Лабораторная работа №3

Файловая система и командный интерфейс ОС Linux

Теоретические сведения

Цель работы

Ознакомиться с командным интерфейсом, структурой файловой системы Линукс, типами файлов и командами управления файловой системой.

1.1 Организация файловой системы

Файловая система ОС Линукс (как и прочих unix-подобных систем) устроена так, что все ресурсы представлены единообразно, в виде файлов. Такой подход позволяет обеспечить универсальный интерфейс доступа к любым ресурсам: от физических устройств, до процессов, выполняющихся в системе. С точки зрения пользователя файловая система представляет логическую структуру каталогов и файлов. С другой стороны, невидимой пользователю, внутреннее устройство файловой системы реализует физические операции чтения/записи файлов на различные носители, алгоритмы доступа и многое другое.

Типы файлов

Для обеспечения единообразного доступа к файлам их прежде всего необходимо классифицировать. В Линукс это сделано следующим образом:

- обычные (regular) файлы - текстовые, исполняемые, графические и пр. файлы, создаваемые пользователями и прикладными программами;
- каталоги (directories) - именованные группы файлов и вложенных каталогов (т.е. содержимое каталога - суть файлы и другие каталоги);
- файлы устройств (devices) - соответствуют присутствующим в системе реальным (жесткие диски, принтеры, мыши, ЦП и т.д.) устройствам и т.н. псевдоустройствам (например, /dev/null). Файлы устройств представляют символьные (последовательного доступа) и блочные (произвольного доступа) устройства. К первыми относятся, например, параллельные и последовательные порты, ко вторым - жесткие диски;
- специальные файлы - сокеты (sockets) и именованные каналы (named pipes), которые предназначены для обмена информацией между процессами;
- символьные ссылки (symlinks) - именованные указатели на физические файлы (аналог ярлыков ОС Windows), содержащие только путь к некоторому файлу. Символьные ссылки могут указывать на файлы, хранящиеся как локальных, так и в сетевых каталогах.

Символьные ссылки (или "мягкие") не нужно путать с "жесткими", которые указывают на inode файла. Inode (идентификатор узла) - это уникальный числовой идентификатор узла (файла или каталога) файловой системы, по которому и осуществляется доступ к нему. Имя же файла (включая полный путь) ориентировано на пользовательское восприятие. Для оператора проще оперировать с осмысленными именами файлов (например: report.txt, myfoto.jpg и т.п.), чем с числовыми значениями. Прочие отличия "жестких" и "мягких" ссылок вам предстоит выяснить в ходе выполнения этой лабораторной работы.

Каталоги Линукс

Все файлы упорядочены по каталогам. Структура и назначение каждого из каталогов, созданных на этапе установке предопределены, хотя и могут быть (что крайне не рекомендуется) изменены суперпользователем.

Файловая система имеет иерархическую структуру и начинается от корневого каталога (/). Его подкаталогами являются:

- /bin - исполняемые файлы общего назначения;
- /boot - содержит образ загружаемого ядра;
- /dev - файлы устройств;
- /etc - конфигурационные файлы общего пользования;
- /home - домашние каталоги пользователей, включая программы и файлы личных предпочтений;
- /lib - общесистемные библиотеки;
- /mnt - каталог монтирования внешних файловых систем;
- /proc - виртуальная файловая система для чтения информации о процессах;
- /root - домашний каталог суперпользователя;
- /sbin - программы системного администрирования;
- /tmp - каталог для хранения временной информации;
- /usr - каталог пользовательских прикладных программ со всеми их исполнимыми и конфигурационными файлами. Например, в подкаталог /usr/local инсталлируются программы, не входящие в дистрибутив Линукс, или собираемые из исходных текстов.
- /var - каталог для хранения часто изменяющихся файлов. Например, спулера печати, различных лог-файлов, почтовых сообщений и т.п.
- /lost+found - каталог для нарушенных фрагментов файлов, обнаруженных в результате проверки файловой системы после сбоя.

Такая стуктура типична для большинства дистрибутивов Линукс, но могут иметься и дополнительные каталоги.

Именование файлов и каталогов

Файловая система Линукс поддерживает "длинные" имена, содержащие символы латиницы, национальных алфавитов, знаки пунктуации и спецсимволы. Абсолютно запрещенными к использованию в имени являются прямой и обратный слэши (/ и \). Максимальное количество символов в имени - 255. Понятие "расширения файла" в unix-системах отсутсвует как таковое, поэтому в имени может быть несколько частей, разделенных точками. Все имена - регистрозависимые.
Приведенные выше правила справедливы и для каталогов.
Файлы и каталоги, названия которых начинаются с точки (т.н. dot-файлы), являются аналогами "скрытых" файлов MS-DOS. Т.е. в общем случае они не отображаются при просмотре содержимого файловой системы.
Для быстрого доступа к файлам в оболочке имеются несколько переменных окружения, хранящих соответвующие пути. Это, например, переменная $HOME, в которой содержится пути к домашнему каталогу текущего пользователя. Т.е. действия команд

```
[usr1@localhost var]$ cd /home/usr1
```

и

```
[usr1@localhost var]$ cd $HOME
```

приведут к одному результату - переходу в домашний каталог пользователя usr1. Более того, в оболочке определен псевдоним для домашнего каталога - символ ~ (тильда) можно использовать аналогично $HOME. Например:

```
[usr1@localhost var]$ cd ~
[usr1@localhost ~]$ pwd
/home/usr1
```

[usr1@localhost var]$

1.2 Регистрация пользователя в системе

Для входа пользователя с терминала в многопользовательскую операционную систему LINUX необходимо зарегистрироваться в качестве пользователя. Для этого нужно после сообщения Login: ввести системное имя пользователя, например, "student". Если имя задано верно, выводится запрос на ввод пароля:
Password:
Наберите пароль "student" и нажмите клавишу *Enter*.
Если имя или пароль указаны неверно, сообщение *login* повторяется. Значение пароля проверяется в системном файле *password*, где приводятся и другие сведения о пользователях. После правильного ответа появляется приветствие LINUX и приглашение:
student@linux:>
Вы получили доступ к ресурсам ОС LINUX.
Выход из системы:
exit - окончание сеанса пользователя.

После успешного ввода имени пользователя и пароля система выводит приглашение к вводу команды.

- для суперпользователя root
$ - для всех остальных пользователей

Часто при первом входе в систему пользователя требуется поменять пароль, назначенный пользователю администратором - используйте команду passwd.
$ passwd

2. Командный интерфейс

Формат команд в ОС LINUX следующий:
имя команды [аргументы] [параметры] [метасимволы]
Имя команды может содержать любое допустимое имя файла; аргументы - одна или несколько букв со знаком минус (-); параметры - передаваемые значения для обработки; метасимволы интерпретируются как специальные операции. В квадратных скобках указываются необязательные части команд.

2.1. Группирование команд

Группы команд или сложные команды могут формироваться с помощью специальных символов (метасимволов):

- & - процесс выполняется в фоновом режиме, не дожидаясь окончания предыдущих процессов;
- ? - шаблон, распространяется только на один символ;
- - шаблон, распространяется на все оставшиеся символы;
- | - программный канал - стандартный вывод одного процесса является стандартным вводом другого;
- - переадресация вывода в файл;
- < - переадресация ввода из файла;
- ; - если в списке команд команды отделяются друг от друга точкой с запятой, то они выполняются друг за другом;

- && - эта конструкция между командами означает, что последующая команда выполняется только при нормальном завершении предыдущей команды (код возврата 0);
- || - последующая команда выполняется только, если не выполнилась предыдущая команда (код возврата 1);
- () - группирование команд в скобки;
- { } - группирование команд с объединенным выводом;
- [] - указание диапазона или явное перечисление (без запятых);
- >> - добавление содержимого файла в конец другого файла.

Примеры.

who | wc - подсчет количества работающих пользователей командой **wc** (word count - счет слов);

cat text.1 > text.2 - содержимое файла text.1 пересылается в файл text.2;

mail student < file.txt - электронная почта передает файл file.txt всем пользователям, перечисленным в командной строке;

cat text.1,text.2 - просматриваются файлы text.1 и text.2;

cat text.1 >> text.2 - добавление файла text.1 в конец файла text.2;

cc primer.c & - трансляция СИ - программы в фоновом режиме.

cc -o primer.o primer.c - трансляция СИ-программы с образованием файла выполняемой программы с именем primer.o;

rm text.* - удаление всех файлов с именем text;

{cat text.1; cat text.2} | lpr - просмотр файлов text.1 и text.2 и вывод их на печать;

2.2 Основные команды ОС

man <название команды> - вызов электронного справочника об указанной команде.
Выход из справочника - нажатие клавиши Q.
Команда **man man** сообщает информацию о том, как пользоваться справочником.

echo выдача на стандартный вывод строки символов, которая задана ей в качестве аргумента. Синтаксис команды **echo**:
echo [-n] [arg1] [arg2] [arg3]...
Команда помещает в стандартный вывод свои аргументы, разделенные пробелами и завершаемые символом перевода строки. При наличии флага *-n* символ перевода строки исключается. Передаваемая строка может быть перенаправлена в файл с использованием оператора перенаправления вывода > . Например:
$echo "Hello, world!" > myfile

who [am i] - получение информации о работающих пользователях.
В квадратных скобках указываются аргументы команды, которые можно опустить. Ответ представляется в виде таблицы, которая содержит следующую информацию:

- идентификатор пользователя;
- идентификатор терминала;

- дата подключения;
- время подключения.
-

date - вывод на экран текущей даты и текущего времени.

cal [[месяц]год] - календарь; если календарь не помещается на одном экране, то используется команда **cal год | more** и клавишей пробела производится постраничный вывод информации.

top – показывает список работающих в данный момент процессов и информацию о них, включая использование ими памяти и процессора. Список интерактивно формируется в реальном времени. Чтобы выйти из программы top, нажмите клавишу [q].

ps [Опции] **[number]** - команда для вывода информации о процессах:
Опции
- -a все терминальные процессы
- -e все процессы.
- -g*список* выбирать процессы по *списку* лидеров групп.
- -p*список* выбирать процессы по *списку* идентификаторов процессов.
- -t*список* выбирать процессы по *списку* терминалов.
- -u*список* выбирать процессы по *списку* идентификаторов пользователей.
- f генерировать полный листинг.
- -l генерировать листинг в длинном формате.
- number - номер процесса.

Команда ps без параметров выводит информацию только об активных процессах, запущенных с данного терминала, в том числе и фоновых. На экран выводится подробная информация обо всех активных процессах в следующей форме:

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
1	S	200	210	7	0	2	20	80	30	703a	03	0:07	cc
1	R	12	419	7	11	5	20	56	20	03	0:12	ps	

- F - флаг процесса (1 - в оперативной памяти, 2 - системный процесс, 4 - заблокирован в ОЗУ, 20 - находится под управлением другого процесса, 10 - подвергнут свопингу);
- S - состояние процесса (O - выполняется процессором , S - задержан, R - готов к выполнению, I - создается);
- UID - идентификатор пользователя;
- PID - идентификатор процесса;
- PPID - номер родительского процесса;
- C - степень загруженности процессора;
- PRI - приоритет процесса, вычисляется по значению переменной
- NICE и чем больше число, тем меньше его приоритет;
- NI - значение переменной NICE для вычисления динамического
- приоритета, принимает величины от 0 до 39;
- ADDR - адрес процесса в памяти;
- SZ - объем ОЗУ, занимаемый процессом;
- WCHAN - имя события, до которого процесс задержан, для ак-
- тивного процесса - пробел;
- TTY - номер управляющего терминала для процесса;
- TIME - время выполнения процесса;
- CMD - команда, которая породила процесс.

nice [-приращение приоритета] команда[аргументы] - команда изменения приоритета. Каждое запущенное задание (процесс) имеет номер приоритета в диапазоне от 0 до 39, на основе которого ядро вычисляет фактический приоритет, используемый для планирования процесса. Значение 0 представляет наивысший приоритет, а 39 - самый низший. Увеличение номера приоритета приводит к понижению приоритета, присвоенного процессу. Команда
nice -10 ls -l
увеличивает номер приоритета, присвоенный процессу ls -l на 10.
renice 5 1836 - команда устанавливает значение номера приоритета процесса с идентификатором 1836 равным 5. Увеличить приоритет процесса может только администратор системы.

kill [-sig] <идентификатор процесса> - прекращение процесса до его программного завершения. sig - номер сигнала. Sig = -15 означает программное (нормальное) завершение процесса, номер сигнала = -9 - уничтожение процесса. По умолчанию sig= -9. Вывести себя из системы можно командой kill -9 0. Пользователь с низким приоритетом может прервать процессы, связанные только с его терминалом.

free – Показывает общее количество свободной и используемой физической памяти и памяти отведенной для свопирования в системе, так же и совместно используемую память и буфера используемые ядром. Синтаксис :
free [-b | -k | -m] [-o] [-s *delay*] [-t] [-V]
Опции :

- -b показывает количество памяти в байтах; опция -k (по умолчанию) показывает количество памяти в килобайтах;
- Опция -m показывает количество памяти в мегабайтах.
- -t показывает строки содержащие полное количество памяти.
- -o запрещает показывать строки относящиеся к "массиву буфера" . Если не определено отнять/добавить память буферов из/в используемую/свободную память (соответственно!).
- -s разрешает безостановочно выводить информацию с промежутком в *delay* секунд.
- -V показывает информацию о версии программы.

tty – информация о терминалах пользователя.

2.3 Команды работы с файловой системой

Для управления файловой системой имеются различные команды, реализующие операции по созданию, чтению, копированию, переименованию/перемещению, изменению и удалению файлов и каталогов. Как правило, это специализированные команды, хорошо выполняющие свою задачу, однако некоторые функции могут частично дублироваться другими командами, что только добавляет гибкости управлению файлами.
Основными командами для выполнения файловых операций являются: pwd, ls, cp, mv, dir, rm, cd, rmdir, mvdir, mkdir, ln. Информацию о их назначении и параметрах доступна в справке.

pwd – Выдача имени текущего каталога

cd – Смена текущего каталога Синтаксис команды:
cd [каталог]

Команда cd применяется для того, чтобы сделать заданный каталог текущим. Если каталог не указан, используется значение переменной окружения $HOME (обычно это каталог, в который Вы попадаете сразу после входа в систему). Если каталог задан полным маршрутным именем, он становится текущим. По отношению к новому каталогу нужно иметь право на выполнение, которое в данном случае трактуется как разрешение на поиск. Текущий каталог - это каталог, в котором в данный момент находится пользователь. При наличии прав доступа, пользователь может перейти после входа в систему в другой каталог. Текущий каталог обозначается точкой (**.**); родительский каталог, которому принадлежит текущий, обозначается двумя точками (**..**).

cat <имя файла> - вывод содержимого файла на экран.
Команда **cat > text.1 –** создает новый файл с именем text.1, который можно заполнить символьными строками, вводя их с клавиатуры. Нажатие клавиши *Enter* создает новую строку. Завершение ввода – нажатие *Ctrl - d*.
Команда **cat text.1 > text.2** пересылает содержимое файла text.1 в файл text.2.
Слияние файлов осуществляется командой **cat text.1 text.2 > text.3**.

ls [-опции] [имя] - вывод содержимого каталога на экран. Если аргумент не указан, выдается содержимое текущего каталога. Аргументы команды:

- l — список включает всю информацию о файлах;
- t – сортировка по времени модификации файлов;
- a – в список включаются все файлы, в том числе и те, которые начинаются с точки;
- s – размеры файлов указываются в блоках;
- d – вывести имя самого каталога, но не содержимое;
- r – сортировка строк вывода;
- i – указать идентификационный номер каждого файла;
- v – сортировка файлов по времени последнего доступа;
- q – непечатаемые символы заменить на знак ?;
- c – использовать время создания файла при сортировке;
- g – то же что -l, но с указанием имени группы пользователей;
- f – вывод содержимого всех указанных каталогов, отменяет флаги -l, -t, -s, -r и активизирует флаг -a;
- C – вывод элементов каталога в несколько столбцов;
- F – добавление к имени каталога символа / и символа * к имени файла, для которых разрешено выполнение;
- R – рекурсивный вывод содержимого подкаталогов заданного каталога.

Пример
ls -l file.1 - чтение атрибутов файла;

mkdir – Создание каталога. Синтаксис:
mkdir [-m режим_доступа] [-p] каталог ...
По команде mkdir создается дин или несколько каталогов с режимом доступа 0777 [возможно измененном с учетом umask и опции -m]. Стандартные файлы (. - для самого каталога и .. - для вышележащего) создаются автоматически; их нельзя создать по имени. Для создания каталога необходимо располагать правом записи в вышележащий каталог. Идентификаторы владельца и группы новых каталогов устанавливаются соответственно равными реальным идентификаторам владельца и группы процесса. Командой mkdir обрабатываются две опции:

- -m режим_доступа - (явное задание режима_доступа для создаваемых каталогов [см. chmod]).

- -p (при указании этой опции перед созданием нового каталога предварительно создаются все несуществующие вышележащие каталоги).

cp – Копирование файлов. Синтаксис :
cp файл1 [файл2 ...] целевой_файл
Команда cp копирует файл1 в целевой_файл. Файл1 не должен совпадать с целевым_файлом (будьте внимательны при использовании метасимволов shell'a). Если целевой_файл является каталогом, то файл1, файл2, ..., копируются в него под своими именами. Только в этом случае можно указывать несколько исходных файлов.
Если целевой_файл существует и не является каталогом, его старое содержимое теряется. Режим, владелец и группа целевого_файла при этом не меняются.
Если целевой_файл не существует или является каталогом, новые файлы создаются с теми же режимами, что и исходные (кроме бита навязчивости, если Вы не суперпользователь). Время последней модификации целевого_файла (и последнего доступа, если он не существовал), а также время последнего доступа к исходным файлам устанавливается равным времени, когда выполняется копирование.
Если целевой_файл был ссылкой на другой файл, все ссылки сохраняются, а содержимое файла изменяется.
Пример:
cp file.1 file.2 - копирование файла с переименованием;

mv – Перемещение (переименование) файлов. Синтаксис команды:
mv [-f] файл1 [файл2 ...] целевой_файл
Команда mv перемещает (переименовывает) файл1 в целевой_файл. Файл1 не должен совпадать с целевым_файлом (будьте внимательны при использовании метасимволов shell'a).
Если целевой_файл является каталогом, то файл1, файл2, ..., перемещаются в него под своими именами. Только в этом случае можно указывать несколько исходных файлов.
Если целевой_файл существует и не является каталогом, его старое содержимое теряется. Если при этом обнаруживается, что в целевой_файл не разрешена запись, то выводится режим этого файла [см. chmod] и запрашивается строка со стандартного ввода. Если эта строка начинается с символа y, то требуемые действия все же выполняются, при условии, что у пользователя достаточно прав для удаления целевого_файла. Если была указана опция –f или стандартный ввод назначен не на терминал, то требуемые действия выполняются без всяких запросов. Вместе с содержимым целевой_файл наследует режим файла1.
Если файл1 является каталогом, то он переименовывается в целевой_файл, только если у этих двух каталогов общий надкаталог; при этом все файлы, находившиеся в файле1, перемещаются под своими именами в целевой_файл. Если файл1 является файлом, а целевой_файл - ссылкой, причем не единственной, на другой файл, то все остальные ссылки сохраняются, а целевой_файл становится новым независимым файлом.
Пример:
mv file.1 file.2 - переименование файла file.1 в file.2;
mv file.1 file.2 file.3 directory - перемещение файлов file.1, file.2, file.3 в указанную директорию;

rm – удаление файлов**.** Синтаксис команды:
rm [-f] [-i] файл ...
rm -r [-f] [-i] каталог ... [файл ...]

Команда rm служит для удаления указанных имен файлов из каталога. Если заданное имя было последней ссылкой на файл, то файл уничтожается. Для удаления пользователь должен обладать правом записи в каталог; иметь право на чтение или запись файла не обязательно. Следует заметить, что при удалении файла в Linux, он удаляется навсегда. Здесь нет возможностей вроде "мусорной корзины" в windows 95/98/NT или команды undelete в DOS. Так что, если файл удален, то он удален!
Если нет права на запись в файл и стандартный ввод назначен на терминал, то выдается (в восьмеричном виде) режим доступа к файлу и запрашивается подтверждение; если оно начинается с буквы y, то файл удаляется, иначе - нет. Если стандартный ввод назначен не на терминал, команда rm ведет себя так же, как при наличии опции -f. Допускаются следующие три опции:

- -f Команда не выдает сообщений, когда удаляемый файл не существует, не запрашивает подтверждения при удалении файлов, на запись в которые нет прав. Если нет права и на запись в каталог, файлы не удаляются. Сообщение об ошибке выдается лишь при попытке удалить каталог, на запись в который нет прав (см. опцию -r).
- -r Происходит рекурсивное удаление всех каталогов и подкаталогов, перечисленных в списке аргументов. Сначала каталоги опустошаются, затем удаляются. Подтверждение при удалении файлов, на запись в которые нет прав, не запрашивается, если задана опция -f или стандартный ввод не назначен на терминал и не задана опция -i. При удалении непустых каталогов команда rm -r предпочтительнее команды rmdir, так как последняя способна удалить только пустой каталог. Но команда rm -r может доставить немало острых впечатлений при ошибочном указании каталога!
- -i Перед удалением каждого файла запрашивается подтверждение. Опция -i устраняет действие опции -f; она действует даже тогда, когда стандартный ввод не назначен на терминал.

ПРИМЕРЫ
rm file.1 file.2 file.3 - удаление файлов file.1, file.2, file.3;.

Опция -i часто используется совместно с -r. По команде:
rm -ir dirname
запрашивается подтверждение: **directory dirname: ?**
При положительном ответе запрашиваются подтверждения на удаление всех содержащихся в каталоге файлов (для подкаталогов выполняются те же действия), а затем подтверждение на удаление самого каталога.

rmdir – Удаление каталогов. Синтаксис команды:
rmdir [-p] [-s] каталог ...
Команда rmdir удаляет указанные каталоги, которые должны быть пустыми. Для удаления каталога вместе с содержимым следует воспользоваться командой rm с опцией -r. Текущий каталог [см. pwd] не должен принадлежать поддереву иерархии файлов с корнем – удаляемым каталогом. Для удаления каталогов нужно иметь те же права доступа, что и в случае удаления обычных файлов [см. rm]. Командой rmdir обрабатываются следующие опции:

- -p Позволяет удалить каталог и вышележащие каталоги, оказавшиеся пустыми. На стандартный вывод выдается сообщение об удалении всех указанных в маршруте каталогов или о сохранении части из них по каким-либо причинам.
- -s Подавление сообщения, выдаваемого при действии опции -p.

grep – поиск файлов с указанием или без указания контекста (шаблона поиска). Синтаксис:
grep [-vcilns] [шаблон поиска] <имя файла>

Значение ключей:

- v – выводятся строки, не содержащие шаблон поиска;
- c – выводится только число строк, содержащих или не содер-
- жащих шаблон;
- i – при поиске не различаются прописные и строчные буквы;
- l – выводятся только имена файлов, содержащие указанный
- шаблон;
- n – перенумеровать выводимые строки;
- s – формируется только код завершения.

ln (link) – создание ссылок**.** Один файл можно сделать принадлежащим нескольким каталогам. Для этого используется команда:

ln <имя файла 1> <имя файла 2>

Имя 1-го файла - это полное составное имя файла, с которым устанавливается связь; имя 2-го файла - это полное имя файла в новом каталоге, где будет использоваться эта связь. Новое имя может не отличаться от старого. Каждый файл может иметь несколько связей, т.е. он может использоваться в разных каталогах под разными именами.

Команда **ln** с аргументом -s создает символическую связь:

ln -s <имя файла 1> <имя файла 2>

Здесь имя 2-го файла является именем символической связи. Символическая связь является особым видом файла, в котором хранится имя файла, на который символическая связь ссылается. LINUX работает с символической связью не так, как с обычным файлом - например, при выводе на экран содержимого символической связи появятся данные файла, на который эта символическая связь ссылается.

2.4 Режимы доступа к файлам

В LINUX различаются 3 уровня доступа к файлам и каталогам:

1) доступ владельца файла;
2) доступ группы пользователей, к которой принадлежит владелец файла;
3) остальные пользователи.

Для каждого уровня существуют свои байты атрибутов, значение которых расшифровывается следующим образом:

r – разрешение на чтение;
w – разрешение на запись;
x – разрешение на выполнение;
- – отсутствие разрешения.

Первый символ байта атрибутов определяет тип файла и может интерпретироваться со следующими значениями:

- – обычный файл;
d – каталог;
l – символическая связь;
в – блок-ориентированный специальный файл, который соответствует таким периферийным устройствам, как накопители на магнитных дисках;
c – байт-ориентированный специальный файл, который может соответствовать таким периферийным устройствам как принтер, терминал.

В домашнем каталоге пользователь имеет полный доступ к файлам (READ, WRITE, EXECUTE; r, w, x). Атрибуты файла можно просмотреть командой **ls -l** и они представляются в следующем формате:

```
d rwx rwx rwx
```

```
|   |    |  |Доступ для остальных пользователей
|   |    |Доступ к файлу для членов группы
|   |Доступ к файлу владельца
|Тип файла (директория)
```
Пример.Командой **ls -l** получим листинг содержимого текущей директории student:
```
- rwx --- --- 2 student 100 Mar 10 10:30 file_1
- rwx --- r-- 1 adm 200 May 20 11:15 file_2
- rwx --- r-- 1 student 100 May 20 12:50 file_3
```
После байтов атрибутов на экран выводится следующая информация о файле:
- число связей файла;
- имя владельца файла;
- размер файла в байтах;
- дата создания файла (или модификации);
- время;
- имя файла.

Атрибуты файла и доступ к нему, можно изменить командой:
chmod <коды защиты> <имя файла>
Коды защиты могут быть заданы в числовом или символьном виде. Для символьного кода используются:

- знак плюс (+) - добавить права доступа;
- знак минус (-) - отменить права доступа;
- r,w,x - доступ на чтение, запись, выполнение;
- u,g,o - владельца, группы, остальных.

Коды защиты в числовом виде могут быть заданы в восьмеричной форме. Для контроля установленного доступа к своему файлу после каждого изменения кода защиты нужно проверять свои действия с помощью команды ls -l.
Примеры:
chmod g+rw,o+r file.1 - установка атрибутов чтения и записи для группы и чтения для всех остальных пользователей;
ls -l file.1 - чтение атрибутов файла;
chmod o-w file.1 - отмена атрибута записи у остальных пользователей;

2.5 Создание командных файлов

Командный файл в Unix представляет собой обычный текстовый файл, содержащий набор команд Unix и команд Shell. Для того чтобы командный интерпретатор воспринимал этот текстовый файл, как командный необходимо установить атрибут на исполнение.
Установку атрибута на исполнение можно осуществить командой chmod или через mc по клавише F9 выйти в меню и выбрать вкладку File, далее выбрать изменение атрибутов файла.
Например.
$ echo " ps -af " > commandfile
$ chmod +x commandfile
$ ./commandfile
В представленном примере команда echo " ps -af " > commandfile создаст файл с одной строкой " ps -af ", команда chmod +x commandfile установит атрибут на исполнение для этого файла, команда ./commandfile осуществит запуск этого файла.

Задание на лабораторную работу.

1.Ознакомиться с командами Linux. Выполнить команды top, free, ps с различными опциями.

2. Войти в свой домашний каталог. Для этого нужно сделать команду cd ~ . Вы находитесь в своем рабочем каталоге. Здесь хранятся ваши пользовательские файлы и настройки программ, которые вы используете.

3. Создать следующую структуру каталогов и файлов:

1) в домашнем каталоге создать каталог inform
2) Перейти в каталог и inform создать в нем каталог lab1
3) Внутри каталога lab1 создать каталог catalog1, файл file1(например, используя команду echo), каталог catalog2. Перейти в каталог catalog2.
4) Внутри каталога catalog2 создать файлы file3 и file4 , каталог catalog3
5) Внутри каталога catalog3 создать файл file5, жесткую ссылку на файл file1, жесткую
ссылку на каталог catalog2.
6) Создать в каталоге lab1 символичесткую ссылку s_link на файл file5

4. Запустить программу MC (Midnight Commander): mc. Посмотреть структуру созданных вами каталогов и просмотреть содержимое файлов.