

# **2A. PARTE:**

## **CONHECIMENTOS GERAIS**

✓ *Copyright (c) 2002-2005 – Ednei Pacheco de Melo.*

Permission is granted to copy, distribute and/or modify this document under the terms of the *GNU Free Documentation License*, version 1.1 or any later version published by the *Free Software Foundation*; a copy of the license is included in the section entitled “*GNU Free Documentation License*”.

<b>ABERTURA.....</b>	<b>9</b>
<b>I. SISTEMA DE ARQUIVOS.....</b>	<b>10</b>
<i>Introdução.....</i>	<i>10</i>
<b>A FHS.....</b>	<b>10</b>
Sobre o FSSTND.....	10
<b>A estrutura do diretório raiz.....</b>	<b>10</b>
/bin – Binários essenciais.....	11
/boot – Inicialização do sistema.....	11
/dev - Dispositivos.....	11
/etc – Configuração.....	11
/home – Dados pessoais.....	12
/lib – Bibliotecas essenciais.....	12
/media e /mnt – Pontos de montagens.....	12
/opt – Compatibilidade entre aplicativos.....	13
/proc e /sys – Informações e processos do kernel.....	13
/root – Administrador do sistema.....	14
/sbin – Binários essenciais para administração do sistema.....	14
/tmp – Arquivos temporários.....	14
/usr – Aplicativos e utilitários gerais.....	14
/var – Variáveis.....	15
<b>Conclusão.....</b>	<b>16</b>
<b>II. DEVICE (DISPOSITIVOS).....</b>	<b>17</b>
<i>Introdução.....</i>	<i>17</i>
<i>Tipos de devices.....</i>	<i>17</i>
<b>Principais devices.....</b>	<b>18</b>
Unidades externas e de armazenamento.....	18
Discos rígidos IDE & CD-ROMs.....	18
Unidades SCSI.....	18
Disquetes.....	19
Sistema de áudio.....	19
Fax-modem (portas seriais).....	20
Console terminal.....	20
Impressora (porta paralela).....	20
Outros devices importantes.....	21
<b>Criando devices.....</b>	<b>21</b>
MAKEDEV & mknod.....	21
<b>Conclusão.....</b>	<b>22</b>
<b>III. LINHA DE COMANDO.....</b>	<b>23</b>
<i>Introdução.....</i>	<i>23</i>
<b>O BASH.....</b>	<b>23</b>
<b>Informações e métodos essenciais.....</b>	<b>24</b>

Complemento da tecla <TAB>.....	24
Expressões.....	24
Cores personalizadas.....	25
Nomenclatura diferenciada.....	26
Uma simbologia especial.....	26
<b><i>Equivalências entre o BASH e MS-DOS.....</i></b>	<b>26</b>
Estrutura de diretórios.....	27
Acesso às unidades do sistema.....	27
Especificações do diretório.....	28
Exibição de caminho.....	29
Case sensitive.....	29
<b><i>Observações.....</i></b>	<b>29</b>
<b><i>Conclusão.....</i></b>	<b>30</b>
<b>IV. MANIPULAÇÃO DE ARQUIVOS E DIRETÓRIOS.....</b>	<b>31</b>
<b><i>Introdução.....</i></b>	<b>31</b>
<b><i>Operações e atividades básicas.....</i></b>	<b>31</b>
Listagem e navegação.....	31
ls.....	31
cd.....	33
Visualização.....	34
type.....	34
less.....	34
file.....	35
pwd.....	35
more.....	36
du.....	38
Manipulação.....	39
mkdir.....	39
dd.....	39
cp.....	39
mv.....	40
ln.....	40
Editoração.....	41
mcedit.....	41
Exclusão.....	42
rmdir.....	42
rm.....	42
<b><i>Permissões e atributos de arquivos e diretórios.....</i></b>	<b>43</b>
Permissões de acesso.....	43
chmod.....	44
Grupos e categorias.....	47
chown.....	47
chgrp.....	47
umask.....	47
<b><i>Cópias de segurança e compactação.....</i></b>	<b>48</b>
Arquivamento.....	48
tar.....	49
Compactação.....	50
gzip / gunzip.....	50

<i>bzip2 / bunzip2</i> .....	51
<i>zip / unzip</i> .....	52
Utilitários.....	52
<i>split</i> .....	52
<i>cat</i> .....	53

<b>Conclusão</b> .....	53
------------------------	----

## **V. UNIDADES, PARTIÇÕES E FORMATOS.....54**

<b>Introdução</b> .....	54
-------------------------	----

<b>Os sistemas de arquivos</b> .....	54
--------------------------------------	----

Tipos de partições e sistemas de arquivos.....	54
<i>swap</i> .....	54
<i>ext2</i> .....	54
<i>ext3</i> .....	55
<i>ReiserFS</i> .....	55
<i>ISO9660</i> .....	55
<i>msdos e vfat</i> .....	56

Diferenças entre os sistemas de arquivos ReiserFS e Ext3.....	56
---	----

<b>As unidades e as partições</b> .....	57
---	----

<b>Operações e atividades afins</b> .....	57
---	----

Geral.....	57
<i>Trabalhando com partições e unidades</i> .....	57
<i>mount / umount</i> .....	57
<i>sync</i> .....	59
<i>Definição de sistema de arquivos</i> .....	59
<i>mkfs</i> .....	59
<i>mkreiserfs</i> .....	60
<i>mkswap / swapon</i> .....	60
<i>Verificando as partições e unidades do sistema</i> .....	61
<i>df</i> .....	61
<i>badblocks</i> .....	62
<i>fsck</i> .....	62
<i>reiserfs</i> .....	63
<i>Transferindo dados</i> .....	64
<i>dd</i> .....	64
Disquetes.....	64
<i>Utilização</i> .....	64
<i>Formatação</i> .....	64
Formatação de baixo nível.....	64
Sistema de arquivos.....	65
Superformat.....	65
<i>Discos de inicialização</i> .....	66
<i>Sobre o pacote mtools</i> .....	66
CD-R/CD-RW.....	67
<i>mkisofs</i> .....	67
<i>cdrecord</i> .....	68

<b>Conclusão</b> .....	71
------------------------	----

## **VI. CONTAS DE USUÁRIOS E GRUPOS DE ACESSO.....72**

<b>Introdução</b> .....	72
-------------------------	----

<b>Considerações básicas</b> .....	72
------------------------------------	----

Contas de acesso.....	72
-----------------------	----

<i>Root - o administrador de sistema</i> .....	72
<i>Conta de usuário</i> .....	73
<i>Grupos de acesso</i> .....	73
<i>ID</i> .....	73
<i>Permissões</i> .....	74
<i>Senha</i> .....	74
<b><i>Comandos e parâmetros gerais</i>.....</b>	<b>74</b>
<i>Adição de contas de usuário e grupos</i> .....	74
<i>adduser</i> .....	74
<i>groupadd</i> .....	76
<i>Administração de contas</i> .....	77
<i>login / logout / exit</i> .....	77
<i>id</i> .....	77
<i>users / groups</i> .....	77
<i>passwd</i> .....	78
<i>finger</i> .....	79
<i>who</i> .....	79
<i>whoami</i> .....	80
<i>uptime</i> .....	80
<i>Eliminando usuários e grupos</i> .....	80
<i>userdel</i> .....	80
<i>groupdel</i> .....	81
<i>Obtendo os privilégios de outros usuários</i> .....	81
<i>su</i> .....	81
<b><i>Os arquivos de configuração</i>.....</b>	<b>82</b>
<i>/etc/passwd</i> .....	82
<i>/etc/shadow</i> .....	82
<i>/etc/groups</i> .....	83
<b><i>Problemas mais freqüentemente</i>.....</b>	<b>84</b>
<b><i>Conclusão</i>.....</b>	<b>84</b>
<b>VII. PROCESSOS EM EXECUÇÃO.....</b>	<b>85</b>
<b><i>Introdução</i>.....</b>	<b>85</b>
<b><i>Visão geral</i>.....</b>	<b>85</b>
<i>O que é um processo?</i> .....	85
<i>PID</i> .....	85
<i>2o. plano (background)</i> .....	85
<b><i>Gerenciando processos</i>.....</b>	<b>86</b>
<i>Visualização</i> .....	86
<i>ps</i> .....	86
<i>top</i> .....	87
<i>Segundo plano</i> .....	88
<i>Colocando em segundo plano</i> .....	88
<i>"Control-zê"</i> .....	88
<i>bg</i> .....	89
<i>jobs</i> .....	89
<i>fg</i> .....	89
<i>Exclusão</i> .....	89
<i>kill</i> .....	89
<i>killall</i> .....	89

<i>Sobre o diretório /proc.....</i>	<b>90</b>
<i>Conclusão.....</i>	<b>90</b>
<b>VIII. GERENCIADORES DE INICIALIZAÇÃO.....</b>	<b>91</b>
<i>Introdução.....</i>	<b>91</b>
<i>O LILO.....</i>	<b>91</b>
A configuração.....	92
<i>Seção global.....</i>	92
Framebuffer.....	94
<i>Seção de partições.....</i>	94
Liloconfig.....	96
Operações mais freqüentes.....	96
<i>Selecionar o sistema GNU/Linux como padrão.....</i>	96
<i>Mudar a resolução do framebuffer.....</i>	97
<i>Adicionar mais uma entrada no LILO.....</i>	97
<i>Adição de senha.....</i>	98
<i>Parâmetros especiais durante a inicialização.....</i>	98
Atualizando as alterações desejadas.....	99
Disco de recuperação com o lilo.conf.....	99
Problemas mais freqüentes.....	99
<i>Observações finais.....</i>	<b>100</b>
<i>Conclusão.....</i>	<b>101</b>
<b>IX. INICIALIZAÇÃO.....</b>	<b>102</b>
<i>Introdução.....</i>	<b>102</b>
<i>Método de inicialização.....</i>	<b>102</b>
Tipos de métodos.....	102
System V.....	102
Estilo BSD.....	102
<i>Scripts de inicialização.....</i>	<b>102</b>
O diretório /etc/rc.d/.....	103
Inicialização.....	103
Suporte geral.....	104
Suporte à rede.....	104
Configurações locais.....	105
Carregamento de módulos.....	105
Compatibilidade.....	106
A seqüência de scripts na inicialização.....	106
<i>Níveis de execução do Slackware.....</i>	<b>106</b>
Os níveis de execução.....	107
<i>Nível 1 – Manutenção do sistema.....</i>	107
<i>Nível 3 e 4 – Modo multi-usuário.....</i>	107
<i>Nível 6 – Reinicialização do sistema.....</i>	108
<i>Demais níveis de execução.....</i>	108
Diferenças entre o padrão Slackware e os demais.....	108
/etc/inittab.....	108
<i>Operações e ajustes afins.....</i>	<b>110</b>
Ativar / desativar scripts de inicialização.....	110
<i>Método manual.....</i>	110

<i>Método automatizado</i> .....	110
“Melhorando” a compatibilidade com o System V.....	111
<b>Conclusão</b> .....	<b>112</b>
<b>X. GERENCIAMENTO BÁSICO DE PROGRAMAS</b> .....	<b>113</b>
<b>Introdução</b> .....	<b>113</b>
<b>Ferramentas &amp; métodos</b> .....	<b>113</b>
Slackware Package Tools.....	113
<i>Instalação</i> .....	113
<i>Atualização</i> .....	114
<i>Remoção</i> .....	114
Red Hat Package Management.....	114
<i>Instalação</i> .....	114
<i>Atualização</i> .....	114
<i>Remoção</i> .....	115
Compilação do código-fonte.....	115
<i>A configuração</i> .....	115
<i>A compilação</i> .....	115
<i>A instalação</i> .....	115
<i>A desinstalação</i> .....	116
Outros utilitários.....	116
<b>Conclusão</b> .....	<b>116</b>
<b>XI. VARIÁVEIS DE SISTEMA</b> .....	<b>117</b>
<b>Introdução</b> .....	<b>117</b>
<b>As variáveis</b> .....	<b>117</b>
Path / RootPath.....	117
Home.....	118
OsType.....	118
Shell.....	118
Term.....	118
User.....	119
<b>Comandos para manipulação</b> .....	<b>119</b>
echo.....	119
set.....	119
export.....	120
<b>Internacionalização</b> .....	<b>120</b>
<b>Arquivos de configuração</b> .....	<b>121</b>
/etc/profile.....	121
<i>O diretório /etc/profile.d/</i> .....	123
~/bashrc.....	123
<b>Conclusão</b> .....	<b>124</b>
<b>XII. COMANDOS E FUNÇÕES GERAIS</b> .....	<b>125</b>
<b>Introdução</b> .....	<b>125</b>
<b>Comandos gerais</b> .....	<b>125</b>
Administração do sistema.....	125
<i>Arch / uname</i> .....	125

<i>dmesg</i> .....	125
<i>halt</i> .....	126
<i>shutdown</i> .....	126
Ajuda.....	126
<i>man</i> .....	126
<i>info</i> .....	127
<i>--help</i> .....	128
Diversos.....	129
<i>clear</i> .....	129
<i>cal</i> .....	129
<i>free</i> .....	130

<b><i>Conclusão</i></b> .....	<b>130</b>
-------------------------------	------------

<b>ENCERRAMENTO</b> .....	<b>131</b>
---------------------------	------------



Chegamos agora à mais uma nova etapa desta edição do livro. Trata-se da obtenção de conhecimento técnico e aprendizado das operações básicas e essenciais, necessárias para a boa manutenção dos sistemas *GNU/Linux*.

Como qualquer outro sistema operacional, o *kernel* do *Linux* possui agregado diversas ferramentas, utilitários e aplicações específicas que visam prover aos usuários e administradores uma gama de recursos básicos para a boa administração do sistema em si e seus respectivos dados. Atividades simples, básicas e triviais, como copiar arquivos, definir permissões de acesso, acessar unidades, monitorar processos, entre outras (até mesmo mais complexas), são de extrema importância para a garantia de uma boa estabilidade e ótima performance do computador em geral. Nesta parte estarão descritas todas as instruções de caráter indispensável, de forma simples, prática e organizada, com o intuito de facilitar ao máximo a obtenção dos conhecimentos mínimos necessários para este propósito.

A leitura dos próximos capítulos desta parte é de caráter INDISPENSÁVEL para os usuários iniciantes e que pela 1ª vez estão tendo contato com os sistemas *GNU/Linux*, além de ser de grande importância para os usuários de nível médio, pois irão proporcionar todos os conhecimentos necessários para a obtenção de um bom aproveitamento das instruções contidas nas partes subseqüentes. Já àqueles usuários que possuem uma certa experiência ou que são especialistas no ramo, podem tranquilamente seguir adiante para a próxima parte (ou ainda outras posteriores, dependendo de suas habilidades), ficando esta decisão à seu critério. Porém deixaremos bem claro que a nossa satisfação será enorme caso os mesmos realizassem uma breve leitura destes capítulos, com o intuito de analisar os pontos negativos e positivos deste material, para que resultem em críticas e sugestões de alta qualidade para a melhoria da obra.

Para àqueles que recentemente vieram de outras distribuições (ou estão pensando com seriedade em adotar o *Slackware*), sugerimos a leitura dos capítulos *Inicialização* e *Gerenciamento de programas (básico)*. Neles estarão descritas as particularidades da distribuição-base do livro, além de outras instruções necessárias para a boa manutenção do sistema.

Enfim, não tendo nada mais à tratar, tenham uma boa leitura! &:-D

# I. SISTEMA DE ARQUIVOS

---

## INTRODUÇÃO

Como qualquer outro sistema operacional, os sistemas *GNU/Linux* realizam a manipulação de diversos dados e informações, possuindo para isto uma estrutura de arquivos e diretórios bem definida e padronizada. Para cada tipo de arquivo e de acordo com suas funcionalidades e importância, existe um local específico para seu armazenamento. Além do diretório principal do sistema, representado por "/" – <BARRA\_COMUM> –, existe uma série outros diretórios especificados pela padronização.

Neste capítulo, iremos conhecer a estruturação dos dados e diretórios dos sistemas *GNU/Linux*, como também as suas particularidades e algumas recomendações necessárias para a sua boa manutenção.

## A FHS

✓ <<http://www.pathname.com/fhs/>>.

A norma *FHS* – *Filesystem Hierarchy Standard* – é um conjunto de requerimentos técnicos que visam estabelecer normas e padrões para a estrutura do sistema de arquivos *Unix*, derivados e clones. É ela quem define quais são os diretórios que deverão existir, a localização dos arquivos de configuração, etc., com o intuito de promover a padronização e compatibilidade dos sistemas *GNU/Linux* e suas aplicações.

Para informações adicionais, consultem a *9a. Parte – Documentações: -> Normas técnicas e de padronização*.

## SOBRE O FSSTND

O *FSSTND* – *Linux Filesystem Structure* – foi concebido anteriormente e com os mesmos propósitos da *FHS*, porém devido à sua pouca rigidez sobre diversos aspectos, muitas distribuições definiam por si própria a localização de diversos arquivos de sistema. Os arquivos de inicialização e configuração do sistema eram os que mais situavam-se fora de uma padronização específica, mesmo que estas distribuições tomassem como base os métodos de inicialização *SystemV* e *BSD*.

## A ESTRUTURA DO DIRETÓRIO RAÍZ

Os diretórios que compõe a estrutura do diretório raiz conforme as normas da *FHS* são: */bin*, */boot*, */dev*, */etc*, */home*, */lib*, */media*, */mnt*, */opt*, */proc*, */root*, */sbin*, */tmp*, */usr* e */var*. Segue abaixo a descrição detalhada sobre a estrutura e seus respectivos diretórios, além de algumas dicas, conselhos e recomendações úteis para a sua utilização.



## /BIN – BINÁRIOS ESSENCIAIS

O diretório */bin* contém todos (ou a maioria) os arquivos binários com os comandos essenciais dos usuários, tais como os programas da linha de comando, entre outros. Os arquivos contidos neste diretório geralmente não são modificados após a instalação, porém quando de novas atualizações do sistemas, poderão ser alterados.

## /BOOT – INICIALIZAÇÃO DO SISTEMA

O diretório */boot* contém todos os arquivos necessários (estáticos) para a inicialização do sistema (*boot loader*), exceto os arquivos de configuração (*/etc*) e o gerenciador de inicialização (*LILLO*). Em distribuições que utilizam o gerenciador *GRUB*, este encontra-se armazenado em um subdiretório dentro deste diretório chamado */boot/grub*.

## /DEV - DISPOSITIVOS

Todo e qualquer dispositivo, tais como portas seriais, discos rígidos, *scanners*, *mouse*, *modems*, etc., em sistemas baseados em *UNIX* são tratados como arquivos denominados *device node* – *nodo de dispositivo* – ou simplesmente *device*. Para ter acesso às funcionalidades de qualquer dispositivo, deveremos recorrer aos seus respectivos *devices*. E onde se encontram estes arquivos?

O diretório */dev* contém todos os arquivos de dispositivos (*device*) necessários para cada dispositivo em que o *kernel* do *Linux* suporta. Neste diretório também temos um *script* chamado *MAKEDEV*, o qual nos possibilita a criação de novos dispositivos de maneira fácil e prática, conforme nossas necessidades.

## /ETC – CONFIGURAÇÃO

O diretório */etc* contém todos os arquivos diversos de configuração local do computador utilizado, desde os arquivos de configurações diversas tais como a tabela para montagem de partições, o gerenciador de inicialização *LILLO*, *scripts*, etc. Além deste diretório, existem outros diretórios em sua estrutura especificados pela *FHS*.

<i>/etc</i>	
<i>/etc/X11</i>	Arquivos de configuração local para o servidor <i>X</i> .
<i>/etc/rc.d</i>	Arquivos de configuração e <i>scripts</i> para a inicialização.

Para obterem maiores informações, consultem a *9. Parte: Documentações* -> *Normas técnicas e de padronização*.



## /HOME – DADOS PESSOAIS

Em virtude dos sistemas *Unix-likes* terem sido concebidos para serem sistemas multi-usuários, o diretório */home* é designado exclusivamente para o armazenamento dos arquivos pessoais das contas de usuário do sistema, incluindo personalizações específicas de sua conta no sistema.

Para cada conta de usuário criado, é acrescentado neste diretório um subdiretório que utiliza a mesma nomenclatura definida para ser o apelido. Por exemplo, para conta do usuário *darkstar*, teremos um diretório */home/darkstar/* para o armazenamento de todos os arquivos e configurações pessoais desta conta.

## /LIB – BIBLIOTECAS ESSENCIAIS

O diretório */lib* contém bibliotecas compartilhadas necessárias para a execução dos arquivos contidos nos diretórios */bin* e */sbin*. Ainda neste diretório são encontrados os módulos do *kernel*, essenciais para as funcionalidades básicas do sistema. Estes módulos são armazenados numa estrutura especificada em */lib/modules-[VERSÃO]*. As bibliotecas necessárias para as aplicações hospedadas em */usr* não pertencem à */lib*.

## /MEDIA E /MNT – PONTOS DE MONTAGENS

Os diretórios */media* e */mnt* foram definidos para serem utilizados exclusivamente para a montagem de unidades. A diferença entre os dois está justamente no tipo de unidade a ser desmontada.

```
$ ls -l /mnt/
total 14
-rw-r--r-- 1 root root 376 2006-09-26 00:09 README
drwxr-xr-x 2 root root 48 2006-09-25 22:02 cdrecorder
drwxr-xr-x 2 root root 48 2002-03-16 04:34 cdrom
drwxr-xr-x 2 root root 48 2006-09-25 22:02 dvd
drwxr-xr-x 2 root root 48 2006-10-11 21:39 flash
drwxr-xr-x 2 root root 48 2002-03-16 04:34 floppy
drwxr-xr-x 2 root root 48 2002-03-16 04:34 hd
drwxr-xr-x 2 root root 48 2006-10-21 23:21 iso
drwxr-xr-x 2 root root 48 2006-09-25 22:02 memory
drwxr-xr-x 12 darkstar users 472 2006-11-05 01:47 pkg
drwxr-xr-x 2 root root 48 2006-09-25 22:03 tmp
dr-x----- 1 root root 4096 2006-11-03 20:55 win
drwxr-xr-x 2 root root 48 2006-09-25 22:02 zip
$ _
```

*Listagem da estrutura de diretórios do ponto de montagem /mnt.*

O */media* deverá ser utilizado exclusivamente para a montagem de mídias removíveis, como *CD-ROMs*, disquetes e memórias eletrônicas em geral.<sup>1</sup> Já

1 Em algumas distribuições, a montagem de determinadas unidades – disquetes e *CD/DVD-ROMs* – são feitas diretamente num diretório situados na raiz – */floppy* e */cdrom*, respectivamente.

no */mnt* se concentrará a montagem de volumes de uso provisório, como uma *HD*, por exemplo.

## **/OPT – COMPATIBILIDADE ENTRE APLICATIVOS**

O diretório */opt*, apesar de não pertencer à norma *FHS*, foi mantido em virtude da necessidade de manter a compatibilidade com antigos programas que ainda são muito utilizados atualmente. Em uma consulta que realizamos na página eletrônica da *Slackware LinuxBR*, obtivemos outras informações muito interessantes sobre este diretório:

*“Pacotes de software opcional. A idéia atrás do /opt é que cada pacote de software seja instalado para /opt/<software package>, o que facilita para uma desinstalação subsequente. Slackware distribui algumas coisas no /opt (como o KDE em /opt/kde), mas você é livre para colocar o que quiser no /opt.” -> [“Estrutura de diretórios do Slackware LinuxBR”, por r\_linux & mistif].*

Conforme a informação acima, o *KDE* encontra-se instalado neste diretório, em */opt/kde/*. Já o *GNOME*, encontra-se em */usr/share/gnome*<sup>2</sup>. Por último, a versão atual do *OpenOffice.org*, que também é instalada em */opt*.

## **/PROC E /SYS – INFORMAÇÕES E PROCESSOS DO KERNEL**

O diretório */proc* contém um sistema de arquivo virtual, com informações gerais do sistema e processo do *kernel*. Na verdade, o seu conteúdo não faz parte dos arquivos de sistema; ele é apenas um sistema de arquivo virtual para que os administradores do sistema tenham acesso às informações do processamento do *kernel* em forma de arquivos para consulta, onde inclusive podemos realizar passagem de informações ao *kernel* por eles através de parâmetros específicos. Já o */sys* ainda está para ser definido, embora seja utilizado pelo *kernel Linux 2.6* em substituição ao */proc*.

Para obterem maiores informações, consultem a *4a. Parte – Ajustes & Configurações*: -> *A importância das estruturas em /etc e /proc*.

---

2 À partir do *Slackware 10.2*, não são mais disponibilizados os pacotes do *GNOME* no *FTP* oficial da distribuição. Para tê-lo instalado no sistema, deveremos recorrer a projetos externos, como o *Dropline GNOME*, *GNOME.SlackBuild* e *GWARE*.

## **/ROOT – ADMINISTRADOR DO SISTEMA**

O diretório */root* é definido para ser utilizado exclusivamente no armazenamento de dados e arquivos pessoais do superusuário – o *root*. Ele é mantido na raiz principal e não é situado em */home*, em decorrência de uma possibilidade de pane geral do sistema, caso este esteja separado em uma partição. Ao iniciarmos sistema como superusuário para realizar alguma tarefa de manutenção específica, ficaríamos preso à necessidade de ter seus arquivos pessoais disponíveis, e como provavelmente esta partição não se encontrará (pelo fato de utilizar o nível de manutenção), teremos complicações para realizarmos a autenticação do superusuário.

Não é recomendado o uso deste diretório para qualquer finalidades que não seja para a administração e/ou manutenção do sistema, em especial atividades comuns à usuários tais como leitura do correio eletrônico, armazenamento de dados diversos, etc. Para estas atividades, o administrador deverá ter ou criar para si uma conta de usuário comum.

## **/SBIN – BINÁRIOS ESSENCIAIS PARA ADMINISTRAÇÃO DO SISTEMA**

O diretório */sbin* somente armazena arquivos binários essenciais para a administração do sistema, onde os mesmos são utilizado somente pelo superusuário ou durante a inicialização do sistema. Todos os executáveis necessários para diversas outras atividades pertinentes estarão disponíveis, como as operações com pacotes, módulos, processos, configurações, partições, etc.

## **/TMP – ARQUIVOS TEMPORÁRIOS**

O diretório */tmp* armazena arquivos temporários gerados pelo sistema. Todos os usuários têm permissão de leitura e escrita nele. Geralmente este diretório é limpo a cada inicialização ou a intervalos relativamente freqüentes. Por este motivo, deveremos evitar a guarda de arquivos por um determinado tempo neste diretório, mesmo que eles sejam inúteis. &;-D

## **/USR – APLICATIVOS E UTILITÁRIOS GERAIS**

O diretório */usr* é a segunda maior hierarquia de diretórios do sistema. Todos os aplicativos e utilitários do sistema encontram-se aqui. Seria como uma espécie de pasta *Arquivos de Programas* do *Windows*, só que bem mais estruturada. Sua composição contém a seguinte estrutura:

<i>/usr</i>	
<i>/usr/X11R6</i>	Sistema X Windows, versão 11, release 6.
<i>/usr/bin</i>	A maioria dos comandos de usuário.
<i>/usr/dict</i>	Listas de palavras. -



<b><i>/usr</i></b>	
<i>/usr/doc</i>	Documentação miscelânea.
<i>/usr/etc</i>	Configuração do sistema.
<i>/usr/games</i>	Jogos e arquivos educacionais.
<i>/usr/include</i>	Arquivos <i>header</i> (cabecinhos) incluídos por programas <i>C</i> .
<i>/usr/lib</i>	Bibliotecas principais dos programas.
<i>/usr/local</i>	Hierarquia local, utilizado para programas que não “ <i>pertencem</i> ” ao sistema (distribuição).
<i>/usr/man</i>	Manual digital dos principais comandos.
<i>/usr/sbir</i>	Arquivos de administração do sistema não vitais.
<i>/usr/share</i>	Informação independente da arquitetura.
<i>/usr/src</i>	Armazenamento de código fonte de diversas aplicações inerentes da distribuição (inclusive do próprio <i>kernel</i> ).

Os atalhos que compõe a estrutura do diretório */usr* são:

<b><i>Atalho...</i></b>	<b><i>... para:</i></b>
<i>/usr/spool</i>	<i>/var/spool</i>
<i>/usr/tmp</i>	<i>/var/tmp</i>
<i>/usr/spool/locks</i>	<i>/var/lock</i>

Estes são necessários apenas por questão de compatibilidade.

## ***/VAR – VARIÁVEIS***

O diretório */var* contém informações variáveis, como arquivos e diretórios em fila de execução, arquivos temporários transitórios, etc. Segue abaixo sua composição:

<b><i>/var</i></b>	
<i>/var/adm</i>	Informações administrativa do sistema (obsoleto). Atalho simbólico até <i>/var/log</i> .
<i>/var/catman</i>	Páginas do manual formatadas localmente.
<i>/var/lib</i>	Informação do estado das aplicações.
<i>/var/local</i>	Informação variável do software de <i>/usr/local</i> .
<i>/var/named</i>	Arquivos <i>DNS</i> , somente rede.
<i>/var/nis</i>	Arquivos base de dados <i>NIS</i> .
<i>/var/run</i>	Arquivos relevantes a processos execução do sistema.

DB

<i>/var</i>	
<i>/var/spool</i>	Diretórios de trabalhos em fila para realizar-se depois.
<i>/var/tmp</i>	Arquivos temporários, utilizado para manter o <i>/tmp</i> pequeno.

Em virtude da existência de antigos programas ainda em vigor, deverão existir alguns subdiretórios para a compatibilidade com os mesmos em */var*. Os diretórios que compõe esta estrutura são: */var/backups*, */var/cron*, */var/lib*, */var/local*, */var/msgs* e */var/preserve*.

## CONCLUSÃO

O conhecimento das características e particularidades do sistema de arquivos de um sistema operacional é fator de suma importância para a sua administração. Saber as funcionalidades de determinados arquivos e diretórios, localização dos arquivos de configuração, permissões de acesso, tudo isso influencia no momento de uma necessidade de intervenção.

Por mais diferentes que sejam as distribuições, todas elas possuem basicamente a mesma estrutura do sistema de dados. Graças à isto, as administrações e intervenções necessárias no sistema de dados poderão ser realizadas em quaisquer sistema *GNU/Linux* sem maiores transtornos. Eis um dos motivos da importância de se utilizar os recursos da linha de comando para interagirmos na manutenção do sistema! &;-D



## II. DEVICE (DISPOSITIVOS)

---

### INTRODUÇÃO

Os *device drivers* – dispositivos – são arquivos especiais utilizados pelo sistema para obter acesso aos recursos presentes no computador.

Todos os periféricos e recursos do sistema são acessados pelo *kernel* através de *devices*. Se quer formatar um disquete, será necessário a utilização de um desses arquivos; para acessar a *Internet*, será necessário outro. Utilizar um terminal, mais outro. E assim por diante.

Neste capítulo iremos conhecer os principais *device drivers* – chamaremos apenas de *devices* para facilitar a pronúncia – e suas particularidades.

### TIPOS DE DEVICES

Nos sistemas *GNU/Linux* encontramos tradicionalmente duas categorias: os *devices* do tipo caracter e os *devices* do tipo de bloco.

Os *devices* do tipo caracter são aqueles em que a transferência de dados são realizadas de modo *serial*, ou seja, um caracter por vez.

```
$ ls -l /dev/lp*
crw-rw---- 1 root lp 6, 0 1995-04-27 20:17 /dev/lp0
crw-rw---- 1 root lp 6, 1 1995-04-27 20:17 /dev/lp1
crw-rw---- 1 root lp 6, 2 1995-04-27 20:17 /dev/lp2
$ _
```

*Exemplos de devices do tipo caracter.*

Dentre os principais exemplos estão as portas paralelas (impressora), portas seriais (*modems*), *devices* de áudio, terminais, teclado, *mouse*, etc.

Já nos *devices* do tipo bloco diferenciam-se do tipo caracter no que concerne a transferência de dados – pois como o próprio nome diz – é feita por blocos, oferecendo grande quantidade de dados por vez.

```
$ ls -l /dev/sb*
brw-rw---- 1 root cdrom 25, 0 1994-07-18 13:07 sbpcd
brw-rw---- 1 root cdrom 25, 0 1994-07-18 13:07 sbpcd0
brw-rw---- 1 root cdrom 25, 1 1994-07-18 13:07 sbpcd1
brw-rw---- 1 root cdrom 25, 2 1994-07-18 13:07 sbpcd2
brw-rw---- 1 root cdrom 25, 3 1994-07-18 13:07 sbpcd3
$ _
```

*Exemplos de devices do tipo bloco.*

Já nesta categoria estão em geral os *devices* de armazenamento tais como disquetes, discos rígidos, *CD-ROMs*, dispositivos *USB* (memória eletrônica), entre outros, devido ao modo de transferência de dados.



## PRINCIPAIS DEVICES

Existe uma imensa quantidade de *devices* específicos; porém nesta literatura, somente conheceremos os mais utilizados pelos usuários comuns. Estes por sua vez subdividem-se em diversas categorias e encontram-se armazenados no diretório */dev*.

## UNIDADES EXTERNAS E DE ARMAZENAMENTO

### DISCOS RÍGIDOS IDE & CD-ROMs

Todos os discos rígidos conectados à controladora *IDE* utilizam os seguintes *devices* para serem acessados pelo sistema:

<b><i>Discos rígidos IDE &amp; CD-ROMs</i></b>	
<i>hda</i>	1a. unidade na controladora primária – mestre.
<i>hda1, hda2...</i>	Partições da 1a. unidade na controladora primária.
<i>hdb</i>	2a. unidade na controladora primária – escravo.
<i>hdb1, hdb2...</i>	Partições da 2a. unidade na controladora primária – escravo.
<i>hdc</i>	1a. unidade na controladora secundária – mestre.
<i>hdc1, hdc2...</i>	Partições da 1a. unidade na controladora secundária – mestre.
<i>hdd</i>	2a. unidade na controladora secundária – escravo.
<i>hdd1, hdd2...</i>	Partições da 2a. unidade na controladora secundária – escravo.

Quanto aos *CD-ROMs*, na verdade não existem *devices* para acesso as unidades leitoras de *CD-ROMs* atuais, e sim apenas atalhos simbólicos indicando em qual os *devices* reais estes se encontram. Este procedimento é necessário face à utilização dos *devices* para as diferentes localizações (primário, secundário, mestre, escravo, etc.) e antigos *drivers* de *CD-ROMs* que não utilizavam a controladoras *IDE* para serem conectados ao sistema. Para acessá-los, deveremos utilizar o atalho simbólico */dev/cdrom*.

## UNIDADES SCSI

Sem grandes mistérios, estes são os *devices* para as unidades *SCSI*. No *kernel 2.4*, os gravadores de *CD-R/W* e *DVD-R/W* – são acessados através de emulação *SCSI*. Por este motivo, eles devem utilizar estes *devices*.

Normalmente seguem a seguinte nomenclatura:

*hdb*

<b>Unidades SCSI</b>	
<i>sda</i>	1a. unidade nominal (disco rígido, <i>CD-R/RW</i> , memória eletrônica, etc.).
<i>sda1, 2, 3.</i>	Indicativo de particionamento das unidades emuladas.
<i>sda4</i>	Indicativo de unidade, utilizado especificamente pelo <i>Zip-drive</i> .
<i>sdb, sdc...</i>	Unidades sequenciais.

Em tratando-se de disco rígido, segue-se o mesmo padrão das unidades *IDE*, somente atentando-se para alterar o caracter *h* para o caracter *s*.

## DISQUETES

Os disquetes são acessados através dos *devices* */dev/fd[X]*.

<b>Disquetes</b>	
<i>fd0</i>	1a. unidade de disquete – em <i>DOS</i> corresponde à <i>A:</i> .
<i>fd0d360</i>	1a. unid. de disquete, formato baixa densidade, cap. <i>360 KB</i> – corresponde aos disquetes de <i>5.1/4" DD</i> .
<i>fd0h720</i>	1a. unid. de disquete, formato alta densidade, cap. <i>720 KB</i> – corresponde aos disquetes de <i>5.1/4" HD</i> .
<i>fd0u1200</i>	1a. unid. de disquete, formato alta densidade, cap. <i>1200 KB</i> – corresponde aos disquetes de <i>5.1/4", HD</i> .
<i>fd0u1440</i>	1a. unid. de disquete, formato alta densidade, cap. <i>1440 KB</i> – corresponde aos disquetes de <i>3.1/2", HD</i> .
<i>fd1...</i>	2a. unidade de disquete – em <i>DOS</i> corresponde à <i>B:</i> .

De acordo com as necessidades, existirão circunstâncias em que iremos referir-se à um dos *devices* específicos. Por exemplo, para formatação, iremos defini-lo como */dev/fd[X][D]*, onde *[X]* indica o *device* da unidade em questão e *[D]* a densidade da mesma.

## SISTEMA DE ÁUDIO

Em vista dos diferentes recursos de áudio presentes na maioria das placas de som do mercado, os sistemas *GNU/Linux* não utilizam somente um, e sim um conjunto de *devices* para o acesso à estes. Segue abaixo uma simples listagem padrão para uma melhor compreensão.

<b>Sistema de áudio (antigo sistema OSS)</b>	
<i>audio</i>	Funções de áudio (sintetizador <i>wave-table</i> ).
<i>dsp</i>	Voz digitalizada.

<b>Sistema de áudio (antigo sistema OSS)</b>	
<i>midi</i>	Funções de instrumentação ( <i>MIDI</i> ).
<i>mixer</i>	Ajustes e configurações (mixagem).
<i>sequencer</i>	Sequenciador.

Todos estes devices pertencem ao grupo *audio*, onde devemos incluí-lo nas configurações das contas de usuário para que as propriedades de áudio estejam presentes. Para obterem maiores informações de como proceder, consultem na *4a. Parte: Ajustes & Configurações -> Áudio – Placa de som*.

## **FAX-MODEM (PORTAS SERIAIS)**

As tradicionais placas de *fax-modem* – conhecidas popularmente como *hardmodems* – são referidas no sistema através dos *devices* */dev/ttyS[X]*.

<b>Portas seriais</b>	
<i>ttyS0</i>	Porta <i>serial</i> 1 – equivale à <i>COM1</i> .
<i>ttyS1</i>	Porta <i>serial</i> 2 – equivale à <i>COM2</i> .
<i>ttyS2</i>	Porta <i>serial</i> 3 – equivale à <i>COM3</i> .
<i>ttyS3</i>	Porta <i>serial</i> 4 – equivale à <i>COM4</i> .

Já no caso dos *softmodems*, a maior parte destes periféricos utilizam um *device* especial, criado pelos *drivers* para a sua instalação.

Para obterem maiores informações sobre estes periféricos, consultem na *4a. Parte: Ajustes & Configurações -> Modem – placas de fax-modem*.

## **CONSOLE TERMINAL**

Um terminal é uma interface entre o usuário e o sistema. Ao serem inicializados, utiliza o *device* */dev/tty[X]*.

<b>Console terminal</b>	
<i>ttyX</i>	Um terminal propriamente dito.
<i>ttypX</i>	Terminais <i>SSH/Telnet</i> .

Somente teremos à disponibilidade diversos terminais desde que o *kernel* tenha o suporte à terminais virtuais. Felizmente todos os *kernels* são pré-compilados com esta opção habilitada.

## **IMPRESSORA (PORTA PARALELA)**

As impressoras pelo fato de serem instaladas nas portas paralelas, são referenciadas pelo sistema utilizando estes *devices*.



<b><i>Impressoras</i></b>	
<i>lp0, lp1...</i>	Portas paralela – <i>LPT1</i> .

Em se tratando de impressoras que utilizam o barramento *USB*, estes *devices* ficam localizados em */dev/usb*, onde são mantidos as mesmas definições de nomenclatura.

## OUTROS DEVICES IMPORTANTES

Dentre outros *devices* importantes, destacam-se o */dev/dri/card[X]*, os quais as aceleradoras gráficas utilizam para realizar as atividades de renderização gráfica através da extensão *DRI* do servidor gráfico.

## CRIANDO DEVICES

### MAKEDEV & MKNOD

O *MAKEDEV* é apenas um simples *script* que atua como intermediário com o objetivo de facilitar o usuário à criar os *devices* do sistema. O utilitário responsável mesmo pela criação de tais *devices* é o *mknod*.

Para criar *devices* com o *MAKEDEV*, utilizamos a sintaxe...

```
# MAKEDEV /dev/[DEVICE]
```

Simples, prático e extremamente fácil, dispensando comentários adicionais.

Já com o *mknod*, poderemos lançar esta sintaxe:

```
# mknod [PARÂMETROS] [DEVICE] [MAJOR] [MINOR]
```

Onde:

<b><i>mknod</i></b>	
<i>[DEVICE]</i>	O <i>device</i> o qual se deseja criar.
<i>[MAJOR]</i>	Identifica o gerenciador de periféricos.
<i>[MINOR]</i>	Identifica o periférico dentro de um conjunto de periféricos do mesmo gerenciador.
<i>-b</i>	Para criar <i>devices</i> do tipo bloco.
<i>-c</i>	Para criar <i>devices</i> do tipo caracter.
<i>-m[NNN]</i>	Permissões de acesso (veja <i>chmod</i> em <i>Manipulação de arquivos e diretórios</i> ).

Uma utilização básica do comando para criar um *device ttyS0*:

```
# mknod -m 766 /dev/ttyS0 c 4 64
```

Ambos podem (e devem) ser utilizados para restaurar quaisquer *devices* que tenham sido manipulados erroneamente (modificados, apagados, etc.).

Conforme visto a simplicidade acima, dêem preferência ao uso do *script MAKEDEV*, pois o mesmo tornará esta atividade mais simples e produtiva.

## CONCLUSÃO

Acreditamos que um simples passeio pelos *devices* mais utilizados pelo sistema possa deixar os usuários mais familiarizados com a administração e manutenção do sistema operacional em geral. Lembrem-se: os sistemas *Unix* em geral referem-se à quaisquer *devices* de sistema como arquivos. Quaisquer intervenção necessária, sempre tenha consciência de que poderá intervir nos arquivos da estrutura */dev*. &:-D

# III. LINHA DE COMANDO

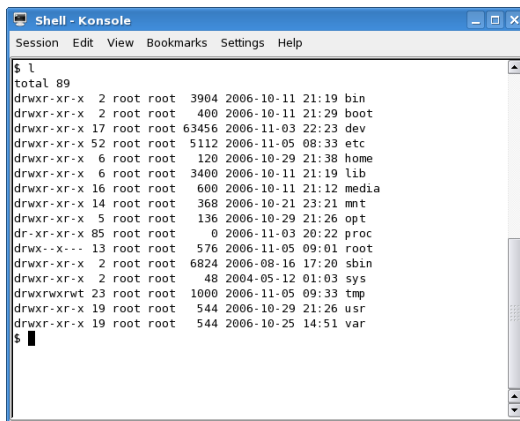
## INTRODUÇÃO

A disponibilidade de diversos utilitários gráficos e em modo texto, facilitam muito a administração de sistemas *GNU/Linux*. Porém, de acordo com as necessidades, facilidades de uso e recursos, existirão diversas circunstâncias em que serão necessários o uso de recursos baseados na linha de comando.<sup>3</sup> Neste capítulo, iremos conhecer alguns recursos e funcionalidades do *BASH*, o interpretador de comando dos sistemas *GNU/Linux*.

## O BASH

✓ <<http://www.gnu.org/software/bash/>>.

O interpretador de comando é um programa especial que permite a interação do usuário com o sistema operacional através da utilização de comandos especiais, que por sua vez são inseridos com interação direta com o teclado. É nele ele que coordenamos muitas das atividades pertinentes ao *kernel*, como a manipulação de arquivos, gerenciamento de processos, entre outros. Nos sistemas *GNU/Linux*, o *BASH* é o seu interpretador de comandos oficial – requerimento do padrão *LSB* – tornando ainda mais importante conhecer suas características e funcionalidades básicas.



```
Shell - Konsole
Session Edit View Bookmarks Settings Help

$ ls
total 89
drwxr-xr-x  2 root root   3904 2006-10-11 21:19 bin
drwxr-xr-x  2 root root    400 2006-10-11 21:29 boot
drwxr-xr-x 17 root root 63456 2006-11-03 22:23 dev
drwxr-xr-x 52 root root  5112 2006-11-05 08:33 etc
drwxr-xr-x  6 root root   120 2006-10-29 21:38 home
drwxr-xr-x  6 root root  3400 2006-10-11 21:19 lib
drwxr-xr-x 16 root root   600 2006-10-11 21:12 media
drwxr-xr-x 14 root root   368 2006-10-21 23:21 mnt
drwxr-xr-x  5 root root   136 2006-10-29 21:26 opt
dr-xr-xr-x 85 root root     0 2006-11-03 20:22 proc
drwx--x--x 13 root root   576 2006-11-05 09:01 root
drwxr-xr-x  2 root root  6824 2006-08-16 17:20 sbin
drwxr-xr-x  2 root root    48 2004-05-12 01:03 sys
drwxrwxrwt 23 root root  1000 2006-11-05 09:33 tmp
drwxr-xr-x 19 root root   544 2006-10-29 21:26 usr
drwxr-xr-x 19 root root   544 2006-10-25 14:51 var
$
```

Terminal virtual (Konsole) do KDE.

<sup>3</sup> Infelizmente muitos usuários têm conceitos errôneos sobre o uso da linha de comando. Já escutamos diversos comentários específicos e sem fundamentos, tais como “isso é coisa do passado...”, “voltar aos tempos do DOS...”, “ninguém mais usa isto...”, “pra quê mexer nessa !%#&\*.”, etc., mas em sistemas *GNU/Linux* ela é essencial e indispensável para o bom funcionamento do sistema.

À primeira vista, o *BASH* lembra-se muito o *MS-DOS*, por ele disponibilizar uma tela de texto preta e uma linha de comando. Mas na verdade, esta ferramenta possui vastos recursos dos quais poderemos tirar grande proveito, conforme nossas necessidades. Neste livro o utilizaremos apenas as instruções básicas e essenciais necessárias para garantirmos a boa manutenção de um sistema para o uso em *desktops*.

## INFORMAÇÕES E MÉTODOS ESSENCIAIS

São inúmeros os recursos disponibilizados pelos interpretadores da linha de comando, onde por isto nos omitiremos de instruções detalhadas e desnecessárias. Seguem abaixo apenas algumas informações e métodos básicos e essenciais para obtermos um excelente rendimento nas intervenções que iremos realizar mais à frente.

### COMPLEMENTO DA TECLA <TAB>

Em muitas circunstâncias teremos a necessidade de digitar toda a nomenclatura de um comando, arquivo ou diretório na linha de comando. Ao invés de teclarmos cada caracter, poderemos apenas digitar as iniciais e complementar com a tecla <TAB>, onde o sistema se encarregará de localizar tal nomenclatura e preencher com os dados restantes. Por exemplo, ao digitarmos...

```
# rpm -ivh quake-1<TAB>
```

... o resultado final será...

```
# rpm -ivh quake-1.1-6cl.i386.rpm _
```

Simple e prático, evitando o inconveniente de ter que redigitar toda a nomenclatura caso ocorram simples erros de digitação e sintaxe. &;-D

### EXPRESSÕES

Expressões são conjuntos de símbolos e caracteres que visam especificar uma determinada informação ou o conjunto delas. Estes são conhecidos popularmente como *curingas*, que podem representar um valor, uma expressão, um conjunto destes e até mesmo um comando. Porém, ao contrário do que muitos pensam, existem diferenças entre eles: os símbolos especiais possuem funções específicas; já os caracteres coringas representam e/ou substituem outros caracteres e/ou conjunto de letras. Descreveremos os principais caracteres utilizados em sistemas *GNU/Linux* e suas principais funcionalidades.

<i>Expressões</i>		
*	<i>Asterisco</i>	Popularmente representa “ <i>tudo</i> ”. Qualquer campo ou instrução que estiver sendo representado por um asterisco, indicará todos os possíveis caracteres.



<b>Expressões</b>		
.	<i>Ponto</i>	Representa entrada de diretórios. Dois pontos - .. - representa o diretório-pai (diretório anterior).
?	<i>Interrogação</i>	Similar ao caracter "*", porém somente representa os caracteres que se situarem na posição onde este se encontra.
/	<i>Pipes</i>	Processa a saída de um comando para que seja usado como dados ou parâmetros em outro comando.  Dois <i>II</i> servem para executar 2 comandos seqüenciais, independente de haver erro no <i>1o.</i> comando. Também utilizado para representar a expressão lógica <i>OR</i> (ou).
&	<i>E-comercial</i>	Execução de aplicações em <i>2o.</i> plano ( <i>background</i> ).  Duas && servem para executar 2 comandos seqüenciais, desde que o primeiro não retorne nenhum erro. Também utilizado para representar a expressão lógica <i>AND</i> (e).

Para obterem maiores informações, consultem o seu manual eletrônico...

```
$ man bash
```

## CORES PERSONALIZADAS

Outra ponto à ser observado são as cores dos arquivos e diretórios à serem exibidos em modo texto, pois os sistemas *GNU/Linux* utilizam como padrão o interpretador de comandos *BASH*, que graças à ele possuem padronizados a seguinte configuração de cores:

<b>Cores &amp; Significado</b>	
<i>Amarelo</i>	Dispositivos do sistema ( <i>device</i> ).
<i>Azul</i>	Diretórios (seguidos pelo caracter "/").
<i>Azul ciano</i>	Atalhos simbólicos ( <i>links</i> ).
<i>Cinza</i>	Arquivos diversos (texto, documentos, etc.) ou desconhecidos.
<i>Magenta</i>	Arquivos de imagens bitmaps ( <i>JPEG, GIF, PNG</i> , etc.).
<i>Verde</i>	Arquivos executáveis (arquivos de lote e binários). <sup>4</sup>
<i>Vermelho</i>	Arquivos compactados (inclusive pacotes de instalação).

Vale lembrar que, dependendo tanto dos atributos específicos dos arquivos, quanto das configurações utilizadas no terminal, muitos poderão ter cores diferentes dos padrões acima citados. É o caso de arquivos com atributos para execução (*flag x*), que aparecem com a cor verde no vídeo, seja uma

<sup>4</sup> Estes indicam arquivos com permissões para execução, ainda que sejam um texto, uma imagem, um pacote compactado e outros quaisquer.

imagem, um arquivo compactado, etc.

## NOMENCLATURA DIFERENCIADA

Da mesma forma que os arquivos e diretórios apresentam um sistema de cores para a sua identificação, os mesmos possuem sinais especiais para informar determinados atributos:

<i>Sinal / Significado / Exemplo</i>		
.	Arquivo oculto.	.Confidencial
*	Arquivo executável.	Programa*
@	Atalho.	Programa@ -> Programa-1.0.2

Lembre-se que de acordo com o interpretador de comando utilizado, poderão ser ou não exibidos tais atributos.

## UMA SIMBOLOGIA ESPECIAL

Para evitarmos a alocação de espaços com informações desnecessárias, utilizaremos neste livro a seqüência de caracteres “-/-” para simbolizar os resultados obtidos com o uso dos comandos mencionados. Por exemplo, na necessidade de exibir um longo texto ou uma longa lista de arquivos, estes serão resumidos da seguinte forma:

```
# less COPYING
                                GNU GENERAL PUBLIC LICENSE
                                Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
    59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
-/-
```

Onde “-/-” representará o restante do conteúdo de texto. Já para a listagem de grandes estruturas composta de diretórios e/ou arquivos:

```
# cd /usr/doc/
# ls -l
-/-
```

Onde “-/-” representará a estrutura arquivos e/ou diretórios não visíveis.

## EQUIVALÊNCIAS ENTRE O BASH E MS-DOS

Apesar de se encontrar praticamente em desuso por usuários do *Windows*, muitos usuários sentem-se confortáveis ao realizar diversas intervenções com a utilização da linha de comando do *MS-DOS*, especialmente quando não há possibilidade de inicializar o *Windows* para estes eventos. Já em sistemas *GNU/Linux*, a linha de comando não só se encontra presente, como também é fundamental para a manutenção geral do sistema e ainda

possui uma eficiência muito superior à que encontramos no velho *MS-DOS*.

## ESTRUTURA DE DIRETÓRIOS

A principal diferença entre os sistemas *GNU/Linux* e o *MS-DOS* encontra-se na estrutura de diretórios do sistema. Enquanto o *MS-DOS* suporta somente a formato de nomes *8.3* (até a versão *6.22*) e dispõe somente de permissões de acesso para somente leitura e ocultação (atributos)...

```
C:\>dir
  O volume na unidade C é DARKSTAR
  O número de série do volume é 0816-A972

  Pasta de C:\

12/05/2004  20:03    <DIR>          WINDOWS
24/05/2004  23:23    <DIR>          Documents and Settings
12/05/2004  20:18    <DIR>          Arquivos de programas
12/05/2004  20:19                0 CONFIG.SYS
12/05/2004  20:19                0 AUTOEXEC.BAT
                2 arquivo(s)                0 bytes
                3 pasta(s) 3.165.028.352 bytes disponíveis


C:\>_
```

... o *BASH* suporta nomes com até 255 caracteres, além do simples e eficiente sistema permissões de acesso de leitura, escrita e execução.

```
$ ls -l
total 82
drwxr-xr-x  2 root    bin          2304 Jun 28 17:09 bin/
drwxr-xr-x  2 root    root         336 Set  5 22:14 boot/
drwxr-xr-x 15 root    root        61072 Set  6 09:58 dev/
drwxr-xr-x 45 root    root        4520 Set  6 10:36 etc/
drwxr-xr-x  5 root    root         128 Ago 20 23:11 home/
drwxr-xr-x  4 root    root        2520 Jun 28 17:08 lib/
drwxr-xr-x  6 root    root         144 Jun 28 17:16 mnt/
drwxr-xr-x  4 root    root          96 Ago 23 18:48 opt/
dr-xr-xr-x 73 root    root           0 Set  6 06:57 proc/
drwx--x--- 25 root    root        1064 Set  5 13:55 root/
drwxr-xr-x  2 root    bin         5456 Jun  1 2002/sbin/
drwxrwxrwt 31 root    root        1352 Set  6 10:09 tmp/
drwxr-xr-x 21 root    root         592 Mar  6 2003/usr/
drwxr-xr-x 17 root    root         456 Mar  2 2003/var/
$ _
```

Além destes, existem outros recursos presentes em sua linha de comando.

## ACESSO ÀS UNIDADES DO SISTEMA

O *MS-DOS* atribui letras para a unidade de armazenamento de dados (*A:*, *C:*, *D:*, etc.); já os sistemas *GNU/Linux* possui apenas um único diretório raiz, do qual as demais unidades encontram-se previamente montadas em seus respectivos subdiretórios. Ainda no *MS-DOS*, para ter acesso às demais unidades do sistema, bastaria apenas indicá-las na linha de comando acrescido de <DOIS\_PONTOS> ("*..*"): 

```
C:\> A:
A:\> D:
D:\> C:
C:\> _
```

No *BASH*, para ter acesso às demais unidades do sistema, basta montar os dispositivos e acessá-los através do ponto de montagem, situados no diretório */mnt/<UNIDADE>* conforme a norma *FHS*. O ponto de montagem */mnt* possui basicamente a seguinte estrutura:

```
$ cd /mnt/
$ ls -l
total 2
drwxr-xr-x  2 root    root          48 Mar 16  2002 cdrom/
drwxr-xr-x  2 root    root          48 Mar 16  2002 floppy/
drwxr-xr-x  2 root    root          48 Mar 16  2002 hd/
$ _
```

Por padrão existe apenas estes diretórios, porém caso necessitem trabalhar com outros dispositivos do sistema, basta criá-los conforme a necessidade. Lembrem-se de que precisarão estar com os poderes de superusuário.

```
$ su
Password:
# mkdir flash
# mkdir zip
# mkdir tape
# ls -l
total 3
drwxr-xr-x  2 root    root          48 Mar 16  2002 cdrom
drwxr-xr-x  2 root    root          48 Mar 16  2002 floppy
drwxr-xr-x  2 root    root          48 Mar 16  2002 hd
drwxr-xr-x  2 root    root          48 Set  7 17:01 flash
drwxr-xr-x  2 root    root          48 Set  7 17:01 tape
drwxr-xr-x  2 root    root          48 Set  7 17:01 zip
# _
```

Após a montagem das partições e/ou unidades, seus respectivos acessos são realizados normalmente, como se fossem simples subdiretórios.

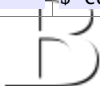
## ESPECIFICAÇÕES DO DIRETÓRIO

Nos sistemas *GNU/Linux*, a especificação dos caminhos de diretórios é feita de forma similar ao *MS-DOS*, porém outra diferença simples está no uso da *<BARRA>* - / - ao invés da *<BARRA\_INVERTIDA>* - | - para a navegação entre os diretórios.

<i><b>MS-DOS</b></i>	<i><b>BASH</b></i>
C:\> cd <UNIDADE>\<DIRETÓRIO>	\$ cd /<DIRETÓRIO>

Em ambos, para navegar até o diretório raiz:

<i><b>MS-DOS</b></i>	<i><b>BASH</b></i>
C:\WINDOWS> cd \	\$ cd /



# EXIBIÇÃO DE CAMINHO

Em ambos os interpretadores – e de acordo com a distribuição utilizada – o *prompt* da linha de comando indicam em que diretório estamos...

<i>MS-DOS</i>	<i>BASH</i>
C:\WINDOWS> _	darkstar@darkstar:/usr/doc\$ _

... tendo o *BASH* a vantagem de informar a conta autenticada no momento.

## CASE SENSITIVE

O sistemas de nomenclatura dos arquivos e diretórios do *GNU/Linux* é *case sensitive*, ou seja, o tratamento de caracteres maiúsculos e minúsculos é diferente em comparação ao *MS-DOS*, onde neste tanto faz utilizar caracteres maiúsculos quanto minúsculos.

<i>MS-DOS</i>	<i>BASH</i>
C:\> CD WINDOWS C:\WINDOWS\> CD .. C:\> cd windows C:\WINDOWS\> _	darkstar@darkstar:/\$ cd /usr darkstar@darkstar:/usr\$ CD .. -bash: CD: command not found darkstar@darkstar:/usr\$ _

Já no *BASH*, o sistema não reconhecerá o comando utilizado caso utilizem caracteres maiúsculos ao invés de maiúsculos, e vice-versa.

## OBSERVAÇÕES

Apesar da aparência “*pobre*” e espartana da linha de comando, deveremos nos aperfeiçoar na utilização de comandos básicos. As aplicações disponibilizadas em interfaces gráficas geralmente tem a simpatia dos usuários menos habituados à linha de comando, graças à facilidades condicionadas, como a intuitividade e o apelo visual, tornando bem mais atrativo e agradável realizar as operações necessárias no sistema, porém existirão circunstâncias específicas e variadas em que a utilização de uma linha de comando será praticamente insubstituível. E mesmo apesar do vastos recursos disponibilizados nas atuais aplicações gráficas, em muitos aspectos teremos maior eficiência, prática e agilidade em realizar as operações necessárias na linha de comando.

Para reforçar, uma das principais vantagens na utilização da linha de comando é a possibilidade de evitar a perda de tempo na realização de tarefas, pois ao invés de gastar bons segundos na navegação nos menus com rápidos comandos, poderemos realizar a maioria das atividades mais corriqueiras existentes. E se estivermos em uma interface gráfica? Para isto, poderemos utilizar os terminais disponibilizados em nosso ambiente gráfico preferido, como exemplo, o *Konsole* do *KDE*, entre outros. Enfim, a escolha fica à critério do usuário, visto que estarão disponíveis todas as funcionalidades básicas necessárias.



## CONCLUSÃO

Apesar da existência de diversas diferenças em comparação à linha de comando do *MS-DOS*, o objetivo o *BASH* é o mesmo: fornecer ao usuário um conjunto de recursos que poderão ser habilitados pela linha de comando. Porém são inúmeras as diferenças e características, além da disponibilidade de recursos extras que o tornam muito superior ao *MS-DOS* e quaisquer outros aplicativos gráficos existentes. &;-D

# IV. MANIPULAÇÃO DE ARQUIVOS E DIRETÓRIOS

## INTRODUÇÃO

Nas mais variadas circunstâncias, a manipulação direta de arquivos e diretórios serão necessários para a realização de diversas intervenções com as mais variadas finalidades. A checagem de conteúdo, a organização de documentos, a edição direta, a definição de permissões, etc., são simples e bons exemplos de operações realizadas em imensas possibilidades.

Neste capítulo, apresentaremos as principais operações de manipulação, que por sua vez se encontram subdividas por categorias, onde suas respectivas instruções estão estruturadas de forma simples e organizada para uma melhor compreensão.

## OPERAÇÕES E ATIVIDADES BÁSICAS

Entende-se como atividades básicas a listagem, visualização, edição e manipulação, dos arquivos e diretórios gerais que compõe o sistema, além de dados e informações adicionais.

## LISTAGEM E NAVEGAÇÃO

### LS

Lista os arquivos e diretórios do diretório corrente ou especificado. Bastante similar ao comando *dir/w* do *DOS*.

Sintaxe:

`ls [PARÂMETROS] [ARQUIVOS/DIRETÓRIOS]`

Onde:

<i>ls</i>	
-a	Exibe todos os arquivos e diretórios ocultos, similar ao comando <i>dir /a</i> do <i>DOS</i> . Em muitos casos observaremos a existência do caracter “.” antes de muitos arquivos e diretórios que até antes não “ <i>constavam</i> ” no diretório com o simples uso do comando <i>ls</i> . Na verdade eles existiam, porém estavam somente ocultos.
-R	Exibe todos os arquivos de diretórios e diretórios de forma “ <i>recursiva</i> ”, ou seja, além dos diretórios existentes, mostra também todos os subdiretórios e arquivos contidos nos diretórios exibidos. Similar parâmetro <i>/s</i> no <i>DOS</i> . Ao clicarem na barra de rolagem, poderemos observar todo o conteúdo do diretório analisado, sendo somente limitado à aqueles não permitidos pelos seus respectivos atributos.



<i>ls</i>	
-i	Utilizado para verificar somente o espaço ocupado em <i>i-node</i> (blocos) pelos arquivos e diretórios. Observem atentamente que, de acordo com o sistema de arquivo utilizado, esta informação refere-se apenas à quantidade de blocos utilizados pelos arquivos, e não o seu tamanho ocupado. Estamos utilizando o sistema de arquivos <i>ReiserFS</i> , porém se optarem por outro sistema – muito provavelmente o <i>ext3</i> –, os valores exibidos serão diferentes, mesmo que seus conteúdos sejam exatamente os mesmos.
-l	Exibe a listagem de arquivos e diretórios em uma única coluna, contendo ainda informações complementares como as permissões referentes de cada item e seu respectivo espaço ocupado. Equivalente ao simples comando <i>dir</i> no <i>DOS</i> .
-l	Utilizado apenas para exibir a listagem de arquivos e diretórios em uma única coluna, similar ao comando <i>dir/b</i> no <i>DOS</i> .

Observem que os parâmetros exemplificados podem ser utilizados combinados, conforme às nossas necessidades. Experimentem utilizar estes parâmetros combinado com outros parâmetros e verifiquem sua saída:

```
$ ls -li
total 83
 11 drwxr-xr-x  2 root    bin          2416 2004-01-31 07:43 bin/
 20 drwxr-xr-x  2 root    root          336 2004-01-31 07:55 boot/
  9 drwxr-xr-x 16 root    root        62352 2004-01-31 12:09 dev/
 12 drwxr-xr-x 47 root    root         4808 2004-01-31 15:06 etc/
  2 drwxr-xr-x  5 root    root           96 2004-01-31 10:27 home/
 15 drwxr-xr-x  4 root    root        2592 2004-01-31 07:42 lib/
 16 drwxr-xr-x  5 root    root         120 2004-01-31 10:08 mnt/
11750 drwxr-xr-x  4 root    root           96 2002-12-13 19:01 opt/
  1 dr-xr-xr-x 86 root    root           0 2004-01-31 10:09 proc/
  7 drwx--x--- 11 root    root         656 2004-01-31 14:31 root/
 22 drwxr-xr-x  2 root    bin        5704 2003-09-01 19:29/sbin/
  2 drwxrwxrwt 19 root    root         648 2004-01-31 15:17 tmp/
  2 drwxr-xr-x 19 root    root         544 2004-01-05 18:32 usr/
  2 drwxr-xr-x 17 root    root         456 2003-08-15 00:17 var/

$ _
```

Além do comando *ls*, podemos também utilizar os comandos *dir*...

```
$ dir
bin/  dev/  home/  mnt/  proc/ /sbin/  usr/
boot/ etc/  lib/   opt/  root/  tmp/   var/

$ _
```

... e *vdir*, que correspondem respectivamente à *ls* e *ls -l*.

```
$ vdir
total 83
drwxr-xr-x  2 root    bin          2416 2004-01-31 07:43 bin/
drwxr-xr-x  2 root    root          336 2004-01-31 07:55 boot/
drwxr-xr-x 16 root    root        62352 2004-01-31 12:09 dev/
drwxr-xr-x 47 root    root         4808 2004-01-31 15:06 etc/
drwxr-xr-x  5 root    root           96 2004-01-31 10:27 home/
```





```
drwxr-xr-x  4 root    root    2592 2004-01-31 07:42 lib/
drwxr-xr-x  5 root    root     120 2004-01-31 10:08 mnt/
drwxr-xr-x  4 root    root      96 2002-12-13 19:01 opt/
dr-xr-xr-x 93 root    root      0 2004-01-31 10:09 proc/
drwx--x--- 11 root    root    656 2004-01-31 14:31 root/
drwxr-xr-x  2 root    bin    5704 2003-09-01 19:29/sbin/
drwxrwxrwt 19 root    root    648 2004-01-31 15:17 tmp/
drwxr-xr-x 19 root    root    544 2004-01-05 18:32 usr/
drwxr-xr-x 17 root    root    456 2003-08-15 00:17 var/
$ _
```

Os comandos *ls -l* e *vdirc* são os mais indicados para verificar todos as informações disponíveis sobre a estrutura de arquivos e diretórios consultados, mas para evitar digitar esta seqüência de caracteres, utilizem o apelido *l*, habilitando-o com o comando...

```
$ alias l="ls -l"
```

Após isto, bastará digitar o novo apelido...

```
$ l
total 83
drwxr-xr-x  2 root    bin    2416 2004-01-31 07:43 bin/
drwxr-xr-x  2 root    root     336 2004-01-31 07:55 boot/
drwxr-xr-x 16 root    root  62352 2004-01-31 12:09 dev/
drwxr-xr-x 47 root    root   4808 2004-01-31 15:06 etc/
drwxr-xr-x  5 root    root     96 2004-01-31 10:27 home/
drwxr-xr-x  4 root    root   2592 2004-01-31 07:42 lib/
drwxr-xr-x  5 root    root    120 2004-01-31 10:08 mnt/
drwxr-xr-x  4 root    root     96 2002-12-13 19:01 opt/
dr-xr-xr-x 86 root    root      0 2004-01-31 10:09 proc/
drwx--x--- 11 root    root    656 2004-01-31 14:31 root/
drwxr-xr-x  2 root    bin    5704 2003-09-01 19:29/sbin/
drwxrwxrwt 19 root    root    648 2004-01-31 15:17 tmp/
drwxr-xr-x 19 root    root    544 2004-01-05 18:32 usr/
drwxr-xr-x 17 root    root    456 2003-08-15 00:17 var/
$ _
```

... ao invés da referida e desconfortável quantidade de parâmetros acima expostos para obtermos os mesmos resultados.

## CD

O comando *cd* é utilizado para navegar entre os diretórios desejados.

Sintaxe:

```
$ cd [NOME_DO_DIRETÓRIO]
```

Exemplo:

```
darkstar@darkstar:~$ cd /usr
darkstar@darkstar:/usr$ ls -l
-//-
darkstar@darkstar:/usr$ cd lib
darkstar@darkstar:/usr/lib$ ls -l
-//-
darkstar@darkstar:/usr/lib$ cd X11
darkstar@darkstar:/usr/lib/X11$ _
```



Observe que, para entrarmos no subdiretório *lib* e *X11*, não foram especificados os caminhos completos, pois estes pertencem ao diretório do qual se encontra localizado no momento (referência local).

Como podem ver, seu uso é bastante simples, não existindo qualquer grau de dificuldade. Ainda assim podemos utilizar outros recursos, tais como:

```
darkstar@darkstar:/usr/lib/X11$ cd ..
darkstar@darkstar:/usr/lib$ cd /
darkstar@darkstar:/$ _
```

É bem mais simples que digitar o caminho completo para acessar o diretório anterior. Observem ainda que podemos “*desfazer*” a ação anterior, ou seja, retornar para o diretório onde estávamos anteriormente com o uso do parâmetro - (hífen).

```
darkstar@darkstar:/usr/lib$ cd /
darkstar@darkstar:/$ cd -
/usr/lib
darkstar@darkstar:/usr/lib$ _
```

Para retornarem ao diretório *\$HOME* específico do usuário (neste caso, */home/darkstar/*), utilizem o parâmetro ~ (til) ou ainda apenas *cd...*

```
darkstar@darkstar:/usr/lib$ cd ~
darkstar@darkstar~$ _
```

...ao invés de utilizar o comando e definir todo o caminho */home/darkstar/*.

## VISUALIZAÇÃO

### TYPE

Permite visualizar o conteúdo de arquivos-textos.

Sintaxe:

```
$ type [ARQUIVO]
```

As mesmas regras do *MS-DOS* também são aplicadas aqui: os arquivos textos serão exibidos normalmente, ao contrário dos arquivos binários.

### LESS

Possui a mesma finalidade que o comando *type* – visualização de conteúdos –, porém possibilita realizar a paginação como a utilização das teclas <SETA\_ACIMA>, <SETA\_ABAIXO>, <PÁGINA\_ACIMA> e <PÁGINA\_ABAIXO> para rolar o texto à ser visualizado.

Sintaxe:

```
$ less [PARÂMETROS] [ARQUIVO]
```

O recurso de paginação é muito útil para visualizar o conteúdo de arquivos com grande quantidade de textos. Existem diversos parâmetros, mas caso queiram realizar uma consulta mais refinada, experimentem digitar...



```
$ less --help
-//-
```

... e analisem os resultados exibidos.

## FILE

A principal finalidade do comando *file* é exibir informações sobre o tipo de arquivo desejado. Muito importante para circunstâncias em que não sabemos suas origens ou do que se tratam.

Sintaxe:

```
$ file [ARQUIVO]
```

Observem atentamente os exemplos abaixo para uma melhor compreensão.

```
$ file fotografia
fotografia: JPEG image data, JFIF standard 1.01, resolution (DPI), 72 x 72
$ _
```

O comando *file* detectou que o arquivo *fotografia* trata-se de uma imagem gravada no formato *JPEG*, fornecendo ainda outras informações adicionais.

```
$ file tutorial
tutorial: HTML document text
$ _
```

Novamente o comando *file* detectou a existência do arquivo-texto *tutorial* informando ainda que trata-se de um documento *HTML*, como também poderia ser um uma especificação dentro dos padrões *XML*. Logo abaixo, segue o resultado de um arquivo escrito com as especificações do *XHTML*.

```
$ file index.html
index.html: XML 1.0 document text
$ _
```

Simple e prático, não? &;-D

## PWD

Este comando apenas mostra em qual diretório a linha de comando está situado. Basta apenas digitarmos...

```
$ pwd
/home/darkstar
$ _
```

... para obtermos as informações desejadas.

Na linha de comando do *Slackware*, este comando é desnecessário, pois o próprio sinal de prontidão já exibe a sua localização.

```
darkstar@darkstar: /usr/local$ _
```

Porém existirão circunstâncias em que este, ao invés de exibir o caminho o qual se encontra, exibirá apenas o sinal *~*. Isto significa que nos encontramos no diretório raiz do usuário autenticado.

```
darkstar@darkstar: ~$ pwd
```

```
/home/darkstar
darkstar@darkstar:~$ _
```

## MORE

O comando *more* nada mais é do que um filtro paginador das informações exibidas no vídeo. Serve para realizar pausa de informações exibidas seqüencialmente na tela, como por exemplo, a listagem de um diretório que possui uma imensa quantidade de itens ou a exibição de conteúdo de arquivos-textos pausadamente.

Sintaxe:

```
$ [COMANDO] [PARÂMETROS] | more
```

Ou...

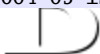
```
$ more [ARQUIVO_TEXTO]
```

Existem diversas situações em que o uso deste comando será de bastante utilidade, mas basicamente o utilizaremos apenas para realizar uma listagem pausada. Vejam o exemplo à seguir:

```
$ ls -l /usr/share/ | more
total 520
drwxr-xr-x  7 root root    296 2004-05-07 22:32 AbiSuite-2.0
drwxr-xr-x  2 root root     48 2004-08-07 20:16 ImageMagick
drwxr-xr-x  5 root root    264 2004-08-07 20:16 ImageMagick-6.0.4
-rw-r--r--  1 root root    600 2004-04-23 18:59 Ssh.bin
drwxr-xr-x  2 root root    176 2003-02-11 22:41 WINGS
drwxr-xr-x  8 root root   1600 2003-02-11 22:41 WindowMaker
drwxr-xr-x  8 root root    216 2002-02-24 17:38 a2ps
drwxr-xr-x  2 root root   2208 2004-09-19 21:07 aclocal
drwxr-xr-x  2 root root    968 2004-05-21 03:38 aclocal-1.8
drwxr-xr-x  4 root root    160 2004-05-29 17:06 alsa
drwxr-xr-x  2 root root    536 2004-06-07 19:36 application-registry
drwxr-xr-x  2 root root   2320 2004-10-22 20:42 applications
drwxr-xr-x  6 root root    168 2003-01-07 20:15 apsfiler
drwxr-xr-x  2 root root   1112 2003-01-13 22:27 aspell
drwxr-xr-x  2 root root    216 2003-08-29 03:17 aumix
drwxr-xr-x  7 root root    224 2004-02-15 23:44 autoconf
drwxr-xr-x  4 root root    648 2004-05-21 03:38 automake-1.8
drwxr-xr-x  2 root root    608 2003-07-28 18:30 awk
drwxr-xr-x  2 root root     88 2004-05-13 17:27 battstat_applet
drwxr-xr-x  2 root root    112 2002-10-15 22:19 bison
drwxr-xr-x  2 root root    200 2004-05-16 20:47 blackjack
--Mais--
```

O diretório */usr/share* possui uma infinidade de diretório em sua estrutura. Uma simples listagem iria exibir todo o seu conteúdo, porém não realizaria uma pausa para a verificação, de forma que apenas conseguiríamos ver as informações da saída de vídeo do final da operação. Para continuar com a exibição dos dados, basta apenas pressionar <BARRA\_DE\_ESPAÇO>:

```
drwxr-xr-x  2 root root    200 2004-05-16 20:47 blackjack
drwxr-xr-x  3 root root    224 2004-05-07 16:02 bug-buddy
drwxr-xr-x  2 root root     72 2004-06-08 16:57 cdrdao
drwxr-xr-x  5 root root    128 2004-05-13 03:47 control-center
```



```
drwxr-xr-x 7 root root 176 2004-06-19 21:09 control-center-2.0
drwxr-xr-x 8 root root 232 2004-09-19 20:50 cups
drwxr-xr-x 2 root root 88 2004-06-06 22:48 curl
drwxr-xr-x 3 root root 72 2004-06-09 15:28 cvs
drwxr-xr-x 4 root root 544 2004-11-15 10:24 d4x
drwxr-xr-x 2 root root 72 2003-03-13 22:00 dict
drwxr-xr-x 3 root root 72 2004-05-02 19:22 distcc
lrwxrwxrwx 1 root root 6 2004-11-01 16:27 doc -> ../doc
drwxr-xr-x 2 root root 3120 2004-06-19 20:52 eazel-engine
drwxr-xr-x 7 root root 888 2004-02-22 03:25 elvis-2.2_0
drwxr-xr-x 4 root root 104 2003-05-23 02:28 emacs
drwxr-xr-x 4 root root 936 2002-02-26 02:05 encrypt
drwxr-xr-x 3 root root 104 2004-05-07 16:40 eog
drwxr-xr-x 4 root root 96 2004-02-21 23:24 epic
drwxr-xr-x 4 root root 496 2004-06-19 17:27 epiphany
drwxr-xr-x 4 root root 136 2004-06-19 18:50 epiphany-extensions
drwxr-xr-x 2 root root 48 2004-06-19 19:52 faces
drwxr-xr-x 3 root root 72 2004-05-07 17:22 file-roller
drwxr-xr-x 3 root root 104 2004-11-01 16:40 fonts
--Mais--
```

Ao pressionarmos <ENTER>, apenas será exibida a linha seguinte:

```
drwxr-xr-x 3 root root 72 2004-05-07 17:22 file-roller
drwxr-xr-x 3 root root 104 2004-11-01 16:40 fonts
drwxr-xr-x 2 root root 2272 2004-06-19 02:08 galeon
--Mais--
```

Já na leitura de um arquivo texto, o comando *more* se comportaria da mesma forma:

```
$ more /mnt/cdrom/COPYRIGHT.TXT
```

```
The Linux(R) kernel is Copyright 1991, 1992, 1993, 1994, 1995,
1996, 1997, 1998 Linus Torvalds (others hold copyrights on some
of the drivers, filesystems, and other parts of the kernel) and
is licensed under the terms of the GNU General Public License.
```

```
LINUX is a registered trademark of Linus Torvalds.
```

```
(see COPYING in /usr/src/linux)
```

```
-----

Many other software packages included in Slackware are licensed under the
GNU
General Public License, which is included in the file COPYING.
```

```
-----

This product includes software developed by the University of
California, Berkeley and its contributors:
```

```
Copyright (c) 1980,1983,1986,1988,1990,1991 The Regents of the University
of California. All rights reserved.
```

```
--Mais--(8%)
```

Observem a existência dos caracteres *--Mais--*, indicando a existência de



mais informações após as exibidas, diferenciando-se apenas a presença de um valor percentual na exibição do conteúdo de arquivos-textos, indicando a proporção das informações já visualizadas durante todo o processo.

Reforçando novamente, ao tecarmos <ENTER> teremos acesso à linha seguinte das informações à serem exibidas na tela, similar à utilização da tecla <SETA\_ABAIXO>; já tecando <BARRA\_DE\_ESPAÇO>, as informações posteriores ocultas preencherão toda a área da tela, similar à utilização da tecla <PÁGINA\_ABAIXO>.

Para realizarmos uma pausa durante a listagem de arquivos e diretórios, o comando *more* auxilia bastante; porém para a exibição do conteúdo de arquivos-textos, o comando *less* proporciona melhor conforto, pois diferente deste último, o comando *more* não permite que se possa retornar as informações passadas com as teclas <SETA\_ACIMA> e <PÁGINA\_ACIMA>.

## DU

Mostra o espaço ocupado por um arquivo ou conjunto de arquivos.

Sintaxe:

```
$ du [PARÂMETROS] [ARQUIVO/DIRETÓRIO]
```

Onde:

<i>du</i>	
<i>-a</i>	Mostra não só apenas o tamanho dos diretórios, como também dos arquivos encontrados.
<i>-b, -k, -m</i>	Mostra o tamanho dos arquivos e diretórios em <i>bytes</i> , <i>KB</i> e <i>MB</i> respectivamente (valor padrão: <i>byte</i> ).
<i>-c</i>	Acrescenta uma linha mostrando o tamanho total de todos os arquivos e diretórios.

Existem diversos outros parâmetros úteis de acordo com as circunstâncias, mas para nós, simples usuários, nos interessa apenas conhecer sua simples e direta utilização, que por sua vez mostrará apenas o tamanho do arquivo e diretórios neles aplicados:

```
$ du texto.txt
104  texto.txt
$ _
```

Neste caso, o arquivo *texto.txt* possui apenas *104 KB*. Já neste outro...

```
$ du "Prova 3o. Bimestre"/
149  Prova 3o. Bimestre/Português
245  Prova 3o. Bimestre/Matemática
394  Prova 3o. Bimestre/
$ _
```

... o comando mostra os tamanhos de cada subdiretórios, além do total do



diretório principal consultado, contando com o espaço ocupado de todos os subdiretórios.

## MANIPULAÇÃO

### MKDIR

Este comando é utilizado unicamente para criar diretórios.

```
$ mkdir [PARÂMETROS] [NOME_DO_DIRETÓRIO]
```

Exemplo:

```
$ mkdir /home/darkstar/teste
$ ls -l /home/darkstar/
total 3
drwx-----  3 darkstar users      128 Ago 16 00:25 Desktop/
drwxr-xr-x  26 darkstar users    1008 Ago  9 22:12 docs/
drwxr-xr-x  3 darkstar users     320 Jul 11 18:54 OpenOffice.org1.0.0/
drwxr-xr-x  2 darkstar users     48 Ago 16 20:25 teste/
drwxr-xr-x  5 darkstar users     256 Ago 16 20:16 z/
$ _
```

Para criar um conjunto de diretórios e respectivos subdiretórios diretamente com um único comando, utilize o parâmetro *-p*.

```
$ mkdir -p /[DIRETÓRIO]/[SUBDIRETÓRIO]/
```

Exemplo:

```
$ mkdir -p teste/subteste/
$ _
```

### DD

Abreviatura de *direct copy*, o comando *dd* possui a finalidade de copiar e transferir dados utilizando as especificações de bloco de entrada e saída, além de inúmeras outras finalidades. Basicamente é utilizado para realizar cópia de arquivos e transferência de dados conforme sua estrutura. Sua sintaxe básica é:

```
$ dd if=[ARQUIVO/DADOS] of=[ARQUIVO/DADOS]
```

Onde estes dados serão formatados de acordo com as definições dos parâmetros de entrada (*if*) e saída (*of*). Observe este simples exemplo para a cópia de um arquivo:

```
$ dd if=/dev/hda9 of=/mnt/flash
```

Este comando copiará todo o conteúdo da partição *hda9* (o diretório */home*) para um dispositivo de memória eletrônica (*Pendrive*).

### CP

Realiza a cópia de arquivos e/ou o conteúdo de um diretório. Sintaxe:

```
$ cp [PARÂMETROS] [ORIGEM] [DESTINO]
```

Sua utilização é tão simples quanto o tradicional comando *copy* do *MS-DOS*, tendo apenas o diferencial de possuir diversos parâmetros, como segue:

<i>cp</i>	
-f	Sobrescreve o arquivo, caso já exista no local de destino.
-p	Preserva os atributos de permissões e grupos de acesso originais do arquivo copiado.
-r	Realiza a cópia de diretórios (recursivo).

Segue um simples exemplo para sua utilização:

```
$ cp /home/darkstar/docs/texto.txt /mnt/floppy
```

O exemplo acima copia o arquivo *texto.txt* para o ponto de montagem */mnt/floppy* (um disquete que deverá estar previamente montado).

### MV

Abreviatura de *move*, movimenta o(s) arquivo(s) desejado(s) para o local desejado ou renomeia o arquivo ou diretório desejado.

```
$ mv [PARÂMETROS] [ORIGEM] [DESTINO]
```

Exemplo:

```
$ mv /home/darkstar/teste/* /home/darkstar/
```

O comando *mv* também pode ser utilizado para renomear diretórios...

```
$ mv /home/darkstar/teste/ /home/darkstar/ok
```

... e arquivos...

```
$ mv /home/darkstar/rasura /home/darkstar/texto.txt
```

### LN

Cria atalhos para apontar determinados arquivos ou diretórios do sistema. Possuem funcionalidades similares aos tradicionais atalhos do *Windows* (*.lnk*), porém bem mais poderosos.

Sintaxe:

```
$ ln [PARÂMETROS] [ORIGEM] [DESTINO]
```

Onde:

<i>ln</i>	
-s	Atalho simbólico.
-d	Atalho físico (somente disponível para o superusuário).

A diferença entre atalhos simbólicos para atalhos físicos (ou *hardlinks*) está na manipulação direta do mesmo. O atalho simbólico apenas fornece um caminho apontado por ele; já o atalho físico faz uma referência direta,





sendo perfeitamente idêntico ao arquivo original em tamanho e permissões de acesso. Sua única limitação está no fato de não fazer referências à diretórios ou arquivos de outras partições.

Uma das principais vantagens da utilização do atalho é a possibilidade de fazer referências à determinados arquivos, ao invés de realizar outras intervenções mais elaboradas. Por exemplo, na necessidade de dispor do navegador *firefox* para todos os usuários, basta simplesmente...

```
# ln -s /usr/local/firefox/firefox /usr/bin/firefox
```

... ao invés de atualizar a variável *\$PATH* para o caminho do *Firefox*.

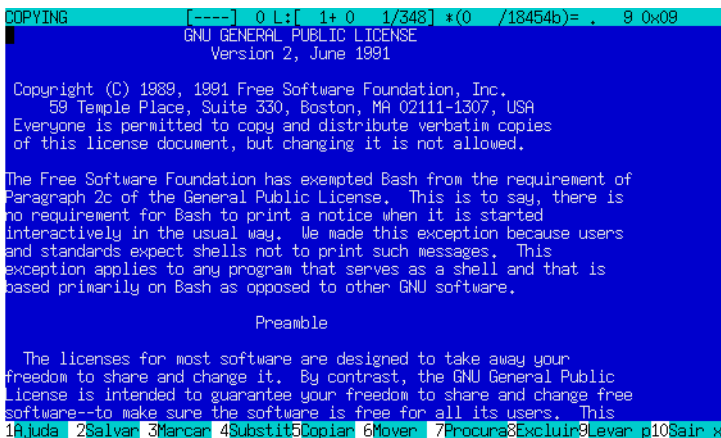
## EDITORIAÇÃO

Muitas das vezes, os arquivos que manipulamos em sistemas *GNU/Linux* são textos, que por sua vez são passíveis de intervenções especiais, em que a edição direta é a principal delas.

Nos sistemas *GNU/Linux*, além dos tradicionais editores de textos em modo gráfico, existem ferramentas interessantes como o *MCedit*.

### MCEDIT

Componente indispensável das ferramentas *GNU Midnight Commander*, o *MCedit* é um excelente editor de textos *ASCII*, com uma aparência muito similar ao já conhecido *Edit*, do *MS-DOS*.



```
COPYING [----] 0 L: [ 1+ 0 1/348] *(0 /18454b)= . 9 0x09
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

The Free Software Foundation has exempted Bash from the requirement of
Paragraph 2c of the General Public License. This is to say, there is
no requirement for Bash to print a notice when it is started
interactively in the usual way. We made this exception because users
and standards expect shells not to print such messages. This
exception applies to any program that serves as a shell and that is
based primarily on Bash as opposed to other GNU software.

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users. This
```

Dentre seus comandos, para acionar o menu principal, basta teclarmos <F9>; para que o programaseja encerrado, basta pressionarem <F10>.

Para os iniciantes, recomendamos a utilização deste editor de textos. Mas como estamos num mundo livre, fica a livre-escolha! &;-D



## EXCLUSÃO

### RMDIR

Remover um diretório já existente.

```
$ rmdir [PARÂMETROS] [NOME_DO_DIRETÓRIO]
```

Exemplo:

```
$ rmdir /home/darkstar/teste/  
$ _
```

Porém este comando não pode remover nenhum diretório que não esteja vazio. Veja o exemplo abaixo:

```
$ rmdir /home/darkstar/teste/  
rmdir: `/home/darkstar/teste': Diretório não vazio  
$ _
```

Para estas circunstâncias, o comando *rm* possui o parâmetro *-r* (recursivo), que apaga todos os diretórios e seus respectivos arquivos. Veja à seguir.

### RM

Remove um arquivo já existente.

```
$ rmdir [PARÂMETROS] [NOME_DO_ARQUIVO]
```

Exemplo:

```
$ rm /home/darkstar/texto.txt
```

Podemos utilizar o parâmetro *-i*, o qual solicita a interação do usuário para a exclusão dos arquivos solicitados.

```
$ rm -i *.txt  
rm: remove arquivo comum `texto1.txt'? s  
rm: remove arquivo comum `texto2.txt'? n  
rm: remove arquivo comum `texto3.txt'? s  
$ _
```

Ao teclarmos <N> no ato da verificação, observem que o arquivo-texto *texto2.txt* foi mantido.

```
$ ls -l  
total 4  
-rw-r--r--    1 darkstar users          9 2003-12-26 19:47 texto2.txt  
$ _
```

Ao estudarmos o comando *rmdir*, vimos que ele não pode remover diretórios que possuem conteúdo. Para esta situação, podemos utilizar o comando *rm* junto com o parâmetro *-r* para resolver este problema.

```
$ rmdir teste/  
rmdir: `teste/': Diretório não vazio  
$ rm -r teste/  
$ _
```



# PERMISSÕES E ATRIBUTOS DE ARQUIVOS E DIRETÓRIOS

Além da manipulação básica dos arquivos em sistemas *GNU/Linux*, poderemos também definir as permissões e atributos específicos deste elementos de acordo com as nossas necessidades.

## PERMISSÕES DE ACESSO

Ao listarmos o conteúdo de um determinado diretório com o comando *ls -l*, poderemos observar a existência de várias colunas contendo informações referentes a: *tipo e permissão de acesso, número de atalhos, proprietário, grupo, tamanho e nome*, respectivamente.

```
lrwxrwxrwx 1 darkstar users 11 2005-02-18 22:03 cdrom -> /mnt/cdrom/
drwxr-xr-x 17 darkstar users 464 2005-03-10 23:57 docs
lrwxrwxrwx 1 darkstar users 11 2005-02-18 22:03 flash -> /mnt/flash/
lrwxrwxrwx 1 darkstar users 12 2005-02-18 22:03 floppy -> /mnt/floppy/
lrwxrwxrwx 1 darkstar users 9 2005-02-18 22:03 pkg -> /mnt/pkg/
drwxr-xr-x 4 darkstar users 208 2005-03-12 21:17 z
```

No modo texto, existem *10* seqüências de caracteres chamados *flag*.

```
lrwxrwxrwx 1 darkstar users
drwxr-xr-x 17 darkstar users
lrwxrwxrwx 1 darkstar users
lrwxrwxrwx 1 darkstar users
lrwxrwxrwx 1 darkstar users
drwxr-xr-x 4 darkstar users
```

O primeiro caracter indica o tipo do elemento:

<i><b>Tipo de elemento</b></i>	
<i>d</i>	Diretório.
<i>l</i>	Atalho ( <i>link</i> ).
<i>c</i>	Dispositivo tipo <i>character</i> .
<i>b</i>	Dispositivo tipo bloco.

Aos demais caracteres da coluna de permissões de acesso refere-se às *flags* de permissões definidas para os arquivos e/ou diretórios presentes. No modo gráfico, a *1a.* coluna é omitida, visto que os recursos visuais facilitam a exibição do tipo do item.



Name	Size	Modified	Permissions	Owner	Group
bin	2.5 KB	24.09.2005 17:42	rwxf-xr-x	root	root
boot	400 B	24.09.2005 10:35	rwxf-xr-x	root	root
dev	61.1 KB	25.09.2005 09:23	rwxf-xr-x	root	root
etc	4.6 KB	25.09.2005 09:23	rwxf-xr-x	root	root
home	96 B	14.07.2005 05:14	rwxf-xr-x	root	root
lib	3.3 KB	24.09.2005 17:43	rwxf-xr-x	root	root
mnt	216 B	24.09.2005 11:16	rwxf-xr-x	root	root
opt	96 B	24.09.2005 17:54	rwxf-xr-x	root	root
proc	0 B	25.09.2005 09:22	-xr-xr-x	root	root
root	528 B	24.09.2005 16:38	rw-x--	root	root
sbin	6.6 KB	25.07.2005 07:19	rwxf-xr-x	root	bin
share	288 B	24.09.2005 17:42	rwxf-xr-x	root	root
sys	48 B	12.05.2004 04:03	rwxf-xr-x	root	root
tmp	520 B	25.09.2005 10:22	rw-rwxrwt	root	root
usr	544 B	24.09.2005 22:11	rwxf-xr-x	root	root
var	456 B	15.09.2003 20:11	rwxf-xr-x	root	root

*Diretório raíz do sistema através do Konqueror, onde são exibidos suas respectivas permissões de acesso, definições de usuário e grupo.*

Estas 9 seqüências de letras subdividem-se nos seguintes grupos:

<b><i>Categoria &amp; Especificação</i></b>	
<i>Dono</i>	1a. a 3a. letras da seqüência, refere-se às permissões de acesso do dono do arquivo em questão – neste caso, <i>darkstar</i> .
<i>Grupo</i>	4a. a 6a. letras da seqüência, refere-se às permissões de acesso do grupo o qual o dono do arquivo pertence – neste caso, <i>users</i> .
<i>Todos</i>	7a. a 9a. letras da seqüência, refere-se às permissões de acesso dadas àqueles que não sejam donos e nem pertencem ao grupo de acesso do qual o arquivo pertence.

## CHMOD

Em modo texto, poderemos utilizar o comando *chmod* para alterar as permissões de acesso destes elementos de acordo com sua necessidade.

Sintaxes:

```
# chmod [ugoa] {+-} [rwx] [ARQUIVO_OU_DIRETÓRIO]
```

...ou...

```
# chmod [NNN] [ARQUIVO_OU_DIRETÓRIO]
```

Onde:

### *Categoria [ugoa]*

<b><i>Caracter</i></b>	<b><i>Definição de categoria (ajusta as flags...)</i></b>	
<i>u</i>	<i>users</i>	... apenas para usuário (dono) do arquivo.
<i>g</i>	<i>group</i>	... apenas para o grupo o qual o usuário se encontra.

<b>Caracter</b>		<b>Definição de categoria (ajusta as flags...)</b>
<i>o</i>	<i>other</i>	... para outros que não pertençam ao grupo do usuário.
<i>a</i>	<i>all</i>	... para todos.

### **Sinais de atribuição {+-}**

<b>Sinal</b>	<b>Significado / Função</b>
<b>+</b>	Habilita os parâmetros indicados.
<b>-</b>	Desabilita os parâmetros indicados.

### **flags [rwx]**

<i>flags</i>		<i>Arquivo</i>	<i>Diretório</i>
<i>r</i>	<i>read</i>	Leitura.	Acesso ao dados de seu conteúdo
<i>w</i>	<i>write</i>	Escrita.	Gravação de dados em seu conteúdo.
<i>x</i>	<i>execute</i>	Execução.	Visualização de dados conteúdo de seu conteúdo.

### **nnn – valor numérico**

<b>[NNN]</b>	<b>Categoria (Equivale de...)</b>
<i>1o. número</i>	... 2a. a 4a. seqüência de letras (dono).
<i>2o. número</i>	... de 5a. a 7a. seqüência de letras (grupo).
<i>3o. número</i>	... de 8a. a 10a. seqüência de letras (outros).

O valor de *[NNN]* à ser especificado na 2a. sintaxe varia de 0 à 7, e seu significado corresponde ao seguinte:

<b>N</b>	<b>Funcionalidade básica (permissão...)</b>
<i>0</i>	Nenhum.
<i>1</i>	... apenas para executar.
<i>2</i>	... apenas para gravar.
<i>4</i>	... apenas para ler.

Outro recurso interessante da atribuição dos valores numéricos é a sua combinação para a definição de múltiplas *flags*. Observe a tabela abaixo:

<b>N</b>	<b>Combinação</b>	<b>Funcionalidade combinada</b>
<i>3</i>	<i>1 + 2</i>	Permissão para executar ( <i>1</i> ) e gravar ( <i>2</i> ).
<i>5</i>	<i>1 + 4</i>	Permissão para executar ( <i>1</i> ) e ler ( <i>4</i> ).
<i>6</i>	<i>2 + 4</i>	Permissão para gravar ( <i>2</i> ) e ler ( <i>4</i> ).

<b><i>N</i></b>	<b><i>Combinação</i></b>	<b><i>Funcionalidade combinada</i></b>
7	1 + 2 + 4	Permissão para executar (1), gravar (2) e ler (4).

No geral fica assim:

<b><i>N</i></b>	<b><i>Funcionalidades totais</i></b>
0	sem permissão.
1	Permissão apenas para executar.
2	Permissão apenas para gravar.
3	Permissão para gravar e executar.
4	Permissão apenas para ler.
5	Permissão para executar e ler.
6	Permissão para gravar e ler.
7	Permissão para executar, gravar e ler.

### ***Exemplos práticos***

Realizaremos alguns simples exemplos da utilização de cada sintaxe para que possamos facilitar a compreensão de seu funcionamento.

#### **Exemplo 1:**

\$ chmod u-x [ARQ/DIR] Desabilita a execução do arquivo ou a visualização do conteúdo do diretório especificado apenas para o usuário.

\$ chmod g+r [ARQ/DIR] Habilita a leitura do arquivo ou o acesso ao diretório especificado apenas para o grupo do qual o usuário se situa.

\$ chmod o-w [ARQ/DIR] Desabilita a escrita do arquivo ou a gravação de arquivos no diretório especificado somente para outros que não seja o usuário, nem pertençam ao grupo deste.

\$ chmod a+x [ARQ/DIR] Desabilita a execução do arquivo ou a visualização do conteúdo do diretório especificado para todos os usuários.

#### **Exemplo 2:**

\$ chmod 754 [ARQ/DIR] Permissão para executar, gravar e ler pelo usuário (7); ler e executar pelo grupo o qual pertença o usuário (5); apenas leitura para outros que não pertençam ao grupo (4).

Parece um pouco complicado, mas com a prática aos poucos nós iremos nos

B

familiarizando. &;-D

## GRUPOS E CATEGORIAS

Além das *flags* de permissões de acesso, os sistemas *GNU/Linux* possuem também definições de usuários e grupos para cada arquivo e/ou diretório criado no sistema. Somente o superusuário é que tem permissões para realizar as alterações de usuários e grupos.

### CHOWN

O comando *chown* é utilizado para mudar dono e grupo de um arquivo ou diretório. Sua sintaxe básica é:

```
# chown [USUÁRIO].[GRUPO] [ARQUIVO_OU_DIRETÓRIO]
```

Onde *[USUÁRIO].[GRUPO]* são as especificações do novo usuário e grupo para o arquivo e/ou diretório desejado. Segue um simples exemplo:

```
# chown darkstar.users /home/darkstar/texto.txt
```

... ou...

```
# chown root.root /usr
```

Para definir os mesmos valores nos arquivos e diretórios de um determinado diretório, deveremos então utilizar o parâmetro *-R* (recursivo).

```
# chown -R root.root /usr
```

### CHGRP

Possui a mesma finalidade que o comando *chown*, porém apenas atua na modificação do grupo. Sintaxe básica:

```
# chgrp [GRUPO] [ARQUIVO_OU_DIRETÓRIO]
```

Exemplo:

```
# chgrp root /usr
```

Da mesma maneira, podemos definir o conteúdo do diretório recursivamente com o parâmetro *-R*.

```
# chgrp -R root /usr
```

### UMASK

Define as permissões padrões que os arquivos e diretórios deverão ter no momento em que serão criados.

```
# umask [NNN]
```

Apesar de atribuir as permissões de acesso utilizando a mesma metodologia do comando *chmod*, o comando *umask* utiliza valores diferenciados para os números os quais utiliza. Segue sua tabela de permissões de acesso, equivalente ao *chmod*, porém com valores invertidos, conforme vemos:



<i><b>N</b></i>	<i><b>Valor (permissão...)</b></i>
<i><b>6</b></i>	sem permissão.
<i><b>5</b></i>	Permissão apenas para executar.
<i><b>4</b></i>	Permissão apenas para gravar.
<i><b>3</b></i>	Permissão para gravar e executar.
<i><b>2</b></i>	Permissão apenas para ler.
<i><b>1</b></i>	Permissão para executar e ler.
<i><b>0</b></i>	Completo – leitura / execução / escrita.

Por padrão, o valor das permissões praticadas pelo *umask* é *022*, que corresponde ao *644* utilizado pelo *chmod*, ou seja, apesar de serem as mesmas definições de permissões de acesso, os valores numéricos são exatamente opostos.

Para alterar o valor *umask*, basta utilizar o comando...

```
$ umask [VALOR]
```

Onde [VALOR] deverá ser o novo perfil de valores das permissões à serem adotadas. Para torná-lo padrão à todos usuários, basta editar o arquivo */etc/profile* e alterar suas definições...

```
# Default umask. A umask of 022 prevents new files from being created group
# and world writable.
umask 022
```

... para...

```
umask [VALOR]
```

Uma questão importante está no uso da permissão para execução. Mesmo que na criação da grande maioria dos arquivos não necessitem, os diretórios precisam dela para que possamos acessá-los.

## CÓPIAS DE SEGURANÇA E COMPACTAÇÃO

A cópia de segurança é uma das operações de vital importância para a boa manutenção dos dados e informações contidas em uma unidade de armazenamento. Basicamente é dividida em 2 etapas: arquivamento e compactação.

### ARQUIVAMENTO

O ato de arquivar consiste basicamente em reunir um conjunto de arquivos, diretórios ou ainda, uma estrutura de arquivos e diretórios, em apenas um único arquivo. Para esta operação, temos um ótimo utilitário de linha de comando, chamado *TAR*.

B



## TAR

O *TAR* – *Type ARchive* –, conforme acima explicado, é um eficiente arquivador de dados. Sua sintaxe básica é:

```
$ tar [PARÂMETROS] [ARQUIVO_OU_DIRETÓRIO_OU_ESTRUTURA]
```

Pelo fato de possuir vastos recursos, existe uma imensidade de parâmetros relacionados. Descreveremos aqui apenas os mais utilizados.

<b><i>Gerais</i></b>	
<b><i>-v</i></b>	Modo verbose – exibe o andamento da operação no vídeo.
<b><i>-f</i></b>	Tratamento de arquivo, <i>device ARQ</i> .
<b><i>-w</i></b>	Solicita a confirmação para cada operação à realizar.
<b><i>-M</i></b>	Criação / listagem / extração de múltiplos volumes

Além dos parâmetros gerais, encontram-se também outros para o arquivamento propriamente dito. Segue tabela abaixo:

<b><i>Arquivamento</i></b>	
<b><i>-c</i></b>	Criação de um novo arquivo, o destino.
<b><i>-r</i></b>	Acrescenta arquivos à um pacote já criado.

Com o *TAR* podemos também visualizar o conteúdo de pacotes gerados, além de realizar alguns processos de comparação.

<b><i>Verificação</i></b>	
<b><i>-t</i></b>	Lista o conteúdo de um arquivo gerado.
<b><i>-d</i></b>	Compara o arquivo gerado com os arquivos atuais.

Ainda poderemos utilizar parâmetros específicos para compactar ou descompactar os pacotes.

<b><i>Compressão / Descompressão</i></b>	
<b><i>-j</i></b>	Compressão / descompressão com <i>Bzip2</i> .
<b><i>-z</i></b>	Compressão / descompressão com <i>Gzip</i> .

Por último, também existem alguns parâmetros extras para facilitar as atividades de extração dos pacotes criados, tendo destaque a descompactação e extração simultânea.

<b><i>Extração</i></b>	
<b><i>-x</i></b>	Extração de dados arquivados.
<b><i>-p</i></b>	Mantém as permissões originais dos dados arquivados.



*Exemplos básicos*

```
$ tar -cvf BACKUP.tar *
```

Serão empacotados todos os arquivos e subdiretórios do diretório corrente no arquivo *BACKUP.tar*.

```
$ tar -rf TEXTO.txt BACKUP.tar
```

Será anexo o arquivo *TEXTO.txt* ao pacote *BACKUP.tar*.

```
$ tar -xvf BACKUP.tar
```

Será desempacotado o pacote *BACKUP.tar.gz*.

```
$ tar -xvzf BACKUP.tar.gz
```

Será descompactado e desempacotado o pacote *BACKUP.tar.gz*.

*Observações*

Apesar de ser apenas um empacotador, o *TAR* também pode gerar pacotes compactados com o auxílio dos compactadores *gzip* e *bzip2*, com a utilização dos parâmetros *z* (zê) e *j* (jota), respectivamente. Já os utilitários de compactação descritos apenas geram um único arquivo por vez, sendo necessário a utilização dos empacotadores para criar um único arquivo compactado com uma estrutura de arquivos e diretórios, quando necessário. Por último, em algumas circunstâncias serão necessárias ferramentas como *cat* e *split* para o manuseio de arquivos e pacotes com grandes e/ou diversos volumes.

**COMPACTAÇÃO**

A compactação de arquivos nos sistemas *GNU/Linux* é feita tradicionalmente através dos seus próprios utilitários de linha de comando disponíveis na distribuição, não necessitando de quaisquer outro para as funcionalidades básicas. Os arquivos ou estrutura de arquivos e diretórios também podem ser manipulados por outros utilitários gráficos, conforme o interesse do usuário. Existem diversos compactadores para o sistema, mas os principais utilizados são *gzip*, *bzip2* e – em alguns casos – *zip*.

**GZIP / GUNZIP**

Atualmente é o compactador mais utilizado entre os disponíveis.  
Sintaxe:

```
$ gzip [PARÂMETROS] [ARQUIVO_ORIGEM]
```

Onde:

<i>gzip</i>	
<i>-c</i>	Mantém o arquivo original (não apaga).
<i>-d</i>	Descompacta o arquivo (o mesmo que utilizar apenas <i>gunzip</i> ).



<b><i>gzip</i></b>	
<b>-l</b>	Listagem de conteúdo do arquivo compactado.
<b>-v</b>	Exibe informações do processo.
<b>- (1 até 9)</b>	Variação da taxa de compressão, onde 1 = compressão rápida (baixa) e 9 = compressão lenta (alta).

Para simplesmente compactar um arquivo...

```
$ gzip [ARQUIVO]
```

Para compactar um arquivo com alta taxa de compressão...

```
$ gzip -9 [ARQUIVO]
```

Para descompactar um arquivo comprimido.

```
$ gzip -d [ARQUIVO]
```

... ou...

```
$ gunzip [ARQUIVO]
```

## **BZIP2 / BUNZIP2**

O 2o. compactador mais utilizado pelos *linuxistas*. Além de sua vasta utilização, possui uma boa vantagem em comparação *gzip*: consegue obter taxas maiores de compressão, chegando à reduzir o arquivo em até 20% à mais. Em contrapartida, exige uma maior demanda de processamento, onde a sua utilização em computadores de pouco desempenho ou em arquivos de grandes volumes faz sentir uma grande perda de tempo.

Sintaxe:

```
$ bzip2 [PARÂMETROS] [ARQUIVO]
```

Onde:

<b><i>bzip2</i></b>	
<b>-d</b>	Descompacta o arquivo (o mesmo que utilizar apenas <i>bunzip2</i> ).
<b>-f</b>	Força o modo sobre-escrita (cria um pacote e grava em cima de um já existente).
<b>-s</b>	Reduz o consumo de memória durante a compactação (ideal para máquinas com pouca memória).
<b>-v</b>	Modo <i>verbose</i> – exibe o andamento da operação no vídeo.

Para realizar a compactação de um arquivo, basta...

```
$ bzip2 [ARQUIVO]
```

Para descompactar...

```
$ bunzip2 [ARQUIVO].bz2
```



## ZIP / UNZIP

Os comandos *zip* e *unzip* são respectivamente compactador e descompactador de volumes *.zip* criados pelos utilitários *\*Zip* dos sistemas *MS/DOS* e *Windows*.

Sintaxe:

```
$ zip [PARÂMETROS] [DESTINO] [ORIGEM]
```

Onde:

<i>zip / unzip</i>	
<i>-e</i>	Modo seguro – permite a utilização de senha de proteção, esta solicitada no ato da compactação / descompactação.
<i>-r</i>	<i>Recursive</i> – Compacta arquivos e diretórios.
<i>- (1 até 9)</i>	Variação da taxa de compressão, onde 1 = compressão rápida (baixa) e 9 = compressão lenta (alta).

Para realizar uma simples compactação, basta utilizar.

```
$ zip BACKUP.tar NOVO-BACKUP.zip
```

... ou...

```
$ zip -r SOURCE PROGRAMA.zip
```

Para descompactar um arquivo para um diretório específico...

```
$ unzip -d /tmp BACKUP.zip
```

## UTILITÁRIOS

### SPLIT

Divide um arquivo em várias partes. Sintaxe:

```
$ split [PARÂMETROS] [TAMANHO][UNIDADE_DE_MEDIDA] [ARQUIVO]
```

Onde:

<i>split</i>	
<i>-b</i>	Define a unidade de medida: <i>byte</i> ( <i>b</i> ), <i>KB</i> ( <i>kb</i> ) e <i>MB</i> ( <i>mb</i> ).

Exemplo:

```
$ split -b 1440kb BACKUP.tar.gz
```

Serão gerados diversos arquivos com o tamanho especificado com a nomenclatura *xaa*, *xab*, *xac*, etc., bastando apenas gravá-los em disquetes para o seu transporte. Para reuni-los novamente em um só (concatená-los), utilize o comando *cat*.

B

## CAT

O comando *cat* possui diversas opções e funcionalidades, como poderemos ver ao checar sua documentação. Mas para operações básicas, ele é bastante utilizado em operações de visualização e concatenação.

Sintaxe:

```
$ cat [OPTION] [ARQUIVO]
```

### Visualização de conteúdo

Para realizar uma visualização do arquivo *texto.txt*, basta utilizar...

```
$ cat texto.txt  
-//-
```

... onde o conteúdo do arquivo mencionado será impresso no vídeo.

### Concatenação

Além das funções de visualização propriamente dita, o comando *cat* também realiza a concatenação (união) de arquivos em um só. Basta utilizar a sintaxe...

```
$ cat [ARQUIVO_TEXTO_1] [ARQUIVO_TEXTO_2] > [ARQUIVO_TEXTO_FINAL]
```

... para que sejam unidos os arquivos desejados em apenas um só. Um detalhe importante que deverá ser observado está na ordem dos arquivos a serem concatenados.

Segue um exemplo simples e prático:

```
$ cat texto.txt mais_info.txt > texto1.txt
```

Para mesclar arquivos gerados pelo *split* (veja seção anterior)...

```
$ cat x* > [ARQUIVO_ORIGINAL]
```

Uma recomendação dada à utilização deste comando refere-se ao arquivo destino. Não devemos utilizar o mesmo arquivo para receber o conteúdo dos arquivos mesclados, podendo ter o risco de ocorrências imprevisíveis, onde a perda dos dados é certamente indesejada.

## CONCLUSÃO

Existe uma infinidade de possíveis operações onde é necessária a manipulação direta de arquivos e diretórios. As operações descritas neste capítulo são apenas as mais comuns, necessárias para a grande maioria dos usuários em *desktops*. Caso queiram se aprofundar, consultem as páginas de manual disponíveis na distribuição.

Novamente reforçando: outra recomendação bastante interessante para esta atividade está no uso do gerenciador de arquivos *Konqueror*, que provê excelentes recursos para estas atividades. &;-D

B

## V. UNIDADES, PARTIÇÕES E FORMATOS

---

### INTRODUÇÃO

Como nós já sabemos, todos os sistemas operacionais alocam suas informações em sistemas de armazenamento de dados de vários tipos (unidades), que podem ser subdivididos em várias partes (partições) e utilizarem um método de escrita específico (sistema de arquivos).

Neste capítulo iremos conhecer as particularidades das distribuições *GNU/Linux* quanto a este aspectos, além de obter instruções para a manipulação através das ferramentas disponíveis na linha de comando.

### OS SISTEMAS DE ARQUIVOS

Sistemas de arquivos são métodos de representação para a organização dos arquivos (dados) em um determinado meio de armazenamento. Ao realizarmos a formatação de uma unidade qualquer (seja disco rígido, disquete, *zips*, etc.), estaremos condicionando a sua estrutura para que esteja pronto para receber dados.

### TIPOS DE PARTIÇÕES E SISTEMAS DE ARQUIVOS

Os sistemas *GNU/Linux* suportam uma infinidade de tipos de sistemas de arquivos, onde as mais importantes são:

#### SWAP

A partição *swap* é um espaço do disco rígido reservado para o uso do sistema operacional como “*complemento*” da memória *RAM*. Quando a memória principal do sistema operacional está completamente “*cheia*” e existe a necessidade de executar alguma tarefa que exija mais consumo de memória, as informações que estão contida na memória principal são gravadas nesta partição em separado enquanto o sistema carrega para a memória principal as informações requeridas por esta tarefa. Assim que é encerrada, a memória principal é “*esvaziada*” e novamente recarregada com as informações contidas na partição *swap*.

#### EXT2

O sistema de arquivos *ext2* é o padrão dos sistemas *GNU/Linux*. Possui muitas similaridades em comparação aos sistema *VFAT* utilizado no *Windows*, como o suporte a nomenclatura de arquivos com 256 caracteres e tamanho de *clusters* fixo em 4 KB, além da limitar o tamanho de arquivos para 2 GB. Porém um dos principais diferenciais é a possibilidade de atribuição de permissões especiais aos arquivos criados neste sistema. Somente iremos operar nestes arquivos de acordo com os atributos



definidos pelo usuário que o criou (dono) ou o administrador. Estes atributos são a leitura, a escrita e a gravação.

### EXT3

Na verdade, o *ext3* não difere muito do sistema de arquivos *ext2*, porém apresenta o recurso *journaling*, que caracteriza-se por manter arquivadas no disco rígido uma imagem do estado do sistema de arquivos atualizada constantemente – um arquivo na raiz da partição chamado *journal*. Com isto, toda vez que houver algum erro no sistema que necessite da reinicialização, o estado do sistema de arquivos é automaticamente restaurado baseando-se apenas nas informações registradas neste arquivo. Na prática, isto substitui a necessidade da utilização do *fsck*, em que ao realizar a checagem da integridade do sistema de arquivos, consome muito tempo e indisponibiliza o sistema por longos intervalos.

Atualmente o sistema de arquivo *ext3* está sendo gradativamente “substituído” pelo *ReiserFS* em virtude de suas qualidades, em especial o conceito “Alta disponibilidade”, extremamente importante em servidores e sistemas que necessitam ter o menor tempo de parada (fora do ar) possível.

### REISERFS

✓ <<http://www.namesys.com/>>.

Criado pela empresa *Namesys*, o sistema de arquivos *ReiserFS* foi desenvolvido visando adequar-se ao conceito “Alta disponibilidade”. Por exemplo, quando ocorre uma queda no sistema resultante de falhas em arquivos ou falta de energia, ao reiniciarmos as partições *ext2* e *ext3*, teremos uma verificação de sua integridade feita automaticamente pelo programa *e2fsck*. Mas dependendo do tamanho e da quantidade de partições, este processo requer um tempo considerável – vários minutos, algo bastante indesejado em circunstâncias em que existe a necessidade de alta disponibilidade das máquinas. No sistema de arquivos *ReiserFS*, ao invés de realizar a checagem total, ele apenas consulta o arquivo *journal*, onde o mesmo apenas informa as definições prévias para a restauração.

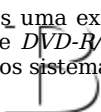
Dentre outras características importante do *ReiserFS* está nos seus pequenos *clusters*, que possuem tamanho máximo de *512 bytes*, ideal para a utilização em armazenamento de inúmeros arquivos de pequeno volume, o que acarreta menor perda de espaço. Além disso, não há a limitação de *2 GB* para arquivos que ultrapassem este tamanho.<sup>5</sup>

### ISO9660

O sistema de arquivos *ISO9660* é somente utilizado para os *CD-ROMs*,

---

5 Isto torna este sistema de arquivos uma excelente opção para trabalharmos com geração de imagens e gravação de *DVD-R/W*, o que não quer dizer que não seja possível realizar estas atividades nos sistemas *ext2* e *ext3*.



devido à natureza de seu armazenamento de dados permanente, impossibilitando o sistema operacional de definir um sistema de arquivos.

A imagem *ISO* nada mais é um arquivo especial que contém as informações sobre todos os arquivos que serão gravados em uma mídia de *CD-R/RW*, utilizando-se o formato *ISO9660*.

## MSDOS E VFAT

Os tradicionais formatos de sistema de arquivos do *MS-DOS* e *Windows*. Apesar de não serem utilizados pelos sistemas *GNU/Linux* e em virtude de sua popularização, estes sistemas de arquivos são plenamente bem suportados por praticamente todas as distribuições existentes.

Na utilização de disquetes como transporte de dados, é recomendável a utilização destas formatações para que os mesmos possam ser lidos em outros computadores providos do sistema operacional *Windows*. Em especial, o sistema *vfat*, que possibilita visualizar os arquivos com nomes longos (mais de 8.3 caracteres), o qual é recomendada a sua utilização no acesso à estes dispositivos.

## DIFERENÇAS ENTRE OS SISTEMAS DE ARQUIVOS REISERFS E EXT3

Já que este sistema se comporta da mesma maneira que o *ext3*, por que então o *ReiserFS* está “*substituindo*” o sistema de arquivos *ext3*?

A principal diferença está exatamente no recurso *journaling* do sistema de arquivos *ReiserFS* funciona de forma diferente do sistema *ext3*: O *ReiserFS* apenas armazena informações sobre o espaço dos arquivos e permissões, ao passo que o *Ext3* além de executar estas funções ainda salvaguarda o próprio conteúdo dos arquivos afetados durante uma queda do sistema.

A grande vantagem do sistema *ReiserFS* está na facilidade de recuperar a consistência do sistema de arquivos em um tempo mínimo (décimos de segundos). Praticamente torna-se inexistente a possibilidade de uma pane em alguma pasta ou até mesmo nas partições do disco rígido. Em contrapartida, caso o sistema esteja sofrendo gravações de dados no exato momento da queda, estas arquivos infelizmente não poderão ser recuperados, pois seu conteúdo estará truncado ou incompleto.

Já o sistema *Ext3* apresenta características quase que “*opostas*” ao *ReiserFS*: pelo fato de realizar a manutenção dos dados gravados no exato momento da queda do sistema, as possibilidades de recuperação destes arquivos são infinitamente maiores. Em contrapartida, é praticamente certo a degradação de desempenho face às constantes gravações em seu *log* de sistema. Além disso, existe a possibilidade do próprio *journal* se corromper durante a queda do sistema, que com isto acarreta na necessidade da utilização do demorado processo de recuperação com o *fsck*.





## AS UNIDADES E AS PARTIÇÕES

As unidades, são como o próprio nome diz, dispositivos físicos especiais utilizados para o armazenamento de dados. Já as partições são divisões criadas no disco rígido que são utilizadas para diversas finalidades. Entre elas, a mais importante é a organização e otimização dos dados nela arquivados e conseqüentemente do sistema como um todo. Um disco rígido poderá ter diferentes tipos de partições e com isto poderemos até mesmo alocar diferentes sistemas operacionais em uma única unidade, conforme veremos mais adiante.

Existem diversos tipos de unidades, das quais as principais são os disquetes, os discos rígidos e a unidade de *CD-ROM*. Além destas, existem outras menos comuns, como o *Zip-drive*, a memória eletrônica, o gravador de *CD-R/RW*, entre muitos outros. Todas elas são suportadas e acessíveis pelos sistemas *GNU/Linux*, mas isto no momento não é o mais importante. O que deveremos realmente dar importância é o sistema de arquivos.

## OPERAÇÕES E ATIVIDADES AFINS

### GERAL

#### TRABALHANDO COM PARTIÇÕES E UNIDADES

Existem uma infinidade de operações que poderão ser realizadas, mas antes de mais nada, será necessário obter acesso à estas para realizar as operações mais cotidianas, não é mesmo?

#### MOUNT / UMount

Os comandos *mount* e *umount* são utilizados respectivamente para “*montar*” (ter acesso) e “*desmontar*” (retirar acesso) partições e unidades. Sua sintaxe básica é a seguinte:

```
$ mount [PARÂMETROS] [DEVICE] [PONTO_DE_MONTAGEM]
```

Onde:

<b><i>mount / umount</i></b>	
<i>Parâmetros</i>	Opções diversas de utilização.
<i>Device</i>	Dispositivos que se deseja montar.
<i>Ponto de montagem</i>	Diretório de acesso destes dispositivos.

Dentre os principais parâmetros existentes, destaca-se:

<b><i>mount / umount</i></b>	
<i>-a</i>	<i>Auto.</i> Montagem automática.

B

<b><i>mount / umount</i></b>	
<b><i>-F</i></b>	<b><i>Force.</i></b> Montagem de modo forçado.
<b><i>-r</i></b>	<b><i>Read only.</i></b> Permissão somente para leitura.
<b><i>-t</i></b>	<b><i>Type.</i></b> Pré-define o sistema de arquivos o qual a partição se encontra para ser montado. Dentre os tipos suportados, segue: <i>ext2</i> , <i>ext3</i> , <i>iso9660</i> , <i>msdos</i> / <i>umsdos</i> , <i>reiserfs</i> e <i>vfat</i> .
<b><i>-v</i></b>	<b><i>Verbose.</i></b> Exibe informações adicionais do processo.
<b><i>-w</i></b>	<b><i>Write.</i></b> Permissão para escrita.

Segue alguns exemplos para melhor ilustrar:

```
$ mount /dev/hda5 /mnt/hd
$ mount -t vfat /dev/hda1 /mnt/win
```

Para obter maiores informações, consulte a ajuda ou o manual eletrônico do comando mencionado.

### ***Disquetes***

Os disquetes são acessados tradicionalmente utilizando seus respectivos *devices*, conforme tabela abaixo:

<b><i>DOS / GNU/Linux</i></b>	
<b><i>A:</i></b>	<b><i>/dev/fd0</i></b>
<b><i>B:</i></b>	<b><i>/dev/fd1</i></b>

Para acessá-los, bastará utilizar a seguinte sintaxe:

```
$ mount /dev/fd0 /mnt/floppy
```

Onde */mnt/floppy* poderá ser omitida caso a unidade esteja especificado em */etc/fstab*.

Tanto para os disquetes como quaisquer outras unidades que utilizam o sistema *VFAT*, deveremos incrementar este comando o seguinte parâmetro:

```
$ mount -t vfat /dev/fd0 /mnt/floppy
```

Será desnecessária a utilização do parâmetro *-t vfat* se ele estiver previamente especificado em */etc/fstab*.

### ***CD-ROM/RW***

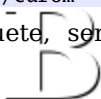
Estas unidades são acessados com a utilização do comando...

```
$ mount /dev/cdrom /mnt/cdrom
```

Onde */mnt/cdrom* poderá ser omitida caso a unidade esteja especificado em */etc/fstab*. Caso contrário...

```
$ mount -t iso9660 /dev/cdrom /mnt/cdrom
```

Da mesma forma que no disquete, será desnecessária a utilização do



parâmetro *-t iso9660* se ele estiver previamente especificado em */etc/fstab*.

Lembre-se de que */dev/cdrom* é apenas um atalho apontado para o verdadeiro dispositivo do *CD-ROM/RW*, onde provavelmente serão */dev/hdb* ou */dev/hdc*, de acordo com as configurações da máquina em uso.

### Memória eletrônica

Para montar uma unidade de memória eletrônica, deveremos recorrer ao *device* responsável pela emulação *SCSI*.

```
$ mount /dev/sda1 /mnt/[PONTO_DE_MONTAGEM]
```

Onde [PONTO\_DE\_MONTAGEM] será o diretório a ser criado para aquele dispositivo (p. ex.: */mnt/flash*).

### Zip-drive

O mesmo ocorre com unidades de *zip-drive*, porém o acesso à esta será realizado somente com a utilização do *device sda4*, ao invés do *sda1*.

```
$ mount /dev/sda4 /mnt/[DIRETÓRIO_CRIADO_PARA_O_DISPOSITIVO]
```

### Desmontagem

Para desmontar quaisquer dessas unidades, deveremos apenas utilizar...

```
$ umount /dev/[UNIDADE]
```

... ou...

```
$ umount [PONTO_DE_MONTAGEM]
```

Um detalhe importante é que o comando *umount* checa se os dados à serem gravados nas unidades a ser desmontadas foram realizados, para depois efetivar a desmontagem propriamente dita dos dispositivos.

### SYNC

Realiza toda a transferência de dados da *cache* do sistema (arquivos e diretórios) para a unidade montada (disquete, memória eletrônica, *CD-ROM*, etc.), para que possamos desmontar a unidade imediatamente.

```
$ sync
```

É muito útil em circunstâncias em que não sabemos porque o dispositivo não desmonta, mesmo que todas as operações estejam concluídas.

## DEFINIÇÃO DE SISTEMA DE ARQUIVOS

Os respectivos utilitários que utilizaremos em linha de comando para estas atividades são *mkfs*, *mkreiserfs* e *mkswap*.

### MKFS

O *mkfs* é o utilitário usado para criar um sistema de arquivos.



Sintaxe:

```
# mkfs.ext2 [PARÂMETROS] /dev/[PARTIÇÃO]
```

Onde:

<b><i>mkfs</i></b>	
<b><i>-b</i></b>	Definição do tamanho do bloco ( <i>cluster</i> ).
<b><i>-c</i></b>	Checagem de blocos danificados.
<b><i>-L [NOME]</i></b>	<i>Label</i> – Define um nome para o sistema de arquivos.

O *mkfs* possui “*extensões*”, das quais cada uma possui a finalidade de criar um sistema de arquivos específicos:

<b><i>mkfs</i></b>	
<i>mkfs.ext2</i>	<i>Extend 2.</i>
<i>mkfs.ext3</i>	<i>Extend 3.</i>
<i>mkfs.msdos</i>	<i>MS -DOS.</i>

Estas opções poderão ser omitidas, desde que utilizemos o parâmetro *-t*, acompanhado do sistema que se queira criar (*msdos*, *ext2*, *ext3*, etc.).

```
# mkfs -t [SIST._ARQUIVOS] [PARÂMETROS] /dev/[PARTIÇÃO]
```

Observem também que o *mkfs* não suporta o sistema de arquivos *ReiserFS*, sendo necessário então utilizar outra ferramenta, o *mkreiserfs*.

## MKREISERFS

Conforme dito na seção anterior, o utilitário *mkreiserfs* é utilizado para definir um sistema de arquivos *ReiserFS* em uma partição.

Sintaxe:

```
# mkreiserfs [PARÂMETROS] /dev/[PARTIÇÃO]
```

Onde:

<b><i>mkreiserfs</i></b>	
<b><i>-b</i></b>	Definição do tamanho do bloco ( <i>cluster</i> ).
<b><i>-L [NOME]</i></b>	<i>Label</i> – Define um nome para o sistema de arquivos.
<b><i>-s [VALOR]</i></b>	Define o tamanho do arquivo de <i>journal</i> em blocos.

## MKSWAP / SWAPON

Formata uma partição *swap* para ser utilizada no sistema.

Sua utilização é simples, bastando digitar na linha de comando...

```
# mkswap /dev/[PARTIÇÃO]
```

Para ativá-la ao sistema, deveremos utilizar...

```
# swapon /dev/[PARTIÇÃO]
```

## VERIFICANDO AS PARTIÇÕES E UNIDADES DO SISTEMA

Existem diversas ferramentas para a checagem das partições e unidades do sistema, dentre as quais, as principais usadas são:

DF

Verifica o espaço ocupado das unidades montadas.

Sintaxe:

```
$ df [PARÂMETROS]
```

Onde:

<i>df</i>	
<i>-h</i>	Exibe as dimensões em <i>MB (hm)</i> e <i>GB (h)</i> .
<i>-T</i>	Exibe o sistema de arquivos utilizado.

Ao utilizar somente o *df*, teremos apenas esta simples avaliação:

```
# df
Filesystem      1k-blocks      Used Available Use% Mounted on
/dev/hda5        2104408        339684   1764724   17% /
/dev/hda6        7341440       1841004   5500436   26% /usr
/dev/hda7        2104408        51984   2052424    3% /var
/dev/hda8        1052184        39240   1012944    4% /tmp
/dev/hda9        1052184       177076   875108   17% /home
/dev/hda10       24879804       282344  24597460    2% /usr/pkg
# _
```

Definindo o parâmetro *-T*, veremos o sistema de arquivos utilizado...

```
# df -T
Filesystem      Type      1k-blocks      Used Available Use% Mounted on
/dev/hda5 reiserfs   2104408        339808   1764600   17% /
/dev/hda6 reiserfs   7341440       1841004   5500436   26% /usr
/dev/hda7 reiserfs   2104408        51984   2052424    3% /var
/dev/hda8 reiserfs   1052184        39240   1012944    4% /tmp
/dev/hda9 reiserfs   1052184       177076   875108   17% /home
/dev/hda10 reiserfs  24879804       282344  24597460    2% /usr/pkg
# _
```

Com a utilização do parâmetro *-h*, teremos as informações desejadas em *GB*.

```
# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/hda5       2.0G  332M  1.6G   17% /
/dev/hda6       7.0G  1.8G  5.2G   26% /usr
/dev/hda7       2.0G   51M  1.9G    3% /var
/dev/hda8       1.0G   39M  989M    4% /tmp
/dev/hda9       1.0G  173M  854M   17% /home
```



```
/dev/hda10          23G  276M   22G   2% /usr/pkg
# _
```

Os parâmetros também podem ser utilizados combinados (p. ex. *-Th*).

## BADBLOCKS

Utilizado para checar blocos defeituosos na unidade de disco rígido.

Sintaxe:

```
# badblocks [PARÂMETRO] /dev/[PARTIÇÃO]
```

Onde:

<b><i>badblocks</i></b>	
<b><i>-b</i></b>	Define o tamanho do bloco à ser checado.
<b><i>-v</i></b>	Modo <i>verbose</i> – mostra as operações em andamento.

Para realizarem uma checagem básica, utilizem...

```
# badblocks -b 4096 /dev/[PARTIÇÃO]
```

Normalmente os tamanhos de blocos utilizados pelos atuais sistemas de arquivos são de *4.096 bytes*.

## FSCK

Realiza uma checagem da unidade em questão, procurando por áreas e blocos danificados que porventura possam existir na unidade de disco rígido.

Sintaxe:

```
# fsck.[FS] [PARÂMETRO] /dev/[PARTIÇÃO]
```

Onde:

<b><i>fsck</i></b>	
<b><i>[FS]</i></b>	<i>Filesystem</i> – sistema de arquivos propriamente dito ( <i>ext2</i> , <i>ext3</i> , <i>xiafs</i> , <i>minix</i> , etc.).
<b><i>-a</i></b>	Modo automático – realiza as operações sem realizar qualquer pergunta (inverso de <i>-r</i> – PERIGOSO!). Para sistemas de arquivos <i>ext2</i> , deverá ser utilizado o parâmetro <i>-p</i> para obter a mesma funcionalidade, visto que <i>-a</i> somente encontra-se por questões de compatibilidade.
<b><i>-c</i></b>	Verifica os blocos defeituosos e atualiza a tabela da unidade, marcando-os como inválidos.
<b><i>-f</i></b>	<i>Force</i> – realiza a checagem em modo forçado.
<b><i>-r</i></b>	Modo interativo – realiza algumas perguntas antes de efetuar as

<b><i>fsck</i></b>	
	operações (inverso de <i>-a</i> ).
<i>-t</i>	Define o sistema de arquivos utilizado.
<i>-v</i>	Modo <i>verbose</i> – visualizar as operações em execução.
<i>-y</i>	Confirmação para aceitar todas as perguntas realizadas no modo interativo.

Para uma simples e básica checagem em uma partição *ext3*:

```
# fsck.ext3 /dev/[PARTIÇÃO]
```

Para uma partição *ext2* também poderemos utilizar...

```
# e2fsck /dev/[PARTIÇÃO]
```

Onde *e2fsck* é um apelido para *fsck.ext2*.

Em um disquete formatado para *DOS* basta utilizar...

```
# fsck -t msdos /dev/fd0
```

É importante observar que as unidades avaliadas deverão estar desmontadas. Além disso, o *fsck* somente funciona em sistemas de arquivos *ext2* e *ext3*, não sendo útil em partições *ReiserFS* e *VFAT*.

## REISERFS

Também realiza uma checagem da unidade em questão, procurando por áreas e blocos danificados que porventura possam existir na unidade de disco rígido; sendo que esta ferramenta foi desenvolvida para ser utilizada unicamente em sistemas de arquivos *ReiserFS*.

Sintaxe:

```
# reiserfs [PARÂMETRO] /dev/[PARTIÇÃO]
```

Exemplo:

<b><i>reiserfs</i></b>	
<i>-a</i>	Exibe detalhes sobre o sistema de arquivos.
<i>-j</i>	Especifica um arquivo <i>journaling</i> alternativo para ser utilizado.
<i>-q</i>	Não exibe nenhuma mensagem de <i>status</i> .

Para uma simples e básica checagem, utilizem...

```
# reiserfs /dev/[PARTIÇÃO]
```

Lembrem-se: as partições devem estar previamente desmontadas.



## TRANSFERINDO DADOS

DD

Além da cópia direta de arquivos e dados, o comando *dd* também poderá ser utilizado para criar cópias de dados das partições desejadas.

Para obterem maiores informações, consultem nesta parte o capítulo *Manipulação de arquivos de diretórios*.

## DISQUETES

Em virtude da imensa utilização desta unidades de armazenamento, resolvemos descrever um conjunto de instruções para facilitar ao máximo sua utilização em sistemas *GNU/Linux*.

## UTILIZAÇÃO

No *MS-DOS*, o processo de utilização de disquetes é algo relativamente simples, onde basta apenas inserir a mídia no drive leitor e acessar diretamente o dispositivo. Em sistemas *GNU/Linux*, este dispositivo deverá ser antes montado como qualquer outro, porém lembre-se que as unidades são referenciadas de forma diferente.

Para obterem acesso a estes dispositivos, consultem neste capítulo a seção *Unidades e as partições -> Trabalhando com partições e unidades*.

## FORMATAÇÃO

Diferente dos sistemas da *Microsoft*, onde seus sistemas operacionais realizam uma única operação – formatação – para preparar os disquetes para o uso, nos sistemas *GNU/Linux*, por padrão estes processos se dividem em 2 fases distintas:

- Formatação de baixo nível;
- Criação do sistema de arquivos.

Para trabalhar com disquetes, teremos em mãos alguns utilitários específicos para esta finalidade, os quais iremos conhecer agora.

### FORMATAÇÃO DE BAIXO NÍVEL

Na formatação de baixo nível serão recriados todos o setores e trilhas para posteriormente ser definido um sistema de arquivo. Este processo é de suma importância, pois só assim teremos certeza de que armazenaremos nossos dados em uma unidade isenta de defeitos, pois a mídia de armazenamento magnética é muito sensível.

```
# fdformat /dev/fd0H1440
```

Observe que para realizar a formatação de um disquete, o mesmo não

B



poderá estar montado previamente no sistema.

## SISTEMA DE ARQUIVOS

Após a baixa formatação, teremos então a possibilidade de definir o sistema de arquivos que desejarmos, diferente dos utilitários do *MS-DOS* que nos permite apenas a utilização de um único sistema de arquivos.

Dentre os formatos suportados, destacam-se:

<b><i>Sistema de arquivos</i></b>	
<i>msdos</i>	Utilizado pelo <i>MS-DOS/Windows</i> , da <i>Microsoft</i> .
<i>ext2</i>	Utilizado pelos sistemas <i>GNU/Linux</i> .

Para criar um sistema de arquivos com as opções desejadas, basta utilizar o comando *mkfs* e definir o formato desejado:

```
$ mkfs.msdos /dev/fd0
```

... ou...

```
$ mkfs -t ext2 /dev/fd0
```

## SUPERFORMAT

Outro bom utilitário para esta atividade é o *Superformat*, que por sua vez já realiza a formatação e define o sistema de arquivos da unidade em questão.

Sintaxe:

```
$ superformat [PARÂMETROS] /dev/[UNIDADE]
```

Segue um exemplo simples e básico:

```
$ superformat /dev/fd0
Measuring drive 0's raw capacity
warmup cycle: 7 200150 200148
```

Não irá demorar para que todo o processo seja concluído.

```
Measuring drive 0's raw capacity
In order to avoid this time consuming measurement in the future,
add the following line to /etc/driveprm:
drive0: deviation=720
CAUTION: The line is drive and controller specific, so it should be
removed before installing a new drive 0 or floppy controller.
```

```
Verifying cylinder 79, head 1
mformat -s18 -t80 -h2 -S2 -M512 a:
```

```
$ _
```

Este comando utiliza apenas seus parâmetros padrões.

Para obterem maiores informações, consultem o seu manual eletrônico.



## DISCOS DE INICIALIZAÇÃO

Para criarmos discos de inicialização no *Slackware*, basta apenas utilizarmos o comando *mkbootdisk*.

Sintaxe:

```
# mkbootdisk --device /dev/[DISQUETE] [KERNEL]
```

Onde:

<i>mkbootdisk</i>	
<i>--device</i>	Define o <i>device</i> da unidade (p. ex.: <i>/dev/fd0</i> , <i>/dev/fd1</i> , etc.).
<i>[KERNEL]</i>	Define a versão do <i>kernel</i> atual do sistema (p. ex.: <i>2.4.18</i> ).

## SOBRE O PACOTE MTOOLS

Face à dificuldade de adaptação de diversos usuários que antes utilizavam a linha de comando do *MS-DOS*, foi desenvolvido um pacote chamado *mtools*, que tem como objetivo simular os comandos mais usados pelo *MS-DOS* no *GNU/Linux*.

Os nomes dos comandos do pacote *mtools* e suas sintaxes são idênticos aos comandos tradicionais do *MS-DOS*, onde, como diferencial, os comandos pertencentes ao pacote *mtools* possuem como prefixo o caracter <M> antes para ser executado. Por exemplo, o *dir* do *MS-DOS* equivale ao comando *mdir*; *mkdir* equivale ao *copy*, e assim por diante.

Para aqueles que conhecem o *MS-DOS*, possivelmente estarão sentindo uma certa dificuldade em utilizar a linha de comando, pois provavelmente deverão conhecer seus parâmetros e sintaxes (eis o motivo de não estarem descritos aqui); caso não o conheçam, não há necessidade de estudar estes comandos, pois diversos pacotes e utilitários disponíveis na linha de comando do *GNU/Linux* satisfazem plenamente nossas necessidades.

<i>MTools</i>	
<i>mattrib</i>	Ajusta e modifica os atributos de arquivos, como leitura, gravação e ocultação.
<i>mcd</i>	Navega entre os diretórios do sistema.
<i>mcopy</i>	Copiar arquivos e diretórios.
<i>mdel</i>	Remover arquivos.
<i>mdeltree</i>	Remover diretórios e respectivos subdiretórios.
<i>mdir</i>	Listar o conteúdo do diretório.
<i>mformat</i>	Formatar o disquete.
<i>minfo</i>	Imprime os parâmetros do sistema de arquivos <i>MS-DOS</i> .



<b><i>MTools</i></b>	
<i>mlabel</i>	Renomeia o volume do disquete.
<i>mmd</i>	Cria diretórios.
<i>mmount</i>	Monta a unidade de disquete (análogo ao <i>mount /dev/fd0</i> ).
<i>mmove</i>	Movimenta arquivos e diretórios.
<i>mpartition</i>	Particiona uma unidade para ser usado no <i>MS-DOS</i> .
<i>mrd</i>	Remove diretórios.
<i>mren</i>	Renomeia arquivos.
<i>mtype</i>	Exibe o conteúdo de um arquivo.
<i>mshowfat</i>	Exibe quantos blocos foram alocados por um arquivo.
<i>mbadbblocks</i>	Verifica a existência de blocos defeituosos.
<i>mzip</i>	Compacta no formato <i>zip</i> os arquivos e/ou diretórios desejados.
<i>mkmanifest</i>	Cria uma lista de arquivos com seus nomes equivalentes no formato <i>DOS 8.3</i> .
<i>mcheck</i>	Realiza a checagem de integridade (similar <i>chkdsk</i> ).

Para obterem maiores informações, digitem na linha de comando:

```
$ man [COMANDO_PERTENCENTE_AO_PACOTE_MTOOLS]
```

## CD-R/CD-RW

Os seguintes utilitários são essenciais para a manipulação da unidade de *CD-R/CD-RW* do sistema:

### mkisofs

O *mkisofs* – abreviação de *mk* (*make* = fazer) *iso* (imagem *ISO*) e *fs* (*filesystem* = sistema de arquivos) – permite construir imagens *ISO* à partir de dados disponíveis de uma unidade.

Sintaxe:

```
$ mkisofs [PARÂMETROS] -o [IMAGEM].iso [DIRETÓRIO_ORIGEM]
```

Onde:

<b><i>mkisofs</i></b>	
<i>-C</i>	Para criação de imagens com múltiplas sessões.
<i>-J</i>	Habilita as extensões <i>Joilet</i> , requerimento indispensável para manter compatibilidade com os demais sistemas operacionais (como o

<b><i>mkisofs</i></b>	
	<i>Windows</i> , por exemplo).
<b><i>-l</i></b>	Habilita o suporte à nomes longos de até 31 caracteres. O padrão <i>ISO9660</i> suporta apenas o formato 8.3, que por sua vez é compatível com o <i>MS-DOS</i> . Deverá ser usado com precauções, conforme a documentação do programa.
<b><i>-L</i></b>	Permite a gravação de arquivos ocultos – os que iniciam com a utilização do ponto (.).
<b><i>-o</i></b>	Especifica o nome da imagem à ser gerada.
<b><i>-r</i></b>	Abreviação de <i>Rock Ridge</i> , um protocolo que possibilita evitarmos a truncagem dos nomes de arquivos longos para a gravação do <i>CD-R/RW</i> .
<b><i>-v</i></b>	Modo <i>verbose</i> – mostra todo o processo em execução.
<b><i>-V</i></b>	Define o <i>label</i> – nome do volume.

Como exemplo, criaremos uma imagem simples apenas com uma pequena estrutura de diretórios para teste.

```
$ mkisofs -J -r -o TESTE.iso /home/darkstar/teste
Total translation table size: 0
Total rockridge attributes bytes: 1693
Total directory bytes: 0
Path table size(bytes): 10
Max brk space used 8284
464 extents written (0 Mb)
$ _
```

Estes são os mínimos comandos necessários para se criar uma imagem para serem gravadas posteriormente em *CD-ROM*.

## CDRECORD

O *cdrecord* é o utilitário padrão dos sistemas *GNU/Linux* para realizar as atividades de gravação de *CD-R/RW*. Atualmente suporta apenas interfaces *SCSI*, e eis o motivo pelo qual necessitamos habilitar a emulação *SCSI* para acessarmos nossas unidades na controladora *IDE*.

Sintaxe geral:

```
$ cdrecord [PARÂMETROS]
```

Sintaxe para gravação:

```
$ cdrecord -fs=[BUFFER]M speed=[VELOCIDADE] dev=[LOCALIZAÇÃO] -
[ESPECIFICAÇÃO] [IMAGEM].iso
```

Onde:



<b><i>cdrecord</i></b>	
<b><i>-blank=[TIPO]</i></b>	Apaga os dados armazenados em uma mídia <i>CD-RW</i> . Os subparâmetros (tipos) específicos deste são: <i>all</i> -> Apaga tudo; <i>help</i> -> Ajuda; <i>session</i> -> Uma sessão (para <i>CD-R/RW</i> multisessão); <i>track</i> -> faixa de áudio.
<b><i>-dev[N,N,N.]</i></b>	Localização da unidade <i>SCSI</i> em questão (veja <i>-scanbus</i> ).
<b><i>-dummy</i></b>	Realiza apenas um teste ao invés de realizar a gravação.
<b><i>-eject</i></b>	Ejeta o disco no final da operação.
<b><i>[ESPECIFIC.]</i></b> <b><i>-data, -audio</i></b>	Define qual tipo de imagem será criado: <i>-data</i> (para dados), <i>-audio</i> (para <i>CDs</i> de áudio).
<b><i>-fs=[BUFFER]</i></b>	Especifica o tamanho do <i>buffer</i> de memória para a gravação. Quando omitido, o valor padrão é <i>4 MB</i> .
<b><i>-multi</i></b>	Utiliza o recurso de multi-sessão.
<b><i>-scanbus</i></b>	Exibe os dados da unidade do sistema.
<b><i>-speed=[VEL.]</i></b>	A velocidade de gravação (a máxima deverá ser a suportada pelo aparelho e mídia).
<b><i>-v</i></b>	Exibe mensagens sobre o andamento do processo.

Lembrem-se de que – dependendo da versão utilizada – o *cdrecord* apenas grava em unidades *SCSI* (os quais as unidades *IDE* são emuladas pelo *kernel* através do módulo *ide-scsi*). Nas versões atuais esta restrição já não existe.

## Exemplos

Segue um simples exemplo para:

<b><i>Exemplos</i></b>	
<b><i>Dados (ISO)</i></b>	<code>\$ cdrecord -v -fs=8M speed=16 dev=0,0,0 -data backup.iso</code>
<b><i>Áudio</i></b>	<code>\$ cdrecord -v -fs=8M speed=16 dev=0,0,0 -audio [FAIXA1], [FAIXA2], [FAIXA3], [...]</code>
<b><i>Dados + áudio</i></b>	<code>\$ cdrecord -v -fs=8M speed=16 dev=0,0,0 -data backup.iso -audio [FAIXA1], [FAIXA2], [FAIXA3], [...]</code>

Lembrem-se de que, apesar de improvável, de acordo com o equipamento o *device* correspondente ao do usuário poderá ser diferente ao acima especificado (*dev=0,0,0*). Para descobrir o *dev* correspondente, utilize:

```
$ cdrecord -scanbus
Cdrecord 2.00.3 (i686-pc-linux-gnu) Copyright (C) 1995-2002 Jörg Schilling
Linux sg driver version: 3.1.25
Using libscg version 'schily-0.7'
scsibus0:
  0,0,0      0) 'HL-DT-ST' 'CD-RW GCE-8525B ' '1.03' Removable CD-ROM
  0,1,0      1) *
```

```

0,2,0    2) *
0,3,0    3) *
0,4,0    4) *
0,5,0    5) *
0,6,0    6) *
0,7,0    7) *
$ _

```

Já as faixas de áudio deverão ser arquivos nos formatos de *.wav* e/ou *.cdr*.

### *Outros recursos*

Um recurso interessante do *cdrecord* está na possibilidade de se descobrir todos os dados de uma determinada mídia de *CD-R/RW*. Para isto, basta inseri-lo na bandeja e (sem montar a unidade) digitar o seguinte comando:

```
# cdrecord -atip dev=0,0,0
```

A linha que nos importa será a seguinte:

```
Manufacturer: CMC Magnetics Corporation
```

Além do fabricante, observe que também são exibidas outras informações gerais, tanto da unidade gravadora quanto da mídia utilizada.

```

$ cdrecord -atip dev=0,0,0
Cdrecord 2.00.3 (i686-pc-linux-gnu) Copyright (C) 1995-2002 Jörg Schilling
scsudev: '0,0,0'
scsibus: 0 target: 0 lun: 0
Linux sg driver version: 3.1.25
Using libscg version 'schily-0.7'
Device type      : Removable CD-ROM
Version          : 0
Response Format: 2
Capabilities     :
Vendor_info      : 'HL-DT-ST'
Identifikation   : 'CD-RW GCE-8525B '
Revision         : '1.03'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags     : MMC-2 SWABAUDIO BURNFREE
Supported modes: TAO PACKET SAO SAO/R96P SAO/R96R RAW/R16 RAW/R96P RAW/R96R
cdrecord: Input/output error. test unit ready: scsi sendcmd: no error
CDB: 00 00 00 00 00 00
status: 0x2 (CHECK CONDITION)
Sense Bytes: 70 00 02 00 00 00 00 0A 00 00 00 00 3A 01 00 00
Sense Key: 0x2 Not Ready, Segment 0
Sense Code: 0x3A Qual 0x01 (medium not present - tray closed) Fru 0x0
Sense flags: Blk 0 (not valid)
cmd finished after 0.000s timeout 40s
cdrecord: No disk / Wrong disk!
bash-2.05b$ cdrecord -atip dev=0,0,0
Cdrecord 2.00.3 (i686-pc-linux-gnu) Copyright (C) 1995-2002 Jörg Schilling
scsudev: '0,0,0'
scsibus: 0 target: 0 lun: 0
Linux sg driver version: 3.1.25
Using libscg version 'schily-0.7'
Device type      : Removable CD-ROM
Version          : 0
Response Format: 2

```

```

Capabilities      :
Vendor_info       : 'HL-DT-ST'
Identifikation    : 'CD-RW GCE-8525B '
Revision          : '1.03'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags      : MMC-2 SWABAUDIO BURNFREE
Supported modes:
ATIP info from disk:
  Indicated writing power: 5
  Is not unrestricted
  Is not erasable
  Disk sub type: Medium Type A, high Beta category (A+) (3)
  ATIP start of lead in: -11634 (97:26/66)
  ATIP start of lead out: 359846 (79:59/71)
Disk type:        Short strategy type (Phthalocyanine or similar)
Manuf. index: 3
Manufacturer: CMC Magnetics Corporation
$ _

```

## CONCLUSÃO

A necessidade de ter conhecimentos mais aprofundados para a manipulação de unidades, partições e sistemas de arquivos em sistemas *GNU/Linux* em comparação ao *Windows* pode tornar sua utilização cansativa nestas atividades (em alguns casos, complicados), porém serão vitais para realizar atividades de manutenção em situações que não tenhamos disponíveis ambientes gráficos e suas respectivas ferramentas.

Para aqueles que não simpatizam muito com a utilização da linha de comando, recomendamos realizarem uma boa pesquisa sobre as distribuições existentes para esta finalidade. Por exemplo, apesar de não focar unicamente esta área, o *Kurumim* possui bons utilitários disponíveis no *CD-ROM*, bastando apenas colocá-lo na unidade, reiniciar o sistema à partir do *drive* de *CD-ROM* e aguardar o carregamento *KDE*. &;-D



# VI. CONTAS DE USUÁRIOS E GRUPOS DE ACESSO

---

## INTRODUÇÃO

O conhecimento em administração de contas usuários, grupos de acesso e edição de arquivos de configuração pertinentes é de extrema importância para assegurar a boa coexistência entre diferentes utilizadores de um único sistema *GNU/Linux*. E justamente isto é o que propomos aos leitores um estudo mais aprofundado deste capítulo.

## CONSIDERAÇÕES BÁSICAS

### CONTAS DE ACESSO

Conforme já dito anteriormente, os sistemas *GNU/Linux* são ambientes multi-usuários. Apesar disto – e de acordo com sua categoria – nem todos possuem um mesmo perfil, necessidades e/ou responsabilidades. Para isto existem as contas de autenticação – ou contas de acesso.

Por mais diferentes que tais perfis sejam, estes se enquadram em duas classes distintas: Superusuário – também conhecido como administrador do sistema –, e usuário comum – ou simplesmente usuários.

### ROOT - O ADMINISTRADOR DO SISTEMA

Conforme já comentado diversas vezes, o *root* é o administrador do sistema – muitas vezes também chamado de superusuário. De acordo com a distribuição utilizada, existirá um momento na instalação do sistema em que será solicitado a senha para acesso do superusuário.

```
Changing password for root
```

```
Enter the new password (minimum of 5, maximum of 127 characters)
```

```
Please use a combination of upper and lower case letters and numbers.
```

```
New password: _
```

Somente o *root* possui acesso total ao sistema instalado. É ele quem define as permissões gerais para acesso aos recursos e dados gerais do sistema para o usuário. Em algumas circunstâncias (de acordo com a distribuição utilizada), a sua única limitação existente está no acesso à *Internet*. Por tratar-se de um usuários com acesso total ao sistema, muitas distribuições definiram as permissões de acesso referentes à *Internet* somente disponíveis para os usuários, que por sua vez, caso sejam acessados involuntariamente por invasores da rede, estes não terão as permissões necessárias para ocasionar males ao sistema. Porém em algumas distribuições como o *Slackware*, por ser destinada à usuários de médio e alto nível técnico, esta limitação não existe, deixando a responsabilidade à cargo dos administradores do sistema.

Após autenticado, o símbolo de indicador na linha de comando é:

B



```
Login: root
Passwd:
# _
```

## CONTA DE USUÁRIO

A conta de usuário – ou conta comum –, diferente do superusuário, pois possui apenas permissões básicas para a realização de atividades gerais no sistema e algumas configurações locais, onde não é possível a realização de intervenções de encargo do superusuário, como a instalação / atribuição de permissão / configuração / remoção de programas, arquivos e dispositivos do sistema, entre outros.

Porém, de acordo com as definições do superusuário, as configurações gerais de permissão do usuário poderão ser editadas para que possam atender a determinadas circunstâncias. Por exemplo, podemos atribuir grupos para que estes tenham permissão para acessar a *Internet*, ao sistema de impressão, etc.

Após autenticado, o símbolo de indicador na linha de comando é:

```
Login: darkstar
Passwd:
$ _
```

## GRUPOS DE ACESSO

Grupos de acesso são definições especiais de permissões para serem utilizadas quando houver a necessidade de disponibilizar o acesso à um sistema de dados (arquivos e diretórios) ou um determinado serviço para uma categoria distinta. Por exemplo, num escritório de recursos humanos, onde as planilhas de contabilidade somente poderão ser acessadas pelos profissionais de contabilidade, poderão ser definidos um grupo “*contabilidade*” para estes, onde também o grupo e as permissões de acesso para cada arquivo deverá estar previamente definido.

## ID

O *ID* é um número de identificação para qualquer um dos elemento do sistema – usuário e grupo. Este por sua vez se subdivide em:

<i>ID</i>	
<i>GID</i>	Define-se <i>GID</i> a numeração de <i>ID</i> específica para os grupos de acesso do sistema. Todos os <i>IDs</i> de contas de grupos de acesso pertencem ao intervalo de 1 à 499.
<i>UID</i>	Define-se <i>UID</i> a numeração de <i>ID</i> específica para as contas de usuário do sistema. À exceção do superusuário ( <i>UID</i> = 0), todos os <i>IDs</i> de contas de usuário iniciam à partir de 500.



## PERMISSÕES

Cada arquivo, diretório e/ou dispositivo do sistema e respectivos recursos, são regidos por um sistema de grupos e permissões de acesso que definem qual o sistema de acesso, tipo de acesso e limitações à estas itens. Para evitar estender este assunto, consultem nesta parte o capítulo *Manipulação de arquivos e diretórios* para obterem maiores informações.

## SENHA

Sem maiores dúvidas, os usuários dos sistemas *GNU/Linux* somente poderão ter acesso as suas contas de acesso e tudo o que nela estiver após definirem uma senha de autenticação. Durante a criação de uma nova conta de acesso, será solicitado ao usuário a definição de uma senha, o qual bastará o usuário utilizar um conjunto de caracteres.

## COMANDOS E PARÂMETROS GERAIS

### ADIÇÃO DE CONTAS DE USUÁRIO E GRUPOS

#### ADDUSER

O comando *adduser* é utilizado para criar contas de usuários e é somente utilizado pelo administrador do sistema.

Sintaxe:

```
# adduser [APELIDO]
```

...ou simplesmente...

```
# adduser
```

Exemplificaremos detalhadamente o processo de criação da conta de um usuário. Para começar, digite apenas o comando com administrador do sistema, onde será solicitado um apelido (*nick*) para o usuário.

```
Login name for new user []: darkstar
```

```
- User 'Darkstar' contains illegal characters (uppercase); please choose another  
Login name for new user []: _
```

Observe que o comando não aceita o uso de caracteres iniciais maiúsculas (caixa alta), solicitando novamente a inclusão do apelido desejado.

```
Login name for new user []: darkstar
```

```
User ID ('UID') [ defaults to next available ]: _
```

Nesta seção deverá ser definido o *UID* do usuário. Apenas teclem <ENTER> para que o sistema possa definir uma *UID* disponível.

```
Initial group [ users ]: _
```

Aqui deverá ser definido o grupo padrão do usuário ou o grupo inicial, conforme a descrição acima. Digitem o grupo desejado ou apenas teclem

<ENTER> para defini-lo no grupo padrão do sistema (*users*).

```
Additional groups (comma separated) []: _
```

Nesta próxima etapa, o comando solicita a informação de outros grupos para a conta à ser criada. O usuário terá acesso à recursos de acordo com os grupos inclusos para este. Para obterem maiores informações, consultem a seção *Considerações básicas -> Grupos de acesso*.

Após definir os grupos de acesso, teclem <ENTER>.

```
Home directory [ /home/darkstar ] _
```

O sistema definirá o diretório do usuário utilizando o próprio nome da conta. Caso haja necessidade de definir outro caminho, digitem-no na linha de comando ou simplesmente teclem <ENTER> para manter o padrão.

```
Shell [ /bin/bash ] _
```

Definição padrão do interpretador da linha de comando é o *Bash*, indicado acima na opção *Shell*. Caso queiram utilizar outro que não seja o *Bash*, basta especificá-lo aqui. Por exemplo, para o interpretador *Ash*, */bin/ash*; para o *Csh*, */bin/csh*. Prefiram manter o interpretador padrão, apenas teclando <ENTER>.

```
Expiry date (YYYY-MM-DD) []: _
```

A data de validade poderá ser definida nesta seção, bastando apenas digitá-la no formato *ANO-MÊS-DIA*. Caso queiram que esta não tenha prazo de expiração definido, apenas teclem <ENTER>.

New account will be created as follows:

```
-----
Login name.....:  darkstar
UID.....:         [ Next available ]
Initial group....:  users
Additional groups:  [ None ]
Home directory...:  /home/darkstar
Shell.....:        /bin/bash
Expiry date.....:  [ Never ]
```

This is it... if you want to bail out, hit Control-C. Otherwise, press ENTER to go ahead and make the account.

Enfim, tendo a conta criada, será mostrado conforme acima todos os dados digitados pelo superusuário para a criação da nova conta. Logo em seguida, será solicitado pelo sistema os dados adicionais para a nova conta criada. Tecle <ENTER> e iniciaremos o cadastro das informações adicionais.

Creating new account...

Changing the user information for *darkstar*

Enter the new value, or press ENTER for the default

```
Full Name []:
Room Number []:
Work Phone []:
Home Phone []:
```



Other []:

Preencha os dados solicitados como *nome completo, endereço, telefone do trabalho, telefone do lar*, além de *outras* informações que considerarem necessárias, se assim desejarem (estas informações são opcionais). Por último, deverá ser definido a senha de acesso.

Changing password for **darkstar**

Enter the new password (minimum of 5, maximum of 127 characters)  
Please use a combination of upper and lower case letters and numbers.  
New password: \_

Ao utilizarem um conjunto de caracteres curtos – com 5 ou menos – o comando rejeitará a cadeia definida devido à sua política de segurança para a definição de senhas, onde há o requerimento mínimo de 6 caracteres.

Bad password: too short.

Warning: weak password (enter it again to use it anyway).

New password: \_

Novamente defina uma nova senha de acesso respeitando esta política. Se a cadeia de caracteres estiver de acordo como o exigido, o comando solicitará para que seja repetida a sequência utilizada para confirmar. Este procedimento é ideal para aquelas circunstâncias em que digitamos algum número ou caracter diferente do desejado, possibilitando com isto corrigir a senha definida.

New password:

Re-enter new password:

No final da operação, serão exibidas as seguintes mensagens:

Password changed.

Account setup complete.

# \_

## GROUPADD

Adiciona grupos ao sistema operacional.

Sintaxe:

```
# groupadd [PARÂMETROS] [GRUPO]
```

Onde:

<b>groupadd</b>	
<b>-g</b>	Define manualmente um <i>ID</i> específico para o grupo.
<b>-r</b>	Adiciona uma conta do sistema.
<b>-f</b>	Modo forçado – mantém um grupo já existente no sistema.

Basta digitarem...

```
# groupadd suporte
```



... para que seja criado um novo grupo chamado *suporte*.

## ADMINISTRAÇÃO DE CONTAS

### LOGIN / LOGOUT / EXIT

Toda vez que o sistema é inicializado, você deverá notar a existência de uma linha de comando com a seguinte mensagem:

```
login: _
```

Sua finalidade é a autenticação do usuário: para que possamos acessar o sistema e usufruir de seus recursos, teremos que nos tornar “*autênticos*”.

```
login: darkstar
Password: [SENHA]
$ _
```

E como fazer para nos “*deslogarmos*”? Para isto, existe o comando *logout*.

```
$ logout
```

Outra forma de finalizar uma seção é utilizando o comando *exit*:

```
$ exit
```

Na utilização destes comandos nas interfaces gráficas, como o *Konsole* ou o *XTerm*, estes apenas têm suas caixas de diálogos finalizadas.

### ID

O comando *id* é utilizado basicamente para obter informações dos *IDs* dos usuários autenticados no sistema, como o seu próprio e também o dos grupos os quais este se encontra.

Para a sua utilização, basta apenas digitar o comando com as contas desejadas autenticadas. Observe nos exemplos abaixo que as informações são exibidas em duas categorias: *UID* (*ID* do usuário) e *gid* (*ID* do grupo).

Superusuário:

```
# id
uid=0(root) gid=0(root)
grupos=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel),11(floppy)
# _
```

Usuário comum:

```
$ id
uid=1000(darkstar) gid=100(users) grupos=100(users),14(uucp)
$ _
```

### USERS / GROUPS

Exibem respectivamente os usuários autenticados naquele momento e seus respectivos grupos de acesso.

- B

<i>Conta</i>	<i>users</i>	<i>groups</i>
<i>Superusuário</i>	\$ users root \$ _	\$ groups root bin daemon sys adm disk wheel floppy \$ _
<i>Usuário</i>	\$ users darkstar \$ _	\$ groups users disk floppy uucp audio video cdrom \$ _

## PASSWD

O comando *passwd* é utilizado para especificar uma nova senha ou alterar a senha atual de conta de usuário. Basta digitarem...

```
# passwd [USUÁRIO]
```

... ou apenas...

```
$ passwd
```

... para alterarmos a senha do usuário desejado. Segue um exemplo simples para a alteração de uma senha feita pelo administrador, para melhor entendimento.

```
# passwd darkstar
```

```
Changing password for darkstar
```

```
Old password: _
```

Observem que o comando não aceita de forma alguma a utilização de novas senhas que não possuam um mínimo de 5 caracteres.

```
Enter the new password (minimum of 5, maximum of 127 characters)
```

```
Please use a combination of upper and lower case letters and numbers.
```

```
New password:
```

```
passwd: all authentication tokens update sucessfully
```

```
# _
```

Ao tentar utilizarmos uma senha fora dos requerimentos necessários...

```
New password: [SENHA]
```

```
Bad password: no change. Try again.
```

Dentro de uma seção de usuário, bastará utilizarem apenas...

```
$ passwd
```

... e o sistema solicitará que o mesmo atualize a senha desta conta.

```
$ passwd
```

```
Changing password for darkstar
```

```
Old password: [SENHA ANTIGA]
```

```
Enter the new password (minimum of 5, maximum of 127 characters)
```

```
Please use a combination of upper and lower case letters and numbers.
```

```
New password: [SENHA NOVA]
```

```
passwd: all authentication tokens update sucessfully
```

```
$ _
```

B

## FINGER

Exibe informações gerais de autenticação das contas do sistema.

Sintaxe:

```
$ finger [PARÂMETROS] [USUÁRIO]
```

Onde:

<i>finger</i>	
-s	Informações adicionais (nome completo, telefone, etc., ou seja, todas as informações solicitadas no ato da criação da conta).
-l	Exibe as informações em diversas linhas ( <i>multi-line</i> ).

A conta *darkstar* esta conectada no momento da utilização deste comando...

```
$ finger darkstar
Login: darkstar                      Name: (null)
Directory: /home/darkstar           Shell: /bin/bash
On since Tue Apr  6 09:27 (UTC) on :0 (messages off)
On since Tue Apr  6 09:27 (UTC) on pts/0  2 hours 25 minutes idle
On since Tue Apr  6 09:42 (UTC) on pts/2 (messages off)
No mail.
No Plan.
$ _
```

Ainda autenticado como *darkstar*...

```
$ finger root
Login: root                          Name: (null)
Directory: /root                    Shell: /bin/bash
Last login Fri Apr  2 18:00 (UTC) on tty1
Mail last read Tue Mar 30 23:23 2004 (UTC)
No Plan.
$ _
```

Observem que, pelo fato do superusuário não estar autenticado, o comando apenas exibe a última vez em que este autenticou-se no sistema.

## WHO

Mostra quais são os usuários que se encontram autenticados no momento.

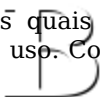
Sintaxe:

```
$ who [PARÂMETROS]
```

Segue um simples exemplo, com o usuário *darkstar*:

```
$ who
darkstar :0          Apr  6 09:27
darkstar pts/0       Apr  6 09:27
darkstar pts/1       Apr  6 12:24
darkstar pts/2       Apr  6 09:42
$ _
```

Existem diversos parâmetros dos quais para nós, simples usuários, não teremos maiores necessidades de uso. Consultem o manual eletrônico para



obterem maiores detalhes, se desejarem.

**WHOAMI**

Mostra qual o usuário está autenticado no sistema no momento.

Sintaxe:

```
$ whoami
```

Segue um simples exemplo, com o usuário *darkstar*:

```
$ whoami
darkstar
$ _
```

**UPTIME**

Exibe o tempo de autenticação do usuário no sistema.

Sintaxe:

```
$ uptime
```

Ao digitar o comando acima descrito...

```
$ uptime
12:29:37 up 3:13, 3 users, load average: 0.03, 0.05, 0.00
$ _
```

**ELIMINANDO USUÁRIOS E GRUPOS**

**USERDEL**

Elimina a conta de usuário do sistema.

Sintaxe:

```
# userdel [PARÂMETROS] [USUÁRIO]
```

Onde:

<i>userdel</i>	
-r	Elimina todos os dados gravados em seu diretório pessoal (/home/[USUÁRIO]).

Segue um exemplo simples e prático:

```
# userdel darkstar
```

Caso não sejam mais necessários os dados arquivados em seu diretório pessoal, utilizem o parâmetro *-r* desta forma:

```
# userdel -r darkstar
```





## GROUPDEL

Elimina o grupo de acesso do sistema.

Sintaxe:

```
# groupdel [GRUPO]
```

Uma dica importante encontrada na documentação eletrônica do comando está na restrição da remoção de um grupo primário de um grupo existente, onde a prévia remoção dos usuários faz-se necessária.

## OBTENDO OS PRIVILÉGIOS DE OUTROS USUÁRIOS

Apesar da pouca utilização das acessibilidades dos demais usuários – e especialmente o superusuário –, existirão circunstâncias em que haverá a necessidade da obtenção de específicas permissões, os quais poderão estar disponíveis com a utilização dos seguintes comandos:

### SU

Apesar possuírem plenos poderes, os superusuários somente devem utilizar sua senha de acesso apenas para a administração do sistema, onde sua utilização para tarefas triviais de usuários, além de desnecessária, poderá acarretar riscos à integridade do sistema. Na maioria das vezes estes necessitam apenas de uma simples conta de acesso para fazer uso dos recursos do sistema em si, porém poderão surgir algumas circunstâncias em que será necessário realizar intervenções ao sistema. Mas como fazer isto, sem ter que estar autenticando-se à todo momento?

Para obtermos os privilégios de administrador do sistema, sem ter que nos “deslogarmos”, deveremos utilizar o comando *su*. Sua sintaxe é simples e básica, onde bastará apenas digitar na linha de comando...

```
$ su
Password: [SENHA DO SUPERUSUÁRIO]
# _
```

E fornecer a senha de superusuário para ter as permissões de acesso desejadas. Note que indicador também mudou (\$ -> #).

A utilidade deste comando basicamente é esta: ter acesso aos poderes de superusuário apenas para realizar intervenções básicas e rápidas, retornando logo em seguida às suas atividades comuns.

Dentre algumas limitações deste recurso está a impossibilidade de executar algumas aplicações que façam a utilização das bibliotecas gráficas – estes sequer se encontrarão disponíveis, ao digitar as *1as.* Iniciais do executável e teclar <TAB>. Por exemplo, ao utilizar o comando *su* para obter os privilégios de superusuário e tentar executar...

```
# kppp
bash: kppp: command not found
# _
```



... o interpretador retorna uma mensagem de que não se encontra o programa desejado, mesmo estando ciente de que se encontra instalado no sistema e que o superusuário tem plenos poderes para a sua execução.

Dependendo destas limitações, talvez será até mesmo necessário autenticar-se novamente ao sistema como superusuário e iniciar o ambiente gráfico para realizarmos as atividades administrativas desejadas.

Para retornar ao estado anterior, bastará apenas digitarmos...

```
# exit
$ _
```

## OS ARQUIVOS DE CONFIGURAÇÃO

### /ETC/PASSWD

Em */etc/passwd* estão as definições gerais de usuários cadastrados no sistema. Ao ser adicionado um novo usuário, todas as informações geradas são gravadas neste arquivo, como o apelido, o *UID*, diretório padrão e interpretador de comandos. Porém as senhas são armazenadas de forma criptografadas em outro arquivo, o */etc/shadow*.

```
root:x:0:0:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/log:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/:
news:x:9:13:news:/usr/lib/news:
uucp:x:10:14:uucp:/var/spool/uucppublic:
operator:x:11:0:operator:/root:/bin/bash
games:x:12:100:games:/usr/games:
ftp:x:14:50:./home/ftp:
smmsp:x:25:25:smmsp:/var/spool/clientmqueue:
mysql:x:27:27:MySQL:/var/lib/mysql:/bin/bash
rpc:x:32:32:RPC portmap user:/bin/false
sshd:x:33:33:sshd:/:
gdm:x:42:42:GDM:/var/state/gdm:/bin/bash
pop:x:90:90:POP:/:
nobody:x:99:99:nobody:/:
darkstar:x:1000:100:./home/darkstar:/bin/bash
```

### /ETC/SHADOW

No arquivo */etc/shadow* é que são armazenadas todas as senhas de usuário no sistema, porém utilizando o recurso de criptografia.

```
root:$1$dju/fKjR$RfpzV10HJySdQgAaR2Q921:12517:0::::::
bin:!:9797:0::::::
daemon:!:9797:0::::::
adm:!:9797:0::::::
```



```
lp*:9797:0::::  
sync*:9797:0::::  
shutdown*:9797:0::::  
halt*:9797:0::::  
mail*:9797:0::::  
news*:9797:0::::  
uucp*:9797:0::::  
operator*:9797:0::::  
games*:9797:0::::  
ftp*:9797:0::::  
smmsp*:9797:0::::  
mysql*:9797:0::::  
rpc*:9797:0::::  
sshd*:9797:0::::  
gdm*:9797:0::::  
pop*:9797:0::::  
nobody*:9797:0::::  
darkstar:$1$9xI0RViR$6ZdNL8e.wTN99xkKEqfJx1:12517:0:99999:7:::
```

## **/ETC/GROUPS**

No arquivo */etc/groups* estão todas as definições de grupos de autenticação. Todos os grupos padrões do sistema, mais os grupos específicos das aplicações e ainda os grupos criados pelo administrador possuem suas especificações aqui descritas.

```
root::0:root  
bin::1:root,bin,daemon  
daemon::2:root,bin,daemon  
sys::3:root,bin,adm,darkstar  
adm::4:root,adm,daemon  
tty::5:  
disk::6:root,adm  
lp::7:lp  
mem::8:  
kmem::9:  
wheel::10:root  
floppy::11:root  
mail::12:mail  
news::13:news  
uucp::14:uucp,darkstar  
man::15:  
games::20:  
slocate::21:  
utmp::22:  
smmsp::25:smmsp  
mysql::27:  
rpc::32:  
sshd::33:sshd  
gdm::42:  
shadow::43:  
ftp::50:  
pop::90:pop  
nobody::98:nobody  
nogroup::99:  
users::100:darkstar  
console::101:
```



## PROBLEMAS MAIS FREQUENTEMENTE

### *Perdi a senha de usuário. O quê fazer?*

Editem o arquivo de configuração `/etc/shadow` e procure a linha correspondente ao apelido do usuário em questão:

```
darkstar:$1$xRKiwuNz$rfBSEiom7PIEtCgVq169G1:12312:0:99999:7:::
```

Observe realçado na cor vermelha a existência de um conjunto de caracteres entre os dois primeiros limitadores ":". Deveremos apenas apagar estes dados, salvar o arquivo de configuração e reiniciar a autenticação. Digitem normalmente o apelido e, ao ser solicitado a senha, apenas tecle <ENTER>. Pronto! Estaremos novamente autenticado.

Em seguida, utilizem o comando `passwd` e mudem a senha normalmente. Se preferirem, façam antes uma cópia de segurança do arquivo de configuração para garantir que não sejam feitas alterações descuidadas ou involuntárias.

### *E se for a senha do superusuário?*

O procedimento para a recuperação da senha do superusuário é o mesmo, onde deverá ser procurado o usuário *root* para editar o arquivo de configuração. Porém, teremos a necessidade de utilizarmos disquete de recuperação criado durante a instalação do *Slackware* para inicializar o sistema, já que sem a senha do superusuário, não há como nos autenticar. Caso não possuam o disquete de inicialização, consultem nesta parte o capítulo *Unidades, partições e sistemas de arquivos*, para obterem informações adicionais de como proceder para desenvolver um novo disquete – e de preferência antes da ocorrência do problema... &;-D

## CONCLUSÃO

Apesar da existência de diversos comandos e recursos para a administração de contas, grupos e outras propriedades de um sistema multi-usuário, o bom conhecimento das atividades de gerenciamento de usuários é muito importante, visto que além de manter todo o sistema de dados em segurança, não teremos ocorrências e inconvenientes desagradáveis como a perda de privacidade ou reclamações deste tipo do próprio usuário ou de outras pessoas que fizeram uso da máquina. Além disso, nos dará liberdade de personalizarmos o sistema de acordo com nossas preferências. &;-D

## VII. PROCESSOS EM EXECUÇÃO

---

### INTRODUÇÃO

Conforme enfatizamos diversas vezes, o *kernel* é o principal responsável pelo gerenciamento das atividades do sistema, que por sua vez são caracterizadas como processo. Pelo fato do sistema operacional ser multitarefa, existem diversos processos em andamento. Neste capítulo iremos conhecer como os sistemas *GNU/Linux* (mais precisamente o *kernel*) administram os processos em execução, bem como todas as instruções e comandos para um bom gerenciamento destes.

### VISÃO GERAL

#### O QUE É UM PROCESSO?

O processo é qualquer atividade executada pelo sistema que envolve uma rotina de instruções com seus respectivos dados e informações. Um processo poderá ser um programa em execução, um comando sendo acionado, uma chamada do sistema, etc.

Existem 3 tipos de processos distintos:

<i><b>Tipos de processos</b></i>	
<i>Interativos</i>	São processos executados à partir e controlados por um terminal. Ex.: Comandos gerais da linha de comando.
<i>Lotes</i>	São todos os processos agendados pelo usuário e/ou o sistema. Ex.: Programação de tarefas e atividades com o <i>cron</i> .
<i>Servidores</i>	São processos executados na inicialização do sistema e que necessitam estar em execução permanente para a utilização de seus serviços por outros processos que o necessitem. Ex.: O banco de dados <i>MySQL</i> e o servidor <i>Apache</i> .

### PID

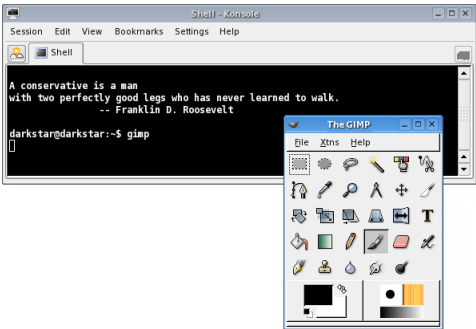
O *PID* é nada mais que uma identificação numérica de um determinado processo existente, que por sua vez, quando criado, passa à ser identificado e manipulado por este número. Já o *PPID* de um processo é nada mais que a informação do processo-pai que o gerou. Seria análogo como a aplicação que executou outra aplicação.

### 2o. PLANO (BACKGROUND)

Muitas vezes, ao executarmos um aplicativo na linha de comando, esta fica indisponível até o encerramento do mesmo. Apesar da disponibilidade de

12

vários terminais virtuais, seria deselegante toda vez que executarmos um programa, abrir um terminal que deverá ficar aberto até o encerramento deste, poluindo e dificultando o gerenciamento das janelas abertas em seu ambiente gráfico preferido.



*Observem que, quando o GIMP é executado, o sinal de prontidão fica indisponível.*

Para isto temos o recurso *background*, ou seja, a possibilidade de executar um determinado programa ou atividade na linha de comando e ainda mantê-la disponível conforme nossas necessidades. O processo gerado para a execução do programa fica em segundo plano, o qual poderá ser manipulado através de outros comandos pela própria linha de comando, desta vez liberada para outras atividades.

## GERENCIANDO PROCESSOS

Os principais comandos para o gerenciamento de processos são:

### VISUALIZAÇÃO

#### PS

Exibe os processos os quais estão sendo rodados no sistema.

Sintaxe:

```
$ ps [PARÂMETROS]
```

Onde:

<i>ps</i>	
-A	Exibe todos os processos.
-a	Exibe os processos somente da seção corrente.
-p [N]	Verifica a existência de um determinado processo, onde [N] é o número do processo.
-u	Mostra os processos iniciados pelo usuário.

Experimentem os seguintes comandos e analisem os resultados obtidos de acordo com a tabela apresentada:

```
$ ps
$ ps -a
$ ps -A
$ ps -p [NÚMERO DE UM PROCESSO PRÉ-VISUALIZADO COM OS COMANDOS ANTERIORES]
$ ps -u
$ -
```

Para obterem maiores detalhes, consultem a sua documentação eletrônica.

**TOP**

Exibe todos os processos em execução, além de outros fatores importantes, como a utilização geral da *CPU* e ocupação da memória.

Sintaxe:

```
$ top [PARÂMETROS]
```

O *top* possui vários parâmetros disponível, mas para utilizarmos suas funcionalidades básicas, apenas digitem na linha de comando...

```
$ top
```

..., onde em poucos instantes...

```
top - 18:40:14 up 37 min, 1 user, load average: 0.07, 0.03, 0.00
Tasks: 46 total, 2 running, 44 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.7% user, 0.3% system, 0.0% nice, 99.0% idle
Mem: 515444k total, 295952k used, 219492k free, 40912k buffers
Swap: 506008k total, 0k used, 506008k free, 170472k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM     TIME+  COMMAND
  159 root        16   0 79428  12m 2244 S   0.3   2.6   0:33.61 X
  538 darkstar   12   0 15948  15m 13m R   0.3   3.1   0:00.45 kdeinit
  551 darkstar  12   0  988   988 800 R   0.3   0.2   0:00.08 top
    1 root         8   0  236   236 208 S   0.0   0.0   0:04.77 init
    2 root         9   0    0    0  0 S   0.0   0.0   0:00.02 keventd
    3 root        19  19    0    0  0 S   0.0   0.0   0:00.00 ksoftirqd_CPU0
    4 root         9   0    0    0  0 S   0.0   0.0   0:00.00 kswapd
    5 root         9   0    0    0  0 S   0.0   0.0   0:00.00 bdflush
    6 root         9   0    0    0  0 S   0.0   0.0   0:00.10 kupdated
   10 root        -1 -20    0    0  0 S   0.0   0.0   0:00.00 mdrecoveryd
   11 root         9   0    0    0  0 S   0.0   0.0   0:00.00 kreiserfsd
   35 root         9   0    0    0  0 S   0.0   0.0   0:00.00 kapmd
   81 root         9   0  592  592 512 S   0.0   0.1   0:00.02 syslogd
   84 root         9   0  444  444 388 S   0.0   0.1   0:00.01 klogd
  124 root         9   0  520  520 464 S   0.0   0.1   0:00.00 inetd
  134 root         9   0  528  528 464 S   0.0   0.1   0:00.00 crond
  136 daemon       9   0  616  616 548 S   0.0   0.1   0:00.00 atd
  140 root         9   0  500  500 448 S   0.0   0.1   0:00.00 apmd
```

Bastam analisar as informações gerais disponíveis na saída do *monitor*. O comando também possui umas teclas especiais para funcionalidades específicas, as quais seguem:



<b>Teclas &amp; Funções</b>	
<ESPAÇO>	Atualiza a tela.
<CTRL>+<L>	Atualiza a tela, porém apagando o conteúdo anterior.
<H> ou <?>	Ajuda eletrônica.
<I>	Ignora processos “zumbis” (ociosos).
<K>	<i>Kill</i> – finaliza um determinado processo, onde o usuário será questionado qual o número referente.
<Q>	<i>Quit</i> – encerra o programa.

Para sairmos desta tela, bastará apenas pressionarmos a tecla <Q>.

## SEGUNDO PLANO

### COLOCANDO EM SEGUNDO PLANO

Para colocarmos qualquer processo em segundo plano na linha de comando, bastará utilizarmos o caracter & (“e-comercial”) no final do comando invocado.

```
$ gimp &
[1] 1212
$ _
```

O programa será executado normalmente, porém com a prontidão da linha de comando. Mas se nós esquecermos este detalhe desejarmos colocá-lo em segundo plano?

### “CONTROL-ZÊ”

Conforme sabemos, a linha de comando ficará indisponível toda vez que necessitarmos executar outro programa (e conseqüentemente gerar um novo processo).

```
$ gimp
_
```

Para que possamos retornar à linha de comando, bastará pressionar <CTRL> + <Z>, do qual retornaremos imediatamente ao sinal de prontidão.

```
$ gimp
[1]+  Stopped                  gimp
$ _
```

Porém o aplicativo simplesmente ficará inoperante. Como fazer para reverter este quadro?

B



## BG

Para tornar o aplicativo novamente disponível, deveremos utilizar o comando *bg* – *BackGround* –, onde sua finalidade é colocar qualquer programa interrompido em segundo plano. Digitem na linha de comando...

```
$ bg  
[1]+  gimp &  
$ _
```

... para dar continuidade às atividades relacionadas ao aplicativo em execução (no exemplo mencionado, *Gimp*).

## JOBS

Apenas exibe os programas que se encontram em segundo plano:

```
$ jobs  
[1]+  Running                  gimp &  
$ _
```

## FG

Retorna qualquer processo parado ou em segundo plano para o primeiro.

```
$ fg 1  
gimp  
_
```

## EXCLUSÃO

### KILL

Elimina um processo existente no sistema através do *PID*.

Sintaxe:

```
$ kill [PARÂMETROS] [PROCESSO]
```

Exemplo: Caso desejarmos derrubar um determinado processo...

```
$ ps -p 1084  
  PID TTY          TIME CMD  
 1084 ttyS4      00:00:00 pppd  
$ _
```

... basta utilizar o comando *kill* e fornecer o número do processo do programa ou atividade que desejamos encerrar...

```
$ kill 1084
```

### KILLALL

Elimina um processo existente no sistema através do nome.

Sintaxe:

B

```
$ killall [PARÂMETROS] [PROCESSO]
```

Exemplo: de forma similar ao comando *kill*, caso desejarmos derrubar um determinado processo...

```
$ ps -p 1084
  PID TTY          TIME CMD
 1084 ttyS4      00:00:00 pppd
$ _
```

... basta utilizar o comando *killall* e fornecer o nome do processo do programa ou atividade que desejamos encerrar...

```
$ killall pppd
```

## SOBRE O DIRETÓRIO /PROC

Na *4a. Parte: Ajustes & Configurações -> Importância das estruturas em /etc e /proc ->* temos uma descrição abreviada dos subdiretórios numéricos existentes em sua estrutura. Cada um deste diretório e todos os arquivos virtuais gravados em seu interior referem-se *PID* dos processos em execução no sistema.

## CONCLUSÃO

Na grande maioria das circunstâncias – em especial para os usuários menos habituados – sequer nos importaremos com os processos em execução e suas atividades relacionadas. Face à isto, apenas dispomos por incluir os comandos e parâmetros mais essenciais desta atividade. Mas, se ainda assim quiserem obter mais conforto, experimentem utilizar os utilitários disponíveis dos principais ambientes gráficos existentes. &;-D

## VIII. GERENCIADORES DE INICIALIZAÇÃO

### INTRODUÇÃO

Quando se utiliza um computador com mais de um sistema operacional, será necessário definir por qual sistema utilizar. Quanto se trata de diferentes sistemas instalados em diferentes unidades de disco rígido, bastaria inverter no *setup* da *BIOS* a ordem de inicialização. Mas mesmo assim seria um desconforto muito grande, pois toda vez que for necessário inicializar o outro sistema, teríamos que editar as configurações do *setup* para inverter a ordem de inicialização. E para sistemas instalados em um mesmo disco rígido, divididos em diferentes partições, como fazer?

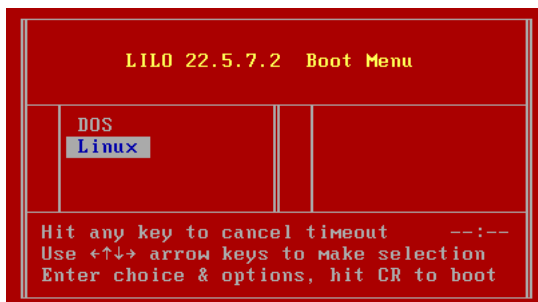
Para a solução destes problemas, existem os gerenciadores de inicialização, que são programas desenvolvidos para o gerenciamento da inicialização de máquinas que possuem mais de um sistema operacional instalados.

Neste capítulo iremos realizar um estudo aprofundado do gerenciador de inicialização padrão do *Slackware* – o *LILLO* –, onde iremos conhecer suas particularidades, métodos de instalação, configuração e ajustes adicionais necessários para o seu bom funcionamento.

### O LILO

✓ <<http://lilo.go.dyndns.org/>>.

O *LILLO* – *Linux LOader* – é um dos gerenciadores mais antigos existentes em sistemas *GNU/Linux*. Ele é o principal responsável pelo carregamento dos sistemas operacionais instalados, onde fornece ao usuário um simples menu de opções, bastando apenas ao usuário selecionar o sistema desejado. Ele fica residente no setor *MBR* do disco rígido.



*Slackware 10.0.*

Welcome to the LILLO Boot Loader!

Please enter the name of the partition you would like to boot at the prompt below. The choices are:



```
DOS    - DOS or Windows (FAT/FAT32 partition)
Linux  - Linux (Linux native partition)
```

O *LILLO* é adicionado ao sistema por padrão durante a instalação do *Slackware*, onde será necessário apenas definir o perfil de configuração e o local onde deverá ser gravado. Para obterem maiores informações, consultem a *3a. Parte: A Instalação -> Instalação do Slackware*.

## A CONFIGURAÇÃO

Os parâmetros de configuração do *LILLO* se encontram armazenados no arquivo */etc/lilo.conf*, o qual poderá ser alterado de acordo com as necessidades do usuário. Neste arquivo existem duas seções distintas para a configuração: a seção global e a seção de partições. Segue abaixo o conteúdo destas seções e as respectivas utilidades para cada parâmetro.

### SEÇÃO GLOBAL

A seção global é responsável pelo funcionamento geral do *LILLO*. É nela onde estarão todos os parâmetros pertinentes.

```
# LILLO configuration file
# generated by 'liloconfig'
#
# Start LILLO global section
boot = /dev/hda
message = /boot/boot_message.txt
prompt
timeout = 1200
# Override dangerous defaults that rewrite the partition table:
change-rules
    reset
-/-
```

Constam os seguintes parâmetros passíveis de edição:

***boot = /dev/[DISCO\_RÍGIDO]***

Informa qual unidade do disco rígido deverá inicializar.

***Message = /boot/[MENSAGEM\_TEXTO]***

Exibe um conjunto de instruções e informações básicas sobre os sistemas disponíveis (neste caso, *Windows* e *GNU/Linux*).

Observem que o parâmetro *message* aponta o arquivo-texto *boot\_message.txt*, situado no diretório */boot*. Este arquivo texto contém informações que serão exibidas na tela durante a inicialização do *LILLO*, podendo simplesmente ser editado de acordo com as necessidades do usuário, para melhor compreensão e ajustes.

```
Welcome to the LILLO Boot Loader!
```

```
Please enter the name of the partition you would like to boot
at the prompt below. The choices are:
```



```
DOS      - DOS or Windows (FAT/FAT32 partition)
Linux    - Linux (Linux native partition)
```

É sempre bom alterá-lo, colocando instruções básicas sobre os sistemas disponíveis ou personalizando-o de acordo com suas preferências. Segue um pequeno exemplo:

Bem vindo ao gerenciador de inicialização LILO!

Por favor selecione o nome da partição que você gostaria de inicializar neste prompt: Suas opções são:

```
DOS      - MS-DOS ou Windows (Partição FAT/FAT32)
Linux    - Slackware GNU/Linux
```

Após isto, deveremos atualizar as informações na *MBR*, digitando...

```
# lilo
Added DOS *
Added Linux
```

Para usuários iniciantes ou leigos, a tradução do texto para o português é sempre bem vinda...

### ***default***

Especifica o sistema à ser carregado por padrão (por ordem de disponibilidade), quando houver mais de um.

### ***prompt***

Indica a inicialização automática da linha de comando do *LILO*. Caso esta opção não esteja habilitada e esta linha de comando não esteja presente, bastará utilizar a sequência <CTRL> + <ALT> ou <CTRL> + <SHIFT> para ativá-lo.

### ***timeout = [TEMPO\_DE\_ESPERA]***

Tempo o qual o gerenciador permanece aguardando uma resposta via teclado para a seleção dos sistemas. Caso não sejam selecionados, o gerenciador automaticamente carrega o sistema padrão da máquina.

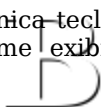
### ***password = [SENHA]***

Define uma senha para o acesso às opções de inicialização do sistema. Observe que esta senha não poderá ser criptografada, o que acarretará em sua visualização por qualquer usuário do sistema – à não ser que sejam redefinidos os atributos de permissão para o arquivo */etc/lilo.conf*. Para isto, basta apenas digitar...

```
# chmod 700 /etc/lilo.conf
```

### ***single-key***

Possibilita a utilização de uma única tecla para a inicialização do sistema, ao invés de digitar todo o nome exibido. Este parâmetro deverá ser



configurado junto com a opção *alias*, disponível na seção de partições.

## FRAMEBUFFER

Ainda dentro da seção global, existe um outro trecho para caso você não tenha definido a forma de exibição das fontes no modo texto durante a instalação do *Slackware* e desejar utilizar o recurso gráfico *framebuffer*.

Conforme será enfatizado durante a *3a. Parte: A Instalação -> Instalação do Slackware*, o *framebuffer* é um recurso especial em que o *kernel* acessa diretamente a memória de vídeo e conseqüentemente utilizando seus recursos, como a resolução gráfica. Mas como fazer para habilitar/desabilitar este recurso, além de alterar as resoluções presentes?

```
-/-  
# Normal VGA console  
vga = normal  
# VESA framebuffer console @ 1024x768x64k  
# vga=791  
# VESA framebuffer console @ 1024x768x32k  
# vga=790  
# VESA framebuffer console @ 1024x768x256  
# vga=773  
# VESA framebuffer console @ 800x600x64k  
# vga=788  
# VESA framebuffer console @ 800x600x32k  
# vga=787  
# VESA framebuffer console @ 800x600x256  
# vga=771  
# VESA framebuffer console @ 640x480x64k  
# vga=785  
# VESA framebuffer console @ 640x480x32k  
# vga=784  
# VESA framebuffer console @ 640x480x256  
# vga=769  
# End LILO global section  
-/-
```

Observe que a opção padrão utilizada pelo gerenciador de inicialização é *vga = normal*. Caso queira habilitar o *framebuffer*, bastará apenas selecionar a opção de resolução e profundidade de cor e desmarcar o comentário ao lado. Na próxima inicialização, este recurso estará em vigor.

## SEÇÃO DE PARTIÇÕES

A seção de partições contém descritas as opções para de inicialização dos sistemas operacionais já instalados no micro.

Em uma unidade de disco rígido, onde existem duas partições para o *Windows* e *GNU/Linux*, a estrutura da seção de partições seria a seguinte:

```
-/-  
# DOS bootable partition config begins  
other = /dev/hda1  
label = DOS  
table = /dev/hda1
```

```
# DOS bootable partition config ends
# Linux bootable partition config begins
image = /boot/vmlinuz
    root = /dev/hda6
    label = Linux
    read-only
# Linux bootable partition config ends
-//-
```

Observe que, por sua vez, esta seção subdivide-se em mais duas distintas: uma especial para a inicialização de outros sistemas (*DOS*), e outra para a inicialização de sistemas *GNU/Linux* (*Linux*).

Ademais, segue as definições de cada parâmetro desta seção:

### ***image = [IMAGEM\_COMPACTADA]***

Inicializa a imagem compactada do *kernel* utilizado. Este geralmente se encontra armazenado no diretório */boot*, porém você poderá utilizar outros locais conforme desejar. Por exemplo, sempre mantenho a seguinte opção:

```
image = /usr/src/linux/arch/i386/boot/bzImage
    root = /dev/hda6
    label = Experimental
    read-only
# Linux bootable partition config ends
```

É muito útil para experimentar novos *kernels*, sem ter que ficar remexendo no sistema toda vez para realizamos uma nova compilação.

### ***root = /dev/[PARTIÇÃO\_GNU/LINUX]***

Informa onde se encontra a partição de sistemas *GNU/Linux* instalada. Atendem-se para que diferentes sistemas (*kernels*) possam estar localizadas em uma mesma partição, como é recomendável ser feito no caso da compilação de um *kernel* experimental.

### ***label = [NOME\_DO\_SISTEMA]***

Exibe o nome do sistema instalado. Ao utilizarmos diferentes distribuições, deveremos atribuir sua nomenclaturas no *label* correspondente.

### ***read-only***

Pelo fato da necessidade de checagem do sistema de arquivos antes de inicializado, esta partição deverá ser montada antes como somente leitura para depois ser montada normalmente.

### ***other = /dev/[PARTIÇÃO\_COM\_OUTROS\_SISTEMAS]***

Indica a partição onde se encontram outros sistemas operacionais para serem inicializados.

### ***table = /dev/[DISCO\_RÍGIDO]***

Informa em qual disco rígido se encontra algum outro sistema operacional



instalado.

### ***alias = [CARACTER]***

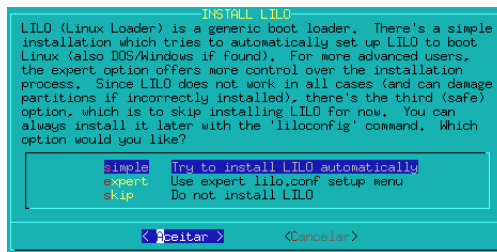
Utilizado em conjunto com a opção *single-key* da seção global, atribui um único caracter para selecionar o sistema operacional desejado (esta opção deverá estar dentro das definições dos sistemas listados, da mesma forma que *label*).

## **LILOCONFIG**

O *liloconfig* é um utilitário especialmente desenvolvido para definir as configurações básicas deste gerenciador de inicialização da máquina. Para executá-lo, basta digitar...

```
# liloconfig
```

... onde será apresentada uma tela com as instruções de como proceder passo-a-passo. Para alternarem entre as opções disponíveis no menu, utilizem a tecla <SETA\_ACIMA> e <SETA\_ABAIXO> para selecionar as opções do menu; <TAB> para posicionar o curso nas opções *Aceitar* ou *Cancelar*, onde <SETA\_ESQ.> e <SETA\_DIR.> alterna entre eles; e <ENTER> para definir as opções desejadas.



O *LILO* também é automaticamente instalado e definido durante a instalação do *Slackware*. Ao ser executado, deveremos definir qual o tipo de instalação. Seleccionem *simple* para uma instalação básica.

Estas e outras informações estão disponíveis na *3a. Parte: A instalação -> Instalando o Slackware*, na seção *Configure -> LILO*.

## **OPERAÇÕES MAIS FREQUENTES**

Existe algumas operações ocasionalmente necessárias para a boa manutenção da inicialização do sistema, que dentre as quais, segue:

### **SELECIONAR O SISTEMA GNU/LINUX COMO PADRÃO**

Após a inicialização do *LILO*, o sistema aguardará o usuário selecionar qual o sistema operacional à ser carregado, quando coexistirem o *Windows* e o *GNU/Linux*. Por padrão o *Windows* é carregado quando não é feita a





escolha pelo usuário, e o tempo de espera é consumido. Mas podemos alterar a opção padrão para que ela seja um sistema *GNU/Linux*. Para isto, acrescentem o parâmetro...

```
# LILO configuration file
# generated by 'liloconfig'
#
# Start LILO global section
append="hdd=ide-scsi"
boot = /dev/hda
message = /boot/boot_message.txt
prompt
default=[LABEL]
timeout = 1200
-/-
```

..., onde *[LABEL]* será o nome (rótulo) dado ao sistema descrito na seção de partições. O modo padrão é *Linux*. Em seguida, rodem o *lilo* novamente. Será exibida a seguinte mensagem.

```
# lilo
Added Windows
Added Linux *
# _
```

Significa que a operação foi realizada com sucesso.

## MUDAR A RESOLUÇÃO DO FRAMEBUFFER

Podemos definir a resolução de vídeo manualmente, editando diretamente o arquivo */etc/lilo.conf* na seção global. Vamos supor que desejássemos utilizar a resolução de *1024x768x32k*?

```
# VESA framebuffer console @ 1024x768x32k
# vga=790
```

Ou então *800x600x32k*:

```
# VESA framebuffer console @ 800x600x32k
# vga=787
```

Basta apenas remarcar (ou apagar) o padrão atual (*vga = normal*) e desmarcar a opção desejada (*vga = [DESEJADO]*). Lembre-se que sua placa de vídeo deverá suportar este recurso, e o *monitor*, a resolução desejada. Após isto, salve o arquivo e digite na linha de comando...

```
# lilo
```

... para gravar um nova definição do *LILO* no setor *MBR* do disco rígido com as alterações realizadas.

## ADICIONAR MAIS UMA ENTRADA NO LILO

Se fosse instalado mais um sistema operacional além do *Windows* e o *GNU/Linux* padrão, e o quisesse incluir no gerenciador, bastaria incluir as informações necessárias dentro desta seção. Vamos supor que você resolveu instalar uma outra versão de sistema *GNU/Linux* em um outro

disco rígido, de um micro qualquer que estava encostado no canto. As instruções necessárias seria estas:

```
image = /[IMAGEM_DO_KERNEL]
root = /dev/[DISCO_RÍGIDO]
label = [NOME_DO_SISTEMA]
read-only
```

Ou se fosse um outro sistema não *GNU/Linux*:

```
other = /dev/hdb1
label = [SISTEMA_OPERACIONAL_DESCONHECIDO]
table = /dev/hdb
```

Ao executar o *LILLO* para atualizar as alterações na *MBR*, será exibido o novo nome (*label*) do sistema operacional disponibilizado:

```
# lilo
Added DOS *
Added Linux
Added [NOVO_SISTEMA]
# _
```

## ADIÇÃO DE SENHA

Sabendo-se da possibilidade de ter acesso ao sistema com a simples utilização do comando *linux single* na linha de comando do *LILLO*, a única forma de proteger-se deste inconveniente é atribuir uma senha de acesso. Para isto, insira a seguinte linha no arquivo */etc/lilo.conf*:

```
-//-
# Start LILLO global section
boot = /dev/hda
message = /boot/boot_message.txt
password = [SENHA]
prompt
timeout = 1200
# Override dangerous defaults that rewrite the partition table:
-//-
```

Onde no lugar de [SENHA] deverá ser digitada a senha desejada. Lembre-se de que estes parâmetros deverão estar situados antes da linha *prompt*.

## PARÂMETROS ESPECIAIS DURANTE A INICIALIZAÇÃO

Você pode utilizar alguns parâmetros especiais na linha de comando.

```
boot: _
```

Quando se desejava utilizar qualquer um dos sistemas operacionais, bastava teclar <TAB> e digitar o nome exibido do sistema operacional referente. Porém, com a utilização dos parâmetros...

<i><b>Parâmetros especiais</b></i>	
<i>single user</i>	Em monousuário (sem senha), para a manutenção do sistema.

... serão aplicadas as definições descritas.



## ATUALIZANDO AS ALTERAÇÕES DESEJADAS

Para atualizar qualquer modificação realizada no arquivo */etc/lilo.conf*, basta apenas executarmos o *LILO* e verificar a saída de vídeo:

```
# lilo
Added DOS *
Added Linux
# _
```

Pronto! O *LILO* já atualizou as configurações desejadas e na próxima inicialização estarão habilitadas as alterações realizadas. Caso ocorra alguma anomalia, reeditem novamente o arquivo de configuração e corrijam as linhas alteradas.

## DISCO DE RECUPERAÇÃO COM O LILO.CONF

Outra necessidade eventual está na utilização de um disco de recuperação que suporte o *LILO*, caso este não possa selecionar o sistema à ser inicializado. Para isto, digitem na linha de comando...

```
# lilo -b /dev/[DEVICE_DISQUETE]
```

Segue um exemplo prático, utilizando a *1a.* unidade do sistema:

```
# lilo -b /dev/fd0
```

## PROBLEMAS MAIS FREQUENTES

*Ao invés do sistema inicializar, é exibido LI, LI- ou 01 01 01...*

Isto pode ocorrer com o posterior travamento do sistema devido possivelmente às seguintes circunstâncias:

- A não gravação e/ou atualização do *LILO* na *MBR*;
- Substituição da posição das unidades de disco rígido após a instalação – neste caso os *devices* dos discos rígidos não corresponderão às definições gravadas pelo *LILO*.
- Partição de inicialização após cilindro *1024* – isto ocorre nas versões anteriores à *21.4.3* do *LILO*. Em virtude da disponibilidade de versões atualizadas, não existem maiores preocupações quanto à esta ocorrência.
- Antivírus da *BIOS* ativado – Não somente nos sistemas *GNU/Linux*, como no próprio *Windows* a manutenção de um programa antivírus da *BIOS* acarreta diversos problemas, em especial durante a instalação de programas. É recomendável desativá-lo.

Estas são as principais causas do travamento na inicialização do *LILO*. Poderão existir quaisquer outras, mas as soluções poderão ser específicas, de acordo com a máquina utilizada. Para estas e outras circunstâncias, recomendamos elaborar durante ou logo após a instalação do *Slackware*



um disquete de inicialização.

### *O sistema não reconhece mais que 64 MB de memória...*

Isto ocorre apenas nas versões do *kernel* anteriores à 2.4. Editem o arquivo */etc/lilo.conf* e adicione na seção global a seguinte linha ao arquivo:

```
append="mem=[QUANTIDADE DE MEMÓRIA EM MB]M"
```

Salvem o arquivo e atualize o *LILO*...

```
# lilo
```

Basta reiniciar o sistema e as alterações entrarão em vigor.

### *Remoção das definições do LILO no setor MBR*

Basicamente existem duas formas de remover as definições do *LILO* com a utilização das linhas de comando:

<b>Sistema</b>	<b>Comando</b>
<i>MS-DOS</i>	C:\> FDISK /MBR
<i>Bash</i>	# /sbin/lilo -U

Geralmente a remoção do *LILO* na *MBR* se faz quando há necessidade da desinstalação dos sistemas *GNU/Linux* ou utilização de discos rígidos os quais possuem ou tiveram um sistema *GNU/Linux* instalado.

### *Recuperar a senha do superusuário*

Em algumas circunstâncias poderão ocorrer a perda (esquecimento, erro de digitação) da senha do superusuário para a administração do sistema. Caso isto ocorra, simplesmente digitem na linha de comando do *LILO*...

```
LIL0: linux single init=/dev/bash.
```

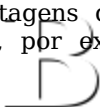
...ou apenas...

```
LIL0: linux single
```

O sistema será executado em modo monousuário, o qual com a utilização do comando *passwd*, poderemos alterar a senha original para um termo conhecido. Lembrem-se que este recurso somente funcionará caso não tenhamos definido a opção *password* na configuração do *LILO*. Neste caso deveremos ter em mãos a senha de acesso do *LILO* para que possamos iniciar o sistema em modo monousuário.

## **OBSERVAÇÕES FINAIS**

Apesar de simples e com uma aparência “*arcaica*” (como muitos dizem por aí), o *LILO* disponibiliza as principais funcionalidades para o gerenciamento de múltiplos sistemas operacionais do computador. Pode não ser tão flexível ou possuir maiores vantagens que a utilização de outros bons gerenciadores – como o *GRUB*, por exemplo – porém devido às suas



características únicas, seu uso e manutenção é simples e fácil. O gerenciador ainda pode ter sua tela de apresentação personalizável, porém este não é o escopo deste livro, o qual com ele nos comprometemos a fornecer informações indispensáveis para o seu bom funcionamento. Mas não se preocupe. Existem inúmeros artigos na *Internet* disponíveis para esta finalidade. &;-D

## CONCLUSÃO

Independente da escolha do gerenciador de inicialização, estejam certo de que temos à disposição excelentes programas para esta finalidade. Cada um possui características e particularidades inerentes dos quais serão fatores preponderantes de preferência aos usuários do sistema. O *Slackware* prima pelo uso do *LILO* pelo fato do mesmo apresentar maior simplicidade, mas nada que impeça a utilização do *GRUB*, uma excelente alternativa. &;-D

# IX. INICIALIZAÇÃO

---

## INTRODUÇÃO

Diferente do *MS-DOS* e *Windows* – os sistemas *GNU/Linux* lidam de forma diferente com o processo de inicialização do computador. Neste capítulo, iremos conhecer quais são e como funciona o processo de inicialização dos sistemas *GNU/Linux*, bem como suas características e particularidades, tendo um enfoque especial ao método de inicialização do *Slackware*.

## MÉTODO DE INICIALIZAÇÃO

Os métodos de inicialização padrão dos sistemas *Unix*, clones e variantes são respectivamente o *System V* e o estilo *BSD*.

### TIPOS DE MÉTODOS

#### SYSTEM V

O *System V (AT&T)* é o método mais utilizado pelas distribuições atuais. Consiste em utilizar dezenas de *scripts* para cada serviço à ser inicializado, todos armazenados em um diretório específico de acordo o nível de execução utilizado. Além disso, operar em vários modos existentes, todos numerados de 0 a 6.

#### ESTILO BSD

O estilo *BSD (Berkeley Software Distribution)* é atualmente adotado pelas distribuições *Slackware*, *Debian* e *SuSe*. Diferente do outro sistema de inicialização, o estilo *BSD* utiliza apenas alguns *scripts* são carregados durante o processo de inicialização, estes considerados mais rápidos e eficientes, além de uma maior simplicidade quando de sua manutenção. Além disso, opera em apenas 2 modos: o *single-user* (“S” – para manutenção) e o *multi-user* (“M” – uso para produção).

Os *scripts* de inicialização do *Slackware* obedece à este estilo, que por sua vez é simples e extremamente rápido, enquanto que as demais distribuições se baseiam na utilização do método de inicialização *System V*. Apesar disto, os níveis de execução são conforme o método *System V*.

## SCRIPTS DE INICIALIZAÇÃO

Os *scripts* de inicialização são arquivos de lote que armazenam as definições necessárias para a habilitação dos principais serviços necessários para o sistema, conforme especificado pelo seu nível de execução – o *runlevel*.



Quando o sistema inicializa, o *kernel* é carregado para a memória do sistema, após a inicialização dos dispositivos, este roda o *init*, o primeiro processo de execução do sistema – *PID 1*. Após ser carregado, o *kernel* executa o *runlevel* especificado pelo arquivo de configuração */etc/inittab*. Ainda neste mesmo arquivo de definições, especifica-se os *scripts* necessários para a execução do *runlevel* desejado.

## O DIRETÓRIO /ETC/RC.D/

Conforme dito anteriormente, todos estes *scripts* de inicialização são mantidos em */etc/rc.d/*. Para cada um serviço ou categoria de serviços, haverá um *script* específico para a sua habilitação, além de parâmetros para ajuste e configuração do mesmo que garantem o bom funcionamento nas máquinas os quais se encontram instalados.

```
# ls /etc/rc.d/
$ ls
rc.0@      rc.bind      rc.inet2*    rc.pcmcia    rc.sysvinit*
rc.4*      rc.cups      rc.inetd*    rc.rpc       rc.udev*
rc.6*      rc.dnsmasq   rc.ip_forward rc.samba     rc.wireless*
rc.K*      rc.font.new* rc.keymap*    rc.saslauthd
rc.wireless.conf
rc.M*      rc.gpm*      rc.local*    rc.scanluns* rc.yp*
rc.S*      rc.hotplug*  rc.modules@  rc.sendmail
rc.acpid*  rc.httpd     rc.modules-2.4.33.3* rc.serial
rc.alsa*   rc.inet1*    rc.mysqlld   rc.sshd
rc.ataalk  rc.inet1.conf rc.nfsd*      rc.syslog*
```

Subdividiremos por categoria e descreveremos breves comentários sobre os mesmos, para que possamos ter uma melhor compreensão.

## INICIALIZAÇÃO

Como o próprio nome diz, os *scripts* de inicialização são os responsáveis pela inicialização tanto da máquina em si quanto dos níveis de execução que são definidos pela sua configuração e até mesmo no ato de seu desligamento.

Inicialização	
<i>rc.S</i>	Inicialização ( <i>start</i> ) do <i>Slackware</i> .
<i>rc.K</i>	Responsável pelo nível de execução ( <i>runlevel</i> ) 1, necessário para a manutenção do sistema – <i>single user</i> .
<i>rc.M</i>	Utilizado nos demais níveis de execução, ou seja, para modo multi-usuário (2, 3 e 5).
<i>rc.4</i>	Necessário especialmente para carregar em modo gráfico (através dos gerenciadores <i>KDM</i> , <i>GDM</i> e <i>XDM</i> ).
<i>rc.0</i>	Atalho simbólico para <i>rc.6</i> . -



<b>Inicialização</b>	
<i>rc.6</i>	Reinicialização da máquina (liga) e encerramento (desliga).

Todos estes *scripts* são executados pelo *Bash* e podem ser editados manualmente, porém recomenda-se a realização prévia de cópias de segurança para nos resguardarmos de algum imprevisto qualquer.

## SUPORTE GERAL

Além destes *scripts*, existem vários outros de extrema importância, que são utilizados pelos *scripts* acima descritos, cada um com as mais variadas finalidades. Segue abaixo sua descrição:

<b>Suporte geral</b>	
<i>rc.acpid</i>	Gerenciador de recursos de energia do sistema.
<i>rc.alsa</i>	<i>Drivers ALSA (Advanced Linux Sound Architecture).</i>
<i>rc.cups</i>	Servidor de impressão <i>CUPS (Common Unix Printers Services)</i>
<i>rc.font.new</i>	Carregamento das fontes de console a serem utilizadas.
<i>rc.gpm</i>	Configuração e carregamento do <i>gpm</i> (suporte a <i>mouse</i> ).
<i>rc.keymap</i>	Ativação do mapa de teclado do sistema.
<i>rc.mysql</i>	Banco de dados <i>MySQL</i> .
<i>rc.pcmcia</i>	Suporte a dispositivos <i>PCMCIA</i> (utilizados em <i>notebooks</i> ).
<i>rc.hotplug</i>	Suporte ao subsistema de detecção de <i>hardware</i> .
<i>rc.scanluns</i>	Detecção de dispositivos <i>SCSI, USB</i> e <i>FireWire</i> .
<i>rc.serial</i>	Configuração das portas seriais da máquina.
<i>rc.syslog</i>	Gerador de <i>LOGs</i> do sistema.
<i>rc.udev</i>	Sistema dinâmico de configuração para suporte aos dispositivos do sistema ( <i>devices</i> ).
<i>rc.wireless</i>	Suporte à dispositivos <i>Wireless</i> .

## SUPORTE À REDE

Além destes, ainda existem outros *scripts* utilizados exclusivamente para o suporte à rede, segue:

<b>Suporte à rede</b>	
<i>rc.atalk</i>	Uso do protocolo <i>AppleTalk</i> .
<i>rc.bind</i>	Servidor <i>BIND</i> .





<b><i>Suporte à rede</i></b>	
<i>rc.dnsmasq</i>	Um pequeno servidor <i>DNS</i> .
<i>rc.httpd</i>	Servidor <i>WEB Apache</i> .
<i>rc.inetd</i>	Servidor <i>BSD</i> para <i>Internet</i> .
<i>rc.inet1</i> <i>rc.inet2</i>	Carregamento de vários serviços relacionados à redes como o <i>IP</i> , <i>Gateway</i> , <i>rpc.portmap</i> , <i>nfsd</i> , <i>lpd</i> , <i>smb</i> , etc.
<i>rc.ip_forward</i>	Definições para o uso de roteador ( <i>IPTABLES</i> ).
<i>rc.nfsd</i> <i>rc.rpc</i>	Suporte ao sist. de arquivos <i>NFS (Networks File System)</i> .
<i>rc.samba</i>	Protocolo <i>SMB (Samba)</i> .
<i>rc.saslauthd</i>	Sistema de autenticação <i>Cyrus SASL</i> .
<i>rc.sendmail</i>	Servidor <i>SMTP SendMail</i> (para envio de mensagens).
<i>rc.sshd</i>	Servidor <i>OpenSSH</i> (para conexão remota segura).
<i>rc.ypp</i>	Servidor <i>NIS (Networks Information Service)</i> .

## CONFIGURAÇÕES LOCAIS

O *script rc.local* foi designado para o carregamento das configurações particulares da máquina local, sendo o último à ser executado.

<b><i>Configurações locais</i></b>	
<i>rc.local</i>	Necessário para o carregamento de serviços, aplicações e parâmetros específicos da máquina local.

Observem que este *script* não possui nenhuma definição de comandos e opções para a inicialização de serviços e aplicações, ficando totalmente à cargo do administrador definir quais destes é que deverão ser carregados.

```
#!/bin/sh
#
# /etc/rc.d/rc.local: Local system initialization script.
#
# Put any local setup commands in here:
```

Aplicações como o *MySQL* (quando instalado manualmente) e o *HDParm* são bons exemplos de programas os quais necessitam ter comandos definidos para sua habilitação.

## CARREGAMENTO DE MÓDULOS

Para o carregamento de módulos, existe um *script* definido unicamente para esta função: o *rc.modules*.



<b>Carregamento de módulos</b>	
<i>rc.modules</i>	Carregamento de diversos módulos inerentes do sistema.

Este *script* possui aproximadamente 700 linhas, em que seu conteúdo por sua vez são subdivididos em diversas seções para facilitar a organização.

Uma observação interessante está no fato da inclusão de instruções para a habilitação de módulos particulares do sistema, como o suporte à uma placa de *fax-modem* (especialmente *softmodem*) ou qualquer outro periférico não suportado oficialmente pela distribuição. Alguns instruem em colocar estas definições em *rc.modules*; outros em *rc.local*. Independente do arquivo utilizado, inclua as instruções sempre no final do arquivo, para que possamos localizá-la de forma padronizada e com maiores facilidades.

## COMPATIBILIDADE

A grande maioria das distribuições *GNU/Linux* optam por utilizar o método de inicialização *System V*, que por sua vez as aplicações disponíveis para o sistema estão condicionadas à utilizar as definições gerais deste método.

<b>Compatibilidade</b>	
<i>rc.sysvinit</i>	Compatibilidade com o método <i>System V</i> .

O *Slackware*, por não utilizar este método de inicialização, e por necessitar obter compatibilidade com tais programas, definiu o *script rc.sysvinit*.

## A SEQUÊNCIA DE SCRIPTS NA INICIALIZAÇÃO

O primeiro *script* à ser executado é o *rc.S*, que roda apenas uma única vez durante a inicialização do sistema. Após o *rc.S*, vem o *rc.M*, responsável pela utilização do sistema no modo multi-usuário, desde que o nível de execução esteja pré-configurado em */etc/inittab* para isto.

O *script rc.K* somente é executado quando houver a necessidade de manutenção do sistema. Por entrar no modo monousuário, todos recursos multi-usuários são desabilitados, como também qualquer serviço (processos) para que possamos intervir no sistema.

## NÍVEIS DE EXECUÇÃO DO SLACKWARE

*Runlevels* – níveis de execução – são estágios de execução específicos que visam habilitar e/ou desabilitar um conjunto de serviços específicos para cada necessidade da máquina em uso. Por exemplo, em uma necessidade de uso comum, de manutenção, para propósitos específicos do sistema, enfim, existirá um nível de execução apropriado para cada um. Os níveis de execução poderão ser diferentes de acordo com a distribuição utilizada.



## OS NÍVEIS DE EXECUÇÃO

Segue abaixo os níveis de execução do *Slackware*:

<b>Níveis de execução</b>	
<i>Runlevel 0</i>	Atalho simbólico para <i>rc.6</i> .
<i>Runlevel 1</i>	Modo <i>single user</i> , necessário para a manutenção do sistema.
<i>Runlevel 2</i>	Sem uso.
<i>Runlevel 3</i>	Modo multi-usuário, autenticação em modo texto (console).
<i>Runlevel 4</i>	Modo multi-usuário, autenticação em modo gráfico.
<i>Runlevel 5</i>	Sem uso.
<i>Runlevel 6</i>	Reinicialização e desligamento do sistema – <i>System Halt</i> e <i>Reboot</i> , respectivamente.

### NÍVEL 1 – MANUTENÇÃO DO SISTEMA

O nível *1* somente é executado quando da necessidade de manutenção do sistema. Somente a partição raiz é montada para as intervenções necessárias, sendo este o principal motivo pelo qual não podemos definir uma partição especial para o diretório padrão do superusuário (*root*).

### NÍVEL 3 E 4 – MODO MULTI-USUÁRIO

Os níveis *3* e *4* são definidos para a utilização normal do sistema. É com eles que iniciamos o sistema para realizar as atividades do nosso dia-a-dia – ou finais de semana, dependendo da nossa disponibilidade... &;-D

A principal diferença está no carregamento da interface gráfica e seus respectivos gerenciadores de autenticação. O nível *3* inicializa o sistema em modo texto, onde o usuário digita seu apelido de autenticação e senha de acesso, para que acionem logo em seguida o ambiente gráfico, digitando para isto...

```
$ startx
```

Já o nível *4* inicializa o sistema em modo gráfico, disponibilizando um gerenciador de autenticação gráfico pré-definido que disponibiliza diversos recursos gráficos para um melhor conforto e facilidades aos usuários.

Apesar do nível *3* ser a opção menos confortável em termos de autenticação e seleção de ambiente gráfico, é em muitas circunstâncias a mais confortável quando ocorrem problemas que impedem a inicialização da interface gráfica. Já o nível *4* é indicado especialmente para usuários leigos, os quais não possuem conhecimentos técnicos para a manipulação do sistema em modo texto, como a seleção e inicialização do ambiente gráfico (na utilização dos gerenciadores *KDM* e *GDM*), desligamento do sistema, etc.



Somente o nível 4 possui um *script* de inicialização, o *rc.4*, que por sua vez define quais os gerenciadores de autenticação deverão ser rodados, onde deveremos comentar e/ou descomentar as linhas correspondentes aos gerenciadores que desejamos utilizar (ou não), ou ainda modificar a ordem de execução dos mesmos, para dar prioridade ao gerenciador desejado. Estas instruções e muitas outras se encontram com maiores detalhes na 6a. Parte: Ambientes Gráficos -> Operações e ajustes afins.

## NÍVEL 6 – REINICIALIZAÇÃO DO SISTEMA

À partir do momento em que reinicializamos ou desligamos o computador, automaticamente o sistema utiliza o nível de execução 6 para que todos os serviços e processos sejam finalizados antes do sistema ser reinicializado. Ao utilizarmos o comando *shutdown*, estaremos acionando este nível.

## DEMAIS NÍVEIS DE EXECUÇÃO

Os níveis 2 e 5 não são utilizados pelo sistema, enquanto que existem outros níveis não especificados, de 7 a 9, que são utilizados especificamente para o desenvolvimento de níveis de execução customizados de acordo com as necessidades da máquina local.

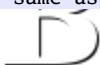
## DIFERENÇAS ENTRE O PADRÃO SLACKWARE E OS DEMAIS

Em outras distribuições, as definições dos níveis de execução podem variar. Por exemplo, a *Red Hat/Fedora* & variantes utilizam o nível 5, que é destinado ao modo multi-usuário com autenticação gráfica, ao invés do equivalente nível 4.

## /ETC/INITTAB

O arquivo de configuração */etc/inittab* define todos os parâmetros e definições gerais do método de inicialização sistema.

```
#
# inittab          This file describes how the INIT process should set up
#                  the system in a certain run-level.
#
# Version:        @(#)inittab          2.04    17/05/93      MvS
#                  2.10    02/10/95      PV
#                  3.00    02/06/1999     PV
#                  4.00    04/10/2002     PV
#
# Author:         Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
# Modified by:    Patrick J. Volkerding, <volkerdi@slackware.com>
#
# These are the default runlevels in Slackware:
# 0 = halt
# 1 = single user mode
# 2 = unused (but configured the same as runlevel 3)
```



```

# 3 = multiuser mode (default Slackware runlevel)
# 4 = X11 with KDM/GDM/XDM (session managers)
# 5 = unused (but configured the same as runlevel 3)
# 6 = reboot

# Default runlevel. (Do not set to 0 or 6)
id:3:initdefault:

# System initialization (runs when system boots).
si:S:sysinit:/etc/rc.d/rc.S

# Script to run when going single user (runlevel 1).
su:1S:wait:/etc/rc.d/rc.K

# Script to run when going multi user.
rc:2345:wait:/etc/rc.d/rc.M

# What to do at the "Three Finger Salute".
ca::ctrlaltdel:/sbin/shutdown -t5 -r now

# Runlevel 0 halts the system.
l0:0:wait:/etc/rc.d/rc.0

# Runlevel 6 reboots the system.
l6:6:wait:/etc/rc.d/rc.6

# What to do when power fails.
pf::powerfail:/sbin/genpowerfail start

# If power is back, cancel the running shutdown.
pg::powerokwait:/sbin/genpowerfail stop

# These are the standard console login getties in multiuser mode:
c1:1235:respawn:/sbin/agetty 38400 tty1 linux
c2:1235:respawn:/sbin/agetty 38400 tty2 linux
c3:1235:respawn:/sbin/agetty 38400 tty3 linux
c4:1235:respawn:/sbin/agetty 38400 tty4 linux
c5:1235:respawn:/sbin/agetty 38400 tty5 linux
c6:12345:respawn:/sbin/agetty 38400 tty6 linux

# Local serial lines:
#s1:12345:respawn:/sbin/agetty -L ttyS0 9600 vt100
#s2:12345:respawn:/sbin/agetty -L ttyS1 9600 vt100

# Dialup lines:
#d1:12345:respawn:/sbin/agetty -mt60 38400,19200,9600,2400,1200 ttyS0 vt100
#d2:12345:respawn:/sbin/agetty -mt60 38400,19200,9600,2400,1200 ttyS1 vt100

# Runlevel 4 used to be for an X window only system, until we discovered
# that it throws init into a loop that keeps your load avg at least 1 all
# the time. Thus, there is now one getty opened on tty6. Hopefully no one
# will notice. ;^)
# It might not be bad to have one text console anyway, in case something
# happens to X.
x1:4:wait:/etc/rc.d/rc.4

# End of /etc/inittab

```



Dentre as intervenções mais realizadas, está na mudança do nível de execução. Por padrão, o *Slackware* utiliza o nível 3, mas em virtude da facilidade de uso para os mais inexperientes, é recomendável utilizar o nível 4, o qual o sistema acessa um gerenciador de autenticação simples, intuitivo e fácil de usar. Novamente, consultem a *6a. Parte: Ambientes Gráficos -> Operações e ajustes afins*, para obterem maiores informações de como proceder para realizar esta e muitas outras intervenções.

## OPERAÇÕES E AJUSTES AFINS

### ATIVAR / DESATIVAR SCRIPTS DE INICIALIZAÇÃO

#### MÉTODO MANUAL

O método manual de ativar e/ou desativar os serviços na inicialização é feito através da mudança das permissões de execução (*flag x*) de seus respectivos *scripts*. Dentro do diretório */etc/rc.d*, basta executar...

```
# chmod a+x [SCRIPT]
```

... para ativarmos, ou...

```
# chmod a-x [SCRIPT]
```

... para desativarmos.

Poderemos também definir as demais permissões juntas da seguinte forma:

```
# chmod [PERMISSÃO] [SCRIPT]
```

Onde *[PERMISSÃO]* deve ser substituído por *flags* pré-definidas. Por exemplo, para ativarmos...

```
# chmod 755 rc.yo
```

... ou para desativarmos...

```
# chmod 644 rc.yo
```

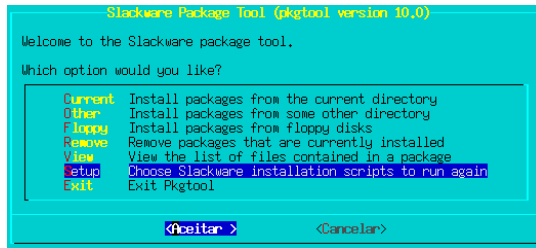
#### MÉTODO AUTOMATIZADO

O *Slackware* possui um atalho no *Pkgtool* para ativar e/ou desativar os *scripts* de inicialização do sistema. Basta carregarmos o utilitário...

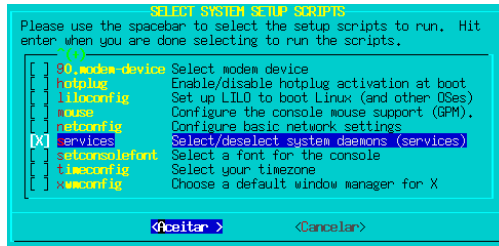
```
# pkgtool
```

... e acionar a opção *Setup* disponível na tela principal:

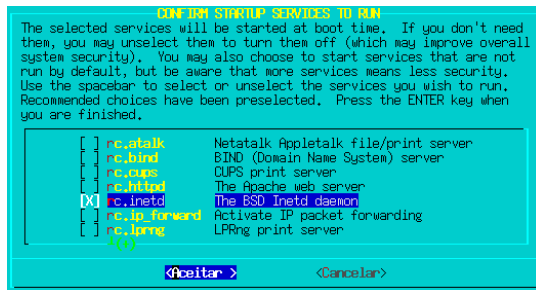




Em seguida, marcar a opção *services* e teclar <ENTER> em seguida...



Nesta 3a. tela, deveremos apenas definir quais os *scripts* que deverão ser ativados (marcando-os com a tecla <ESPAÇO>) e/ou desativados (desmarcando-os).



## “MELHORANDO” A COMPATIBILIDADE COM O SYSTEM V

Uma vez testamos o *VMWare*, um emulador de *PCs x86* que necessitava ter definidas a localização dos *scripts rc.0* a *rc.6* e seus respectivos diretórios. Este programa busca a destes *scripts* localização utilizando as referências técnicas do método *System V*, o que não encontrava no *Slackware*. Então para sanar esta deficiência, será necessário procedermos desta forma:

1. Criar os respectivos diretórios dos níveis de inicialização:

Todos os subdiretórios deverão ser criados conforme o método *System V*.

```
# mkdir /etc/rc.d/rc0.d
# mkdir /etc/rc.d/rc1.d
# mkdir /etc/rc.d/rc2.d
```



```
# mkdir /etc/rc.d/rc3.d
# mkdir /etc/rc.d/rc4.d
# mkdir /etc/rc.d/rc5.d
# mkdir /etc/rc.d/rc6.d
# _
```

2. Criar atalhos simbólicos para os scripts de inicialização:

Todos os atalhos simbólicos dentro destes diretórios deverão ser criados conforme a sua localização no método *System V* para representar os arquivos de configuração referentes aos níveis de inicialização.

```
# ln -s /etc/rc.d/rc.0 /etc/rc.d/rc0.d/rc.0
# ln -s /etc/rc.d/rc.1 /etc/rc.d/rc1.d/rc.1
# ln -s /etc/rc.d/rc.2 /etc/rc.d/rc2.d/rc.2
# ln -s /etc/rc.d/rc.3 /etc/rc.d/rc3.d/rc.3
# ln -s /etc/rc.d/rc.4 /etc/rc.d/rc4.d/rc.4
# ln -s /etc/rc.d/rc.5 /etc/rc.d/rc5.d/rc.5
# ln -s /etc/rc.d/rc.6 /etc/rc.d/rc6.d/rc.6
# _
```

Após iniciar a instalação do *VMWare*, seus utilitários de configuração encontrou a localização destes *scripts* sem maiores problemas.

## CONCLUSÃO

Conforme já dito diversas vezes, uma característica interessante do método de inicialização *BSD* está na velocidade e facilidade de customização. O fato de dispor apenas de alguns *scripts* com todas as seções pré-organizadas e comentadas não só facilitam as intervenções necessárias, como também agilizam na procura de parâmetros específicos. No método *System V*, teríamos o trabalho de identificar qual o *script* contém a configuração o qual desejamos editar, ao passo que no estilo *BSD* apenas bastaria localizar o *script* o qual contém a categoria do perfil de configuração e navegar nas seções comentadas. Poderemos também utilizar as ferramentas de busca dos editores de textos disponíveis para realizar a localização desejada. &;-D





# X. GERENCIAMENTO BÁSICO DE PROGRAMAS

---

## INTRODUÇÃO

Como qualquer outra distribuição *GNU/Linux*, o *Slackware* também possui seu sistema de gerenciamento de pacotes, além de diversos outros utilitários que auxiliam a administração dos programas utilizados.

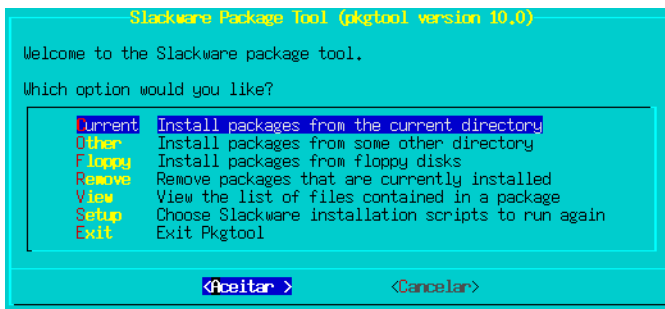
Neste capítulo conheceremos as principais ferramentas e instruções para as necessidades mais rotineiras, além de informações gerais sobre os demais processos pertinentes.

## FERRAMENTAS & MÉTODOS...

### SLACKWARE PACKAGE TOOLS

✓ <<http://www.slackware.org/>>.

O gerenciador de pacotes padrão do *Slackware* é o *Slackware Package Tools*, um conjunto de ferramentas além do menu interativo que visa facilitar ao máximo o gerenciamento de pacotes.



*Pkgtools.*

Para ter acesso ao gerenciador, digitem na linha de comando...

```
# pkgtools
```

... onde será apresentada a tela principal da ferramenta.

Além do aplicativo, existem diversos utilitários de linha de comando que complementam o gerenciador, os quais são: *installpkg*, *upgradepkg*, *removepkg*, *makepkg* e *explodepkg*. Por este capítulo tratar de apenas instruções básicas para o uso em *desktops*, apenas descreveremos os três primeiros, já que atendem satisfatoriamente suas necessidades.

## INSTALAÇÃO

Para instalar um pacote:



```
# installpkg [PACOTE].tgz
```

## ATUALIZAÇÃO

Para atualizar um pacote:

```
# upgradepkg [PACOTE].tgz
```

## REMOÇÃO

Para remover um pacote:

```
# removepkg [PACOTE].tgz
```

## RED HAT PACKAGE MANAGEMENT

✓ <<http://www.rpm.org/>>.

O *RPM* foi o primeiro gerenciador de pacotes desenvolvido para a instalação de programas pré-compilados.

```
RPM(8)                                Red Hat Linux                                RPM(8)

NAME
    rpm - RPM Package Manager

SYNOPSIS
    QUERYING AND VERIFYING PACKAGES:
    rpm {-q|--query} [select-options] [query-options]
    rpm {-V|--verify} [select-options] [verify-options]
    rpm --import PUBKEY ...
    rpm {-K|--checksig} [--nosignature] [--nodigest]
        PACKAGE_FILE ...

    INSTALLING, UPGRADING, AND REMOVING PACKAGES:
    rpm {-i|--install} [install-options] PACKAGE_FILE ...
    rpm {-U|--upgrade} [install-options] PACKAGE_FILE ...
    rpm {-F|--freshen} [install-options] PACKAGE_FILE ...

lines 1-36
```

*Manual Eletrônico do RPM.*

Criado pela *Red Hat*, é o gestor padrão da maioria das distribuições, além de ser um dos requerimentos estabelecidos pelo padrão *LSB*. Por isto o *Slackware* também suporta a instalação de pacotes no padrão *RPM*.

## INSTALAÇÃO

Para instalar um pacote:

```
# rpm -ivh [PACOTE].rpm --nodeps
```

## ATUALIZAÇÃO

Para atualizar um pacote:



```
# rpm -Uvh [PACOTE].rpm --nodeps
```

## REMOÇÃO

Para remover um pacote:

```
# rpm -e [PACOTE].rpm --force
```

## COMPILAÇÃO DO CÓDIGO-FONTE

Na necessidade de se instalar um pacote disponível somente em código-fonte, teremos que realizar manualmente a sua configuração, a compilação e por fim, a instalação. Apesar de não ser um processo tão fácil em comparação à instalação do pacote compilado, a maioria dos programas livres existentes utilizam os comandos básicos para esta finalidade.

Segue abaixo instruções básicas para a realização destas etapas. Para obterem detalhes adicionais, recomendamos a leitura da *5a. Parte: Gerenciamento de Programas -> Compilação do código-fonte*.

### A CONFIGURAÇÃO

Para que possamos compilar qualquer código-fonte, necessitaremos criar os *Makefiles*, que são arquivos que contém as instruções (parâmetros) necessárias para o sistema gerar os arquivos binários do programa desejado. Para isto, executem no diretório do pacote:

```
# ./configure --[OPÇÕES]
```

Lembrem-se de que em algumas circunstâncias – devido à natureza do pacote em questão – não será necessária a realização desta etapa, sendo apenas necessária as duas descritas logo à seguir.

### A COMPILAÇÃO

Após realizarmos a configuração, deveremos compilar o pacote com...

```
# make
```

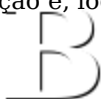
Baseado nas instruções geradas pelo *configure* e gravadas nos *Makefiles*, o comando *make* criará os arquivos binários necessários para compor o corpo do programa que desejamos instalar.

### A INSTALAÇÃO

Terminada a compilação do programa, digitem na linha de comando...

```
# make install
```

... para concluir a instalação do programa. Poderemos também omitir a instrução *make* do processo de compilação e lançar diretamente o *make install*. Assim será feita a compilação e, logo em seguida, a instalação.



## A DESINSTALAÇÃO

Para que seja desinstalado os binários de um arquivo gerados à partir do código-fonte, deveremos digitar...

```
# make uninstall
```

... lembrando que nem todos os programas possuem esta opção disponível.

## OUTROS UTILITÁRIOS

Além de suportar os dois processos de instalação de pacotes, o *Slackware* também possui utilitários que facilitam o gerenciamento destes, tais como o *rpm2tgz*, que converte os pacotes *RPM* para o formato nativo do *Slackware*, e o *script Checkinstall*, que possibilita empacotar os arquivos gerados pela compilação do código-fonte de aplicativos e utilitários.

## CONCLUSÃO

Em virtude da necessidade de estarem disponíveis na parte *Conhecimentos Gerais* todas as instruções básicas para a utilização das informações nas partes posteriores deste livro, damos apenas uma pequena introdução básica ao que se refere à instalação de programas nos sistemas *GNU/Linux*, para que os usuários tenham uma pequena base-técnica que os auxiliem à procederem com o uso das instruções disponíveis nas partes seguintes. Por isto, se quiserem se aprofundar mais, recomendamos que consultem a *5a. Parte – Gerenciamento de Programas*. &;-D



# XI. VARIÁVEIS DE SISTEMA

---

## INTRODUÇÃO

Quando trabalhamos com o *Windows*, em algumas situações necessitaremos lidar com algumas variáveis do sistema (como a *path*, por exemplo). Em algumas circunstâncias (especialmente nas versões *9x*), teremos que utilizar a linha de comando para realizar as intervenções.

Com estes mesmos conceitos, as variáveis dos sistemas *GNU/Linux* não só apresentam as mesmas funcionalidades conhecidas no sistema operacional habitual, como outras diversas, das quais iremos conhecer neste capítulo.

## AS VARIÁVEIS

### PATH / ROOTPATH

Para aqueles que migraram do *MS-DOS* acreditamos que estão bem familiarizados com a variável *PATH*, muito utilizada para habilitar seus comandos em qualquer ponto (diretório) do sistema. *PATH* – pronuncia-se “pá” – é apenas o caminho de procura para arquivos executáveis. A variável armazena na memória do sistema caminhos para a localização de comandos e executáveis evocados em qualquer ponto do sistema para a execução.

Observem os caminhos */bin*, */sbin*, */usr/bin* e */usr/local/bin*: estas são as localizações dos comandos utilizados pelos usuários, comandos do administrador e evocados pelo próprio sistema, binários de aplicações do sistema e binários de aplicações instaladas que não fazem parte dos pacotes disponíveis na distribuição (*local*) respectivamente.

```
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/usr/games:/opt/www/htdig/bin:/usr/lib/java/bin:/usr/lib/java/jre/bin:/opt/kde/bin:/usr/lib/qt-3.2.1/bin:/usr/share/texmf/bin:.
```

Isto quer dizer que, toda vez ao evocarmos um comando ou aplicação, o interpretador inicialmente irá buscar tais executáveis nos diretórios específica dos pela variável *\$PATH*. Caso não seja encontrado, será exibida a seguinte mensagem:

```
bash: [EXECUTÁVEL]: command not found
```

Se estas especificações não existissem, teríamos que especificar todo o caminho onde se encontra o executável:

```
$ /bin/free
$ /usr/bin/emacs
$ /usr/local/firefox/firefox
```

E para evocar uma aplicação instalada em um local exótico? Vejam o exemplo da *SDK Java*. Para que possamos acionar seus arquivos binários,

teríamos que digitar...

```
$ /usr/lib/java/bin/java [PARÂMETROS]
```

Para resolver este problema, bastaria apenas atualizar a variável *\$PATH* para conter o caminho do binário *java*, e assim digitar apenas...

```
$ java [PARÂMETROS]
```

Os sistemas *GNU/Linux* – neste caso, o *Slackware* – ao serem instalados possuem este caminho pré-configurados numa seção do arquivo */etc/profile*.

```
# Set the default system $PATH:
```

```
PATH="/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/usr/games"
```

Para que reconfiguremos toda vez que inicializar o sistema, termos que acrescentar o caminho aos já existentes. No caso do *Java*, seria...

```
# Set the default system $PATH:
```

```
PATH="/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/usr/games:/usr/lib/java/bin/"
```

Sem maiores detalhes, acreditamos que devam ter entendido a finalidade desta variável. Quanto à variável *ROOTPATH*, sem grandes mistérios, ela apenas define o caminho dos executáveis para o administrador do sistema.

## HOME

Exibe o diretório atual (raíz) onde se encontra o usuário autenticado.

```
$ echo $HOME
/home/darkstar
$ _
```

## OSTYPE

Exibe o sistema operacional (*kernel*) em uso.

```
$ echo $OSTYPE
linux-gnu
$ _
```

## SHELL

Exibe qual o interpretador de comandos usado na seção.

```
$ echo $SHELL
/bin/bash
$ _
```

## TERM

Já o *TERM* apenas indica o tipo de terminal utilizado no momento.

```
$ echo $TERM
xterm
$ _
```



## USER

Exibe a conta de usuário autenticada naquele instante.

```
$ echo $USER
darkstar
$ _
```

## COMANDOS PARA MANIPULAÇÃO

### ECHO

Apenas exibe o valor das variáveis em questão.

Sintaxe:

```
$ echo $[VARIÁVEL]
```

Exemplo:

```
$ echo $HOME
/home/darkstar
$ _
```

### SET

O comando *set* é o principal responsável pela atualização das variáveis do sistema.

Sintaxe:

```
$ set [VARIÁVEL] "[VALOR]"
```

Observem que o valor (conteúdo) da variável deverá se encontrar entre aspas duplas. Para habilitarmos um conteúdo para a variável *PATH*, deveremos utilizar...

```
$ set PATH "/usr/local/tools"
```

Somente utilizando este comando, serão apagadas todas as definições anteriores. Para mantê-las, utilizamos a variável *\$PATH*. Com isto, poderemos acrescentar este valor utilizando o comando...

```
$ set PATH $PATH;"/usr/local/tools";"[DEFINIÇÃO_2]";"[DEFINIÇÃO_3]"
```

... sem apagar as definições anteriores. Lembrem-se que estas definições somente se encontrarão presentes na sessão atual do terminal.

Um detalhe interessante é que, quando o comando é digitado em um terminal, são exibidos os valores correntes de todas as variáveis do sistema:

```
$ set
BASH=/bin/bash
BASH_VERSION=([0]="2" [1]="05b" [2]="0" [3]="1" [4]="release" [5]="i486-slackware-linux-gnu")
BASH_VERSION='2.05b.0(1)-release'
COLORTERM=
COLUMNS=80
```



```
CPLUS_INCLUDE_PATH=/usr/lib/qt-3.2.1/include:/usr/lib/qt-3.2.1/include
DIRSTACK=()
DISPLAY=:0.0
-//-
$ _
```

## EXPORT

Exporta valores de variáveis

Sintaxe:

```
$ export [VARIÁVEL]=[VALOR]
$ export [VARIÁVEL]="[VALOR COM ESPAÇOS]"
$ export [VARIÁVEL]='[VALOR COM ESPAÇOS]'
```

Exemplo:

```
$ export JAVA_HOME=/usr/java/bin
```

Da mesma forma que *set*, apagaremos as definições anteriores da variável. Se quisermos mantê-las, teremos que utilizar...

```
export JAVA_HOME=$JAVA_HOME:/usr/java/bin
```

Devemos lembrar que após estas definições, a variável atualizada estará presente apenas durante esta seção do comando. Da mesma forma que *set*, quando o comando é digitado em um terminal, são exibidos os valores correntes de todas as variáveis do sistema.

## INTERNACIONALIZAÇÃO

As variáveis de internacionalização são extremamente importantes para que possamos configurar determinados aplicativos e ambientes gráficos para a nossa língua nativa.

A forma clássica de ajustar uma variável é:

```
# export [VARIÁVEL]=[VALOR]
```

As principais variáveis são:

<b><i>Internacionalização</i></b>	
<i>LC_ALL</i>	Geral (todas as variáveis de uma só vez).
<i>LANG</i>	Idioma local.
<i>LC_MESSAGES</i>	Exibição de mensagens dos aplicativos.
<i>LC_TYPE</i>	<i>Layout</i> do teclado e comportamento de teclas especiais.

Basicamente os valores que deveremos atribuir são as especificações de língua. Para definir os valores do *Brasil*, deveremos utilizar a *flag pt\_BR*.

Segue um exemplo de como definir todas as variáveis de internacionalização do sistema, utilizando o comando...

```
# export LC_ALL=pt_BR
```



# ARQUIVOS DE CONFIGURAÇÃO

São inúmeros os arquivos que definem as variáveis do sistema, sejam globais, para cada aplicação ou um usuário específico. Aqui descreveremos apenas os principais.

## /ETC/PROFILE

Conforme comentado no capítulo *A importância das estruturas em /etc e /proc*, o arquivo */etc/profile* contém as definições globais utilizadas pelo sistema em geral. Qualquer parâmetro aqui definidos irão refletir em quaisquer seções e contas de acesso autenticados ao sistema.

```
# /etc/profile: This file contains system-wide defaults used by
# all Bourne (and related) shells.
-/-
```

Segue as principais seções deste arquivo:

### *Variáveis de ambiente*

```
# Set the values for some environment variables:
export MINICOM="-c on"
export MANPATH=/usr/local/man:/usr/man:/usr/X11R6/man
export HOSTNAME=`cat /etc/HOSTNAME`
export LESSOPEN="|lesspipe.sh %s"
export LESS="-M"

# If the user doesn't have a .inputrc, use the one in /etc.
if [ ! -r "$HOME/.inputrc" ]; then
    export INPUTRC=/etc/inputrc
fi
```

### *Definições de caminhos*

```
# Set the default system $PATH:
PATH="/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/usr/games"
```

### *Definições de caminhos (superusuário)*

```
# For root users, ensure that /usr/local/sbin, /usr/sbin, and /sbin are in
# the $PATH. Some means of connection don't add these by default (sshd comes
# to mind).
if [ "`id -u`" = "0" ]; then
    echo $PATH | grep /usr/local/sbin 1> /dev/null 2> /dev/null
    if [ ! $? = 0 ]; then
        PATH=/usr/local/sbin:/usr/sbin:/sbin:$PATH
    fi
fi
```

### *Terminal*

```
# I had problems using 'eval tset' instead of 'TERM=', but you might want to
# try it anyway. I think with the right /etc/termcap it would work great.
# eval `tset -sQ "$TERM"`
if [ "$TERM" = "" -o "$TERM" = "unknown" ]; then
    TERM=linux
fi
```



## *Personalizações ksh93*

```
# Set ksh93 visual editing mode:
if [ "$SHELL" = "/bin/ksh" ]; then
    VISUAL=emacs
#    VISUAL=gmacs
#    VISUAL=vi
fi
```

## *Sinal de prontidão*

```
# Set a default shell prompt:
#PS1='hostname:`pwd`# '
if [ "$SHELL" = "/bin/pdksh" ]; then
    PS1='! $ '
elif [ "$SHELL" = "/bin/ksh" ]; then
    PS1='! ${PWD/#$HOME/~}$ '
elif [ "$SHELL" = "/bin/zsh" ]; then
    PS1='%n@m:%~%# '
elif [ "$SHELL" = "/bin/ash" ]; then
    PS1='$ '
else
    PS1='\u@\h:\w\$ '
fi
PS2='> '
export PATH DISPLAY LESS TERM PS1 PS2
```

## *Permissões de acesso padrão*

```
# Default umask. A umask of 022 prevents new files from being created group
# and world writable.
umask 022
```

## *Cores do interpretador de comandos*

```
# Set up the LS_COLORS and LS_OPTIONS environment variables for color ls:
if [ "$SHELL" = "/bin/zsh" ]; then
    eval `dircolors -z`
elif [ "$SHELL" = "/bin/ash" ]; then
    eval `dircolors -s`
else
    eval `dircolors -b`
fi
```

## *Notificação de recebimento de correio*

```
# Notify user of incoming mail. This can be overridden in the user's
# local startup file (~/.bash.login or whatever, depending on the shell)
if [ -x /usr/bin/biff ]; then
    biff y
fi
```

## *Definições adicionais do profile*

```
# Append any additional sh scripts found in /etc/profile.d/:
for file in /etc/profile.d/*.sh ; do
    if [ -x $file ]; then
        . $file
    fi
done
```



done

## Definições de caminhos para usuários comuns

```
# For non-root users, add the current directory to the search path:
if [ ! "`id -u`" = "0" ]; then
    PATH="$PATH:."
fi
```

## O DIRETÓRIO /ETC/PROFILE.D/

O diretório `/etc/profile.d` possui definições de *scripts* adicionais para diversos ambientes gráficos e aplicações.

```
# ls -l /etc/profile.d/
total 88
-rwxr-xr-x 1 root root 164 2003-03-14 01:00 bsd-games-login-
fortune.csh*
-rwxr-xr-x 1 root root 141 2003-03-14 01:00 bsd-games-login-
fortune.sh*
-rwxr-xr-x 1 root root 32 2003-01-11 06:27 gtk+.csh*
-rwxr-xr-x 1 root root 43 2003-01-11 06:28 gtk+.sh*
-rwxr-xr-x 1 root root 102 2000-10-30 23:42 htdig.csh*
-rwxr-xr-x 1 root root 101 2000-10-30 23:43 htdig.sh*
-rwxr-xr-x 1 root root 146 2003-09-12 21:00 j2sdk.csh*
-rwxr-xr-x 1 root root 145 2003-09-12 21:00 j2sdk.sh*
-rwxr-xr-x 1 root root 176 2003-09-15 03:29 kde.csh*
-rwxr-xr-x 1 root root 85 2003-09-15 03:29 kde.sh*
-rwxr-xr-x 1 root root 227 2003-03-10 03:30 lang.csh*
-rwxr-xr-x 1 root root 225 2003-03-10 03:31 lang.sh*
-rwxr-xr-x 1 root root 51 2003-02-10 04:25 mc.csh*
-rwxr-xr-x 1 root root 45 2003-02-10 04:25 mc.sh*
-rwxr-xr-x 1 root root 31 2003-01-16 02:42 metacity.csh*
-rwxr-xr-x 1 root root 31 2003-01-16 02:41 metacity.sh*
-rwxr-xr-x 1 root root 443 2003-09-14 13:59 qt.csh*
-rwxr-xr-x 1 root root 396 2003-09-14 13:59 qt.sh*
-rwxr-xr-x 1 root root 50 2002-10-22 02:13 t1lib.csh*
-rwxr-xr-x 1 root root 63 2002-10-22 02:13 t1lib.sh*
-rwxr-xr-x 1 root root 134 2000-04-24 19:46 tetex.csh*
-rwxr-xr-x 1 root root 118 2000-04-24 19:46 tetex.sh*
# _
```

Cada arquivo possui definições de variáveis especiais para cada aplicação, de acordo com a nomenclatura dos mesmos. Com apenas uma visualização em seu conteúdo e, de acordo com as necessidades, deveremos realizar a edição direta destas para que as propriedades desejadas estejam presentes nas próximas seções.

## ~/BASHRC

Para àqueles usuários que preferem definir personalizações especiais (e por isto não podem alterar as definições globais `/etc/profile`) para as suas necessidades, eles poderão especificá-las em um arquivo chamado `.bashrc`. No *Slackware*, este não existe, sendo necessário criá-lo manualmente.

```
~$ mcedit .bashrc
```

A partir daí é só acrescentar as definições desejadas.

```
#!/bin/bash
```

```
[DEFINIÇÕES PERSONALIZADAS]
```

Lembrem-se que a definição `#!/bin/bash` é opcional.

## CONCLUSÃO

Existem diversas variáveis e opções de ajustes através da linha de comando disponíveis nos sistemas *GNU/Linux*. Definir e exemplificá-las aqui, além de trabalhoso, seria desnecessário, visto que a grande maioria dos simples usuários sequer têm necessidade delas, quanto mais vontade de intervir em ajustes desta categoria. Por isto optamos simplesmente por colocar apenas as variáveis de maior relevância à categoria destes usuários. &;-D



## XII. COMANDOS E FUNÇÕES GERAIS

---

### INTRODUÇÃO

Neste capítulo estarão listados todos os comandos gerais providos pelo sistema que até o momento não foram classificados em outras categorias.

### COMANDOS GERAIS

#### ADMINISTRAÇÃO DO SISTEMA

##### ARCH / UNAME

Exibe a arquitetura do processador presente no sistema. Sem grandes definições, basta digitar na linha de comando...

```
$ arch
i686
$ _
```

... que logo em seguida será exibida a arquitetura do processador utilizado. Existe o equivalente *uname*, que ao utilizar o parâmetro *-m*...

```
$ uname -m
i686
$ _
```

... exibe os mesmos resultados.

##### DMESG

Mostra as mensagens da última inicialização do *kernel*, além de outras informações geradas ao longo das operações realizadas no sistema.

```
$ dmesg
Linux version 2.4.22 (root@midas) (gcc version 3.2.3) #6 Tue Sep 2 17:43:01
PDT 2003
BIOS-provided physical RAM map:
 BIOS-e820: 0000000000000000 - 00000000000009fc00 (usable)
 BIOS-e820: 00000000000009fc00 - 000000000000a0000 (reserved)
 BIOS-e820: 000000000000f0000 - 00000000000100000 (reserved)
 BIOS-e820: 00000000000100000 - 000000000007fec000 (usable)
 BIOS-e820: 000000000007fec000 - 000000000007fef000 (ACPI data)
 BIOS-e820: 000000000007fef000 - 000000000007fff000 (reserved)
 BIOS-e820: 000000000007fff000 - 000000000008000000 (ACPI NVS)
 BIOS-e820: 000000000008000000 - 00000000000100000000 (reserved)
127MB LOWMEM available.
On node 0 totalpages: 32748
zone(0): 4096 pages.
zone(1): 28652 pages.
zone(2): 0 pages.
Kernel command line: BOOT_IMAGE=Linux ro root=306
Initializing CPU#0
```

```
Detected 799.779 MHz processor.  
Console: colour VGA+ 80x25  
Calibrating delay loop... 1595.80 BogoMIPS  
Memory: 126368k/130992k available (1813k kernel code, 4236k reserved, 614k  
data, 116k init, 0k highmem)  
-//-
```

## HALT

Desliga o sistema.

```
# halt
```

Neste caso o módulo *apm* deverá estar previamente carregado.

## SHUTDOWN

Reinicializa o sistema após finalizadas as tarefas em execução.

Sintaxe:

```
# shutdown [PARÂMETROS] [HORÁRIO]
```

Onde:

<b><i>shutdown</i></b>	
<b><i>-h</i></b>	Desliga o sistema.
<b><i>-r</i></b>	Reinicializa o sistema.
<b><i>[HORÁRIO]</i></b>	Definição de minutos.

Para se ter uma noção exata de sua utilidade, vejam os exemplos abaixo:

<b><i>shutdown</i></b>	
<b># shutdown -h 30</b>	Desliga o sistema após <i>30</i> minutos de sua execução.
<b># shutdown -r now</b>	Desliga o sistema imediatamente ( <i>now</i> = agora).

## AJUDA

Uma das características interessantes dos sistemas *GNU/Linux* está na forte documentação eletrônica dos utilitários e comandos disponíveis, onde uma simples consulta resolve a maioria das dúvidas existentes.

## MAN

Habilita as páginas do manual eletrônico de cada comando, aplicativo ou utilitário disponível.

Sintaxe:

```
$ man [COMANDO/APLICATIVO/UTILITÁRIO]
```

Ao digitarem *man* juntamente com os caracteres do comando +



<ENTER>...

```
$ man cdrecord
```

... em poucos segundos seu respectivo manual estará disponível.

```
CDRECORD(1)                Schily's USER COMMANDS                CDRECORD(1)
```

#### NAME

cdrecord - record audio or data Compact Discs from a master

#### SYNOPSIS

cdrecord [ general options ] dev=device [ track options ] track1...trackn

#### DESCRIPTION

Cdrecord is used to record data or audio Compact Discs on an Orange Book CD-Recorder.

The device refers to scsibus/target/lun of the CD-Recorder. Communication on SunOS is done with the SCSI general driver scg. Other operating systems are using a library simulation of this driver. Possible syntax is: dev= scsibus,target,lun or dev= target,lun. In the latter case, the CD-Recorder has to be connected to the default SCSI bus of the machine. Scsibus, target and lun are integer numbers. Some operating systems or SCSI transport implementations may require to specify a filename in addition. In this case the correct syntax for the device is:

lines 1-27

--/--

#### INFO

Outra excelente ferramenta à ser utilizada para obtermos ajuda e informações sobre um determinado comando.

Sintaxe:

```
$ man [COMANDO]
```

Exemplo:

```
$ man cdrecord
```

Em poucos instantes...

```
File: *manpages*, Node: cdrecord, Up: (dir)
```

```
CDRECORD(1)                Schily's USER COMMANDS
```

```
CDRECORD(1)
```

#### NAME

cdrecord - record audio or data Compact Disks or Digital Versatile Disks from a master

#### SYNOPSIS

cdrecord [ general options ] dev=device [ track options ] track1...trackn



## DESCRIPTION

Cdrecord is used to record data or audio Compact Discs on an Orange Book CD-Recorder or to write DVD media on a DVD-Recorder.

The device refers to scsibus/target/lun of the CD/DVD-Recorder. Communication on SunOS is done with the SCSI general driver scg. Other operating systems are using a library simulation of this driver. Possible syntax is: dev= scsibus,target,lun or dev= target,lun. In the latter case, the CD/DVD-Recorder has to be connected to the default SCSI bus of the machine. Scsibus, target and lun are integer numbers.

-----Info: (\*manpages\*)cdrecord, 1663 lines --

Top-----

Welcome to Info version 4.8. Type ? for help, m for menu item.

## --HELP

Diferente das páginas de manual, o parâmetro *--help* se encontra disponível na grande maioria dos comandos (e executáveis de programas), fornecendo um resumo das instruções mais necessárias sobre o comando em questão.

\$ ps --help

```
***** simple selection *****          ***** selection by list *****
-A all processes                        -C by command name
-N negate selection                    -G by real group ID (supports names)
-a all w/ tty except session leaders  -U by real user ID (supports names)
-d all except session leaders         -g by session leader OR by group name
-e all processes                      -p by process ID
T all processes on this terminal       -s processes in the sessions given
a all w/ tty, including other users    -t by tty
g all, even group leaders!             -u by effective user ID (supports
names)
r only running processes              U processes for specified users
x processes w/o controlling ttys      t by tty
***** output format *****            ***** long options *****
-o,o user-defined -f full              --Group --User --pid --cols
-j,j job control  s signal             --group --user --sid --rows
-O,0 preloaded -o v virtual memory    --cumulative --format --deselect
-l,l long         u user-oriented      --sort --tty --forest --version
                  X registers          --heading --no-heading
***** misc options *****
-V,V show version      L list format codes  f ASCII art forest
-m,m show threads      S children in sum   -y change -l format
-n,N set namelist file c true command name n numeric WCHAN,UID
-w,w wide output       e show environment  -H process heirarchy
$ -
```

Note a quantidade de definições exibidas com poucos caracteres. Se houver a necessidade de instruções mais detalhadas, utilizem o comando *man...*

\$ man ps

..., onde será exibido...

```
PS(1)                                Linux User's Manual                                PS(1)
,
NAME
```



```
ps - report process status
```

#### SYNOPSIS

```
ps [options]
```

#### DESCRIPTION

ps(1) gives a snapshot of the current processes. If you want a repetitive update of this status, use top.

#### COMMAND-LINE OPTIONS

This version of ps accepts several kinds of options.

Unix98 options may be grouped and must be preceeded by a dash.

BSD options may be grouped and must not be used with a dash.

GNU long options are preceeded by two dashes.

Options of different types may be freely mixed.

Set the I\_WANT\_A\_BROKEN\_PS environment variable to force

lines 1-28

-///-

## DIVERSOS

### CLEAR

Sem grandes mistérios, este comando possui a mesma função que o comando *CLS* do *MS-DOS*. Digitem...

```
$ clear
```

... e no instante seguinte a tela estará limpa.

### CAL

Exibe um simples calendário contendo a data e mês corrente.

```
$ cal
```

```
    setembro 2003
```

```
Do Se Te Qu Qu Se Sá
    1  2  3  4  5  6
  7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
```

```
$ _
```

Por padrão, o calendário exibe apenas o mês corrente, mas caso desejarmos ver os meses anterior e posterior, deveremos utilizar o parâmetro -3.

```
$ cal -3
```

```
    janeiro 2005
```

```
    fevereiro 2005
```

```
    março 2005
```

```
Do Se Te Qu Qu Se Sá Do Se Te Qu Qu Se Sá Do Se Te Qu Qu Se Sá
    1                1  2  3  4  5                1  2  3  4  5
  2  3  4  5  6  7  8  6  7  8  9 10 11 12  6  7  8  9 10 11 12
  9 10 11 12 13 14 15 13 14 15 16 17 18 19 13 14 15 16 17 18 19
```



```
16 17 18 19 20 21 22 20 21 22 23 24 25 26 20 21 22 23 24 25 26
23 24 25 26 27 28 29 27 28 27 28 29 30 31
30 31
$ _
```

## FREE

Exibe um relatório sobre a memória *RAM* e *SWAP* do sistema.

Sintaxe:

```
$ free [PARÂMETROS]
```

Onde:

<i>free</i>	
<i>-b / -k / -m / -g</i>	Exibe os valores em <i>bytes / KB / MB / GB</i> .
<i>-l</i>	Faz um comparativo entre as memórias alta e baixa.
<i>-t</i>	Exibe os valores totais (memória <i>RAM</i> + <i>SWAP</i> ).

Existem outros parâmetros extras, mas para as nossas necessidades básicas, somente precisaremos utilizar...

```
$ free
      total        used        free      shared    buffers     cached
Mem:    256144      250920         5224          0       21580     120060
-/+ buffers/cache:    109280      146864
Swap:    530104       12936      517168
$ _
```

## CONCLUSÃO

Existem uma infinidade de comandos e funções para a utilização, cada um com um propósito específico. Muitos (e provavelmente até a grande maioria) não serão necessários para nós, simples usuários que apenas utiliza um computador para as atividades mais corriqueiras. Mas para os administradores de sistemas e aqueles desejosos por obterem aprendizado, esta é uma oportunidade única para conhecerem à fundo o sistema e seus recursos. Para este propósito, recomendamos realizar consultas mais apuradas em literaturas especializadas. &;-D



## ENCERRAMENTO

---

Por mais diferentes que sejam as distribuições, a grande maioria dos comandos descritos nos capítulos anteriores (à salvo àqueles específicos para o *Slackware*) são aplicáveis na grande maioria dos sistemas *GNU/Linux*. O seu aprendizado, tanto utilizando esta quanto qualquer outra distribuição, será de grande valia para quaisquer intervenções que desejam realiza. Por isso, não se preocupem em pensar que o aprendizado aqui obtido será perdido em caso de mudança de distribuição preferida.

Gostaríamos mais uma vez de enfatizar que abordamos nesta parte apenas os comandos, parâmetros e funções consideradas indispensáveis aos usuários *desktops*, em especial de nível intermediário, mesmo apesar de constar algumas instruções para o nível avançado. Nos sistemas *GNU/Linux*, existe uma infinidade de ferramentas para as mais variadas finalidades, além de diversos graus de complexidade, onde muitas destas somente serão aplicáveis à tarefas específicas que não se enquadram nas necessidades destes usuários.

Caso queiram obter maiores confortos e facilidades, durante este processo poderemos utilizar largamente os gerenciadores de arquivos *Konqueror* ou *Nautilus*, além de diversos outros utilitários disponíveis na interfaces gráficas. Deveremos apenas configurá-los de acordo com nossas preferências. Poderemos navegar na estrutura de diretórios e utilizar seus vastos recursos com um simples clique de botão do *mouse* sobre o arquivo, diretório ou qualquer outro ponto configurável, além de acessar as ferramentas agregadas ao utilitário.

Mas se ainda assim os leitores sentirem a necessidade de fazer alguma recomendação como, incluir instruções sobre o comando *X*, detalhar os parâmetros e funcionalidades do utilitário *Y*, tecer comentários e observações sobre o processo *Z*, e até mesmo remover alguns termos e definições que não sejam tão necessários, suas palavras serão muito bem vindas, pois conforme fora (e ainda será) dito em diversas circunstâncias, o proposto deste livro é o fornecimento de instruções de forma simples, clara e objetiva para o fácil aprendizado e aplicação. Estaremos profundamente gratos por quaisquer opiniões que visem prover melhorias nestes aspectos.

Enfim, consciente dos conhecimentos técnicos necessárias para obter um bom aproveitamento das instruções contidas neste livro, iremos agora para a próxima parte – *A Instalação* – onde conheceremos todos os aspectos necessários para o bom êxito na realização desta atividade e intervenções relacionadas. Agora a aventura começa! &;-D

