GeekBrains

Общая версия Linux. Уровень 1

Устройство файловой системы Linux. Понятия файла и каталога

На этом уроке

1. Разберём структуру файловой системы и назначение каталогов.
2. Узнаем, что такое точка монтирования.
3. Разберём типы существующих файлов.
4. Выясним, что такое inode и каталог.
5. Посмотрим, как осуществляется разграничение доступа к файлам и каталогам.

Оглавление

Глоссарий

Раздел — часть долговременной памяти жёсткого диска или флеш-накопителя, выделенная для удобства работы и состоящая из смежных блоков. На одном устройстве хранения может быть несколько разделов.

Точка монтирования — это каталог, с помощью которого обеспечивается доступ к новой файловой системе, каталогу или файлу. Точка монтирования используется для реализации возможности динамически присоединять разделы диска к файловой системе и отсоединять их во время работы операционной системы.

Файловая система — часть операционной системы, которая обеспечивает чтение и запись файлов на дисковых носителях информации. Файловая система устанавливает физическую и логическую структуру файлов, правила их создания и управления ими, а также сопутствующие данные файла и

идентификацию. Конкретная файловая система определяет размер имени файла и максимально возможный размер файла.

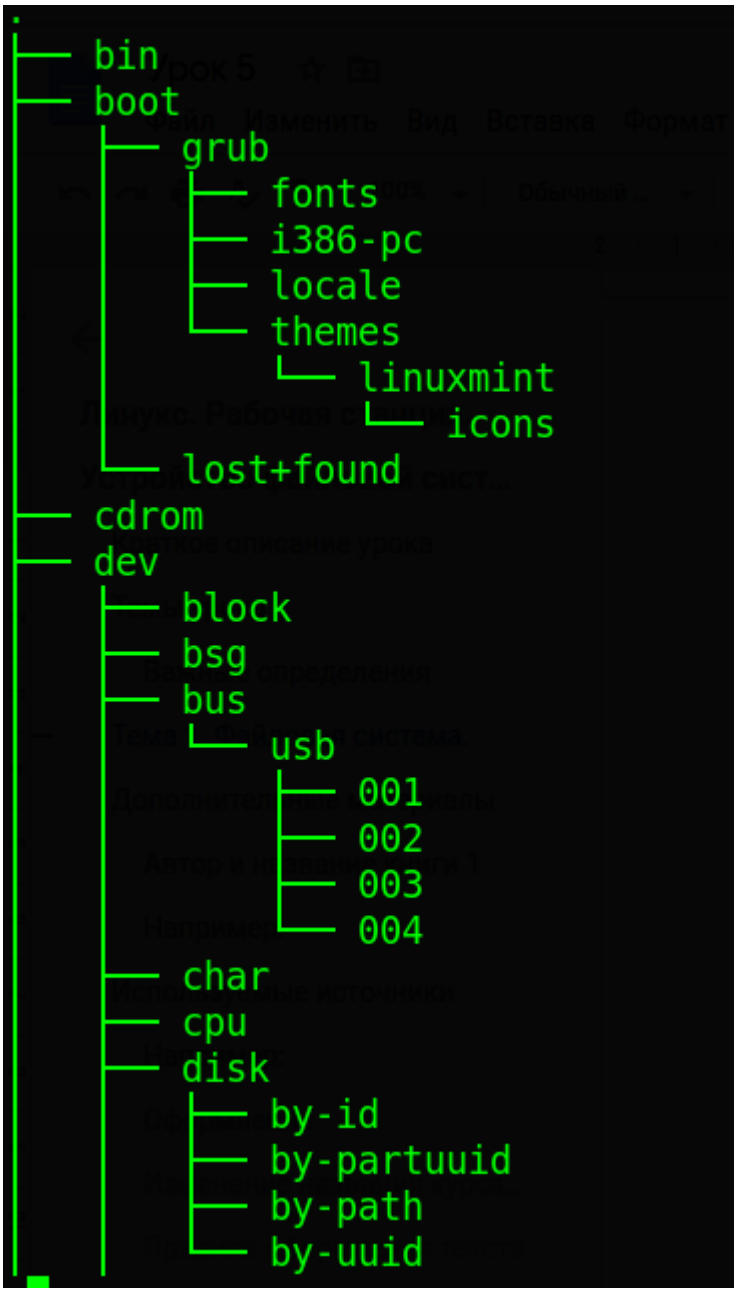
Inode — индексный дескриптор, структура данных в файловых системах Linux. Предназначена для хранения метаинформации о стандартных файлах, каталогах или других объектах файловой системы, кроме непосредственно данных и имени.

Файл — именованная область данных на носителе информации.

Каталог — объект файловой системы, упрощающий организацию файлов. В Linux реализован как специальный файл, где регистрируется информация о других файлах и каталогах файловой системы.

Файловая система

Хранение данных на диске организуется операционной системой при помощи файловых систем. В Linux файловая система представляет данные в виде вложенных друг в друга каталогов, в которых хранятся файлы. Файловая система в Linux — это древовидная структура, в ней есть корневой каталог, от которого «растут» все остальные каталоги и файлы. Дерево каталогов можно посмотреть, используя утилиту **tree.** Она не идёт в стандартной поставке, поэтому её необходимо установить, используя команду `sudo apt install tree -y`. Утилита показывает содержимое каталогов в виде дерева. Корневой каталог обозначается точкой. Для удобства вывода используется параметр **-d** — показывать только каталоги.



Корневой каталог носит название «/» (root), при этом важно не путать его с домашним каталогом суперпользователя /root. Вложенные каталоги могут быть как самостоятельными единицами файловой системы, хранящими в себе файлы и другие каталоги, так и точками монтирования для разделов. Разделы либо организуются на части жёсткого диска, либо занимают всё пространство жесткого диска. На каждом разделе создаётся собственная файловая система. Все разделы изолированы друг от друга.

В целях организации единого пространства разделы подключаются к корневой файловой системе при помощи процедуры монтирования. Монтирование — это подключение созданного раздела жёсткого диска с файловой системой в каталог, который носит название точки монтирования. Такая технология позволяет нам обращаться к подключённым разделам (например, разделу на флешке) как к обычным каталогам. Точкой монтирования может быть любой каталог. Монтирование осуществляется при

помощи команды **mount**. После монтирования в каталоге (точке монтирования) появятся все данные (файлы и каталоги), содержащиеся в разделе. В результате для конечного пользователя всё будет представляться как единое пространство данных. Абсолютно не важно, на каком разделе хранится информация: пользователь будет её видеть в одном из каталогов корневой файловой системы.

Монтирование разделов в точки монтирования при старте операционной системы организуется через специальный файл **/etc/fstab**, в котором прописываются имя устройства, тип файловой системы, точка монтирования и дополнительные опции монтирования.

Операция отключения раздела от точки монтирования называется размонтирование и выполняется при помощи утилиты umount. Более подробно о утилитах mount, umount, а также файле /etc/fstab можно прочитать на соответствующих страницах встроенного справочного руководства man.

Рассмотрим назначение основных каталогов корневой файловой системы:

1. **/boot** — каталог, который хранит в себе конфигурационные файлы загрузчика ОС, образы ядра и файлы initrd.
2. **/bin** и **/sbin/** — в данных каталогах хранятся основные исполняемые файлы и утилиты, необходимые для работы и администрирования операционной системы.
3. **/etc** — каталог предназначен для хранения конфигурационных файлов операционной системы и всех служб.
4. **/home** — стандартный каталог для хранения личных файлов и каталогов пользователей.
5. **/dev** — каталог, в котором хранятся файлы устройств (оборудование ПК, которое подключается при старте ядра ОС).
6. **/proc** — точка монтирования для виртуальной файловой системы procfs, которая используется для хранения и предоставления информации о процессах.
7. **/sys** — точка монтирования для виртуальной файловой системы sysfs, которая используется для хранения и предоставления информации об инициализированных устройствах, сгруппированных по разным критериям: типам устройств, шинам, диагностическим протоколам доступа и т. п. Одни и те же устройства могут быть показаны в разных каталогах через особый тип файлов, который называется символические ссылки, о них поговорим ниже.
8. **/usr** — предназначен для хранения пользовательских программ, документации, исходных кодов программ и ядра.
9. **/var** — каталог содержит в себе постоянно изменяемые файлы, например журналы работы операционной системы.
10. **/lib** — каталог, который хранит в себе библиотеки и модули ядра, необходимые для функционирования ОС.

Понятия файла и каталога

Типы файлов в Linux

Файл — ключевое понятие в Linux. Посредством файлов операционная система взаимодействует с пользователем, процессы взаимодействуют между собой и с пользователем, ядро операционной системы взаимодействует с устройствами компьютера.

Файлы в Linux бывают нескольких типов:

1. **Обычные файлы** — файлы программ, конфигурационные файлы, пользовательские файлы.
2. **Каталоги** — особый тип файла, хранящий в себе информацию о других файлах и каталогах файловой системы.
3. **Файлы физических устройств** включают в себя блочные и символьные файлы. **Блочные файлы** — это файлы, которые обеспечивают буферизированный доступ к устройствам, то есть в буфере накапливается определённое количество данных, которое единым блоком записывается на устройство. Такие файлы используются, например, для работы с жёсткими дисками и разделами жёстких дисков. **Символьные файлы** — файлы, которые не имеют буфера, данные на устройство выводятся посимвольно. Используются, например, для вывода информации на консоль.
4. **Именованные каналы (pipe)** — файлы, предназначенные для взаимодействия между процессами. Один процесс записывает информацию в канал, второй её оттуда считывает.
5. **Файлы сокеты (sockets)** — файлы, предназначенные для взаимодействия между процессами, но, в отличие от именованных каналов, данные файлы работают в обе стороны. То есть один процесс записывает информацию в такой файл, второй её считывает, обрабатывает и записывает обратно в этот файл, чтобы первый процесс смог её обработать.
6. **Файлы символические ссылки** — аналоги ярлыков в ОС семейства Windows, используются в случаях, когда у файла должно быть несколько имён.

Inode и каталог

С каждым файлом в операционной системе Linux связана особая структура данных — индексный дескриптор (**inode**). Эти данные хранят метаинформацию о файле: владелец, права доступа, время последнего изменения и т. д. Inode также содержит информацию о физическом расположении данных.

Inode уникальны на уровне разделов. Вы можете иметь два файла с одинаковым номером inode, если они находятся в разных разделах. Информация inode хранится в табличном формате в виде структуры в начале каждого раздела. Просмотреть inode можно, используя команду **ls -li**.

Каталог — это файл особого типа, который содержит таблицу соответствия **имя_файла → inode**. В этой таблице требуется уникальность имён, но не уникальность номеров inode. Благодаря этому

каждый объект файловой системы может иметь несколько имён. Счётчик имён хранится в inode объекта.

Жёсткие и символические ссылки

Ссылки — это особенность файловой системы, которая позволяет размещать один и тот же файл в разных каталогах.

Жёсткая ссылка — это запись в каталоге, указывающая на inode. Жёсткая ссылка создаётся только для файлов, за исключением специальных записей, указывающих на саму директорию (**.**) и родительскую директорию (**..**). Жёсткие ссылки используются только в пределах одного раздела. На практике жёсткие ссылки уже почти не применяются в работе. Создать жёсткую ссылку можно, используя команду **ln** без параметров, например `ln file link`, где `file` — имя файла-источника, `link` — имя ссылки на файл. Родительский файл можно перемещать или даже удалять без вреда для ссылки, то есть фактически файл, на который есть хотя бы одна жёсткая ссылка, не перестанет существовать. Изменение содержимого файла и разрешений также повлияет и на ссылку.

Символическая ссылка — это запись в каталоге, указывающая на имя объекта с другим inode. Символическая ссылка наиболее близка к ярлыку в Windows. Символическая ссылка может ссылаться на файл и на каталог. Символические ссылки могут существовать на разных разделах. Права доступа и inode у символической ссылки отличаются от родительского файла. При перемещении или удалении файла-родителя символическая ссылка «ломается», так как ссылка привязывается именно к имени файла. Создать символическую ссылку можно командой `ln -s file link`, где параметр **-s** говорит, что мы создаём символическую ссылку, `file` — имя источника файла, `link` — имя ссылки на файл.

Права доступа к файлам и каталогам

У файлов и каталогов есть ряд атрибутов, хранящихся в inode: владелец и группа владельца, права доступа к файлу или каталогу, время последнего изменения и т. п. Полный вывод атрибутов мы можем посмотреть, выполнив команду `ls -l`**.** На скриншоте вы видите вывод работы команды ls -l в домашней директории пользователя user1:

```
drwxr-xr-x 1 user1 user1   32 мар  6 20:10 .config
drwxrwxr-x 1 user1 user1    0 мар 16 10:24 dir
-rw-rw-r-- 1 user1 user1    0 мар 16 10:24 file
```

Первый столбец вывода покажет права доступа к файлу или каталога. Символы столбца можно условно разделить на четыре группы:

1. Первая группа, состоящая из единственного символа (самый первый символ в строке), определяет тип файла согласно следующему списку:

- — обычный файл;

d — каталог;

b — файл блочного устройства;

c — файл символьного устройства;

s — socket;

p — именованный канал (pipe);

l — символическая ссылка (link).

Следующие три группы состоят из трёх символов **rwx**, которые определяют права доступа к файлу или каталогу следующим образом:

2. Первый блок **rwx** определяет права доступа для **владельца файла или каталога**, буквенное обозначение — **u.**
3. Второй блок **rwx** определяет права доступа для **владельца группы**, участники которой имеют доступ к файлу или каталогу согласно установленным разрешениям, буквенное обозначение — **g**.
4. Третий блок **rwx** определяет права доступа для всех остальных пользователей, не подходящих ни под одно определение, буквенное обозначение — **o**.

Символы в блоках — это права доступа. Они расшифровываются следующим образом:

- **r (read)** — возможность открытия и чтения файла или просмотр содержимого каталога.
- **w (write)** — возможность изменить содержимое файла или возможность создавать, удалять или переименовывать объекты в каталоге.
- **x (execute)** — возможность выполнить файл (запустить программу, скрипт) или возможность войти в каталог и получить атрибуты объектов.

Права доступа можно представить в численном виде, используя восьмеричную систему счисления, согласно таблице:

	r	w	x		r	w	x		
–	–	–	–		0	0	0		0
x	–	–	x		0	0	1		1
w	–	w	–		0	1	0		2
wx	–	w	x		0	1	1		3
r	r	–	–		1	0	0		4
rx	r	–	x		1	0	1		5

rw	r	w	–		1	1	0		6
rwx	r	w	x		1	1	1		7

Предоставление прав пользователю в таком виде происходит путём сложения соответствующих чисел в таблице.

Для назначения прав доступа к файлам и каталогам используется команда **chmod**. Важно помнить, что **chmod** не изменяет права доступа к символическим ссылкам. Задавать права доступа можно, используя как буквенную запись прав, так и численную.

Буквенная запись

`chmod [объект назначения] [+|-| =] [права доступа] [файл | каталог ]`, где:

- **[объект назначения]** — один из типов пользователей: владелец файла (u), группа-владелец (g), все остальные (o);
- **[+|-|=]** — одно из действий: «**+**» — добавить указанные права к существующим правам объекта, «**-**» — убрать указанные права с объекта, «**=**» — заменить права объекта на указанные;
- **[права доступа]** — это права доступа rwx из предыдущего пункта;
- **[файл | каталог]** — файл или каталог, которому назначаем права доступа.

Например, `chmod u+x file` добавит владельцу файла право на выполнение, `chmod g-w file` уберёт у группы-владельца право на запись в файл, `chmod go=r file` сменит права группы-владельца и всех остальных на право только чтения файла.

Для каталогов команда **chmod** используется таким же образом, как и для файлов, но важно учитывать, что, если мы хотим присвоить права не только каталогу, но и объектам внутри каталога, необходимо использовать параметр **-R (рекурсивно)**. `Chmod -R 655 dir1` присвоит права чтение-запись чтение-исполнение для каталога **dir1** и для всего содержимого этого каталога, файлы в этом каталоге получат право на выполнение для группы-владельца и всех остальных.

Численная запись

Такой принцип задания прав доступа наиболее распространён:

```
chmod [права доступа в численном виде] [файл | каталог ]
```

Например, **chmod 766 file** присвоит права чтение-запись-выполнение для владельца и чтение-запись для группы-владельца и всех остальных.

Специальные биты

Кроме вышеперечисленных атрибутов прав доступа к файлам и каталогам, существуют ещё специальные биты **SUID, SGID, Sticky**.

SUID (set user ID upon execution) — установка ID пользователя во время выполнения. Разрешает пользователям запускать файл на исполнение с правами того пользователя, которому принадлежит данный файл. В символьной записи присваивается следующим образом: `chmod u+s file`, где s — bit suid. В численной записи имеет значение **4000**, например `chmod 4755 file` присвоит права на чтение-запись-выполнение для владельца, права на чтение-выполнение — для группы владельца и всех остальных, 4 — выставит bit suid. **SUID** работает с файлами.

SGID (set group ID upon execution) — установка ID группы во время выполнения, применяется преимущественно к каталогам. Данный атрибут устанавливает идентификатор группы каталога, а не группы владельца, который создал файл в этом каталоге. В символьной записи присваивается следующим образом: `chmod g+s dir1`, где **s** — bit sgid. В численной записи имеет значение **2000,** например `chmod 2665 dir1` присвоит права на чтение-запись-просмотр для владельца и группы владельца и чтение-просмотр для всех остальных, а также установит на каталог bit sgid — 2.

Sticky — дополнительный атрибут, который устанавливается для каталогов. Файлы из каталога с таким битом может удалить только владелец (пользователь, создавший этот файл). В символьной записи присваивается следующим образом: `chmod +t dir1` добавит к каталогу bit sticky. В численной записи имеет значение **1000**, например `chmod 1666 dir1` установит на каталог права на чтение-запись-просмотр для всех типов пользователей, но при этом удалять файлы могут только создатели благодаря биту sticky 1.

Практическое задание

1. Создать файл file1 и наполнить его произвольным содержимым. Скопировать его в file2. Создать символическую ссылку file3 на file1. Создать жёсткую ссылку file4 на file1. Посмотреть, какие inode у файлов. Удалить file1. Что стало с остальными созданными файлами? Попробовать вывести их на экран.
2. Дать созданным файлам другие, произвольные имена. Создать новую символическую ссылку. Переместить ссылки в другую директорию.
3. Создать два произвольных файла. Первому присвоить права на чтение и запись для владельца и группы, только на чтение — для всех. Второму присвоить права на чтение и запись только для владельца. Сделать это в численном и символьном виде.
4. * Создать группу developer и нескольких пользователей, входящих в неё. Создать директорию для совместной работы. Сделать так, чтобы созданные одними пользователями файлы могли изменять другие пользователи этой группы.

5. * Создать в директории для совместной работы поддиректорию для обмена файлами, но чтобы удалять файлы могли только их создатели.
6. * Создать директорию, в которой есть несколько файлов. Сделать так, чтобы открыть файлы можно было, только зная имя файла, а через ls список файлов посмотреть было нельзя.

Дополнительные материалы

Типы файловых систем Linux

inode и каталоги

Права доступа к файлам в Linux

Используемые источники

Робачевский Андрей М. Операционная система Unix

В.Костромин, "Linux для пользователя",изд."БХВ-Петербург", 2002 г., серия "Самоучитель"

Права доступа и специальные биты в Ubuntu