

Linux

Comandos Básicos e Avançados

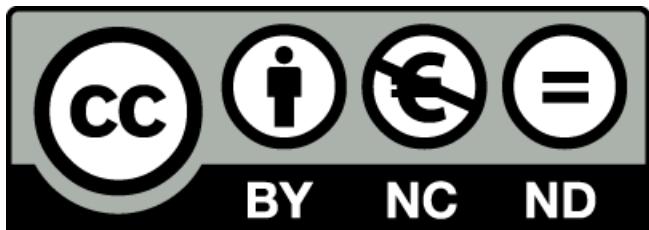
**Alessandro Vivas Andrade
Leonardo Carneiro de Araújo
Cristiano Grijó Pitangui
Luciana Pereira de Assis**

**Diamantina, Minas Gerais
20 de novembro de 2015**

Alessandro Vivas Andrade, Leonardo Carneiro Araújo, Cristiano Grijó Pitangui,
Luciana Pereira de Assis

LINUX: COMANDOS BÁSICOS E AVANÇADOS

Diamantina
2015



EDITOR: Alessandro Vivas Andrade

PROJETO GRÁFICO: Alessandro Vivas Andrade

CAPA: Produzida por Alessandro Vivas Andrade com foto de autoria de Alessandro Vivas Andrade

Dados Internacionais de Catalogação na Publicação (CIP)

(eDOC BRASIL, Belo Horizonte/MG)

A5531

Andrade, Alessandro Vivas.

Linux [recurso eletrônico]: comandos básicos e avançados /
Alessandro Vivas Andrade... [et al.]. – Diamantinha (MG): A. V.
Andrade, 2015.

141 p. : il. ; PDF

Inclui bibliografia

ISBN 978-85-920329-0-6

Requisitos de sistema: Adobe Acrobat Reader

1. Linux (Sistema operacional de computador). 2. Sistemas
operacionais (Computadores). I. Araújo, Leonardo Carneiro. II.
Pitangui, Cristiano Grijó. III. Assis, Luciana Pereira de. IV. Título.

CDD-005.43

Agência Brasileira do ISBN

ISBN 978-85-920329-0-6

9 788592 032906

Prefácio

Este livro tem como objetivo apresentar de uma maneira simples e didática os principais comandos do **shell** do Sistema Operacional Linux. Nele são abordados a maioria dos comandos disponíveis pelo Sistema Operacional Linux onde grande parte destes são compatíveis com o Sistema Operacional MacOs. Reunimos profissionais com formações experiências e formações distintas para apresentar visões diferentes deste mesmo tema.

A motivação de escrever este livro surgiu a partir de três realidades distintas, a primeira de reunir em um único material um conteúdo para auxiliar alunos dos cursos de Computação e Engenharia que trabalham com o Sistema Operacional Linux. A segunda surgiu de apresentar conceitos úteis para os administradores de sistemas Linux. A terceira motivação foi de reunir em um único material comandos úteis para os pesquisadores de todas as áreas que trabalham com o Linux.

Este livro pode ser utilizado como bibliografia principal ou complementar em cursos de Ciência da Computação ou Sistemas de Informação para as disciplinas de Sistemas Operacionais e Redes de Computadores.

Encontrou algum erro no livro ou tem alguma sugestão? Favor encaminhar e-mail para
alessandro.vivas@gmail.com.

Sobre os Autores



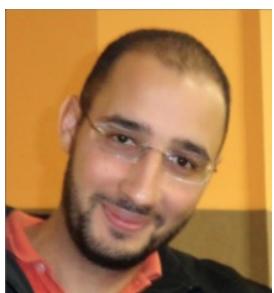
Alessandro Vivas Andrade é natural de Lavras/MG e atualmente reside em Diamantina/MG. É professor do Curso de Sistemas de Informação da Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM) onde leciona as disciplinas de Sistemas Operacionais, Redes de Computadores e Sistemas Distribuídos. Também atua como Professor do Curso de Mestrado em Gestão em Instituições de Ensino (PPGGIED/UFVJM). Graduou-se em Engenharia Elétrica (UFMG) e depois cursou Mestrado e Doutorado ambos em Engenharia Elétrica na UFMG. Tem interesses nas áreas de Otimização, Inteligência Artificial e Redes de Computadores.



Luciana Pereira de Assis é natural de Belo Horizonte/MG e atualmente reside em Diamantina/MG. É professora do Curso de Sistemas de Informação da Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM) onde leciona as disciplinas de Algoritmos e Estrutura de Dados, Pesquisa Operacional e Inteligência Artificial. Também atua como Professor do Curso de Mestrado em Gestão em Instituições de Ensino (PPGGIED/UFVJM). Graduou-se em Ciência da Computação (UNIBH) e depois cursou Mestrado em Ciência da Computação (UFMG) e Doutorado em Engenharia Elétrica na UFMG. Tem interesses nas áreas de Otimização, Inteligência Artificial e Análise de Redes Sociais.



Leonardo Carneiro de Araújo é natural de Belo Horizonte/MG. É professor da Universidade Federal de São João del Rei (UFSJ) onde leciona as disciplinas de Teoria da Informação, Processamento de Áudio e Vídeo, Análise de Sinais e Sistemas, dentre outras. Graduou-se em Engenharia Elétrica (UFMG), depois cursou Mestrado e Doutorado em Engenharia Elétrica (UFMG). Tem interesse nas áreas de Teoria da Informação, Linguística e Linguística Quantitativa, Reconhecimento de Fala, Reconhecimento de Padrões e Inteligência Artificial.



Cristiano Grijó Pitangui é natural de Ouro Branco/MG e atualmente reside em Diamantina/MG. É professor do Curso de Sistemas de Informação da Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM) onde leciona as disciplinas de Teoria da Computação, Linguagens de Programação e Inteligência Artificial. Também atua como Professor do Curso de Mestrado em Gestão em Instituições de Ensino (PPGGIED/UFVJM). Graduou-se em Ciência da Computação (UFJF) e depois cursou Mestrado e Doutorado em Engenharia de Sistemas e Computação com ênfase em Inteligência Artificial na COPPE-UFRJ. Tem interesses nas áreas de Aprendizado de Máquina, Inteligência Artificial e Lógica de Primeira Ordem.

Conteúdo

1	Conceitos Básicos	1
1.1	Instalação do Sistema Operacional Linux	1
1.2	Qual Distribuição?	1
1.3	Acessando o Sistema Operacional Linux	1
1.4	Acessando o Terminal do Linux	1
1.5	Entrando no Sistema	2
1.6	Significado do Shell	2
1.7	Formato dos comandos	3
1.8	Shells	4
1.8.1	Descobrindo o Shell	4
1.9	Case Sensitive	5
1.10	Movimentação no terminal	5
1.11	Primeiros comandos	5
1.12	Visualizando textos longos no terminal	6
1.13	Exibindo Mensagens	6
1.14	Histórico do Terminal	6
1.14.1	Comando <i>history</i>	7
2	Ligando e Desligando o Linux	9
2.1	Saindo do sistema	9
2.1.1	Saindo do Sistema com Logout	9
2.2	Saindo dom Sistema com Exit	9
2.3	Desligando e Reinicializando o Sistema	9
2.3.1	Desligando Imediatamente	10
2.3.2	Desligando após um determinado tempo	10
2.3.3	Desligando em uma hora específica	10
2.3.4	Cancelando um shutdown	11
2.4	Reinicializado a máquina	11
2.4.1	Reinicializando após determinado tempo	11
2.4.2	Reinicializando em uma determinada hora	11
3	Operações em Diretórios e Arquivos	13
3.1	Árvore de Diretórios	13
3.2	Estrutura de Diretórios do Sistema Linux	13
3.3	Listando o Conteúdo do diretório	14
3.4	Listando uma única entrada por linha	14
3.5	Listando o Conteúdo no Formato Longo	15
3.6	Informações sobre os arquivos e diretórios	15
3.7	Obtendo informações sobre diretórios	16

3.8	Listando Arquivos Ocultos	16
3.9	Classificando Arquivos e Diretórios	16
3.10	Imprimindo Informações sobre o Tamanho dos arquivos.	17
3.11	Listando Recursivamente	17
3.12	Navegando em Diretórios	18
3.13	Comando pwd	19
3.14	Copiando Arquivos	19
3.14.1	Copiando Arquivo para Diretório	19
3.15	Copiando Múltiplos arquivos	20
3.16	Copiando Diretórios e Sub-diretórios	20
3.17	Renomeando Arquivos	20
3.18	Criando um Arquivo Vazio com touch	21
3.19	Apagando Arquivos	21
3.19.1	Apagando Múltiplos Arquivos	21
3.20	Apagando um Diretório	21
3.21	Apagando Diretório com rmdir	21
3.22	Nomes de arquivos	22
3.22.1	Barra invertida	22
3.23	Criando Diretório	22
3.24	Criando Múltiplos Diretórios	23
3.25	Criar Hierarquia de Diretórios	23
3.26	Links	23
3.26.1	Hard Links	24
4	Comandos para Manipulação de Arquivos Texto	27
4.1	Comando cut	27
4.2	Comando expand	27
4.3	Comando tr	28
4.4	Comando fmt	30
4.5	Comando grep	30
4.6	Comando head	30
4.7	Comando iconv	31
4.8	Comando look	31
4.9	Comando more	31
4.10	Contar Número de Linhas - Comando nl	32
4.11	Comando paste	32
4.12	Comando rev	33
4.13	Comando sort	33
4.14	Comando tail	34
4.15	Comando uniq	34
4.16	Contar Número de Caracteres - Comando wc	35
5	Comandos de Sistema	37
5.1	Gerando Todos os Comandos	37
5.2	Quem sou eu e onde estou?	37
5.2.1	Arquivo passwd	38
5.3	Comando id	39
5.4	Alterando a Senha	39
5.4.1	Usuários Logados	40
5.5	Comando Date	40

5.6	Comando cal	40
5.7	Comando finger	40
5.8	Comando free	41
5.9	Comando su	42
5.10	Comando uname	42
5.11	Comando uptime	43
5.12	Verificando a versão de um comando	43
5.13	Variável PATH	44
5.14	Comando w	44
5.15	Comando whereis	45
5.16	Comando locate	45
5.17	Comando which	45
5.18	Comando whatis	45
5.19	Rodando múltiplos comandos	46
5.20	Rodando um comando em background	47
6	Gerenciamento de Processos	49
6.1	Visualizando Todos os Processos em Execução	49
6.2	Todos os Processos de um Usuário Específico	49
6.3	Lista de Processos Ordenadas pelo Consumo de CPU	49
6.4	Lista dos Processos que Mais Consomem Memória	50
6.5	Obtendo Informações de um Processo Específico	50
6.6	Comando pstree	50
6.7	Comando top	51
6.8	Listando todos os Sinais com o Comando kill	51
6.9	Matando um Processo com o Comando Kill	52
6.10	Comando time	52
6.11	Controlando Processos	53
7	Permissão e Propriedade	63
7.1	Permissão e Propriedade	63
8	Gerenciando Usuários	65
8.1	Listando Todos os Usuários do Sistema	65
8.2	Listando Grupos	65
8.3	Adicionando Usuários	66
8.4	Definindo Senha para Novos Usuários	66
8.5	Apagando uma Conta de Usuário	66
8.6	Modificando Conta de Usuário	67
8.7	Adicionando um Novo Grupo	67
8.8	Deletando um Grupo	67
8.9	Modificando um Grupo	67
9	Comandos para Redes de Computadores	69
9.1	Comando hostname	69
9.2	Comando e Tabela ARP	69
9.3	Verificando o Endereço IP de sua Máquina	70
9.3.1	Verificando Endereço IP	70
9.4	Habilitando e Desabilitando a Interface de Rede	70
9.5	Alterando a MTU de uma Interface	71
9.6	Alterando Endereço IP	72

9.7	Comando ping	73
9.8	Descobrir endereço IP de um Determinado Host	74
9.9	Informações sobre Domínios	74
9.9.1	Comando dig	74
9.9.2	Comando nslookup	75
9.10	Traçando caminhos de um host a outro	75
9.10.1	Descobrindo o Endereço do seu Roteador sem Fio	76
9.11	Comando tracepath	76
9.12	Comando netstat	77
9.12.1	Tabela de Roteamento	78
9.13	Network Mapper	78
9.13.1	Instalação	78
9.13.2	Analizando portas abertas	79
9.13.3	Comando nmap com opção de mais informações	79
9.13.4	Rastreando Múltiplos Hosts	79
9.14	Comando route	81
9.15	Comando telnet	81
9.15.1	Acessando Servidor Web via Telnet	81
9.16	Acesso Remoto com ssh	82
9.16.1	Acesso Remoto	82
9.16.2	Rodando Aplicativos Gráficos Remotamente	82
9.17	Copiando Arquivos com scp	83
9.18	Copiando um Diretório em um Servidor Remoto	84
9.19	Comando tcpdump	84
9.20	Navegando no Terminal	84
9.21	Baixando Sites com wget	85
10	Gerenciamento de Pacotes	87
10.1	Atualização de Pacotes	87
10.2	Atualizando a Distribuição	87
10.3	Instalando Softwares	87
10.4	Removendo Pacotes	87
10.5	Instalando Software no Fedora	88
11	Comandos Úteis	89
11.1	Comando unit	89
11.2	Comando yes	89
12	Comandos Divertidos	91
12.1	Comando cowsay	91
12.2	Comando xcowsay	91
12.3	Comando fortune	92
12.4	Comando xcowfortune	92
12.5	Comando sl	92
12.6	Comando xeyes	93
12.7	Comando oneko	93

13 Comandos para Sistema de Arquivos	95
13.1 Entendendo Discos e Partições	95
13.2 Imprimir Tabela de Partições do Linux	95
13.3 Obtendo Informações sobre o Disco com fdisk	95
13.4 Comando sfdisk	96
13.5 Listando Informações sobre as Partições com lsblk	96
13.6 Listando Informações sobre Permissões do Disco com lsblk	96
14 Comandos para Analisar o Desempenho do Linux	99
14.1 Analisando Consumo de CPU com o Comando sar	99
14.2 Analisando Desempenho de CPU com mpstat	99
14.3 Estatísticas de Entrada e Saída com iostat	100
14.4 Analisando a Memória com vm_stat	100
14.5 Comando pidstat	100
14.6 Comando top	102
15 Verificando Configuração de Hardware e Software	105
15.1 Visualizando Informações sobre a Versão do Kernel	105
15.2 Verificando sua Distribuição	105
15.3 Visualizando Informações sobre a sua CPU	105
15.4 Visualizando Informações sobre os Dispositivos USB	105
15.5 Listando Todos os Dispositivos PCI	106
15.6 Listando Todos os Dispositivos de Bloco	107
15.7 Verificando Todas as Partições	107
15.8 Listando Dispositivos PCMCIA	108
15.9 Obtendo Informações sobre a Memória	108
15.10 Listando Todos os Dispositivos de Hardware	108

Lista de Figuras

3.1 Árvore de Diretórios do Linux	13
3.2 Estrutura de Diretórios do Linux	14
9.1 Firefox do computador Ubuntu rodando em um Mac	83
9.2 Interface do lynx	85
12.1 Comando xcowsay	92
12.2 Comando xcowfortune	93
12.3 Comando sl	93
12.4 Comando xyes	94
13.1 Comando fdisk	96
13.2 Comando sfdisk	97
13.3 Comando lsblk	97
13.4 Listando Permissões com lsblk	97
14.1 Uso do top para Obter Estatísticas de CPU	103
15.1 Obtendo Informações sobre a CPU	106
15.2 Obtendo Informações sobre as Conexões USB	106
15.3 Exemplo de Conexão de Dispositivo USB	106
15.4 Obtendo Informações sobre os Dispositivos PCI	107
15.5 Obtendo Informações sobre os Dispositivos de Bloco	107

Listas de Listagens

1.1	Acessando o Sistema	2
1.2	Prompt de Login	2
1.3	Alterando o Nome da Máquina	3
1.4	Explicando a Padronização de Apresentação	3
1.5	Formato dos Comandos	3
1.6	Opções dos Comandos	4
1.7	Opções Múltiplas	4
1.8	Shell Utilizado	4
1.9	Shells Instalados	4
1.10	Caminho Completo	5
1.11	Limpando a Tela	6
1.12	Exibindo Mensagens no Terminal	6
1.13	Histórico	7
1.14	Comando history	7
1.15	Executar um determinado comando do history	7
1.16	Exemplo de busca no history	7
1.17	Limpar o histórico	7
1.18	Tamanho do histórico	8
1.19	Aumentando o Tamanho de Comandos Armazenados	8
1.20	Reduzindo o Tamanho de Comandos Armazenados	8
1.21	Arquivo contendo o histórico de comandos	8
2.1	Comando logout	9
2.2	Comando exit	9
2.3	Desligando Imediatamente com shutdown	10
2.4	Desligando Imediatamente com poweroff	10
2.5	Desligando Após Determinado Intervalo de Tempo	10
2.6	Mensagens recebidas	10
2.7	Desligando Imediatamente	11
2.8	Desligando em 5 minutos	11
2.9	Cancelando Shutdown	11
2.10	Reinicializando com reboot	11
2.11	Reinicializando com shutdown	11
2.12	Reinicializando a Máquina após Determinado Intervalo de Tempo	11
2.13	Reinicializando a Máquina em Horário Específico	12
3.1	Listando o Conteúdo de um Diretório	14
3.2	Uma Entrada por Linha	14
3.3	Comando ls no formato longo	15
3.4	Símbolo -	15
3.5	Símbolo d	15
3.6	Símbolo 1	16

3.7	Obtendo informações sobre diretórios	16
3.8	Listando Todos os Arquivos Inclusive os Ocultos	16
3.9	Listando Apenas os Arquivos Ocultos	16
3.10	Classificando Arquivos e Diretórios	16
3.11	Tamanho em Blocos	17
3.12	Tamanho dos Arquivos	17
3.13	Combinando Opções do Comando ls	17
3.14	Visualizando Informações sobre o Diretório	17
3.15	Listando Recursivamente	18
3.16	Comando cd	18
3.17	Atalho para o Diretório Raiz do Usuário	18
3.18	Significado de ~	18
3.19	Comando cd Sem Opções	18
3.20	Retornando ao Diretório do Usuário	18
3.21	Comando pwd	19
3.22	Outro Exemplo do Comando pwd	19
3.23	Sintaxe do Comando cp	19
3.24	Copiando Arquivo para Diretório	19
3.25	Copiando Arquivo para um Diretório	19
3.26	Copiando Múltiplos Arquivos	20
3.27	Copiando um Arquivo em Outro	20
3.28	Copiando com a Opção -i	20
3.29	Copiando Diretórios	20
3.30	Renomeando Arquivos com rename	20
3.31	Criando Arquivos com touch	21
3.32	Apagando um Arquivo	21
3.33	Apagando Múltiplos Arquivos	21
3.34	Apagando Diretório	21
3.35	Apagando Diretório com rmdir	21
3.36	Caracteres permitidos pelo POSIX	22
3.37	Criando Diretório com mkdir	22
3.38	Criando Múltiplos Diretórios	23
3.39	Criando Árvore de Diretórios	23
3.40	Links versus Arquivos e Diretórios	23
3.41	Criando um Arquivo	23
3.42	Criando um Link Simbólico	24
3.43	Verificando o Conteúdo do Arquivo	24
3.44	Movendo o Arquivo	24
3.45	Visualizando os Links	24
3.46	Hard Links - Passo 1	24
3.47	Hard Links - Passo 2	25
3.48	Hard Links - Passo 3	25
3.49	Hard Links - Passo 4	25
4.1	Comando cut	27
4.2	Arquivo com Cidades	27
4.3	Separando dados de um Arquivo	27
4.4	Listagem Original	28
4.5	Tabulações Convertidas para 1 Espaço	28
4.6	Convertendo Tabulações	28
4.7	Convertendo tabulações em espaço simples	28

4.8	Convertendo tabulações em espaço simples e removendo múltiplas ocorrências de espaços	29
4.9	Convertendo MAIÚSCULA em minúsculas	29
4.10	Outra forma de converter MAIÚSCULA em minúsculas	29
4.11	Transformar espaços em quebra de linha	29
4.12	Substituir chaves por parênteses	29
4.13	Duas maneiras para se remover dígitos	29
4.14	Formatando Linhas com o Comando fmt	30
4.15	Listagem para uso do grep	30
4.16	Filtrando com grep	30
4.17	Imprime as Linhas Iniciais de um Arquivo	31
4.18	Imprime as Duas Linhas Iniciais de um Arquivo	31
4.19	Convertendo Padrões de Caracteres	31
4.20	Comando look	31
4.21	Visualizando Arquivos Longos	32
4.22	Contando o Número de Linhas	32
4.23	Arquivos de Exemplo para o Comando paste	33
4.24	Combinando Dois Arquivos com o Comando paste	33
4.25	Comando para Inverter os Caracteres - rev	33
4.26	Invertendo Caracteres com rev	33
4.27	Ordenando Arquivos com sort	34
4.28	Ordenando Arquivos com sort	34
4.29	Imprime as Linhas Finais de um Arquivo	34
4.30	Imprime as Duas Linhas Finais de um Arquivo	34
4.31	Comando uniq	35
4.32	Contando quantas palavras distintas existem em um texto	35
4.33	Contando o Número de Caracteres	35
4.34	Contando o Número de Linhas	35
4.35	Contando o Número de Palavras	36
4.36	Contando o Número de Bytes	36
5.1	Opções Múltiplas	37
5.2	Comando whoami e pwd	37
5.3	Arquivo passwd	38
5.4	Identificadores no Mac	39
5.5	Identificadores no Linux	39
5.6	Alterando a Senha	39
5.7	Usuários Logados	40
5.8	Visualizando Data e hora	40
5.9	Comando cal	40
5.10	Instalação do finger	40
5.11	Comando finger	41
5.12	Comando finger -l	41
5.13	Comando finger no Linux	41
5.14	Comando free	41
5.15	Comando free	42
5.16	Logar como Super Usuário	42
5.17	Verificar Informações sobre o Linux	42
5.18	Exemplo do Sistema Operacional Mac	42
5.19	Verificar sua Versão do kernel	43
5.20	Verificando a Plataforma	43
5.21	Verificar o Nome de sua Máquina	43

5.22	Apresenta todas as informações sobre seu sistema operacional	43
5.23	Tempo de Funcionamento	43
5.24	Versão de um comando	43
5.25	Localização de um Comando	44
5.26	Variável PATH	44
5.27	Comando w	44
5.28	Comando whereis	45
5.29	Comando locate	45
5.30	Comando which	45
5.31	Comando whatis	46
5.32	Rodando Múltiplos Comandos	46
5.33	Rodando Múltiplos Comandos Condicionados	46
5.34	Rodando Comando em Background	47
6.1	Todos os Processos em Execução	49
6.2	Todos os Processos em Execução de um Usuário Específico	49
6.3	Processos que Mais Consomem CPU	50
6.4	Processos que Mais Consomem CPU	50
6.5	Obtendo Informações de um Processo Específico	50
6.6	Obtendo a Lista de Processos em forma de Árvore	51
6.7	Comando top	51
6.8	Comando kill	52
6.9	Comando kill	52
6.10	Matando o Processo Firefox	52
6.11	Calculando Tempo de Execução de um Programa ou Script	52
6.12	Calculando Tempo de Execução de um Programa ou Script	53
6.13	Controlando Processos	54
6.14	Controlando Processos	54
6.15	Controlando Processos	54
6.16	Controlando Processos	55
6.17	Controlando Processos	55
6.18	Controlando Processos	55
6.19	Controlando Processos	56
6.20	Controlando Processos	56
6.21	Controlando Processos	56
6.22	Controlando Processos	56
6.23	Controlando Processos	56
6.24	Controlando Processos	57
6.25	Controlando Processos	57
6.26	Controlando Processos	57
6.27	Controlando Processos	58
6.28	Controlando Processos	59
6.29	Controlando Processos	59
6.30	Controlando Processos	59
6.31	Controlando Processos	59
6.32	Controlando Processos	60
6.33	Controlando Processos	60
6.34	Controlando Processos	61
7.1	Listando o Proprietário e as Permissões dos Arquivos	63
7.2	Trocando o Dono de um Arquivo	63
7.3	Trocando as Permissões de um Arquivo	64

7.4	Exemplos de Utilização do chmod	64
8.1	Listando Todos os Usuários do Linux	65
8.2	Listando Todos os Usuários do Linux	65
8.3	Listando Todos os Usuários do Linux	66
8.4	Adicionando Usuários	66
8.5	Definindo Senha de Usuário	66
8.6	Apagando Usuários	66
8.7	Apagando Usuários e Arquivos do Usuário	67
8.8	Modificando Conta do Usuário	67
8.9	Adicionando um Novo Grupo	67
8.10	Apagando um Grupo	67
8.11	Modificando um Grupo	67
9.1	Comando hostname	69
9.2	Nome da Máquina com cat	69
9.3	Tabela ARP	70
9.4	Verificando o Endereço IP	70
9.5	Verificando Status da Rede Ethernet	70
9.6	Desabilitando a Interface de Rede Ethernet	71
9.7	Verificando a Ação Realizada na Listagem 9.6	71
9.8	Habilitando a Interface de Rede Ethernet	71
9.9	Verificando a Ação Realizada na Listagem 9.8	71
9.10	Verificando MTU da Rede Ethernet	72
9.11	Alterando o MTU da Placa de Rede	72
9.12	Verificando o Endereço IP	72
9.13	Alterando Endereço IP	72
9.14	Comando ping	73
9.15	Exemplo do Comando ping	73
9.16	Comando ping	73
9.17	Comando ping com Opção de Tempo	74
9.18	Descobrindo o Endereço Ip de um Host	74
9.19	Descobrir Informações sobre um Domínio	74
9.20	Utilizando o comando nslookup	75
9.21	Instalação do Traceroute	75
9.22	Rotas	75
9.23	Endereço do Roteador sem Fio	76
9.24	Rotas com tracepath	76
9.25	Estatísticas de Rede com netstat	77
9.26	Comando netstat -i	78
9.27	Visualizando Tabela de Roteamento com netstat	78
9.28	Instalação do nmap	78
9.29	Verificando Portas Abertas	79
9.30	Comando nmap com opção -v	79
9.31	Rastreando Múltiplos Hosts.numbers	79
9.32	Rastreando Múltiplos Domínios	80
9.33	Rastreando uma Sub-rede	80
9.34	Visualizando a Tabela de Roteamento	81
9.35	Uso do Telnet	81
9.36	Resposta do Servidor	81
9.37	Acessando com Telnet o Servidor Web	81
9.38	Resposta do Servidor	82

9.39 Instalando ssh	82
9.40 Utilizando o ssh	82
9.41 Rodando Aplicativos Gráficos Remotamente	82
9.42 Logando com ssh -X	83
9.43 Abrindo Firefox Remotamente	83
9.44 Copiando Arquivo em Servidor Remoto	83
9.45 Copiando um Diretório de um Servidor Remoto	84
9.46 Interfaces que podem ser utilizadas com tcpdump	84
9.47 Capturando Pacotes da Interface de Rede Sem Fio	84
9.48 Instalação do lynx	85
9.49 Utilizando o lynx	85
9.50 Baixando Sites com wget	85
10.1 Atualização da Lista de Pacotes Disponíveis	87
10.2 Atualização de Pacotes	87
10.3 Instalação do Pacote vim	87
10.4 Removendo o Pacote vim	88
10.5 Instalação do Pacote lshw no Fedora	88
11.1 Exemplos de utilização do comando units	89
11.2 Exemplo de utilização do comando yes para responder automaticamente a perguntas com ‘yes’	89
11.3 Imprimindo uma mensagem indefinidamente no terminal utilizando o yes	89
11.4 Utilizando o Comando yes para Processamento de Latex	90
12.1 Instalação do Comando cowsay no Debian/Ubuntu	91
12.2 Instalação do Comando cowsay no Fedora	91
12.3 Comando cowsay	91
12.4 Comando cowsay	91
12.5 Comando cowsay	92
12.6 Comando fortune	92
12.7 Comando xcowfortune	92
12.8 Comando sl	93
12.9 Comando sl	93
12.10 Comando xeyes	93
12.11 Comando oneko	94
13.1 Listando as Partições do Disco Rígido	95
14.1 Analisando Desempenho da CPU com sar	99
14.2 Analisando Desempenho de Todos os Núcleos com mpstat	99
14.3 Analisando Estatísticas de Entrada e Saída com iostat	100
14.4 Analisando a Memória com vm_stat	100
14.5 Comando pidstat	101
15.1 Versão do Kernel	105
15.2 Verificando sua Distribuição	105
15.3 Verificando a CPU	105
15.4 Imprimindo as Partições	108
15.5 Listando Dispositivos PCMCIA	108
15.6 Listando Informações sobre a Memória	108
15.7 Analisando os Dispositivos de Hardware	109
15.8 Versão Resumida do Relatório do lshw	115

Capítulo 1 | Conceitos Básicos

Uma das grandes vantagens do Linux é sua extensa variedade de comandos. Este capítulo tem como objetivo apresentar os conceitos básicos sobre a utilização do Linux.

1.1 Instalação do Sistema Operacional Linux

Existem basicamente três maneiras de instalar e acessar o Linux. A primeira maneira, mais recomendada, é utilizar o Linux como sistema único. Para isto baixe a imagem de instalação e faça o procedimento descrito no site da distribuição de Linux.

A segunda maneira de instalar o Linux é compartilhar seu disco rígido com outro sistema operacional. É possível instalar o Linux e outro sistema operacional no mesmo computador. Neste caso teremos um gerenciador de boot e no momento da inicialização da máquina você deverá escolher o sistema operacional desejado.

A terceira forma é rodar o sistema operacional Linux ao mesmo tempo que outro sistema operacional. Para isto é necessário instalar um programa de virtualização, como por exemplo **Virtualbox** [5] ou outro similar. Este programa permite rodar dois sistemas operacionais ao mesmo tempo. Lembre-se que neste caso vamos precisar de mais memória para conseguirmos um desempenho razoável.

Não é objetivo deste livro explicar os processos de instalação pois variam para cada distribuição. Caso tenha alguma dúvida consulte o manual de instalação da distribuição escolhida.

1.2 Qual Distribuição?

Esta é uma pergunta difícil de responder, pois isto depende do gosto do desenvolvedor. Comece sempre de uma distribuição que o procedimento de instalação seja bem simplificado. Por esta razão sempre indicamos as distribuições **Ubuntu** [1], **Fedora** [3], **Debian** [2] ou **OpenSuse** [4].

1.3 Acessando o Sistema Operacional Linux

Existem várias maneiras de acessar o Linux e isto vai depender da forma que instalou o sistema. A maneira mais simples é entrar diretamente após o **boot** do computador quando apenas um sistema existe no disco rígido.

Se você compartilhou o disco rígido com outros sistemas operacionais, terá que realizar a seleção no momento do boot. Assim, ao ligar o computador, será apresentado o gerenciador de boot **grub** ou **lilo**. Caso a máquina já esteja ligada e exibindo a janela de login do Windows deve-se reinicializá-la, pressionando a combinação <CTRL+ALT+DEL>.

Escolha a opção Linux e aperte a tecla *ENTER*.

1.4 Acessando o Terminal do Linux

Existem duas maneiras de acessar o terminal (shell) do Linux. A primeira maneira é entrar na parte gráfica e selecionar o shell. A segunda maneira é utilizar os terminais virtuais presentes na distribuição.

Para acessar os terminais virtuais basta pressionar <CTRL+ALT+F1> para obter o primeiro dos 6 terminais (F1 a F6) virtuais disponíveis para se acessar o sistema.

O Linux possibilita o uso de até 63 terminais virtuais simultaneamente. Por default, são disponibilizados os 6 terminais, onde o usuário pode estar executando tarefas distintas em cada um deles.

Atalhos para os terminais:

- <CTRL+ALT+FN>: vai da interface gráfica para a console N(onde N vale de 1 a 6)
- <CTRL+ALT+F1>: vai para console 1
- <CTRL+ALT+F2>: vai para console 2
- <CTRL+ALT+F3>: vai para console 3
- ...
- <CTRL+ALT+F6>: vai para console 6
- <CTRL+ALT+F7>: volta para a interface gráfica

1.5 Entrando no Sistema

Escolha um terminal modo texto e entre no sistema. A Listagem 1.1 ilustra o procedimento de acesso. A palavra **login** indica o nome do usuário e **Password** indica a senha do usuário.

Listagem 1.1: Acessando o Sistema

```
Login: <digite o seu login>
Password: <digite a sua senha>
```

Para conseguir logar no sistema você deve ser previamente cadastrado no sistema pelo administrador. Ao digitar a senha, o Linux consultará o arquivo **passwd**, localizado no diretório **/etc**. Caso as informações estejam corretas, o sistema permitirá o acesso.

Ao realizar este processo, ocorre a distinção entre usuários, sendo permitido que várias pessoas possam usar a mesma máquina simultaneamente e que somente você (usuário) tenha acesso aos seus arquivos. Você não tem permissão para apagar ou modificar arquivos do sistema; isto é a grande diferença do Linux. Apenas o administrador tem o privilégio de acesso a estes arquivos.

No Linux chamamos o administrador do sistema de **root** e somente ele tem privilégios para fazer alterações no sistema operacional. Por isto não fique com medo de danificar alguma coisa no sistema, pois isto só poderá ser realizado se tiver privilégio do usuário **root**.

1.6 Significado do Shell

Se você teve sucesso no processo de login, irá aparecer um prompt. O símbolo do prompt não é fixo e varia conforme o interpretador de comandos (**shell**) ou configuração, Listagem 1.2.

Listagem 1.2: Prompt de Login

```
Last login: Fri Sep 4 14:13:10 2015 from 192.168.0.101
[avivas@musashi ~]$
```

Onde *avivas* é o nome do usuário e **musashi** é o nome da máquina. Deseja alterar o nome de sua máquina? Realize o procedimento descrito na Listagem 1.3.

Listagem 1.3: Alterando o Nome da Máquina

```
# logar como root
$ su -
Senha:
# edite o arquivo hostname
[root@musashi ~]# vi /etc/hostname
musashi.vivascorp
# salve o arquivo
```

Para os usuários comuns o prompt é o sinal \$ e pode também conter o nome do diretório em que você está naquele instante (diretório corrente). Geralmente, ao entrar no sistema, um interpretador de comandos - shell - é iniciado, o qual está associado à sua conta em seu diretório home. A Listagem 1.4 apresenta o formato apresentado no terminal.

Listagem 1.4: Explicando a Padronização de Apresentação

```
/home/jose$
```

Diretórios no Linux/UNIX são especificados por uma / e não uma \, diferentemente do que é definido para outros sistemas, como por exemplo o DOS.

1.7 Formato dos comandos

Os comandos (arquivos executáveis, chamados também de programas) no Linux, passados via shell, possuem a seguinte forma:

- <comando> → ls
- <comando><espaço><opções> → ls -lF
- <comando><espaço><opções><espaço><argumentos> → cp -R /home/vivas/teste /home/vivas/ensino/

A quase totalidade dos comandos possui todos os três elementos acima. A Listagem 1.5 apresenta o comando de copiar um arquivo de um diretório para outro arquivo que está em outro diretório, preservando os atributos do arquivo (permissões, dono, marca de tempo).

Listagem 1.5: Formato dos Comandos

```
cp -p /home/origem/texto.txt /home/destino/texto.txt
```

Entretanto, existem alguns comandos que possuem apenas opções, apenas argumentos ou nenhum destes, i.e., somente o próprio comando é necessário. O comando **clear**, utilizado para limpar a tela do terminal, não possui argumento algum.

É importante atentar para a existência ou não de espaços entre os caracteres ao se definir uma ação completa (comando <espaço> opção1 <espaço> opção2 <espaço> opção3 <espaço> argumento1 <espaço> argumento2 <espaço> argumento3).

Existe uma flexibilidade para se passar opções ao sistema operacional. Quando há a necessidade de se especificar mais de uma opção o usuário pode utilizar um aninhamento de opções, i.e., usar o mesmo hífen para especificar mais de uma opção:

- <comando> -<opção1> -<opção2> -<opção3>
- <comando> -<opção1><opção2><opção3>

Toda opção é precedida de um ou dois hífens(-), colocado sem espaços. Os dois hífens são usados para opções por extenso. Ex:

Listagem 1.6: Opções dos Comandos

```
$ls<espaco>-a<espaco>-l  
$ls<espaco>--all  
$pwd<espaco>--version  
$cd<espaco>--help
```

A opção é definida alternativamente por uma letra (-a, -l,-H) ou por extenso(--color,--size,--count). O uso de dois hífens isolados na linha de comando estabelece para o interpretador que não haverá mais opções a serem passadas para se efetuar aquele comando. Caso exista algo do tipo -texto escrito logo em seguida aos dois hífens em sequência, este não será interpretado pelo shell como uma opção, mas sim como um argumento. A Listagem 1.7 apresenta um exemplo, no qual a opção -F não será interpretada como opção, conforme explicado anteriormente.

Listagem 1.7: Opções Múltiplas

```
$ls -la -- -F
```

1.8 Shells

Ao entrar no Linux, o sistema habilita um shell para trabalho. O **shell** (interpretador de comandos) default, especificado pela configuração inicial, é o **bash** (localizado no diretório */bin/bash*), mas pode-se alterar essa escolha.

- bash - \$ - Bourne Again Shell. O shell mais utilizado (e mais poderoso) do Linux. Criado e distribuído pelo projeto GNU. Oferece comandos de edição de linha, substituição baseado no histórico e compatibilidade com o Bourne shell(sh).
- csh - % C shell. Desenvolvido em Berkeley. Compatível com Bourne Shell para uso interativo, mas tem uma interface diferente de programação. Não oferece comandos de edição.
- ksh - Korn shell - O shell mais popular do Unix e o primeiro a introduzir as técnicas modernas de shell no Bourne shell. Oferece comandos de edição de linha.
- sh - \$ - Bourne Shell. Shell original do linux. Não oferece comandos de edição.
- zsh - z shell. O mais novo dos shells. Compatível com Bourne shell e oferece comandos de edição.
- tcsh - % - um C shell melhorado.

1.8.1 Descobrindo o Shell

Para saber qual shell você está utilizando basta digitar o comando apresentado na Listagem 1.8. O comando **echo** é utilizado para imprimir variáveis de ambiente ou textos no terminal.

Listagem 1.8: Shell Utilizado

```
[avivas@musashi ~]$ echo $SHELL  
/bin/bash
```

Para saber quais interpretadores de comandos estão instalados em seu sistema utilize o comando apresentado na Listagem 1.9. O comando **cat** é utilizado para concatenar arquivos, ou entrada padrão, e imprimir o resultado na saída padrão, sendo, muitas vezes utilizado para ler conteúdos de arquivos, visualizando-os no terminal.

Listagem 1.9: Shells Instalados

```
[avivas@musashi ~]$ cat /etc/shells
```

```
/bin/sh  
/bin/bash  
/sbin/nologin  
/usr/bin/sh  
/usr/bin/bash  
/usr/sbin/nologin
```

1.9 Case Sensitive

O sistema é do tipo Case Sensitive, i.e., letras maiúsculas são diferenciadas de minúsculas. Sendo assim, os arquivos Exemplo.txt, Exemplo.TXT, ExEmPIO.tXt e EXEMPLO.txt são entidades diferentes para o sistema. Isso inclui comandos, programas, opções de comandos e argumentos. Como iremos ver, tudo no Linux é tratado como arquivo.

1.10 Movimentação no terminal

- Apagar um caractere à esquerda: <backspace>
- Apagar uma linha inteira: <CTRL+U>
- Andar na linha de comandos: para percorrer os caracteres na linha do shell basta utilizar a seta de direção para esquerda (o cursor move para o próximo caractere à esquerda) ou seta para direita (o cursor move para o próximo caractere à direita)
- Apagando o caractere localizado sobre o cursor: <delete>
- Mover o cursor para o início da linha de comandos: <CTRL+A>
- Mover o cursor para o fim da linha de comandos: <CTRL+E>
- Apagar todos os caracteres localizados à esquerda do cursor: <CTRL+U>
- Copiar um conteúdo: o conteúdo recentemente apagado é copiado com a combinação <CTRL+Y>
- Apagar o que estiver à direita do cursor: <CTRL+K>

A combinação <CTRL+D> ao ser usada numa linha que contenha um grupo de caracteres, desempenhará a função da tecla <delete>. Caso não exista nada na linha de comando corrente, essa combinação desempenhará a função de logout.

1.11 Primeiros comandos

Um comando é um software que realiza uma determinada função - usualmente uma função especializada. Nos sistemas Unix, comando é um simples arquivo localizado geralmente no diretório `/bin` ou `/sbin`. Assim, define-se como caminho absoluto aquele caminho completo, desde o diretório raiz. A Listagem 1.10 apresenta o caminho completo para o comando `ls`.

Listagem 1.10: Caminho Completo

```
// caminho completo  
$ /bin/ls
```

A seguir são dados os comandos mais básicos. Para limpar a tela do terminal, use o comando `clear` ou a combinação <CTRL+L>. O cursor será posicionado no canto superior esquerdo:

Listagem 1.11: Limpando a Tela

```
$ clear
```

1.12 Visualizando textos longos no terminal

Para visualizar textos longos no terminal que, por ventura, tenham sido escondidos na parte superior da tela, basta pressionar as teclas:

- <SHIFT+PAGEUP>
- <SHIFT+PAGEDOWN>

Essas teclas rolarão o conteúdo que ficou além ou aquém do espaço de tela para baixo ou para cima.

Quando apagamos o conteúdo da tela com o comando `clear`(ou <CTRL+L>), na verdade estamos reposicionando o cursor. Isso quer dizer que parte da informação exibida na tela pouco antes da execução desse comando é deslocada para além do limite superior da tela. Para comprovar o fato, experimente limpar o conteúdo da tela e depois teclar <SHIFT+PAGEUP>.

1.13 Exibindo Mensagens

O comando `echo` disponibiliza mensagens na saída padrão (vídeo). Além disso, ele é usado para visualizar o conteúdo de variáveis de shell, que serão estudadas mais adiante. Exemplos na Listagem 1.12.

Listagem 1.12: Exibindo Mensagens no Terminal

```
[avivas@musashi ~]$ echo teste      ou
[avivas@musashi ~]$ echo 'teste'   ou
[avivas@musashi ~]$ echo
[avivas@musashi ~]$ echo -e
```

A opção `-e` habilita a interpretação de caracteres especiais, tais como:

- `\` : nova linha
- `\\\` : barra invertida
- `\t` : tabulação horizontal
- `\v` : tabulação vertical
- `\r` : retorno de linha
- `\nnn` : código ASCII correspondente

1.14 Histórico do Terminal

Para facilitar as coisas o Linux mantém o histórico dos comandos digitados, tanto válidos quanto inválidos. Isto evita que você fique perdendo tempo em digitar tudo novamente. Assim para navegar entre os últimos comandos passados ao sistema, utilize as setas direcionais (`↑` ou `↓`). Ao apertar diversas vezes, tais comandos irão aparecer na ordem cronológica inversa, i.e., do mais recente para o mais antigo.

Quando se tem um histórico com poucos comandos a navegação por setas direcionais pode ser feita sem problemas. No entanto, quando a lista passa a contar com 50 ou mais comandos, a busca de um dado comando passa a

ser enfadonha. Nesse caso, utilize o mecanismo de procura <CTRL+R>, cuja interface é apresentada na Listagem 1.13.

Listagem 1.13: Histórico

```
(reverse-i-search) `` :  
// comece a digitar aquele comando find  
(reverse-i-search)`fi': find / | grep a  
// basta digitar enter para executar o comando
```

Ao digitar o primeiro caractere, surgirá o comando mais recente que possui aquele caractere. Para refinar a seleção, deve-se continuar digitando outros caracteres e o comando mais próximo da seqüência digitada irá aparecer ao lado. Para executar a escolha reconhecida na busca basta apertar a tecla <ENTER>. Para editar o comando a tecla <backspace> deve ser usada.

Algumas vezes pode ser necessário editar algum comando do histórico antes de executá-lo. Para tanto você deverá utilizar as setas direcionais para direita ou esquerda (\leftarrow ou \rightarrow) quando ver o comando desejado para editá-lo antes de executar.

1.14.1 Comando *history*

O comando **history** pode ser executado para listar o histórico de comandos utilizados no Terminal. A Listagem 1.14 apresenta o resultado da execução do comando.

Listagem 1.14: Comando history

```
[avivas@musashi ~]$ history  
1 tar cvf etc.tar /etc/  
2 cp /etc/passwd /backup  
3 ps -ef | grep http  
4 service sshd restart  
5 /usr/local/apache2/bin/apachectl restart
```

Os comandos armazenados no histórico são apresentados em ordem cronológica e numerados. Para executar um dos comandos anteriores, basta utilizar exclamação e o número do comando. Por exemplo, para executar novamente o quarto comando, basta fazer

Listagem 1.15: Executar um determinado comando do history

```
[avivas@musashi ~]$ !4  
service sshd restart
```

Para procurar um determinado comando no *history*, basta usá-lo em combinação com o comando *grep*. O exemplo abaixo na Listagem 1.16 ilustra o caso em que desejamos localizar um comando utilizado que contenha a palavra-chave *sshd*.

Listagem 1.16: Exemplo de busca no history

```
[avivas@musashi ~]$ history | grep sshd  
4 service sshd restart  
6 history | grep sshd
```

Você pode limpar todo o histórico utilizando o comando exemplificado na Listagem 1.17.

Listagem 1.17: Limpar o histórico

```
[avivas@musashi ~]$ history -c
```

O tamanho máximo do histórico é definido pela variável de ambiente *HISTSIZE*. Você poderá verificar o valor desta variável e modificá-lo, se julgar necessário, conforme exemplificado a seguir.

Para listar o valor atual da variável de ambiente que define o tamanho máximo utilize o código da Listagem 1.18.

Listagem 1.18: Tamanho do histórico

```
[avivas@musashi ~]$ echo $HISTSIZE  
1000
```

Suponha que deseja aumentar o tamanho para 1000 linhas, então proceda como na Listagem 1.19.

Listagem 1.19: Aumentando o Tamanho de Comandos Armazenados

```
[avivas@musashi ~]$ export HISTSIZE=1000
```

Suponha que deseja diminuir o tamanho para 0 linhas, isto é, não irá armazenar mais nenhuma linha. Para isto proceda como na Listagem 1.20.

Listagem 1.20: Reduzindo o Tamanho de Comandos Armazenados

```
[avivas@musashi ~]$ export HISTSIZE=0
```

Quando o shell é inicializado o histórico de comandos é iniciado no arquivo *~/.bash_history*. O histórico de comandos pode então ser acessado através deste arquivo. Verifique digitando o comando na Listagem 1.21.

Listagem 1.21: Arquivo contendo o histórico de comandos

```
[avivas@musashi ~]$ cat ~/.bash_history
```

Capítulo 2 | Ligando e Desligando o Linux

Este capítulo tem como objetivo apresentar as maneiras corretas de desligar o sistema pelo terminal. Apresentaremos também os comandos utilizados para logar e deslogar do sistema.

2.1 Saindo do sistema

2.1.1 Saindo do Sistema com Logout

Ao terminar seu trabalho você deve sair do sistema, o comando **logout** é utilizado para fechar sua conta para que outras pessoas não entre no seu sistema e accesse seus arquivos. A sintaxe é bastante simples e funciona quando você entrou no sistema via terminal, o procedimento é apresentado na Listagem 2.1.

Listagem 2.1: Comando logout

```
[avivas@musashi ~]$ logout
```

2.2 Saindo dom Sistema com Exit

Você pode também sair do terminal usando o comando **exit**. Apesar de serem praticamente iguais, o **exit** pode ser utilizado em qualquer script enquanto o **logout** não. O procedimento de uso do comando **exit** é apresentado na Listagem 2.2.

Listagem 2.2: Comando exit

```
[avivas@musashi ~]$ exit
```

Caso você tenha entrado via interface gráfica (blackbox, kde, gnome, windowmaker) existem alternativas gráficas para realizar esta tarefa. Você pode também utilizar a combinação <CTRL+ALT+backspace>, ou ainda, a opção mais rápida <CTRL+D>.

2.3 Desligando e Reinicializando o Sistema

Outra forma de sair do sistema é desligando a máquina. Nunca desligue a máquina sem os comandos apropriados, pois isto pode corromper o sistema de arquivos do Linux. Ao se desligar a máquina corretamente, o Linux finalizará os programas, gravará os dados no disco rígido e começará a mostrar procedimentos de finalização (FS, sinais KILL, SIGTERM para os processos residentes na memória).

2.3.1 Desligando Imediatamente

Para desligar o computador utilizamos o comando **shutdown**. Se quiser desligar imediatamente utilize o código da Listagem 2.3. Ele vai pedir a senha do administrador para desligar a máquina. A opção **h** significa que é para parar o computador. Para desligar a máquina você terá que ter a senha de **root**.

Listagem 2.3: Desligando Imediatamente com shutdown

```
[avivas@musashi ~]$ shutdown -h now
===== AUTHENTICATING FOR org.freedesktop.login1.power-off ===
É necessária autenticação para desligar o sistema.
Authenticating as: root
Password:
```

Outra maneira de desligar a máquina é utilizar o comando **poweroff**. Para utilizá-lo veja a Listagem 2.4. Ele funciona da mesma maneira que o comando **shutdown -h now**.

Listagem 2.4: Desligando Imediatamente com poweroff

```
$ sudo poweroff
```

2.3.2 Desligando após um determinado tempo

Se você quer desligar a máquina após 3 minutos utilizamos o mesmo comando, mas passamos como argumento o tempo como na Listagem 2.5.

Listagem 2.5: Desligando Após Determinado Intervalo de Tempo

```
$ sudo shutdown -h +3
```

Se outra pessoa estiver logada no sistema irá receber mensagens parecidas com as da Listagem 2.6.

Listagem 2.6: Mensagens recebidas

```
$
Espalhar mensagem de vivas@zafu
(/dev/pts/0) em 9:14 ...

The system is going down for halt in 2 minutes!
Desligando

Espalhar mensagem de vivas@zafu
(/dev/pts/0) em 9:15 ...

The system is going down for halt in 1 minute!
Desligando

Espalhar mensagem de vivas@zafu
(/dev/pts/0) em 9:16 ...

O sistema está sendo paralisado AGORA!
Desligando
```

2.3.3 Desligando em uma hora específica

Para desligar em uma hora determinada basta passar a hora desejada como argumento. A Listagem 2.7 apresenta o comando para desligar a máquina às 10:10 da manhã.

Listagem 2.7: Desligando Imediatamente

```
$ sudo shutdown -h 10:10
```

2.3.4 Cancelando um shutdown

Quer interromper o comando de shutdown? Vamos supor que tenha digitado o seguinte comando da Listagem 2.8.

Listagem 2.8: Desligando em 5 minutos

```
$ sudo shutdown -h +5
```

Para cancelar um **shutdown** vá em outro terminal e digite o comando da Listagem 2.9.

Listagem 2.9: Cancelando Shutdown

```
$ sudo shutdown -c  
shutdown: Desligamento cancelado
```

Outra maneira de cancelar o desligamento da máquina é ir no terminal e digitar <CONTROL+C>.

2.4 Reinicializado a máquina

Para reiniciar uma máquina podemos utilizar o comando **reboot** , Listagem 2.10.

Listagem 2.10: Reinicializando com reboot

```
$ sudo reboot  
Password:
```

O comando apresentado na Listagem 2.11 tem o mesmo resultado do comando **reboot**.

Listagem 2.11: Reinicializando com shutdown

```
$ sudo shutdown -r now  
Password:
```

2.4.1 Reinicializando após determinado tempo

Para programar o tempo na qual a máquina irá reiniciar utilizamos a opção **-r +tempo**. A Listagem 2.12 apresenta o comando.

Listagem 2.12: Reinicializando a Máquina após Determinado Intervalo de Tempo

```
$ sudo shutdown -r +3  
Password:
```

2.4.2 Reinicializando em uma determinada hora

Para programar a hora na qual a máquina irá reiniciar utilizamos a opção **-r +tempo**. A Listagem 2.13 apresenta o comando.

Listagem 2.13: Reinicializando a Máquina em Horário Específico

```
$ sudo shutdown -r +8:25  
Password:
```

Capítulo 3 | Operações em Diretórios e Arquivos

Este capítulo tem como objetivo apresentar os principais comandos para realizar operações em arquivos e diretórios.

3.1 Árvore de Diretórios

A estrutura de diretórios do Linux é uma árvore invertida, isto é, a raiz da árvore de diretórios é o topo. O diretório raiz é representado por uma barra (/) e é chamado de **root**. A Figura 3.1 apresenta a árvore de diretórios do Linux.

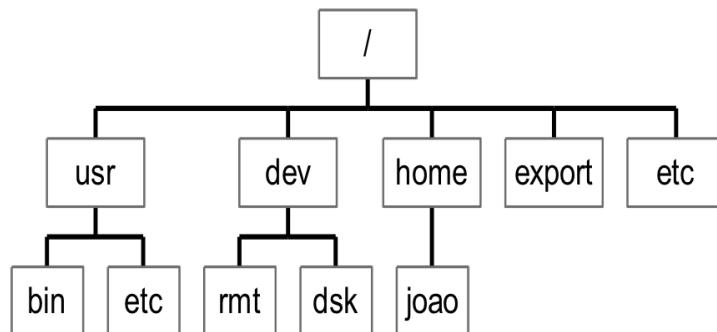


Figura 3.1: Árvore de Diretórios do Linux

3.2 Estrutura de Diretórios do Sistema Linux

A estrutura do sistema de arquivos adotada pelo sistema Linux é ilustrada na Figura 3.2. O diretório raiz (*root*) é o local aonde todos os arquivos e diretório se iniciam. Apenas o usuário administrador (*root*) possui privilégio para escrever neste diretório. Note que o diretório */root* é o diretório *home* do administrador (usuário *root*), o que é diferente do diretório / (diretório raiz).

/bin contém arquivos binários executáveis, inclusive os comandos comuns utilizados, como por exemplo **ps**, **ls**, **ping**, **cp**, etc.

/sbin contém arquivos binários executáveis do sistema, como por exemplo **iptables**, **reboot**, **fdisk**, **ifconfig**, etc.

/etc contém arquivos de configuração necessários por todos os programas; contém também *scripts* de inicialização e finalização de programas

/dev contém arquivos de dispositivos como disco rígido, dispositivos usb, etc.

/proc contém informações sobre os processos do sistema

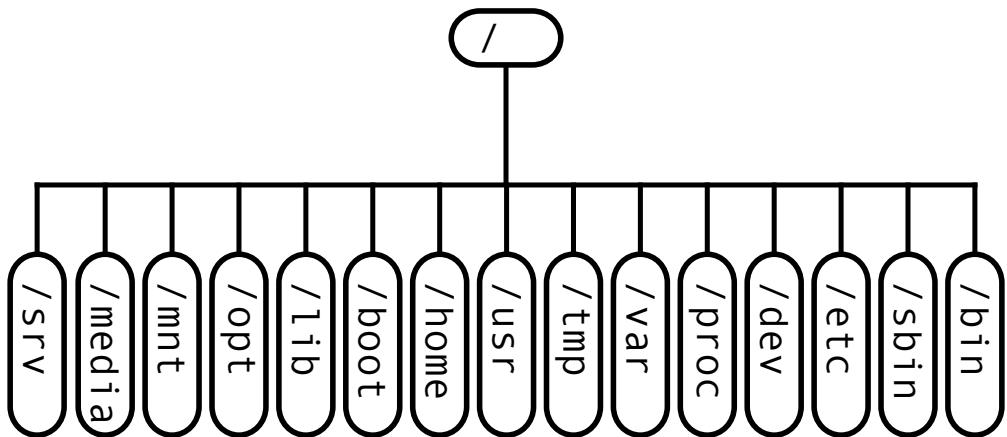


Figura 3.2: Estrutura de Diretórios do Linux

/var contém arquivos variáveis (mudam com o tempo), como por exemplo, arquivos de *log* no diretório */var/log*, arquivos de pacotes e bases de dados em */var/lib*, fila de impressão em */var/spool*; etc

/tmp contém arquivos temporários criados pelo sistema e usuários

/usr contém arquivos binários, bibliotecas, documentação e códigos fonte de programas de segundo nível

/home contém os diretórios de todos os usuários e seus arquivos pessoais

/lib contém arquivos de biblioteca que suportam os binários localizados em */bin* e */sbin*

/opt contém aplicativos de fornecedores individuais

/mnt diretório temporário de montagem de sistema de arquivos

/media diretório temporário de montagem de dispositivos removíveis, por exemplo, */media/cdrom*, */media/floppy*

/srv contém dados específicos de alguns serviços

3.3 Listando o Conteúdo do diretório

O comando **ls** executa essa tarefa. O comando **dir** (herdado de outros S.O.s) pode também existir devido a uma pré-configuração. As principais opções utilizadas são mostradas a seguir. O comando **ls**, Listagem 3.1, sem opções exibe o conteúdo na forma de uma lista.

Listagem 3.1: Listando o Conteúdo de um Diretório

```
$ ls
```

3.4 Listando uma única entrada por linha

O comando **ls -1** lista o conteúdo do diretório adicionando uma entrada (diretório e/ou arquivos) por linha. A Listagem 3.2 apresenta o resultado do comando.

Listagem 3.2: Uma Entrada por Linha

```
$ ls -1
afpovertcp.cfg
aliases
aliases.db
```

```
apache2
asl
asl.conf
authorization.deprecated
...
```

3.5 Listando o Conteúdo no Formato Longo

Imprimir uma ocorrência por linha em formato longo, contendo as informações de cada arquivo (permissões, dono, data de criação ou modificação, tamanho, etc.), Listagem 3.3.

Listagem 3.3: Comando ls no formato longo

```
% ls -l
total 9292

-rw-r--r--  1 maria    cvs          3640 Nov  32001 Coments.txt~
drwx-----  3 maria    maria        4096 Oct  16 10:11 Desktop
drwxr-xr-x  2 root     root         4096 Oct  16 16:41 bg
drwxr-xr-x  2 maria    cvs          4096 Oct  15 19:47 codigos
-rw-r--r--  1 root     root         523930 Oct  16 16:33 dist10.tgz
-rw-r--r--  1 root     root         18 Oct   9 18:03 ftpusers
drwxr-x---  3 maria    staff        4096 Oct  15 21:03 ptalk
-rw-r--r--  1 root     root         8304640 Oct  16 16:34 rtp.4.0.tar
drwxr-xr-x  5 maria    cvs          4096 Oct  16 16:57 rtp-uni
-rw-r--r--  1 maria    root         481280 Oct  16 16:34 rtp24dez.tar
-rw-r--r--  1 root     root         122689 Oct  16 16:33 rtplib.tar.Z
-rw-r--r--  1 vivas   cvs          239 Oct   16 16:54 scilab.hist
drwxr-xr-x  2 maria    cvs          4096 Oct  15 21:20 spim
drwxr-xr-x  2 maria    cvs          4096 Nov  13 18:25 temp
drwxr-xr-x  3 maria    cvs          4096 Nov  13 17:55 templ
-rw-r--r--  1 maria    cvs          3 Nov   13 18:04 teste2.c
```

3.6 Informações sobre os arquivos e diretórios

Você já viu que quando listamos (formato longo) um arquivo aparecem algumas informações? Vamos nos concentrar no primeiro símbolo, Listagem 3.4. Como podemos observar se trata de um arquivo, pois o primeiro símbolo é um traço (-).

Listagem 3.4: Símbolo -

```
$ ls -l /etc/hosts
-rw-r--r-- 1 root wheel 372 4 Out 2012 /etc/hosts
```

Se for um diretório iremos encontrar o símbolo **d**, Listagem 3.5.

Listagem 3.5: Símbolo d

```
drwxr-xr-x@ 12 root wheel 408 24 Out 17:38 usr
```

Pode ser também um link quando a primeira letra é um **l**.

Listagem 3.6: Símbolo 1

```
lrwxr-xr-x@ 1 root wheel 11 24 Out 17:11 tmp -> private/tmp
```

3.7 Obtendo informações sobre diretórios

Se você deseja obter informações sobre um determinado diretório mas não quer listar o conteúdo, utilize a opção **ls -ld**. A Listagem 3.7 apresenta o resultado do comando aplicado ao diretório **/etc**.

Listagem 3.7: Obtendo informações sobre diretórios

```
$ ls -ld /etc
lrwxr-xr-x@ 1 root wheel 11 24 Out 2013 /etc -> private/etc
```

3.8 Listando Arquivos Ocultos

Exibir todos os arquivos, inclusive os arquivos escondidos (iniciados pelo ponto final **.arq_oculto**), devemos utilizar o comando **ls -a** ou **ls -all**, Listagem 3.8.

Listagem 3.8: Listando Todos os Arquivos Inclusive os Ocultos

```
$ ls -a
.
..
.DCOPserver_dcomp .gtkrc-kde Comentarios.txt~ rtp24dez.tar
.DCOPserver_dcomp.adm.unibh.br .kde Desktop rtplib.tar.Z
.ICEauthority .mcop bg scilab.hist
.MCOP-random-seed .mcoprc codigos spim .Xauthority
.msgnow dist10.tgz temp .bash_history .octave_hist
ftpusers temp1
```

Caso desejemos listar apenas os arquivos ocultos, poderemos utilizar um dos seguintes comandos exemplificados na Listagem 3.9.

Listagem 3.9: Listando Apenas os Arquivos Ocultos

```
ls -d .??
ls -a | egrep '^\..'
```

3.9 Classificando Arquivos e Diretórios

Classificar os arquivos (**/** para diretórios, ***** para executáveis, **@** para links simbólicos, **l** para FIFOs e **=** para sockets), Listagem 3.10.

Listagem 3.10: Classificando Arquivos e Diretórios

```
$ ls -F
Co.txt~ codigos/ ptalk-gui/ rtplib.tar.Z temp1/
```

```
Desktop/          dist10.tgz  rtp-1.4.0.tar  scilab.hist teste.c  
a.out*           ftpusers    rtp-uni/       spim/ teste.o bg/  
ptalk/          rtp24dez.tar  temp/ teste2.c
```

3.10 Imprimindo Informações sobre o Tamanho dos arquivos.

Para imprimir os arquivos com o tamanho em blocos, Listagem 3.11.

Listagem 3.11: Tamanho em Blocos

```
% ls -s  
total 9308  
4 Comentarios.txt~  516 dist10.tgz      4 rtp-uni        4 temp  
4 Desktop          4 ftpusers       476 rtp24dez.tar   4 temp1  
8 a.out            4 ptalk         124 rtpplib.tar.Z  4 teste.c  
4 bg               4 ptalk-gui     4 scilab.hist    4 teste.o  
4 codigos          8124 rtp-1.4.0.tar   4 spim          4 teste2.c
```

Para imprimir o tamanho em Kilobytes, Megabytes ou Terabytes utilize o código da Listagem 3.12.

Listagem 3.12: Tamanho dos Arquivos

```
% ls -lah  
-rw-r--r--@ 1 alessandrovivas staff 72K 5 Nov 2012 PlanilhaTreinamento.xlsx  
-rwxrwxrwx 1 alessandrovivas staff 194K 19 Nov 2010 Plano Trabalho Bolsista.doc
```

Pode-se usar uma composição de opções **ls -alF**.

Listagem 3.13: Combinando Opções do Comando ls

```
$ ls -aFl  
total 9388  
  
rw----- 1 maria    maria        4360 Nov 13 19:30 .bash_history~  
-rw-r--r-- 1 maria    maria        506 Nov 11 13:30 .bash_profile  
-rw-r--r-- 1 maria    maria       1120 Nov 11 13:30 .bashrc
```

Algumas vezes você deseja visualizar o diretório e não o conteúdo do mesmo. Para isto, basta usar a opção **-d**:

Listagem 3.14: Visualizando Informações sobre o Diretório

```
$ ls -dl /usr /usr/bin/  
  
drwxr-xr-x 13 root      root      4096 Nov 24 18:26 /usr  
drwxr-xr-x  2 root      root     12288 Nov 25 08:56 /usr/bin/
```

3.11 Listando Recursivamente

Para listar recursivamente o conteúdo de um diretório e seus sub-diretórios utilize o comando da Listagem 3.15. Para interromper o comando digite <**CONTROL+R**>.

Listagem 3.15: Listando Recursivamente

```
$ ls -R
3Area de Trabalho
    Artigos-Tutoriais-Livros
    Biblioteca
6...
```

3.12 Navegando em Diretórios

Use o comando **cd** para a navegação nos diretórios do sistema de arquivo do Linux. Para ir ao seu diretório home (seu diretório padrão de trabalho), basta digitar o comando da Listagem 3.16. Se o usuário for pedro então para ir para o diretório é necessário digitar o caminho completo. O comando para navegar nos diretórios é o **cd**

Listagem 3.16: Comando cd

```
$ cd /home/pedro
```

Para ir rapidamente ao seu diretório home, apenas digite:

Listagem 3.17: Atalho para o Diretório Raiz do Usuário

```
$ cd ~
```

O símbolo **~** é expandido pelo interpretador como */home/seu_usuario* ao executar o comando. Para verificar isso, experimente:

Listagem 3.18: Significado de **~**

```
$ echo ~
/home/vivas
```

Alternativamente pode ser usada uma forma reduzida que produzirá o mesmo efeito, ou seja, o comando **cd** sozinho, Listagem 3.19.

Listagem 3.19: Comando cd Sem Opções

```
$ cd
```

Os diretórios **.** e **..** referem-se ao diretório corrente e diretório-pai, respectivamente. Esses diretórios podem ser usados em sintaxes de comandos, assim como o **~** também. Exemplos:

- **ls -la .**
- **ls -F ..**
- **cd ..**
- **cd ./dir1/dir2**

Para retornar para o diretório-pai basta usar o código da Listagem 3.20.

Listagem 3.20: Retornando ao Diretório do Usuário

```
$ cd /home/usuario
```

```
3$ cd ..
```

3.13 Comando pwd

Sempre que você está no terminal do sistema, você está sempre dentro de algum diretório. Para saber qual é a sua localização atual, você poderá utilizar o comando **pwd** como na Listagem 3.21.

Listagem 3.21: Comando pwd

```
$ pwd  
/home/john
```

Todo diretório possui dois arquivos especiais cujos nomes consiste em um ou dois pontos: ‘.’ ou ‘..’. Estes designam o diretório corrente e o diretório pai, respectivamente.

Ao designar um arquivo, subentende-se que está sendo referenciado o arquivo no diretório corrente. Outra maneira é especificar o arquivo com o caminho completo, como ilustrado na Listagem 3.22.

Listagem 3.22: Outro Exemplo do Comando pwd

```
$ pwd  
/home/john  
$ ls test.txt  
$ ls /home/john/test.txt  
$ ls ~/test.txt
```

3.14 Copiando Arquivos

Ao usar o comando **cp (copy)**, pode-se efetuar cópias de arquivo ou grupo de arquivos, bem como diretórios inteiros. O comando precisa de dois argumentos: o arquivo original e o destino. A sintaxe geral é apresentada na Listagem 3.23.

Listagem 3.23: Sintaxe do Comando cp

```
$ cp <opcoes> </localfonte/arquivo> </localdestino/>
```

3.14.1 Copiando Arquivo para Diretório

Para copiar um arquivo para um diretório utilize o comando da Listagem 3.24. neste caso o arquivo está no diretório corrente e será copiado para o diretório /home/pedro/documentos.

Listagem 3.24: Copiando Arquivo para Diretório

```
$ cp code.c /home/pedro/documentos
```

A barra final como indicação de diretório de destino é essencial. Caso não seja colocado, o sistema interpretará o último elemento do caminho de destino como sendo um nome de arquivo. Assim, ao copiar o arquivo code.c (exemplo acima) ele teria seu nome alterado para seu_usuário.

Suponha que esteja no diretório /home/pedro/programas e deseja copiar o arquivo teste.c do diretório /home/pedro/aulas para /home/pedro/testes. Para isto você vai precisar de passar o caminho completo do diretório ou o caminho relativo, como apresentado na Listagem 3.25.

Listagem 3.25: Copiando Arquivo para um Diretório

```
$ cp /home/pedro/aulas/teste.c /home/pedro/testes  
$ cp ../aulas/teste.c ../testes
```

3.15 Copiando Múltiplos arquivos

Use o comando **cp** para copiar múltiplos arquivos para um diretório, podendo estar tais arquivos em locais diferentes como apresentado na Listagem 3.26.

Neste exemplo copiamos os arquivos *code.c* no diretório */home/pedro/dir1/* e o arquivo *main.c* no diretório */home/pedro/dir2/* e finalmente o arquivo *teste.c* do diretório */home/pedro/dir3/* para o diretório */home/pedro/*. Pelo exemplo, podemos ver a flexibilidade do sistema, tendo o usuário total liberdade de definir quantos e quais arquivos devem ser copiados para o diretório de destino.

Listagem 3.26: Copiando Múltiplos Arquivos

```
$ cp /home/pedro/dir1/code.c \home/pedro/dir2/main.c /home/pedro/dir3/teste.c /home/pedro/
```

Use o comando **cp** para copiar um arquivo para outro, Listagem 3.27. Dessa forma, o sistema não interpelará, i.e., caso haja um outro arquivo com o nome do arquivo a ser criado ele será sobreescrito sem nenhum impedimento. O sistema admitirá que você, usuário, sabe o que está fazendo. Para se efetuar a mesma ação com a necessidade de confirmação, você deve usar a opção **-i** para alertá-lo, caso seja necessário sobreescriver algum arquivo.

Listagem 3.27: Copiando um Arquivo em Outro

```
$ cp code.c main.c
```

A Listagem 3.28 apresenta o comando **copy** com confirmação (opção **-i**). Neste exemplo, cria o arquivo *main.c* caso não exista e lhe pede confirmação da ação caso seja necessário sobreescriver um arquivo já existente.

Listagem 3.28: Copiando com a Opção **-i**

```
$ cp -i code.c main.c  
cp: overwrite 'main.c'?
```

3.16 Copiando Diretórios e Sub-diretórios

Use a opção **-r** para copiar hierarquias inteiras de arquivos e sub-diretórios(inclusive arquivos ocultos) como apresentado na Listagem 3.29. Neste caso iremos copiar o diretório */home/maria/teste* para o diretório */home/maria/temp*.

Listagem 3.29: Copiando Diretórios

```
$ cp -r /home/maria/teste /home/maria/temp
```

3.17 Renomeando Arquivos

O comando **rename** (**rename**) é utilizado para renomear um ou mais arquivos. A sintaxe geral é apresentada na Listagem 3.30, através de um exemplo em que o arquivo *teste.txt* será renomeado para *teste2.txt*.

Listagem 3.30: Renomeando Arquivos com rename

```
$ rename teste.txt teste2.txt
```

3.18 Criando um Arquivo Vazio com touch

O comando **touch** (**touch**) é utilizado para criar um ou mais arquivos vazios. A sintaxe geral é apresentada na Listagem 3.31, através do exemplo que criará o arquivo vazio *teste.txt*.

Listagem 3.31: Criando Arquivos com touch

```
$ touch teste.txt
```

3.19 Apagando Arquivos

Para apagar um arquivo utilizamos o comando **rm** como mostrado na Listagem 3.32. Vamos supor que precisamos apagar o arquivo *teste.c*.

Listagem 3.32: Apagando um Arquivo

```
$ rm teste.c
```

3.19.1 Apagando Múltiplos Arquivos

Vamo supor que precisamos apagar todos os arquivos com a extensão **.c**. Utilize a Listagem 3.33.

Listagem 3.33: Apagando Múltiplos Arquivos

```
$ rm *.c
```

3.20 Apagando um Diretório

Quer apagar um diretório inteiro? Vamos utilizar o comando **rm** como na Listagem 3.34. Vamos supor que precisamos apagar o sub-diretório *musicas*, para tanto, será necessário apagar os arquivos dentro deste diretório e o próprio diretório. Podemos passar o argumento **-r** ou **-R** para que sejam apagados recursivamente os arquivos e subdiretórios dentro do diretório que desejamos remover e também o próprio diretório a ser removido.

Listagem 3.34: Apagando Diretório

```
$ rm -Rf musicas/
```

3.21 Apagando Diretório com rmdir

O comando **rmdir** pode ser utilizado para apagar diretórios como realizado na Listagem 3.35, onde é criado o diretório *teste* e em seguido é apagado.

Listagem 3.35: Apagando Diretório com rmdir

```
mkdir teste  
rmdir teste
```

3.22 Nomes de arquivos

Um arquivo é uma coleção de dados relacionados que compõem uma unidade, um bloco contíguo de informação que é armazenado, por exemplo, em um disco rígido, ou um disco ótico, etc. Os nomes são dados aos arquivos para permitir ao usuário identificar e acessar de forma simples estes dados. Os nomes de arquivos e diretórios são então apenas uma conveniência para os usuários.

Os nomes de arquivos em Linux podem conter qualquer caractere, exceto a barra /, que é reservada como separador de diretórios e para identificar o diretório raiz, e o caractere nulo (*null*) que é utilizado como terminação de um segmento. Espaços e caracteres especiais são permitidos, embora devam ser evitados para não causar incompatibilidade entre softwares.

Tipicamente os nomes de arquivos contém apenas caracteres alfanuméricos (em caixa baixa, majoritariamente), sublinhado (*underscore*), hifens e ponto final. Outros caracteres são evitados pois possuem significado especial para o *shell* e podem complicar a vida do usuário.

Embora os sistemas Unix não requerem a utilização de extensão de arquivo (uma sequência no final do nome do arquivo constituída de ponto final seguido de um a três caracteres), pode ser conveniente e útil a sua utilização, tornando simples a tarefa de identificar tipos de arquivo à uma primeira vista.

Os diretórios são vistos como apenas um tipo especial de arquivo e por isto as convenções de utilização de nomes para eles também são as mesmas.

A definição da interface POSIX, a fim de garantir compatibilidade e interoperabilidade entre sistemas operacionais, especifica algumas regras para a nomenclatura de arquivos. Os caracteres que podem ser utilizados são aqueles listados na Listagem 3.36. O tamanho máximo do nome de um arquivo deve ser 14. Desta forma, existem 24407335764928225040435790 combinações para compor nomes de arquivos com até 14 caracteres utilizando os 65 caracteres na Listagem 3.36.

Listagem 3.36: Caracteres permitidos pelo POSIX

```
ABCDEFGHIJKLMNPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789._-
```

Outras duas regras, para não criar distúrbios, são: não iniciar o nome de um arquivo com ponto o traço. Os arquivos cujo nome começam com ponto são os arquivos ocultos. Arquivos iniciados com traço podem ser entendidos como um argumento para muitos comandos.

Alguns arquivos possuem nomes tão diferentes que os tornam difíceis de serem apagados. Um exemplo são os arquivos cujos nomes se iniciam com um hífen. Se o arquivo possui espaços ou caracteres passíveis de interpretação pelo Linux (eg, * ou -) deve-se colocar aspas simples ao referenciá-lo:

3.22.1 Barra invertida

A barra invertida é utilizada para se possibilitar o uso de mais de uma linha na escrita de uma sintaxe ou de um encadeamento de comandos muito extenso. O interpretador tratará as linhas terminadas com \ como se fossem uma única linha, justamente para dar uma maior flexibilidade ao usuário nas suas definições. Um sinal de maior é incluído pelo interpretador para mostrar a continuidade da linha anterior.

3.23 Criando Diretório

O comando **mkdir** (*make directory*) cria um ou mais sub-diretórios. A Listagem 3.37 apresenta o código de criação do diretório teste.

Listagem 3.37: Criando Diretório com mkdir

```
$ mkdir teste
```

3.24 Criando Múltiplos Diretórios

Quer criar vários diretórios de uma única vez. Utilize o comando **mkdir** como apresentado na Listagem 3.38. Neste exemplo são criados os diretórios *tempdir1*, *tempdir2* e *tempdir3*.

Listagem 3.38: Criando Múltiplos Diretórios

```
$ mkdir tempdir1 tempdir2 tempdir3
```

3.25 Criar Hierarquia de Diretórios

Se você precisa criar uma hierarquia de sub-diretórios de uma única vez, basta usar a opção **-p** como na Listagem 3.39, desta forma todos os diretório ha hierarquia serão criados, caso não existam.

Listagem 3.39: Criando Árvore de Diretórios

```
$ mkdir -p temp/temp1/tempdir
```

3.26 Links

Sistemas Operacionais possuem um recurso que permite dar apelido para arquivos e ou diretórios. Este recurso é denominado atalho e é implementado pelo comando **In**. O comando **In** permite que isto seja realizado e deno-minamos estes apelidos de **links**. Os **links** podem ser simbólicos (**soft links**) ou (**hard links**). A Listagem 3.40 apresenta a listagem do diretório */etc* onde o símbolo **l** indica que *blkid.tab* é um link.

Listagem 3.40: Links versus Arquivos e Diretórios

```
vivas@zafu:~$ ls -l /etc
total 1140
3drwxr-xr-x  3 root root    4096 Ago 20 14:59 acpi
...
-rw-r--r--  1 root root    2981 Ago 20 14:56 adduser.conf
6...
lrwxrwxrwx  1 root root      15 Nov 21 19:08 blkid.tab -> /dev/.blkid.tab
...
```

Utilizamos links simbólicos para criar atalhos para arquivos ou diretórios. Por exemplo, para garantir que compatibilidade com um software antigo.

Vamos aprender a criar links simbólicos através do exemplo a seguir. O primeiro passo é criar um arquivo chamado *arq1* como na Listagem 3.41.

Listagem 3.41: Criando um Arquivo

```
$ touch arq1
$ echo 'Laranja' >> arq1
```

Agora vamos criar um link simbólico chamado *arq1-soft* para o arquivo *arq1* e listar os dois arquivos, Listagem 3.42.

Listagem 3.42: Criando um Link Simbólico

```
$ ln -s arq1 arq1-soft
$ ls -l arq1*
-rw-rw-r-- 1 vivas vivas 8 Jan  3 16:18 arq1
lrwxrwxrwx 1 vivas vivas 4 Jan  3 16:18 arq1-soft -> arq1
```

Agora vamos listar o conteúdo do arquivo *arq1-soft* como na Listagem 3.43. Como pode ser observado o conteúdo é o mesmo, pois o arquivo *arq1-soft* é um link para o arquivo *arq1*.

Listagem 3.43: Verificando o Conteúdo do Arquivo

```
$ cat arq1-soft
Laranja
```

Agora vamos mover o *arq1* para um novo arquivo chamado *arq2*. Após mover o arquivo vamos tentar listar o conteúdo do link simbólico como na Listagem 3.44.

Listagem 3.44: Movendo o Arquivo

```
$ mv arq1 arq2
$ cat arq1-soft
cat: arq1-soft: Arquivo ou diretório não encontrado
```

Vamos agora listar o *arq1-soft* e verificar que ele está apontando para o arquivo *arq1* que agora não existe mais como na Listagem 3.45

Listagem 3.45: Visualizando os Links

```
vivas@zafu:~$ ls -l arq1-soft
lrwxrwxrwx 1 vivas vivas 4 Jan  3 16:18 arq1-soft -> arq1
```

3.26.1 Hard Links

Os links simbólicos (*soft links*) são diferentes dos hard links pois os simbólicos referem-se a outros arquivos pelo nome. Os hard links fazem esta ligação pelo inode e por isto possuem o mesmo inode. Os links simbólicos fazem um atalho para o nome do arquivo e não para a posição do arquivo no disco.

Observações gerais

- Hard link apresenta em suas informações o mesmo tamanho do arquivo original.
- Link simbólico apresenta um tamanho reduzido, pois guarda o nome do arquivo original. O tamanho é tão grande quanto for o nome desse arquivo.
- Link simbólico possui inode distinto do arquivo original.
- Hard link possui o mesmo inode do arquivo original.

Vamos agora aprender a trabalhar com hard links. Um hard link aponta para o inode, isto é, para a posição do disco e não para o nome do arquivo (como os soft-link). O primeiro passo é criar um arquivo e preenche-lo com alguma informação como na Listagem 3.46.

Listagem 3.46: Hard Links - Passo 1

```
$ touch arq1
$ echo                  >> arq1
$ cat arq1
Laranja
```

Agora vamos criar um hard link e listar o conteúdo do mesmo como na Listagem 3.47.

Listagem 3.47: Hard Links - Passo 2

```
$ ln arq1 arq1-hard  
$ cat arq1-hard  
Laranja
```

Liste os arquivos criados e verifique o que aconteceu como na Listagem 3.48.

Listagem 3.48: Hard Links - Passo 3

```
$ ls -l arq1*  
-rw-rw-r-- 2 vivas vivas 8 Jan 3 16:45 arq1  
-rw-rw-r-- 2 vivas vivas 8 Jan 3 16:45 arq1-hard
```

Agora vamos mover o *arq1* para *arq2* e verificar o que aconteceu como na Listagem 3.49.

Listagem 3.49: Hard Links - Passo 4

```
$ mv arq1 arq2  
$ cat arq1-hard  
3Laranja  
$ ls -l arq1-hard  
-rw-rw-r-- 2 vivas vivas 8 Jan 3 16:45 arq1-hard
```

Como verificou no exemplo o *arq1-hard* continuou existindo pois quando um hard link é criado ele aponta para o inove (posição no disco) e não para o nome do arquivo.

Symlinks são distintamente diferentes de arquivos comuns. Portanto, podemos distinguir um symlink do arquivo original para o qual ele aponta. Symlinks podem referenciar(apontar) qualquer tipo de arquivo. Symlinks referem-se a nomes e, daí podem apontar para arquivos localizados em outro sistema de arquivos. Se você renomeia ou apaga o arquivo original apontado pelo symlink, o symlink se rompe. Symlinks podem ocupar espaço em disco adicional para armazenar o nome do arquivo apontado.

Hard links múltiplos, com nomes diversos para um mesmo arquivo, são ilimitados. Hard links trabalham com o número do inode e, portanto, eles só podem atuar dentro de um único sistema de arquivo. Ao copiar ou deletar o arquivo original apontado pelo hard link não se tem nenhum efeito sobre o hard link. Hard links necessitam somente de espaço suficiente para o armazenamento de uma entrada de diretório.

Capítulo 4 | Comandos para Manipulação de Arquivos Texto

4.1 Comando cut

Precisa cortar caracteres de uma frase? Vamos utilizar o comando **cut**. Neste exemplo, Listagem 4.1, vamos imprimir somente os caracteres 1 até 10 da palavra "alessandro vivas andrade".

Listagem 4.1: Comando cut

```
$ echo 'alessandro vivas andrade' | cut -c 1-10  
alessandro
```

Outro problema recorrente é separação de valores em um arquivo. Na Listagem 4.2 temos um arquivo com estado com dois caracteres e após o nome da cidade.

Listagem 4.2: Arquivo com Cidades

```
SP Sao Paulo  
SP Campinas  
MG Belo Horizonte  
MG Diamantina  
MG Lavras  
MG Bom Sucesso  
RJ Rio de Janeiro
```

Partindo da Listagem 4.2 vamos imprimir somente os dois primeiros caracteres de cada linha de um arquivo, Listagem 4.3.

Listagem 4.3: Separando dados de um Arquivo

```
$ cat cidades.txt | cut -c 1-2  
SP  
SP  
MG  
MG  
MG  
MG  
RJ
```

4.2 Comando expand

Imagine que agora nosso arquivo tem tabulações, Listagem 4.4 separando os campos e precisamos converter as tabulações em caracteres. Para realizar esta tarefa utilize o comando **expand**. A Listagem 4.4 apresenta os dados originais, armazenados no arquivo *cidades.txt*.

Listagem 4.4: Listagem Original

```
SP Sao Paulo:pedro
SP Campinas:andre
MG Belo Horizonte:marta
MG Diamantina:lucas
MG Lavras: rafael
MG Bom Sucesso:lisa
RJ Rio de Janeiro:alexandro
```

Convertendo a tabulação para 1 caractere na Listagem 4.5.

Listagem 4.5: Tabulações Convertidas para 1 Espaço

```
$ cat cidades.txt | expand -t 1
SP Sao Paulo:pedro
SP Campinas:andre
MG Belo Horizonte:marta
MG Diamantina:lucas
MG Lavras: rafael
MG Bom Sucesso:lisa
RJ Rio de Janeiro:alexandro
```

Convertendo a tabulação para 10 caracteres na Listagem 4.6.

Listagem 4.6: Convertendo Tabulações

```
cat cidades.txt | expand -t 10
SP      Sao Paulo:pedro
SP      Campinas:andre
MG      Belo Horizonte:marta
MG      Diamantina:lucas
MG      Lavras: rafael
MG      Bom Sucesso:lisa
RJ      Rio de Janeiro:alexandro
```

Podemos fazer este tipo de substituição utilizando um comando mais genérico, o comando **tr**, como veremos em seguida.

4.3 Comando tr

O comando **tr** é utilizado para efetuar substituições (ou tradução) e apagar caracteres.

A Listagem 4.7 apresenta um exemplo em que utilizaremos o comando **tr** para substituir tabulações por um único espaço simples.

Listagem 4.7: Convertendo tabulações em espaço simples

```
$ cat cidades.txt | tr '\t' ' '
SP Sao Paulo:pedro
SP Campinas:andre
MG Belo Horizonte:marta
MG Diamantina:lucas
MG Lavras: rafael
```

```
MG Bom Sucesso:lisa
RJ Rio de Janeiro:alexandro
```

Se além disso, queremos substituir as múltiplas ocorrências de espaços por um único espaço, podemos proceder como ilustrado na Listagem 4.8.

Listagem 4.8: Convertendo tabulações em espaço simples e removendo múltiplas ocorrências de espaços

```
$ cat cidades.txt | tr '\t' ' ' | tr -s ' '
SP Sao Paulo:pedro
SP Campinas:andre
MG Belo Horizonte:marta
MG Diamantina:luca
MG Lavras: rafael
MG Bom Sucesso:lisa
RJ Rio de Janeiro:alexandro
```

O comando **tr** pode ser utilizado para realizar diversos outros tipos de substituições ou para apagar caracteres indesejáveis. Veremos abaixo alguns exemplos.

Listagem 4.9: Convertendo MAIÚSCULA em minúsculas

```
$ cat cidades.txt | tr 'A-Z' 'a-z'
sp      sao paulo:pedro
sp      campinas:andre
mg     belo horizonte:marta
mg     diamantina:luca
mg     lavras: rafael
mg     bom sucesso:lisa
rj      rio de janeiro:alexandro
```

Uma outra forma de realizar a substituição de maiúsculas por minúsculas é apresentada na Listagem 4.10.

Listagem 4.10: Outra forma de converter MAIÚSCULA em minúsculas

```
$ tr [:upper:] [:lower:]
```

O exemplo da Listagem 4.11 ilustra como transformar espaços em branco (incluindo aqui tabulações e quebras de linhas) em uma quebra de linha. Para tanto, utilizaremos `[:space:]` para designar qualquer um dos caracteres: espaço, tabulação e quebra de linha.

Listagem 4.11: Transformar espaços em quebra de linha

```
$ tr -s [:space:] | tr [:space:] '\n'
```

Podemos utilizar o comando **tr** para substituir um conjunto de caracteres. Para tanto, será considerada a ordem em que eles parecem. No exemplo apresentado na Listagem 4.12 iremos substituir `{` por `(` e `}` por `)`.

Listagem 4.12: Substituir chaves por parênteses

```
$ tr '{}()' '()'
```

Caso deseje remover os algarismos de 0 a 9, basta utilizar uma das duas formas ilustradas na Listagem 4.13.

Listagem 4.13: Duas maneiras para se remover dígitos

```
$ tr -d [:digit:]
$ tr -d '0-9'
```

4.4 Comando fmt

O comando **fmt** é usado para formatar arquivos texto. Usado para organizar as palavras(grupos de caracteres) de um arquivo para uma forma consistente, i.e., com um número de caracteres por linha definido.

Seja o seguinte texto, sem delimitação de caracteres: "Antes da chegada dos colonizadores portugueses, no século XVI (os primeiros relatos dão conta de expedições que subiram o Rio Jequitinhonha e São Francisco), Diamantina, como toda a região do atual estado de Minas Gerais, era ocupada por povos indígenas do tronco linguístico ". Vamos supor que precisamos formatar o texto e limitar 10 caracteres por linha. Para fazer isto utilizamos o comando **fmt** da Listagem 4.14.

Listagem 4.14: Formatando Linhas com o Comando fmt

```
$ fmt -w10 texto.txt
Antes da
chegada
dos
colonizadores
portugueses,
no
...
continua
```

4.5 Comando grep

O comando **grep** pode ser utilizar para procurar padrões em arquivos texto. Ele pode ser utilizado sozinho ou em conjunto com outros comandos. Vamos usar a Listagem 4.15 como exemplo. Suponha que você deseja encontrar em um arquivo um determinado padrão, como por exemplo, alexandro.

Listagem 4.15: Listagem para uso do grep

```
SP Sao Paulo:pedro
SP Campinas:andre
MG Belo Horizonte:marta
MG Diamantina:lucas
MG Lavras: rafael
MG Bom Sucesso:lisa
RJ Rio de Janeiro:alexandro
```

Se os dados estiverem em um arquivo com nome *lista.txt* use a Listagem 4.16.

Listagem 4.16: Filtrando com grep

```
$ cat lista.txt | grep alexandro
RJ Rio de Janeiro:alexandro
```

O comando **|** é chamado de pipe e é utilizado para concatenar um ou mais comandos.

4.6 Comando head

O comando **head** é utilizado para imprimir os n linhas iniciais de um arquivo. Imagine um arquivo com o nome de *cidades.txt* com todas as cidades do Brasil. A Listagem 4.17 apresenta a lista impressa do arquivo *cidades.txt*.

Listagem 4.17: Imprime as Linhas Iniciais de um Arquivo

```
$ head cidades.txt
Abadia de Goias (GO)
Abadia dos Dourados (MG)
Abadiania (GO)
Abaete (MG)
Abaetetuba (PA)
Abaiara (CE)
Abaira (BA)
Abare (BA)
Abatia (PR)
Abdon Batista (SC)
```

Para imprimir as N linhas iniciais utilize o comando da Listagem 4.18. Neste exemplo iremos imprimir as duas linhas iniciais.

Listagem 4.18: Imprime as Duas Linhas Iniciais de um Arquivo

```
$ head -2 cidades.txt
Abadia de Goias (GO)
Abadia dos Dourados (MG)
```

4.7 Comando iconv

O comando **iconv** é utilizado para realizar conversões de caracteres, isto é, pegar uma string, texto ou arquivo codificado em um padrão de caracteres para outro padrão. Este comando pode ser útil, por exemplo, quando temos um arquivo texto feito no Windows e desejamos utilizá-lo no Linux. A Listagem 4.19 ilustra o exemplo do comando **iconv**.

Listagem 4.19: Convertendo Padrões de Caracteres

```
$ iconv -f iso-8859-1 -t utf-8 cidades.txt > cidadeutf8.txt
```

4.8 Comando look

O comando **look** é utilizado para visualizar linhas que possuem uma determinada string. Na Listagem 4.20 vamos procurar no arquivo *cidades.txt* as linhas que possuem a string Bom.

Listagem 4.20: Comando look

```
$ look Belo cidades.txt
Belo Campo (BA)
Belo Horizonte (MG)
Belo Jardim (PE)
Belo Monte (AL)
Belo Oriente (MG)
Belo Vale (MG)
```

4.9 Comando more

O comando **more** é utilizado para processar e visualizar arquivos longos que não cabem na tela do terminal. Ele pode ser utilizado em conjunto com outros comandos. O arquivo *cidades.txt* é um arquivo muito grande e uma opção de visualização é usar o **more** como na Listagem 4.21. Ele coloca as informações suficientes para ocupar o tamanho da tela. Para navegar no arquivo basta apertar espaço ou o número da página após os dois pontos. Para sair digite q.

Listagem 4.21: Visualizando Arquivos Longos

```
Abadia de Goias (GO)
Abadia dos Dourados (MG)
Abadiânia (GO)
Abaeté (MG)
Abaetetuba (PA)
Abaiara (CE)
Abaíra (BA)
Abaré (BA)
Abatiá (PR)
Abdon Batista (SC)
Abelardo Luz (SC)
Abel Figueiredo (PA)
Abre-Campo (MG)
Abreu e Lima (PE)
Abreulândia (TO)
Acaiaca (MG)
Açailândia (MA)
Acajutiba (BA)
Acará (PA)
Acarape (CE)
Acaraú (CE)
Acari (RN)
Acauã (PI)
:
```

4.10 Contar Número de Linhas - Comando nl

O comando **nl** conta o número de linhas de um arquivo. A Listagem 4.22 apresenta o uso do comando processando o arquivo *cidades.txt*.

Listagem 4.22: Contando o Número de Linhas

```
$ nl cidades.txt
...
5570 Xexeu (PE)
5571 Xinguara (PA)
5572 Xique-Xique (BA)
5573 Zabele(PB)
5574 Zacarias (SP)
```

4.11 Comando paste

O comando **paste** é interessante para unir arquivos diferentes formatando as colunas. Imagine que eu tenha separado em dois arquivos, *nomes.txt* com os nomes dos alunos e *notas.txt* com as notas dos alunos. A Listagem 4.23 mostra os dois arquivos.

Listagem 4.23: Arquivos de Exemplo para o Comando paste

```
$ cat nomes.txt  
alessandro  
pedro  
marcos  
andre  
luciana  
  
$ cat notas.txt  
10 20  
30 30  
40 10  
50 60  
NC 10
```

Vamos supor que agora eu necessite juntar os dois arquivos em colunas como na Listagem 4.24.

Listagem 4.24: Combinando Dois Arquivos com o Comando paste

```
$ paste nomes.txt notas.txt  
alessandro 10 20  
pedro 30 30  
marcos 40 10  
andre 50 60  
luciana NC 10
```

4.12 Comando rev

O comando **rev** inverte os caracteres de uma linha e envia o resultado para a saída padrão. Bastante útil para a manipulação de caracteres e programas de codificação. Ele recebe como argumento um arquivo ou uma mensagem. A Listagem 4.25 mostra a sintaxe para passar um arquivo como argumento,

Listagem 4.25: Comando para Inverter os Caracteres - rev

```
$ cat arquivo | rev
```

A Listagem 4.26 apresenta a sintaxe para passar uma frase como argumento.

Listagem 4.26: Invertendo Caracteres com rev

```
$ echo 'teste' | rev  
etset
```

4.13 Comando sort

Para ordenar arquivos utilizamos o comando **sort**. Seja um arquivo *teste.txt* com os seguintes itens na Listagem 4.27.

Listagem 4.27: Ordenando Arquivos com sort

```
abacate
maca
pera
uva
graviola
limao
mamao
```

Para ordenar o arquivo *teste.txt* vamos utilizar o comando **sort** na Listagem 4.28.

Listagem 4.28: Ordenando Arquivos com sort

```
$ cat teste.txt | sort -n
abacate
graviola
limao
maca
mamao
pera
uva
```

4.14 Comando tail

O comando **tail** é utilizado para imprimir as n linhas finais de um arquivo. Imagine um arquivo com o nome de *cidades.txt* com todas as cidades do Brasil. A Listagem 4.29 apresenta a lista impressa do arquivo *cidades.txt*.

Listagem 4.29: Imprime as Linhas Finais de um Arquivo

```
$ tail cidades.txt
Xangri-la (RS)
Xanxere (SC)
Xapuri (AC)
Xavantina (SC)
Xaxim (SC)
Xexeu (PE)
Xinguara (PA)
Xiique-Xique (BA)
Zabele (PB)
Zacarias (SP)
```

Para imprimir as N linhas finais utilize o comando da Listagem 4.30. Neste exemplo iremos imprimir as duas linhas finais.

Listagem 4.30: Imprime as Duas Linhas Finais de um Arquivo

```
$ tail -2 cidades.txt
Zabele (PB)
Zacarias (SP)
```

4.15 Comando uniq

O comando **uniq** é usado para encontrar linhas únicas num arquivos, i.e., ele remove linhas duplicadas consecutivas contidas em arquivos. É importante que o arquivo já esteja organizado para que ele possa remover todas as duplicações. Geralmente esse comando trabalha em conjunto com o sort. Ele possui as seguintes opções :

- -c : conta quantas vezes cada linha apareceu
- -u: imprime somente as linhas únicas
- -d: imprime somente linhas duplicadas

Vamos supor que você quer saber quantas palavras distintas existem em uma lista de palavras. Vamos utilizar a combinação dos seguintes comandos:

- sort : ordena as palavras
- uniq: retira frases com palavras iguais
- wc: conta as palavras

A Listagem 4.31 ilustra o comando

Listagem 4.31: Comando uniq

```
$ sort /usr/share/dict/words | uniq | wc -w  
99171
```

Caso deseje contar as palavras em um texto, por exemplo, em Dom Casmurro de Machado de Assis. Vamos utilizar os comandos vistos anteriormente para realizar esta tarefa. Primeiramente vamos substituir todas maiúsculas por minúsculas, em seguida vamos remover todas os caracteres que não estiverem entre a-z e também não forem espaço (espaço em branco, tabulação, quebra de linha). Feito isso, iremos substituir todo espaço por quebra de linha, em seguida ordenar as palavras, contabilizar apenas uma ocorrência de cada palavra e por fim contar quantas linhas foram geradas, ou seja, quantas são as palavras (tipos) utilizadas no texto. O código utilizado está ilustrado na Listagem 4.32.

Listagem 4.32: Contando quantas palavras distintas existem em um texto

```
$ cat dom_casmurro.txt | tr 'A-Z' 'a-z' | tr -dc 'a-z[:space:]' | tr [:space:] '\n' |  
sort | uniq | wc -l  
9125
```

4.16 Contar Número de Caracteres - Comando wc

O comando **wc** conta o número de linhas, palavras e bytes (ou caracteres) de um arquivo. A Listagem 4.33 apresenta o uso do comando processando o arquivo *cidades.txt*.

Listagem 4.33: Contando o Número de Caracteres

```
$ wc cidades.txt  
5593 15850 101042 cidades.txt
```

O comando imprime o número de linhas (5593), palavras (15850) e caracteres (101042).

Para contar apenas o número de linhas utilize o comando **wc -l** como na Listagem 4.34.

Listagem 4.34: Contando o Número de Linhas

```
$ wc -l cidades.txt  
5593
```

Para contar o número de palavras utilize o comando **wc -w** como na Listagem 4.35.

Listagem 4.35: Contando o Número de Palavras

```
$ wc -w cidades.txt  
15850 cidades.txt
```

Para contar o número de bytes utilize o comando **wc -c** como na Listagem 4.36, ou utilize, **wc -m** para contar o número de caracteres.

Listagem 4.36: Contando o Número de Bytes

```
$ wc -c cidades.txt  
101042 cidades.txt
```

Capítulo 5 | Comandos de Sistema

5.1 Gerando Todos os Comandos

Se você deseja saber todos os comandos incluídos em sua distribuição basta digitar o comando **compgen**. Para isto utilize a opção **-c** como na Listagem 5.1.

Listagem 5.1: Opções Múltiplas

```
$ compgen -c
if
then
else
elif
fi
...
xdvi-xaw
xdvipdfmx
xelatex
xetex
xindy
xindy.run
xmltex
```

5.2 Quem sou eu e onde estou?

Se você tem problemas de múltiplas personalidades esta é uma boa opção. Brincadeiras a parte, estes comandos são muito importantes no uso diário.

- Quem sou eu? Para saber quem é você (seu username) utilize o comando **whoami**. Este comando é muito utilizado para saber com que usuário você está logado. Às vezes fica muito confuso quando somos o superusuário, ou quando utilizamos mais de um usuário no sistema.
- Onde estou? Para saber o local em que você se encontra na árvore de diretórios (diretório de trabalho ou diretório corrente) use o comando **pwd**.

A Listagem 5.2 apresenta o resultado dos comandos.

Listagem 5.2: Comando whoami e pwd

```
$ whoami
pedro
$ pwd
```

5.2.1 Arquivo passwd

Quando cadastramos um usuário no sistema Linux é criado uma entrada no arquivo */etc/passwd*. Neste arquivo ficam armazenados todos os logins e algumas informações sobre os usuários do Linux. Quer listar o conteúdo do */etc/passwd* basta utilizar o comando **cat**. A Listagem 5.3 apresenta o resultado do comando.

Listagem 5.3: Arquivo passwd

```
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
messagebus:x:102:105::/var/run/dbus:/bin/false
colord:x:103:108:colord colour management daemon,,,:/var/lib/colord:/bin/false
lightdm:x:104:111:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:105:114::/nonexistent:/bin/false
avahi-autoipd:x:106:117:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:107:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
usbmux:x:108:46:usbmux daemon,,,:/home/usbmux:/bin/false
kernoops:x:109:65534:Kernel Oops Tracking Daemon,,,:/bin/false
pulse:x:110:119:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:111:122:RealtimeKit,,,:/proc:/bin/false
speech-dispatcher:x:112:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/sh
hplip:x:113:7:HPLIP system user,,,:/var/run/hplip:/bin/false
saned:x:114:123::/home/saned:/bin/false
vivas:x:1000:1000:vivas,,,:/home/vivas:/bin/bash
vboxadd:x:999:1::/var/run/vboxadd:/bin/false
sshd:x:115:65534::/var/run/sshd:/usr/sbin/nologin
```

Note que cada linha do arquivo */etc/passwd* contém 7 campos separados por dois-pontos (:) para fornecer algumas informações sobre os usuários do sistema. Em ordem, os campos são

1. usuário;
2. senha (o caractere x é utilizado para informar que a senha está encriptada, neste caso, armazenada em */etc/shadow*);
3. id do usuário (zero é reservado para o administrador e os números de 1 a 99 são reservados para algumas contas pré-definidas);
4. id do grupo primários (veja os grupos no arquivo */etc/group*);
5. informação sobre o usuário;

6. localização do diretório *home* do usuário;
7. caminho absoluto para um comando ou shell (*/bin/bash*)

5.3 Comando id

Quando se cria um usuário, **login**, tanto no Linux quanto no Mac, ele recebe um identificador numérico (inteiro) indicando o número do usuário no sistema. É como se fosse o CPF do usuário no sistema e qualquer manipulação será realizada em cima do número do usuário e não do nome. Como todos sabem computadores são melhores em manipulação de números, índices, do que realizar operações em nomes.

O comando **id** faz a relação entre usuários e identificadores de usuários. A Listagem 5.4 e 5.5 apresenta o resultado do comando para Linux e Mac respectivamente. Como pode ser observado meu login no Mac tem identificador 501 e no Linux 1000.

Listagem 5.4: Identificadores no Mac

```
$ id
uid=501(alessandrovivas) gid=20(staff) groups=20(staff),402(com.apple.sharepoint.group.1)
,12(everyone),61(localaccounts),79(_appserverusr),80(admin),81(_appserveradm),98(
_lpadmin),33(_appstore),100(_lpoperator),204(_developer),398(com.apple.
access_screensharing),399(com.apple.access_ssh)uid=501(alessandrovivas) gid=20(staff)
groups=20(staff),402(com.apple.sharepoint.group.1),12(everyone),61(localaccounts)
,79(_appserverusr),80(admin),81(_appserveradm),98(_lpadmin),33(_appstore),100(
_lpoperator),204(_developer),398(com.apple.access_screensharing),399(com.apple.
access_ssh)
```

Listagem 5.5: Identificadores no Linux

```
uid=1000(vivas) gid=1000(vivas) grupos=1000(vivas),4(adm),24(cdrom),27(sudo),30(dip),46(
plugdev),109(lpadmin),124(sambashare)
```

5.4 Alterando a Senha

O comando **passwd** permite a alteração da senha pelo usuário a qualquer tempo. A Listagem 5.6 apresenta o procedimento para alteração da senha.

Listagem 5.6: Alterando a Senha

```
$ passwd
Changing password for maria
# digite a sua senha atual
(current) UNIX password:
# entre com a sua nova senha
Enter new UNIX password:
# digite novamente sua senha
Retype new UNIX password:
passwd: password updated successfully
```

Será necessário fornecer a senha atual para que o sistema autorize a mudança. A senha deve ser digitada e re-digitada para que a mudança seja efetuada.

5.4.1 Usuários Logados

Quer descobrir quais são os usuários que estão logados nos sistema? Utilize o comando **users** da Listagem 5.7.

Listagem 5.7: Usuários Logados

```
$ users  
vivas vivas
```

5.5 Comando Date

O comando **date** exibe a hora e data do sistema. A sintaxe do comando é igual para o Linux e o Mac. A Listagem 5.8 apresenta o resultado do comando **date**.

Listagem 5.8: Visualizando Data e hora

```
$ date  
Qua 27 Nov 2013 07:31:52 BRST
```

5.6 Comando cal

Exibe um calendário em formato texto no terminal. Pode-se definir o ano e nesse caso todos os meses serão apresentados. Pode-se definir o mês específico. Se nenhum parâmetro for passado, o mês atual é exibido. A Listagem 5.9 apresenta o comando **cal**.

Listagem 5.9: Comando cal

```
Agosto 2015  
Do Se Te Qu Qu Se Sa  
1  
2 3 4 5 6 7 8  
9 10 11 12 13 14 15  
16 17 18 19 20 21 22  
23 24 25 26 27 28 29  
30 31
```

5.7 Comando finger

O comando **finger** fornece informações sobre os usuários cadastradas no sistema. Dentre essas informações estão: nome, login, diretório inicial, último login efetuado com sucesso, shell de uso. Ao se executar o comando **finger** sem argumentos, ele exibe num formato padrão de informação. Utilize o argumento **-l** para listar informações de todos os usuários logados na máquina naquele momento. A Listagem 5.11 apresenta o resultado para o sistema Mac, mas é similar ao Linux. Se o comando não estiver instalado proceda a instalação do mesmo, Listagem 5.10.

Listagem 5.10: Instalação do finger

```
$ sudo apt-get install finger  
Password:
```

Para visualizar informações sobre um usuários específico utilize a Listagem 5.11.

Listagem 5.11: Comando finger

```
$ finger vivas
Login: alessandrovivas      Name: Alessandro Vivas
Directory: /Users/alessandrovivas   Shell: /bin/bash
On since Ter 26 Nov 18:19 (BRST) on console, idle 1 day 13:03 (messages off)
On since Qui 28 Nov 07:22 (BRST) on ttys000
No Mail.
No Plan.
```

Para apresentar todos os usuários que estão logados naquele momento é usado **finger -l**. A Listagem 5.12 do resultado no mac.

Listagem 5.12: Comando finger -l

```
$ finger -l
Login: alessandrovivas      Name: Alessandro Vivas
Directory: /Users/alessandrovivas   Shell: /bin/bash
On since Ter 26 Nov 18:19 (BRST) on console, idle 1 day 13:03 (messages off)
On since Qui 28 Nov 07:22 (BRST) on ttys000
No Mail.
No Plan.
```

Outra variação é utilizar o comando sem argumentos, isto é, digitando apenas **finger**. A coluna Login é o nome do login do usuário, a coluna Name é o nome completo do usuário, Tty é o terminal onde o usuário está logado, Idle mostra o tempo ocioso, Login Time mostra a data e a hora quando o usuário logou, Office mostra a localização física do usuário e Office Phone mostra o telefone do usuário. A Listagem 5.13 mostra o resultado do comando no Linux

Listagem 5.13: Comando finger no Linux

Login	Name	Tty	Idle	Login Time	Office	Office Phone
leoca	leoca	*:0		Oct 29 11:51 (:0)		
leoca	leoca	pts/0	1	Oct 29 11:52 (:0)		
leoca	leoca	pts/1	25	Oct 29 11:52 (:0)		
leoca	leoca	pts/2	1	Oct 29 12:13 (:0)		
leoca	leoca	pts/3		Oct 29 12:17 (:0)		

5.8 Comando free

O comando **free** mostra a estatística de uso de memória, incluindo memória livre total, memória utilizada, memória física, memória swap, memória compartilhada e buffers utilizados pelo kernel. As mesmas informações podem ser encontradas no arquivo meminfo num formato ligeiramente diferente. A Listagem 5.15 apresenta o resultado deste comando para o Sistema Operacional Linux. No Mac pode-se utilizar o comando **vm_stat** apresentado na Listagem 5.14.

Listagem 5.14: Comando free

```
Mach Virtual Memory Statistics: (page size of 4096 bytes)
Pages free:                      2151369.
Pages active:                     796598.
Pages inactive:                  127372.
Pages speculative:                411792.
```

```
Pages throttled: 0.
Pages wired down: 706471.
Pages purgeable: 72490.
:
Pages copy-on-write: 3674787.
Pages zero filled: 147901.
Pages reactivated: 2462641.
Pages purged: 10.
File-backed pages: 0.
Anonymous pages: 620935.
Pages stored in compressor: 714827.
Pages occupied by compressor: 0.
Decompressions: 0.
Compressions: 0.
Pageins: 151389.
Pageouts: 0.
Swapins: 0.
Swapouts: 0.
```

Listagem 5.15: Comando free

	total	used	free	shared	buffers	cached
Mem:	2061712	490924	1570788	0	60984	220236
-/+ buffers/cache:		209704	1852008			
Swap:	587768	0	587768			

5.9 Comando su

Executa o interpretador de comandos com a substituição do usuário e do grupo. Possibilidade de logar imediatamente no mesmo terminal em uso com outro usuário. Prática comum de super-usuário. A Listagem 5.16 apresenta a execução do comando **su**.

Listagem 5.16: Logar como Super Usuário

```
$ su -
Password:
```

5.10 Comando uname

O comando **uname** é utilizado para apresentar informações sobre o sistema operacional de sua máquina. A Listagem 5.17 apresenta o comando, no Linux, para verificar qual sistema operacional está utilizando. A Listagem 5.18 ilustra o resultado para o Mac.

Listagem 5.17: Verificar Informações sobre o Linux

```
uname -s
Linux
```

Listagem 5.18: Exemplo do Sistema Operacional Mac

```
$ uname -s
```

Para verificar a versão do seu kernel utilize o comando apresentando na Listagem 5.19.

Listagem 5.19: Verificar sua Versão do kernel

```
$ uname -r  
3.8.0-33-generic
```

A Listagem 5.20 apresenta o comando para verificar se sua plataforma é de 32 ou 64 bits. Neste caso a plataforma é de 64 bits, pois a resposta foi **x86_64**.

Listagem 5.20: Verificando a Plataforma

```
$ uname -m  
x86_64
```

Para descobrir o nome de sua máquina utilize o comando **uname -n**. A Listagem 5.21 apresenta o resultado.

Listagem 5.21: Verificar o Nome de sua Máquina

```
$ uname -n  
musashi
```

Para apresentar todas as informações sobre seu sistema operacional utilize o comando **uname -a**. A Listagem 5.22 apresenta o resultado.

Listagem 5.22: Apresenta todas as informações sobre seu sistema operacional

```
$ uname -a  
Linux vivas-VirtualBox 3.8.0-33-generic #48~precise1-Ubuntu SMP Thu Oct 24 16:28:06 UTC  
2013 x86_64 x86_64 x86_64 GNU/Linux
```

5.11 Comando **uptime**

O comando **uptime** apresenta as seguintes informações: a hora corrente, há quanto tempo o seu computador está ligado, quantidade de usuários logados e a carga média do sistema a 1, 5 e 15 minutos passados. A Listagem 5.23 ilustra o resultado do comando.

Listagem 5.23: Tempo de Funcionamento

```
15:24 up 52 mins, 2 users, load averages: 1,72 1,65 1,59
```

5.12 Verificando a versão de um comando

Os comandos são programas e apresentam versões. Se quiser verificar a versão de um comando utilize a opção *version* como na Listagem 5.24.

Listagem 5.24: Versão de um comando

```
$ ls --version  
ls (GNU coreutils) 8.13
```

Copyright (C) 2011 Free Software Foundation, Inc.
Licença GPLv3+:: GNU GPL versão 3 ou posterior <<http://gnu.org/licenses/gpl.html>>
Este é um software livre: você é livre para alterá-lo e redistribuí-lo
NAO HA GARANTIA, na máxima extensão permitida pela lei.

Escrito por Richard M. Stallman e David MacKenzie.

5.13 Variável PATH

Ao se digitar um comando, o arquivo deve ser localizado pelo sistema operacional para ser executado. Se ele não o encontra uma mensagem de erro é exibida em seguida. Algumas vezes, quando criamos, por exemplo, arquivos executáveis, necessitamos passar o local do arquivo como por exemplo:

Listagem 5.25: Localização de um Comando

```
./arquivo_executavel  
./usr/bin/arquivo_executavel
```

Acima, o ponto indica o caminho desde o diretório raiz até o diretório corrente. Entretanto, existem locais padrões a serem buscados e tais locais são definidos por uma variável chamada PATH.

Quando você digita um comando e o shell não encontra, pode estar acontecendo duas coisas: o comando não foi instalado ou o seu shell não está procurando no local correto. Para saber todos os caminhos onde seu shell procura os comandos digite:

Listagem 5.26: Variável PATH

```
$ echo $PATH  
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games
```

Os caminhos são separados por dois-pontos (:). No exemplo acima o primeiro local onde o shell procura os comandos é o diretório /usr/local/bin. Como pode ser observado, o shell não procura diretamente em seu diretório. Você pode imaginar o motivo? Razões de segurança: se um intruso inserisse um programa malicioso(ex. uma modificação do comando ls) e se o shell procurasse diretamente no seu diretório, a execução desse programa poderia danificar seus arquivos.

Os caminhos são separados por dois-pontos (:). No exemplo acima o primeiro local onde o shell procura os comandos é o diretório /usr/bin ou /usr/local/bin.

5.14 Comando w

O comando w verifica quais usuários estão logados e o que eles estão fazendo. A Listagem 5.27 apresenta o resultado do comando.

Listagem 5.27: Comando w

```
$ w  
7:44 up 10 days, 16:57, 3 users, load averages: 2,21 2,01 2,26  
USER TTY FROM LOGIN@ IDLE WHAT  
alessandrovivas console - 24Ago15 10days -  
alessandrovivas s000 - 24Ago15 15:47 -bash  
alessandrovivas s001 - 26Ago15 - w
```

5.15 Comando whereis

O comando **whereis** determina a localização de seu programa executável(binário), fonte e páginas de manual referente a um comando. Ele é mais completo que o comando **which**.

Se quiser encontrar a localização de um programa, por exemplo o **gcc**, utilizamos o exemplo na Listagem 5.28.

Listagem 5.28: Comando whereis

```
$ whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/bin/X11/gcc /usr/share/man/man1/gcc.1.gz
```

5.16 Comando locate

O comando **locate** lista arquivos que contenham o texto dado (semelhante ao whereis). A Listagem 5.29 utiliza o comando para listar todas as ocorrências de **passwd**.

Listagem 5.29: Comando locate

```
$ locate passwd
/etc/passwd
/etc/passwd-
/etc/cron.daily/passwd
/etc/init/passwd.conf
/etc/init.d/passwd
/etc/pam.d/chpasswd
/etc/pam.d/passwd
/etc/security/opasswd
/usr/bin/gpasswd
/usr/bin/grub-mkpasswd-pbkdf2
/usr/bin/lppasswd
/usr/bin/mkpasswd
/usr/bin/passwd
/usr/bin/smbpasswd
/usr/bin/vino-passwd
...
...
```

5.17 Comando which

Quer descobrir se um programa está no seu PATH? Utilize o comando **which** para resolver este problema como ilustrado na Listagem 5.30.

Listagem 5.30: Comando which

```
$ which ls
/bin/ls
$ which gcc
/usr/bin/gcc
```

5.18 Comando whatis

O comando **whatis** pode ser utilizado para busca no banco de dados do sistema sobre informações de comandos como na Listagem 5.31.

Listagem 5.31: Comando whatis

```
vivas@zafu:~$ whatis clear
clear (1)           - clear the terminal screen
vivas@zafu:~$ whatis ps
ps (1)             - report a snapshot of the current processes.
vivas@zafu:~$ whatis netstat
netstat (8)         - Mostra conexoes de rede, tabelas de roteamento, estatisticas de
                      interface e conexoes...
vivas@zafu:~$ whatis route
route (8)           - mostra / manipula a tabela de roteamento IP
vivas@zafu:~$ whatis ls
ls (1)              - list directory contents
LS (6)              - display animations aimed to correct users who accidentally enter
                      LS instead of ls .
vivas@zafu:~$ whatis touch
touch (1)            - change file timestamps
vivas@zafu:~$ whatis cat
cat (1)              - concatenate files and print on the standard output
vivas@zafu:~$ whatis date
date (1)             - print or set the system date and time
```

5.19 Rodando múltiplos comandos

Para rodar múltiplos comandos basta separar com ponto e vírgula. A Listagem 5.32 apresenta o resultado do comando.

Listagem 5.32: Rodando Múltiplos Comandos

```
$ hostname; date; ls;
zafu
3Sex Jan  3 17:22:40 BRST 2014
  Area de Trabalho  Docs          Downloads        ifconfig1-modificado  Musica    teste2.txt
          ubuntu
  arq1-hard       Docs2         examples.desktop  Imagens           Publico   teste3.txt
          Videos
  arq2           Documentos     ifconfig1        Modelos           teste    teste.txt
```

O exemplo ilustrado na Listagem 5.32 mostra como rodar comandos em sequência, de forma que eles sempre serão executados, independente do resultado da execução do comando anterior. Uma outra forma é condicionar a execução do comando subsequente à execução do primeiro comando. Podemos utilizar o `&&` para executar o segundo comando apenas se o primeiro comando tiver sucesso. Outra opção é condicionar a execução do segundo ao fracasso do primeiro comando, ou seja, o segundo só será executado se o primeiro retornar erro. Para este fim utilizaremos o `||`. Ambos exemplos são ilustrados na Listagem 5.33.

Listagem 5.33: Rodando Múltiplos Comandos Condicionados

```
$ test -f teste.existe && echo 'existe'
$ test -f teste.existe || echo 'existe'
3$ touch teste.existe && test -f teste.existe && echo 'existe'
```

Note que o primeiro só irá imprimir ‘existe’ na tela se o arquivo *teste.existe* já existir no diretório corrente. Caso o arquivo não exista, apenas a terceira linha imprimirá ‘existe’ na tela, umas vez que o arquivo acaba de ser criado pelo comando **touch**.

5.20 Rodando um comando em background

Outra forma muito útil de executar programas é executá-los em background. Isto implica que o programa estará rodando, mas você não verá os resultados e não poderá passar sinais ao programa através do teclado. Isto implicará que o programa não poderá ser finalizado utilizando a combinação de teclas <Ctrl-C>. Ao executar um programa em background, o shell ficará livre para a execução de outros programas. Para executar um programa em background, você deverá utilizar o & após chamar o programa. Veja o exemplo na Listagem 5.34, no qual o **emacs** será executado em background.

Listagem 5.34: Rodando Comando em Background

```
$ emacs &
```

Capítulo 6 | Gerenciamento de Processos

Este capítulo tem como objetivo apresentar os principais comandos para manipulação de processos no Sistema Operacional Linux.

6.1 Visualizando Todos os Processos em Execução

Definimos processo como um programa em execução. Podemos definir como espaço de endereçamento de um processo como: código do programa, variáveis utilizadas, pilha do processo e outras informações necessárias.

Os processos são constituídos da localização do espaço de endereçamento, status, prioridade de execução, informações sobre os recursos utilizados, máscara de sinal do processo e identificação do proprietário. Os principais atributos são: identificador do processo (PID), Identificador do processo pai (PPID), prioridade de execução (nice), TTY (terminal), identificação real do usuário e do grupo.

O comando **ps** com a opção **-aef** apresenta todos os processos em execução como na Listagem 6.1.

Listagem 6.1: Todos os Processos em Execução

```
$ ps -aef
UID  PID  PPID  C STIME   TTY          TIME CMD
 0    1      0  0 Seg02 ??          11:25.68 /sbin/launchd
 0    45     1  0 Seg02 ??          1:36.14 /usr/sbin/syslogd
 0    46     1  0 Seg02 ??          0:48.74 /usr/libexec/UserEventAgent (System)
 0    48     1  0 Seg02 ??          0:04.10 /usr/libexec/kextd
...
...
```

6.2 Todos os Processos de um Usuário Específico

O comando **ps** com a opção **-u** pode ser utilizado para visualizar todos os processos de um determinado usuário como na Listagem 6.2.

Listagem 6.2: Todos os Processos em Execução de um Usuário Específico

```
$ ps -u alessandrovivas
UID  PID TTY          TIME CMD
501  287 ??          0:15.25 /usr/libexec/UserEventAgent (Aqua)
501  289 ??          3:17.49 /usr/sbin/distnoted agent
501  290 ??          0:06.69 /usr/sbin/universalaccessd launchd -s
501  291 ??          0:14.95 /usr/sbin/cfprefsd agent
501  302 ??          0:08.25 /usr/sbin/usernoted
501  303 ??          0:04.27 /usr/libexec/nsurlsessiond
```

6.3 Lista de Processos Ordenadas pelo Consumo de CPU

Para obter uma lista de processos ordenadas pelo consumo de CPU utilizamos a Listagem 6.3. Neste caso utilizamos o comando **head** para imprimir somente as 5 primeiros ocorrências.

Listagem 6.3: Processos que Mais Consomem CPU

```
$ ps -aef -r | head -5
UID  PID  PPID  C STIME   TTY           TIME CMD
202  339     1  0 Seg02   ??        129:30.76 /usr/sbin/coreaudiod
501  517     1  0 Seg02   ??        5:50.05 /Applications/Utilities/Terminal.app/
    Contents/MacOS/Terminal -psn_0_61455
501 1113     1  0 Seg03   ??        73:46.47 /Applications/iTunes.app/Contents/MacOS/
iTunes
```

6.4 Lista dos Processos que Mais Consomem Memória

Para obter uma lista de processos ordenadas pelo consumo de memória utilizamos a Listagem 6.4. Neste caso utilizamos o comando **head** para imprimir somente as 5 primeiros ocorrências.

Listagem 6.4: Processos que Mais Consomem CPU

```
$ ps -aef -m | head -5
UID  PID  PPID  C STIME   TTY           TIME CMD
200  248     1  0 Seg02   ??        2:38.10 /System/Library/CoreServices/Software
    Update.app/Contents/Resources/softwareupdated
  0  259     1  0 Seg02   ??        6:15.90 /System/Library/Frameworks/CoreServices.
    framework/Frameworks/Metadata.framework/Versions/A/Support/mds_stores
501  521     1  0 Seg02   ??        13:19.81 /System/Library/CoreServices/Finder.app/
    Contents/MacOS/Finder
501 1113     1  0 Seg03   ??        73:52.45 /Applications/iTunes.app/Contents/MacOS/
iTunes
```

6.5 Obtendo Informações de um Processo Específico

Para visualizar informações de um processo específico utilizamos o comando **ps** em conjunto com o comando **grep** como na Listagem 6.5.

Listagem 6.5: Obtendo Informações de um Processo Específico

```
$ ps -aef | grep texmaker
501 61850     1  0  4:05   ??        1:28.48 /Applications/texmaker.app/Contents/
    MacOS/texmaker
501 62417  35019  0  4:27   ttys001  0:00.00 grep texmaker
```

6.6 Comando pstree

O comando **pstree** apresenta todos os comandos em execução no formato de uma árvore relacionando a dependência entre eles. A Listagem 6.6 apresenta o resultado do comando. Neste exemplo utilizamos o comando **head** apenas para limitar o número de linhas.

Listagem 6.6: Obtendo a Lista de Processos em forma de Árvore

```
$ pstree | head -15
systemd--ModemManager---{gdbus}
|           '-{gmain}
|-NetworkManager---2*[dhclient]
|           |-dnsmasq
|           |-{NetworkManager}
|           |-{gdbus}
|           '-{gmain}
|-accounts-daemon---{gdbus}
|           '-{gmain}
|-acpid
|-2*[agetty]
|-avahi-daemon---avahi-daemon
|-chrome---2*[cat]
|           |-chrome---chrome---chrome---{Chrome_ChildIOT}
|           |           |           |           |-2*[{CompositorTileW}]
```

6.7 Comando top

O comando **top** é utilizado para obter informações sobre os processos que estão rodando em sua máquina. A Listagem 6.7 apresenta o resultado do comando.

Os estados possíveis de um processo são:

- runnable - rodando
- sleeping - está esperando por um evento
- swapped - não está executando e foi armazenado na memória virtual
- zombie - está tentando morrer (pode ter perdido seu pai)
- stopped - está proibido de executar (através de CTRL-Z ou um SIGSTOP)

Listagem 6.7: Comando top

```
$ top
Processes: 274 total, 3 running, 7 stuck, 264 sleeping, 2185 threads                                         08:20:18
Load Avg: 2.62, 2.52, 2.29 CPU usage: 13.1% user, 6.98% sys, 80.0% idle
SharedLibs: 18M resident, 19M data, 0B linkedit.
MemRegions: 59765 total, 4041M resident, 160M private, 1656M shared.
PhysMem: 11G used (2077M wired), 5188M unused. VM: 651G vszie, 1064M framework vszie, 0(0) swapins, 0(0) swapouts.
Networks: packets: 8319390/4990M in, 6708488/3710M out. Disks: 1263551/27G read, 1420382/50G written.

PID  COMMAND %CPU   TIME #TH #WQ #PORT MEM PURG CMPR PGRP PPID STATE BOOSTS
80264 com.apple.ap 0.0 00:04.97 3 0 210 27M 20K 0B 80264 1 sleeping *0[128]
80255 com.apple.Co 0.0 00:00.01 2 1 21 760K 0B 0B 80255 1 sleeping 0[17]
80246 com.apple.hi 0.0 00:00.01 2 0 26 972K 0B 0B 80246 1 sleeping 0[3]
80236 Preview 0.2 00:23.67 4 0 299- 72M- 5128K 0B 80236 1 sleeping *0[107]
80232 QuickLookUIH 0.0 00:00.25 2 0 126 10M 0B 0B 80232 1 sleeping *0[1]
80212 com.apple.hi 0.0 00:00.01 2 0 26 968K 0B 0B 80212 1 sleeping 0[4]
80206 com.apple.ap 0.0 00:02.13 3 0 203 17M 12K 0B 80206 1 sleeping *0[545]
80205 CVMCompiler 0.0 00:00.63 2 1 31 17M 16K 0B 80205 1 sleeping *0[1]
80204 com.apple.Co 0.0 00:00.01 2 1 21 748K 0B 0B 80204 1 sleeping 0[34]
80202 Paint X Lite 0.1 00:11.27 6 0 232 96M 10M 0B 80202 1 sleeping *0[118]
80118 com.apple.ap 0.0 00:12.34 3 0 205 32M 20K 0B 80118 1 sleeping *0[84]
80117 com.apple.hi 0.0 00:00.01 2 0 26 964K 0B 0B 80117 1 sleeping 0[4]
80110 SketchBookPr 0.0 00:12.05 6 0 252 154M 5216K 0B 80110 1 sleeping *0[207]
80049 com.apple.se 0.0 00:00.20 3 0 141- 6668K- 0B 0B 80049 1 sleeping 0[77]
...
```

6.8 Listando todos os Sinais com o Comando kill

O comando **kill** é utilizado para enviar sinais para um processo. A Listagem 6.8 apresenta todos os sinais disponíveis.

Listagem 6.8: Comando kill

```
$ kill -l
 1) SIGHUP    2) SIGINT    3) SIGQUIT   4) SIGILL
 5) SIGTRAP   6) SIGABRT   7) SIGEMT    8) SIGFPE
 9) SIGKILL   10) SIGBUS   11) SIGSEGV  12) SIGSYS
13) SIGPIPE   14) SIGALRM   15) SIGTERM   16) SIGURG
17) SIGSTOP   18) SIGTSTP   19) SIGCONT   20) SIGCHLD
21) SIGTTIN   22) SIGTTOU   23) SIGIO    24) SIGXCPU
25) SIGXFSZ   26) SIGVTALRM 27) SIGPROF   28) SIGWINCH
29) SIGINFO   30) SIGUSR1   31) SIGUSR2
```

6.9 Matando um Processo com o Comando Kill

Para eliminar um processo com o comando **kill** precisamos saber o PID do processo. Vamos supor que você deseja eliminar o processo do firefox. O primeiro passo é saber o PID do firefox e para isto faça o procedimento da Listagem 6.9.

Listagem 6.9: Comando kill

```
$ ps -aef | grep firefox
 501 79667      1  0  7:52    ??        19:08.19 /Applications/Firefox.app/Contents/MacOS
 /firefox -foreground
```

Como podemos perceber o PID do firefox é 79667. Agora para matar o processo firefox enviamos um sinal SIGKILL, número 9, como na Listagem 6.10.

Listagem 6.10: Matando o Processo Firefox

```
$ kill -9 79667
```

6.10 Comando time

O comando **time** é utilizado para mostrar o tempo de execução de um script ou processo. Este comando calcula o tempo utilizado pelo programa no modo usuário, no modo kernel e a quantidade de tempo realmente utilizado. A quantidade de tempo total utilizado é sempre maior do que a soma do tempo no modo Kernel mais o tempo no modo usuário. Ele envolve todos os tempos: processamento de interrupção e tempo de espera na fila de processos prontos ou aguardando o processamento de entrada e saída. A Listagem 6.11 calcula o tempo gasto pelo comando date. Neste caso ele gastou 2 ms de tempo no modo kernel, 1 ms no modo usuário e o tempo total de 4 ms.

Listagem 6.11: Calculando Tempo de Execução de um Programa ou Script

```
time date
Sex 4 Set 2015 13:54:02 BRT
```

```
real 0m0.004s
user 0m0.001s
sys 0m0.002s
```

Vamos dar o usar o comando da Listagem 6.11 em um notebook rodando Fedora. Os dois tem processadores semelhantes, mas o primeiro roda Mac OSX e o segundo Fedora. O resultado do comando é apresentado na Listagem 6.12. Como pode ser visualizado o notebook rodando Fedora teve um desempenho melhor do que o Mac OSX.

Listagem 6.12: Calculando Tempo de Execução de um Programa ou Script

```
time date
Sun Sep 4 14:13:13 BRT 2015

real 0m0.002s
user 0m0.000s
sys 0m0.002s
```

6.11 Controlando Processos

No módulo anterior aprendemos que, para finalizar um processo, basta utilizar o comando CTRL-C. Para interromper um processo basta utilizar o comando CTRL-Z. Você sabe o que é um processo?

Processo é um programa que está rodando em seu espaço virtual de endereçamento. Ou melhor, um processo é um programa independente, em estado de execução, que tem seu próprio conjunto de recursos. Tudo que está rodando no Linux é um processo. Uma tarefa, por outro lado, pode envolver vários comandos executando em série.

Podemos definir como espaço de endereçamento: código do programa, variáveis utilizadas, pilha do processo e outras informações necessárias.

Os processos são constituídos da localização do espaço de endereçamento, status, prioridade de execução, informações sobre os recursos utilizados, máscara de sinal do processo e identificação do proprietário. Os principais atributos são: identificador do processo (PID), Identificador do processo pai (PPID), prioridade de execução (nice), TTY (terminal), identificação real do usuário e do grupo.

Os processos podem ser criados utilizando a rotina fork. Com o fork você cria uma cópia idêntica ao processo pai, mas com outro PID.

O sistema operacional linux tem vários tipos de processos:

- Processos interativos: é um processo inicializado (e controlado por) um Shell. Um processo interativo pode estar em background ou foreground.
- Processos batch: é um processo que não está associado a um terminal, mas é submetido a uma fila para ser executado seqüencialmente.
- Processos daemon : é um processo que fica rodando em background até ser requisitado. Este tipo de processo é usualmente gerado no processo de inicialização da máquina

Um programa no Linux pode ser executado de duas formas: primeiro plano (foreground) e segundo plano (background). Ao executar no primeiro plano devemos esperar o término da execução do comando para entrar com um novo comando, somente é mostrado o aviso de comando após o término de execução do comando/programa. Isto fica bem claro quando executamos o comando ls, somente entramos com um novo comando quando ele termina sua execução; isto é a forma usual de execução de comandos no Linux. Quando trabalhamos com a execução em segundo plano não precisamos esperar o término da execução de um programa para executar um novo comando. O comando fica sendo executado internamente e ao terminar ele mostra uma mensagem de pronto acompanhado do número PID do processo que terminou.

Para colocar um processo rodando em segundo plano colocamos o modificador "&".

Listagem 6.13: Controlando Processos

```
# entre como root
$ find / -name boot.b &
[1] 7998
/boot/boot.b
```

O comando para listar os processos que estão rodando em sua máquina é o **ps** (process status). Este comando pode ser usado por todos os usuários, mas sua saída muda quando você é o root.

Listagem 6.14: Controlando Processos

```
$ ps
 PID TTY          TIME CMD
 7012 pts/0    00:00:00 sh
 7081 pts/0    00:00:00 ps
```

Este comando é organizado em colunas. A primeira coluna, PID, indica o número de identificação do processo. Todos os processos que rodam no Linux recebem um identificador (número inteiro) e para manipulação dos processos devemos utilizar este número. Este número inicia em 0 e é incrementado de 1 para cada novo processo, o número final é 65564. Quando o Linux chega ao último número, ele começa a numeração do menor número pulando os que estiverem ativos. Os processos que possuem menor número são os dos sistemas do kernel e os daemons, que iniciam quando o Linux é inicializado (boot) e continuam ativo enquanto o sistema estiver rodando.

A coluna TTY no comando **ps** mostra em qual terminal você iniciou o processo. A coluna STAT mostra o status corrente do processo, os estados podem ser:

- S processo está dormindo
- R processo está rodando.

Um processo está dormindo quando ele não está ativo. A coluna STAT não apareceu quando rodamos o comando **ps**. Um processo está rodando quando ele está ativo na CPU.

A coluna TIME mostra a quantidade de tempo da CPU que o processo está utilizando. Deve ser ressaltado que é a quantidade de tempo da CPU e não a quantidade de tempo que o processo está ativo.

A última coluna indica o nome do processo que está rodando. Este nome é usualmente o comando que você digitou. No exemplo acima, temos dois comandos o **sh** (estou acessando a máquina remotamente via "sh") e o **ps** (process status) comando que acabei de digitar.

Outro conceito importante é o parentesco entre processos. Quando um processo inicia um segundo processo, o segundo processo é chamado de processo filho.

Este comando tem várias variações e começaremos a estudá-las agora. O **ps -u** é o comando que lista os processos que estão rodando

Listagem 6.15: Controlando Processos

```
$ ps -u joao
PID TTY          TIME CMD
 7011 ?        00:00:00 sshd
 7012 pts/0    00:00:00 sh
 7295 pts/0    00:00:00 ps
$ ps u
USER      PID %CPU %MEM   VSZ   RSS TTY      STAT START   TIME COMMAND
joao      7012  0.0  1.0  2456 1304 pts/0      S    08:46   0:00 -sh
joao      7221  0.0  1.1  3288 1404 pts/0      R    09:19   0:00 ps u
```

Para matar um processo, parar ou continuar, o Linux utiliza uma forma especial de comunicação chamada de sinais. É a mesma coisa quando usamos os comandos CTRL-C e CTRL-Z. Para fazer estes testes vamos utilizar o comando **top**.

O comando top é utilizado para monitorar todos os processos que estão rodando na máquina. A cada 5 segundos (você pode especificar o número de atualizações por segundo) ele tira uma fotografia dos processos em sua máquina. Ele mostra as tarefas que mais consomem o tempo da CPU em sua máquina.

A figura abaixo mostra o resultado após a execução do comando **top**.

Listagem 6.16: Controlando Processos

```
09:35:28 up 18:48, 1 user, load average: 0.00, 0.00, 0.00
37 processes: 35 sleeping, 2 running, 0 zombie, 0 stopped
CPU states: 0.0% user, 0.0% system, 0.0% nice, 100.0% idle
Mem: 126820K total, 119912K used, 6908K free, 12264K buffers
Swap: 248968K total, 5744K used, 243224K free, 44176K cached

PID USER      PRI  NI   SIZE  RSS SHARE STAT %CPU %MEM     TIME COMMAND
 1 root       8   0    484  444    424 S    0.0  0.3  0:00 init
 2 root       9   0     0   0     0 SW    0.0  0.0  0:00 keventd
 3 root      19  19     0   0     0 SWN   0.0  0.0  0:00 ksoftirqd_CPU0
 4 root       9   0     0   0     0 SW    0.0  0.0  0:05 kswapd
 5 root       9   0     0   0     0 SW    0.0  0.0  0:00 bdflush
 6 root       9   0     0   0     0 SW    0.0  0.0  0:01 kupdated
133 root      9   0    628  576    496 S    0.0  0.4  0:00 dhclient-2.2.x
208 root      9   0    600  596    492 S    0.0  0.4  0:02 syslogd
211 root      9   0   1052 1044    412 S    0.0  0.8  0:01 klogd
243 root      9   0   1244  960    804 S    0.0  0.7  0:09 nmbd
245 root      9   0   1100  788    636 S    0.0  0.6  0:00 smbd
342 root      9   0   1120 1052    944 S    0.0  0.8  0:00 sshd
352 root      9   0    872  832    696 S    0.0  0.6  0:00 xinetd
355 nobody    8   0   1148 1028    888 S    0.0  0.8  0:00 proftpd
358 daemon    9   0    580  556    504 S    0.0  0.4  0:00 atd
361 root      0   0    684  680    576 S    0.0  0.5  0:00 cron
365 root      9   0   1416 1372   1304 S    0.0  1.0  0:00 apache
374 root      9   0   1808 1496   1496 S    0.0  1.1  0:00 gdm
379 root      9   0   2028 1672   1672 S    0.0  1.3  0:00 gdm
380 root      5 -10 27128 7676   968 S <  0.0  6.0  0:04 XFree86
383 root      9   0    468  412    412 S    0.0  0.3  0:00 getty
384 root      9   0    468  412    412 S    0.0  0.3  0:00 getty
385 root      9   0    468  412    412 S    0.0  0.3  0:00 getty
386 root      9   0    468  412    412 S    0.0  0.3  0:00 getty
387 root      9   0    468  412    412 S    0.0  0.3  0:00 getty
394 gdm      9   0   3164 2492   2176 S    0.0  1.9  0:00 gdmlogin
408 root      9   0   2028 1832   1568 S    0.0  1.4  0:00 sendmail
```

Para sair da tela basta digitar "q".

A sintaxe deste comando é:

Listagem 6.17: Controlando Processos

```
top [-] [d delay] [q] [S] [s] [i]
```

Onde:

d Especifica o tempo de atraso entre as atualizações da tela
q Força o comando top a atualizar sem atraso
S Usa o modo cumulativo
s Roda em modo seguro desabilitando comandos interativos
i Ignora processos zombies

Vamos exercitar todos os comandos

Listagem 6.18: Controlando Processos

```
$ top -d 1
$ top -q
```

```
$ top -S  
$ top -s  
$ top -i
```

Para colocar um processo rodando em background, vamos usar o comando com o top.

Listagem 6.19: Controlando Processos

```
$ top &  
[1] 7528  
  
[1]+ Stopped(SIGTTOU) top
```

Para matar, parar ou continuar processos utilizamos os sinais. Para enviar um sinal para um processo utilizamos o comando **kill**. Como colocamos o processo top rodando em background, vamos utilizar o comando **jobs**.

Listagem 6.20: Controlando Processos

```
$ jobs -l  
[1]+ 7528 Parado (saída tty) top
```

Como podemos ver a tarefa top está parada e o pid é o número 7528. Para matar este processo usamos o comando kill mais o número do processo:

Listagem 6.21: Controlando Processos

```
$ kill -9 7528  
$ jobs -l  
[1]+ 7528 Morto top
```

Assim o processo está morto e se listarmos novamente nada vai aparecer:

Listagem 6.22: Controlando Processos

```
$ jobs -l
```

Agora vamos colocar o processo top rodando novamente:

Listagem 6.23: Controlando Processos

```
$ top  
// entre em outro terminal e liste os processos que estão rodando  
// use o comando ps -u  
$ ps -u joao  
  PID TTY      TIME CMD  
7011 ?        00:00:01 sshd  
7012 pts/0    00:00:00 sh  
7587 pts/0    00:00:00 top  
7596 ?        00:00:00 sshd  
7597 pts/1    00:00:00 sh  
7599 pts/1    00:00:00 ps  
// o número do processo top é 7487, agora vamos mandar um sinal para  
// este processo parar sua execução  
$ kill -s SIGSTOP 7587  
// entre no outro terminal e liste as tarefas que estão rodando  
$ jobs -l  
[1]+ 7587 Parado (sinal) top  
// agora vamos enviar um sinal para matá-lo, além do valor numérico  
// podemos utilizar a palavra SIGKILL acompanhada do número do  
// processo  
joao@dcomp:~$ kill -s SIGKILL 7587  
joao@dcomp:~$ jobs -l  
[1]+ 7587 Morto top
```

Assim, podemos utilizar o comando **kill** com valores numéricos ou a opção **-s** onde passamos nome do sinal a ser enviado. Outra opção é o sinal SIGHUP que faz com que o processo releia seu arquivo de configuração. O sinal SIGSTOP mantém o processo parado até ele receber o sinal SIGCONT, vamos testar isto:

Listagem 6.24: Controlando Processos

```
// entre em um terminal e digite:  
$ top  
// entre em outro terminal e liste os processos que estão rodando  
// use o comando ps -u  
$ ps -u joao  
  PID TTY      TIME CMD  
7011 ?    00:00:01 sshd  
7012 pts/0  00:00:00 sh  
7596 ?    00:00:00 sshd  
7597 pts/1  00:00:00 sh  
7660 pts/0  00:00:00 top  
7662 pts/1  00:00:00 ps  
// o número do processo top é 7660, agora vamos mandar um sinal para  
// este processo parar sua execução  
$ kill -s SIGSTOP 7660  
// entre no outro terminal e liste as tarefas que estão rodando  
$ jobs -l  
joao@dcomp:~$ jobs -l  
[1]+ 7660 Parado (sinal)          top  
// agora vamos enviar um sinal para continuar a sua execução, além do // valor numérico  
// podemos utilizar a palavra SIGCONT acompanhada do // número do processo  
joao@dcomp:~$ kill -s SIGCONT 7660
```

Se o comando funcionou normalmente, no outro terminal o **top** continuou sua execução. Agora termine o mesmo digitando "q".

Para finalizar um processo pelo nome utilizamos o comando **killall**. Tente descobrir qual o terminal você está "logado" e mande um sinal para matá-lo.

Listagem 6.25: Controlando Processos

```
$ ps  
$ killall -9
```

Vamos agora utilizar outras opções do comando **ps**. Deve ser ressaltado que o hífen neste comando não é necessário.

Tente utilizar ao opção **ps -aux**:

Listagem 6.26: Controlando Processos

```
$ ps aux  
root@dcomp:/usr/lib# ps aux  
USER      PID %CPU %MEM   VSZ   RSS TTY      STAT START   TIME COMMAND  
root        1  0.0  0.3  1272   444 ?          S Dec11  0:00 init  
root        2  0.0  0.0     0    0 ?          SW Dec11  0:00 [keventd]  
root        3  0.0  0.0     0    0 ?          SWN Dec11  0:00 [ksoftirqd_CPU0]  
root        4  0.0  0.0     0    0 ?          SW Dec11  0:05 [kswapd]  
root        5  0.0  0.0     0    0 ?          SW Dec11  0:00 [bdfflush]  
root        6  0.0  0.0     0    0 ?          SW Dec11  0:01 [kupdated]  
root      133  0.0  0.4  1468   576 ?          S Dec11  0:00 /sbin/dhclient-2.2.x -q  
eth0  
root      208  0.0  0.4  1344   596 ?          S Dec11  0:03 /sbin/syslogd  
root      211  0.0  0.8  1836  1044 ?          S Dec11  0:01 /sbin/klogd  
root      243  0.0  0.7  2772   960 ?          S Dec11  0:10 /usr/sbin/nmbd -D  
root      245  0.0  0.6  3556   788 ?          S Dec11  0:00 /usr/sbin/smbd -D  
root      342  0.0  0.8  2788  1052 ?          S Dec11  0:00 /usr/sbin/sshd
```

root	352	0.0	0.6	2056	832	?	S	Dec11	0:00	/usr/sbin/xinetd -reuse
nobody	355	0.0	0.8	2936	1028	?	S	Dec11	0:00	proftpd (accepting connections)
daemon	358	0.0	0.4	1384	556	?	S	Dec11	0:00	/usr/sbin/atd
root	361	0.0	0.5	1652	680	?	S	Dec11	0:00	/usr/sbin/cron
root	365	0.0	1.0	2932	1372	?	S	Dec11	0:00	/usr/sbin/apache
root	374	0.0	1.1	7252	1496	?	S	Dec11	0:00	/usr/bin/gdm
root	379	0.0	1.3	7336	1672	?	S	Dec11	0:00	/usr/bin/gdm
root	380	0.0	6.0	30868	7676	?	S<	Dec11	0:04	/usr/bin/X11/X :0 - deferglyphs 16 -
root	383	0.0	0.3	1256	412	tty2	S	Dec11	0:00	/sbin/getty -f /etc/issue.linuxlogo
root	384	0.0	0.3	1256	412	tty3	S	Dec11	0:00	/sbin/getty -f /etc/issue.linuxlogo
root	385	0.0	0.3	1256	412	tty4	S	Dec11	0:00	/sbin/getty -f /etc/issue.linuxlogo
root	386	0.0	0.3	1256	412	tty5	S	Dec11	0:00	/sbin/getty -f /etc/issue.linuxlogo
root	387	0.0	0.3	1256	412	tty6	S	Dec11	0:00	/sbin/getty -f /etc/issue.linuxlogo
gdm	394	0.0	1.9	8700	2492	?	S	Dec11	0:00	/usr/bin/gdmlogin -- disable-sound -
root	408	0.0	1.4	4912	1832	?	S	Dec11	0:00	sendmail: MTA: accepting connection
root	439	0.0	0.3	1256	412	tty1	S	Dec11	0:00	/sbin/getty -f /etc/issue.linuxlogo
www-data	6069	0.0	1.1	2944	1396	?	S	06:26	0:00	/usr/sbin/apache
www-data	6070	0.0	1.1	2944	1396	?	S	06:26	0:00	/usr/sbin/apache
www-data	6071	0.0	1.1	2944	1396	?	S	06:26	0:00	/usr/sbin/apache
www-data	6072	0.0	1.1	2944	1396	?	S	06:26	0:00	/usr/sbin/apache
www-data	6073	0.0	1.1	2944	1396	?	S	06:26	0:00	/usr/sbin/apache
root	7009	0.0	1.2	5704	1632	?	S	08:46	0:00	/usr/sbin/sshd
joao	7011	0.0	1.3	5812	1740	?	S	08:46	0:01	/usr/sbin/sshd
joao	7012	0.0	1.1	2620	1492	pts/0	S	08:46	0:00	-sh
root	7591	0.0	1.2	5704	1632	?	S	10:14	0:00	/usr/sbin/sshd
joao	7596	0.0	1.3	5812	1740	?	S	10:15	0:00	/usr/sbin/sshd
joao	7597	0.0	1.0	2468	1340	pts/1	S	10:15	0:00	-sh
joao	7660	0.0	0.8	2016	1028	pts/0	T	10:25	0:00	top
root	7992	0.0	1.0	2472	1336	pts/1	S	11:14	0:00	bash
root	8012	0.0	1.1	3288	1408	pts/1	R	11:19	0:00	ps aux

Opções:

- a : mostra os processos criados por todos os usuários do sistema
- x : mostra processos que não são controlados por terminal
- u : mostra o nome de usuário que iniciou o processo e hora em que o processo foi iniciado.

O terminal onde você inicia uma tarefa é chamado de terminal que controla a tarefa.

Outra opção é utilizar **--forest** que mostra a hierarquia de processos:

Listagem 6.27: Controlando Processos

```
$ ps x --forest
  PID TTY      STAT   TIME COMMAND
 8100 ?        S      0:00 /usr/sbin/sshd
 8101 pts/0    S      0:00  \_ -sh
 8168 pts/0    R      0:00      \_ ps x --forest
```

Cada processo no Linux tem uma prioridade, esta prioridade determina a velocidade relativa que o processo irá rodar em seu sistema. Você pode mudar a prioridade de um processo com o comando **nice**. Quanto menor o seu valor, maior a prioridade do processo (varia de -20 a 19). Os processos recém-criados herdam do pai o valor do **nice**.

Crie o programa abaixo:

Listagem 6.28: Controlando Processos

```
$ vi teste.c
// digite o código abaixo
#include <stdio.h>

main()
{
    while (1)
    {

    }
}
```

Este código cria um programa com um loop infinito. Agora compile o mesmo.

Listagem 6.29: Controlando Processos

```
$ gcc -c teste.c
$ gcc -o teste teste.o
// pronto agora temos o arquivo executável teste
// agora copie o executável teste para teste1
$ cp teste teste1
```

Humm... precisamos colocar os programas rodando em background. Como se faz isto?

Listagem 6.30: Controlando Processos

```
$ ./teste &
$ ./teste1 &
// vamos listar os processos
$ ps
  PID TTY          TIME CMD
 8224 pts/0    00:00:00 sh
 8241 pts/0    00:04:14 teste
 8250 pts/0    00:02:38 teste1
 8301 pts/0    00:00:00 ps
$ ps -l
  F S   UID   PID  PPID   C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
 000 S 1000  8224  8223  0 75    0 -    617 wait4  pts/0    00:00:00 sh
 000 R 1000  8241  8224  59 77    0 -    308 -      pts/0    00:04:49 teste
 000 R 1000  8250  8224  49 80    0 -    308 -      pts/0    00:03:13 teste1
 000 R 1000  8305  8224  0 76    0 -    822 -      pts/0    00:00:00 ps
```

O campo PRI indica a prioridade do processo. O processo teste tem a prioridade 77 e o processo teste1 tem a prioridade 80. Quem vai ficar mais tempo na CPU? Logicamente o processo com maior prioridade, mas os que tem a maior prioridade tem o menor número. Confuso? Vamos usar o top para verificar isto:

Listagem 6.31: Controlando Processos

```
11:56:23 up 21:09,  1 user,  load average: 1.99, 1.76, 0.99
39 processes: 35 sleeping, 4 running, 0 zombie, 0 stopped
CPU states: 100.0% user, 0.0% system, 0.0% nice, 0.0% idle
Mem: 126820K total, 115752K used, 11068K free, 12228K buffers
Swap: 248968K total, 6128K used, 242840K free, 45564K cached

  PID USER      PRI  NI   SIZE   RSS SHARE STAT %CPU %MEM      TIME COMMAND
 8241 joao      20   0   244   244    204 R     50.8  0.1   6:35 teste
 8250 joao      14   0   244   244    204 R     49.2  0.1   4:59 teste1
     1 root      8   0   484   444    424 S     0.0   0.3   0:00 init
```

```

2 root      9  0    0    0    0 SW    0.0  0.0  0:00 keventd
3 root     19 19    0    0    0 SWN   0.0  0.0  0:00 ksoftirqd_CPU0
4 root      9  0    0    0    0 SW    0.0  0.0  0:06 kswapd
5 root      9  0    0    0    0 SW    0.0  0.0  0:00 bdflush
6 root      9  0    0    0    0 SW    0.0  0.0  0:01 kupdated
133 root     9  0  628  576  496 S    0.0  0.4  0:00 dhclient-2.2.x
208 root     9  0  600  596  492 S    0.0  0.4  0:03 syslogd
211 root     9  0 1052 1044  412 S    0.0  0.8  0:01 klogd
243 root     9  0 1244  960  804 S    0.0  0.7  0:11 nmbd
245 root     9  0 1100  788  636 S    0.0  0.6  0:00 smbd
342 root     9  0 1120 1052  944 S    0.0  0.8  0:00 sshd
352 root     9  0  872  832  696 S    0.0  0.6  0:00 xinetd
355 nobody    8  0 1148 1028  888 S    0.0  0.8  0:00 proftpd
358 daemon    9  0  580  556  504 S    0.0  0.4  0:00 atd

```

Como podemos ver o processo teste está consumindo a maior parte da CPU. Vamos matar os dois processos para entender mais sobre prioridade.

Listagem 6.32: Controlando Processos

```

$ ps
  PID TTY          TIME CMD
8224 pts/0        00:00:00 sh
8241 pts/0        00:07:54 teste
8250 pts/0        00:06:18 testel
8330 pts/0        00:00:00 ps
$ kill -9 8241
$ kill -9 8250
[1]- Morto                  ./teste
[2]+ Morto                  ./testel
$ ps
  PID TTY          TIME CMD
8224 pts/0        00:00:00 sh
8332 pts/0        00:00:00 ps

```

O comando **nice** permite que ao inicializar um programa conseguimos determinar sua prioridade de escalonamento. Conseguimos determinar as prioridades de escalonamento do **nice** de -19 a -1.

Listagem 6.33: Controlando Processos

```

$ nice -19 ./teste
$ nice -10 ./teste
$ top
12:12:41 up 21:25,  1 user,  load average: 1.13, 0.57, 0.71
39 processes: 35 sleeping, 4 running, 0 zombie, 0 stopped
CPU states:  0.0% user,  0.0% system, 100.0% nice,  0.0% idle
Mem: 126820K total, 115820K used, 11000K free, 12228K buffers
Swap: 248968K total, 6128K used, 242840K free, 45584K cached

  PID USER      PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM      TIME COMMAND
8451 joao      20  10 244  244  204 R N  75.0  0.1  0:32 testel
8450 joao      20  19 244  244  204 R N  25.0  0.1  0:16 teste
  1 root      8  0  484  444  424 S    0.0  0.3  0:00 init
  2 root      9  0    0    0    0 SW    0.0  0.0  0:00 keventd
  3 root     19 19    0    0    0 SWN   0.0  0.0  0:00 ksoftirqd_CPU0
  4 root      9  0    0    0    0 SW    0.0  0.0  0:06 kswapd
  5 root      9  0    0    0    0 SW    0.0  0.0  0:00 bdflush
  6 root      9  0    0    0    0 SW    0.0  0.0  0:01 kupdated
133 root     9  0  628  576  496 S    0.0  0.4  0:00 dhclient-2.2.x
208 root     9  0  600  596  492 S    0.0  0.4  0:03 syslogd
211 root     9  0 1052 1044  412 S    0.0  0.8  0:01 klogd
243 root     9  0 1244  960  804 S    0.0  0.7  0:11 nmbd

```

```

245 root      9  0  1100  788   636 S    0.0  0.6  0:00 smbd
342 root      9  0  1120 1052   944 S    0.0  0.8  0:00 sshd
352 root      9  0   872  832   696 S    0.0  0.6  0:00 xinetd
355 nobody    8  0  1148 1028   888 S    0.0  0.8  0:00 proftpd
358 daemon    9  0   580  556   504 S    0.0  0.4  0:00 atd

```

Como podemos observar na coluna NI o programa teste1 tem agora maior prioridade de escalonamento do que o programa teste. Outra forma de verificar é o %CPU. Quem está consumindo mais.

Vamos supor que agora desejamos mudar a prioridade de escalonamento do teste. O comando **nice** somente a prioridade quando o comando é iniciado. Assim vamos utilizar o **renice**. A faixa do **renice** varia de 0 a +20.

Listagem 6.34: Controlando Processos

```

// primeiro precisamos do PID do programa teste
$ ps
  PID TTY          TIME CMD
 8224 pts/0    00:00:00 sh
 8450 pts/0    00:01:28 teste
 8451 pts/0    00:04:09 testel
 8471 pts/0    00:00:00 ps
// hummm é o número 8450
// vamos mudar para 9 a prioridade
$ renice 9 8450
// se der algum erro de permissão mude para o usuário root
# renice 9 8450
8450: old priority 19, new priority 9

$ top

12:20:39 up 21:33,  1 user,  load average: 1.99, 1.70, 1.21
44 processes: 40 sleeping, 4 running, 0 zombie, 0 stopped
CPU states:  0.2% user,  0.0% system, 99.8% nice,  0.0% idle
Mem: 126820K total, 116968K used,   9852K free, 122228K buffers
Swap: 248968K total,   6128K used, 242840K free, 45600K cached
  PID USER      PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM      TIME COMMAND
 8450 joao     20   9  244   244   204 R N  49.9  0.1  2:30 teste
 8451 joao     20  10  244   244   204 R N  49.7  0.1  6:14 testel
 8223 joao      9   0 1768  1740  1552 R      0.1  1.3  0:00 sshd
    1 root      8   0   484   444   424 S      0.0  0.3  0:00 init
    2 root      9   0     0     0     0 SW      0.0  0.0  0:00 keventd
    3 root     19  19     0     0     0 SWN     0.0  0.0  0:00 ksoftirqd_CPU0
    4 root      9   0     0     0     0 SW      0.0  0.0  0:06 kswapd
    5 root      9   0     0     0     0 SW      0.0  0.0  0:00 bdflush
    6 root      9   0     0     0     0 SW      0.0  0.0  0:01 kupdated
133 root      9   0   628   576   496 S      0.0  0.4  0:00 dhclient-2.2.x
208 root      9   0   600   596   492 S      0.0  0.4  0:03 syslogd
211 root      9   0  1052  1044   412 S      0.0  0.8  0:01 klogd
243 root      9   0  1244   960   804 S      0.0  0.7  0:11 nmbd
245 root      9   0  1100   788   636 S      0.0  0.6  0:00 smbd
342 root      9   0  1120  1052   944 S      0.0  0.8  0:00 sshd
352 root      9   0   872   832   696 S      0.0  0.6  0:00 xinetd
355 nobody    8   0  1148  1028   888 S      0.0  0.8  0:00 proftpd

```

Após algum tempo o programa teste está consumindo a maior parte da CPU pois alteramos a prioridade de escalonamento. Agora mate todos estes processos antes que eles acabem com os recursos da sua máquina.

Capítulo 7 | Permissão e Propriedade

7.1 Permissão e Propriedade

O Linux herdou o conceito de permissão e propriedade de arquivos utilizando no UNIX. Os sistemas UNIX supõem que a máquina pode ser compartilhada por diferentes usuários, por este motivo é necessário atribuir posse e permissões diferentes para cada usuário do sistema. Cada arquivo possui então um dono e permissões diferenciadas para quem é o dono e quem não é. Para verificar estas características em um arquivo, você pode usar o comando `ls` conforme o exemplo na Listagem 7.1.

Listagem 7.1: Listando o Proprietário e as Permissões dos Arquivos

```
$ ls -l  
-rw-rw-r-- 1 bob users 375600 Dez 5 14:30 myfile
```

O arquivo que foi listado no exemplo na Listagem 7.1 pertence ao usuário *bob*. Este, por sua vez, está no grupo de usuários chamado *users*. As permissões do arquivo estão listadas no código `-rw-rw-r-`. O primeiro traço à esquerda significa que este é um arquivo normal contendo qualquer tipo de dados. Um diretório teria um `d` ao invés do traço `-`. Os próximos 9 caracteres são as permissões do arquivo. Os 3 primeiros dizem quais são as permissões do usuário dono do arquivo. Os próximos 3 dizem quais são as permissões que os usuários do grupo possuem sobre o arquivo. Por fim, os 3 últimos caracteres dizem quais são as permissões para qualquer outra pessoa. Cada grupo de 3 caracteres versa sobre as permissões de leitura (r, read), escrita (w, write) e execução (x, execute) do arquivo, nesta ordem. O arquivo ilustrado na Listagem 7.1 possui permissão de leitura e escrita para o usuário *bob* e os membros do grupo *users*, os demais usuários possuem permissão apenas de leitura. Cada uma das permissões é binária (tem permissão ou não tem), desta forma cada conjunto de permissão expresso pelos 3 caracteres pode ser representado por um número binário com 3 bits. Desta forma, temos o seguinte:

<code>---</code>	$(000)_2 = 0$	todas permissões negadas
<code>--x</code>	$(001)_2 = 1$	permissão apenas de execução
<code>-w-</code>	$(010)_2 = 2$	permissão apenas de escrita
<code>-wx</code>	$(011)_2 = 3$	permissão para escrita e execução
<code>r--</code>	$(100)_2 = 4$	permissão para leitura
<code>r-x</code>	$(101)_2 = 5$	permissão para leitura e execução
<code>rw-</code>	$(110)_2 = 6$	permissão para leitura e escrita
<code>rwx</code>	$(111)_2 = 7$	permissão para leitura, escrita e execução

Podemos trocar o dono de um arquivo, assim como as permissões. Para tanto utilizaremos os comandos `chown` e `chmod`, respectivamente.

Listagem 7.2: Trocando o Dono de um Arquivo

```
$ chown john:users2 myfile  
$ ls -l myfile  
-rw-rw-r-- 1 john users2 375600 Dez 5 14:30 myfile
```

Listagem 7.3: Trocando as Permissões de um Arquivo

```
$ chmod 777 myfile
$ ls -l myfile
-rwxrwxrwx 1 john users2 375600 Dez  5 14:30 myfile
$ chmod 600 myfile
$ ls -l myfile
-rw----- 1 john users2 375600 Dez  5 14:30 myfile
```

Podemos também mudar as permissões utilizando as representações simbólicas:

u para o usuário

g para o grupo

o para outros

a para todos

r permissão de leitura

w permissão de escrita

x permissão de execução

Os seguintes exemplos na Listagem 7.4 ilustram algumas possíveis utilizações destas representações para alterar as permissões de um arquivo ou diretório.

Listagem 7.4: Exemplos de Utilização do chmod

```
# adicionar a permissao de execucao apenas ao usuario
$ chmod u+x arquivo

# adicionar multiplas permissoes (por exemplo, leitura e execucao)
$ chmod u+rwx arquivo

# adicionar permissoes diferentes a usuario (permissao de leitura) e (permissao de grupo)
      grupo
$ chmod u+r,g+x arquivo

# remover permissoes (leitura e execucao)
$ chmod u-rx

# adicionar a permissao de execucao a todos os usuarios
$ chmod a+x
```

Capítulo 8 | Gerenciando Usuários

Este Capítulo tem como objetivo apresentar todos os comandos para o gerenciamento de usuários no Sistema Operacional Linux.

8.1 Listando Todos os Usuários do Sistema

Para listar todos os usuários do sistema Linux basta verificar quais estão listados no arquivo */etc/passwd* utilizando, para tanto, o comando **cat** como na Listagem 8.1, na qual utilizando o **head** para limitar aos 10 primeiros.

Listagem 8.1: Listando Todos os Usuários do Linux

```
$ cat /etc/passwd | head -10
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
```

Ou ainda, podemos utilizar o **cut**, como na Listagem 8.2, aonde utilizamos novamente o **head** para limitar aos 10 primeiros.

Listagem 8.2: Listando Todos os Usuários do Linux

```
$ cut -d: -f1 /etc/passwd | head -10
root
daemon
bin
sys
sync
games
man
lp
mail
news
```

8.2 Listando Grupos

Para listar os grupos disponíveis no Linux verificar o arquivo */etc/group*, por exemplo, utilizando o comando **cat** como na Listagem 8.3.

Listagem 8.3: Listando Todos os Usuários do Linux

```
$ cat /etc/group | head -10
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,leoca
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
```

8.3 Adicionando Usuários

Para adicionar um usuário basta utilizar o comando **useradd**. A sintaxe para adicionar usuário está apresentada na Listagem 8.4. Esta operação é privilegiada e só deve ser utilizada como **root**.

Listagem 8.4: Adicionando Usuários

```
useradd aluno
```

8.4 Definindo Senha para Novos Usuários

Para adicionar ou mudar a senha de um usuário basta utilizar o comando **passwd**. A sintaxe para adicionar usuário está apresentada na Listagem 8.5.

Listagem 8.5: Definindo Senha de Usuário

```
passwd aluno
Changing password for user aluno.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

8.5 Apagando uma Conta de Usuário

Para apagar uma conta de usuário basta utilizar o comando **userdel**. A sintaxe para adicionar usuário está apresentada na Listagem 8.6.

Listagem 8.6: Apagando Usuários

```
userdel aluno
```

Se quiser apagar também o seu diretório utilize o comando na Listagem 8.7.

Listagem 8.7: Apagando Usuários e Arquivos do Usuário

```
userdel -r aluno
```

8.6 Modificando Conta de Usuário

Para modificar uma conta de usuário basta utilizar o comando **usermod**. A sintaxe para inserir uma data para a conta do usuário expirar está apresentada na Listagem 8.8.

Listagem 8.8: Modificando Conta do Usuário

```
usermod -e 2015-11-25 aluno
```

8.7 Adicionando um Novo Grupo

Para adicionar um grupo basta utilizar o comando **groupadd**. A sintaxe para inserir um novo grupo está listado na Listagem 8.9.

Listagem 8.9: Adicionando um Novo Grupo

```
groupadd professores
```

8.8 Deletando um Grupo

Para apagar um grupo basta utilizar o comando **groupdel**. A sintaxe para apagar um grupo está listado na Listagem 8.10.

Listagem 8.10: Apagando um Grupo

```
groupdel professores
```

8.9 Modificando um Grupo

Para modificar um grupo basta utilizar o comando **groupmod**. A sintaxe para modificar um grupo está listado na Listagem 8.11. Com este comando renomeamos o grupo professores para funcionários.

Listagem 8.11: Modificando um Grupo

```
groupmod -n professores funcionarios
```

Capítulo 9 | Comandos para Redes de Computadores

9.1 Comando hostname

Quer descobrir o nome de sua máquina? Uma maneira simples é entrar no terminal e você conseguirá visualizar. Como exemplo, meu login recebe **vivaszafu:\$**, neste caso estou logado com o usuário **vivas** na máquina **zafu**.

Você pode também utilizar o comando **hostname** apresentado na Listagem 9.1

Listagem 9.1: Comando hostname

```
$ hostname  
zafu
```

O nome da máquina é armazenado no arquivo **hosts** que fica no diretório **etc**. Assim, podemos obter o nome da máquina utilizando o comando como na Listagem 9.2.

Listagem 9.2: Nome da Máquina com cat

```
$ cat /etc/hosts  
vivas@zafu:~$ cat /etc/hosts  
127.0.0.1 localhost  
127.0.1.1 zafu  
  
# The following lines are desirable for IPv6 capable hosts  
::1      ip6-localhost ip6-loopback  
fe00::0  ip6-localnet  
ff00::0  ip6-mcastprefix  
ff02::1  ip6-allnodes  
ff02::2  ip6-allrouters
```

9.2 Comando e Tabela ARP

Em uma rede local precisamos do endereço MAC antes de qualquer comunicação e geralmente temos o endereço IP. Assim, cada computador possui um servidor ARP (Address Resolution Protocol) que fornece o endereço MAC de nossa máquina quando solicitado.

O protocolo ARP é responsável por receber um pacote com o endereço IP e enviar para o destinatário o endereço MAC. Quer saber a tabela ARP do seu computador? Isto é, os computadores que de alguma forma você entrou em contato? Este comando funciona no MAC, no Linux e até no Windows, Listagem 9.3. Através do comando **arp** é possível visualizar a tabela ARP.

Listagem 9.3: Tabela ARP

```
$ arp -a
? (192.168.0.1) at 1c:7e:e5:46:92:e7 on en1 ifscope [ethernet]
? (192.168.0.100) at 4c:e6:76:be:ee:a9 on en1 ifscope [ethernet]
```

9.3 Verificando o Endereço IP de sua Máquina

Para verificar seu endereço IP basta utilizar o comando **ifconfig**. Como pode ser vista aparecem duas interfaces neste comando

- eth0: é a interface de rede padrão (rede cabeadas)
- lo: loopback interface utilizada para realização de testes. Ao enviar um pacote para esta interface o pacote não vai para rede externa.

A Listagem 9.4 ilustra o resultado do comando **ifconfig**.

9.3.1 Verificando Endereço IP

Listagem 9.4: Verificando o Endereço IP

```
$ ifconfigvivas@zafu:~$ ifconfig
eth0      Link encap:Ethernet  Endereco de HW 08:00:27:3e:1f:1c
          inet end.: 192.168.0.107  Bcast:192.168.0.255  Masc:255.255.255.0
          endereço inet6: fe80::a00:27ff:fe3e:1f1c/64 Escopo:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metrica:1
          pacotes RX:59858 erros:0 descartados:1 excesso:0 quadro:0
          Pacotes TX:4055 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:1000
          RX bytes:5992957 (5.9 MB)  TX bytes:562733 (562.7 KB)

lo        Link encap:Loopback Local
          inet end.: 127.0.0.1  Masc:255.0.0.0
          endereço inet6: ::1/128 Escopo:Máquina
          UP LOOPBACK RUNNING  MTU:65536  Metrica:1
          pacotes RX:1064 erros:0 descartados:0 excesso:0 quadro:0
          Pacotes TX:1064 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:0
          RX bytes:83202 (83.2 KB)  TX bytes:83202 (83.2 KB)
```

9.4 Habilitando e Desabilitando a Interface de Rede

Para habilitar ou desabilitar uma interface de rede utilizamos o comando **ifconfig**. Primeiro vamos verificar o status da interface de rede Ethernet, Listagem 9.5.

Listagem 9.5: Verificando Status da Rede Ethernet

```
vivas@zafu:~$ ifconfig eth0
eth0      Link encap:Ethernet  Endereco de HW 08:00:27:3e:1f:1c
          inet end.: 192.168.0.107  Bcast:192.168.0.255  Masc:255.255.255.0
          endereço inet6: fe80::a00:27ff:fe3e:1f1c/64 Escopo:Link
```

```
UP BROADCAST RUNNING MULTICAST MTU:1500 Metrica:1
pacotes RX:58889 erros:0 descartados:0 excesso:0 quadro:0
Pacotes TX:3786 erros:0 descartados:0 excesso:0 portadora:0
colisoes:0 txqueuelen:1000
RX bytes:5886444 (5.8 MB) TX bytes:527634 (527.6 KB)
```

Para desabilitar utilizamos o comando da Listagem 9.5. Para verificar vamos utilizar o comando **ifconfig** na Listagem 9.6.

Listagem 9.6: Desabilitando a Interface de Rede Ethernet

```
vivas@zafu:~$ sudo ifconfig eth0 down
Password:
```

Para verificar se a interface está desabilitada, proceda com o comando **ifconfig** como na Listagem 9.7.

Listagem 9.7: Verificando a Ação Realizada na Listagem 9.6

```
vivas@zafu:~$ ifconfig
lo      Link encap:Loopback Local
        inet end.: 127.0.0.1 Masc:255.0.0.0
        Endereco inet6: ::1/128 Escopo:Maquina
        UP LOOPBACK RUNNING MTU:65536 Metrica:1
        pacotes RX:1046 erros:0 descartados:0 excesso:0 quadro:0
        Pacotes TX:1046 erros:0 descartados:0 excesso:0 portadora:0
        colisoes:0 txqueuelen:0
        RX bytes:81786 (81.7 KB) TX bytes:81786 (81.7 KB)
```

Para habilitar, utilizamos o comando da Listagem 9.8 e, para verificar, vamos utilizar o comando **ifconfig** na Listagem 9.9.

Listagem 9.8: Habilitando a Interface de Rede Ethernet

```
vivas@zafu:~$ sudo ifconfig eth0 up
Password:
```

Listagem 9.9: Verificando a Ação Realizada na Listagem 9.8

```
vivas@zafu:~$ ifconfig
eth0      Link encap:Ethernet Endereco de HW 08:00:27:3e:1f:1c
        inet end.: 192.168.0.107 Bcast:192.168.0.255 Masc:255.255.255.0
        Endereco inet6: fe80::a00:27ff:fe3e:1f1c/64 Escopo:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metrica:1
        pacotes RX:59662 erros:0 descartados:1 excesso:0 quadro:0
        Pacotes TX:4009 erros:0 descartados:0 excesso:0 portadora:0
        colisoes:0 txqueuelen:1000
        RX bytes:5955555 (5.9 MB) TX bytes:556911 (556.9 KB)

lo      Link encap:Loopback Local
        inet end.: 127.0.0.1 Masc:255.0.0.0
        Endereco inet6: ::1/128 Escopo:Máquina
        UP LOOPBACK RUNNING MTU:65536 Metrica:1
        pacotes RX:1056 erros:0 descartados:0 excesso:0 quadro:0
        Pacotes TX:1056 erros:0 descartados:0 excesso:0 portadora:0
        colisoes:0 txqueuelen:0
        RX bytes:82566 (82.5 KB) TX bytes:82566 (82.5 KB)
```

9.5 Alterando a MTU de uma Interface

MTU (*Maximum Transmission Unit*) é o tamanho do maior datagrama que pode ser transmitido em uma determinada rede. As redes Ethernet modernas utilizam o tamanho de 1500 bytes. Para listar a MTU utilizada usamos o comando **ifconfig** da Listagem 9.10. Para verificar isto utilizamos o comando **ifconfig**.

Listagem 9.10: Verificando MTU da Rede Ethernet

```
vivas@zafu:~$ ifconfig eth0
eth0      Link encap:Ethernet  Endereco de HW 08:00:27:3e:1f:1c
          inet end.: 192.168.0.107  Bcast:192.168.0.255  Masc:255.255.255.0
          endereço inet6: fe80::a00:27ff:fe3e:1f1c/64 Escopo:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metrica:1
          pacotes RX:58889 erros:0 descartados:0 excesso:0 quadro:0
          Pacotes TX:3786 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:1000
          RX bytes:5886444 (5.8 MB)  TX bytes:527634 (527.6 KB)
```

É possível alterar a MTU utilizando o comando **ifconfig** da Listagem 9.11.

Listagem 9.11: Alterando o MTU da Placa de Rede

```
vivas@zafu:~$ sudo ifconfig eth0 mtu 100
vivas@zafu:~$ ifconfig
eth0      Link encap:Ethernet  Endereco de HW 08:00:27:3e:1f:1c
          inet end.: 192.168.0.107  Bcast:192.168.0.255  Masc:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:100  Metrica:1
          pacotes RX:60974 erros:0 descartados:1 excesso:0 quadro:0
          Pacotes TX:4240 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:1000
          RX bytes:6087589 (6.0 MB)  TX bytes:585971 (585.9 KB)
```

9.6 Alterando Endereço IP

Se quiser alterar seu endereço IP basta utilizar o comando **ifconfig**. O comando recebe como parâmetros os parâmetros: interface, endereço IP e máscara. A Listagem 9.12 apresenta o endereço IP atual da interface eth0.

Listagem 9.12: Verificando o Endereço IP

```
vivas@zafu:~$ ifconfig eth0
eth0      Link encap:Ethernet  Endereço de HW 08:00:27:3e:1f:1c
          inet end.: 192.168.0.108  Bcast:192.168.0.255  Masc:255.255.255.0
          endereço inet6: fe80::a00:27ff:fe3e:1f1c/64 Escopo:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metrica:1
          pacotes RX:76743 erros:0 descartados:1 excesso:0 quadro:0
          Pacotes TX:4965 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:1000
          RX bytes:7496839 (7.4 MB)  TX bytes:672140 (672.1 KB)
```

A Listagem 9.13 apresenta o comando de alteração do endereço IP.

Listagem 9.13: Alterando Endereço IP

```
vivas@zafu:~$ sudo ifconfig eth0 192.168.0.107 netmask 255.255.255.0
Password:
```

```
vivas@zafu:~$ ifconfig eth0
eth0      Link encap:Ethernet  Endereço de HW 08:00:27:3e:1f:1c
          inet end.: 192.168.0.107  Bcast:192.168.0.255  Masc:255.255.255.0
          endereço inet6: fe80::a00:27ff:fe3e:1f1c/64 Escopo:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metrica:1
          pacotes RX:76743 erros:0 descartados:1 excesso:0 quadro:0
          Pacotes TX:4965 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:1000
          RX bytes:7496839 (7.4 MB) TX bytes:672140 (672.1 KB)
```

9.7 Comando ping

O comando **ping** serve para fazer verificações sobre o status de funcionamento de computadores em uma rede. Com ele podemos medir o tempo de ida e volta (*round time trip*) que um pacote demora para ir do seu host para outro. Você pode usar tanto o endereço IP do host ou o endereço Web. A Listagem 9.14 mostra o funcionamento do comando **ping**. Podemos passar um endereço IP, como ilustrado na Listagem 9.14, ou utilizar como argumento um endereço Web, como ilustrado na Listagem 9.15. Para interromper o comando basta digitar **<Ctrl+C>**. Quando interromper o comando serão mostradas as estatísticas dos testes realizados.

Listagem 9.14: Comando ping

```
vivas@zafu:~$ ping 192.168.0.107
PING 192.168.0.107 (192.168.0.107): 56 data bytes
64 bytes from 192.168.0.107: icmp_seq=0 ttl=64 time=0.476 ms
64 bytes from 192.168.0.107: icmp_seq=1 ttl=64 time=0.406 ms
64 bytes from 192.168.0.107: icmp_seq=2 ttl=64 time=0.400 ms
64 bytes from 192.168.0.107: icmp_seq=3 ttl=64 time=0.372 ms
64 bytes from 192.168.0.107: icmp_seq=4 ttl=64 time=0.348 ms
64 bytes from 192.168.0.107: icmp_seq=5 ttl=64 time=0.199 ms
64 bytes from 192.168.0.107: icmp_seq=6 ttl=64 time=0.722 ms
64 bytes from 192.168.0.107: icmp_seq=7 ttl=64 time=0.801 ms
64 bytes from 192.168.0.107: icmp_seq=8 ttl=64 time=0.732 ms
64 bytes from 192.168.0.107: icmp_seq=9 ttl=64 time=0.663 ms
64 bytes from 192.168.0.107: icmp_seq=10 ttl=64 time=0.709 ms
^C
--- 192.168.0.107 ping statistics ---
11 packets transmitted, 11 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.199/0.530/0.801/0.192 ms
```

Listagem 9.15: Exemplo do Comando ping

```
$ ping www.ufsj.edu.br
```

Você pode utilizar a opção **-c** com o **ping** onde especifica o número de pacotes enviados pelo ping. A Listagem 9.16 apresenta o comando para 10 tentativas.

Listagem 9.16: Comando ping

```
$ ping -c 10 www.ufsj.edu.br
PING www.ufsj.edu.br (200.17.67.239): 56 data bytes
Request timeout for icmp_seq 0
64 bytes from 200.17.67.239: icmp_seq=1 ttl=56 time=53.561 ms
64 bytes from 200.17.67.239: icmp_seq=2 ttl=56 time=50.592 ms
64 bytes from 200.17.67.239: icmp_seq=3 ttl=56 time=51.065 ms
64 bytes from 200.17.67.239: icmp_seq=4 ttl=56 time=50.581 ms
```

```
Request timeout for icmp_seq 5
64 bytes from 200.17.67.239: icmp_seq=6 ttl=56 time=53.212 ms
64 bytes from 200.17.67.239: icmp_seq=7 ttl=56 time=50.612 ms
64 bytes from 200.17.67.239: icmp_seq=8 ttl=56 time=54.141 ms
64 bytes from 200.17.67.239: icmp_seq=9 ttl=56 time=54.225 ms

--- www.ufsj.edu.br ping statistics ---
10 packets transmitted, 8 packets received, 20.0% packet loss
round-trip min/avg/max/stddev = 50.581/52.249/54.225/1.571 ms
```

Quer controlar o intervalo de tempo entre os pacotes enviados? Utilize a opção **-i** como mostrado na Listagem 9.17, vamos aproveitar e utilizar a opção de enviar 3 pacotes.

Listagem 9.17: Comando ping com Opção de Tempo

```
$ ping -i 5 -c 3 192.168.0.1
PING 192.168.0.1 (192.168.0.1): 56 data bytes
64 bytes from 192.168.0.1: icmp_seq=0 ttl=64 time=3.772 ms
64 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time=0.926 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=64 time=4.215 ms

--- 192.168.0.1 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.926/2.971/4.215/1.457 ms
```

9.8 Descobrir endereço IP de um Determinado Host

Muitas vezes precisamos descobrir o endereço IP de um determinado host. Para realizar esta tarefa utilizamos o comando **host**. A Listagem 9.18 mostra como descobrir o endereço IP do Google.

Listagem 9.18: Descobrindo o Endereço Ip de um Host

```
$ host www.google.com
www.google.com has address 173.194.118.48
www.google.com has address 173.194.118.49
www.google.com has address 173.194.118.52
www.google.com has address 173.194.118.50
www.google.com has address 173.194.118.51
www.google.com has IPv6 address 2800:3f0:4001:811::1011
```

9.9 Informações sobre Domínios

9.9.1 Comando dig

Uma maneira de obter informações sobre domínios é utilizar o comando **dig**. A sintaxe é bem simples, pois basta usar como parâmetro o domínio desejado. A Listagem 9.19 ilustra a utilização do comando.

Listagem 9.19: Descobrir Informações sobre um Domínio

```
$ dig www.vivas.eng.br
; <>> DiG 9.8.1-P1 <>> www.vivas.eng.br
```

```

;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 48180
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.vivas.eng.br. IN A

;; ANSWER SECTION:
www.vivas.eng.br. 14400 IN CNAME vivas.eng.br.
vivas.eng.br. 14400 IN A 208.115.217.250

;; Query time: 437 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Dec 16 16:33:28 2013
;; MSG SIZE rcvd: 64

```

9.9.2 Comando nslookup

Outra maneira é utilizar o comando **nslookup**. A Listagem 9.20 ilustra a utilização do comando.

Listagem 9.20: Utilizando o comando nslookup

```

$ nslookup www.vivas.eng.br
Server: 127.0.0.1
Address: 127.0.0.1#53

Non-authoritative answer:
www.vivas.eng.br canonical name = vivas.eng.br.
Name: vivas.eng.br
Address: 208.115.217.250

```

9.10 Traçando caminhos de um host a outro

O comando **traceroute** é uma ferramenta para imprimir os caminhos de seu host até um destino. Ele mostra todos os roteadores que o pacote enviado passa e imprime informações sobre o tempo decorrido. A Listagem 9.21 mostra como instalar o comando.

Listagem 9.21: Instalação do Traceroute

```

$ sudo apt-get install traceroute
Password:

```

Para usar basta inserir o destino como na Listagem 9.22. Alguns roteadores são programados para não responder e por isto aparecem o símbolo *. Outra situação é ocorrência de perda de pacotes.

Listagem 9.22: Rotas

```

vivas@zafu:~$ traceroute www.ufvjm.edu.br
traceroute to www.ufvjm.edu.br (200.131.252.28), 30 hops max, 60 byte packets
 1 192.168.0.1 (192.168.0.1) 1.110 ms 1.462 ms 1.356 ms
 2 10.0.0.1 (10.0.0.1) 3.111 ms 3.750 ms 4.408 ms
 3 200-217-90-93.host.telemar.net.br (200.217.90.93) 27.038 ms 30.066 ms 34.359 ms
 4 xe-3-0-0-hga-mg-rotm-j01.telemar.net.br (200.164.13.101) 39.164 ms 42.413 ms
               48.474 ms

```

```
5 xe-14-0-1.0-bot-rj-rotn-j01.telemar.net.br (200.223.46.183) 57.143 ms so-0-1-0.0-bot
   -rj-rotn-j01.telemar.net.br (200.164.197.218) 60.249 ms 64.807 ms
6 pos9-0-arc-rj-rotb-03.telemar.net.br (200.223.254.117) 68.923 ms pos12-0-arc-rj-rotb
   -03.telemar.net.br (200.223.131.217) 36.913 ms gigabitEthernet8-0-0-arc-rj-rotb-03.
   telemar.net.br (200.223.45.78) 40.338 ms
7 as1916.rj.ptt.br (200.219.138.101) 42.654 ms 33.228 ms 36.886 ms
8 rj-df-10g-oi.bkb.rnp.br (200.143.252.78) 60.204 ms 65.422 ms 70.177 ms
9 df-mg-10g-oi.bkb.rnp.br (200.143.252.81) 65.159 ms 69.861 ms 72.669 ms
10 lanmg-mxmg-10g-int.bkb.rnp.br (200.143.255.174) 81.736 ms 82.942 ms 88.463 ms
11 ufvjm.pop-mg.rnp.br (200.131.2.90) 166.962 ms 168.942 ms 156.812 ms
12 * * *
13 * * *
```

9.10.1 Descobrindo o Endereço do seu Roteador sem Fio

Uma das maneiras é utilizar o comando traceroute. O primeiro salto sempre será o IP do seu roteador sem fio. Verifique a Listagem 9.23. Neste caso enviamos o comando para dois destinos diferentes e observamos que os dois sempre no primeiro salto passam pelo o IP 192.168.0.1 que é o do roteador sem fio.

Listagem 9.23: Endereço do Roteador sem Fio

```
vivas@zafu:~$ traceroute www.ufmg.br
traceroute to www.ufmg.br (150.164.250.1), 30 hops max, 60 byte packets
 1  192.168.0.1 (192.168.0.1)  1.030 ms  1.179 ms  2.036 ms
^C
vivas@zafu:~$ traceroute www.mit.edu
traceroute to www.mit.edu (23.65.134.151), 30 hops max, 60 byte packets
 1  192.168.0.1 (192.168.0.1)  1.106 ms  1.415 ms  1.372 ms
```

9.11 Comando tracepath

O comando **tracepath** é similar ao comando **traceroute**, mas possui opções menos complicadas. Para utilizar o comando siga a sintaxe da Listagem 9.24.

Listagem 9.24: Rotas com tracepath

```
vivas@zafu:~$ tracepath www.ufsj.edu.br
 1: zafu.local                               0.058ms pmtu 1500
 1: 192.168.0.1                             1.206ms
 1: 192.168.0.1                             1.619ms
 2: 10.0.0.1                                3.714ms
 3: 10.0.0.1                                2.366ms pmtu 1492
 3: 200-217-90-93.host.telemar.net.br        63.392ms
 4: xe-3-0-0-0-mpi-mg-rotn-j01.telemar.net.br 63.401ms asymm 5
 5: ge-11-0-0-0-arc-rj-rotn-j01.telemar.net.br 71.083ms asymm 6
 6: gigabitEthernet8-0-0-arc-rj-rotb-03.telemar.net.br 70.981ms
 7: as1916.rj.ptt.br                         72.993ms
 8: rj-df-10g-oi.bkb.rnp.br                  90.410ms
 9: df-mg-10g-oi.bkb.rnp.br                  86.591ms asymm 6
10: lanmg-mxmg-10g-int.bkb.rnp.br          88.762ms asymm 7
11: ufsj-n.pop-mg.rnp.br                   95.267ms asymm 9
12: no reply
13: no reply
14: no reply
```

9.12 Comando netstat

O comando **netstat** é uma ferramenta essencial para administradores de rede. Ele possibilita fazer rastreamento das portas que são utilizadas no seu computador. A Listagem 9.25 apresenta estatística dos protocolos.

Listagem 9.25: Estatísticas de Rede com netstat

```
vivas@zafu:~$ netstat -s
Ip:
    14476 total de pacotes recebidos
    0 encaminhado
    0 pacotes de entrada descartados
    14470 pacotes de entrada entregues
    10142 requisições enviadas
Icmp:
    318 mensagens ICMP recebidas
    0 mensagens ICMP de entrada com problemas.
    Histograma de entrada ICMP:
        destino inalcancavel: 96
        tempo expirou em transito: 222
    434 mensagens ICMP enviadas
    0 mensagens ICMP falharam
    Histograma de saída ICMP
        destino inalcancavel: 434
IcmpMsg:
    InType3: 96
    InType11: 222
    OutType3: 434
Tcp:
    34 conexões ativas abertas
    2 conexões passivas abertas
    0 tentativas de conexão que falharam
    10 reinícios de conexões recebidos
    3 conexões estabelecidas
    9272 segmentos recebidos
    7445 segmentos enviados
    11 segmentos retransmitidos
    0 segmentos inválidos recebidos
    30 reinícios enviados
Udp:
    2266 pacotes recebidos
    354 pacotes recebidos para uma porta desconhecida
    0 erros na recepção de pacotes
    2269 pacotes enviados
UdpLite:
TcpExt:
    3 soquetes TCP concluíram o tempo de espera mais rápido que o normal
    95 acks retardados enviados
    1 confirmações adiadas foram novamente adiadas devido a um soquete bloqueado
    6 pacotes diretamente enfileirados em recebmsg pre-fila.
    6748 bytes directly received in process context from prequeue
    3656 cabecalhos de pacotes previstos
    4 cabecalhos de pacote previstos e diretamente enfileirados ao usuário
    2156 acknowledgments not containing data payload received
    1341 reconhecimentos preditos
    2 outras expirações de tempo TCP
    6 conexões resetadas devido a dados não esperados
    10 conexões restauradas por cancelamento do usuário
    2 conexões abortadas por tempo expirado
    IPReversePathFilter: 5
```

```
TCPRcvCoalesce: 572
TCPOFOQueue: 603
IpExt:
  InNoRoutes: 1
  InMcastPkts: 754
  OutMcastPkts: 185
  InBcastPkts: 2341
  OutBcastPkts: 7
  InOctets: 6692920
  OutOctets: 1130339
  InMcastOctets: 187110
  OutMcastOctets: 19647
  InBcastOctets: 1011354
  OutBcastOctets: 328
```

A Listagem 9.26 apresenta a maneira de obter estatísticas das interfaces de rede.

Listagem 9.26: Comando netstat -i

```
vivas@zafu:~$ netstat -i
Tabela de Interfaces do Kernel
Iface MTU Met RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500 0      46078     0      0 0          6906     0      0      0 BMRU
lo        65536 0      4093     0      0 0          4093     0      0      0 LRU
```

9.12.1 Tabela de Roteamento

Para visualizar a tabela de roteamento de um host podemos utilizar também o comando **netstat**. A Listagem 9.27 apresenta o comando.

Listagem 9.27: Visualizando Tabela de Roteamento com netstat

```
vivas@zafu:~$ netstat -rn
Tabela de Roteamento IP do Kernel
Destino      Roteador      MascaraGen.      Opcoes      MSS Janela  irtt Iface
0.0.0.0      192.168.0.1  0.0.0.0      UG          0 0      0 eth0
169.254.0.0  0.0.0.0      255.255.0.0    U          0 0      0 eth0
192.168.0.0  0.0.0.0      255.255.255.0  U          0 0      0 eth0
```

9.13 Network Mapper

O comando **nmap** é uma ferramenta excelente para fazer varreduras em redes de computadores.

9.13.1 Instalação

O processo de instalação é bem simples e pode ser visualizado na Listagem 9.28.

Listagem 9.28: Instalação do nmap

```
$ sudo apt-get install nmap
Password:
```

9.13.2 Analisando portas abertas

O comando da Listagem 9.29 apresenta a versão básica do comando para listar as portas abertas de um determinado domínio.

Listagem 9.29: Verificando Portas Abertas

```
vivas@zafu:~$ nmap www.ufvjm.edu.br
Starting Nmap 5.21 ( http://nmap.org ) at 2013-12-17 15:24 BRST
Nmap scan report for www.ufvjm.edu.br (200.131.252.28)
Host is up (0.35s latency).
Not shown: 997 filtered ports
PORT      STATE    SERVICE
80/tcp    open     http
443/tcp   open     https
8080/tcp  closed   http-proxy

Nmap done: 1 IP address (1 host up) scanned in 44.78 seconds
```

9.13.3 Comando nmap com opção de mais informações

Se quiser mais informações sobre o procedimento utilize a opção **-v** como na Listagem 9.30.

Listagem 9.30: Comando nmap com opção -v

```
vivas@zafu:~$ nmap -v www.ufvjm.edu.br
Starting Nmap 5.21 ( http://nmap.org ) at 2013-12-17 15:34 BRST
Initiating Ping Scan at 15:34
Scanning www.ufvjm.edu.br (200.131.252.28) [2 ports]
Completed Ping Scan at 15:34, 0.40s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 15:34
Completed Parallel DNS resolution of 1 host. at 15:34, 0.00s elapsed
Initiating Connect Scan at 15:34
Scanning www.ufvjm.edu.br (200.131.252.28) [1000 ports]
Discovered open port 80/tcp on 200.131.252.28
Discovered open port 443/tcp on 200.131.252.28
Completed Connect Scan at 15:35, 23.50s elapsed (1000 total ports)
Nmap scan report for www.ufvjm.edu.br (200.131.252.28)
Host is up (0.26s latency).
Not shown: 997 filtered ports
PORT      STATE    SERVICE
80/tcp    open     http
443/tcp   open     https
8080/tcp  closed   http-proxy

Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 23.97 seconds
```

9.13.4 Rastreando Múltiplos Hosts

Para rastrear múltiplos hosts basta passar os endereços de IPs desejados. A Listagem 9.31 ilustra o procedimento para dois hosts

Listagem 9.31: Rastreando Múltiplos Hosts.numbers

```
vivas@zafu:~$ nmap 192.168.0.1 192.168.0.104
Starting Nmap 5.21 ( http://nmap.org ) at 2013-12-17 15:42 BRST
```

```
Nmap scan report for 192.168.0.1
Host is up (0.0074s latency).
6Not shown: 998 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
980/tcp   open  http

Nmap scan report for 192.168.0.104
12Host is up (0.00018s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
1522/tcp  open  ssh

Nmap done: 2 IP addresses (2 hosts up) scanned in 0.47 seconds
```

É possível rastrear múltiplos domínios como na Listagem 9.32

Listagem 9.32: Rastreando Múltiplos Domínios

```
vivas@zafu:~$ nmap www.google.com www.facebook.com

Starting Nmap 5.21 ( http://nmap.org ) at 2013-12-17 15:45 BRST
Nmap scan report for www.google.com (74.125.131.103)
Host is up (0.21s latency).

Hostname www.google.com resolves to 6 IPs. Only scanned 74.125.131.103
rDNS record for 74.125.131.103: vc-in-f103.1e100.net
Not shown: 998 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https

Nmap scan report for www.facebook.com (31.13.69.160)
Host is up (0.18s latency).
rDNS record for 31.13.69.160: edge-star-shv-12-iad1.facebook.com
Not shown: 997 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https
843/tcp   closed unknown

Nmap done: 2 IP addresses (2 hosts up) scanned in 31.96 seconds
```

Outra opção é rastrear uma rede completa como na Listagem 9.33.

Listagem 9.33: Rastreando uma Sub-rede

```
vivas@zafu:~$ nmap 192.168.0./*

Starting Nmap 5.21 ( http://nmap.org ) at 2013-12-17 15:49 BRST
Nmap scan report for 192.168.0.1
Host is up (0.048s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http

Nmap scan report for 192.168.0.101
Host is up (0.00034s latency).
All 1000 scanned ports on 192.168.0.101 are closed

Nmap scan report for 192.168.0.102
Host is up (0.0028s latency).
```

```
Not shown: 865 closed ports, 134 filtered ports
PORT      STATE SERVICE
62078/tcp open  iphone-sync

Nmap scan report for 192.168.0.104
Host is up (0.0082s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap done: 256 IP addresses (4 hosts up) scanned in 170.59 seconds
```

9.14 Comando route

O comando **route** possibilita a manipulação de rotas de roteamento. Se quiser verificar as rotas presentes em seu computador basta utilizar a Listagem 9.34

Listagem 9.34: Visualizando a Tabela de Roteamento

```
vivas@zafu:~$ route
Tabela de Roteamento IP do Kernel
Destino      Roteador      MascaraGen.      Opcoes Metrica Ref   Uso Iface
default      192.168.0.1    0.0.0.0          UG      0      0      0 eth0
link-local   *             255.255.0.0      U       1000   0      0 eth0
192.168.0.0  *             255.255.255.0    U       1      0      0 eth0
```

9.15 Comando telnet

O comando **telnet** foi muito utilizado como protocolo de acesso remoto. Aos poucos foi substituído pelo comando **ssh** devido a problemas de segurança.

9.15.1 Acessando Servidor Web via Telnet

Na Listagem 9.35 apresenta o comando para conectar ao servidor desejado.

Listagem 9.35: Uso do Telnet

```
$ telnet www.vivas.eng.br 80
```

Você irá receber a seguinte resposta do servidor como na Listagem 9.36.

Listagem 9.36: Resposta do Servidor

```
Trying 208.115.217.250...
Connected to vivas.eng.br.
Escape character is '^]'.
```

Digite os comandos seguintes da Listagem 9.37 e termine pressionando duas vezes enter.

Listagem 9.37: Acessando com Telnet o Servidor Web

```
GET / HTTP/1.1
Host: vivas.eng.br
```

Como resposta o servidor enviará informações do protocolo e enviará a página desejada como na Listagem 9.38. Retiramos a página do código para economia de espaço

Listagem 9.38: Resposta do Servidor

```
HTTP/1.1 200 OK
Server: nginx admin
Date: Tue, 17 Dec 2013 21:37:38 GMT
Content-Type: text/html
Content-Length: 7032
Connection: keep-alive
Vary: Accept-Encoding
Last-Modified: Mon, 16 Dec 2013 19:40:00 GMT
Accept-Ranges: bytes
X-Cache: HIT from Backend

Pagina

Connection closed by foreign host.
```

9.16 Acesso Remoto com ssh

O comando **ssh** permite o acesso remoto a um servidor. O primeiro passo é a instalação do pacote. A Listagem 9.39 apresenta o código para instalação do aplicativo.

Listagem 9.39: Instalando ssh

```
$ sudo apt-get install openssh-client
$ sudo apt-get install openssh-server
```

9.16.1 Acesso Remoto

Para acessar remotamente um servidor basta você fazer o procedimento da Listagem ???. Onde **vivas** é o usuário e o endereço IP do servidor é 192.168.0.1.

Listagem 9.40: Utilizando o ssh

```
$ ssh vivas@192.168.0.1
```

9.16.2 Rodando Aplicativos Gráficos Remotamente

Para rodar aplicativos gráficos remotamente via **ssh** você precisa alterar o arquivo de configuração do arquivo */etc/ssh/sshd_config*. Para isto abra o arquivo com modo privilegiado e mude a seguinte linha do arquivo: *ForwardX11 no* para *ForwardX11 yes*.

Depois reinicie o servidor **ssh** conforme Listagem 9.41.

Listagem 9.41: Rodando Aplicativos Gráficos Remotamente

```
sudo /etc/init.d/ssh restart
Rather than invoking init scripts through /etc/init.d, use the service(8)
utility, e.g. service ssh restart

Since the script you are attempting to invoke has been converted to an
Upstart job, you may also use the stop(8) and then start(8) utilities,
```

```
e.g. stop ssh ; start ssh. The restart(8) utility is also available.  
ssh stop/waiting  
ssh start/running, process 2532
```

Para logar exportando a parte gráfica utilizando a opção `-X` conforme Listagem 9.42.

Listagem 9.42: Logando com ssh -X

```
$ ssh -X vivas@192.168.0.104
```

Depois de logar digite o nome do aplicativo desejado seguido de `&` conforme Listagem 9.43.

Listagem 9.43: Abrindo Firefox Remotamente

```
$ firefox &
```

A Figura 9.1 apresenta o resultado do comando. Abrimos o aplicativo **firefox** remotamente em nosso terminal.



Figura 9.1: Firefox do computador Ubuntu rodando em um Mac

9.17 Copiando Arquivos com scp

O ssh permite também que você copie um arquivo de um computador remoto para outro computador remoto. Neste exemplo vou copiar o arquivo `teste.txt` (que está no diretório `/home/vivas`) que está no computador 192.168.0.104 para o meu computador para o diretório `/users/alessandrovivas`. Repare que você vai digitar a senha do seu computador remoto e não do computador que você está logado . Para fazer esta tarefa utilize o comando **scp** e o código está na Listagem 9.44.

Listagem 9.44: Copiando Arquivo em Servidor Remoto

```
$ scp vivas@192.168.0.104:/home/vivas/teste.txt /users/alessandrovivas
```

```
vivas@192.168.0.104's password:  
teste.txt 100% 438 0.4KB/s 00:00
```

9.18 Copiando um Diretório em um Servidor Remoto

Imagine que você criou um diretório em um servidor remoto. Vamos supor que o diretório tem o nome de `ubuntu` (`/home/vivas/ubuntu`). Para copiar o diretório inteiro, todos os arquivos, e criar a mesma estrutura no seu computador basta usar o comando `scp`. O código está apresentado na Listagem 9.45 e vamos utilizar o comando `scp`.

Listagem 9.45: Copiando um Diretório de um Servidor Remoto

```
$ scp -r vivas@192.168.0.104:/home/vivas/ubuntu /users/alessandrovivas/  
vivas@192.168.0.104's password:  
arquivo2      100%    0     0.0KB/s   00:00  
arquivo1      100%    0     0.0KB/s   00:00  
arquivo3      100%    0     0.0KB/s   00:00
```

9.19 Comando tcpdump

O comando **tcpdump** é utilizado para obter informações de suas conexões de rede e pode atuar como um sniffer. Para listar as interfaces de rede que ele pode escutar utilize a Listagem 9.46.

Listagem 9.46: Interfaces que podem ser utilizadas com tcpdump

```
$ tcpdump -D  
1.en0  
2/fw0  
3.bridge0  
4.utun0  
5.en1  
6.en2  
7.p2p0  
8.lo0
```

Para realizar a captura de pacotes da interface de rede utilize a Listagem 9.47. Para sair digite <Ctrl+C>.

Listagem 9.47: Capturando Pacotes da Interface de Rede Sem Fio

```
$ head teste.txt  
07:36:08.930576 IP 72-44-118-73.spartan-net.net.51413 > 192.168.0.101.24874: UDP, length  
1402  
07:36:08.935776 IP 72-44-118-73.spartan-net.net.51413 > 192.168.0.101.24874: UDP, length  
1402  
07:36:08.935911 IP 192.168.0.101.24874 > 72-44-118-73.spartan-net.net.51413: UDP, length  
20  
07:36:08.940160 IP 72-44-118-73.spartan-net.net.51413 > 192.168.0.101.24874: UDP, length  
1402  
07:36:08.940268 IP 192.168.0.101.24874 > 78.194.14.217.51412: UDP, length 20
```

9.20 Navegando no Terminal

O aplicativo **lynx** permite a navegação na Internet no terminal. Para utilizar este aplicativo primeiro realize a instalação como na Listagem 9.48.

Listagem 9.48: Instalação do lynx

```
$ sudo apt-get install lynx  
Password:
```

Depois é só utilizar através do comando da Listagem 9.49.

Listagem 9.49: Utilizando o lynx

```
$lynx www.vivas.eng.br
```

A Figura 9.2 apresenta o resultado do comando.



Figura 9.2: Interface do lynx

9.21 Baixando Sites com wget

Para baixar um site inteiro podemos utilizar o comando **wget**. A sintaxe é apresentada na Listagem 9.50 e para interromper digite <Ctrl+C>.

Listagem 9.50: Baixando Sites com wget

```
$ wget --recursive www.vivas.eng.br  
--2013-12-18 07:49:33-- http://www.vivas.eng.br/  
Resolvendo www.vivas.eng.br (www.vivas.eng.br)... 208.115.217.250  
Conectando-se a www.vivas.eng.br (www.vivas.eng.br)|208.115.217.250|:80... conectado.  
A requisição HTTP foi enviada, aguardando resposta... 200 OK
```

Tamanho: 7032 (6,9K) [text/html]
Salvando em: www.vivas.eng.br/index.html

100% [=====>] 7.032
11,3K/s em 0,6s

2013-12-18 07:49:35 (11,3 KB/s) - www.vivas.eng.br/index.html salvo [7032/7032]

Capítulo 10 | Gerenciamento de Pacotes

O Linux utiliza um repositório de pacotes e todas as operações de instalação e remoção podem ser feitas utilizando comandos. Um repositório é um servidor onde os pacotes estão armazenados. Iremos tratar aqui do gerenciamento de pacotes das distribuições da família Debian (Ubuntu, Linux Mint, TAILS, Knoppix, dentre outras). Para instalar você digita o comando e o nome do pacote, nada além disto. Sua máquina entra em contato com o servidor, faz o download do pacote e depois instala automaticamente o software.

10.1 Atualização de Pacotes

Para atualizar a listagem dos pacotes disponíveis utilizamos o comando **apt-get**. A Listagem 10.1 ilustra o comando.

Listagem 10.1: Atualização da Lista de Pacotes Disponíveis

```
apt-get update
```

10.2 Atualizando a Distribuição

A medida que o tempo vai passando novas versões de software são disponibilizadas. Diversas atualizações de segurança são realizadas em cada semana. Para manter sua distribuição atualizada você precisa utilizar o comando upgrade como na Listagem 10.2

Listagem 10.2: Atualização de Pacotes

```
apt-get upgrade
```

10.3 Instalando Softwares

Para instalar um novo pacote você precisa saber o nome do software e utilizar o comando install como na Listagem 10.3.

Listagem 10.3: Instalação do Pacote vim

```
apt-get install vim
```

10.4 Removendo Pacotes

Para remover um pacote específico você precisa saber o nome do software e utilizar o comando `remove` como na Listagem 10.4.

Listagem 10.4: Removendo o Pacote vim

```
apt-get remove vim
```

10.5 Instalando Software no Fedora

Para instalar um novo pacote no Fedora você precisa utilizar o comando `dnf`. A Listagem 10.5 apresenta o procedimento para instalação do pacote `lshw`.

Listagem 10.5: Instalação do Pacote lshw no Fedora

```
dnf install lshw
```

Capítulo 11 | Comandos Úteis

11.1 Comando unit

O comando **unit** é utilizado para realizar conversões entre unidades. A Listagem 11.1 mostra como converter de metros para quilômetros.

Listagem 11.1: Exemplos de utilização do comando units

```
$ units 5inches cm
      * 12.7
      / 0.078740157
$ units 1mile km
      * 1.609344
      / 0.62137119
$ units
Currency exchange rates from www.timegenie.com on 2014-04-02
2866 units, 109 prefixes, 79 nonlinear units

You have: 10 ounces
You want: grams
      * 283.49523
      / 0.0035273962
```

11.2 Comando yes

O comando **yes** é utilizado para responder automaticamente a perguntas em scripts. A Listagem 11.2 mostra um exemplo de utilização. Para terminar digite <Ctrl+C>.

Listagem 11.2: Exemplo de utilização do comando yes para responder automaticamente a perguntas com ‘yes’

```
$ touch file1 file2 file3 && yes | rm -i file1 file2 file3
```

Ele pode ser utilizado para imprimir mensagens repetidas indefinidamente em seu terminal como na Listagem 11.3.

Listagem 11.3: Imprimindo uma mensagem indefinidamente no terminal utilizando o yes

```
$ yes 'hoje é sexta!'
hoje é sexta!
hoje é sexta!
hoje é sexta!
hoje é sexta!
```

hoje é sexta!^C

Em um script para compilar texto em Latex utilizo o comando **yes** para responder **r** quando ocorre erro no processamento. A Listagem 11.4 apresenta o exemplo do uso do comando.

Listagem 11.4: Utilizando o Comando yes para Processamento de Latex

```
#!/bin/bash
yes r | pdflatex artigo.tex
bibtex biblio
makeindex artigo
```

Capítulo 12 | Comandos Divertidos

12.1 Comando cowsay

O comando **cowsay** funciona apenas no Linux. Para utilizar você precisa realizar a instalação do mesmo, Listagem 12.1 para Debian e Ubuntu. Para Fedora utilize o comando da Listagem 12.2.

Listagem 12.1: Instalação do Comando cowsay no Debian/Ubuntu

```
$ sudo apt-get install cowsay  
Password:
```

Listagem 12.2: Instalação do Comando cowsay no Fedora

```
[root@musashi ~]# dnf install cowsay
```

Após a instalação o comando está pronto para o uso, Listagem 12.3.

Listagem 12.3: Comando cowsay

```
[avivas@musashi ~]$ cowsay  
< Eu amo Linux >  
-----  
 \  ^__^  
  \  (oo)\_____  
   (__)\ \        )\/\|  
     ||----w |  
     ||     ||
```

12.2 Comando xcowsay

O comando **xcowsay** funciona apenas no Linux. Para utilizar você precisa realizar a instalação do mesmo, Listagem 12.4.

Listagem 12.4: Comando cowsay

```
# para Ubuntu e Debian  
$ sudo apt-get install xcowsay  
Password:  
# para Fedora  
[root@musashi ~]$ dnf install xcowsay
```

Após a instalação o comando está pronto para o uso, Listagem 12.5.

Listagem 12.5: Comando cowsay

```
$ xcowsay Eu amo Linux
```

A Figura 12.1 apresenta o resultado.



Figura 12.1: Comando xcowsay

12.3 Comando fortune

O comando **fortune** envia frases aleatórias no terminal. A Listagem 12.6 explica como utilizar o comando.

Listagem 12.6: Comando fortune

```
[avivas@musashi ~]$ fortune  
I think... I think it's in my basement... Let me go upstairs and check.  
-- Escher
```

12.4 Comando xcowfortune

O comando **xcowfortune** comando utiliza o **xcowsay** e o **bffortune** em conjunto. Para utilizar digite o comando da Listagem 12.7.

Listagem 12.7: Comando xcowfortune

```
$ xcowfortune
```

A Figura 12.2 apresenta o resultado.

12.5 Comando sl

Outro comando interessante é o **sl**. O procedimento de instalação é apresentado na Listagem 12.8. Para executá-lo utilize a Listagem 12.9 e o resultado é apresentado na Figura 12.3.

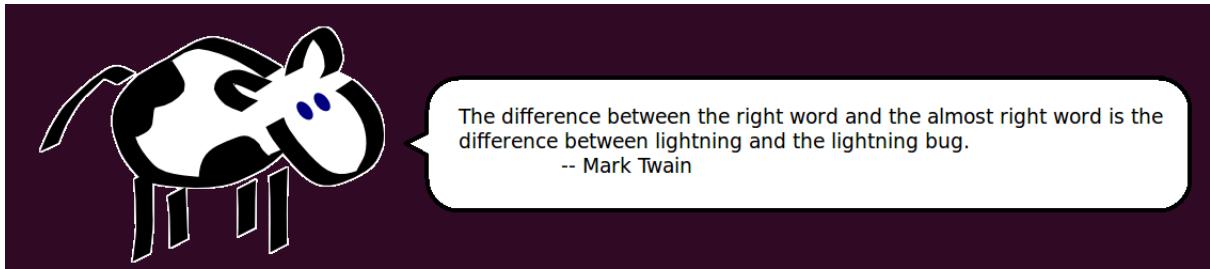


Figura 12.2: Comando xcowfortune

Listagem 12.8: Comando sl

```
# Ubuntu ou Debian
$ sudo apt-get install sl
# Fedora
[root@musashi ~]# dnf install sl
```

Listagem 12.9: Comando sl

```
$ sl
```

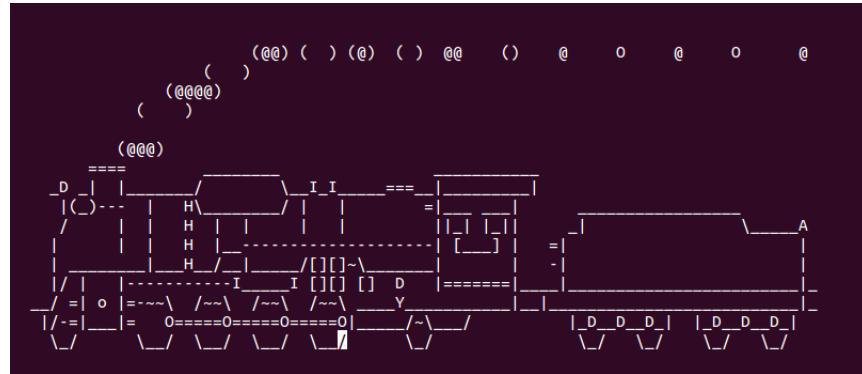


Figura 12.3: Comando sl

12.6 Comando xeyes

O comando **xeyes** apresenta dois olhos que acompanham o posicionamento do mouse. Para utilizá-lo digite apenas **xeyes** no terminal como apresentado na Figura 12.4. Para instalar o aplicativo utilize o comando da Listagem 12.10.

Listagem 12.10: Comando xeyes

```
# Ubuntu ou Debian
$ sudo apt-get install xeyes
# Fedora
[root@musashi ~]# dnf install xeyes
```



Figura 12.4: Comando xyes

12.7 Comando oneko

Oneko é um comando que cria um gatinho no seu desktop. O ponteiro de seu mouse vira um ratinho e o gatinho sempre fica tentando capturá-lo. A Listagem 12.11 mostra como instalar e como rodar o programa.

Listagem 12.11: Comando oneko

```
$ sudo apt-get install oneko  
$ oneko
```

Se quiser mudar a imagem para um cachorro digite **\$ oneko -dog** e para um tigre digite **\$ oneko -tora**.

Capítulo 13 | Comandos para Sistema de Arquivos

13.1 Entendendo Discos e Partições

Seu computador possui um ou mais discos rígidos que são utilizados para armazenar todas as informações (arquivos e programas). Cada disco pode ter uma ou mais partições e cada partição pode armazenar um sistema operacional diferente.

Os discos são montados no diretório `/dev` nos subdiretórios `sda`, `b`, etc.. Se seu computador tiver três discos ele terá os dispositivos `/dev/sda` para o disco 1, `/dev/sdb` para o disco 2 e `sdc` para o disco 3.

As partições são os valores numéricos após a indicação do disco. Se seu computador tem um disco com 6 partições você terá os seguintes dispositivos `/dev/sda1`, `/dev/sda2`, `/dev/sda3`, `/dev/sda4`, `/dev/sda5` e `/dev/sda6`.

13.2 Imprimir Tabela de Partições do Linux

O comando **parted** pode ser utilizado para imprimir a tabela de partições de seu disco rígido. A Listagem 13.1 apresenta o procedimento para listar as partições do seu disco rígido.

Listagem 13.1: Listando as Partições do Disco Rígido

```
root@musashi:~# parted /dev/sda 'print'
Model: ATA WDC WD5000AAKS-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type      File system    Flags
 1       32,3kB  197MB   197MB   primary   fat16          diag
 2       198MB   985MB   786MB   primary   ntfs
 3       985MB   262GB   261GB   primary   ntfs
 4       262GB   500GB   238GB   extended
 5       262GB   326GB   64,0GB  logical   linux-swap(v1)
 6       326GB   500GB   174GB   logical   ext4          boot
```

13.3 Obtendo Informações sobre o Disco com fdisk

O comando **fdisk** pode ser utilizado para obter informações detalhadas sobre suas partições. A Figura 13.1 apresenta o resultado do comando.

```
[root@musashi ~]# fdisk -l
Disco /dev/sda: 465,8 GiB, 500107862016 bytes, 976773168 setores
Unidades: setor de 1 * 512 = 512 bytes
Tamanho de setor (lógico/físico): 512 bytes / 512 bytes
Tamanho E/S (mínimo/ótimo): 512 bytes / 512 bytes
Tipo de rótulo do disco: dos
Identificador do disco: 0x50000000

Dispositivo Inicializar Início Fim Setores Tamanho Id Tipo
/dev/sda1          *      63 240974 240912 117,6M 6 FAT16
/dev/sda2          *    241664 20758527 20516864 9,8G 7 HPFS/NTFS/exFAT
/dev/sda3          * 20758528 544805064 524046537 249,9G 7 HPFS/NTFS/exFAT
/dev/sda4          * 544806912 976773119 431966208 206G 5 Estendida
/dev/sda5          * 544808960 545832959 1024000 500M 83 Linux
/dev/sda6          * 545835008 976773119 430938112 205,5G 8e Linux LVM

Disco /dev/mapper/fedora-swap: 3,8 GiB, 4026531840 bytes, 7864320 setores
Unidades: setor de 1 * 512 = 512 bytes
Tamanho de setor (lógico/físico): 512 bytes / 512 bytes
Tamanho E/S (mínimo/ótimo): 512 bytes / 512 bytes

Disco /dev/mapper/fedora-root: 50 GiB, 53687091200 bytes, 104857600 setores
Unidades: setor de 1 * 512 = 512 bytes
Tamanho de setor (lógico/físico): 512 bytes / 512 bytes
Tamanho E/S (mínimo/ótimo): 512 bytes / 512 bytes

Disco /dev/mapper/fedora-home: 151,7 GiB, 162919350272 bytes, 318201856 setores
Unidades: setor de 1 * 512 = 512 bytes
Tamanho de setor (lógico/físico): 512 bytes / 512 bytes
Tamanho E/S (mínimo/ótimo): 512 bytes / 512 bytes

Disco /dev/sdb: 465,8 GiB, 500107862016 bytes, 976773168 setores
Unidades: setor de 1 * 512 = 512 bytes
Tamanho de setor (lógico/físico): 512 bytes / 512 bytes
Tamanho E/S (mínimo/ótimo): 512 bytes / 512 bytes
Tipo de rótulo do disco: gpt
Identificador do disco: 03D8F472-1B09-45D2-8667-3D232A61EEE1

Dispositivo Início Fim Setores Tamanho Tipo
/dev/sdb1      40 409639 409600 200M Sistema EFI
/dev/sdb2 409640 976510983 976101344 465,5G Apple HFS/HFS+
```

Figura 13.1: Comando fdisk

13.4 Comando sfdisk

O comando **sfdisk** pode ser utilizado para obter e realizar operações nos discos de seu computador. A Figura 13.2 apresenta o resultado do comando no disco */dev/sda*.

13.5 Listando Informações sobre as Partições com lsblk

O comando **lsblk** pode ser utilizado para imprimir informações sobre o seu disco rígido. A Figura 13.3 apresenta o procedimento para listar as partições do seu disco rígido.

```
[root@musashi ~]# sfdisk /dev/sda -l
Disco /dev/sda: 465,8 GiB, 500107862016 bytes, 976773168 setores
Unidades: setor de 1 * 512 = 512 bytes
Tamanho de setor (lógico/físico): 512 bytes / 512 bytes
Tamanho E/S (mínimo/ótimo): 512 bytes / 512 bytes
Tipo de rótulo do disco: dos
Identificador do disco: 0x50000000

Dispositivo Inicializar Início Fim Setores Tamanho Id Tipo
/dev/sda1          63 240974 240912 117,6M 6 FAT16
/dev/sda2 *        241664 20758527 20516864 9,8G 7 HPFS/NTFS/exFAT
/dev/sda3          20758528 544805064 524046537 249,9G 7 HPFS/NTFS/exFAT
/dev/sda4          544806912 976773119 431966208 206G 5 Estendida
/dev/sda5          544808960 545832959 1024000 500M 83 Linux
/dev/sda6          545835008 976773119 430938112 205,5G 8e Linux LVM
```

Figura 13.2: Comando sfdisk

```
root@musashi:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0 465,8G  0 disk
└─sda1   8:1    0 188,2M  0 part
└─sda2   8:2    0 750M  0 part
└─sda3   8:3    0 243,2G  0 part
└─sda4   8:4    0     1K  0 part
└─sda5   8:5    0 59,6G  0 part [SWAP]
└─sda6   8:6    0 162G  0 part /
sr0     11:0   1 1024M  0 rom
```

Figura 13.3: Comando lsblk

13.6 Listando Informações sobre Permissões do Disco com lsblk

O comando **lsblk** pode ser utilizado para imprimir informações sobre permissões do seu disco rígido e partições. A Figura 13.4 apresenta o procedimento de utilização do comando

```
root@musashi:~# lsblk -m
NAME   SIZE OWNER GROUP MODE
sda   465,8G root  disk brw-rw----
└─sda1 188,2M root  disk brw-rw----
└─sda2 750M root  disk brw-rw----
└─sda3 243,2G root  disk brw-rw----
└─sda4 1K root  disk brw-rw----
└─sda5 59,6G root  disk brw-rw----
└─sda6 162G root  disk brw-rw----
sr0   1024M root cdrom brw-rw----
```

Figura 13.4: Listando Permissões com lsblk

Capítulo 14 | Comandos para Analisar o Desempenho do Linux

14.1 Analisando Consumo de CPU com o Comando sar

O comando **sar** é utilizado para medir o consumo de CPU de qualquer computador. O comando **sar -u** precisa de dois argumentos: a) o número de segundos onde cada leitura é feita e b) a quantidade de relatórios que ele irá imprimir. A Listagem 14.1 apresenta o relatório de consumo de CPU medida a cada 1 segundo durante 10 medições.

Listagem 14.1: Analisando Desempenho da CPU com sar

```
[root@musashi ~]# sar -u 1 10
Linux 4.0.4-301.fc22.x86_64 (musashi.vivascorp) 18-09-2015 _x86_64_ (4 CPU)

15:28:39      CPU    %user    %nice   %system   %iowait   %steal    %idle
15:28:40      all     0,00     0,00     0,00     3,52     0,00    96,48
15:28:41      all     0,00     0,00     0,25     0,25     0,00    99,50
15:28:42      all     0,00     0,00     0,00     0,00     0,00   100,00
15:28:43      all     0,00     0,00     0,00     0,00     0,00   100,00
15:28:44      all     0,00     0,00     0,25     0,25     0,00    99,50
15:28:45      all     0,00     0,00     0,00     0,75     0,00    99,25
15:28:46      all     0,00     0,00     0,00     0,00     0,00   100,00
15:28:47      all     0,00     0,00     0,00     0,00     0,00   100,00
15:28:48      all     0,00     0,00     0,25     0,00     0,00    99,75
15:28:49      all     0,00     0,00     0,00     0,00     0,00   100,00
Média:        all     0,00     0,00     0,08     0,48     0,00    99,45
```

Onde **%usr** é a quantidade de CPU utilizada pelo sistema com processos dos usuários , **%sys** é o percentual de processo consumido por processos do sistema, **%idle** é o percentual de CPU ocioso e **%nice** é o percentual de CPU consumidos por processos que tenham algum tipo de prioridade de escalonamento.

14.2 Analisando Desempenho de CPU com mpstat

O comando **mpstat** é utilizado para medir o desempenho de CPU de qualquer computador. É possível analisar como está cada núcleo de sua CPU. A Listagem 14.2 apresenta o relatório de desempenho de CPU para todos os núcleos.

Listagem 14.2: Analisando Desempenho de Todos os Núcleos com mpstat

```
[root@musashi ~]# mpstat -P ALL
```

```
Linux 4.0.4-301.fc22.x86_64 (musashi.vivascorp) 18-09-2015 _x86_64_ (4 CPU)
```

```
15:31:59 CPU %usr %nice %sys %iowait %irq %soft %steal %guest %gnice %idle
15:31:59 all 0,09 0,01 0,04 0,52 0,00 0,00 0,00 0,00 0,00 0,00 99,35
15:31:59 0 0,07 0,00 0,04 0,70 0,00 0,00 0,00 0,00 0,00 0,00 99,18
15:31:59 1 0,06 0,01 0,04 0,58 0,00 0,00 0,00 0,00 0,00 0,00 99,31
15:31:59 2 0,10 0,01 0,03 0,40 0,00 0,00 0,00 0,00 0,00 0,00 99,46
15:31:59 3 0,14 0,00 0,04 0,39 0,00 0,00 0,00 0,00 0,00 0,00 99,43
```

14.3 Estatísticas de Entrada e Saída com iostat

O comando **iostat** é utilizado para fornecer estatísticas de entrada e saída e CPU de qualquer computador. A Listagem 14.3 apresenta o relatório das estatísticas de entrada e saída..

Listagem 14.3: Analisando Estatísticas de Entrada e Saída com iostat

```
[root@musashi ~]# iostat
Linux 4.0.4-301.fc22.x86_64 (musashi.vivascorp) 18-09-2015 _x86_64_ (4 CPU)

avg-cpu: %user %nice %system %iowait %steal %idle
          0,09   0,01   0,04   0,51   0,00  99,35

Device:      tps   kB_read/s   kB_wrtn/s   kB_read   kB_wrtn
sda       3,48     7,97     29,52    724079   2681096
dm-0      0,00     0,01     0,00     1156        0
dm-1      3,80     7,47     25,17   678137   2285372
dm-2      0,09     0,19     4,36    17629   395708
sdb       0,01     0,06     0,00     5386        1
```

14.4 Analisando a Memória com vm_stat

Listagem 14.4: Analisando a Memória com vm_stat

```
$ vm_stat -c 5
Mach Virtual Memory Statistics: (page size of 4096 bytes)
 free      active      specul      inactive      throttle      wired      prgable      faults      copy      0fill      reactive      purged      file-backed      anonymous
 627443  2405915  362634  283221      0  514661  115323  185938K  1309331  24597401      836      896    819123  2232647
      0      0      0      0      0  1162355      0      0      0      0      0      0      0      0
 626592  2405083  363008  283317      0  515753  115323  85428      38      3248      0      0    819500  2231908
      0      0      0      0      0      3      0      0      0      0      0      0      0      0
 626649  2406337  363009  283317      0  514342  115323  82921      71      1499      0      0    819501  2233162
      0      0      0      0      0      0      0      0      0      0      0      0      0      0
 626361  2405981  363007  283319      0  515015  115323  83764      35      1718      0      0    819501  2232806
      0      0      0      0      0      0      0      0      0      0      0      0      0      0
 625807  2407119  363008  283320      0  514342  115356  84127      70      1478      0      0    819502  2233945
      0      0      0      0      0      0      0      0      0      0      0      0      0      0
```

14.5 Comando pidstat

O comando **pidstat** dá estatísticas sobre todos os processos que estão rodando em sua máquina. A Listagem 14.5 apresenta o resultado do comando.

Listagem 14.5: Comando pidstat

```
[root@musashi ~]# pidstat
Linux 4.0.4-301.fc22.x86_64 (musashi.vivascorp) 18-09-2015 _x86_64_ (4 CPU)

 16:02:01   UID     PID   %usr %system  %guest   %CPU    CPU  Command
 16:02:01     0       1   0,00   0,00   0,00   0,00      3  systemd
 16:02:01     0       2   0,00   0,00   0,00   0,00      0  kthreadd
 16:02:01     0       3   0,00   0,00   0,00   0,00      0  ksoftirqd/0
 16:02:01     0       7   0,00   0,06   0,00   0,06      1  rcu_sched
 16:02:01     0       9   0,00   0,02   0,00   0,02      1  rcuos/0
 16:02:01     0      11   0,00   0,00   0,00   0,00      0  migration/0
 16:02:01     0      12   0,00   0,00   0,00   0,00      0  watchdog/0
 16:02:01     0      13   0,00   0,00   0,00   0,00      1  watchdog/1
 16:02:01     0      14   0,00   0,00   0,00   0,00      1  migration/1
 16:02:01     0      15   0,00   0,00   0,00   0,00      1  ksoftirqd/1
 16:02:01     0      18   0,00   0,01   0,00   0,01      0  rcuos/1
 16:02:01     0      20   0,00   0,00   0,00   0,00      2  watchdog/2
 16:02:01     0      21   0,00   0,00   0,00   0,00      2  migration/2
 16:02:01     0      22   0,00   0,00   0,00   0,00      2  ksoftirqd/2
 16:02:01     0      25   0,00   0,02   0,00   0,02      3  rcuos/2
 16:02:01     0      27   0,00   0,00   0,00   0,00      3  watchdog/3
 16:02:01     0      28   0,00   0,00   0,00   0,00      3  migration/3
 16:02:01     0      29   0,00   0,00   0,00   0,00      3  ksoftirqd/3
 16:02:01     0      32   0,00   0,01   0,00   0,01      1  rcuos/3
 16:02:01     0      52   0,00   0,00   0,00   0,00      2  fsnotify_mark
 16:02:01     0     131   0,00   0,00   0,00   0,00      3  kauditd
 16:02:01     0     319   0,00   0,00   0,00   0,00      2  kworker/2:1H
 16:02:01     0     320   0,00   0,00   0,00   0,00      3  kworker/3:1H
 16:02:01     0     322   0,00   0,00   0,00   0,00      1  kworker/1:1H
 16:02:01     0     328   0,00   0,00   0,00   0,00      0  kworker/0:1H
 16:02:01     0     420   0,00   0,00   0,00   0,00      1  jbd2/dm-1-8
 16:02:01     0     506   0,00   0,01   0,00   0,01      1  systemd-journal
 16:02:01     0     525   0,00   0,00   0,00   0,00      0  lvmetad
 16:02:01     0     550   0,00   0,00   0,00   0,00      2  systemd-udevd
 16:02:01     0     580   0,00   0,00   0,00   0,00      1  ips-adjust
 16:02:01     0     581   0,00   0,01   0,00   0,01      1  ips-monitor
 16:02:01     0     687   0,00   0,00   0,00   0,00      0  jbd2/dm-2-8
 16:02:01     0     708   0,00   0,00   0,00   0,00      1  auditd
 16:02:01     0     717   0,00   0,00   0,00   0,00      1  alsactl
 16:02:01     0     719   0,00   0,00   0,00   0,00      3  firewalld
 16:02:01     0     720   0,00   0,00   0,00   0,00      0  systemd-logind
 16:02:01     81     721   0,01   0,00   0,00   0,01      1  dbus-daemon
 16:02:01     0     727   0,00   0,00   0,00   0,00      3  audispd
 16:02:01     0     729   0,00   0,00   0,00   0,00      3  sedispatch
 16:02:01     70     734   0,00   0,00   0,00   0,00      0  avahi-daemon
 16:02:01    172     735   0,00   0,00   0,00   0,00      3  rtkit-daemon
 16:02:01     0     738   0,00   0,00   0,00   0,00      1  accounts-daemon
 16:02:01     0     739   0,00   0,00   0,00   0,00      0  abrtd
 16:02:01    991     747   0,00   0,00   0,00   0,00      3  chronyd
 16:02:01     0     757   0,00   0,00   0,00   0,00      1  gssproxy
 16:02:01     0     769   0,00   0,00   0,00   0,00      3  abrt-watch-log
 16:02:01     0     771   0,00   0,00   0,00   0,00      3  abrt-dump-journ
 16:02:01   995     782   0,00   0,00   0,00   0,00      3  polkitd
 16:02:01     0     839   0,01   0,01   0,00   0,01      2  NetworkManager
 16:02:01     0     904   0,00   0,01   0,00   0,01      3  kworker/u8:2
 16:02:01     0     916   0,00   0,00   0,00   0,00      2  libvirtd
 16:02:01     0     989   0,00   0,00   0,00   0,00      1  crond
 16:02:01     0     990   0,00   0,00   0,00   0,00      1  gdm
 16:02:01     0     992   0,00   0,00   0,00   0,00      0  atd
 16:02:01     0    1037   0,00   0,00   0,00   0,00      3  wpa_supplicant
 16:02:01     0    1038   0,00   0,00   0,00   0,00      2  kworker/2:1
 16:02:01     0    1344   0,00   0,00   0,00   0,00      0  gdm-session-wor
 16:02:01     42    1356   0,00   0,00   0,00   0,00      1  Xorg
 16:02:01     42    1385   0,00   0,00   0,00   0,00      0  dbus-daemon
 16:02:01     42    1388   0,00   0,00   0,00   0,00      1  gnome-session
 16:02:01     42    1391   0,00   0,00   0,00   0,00      0  at-spi-bus-laun
 16:02:01     42    1394   0,00   0,00   0,00   0,00      1  gvfsd
 16:02:01     42    1404   0,00   0,00   0,00   0,00      2  at-spi2-registr
 16:02:01     42    1422   0,00   0,00   0,00   0,00      2  gnome-settings-
 16:02:01     0    1428   0,00   0,00   0,00   0,00      0  upowerd
 16:02:01     0    1454   0,00   0,00   0,00   0,00      3  kworker/3:1
 16:02:01     42    1455   0,03   0,00   0,00   0,03      0  gnome-shell
 16:02:01   994    1457   0,00   0,00   0,00   0,00      3  colord
 16:02:01     42    1470   0,00   0,00   0,00   0,00      0  pulseaudio
 16:02:01     42    1519   0,00   0,00   0,00   0,00      3  ibus-daemon
 16:02:01     42    1524   0,00   0,00   0,00   0,00      2  ibus-dconf
 16:02:01     42    1526   0,00   0,00   0,00   0,00      1  ibus-x11
 16:02:01     0    1532   0,00   0,00   0,00   0,00      1  dhclient
 16:02:01     0    1536   0,12   0,02   0,00   0,14      0  packagekitd
 16:02:01     42    1538   0,00   0,00   0,00   0,00      1  gvfs-udisks2-vo
 16:02:01     0    1542   0,01   0,00   0,00   0,02      3  udisksd
 16:02:01     42    1566   0,00   0,00   0,00   0,00      1  goa-daemon
 16:02:01     42    1572   0,00   0,00   0,00   0,00      1  mission-control
 16:02:01     42    1629   0,00   0,00   0,00   0,00      1  ibus-engine-sim
 16:02:01     0    1662   0,00   0,00   0,00   0,00      0  kworker/0:1
 16:02:01     0    1663   0,00   0,00   0,00   0,00      2  kworker/u8:1
 16:02:01     0    1685   0,00   0,00   0,00   0,00      0  sshd
 16:02:01   1000   1690   0,00   0,00   0,00   0,00      2  sshd
 16:02:01   1000   1693   0,00   0,00   0,00   0,00      0  bash
 16:02:01     0    1694   0,00   0,00   0,00   0,00      0  gdm-session-wor
 16:02:01   1000   1707   0,00   0,00   0,00   0,00      1  systemd
 16:02:01   1000   1732   0,00   0,00   0,00   0,00      3  gnome-keyring-d
 16:02:01   1000   1766   0,00   0,00   0,00   0,01      3  Xorg
 16:02:01   1000   1829   0,00   0,00   0,00   0,00      1  dbus-daemon
 16:02:01   1000   1834   0,00   0,00   0,00   0,00      2  gnome-session
```

16:02:01	1000	1921	0,00	0,00	0,00	0,00	1	at-spi-bus-laun
16:02:01	1000	1924	0,00	0,00	0,00	0,00	0	gvfsd
16:02:01	1000	1928	0,00	0,00	0,00	0,00	2	gvfsd-fuse
16:02:01	1000	1938	0,00	0,00	0,00	0,00	0	dbus-daemon
16:02:01	1000	1943	0,00	0,00	0,00	0,00	0	at-spi2-registr
16:02:01	1000	1959	0,00	0,00	0,00	0,00	1	su
16:02:01	0	1967	0,00	0,00	0,00	0,00	1	kworker/1:1
16:02:01	0	1970	0,00	0,00	0,00	0,00	0	bash
16:02:01	1000	1971	0,00	0,00	0,00	0,00	1	gnome-settings-
16:02:01	1000	1976	0,00	0,00	0,00	0,00	3	pulseaudio
16:02:01	1000	2037	0,05	0,00	0,00	0,05	2	gnome-shell
16:02:01	0	2041	0,00	0,00	0,00	0,00	2	cupsd
16:02:01	1000	2051	0,00	0,00	0,00	0,00	3	gsd-printer
16:02:01	1000	2059	0,00	0,00	0,00	0,00	3	gnome-shell-cal
16:02:01	1000	2060	0,00	0,00	0,00	0,00	0	ibus-daemon
16:02:01	1000	2065	0,00	0,00	0,00	0,00	2	ibus-dconf
16:02:01	1000	2067	0,00	0,00	0,00	0,00	2	ibus-x11
16:02:01	1000	2076	0,00	0,00	0,00	0,00	1	evolution-sourc
16:02:01	0	2091	0,00	0,00	0,00	0,00	2	kworker/2:2
16:02:01	1000	2097	0,00	0,00	0,00	0,00	3	mission-control
16:02:01	1000	2102	0,00	0,00	0,00	0,00	1	goa-daemon
16:02:01	1000	2116	0,00	0,00	0,00	0,00	2	gvfs-udisks2-vo
16:02:01	0	2183	0,00	0,00	0,00	0,00	0	pidstat
16:02:01	1000	2219	0,00	0,00	0,00	0,00	0	tracker-store
16:02:01	1000	2220	0,00	0,00	0,00	0,00	0	abrt-applet
16:02:01	1000	2221	0,00	0,00	0,00	0,00	1	tracker-miner-f
16:02:01	1000	2227	0,00	0,00	0,00	0,00	1	evolution-alarm
16:02:01	1000	2239	0,00	0,00	0,00	0,00	1	gnome-software
16:02:01	1000	2255	0,00	0,00	0,00	0,00	0	ibus-engine-sim
16:02:01	1000	2302	0,00	0,00	0,00	0,00	1	seapplet
16:02:01	1000	2316	0,00	0,00	0,00	0,00	0	tracker-miner-u
16:02:01	1000	2318	0,00	0,00	0,00	0,00	1	evolution-caLEN
16:02:01	1000	2324	0,00	0,00	0,00	0,00	3	tracker-extract
16:02:01	1000	2327	0,00	0,00	0,00	0,00	0	tracker-miner-a
16:02:01	1000	2346	0,00	0,00	0,00	0,00	1	evolution-caLEN
16:02:01	1000	2364	0,00	0,00	0,00	0,00	2	evolution-addre
16:02:01	1000	2367	0,00	0,00	0,00	0,00	2	evolution-caLEN
16:02:01	1000	2391	0,00	0,00	0,00	0,00	3	evolution-addre
16:02:01	1000	2425	0,00	0,00	0,00	0,00	2	gnome-terminal-
16:02:01	1000	2429	0,00	0,00	0,00	0,00	3	bash
16:02:01	1000	2559	0,05	0,01	0,00	0,06	1	firefox
16:02:01	0	2814	0,00	0,00	0,00	0,00	2	sshd
16:02:01	0	3456	0,00	0,00	0,00	0,00	3	usb-storage

14.6 Comando top

O comando **top** é utilizado para fornecer estatísticas de uso de CPU, memória e diversas estatísticas de uso. A Figura 14.1 apresenta o relatório das estatísticas de uso de CPU. Para sair digite **q**.

```

top - 15:41:04 up 1 day, 1:18, 3 users, load average: 0,00, 0,01, 0,05
Tasks: 216 total, 1 running, 215 sleeping, 0 stopped, 0 zombie
%Cpu0 : 0,0 us, 0,3 sy, 0,0 ni, 99,7 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
%Cpu1 : 0,0 us, 0,0 sy, 0,0 ni, 97,0 id, 3,0 wa, 0,0 hi, 0,0 si, 0,0 st
%Cpu2 : 0,0 us, 0,0 sy, 0,0 ni, 100,0 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
%Cpu3 : 0,0 us, 0,0 sy, 0,0 ni, 99,3 id, 0,7 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem : 3843788 total, 1452360 free, 950372 used, 1441056 buff/cache
KiB Swap: 3932156 total, 3932156 free, 0 used. 2618640 avail Mem

```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	194536	9124	4204	S	0,0	0,2	0:04.09	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.04	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.51	ksoftirqd/0
5	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/0:0H
7	root	20	0	0	0	0	S	0,0	0,0	0:53.32	rcu_sched
8	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_bh
9	root	20	0	0	0	0	S	0,0	0,0	0:17.99	rcuos/0
10	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcuob/0
11	root	rt	0	0	0	0	S	0,0	0,0	0:00.01	migration/0
12	root	rt	0	0	0	0	S	0,0	0,0	0:00.30	watchdog/0
13	root	rt	0	0	0	0	S	0,0	0,0	0:00.29	watchdog/1
14	root	rt	0	0	0	0	S	0,0	0,0	0:00.01	migration/1
15	root	20	0	0	0	0	S	0,0	0,0	0:00.51	ksoftirqd/1
17	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/1:0H
18	root	20	0	0	0	0	S	0,0	0,0	0:06.36	rcuos/1
19	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcuob/1
20	root	rt	0	0	0	0	S	0,0	0,0	0:00.27	watchdog/2
21	root	rt	0	0	0	0	S	0,0	0,0	0:00.01	migration/2
22	root	20	0	0	0	0	S	0,0	0,0	0:00.34	ksoftirqd/2
24	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/2:0H
25	root	20	0	0	0	0	S	0,0	0,0	0:15.34	rcuos/2
26	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcuob/2
27	root	rt	0	0	0	0	S	0,0	0,0	0:00.25	watchdog/3
28	root	rt	0	0	0	0	S	0,0	0,0	0:00.01	migration/3
29	root	20	0	0	0	0	S	0,0	0,0	0:00.33	ksoftirqd/3

Figura 14.1: Uso do top para Obter Estatísticas de CPU

Capítulo 15 | Verificando Configuração de Hardware e Software

15.1 Visualizando Informações sobre a Versão do Kernel

Para verificar informações sobre o seu Linux utilize o comando **uname** como na Listagem 15.1.

Listagem 15.1: Versão do Kernel

```
[root@musashi ~]# uname -a
Linux musashi.vivascorp 4.0.4-301.fc22.x86_64 #1 SMP Thu May 21 13:10:33 UTC 2015 x86_64
x86_64 GNU/Linux
```

15.2 Verificando sua Distribuição

Para verificar informações sobre sua distribuição utilize o comando **cat** como na Listagem 15.2.

Listagem 15.2: Verificando sua Distribuição

```
[root@musashi ~]# head -n1 /etc/issue
Fedora release 22 (Twenty Two)
```

15.3 Visualizando Informações sobre a sua CPU

O comando **lscpu** é utilizado para listar todas as informações sobre a sua CPU. A Figura 15.1 apresenta o resultado do comando.

Outra maneira é consultar o diretório */proc* como na Listagem 15.3.

Listagem 15.3: Verificando a CPU

```
[root@musashi ~]# grep "model name" /proc/cpuinfo
model name : Intel(R) Core(TM) i5 CPU M 460 @ 2.53GHz
model name : Intel(R) Core(TM) i5 CPU M 460 @ 2.53GHz
model name : Intel(R) Core(TM) i5 CPU M 460 @ 2.53GHz
model name : Intel(R) Core(TM) i5 CPU M 460 @ 2.53GHz
```

```
[avivas@musashi ~]$ lscpu
Arquitetura:          x86_64
Modo(s) operacional da CPU:32-bit, 64-bit
Ordem dos bytes:       Little Endian
CPU(s):                4
Lista de CPU(s) on-line:0-3
Thread(s) per núcleo 2
Núcleo(s) por soquete:2
Soquete(s):           1
Nó(s) de NUMA:        1
ID de fornecedor:    GenuineIntel
Família da CPU:      6
Modelo:               37
Nome do modelo:       Intel(R) Core(TM) i5 CPU      M 460 @ 2.53GHz
Step:                 5
CPU MHz:              1197.000
CPU MHz máx.:         2528,0000
CPU MHz mín.:         1197,0000
BogoMIPS:             5054.25
Virtualização:       VT-x
cache de L1d:         32K
cache de L1i:         32K
cache de L2:           256K
cache de L3:           3072K
CPU(s) de nó NUMA:   0-3
```

Figura 15.1: Obtendo Informações sobre a CPU

15.4 Visualizando Informações sobre os Dispositivos USB

O comando **lsusb** é utilizado para listar todas as informações sobre as conexões USB. A Figura 15.2 apresenta o resultado do comando.

```
[avivas@musashi ~]$ lsusb
Bus 002 Device 003: ID 0c45:6481 Microdia
Bus 002 Device 002: ID 8087:0020 Intel Corp. Integrated Rate Matching Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 004: ID 138a:0008 Validity Sensors, Inc. VFS300 Fingerprint Reader
Bus 001 Device 006: ID 413c:8162 Dell Computer Corp. Integrated Touchpad [Synaptics]
Bus 001 Device 005: ID 413c:8161 Dell Computer Corp. Integrated Keyboard
Bus 001 Device 003: ID 0a5c:4500 Broadcom Corp. BCM2046B1 USB 2.0 Hub (part of BCM2046 Bluetooth)
Bus 001 Device 002: ID 8087:0020 Intel Corp. Integrated Rate Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Figura 15.2: Obtendo Informações sobre as Conexões USB

A Figura 15.3 apresenta o resultado do comando quando é ligado um HD Externo em uma porta USB. Como pode ser percebido aparece o dispositivo Samsung conectado ao barramento.

```
[avivas@musashi ~]$ lsusb
Bus 002 Device 003: ID 0c45:6481 Microdia
Bus 002 Device 004: ID 04e8:1f05 Samsung Electronics Co., Ltd S2 Portable [JMicron] (500GB)
Bus 002 Device 002: ID 8087:0020 Intel Corp. Integrated Rate Matching Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 004: ID 138a:0008 Validity Sensors, Inc. VFS300 Fingerprint Reader
Bus 001 Device 006: ID 413c:8162 Dell Computer Corp. Integrated Touchpad [Synaptics]
Bus 001 Device 005: ID 413c:8161 Dell Computer Corp. Integrated Keyboard
Bus 001 Device 003: ID 0a5c:4500 Broadcom Corp. BCM2046B1 USB 2.0 Hub (part of BCM2046 Bluetooth)
Bus 001 Device 002: ID 8087:0020 Intel Corp. Integrated Rate Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Figura 15.3: Exemplo de Conexão de Dispositivo USB

15.5 Listando Todos os Dispositivos PCI

O comando **lspci** é utilizado para listar todas as informações sobre os dispositivos PCI. A Figura 15.4 apresenta o resultado do comando.

```
[avivas@musashi ~]$ lspci
00:00.0 Host bridge: Intel Corporation Core Processor DRAM Controller (rev 18)
00:01.0 PCI bridge: Intel Corporation Core Processor PCI Express x16 Root Port (rev 18)
00:02.0 VGA compatible controller: Intel Corporation Core Processor Integrated Graphics Controller (rev 18)
00:16.0 Communication controller: Intel Corporation 5 Series/3400 Series Chipset HECI Controller (rev 06)
00:1a.0 USB controller: Intel Corporation 5 Series/3400 Series Chipset USB2 Enhanced Host Controller (rev 06)
00:1b.0 Audio device: Intel Corporation 5 Series/3400 Series Chipset High Definition Audio (rev 06)
00:1c.0 PCI bridge: Intel Corporation 5 Series/3400 Series Chipset PCI Express Root Port 1 (rev 06)
00:1c.1 PCI bridge: Intel Corporation 5 Series/3400 Series Chipset PCI Express Root Port 2 (rev 06)
00:1c.2 PCI bridge: Intel Corporation 5 Series/3400 Series Chipset PCI Express Root Port 3 (rev 06)
00:1c.4 PCI bridge: Intel Corporation 5 Series/3400 Series Chipset PCI Express Root Port 5 (rev 06)
00:1d.0 USB controller: Intel Corporation 5 Series/3400 Series Chipset USB2 Enhanced Host Controller (rev 06)
00:1e.0 PCI bridge: Intel Corporation 82801 Mobile PCI Bridge (rev a6)
00:1f.0 ISA bridge: Intel Corporation Mobile 5 Series Chipset LPC Interface Controller (rev 06)
00:1f.2 SATA controller: Intel Corporation 5 Series/3400 Series Chipset 6 port SATA AHCI Controller (rev 06)
00:1f.3 SMBus: Intel Corporation 5 Series/3400 Series Chipset SMBus Controller (rev 06)
00:1f.6 Signal processing controller: Intel Corporation 5 Series/3400 Series Chipset Thermal Subsystem (rev 06)
01:00.0 VGA compatible controller: NVIDIA Corporation GT218M [GeForce 310M] (rev a2)
01:00.1 Audio device: NVIDIA Corporation High Definition Audio Controller (rev a1)
12:00.0 Network controller: Broadcom Corporation BCM4313 802.11bgn Wireless Network Adapter (rev 01)
13:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller (rev 03)
ff:00.0 Host bridge: Intel Corporation Core Processor QuickPath Architecture Generic Non-core Registers (rev 05)
ff:00.1 Host bridge: Intel Corporation Core Processor QuickPath Architecture System Address Decoder (rev 05)
ff:02.0 Host bridge: Intel Corporation Core Processor QPI Link 0 (rev 05)
ff:02.1 Host bridge: Intel Corporation 1st Generation Core i3/5/7 Processor QPI Physical 0 (rev 05)
ff:02.2 Host bridge: Intel Corporation 1st Generation Core i3/5/7 Processor Reserved (rev 05)
ff:02.3 Host bridge: Intel Corporation 1st Generation Core i3/5/7 Processor Reserved (rev 05)
```

Figura 15.4: Obtendo Informações sobre as Dispositivos PCI

15.6 Listando Todos os Dispositivos de Bloco

O comando **lsblk** lista todos os dispositivos de bloco de seu computador. A Figura 15.5 apresenta o resultado do comando. Neste comando aparecerá dois dispositivos de bloco */dev/sda* e */dev/sdb* pois o dispositivo *sbd* é um HD Externo conectado ao computador.

```
[root@musashi ~]# lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda        8:0    0 465,8G  0 disk
└─sda1     8:1    0 117,6M  0 part
└─sda2     8:2    0   9,8G  0 part
└─sda3     8:3    0 249,9G  0 part
└─sda4     8:4    0    1K  0 part
└─sda5     8:5    0   500M  0 part /boot
└─sda6     8:6    0 205,5G  0 part
    ├─fedora-swap 253:0 0   3,8G  0 lvm  [SWAP]
    ├─fedora-root 253:1 0   50G  0 lvm /
    └─fedora-home 253:2 0 151,7G  0 lvm /home
sdb        8:16   0 465,8G  0 disk
└─sdb1     8:17   0  200M  0 part
└─sdb2     8:18   0 465,5G  0 part
sr0       11:0   1 1024M  0 rom
```

Figura 15.5: Obtendo Informações sobre as Dispositivos de Bloco

15.7 Verificando Todas as Partições

Com o comando **cat** é possível visualizar as informações sobre as partições em seu disco. A Listagem 15.4 apresenta o resultado do comando.

Listagem 15.4: Imprimindo as Partições

```
[root@musashi ~]# cat /proc/partitions
major minor #blocks name

8          0   488386584 sda
8          1    120456 sda1
8          2   10258432 sda2
8          3   262023268 sda3
8          4        1 sda4
8          5   512000 sda5
8          6  215469056 sda6
11         0   1048575 sr0
253        0   3932160 dm-0
253        1   52428800 dm-1
253        2   159100928 dm-2
8          16  488386584 sdb
8          17   204800 sdb1
8          18  488050672 sdb2
```

15.8 Listando Dispositivos PCMCIA

O comando **lspcmcia** lista todos os dispositivos PCMCIA em seu computador. A Listagem 15.5 apresenta a sintaxe do comando. Neste caso não há dispositivos PCMCIA no computador.

Listagem 15.5: Listando Dispositivos PCMCIA

```
[root@musashi ~]# lspcmcia
```

15.9 Obtendo Informações sobre a Memória

O comando **free** lista a quantidade de memória disponível em seu computador. A Listagem 15.6 apresenta o resultado do comando.

Listagem 15.6: Listando Informações sobre a Memória

```
[root@musashi ~]# free
              total        used        free      shared  buff/cache   available
Mem:       3843788     866308    1901028        4456    1076452     2721300
Swap:      3932156           0    3932156
```

15.10 Listando Todos os Dispositivos de Hardware

O comando **hwinfo** é utilizado para listar informações sobre todos os dispositivos de Hardware de seu computador. O **hwinfo** não está disponível para o Fedora. Neste caso utilize o comando **lshw**. A Figura 15.7 apresenta o resultado do comando.

Se o comando não estiver disponível realize o procedimento de instalação que deverá ser específico para sua distribuição.

Listagem 15.7: Analisando os Dispositivos de Hardware

```
musashi.vivascorp
  description: Portable Computer
  product: Vostro 3500 (To be filled by O.E.M.)
  vendor: Dell Inc.
  version: Not Specified
  serial: D3HQ4PI
  width: 64 bits
  capabilities: smbios-2.6 dmi-2.6 vsyscall32
  configuration: boot=normal chassis=portable sku=To be filled by O.E.M. uuid=44454C4C-3300-1048-8051-C4C04F345031
*-core
  description: Motherboard
  product: 0PXM4R
  vendor: Dell Inc.
  physical id: 0
  version: A00
  serial: .D3HQ4PI.CN701660860ILT.
  slot: To Be Filled By O.E.M.
*-cpu
  description: CPU
  product: (To Be Filled By O.E.M.)
  vendor: Intel Corp.
  physical id: 4
  bus info: cpu@0
  version: Intel(R) Core(TM) i5 CPU      M 460 @ 2.53GHz
  serial: To Be Filled By O.E.M.
  slot: CPU 1
  size: 2528MHz
  capacity: 2793MHz
  width: 64 bits
  clock: 533MHz
  capabilities: x86_64 fpu fpu_exception wp vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts
    acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx rdtscp constant_tsc arch_perfmon pebs bts rep_good nopl xttopology
    nonstop_tsc aperfmpf perf pn1 dtes64 monitor ds_cpl vmx est tm2 ssse3 cx16 xtpr pdem pcid sse4_1 sse4_2 popcnt lahf_lm
    ida arat dtherm tpr_shadow vnmi flexpriority ept vpid cpufreq
  configuration: cores=2 enabledcores=1 threads=2
*-cache:0
  description: L1 cache
  physical id: 5
  slot: L1-Cache
  size: 64KiB
  capacity: 64KiB
  capabilities: internal write-back unified
*-cache:1
  description: L2 cache
  physical id: 6
  slot: L2-Cache
  size: 512KiB
  capacity: 512KiB
  capabilities: internal varies unified
*-cache:2
  description: L3 cache
  physical id: 7
  slot: L3-Cache
  size: 3MiB
  capacity: 3MiB
  capabilities: internal varies unified
*-memory
  description: System Memory
  physical id: 1d
  slot: System board or motherboard
  size: 4GiB
*-bank:0
  description: DIMM DDR3 Synchronous 1333 MHz (0,8 ns)
  product: M471B5673FH0-CH9
  vendor: Undefined
  physical id: 0
  serial: 05F81D2B
  slot: A1_DIMM0
  size: 2GiB
  width: 64 bits
  clock: 1333MHz (0.8 ns)
*-bank:1
  description: DIMM [empty]
  physical id: 1
  slot: None
  width: 64 bits
*-bank:2
  description: DIMM DDR3 Synchronous 1333 MHz (0,8 ns)
  product: M471B5673FH0-CH9
  vendor: Undefined
  physical id: 2
  serial: 05F81E2E
  slot: A1_DIMM2
  size: 2GiB
  width: 64 bits
  clock: 1333MHz (0.8 ns)
*-bank:3
  description: DIMM [empty]
  physical id: 3
```

```

slot: None
width: 64 bits
*--firmware
    description: BIOS
    vendor: Dell Inc.
    physical id: 0
    version: A08
    date: 09/10/2010
    size: 64KiB
    capacity: 1984KiB
    capabilities: mca pci upgrade shadowing escd cdboot bootselect socketedrom edd int13floppy1200 int13floppy720
                  int13floppy2880 int5printscreens int9keyboard int14serial int17printer int10video acpi usb zipboot
                  biosbootspecification
*--pci:0
    description: Host bridge
    product: Core Processor DRAM Controller
    vendor: Intel Corporation
    physical id: 100
    bus info: pci@0000:00:00.0
    version: 18
    width: 32 bits
    clock: 33MHz
    configuration: driver=agpgart-intel
    resources: irq:0
*--pci:0
    description: PCI bridge
    product: Core Processor PCI Express x16 Root Port
    vendor: Intel Corporation
    physical id: 1
    bus info: pci@0000:00:01.0
    version: 18
    width: 32 bits
    clock: 33MHz
    capabilities: pci pm msi pciexpress normal_decode bus_master cap_list
    configuration: driver=pcieport
    resources: irq:24 ioport:e000(size=4096) memory:f9000000-fa0fffff ioport:c0000000(size=301989888)
*--display UNCLAIMED
    description: VGA compatible controller
    product: GT218M [GeForce 310M]
    vendor: NVIDIA Corporation
    physical id: 0
    bus info: pci@0000:01:00.0
    version: a2
    width: 64 bits
    clock: 33MHz
    capabilities: pm msi pciexpress vga_controller cap_list
    configuration: latency=0
    resources: memory:f9000000-f9ffffff memory:c0000000-cfffffff memory:d0000000-d1ffffff ioport:e000(size=128) memory:
                  fa000000-fa07ffff
*--multimedia
    description: Audio device
    product: High Definition Audio Controller
    vendor: NVIDIA Corporation
    physical id: 0.1
    bus info: pci@0000:01:00.1
    version: a1
    width: 32 bits
    clock: 33MHz
    capabilities: pm msi pciexpress bus_master cap_list
    configuration: driver=snd_hda_intel latency=0
    resources: irq:16 memory:fa080000-fa083fff
*--display UNCLAIMED
    description: VGA compatible controller
    product: Core Processor Integrated Graphics Controller
    vendor: Intel Corporation
    physical id: 2
    bus info: pci@0000:00:02.0
    version: 18
    width: 64 bits
    clock: 33MHz
    capabilities: msi pm vga_controller bus_master cap_list
    configuration: latency=0
    resources: memory:fa400000-fa7fffff memory:b0000000-bfffffff ioport:f080(size=8)
*--communication
    description: Communication controller
    product: 5 Series/3400 Series Chipset HECI Controller
    vendor: Intel Corporation
    physical id: 16
    bus info: pci@0000:00:16.0
    version: 06
    width: 64 bits
    clock: 33MHz
    capabilities: pm msi bus_master cap_list
    configuration: driver=mei_me latency=0
    resources: irq:31 memory:fb409000-fb40900f
*--usb:0
    description: USB controller
    product: 5 Series/3400 Series Chipset USB2 Enhanced Host Controller
    vendor: Intel Corporation
    physical id: 1a
    bus info: pci@0000:00:1a.0
    version: 06
    width: 32 bits
    clock: 33MHz
    capabilities: pm debug ehci bus_master cap_list
    configuration: driver=ehci-pci latency=0

```

```

resources: irq:16 memory:fb408000-fb4083ff
*-usbhost
    product: EHCI Host Controller
    vendor: Linux 4.0.4-301.fc22.x86_64 ehci_hcd
    physical id: 1
    bus info: usb@1
    logical name: usb1
    version: 4.00
    capabilities: usb-2.00
    configuration: driver=hub slots=2 speed=480Mbit/s
*-usb
    description: USB hub
    product: Integrated Rate Matching Hub
    vendor: Intel Corp.
    physical id: 1
    bus info: usb@1:1
    version: 0.00
    capabilities: usb-2.00
    configuration: driver=hub slots=6 speed=480Mbit/s
*-usb:0
    description: USB hub
    product: BCM2046BI
    vendor: Broadcom
    physical id: 1
    bus info: usb@1:1.1
    version: 1.00
    capabilities: usb-2.00
    configuration: driver=hub maxpower=94mA slots=3 speed=12Mbit/s
*-usb:0
    description: Keyboard
    product: Integrated Keyboard
    vendor: Dell Computer Corp.
    physical id: 1
    bus info: usb@1:1.1.1
    version: 1.00
    capabilities: usb-2.00
    configuration: driver=usbhid maxpower=2mA speed=12Mbit/s
*-usb:1
    description: Mouse
    product: Integrated Touchpad [Synaptics]
    vendor: Dell Computer Corp.
    physical id: 2
    bus info: usb@1:1.1.2
    version: 1.00
    capabilities: usb-2.00
    configuration: driver=usbhid maxpower=2mA speed=12Mbit/s
*-usb:1 UNCLAIMED
    description: Generic USB device
    product: VFS300 Fingerprint Reader
    vendor: Validity Sensors, Inc.
    physical id: 4
    bus info: usb@1:1.4
    version: c.90
    serial: 3711dad5ed00
    capabilities: usb-1.10
    configuration: maxpower=100mA speed=12Mbit/s
*-multimedia
    description: Audio device
    product: 5 Series/3400 Series Chipset High Definition Audio
    vendor: Intel Corporation
    physical id: 1b
    bus info: pci@0000:00:1b.0
    version: 06
    width: 64 bits
    clock: 33MHz
    capabilities: pm msi pciexpress bus_master cap_list
    configuration: driver=snd_hda_intel latency=0
    resources: irq:32 memory:fb400000-fb403fff
*-pci:1
    description: PCI bridge
    product: 5 Series/3400 Series Chipset PCI Express Root Port 1
    vendor: Intel Corporation
    physical id: 1c
    bus info: pci@0000:00:1c.0
    version: 06
    width: 32 bits
    clock: 33MHz
    capabilities: pci pciexpress msi pm normal_decode bus_master cap_list
    configuration: driver=pcieport
    resources: irq:25 ioport:2000(size=4096) memory:d2d00000-d2efffff ioport:d2f00000(size=2097152)
*-pci:2
    description: PCI bridge
    product: 5 Series/3400 Series Chipset PCI Express Root Port 2
    vendor: Intel Corporation
    physical id: 1c.1
    bus info: pci@0000:00:1c.1
    version: 06
    width: 32 bits
    clock: 33MHz
    capabilities: pci pciexpress msi pm normal_decode bus_master cap_list
    configuration: driver=pcieport
    resources: irq:26 ioport:3000(size=4096) memory:fb300000-fb3fffff ioport:d3100000(size=2097152)
*-network
    description: Network controller
    product: BCM4313 802.11bgn Wireless Network Adapter
    vendor: Broadcom Corporation

```

```

physical id: 0
bus info: pci@0000:12:00.0
version: 01
width: 64 bits
clock: 33MHz
capabilities: pm msi pciexpress bus_master cap_list
configuration: driver=bcm-a-pci-bridge latency=0
resources: irq:17 memory:fb300000-fb303fff

*-pci:3
    description: PCI bridge
    product: 5 Series/3400 Series Chipset PCI Express Root Port 3
    vendor: Intel Corporation
    physical id: 1c.2
    bus info: pci@0000:00:1c.2
    version: 06
    width: 32 bits
    clock: 33MHz
    capabilities: pci pciexpress msi pm normal_decode bus_master cap_list
    configuration: driver=pcieport
    resources: irq:27 ioport:d000(size=4096) memory:fb200000-fb2fffff ioport:d2c00000(size=1048576)

*-network
    description: Ethernet interface
    product: RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller
    vendor: Realtek Semiconductor Co., Ltd.
    physical id: 0
    bus info: pci@0000:13:00.0
    logical name: enp19s0
    version: 03
    serial: f0:4d:a2:9d:3f:3f
    size: 10Mbit/s
    capacity: 1Gbit/s
    width: 64 bits
    clock: 33MHz
    capabilities: pm msi pciexpress msix vpd bus_master cap_list rom ethernet physical tp mii 10bt 100bt 100bt-fd 1000bt 1000bt-fd autonegotiation
    configuration: autonegotiation=on broadcast=yes driver=r8169 driverversion=2.3LK-NAPI duplex=half firmware=rtl_nic /rtl8168d-2.fw latency=0 link=no multicast=yes port=MII speed=10Mbit/s
    resources: irq:30 ioport:d000(size=256) memory:d2c04000-d2c04fff memory:fb200000-fb21ffff

*-pci:4
    description: PCI bridge
    product: 5 Series/3400 Series Chipset PCI Express Root Port 5
    vendor: Intel Corporation
    physical id: 1c.4
    bus info: pci@0000:00:1c.4
    version: 06
    width: 32 bits
    clock: 33MHz
    capabilities: pci pciexpress msi pm normal_decode bus_master cap_list
    configuration: driver=pcieport
    resources: irq:28 ioport:c000(size=4096) memory:fa800000-fb1fffff ioport:d2100000(size=10485760)

*-usb:1
    description: USB controller
    product: 5 Series/3400 Series Chipset USB2 Enhanced Host Controller
    vendor: Intel Corporation
    physical id: 1d
    bus info: pci@0000:00:1d.0
    version: 06
    width: 32 bits
    clock: 33MHz
    capabilities: pm debug ehci bus_master cap_list
    configuration: driver=ehci-pci latency=0
    resources: irq:23 memory:fb407000-fb4073ff

*-usbhost
    product: EHCI Host Controller
    vendor: Linux 4.0.4-301.fc22.x86_64 ehci_hcd
    physical id: 1
    bus info: usb@2
    logical name: usb2
    version: 4.00
    capabilities: usb-2.00
    configuration: driver=hub slots=2 speed=480Mbit/s

*-usb
    description: USB hub
    product: Integrated Rate Matching Hub
    vendor: Intel Corp.
    physical id: 1
    bus info: usb@2:1
    version: 0.00
    capabilities: usb-2.00
    configuration: driver=hub slots=8 speed=480Mbit/s

*-usb:0
    description: Mass storage device
    product: Samsung S2 Portable
    vendor: JMicron
    physical id: 3
    bus info: usb@2:1.3
    logical name: scsi6
    version: 0.00
    serial: 0000002CE3009
    capabilities: usb-2.00 scsi emulated scsi-host
    configuration: driver=usb-storage maxpower=2mA speed=480Mbit/s

*-disk
    description: SCSI Disk
    physical id: 0.0.0
    bus info: scsi@6:0.0.0
    logical name: /dev/sdb

```

```

size: 465GiB (500GB)
capabilities: gpt-1.00 partitioned partitioned:gpt
configuration: guid=03d8f472-1b09-45d2-8667-3d232a61eeel logicalsectorsize=512 sectorsize=512
*--volume:0
    description: Windows FAT volume
    vendor: BSD 4.4
    physical id: 1
    bus info: scsi@6:0.0.0,1
    logical name: /dev/sdb1
    version: FAT32
    serial: 67e3-17ed
    size: 199MiB
    capacity: 199MiB
    capabilities: boot fat initialized
    configuration: FATS=2 filesystem=fat label=EFI name=EFI System Partition
*--volume:1
    description: Apple HFS+ partition
    vendor: Mac OS X (journaled)
    physical id: 2
    bus info: scsi@6:0.0.0,2
    logical name: /dev/sdb2
    version: 4
    serial: 32b79db1-d1ab-a8d7-0000-000000e8c000
    size: 465GiB
    capabilities: journaled hfsplus initialized
    configuration: checked=2014-10-05 07:02:32 created=2014-10-05 07:02:32 filesystem=hfsplus lastmountedby=
HFSJ modified=2015-08-30 06:36:27 name=Sem Título state=unclean
*--usb:1
    description: Video
    product: Laptop_Integrated_Webcam_2M
    vendor: CKF9237G274030003FG0
    physical id: 8
    bus info: usb@2:1.8
    version: 2.01
    capabilities: usb-2.00
    configuration: driver=uvcvideo maxpower=168mA speed=480Mbit/s
*--pci:5
    description: PCI bridge
    product: 82801 Mobile PCI Bridge
    vendor: Intel Corporation
    physical id: 1e
    bus info: pci@0000:00:1e.0
    version: a6
    width: 32 bits
    clock: 33MHz
    capabilities: pci subtractive_decode bus_master cap_list
*--isa
    description: ISA bridge
    product: Mobile 5 Series Chipset LPC Interface Controller
    vendor: Intel Corporation
    physical id: 1f
    bus info: pci@0000:00:1f.0
    version: 06
    width: 32 bits
    clock: 33MHz
    capabilities: isa bus_master cap_list
    configuration: driver=lpc_ich latency=0
    resources: irq:0
*--storage
    description: SATA controller
    product: 5 Series/3400 Series Chipset 6 port SATA AHCI Controller
    vendor: Intel Corporation
    physical id: 1f.2
    bus info: pci@0000:00:1f.2
    version: 06
    width: 32 bits
    clock: 66MHz
    capabilities: storage msi pm ahci_1.0 bus_master cap_list
    configuration: driver=ahci latency=0
    resources: irq:29 ioport:f070(size=8) ioport:f060(size=4) ioport:f050(size=8) ioport:f040(size=4) ioport:f020(size=32)
               memory:fb406000-fb4067ff
*--serial UNCLAIMED
    description: SMBus
    product: 5 Series/3400 Series Chipset SMBus Controller
    vendor: Intel Corporation
    physical id: 1f.3
    bus info: pci@0000:00:1f.3
    version: 06
    width: 64 bits
    clock: 33MHz
    configuration: latency=0
    resources: memory:fb405000-fb4050ff ioport:f000(size=32)
*--generic
    description: Signal processing controller
    product: 5 Series/3400 Series Chipset Thermal Subsystem
    vendor: Intel Corporation
    physical id: 1f.6
    bus info: pci@0000:00:1f.6
    version: 06
    width: 64 bits
    clock: 33MHz
    capabilities: pm msi bus_master cap_list
    configuration: driver=intel ips latency=0
    resources: irq:18 memory:fb404000-fb404fff
*--pci:1
    description: Host bridge

```

```

product: Core Processor QuickPath Architecture Generic Non-core Registers
vendor: Intel Corporation
physical id: 101
bus info: pci@0000:ff:00.0
version: 05
width: 32 bits
clock: 33MHz
*-pci:2
    description: Host bridge
    product: Core Processor QuickPath Architecture System Address Decoder
    vendor: Intel Corporation
    physical id: 102
    bus info: pci@0000:ff:00.1
    version: 05
    width: 32 bits
    clock: 33MHz
*-pci:3
    description: Host bridge
    product: Core Processor QPI Link 0
    vendor: Intel Corporation
    physical id: 103
    bus info: pci@0000:ff:02.0
    version: 05
    width: 32 bits
    clock: 33MHz
*-pci:4
    description: Host bridge
    product: 1st Generation Core i3/5/7 Processor QPI Physical 0
    vendor: Intel Corporation
    physical id: 104
    bus info: pci@0000:ff:02.1
    version: 05
    width: 32 bits
    clock: 33MHz
*-pci:5
    description: Host bridge
    product: 1st Generation Core i3/5/7 Processor Reserved
    vendor: Intel Corporation
    physical id: 105
    bus info: pci@0000:ff:02.2
    version: 05
    width: 32 bits
    clock: 33MHz
*-pci:6
    description: Host bridge
    product: 1st Generation Core i3/5/7 Processor Reserved
    vendor: Intel Corporation
    physical id: 106
    bus info: pci@0000:ff:02.3
    version: 05
    width: 32 bits
    clock: 33MHz
*-scsi:0
    physical id: 1
    logical name: scsi0
    capabilities: emulated
    *--disk
        description: ATA Disk
        product: ST9500420AS
        vendor: Seagate
        physical id: 0.0.0
        bus info: scsi@0:0.0.0
        logical name: /dev/sda
        version: SDMI
        serial: 5VJ8PR42
        size: 465GiB (500GB)
        capabilities: partitioned partitioned:dos
        configuration: ansiversion=5 logicalssectorsize=512 sectorsize=512 signature=50000000
    *--volume:0
        description: Windows FAT volume
        vendor: Dell 8.0
        physical id: 1
        bus info: scsi@0:0.0.0,1
        logical name: /dev/sdai
        version: FAT16
        serial: 07da-0b0b
        size: 117MiB
        capacity: 117MiB
        capabilities: primary fat initialized
        configuration: FATS=2 filesystem=fat label=DellUtility
    *--volume:1
        description: Windows NTFS volume
        physical id: 2
        bus info: scsi@0:0.0.0,2
        logical name: /dev/sda2
        version: 3.1
        serial: 3e7b-bf98
        size: 9995MiB
        capacity: 10018MiB
        capabilities: primary bootable ntfs initialized
        configuration: clustersize=4096 created=2010-11-11 18:39:03 filesystem=ntfs label=RECOVERY state=clean
    *--volume:2
        description: Windows NTFS volume
        physical id: 3
        bus info: scsi@0:0.0.0,3
        logical name: /dev/sda3

```

```

version: 3.1
serial: bc498c3f-c682-024c-9c12-013707bd9e90
size: 249GiB
capacity: 249GiB
capabilities: primary ntfs initialized
configuration: clustersize=4096 created=2010-11-11 18:39:08 filesystem=ntfs label=OS modified_by_chkdsk=true
mounted_on_nt4=true resize_log_file=true state=dirty upgrade_on_mount=true
*--volume:3
    description: Extended partition
    physical id: 4
    bus info: scsi@0:0.0.0.4
    logical name: /dev/sda4
    size: 205GiB
    capacity: 205GiB
    capabilities: primary extended partitioned partitioned:extended
*--logicalvolume:0
    description: Linux filesystem partition
    physical id: 5
    logical name: /dev/sda5
    logical name: /boot
    capacity: 500MiB
    configuration: mount.fstype=ext4 mount.options=rw,seclabel,relatime,data=ordered state=mounted
*--logicalvolume:1
    description: Linux LVM Physical Volume partition
    physical id: 6
    logical name: /dev/sda6
    serial: 71Jkoy-At38-Vebn-Gcam-iYC6-ye2s-FeZTpD
    size: 205GiB
    capacity: 205GiB
    capabilities: multi lvm2
*--scsi:1
    physical id: 2
    logical name: scsil
    capabilities: emulated
*--cdrom
    description: DVD-RAM writer
    product: DVD+RW GT32N
    vendor: HL-DT-ST
    physical id: 0.0.0
    bus info: scsi@1:0.0.0
    logical name: /dev/cdrom
    logical name: /dev/sr0
    version: A200
    capabilities: removable audio cd-r cd-rw dvd dvd-r dvd-ram
    configuration: ansiversion=5 status=nodisc
*--battery
    description: Lithium Ion Battery
    product: DELL TXWR08
    vendor: SMP
    physical id: 1
    version: 10/31/2006
    serial: 3387
    slot: Sys. Battery Bay
    capacity: 52000mWh
    configuration: voltage=11,1V
*--network
    description: Wireless interface
    physical id: 2
    logical name: wlp18s0b1
    serial: 1c:65:9d:82:e2:cd
    capabilities: ethernet physical wireless
    configuration: broadcast=yes driver=brcmsmac driverversion=4.0.4-301.fc22.x86_64 firmware=610.812 ip=192.168.0.104 link=yes
    multicast=yes wireless=IEEE 802.11bgn

```

Se quiser uma versão resumida do relatório utilize a Listagem 15.8.

Listagem 15.8: Versão Resumida do Relatório do lshw

H/W path	Device	Class	Description
/0		system	Vostro 3500 (To be filled by O.E.M.)
/0/4		bus	OPXM4R
/0/4/5		processor	(To Be Filled By O.E.M.)
/0/4/6		memory	64KiB L1 cache
/0/4/7		memory	512KiB L2 cache
/0/1d		memory	3MiB L3 cache
/0/1d/0		memory	4GiB System Memory
/0/1d/1		memory	2GiB DIMM DDR3 Synchronous 1333 MHz (0,8 ns)
/0/1d/2		memory	DIMM [empty]
/0/1d/3		memory	2GiB DIMM DDR3 Synchronous 1333 MHz (0,8 ns)
/0/0		memory	DIMM [empty]
/0/100		bridge	64KiB BIOS
/0/100/1		bridge	Core Processor DRAM Controller
/0/100/1/0		bridge	Core Processor PCI Express x16 Root Port
/0/100/1/0.1		display	GT218M [GeForce 310M]
/0/100/1/0.1		multimedia	High Definition Audio Controller
/0/100/2		display	Core Processor Integrated Graphics Controller
/0/100/16		communication	5 Series/3400 Series Chipset HECI Controller
/0/100/1/a		bus	5 Series/3400 Series Chipset USB2 Enhanced Host Controller
/0/100/1/a/1	usb1	bus	EHCI Host Controller
/0/100/1/a/1/1		bus	Integrated Rate Matching Hub
/0/100/1/a/1/1/1		bus	BCM2046B1
/0/100/1/a/1/1/1/1		input	Integrated Keyboard

/0/100/1a/1/1/1/2	input	Integrated Touchpad [Synaptics]
/0/100/1a/1/1/4	generic	VFS300 Fingerprint Reader
/0/100/1b	multimedia	5 Series/3400 Series Chipset High Definition Audio
/0/100/1c	bridge	5 Series/3400 Series Chipset PCI Express Root Port 1
/0/100/1c.1	bridge	5 Series/3400 Series Chipset PCI Express Root Port 2
/0/100/1c.1.0	network	BCM4313 802.11bgn Wireless Network Adapter
/0/100/1c.2	bridge	5 Series/3400 Series Chipset PCI Express Root Port 3
/0/100/1c.2.0	enp19s0	RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller
/0/100/1c.4	bridge	5 Series/3400 Series Chipset PCI Express Root Port 5
/0/100/1d	bus	5 Series/3400 Series Chipset USB2 Enhanced Host Controller
/0/100/1d/1	usb2	EHCI Host Controller
/0/100/1d/1/1	bus	Integrated Rate Matching Hub
/0/100/1d/1/1/3	scsi6	Samsung S2 Portable
/0/100/1d/1/1/3/0.0.0	/dev/sdb	500GB SCSI Disk
/0/100/1d/1/1/3/0.0.0/1	/dev/sdb1	199MiB Windows FAT volume
/0/100/1d/1/1/3/0.0.0/2	/dev/sdb2	465GiB Apple HFS+ partition
/0/100/1d/1/1/8	multimedia	Laptop_Integrated_Webcam_2M
/0/100/1e	bridge	82801 Mobile PCI Bridge
/0/100/1f	bridge	Mobile 5 Series Chipset LPC Interface Controller
/0/100/1f.2	storage	5 Series/3400 Series Chipset 6 port SATA AHCI Controller
/0/100/1f.3	bus	5 Series/3400 Series Chipset SMBus Controller
/0/100/1f.6	generic	5 Series/3400 Series Chipset Thermal Subsystem
/0/101	bridge	Core Processor QuickPath Architecture Generic Non-core Register
/0/102	bridge	Core Processor QuickPath Architecture System Address Decoder
/0/103	bridge	Core Processor QPI Link 0
/0/104	bridge	1st Generation Core i3/5/7 Processor QPI Physical 0
/0/105	bridge	1st Generation Core i3/5/7 Processor Reserved
/0/106	bridge	1st Generation Core i3/5/7 Processor Reserved
/0/1	scsi0	
/0/1/0.0.0	/dev/sda	storage
/0/1/0.0.0/1	/dev/sda1	disk
/0/1/0.0.0/2	/dev/sda2	117MiB Windows FAT volume
/0/1/0.0.0/3	/dev/sda3	10018MiB Windows NTFS volume
/0/1/0.0.0/4	/dev/sda4	249GiB Windows NTFS volume
/0/1/0.0.0/4/5	/dev/sda5	205GiB Extended partition
/0/1/0.0.0/4/6	/dev/sda6	500MiB Linux filesystem partition
/0/2	scsil	205GiB Linux LVM Physical Volume partition
/0/2/0.0.0	/dev/cdrom	DVD+-RW GT32N
/1	power	DELL TXWR08
/2	wlp18s0b1	network
		Wireless interface

Índice

Análise de Desempenho

- iostat, 100
- mpstat, 99
- pidstat, 100
- sar, 99
- top, 102

Aplicativos

- Virtual Box, 1

Boot

- boot, 1
- grub, 1
- lilo, 1

Comandos Úteis

- unit, 89
- yes, 89

Comandos de Manipulação de Arquivos e Diretórios

- cat, 4
- cd, 18
- cp, 19
- echo, 6
- mkdir, 22
- pwd, 19
- rename, 20
- rm, 21
- rmdir, 21
- touch, 21

Comandos de Processamento de Texto

- cat, 38, 65, 66, 69
- cut, 27
- echo, 4
- expand, 28
- fmt, 30
- grep, 30
- head, 30
- iconv, 31
- look, 31
- more, 32
- nl, 32
- paste, 33
- rev, 33

- sort, 33

- tail, 34
- uniq, 35
- wc, 35

Comandos de Redes

- arp, 69
- dig, 74
- host, 74
- hostname, 69
- ifconfig, 70–72
- lynx, 85
- netstat, 77
- nmap, 78
- nslookup, 75
- ping, 73
- route, 81
- scp, 83
- ssh, 82
- tcpdump, 84
- telnet, 81
- tracepath, 76
- traceroute, 75
- wget, 85

Comandos de Sistema

- cal, 40
- clear, 3
- compgen, 37
- date, 40
- exit, 9
- finger, 40
- free, 41
- history, 7
- HISTSIZE, 8
- id, 39
- ln, 23
- locate, 45
- login, 2
- logout, 9
- ls, 5, 14
- passwd, 2, 39
- pipe, 30

poweroff, 10
pwd, 37
reboot, 11
shutdown, 10, 11
su, 42
uname, 42
uptime, 43
users, 40
w, 44
whatis, 46
whereis, 45
which, 45
whoami, 37

Comandos de Sistemas de Arquivos
fdisk, 96
lsblk, 96, 97
parted, 95
sfdisk, 96

Comandos Divertidos
cowsay, 91
fortune, 92
sl, 92
xcoffortune, 92
xcowsay, 91
xyes, 93

Distribuições
Debian, 1
Fedora, 1
OpenSuse, 1
Ubuntu, 1

Gerenciamento de Pacotes
dnf, 88

Gerenciamento de Processos
kill, 52
ps, 49
time, 52
top, 51

Gerenciamento de Usuários e Grupos
groupadd, 67
groupdel, 67
groupmod, 67
passwd, 66
useradd, 66
userdel, 66
usermod, 67

Hardware
lshw, 108

Hardware e Software, 108

hwinfo, 108
lsblk, 107
lscpu, 105
lspci, 107
lspcmcia, 108
lsusb, 106
uname, 105

Shells
bash, 4
csh, 4
ksh, 4
sh, 4
tcsh, 4
zsh, 4

Bibliografia

[1] Canonical Ltda. Ubuntu, 2015.

[2] Debian. Debian, 2015.

[3] Fedora. Fedora, 2015.

[4] OpenSuse. OpenSuse, 2015.

[5] VirtualBox. Virtual Box, 2015.