UNIVERSIDADE
VILA VELHA
ESPÍRITO SANTO

Grupos, Usuários, Poderes e Permissões

Prof. Jean-Rémi Bourguet

Sistemas Operacionais

Identificação de grupos (GID)

- Os **grupos** são **identificados** por o número **GID** (**G**roup **ID**entifier).

addgroup

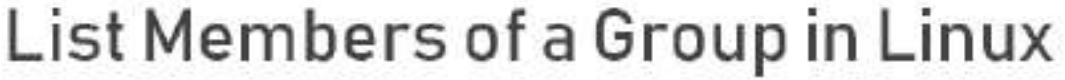
- O comando **addgroup adiciona um grupo** (**criado sem usuários**).

```
root@linux:~# addgroup penguins
Adding group 'penguins' (GID 1001) ...
Done.
```

Arquivo de configuração dos grupos

- O **arquivo `/etc/group`** terá a seguinte linha: `penguins:x:1001:`
- O **caractere x** indica que a **senha do grupo** (em **`/etc/gshadow`**).

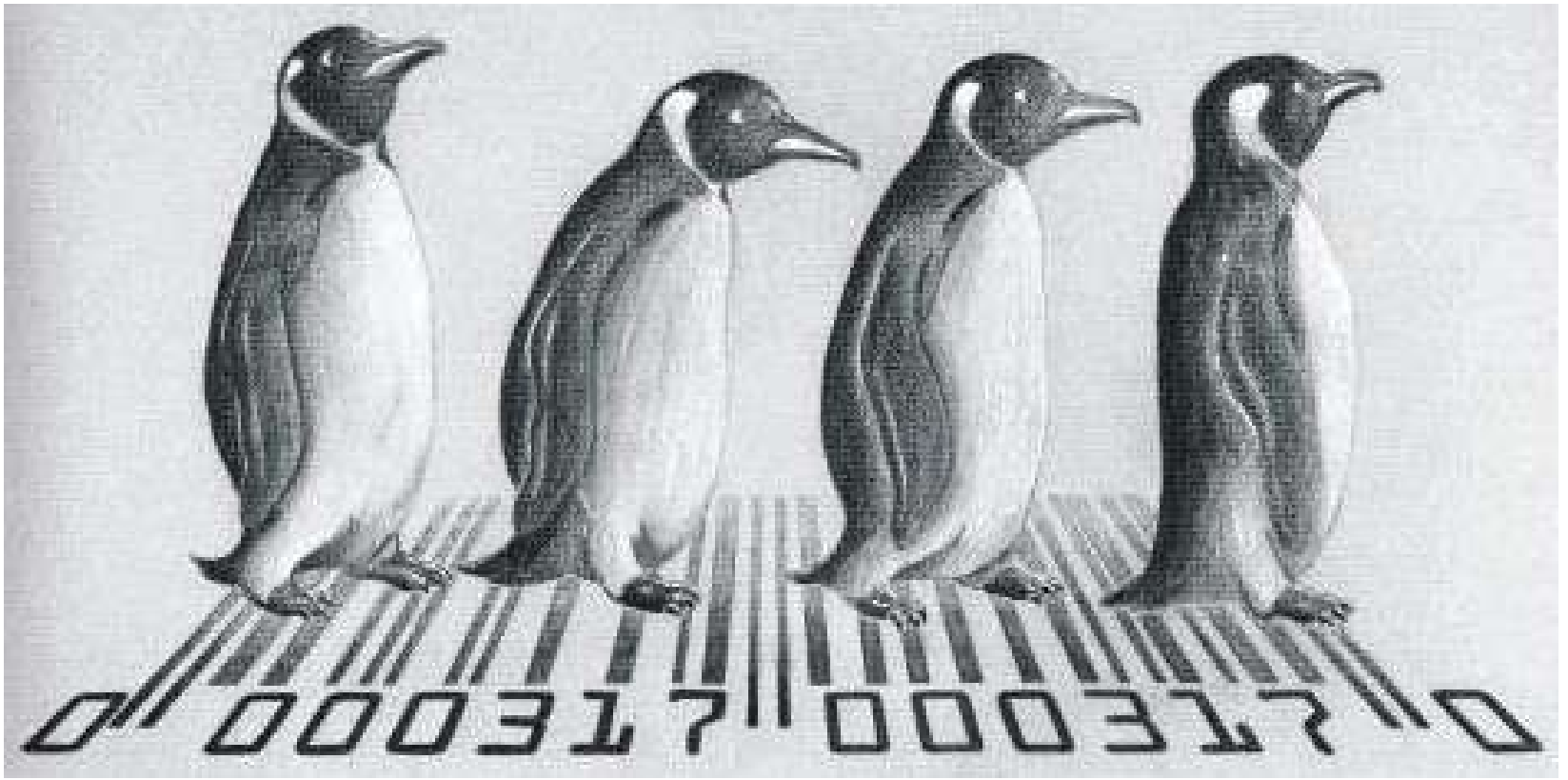
```
tux@linux:~$ grep "penguins" /etc/group
penguins:x:1001:
```



LinOxide

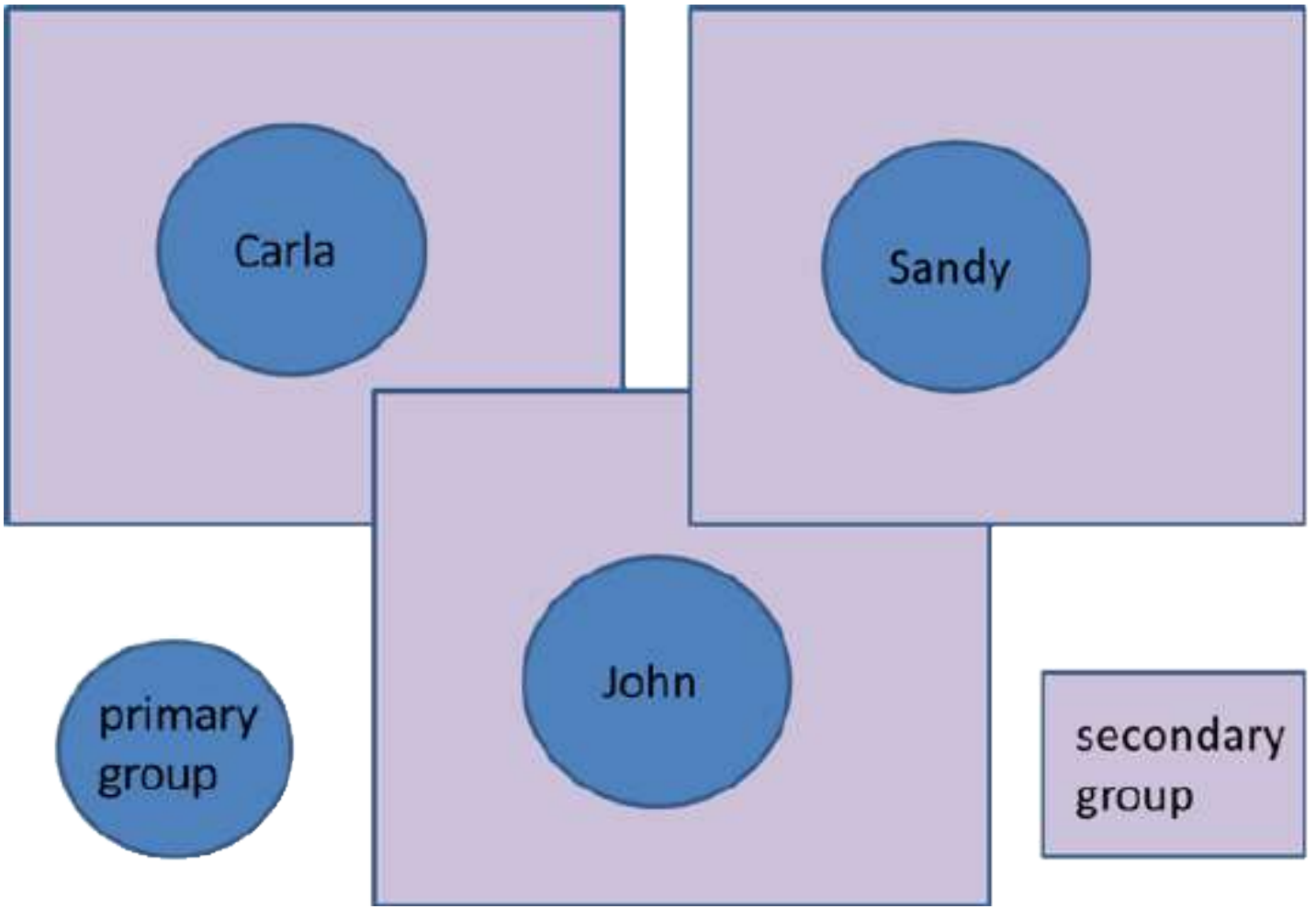
groups

```
tux@linux:~$ groups
tux adm cdrom sudo dip plugdev lpadmin lxd sambash
tux@linux:~$ groups root
root : root
```



Grupos primários

- Cada user possui um **grupo primário** onde se encontra **por padrão**.



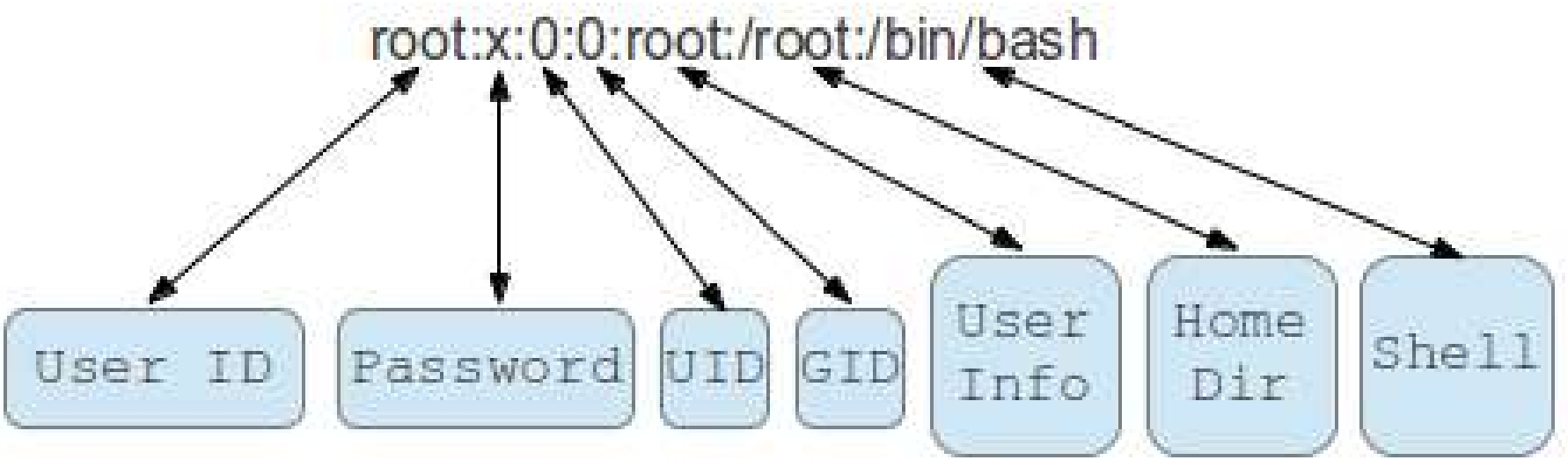
Identificação de usuário (UID)

- Os **usuários** são **identificado** por um **número UID** (**U**ser **ID**entifier).
- 0 (root) a 99: **Kernel**. 100 a 999: **Admins**. 1000 a 59999: **Regular Users**.

Identificação de usuário (UID)

- Vamos **procurar** seu **usuário** dentro do **arquivo `/etc/passwd`**:

```
# grep tux /etc/passwd
tux:x:1000:1000:,,,:/home/tux:/bin/bash
```



O arquivo shadow

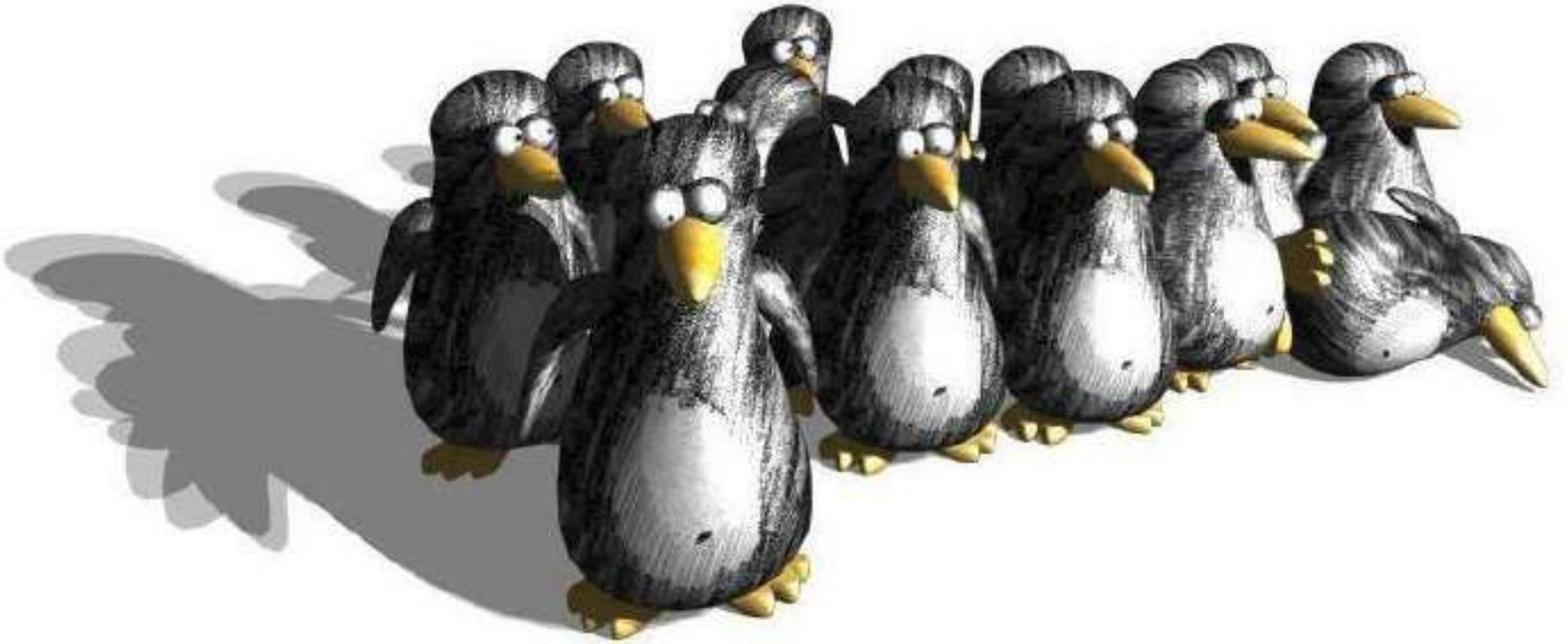
- `/etc/shadow` armazena **senhas criptografadas** com nome de users.
- No **arquivo `/etc/passwd`** há o **mapeamento entre UID e nome**.

O arquivo shadow

- O **arquivo `/etc/shadow`** terá as seguintes informações:

  tux:$1$30RS7b2w$AU/cwF0nFy0oiWMF8t540.:12853:0:99999:7:::

- `$1$30RS7b2w$AU/cwF0nFy0oiWMF8t540.` é o **hash** da **senha 123**.



O arquivo shadow

- Óbvio que as senhas precisam ser **conhecidas apenas** pelo **usuário**.
- Para as **manter secretas**, são criptografadas com **hash unidirecional**.

O arquivo shadow

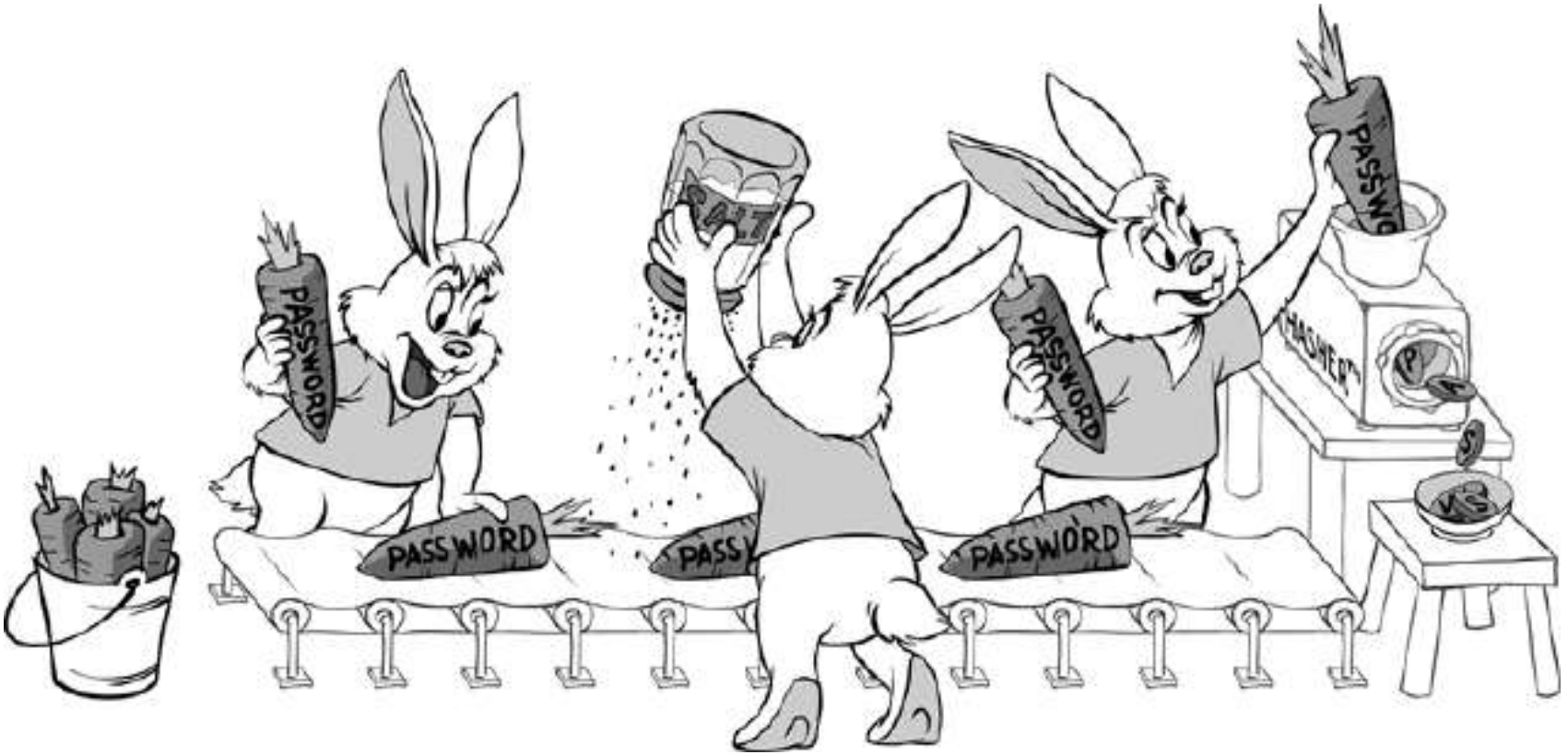
- **Formato da senha**: id**salt**$**hashed** `# fgrep "$" /etc/shadow`
  - **$id** é o algoritmo usado no GNU/Linux da seguinte forma:
    1 is MD5; $2a$ is Blowfish; 5 is SHA-256; 6 is SHA-512

O arquivo shadow

+ **$salt** é combinado a sua senha para gerar um novo hash único.

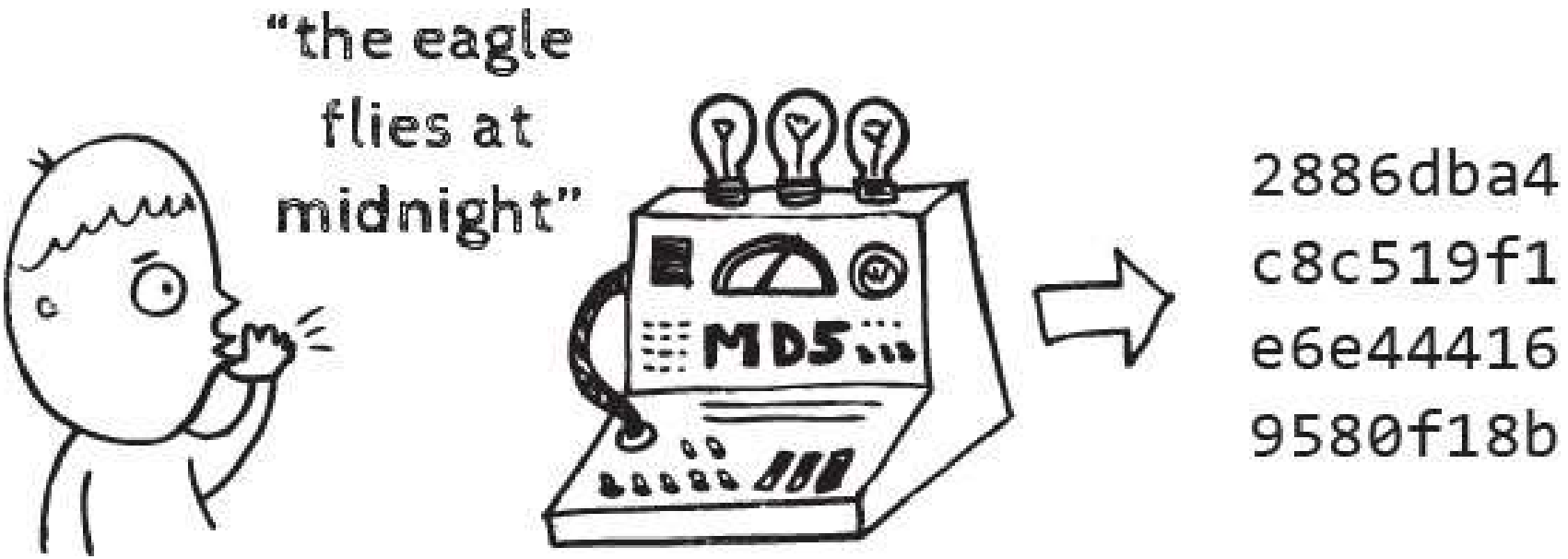
  Ele **evita** que **2 senhas idênticas** produzam **hashes idênticos**.

  Ele dificulta **ataques de rainbow table** (tabelas de hash vazada).



md5sum, sha1sum, sha224sum, ..., sha512sum

+ **$hashed**: **operação irreversível**: **se obtê um hash** mas não o contrário.
- É uma **cadeia de números hexadecimais** de **tamanho fixo**.
- A **simples alteração de um** bit irá **mudar totalmente** o hash!



https://www.youtube.com/watch?v=Rwyf04a1tAc

adduser

- **Criaremos um usuário** `tux` (pode ser você!) com sua senha...
- Um **grupo primário** é **criado para cada usuário** (senhas iguais).

adduser

```
root@linux:~# adduser tux
Adding user 'tux' ...
Adding new group 'tux' (1001) ...
Adding new user 'tux' (1001) with group 'tux' ...
Creating home directory '/home/tux' ...
Copying files from '/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for test
Enter the new value, or press ENTER for the default
   Full Name []:
   Room Number []:
   Work Phone []:
   Home Phone []:
   Other []:
Is the information correct? [Y/n]
```

adduser

- **adduser** também pode **adicionar** um **usuário a um grupo**.

root@linux:~# addgroup penguins
root@linux:~# adduser tux penguins

Chave	Função
`--home` DIR	define o diretório home do usuário.
`--uid` UID	especifica o UID do novo usuário.
`--gid` GID	especifica o GID do grupo primário do usuário.
`--shell` SHELL	especifica o shell padrão do usuário.
`--ingroup` GROUP	define GROUP como o grupo primário.

** Com RedHat -g (GID ou GROUP), -d (home, criar subdir), -p senha cripto.*

passwd

- Para **alterar** a **senha de qualquer usuário**, inclusive a do `root`:

```
root@linux:~# psswd tux
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

- **`root` pode alterar** qualquer senha, **usuário comum** somente **dele**!

passwd

- Para **bloqueiar um usuário** (lock).

  root@linux:~# passwd -l tux

- Para **desbloqueiar um usuário** (unlock).

  root@linux:~# passwd -u tux

https://www.youtube.com/watch?v=-VXt9L_UjiY (opt)

deluser

- **Remove** o **usuário de um grupo**.

  root@linux:~# deluser tux penguins

- **Remove um usuário** normal do sistema.

  root@linux:~# deluser tux

- **Remove um grupo** do sistema.

  root@linux:~# deluser --group penguins

Chave	Função
`--remove-home`	Remove o diretório pessoal.
`--remove-all-files`	Remove todos os arquivos deste dono.

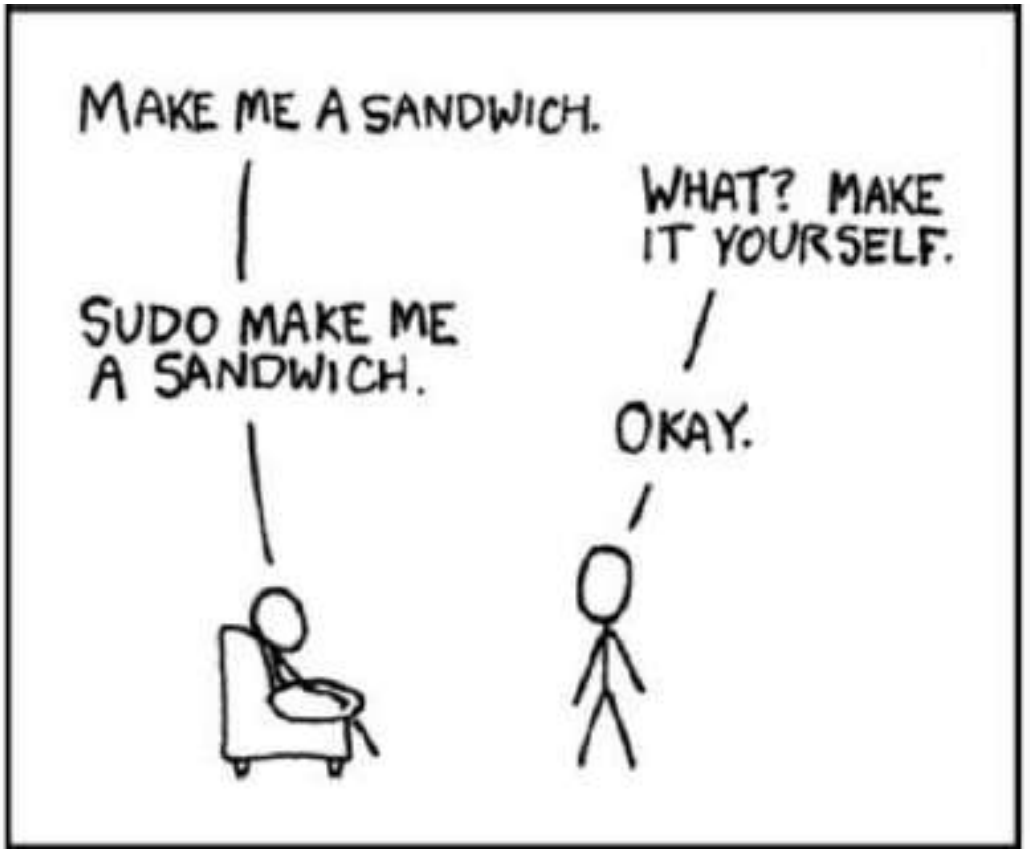
```
tux@linux:~$ file $(which deluser)
/usr/sbin/deluser: Perl script text executable
tux@linux:~$ grep "userdel" $(which deluser)
my $userdel = &which('userdel');
```

sudo (super-user do)

- **Privilégios extras** podem ser concedidos a **usuários temporariamente**.

sudo (super-user do)

- **Ninguém precisa saber** a **senha do root** (só a senha do usuário atual).
- Usar **sudo é melhor** (+ seguro) do que abrir uma **sessão como root**!



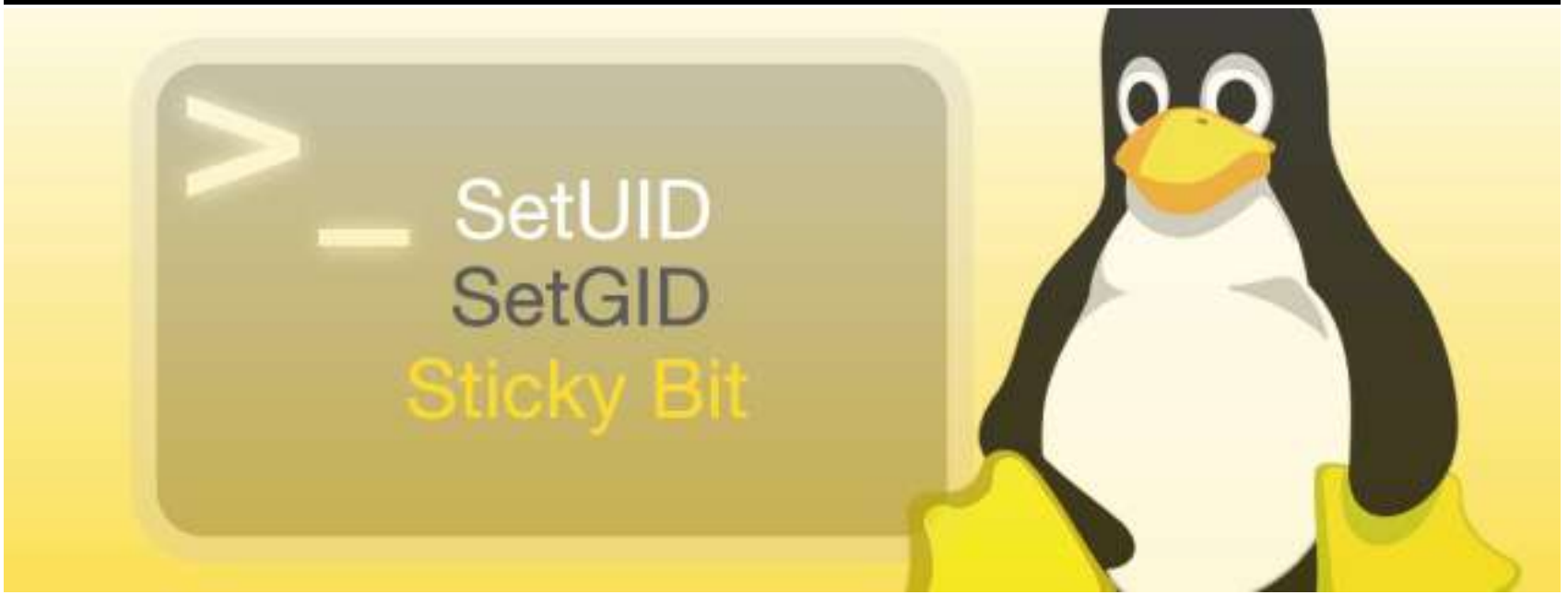
sudo (super-user do)

- Se sudo **solicite uma senha** é a **senha do usuário, não do root**!
- Ao **inves do su**, precisa dos **credenciais do** chamando, **não do destino**.

sudo (super-user do)

- **`/usr/bin/sudo`** é um **arquivo executável** com **setuid** configurado.
- **Usuários** rodam **executável** com **permissões do dono/grupo** dele.

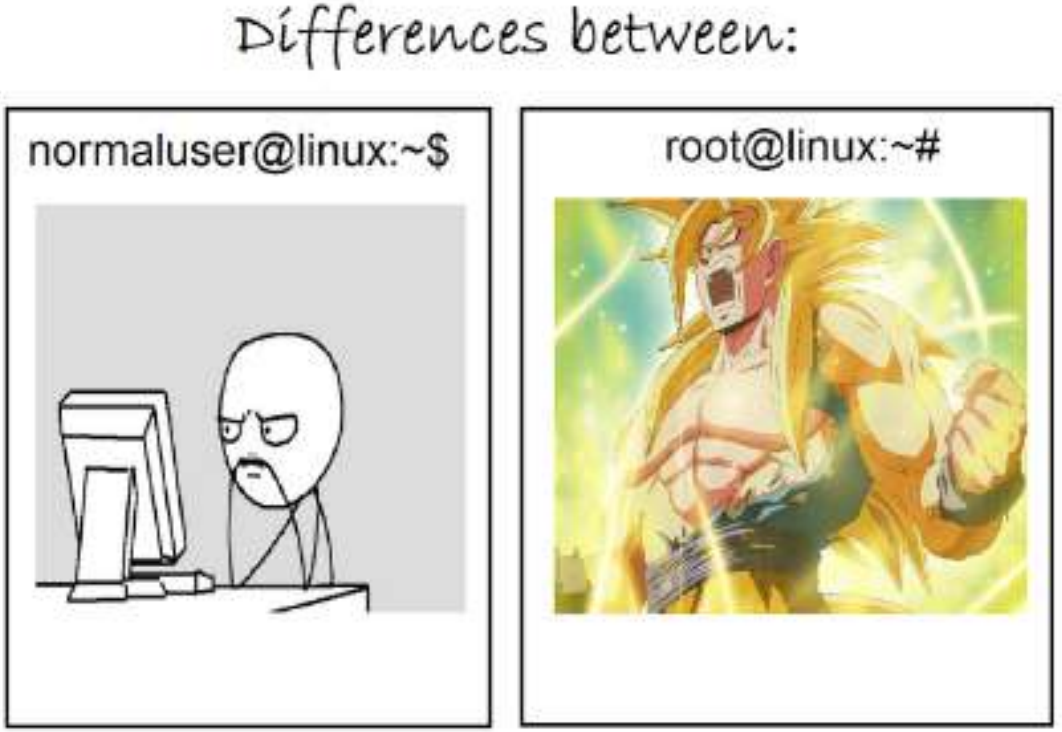
```
✗ ls -l /usr/bin/sudo
-rwsr-xr-x 1 root root 136808 Jul  4  2017 /usr/bin/sudo
   ^
   The magic bit here is "s" making it a setuid binary
```



sudo (super-user do)

Uma vez que sudo está sendo executado como root, faz três coisas:

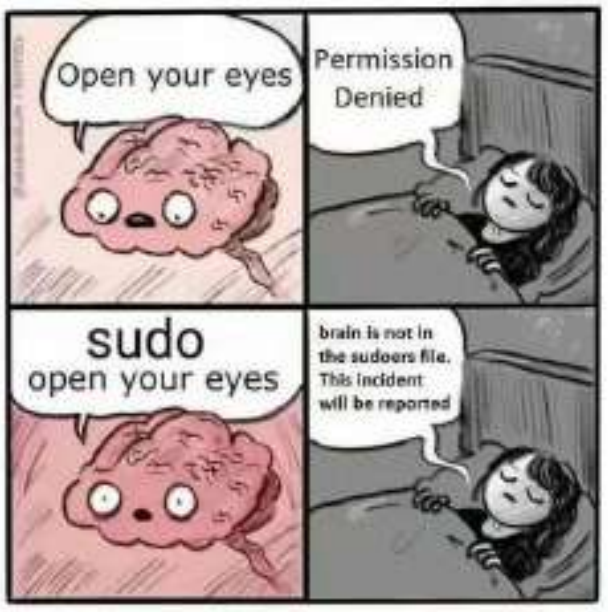
1. **Peça sua senha** e **compare-a com /etc/shadow**.
2. **Verifique** se seu **nome de usuário ou grupo** está **no `/etc/sudoers`**.
3. **exec o comando** que você deseja **executar como root**.

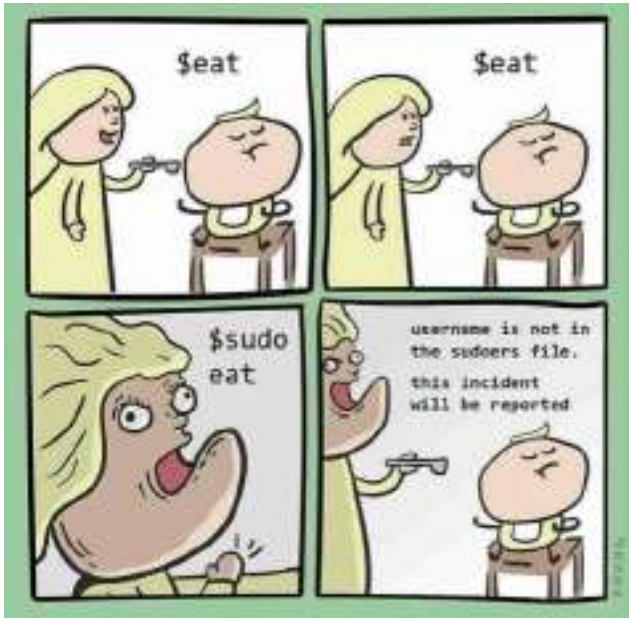


sudo (super-user do)

- **sudo** faz check-in `/etc/sudoers` para **ver seus privilégios**.

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL
# Allow members of group Sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```



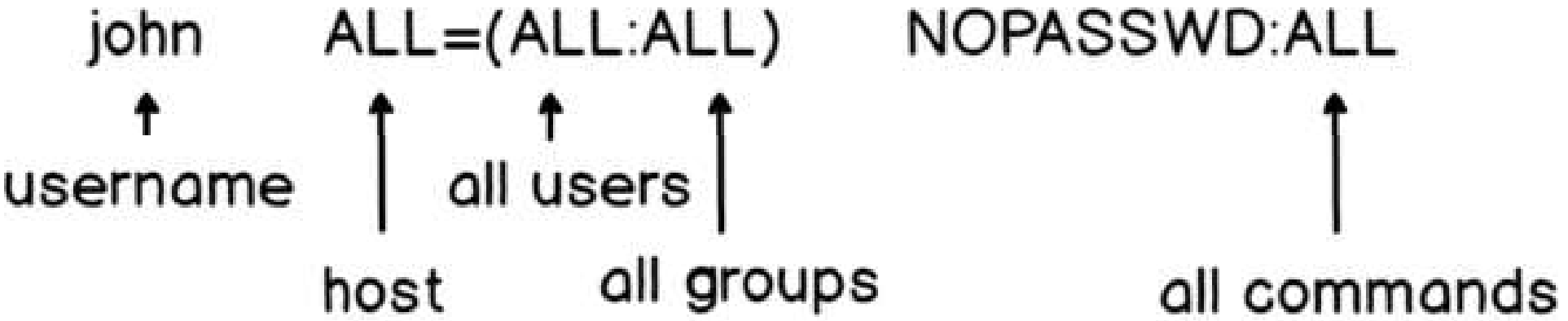


sudo (super-user do)

- Quando o **Linux configura a primeira conta** durante a instalação...
- Ele é **adicionado ao grupo "sudo"**.

sudo (super-user do)

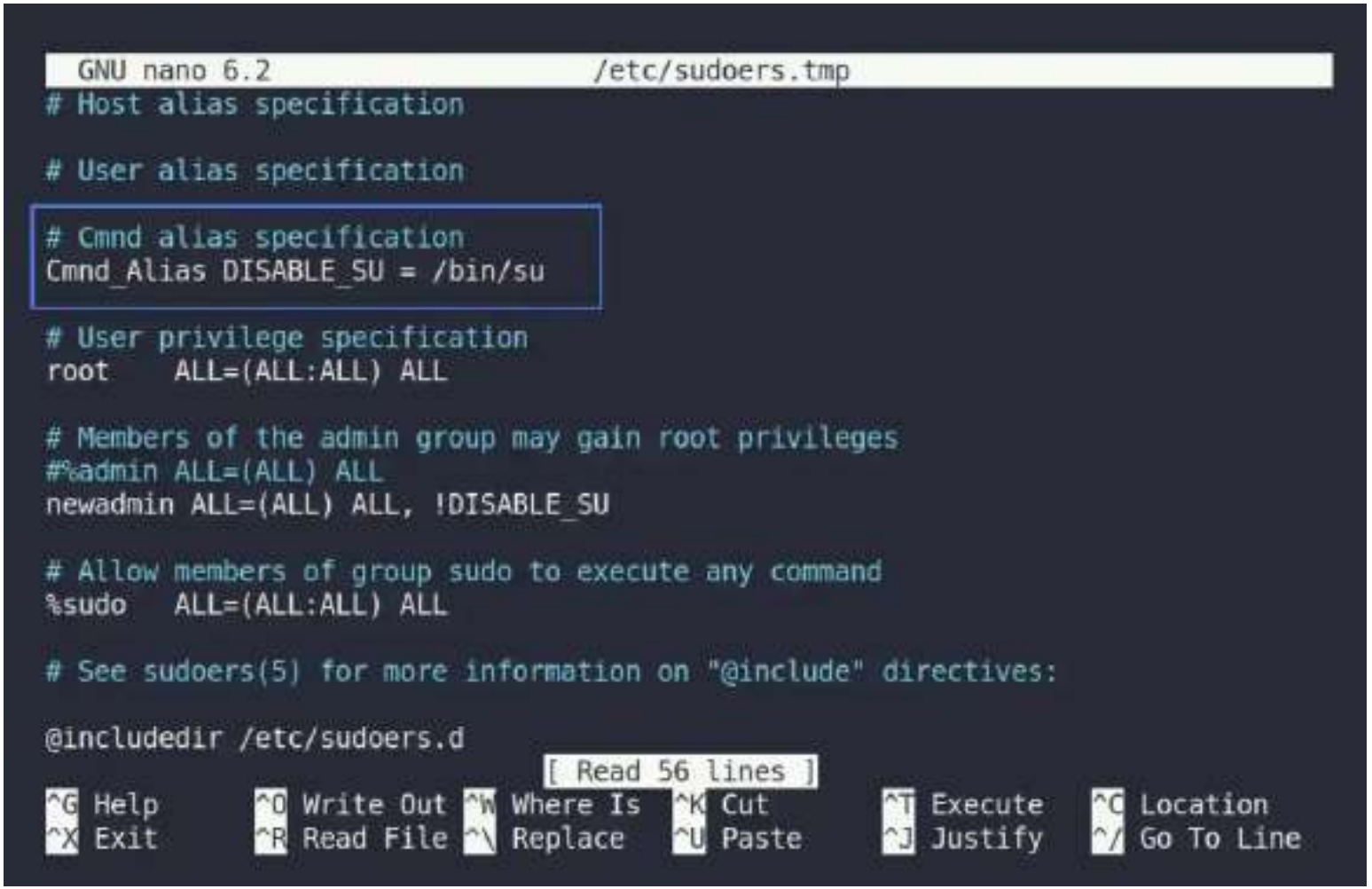
1. A lista de **hosts** corresponde aos nomes de **maquinas**, **endereços IP**...
2. (ALL: permissão para executar como **qualquer usuário**: sudo -u <user>
3. :ALL) permissão para executar como **qualquer grupo**: sudo -g <group>



4. Comandos à direita de **NOPASSWD** executados **sem senha** user/group. Aqueles deixados à sua esquerda ainda estão **sujeitos à autenticação**.

sudo (super-user do)

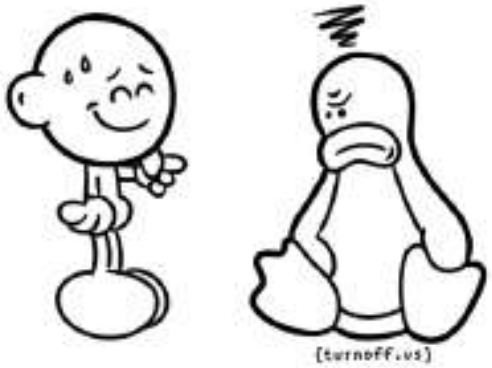
- ALL: **todos os comandos**, se não **caminhos** listados (/bin/ls) sep por ,
- **Excluir** host,user,group,comandos **com !** (e.g. !/usr/bin/passwd)



sudo (super-user do)

- Pode **mudar para root** usando `sudo -i` (equivalente a `sudo su`).

```
$ sudo su
Sorry, your user
is not allowed
$ su jane
Hi jane
$ sudo su
root# _
```



JUST SUDO IT

sudo (super-user do)

- *Eu pensei que estava mudando para o usuário root.*
  Você ganha poderes de root por causa do setuid de `/usr/bin/sudo`.

- *Existe um usuário root?*
  Sim, existe uma conta root, separada da sua conta de usuário.

- *Eu sou root?*
  Não, você não é root. Você apenas tem privilégios iguais!

- *Por que executar comandos sudo com minha senha sendo já logado?*
  Você precisa inserir a senha do usuário apenas por segurança.

https://www.ti-enxame.com/pt/sudo/por-que-o-comando-sudo-nao-precisa-da-senha-root/960787438/

Permissões de arquivos e diretórios

- Existem **permissões** de **escrita**, de **leitura** e de **execução**.

RHA Chapter 7: Controle de acesso a arquivos

Permissões de arquivos e diretórios

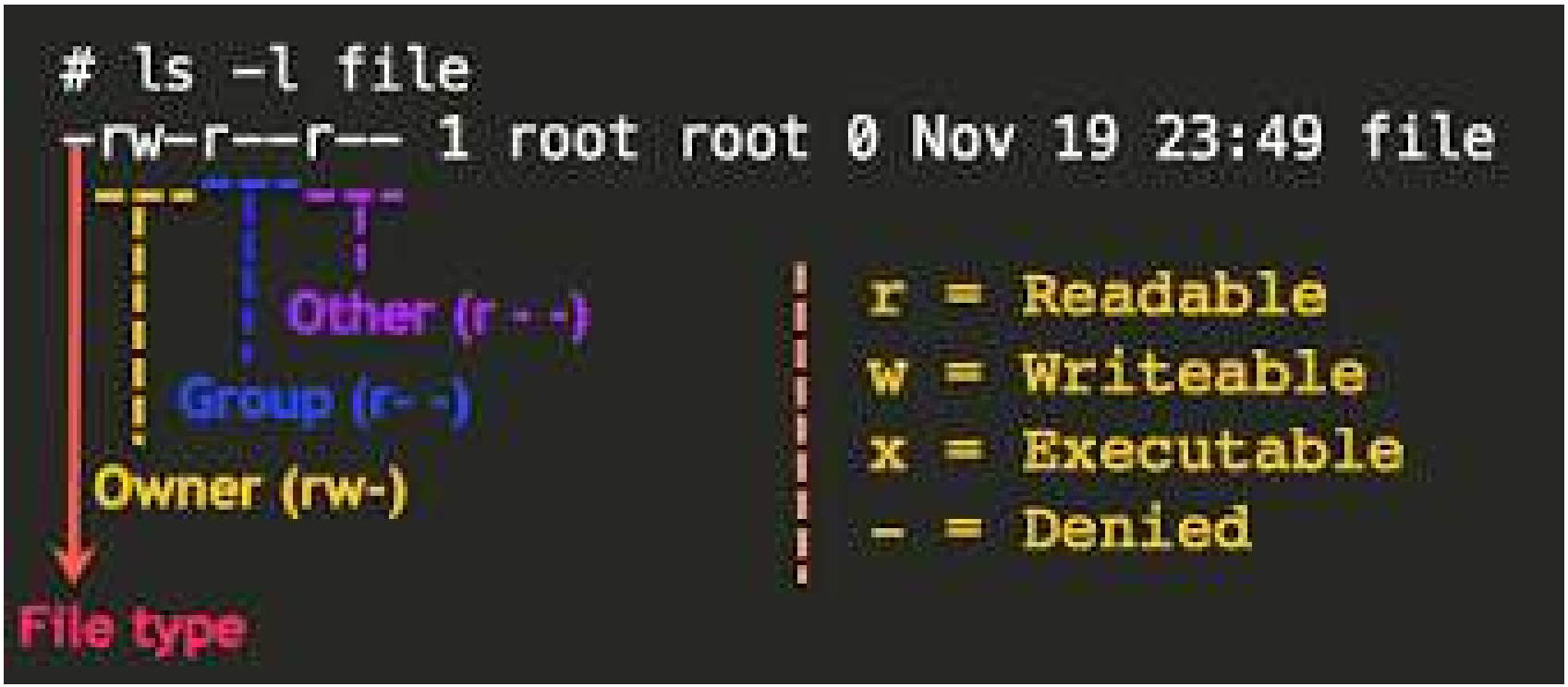
- O **root pode fazer tudo** com todos os arquivos e diretórios do OS!

Permissões de arquivos e diretórios

- Cada **arquivo ou diretório** tem permissões para **dono**, **grupo** e...

Permissões de arquivos e diretórios

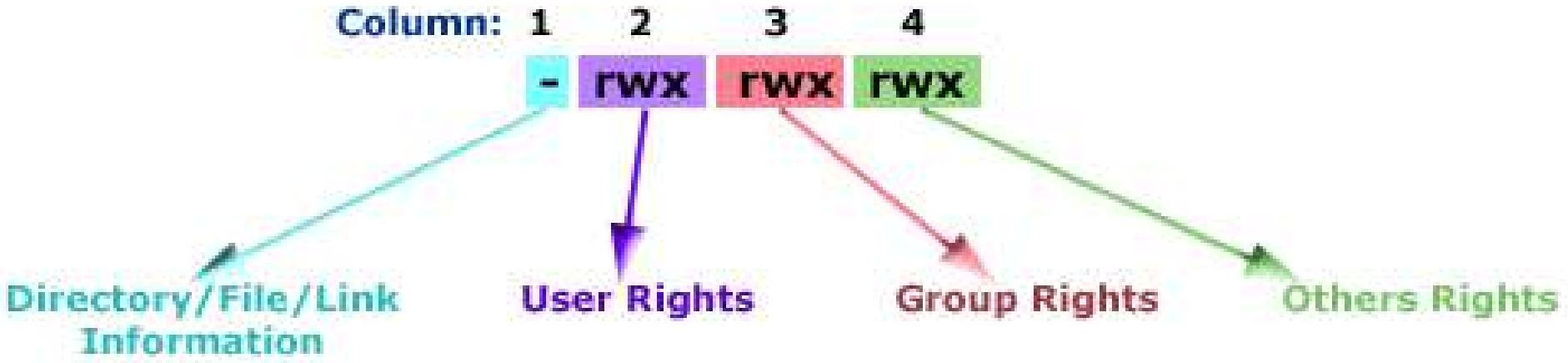
```
# cd /etc; ls -l
drwxr-xr-x 6 root root    4096 Feb  9  2021 kernel
-rw-r--r-- 1 root root    2842 Mar 21 18:48 passwd
lrwxrwxrwx 1 root root      13 Mar 15 09:59 rmt -> /usr/sbin/rmt
-rw-r----- 1 root shadow 1584 Mar 21 18:48 shadow
-r--r----- 1 root root     755 Feb  3  2020 sudoers
```



Permissões de arquivos e diretórios

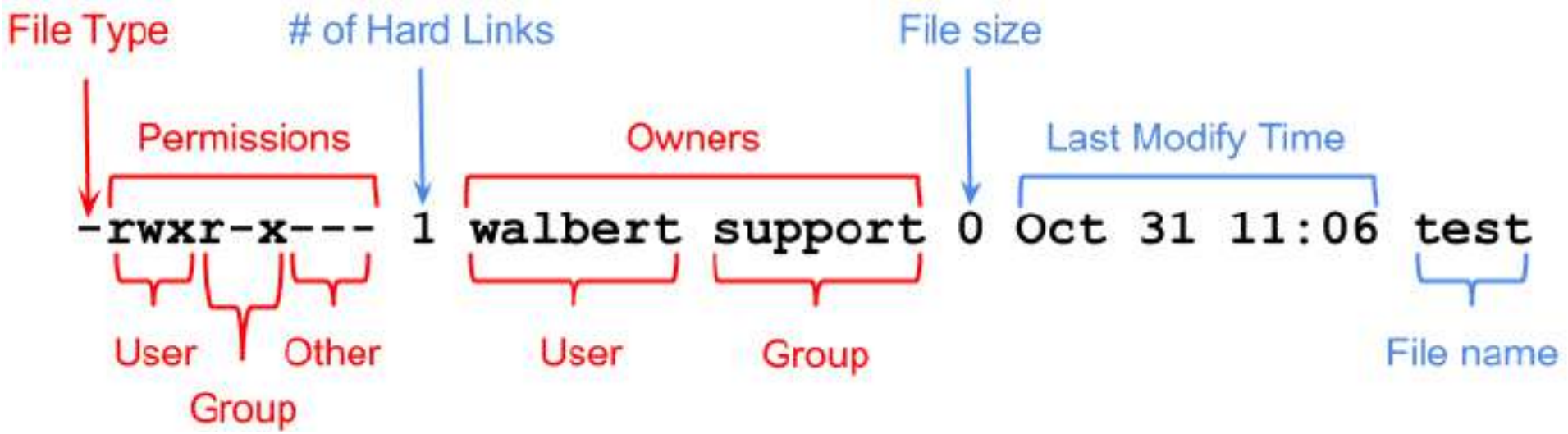
- Visualizamos uma **primeira sequencia** composto por **10 caracteres**.

Posição	Alter.	Ref	Arquivos	Diretórios
1°	-, d, l	tipo	regular (-) dir, link	diretório (d)
2°,5° e 8°	r ou -	d, g, o	pode ser lido (r)	conteúdo lido (r)
3°, 6° e 9°	w ou -	d, g, o	alterável (w)	conteúdo cud (w)
4°, 7° e 10°	x ou -	d, g, o	executável (x)	acesso no diretório (x)



Permissões de arquivos e diretórios

- Depois mostra quem é o **dono** e **grupo dono** e **tamanho** do item.
- **Criação do arquivo/diretório** ou **última alteração** e o **nome**.



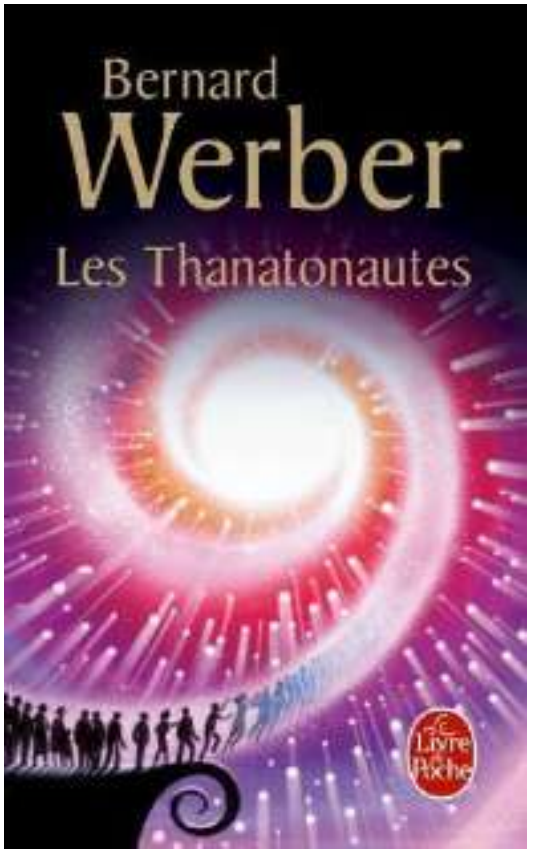
chmod (change mode/octal)

- Para **alterar as permissões**, utliza-se o **comando chmod**:

  $ chmod dgo <arquivo ou diretório ou link> *dono, grupo e outros

Octal Representation

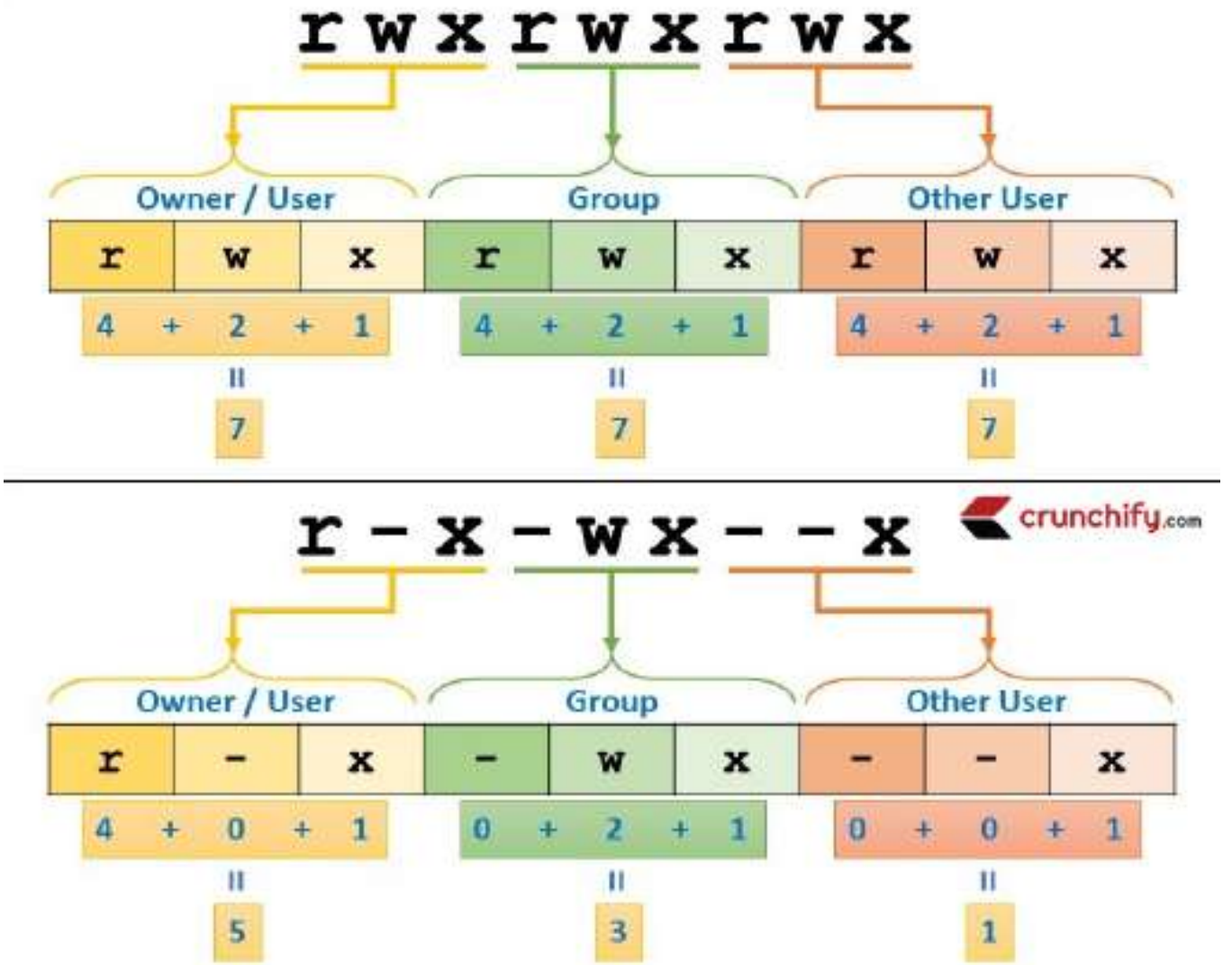
0	000	-	-	-	No permissions
1	001	-	-	x	Only Execute
2	010	-	w	-	Only Write
3	011	-	w	x	Write and Execute
4	100	r	-	-	Only Read
5	101	r	-	x	Read and Execute
6	110	r	w	-	Read and Write
7	111	r	w	x	Read, Write and Execute



http://sofaraway.unblog.fr/werber-la-theorie-des-chiffres

chmod (change mode/octal)

- Cada uma das **variáveis** é **substituída** pela **soma dos valores**.



chmod (change mode/simbólico)

- Se **alterar**... não é necessário **definir novo grupo** de permissões.

  $ chmod Who/What/Which <arquivo ou diretório ou link>

1. **Who** é a classe do user u (dono), g (grupo), o (outros) ou a (all).

2. **What** é o **operador que modifica**.

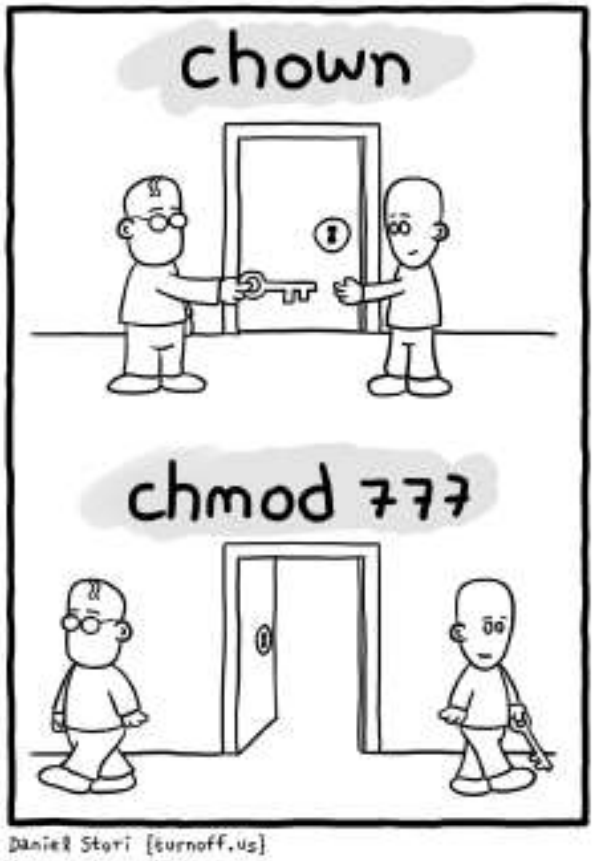
O que	Conjunto	Descrição
+	add	Adiciona as permissões ao arquivo.
-	remove	Remove as permissões do arquivo.
=	set exactly	Define as permissões fornecidas para o arquivo.

3. **Which** especifica as **permissões**: r, w, x, e X

   $ chmod u=rwx simbolic1
   $ chmod go-rw simbolic2
   $ chmod a+x simbolic3

chown change owner (alteração do dono)

- Todo **arquivo**, **diretório** ou **link** pertence a um **usuário e** um **grupo**.



Check the current file ownership using ls -l

```
-rw-rw-r--  1 root n10    18 2012-09-16 18:17 sample.txt
```

Change the file owner to n100 . You will need sudo

```
n10@N100:~$ sudo chown n100 sample.txt
```

ownership changed to n100

```
-rw-rw-r--  1 n100 n10    18 2012-09-16 18:17 sample.txt
```

Changing user and group to root 'chown user:group file'

```
n10@N100:~$ sudo chown root:root sample.txt
```

user and Group ownership changed to root

```
-rw-rw-r--  1 root root    18 2012-09-16 18:17 sample.txt
```

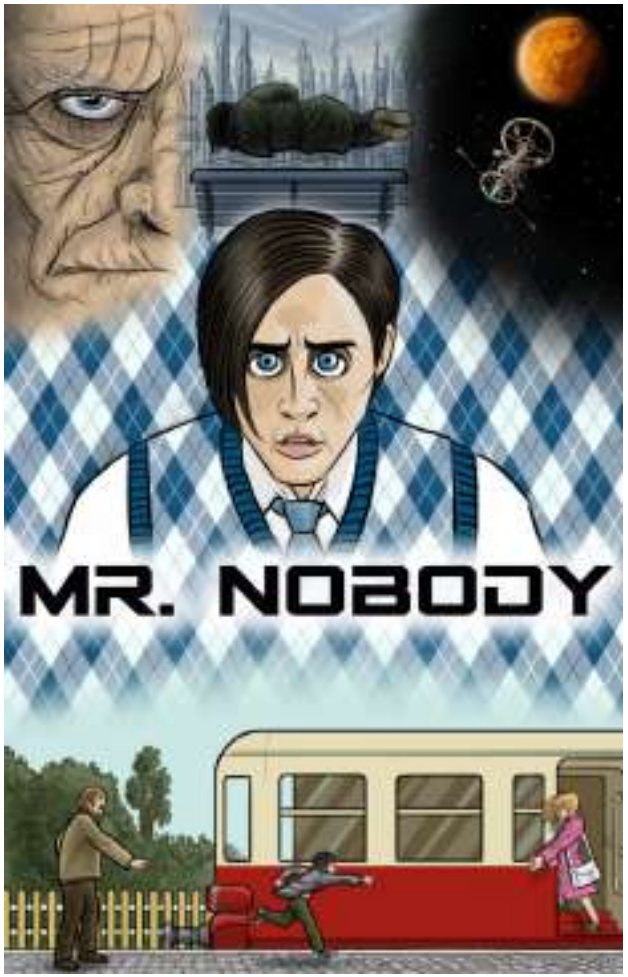
chown change owner (alteração do dono)

- Para **alterar dono e grupo** deveremos utilizar o **comando chown**.
- O nome do dono deverá estar **separado** do nome do grupo **por . ou :**

Comando	Resultado
`# chown linus.fsf /etc/xwz`	Alterar o dono e grupo de wxz.
`# chown linus:fsf /etc/xwz`	Mesmo efeito do comando anterior.
`# chown linus /etc/xwz`	Altera somente o dono do xwz.
`# chown .games /etc/xwz`	Altera somente o grupo do xwz.
`# chown nobody.nogroup /dir`	Altera dono nobody, grupo nogroup.

chown change owner (alteração do dono)

- **nobody** e/ou **nogroup torna um arquivo ou diretório público**.



Recursividade

- Os **comandos chmod e chown** aceitam **recursividade** com a **chave -R**.

Comando	Resultado
`# chown -R linus ./dir`	Altera recursivamente o dono.
`# chmod -R 764 /dir`	Altera recursivamente as permissões.
`# chmod -R g+rwx ./dir`	Altera recursivamente as permissões.