ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ имени академика С.П. КОРОЛЕВА»

Изучение операционной системы Linux: файловая структура и инсталляция программного обеспечения

Утверждено Редакционно-издательским советом университета
в качестве методических указаний к лабораторной работе №9

С А М А Р А
Издательство СГАУ
2010

УДК

Составители А.М. С у х о в, Г.М. Г а й н у л л и н а

Рецензент: к.т. н., доц. Попов С.Б.

Изучение операционной системы Linux: файловая структура и инсталляция программного обеспечения: Методические указания к лабораторной работе/ Сост. А.М. Сухов, Г.М. Гайнуллина .-Самара: Изд-во Самарского государственного аэрокосмического университета, 2010. 23 с.

В настоящих методических указаниях приведен материал, необходимый для выполнения лабораторных работ по дисциплине «Информатика».

Целью лабораторных работ является изучение основных возможностей и приобретение навыков работы в операционной системе Linux.

Предназначено для студентов 010900, 140500, 160900, 150100, 200100, 080100 специальностей аэрокосмического профиля.

1. Цель лабораторной работы

- освоить основные принципы работы в операционной системе Linux:
- изучить интерфейс и основные команды
- получить практические навыки работы в операционной системе Linux

2. Основные 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. Для того, чтобы действительно понять *файловую систему* UNIX (а это также касается и файловых систем *Linux*), необходимо заново определить понятие «Что такое файл». С точки зрения UNIX все является файлом. Здесь «всё» действительно означает всё. Жесткий диск, раздел на жестком диске, параллельный порт, подключение к веб-сайту, карта Ethernet - всё это файлы. Даже каталоги являются файлами.

Linux различает много типов файлов в дополнение к стандартным файлам и каталогам. Обратите внимание, что здесь под типом файла мы подразумеваем не *содержимое* файла: в GNU/Linux, как и в любой другой системе UNIX, файл, будь то изображение PNG, двоичный файл или что-либо еще, - это просто поток байтов. Разделение файлов согласно их содержимому оставляется приложениям. Файл — один из базовых элементов любой операционной системы, и Linux здесь не исключение. Но в этой ОС файлу придается особое значение, ведь им описывается любой объект — от текстового документа до устройства. А технологии разграничения прав доступа к файлам являются основой концепции безопасности Linux.

Файловая система Linux

Операционные системы хранят данные на диске при помощи ***файловых систем***. Классическая файловая система представляет данные в виде вложенных друг в друга ***каталогов*** (их ещё называют папками), в которых содержатся ***файлы***. Один из каталогов является «вершиной» файловой системы (а выражаясь технически — «корнем»), в нём содержатся (или, если угодно, из него растут) все остальные каталоги и файлы.

Имена файлов в Linux могут иметь длину до 255 символов и состоять из любых символов, кроме символа с кодом *0* и символа / (слэша). Однако имеется еще ряд символов, которые имеют в оболочке shell специальное значение и которые поэтому не рекомендуется включать в имена. Это следующие символы:

*! @ # $ & ~ % * ( ) [ ] { } ' " \ : ; > < `пробел.*

Если имя файла содержит один из этих символов (это не рекомендуется, но возможно), то вы должны перед этим символом поставить символ обратного слэша "\" (в том числе и перед самим этим слэшем, т. е. повторить его дважды). В Linux различаются символы верхнего и нижнего регистра в именах файлов. Поэтому FILENAME.tar.gz и filename.tar.gz вполне могут существовать одновременно и являться именами разных файлов.

Если жёсткий диск разбит на разделы, то на *каждом* разделе организуется отдельная файловая система с собственным корнем и структурой каталогов (ведь разделы полностью изолированы друг от друга).

В Linux корневой каталог называется весьма лаконично — "/". Полные имена (пути) всех остальных каталогов получаются из "/", к которому дописываются справа имена последовательно вложенных друг в друга каталогов. Имена каталогов в пути также разделяются символом "/" («слэш»). Например, запись */home* обозначает каталог "home" в корневом каталоге ("/"), а */home/user* — каталог "user" в каталоге "home" (который, в свою очередь, в корневом каталоге)[3]. Перечисленные таким образом каталоги, завершающиеся именем файла составляют **полный путь** к файлу.

Относительный путь строится точно так же, как и полный — перечислением через "/" всех названий каталогов, встретившихся при движении к искомому каталогу или файлу. Между полным путём и относительным есть только одно существенное различие: относительный путь начинается *от текущего каталога*, в то время как полный путь всегда начинается *от корневого каталога*. Относительный путь любого файла или каталога в файловой системе может иметь любую конфигурацию: чтобы добраться до искомого файла можно двигаться как по направлению к корневому каталогу, так и от него. Linux различает полный и относительный пути очень просто: если имя объекта *начинается* на "/" — это полный путь, в любом другом случае — относительный.

Стандартные каталоги

В корневом каталоге Linux-системы обычно находятся только подкаталоги со *стандартным* именами. Более того, не только имена, но и *тип данных*, которые могут попасть в тот или иной каталог, также регламентированы стандартом. Этот стандарт довольно последовательно соблюдается во всех Linux-системах: так, в любой Linux вы всегда найдёте каталоги */etc, /home, /usr/bin* и т. п. и сможете довольно точно предсказать, что именно в них находится.

Стандартное размещение файлов позволяет и человеку, и даже программе предсказать, где находится тот или иной компонент системы. Для человека это означает, что он сможет быстро сориентироваться в любой системе Linux (где файловая система организована в соответствии со стандартом) и найти то, что ему нужно. Для программ стандартное расположение файлов — это возможность организации автоматического взаимодействия между разными компонентами системы.

Операции с файлами

Для создания файлов проще всего обратиться к команде cat, используя перенаправление вывода:

$cat > [имя файла].

В этом случае в объект будет помещено всё, что вводится с клавиатуры (окончание операции — одновременное нажатие клавиш Ctrl и D). Разумеется, на практике данный метод используется редко — разве что при необходимости создать небольшой текстовый файл, состоящий из одной-двух строк.

Просмотреть только что созданный файл можно с помощью той же самой команды. Только при этом никакого перенаправления не будет, поскольку задействуется стандартный вывод: *cat [имя файла]*. Обратите внимание, как элегантно и экономно здесь работают консольные команды.

Впрочем, на практике чаще применяются другие программы: *more* и *less*. Синтаксис их довольно прост, в чем вы убедитесь, набрав в консоли команду

$man [название программы]

Для копирования, переименования или перемещения файлов вы можете использовать любой файловый менеджер. Однако тем, кто успел оценить достоинства командной строки, предлагаются другие решения.

Для копирования файлов в Linux существует команда *cp*. Набирать ее следует так:

$cp [параметры] [источник] [приемник]

В роли приемника выступает либо имя файла, либо название каталога, в котором объект будет продублирован с тем же наименованием.

Для сокращения набора как в этой, так и в других командах можно использовать специальные символы. Например, точка обозначает ссылку на текущий каталог, тильда указывает на домашнюю директорию.

За перемещение или переименование файлов отвечает команда *mv*. Скажем, если надо перенести несколько объектов из одного каталога в другой, то следует набрать в консоли

$mv ~/.[расширение] /[каталог назначения]*

С этой командой надо обращаться особенно внимательно. Если в каталоге назначения имеется файл с точно таким же именем, то он будет изменен без предупреждения. С непривычки это кажется не совсем удобным, однако это всего-навсего иная концепция интерфейса: здесь программа является не столько партнером, сколько безмолвным слугой. Таким образом, от пользователя требуется более высокий уровень ответственности.

Удаление объектов системы осуществляется командой *rm*. Если набрать ее без параметров, то никакого предупреждения выдаваться не будет. Учитывая, что использование этой команды (особенно от имени суперпользователя) потенциально опасно, лучше ввести в консоли следующее:

$rm -i [файл ли группа файлов].

В этом случае у вас будет шанс передумать, поскольку система потребует подтверждения.

Для поиска файла предназначена команду *locate*. При этом вместо полного имени можно указывать его часть. Весьма полезная возможность для рассеянных людей, которые не могут удержать всего в памяти.

Если речь идет о команде (программе), то в командной строке нужно набрать *which [имя]*. Этот подход удобен не только для нахождения файла. В частности, таким методом можно быстро узнать, установлено или нет какое-либо приложение. Впрочем, той же цели можно достичь более коротким путем.

Поскольку система Linux поддерживает функцию автозаполнения командной строки, то пользователю достаточно ввести несколько первых символов и нажать на клавишу *Tab*. Вам будет предложено несколько вариантов названия — вспомнить слово, напечатанное на дисплее, всегда проще, чем извлечь его из памяти (естественно, не из оперативной, а из своей собственной).

Поиск

Поиск информации в файловой системе можно условно разделить на поиск по атрибутам файла (понимая их расширительно, то есть включая имя, путь и т. п.) и поиск по содержимому.

Существуют не менее пяти способов поиска файлов в Linux. Здесь мы изложим лишь некоторые из них.

Начнем с простой команды – *find*, которая может выглядеть как:

$find / -name filename.txt

Эта команда задает поиск по всем каталогам от корневого /

Другие примеры использования команды *find*:

$ find /home -user serhiy

Найти все файлы в директории */home* и всех поддиректориях принадлежащие пользователю *serhiy*

$find ~ -name *.c

В вашей домашней директории найдет все файлы с расширением .c. Например *helloworld.c*

$ find . -name "[A-Z]*"

В текущем каталоге и его подкаталогах найдет файлы, начинающиеся с большой буквы. Заметьте что выражение для поиска задано в "..."

$ find /var/www/ -mtime -10

Найти файлы в каталоге */var/www/* и его подкаталогах, которые были изменены менее чем 10 дней назад

$ find /var/www/ -mtime +30 -name "*.php"

Найти все .php файлы в каталоге /var/www/ и его подкаталогах, которые были изменены более чем 30 дней назад

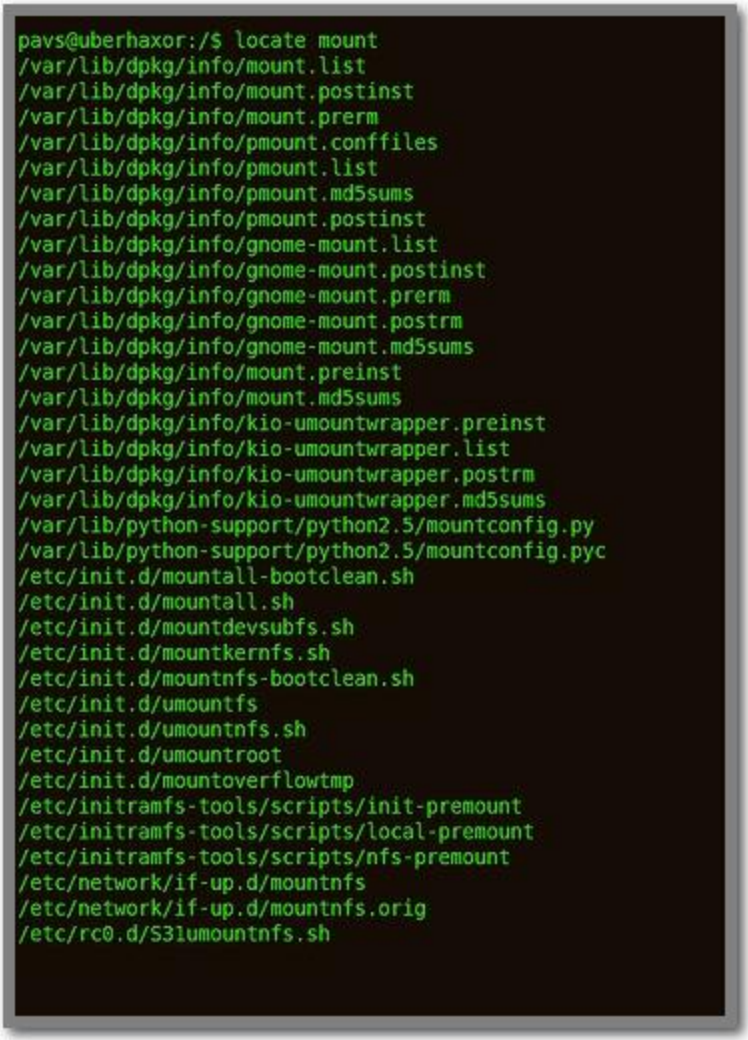
$ find . -perm 777

Найти все файлы в текущем каталоге, которые имеют права доступа 777.

Команда ***locate***

Команда *locate* это альтернатива команде *find -name*. Команда *find* ищет файлы в выбранной части файловой системы и процесс может быть не очень быстрым. С другой стороны, команда *locate* ищет файлы в базе данных,

созданной специально для этих целей */var/lib/locatedb*, что происходит намного быстрее. Для обновления базы используется команда ***updatedb***.



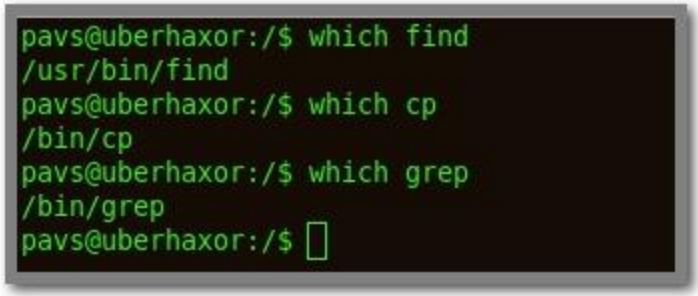
Команда ***whereis***

whereis возвращает место расположения кода (опция -b), ман-страниц (опция -m), и исходные файлы (опция -s) для указанной команды. Если опции не

указываются, выводится вся доступная информация. Эта команда быстрее чем *find,* но менее полная.

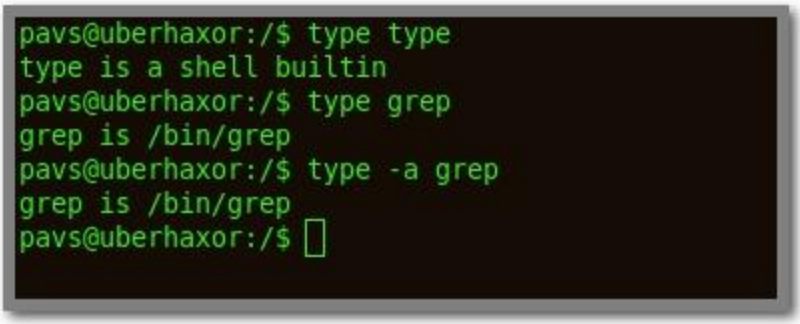
Команда ***which***

Команда *which* ищет все пути, перечисленные в переменной PATH для указанной команды.



Команда ***type***

При вызове без опций показывает, как имена будут интерпретироваться при использовании в качестве имени команды. Если использована опция ***-a***, команда ***type*** выдает список всех каталогов, где есть выполняемый файл с соответствующим именем. В список включаются также псевдонимы и функции, если только не указана опция ***-p***. К хэшу команд не обращаются, если указана опция ***-a***. Команда ***type*** возвращает 0, если хоть один из аргументов найден, и 1 в противном случае.



Монтирование

Корневой каталог в Linux всегда только *один*, а все остальные каталоги в него вложены, т. е. для пользователя файловая система представляет собой единое целое. В действительности, разные части файловой системы могут находиться на совершенно разных устройствах: разных разделах жёсткого диска, на разнообразных съёмных носителях (лазерных дисках, дискетах, флэш-картах), даже на других компьютерах (с доступом через сеть). Для того, чтобы соорудить из этого хозяйства единое дерево с одним корнем, используется процедура **монтирования.**

Монтирование — это подключение в один из каталогов целой файловой системы, находящейся где-то на другом устройстве. Эту операцию можно представить как «прививание» ветки к дереву. Для монтирования необходим пустой каталог — он называется **точкой монтирования**. Точкой монтирования может служить любой каталог, никаких ограничений на этот счёт в Linux нет. При помощи специальной команды (*mount*) мы объявляем, что в данном *каталоге* (пока пустом) нужно отображать файловую систему, доступную на таком-то *устройстве* или же по сети. После этой операции в каталоге (точке монтирования) появятся все те файлы и каталоги, которые находятся на соответствующем устройстве. В результате пользователь может даже и не знать, на каком устройстве какие файлы располагаются.

Подключённую таким образом («смонтированную») файловую систему можно в любой момент отключить — **размонтировать** (для этого имеется специальная команда *umount*), после чего тот каталог, куда она была смонтирована, снова окажется пустым.

Для Linux самой важной является **корневая файловая система** (root filesystem). Именно к ней затем будут подключаться (монтироваться) все остальные файловые системы на других устройствах. Обратите внимание, что

корневая файловая система тоже монтируется, но только не к другой файловой системе, а к «самой Linux», причём точкой монтирования служит "/" (корневой каталог). Поэтому при загрузке системы прежде всего монтируется корневая файловая система, а при останове она размонтируется (в последнюю очередь).

Пользователю обычно не требуется выполнять монтирование и размонтирование вручную: при загрузке системы будут смонтированы все устройства, на которых хранятся части файловой системы, а при останове (перед выключением) системы все они будут размонтированы. Файловые системы на съёмных носителях (лазерных дисках, дискетах и пр.) также монтируются и размонтируются автоматически — либо при подключении носителя, либо при обращении к соответствующему каталогу.

Права доступа в системе Linux

Пользователи и группы

Поскольку система **Linux** с самого начала разрабатывалась как многопользовательская система, в ней предусмотрен такой механизм, как права доступа к файлам и каталогам. Он позволяет разграничить полномочия пользователей, работающих в системе. В частности, права доступа позволяют отдельным пользователям иметь «личные» файлы и каталоги. Например, если пользователь **user** создал в своём домашнем каталоге файлы, то он является владельцем этих файлов и может определить права доступа к ним для себя и остальных пользователей. Он может, например, полностью закрыть доступ к своим файлам для остальных пользователей, или разрешить им читать свои файлы, запретив изменять и исполнять их.

Правильная настройка прав доступа позволяет повысить надёжность системы, защитив от изменения или удаления важные системные файлы. Наконец, поскольку внешние устройства с точки зрения **Linux** также являются объектами файловой системы, механизм прав доступа можно применять и для управления доступом к устройствам.

У любого файла в системе есть *владелец* — один из пользователей. Однако каждый файл одновременно принадлежит и некоторой *группе* пользователей системы. Каждый пользователь может входить в любое количество групп, и в каждую группу может входить любое количество пользователей из числа определённых в системе.

Кроме учётных записей, которые используют люди для работы с системой, в **Linux** предусмотрены учётные записи для т. н. *системных пользователей*: с точки зрения системы это такие же пользователи, которые могут быть

владельцами файлов, однако эти учётные записи используются только для работы некоторых программ-серверов. Например, стандартный системный пользователь **mail** используется программами доставки почты.

Когда в системе создаётся новый пользователь, он добавляется по крайней мере в одну группу. В дальнейшем администратор может добавить пользователя к другим группам.

Механизм групп может применяться для организации совместного доступа нескольких пользователей к определённым ресурсам. Например, на сервере организации для каждого проекта может быть создана отдельная группа, в которую войдут учётные записи (имена пользователей) сотрудников, работающих над этим проектом. При этом файлы, относящиеся к проекту, могут принадлежать этой группе и быть доступными для её членов.

Виды прав доступа

Права доступа определяются по отношению к трём типам действий: чтение **(r)**, запись **(w)** и исполнение **(x).** Эти права доступа могут быть предоставлены трём классам пользователей: владельцу файла (пользователю), группе, которой принадлежит файл, а также всем остальным пользователям, не входящим в эту группу. Право на чтение даёт пользователю возможность читать содержимое файла или, если такой доступ разрешён к каталогам, просматривать содержимое каталога (используя команду *ls*). Право на запись даёт пользователю возможность записывать или изменять файл, а право на запись для каталога — возможность создавать новые файлы или удалять файлы из этого каталога. Наконец, право на исполнение позволяет пользователю запускать файл как программу или сценарий командной оболочки (разумеется, это действие имеет смысл лишь в том случае, если файл является программой или сценарием). Владение правами на исполнение для каталога позволяет перейти (командой *cd*) в этот каталог.

Чтобы получить информацию о правах доступа, используйте команду ***ls*** с ключом ***-l***. При этом будет выведена подробная информация о файлах и каталогах, в которой будут, среди прочего, отражены права доступа. Рассмотрим следующий пример:

$ls –l > ls.txt

$ls -l

-rw—r—r-- user user 430 Ноя 9 11:11 ls.txt

Первое поле в этой строке *(-rw-r--r--)* отражает права доступа к файлу. Третье поле указывает на владельца файла, четвёртое поле указывает на группу,

которая владеет этим файлом. Последнее поле — это имя файла (**ls.txt**). Другие поля описаны в документации к команде *ls*.

Первый символ из этого ряда (-) обозначает тип файла. Символ - означает, что это — обычный файл, который не является каталогом (в этом случае первым символом было бы **d**) или псевдофайлом устройства. Следующие три символа (**rw-**) представляют собой права доступа, предоставленные владельцу **user**. Символ **r** — сокращение от read (англ. читать), а **w** — сокращение от write (англ. писать). Таким образом, **user** имеет право на чтение и запись (изменение) файла.

После символа **w** мог бы стоять символ **x**, означающий наличие прав на исполнение (англ. execute, исполнять) файла. Однако символ -, стоящий здесь вместо **x**, указывает, что **user** не имеет права на исполнение этого файла. Это разумно, так как файл не является программой. В то же время, пользователь, зарегистрировавшийся в системе как **user**, при желании может предоставить себе право на исполнение данного файла, поскольку является его владельцем. Для изменения прав доступа к файлу или каталогу используется команда **chmod**.

Следующие три символа (**r--**) отражают права доступа группы к файлу. Наконец, последние три символа (это опять **r--**) отражают собой права доступа к этому файлу всех других пользователей, помимо собственника файла и пользователей из группы. Так как здесь указан только символ **r**, эти пользователи тоже могут читать файл.

Вот ещё несколько примеров:

-rwxr-x--x

Пользователь-владелец файла может читать файл, изменять и исполнять его; пользователи, члены группы-владельца могут читать и исполнять файл, но не изменять его; все остальные пользователи могут лишь запускать файл на выполнение.

-rw-------

Только владелец файла может читать и изменять его.

-rwxrwxrwx

Все пользователи могут читать файл, изменять его и запускать на выполнение.

Никто, включая самого владельца файла, не имеет прав на его чтение, запись или выполнение. Хотя такая ситуация вряд ли имеет практический смысл, с точки зрения системы она является вполне корректной. Разумеется, владелец файла может в любой момент изменить права доступа к нему.

Возможность доступа к файлу зависит также от прав доступа к каталогу, в котором находится файл. Например, даже если права доступа к файлу установлены как **-rwxrwxrwx**, другие пользователи не могут получить доступ к файлу, пока они не имеют прав на исполнение для каталога, в котором находится файл. Другими словами, чтобы воспользоваться имеющимися у вас правами доступа к файлу, вы должны иметь право на исполнение для всех каталогов вдоль пути к файлу.

«Пользователями» системы **Linux**, выполняющими различные действия с файлами и каталогами, являются на самом деле вовсе не люди, а программы, выполняемые в системе — *процессы*. Одна из таких программ — командная оболочка, которая считывает команды пользователя из командной строки и передаёт их системе на выполнение. Каждая программа (процесс) выполняется от имени определённого пользователя. Её возможности работы с файлами и каталогами определяются правами доступа, определёнными для этого пользователя.

С целью оптимальной настройки прав доступа для ряда программ-серверов в системе созданы системные пользователи (учётные записи), от имени которых работают эти программы. Например, веб-сервер (Apache) выполняется от имени пользователя **apache**, а ftp-сервер — от имени пользователя **ftp**. Такие учётные записи не предназначены для работы людей-пользователей.

Права доступа и администрирование системы

Установка и поддержание оптимальных прав доступа является одной из важнейших задач системного администратора. Права должны быть достаточными для нормальной работы пользователей и программ, но не большими, чем необходимо для такой работы.

Поскольку программы, исполняемые от имени суперпользователя (**root**), могут совершать любые действия с любыми файлами и каталогами, их выполнение может нанести системе серьёзный ущерб. Это может быть как следствием уязвимостей или ошибок в программах, так и результатом ошибочных действий самого пользователя. Поэтому работа с правами суперпользователя требует особой осторожности. Чтобы уменьшить

связанные с этим риски рекомендуется для выполнения задач, требующих таких прав, использовать утилиту ***sudo***.

Основные команды

Ниже перечислены важнейшие команды для решения задач, связанных с правами доступа. Обратитесь к документации по системе для получения информации об использовании этих команд.

$chmod

Изменение прав доступа к файлу или каталогу.

$chown

Изменение владельца файла.

$chgroup

Изменение группы, которой принадлежит файл.

$umask

– определение прав доступа по умолчанию для файлов, создаваемых пользователем

Установка приложений под Linux.

Для Linux приложения поставляются в основном в виде *rpm*-пакетов или **.tar.gz*-архивов. Устанавливается *rpm*-пакет командой

$rpm -i имя пакета

Программа *rpm* сама создаст все необходимые для работы приложения каталоги и положит туда файлы. Если у вас уже установлена предыдущая версия приложения, то в командной строке надо дать ключ *–force* для замены старой версии. *rpm*-пакеты содержатся на CD с большинством дистрибутивов Linux, а также на многих ftp-серверах в Интернете. Если расширение *rpm*-файла выглядит как **.src.rpm*, то это - исходный код приложения, и перед запуском его необходимо самостоятельно откомпилировать (обычно такие пакеты содержат инструкцию о том, как это сделать). Для удаления пакета из системы дайте команду

$rpm -e имя пакета.

Если вам досталось приложение в виде упакованного файла с расширением ****.tar.gz***, то для его распаковки надо дать команду

$tar xzvf имя архив.

Далее необходимо найти файл с инструкциями по установке приложения, каковые в каждом конкретном случае могут различаться.

Кстати, чтобы просмотреть содержимое архивов, не распаковывая их вручную, удобно пользоваться Midnight Commander. При нажатии *Enter* на имени архива вы входите в него, как в обычный каталог.

Хотелось отметить одну полезную программу - *fsck*. Если, к примеру, во время работы в Linux у вас отключилось питание или вы случайно нажали *reset* :-), то при загрузке ОС спросит пароль root, и вы попадете в однопользовательский режим. В нем файловая система смонтирована только для чтения и исполнения. Для того, чтобы восстановить поврежденную файловую систему, дайте команду

$fsck -Aa

После ее окончания дайте команду *reboot*, и после перезагрузки все должно заработать, как раньше.

Для нас также представляет интерес приложение *kpackage - Менеджер пакетов*. Он позволяет устанавливать и удалять приложения Linux, не прибегая к консольной утилите *rpm*, описанной выше. В левой части экрана находится список установленных пакетов, разделенный по категориям. При выборе одного из них в правой части экрана показывается его описание.

Менеджер пакетов Synaptic

Synaptic - это графическая программа, позволяющая управлять пакетами. Она совмещает в себе все возможности консольной системы управления пакетами *apt* и удобство графического интерфейса. С помощью *Synaptic* можно устанавливать, удалять, настраивать и обновлять пакеты в системе, просматривать списки доступных и установленных пакетов, управлять репозиториями и обновлять систему до новой версии.

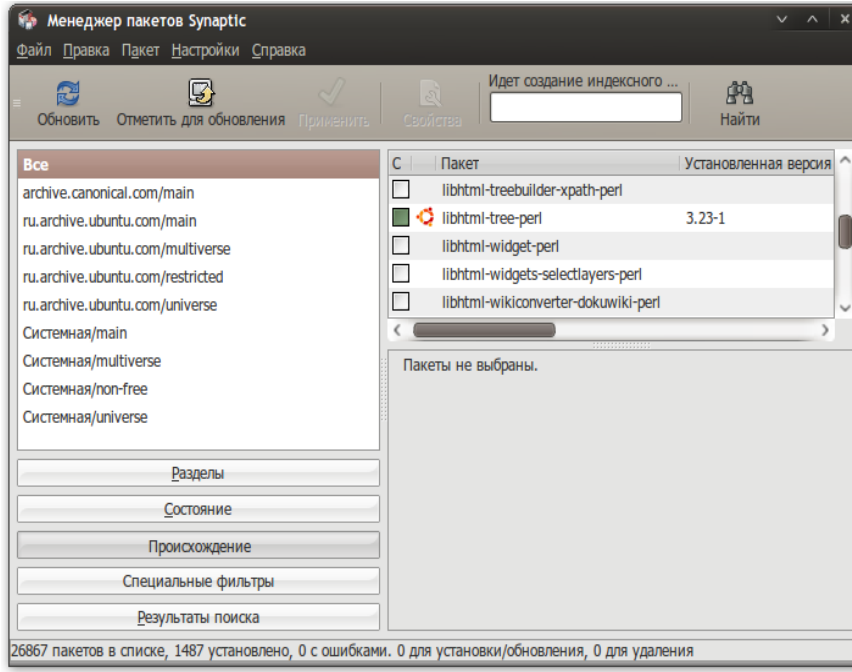
Перед запуском программы вы увидите окно, в которое вам нужно будет ввести свой пароль, для дальнейшей работы с приложением.

Для запуска Synaptic откройте:

Система→Администрирование→Менеджер пакетов Synaptic или введите в терминале:

$sudo synaptec

При запуске вы увидите главное окно программы:



Главное меню вверху, панель с несколькими кнопками, роль которых станет ясна чуть позже.

В левой части экрана внизу есть пять кнопок, которые определяют, что будет показываться в списке над ними, так что вы можете выбирать пакеты в списке, группируя их по статусу.

Если вы выберет «Все», вы увидите полный список доступных и установленных пакетов. При нажатии «Установленные» будут показываться лишь установленные пакеты и так далее. Правая часть окна разделена на верхнюю и нижнюю части. В верхней части выводится список пакетов, и при выбора пакета из этого списка в нижней части отображаются сведения о нем и его описание.

Пакеты могут быть сгруппированы по функциональности (текстовые редакторы, документация, почтовые клиенты и т.д.). Для этого используется кнопка «Разделы». После нажатия на нее вы сможете выбирать пакеты из различных секций.

Для получения подробной информации о пакете, кликните по нему правой кнопкой мыши и в появившемся меню выберите «Свойства».

Установка и удаление ПО

Установка

- Щелкните по кнопке «Обновить» или нажмите *Ctrl + r* для того чтобы скачать список самых последних версий ПО.
- Правый клик на нужном пакете и выберите в появившемся меню «Отметить для установки», или нажмите *Ctrl + I*. Если пакет требует установки другого пакета, то появиться диалоговое окно с изменениями которые будут сделаны, если вы действительно хотите продолжить установку, то щелкните по клавише «Применить» или нажмите *Ctrl + P*.
- Для установки, нажмите кнопку «*Применить*» на главной панели Менеджера пакетов *Synaptic*

Удаление

- Правый клик на нужном пакете и выберите в появившемся меню выберите «Отметить для удаления»
- Появиться диалоговое окно с изменениями которые будут сделаны, если вы действительно хотите продолжить удаление, то щелкните по клавише «Применить» или нажмите *Ctrl + P*.
- Для удаления, нажмите кнопку «*Применить*» на главной панели Менеджера пакетов Synaptic

Если вы отметите пакет маркером «Отметить для полного удаления» то удалится не только выбранный вами пакет, но и все зависимости.

Как исправить сломанные пакеты

«Сломанные пакеты» - это пакеты которые имеют неудовлетворённые зависимости. Если сломанные пакеты обнаружены, то *Synaptic* не позволит проводить ни каких изменений в системе с пакетами до тех пор пока все сломанные пакеты не будут исправлены.

Для исправления сломанных пакетов

1. Выберите *Правка→Исправить пакеты с ошибками* в главном меню.
2. Выберите «Внести отмеченные изменения» в меню «Правка» или нажмите *Ctrl + P*
3. Подтвердите изменения, щелкнув по кнопке «*Применить*».

Горячие клавиши в Synaptic

Команда	**Сочетание клавиш**
Обновить список доступных пакетов	*Ctrl + R*
Открыть диалоговое окно поиска	*Ctrl + F*
Показать окно с свойствами выбранного пакета	*Ctrl + O*
Отметить выбранный(е) пакет(ы) для установки	*Ctrl + I*
Отметить выбранный(е) пакет(ы) для обновления	*Ctrl + U*
Отметить выбранный(е) пакет(ы) для удаления	*Delete*
Отметить выбранный(е) пакет(ы) для полного удаления	*Shift + Delete*
Снять какие-либо изменения в пакетах	*Ctrl + N*
Отметить все возможные обновления	*Ctrl + G*
Быстрая установка специфической версии для пакета	*Ctrl + E*
Отменить последнее изменение	*Ctrl + Z*
Повторить последнее изменение	*Ctrl + Shift + Z*
Применить все выбранные действия	*Ctrl + P*
Выйти из Synaptic	*Ctrl + Q*

Список контрольных вопросов

1. Дайте определение файловой системы
2. Правила для имен файлов в Linux
3. Какие стандартные каталоги Вы знаете?
4. Приведите простейшие операции с файлами и команды их исполняющие.
5. Как производится поиск файла по названию?
6. Расскажите об основных принципах монтирования файловых систем
7. Какие типы прав доступа Вы знаете?
8. Расскажите об основных командах по изменению прав доступа.
9. Расскажите об основных путях инсталляции нового ПО.
10. Менеджер Synaptic.

Задания по лабораторной работе

1. Создайте файл list при помощи команды *cat*.
2. Переместите это файл в любой каталог.
3. Найдите путь к файлу *ifconfig*.
4. Примонтируйте flash карту.
5. Найдите права доступа для файла list.
6. Измените права доступа на полный доступ к файлу.
7. Инсталлируйте утилиту iperf.

Список литературы

1. Уэлш. М. и~др., Руководство по установке и использованию системы **Linux**. М.: ИЛКиРЛ, 1999
2. Александр Боковой, Александр Колотов, Александр Прокудин, Алексей Новодворский, Алексей Смирнов,Анатолий Якушин, Антон Бояршинов, Антон Ионов, Вадим Виниченко, Виталий Липатов, Георгий Курячий, Даниил Смирнов, Дмитрий Аленичев, Дмитрий Левин, Илья Трунин, Кирилл Маслинский, Максим Отставнов, Мэтт Уэлш, Олег Власенко, Сергей Турчин, Станислав Иевлев, Юрий Коновалов и другие; ALT Linux снаружи. ALT Linux изнутри, ISBN 5-9706-0029-6, Издатель: ДМК пресс, 2006 г. Москва
3. Марк Г. Собелл, Практическое руководство по Red Hat Linux: Fedora Core и Red Hat Enterprise Linux, 2-е издание (Practical Guide to Red Hat Linux: Fedora Core and Red Hat Enterprise Linux), 1072 стр., с ил.; ISBN 5-8459-0841-8, 0-13-147024-8; формат 70x100/16;твердый переплет DVD-ROM; 2005, 2 кв.; Вильямс.
4. Разработка приложений в среде Linux. Программирование для linux, 2-е издание, Майкл К. Джонсон, Эрик В. Троан
5. Руководство администратора Linux. Установка и настройка. 2-е издание, Эви Немет, Гарт Снайдер, Трент Хейн
6. Linux. Библия пользователя, Кристофер Негус
7. Linux для чайников , 6-е издание, Ди-Анн Лебланк
8. Разработка ядра Linux, 2-е издание, Роберт Лав
9. Библиотека Qt 4. Программирование прикладных приложений в среде Linux., Чеботарев Арсений Викторович
10. Red Hat Linux Fedora 4. Полное руководство, Пол Хадсон, Эндрю Хадсон, Билл Болл, Хойт Дафф
11. Искусство программирования для Unix, Эрик С. Реймонд
12. Linux для "чайников", 5-е издание, Ди-Анн Лебланк
13. Red Hat Linux. Секреты профессионала, Наба Баркакати
14. Использование Linux, Apache, MySQL и PHP для разработки Web-приложений, Джеймс Ли, Брент Уэр

15. Секреты хакеров. Безопасность сетей - готовые решения, 4-е издание, Стюарт Мак-Клар, Джоэл Скембрей, Джордж Курц
16. FreeBSD: полный справочник., Родерик Смит
17. Секреты хакеров. Безопасность Linux — готовые решения, 2-е издание, Брайан Хатч, Джеймс Ли, Джордж Курц
18. Red Hat Linux 8. Библия пользователя, Кристофер Негус
19. Серверы Linux. Самоучитель, Птицын Константин Александрович
20. Безопасность Linux, 2-е издание, Скотт Манн, Эллен Л. Митчелл, Митчелл Крелл
21. Сетевые средства Linux, Родерик Смит
22. Руководство администратора Linux, Эви Немет, Гарт Снайдер, Трент Хейн
23. Сети TCP/IP, том 3. Разработка приложений типа клиент/сервер для Linux/POSIX, Дуглас Камер, Дэвид Л. Стивенс
24. Секреты хакеров. Безопасность Linux — готовые решения, Брайан Хатч, Джеймс Ли, Джордж Курц
25. Программирование для Linux. Профессиональный подход, Марк Митчелл, Джеффри Оулдем, Алекс Самьюэл
26. Использование Linux, 6-е издание. Специальное издание, Дэвид Бендел, Роберт Нейпир
27. Создание сетевых приложений в среде Linux, Шон Уолтон
28. Освой самостоятельно Linux за 24 часа, 3-е издание,
29. Система электронной почты на основе Linux. Руководство администратора, Ричард Блам
30. Системное администрирование Linux, М. Карлинг, Стефан Деглер, Джеймс Деннис

Учебное издание

Изучение операционной системы Linux: файловая структура и инсталляция программного обеспечения

Методические указания

Составители Сухов Андрей Михайлович

Гайнуллина Гелия Мухаматкамиловна

Изд-во Самарского государственного аэрокосмического университета.

443086 Самара, Московское шоссе, 34.