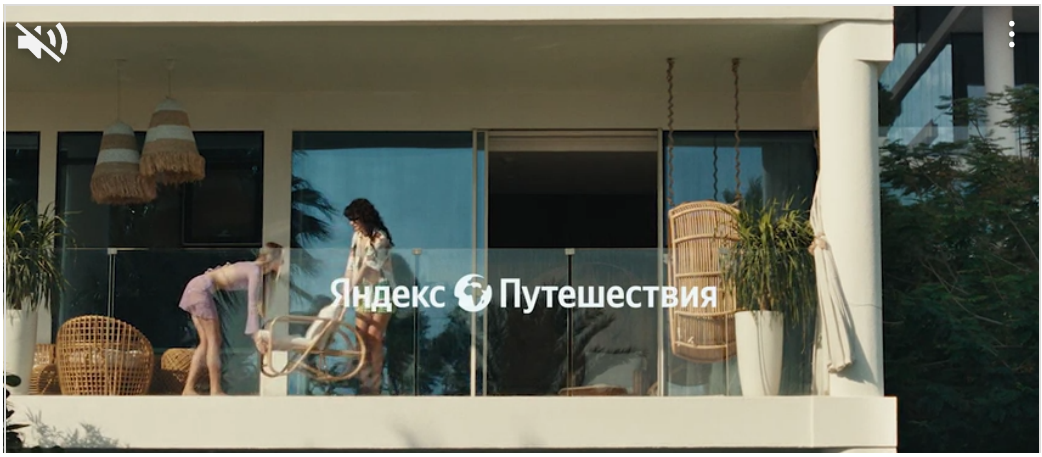
Sysadminium
База знаний системного администратора

Ссылки в Linux

В предыдущей статье мы познакомились с разными типами файлов в Linux, в этой статье подробнее разберем ссылки в этой операционной системе.

Оглавление [скрыть]



Теория

Файлы и каталоги на жестком диске хранятся в виде набора блоков. Информация о файле (метаданные), например время создания файла, хранится в индексном дескрипторе – **inode**. Номер inode является уникальным на файловой системе и связывается с определенным набором блоков.

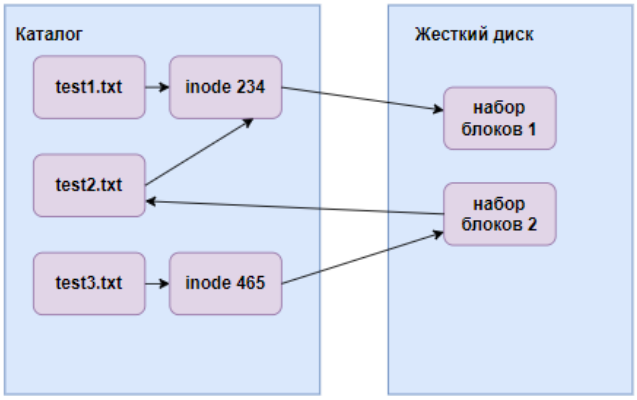
А имя файла это всего лишь понятное для человека наименование которое относится к определенному inode.

Каталог – это файл особого типа в котором содержится список имен файлов или других каталогов и указатели на inode для этих файлов.

Ссылки в LInux – это дополнительные записи в каталоге, позволяющие обращаться к файлам или каталогам по другим именам. В Linux различают два вида ссылок:

- **Жесткая** – дополнительная запись в каталоге указывающая на уже созданный inode.
- **Символьная** – запись в каталоге указывающая на свой inode, но при чтении ссылающая на другой файл. При этом ссылается на другой файл по имени а не по inode.

Чтобы было понятнее посмотрите на следующий рисунок:



Жесткие и символьные ссылки в Linux

Есть файл с именем **test1.txt** и он имеет свой уникальный индексный дескриптор (**inode 234**), который ссылается на набор блоков (**набор блоков 1**) на жестком диске.

Файл **test2.txt** это жесткая ссылка. То есть, просто, дополнительное имя файла, которое ссылается на тот же **inode** что и **test1.txt**.

Файл **test3.txt** это символьная ссылка. Этот файл имеет свой индексный дескриптор (**inode 465**), и обращается к своему набору блоков (**набор блоков 2**). Этот файл является файлом отдельного типа (**символьная ссылка / symbolic link**) и в наборе данных у таких файлов записан путь к файлу на который она ссылается. То есть, в файле **test3.txt** указано что ссылаться нужно на **test2.txt**. При этом, если удалить файл **test2.txt** и попробовать прочитать **test3.txt** то увидим ошибку, так как ссылка не найдет файл на который она ссылается.

Жесткие ссылки в Linux имеют свои ограничения, их можно:

- создавать только для файлов, а для каталогов нельзя;
- использовать только в пределах одной файловой системы.

При удалении жёстких ссылок, физически файл удаляется (точнее его inode удаляется) только тогда, когда все жесткие ссылки идущие на этот inode удаляются.

Особенность символьных ссылок:

- удаление ссылки не приведет к удалению файла на который она ссылается;
- их можно создавать на объекты других файловых систем;
- их можно создавать и на файлы и на каталоги.

Практика

Все примеры я проведу на Debian 11, но в Ubuntu 22.04 всё происходит аналогично.

Для работы создадим один каталог, а в нем один файл:

```
alex@deb:~$ mkdir dir1
alex@deb:~$ touch dir1/file1.txt
alex@deb:~$ ls -R
.:
dir1

./dir1:
file1.txt
```

Для создания ссылок используется команда **ln**. Создадим **жесткую ссылку** на файл **file1.txt** в домашнем каталоге:

```
alex@deb:~$ ln dir1/file1.txt file2.txt
```

При этом вначале указывается объект на который будем ссылаться, а затем новое имя файла.

Запишем что-нибудь в этот файл:

```
alex@deb:~$ echo 12345 > file2.txt
alex@deb:~$ cat file2.txt
12345
```

Теперь прочитаем файл dir1/file1.txt:

```
alex@deb:~$ cat dir1/file1.txt
12345
```

Как видим два файла ссылаются на один объект. При правке одного файла, правится и второй.

Теперь переместим file2 в dir1 и проверим что ничего не сломалось и это тот же самый файл:

```
alex@deb:~$ mv file2.txt dir1/
alex@deb:~$ ls dir1/
file1.txt  file2.txt
alex@deb:~$ cat dir1/file2.txt
12345
alex@deb:~$ cat dir1/file1.txt
12345
```

Теперь создадим **символьную ссылку** file3.txt на файл file2.txt, для этого нужно команде **ln** указать опцию **-s**:

```
alex@deb:~$ ln -s dir1/file2.txt file3.txt
alex@deb:~$ cat file3.txt
12345
```

То что это символьная ссылка можно выяснить посмотрев вывод команды **ls -l**:

```
alex@deb:~$ ls -l
итого 4
```

```
drwxr-xr-x 2 alex alex 4096 янв 12 11:40 dir1
lrwxrwxrwx 1 alex alex   14 янв 12 11:42 file3.txt -> dir1/file2.txt
```

Обратите внимание на самый первый символ (**l**), что означает что этот файл – символьная ссылка.

Также вывод показывает на что ссылается данный объект. В данном примере ссылка идет по относительному пути, то есть не начинается с корня (/). Про относительный и абсолютный пути я писал в этой статье.

Теперь давайте переместим ссылку file3.txt в каталог dir1 и посмотрим что произойдет:

```
alex@deb:~$ mv file3.txt dir1/
alex@deb:~$ ls -l dir1/
итого 8
-rw-r--r-- 2 alex alex  6 янв 12 11:38 file1.txt
-rw-r--r-- 2 alex alex  6 янв 12 11:38 file2.txt
lrwxrwxrwx 1 alex alex 14 янв 12 11:42 file3.txt -> dir1/file2.txt
alex@deb:~$ cat dir1/file3.txt
cat: dir1/file3.txt: Нет такого файла или каталога
```

Так как путь ссылки не изменился, а файл ищется по относительному пути **dir1/file2.txt**, а в каталоге **dir1** нет каталога **dir1**, то мы получили ошибку при попытке прочитать данный файл. Такие ссылки, которые ведут на объект которого нет, называются **битыми ссылками**.

Следует иметь ввиду, если вы создаете символьную ссылку с относительным путем, то подразумевается что ссылку в дальнейшим не будут перемещать из её каталога.

Если бы мы использовали абсолютный путь при создании ссылки, например так:

```
alex@deb:~$ ln -s /home/alex/dir1/file2.txt file3.txt
alex@deb:~$ ls -l
итого 4
drwxr-xr-x 2 alex alex 4096 янв 12 11:50 dir1
lrwxrwxrwx 1 alex alex   25 янв 12 11:59 file3.txt ->
/home/alex/dir1/file2.txt
```

То такую ссылку можно перемещать. Но если переместить или удалить сам файл на который ссылается ссылка, то ссылка в любом случае станет битой.

Как распознать ссылки

То что файл является символьной ссылкой видно из вывода **ls -l**, это мы уже разбирали:

```
lrwxrwxrwx 1 alex alex   14 июл 18 16:44 file3.txt -> dir1/file2.txt
```

Но чтобы проверить битая ли это ссылка можно воспользоваться командой **ls** с опцией **-L**, которая попытается достучаться до всех файлов на которые есть символьные ссылки:

```
alex@deb:~$ ls -lL dir1/
ls: невозможно получить доступ к 'dir1/file3.txt': Нет такого файла или
каталога
итого 8
-rw-r--r-- 2 alex alex 6 янв 12 11:38 file1.txt
-rw-r--r-- 2 alex alex 6 янв 12 11:38 file2.txt
l????????? ? ?    ?    ?            ? file3.txt
```

В выводе мы сразу заметим что файла **dir1/file3.txt** не существует, а ниже вопросики означают что **file3.txt** это битая ссылка.

Немного сложнее дело обстоит с жесткими ссылками. Давайте еще раз посмотрим на файлы file1.txt и file2.txt, которые ссылаются на один inode и являются жесткими ссылками друг для друга:

```
alex@deb:~$ ls -l dir1/
итого 8
```

```
-rw-r--r-- 2 alex alex  6 янв 12 11:38 file1.txt
-rw-r--r-- 2 alex alex  6 янв 12 11:38 file2.txt
lrwxrwxrwx 1 alex alex 14 янв 12 11:42 file3.txt -> dir1/file2.txt
```

Обратите внимание на двоечки после -rw-r-r- , это число жестких ссылок. Тут мы можем понять что file1.txt ссылается на объект у которого две жесткие ссылки, и тоже самое про file2.txt. Но то что это жесткие ссылки одного и того же объекта еще не понятно.

У **ls** есть опция **-i**, которая дополнительно выводит номер inode:

```
alex@deb:~$ ls -li dir1/
итого 8
783380 -rw-r--r-- 2 alex alex  6 янв 12 11:38 file1.txt
783380 -rw-r--r-- 2 alex alex  6 янв 12 11:38 file2.txt
783384 lrwxrwxrwx 1 alex alex 14 янв 12 11:42 file3.txt -> dir1/file2.txt
```

Как видно file1.txt и file2.txt ссылаются на один inode, под номером 783380. Значит это жесткие ссылки для одного и того же объекта.

В конце удалим все что мы создали:

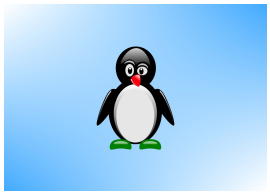
```
alex@deb:~$ rm -rf file3.txt dir1/
```


Предыдущая статья

Вернуться к оглавлению

Следующая статья

Сводка



Имя статьи Ссылки в Linux

Описание В предыдущей статье мы познакомились с разными типами файлов в Linux, в этой статье подробнее разберем ссылки в этой операционной системе

debian linux ubuntu

admin 12.01.2022 Linux Leave a Comment

Слишком много фактических параметров в 1С

Группы и пользователи в Linux

Добавить комментарий

Ваш адрес email не будет опубликован. Обязательные поля помечены *

Комментарий *

Имя *

Email *

Сайт

Сохранить моё имя, email и адрес сайта в этом браузере для последующих моих комментариев.

Отправить комментарий