Bash команды

Редактирование командной строки

Ctrl+A в начало строки
Ctrl+E в конец строки
Ctrl+U удаление всей строки перед курсором
Ctrl+K удаление всей строки после курсора
Ctrl+L очищает экран от предыдущих команд (подобно clear)
Alt+. предыдущий аргумент (из history). Не предыдущая команда!

!! Предыдущая команда

Выполнить предыдущую команду (!!)

без sudo

```
rus@rus-400b4b:~/shells/dir/август/зарплата$ fdisk -l
fdisk: невозможно открыть /dev/ram0: Отказано в доступе
```

с sudo но не вводя повторно предыдущую команду без sudo

```
rus@rus-400b4b:~/shells/dir/август/зарплата$ sudo !!
sudo fdisk -l
sudo: unable to resolve host rus-400b4b
[sudo] пароль для rus:
Диск /dev/ram0: 64 MiB, 67108864 байтов, 131072 секторов
Единицы измерения: секторов из 1 * 512 = 512 байтов
Размер сектора (логический/физический): 512 байт / 4096 байт
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```

hystory

Показать список введённых ранее команд (history)

```
rus@rus-400b4b:~/shells/dir/август/зарплата$ history
624  cd август/
625  ls
626  cd зарплата/
627  ls
628  history
```

выполнить команду № 625 (!625)

```
rus@rus-400b4b:~/shells/dir/август/зарплата$ !625
ls
иванов.txt  петров.txt  сидоров.txt
```

Ctrl+R поиск по history

(reverse-i-search)`':
(reverse-i-search)`l': ls

Grep чтение строк из текстового файла

Чтение строк из текстовых файлов

например строка в файле passwd, в котором есть слово root:

```
rus@rus-400b4b:~$ grep 'root' /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

или так:

```
rus@rus-400b4b:~$ grep ^root /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

а можно и так:

```
rus@rus-400b4b:~$ cat /etc/passwd | grep ^root
root:x:0:0:root:/root:/bin/bash
```

Процесс в фоне &

Запуск процесса в фоне и возможность продолжать работать в консоли (открепление процесса от данного терминала)

```
rus@rus-400b4b:~/shells$ firefox &
[1] 2849
rus@rus-400b4b:~/shells$
```

Имена файлов

Если нам необходимо задать имя файла, например -a, то чтобы команда touch не думала, что -a это опция, указываем перед именем создаваемого файла два минуса --

например:

touch -- -a

два минуса после команды означает, что опций больше нет

Поиск файла

Поиск файлов по маске

sudo find / -name "pdb.??c"
sudo find / -name "pdb.d*"
sudo find / -name "pdb.*"
sudo find / -name "*.doc"

locate имя файла
sudo updatedb

например:

```
test@test-VirtualBox:~$ locate arifm
/home/test/.arifm.sh.swp
/home/test/arifm.sh
/home/test/mnogo2/arifm.sh
```

whereis поиск положения исполняемых файлов, исходников и man-страниц

например:

```
test@test-VirtualBox:~$ whereis nano
nano: /bin/nano /usr/share/nano /usr/share/man/man1/nano.1.gz /usr/share/info/
nano.info.gz
```

whereis имя программы - поиск и показ расположения установленных программ

which - поиск программ

Постраничный вывод

например:

ls -al /etc/ | less

клавиши управления:

[пробел] переход на сл.страницу
[B]
[Q]
[↑↓] построковое перемещение вверх/вниз

Тоже самое делает команда more (она более примитивная)

Можно использовать less для других команд, например:

mv -iv * ../newdir | less

перемещение всех файлов в тек.каталоге с постраничным выводом сообщений процесса

Перенаправление вывода

./script.sh > sysinfo_page.html

Операции с файлами и каталогами

- Создание файла
- Определить тип файла
- Переход по каталогам
- Копирование файла или каталога
- Копирование каталогов
- Создание копии каталога
- Удаление файла (с вопросами y/n)
- Удаление каталога и его содержимого:
- Переименование/перенос файла или каталога
- Просмотр содержимого каталога ls
- Создание каталога, название которого с пробелами mkdir

Создание файла

touch filename
cat > имя нового файла, нажимаем Enter, вводим любой текст. Чтобы выйти нажимаем Ctrl-q
nano (или vi) filename

создать несколько файлов в нескольких каталогах и подкаталогах

например:

```
rus@rus-400b4b:~/shells/dir$ touch {июнь,июль,август}/{отчёт,резюме,зарплата}/
{иванов.txt,петров.txt,сидоров.txt}
```

создать несколько файлов одной командой

вариант №1

```
rus@rus-400b4b:~/shells/long long$ touch file1 file2 file3
rus@rus-400b4b:~/shells/long long$ ls
file1  file2  file3
```

вариант №2

```
rus@rus-400b4b:~/shells/long long$ touch file{1,2,3}
rus@rus-400b4b:~/shells/long long$ ls
file1  file2  file3
```

вариант №3 файл с расширением .txt

```
rus@rus-400b4b:~/shells/long long$ touch file{1,2,3}.txt
rus@rus-400b4b:~/shells/long long$ ls
file1.txt  file2.txt  file3.txt
```

Определить тип файла

file filename

например:

```
test@test-VirtualBox:~$ file arifm.sh
arifm.sh: Bourne-Again shell script, UTF-8 Unicode text executable
```

Просмотр файла

cat filename

Переход по каталогам

cd dirname

cd .. переход на уровень вверх

cd ../.. переход на два уровня вверх

cd ~ возврат в домашний каталог

cd - (минус) вернуться в предыдущий каталог

pwd абсолютный путь к текущему каталогу

Копирование файла или каталога

cp -i filename (dirname)

например:

cp -i arifm.sh mnogo/

или с перечислением нескольких файлов:

cp -i arifm.sh timeout.sh randpass.sh usname.sh memo.sh mnogo/

копирование всех файлов в текущем каталоге

cp -i * имя каталога

например:

cp -i * /home/test/mnogo

копирование рекурсивно с сохранением атрибутов файлов и папок

cp -pr * /media/user/Cop/

Копирование каталогов

cp -ir dirname

Создание копии каталога

cp -ir dirname1 dirname2

Удаление файла (с вопросами y/n)

rm -i имя файла

rm -i * удаление всех файлов в текущем каталоге

Удаление каталога и его содержимого:

без вопросов

rm -r имя каталога

с вопросами

rm -ir имя каталога

rmdir имя каталога

Переименование/перенос файла или каталога

mv -i имя файла новое имя файла

например:

mv -i filename1 filename2

переименование каталога

mv -i oldname newname

например:

mv -i dirname1 dirname2

Просмотр содержимого каталога ls

например, просмотр файлов, начинающихся с буквы t (по маске *):

ls t*

```
rus@rus-400b4b:~/shells$ ls t*
tcicle.sh  testx.sh
```

или:

```
rus@rus-400b4b:~/Изображения/Фото$ ls -l *[.mp4]
-rw-r--r-- 1 rus rus    628894 окт 28 06:47 20160701_134732.mp4
-rw-r--r-- 1 rus rus  23665757 окт 28 06:48 20160701_141409.mp4
-rw-r--r-- 1 rus rus  48435552 окт 28 06:48 20160701_142843.mp4
```

ещё

```
rus@rus-400b4b:~/Изображения/Фото$ ls -l *[^4]
-rw-r--r-- 1 rus rus 2968934 окт 28 06:47 20160630_161318.jpg
-rw-r--r-- 1 rus rus 3626570 окт 28 06:47 20160701_105141.jpg
```

ещё вариант маски, показ файлов, начинающихся с K и имеющими в расширении окончание g:

```
rus@rus-400b4b:~/Изображения/CCCP$ ls K*.*[g]
KaJya8EMrTI.jpg  KV8kP1jTEJQ.jpg
```

постраничный вывод содержимого каталога

ls -a | less # управление: пробел - переход на сл.страницу, B - возврат к предыдущей, Q - выйти, стрелки вверх,вниз - постоковый просмотр.

Создание каталога, название которого с пробелами mkdir

mkdir ‘long long’

создание нескольких каталогов и вложенных каталогов

например:

```
rus@rus-400b4b:~/shells/dir$ mkdir -p {июнь,июль,август}/{отчёт,резюме,зарплата}
```

результат:

```
rus@rus-400b4b:~/shells/dir$ ls [иа]*
август:
зарплата  отчёт  резюме

июль:
зарплата  отчёт  резюме

июнь:
зарплата  отчёт  резюме
```

разъяснение: {основные каталоги}/{вложенные каталоги}, [иа] Август,Июнь,Июль

Копирование с терминала

https://losst.ru/kopirovanie-fajlov-v-linux

Синтаксис и опции

Общий синтаксис cp выглядит вот так:

$ cp опции файл-источник файл-приемник

Или:

$ cp опции файл-источник директория-приемник/

После выполнения команды файл-источник будет полностью перенесен в файл-приемник. Если в конце указан слэш, файл будет записан в заданную директорию с оригинальным именем.

Утилита имеет несколько интересных опций, которые могут сильно помочь при нестандартных задачах копирования, поэтому давайте их рассмотрим:

—attributes-only — не копировать содержимое файла, а только флаги доступа и владельца;
-f, —force — перезаписывать существующие файлы;
-i, —interactive — спрашивать, нужно ли перезаписывать существующие файлы;
-L — копировать несимволические ссылки, а то на что они указывают;
-n — не перезаписывать существующие файлы;
-P — не следовать символическим ссылкам;
-r — копировать папку linux рекурсивно;
-s — не выполнять копирование файлов в linux, а создавать символические ссылки;
-u — скопировать файл, только если он был изменен;
-x — не выходить за пределы этой файловой системы;
-p — сохранять владельца, временные метки и флаги доступа при копировании;
-t — считать файл-приемник директорией и копировать файл-источник в эту директорию.

Примеры копирования файлов в linux

Теперь, когда вы знаете основные опции можно рассмотреть примеры.Например, мы хотим скопировать некую картинку из домашней папки в подкаталог pictures:

cp ~/pic.png ~/pictures/

Или можем явно указать имя новой картинки:

cp ~/pic.png ~/pictures/wallpaper.png

Копирование папок осуществляется с помощью ключа -r:

cp -R ~/папка ~/Документы/

После выполнения этой команды копирования ~/папка будет скопирована в папку ~/Документы. Главное, не забывайте поставить слэш в конце выражения или использовать опцию -t. Иначе папка ~/документы будет перезаписана.

По умолчанию команда cp linux перезаписывает существующие файлы или папки, но можно заставить утилиту спрашивать нужно ли перезаписывать каждый файл если вы неуверены в правильности составления команды:

cp -i ~/test ~/Documents/test

Есть и противоположная опция -n, означающая никогда не перезаписывать существующие файлы.

Опция -u позволяет копировать файл, только если уже существующий более старой версии, чем новый:

cp -u ~/test ~/Documents/test

cp также поддерживает специальные символы замены * и ?. Например, следующая команда скопирует все файлы, начинающиеся на test:

cp ~/test* ~/Документы/

Если нужно применить более сложные регулярные выражения, придется комбинировать утилиту cp с find или egrep.

В случае если важно сохранить права доступа к файлу и его владельца нужно использовать опцию -p:

cp -p ~/test* ~/Документы/

Для упрощения использования команды можно использовать синтаксис фигурных скобок. Например, чтобы создать резервную копию файла выполните:

$ cp test.conf{,.bak}

Будет создан файл с таким же именем и расширением .bak

По умолчанию в cp не отображается прогресс копирования файла, что очень неудобно при работе с большими файлами, но его можно легко посмотреть с помощью утилиты cv.
Копирование файлов по регулярным выражениям в Linux

В утилите find можно применить различные условия и регулярные выражения для поиска файлов. Я уже немного писал о ней в статье как найти новые файлы в Linux. Мы можем скопировать все найденные с помощью find файлы вызвав для каждого из них команду cp. Например, копируем все файлы в

текущей директории содержащие в имени только цифры:

find . -name [0-9] -exec cp {} ~/Документы \

Здесь точка указывает на текущую директорию, а параметр name задает регулярное выражение. Параметром exec мы задаем какую команду нужно выполнить для обнаруженных файлов. Символ {} — подставляет имя каждого файла.

Но не find'ом единым можно такое делать. То же самое можно получить, запросив список файлов директории в ls, отфильтровав его по регулярному выражению egrep и передав имена файлов по очереди в cp с помощью xargs:

ls -1 ~/ | egrep '[a-zA-Z]' | xargs cp -t ~/Папка/

Это достаточно неудобный способ копировать файлы linux но все же, он возможен. Будут скопированы все файлы из домашней директории, содержащие в имени только английские буквы.
Копирование содержимого файлов в Linux

Вы можете копировать не только сами файлы, они управлять их содержимым. Например, склеить несколько файлов в один или разрезать файл на несколько частей. Утилита cat используется для вывода содержимого файла, в комбинации с операторами перенаправления вывода Bash вы можете выполнять копирование содержимого файла linux в другой файл. Например:

cat файл1 > файл2

Если файл был не пустым, он будет перезаписан. Или мы можем склеить два отдельных файла в один:

cat файл1 файл2 > файл3

Специальное копирование файлов в Linux с помощью tar

Linux интересен тем, что позволяет выполнять одно и то же действие различными путями. Копирование в Linux тоже может быть выполнено не только с помощью cp. При переносе системных файлов в другой каталог, резервном копировании системных файлов и т д важно чтобы сохранились атрибуты, значения владельцев файлов и символические ссылки как они есть без какой-либо модификации.

Утилита cp тоже может справиться с такой задачей если указать опцию -p, но можно использовать утилиту архивации tar. Мы не будем создавать никаких файлов архивов, а построим туннель. Первая часть команды пакует файл и отправляет на стандартный вывод, а другая, сразу же распаковывает в нужную папку:

tar cf - /var | (cd /mnt/var && tar xvf -)

Здесь мы полностью копируем содержимое папки /var в папку /mnt/var. Так вы можете копировать папку linux, при чем абсолютно любую или даже целую

операционную систему.

Команды администрирования

Выключение по времени

sudo shutdown -h 00:00

Диск и монтирование

sudo fdisk -l разделы текущего жесткого диска

mount все смонтированные разделы

Права доступа к файлам

chmod +x имя файла

например:

```
test@test-VirtualBox:~$ chmod +x timeout.sh
```

Редактор VI

https://losst.ru/kak-polzovatsya-tekstovym-redaktorom-vim

Как пользоваться текстовым редактором vim

Опытные пользователи Linux часто используют терминал, потому что так можно намного быстрее выполнить необходимые действия. Во время настройки системы нам довольно часто приходится редактировать различные файлы. Это могут быть настройки программ, какие-нибудь данные или обычные текстовые файлы.

В операционной системе Linux есть несколько текстовых редакторов, которые работают в терминале. Чаще всего новички используют редактор nano, но если вы заметили на нашем сайте во всех статьях используется текстовый редактор vi. Nano неудобный, и недостаточно функционален. Я сознательно не пишу в своих статьях о nano. Есть намного лучший текстовый редактор, это редактор vi. Здесь поддерживается быстрое перемещение по тексту, удобное редактирование, команды для изменения настроек работы, выполнение команд терминала из редактора, а также плагины для расширения функциональности. Но он немного сложный для новичков и очень непривычный.

В этой статье мы рассмотрим как пользоваться vim, рассмотрим основы работы с этим редактором, а также его основные команды.
Минимальные основы

На данный момент существует две версии редактора - vi и vim. Vim расшифровывается как Vi Improved, улучшенный vi. Это новая версия, которая принесла очень много улучшений. В большинстве современных дистрибутивов используется именно она. Поэтому если я буду писать vi, это значит, что я предполагаю использование vim.

Текстовый редактор Vim может работать в двух режимах. Это и есть его главная особенность. Первый режим, который используется по умолчанию при открытии редактора - это командный. В этом режиме вы можете вводить команды vi, а также использовать символьные клавиши для управления редактором. Второй режим - обычное редактирование текста, он работает так же как и редактирование текста в nano. Для переключения в командный режим используется клавиша Esc. Для переключения в режим редактирования - клавиша i. Если вас интересует только как в редакторе vi сохранить и выйти, листайте вниз, но если вы хотите узнать как пользоваться текстовым редактором vim, эта статья для вас.

Перед тем как идти дальше я бы посоветовал вам пройти курс обучения встроенный в редакторе. Выполнение всех обучающих заданий займет 25-30 минут. Но после того как вы освоите все что там написано, эта статья поможет вам закрепить материал. Дело в том, что команд и сочетаний клавиш у vim очень много и запомнить их все без практики невозможно. Для запуска обучения наберите:
$ vimtutor

Но делать это сейчас необязательно, в этой статье есть вся необходимая базовая информация и после ее прочтения вы уже сможете уверенно пользоваться vim, а обучение пройти чуть позже.

Как использовать редактор Vim

Начнем мы, как обычно с запуска программы, а также опций, которые ей можно передать. Синтаксис Vim очень прост:

```
$ vim опции имя_файла
```

Или:

```
$ vi опции имя_файла
```

Простой запуск vim без указания имени файла приведет к созданию пустого файла. А теперь давайте рассмотрим основные опции запуска:

+номер - переместить курсор к указной строке после запуска.
+/шаблон - выполнить поиск по шаблону и переместить курсор к первому вхождению
"+команда" - выполнить команду после запуска программы
-b - двоичный режим, для редактирования исполняемых файлов.
-d - режим поиска различий в файлах, нужно указать несколько файлов для открытия.
-g - графический режим.
-n - не использовать автосохранение для восстановления файла при сбое.
-R - режим только для чтения.
-w - сохранить все действия в файл.
-x - шифровать файл при записи.
-C - режим совместимости с Vi.

Круто, правда? Но это только начало. Опции ничего по сравнению с командами редактора.

Командный режим Vim

Чтобы перейти в режим редактирования нужно нажать A
Чтобы выйти из режима редактирования нужно нажать Esc
Чтобы перейти в режим командной строки нужно нажать Shift-:

В командном режиме вы можете перемещаться по редактируемому тексту и выполнять действия над ним с помощью буквенных клавиш. Именно этот режим открывается по умолчанию при старте редактора. Здесь вы будете использовать краткие команды, перед которыми может устанавливаться номер, чтобы повторить команду несколько раз. Для начинающих может быть поначалу очень запутанно то, что в командном режиме символы интерпретируются как команды.

Для перемещения используются такие команды:

h - на один символ влево;
l - на один символ вправо;
j - на одну строку вниз;

k - на одну строку вверх;
w - на слово вправо;
b - на слово влево;
H - перейти в низ экрана;
G - перейти в конец файла;

Можете запустить редактор и поэкспериментировать, чтобы было легче понять как это работает. Если перед тем как нажать кнопку буквы нажать цифру, то эта команда будет повторена несколько раз. Например, 3j переведет курсор на три строки вверх.

Для переключения в режим редактирования используются такие команды:

i - вставить текст с позиции курсора, символ под курсором будет заменен;
I - вставить текст в начало строки;
a - добавить текст начиная от позиции курсора;
o - вставить новую строку после этой и начать редактирование;
O - вставить новую строку перед этой и начать редактирование;
r - заменить текущий символ;
R - заменить несколько символов.

К этим командам тоже применимы символы повторения. Поэкспериментируйте, можно получить интересный и не совсем ожиданий результат.

Более сложны команды редактирования текста. Вы можете править текст не только в обычном режиме, но и в командном с помощью команд. Для этого применяются такие команды:

d - удалить символ;
dd - удалить всю строку;
D - удалить символы начиная от курсора и до конца строки;
y - копировать символ;
yy или Y - скопировать всю строку;
v - выделить текст;

Эти команды редактора vim работают немного по-другому после нажатия одной из них ничего не произойдет. Мы еще можем задать количество символов, к которым будет применена команда и направление, с помощью кнопок перемещения курсора. Например, чтобы удалить два символа справа от курсора нажмите d3l, а чтобы удалить три строки вниз - d3j. Команды yy, dd, Y - не что иное, как сокращения.

Кроме этих команд, есть еще несколько полезных, которые мы не можем не рассмотреть:

p - вставить после позиции курсора;
P - вставить перед позицией курсора;
u - отменить последнее действие;
. - повторить еще раз последнее действие;
U - отменить последнее действие в текущей строке;
/шаблон - искать вхождение;

%s/шаблон/заменить - заменить первое слово на второе;
n - продолжить поиск вперед;
N - продолжить поиск назад;

С основными командами разобрались. Но у нас есть еще командная строка Vim, которая сама по себе тоже представляет огромный интерес.
Командная строка Vim

Командная строка Vim запускается в командном режиме нажатием двоеточия - ":". Здесь доступны команды для сохранения файла и выхода из редактора, настройки внешнего вида и взаимодействия с внешней оболочкой. Рассмотрим наиболее часто используемые команды редактора vim:

:w - сохранить файл;
:q - закрыть редактор;
:q! - закрыть редактор без сохранения;
:e файл - прочитать содержимое файла в позицию курсора;
:r файл - вставить в содержимое файла в следующую строку;
:r! - выполнить команду оболочки и вставить ответ в редактор;
:set переменная=значение - установить значение переменной, например, tabstop=4, или set number, с помощью этой команды можно управлять многими аспектами работы vim.
:buffers - посмотреть открытые файлы.

Со всеми основами разобрались, и вы теперь использование vim не будет казаться вам таким сложным. Но это еще далеко не все, этот мощный редактор может еще очень многое. Дальше мы рассмотрим несколько примеров использования vim, чтобы вам было легче справиться с новой программой.
Редактирование файла в Vim

Несмотря на то, что из всего вышесказанного можно понять как это делается рассмотрим еще раз. Чтобы открыть файл выполните:
$ vim имя_файла

Затем, если вы не хотите пока использовать возможности командного режима просто нажмите i, чтобы перейти в режим редактирования. Здесь вы можете редактировать файл так же, как и в nano. После того как завершите нажмите Esc, чтобы перейти в командный режим и наберите :wq. Записать и выйти. Все, готово.
Поиск и замена в Vim

Довольно часто нам нужно найти определенную последовательность в тексте. Текстовый редактор Vim умеет это делать.

Во-первых, если нужно найти символ в строке, нажмите f и наберите нужный символ, курсор будет перемещен к его позиции.

Для поиска по всему файлу используйте команду /. После нее нужно ввести слово, которое нужно найти. Для поиска следующего вхождения используйте n, для предыдущего - N.

Для замены будет использоваться немного другая конструкция:
:%s/искать/заменить/g

Двоеточие запускает командную оболочку с командой s для замены. Символ % означает что обрабатывать нужно весь файл, а g значит, что нужно обработать все найденные строки, а не только первую. Чтобы программа спрашивала перед каждой заменой можно добавить в конец строки опцию.
Одновременное редактирование нескольких файлов

Чтобы открыть несколько файлов, просто передайте их в параметры при запуске программы:
$ vim файл1 файл2 файл3

Редактор vim linux откроет первый файл, для переключения ко второму используйте команду :n, чтобы вернутся назад :N.

С помощью команды :buffers вы можете посмотреть все открытые файлы, а командой :buffer 3 переключится на третий файл.
Буфер обмена Vim

Текстовый редактор Vim имеет свой буфер обмена. Например, вам нужно скопировать в четыре строки и вставить их в другое место программы, для этого выполните такую последовательность действий:

- Нажмите Esc, чтобы перейти в командный режим;
- Наберите 4yy чтобы скопировать четыре строки;
- Переместите курсор в место где нужно вставить эти строки;
- Нажмите p для вставки.

Также можно использовать выделение vim, чтобы скопировать строки. Выделите текст с помощью v, а затем нажмите y, чтобы скопировать.
Кириллица в Vim

Кириллица в Vim работает превосходно. Но есть одно но, когда включена кириллица в системе, все команды vim не работают, им и не нужно работать, они же не приспособлены для кириллицы.

Но переключать каждый раз раскладку, когда работаете в командном режиме тоже не очень удобно, поэтому открываем файл ~/.vimrc и добавляем туда такие строки:
set keymap=russian-jcukenwin
set iminsert=0
set imsearch=0

Теперь раскладка клавиатуры в командном режиме переключается по Ctrl+^ и все команды работают.

Локаль в консоле

Узнать локаль из консоли:

```
rus@rus-400b4b:~$ echo $LANG
ru_RU.UTF-8
```

Установить локаль для конкретной программы:

```
rus@rus-400b4b:~$ LANG=en_EN.UTF-8 df
Filesystem     1K-blocks      Used Available Use% Mounted on
udev             4020456         0   4020456   0% /dev
tmpfs             808136      9664    798472   2% /run
```

Получить список локалей:

locale -a

Список доступных локалей можно посмотреть так:

less /usr/share/i18n/SUPPORTED

Список русских локалей можно получить так:

cat /usr/share/i18n/SUPPORTED | grep ^ru

Номер файла

Иногда из-за проблем с кодировкой файлы или каталоги отображаются неправильно. Например в виде восклицательных знаков. Чтобы выполнять операции с файлами в подобных случаях применяют ls -li отобразить номер файла или каталога:

узнать номер файла ls -li

например:

```
rus@rus-400b4b:~/shells$ ls -li -- *
2978845 -rw-rw-r-- 1 rus rus      0 дек 12 20:34 foo.txt
2979128 -rwxrwxr-x 1 rus rus    112 дек 12 20:53 tcicle.sh
2978860 -rwxrwxr-x 1 rus rus    508 дек 12 20:44 testx.sh
2995012 -rw-r--r-- 1 rus rus 591992 дек 15 05:46 vboxdrv.ko
2995013 -rw-r--r-- 1 rus rus  16392 дек 15 05:46 vboxnetadp.ko
2995014 -rw-r--r-- 1 rus rus  41448 дек 15 05:46 vboxnetflt.ko
2995015 -rw-r--r-- 1 rus rus  39736 дек 15 05:46 vboxpci.ko
```

или

```
rus@rus-400b4b:~/shells/mnogo3$ ls -li
итого 20
17981 -rwxrwxr-x 1 test test 891 дек 24 05:59 arifm.sh
17982 -rwxrwxr-x 1 test test  92 дек 24 05:59 memo.sh
17983 -rwxrwxr-x 1 test test 167 дек 24 05:59 randpass.sh
17984 -rwxrwxr-x 1 test test 127 дек 24 05:59 timeout.sh
17985 -rwxrwxr-x 1 test test 353 дек 24 05:59 usname.sh
```

выполнить действие с файлом - удаление файла с № 2978845

```
rus@rus-400b4b:~/shells$ find -inum 2978845 -exec rm {} \;
rus@rus-400b4b:~/shells$ ls -li *
2979128 -rwxrwxr-x 1 rus rus    112 дек 12 20:53 tcicle.sh
2978860 -rwxrwxr-x 1 rus rus    508 дек 12 20:44 testx.sh
2995012 -rw-r--r-- 1 rus rus 591992 дек 15 05:46 vboxdrv.ko
2995013 -rw-r--r-- 1 rus rus  16392 дек 15 05:46 vboxnetadp.ko
2995014 -rw-r--r-- 1 rus rus  41448 дек 15 05:46 vboxnetflt.ko
2995015 -rw-r--r-- 1 rus rus  39736 дек 15 05:46 vboxpci.ko
```

выполнить действие с файлом - удаление каталога с №

```
test@test-VirtualBox:~/shells$ find -inum 17980 -exec rm -r {} \;
find: «./mnogo3»: Нет такого файла или каталога
```

Администрирование ОС Linux

Оглавление:

Команды администрирования

Управление ОС

shutdown -r now перезагрузить ОС
shutdown -h now выключить компьютер
shutdown -h 00:00 выключение по времени
reboot перезагрузить ОС
halt завершить работу ОС
halt -f быстрое выключение компьютера
logout выход из текущего сеанса
kill -9 0000 Завершить процесс (№ процесса узнать по ps-d)
date 1207130816 Установка даты и времени (12/07/13/08/16 – месяц/день/час/мин/год)
hwclock -w Сохранить системные время и дату
http://localhost:631/
адрес в браузере для настройки сервера печати (принтера)
useradd -g wheel -G adm,cdrom,lp -s /bin/bash -d /home/rus -m rus
создание нового пользователя rus с каталогом /home/rus
passwd rus задать пароль пользователю rus

Монтирование устройств

fdisk -l показать все подключенные устройства хранения данных
mount показать все смонтированные системы;
mount /dev/sdc /mnt/usb/ монтирование USB (флэшки)
umount /dev/sdc /mnt/usb/ Размонтирование USB (флэшки)
mount /dev/sr0 /mnt/cdrom/ Монтирование CDROM
umount /dev/sr0 /mnt/cdrom/ Размонтирование CDROM
mount /dev/sda3 /mnt/huge/ Монтирование SSD-диск
umount /dev/sda3 /mnt/huge/ Размонтирование SSD-диск

Системная информация

man [имя команды] показать справку
arch показать архитектуру процессора
uname -a показать название и версию ОС, версию ядра, текущую дату

sudo lshw показать общую информацию об оборудовании
cat /etc/*version версия ОС
cat /proc/cpuinfo показать информацию о процессоре
cat /proc/interrupts показать прерывания
free -k показать объем оперативной памяти в килобайтах
free -m показать объем оперативной памяти в мегабайтах
date показать системную системные дату и время
lspci показать все подключенные устройства PCI
lspci -v показать подробную информацию о всех устройствах PCI
lsusb показать информацию о всех подключенных устройствах USB
dpkg -l | less показать все установленные программы
who показать сведения о пользователях, работающих в ОС
ps -a показать все запущенные в текущий момент программы
ps -d показать все запущенные программы и процессы
lspci -k | grep -E "VGA|3D" показать информацию о видеоконтроллере
lspci -k | grep VGA -A2 показать информацию о видеодрайвере
glxinfo | grep OpenGL показать информацию о поддержке OpenGL

Сеть и Интернет

ifconfig -a Показать информацию о всех сетевых интерфейах
ifconfig eth0 Показать информацию о сетевом интерфейсе eth0
netstat -rn Показать локальную таблицу маршрутизации
route -n Показать таблицу маршрутизации
hostname Показать имя компьютера
hostname newname Указать новое имя (newname) компьютера
ip link show Показать тип и статус соединения eth0
cat /etc/sysconfig/network-devices/ifconfig.eth0/ipv4
Показать настройки IP .
Пример файла ipv4:
IP=192.168.1.211
BROADCAST=192.168.1.255
GATEWAY=192.168.1.1
CHECK_LINK=yes
ONBOOT=yes
TYPE=Ethernet
USERCTL=no
PEERDNS=no
SERVICE=ipv4-static
PREFIX=24
vi /etc/sysconfig/network-devices/ifconfig.eth0/ipv4
Настроить IP
dhcient
Запуск dhc для просмотра веб-страниц
/etc/init.d/network restart
Перезапуск сетевого интерфейса
/etc/init.d/network stop
Останов сетевого интерфейса
/etc/init.d/network start
Запуск сетевого интерфейса

Пользовательские команды программы

Файловая система

mc Файловый менеджер
cat Просмотр файла
cd Переход в каталог
ls Просмотр содержимого каталога
vi Тектовый редактор командной строки
mkdir Создание нового каталога
cp -i Копирование файла
mv Перемещение или переименование файла или каталога
rm -i Удаление файла
rm -ir Удаление директории
df -h Показать объём и свободное пространство жесткого диска
du -hsx Общий размер текущего или другого (указать какого) каталога в Mb
file Определить тип файла
chmod +x Сделать указанный файл исполняемым
updatedb Обновить базу поиска файлов и каталогов
locate Поиск файлов и каталогов
find Поиск файлов и каталогов
which Поиск исполняемых файлов (программ)
whereis Показать местоположение исполняемых файлов

Прикладные программы

cal 2016
Показать календарь на 2016 год
echo $((25+25))
Калькулятор командной строки

TAR

Как создать архив .tar.gz

tar -cvf file.tar /full/path - создать .tar

tar -czvf file.tar.gz /full/path - создать .tar.gz (архив)

tar -cjvf file.tar.bz2 /full/path - создать .tar.bz2 (архив)

Синтаксис этих примеров:
tar [-ключи] [название архива] [путь, что запаковать]

Как открыть (распаковать) .tar

Чтобы распаковать .tar: tar -xvf file.tar.gz

Синтаксис: tar [-ключи] [название архива]

Ключи архиватора

c - (Create) создать архив.
z – создает архив .tar.gz
j - создает архив .tar.bz2
x - (eXtract) позволяет вам извлекать файлы из архива.
v - делает вывод tar подробным. Это означает, что на экран будут выведены все найденные в архиве файлы. Если эта опция опущена, информация, выводимая в процессе обработки, будет ограничена.
f - является обязательной опцией. Без неё tar пытается использовать магнитную ленту вместо файла архива.
z - позволяет вам обрабатывать архив, сжатый gzip'ом (с расширением .gz). Если вы забудете указать эту опцию, tar выдаст ошибку. И наоборот, эта опция не должна использоваться для несжатых архивов.
t - (Test) просмотреть содержимое архива.

Более подробно о ключах и возможностях вы можете узнать, набрав команду в среде Unix
[~]# man tar

В файлах .tar можно хранить несколько папок (структуру папок и файлов). Лучше использовать с ключами для архивации файла .tar, чтобы в результате получились файлы с расширением .tar.gz (файл .tar сжатый архиватором gzip) или .tar.bz2 (файл .tar сжатый bzip2).
bzip2 лучше сжимает, но с gzip более распространён, поэтому лучше сжимать этим архиватором.
Если у вас установлен WinRAR, то проблем с открытием .tar.bz2 и .tar.gz не будет.

История архиватора

tar (англ. tape archive) — формат битового потока или файла архива, а также название традиционной для Unix программы для работы с такими архивами.

Программа tar была стандартизирована в POSIX.1-1998, а также позднее в POSIX.1-2001. Первоначально программа tar использовалась для создания архивов на магнитной ленте, а в настоящее время tar используется для хранения нескольких файлов внутри одного файла, для распространения программного обеспечения, а также по прямому назначению — для создания архива файловой системы. Одним из преимуществ формата tar при создании архивов является то, что в архив записывается информация о структуре каталогов, о владельце и группе отдельных файлов, а также временные метки файлов.
Как и другие утилиты Unix, tar — специализированная программа, которая следует философии Unix: «делать только одну вещь» (работать с архивами), «но делать её хорошо». Поэтому tar не создаёт сжатых архивов, а использует для сжатия внешние утилиты, такие как gzip и bzip2. Ранее для сжатия использовалась также утилита compress, которая практически вышла из употребления.

Примечание

Из-за достаточно поздней стандартизации существует несколько похожих, но не до конца совместимых форматов. В частности различие между GNU tar и SUN Solaris tar наблюдается при длине имени файла, включаемого в архив, более 100 символов или размере включаемого в архив файла более 8 ГБ.

Расширения имён файлов

Для файлов, содержащих архивы tar, традиционно применяются следующие расширения имён файлов:

архив tar:
.tar

архив tar, сжатый программой gzip:
.tar.gz
.tgz (в случае ограничений файловой системы на длину расширения)
.tar.gzip

архив tar, сжатый программой bzip2
.tar.bz2
.tar.bzip2
.tbz2
.tb2
.tbz

архив tar, сжатый программой compress
.tar.Z
.taz

архив tar, сжатый программой LZMA
.tar.lzma

архив tar, сжатый программой XZ
.tar.xz

архив tar, сжатый программой lzop
.tar.lzo
.tzo

Размер файла или каталога

Узнать размер файлов или каталогов

du -h узнать размер файлов и каталогов в текущем каталоге в Mb

du -chs с общим итогом

du -h --max-depth=1

du -hsx общий размер указанного каталога или файла

du -h | sort - список каталогов с их размерами от наибольшего к меньшему

df -h общий размер основных каталогов ОС

Пример:

sudo du -hd1a /home/sam/

результат:

```
rus@rus-400b4b:~$ sudo du -hd1a /home/sam/
sudo: unable to resolve host rus-400b4b
12K     /home/sam/Рабочий стол
0       /home/sam/Загрузки
0       /home/sam/Шаблоны
0       /home/sam/Общедоступные
0       /home/sam/Документы
0       /home/sam/Музыка
104M    /home/sam/Изображения
0       /home/sam/Видео
1,9G    /home/sam/.cache
196M    /home/sam/.local
34M     /home/sam/.mozilla
1,4G    /home/sam/.wine
```

Хитрости

find . -type f -print | wc -l общее количество файлов в текущем каталоге

Подчёт строк

Подсчитать количество строк вывода. Команда wc

```
rus@rus-400b4b:~$ cat /proc/cpuinfo | grep MHz
cpu MHz         : 800.000
cpu MHz         : 828.906
cpu MHz         : 852.343
cpu MHz         : 807.910
rus@rus-400b4b:~$ cat /proc/cpuinfo | grep MHz | wc -l
4
```

Время выполнения скрипта

Время выполнения скрипта time

пример

```
rus@rus-400b4b:~$ \time -f %E sleep 5 1>/dev/null
0:05.00
```

пояснения:

\time - запуск системной команды, не встроенной bash!
-f %E — формат вывода для time. E = реальное время, % — обязательный метасимвол

Узнать погоду

Узнать погоду можно так:
wget -O - wttr.in -q
или так:
wget -O - wttr.in/Krasnoyarsk -q
Если в название города имеются пробел или другие символы, то необходимо добавить кавычки:
wget -O - wttr.in/"Saint Petersburg" -q
wget -O - wttr.in/"Leninsk Kuznetsk" -q
wget -O - wttr.in/"Petropavlovsk Kamchatsky" -q
Так же вместо названия города, можно использовать код аэропорта. Например Толмачёво - международный аэропорт Новосибирска:
wget -O - wttr.in/OVB -q
Можно в браузере открыть:
wttr.in/krasnoyarsk
будет так:
http://storage5.static.itmages.ru/i/17/0513/h_1494701299_9787257_6d81a0fd65.png

Распаковать ZIP

```
sudo unzip filename
```

CHMOD

выполняем команду для установки прав на файлы (вместо 644 указывает нужные права)

```
find . -type f -exec chmod 644 {} \;
```

выполняем команду для установки прав на каталоги (вместо 755 указываем нужные права)

```
find . -type d -exec chmod 755 {} \;
```

Секреты Bash

Трюк №1

```
$ sudo !!
```

Запускает предыдущую команду с root правами.

Полезно, когда забываете написать sudo. «!!» сам захватывает предыдущую строчку.

Трюк №2

```
$ python -m SimpleHTTPServer
```

Запускает простой сервер прямо в консоли Linux для просмотра текущей директории в виде вебстраницы. Просмотреть можно например так:

```
$ firefox http://$HOSTNAME:8000/
```

Трюк №3

```
$ ^foo^bar
```

Запускает предыдущую команду, но с заменой.

Удобно применять, если сделали опечатку. По умолчанию аргументы пустые, поэтому, запустив что-нибудь такое:

```
$ echo "no typozs"
```

можете легко исправить с помощью

```
$ ^z
```

Трюк №4

```
$fc
```

или

Ctrl+X+E

Открывает текстовый редактор, чтобы написать длинную команду.

Писать можно на ваше усмотрение в vi, emacs, nano... Комбинация клавиш захватывает написанный текст и открывает его в редакторе, задаваемом в $EDITOR.

Трюк №5

Alt+. или Esc+.

Вставляет последний аргумент предыдущей команды. Например, если вы написали:

```
$ cp file.txt /var/www/proglib/
$ cd
```

и нажали Alt+. , то строка станет

```
$ cd /var/www/proglib/
```

Повторение этой комбинации клавиш подставляет аргументы из более давних команд.

Трюк №6

```
$ mount | column -t
```

Выводит все подключенные файловые системы в человеческом виде.

Трюк №7

```
$ echo "ls -l" | at midnight
```

Выполняет заданную команду в назначенное время.

Подробнее про at и похожие программы: https://www.computerhope.com/unix/uat.htm

Трюк №8

```
$curl ifconfig.me
```

Выводит глобальный IP адрес. Также:

```
curl ifconfig.me/host -&gt; Remote Host
curl ifconfig.me/ua -&gt;User Agent
curl ifconfig.me/port -&gt; Port
```

Работает это благодаря сайту ifconfig.me

Трюк №9

```
$ man ascii
```

Позволяет быстро посмотреть таблицу символов Ascii.

Трюк №10

```
$ mount -t tmpfs tmpfs /mnt -o size=1024m
```

Монтирует часть оперативной памяти как временный раздел в /mnt. Наверняка нет смысла объяснять преимущества использования RAM. Данные будут сохранены до следующей перезагрузки или до

```
$ umount /mnt
```

Трюк №11

Ctrl-L

Просто очищает терминал.

Трюк №12

```
$ disown -a &amp;&amp; exit
```

Закрывает терминал, оставляя работать все запущенные подпроцессы.

Трюк №13

```
$ mv filename.{old,new}
```

Быстрый способ переименовать файл

Трюк №14

```
$ pushd /tmp
```

Добавляет директорию в стек, чтобы после вернуть к текущей. Например:

```
$ cd /complicated/path/.I/dont/want/to/forget
$ pushd /tmp
$ cd thing/in/tmp
$ popd
```

Вы снова на коне в /complicated/path/.I/dont/want/to/forget.

Трюк №15

```
$ rm !(*.foo|*.bar|*.baz)
```

Удаляет все файлы, не имеющие заданное расширение.

Трюк №16

Ну и напоследок:

```
$ telnet towel.blinkenlights.nl
```

Демонстрирует Звёздные Войны (4 эпизод) прямо в консоли.

В самом начале нужно немного подождать, а для остановки использовать Ctrl+].