Механизмы безопасной работы с файлами

Стандартные права доступа к файлам

В данной главе содержится подробная информация о системе разграничения доступа к файлам на основе механизма **владения файлами** и **прав доступа к файлам**.

Механизм владения файлами

Пользователь и группа, владеющие файлом

Пользователи и **группы пользователей** в рамках системы могут быть описаны в локальных файлах **/etc/passwd** и **/etc/group** или объявлены на уровне сервера NIS, LDAP или домена Samba. Эти пользователи и группы пользователей могут владеть файлами. На самом деле каждым файлом владеет как **пользователь**, так и **группа пользователей**, что можно увидеть в следующем примере.

```
paul@rhel65:~/owners$ ls -lh
итого 636K
-rw-r--r--. 1 paul snooker 1.1K апр  8 18:47 data.odt
-rw-r--r--. 1 paul paul    626K апр  8 18:46 file1
-rw-r--r--. 1 paul tennis   185 апр  8 18:46 file2
-rw-rw-r--. 1 root root       0 апр  8 18:47 stuff.txt
paul@rhel65:~/owners$
```

Пользователь `paul` владеет тремя файлами; файлом `file1` **владеет пользователь** `paul` и **группа пользователей** `paul`, файлом `data.odt` владеет **группа пользователей** `snooker`, а файлом `file2` - группа пользователей `tennis`.

Последний файл носит имя `stuff.txt` и его владельцами является пользователь `root` и группа пользователей `root`.

Вывод списка учетных записей пользователей

Вы можете воспользоваться следующей командой для вывода списка всех локальных учетных записей пользователей.

```
paul@debian7~$ cut -d: -f1 /etc/passwd | column
root            ntp             sam             bert
naomi
```

daemon	mysql	tom	rino
matthias2			
bin	paul	wouter	antonio
bram			
sys	maarten	robrecht	simon
fabrice			
sync	kevin	bilal	sven
chimene			
games	yuri	dimitri	wouter2
messagebus			
man	william	ahmed	tarik
roger			
lp	yves	dylan	jan
frank			
mail	kris	robin	ian
toon			
news	hamid	matthias	ivan
rinus			
uucp	vladimir	ben	azeddine
eddy			
proxy	abiy	mike	eric
bram2			
www-data	david	kevin2	kamel
keith			
backup	chahid	kenzo	ischa
jesse			
list	stef	aaron	bart
frederick			
irc	joeri	lorenzo	omer
hans			
gnats	glenn	jens	kurt
dries			
nobody	yannick	ruben	steve
steve2			
libuuid	christof	jelle	
constantin	tomas		
Debian-exim	george	stefaan	sam2
johan			

```
statd           joost           marc            bjorn
tom2
sshd            arno            thomas          ronald
chgrp
```

Утилита chgrp

Вы можете изменить имя группы пользователей, владеющей файлом, с помощью утилиты **chgrp**.

```
root@rhel65:/home/paul/owners# ls -l file2
-rw-r--r--. 1 root tennis 185 апр  8 18:46 file2
root@rhel65:/home/paul/owners# chgrp snooker file2
root@rhel65:/home/paul/owners# ls -l file2
-rw-r--r--. 1 root snooker 185 апр  8 18:46 file2
root@rhel65:/home/paul/owners#
```

Утилита chown

Имя пользователя, владеющего файлом, может быть изменено с помощью утилиты **chown**.

```
root@laika:/home/paul# ls -l FileForPaul
-rw-r--r-- 1 root paul 0 2008-08-06 14:11 FileForPaul
root@laika:/home/paul# chown paul FileForPaul
root@laika:/home/paul# ls -l FileForPaul
-rw-r--r-- 1 paul paul 0 2008-08-06 14:11 FileForPaul
```

Вы также можете использовать утилиту **chown** для одновременного изменения имен пользователя и группы пользователей, владеющих файлом.

```
root@laika:/home/paul# ls -l FileForPaul
-rw-r--r-- 1 paul paul 0 2008-08-06 14:11 FileForPaul
root@laika:/home/paul# chown root:project42 FileForPaul
root@laika:/home/paul# ls -l FileForPaul
-rw-r--r-- 1 root project42 0 2008-08-06 14:11
FileForPaul
```

Список специальных типов файлов

При использовании команды **ls -l** в строке с информацией о каждом из файлов вы можете обнаружить десять символов перед именами владеющих файлом пользователя и группы пользователей. Первый символ сообщает нам о типе файла. Для обычных файлов ис-

пользуется символ **-**, для директорий - символ **d**, символьные ссылки обозначаются с помощью символа **l**, именованные каналы - с помощью символа **p**, символьные устройства - с помощью символа **c**, блочные устройства - с помощью символа **b**, а сокеты - с помощью символа **s**.

Таблица 30.1 - Специальные типы файлов Unix

Первый символ	Тип файла
-	Обычный файл
d	Директория
l	Символьная ссылка
p	Именованный канал
b	Блочное устройство
c	Символьное устройство
s	Сокет

В примере ниже выводится информация о файле символьного устройства (консоли) и блочного устройства (жесткого диска).

```
paul@debian6lt~$ ls -ld /dev/console /dev/sda
crw-------   1 root root  5, 1 мар 15 12:45 /dev/console
brw-rw----   1 root disk  8, 0 мар 15 12:45 /dev/sda
```

А в следующем примере вы можете увидеть информацию о директории, обычном файле и символьной ссылке.

```
paul@debian6lt~$ ls -ld /etc /etc/hosts /etc/motd
drwxr-xr-x 128 root root 12288 мар 15 18:34 /etc
-rw-r--r--   1 root root   372 дек 10 17:36 /etc/hosts
lrwxrwxrwx   1 root root    13 дек  5 10:36 /etc/motd
-> /var/run/motd
```

Права доступа

Символы rwx

Девять символов, следующих после символа типа файла, отображают права доступа к файлу, разделенные на три триплета. Символом права доступа может быть символ **r**, обозначающий возможность чтения данных из файла, символ **w**, обозначающий возможность записи данных в файл, и символ **x**, обозначающий возможность исполнения файла. Вам понадобится право на чтение директории (обозначаемое с помощью символа **r**) для вывода списка содержимого этой директории (например, с помощью команды `ls`). Для входа в ди-

ректорию (например, с помощью команды `cd`) вам понадобится право на исполнение (обозначаемое с помощью символа **`x`**). А для создания новых файлов в директории и удаления существующих файлов из нее вам понадобится право на запись в эту директорию (обозначаемое с помощью символа **`w`**).

Таблица 30.2. Стандартные права доступа к файлам Unix

Право доступа	К файлу	К директории
`r` (чтение)	Чтение содержимого файла (`cat`)	Чтение содержимого директории (`ls`)
`w` (запись)	Изменение содержимого файла (`vi`)	Создание файлов в директории (`touch`)
`x` (исполнение)	Исполнение файла	Вход в директорию (`cd`)

30.2.2. Три набора символов `rwx`

Мы уже знаем о том, что вывод команды **`ls -l`** начинается с десяти символов для каждого из файлов. В примере ниже показан вывод в случае нахождения в директории обычного файла (так как первым символом является символ `-`).

```
paul@RHELv4u4:~/test$ ls -l proc42.bash
-rwxr-xr--  1 paul proj 984 фев  6 12:01 proc42.bash
```

В таблице ниже описано назначение всех десяти символов.

Таблица 30.3. Права доступа к файлам в Unix

Позиция	Символы	Назначение
1	`-`	Указание на то, что это обычный файл.
2-4	`rwx`	Права доступа для пользователя, владеющего файлом.
5-7	`r-x`	Права доступа для пользователей из группы, владеющей файлом.
8-10	`r--`	Права доступа для остальных пользователей.

В том случае, если вы являетесь **владельцем файла**, ваш доступ к содержимому этого файла будет регламентироваться **правами доступа для пользователя, владеющего файлом**. Остальные права доступа не будут оказывать ровным счетом никакого влияния на вашу возможность доступа к содержимому этого файла.

В том случае, если вы состоите в **группе пользователей**, владеющей данным файлом, ваш доступ к содержимому этого файла бу-

дет регламентироваться **правами доступа для пользователей из группы, владеющей файлом**. По аналогии, остальные права доступа не будут оказывать никакого влияния на вашу возможность доступа к содержимому этого файла.

В том же случае, если вы не являетесь **владельцем этого файла**, а также не состоите в **группе пользователей, владеющей этим файлом**, ваш доступ к содержимому файла будет регламентироваться правами доступа для остальных пользователей. Как и раньше, остальные права доступа не будут оказывать никакого влияния на вашу возможность доступа к содержимому этого файла.

Примеры прав доступа к файлам

В примере ниже показаны некоторые комбинации прав доступа к фалам и директориям. Имена файлов и директорий описывают права доступа к ним.

```
paul@laika:~/perms$ ls -lh
итого 12K
drwxr-xr-x 2 paul paul 4.0K 2007-02-07 22:26 Все_пользо-
ватели_могут_осуществлять_вход_Владелец_-
_создавать_и_удалять_файлы
-rwxrwxrwx 1 paul paul    0 2007-02-07 22:21 Все_пользо-
ватели_имеют_полный_контроль_над_файлом.txt
-r--r----- 1 paul paul    0 2007-02-07 22:21 Только_вла-
дельцы_могут_осуществлять_чтение.txt
-rwxrwx--- 1 paul paul    0 2007-02-07 22:21
Владельцы_могут_делать_все_Остальные_-_ничего.txt
dr-xr-x--- 2 paul paul 4.0K 2007-02-07 22:25
Владелец_и_участники_группы_могут_осуществлять_вход
dr-x------ 2 paul paul 4.0K 2007-02-07 22:25 Исключитель-
но_владелец_может_осуществлять_вход
paul@laika:~/perms$
```

Подводя итог, можно сказать, что первый триплет **rwx** представляет права доступа **владельца файла**. Второй триплет соответствует правам доступа **пользователей из группы, владеющей файлом**; он указывает права доступа каждого пользователя из этой группы. Третий триплет описывает права доступа **всех остальных пользователей**, не являющихся владельцами файла или директории и не состоящих в группе пользователей, владеющей файлом или директорией.

Установка прав доступа (chmod)

Права доступа могут быть изменены с помощью утилиты **chmod**. В первом примере владельцу файла permissions.txt дается право на его исполнение.

```
paul@laika:~/perms$ ls -l permissions.txt
-rw-r--r-- 1 paul paul 0 2007-02-07 22:34 permissions.txt
paul@laika:~/perms$ chmod u+x permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwxr--r-- 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

В следующем примере у группы пользователей, владеющей файлом, изымается право на чтение этого файла.

```
paul@laika:~/perms$ chmod g-r permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwx---r-- 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

А в следующем примере право на чтение файла изымается у всех пользователей, не являющихся владельцами этого файла и не входящих в группу, владеющую этим файлом.

```
paul@laika:~/perms$ chmod o-r permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwx------ 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

В примере ниже всем пользователям дается право модификации содержимого файла.

```
paul@laika:~/perms$ chmod a+w permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwx-w--w- 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

В подобных случаях от вас даже не требуется использовать символ a.

```
paul@laika:~/perms$ chmod +x permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwx-wx-wx 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

Кроме того, вы можете задавать права доступа явным образом.

```
paul@laika:~/perms$ chmod u=rw permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rw--wx-wx 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

Не стесняйтесь использовать любые комбинации прав доступа.

```
paul@laika:~/perms$ chmod u=rw,g=rw,o=r permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rw-rw-r-- 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

Утилита chmod будет принимать даже сомнительные комбинации прав доступа.

```
paul@laika:~/perms$ chmod u=rwx,ug+rw,o=r permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwxrw-r-- 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

Установка прав доступа в восьмеричном представлении

Большинство администраторов систем Unix наверняка предпочтет использовать **классическое** восьмеричное представление при просмотре и установке прав доступа. Рассмотрим триплет на битовом уровне, причем символу `r` будет соответствовать значение 4, символу `w` - 2, а символу `x` - 1.

Таблица 30.4. Права доступа в восьмеричном представлении

Двоичное представление	Восьмеричное представление	Права доступа
000	0	---
001	1	--x
010	2	-w-
011	3	-wx
100	4	r--
101	5	r-x
110	6	rw-
111	7	rwx

Исходя из вышеописанного, значение 777 является эквивалентным правам доступа `rwxrwxrwx` и, по той же логике, значение 654 соответствует правам доступа `rw-r-xr--`. Утилита **chmod** будет принимать и эти числовые значения.

```
paul@laika:~/perms$ chmod 777 permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwxrwxrwx 1 paul paul 0 2007-02-07 22:34 permissions.txt
paul@laika:~/perms$ chmod 664 permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
```

```
-rw-rw-r-- 1 paul paul 0 2007-02-07 22:34 permissions.txt
paul@laika:~/perms$ chmod 750 permissions.txt
paul@laika:~/perms$ ls -l permissions.txt
-rwxr-x--- 1 paul paul 0 2007-02-07 22:34 permissions.txt
```

Значение umask

При создании файла или директории используется набор стандартных прав доступа. Эти стандартные права доступа устанавливаются на основе значения **umask**. Значение **umask** позволяет задать права доступа, которые вы не хотите устанавливать по умолчанию. Вы можете вывести информацию о наборе стандартных прав доступа, выполнив команду **umask**.

```
[Harry@RHEL4b ~]$ umask
0002
[Harry@RHEL4b ~]$ touch test
[Harry@RHEL4b ~]$ ls -l test
-rw-rw-r--  1 Harry Harry 0 июл 24 06:03 test
[Harry@RHEL4b ~]$
```

Как вы можете увидеть, по умолчанию файл также не делается исполняемым. Это стандартная функция, предназначенная для повышения безопасности систем Unix; создаваемые файлы никогда не делаются исполняемыми по умолчанию. Вам придется самостоятельно выполнить команду **chmod +x** для того, чтобы сделать файл исполняемым. Это также означает, что 1 бит значения **umask** не оказывает какого-либо влияния на права доступа к создаваемым файлам - значение **umask** 0022 эквивалентно значению **umask** 0033.

Команда mkdir -m

При создании директорий с помощью команды **mkdir** вы можете использовать параметр **-m** для задания **прав доступа**. Методика использования данного параметра освещена в примере ниже.

```
paul@debian5~$ mkdir -m 700 MyDir
paul@debian5~$ mkdir -m 777 Public
paul@debian5~$ ls -dl MyDir/ Public/
drwx------ 2 paul paul 4096 2011-10-16 19:16 MyDir/
drwxrwxrwx 2 paul paul 4096 2011-10-16 19:16 Public/
```

Команда `cp -p`

Для сохранения прав доступа и меток времени файлов при их копировании следует использовать команду **`cp -p`**.

```
paul@laika:~/perms$ cp file* cp
paul@laika:~/perms$ cp -p file* cpp
paul@laika:~/perms$ ll *
-rwx------ 1 paul paul    0 2008-08-25 13:26 file33
-rwxr-x--- 1 paul paul    0 2008-08-25 13:26 file42

cp:
total 0
-rwx------ 1 paul paul 0 2008-08-25 13:34 file33
-rwxr-x--- 1 paul paul 0 2008-08-25 13:34 file42

cpp:
total 0
-rwx------ 1 paul paul 0 2008-08-25 13:26 file33
-rwxr-x--- 1 paul paul 0 2008-08-25 13:26 file42
```