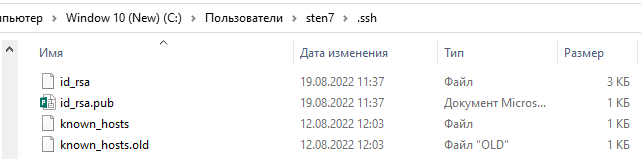
**Команды Linux ssh** [**xsp@telran.ab88.space**](mailto:xsp@telran.ab88.space) **telran23**  
[www.hostinger.com.ua/rukovodstva/osnovnyje-komandy-linux#Бонусные\_Советы\_и\_Приёмы](http://www.hostinger.com.ua/rukovodstva/osnovnyje-komandy-linux#Бонусные_Советы_и_Приёмы)

**Генерация ключей, подключение без пароля** (Ассиметричная криптография – почитать)

**1. ssh-keygen -t rsa  
2. ls -lah ~/.ssh –** просмотр ключей в папкеКлючи сохраняются в папке   
  
  
**3.** **ssh-copy-id Пользователь@АдресСервера (USER@SERVER)** и далее запросится пароль **telran23**4. И теперь подключаемся к серверу без пароля, т.к. мы обменялись ключами   
**ssh** [**xsp@telran.ab88.space**](mailto:xsp@telran.ab88.space)

**Учебный сервер  
1. cat .ssh/id\_rsa.pub -** просмотр публичного/открытого ключа в локальной папке, выделяем его и копируем в буфер2. Открываем список ключей на сервере, для того чтобы его добавить в файл ключей **nano ~/.ssh/authorized\_keys –** просмотр (+добавление) ключей на сервере в редакторе.  
3. После этого можем заходить на сервер без пароля, т.к. ключ на сервере  
**ssh** [**ec2-user@linux.telran-edu.de**](mailto:ec2-user@linux.telran-edu.de) **–** доступ на учебный сервер

**Ctrl+C или Ctrl+D** – прерывание процесса

**~** означает находимся в домашней директории   
  
**3 способа проверить, в полном или укороченном терминале мы находимся: bash, ash, sh**1. Tab и подсвечиваются команды  
2.   
3. cat /etc/release  
  
**cd** переход в домашнюю директорию из любого места **cd \** - переход в корневую дирректорию **exit** или **q** – выход   
**clear** – очистить экран **ctrl +c** – прервать процесс  
**команда -- help** - выводит на экран мануал по команде,  **-h** git- кратко опции по команде  
**man команда** – ???

**cat /etc/release** – посмотреть информацию о дистрибутиве **htop** или **top** - нагрузка сервера  
**id –**информация кто я  
**whoami** – краткая информация кто я **w -** кто подключен к серверу   
**du -hs –** просмотр занимаемого места (по умолчанию текущий каталог)  
**df –h** свободное место на дисках (h – человеческий вид)  
**pwd** – домашнее директория, где находимся   
**file –s файл** – посмотреть тип/формат файла  
**ls –** список папок и файлов -**l** c полными сведениями (права, дата, размер) **-la** со скрытими файлами   
**ls –lh /папка или файл** - полная информация с размерами по папке (файлу)  
**mkdir** создать каталог. Перечислив каталоги через пробел - создаем сразу несколько каталогов **mkdir –p** создать подкаталог, если катлоги выше не созданы  
**rm –rf** удалить текущий каталог со всем содержимым  
**rm файл** - удаляем файл   
**cp** – копировать файл  
**mv –** переименовать файл (указав каталог можно с переносом файла в другой каталог)  
**touch** или **>** создать пустой файл   
**echo текст > файл**  - записывает в файл текст, который укажем после echo   
**cat > файл** – записывает в файл текст, который будем вводить с клавиатуры  
**cat файл** – выводим содержимое файла на экран  
**.. | tail –количество** (выводит указанное количество строк с конца, по умолчанию 10) **.. | head –количество** (выводит указанное количество строк с начала, по умолчанию 10)  
.. **| head –кол | tail –кол** – можно комбинировать и использовать вместе (например вывести первые 100 строк и из них выбрать нужное) **cat /etc/\*release\*** - вывести на экран информацию о ОС  
**cat ps -f** - вывести на экран информацию о процесах  
**>>** дозаписать в файл  
**.. | wc** – счетчик (слева направо) слов (**-w**), строк (**-l**), символов (**-m**), байт (**-c**), длину самой длиной строки (**-L**)  
**grep** **слово** – выводит всю строку с найденным словом  
**grep – w** выводит найденные слова, сколько их нашел, без др. данных в этих строках  
**grep –o** выводит только найденное слово один раз **awk ‘{print $номерстолбца}’ –** вывод данных из указанного столбца  
**sed ‘s/%/$/g’ –** заменяет символ % на $ только в выбранных данных или глобально **(g)**   
**df –h | grep –w / | awk’{print $5}’** - сначала в информации df –h ищем строку с символом / и в этой строке показываем только 5й столбец. Т.е. элемент из нужной строки и нужного столбца  
**tar -cf /папка/arhiv.tar файл1 файл2** – упаковщик (cоздаем архив, но он не сжат.  
**tar -czf /папка/arhiv.tar.gz файл1 файл2 – упаковывает и сжимает  
tar –xzf /папка/arhiv.tar.gz –С /папка\_куда\_разархивирует  
gzip файлы ключ**–сжимает файлы, можно использовать ключ **–цифра от 1 до 9**, что означает степень сжатия  
**gunzip** - разархивирует файлы  
  
**scp путь/имя\_архива USER@SERVER**: - копируем архив с локального компа на сервер  
**scp USER@SERVER**:**путь/файл** - копируем файл сервера на компьютер

**vi** **файл**– редактор. В нем **i** – режим ввода текста, **esc** - выход из режима команд,   
**клавиша D** – удаление строки  
 **Shift + : + q!**– выйти без сохранения  
 **w** **файл**- записать в файл, если запустили **vi** без имени файла  
 **wq** или **x** - again, save and exit **G** (shift g) - move to the last line **/** - search (n - next) **i** - insert text, edit mode. ECS to exit **u** - undo last action **dd** - delete the current line ( i mode not needed)  **Del** - delete single character ( i mode not needed)  
4 выхода из vi: без сохранения - 1:q! с сохранением- 2:wq 3 Shift+z+z 4:x **nano –** редактор, более удобный чем vi. **less файл** – просмотр содержимого файла  
 **Переменные  
env** – просмотр стандартных переменных, переменные пишутся с большой буквы  
**PATH** – переменная для переадресации, помогает выполнить команды, расположенные в bin и   
 sbin, независимо от того, в какой папке находится пользователь  
**export NAME=Stas** – создание переменной NAME со значением Stas  
**echo $NAME** – выводим на экран значение переменной (можно несколько переменных). Необходимо использовать именно echo, а не cat. **$NAME** – вызов переменной **unset NAME**  - удаляем переменную  
Если вдруг удалили PATH, то можно запустить export и создать переменную PATH  
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin   
 **.** – наша локальная папка  
**..** – папка на уровень выше  
**drwxrwxrwx**  
d/l/c – директория/линк(перенаправление)/системный файл   
1й сегмент 3символа (u) – владелец файла/директории (кто создал или кому назначили)2й сегмент 3символа (g) – права группы владельцев  
3й сегмент 3символа (o) – права всех остальных **Права на файлы/папки  
Способ1 - буквенный  
chmod ugo+/-rwx файл** – дает/забирает права файлу. Если вместо ugo пишем пробел, то оно по умолчанию ugo   
 u-user, g-group, o-other  
 r-read, w-write (создание/изменение), x-execute,   
 t-стикибит (особые права на директорию-позволяют удалить файл только создателем)  
 s - наследование группы на созданных в папке файлов   
Если перезапускам терминал, то переменные пропадают. Их необходимо записать в специальные файлы

**~./ash\_profile (или bash)** – сюда записываем наши переменные  
**зеленый цвет у файла** – это исполняемый файл (скрипт)  
**синий цвет** – папка  
 **Способ2 - цифровой**0 – нет прав 4 – только чтение 5 – запись 7- все права  
 **Скрипты**   
**nano script.sh** – открываем редактором файл для написания скрипта и построчно записываем команды  
#!/bin/bash   
echo hello world  
date  
#! – Шебанг, указывает что интерпретатору необходимо запустить bash   
  
**Вызов скрипта - варианты**1) **./имяскрипта.sh** – если запускаем скрипт, находясь в текущей папке со скриптом  
2) **/tmp/ имяскрипта.sh** – запуск скрипта из любого места, указывая путь, где находится скрипт  
3) **bash script** – это если в скрипте нет первой строчки #!/bin/bash  
Скрипт запускается также и на локальном. Можно использовать другие интерпретаторы, например  
python file.py и т.д.  
4) поместить наш скрипт в одну из папок /bin, usr/bin, usr/local/bin  
  
**Примеры скрипта**

|  |  |  |  |
| --- | --- | --- | --- |
| #!/bin/bash for run in {1..10} **do**  echo $run  sleep 1 **done** | #!/bin/bash echo –n “Enter your name:” read NAME echo Hello $NAME  (-n нет переноса в строчке, переменная живет только внутри скрипта) | #!/bin/bash for var in $(cat $file)  do echo “$var”  done | #!/bin/bash for ((n=1; n<10; n++))  do  echo”$var”  done |

Делаем цикл отработки операции, все что между do и done. sleep – это задержка для визуально  
  
**Папки  
cd /opt/35\_36\_morning\_b/st\_tonkich –** моя рабочая папка  
**cd /opt/old\_opt/chumak –** Ивана Чумака **Работа с сервером** (пароль telran23)  
**ssh** [**xsp@telran.ab88.space**](mailto:xsp@telran.ab88.space) **ls (папка) –** листинг домашнего каталога (или папки) без захода на сервер  
**ssh** [**xsp@telran.ab88.space**](mailto:xsp@telran.ab88.space) **bash(python) /папка/script.sh –** выполняем скрипт на сервере, не заходя туда, вместо bash можно использовать интерпретаторы python3, php и т.д. **scp файл user@server:папка –** копируем файл с локальной папки на сервер

**Cron**

**ssh**[**ec2-user@linux.telran-edu.de**](mailto:ec2-user@linux.telran-edu.de)

**crontab –l -**смотрим, какие задачи находятся в планировщике  
**crontab –r -**удаляем задачи, находящиеся в планировщике  
**crontab –e -**редактор планировщика

На сайте **crontab.guru** смотрим параметры интервалов для планировщика  
  
1. touch /tmp/script\_tonkich.sh /tmp/script\_out.txt  
2. echo –e ‘#!/bin/bash\n date\n echo “it works!”’ >> /tmp/script\_tonkich.sh