Работа с операционной системой GNU/Linux

Введение

В настоящее время основным интерфейсом взаимодействия пользователя и настольной операционной системой является графический интерфейс пользователя (Graphic User Interface - GUI), имея множество достоинств он, тем не менее, не способствует изучению основных принципов управления операционной системой (Operating System - OS) и файловой подсистемой, поскольку в GUI эти принципы сокрыты под множеством дополнительных слоев программного обеспечения (ПО). В свою очередь текстовый интерфейс пользователя (Text User Interface - TUI) располагается "ближе" к ОС и позволяет более легко освоить основные элементы, из которых состоит операционная система, при этом, TUI позволяет значительно ускорить выполнение обычных пользовательских и административных задач, даже таких простых как копирование файлов и каталогов. В данном пособии рассматривается работа с операционной системой GNU/Linux в текстовом режиме, стоит отметить, что, как и для остальных представителей семейства UNIX, для рассматриваемой системы работа в текстовом режиме исторически является основным способом взаимодействия с ОС, и часто позволяют выполнять задачи недоступные из графического интерфейса.

Регистрация в системе

Многие современные дистрибутивы GNU/Linux (особенно ориентированные на конечных пользователей) при загрузке автоматически запускают графическую подсистему. Несмотря на то, что в GUI также возможна работа с текстовым интерфейсом, в данном пособии предполагается работа в истинно текстовом режиме работы. Переход в текстовый режим возможен в любой момент работы и осуществляется по нажатию комбинации **`Ctrl+Alt+F(1-6)`**, при этом будет произведено переключение на другую виртуальную консоль.

Под термином консоль, в дальнейшем, будем понимать связку монитора и клавиатуры. Виртуальная консоль понимается в том смысле, что реальные монитор и клавиатура присутствуют только в единственном числе, но на разных консолях может отображаться различная информация и команды, вводимые с клавиатуры, отображаются только в текущей активной консоли. Данный инструмент является отголоском тех лет, когда к вычислительным мейнфреймам подключалось одновременно по несколько клавиатур и мониторов или алфавитно-цифровых терминалов, которые обеспечивали одновременную работу нескольких пользователей.

В современных дистрибутивах GNU/Linux номера текстовых консолей с 1 по 6, с номера 7 и далее идут графические консоли, которых также может быть несколько. Переключение между ними осуществляется с помощью комбинации **`Alt+F{1-12}`** из текстового режима и **`Ctrl+Alt+F{1-12}`** из графического.

Изначально текстовая консоль отображает следующие данные:

```
Debian GNU/Linux 5.0 rombik tty2
rombik login:
```

Данный текст означает следующее **`Debian GNU/Linux`** - название дистрибутива, **`5.0`** - номер версии дистрибутива, **`rombik`** - имя компьютера, **`tty2`** - название виртуальной консоли и ее номер (2)

`rombik login:` - приглашение к регистрации в системе.

Для дальнейшей работы вам необходим ввести ваше имя пользователя (выдаются администратором или преподавателем) и нажать клавишу **`Enter`**. После чего на экране появится требование ввести пароль. При вводе пароля он не отображается на экране из

соображений безопасности. После ввода пароля также нажмите клавишу **Enter** (в дальнейшем предполагается, что каждая команда и ввод данных с клавиатуры должно завершаться нажатием клавиши **Enter**). Если при вводе имени пользователя или пароля была допущена ошибка, то будет выдано соответствующее предупреждение и система запросит имя пользователя и пароль снова.

Если имя пользователь и пароль введены правильно, то на экране будет выведена различная информация, включая текущее время и дату, наличие новой почты и т.д. Последняя строка будет иметь следующий вид:

pmuser@rombik:~\$

здесь **pmuser** - имя пользователя, **rombik** - имя компьютера, **~** - текущий каталог, **\$** - приглашение для ввода команд. Данная строка также называется приглашением командной строки и означает, что система ждет от пользователя ввода команд с клавиатуры.

Отдельно следует сказать, что после регистрации в системе запускается специальная программа называемая оболочкой (**shell**) или командным интерпретатором. Данная программа является основным посредником между OS и пользователем при работе с текстовым интерфейсом. Основное назначение данной программы заключается в выполнении команд, введенных с клавиатуры и запуске остальных программ по требованию пользователя. После выполнения каждой команды оболочка вновь выводит на экран приглашение командной строки и ждет ввода новых команд (предполагается, что каждая команда завершается нажатием клавиши **Enter**).

В первую очередь мы рассмотрим команды выхода из системы (предполагается, что после окончания вашей работы за компьютером, вы закончите сеанс работы, чтобы другие пользователи могли зарегистрироваться и работать от своего имени, а не от вашего. Это также рекомендуется делать для соблюдения правил базовой безопасности). Стоит отметить, что после выхода из системы все запущенные вами программы в текущем сеансе будут закрыты, поэтому перед выходом следует сохранить все свои данные.

Для завершения текущего сеанса работы в приглашении оболочки необходимо выполнить команду **logout**, при этом содержимое экрана очистится и снова будет выведен запрос на ввод имени пользователя. К такому же результату приведет команда **exit** или нажатие на клавиатуре комбинации **Ctrl+D** (соответствует концу ввода данных).

Для удобства работы можно зарегистрироваться под одним и тем же пользователем на разных виртуальных консолях. При этом на одной можно открыть для просмотра данное руководство, а на другой выполнять различные команды, следуя данному материалу. При переключении на другую консоль текущий сеанс не завершается, но при этом по окончанию работы необходимо выйти из всех консолей с помощью приведенных выше команд.

Справочное руководство

В дальнейшем вам будут часто встречаться описания различных программ для работы с ОС, при этом рамки данного пособия не позволяют детально описать все возможности различных команд. Для более детального разъяснения назначения той или иной команды и их опций пользователь всегда может воспользоваться подробным справочным руководством присутствующим в любой UNIX системе. Для вызова справки можно использовать команду **man** (manual - руководство), например

$ man pwd - выведет справку по команде **pwd**

$ man ls — выведет справку по команде **ls**

Подобную справку можно получить по большей части команд, встречающихся в данном пособии. Кроме команды **man** можно использовать справку по команде **info**.

Навигация по файловой системе

Одной из важнейших функций OS является управление файлами. Для пользователя файловая система традиционно представляется в виде набора файлов и каталогов. Файлы хранят данные, а каталоги хранят информацию о файлах (например имена файлов). При этом, благодаря вложенности каталогов, файловую систему можно представить в виде дерева, в котором каждый узел содержащий поддеревья является каталогом, узлы не имеющие потомков это файлы (или пустые каталоги), а ветви указывают на вложенность файлов и каталогов в другие каталоги.

Во время работы с оболочкой пользователь постоянно находится в некотором каталоге, который называют текущим. Для того чтобы узнать текущий каталог можно воспользоваться командой **pwd** (текущей каталог также указан в строке приглашения после двоеточия). Пример вывода команды

```
$pwd
 /home/pmuser
```

Данная запись означает, что текущий каталог **pmuser**, содержится в каталоге **home**, который содержится в корневом каталоге (**/** - символ разделения каталогов).

Стоит отдельно сказать, что в отлчии от семейства DOS и Windows, где каждый раздел жесткого диска образует свой отдельный корень файловой системы (**A:,C:,D:**), в UNIX есть только один корень файловой системы, обозначаемый символом **/**. Все разделы файловой системы подключаются как поддеревья (процесс подключения раздела называется монтированием). По этой причине все абсолютные пути в UNIX начинаются с символа корневого каталога - **/**. Из приведенного выше примера видно, что каталог **home** находится в корневом каталоге, а каталог **pmuser** в каталоге **home**.

Для перемещения (навигации) по каталогам используется команда **cd** - change directory (сменить каталог). Например команда **cd /** изменит текущий каталог на корневой, в чем можно убедиться выполнив команду **pwd** или обратив внимание, что в предложении командной строки в качестве текущего каталога указан корневой

```
pmuser@rombik:/$
```

Чтобы просмотреть содержимое текущего каталога следует воспользоваться командой **ls** (list). При этом список файлов и каталогов будет сгруппирован в несколько столбцов

```
bin   cdrom  etc   initrd  lost+found  mnt  proc  sbin     srv
tmp  var
boot  dev    home  lib     media       opt  root  selinux  sys
usr
```

В корне содержатся основные системные каталоги OS.

Приведем краткую характеристику некоторых из системных каталогов, содержащихся в корне.

bin - основные утилиты,

etc - конфигурационные файлы.

mnt - для монтирования внешних файловых систем

boot - загрузочные файлы

dev - файлы устройств

home - домашние каталоги пользователя

lib - системные библиотеки и модули

root - домашний каталог суперпользователя

sbin - различные системные программы и утилиты

usr - программы, файлы, документации, заголовочные файлы

var - служит для хранения постоянно изменяющихся файлов, файлов журналов

tmp - хранение временных файлов

У каждого пользователя системы есть собственный отдельный каталог, называемый Домашний. Его имя традиционно совпадает с именем пользователя и он располагается в каталоге **/home** или его подкаталоге. Предполагается, что все свои личные файлы пользователь хранит в своем домашнем каталоге. Для того, чтобы из любого места файловой системы сразу перейти в домашний каталог пользователь может выполнить команду **cd** (change directory — сменить каталог) без параметров. При этом в приглашении командной строки текущий каталог будет обозначаться знаком **"~"** - это специальный символ для обозначения домашнего каталога.

В каждом каталоге присутствует два особых каталога **"."** - обозначает данный каталог и **".."** - обозначает родительский каталог. Например чтобы из домашнего каталога попасть в родительский каталог можно выполнить команду: **cd ..**

Краткий перечень основных команд раздела.

pwd - указывает текущий каталог пользователя

cd [путь к каталогу] — перейти к указанному каталогу

ls - просмотр содержимого каталога

Создание файлов и каталогов

Создавать новые файлы и каталоги пользователь может только в своей домашней директории. Для того чтобы в текущей директории создать новый каталог можно использовать команду

mkdir reports

при этом в текущем каталоге будет создан новый каталог с именем **reports**. Для того чтобы внутри каталога **reports** создать еще подкаталог можно выполнить команду

mkdir reports/lab1

При этом в каталоге **reports** будет создан подкаталог **lab1**. Также можно последовательно выполнить 2 команды:

cd reports

mkdir lab2

При этом первая команда сменит текущий каталог на **reports** и создаст в нем подкаталог **lab2**. В качестве параметра в команду **mkdir** можно передавать как относительный так и абсолютные пути к новому каталогу. Под абсолютным путем понимается путь от корневого каталога, т.е. путь начинающийся с символа **/**, под относительным путем понимается путь от текущего каталога. При этом предполагается, что все каталоги указанные в пути существуют, иначе их надо создавать отдельными командами или использовать команду

mkdir с опцией **-p** Так команда

$mkdir -p reports/lab3/part1

создаст каталог part1 и каталоги **reports** и **lab3**, если они не существуют.

Для удаления каталогов служит команда **rmdir**, например

$rmdir reports/lab1

удалит каталог lab1, расположенный в каталоге **reports**. С помощью **rmdir** можно удалять только пустые каталоги. Для удаления каталога со всем его содержимом, включая подкаталоги, можно использовать команду **rm** с опцией **-r**

Так команда

$rm -r reports/lab3

полностью удалит каталог **lab3**.

Для создания новый пустых файлов можно использовать команду **touch**

Для того, чтобы в каталоге **lab1** создать файл с именем **first.txt** необходимо выполнить команду

$touch ~/reports/lab1/first.txt

Здесь вместо символа **~** автоматически подставляется домашняя директория пользователя. Данную команду можно разбить на две

$cd ~/reports/lab1 — перейти в каталог **lab1**

$touch second.txt — создать в текущем каталоге файл с именем **second.txt**.

Для копирования файлов и каталогов используется команда cp (copy)

Для того, чтобы скопировать файл **first.txt** , находящийся в текущим каталоге, в файл **first.backup.txt** необходимо выполнить команду

$cp first.txt first.backup.txt

Стоит отметить, что в UNIX-системах символ "**.**" в имени файла не несет какого-либо особого смысла и может встречаться в имени файла произвольное количество раз. За исключением случая, когда имя файла или каталога начинается с "**.**". Это означает, что данный файл или каталог является скрытым. Для того, чтобы увидеть список скрытых файлов или каталогов, необходимо использовать **ls** с опцией **-a**.

Команда **cp** позволяет копировать несколько файлов

Также в команде **cp** можно использовать шаблонные выражения:

$cp *.txt ~/ - скопировать из текущего каталога все файлы, чьи имена заканчиваются на **.txt** в домашний каталог.

$cp a*.pdf ~/reports - скопировать из текущего каталога все файлы, чьи имена начинаются с буквы **a** и заканчиваются на **.pdf** в каталог **reports**.

Для копирования целых каталогов со всем их содержимым можно использовать команду **cp** с опцией **-r**

Для перемещения файлов и каталогов можно использовать команду **mv** (move). Для удаления файлов используется команда **rm**.

Работа с текстовым редактором

Для создания и изменения текстовых файлов можно использовать различные текстовые редакторы, данный инструмент является вторым по важности после командного интерпретатора, поскольку позволяет управлять ОС путем изменения файлов настроек и также позволяет вводить исходные коды программ на различных языках для последующей компиляции в выполнения.

Одним из достаточно функциональных текстовых редакторов является редактор **joe**, присутствующий во многих дистрибутивах GNU/Linux. Для вызова редактора нужно использовать команду **joe**. Так же можно сразу задать имя редактируемого файла

$joe file.my

если файла с таким именем не существует, то он будет создан при сохранении в редакторе.

Вызвать справку в редакторе **joe** можно сочетанием клавиш **Ctrl + K + H**. В файле справки под символом "**^**" подразумевается клавиша **Ctrl**.

Краткая справка по командам:

^K+D — сохранить

^K+X — выйти с сохранением

^C — выйти без сохранения

^T — вывести дополнительные опции

^T+N — вкл/выкл нумерацию строк

Более функциональным являются редакторы **vim** и **emacs**. Изучение широких возможностей данных редакторов выходит за рамки данного пособия. Для самостоятельного изучения можно использовать встроенные учебники:

$vimtutor — краткий учебник по **vim**

$emacs — запуск редактора **emacs** с возможностью вызвать учебник.

Дополнительные утилиты

Существует множество полезных утилит для работы с текстовыми файлами. Рассмотрим некоторые из них.

cat — утилита служит для конкатенации (сцепления файлов), но так же может быть использована для вывода содержимого файла на экран.

$cat file.my — выведет содержимое файла **file.my** на экран

Команда **wc** (word count) используется для подсчета символов, слов и строк в текстовых файлах.

$wc file.my — выведет количество строк, слов и символов в файле **file.my**.

wc можно использовать с различными опциями

$wc -c file.my — выведет количество символов (chars)

$wc -w file.my — выведет количество слов (words)

$wc -l file.my — выведет количество строк (lines)

Команда **grep** осуществляет вывод данных соответствующих заданному фильтру

$grep word file.my — выведет все строки из файла **file.my**, в которых встречается последовательность символов "word".

Команда **echo** выводит на экран свои параметры.

Команды **more** и **less** организуют постраничный вывод файлов на экран. Для постраничной навигации также удобно использовать сочетания **Shift+PageUp** и **Shift+PageDown**.

Перенаправление ввода вывода

Многие утилиты в среде UNIX работают с потоками ввода и вывода. При этом у каждой программы есть стандартный поток ввода (обычно клавиатура) стандартный поток вывода (монитор) и стандартный поток вывода ошибок (тоже монитор). Во многих оболочках имеется возможность изменять стандартные потоки, например, чтобы программа читала данные не с клавиатуры, а из файла, или чтобы программа выводила данные не на экран, а сразу записывала их в файл. Кроме того, можно сделать так, чтобы поток вывода одной программы являлся потоком ввода другой. Эти механизмы называются перенаправлением потоков и значительно увеличивают возможности системы по манипулированию различными данными.

Рассмотим этот механизм на конкретных примерах.

$cat file.my > file.my.new

команда **cat file.my** должна вывести содержимое файла **file.my** на экран, но в

данном случае мы указали, чтобы все данные были помещены в файл **file.my.new** с помощью спецсимвола "**>**" - перенаправления стандартного потока вывода. Таким образом, мы получили еще один способ копирования файлов.

$grep name file.my > names — сохраняет все строчки файла **file.my**, содержащие сочетание букв "name" в файле **names**.

При этом предыдущее содержимое файла names будет удалено, чтобы избежать этого можно добавить новые данные в конец файла с помощью символов "**»**"

$grep name file.my >> names — новые данные будут добавлены в конец файла names.

Для перенаправления потока ввода следует использовать символ "**<**"

$sort < file.my — отсортировать строки в файле **file.my** и вывести на экран.

Перенаправление можно комбинировать

$sort <file.my > file.my.sorted — отсортировать и сохранить в файле **file.my.sorted**

Для перенаправления потока вывода ошибок можно использовать символы "2>"

$prog 2>errors — запустить программу **prog** и все ошибки сохранить в файле **errors**

Потоки вывода и ошибок можно совместить

$ prog 2>&1 > output — перенаправлять поток ошибок (2) туда же куда и поток вывода (1), т.е. в файл **output**.

С помощью символа "|" можно организовать перенаправление потока вывода одной программы в поток ввода другой.

$cat file.my | grep name > names — содержимое файла **file.my** направить команде grep name и результат работы последней сохранить в файле **names**.

Такая составная команда называется конвейер. В конвейер можно объединять произвольное число команд

$cat file.my | grep name | wc -l — подсчитать количество строк в файле **file.my**, в которых встречается последовательность символов "name"

$cat words | uniq | sort — вывести в отсортированном виде содержимое файла **words** без дубликатов (команда **uniq** удаляет дубликаты).

Стоит отметить, что отдельной программы выполняющей такую функцию нет, но благодаря использованию конвейера такой результат можно получить одной командой.

Пользователи и группы

Любая современная UNIX система является многопользовательской и обеспечивает совместную и одновременную работу нескольких пользователей. При этом ОС следит за тем, чтобы пользователи не мешали работе друг друга и обеспечивает разграничение прав доступа к различным ресурсам, например таким как файлы. Пользователи могут быть включены в группы, обеспечивая, таким образом, более гибкое разделение прав пользования .

Пользователи различаются по именам и идентификаторам, чтобы узнать имя пользователя под чьей учетной записью вы работаете можно выполнить команду **whoami**. Чтобы узнать в каких группах вы числитесь можно выполнить команду **groups**. Более подробную информацию о пользователе и группах, включая идентификаторы, можно получить с помощью команды **id**.

Информация о пользователях традиционно хранится в файле **/etc/passwd**. Данный

файл является простым текстовым файлом, на каждого пользователя отводится по одной строчке. Чтобы просмотреть информацию о себе можно воспользоваться командой

$cat /etc/passwd | grep `whoami` - вместо команды **`whoami`** будет подставлен результат выполнения (такое поведение определяется использованием обратных кавычек), т.е. имя текущего пользователя и будет выведена только строчка касающаяся текущего пользователя.

В случае если информация о пользователях хранится удаленно следует использовать команду

```
$getent passwd | grep `whoami`
pmuser:x:1004:1005:,"Student",,:/home/pmuser:/bin/bash
```

Информация о пользователе состоит из нескольких полей разделенных символом **":"**. Разберем какие поля содержатся в этой строчке

pmuser — символьное имя пользователя (должно состоять только из строчных букв)

x — поле связанное с паролем пользователя, о нем будет упомянуто отдельно

1004 — **uid** (user identificator — идентификатор пользователя) число уникальное для каждого пользователя. Используется ОС поскольку ей удобней работать с числовыми идентификаторами, а не с символьными именами.

1005 — **gid** (group identificator — идентификатор группы) главная группа пользователя.

,**"Student"**,, — дополнительная информация о пользователе разделенная запятыми

/home/pmuser — домашний каталог пользователя

/bin/bash — оболочка (программа запускаемая при входе в систему).

Стоит отметить, что файл **/etc/passwd** доступен для чтения всем. Именно поэтому информация о паролях пользователей хранится отдельно в фале **/etc/shadow**. Сами пароли в файле **/etc/shadow** не хранятся. Там находятся только "хэши" паролей (одностороннее отображение пароля в символьную последовательность). Получить на основании пароля его "хэш" достаточно легко, но обратное (в случае сильного пароля) практически невозможно, поэтому восстановить забытый пароль нельзя. Администратор системы может только сменить пароль на новый.

Информация о группах хранится в файле **/etc/group**, в нем также через **":"** указаны имя группы, её **gid** и список пользователей входящих в данную группу. Информацию о всех группах можно получить с помощью команд:

```
$cat /etc/group
$getent group
```

Права доступа к файлам

Для каждого файла и каталога кроме имени ОС хранит различную информацию. Что бы вывести более подробную информацию по файлам и каталогам можно использовать команду **ls** с опцией **-l**

```
$ls -l
-rw-r--r-- 1 pmuser pmuser   21 Окт 22 17:10 file.my
drwxr-xr-x 2 pmuser pmuser 4096 Окт 22 17:10 reports
```

Для каждого файла выводится информация разделенная на столбцы. Разберем приведенный выше результат работы **ls -l**

-rw-r--r-- - тип файла и атрибуты доступа. Данное поле является очень важным для

понимания разграничения прав доступа и мы обязательно вернемся к нему позже.

1 — количество жестких ссылок на файл.

pmuser — имя пользователя владельца файла. В ОС каждый файл имеет своего владельца

pmuser - имя группы владельца файла. Кроме пользователя каждым файлом владеет одна из групп.

21 — размер файла в байтах

Окт 11 17:10 — время последней модификации файла

file.my — имя файла

Теперь разберем подробней первое поле, связанное с типом и атрибутами файла. Оно состоит из 10 полей, первое поле определяет тип файла остальные 9 — права доступа к нему.

Рассмотрим какие бывают типы файлов:

- обычный файл

d каталог

l символическая ссылка

c символьное устройство

b блочное устройство

s сокет

Обычно пользователю встречаются только обычные файлы, каталоги и символические ссылки, остальные типы относятся к системным ресурсам ОС. Стоит отметить тот факт, что каталог так же является файлом, хранящим имена файлов расположенные в нем. Такой факт может внести некоторую путаницу, но стоит понимать, что это сделано с единственной целью не плодить лишних сущностей в ОС. В случае с ОС семейства UNIX можно сказать, что в ней есть только файлы (все физические устройства также считаются файлами) и процессы (пользовательские и системные программы).

Все 9 полей прав доступа можно поделить на тройки из 3-х полей. В каждой из этом 3-х полей первое поле определяет наличие прав на запись(r - read), второе — прав на чтение(w - write) и третье - права на выполнение (x - execute). Первая тройка относится к владельцу файла, вторая к группе владельцу и последняя - ко всем остальным. Рассмотрим несколько примеров.

-rw-r--r-- - обычный файл со следующими правами:

Пользователь может читать и изменять файл, но не может запускать на выполнение (не может запускать как обычную программу), члены группы владельца могут только читать файл, все остальные могут только читать файл.

Для каталогов возможность записи означает возможность создания и удаления из каталога файлов, возможность чтения означает возможность просмотра содержимого каталога, а возможность исполнения — возможность переходить в данный каталог.

drwxr-xr-x — каталог, владелец которого имеет полные права, а челны группы и все остальные могут только просматривать содержимое и переходить в этот каталог.

Атрибуты доступа можно рассматривать как набор из 9 бит (примерно в таком виде и хранится данная информация), если соответствующий бит равен 0, то данных прав нет, если 1 то данное право есть. С учетом этого для удобства работы можно использовать восьмеричное представление прав доступа:

rw-r--r-- 644

rwxr-xr-x 755

rwxrwxrwx 777

rwx------ 700
rw------- 600

Для изменения прав доступа используется команда **chmod**

$chmod 600 file.my установить права 600 на файл **file.my**

Изменить владельца и группу можно командами **chown** и **chgrp**. Вносить такие изменения может только владелец файла или суперпользователь.

Среди всех пользователей обычно выделяют одного пользователя, у которого **uid** = 0 такой пользователь называется суперпользователем и традиционно имеет имя **root**. Отличительной особенностью суперпользователя является то, что на него не распространяются атрибуты доступа, и он обладает всеми правами на все файлы в системе. Под учетной записью суперпользователя должен работать только администратор системы.

Краткая справка по комбинациям клавиш в оболочке **bash**

Alt+b - переместить на начало предыдущего слова

Alt+F - переместить на начало следующего слова

Alt+d - удалить слово, в начале которого стоит указатель

Ctrl+b — перейти на символ назад

Ctrl+f - перейти на символ вперед

Ctrl+d — удалить символ

Ctrl+k - удалить все символы

Ctrl+p - восстановить предыдущую команду

Ctrl+u - удалить строку до курсора

Ctrl+e - переместиться в конец строки

Ctrl+l - очистить весь экран

Ctrl+c - прервать выполнение команды

Ctrl+d - нулевой символ

стрелка вверх (**Ctrl — p**) предыдущая команда

стрелка вниз (**Ctrl — n**) следующая команда

Shift+PageUp — пролистать экран вверх

Shift+PageDown — пролистать экран вниз