

II) OPERANDO O SISTEMA:**1) Acessando o GNU/Linux**

Antes de usar o sistema pela primeira vez, deve-se procurar o administrador do sistema para que ele atualize o cadastro de usuários. Basicamente, são definidos um nome e uma senha para o novo usuário.

Para iniciar uma sessão de trabalho, deve-se oferecer o nome ou login ID, em resposta à pergunta "login:", e a senha, em resposta à pergunta "password:":

```
login: fulano
password: < A senha digitada não aparece no vídeo e o cursor permanece imóvel >
*****
* Bem Vindo ao Linux *
* As 9 horas haverá backup *
*****
```

\$ _
Por questões de segurança a senha não é mostrada no vídeo enquanto ela é digitada.
Caso o código de acesso ou a senha do usuário estiverem errados é impressa uma mensagem do tipo "Login incorrect.".

Se a senha estiver correta, aparece uma mensagem indicando a data do ultimo login, seguida de outra mensagem opcional, seguida do prompt (sinal de pronto ou espera por comandos) do sistema. A mensagem opcional exibida é chamada de "message of the day" (/etc/motd). O sinal de pronto, que geralmente é um sinal de percentagem (%) para o C shell ou um sinal de cifrão (\$) para o Bash (shell padrão), Bourne ou Korn Shell, indica que o usuário entrou no sistema e o shell está pronto para receber os comandos. O sinal de pronto do superusuário (root) é #.

Ao entrar no sistema, o Linux disponibiliza 6 terminais (tty1 a tty6) no modo texto. Para alternar entre eles, basta pressionar alt+f1 até alt+f6 (ou chvt N – onde N é o número do tty). Podemos ter seções independentes em diferentes terminais.

2) Trocando a Senha:

A senha que o usuário utiliza para acessar o sistema pode ser trocada através do comando passwd. No exemplo abaixo um usuário está logado como joao. Para trocar a senha deste usuário é necessário conhecer a senha corrente. Se esta senha não for conhecida, não será possível trocar a senha e uma mensagem de erro será exibida.

Quando a senha for digitada, ela não aparece no vídeo e o cursor não se move.

```
passwd < Comando para troca de senha >
Changing password for joao
Old password: < Informar senha atual >
New password: < Informar a nova senha >
Re-enter new password:
```

Após informar a senha corrente, será necessário digitar a nova senha duas vezes, para garantir que não houve nenhum erro de digitação.

Apenas o administrador do sistema (usuário root) pode trocar a senha de um outro usuário. Todos os outros usuários podem trocar apenas as suas próprias senhas. Portanto, se o usuário perder ou esquecer a senha, ele deverá procurar o administrador do sistema.

Por questões de segurança, as senhas do Unix seguem, conforme configurações, todas ou algumas das regras a seguir, válidas para usuários sem privilégios:

Cada senha deve ter 6 ou mais caracteres; senhas devem conter pelo menos 2 caracteres alfabéticos e um caractere numérico ou caractere especial, senhas devem diferir do código de acesso do usuário (login ID), em ordem direta, reversa ou qualquer deslocamento circular; novas senhas devem diferir das senhas antigas em pelo menos 3 caracteres. Trocas periódicas de senhas podem ser obrigatórias.

- Para encerrar a sessão utiliza-se o comando logout, exit ou teclas ctrl+d.
- Somente o root pode desligar o sistema (CLI), com: halt ou shutdown ou init 0.
- Para usuário comum desligar, deve estar ativado o suid em /sbin/halt.
- A existência do arquivo /etc/nologin impede login de usuários comuns.

3) Ambiente de Trabalho – Linha de comando:

A aparência da área de trabalho, normalmente é oferecida pelo shell (interpretador de comandos ou linguagem de controle). As áreas de trabalho mais comuns são do tipo linha de comando (CLI - Command Line Interface), ou interface gráfica (GUI – Graphic User Interface), oferecida pelo X Window.

A linha de comando, normalmente, tem o seguinte formato:

Nome-do-comando [opções] [argumentos] <enter>

Opções: Sequência de um ou mais caracteres, geralmente precedidos pelo hífen (-), que controlam a operação do comando.

Argumentos: Normalmente seguem as opções e indicam o objeto para o comando.

ls -l /etc ; ls -ld /etc ; ls -la

Os comandos acima listam o conteúdo do diretório /etc no formato longo (-l); a entrada /etc no formato longo (-ld); e os arquivos, inclusive os ocultos, no formato longo (-la) do diretório corrente.

4) Arquivos de personalização:

Cada usuário possui um diretório pessoal sob /home, para manter seus arquivos pessoais e outros arquivos, com nomes iniciados por ".", que o bash utiliza para obter o perfil definido pelo usuário, além do perfil oferecido pelo sistema em /etc/profile.

Quando o usuário inicia uma sessão *login shell*, o bash executa os comandos do arquivo /etc/profile e, após, procura no diretório pessoal do usuário os arquivos *~/.bash_profile*, *~/.bash_login* e *~/.profile* e executa os comandos do primeiro que encontrar. O arquivo /etc/profile força a execução do /etc/bash.bashrc e o *~/.profile* força a execução do *~/.bashrc*.

Quando o bash é invocado como interativo e *não login* (abre o terminal da interface gráfica) são executados os arquivos `/etc/bash.bashrc` e `~/.bashrc`.

Arquivos de personalização do usuário, em seu diretório pessoal:

- `~/.profile` : É lido pelo bash no momento do login, usado para ativar procedimentos iniciais, como criar/alterar variáveis (PATH, PS1,...). Chama `~/.bashrc`
- `~/.bashrc`: É lido para sessões non-login shell e login Shell (via `~/.profile`). Especifica aliases pessoais do usuário, alguns já fornecidos.
- `~/.bash_history`: Histórico dos comandos, para recuperá-los basta pressionar seta para cima. (!nn – Executa o comando de número nn do `.bash_history`)
- `~/.bash_logout`: Executado no momento do encerramento da sessão (login shell).

5) Edição de textos:

Para editar tabelas pode-se usar o editor **vim**.

O vim executa em dois modos: modo de comando ou modo de insert/replace.

Ao executar o vim, ele entra em modo de comando. Para passar para insert, pressione a tecla INSERT ou digite-se i. Em modo insert a tecla INSERT alterna entre INSERT e REPLACE (para inserir ou substituir dados).

Para passar de INSERT/REPLACE para o modo de comando deve-se pressionar a tecla ESCAPE.

Em modo de comando:

Digita-se ":" para indicar uma ação de saída do arquivo editado, e após:

- `:q` - Sai de arquivo não modificado sem gravar.
- `:q!` - Sai do arquivo modificado sem gravar.
- `:wq` - Grava arquivo e sai. (igual a "x").
- `:w` - Grava e permanece editado.
- `:shell` - Saída temporária para o sh, retorna ao vi com ctrl+d.
- `:X` - Criptografa o arquivo.

Mover linhas:

`:intervalo mov posição - 1,5 mov 10` (move linhas 1 a 5 p/ linha 10).

Copiar linhas:

- `yy` - Seleciona a linha
- `p` - (paste) Cola a linha abaixo do cursor
- `y2` - Seleciona a linha do cursor mais as 2 abaixo ou acima.

Apagar linhas:

- `dd` - Apaga uma linha na posição do cursor.
- `d3` - Apaga 4 linhas (do cursor mais as 3 abaixo).
- `u` - Undo (recupera linha apagada).

Dividir tela:

`: split /etc/bash.bashrc` (divide tela e edita `/etc/bash.bashrc`)
`CTRL + ww` alterna entre os dois arquivos.

Busca:

`/linux` (procura e pára na linha que contiver "linux").
`n` próxima ocorrência do texto.

Help:

`:h`

- Obter o arquivo de configuração do vim: cp /etc/vim/vimrc ~/.vimrc
- O comando **vimtutor** oferece um tutorial para praticar o vim.

6) Interpretadores de comandos mais utilizados:

sh ou Bourne shell: O shell original do UNIX, ainda utilizado, pequeno e com poucos recursos, pode ser simulado pelo bash.

bash ou Bourne Again shell: O shell GNU padrão, intuitivo e flexível, bom para iniciantes e profissionais, considerado como *superset* do Bourne Shell, ou seja, comandos do sh funcionam no bash mas o contrário nem sempre.

csh ou C shell: Sintaxe similar à programação em linguagem C.

tcsh or Turbo C shell: Superset do C shell padrão.

ksh ou Korn shell: Superset do Bourne shell, apreciado pelos que utilizam UNIX.

Obs.:

- a) O Linux é case-sensitive devendo-se observar que A difere de a.
- b) No Linux o shell padrão é o bash (bourne again shell).
- c) O usuário pode optar por um shell entre os listados no arquivo /etc/shells, podendo ainda mudar o shell padrão do login, através do comando chsh:

chsh -s /bin/dash - Muda o shell padrão para /bin/dash (em /etc/passwd), no próximo login.

- d) Pode-se criar uma sessão temporária de um outro shell, digitando-se o nome do shell e retornar ao anterior pressionando-se simultaneamente ctrl+d.
- e) Observe que as entradas "/bin/*sh*" indicam os shell instalados.
- f) Para saber o shell em uso, digite echo \$SHELL ou verifique a linha de sua conta no arquivo /etc/passwd, ou digite env e procure a variável SHELL.
- g) Comandos podem ser inseridos em sequência, como parte de uma mesma linha, separando-os com os caracteres ";", "&&" e "||", que podem aparecer várias vezes na linha.

6.1) Encadeamento de comandos:

Pode-se usar os caracteres abaixo, entre comandos na mesma linha para:

- ;** - Executar os comandos um após o outro, de forma independente.
- &&** - O segundo comando só é executado se o primeiro funcionar.
- ||** - O segundo comando só é executado se o primeiro falhar.

Algumas teclas importantes:

Ctrl+C	-Termina o programa em execução e retorna ao prompt.
Ctrl+D	-Abandona a sessão corrente do Shell (igual a exit ou logout).
Ctrl+L	-Limpa a tela (igual a clear).
Ctrl+z	-Suspende a execução de um programa.(fg retoma)
Seta acima	-Edita o arquivo de histórico dos comandos.
Seta abaixo	-Retorno de seta acima.
Tab	-Completa o comando ou nome do arquivo, caso haja possibilidade.
Tab tab	-Mostra as possibilidades de nomes para completar.
Shift+PgUp	-Mostra o buffer com as últimas 13 telas, shift+PgDown retorna.
Ctrl+alt+f1	-Muda de GUI para CLI. Ctrl+alt+f7 (ou chvt 7) retorna para GUI obs.: No Debian 9 (stretch) : Ctrl+alt+f1 para desbloquear a sessão (timeout). Ctrl+alt+f3 para passar de GUI para CLI (tty3). Ctrl+alt+f2 para passar de CLI para GUI.

7) Diretórios e arquivos:

Diretórios são arquivos que contém a lista de arquivos e de outros diretórios criados dentro dos diretórios, seguindo a estrutura de uma árvore invertida, onde o nível mais alto é chamado de diretório raiz, identificado por uma /, a partir do qual derivam todos os outros diretórios da árvore. A localização de um arquivo ou diretório pode ser informada através de dois modos:

- Caminho absoluto:** Partindo do diretório raiz, o caminho é iniciado por uma / e relaciona todos os diretórios do caminho, separados por /, até chegar ao arquivo ou diretório desejado.
- Caminho relativo:** Parte no diretório corrente. O caminho pode ser iniciado por ./ ou pelo nome de um diretório registrado no diretório corrente ou por .. (Aponta para o diretório pai do diretório corrente).

Para comandos em geral não é necessário informar o caminho, pois o sistema procurará o comando nos diretórios relacionados na variável PATH (echo \$PATH ou env para ver os caminhos de busca.)

Arquivos são mecanismos de abstração que fornecem uma forma de armazenar e recuperar informações. Quando um arquivo é criado por um processo ou comando, ele deve receber um nome, que é acoplado ao caminho, pelo qual será conhecido no sistema. Os nomes dos arquivos podem ter nenhuma, uma ou mais extensões (como prog.c.z).

Existem 3 formas para executar um arquivo de comandos (shell scripts):

```
sh < arqcom ou
sh arqcom ou
chmod a+x arqcom && ./arqcom
```

Para executar programas compilados basta digitar seu caminho e/ou nome.
Lembrando que o ./ antes do nome do arquivo, indica que o arquivo está no diretório corrente e é necessário.

7.1) Estrutura padrão dos diretórios (stndfs):

O sistema de arquivos padrão do Linux (STNDFS) busca definir uma padronização para a organização da árvore de diretórios, visando facilitar desenvolvimento, porte e administração de máquinas Linux, uma vez que tudo estará localizado em caminhos conhecidos. Podemos nos posicionar em um diretório com o comando **cd caminho**.

- | | |
|--------|---|
| /bin | - Arquivos executáveis (binários) de comandos essenciais pertencentes ao sistema e que são usados com freqüência pelo administrador e por usuários. |
| /sbin | - Similar ao /bin com comandos não destinados aos usuários comuns, apesar de poderem ser utilizados por estes. |
| /boot | - Arquivos boot utilizados pelo gerenciador de inicialização LILO ou GRUB (Kernel já compilado) |
| /dev | - Arquivos de dispositivos de entrada/saída |
| /etc | - Arquivos de configuração do sistema da máquina local. |
| /home | - Diretório onde são definidos os diretórios dos usuários. |
| /lib | - Bibliotecas necessárias para execução dos binários residentes em /bin e /sbin. |
| /media | - Ponto de montagem temporária de partições (/media/cdrom; /media/floppy,...). |
| /root | - Diretório pessoal do superusuário (root). |
| /tmp | - Arquivos temporários gerados por alguns utilitários. |
| /usr | - Arquivos compartilhados pelos usuários ou pela rede, montado como somente leitura. Esses arquivos normalmente são instalados pelo distribuidor Linux. |
| /var | - Arquivos que são alterados durante a execução do sistema, sendo específico de cada sistema, como arquivos de log (em /var/log/) e spool (/em var/spool/). |
| /proc | - Sistema de arquivos ilusório que é criado e mantido em memória pelo kernel, usado para disponibilizar informações sobre o sistema. |

O conteúdo de diretórios, mostrado pelo comando **ls** utiliza cores para diferenciar as entradas:

Azul	- diretório	Ciano	- link
Vermelho	- Arquivo comprimido	Rosa	- Imagens
Cinza	- Arquivo normal	Verde	- arquivo executável
Amarelo	- Arquivo de dispositivo	Vermelho piscante – link quebrado	

7.2) Tipos de arquivos:

ls Mostra a relação dos arquivos em um diretório.

Sintaxe:

ls [opções] [arquivo]

ls -l

Lista os arquivos do diretório, cada um em uma linha com várias informações, como:

-rwxr-xr-x	6	root	root	4096	dec 28	2000	carta	
1	2	3	4	5	6	7	←	Colunas

Coluna 1:

A primeira posição identifica o tipo da entrada:

-	=	arquivo
d	=	diretório
l	=	link simbólico
c	=	arquivo de dispositivo caractere
b	=	arquivo de dispositivo de bloco

Posições 2 a 4 : Indicam permissões de leitura, gravação e execução para o dono do arquivo que, se ativadas, contém r,w ou x respectivamente, se não ativadas contém -.

Posições 5 a 7 : O mesmo para usuários pertencentes ao grupo do dono.

Posições 8 a 10 : O mesmo para qualquer outro usuário.

Coluna 2: Quantidade de ligações físicas +1 p/ o arquivo, ou quantidade de subdiretórios +2 se for diretório.

Coluna 3: Nome do dono do arquivo.

Coluna 4: Nome do grupo do dono.

Coluna 5: Tamanho do arquivo (bytes) ou MB se ls -h

Coluna 6: Dia mês e ano (se do ano anterior) ou dia e mês (se do ano corrente)

Coluna 7: Nome do arquivo.

7.3) Mudando permissões de arquivos e diretórios:

chmod - Altera as permissões, de cada arquivo fornecido, de acordo com o modo, através de uma representação simbólica ou de um número octal que representa um padrão de bits para as novas permissões.

Formato simbólico:

chmod [u g o a] { + - = } { r w x s t } arquivo

O comando simbólico é seguido por uma ou mais letras (ugoa):

- “u” -para indicar mudança para o dono do arquivo (user).
- “g” -para o grupo do dono,
- “o” -para outros usuários e
- “a” -para todos os anteriores.

Se não for digitada nenhuma letra será assumido “a”.

O sinal “+” adiciona uma permissão, o “-“ retira e o “=” substitui as permissões existentes pelas informadas.

As letras ‘rwxst’ indicam as novas permissões para os usuários afetados: (r) ler, (w) gravar, (x) executar o arquivo (ou acesso para diretórios), (s) seleciona o usuário (suid) ou identificação do grupo (sgid) na execução de programas, (t) sticky bit em diretórios indica permissão de deletar um arquivo somente para o dono do arquivo ou do diretório onde encontra-se o arquivo (caso típico para diretório /tmp), e quando aplicado em arquivos executáveis originalmente indica que o programa deve ser mantido na memória quando terminar sua execução, mas existem outros usos.

Para diretórios, a permissão "r" permite ler o conteúdo do diretório, "w" gravar e "x" permite entrar no diretório via comando "cd".

Sintaxe:

chmod [opções] modo arquivo

Opções referem-se à recursividade (-R), informar ocorrências (-c) ou não (-f).

Ex.: chmod u+r+w+x,g+r+w-x,o+r-w-x carta ou
chmod u+wx,g=rw,o=r carta

Dará todas as permissões para o dono, leitura e gravação para o grupo e somente leitura para outros usuários (retirando as demais permissões que existirem para outros).

Formato numérico octal:

O formato numérico octal é composto por três dígitos cujos bits afetam rwx (e um quarto dígito especial que afeta os bits s e t que será visto adiante): Um que afeta o user, outro o grupo e o último outros. Números não utilizados são considerados como 0, assim chmod 3 arq, será considerado como chmod 003 arq.

Ex.: chmod 765 arq-1

No exemplo acima o 7 altera as permissões do user, o 6 do grupo e o 5 de outros.

Convertendo os dígitos octal em binário temos: $7_{(8)} = 111_{(2)}$; $6_{(8)} = 110_{(2)}$; $5_{(8)} = 101_{(2)}$

Os bits referem-se a permissões de leitura, gravação e execução (rwx) que, quando ligados concedem a permissão e quando desligados retiram a permissão. Da esquerda para a direita: 1º bit para r, 2º bit para w e 3º bit para x.

Portanto chmod 765 arq-1, concede leitura, gravação e execução para o user; leitura e gravação para o grupo; e leitura e execução para outros. O comando será interpretado como chmod 0765 arq-1 pois o quarto dígito para suid, sgid e sticky bit não foi informado.

No formato numérico podemos utilizar um quarto dígito octal (1º dígito à esquerda) que irá alterar os bits suid (s para o user), sgid (s para o grupo) e sticky bit (t), assim:

- 0 → Desativa os bits suid, sgid e sticky bit (t).
- 1 → Ativa o bit t (sticky) e desativa suid e sgid.
- 2 → Ativa o sgid e desativa suid e sticky (t).
- 4 → Ativa o suid e desativa sgid e sticky (t).

Pode-se reunir essas permissões, utilizando o resultado da soma desses números: 3 representa as permissões de 1 e 2; 5 as permissões de 1 e 4; 6 as permissões de 2 e 4; e 7 reúne as permissões de 1, 2 e 4.

- Obs.:
- a) Ligar suid sem permissão de execução para o dono mostra suid = S e não s.
 - b) Ligar sgid sem permissão de execução para o grupo mostra sgid = S e não s.
 - c) Ligar o bit t (chmod +t dir), sem permissão o+x no dir, mostra T e não t.
 - d) chattr ativa/desativa outros atributos, alguns dos quais somente como root.

- e) o = rw-
- Não entra no diretório.
 - Não avança no caminho.
 - Não cria arquivos no diretório.
 - Não lista adequadamente conteúdo de diretório.
 - Não edita arquivos o=rw- no diretório.
 - Não deleta arquivos o=rw- no diretório.
- f) o = - -x
- Entra no diretório.
 - Avança no caminho.
 - Não cria arquivos no diretório.
 - Não lista o diretório
 - Edita/altera arquivos com o=rw- (digitar nome inteiro).
 - Não deleta arquivos o=rw- no diretório.
- g) o=rwx
- Apaga arquivo com o= - - - dentro do diretório.
 - Apaga diretório vazio com o=---/rw- dentro do diretório.

Permissões no esquema ACL (Access Control List) podem ser visualizadas ou incluídas/excluídas com os comandos **getfacl** e **setfacl** para arquivos e diretórios.

Permissão suid:

A permissão setuid é desativada para scripts por motivos de segurança, **funcionando apenas para programas compilados**. Portanto o script abaixo (**teste-uid.sh**) só funcionará se executado pelo root, ou se for chamado de um programa compilado com suid ligado.

Para testar isso, crie no **diretório do usuário comum**, o script abaixo como **root**:

```
$ su
Senha:
# vim teste-uid.sh # e insira as linhas abaixo.

#!/bin/bash
if [[ $EUID -ne 0 ]]; then      # testa executor no formato #!/bin/bash "
    echo "somente o super user pode executar este script"
## exit 23                      # exit comentado para atingir cat /etc/passwd
fi
###                                # == testa executor com modo alternativo ==
## if [ "$(id -u)" != "0" ]; then   # formato #!/bin/sh (/bin/dash) e #!/bin/bash
##   echo "somente o root pode executar-me"
# fi
cat /etc/shadow                  # arquivo /etc/shadow só pode ser lido pelo root
if [[ $? -eq 0 ]]; then           # testa Return Code (RC) do comando
    echo "P A S S O U";
    exit 0
else echo "F A L H O U"
    exit 1
fi
## -- fim do script
```

Teste o script como root para certificar-se que funciona, devendo mostrar na tela o conteúdo do arquivo /etc/shadow e a mensagem "P A S S O U":

Agora ative as permissões **suid** e **execução** para outros e execute-o como um usuário comum:

```
# chmod 4755 teste-uid.sh  
# exit  
$ ./teste-uid.sh
```

O comando **cat /etc/shadow** terá permissão negada e mostrará as mensagens do script indicando que o usuário não é o root, pois o **suid** não é reconhecido para scripts.

Agora podemos mudar o dono e grupo do script para o usuário comum e retirar o **suid**.

```
$ su
```

Senha:

```
# chown aluno:aluno teste-uid.sh # quando muda o dono do script, o suid é retirado.
```

Para um usuário comum executar o script é necessário ativá-lo a partir de um programa compilado (dono é root), por exemplo em linguagem c, com setuid ativado (u+s), assim: Ainda como root, crie o programa abaixo, que deve estar no diretório do usuário que vai executá-lo, juntamente com o script:

```
# vim teste-suid.c
```

```
#include <stdio.h>  
#include <unistd.h>  
#include <stdlib.h>  
int main() {  
    setuid(0) ;  
    system("./teste-uid.sh") ;  
}
```

Salve o arquivo (teste-suid.c), compile e ative o setuid:

```
# gcc -o teste-suid teste-suid.c  
# chmod u+s teste-suid  
# exit # volta a ser usuário comum  
$ ./teste-suid # executa o programa como usuário comum
```

Aparecerá na tela a saída do comando **cat /etc/shadow** como se executado pelo root.

7.4) Mudando dono/grupo de arquivos e diretórios:

O dono e grupo do dono de arquivos e diretórios só podem ser modificados pelo root, com os comandos **chown** e **chgrp**, a opção **-R** altera recursivamente:

```
chown [-R] novodono:novogruo arq1 # altera o dono e o grupo  
chown [-R] novodono dir1 # altera só o dono  
chgrp [-R] novogruo arqx # altera só o grupo
```

7.5) Permissões geradas por padrão:

As permissões de acesso padrão são geradas a partir do valor da umask (Máscara do usuário) na criação de arquivos e diretórios do usuário.

umask:

Para gerar as permissões (mode) de diretórios e arquivos criados, o sistema utiliza os valores em octal da umask e dois valores base: 0777 para diretórios e 0666 para arquivos.

Observe que o **0** à esquerda de 0777 e 0666 é alinhado com o quarto dígito à esquerda da umask, e atua sobre os bits **suid**, **sgid** e **sticky bit (t)** que nunca são ligados automaticamente - umask 0022 - e será ignorado nos exemplos abaixo.

Esses valores base em conjunto com a umask definem as permissões a serem concedidas automaticamente, da seguinte forma:

- a) Aplica-se um **not** no valor da umask.
- b) Aplica-se um **and** entre o valor base e o valor obtido em a).

ex.: umask 022

Para diretórios:

not 022 (valor obtido = 755)
and 755 , 777 (valor obtido = 755 ==> **rwxr-xr-x**)

Para arquivos:

not 022 (valor obtido = 755)
and 755 , 666 (valor obtido = 644 ==> **rw-r--r--**)

Cada usuário pode (e deve) criar sua própria umask no arquivo **~/.bashrc**. O dono e/ou o grupo do dono de um arquivo podem ser modificados pelos comandos **chown** e **chgrp** respectivamente, desde que seja feito pelo root.

8) Redirecionando a entrada e saída padrão:

Muitos comandos utilizam a entrada padrão (stdin) e a saída padrão (stdout), normalmente teclado e monitor, que são os dispositivos de onde recebem dados e onde colocam o resultado do comando. Por exemplo a entrada padrão para o comando ls é o diretório atual e a saída padrão é o monitor de vídeo.

Isso pode ser mudado, redirecionando-se a entrada e/ou a saída padrão, utilizando os caracteres < (menor) para redirecionar a entrada (nem sempre necessário), > (maior) para redirecionar a saída e >> (duplo maior) para redirecionar a saída acrescentando.

Exs.: ls -l > meu-dir # Grava saída no arquivo meu-dir que será criado ou substituído.
ls -l >> meu-dir # Acrescenta a listagem do diretório no arquivo meu-dir.

Existe ainda a saída padrão de erros (stderr), que direcionada para /dev/null evita a poluição do terminal com avisos de erro (e não utiliza espaço em disco para os erros).

Ex.: find / -iname find 2> /dev/null

9) Metacaracteres e pipes:

9.1) Metacaracteres

O shell provê recursos que permitem facilmente especificar múltiplos nomes de arquivos como argumentos de comandos.

Isto é conseguido através de caracteres especiais que podem fazer referência a vários arquivos a partir de uma só especificação. A estes caracteres dá-se o nome de metacaracteres; conhecidos internacionalmente por wild-cards ou curingas.

<u>Metacaractere</u>	<u>Significado</u>
?	Substitui por 1 caractere
*	Substitui por 0 a n caracteres (no início do nome não emparelha .)
[Inicia grupo de caracteres
]	Termina grupo de caracteres
-	Indica faixa de caracteres

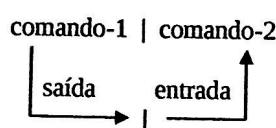
Para retirar a funcionalidade de metacaracteres utiliza-se o caractere \ (barra à esquerda).

Exemplos:

- | | |
|-----------------------|---|
| ls -l arq* | - Lista as entradas que iniciam com arq |
| ls -l arq* | - Lista o arquivo de nome arq* |
| ls -l arq? | - Lista as entradas que iniciam com arq + um caractere qualquer. |
| ls -l arq[123] | - Lista as entradas arq1, arq2 e arq3 |
| ls -l arq[1-3] | - Lista as entradas arq1, arq2 e arq3. |
| ls -l arq[12] [12] | - Lista as entradas arq11, arq12, arq21 e arq22. |
| ls -l arq[5-7][0-9] | - Lista as entradas arq seguida por um número entre 50 e 79. |
| ls -l [A-Z][A-Z][A-Z] | - Lista as entradas com três letras maiúsculas no nome |
| ls -l [A-Z,a-z]*[0-9] | - Lista as entradas com nome iniciado por letra maiúscula ou minúscula e terminado por um número entre 0 e 9. |
| ls -l ???? | - Lista as entradas cujo nome tenha 4 caracteres quaisquer. |

9.2) Pipes:

O pipe é utilizado quando desejamos que a saída de um comando seja a entrada de outro.



Por exemplo: ls /etc | more

Diz o seguinte: "mostre o conteúdo do diretório /etc página a página". A saída do comando ls é utilizada como entrada para o comando more.

Podemos também combinar redirecionamento de E/S com o operador pipe. Em vez de olharmos a lista no terminal, podemos redirecioná-la para um arquivo.

ls -l | grep -e drwx > arquivo

Grava em arquivo as linhas das entradas do diretório que contiverem a sequência drwx.

.....

10) Manipulando arquivos e diretórios:**10.1 Nomes de arquivos.**

O nome de um arquivo é formado por um conjunto de até 255 caracteres que pode incluir praticamente qualquer caractere. Entretanto, deve-se utilizar o bom senso na hora de criar o nome dos arquivos (evitar caracteres como “ * ? | > ”). Não existe uma extensão obrigatória (explícita) como no MS-DOS. O tipo de arquivo é identificado pelo sistema por um conjunto de caracteres específicos no início do arquivo (magic number) e não pelo seu nome.

O sistema de arquivos no Linux possui as seguintes características:

- Dispositivos de E/S aparecem como arquivos especiais.
- Não podem existir arquivos com nomes iguais dentro do mesmo diretório.
- Cada diretório tem, no mínimo, as entradas “.” (ele mesmo) e “..” (diretório pai).
- Cada usuário tem o seu diretório pessoal, cujo nome é o nome de login (/home/usuario).
- A cada momento existe um diretório corrente que pode ser alterado pelo usuário.
- Ao iniciar a sessão de trabalho, o diretório corrente é o diretório pessoal do usuário.
- Se o nome do arquivo inicia com “/”, o caminho inicia no diretório raiz, caso contrário inicia no diretório corrente.
- Os nomes no caminho são separados por “/”; nome “..” permite subir na árvore.
- Não são permitidos brancos dentro do nome (se eles existirem, é necessário colocar o nome do arquivo sempre entre aspas).
- Todos os arquivos cujos nomes começam com “.” é considerado um arquivo oculto. (os diretórios “.” e “..” são ocultos). Para listá-los é necessário comandar ls -a.
- O caractere ~ pode ser usado para substituir o nome do diretório pessoal do usuário. E, ~user indica o diretório pessoal do usuário “user”.
- O Linux é “case sensitive”, o que quer dizer que letras maiúsculas são diferentes de letras minúsculas.

11) Variáveis de ambiente:

Variáveis armazenam informações geradas pelo BASH ou por um usuário e são representadas no ambiente padrão do BASH. Seu conteúdo representa parâmetros necessários ao funcionamento do sistema e de programas diversos:

- | | |
|------------------------------------|--------------------------|
| - Criar uma variável com um valor: | V1=”valor V1” ou V2=123. |
| - Acessar o valor da variável: | echo \$V1. |
| - Estender o conteúdo da variável: | PATH=\$PATH:~/bin |
| - Listar as variáveis definidas: | env ou set ou printenv. |
| - Inserir variável no ambiente: | export nome-da-variável. |
| - Retirar variável do ambiente: | unset nome-da-variável. |

11.1 Algumas variáveis do GNU/Linux:

PPID	-O ID do processo-pai do Shell.
PWD	-O diretório corrente do usuário.
UID	-O UID do usuário.
USER	-O nome do usuário.
SECONDS	-O número de segundos decorridos desde que o Shell atual foi iniciado.
PATH	-Relação dos diretórios (separados por ":") onde procurar comandos.
HOME	-Caminho absoluto do diretório pessoal do usuário.
PS1	-Parâmetros para definir o formato do prompt do usuário.
RANDOM	-Fornece um número diferente (0 a 32767) cada vez que é chamada.
TMOUT	-Especifica a quantidade de segundos que uma sessão shell pode ficar inativa, após o que será feito logout automático.

12) Manuais de comandos:

12.1 man Fornece informações sobre comandos, arquivos, etc.

Sintaxe:

man [seção] tópico
man -K assunto

Parâmetros:

seção: indica a seção do manual a procurar. Serve para procurar um determinado tópico em uma seção específica do manual, do contrário o man apresenta o tópico encontrado na primeira seção onde ele existir:

As 8 seções do manual:

- 1 – comandos.
- 2 - chamadas ao sistema.
- 3 - linguagem de programação.
- 4 - dispositivos.
- 5 - arquivos do sistema.
- 6 - jogos.
- 7 - informações gerais.
- 8 - administração do sistema.

tópico: Indica qual tópico procurar. O tópico deve ser o nome de um comando, chamada de sistema, etc., exatamente como está no manual, do contrário o man não conseguirá encontrar as páginas correspondentes do manual.

assunto: Na segunda forma de uso do man, o assunto especifica a palavra que deve ser procurada nas várias páginas do man. O comando apresenta na tela o primeiro tópico encontrado e, ao sair desse tópico (q), sugere os tópicos seguintes onde a palavra for encontrada, solicitando a opção do usuário: Ver o tópico (enter); pular o tópico (ctrl+d) ou abandonar a pesquisa (ctrl+c).

ISO200 (SOII) - MUDANDO PERMISSÕES - EXERCÍCIOS

Resolva os itens abaixo com um único comando chmod, utilizando o modelo numérico para alterar as permissões, como indicadas.

Crie os arquivos a modificar permissões executando: touch nome-do arquivo.

1. Arquivo: arq-ep-1

Permissões:

Dono: leitura e gravação; grupo: gravação; outros: leitura.
Bits: suid, sgid e sticky : desligados.

2. Arquivo: arq-ep-2

Permissões:

Dono: leitura ; grupo: gravação e execução; outros: gravação.
Bits suid e sticky: ligados.

3. Arquivo: arq-ep-3

Permissões:

Dono: gravação; grupo: leitura e execução; outros: leitura e execução.
Bits sgid e sticky ligados.

4. Arquivo: arq-ep-4

Permissões:

Dono: leitura, gravação e execução; grupo: gravação; outros: leitura.
Bit suid ligado..

5. Arquivo: arq-ep-5

Permissões:

Dono: leitura e execução; grupo: leitura; outros: nenhuma permissão.
Bit sgid ligado.