Linux: Utilização

1. Conectando-se ao sistema: *login*

O acesso a um sistema Linux depende da identificação de um nome de usuário e da senha correspondente, num procedimento de ***login***. Enquanto o usuário ***root*** tem **direito total sobre todos os recursos do sistema**, outros usuários têm acesso para **leitura** de grande parte dos arquivos, podem **executar** a maioria das aplicações, e têm direito de escrita somente sobre seus respectivos diretórios de trabalho (*/home/*login**) e área temporária (*/tmp*).

Caso a configuração do ambiente gráfico tenha sido realizada no processo de instalação, a opção de ***login*** através de uma **interface gráfica** pode ter sido selecionada automaticamente. Independentemente de o modo gráfico estar configurado, é possível executar-se o procedimento de ***login*** em modo **texto**.

1.1 *Login em modo texto*

No Linux, a combinação de teclas <ctrl> <Fn> (n={1,2,.., 6}) normalmente dá acesso a 6 ***terminais virutuais*** onde pode-se executar o procedimento de ***login*** numa interface de texto com acesso direto a um ***shell***. O terminal selecionado por <ctrl><F7> normalmente está associado à interface gráfica.

Um *shell* é um programa **interpretador de comandos**, que tem a finalidade de controlar a ativação e manipulação de programas determinados pelo usuário. Comandos podem ser passados para o *shell* de maneira **interativa** ou através de **programas** (*scripts*).

Diversas versões de *shell* podem estar instaladas num sistema, como listado no arquivo ***/etc/shells***. Os mais comuns são: **sh**, **bash**, **csh**, **ksh**, **tcsh** e **zsh**. As diferenças entre esses programas podem ser notadas tanto no tratamento dos comandos interativos quanto na "linguagem" usada para escrever os *scripts***.**

Variáveis de ambiente são utilizadas em *shells* para armazenar informações úteis sobre o usuário e suas preferências de comandos e outras configurações. Ex: HOME, SHELL, USER, PATH, etc. A sintaxe para suas configurações depende do *shell* utilizado. No ***bash***, é feita como segue:

bash: PATH=$PATH:$HOME/bin; export PATH *ou*
export PATH=$PATH:$HOME/bin

O comando ***export*** faz com que o valor de uma variável de ambiente seja visível também para os programas iniciados a partir do *shell* corrente. Uma lista completa das variáveis de ambiente definidas numa sessão de *shell* pode ser exibida com o comando ***set***. O comando ***echo***, que exibe uma linha de texto, pode também ser usado para verificar o conteúdo de variáveis de ambiente.

set, ***unset***, ***export***: funções do *shell* para ler, remover e exportar variáveis de ambiente
echo: exibe uma linha de texto. Ex: *echo teste*, *echo $USER*

Comandos do terminal permitem o ajuste de aspectos da sessão do *shell* corrente, incluindo a configuração da interface do terminal de entrada e saída de dados:

stty: exibe e ajusta configurações do terminal. Ex: *stty –a*, *stty erase ^H*
clear: limpa a tela do terminal

Diversos utilitários e recursos oferecem facilidades para a ativação de programas e comandos:

history: manipula a lista interna de comandos realizados no *shell*. Enquanto a variável de ambiente HISTSIZE define o tamanho do histórico guardado, HISTFILE e HISTFILESIZE definem o nome e o tamanho do arquivo onde as informações são salvas entre sessões.
!! : repete o último comando digitado
!num : repete o comando número num do histórico
!cmd : repete o último comando iniciado com a palavra ***cmd***.
<Ctrl> <r>: faz uma busca reversa na lista com os últimos comandos executados

alias: exibe ou ajusta *apelidos* para comandos. **unalias** os remove. Ex: *alias rm='rm –i'*

Arquivos específicos presentes na área do usuário e no diretório de configurações do sistema permitem a configuração das sessões de utilização de um *shell*. Para o *shell* ***bash***, os seguintes arquivos de configuração são relevantes.

Para um *shell* de *login*:

/etc/profile, ~/.bash_profile, ~/.bash_login, e ~/.profile

Quando esse *shell* de *login* termina, são executados os comandos em ~/.bash_logout

Para um *shell* interativo:

~/.bashrc

Comandos úteis nos arquivos de configuração incluem: ***set***, ***alias***, ***stty***, etc.

Uma sessão de *login* em modo texto é encerrada com o comando *logout*. A combinação de teclas <**Ctrl**> <**D**>, normalmente definida como **fim de arquivo** (EOF) também pode comumente ser usada para encerrar uma sessão de *login*. ***Exit*** termina uma seqüência de comandos num *shell* ou encerra uma sessão de *login*.

logout: comando interno do *shell* para encerrar uma sessão num computador
exit: função interna do *shell* para encerrar um nível na seqüência de operações sendo executadas ou um *shell* interativo. No *shell* de *login* tem o mesmo efeito de *logout*

1.2 *Login via Interface gráfica*

Efetuando-se o *login* numa interface gráfica, o usuário passa a interagir com o sistema Linux através de um ambiente de janelas. Entre as opções de configuração do ambiente gráfico, as mais comuns são o ***KDE***, o ***GNOME*** e o ***Window Maker***, que podem ser selecionados e configurados pelo usuário.

Alguns aspectos de configuração do ambiente de trabalho com a interface KDE3, padrão no sistema Conectiva 8.0, são apresentados a seguir.

Barra de tarefas:

- **Centro de Controle**: ferramenta para configurações gerais do sistema. Permite o ajuste de aspectos da área de trabalho, dos periféricos, do navegador de arquivos (konqueror) e possui atalhos para diversos outros utilitários de configuração.
- **konsole**: abre uma janela com o interpretador de comandos padrão (*shell*).
- **home**: ativa o gerenciador de arquivos *konqueror* com a área de trabalho do usuário
- **Área de trabalho**: permite acesso às outras áreas de trabalho e suas janelas.
- **Iniciar Aplicação**: ativador de aplicativos
  - **Preferências**: gerenciador de software e hardware
  - **Sistema**: Utilitários de administração

Ícones dos dispositivos:

- **CDROM**: abre gerenciador de arquivos no diretório ***/mnt/cdrom***, montado automaticamente
- **floppy**: abre gerenciador de arquivos no diretório ***/mnt/floppy***, montado automaticamente

Ícones de atalho:

- **Lixo**: acesso via gerenciador de arquivos ao diretório da lixeira (***~/Desktop/Lixo***)
- **Gerenciador de Pacotes** (**kpackage**): interface gráfica para o gerenciamento de pacotes **rpm** (aplicativos agrupados para instalação).
- **Synaptic - Atualizador de Pacotes**: interface gráfica para a instalação e atualização de pacotes e do sistema.

Logout: Na *barra de tarefas*, *Iniciar Aplicação*, o botão **Sair** permite encerrar uma sessão de ***login***.:

< > *Acessar como outro usuário*
< > Desligar o comutador
< > Reiniciar o computador

2. Obtendo ajuda

Informações sobre o sistema, seus comandos e arquivos de configuração são comumente instaladas num sistema Unix. Digitando-se comandos diretamente em um interpretador de comandos (*shell*), é possível acessar as suas funcionalidades.

man: comando de acesso às páginas do manual *on-line* do sistema.

Seções: (1) Comandos do usuário, (2) Chamadas do sistema, (3) Funções da biblioteca C, (4) Dispositivos e interfaces de rede, (5) Formatos de arquivos, (6) Jogos e demonstrativos, (7) Ambiente de trabalho, tabelas e macros *troff*, (8) Manutenção do sistema.

Ex: *man man*, *man 2 sleep*, *man -a passwd*, *man -k ...*

Páginas do manual são normalmente armazenadas como arquivos formatados para exibição com comandos *roff* e compactados. O arquivo ***/etc/man.config*** (ou ***/usr/lib/man.conf***) contém informações sobre os diretórios que contém páginas de manual, o que também pode ser especificado pela variável de ambiente MANPATH.

whatis: apresenta a descrição de comandos. Ex: *whatis ls*
apropos: mostra seções e páginas do manual que contém referências a um comando, pesquisando a base de dados do comando *whatis*
makewhatis: cria a base de dados para os comandos *whatis* e *apropos*
whereis: fornece a localização de um comando executável. Ex: *whereis man*
which: apresenta o nome (caminho) completo de um comando. Ex: *which ls*

3. Acesso e identificação dos usuários

Uma vez conectado ao sistema, comandos podem ser utilizados para iniciar ou encerrar uma sessão e para identificação do usuário e de outros usuários conectados ao sistema.

login: permite que um usuário abra uma sessão num computador

passwd: altera a senha de um usuário
yppasswd: altera a senha na base de dados do NIS (*Network Information Service*)
smbpasswd: altera a senha na base de dados para autenticação via ***samba***, que implementa o protocolo SMB **(***Server Message Block*) comumente chamado de **LanManager** ou **NetBIOS**.

su: substitui a identidade do usuário corrente. Sem parâmetros, refere-se ao usuário *root*. Ex: *su*, *su – fulano*, *su –c comando*.
sudo: permite executar um comando como outro usuário. O arquivo **/etc/sudoers** contém uma relação dos usuários autorizados a executar comandos específicos.

id: informa o número de identificação associado à conta do usuário
whoami: mostra o *username* efetivo do usuário corrente
groups: informa os grupos a que pertence o usuário corrente
who: mostra quem está usando o sistema

w: mostra informações sobre os usuários conectados ao sistema
users: mostra uma lista compacta dos usuários conectados ao sistema
last: mostra informações sobre o *login* e o *logout* dos usuários e terminais utilizados. Ex: *last*, *last root*
finger: programa para pesquisa de informações sobre usuários. Ex: *finger fulano*, *finger fulano@host.dom* (depende de um *servidor finger* estar ativo em *host.dom*)

Arquivos de informação sobre o usuário (consultados pelo comando finger):
~/.plan:
~/.project:

4. Informações do sistema

Além das informações sobre usuários, diversos comandos fornecem informações sobre o sistema operacional, sua configuração e sobre a configuração de dispositivos.

uname: exibe informações sobre o sistema. Ex: *uname –a*
uptime: informa há quanto tempo o sistema está sendo executado
hostname: exibe ou ajusta o nome do computador
domainname: informa ou ajusta o nome de domínio NIS/YP do sistema
date: informa ou ajusta a data e o horário do sistema
hwclock: consulta ou ajusta o relógio do *harware*. Pode ser relacionado com o relógio do sistema e vice-versa.
dmesg: exibe ou controla o buffer circular de mensagens do *kernel*, tipicamente as mensagens do *boot*. Ex: *dmesg | more*

5. Organização do sistema de arquivos Unix

O sistema de arquivos em ambientes Unix é baseado numa única estrutura hierárquica de diretórios. Num console modo texto, interagindo diretamente com o interpretador de comandos, o comando ***ls*** (***ls –l /***) é utilizado para listar os conteúdos dos diretórios. Entre os diretórios principais, podem ser destacados:

- / – diretório raiz, concentra toda a estrutura de diretórios
- **/etc** – arquivos de configuração
- **/bin** – utilitários de uso geral
- **/sbin** – utilitários de administração, alguns com uso restrito ao usuário ***root***.
- **/dev** – arquivos especiais que representam dispositivos, criados com o comando *mknod* (/dev/MAKEDEV).
- **/usr** – utilitários não essenciais
- **/usr/bin**
- **/usr/sbin**
- **/usr/lib -** bibliotecas de programas (*linkad*as com programas de usuário: lib*.a)
- **/usr/include** arquivos de definições e protótipos de funções (*header files*: *.h)
- **/usr/man** – diretório onde comumente são armazenadas as páginas de manual
- **/home** – área de trabalho dos usuários
- **/var** – diretório para área de *spool* de impressão, e-mails e arquivos de *log*

- **/boot** – arquivos para iniciação do sistema (*boot*) e configurações
- **/mnt** – diretório onde comumente são *montados* sistemas de arquivos de dispositivos removíveis (*cdrom* e *floppy*)
- **/tmp** – armazenamento temporário
- **/proc** – sistema de arquivos em memória com informações sobre o sistema e seus processos
- **/opt** – aplicativos não fornecidos com o sistema

6. Manipulação de arquivos e diretórios

ls: mostra o conteúdo de diretórios. Ex: *ls –l /home*
pwd: informa o nome do diretório corrente. Ex: *pwd*
cd: muda o diretório de trabalho. Ex: *cd /etc*, *cd ..*, *cd ~/www*, *cd ../../local*
cp: copia arquivos. Ex: *cp /tmp/arq .*, *cp –r dir1 ../dir2*, *cp arq1 arq2*
mv: move ou renomeia arquivos ou diretórios. Ex: *mv /tmp/arq .*, *mv arq novo_nome*
mkdir: cria diretórios. Ex: *mkdir dir*, *mkdir www pub tmp*
rm: remove arquivos. Ex: *rm arq*, *rm –i arq*, *rm –f arq*, *rm –rf diret*
rmdir: remove diretórios. Ex: *rmdir diret*, *rmdir dir1 dir2 /tmp/dir3*
ln: cria um *link* para um arquivo ou diretório. Ex: *ln –s /bin/ls dir*, *ln –s /tmp tmp*
cat: lista o conteúdo de arquivos. Ex: *cat /etc/fstab*
more: filtro de exibição de dados. Ex: *more /etc/fstab*
less: filtro de exibição de dados. Ex: *less /etc/fstab*
cmp: compara 2 arquivos. Ex: *cmp arq1 arq1*
diff: exibe as diferenças entre 2 arquivos texto, linha por linha. Ex: *diff arq1 arq2*
find: percorre uma hierarquia de diretórios. Ex: *find . –name .doc –print*, *find / -name *.jpg –exec rm –f {} \;*
grep: imprime linhas que possuem um padrão especificado. Ex: *grep root /etc/passwd*
file: determina o tipo de um arquivo (texto, binário, script do shell, etc.). Ex: *file /bin/ls*, *file /etc/passwd*
tail: exibe a parte final de um arquivo. Ex: *tail /var/log/messages*, *tail –20 /var/log/messages*
head: exibe as primeiras linhas de um arquivo. Ex: *head /var/log/messages*, *head –20 /var/log/messages*
cut: seleciona partes de uma linha de texto. Ex: *cut –c 10-20 /etc/passwd*, *cut –d: –f 5 /etc/passwd*
wc: contador de palavras, linhas e bytes. Ex: *wc –l /etc/passwd*
sort: ordena linhas de arquivo texto. Ex: *sort arq*, *sort –n –r arq*
touch: altera as datas de acesso e modificação de arquivos. Ex: *touch *.h *.c*
lsof: lista arquivos abertos.

7. Agrupamento, conversão e compressão de arquivos

tar: cria arquivos para *tapes* e adiciona ou remove arquivos. Ex: *tar –tvf arq.tar*, *tar –cvf dir.tar dir*, *tar –xvf dir.tar*
dd: copia arquivos, podendo realizar conversões de formato. Ex: *dd if=bootnet.img of=/dev/fd0*. Pode acessar dispositivos diretamente, sem passar pelo sistema de arquivos.

cpio: copia arquivos de/para dispositivos de E/S
uuencode / ***uudecode***: codifica um arquivo binário para uma representação que pode ser enviada por e-mail. Uudecode decodifica o arquivo.
compress / ***uncompress***: comprime e descomprime dados. Ex: *compress log*, *uncompress log.Z*
gzip / ***gunzip***: comprime ou expande arquivos. Ex: *gzip arq.ext*, *gunzip arq.ext.gz*
zip / **unzip**: empacota e comprime / descomprime arquivos. Ex: *zip arq.ext*, *unzip arq.ext.zip*
bzip2 / ***bunzip2***: comprime / descomprime arquivos. Ex: *bzip2 arq.ext*, *bunzip2 arq.ext.bz2*

8. Gerenciamento de partições e sistemas de arquivo

Assim como em outros sistemas operacionais, Unix permite a manipulação e o acesso a diversas partições nos discos. O utilitário **fdisk** é utilizado para essas manipulações. O nome do dispositivo a ser manipulado é especificado como parâmetro (e.g. ***/dev/hda***, ***/dev/hdb***, ...). Um sistema de arquivos deve ser criado em cada partição que se deseja utilizar (formatação).

fdisk: manipula tabelas de partições de discos rígidos. Ex. *fdisk /dev/hda*
du: exibe estatísticas da utilização do disco. Ex: *du –sk* .
df: exibe informações sobre o espaço livre no disco. Ex: *df –k*
quota: informa limites estabelecidos e a ocupação do disco pelo usuário. Ex: *quota –v*
mkfs: constrói sistemas de arquivos em partições. Ex: *mkfs –t ext2 /dev/hda2*, *mkfs –t msdos /dev/fd0*
fsck: verifica e repara sistemas de arquivos. Ex: *fsck /dev/hda3*
mkswap (linux): cria uma área de swap no Linux. Ex: *mkswap /dev/hda5*
swapon / ***swapoff*** : ativa / desativa arquivos e dispositivos de memória virtual e swap
mount / ***umount***: monta / desmonta sistemas de arquivos.
showmount: exibe informações sobre os sistemas de arquivo NFS de um servidor
tune2fs: ajusta parâmetros de um sistema de arquivos **ext2**. Ex: *tune2fs –l /dev/hda3*, *tune2fs –j /dev/hda2*

9. Segurança e direitos de acesso

A segurança e os direitos de acesso a **arquivos** e **diretórios** em sistemas Unix são baseados na identificação dos seus **proprietários** e **grupos** associados. **Chown**, **chmod** e **chgrp** realizam os ajustes necessários, considerando direitos para **leitura**, **escrita** e **execução** (wrxwrxwrx).

ls –l exibe informações sobre os direitos de acesso a arquivos e diretórios:

d_____	indica diretório
l_____	indica *link*
b_____	indica dispositivo de bloco
c_____	indica dispositivo de caracter
s_____	indica *socket*
p_____	indica *pipe*

_rwx_____	direitos do **proprietário** (*owner*) ao arquivo ou diretório
____rwx___	direitos do **grupo** ao arquivo ou diretório
_____rwx	direitos dos **outros** usuários (não *owner* ou grupo) ao arq. ou diret.

r = leitura; w = escrita; x = execução (em diretórios, permite entrar – cd)

- **chown** – altera proprietário de arquivos e diretórios
- **chgrp** – altera o grupo associado a arquivos e diretórios
- **chmod** – ajusta direitos de acesso

Ex: *chown –R fulano /home/fulano* // ajusta o proprietário do diretório fulano
chgrp grupo diretório // ajusta o grupo associado ao diretório

chmod [u,g,o,a][+,-,=][r,w,x,X,s,t] arq
chmod +x arq, chmod o-w arq, chmod u+w,g+r,o-r arq

Considerando as informações de direitos de acesso como **bits em dígitos octais** (0-7), é possível ajustar-se diretamente os atributos dos arquivos e diretórios. Para tanto, 3 dígitos são utilizados, respectivamente para direitos do **proprietário**, do **grupo** e dos **demais** usuários. Os 3 bits de cada dígito correspondem, em ordem, aos direitos para **leitura**, **escrita** e **execução**.

chmod 755 arq -> 111 101 101 -> rwxr_xr_x arq
chmod 640 arq -> 110 100 000 -> rw_r_____ arq

O comando ***umask***, normalmente executado nos arquivos de configuração do *shell*, define os direitos de acesso que devem ser **excluídos** aos arquivos criados. Ex: *umask 022* define que membros do **grupo** associado e demais usuários (**outros**) **não** terão direito de escrita (**w**) automático aos arquivos ou diretórios criados.

Independentemente do **proprietário** associado a um arquivo, **processos** iniciados preservam a **identidade** do **usuário** que os inicia. Entretanto, para permitir que processos especiais sejam executados com direitos de acesso de usuários ou grupos específicos (tipicamente o *root*), é possível forçar a manutenção da identidade do usuário (***setuid***) ou grupo (***setgid***) do arquivo no processo criado a partir dele.

chmod u+s arq, *chmod g+s arq*, *chmod +s arq, chmod g-s arq, chmod –s arq*

Para preservar os direitos de acesso a arquivos em diretórios compartilhados, é possível ainda ajustar um outro atributo, chamado ***sticky bit***. Atribuído ao diretório /tmp, e.g., faz com que, embora todos os usuários possam escrever nesse diretório, os direitos de cada arquivo sejam preservados.

chmod +t /tmp
chmod –t dir

Usando a notação com números **octais**, o ajuste dos atributos **setuid**, **setgid** e **sticky bit** pode sem feito com um dígito a mais, anterior aos dos direitos de acesso do proprietário do arquivo ou diretório.

chmod 4xyz prog -> ajusta o bit **setuid**
chmod 2xyz prog -> ajusta o bit **setgid**
chmod 1xyz /tmp -> ajusta o **sticky bit**

No ambiente gráfico, é possível ajustar direitos de acesso de maneira mais simplificada através de algum gerenciador de arquivos. Com o gerenciador ***konqueror***, basta selecionar-se uma **pasta** ou **arquivo** e clicar-se o botão da direita sobre ele. A opção ***propriedades*** permite o ajuste das **permissões** de acesso, do **proprietário** e do **grupo** associados.

10. Manipulação de processos

Processos no Linux podem ser iniciados diretamente através do *shell* utilizado. Comandos permitem o controle de suas execuções.

prog <enter> /* execução em **primeiro plano** (*foreground*), bloqueando o *shell* */
prog **&** <enter> /* execução em **segundo plano** (*background*), liberando o *shell* */

Quando passando comando ao *shell*, o sinal "**;**" permite separar diversos comandos numa única linha. Ex: *prog1; prog2; ...*

Por outro lado, é possível que um comando ou conjunto de comandos expandam-se por mais de uma linha. Para tanto, usa-se o sinal "\" ao final de cada linha que não a última do comando.
Ex: *programa –parâmetros .... \ <enter>*
– mais_parâmetros <enter>

<ctrl> **C** /* encerra (termina) a execução de um processo em *foreground* */
<ctrl> **Z** /* suspende (pára) a execução de um processo em *foreground* */

bg: envia para execução em *background* um processo suspenso ou parado, cuja ativação foi feita pelo *shell* corrente.

jobs: mostra os processos parados ou sendo executados em *background* que foram iniciados a partir do *shell* corrente.

fg: envia um processo parado ou em *background* para execução em *foreground*

ps: informa o *status* de processos. Ex: *ps –ef*, *ps –aux*

kill: envia um sinal (*signal*) para um processo especificado pelo seu pid. Ex: *kill –l* (lista sinais disponíveis), *kill 1234*, *kill –9 666*.
Caso o sinal não tenha sido tratado pelo processo, uma ação padrão é executada, normalmente terminando sua execução.

killall: envia um sinal para o processo especificado pelo seu nome. Ex: *killall vi*

Além do envio de sinais através do comando *kill*, a configuração do terminal de acesso apresenta combinações de teclas para o envio direto de alguns sinais relevantes no controle de execução de processos em *foreground*.

stty –a: exibe a configuração dos comandos especiais do console. Entre as configurações, podem ser destacados:

***intr CHAR*:** envia um sinal SIGINT (2). Ex: *stty intr ^C*
start CHAR: envia um sinal SIGCONT (18), reiniciando a saída de dados interrompida. Ex: *stty start ^Q*
stop CHAR: envia um sinal SIGSTOP (19), interrompendo a saída de dados. Ex: *stty stop ^S*
***susp CHAR*:** envia um sinal SIGSTP (20), suspendendo a execução de um processo. Ex: *stty susp ^Z*

nice: executa um comando com baixa prioridade de escalonamento.
renice: ajusta a prioridade de um processo já criado.Ex: *renice +1 pid*, *renice 20 pid*
nohup: inicia a execução de um programa imune a *hangups* (não é encerrado com o fim do *shell* a partir do qual foi iniciado). Ex: *nohup prog &*

at: enfileira um *job* para execução posterior. É preciso que o servidor at (atd) esteja sendo executado. Ex: *at -f lote 23:00 15.11.03*.
atq: examina os *jobs* selecionados para execução posterior
atrm: remove os *jobs* selecionados para execução posterior
atrun: programa a execução posterior de *jobs*, em função da carga do sistema.
batch: executa *jobs* quando a carga do sistema estiver abaixo de um valor especificado

crontab: programa usado para instalar, desinstalar ou listar as tabelas usadas pelo *daemon* **cron**. Cada usuário pode ter sua própria tabela.

time: contabiliza o tempo de execução de um processo. Ex: *time gcc ...*
sleep: suspende a execução de um comando por pelo menos o tempo especificado em segundos. Ex: *sleep 10; echo fim!*

top: mostra processos com maior ocupação da UCP e controla suas execuções.

Uma vez iniciado um processo, o seu encerramento pode ser feito de diversas maneiras, dependendo de como foi feita a sua ativação.

E*ncerrando a execução de um processo:*
a) em *foreground*: **<ctrl> C** /* caso não tenha ignorado este sinal */
b) em *background*: **fg; <ctrl> C** /* caso não tenha ignorado este sinal */
c) iniciado por outro *shell*:
ps –ef | grep [nome_prog ou "*login"*]**; kill –9 pid**
ps –U *login*; kill –9 pid
killall –9 nome_prog

11. Redirecionamento de entrada e saída de dados

A entrada e saída de dados em processos no ambiente Unix é feita através de 3 arquivos abertos automaticamente nas suas ativações: ***stdin*** (**0**), ***stdout*** (**1**) e ***stderr*** (**2**), que apontam naturalmente para o terminal ou janela onde o *shell* de ativação está sendo executado.

Caracteres especiais usados na ativação de um processo, entretanto, permitem redirecionar esses dados para outros arquivos, ou mesmo para um mecanismo de comunicação entre processos.

> *ou* 1>: redirecionamento da saída de dados de um processo (*overwrite*). Ex: *ls > diret*
>> *ou* 1>>: redirecionamento da saída de dados (*append*). Ex: *ls >> diret*
>& *ou* 2> : redirecionamento das mensagens de erro. Ex: *make >& arq_msg_erro*
2>> : redirecionamento das mensagens de erro (*append*). Ex: *prog 2>> msg_erro*
< : redirecionamento da entrada de dados de um processo. Ex: *prog1 < arq_dados*
| (***pipe***): cria um mecanismo de comunicação entre processos. Ex: *ls –la | more*
tee: copia os dados do arquivo padrão de entrada para o arquivo padrão de saída, fazendo opcionalmente uma cópia para outros arquivos de saída. Ex: *p1 | tee result*

12. Monitoramento da memória e memória compartilhada

free: (linux) mostra as quantidades de memória livre e ocupada
ipcs: informa o *status* de mecanismos de comunicação inter-processo. Ex: *ipcs*
ipcrm: remove uma fila de mensagem, semáforo ou memória compartilhada. Ex: *ipcrm {shm | msg | sem} id ...*

13. Desligando o sistema

Para oferecer um melhor desempenho nos sistemas de arquivos, caches são mantidos pelo sistema operacional numa área de memória. Entretanto, uma desvantagem dessa técnica é que, se o sistema for desligado de maneira abrupta, é possível que arquivos abertos sejam corrompidos. Assim, um procedimento de desligamento do sistema deve ser realizado antes que o computador possa ser desligado.

sync: força a conclusão de operações de disco pendentes, esvaziando o *cache* do disco
shutdown: encerra a operação do sistema. Ex: *shutdown –h now*, *shutdown –h –t 5*, *shutdown –r now*
halt: encerra a operação do sistema
reboot: reinicia a operação do sistema operacional
<Ctrl> <alt> <del>: reinicia o sistema (linux)
init / ***telinit***: controla a iniciação e o nível de execução do sistema (*runlevel*). Enquanto o nível **6** promove a **reiniciação** do sistema, o nível **0** faz o seu **encerramento**. Ex: *init 0*, *init 6*

14. Utilitários de rede

ifconfig: configura e exibe parâmetros de interfaces de rede. Ex: *ifconfig*

route: manipula e exibe a tabela de rotas. Ex: *route*
netstat: informa sobre as conexões de rede, tabelas de rotas, estatísticas sobre as interfaces e outras informações. Ex: *netstat –na*, *netstat –r*, *netstat –tcp*

arp: consulta e atualização da tabela de resolução de endereços. Ex: *arp -a*
ping: envia pacotes ICMP ECHO_REQUEST para *hosts* na rede, que enviam pacotes ICMP ECHO_RESPONSE caso estejam ativos
nslookup, ***host***, ***dig***: utilitários para consultas de nomes de domínio na Internet
traceroute: imprime o caminho dos pacotes entre o computador local e uma estação remota

ftp: programa de transferência de arquivos da ARPANET
telnet: programa para acesso de computadores remotos através do protocolo TELNET

ssh: programa de *shell* remoto com transmissão de dados criptografados
scp: programa de transferência de arquivos com transmissão criptografada

rlogin: programa de *login* remoto
rsh: programa de *shell* remoto. Permite executar comandos ou abrir uma sessão interativa
rcp: programa para cópia de arquivos entre estações

mail: programa para envio e recebimento de mensagens do correio eletrônico
pine: programa para email e news na Internet

15. Impressão

lpr: impressão *off line*. Usa o servidor de *spool* para impressão de arquivos quando o dispositivo estiver disponível. Ex: *lpr arquivo*
lpq: examina a fila de impressão. Ex: *lpq*
lprm: remove *jobs* da fila do *spool* de impressão. Ex: *lprm 123*

16. Editores de texto

Modo texto: ***vi***, ***emacs***, ***pico***
Ambiente gráfico: ***kedit, kwrite, xedit, ...***

17. Gerenciamento de pacotes

A instalação de programas num sistema Linux pode ocorrer de diversas formas. Tendo um compilador C instalado, é possível copiar os arquivos fontes de um programa desejado para um diretório local, configurá-lo, compilá-lo e copiar os arquivos gerados para os diretórios apropriados.

```
gunzip pacote.src.tar.gz
tar –xvf pacote.src.tar
cd pacote.src
./configure
make
```

make install

Outros pacotes necessários para uma aplicação devem ser copiados e configurados num procedimento semelhante. A instalação de novas versões desses pacotes envolve a retirada manual dos arquivos e a configuração da nova versão.

Para simplificar esse procedimento, foi criado o conceito de pacote de software no formato RPM (*RedHat Package Manager*). Pacotes no formato **.rpm** podem ser facilmente instalados, verificados, atualizados e desinstalados. Para tanto, basta copiar o arquivo no formato apropriado, que já concentra num único arquivo todos os arquivos necessários para a sua configuração e instalação.

Outros pacotes necessários para uma aplicação devem ser previamente instalados.

rpm –ivh pacote.rpm	: instala pacote
rpm –Uvh pacote.rpm	: instala pacote, removendo suas versões anteriores
rpm –e pacote	: remove pacote
rpm – q pacote	: verifica status de pacote

A ferramenta **apt** fornece facilidades estendidas para o gerenciamento de pacotes rpm. Através dela, é possível instalar e atualizar pacotes de maneira automatizada, resolvendo inclusive problemas de dependências entre eles.

Para tanto, basta especificar-se uma fonte para os programas a serem instalados.

/etc/apt/sources.list: contém uma relação de *mirrors* para *downloads* de programas e atualizações

apt-get update: atualiza a lista de pacotes que podem ser instalados ou atualizados a partir das fontes especificadas.
apt-get install pacote: instala o pacote, com suas respectivas dependências
apt-get upgrade: atualiza as versões dos pacotes instalados
apt-get dist-upgrade: atualiza uma distribuição
apt-get remove pacote: remove pacote
apt-cdrom add: adiciona informações sobre CDs como fonte para instalações
apt-get source pacote: copia código fonte do pacote, incluindo suas dependências.

No ambiente gráfico, o programa ***kpackage*** fornece uma interface gráfica para o gerenciamento de pacotes **rpm**. Com essa ferramenta, é possível listar os pacotes instalados e, selecionar-se outros que podem ser instalados a partir de diretórios especificados.

Também no ambiente gráfico, o programa ***synaptic*** pode permite a configuração e a atualização de pacotes e distribuições, fazendo chamada aos comandos **apt**.

18. Comunicação entre usuários

write: escreve uma mensagem para outro usuário conectado ao sistema
wall: envia uma mensagem para o terminal de todos os usuários conectados
mesg: controla a escrita de mensagens no terminal. Ex: *mesg y*, *mesg n*

talk: programa para conversa com outros usuários – *chat*, Ex:. *talk fulano*, *talk fulano@host.dom* (programa ***talkd*** deve estar habilitado na máquina remota)

19. Miscelânea

expr: interpreta argumentos como uma expressão. Ex: *expr 2 + 2*, *expr 2 * 3*, *expr 5 / 2*, *expr 5 % 2*
cal: exibe um calendário. Ex: *cal*, *cal 2003*