

Права доступа

Общие понятия

Поскольку система Linux с самого начала разрабатывалась как многопользовательская, в ней предусмотрен такой механизм, как права доступа к файлам и каталогам. Он позволяет разграничить полномочия пользователей, работающих в системе.

У каждого объекта в Linux есть свой **идентификатор**, а так же **права доступа**, применяемые к данному идентификатору. Идентификатор есть у пользователя - UID, у группы - GID, у файла - inode. Собственно inode является, как идентификатором файла/каталога, так и сущностью, которая содержит в себе информацию о файле/каталоге. Например такую, как: принадлежность к **владельцу/группе**, тип файла и **права доступа к файлу**.

Пример

```
main:/# ls -li
```

```
итого 50
```

```
22089 drwxr-xr-x 2 root root 3072 Ноя 15 14:15 bin
```

```
2129 drwxr-xr-x 3 root root 1024 Окт 1 18:03 boot
```

```
12 lrwxrwxrwx 1 root root 11 Окт 1 15:36 cdrom -> media/cdrom
```

```
0121 drwxr-xr-x 50 root root 4096 Ноя 15 14:46 etc
```

```
....
```

22089 это есть номер inode

drwxr-xr-x это есть те самые права доступа и тип файла

2 количество жестких ссылок на файл

root имя владельца файла

root имя группы владельца файла

3072 размер файла

Ноя 15 14:15 дата создания файла

bin имя файла/каталога

Права доступа

- Для каждого объекта файловой системы в модели полномочий Linux есть три типа полномочий: полномочия чтения (*r* от *read*), записи (*w* от *write*) и выполнения (*x* от *execution*). В полномочия записи входят также возможности удаления и изменения объекта. Право выполнения можно установить для любого файла. Потенциально, любой файл в системе можно запустить на выполнение, как программу в Windows. В Linux является ли файл исполняемым или нет, определяется не по его расширению, а по правам доступа. Кроме того, эти полномочия указываются отдельно для владельца файла, членов группы файла и для всех остальных.

Пример

Представив 3 правила (rwx) для трех групп (владелец, группа, остальные) запись прав доступа будет выглядеть вот так: **rwx rwx rwx** (то есть владельцу разрешено чтение, выполнение и запись, группе разрешено то же самое и остальным).

```
drwxr-xr-x
```

```
|||||||
```

```
|||||||+--исполнение для всех остальных - разрешено
```

```
|||||||+--запись для всех остальных - НЕ разрешено
```

```
|||||||+---чтение для всех остальных - разрешено
```

```
|||||+----исполнение для группы владельца - разрешено
```

```
|||||+-----запись для группы владельца - НЕ разрешено
```

```
|||+-----чтение для группы владельца - разрешено
```

```
||+-----исполнение для владельца - разрешено
```

```
|+-----запись для владельца - разрешено
```

```
+-----чтение для владельца - разрешено
```

```
+-----тип файла
```

Числовое представление

Кроме указанного представления полномочий доступа (символьного), существует так же и числовое представление. Для общего понимания, приведу таблицу соответствия числового (двоичного и десятичного) значения прав доступа и буквенного:

	владелец	группа	остальные
буквенное	rwX	r-x	r--
числовое	421	401	400
итоговое	7	5	4

Особенности прав доступа для каталогов

Права доступа для каталогов немного отличаются. Это в первую очередь связано с тем, что система трактует операции чтения и записи для каталогов отлично от остальных файлов.

Право чтения каталога позволяет получить имена (и только имена) файлов, находящихся в данном каталоге. Чтобы получить дополнительную информацию о файлах каталога (например, подробный листинг команды `ls -l`), системе необходимо просмотреть метаданные файлов, что требует права на выполнения для каталога. Право на *выполнение* также потребуется для каталога, в который Вы захотите перейти (т.е. сделать его текущим).

Итого, право чтения на каталог, позволяет читать имя содержимого, право выполнения - чтение содержимого с метаданными (правами и др.)

Управление правами доступа (chmod)

`chmod [к_какой_группе][что_сделать_с_правами][какие_права] над_чем`
ИЛИ
`chmod [права] над_чем`

к_какой_группе может быть

- u (от user) - владелец-пользователь,
- g (от group) - владелец-группа,
- o (от other) - остальные пользователи,
- a (от all) - все вышеперечисленные группы вместе

что_сделать_с_правами:

- + - добавить,
- - -убрать,
- = - присвоить указанное

какие_права: r- чтение, w - запись, x - выполнение

над_чем: имя или путь к файлу

права: числовое обозначение прав доступа (755, 644 и т.п.)

Пример

```
[Print-server]$ chmod a=rw file  
[Print-server]$ ls -l file  
-rw-rw-rw- 1 user users 78 Nov 20 file
```

это будет аналогично:

```
[Print-server]$ chmod ugo=rw file1  
[Print-server]$ ls -l file1  
-rw-rw-rw- 1 user users 78 Nov 20 file1
```

а так же:

```
[Print-server]$ chmod 666 file2  
[Print-server]$ ls -l file2  
-rw-rw-rw- 1 user users 78 Nov 20 file2
```

Смена владельца и группы

`chown user:group file`

Например

`chown antonk:www-data /apache`

Дополнительные атрибуты файлов

В Linux кроме прав чтения, выполнения и записи, есть еще 3 дополнительных атрибута:

Sticky bit. Используется для каталогов, чтобы защитить в них файлы. В такой каталог может писать ЛЮБОЙ пользователь, но удалить может только те файлы, владельцем которых он является.

Примером может служить директория `/tmp`, в которой запись открыта для всех пользователей, но нежелательно удаление чужих файлов.

Дополнительные атрибуты файлов

2. SUID (Set User ID). Атрибут исполняемого файла, позволяющий запустить его с правами владельца.

В Unix-подобных системах приложение запускается с правами пользователя, запустившего указанное приложение. Это обеспечивает дополнительную безопасность т.к. процесс с правами пользователя не сможет получить доступ на запись к важным системным файлам.

Если на исполняемый файл установлен бит `suid`, то при выполнении эта программа автоматически меняет "эффективный `userID`" на идентификатор того юзера, который является владельцем этого файла. То есть, независимо от того - кто запускает эту программу, она при выполнении имеет права хозяина этого файла.

3. SGID (он же Set Group ID)

Аналогичен SUID, но относится к группе.

Как узнать доп. атрибуты

ls -l file

rwSrwsrwt

где **S** - SUID, **s** - SGID, **t** - Sticky

установка:

chmod 666+t file

Права доступа к символическим ссылкам

Если посмотреть на права символических ссылок, то они всегда выглядят так: `rwxrwxrwx`.

Дело в том, что права на символическую ссылку не имеют особого значения. При использовании ссылки драйвер файловой системы пересчитывает реальный путь к файлу и применяет права доступа, определенные для реального пути уже без учета символической ссылки.