанию;*

*-D, --debug - режим отладки, позволяет вводить команды интерактивно с клавиатуры;*

*-e, --source - выполнить указанный код на языке awk;*

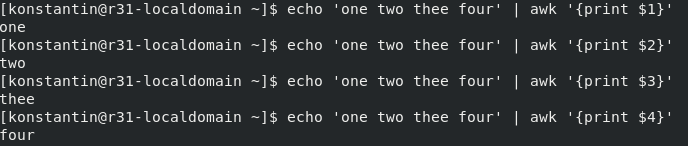


*-o, --pretty-print - вывести результат работы программы в файл;*

*-V, --version - вывести версию утилиты.*

Это далеко не все опции awk. Теперь перечислим несколько функций-действий, которые вы можете использовать:

*print(строка) - вывод чего либо в стандартный поток вывода;*



*printf(строка) - форматированный вывод в стандартный поток вывода;*

*system(команда) - выполняет команду в системе;*

*length(строка) - возвращает длину строки;*

*substr(строка, старт, количество) - обрезает строку и возвращает результат;*

*tolower(строка) - переводит строку в нижний регистр;*

*toupper(строка) - переводить строку в верхний регистр.*

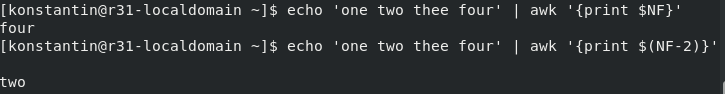
Функций намного больше, но чтобы не загромождать статью я привел только те, которые мы будем использовать сегодня, а также ещё несколько для чтобы вы могли оценить масштаб возможностей утилиты.

В функциях-действиях можно использовать различные переменные и операторы, вот несколько из них:

*FNR - номер обрабатываемой строки в файле;*

*FS - разделитель полей;*

*NF - количество колонок в данной строке;*



*NR - общее количество строк в обрабатываемом тексте;*

*RS - разделитель строк, по умолчанию символ новой строки;*

*$ - ссылка на колонку по номеру.*

Задания:

Создайте текстовый файл, скопировав конфигурационный cp /etc/ssh/**sshd\_config** /home. Приветствуется использование собственного текстового файла. Выполните над ним действия.

**Для чётных вариантов**

1. Используя командную строку удалите все строки комментариев, то есть начинающиеся с символа #.

2\*.Выведите содержимое буфера шаблона файла root.

3. Написать 5 слов (можно из примеров), вывести чётные слова одним print с помощью awk.

4. Перевести слова в нижний регистр и вывести в терминал.

5. Сложить два числа с помощью переменной.

6. Замените все двузначные числа на ХХ.

**Для нечётных вариантов**

1.Используя командную строку удалите все строки, содержащие цифры.

2\*.Выведите содержимое буфера шаблона файла root.

3. Написать 5 слов (можно из примеров), вывести нечётные слова одним print с помощью awk.

4. Перевести слова в верхний регистр и вывести в терминал.

5. Вычесть два числа с помощью переменной.

6. Оберните все двузначные числа фигурными скобками.

# ЛР №5. Обновление системы и пакетов.

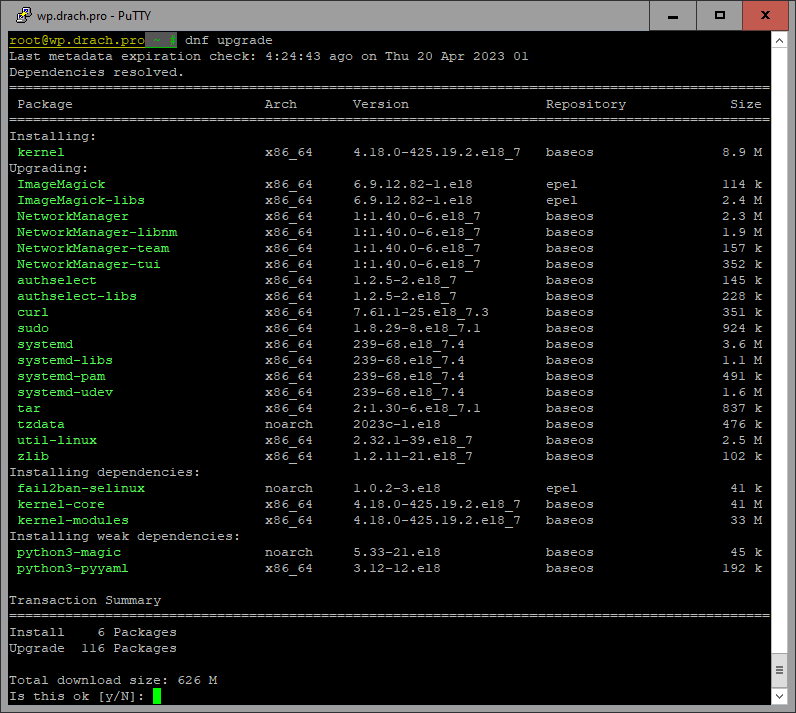
Цель работы: научиться обновлять операционную систему Linux  
 **Теоретические сведения**

Линукс – это операционная система, которая как правило (но не всегда) состоит из пакетов. Например, если в системе отсутствует архиватор zip, обычно его можно установив, добавив пакет zip. Пакеты можно добавлять, удалять, обновлять. Основных форматов пакета всего два – deb и rpm.

Системы на базе формата rpm как правило имеют менеджер пакетов yum или dnf. Интересно отметить, что их синтаксис идентичен. Ещё более интересно, что обычно в системе установлен только один менеджер пакетов, а имя второго является псевдонимом для установленного. Вывод: каждый сам выбирает, какую команду писать dnf или yum – по зову сердца.

**Ход выполнения работы**

1) Вывести информацию о пакетах, которые можно обновить: dnf upgrade



Система покажет, какие более свежие версии каких пакетом могут быть обновлены. В первом столбце перечислены все пакеты, которые попадают под обновление; причём выделены категории

* Installing – будет установлено (это новые пакеты, которых раньше не было, или же свежая версия ядра)
* Updating – будет обновлено до актуальной версии
* Installing Dependencies – будет установлено не потому, что это требуется для новых пакетов
* Installing Weak Dependencies – будет установлено не потому, что может потребоваться для новых пакетов

Менеджер пакетов останавливается на вопросе: обновить всю систему целиком?

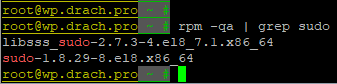
Всегда отвечаем «нет» - вводим N.

2) Выберите любой пакет (кроме kernel) из списка выше и узнайте его версию

rpm -qa | grep имя

Например, мы хотим узнать про пакет “sudo”:

rpm -qa | grep sudo



На рисунке видим, что версия установленного пакета 1.8.29.8

Суффикс el8 указывает на то, что данный дистрибутив операционной системы построен на Red Hat **E**nterprise **L**inux **8** (EL8).

3) обновите выбранный пакет: dnf имя

Например, мы хотим обновить подсистему выполнения команд от имени другого пользователя sudo:

dnf sudo

Соглашаемся с предложением – y.

4) Повторяем п.2, удивляемся изменениям.

**Содержание отчета**

* Титульный лист
* Цель
* Выполнить п.1, привести примеры пакетов, которые могут быть обновлены, их размер, из какого репозитория они будут скачиваться
* Выполнить п.2, привести имя, версию, архитектуру и разрядность пакета
* Выполнить п.3, объяснить, что произошло, сколько пакетов обновилось (установилось), какой объём информации был скачан, сколько времени потребовалось
* Выполнить п.4, сравнить с результатами п.2.
* Выводы

# ЛР №6. Установка ZSH и модификация консоли

Цель работы: научиться устанавливать ZSH и модифицировать консоль Linux

Редкий случай, когда в лабораторно работе НЕ следует использовать постоянное повышение прав до суперпользователя (команда sudo su), а следует работать из-под своей учётной записи, вставляя перед командами sudo если требуется.

**Теоретические сведения**

**Оболочка** - это специальная пользовательская программа, которая предоставляет пользователю интерфейс для использования служб операционной системы. Оболочка принимает понятные пользователю команды от пользователя и преобразует их во что-то, что может понять ядро. Это интерпретатор командного языка, который выполняет команды, считываемые с устройств ввода, таких как клавиатуры, или из файлов. Оболочка запускается, когда пользователь входит в систему или запускает терминал. Оболочка в целом подразделяется на две категории –

* Оболочка командной строки: наиболее широко используемая оболочка BASH (Bourne Again), CSH (оболочка C), KSH (оболочка Korn), DASH, ZSH…
* Графическая оболочка: KDE, MATE, Gnome…

**# или $** - приглашение для ввода команды. Набираем команду, которую необходимо ввести (сам символ вводить не требуется)

**ZSH** – Современная командная оболочка, используемая в качестве интерактивной оболочки терминала или скриптового интерпретатора

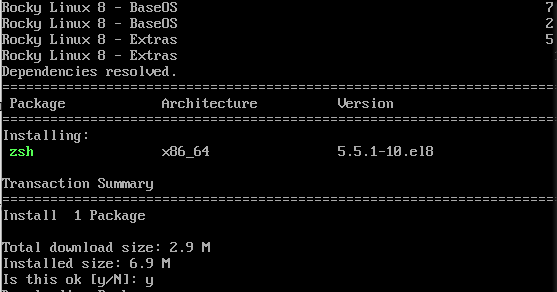
**GNU nano** – консольный текстовый редактор для UNIX и unix-подобных систем.

Вместо nano можно использовать mcedit.

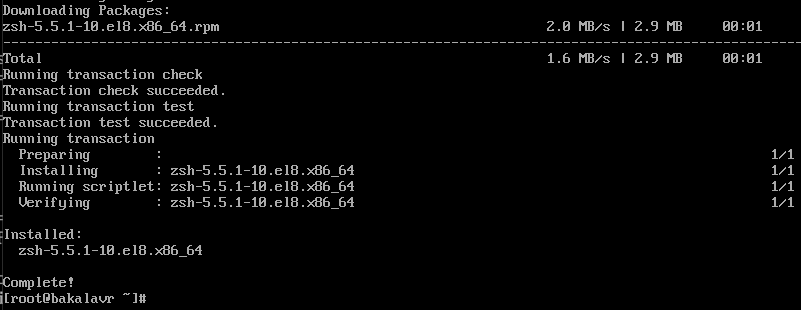
**Установка ZSH пакетным менеджером**

Для установки ZSH нам необходимо ввести команду  
  
 $ sudo dnf install zsh

Получаем подобную картину



Cоглашаемся с правильностью, нажимая y.



ZSH установлен. Запускаем.

$ sudo zsh



Как можно заметить строка ввода **слегка** изменилась - значит вы уже в zsh. Но данные изменения кажутся слишком незначительными, самое время установить плагин. ZSH популярен из-за возможности глубочайшей модернизации и персонализации, а это важно когда вы часто пользуетесь терминалом.

**Установка oh-my-zsh**

Однако для установки плагина нам понадобится git, установим его

$ sudo dnf install git

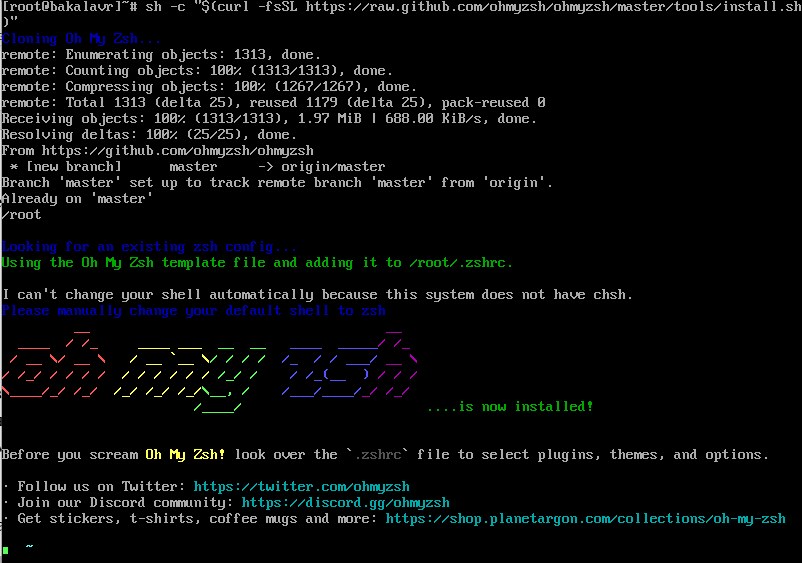
Теперь установим надстройку oh-my-zsh

$ sudo sh -c "$(curl -fsSL [https://raw.github.com/ohmyzsh/ohmyzsh/master/tools/install.sh](https://vk.com/away.php?to=https%3A%2F%2Fraw.github.com%2Fohmyzsh%2Fohmyzsh%2Fmaster%2Ftools%2Finstall.sh&cc_key=))"

Если не получилось, но новая команда:

$ sudo -c "$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"

Наш терминал теперь приобрёл цвета.



Еще больше самых разных плагинов можно найти здесь <https://ohmyz.sh>

Теперь все светится и излучает оптимизм, но хотелось бы, чтобы весь этот карнавал приносил пользу.

**Установим zsh-syntax-highlights**

Установим zsh-syntax-highlighting

$ sudo git clone [https://github.com/zsh-users/zsh-syntax-highlighting.git](https://vk.com/away.php?to=https%3A%2F%2Fgithub.com%2Fzsh-users%2Fzsh-syntax-highlighting.git&cc_key=) ${ZSH\_CUSTOM:-~/.oh-my-zsh/custom}/plugins/zsh-syntax-highlighting

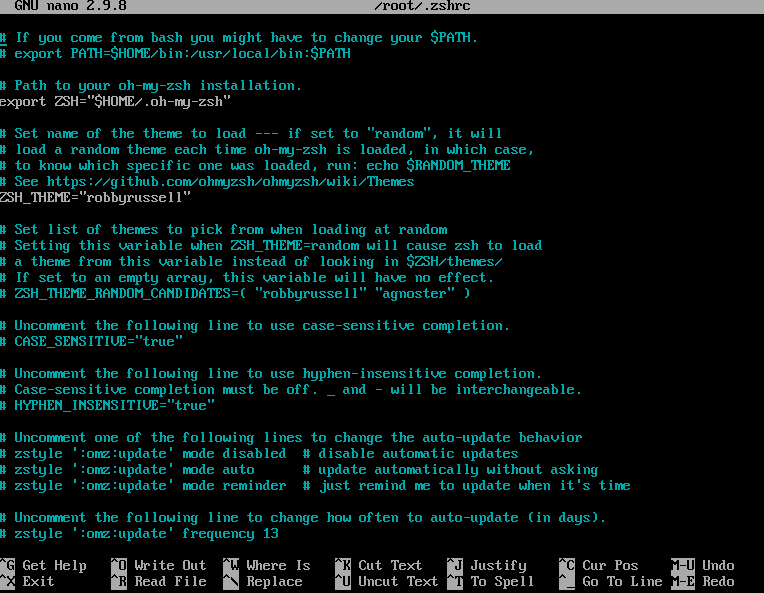
После установки нам нужно внести изменения в файл ~/.zshrc  
но для работы с такими файлами нам нужен текстовый редактор   
установим nano (внимание: можно пользоваться mcedit или mc по старо памяти):

$ sudo dnf nano

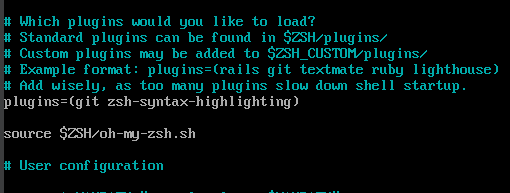
Теперь мы можем редактировать файлы с помощью команды nano

$ sudo nano ~/.zshrc

После чего мы попадаем в текстовый редактор GNU nano



Для перемещения используем стрелки.  
Нам необходимо включить плагин zsh-syntax-highlights, для этого находим строку plugins и добавляем в нее название плагина следующим образом:



Для сохранения нажимаем Ctrl+O, после чего нам предлагают ввести имя файла жмем Enter ничего не меняя. Затем, чтобы выйти из редактора, зажимаем Ctrl+X.

Чтобы изменения вступили в силу, нужно перезайти в ZSH  
последовательно прописываем

# exit

# zsh

Мы снова в zsh теперь при написании команд они будут разделяться цветами, что упрощает ориентирование.

**Установка темы powerlevel10k для ZSH**

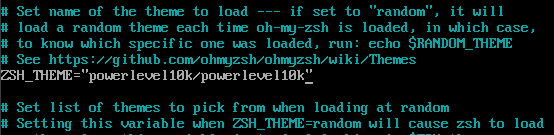
Установим целую тему powerlevel10k

Скачиваем

$ sudo git clone --depth=1 [https://github.com/romkatv/powerlevel10k.git](https://vk.com/away.php?to=https%3A%2F%2Fgithub.com%2Fromkatv%2Fpowerlevel10k.git&cc_key=) ${ZSH\_CUSTOM:-$HOME/.oh-my-zsh/custom}/themes/powerlevel10k  
  
Далее нам снова нужно изменить ~/.zshrc для включения нужной темы

$ sudo nano ~/.zshrc

Меняем параметр ZSH\_THEME на powerlevel10k/powerlevel10k



Для сохранения нажимаем Ctrl+O, после чего нам предлагают ввести имя файла жмем Enter ничего не меняя. Чтобы выйти из редактора зажимаем Ctrl+X.  
  
Чтобы изменения вступили в силу, нужно перезайти в ZSH, последовательно прописываем

# exit

# zsh

Нам предложат настроить время

Выбираем формат 24 (прописываем 2)



Далее предлагают высоту командной строки

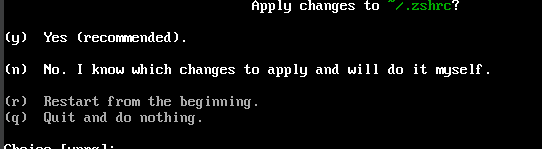


Выбираем One line(прописываем 1)

Остальные параметры на свой вкус, например:

* Prompt Spacing – Compact (1)
* Prompt Flow – Concise (1)
* Transient Prompt – No (n)
* Instant prompt – Verbose (1)

Подтверждаем изменения



Y

Теперь командная строка выглядит примерно так



Вы можете выставить свои значения и строка будет отличаться!

Прочие настройки темы можно найти в файле ~/.p10k.zsh

**Выполнение работы**

-Установите ZSH и надстройку oh-my-zsh

-Установите плагин zsh-syntax-highlighting и тему powerlevel10k

-Настройте свою **УНИКАЛЬНУЮ** среду в терминале

**Содержание отчета**

- Цель

- Основные этапы с иллюстрациями

- Скрин вашей **УНИКАЛЬНОЙ командной** строки, выполнение 3-5 простых команд из ЛР №1.

- Выводы

# ЛР №7. Установка и конфигурация neofetch

Цель работы: Научиться устанавливать утилиту neofetch, извлекать информацию о системе и настраивать его под себя.

**Теоретические сведения**

**# или $** - Указывает команду которую необходимо ввести (сам символ вводить не требуется)

nano и mcedit – текстовые редакторы, которые можно использовать, исходя из собственных предпочтений.

**neofetch –** кроссплатформенная утилита позволяющая отображать сведения о системе

**Установка neofetch пакетным менеджером**

В установке neofetch нет ничего сложного, для этого мы используем всем знакомую команду dnf

# sudo dnf install epel-release

# sudo dnf install neofetch

Возможно, придётся вытянуть ещё несколько дополнительных пакетов во время установки.

После установки мы можем использовать команду neofetch для вывода на экран сведений о системе:

# neofetch



Здесь мы можем видеть всю основную информацию о системе.  
Проверить используемые компоненты, память, процессорное время и дисплей на отображение цветов

Однако самые внимательные заметили что на логотипе пингвин – символ Linux, где же наш логотип Alma или Rocky?

Установим наш логотип командами

# curl -s https://raw.githubusercontent.com/dylanaraps/neofetch/master/neofetch -o /usr/bin/neofetch

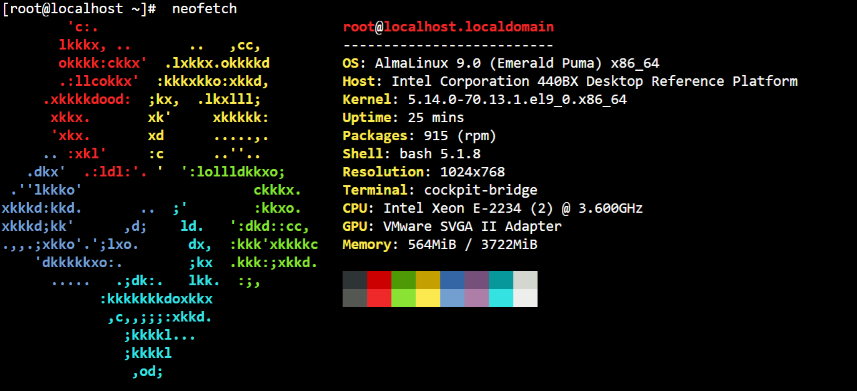
# curl -s https://raw.githubusercontent.com/dylanaraps/neofetch/master/neofetch | grep -o AlmaLinux

Снова откроем neofetch



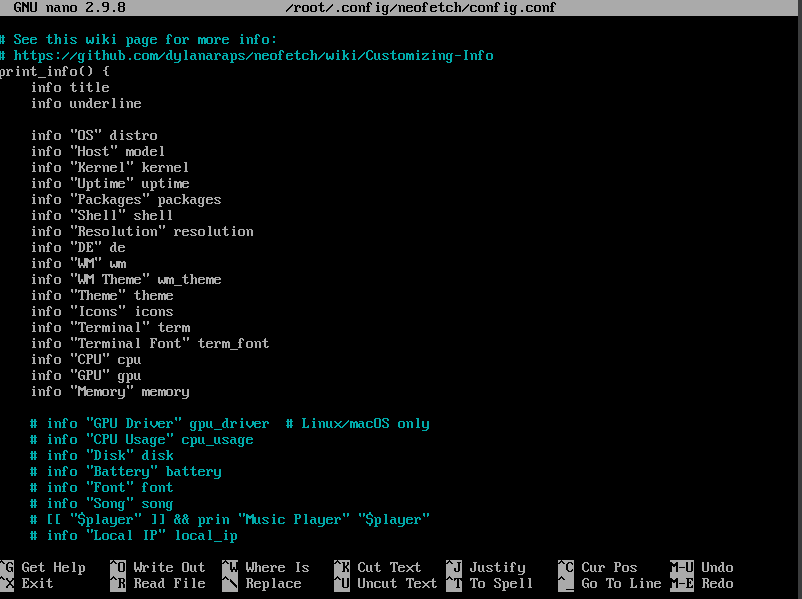
В моем случае появился логотип Rocky поскольку я использую его.

В вашем случае должно быть нечто подобное.



Но neofetch способен на немного большее.  
  
Настроим конфигурацию neofetch

# nano ~/.config/neofetch/config.conf

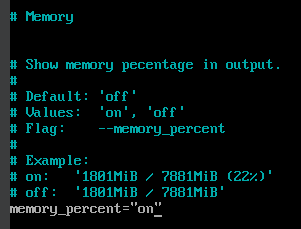


В Файле конфигурации включим отображение информации о диске, а так же отображение IP

Для это нужно раскомментировать (убрать # перед строкой) эти строки



Так же добавим в отображение памяти использование в %  
Для этого найдем раздел memory и переключим параметр memory\_percent на on

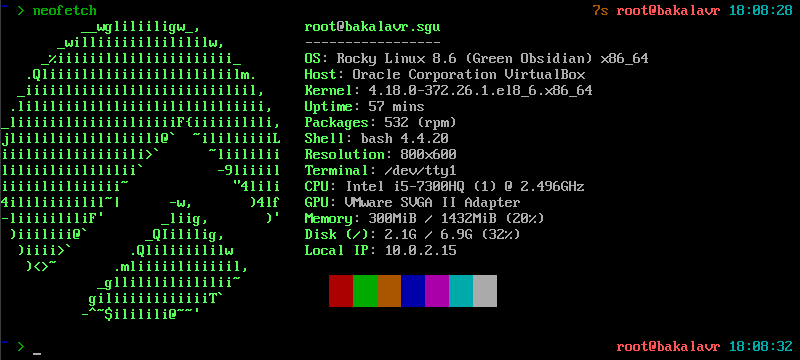


Так же можете изменить другие параметры под себя

Для сохранения Ctrl+O затем Enter

Для выхода из nano Ctrl+X

Снова переходим в neofetch и видим свою уникальную конфигурацию



**Выполнение работы**

-Установите neofetch

-Установите логотип вашей ОС в neofetch

-Настройте свое **УНИКАЛЬНОЕ** отображение neofetch

**Содержание отчета**

- Цель

- Основные этапы с иллюстрациями

- Скрин вашей **УНИКАЛЬНОЙ** конфигурации neofetch

- Выводы

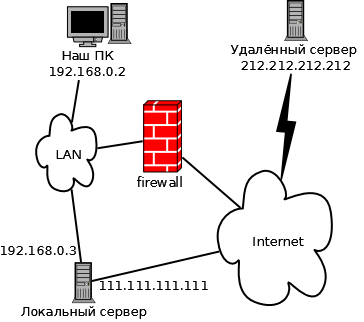
# ЛР №8. Сетевой Туннель

С помощью приложения PuTTY можно строить сетевые туннели, трафик по которым будет передаваться в зашифрованном виде.

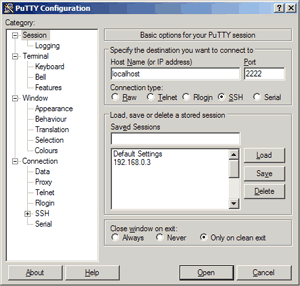
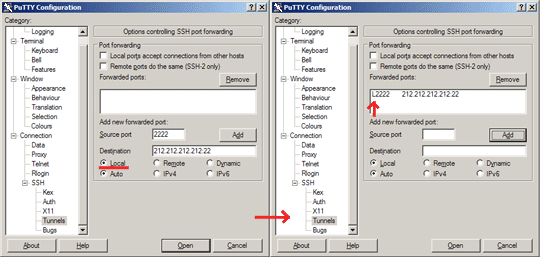
### 1. Локальный проброс порта

Рассмотрим следующую ситуацию. Мы находимся внутри корпоративной сети, у нашего компьютера адрес 192.168.0.2, доступ во внешний мир полностью закрыт (то есть никакого NAT-а, proxy и т.п.). Влиять на политику ограничения доступа у нас возможности нет, но зато есть SSH-доступ на один из серверов с маршрутизируемым IP-адресом, который доступен из Интернет. Внутренний адрес этого сервера, пусть будет для примера 192.168.0.3. Структура сети изображена на рисунке:

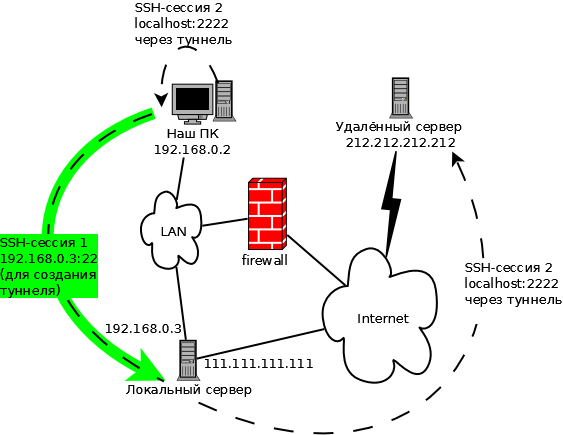
Предположим, что нам очень нужно подключиться, к примеру, по SSH на некоторый удалённый сервер с IP-адресом 212.212.212.212 где-то далеко в Интернет. Для этого запускаем PuTTY, создаём SSH-подключение к серверу 192.168.0.3 (далее по тексту SSH-сессия 1), идем в пункт Tunnels:



и указываем, что локальный порт 2222 нашего компьютера должен быть поставлен в соответствие порту 22 на сервере с IP-адресом 212.212.212.212. Далее жмем кнопку "Open", авторизуемся на сервере 192.168.0.3. Затем создаём ещё одно подключение (далее по тексту SSH-сессия 2), но уже на localhost, порт 2222 и жмём кнопку "Open":

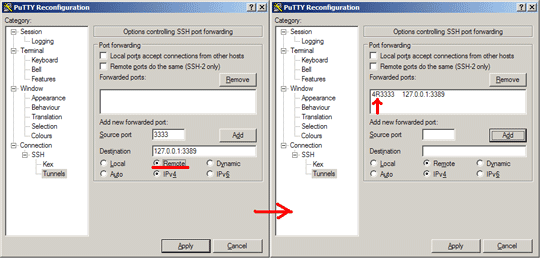
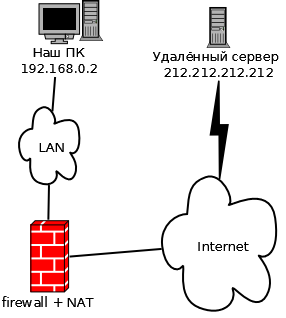


В результате SSH-сессия 2 будет туннелироваться (т.е. будет установлена внутри ранее установленной SSH-сессии 1). Для удалённого сервера 212.212.212.212 всё будет выглядеть так, как будто к нему подключается 111.111.111.111:



### 2. Удалённый проброс порта

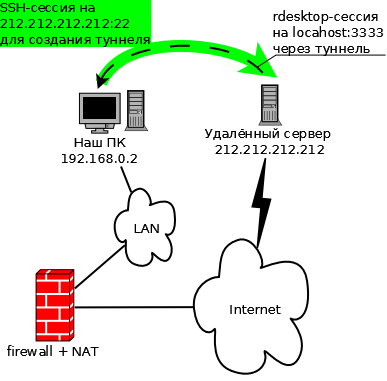
В этом случае подключение внутри SSH-туннеля устанавливается в другую сторону – от удаленного сервера на наш локальный компьютер. Может быть полезно, если требуется открыть доступ к локальным сервисам нашего компьютера. Рассмотрим ту же сеть, что и в пункте 1, но для простоты предположим, что теперь у нас есть NAT:  
  
Здесь уже у нас есть возможность подключаться через SSH напрямую к 212.212.212.212 благодаря наличию NAT-а. А вот 212.212.212.212 подключиться на 192.168.0.2 без специальных ухищрений, понятное дело, не сможет, т.к. 192.168.0.2 не подключен к Интернет непосредственно. Предположим, что пользователю, сидящему под X-ами на 212.212.212.212 нужно через remote desktop попасть на наш компьютер 192.168.0.2. Для этого в SSH-сеансе подключения с 192.168.0.2 на 212.212.212.212 нужно изменить настройки в разделе Tunnels следующим образом:  
  
В результате после успешной авторизации на 212.212.212.212 можно увидеть следующее:



*#lsof -i -nP | grep 3333*

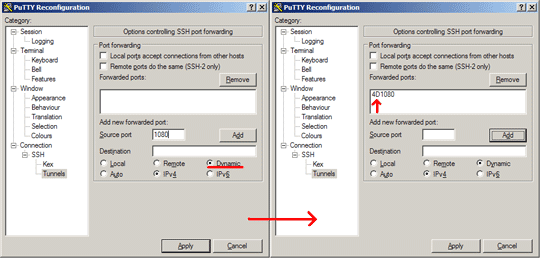
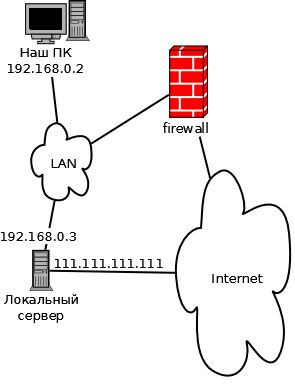
sshd 18598 avz 11u IPv4 592868957 TCP 127.0.0.1:3333 **(**LISTEN**)**

То есть sshd ожидает подключений на TCP-порт 3333, которые затем по SSH-туннелю будут перенаправлены на 192.168.0.2 порт 3389. И юзер сидящий за 212.212.212.212 сможет с помощью rdesktop увидеть наш рабочий стол:



### 3. Socks-proxy

В этом случае мы можем использовать сервер с SSH-демоном как промежуточный (proxy). Схема сети как в случае #1 (без NAT и штатных прокси):  
  
Чтобы заставить PuTTY испольнять роль socks-прокси, нужно параметры SSH-сессии с 192.168.0.2 на 192.168.0.3 изменить следующим образом:  
В результате после успешной авторизации со стороны клиента можно будет наблюдать следующее:



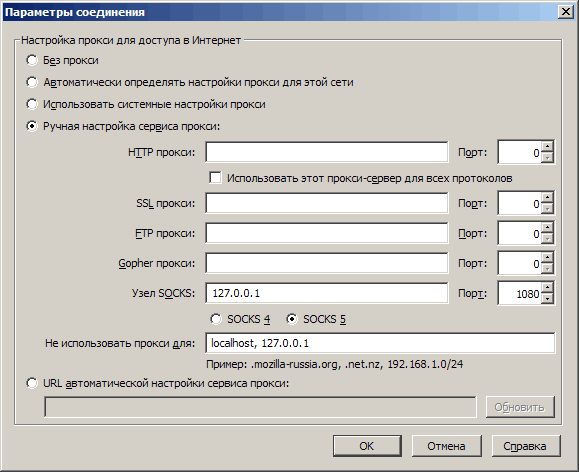
C:\**>netstat** -ano **|** **find** "1080"

TCP 127.0.0.1:1080 0.0.0.0:0 LISTENING 2392

C:\**>**tasklist **|** **find** **/**i "2392"

putty.exe 2392 Console 0 5420 КБ

То есть putty, выполняющийся с PID-ом 2392, начинает слушать порт 1080, ожидая подключений. Далее берем любое приложение, умеющее работать с SOCKS-прокси, например Firefox, и указываем ему использовать наш прокси:  
Теперь все запросы от браузера будут проходить через сервер 192.168.0.3. В логах веб-сайтов, по которым мы таким образом будем ходить, будет отображаться внешний IP-адрес нашего сервера - 111.111.111.111.



Выполнение работы

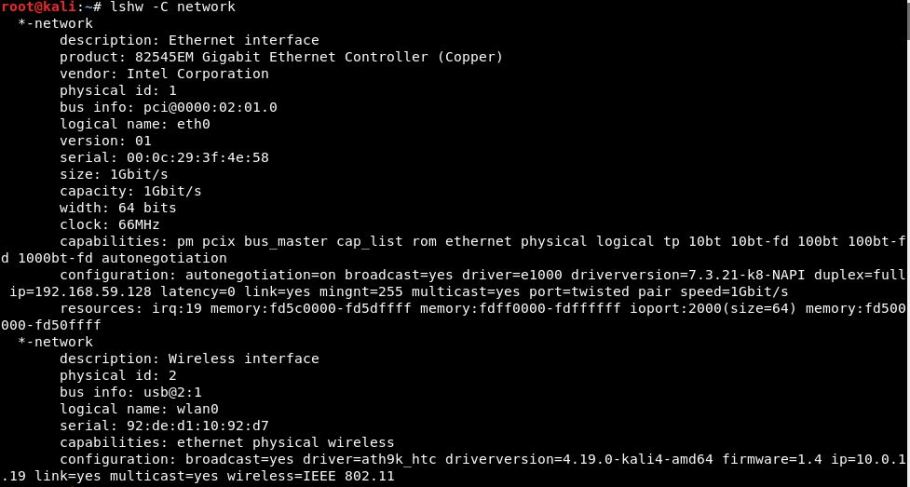
1. Определить свой адрес и местоположение с помощью специальных сервисов
   1. https://cyber-gateway.net/client-area/what-is-my-ip
   2. или любой другой по выбору
2. Реализовать туннель - подключение к серверу по методу (3) теоретической части. Порт выбрать любой в диапазоне 1000..65000.
3. Настроить на локальном компьютере браузер на работу через прокси: тип SOCKS, адрес – 127.0.0.1, порт – который мы выбрали.
4. Проверить работу через браузер в сети Интернет
5. Проверить, как сервера в интернете определяют наш адрес и местоположение, см.п.1.

# ЛР №9\*. Работа с сетью

Перед тем как работать с сетью, иногда (а лучше всегда) стоит узнать какая сетевая карта установлена в системе; например, решать вопросы совместимости довольно трудно не зная модель сетевой карты. Проще всего использовать для этого утилиту lshw. Установим её и запустим:

sudo yum install lshw

sudo lshw -class network



Если запустить её без дополнительных параметров, она выдаст подробную информацию о всех компонентах и устройствах компьютера.

В результате мы видим довольно подробную информацию об установленных сетевых адаптерах: производитель, модель, способ подключения, МАС-адрес, конфигурационные настройки и т.д.

### Ping

Суть работы этой утилиты заключается в отправке пакетов ICMP удалённому серверу, а после возврата пакета, **ping** подсчитывает время, которое понадобилось ему, чтобы дойти до сервера и вернуться.

ping drach.pro

В результате работы утилиты ping мы видим что:

* Мы можем установить соединение с сервером
* DNS работает корректно (имя запрашиваемого сервера было успешно преобразовано в ip)
* Соединение устойчиво, об этом говорит отсутствие пропусков в параметре icmp\_seq
* Значение time говорит нам, сколько времени понадобилось пакету, чтобы добраться до хоста и обратно (именно этот параметр часто называют скоростью пинга)

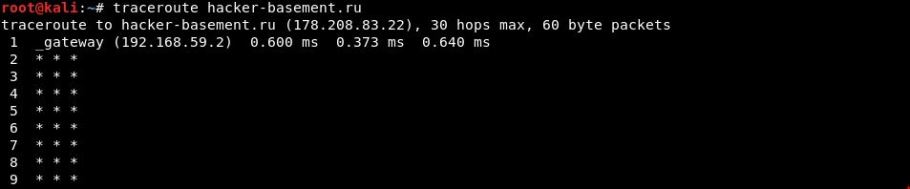
Пару слов о возможных неисправностях: если появилась ошибка «сеть недоступна», то можно попробовать определить на каком этапе возникает проблема. Для начала сделаем «пинг» на ip адрес роутера, если ответа нет, значит проблема на нашей стороне (обрыв кабеля, настройки системы, параметры адаптера и т.д.) Если роутер отозвался, можно послать запрос серверу провайдера, если ответ не получен, значит проблема в сетях провайдера. Если сервер провайдера отвечает, можно попробовать послать запрос сайту по IP, если ответ придет, значит некорректно работает DNS. Если всё это не помогло, то провести более детальный анализ можно с помощью утилиты traceroute.

### Traceroute

Каждый отправленный нами пакет, прежде чем достигнуть своей цели вынужден пройти некоторое количество узлов. При этом каждый пакет имеет, так называемое, время жизни — это количество узлов которое может пройти пакет, перед тем как будет сброшен. Это значение записывается в заголовке пакета, и уменьшается на 1 при прохождении каждого узла, если значение достигает 0 — пакет уничтожается, а отправитель получает сообщение что время превышено. Утилита traceroute отправляет на случайный порт по три UDP пакета со значением времени жизни сначала 1, потом 2, потом 3 и так пока не достигнет цели, при этом каждый раз замеряется время прохождения пакета. Трассировка считается оконченной, когда traceroute от целевого узла получает сообщение, что порт недоступен.

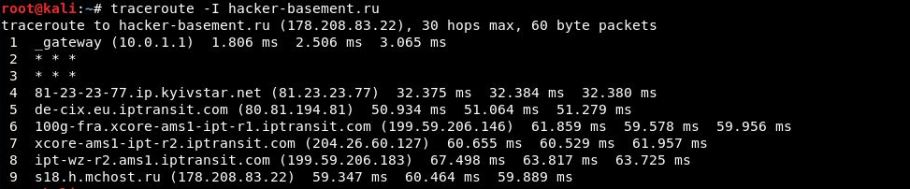
traceroute drach.pro

И очень часто в ответ видны лишь звёздочки:



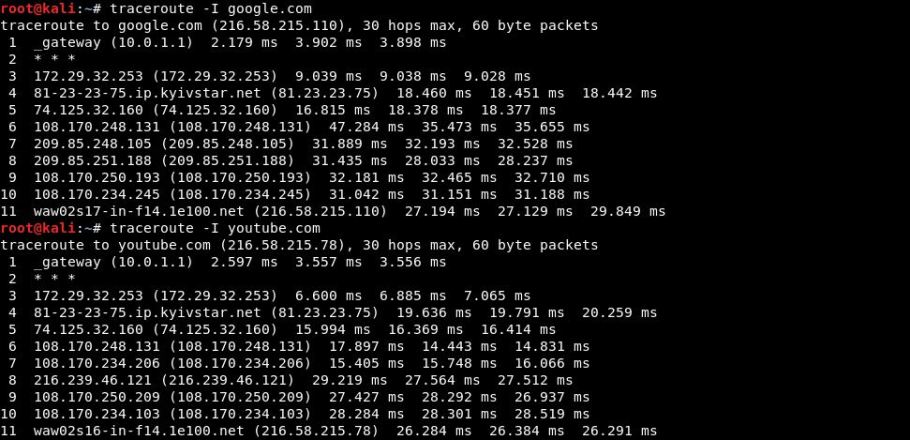
Это не значит, что всё пропало, это значит, что фаервол заблокировал все лишние пакеты, и соответственно трассировка UDP пакетами не работает, но сработает трассировка ICMP пакетами, чтобы её выполнить надо добавить параметр -I:

traceroute -I drach.pro



Теперь мы видим, что наши пакеты прошли через 8 узлов, перед тем как достигнуть цели. Также мы видим время, которое понадобилось каждому из трёх отправленных пакетов. Если бы на каком-то этапе возникла ошибка, мы бы увидели соответствующее сообщение.

В четвертой строке мы можем увидеть ip адрес нашего провайдера, а если сделать ещё парочку трассировок на другие адреса:

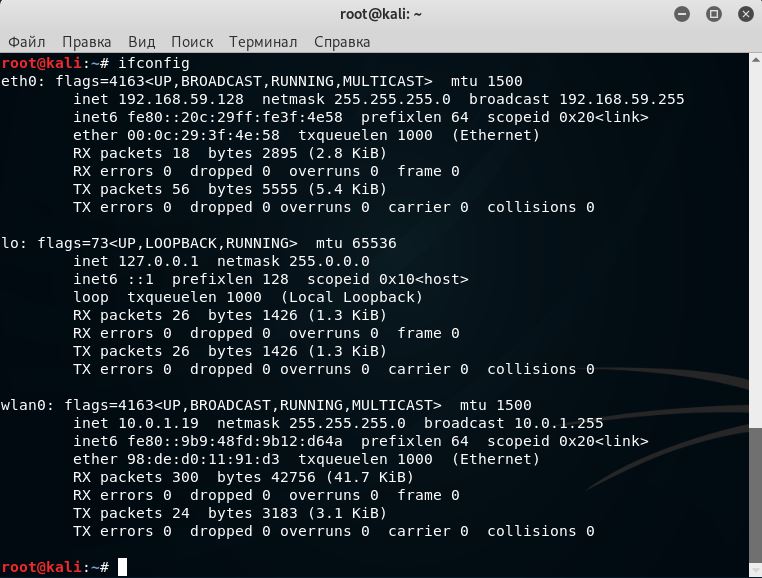


Можно прийти к выводу, что провайдер осуществляет доступ к глобальной сети из сети 81.23.23.0/24, а локальная сеть провайдера (к которой непосредственно подключен наш роутер) не очень охотно отвечает на сервисные запросы.

### Ifconfig

Чтобы получить общую информацию о сетевых интрефейсах, проще всего использовать утилиту:

Ifconfig



После ввода этой команды мы увидим на экране достаточно большое количество технической информации, распределенной на группы в зависимости от типа сетевого интерфейса:

* eth0 — проводное соединение;
* wlan0 — беспроводное соединение;
* lo — петлевой интерфейс т.е. интерфейс с обратной связью (используется компьютером для «общения с самим собой», он нужен, в основном, для диагностики, а также для подключения к серверам, работающим на локальном компьютере).

В зависимости от используемого дистрибутива название сетевых интерфейсов может отличаться, но смысл от этого не меняется

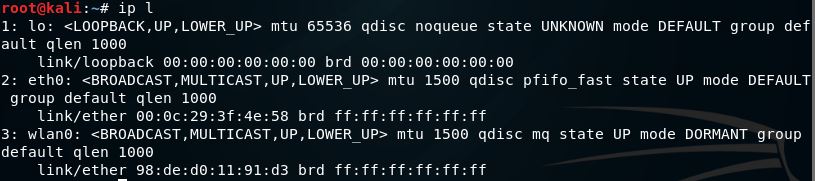
Что касается самой полученной информации, то из всего этого набора непонятных слов и символов стоит выделить наиболее необходимые сведения:

* RUNNING — означает, что интерфейс активен и пропускает трафик;
* BROADCAST — широковещательное соединение;
* inet — ip адрес протокола IPv4;
* netmask — маска подсети, это информация о том, какая часть IP-адреса узла сети относится к адресу сети, а какая — к адресу самого узла в этой сети;
* inet6 — ip адрес протокола IPv6;
* ether — уникальный адрес сетевой карты;
* RX/TX — активность сетевого интерфейса.

### IP

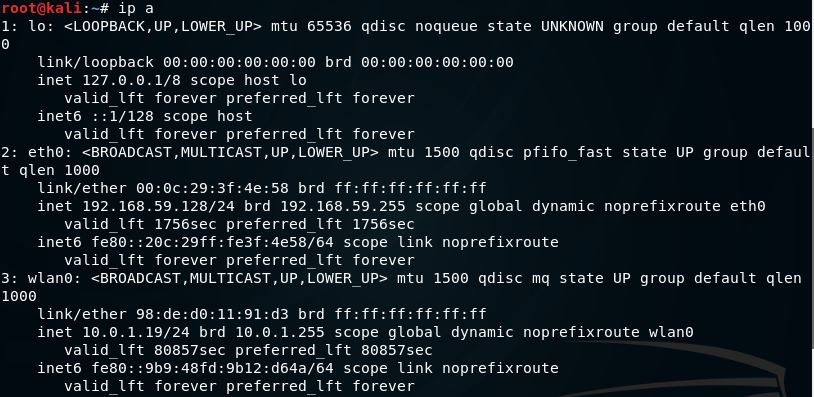
Помимо ifconfig общую информацию о сетевых интерфейсах можно получить использовав утилиту ip (более новая и в некоторых моментах более функциональная, чем ifconfig):

ip l



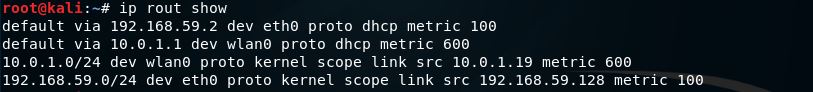
Чтобы получить более подробную информацию о сетевых интерфейсах, необходимо выполнить команду (сведения аналогичны полученным от ifconfig) :

ip a



Иногда возникает необходимость увидеть маршруты в таблице маршрутизации, для этого используется утилита ip:

ip rout show



Здесь мы видим ip нашего маршрутизатора, ip нашего компьютера и в какой подсети он находится.

Чтобы отключить какой-либо интерфейс, вводим команду:

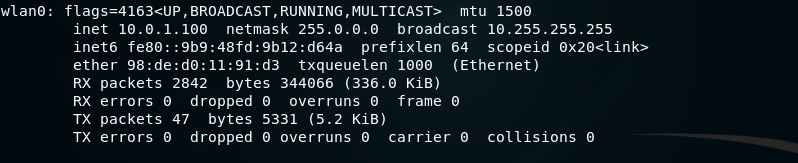
ip link set eth0 down

Чтобы включить:

ip link set eth0 up

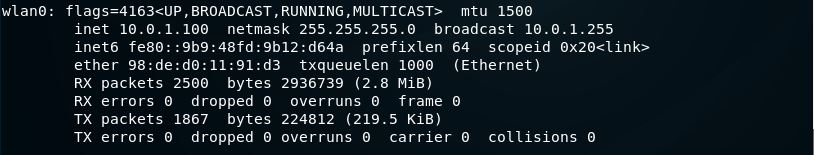
Если возникла необходимость сменить ip адрес, сделать это можно через утилиту ifconfig, указав какому интерфейсу, какой ip мы хотим назначить;

ifconfig wlan0 10.0.1.100



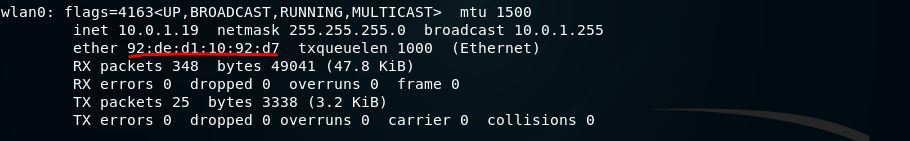
Также можно задать не только ip, но и подсеть и широковещательный адрес (можно задавать каждый параметр по отдельности):

ifconfig wlan0 10.0.1.100 netmask 255.255.255.0 broadcast 10.0.1.255



По такому же принципу можно изменить MAC-адрес нашей сетевой карты, что иногда бывает полезно с точки зрения «заметания следов» (например если не хочется оставлять улики в общественной wi-fi сети). Нужно предварительно отключить сетевой интерфейс и задать новый МАС, предварительно указав какому именно интерфейсу мы меняем МАС-адрес:

ifconfig wlan0 hw ether 92:de:d1:10:92:d7

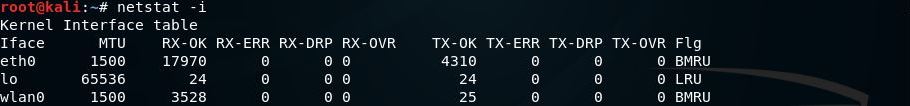


### Контроль состояния сети

Чтобы контролировать состояние сети, существует масса сторонних приложений (например NetHogs), но в самой системе для этого существует утилита netstat.

Для начала используем флаг -i, чтобы увидеть состояние наших сетевых интерфейсов, с данные счётчиков трафика:

netstat -i



В результате мы видим данные по каждому интерфейсу. Где RX — это общее количество пакетов полученных интерфейсом, а TX — это количество пакетов пропущенных через интерфейс с момента текущей активации. RX-ERR и TX-ERR — это количество ошибок, есть разные мнения о допустимом количество ошибок, но, как правило, если их меньше 5% от общего количества — это нормально, если больше значит, в большинстве случаев, есть проблемы с сетевой картой или с самим соединением.

Если ввести команду netstat без каких-либо дополнительных параметров, то мы увидим список процессов, для которых установлены сетевые соединения, с адресом, портом, типом соединения, службой и количеством запросов в очереди:



Для того чтобы определить какие именно процессы какие порты слушают и по какому протоколу, нужно использовать параметры lpn:

netstat -lpn



Чтобы получить информацию о таблице маршрутизации, необходимо использовать параметр r:

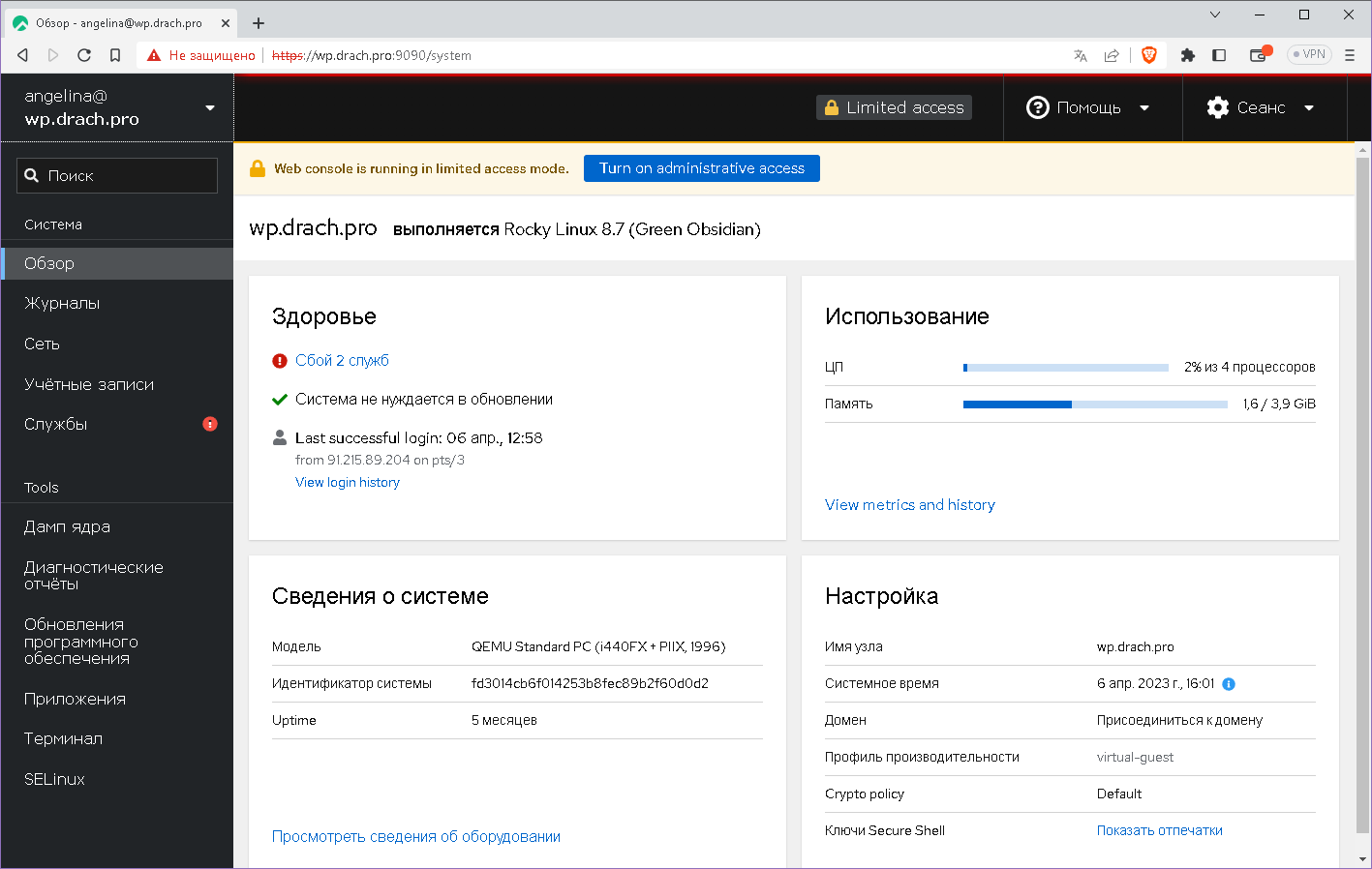
netstat -r



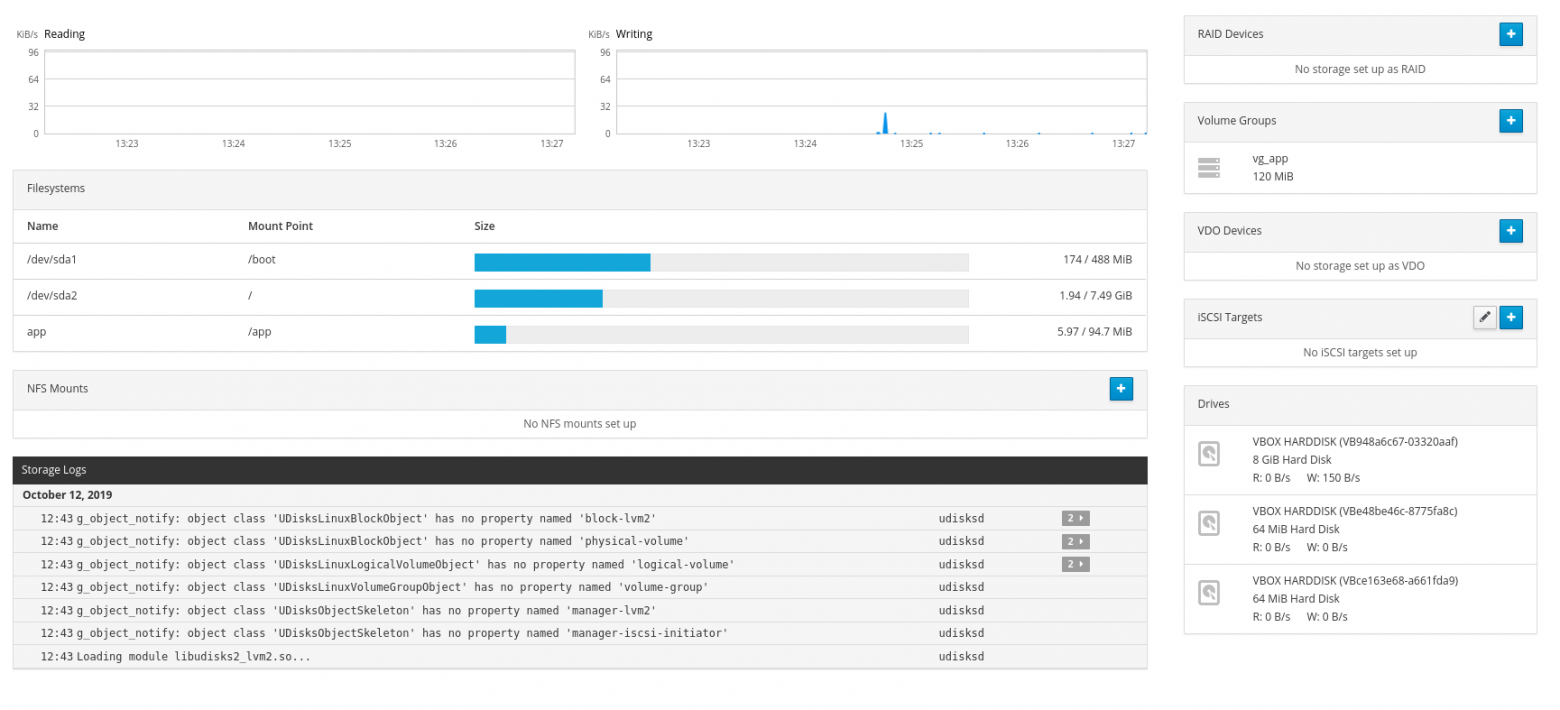
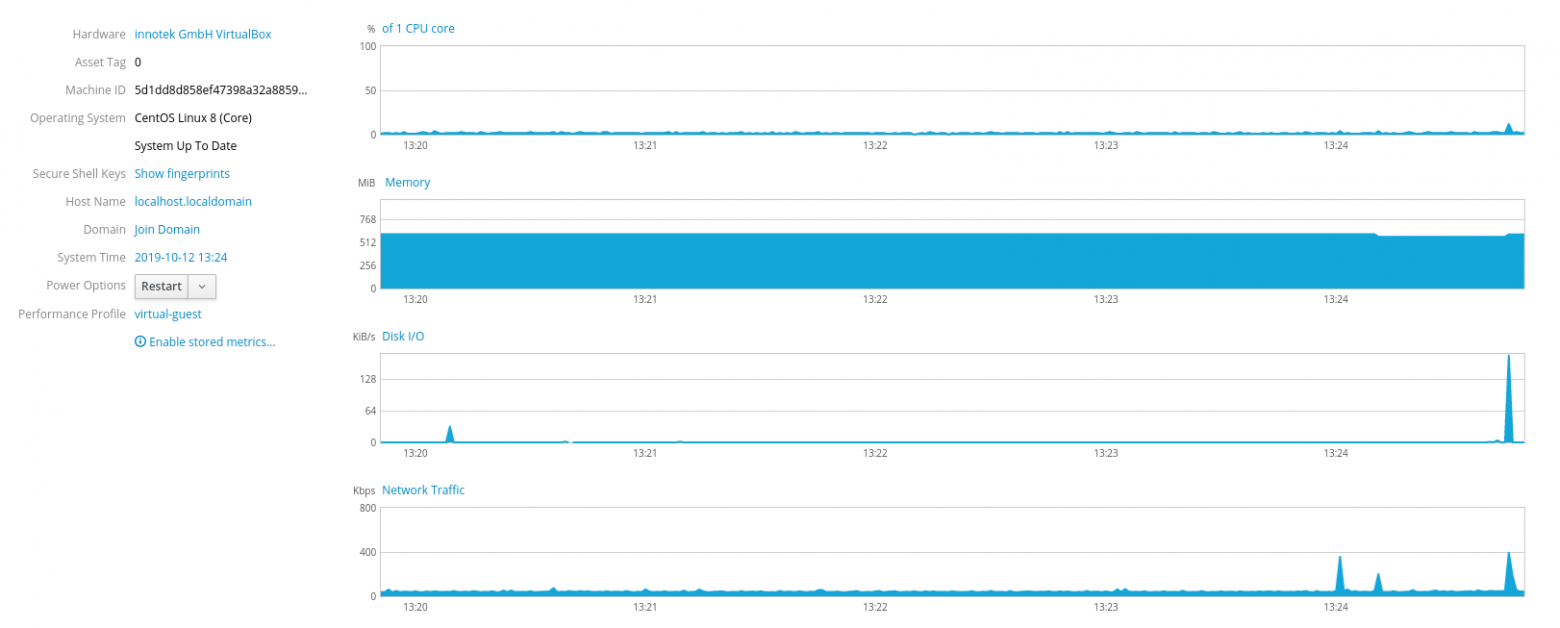
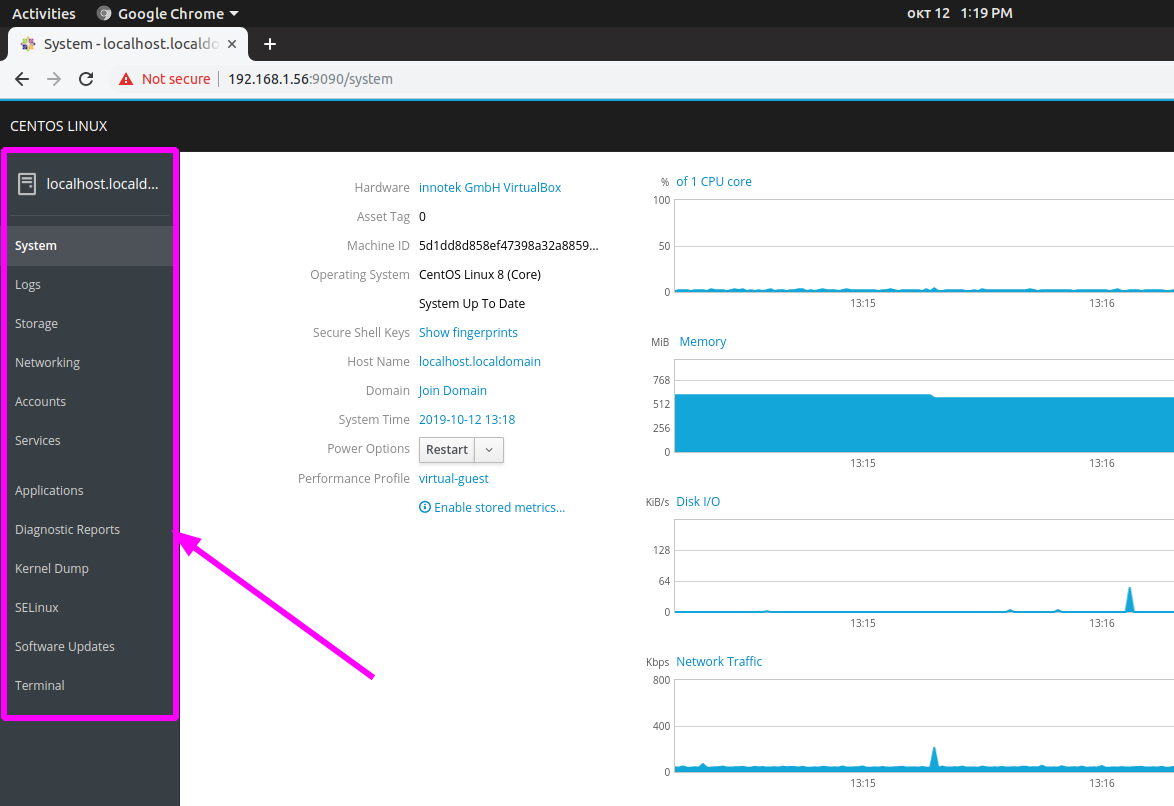
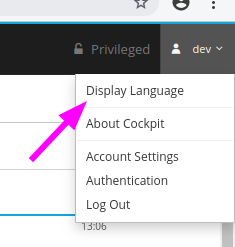
Перечисленных утилит вполне достаточно начинающему администратору для комфортной работы с сетью.

# ЛР №10. Управление через веб-интерфейс

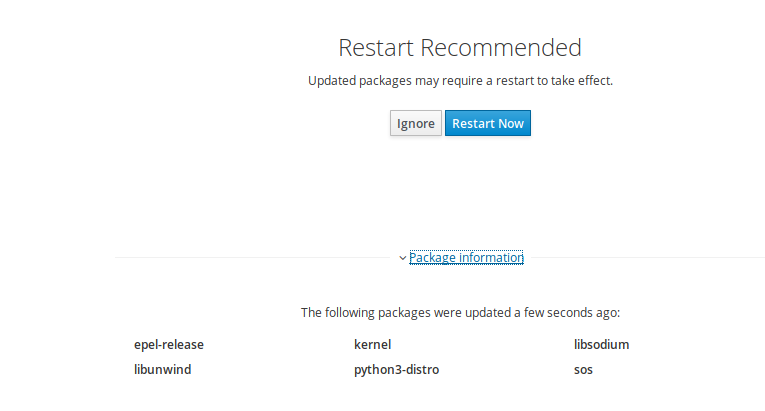
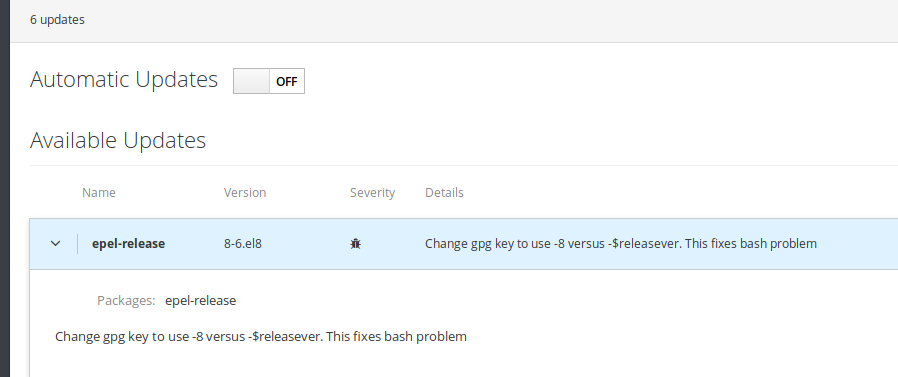
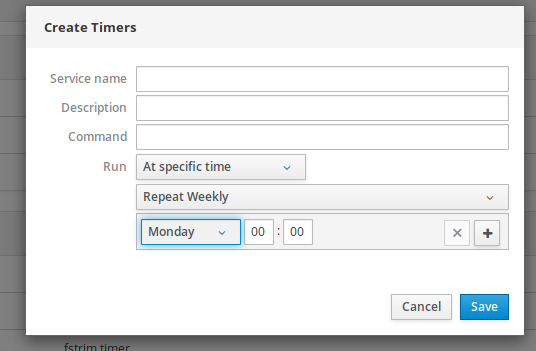
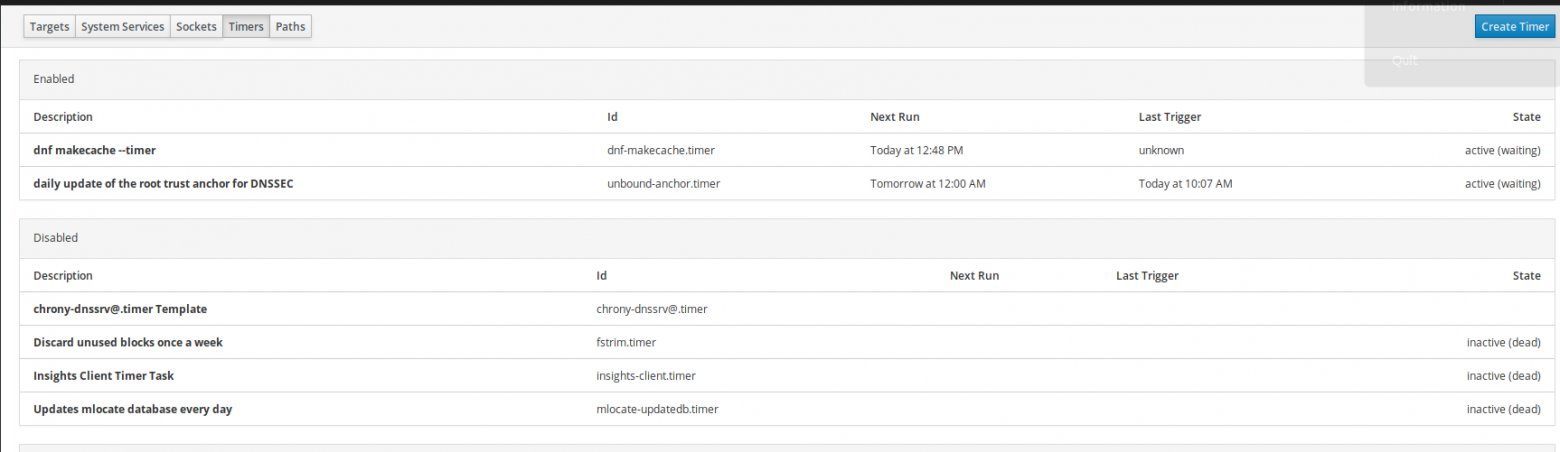
Cockpit создан с целью облегчения администрирования ОС Linux. Если говорить кратко, то он позволяет выполнять наиболее типичные задачи администратора Linux через приятный веб-интерфейс. Возможности Cockpit: установка и проверка обновление для системы и включение автообновления (процесс патчинга), управление пользователями (создание\удаление\смена паролей\блокировка\выдача прав суперпользователя), управление дисками (создание\редактирование lvm, создание\монтирования файловых систем), настройка сети (team, bonding, ip managing и т. д.), управление systemd-юнитами\таймерами.  
  
  
  
Начиная с Centos 8, где Cockpit уже встроен в систему и требуется его лишь активировать командой «systemctl enable --now cockpit.socket». На других дистрибутивах потребуется ручная установка из пакетного репозитория. Рассматривать тут установку мы не будем, смотрите [официальное руководство.](https://cockpit-project.org/running.html)



После установки нам необходимо в браузере перейти на 9090 порт сервера, на котором установлен Cockpit (т. е. ip-сервера:9090). Например, [192.168.1.56](https://192.168.1.56):9090  
  
Вводим обычный логин\пароль от локального аккаунта и ставим галочку «Reuse my password for privileged tasks», чтобы была возможность запускать некоторые команды от имени привилегированного пользователя (root). Естественно, ваш аккаунт должен обладать возможностью выполнять команды через sudo.  
  
После входа вы увидите красивый и понятный веб-интерфейс. Первым делом переключите язык интерфейса на английский, поскольку перевод выполнен просто ужасно.  
  
  
  
Интерфейс выглядит очень понятным и логичным, слева вы увидите навигационную панель:  
  
  
  
Стартовый раздел называется «system», где вы можете увидеть информацию по утилизации ресурсов сервера (CPU, RAM, Network, Disks):  
  
  
  
Чтобы посмотреть более подробную информацию, например, по дискам, просто нажмите на соответствующую надпись и вы попадёте сразу в другой раздел (storage):



В «services» можно управлять systemd юнитами и таймерами: останавливать их, перезапускать, убирать из автозагрузки. Так же очень быстро создать свой таймер:  
  
  
  
  
  
Единственное, что сделано плохо: не понятно, с какой периодичностью запускается таймер. Можно лишь посмотреть, когда он запускался в последний раз и когда запустится вновь.  
  
В «Software updates», как не сложно догадаться, можно посмотреть все доступные обновления и установить их:  
  
  
  
Система оповестит нас, если потребуется перезагрузка:



**Основные преимущества:**

* Порог вхождение в администрирование ОС Linux существенно снижается благодаря таким инструментам. Выполнять стандартные и базовые действия может практически любой человек. Администрирование можно частично делегировать разработчикам или аналитикам для удешевления производства и ускорения работы. Ведь теперь не надо набирать в консоли pvcreate, vgcreate, lvcreate, mkfs.xfs, создать точку монтирования, редактировать fstab и, наконец, набрать mount -a, достаточно кликнуть мышью пару раз
* Можно разгрузить администраторов Linux и избавить их от рутины, чтобы они фокусировались на более сложных задачах
* Можно уменьшить количество человеческих ошибок. Согласитесь, что через веб-интерфейс сделать ошибку сложнее, чем через консоль

### Выполнение работы

Подключиться к wp.drach.pro из браузера https://wp.drach.pro:9090/ со своим именем и паролем.

Собрать информацию о системе: какая установлена ОС, какое есть оборудование (память, процессор и т.п.)

Собрать информацию о здоровье сервера: загрузка, использование канала сети, использование памяти.

# 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