Пользователь и группа. Команды chown и chgrp. Права доступа к файлу

Мы уже говорили ранее, что для входа в операционную систему UNIX каждый пользователь должен быть зарегистрирован в ней под определенным именем. Вычислительные системы не умеют оперировать именами, поэтому каждому имени пользователя в системе соответствует некоторое числовое значение - его идентификатор - UID (User IDentificator).

Все пользователи в системе делятся на группы пользователей. Например, все студенты одной учебной группы могут составлять свою собственную группу пользователей. Группы пользователей также получают свои имена и соответствующие идентификационные номера - GID (Group IDentificator). В некоторых версиях UNIX каждый пользователь может входить ровно в одну группу, в некоторых - в несколько разных групп. В современных системах Linux пользователь имеет первичную группу, имя которой совпадает с именем пользователя и может входить в несколько дополнительных групп.

Группы используются не только для администрирования, но и для управления правами доступа для различных программ. Список групп можно посмотреть в файле /etc/group, например при помощи команды

```
$ cat /etc/group
```

Для каждого файла, созданного в файловой системе запоминаются имена его хозяина и группы хозяев. Заметим, что группа хозяев не обязательно должна быть группой, в которую входит хозяин. В операционной системе Linux при создании файла его хозяином становится пользователь, создавший файл, а его группой хозяев - первичная группа, к которой он принадлежит. Впоследствии хозяин файла или системный администратор может передать его в собственность другому пользователю или изменить его группу хозяев с помощью команд chown и chgrp, описание которых можно найти в UNIX Manual.

Как видим, для каждого файла выделяется три категории пользователей:

- Пользователь, являющийся хозяином файла.
- Пользователи, относящиеся к группе хозяев файла.
- Все остальные пользователи.

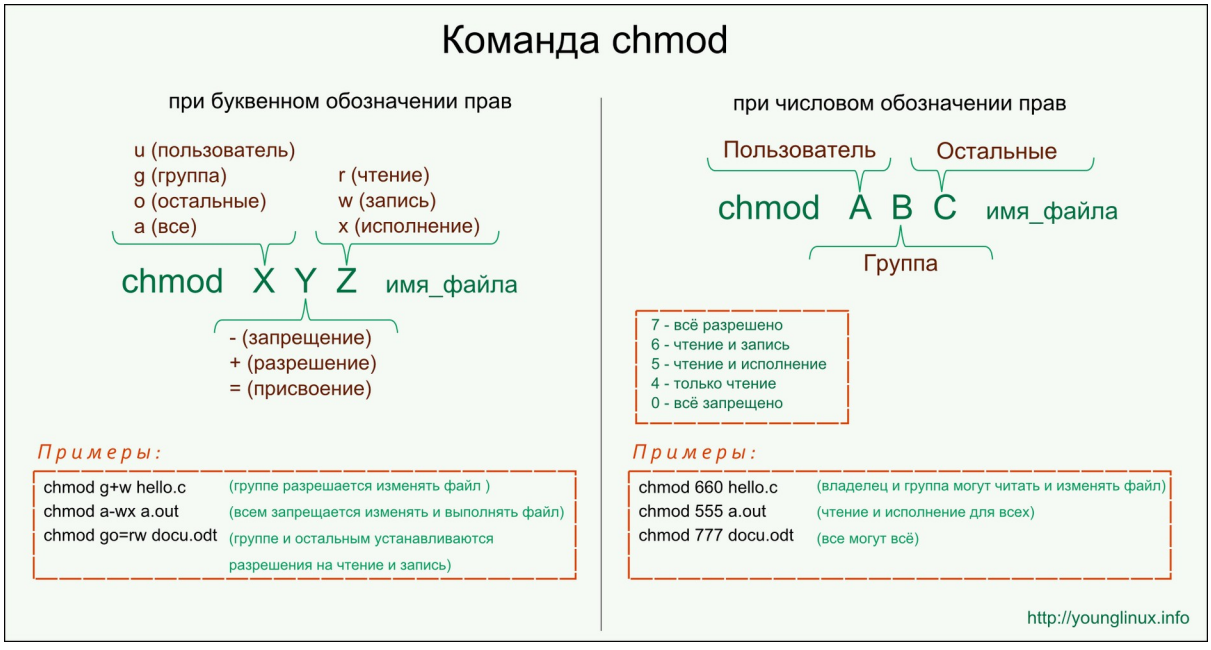
Для каждой из этих категорий пользователей хозяин файла может определить различные права доступа к файлу. Различают три вида прав доступа: право на чтение файла - r (от слова read), право на модификацию файла - w (от слова write) и право на исполнение файла - x (от слова execute). Для регулярных файлов смысл этих прав совпадает с указанным выше. Для директорий он несколько меняется. Право чтения для каталогов позволяет читать имена файлов, находящихся в этом каталоге (и только имена). Поскольку "исполнять" директорию бессмысленно (как, впрочем, и не исполняемый регулярный файл) право доступа на исполнение для директорий меняет смысл: наличие этого права позволяет получить дополнительную информацию о файлах, входящих в каталог: их размер, кто их хозяин, дата создания и т.д. Право на исполнение также требуется для директории, чтобы сделать ее текущей, а также для всех директорий по пути к указанной. Право записи для директории позволяет изменять ее

содержимое: создавать и удалять в ней файлы, переименовывать их. Отметим, что для удаления файла достаточно иметь право записи для директории, в которую непосредственно входит данный файл, независимо от прав доступа к самому файлу.

Команда ls с опциями -al. Использование команды chmod

Посмотреть подробную информацию о файлах в некоторой директории, включая имена хозяина, группы хозяев и права доступа, можно с помощью уже известной нам команды ls с опциями - al. В выдаче этой команды третья колонка слева содержит имена пользователей хозяев файлов, а четвертая колонка слева - имена групп хозяев файла. Самая левая колонка содержит типы файлов и права доступа к ним. Тип файла определяет первый символ в наборе символов. Если это символ 'd' - то тип файла - директория, если там стоит символ '-', то это - регулярный файл. Следующие три символа определяют права доступа для хозяина файла, следующие три - для пользователей, входящих в группу хозяев файла, и последние три - для всех остальных пользователей. Наличие символа (r, w или x), соответствующего праву, для некоторой категории пользователей означает, что данная категория пользователей обладает этим правом.

Вызовите команду ls -al для своей домашней директории и проанализируйте ее выдачу. Хозяин файла может изменять права доступа к нему, пользуясь командой chmod. Также можете заглянуть в директорию /dev, которая заполнена специальными файлами устройств, которые не являются регулярными файлами, находящимися на диске, а по сути, являются указателями на драйверы устройств. Здесь вы можете увидеть типы символьных и блочных файлов устройств (c и b соответственно), а также более необычные группы хозяев этих файлов. Менять в этой директории, разумеется, ничего не надо.



Создайте новый файл и посмотрите на права доступа к нему, установленные системой при его создании. Чем руководствуется операционная система при назначении этих прав? Она использует для этого маску создания файлов для программы, которая файл создает. Изначально для программы-оболочки она имеет некоторое значение по умолчанию. Установление значения какого-либо бита маски равным 1 запрещает инициализацию соответствующего права доступа для вновь создаваемого файла.

Значение маски создания файлов может изменяться с помощью системного вызова umask() или команды umask. На ближайших семинарах мы рассмотрим механизм наследования маски процессами.

Системные вызовы getuid и getgid

Узнать идентификатор пользователя, запустившего программу на исполнение, - UID и идентификатор группы, к которой он относится, - GID можно с помощью системных вызовов getuid() и getgid(), применив их внутри этой программы. Ознакомиться с этими вызовами подробнее можно при помощи команды man.

Компиляция программ на языке C в UNIX и запуск их на счёт

Теперь вы практически созрели для того, чтобы написать первую программу в нашем курсе. Осталось только научиться компилировать программы на языке C и запускать их на счёт. Для компиляции программ в Linux мы будем применять компилятор gcc. Для того, чтобы он нормально работал, необходимо, чтобы исходные файлы, содержащие текст программы, имели имена, заканчивающиеся на .c. В простейшем случае откомпилировать программу можно, запустив компилятор командой

```
$ gcc имя_исходного_файла
```

Если программа была написана без ошибок, то компилятор создаст исполняемый файл с именем a.out. Изменить имя создаваемого исполняемого файла можно, задав его с помощью опции -o

```
$ gcc имя_исходного_файла -o имя_исполняемого_файла
```

Компилятор gcc имеет несколько сотен возможных опций. Получить информацию о них вы можете в UNIX Manual.

Запустить программу на исполнение можно, набрав полное или каноническое относительное имя исполняемого файла и нажав клавишу <Enter>.

Задачи на семинар

Задача 1 (1 балл):

Напишите, откомпилируйте и запустите программу, которая печатала бы идентификатор пользователя, запустившего программу, и идентификатор его группы.

Задача 2 (1 балл):

Для вычисления значения квадратного корня из неотрицательного действительного числа часто используется итерационная формула Герона.

Допустим, мы хотим извлечь корень квадратный из числа $a \geq 0$. *Берем произвольное положительное начальное приближение* x_0. *Следующее приближение рассчитываем по формуле*

$$x_{n+1}=\frac{1}{2}\left(x_n+\frac{a}{x_n}\right).$$

Этот процесс продолжается до тех пор, пока не выполнится условие

$$\left|x_{n+1}-x_n\right|<\varepsilon,$$

где ε — небольшое положительное число.

Приняв $\varepsilon=10^{-6}$*, напишите программу извлекающую корень квадратный из произвольного неотрицательного действительного числа, которое программа запрашивает с клавиатуры. Программа должна проверять, что вводится действительно неотрицательное число.*