Перечень практических заданий
по дисциплине: Операционные системы

Практическая работа №1
Терминал и командная оболочка операционной системы Linux

Цель работы: Приобрести опыт работы с командной строкой ОС Linux, изучить основные команды (рабочая станция, рабочий директорий, пользователи, дата, календарь, список процессов, завершение работы)

План проведения занятия:

Ознакомиться с краткими теоретическими сведениями.

Приобрести навыки работы в терминале Linux. Научиться создавать новых пользователей при помощи терминала Linux, задавать несложные команды.

Подготовить отчет для преподавателя о выполнении лабораторной работы и представить его в соответствии с графиком.

Оборудование:

Аппаратная часть: персональный компьютер, сетевой или локальный принтер.
Программная часть: операционная система Linux Ubuntu, текстовый процессор Microsoft Word.

Краткие теоретические сведения:

Стандартные команды в Linux отличаются от команд DOS и Windows - обычно они короче. При работе с командной строкой как обычно мигающий курсор обозначает позицию ввода текста , командная строка начинается с текущего пути и имени компьютера, за которым следует символ $, % или #. Последний означает, что команды будут выполняться от имени суперпользователя root. Символ ~ означает путь к текущей домашней директории пользователя.

Большинство команд в Linux, не требующих вывода информации пользователю, в случае успешного завершения вообще ничего не выводят на экран. Выводятся только ошибки и предупреждения в случае нарушения нормального выполнения команды. Т.е. в Linux действует общий принцип "молчит, значит работает".

В любом терминале Linux стрелками вверх/вниз на клавиатуре можно листать историю команд, которая сохраняется между сеансами работы и различается для разных пользователей и хостов. Набранное частично команда или имя файла или каталога в текущей директории может быть автоматически дописано клавишей TAB. Если найдено более одного варианта и однозначно продолжить команду по TAB невозможно, то выводятся все подходящие варианты.

При работе в графической среде удобны эмуляторы терминала. Как правило они поддерживают закладки - несколько терминалов в одном окне, поддерживают цветовые схемы. Наиболее распространены эмуляторы терминала Gnome Terminal, Konsole, XFCE Terminal.

Терминал — эмулятор консоли. Именно в терминале мы будем работать с CLI (интерфейсом командной строки). Терминал часто также называют консолью или шеллом (от англ. shell — оболочка). В будущем для объяснения я буду использовать все три эти понятия, главное не забывайте что они синонимы.

Многие пользователи и в особенности администраторы серверов под Linux в работе используют именно консоль, а не графическую оболочку, это связано с тем, что настройка и конфигурация Linux в основном заключается, в редактировании текстовых конфигурационных файлов. Даже если вы являетесь простым пользователем ОС Linux,

большинство инструкций по настройке написаны с использованием консоли и знать основные команды жизненно необходимо.

Стоит обратить внимание на системные каталоги ОС в которых находятся файлы, необходимые для управления и сопровождения системы, а также стандартные программы. Их имена, расположение и содержание почти одинаковы почти во всех ОС Linux, поэтому эти каталоги называют также стандартными. Впрочем, на данный момент эпитет «стандартные» отражает скорее благие пожелания, чем действительность: иерархия каталогов одинакова только для дистрибутивов, связанных единством происхождения, а исторически сложившиеся различия создают опасность несовместимости разных дистрибутивов.

С точки зрения UNIX-подобных ОС, файл представляет собой поток или последовательность байтов. Такой подход позволяет распространить понятие файла на множество ресурсов не только локального компьютера, но и удаленного, связанного с локальным сетью любого рода. Доступ к любому такому ресурсу осуществляется через универсальный интерфейс, благодаря чему запись данных в файл, отправка их на физическое устройство или обмен ими с другой работающей программой происходит аналогично. Это очень упрощает организацию данных и обмен ими.

В ОС Linux можно выделить следующие типы файлов:

- обычные файлы — последовательность байтов (текстовые документы, исполняемые программы, библиотеки и т.п.);
- каталоги — именованные наборы ссылок на другие файлы;
- файлы физических устройств, подразделяющихся на:

o файлы блочных устройств, драйверы которых буферизуют ввод-вывод с помощью ядра и файлы байт-ориентированных, или символьных, устройств, позволяющих связанным с ними драйверам выполнять буферизацию собственными средствами;

o символические ссылки (symlink, symbolic link); o
именованные каналы (named pipes);
o гнезда (sockets).

Таблица 2.

halt	стремительное и корректное выключение системы.
poweroff	корректное выключение системы.
reboot	корректное выключение с последующей загрузкой.
adduser	создание нового пользователя.
date	показывает нынешние дату и время, по системным часам ядра.
oclock	обычные часы
finger	отображение информации о пользователе
hostname	команда показывает личный номер этого узла сети
hwclock	интегрированные часы
uname	выводит информацию об используемой операционной системе
uptime	проявляет текущее время, длительность сеанса, число пользователей и загруженность процессора.
usermod	изменение параметров пользователя.
users	отражает короткий перечень пользователя работающих в системе в этот эпизод
whoami	демонстрирует нынешний личный номер пользователя, работающего в этом терминале.
write	посылает известие иному пользователя, окружающему в системе, маршрутом копирования строчек с терминала отправителя на терминал получателя.

history	демонстрирует пронумерованный перечень команд, которые Вы исполняли в данном и прошлом сеансе. Само собой разумеется, что если в перечне истории их очень немало, то увидите заключительные.
passwd	изменение пароля пользователя
ps	выводит перечень всех работающих действий.
times	проявляет абсолютное время исполнения действий для всей системы и этого пользователя.
free	отражает информацию о своевременной памяти, подкачки, кэше, свободная память, общественная и т.п.
ls	указывает все файлы в текущем каталоге в алфавитном порядке. По всей вероятности аналогична dir.
clear	чистит экран терминала (в случае если данное вполне вероятно).
ifconfig	отражает состояние текущей конфигурации сети или же настраивает сетевой интерфейс.
less	отражает содержимое указанного файла на экране и дозволяет комфортно просмотреть.
mkpasswd	создает качественный пароль, состоящий по умолчанию из 9 знаков и имеющий как минимум буквы в различном регистре и числа.

Ход работы:

Для выполнения данной работы будем использовать ранее установленный Linux Ubuntu. Запускаем Linux. После прохождения идентификации включаем терминал (рис. 34).

Applications > Accessories > Terminal

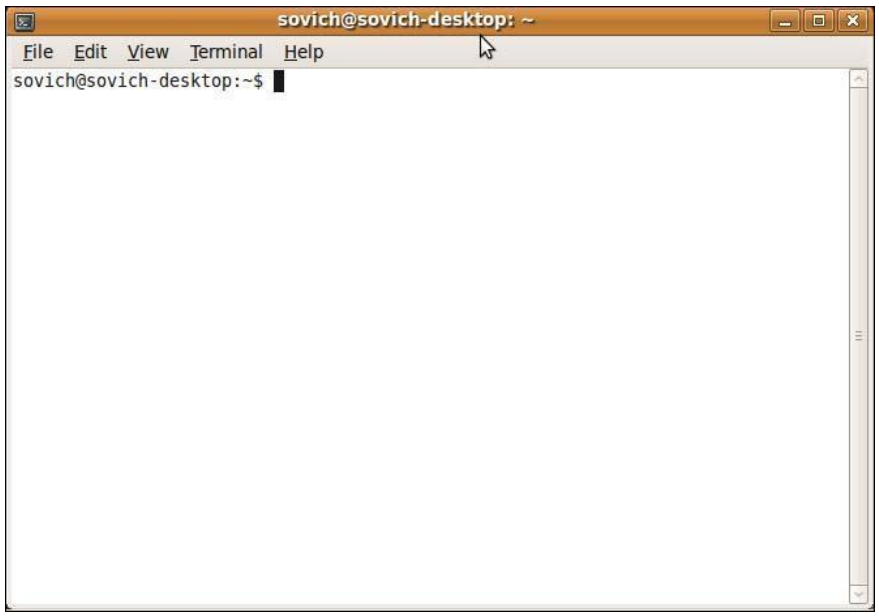


Рис. 34. Терминал Linux Ubuntu

2. Для работы в терминале Ubuntu требуются права пользователя root, но, к сожалению, по умолчанию, он недоступен, поэтому для выполнения некоторых (не всех) команд надо писать sudo <команда>, и подтверждать свои права вводом пароля. И не пугайтесь того, что его не видно в терминале! Наберите точно по памяти, по окончании ввода нажмите Enter.
3. Для получении справки о дополнительных возможностях некоторых программ следует набрать <команда> --help
4. Потренируйтесь в выполнении команд:

- date
- oclock
- finger
- hwclock
- uname
- history
- clear
- ls

Найдите данные команды в таблице 2, опишите их. После выполнения результат внесите в отчет.

5. Создайте нового пользователя, при помощи терминала Ubuntu, и введите его в группу admin. Создайте пароль пользователю. Войдите под ним в систему. Процесс создания и ввода в группу внесите в отчет.
6. Разберите выполнение незадействованных команд таблицы 2. Потренируйтесь в выполнении, определите их назначение и область применения. Результат работы внесите в отчет.
7. Подготовьте отчет о выполнении лабораторной работы и сдайте преподавателю в соответствии с графиком.

Контрольные вопросы:

1. Что такое терминал?
2. Перечислите основные системные каталоги.
3. Расскажите о типах файлов в ОС Linux.

Практическая работа №2
Работа с файловой системой ОС Linux

Цель работы: Приобрести опыт работы с файлами и каталогами в ОС Linux, настройки

прав на доступ к файлам и каталогам.

План проведения занятия:

Ознакомиться с краткими теоретическими сведениями. Приобрести навыки работы в терминале Linux.

Научиться создавать новые файлы и каталоги, разобрать назначение прав доступа к файлам и папкам.

Подготовить отчет для преподавателя о выполнении лабораторной работы и представить его в соответствии с графиком.

Оборудование:

Аппаратная часть: персональный компьютер, сетевой или локальный принтер.

Программная часть: операционная система Linux Ubuntu, текстовый процессор Microsoft Word.

Краткие теоретические сведения

Файловая система - это структура, с помощью которой ядро операционной системы предоставляет пользователям (и процессам) ресурсы долговременной памяти системы, т. е. памяти на различного вида долговременных носителях информации - жестких дисках, магнитных лентах, CD-ROM и т. п.

С точки зрения пользователя, файловая система — это логическая структура каталогов и файлов. В отличие от Windows, где каждый логический диск хранит отдельное дерево каталогов, во всех UNIX-подобных системах эта древовидная структура растет из одного корня: она начинается с корневого каталога, родительского по отношению ко всем остальным, а физические файловые системы разного типа, находящиеся на разных разделах и даже на удаленных машинах, представляются как ветви этого дерева.

Имена файлов и каталогов могут иметь длину до 255 символов. Символы «/» (слэш) и символ с кодом 0 запрещены. Кроме того, ряд символов имеет специальное значение для командного интерпретатора, и их использование не рекомендуется. Это символы:
~ ! @ # $ & % * ( ) [ ] { } ' " \ : ; > < пробел Заметьте, что точки среди специальных символов нет, и имена вроде
this.is.a.text.file.containing.the.famous.string.hello.world допустимы и широко распространены. Часто последняя отделенная точкой часть имени используется подобно «расширению имени» в Windows, обозначая файл определенного типа, но это обозначение несет смысл только для человека. Так, человеку имя файла ivan_home_tar.gz подсказывает, что это домашний каталог пользователя ivan, упакованный архиватором tar и сжатый компрессором gzip.

Если имя файла начинается с точки, то этот файл считается скрытым: некоторые команды его «не видят». Например, введя в своем домашнем каталоге команду просмотра содержимого каталога ls с ключом - a, означающим «показывать скрытые файлы», вы увидите больше файлов, чем введя ту же команду без ключей.

Linux различает регистр символов в именах файлов: так, в одном каталоге могут находиться два разных файла README и Readme.

Имена каталогов строятся по точно тем же правилам, что и имена файлов.

Полным именем файла (или путем к файлу) называется список вложенных друг в друга каталогов, заканчивающийся собственно именем файла. Начинаться он может с любого каталога, потому что в древовидной структуре между любыми двумя узлами существует путь. Если этот список начинается с корневого каталога, то путь называется абсолютным. Если с любого другого — то относительным (по отношению к этому каталогу).

Корневой каталог обозначается символом «/» (слэш), и этим же символом разделяются имена каталогов в списке. Таким образом, абсолютным именем файла README в домашнем каталоге пользователя ivanov будет /home/petrov/README.

В каждом каталоге существуют два особых «подкаталога» с именами «две точки» и «точка». Первый из них служит указанием на однозначно определенный родительский каталог, а второй — на сам данный каталог. Для корневого каталога , у которого нет

родителя , оба эти «подкаталога» указывают на корневой каталог. С помощью этих имен образуются относительные имена файлов. Так, именем вышеупомянутого файла README относительно домашнего каталога /home/ivanov
пользователя ivan будет ../petrov/README.

Жесткая ссылка является просто другим именем для исходного файла . После создания такой ссылки ее невозможно отличить от исходного имени файла. «Настоящего» имени у файла нет, точнее, все такие имена будут настоящими. Команда ls показывает количество именно таких жестких ссылок. Удаление файла по любому из его имен уменьшает на единицу количество ссылок, и окончательно файл будет удален только тогда, когда это количество станет равным нулю. Поэтому удобно использовать жесткие ссылки для того, чтобы предотвратить случайное удаление важного файла.

Создадим жесткую ссылку на файл README и посмотрим, что изменилось в его свойствах:

```
$ln   /home/ivanov/README /home/ivanov/readme_too
$ls   -1 /home/ivanov/README
-rwxr-xr-- 2 ivanov          users          0  Feb 14 19:08 /home/ivanov/README
```

Жесткую ссылку можно создавать в любом каталоге, но обязательно на том же физическом носителе (то есть в той же файловой системе), что и исходный файл. О причине этого будет сказано позже.

Другой тип ссылок представляют собой символические ссылки. По назначению они аналогичны ярлыкам в ОС Windows: указывают на файл , расположенный где угодно (например, на съемном носителе), и после удаления такого файла или размонтирования съемного носителя становятся бесполезны.

Символическая ссылка создается той же командой ln с ключом -s: $ln -s /home/ivanov/README /home/ivanov/do.not.readme $ls -1 /home/ivanov/do.not.readme
lrwxrwxrwx 1 ivanov users 16 Feb 14 19:17 /home/ivanov/do.not. readme ->
/home/ivanov/README

В поле имени файла после стрелки указано его настоящее имя. Права доступа у всех символических ссылок одинаковы и не значат ничего: возможность доступа к файлу определяется правами исходного файла. Заметьте, что в отличие от файла-оригинала файл-ссылка имеет ненулевую длину: в нем хранится абсолютное имя исходного файла. Попытайтесь вывести файл-ссылку на экран с помощью команды cat, и вы увидите содержание исходного, пустого, файла:

```
$ cat /home/ivanov/do.not.readme
```

Значение самой ссылки, то есть имя файла, на который она ссылается, можно узнать с помощью команды readlink.

Права доступа к файлам и каталогам

Поскольку Linux - система многопользовательская, вопрос об организации разграничения доступа к файлам и каталогам является одним из существенных вопросов, которые должна решать операционная система. Механизмы разграничения доступа, разработанные для системы UNIX в 70-х годах (возможно, впрочем, они предлагались кемто и раньше), очень просты, но они оказались настолько эффективными, что просуществовали уже более 30 лет и по сей день успешно выполняют стоящие перед ними задачи.

В основе механизмов разграничения доступа лежат имена пользователей и имена групп пользователей. Вы уже знаете, что в Linux каждый пользователь имеет уникальное имя, под которым он входит в систему (логируется). Кроме того, в системе создается некоторое число групп пользователей, причем каждый пользователь может быть включен в одну или несколько групп. Создает и удаляет группы суперпользователь, он же может изменять состав участников той или иной группы. Члены разных групп могут иметь разные права по доступу к файлам, например, группа администраторов может иметь больше прав, чем группа программистов.

В индексном дескрипторе каждого файла записаны имя так называемого владельца

файла и группы, которая имеет права на этот файл. Первоначально, при создании файла его владельцем объявляется тот пользователь, который этот файл создал. Точнее - тот пользователь, от чьего имени запущен процесс, создающий файл. Группа тоже назначается при создании файла - по идентификатору группы процесса, создающего файл. Владельца и группу файла можно поменять в ходе дальнейшей работы с помощью команд chown и chgrp (подробнее о них будет сказано чуть позже).

Теперь давайте еще раз выполним команду ls -l. Но зададим ей в качестве дополнительного параметра имя конкретного файла, например, файла, задающего саму команду ls. (Обратите, кстати, внимание на эту возможность команды ls -l - получить информацию о конкретном файле, а не о всех файлах каталога сразу):

```
[user]$ ls -l /bin/ls
-rwxr-xr-x 1 root root 49940 Sep 12 1999 /bin/ls
```

Вы видие, что в данном случае владельцем файла является пользователь root и группа root. Но нас сейчас в выводе этой команды больше интересует первое поле, определяющее тип файла и права доступа к файлу. Это поле в приведенном примере представлено цепочкой символов -rwxr-xr-x. Эти символы можно условно разделить на 4 группы.

Првая группа, состоящая из единственного символа, определяет тип файла. Этот символ в соответствии с возможными типами файлов, рассмотренными в предыдущем разделе, может принимать такие значения:

- - = - обычный файл;
- d = - каталог;
- b = - файл блочного устройства;
- c = - файл символьного устройства;
- s = - доменное гнездо (socket);
- p = - именованный канал (pipe);
- l = - символическая ссылка (link).

Далее следуют три группы по три символа, которые и определяют права доступа к файлу соответственно для владельца файла, для группы пользователей, которая сопоставлена данному файлу, и для всех остальных пользователей системы. В нашем примере права доступа для владельца определены как rwx, что означает, что владелец (root) имеет право читать файл (r), производить запись в этот файл (w), и запускать файл на выполнение (x). Замена любого из этих символов прочерком будет означать, что пользователь лишается соответствующего права. В том же примере мы видим, что все остальные пользователи (включая и тех, которые вошли в группу root) лишены права записи в этот файл, т. е. не могут файл редактировать и вообще как-то изменять.

Вообще говоря, права доступа и информация о типе файла в UNIX-системах хранятся в индексных дескрипторах в отдельной структуре, состоящей из двух байтов, т. е. из 16 бит (это естественно, ведь компьютер оперирует битами, а не символами r, w, x). Четыре бита из этих 16-ти отведены для кодированной записи о типе файла. Следующие три бита задают особые свойства исполняемых файлов, о которых мы скажем чуть позже. И, наконец, оставшиеся 9 бит определяют права доступа к файлу. Эти 9 бит разделяются на 3 группы по три бита. Первые три бита задают права пользователя, следующие три бита - права группы, последние 3 бита определяют права всех остальных пользователей (т. е. всех пользователей, за исключением владельца файла и группы файла).

При этом, если соответствующий бит имеет значение 1, то право предоставляется, а если он равен 0, то право не предоставляется. В символьной форме записи прав единица заменяется соответствующим символом (r, w или x), а 0 представляется прочерком.

Право на чтение (r) файла означает, что пользователь может просматривать содержимое файла с помощью различных команд просмотра, например, командой more или

с помощью любого текстового редактора. Но, подредактировав содержимое файла в текстовом редакторе, вы не сможете сохранить изменения в файле на диске, если не имеете права на запись (w) в этот файл. Право на выполнение (x) означает, что вы можете загрузить файл в память и попытаться запустить его на выполнение как исполняемую программу. Конечно, если в действительности файл не является программой (или скриптом shell), то запустить этот файл на выполнение не удастся, но, с другой стороны, даже если файл действительно является программой, но право на выполнение для него не установлено, то он тоже не запустится.

Вот мы и узнали, какие файлы в Linux являются исполняемыми! Как видите, расширение имени файла тут не при чем, все определяется установкой атрибута "исполняемый", причем право на исполнение может быть предоставлено не всем!

Если выполнить ту же команду ls -l, но в качестве последнего аргумента ей указать не имя файла, а имя каталога, мы увидим, что для каталогов тоже определены права доступа, причем они задаются теми же самыми символами rwx. Например, выполнив команду ls -l /, мы увидим, что каталогу bin соответствует строка:

drwxr-xr-x 2 root root 2048 Jun 21 21:11 bin

Естественно, что по отношению к каталогам трактовка понятий "право на чтение", "право на запись" и "право на выполнение" несколько изменяется. Право на чтение по отношению к каталогам легко понять, если вспомнить, что каталог - это просто файл, содержащий список файлов в данном каталоге. Следовательно, если вы имеете право на чтение каталога, то вы можете просматривать его содержимое (этот самый список файлов в каталоге). Право на запись тоже понятно - имея такое право, вы сможете создавать и удалять файлы в этом каталоге, т. е. просто добавлять в каталог или удалять из него запись, содержащую имя какого-то файла и соответствующие ссылки. Право на выполнение интуитивно менее понятно. Оно в данном случае означает право переходить в этот каталог. Если вы, как владелец, хотите дать доступ другим пользователям на просмотр какого-то файла в своем каталоге, вы должны дать им право доступа в каталог, т. е. дать им "право на выполнение каталога". Более того, надо дать пользователю право на выполнение для всех каталогов, стоящих в дереве выше данного каталога. Поэтому в принципе для всех каталогов по умолчанию устанавливается право на выполнение как для владельца и группы, так и для всех остальных пользователей. И, уж если вы хотите закрыть доступ в каталог, то лишите всех пользователей (включая группу) права входить в этот каталог. Только не лишайте и себя такого права, а то придется обращаться к суперпользователю![3)]

После прочтения предыдущего абзаца может показаться, что право на чтение каталога не дает ничего нового по сравнению с правом на выполнение. Однако разница в этих правах все же есть. Если задать только право на выполнение, вы сможете войти в каталог, но не увидите там ни одного файла (этот эффект особенно наглядно проявляется в том случае, если вы пользуетесь каким-то файловым менеджером, например, программой Midnight Commander). Если вы имеете право доступа в каком- то из подкаталогов этого каталога, то вы можете перейти в него (командой cd), но, как говорится "вслепую", по памяти, потому что списка файлов и подкаталогов текущего каталога вы не увидите.

Алгоритм проверки прав пользователя при обращении к файлу можно описать следующим образом. Система вначале проверяет, совпадает ли имя пользователя с именем владельца файла. Если эти имена совпадают (т. е. владелец обращается к своему файлу), то проверяется, имеет ли владелец соответствующее право доступа: на чтение, на запись или на выполнение (не удивляйтесь, суперпользователь может лишить некоторых прав и владельца файла). Если право такое есть, то соответствующая операция разрешается. Если же нужного права владелец не имеет, то проверка прав, предоставляемых через группу или через группу атрибутов доступа для остальных пользователей, уже даже не проверяются, а пользователю выдается сообщение о невозможности выполнения затребованного действия (обычно что-то вроде "Permission denied").

Если имя пользователя, обращающегося к файлу, не совпадает с именем владельца,

то система проверяет, принадлежит ли владелец к группе, которая сопоставлена данному файлу (далее будем просто называть ее группой файла). Если принадлежит, то для определения возможности доступа к файлу используются атрибуты, относящиеся к группе, а на атрибуты для владельца и всех остальных пользователей внимания не обращается. Если же пользователь не является владельцем файла и не входит в группу файла, то его права определяются атрибутами для остальных пользователей. Таким образом, третья группа атрибутов, определяющих права доступа к файлу, относится ко всем пользователям, кроме владельца файла и пользователей, входящих в группу файла.

Для изменения прав доступа к файлу используется команда chmod. Ее можно использовать в двух вариантах. В первом варианте вы должны явно указать, кому какое право даете или кого этого права лишаете:

[user]$ chmod wXp имя-файла

где вместо символа w подставляется

- либо символ u (т. е. пользователь, который является владельцем);
- либо g (группа);
- либо o (все пользователи, не входящие в группу, которой принадлежит данный файл);
- либо a (все пользователи системы, т. е. и владелец, и группа, и все остальные). Вместо X ставится:
- либо + (предоставляем право);
- либо - (лишаем соответствующего права);
- либо = (установить указанные права вместо имеющихся).

Вместо p - символ, обозначающий соответствующее право:

- r (чтение);
- w (запись);
- x (выполнение).

Вот несколько примеров использования команды chmod:

[user]$ chmod a+x file_name

предоставляет всем пользователям системы право на выполнение данного файла.

[user]$ chmod go-rw file_name

удаляет право на чтение и запись для всех, кроме владельца файла.

[user]$ chmod ugo+rwx file_name

дает всем права на чтение, запись и выполнение.

Если опустить указание на то, кому предоставляется данное право, то подразумевается, что речь идет вообще обо всех пользователях, т. е. вместо

[user]$ chmod a+x file_name

можно записать просто

[user]$ chmod +x file_name

Второй вариант задания команды chmod (он используется чаще) основан на цифровом представлении прав. Для этого мы кодируем символ r цифрой 4, символ w - цифрой 2, а символ x - цифрой 1. Для того, чтобы предоставить пользователям какой-то набор прав, надо сложить соответствующие цифры. Получив, таким образом, нужные цифровые значения для владельца файла, для группы файла и для всех остальных пользователей, задаем эти три цифры в качестве аргумента команды chmod (ставим эти цифры после имени команды перед вторым аргументом, который задает имя файла). Например, если надо дать все права владельцу (4+2+1=7), право на чтение и запись - группе (4+2=6), и не давать никаких прав остальным, то следует дать такую команду:

[user]$ chmod 760 file_name

Цифры после имени команды в этой форме ее представления есть не что иное, как восьмеричная запись тех самых 9 бит, которые задают права для владельца файла, группы файла и для всех пользователей.

Выполнять смену прав доступа к файлу с помощью команды chmod может только сам владелец файла или суперпользователь. Для того, чтобы иметь возможность изменить

права группы, владелец должен дополнительно быть членом той группы, которой он хочет дать права на данный файл.

Чтобы завершить рассказ о правах доступа к файлам, надо рассказать еще о трех возможных атрибутах файла, устанавливаемых с помощью той же команды chmod. Это те самые атрибуты для исполняемых файлов, которые в индексном дескрипторе файла в двухбайтовой структуре, определяющей права на файл, занимают позиции 5-7, сразу после кода типа файла.

Первый из этих атрибутов - так называемый "бит смены идентификатора пользователя". Смысл этого бита состоит в следующем.

Обычно, когда пользователь запускает некоторую программу на выполнение, эта программа получает те же права доступа к файлам и каталогам, которые имеет пользователь, запустивший программу. Если же установлен "бит смены идентификатора пользователя", то программа получит права доступа к файлам и каталогам, которые имеет владелец файла программы (таким образом, рассматриваемый атрибут лучше называть "битом смены идентификатора владельца"). Это позволяет решать некоторые задачи, которые иначе было бы трудно выполнить. Самый характерный пример - команда смены пароля passwd. Все пароли пользователей хранятся в файле /etc/passwd, владельцем которого является суперпользователь root. Поэтому программы, запущенные обычными пользователями, в том числе команда passwd, не могут производить запись в этот файл. А, значит, пользователь как бы не может менять свой собственный пароль. Но для файла /usr/bin/passwd установлен "бит смены идентификатора владельца", каковым является пользователь root. Следовательно, программа смены пароля passwd запускается с правами root и получает право записи в файл /etc/passwd (уже средствами самой программы обеспечивается то, что пользователь может изменить только одну строку в этом файле).

Установить "бит смены идентификатора владельца" может суперпользователь с помощью команды

```
[root]# chmod +s file_name
```

Аналогичным образом работает "бит смены идентификатора группы".

Еще один возможный атрибут исполняемого файла - это "бит сохранения задачи" или "sticky bit" (дословно - " бит прилипчивости"). Этот бит указывает системе, что после завершения программы надо сохранить ее в оперативной памяти. Удобно включить этот бит для задач, которые часто вызываются на выполнение, так как в этом случае экономится время на загрузку программы при каждом новом запуске. Этот атрибут был необходим на старых моделях компьютеров. На современных быстродействующих системах он используется редко.

Если используется цифровой вариант задания атрибутов в команде chmod, то цифровое значение этих атрибутов должно предшествовать цифрам, задающим права пользователя:

```
[root]# chmod 4775 file_name
```

При этом веса этих битов для получения нужного суммарного результата задаются следующим образом:

- 4 - "бит смены идентификатора пользователя",
- 2 - "бит смены идентификатора группы",
- 1 - "бит сохранения задачи (sticky bit)".

Если какие-то из этих трех битов установлены в 1, то несколько изменяется вывод команды ls -l в части отображения установленных атрибутов прав доступа. Если установлен в 1 "бит смены идентификатора пользователя", то символ "x" в группе, определяющей права владельца файла, заменяется символом "s". Причем, если владелец имеет право на выполнение файла, то символ "x" заменяется на маленькое "s", а если владелец не имеет права на выполнение файла (например, файл вообще не исполняемый), то вместо "x" ставится "S". Аналогичные замены имеют место при задании "бита смены идентификатора группы", но заменяется символ "x" в группе атрибутов, задающих права группы. Если равен

1 "бит сохранения задачи (sticky bit)", то заменяется символ "x" в группе атрибутов, определяющей права для всех остальных пользователей, причем "x" заменяется символом "t", если все пользователи могут запускать файл на выполнение, и символом "T", если они такого права не имеют.

Таким образом, хотя в выводе команды ls -l не предусмотрено отдельных позиций для отображения значений битов смены идентификаторов и бита сохранения задачи, соответствующая информация выводится. Вот небольшой пример того, как это будет выглядеть:

```
[root]# ls -l prim1
-rwSrwsrwT 1 kos root 12 Dec 18 23:17 prim1
```

Команды для работы с файлами и каталогами

В предыдущих разделах и работах уже упоминали некоторые команды для работы с файлами и каталогами: pwd (имя текущего каталога), cd, ls, ln, chmod. В этом разделе рассмотрим еще несколько часто используемых команд.

Команды chown и chgrp

Эти команды служат для смены владельца файла и группы файла. Выполнять смену владельца может только суперпользователь, смену группы может выполнить сам владелец файла или суперпользователь. Для того, чтобы иметь право сменить группу, владелец должен дополнительно быть членом той группы, которой он хочет дать права на данный файл. Формат этих двух команд аналогичен:

```
[root]# chown vasja имя-файла [root]# chgrp usersgrp имя-файла
```

Команда mkdir

Команда mkdir позволяет создать подкаталог в текущем каталоге. В качестве аргумента этой команде надо дать имя создаваемого каталога. Во вновь созданном каталоге автоматически создаются две записи: . (ссылка на этот самый каталог) и .. (ссылка на родительский каталог). Чтобы создать подкаталог, вы должны иметь в текущем каталоге право записи. Можно создать подкаталог не в текущем, а в каком-то другом каталоге, но тогда необходимо указать путь к создаваемому каталогу:

```
[user]$ mkdir /home/kos/book/glava5/part1
```

Команда mkdir может использоваться со следующими опциями:

- -m mode - задает режим доступа для нового каталога (например, -m 755);
- -p - создавать указанные промежуточные каталоги (если они не существуют).

Команда touch

Создать пустой файл можно командой touch <имя_файла>. Вообще-то она предназначена для того, чтобы для всех заинтересованных программ (например, утилиты сборки проекта **make)** файл выглядел новее, чем на самом деле: она меняет время последнего изменения файла на текущее время. Но если файла с таким именем не существует, то она его создаст.

Команда cat

Команда cat часто используется для создания файлов (хотя можно воспользоваться и командой touch). По команде cat на стандартный вывод (т. е. на экран) выводится содержимое указанного файла (или нескольких файлов, если их имена последовательно задать в качестве аргументов команды). Если вывод команды cat перенаправить в файл, то можно получить копию какого-то файла:

```
[user]$ cat file1 > file2
```

Собственно, первоначальное предназначение команды cat как раз и предполагало перенаправление вывода, так как эта команда создана для конкатенации, т. е. объединения нескольких файлов в один:

```
[user]$ cat file1 file2 ... fileN > new-file
```

Именно возможности перенаправления ввода и вывода этой команды и используются для создания новых файлов. Для этого на вход команды cat направляют

данные со стандартного ввода (т. е. с клавиатуры), а вывод команды - в новый файл:
[user]$ cat > newfile

После того, как вы напечатаете все, что хотите, нажмите комбинацию клавиш <Ctrl>+<D> или <Ctrl>+<C>, и все, что вы ввели, будет записано в newfile. Конечно, таким образом создаются, в основном, короткие текстовые файлы.

Команда mv

Если вам необходимо не скопировать, а переместить файл из одного каталога в другой, вы можете воспользоваться командой mv. Синтаксис этой команды аналогичен синтаксису команды cp. Более того, она сначала копирует файл (или каталог), а только потом удаляет исходный файл (каталог). И опции у нее такие же, как у cp.[4)]

Команда mv может использоваться не только для перемещения, но и для переименования файлов и каталогов (т. е. перемещения их внутри одного каталога). Для этого надо просто задать в качестве аргументов старое и новое имя файла:
[user]$ mv oldname newname

Но учтите, что команда mv не позволяет переименовать сразу несколько файлов (используя шаблон имени), так что команда mv *.xxx *.yyy не будет работать.

При использовании команды mv, также как и при использовании cp, не забывайте применять опцию -i для того, чтобы получить предупреждение, когда файл будет перезаписываться.

Команды rm и rmdir

Для удаления ненужных файлов и каталогов в Linux служат команды rm (удаляет файлы) и rmdir (удаляет пустой каталог). Для того, чтобы воспользовался этими командами, вы должны иметь право записи в каталоге, в котором расположены удаляемые файлы или каталоги. При этом полномочия на изменение самих файлов не обязательны. Если хотите перед удалением файла получить дополнительный запрос на подтверждение операции, используйте опцию -i.

Если вы попытаетесь использовать команду rm (без всяких опций) для удаления каталога, то будет выдано сообщение, что это каталог, и удаления не произойдет. Для удаления каталога надо удалить в нем все файлы, после чего удалить сам каталог с помощью команды rmdir. Однако можно удалить и непустой каталог со всеми входящими в него подкаталогами и файлами, если использовать команду rm с опцией -r.

Если вы дадите команду rm *, то удалите все файлы в текущем каталоге. Подкаталоги при этом не удалятся. Для удаления как файлов, так и подкаталогов текущего каталога надо тоже воспользоваться опцией -r. Однако всегда помните, что в Linux нет команды восстановления файлов после их удаления (даже если вы спохватились сразу же после ошибочного удаления файла или каталога)!
Так что дважды подумайте до удаления чего-либо и не пренебрегайте опцией -i.

Команды more и less

Команда cat позволяет вывести на стандартный вывод (на экран) содержимое любого файла, однако она используется для этих целей очень редко, разве что для вывода очень небольших по объему файлов. Дело в том, что содержимое большого файла мгновенно проскакивает на экране, и пользователь видит только последние строки файла. Поэтому cat используется в основном по ее прямому назначению - для конкатенации файлов, а для просмотра содержимого файлов (конечно, текстовых) используются команды more и less (или текстовые редакторы).

Команда-фильтр more выводит содержимое файла на экран отдельными страницами, размером как раз в целый экран. Для того, чтобы увидеть следующую страницу, надо нажать на клавишу пробела. Нажатие на клавишу <Enter> приводит к смещению на одну строку. Кроме клавиш пробела и <Enter> в режиме паузы еще некоторые клавиши действуют как управляющие (например, клавиша <B> возвращает вас на один экран назад), но мы здесь не будем приводить полного их перечня, как и перечня опций команды. Вам для начала надо еще только запомнить, что выйти из режима просмотра можно с помощью

клавиши <Q>, так как если вы этого не знаете, то вам придется долго и нудно нажимать пробел, пока вы не доберетесь до конца длинного файла. Обо всех опциях команды more вы можете прочитать в интерактивном руководстве man или info.

Утилита less, разработанная в рамках проекта GNU, содержит все функции и команды управления выводом, имеющиеся в программе more, и некоторые дополнительные, например, позволяет использовать клавиши управления курсором (<Стрелка вверх>, <Стрелка вниз>, <PgUp>, <PgDown>) для перемещения по тексту . Вспомните, мы уже говорили об этом, когда рассматривали интерактивную подсказку man.

Команды more и less позволяют производить поиск подстроки в просматриваемом файле, причем команда less позволяет производить поиск как в прямом, так и в обратном направлении. Для организации поиска строки символов string надо набрать в командной строке программы в нижней части экрана (там, где двоеточие) /string. Если искомая строка будет найдена, будет отображен соответствующий кусок текста, причем найденная строка будет находиться в самом верху экрана.

Команда find и символы шаблонов для имен файлов

Еще одной часто используемой командой для работы с файлами в Linux является команда поиска нужного файла find. Команда find может искать файлы по имени, размеру, дате создания или модификации и некоторым другим критериям.

Общий синтаксис команды find имеет следующий вид: find [список_каталогов] критерий_поиска

Параметр "список_каталогов" определяет, где искать нужный файл. Проще всего задать в качестве начального каталога поиска корневой каталог /, однако, в таком случае поиск может затянуться очень надолго, так как будет просматриваться вся структура каталогов, включая смонтированные файловые системы (в том числе сетевые, если таковые есть). Можно сократить объем поиска, если задать вместо одного корневого каталога список из нескольких каталогов (естественно, тех, в которых может находиться искомый файл):

```
[user]$ find /usr/share/doc /usr/doc /usr/locale/doc -name instr.txt
```

Началом "критерия_поиска", определяющего , что именно должна искать программа find, считается первый аргумент, начинающийся на "-", "(", ")"', "," или "!". Все аргументы, предшествующие " критерию_поиска", трактуются как имена каталогов, в которых надо производить поиск. Если не указано ни одного пути, поиск производится только в текущем каталоге и его подкаталогах.

Чаще всего поиск проводится по именам файлов, как это показано в предыдущем примере, т. е. "критерий_поиска" задается как "-name имя_файла". Вместо опции -name можно использовать опцию -path, тогда команда будет искать совпадения в полном имени файла, с указанием пути. Например, команда

```
[user]$ find . -path './sr*sc'
```

найдет в текущем каталоге подкаталог './src/misc'. Вместо полного имени файла или каталога в этом примере использован так называемый "шаблон имени". Поскольку шаблоны имен файлов могут использоваться не только с командой find, но и со многими другими командами (включая уже рассмотренные команды chmod, chown, chgrp, cp, rm, cat, mv), то правилам составления шаблонов стоит уделить некоторое внимание.

Чаще всего шаблоны имен файлов строятся с помощью специальных символов "*" и "?". Значок "*" используется для замены произвольной строки символов. В Linux

- "*" - соответствует всем файлам, за исключением скрытых;
- ".*" - соответствует всем скрытым файлам (но также текущему каталогу "." и каталогу уровнем выше "..": не забывайте об этом!);
- "*.*" - соответствует только тем файлам и каталогам, которые имеют "." в середине имени, или оканчиваются на точку;
- "p*r" - соответствует и "peter" и "piper";
- "*c*" - соответствует и "picked" и "peck".

Значок ? заменяет один произвольный символ , поэтому index?.htm будет соответствовать именам index0.htm, index5.htm и indexa.htm.

Кроме "*" и "?" в Linux при задании шаблонов имен можно использовать квадратные скобки [], в которых дается либо список возможных символов, либо интервал, в который должны попадать возможные символы. Например, [abc]* соответствует всем именам файлов, начинающимся с a, b, c; *[I-N1-3] соответствует файлам, имена которых оканчиваются на I, J, K, L, M, N, 1, 2, 3.

А теперь вернемся к команде find и расскажем подробнее о том, какие критерии поиска возможны. Несколько примеров простых критериев поиска перечислены в табл. 4.4.

Таблица 4.4. Критерии поиска для команды find

Опция	Значение
-name шаблон	Ищет файлы, имена которых соответствуют шаблону
-group имя	Ищет файлы, принадлежащие указанной группе
-size число[c]	Ищет файлы, размером в число 512-байтных блоков. Если после числа стоит символ c, значит размер указан в байтах (символах)
-mtime число	Ищет файлы, которые в последний раз изменялись указанное число дней назад
-newer образец	Ищет файлы, которые изменялись после изменения файла, указанного в образце
-type тип_файла	Ищет файлы указанного типа. Тип задается одним из символов b (блок-ориентированные устройства), c (байт-ориентированные устройства), d (файл каталога), f (обычный файл), p (именованный канал) либо l (символическая ссылка)

Другие простые критерии вы можете узнать, если просмотрите man-страницу о команде find. Здесь же надо только сказать, что из простых критериев можно строить более сложные с помощью логических операций and, or или операции отрицания, знаком которой служит восклицательный знак. Например , если вы хотите найти все файлы, имена которых оканчиваются на .txt и .doc, то критерий можно записать как (-name *.txt -or -name * .doc). Можно комбинировать таким образом любое число критериев (и не только простых!). Если операция не указана явно, то подразумевается -and, т. е. вместо (-name *.txt -and -name *.doc) можно записать просто (-name * .txt -name *.doc). Если применяется только одна операция -and или !, то скобки обычно можно опустить, а с операцией -or и в сложных выражениях скобки необходимы. Перед скобкой нужно поставить обратную косую черту, а после скобки - пробел. Например, если вы хотите найти каталог по его имени, то можно сделать это командой

```
[user]$ find /usr -name doc -type d
```

или (с соблюдением правил построения сложных критериев) [user]$ find /usr \(-name doc -and -type d \)

В следующем примере мы ищем файлы по такому критерию: либо имя файла оканчивается на .tmp, либо размер файла больше 100 Кбайт.

```
[user]$ find /home/kos \( \( -name *.tmp \) -or \( -size +200 \) \)
```

В последнем примере стоит обратить внимание на то, что перед значением размера стоит знак "+". Такой знак можно использовать с любым числовым параметром в критериях

поиска команды find. Он означает, что нужно искать файлы, у которых значение параметра больше заданного. Соответственно, знак "-" означает, что надо искать файлы, у которых значение параметра меньше заданного. Если знаки + или - отсутствует, ищутся файлы, у которых значение параметра равно заданному.

Чтобы закончить рассмотрение команды find, надо сказать еще о том, что после критерия поиска в этой команде можно сразу же задать операцию, которая будет применяться ко всем файлам, найденным по указанному критерию. Простейшим примером использования такой возможности является указание команды -print.

```
[user]$ find /home/kos -name *.tmp -print
```

По которой выдается на экран список имен всех найденных файлов с указанием полного пути к файлу. Эта операция применяется по умолчанию, т. е. когда никаких операций вообще не указано (как это было во всех приведенных выше примерах). Другим примером операции, применяемой ко всем найденным файлам, может служить операция -exec cmd {} \;, где cmd - произвольная команда оболочки shell. То есть в этом случае ко всем найденным файлам (их именами заменяются поочередно фигурные скобки) применяется команда cmd. За cmd {} в этом случае должна следовать точка с запятой, экранированная обратной косой чертой.

Например, если вы хотите удалить в текущем каталоге все файлы, к которым пользователи не обращались в течение 30 дней, дайте команду:

```
[root]# find . -type f -atime +30 -exec rm {} \;
```

Вместо -exec можно поставить -ok, тогда перед выполнением указанной команды cmd применительно к каждому файлу будет запрашиваться подтверждение.

В общем, команда find является очень мощным, полезным и чрезвычайно адаптируемым инструментом поиска в файловой системе . Все ее возможности здесь не перечислены, изучайте соответствующую man-страницу. И будьте очень осторожны с применением таких возможностей команды, как вызов других команд, применяемых ко всем найденным файлам. Помните, что изменения часто необратимы!

Редактирование текстовых файлов

Мелкие правки конфигурационных файлов — обычное дело для администратора, поэтому средство их внесения присутствовало в UNIX-системах всегда. Наиболее распространенное такое средство, присутствующее в л.бой системе Linux – это консольный полноэкранный редактор **vi.** Как полноэкранный редактор, vi может находиться в одном из двух режимов. В режиме вставки вводимые символы поступают в редактируемый файл, в командном режиме они воспринимаются как команды. Перечислим коротко самые употребительные команды редактора **vi:**

РЕЖИМ ВСТАВКИ. Включение режима вставки:

- i в текущей позиции курсора;
- I перед первым непробельным символом в текущей строке;
- w в новой строке, добавленной после текущей;
- W в новой строке, добавленной перед текущей.

Выключение режима вставки:

- <Esc>

Команды режима вставки:

- Ctrl+a повторить предыдущую вставку;
- Ctrl+y вставить символ, находящийся над курсором (в предыдущей строке);
- Ctrl+e вставить символ, находящийся под курсором (в следующей строке).

КОМАНДНЫЙ РЕЖИМ.

Удалеие (здесь и далее N — это число):

- N x N символов под курсором и справа от него;
- N X N символов слева от курсора;
- N dd N строк;

• D до конца текущей строки;
• N D до конца текущей строки и еще N-1 строку.

Копиование и вставка строк:

• N yy взять в буфер N строк от текущей и ниже;
• p вставить содержимое буфера после текущей строки;
• P вставить содержимое буфера перед текущей строкой.

Поиск и переход:

• N G перейти к строке с номером N;
• $ G перейти к последней строке файла;
• /< образец > искать образец вниз от курсора;
• ?< образец > искать образец вверх от курсора;
• n повторить поиск в том же направлении;
• N (буквально N): повторить поиск в обратном направлении.

Сохранение и выход:

• :w сохранить текущий файл;
• :w <имя> сохранить под новым именем, если файл <имя> еще не существует;
• :w! <имя> сохранить под новым именем, переписав существующий файл;
• :q выйти;
• :q! принудительно выйти без сохранения;
• :wq сохранить и выйти.

Разное полезное:

• N u отменить последние N изменений;
• N Ctrl+r вернуть последние N отмененных изменений;
• U отменить изменения в последней строке;
• N r < символ > заменить N следующих символов на < символ >;
• N > > добавить отступ (Tab) в N следующих строк;
• N < < удалить один отступ (Tab) из N следующих строк;
• :sh временно выйти в оболочку (вернуться — exit);
• :!<команда> выполнить команду оболочки.

Работа с vi в простых случаях сводится к использованию следующего небольшого набора команд:

vi <имя файла> # открыли файл для просмотра или редактирования или создания
i перешли в режим ввода ход в режим ввода текста (если требуется)
ESC вышли из режима редактирования в режим команд
перешли из режима команд в режим командной строки
w записали изменения (если требуется)
q –> Enter вышли из редактора (если изменения уже записаны или их не было)
q! - > Enter вышли из редактора без сохранения изменений (если требуется).

Порядок выполнения работы:

1. Запустить виртуальную машину с Linux Ubuntu.
2. Загрузиться пользователем root. Для его подключения достаточно войти под первым зарегистрированным пользователем, и при помощи терминала поставить пользователю root новый пароль. Процесс изменения пароля смотри в лабороторной работе №5.
3. Ознакомиться со структурой системных каталогов ОС Linux на рабочем месте. Привести в отчете перечень каталогов с указанием их назначения.
4. Просмотреть содержимое каталога файлов физических устройств. В отчете привести перечень файлов физических устройств на рабочем месте с указанием назначения файлов.
5. Перейти в директорий пользователя root. Просмотреть содержимое каталога. Просмотреть содержимое файла vmlinuz. Просмотреть и пояснить права доступа к файлу vmlinuz.
6. Создать в директории пользователя user три файла 1.txt, 2.txt и 3.txt, используя команды

touch, cat и редактор vi. Просмотреть и пояснить права доступа к файлам.
7. Перейти в директории пользователя root. В отчете описать результат.
8. Изменить права доступа на файл 1.txt в директории пользователя user.
9. Создать жесткую и символическую ссылки на файл 2.txt. Просмотреть результаты.
10. Создать каталог new в каталоге пользователя user.
11. Скопировать файл 1.txt в каталог new.
12. Переместить файл 2.txt в каталог new.
13. Изменить владельца файла 3.txt и каталога new.
14. Удалить файл 1.txt в каталоге new.
15. Удалить каталог new.
16. Найти, используя команду find, файл vga2iso (или другой файл по заданию преподавателя).

Контрольные вопросы
1. Что такое файловая система?
2. Жесткая ссылка в Linux. Основные сведения.
3. Команда поиска в Linux. Основные сведения.
4. Перечислите основные команды работы с каталогами.

Практическая работа №3.
Процессы в операционной системе Linux

Цель работы**:** Ознакомиться на практике с понятием процесса в операционной системе. Приобрести опыт и навыки управления процессами в операционной системе Linux.

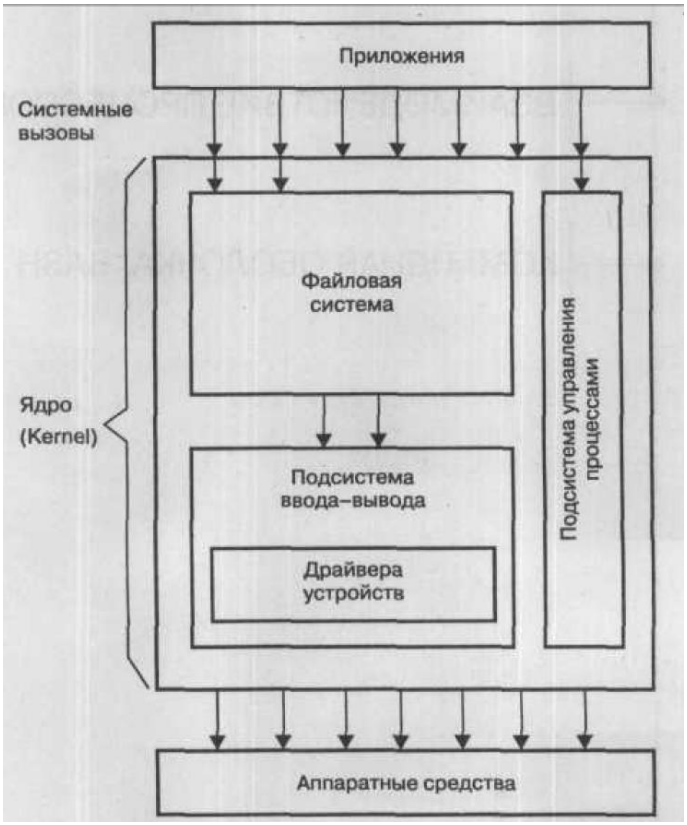
План проведения занятия

1. Используя теоретические сведения изучить порядок работы с текстовым редактором Vi и создать два сценария для исследования процессов.
2. Следуя указаниям ознакомиться на практике с командами и сигналами для управления процессами: запуском, остановкой, переводом на передний план, удалением процесса и др.

3. Составить отчет.

Краткие теоретические сведения

1. Устройство Linux: ядро и процессы



Главная, постоянно находящаяся в оперативной памяти, часть ОС Linux называется ядром (Kernel). Ядро ОС обрабатывает прерывания от устройств, выполняет запросы системных процессов и пользовательских приложений, распределяет виртуальную память, создает и уничтожает процессы, обеспечивает многозадачность посредством переключения между ними, содержит драйверы устройств, обслуживает файловую систему (см. рис. 34).

Пользовательские процессы не могут непосредственно, например, порождать другие процессы, производить чтение или запись на диск, выводить данные на экран или создавать гнездо *(socket)* для обмена по сети. Для выполнения этих действий они должны воспользоваться сервисами ядра. Обращения за такими услугами называются системными вызовами.

Рис. 34. Устройство Linux

Начальная загрузка системы состоит в том, что файл с образом ядра счи-тывается в оперативную память, начиная с нулевого адреса. Этот файл находится в каталоге /boot и называется vmlinuz-x.y.z, где x.y.z — это номер версии ядра. На текущий момент большинство дистрибутивов основано на ядре версии 2.4, хотя уже вышло ядро 2.6 и кое-где еще встречается версия 2.2.

В UNIX-подобных системах в отличие от других ОС ядро минимизировано и не выполняет ни одной функции, служащей непосредственно пользователю. Для этой цели применяются многочисленные утилиты, выступающие в качестве посредников между пользователем и ядром. Только в комплекте с ними ядро образует полноценную операционную систему.

Ядро обслуживает запросы процессов. Что же такое процесс? Это понятие является базовым в UNIX-подобных системах. Процесс можно представить себе как виртуальную машину, отданную в распоряжение одной задачи. Каждый процесс считает, что он на машине один и может распоряжаться всеми ее ресурсами . На самом же деле процессы надежно изолированы друг от друга, так что крушение одного не может повредить всей системе

Каждый процесс выполняется в собственной виртуальной памяти, в которую никакой другой процесс вмешаться не может. Этим и обеспечивается устойчивость всей системы.

Напомним, что такое виртуальная память. Каждому процессу разрешено считать, что его адреса начинаются с нулевого адреса и далее наращиваются. Таким образом, в 32-разрядной ОС процесс может адресовать 4 гигабайта оперативной памяти. Механизм

виртуальной памяти позволяет процессу считать, что именно столько ему и выделено, хотя физически объем ОЗУ вашей машины может быть значительно меньше. Недостающую память заменяет жесткий диск путем записи временно не используемых страниц памяти в раздел подкачки (свопинга).

Разделяемость библиотек между процессами обеспечивается тем, что их код и статические данные отображаются на один и тот же участок физической оперативной памяти.

Таблица процессов

С точки зрения ядра процесс представляет собой запись в таблице процессов. Эта запись содержит сведения о состоянии процесса и данные, существующие в течение всего времени его жизни. Размер таблицы процессов позволяет запускать несколько сотен процессов одновременно. Другая важная информация о процессе — например, таблица всех открытых процессом файлов — хранится в его адресном пространстве. Запись в таблице процессов и пространство процесса вместе составляют контекст, или окружение, процесса. В него входят:

- **PID** — идентификатор процесса. Он принудительно назначается планировщиком при запуске процесса.
- **PPID** — идентификатор родительского процесса.
- **TTY** — имя управляющего терминала - терминала, с которого запущен процесс.
- **WD** — текущий каталог процесса, от которого отсчитываются относительные пути.
- **RID, RGID** — реальные ID и групповой ID пользователя, запустившего процесс.
- **EUID, EGID** — эффективные ID и GID.
- **NICE** — показатель уступчивости. Процессы выполняются в режиме разделения времени, то есть время центрального процессора делится между готовыми к выполнению процессами с учетом их приоритета. Чем выше показатель уступчивости, тем ниже приоритет.
- Переменные окружения.

Системные вызовы fork() и exec() или как размножаются процессы

Каждый процесс порождается другим процессом, использующим для этого системный вызов *fork()*. Таким образом, структура процессов, подобно файловой системе, древовидна. Корнем этого дерева служит *init* — процесс инициализации системы (см. рис. 35). Он запускается ядром первым, получает идентификатор 1 и порождает еще несколько процессов (сколько и каких, можно узнать из его конфигурационного файла /etc/inittab), которые, в свою очередь, при участии пользователя порождают другие процессы.

В результате системного вызова *fork()* родительский процесс полностью копирует свое окружение, включая адресное пространство, в дочерний, так что в моментрождения дочерний процесс отличается только своим ID. Потом дочерний процесс с помощью вызова *exec()* загружает в свое адресное пространство какой-нибудь исполняемый файл и начинает исполнять содержащуюся в нем программу.

Каждый процесс , завершившись, возвращает родительскому процессу какое-то значение, называемое кодом завершения или кодом возврата. По соглашению разработчиков , нулевой код возврата означает успешное завершение, а ненулевые — разнообразные ошибки. Процесс-родитель может приостановить свое выполнение до завершения потомка и выполнить разные действия в зависимости от возвращенного дочерним процессом значения, а может и не делать этого.

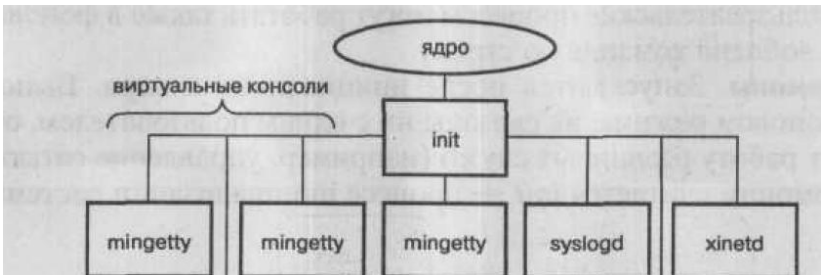


Рис. 35. Иерархия процессов.

Категории процессов

Процессы делятся на три категории:

♦ **Системные**. Они порождаются ядром особым образом в процессе загрузки и выполняют системные функции (например, планирование процессов или смену страниц виртуальной памяти). Выполняемая ими программа берется не из исполняемого файла, а является частью ядра.

и **Пользовательские**. Как правило, они порождаются во время сеанса работы пользователя связаны с терминалом. Если пользовательский процесс работает в интерактивном режиме, то он захватывает терминал в монопольное владение и, пока он не завершится, пользователь не имеет доступа к командной строке на этом терминале. Пользовательские процессы могут работать также в фоновом режиме, освободив командную строку.

• **Демоны (**daemon, сокращение от Disk And Execution MONitor**).** Запускаются после инициализации ядра. Выполняются в фоновом режиме, не связаны ни с одним пользователем, обеспечивают работу различных служб (например, управление сетью). Главным демоном считается *init* — процесс инициализации системы.

Сценарии в Linux. Активные и фоновые процессы

Исполняемые файлы в Linux бывают двух видов. Первый – это файлы в собственно исполняемом (executable) формате. Как правило, такие файлы – результат компиляции программ, написанных на одном из языков программирования. В Linux используется несколько форматов исполняемых файлов, состоящих из машинных кодов и служебной информации, необходимой операционной системе для запуска программы: согласно этой информации, ядро Linux выделяет для запускаемой программы оперативную память, загружает программу из файла и передает ей управление. Большинство утилит Linux – программы именно такого, "двоичного" формата.

Второй вид исполняемых файлов – сценарии. **Сценарий** – это текстовый файл, предназначенный для обработки какой-нибудь утилитой. Чаще всего такая утилита – это интерпретатор некоторого языка программирования, а содержимое такого файла – программа на этом языке.

Запустить сценарий на исполнение можно командой sh имя_сценария

Для того чтобы запустить *процесс* сценария параллельно, достаточно добавить в конец командной строки символ "&":

sh имя_сценария&

Процесс, запускаемый параллельно, называется фоновым (background). Фоновые процессы не имеют возможности вводить данные с того же терминала, что и породивший их shell (только из файла), зато выводить данные на этот терминал могут (правда, когда на одном и том же терминале вперемежку появляются сообщения от нескольких фоновых процессов, начинается неразбериха). При каждом терминале в каждый момент времени может быть не больше одного активного (foreground) процесса, которому разрешено вводить данные с этого терминала. На время, пока команда работает в активном режиме, породивший ее командный интерпретатор "уходит в фон", и там, в фоне, выполняет свой wait().

Активный процесс, foreground process - процесс, имеющий возможность вводить данные с терминала . В каждый момент у каждого терминала может быть не более одного активного процесса.

Фоновый процесс, background process - процесс, не имеющий возможности вводить данные с терминала. Пользователь может запустить любое, но не превосходящее заранее заданного в системе, число фоновых процессов.

Сигналы в Linux

Механизм сигналов — это средство, позволяющее сообщать процессам о некоторых событиях в системе, а процессу-получателю — должным образом на эти сообщения реагировать. Послать сигнал может сам процесс (например , при попытке деления на ноль), ядро (при сбое оборудования), пользователь или другой процесс (требуя прервать выполнение задачи).

Всего в Linux 63 сигнала, обозначаемых своими номерами или символическими именами. Имена всех сигналов начинаются с SIG, и эту приставку часто опускают: так, сигнал, требующий прекратить выполнение процесса, называется SIGKILL, или KILL, или сигнал 9.

Получив сигнал , процесс может: игнорировать его; вызвать для обработки установленную по умолчанию функцию; вызвать собственный обработчик (перехватить сигнал). Некоторые сигналы (например, KILL) перехватить или игнорировать невозможно.

Пользователь может послать сигнал процессу с идентификатором PID командой $ kill [-s <сигнал>] <PID>

где <сигнал> — это номер или символическое имя.

Несколько часто встречающихся сигналов перечислены в таблице 1. Полный список можно получить по команде kill -l *(list)*.

Таблица 1. Сигналы Linux

№	Имя	Назначение	Реакция процесса-получателя
1	HUP	Hangup — отбой	Демоны перечитывают свои конфигурационные файлы
2	INT	Interrupt	Прекратить выполнение (перехватывается)
3	QUIT	Сильнее, чем INT	тоже
4	ILL	Illegal instruction. Программная ошибка	Обработать ошибку. По умолчанию — прекратить выполнение
8	FPE	Floating point exception, вычислительная ошибка (деление на ноль)	Обработать ошибку. По умолчанию — прекратить выполнение
9	KILL	Убить процесс	Немедленно прекратить выполнение. Не перехватывается
11	SEGV	Segmentation violation. Попытка доступа к чужой области памяти	Обработать ошибку. По умолчанию — прекратить выполнение

13	PIPE	Нет процесса, читающего из конвейера	Обработать ошибку
15	TERM	Termination. Завершить процесс	Корректно завершить выполнение. Перехватывается
17	CHLD	Завершился дочерний процесс	Принять возвращенное им значение
18	CONT	Продолжить работу	Продолжить работу приостановленного процесса
19	STOP	Приостановить процесс	Приостановить выполнение

Сообщение-сигнал не содержит никакой информации, кроме номера сигнала (для удобства вместо номера можно использовать предопределенное системой имя). Для того чтобы передать сигнал, процессу достаточно задействовать системный вызов kill(), а для того чтобы принять сигнал, не нужно ничего. Если процессу необходимо как-то по-особенному реагировать на сигнал , он может зарегистрировать обработчик, а если обработчика нет, за него отреагирует система. Как правило, это приводит к немедленному завершению процесса, получившего сигнал. Обработчик сигнала запускается асинхронно, немедленно после получения сигнала, что бы процесс в это время ни делал.

Два сигнала – 9 (KILL) и 19 (STOP) – всегда обрабатывает система. Первый из них нужен для того, чтобы убить процесс наверняка (отсюда и название).

Сигнал STOP приостанавливает процесс: в таком состоянии процесс не удаляется из таблицы процессов, но и не выполняется до тех пор, пока не получит сигнал 18 (CONT) – после чего продолжит работу.

В Linux сигналы можно передать активному процессу с помощью управляющих символов:

Interrupt - ^C
(Ctrl+C) Stop - ^Z"
Terminate - ^D.

3. Команды для управления процессами в Linux

Моментальный снимок протекающих в системе процессов – команда ps

Моментальный снимок протекающих в системе процессов можно посмотреть с помощью команды ps *(process status)*. Без аргументов она покажет список процессов, связанных с текущей консолью (или виртуальным терминалом). Список возможных ключей команды можно, как обычно, получить по команде ps --help.

Вот некоторые полезные из них:

- ♦ -p < список_PID> : только процессы с указанными ID;
- ♦ -u < список_USERID > : только процессы, запущенные указанными пользователями;
- ♦ -e : все процессы в системе;
- ♦ -f : полная форма вывода;
- ♦ -H : вывод иерархии процессов в форме дерева.

Динамика процессов — команда top

Представление о динамике процессов дает команда top. Она выводит список процессов, отсортированный по количеству занятой памяти или использованного процессорного времени, и обновляет его через указанные промежутки времени (по умолчанию через каждые 3 секунды).

Последний процесс, запущенный из оболочки в фоне, можно из **этой** оболочки сделать *активным* при помощи команды **fg** ("**f**ore**g**round" – "передний план").

Команда **bg** (**b**ack **g**round), запускает *в фоне* последний остановленный процесс.

Командой kill, как уже говорилось, можно передать процессу *сигнал*. Команда имеет два параметра - номер сигнала и идентификатор процесса, которому передается сигнал:

kill –номер_сигнала PID

Порядок выполнения работы:

1) Загрузиться не root, а пользователем.
2) Найти файл с образом ядра. Выяснить по имени файла номер версии Linux.
3) Посмотреть процессы ps –f. Прокомментировать. Для этого почитать man ps.
4) Написать с помощью редактора vi два сценария loop и loop2. Текст сценариев: Loop:
   while true; do true; done
   Loop2:
   while true; do true; echo 'Hello'; done
5) Запустить loop2 на переднем плане: sh loop2.
6) Остановить, послав сигнал STOP.
7) Посмотреть последовательно несколько раз ps –f. Записать сообщение, объяснить.
8) Убить процесс loop2, послав сигнал kill -9 PID. Записать сообщение. Прокомментировать.
9) Запустить в фоне процесс loop: sh loop&. Не останавливая, посмотреть несколько раз: ps –f. Записать значение, объяснить.
10) Завершить процесс loop командой kill -15 PID. Записать сообщение, прокомментировать.
11) Третий раз запустить в фоне. Не останавливая убить командой kill -9 PID.
10. Запустить еще один экземпляр оболочки: bash.
11. Запустить несколько процессов в фоне. Останавливать их и снова запускать.
12. Записать результаты просмотра командой ps –f.

Практическая работа № 4
Организация ввода-вывода в ОС Linux.

Цель работы: является ознакомиться на практике с организацией ввода-вывода в операционной системе Linux, понятием виртуальной файловой системой, блочными и символьными устройствами, понятием драйвера, блочными, символьными драйверами, драйверами низкого уровня. Приобрести опыт монтирования файловых систем.

2. Общие теоретические сведения

K Desktop Environment (Среда рабочего стола K) KDE предназначена для поддержания тех же функциональных возможностей графического интерфейса, какие предоставляют и другие популярные системы, например MacOS и Windows. Кроме выполнения стандартных функций, KDE обладает рядом специфических характеристик, которые расширяют возможности графической среды. Для Linux разработано несколько диспетчеров окон, таких, как *olwm, fvwm, afterstep* и другие. Однако, их возможности не идут ни в какое сравнение с возможностями KDE.

2.1 Оконная среда KDE

Как и большинство оконных менеджеров, KDE представляет собой интегрированную среду, содержащую базовые средства для решения ряда повседневных задач. Например, с помощью **KDE** можно выполнять ряд операций:

• Размещение на рабочем столе ярлыков гибких дисков для их монтирования, размонтирования и работы с ними.

• Отображение в графическом виде файловой структуры и перемещение по ней.

• Сопоставление приложений с файлами определенных типов. При этом если щелкнуть на выбранном файле, автоматически будет загружаться нужное приложение.

• Создание на рабочем столе ярлыков принтеров. Если мышью перетащить к такому ярлыку файл, он будет распечатан.

В состав KDE входит не только рабочий стол, но и целый набор приложений и утилит для работы с ним. В стандартном дистрибутиве KDE имеется более сотни программ — от игр и системных утилит до целых блоков офисных программ. Кроме того, приложения KDE могут взаимодействовать друг с другом для упрощения выполнения всевозможных операций.

2.2 Компоненты рабочего стола KDE.

Рабочий стол KDE разделен на три основные части - "поверхность" рабочего стола, панель и линейку задач. Основная рабочая область среды KDE называется рабочим столом. Это тот фон, на котором отображаются все другие компоненты. На рабочем столе можно размещать ярлыки программ, документов и устройств, к которым чаще всего приходится обращаться. Это позволяет легко получать доступ к соответствующим объектам для работы с ними. Кроме той области, что отображается на экране, KDE предоставляет дополнительное виртуальное рабочее пространство для выполнения программ. По умолчанию поддерживается четыре виртуальных рабочих стола. Виртуальный рабочий стол — это, по сути, другой экран, на который можно переключиться для того, чтобы запустить приложение или выполнить еще какую-то работу. Программы и окна легко перемещаются между различными виртуальными рабочими столами. Дополнительные возможности, предоставляемые за счет использования виртуальных рабочих столов, могут быть использованы самыми разными программами. При этом нет необходимости сворачивать и разворачивать окна выполняемых приложений. Можно просто отложить выполняемое приложение в таком виде, как есть, а затем вернуться к нему по завершении выполнения.

2.2.1 Панель

Панель располагается в нижней части экрана. На панели размещаются кнопки, позволяющие выполнять основные процедуры KDE, а также ярлыки наиболее часто используемых программ. Одним из особо важных элементов на панели является кнопка Application Starter (Запуск Приложений), которая расположена (по умолчанию) в левой части панели. Это кнопка с литерой "К" над изображением зубчатого колеса. С ее помощью можно открыть меню, в котором представлены все приложения, установленные на данную систему. Кроме того, это же меню может быть использовано для доступа к некоторым другим разделам KDE, таким, как диалоговая справка и *Панель Управления (Control Panel).* На панели размещен переключатель виртуальных рабочих столов *Пейджер*, *Панель Задач (Taskbar)* и *Часы (Clock).* Панель задач отображает открытые на текущем рабочем столе окна. Чтобы получить немедленный доступ к программе, нужно просто щелкнуть в соответствующем месте на панели задач.

Запустить на выполнение программу можно одним из перечисленных ниже способов.

• **Щелкнуть кнопкой на панели.** Некоторые программы представлены по умолчанию на панели в виде ярлыков или кнопок, например эмулятор виртуальных рабочих столов, панель управления, вызов справки и текстовой редактор.

• **Щелкнуть на элементе рабочего стола.** По умолчанию на рабочем столе размещается только два объекта. Это *Корзина* и ярлык рабочего каталога. Пользователи сами размещают на рабочем столе наиболее нужные и часто используемые программы.

• **Выбрать программу из меню запуска приложений.** Достаточно щелкнуть на литере "К" и выбрать тот пункт меню, который соответствует запускаемому приложению.

• **Использовать диспетчер файлов.** В окне диспетчера файлов нужно выбрать соответствующий файл и щелкнуть на нем мышью.

Можно, конечно, запустить программу на выполнение в командной строке окна терминала – задать название программы. Можно также нажатием клавиш <Alt+F2> вызвать окно запуска программ и ввести туда название программы.

Ряд полезных программ облегчает работу пользователя.

В первую очередь, это программа эмуляции терминала konsole, позволяющая открывать окна и получать доступ к стандартной командной строке. На панели имеется соответствующая кнопка с изображением маленького монитора и ракушки.

Справку в диалоговом режиме можно получить , если щелкнуть на кнопке панели с изображением спасательного круга. Справка включает в себя разные темы, как, например, программа-гид для начинающих пользователей и система контекстного поиска для используемых в KDE приложений.

Просмотреть файловую систему или получить доступ к ресурсам World Wide Web можно,

используя окно диспетчера файлов. Для того чтобы диспетчер файлов отобразил в своем окне содержимое рабочего каталога, нужно щелкнуть на папке панели с изображением домика.
Щелканье по кнопке ► удаляет панель с экрана. Эта кнопка остается при этом на экране, так что можно вернуть панель обратно. Это свойство действует только на открытый в данный момент рабочий стол; другие рабочие столы сохраняют вид мини - или главной панели.
Список задач - кнопка, расположенная справа от меню приложений (обозначена пиктограммой монитора), несет меню, содержащее все активные на данный момент окна, отсортированные по имени. Это позволяет легко и быстро найти необходимое окно и уменьшает захламленность экрана при работе с несколькими окнами.
2.3.2 Настройка KDE
Центр управления (Control Center) (кнопка с изображением гаечного ключа) составляет основу всей системы настроек KDE. В ее входит множество панелей для всевозможных компонентов рабочей среды и даже некоторых приложений KDE.
В центре управления используется деление на группы, щелкнув на знаке "плюс" в углу группы, можно увидеть список входящих в группу компонентов. Щелкнув на знаке "минус" в том же углу группы, этот список можно свернуть. Доступ к любому диалогу с раскрытым деревом меню можно получить при помощи Preferences (Предпочтения) и из меню запуска программ Start Application.
Большинство диалоговых окон имеют кнопку вызова справки. В самом простом случае это контекстная справка. Для ее получения нужно щелкнуть мышью на знаке вопроса на рамке окна. Курсор мыши при этом изменит свой вид на стрелку с большим знаком вопроса. Если теперь щелкнуть на том элементе диалогового окна, с которым возникли трудности, появится прямоугольник желтого цвета с текстом справки. Для получения более детальной справки можно воспользоваться опцией Help (Справка) на левой панели. Наконец, если возникли проблемы с поиском необходимого диалогового окна, на этой же панели нужно выбрать опцию Search (Поиск). Затем нужно ввести ключевое слово, по которому и будет осуществляться поиск.
Control Center
KDE предоставляет широкие возможности по модифицированию внешнего вида окон и рабочей области, включая отображение фона, ярлыков, шрифтов и тому подобное. Не представляет труда и управление работой отдельных компонентов, вроде того же рабочего стола или окна. Например, можно управлять реакцией элемента на щелчок мышью, процессом загрузки и отображения выбранных окон, выбирать хранитель экрана. Все эти и многие другие возможности может предоставить рассматриваемая группа Control Center.
Для настройки параметров работы рабочего стола и окон следует выбрать опцию нужного диалога настройки под названием Desktop на дереве центра управления.
Изменение схемы цветов
Диалоговое окно выбора цвета Appearance&Themes ⇒ Colors (Цвета) предназначено для изменения используемой цветовой схемы для окон KDE и других графических приложений.
Цветовая схема включает в себя 18 пунктов выбора цвета для различных элементов окна программы и установки контрастов. В области предварительного просмотра отображаются все элементы окна, реагирующие на изменение цветовой схемы. Как только пользователь меняет параметры или установки, в области просмотра отражаются внесенные изменения. Можно выбрать уже готовую цветовую схему из списка Color Scheme (Схема Цветов).
Для изменения какой-то конкретной установки нужно выбрать соответствующий элемент из выпадающего меню области цветов Widget Color (Декорация) или щелкнуть в нужной части окна предварительного просмотра. После того как элемент выбран, можно изменить его цвет. Для этого достаточно щелкнуть на кнопке и выбрать понравившийся цвет из появившегося диалогового окна выбора цвета.

Контраст изменяется при помощи позиционирования специального рычажка контраста, который может размещаться в диапазоне от Low (Низкий) до High (Высокий). Эти установки применяются при отображении трехмерных рамок вокруг элементов интерфейса приложений KDE.
Для подтверждения выбора следует щелкнуть на кнопке Apply (Применить). Если приходится часто менять цветовые установки, бывает полезно внести изменения в список цветовых схем. Для этого нужно щелкнуть на кнопке Save Scheme и задать название для своей схемы. Для удаления схемы из списка нужно выделить ее и щелкнуть на кнопке Remove (Удалить).
Изменение фона
Для изменения цвета фона или фонового узора рабочего стола нужно на дереве опций центра управления последовательно выбрать Control Center=>Appearance&Themes=>Background. В результате появится диалоговое окно, имеющее три основные области:
• список виртуальных рабочих столов;
• окно предварительного просмотра;
• окно настройки параметров.
Каждый виртуальный стол в KDE имеет собственные настройки фона. Для каждого такого стола можно выбрать фон с одноцветной или двухцветной палитрой, а также фоновый узор . Если используется фоновый узор, можно задать способ его отображения. Можно также выбрать несколько узоров и автоматически переключаться между ними. Доступны и более усовершенствованные опции, позволяющие сочетать цвета и узоры, а также поддерживать динамические настройки фона.
В процессе внесения изменений в установки они отображаются в окне предварительного просмотра.
Виртуальные рабочие столы
Производить настройку параметров виртуальных рабочих столов в KDE можно в диалоговом окне Control Center ⇒Desktop ⇒Multiple Desktops.
Указатель Number of Desktops (Количество рабочих столов) показывает, сколько виртуальных рабочих столов доступно. Их число может изменяться в диапазоне от одного до шестнадцати. Здесь же можно задать название для рабочего стола, которое потом будет отображено в списке окон (Window List) или использовано в настройках панели.
Хранитель экрана
Диалоговое окно выбора хранителя экрана Appearance&Themes ⇒ Screensaver позволяет выбрать хранитель экрана и осуществить настройку его параметров. Опции настройки бывают глобальные, как, например, опция установки времени запуска хранителя экрана, и индивидуальные — для каждого отдельного хранителя. Диалоговое окно выбора хранителя экрана имеет три основные секции:
- окно предварительного просмотра;
- список программ — хранителей экрана;
- опции настройки.

Необходимо выбрать название нужной программы из предложенного списка. Для настройки параметров необходимо щелкнуть на кнопке Setup (Настройка) и в появившемся диалоговом окне произвести установку нужных характеристик.
Для установки интервала времени, через который будет запускаться хранитель экрана, нужно ввести в поле опции Settings (Установки) величину данного интервала в минутах.
Параметр Priority (Приоритет) позволяет определить распределение процессором времени на работу хранителя экрана . Это пример того, как в Linux организована многозадачность. Если нужно, чтобы у хранителя был наивысший приоритет (например, для качественного вывода анимации), следует передвинуть рычажок в позицию High (Высокий). Если же, наоборот, необходимо обеспечить высокий приоритет других процессов, нужная позиция для рычажка приоритетности хранителя — Low (Низкий).

Для того чтобы проверить выполненные установки, следует щелкнуть на кнопке Test (Просмотр). Для подтверждения сделанного выбора нажмите кнопку OK или Apply.

Настройка диспетчера окон

С помощью опций Control Center Appearance&Themes ⇒ Desktop/Window behavior (Поведение Окон) центра управления можно устанавливать поведение диспетчера окон. Эти настройки определяют способ отображения окон в случае их перемещения и изменения размера, а также управляют процессом разворачивания, размещения и выделения окон при работе с диспетчером окон. Опции в верхней части диалогового окна позволяют выполнить настройку параметров, задающих режим перемещения окна и изменения его размеров, а также определяют функциональность команды Maximize (Развернуть). Можно задать такой режим отображения окна при перемещении или изменении его размера, что окно будет отображаться вместе со всем своим содержимым или же в виде прозрачной рамки. Если выбран режим отображения всего содержимого окна, процесс перемещения или изменения размеров окна будет требовать дополнительного времени для обновления отображаемых на экране элементов.

Если используются окна с изменяемыми размерами, можно выбрать режим обновления содержимого окна при каждом изменении его размера. Для этого следует воспользоваться установками Resize (Изменение размера). Для выбора частоты обновления можно воспользоваться специальным рычажком. Если сделан выбор, отличный от None (Никакой), то каждый раз, при изменении размеров окна, его содержимое будет обновляться. Это дает возможность отслеживать процесс заполнения окна программой и позволяет выбрать оптимальные размеры последнего.

Window behavior/Moving – меню установок размещения окна на экране Placement (Расположение) позволяет определить место на экране, где будет отображаться окно. Поддерживаются такие методы.

- **Smart (Умный)** — минимизируется перекрытие между окнами.
- **Cascade (Каскад)** — первое окно отображается в левом верхнем углу. Следующее окно отображается сдвинутым немного вправо и вниз, так что окна практически полностью перекрываются. И так далее. Окна расположены, как карты в руке при игре в преферанс.
- **Random (Произвольный)** — окна располагаются на экране в произвольном порядке.

Метод получения фокуса (т.е . метод выделения отдельных окон или элементов) является, пожалуй, индивидуальным методом настроек KDE. С помощью этого метода определяется, какое из открытых окон активно и какие следует выполнить действия при активизации окна. Более детально это выглядит так.

- **Click to focus (Передача фокуса щелчком).** Окно получает фокус (т.е. становится активным) при щелчке на нем мышью. При этом окно автоматически выводится на первый план по отношению к другим окнам. Такой метод используется по умолчанию.
- **Focus follows mouse (Фокус за мышью).** Окно получает фокус при непосредственном обращении к нему (это можно сделать с помощью указателя мыши, используя комбинацию клавиш <Alt+Tab> и тому подобное). При этом окно может подниматься поверх других окон, а может и не подниматься. Перемещение указателя мыши на рабочую область за пределы окна не означает потерю последним фокуса. При выборе опции Auto Raise (Всплывать автоматически) окно будет всплывать на экране при перемещении в его область курсора в течение нескольких миллисекунд. Число этих миллисекунд устанавливается с помощью рычажка Delay (Задержка). Если выбрана опция Click Raise (Всплывать при щелчке), окно будет подниматься поверх других окон при щелчке в любой части окна. В противном случае такая реакция окна будет наблюдаться только при щелчке на его заголовке. Это исключительно полезный метод передачи фокуса, поскольку позволяет набирать текст в одном окне и одновременно читать содержимое другого окна, расположенного частично поверх указанного.
- **Focus Under Mouse (Фокус под мышью).** Окно получает фокус при любом

перемещении на него указателя мыши. При этом комбинация клавиш <Alt+Tab> может и не помочь.

• **Focus Strictly Under Mouse (Фокус только под мышью).** Окно получает фокус, только если указатель мыши находится внутри окна. Если указатель мыши находится в рабочей области, где нет окон, ни одно из окон не получит фокус.

Использование окон

Открытое окно состоит из следующих элементов.

Window menu (Меню управления окном) - В левом верхнем углу каждого окна находится пиктограмма манипулирования окном. При щелчке на ней появляется меню, содержащее команды с помощью которых можно манипулировать данным окном . Maximize (Максимизировать) увеличит окно до максимально возможного размера . Minimize (Минимизировать) сделает ваше окно невидимым. Move (Переместить) позволяет передвигать окно с помощью мыши. Size (Изменить размер) позволит вам увеличить или уменьшить окно. Shade – свернет окно до заголовка. To desktop ...(На рабочий стол) позволит перевести окно на другой рабочий стол. Выберите рабочий стол, на который вы хотите переместить это окно. Окно при этом исчезнет. Для того чтобы увидеть его снова, выберите имя на Линейке задач, или щелкните на соответствующую кнопку рабочего стола на панели KDE. Close (Закрыть) закроет данное окно. Always on Top – оставляет окно поверх всех открытых окон.

Использование панели меню каждого окна в KDE очень просто. Щелкните на команду, и она будет исполнена. При нажатии на правую кнопку мыши появится контекстное меню, позволяющее вывести на экран панель меню. Можете отсоединить меню от окна и оставить его "плавать" по экрану.

Ниже панели меню находятся пиктограммы инструментов, которые позволяют исполнять различные команды. Можно передвинуть инструментальную панель - влево, вправо, вверх, вниз, и, конечно, она тоже может "плавать".

3. Порядок выполнения работы

1. Запустите Центр управлений.

Поменяйте Фон, сначала на одноцветный, а затем вставьте фоновое изображение.

3. Установите хранитель экрана, на своё усмотрение, и режим ожидания равный минуте.
4. Сделайте так, чтобы окна передвигались вместе со всем их содержимым.
5. Задайте звуковой щелчок, подтверждающий нажатие каждой клавиши.
6. Измените ширину линейки панели.
7. Запустите диспетчер приложений. И запустите программу текстового процессора KWord.
8. В другом рабочем столе откройте программу растрового редактора Paint.
9. Откройте KWord и наберите следующий текст:

The Quick Brown Fox Jumps Over The Lazy Dog, используя два разных стиля по вашему выбору. Сохраните этот файл в домашнем каталоге пользователя, закройте KWord.

10. Откройте ваш домашний каталог пользователя Konqueror'ом, создайте в нем каталог, скопируйте ваш текстовой файл в этот каталог.
11. Ознакомьтесь с содержанием домашнего каталога, скопируйте с дискеты файлы.
12. Получите справку об интересующем вас объекте.
13. Создайте любой рисунок с помощью Paint, чтобы в нем были ВСЕ фигуры (1. эллипс, 2. окружность, 3. линия, 4. прямоугольник, 5. круг) хотя бы по одному разу и присутствовало не менее четырех цветов.
14. Сохраните файл с рисунком в домашнем каталоге, закройте Paint.
15. Скопируйте файл с рисунком в тот же созданный вами каталог.
16. Измените атрибуты доступа к созданным файлам.
17. Покажите преподавателю ваши файлы, затем удалите их.

4. Возможность обработки объектов

Диспетчер файлов Konqueror предназначен не только для просмотра документов. Он может также быть использован для настройки просматриваемых объектов и активной работы с ними. Например, он использует систему типизации файлов KDE для запуска программ и загрузки документов. Распознавание типа файла производится как для локальных файлов, так и для файлов на удаленных узлах. Поэтому, таких проблем, как, скажем, выбор отображаемых ярлыков или контекстных меню для данного элемента, не возникает. При запуске файла на удаленной системе KDE загрузит этот файл и запустит нужное для работы с ним приложение. Можно перетаскивать элементы из одного окна диспетчера файлов в другое, создавая, таким образом, копию элемента или связь с этим элементом. Наконец, при работе с локальной файловой системой, Konqueror может быть использован для изменения атрибутов этой самой файловой системы (как, например, владелец файла и права доступа) с помощью простого графического диалогового окна.

Прежде чем начинать работать с диспетчером файлов, полезно сначала познакомиться с некоторыми элементами интерфейса пользователя. Ниже описаны те задачи, которые можно решать с помощью Konqueror.

4.1. Работа с файлами и каталогами.

Чтобы начать работу с диспетчером файлов, достаточно щелкнуть на папке любого каталога или щелкнуть кнопкой с изображением глобуса на панели. В результате запустится диспетчер файлов, и в главном окне будет выведено содержимое рабочего каталога. Эта область называется областью просмотра. Как правило, в основном окне такая область просмотра единственная. Тем не менее, там также есть область, где отображается дерево каталогов, которая вместе с областью просмотра занимает место в основном окне при его отображении на экране. Наконец, там еще можно увидеть и область эмуляции окна терминала.

4.1.1. Область просмотра

В области просмотра, как правило, отображается содержимое выбранного каталога. Для обозначения элементов в каталоге используются ярлыки, которые определяются типом файлов. Вид окна просмотра можно изменить. Для этого нужно воспользоваться опциями меню View (Вид)⇒View Mode (Способ отображения). Существует пять отображаемых типов (согласно опциям меню View).

- **Icon View (ярлыки).** Содержимое каталога отображается в виде больших ярлыков, помещенных в рамку.
- **Text View (Текст).** Выводится детальный листинг файлов и каталогов со всеми их атрибутами.
- **MultiColumn View (Колонки).** Выводится в виде колонок только название файлов и их мини-ярлыки.
- **Detailed List View (Список).** Выводится та же информация, что и в режиме Text View, только включая еще и мини-ярлыки для идентификации типа файлов.
- **Tree View (Дерево).** Все точно так же, как в предыдущем случае, только теперь каждый ярлык может быть развернут в дерево подкаталогов.

Обычно скрытые файлы (те, имена которых начинаются с точки) в описанных выше списках файлов не отображаются. Для того чтобы включить такие файлы в листинги, нужно выбрать опции меню View(Вид)⇒Show Hidden Files (Показывать скрытые файлы).

Переходить между каталогами с помощью диспетчера файлов можно несколькими способами. Для перехода в подкаталог нужно щелкнуть на названии папки или каталога в области просмотра. Для перехода в каталог высшего уровня можно щелкнуть на кнопке со стрелкой вверх на панели инструментов диспетчера файлов. Для переключения между каталогами, которые уже посещались, можно использовать кнопки со стрелками влево и вправо на той же панели инструментов. Каждая из этих кнопок имеет маленькую стрелочку, направленную вниз. Это значит, что если удерживать кнопку нажатой, можно будет увидеть список адресов, куда можно перейти. Для кнопки со стрелкой вверх такой список будет

состоять из каталогов вышестоящих уровней – первого, второго и так далее. Для кнопок со стрелками влево и вправо это будут каталоги, которые последовательно посещались. Их хронологический порядок записан в кэш-памяти.
Для переключения на элемент, помеченный закладкой, нужно выбрать соответствующий пункт из меню закладок Bookmarks (Закладки).
Наконец, для перехода в нужное место файловой системы можно просто ввести адрес перехода в поле Location (Адрес), размещенном в верхней части окна диспетчера файлов. Это же можно сделать и при помощи выпадающего диалогового окна Open Location (Открыть Адрес). Для этого достаточно выбрать опции меню Location ⇒Open Location. К тому же результату приведет нажатие комбинации клавиш <Ctrl+O>. При вводе адреса его можно указывать как обычный путь к каталогу или как URL. При работе с локальной файловой системой следует использовать префикс file.

4.1.2. Дерево каталогов

Левое подокно окна диспетчера файлов, как правило, используется для отображения дерева каталогов. По умолчанию оно скрыто, но при помощи опций в меню диспетчера файлов Window (Окно) ⇒Show Navigation Panel (Показывать дерево директорий) его можно вывести на экран. Подокно Tree View (Просмотр дерева) имеет три каталога наивысшего уровня, соответствующих тем областям, в которых производится просмотр файловой системы.

- **The Home Directory (Рабочий каталог).** Соответствует рабочему каталогу пользователя.
- **The Network (Сетевая папка).** Эта папка содержит еще три папки: FTP Archives (Архивы FTP) для работы с FTP узлами; Web Sites (Страницы Web), в которой хранятся используемые закладки; Windows Shares (Сетевые ресурсы Windows), используемая для доступа к совместно используемым ресурсам с помощью SMB.
- **The Root Directory (Корневой каталог).** Соответствует корневому каталогу файловой системы.

В подокне просмотра дерева каталогов отображаются только каталоги. Файлы и связи там не показаны. Для того чтобы развернуть или свернуть каталог, нужно щелкнуть на квадрате со знаком плюс или минус соответственно, размещенном слева от названия каталога. Когда каталог свернут, в квадрате отображается плюс, когда же каталог раскрыт,
квадрате появляется минус. Для отображения в окне просмотра содержимого каталога нужно в подокне просмотра дерева каталогов щелкнуть мышью на названии этого каталога 68 (при этом должна быть установлена связь между окнами, подробнее об этом рассказано в разделе "Установка связи между окнами").

4.1.3. Окно эмуляции терминала

В нижней части окна диспетчера файлов можно вывести эмуляцию окна терминала (что-то вроде консольного устройства). Для этого достаточно выбрать опции меню Window(Окно)⇒Show Terminal emulator (Показать окно эмуляции терминала). Это позволит получить доступ к командной строке, где можно обычным способом вводить команды LINUX. Когда с помощью дерева каталогов или в окне просмотра пользователь переходит в другой каталог, изменение текущего каталога будет автоматически отображаться и в этом окне (при установленной связи между окнами). А вот перемещения, выполняемые в окне эмуляции терминала с помощью команды **cd**, не изменят содержимого области просмотра и подокна дерева каталогов.

4.1.4. Установка связи между окнами

Изменения, внесенные в области просмотра диспетчера файлов или подокна дерева каталогов, могут отражаться и в других окнах. По умолчанию все окна связаны друг с другом, так что они будут отражать одни и те же каталоги. Иногда бывает полезно убрать такую связь для отдельного окна, чтобы оно отражало какой-то один каталог.
Для установки или снятия связи между окнами, нужно воспользоваться опцией View ⇒ Link View (Связать). Между всеми выделенными таким способом окнами

устанавливается связь, поэтому они будут отображать одинаковые каталоги. Единственным исключением является эмулятор окна терминала. При использовании команды **cd** эмулятор терминала с другими окнами работать синхронно не будет.

4.1.5. Создание окон

Три окна, используемые в диспетчере файлов, — это только начало. Пользователь может создать несколько копий окна просмотра или эмулятора терминала. Причем каждая копия может работать с разными каталогами или узлами Web.

Для создания нового окна нужно выбрать существующее окно того типа, который нужно создать. Затем следует выбрать одну из перечисленных ниже опций меню.

- Window (Окно)⇒Split View Left/Right (Разделить по вертикали) – текущее окно разбивается по вертикали на два окна того же типа. Это же можно сделать, используя комбинацию клавиш <Ctrl+Shift+L>.

Window (Окно)⇒Split View Top/Bottom (Разделить по горизонтали) — текущее окно разбивается по горизонтали на два окна того же типа. Комбинация клавиш в данном случае <Ctrl+Shift+T>.

Для создаваемых окон по умолчанию связи с другими окнами не устанавливаются. Это удобно для того, чтобы просматривать разные каталоги. Для изменения просматриваемой области нужно сначала просто щелкнуть мышью на окне. Маленький зеленый индикатор указывает активное в данный момент окно. Затем следует ввести новый адрес для просмотра в поле Location в верхней части окна Konqueror (Location ⇒ Duplicate Windows). Для того чтобы убрать существующее окно просмотра или эмулятор терминала, нужно выделить его щелчком мыши, а затем выбрать опции контекстного меню Window⇒Remove Active View (Убрать окно) или нажать комбинацию клавиш <Ctrl+Shift+R>. Для изменения размера нужно при помощи мыши переместить границу между двумя соседними окнами.

4.1.6. Сохранение формата

Для использования созданного пользователем формата для отображения на экране диспетчера файлов, этот формат нужно сохранить. Для этого следует воспользоваться опциями Settings(Установки) => Save View Profile (Сохранить профиль).

Сначала для профиля нужно задать название или воспользоваться уже существующим. Затем следует определить, нужно ли сохранять в профиле URL (опция Save URLs in profile). Если соответствующая опция выбрана, то каждый раз при загрузке профиля будет происходить обращение к URL.

4.2. Задачи управления

В этом разделе описываются те задачи по управлению файловой системой, которые можно решать с помощью диспетчера файлов. Сюда следует включить получение информации о файлах, копирование, перемещение и удаление файлов, изменение таких атрибутов файлов, как название, владельцы и права доступа к ним.

4.2.1. Получение информации о файле

Одна из самых основных процедур заключается в получении информации о файле. Получить ее можно самыми разнообразными способами.

Например, панель состояния отображает данные о размере и типе объекта, который выделен при помощи курсора мыши. Поэтому для получения такой информации достаточно навести указатель мыши на интересующий объект.

Более детальную информацию о файле можно получить, если в меню View/View Mode (Вид) выбрать опции Text View (Текст) или Detailed List (Список). В этом случае о каждом элементе в выведенном на экран списке можно узнать такие подробности, как тип, размер, название, время изменения, права доступа, владелец, группа и наличие связей. Ярлыки в окне просмотра, устанавливающиеся KDE автоматически, соответствуют типу каждого элемента. В KDE для отображения файлов различных типов используется большое число ярлыков. По негласному соглашению для каталогов используется ярлык в виде папки, для документов — ярлыки с изображением листа бумаги, а для программ — ярлык с

изображением зубчатого колеса.

4.2.2. Выбор элемента

Некоторые действия с объектами можно выполнять прямо в окне диспетчера файлов. Существует много способов, с помощью которых можно выделять группы объектов и производить с ними разнообразные процедуры.

Для выбора объекта без его запуска нужно щелкнуть на нем мышью, удерживая при этом нажатой клавишу <Ctrl>. Объект будет при этом затемнен. Это означает, что он выделен. Для того чтобы к выделенному объекту добавить еще один, или убрать объект из группы выделенных объектов, используется та же описанная процедура. Можно выделить группу объектов, захватив их в рамку при помощи курсора мыши.

Большую группу объектов, которую неудобно или просто невозможно выделить с помощью мыши, можно выделить, используя названия и спецификацию шаблона. Для этого нужно в меню Edit (Правка) выбрать опцию Select (Выделить) или нажать комбинацию клавиш цифровой панели <Ctrl+Плюс>. Затем в диалоговом окне Select files (Выделить файлы) следует ввести названия файлов или спецификацию шаблона. Затем нужно щелкнуть на кнопке ОК, после чего будут выделены все файлы, отвечающие заданному шаблону. Аналогично, из группы выделенных файлов можно и убирать файлы. Для этого в меню Edit следует выбрать опцию Unselect (Отменить выделение) или нажать комбинацию клавиш цифровой панели < Ctrl+Минус>. Если воспользоваться опцией Edit⇒UnselectAll (Отменить для всех), будет отменено выделение для всех выделенных до этого объектов. К тому же результату приводит нажатие комбинации <Ctrl+U>. Опция Edit⇒Invert (Наоборот) приведет к выделению невыделенных файлов и отмене выделения для выделенных. То же можно сделать, нажав <Ctrl+*>.

4.2.3. Перемещение и копирование файлов

Наиболее простой способ переместить или скопировать файл из одного места файловой системы в другое или создать связь с файлом заключается в том, чтобы выделить его и просто перетащить мышью в нужное место. Таким способом можно перетаскивать файлы из одного открытого окна диспетчера файлов в другое, между окном диспетчера файлов и рабочим столом.

Иногда бывает трудно определить, какая из операций — копирование (Copy), или перемещение (Move) – является наиболее приемлемой. Это особенно актуально для тех, кто начинает работу с особыми файлами и каталогами KDE, вроде рабочего стола (Desktop). Ниже приведены некоторые соображения по этому поводу.

- *Если действительно необходимо создать копию объекта*, следует выбирать Copy и только Copy. Для программ такая процедура копирования используется редко. Что касается документов и других подобных файлов, то все зависит от конкретных обстоятельств. Здесь следует помнить, что если используется несколько копий одного файла, то внесение изменений в одну из этих копий на других копиях не отражается. Поэтому, хотя и можно размещать документы прямо на рабочем столе, желательно размещать там только связи с документом, или перемещать туда файлы только на время.
- *Если нужно изменить место хранения файла*, следует выбирать Move. Как и в предыдущем случае, эта процедура редко используется для программ и командных файлов. Программы обычно хранятся в специальных каталогах, что отражено в указании используемых в приложениях путей. Перемещение программы в другое место может привести к тому, что она станет недоступной для использования при вызове из командной строки.
- *Для файлов конфигурации рабочего стола* обычно используется команда копирования или создается связь. Иногда смещение файла рабочего стола с привычного места приводит к некорректной его работе. Например, файл рабочего стола MimeType может быть использован, только если он находится в каталоге mimelink. Поэтому если нет уверенности в правильности предпринимаемых действий, перемещать файлы рабочего стола из их специальных каталогов в другие не стоит.

4.2.4. Удаление файлов

Для того чтобы удалить файл из файловой системы, **можно** воспользоваться одним из трех способов: переместить файл в корзину (Trash), непосредственно удалить файл, или вытереть его. Помещение файла в корзину означает его перемещение в каталог Trash, т.е. файл будет продолжать занимать место в системе и его можно будет в будущем, если потребуется, восстановить. Непосредственное удаление файла означает, что файл из системы полностью удаляется с освобождением места. В этом случае восстановление файла невозможно. Вытирание подразумевает предварительную запись в файл набора специальных данных перед удалением. Это делается для того, чтобы быть уверенным, что даже самая совершенная технология восстановления файлов не сможет восстановить его содержимое.

Для перемещения файла в корзину нужно в меню Edit или контекстном меню выбрать опцию Move to Trash (Поместить в Корзину). Можно также просто перетащить нужный объект из окна диспетчера файлов на пиктограмму с изображением корзины на рабочем столе. Аналогично, для удаления файла (или файлов) нужно выделить этот файл (или файлы) и затем выбрать опцию Delete (Удалить) из одного из упоминавшихся выше меню.

4.2.5. Запуск файлов

Запускать файлы из окна диспетчера файлов можно так же, как это делается при использовании рабочего стола. Можно либо просто щелкнуть на выбранном объекте, либо перетащить объект к нужной программе, либо войти в контекстное меню документа и с помощью опции Open With (Открыть с помощью) выбрать программу из списка.

Если щелкнуть один раз мышью на объекте, KDE выберет необходимый способ действий, исходя из типа файла. Если KDE не в состоянии определить программу, используемую по умолчанию для работы с файлом, KDE предложит пользователю самостоятельно выбрать такую программу.

4.2.6. Изменение файлов и каталогов

В KDE, за счет использования простого и понятного графического диалога для работы с объектами, изменение атрибутов объектов файловой системы является очень простой задачей. Для получения доступа к этому диалоговому окну нужно выбрать опцию Properties (Свойства) из контекстного меню объекта. После этого появится диалоговое окно со вкладками, разными для объектов разного типа. Но первые две вкладки одинаковы для объектов всех типов. Это вкладка General (Общие) и Permissions (Доступ).

4.2.6.1. Изменение названия файла

Для того, чтобы изменить название файла, нужно выбрать из контекстного меню этого файла опцию Properties. После того, как появится окно диалога, на вкладке General нужно отредактировать название файла в поле Name (Название), затем нажать кнопку ОК.

4.2.6.2. Замена владельца и изменение прав доступа

Для замены владельца файла и прав доступа к нему в диалоговом окне Properties нужно перейти на закладку Properties/Permissions (Права доступа). Для изменения прав доступа к объекту в секции Access permissions (Права доступа) диалогового окна следует напротив нужных опций поставить флажки. Чтобы изменить владельца файла или его группы, нужно воспользоваться элементами управления в секции Ownership (Принадлежность).

5. Порядок выполнения работы

1. Откройте окно диспетчера файлов Konqueror, щелкнув на кнопке Home на панели.
2. Разверните окно.
3. Выберите опцию Window⇒Show Terminal Emulator.

Выберите опцию Window⇒Split View Left/Right.

5. Переместите панель кнопок Button и панель адреса Location с помощью мыши.
6. Измените размеры панелей, используя специальные метки изменения размера на границах окон.
7. Запустите на выполнение в правой панели команду vi.

8. На левой панели просмотра откройте аплет, нажав клавиши <Ctrl+O>.
9. На нижней панели воспользуйтесь компилятором и другими средствами командной строки.
10. Выберите опцию Setting⇒Save View Profile для сохранения профиля.
Введите название профиля, а затем выберите опцию Save window size in profile. После этого нужно для сохранения установок щелкнуть на кнопке Save