SEG12 - Atividades - Semana 1

Francisco Marcelo, Marcelo Karam e Felipe Scarel

06-08-2018

Introdução ao sistema operacional Linux

1) Identificando bits de permissão

1. Verifique as permissões do diretório `/tmp`. O que você percebe de diferente em relação às permissões de *outros*?

   ```
   $ ls -lha / | grep 'tmp$'
   drwxrwxrwt  7 root root 4,0K Ago  7 01:01 tmp
   ```

   O sticky bit está definido: `t`.

2. Considerando que há permissão de escrita no diretório para todos, o que o impediria de remover um arquivo de outra pessoa?

   ```
   $ rm -f /tmp/file_root
   rm: não foi possível remover “/tmp/file_root”: Operação não permitida
   ```

   Com o sticky bit definido somente o dono de um arquivo pode removê-lo.

2) Identificando e entendendo *hard links*

O número de *links* (*link counter*) que apontam para um arquivo é mantido em seu *inode*. Esse contador é utilizado pelo sistema para controlar a liberação dos blocos do disco alocados ao arquivo quando o contador atingir o valor zero ,ou seja, quando nenhum outro arquivo estiver apontando para o *inode*.

1. Qual o número de *links* do seu diretório *home*?

   ```
   $ ls -lha /home/ | egrep ' aluno$'
   drwxr-xr-x  2 aluno  aluno  4,0K Ago  7 01:45 aluno
   ```

   Como visto acima, `2`. Esse número não é fixo, mas depende do conteúdo do diretório. Um diretório recém criado, que não tenha nenhum conteúdo possui dois *links* (um referente ao próprio diretório e outro referente à entrada especial ".").

2. Crie o arquivo `arqses1ex3` no seu diretório *home*. Utilize o comando *touch*.

   ```
   $ touch ~/arqses1ex3
   $ ls /home/aluno
   arqses1ex3
   ```

3. Verifique o número de *links* do arquivo `arqses1ex3` e anote o resultado. Você pode utilizar o

redirecionamento de saída para registrar esse resultado no próprio arquivo criado. Essa informação será necessária para uma atividade posterior.

```
$ mytemp=$(mktemp) && ls -lha ~/arqses1ex3 | tee nlinks && awk '{print $2}' nlinks
> $mytemp && mv $mytemp nlinks
-rw-r--r-- 1 aluno aluno 0 Ago  7 01:52 /home/aluno/arqses1ex3
$ cat nlinks
1
```

O arquivo `arqses1ex3` possui apenas um link.

4. Verifique se mudou o número de *links* do seu diretório *home*.

```
$ ls -lha /home/ | egrep ' aluno$'
drwxr-xr-x  2 aluno  aluno  4,0K Ago  7 02:05 aluno
```

O número de *links* continuou o mesmo.

5. Crie um diretório com o nome de `dirses1ex3`, também no seu diretório *home*.

```
$ mkdir /home/aluno/dirses1ex3
$ ls ~
arqses1ex3  dirses1ex3  nlinks
```

6. Mais uma vez, verifique o número de *links* do seu diretório *home*. Ele mudou? Você saberia dizer por quê?

```
$ ls -lha /home/ | egrep ' aluno$'
drwxr-xr-x  3 aluno  aluno  4,0K Ago  7 02:11 aluno
```

O número de *links* aumentou em uma unidade, por conta de entrada especial ".." presente no diretório `/home/aluno/dirses1ex3`, que aponta para o diretório `/home/aluno`.

7. Qual o número de links do diretório `dirses1ex3`?

```
$ ls -lha ~ | egrep ' dirses1ex3$'
drwxr-xr-x 2 aluno aluno 4,0K Ago  7 02:11 dirses1ex3
```

Como visto acima, `2`.

8. Verifique qual opção deve ser passada ao comando `ls` para que ele liste as informações do diretório `dirses1ex3` e não o seu conteúdo.

```
$ ls -dl ~/dirses1ex3/
drwxr-xr-x 2 aluno aluno 4096 Ago  7 02:11 /home/aluno/dirses1ex3/
```

Devem ser passadas as opções `-d` e `-l`.

9. Você saberia explicar por que o número de *links* do diretório `dirses1ex3` é maior que um?

   Os dois *links* são relativos ao próprio diretório. Um aponta o caminho direto `/home/aluno` → `/home/aluno/dirses1ex3` e o outro corresponde à entrada especial ".", presente no próprio diretório `/home/aluno/dirses1ex3`.

3) Conhecendo diferenças entre *hard link* e *symbolic link*

Foi explicada a importância dos *links* criados com o comando `ln`. Para criar um *symbolic link*, a opção `-s` deve ser informada na linha de comando. Consulte as páginas do manual para conhecer outras opções.

1. No seu diretório de trabalho, crie um *hard link* para o arquivo `arqses1ex3`. O nome do arquivo criado deverá ser `hosts.hard`.

```
$ ln /home/aluno/arqses1ex3 /home/aluno/hosts.hard
$ ls ~
arqses1ex3  dirses1ex3  hosts.hard  nlinks
```

2. Verifique agora o número de links do arquivo `arqses1ex3` e compare com aquele obtido na atividade 2. Explique a diferença.

```
$ ls -lha /home/aluno/arqses1ex3 | awk '{print $2}'
2
$ cat nlinks
1
```

   O número de *links* foi aumentado de 1 para 2 devido à criação do *link* `hosts.hard`.

3. Crie um *symbolic link* para o arquivo `arqses1ex3`, que deverá se chamar `hosts.symbolic`.

```
$ ln -s /home/aluno/arqses1ex3 /home/aluno/hosts.symbolic
$ ls
arqses1ex3  dirses1ex3  hosts.hard  hosts.symbolic  nlinks
```

4. O número de *links* do arquivo `arqses1ex3` aumentou?

```
$ ls -lha /home/aluno/arqses1ex3
-rw-r--r-- 2 aluno aluno 0 Ago  7 01:52 /home/aluno/arqses1ex3
```

Não, não aumentou.

5. Caso não tenha aumentado, por que isso aconteceu, considerando que foi criado um *link* para ele?

   Porque o *symbolic link* aponta para outro *inode*.

6. Qual o tamanho do arquivo *hosts.symbolic*?

   ```
   $ du -sb ~/hosts.symbolic
   22      /home/aluno/hosts.symbolic
   ```

   Como mostrado acima, 22 bytes.

7. Você percebe alguma correlação entre o tamanho e o arquivo para o qual ele aponta?

   ```
   $ ls -d /home/aluno/arqses1ex3 | tr -d '\n' | wc -c
   22
   ```

   Esse tamanho representa o número de caracteres presentes no *path* completo do arquivo original linkado, sendo cada caractere representado por 1 byte.

4) Trabalhando com *hard link* e *symbolic link*

1. Se o arquivo original `arqses1ex3` fosse removido, o que aconteceria se tentássemos acessá-lo pelo *hard link*? E pelo *symbolic link*?

   Pelo *hard link* conseguiríamos acessar o conteúdo do arquivo normalmente. Já pelo *symbolic link* não conseguiríamos acessar o conteúdo do arquivo, uma vez que o mesmo é somente uma referência para o arquivo original.

2. Depois de responder a essas questões, remova o arquivo criado (`arqses1ex3`) e verifique se as suas respostas estão corretas.

```
$ rm arqses1ex3

$ ls -l hosts.hard
-rw-r--r-- 1 aluno aluno 0 Ago  7 01:52 hosts.hard
$ ls -l hosts.symbolic
lrwxrwxrwx 1 aluno aluno 22 Ago  7 02:38 hosts.symbolic -> /home/aluno/arqses1ex3

$ cat hosts.hard
$ cat hosts.symbolic
cat: hosts.symbolic: Arquivo ou diretório não encontrado
```

As respostas acima estão corretas.

5) Conhecendo algumas limitações do *hard link*

1. Crie um arquivo chamado `arqses1ex6`. Em seguida, crie um *hard link* para esse arquivo com o nome `link-arqses1ex6` no diretório `/tmp`. O que aconteceu? Por quê? Como resolver esse problema?

   Observação: Para que esta atividade tenha efeito, o diretório `/tmp` deverá ter sido criado numa partição diferente da partição onde se encontra o *home* do usuário. Caso essa situação não ocorra, verifique se existe o diretório `/var/tmp` e veja se ele está em outra partição. Se for o caso, use este último para fazer o exercício.

```
$ touch ~/arqses1ex6
$ ln ~/arqses1ex6 /tmp/link-arqses1ex6
ln: failed to create hard link "/tmp/link-arqses1ex6" => "/home/aluno/arqses1ex6":
Link entre dispositivos inválido

$ df -h | sed -n '1!p' | egrep -v '^tmpfs |^udev ' | awk '{printf "%s\t mounted on:
%s\n", $6, $1}'
/        mounted on: /dev/sda1
/tmp     mounted on: /dev/sda6
```

   Não foi possível criar o *hard link*, porque o diretório `/tmp` está em outra partição.

6) Criando *links* para diretórios

Crie, no seu diretório *home*, um *link* simbólico para o diretório `/usr/bin` com o nome de `link-bin`. Com o *link* criado, execute o seguinte:

1. Mude para o diretório `link-bin`.

```
$ ln -s /usr/bin /home/aluno/link-bin ; cd link-bin
$ pwd
/home/aluno/link-bin
```

2. Agora, vá para o diretório pai (utilize a notação ".."). Você saberia explicar por que se encontra no seu diretório *home* e não no diretório `/usr`?

```
$ cd ..
$ pwd
/home/aluno
```

   Porque o *link* simbólico é apenas uma referência para o diretório.

7) Alterando permissões de arquivos e diretórios

O comando `chmod` é utilizado para modificar as permissões de um arquivo. Utilizando a notação octal, execute a seguinte sequência:

1. Modifique a permissão do seu diretório *home* de modo a retirar a permissão de escrita do seu dono.

```
$ chmod 555 /home/aluno
$ ls -ld /home/aluno
dr-xr-xr-x 3 aluno aluno 4096 Ago  7 03:38 /home/aluno
```

2. Verifique as permissões associadas ao arquivo `arqses1ex6`. Você tem permissão para escrever nesse arquivo? O grupo tem?

```
$ ls -lha ~/arqses1ex6
-rw-r--r-- 1 aluno aluno 0 Ago  7 02:55 /home/aluno/arqses1ex6
```

   Somente o dono do arquivo tem permissão para escrever no mesmo.

3. Tente remover o arquivo `arqses1ex6`. Você conseguiu? Em caso negativo, você sabe explicar o motivo?

```
$ rm ~/arqses1ex6
rm: não foi possível remover “/home/aluno/arqses1ex6”: Permissão negada
```

   Não, porque o diretório `/home/aluno` está sem permissão de escrita para o dono.

4. Modifique as permissões do arquivo `arqses1ex6` de forma a retirar a permissão de escrita para o dono e colocá-la para o grupo.

```
$ chmod 464 ~/arqses1ex6
$ ls -ld ~/arqses1ex6
-r--rw-r-- 1 aluno aluno 0 Ago  7 02:55 /home/aluno/arqses1ex6
```

5. Com o uso de redirecionamento, tente copiar o conteúdo do seu diretório *home* para dentro do arquivo `arqses1ex6`.

```
$ ls -lha /home/aluno > /home/aluno/arqses1ex6
-bash: /home/aluno/arqses1ex6: Permissão negada
```

   Apresentou erro de permissão de gravação no diretório por parte do dono.

6. Torne a colocar a permissão para escrita no seu diretório *home* para o dono.

```
$ chmod 755 /home/aluno
$ ls -ld ~
drwxr-xr-x 3 aluno aluno 4096 Ago  7 03:38 /home/aluno
```

8) Atribuindo as permissões padrão

1. Crie arquivos (`arq1ses1ex9`, `arq2ses1ex9`, etc.) e diretórios (`dir1ses1ex9`, `dir2ses1ex9`, etc.) em seu diretório *home*, após definir cada uma das seguintes *umasks*: `000`; `002`; `003`; `023`; `222`; `022`. Em seguida, observe as permissões que foram associadas a cada um dos arquivos e diretórios.

```
$ umask 000 ; touch arq1ses1ex9 ; mkdir dir1ses1ex9
$ umask 002 ; touch arq2ses1ex9 ; mkdir dir2ses1ex9
$ umask 003 ; touch arq3ses1ex9 ; mkdir dir3ses1ex9
$ umask 023 ; touch arq4ses1ex9 ; mkdir dir4ses1ex9
$ umask 222 ; touch arq5ses1ex9 ; mkdir dir5ses1ex9
$ umask 022 ; touch arq6ses1ex9 ; mkdir dir6ses1ex9

$ ls -lha /home/aluno | egrep 'arq[1-6]ses1ex9|dir[1-6]ses1ex9'
-rw-rw-rw- 1 aluno aluno    0 Ago  7 03:50 arq1ses1ex9
-rw-rw-r-- 1 aluno aluno    0 Ago  7 03:50 arq2ses1ex9
-rw-rw-r-- 1 aluno aluno    0 Ago  7 03:50 arq3ses1ex9
-rw-r--r-- 1 aluno aluno    0 Ago  7 03:52 arq4ses1ex9
-r--r--r-- 1 aluno aluno    0 Ago  7 03:52 arq5ses1ex9
-rw-r--r-- 1 aluno aluno    0 Ago  7 03:52 arq6ses1ex9
drwxrwxrwx 2 aluno aluno 4,0K Ago  7 03:50 dir1ses1ex9
drwxrwxr-x 2 aluno aluno 4,0K Ago  7 03:50 dir2ses1ex9
drwxrwxr-- 2 aluno aluno 4,0K Ago  7 03:50 dir3ses1ex9
drwxr-xr-- 2 aluno aluno 4,0K Ago  7 03:52 dir4ses1ex9
dr-xr-xr-x 2 aluno aluno 4,0K Ago  7 03:52 dir5ses1ex9
drwxr-xr-x 2 aluno aluno 4,0K Ago  7 03:52 dir6ses1ex9
```

9) Entendendo as permissões padrões

1. Na execução do exercício anterior, você saberia explicar por que, ainda que utilizando a mesma *umask*, as permissões associadas ao arquivo criado diferem das do diretório?

   O comando `umask` trabalha de forma diferente com arquivos e diretórios. Por motivos de segurança um um novo arquivo nunca recebe a permissão de execução quando da sua criação.

Usuários e grupos

1) Criando contas de usuários

Uma das atividades que fazem parte da rotina diária de um administrador de sistemas é o gerenciamento de contas de usuários. Frequentemente, usuários são criados, modificados, desabilitados ou excluídos do sistema

1. Descubra se o sistema faz uso de *shadow passwords* ou se ainda utiliza o esquema tradicional.

   ```
   $ ls -ld /etc/gshadow /etc/shadow
   -rw-r----- 1 root shadow  666 Ago  5 16:52 /etc/gshadow
   -rw-r----- 1 root shadow 1125 Ago  5 16:51 /etc/shadow
   ```

   O aluno deve verificar se os arquivos `/etc/shadow` e `/etc/gshadow` existem.

2. Crie uma conta para você no sistema, seguindo os passos descritos na aula teórica e no material didático.
   - Editar o arquivo `/etc/group` e inserir uma nova linha com os parâmetros relativos ao grupo do novo usuário:
     - Nome do grupo;
     - Senha ("x");
     - GID;
     - Membros do grupo.

     ```
     marcelo:x:1001:
     ```

   - Editar o arquivo `/etc/gshadow` e inserir uma nova linha com os parâmetros relativos ao grupo do novo usuário:
     - Nome do grupo;
     - Senha criptografada do grupo ("!");
     - Administradores do grupo;
     - Membros do grupo.

     ```
     marcelo:!::
     ```

   - Editar o arquivo `/etc/passwd` e inserir uma nova linha com os parâmetros relativos à conta do novo usuário:
     - Nome do usuário;
     - Senha ("x");

    - UID;
    - GID;
    - GECOS: campo com comentários informativos do usuário;
    - Diretório *home*;
    - Shell de login.

```
marcelo:x:1001:1001:,,,:/home/marcelo:/bin/bash
```

- Editar o arquivo `/etc/shadow` e inserir uma nova linha os parâmetros relativos à conta do novo usuário:
    - Nome do usuário;
    - Senha criptografada: inserir valor "*", que será alterado a seguir;
    - *last_change*: número de dias desde a última alteração de senha;
    - *minimum*: número mínimo de dias até que senha possa ser alterada novamente;
    - *maximum*: número máximo de dias até que a senha deva ser alterada;
    - *warning*: número de dias para aviso de expiração de senha;
    - *inactive*: número de dias após expiração em que a senha será aceita;
    - *expire*: data para expiração da senha.

```
marcelo:*:16846:0:99999:7:::
```

- Definir uma senha para a nova conta, utilizando o comando `passwd`:

```
# passwd marcelo
```

- Copiar os arquivos de inicialização contidos no diretório `/etc/skel` para o diretório *home* do usuário.

```
# cp -r /etc/skel /home/marcelo
```

- Alterar o usuário e grupo donos dos arquivos na pasta *home* do novo usuário:

```
# chown -R marcelo.marcelo /home/marcelo
```

- Configurar a *quota* de disco para o usuário, se o sistema utilizar *quotas*.
- Testar se a conta foi criada corretamente, fazendo login no sistema e verificando se o diretório corrente é o diretório *home* do usuário, definido no arquivo `/etc/passwd`.
- O *script shell* abaixo mostra uma maneira como os comandos executados manualmente

nesta atividade poderiam ser automatizados por um administrador de sistemas:

```
#!/bin/bash

usage() {
  echo "  Usage: $0 -u USER -p PASSWORD"
  exit 1
}


if [[ $EUID -ne 0 ]]; then
  echo "  [*] Not root!" 1>&2
  exit 1
fi

while getopts ":u:p:" opt; do
  case "$opt" in
    u)
      user=${OPTARG}
      ;;
    p)
      pass=${OPTARG}
      ;;
    *)
      usage
      ;;
  esac
done

[ -z $user ] && { echo "  [*] No user?"; usage; }
[ -z $pass ] && { echo "  [*] No password?"; usage; }

if egrep "^${user}:" /etc/passwd &> /dev/null; then
  echo "  [*] User exists!"
  exit 1
fi

lastgid=$( getent group | grep -v 'nogroup' | cut -d':' -f3 | sort -n | tail -n1
)
((lastgid++))

echo "$user:x:$lastgid:" >> /etc/group
echo "$user:!::" >> /etc/gshadow

lastuid=$( getent passwd | grep -v 'nobody' | cut -d':' -f3 | sort -n | tail -n1
)
((lastuid++))

echo "$user:x:$lastuid:$lastgid:,,,:/home/$user:/bin/bash" >> /etc/passwd
```

```
salt="$( cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 8 | head -n 1 )"
hpass="$( mkpasswd -m sha-512 -S $salt -s <<< $pass )"
echo "$user:$hpass:16842:0:99999:7:::" >> /etc/shadow

cp -r /etc/skel /home/$user
chown -R ${user}.${user} /home/$user
```