Часть 5.
Файловая система и права доступа. Пользователи, группы.

Влад 'mend0za' Шахов
Linux & Embedded Team Leader

Linux & Embedded Department

Владельцы файлов.Определения

В UNIX (и Linux) любой файл имеет двух владельцев:

1. владельца-пользователя
2. владельца-группу.

```
$ ls -l
total 12
drwxr-xr-x 2 user user 4096 Jan 14 14:59 bin
-rw-r--r-- 1 user user  124 Jan  9 16:55 err
drwx------ 2 user user 4096 Jan  3 22:19 Mail
```

[1] primary (eng.)

Владельцы файлов.Определения

В UNIX (и Linux) любой файл имеет двух владельцев:

1. владельца-пользователя
2. владельца-группу.

```
$ ls -l
total 12
drwxr-xr-x 2 user user 4096 Jan 14 14:59 bin
-rw-r--r-- 1 user user  124 Jan  9 16:55 err
drwx------ 2 user user 4096 Jan  3 22:19 Mail
```

- Группой - определенный список пользователей системы.
- Пользователь может быть членом нескольких групп.
- Одна группа пользователя является первичной[1], а остальные - дополнительными.

Замечание: владелец-пользователь не обязан входить в группу-владельца.

```
~$ id user
uid=1000(user) gid=1000(user) groups=1000(user),4(adm),44(video),46(plugdev)
```

[1] primary (eng.)

Владельцы файла. Внутреннее представление

- Владелец-пользователь и владелец-группа файла определяются по идентификаторам UID[2] и GID[3], а не по именам.

```
~$ ls -l -n /var/cache
drwxr-xr-x 5   0   0 4096 Dec 28 18:50 apt
drwxrwsr-x 2 100 101 4096 Dec 28 17:24 libuuid
drwx------ 3 104 106 4096 Jan 29 20:45 mysql
drwxr-xr-x 5 102   0 4096 Dec 28 17:29 nfs
drwxr-xr-x 3 105 108 4096 Dec 28 17:36 postgresql
drwxr-xr-x 3   0   0 4096 Dec 28 17:24 vim
```

[2]UID - User IDentifier
[3]GID - Group IDentifier

Владельцы файла. Внутреннее представление

- Владелец-пользователь и владелец-группа файла определяются по идентификаторам UID[2] и GID[3], а не по именам.

```
~$ ls -l -n /var/cache
drwxr-xr-x 5   0   0 4096 Dec 28 18:50 apt
drwxrwsr-x 2 100 101 4096 Dec 28 17:24 libuuid
drwx------ 3 104 106 4096 Jan 29 20:45 mysql
drwxr-xr-x 5 102   0 4096 Dec 28 17:29 nfs
drwxr-xr-x 3 105 108 4096 Dec 28 17:36 postgresql
drwxr-xr-x 3   0   0 4096 Dec 28 17:24 vim
```

- Зарегистрированные пользователи и группы определены в файлах /etc/passwd и /etc/group
  Следствие: могут существовать файлы, принадлежащие не зарегистрированным в системе пользователям и группам.

[2] UID - User IDentifier
[3] GID - Group IDentifier

Владельцы файла. Внутреннее представление

- Владелец-пользователь и владелец-группа файла определяются по идентификаторам UID[2] и GID[3], а не по именам.

```
~$ ls -l -n /var/cache
drwxr-xr-x 5   0   0 4096 Dec 28 18:50 apt
drwxrwsr-x 2 100 101 4096 Dec 28 17:24 libuuid
drwx------ 3 104 106 4096 Jan 29 20:45 mysql
drwxr-xr-x 5 102   0 4096 Dec 28 17:29 nfs
drwxr-xr-x 3 105 108 4096 Dec 28 17:36 postgresql
drwxr-xr-x 3   0   0 4096 Dec 28 17:24 vim
```

- Зарегистрированные пользователи и группы определены в файлах /etc/passwd и /etc/group
  Следствие: могут существовать файлы, принадлежащие не зарегистрированным в системе пользователям и группам.
- Новые файлы создаются с владельцем-пользователем, запустившим команду создания и его первичной группой, как владельцем группой.

[2] UID - User IDentifier
[3] GID - Group IDentifier

Владельцы файла. Внутреннее представление

- Владелец-пользователь и владелец-группа файла определяются по идентификаторам UID[2] и GID[3], а не по именам.

```
~$ ls -l -n /var/cache
drwxr-xr-x 5   0   0 4096 Dec 28 18:50 apt
drwxrwsr-x 2 100 101 4096 Dec 28 17:24 libuuid
drwx------ 3 104 106 4096 Jan 29 20:45 mysql
drwxr-xr-x 5 102   0 4096 Dec 28 17:29 nfs
drwxr-xr-x 3 105 108 4096 Dec 28 17:36 postgresql
drwxr-xr-x 3   0   0 4096 Dec 28 17:24 vim
```

- Зарегистрированные пользователи и группы определены в файлах /etc/passwd и /etc/group
  Следствие: могут существовать файлы, принадлежащие не зарегистрированным в системе пользователям и группам.
- Новые файлы создаются с владельцем-пользователем, запустившим команду создания и его первичной группой, как владельцем группой.

[2] UID - User IDentifier
[3] GID - Group IDentifier

Управление владельцами

Команды изменения владельцев

- chown : смена владельца-пользователя
  Пример: `chown sys something.doc`
- chgrp : смена владельца-группы
  Пример: `chgrp adm file.txt`

[4] На пользователя с UID=0 не распространяются ограничения прав доступа

Управление владельцами

Команды изменения владельцев

- chown : смена владельца-пользователя
  Пример: `chown sys something.doc`
- chgrp : смена владельца-группы
  Пример: `chgrp adm file.txt`

Кто может сменить владельцев?

- владельца-пользователя - текущий владелец
- владельца-группу может владелец-пользователь для группы, к которой он сам принадлежит
- администратор (root, UID=0)[4]

[4] На пользователя с UID=0 не распространяются ограничения прав доступа

Практика: Управление владельцами файла

Упражнение 1. Узнать имена своих групп, первичную группу.
Упражнение 2. В папке /tmp создать файл с произвольным именем. Сменить группу-владельца файла на другую (из списка своих групп). Проверить доступность файла на редактирование и удаление.
Упражнение 3. В папке /tmp создать файл с произвольным именем. Сменить пользователя-владельца файла на другого (одного из определённых в системе). Проверить доступность файла на редактирование и удаление.
Упражнение 4. В папке /tmp создать каталог с произвольным именем. Скопировать в него несколько файлов из своего домашнего каталога. Рекурсивно сменить группу-владельца (см Упражнение 2) и пользователя-владельца (см Упражнение 3). Проверить доступ к созданным файлам и каталогам.

Введение в права доступа

У каждого файла присвоены атрибуты, называемые правами доступа.

Проверяются при каждом обращении к любому файлу с любой операцией (чтение, запись, выполнение).

Введение в права доступа

У каждого файла присвоены атрибуты, называемые правами доступа.

Проверяются при каждом обращении к любому файлу с любой операцией (чтение, запись, выполнение).

Классы доступа и права доступа

В UNIX три базовых типа (класса) доступа:

1. u (user) для владельца-пользователя
2. g (group) для владельца-группы
3. o (other) для всех остальных

Классы доступа и права доступа

В UNIX три базовых типа (класса) доступа:

1. u (user) для владельца-пользователя
2. g (group) для владельца-группы
3. o (other) для всех остальных
4. a (all) - объединяет 3 предыдущих класса. Для всех классов пользователей

Классы доступа и права доступа

В UNIX три базовых типа (класса) доступа:

1. u (user) для владельца-пользователя
2. g (group) для владельца-группы
3. o (other) для всех остальных
4. a (all) - объединяет 3 предыдущих класса. Для всех классов пользователей

.

Три основных права доступа для каждого из классов:

1. r (read) право на чтение
2. w (write) право на запись
3. x (execute) право на выполнение

Разбор прав доступа в выводе ls -l

Вывод команды ls -l содержит информацию о правах доступа:

```
~$ ls -l
-rwxrw-r--   1   stud1   users   ...   f1
0 1 2 3 4 5 6 7 8 9
```

Позиции:

- 0 - тип файла: - обычный;
- 1-3 - (u) права доступа для владельца-пользователя.
- 4-6 - (g) права доступа для владельца-группы.
- 7-9 - (o) права доступа для остальных.

Значение прав доступа. Файлы и ссылки

Обычные файлы

- чтение (r) надо, чтобы прочитать файл [a]
- запись (w), чтобы файл изменить
- выполнение (x), чтобы запустить программу или скрипт.

---

[a] Для успешного запуска скрипта необходимо установить атрибут r, чтобы командный интерпретатор мог построчно считывать текст скрипта.

Значение прав доступа. Файлы и ссылки

Обычные файлы

- чтение (r) надо, чтобы прочитать файл [a]
- запись (w), чтобы файл изменить
- выполнение (x), чтобы запустить программу или скрипт.

[a] Для успешного запуска скрипта необходимо установить атрибут r, чтобы командный интерпретатор мог построчно считывать текст скрипта.

Символические ссылки

Права символических ссылок совпадают с файлом, на который она указывает. На самой ссылке стоит 'всем всё разрешено'.

Значение прав доступа. Каталоги

Каталоги

- (r) позволяет получить имена (и только имена) файлов в нём[a].
- (w) TODO
- (x) позволяет "выполнить" каталог
  то есть заглянуть в метаданные и получить полную информацию о каталоге и файлах в нём[b].

---

[a] ls dir
[b] ls -l dir