Rubens Queiroz de Almeida

Linux: Dicas & Truques

Conectiva S.A.
http://www.conectiva.com.br

Curitiba - Paraná - Brasil

Catalogação na Fonte do
Departamento Nacional do Livro

A4471
Almeida, Rubens Queiroz.
Linux: Dicas & Truques / Rubens Queiros Almeida. - Curitiba: Conectiva S.A., 2000.
...p.; il.; cm.

ISBN 85-87118-13-7

1. Linux (Sistema operacional de computador).
I. Título

CDD-005.43

É permitido:

Copiar, distribuir, exibir ou criar obras derivadas sob as seguintes condições:

- **Atribuição**: Deverá ser dado ao autor original o devido crédito por seu trabalho.
- **Uso Não Comercial**: Este trabalho não pode ser usado para fins comerciais
- **Compartilhamento**: Obras resultantes da alteração, transformação ou derivadas deste trabalho deverão ser distribuídas sob uma licença idêntica a esta.

Em caso de reuso ou distribuição, os recipientes da obra deverão estar cientes das condições de licenciamento.

Qualquer destas condições poderá ser alterada mediante a obtenção de permissão do detentor do direito autoral.

Livros Publicados pelo Autor sobre Aprendizado da Língua Inglesa

A língua inglesa pode ser aprendida com diferentes propósitos e abordagens, entretanto poucos conhecem esse fato. O domínio completo da língua inglesa, que requer o desenvolvimento das habilidades de audição, fala, escrita e leitura, é um processo demorado, entre seis e oito anos de estudos e dedicação. Já a leitura, podemos dominar em um prazo consideravelmente mais curto, entre seis meses e um ano, dependendo de nosso interesse e motivação. No Brasil, em nossas atividades diárias, raramente necessitamos nos expressar fluentemente na língua inglesa. O que precisamos, na maioria das vezes, é compreender textos em inglês, seja para obter informações na Internet, compreender literatura técnica especializada ou desempenhar outras funções rotineiras, tais como a leitura de correspondências.

Este livro preenche uma lacuna importante no ensino da língua inglesa. Apresenta a lista das palavras mais usadas desse idioma, com exemplos de utilização, permitindo conhecer o vocabulário básico da língua inglesa em pouco tempo. Apresenta também uma metodologia de aprendizagem para utilizar esta ferramenta de acesso à informação de vital importância, que é a língua inglesa.

http://novateceditora.com.br/livros/linguainglesa2/

Este livro é uma coletânea de textos, citações e histórias em inglês, com o vocabulário comentado. Tem como objetivo facilitar o desenvolvimento de seu vocabulário da língua inglesa de uma maneira divertida e agradável. A seleção dos textos foi feita considerando-se, antes de tudo, a capacidade que eles têm de entreter e instruir simultaneamente.

Dedicando diariamente dez minutos à leitura deste livro, você poderá comprovar um aumento significativo na sua capacidade de ler e entender textos em inglês, ao mesmo tempo em que se diverte. Com um vocabulário de aproximadamente **5.000** palavras, esta publicação é o complemento ideal ao livro "As Palavras Mais Comuns da Língua Inglesa", também publicado pela Novatec Editora.

http://novateceditora.com.br/livros/readinenglish/

Sumário

1 Introdução **13**

1.1 Comentários do Editor 16

1.2 Conteúdo do CDROM 17

2 Por que usar software livre? **19**

3 Software Livre: Bom, bonito e barato **33**

4 Introdução ao DNS **41**

4.1 DNS - Root Name Servers 46

4.2 Endereços IP 48

5 Um Breve Histórico da Internet **51**

6 Correio Eletrônico: Uso da Auto-Resposta **55**

7 A Reconstrução da Torre de Babel **59**

8 Netscape **69**

8.1 Configuração de Aplicações Auxiliares (Helpers) 69

8.2 Opções da linha de comando para o programa *Netscape* . . 75

9 Linux / NT – Comparação de Desempenho 79

10 Correio Eletrônico 83

10.1 Sendmail 83

10.1.1 Redirecionamento de mensagens para o diretório pessoal dos usuários 83

10.1.2 Comandos expn e vrfy 86

10.2 Processamento Seletivo da Fila 87

10.2.1 mailcheck.sh 88

10.2.2 BCC – Blind Carbon Copy 92

10.3 Interação Sendmail x DNS 93

10.3.1 User Database 96

10.3.2 Relays 99

10.3.3 Autorização para Relay 100

10.3.4 Modo de Operação queueonly 102

10.3.5 Envio de mensagens diretamente com Sendmail . 103

10.3.6 Verificação de endereços eletrônicos 105

10.3.7 Manuseio de Mensagens 106

10.3.8 Determinando a versão do sendmail 107

10.4 Reenvio de Pastas de Correio Eletrônico 108

10.5 Envio de Mensagens no Formato HTML 108

10.6 Pine 109

10.7 Postagem Cruzada de Mensagens 109

11 Integração Windows e Linux **111**

11.1 Como nasceu o *Samba* 111

11.2 Histórico dos Protocolos 113

11.3 Descrição do Pacote *Samba* 115

11.4 Instalação 117

11.5 Configuração 120

11.6 Teste da Sintaxe do Arquivo de Configuração 124

11.7 Conclusão 126

11.8 Referências 126

12 vi – Visual Editor **127**

12.1 Introdução 127

12.2 Substituição de Caracteres 128

12.3 Substituição de Caracteres de Controle 129

12.4 Opção ignorecase 130

12.5 Inclusão de Resultados de Comandos do Sistema 131

12.6 Substituição Global Interativa 132

12.7 Deleção de Linhas em Branco 132

12.8 Inversão da ordem das linhas em um arquivo 133

12.9 Substituição utilizando metacaracteres 133

12.10 Gravação Seletiva 134

12.11 Deleção de Colunas 134

12.12 Inserção de Linhas 135

12.13 Diretório Temporário Alternativo 136

12.14 Posicionamento Automático 137
12.15 Abreviações 137
12.16 Inserção e Quebra de Linhas 138
12.17 Criação de Abreviações para Edição de Arquivos HTML . 139
12.18 Mapeamento de Teclas 141
12.19 Substituições 142
12.20 Desfazendo Alterações 143
12.21 Releitura de Arquivos 143
12.22 Substituições 144
12.23 Substituições em Intervalos de Linhas 144
12.24 Opções de Gravação 145
12.25 Remoção de Espaços em Branco 146
12.26 Busca e Substituição 147
12.27 Deleção de caracteres 147
12.28 Edição de Múltiplos Arquivos 148
12.29 Troca de Caixa 149
12.30 Opções Magic/Nomagic 150
12.31 dos2unix 151
12.32 Aprendendo Mais 151

13 Comandos do Sistema **153**

13.1 Aliases 153
13.2 unalias 155
13.3 Alien: Conversor de Formatos 156

13.4 apropos 157
13.5 Arquivos – Tipos 158
13.6 awk 160
13.6.1 Impressão de Campos Selecionados 160
13.6.2 A Variável NF – Number of Fields 161
13.7 Manual AWK 162
13.8 bzip2 162
13.9 Backup 162
13.9.1 Recomendações Gerais 162
13.9.2 Estratégias 164
13.9.3 Como desenvolver uma estratégia de backups . . 165
13.9.4 Amanda 166
13.9.5 Backups via e-mail 167
13.9.6 Dump em um arquivo remoto 169
13.10 basename/dirname 170
13.11 bash – Histórico de Comandos 170
13.12 Boot – Mensagens 171
13.13 tar 172
13.13.1 Cópia de Arquivos com tar 172
13.13.2 Identificação de backups 173
13.14 Backup em arquivos 174
13.15 cd – Change Directory 175
13.15.1 Retornar ao diretório anterior 175

13.16 cp – copy 176
13.16.1 Cópia rápida de arquivos 176
13.16.2 Redirecionamento da saída padrão em cshell . . . 176
13.17 cron 176
13.18 cat – Concatenate 178
13.18.1 Numeração de linhas 178
13.18.2 Eliminando linhas em branco 178
13.19 csplit – Content Split 179
13.20 date 182
13.21 daytime 183
13.22 Dutos 184
13.23 echo 185
13.24 Encadeamento de comandos 186
13.25 Endereçamento IP – Classes Reservadas 186
13.26 find 188
13.26.1 Localizando Arquivos sem Dono 188
13.26.2 Procurando arquivos no sistema 189
13.26.3 Cópia de uma Árvore de Diretórios 190
13.27 FTP através do Browser Netscape 190
13.28 ftp 191
13.28.1 Automatização de Sessões FTP 191
13.28.2 Transferência de Múltiplos Arquivos 193
13.28.3 Manipulação direta de arquivos 193

13.28.4 Transferência de Diretórios 194
13.29 Gerenciamento de Espaço em Disco 195
13.30 Grafia do nome de computadores 196
13.31 grep 197
13.31.1 Alguns recursos úteis do comando grep 197
13.32 gzip 197
13.33 Recuperação da Senha de Root 198
13.34 Como desmontar sistemas de arquivos em uso 201
13.35 /etc/inetd.conf 202
13.36 logbook.sh – Registro de Atividades 203
13.37 ls 207
13.37.1 Opções de Uso 207
13.37.2 Listagem de Permissões de Diretórios 210
13.37.3 Listagem Ordenada da Esquerda para a Direita . 210
13.37.4 Listagem Seletiva de Arquivos 213
13.37.5 Listando apenas os diretórios 213
13.38 man 214
13.39 Memória – Informações 214
13.40 mkdir 216
13.40.1 Criação de uma árvore completa de diretórios . . 216
13.41 MANPATH – Variável de Ambiente 216
13.42 MindTerm 217
13.43 more 218

13.44 Mudanças remotas em sistemas 218
13.45 Nessus - Ferramenta de Segurança para Unix 219
13.46 ncftp 219
13.47 nologin – Restrição de Acesso 221
13.48 Partições Windows 221
13.49 Partições – Redimensionamento 222
13.50 Os Modos de Permissão no Unix 222
13.51 Permissões de arquivos e diretórios em Sistemas Unix . . . 227
13.52 Propriedade de Arquivos Apagados 229
13.53 Preenchimento Automático de Caminhos de Diretórios e Comandos 231
13.54 Programação Shell 232
13.54.1 Caminho de Busca de Executáveis 232
13.54.2 numger.sh - Geração de números 234
13.54.3 Criação de Scripts com o bit de Execução Ligado 235
13.54.4 Remoção de Acentuação 236
13.54.5 Verificação dos parâmetros em shell scripts 238
13.54.6 Atribuição de Valores a Variáveis em Shell Scripts 239
13.54.7 Parâmetros em Shell Scripts (Bourne Shell) . . . 240
13.54.8 Depurando shell scripts 244
13.55 rgrep – Recursive Grep 245
13.56 rm 247
13.56.1 Proteção de diretórios 247
13.57 Remoção de Arquivos com Nomes Estranhos 248

13.57.1 Remoção de Arquivos Iniciados em "-" 248
13.58 sed – Stream Editor 249
13.58.1 Substituição Global 249
13.58.2 Deleção de Linhas 252
13.58.3 Caracteres delimitadores 253
13.58.4 Execução de Múltiplos Comandos 253
13.58.5 Impressão de Linhas de Arquivo 254
13.59 Senhas – Geração automática 255
13.60 Senhas Seguras 257
13.61 slice 258
13.62 paste 259
13.63 slocate – Secure Locate 260
13.64 sort 262
13.64.1 Ordenação por Campos 262
13.64.2 Ordenação e Combinação de Arquivos 264
13.64.3 Redirecionamento para Arquivos 265
13.64.4 Comparação de Caracteres 266
13.64.5 Ordenação com Remoção de Linhas Duplicadas . 267
13.65 split 268
13.65.1 Divisão de arquivos por número de bytes e linhas 268
13.66 Divisão de um Arquivo Para Gravação em Disquetes . . . 269
13.67 su – Substitute User 269
13.67.1 Suspensão de processos 270

13.68 tail 271
13.69 tee 272
13.70 tr – Translate 273
13.70.1 Maiúsculas/Minúsculas 273
13.71 traceroute 275
13.72 wget 276
13.72.1 Download de Páginas ou Arquivos na Web 276
13.72.2 Espelhamento de Diretórios 277
13.73 Configuração do prompt 277
13.74 wc – Word Count 279
13.75 WvDial – Conexões PPP 280
13.76 Arquivos setuid/setgid 280
13.76.1 Controle de laço em shells 282
13.77 X Window – Fontes True Type 284
13.78 xlock 285
13.79 xargs 286
13.80 xterm – X Terminal 287
13.81 xwpick – Captura de Telas 287

14 Conectiva Linux – Descrição dos Aplicativos **289**

Índice **292**

Capítulo 1

Introdução

Este livro tem uma abordagem diferente dos livros tradicionais de informática e tem também uma longa história. É uma espécie de diário de bordo de um administrador de sistemas Unix. Tudo o que se encontra aqui foi aprendido ao longo de vários anos. Estas dicas foram usadas para resolver os problemas mais corriqueiros do dia a dia.

Como diversas vezes me vi na situação de ter resolvido um problema e, algum tempo depois, quando o mesmo problema tornou a voltar, não mais me lembrar da solução adotada, decidi então começar a guardar estas informações. Primeiramente fiz uma pequena *shell script*, chamada *logbook.sh*, que me permitia registrar de forma rápida e simples tudo que fazia. Esta pequena *shell script* serviu também para um melhor sincronismo dos trabalhos da equipe de administradores. Além das dicas em si, registrávamos também as alterações que fazíamos nos diversos sistemas. Estes registros são extremamente válidos para identificar rapidamente se alguma mudança originou um problema maior.

Além disto estas informações podem ser usadas no treinamento de estagiários e novos funcionários. O tempo hoje é extremamente escasso devido ao grande número de inovações tecnológicas que precisamos implementar, freqüentemente com prazos curtíssimos. Dispor de um banco de informa-

ções que pode ser consultado por todos representa um enorme ganho de tempo e produtividade.

No dia 3 de março de 1997 resolvi disponibilizar estas informações também através da Internet e criei uma lista eletrônica chamada Dicas-L. Esta lista veicula, nos dias úteis, uma dica sobre informática, com um foco maior em sistemas Unix e derivados. Menos de um mês após sua entrada em funcionamento a lista já contava com mais de 400 integrantes. Hoje já são alguns milhares de assinantes e o número não pára de crescer.

O objetivo inicial era colocar ao alcance de todos os interessados o conhecimento que íamos construindo e ao mesmo tempo incorporar a este banco de dados também as informações de outras pessoas. Das mensagens veiculadas pela Dicas-L desde seu início de funcionamento, cerca de um terço são enviadas por colaboradores, tornando possível o seu funcionamento praticamente ininterrupto desde sua inauguração.

Como não podia deixar de ser, a lista Dicas-L incorporou rapidamente as facilidades oferecidas pela Web e tratou de arrumar um endereço na Web. A página está hospedada atualmente no endereço *http://www.Dicas-L.unicamp.br*. Desta forma os leitores passaram a ter acesso às mensagens já veiculadas, diretamente ou através de um mecanismo de busca e indexação baseado no software *ht://dig*.

Um outro recurso valiosíssimo para administradores de sistemas é a própria Internet. Fazemos uso de diversos serviços disponíveis nesta para nos ajudar em nosso dia a dia. Entretanto, devido à sua extrema volatilidade, tentei evitar ao máximo a menção de URL's neste texto. Pela natureza do texto impresso, este livro corria o risco de ficar rapidamente obsoleto através da inclusão de endereços Web. Eu tive uma prova viva disto ao analisar os links de um CDROM sobre a Internet criado em 1994. Cerca de 90% dos endereços não mais existiam.

Gostaria então de citar como complemento a este livro o serviço de bookmarks, também mantido no endereço Web da Dicas-L, chamado *HotLinks*[1].

[1] *http://www.Dicas-L.unicamp.br/hotlinks*

Este serviço é constantemente atualizado tanto pela inserção de novos endereços como pela remoção de referências desatualizadas. A manutenção deste serviço é feita com o software livre *bk2site* [2], que converte bookmarks no formato Netscape em páginas Web semelhantes ao serviço de busca *Yahoo!*. Uma cópia destas páginas foi incluída no CDROM que acompanha este livro.

Uma outra característica deste livro é a menção freqüente de softwares de uso livre. Hoje em dia é perfeitamente possível, com software livre, se montar praticamente qualquer tipo de solução tecnológica baseada em computadores. Servidores Web, DNS, FTP, correio eletrônico, intranets, segurança, linguagens de programação e bancos de dados podem ser implantados sem custo algum em sistemas livres e abertos.

Mesmo este livro foi inteiramente criado com ferramentas livres: o excelente editor *vi* e o sistema LaTeX. Estas duas ferramentas liberam o autor para se concentrar no conteúdo do que está escrevendo e o sistema de preparação de documentos LaTeX se encarrega, com competência inigualável, do layout, criação do sumário e do índice remissivo. O sistema operacional utilizado foi o *Conectiva Linux*. Todos os comandos foram testados neste ambiente. Sistemas Linux e ferramentas livres e abertas vêm sendo utilizados crescentemente no mundo empresarial e constituem uma alternativa de excelente qualidade a sistemas proprietários.

Este não é um livro para leitura seqüencial, da primeira à última página. Pode ser lido na ordem que se desejar. São dicas, geralmente curtas, que enfocam uma determinada tarefa. É um livro voltado para a resolução de problemas. Não tem por objetivo esgotar qualquer assunto. Mostra um caminho, dentre os vários possíveis, para se tratar diversos problemas. Podem existir, para cada situação, dezenas de soluções diferentes. Não existe a melhor solução. Por mais engenhoso que seja o enfoque adotado para lidar com uma situação, certamente sempre haverá uma forma melhor e mais elegante. Uma vantagem adicional da lista Dicas-L reside justamente no fato de que muitas das soluções apresentadas foram aperfeiçoadas através

[2] *http://bk2site.sourceforge.net*

de sugestões enviadas por seus leitores.

Quanto mais sofisticado tecnicamente você se tornar, melhores serão suas soluções. O objetivo deste livro é ajudá-lo a vislumbrar maneiras mais simples de desempenhar suas tarefas.

Além das dicas em si, incluímos no livro alguns artigos mais longos, resultado de uma reflexão maior sobre determinados assuntos. Incluímos até mesmo um texto sobre o aprendizado do inglês instrumental. chamado *A Reconstrução da Torre de Babel*. Embora não diretamente relacionado à informática, é inegável que o domínio da língua inglesa é imprescindível ao exercício competente de praticamente qualquer profissão nos dias de hoje. Em minha vida profissional na área de informática tenho observado vários profissionais vivenciando situações limitantes justamente por não terem conhecimento suficiente da língua inglesa para a leitura de textos técnicos. O texto aborda esta questão e tenta mostrar que a língua inglesa, para fins de leitura, pode ser aprendida com relativamente pouco esforço e em um tempo razoavelmente curto.

Novamente, este livro é um diário de bordo e tudo que se encontra escrito aqui foi o resultado direto ou indireto de alguma situação com a qual precisamos lidar um dia. Da mesma forma que o endereço Web da Dicas-L é consultado por dezenas de milhares de pessoas mensalmente, esperamos que este livro lhe seja útil e contribua para melhorar o seu desempenho e produtividade. E sem gastar muito, é claro.

Para facilitar o seu aproveitamento ao ler o livro, incluimos o sistema *Conectiva Linux* e as scripts referenciadas no decorrer do mesmo em um CD-ROM.

O Autor.

1.1 Comentários do Editor

A idéia de um livro sobre o conteúdo veiculado na lista Dicas-L me foi bastante atraente. Eu mal sabia a quantidade de trabalho que teria pela

frente no trabalho de editor deste livro.

Bem, não apenas como editor, mas também trabalhando na codificação de estilos e ajudando ao Queiroz com o LaTeX.

O trabalho foi grande, mas valeu a pena no momento em que vi o estado do livro. Repentinamente ele saiu de apenas alguns arquivos num computador para o papel. Tomou forma e ganhou vida.

O que espero é que esse documento vivo os agrade e seja-lhes útil.

O Editor.

1.2 Conteúdo do CDROM

O CD que acompanha este livro possui os scripts mais importantes e que foram colocados aqui para sua comodidade e um melhor aproveitamento. O conteúdo do CD está disposto da seguinte maneira:

- Descrição dos aplicativos do *Conectiva Linux* 4.9.

  Uma descrição dos aplicativos presentes no *Conectiva Linux* 4.9 encontra-se no diretório */usr/doc/betaapp-1.0*. O pacote que contém os arquivos deste livro é o *betaapp-1.0-1cl.noarch.rpm*.

- Livro "Using Samba"

  Este livro, liberado gratuitamente na internet, vem completar as informações disponibilizadas no capítulo sobre Samba. O mesmo está disponível nos formatos PostScript e PDF no diretório */usr/doc/usingsamba-1.0*. O pacote que o contém é o *usingsamba-1.0-1cl.noarch.rpm*.

- Livro "Dicas & Truques"

  O conteúdo deste livro encontra-se em PostScript e em PDF no diretório */usr/doc/dicas_e_truques-1.0*. Os fontes do mesmo, em

LaTeX e imagens em PostScript, encontram-se no pacote *dicas_e_truques-1.0-1cl.noarch.src.rpm*. No pacote com o livro já formatado, *dicas_e_truques-1.0-1cl.noarch.rpm*, também estão contidos os arquivos que geram o site de *Hotlinks*. Estes estão acessíveis logo após a instalação do sistema no endereço *http://localhost/hotlinks*.

Capítulo 2

Por que usar software livre?

Sim, porque usar software livre? Pela simples razão de que nos dias de hoje o computador representa um papel importante, equivalente ao que o lápis e papel desempenhavam alguns anos atrás, para o desempenho da maior parte das profissões. Grande número de empresas fornece computadores a seus funcionários para realizar suas tarefas diárias. A nossa sociedade é extremamente dependente de computadores para seu funcionamento. A enorme fortuna gasta para corrigir os computadores para a virada do milênio é um claro indicador desta dependência.

Pois então, é inconcebível que num mundo tão dependente destas máquinas para seu funcionamento, a maior parte dos programas que regem o seu funcionamento sejam fornecidas por um único fabricante e a preços cada vez maiores. Estima-se hoje que mais de 90% dos computadores pessoais sejam baseados nos sistemas operacionais da Microsoft e em seus aplicativos de produtividade como o conjunto de programas Office.

Há alguns anos atrás a IBM pagava US$ 9,00 por cada PC que vendia com o sistema operacional Windows 3.1 instalado. O preço pago nos dias de hoje situa-se por volta de US$ 60,00 por cada PC vendido com Windows 9x instalado (o preço real é US$ 75,00 porém é possível se obter alguns descontos caso o fabricante assine um contrato com a Microsoft concordando

com algumas exigências da empresa)[1].

Além do alto custo dos programas, muitos fabricantes de computadores temem retaliações por parte da Microsoft caso vendam seus PCs com outros sistemas operacionais pré-instalados. A Compaq, em audiência no processo movido pela empresa Netscape contra a Microsoft alegando práticas monopolistas[2], revelou documentos internos, datados de 1993, onde admitia estes temores. Imagine os preços que teremos que pagar caso este monopólio cada vez maior realmente se consolide eliminando todas as alternativas hoje existentes?

Como se tudo isto já não bastasse, existem rumores de que os termos de licenciamento de produtos Microsoft irão mudar. Nas bases atuais o usuário ao comprar seu computador adquire o direito de uso por tempo indeterminado. Na nova versão o software não será mais adquirido e sim licenciado em bases anuais, exigindo o pagamento de uma nova licença para o uso continuado. Interessante, não?

Esta dominância quase que absoluta, representa perigos reais. A impossibilidade do acesso a computadores irá representar em futuro muito próximo a marginalização das populações ou países que não tenham os recursos necessários para investimento nesta área. Como cidadãos precisamos garantir a qualquer preço o direito ao acesso à computação e à informação cada vez mais essencial para o desempenho de nossas profissões e à nossa vida. Sem esta garantia sem dúvida alguma nos tornaremos no futuro próximo cidadãos ou países de quinta categoria. Não é sem razão que diversos países no mundo, notadamente na Europa, têm apoiado incondicionalmente o movimento de uso de software livre. Estrategicamente não é recomendável que a situação atual de monopólio evolua para um mundo com ainda menos

[1] *Making Money in the Next Free Economy*
http://linuxworld.com/linuxworld/lw-1999-06/lw-06-penguin_2.html

[2] *Even Compaq Feared Microsoft*
http://www.zdnet.com/zdnn/stories/news/0,4586,2212322,00.html
Compaq Feared Microsoft Retaliation
http://news.cnet.com/news/0-1003-200-338898.html
Compaq Feared Microsoft
http://abcnews.go.com/sections/tech/DailyNews/msdoj_compaq990218.html

alternativas.

O governo brasileiro já compreendeu a importância da disseminação da cultura em informática para sua população. Infelizmente o programa nacional de informatização de escolas, PROINFO[3], é todo baseado em software proprietário. O alto investimento em software necessariamente irá limitar o número de alunos contemplados pelo projeto.

Entretanto, mesmo no Brasil existem iniciativas inovadoras nesta área. O governo gaúcho[4] está promovendo estudos para adoção em grande escala do Linux, tanto em nível administrativo quanto em suas escolas. Na esfera federal, o SERPRO, já há algum tempo, vem conduzindo estudos sobre o Linux[5].

O governo mexicano[6] lançou um programa nacional de informatização de suas escolas baseado exclusivamente no Linux e em software livre. Este programa objetiva equipar entre 20.000 a 35.000 laboratórios anualmente durante os próximos cinco anos. O preço para equipar estes laboratórios com software Microsoft seria de US$ 124 milhões. Com Linux basta comprar um CD de distribuição, vendido por aproximadamente US$ 50,00. E mesmo este CD pode ser duplicado infinitamente sem quaisquer implicações legais, pois o software é totalmente livre. Este programa prevê, num futuro próximo, a transição para Linux mesmo nos laboratórios equipados com Windows.

Além desta iniciativa pioneira do governo mexicano, pode-se encontrar na Internet inúmeros relatos de iniciativas isoladas de educadores que conseguiram criar laboratórios de informática para seus alunos utilizando equipamentos obsoletos, descartados por empresas e pelas próprias escolas em que

[3] *http://www.proinfo.gov.br*

[4] *Linux no Rio Grande do Sul*
http://www.revista.unicamp.br/revista/infotec/linux/linux4-1.html
http://www.Dicas-L.unicamp.br/dicas-l/19990803.shtml

[5] *O SERPRO e o Linux*
http://www.Dicas-L.unicamp.br/dicas-l/19990803.shtml

[6] *Mexican Schools Embrace Linux*
http://www.wired.com/news/news/technology/story/16107.html
Mexican Project Plans to Deploy 140,000 Linux Labs in Schools
http://linuxtoday.com/stories/462.html

trabalhavam. Ao passo que os ambientes proprietários requerem computadores cada vez mais poderosos, os softwares de código aberto funcionam de maneira bastante satisfatória em computadores com processador Intel 486 e até mesmo 386.

A informatização do setor educacional, extremamente necessária, se vê frente a uma redução do custo e aumento da potência dos computadores no tocante ao hardware e um aumento significativo no custo do software. O setor educacional nacional possui uma parcela administrada pela iniciativa privada e outra administrada pelo governo federal e estados. As universidades e escolas públicas operam sem cobrar absolutamente nada de seus alunos. Não obstante este propósito nobre e sua importante função social, a maioria dos programas de computadores utilizados por estas instituições, embora com descontos, são pagos. O preço educacional do pacote Office no Brasil é de aproximadamente R$ 250,00, ou US$ 125,00, o que é uma quantia significativa levando-se em consideração a condição financeira da maioria das escolas e universidades nacionais.

Nos EUA, através de acordos celebrados com algumas universidades, o mesmo software é vendido por US$ 5,00, praticamente o custo de criação do CD. A Microsoft porém exige como contrapartida, em muitos casos, uma exclusividade do uso de seus softwares tanto na administração quanto na área acadêmica, o que pode ser desastroso a longo prazo[7].

O uso de computadores nas escolas é algo extremamente importante. Existem várias correntes que discutem como e se os computadores devem ser usados na educação. Não gostaria de me intrometer nesta questão polêmica. A minha visão do assunto é bastante simplista. Julgo que o maior valor dos computadores na educação não reside em programas para editar textos, fazer figuras, brincar ou qualquer outra atividade do tipo. O mais importante é a comunicação e o acesso à informação que os computadores nos propiciam. Ligado à Internet o computador, qualquer que seja ele, se transforma numa ferramenta de grande poder, que nos permite entrar

[7] *Universities and Linux*
http://207.178.22.52/articles/currents/007.html

em contato com culturas diferentes, pessoas interessantes e virtualmente qualquer tipo de informação.

A maior parte de nós certamente já passou pela experiência massacrante de passar por uma escola, por aulas desinteressantes, onde o mais importante é manter a disciplina. A nossa curiosidade natural é sistematicamente eliminada em defesa da abominável disciplina. Disciplina tão destoante das crianças brilhantes, inteligentes e interessadas em aprender que todos nós fomos um dia. Para crianças em idade escolar o constante aprender é tão fundamental quanto respirar. Crianças aprendem o tempo inteiro, mesmo quando não estão na escola. Infelizmente a nossa cultura tende a julgar que o que vale a pena aprender é o que ensinam na escola, o que não poderia estar mais distante da verdade. O poder dos computadores conectados à Internet na educação é justamente atuar como um portal através do qual a curiosidade e ânsia de aprendizado manifestada por toda criança pode ser atendida.

Como conciliar estas necessidades, dentro dos padrões vigentes, com o alto custo de aquisição e configuração de um computador? Os softwares livres são uma alternativa extremamente viável. Para satisfazer a estas necessidades um obsoleto e talvez abandonado computador 386, com Linux, ferramentas de correio eletrônico e um browser Web simples, todos gratuitos, são mais do que suficientes.

Desta forma reduz-se dramaticamente o custo de um computador totalmente configurado. Pelo hardware paga-se pouco ou nada e também o software, de ótima qualidade, está disponível gratuitamente. Lembramos novamente, o importante não é o computador de última geração, dispendioso e carregado com softwares vendidos a preços exorbitantes e sim o que podemos obter através dele, a forma através da qual ele pode ampliar as fronteiras de nosso conhecimento. O computador na educação é importante sim, mas apenas como um instrumento. Mais vale investir em dez computadores obsoletos do que em apenas um de última geração. Considerando-se que inúmeras empresas trocam computadores ainda em condição de serem utilizados por outros mais potentes, uma integração empresa e escola pode trazer ganhos

significativos à sociedade. As escolas ganham por poder oferecer uma melhor formação a seus alunos e as empresas por sua vez poderão contar no futuro com profissionais mais capacitados, cuja demanda cresce vertiginosamente na sociedade de informação em que vivemos.

Felizmente existe uma esperança em nosso futuro. O movimento pelo software livre teve um impulso sem precedentes nos dois últimos anos. Temos hoje uma batalha, cada vez mais feroz, entre dois campos. De um lado, empresas poderosas criando software proprietário e de outro um grupo enorme de programadores espalhados pelo mundo inteiro, cada um deles dedicando-se a criar e desenvolver software livre. A ponta mais visível deste movimento singular é o sistema operacional Linux, que veio ao mundo por meio das mãos de Linus Torvalds. Em 1991, Linus, então um estudante de ciência da computação na Finlândia, criou um clone do sistema Minix. O primeiro anúncio do Linux apareceu no fórum comp.os.minix, dedicado à discussão do sistema operacional Minix, também semelhante ao Unix, criado por Andrew Tannenbaun, um respeitável professor de ciência da computação.

Desde seu aparecimento o uso do Linux não para de crescer. Nos dois últimos anos, sua popularidade atingiu níveis nunca esperados. Estima-se que sua base de usuários se situe hoje por volta de 10 milhões. Diversas empresas respeitadas e famosas do mundo da computação como IBM, Dell, Compaq, Oracle, já anunciaram seu suporte e respeito elogioso ao Linux.

A despeito do sucesso estrondoso do Linux, o principal criador deste movimento foi na verdade Richard Stallman. Stallman começou o movimento pelo software aberto e livre em 1984, nos primeiros momentos da indústria de computadores, quando começaram a aparecer os primeiros softwares comerciais. Stallman era então um pesquisador no MIT e trabalhava na área de inteligência artificial. A tradição dos hackers de então era compartilhar mutuamente seu conhecimento, num ambiente de intensa colaboração. Em determinada ocasião Stallman precisou corrigir o *driver* de uma impressora que não estava funcionando corretamente. Solicitou então, ao fabricante, o código fonte do programa para que pudesse realizar as correções necessárias.

Para sua surpresa e indignação, seu pedido foi negado. O fabricante alegou que o código fonte era segredo comercial e não podia ser cedido a terceiros. Stallman iniciou então seu esforço gigantesco de criar versões abertas para todas as categorias de software existentes, comercializadas sem acesso ao código fonte. Criou então um compilador C, um editor de textos extremamente poderoso e popular chamado emacs e fundou a *Free Software Foundation* (FSF)[8]. Nos anos que se seguiram a FSF criou uma grande parte dos aplicativos utilizados por todos os sistemas semelhantes ao Unix, como Linux e FreeBSD, hoje tão populares.

A maior façanha de Stallman não foi a criação de vários programas poderosos e bem escritos. Escrever programas e disponibilizar o código fonte para quem quer que seja poderia resultar justamente no contrário do que pretendia, a liberdade no uso do software. Afinal de contas, se o código é aberto, o que impede que alguém se utilize deste mesmo código, faça algumas modificações, declare que o programa é proprietário e restrinja o acesso ao código modificado? Para impedir que isto acontecesse, Stallman escreveu um documento que estabelece a forma sob a qual programas de código fonte aberto podem ser distribuídos. O documento especifica que o programa pode ser usado e modificado por quem quer que seja, desde que as modificações efetuadas sejam também disponibilizadas juntamente com seu código fonte. Este documento chama-se *Gnu Public License* ou GPL[9] como é mais conhecido.

O Linux somente possui toda esta força e penetração atual devido ao enorme trabalho desenvolvido por Stallman e por um batalhão de outras pessoas. Estes voluntários dedicaram muito de seu tempo criando programas excepcionalmente bons, que junto com o kernel do Linux, propiciaram a milhões de pessoas um ambiente computacional de trabalho excepcionalmente bom e que melhora a cada dia.

O Linux entretanto, na pessoa de seu criador e coordenador, soube melhor aglutinar o imenso potencial de colaboração da Internet em torno de

[8] *http://www.gnu.org*
[9] *http://www.gnu.org/copyleft/gpl.html*

seu projeto. Contribuições são aceitas, testadas e incorporadas ao sistema operacional a uma velocidade nunca vista.

E é justamente este movimento, cada vez mais poderoso, que nos dá grandes esperanças de tornar a computação disponível a um grande universo de pessoas, por inúmeras razões. Em primeiro lugar vem a questão de custo. Por custo entenda-se tanto a questão do software em si quanto a possibilidade de reutilizar computadores já fora de uso. O software é totalmente gratuito e o hardware pode ser obtido a um custo baixo ou nulo, no caso de doações. Além disto podemos contar com a boa vontade e disposição de ajudar de um número incontável de pessoas. O caráter quase que religioso dos adeptos do movimento de software livre garante um excelente suporte para quem precisa de ajuda. Existem espalhados por todo o mundo os chamados LUG, ou *Linux User Groups*. Em Phoenix, nos EUA, o grupo de usuários Linux promove reuniões mensais para ensinar os conceitos do Linux aos novos usuários. Nestas reuniões mensais os interessados em aprender mais sobre Linux podem trazer seus computadores para serem configurados gratuitamente[10]. Onde já se viu?

Ótimo, então você chegou até aqui. Possivelmente podemos contar com mais um adepto à propagação do uso de software livre. A pergunta agora é: "O que pode ser feito para fortalecer este movimento?". Várias coisas, dependendo da sua habilidade. Se você é um programador, em qualquer linguagem, porque não começar compartilhando aqueles pequenos ou grandes programas que escreveu? Se você não o fizer e ciumentamente guardar o programa para você mesmo, muito em breve ele será inútil ou obsoleto e não servirá para mais nada. Se for compartilhado quem sabe um dia você poderá alcançar a mesma notoriedade e projeção de Linus Torvalds? Nada é impossível.

Se você não sabe programar não tem problema. Documente o que sabe. Todos profissionais de informática sabem que uma boa documentação pode vir a ser até mais importante do que o programa em si. Se você não tem

[10] *Phoenix Linux User Group Let Us Plug You In*
http://www.plug.phoenix.az.us/plug_form.html

tempo para escrever documentos extensos ou complexos não tem problema. Escreva notas curtas ou dicas relatando o que aprendeu, aqueles truques que ajudam a resolver problemas. Participe de listas de discussão (ou crie a sua própria[11], compartilhando o que sabe e também aprendendo com os outros.

Tudo bem, você não quer fazer nada disto. Não tem problema. Comece a usar software livre. A transição da plataforma Windows para Linux não é algo trivial, embora esteja se tornando cada vez mais fácil. Tente usar software livre na plataforma que está acostumado, Windows ou Macintosh, apenas para citar algumas. Por exemplo, use o StarOffice[12], um conjunto de aplicativos para automação de escritórios disponível gratuitamente para download na Internet. Como o StarOffice possui versões para diversos sistemas operacionais, quando você fizer a transição para um sistema Linux poderá continuar usando os aplicativos com que se familiarizou. Outro exemplo notável é o software GIMP[13], abreviação de *Gnu Image Manipulation Program*, usado para tratamento de imagens. Nascido em ambiente Linux, este software possui funcionalidade em muitos pontos equivalente ao Adobe Photoshop. Existe também uma versão adaptada para o ambiente Windows[14]. Então porque não usá-la? Não custa nada.

Lembre-se, o que quer que possamos fazer para contribuir para o movimento de software livre é válido. Comece usando um software livre, qualquer que seja ele. Não precisa mudar para Linux imediatamente. Comece devagar. O importante é compreender que a causa do software livre é válida e importante para seu futuro e do seu país. Compartilhe sua opinião com seus amigos. Peça-lhes para contarem aos seus amigos, e aos amigos dos amigos. Crie uma lista de discussão. Não pense que seu esforço é insignificante. A Internet e a possibilidade de comunicação que nos oferece serve como um amplificador de nossas idéias e convicções como nunca existiu antes. Não se esqueça de que o movimento em prol do software livre se reforça justamente

[11] *http://www.egroups.com*
[12] *http://www.sun.com/staroffice*
[13] *http://www.gimp.org*
[14] *GIMP for Windows*
http://user.sgic.fi/~tml/gimp/win32/

por meio de pessoas que o usam e o julgam de boa qualidade. A maioria dos programadores não obtém ganhos financeiros. O que os gratifica e incentiva a produzirem mais é justamente o reconhecimento de sua competência e seu talento.

O mais difícil entretanto é vencer hábitos profundamente arraigados. Todos nós criamos, com o passar do tempo, uma profunda resistência a mudarmos os nossos hábitos. A argumentação geralmente segue a linha de que se foi tão difícil aprender a usar os programas de computador que possuímos por que deveríamos mudar tudo de uma hora para outra? Novamente, lembre-se de que um mundo sem opções, em qualquer área que seja, em última instância será prejudicial a todos nós. Você se lembra ou já ouviu falar da crise do petróleo? Pois então, o risco que corremos com os programas de computador é semelhante. À medida que o preço dos programas de computadores sobe os reflexos se manifestam em praticamente todos os aspectos de nossa vida. Lembre-se que todos os custos envolvidos no desenvolvimento dos serviços por nós utilizados em nosso dia a dia são pagos por nós. O dentista que usa um computador para manter sua lista de clientes, a empresa que fabrica a televisão que adquirimos, a alimentação que consumimos. Como a nossa sociedade é extremamente informatizada qualquer aumento no custo de manutenção de sistemas computacionais representa um acréscimo na conta que todos nós, de uma forma ou de outra, temos que pagar.

É proibido então ganhar dinheiro vendendo software? De forma alguma, o que afirmamos é que o software deve ser livre. Sistemas computadorizados são indispensáveis ao funcionamento de diversas empresas. O que ocorre em caso de problemas? A empresa tem que recorrer a quem lhe vendeu o software. Esta empresa pode ou não ter os recursos humanos e a vontade para corrigir o problema imediatamente. O preço cobrado pela manutenção pode ser exorbitante, fora do alcance da empresa. Se o código fosse livre qualquer pessoa capacitada tecnicamente poderia corrigir o problema.

Além do mais, mesmo com software livre é possível se ganhar dinheiro. Apenas para citar o exemplo mais marcante, a empresa RedHat, que comercializa uma das versões mais difundidas do Linux obteve uma valori-

zação astrônomica no primeiro dia em que teve suas ações comercializadas na bolsa de valores. Dois de seus proprietários possuem hoje bilhões de dólares devido a esta valorização fantástica. Existem também várias outras empresas e profissionais estabelecidos no mercado que derivam ganhos significativos trabalhando ou fornecendo suporte a software livre.

Mas não se esqueça, não seja um fanático ou um chato. O importante é ser sutil e conquistar novos adeptos de maneira suave e irreversível. Ao invés de criar novos inimigos, crie novos amigos. Afinal de contas quem não é grato por uma informação que permita economia ou ganho financeiro? Pois é isto mesmo, o computador e os programas que usamos servem como meio de vida para cada vez mais pessoas.

Está convencido? Então deixe-me contar-lhe onde descobrir software gratuito. Na Internet existem vários endereços onde podemos obter informação sobre software realmente livre. Para a plataforma Windows um ponto de partida é o sítio *Freewarehome*[15], extremamente bem organizado o que simplifica em muito a tarefa de se encontrar aquele software gratuito de que tanto necessitamos. Para o ambiente Linux não existe nada melhor do que o sítio *Freshmeat*[16].

Como proceder para começar a usar o Linux? O primeiro passo é obter informação sobre o sistema, como funciona, como instalar, como resolver problemas. No Brasil temos o sítio de Linux hospedado na Unicamp[17]. Temos também o sítio da empresa Conectiva[18], sediada em Curitiba, com filiais em São Paulo e Rio de Janeiro que criou uma versão nacional do Linux. Esta versão fornece suporte à acentuação e nos oferece vários documentos em português, cobrindo tanto a documentação do sistema quanto vários manuais para usuários. Isto é apenas o começo, documentação sobre Linux é o que mais existe na Internet.

Vença seus velhos hábitos e comece a trabalhar por um mundo com alter-

[15] *http://www.freewarehome.com*
[16] *http://www.freshmeat.net*
[17] *http://linux.unicamp.br*
[18] *http://www.conectiva.com.br*

nativas, onde seja possível exercer permanentemente o direito de escolha. Esperamos também que o software de uso livre lhe seja útil e proveitoso.

Referências Adicionais

- *Practical Managers Guide to Linux*[19]
  Este excelente documento cobre em detalhes abordagens contra e a favor da adoção de software livre em geral e do Linux em particular. Abordagem bastante didática e clara com links para dezenas de documentos relacionados. Leitura mais do que necessária.

- *Voices from the OpenSource Revolution*[20]
  Publicação da editora O'Reilly and Associates, abordando o fenômeno do software livre. Cada capítulo do livro foi escrito por personalidades chave deste movimento como Richard Stallman, Linus Torvalds e Larry Wall (criador da linguagem Perl), para citar apenas alguns. Leitura essencial para quem quiser obter uma melhor compreensão deste movimento revolucionário. Este livro, como não poderia deixar de ser, é livre e pode ser obtido integralmente na Internet.

- *The Cathedral and the Bazaar*[21]
  Documento escrito por Eric Raymond descrevendo as motivações dos programadores que trabalham gratuitamente no desenvolvimento de software livre. Raymond faz uma analogia do ambiente proprietário e aberto de desenvolvimento de software com a catedral e o bazar. Imperdível.

- *The Halloween Documents*[22]
  Documentos internos da Microsoft que discutem a ameaça à hegemonia da Microsoft trazida pelos movimento de software livre. Revelador de táticas, nem sempre lícitas, utilizadas pela empresa para manter sua dominância do mercado.

[19] *http://www.osopinion.com/Opinions/GaneshCPrasad/GaneshCPrasad2.html*
[20] *http://www.oreilly.com/catalog/opensources/book/toc.html*
[21] *http://earthspace.net/~esr/writings/cathedral-bazaar*
[22] *http://www.opensource.org/halloween.html*

- *Uncle Sam, the GNU and the Penguim*[23]
  Comentários sobre um documento do governo americano afirmando que os produtos Microsoft são caros e inseguros.

- *Linux Hotlinks*[24]
  Minha coleção pessoal de links sobre Linux.

Este artigo foi publicado originalmente na Revista do Linux[25], da empresa Conectiva.

[23] *http://www.osopinion.com/Opinions/XavierBasora/XavierBasora21.html*
[24] *http://www.Dicas-L.unicamp.br/hotlinks/Linux/*
[25] *http://www.RevistaDoLinux.com.br*

Capítulo 3

Software Livre: Bom, bonito e barato

No início da era dos computadores, dos poderosos e inacessíveis *mainframes*, como são mais conhecidos, apenas empresas com um caixa reforçado podiam contar com esta importante ferramenta. Além do custo exorbitante do hardware, o software também não ficava atrás. O alto custo se devia ao fato de que tanto o hardware quanto o software eram obrigatoriamente adquiridos junto ao mesmo fornecedor. A solução fornecida era um pacote fechado e a única alternativa era aceitar calado o que era imposto pelo fabricante.

Felizmente, com a redução do custo dos computadores e também pela disseminação do uso do Unix no mundo acadêmico, o panorama começou a mudar. A empresa AT&T, criadora do Unix, por força da legislação existente que a impedia de comercializar software, tornou o Unix e seu código-fonte disponível a universidades e instituições de pesquisa. Bastava pagar o custo da fita e as despesas de transporte. Em pouco tempo o Unix se tornou extremamente popular no meio acadêmico. Dentro do espírito de colaboração vigente neste meio, vários programas começaram a ser desenvolvidos e compartilhados sem custo algum. O objetivo maior era ajudar e não obter ganho financeiro imediato. Deste início modesto, na década de 70, até os

dias de hoje, o software livre percorreu um longo caminho. O mundo do software comercial se sente ameaçado por esta proliferação de software de boa qualidade e realiza inúmeros ataques[1] tentando minar a credibilidade do software livre.

Os argumentos utilizados com mais freqüência contra o software livre são aqueles que afirmam que sua qualidade deixa a desejar e que o suporte é nulo ou inexistente. Não vou chegar ao exagero de afirmar que todo software livre é de excelente qualidade e que pode ser usado em total confiança. Mas também não ouso afirmar que todo software comercial é confiável e representa um excelente retorno no investimento (às vezes exagerado) realizado. O importante é que o acesso aos computadores e, conseqüentemente, à informação que armazenam seja com software proprietário ou livre, é essencial. O computador é um instrumento de acesso à informação, vital à nossa educação e formação continuada. O acesso a este instrumento deve ser o mais difundido possível a todos. Por quê insistir com soluções dispendiosas quando funcionalidade equivalente ou até mesmo superior pode ser alcançada com sistemas totalmente configurados com software livre? Nos próximos parágrafos faço um relato resumido de algumas experiências bem sucedidas no uso de software livre.

Considero-me um privilegiado por ter sido um dos primeiros a ter acesso à Internet no Brasil, já em 1990. Na época o programa que fazia o maior sucesso chamava-se *Archie*[2] . Este programa fornecia informação sobre programas disponíveis para download existentes na Internet. As dificuldades entretanto eram imensas. A nossa conexão com a Internet na época era de 2400 bps. O acesso interativo ao programa *Archie* era praticamente impossível, tanto pela lentidão de nosso acesso quanto pela sobrecarga do servidor, que era extremamente útil e popular. Isto entretanto não nos fazia desistir. O banco de dados *Archie* podia também ser consultado via email. A pergunta era enviada e no dia seguinte, com sorte, recebíamos

[1] *Linux nas Empresas*
http://www.Dicas-L.unicamp.br/hotlinks/Linux/Linux_nas_Empresas
http://www.osopinion.com/

[2] *Archie – http://www-ns.rutgers.edu/htbin/archie*

uma lista com sítios ftp que continham algo mais ou menos na linha do que buscávamos. Embora isto tudo possa parecer extremamente rudimentar na Internet de hoje, povoada de websites, o nosso deslumbramento com o potencial desta ferramenta fantástica era enorme. Tínhamos um problema e não tínhamos o dinheiro. Tudo bem. O super *Archie* estava ali mesmo para nos ajudar.

Um dos primeiros problemas que resolvemos desta forma foi quando realizamos a migração de computadores Digital/VMS para uma arquitetura mais nova. Um dos problemas é que as unidades de fita de rolo, não mais seriam suportadas no novo equipamento. Uma das alternativas era copiar todas as fitas, uma a uma, para disco e em seguida gravá-las na nova unidade de armazenamento. Tal procedimento, é claro, era inviável. Eram centenas de fitas e certamente, se houvesse uma alternativa, seria a mais completa salvação. Bom, o único computador onde tínhamos uma unidade de fita de rolo que poderia ler as fitas gravadas no sistema VMS era um equipamento da Sun, rodando o venerável SunOS. Restava então achar algo que entendesse o formato de *backup* do sistema Digital/VMS e o gravasse em um sistema Unix, ou seja, fazer a transferência do VMS para o Unix. Consultamos o Archie com o termo *vmsbackup* e surpreendentemente, achamos um programa[3] com este nome que fazia exatamente o que desejávamos. Totalmente gratuito, 100% funcional, sem bugs, não pagamos nada por ele e economizamos diversas (possivelmente centenas) de horas de trabalho. Vitória!

Outro problema relativamente sério que resolvemos com o auxílio de software livre foi a questão do backup. De algumas poucas estações de trabalho adquiridas em 1990, o nosso ambiente, com a redução do preço dos computadores, cresceu enormemente. Com este crescimento veio a necessidade de se fazer o backup de dados distribuídos por vários equipamentos. O Unix provê mecanismos de backup via rede mas esta alternativa simplesmente não era viável devido à lentidão da unidade de fita (256kbps) e ao volume de dados. O tempo dispendido era simplesmente inaceitável. Além do mais existia a questão da catalogação dos backups. As fitas DAT 8MM

[3] *vmsbackup – http://ftp.unicamp.br/pub/unix-c/tapes/vms-backup.tar.gz*

que utilizávamos então podiam armazenar até 5GB. Com as fitas de rolo, cuja capacidade era menor, por volta de 150MB, a prática era se usar uma fita por sistema de arquivos. Bastava colar uma etiqueta na fita dizendo a data do backup, nível e o nome do sistema de arquivos: "*/usr nível 0 10/11/1990*". Uma fita DAT podia conter dezenas de sistemas de arquivos.

Sem um sistema de catalogação eficaz, o *backup* até que podia ser feito porém a recuperação dos dados consumiria um tempo inaceitável. Novamente, sem os recursos financeiros para a aquisição de uma solução comercial, a alternativa que nos restava era achar um software livre que atendesse a estas necessidades.

Após alguns estudos e pesquisa, descobrimos o software *Amanda*[4] , acrônimo de *Advanced Maryland Network and Disk Archiver*. O *Amanda* utiliza vários conceitos inovadores. O backup de diversas máquinas da rede é feito primeiramente no disco do servidor. Desta forma diminui-se o tempo gasto, visto que vários backups podem ser realizados em paralelo. Uma vez gravados os backups no disco, faz-se então a transferência destes arquivos para a fita. Adicionalmente, no início de cada fita é gravado um cabeçalho que permite, durante operações de recuperação de dados, que se posicione a fita no local desejado em poucos segundos. Tivemos algum trabalho para adaptá-lo ao nosso ambiente, porém com a ajuda dos desenvolvedores e o acesso ao código-fonte, conseguimos colocá-lo em produção, onde já é utilizado com sucesso há vários anos.

Um outro problema que resolvemos com auxílio de software livre foi a indexação de todos os sítios Web de nossa instituição. A indexação de um único servidor Web é tarefa trivial, existe uma grande variedade de programas livres disponíveis para isto. Como possuímos um grande número de servidores Web, por volta de 300, num total de 200.000 documentos, tínhamos o problema de acesso a toda esta informação. Precisávamos criar um serviço de busca e indexação semelhante ao oferecido pelo Altavista ou Excite. Já possuíamos o excelente software Altavista para indexação do acervo da biblioteca. Poderíamos facilmente usá-lo para indexar todos os

[4] *Amanda – http://www.amanda.org*

sítios, porém nos deparamos com um problema. A nossa versão do software roda exclusivamente em plataforma Digital. O equipamento de que dispúnhamos não comportava discos adicionais, e não dispunha de potência computacional suficiente para suportar a carga adicional de processamento. Após alguma pesquisa na Internet, descobrimos o software ht://dig [5] . A configuração do software foi feita por nosso guru[6] em apenas uma manhã, e após alguns ajustes com os webdesigners, em uma semana estava tudo funcionando perfeitamente. Novamente, não precisamos gastar nada para colocar no ar um serviço extremamente útil à nossa comunidade.

Um relato de todos os problemas que resolvemos com software livre seria praticamente impossível. São diversos os casos bem sucedidos e realmente é difícil nos lembrarmos de todos eles. Devido ao orçamento limitado adotamos, com sucesso, uma estratégia totalmente diversa das maioria das empresas. Sempre que nos deparamos com um problema, procuramos por uma solução livre. Em caso de insucesso (o que é cada vez mais raro), buscamos então, com muita relutância, uma alternativa comercial. Existe também outra alternativa, que é convencer seu chefe de que o que você encontrou na Internet é exatamente o que ele ou ela precisa ;-)

Na questão do suporte a história não poderia ser diferente. Em diversas ocasiões, mesmo com contrato de suporte com fabricantes de produtos comerciais, a solução foi encontrada através da Internet. Problemas relativamente simples, para os quais o nosso canal oficial afirmou não possuir uma solução, foram resolvidos com apenas algumas mensagens enviadas a listas de discussão e newsgroups da Usenet. Após um certo tempo passamos a também adotar como regra utilizar os recursos da Internet primeiro e apenas em último caso recorrer ao fabricante.

Como não poderia de ser, o acesso ao código-fonte representa uma vantagem e uma segurança para o usuário. Um dos sistemas Unix que utilizávamos apresentava um bug incômodo. Se por alguma razão um administrador estivesse logado em uma tela como o superusuário (root) e houvesse uma queda

[5] *ht://dig – http://www.htdig.org*

[6] Fábio Mengue, webwizard do Centro de Computação da Unicamp

de energia ou algo parecido na estação de trabalho remota, a janela, onde o usuário root estava logado, não era desconectada pelo sistema servidor. Ela ficava "vagando" pelo sistema operacional. Um usuário normal que logasse poderia receber de presente esta tela já com o root logado. Felizmente nenhum de nossos usuários se aproveitou desta situação para realizar estragos no sistema e fomos mesmo notificados do fato várias vezes.Tentamos, então, descobrir como resolver o problema. Consultamos algumas listas e a resposta veio quase que imediatamente. Não havia uma correção oficial da empresa e para resolver o problema precisávamos alterar o código-fonte. O problema é que a licença de acesso ao código-fonte custava algumas centenas de milhares de dólares, ou seja, fora de questão. O único caminho que nos restou foi esperar por uma nova versão e neste meio tempo torcer para que não ocorresse nada mais sério. Felizmente conseguimos sobreviver a isto. Não preciso nem dizer o que aconteceria se um problema semelhante ocorresse em sistemas Linux. Em questão de algumas horas o problema certamente seria resolvido.

Se há alguns anos atrás a localização do software livre era feita por meio de mecanismos rudimentares e ineficientes, o mesmo já não ocorre hoje. No mundo Linux temos centenas de portais, onde toda esta informação encontra-se muito bem organizada. O site Freshmeat[7] , por exemplo, é um ponto obrigatório para quem deseja implementar soluções baseadas em software livre. Todos os dias são exibidas na primeira página as novidades (e que novidades) do mundo do software livre.

Instalar software livre também ficou infinitamente mais fácil. Se antes o processo de instalação envolvia a leitura da documentação, compilação, acerto no código-fonte e muito mais, hoje em dia, em 99% dos casos, tudo que é necessário é emitir um único comando[8].

Dos softwares citados anteriormente, apenas o programa *vms2unix* (a demanda não deve ser muito alta mesmo) não se encontra disponível para o

[7] *Freshmeat – http://www.freshmeat.net*

[8] *Gerenciamento de Software com RPM*
http://www.rpm.org
http://www.revista.unicamp.br/navegacao/index3.html

ambiente Linux. Instalar o *Amanda* ou *ht://dig* é bastante simples. Estamos vivendo em um mundo onde temos uma grande variedade de software livre à nossa disposição, de uma qualidade cada vez maior e com os quais podemos experimentar à vontade e modificar da maneira que desejarmos.

Tudo que foi dito acima está longe de ser um caso isolado. A Internet, desde seus primórdios, funciona baseada quase que exclusivamente em software livre. O protocolo de comunicações TCP/IP, pedra fundamental da Internet, é livre. O software *Bind*[9], que nos ajuda a localizar computadores na Internet, e o programa *sendmail*[10], de longe o mais popular processador de mensagens de correio eletrônico, também são livres. O servidor *Apache*[11], é o preferido por 55% dos sítios na Internet. Em suma: sem software livre a Internet pára. Se a Internet, que é um enorme sucesso, funciona com software livre, porque a sua empresa não pode copiar o modelo?

É evidente que ainda possuímos diversos softwares comerciais em uso em nosso ambiente, particularmente nos ambientes de computação pessoal. O que é notável, entretanto, é que nos últimos tempos um crescente número de usuários, voluntariamente, realiza a substituição de seu ambiente de software proprietário por alternativas livres, com particular destaque para o Linux. Em contraste com a situação de alguns anos atrás, onde sistemas Unix e software livre eram utilizados por muito poucos, a popularização e crescente aperfeiçoamento desta alternativa faz com sistemas totalmente baseados em software livre sejam uma solução viável para uma ampla faixa de usuários. A combinação Conectiva Linux, com suporte total à acentuação e com um grande número de documentos e tutoriais traduzidos para o português,com StarOffice[12], é uma solução de alta qualidade para a computação pessoal. Basta ousar um pouco. Após as dificuldades iniciais, os ganhos obtidos pela utilização da grande variedade de software livre são

[9] *Internet Software Consortium: BIND*
http://www.isc.org/products/BIND

[10] *Sendmail*
http://www.sendmail.org

[11] *Apache – http://www.apache.org*
http://www.e-softinc.com/survey/data/index.html

[12] *StarOffice – http://www.sun.com/staroffice*

substanciais.

Concluindo, o software livre certamente já provou seu valor e qualidade. O software livre não exclui, nem deve excluir, as alternativas comerciais ou vice-versa. O melhor dos mundos será aquele onde existam alternativas adequadas a todos os tipos de necessidade e recursos. Gerentes de informática que optem por uma equipe de informática reduzida ou disponham de mais dinheiro do que competência técnica podem optar, com um mínimo de riscos, por alternativas comerciais estabelecidas e comprovadas. Os mais engenhosos e empreendedores poderão optar com sucesso por alternativas livres. O mais importante é que existam alternativas. Imagine um mundo onde todo o software tem que ser adquirido de um único fabricante. A julgar pela importância que os computadores possuem hoje em nossa sociedade, não é exagero traçar um paralelo com a situação de alguém vivendo na Lua tendo que comprar o ar de um único vendedor. Sentiu um calafrio?

Pois então, você certamente já ouviu várias vezes que as melhores coisas da vida são gratuitas. O pôr do sol, uma noite de lua cheia e, porque não, software livre...

Capítulo 4

Introdução ao DNS

Você já parou para pensar como o seu computador é bem relacionado? No seu browser Web, basta digitar o nome de qualquer computador existente na Internet, que instantaneamente a conexão é efetuada. Seu computador conhece todos eles. Mas como isto é possível? O seu computador não passa de um velho 486, com 16MB de memória. E a Internet já possui milhões de computadores conectados. Como pode ele conhecer e conversar com todos eles?

O segredo está no DNS, ou *Domain Name Service*. Este é o protocolo que torna possível que qualquer computador encontre qualquer outro dentro da Internet em questão de segundos (ou muito menos do que isto). O seu computador pessoal faz uma pergunta a um outro computador que por sua vez se encarrega de encontrar a informação que você precisa, também fazendo perguntas a outros computadores.

Mas vamos voltar um pouco mais no tempo, aos primórdios da Internet. Naquele tempo, existiam poucos computadores na Internet (que nem se chamava Internet ainda). Além de existirem poucos computadores, as pessoas que cuidavam destes computadores também se conheciam. E existia uma lista contendo os nomes de todos os computadores existentes. Esta lista na verdade não continha apenas nomes. Ela continha linhas relacionando no-

mes com números. Isto porque os computadores não se comunicam através dos nomes que possuem e sim por meio de números que lhes são atribuídos. Os chamados números IP (de Internet Protocol, você já deve ter ouvido falar de TCP/IP, não?). Mas é muito difícil memorizar números, nós seres humanos nos lembramos muito mais facilmente de nomes. O que é mais fácil, lembrar-se que um servidor web atende pelo nome de *www.dominio.com.br* ou que o seu número IP é 10.0.0.1? Como os computadores só se conhecem pelo numero, criou-se um mecanismo que permitiu a tradução do nome, usado pelos seres humanos que operam os computadores, para o número que os computadores usam em sua comunicação. E começamos com a lista.

O cadastro era mantido por uma entidade central, que cuidava da distribuição de números aos computadores que se ligavam à Internet. Sempre que um novo computador era cadastrado, a lista atualizada era distribuída aos administradores dos computadores ligados à Internet. Desta forma, cada computador conseguia se comunicar com todos os demais. Bastava olhar em sua lista. A Internet era então uma típica cidade do interior, todos se conheciam diretamente e os novatos eram apresentados a todos que já faziam parte da comunidade.

É claro que nem tudo dura para sempre. A Internet foi invadida por todos os tipos de pessoas e se tornou uma comunidade virtual, um espelho do mundo real. E o velho esquema de manter a listinha, contendo os nomes de todos os computadores passou a não ser mais viável. Afinal de contas, qual computador tem o poder de consultar uma "listinha" de alguns milhões de linhas sempre que precisasse enviar alguma coisa para outro computador na Internet? Certamente não um velho e ultrapassado 486. Precisou-se inventar uma outra maneira de fazer com que os computadores se encontrassem, mesmo sem possuir a tal listinha (que já nem era mais listinha).

Foi então que inventaram o DNS. Com o DNS, abolia-se a centralização da informação. Não existe mais um computador na Internet que conheça todos os demais. O que aconteceu foi que a autoridade sobre a informação foi diluída. Desta forma, não existe mais um dono da verdade, a informação está distribuída por milhares de computadores, que conhecem muito bem apenas

alguns computadores. Tomemos o exemplo da Unicamp. Nós temos aqui um computador que tem uma lista de todos os computadores conectados à Internet dentro da Universidade. Qualquer computador na Internet que queira achar algum computador dentro da Unicamp tem que perguntar a este computador. E, desde que o computador procurado exista, ele fornece a informação solicitada. Mas e como chegar até a Unicamp? Novamente não é difícil descobrir. Os projetistas do DNS na verdade nada mais fizeram do que imitar a vida real. Imagine que você esteja em uma cidade grande e deseja chegar até um determinado bairro. Você pára alguém na rua e pergunta: "Como faço para chegar até o bairro X?". O seu interlocutor não sabe, mas lhe orienta para ir até determinado lugar e perguntar novamente. E lá vai você perguntando, até que encontra alguém que sabe lhe dizer onde se encontra o local que está procurando.

A tradução de nomes em números na Internet funciona exatamente da mesma forma. Você configura o seu computador com o nome do servidor de nomes local. E é ele quem vai fazendo as perguntas para você, até obter uma resposta. A resposta pode ser o número IP do computador com o qual você quer se comunicar ou uma negativa, dizendo que o computador procurado não existe.

Mas para entender melhor tudo o que foi dito, vamos analisar o nome de um computador. Vejamos o nome *www.dominio.com.br*. Dá para ver que o nome é composto por cinco componentes:

$$\left|\text{www}\right| + \left|\text{dominio}\right| + \left|\text{com}\right| + \left|\text{br}\right| + \text{“.”}$$

Isto mesmo, cinco componentes. Embora não pareça, o “.”, que a maioria de nós nem se lembra de digitar quando escreve o nome de um computador (e muitos de nós nem mesmo sabemos que este ponto existe) representa o domínio de mais alto nível na hierarquia de nomes de computadores. Então no nome de computadores, a hierarquia mais alta fica à direita, ao passo que a mais baixa fica à esquerda. Isto é fácil de se visualizar. O “.” contém os servidores de nomes de mais alto nível, que possuem apontadores para os computadores de nível imediatamente inferior, os domínios geográficos,

aos quais pertencem o Brasil (*br*), Japão (*jp*), Canadá (*ca*), Portugal (*pt*) e todos os demais países. Neste mesmo nível situam-se os domínios *com* (entidades comerciais), *net* (comunicações), *org* (organizações), *edu* (educação), *gov* (governo) e *mil* (militares). Os domínios de segundo nível apontam para os domínios de terceiro nível. O nosso domínio fictício, por exemplo, está dentro do Brasil. Dentro do servidor de nomes do domínio *br* existe uma referência ao servidor de nomes do domínio *dominio*, que conhece todos os computadores do domínio *dominio.com.br*.

Agora vamos ver a situação em que alguém deseje encontrar o computador *acme.com.br*. Ele primeiro faz a pergunta ao seu servidor de nomes local. Este servidor de nomes não conhece o computador *acme.com.br*. O que faz então? Pergunta a outro, neste caso, aos servidores do domínio ".". Estes servidores de nomes também não conhecem o computador *acme.com.br* mas analisando o nome descobrem que este computador está dentro do Brasil (*br*) e instruem o servidor de nomes local a perguntar aos servidores do domínio *br*. E lá vai o nosso servidor de nomes perguntar aos servidores do domínio *br* onde está *acme.com.br*. Eles (servidores do domínio *br*) também não sabem, mas sabem que tal computador, se existir, está dentro da empresa chamada *Acme*. E novamente instruem o servidor de nomes local a perguntar aos servidores de nomes da *Acme*. Desta vez a resposta é definitiva. O computador *acme.com.br* existe e o número IP correspondente é *200.200.20.1*. Acabou a busca. Não foi tão difícil assim. Para localizar o computador *acme.com.br*. dentre os milhões existentes na Internet, foram necessárias apenas quatro perguntas (ver figura). Antes que vocês saiam procurando o computador *acme.com.br*, ele não existe (ainda). Utilizei este nome aqui apenas para ilustrar o conceito. Veja a figura 4.1.

Agora, quando é usado o DNS? Sempre que você usar um programa que usa o nome de um computador o DNS entra em ação. Você está mandando uma mensagem eletrônica para *usuario@dominio.com.br*. O DNS tem que descobrir para você qual o número IP do computador que recebe mensagens destinadas ao domínio *dominio.com.br*. Você está acessando o servidor Web da Disney. Lá vai o DNS novamente buscar a tradução do nome *www.disney.com* para um número IP. Chat, FTP, e mil outras coisas

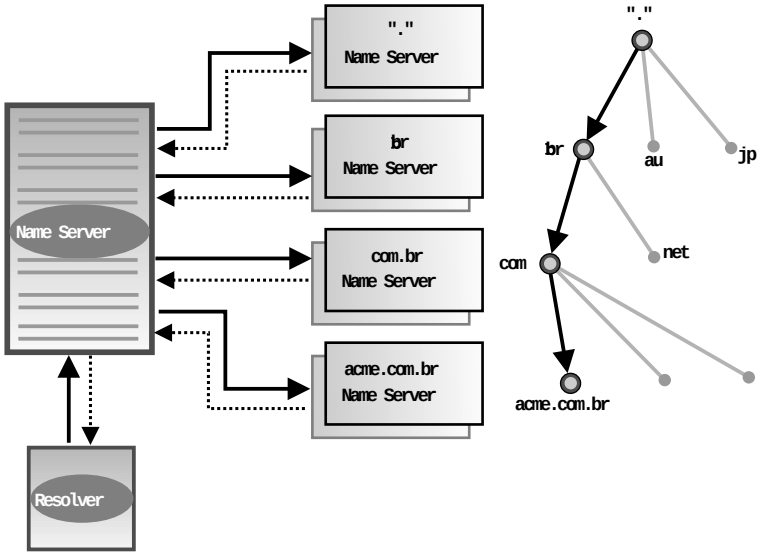

Figura 4.1: Estrutura de uma busca DNS

que você se habituou a usar na Internet, todas fazem uso do DNS.

Por agora você já deve ter visto a importância do DNS no uso dos recursos da Internet. Por isto mesmo não se esqueça que a sua configuração correta é muito importante. A não ser que você saiba de cor o número IP de todos os computadores da Internet, o que é pouco provável. Afinal de contas nem o seu computador consegue fazer isto e ele é muito melhor do que você para memorizar coisas.

Não se esqueça, uma grande parte das maravilhas da Internet se deve ao fato de que diversos servidores DNS estão silenciosamente fazendo o seu trabalho, nos unindo nesta maravilhosa comunidade virtual.

4.1 DNS - Root Name Servers

Para o correto funcionamento de um servidor DNS, é necessário que se atualize periodicamente o arquivo chamado *named.root*. Este arquivo está disponível, via ftp anônimo, em *rs.internic.net:/domain/named.root*.

Este arquivo contém a lista dos servidores de mais alto nível na Internet, o domínio ".", ou domínio raiz (root).

Caso esta lista esteja desatualizada os efeitos podem ser desastrosos e potencialmente difíceis de se identificar. Todo servidor DNS ao ser ativado precisa desta lista para então chegar aos servidores dos domínios mais baixos, como *.com* e *.edu*, e os domínios geográficos (*.br*, *.jp*, *.au*, etc.)

Esta tarefa pode ser facilmente automatizada a partir de uma *shell script*, como a que se segue

rootns.sh

```
#!/bin/bash
# A seguir, é  realizada  uma sessão de ftp em
# modo de  lote. Os  parâmetros  fornecidos ao
# programa  ftp  são  fornecidos  na  linha de
# comando  até  se chegar  ao  delimitador, no
# nosso caso, EOF.
ftp -ni rs.internic.net << EOF
user anonymous usuario@dominio.com.br
cd /domain
get named.root
EOF

# Uma  vez  recuperado  o  arquivo  compará-lo
# com a versão em uso. Se  diferentes, efetuar
# a  substituição e reiniciar o  servidor  DNS
# A  localização  do  arquivo em  uso pelo seu
```

```
# servidor  é  variável e  é  determinada pelo
# valor das diretivas
# directory     /var/named
# cache     .   named.root

# contidas no arquivo /etc/named.boot. Em nos-
# so exemplo estamos assumindo que os arquivos
# de configuração encontrem-se em
# /var/named

# No laço abaixo, caso o comando diff retorne um
# valor  diferente  de  zero,  serão  executados
# os comandos contidos na diretiva else.É envia-
# do também um mail para o  administrador  noti-
# ficando-o  que  houve uma alteração no arquivo
# named.root

if diff named.root \
   /var/named/named.root \
       1> /dev/null 2> /dev/null
 then
# Tudo normal, sai da shell
   exit
 else
# os arquivos são diferentes, executar a subs-
# tituição,  reiniciar  o processo named e en-
# viar uma mensagem à equipe de suporte.

cp named.root /var/named/named.root
# ir ao diretório /etc/rc.d/init.d
cds
# reiniciar o processo named
./named restart
mail -s "Named substituído" \
```

```
     suporte@acme.com.br < /dev/null
fi
```

4.2 Endereços IP

Às vezes existe a necessidade de se fazer com que a tradução de um nome forneça endereços IP diferentes.

Um site Web que receba milhões de visitantes diariamente possivelmente não aguentaria uma carga tão grande. Uma solução é fazer com que cada usuário que faça uma consulta receba um número IP diferente. Desta forma a carga é distribuída por vários servidores, tornando o serviço mais rápido e eficiente.

Esta estratégia é conhecida como *round-robin*.

Os servidores *bind* mais recentes, versão superior a 4.9, já implementam esta funcionalidade nativamente. Veja este exemplo, de um domínio fictício chamado *www.acme.com.br*, cujo conteúdo tenha sido replicado em quatro servidores distintos:

```
% nslookup www.acme.com.br
Server:  localhost
Address:  127.0.0.1

Name:    www.acme.com.br
Addresses:  200.200.20.1, 200.200.20.2, 200.200.20.3

% nslookup www.acme.com.br
Server:  localhost
Address:  127.0.0.1

Name:    www.acme.com.br
Addresses:  200.200.20.2, 200.200.20.3, 200.200.20.1
```

Na primeira invocação do comando, o primeiro endereço fornecido foi *200*.

200.20.1, na segunda *200.200.20.2*, e assim por diante. O servidor DNS faz uma rotação dos endereços, ou seja, a cada solicitação será fornecido um endereço IP diferente.

Os registros DNS correspondentes para se obter esta funcionalidade são:

```
@     IN     SOA       ns.acme.com.br. suporte.acme.com.br. (
             0001      ; Serial
             3600      ; Refresh
             300       ; Retry
             3600000   ; Expire
             3600 )    ; Minimum
                    IN      NS      ns.acme.com.br.
www.acme.com.br.    IN      A       200.200.20.1
                    IN      A       200.200.20.2
                    IN      A       200.200.20.3
```

Como se pode ver acima, a um mesmo nome, *www.acme.com.br*, foram atribuídos quatro números IP distintos.

Capítulo 5

Um Breve Histórico da Internet

Durante sua existência a Internet sofreu muitas mutações, sempre se adaptando a novas realidades. Mudou o perfil de seus usuários, mudaram as características dos computadores a ela ligados, a velocidade das redes, aplicativos, enfim, praticamente tudo.

E para infelicidade de todos aqueles que previam o fim da grande rede mundial, a Internet continua cada vez mais firme e passando a invadir (ou ser convidada) à intimidade de cada vez mais empresas, lares, escolas, universidades e muitos outros locais. Hoje pode-se encontrar computadores ligados à Internet em praticamente todos os lugares.

Uma revolução deste porte, que tem em sua essência a comunicação, tem alterado fortemente o nosso estilo de vida. O modo como pensamos, trabalhamos, e vivemos, estão sendo alterados com uma velocidade nunca vista.

Esta alteração se dá pela incrível sinergia de milhões de pessoas utilizando um meio comum de comunicação, a Internet. Novos conhecimentos, novas tecnologias são criadas e postas à disposição de quem delas precisa em uma velocidade nunca vista. A informação já existente é continuamente trabalhada e aperfeiçoada por pessoas espalhadas por todo o mundo, 24

horas por dia, 7 dias por semana.

Originalmente, antes da sua extensa popularização, iniciada em 1993 com a criação do primeiro browser web, a utilização eficiente da Internet exigia o conhecimento de vários programas diferentes (*ftp*, *gopher*, *telnet* e vários outros). Além de conhecê-los, era necessário também conhecer onde a informação se encontrava. Existiam algumas facilidades para busca de informação, mas nada comparado aos mecanismos de busca hoje existentes. E a informação disponível era em sua maioria composta apenas por texto, sem imagens e sons.

O primeiro browser Web, o Mosaic, veio mudar radicalmente esta situação. O acesso à informação disponível na Internet ficou ao alcance de praticamente todos, mesmo aqueles com pouca cultura em informática.

A informação passou a ficar disponível de uma maneira simples e intuitiva. A transição entre um computador e outro passou a se dar de forma totalmente transparente para o usuário. A Internet deixou de ser um reduto dos iniciados, dos experts em informática.

A revolução criada pelo Mosaic, se deu pela possibilidade, até então inexistente, de se integrar imagens aos documentos e pela implementação do formato hipertexto. Em documentos hipertexto nós temos informações ligadas, ou seja, o documento deixa de ser linear. A leitura não mais necessita ser feita do começo ao fim. O documento se abre lateralmente, permitindo uma leitura por associações. Através de um documento, em tese, tem-se acesso a toda a informação existente na Web. É o documento sem fronteiras.

O browser Web na verdade é apenas um componente de um sistema de informações mais amplo organizado segundo o protocolo chamado HTTP ou *Hyper Text Transport Protocol*. Este protocolo foi criado, em 1990, por Tim Berners Lee, que trabalhava então no CERN, na Suiça.

Como se vê, o protocolo HTTP já existia há algum tempo e era muito pouco utilizado. Um outro sistema de informações, chamado Gopher, era então a estrela da Internet. A informação era estruturada hierarquicamente, de forma semelhante à estrutura de diretórios de microcomputadores.

Qualquer instituição ligada à Internet que não possuísse o seu servidor Gopher estava condenada ao esquecimento. Após o surgimento do Mosaic, a maioria dos servidores Gopher foi gradualmente substituída por servidores Web e a grande teia mundial começou a se formar. Esta popularização imediata da Web se deu principalmente por duas razões. A primeira delas foi a facilidade de integração entre diversos servidores de informação propiciada pelo protocolo HTTP associada à facilidade de uso do programa Mosaic e da integração de imagens aos documentos. O segundo fator, não menos importante, foi a disponibilização gratuita do código fonte, tanto do servidor HTTP quanto do browser Mosaic.

Desta forma, apareceram versões de ambos os programas para praticamente qualquer tipo de computador existente. A partir de então, o número de usuários, e paralelamente, a quantidade de informação disponível na Internet apresentaram taxas de crescimento nunca vistas.

Com este crescimento apareceram alguns problemas, o mais grave deles sendo justamente a questão da organização e acesso à informação. A Internet passou a ser então o equivalente a uma biblioteca, imensa, sem fichas catalográficas. Um perfeito caos. Da mesma forma que o valor de uma biblioteca está diretamente relacionado ao índice que lista seus livros, o valor da Web é estreitamente dependente dos mecanismos de pesquisa que a servem. Algo precisava ser feito urgentemente. E foi. Como em outras ocasiões, a Internet se adaptou. Se o problema é achar a informação, que se criem então ferramentas de busca que coletem o conhecimento armazenado na Web e o organizem de forma a ser facilmente consultado.

O primeiro mecanismo de busca, *Yahoo!*, apareceu já em 1994. O sítio Altavista, patrocinado pela empresa Digital, surgiu em 1995, juntamente com o Excite e Infoseek. Em 1996 foram criados os sítios *HotBot* e *LookSmart*.

A tarefa de indexação da Web, entretanto, não é tarefa das mais simples. Em seguida ao deslumbramento inicial, de ter a informação disponível facilmente, os usuários sofreram alguns desapontamentos. O primeiro deles, a informação chegava em grandes quantidades e nem sempre o que se obtinha era o que se desejava. E os mecanismos de busca tiveram que se adaptar

à esta nova realidade. Esta é uma luta que não tem fim. Cresce a quantidade de informação na Internet, cresce o número daqueles que tentam, de forma honesta ou fraudulenta, obter as primeiras posições nas listagens dos mecanismos de busca.

É hoje essencial que os profissionais de qualquer área consigam compreender os mecanismos de funcionamento destes serviços de busca. Entender as peculiaridades de cada um deles, as novas tendências em tecnologia de informação, o que está acontecendo de novo nesta área é vital ao exercício competente de qualquer profissão.

Capítulo 6

Correio Eletrônico: Uso da Auto-Resposta

O correio eletrônico nos oferece inúmeras facilidades: podemos colocar programas para responder as mensagens por nós, podemos arquivar mensagens não prioritárias em arquivos separados para leitura posterior, apagar automaticamente mensagens de correspondentes indesejados e muito mais. A lista de facilidades é quase infinita.

Uma destas muitas facilidades é a resposta automática. Nos primórdios da Internet a resposta automática normalmente era utilizada quando a pessoa saia de férias e servia para alertar seus correspondentes da data de saída, data de retorno e o que fazer neste meio tempo em caso de urgência. Com a popularização da Internet, o correio eletrônico passou a ser utilizado por um grande número de pessoas , a maioria das quais desconhece a chamada "etiqueta" da Internet. Por etiqueta devemos entender o modo apropriado de se comportar. O correio eletrônico, como muitas aplicações Internet, também tem a sua etiqueta, alguns pontos das quais serão discutidos aqui.

Primeiramente, nunca ative a auto-resposta para mensagens comuns ou mensagens enviadas através de listas eletrônicas. Estas auto-resposta são um aborrecimento para o mantenedor das listas, que tem que apagar cen-

tenas de mensagens diariamente. Algumas listas chegam mesmo a proibir a inscrição de pessoas que tenham a auto-resposta ativada. Pense bem, o mundo virtual não é tão diferente assim do mundo real.

Se você assina uma revista, você liga para a editora todos os meses avisando que o seu exemplar já chegou? Certamente que não. E porque não? Porque enviar cartas no mundo real envolve gastar dinheiro com envelope, selo, caminhar até a agência dos correios, esperar na fila e por aí vai. Já no correio eletrônico é diferente. É muito fácil enviar mensagens. Daí o abuso. Envia-se mensagem para todos e por qualquer razão. Você também não escreve uma carta para alguém falando que recebeu a carta e que vai responder em breve. Ninguém, no mundo real, envia duas cartas para responder uma.

Em segundo lugar, se você precisa mesmo usar a auto-resposta (pense de novo, precisa mesmo?), ative esta opção apenas se sair de férias ou se ausentar do trabalho ou de sua casa por um período prolongado, nunca inferior a quinze dias. Antes de ativar a auto-resposta, descadastre-se de todas as listas que assina para evitar transtornos aos assinantes e mantenedores destas listas.

Não se lembra quais listas assina? Então passe a adotar como regra salvar em uma pasta as mensagens de boas vindas das listas que assinar. Estas mensagens geralmente contém o objetivo da lista e fornecem instruções para realizar o descadastramento. Evite aborrecer (ou insultar) o mantenedor da lista com pedidos de descadastramento. A maior parte destas pessoas realizam seu trabalho voluntariamente e no mínimo devem ser tratados com respeito e consideração.

Mesmo quando sair de férias o correto é redirecionar as suas mensagens para que um companheiro de trabalho as leia e tome as devidas providências. Para que isto possa ser feito nunca forneça seu endereço eletrônico de trabalho para fins particulares. Utilize um endereço separado para uso pessoal. A Internet possui dezenas de serviços que oferecem endereços eletrônicos gratuitamente. Se a mensagem for urgente e requerer ação imediata da parte de alguém é muito melhor que alguém leia, não é mesmo? Ou você é do tipo insubstituível?

E finalmente, já pensou na situação em que você recebe uma mensagem de alguém que também tem auto-resposta ativada? Vai começar um ping pong na Internet com auto resposta indo e vindo. Se você é do tipo que reclama que a Internet é lenta e usa auto-resposta, saiba que a culpa também é sua. Colabore para diminuir a congestão da Internet, não use auto-resposta.

Resumindo, as regras são simples. Não se comporte no mundo virtual de forma diferente do mundo real. O mundo virtual foi em grande parte concebido baseando-se na forma como a vida é conduzida normalmente. Só use a auto-resposta se absolutamente necessário e tente evitá-la a todo custo. Você estará fazendo um grande favor a si mesmo, aos mantenedores de diversas listas de discussão de ótima qualidade e à própria Internet, que certamente ficará mais rápida se não tiver que carregar para um lado e outro centenas de milhares de mensagens inúteis.

Capítulo 7

A Reconstrução da Torre de Babel

Diz a Bíblia que há muitos anos atrás todos os habitantes da Terra se uniram para construir uma torre que chegasse até o céu, para tornar seu nome célebre e impedir que fossem espalhados pelo mundo. Para punir os homens por sua ambição demasiada, Deus confundiu sua linguagem e depois os dispersou pelo mundo.

Ainda hoje os povos da Terra falam uma imensidão de línguas diferentes. Na Internet entretanto, apesar dos muitos povos que a utilizam, existe um meio de comunicação comum. Da mesma forma que os computadores se comunicam independentemente de cor e raça, ou melhor, de fabricante e protocolo de comunicação, também os internautas possuem uma linguagem comum: a língua inglesa. Será a Internet uma nova Torre de Babel, construída para reunificar eletronicamente os habitantes deste lindo mundo azul?

É claro que nem todos que utilizam a Internet compreendem a língua inglesa. Porém mais de 80% dos documentos e das comunicações feitas através da Internet encontram-se em inglês. Apenas 0,7% do oceano de informação que é a Internet está em português. É perfeitamente possível usar a Internet

e se divertir muito navegando apenas por sites escritos em português. Fazer isto entretanto é o equivalente a ir à praia, não entrar na água e ficar se molhando com um baldinho de água que alguém encher para você.

O que fazer? Aprender inglês é difícil e demora muitos anos. Como então adquirir o domínio desta ferramenta tão essencial à utilização plena da Internet? Realmente, para se ler, falar, escrever e ouvir com fluência a língua inglesa são necessários de seis a oito anos de estudo constante. Por quê aprender tanta coisa se o mais importante é apenas ler? É muito mais fácil dominar um dos aspectos de um idioma (leitura) do que todos os quatro simultaneamente (leitura, audição, fala e escrita). A Internet possui muito conteúdo interativo, onde a capacidade de se falar e escrever bem a língua inglesa certamente é uma grande vantagem, mas o mais importante certamente é saber ler. Ler para utilizar a informação existente na Internet para aprender, resolver problemas pessoais ou profissionais, se divertir, enfim, para uma infinidade de propósitos.

Como aprender a ler? É raro encontrar um curso de inglês onde se ensine o aluno apenas a ler. Só vendem o "pacote" completo, o que é totalmente insensato. Se precisamos investir vários anos para dominar o idioma em todos os seus aspectos, aprender a ler certamente demora muito menos. Em apenas quatro meses é possível obter uma compreensão razoável do idioma que nos permite começar a compreender textos em inglês.

Mas porque a leitura é a habilidade mais fácil de se dominar? A própria Internet nos ajuda a encontrar a resposta. Em um estudo de 1997, realizamos um trabalho para determinar as palavras mais comuns da língua inglesa e seu percentual de ocorrência. Para este estudo utilizamos os livros online do Projeto Gutemberg[1]. Este projeto, integrado por voluntários, tem por objetivo digitalizar obras de literatura cujos direitos autorais tenham expirado. Nos Estados Unidos uma obra é colocada no domínio público 60 anos após a morte do autor. Obras de autores como Jane Austen, Conan Doyle, Edgar Rice Burroughs, e muitos outros estão disponíveis gratuitamente na Internet.

[1] *http://www.promo.net/pg*

De posse destes livros, 1600 ao todo na época da pesquisa, fizemos então nossos cálculos. Os 1600 livros combinados geraram um arquivo de 680 MB contendo aproximadamente sete milhões de palavras. Os resultados foram bastante surpreendentes. As 250 palavras mais comuns compõem cerca de 60% de qualquer texto. Em outras palavras, se você conhece as 250 palavras mais comuns, 60% de qualquer texto em inglês é composto de palavras familiares. Para facilitar ainda mais a nossa tarefa os cognatos, que são as palavras parecidas em ambos os idiomas (*possible* e *possível*, por exemplo), bastante comuns na literatura técnica, totalizam entre 20 a 25% do total das palavras. Temos então de 80 a 85% do problema de vocabulário resolvido. Se subirmos o número de palavras mais comuns a 1.000, chegamos a 70%. Somando a este valor os cognatos chegamos a valores, para textos técnicos, entre 90 e 95% do todo.

É claro que 90 ou 95% ainda não chega a 100%. Como fazer com o restante das palavras? Mais uma vez, usamos nossa intuição (se você não sabe, a nossa intuição está correta em 99,999% das vezes, basta confiar nela). Pensemos em nosso texto como um enigma a ser desvendado. Possuímos alguns elementos familiares, as palavras que conhecemos, e outros que nos são desconhecidos. Devemos deduzir, por meio de nossa intuição, de nossos conhecimentos anteriores, o que as palavras desconhecidas podem significar. Não precisamos nos preocupar com todas as palavras, mas apenas com aquelas que desempenhem um papel importante no texto. Quais são elas? Se uma palavra aparece com relativa freqüência em um texto, ela certamente desempenha um papel importante na compreensão do todo. Se uma palavra aparece apenas uma vez, muito provavelmente não precisaremos nos preocupar com ela.

O maior problema é que tal enfoque é encarado de forma suspeita pela maioria das pessoas. Como é possível, ignorar uma palavra desconhecida e continuar lendo como se nada houvesse acontecido? O que estamos propondo não é nada absurdo. Qual foi a última vez em que consultou um dicionário? Toda vez que encontramos uma palavra desconhecida vamos em busca do dicionário? Muito provavelmente não. O que acontece é que, como a nossa familiaridade com o português é grande, na hipótese de depararmo-nos com

uma palavra desconhecida, o seu sentido, dado o contexto que a cerca, será facilmente deduzido. Isto tudo praticamente sem mesmo nos darmos conta do ocorrido.

A não ser que nos proponhamos a tarefa de parar a cada vez que encontrarmos uma palavra desconhecida, a nossa leitura se dá com freqüência sem interrupções. As palavras desconhecidas são intuídas, quase que subconscientemente, e passam a integrar o nosso vocabulário. Considerando-se que o vocabulário de um adulto consiste de aproximadamente 50.000 palavras, é ridículo imaginar que tal conhecimento tenha sido adquirido através de 50.000 visitas ao dicionário. Este vocabulário foi adquirido, em um processo iniciado em nossa infância, de forma contínua e através da observação do nosso ambiente, observando outras pessoas falarem, prestando atenção nas palavras utilizadas em determinadas situações e também através da leitura.

A nossa estratégia para o domínio da língua inglesa para leitura é exatamente aquela utilizada há milhares de anos, com excelentes resultados, pela raça humana: aprendizado natural, seguindo nossos instintos e interagindo com o ambiente que nos cerca.

Como vimos, o domínio das palavras mais freqüentes da língua inglesa, pode nos ajudar a dar um impulso substancial em nosso aprendizado. Nós criamos uma listagem, disponível na Internet[2], e também no CDROM que acompanha este livro, contendo as 750 palavras mais comuns da língua inglesa. Nesta listagem as palavras não estão organizadas alfabeticamente, mesmo porque não é nosso objetivo reproduzir um dicionário. Também não incluímos todos os significados possíveis das palavras apresentadas. Todas as palavras são apresentadas em um contexto, com exemplos de utilização. Não fornecemos a definição da palavra. Para cada palavra são listados em média três exemplos de utilização, com a respectiva tradução.

É muito importante ressaltar que estas palavras não devem ser memorizadas de forma alguma. O ser humano não funciona de forma semelhante ao computador, onde as informações podem ser armazenadas de qualquer forma, e ainda assim estão disponíveis em milésimos de segundos quando

[2] *http://www.Dicas-L.unicamp.br/dict.pdf*

necessitamos. O ser humano, para reter alguma informação, precisa situá-la dentro de um referencial de conhecimentos. A informação nova precisa se integrar à nossa visão do mundo, à nossa experiência prévia. Apenas desta forma podemos esperar que o conhecimento adquirido seja duradouro. A maioria de nós certamente já vivenciou situações em que dados memorizados desapareceram de nossa memória quando não mais necessários. Ao contrário, tudo que aprendemos ativamente, permanece presente em nossa memória de forma vívida por muitos e muitos anos.

Embora esteja sendo fornecida uma lista de palavras, não adote de forma alguma o procedimento padrão de memorização, que é a repetição intensiva dos ítens a serem memorizados. É certo que cada um de nós possui estratégias distintas para lidar com o aprendizado, mas eu gostaria de sugerir uma forma de estudo que certamente funciona.

Primeiramente, não tenha pressa. Não memorize, procure entender os exemplos. Para cada palavra apresentada, leia os exemplos e suas respectivas traduções. Não se preocupe em reter na memória o formato exato das frases e nem de sua tradução. O objetivo é apenas compreender o significado da palavra apresentada e apenas isto. Uma vez compreendido este significado o objetivo foi alcançado.

Em segundo lugar, procure ler apenas enquanto estiver interessado. Não adianta nada ler todas as palavras de uma vez e esquecer tudo dez minutos depois. Se nos forçarmos a executar uma atividade monótona por muito tempo, depois de alguns momentos a nossa atenção se dispersa e nada do que lemos é aproveitado. Eu sugiro a leitura de dez palavras diariamente. Caso você ache que 10 palavras diárias é muito, não tem importância, este número é sua decisão. Se quiser ler apenas uma palavra, o efeito é o mesmo. Irá demorar um pouco mais, mas chegar ao final é o que importa. É só não esquecer, você deve LER as palavras e NUNCA tentar memorizar as palavras e os exemplos.

E finalmente, faça revisão. No primeiro dia leia e entenda dez palavras (ou quantas julgar conveniente). No segundo dia leia mais dez palavras e faça a revisão das dez palavras aprendidas no dia anterior. No terceiro

dia, aprenda mais dez palavras e revise as vinte palavras aprendidas nos dias anteriores. E assim por diante até o último dia, onde aprenderá as últimas dez palavras e revisará as 240 palavras anteriores. Muito importante, por revisão não quero dizer que se deve fazer a leitura de todas as palavras e exemplos anteriores. As palavras mais freqüentes estão grafadas em tipo diferente e em negrito, para que possamos localizá-las facilmente na página. Apenas examine as palavras anteriores em sua revisão. Caso não se recorde de seu significado, então, e apenas então, leia os exemplos. A revisão é extremamente importante. Nós realmente aprendemos quando revisamos conceitos aos quais já fomos expostos. Procedendo desta forma, tenha certeza de que tudo o que aprendeu será absorvido de forma permanente, constituindo a base fundamental de tudo que irá aprender em seus estudos da língua inglesa.

Caso a sua motivação seja realmente alta e você queira reler todos os exemplos já estudados, vá em frente. Como você pode notar, os exemplos empregam um vocabulário bastante rico. A leitura mais freqüente dos exemplos fará com que ao final do estudo o seu vocabulário tenha se enriquecido muito além das 750 palavras básicas.

Outro ponto importante é a questão do estudo da gramática. A gramática, ou o estudo da estrutura da língua, deve ser apenas para ajudar o aluno a identificar as construções verbais. Não é necessária, para fins de aprendizado da leitura, a memorização de estruturas gramaticais. Como já afirmado, o nosso aprendizado se dá de forma natural. Da mesma forma que uma criança não tem aulas de gramática para aprender sua língua materna, nós também não devemos nos preocupar com este aspecto em nosso estudo. A leitura dos exemplos das palavras mais comuns irá lançar os fundamentos iniciais do conhecimento da estrutura da língua inglesa.

Resta agora esclarecer um ponto, que é a desculpa favorita de todos nós nos dias de hoje: a falta de tempo. Tempo certamente é fácil de se encontrar para fazer aquilo que nos dá prazer. Para resolver o problema de tempo para este estudo, pense nesta atividade como algo prazeroso e que lhe trará benefícios enormes, tanto no campo pessoal como profissional. E além do

mais, o aprendizado e a revisão das palavras pode ser feito diariamente em não mais de quinze minutos. Se levarmos em conta que os intervalos comerciais em programas de televisão geralmente duram entre quatro a cinco minutos, todo o tempo necessário para este estudo pode ser encaixado nos intervalos de sua novela favorita, certo?

Então, mãos a obra. Depois que você conhecer as 250 palavras mais comuns da língua inglesa, poderá verificar como o aprendizado da leitura da língua inglesa se torna muito mais fácil. Na lista das palavras mais comuns foram incluídas 750 palavras. Faça um esforço e tente conhecer todas elas. A sua tarefa vai ficar ainda mais fácil.

Além do aprendizado das palavras mais comuns, o interessado em aprender o inglês para leitura, deve procurar intensificar o seu contato diário com a língua inglesa. Para isso a Internet pode novamente vir em nosso auxílio. Basta procurar nela pelo que nos interessa. Na Internet existe informação de todos os tipos e para todos os gostos. Basta saber e querer procurar. Um destes recursos é a lista eletrônica EFR *(English for Reading)*. Esta lista está hospedada no serviço *eGroups*[3]. Nesta lista é veiculada diariamente uma história, preferencialmente engraçada (afinal, quem não gosta de uma boa piada?) ou uma citação. As histórias são em inglês e as palavras mais incomuns são comentadas. Desta forma os alunos aprendem todos os dias duas ou mais palavras novas. Todos os dias. Em um ano este pequeno esforço diário pode vir a fazer uma grande diferença. O cadastramento nesta lista é aberto a todos os interessados, bastando enviar uma mensagem vazia para o endereço *efr2-subscribe@eGroups.com*.

Quanto mais nos habituarmos a ler em inglês mais fácilmente aprenderemos o idioma, o que não é novidade nenhuma. Como vivemos no Brasil, país de língua portuguesa, as nossas necessidades de utilizar outra habilidade que não a leitura em inglês são bastante esporádicas. Mas não precisamos parar por aí. A leitura serve também para desenvolver as outras habilidades necessárias ao domínio da língua inglesa: a fala, a escrita e a compreensão da língua falada. O principal é que em um período de tempo bastante curto

[3] *http://www.eGroups.com*

já estaremos habilitados a navegar pela Internet inteira e não apenas pela pequena porção representada pela língua portuguesa.

Finalmente, queria lembrar a todos que aprender o inglês é bastante fácil. Basta deixar de lado os preconceitos e traumas que temos com a língua inglesa e realmente acreditarmos em nossa capacidade de aprender. Não leva a nada guardar rancores de tentativas frustradas de aprendizado ocorridas no passado. O domínio da língua inglesa é hoje o nosso passaporte para um mundo de informações que podem nos ser úteis tanto na esfera pessoal quanto profissional. Se você não domina a língua inglesa o momento certo para começar é hoje. Consulte as referências deste capítulo, estude com calma a lista das palavras mais comuns e assine a lista EFR. Você vai ver que sem fazer muita força em, pouco você estará se locomovendo com desenvoltura cada vez maior pela Torre de Babel reconstruída que é a Internet. Depois me escreva contando os resultados.

Referências

- *Global Internet Statistics (by Language)* [4]

- *Projeto Gutemberg* [5]
  Milhares de obras digitalizadas da literatura universal, em formato texto e disponíveis para download gratuitamente. Imperdível.

- *Dicas-L* [6]

- *Palavras mais Comuns da Língua Inglesa* [7]
  Este documento descreve os procedimentos utilizados para se determinar as 1000 palavras mais comuns da língua inglesa e faz uma apresentação dos resultados obtidos. Neste documento as palavras mais comuns são listadas juntamente com seu percentual de ocorrência.

[4] *http://www.euromktg.com/globstats/*
[5] *http://www.promo.net/pg/*
[6] *http://www.Dicas-L.unicamp.br*
[7] *http://www.dicas-l.unicamp.br/dicas-l/19971002.shtml*

- *As 750 palavras mais comuns da Língua Inglesa* [8]
  Este documento, no formato PDF, contém as 750 palavras mais comuns da língua inglesa com exemplos de utilização. Para ler e imprimir arquivos no formato PDF é necessário instalar em seu computador o programa *Adobe Acrobat Reader*, que pode ser encontrado no endereço *http://www.adobe.com*. Além do *Acrobat Reader*, usuários *Conectiva Linux* dispõem também dos softwares *xpdf* e *Ghostview (gv)*.

- *As 1000 Palavras Mais Comuns da Língua Inglesa* [9]
  Este artigo lista as 1000 palavras mais comuns da língua inglesa e seu percentual de ocorrência.

Notas

- A comparação da Torre de Babel com a Internet eu encontrei em um artigo escrito por Luiz de Rezende Puech. Este artigo infelizmente não mais se encontra neste universo volátil de informações que é a Internet. Esta analogia, bastante criativa, nunca mais me saiu da cabeça e aproveitei este artigo para abordar o assunto a partir de uma ótica diferente.

- A história da Torre de Babel encontra-se na Bíblia, no livro de Genesis, capítulo 11. Reproduzo a seguir alguns dos versículos da história, tal como se encontra na Bíblia.

  - (Genesis 11:6) e o SENHOR disse: Eis que o povo é um, e todos têm a mesma linguagem. Isto é apenas o começo; agora não haverá restrição para tudo que intentam fazer.
  - (Genesis 11:7) Vinde, desçamos e confundamos ali a sua linguagem, para que um não entenda a linguagem de outro.
  - (Genesis 11:8) Destarte, o SENHOR os dispersou dali pela superfície da terra; e cessaram de edificar a cidade.

[8] *http://www.Dicas-L.unicamp.br/dict.pdf*
[9] *http://www.dicas-l.unicamp.br/dicas-l/dicas-l/971003.html*

– (Genesis 11:9) Chamou-se-lhe, por isso, o nome de Babel, porque ali confundiu o SENHOR a linguagem de toda a terra e dali o SENHOR os dispersou por toda a superfície dela.

Capítulo 8

Netscape

8.1 Configuração de Aplicações Auxiliares (Helpers)

Um dos maiores inconvenientes de quem usa sistemas Linux para seu trabalho diário é o recebimento de documentos anexados gravados no formato Microsoft Word (extensão doc).

Existem duas alternativas, responder ao remetente para enviar o documento num formato mais aceitável, como texto puro, que pode ser lido em qualquer computador e sistema operacional existente na face da Terra, ou então fazer uma conversão para o formato html, por exemplo.

O aplicativo *Netscape Messenger* pode ser configurado para realizar todas estas tarefas automaticamente, a saber, a conversão do documento do formato MS Word para HTML e exibir o documento convertido na tela do browser, a partir de onde pode ser salvo com outro nome ou o que for mais conveniente.

O programa que faz a conversão de arquivos no formato MS Word para HTML costumava se chamar *mswordview*. O autor resolveu entretanto

renomear o produto para simplesmente *wv*[1], visto que o nome original, *mswordview*, era bastante semelhante ao nome de um produto da Microsoft chamado *wordview*. O produto é distribuído sob a licença GPL, ou seja, qualquer pessoa pode se utilizar dele livremente. *wv* é uma biblioteca que permite acesso a arquivos gerados pelo software Microsoft Word, nos formatos Word 2000, 97, 95 e 6, conhecidos internamente como word 9, 8, 7 e 6.

O programa é excelente e são raríssimos os casos em que não consegue converter corretamente os documentos. O software compila facilmente em ambientes Linux, bastando seguir as instruções contidas na distribuição original. A sua utilização é também bastante simples, bastando digitar:

```
$ wvHtml arquivo.doc
```

Será gerado então um arquivo de igual nome porém com extensão *.html* (em nosso caso, *arquivo.doc.html)*. O nome do arquivo gerado pode ser alterado, através da especificação do sinal "-o":

```
wvHtml arquivo.doc -o arquivo.html
```

A utilização do software é extremamente simples, como se pode ver dos exemplos acima. Entretanto existe um grande número de opções de utilização que podem ser explorados. Para maiores detalhes ler a documentação do programa.

Resta agora criar uma shell que faça automaticamente a tradução de formatos e exiba o resultado em seu browser Web:

msw

```
#!/bin/bash
# msw - Tradução de arquivos MS Word para HTML
```

[1] *http://www.wvWare.com*

```
#
wvHtml $1 > /tmp/tmp.html
Netscape -remote 'openURL(file:/tmp/tmp.html)'
```

A segunda linha invoca o programa *Netscape* com a opção *-remote*. Esta opção sinaliza ao *Netscape* para realizar a ação solicitada utilizando-se uma sessão existente. Ou seja, se o browser *Netscape* já se encontra aberto em seu ambiente de trabalho, a tela solicitada será aberta em uma das telas já existentes. A ação solicitada em nosso caso é a leitura do arquivo criado pelo programa *wvHtml*, */tmp/tmp.html*.

Criada a shell, resta agora configurar o browser para fazer uso deste recurso. Os passos descritos a seguir aplicam-se às versões 4.x do programa *Netscape*.

Primeiramente selecionar, no menu *Edit*, o submenu *Preferences*. Selecionar então, na opção *Communicator*, a opção *Applications*. Procurar então na tela da direita uma entrada para *Microsoft Word Document*. Selecionar esta entrada com o mouse e em seguida clicar no botão *Edit*. Veja a figura 8.1.

Na tela que se segue, na parte inferior, selecionar o botão *Aplicações* e escrever no campo apropriado:

```
msw %s 2>&1 >/dev/null
```

No comando acima eventuais mensagens de erro geradas serão descartadas (enviadas para o dispositivo */dev/null)*. Caso se opte por receber as mensagens de erro basta remover o redirecionamento (*2>&1 >/dev/null*). Veja a figura 8.2.

A shell *msw*, criada anteriormente, deve estar em algum diretório que esteja definido na variável de ambiente *PATH*, para que possa ser encontrada pelo *Netscape*. O parâmetro *%s* é interpretado pelo browser como uma informação a ser passada ao programa especificado, *msw*. Desta forma, sempre que solicitarmos ao nosso browser a abertura de um documento com extensão *.doc*, através do *Netscape Messenger* ou do próprio browser, a ação tomada será a execução da shell *msw* que receberá como parâmetro o nome do

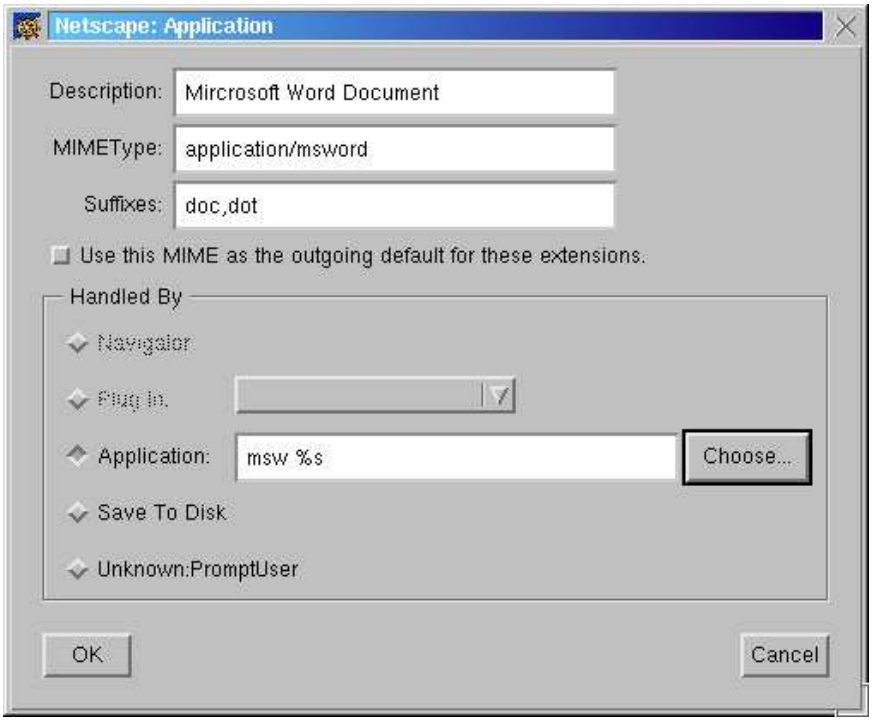

Figura 8.1: Netscape: Inserindo um novo tipo de aplicação

arquivo selecionado.

Isto feito, aceitar as modificações, clicando nos botões "OK" e pronto. Daqui para frente o seu browser Web está capacitado a traduzir automaticamente documentos nos vários formatos do aplicativo MS Word, tanto aqueles que vierem anexados a mensagens eletrônicas quanto aqueles que estiverem em seu próprio sistema, quando estiver utilizando o seu browser como um gerenciador de arquivos.

Na hipótese de se desejar realmente salvar um arquivo anexado no formato original, basta clicar com o mouse sobre o nome do arquivo pressionando-se simultaneamente a tecla SHIFT. O procedimento padrão será a conversão e exibição na tela do browser.

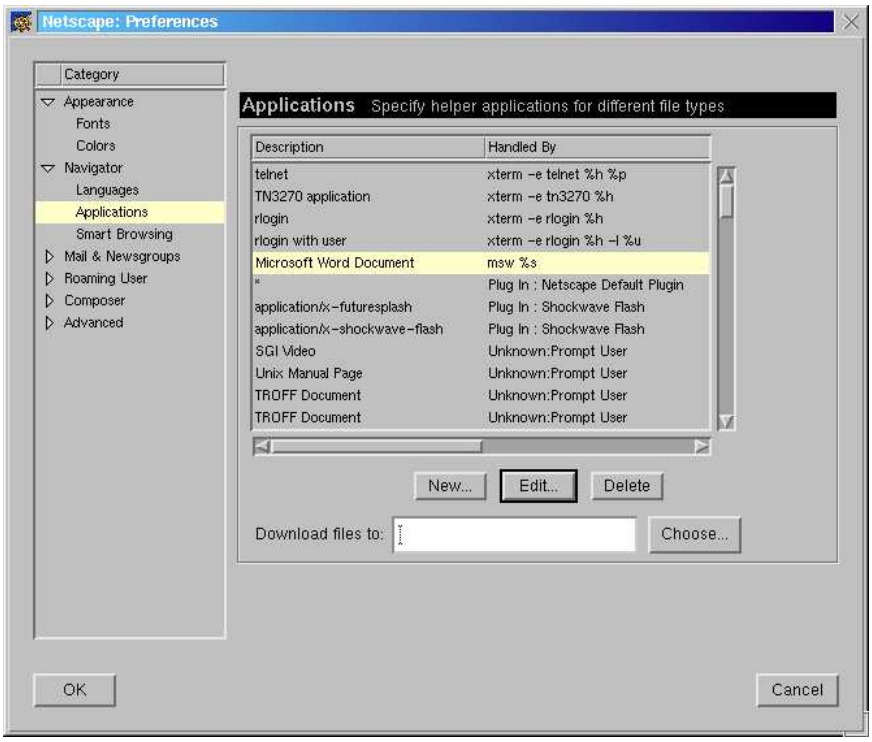

Figura 8.2: Netscape: Visualizando as configurações existentes

O pacote *wv* traz também um outro utilitário que realiza a conversão de arquivos em formato MS Word para texto puro, chamado *wvText*. O formato texto é ideal para se portar documentos entre plataformas e aplicativos diferentes.

Neste capítulo abordou-se o uso destes programas a partir do aplicativo *Netscape Communicator*, porém uma grande variedade de aplicações é possível, como a conversão em massa de documentos MS Word para seu equivalente no formato texto ou *html*, invocando-se os aplicativos diretamente a partir da linha de comandos ou através de shell scripts. Uma shell como abaixo pode ser utilizada para tal tarefa:

doc2txt.sh

```
#!/bin/bash
# conversão de arquivos no formato word
# (extensão .doc)  para html
# doc2txt.sh
for file
do
  wvHtml $file > ‘echo $file | sed ’s/doc$/html/’‘
done
```

Para invocar esta shell basta digitar *doc2txt.sh* (ou qualquer outro nome com o qual a shell tenha sido batizada) e fornecer como parâmetro o nome dos arquivos a serem convertidos:

```
$ doc2txt.sh *.doc
```

O sufixo *.doc* dos arquivos originais é substituído por *.html*. Se desejarmos converter para o formato texto puro basta substituir o comando *wvHtml* por *wvText*.

O software *xlHtml*[2] faz a conversão de planilhas MS Excel, nos formatos 95 e 97, para HTML, e pode ser configurado de forma similar ao *wvHtml*. Para isto basta editar, como explicado acima, o item *Microsoft Excel Worksheet* e definir a aplicação a ser invocada como

```
/usr/bin/nsxlview %s 2>&1 >/dev/null
```

Para maiores informações sobre configuração de *plugins* para o *Netscape*, consultar *The Solaris Helpers Page*[3]. Esta página contém diversas informações úteis sobre a configuração de aplicações auxiliares no browser *Netscape* em ambiente Solaris, que podem ser facilmente adaptáveis a qualquer tipo de sistema operacional semelhante ao Unix, como Linux e FreeBSD.

[2] *http://www.gate.net/ ddata/xlHtml/index.htm*
[3] *http://home1.swipnet.se/ w-10694/helpers.html*

8.2 Opções da linha de comando para o programa *Netscape*

O programa *Netscape* pode ser invocado fornecendo-se parâmetros na linha de comando que irão determinar a função a ser invocada. Por exemplo:

```
$ netscape -mail
```

irá fazer com que o *Netscape* inicialize apenas a janela para trabalho com correio eletrônico. Já o comando

```
$ netscape -news
```

criará uma janela para leitura de *Usenet News*.

Podemos fornecer também o nome de uma URL ou arquivo na linha de comandos. Desta forma o programa *Netscape*, ao completar sua carga, irá apresentar a página ou URL especificada na linha de comandos:

```
$ netscape listagem.html
```

ou

```
$ netscape http://www.conectiva.com.br
```

e ainda

```
$ netscape -edit
```

abre uma janela do editor de documentos html.

Além das opções da linha de comando descritas acima, o *Netscape* oferece também a possibilidade de se enviar instruções para uma cópia do *Netscape* já em execução.

O comando abaixo, por exemplo, irá instruir o *Netscape* já em execução em seu ambiente de trabalho a abrir a *home page* do servidor de ftp anônimo *ftp.dominio.com.br*.

```
$ netscape -remote 'openURL(http://ftp.dominio.com.br)'
```

Outra opção

```
$ netscape -remote 'addBookmark(http://ftp.dominio.com.br)'
```

irá acrescentar no arquivo de bookmarks o endereço *ftp.dominio.com.br*.

Além destes dois exemplos, são também suportadas várias outras diretivas. Nas diretivas abaixo, sempre que não se especificar o documento destino, o *Netscape* irá exibir uma janela de diálogo para tomar então a ação desejada.

- *openURL ( ), openURL (URL), openURL (URL,new-window)*
  Abre uma janela exibindo o documento solicitado. Caso não seja especificado o browser solicita qual documento se quer visualizar.
- *openFile (Arquivo), openFile ( )*
  Abre o arquivo especificado diretamente ou então solicita este valor do usuário. Lembre-se de fornecer o caminho completo. O caminho relativo sempre é calculado a partir do diretório a partir de onde o *Netscape* foi invocado.
- *saveAs, saveAs (Arquivo-de-Saida), saveAs (Arquivo-de-Saida,formato)*
  Salva o documento presente no formato especificado (HTML, texto ou postscript)
- *mailto(a, b, c), mailto ( )*
  Coloca os endereços entre parenteses no campo *To:*
- *addBookmark ( ), addBookmark (URL), addBookmark (URL,Título)*
  Adiciona o documento corrente à lista de bookmarks, podendo incluir, opcionalmente, o título que a bookmark receberá.

Estes recursos são implementados utilizando recursos do ambiente X Window, de maneira que não é necessário que o comando seja executado na mesma máquina onde o programa *Netscape* esteja em execução.

Capítulo 9

Linux e Windows NT – Comparação de Desempenho

Alguns tempos atrás resolvi realizar uma comparação entre sistemas *Conectiva Linux* e Windows NT. Fiz apenas uma comparação do desempenho da transferência de arquivos com o programa FTP. Não pude utilizar equipamentos idênticos, mas optei por uma configuração igualmente polêmica. Um ultrapassado 486 contra um Pentium 200 MMX.

A máquina rodando Linux é um 486DX/66, com 32 MB de memória, fabricado em 1993. O sistema operacional é o *Conectiva Linux*, versão do kernel 2.0.38. O MTU da interface ethernet do Linux é de 1500 bytes.

A estação NT é um Pentium 200/MMX, com 64 MB de memória, fabricado em 1998. O sistema operacional desta máquina é Windows NT server versão 4.0.

Realizei cinco transferências, usando FTP, em cada máquina, de um arquivo de 19MB de tamanho. As duas máquinas estão conectadas na mesma subrede, uma ethernet a 10Mbits/s. As transferências foram intercaladas. Primeiro o Linux, depois NT e assim sucessivamente. Nenhuma das máquinas estava rodando processos que pudessem interferir no seu desempenho.

Em todas as transferências o Linux ganhou. Considerando-se que as duas máquinas são brutalmente diferentes em termos de performance, dá para se notar que a implementação TCP/IP do Linux é consideravelmente melhor do que a do Windows NT server no tocante ao aspecto analisado, a saber, transferência de arquivos usando FTP.

Abaixo incluo as linhas de resultados das transferências nos dois sistemas. A média de tempo do Linux foi de 23,4 segundos, e a taxa média de transferência foi de 816 kbytes/segundo. No NT a taxa média de tempo foi de 29,51 segundos e a taxa de transferência foi de 664,60 kbytes/segundo. O Linux obteve um resultado 28% melhor no tempo de transferência em relação ao NT.

- Linux

```
1. 19529038 bytes received in 22.3 secs (8.5e+02 Kbytes/sec)
2. 19529038 bytes received in 22.6 secs (8.4e+02 Kbytes/sec)
3. 19529038 bytes received in 26.4 secs (7.2e+02 Kbytes/sec)
4. 19529038 bytes received in 23 secs (8.3e+02 Kbytes/sec)
5. 19529038 bytes received in 22.7 secs (8.4e+02 Kbytes/sec)
```

- Windows NT

```
1. 19529038 bytes received in 30,7 secs (649,39 kbytes/sec)
2. 19529038 bytes received in 29,20 secs (668,76 kbytes/sec)
3. 19529038 bytes received in 29,06 secs (671,98 kbytes/sec)
4. 19529038 bytes received in 28,96 secs (674,30 kbytes/sec)
5. 19529038 bytes received in 29,65 secs (658,59 kbytes/sec)
```

No teste acima, o Windows NT levou uma grande vantagem por estar instalado em uma máquina mais moderna, com discos e interfaces rápidas. Desta forma a taxa de transferência foi afetada pelo tempo de gravação em disco, consideravelmente maior no micro mais obsoleto no tocante a hardware, o 486. Para eliminar esta disparidade, realizei uma outra medição, desta vez sem realizar a gravação em disco.

O arquivo transferido foi gravado no *Conectiva Linux* em /dev/null, que na verdade é um buraco negro do Unix, ou seja, nada é gravado. O dispositivo equivalente no Windows NT é *nul*. O tempo de transferência não sofre o impacto do I/O e podemos ter uma boa idéia de quanto tempo efetivamente foi gasto apenas no trânsito dos dados via rede.

Nesta outra medição utilizei o comando *wget*, disponível em ambas as plataformas.

No NT a sintaxe do comando foi:

```
wget -O null ftp://ftp.unicamp.br/pub/dicas-l.zip
```

No Linux a sintaxe do comando foi:

```
wget -O /dev/null ftp://ftp.unicamp.br/pub/dicas-l.zip
```

Os dados obtidos neste teste estão incluídos abaixo. Foram feitas dez medições em cada sistema.

- WindowsNT

```
12:45:25 (631.78 KB/s) - 'nul' saved [2587767]
12:46:22 (631.78 KB/s) - 'nul' saved [2587767]
12:46:22 (631.78 KB/s) - 'nul' saved [2587767]
12:46:47 (631.78 KB/s) - 'nul' saved [2587767]
12:47:05 (631.78 KB/s) - 'nul' saved [2587767]
12:47:20 (631.78 KB/s) - 'nul' saved [2587767]
12:47:34 (505.42 KB/s) - 'nul' saved [2587767]
12:51:17 (631.78 KB/s) - 'nul' saved [2587767]
12:51:49 (631.78 KB/s) - 'nul' saved [2587767]
12:52:24 (631.78 KB/s) - 'nul' saved [2587767]
```

- Linux

```
09:45:29 (711.26 KB/s) - '/dev/null' recebido [2587767]
09:45:48 (751.00 KB/s) - '/dev/null' recebido [2587767]
09:46:10 (719.98 KB/s) - '/dev/null' recebido [2587767]
09:46:22 (779.25 KB/s) - '/dev/null' recebido [2587767]
09:46:34 (736.12 KB/s) - '/dev/null' recebido [2587767]
09:46:45 (724.31 KB/s) - '/dev/null' recebido [2587767]
09:46:56 (720.59 KB/s) - '/dev/null' recebido [2587767]
09:47:07 (706.69 KB/s) - '/dev/null' recebido [2587767]
09:47:18 (684.30 KB/s) - '/dev/null' recebido [2587767]
09:47:35 (661.90 KB/s) - '/dev/null' recebido [2587767]
```

Na média, a máquina Linux teve uma performance em 15% superior. Este valor é claro não é o correto para uma avaliação de desempenho visto a enorme diferença entre os dois equipamentos.

Capítulo 10

Correio Eletrônico

10.1 Sendmail

10.1.1 Redirecionamento de mensagens para o diretório pessoal dos usuários

Existem inúmeras vantagens em se configurar o seu sistema de correio eletrônico para redirecionar as mensagens para o diretório pessoal do usuário ao invés do diretório */var/spool/mail*.

A primeira delas é que o sistema de arquivos onde se encontra o diretório */var/spool/mail* nunca irá lotar devido ao mau uso do sistema por parte de alguns. O */var* lotado acarreta inúmeros inconvenientes: paralisação do correio eletrônico (ninguém consegue receber ou enviar mensagens), a contabilização do sistema e impressão é interrompida e o registro de eventos (*log*) também é desativado. Enfim, um grande transtorno é causado à comunidade de usuários e aos administradores do sistema.

Já quando as mensagens vão para o diretório pessoal do usuário, quaisquer problemas ficam restritos ao usuário em questão. Se ele exceder a sua quota em disco, apenas as mensagens a ele destinadas serão afetadas. Desta forma, favorece-se a criação de uma conscientização maior da comunidade para

estas questões.

Para configurar o sistema de mail para que as mensagens sejam gravadas no diretório pessoal de cada usuário é necessária a instalação do programa *procmail*, e a configuração dos programas que os usuários utilizam para ler suas mensagens, os chamados *MUAs* (*Mail User Agents*). Entre estes, os mais utilizados são *elm, pine* e *mail*. Existe também o serviço POP (*Post Office Protocol*), que faz a coleta, a partir do servidor de correio eletrônico, das mensagens para serem lidas na estação de trabalho ou microcomputador do usuário. Os programas *Eudora* e *Netscape Mail* utilizam o protocolo POP.

O POP não é exatamente um *MUA* mas ele precisa reconhecer onde buscar as mensagens para entregá-las então ao *MUA* utilizado pelo usuário.

No nosso exemplo utilizamos a versão 2.53 do programa *qpopper*[1].

Para a compilação do programa POP, realizar o download do arquivo de distribuição, expandi-lo no diretório de sua preferência e executar o programa *configure*:

```
# cd /usr/local/src
# wget -t0 ftp://ftp.qualcomm.com/eudora/servers/unix/popper/ \
  qpopper2.53.tar.Z
# tar xvzf qpopper2.53.tar.Z
# cd qpopper2.53
# ./configure
```

O processo de instalação do programa *qpopper* cria um arquivo *Makefile* através da execução do programa *configure*. Este programa obtém informações a respeito da arquitetura de sua estação de trabalho e cria um arquivo *Makefile* apropriado a este ambiente.

Uma vez criado o arquivo *Makefile*, editá-lo alterando a linha

```
DEFS            =       -DHAVE_CONFIG_H
```

[1] *ftp://ftp.qualcomm.com/eudora/servers/unix/popper/qpopper2.53.tar.Z*

para

```
DEFS   =  -DHAVE_CONFIG_H -DHOMEDIRMAIL
```

O próximo passo é configurar o programa *procmail* para fazer o redirecionamento das mensagens dos usuários para seus diretórios pessoais.

O programa *procmail* pode ser baixado a partir do endereço ftp://ftp.informatik.rwth-aachen.de/pub/packages/procmail/. Neste diretório escolha a versão mais recente. Este texto foi escrito baseado na versão 3.14. Nunca se esqueça de ler a documentação de instalação para certificar-se de que os passos descritos a seguir não foram alterados.

Como para o POP, faça a expansão do arquivo baixado da Internet e siga os seguintes passos para a instalação:

```
# cd /usr/local/src
# tar xvzf procmail-3.14.tar.gz
# cd procmail-3.14
# cd src
# vi authenticate.c
```

A configuração do procmail é feita descomentando-se, dentro do arquivo *authenticate.c*, que se encontra no diretório de fontes (*src*), a linha

```
#define MAILSPOOLHOME "/.mail"
```

Desta forma, todas as mensagens que chegarem serão gravadas dentro do diretório *$HOME/.mail*.

Resta então alterar os demais *MUAs* para que peguem suas mensagens a partir da nova localização.

Para realizar esta alteração no *pine*, por exemplo, entre na opção *Setup* e altere o valor *inbox-path* para o caminho desejado:

```
# Path of (local or remote) INBOX, e.g.
# ={mail.somewhere.edu}inbox
# Normal Unix default is the local INBOX
# (usually /usr/spool/mail/$USER).
inbox-path=$HOME/.mail
```

Os programas *Eudora* e *Netscape*, enfim, qualquer um que utilize o protocolo POP, irão obter estas informações diretamente do servidor POP e não há necessidade de alterar suas configurações.

10.1.2 Comandos expn e vrfy

Não é surpresa que muitos usuários de sistemas utilizam senhas fáceis de serem adivinhadas, como sua identificação no sistema, ou esta mesma identificação invertida, e outras variações.

O *sendmail* possui dois comandos, que podem ser acessados a partir da porta 25, chamados *expn* e *vrfy* que fornecem informações sobre seus usuários. Veja o exemplo:

```
$ telnet abc.com.br 25
[No write since last change]
Trying...
Connected to abc.com.br.
Escape character is '^]'.
220 abc.com.br ESMTP Sendmail 8.9.3/8.9.3; Wed, 12 Jan 2000 19:00:56 -02)
vrfy suporte
250 <suporte@abc.com.br>
expn suporte
250-<user1@abc.com.br>
250-<user2@abc.com.br>
250-<user3@abc.com.br>
250 <user4@abc.com.br>
```

O comando *vrfy* verificou a existência do usuário suporte, que na verdade é um apelido (*alias*). Já o comando *expn* fez a expansão do alias e mostrou todos os usuários incluídos na definição do alias suporte.

Para evitar isto, inclua no arquivo *sendmail.cf* a diretiva

```
O PrivacyOptions=goaway
```

Isto feito, veja o que acontece quando se tenta novamente utilizar os comandos *vrfy* ou *expn*:

```
% telnet abc.com.br 25
Trying...
Connected to abc.com.br
Escape character is '^]'.
220 abc.com.br ESMTP Sendmail 8.9.3/8.9.3; Wed, 12 Jan 2000 15:53:14 -020
vrfy queiroz
252 Cannot VRFY user; try RCPT to attempt delivery (or try finger)
expn suporte
502 Sorry, we do not allow this operation
```

10.2 Processamento Seletivo da Fila

O *sendmail* pode fazer o processamento seletivo de mensagens, ou seja, dentre as mensagens que ainda se encontram na fila, apenas aquelas que satisfizerem o critério especificado serão processadas.

O comando

```
/usr/sbin/sendmail -qRroot -qSqueiroz@abc.com.br
```

fará com que apenas as mensagens destinadas ao usuário *root* enviadas por *queiroz@abc.com.br* sejam processadas:

```
# mailq
             Mail Queue (2 requests)
--Q-ID-- --Size-- -----Q-Time----- -------Sender/Recipient-
IAA00816        0 Thu Apr 15 08:50 <queiroz@abc.com.br>
                                   <root@abc.com.br>
IAA00818        0 Thu Apr 15 08:50 <queiroz@abc.com.br>
                                   <queiroz@abc.com.br>
# /usr/sbin/sendmail -v -qRroot -qSqueiroz@abc.com.br
```

```
Running IAA00816 (sequence 1 of 1)
<root@abc.com.br>... Connecting to local...
<root@abc.com.br>... Sent
[root@ spool]# mailq
             Mail Queue (1 request)
--Q-ID-- --Size-- -----Q-Time----- -------Sender/Recipient-
IAA00818        0 Thu Apr 15 08:50 <queiroz@abc.com.br>
                                   <queiroz@abc.com.br>
```

Como se pode ver acima, a fila continha originalmente duas mensagens. Após o processamento da fila com as opções fornecidas (destinatário *root* e remetente *queiroz@abc.com.br*) a fila ficou com apenas uma mensagem.

A opção "*q*" pode receber adicionalmente os valores:

Opção	Significado
R	Endereço do Destinatário (Recipient)
S	Endereço do Remetente (Sender)
I	Identificação da Mensagem na Fila

Como exemplificado, diversos argumentos podem ser fornecidos na linha de comandos.

10.2.1 mailcheck.sh

Um dos problemas que administradores de sistemas encontram com bastante freqüência é o gerenciamento do espaço em disco. O diretório */var*, por exemplo, é vital para o funcionamento normal do sistema.

Se o sistema de arquivos onde se encontra o diretório */var* atingir uma taxa de uso de 100% vários serviços serão interrompidos, como por exemplo a coleta de dados para contabilização de uso de recursos do sistema, impressão e o mais importante de todos, o correio eletrônico.

Quando o */var* atinge 100% nenhuma mensagem pode ser enviada ou recebida visto que mensagens que chegam são gravadas no diretório */var/spool/mail*

e mensagens que saem vão para a fila em */var/spool/mqueue*.

Várias medidas podem ser tomadas para evitar tamanho transtorno. A shell script que se segue pode ajudar nesta tarefa transferindo arquivos do diretório */var/ spool/ mqueue* para o diretório pessoal do usuário. O usuário recebe também uma notificação informando-o da transferência dos arquivos e os procedimentos que devem ser seguidos para recuperar suas mensagens.

mailcheck.sh

```
#!/bin/bash
# Definição de Variáveis
DATA =  `date +%d%m%y`
DIR = /usr/sys
dir_home = '$''home'

# O laço abaixo irá  primeiramente  gerar
# uma listagem  dos arquivos  contidos em
# /var/spool/mail  e  selecionar  aqueles
# que sejam maiores que  400K,  ou  seja,
# os usuários afetados serão apenas aque-
# les que  não  lêem  suas  mensagens com
# frequencia.  Os arquivos que forem mai-
# ores  que este valor serão movidos para
# o diretório home do usuário e receberão
# o  nome mbox.ddmmyy, onde os valores do
# dia, mes, ano serão determinados a par-
# tir do comando "date +%d%m%y". O arqui-
# vo em /var/spool/mail é então  deletado
# e sua cópia, agora no diretório do usu-
# ário, é  compactada. O último passo é a
# mudança de  ownership do arquivo para o
# usuário (visto que o arquivo foi criado
# originalmente pelo usuário root.
```

```
# Todos  os comandos contidos neste  laço
# serão executados para  todos  os  usuá-
# rios cujas caixas  postais  excedam 400
# Kbytes.

for USER in `/bin/ls -l /var/spool/mail|awk \
    '$5 > 400000 {print $9}'`
do
 mv /var/spool/mail/$USER ~$USER/mbox.$DATA
 rm ~$USER/mbox.$DATA.gz >&! /dev/null
 /bin/gzip ~$USER/mbox.$DATA
 chown $USER  ~$USER/mbox.$DATA.gz

# A seguir é composta a mensagem que será
# enviada para o usuário. O  comando  cat
# abaixo irá criar um arquivo em
# $DIR/mensagem.mail que conterá todos os
# caracteres digitados até que  sejam en-
# contrados os caracteres EOF, que  atuam
# como delimitadores. Vale a pena prestar
# atenção nessa facilidade, bastante útil.

cat >! $DIR/mensagem.mail << EOF

Prezado(a) Usuario(a),

Visto que sua caixa postal (/var/spool/mail/$USER)
está excedendo o tamanho máximo permitido de 400K,
ela foi transferida para o seu diretório pessoal com
o nome mbox.$DATA.gz.

Isto se fez necessário visto que o sistema de arquivos
/var estava com uma taxa de ocupação próxima a 100%,
```

```
impedindo que nossos usuários
enviassem ou recebessem mensagens.

Favor seguir os seguintes passos para
conseguir ler os suas mensagens novamente :

1) cd   (para ir para o seu diretorio principal)
2) /bin/gzip -d  mbox.$DATA.gz
   (para descompactar as mensagens)
3) cat mbox.$DATA >> /var/mail/$USER

Gostaríamos de lembra-lo que seus mails
devem ser lidos e movidos para o seu espaço
de armazenamento pessoal e nunca deixados
na caixa de correio localizada no diretório /var.

A maioria dos programas utilizados para
leitura de mail (elm, pine, netscape, etc.)
oferece facilidades para se criar folders,
onde suas mensagens podem ser guardadas separadas
por assuntos em outro local que não o diretório
/var/spool/mail/$USER.

Em caso de dúvidas envie mensagem para o
endereco suporte@abc.com.br
EOF
mail -s "Modificações em sua mailbox..." \
             $USER  < $DIR/mensagem.mail
done
```

10.2.2 BCC – Blind Carbon Copy

Existem ocasiões em que se deseja enviar uma mensagem a alguém e uma cópia da mensagem a outra pessoa, porém sem que o destinatário original saiba da existência da cópia.

Para isto existe a opção *BCC* ou *Blind Carbon Copy*. Ao se especificar esta opção o programa *sendmail*, após realizar o processamento dos destinatários, retira a linha contendo a opção *BCC* e realiza a entrega da mensagem.

Desta forma o destinatário original da mensagem, ao examinar os cabeçalhos, não verá que a mensagem foi entregue a outras pessoas.

Para exemplicar, no programa *elm*, na tela *Message Header Edit Screen*, que se invoca após digitar uma mensagem, temos:

```
                        Message Header Edit Screen
T)o: queiroz (Rubens Queiroz de Almeida)
C)c:
B)cc: joao@abc.com.br
S)ubject: TESTE
R)eply-to:
A)ction:                                E)xpires:
P)riority:                              Precede(n)ce:
I)n-reply-to:
  Choose header, u)ser defined header, d)omainize, !)shell, or <return>.
Choice:
```

A mensagem que foi enviada ao usuário queiroz, será também enviada para o usuário *joao@abc.com.br*. O usuário *queiroz* não terá como identicar que uma cópia da mensagem foi enviada a outra pessoa, pois o *sendmail* retira esta informação do cabeçalho da mensagem.

Já a opção *CC* (*Carbon Copy*) aparece nos cabeçalhos e todos os que receberem a mensagem conseguirão vê-la.

10.3 Interação Sendmail x DNS

Todo domínio de email deve possuir um registro no banco de dados do DNS. Estes registros chamam-se *MX* ou *Mail Exchange* e indicam qual computador realiza entrega de mensagens para determinado domínio.

Tomemos por exemplo o endereço *usuario@abc.com.br*. Para se realizar entrega de mensagens para endereços como este é necessário que o programa *sendmail* interaja com o DNS para obter nomes de computadores reais que façam a entrega destas mensagens.

Vejamos como isto ocorre no DNS:

```
% nslookup
Default Server: ns.abc.com.br
Address:  200.200.10.10
> set type=mx
```

O comando acima indica que se deseja obter do servidor DNS apenas registros do tipo MX.

A seguir especificamos o domínio a respeito do qual desejamos informações, no caso o domínio *abc.com.br*:

```
> abc.com.br
Non-authoritative answer:
abc.com.br  preference = 50, mail exchanger = smtp1.abc.com.br
abc.com.br  preference = 75, mail exchanger = smtp2.abc.com.br
abc.com.br  preference = 100, mail exchanger = smtp3.abc.com.br
 ...
```

O domínio *abc.com.br* possui três servidores que podem atendê-lo, *smtp1.abc.com.br*, *smtp2.abc.com.br* e *smtp3.abc.com.br*. A cada um destes servidores está associado um número. Este número indica a prioridade que cada um dos servidores possui. Quanto mais baixo o número mais alta a prioridade.

Desta forma, mensagens enviadas para o domínio *abc.com.br* devem ser entregues primeiramente à máquina *smtp1.abc.com.br* (valor 50). Caso esta máquina esteja fora do ar, as mensagens devem então ser entregues à máquina *smtp2.abc.com.br* (valor 75). E em último caso, as mensagens devem ser entregues à máquina *smtp3.abc.com.br* (valor 100).

Do visto acima, todas as mensagens para determinado domínio possuem uma ordem de precedência quanto ao computador que irá recebe-las.

O fato de termos três servidores que possam receber mensagens endereçadas a um domínio como *abc.com.br* não implica no fato de que os usuários tenham contas nas três máquinas.

Quando alguém manda uma mensagem para *joao@abc.com.br*, o programa *sendmail* que irá fazer a entrega da mensagem primeiramente faz uma consulta ao DNS pedindo os registros MX associados ao domínio *abc.com.br*. De posse das respostas, ele tentará entrar em contato com o servidor especificado no registro MX de mais alta prioridade (número mais baixo). Este servidor, o de prioridade mais alta, normalmente é o servidor onde as pessoas tem suas contas. Ou seja, se este servidor estiver no ar e funcionando perfeitamente, o programa *sendmail* deste servidor recebe a mensagem e a repassa ao programa de entrega (recomendável usar o programa *procmail* para esta tarefa) que a colocará na caixa postal do usuário.

Caso este servidor não esteja funcionando, o programa *sendmail* tentará então entrar em contato com o próximo na lista. O usuário *joao* não possui conta neste segundo servidor. O que ocorre então? A mensagem é recebida e ao invés de ser gravada na caixa postal do usuário (que não existe neste equipamento), é mantida na área de *spool* de mensagens. Esta mensagem ficará então na fila e de tempos em tempos o *sendmail* tentará conectar com a máquina destino, para enviar a mensagem.

É claro que, caso a máquina cujo registro MX tenha a prioridade mais alta demore mais que cinco dias[2] para voltar a funcionar, existe o risco das mensagens retornarem a quem as enviou dizendo que o destino está

[2]Este é o valor padrão e pode ser alterado através da reconfiguração do *Sendmail*

inacessível.

Alguns pontos importantes a serem lembrados. Os servidores MX de determinado domínio devem residir em redes diferentes, o mais afastadas possível. Caso residam em redes diferentes mas compartilhem a mesma linha de comunicação para acesso à Internet não adianta muita coisa. Se a linha cair as duas ou mais máquinas ficam inacessíveis e a mensagem não será entregue. Se possível, coloque os servidores MX de seu domínio em redes diferentes, como por exemplo, uma delas no tronco RNP[3] da Internet Brasil e outra no tronco Embratel. Desta forma você garante que mensagens enviadas para o seu domínio sempre cheguem.

E se a sua máquina demorar muito para ser consertada? Contacte o administrador do seu servidor MX secundário e peça a ele para ir salvando as mensagens destinadas ao seu domínio em uma área alternativa. Desta forma não se perdem mensagens e você economiza tempo de processamento do servidor secundário visto que ele não ficará tentando entregar, a intervalos normalmente definidos em uma hora, mensagens que não podem ser entregues.

Nunca se esqueça de avisar o administrador do servidor MX secundário quando houverem problemas com o servidor primário. Especialmente se o seu domínio recebe muitas mensagens. Se o servidor secundário for uma máquina com poucos recursos de armazenamento, problemas sérios poderão ocorrer. Se o sistema de arquivos ficar cheio mensagens para o domínio que o servidor secundário atende também serão devolvidas.

Uma vez que a sua máquina voltar a funcionar, o administrador do servidor MX secundário poderá invocar o *sendmail* para processar especificamente a fila de mensagens para o seu domínio.

Suponhamos que isto tenha sido feito movendo-se as mensagens para o diretório */var/spool/backup*. Para invocar o *sendmail* fazendo com que ele utilize este diretório basta invocá-lo com as seguintes opções abaixo:

[3]Rede Nacional de Pesquisa

```
# /usr/sbin/sendmail -OQueueDirectory=/var/spool/backup -q -v
```

Uma consideração final, todas as precauções sugeridas para os servidores MX secundários se aplicam também aos servidores DNS secundários. Para ficar garantido, coloque o seu servidor primário no Brasil e um secundário na Austrália. Só para garantir ;-)

10.3.1 User Database

O *sendmail* é um programa de entrega de mensagens que oferece inúmeras opções de configuração. Uma delas é a facilidade de se realizar alterações no campo de endereço das mensagens. Os endereços dos destinatários das mensagens podem ser alterados de acordo com especificações fornecidas pelo administrador de redes. Estas definições são armazenadas em um banco de dados externo, denominado *userdb*, que é consultado pelo *sendmail* sempre que processa uma mensagem.

Normalmente, os endereços são consultados no arquivo */etc/aliases*. Se não forem encontrados lá, é então consultado o arquivo chamado *.forward*, no diretório pessoal do usuário. Se o banco de dados dos usuários (*userdb*) estiver habilitado, as definições deste banco de dados serão pesquisadas após a consulta ao arquivo */etc/aliases* e antes da consulta ao arquivo *.forward*.

O *userdb* é habilitado automaticamente quando o *sendmail* é compilado com suporte ao *NEWDB* ou *HESIOD*. Para verificar se o seu programa *sendmail* provê este suporte, emitir o comando:

```
Version 8.9.3
Compiled with: LOG MATCHGECOS MIME7TO8 MIME8TO7 NAMED_BIND NETINET
               NETUNIX NEWDB NIS QUEUE SCANF SMTP USERDB XDEBUG
                                                      ^^^^^^
```

Como podemos ver, a versão do *sendmail* é 8.9.3 e o suporte ao *USERDB* está habilitado.

Precisamos agora sinalizar ao *sendmail* onde se encontra este banco de dados. Precisamos incluir no arquivo */etc/sendmail.cf* a definição:

```
O UserDatabaseSpec=/etc/mail/userdb.db
```

Caso o arquivo */etc/sendmail.cf* seja criado a partir do *m4*, incluir no arquivo *mc* correspondente a seguinte linha:

```
define(‘confUSERDB_SPEC’,/etc/userdb.db)
```

O passo seguinte é criar o banco de dados de usuários. Isto é feito usando-se o comando *makemap*:

```
# makemap btree /etc/userdb.db < /etc/userdb
```

O banco de dados, ou mapa, será criado tomando por base o arquivo no formato texto que se encontra em */etc/userdb*.

O arquivo */etc/userdb* consiste de entradas contendo dois valores: uma chave e seu valor correspondente. A chave é a identificação de um usuário (username) seguido de ":" e um de dois valores possíveis: *maildrop* ou *mailname*. A palavra chave utilizada determina a natureza do campo seguinte.

Por exemplo:

```
root:maildrop  queiroz@abc.com.br,joao@abc.com.br
```

No exemplo acima, mensagens endereçadas para o usuário *root* são direcionadas para os usuários *queiroz@abc.com.br* e *joao@abc.com.br*. Os endereços para onde as mensagens são redirecionadas devem ser separados por vírgulas. É também possível incluir-se um endereço por linha, como abaixo:

```
root:maildrop          queiroz@abc.com.br
root:maildrop          joao@abc.com.br
```

Neste segundo caso, o comando *makemap* deve ser invocado com a opção "-*i*".

A outra possibilidade é a utilização da palavra chave mailname:

```
queiroz:mailname        queiroz@abc.com.br
joao:mailname           joao@acme.com.br
maria:mailname          maria@netroad.com
```

Os valores acima fazem com que mensagens enviadas pelo usuário *queiroz* saiam identificadas como se tivessem sido enviadas por *queiroz@abc.com.br*. As mensagens enviadas pelo usuário *joao* sairão como se tivessem sido enviadas por *joao@acme.com.br*.

Esta configuração é bastante útil para provedores de acesso, que possuem vários clientes, cada um dos quais com um domínio diferente. Desta forma cada um dos clientes poderá enviar mensagens e sua identificação corresponderá ao domínio de sua empresa.

Veja o trecho do cabeçalho de uma mensagem:

```
From queiroz  Thu Jun  3 08:01:37 1999
Return-Path: <queiroz@abc.com.br>
Received: (from queiroz@localhost)
     by galaxy.abc.com.br (8.9.3/8.9.1) id IAA05608
     for queiroz; Thu, 3 Jun 1999 08:00:21 -0300
```

Como se pode ver, o valor do caminho de retorno (*Return-Path*) é *queiroz@abc.com.br*.

Embora sistemas *Conectiva Linux* permitam nomes de usuários maiores que 8 caracteres, isto não é aconselhável pois podem existir programas que imponham esta limitação na prática. A obediência a esta restrição não nos permite ser muito criativos na forma como os nomes dos usuários são definidos. A facilidade *userdb* nos permite também um maior grau de flexibilidade neste aspecto. Veja o exemplo:

```
queiroz:mailname                Rubens.Queiroz
Rubens.Queiroz:maildrop         queiroz
```

Não esquecer que neste caso, onde o recipiente das mensagens é alterado, as entradas *mailname* e *maildrop* devem ocorrer em pares. Isto para que quando alguém responder a *Rubens.Queiroz@abc.com.br* a mensagem possa ser entregue corretamente ao usuário Linux chamado *queiroz*.

10.3.2 Relays

Na configuração do programa *sendmail* é possível fazer com que todas as mensagens geradas localmente sejam enviadas para processamento em outro computador. Isto pode ser usado em computadores menos poderosos que não tenham condição de processar grande volume de mensagens. O termo *relay*, bastante utilizado, indica o computador que faz o processamento da mensagem no lugar de outro.

No arquivo *m4* de configuração, todos os tipos de relay são definidos, por exemplo, da seguinte forma:

```
define(‘LOCAL_RELAY’,‘agente:computador.’)
```

Na linha acima, “agente” indica o nome de um agente de entrega de mensagens. “computador” é o nome do servidor de correio eletrônico para onde serão enviadas as mensagens.

Por exemplo:

```
define(‘LOCAL_RELAY’,‘smtp:abc.com.br.’)
```

Neste exemplo, a definição *LOCAL_RELAY* especifica que todas as mensagens originadas a partir de um endereço não qualificado, ou seja, sem a parte *@host*, será entregue para processamento ao computador denominado *abc.com.br*.

Outras possibilidades:

- *MAIL_HUB*
  A macro *MAIL_HUB* especifica que todas as mensagens locais sejam enviadas para um servidor central para processamento. Esta opção pode ser usada em um ambiente onde o diretório de spool de mensagens é compartilhado. Para evitar problemas de lock de arquivos, o processamento das mensagens é feito em um servidor central. Se forem definidas as macros *LOCAL_RELAY* e *MAIL_HUB*, as mensagens cujo endereço não contenham a parte do computador ou domínio, serão enviadas para o computador definido pela macro *LOCAL_RELAY*. As demais mensagens serão enviadas para o computador definido pela macro *MAIL_HUB*.

- *SMART_HOST*
  Caso se deseje que TODAS as mensagens sejam processadas em outro computador, definir a macro *SMART_HOST*. A sintaxe é semelhante à das macros *MAIL_HUB* e *LOCAL_RELAY*.

10.3.3 Autorização para Relay

Normalmente, nas versões mais recentes do *sendmail*, o relay de mensagens é proibido. Desta forma faz-se necessário que se autorize especificamente os domínios para os quais o seu servidor SMTP concorda em fazer o relay de mensagens.

No arquivo *sendmail.cf* encontra-se a seguinte definição:

```
# Hosts that will permit relaying ($=R)
FR-o /etc/mail/relay-domains
```

Esta é uma macro de classe (múltiplos valores são permitidos), cujos valores são lidos a partir do arquivo */etc/mail/relay-domains*.

Este arquivo contém todos os domínios para os quais se deseja autorizar o relay de mensagens. Este arquivo é lido pelo *sendmail* quando de sua ativação. Quaisquer mudanças neste arquivo requerem que o *sendmail* seja reinicializado para se tornarem efetivas.

A forma através da qual o relay pode ser controlado em seu servidor pode ser adicionalmente configurada por meio da definição das seguintes macros:

- *FEATURE(relay_hosts_only)*

  Por definição, todos os computadores pertencentes aos domínios listados no arquivo */etc/mail/relay-domains* são autorizados a realizar relay. Se este recurso for ativado apenas os computadores listados serão autorizados a realizar relay.

- *FEATURE(relay_entire_domain)*

  Esta facilidade permite que todos os computadores dentro do domínio possam realizar relay. É equivalente a se incluir o nome do domínio no arquivo *relay-domains*.

- *FEATURE(access_db)*

  A permissão de acesso é fornecida após consulta ao banco de dados */etc/mail/access*. O acesso pode ser concedido a domínios ou computadores.

- *FEATURE(rbl)*

  Rejeita mensagens baseado no serviço chamado *Realtime Blackhole List* mantido por Paul Vixie. Este serviço lista *spammers* conhecidos e empresas que fazem uso indevido ou ilícito do correio eletrônico.

- *FEATURE(accept_unqualified_senders)*

  Normalmente o *sendmail* não aceita mensagens de um remetente sem um domínio. Se esta opção for ativada estas mensagens passam a ser aceitas.

- *FEATURE(relay_local_from)*

Permite o relay de mensagens que supostamente se originem de seu domínio. Como isto é fácil de se falsificar, ativar esta opção não é recomendável.

- *FEATURE(promiscuous_relay)*

  Esta opção desativa por completo todos os controles de relay utilizados pelo *sendmail*. A não ser que você saiba o que está fazendo esta é uma péssima idéia.

10.3.4 Modo de Operação queueonly

Se por alguma razão desejarmos fazer um pré-processamento das mensagens enviadas para o seu sistema (detecção de *mail bombs, spamming*, etc), você pode colocar o *sendmail* para funcionar no modo *queueonly*.

Neste modo de funcionamento, as mensagens recebidas por seu sistema não são entregues imediatamente a seus usuários mas ficam armazenadas na fila do sistema (*/var/spool/mqueue)*.

Para especificar que o *sendmail* funcione deste modo, devemos alterar o arquivo */etc/sendmail.cf*. Na opção

```
O DeliveryMode=background
```

mude para

```
O DeliveryMode=queueonly
```

Alterar a forma como o *sendmail* é invocado em seu sistema. Geralmente o *sendmail* é inicializado com as seguintes opções:

```
/usr/sbin/sendmail -bd -q1h
```

O comando acima especifica que o *sendmail* deve rodar como um daemon ("*-bd*") aceitando conexões e que a fila deve ser processada de hora em hora

("*-q1h*"). Para evitar que a fila seja processada ativar o *sendmail* da seguinte forma:

```
/usr/sbin/sendmail -bd
```

Desta forma, o *sendmail* continuará aceitando conexões e mensagens mas não haverá entrega para os usuários.

A entrega efetiva das mensagens será feita invocando-se o *sendmail* de tempos em tempos, conforme a conveniência do administrador.

Continuando em nosso exemplo, executamos uma shell script ou algum programa para fazer o pré-processamento. As mensagens que passarem pelo critério de seleção são então movidas para um outro diretório, como por exemplo, */var/spool/MQUEUE*. Todas as mensagens neste diretório podem ser entregues sem problemas. Ao final do processamento o programa *sendmail* é invocado com as seguintes opções:

```
/usr/sbin/sendmail -oQ/var/spool/MQUEUE -q
```

Desta forma, através da opção "*-oQ*", o *sendmail* irá processar as mensagens que se encontram em */var/spool/MQUEUE* e não no diretório padrão (*/var/spool/mqueue*). A opção "*-q*" sinaliza ao *sendmail* para processar a fila.

É claro que existem outras maneiras de se fazer isto. Esta é apenas uma delas.

10.3.5 Envio de mensagens diretamente com Sendmail

É possível se enviar mensagens diretamente com o programa *sendmail* (MTA ou *Mail Transport Agent*), sem o intermédio de um software intermediário (*MUA* ou *Mail User Agent*).

É claro que esta opção não é das mais convenientes, visto que é muito mais fácil enviar mensagens com programas como *elm*, *Netscape Composer*, *pine* e outros.

Entretanto, especialmente do ponto de vista do administrador de sistemas, existem ocasiões em que pode ser mais conveniente usar diretamente o programa *sendmail*.

Uma destas vantagens é a configuração do cabeçalho de forma a atender determinadas necessidades.

Em toda mensagem de correio eletrônico o cabeçalho é separado do corpo da mensagem por uma linha em branco.

Suponhamos então que queiramos enviar uma mensagem para todos os usuários, como abaixo:

```
Reply-To: suporte@abc.com.br
Subject: Parada Programada

Senhores Usuários(as),

Hoje haverá uma parada programada às 17:00horas
com retorno previsto para as 17:30 hs.

Atenciosamente,

Suporte Técnico
```

Foram incluídas duas linhas de cabeçalho. A primeira delas, *Reply-To*, define quem irá receber a resposta. Neste caso o endereço *suporte*, que consiste de várias pessoas. Não é conveniente que respostas a mensagens deste tipo retornem para apenas uma pessoa. A segunda linha define o *Subject*, ou assunto, da mensagem. Em seguida, uma linha em branco e finalmente a mensagem.

Para enviar esta mensagem, podemos fazer a seguinte *shell script*

anuncio.sh

```
#!/bin/bash
for user in `awk -F: '{print $1}' /etc/passwd`
do
  /usr/lib/sendmail $user < msg
  echo $user
  echo $user > ultimo-endereço
  sleep 2
done
```

Foi colocado um controle, gravando o nome do último usuário para quem a mensagem foi enviada no arquivo ultimo-endereço. Desta forma, em caso de queda de sistema ou algum outro contratempo, o processo pode ser retomado do ponto em que foi interrompido. Basta remover da lista de usuários os nomes para os quais a mensagem já foi enviada.

10.3.6 Verificação de endereços eletrônicos

Podemos fazer a verificação da sintaxe de endereços eletrônicos com o próprio programa *sendmail*. Isto é particularmente útil em listas eletrônicas com um grande número de assinantes para garantir que o *sendmail* não irá gastar tempo processamento endereços inválidos.

O shell script abaixo faz esta verificação:

emailcheck.sh

```
#!/bin/bash
for user in `cat list`
do
  /usr/lib/sendmail -bv $user >> check
done
```

O *sendmail*, quando invocado no modo *-bv* faz apenas uma verificação do endereço. Nenhuma mensagem é entregue.

Para cada endereço válido, aparece algo do tipo:

```
souza@acme.com... deliverable: mailer esmtp,
host acme.com., user souza@acme.com
```

Para cada endereço válido foi determinado o agente de entrega (*mailer*) a ser utilizado, o nome do computador onde a mensagem será entregue e para qual usuário.

Para endereços com erro:

```
opera@abc.com.br... User unknown
```

Então, de posse do arquivo *check*, executamos o comando:

```
cat check | grep -v deliverable > erros
```

No arquivo *erros* são gravados todos os endereços que apresentaram algum erro. De posse desta lista, os endereços incorretos podem então ser removidos.

10.3.7 Manuseio de Mensagens

Para dividir uma caixa postal em mensagens individuais, utilize o comando *slice*.

Uma vez que se tenha o programa compilado, basta executar:

```
% slice -m -f mailfolder
```

Serão criados vários arquivos com o nome

```
mailfolder:1997-10-24.18:01:11
mailfolder:1997-10-24.18:15:19
mailfolder:1997-10-24.18:15:20
```

Resta agora eliminar os cabeçalhos gravados pelo *sendmail*. Como o que divide a mensagem dos cabeçalhos é uma linha em branco, a shell script abaixo dá conta do recado:

mailsplit.sh

```
#!/bin/bash
for file
do
  echo $file
  ed $file << EOF
  1,/^$/ delete
  w
  q
  EOF
done
```

Esta shell script apaga da primeira linha do arquivo até a primeira linha em branco, salvando-o em seguida.

10.3.8 Determinando a versão do sendmail

Para descobrir a versão do *sendmail* rodando em determinado servidor realize uma conexão no serviço *telnet* na porta 25:

```
% telnet abc.com.br 25
Trying...
Connected to abc.com.br.
Escape character is '^]'.
20 abc.com.br ESMTP Sendmail 8.9.3/8.9.3; Wed, 18 Jan 2000 08:11:18 -0300
                             ^^^^^^^^^^
```

Os dois valores assinalados identificam a versão do *sendmail* e a versão de seu artigo de configuração (*sendmail.cf*)

10.4 Reenvio de Pastas de Correio Eletrônico

Para enviar um folder inteiro de um computador para outro pode-se usar o comando *formail*, integrante do pacote *procmail*.

Basta fazer como abaixo:

```
formail -s sendmail username@dominio.com < folder
```

10.5 Envio de Mensagens no Formato HTML

O próprio sendmail para ser utilizado para enviar mensagens. Toda mensagem de correio eletrônico é composta de um cabeçalho e a mensagem em si. O que separa uma parte da outra é uma linha em branco.

Se criarmos um arquivo com o seguinte conteúdo:

```
Subject: Pagina html de teste
Content-Type: text/html; charset="us-ascii"

<HTML>
<HEAD>
 <META HTTP-EQUIV="Content-Type"
   CONTENT="text/html; charset=iso-8859-1">
 <META NAME="Author" CONTENT="Rubens Queiroz">
   <TITLE>Dicas-L - Sumario</TITLE>
</HEAD>
<BODY TEXT="#000000" BGCOLOR="#FFFFFF"
  LINK="#006600" VLINK="#D96C00" ALINK="#FF0000"
  BACKGROUND="back2.jpg">
```

```
<FONT FACE="Verdana,Arial,Helvetica">
```

e emitirmos o seguinte comando:

```
sendmail usuario@abc.com.br < arquivo.html
```

a mensagem será formatada seguindo as diretivas HTML. A formatação é sinalizada pela linha de cabeçalho

```
Content-Type: text/html; charset="us-ascii"
```

que indica o formato do conteúdo da mensagem (*txt/html*).

Isto, é claro, desde que você use um cliente de correio eletrônico que esteja habilitado a entender esta formatação, como o *Netscape Messenger*.

A título de experiência, tente enviar a mesma mensagem removendo a linha contendo a diretiva *Content-Type*. A mensagem não será formatada e tudo o que veremos será o código html.

10.6 Pine

O pine, para quem não conhece, é uma excelente ferramenta para leitura, composição, envio, etc, de mensagens e leitura de notícias (*Usenet News*).

Existe um sítio[4] dedicado ao pine na Universidade de Washington e uma lista de discussão cujo conteúdo está disponível na Web.

10.7 Postagem Cruzada de Mensagens

A maioria das pessoas que assinam listas cobrindo assuntos semelhantes freqüentemente recebem a mesma mensagem, enviadas pelo remetente para

[4] *http://www.washington.edu/pine*

diversas listas. Isto é um tremendo inconveniente para o assinante, que vê a mesma mensagem repetidas vezes. Isto é o que se chama de postagem cruzada ou *cross-posting*.

Para evitar este transtorno podemos lançar mão do programa *procmail*, que é capaz de identificar este tipo de mensagens e tratá-las adequadamente.

Isto é feito através da configuração de filtros para eliminar as mensagens duplicadas.

Acrescente as seguintes linhas antes das regras de filtragem de e-mail ao seu arquivo */.procmailrc*[5]:

```
# Evitar cros-posting
# 8K de "cache" para guardar os IDs
# das mensagens, que são únicos.
:0 Wh: msgid.lock
| formail -D 8192 msgid.cache
```

O tamanho do *buffer* pode ser aumentado ou diminuído a contento. A mensagem que é arquivada vai para a caixa de entrada que satisfizer o primeiro dos filtros/regras, enquanto que as demais são descartadas.

Um procedimento saudável antes de se mexer no arquivo *.procmailrc* é criar uma pasta de segurança para garantir que não se perca nenhuma mensagem e colocar as seguintes linhas logo após as definições de variáveis de ambiente:

```
:0 c            # Habilitar para testes.
backup
```

[5]Colaboração: Jorge Luiz Godoy Filho

Capítulo 11

Integração Windows e Linux

11.1 Como nasceu o *Samba*

O criador do *Samba* chama-se Andrew Tridgell, e é natural da Austrália. Quando escreveu o *Samba*, Andrew era ainda um estudante de ciência da computação na Universidade Nacional Australiana em Camberra O *Samba* nasceu a partir de um problema que Andrew tinha, o de integrar um servidor Unix com um PC rodando DOS. Esta integração na verdade não era o problema, visto que ele possuía uma versão do NFS (*Network File System*) que permitia que o PC acessasse os arquivos do seu servidor Unix. O problema é que uma de suas aplicações exigia a interface NetBIOS. E daí, graças a este pequeno problema doméstico, o mundo inteiro e centenas de empresas podem hoje se beneficiar de um software de altíssima qualidade que chega mesmo a suplantar em performance e recursos os seus equivalentes da plataforma Microsoft.

Escrever o *Samba* não foi tarefa das mais fáceis. À época as especificações do protocolo SMB (*Server Message Block*), utilizado pela Microsoft para realizar o compartilhamento de recursos entre seus computadores não era aberta. Andrew teve que realizar a engenharia reversa do protocolo utilizando um software de análise de rede (*Packet Sniffer*). Uma vez decifrado

o modo de operação do protocolo SMB, Andrew fez a implementação do protocolo em seu computador Unix. Desta forma o seu computador Unix aparecia na rede NetBIOS como um servidor de arquivos. Os sistemas de arquivos do servidor Unix podiam então ser montados e acessados por aplicações NetBIOS.

Andrew publicou seu trabalho em 1992 e continuou trabalhando no código por mais algum tempo após o qual o projeto foi abandonado. Dois anos mais tarde Andrew precisou conectar seu sistema Linux com o computador Windows de sua esposa. Na falta de uma opção melhor, utilizou o seu próprio código. Para sua surpresa, tudo funcionou perfeitamente.

Nestes dois anos, mais uma surpresa agradável. A Microsoft tornou pública a especificação do protocolo SMB e da arquitetura NetBIOS. Com esta informação nas mãos ele retomou seu trabalho. Desde então a adesão ao *Samba* cresceu vertiginosamente. Existem hoje vários colaboradores e o projeto deixou de ser a iniciativa isolada de apenas uma pessoa. Sempre que uma nova versão é anunciada milhares de cópias são descarregadas das dezenas de sítios Web que distribuem o *Samba*.

Além disto, existem já empresas que incluem o *Samba* na distribuição de seus sistemas Unix comerciais. O exemplo mais notável desta tendência é a empresa Silicon Graphics, que distribui uma versão adaptada do *Samba*.

Mas a real medida do sucesso que o *Samba* tem feito no mercado é a sua menção nos famosos memorandos internos da Microsoft denominados *Halloween Documents*[1]. Estes documentos internos da Microsoft vazaram para a comunidade que desenvolve software de código aberto na Internet. Nestes documentos são listados produtos de código aberto que podem vir a ameaçar o império da Microsoft. Entre eles encontra-se, é claro, o *Samba*.

De seu início modesto como um projeto pessoal, o *Samba* conta hoje com uma equipe de cerca de 18 desenvolvedores espalhados por todo o mundo. Além dos usuários fiéis de universidades, que procuram primeiramente os softwares de código aberto para resolver seus problemas, o *Samba* conta

[1] *http://www.opensource.org/halloween.html*

em sua lista de usuários pesos pesados da indústria como Xerox, Bank of America, Hewlett-Packard, Motorola, Saab, Ericsson, Siemens, o Exército Americano e muitos outros. No sítio do *Samba* existe um ítem chamado *Survey*[2], onde mais de 1500 usuários se registraram e emitiram suas opiniões sobre o software.

Mas de onde vem o nome *Samba*? Seria Andrew Tridgell um admirador secreto da alegria contagiante do nosso carnaval e um freqüentador assíduo dos desfiles da Marquês de Sapucaí? Infelizmente não. O nome *Samba* foi adotado visto que o nome original para seu software, *SMB Server*, já ser uma marca registrada de outro produto. Andrew consultou então seu verificador ortográfico e procurou palavras que contivessem as letras SMB. A palavra *Samba* era uma delas. Desta forma tivemos, de forma indireta, uma contribuição para a disseminação de um produto muito brasileiro, o carnaval e o samba ... E de quebra, um software de altíssima qualidade que está permitindo que muitas empresas ao redor do mundo economizem um bom dinheiro.

11.2 Histórico dos Protocolos

Há muitos anos atrás, no começo da era dos computadores pessoais, a IBM e a empresa SYTEC desenvolveram um sistema de comunicação em redes bastante simples para uso em pequenas redes de computadores. Este sistema implementou o NetBIOS, que significa *Network Basic Input Output System*. Traduzindo, Sistema de Redes Básico para Entrada e Saída de Dados. O nome então nos diz que seu objetivo era prover a comunicação entre computadores, com recepção e transmissão de dados. Tudo isto implementado de maneira simples. Em outras palavras, NetBIOS consiste em um software que é carregado na memória para prover uma interface entre programas e o hardware de comunicação de um computador. Entre seus componentes encontra-se um esquema de endereçamento que usa 16 bytes para identificar os computadores e as aplicações de rede. Em seguida a Mi-

[2] *http://anu.samba.org/samba/survey*

crosoft acrescentou mecanismos para redirecionar para a rede solicitações de acesso a disco para leitura e gravação, o que tornou o acesso a disco um recurso compartilhável em uma rede local. O protocolo que permite este compartilhamento de recursos veio a se chamar SMB, ou *Server Message Block*.

Uma quantidade imensa de software foi escrita para fazer uso das facilidades implementadas pela API do NetBIOS. Uma API (*Application Programmer's Interface*) consiste em uma série de convenções que devem ser seguidas pelos programadores para fazer uso dos recursos de comunicação implementados pelo protocolo SMB.

A especificação NetBIOS foi o primeiro passo. Em seguida veio o NetBEUI (*NetBios Enhanced User Interface*), criado pela IBM. O NetBEUI implementou um mecanismo que permitia que pacotes NetBIOS fossem transportados sobre redes Ethernet e Token Ring. Em seguida várias outras empresas desenvolveram a emulação NetBIOS sobre protocolos de mais alto nível como DECnet, IPX/SPX e, como não poderia deixar de ser, TCP/IP.

A combinação NetBIOS e TCP/IP resultou em uma parceria proveitosa. Uma rede NetBIOS não é roteável, ou seja, pode ser utilizada apenas em uma rede local isolada. Já o contrário acontece com o TCP/IP, utilizado em redes interconectadas, cujo maior exemplo é a Internet global, onde computadores localizados nas partes mais remotas do mundo conseguem trocar informações entre si. O protocolo NetBIOS faz uso das facilidades de interconexão dos protocolos TCP/IP através do mapeamento dos nomes NetBIOS a endereços IP. Na pilha de protocolos TCP/IP o NetBIOS passa a ser mais uma aplicação e como tal transportado de um local para outro. A forma como esta associação foi feita está descrita nas RFCs 1001[3] e 1002[4].

[3] *http://ftp.unicamp.br/pub/RFC/rfc1001.txt.gz*
[4] *http://ftp.unicamp.br/pub/RFC/rfc1002.txt.gz*

11.3 Descrição do Pacote *Samba*

O software *Samba* consiste de dois programas principais e alguns outros componentes secundários, que também serão abordados. Estes programas chamam-se *smbd* e *nmbd*.

O programa *smbd* é o componente que permite que servidores Linux compartilhem seus recursos de disco e impressão com clientes Windows. Estes serviços são fornecidos por meio do protocolo SMB ou CIFS (*Common Internet File System*).

O daemon *smbd* permite o compartilhamento de seus recursos tanto no modo de compartilhamento (*share mode*) ou no modo de usuário (*user mode*). No modo de compartilhamento, que é o modo mais simples e menos recomendável, a cada recurso compartilhado atribui-se uma senha. Se o usuário desejando utilizar tal recurso fornecer a senha correta, então se é permitido o acesso. Este enfoque tem problemas claros de segurança visto que uma senha é distribuída a todos os que desejarem acessar um recurso. Já no modo de compartilhamento em nível de usuário, cada usuário possui sua própria senha e os privilégios de acesso a recursos são outorgados pelo administrador do sistema. Este modo de acesso a recursos, além de mais seguro, é mais conveniente para os usuários, visto que, uma vez identificado pelo sistema, não existe a necessidade de se fornecer senhas várias vezes para acessar recursos diferentes. Em domínios Windows NT este controle de acesso é gerenciado pelo Controlador do Domínio (*Domain Controller*). A partir da versão 2.0, o *Samba* já provê suporte para este tipo de autenticação.

Cada sessão estabelecida gera a criação de uma cópia do programa *smbd*. Esta cópia atende a todas as conexões solicitadas pelo cliente durante a sessão. Quando todas as conexões estabelecidas pelo cliente são encerradas, a cópia do programa *smbd* criada é terminada.

O programa *smbd* irá se comportar de acordo com as definições contidas no arquivo */etc/smb.conf*. As opções de configuração do programa *smbd* são bastante extensas e oferecem um alto grau de flexibilidade. O *Samba* chega a oferecer um grau de configurabilidade que mesmo sistemas NT não

oferecem.

O outro componente do pacote *Samba*, *nmbd*, é o servidor de nomes NetBIOS. Estes servidor entende e responde a solicitações de resolução de nomes NetBIOS sobre IP. No Windows Explorer, no ítem *Network Neighborhood*, é o responsável pelo aparecimento do ícone do servidor samba. Isto se dá devido ao fato de que o daemon *nmbd* participa e responde a solicitações emitidas pelos protocolos de varredura da rede (browsing).

A resolução de nomes pode ser feita de duas formas: por meio de broadcasts e ponto a ponto. Ambos os métodos podem ser utilizados, dependendo da configuração adotada. A resolução por meio de broadcasts é a que mais se aproxima do mecanismo empregado originalmente pelo NetBIOS. Sempre que uma máquina desejar conectar-se com outro computador, uma mensagem é enviada para a rede. Todas as máquinas recebem este pacote e a máquina com a qual se deseja estabelecer a conexão responde com seu número IP.

O outro tipo de resolução de nomes envolve o uso de servidores NBNS (*NetBIOS Name Service*), ou como são mais comumente conhecidos, servidores WINS. A Microsoft batizou a sua implementação do serviço NBNS de WINS, ou *Windows Internet Name Service*. Este serviço realiza a coleta de nomes e números IP e disponibiliza esta informação para quem desejar. Os clientes enviam seus nomes NetBIOS e números IP para o servidor NBNS, que por sua vez armazena estes dados em um banco de dados. Quando um cliente desejar se comunicar com um outro cliente, ele envia o nome do outro cliente para o servidor NBNS. Se o nome constar de seu banco de dados, o servidor NBNS retorna ao solicitante o número IP.

Ao contrário do broadcast, que funciona apenas em uma rede local, clientes em redes diferentes conseguem utilizar os serviços de um único servidor WINS. Este mecanismo de resolução de nomes ponto a ponto não se restringe a uma rede local.

O daemon *nmbd*, à semelhança de servidores WINS, mantém uma lista de serviços de compartilhamento de recursos de impressão e armazenamento

oferecidas pelos clientes em uma rede. Esta lista está disponível para consulta para os computadores da rede.

Em uma rede local, os computadores participantes realizam uma eleição para decidir qual deles se tornará o *Local Master Browser* (LMB). O vencedor então se identifica reivindicando um nome NetBIOS especial (em acréscimo aos nomes que já possui). A função do LMB é manter uma lista dos serviços disponíveis e é esta lista que aparece ao se clicar no ícone *Network Neighborhood* na área de trabalho em clientes Windows.

Além dos LMBs, existem também os *Domain Master Browsers* (DMB). Os DMBs coordenam suas listas de recursos por entre domínios NT. Usando serviços oferecidos por um servidor WINS, os LMBs conseguem localizar os DMBs para trocar informações e combinar listas de recursos. Desta forma, esta lista de recursos é propagada para todos os computadores em um domínio NT. Esta propagação de recursos entretanto pode tomar um tempo considerável até que todas as mudanças estejam sincronizadas.

Sistemas Linux oferecem ainda um nível adicional de funcionalidade através do suporte a sistemas de arquivos do tipo *smbfs*. Desta forma é possível se utilizar em sistemas Linux arquivos compartilhados do ambiente Microsoft como se fossem sistemas de arquivos locais.

11.4 Instalação

A instalação do *Samba* em sistemas Linux não poderia ser mais simples. Dependendo da variante de Linux utilizada, tudo o que é necessário é localizar um sítio na Internet que distribua os binários para sua arquitetura e instalá-los. O *Samba* é parte integrante das distribuições *Conectiva Linux*.

O sítio do *Samba*, distribui os binários para praticamente todas as variantes Unix existentes. Entretanto, se você for um dos poucos felizardos que trabalha em uma arquitetura não muito comum, não se desespere. O código fonte do *Samba*, seguindo a tradição de todos os softwares abertos, também está disponível. A instalação a partir do código fonte normalmente se dá se

maiores problemas.

Para sistemas *Conectiva Linux*, a instalação é feita através do gerenciador de pacotes do RPM (*RedHat Package Manager*). A instalação deve ser feita a partir da conta do superusuário(*root*):

```
# rpm -i sambaxxx.rpm
```

Através deste único comando todas as tarefas de configuração são executadas. Os binários são colocados em */usr/bin/*, a documentação é colocada em */usr/doc/samba-xxx*, um arquivo padrão de configuração é copiado para */etc/* e assim por diante. Além da instalação de todos os binários, arquivos de documentação e configuração, são também configurados os scripts de inicialização para que já durante o boot estes serviços (*nmbd* e *smbd*) sejam ativados.

A opção de se instalar ou não o *Samba* é dada a usuários de sistemas *Conectiva Linux* já na instalação do sistema, através da seleção dos serviços SMB.

A instalação a partir do código fonte, para versões iguais ou superiores a 2.0, é bastante simples. Obtenha o código fonte de algum sítio de distribuição do samba, e expanda-o em algum diretório de seu sistema. A regra não oficial vigente em ambientes Unix é que todo o código fonte seja expandido no diretório */usr/local/src*:

```
# cd /usr/local/src                              [1]
# gzip -dc sambaxxx.tar.gz | tar xvf -           [2]
# ln -s sambaxxx samba                           [3]
# cd samba                                       [4]
# ./configure                                    [5]
# make                                           [6]
# make install                                   [7]
```

Nas linhas acima, os números entre colchetes foram incluídos apenas para facilitar a explanação que se segue.

Em [1] deslocamo-nos para o diretório */usr/local/src*, onde se encontra armazenado o código fonte de todos os programas instalados em nosso sistema. O passo [2] realiza a expansão dos arquivos integrantes do software *Samba*[5]. A distribuição original consiste de um arquivo no formato tar compactado com o programa gzip. O comando *gzip -dc* realiza a descompactação do arquivo *sambaxxx.tar.gz*[6] e o resultado é fornecido como entrada para o programa tar, que faz então a expansão final. O passo seguinte [3] cria um link de nome samba, para o diretório criado em [2]. O comando executado em [5] faz uma análise do ambiente computacional, define diversas variáveis e cria um arquivo *Makefile* apropriado para a compilação final do produto. Embora seja possível se fornecer uma enorme variedade de opções ao programa *configure* a minha recomendação é que sejam usados os valores padrão, ou seja, aqueles definidos pelo comando *configure* invocado sem se fornecer nenhum argumento. Mas se mesmo assim você desejar conhecer as opções disponíveis, basta invocar o comando *configure -help*. A etapa final [7] realiza a instalação do produto no sistema, criando a árvore de diretórios necessária e copiando os binários e outros arquivos relevantes para os locais definitivos.

O pacote RPM (*RedHat Package Manager*) que contém o software *Samba* em formato pré-compilado, pronto para instalação em sistemas *Conectiva Linux*, contém 271 arquivos, dos quais 217 são arquivos documentando os vários componentes do software. Não é por falta de informação que o *Samba* não vai funcionar em seu sistema, não?

Neste ambiente, os 271 arquivos são distribuídos pelos seguintes diretórios:

```
/etc/codepages
/usr/bin
/usr/sbin
/usr/doc/samba-xxx
/usr/lib
```

[5] O programa *tar* da GNU, padrão em sistemas Linux, pode realizar a descompressão simultaneamente: `tar xvzf sambaxxx.tar.gz`

[6] Os caracteres *xxx* indicam a versão do *Samba* utilizada

```
/usr/man
/usr/share/
/var
```

11.5 Configuração

O *Samba* pode ser configurado de inúmeras formas, adequando-se a praticamente qualquer tipo de ambiente. O *Samba* pode ser usado como servidor de arquivos e de impressão, pode utilizar os serviços de autenticação de um servidor Windows NT (*Primary Domain Controller*) ou então realizar ele próprio esta autenticação. As versões mais recentes, ainda em desenvolvimento, permitem até mesmo que um servidor *Samba* atue como um *Primary Domain Controller* (PDC).

O coração de tudo está no arquivo *smb.conf*. Em sistemas *Conectiva Linux* este arquivo de configuração fica no diretório */etc*. O arquivo é fartamente comentado e permite que configurações mais simples sejam feitas simplesmente através da leitura destas informações e da modificação de alguns poucos parâmetros.

O arquivo é dividido em três seções principais denominadas `[global]`, `[home]` e `[printers]`. O nome da seção (à exceção da seção `[global]`) é o nome do recurso compartilhado e os parâmetros dentro delas definem os atributos destes recursos.

O parâmetro `[global]` define os parâmetros padrão do sistema (defaults). Como veremos nos exemplos que se seguem, à exceção de alguns poucos parâmetros, normalmente não necessitamos nos preocupar muito com estas definições. Os valores padrão assumidos pelo *Samba* são via de regra bastante razoáveis.

Em `[homes]` são definidas as permissões padrão para os diretórios de usuários quando montados a partir de computadores NT ou Windows. Em `[printers]` temos a definição de permissão de acesso às impressoras compartilhadas pelo servidor *Samba*.

Um compartilhamento (share) consiste de um diretório ao qual se está permitindo o acesso acrescido de uma descrição dos direitos de acesso outorgados ao usuário do serviço, além de outras opções. As sessões podem descrever recursos de disco compartilhados ou impressoras.

A sessão denominada `[global]` especifica opções que se aplicam ao servidor como um todo, ou definem parâmetros padrão (default). A seguir, um breve exemplo de definição da sessão `[global]` do arquivo. Todos os comentários foram removidos para uma melhor visualização dos parâmetros.

```
[global]
  workgroup = computacao
  hosts allow = 10.0.0. 127.
  server string = Servidor \emph{Samba} - Computacao
  printcap name = /etc/printcap
  load printers = yes
  log file = /var/log/samba/log.%m
  max log size = 50
  security = server
  password server = 10.0.0.1
  encrypt passwords = yes
```

A seguir incluímos a descrição dos ítens acima:

- workgroup = computacao

  Esta opção define define o nome de um grupo de trabalho ou o nome de um domínio NT. Em nosso exemplo atribuímos o valor `computacao` a esta opção

- hosts allow = 10.0.0. 127.

  Aqui são especificadas as máquinas que estão autorizadas a se conectarem a este servidor. Os valores 10.0.0. e 127. indicam que apenas computadores pertencentes a classe A de número 10.0.0. e a própria máquina (por meio da especificação do endereço de loopback 127.) estão autorizadas a usar os serviços deste servidor samba.

- server string = Servidor *Samba* - Dep.de Computação

  Este valor é equivalente ao campo de descrição de um servidor NT (*NT Description Field*). É o valor que aparece ao lado do nome do computador quando exibido na janela do Windows Explorer.

- printcap name = /etc/printcap

  Em sistemas Unix, as impressoras costumam ser definidas em um arquivo chamado *printcap*, que normalmente fica no diretório */etc*. Esta opção permite que seja especificada uma outra localização para este arquivo. No nosso exemplo foi mantida a localização padrão.

- load printers = yes

  Esta diretiva sinaliza ao servidor *Samba* para realizar a carga automática da lista de impressoras ao invés de optar pela carga individual de cada uma delas. Esta lista é obtida do arquivo *printcap*.

- log file = /var/log/samba/log.%m

  Como especificado acima, cada máquina que se conectar a este servidor *Samba* terá um arquivo de log separado indicando todas as solicitações feitas e tudo o mais que se relacionar com estas conexões. Esta separação é bastante útil, principalmente quando se está realizando a configuração inicial, permitindo identificar rapidamente qualquer erro.

- max log size = 50

  Este é o valor máximo em Kb que os arquivos de log podem ter.

- security = server

  O modo de segurança em que o servidor samba irá operar. As opções possíveis são *share*, *user* e *server*. Na opção *share* (padrão), o cliente se autentica separadamente para cada recurso que desejar acessar. A senha é enviada juntamente com cada solicitação de acesso. Já na opção *user* o cliente envia o nome do usuário e a senha quando do estabelecimento da sessão. Neste momento o servidor não sabe qual recurso será solicitado. Se o servidor aceitar a conexão (identificação e senha válidas) então o cliente terá direito de acesso a todos os

recursos para os quais está autorizado. Na opção *user*, o acesso aos recursos compartilhados será permitido ou negado levando-se em conta as permissões de acesso dos usuários relativamente ao próprio recurso. Na opção *server*, o servidor *Samba* envia ao servidor de senhas (`password server`, próxima opção), o conjunto *identificação/senha* da mesma forma como recebeu. Se o servidor de senhas confirmar a identificação, o servidor *Samba* aceita a conexão. Isto permite que um servidor *Samba* utilize os serviços de outro servidor SMB para autenticação das conexões. Para maiores informações consultar o documento *security_level.txt*, distribuído juntamente com o *Samba*.

- password server = 10.0.0.1

  Temos aqui o endereço IP do servidor SMB que irá autenticar os usuários que desejarem se utilizar dos serviços do servidor *Samba*.

- encrypt passwords = yes

  Se possível, sempre especificar esta opção como *yes*, para evitar que senhas sejam descobertas através da análise dos pacotes circulando na rede (*packet sniffers*). Dependendo de seu ambiente, podem existir algumas restrições que impeçam ou dificultem a criptografia. Para maiores detalhes ler os documentos *ENCRIPTION.txt*, *Win95.txt* e *WinNT.txt* da documentação do *Samba*.

```
[Homes]
  comment = Diretórios de Trabalho
  browseable = no
  writable = yes
```

Esta seção define os diretórios dos usuários. Basicamente se especifica que estes diretórios não estão disponíveis para visualização pública (browseable = no). Apenas usuários autorizados conseguirão examinar seu conteúdo. Estes diretórios podem ser montados para gravação uma vez que o usuário tenha sido autenticado (writable = yes).

Até aqui, a configuração fornecida especifica que o servidor *Samba* irá utilizar como servidor SMB, para autenticação de usuários, o computador cujo endereço IP é 10.0.0.1. Uma vez que este usuário tenha sido autenticado, ele terá acesso de leitura e gravação a um diretório existente no servidor *Samba*. As senhas serão criptografadas e apenas os computadores da rede 10.0.0. e o próprio servidor *Samba* terão acesso aos arquivos.

Do arquivo */etc/smb.conf* original, apenas cinco opções foram modificadas para disponibilizar este servidor *Samba* como servidor de arquivos para um domínio NT:

```
workgroup = computacao
hosts allow = 10.0.0. 127.
security = server
password server = 10.0.0.1
encrypt passwords = yes
```

11.6 Teste da Sintaxe do Arquivo de Configuração

Uma vez modificado o arquivo */etc/smb.conf* para atender às especificações de sua rede, é possível realizar um teste da configuração através do utilitário *testparm*. A melhor forma de se usar este utilitário é redirecionando a saída para um arquivo e em seguida analisando-se o resultado. Importante, o script fica aguardando que se pressione alguma tecla para encerrar o processamento. O resultado gerado exibe uma lista completa das variáveis e os valores de cada uma delas. A mensagem *Loaded services file OK*, como abaixo, indica que o arquivo foi processado sem erros.

```
# testparm /tmp/testparm.out
# vi /tmp/testparm.out
Load smb config files from /etc/
Processing section '[homes]'
Processing section '[printers]'
```

```
Loaded services file OK.
Press enter to see a dump of your service definitions
# Global parameters
workgroup = MYGROUP
netbios name =
netbios aliases =
server string = \emph{Samba} Server
interfaces =
bind interfaces only = No
security = USER
encrypt passwords = No
update encrypted = No
use rhosts = No
```

... várias linhas omitidas

Simulemos então um erro de configuração no arquivo. Ativemos simultaneamente os valores *wins support = yes* e *wins server = wins*. Estes valores são mutuamente excludentes, pois especificam que o servidor *Samba* irá atuar como um servidor Wins e ao mesmo tempo especificamos o nome de um outro servidor Wins. Ao executar o comando *testparm* neste arquivo de configuração temos:

```
# testparm
Load smb config files from /etc/
Processing section [homes]
Processing section [printers]
Loaded services file OK.
ERROR: both 'wins support = true' and 'wins server = '
cannot be set in the file. nmbd will abort with this setting.
Press enter to see a dump of your service definitions
```

O erro foi corretamente identificado e somos avisados de que o programa *nmbd* não irá funcionar com esta configuração. Pela análise do arquivo de saída é possível se identificar quaisquer erros eventualmente cometidos

11.7 Conclusão

Este documento certamente não esgota todos os recursos oferecidos pelo software *Samba*. Foram ilustradas algumas das possibilidades e também exemplificado como, de uma forma simples e com a alteração de alguns poucos parâmetros, se é possível configurar um servidor *Samba*. Como já dito anteriormente, um dos pontos fortes do *Samba* é justamente sua documentação abundante onde encontramos informação sobre praticamente todos os aspectos relativos ao seu uso e configuração.

11.8 Referências

Para a criação deste documento consultou-se um grande número de artigos publicados em revistas eletrônicas, diversos tutoriais e a própria documentação do pacote. Estes documentos podem ser encontrados no endereço *http://www.Dicas-L.unicamp.br/hotlinks/Samba/index.html*.

Foi acrescentado ao CD-ROM o original do Livro *Using Samba* nos formatos PostScript e PDF. Tais arquivos encontram-se no diretório */home/using_samba*.

Capítulo 12

vi – Visual Editor

12.1 Introdução

Administradores de sistemas freqüentemente se vêm às voltas com a necessidade de editar uma grande quantidade de arquivos. O *Conectiva Linux* possui uma grande infinidade de editores de texto, adequados a todos os gostos.

De todos estes editores, os mais populares entre os administradores de sistemas são o *vi* e o *emacs*. O editor *vi*, desenvolvido por Bill Joy, na Universidade da Califórnia, é o único editor de texto que certamente encontraremos em todas as variantes de Unix. Em sistemas que não estejam funcionando direito ou mesmo sistemas que não tenhamos tido a oportunidade de configurar de acordo com nossas preferências, podemos estar certo de lá encontrar o bom e velho *vi*.

O *emacs*, também infinitamente poderoso, possui uma enorme legião de admiradores. O *vi* não fica atrás e possui também um grande número de amantes. A questão de qual editor é o melhor, *vi* ou *emacs*, chega às raias da guerra religiosa. Apesar de tudo isto, o mais importante é escolher um editor de textos e investir uma parcela de seu tempo no aprendizado aprofundado de seus recursos. Certamente este investimento será amplamente

recompensado com o ganho de produtividade que se obterá.

Embora de aprendizado um pouco difícil, e bastante irritante por sua insistência em continuamente emitir alertas sonoros quando cometemos erros, o *vi* certamente é um editor de textos bastante poderoso com o qual podemos executar uma infinidade de tarefas que mesmo editores mais populares e supostamente mais poderosos não conseguem.

O *Conectiva Linux* possui uma vantagem adicional. O editor *vi* tradicional foi substituído pelo *vim* (*Vi IMproved*), ainda mais poderoso e amigável.

Pressupomos que o leitor possua um conhecimento básico do *vi* para poder entender as operações que serão apresentadas nas próximas páginas. Foram selecionadas as operações mais freqüentemente utilizadas no dia a dia de um administrador de sistemas e que certamente contribuirão para economizar um bom tempo de quem se empenhar em dominá-las corretamente.

Além do mais, todo administrador de sistemas Linux digno do nome tem obrigatoriamente que dominar as peculiaridades do editor *vi*, mesmo que um dia venha a aprender o *emacs*.

12.2 Substituição de Caracteres

Às vezes desejamos substituir caracteres que se encontram em determinada posição fixa em um texto, mas apenas aqueles caracteres e não atuar em caracteres semelhantes que se encontram em outra posição.

Tomemos o seguinte arquivo:

```
12345abcde12345abcdeabcde12345
12345abcde12345abcdeabcde12345
abcdeabcde12345abcdeabcde12345
```

Eu quero transformar a cadeia de caracteres *abcde*, a partir da coluna 6, em *ABCDE*.

O comando

```
%s/^\(.....\)abcde/\1ABCDE/
```

faz exatamente o que preciso, resultando em

```
12345ABCDE12345abcdeabcde12345
12345ABCDE12345abcdeabcde12345
abcdeABCDE12345abcdeabcde12345
```

O caractere "\1" instrui o *vi* para deixar intactos os caracteres anteriores, representados por ".....". Caso não houvéssemos digitado caractere "\1" o resultado seria:

```
ABCDE12345abcdeabcde12345
ABCDE12345abcdeabcde12345
ABCDE12345abcdeabcde12345
```

Como visto acima, os caracteres que precedem a cadeia de caracteres que eu desejava substituir foram apagados.

12.3 Substituição de Caracteres de Controle

Existem alguns caracteres utilizados pelo *vi* que são empregados como caracteres de controle. Isto pode causar alguns inconvenientes, como por exemplo, se desejarmos mudar todas as ocorrências da cadeia de caracteres */home/cesar/bin/local-programs* por */usr/bin/local-programs*, teríamos que digitar o seguinte comando:

```
:%s/\/home\/cesar\/bin\/local/\/usr\/bin\/local/g
```

Notem que todos os caracteres "/" precisam ser precedidos pelo caractere "\". Isto se dá devido ao fato de que o caractere "/" é normalmente utilizado como um separador. Para ser entendido literalmente e não como um caractere de controle, devemos precedê-lo por outro que remova o seu significado especial, como se pode ver no exemplo anterior.

É fácil de se concluir que situações como esta são bastante propensas a erros e cansativas, se tiverem que ser repetidas muitas vezes.

Mas uma característica muito pouco conhecida do *vi* é que nós podemos substituir a "/" por qualquer caractere que desejarmos. Desta forma, o exemplo acima pode ser escrito da seguinte forma:

```
:%s:/home/cesar/bin/local:/usr/bin/local:g
  ^                    ^              ^
```

Neste caso o caractere "/" foi substituído pelo caractere ":" tornando desnecessário que se preceda "/" por "\". É claro que se o caractere ":" aparecesse na cadeia de caracteres a ser substituída ele teria que ser precedido por "\".

```
:%s:/home/queiroz\::/home/rubens:g
                  ^^
```

12.4 Opção ignorecase

Para localizar palavras ou caracteres dentro de um texto de forma a não se diferenciar entre caracteres maiúsculos e minúsculos, especifique a opção *ignorecase*:

```
set ignorecase
```

ou abreviadamente

```
set ic
```

Desta forma a busca pela palavra *Roma* vai identificar as palavras *roma, rOma*, e qualquer combinação possível.

A qualquer momento que se queira desativar esta opção basta especificar:

```
set noic
```

Esta opção pode ser tornada padrão através da inclusão de uma linha contendo as diretivas *set ic* dentro do arquivo *.exrc* ou *.vimrc*, que são os arquivos lidos pelo *vi* para obter sua parametrização inicial.

12.5 Inclusão de Resultados de Comandos do Sistema

Para incluir dentro do texto que se está editando a saída de um comando executar:

```
:r!cmd
```

O comando

```
:r!ls /tmp
```

incluirá dentro do texto as seguintes linhas

```
Nm8CZt
PhgDM2
PjgDmb
Q6ADM2
```

equivalentes à saída do comando *ls* executada sobre o diretório */tmp*.

12.6 Substituição Global Interativa

O *vi* oferece a facilidade de se fazer uma substituição global interativa, ou seja, a cada cadeia de caracteres a ser substituída, o usuário precisa confirmar se deseja ou não que a substituição seja efetuada.

Por exemplo, para substituir todas as ocorrências da cadeia de caracteres *home* por *usr* emitir o comando:

```
:%s/home/usr/c
             ^
```

O sinal *c* ao final do comando indica que se deseja a confirmação antes de se efetuar as mudanças.

A cadeia de caracteres a ser substituída aparece sublinhada pelo caractere "^", como indicado no exemplo abaixo.

O usuário deve então pressionar a tecla *y* para efetuar a substituição ou a tecla *n* em caso contrário.

```
/home/cesar/supsof/queiroz
 ^^^^
```

12.7 Deleção de Linhas em Branco

Para removermos todas as linhas em branco de um arquivo podemos utilizar o comando:

```
:global /^$/ delete
```

A diretiva *global* indica que o comando se aplica a todo o arquivo, ou seja tem um efeito *global*.

Este comando pode ser abreviado como

```
:g /^$/d
```

O metacaractere "^" indica o começo e "$" indica o fim da linha. Desta forma, os dois caracteres conjugados ("^$") indicam uma linha que não contem nada.

12.8 Inversão da ordem das linhas em um arquivo

Se desejarmos que a primeira linha de um arquivo seja a última e a última a primeira, o comando abaixo realiza esta tarefa:

```
:global /^/ m 0
```

ou, simplificadamente

```
:g/^/m0
```

O caractere "^" marca todas as linhas no texto e a diretiva "m" move todas as linhas marcadas para a primeira posição.

12.9 Substituição utilizando metacaracteres

Caso queiramos substituir em um arquivo, todas as cinco primeiras colunas, independentemente de seu conteúdo, pela cadeia de caracteres "12345", podemos utilizar o comando:

```
:%s/^...../12345/
```

O comando acima, no arquivo abaixo

```
abcdet.....................
fghij$$$.....................
klmnoMMMMM...................
pqrst.......................
```

resultará em

```
12345t.....................
12345$$$.....................
12345MMMMM...................
12345.......................
```

12.10 Gravação Seletiva

O comando

```
:global /^Capítulo [0-9]$/ . w >> indice.txt
```

irá varrer todo o arquivo e gravar no arquivo *indice.txt* todas as linhas que contenham a cadeia de caracteres *Capítulo* seguida de um espaço em branco e um algarismo variando de 0 a 9.

O comando

```
:global /^Capítulo [0-9]$/ . copy $
```

irá efetuar a mesma pesquisa, porém as linhas que se encaixarem no argumento de pesquisa serão copiadas para o final do arquivo.

12.11 Deleção de Colunas

Suponhamos o seguinte arquivo:

```
12345,---------------------------
12345,---------------------------
12345,---------------------------
12345,---------------------------
```

O comando

```
:%s/,.*//
```

irá apagar todos os caracteres situados após a vírgula, inclusive. O arquivo resultante será

```
12345
12345
12345
12345
```

É claro que a vírgula pode ser qualquer caractere. O "." significa o caractere após a vírgula, qualquer que seja ele e o "*" indica todos os demais caracteres em seguida. Todos estes caracteres serão substituídos por nada ("//").

12.12 Inserção de Linhas

Suponhamos que você queira incluir uma linha pontilhada no final das linhas de título de um documento. Estas linhas são identificadas por um terminador especial, tal como o caractere "-".

```
1. Título-
blablabla
blablabla
blablabla
blablabla
```

Para mudar basta fazer:

```
:%s/-$/^M----------------------------------^M/
```

e temos então

```
1. Titulo
---------------------------------
blablabla
blablabla
blablabla
blablabla
```

O caractere "^M" é um caractere de controle. Para inserir caracteres de controle em documentos editados pelo *vi,* você deve preceder o caractere desejado por CTRL-V. No nosso caso a sequência é CTRL-V CTRL-M. O caractere CTRL-M indica a quebra de linha, que é feita após o caractere "-" e ao fim da linha, para evitar que a linha tracejada e a próxima fiquem juntas.

12.13 Diretório Temporário Alternativo

Muitas vezes o diretório */tmp* fica lotado e você não consegue editar arquivos com o editor *vi.* E pior, você não consegue achar o administrador de sistemas para fazer isto para você ou, pior ainda, o administrador de sistemas é um sádico que gosta de fazer os usuários sofrerem. Às vezes acontece...

Existe a possibilidade de se definir, provisória ou permanentemente, um diretório que não o */tmp* como diretório de trabalho.

Para isto, proceda da seguinte forma:

```
% vi
```

Não forneça argumentos, pois o *vi* não conseguirá carregar o arquivo pois o */tmp* está lotado.

```
:set directory=/home/usuario/tmp
```

Esta diretiva você muda conforme a sua conveniência.

E pronto, o *vi* vai usar o diretório especificado e o problema está resolvido.

Esta definição pode ser permanente. Basta inserir esta definição diretamente no arquivo *.vimrc*:

```
set directory=/home/usuario/tmp
```

12.14 Posicionamento Automático

O vi possui um recurso poderoso para programadores, que é o de encontrar os pares de caracteres delimitadores de funções como "{", "}", "[", "]", "(" e ")". Posicionando o cursor sobre um destes caracteres e pressionando-se a tecla "%", o cursor irá se posicionar automaticamente sobre o caractere que abre ou fecha, conforme for o caso, a seção em questão.

12.15 Abreviações

Outro recurso extremamente útil é a possibilidade de fazer com que ao se digitar um grupo pré-determinado de caracteres os mesmos sejam expandidos para uma cadeia de caracteres definida. Isto é particularmente útil para simplificar a digitação de palavras que ocorrem com freqüência.

Por exemplo, para facilitar a inserção do endereço eletrônico *usuario@dominio.com.br* podemos fazer uma associação deste endereço às letras ml. Sempre que for digitado *ml* o texto é expandido para *mailto:usuario@dominio.com.br*. Com este artifício economizamos 27 toques. Nada mal

para nós que vivemos neste mundo da informática, afligido pela *Lesão de Esforço Repetitivo* (L.E.R.).

Isto é bastante simples de se fazer. Basta editar o arquivo *.vimrc* e incluir linhas do tipo:

```
ab ml mailto:usuario@domain.com.br
```

Importante, quando criar as suas abreviações, não escolha uma seqüência de caracteres que seja comum na língua portuguesa. Se você criar uma abreviação como:

```
ab    sa       #!/bin/sh
```

você se verá na incômoda situação de ver todos os "sa" que digitar transformados em *#!/bin/sh*. Para desabreviar, existe o comando *unab*, que deve ser executado em modo de comando:

```
unab sa
```

12.16 Inserção e Quebra de Linhas

Além disto você pode incluir textos em múltiplas linhas. Basta inserir no ponto onde será feita a quebra de linha o caractere "^". Por exemplo:

```
ab vc --^MVocabulário
```

Esta abreviação eu uso freqüentemente em outra lista que mantenho, chamada EFR (*English For Reading*). A lista envia diariamente uma piada ou citação em inglês com comentários sobre as palavras mais difíceis. Para simplificar a minha vida eu criei esta abreviação de forma que quando digito *vc* tenho

```
--
Vocabulário
```

Então, pense em seus hábitos de digitação e crie algumas abreviações para lhe ajudar. Economize toques e dê um descanso aos seus dedos, pulso, e braço. Isto é sério!

12.17 Criação de Abreviações para Edição de Arquivos HTML

Uma aplicação interessante das abreviações é a criação de macros que permitam a criação automática de diretivas HTML.

Abaixo incluo um exemplo de um arquivo *.exrc* com diretivas para criação das macros html com algumas explicações.

Pode-se ver que com este recurso o *vi* pode desempenhar tarefas extremamente complexas, reduzindo o stress causando pela digitação constante. Basta deixar a imaginação fluir ...

exrc.html

```
"
"          Editor HTML
"
"   ab .... abreviações (utilizar em modo de inserção)
"
"        xy  gera <XY>,
"       nxy  gera </XY>
"
ab pg <HTML><HEAD><TITLE>#</TITLE>
</HEAD><BODY>
</BODY></HTML>
```

```
ab hr <HR>
ab pp <P>
ab ht <HTML>
ab nht </HTML>
ab hd <HEAD>
ab nhd </HEAD>
ab ti <TITLE>
ab nti </TITLE>
ab bd <BODY>
ab nbd </BODY>
ab str <STRONG>
ab nstr </STRONG>
ab bo <B>
ab nbo </B>
ab it <I>
ab nit </I>
ab pre <PRE>
ab npre </PRE>
ab ul <UL>
ab nul </UL>
ab ol <OL>
ab nol </OL>
ab li <LI>
ab dl <DL>
ab ndl </DL>
ab dt <DT>
ab dd <DD>
ab adr <ADDRESS>
ab nadr </ADDRESS>
ab h1 <H1>
ab nh1 </H1>
ab h2 <H2>
ab nh2 </H2>
ab h3 <H3>
```

```
ab nh3 </H3>
ab h4 <H4>
ab nh4 </H4>
ab h5 <H5>
ab nh5 </H5>
ab h6 <H6>
ab nh6 </H6>
ab ig <IMG SRC="#">
ab fr <FORM ACTION="#" METHOD="#">
ab nfr </FORM>
```

12.18 Mapeamento de Teclas

O editor *vi* nos permite fazer com que uma série de comandos seja executada simplesmente pressionando-se uma tecla. Esta facilidade é explorada com o comando *:map*.

Programadores podem achar útil um comando que insira comentários automaticamente na linha em que se encontra o cursor. Podemos fazer isto inserindo no arquivo *.vimrc* a seguinte linha, que fará com que ao se pressionar a tecla "@" a linha onde se encontra o cursor seja delimitada à esquerda e à direita pelos caracteres "*/**" e "***/*" respectivamente.

```
:map @ I/* <esc>A */<esc>0
```

Vejamos o que acontece. Ao se teclar, em modo de comandos, o caractere "@", será inserido no começo da linha ("I") os caracteres */**", que representa o início de um comentário. Em seguida, será acionada a tecla *<esc>*, retornando ao modo de comando. O cursor será então posicionado no final da linha (*A*), em modo de inserção, e em seguida será teclado novamente *<esc>*, para retornar ao modo de comandos. O "0" faz com que o cursor retorne ao começo da linha.

Complicado, não? A vantagem é que uma vez que tenhamos inserido esta

definição no arquivo *.vimrc* tudo o que precisaremos fazer para transformar uma linha em comentário é pressionar a tecla "@". Na verdade não tem mistério algum, basta escrever todos os passos seguidos.

Sempre tente usar para fazer o mapeamento de comandos caracteres pouco utilizados, especialmente se o comando for muito complexo e efetuar modificações de grande porte nos documentos. Enganos acontecem.

Outro exemplo:

```
:map + GoJ. Marcos^Memail:marc@abc.com.br^MFone:2222222<esc>
```

Esta diretiva instrui o editor *vi* a inserir, sempre que for pressionada a tecla + em modo de comando, no final do arquivo (G posiciona o cursor na última linha e "o" insere uma nova linha), as seguintes linhas

```
J. Marcos
email:marc@abc.com.br
Fone:2222222
```

Observe o caractere "^M" ao final de cada linha. Este caractere ocasiona uma quebra de linha, formatando o texto como desejado.

12.19 Substituições

Para se mudar todas as ocorrências das palavras *hot*, *hit*, *h0t*, *hat* por *host* (ou qualquer palavra de três letras iniciada em h e terminada em t) mas apenas nas linhas em que estas palavras ocorram isoladamente:

```
:%s/^h.t$/host/
```

Caso realmente queiramos substituir apenas as palavras *hot*, *hit*, *h0t* e *hat* basta substituir o ".", que é uma espécie de coringa que aceita qualquer caractere, pela seqüência de caracteres aceitáveis:

```
:%s/^h[oi0a]t$/host/
```

12.20 Desfazendo Alterações

Versões tradicionais do editor *vi*, distribuídas juntamente com o sistema operacional, possuem a facilidade de desfazer comandos emitidos ou alterações feitas no texto.

Se após realizarmos uma mudança global descobrirmos que o resultado não foi exatamente o esperado, basta digitar, em modo de comando, a letra "u", e tudo volta ao estado anterior. A letra "U" desfaz as modificações efetuadas apenas na linha corrente. A limitação é que o nível de modificações que podemos cancelar, em editores *vi* comuns, é apenas um comando.

Já o clone do *vi* chamado *vim*, padrão no *Conectiva Linux* além de oferecer muito mais facilidades, pode desfazer múltiplas alterações. Desta forma, caso nos arrependamos de vários comandos emitidos, basta pressionar repetidamente a tecla *u* até chegar no ponto que desejarmos.

12.21 Releitura de Arquivos

Uma outra alternativa, após uma ou mais alterações mal sucedidas e não salvas, é recarregar o arquivo inteiro. Isto pode ser feito com o comando

```
:e!
```

Todo o arquivo é recarregado com o formato em que estava em seguida à última gravação (*w*).

12.22 Substituições

Para substituir caracteres em um documento, usar em modo de comando a diretiva "*s*":

```
:s/joao/maria/
```

O comando acima substitui, na linha corrente, a primeira ocorrência da palavra *joao* por *maria*. Na linha abaixo

```
joao maria joao maria joao maria
```

após a execução do comando acima temos:

```
maria maria joao maria joao maria
```

Ainda ficamos com alguns *joao*.

Para se substituir todas as ocorrências, na linha corrente, da palavra "joao" por "maria":

```
:s/joao/maria/g
```

Se quisermos que a substituição se aplique ao documento inteiro

```
:%s/joao/maria/g
```

E ficamos então sem nenhum "joao", só com "maria".

12.23 Substituições em Intervalos de Linhas

Podemos definir um intervalo onde determinados caracteres serão substituídos por outros. Este intervalo é delimitado por determinada seqüência de caracteres.

Tomemos o texto abaixo:

```
111111111111
222222222222
333333333333
111111111111
333333333333
```

O comando

```
:/1/,/3/s/1/A/g
```

resulta em

```
AAAAAAAAAAA
222222222222
333333333333
111111111111
333333333333
```

A faixa de linhas é determinada por */1/* e */2/*. O editor *vi* busca um intervalo delimitado desta forma e apenas neste intervalo aplica a substituição. Note que no resultado do comando apenas a primeira linha foi afetada. A quarta linha, também contendo o caractere "1" foi mantida inalterada.

12.24 Opções de Gravação

Para salvar o trabalho realizado no *vi* basta usar o comando "w". Entretanto é possível se fazer muito mais. O comando:

```
:1,10 w parte1.txt
```

Grava no arquivo parte1.txt as dez primeiras linhas.

Já o comando

```
:100,110 w >> parte1.txt
```

adiciona ao conteúdo do arquivo parte1.txt o conteúdo das linhas de número 100 a 110.

```
:/Capítulo 1/,/Capítulo 2/ w >>  livro.txt
```

Adiciona ao conteúdo do arquivo livro.txt todo o texto compreendido entre a ocorrência da cadeia de caracteres *Capítulo 1* e *Capítulo 2*.

12.25 Remoção de Espaços em Branco

Para remover todos os espaços em branco consecutivos de um texto, utilize o comando:

```
:%s/␣␣*/␣/g
```

Tomemos o seguinte texto:

```
João␣␣␣␣␣␣se␣␣␣casou␣␣␣␣com␣Maria,␣mas␣␣␣o␣casamento
durou␣␣␣␣␣pouco.␣␣␣␣␣␣␣Maria␣␣␣␣␣␣não␣␣␣␣␣␣␣amava␣João.
```

Após a execução do comando acima temos:

```
João␣se␣casou␣com␣Maria,␣mas␣o␣casamento
durou␣pouco.␣Maria␣não␣amava␣João.
```

O caractere "*" significa a ocorrência de zero ou mais caracteres iguais ao caractere que o precede, em nosso caso, o espaço em branco. Note bem que são DOIS espaços em branco no argumento de busca. Sempre que se encontrar um ou mais espaços em branco o comando irá substituí-los por apenas um.

12.26 Busca e Substituição

É bastante comum nos encontrarmos em situações onde precisamos buscar alguma cadeia de caracteres e acrescentarmos alguma coisa ao que for encontrado.

Façamos o acréscimo a todos os valores numéricos encontrados em um texto dos caracteres "R$".

```
:%s/\([0-9][0-9]*\,[0-9]*\)/R$ \1/g
```

O parâmetro principal de busca, caracteres numéricos indicando valores monetários, é delimitado por parênteses. Uma vez localizados caracteres que atendam ao argumento de busca, a eles são acrescidos os caracteres "R$", seguidos de um espaço em branco. O sinal "\1", indica que neste ponto devem ser inseridos os caracteres localizados pelo parâmetro de busca.

12.27 Deleção de caracteres

Para remover um número determinado de caracteres de um texto, podemos usar o seguinte comando:

```
:%s/^.....//
```

O "." se aplica a qualquer caractere. Desta forma, os primeiros cinco caracteres contados a partir do início da linha (indicado pelo caractere "^") serão eliminados.

Um arquivo contendo

```
12345abcde
12345abcde
12345abcde
```

após a execução do comando acima, ficará da seguinte forma:

```
abcde
abcde
abcde
```

Da mesma forma, se quisermos apagar os últimos cinco caracteres do mesmo arquivo:

```
:%s/.....$//
```

e o nosso arquivo ficará da seguinte forma:

```
12345
12345
12345
```

12.28 Edição de Múltiplos Arquivos

Normalmente o *vi* é invocado fornecendo-se como argumento o nome de um arquivo. Entretanto é possível especificar-se o nome de vários arquivos, como em

```
$ vi arquivo00.txt arquivo01.txt arquivo02.txt
```

ou ainda

```
$ vi arquivo*
```

ou mesmo

```
$ vi *
```

Após encerrada a edição do primeiro arquivo e salvo o trabalho, para passar ao próximo arquivo da lista basta digitar, em modo de comando, "*:n*". Caso tentemos encerrar a edição digitando "*:wq*", somos então alertados do número de arquivos que ainda não foram editados:

```
23 more files to edit
```

Para retornar ao primeiro arquivo sendo editado basta fornecer o comando "*:rew*".

Para realmente encerrar a edição, devemos digitar "*:q!*".

Outro recurso particularmente útil para quem está editando múltiplos arquivos, é saber exatamente em que ponto da edição nos encontramos. Digitando CTRL-G em modo de comando obtermos exatamente esta informação:

```
vi.tex  [Modified] line 925 of 1016 --91%-- col 1
```

Como visto, em primeiro lugar temos o nome do arquivo, seguido pelo número da linha em que nos encontramos e do número total de linhas. Temos também o percentual do arquivo já visto e a coluna em que nos encontramos.

12.29 Troca de Caixa

Para fazermos com que determinados caracteres em um texto sejam convertidos de maiúsculas para minúsculas, posicionamos o cursor sobre eles

e digitamos " ~ " seguido da tecla ENTER, ou pressionamos a barra de espaços. Se o caractere estiver grafado em maíusculas se converterá em seu correspondente minúsculo e vice-versa.

Este comando pode se aplicar a mais de um caractere. O comando "*10*~" fará com que os próximos 10 caracteres tenham a caixa trocada. Para melhor visualização, tomemos a seguinte linha como exemplo:

```
Maria foi ao mercado e encontrou João.
```

O comando "*50*~" resulta em

```
mARIA FOI AO MERCADO E ENCONTROU jOÃO.
```

Como podem ver, o que estava maiúsculo ficou minúsculo e vice-versa.

12.30 Opções Magic/Nomagic

Diversos caracteres possuem um significado especial dentro do *vi*, como "[", "-" e "\", apenas para citar alguns. Quando estes caracteres fazem parte da sequência de caracteres sobre as quais se quer atuar, é mais conveniente desligar esta característica 'mágica' que possuem para que se tornem caracteres comuns.

O comando

```
:%s/[a-z]123//g
```

com a opção "magic" ativada irá atuar sobre qualquer palavra que comece com uma letra no intervalo de "a" até "z". Com esta opção desligada o mesmo comando irá atuar sobre a cadeia de caracteres "[a-z]123", exatamente como digitado.

Isto pode ser feito ativando a opção *nomagic*":

```
:set nomagic
```

Para fazer com que tudo volte ao normal, basta reativar a opção *magic*:

```
:set magic
```

12.31 dos2unix

Usuários que frequentemente transferem arquivos entre sistemas DOS ou similares e ambientes Unix, frequentemente precisam remover os caracteres CTRL-M ao final de cada linha.

Dentro do editor *vi*, o seguinte comando remove estes caracteres:

```
:s/^V^M//g
```

O caractere CTRL-V deve ser digitado antes da inserção de qualquer caractere de controle dentro do *vi*, em nosso caso CTRL-M.

12.32 Aprendendo Mais

O editor *vim*, padrão do *Conectiva Linux*, além de extremamente poderoso, é distribuído com uma farta documentação, detalhada e rica em dicas úteis. Para determinar os componentes do vim instalados em seu sistema, utilize o comando

```
$ rpm -qa | grep vim
vim-common-5.3-12cl
vim-enhanced-5.3-12cl
vim-help-5.3-12cl
vim-minimal-5.3-12cl
vim-syntax-5.3-12cl
vim-X11-5.3-12cl
```

A documentação do *vim* está incluída no pacote *vim-common*. Para listar os arquivos de documentação usamos novamente o comando *rpm*:

```
$ rpm -qd vim-common
/usr/doc/vim-common-5.3/README.txt
/usr/doc/vim-common-5.3/README_src.txt
/usr/doc/vim-common-5.3/bugreport.vim
/usr/doc/vim-common-5.3/gvimrc_example
/usr/doc/vim-common-5.3/menu.vim
/usr/doc/vim-common-5.3/mswin.vim
/usr/doc/vim-common-5.3/termcap
/usr/doc/vim-common-5.3/tutor/README.txt
/usr/doc/vim-common-5.3/tutor/tutor
/usr/doc/vim-common-5.3/vimrc_example
/usr/man/man1/ex.1.gz
/usr/man/man1/rvi.1.gz
/usr/man/man1/rview.1.gz
/usr/man/man1/vi.1.gz
/usr/man/man1/view.1.gz
/usr/man/man1/vim.1.gz
/usr/man/man1/xxd.1.gz
```

Destes documentos, vale a pena ler atentamente o arquivo *tutor* que cobre alguns aspectos básicos do uso do editor *vim*. As páginas de manual, localizadas em */usr/man/man1*, podem ser invocadas com o comando *man*.

Além do grande número de sítios sobre vi mantidos pelos admiradores deste magnífico editor, uma boa pedida é visitar o site do *vim,* que fica em http://www.vim.org. Neste endereço, além do software, você encontra uma infinidade de dicas sobre edição de texto com *vim*.

Capítulo 13

Comandos do Sistema

13.1 Aliases

Todo administrador de sistemas deve dar o comando abaixo

```
# ps aux | grep [usuário]
```

ao menos umas vinte vezes por dia. Cuidado, não se esqueça das Lesões por Esforços Repetitivos (LER).

Porque digitar tanto se com um simples alias:

```
alias psg 'ps aux | grep $1'
```

você consegue o mesmo efeito. Uma vez definido o alias basta digitar:

```
# psg root
```

e você vai obter uma listagem de todos os processos rodando sob a identificação do usuário *root* (ou qualquer outro). Como o alias utiliza o comando *grep*, podemos fornecer como parâmetro qualquer valor constante na saída do comando *ls*:

```
$ psg tty
root 412 0.0 1.0  1140  316 ttyS0 S 20:40 0:00 gpm -t ms
root 430 0.0 1.2  1084  392 tty2  S 20:40 0:00 /sbin/mingetty tt
root 431 0.0 1.2  1084  392 tty3  S 20:40 0:00 /sbin/mingetty tt
root 432 0.0 1.2  1084  392 tty4  S 20:40 0:00 /sbin/mingetty tt
root 433 0.0 1.2  1084  392 tty5  S 20:40 0:00 /sbin/mingetty tt
root 434 0.0 1.2  1084  392 tty6  S 20:40 0:00 /sbin/mingetty tt
```

A definição do alias deve ser colocada dentro do arquivo de configuração *.bashrc* (bash), por exemplo.

```
alias psg='ps aux | grep $1'
```

Sofisticando um pouco mais este alias, podemos incluir o parâmetro $* ao invés de $1. Desta forma podemos especificar diretivas para o comando *grep*.

```
alias psg="ps aux | grep $*"
```

Vejamos alguns exemplos:

```
$ psg X11             [1]
queiroz 449 0.0 2.5  2296  768 tty1 S 20:40 0:00 xinit /etc/X11/xi
root    456 3.5 8.6 10792 2652 ?    R 20:40 3:24 /etc/X11/X :0 -au
$ psg -i x11          [2]
queiroz 449 0.0  2.5  2296  768 tty1 S 20:40 0:00 xinit /etc/X11/xi
root    456 3.4 13.2 10792 4068 ?    R 20:40 3:15 /etc/X11/X :0 -au
$ psg -ic x11         [3]
2
```

Em (1), listamos todos os processos que continham os caracteres *X11*. Em (2) especificamos a diretiva *-i*, que instrui o comando *grep* a buscar pelos parâmetros fornecidos, independentemente se grafados com letras maiúsculas ou minúsculas. E finalmente, em 3, estamos interessados apenas no número

de processos (diretiva *-c*) que contenham os caracteres *X11*, e obtemos o resultado igual a 2.

Pense um pouco. Veja os comandos que usa com mais frequência e crie *aliases* para diminuir o seu esforço de digitação e quem sabe ganhar alguns preciosos minutos no seu dia a dia?

Algumas sugestões:

```
alias dir="ls -l"
alias rlogin="ssh"
alias telnet="ssh"
alias ldir='ls -l | grep "^d"'
alias ll='ls -al'
alias ^L=clear
alias m=more
alias r=rlogin
```

Um alias extremamente útil, definido no ambiente do usuário root de sistemas *Conectiva Linux*, é o alias cds:

```
alias cds='cd /etc/rc.d/init.d && ls'
```

Como pode se ver, o diretório corrente é alterado para */etc/rc.d/init.d*, onde se encontram os scripts de inicialização de processos. Em seguida à mudança de diretório é emitido o comando "*ls*", para se visualizar quais serviços estão disponíveis.

13.2 unalias

Recomenda-se que o comando *rm* principalmente de usuários poderosos como *root*, sejam transformados em algo do tipo *rm -i*. Desta forma, a cada arquivo que se desejar remover, o sistema solicitará a confirmação. Este é o padrão da configuração do usuário *root* de sistemas *Conectiva Linux*.

É claro, que para um grande número de arquivos, isto passa a ser mais um aborrecimento do que uma medida de precaução. Nestas situações, para utilizar o comando sem a conversão imposta pelo alias, basta precedê-lo pelo carácter "\".

Veja só:

```
# rm *
rm: remove a? s
rm: remove? n
rm: remove 00index.txt? s
 ...
```

Já, quando o número de arquivos for grande e você tiver certeza do que está fazendo:

```
# \rm *
```

Concluindo, deixe o comando *rm* do usuário *root* como um alias para *rm -i*. Quando precisar fazer deleções em massa, preceda o comando *rm* com "\" para evitar a confirmação apenas naquele momento.

13.3 Alien: Conversor de Formatos

O *Alien* é um programa que realiza conversão entre os formatos *rpm* (RedHat), *dpkg* (Debian), *stampede slp*, e o formato *tgz* (Slackware). Bastante útil para permitir que um software de uma distribuição seja usado em outra. O software *alien* pode ser usado para converter para o seu formato preferido e instalar o pacote em seu sistema.

13.4 apropos

Comandos extremamente úteis, os comandos *man*, *apropos* e *whatis* são indispensáveis para a boa convivência com o Linux. O comando *man* permite acesso aos manuais on-line do sistema. Se não soubermos o nome exato do comando a coisa fica mais complicada. Para isto usamos os comandos *apropos* e *whatis*.

O comando *apropos* consulta um banco de dados consistindo da descrições curtas dos comandos do sistema e utilitários.

É bastante útil em situações em que se deseja executar determinada tarefa e não se conhece o nome do comando. Por exemplo, caso queiramos obter informação a respeito de compiladores instalados no sistema, podemos usar o comando *apropos*, ou seu equivalente *man -k*, da seguinte forma:

```
% apropos compiler
cccp, cpp(1) - The GNU C-Compatible Compiler Preprocessor.
gcc, g++(1)  - GNU project C and C++ Compiler
tic (1m)     - the terminfo entry-description compiler
xsubpp(1)    - compiler to convert Perl XS code into C code
zic(8)       - time zone compiler
```

Examinando a saída do comando apropos descobrimos que o que buscamos, um compilador para a linguagem C, é o comando *gcc*. Podemos então obter informações mais detalhadas deste compilador com o comando *man*:

```
$ man gcc
```

Todavia, este banco de dados não é criado automaticamente. O administrador de sistemas precisa criá-lo através do comando *makewhatis*. Este comando irá varrer todos os diretórios especificados na variável de ambiente *MANPATH* e irá construir um arquivo chamado *whatis*, onde serão colocadas as descrições dos programas.

Para construir este banco de dados emitir, como usuário *root*, o comando:

```
# makewhatis
```

Uma vez criado o banco de dados o comando *apropos* (ou *man -k*) poderá então ser utilizado.

E finalmente, o comando *whatis* nos permite obter uma descrição resumida de um comando, também consultando o banco de dados *whatis*:

```
$ whatis ls
ls (1)               - list directory contents
```

13.5 Arquivos – Tipos

Em sistemas Linux, encontramos diversos tipos de arquivos, geralmente identificados por suas terminações:

- .Z – Arquivos compactados com o programa compress
- .tar – Arquivos criados com o comando '*tar*' (tape archive)
- .gz – Arquivos compactados com o programa gzip
- .tgz – Arquivos criados com o comando '*tar*' e compactados com o programa '*gzip*'
- .txt – Arquivos texto
- .html – Arquivos html
- .ps – Arquivos PostScript
- .au – Arquivos de áudio
- .xpm, .jpg, .gif, .png – Arquivos de imagens
- .rpm – Arquivos de Distribuição de Software (*Red Hat Package Manager)*

- .conf – Arquivos de configuração
- .c – Código fonte de programas C
- .h – Arquivos de Cabeçalho
- .o – Código Objeto
- .pl – Programas Perl
- .tcl – Programas TCL
- .so – Código Compartilhado

Esta é a convenção, que nem sempre é usada. Pode-se perfeitamente criar um arquivo texto que não tenha a terminação .txt. O comando *file* nos permite identificar a que categoria um arquivo pertence. Vejamos alguns exemplos:

```
$ file /etc/passwd n*
/etc/passwd: ASCII text
networks:      empty
nscd.conf:     ASCII text
nsswitch.conf: English text
ntp:           directory
ntp.conf:      English text
```

O comando *file* baseia suas decisões consultando o arquivo */usr/share/magic*, onde estão registradas as características dos principais tipos de arquivos do sistema. Para maiores informações consultar a documentação do arquivo *magic*.

13.6 awk

13.6.1 Impressão de Campos Selecionados

Um comando bastante útil para manipulação de cadeias de caracteres é o *awk*. O nome (bastante estranho por sinal) é derivado das iniciais de seus três criadores, Aho, Kernighan, e Weinberger. Funciona com dutos (*pipes*) ou diretamente com arquivos.

Por exemplo, suponhamos que queremos fazer alguma formatação dos resultados gerados pelo comando *ls*:

```
$ ls -l /tmp
total 184
srwxrwxrwx 1 joao wheel   0 May  5 18:12 FvConSocke
-rw-r--r-- 1 root system 193 Apr 29 14:00 SM_OPO13zqd
-rw-r--r-- 1 root system 220 Apr 25 16:31 XX
-rw-r--r-- 1 root system 949 Apr 25 15:28 a
-rw-rw-rw- 1 root system   0 Apr 25 19:12 errdemon.1708
```

Se não estivermos interessados em todos estes campos, podemos fazer uma seleção com o programa *awk*:

```
$ ls -l /tmp | awk '{print $9}'
FvConSocke
SM_OPO13zqd
XX
```

Se quisermos fazer uma listagem dos usuários de uma máquina em ordem alfabética podemos utilizar o arquivo */etc/passwd*. A diferença é que o arquivo */etc/passwd* possui o caractere "**:** " como delimitador de seus campos. O *awk* considera como delimitador padrão espaços em branco. Para alterar esta especificação utilizar a diretiva **F** seguida do valor do delimitador:

```
$ awk -F: '{print $1}' /etc/passwd | sort > list.txt
```

A saída do comando *awk* é redirecionada para o comando *sort* que faz a ordenação gravando o resultado em *list.txt*.

13.6.2 A Variável NF – Number of Fields

Para se imprimir apenas o último campo de um arquivo com o comando *awk*, podemos utilizar o comando:

```
awk '{print $NF}' arquivo.exemplo
```

A variável *NF* significa número de campos. Quando precedida por $ indica o último campo, à semelhança de $1, $2, e assim por diante.

Para imprimir a contagem do número de campos de um registro:

```
awk '{print NF}' arquivo.exemplo
```

Para imprimir apenas as linhas que contenham mais de dez campos:

```
awk 'NF > 10 {print}' arquivo.exemplo
```

Ou, para imprimir apenas as linhas que possuam exatamente 10 campos:

```
awk 'NF == 10 {print}' arquivo.exemplo
```

Se quisermos imprimir apenas o segundo campo de registros que contenham, em qualquer posição, a palavra *teste*:

```
awk '/teste/ {print $2}' arquivo.exemplo
```

13.7 Manual AWK

O pacote *gawk* (de *Gnu AWK*), mantido e distribuído pela *Free Software Foundation*, traz também um livro excelente sobre o uso do comando *awk*, com muitos exemplos.

Além deste manual, existe também um cartão de referência.

A documentação é distribuída separadamente do código fonte e dos binários do programa *gawk*. O livro e o cartão de referência fazem parte do pacote *gawk-doc*, integrante da distribuição *Conectiva Linux*.

Vale a pena ler. Primeiramente porque o material é excelente e segundo porque não custa nada ...

13.8 bzip2

Espaço em disco, mesmo com os discos rígidos cada vez maiores, certamente é um problema hoje em dia para qualquer administrador de sistemas. Qualquer ferramenta de compactação que nos permita obter um pouco mais de espaço em nossos discos rígidos é extremamente bem-vinda.

Sistemas *Conectiva Linux* dispõem do compactador *bzip2*, que consegue compactar, com uma taxa de eficiência 10% a 15% melhor do que os demais compactadores existentes.

13.9 Backup

13.9.1 Recomendações Gerais

A atualização de sistemas operacionais é um fato comum na vida de todos os administradores de sistemas.

Estas atualizações exigem que seja feito um backup dos arquivos dos usuá-

rios e de vários arquivos de configuração do sistema, como por exemplo, */etc/passwd*, */etc/shadow*, */etc/sendmail.cf*, e assim por diante.

As recomendações que seguem podem parecer simples e sem sentido, mas muitas tragédias já aconteceram pela não observação destes princípios tão importantes.

Desta forma, sempre que for fazer qualquer tipo de manutenção de porte em seu sistema, siga os seguintes passos:

- Faça backup
  Recomendo sempre que backups sejam feitos com o comando *dump*. É o mais seguro e confiável. Se possível (altamente recomendável) faça dois ou mais backups. A sua unidade de fita pode estar gasta ou velha e pode estragar a fita do seu (às vezes único) backup. E aí não tem mais jeito.

  Se possível, faça um backup extra, possivelmente no disco de uma outra máquina, daqueles arquivos extremamente importantes (como por exemplo, o trabalho de vários anos de seus usuários). Os backups em disco aceleram a recuperação e são uma garantia a mais para aqueles dados extremamente valiosos que voce possui.

- Proteja a fita após o backup
  É comum se ver nas listas de discussão de Unix a seguinte pergunta:

  > Eu dei o comando "tar cvf" ao invés de "tar xvf" e como a fita estava desprotegida, foi iniciada uma gravação. Existe alguma maneira de se recuperar o conteúdo desta fita?
  > Assinado: Des E. Sperado

  Este tipo de erro é extremamente comum. Se a fita estivesse protegida o máximo que poderia ocorrer seria uma mensagem de erro.

  Portanto, SEMPRE PROTEJA CONTRA GRAVAÇÃO SUAS FITAS DE BACKUP!

- Sempre obedeça às duas primeiras regras e evite ser assassinado (ou pior) pelos seus usuários.

13.9.2 Estratégias

Nenhuma estratégia de backup atende a todos os sistemas. Uma estratégia que é adequada para sistemas com um usuário pode ser imprópria para sistemas que atendam dez ou mais usuários. Da mesma forma, uma estratégia adequada para um sistema em que os arquivos são modificados freqüentemente não se adequa a um sistema em que tais alterações são raras. Apenas o administrador pode determinar com precisão a estratégia que melhor se adequa a cada situação. Na escolha de uma estratégia de backup tente levar em consideração os seguintes fatores:

- Capacidade de recuperação em caso de crash total do sistema
  Você consegue recuperar o seu sistema se um disco quebrar? Você conseguirá recuperar o seu sistema se TODOS os discos quebrarem? E se tudo pegar fogo, inclusive os backups? Embora isto seja quase impossível, estes fatores devem ser considerados quando da definição da estratégia de backup.

- Verifique os seus backups periodicamente
  O meio de armazenamento pode não ser totalmente confiável. Um conjunto de fitas ou disquetes muito grande é totalmente inútil se os dados neles contidos não puderem ser restaurados. Para certificar-se de que os dados em uma fita podem ser lidos, faça periodicamente a verificação dos mesmos (usando, por exemplo, os comandos *tar tv* ou *restore -t*).

- Estabeleça uma política de retenção de fitas
  Determine um ciclo para reutilização de fitas. Você não deve, entretanto, reutilizar todas as suas fitas. Ás vezes se transcorrem meses antes que você ou mesmo algum usuário sinta a necessidade de restaurar algum arquivo importante que tenha sido apagado por engano. Devido a isto backups antigos, dentro de certos limites, devem ser mantidos. Existem várias formas de se fazer isto, que irão depender em grande parte dos recursos, das peculiaridades e das necessidades

de cada instalação. E o mais importante, esta política deve ser bem conhecida por todos os usuários de seus sistemas.

- Verifique os sistemas de arquivos antes de cada backup
  Um backup efetuado a partir de um sistema de arquivos corrompido pode ser inútil. Antes de efetuar backups é aconselhável verificar a integridade dos sistemas de arquivos usando o comando *fsck*.

- Faça backups em horários em que o sistema se encontre em estado de mínima (ou nenhuma) atividade. Se possível coloque o sistema em modo monousuário (o que pode ser bastante difícil).

- Faça um, preferencialmente mais de um, backup antes de efetuar alterações substanciais no sistema. É sempre aconselhável fazer um backup antes de efetuar mudanças de porte no sistema operacional, instalação de correções, mudanças significativas em programas aplicativos, enfim, tudo o que possa representar uma ameaça ao funcionamento normal do sistema. Em caso de problemas o backup significa a volta a um status original em que tudo estava funcionando a contento.

13.9.3 Como desenvolver uma estratégia de backups

Os dados (programas ou texto), podem ser divididos em duas categorias:

- Dados do sistema
  Compreendem o sistema operacional e suas extensões. Estes dados devem sempre ser mantidos nos sistemas de arquivos do sistema operacional (*/*, */usr*, */tmp*, */var*, etc.).

- Dados dos usuários
  São os dados que os usuários necessitam para desempenhar suas tarefas. Estes dados são normalmente mantidos no sistema de arquivos */home* ou em sistemas de arquivos especificamente criados para esta finalidade.

Se os dados dos usuários são mantidos em sistemas de arquivos separados, a tarefa de gerenciar backups fica mais fácil. Normalmente o backup dos dados dos usuários é feito separadamente dos backups dos dados do sistema por duas razões:

- Os dados dos usuários são alterados com mais freqüência do que os dados do sistema. A imagem do backup dos dados dos usuários é também muito menor do que a dos dados do sistema.
- É mais fácil e rápido restaurar os dados dos usuários quando estes são mantidos separados. A restauração do sistema operacional juntamente com os dados dos usuários requer um tempo e esforço consideráveis. A razão é que o método usado para recuperar o sistema operacional requer um boot do sistema a partir de fita, disquete, ou um outro meio e a instalação do backup do sistema.

13.9.4 Amanda

Uma alternativa bastante barata (grátis) para se fazer backups de várias máquinas rodando Unix e derivados, ligadas em rede, é o programa Amanda (*Advanced Maryland Automated Disk Archiver*). Foi desenvolvido na Universidade de Maryland, onde realiza o backup de 28 GB de dados distribuídos por 321 filesystems em mais de 128 workstations (informação extraída do arquivo README da distribuição do software).

Este programa vem sendo bastante usado, inclusive no Brasil, em várias empresas e instituições de ensino. É bastante eficiente e rápido.

A pesquisa por sistemas de arquivos para recuperação é extremamente otimizada pela gravação de cabeçalhos no começo da fita, que fazem com que o posicionamento no arquivo desejado e a recuperação sejam efetuadas de forma bastante eficiente e rápida.

O pacote pode ser obtido em *ftp://ftp.amanda.org/pub/amanda*. O *Conectiva Linux* inclui o software *amanda* em sua distribuição, o que facilita em muito a sua instalação, por meio do gerenciador RPM.

Existem várias listas de discussão sobre o Amanda. Para assinar qualquer destas listas envie uma mensagem para os endereços listados a seguir e inclua no corpo da mensagem "*subscribe <nome-da-lista> <seu-endereço-eletrônico>*".

- amanda-announce
  Lista de divulgação de novidades sobre o pacote Amanda. Para assinar envie uma mensagem para *amanda-announce-request@cs.umd.edu*
- amanda-users
  Lista para os usuários do Amanda discutirem o software e resolução de dúvidas e probremas. Para se cadastrar envie email para *amanda-users-request@cs.umd.edu*
- amanda-hackers
  Lista para discussão de detalhes técnicos do software Amanda. Para assinar envie email para *amanda-hackers-request@cs.umd.edu.*

Se você gasta cerca de 3 ou mais horas procurando em apenas uma fita pelo sistema de arquivos ou arquivo que quer restaurar então este software pode facilitar em muito a sua vida.

13.9.5 Backups via e-mail

Muitas vezes não é necessário que se faça um backup completo de todos os sistemas de arquivos de uma máquina. Todavia, ainda assim é importante que se salvem alguns arquivos importantes.

Uma solução para este problema é realizar o backup destes arquivos, em sua maioria arquivos de configuração, através de e-mail.

Desta forma, os arquivos são gravados diariamente e enviados via email para uma ou mais máquinas ou pessoas, onde são gravados em um arquivo em disco.

A automatização via *cron*, pode ser criada através da adição à *crontab* de uma linha como:

```
0 0 * * * /usr/local/etc/bck.sh 1>/dev/null 2>/dev/null
```

Todos os dias, as 00:00 horas, o script */usr/local/etc/bck.sh* é executado.

Para melhor entender o procedimento, encontra-se abaixo o script *bck.sh*, contendo explicações detalhadas sobre os passos seguidos:

bck.sh

```
#!/bin/bash
#
# Script para realização de backup de arquivos de
# configuração via correio eletrônico
#
PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/bin
export PATH
# A seguir é criado um arquivo no formato tar,
# contendo todos os arquivos de configuração
# que se deseja salvar. A lista dos arquivos
# deve ser criada em conjunto com todos os
# usuários e administradores da máquina. O
# arquivo tar criado é compactado utilizando-se
# o programa gzip e redirecionado para o arquivo
# /tmp/host.config.tar.gz
cd /
tar cvzf - \
         /var/spool/cron   \
         /var/named        \
         /etc/aliases      \
         /etc/dumpdates    \
         /etc/gated.conf   \
         /etc/cron*        \
         /etc/dumpdates    \
         /etc/lilo.conf    \
```

```
        /etc/passwd       \
        /etc/shadow       \
        /etc/mail         \
        /etc/fstab        \
        /etc/smb.conf > /tmp/host.config.tar.gz
# Neste ponto, o formato do arquivo é transformado
# pelo comando uuencode para permitir a sua trans-
# ferência via mail
uuencode /tmp/host.config.tar.gz \
  host.config.tar.gz > /tmp/host.config.tar.gz.uu
# Em seguida, o arquivo é enviado para a(s)
# máquinas de destino onde deverá ser criado
# um alias que se encarregará de realizar a
# gravação no local apropriado. A mensagem
# pode se destinar usuário root que, por
# meio de filtros do programa procmail
# redireciona os backups para um arquivo.
mail -s "Configuracao `date +%d/%m` (host)" \
  backup@dominio.com.br < \
  /tmp/host.config.tar.gz.uu
# Finalmente, os arquivos temporários gerados devem
# ser removidos
rm /tmp/host.config.*
```

13.9.6 Dump em um arquivo remoto

Pode ser que você possua espaço, mas o espaço disponível está em uma máquina remota.

Neste caso certifique-se que a máquina que irá receber o arquivo de backup confia na primeira (arquivo *.rhosts)*.

Em seguida,emita o comando abaixo, fazendo as modificações necessárias:

```
# dump 0fb - 126 [filesystem] | rsh [máquina remota] \
```

```
'(cd [diretório destino];dd of=[arquivo destino]  os=126b)'
```

13.10 basename/dirname

Dois comandos que permitem a manipulação de nomes e caminhos de arquivos dentro de sistemas Unix são os comandos *basename* e *dirname*.

O comando *basename* irá retirar o último nome após o último "/ ":

```
$ basename /usr/local/bin/gzip
gzip
```

Ou seja, este comando pode ser utilizado para extrair apenas o nome de um arquivo a partir do caminho completo, neste caso, o nome *gzip*.

Já o comando *dirname* retorna como resultado o caminho inteiro fornecido à exceção do último nome, como exemplificado abaixo:

```
$ dirname /usr/local/bin/gzip
/usr/local/bin
```

13.11 bash – Histórico de Comandos

Em sistemas *Conectiva Linux*, a shell padrão, *bash*, armazena no arquivo *.bash_history*, os comandos executados. O número de comandos armazenado é determinado pelo valor da variável de ambiente *HISTSIZE*:

```
$ env | grep HISTSIZE
HISTSIZE=1000
```

Como podemos ver, o valor da variável *HISTSIZE* é 1000, ou seja, são armazenados os últimos 1000 comandos emitidos, muito mais do que um usuário comum precisa se lembrar.

```
$ wc .bash_history
1000    1894   10841 .bash_history
```

Para recuperar um comando que tenha sido emitido recentemente, podemos usar o comando *grep*:

```
$ grep slocate .bash_history
slocate -U ~
slocate -U ~ -d slocate.db
slocate -U ~ -o slocate.db
slocate --database=slocate.db *.tex
slocate --database=slocate.db tex
```

Ou ainda

```
$ history | grep slocate
```

Basta então identificar qual comando desejamos repetir e executá-lo.

Mais simples ainda é usar o caractere "**!** " seguido de caracteres que permitam identificar o comando emitido sem conflitos com outros comandos:

```
$ !s
slocate
```

O comando que coincidir com o padrão especificado é ecoado para a tela e executado em seguida.

13.12 Boot – Mensagens

Durante o boot do sistema aparecem diversas mensagens que mesmo que tentemos ler não conseguimos. Passam pela tela muito rápido. Embora não

pareça, muitas mensagens importantes aparecem na tela. Quantidade de memória disponível, placas de som e de rede detectadas, CDROM, monitor de vídeo, discos rígidos, e muito mais.

Para visualizar estas mensagens com o sistema no ar, basta usar o comando *dmesg*:

```
$ dmesg | more
```

Ou melhor ainda, para analisar com mais calma, redirecione a saída do comando dmesg para um arquivo:

```
$ dmesg > mensagens.boot
```

Normalmente não precisamos nos preocupar com estas mensagens. Quando a máquina começa a apresentar problemas, pode ser interessante analisarmos estas mensagens para ver se algum problema foi detectado durante o boot do sistema operacional.

13.13 tar

13.13.1 Cópia de Arquivos com tar

O comando *tar* pode ser usado para copiar arquivos mantendo-se as permissões originais e os donos, tanto localmente quanto através da rede.

Para realizar esta cópia localmente:

```
% tar cvf - [origem] | (cd [destino]; tar xvf - )
```

Para realizar a cópia remotamente (não se esqueça de incluir a definição das máquinas no arquivo *.rhosts* ou */etc/hosts.equiv*

```
% tar cvf - [origem] | rsh [sistema remoto] \
     '(cd [destino]; tar xvfB - )'
```

13.13.2 Identificação de backups

É claro que todas as fitas devem ser rotuladas com o conteúdo e data de realização dos backups. O comando *dump* retém esta informação e é a minha recomendação maior para se fazer backups. Sempre que possível use o comando *dump* ou equivalente.

Em situações onde se deseja copiar alguns poucos arquivos, o comando *tar* talvez seja mais conveniente. Para identificar de maneira clara a data em que a fita foi gerada, você pode criar um arquivo cujo nome forneça esta informação. Este arquivo pode ser vazio e seu nome indica o nome da máquina, a data e a hora em que foi criado.

Por exemplo:

```
% BACKUPPATH=/u/user/backup/
% DATEFILE=${BACKUPPATH}/B.‘hostname‘.‘date +"%m%d%y.%H:%M"‘
% touch $DATEFILE
```

A letra B indica uma convenção tal como B para bancos de dados, S para software, etc. Você decide.

Uma vez criado o arquivo, execute então o backup:

```
$ tar cvf /dev/rstX $DATEFILE <demais arquivos>
```

Dias, semanas ou meses depois, quando a identificação já sumiu, basta dar o comando:

```
$ tar tvf /dev/rstX | head
```

O comando *head* vai filtrar as 10 primeiras linhas geradas pelo comando *tar*. Como o arquivo que criamos está em letras maiúsculas, ele possivelmente estará entre os dez primeiros:

```
B.<hostname>.071696.11:15
```

Pronto, você achou o que precisava.

É claro que a criação deste arquivo pode ser automatizada através de um shell script, o que é muito mais fácil e menos suscetível a erros.

Mas o melhor mesmo é gastar um pouco de tempo estudando o *Amanda* e partir para uma solução definitiva.

13.14 Backup em arquivos

Uma maneira rápida de fazer backups, é realizá-los diretamente no disco. A maioria das fitas DAT são extremamente lentas (taxa de transferência entre 256kbps e 500kpbs) e o processo de backup e restore, dependendo do tamanho do sistema de arquivos, pode consumir muitas horas ou até mesmo dias.

No caso de instalação de sistemas, e caso você tenha espaço em disco disponível, uma alternativa interessante é realizar o *dump* diretamente em um arquivo.

Por exemplo:

```
# dump -0uf /work/root.bck /
```

O exemplo acima irá gravar o backup do sistema de arquivos raiz no diretório */work/root.bck*.

Uma outra alternativa, para quem não tem muito espaço em disco, é gravar o backup já compactado:

```
# dump -0uf - | gzip > /work/root.bck.gz
```

E para ler de volta:

```
# restore -rvf /work/root.bck
```

ou, caso se use a compactação:

```
# gzip -dc /work/root.bck.gz | restore -rvf -
```

Nunca se esqueça, não importa onde você gravar o seu backup (disco, fita, disquete, etc), sempre teste o que você fez. Se possível (e sempre é possível), faça mais de um backup. Para este livro que você está lendo, no processo de desenvolvimento eu fazia oito backups, seis em disquete e dois em disco, e mais um para o editor. Cuidado nunca é demais.

Para testar:

```
# restore -tvf /work/root.bck
```

ou

```
# gzip -dc /work/root.bck.gz | restore -tvf -
```

13.15 cd – Change Directory

13.15.1 Retornar ao diretório anterior

Se estivermos em */usr/local/bin/* e formos em seguida para */home/usuario/documentos/oficiais* e quisermos voltar para o diretório anterior não precisamos digitar todo o caminho até lá.

Basta digitar

```
$ cd -
```

13.16 cp – copy

13.16.1 Cópia rápida de arquivos

Uma alternativa ao comando *cp* é a cópia utilizando o comando *tar*:

```
$ ( cd ../origem/; tar cf - . ) | tar xvf -
```

Dizem as más línguas que assim funciona até mais rápido do que com o comando *cp*. Faça o seu teste.

13.16.2 Redirecionamento da saída padrão em cshell

A *cshell* não oferece uma maneira simples de se redirecionar a saída padrão *(standard output)* para um arquivo e as mensagens de erro para um outro *(standard error)*.

Para fazer isto proceder como abaixo:

```
$ (ls -l ab bc > /tmp/saida-padrao) >& /tmp/saida-de-erro
```

No nosso exemplo o arquivo *ab* não existe e irá gerar uma mensagem de erro que será gravada no arquivo */tmp/saida-de-erro* e as características do arquivo bc serão gravadas no arquivo */tmp/saida-padrao*.

```
$ more /tmp/saida-padrao
-rw-rw-r--   1 queiroz  queiroz         0 Nov 28 22:48 ab
$ more /tmp/saida-de-erro
ls: bc: Arquivo ou diretório não encontrado
```

13.17 cron

Existem determinadas tarefas que precisam ser realizadas periodicamente no sistema. A automatização destas tarefas pode ser feita através do programa

cron. Este processo periodicamente lê uma tabela chamada *crontab* para identificar quais tarefas devem ser executadas.

O arquivo *crontab* consiste de várias linhas, e cada uma delas contém seis campos:

1	2	3	4	5	6
0	0	*	*	*	calendar

Estes campos significam:

Campo	Valor	Intervalo
1	Minutos	0 a 59
2	Hora	0 a 23
3	Dia do mês	1 a 31
4	Mês	1 a 12
5	Dia da Semana	0 a 6 (0 = Domingo)
6	Comando a ser executado	

No exemplo acima, o comando calendar será executado em todos os dias do ano, exatamente às 0 horas e 0 minutos (meia noite). O * indica todos os valores possíveis do campo em questão.

Outro exemplo

```
0,10,20,30,40,50  *  *   *  apagalixo
```

Ou ainda, com exatamente o mesmo significado:

```
*/10      *  *   *  apagalixo
```

O comando *apagalixo* será executado todos os dias do ano, a cada dez minutos.

Ainda outro exemplo:

```
0-10   *   *  *  *   apagalixo
```

No caso acima o sinal "**-** " indica um intervalo de valores, ou seja, o comando *apagalixo* será executado a cada minuto nos dez primeiros minutos de todas as horas, de todos os dias do ano.

13.18 cat – Concatenate

13.18.1 Numeração de linhas

Freqüentemente nos deparamos com a necessidade de numerar as linhas de um arquivo. O comando *cat*, invocado com o sinal "**-n** " realiza esta tarefa rapidamente. O comando:

```
cat -n teste
```

irá exibir na tela todas as linhas do arquivo *teste* precedidas por uma numeração. Para salvar o arquivo com a numeração, basta redirecionar a saída do comando para um outro arquivo:

```
cat -n teste > teste01
```

O arquivo criado, *teste01*, é idêntico ao arquivo *teste* com as linhas numeradas.

13.18.2 Eliminando linhas em branco

O comando *cat*, pode também ser utilizado para eliminar linhas em branco consecutivas dentro de um arquivo. Onde existirem duas ou mais linhas em branco consecutivas, apenas uma delas será deixada.

Tomemos o arquivo chamado teste, cujo conteúdo é:

```
1


2


5
```

Existem duas linhas em branco separando cada linha com conteúdo. O comando

```
cat -s teste
```

irá gerar a seguinte saída

```
1

2

5
```

onde foi deixada apenas uma linha em branco entre cada linha contendo algum caracter. As demais linhas foram eliminadas.

13.19 csplit – Content Split

Outro comando também utilizado para se dividir um arquivo em vários outros é o comando *csplit* (*Content Split*).

Ao contrário do comando *split*, o comando *csplit* permite que se especifique uma cadeia de caracteres que irá indicar o delimitador de cada um dos novos arquivos.

Tomemos como exemplo o arquivo abaixo, chamado *arq1*:

```
Capítulo 1

Era uma vez, era uma vez, três porquinhos,
Palhaço, Palito e Pedrito.

Capítulo 2
E o Lobo Mau, ...

Capítulo 3
E o caçador, matou o Lobo Mau, casou-se com
Chapeuzinho Vermelho, e viveram felizes para sempre.

The End
```

O autor, colocou todos os capítulos do livro em apenas um arquivo e depois se arrependeu. Agora ele quer criar vários arquivos contendo um capítulo cada. O comando abaixo pode resolver este problema:

```
$ csplit -f Capit arq1 "/Capítulo/" {2}
$ ls -l
total 4
-rw-r--r--   1 queiroz  wheel   0 Jun 17 18:31 Capit00
-rw-r--r--   1 queiroz  wheel  85 Jun 17 18:31 Capit01
-rw-r--r--   1 queiroz  wheel  29 Jun 17 18:31 Capit02
-rw-r--r--   1 queiroz  wheel 136 Jun 17 18:31 Capit03
-rw-r--r--   1 queiroz  wheel 250 Jun 17 18:11 arq1
```

Traduzindo, o comando *csplit* irá criar vários arquivos iniciados em *Capit*, até um máximo de 3 arquivos (parâmetro {2}, computa-se o número entre colchetes acrescido de um). Este valor indica o número de vezes que o comando será repetido. No nosso exemplo, foi especificado exatamente o

número de capítulos contidos no arquivo original (3). Caso não conheçamos este valor, podemos especificar um número que sabemos maior que o número de arquivos existentes. O comando *csplit* irá reclamar, e apagar todos os arquivos já criados. Para evitarmos que isto aconteça, basta especificar a diretiva *k*, ou seja, a reclamação continuará sendo feita, mas o trabalho já feito não será removido. O que não devemos fazer é especificar um número inferior ao desejado. Neste caso, o comando ficaria como:

```
$ csplit -k -f Capit arq1 "/Capítulo/" {9}
0
85
29
csplit: {9} - out of range
136
```

A quebra será feita, tomando-se por base o nosso exemplo, antes da cadeia de caracteres *Capítulo*, exclusive. Devido a isto, o primeiro arquivo, *Capit00*, está vazio. Os arquivos criados, à exceção do arquivo *Capit00* que está vazio, contêm:

- Capit01

  ```
  Capitulo 1

  Era uma vez, era uma vez três porquinhos,
  Palhaço, Palito e Pedrito.
    ...
  ```

- Capit02

  ```
  Capitulo 2
  E o Lobo Mau, ...
  ```

- Capit03

```
Capitulo 3
E o caçador, matou o Lobo Mau, casou-se com
a Chapeuzinho Vermelho, e viveram felizes para
sempre.

The End
```

13.20 date

O comando *date* pode ser utilizado de diversas formas. Uma utilização bastante comum é a criação de arquivos com parte do nome identificando a data de criação. O comando

```
$ ARQUIVO=‘date +%y%m%d‘
$ echo $ARQUIVO
000208
```

Os dois primeiros dígitos (00) identificam os dois últimos caracteres do ano corrente (2000), o mês de fevereiro (02) e o dia corrente (08). Como os problemas do ano 2000 nos ensinaram, não é muito inteligente usar como representação do ano apenas dois caracteres. É mais sensato definir o ano com quatro dígitos, para evitar problemas futuros:

```
$ ARQUIVO=‘date +%Y%m%d‘
$ echo $ARQUIVO
20000208
```

Ainda uma outra maneira de se representar a data é utilizando o formato Juliano, em que os dias são identificados de forma sequencial no transcorrer do ano:

```
$ ARQUIVO=‘date +%Y%j‘
```

```
$ echo $ARQUIVO
2000039
```

O dia 8 de fevereiro corresponde ao trigésimo nono dia do ano, contado seqüencialmente.

13.21 daytime

Na porta 13 de sistemas Linux funciona o serviço *daytime*. Podemos verificar a hora de um determinado sistema estabelecendo uma conexão via telnet na porta 13 desta máquina:

```
$ telnet host.domain.br 13
Trying...
Connected to host.domain.br.
Escape character is '^]'.
Tue Jan 28 16:09:45 2000
Connection closed.
```

Uma *shell script* simples, que estabeleça uma conexão na porta 13 pode ser bastante útil para administradores de sistemas que necessitem verificar o horário de diversas máquinas remotas:

hora.sh

```
#!/bin/bash

for h in apolo medusa obelix panoramix cesar
do
echo $h
telnet $h.host.domain.br 13
done
```

13.22 Dutos

Um dos fatores que marcam a superioridade do Unix é o conceito de dutos. No Unix a filosofia original previa o desenvolvimento de módulos que desempenhassem uma tarefa e a desempenhassem bem.

Vejam só este exemplo:

```
cat livro*|deroff -w|dd conv=lcase|sort|uniq -c|sort -nr|more
```

Esta sequência de comandos irá realizar uma tarefa que iria requerer muitas linhas de código e muito planejamento. Vamos analisá-la:

- `cat livro*`
  redireciona para a saída padrão o conteúdo de arquivos iniciados em livro
- `deroff -w`
  transforma o resultado do comando *cat* em uma saída contendo uma palavra por linha
- `dd conv=lcase`
  converte para letras minúsculas o resultado do comando *deroff*
- `sort`
  ordena alfabeticamente o resultado
- `uniq -c`
  agrupa palavras iguais registrando o número de ocorrências
- `sort -nr`
  realiza uma nova ordenação, porém numérica, colocando no topo da lista as palavras que ocorrem com mais freqüência
- `more`
  exibe uma tela de cada vez

Este exemplo ilustra de maneira clara como diversos comandos reunidos podem realizar tarefas complexas e em um tempo ínfimo, comparado com o esforço necessário para escrever um programa, em outros ambientes, que desempenhe tarefa similar. Cada um dos comandos acima realiza um processamento sobre os dados que lhe são fornecidos, preparando-os para serem processados pelo próximo comando na seqüência.

Agora, para que serve isto? Por exemplo, um professor de literatura analisando a obra de Shakespeare, poderia querer determinar um padrão no uso de palavras. Outro exemplo, um professor de inglês instrumental poderia querer determinar as palavras mais utilizadas para elaborar as suas aulas de maneira mais eficiente. Poderia até mesmo determinar padrões de uso diferentes para cada área de especialização, como medicina, engenharia, administração, e assim por diante. Ainda outro exemplo, um administrador de sistemas pode desejar analisar arquivos de log gerados por programas como *tcpwrapper*, *syslog*, e outros para identificar padrões de invasão de seus sistemas. As possibilidades são infinitas e certamente muito interessantes

13.23 echo

Às vezes nos encontramos em situações, em sistemas danificados, por exemplo, em que nem o comando *ls* se encontra disponível. E sem ele fica quase que impossível fazer qualquer coisa.

Se a sua shell entende expressões do tipo "***** ", então o comando *echo* pode ser um substituto razoável. Desta forma, o comando

```
# echo *
```

pode ser usado para exibir a lista de arquivos de forma similar ao comando *ls*.

13.24 Encadeamento de comandos

No *Conectiva Linux* é possível encadear vários comandos, para serem executados um em seguida ao outro. Basta separá-los pelo caractere "**;** " como abaixo:

```
$ ls;mv a b;mail queiroz@unicamp.br < \
   relatorio.txt;df | mail queiroz
```

Isto pode ser particularmente útil em situações onde estamos executando comandos demorados, como por exemplo, compilações e instalação de softwares.

Um exemplo interessante é a compilação de softwares da GNU. A maioria deles pode ser compilada seguindo basicamente três passos:

```
# configure
# make
# make install
```

Se você sabe que tudo se dá nesta ordem, basta então digitar:

```
# configure;make;make install
```

E você pode ir para casa sossegado, porque se tudo der certo, no dia seguinte o pacote já estará compilado e instalado.

É claro que existem outras maneiras de se fazer isto (aliás como em tudo na vida :-)

13.25 Endereçamento IP – Classes Reservadas

É fato conhecido de todos que os endereços disponíveis na Internet são cada vez mais escassos, requerendo dos administradores uma criatividade cada

vez maior para acomodar o número crescente de computadores com cada vez menos endereços.

Uma forma de resolver este problema é utilizar, nas intranets, classes de endereços IP reservadas. Estas classes de endereços não são encontradas na Internet global. Qualquer endereço pertencente a estas redes que forem encontrados em pacotes IP circulando na Internet são descartados pela maioria dos roteadores.

A vantagem de se utilizar estes endereços é que o administrador precisa apenas de um endereço 'real', para conectar o seu gateway principal à Internet. Este gateway atua como um tradutor de endereços entre as duas redes, a intranet de sua empresa e a Internet. Esta tradução é necessária visto que sua rede interna utiliza endereços reservados. O protocolo que realiza esta tradução denomina-se NAT, ou *Network Address Translator*.

As classes IP reservadas são as seguintes:

Classe	Endereço Inicial	Endereço Final
A	10.0.0.0	10.255.255.255
B	172.16.0.0	172.31.255.255
C	192.168.0.0	192.168.255.255

Endereços na Internet são divididos em três classes principais: A, B e C. Redes classe A podem endereçar até 16 milhões de computadores, redes classe B endereçam 65.534 computadores e redes classe C endereçam até 254 computadores. Os endereços acima disponibilizam uma classe A, 21 classes B e 255 classes C. Se você é espaçoso ou tem mania de grandeza, monte sua rede interna com os endereços da classe A. Pode ter certeza que nunca vão faltar endereços ...

13.26 find

13.26.1 Localizando Arquivos sem Dono

O comando *find* é extremamente poderoso e flexível e pode ser usado para descobrir arquivos que atendam a determinadas especificações.

Por exemplo, suponhamos que queiramos descobrir todos os arquivos que não possuem dono em nosso sistema. Esta situação é extremamente comum, visto que usuários são criados e apagados diariamente e seus arquivos muitas vezes permanecem no sistema, podendo mesmo vir a comprometer a segurança. O comando

```
find / -nouser
```

irá gerar uma listagem com todos os arquivos do sistema que não pertencem a ninguém.

Caso queiramos simplesmente apagar estes arquivos (não recomendável!!!) basta redirecionar a saída deste comando para o comando *xargs*, da seguinte forma:

```
find / -nouser -print | xargs rm
```

O mais recomendável é gerar uma cópia de segurança destes arquivos, para em seguida apagá-los:

```
$ find / -nouser | tar cvzf backup.tar.gz -T -
```

O comando *find* irá gerar uma lista de todos os arquivos que não possuam um usuário ativo a eles associado. Esta listagem é então passada através do duto ao comando *tar*. O sinal "**-T** " indica o arquivo contendo a lista de arquivos a serem gravados em *backup.tar.gz*. Neste caso não se trata de um arquivo e sim dos dados gerados pelo comando *find* e indicados por "**-** ".

Para restaurar estes arquivos

```
tar xvzf backup.tar.gz
```

13.26.2 Procurando arquivos no sistema

Com frequência precisamos descobrir arquivos em nosso sistema. Para isto podemos utilizar o comando *find*. Dependendo do tamanho dos sistemas de arquivos, o comando pode demorar desde alguns segundos até vários minutos.

Para reduzir este tempo de espera, uma solução interessante é criar uma listagem previamente e utilizar o comando *grep* para realizar a pesquisa. A listagem seria gerada através de uma entrada na *crontab* e a pesquisa através de uma shell script chamada, por exemplo, de *ff*.

A geração da listagem seria gerada periodicamente através do comando *find* por meio de uma entrada na *crontab* do sistema:

```
0 0 * * * find / -print > /usr/local/filelist
```

Desta forma, à meia noite de todos os dias, será gerada a listagem. Dependendo das características de seu sistema, esta listagem poderá ser gerada duas vezes por dia, uma vez a cada hora, conforme a necessidade.

E a shell script, *ff.sh*:

ff.sh

```
#!/bin/bash
if [ $# -eq 0 ]; then
        echo 1>&2 Sintaxe: $0 parâmetro_de_pesquisa
        exit 1
fi
grep $1 /usr/local/filelist
```

Mas existem formas melhores de se fazer isto, como não podia deixar de ser. Existe também o comando *locate*, que será abordado em breve.

13.26.3 Cópia de uma Árvore de Diretórios

Para copiar uma árvore inteira de diretórios, os programas *find* e *cpio* oferecem a combinação ideal.

Para copiar o diretório */pub/simtel20* para */pub1/simtel20* executar a seguinte seqüência de comandos:

```
$ mkdir /pub1/simtel20
$ cd /pub/simtel20
$ find . -print | cpio -pdl /pub1/simtel20
```

Esta cópia pode também ser realizada por intermédio do comando *cp*:

```
$ cp -R /pub/simtel20 /pub1/simtel20
```

As duas opções acima funcionam. Escolha a sua...

13.27 FTP através do Browser Netscape

O programa *netscape* também pode ser utilizado para estabelecimento de sessões ftp. Para isto, basta selecionar uma URL do tipo:

```
ftp://ftp.dominio.com.br
```

Neste caso será estabelecida uma conexão anônima com o servidor *ftp.dominio.com.br*.

Entretanto é também possível realizar uma sessão ftp identificada. Para isto, basta fornecer, juntamente com o nome do computador, a identificação do usuário:

```
ftp://queiroz@ftp.unicamp.br
```

Neste caso, será solicitado ao usuário o fornecimento de sua senha para o estabelecimento da sessão. Uma outra possibilidade é fornecer a senha juntamente com a identificação do usuário, de forma a não ter que digitar a senha separadamente:

```
ftp://usuario:casinha@ftp.dominio.com.br
```

A URL acima instrui o programa *Netscape* a estabelecer uma conexão de ftp com a máquina *ftp.dominio.com.br*, na conta do usuário *usuario* cuja senha é *casinha*.

13.28 ftp

13.28.1 Automatização de Sessões FTP

Os piores horários para fazer transferência de arquivos são justamente aqueles em que estamos ou trabalhando ou acordados. Todavia é possível a criação de procedimentos que permitam a transferência dos arquivos em qualquer horário que nos convier.

Para isto, precisamos primeiramente criar um arquivo contendo a seqüencia de comandos que desejamos executar no servidor ftp.

Criamos então um arquivo chamado, por exemplo, *ftp.cmds* com o seguinte conteúdo:

ftp.cmds

```
user anonymous jose@host.domain (1)
bin                             (2)
cd /pub/simtelnet/win95/mail    (3)
```

```
get eudor154.zip                (4)
quit                            (5)
```

No exemplo acima, os números servem apenas como referência para a explicação que se segue e não devem ser incluídos nos arquivos de trabalho.

Na linha (1) é feita a identificação. Neste caso está sendo acessado um servidor de ftp anônimo (usuário *anonymous*) e como de praxe, se fornece o endereço eletrônico como senha. Poderia também ser especificado aqui um usuário real e sua senha (cuidado com a segurança!!!). Na linha (2) se define o modo binário para transferência de arquivos. Em (3) é feito o deslocamento para o diretório onde se encontra o programa desejado. Em (4) se solicita a transferência do arquivo eudor154.zip. Finalmente, em (5), se encerra a conexão com o comando *quit*.

Este exemplo é bastante simples. Poderiam ser incluídos vários arquivos, residentes em vários diretórios, etc.

Uma vez criado o arquivo com os comandos, criar um outro arquivo, chamado, por exemplo, de *ftp.sh*.

Este arquivo deverá ter o bit de execução ligado (700, 755, 750, ...) com o seguinte conteúdo:

ftp.sh

```
$ ftp -ni ftp.host.domain < ftp.cmds
```

Desta forma será efetuado uma conexão ftp para a máquina *ftp.host.domain* onde serão executados os comandos contidos dentro do arquivo *ftp.cmds*. Os sinais fornecidos indicam que o comando ftp deve ser executado no modo não interativo ("**-i** "), e para não tentar o autologin durante a conexão inicial ("**-n** ").

Isto feito, programar então para a hora desejada, a execução do comando. Para isto podemos utilizar o comando *at*:

```
at -f ftp.sh midnight
```

O comando será executado à meia noite do dia corrente, um horário bastante conveniente para transferência de arquivos via FTP.

13.28.2 Transferência de Múltiplos Arquivos

Quando se deseja baixar vários arquivos em uma conexão ftp através do comando *mget* o padrão é que seja solicitada autorização para cada arquivo solicitado:

```
ftp> mget *
mget 00-index.txt? y
150 Opening BINARY mode data connection for 00-index.txt (10198 bytes).
226 Transfer complete.
local: 00-index.txt remote: 00-index.txt
10198 bytes received in 0.053 seconds (1.9e+02 Kbytes/s)
mget 3dchem.zip? y
```

Para desligar esta opção basta emitir o comando *prompt*:

```
ftp> prompt
Interactive mode off.
ftp> mget *
```

Desta maneira, todos os arquivos solicitados serão transferidos automaticamente.

13.28.3 Manipulação direta de arquivos

É possível exibir-se diretamente na tela o conteúdo de arquivos recuperados via ftp. Por exemplo, se desejarmos analisar o conteúdo de um arquivo chamado *index.txt*, diretamente na tela:

```
ftp> get index.txt "| more"
```

Esta mesma funcionalidade pode ser estendida a outras situações, como por exemplo:

```
ftp> get sol.ps.gz "|gzip -dc | gs -"
```

Neste exemplo, o arquivo *sol.ps.gz*, será trazido via ftp, descompactado, e o programa *ghostscript* realizará a formatação da informação transmitida através do duto.

Ainda um outro exemplo:

```
ftp> get rcs.tar.gz "| tar xvzf -"
```

O arquivo *rcs.tar.gz* será transferido e automaticamente descompactado e expandido.

Não se esqueça nunca das aspas duplas delimitando os comandos. As aspas são necessárias visto que os comandos contêm espaços em branco.

13.28.4 Transferência de Diretórios

Para transferir, via ftp, todo o conteúdo do seu diretório home para uma outra máquina proceder como abaixo:

```
ftp>put "|tar cvf - ." homedir.tar
```

Será gerado um arquivo chamado *homedir.tar* e este arquivo será então enviado para a máquina destino.

Para melhorar ainda mais, o arquivo pode ser enviado compactado:

```
ftp> put "|tar cvf - . |compress -c" homedir.tar.Z
```

E o caminho contrário, pegar um arquivo via ftp, descompactá-lo e expandi-lo imediatamente:

```
ftp> get example.tar.Z "| zcat | tar xvf -"
```

13.29 Gerenciamento de Espaço em Disco

A tarefa de manter espaço em disco em quantidade suficiente para que os usuários consigam trabalhar e o sistema operacional consiga funcionar decentemente é uma das mais importantes do administrador de sistemas.

O comando *find* é uma ferramenta excelente para desempenhar este tipo de tarefa.

Existem arquivos que aparecem no sistema de tempos em tempos e que ocupam um espaço considerável e não servem para nada.

Arquivos chamados *core* por exemplo são resultados de um dump de memória realizado por um programa para que o programador determine o que ocorreu de errado. Normalmente a maioria das pessoas não se dão ao trabalho de remover estes arquivos e eles continuam a existir no seu disco por um longo tempo, apenas ocupando espaço.

Arquivos postscript também ocupam muito espaço. Não recomendo que se remova estes arquivos, mas pelo menos eles podem ser compactados.

Arquivos há mais de um dia sem serem modificados e residentes no diretório */tmp*, também não devem ser mantidos no sistema.

Alguns administradores também têm por hábito remover os arquivos chamados *a.out* (embora isto possa gerar atritos com alguns usuários).

Também pode ser interessante descobrir quem está abrigando arquivos realmente grandes em seus diretórios para se tomar providências.

Bem, a descoberta de todos estes arquivos e as ações apropriadas podem ser tomadas automaticamente através de uma simples shell script:

```
#! /bin/bash
# Encontra e apaga arquivos chamados core
```

```
find / -name core -print | xargs rm
# Encontra e compacta arquivos postscript
find / -name \*.ps -print | xargs gzip
# Encontra e remove arquivos a.out
find / -name a.out -print | xargs rm
# Descobre arquivos maiores que 2048 blo-
# cos de 512 bytes e envia a relação para
# o administrador de sistemas

find / -size +2048 -print | \
  xargs ls -l | mail root

# Apaga arquivos residentes no  diretório
# /tmp que não tenham sido modificados há
# mais de um dia

find /tmp -mtime +1 -print | xargs rm
```

13.30 Grafia do nome de computadores

No DNS é indiferente que o nome de um computador seja escrito em letras maiúsculas, minúsculas, ou uma combinação de ambas. Desta forma, se você escrever *www.dominio.com.br*, *WWW.dominio.com.BR* ou até *WWW.DOMINIO.COM.BR*, dá tudo no mesmo.

Da mesma forma que o nome do computador pode ser escrito com considerável grau de liberdade, também o endereço do usuário pode ser representado seguindo as mesmas regras. Por exemplo, se você quiser enviar email para alguém, pode escrever "Alguem@dominio.BR", que provavelmente chegará ao seu destino.

Esta regra, de independência de grafia do username, não se aplica a todos os MTA (*Mail Transport Agents*). As versões mais recentes do Sendmail aceitam esta independência mas podem existir programas em que podem

ocorrer problemas. Na dúvida faça um teste antes.

13.31 grep

13.31.1 Alguns recursos úteis do comando grep

- Pesquisa de múltiplos parâmetros:

```
$ grep "arg1|arg2|arg3" file1 file2 file3 file4
```

- Pesquisa de uma cadeia de caracteres, independentemente se maiúsculas ou minúsculas:

```
$ grep -i arg file1 file2 file3
```

No primeiro exemplo, os arquivos *file1*, *file2* ... até *file4* serão pesquisados para se determinar a ocorrência de qualquer um dos parâmetros fornecidos entre aspas (*arg1*, *arg2* ou *arg3*).

No segundo exemplo, o sinalizador *-i* indica que se quer achar o parâmetro *arg1* nos arquivos fornecidos, independentemente se maiúsculos ou minúsculos. Desta forma, *Arg1*, *aRg1*, *ARG1* e qualquer combinação possível de letras maiúsculas ou minúsculas, satisfariam a pesquisa.

13.32 gzip

O programa *gzip* é hoje extremamente popular na Internet. A maioria dos servidores de ftp anônimo compactam os seus arquivos com este programa. Diferentemente dos sistemas Unix comerciais, a grande parte dos sistemas abertos como Linux e FreeBSD utilizam este compactador como padrão.

Programas compactados com o programa gzip apresentam a extensão .gz. como em *sendmail.tar.gz*.

Para descompactar arquivos como este, visando a instalação do produto, no nosso caso, sendmail, é comum seguir os seguintes passos

```
$ gunzip sendmail.tar.gz
$ tar xvf sendmail.tar
```

Na verdade não é necessária a realização do primeiro passo. O arquivo pode ser descompactado para a saída padrão e redirecionado para o programa tar. Desta forma ocupa-se menos disco.

O comando

```
gzip -dc sendmail.tar.gz | tar xvf -
```

obtem exatamente os mesmos resultados dos comandos anteriores sem descompactar o arquivo de distribuição original.

Na verdade nem o programa gzip é necessário. Sistemas *Conectiva Linux* utilizam a versão GNU do programa *tar*, que possui a funcionalidade adicional de poder descompactar (e também compactar) arquivos gerados com o programa *gzip*. Podemos então, com um único comando, realizar a extração completa do arquivo

```
tar xvzf sendmail.tar.gz
```

13.33 Recuperação da Senha de Root

Você se esqueceu da senha de root? Não desespere. Em sistemas Linux basta realizar o boot do sistema em modo monousuário.

Para isto, ao aparecer o prompt do LILO:

```
LILO boot:
```

digitar

```
linux -s
```

Isto fará com que o sistema seja carregado em modo monousuário. Será exibido, após o final do boot, o prompt da shell *bash*:

```
bash#
```

Editar então o arquivo */etc/passwd* e apagar a senha do root. A conta root terá então o acesso liberado.

Por exemplo:

```
root:TJhj9wMmjQvhk:0:0:root:/root:/bin/bash
```

Remover a senha, deixando a linha correspondente ao *root* da seguinte forma:

```
root::0:0:root:/root:/bin/bash
```

Rebootar a máquina normalmente, após rodar o comando *sync*, logar como *root* (não será solicitada a senha) e realizar a troca da senha do root com o comando passwd.

Muito fácil, não? Tanto para o usuário como para quem estiver passando. Qualquer um conseguirá ter acesso a TUDO que estiver em seu computador, em apenas alguns minutos. Fácil e assustador. Mas existem mecanismos para aperfeiçoar esta segurança. Podemos configurar o *lilo* com uma senha. Desta forma o boot somente será realizado se fornecermos a senha correta, impedindo o acesso por pessoas não autorizadas. O problema é que temos mais uma senha para memorizar e, em caso de esquecimento, aí então somente com um disquete de recuperação.

Normalmente, servidores críticos não são deixados sem senha de Lilo justamente por esta possibilidade; não é interessante que algum engraçadinho que pensa saber um pouco de Linux, dê um reboot na sua máquina e simplesmente mude a senha do *root*.

O arquivo */etc/lilo.conf* abaixo protege com a senha *svaI2ni8U* a imagem do Linux.

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.14-1cl
  label=linux
  root=/dev/hdb1
  read-only
  password=svaI2ni8U
```

Como se vê, a senha para boot está gravada no arquivo */etc/lilo.conf*. As permissões deste arquivo permitem a leitura universal. Para sistemas multi-usuários isto implica que qualquer um pode ver a senha gravada no arquivo */etc/lilo.conf*. Para realmente garantir a segurança, altere o modo de permissão deste arquivo para 0600, barrando o acesso a todos os usuários que não o *root*.

O lilo é um programa amplamente documentado. A documentação é distribuída no pacote *lilo-doc*.

```
$ rpm -ql lilo-doc
/usr/doc/lilo-doc-0.21/README
/usr/doc/lilo-doc-0.21/doc/Makefile
/usr/doc/lilo-doc-0.21/doc/README
/usr/doc/lilo-doc-0.21/doc/Technical_Guide.ps
/usr/doc/lilo-doc-0.21/doc/User_Guide.ps
```

```
/usr/doc/lilo-doc-0.21/doc/bootloader.fig
/usr/doc/lilo-doc-0.21/doc/bootloader.tex
/usr/doc/lilo-doc-0.21/doc/fullpage.sty
/usr/doc/lilo-doc-0.21/doc/image.fig
/usr/doc/lilo-doc-0.21/doc/image.tex
/usr/doc/lilo-doc-0.21/doc/map.fig
/usr/doc/lilo-doc-0.21/doc/map.tex
/usr/doc/lilo-doc-0.21/doc/other.fig
/usr/doc/lilo-doc-0.21/doc/other.tex
/usr/doc/lilo-doc-0.21/doc/parameter.fig
/usr/doc/lilo-doc-0.21/doc/parameter.tex
/usr/doc/lilo-doc-0.21/doc/rlatex
/usr/doc/lilo-doc-0.21/doc/t2a.pl
/usr/doc/lilo-doc-0.21/doc/tech.dvi
/usr/doc/lilo-doc-0.21/doc/tech.tex
/usr/doc/lilo-doc-0.21/doc/user.dvi
/usr/doc/lilo-doc-0.21/doc/user.tex
```

13.34 Como desmontar sistemas de arquivos em uso

Para desmontar sistemas de arquivos em uso, a melhor opção é combinar a saída do programa *lsof* com o comando *kill*. É claro que você vai avisar seus usuários antes para que eles possam encerrar suas tarefas com antecedência.

O comando *lsof*, ou *List Open Files*, indica todos os processos que estão acessando determinado arquivo ou sistema de arquivos. Veja um exemplo da saída deste programa:

```
% lsof /home
COMMAND   PID    USER  FD  TYPE  DEVICE SIZE/OFF  INODE NAME
lsof    19958 queiroz cwd  VDIR  10,  8     5120  1024 /home (/d)
ksh     26800 queiroz cwd  VDIR  10,  8     5120  1024 /home (/d)
ksh     26800 queiroz  63u VREG  10,  8     2384  1060 /home (/d)
```

```
lsof    28152 queiroz cwd  VDIR  10,  8     5120  1024 /home (/d)
```

A segunda coluna indica a identificação do processo, o que nos interessa mais diretamente. Vejamos então o script:

lsof.sh

```
#!/bin/bash
for pid in `lsof /home|grep -v COMMAND|awk '{print $2}'`
do
kill -9 $pid
done
umount /home
```

O comando *umount* tem que ser dado imediatamente após o falecimento dos processos que estão acessando o sistema de arquivos em questão para evitar que novos processos sejam criados, novamente impedindo que seja desmontado.

O pacote *lsof* pode ser encontrado no servidor de ftp anônimo da Unicamp em *http://ftp.unicamp.br/pub/unix-tools/lsof*.

13.35 /etc/inetd.conf

A maioria dos sistemas Unix vem configurada disponibilizando a maior parte dos serviços. Dependendo da finalidade da sua estação de trabalho uma grande parte dos processos ativados não são necessários.

O arquivo */etc/inetd.conf* contém uma lista de processos que são ativados sob demanda pelo daemon *inetd* (Internet Daemon). Vale a pena analisar este arquivo para verificar da real necessidade dos serviços listados neste arquivo ficarem disponíveis.

Outra razão é a segurança. Se a máquina é um desktop e seu usuário não

acompanha listas de segurança, o mais provável é que problemas de segurança com qualquer dos daemons ativados pelo inetd não serão corrigidos.

Em alguns casos, recomenda-se até mesmo a remoção do arquivo */etc/inetd.conf*, desabilitando todos os serviços por ele ativados. Um caso em que se pode fazer isto é quando se tem um servidor que funciona exclusivamente como servidor Web.

Dos daemons ativados através do daemon inetd, segue uma lista de processos que podem ser desativados sem maiores problemas:

- finger
- uucp
- http
- bootp
- dhcp_bootp
- tftp
- ntalk
- walld

Além destes podem existir outros, mas cada caso é um caso ...

13.36 logbook.sh – Registro de Atividades

Periodicamente, todos os administradores de sistemas são avaliados. E sempre surge a pergunta: *O que você fez de importante no ano que se passou?* E então, apesar de ter trabalhado como louco, a resposta não vem, você começa a gaguejar e o máximo que consegue deixar é a impressão de que gastou todo o seu tempo navegando na Web ou pior.

Para fazer com que neste dia, você possua dados para justificar o seu salário exorbitante (por exorbitante entenda-se exatamente aquilo que você ganha, não importa quanto), a shell script *logbook.sh* pode ser uma ferramenta bem útil.

A *shell script logbook.sh*, abaixo, fornece meios para que você registre de uma maneira fácil o que fez, gere relatórios impressos e envie o comunicado via mail para quem desejar. Todo arquivo é criado com um cabeçalho com o nome da pessoa que invocou a shell, a data, hora e título do documento, como abaixo:

```
================================================
RESPONSÁVEL: Rubens
DATA: 09/09/97
HORA: 12:54:43
Assunto: Script para registro de atividades
```

Outra utilidade prática, quando forem procurar o culpado por alguma coisa, se você registrou que fez o que devia ser feito e quando, você pode certificar-se de continuar ganhando o seu salário exorbitante.

O script *logbook.sh*, além da edição do arquivo, gera também um índice contendo o nome do arquivo e o título que fornecido:

```
Arquivo                Descrição
================================================
140897-17:02.doc ----- Migracao das Maquinas
140897-17:15.doc ----- Logbook
```

Além de todas estas utilidades, o *logbook.sh* serve também para registro de dicas. Aquela configuração que você levou um mês para terminar fica registrada. Se daqui a três anos você precisar realizar o mesmo serviço novamente, está tudo registrado. Você então pode realizar a mesma tarefa em um dia e utilizar o tempo restante do prazo que você especificou para navegar na Web e fazer tudo que sempre quis mas não tinha tempo.

E finalmente, estas dicas podem servir como um repositório de informações para o pessoal da sua área e ninguém mais vai te interromper durante as férias para perguntar como se configura uma impressora no *Conectiva Linux*. Não se preocupe em passar seu conhecimento para os outros. Enquanto eles estiverem configurando impressoras você poderá estar aprendendo outras coisas e mantendo sua dianteira e fama de guru.

Todos os documentos são criados sob o diretório Log. Se este diretório não existir, o script então irá criá-lo. Este script pode necessitar de algumas alterações, dependendo de variáveis tais como o seu PATH e particularidades do sistema operacional que você usa.

A seguir, a *shell script logbook.sh*:

logbook.sh

```
#! /bin/bash
#
#  Script para Registro das Atividades
#
# Teste existência diretório de Log
if [ -d $HOME/Log ]
then
  LOGDIR=$HOME/Log
else
  mkdir $HOME/Log
fi
umask 006
echo ""
echo ""
echo -n "ASSUNTO: "
read ASSUNTO
FILE=$LOGDIR/`date "+%d%m%y-%H:%M"`.doc
#FILE=$LOGDIR/`echo $ASSUNTO | sed "s/ /_/g"`
echo "================================================"\
  >> $FILE
echo "RESPONSÁVEL: `whoami | dd conv=ucase 2> /dev/null`"\
```

```
  >> $FILE
echo "‘date ’+DATA: %d/%m/%Y’‘" >> $FILE
echo "‘date ’+HORA: %H:%M:%S’‘" >> $FILE
echo "Assunto: $ASSUNTO" >> $FILE
echo "‘basename $FILE‘ ----- $ASSUNTO" \
  >> $LOGDIR/00index.txt
echo "" >> $FILE
echo "" >> $FILE
vi +$ $FILE
echo ""
echo ""
echo -n "Deseja imprimir este documento?  (s/n) [s] "
read ANS
case $ANS in
    s|S|"")
    echo ""
    lpr $FILE
    ;;
    *)
    ;;
esac
echo -n "Deseja enviar copias desta \
  mensagem a alguém ? (s/n) [s] "
read ANS
case $ANS in
    s|S|"")
    echo ""
    echo "para: \c"
    read recipients
    mail -s "$ASSUNTO" $recipients < $FILE
    ;;
    *)
    ;;
esac
```

13.37 ls

13.37.1 Opções de Uso

Um dos comandos mais utilizados em sistemas *Conectiva Linux* é o comando *ls*. Forma uma dupla inseparável com o comando *cd*.

Embora simples de se usar, existem algumas características do comando *ls* que podem nos ajudar a economizar tempo e trabalhar mais produtivamente.

Geralmente trabalhamos com o comando *ls* da seguinte forma:

```
$ ls
Desktop  TIPS2  dns   figs    list    queiroz   tips-tricks
GNUstep  TIPS3  doc   filenf  livro   root      tmp
IN.efr   bin    docs  html    nsmail  software
Mail     efr    humor passwd  tex
```

Desta forma, listamos os arquivos do diretório corrente pelo nome. O comando *ls*, sem nenhum parâmetro, não lista todos os arquivos. Os arquivos iniciados por ponto final (".") são omitidos. Estes arquivos são criados pelos aplicativos que usamos, tal como *Netscape*, *elm*, *pine*, shells e outros. Nestes arquivos são gravadas as informações de configuração utilizadas por estes programas e também a configuração de ambiente de trabalho dos usuários. São omitidos visto que não precisamos vê-los toda vez que listamos nosso diretório home. Veja só:

```
$ ls -a
.             .kde         .tcshrc   bin      html   software
..            .kderc       .viminfo  desktop  humor  tex
.Xdefaults    .mailcap     Desktop   dns      list   tips-tricks
.bash_logout  .mc          GNUstep   doc      livro  tmp
.bashrc       .mime.types  IN.efr    docs     nsmail
.cshrc        .ncftp       Mail      efr      passwd
```

```
.elm          .netscape   TIPS2     figs     queiroz
.exrc         .ssh        TIPS3     filenf   root
```

Como se pode ver, a listagem ficou consideravelmente maior.

Tomemos agora apenas as primeiras duas linhas da listagem anterior:

```
.             .kde        .tcshrc    bin     html    software
..            .kderc      .viminfo   desktop humor   tex
```

As entradas "." e ".." indicam respectivamente o diretório corrente e o diretório um nível acima. Todos os diretórios em sistemas Unix e derivados contém estas duas entradas. Ou seja, são perfeitamente dispensáveis de qualquer listagem.

O comando

```
$ ls -A
```

gera uma listagem completa, inclusive com os arquivos escondidos, porém não exibe as entradas para o diretório corrente (.) e o diretório um nível acima (..).

Outro problema quando se emite apenas o comando *ls* sem parâmetros, é que não conseguimos identificar o tipo de arquivos. Para remediar este problema podemos emitir o comando:

```
$ ls -l
total 79
drwxr-xr-x   5 queiroz  queiroz 1024 Sep 11 16:55 Desktop
drwxrwxr-x   5 queiroz  queiroz 1024 Oct 24 10:49 GNUstep
drwxr-xr-x   2 queiroz  queiroz 7168 Oct 16 20:22 IN.efr
drwx------   2 queiroz  queiroz 1024 Sep 11 17:14 Mail
drwxrwxr-x   5 queiroz  queiroz 1024 Dec 14 21:13 TIPS2
drwxrwxr-x   6 queiroz  queiroz 1024 Dec 14 21:13 TIPS3
```

```
drwxrwxr-x   2 queiroz  queiroz 1024 Jan 16 23:27 bin
drwx------   5 queiroz  queiroz 1024 Sep 22 18:09 desktop
drwxrwxr-x   2 queiroz  queiroz 1024 Oct 13 19:11 dns
drwx------   3 queiroz  queiroz 1024 Sep 11 17:01 doc
drwxr-xr-x   9 queiroz  queiroz 1024 Jan 23 13:06 docs
drwxrwxr-x   3 queiroz  queiroz 1024 Oct 24 18:58 efr
drwxrwxr-x   2 queiroz  queiroz 1024 Nov  9 22:17 figs
-r-xr-xr-x   1 queiroz  queiroz 1966 Sep 26 18:35 filenf
drwxrwxr-x   2 queiroz  queiroz 1024 Jan 23 12:55 html
-rw-rw-r--   1 queiroz  queiroz 7669 Nov  1 17:15 list
drwxrwxr-x   5 queiroz  queiroz 1024 Nov 12 13:42 livro
drwx------   2 queiroz  queiroz 1024 Sep 26 18:29 nsmail
drwxrwxr-x   2 queiroz  queiroz 1024 Jan 22 20:02 root
-rw-rw-r--   1 queiroz  queiroz 7669 Nov  9 21:09 software
drwxrwxr-x   2 queiroz  queiroz 1024 Nov 12 22:58 tex
drwxrwxr-x   6 queiroz  queiroz 1024 Jan 23 12:31 tips-tricks
drwxrwxr-x   2 queiroz  queiroz 1024 Jan 23 12:19 tmp
```

O primeiro caractere indica o tipo de arquivo. Na listagem anterior encontram-se presentes os seguintes tipos de arquivos:

Sigla	Descrição
l	link
d	diretório
-	arquivo regular

Entretanto, se a minha intenção é apenas saber o tipo de arquivo, a opção *-l* nos fornece muito mais informação de que realmente necessitamos. O comando

```
$ ls -F
Desktop/ TIPS2/   dns/  figs/   list    queiroz  tips-tricks/
GNUstep/ TIPS3/   doc/  filenf* livro/  root/    tmp/
IN.efr/  bin/     docs/ html/   nsmail/ software
```

```
Mail/    desktop/ efr/  humor/  passwd@ tex/
```

nos fornece a mesma informação, porém de forma muito mais sucinta. Os nomes de diretórios são seguidos por /, os links por @ e os arquivos regulares são apresentados normalmente. Os arquivos com o bit de execução ligado são seguidos pelo caractere "*".

13.37.2 Listagem de Permissões de Diretórios

Para se listar os modos de permissão de um diretório sem listar o seu conteúdo, utilize o comando *ls* com o sinal *-d*:

```
$ ls -ld /home
drwxr-xr-x   7 root root    1024 Dec 14 21:26 /home
```

Este comando retorna as permissões do diretório */home*, que é exatamente o que desejamos. Já o comando

```
$ ls -l /home
total 1
drwxr-xr-x   5 root root    1024 Oct 23 21:12 cesar
```

retorna as características dos conteúdo do diretório */home*. No exemplo acima o diretório */home* possui apenas um subdiretório, chamado cesar, cujas informações foram exibidas pelo comando emitido.

13.37.3 Listagem Ordenada da Esquerda para a Direita

Normalmente a saída do comando *ls* é formatada em múltiplas colunas. Todavia, se a saída for redirecionada para um outro comando através de um duto, é exibida apenas uma coluna, o que pode ser inconveniente para visualização na tela com os comandos *more* ou *less*. Para o processamento

por outro programa, como *awk*, a formatação em uma coluna é não apenas desejável como necessária.

Podemos todavia forçar que, mesmo utilizando um duto, a saída seja formatada em colunas:

```
$ ls -C | more
```

Este comando possui ainda outro inconveniente. A saída gerada é disposta alfabeticamente, de cima para baixo. Ou seja, se estivermos analisando a saída na tela de um computador, podemos ter algo do tipo:

```
00index.txt         970903.html         980227.doc
970303.doc          970903.src          980227.html
970303.html         970903.txt          980227.src
970303.src          970904.doc          980227.txt
970303.txt          970904.html         980228.doc
970304.doc          970904.src          980228.html
970304.html         970904.txt          980228.src
970304.src          970905.doc          980228.txt
970304.txt          970905.html         980301.doc
970305.doc          970905.src          980301.html
970305.html         970905.txt          980301.src
970305.src          970906.doc          980301.txt
970305.txt          970906.html         980302.doc
970306.doc          970906.src          980302.html
970306.html         970906.txt          980302.src
970306.src          970907.doc          980302.txt
970306.txt          970907.html         980303.doc
970307.doc          970907.src          980303.html
970307.html         970907.txt          980303.src
970307.src          970908.doc          980303.txt
970307.txt          970908.html         980306.doc
970310.doc          970908.src          980306.html
 --More--
```

Veja o final da primeira coluna. O arquivo é 970310.doc. O próximo na lista deveria ser 970310.html. Na segunda coluna o primeiro arquivo é 970903.html. Ou seja, a saída funciona da forma indicada na figura 13.1.

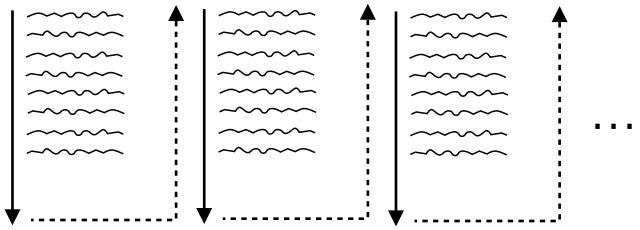

Figura 13.1: Funcionamento de uma listagem de diretórios

Este comportamento entretanto pode ser mudado. Podemos f azer com que a saída seja algo do tipo indicado na figura 13.2.

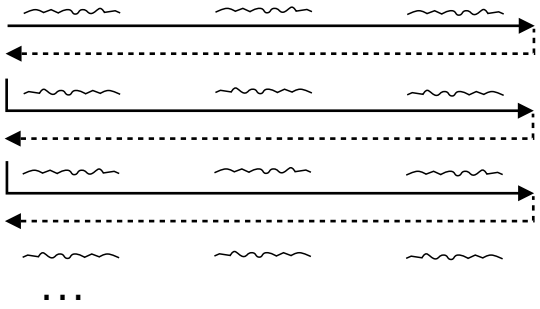

Figura 13.2: Comportamento alternativo para listagens de diretórios

O comando

```
$ ls -AFx | more
```

gera uma listagem em ordem alfabética ordenada da esquerda para a direita e não de cima para baixo, o que é mais conveniente para visualização na tela de um computador.

13.37.4 Listagem Seletiva de Arquivos

Às vezes necessitamos listar apenas todos os arquivos de um diretório à exceção de alguns, que atendem a determinada regra de formação.

Na Korn Shell (*ksh*) o comando

```
$ ls !(*.Z)
```

irá exibir apenas arquivos que NÃO terminem em *.Z*. Outro exemplo:

```
$ ls
a     a.gz  b.gz
$ ls !(*.gz)
a
```

13.37.5 Listando apenas os diretórios

Às vezes precisamos listar apenas os subdiretórios de um determinado diretório, sem nos preocuparmos com arquivos comuns.

Para isto podemos criar um alias, chamado *dir*:

- bash

  ```
  alias dir='ls -l | grep "^d"'
  ```

- cshell

  ```
  alias dir 'ls -l | grep "^d"'
  ```

13.38 man

Uma característica bastante interessante do comando man é a diretiva *-t*. Esta diretiva faz com que o comando *man* redirecione a saída gerada para um formatador que por sua vez irá criar um arquivo no formato postscript.

```
$ man -t tcpdump > tcpdump.ps
```

Em sistemas *Conectiva Linux* o formatador utilizado é o *groff*, que normalmente faz parte da instalação básica. Os arquivos postscript resultantes são bastante mais agradáveis de se ler e podem ser visualizados com o comando *gv* (*GhostView*) ou mesmo impressos em uma impressora postscript.

E não precisa nem dizer, 90% da informação está nas páginas de manuais *(man pages)*. E o melhor conselho para quem está começando a trabalhar com *Conectiva Linux* certamente é ler a documentação ...

13.39 Memória – Informações

As informações a respeito da memória de seu sistema podem ser obtidas a partir do arquivo */proc/meminfo*:

```
$ cat /proc/meminfo
        total:    used:    free:  shared: buffers:  cached:
Mem:  31379456 29294592  2084864 28839936   765952 13189120
Swap: 84602880   929792 83673088
MemTotal:     30644 kB
MemFree:       2036 kB
MemShared:    28164 kB
Buffers:        748 kB
Cached:       12880 kB
SwapTotal:    82620 kB
SwapFree:     81712 kB
```

Normalmente sistemas Linux reconhecem toda a memória existente em um sistema. Em meu sistema pessoal, conforme pode-se ver acima, estão disponíveis 32MB de memória real, 84MB de área de Swap e várias outras informações. Caso esta informação esteja diferente da configuração de seu sistema, que utiliza menos memória do que o disponível, basta editar o arquivo */etc/lilo.conf* e incluir na primeira linha a seguinte diretiva:

```
append="mem=128M"
```

O arquivo */etc/lilo.conf* completo deverá ficar como abaixo:

```
append="mem=128M"
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
other=/dev/hda1
    label=dos
    table=/dev/hda
image=/boot/vmlinuz-2.2.14-1cl
    label=linux
    root=/dev/hdb1
    read-only
    password=root
```

Uma vez realizadas as alterações é necessário executar o programa */sbin/lilo* para atualizar o registro de inicialização do sistema.

13.40 mkdir

13.40.1 Criação de uma árvore completa de diretórios

Para criar uma árvore completa de diretórios, invoque o comando *mkdir* com o sinal *-p*

O comando

```
$ mkdir -p /home/users/cs/ano1/graduacao/jose
```

é equivalente aos comandos

```
$ mkdir /home
$ mkdir /home/users
$ mkdir /home/users/cs
$ mkdir /home/users/cs/ano1
$ mkdir /home/users/cs/ano1/graduacao
$ mkdir /home/users/cs/ano1/graduacao/jose
```

13.41 MANPATH – Variável de Ambiente

Ao se usar o comando *man*, o sistema normalmente procura pelas páginas de ajuda no diretório */usr/man*.

Acontece que a maioria dos administradores instala os programas locais e de domínio público sob o diretório */usr/local*. As man pages também são armazenadas no diretório */usr/local/man*.

Para tornar as man pages instaladas sob o diretório */usr/local/man* disponíveis, é necessário alterar a variável de ambiente *MANPATH*:

```
export MANPATH=/usr/man:/usr/local/man
```

Sistemas *Conectiva Linux* possuem um arquivo de configuração, */etc/man.config*, onde são especificados valores a serem usados pelo sistema de busca de páginas de manuais. Neste arquivo, entre outras, encontramos as seguintes linhas:

```
MANPATH /usr/man
MANPATH /usr/local/man
MANPATH /usr/X11R6/man
MANPATH /usr/lib/perl5/man
```

Sempre que o comando *man* for invocado, a página solicitada será procurada em um dos diretórios acima: */usr/man*, */usr/local/man*, */usr/X11R6/man*, */usr/lib/perl5/man*. Se algum software novo tiver suas páginas de manual instaladas em algum diretório diferente receberemos uma mensagem de erro.

A grande vantagem da definição destas variáveis em um arquivo de configuração é justamente deixar o usuário final livre de tais preocupações.

13.42 MindTerm

Uma outra alternativa para quem busca clientes que implementem o protocolo SSH é o programa *MindTerm*.

Este programa é um cliente SSH inteiramente gratuito escrito totalmente em Java. Pode ser rodado como um programa normal ou como um applet em uma página Web.

O software e também maiores informações podem ser obtidas a partir do endereço *http://www.mindbright.se/mindterm*

13.43 more

O comando *more*, além da sua função básica de exibir um arquivo na tela aos poucos, oferece algumas outras facilidades.

Durante a exibição do arquivo, se você digitar a letra "**v** ", será invocado o comando *vi* para editar o arquivo sendo exibido.

Se você digitar a letra **"b"** , você consegue voltar uma página atrás.

Mas não é só isto, digitando "**h** ", você recebe uma página de ajuda, que explica as várias opções disponíveis.

Além disto tudo, o comando *more* aceita outras opções na linha de comando. Consulte a documentação online para maiores detalhes.

Podemos também examinar o conteúdo de vários arquivos:

```
$ more *.txt
```

Caso você não queira analisar todos os arquivos do começo ao fim, para passar de um arquivo para o próximo basta digitar "**:n** ", de forma semelhante a quando se editam vários arquivos com o editor *vi*.

13.44 Mudanças remotas em sistemas

Sempre que for fazer manutenções remotas, lembre-se que qualquer erro ao configurar roteamento, interfaces de redes e outras coisas ligadas à conectividade da sua máquina à rede, fará com que você tenha que se deslocar até a máquina em questão para corrigir o erro. Isto pode ser complicado se o computador estiver em outra cidade, estado ou país.

Por exemplo evite:

- Modificar a configuração de interfaces de rede com o comando *ifconfig*;
- Modificar o endereço de um *gateway*;

- Modificar a configuração de clientes e mestres de NIS;
- Encerrar processos de rede (*automount, gated, roted*, etc)
- Apagar a tabela de roteamento

Se realmente tiver que fazer algo do tipo, certifique-se de ter um contato telefônico com alguém onde o computador se encontra para fazer os necessários ajustes em caso de problemas.

13.45 Nessus - Ferramenta de Segurança para Unix

Nessus[1] é uma ferramenta gratuita para auditar a segurança de sistemas Unix. Possui uma interface gráfica agradável e realiza remotamente mais de 200 verificações de segurança.

13.46 ncftp

O programa *ncftp*, distribuído juntamente com o *Conectiva Linux*, provê a mesma funcionalidade do programa ftp padrão, oferecendo uma interface mais amigável e poderosa. Particularmente interessante são outros programas que fazem parte da distribuição, adequados à utilização em shell scripts. O primeiro deles, *ncftpget*, pode ser utilizado para se baixar vários arquivos, de múltiplos sítios, utilizando apenas uma diretiva para cada arquivo. Para tal basta criar um arquivo como

```
$ ncftpget ftp.unicamp.br /pub/sendmail.tar.gz /home
$ ncftpget wuarchive.wustl.edu /pub/README /home
$ ncftpget ftp.microsoft.com /pub/README /home
```

O primeiro comando irá trazer o arquivo *sendmail.tar.gz*, do servidor *ftp.unicamp.br*, o segundo irá trazer o arquivo *README* de *wuarchive.wustl.*

[1] *http://www.nessus.org/*

edu e o terceiro o arquivo *README* de *ftp.microsoft.com*. Todos os arquivos serão gravados no diretório */home*.

O programa *ncftpget* aceita URLs *(Uniform Resource Locators)* para se especificar o endereço dos arquivos remotos a serem baixados:

```
ncftpget ftp://ftp.unicamp.br/pub/sendmail.tar.gz
ncftpget ftp://wuarchive.wustl.edu/pub/README
ncftpget ftp://ftp.microsoft.com/pub/.welcome
```

Uma vez criado este arquivo a submissão pode ser efetuada em horário de pouco tráfego através de comandos tais como *at* ou *batch*. O programa *ncftpget* automaticamente se identifica como usuário *anonymous* e fornece como senha o endereço eletrônico da pessoa que o invocou, simplificando bastante o processo de transferência de arquivos. Mas não é só isto, o programa *ncftpget* oferece muitas outras opções, entre as quais pode-se citar, tentativas múltiplas de conexão (*-r*) e a recuperação recursiva (*-R*) de diretórios.

Por exemplo, para se transferir todo o diretório sendmail localizado no servidor da Unicamp:

```
$ ncftpget -R ftp://ftp.unicamp.br/pub/sendmail
```

E finalmente, deve-se ressaltar também a interface interativa do programa ncftp, que permite, entre outros recursos, o uso e criação bookmarks, indicador de progresso de transferência.

Mas para saber tudo que o programa *ncftp* faz, vale a pena ler a sua documentação.

A última versão do programa *ncftp* pode sempre ser obtida em *ftp://ftp.probe.net/pub/ncftp*

13.47 nologin – Restrição de Acesso

Para impedir que usuários acessem o sistema durante horários de manutenção programada, ou qualquer outra razão, sistemas *Conectiva Linux* oferecem um recurso bastante simples.

Basta criar um arquivo denominado */etc/nologin* e qualquer usuário, à exceção do usuário *root*, terá seu acesso ao sistema bloqueado.

É conveniente que o administrador coloque uma mensagem dentro deste arquivo justificando o bloqueio de acesso e o horário estimado para retorno às atividades normais. O conteúdo do arquivo */etc/nologin* é exibido sempre que algum usuário tente acessar o sistema.

Caso o arquivo */etc/nologin* esteja vazio, o usuário simplesmente terá o seu acesso bloqueado sem nenhuma explicação, o que poderá gerar telefonemas, possivelmente irados, à operação de computadores ou ao suporte de sistemas, o que ninguém deseja.

13.48 Partições Windows

Sistemas Linux são extremamente sociáveis, entendem praticamente qualquer tipo de sistema de arquivos existente. Podem acessar arquivos de sistemas OS/2, DOS, Windows NT, Windows 95/98 e mais alguns.

Para acessar estes arquivos a partir de seu sistema Linux, inclua no arquivo */etc/fstab* a seguinte linha:

```
/dev/hda1  /win95   vfat  defaults   0 0
```

O diretório de montagem do sistema Windows, */win95*, precisa ser criado previamente para que a montagem funcione. Caso não se deseje a montagem automática durante o boot, o mesmo comando pode ser emitido manualmente:

```
# mount -t vfat /dev/hda1 /win95
```

Como pode-se ver, o sistema do Windows está localizado no primeiro disco, *hda1*. Este normalmente é o recomendado, visto que o Windows não instala em outro disco que não o primeiro da primeira controladora de discos.

Os arquivos da partição do Windows pertencem ao usuário *root* para evitar que algum desavisado faça uma besteira maior.

13.49 Partições – Redimensionamento

Para redimensionar as partições de seu disco rígido, permitindo a instalação de outros sistemas operacionais, existem várias opções, comerciais ou não.

O primeiro deles, gratuito, é o *FIPS*. Este software normalmente é distribuído no diretório *tools*, *dosutils*, da distribuição *Conectiva Linux*.

Existem outras opções. A primeira delas é o software *Partition Resizer*, que pode ser obtido em *http://members.xoom.com/Zeleps/index.htm*

Existe também o software *Ranish Partition Manager*, que pode ser obtido no endereço *http://www.users.intercom.com/~ranish/part/* Nesta página está disponível para download um gerenciador de boot (*boot manager*).

Uma última recomendação, antes de diminuir o tamanho de uma partição já existente, é que o disco seja desfragmentado, agrupando os dados no início do disco. As versões do DOS superiores a 6.0 possuem o programa DEFRAG. Você pode usar também o programa *Norton Speedisk*.

13.50 Os Modos de Permissão no Unix

Todos os arquivos e diretórios em sistemas *Conectiva Linux* podem ser protegidos. Esta proteção se dá através da divisão dos usuários em três categorias principais: o dono do arquivo, o grupo ao qual pertence o dono do arquivo e os demais usuários.

Vejamos o perfil do usuário João. Ele trabalha na seção de contabilidade de sua empresa. No momento ele está trabalhando em uma proposta de alteração do sistema de contabilidade de sua empresa. Como a proposta ainda está em fase de definição, ele não quer que ninguém tenha acesso ao esboço de seu trabalho. Somente ele próprio quer ter acesso. Após a conclusão da proposta, o documento poderá ser então lido por todas as pessoas que pertencem ao grupo *contab*, mas apenas para leitura. João quer fazer todas as modificações para manter a coerência do documento.

```
$ ls -l
total 1
-rw-------   1 joao contab 820 Jan 30 14:55 proposta.txt
```

O comando *ls* nos fornece várias informações. Estamos particularmente interessados no primeiro campo. Este primeiro campo é subdividido em dez partes. A primeira indica o tipo de arquivo. Arquivos normais são indicados por "- ". Os próximos três campos indicam o que o dono do arquivo pode fazer. O João pode ler (r) e gravar (w) alterações no arquivo *proposta.txt*. Como ele deseja total privacidade, os próximos três campos, referentes ao grupo ao qual João pertence (contab) não podem nem ler nem gravar, visto que nenhum dos campos está ligado (--). É claro que os demais usuários, indicados pelos próximos três campos, também estão impedidos de acessar este arquivo.

[-]	[-][-][-]	[-][-][-]	[-][-][-]
tipo	dono	grupo	demais

Passemos então à fase seguinte. A proposta inicial está concluída e João quer a opinião de seus companheiros de trabalho. As permissões de acesso do arquivo *proposta.txt* são então alteradas para permitir que os integrantes de seu grupo de trabalho possam ler o documento. Neste ponto João não quer autorizar alterações feitas por outro que não ele.

```
$ ls -l
```

```
total 1
-rw-r-----   1 joao contab 820 Jan 30 14:55 proposta.txt
```

Vimos que a leitura para o grupo foi ativada.

No estágio final, a proposta deve ficar aberta para que todos da empresa possam opinar. Novamente, os privilégios de gravação permanecem apenas com o João.

```
$ ls -l
total 1
-rw-r--r--   1 joao contab 820 Jan 30 14:55 proposta.txt
```

Mas como fazer isto? Primeiramente devemos entender como funciona o mecanismo de atribuição de permissões? As três posições podem assumir os valores *rwx*, que significam *read* (leitura), *write* (gravação) e *eXecute* (execução) respectivamente.

As permissões são alteradas com o comando *chmod* . O comando *chmod* aceita como parâmetro os valores que definem que tipo de acesso determinado arquivo ou diretório pode ter.

Para alterar a proposta para que apenas ele, João, consiga ler os arquivos, deve ser dado o comando

```
$ chmod 600 proposta.txt
```

A tabela abaixo lista o valor numérico associado a cada tipo de permissão:

Permissão	Representação Simbólica	Representação Numérica
leitura	r	4
gravação	w	2
execução	x	1

O parâmetro fornecido ao comando *chmod*, *600*, significa que foram ativados os bits de leitura (4), somado ao bit de gravação (2), resultando em 6. As

demais permissões, grupo e universal, assumiram o valor *0*, ou seja, não foi outorgado nenhum privilégio de acesso a este arquivo a este grupo de pessoas.

Da mesma forma, para permitir ao grupo que leia o documento, foi emitido o comando

```
$ chmod 640 proposta.txt
```

Ativou-se aqui o bit de leitura para o grupo (valor 4) e apenas ele. A gravação continua vedada a outros que não o autor do documento.

Finalmente, após a conclusão do proposta, é garantido o privilégio de acesso universal, com o comando

```
$ chmod 644 proposta
```

Até o momento foram abordadas apenas as permissões de leitura e gravação. Existe entretanto um terceiro valor que pode ser ativado, a execução (o x em rwx). Este valor, quando ativado, indica que o arquivo pode ser executado, ou seja, é um programa ou uma shell script. Este bit deve necessariamente estar ligado para que o programa possa ser executado.

Para permitir a execução, leitura e gravação de um arquivo executável, deve ser efetuada sobre este arquivo a seguinte operação:

```
$ chmod 700 programa
$ ls -l
total 50
-rwx------   1 queiroz queiroz 49732 Feb  3 12:13 programa
```

Para permitir que apenas o dono realize alterações em um arquivo executável e que os demais o executem

```
$ chmod 755
```

```
$ ls -l programa
total 50
-rwxr-xr-x   1 queiroz queiroz 49732 Feb  3 12:13 programa
```

Vejamos então em mais detalhes:

-rwx	r-x	r-x
421	4-1	4-1
7	5	5

As permissões de leitura, gravação e execução somadas resultam no número 7. As permissões de leitura e execução resultam no número 5. Daí termos o parâmetro 755 passado ao comando *chmod*.

Uma outra forma de atribuir as permissões a arquivos é utilizar a representação alfabética. O comando

```
chmod u=rwx,g=r,o=r arquivo
```

irá atribuir, para o dono do arquivo (u) as permissões de leitura (r), gravação (w) e execução (x). Para o grupo ao qual o arquivo pertence (g), as permissões atribuídas são apenas de leitura (r) e para os demais usuários (o), a permissão também é de leitura:

```
% ls -l arquivo
-rwxr--r--   1 root root 937 Apr 11 20:47 arquivo
```

Com os sinais "+" e "-" podemos respectivamente ativar ou desativar determinadas permissões. Tomando o arquivo acima:

```
% chmod u-x arquivo
% ls -l arquivo
-r-xr--r--   1 root root 937 Apr 11 20:47 arquivo
```

Ou então

```
% chmod g+w arquivo
% ls -l
-r-xrw-r--   1 root root 937 Apr 11 20:47 arquivo
```

Certamente um pouco mais trabalhoso do que a representação numérica, porém mais fácil de memorizar.

13.51 Permissões de arquivos e diretórios em Sistemas Unix

A última linha de defesa contra invasores do sistema são as permissões oferecidas pelo sistema. Cada arquivo ou diretório possui três grupos de bits de permissão a ele associados: um grupo para o usuário ao qual o arquivo pertence, um grupo para os usuários do grupo ao qual o arquivo está associado e um grupo para todos os demais usuários.

Além das permissões normais (leitura, gravação e execução), existe um quarto bit que se ligado atribui as seguintes propriedades aos arquivos e diretórios:

- setuid
  Este bit, se ativado no grupo de permissões do dono do arquivo, indica que todos os que executarem este programa o estarão fazendo com os privilégios do proprietário do arquivo. Por exemplo, o programa *sendmail* é *setuid* para o usuário *root*, que lhe permite gravar arquivos na fila de mensagens do sistema, */var/spool/mqueue*, o que é vedado a usuários normais. Este bit não possui significado em arquivos não executáveis

- setgid
  Este bit atua da mesma forma que o setuid bit, com a diferença que

o programa será executado com as permissões do grupo ao qual pertence. Se aplicados a um diretório fazem com que os arquivos nele criados tenham o mesmo grupo do diretório.

- sticky
  O *sticky* bit instrui o sistema operacional a atuar diferentemente com relação à imagem executável do arquivo. É remanescente de versões antigas do Unix e possui pouco uso hoje. Diretórios que possuem modo 777 que têm este bit ligado não podem ser removidos (*/tmp* por exemplo).

Nas versões mais recentes de sistemas Unix foi acrescentado um novo significado ao sticky bit quando aplicado a diretórios. Quando este bit é ativado em um diretório, usuários não podem apagar ou renomear arquivos pertencentes a outros usuários. Isto é tipicamente útil no caso de diretórios como */tmp*. Normalmente o diretório */tmp* possui acesso universal para gravação, permitindo que qualquer usuário remova arquivos pertencentes a qualquer outro usuário. Ativando o *sticky* bit no diretório */tmp*, os usuários podem remover apenas seus próprios arquivos. Para ativar o *sticky* bit em um diretório, use o comando:

```
# chmod o+t /tmp
```

ou

```
# chmod 1777 /tmp
```

Ao emitirmos o comando:

```
% ls -ld /tmp
drwxrwxrwt   4 root root 2048 Feb 12 12:07 /tmp
```

O caracter "t" em *drwxrwxrwt* indica que o *sticky* bit deste diretório está ligado.

Ao se criar um arquivo normalmente todas as permissões são ativadas. Como isto raramente é o desejado, o valor do *umask* é usado para modificar o grupo de permissões com as quais um arquivo é criado. Ou seja, da mesma forma com que o comando *chmod* especifica quais bits devem ser ligados, o comando *umask* especifica quais bits devem ser desligados.

```
# umask
022
# umask 000
# touch arquivo
# ls -l arquivo
-rw-rw-rw-   1 root root  0 Feb 12 12:09 arquivo
# umask 022
# touch arquivo1
# ls -l arquivo1
-rw-r--r--   1 root root  0 Feb 12 12:10 arquivo1
```

O comando *umask*, emitido sem parâmetros, exibe o valor corrente. Em nosso exemplo o usuário root possui este valor definido como *022*. Em seguida emitimos o comando umask com o valor *000*. Com esta definição, quaisquer arquivos criados pelo usuário root terão permissão universal de gravação, o que é muito perigoso. Retornamos então para a umask padrão, *022*. Com este valor os arquivos criados passam então a ter a permissão *644*, gravação para o dono, *root*, e apenas leitura para os demais.

13.52 Propriedade de Arquivos Apagados

Ao se remover uma conta do sistema normalmente removemos o diretório de trabalho deste usuário. Todavia, arquivos pertencentes a este usuário que estiverem localizados em outros locais continuarão pertencendo à identificação numérica deste usuário.

Veja o exemplo:

```
# cd ~joao
# ls -l
total 1
-rw-r--r--   1 joao  joao       0 Feb 21 21:35 a
-rw-r--r--   1 joao  joao       0 Feb 21 21:35 b
drwxrwxr-x   2 joao  joao    1024 Jan 30 14:54 tmp
# id joao
uid=501(joao) gid=501(joao) grupos=501(joao)
# userdel joao
# ls -l
total 1
-rw-r--r--   1 501  501        0 Feb 21 21:35 a
-rw-r--r--   1 501  501        0 Feb 21 21:35 b
drwxrwxr-x   2 501  501     1024 Jan 30 14:54 tmp
```

Como podemos ver, os arquivos *a*, *b* e o diretório *tmp* pertencem ao usuário *joao*, que na verdade é identificado pelos números 501 (UID) e 501 (GID). Uma vez removida do sistema esta conta (joao), os arquivos permanecem no sistema a não ser que explicitamente removidos. Estes arquivos não mais estão associados a um usuário mas ao número associado ao usuário que os possuia, como podemos ver através da saída do comando *id joao*.

Se criarmos um novo usuário e lhe atribuirmos os atributos que possuíam a conta *joao* teremos que todos os arquivos que anteriormente pertenciam ao *joao* passarão a pertencer ao novo usuário. Se o *joao* foi o antigo presidente da empresa e o novo usuário alguém não muito confiável, segredos importantes podem passar a mãos erradas.

Para evitar tais problemas, sempre que forem removidos usuários, ou mesmo diariamente, execute o seguinte comando:

```
# find / -nouser -print | mail root
```

Importante, certifique-se de que alguém sempre leia o email do usuário *root* e tome as devidas providências.

13.53 Preenchimento Automático de Caminhos de Diretórios e Comandos

Outra característica bastante útil da shell padrão do Linux, *bash*, é o preenchimento automático. Para ir ao diretório */usr/local/src*, podemos fazer da seguinte forma

```
$ cd /u[TAB]/lo[TAB]/g[TAB]
```

Sempre que pressionarmos a tecla TAB o sistema automaticamente preencherá o nome do diretório para nós, desde é claro que não hajam conflitos. As teclas pressionadas acima nos levarão ao diretório */usr/local/games*. Note que para irmos ao diretório *local* precisamos digitar as letras *lo*, isto por que em */usr* existem os diretórios *lib*, *libexec* e *local*. Ao digitarmos a segunda letra, 'o', temos então um nome único e o sistema completa o restante. Se digitarmos a letra 'i', precisamos ir mais além, pois 'lib' e 'libexec' somente se diferenciam a partir da quarta letra.

Sempre que houver algum conflito, o sistema emitirá um alerta sonoro nos avisando que não foi possível completar o nome digitado. Se insistirmos pressionando novamente a tecla TAB, o sistema nos exibirá as opções existentes:

```
$ cd l[TAB][TAB]
lib      libexec  local
```

Escolhemos então a letra conveniente que permita o preenchimento adequado do caminho desejado.

Esta facilidade além de economizar tempo nos economiza também os nossos braços, ombros, pescoço e saúde. Quanto menos digitarmos melhor. Não se esqueçam da L.E.R (Lesão por Esforços Repetitivos).

13.54 Programação Shell

O grande diferencial de sistemas Linux é a poderosa linguagem de programação shell. A shell é o componente do sistema operacional que provê uma interface entre o sistema operacional e o usuário. Os comandos emitidos pelo usuário são interpretados e enviados ao sistema operacional para serem executados.

Além de seu grande poder, conta também a favor da programação shell a sua simplicidade. Sistemas *Conectiva Linux* têm como filosofia principal o fato de cada um de seus componentes deve executar, e bem, uma e apenas uma tarefa. Tarefas maiores são executadas através da combinação de vários elementos. Estes elementos são os comandos que compõem o sistema operacional. Shell scripts, de variados níveis de complexidade, podem ser criadas facilmente pela combinação de comandos do sistema operacional e estruturas simples de controle de laço.

As dicas que se seguem ilustram alguns exemplos do uso de shell scripts utilizadas normalmente por administradores de sistemas Linux. São abordados alguns conceitos importantes e se fornecem vários exemplos de criação de shell scripts diretamente a partir da linha de comandos.

13.54.1 Caminho de Busca de Executáveis

Quando se deseja saber onde se encontra determinado programa, o comando *which* pode ser utilizado. A variável de ambiente *PATH* contém a lista de diretórios que são pesquisados sempre que um comando é invocado sem que seja fornecido o caminho completo até ele. Para exibir com que valores a variável PATH está definida, podemos usar o comando *echo*:

```
$ echo $PATH
/usr/local/bin:/bin:/usr/bin:/home/queiroz/bin
```

O comando *which* fornece seu resultado simulando a busca que a shell realiza quando se executa um comando.

Onde se encontra o comando *vi*?

```
$ which vi
/bin/vi
```

Este comando irá percorrer todos os diretórios especificados na variável de ambiente *PATH* e irá retornar a primeira ocorrência encontrada. No caso acima, sempre que o comando *vi* for invocado será executado o programa contido dentro do diretório */bin*. Entretanto podem existir várias arquivos de mesmo nome dentro do caminho de busca. Caso queiramos identificar todas estas ocorrências, a shell script abaixo pode nos fornecer esta informação:

whereis.sh

```
#!/bin/bash
# buscar - procura por todas as ocorrências
# de um comando dentro do caminho de execução
for dir in `echo $PATH | sed 's/:/ /g'`
do
   if [ -e $dir/$1 ]
   then
     echo $dir/$1
   fi
done
```

A variável "**$1** " assume o valor fornecido na linha de comandos. A construção `if [ -e` *dir*`1]` testa a existência do comando fornecido no diretório definido na variável *$dir*. O comando *echo $PATH* retorna uma lista de diretórios separados pelo caractere "`:` ". Este caractere é substituído por um espaço em branco para que a variável *$dir* possa assumir corretamente os valores de cada um dos diretórios definidos na variável *$PATH*.

Ao usarmos esta shell script novamente para identificarmos todas as ocorrências do comando *vi* temos:

```
$ buscar vi
/bin/vi
/usr/bin/vi
```

Além do diretório */bin*, existe um comando chamado *vi* também dentro do diretório */usr/bin/*.

13.54.2 numger.sh - Geração de números

Algo que frequentemente necessitamos é de um programa que gere uma determinada faixa de números. Isto é algo bem simples de se fazer e conveniente de se ter à mão.

A shell script abaixo serve para fazer isto muito bem:

numger.sh

```
#! /bin/bash
first=$1
last=$2
while [ $first -le $last ]
do
  echo $first " "
  first=‘expr $first + 1‘
done
```

Para executá-lo basta fornecer o intervalo de números:

```
$ numger.sh 1 5
1
```

```
2
3
4
5
```

13.54.3 Criação de Scripts com o bit de Execução Ligado

Muitas vezes criamos shell scripts com o programa *vi*. Todavia o modo de execução destes arquivos normalmente (dependendo do valor que você definiu para o *umask)* é *644*, ou seja, o arquivo não é executável.

Uma shell script bastante simples que resolve este pequeno incômodo é a shell script *xvi.sh* (*eXecutable vi*). Ele usa o *vi* normalmente para se editar o arquivo, porém, após o fim da edição, altera o modo de execução do script.

xvi.sh

```
#!/bin/bash
# xvi - criação de shell scripts
# com o bit de execução ligado
#       (modo 755)
#

if [ $# -eq 0 ]; then
   echo 1>&2 Sintaxe: $0 arquivo[s]
   exit 1
fi

for file
do
 vi $file
 chmod 755 $file
done
```

13.54.4 Remoção de Acentuação

A maneira mais certa de garantir que uma mensagem seja lida corretamente em absolutamente todos os tipos de sistemas é não usar acentuação. Embora sistemas que não consigam interpretar palavras acentuadas corretamente sejam cada vez mais raros, se desejarmos universalidade total devemos então eliminar a acentuação.

É possível criar facilmente, através de uma shell script, uma versão de um documento onde as palavras não estejam acentuadas. Esta tarefa é realizada pelo comando *sed*.

A shell script possui dois componentes, um arquivo com os comandos propriamente ditos e um outro com diretivas para o comando *sed*.

tiraacento.sh

```
#! /bin/bash
MACROS=/home/queiroz/bin
entrada=$1
saida=$2
# Neste ponto verifica-se se o número  de
# parâmetros fornecido foi igual a  dois.
# Caso contrário, é escrito na  tela  uma
# mensagem indicando a  sintaxe  correta,
# atribuindo-se  o  valor  1 ao código de
# saída
if [ $# -ne 2 ]; then
  echo 1>&2 Sintaxe: $0 arquivo-entrada\
  arquivo-saida
  exit 1
# Caso o número de parâmetros esteja cor-
# reto, executa-se então o programa
else
   if [ -f $entrada ]; then
```

```
# A função deste if é verificar se o  ar-
# quivo de entrada  existe. Caso  exista,
# então é executado  o comando, caso con-
# trário, o processamento é encerrado com
# uma mensagem de erro e ao código de saí-
# da é atribuído o valor 1.
   sed -f $MACROS/tiraacento.sed $entrada\
    > $saida
 else
# Nunca se esqueça do aviso de erro.  Ca-
# so contrário o usuário do programa pode
# pensar que tudo deu certo.
      echo "$entrada não existe!!! "
      exit 1
   fi
fi
```

O arquivo *tiraacento.sed* contém as diretivas para o comando *sed*. Desta forma o caractere **ã** será substituído por **~a** e assim por diante. Caso se opte por remover totalmente a acentuação basta substituir **s/ã/~a/g** por **s/ã/a/g** .

A seguir, o arquivo *tiraacento.sed* onde se encontram os vários comandos *sed* a serem executados para realizar a conversão.

tiraacento.sed

```
s/ã/~a/g
s/Ã/~A/g
s/à/‘a/g
s/À/‘A/g
s/ô/\^o/g
s/ô/o/g
s/Õ/~O/g
```

```
s/é/e'/g
s/É/E'/g
s/á/a'/g
s/ó/o'/g
s/Á/A'/g
s/Ó/O'/g
s/ç/,c/g
s/Ç/,C/g
s/ê/\^e/g
s/Ê/\^E/g
s/ú/u'/g
s/Ú/U'/g
s/â/a\^/g
s/Â/\^A/g
s/í/i'/g
s/Í/I'/g
s/Ü/"U/g
s/ü/"u/g
```

Muito cuidado, nunca utilize o mesmo arquivo como entrada e saída. Você corre o risco de danificar irreversívelmente o arquivo original.

13.54.5 Verificação dos parâmetros em shell scripts

É fundamental que toda *shell script* verifique, antes de realizar qualquer ação, que o número correto de parâmetros foi fornecido por quem invoca o programa:

check.sh

```
#!/bin/bash
if [ $# -ne 3 ]; then
  echo  1>&2 Sintaxe: $0 a b c
```

```
   exit 999
fi
```

O script acima requer três parâmetros que, caso não fornecidos, impedem a execução do programa e definem o código de retorno com o valor 999. Normalmente um código de retorno diferente de zero indica a ocorrência de um erro.

13.54.6 Atribuição de Valores a Variáveis em Shell Scripts

Variáveis são sequências de letras, dígitos ou caracteres especiais.

A atribuição de valor a uma variável irá depender da shell sendo utilizada. Na *bash* e compatíveis (*ksh*, *sh*), atribui-se o valor a uma variável da seguinte forma:

```
DATA=6jun60
```

À variável *DATA* foi atribuído o valor *6jun60*.

Outra característica extremamente útil é atribuir o valor de comandos a variáveis. Por exemplo, suponhamos que queiramos que a variável *DATA* assuma o valor da data corrente, no formato **mmddaa** (mês, dia e ano):

```
DATA=‘date +%m%d%y‘
```

O comando *date*, como especificado acima, irá retornar o valor *052799*, onde *05* indica o mês de maio, *27* o dia do mês e *99* o ano.

Importante notar as aspas invertidas (‘). Ao se delimitar um comando por aspas invertidas você está indicando que está interessado no resultado do comando, que por sua vez será atribuído à variável.

Por exemplo, para enviar uma mensagem a todos os usuários de determinado computador:

```
#!/bin/bash
for usuario in `awk -F: '{print $1}' /etc/passwd`
do
  mail -s "Aviso Importante" $usuario < msg
  echo $user
done
```

A shell script acima irá obter, com o auxílio do programa *awk*, uma listagem de todos os usuários da máquina, enviando em seguida uma mensagem a todos eles. À variável *$usuario* serão atribuídos tantos valores quantos os retornados pelo comando

```
$ awk -F: '{print $1}' /etc/passwd
```

Observar que, na shell script, o comando está delimitado por aspas invertidas (`). Como os campos no arquivo */etc/passwd* são delimitados por "**:** ", faz-se necessária a especificação da diretiva "**-F:** " para que o comando *awk* possa selecionar corretamente a identificação do usuário (**$1** , ou o primeiro campo). Na linha onde se envia a mensagem o caracter "< " indica que a mensagem a ser enviada encontra-se dentro do arquivo *msg*. O título da mensagem é informado através da diretiva "**-s** " seguida pelo valor *Aviso Importante*. O comando *echo $user* é apenas informativo e serve para monitorar o progresso da execução do comando.

13.54.7 Parâmetros em Shell Scripts (Bourne Shell)

Uma shell script pode receber, na linha de comandos, parâmetros indicativos de valores necessários para o seu processamento.

Estes parâmetros recebem o nome de variáveis posicionais e são identificadas por *$0, $1, $2,* ...

- $0

  A variável **$0** indica o comando emitido. Por exemplo, no comando

```
$ ls a b c d
```

a variável **$0** assume o valor *ls*. A variável **$1** recebe o valor "**a** ", a variável **$2** recebe o valor **b** e assim por diante.

Esta variável, **$0** , é bastante utilizada para se enviar avisos ao usuário quanto à sintaxe correta de uso de comandos. Por exemplo:

```
echo "Sintaxe: $0 arquivo-entrada arquivo-saida"
```

Se a linha acima se encontrasse em uma shell script chamada *modname* e a invocássemos sem especificar corretamente seus parâmetros, receberíamos o seguinte aviso:

```
Sintaxe: modname arquivo-entrada arquivo-saida
```

- $#
  O número de parâmetros fornecidos a uma shell script é armazenado na variável **$#** . Constitui uma norma de boa programação verificar este valor no início da shell e emitir mensagem de erro caso incorreto:

```
if [ $# -ne 3 ]; then
   echo  1>&2 Sintaxe: $0 a b c
   exit 999
fi
```

No exemplo acima, são necessários três parâmetros. Caso este número de parâmetros não tenha sido fornecido, o processamento é encerrado e ao código de retorno é atribuído o valor *999*.

- $*
  Todos os parâmetros fornecidos como uma cadeia de caracteres separada por brancos encontram-se contidos na variável **$*** .

  Tomemos a shell script abaixo, chamada *parâmetros* que apenas ecoa para a tela a variável **$*** :

```
#!/bin/bash
echo $*
```

Ao invocarmos esta shell teremos o seguinte resultado:

```
$ parâmetros 1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8
```

- $?

  O código de retorno encontra-se na variável **$** . Códigos de retorno iguais a zero indicam que o programa conseguiu executar sua tarefa com sucesso, ao passo que valores diferentes indicam algum tipo de erro. Este valor pode ser definido dentro de uma shell script através da diretiva *exit*.

  recode.sh

```
#!/bin/bash
# recode.sh
if [ $# -lt 3 ]
then
 echo "Uso $0: DE-parâmetro \
      PARA-parâmetro arquivos" >&2
 exit 1
fi
```

  No exemplo que se segue, se o código de retorno for zero temos que *arquivo* é renomeado para *arquivo.20000101*. Caso contrário (código de retorno diferente de zero, *arquivo* é renomeado para *arquivo.erro*.

```
if [ $? -eq 0 ]; then
        mv arquivo arquivo.20000101
else
        mv arquivo arquivo.erro
fi
```

Ao invocarmos esta shell com os três parâmetros solicitados temos o código de retorno igual a 0:

```
$ recode a b c
$ echo $?
0
```

Se invocarmos esta shell novamente, desta vez sem fornecer os três parâmetros, o código de retorno, é definido como 1, indicando um erro de processamento:

```
$ recode
Uso ./recode: DE-parâmetro PARA-parâmetro arquivos
$ echo $?
1
```

- $$

O número do processo sob o qual a *shell script* está sendo executada é armazenado na variável **$$** . Muitos programadores utilizam esta variável para a criação de arquivos temporários:

```
#!/bin/bash
# pidteste - script para testes da variável $$
#
ENTRADA=entrada.$$
SAIDA=saida.$$
cat a b c d e > entrada.$$
cat f g h i j > saida.$$
ls /tmp/*.$$
```

A execução desta shell script irá nos mostrar que o valor atribuído ao processo sob a qual esta shell foi executada em nosso sistema é igual a 510. Este valor, é claro, varia a todo momento; cada processo

possui um identificador único para diferenciá-lo dos demais processos sendo executados. Daí o uso frequente da variável **$$** para uso em arquivos temporários. Mesmo que o programa que se utilize desta facilidade seja invocado diversas vezes simultâneamente, os arquivos temporários criados nunca terão nomes iguais.

```
$ pidteste
/tmp/entrada.510  /tmp/saida.510
```

13.54.8 Depurando shell scripts

Ao se invocar uma shell com a diretiva **-x** serão exibidos todos os comandos executados, atribuição de valores a variáveis, permitindo o acompanhamento do processamento e detecção de eventuais falhas.

Por exemplo, a shell

```
#!/bin/bash -x
# teste debug
echo ‘date‘
```

irá exibir a seguinte saída:

```
$ debug
+ [ -f /etc/bashrc ]
+ . /etc/bashrc
++ PS1=[\u@\h \W]\$
++ alias which=type -path
++ alias l=ls -laF --color=tty
++ alias ls=ls --color=tty
++ alias m=minicom -s -con -L
++ alias minicom=minicom -s -con -L
++ alias tm=tail -f /var/log/messages
++ alias tmm=tail -f /var/log/maillog
```

```
++ alias tms=tail -f /var/log/secure
++ alias cds=cd /etc/rc.d/init.d && ls
++ alias ufd=cd /mnt && umount floppy && ls
++ alias bck=tar cvzf /dev/fd0 ./tips-tricks
++ alias xtract=tar xvzf /dev/fd0
++ date
+ echo ter fev 8 21:14:13 BRDT 2000
ter fev 8 21:14:13 BRDT 2000
```

Além do comando *echo 'date'* são mostradas também as variáveis de ambiente definidas para o usuário que executou o comando.

13.55 rgrep – Recursive Grep

Muitas vezes necessitamos achar um arquivo que contenha uma determinada sequência de caracteres e não sabemos exatamente por onde começar. Este problema pode ser resolvido de maneira simples pelo comando *rgrep* que na verdade nada mais é do que uma shell script combinando os comandos *find* e *grep*.

Ao comando *grep* é fornecido como parâmetro uma lista de arquivos gerada a partir do diretório corrente. Fornece-se uma lista porque o comando *grep* quando atua a partir de um arquivo apenas, não precede os resultados com o nome do arquivo, ou seja, o resultado gerado, quando aplicado da forma que pretendemos aqui, não serve para nada.

A shell script é bastante simples, e é claro que pode ser aperfeiçoada. Considerou-se que a maioria dos usuários usa o comando *grep* a partir do diretório corrente, daí não ter sido incluída na shell a opção de se incluir o nome do diretório a partir do qual a busca será efetuada.

```
#! /bin/bash
#
#  RGREP
```

```
#
#  Esta shell script realiza um grep re-
#  cursivo, a partir  do  diretório cor-
#  rente,  sobre  a cadeia de caracteres
#  fornecida como parâmetro.
#
busca=$1

if [ $# -lt 1 ]; then
   echo 1>&2 Sintaxe: $0 cadeia_de_caracteres
   exit 1
else
   find . -type f -print | \
  xargs grep $busca > /tmp/rgrep.$$
fi
# Visualização do arquivo com os resultados
view /tmp/rgrep.$$
# Remocao dos arquivos de trabalho
rm /tmp/*.$$
```

A shell script *rgrep* foi incluída aqui apenas como um exemplo de programação shell, bastante simples por sinal. Felizmente para quem usa sistemas *Conectiva Linux*, existe um comando chamado precisamente *rgrep*, que realiza exatamente estas funções, com um grande número de opções, semelhantes às do comando *grep*.

Em sua forma mais simplificada, podemos utilizá-lo exatamente como a nossa shell script:

```
$ rgrep Linux /home/                    [1]
$ rgrep -i linux /home/                 [2]
$ rgrep -i linux .                      [3]
$ rgrep -N linux .                      [4]
```

Embora possamos especificar a diretiva "**-r** " o comportamente padrão do

comando *rgrep* é atuar no modo recursivo, como não podia deixar de ser.

Em [1] fazemos uma busca recursiva, a partir do diretório */home*, buscando por arquivos que contenham a palavra *Linux*. Em [2] realizamos exatamente o mesmo porém buscando pela palavra *linux* independentemente de sua grafia. As palavras *Linux*, *LiNuX*, ou qualquer variação atendem ao padrão especificado. Em [3] realizamos a busca a partir do diretório corrente. Finalmente, em [4], utilizamos o comando *rgrep* como o comando *grep*, ao lhe indicar que não desejamos que faça a busca recursivamente (diretiva *-N*).

13.56 rm

13.56.1 Proteção de diretórios

Caso existam diretórios que você realmente queira preservar contra deleções acidentais a todo custo (já pensou no diretório *root*, não), existem algumas medidas simples para implementar esta segurança adicional.

Se alguma pessoa inadvertidamente emitir o comando de deleção, será solicitada a confirmação para deleção a cada passo.

Sempre faça um teste destes procedimentos em uma área segura, como abaixo:

```
$ cd /tmp
$ mkdir teste
$ touch /tmp/teste/\-i
$ chmod 000 /tmp/teste/\-i
```

Sempre utilize o caminho completo para criar o arquivo.

```
$ cd teste
$ touch a e i o u
$ rm -rf *
```

Não se esqueça, sempre teste este procedimento em uma área SEGURA, como descrito acima e confirme a criação do arquivo chamado "**-i** " com o comando *ls* antes de fazer qualquer coisa.

Veja só, mesmo você não especificando o sinal *-i* a remoção de todos os arquivos irá requerer a sua confirmação. Isto se consegue por meio do arquivo chamado *-i*, que em nosso caso não é interpretado como um arquivo e sim como um sinal do comando *rm*. Muito interessante, não? Afinal de contas, quem ainda não apagou o sistema inteiro, ao menos uma vez na vida?

13.57 Remoção de Arquivos com Nomes Estranhos

Uma outra alternativa para se remover arquivos de nome fora do comum é através de sua idenficação no sistema de arquivos, seu i-node (*index node*), através do comando *ls -li*.

Este comando lista, além das informações normais, o número do *i-node* do arquivo, que aparece em primeiro lugar. Este número, por sua vez, pode ser fornecido ao comando *find*, ao qual podemos solicitar alguma ação sobre o mesmo, como por exemplo, sua remoção:

```
$ ls -li arqteste
159486 -rw-rw-r-- 1 queiroz queiroz 0 Feb 8 21:28 arqteste
$ find . -inum 159486 -exec rm {} \;
```

ou também

```
$ find . -inum 159486 | xargs rm
```

13.57.1 Remoção de Arquivos Iniciados em "-"

Uma dúvida bastante comum é como remover arquivos que possuam nomes que comecem com "- ". O *Conectiva Linux* utiliza o caractere "- " como

sinalizador de opções para comandos.

Caso tenhamos um arquivo chamado "**-a** " e tentemos removê-lo

```
$ rm -a
rm: illegal option -- a
Usage: rm [-firRe] [--] file ...
```

A operação não foi efetuada porque o parâmetro fornecido ao comando *rm* foi interpretado como uma opção e não como o nome de um arquivo. Para contornarmos este problema, basta preceder o nome do arquivo por um outro caractere "-":

```
$ rm -- -a
```

13.58 sed – Stream Editor

13.58.1 Substituição Global

Suponhamos que, devido a uma alteração na configuração de seu servidor Web, seja necessário que se substituam todas as ocorrências de uma determinada *URL* por uma outra. Ou seja, precisamos substituir uma cadeia de caracteres por uma outra, em um ou mais arquivos.

Abaixo encontra-se o código de uma shell script que realiza esta tarefa, por meio do comando *sed*.

O código da shell foi comentado para esclarecer as técnicas de programação.

subst.sh

```
#! /bin/bash
# subst.sh -  substituição de uma cadeia  de
# caracteres por outra
```

```
#
if [ $# -lt 3 ]
# na  linha de comando (mínimo de 3, duas
# cadeias de  caracteres (de, para) e no-
# me(s) do(s) arquivo(s).  Caso  os  três
# parâmetros  não tenham sido fornecidos,
# o  processamento é encerrado e é gerada
# uma mensagem de erro.

then
  echo "Uso $0: DE-parâmetro \
  PARA-parâmetro arquivos" >&2
  exit 1
fi

# atribui à variável DE o valor da cadeia
# de  caracteres a ser substituída e des-
# loca as variáveis($3 passa a ser $2, $2
# passa a ser $1). O deslocamento das va-
# riáveis é feito com o comando shift

DE=$1; shift

# atribui à variável PARA o  valor  final
# da cadeia de  caracteres.  $1 agora é o
# parâmetro de número 2, cadeia  final de
# caracteres.

PARA=$1;   shift

# Na entrada do laço, $1 representa o no-
# me  do primeiro arquivo onde serão efe-
# tuadas as alterações
```

```
until [ $# -eq 0 ]

# Continua no laço até que  o  número  de
# parâmetros  seja  igual a zero ou seja,
# até que não existam mais arquivos a se-
# rem processados.

do
# Testa a existência do arquivo
  if [ ! -r $1 ]
     then echo "arquivo $1 não existe" \
  >& 2;shift

# A modificação nos arquivos é feita com
# o comando sed, que redireciona a saída
# para um arquivo temporário, i que  irá
# possuir o nome  da  shell ($0) seguido
# do número de identificação do processo
# ($$)

  else   sed -e "s/\$DE/\$PARA/g" $1\
   > /tmp/$0$$
     mv /tmp/$0$$ $1;
     echo "alterações efetuadas em $1" \
  >&2
     shift
  fi
done
```

Por exemplo, o comando

```
$ subst.sh rubens joao *
```

substituirá todas as ocorrências de *rubens* por *joao* em todos os arquivos do

diretório atual.

13.58.2 Deleção de Linhas

O comando *sed* pode ser utilizado para realizar diversas operações em arquivos texto.

Diferentemente de um editor de textos normal, a operação realizada não é gravada diretamente no arquivo mas enviada para a saída padrão, ou STDOUT.

Tomemos por exemplo o arquivo chamado teste que contém as seguintes linhas:

```
1
2
3
4
5
```

Se executarmos o comando

```
sed 1d teste
```

aparecerá na tela o seguinte conteúdo:

```
2
3
4
5
```

Caso queiramos salvar o resultado desta operação devemos redirecionar a saída para um arquivo:

```
sed 1d teste > teste.out
```

Similarmente podemos especificar a deleção de qualquer linha que desejarmos.

```
sed 10d teste > teste.out
```

13.58.3 Caracteres delimitadores

Todo comando *sed* consiste basicamente na procura de determinados caracteres e na ação sobre os caracteres encontrados ou algum outro caractere que esteja na linha em questão.

Quando queremos fazer substituição do caractere "/ " a coisa fica mais complicada, pois este é justamente o caractere utilizado para delimitar os parâmetros recebidos.

O que poucos sabem é que os caracteres delimitadores podem ser substituídos. Se quisermos substituir em um arquivo a cadeia de caracteres */usr/local/bin* por */bin*, teríamos que fazer algo do tipo:

```
sed 's/\/usr/\local\/bin/\/bin/' teste > teste.out
```

o que é bastante complicado. Podemos fazer também da seguinte forma:

```
sed 's|/usr/local/bin|/bin|' teste > teste.out
```

A "/ " foi substituída pelo caractere "| " e ficou muito mais fácil digitar o comando.

13.58.4 Execução de Múltiplos Comandos

Podemos executar diversas ações, a partir de uma única invocação do comando *sed*. Tomando novamente o arquivo teste como exemplo:

```
1
2
3
4
5
```

e executando o comando abaixo

```
sed -e "s/^/<</" -e "s/$/>>/" teste
```

temos

```
<<1>>
<<2>>
<<3>>
<<4>>
<<5>>
```

13.58.5 Impressão de Linhas de Arquivo

O comando *sed* nos permite imprimir uma determinada linha de um arquivo. O comando:

```
sed -n -e 10p teste
```

irá imprimir a décima linha do arquivo chamado *teste*. A diretiva "**-n** " impede que o conteúdo do arquivo seja exibido ao mesmo tempo. Apenas a linha selecionada será exibida.

Isto pode ser usado em um programa onde o número da linha assume o valor de uma variável permitindo que se trabalhe com valores diferentes a cada momento.

13.59 Senhas – Geração automática

De tempos em tempos administradores de sistemas precisam criar um grande número de contas. Criá-las manualmente, além de ineficiente e propenso a erros, pode demorar uma eternidade. Imagine o caso de universidades ou escolas que recebem centenas ou mesmo milhares de novos alunos a cada semestre.

O *Conectiva Linux* possui um utilitário feito sob medida para esta situação chamado *mkpasswd*.

O comando *mkpasswd*, quando invocado sem parâmetros, retorna uma senha:

```
# mkpasswd
9nn7sJJvj
```

Uma *shell script* simples para criar quantas contas forem necessárias e que atribui a cada usuário uma senha pode ser criada facilmente.

senhas.sh

```
#!/bin/bash
for usuario in `cat novos-usuarios.txt`
    do
      useradd $usuario
      mkpasswd $usuario > $usuario.senha
# Criação da carta  ao  usuário, contendo
# sua  senha,  normas de uso e recomenda-
# ções gerais
cat > $usuario.carta << EOF
Prezado Usuário(a),

Conforme sua solicitação, foi criada uma
```

```
conta em nossos computadores centrais
com as seguintes especificações:

computador:    computador.dominio.com.br
identificação: $usuario
senha:         `echo $usuario.senha`

Solicitamos a memorização das informações con-
tidas neste documento e sua destruição em se-
guida devido ao caráter confidencial destas
informações.

Realize a troca de sua senha já em seu primei-
acesso para algo que lhe seja mais fácil de
lembrar.

Lembre-se, nunca divulgue a sua senha de aces-
so para ninguém. A segurança de seus dados e
do sistema como um todo dependem de você.

Para maiores informações consulte o endereço
http://www.dominio.com.br/suporte ou envie uma
mensagem para suporte@dominio.com.br

Atenciosamente,

Suporte Técnico - Centro de Computação
EOF
    lpr $usuario.carta
    rm $usuario.*
  done
```

O processo acima cria a conta do novo usuário, atribui-lhe uma senha inicial de acesso e imprime, na impressora padrão do sistema, uma carta a ser

entregue ao novo usuário.

Os comandos *useradd* e *mkpasswd* aceitam diversas opções, que podem ser usadas para especificar com mais precisão o ambiente do usuário. Para simplicidade de entendimento, os comandos empregados na shell script acima utilizaram os valores padrão. Após o processamento todos os arquivos temporários são removidos.

Problemas acontecem. Recomendamos sempre que se gere um backup dos arquivos envolvidos (*/etc/passwd*, */etc/shadow* e */etc/group)* antes da execução deste script.

13.60 Senhas Seguras

Um problema bastante comum é que os usuários de um sistema, quando não conseguem memorizar as senhas que lhe são atribuídas pelos administradores de sistema, geralmente a colam em seus terminais ou algum outro local visível, comprometendo a segurança do sistema e de seus dados pessoais.

Senhas como nomes, objetos, animais, nome invertido, carros, apenas para citar alguns, são facilmente quebradas por programas criados para este fim, disponíveis em grande quantidade na Internet.

Mas como fazer? Se a senha é fácil de se lembrar, é fácil de ser quebrada. Então como criar uma senha que seja difícil de ser adivinhada e ao mesmo tempo seja fácil de ser lembrada?

Uma das muitas alternativas é pensar em algum nome favorito e fazer a transposição de letras em números. Alguém que seja fã do cantor George Harrison, dos Beatles, poderia montar uma senha como *930r938a* onde a letra *g* foi substituída por *9* (G de cabeça para baixo), a letra *e* pelo número 3, a letra *o* pelo número 0. O 8 substitui o *H*.

Uma outra maneira é utilizar uma de suas frases favoritas adotando as primeiras letras de cada palavra para montar a senha. A frase *Eu fui no Tororó, beber água e não achei* pode nos ajudar a montar a senha *efntbaena*,

bastante segura. A não ser que você passe o dia inteiro cantando esta frase, ninguém vai conseguir adivinhar a sua senha.

13.61 slice

Este comando incorpora toda a funcionalidade dos comandos *split* e *csplit*, abordados anteriormente. Ao contrário dos comandos *split* e *csplit*, o comando *slice* não é padrão em sistemas Linux. Ele faz parte da distribuição de utilitários para unix, chamada unix-c, e disponível, entre outros lugares, em *http://ftp.unicamp.br/pub/unix-c*.

Podemos utilizá-lo, por exemplo, para dividir um arquivo sempre que for encontrada a cadeia de caracteres *###*

A cadeia de caracteres *###* deve ser eliminada (*-x*) dos arquivos resultantes:

```
$ slice -f arq1 -e "###" -x
```

Além disto, o nome dos arquivos gerados pode ser configurado através de algumas diretivas aceitas pelo comando *slice*. Caso o arquivo original contenha as linhas:

```
---<CUT>--- ARQ1
abcdefghijklmno
---<CUT>--- ARQ2
abcdefghijklmno
---<CUT>--- ARQ3
abcdefghijklmno
---<CUT>--- ARQ4
abcdefghijklmno
---<CUT>--- ARQ5
abcdefghijklmno
---<CUT>--- ARQ6
```

Analisemos o comando

```
$ slice -f arq1 -e "---<CUT>---" -x x.#2
$ ls
arq1  x.ARQ1  x.ARQ2  x.ARQ3  x.ARQ4  x.ARQ5  x.ARQ6
```

Os arquivos gerados receberam o prefixo *x.* e o sufixo é o segundo (#2) campo da linha que preencheu os requisitos para divisão dos arquivos, indicado pelo parâmetro (`-e ¨-<CUT>--¨`), em nosso caso a cadeia de caracteres $ARQx$, onde x varia de 1 a 6.

13.62 paste

O comando *paste* serve para colar o conteúdo de dois arquivos lado a lado. Por exemplo, tomemos os arquivos *arq1* e *arq2*:

arq1

```
1
2
3
4
5
```

arq2

```
a
b
c
d
e
```

O comando

```
$ paste arq1 arq2 > arq3
```

resultaria no arquivo *arq3* com o seguinte conteúdo:

```
1 a
2 b
3 c
4 d
5 e
```

Já o comando

```
$ paste -s arq1 arq2 > arq3
```

resultará no arquivo arq3 com o conteúdo abaixo

```
1       2       3       4       5
a       b       c       d       e
```

13.63 slocate – Secure Locate

Sistemas *Conectiva Linux* possuem uma excelente ferramenta para nos ajudar na localização de arquivos na estrutura de diretórios, o comando *slocate*.

O comando *slocate* cria um banco de dados contendo a listagem dos arquivos do sistema e sua localização na estrutura de diretórios. O banco de dados criado pelo comando *slocate* fica localizado no diretório */var/lib/slocate/* e chama-se *slocate.db*.

O programa *slocate* armazena também, além do nome, as permissões de leitura e propriedade dos arquivos de maneira a garantir que ninguém obtenha informações sobre arquivos aos quais não tenha acesso. Daí o seu nome, *Secure Locate*.

O banco de dados é criado emitindo-se o comando

```
# slocate -u
```

Este comando irá gravar no arquivo */var/lib/slocate/slocate.db* a localização e o nome de todos os arquivos e diretórios a partir do diretório raiz.

Usuários comuns podem também criar seu índice personalizado. A restrição é que o banco de dados (slocate.db) deve ser gravado em um local ao qual o usuário tenha acesso, como por exemplo seu diretório de trabalho pessoal.

```
$ slocate -U /home/usuario -d .slocate.db
```

O banco de dados pessoal foi chamado de *.slocate.db*. O nome inicia-se em ". " para que não conste nas listagens normais do diretório de trabalho.

Ao invocar o comando *slocate* sem parâmetros, o banco de dados pesquisado será o do sistema (*/var/lib/slocate/slocate.db*). Para consultar o banco de dados pessoal, utilizar a opção *–database*:

```
$ slocate --database=.slocate.db tex
```

Forneceu-se no exemplo a instrução de se buscar por arquivos que contenham os caracteres *tex* em seu nome.

O tempo de resposta do comando *slocate* é excelente. Raríssimas vezes será necessária a criação de um banco de dados pessoal, a não ser que o sistema seja muito grande e contenha centenas de milhares de arquivos. Mesmo nestas condições a busca é bastante eficiente.

Caso realmente se opte por um banco de dados individual, uma boa alternativa é criar um alias que especifique as diretivas apropriadas:

```
alias Slocate="slocate --database=$HOME/.slocate.db"
```

Desta forma, se especificarmos *Slocate* estaremos consultando o banco de dados pessoal. O comando *slocate* continuará fazendo uma busca na listagem completa.

O acesso a este banco de dados é feito através do comando *locate*. Usá-lo é extremamente simples, basta fornecer como parâmetro a cadeia de caracteres que se deseja localizar no sistema.

Finalmente, principalmente em sistemas bastante ativos, com uma grande taxa de modificação do sistema de arquivos, o mais indicado é automatizar a atualização do banco de dados *slocate.db*. As seguintes entradas na *crontab* realizam esta atualização uma vez a cada hora:

```
0 * * * * slocate -u
```

Para usuários normais

```
0 * * * * slocate -U /home/usuario -d .slocate.db
```

13.64 sort

13.64.1 Ordenação por Campos

O comando *sort*, também oferece inúmeras facilidades interessantes. Tomemos o arquivo **arq1** como exemplo:

arq1

```
1:2:3:4:5:6
1:1:3:4:5:6
1:4:3:4:5:6
1:2:3:4:5:6
1:0:3:4:5:6
```

```
1:2:3:4:5:6
1:7:3:4:5:6
1:2:3:4:5:6
1:0:3:4:5:6
1:9:3:4:5:6
```

O comando

```
$ sort -t: +1 -n arq1
```

irá gerar a seguinte saída

```
1:0:3:4:5:6
1:0:3:4:5:6
1:1:3:4:5:6
1:2:3:4:5:6
1:2:3:4:5:6
1:2:3:4:5:6
1:2:3:4:5:6
1:4:3:4:5:6
1:7:3:4:5:6
```

Observar que o segundo campo está ordenado numericamente em ordem crescente. Os campos deste arquivo são separados por "**:** ". O tipo de separador é indicado pelo sinal "**-t:** ". Em seguida ao sinal "**-t** " podemos indicar qualquer tipo de separador. O campo a ser ordenado é indicado pelo sinal "+**1** ". Para o comando *sort* a contagem dos campos inicia-se por 0, desta forma, o valor "+**1** " irá indicar na realidade o segundo campo do arquivo. A ordenação também pode ser feita numericamente, do maior para o menor valor:

```
$ sort -t: +1 -nr arq1
1:9:3:4:5:6
```

```
1:7:3:4:5:6
1:4:3:4:5:6
1:2:3:4:5:6
1:2:3:4:5:6
1:2:3:4:5:6
1:2:3:4:5:6
1:1:3:4:5:6
1:0:3:4:5:6
1:0:3:4:5:6
```

13.64.2 Ordenação e Combinação de Arquivos

O comando *sort*, na sua forma mais simples, serve para ordenar o conteúdo de um arquivo. Tomemos o arquivo:

arq1

```
x
a
h
j
k
```

O comando abaixo, executado sobre o arquivo *arq1*, irá gerar a saída:

```
$ sort arq1
a
h
j
k
x
```

Além desta função, o comando *sort* também pode ser utilizado para combinar dois arquivos diferentes. Os arquivos sobre os quais o comando *sort*

irá atuar já devem ter sido previamente ordenados:

arq1

```
aa
yy
```

arq2

```
bb
zz
```

O comando

```
$ sort -m arq1 arq2
```

irá exibir na tela

```
aa
bb
yy
zz
```

13.64.3 Redirecionamento para Arquivos

A saída do comando *sort*, em todos os exemplos apresentados, tem sido redirecionada para a tela. Caso queiramos redirecionar esta saída para um arquivo para processamento posterior, temos duas opções equivalentes:

```
$ sort arq1 arq2 > arq3
```

ou

```
$ sort arq1 arq2 -o arq3
```

13.64.4 Comparação de Caracteres

Uma outra característica interessante do comando *sort* é a possibilidade de fazer as comparações sobre os parâmetros convertidos para minúsculas (sinal -*f*).

Tomemos os arquivos *arq1* e *arq2*:

arq1

```
AA
XX
```

arq2

```
bb
kk
```

O comando

```
$ sort arq1 arq2
AA
XX
bb
kk
```

irá gerar uma saída onde a ordenação será feita primeiramente sobre as letras maiúsculas e em seguida as minúsculas, ou seja, *A-Z* e em seguida *a-z*. Já o comando abaixo

```
$ sort -f arq1 arq2
AA
bb
```

```
kk
XX
```

irá realizar a ordenação dos arquivos independentemente das palavras estarem grafadas em maiúsculas ou minúsculas.

13.64.5 Ordenação com Remoção de Linhas Duplicadas

O comando *sort* pode também ser utilizado para ordenar arquivos removendo eventuais linhas duplicadas. Tomemos o arquivo *arq1*:

arq1

```
joao
maria
jose
maria
joao
heitor
```

O comando

```
$ sort -u arq1
```

irá gerar a saída abaixo

```
heitor
joao
jose
maria
```

A diretiva *-u* fez com que a saída gerada contivesse apenas uma ocorrência de cada uma das linhas.

13.65 split

13.65.1 Divisão de arquivos por número de bytes e linhas

Muitas vezes precisamos dividir um arquivo em vários outros menores, seguindo alguma convenção. Para isto podemos usar o comando *split*.

O comando *split* nos permite dividir um arquivo baseando-se no número de linhas ou número de bytes que cada arquivo novo deve conter.

Por exemplo:

```
$ split -l 10 /etc/passwd
```

Este comando criará vários arquivos denominados *xaa*, *xab*, *xac*, e assim por diante. Nem sempre estes nomes são os mais convenientes. Neste caso podemos, com o acréscimo de mais um parâmetro, determinar o sufixo do nome dos arquivos que serão criados:

```
$ split -l 10 /etc/passwd pas-
$ ls
pas-aa  pas-ab  pas-ac  pas-ad  pas-ae  pas-af  pas-ag  pas-ah
```

Os arquivos criados passaram a conter o prefixo *pas-*, permitindo identificar mais claramente os contadores dos arquivos (*aa, ab, ac, ...*)

Além do particionamento em linhas, o comando *split*, quando invocado com a opção *b*, irá efetuar a divisão do arquivo baseando-se no número de bytes:

```
$ split -b 32k /etc/passwd pas-
```

ou então

```
$ split -b 32 /etc/passwd pas-
```

ou ainda

```
$ split -b 32m /etc/passwd pas-
```

No primeiro exemplo, o arquivo */etc/passwd* será dividido em vários arquivos de 32 kbytes cada um, ao passo que no segundo exemplo, o arquivo será dividido em arquivos de 32 bytes cada. No terceiro exemplo, o arquivo */etc/passwd* é dividido em arquivos de 32MB cada (pouco provável :-)

13.66 Divisão de um Arquivo Para Gravação em Disquetes

Para dividir um arquivo em vários para gravação em disquetes de 1.44MB, o comando *split* é uma das opções possíveis:

```
$ split -b 1400000
```

O comando *split* irá criar vários arquivos chamados *xaa, xab, xac,* e assim por diante. Para restaurá-los basta usar o comando *cat*, como abaixo:

```
$ cat x* > arquivo-original
```

13.67 su – Substitute User

O comando *su* (*Substitute User*), nos permite trocar a identidade de um usuário por outra. É mais frequentemente utilizado por administradores de sistemas que trabalham a maior parte do tempo sob suas identificações normais e que de tempos em tempos precisam executar tarefas que requerem privilégios que somente o superusuário possui.

Quando invocado sem nenhum parâmetro, o comando *su* assume que se deseja assumir a identifidade do usuário root:

```
$ su
```

Neste caso, após fornecer a senha, o usuário adquire os privilégios do superusuário porém com todas as variáveis de ambiente permanecem inalteradas. São utilizadas as definições do usuário original, ou seja, os arquivos .cshrc, login, .profile, etc. do *root* não são lidos.

Já o comando

```
$ su -
```

faz com que o usuário se torne *root* com toda a configuração do ambiente do usuário root. Tal como nascer de novo.

A segunda opção é particularmente mais interessante. É claro que o usuário *root* não é um usuário normal, e como tal, o seu ambiente também é configurado de forma diferente, levando em consideração fatores tais como segurança, programas que podem ser acessados e vários outros. Invocar o comando *su* desta forma também impede que o usuário *root* execute programas que estão definidos no ambiente de usuários normais com os privilégios do superusuário, o que pode vir a comprometer a segurança do sistema.

Um outro ponto a ser lembrado é que o superusuário pode assumir a identidade de qualquer outro usuário sem fornecer senha, bastando para isto invocar o comando *su* fornecendo como parâmetro o nome do usuário que deseja se tornar:

```
# su - joao
```

13.67.1 Suspensão de processos

Quando se é administrador de sistemas, não é conveniente trabalhar diretamente na conta do super usuário (*root).* Eventuais erros têm consequências muito mais trágicas quando quem erra é o usuário root.

Mas também não se pode negar que é aborrecido executar o comando *su* todas as horas em que se precisa realizar uma tarefa que requeira privilégios.

Uma maneira fácil de se contornar este problema é colocar o processo *su* em background quando não se necessitar mais dos privilégios.

```
$ su
Password:
# suspend
[1]+  Stopped (signal)        su
$ fg
su
#
```

Observe bem, se emitirmos o comando "`su -`" já não funciona:

```
$ su -
Password:
[root@paris /root]# suspend
suspend: Can't suspend a login shell
```

Isto se dá devido ao fato de que o comando "`su -`" é equivalente a um novo login, daí a mensagem de erro afirmando que não pode suspender uma shell de login.

13.68 tail

O comando *tail* pode ser utilizado para examinar as últimas linhas de um arquivo.

O comando

```
$ tail /etc/passwd
```

irá exibir as dez últimas linhas do arquivo */etc/passwd*

É possível também especificar o número de linhas a serem exibidas, ao invés das dez linhas que o comando adota como padrão:

```
$ tail -n 100 /etc/passwd
```

No exemplo acima, serão exibidas as 100 últimas linhas do arquivo */etc/passwd*.

Uma diretiva muito útil é "**-f** " que permite a visualização dinâmica de um arquivo, ou seja, as linhas são exibidas na tela na medida em que são geradas. Esta facilidade é particularmente interessante quando se faz a compilação de um software redirecionando a saída para um arquivo. Através do comando tail pode-se acompanhar toda a compilação ao mesmo tempo em que as informações são gravadas em um arquivo:

```
$ make >& make.log
$ tail -f make.log
```

13.69 tee

O comando *tee* permite que a saída de um comando seja direcionada simultaneamente para dois destinos. Por exemplo, gravada em um arquivo ao mesmo tempo em que é exibida na tela.

```
% ls | tee saida.txt
```

A listagem do diretório é exibida na tela ao mesmo tempo em que é gravada no arquivo *saida.txt*.

O comando *tee* aceita a diretiva "**-a** " indicando que a saída do comando deve ser acrescida ao conteúdo do arquivo especificado e a diretiva "**-i** " que especifica que interrupções devem ser ignoradas.

O comando *script* oferece funcionalidade semelhante, porém mais abrangente. O comando script registra tudo o que ocorre em uma sessão interativa, ao passo que o comando *tee* grava o resultado de apenas um comando.

Uma outra possibilidade é redirecionar a saída de um comando executado em uma tela para uma outra:

```
% ls | tee /dev/pts/4
```

Neste exemplo, a saída do comando *ls* será exibida na tela original e na tela identificada por /dev/pts/4.

13.70 tr – Translate

13.70.1 Maiúsculas/Minúsculas

Em sistemas Linux, arquivos com o nome *TESTE*, *teste*, ou ainda *TeStE*, são entidades totalmente distintas. Diversamente de sistemas Windows, onde a combinação de letras maiúsculas ou minúsculas na referência a arquivos ou diretórios não faz diferença, em sistemas Linux a ignorância deste fato pode levar a efeitos totalmente diversos.

O mais conveniente é usar arquivos e diretórios sempre com letras minúsculas, para facilidade de digitação. A grafia com letras maiúsculas pode ser utilizada para fazer com que o arquivo ou diretório apareça primeiro na listagem, pois as letras maiúsculas têm precedência sobre as minúsculas:

```
$ ls
Relatorio arquivo.doc index.html relatorio teste.txt
```

Em algumas situações, particularmente quando arquivos são transferidos de um sistema para outro, freqüentemente nos deparamos com situações em que todos os arquivos transferidos estão grafados com letras maiúsculas. A

mudança manual de todos estes nomes pode ser cansativa e tomar muito tempo.

Felizmente, sistemas *Conectiva Linux* nos oferecem o comando *tr*, ou *translate*, que nos permitem traduzir um caractere por outro.

lcase.sh

```
#!/bin/bash
if [ $# = 0 ]
# Testa o número de parâmetros fornecidos.
# Se igual  a  zero emite mensagem de erro
# e aborta o processamento.
  then
    echo "sintaxe: lcase [arquivo ...]"
    exit
fi
for ARQUIVO in $*
# A variável arquivo irá assumir sucessi-
# vamente o valor de todos os  parâmetros
# fornecidos na # linha de comandos.
  do
    mv $ARQUIVO ‘echo $ARQUIVO |\
    tr ’[A-Z]’ ’[a-z]‘
# o comando tr irá converter todas as le-
# tras  no  intervalo  de A a Z (primeiro
# parâmetro)  em seus equivalentes no in-
# tervalo "a" até "z" (segundo-parâmetro).
  done
```

Se por alguma razão desejarmos fazer o contrário, a conversão de nomes de arquivos ou diretórios de maiúsculas para minúsculas, basta inverter a ordem dos parâmetros fornecidos ao comando *tr*:

```
tr '[a-z]' '[A-Z]'
```

13.71 traceroute

Você já parou para pensar por onde passam os seus dados em suas viagens pela Internet? Se você quer saber, existe um comando em sistemas *Conectiva Linux* que lhe fornece estas informações.

Este comando chama-se *traceroute*. Para determinar o caminho percorrido de meu computador até o servidor ftp da Universidade de Washington basta emitir o comando:

```
# traceroute wuarchive.wustl.edu
traceroute to wuarchive.wustl.edu (128.252.135.4),
  30 hops max, 40 byte packets
 1 paris.unicamp.br (143.106.30.11) 9 ms 2 ms 2 ms
 2 cmp-gw.unicamp.br (143.106.10.40) 10 ms 3 ms 3 ms
 3 ansp-gw.unicamp.br (143.106.1.45) 4 ms 4 ms 4 ms
 4 ansprd2.unicamp.br (143.106.70.1) 7 ms 5 ms 6 ms
 5 143.108.5.7 (143.108.5.7) 156 ms * 186 ms
 6 143.108.5.1 (143.108.5.1) 178 ms 184 ms 146 ms
 7 delta.cora.br (143.108.13.3) 173 ms 173 ms 207 ms
 8 mix.mci.net (204.189.152.193) 514 ms 391 ms 341 ms
 9 * core1.Washington.mci.net (204.70.2.1) 365 ms *
10 core2.mci.net (204.70.4.205) 365 ms 374 ms 390 ms
11 core3.mci.net (204.70.1.93) 383 ms 473 ms 397 ms
12 * border4.Chicago.mci.net (204.70.3.83) 390 ms *
13 star.mci.net (204.70.27.6) 420 ms 445 ms 411 ms
14 * wuarchive.wustl.edu (128.252.135.4) 428 ms *
```

Da saída do comando acima pode-se identificar todo o caminho percorrido até se chegar ao computador destino. No total, a mensagem passa por 13 computadores até chegar ao destino.

Ao lado do nome de cada computador pode-se ver o número IP e três valores em milissegundos. A cada um destes computadores são enviados três pacotes UDP e, para cada um destes pacotes, é medido o tempo de ida e volta do pacote. Se não houver resposta dentro de três segundos, no lugar onde seria exibido o tempo da viagem de ida e volta é colocado um asterisco, como se pode ver acima.

O objetivo deste comando é servir como uma ferramenta para identificação de problemas de rede, roteamento e medição de performance. Se o pacote estiver tomando caminhos totalmente diferentes da melhor rota esta anomalia já pode ser identificada a partir da saída do *traceroute*. Pode-se também se identificar gargalos, a partir dos quais a performance se torna extremamente lenta.

13.72 wget

13.72.1 Download de Páginas ou Arquivos na Web

O comando *wget* nos permite realizar o download de páginas Web e nos oferece diversas facilidades interessantes.

A mais útil é a possibilidade de retomar um download interrompido. Para baixar arquivos grandes, como por exemplo o StarOffice, que tem mais de 60MB, este recurso é realmente fundamental.

Para isto, basta invocar o comando *wget* com a opção *-t0*:

```
wget -t0 ftp://ftp.dominio.com/staroffice.tgz
```

Um outro recurso interessante nos permite realizar o download de documentos web. Muitos sítios não oferecem a possibilidade de se baixar o documento inteiro e nos forçam a navegar por uma infinidade de páginas para obter a informação que queremos. Com o *wget* podemos baixar o documento inteiro da seguinte forma:

```
wget -t0 --no-parent http://www.dominio.com/index.html
```

A opção "`-no-parent`" impede que durante o download o programa saia da página que se pretende baixar, por meio de um link para, por exemplo, a página principal do sítio, e vá para outros locais.

13.72.2 Espelhamento de Diretórios

Existem hoje na Internet diversos sítios que disponibilizam livros, principalmente sobre informática, no formato HTML. Embora este acesso seja uma grande conveniência, que nos permite avaliar com calma se a decisão de compra é acertada ou não, estes livros são geralmente dispostos em um número sem fim de páginas. A leitura é praticamente impossível a não ser que seja feita em frente ao computador, algo que realmente não é muito confortável.

O comando *wget* pode ser usado para espelhamente de um sítio completo, ou parte dele. O comando:

```
$ wget -nH -r -l2 http://www.dominio.com/dir/outrodir/
```

As opções ao comando *wget* indicam para não acessar outros computadores (*-nH*), utilizar o modo recursivo (*-r*) e não descer mais de dois níveis abaixo do nível inicial *-l2*.

Procedendo desta forma evitamos que o programa *wget* se desvie de seu alvo inicial e acabe trazendo para o seu computador centenas de páginas indesejadas.

13.73 Configuração do prompt

Para facilitar a identificação do diretório onde nos encontramos, o prompt pode ser configurado para sempre exibir o diretório corrente.

Os comandos variam de acordo com a shell utilizada. No caso da *cshell*, por exemplo, isto pode ser conseguido acrescentando-se as seguintes linhas ao seu arquivo *.cshrc*:

```
set prompt="[$cwd] "
alias cd 'cd \!* ; set prompt="[$cwd] "'
```

Esta sequencia de comandos fará com que o prompt apareça da seguinte forma:

```
[/var] cd /usr/include
[/usr/include] cd ~
[/home/queiroz]
```

Existe uma forma mais complicada de se configurar o prompt que substitui o seu diretório pessoal, (*/home/usuario*, por exemplo) por "~ " (til). Se as seguintes linhas forem incluídas no arquivo *.cshrc*:

```
set prompt="[`echo $cwd | sed s@$HOME@\~@`] "
alias cd 'cd \!* ; set prompt="[`echo $cwd | sed s@$HOME@\~@`] "'
```

Desta forma o seu prompt ficará da seguinte forma:

```
[~] cd /usr/include
[/usr/include] cd ~/bin
[~/bin]
```

O seu diretório de trabalho será sempre substituído por "~" sempre que aparecer no diretório corrente. Desta forma, */home/usuario/src*, aparecerá como *[~/src]* e assim por diante.

Importante, deve haver um espaço entre o asterisco e o ponto e vírgula na definição do alias do comando *cd*.

13.74 wc – Word Count

O rodapé das mensagens da lista Dicas-L contém um grande número de informações adicionais para comodidade de seus assinantes. Dentre estas informações encontra-se um indicativo do número de assinantes da lista.

Esta informação é obtida por meio do comando *wc*.

```
#!/bin/bash
cat << EOF > tail.txt
-----------------------------------------------
As mensagens da lista Dicas-L são
veiculadas diariamente para
‘wc -l dicas-l|awk ’{print $1}’‘ assinantes.

As mensagens são enviadas única e exclusivamente
para seus assinantes.
 ...e por aí vai....
EOF
```

A opção *wc -l* exibe o número de linhas do arquivo *dicas-l*. Como eu tenho um assinante por linha, o comando *wc*, com a opção "**-l** " retorna o número de assinantes da lista. Só que juntamente com o número de linhas do arquivo *dicas-l* vem também o nome do arquivo. Como estamos interessados apenas no primeiro valor, o número de linhas, utilizamos o comando *awk* para selecionar o que interessa, o primeiro campo (*print $1*).

Como não podia deixar de ser, é possível realizar esta mesma operação utilizando apenas o comando *awk*.

```
‘awk ’END{print NR}’ dicas-l‘
```

O comando *awk*, ao finalizar o processamento do arquivo *dicas-l*, indicado pela diretiva *END*, imprime o valor *NR* (*number of records*).

13.75 WvDial – Conexões PPP

A configuração de um serviço de acesso discado à Internet utilizando sistemas Unix envolve a configuração do PPP (*Point to Point Protocol*). Esta configuração envolve diversos passos e em certas situações pode ser um pouco complexa e passível de erros.

Existem vários programas disponíveis para se fazer a configuração do serviço PPP automaticamente. Uma destas opções é o programa *Wvdial.* Este programa automatiza todos os passos necessários ao estabelecimento da conexão com seu provedor.

A vantagem maior do programa *wvdial* é que tudo é feito automaticamente: o reconhecimento do modem, a discagem do número telefônico, fornecimento do nome do usuário e senha.

13.76 Arquivos setuid/setgid

Programas com *setuid* (*Set User ID*) bit são um mal necessário. Um mal porque se codificados erradamente podem ser utilizados para atacar um sistema e necessários porque muitas funções do sistema operacional dependem de sua existência para o seu funcionamento.

Um exemplo bastante claro é o arquivo *passwd.* Este arquivo não pode ser editado por usuários comuns, porque desta forma a criação de contas ficaria aberta a todos.

O arquivo */etc/passwd* possui as seguintes permissões:

```
$ ls -l /etc/passwd
-rw-r--r--   1 root  root   666 Oct 24 10:08 /etc/passwd
```

Apenas o usuário *root* pode realizar alterações neste arquivo. Os demais usuários podem apenas ler as informações nele contidas.

Os usuários comuns não podem criar entradas mas devem poder alterar suas senhas. Como resolver este impasse? Através de programas *setuid*. O programa *passwd* por sua vez possui as seguintes permissões:

```
$ ls -l /usr/bin/passwd
-r-s--x--x   1 root  root  11056 Jun 30 17:10 /usr/bin/passwd
```

O caractere *s* em *r-s* significa que todos os que executam este programa o executam como o usuário proprietário do arquivo, neste caso, *root*.

Desta forma, quando um usuário digita *passwd* para trocar sua senha, ele vai conseguir gravar a nova senha no arquivo */etc/passwd* independentemente do fato deste arquivo não possuir permissão universal de gravação.

Da explicação anterior pode-se entrever o perigo que programas com o *setuid bit* ligado podem representar para o sistema. Muitos administradores de sistemas criam programas com o *setuid bit* ligado para desempenhar funções privilegiadas em seus sistemas. A norma geral a ser seguida é criar o mínimo possível de programas com estas características, e caso realmente necessário, examinar e testar o código exaustivamente para evitar possíveis erros que possam ser explorados por usuários mal intencionados.

Além disto tudo, o administrador de sistemas tem que controlar o número de programas que possuem o setuid bit ligado em seu sistema.

O comando

```
# find / -type f -a \( -perm 0400 -o -perm 0200 \) -print
```

irá localizar todos os arquivos no sistema que possuam o *setuid* ou *setgid* (*Set Group ID*) bit ligados. O *setgid bit*, possui funcionalidade idêntica ao *setuid bit*, mas atua sobre as permissões do grupo ao qual pertence o programa.

A listagem gerada deverá então ser cuidadosamente examinada para detectar possíveis anormalidades e identificar as providências a serem tomadas.

13.76.1 Controle de laço em shells

Uma facilidade bastante poderosa oferecida pelas shells de sistemas Unix em geral é o controle de laço. Por meio delas pode-se executar rapidamente tarefas bastante complexas.

A sintaxe é bastante simples e consiste basicamente na seguinte construção

```
$ for variavel in a b c d ...
> do
> comandos
> ...
> done
```

Esta construção é válida para *bash* e similares (*ksh* e *sh* por exemplo). Na primeira linha define-se a variável e os valores que irá assumir durante a execução. A diretiva *do* indica o início dos comandos a serem executados até que se esgotem os valores atribuídos à variável. E a diretiva *done* sinaliza o fim da execução. O mais interessante é que a variável pode assumir os valores gerados por um comando do Linux. Para isto basta delimitar o comando entre aspas invertidas (‘).

Todos os exemplos aqui citados são invocados a partir da linha de comandos. Ao se teclar ENTER após a digitação da linha contendo a diretiva *for* o prompt se modifica para o caracter $>$. Ao final do laço, indicado pela diretiva *done*, os comandos são então executados. Nada impede entretanto que estes comandos sejam gravados em um arquivo e executados como uma shell script tradicional.

O objetivo é justamente ensinar estruturas simples para serem utilizadas diretamente da linha de comandos.

Os exemplos abaixo irão ajudar a esclarecer a utilização desta estrutura.

- Em um diretório existem vários arquivos que possuem letras maiúsculas em seu nome. Renomea-los de forma a que em seu nome existam

apenas letras minúsculas:

```
$ for arquivo in `ls`
> do
> mv $arquivo `echo $arquivo | dd conv=lcase`
> echo $arquivo
> done
```

Neste exemplo a variável *arquivo* irá assumir os valores retornados pelo comando *ls*. O comando *mv* irá renomear os arquivos do diretório corrente para o valor retornado pela saída do comando *echo* processada pelo comando *dd*. A opção *conv=lcase* do comando *dd* faz uma tradução dos caracteres maiúsculos para minúsculos. O comando *echo $arquivo* serve apenas para indicar o progresso da execução do comando, ecoando para a tela o nome do arquivo sendo processado no momento.

- Gerar as versões formatadas de vários arquivos que estão no formato *nroff*. Cada documento formatado deverá possuir a terminação *.doc*.

```
$ for arquivo in *
> do
> nroff -man $arquivo | col -b |`echo $arquivo |
> awk -F. '{print $1}'`.doc
> echo $arquivo
> done
```

Neste exemplo, foi invocado o comando *nroff* para processar os arquivos. A saída deste comando é então redirecionada para o comando *col* que irá retirar os sublinhados e caracteres que não são entendidos corretamente por algumas impressoras. O resultado destes dois comandos é gravado em um arquivo cujo nome é formado com o auxílio do comando *awk*, que irá obter do nome original o primeiro qualificador e acrescentar o sufixo *doc*. Por exemplo, o nome *ls.1* seria substituído pelo nome *ls.doc* e assim sucessivamente.

- Enviar a mensagem gravada no arquivo de nome *mensagem* para os endereços eletrônicos gravados no arquivo de nome *usuarios*.

```
$ for u in `cat usuarios`
> do
> mail -s "Convocacao para reuniao semestral" \
        $u < mensagem
> echo $u
> done
```

13.77 X Window – Fontes True Type

A partir da versão 6.0 do RedHat Linux, o suporte a fontes True Type, usadas em sistemas Windows, foi incorporado ao sistema. O pacote *FreeType* foi integrado ao X juntamente com o pacote *xfsft*. Desta forma, o suporte a fontes True Type é incorporado sem gerar incompatibilidades com o sistema de fontes do X11. Versões *Conectiva Linux* superiores a 4.0 do também vêm com suporte a fontes True Type incorporado ao servidor X.

Para utilizar então os fontes do Windows em Linux, é necessário primeiramente obter-se os fontes. No sítio da Microsoft encontram-se disponíveis vários fontes que podem ser baixados gratuitamente[2]. Além dos fontes proprietários é possível se obter um grande número de fontes gratuitos na Internet[3]

Todas aplicações que você instala em sistemas Windows e o próprio Windows vêm com uma grande quantidade de fontes. No Windows os fontes normalmente ficam no diretório *\windows\fonts*. Os fontes são os arquivos terminados em *ttf* (*True Type Fonts*). Caso o Windows tenha sido substituído em seu micro pelo Linux a licença de uso do Windows adquirida junto com o seu computador lhe dá o direito de usar as fontes, e o mesmo se aplica às fontes dos aplicativos Windows que tenha adquirido.

[2] *http://www.darmstadt.gmd.de/~pommnitz/ttfonts.html#MSFONTS*
[3] *http://www.mediabuilder.com*

Uma vez de posse destas fontes, você deve então transferi-las para seu computador com *Conectiva Linux*. Crie um diretório para abrigar estas fontes *TrueType*:

```
# mkdir /usr/X11R6/lib/fonts/truetype
```

Copie em seguida as fontes do Windows para este diretório. Neste ponto você irá necessitar de um outro programa para criar o arquivo *fonts.scale* que irá ficar no diretório das fontes *True Type*. Isto é feito com o programa *ttmkfdir*[4]. Uma vez instalado este programa, vá ao diretório criado e execute o comando

```
# ttmkfdir -o fonts.scale
```

Neste mesmo diretório crie o arquivo *fonts.dir*:

```
# mkfontdir
```

No sítio da RedHat encontra-se um excelente documento[5] descrevendo todo este processo em detalhes. Uma ótima leitura.

E isto é tudo. Reinicialize o ambiente X e chame um sítio que use fontes mais sofisticadas como *http://www.abcnews.com* e veja que maravilha. Para versões anteriores do RedHat o processo é um pouco mais complicado visto que você tem que substituir o *X Font Server* original, *xfs*. Mas na minha opinião, vale mais a pena fazer uma atualização para a versão mais recente do *Conectiva Linux* e aplicativos. Os benefícios são bem maiores.

13.78 xlock

É possível habilitar o programa *xlock* para que também o usuário *root* consiga destravar a estacão de trabalho.

[4] Integrante do pacote *freetype*

[5] *http://www.redhat.com/knowledgebase/newfontsystem/index.html*

Desta forma, caso se deseje usar uma estação cujo dono esteja ausente, o administrador, com a senha do superusuário, conseguirá destravá-la.

Isto pode ser conseguido inserindo-se a linha

```
xlock*allowroot:  true
```

no arquivo *.Xdefaults*.

13.79 xargs

Os resultados obtidos pelo comando *find* podem ser redirecionados para o comando *xargs* para que sejam tomadas ações específicas (remoção, mudança de atributos, listagem, etc) sobre estes arquivos.

O comando *xargs* atua sobre o resultado a ele fornecido pela saída padrão e constrói um comando baseado nesta entrada e em seus próprios parâmetros da linha de comando.

Por exemplo:

```
$ ls | xargs rm -f
```

Este comando irá remover todos os arquivos do diretório corrente. A diferença entre o comando acima e os comandos

```
$ rm -f `ls`
```

e

```
$ rm -f *
```

é que após a shell realizar a expansão dos nomes gerados através do "`rm -f *`" ou "`rm -f `ls``" o tamanho da linha de comandos pode causar um erro.

O comando *xargs* gera linhas de comando de tamanho compatível com as limitações do sistema e executa o comando solicitado tantas vezes quantas forem necessárias para completar a tarefa.

13.80 xterm – X Terminal

O fonte padrão utilizado para o programa xterm geralmente é muito pequeno, quase ilegível. Para aumentar o tamanho do fonte, posicionar o mouse sobre a janela do xterm e pressionar o botão direito do mouse juntamente com a tecla [CTRL]. Aparecerá então um menu onde serão apresentadas diversas opções para redimensionamento do tamanho da fonte utilizada. Eu prefiro o maior valor, *HUGE*.

Outra opção é invocar o xterm já com seu fonte preferido. Para isto digitar:

```
$ xterm -fn 12x24 &
```

O sinal *-fn* significa *font name*. O fonte 12x24 é o meu favorito. É bastante grande e legível. Existem centenas de fontes disponíveis. Para ver a lista completa utilize o comando *xlsfonts*.

13.81 xwpick – Captura de Telas

Um utilitário excelente para se gravar telas ou porções de um ambiente desktop é o programa *xwpick*[6].

Este utilitário permite que se grave arquivos em vários formatos e é extremamente fácil de se usar. Os formatos suportados são *PostScript*, *Encapsulated PostScript*, Gif, PCX e PICT e PPM.

[6] *ftp://sunsite.unc.edu/pub/Linux/X11/xutils/*

Capítulo 14

Conectiva Linux – Descrição dos Aplicativos

Freqüentemente se menciona o fato de que sistemas Linux possuem poucas aplicações, o que está longe de ser uma verdade. Possivelmente as aplicações para sistemas Linux ainda não tenham atingido, na computação pessoal principalmente, o mesmo nível de qualidade de seus equivalentes comerciais em ambiente Windows, porém isto está mudando rapidamente. As alternativas livres e comerciais para sistemas Linux não param de crescer. Diversas empresas têm manifestado nos últimos tempos o seu crescente apoio ao Linux, como Corel, Adobe e IBM, apenas para citar algumas.

No tocante a aplicações servidoras, a superioridade dos aplicativos nativos do ambiente Unix e hoje disponíveis em todos sistemas baseados no Linux, já há vários anos funcionam irrepreensivelmente e constituem a base do maior empreendimento em computação existente, a Internet.

Para a computação pessoal as perspectivas são animadoras. Além das iniciativas públicas e gratuitas como KDE (*K Desktop Environment*) e GNOME (*GNU Object Model Environment*), e seus aplicativos de produtividade, temos também tradicionais desenvolvedores do sistema Windows portando seus produtos para o ambiente Linux. Da parte da Corel temos o Corel

Wordperfect, Corel Photopaint, em breve teremos também o CorelDraw. A Adobe já disponibilizou gratuitamente na Internet a versão beta do amplamente utilizado Framemaker. As novidades são bastante numerosas.

Fora estes exemplos mais marcantes vale a pena citar que a maioria das distribuições Linux trazem centenas de aplicações gratuitas que podem ser instaladas por seus usuários sem custo algum. Os dois CDROMs de distribuição do *Conectiva Linux*, versão 4.9 (beta), trazem 1001 aplicativos, gravados no diretório *RPMS*. Este número entretanto é apenas uma pequena amostra do que está disponível. Estão cadastrados atualmente mais de 46.000 aplicativos e este número certamente não representa a totalidade de aplicações existentes.

O maior problema com esta imensidão de aplicativos é justamente saber o que cada um deles faz. No primeiro cdrom da distribuição Conectiva Linux, no diretório doc, o arquivo PACOTES contém uma descrição de todos os pacotes. Este arquivo tem 7845 linhas e certamente é bastante grande.

Todavia existe uma maneira de se consultar, com o software RPM, o gerenciador de pacotes criado pela empresa Red Hat usado em várias distribuições Linux, a descrição de um pacote. O comando

```
% rpm -qpi xzip-180-4cl.i386.rpm
Name        : xzip               Relocations: (not relocateable)
Version     : 180                Vendor: conectiva
Release     : 4cl                Build Date: qua 12 jan 2000 15:06:29 BRDT
Install date: (not installed)    Build Host: mapinguari.conectiva.com.br
Group       : Passatempos/Jogos  Source RPM: xzip-180-4cl.src.rpm
Size        : 108406             License: Freely redistributable
Summary     : Interpretador X Window para os jogos adventure
              no formato Infocom
Description :
Agora todos os seus jogos tipo "adventure" em texto
podem adquirir uma nova dimensão com este interpretador
para X Window.
```

nos fornece informações detalhadas sobre o pacote *xzip*. Ao final encontra-se a descrição, que na maior parte dos casos é o que buscamos. Observe que fornecemos como entrada para o comando o nome do arquivo no formato

rpm, sem tê-lo instalado. Esta facilidade é extremamente interessante visto que nos permite verificar previamente se determinado software atende às nossas necessidades.

Eu particularmente gosto de ler descrições de softwares para ficar a par do que existe em termos de aplicações, preferencialmente grátis, para o ambiente Linux. Desta forma quando a necessidade surgir eu tenho condições de saber se existe algo que possa resolver o problema. Eu também prefiro ler documentos formatados, de forma a tornar a leitura mais agradável e também onde eu possa acessar a informação que procuro de forma mais rápida.

Pensando nisto, com a ajuda do programa *rpm*, eu criei uma documentação, com o uso do programa LaTeX, contendo a descrição de todos os softwares distribuídos com a versão 4.9 do Conectiva Linux.

No CDROM distribuído com este livro encontra-se o documento gerado, nos formatos PDF e PostScript, contendo uma descrição resumida dos pacotes de software integrantes da versão 4.9 do Conectiva Linux. Esta descrição foi obtida através da execução, no diretório RPMS dos dois cdroms da distribuição, do comando '*rpm -qpi*', sobre todos os arquivos. A listagem gerada foi simplificada para conter apenas o nome do pacote e sua descrição.

A shell script utilizada foi a seguinte:

geradoc.sh

```
#!/bin/bash
mount /mnt/cdrom
cd /mnt/cdrom/conectiva/RPMS
mkdir /root/pkg
for pkg in *.rpm
  do
    rpm -qpi $pkg > /root/pkg/$pkg
    echo $pkg
  done
```

```
cd /root/pkg

for file in [A-Z]*
  do
    mv $file ‘echo $file | dd conv=lcase‘
    cat * >> ../cl-desc_software.tex
  done
```

Como diversos pacotes são grafados em maiúsculas, eu renomeei todos os arquivos de forma a que contivessem apenas letras minúsculas (comando *mv*). Desta forma, o comando *cat* irá criar o arquivo *cl-desc_software.tex* com todas as descrições em ordem alfabética.

Note que esta listagem não divide os aplicativos em categorias, o que poderia ser outra abordagem interessante.

Em seguida foi editado o arquivo *cl-desc_software.tex* para remover as informações não necessárias e criado, com o uso do software LaTeX, os arquivo formatados (*cl-desc_software.pdf* e *cl-desc_software.ps)*.

Espero que este documento lhe seja útil e o ajude a conhecer melhor o potencial de sistemas Linux. Sem dúvida alguma, nenhum sistema operacional traz consigo um número tão grande de aplicações gratuitas e de qualidade.

Uma versão mais completa deste está disponível no CDROM, no diretório */home/aplicativos_conectiva_linux* nos formatos PostScript e PDF.

Índice Remissivo

aliases, 153
alien
    formatos de pacotes, conversão, 156
apropos, 157
arquivos
    divisão, 269
    permissões, 227
    remoção
      nomes estranhos, 248
    tipos, 158
    transferência
      ncftp, 219
awk, 160
    campos, número, 161
    impressão de campos selecionados, 160
    manual, 162
    NF, variável, 161

backup
    amanda, 166
    arquivos, gravação em, 174
    email, 167
    estratégias, 164, 165
    identificação, 173
    recomendações, 162
basename, 170
bash
    comandos
      histórico, 170
    HISTSIZE, 170
boot
    mensagens, 171
bzip2
    arquivos, compactação, 162

cat, 178
    linhas em branco,eliminando, 178
    linhas, numeração, 178
cd, 175
    diretório anterior,retornar, 175
chmod, 224
comandos
    diretórios
      preenchimento automático, 231
    encadeamento, 186
compressão
    gzip, 197
conexão discada
    configuração, 280
correio eletrônico

auto-resposta, 55
cp, 176, 190
arquivos, cópia rápida, 176
cron, 176
cshell
saída padrão, redirecionamento, 176
csplit
arquivos, divisão, 179

date, 182
daytime, 183
dd, 184
deroff, 184
diretórios
proteção, 247
dirname, 170
dns
computadores, nomes, 196
Domain Name Service, 41
documentação
manual, páginas, 214
dump
arquivos remotos, 169
dutos, 184

echo
ls, substituição, 185

find
arquivos sem dono, localizando, 188
arquivos, procurando, 189
diretórios, cópia, 190
espaço em disco, gerenciamento, 195
fontes
true type, 284
formail
reenvio de mensagens, 108
ftp, 79, 191
arquivos, manipulação direta, 193
automatização de sessões, 191
múltiplos arquivos, transferência, 193
sessões, Netscape, 190
transferência de diretórios, 194

gopher, 52
grep, 197
cadeia de caracteres,pesquisa, 197
múltiplos parâmetros, 197

ht://dig, 14
HTTP, Hyper Text Transport Protocol, 52

ifconfig, 218
inetd.conf
serviços, remoção, 202
ingles
leitura, 59
internet
histórico, 51
roteamento, 275
IP
endereçamento

classes reservadas, 186
endereçamento,múltiplos endereços, 48

lilo, 198
linux, WindowsNT, 79
logbook.sh
registro de atividades, 203
ls, 207
arquivos, listagem seletiva, 213
listagem i-nodes, 248
listagem ordenada, 210
permissões de diretórios, listagem, 210
uso, opções, 207
lsof
list open files, 201

man, 214
memória
informações, 214
mkdir, 216
criação de uma árvore completa de diretórios, 216
mkpasswd, 255
more, 184, 218
arquivos, visualização, 218
Mosaic, 52

ncftp, 219
Netscape
opções, 75

partições
redimensionamento, 222
partições windows
acesso, 221
passwd, 280
paste, 259
pine
homepage, 109
ppp
wvdial, 280
processos
suspensão, 270
procmail
postagem cruzada, evitando, 109
programação shell, 232
acentuação
remoção, 236
bit de execução, ativação, 235
controle de laço, 282
parâmetros, 240
verificação, 238
shell scripts
depuração, 244
variáveis
$*, 241
$0, 240
$?, 242
$#, 241
$$, 243
atribuição de valores, 239

rm, 247, 248
remoção de arquivos iniciados em -, 248
root

senha
recuperação, 198

sed, 249
caracteres delimitadores, 253
linhas
impressão seletiva, 254
linhas, deleção, 252
substituição global, 249
segurança
nessus, 219
sendmail, 83
autorização para relay, 100
BCC, Blind Carbon Copy, 92
envio de mensagens diretamente, 103
expn, 86
interação com DNS, 93
mailcheck.sh, 88
mensagens,manuseio, 106
processamento seletivo da fila, 87
redirecionamento de mensagens, 83
relays, 99
user database, 96
verificação de endereços eletrônicos, 105
versão, determinando a, 107
vrfy, 86
senhas
geração automática, 255
seguras, 257
setgid, 227, 280
setuid, 227, 280
shell
prompt, configuração, 277
shell scripts
geração de números, 234
sistema
acesso, restrição, 221
sistema de arquivos
desmontar, 201
sistema operacional
modos de permissão, 222
slice, 106, 258
sort, 184
split, 268, 269
divisão por bytes, linhas, 268
ssh
cliente
MindTerm, 217
sticky, 227
su, 269, 270
substitute user, 269

tail, 271
tar
arquivos, cópia, 172
tee, 272
tr
conversão, maiúsculas, minúsculas, 273
traceroute, 275

unalias, 155
useradd, 255

variáveis de ambiente

MANPATH, 216
PATH, 232
vi, 127
abreviações, 137
alterações, desfazendo, 143
busca e substituição, 147
caracteres de controle, 129
comandos do sistema, inclusão, 131
criador, 127
deleção de caracteres, 147
deleção de colunas, 134
deleção de linhas em branco, 132
gravação seletiva, 134
ignorecase, 130
inserção de linhas, 135
introdução, 127
inversão das linhas, 133
linha, quebra, 138
múltiplos arquivos, edição, 148
opção magic, 150
opção nomagic, 150
opções de gravação, 145
posicionamento automático, 137
referências, 151
releitura de arquivos, 143
remoção de espaços em branco, 146
substituição com metacaracteres, 133
substituição de caracteres, 128
substituições, 142, 144
substituições em intervalos de linhas, 144
teclas, mapeamento, 141
undo, 143
vim-common, 152

wc
word count, 279
wget, 79, 276
diretórios,espelhamento, 277
páginas web, download, 276

xargs, 286
xlock, 285
xterm
tamanho do fonte, 287
xwpick
telas, captura, 287