Домашняя работа 3 по ОС

Мигалин

1. pwd

Печатает на экран текущий каталог. Это может быть полезно, если ваша командная строка Linux не выводит такую информацию, а также особенно полезно в Bash программировании, для получения ссылки на каталог в котором выполняется скрипт.

2. ls

Утилита для просмотра содержимого каталогов. По умолчанию показывает текущий каталог. Если в параметрах передать путь, то она перечислит содержимое того каталога. Полезные опции -l (**L**ist) и -a (**A**ll). Первая форматирует вывод в виде списка с более подробной информацией, а вторая включает показ скрытых файлов.

`ls`	отобразить содержимое текущей директории
`ls -F`	отобразить содержимое текущей директории с добавлением к именам символов, храктеризующих тип
`ls -l`	показать детализированое представление файлов и директорий в текущей директории
`ls -a`	показать скрытые файлы и директории в текущей директории
`ls *[0-9]*`	показать файлы и директории содержащие в имени цифры

3. cat

Печатает содержимое файла, переданного в параметре, в стандартный вывод. Если передать несколько файлов, команда склеит их. Также можно перенаправить вывод в еще один файл, с помощью символа ‘>’. Если нужно вывести только определенное количество строк используйте опцию -n (**N**umber).

`cat file_originale | [operation: sed, grep, awk, grep и т.п.] > result.txt`	общий синтаксис выполнения действий по обработке содержимого файла и вывода

	результата в новый
cat file_originale | [operazione: sed, grep, awk, grepи т.п.] >> result.txt	общий синтаксис выполнения действий по обработке содержимого файла и вывода результата в существующий файл. Если файл не существует, он будет создан
cat -n file1	пронумеровать строки при выводе содержимого файла
cat example.txt | awk 'NR%2==1'	при выводе содержимого файла, не выводить чётные строки файла

4. cp

Копирование файлов и каталогов. Она не копирует каталоги рекурсивно по умолчанию, поэтому не забудьте добавить опцию -r (**R**ecursive) или -a (**A**rchive). Последняя включает режим сохранения атрибутов, владельца и временного штампа, в дополнение к рекурсивному копированию.

cp file1 file2	сопировать файл file1 в файл file2
cp dir/* .	копировать все файлы директории dir в текущую директорию
cp -a /tmp/dir1 .	копировать директорию dir1 со всем содержимым в текущую директорию
cp -a dir1 dir2	копировать директорию dir1 в директорию dir2

5. mv

Перемещение или переименование файлов и каталогов. Примечательно, что переименование и перемещение — это одна и та же операция. Переименование — это перемещение файла в ту же папку с другим именем.

6. pr

Команда pr используется для форматирования файлов перед печатью. По умолчанию заголовок включает в себя имя файла, дату и время создания файла, номер страницы, а также две пустых строки нижнего колонтитула. Когда данные поступают из нескольких файлов или со стандартного устройства ввода, вместо даты и времени создания файла используются текущие дата и время. Можно печатать файлы рядом, каждый в своем столбце, а также управлять многими возможностями форматирования с помощью различных опций. Как обычно, дополнительную информацию вы можете найти на man-странице.

```
pr text1 | head

2009-08-11           text1                Page 1
1 apple
2 pear
3 banana
```

7. lpr

Команда lpr (line printer) печатает файл из терминала Linux. Утилита lpr помещает один или несколько файлов в очередь печати. Концепция очереди позволяет нескольким пользователям системы направлять вывод на один принтер. Очередь печатается последовательно, то есть первый файл в очереди будет напечатан первым. Если вы редактируете большинство своих файлов в консольных редакторах, таких как vim или nano, и хотите быстро распечатать набранный текст, вы найдете команду lpr очень легкой в употреблении и удобной. Для того, чтобы печатать с помощью команды lpr, необходимо установить пакет System printer configuration.

Если имя_файла не задано, *lpr* ожидает ввод данных со стандартного ввода

```
lpr -P printer_name filename
```

8. lpq

Для просмотра содержимого очереди печати используется команда *lpq*. Команда запущенная без аргументов, она возвращает содержимое очереди печати принтера по умолчанию.

Вывод возвращаемый *lpq* может быть полезен для многих целей.

```
$ lpq

lp is ready and printing

Rank   Owner      Job  Files             Total Size

active mwf        31   thesis.txt      682048 bytes
```

9. who

Команда who сообщает имя пользователя, имя терминальной линии, астрономическое время начала сеанса, продолжительность бездействия терминальной линии с момента последнего обмена, идентификатор процесса интерпретатора команд shell для каждого из пользователей, работающих в системе UNIX. Для получения этой информации команда просматривает файл /etc/utmp. Впрочем, вместо него может просматриваться другой файл, имя которого должно быть тогда указано в командной строке (файл должен иметь формат utmp[4]). Обычно в качестве файла указывают /etc/wtmp, где зафиксированы времена начала всех сеансов с момента его последнего создания.

Команда who с опциями am i или am I идентифицирует обратившегося с ней пользователя.

Выдаваемые сообщения имеют, вообще говоря, следующий формат:

NAME [STATE] LINE TIME [IDLE] [PID] [COMMENT] [EXIT]

10. ps

Как уже говорилось, чтобы уничтожить процесс нужен его идентификатор. Один из способов получить его, это утилита ps, которая печатает информацию о запущенных процессах. По умолчанию вывод очень длинный, поэтому используйте опцию -e, чтобы увидеть информацию об определенном процессе. Это только снимок состояния на момент вызова, и информация не будет

обновляться. Команда ps с ключом aux выводит полную информацию о процессах. Pgrep работает следующим образом — вы задаете имя процесса, а утилита показывает его идентификатор.

`ps -eafw`	отобразить запущенные процессы, используемые ими ресурсы и другую полезную информацию (единожды)
`ps -e -o pid,args --forest`	вывести PID'ы и процессы в виде дерева

11. tty

Команда **tty** выводит имя терминала подключенного к стандартному вводу. В случае если стандартный ввод не является терминалом выводит сообщение "not a tty".

12. du

`du -sh dir1`	подсчитывает и выводит размер, занимаемый директорией 'dir1' *(Прим.переводчика. ключ -h работает не во всех *nix системах)*
`du -sk * | sort -rn`	отображает размер и имена файлов и директорий, с соритровкой по размеру

13. cal

Команды cal, ncal в Linux входят в пакет bsdmainutils и позволяют отображать календарь и (как написано в мане) дату Пасхи.По сути cal является ссылкой на программу ncal, но по умолчанию cal выводит календарь с днями по горизонтали и начиная с воскресения, а ncal выводит дни по вертикали и начинает отсчет с понедельника.

14. cd

Позволяет перейти из текущего каталога, в указанный. Если запустить без параметров — возвращает в домашний каталог. Вызов с двумя точками

возвращает на уровень вверх относительно текущего каталога. Вызов с тире (cd -) возвращает к предыдущему каталогу

15. mkdir

Создание новых каталогов. Наиболее удобная опция — p (**P**arents), позволяет создать всю структуру подкаталогов одной командой, даже если они еще не существуют.

```
mkdir dir1                  создать директорию с именем
                            'dir1'
mkdir dir1 dir2             создать две директории
                            одновременно
mkdir -p /tmp/dir1/dir2     создать дерево директорий
```

16. rmdir

Команда rmdir удаляет указанные каталоги, которые должны быть пустыми. Для удаления каталога вместе с содержимым следует воспользоваться командой rm с опцией -r. Текущий каталог [см. pwd] не должен принадлежать поддереву иерархии файлов с корнем - удаляемым каталогом.

Если каталог не пустой или это файл, то будет выведена ошибка

17. rm

Удаляет файлы и папки. Очень полезная команда Linux, с помощью нее вы можете убрать весь беспорядок. Однако будьте осторожны при ее использовании. Хоть и для того чтобы повредить систему вам нужно серьезно постараться, вы можете удалить собственные важные файлы. Rm не удаляет файлы в корзину, из которой потом все можно будет восстановить. Так что будьте осторожны, чтобы потом не говорили: «rm съела мою курсовую». Все действия необратимы. Если нужно рекурсивное удаление, используйте опцию -r.

```
rm -f file1         удалить файл с именем 'file1'
rmdir dir1          удалить директорию с именем 'dir1'
rm -rf dir1         удалить директорию с именем 'dir1' и
                    рекурсивно всё её содержимое
rm -rf dir1 dir2    удалить две директории и рекурсивно их
                    содержимое
```

18. chmod

chmod изменяет права доступа каждого указанного *файла* в соответствии с правами доступа, указанными в параметре *режим*, который может быть

представлен как в символьном виде, так и в виде восьмеричного числа, представляющего битовую маску новых прав доступа.

Формат символьного режима таков: `[ugoa...][[+-=][rwxXstugo...]...][,...]'.

Оператор `+' добавляет выбранные права доступа к уже имеющимся у каждого файла; `-' удаляет эти права; а `=' присваивает только эти права каждому указанному файлу.

Буквы `rwxXstugo' выбирают новые права доступа для пользователя, заданного одной из букв `ugoa': чтение (r); запись (w); выполнение (или доступ к каталогу) (x); выполнение, если файл является каталогом или уже имеет право на выполнение для какого-нибудь пользователя (X); setuid- или setgid-биты (s); sticky-бит (t); установка для остальных таких же прав доступа, которые имеет пользователь, владеющий этим файлом (u); установка для остальных таких же прав доступа, которые имеет группа файла (g); установка для остальных таких же прав доступа, которые имеют остальные пользователи (не входящие в группу файла) (o). (Так, `chmod g-s file' снимает бит set-group-ID (sgid), `chmod ug+s file' устанавливает биты suid и sgid, в то время как `chmod o+s file' ничего не делает).

`chmod ugo+rwx directory1`	добавить полномочия на директорию directory1 ugo(User Group Other)+rwx(Read Write eXecute) - всем полные права. Аналогичное можно сделать таким образом *chmod 777 directory1*
`chmod go-rwx directory1`	отобрать у группы и всех остальных все полномочия на директорию directory1.
`chmod u+s /bin/binary_file`	назначить SUID-бит файлу /bin/binary_file. Это даёт возможность любому пользователю запускать на выполнение файл с полномочиями владельца файла.
`chmod u-s /bin/binary_file`	снять SUID-бит с файла /bin/binary_file.
`chmod g+s /home/public`	назначить SGID-бит директории /home/public.
`chmod g-s /home/public`	снять SGID-бит с директории /home/public.
`chmod o+t /home/public`	назначить STIKY-бит директории /home/public. Позволяет удалять файлы только владельцам

```
chmod o-t /home/public    снять STIKY-бит с директории
                          /home/public
```

19. chown

Изменяет владельца файла. Только суперпользователь может изменять владельцев. Для рекурсивного изменения используйте опцию -R.

```
chown user1 file1           назначить владельцем файла
                            file1 пользователя user1
chown -R user1 directory1   назначить рекурсивно
                            владельцем директории
                            directory1 пользователя user1
chown user1:group1 file1    сменить владельца и группу
                            владельца файла file1
```

20. chgrp

chgrp изменяет группу каждого заданного *файла* на *группу*, которая может быть представлена как именем группы, так и ее числовым идентификатором (GID). Может использоваться непривилегированными пользователями для изменения группы файлов. В отличие от команды chown, chgrp позволяет рядовым пользователям изменять группы, но только те, членами которых они являются.

```
chgrp group1 file1      сменить группу-владельца файла
                        file1 на group1
```

21. cmp

Производится сравнение двух бинарных файлов. При их различии возвращается код ошибки и выводится номер байта, где это произошло

22. find

Поиск в файловой системе, файлов и папок. Это очень гибкая и мощная команда Linux не только из-за своих возможностей поиска, но и благодаря возможности выполнять произвольные команды для найденных файлов.

```
find / -name file1          найти файлы и директории с именем file1. Поиск начать с корня (/)
find / -user user1          найти файл и директорию принадлежащие пользователю user1. Поиск начать с корня (/)
find /home/user1 -name "*.bin"  Найти все файлы и директории, имена которых оканчиваются на '. bin'. Поиск начать с '/ home/user1'
find /usr/bin -type f -atime +100  найти все файлы в '/usr/bin', время последнего обращения к которым более 100 дней
find /usr/bin -type f -mtime -10  найти все файлы в '/usr/bin', созданные или изменённые в течении последних 10 дней
find / -name *.rpm -exec chmod 755 '{}' \;  найти все фалы и директории, имена которых оканчиваются на '.rpm', и изменить права доступа к ним
find / -xdev -name "*.rpm"  найти все фалы и директории, имена которых оканчиваются на '.rpm', игнорируя съёмные носители, такие как cdrom, floppy и т.п.
```

23. file

Показывает тип файла. В Linux файлы не обязаны всегда иметь расширения для того, чтобы с ними работать. Поэтому пользователю иногда трудно определить, что за файл перед ним. Эта маленькая утилита решает проблему.

24. ar

Команда ar предоставляет средства обслуживания группы файлов, об единенных в один архивный файл. Применяется главным образом для создания и изменения библиотечных файлов, используемых редактором связей. Может применяться и для других подобных целей. Магические цепочки и заголовки файлов состоят из печатаемых ASCII-символов, так что если в состав архива входят только печатаемые файлы, то и архив в целом окажется печатаемым. Иными словами архивирование файлов без сжатия

25. ranlib

Программа **ranlib** добавит индекс к архиву и получится полноценная статическая библиотека объектных файлов. Стоит отметить, что на некоторых системах программа **ar** автоматически создает индекс, и использование **ranlib** не имеет никакого эффекта. Но тут надо быть

осторожным при атоматической компиляции библиотеки с помощью файлов **makefile**, если вы не будете использовать утилиту **ranlib**, то возможно на каких-то системах библиотеки будут создаваться не верно и потеряется независимость от платформы. Так что возьмем за правило тот факт, что утилиту **ranlib** надо запускать в любом случае, даже если он нее нет никакого эффекта

```
ranlib libимя_библиотеки.a
```

26. tee

Вы можете использовать команду **tee** для копирования ввода на стандартный вывод и один или несколько выходных файлов за один раз. Использование опции -a с **tee** в результате добавляет ввод в файл(ы). Эта команда полезна, если вы хотите как увидеть, так и сохранить вывод. Операторы > и >> не позволяют выполнить оба действия одновременно.
Этот инструмент обычно называется подачей на конвейер (|), что демонстрируется в следующем примере:

```
mireille ~/test> date | tee file1 file2
Thu Jun 10 11:10:34 CEST 2004
mireille ~/test> cat file1
Thu Jun 10 11:10:34 CEST 2004
mireille ~/test> cat file2
Thu Jun 10 11:10:34 CEST 2004
mireille ~/test> uptime | tee -a file2
 11:10:51 up 21 days, 21:21, 57 users,  load average:
0.04, 0.16, 0.26
mireille ~/test> cat file2
Thu Jun 10 11:10:34 CEST 2004
 11:10:51 up 21 days, 21:21, 57 users,  load average:
0.04, 0.16, 0.26
```

27. sort

Сортировка строк текста по различным критериям. Наиболее полезные: -n (**N**umeric) — по числовому значению, и -r (**R**everse), которая переворачивает вывод. Это может быть полезно для сортировки вывода du. Например, если хотите отсортировать файлы по размеру, просто соедините эти команды.

```
sort file1 file2       отсортировать содержимое двух
                       файлов
```

28. cut

cut извлекает поля/строки/байты из текстовых файлов. Символом-разделителем по умолчанию является символ табуляции. В листинге 16 содержится пример, в котором команда cut используется для разделения двух столбцов файла text2, а затем в качестве разделителя выходных данных используется пробел, что является необычным способом преобразования символов табуляции в пробелы.

Листинг 16. Использование команды cut

```
   cut -f1-2 --output-delimiter=' ' text2

   9 plum
   3 banana
   10 apple
```

29. head/tail

Еще одна пара, но здесь у каждой команды своя область применения. Head выводит несколько первых строк из файла (голова), а tail выдает несколько последних строк (хвост). По умолчанию каждая утилита выводит десять строк. Но это можно изменить с помощью опции -n. Еще один полезный параметр -f. Это сокращение от **F**ollow (следовать), утилита постоянно выводит изменения в файле на экран. Например, если вы хотите следить за лог файлом, вместо того чтобы постоянно открывать и закрывать его используйте tail -nf.

```
head -2 file1              вывести первые две строки файла
                           file1 на стандартное устройство
                           вывода. По-умолчанию выводится
                           десять строк
tail -2 file1              вывести последние две строки
                           файла file1 на стандартное
                           устройство вывода. По-умолчанию
                           выводится десять строк
tail -f /var/log/messages  выводить содержимое файла
                           /var/log/messages на
                           стандартное устройство вывода
                           по мере появления в нём текста.
```

30. tail

~ предыдущее

31. spell

Команда spell сравнивает каждое слово в файле со своим словарем и печатает список всех потенциальных орфографических ошибок на экране. Если в словаре spell нет какого-либо слова (например, персональное имя), то она также выдает его как орфографическую ошибку. Если вы подадите на ввод spell большой файл, то его обработка займет много времени и список ошибок может быть очень большим. Команда spell распечатывает весь список ошибок сразу. Поэтому лучше всего перенаправить вывод spell в файл.

32. grep

Grep как и другие инструменты Linux делает одно действие, но делает его хорошо. Она ищет текст по шаблону. По умолчанию она принимает стандартный ввод, но вы можете искать в файлах. Шаблон может быть строкой, или регулярным выражением. Она может вывести как совпадающие, так и несовпадающие строки и их контекст. Каждый раз, когда вы выполняете команду, которая выдает очень много информации, не нужно анализировать все вручную, пусть grep делает свою магию.

```
grep Aug /var/log/messages      из файла '/var/log/messages' отобрать и вывести на стандартное устройство вывода строки, содержащие "Aug"
grep ^Aug /var/log/messages     из файла '/var/log/messages' отобрать и вывести на стандартное устройство вывода строки, начинающиеся на "Aug"
grep [0-9] /var/log/messages    из файла '/var/log/messages' отобрать и вывести на стандартное устройство вывода строки, содержащие цифры
grep Aug -R /var/log/*          отобрать и вывести на стандартное устройство вывода строки, содержащие "Aug", во всех файлах, находящихся в директории /var/log и ниже
```

33. fgrep

Команда fgrep просматривает входные файлы в поиске строк, содержащих заданную цепочку_символов. Если файлы не указаны, используется стандартный ввод. Обычно каждая успешно сопоставленная строка копируется на стандартный вывод; если исходных файлов несколько, перед найденной строкой выдается имя файла. Отличие команды fgrep от egrep(1) и grep(1) в том, что она выполняет сопоставление с цепочкой символов, а не с шаблоном, заданным регулярным выражением; fgrep быстр и компактен.

Команда fgrep трактует символы $, *, [ ], ^, |, ( ), и \ буквально, а не как элементы полных регулярных выражений. Для экранирования этих символов от интерпретации shell'ом проще всего заключать цепочку_символов в одинарные кавычки.

34. more

Это две простенькие команды терминала, для просмотра длинных текстов, которые не вмещаются на одном экране. Представьте себе очень длинный вывод команды. Или вы вызвали cat для просмотра файла и вашему эмулятору терминала потребовалось несколько секунд, чтобы прокрутить весь текст. Если ваш терминал не поддерживает прокрутки, вы можете сделать это с помощью less. Less новее, чем more и поддерживает больше опций, поэтому использовать more нет причин.

```
more file1  постраничный вывод содержимого файла file1 на
            стандартное устройство вывода
```

35. wc

Команда wc выводит количество содержащихся в файле строк и слов, а также размер файла в байтах, определить который можно также с помощью команды ls –l

```
wc text*

 3  6 24 text1
 3  6 25 text2
 6 12 49 total
```