

Sistemas Operativos

António Godinho

Permissões de ficheiros/diretórios | Owners

Gestão de grupos e utilizadores

- NOTA: os comandos descritos são executados como root. Como tal é necessário estar autenticado como root ou colocar o prefixo sudo.
- Na aula teórica será falado sobre a estrutura de ficheiros do Linux e a localização dos seus ficheiros de configuração.
- No entanto, para se perceber a gestão de grupos e utilizadores é importante focar os seguintes ficheiros de sistema:
 - /etc/passwd
 - /etc/shadow
 - /etc/group
 - /etc/sudoers
 - /etc/gshadow
 - /etc/login.defs

Introdução

- Os Sistemas Operativos *Linux* são sistemas multiutilizador
- Como tal estes, sistemas estão dotados de mecanismos que restringem os acesso aos ficheiros e diretórios
- As restrições podem ser configuradas para os seguintes conjuntos:
 - Utilizador – *User*
 - Grupo – *Group*
 - Outros – *Others*

Verificar permissões

- Para listar o conteúdo de um diretório e verificar as permissões existentes: `$ ls -la`

```
root@merlin:~# ls -la
total 220
drwx----- 18 root root 4096 Dec 14 21:10 .
drwxr-xr-x 35 root root 4096 Dec 24 15:39 ..
-rw----- 1 root root 11396 Dec 23 18:37 .bash_history
-rw-r--r-- 1 root root 562 Jan 25 2017 .bashrc
drwxr-xr-x 3 root root 4096 Dec 4 15:24 bpi-tools
-rw-r--r-- 1 root root 957 Dec 4 15:25 .bpi-tools.lst
drwxr-xr-x 4 root root 4096 May 15 2016 .cache
drwxr-xr-x 6 root root 4096 May 15 2016 .config
drwx----- 3 root root 4096 May 15 2016 .dbus
-rw-r--r-- 1 root root 4662 Mar 28 2018 default
drwxr-xr-x 2 root root 4096 May 15 2016 Desktop
-rw-r--r-- 1 root root 55 May 15 2016 .dmrc
-rw----- 1 root root 84 Jul 24 2017 .dropbox_uploader
-rwxr-xr-x 1 root root 48451 Jul 24 2017 dropbox_uploader.sh
```

Entender as permissões

- O sistemas de permissões tal com referido anteriormente está dividido em: *user*, *group* e *others*
- As permissões são apresentadas da seguinte forma:
 - r – *read* (representa leitura)
 - w – *write* (representa escrita)
 - x – *execute* (representa execução)
 - - opção desativada

```
root@merlin:~# ls -la
total 220
drwx----- 18 root root  4096 Dec 14 21:10 .
drwxr-xr-x  35 root root  4096 Dec 24 15:39 ..
-rw-----  1 root root 11396 Dec 23 18:37 .bash_history
-rw-r--r--  1 root root   562 Jan 25  2017 .bashrc
drwxr-xr-x  3 root root  4096 Dec  4 15:24 bpi-tools
-rw-r--r--  1 root root   957 Dec  4 15:25 .bpi-tools.lst
drwxr-xr-x  4 root root  4096 May 15  2016 .cache
drwxr-xr-x  6 root root  4096 May 15  2016 .config
```

- O primeiro caracter representa: (d) diretório; (-) arquivo; (s) socket; (p) *FIFO*; (L) link simbólico

Entender as permissões

- A seguinte tabela apresenta as permissões mais utilizadas:

Permissões	Significado
r - -	Permissão de leitura
r - x	Permissão de leitura e execução
r w -	Permissão de leitura e escrita
r w x	Permissão de leitura, escrita e execução
- - -	Sem permissões

Atribuir ou retirar permissões

- As permissões podem ser alteradas de duas formas:
 - Numérica
 - Simbolicamente
- Numérica:
 - `$ chmod 644` – Permissões de leitura e escrita (utilizador), permissões de leitura para o grupo e outros
 - `$ chmod 700` – permissões de leitura, escrita e execução apenas para o utilizador.
- Simbolicamente:
 - `$ chmod u+x` – permissão de execução ao utilizador
 - `$ chmod g+rw` – permissão de leitura, escrita e execução
 - As opções (+) (-) (=) permitem: adicionar, remover e atribuir

Combinações numéricas

- As combinações numérica resultam do valor em binário resultante das permissões atribuídas.
- Sendo que o bit a “1” representa ativo e o bit a “0” representa inativo.
- Após a combinação de bits estar definida a mesma é convertida em octal e resulta a permissão a atribuir de forma numérica.
- A tabela seguinte apresenta algumas combinações mais utilizadas:

Binário	Octal	Significado
000	0	Sem permissões
100	4	Permissão de leitura
110	6	Permissão de leitura e escrita
111	7	Permissão de leitura, escrita e execução
101	5	Permissão de leitura e execução

Combinações numéricas

- As permissões são definidas para o utilizador, o grupo e outros, como tal resulta:
 - `$ chmod 644 rsi.txt` `godinho@merlin:~$ chmod 644 rsi.txt`
- Em que:
 - 6 representa as permissões do utilizador
 - 4 representa as permissões do grupo
 - 4 representa as permissões dos outros

u	g	o					
↓	↓	↓					
<code>-rw-r--r-- 1 godinho godinho 0 Dec 24 18:13 rsi.txt</code>							
6	4	4					

Combinações numéricas

User (owner)			Group			Other		
r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1

TABELA AUXILIAR	
Utilizadores	
u	User (owner)
g	Grupo (Group)
o	Outros (other)
a	all (todos)
PERMISSÕES	
r	Leitura (read)
w	Escrita (write)
x	Executar (execute)
Operadores	
+	Adiciona permissão
-	Remove permissão
=	Define permissão, remove as restantes (para u,g,o,a)

Combinações numéricas

TABELA AUXILIAR	
Utilizadores	
u	User (owner)
g	Grupo (Group)
o	Outros (other)
a	all (todos)
PERMISSÕES	
r	Leitura (read)
w	Escrita (write)
x	Executar (execute)
Operadores	
+	Adiciona permissão
-	Remove permissão
=	Define permissão, remove as restantes (para u,g,o,a)

Comando	Permissão
# chmod 777 file	rwxrwxrwx
# chmod 755 file	rwxr-xr-x
# chmod 644 file	rw-r--r-
# chmod 000 file	-----
# chmod a-w files	r-xr-xr-x
# chmod o-x files	rwsrwsrw
# chmod go-rwx files	rwX-----
# chmod u+rw files	rw-----
# chmod a+x files	--X--X--X
# chmod ug+rx files	r-xr-x---

Ownership

- O Linux um sistema multiutilizador, como tal:
 - Os ficheiros e diretorias têm donos
 - Quando um ficheiro é criado, este pertence ao utilizador que o criou, logo este é o seu dono (*owner*)

```
-rw-r--r-- 1 godinho godinho 0 Dec 24 18:13 rsi.txt
```

- É possível alterar o dono de um ficheiro ou diretoria

Ownership – umask

- As permissões por omissão na criação de ficheiros são 0666 e para as diretorias 0777. A maior parte das distribuições de Linux definem outro tipo de permissões através da alteração da umask. Por norma o seu valor é 022.
- Significa que a escrita está bloqueada para o grupo e outros utilizadores.

Ownership – umask

- As máscaras em octal são calculadas através do complemento das máscaras que utilizamos até agora:
 - **Valor octal** : permissões
 - **0** : read, write and execute
 - **1** : read and write
 - **2** : read and execute
 - **3** : read only
 - **4** : write and execute
 - **5** : write only
 - **6** : execute only
 - **7** : no permissions

Ownership – umask

- A definição do umask depende da distribuição de Linux e até dentro de cada distribuição existem sítios diferentes (de versão para versão).
- No debian a umask por omissão é 033
- No Ubuntu 022 (mais uma vez depende da versão)

Ownership – umask

- Como calcular as permissões da umask?
 - Simplesmente subtrair a umask à base das permissões:
 - $777 - 022 = 755$
 - Permissões base: 777
 - umask : 022
 - Subtrair a umask às permissões da diretoria:
 - $(777-022) : 755$ (rwxr-xr-x)

Ownership – umask

- Exemplo:

- \$ umask 077
\$ mkdir dir1
\$ touch file
\$ ls -ld dir1 file

```
drwx----- 2 vivek vivek 4096 2011-03-04 02:05
dir1
-rw----- 1 vivek vivek 0 2011-03-04 02:05
file
```

Ownership – umask

- Exemplo:

- \$ umask 077
\$ mkdir dir1
\$ touch file
\$ ls -ld dir1 file

```
drwx----- 2 vivek vivek 4096 2011-03-04 02:05 dir1  
-rw----- 1 vivek vivek 0 2011-03-04 02:05 file
```

Alterar o dono - *owner*

- Para alterar o dono (owner) utiliza-se o comando `chown`
- Alguns exemplos:
 - `$ chown utilizador ficheiro|diretoria`
 - `$ chown utilizador:grupo ficheiro|diretoria`
 - `$ chown :grupo ficheiro|diretoria` (desta forma o utilizador é omitido)
 - `$ chown utilizador: ficheiro|diretoria` (desta forma o dono passa a ser o utilizador e grupo passa a ser o grupo de login do utilizador)
- É possível utilizar recursividade
 - `$ chown -R utilizador:grupo minhapasta` (desta forma, todos os ficheiros e diretorias existentes dentro da diretoria minhapasta serão afetados)